PENGEMBANGAN SISTEM APLIKASI MANAJEMEN DISTRIBUSI PUPUK BERBASIS WEB

(Studi Kasus: PT Petrokimia Gresik)

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Moh. Wahyu Dwi Ridiansyah
NIM: 155150200111232

PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PENGESAHAN

PENGEMBANGAN SISTEM APLIKASI MANAJEMEN DISTRIBUSI PUPUK BERBASIS WEB
(STUDI KASUS: PT PETROKIMIA GRESIK)

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Moh. Wahyu Dwi Ridiansyah
NIM: 155150200111232

Skripsi ini telah diuji dan dinyatakan lulus pada
17 Juni 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing 1	Dosen Pembimbing 2
Fajar Pradana, S.ST, M.Eng	Nurudin Santoso, S.T, M.T
NIP: 19871121 201504 1 004	NIP: 19740916 200012 1 001

Mengetahui
Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D
NIP: 19710518 200312 1 001

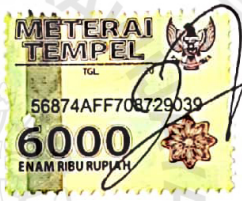
repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar referensi.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 10 Mei 2019



Moh. Wahyu Dwi Ridiansyah

NIM: 155150200111232

PRAKATA

Assalamu'alaikum Warrahmatullahi Wabarakatuh. Alhamdulillahi Rabbil'Alamin. Puji Syukur penulis panjatkan kehadirat Allah SWT, karena atas berkah, rahmat, serta hidayah-Nya penulis dapat menyelesaikan Skripsi yang berjudul "Pengembangan Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web (Studi Kasus: PT Petrokimia Gresik)" dengan baik.

Pada kesempatan ini, penulis menyampaikan rasa terima kasih kepada pihak-pihak yang telah membantu penulis selama proses penyusunan skripsi, diantaranya adalah:

1. Allah SWT yang telah memberi kemudahan dalam semua proses penulisan skripsi ini.
2. Kepada Ibu saya yang selalu mendukung saya serta tidak henti-hetinnya memberikan doa kepada saya.
3. Bapak Fajar Pradana, S.ST, M.Eng selaku dosen pembimbing I, yang telah meluangkan waktu untuk memberikan ilmu serta saran dalam proses penyusunan skripsi ini.
4. Bapak Nurudin Santoso, S.T., M.T selaku dosen pembimbing II, yang juga telah meluangkan waktu untuk memberikan ilmu serta saran dalam proses penyusunan skripsi ini.
5. Mas Yafi Anshori selaku Pengawai Departemen Teknologi Informasi PT Petrokimia Gresik yang telah mendukung serta memberikan ilmu yang berharga dalam proses penyusunan skripsi ini.
6. Mas Edwyk Sony Edaieby dan Mas Rachmad Edy Rafianto selaku Pegawai Departemen Distribusi Wilayah 1 PT Petrokimia Gresik yang telah mendukung serta memberikan ilmunya dalam proses penyusunan skripsi ini.
7. Saudara Nugroho Samiyono yang telah memberikan ilmunya dalam proses penyusunan skripsi ini.
8. Saudari Nurud Diana Syafa'ati yang selalu mendukung dan mendoakan saya dalam proses penyusunan skripsi ini.
9. Teman-teman Anug, Fahmi, Anita, dan Ajeng yang selalu memberikan semangat dan motivasi selama mengerjakan skripsi ini.
10. Teman-teman Buatin Nugik, Ubed, Rayyan dan Villa yang selalu memberikan dukungan dan semangat selama mengerjakan skripsi ini.
11. Seluruh teman-teman Fakultas Ilmu Komputer yang telah memberikan bantuan serta ilmu yang berharga, dan seluruh dukungannya selama proses penyusunan skripsi ini.
12. Seluruh pihak yang tidak dapat disebutkan satu per satu dan telah membantu selama proses penyusunan skripsi ini.

Semoga seluruh jasa dan bantuannya mendapatkan balasan dari Allah SWT. Penulis menyadari dalam penyusunan skripsi ini masih banyak ditemukan kekurangan, semoga skripsi ini dapat bermanfaat baik untuk pembaca dan

pengembangan penelitian selanjutnya. Oleh karena itu, segala bentuk saran dan kritikan yang bersifat membangun skripsi ini dapat disampaikan melalui email penulis.

Malang, 10 Mei 2019

Penulis

wahyuridiansyah@gmail.com

ABSTRAK

Moh. Wahyu Dwi Ridiansyah, Pengembangan Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web (Studi Kasus: PT Petrokimia Gresik).

Pembimbing: Fajar Pradana, S.ST, M.Eng dan Nurudin Santoso, S.T, M.T

PT Petrokimia Gresik Merupakan Perusahaan Badan Usaha Milik Negara yang memproduksi pupuk subsidi maupun non-subsidi. Dalam kegiatan distribusi ditemukan beberapa masalah. Permasalahan pertama dirasakan oleh Staf Departemen Diswil 1 mengalami kendala dalam mengolah data pelaporan dari gudang penyangga karena banyaknya data yang ada. Permasalahan kedua dirasakan oleh Manajer Diswil 1 yang mengeluh kesulitan memantau keadaan stok di daerah secara cepat dan efisien. Permasalahan ketiga dikeluhkan oleh Staf Departemen Diswil 1 mengenai pelaporan pupuk rusak di Gudang Penyangga, proses yang terjadi sekarang tidak efisien dan belum adanya media yang dapat memantau pengiriman pupuk rusak ke gudang pusat untuk didaur ulang. Permasalahan keempat dikeluhkan oleh staf bagian stok opname, mengeluh proses yang tidak efisien karena hasil stok opname harus dilakukan penyesuaian secara manual pada pencatatan stok. Berdasarkan permasalahan tersebut, maka dilakukan pengembang sistem pada penelitian ini yang dapat menyelesaikan permasalah tersebut. Metode *waterfall* digunakan dalam proses pengembangan perangkat lunak pada penelitian ini. Tahapan yang dilakukan pada penelitian ini adalah studi literatur, analisis kebutuhan, perancangan dan implementasi sistem, pengujian, pengambilan kesimpuan dan saran. Pengembangan sistem ini berbasis web dengan menggunakan *framework CodeIgniter*. Untuk membuat peta kondisi stok daerah digunakan *Google Maps API* dilengkapi dengan *layering KML*. Pengujian unit dan integrasi dilakukan dengan menggunakan metode *white box testing* dengan hasil 100% valid. Pengujian Validasi menggunakan metode *black box testing* dengan hasil 100% valid. Pengujian Kompatibilitas dilakukan dengan tiga browser berbeda dengan hasil 100% valid.

Kata kunci: PT Petrokimia Gresik, Distribusi Pupuk, *Waterfall*, *Google Maps API*, *Layering Google Maps*, Stok Opname.

repository.ub.ac.id

ABSTRACT

Moh. Wahyu Dwi Ridiansyah, ***Development of Web-based Fertilizer Distribution Management Application System (Case Study: PT Petrokimia Gresik).***

Supervisors: Fajar Pradana, S.ST, M.Eng dan Nurudin Santoso, S.T, M.T

PT Petrokimia Gresik is a state-owned enterprise that produces subsidized and non-subsidized fertilizers. In the distribution activities, there were several problems found. The first problem felt by a staff of the Department of Regional 1 was having problems in processing reporting data from the buffer warehouse because of the large amount of data available. The second problem was felt by the Regional 1 Manager who complained of the difficulty of monitoring the condition of the stock in the area quickly and efficiently. The third problem was complained by the Department of Regional 1 staff regarding the reporting of damaged fertilizer in Warehouse Buffer, the process that occurs now is inefficient and there are no media that can monitor the delivery of damaged fertilizer to the central warehouse for recycling. The fourth problem complained by the staff in the inventory section complained that the process was not efficient because the results of the stock inventory must be manually adjusted to the stock listing. Based on these problems, the system developer will do this research that can solve the problem. The waterfall method is used in the software development process in this study. The stages carried out in this study are literature study, needs analysis, system design and implementation, testing, taking conclusions and suggestions. The development of this system is web-based using the CodeIgniter framework. To create a map of the condition of the regional stock used the Google Maps API is equipped with KML layering. Unit and integration testing is done using the white box testing method with 100% valid results. Validation testing uses the black box testing method with 100% valid results. Compatibility testing is done with three different browsers with 100% valid results.

Keywords: PT Petrokimia Gresik, Fertilizer Distribution, Waterfall, Google Maps API, Google Maps Layering, Stock Identification.

UNIVERSITAS BRAWIJAYA

DAFTAR ISI

PENGESAHAN ii
PERNYATAAN ORISINALITAS iii
PRAKATA iv
ABSTRAK vi
ABSTRACT vii
DAFTAR ISI viii
DAFTAR TABEL xiv
DAFTAR GAMBAR xx
BAB 1 PENDAHULUAN 1
1.1 Latar Belakang 1
1.2 Rumusan Masalah 2
1.3 Tujuan 3
1.4 Manfaat 3
1.5 Batasan Masalah 3
1.6 Sistematika Pembahasan 4
BAB 2 LANDASAN KEPUSTAKAAN 5
2.1 Kajian Pustaka 5
2.2 Rekayasa Perangkat Lunak 6
2.3 SDLC (*Software Development Life Cycle*) 6
2.3.1 Metode *Waterfall* 6
2.4 Unified Modeling Language (UML) 8
2.4.1 Diagram *Use Case* 9
2.4.2 Diagram Sekuen 10
2.4.3 Kelas Diagram 11
2.5 Entity Relationship Diagram (ERD) 12
2.6 PT Petrokimia Gresik 13
2.6.1 Distribusi Pupuk PT Petrokimia Gresik 14
2.7 PHP 14
2.8 HTML 15
2.9 JavaScript 15

2.10 MySQL 15
2.11 MVC (Model View Controller) 16
2.12 CodeIgniter 17
2.13 Google Maps API 17
2.14 Pengujian Perangkat Lunak 17
2.14.1 Blackbox Testing 18
2.14.2 Whitebox Testing 18
BAB 3 METODOLOGI PENELITIAN 21
3.1 Studi Literatur 21
3.2 Analisis Kebutuhan 21
3.3 Perancangan dan Implementasi Sistem 22
3.4 Pengujian 22
3.5 Penarikan Kesimpulan dan Saran 22
BAB 4 ANALISIS KEBUTUHAN 23
4.1 Gambaran Umum Sistem 23
4.2 Proses Rekayasa Kebutuhan 23
4.2.1 Elisitasi Kebutuhan 23
4.2.2 Identifikasi Aktor 28
4.3 Aturan Penomoran 29
4.4 Kebutuhan Fungsional 30
4.5 Kebutuhan Non-Fungsional 54
4.6 Diagram Use Case 55
4.7 Skenario Use Case 55
BAB 5 PERANCANGAN DAN IMPLEMENTASI 86
5.1 Perancangan 86
5.1.1 Perancangan *Sequence* Diagram 86
5.1.2 Perancangan *Class* Diagram 88
5.1.3 Perancangan Komponen 92
5.1.4 Perancangan Basis Data 98
5.1.5 Perancangan Antarmuka 98
5.2 Implementasi 101
5.2.1 Spesifikasi Sistem 101

5.2.2 Implementasi Basis Data 102
5.2.3 Implementasi Kode Program 107
5.2.4 Implementasi Antarmuka 112
BAB 6 PENGUJIAN 115
6.1 Pengujian Unit 115
6.1.1 Pengujian Unit Klas Operator_transportir_c pada fungsi save_detail_permintaan_pupuk 115
6.1.2 Pengujian Unit Klas Peta_stok_c pada fungsi set_kml 118
6.1.3 Pengujian Unit Klas Operator_stok_opname_c pada fungsi save_detail_stok_opname 123
6.2 Pengujian Integrasi 125
6.3 Pengujian Validasi 127
6.3.1 Pengujian Validasi Melihat Jenis Pupuk 127
6.3.2 Pengujian Validasi Menambah Jenis Pupuk 128
6.3.3 Pengujian Validasi Mengubah Jenis Pupuk 129
6.3.4 Pengujian Validasi Melihat Rekanan Transportir 129
6.3.5 Pengujian Validasi Menambah Rekanan Transportir 130
6.3.6 Pengujian Validasi Mengubah Rekanan Transportir 131
6.3.7 Pengujian Validasi Melihat Batas Stok Kritis. 131
6.3.8 Pengujian Validasi Menambah Batas Stok Kritis 132
6.3.9 Pengujian Validasi Mengubah Batas Stok Kritis. 133
6.3.10 Pengujian Validasi Melihat Rekanan Pengelola Gudang Penyangga. 133
6.3.11 Pengujian Validasi Menambah Rekanan Pengelola Gudang Penyangga. 134
6.3.12 Pengujian Validasi Mengubah Rekanan Pengelola Gudang Penyangga. 135
6.3.13 Pengujian Validasi Melihat Gudang Penyangga. 136
6.3.14 Pengujian Validasi Menambah Gudang Penyangga 136
6.3.15 Pengujian Validasi Mengubah Gudang Penyangga. 137
6.3.16 Pengujian Validasi Melihat Petugas Stok Opname 138
6.3.17 Pengujian Validasi Menambah Petugas Stok Opname. 138
6.3.18 Pengujian Validasi Mengubah Petugas Stok Opname. 140
6.3.19 Pengujian Validasi Membuat Penugasan Stok Opname. 140

6.3.20 Pengujian Validasi Melihat Penugasan Stok Opname (Operator Diswil 1). 141
6.3.21 Pengujian Validasi Membuat Permintaan Pengiriman Pupuk. 142
6.3.22 Pengujian Validasi Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1). 143
6.3.23 Pengujian Validasi Melihat Detail Permintaan Pengiriman Pupuk. 144
6.3.24 Pengujian Validasi Melihat Status Pengiriman Pupuk. 144
6.3.25 Pengujian Validasi Menanggapi Permintaan Pengiriman Pupuk. 144
6.3.26 Pengujian Validasi Melihat Selisih Stok Opname di Gudang Penyangga. 147
6.3.27 Pengujian Validasi Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga). 147
6.3.28 Pengujian Validasi Membuat Perhitungan Stok Opname. 148
6.3.29 Pengujian Validasi Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1). 150
6.3.30 Pengujian Validasi Melihat Peta Kondisi Stok Daerah. 151
6.3.31 Pengujian Validasi Membuat Laporan Pupuk Rusak. 151
6.3.32 Pengujian Validasi Melihat Laporan Pupuk Rusak (Operator Diswil 1). 152
6.3.33 Pengujian Validasi Menerima Pupuk Masuk Gudang Penyangga 153
6.3.34 Pengujian Validasi Menambah Pupuk Keluar Gudang Penyangga. 154
6.3.35 Pengujian Validasi Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga). 155
6.3.36 Pengujian Validasi Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga). 156
6.3.37 Pengujian Validasi Validasi Hasil Stok Opname. 156
6.3.38 Pengujian Validasi Memproses Pemberian Pupuk 157
6.3.39 Pengujian Validasi Melihat Pupuk Hilang Saat Pengiriman. 158
6.3.40 Pengujian Validasi Menerima Pengembalian Pupuk Rusak. 158

6.3.41 Pengujian Validasi Mencetak Surat Jalan 159
6.3.42 Pengujian Validasi Validasi Pupuk Rusak 159
6.3.43 Pengujian Validasi Mengirim Pupuk Rusak 160
6.3.44 Pengujian Validasi Login 161
6.3.45 Pengujian Validasi Logout 162
6.3.46 Pengujian Validasi Melihat Daftar Pemberian Pupuk 162
6.3.47 Pengujian Validasi Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat) 163
6.3.48 Pengujian Validasi Melihat Gudang Penyangga yang Dikelola. 163
6.3.49 Pengujian Validasi Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga). 164
6.3.50 Pengujian Validasi Melihat Penugasan Stok Opname (Manajer Diswil 1) 164
6.3.51 Pengujian Validasi Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1). 165
6.3.52 Pengujian Validasi Melihat Riwayat Penugasan Stok Opname 165
6.3.53 Pengujian Validasi Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1) 166
6.3.54 Pengujian Validasi Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1) 166
6.3.55 Pengujian Validasi Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga). 167
6.3.56 Pengujian Validasi Melihat Laporan Pupuk Rusak (Manajer Diswil 1) 168
6.3.57 Pengujian Validasi Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga) 168
6.3.58 Pengujian Validasi Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir). 169
6.3.59 Pengujian Validasi Melihat Hasil Stok Opname (Operator Gudang Penyangga) 169
6.3.60 Pengujian Validasi Melihat Hasil Stok Opname yang Sudah Divalidasi (Manajer Diswil 1) 170
6.3.61 Pengujian Validasi Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Diswil 1). 170
6.3.62 Pengujian Validasi Melihat Stok Pupuk. 171

6.4 Pengujian Kompatibilitas 171
6.4.1 Pengujian Kompatibilitas Peramban Mozilla Firefox 172
6.4.2 Pengujian Kompatibilitas Peramban Google Chrome 175
6.4.3 Pengujian Kompatibilitas Peramban Microsoft Edge 178
6.5 Hasil Analisis Pengujian 181
BAB 7 Penutup 182
7.1 Kesimpulan 182
7.2 Saran 183
DAFTAR REFERENSI 184
LAMPIRAN A SURAT PERNYATAAN PENELITIAN 186
LAMPIRAN B HASIL WAWANCARA 187
LAMPIRAN C DATA GUDANG PENYANGGA PROVINSI JAWA TIMUR PT PETROKIMIA GRESIK 193
LAMPIRAN D FORM HITUNG STOK OPNAME 195

DAFTAR TABEL

Tabel 2.1 Notasi Diagram Use Case 9
Tabel 2.2 Komponen Diagram Sekuen 11
Tabel 2.3 ERD dengan Notasi Chen 12
Tabel 4.1 Identifikasi Aktor 28
Tabel 4.2 Kebutuhan Fungsional 30
Tabel 4.3 Kebutuhan Non-Fungsional 55
Tabel 4.4 Skenario Use Case Melihat Jenis Pupuk 57
Tabel 4.5 Skenario Use Case Manambah Jenis Pupuk 57
Tabel 4.6 Skenario Use Case Mengubah Jenis Pupuk 57
Tabel 4.7 Skenario Use Case Melihat Rekanan Transportir 58
Tabel 4.8 Skenario Use Case Manambah Rekanan Transportir 58
Tabel 4.9 Skenario Use Case Mengubah Rekanan Transportir 59
Tabel 4.10 Skenario Use Case Melihat Batas Stok Kritis 59
Tabel 4.11 Skenario Use Case Menambah Batas Stok Kritis 60
Tabel 4.12 Skenario Use Case Mengubah Batas Stok Kritis 60
Tabel 4.13 Skenario Use Case Melihat Rekanan Pengelola Gudang Penyangga .. 61
Tabel 4.14 Skenario Use Case Menambah Rekanan Pengelola Gudang Penyangga 61
Tabel 4.15 Skenario Use Case Mengubah Rekanan Pengelola Gudang Penyangga 62
Tabel 4.16 Skenario Use Case Melihat Gudang Penyangga 62
Tabel 4.17 Skenario Use Case Menambah Gudang Penyangga 63
Tabel 4.18 Skenario Use Case Mengubah Gudang Penyangga 63
Tabel 4.19 Skenario Use Case Melihat Petugas Stok Opname 64
Tabel 4.20 Skenario Use Case Menambah Petugas Stok Opname 64
Tabel 4.21 Skenario Use Case Mengubah Petugas Stok Opname 65
Tabel 4.22 Skenario Use Case Membuat Penugasan Stok Opname 65
Tabel 4.23 Skenario Use Case Melihat Penugasan Stok Opname (Operator Diswil 1) 66
Tabel 4.24 Skenario Use Case Membuat Permintaan Pengiriman Pupuk 66
Tabel 4.25 Skenario Use Case Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1) 67

Tabel 4.26 Skenario Use Case Melihat Detail Permintaan Pengiriman Pupuk 67
Tabel 4.27 Skenario Use Case Melihat Status Pengiriman Pupuk 68
Tabel 4.28 Skenario Use Case Menanggapi Permintaan Pengiriman Pupuk 68
Tabel 4.29 Skenario Use Case Melihat Selisih Stok Opname di Gudang Penyangga 69
Tabel 4.30 Skenario Use Case Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga) 69
Tabel 4.31 Skenario Use Case Membuat Perhitungan Stok Opname 70
Tabel 4.32 Skenario Use Case Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1) 71
Tabel 4.33 Skenario Use Case Melihat Peta Kondisi Stok Daerah 71
Tabel 4.34 Skenario Use Case Membuat Laporan Pupuk Rusak 72
Tabel 4.35 Skenario Use Case Melihat Laporan Pupuk Rusak (Operator Diswil 1) 72
Tabel 4.36 Skenario Menerima Pupuk Masuk Gudang Penyangga 73
Tabel 4.37 Skenario Use Case Menambah Pupuk Keluar Gudang Penyangga 73
Tabel 4.38 Skenario Use Case Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga) 74
Tabel 4.39 Skenario Use Case Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga) 74
Tabel 4.40 Skenario Use Case Validasi Hasil Stok Opname 75
Tabel 4.41 Skenario Use Case Memproses Pemberian Pupuk 75
Tabel 4.42 Skenario Use Case Melihat Pupuk Hilang Saat Pengiriman 76
Tabel 4.43 Skenario Use Case Menerima Pengembalian Pupuk Rusak 76
Tabel 4.44 Skenario Use Case Mencetak Surat Jalan 76
Tabel 4.45 Skenario Use Case Validasi Pupuk Rusak 77
Tabel 4.46 Skenario Use Case Mengirim Pupuk Rusak 77
Tabel 4.47 Skenario Use Case Login 78
Tabel 4.48 Skenario Use Case Logout 78
Tabel 4.49 Skenario Use Case Melihat Daftar Pemberian Pupuk 78
Tabel 4.50 Skenario Use Case Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat) 79
Tabel 4.51 Skenario Use Case Melihat Gudang Penyangga yang Dikelola 79
Tabel 4.52 Skenario Use Case Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga) 79

Tabel 4.53 Skenario Use Case Melihat Penugasan Stok Opname (Manajer Diswil 1) 80
Tabel 4.54 Skenario Use Case Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1) 80
Tabel 4.55 Skenario Use Case Melihat Riwayat Penugasan Stok Opname 80
Tabel 4.56 Skenario Use Case Melihat Pupuk masuk Gudang Penyangga (Operator Diswil 1) 81
Tabel 4.57 Skenario Use Case Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1) 81
Tabel 4.58 Skenario Use Case Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga) 82
Tabel 4.59 Skenario Use Case Melihat Laporan Pupuk Rusak (Manajer Diswil 1) 82
Tabel 4.60 Skenario Use Case Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga) 82
Tabel 4.61 Skenario Use Case Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir) 83
Tabel 4.62 Skenario Use Case Melihat Hasil Stok Opname (Operator Gudang Penyangga) 83
Tabel 4.63 Skenario Use Case Melihat Hasil Stok Opname Yang Sudah Divalidasi (Manajer Diswil 1) 84
Tabel 4.64 Skenario Use Case Melihat Hasil Stok Opname Yang Sudah Divalidasi (Operator Diswil 1) 84
Tabel 4.65 Skenario Use Case Melihat Stok Pupuk 85
Tabel 5.1 Spesifikasi Perangkat Keras 101
Tabel 5.2 Spesifikasi Perangkat Lunak 102
Tabel 5.3 Implementasi Basis Data Dengan DDL 102
Tabel 6.1 Hasil Pengujian Unit pada Fungsi save_detail_permintaan_pupuk 117
Tabel 6.2 Hasil Pengujian Unit pada Fungsi set_kml 120
Tabel 6.3 Hasil Pengujian Unit pada Fungsi save_detail_stok_opname 125
Tabel 6.4 Hasil Pengujian Unit pada Fungsi update_rekanan_gudang_penyangga 127
Tabel 6.5 *Test-case* Berhasil Melihat Jenis Pupuk 127
Tabel 6.6 *Test-case* Berhasil Menambah Jenis Pupuk 128
Tabel 6.7 *Test-case* Menambah Jenis Pupuk Alternatif 1 128
Tabel 6.8 *Test-case* Berhasil Mengubah Jenis Pupuk 129
Tabel 6.9 *Test-case* Berhasil Melihat Rekanan Transportir 129

Tabel 6.10 *Test-case* Berhasil Menambah Rekanan Transportir 130
Tabel 6.11 *Test-case* Menambah Rekanan Transportir Alternatif 1 130
Tabel 6.12 *Test-case* Berhasil Mengubah Rekanan Transportir 131
Tabel 6.13 *Test-case* Berhasil Melihat Batas Stok Kritis 132
Tabel 6.14 *Test-case* Berhasil Menambah Stok Kritis 132
Tabel 6.15 *Test-case* Menambah Stok Kritis Alternatif 1 132
Tabel 6.16 *Test-case* Berhasil Mengubah Batas Stok Kritis 133
Tabel 6.17 *Test-case* Berhasil Melihat Rekanan Pengelola Gudang Penyangga . 134
Tabel 6.18 *Test-case* Berhasil Menambah Rekanan Pengelola Gudang Penyangga 134
Tabel 6.19 *Test-case* Menambah Rekanan Pengelola Gudang Penyangga Alternatif 1 135
Tabel 6.20 *Test-case* Berhasil Mengubah Rekanan Pengelola Gudang Penyangga 135
Tabel 6.21 *Test-case* Berhasil Melihat Gudang Penyangga 136
Tabel 6.22 *Test-case* Berhasil Menambah Gudang Penyangga. 136
Tabel 6.23 *Test-case* Menambah Gudang Penyangga Alternatif 1 137
Tabel 6.24 *Test-case* Berhasil Mengubah Gudang Penyangga 138
Tabel 6.25 *Test-case* Berhasil Melihat Petugas Stok Opname 138
Tabel 6.26 *Test-case* Berhasil Menambah Petugas Stok Opname 139
Tabel 6.27 *Test-case* Menambah Petugas Stok Opname Alternatif 1 139
Tabel 6.28 *Test-case* Berhasil Mengubah Petugas Stok Opname 140
Tabel 6.29 *Test-case* Berhasil Membuat Penugasan Stok Opname 140
Tabel 6.30 *Test-case* Membuat Penugasan Stok Opname Alternatif 1 141
Tabel 6.31 *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Diswil 1) 142
Tabel 6.32 *Test-case* Berhasil Membuat Permintaan Pengiriman Pupuk 142
Tabel 6.33 *Test-case* Membuat Permintaan Pengiriman Pupuk Alternatif 1 143
Tabel 6.34 *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1) 143
Tabel 6.35 *Test-case* Berhasil Melihat Detail Permintaan Pengiriman Pupuk 144
Tabel 6.36 *Test-case* Berhasil Melihat Status Pengiriman Pupuk 144
Tabel 6.37 *Test-case* Berhasil Menanggapi Permintaan Pengiriman Pupuk 145
Tabel 6.38 *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 1 . 145

Tabel 6.39 *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 2 . 146
Tabel 6.40 *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 3 . 146
Tabel 6.41 *Test-case* Berhasil Melihat Selisih Stok Opname di Gudang Penyangga 147
Tabel 6.42 *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga) 148
Tabel 6.43 *Test-case* Berhasil Membuat Perhitungan Stok Opname 148
Tabel 6.44 *Test-case* Membuat Perhitungan Stok Opname Alternatif 1 149
Tabel 6.45 *Test-case* Membuat Perhitungan Stok Opname Alternatif 2 150
Tabel 6.46 *Test-case* Berhasil Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1) 150
Tabel 6.47 *Test-case* Berhasil Melihat Peta Kondisi Stok Daerah 151
Tabel 6.48 *Test-case* Berhasil Membuat Laporan Pupuk Rusak 151
Tabel 6.49 *Test-case* Membuat Laporan Pupuk Rusak Alternatif 1 152
Tabel 6.50 *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Diswil 1) 152
Tabel 6.51 *Test-case* Berhasil Menerima Pupuk Masuk Gudang Penyangga 153
Tabel 6.52 *Test-case* Menerima Pupuk Masuk Gudang Penyangga Alternatif 1 153
Tabel 6.53 *Test-case* Berhasil Menambah Pupuk Keluar Gudang Penyangga 154
Tabel 6.54 *Test-case* Menambah Pupuk Keluar Gudang Penyangga Alternatif 1 155
Tabel 6.55 *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga) 155
Tabel 6.56 *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga) 156
Tabel 6.57 *Test-case* Berhasil Validasi Hasil Stok Opname 156
Tabel 6.58 *Test-case* Validasi Hasil Stok Opname Alternatif 1 157
Tabel 6.59 *Test-case* Berhasil Memproses Pemberian Pupuk 157
Tabel 6.60 *Test-case* Berhasil Melihat Pupuk Hilang Saat Pengiriman 158
Tabel 6.61 *Test-case* Berhasil Menerima Pengembalian Pupuk Rusak 158
Tabel 6.62 *Test-case* Berhasil Mencetak Surat Jalan 159
Tabel 6.63 *Test-case* Berhasil Validasi Pupuk Rusak 159
Tabel 6.64 *Test-case* Validasi Hasil Stok Opname Alternatif 1 160
Tabel 6.65 *Test-case* Berhasil Mengirim Pupuk Rusak 160

Tabel 6.66 *Test-case* Berhasil Login 161
Tabel 6.67 *Test-case* Login Alternatif 1 161
Tabel 6.68 *Test-case* Login Alternatif 2 162
Tabel 6.69 *Test-case* Berhasil Logout 162
Tabel 6.70 *Test-case* Berhasil Melihat Daftar Pemberian Pupuk..... 162
Tabel 6.71 *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat)..... 163
Tabel 6.72 *Test-case* Berhasil Melihat Gudang Penyangga yang Dikelola..... 163
Tabel 6.73 *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga)..... 164
Tabel 6.74 *Test-case* Berhasil Melihat Penugasan Stok Opname (Manajer Diswil 1) 164
Tabel 6.75 *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1) 165
Tabel 6.76 *Test-case* Berhasil Melihat Riwayat Penugasan Stok Opname 165
Tabel 6.77 *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1) 166
Tabel 6.78 *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1) 167
Tabel 6.79 *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga) 167
Tabel 6.80 *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Manajer Diswil 1) 168
Tabel 6.81 *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga) 168
Tabel 6.82 *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir)..... 169
Tabel 6.83 *Test-case* Berhasil Melihat Hasil Stok Opname (Operator Gudang Penyangga) 169
Tabel 6.84 *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Manajer Diswil 1) 170
Tabel 6.85 *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Diswil 1) 171
Tabel 6.86 *Test-case* Berhasil Melihat Stok Pupuk 171

DAFTAR GAMBAR

Gambar 2.1 Model Waterfall 7
Gambar 2.2 Alur Distribusi PT Petrokimia Gresik 14
Gambar 2.3 Model Hubungan MVC 17
Gambar 2.4 Flow Graph 19
Gambar 2.5 Integrasi Top Down 20
Gambar 3.1 Diagram Alir Penelitian 21
Gambar 4.1 Proses Bisnis Pelaporan Pupuk Keluar-Masuk Gudang Penyangga Yang Sedang Digunakan 24
Gambar 4.2 Proses Bisnis Stok Opname Yang Sedang Digunakan 25
Gambar 4.3 Proses Bisnis Stok Opname Dengan Menggunakan Sistem 26
Gambar 4.4 Proses Bisnis Pelaporan Pupuk Rusak Yang Sedang Digunakan 27
Gambar 4.5 Proses Bisnis Pelaporan Pupuk Rusak Dengan Menggunakan Sistem 27
Gambar 4.6 Diagram Use Case 56
Gambar 5.1 Sequence Diagram Menanggapi Permintaan Pengiriman Pupuk 87
Gambar 5.2 Sequence Diagram Melihat Peta Kondisi Stok Daerah 88
Gambar 5.3 Sequence Diagram Membuat Perhitungan Stok Opname 89
Gambar 5.4 Diagram Class 90
Gambar 5.5 Detail Class Diagram Model 90
Gambar 5.6 Lanjutan Detail Class Diagram Model 91
Gambar 5.7 Lanjutan Detail Class Diagram Model 92
Gambar 5.8 Detail Class Diagram Controller 92
Gambar 5.9 Lanjutan Detail Class Diagram Controller 93
Gambar 5.10 Entity Relationship Diagram 99
Gambar 5.11 Perancangan Antarmuka Halaman Menu Permintaan Pupuk 100
Gambar 5.12 Perancangan Antarmuka Halaman Kondisi Stok Daerah 100
Gambar 5.13 Perancangan Antarmuka Halaman Hitung Stok Opname 101
Gambar 5.14 Physical Data Model 107
Gambar 5.15 Implementasi Antarmuka Permintaan Pupuk 112
Gambar 5.16 Implementasi Antarmuka Kondisi Stok Daerah 113
Gambar 5.17 Implementasi Antarmuka Hitung Stok Opname 114

Gambar 6.1 Flow Graph save_detail_permintaan_pupuk 116
Gambar 6.2 Flow Graph set_kml 119
Gambar 6.3 Flow Graph save_detail_stok_opname 124
Gambar 6.4 Flow Graph update_rekanan_gudang_penyangga 126
Gambar 6.5 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 1 172
Gambar 6.6 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 2 173
Gambar 6.7 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 3 173
Gambar 6.8 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 4 174
Gambar 6.9 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 5 174
Gambar 6.10 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 6 175
Gambar 6.11 Pengujian Kompatibilitas pada Peramban Google Chrome 1 175
Gambar 6.12 Pengujian Kompatibilitas pada Peramban Google Chrome 2 176
Gambar 6.13 Pengujian Kompatibilitas pada Peramban Google Chrome 3 176
Gambar 6.14 Pengujian Kompatibilitas pada Peramban Google Chrome 4 177
Gambar 6.15 Pengujian Kompatibilitas pada Peramban Google Chrome 5 177
Gambar 6.16 Pengujian Kompatibilitas pada Peramban Google Chrome 6 178
Gambar 6.17 Pengujian Kompatibilitas pada Peramban Microsoft Edge 1 178
Gambar 6.18 Pengujian Kompatibilitas pada Peramban Microsoft Edge 2 179
Gambar 6.19 Pengujian Kompatibilitas pada Peramban Microsoft Edge 3 179
Gambar 6.20 Pengujian Kompatibilitas pada Peramban Microsoft Edge 4 180
Gambar 6.21 Pengujian Kompatibilitas pada Peramban Microsoft Edge 5 180
Gambar 6.22 Pengujian Kompatibilitas pada Peramban Microsoft Edge 6 181

BAB 1 PENDAHULUAN

1.1 Latar Belakang

PT Petrokimia Gresik merupakan salah satu Perusahaan Badan Usaha Milik Negara (BUMN) yang memproduksi pupuk subsidi maupun non-subsidi. Sistem distribusi PT Petrokimia Gresik menggunakan gudang penyangga pada setiap kabupaten. Gudang penyangga ini berfungsi menjadi tempat penyimpanan pupuk yang dikirim dari Gudang Pusat di Gresik agar memudahkan dalam proses distribusi disetiap daerah di Indonesia. Dalam satu kabupaten atau daerah bisa terdiri dari satu atau lebih Gudang penyangga tergantung pada banyaknya kebutuhan pupuk di daerah tersebut dan daya tampung Gudang penyangga. Pengawasan stok di gudang penyangga dilakukan oleh Departemen Diswil (Distribusi Wilayah) 1. Dalam menyelenggarakan fungsinya, Departemen Diswil 1 mempunyai data berupa pendistribusian stok dan informasi stok pada setiap gudang penyangga.

Setiap gudang penyangga memiliki data pergudangan yang berbeda sehingga pengelolaan data pergudangan membutuhkan suatu cara tertentu yang digunakan untuk semua gudang sehingga mendapatkan data yang lebih baik. Selama ini pemantauan data setiap gudang penyangga menggunakan file *excel* yang merupakan hasil pengolahan dari file *document* yang dikirimkan oleh kepala gudang penyangga melalui *email*. Staf Departemen Diswil 1 yang mengolah data pelaporan dari Gudang penyangga mengalami kendala karena cukup banyak data yang harus diolah sehingga sering terjadi data yang terlewat dan tidak terolah. Hal ini berdampak lamanya tindak lanjut dari Departemen Diswil 1 apabila terjadi kekurangan stok pupuk di daerah sehingga dapat mempengaruhi hasil pertanian (Rafianto dan Udaieby, 2018). Keluhan tidak hanya dirasakan oleh Staf Departemen Diswil 1, Manajer mengalami kendala dalam memantau keadaan stok di daerah secara cepat dan efisien karena harus mengecek melalui data *table excel* ditengah banyaknya kegiatan.

Kendala juga terjadi dalam proses pelaporan pupuk rusak di gudang penyangga. Pada saat ini, pelaporan pupuk rusak juga masih dilakukan melalui *email* dengan mengirimkan bukti foto pupuk rusak dan alasan mengapa pupuk tersebut rusak untuk selanjutnya pelaporan yang diterima akan dicatat oleh Staf Departemen Diswil 1. Rafianto dan Udaieby (2018) mengungkapkan proses ini dianggap tidak efisien karena harus mencatat ulang data pelaporan melalui *email* yang cukup banyak dan harus menyesuaikan data jumlah stok pupuk sesuai dengan jumlah pupuk yang rusak. Tidak hanya itu, belum adanya media yang dapat digunakan untuk memantau pengiriman pupuk rusak dari gudang penyangga ke gudang pusat untuk didaur ulang kembali membuat Departemen Diswil 1 kesulitan mengawasi pengelolaan pupuk rusak.

Kendala juga terjadi pada proses stok opname. Stok opname ialah suatu kegiatan yang dilakukan dengan menghitung jumlah stok fisik yang tersedia di gudang. Tujuan dilakukan stok opname ini yaitu untuk dapat mengetahui seberapa

akurat antara stok yang tercatat dengan stok fisik/*real* yang ada. Apabila terjadi selisih antara hasil stok opname dengan stok yang tercatat, maka kemungkinan ada transaksi yang belum tercatat atau terjadi kecurangan dalam persediaan. Secara periodik Departemen Distribusi Wilayah 1 melakukan stok opname disetiap Gudang Penyangga, pada prosesnya masih menggunakan *hardcopy* dalam penghitungan maupun hasil stok opname sehingga meningkatkan resiko kehilangan data dan kurang efisien dalam pengolahan data karena setelah menyelesaikan penugasan stok opname staf harus melakukan *update* pencatatan stok pupuk di kantor sesuai dengan hasil stok opname secara manual. Hal ini diperkuat oleh Rafianto dan Udaieby (2018) yang mengungkapkan bahwa proses stok opname dianggap kurang efisien dan beberapa kali terjadi kehilangan data pada saat stok opname menyebabkan harus dilakukannya hitung ulang sehingga biaya perjalanan membengkak.

Berdasarkan permasalahan-permasalahan yang terjadi, diperlukan sistem yang dapat mengakomodasi proses distribusi pupuk dari gudang pusat ke gudang penyangga yang dilakukan PT Petrokimia Gresik. Sehingga, data stok dan proses distribusi yang sedang berjalan dapat dipantau secara *realtime* oleh pihak departemen distribusi wilayah. Selain itu, sistem juga harus dapat menjadi media pelaporan pupuk rusak beserta proses pengiriman kembali ke gudang pusat. Tidak hanya itu, sistem juga harus dapat menjadi sarana untuk membantu proses penugasan dan penghitungan stok opname yang langsung terintegrasi dengan informasi stok setiap gudang penyangga. Sehingga, proses *update* pencatatan stok yang mengacu pada hasil stok opname dapat berjalan efisien dan dapat menyelesaikan permasalahan kehilangan data yang terjadi. Kemudian, semua data stok pupuk yang ada disetiap gudang penyangga divisualisasikan pada sebuah peta dengan menggunakan *layering kml* dan *Google Maps API* dengan tujuan memberikan informasi kondisi stok disetiap daerah dalam kondisi aman atau tidak. *Layering kml* dan *Google Maps API* diterapkan karena dapat membuat penyampaian informasi stok menjadi lebih informatif. Selain itu, dari segi teknis *Google Maps API* dan *layering kml* memiliki performa yang cukup bagus dalam mentransformasikan data dengan jumlah yang cukup banyak.

Oleh karena itu, dikembangkan sebuah sistem untuk memudahkan pihak Departemen Distribusi Wilayah 1 dalam menyelenggarakan fungsinya yang diberi judul "Pengembangan Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web (Studi Kasus: PT Petrokimia Gresik)". Pengembangan sistem akan diaplikasikan pada platform web untuk memudahkan akses melalui berbagai perangkat komputer yang terhubung ke internet. Metode pengembangan perangkat lunak yang akan digunakan adalah metode *waterfall* dan pengujian perangkat lunak menggunakan metode pengujian *whitebox*, pengujian *blackbox* dan pengujian kompatibilitas.

1.2 Rumusan Masalah

Rumusan masalah dirumuskan dari latar belakang yang telah dijabarkan pada subbab sebelumnya. Berikut ialah rumusan masalah yang telah dirumuskan:

1. Bagaimana hasil yang didapatkan dari proses analisis kebutuhan dari sistem aplikasi manajemen distribusi pupuk yang didapatkan telah didapatkan dari pengguna?
2. Bagaimana hasil perancangan sistem aplikasi manajemen distribusi pupuk yang mengacu pada hasil analisis kebutuhan?
3. Bagaimana hasil implementasi sistem aplikasi manajemen distribusi pupuk yang mengacu pada hasil perancangan sistem?
4. Bagaimana hasil pengujian dari sistem aplikasi manajemen distribusi pupuk?

1.3 Tujuan

Berikut merupakan tujuan utama dalam penelitian ini:

1. Mengidentifikasi dan menganalisis kebutuhan Sistem Aplikasi Manajemen Distribusi Pupuk yang didapatkan dari kebutuhan pengguna.
2. Merancang Sistem Aplikasi Manajemen Distribusi Pupuk yang mengacu pada hasil dari analisis kebutuhan yang diperoleh.
3. Mengimplementasikan hasil perancangan Sistem Aplikasi Manajemen Distribusi Pupuk.
4. Melakukan pengujian terhadap Sistem Aplikasi Manajemen Distribusi Pupuk.

1.4 Manfaat

Dalam mengembangkan Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web ini, diharapkan dapat bermanfaat bagi PT Petrokimia Gresik terutama bagi Departemen Distribusi Wilayah 1. Karena dapat menjadi media distribusi data pendistribusian pupuk di Provinsi Jawa Timur, dapat membantu memantau proses pendistribusian pupuk dari Gudang pusat ke Gudang penyangga, dapat menjadi media yang dapat membantu proses stok opname di gudang penyangga, dapat menjadi media pelaporan dan pengiriman pupuk rusak dari gudang penyangga, dan dapat menyediakan peta kondisi stok pupuk di setiap daerah di Jawa Timur dilengkapi dengan indikator warna kondisi stok pupuk.

1.5 Batasan Masalah

Untuk mencegah terjadinya anomali terhadap tujuan dari penelitian diperlukan batasan masalah. Batasan masalah yang telah ditentukan yakni sebagai berikut:

1. Sistem hanya diaplikasikan di PT Petrokimia Gresik.
2. Sistem hanya akan diterapkan di wilayah distribusi Jawa Timur.
3. Sistem hanya untuk distribusi pupuk bersubsidi melalui gudang penyangga.
4. Sistem dikembangkan hanya pada platform *web* saja.
5. Sistem dikembangkan menggunakan basis data *MySQL*.
6. Sistem diimplementasikan menggunakan *framework* PHP, yaitu CodeIgniter.
7. Aplikasi hanya dapat diakses dengan bantuan koneksi internet untuk dapat

berjalan dengan baik.

8. Sistem hanya akan berjalan dengan baik apabila menggunakan peramban yang mendukung yaitu *Chrome, Firefox* dan *Edge*.

1.6 Sistematika Pembahasan

Pada penelitian ini, secara garis besar terdiri dari beberapa bagian. Untuk menjelaskan bagian-bagian yang akan dibahas maka, dibuatlah sistematika pembahasan yang berisi:

BAB I PENDAHULUAN

Bab ini berisikan mengenai latar belakang masalah yang diambil dalam penelitian, rumusan masalah, tujuan penelitian, manfaat yang diharapkan dari hasil penelitian, dan sistematika pembahasan.

BAB II TINJAUAN PUSTAKA

Bab ini berisikan mengenai metode-metode atau teori yang digunakan untuk mendukung atau mendasari pengembangan sistem dan penelitian ini.

BAB III METODOLOGI PENELITIAN

Bab ini berisikan mengenai metode atau cara penelitian yang menjelaskan metode dan setiap langkah yang dipakai dalam Pengembangan Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web.

BAB IV ANALISIS KEBUTUHAN

Bab ini berisikan mengenai hasil elisitasi dan analisis kebutuhan yang telah dilakukan dalam rangka merumuskan solusi yang tepat sesuai dengan kebutuhan *stakeholder* atau pengguna dalam Pengembangan Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web.

BAB V PERANCANGAN DAN IMPLEMENTASI

Bab ini berisikan mengenai perancangan dan implementasi dari sistem yang akan dibuat berdasarkan kebutuhan *stakeholder* atau pengguna yang telah dirumuskan pada tahap analisis kebutuhan dibab sebelumnya.

BAB VI PENGUJIAN

Bab ini berisikan mengenai pengujian dari sistem yang telah dibuat apakah ada kerusakan atau ketidaksesuaian yang terjadi pada tahap implementasi.

BAB VII KESIMPULAN DAN SARAN

Bab ini berisikan mengenai kesimpulan yang didapatkan dari penelitian dan pengembangan sistem yang telah dilakukan. Selain itu, pada bab ini juga akan memuat saran-saran dari penulis untuk pengembangan selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

Penunjang-penunjang penyusunan penelitian ini meliputi landasan kepustakaan dan juga teori mendasar yang dipakai akan dibahas pada bab ini. Pembahasan penelitian yang pernah dilakukan yang berkaitan dengan Sistem Aplikasi Manajemen Distribusi Pupuk akan dibahas pada bagian kajian pustaka. Pembahasan dilakukan sebagai pembanding dan penunjang pengetahuan yang dibutuhkan dalam mendukung pelaksanaan penelitian ini.

2.1 Kajian Pustaka

Beberapa penelitian berkaitan dengan judul dan domain masalah penelitian ini akan dikaji dalam subbab ini. Mengkaji penelitian terdahulu yang relevan dengan judul dan domain masalah diperlukan agar mendapatkan gambaran dan membantu dalam memutuskan penggunaan cara atau metode dalam pelaksanaan penelitian ini. Ditemukan beberapa penelitian-penelitian yang terkait dengan Sistem Aplikasi Manajemen Distribusi akan tetapi dari semua penelitian yang didapatkan, hanya empat penelitian yang digunakan sebagai referensi. Referensi yang pertama dari penelitian Kamaruddin Tone (2016), lalu penelitian oleh Oky Irnawati (2018), dan penelitian dari Ferry Juniardi dan Heri Azwansyah (2014), serta penelitian oleh Eko Budi Susilo dan Tri Listyorini (2015).

Penelitian pertama adalah penelitian oleh Kamaruddin Tone (2016) yang berjudul *Rancangan Bangun Sistem Informasi Distribusi Bantuan Sosial Beras miskin (Studi Kasus Kecamatan Binau Kabupaten Jeneponto)*. System Development Life Cycle (SDLC) yang digunakan dalam penelitian ini adalah *waterfall* dan dapat mengindentifikasi kebutuhan pengguna dengan baik. Dalam Penelitian ini juga menggunakan webgis untuk memperlihatkan data histori penyaluran raskin kecamatan binamu dan dihasilkan informasi yang lebih informatif dan memudahkan pengguna dalam menerima informasi yang diberikan.

Penelitian kedua yang dijadikan referensi adalah penelitian berjudul *Implementasi Metode Waterfall Pada Sistem informasi Stock Opname* oleh Oky Irnawati (2018). Metode *waterfall* digunakan sebagai metode pengembangan perangkat lunak pada penelitian ini. Penggunaan SDLC waterfall dalam pengembangan sistem dalam penelitian ini mampu menghasilkan sistem yang sesuai dengan kebutuhan pengguna dan dapat dipergunakan untuk menyelesaikan masalah yang ada.

Penelitian ketiga berjudul *Penyusunan Sistem Informasi Geografis Infrastruktur Transportasi Kabupaten Kapuas Hulu Berbasis Web* oleh Ferry Juniardi dan Heri Azwansyah (2014). Tujuan dari penelitian ini adalah untuk dapat memenuhi kebutuhan data dan informasi yang dapat menunjukan situasi dan karakteristik transportasi. *Google Maps API* digunakan dalam implementasi GIS untuk memperlihatkan informasi interaktif tentang data transportasi yang ada di Kabupaten Kapuas Hulu yang terdiri dari Transportasi Darat, Transportasi Laut dan

SDP serta Transportasi Udara. Pengunaan *GIS* dalam hal ini menggunakan *Google Maps API* terbukti dapat membantu pengguna dalam memahami informasi yang diberikan dengan cepat dan mudah.

Penelitian keempat yang dijadikan referensi adalah penelitian oleh Eko Budi Susilo dan Tri Listyorini (2015) yang berjudul *Perancangan Sistem Informasi Distribusi Obat Pasien Rawat Inap.* Aplikasi yang dikembangkan berbasis web dengan menggunakan bahasa pemrograman *PHP* dan database *MySQL* dengan menggunakan model pengembangan *waterfall*. Penggunaan model *waterfall* terbukti dapat dipergunakan dalam pengembangan sistem informasi untuk distribusi dengan baik dan dapat dihasilkan sistem yang sesuai dengan kebutuhan pengguna.

2.2 Rekayasa Perangkat Lunak

Dikutip dari buku Rekayasa Perangkat Lunak oleh Rosa dan Shalahuddin (2018), Rekayasa perangkat lunak (RPL) ialah penggunakan konsep atau prinsip rekayasa yang bertujuan mengembangkan atau menghasilkan mesin agar dapat dipergunakan sebagai sarana membantu setiap tugas manusia dengan efisien. Beberapa hal yang harus difokuskan dalam proses RPL adalah sebagai berikut:

A. Dapat dipelihara atau dikembangkan agar dapat mengikuti perkembangan lingkungan dan teknologi (*maintainability*)
B. Walaupun terjadi perubahan proses bisnis perangkat lunak harus tetap dapat bisa diandalkan (*dependability* dan *robust*)
C. Memiliki efisien yang baik dalam penggunaan maupun *resource* atau sumber daya yang dipakai
D. Memiliki kemampuan yang baik untuk dipakai sesuai kebutuhan yang ada (*usability*)

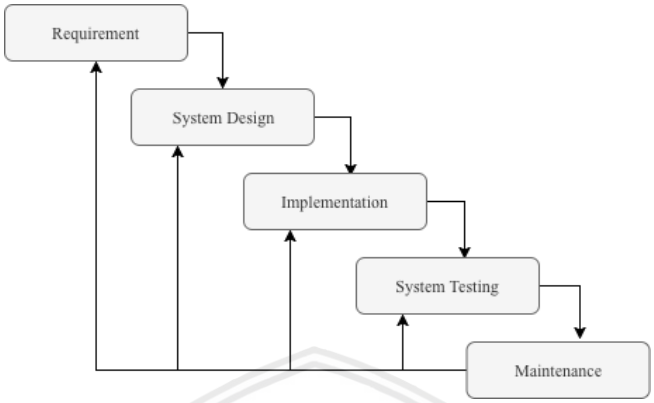
2.3 SDLC (*Software Development Life Cycle*)

Teknik atau metode atau model-model yang telah teruji dan digunakan dalam proses pengembangan atau pembuatan sebuah perangkat lunak yang disebut *Software Development Life Cycele* (SDLC) atau sering juga disebut *System Development life Cycle* (Rosa dan Shalahuddin, 2018). Beberapa metode yang dapat digunakan dalam SDLC ialah prototyping, V-model, *waterfall*, spiral, metode formal, dan *extreme programming*. Pada penelitian ini, metode *waterfall* yang digunakan dalam mengembangan sistem aplikasi manajemen distribusi pupuk.

2.3.1 Metode *Waterfall*

Model SDLC *waterfall* (air terjun) sering juga disebut model sekuensial linier (*sequential linear*). Metode *waterfall* memiliki alur yang mudah dipahami karena memiliki pendekatan berurutan secara sekuensial dimulai dari analisis, desain, implementasi (penulisan kode sumber), pengujian, dan tahap perbaikan/*maintenance* atau pendukung/*support* (Rosa dan Shalahuddin, 2011).

Selain itu, Menurut Sommerville (2011) ada beberapa langkah yang harus dilakukan dalam penggunaan model *waterfall*. Berikut merupakan tahapan-tahapan model *waterfall* dapat dilhat pada Gambar 2.1.



Gambar 2.1 Model Waterfall

Sumber: Sommerville (2011)

1. Analisis Kebutuhan/Requirement

Pada tahap analisis kebutuhan, seseorang yang akan melakukan pembuatan/pengembangan sistem harus menjalin komunikasi dengan pengguna agar dapat memahami masalah yang ada supaya perangkat lunak yang akan dikembangkan sesuai dengan kebutuhan pengguna dan mengetahui batas-batas apa saja yang ada untuk pengembang perangkat lunak ini. Beberapa cara komunikasi yang dapat dilakukan adalah diskusi, wawancara, ataupun *survey*.

2. Perancangan Perangkat Lunak / System Design

Hasil dari tahap analisis kebutuhan yang didapatkan pada tahap sebelumnya, bisa dipergunakan dalam fase ini untuk membuat desain sistem yang baik dan sesuai agar dapat diterapkan atau diimplementasikan dengan benar untuk menjadi sebuah perangkat lunak yang baik dan sesuai persyaratan yang ada.

3. Implementasi / Implementation

Semua hasil dari tahap *design* akan diimplementasikan pada tahap ini dengan melakukan pengkodean program. Pengkodean bisa dimulai dengan membuat unit-unit fungsi kemudian diintegrasikan menjadi sebuat sistem yang sesuai dan dapat diandalkan untuk menyelesaikan kebutuhan yang ada.

4. Pengujian / System Testing

Dalam fase ini, segala perangkat lunak atau sistem yang sudah dibuat dan diimplementasikan diuji dengan berbagai teknik yang memperhatikan tujuan atau hal-hal penting yang telah terdapat didalam sistem untuk mengetahui apakah ada kesalahan, ketidaksesuaian atau kerusakan yang terjadi.

5. Pemeliharaan / Maintenance

Setelah semua tahap selesai, pemeliharaan perlu dilakukan pada saat perangkat lunak telah selesai dibuat dan telah menyelesaikan tahap pengujian.

Setiap kerusakan dan ketidaksesuaian yang telah terdeksi pada tahap pengujian akan dilakukan perbaikan dan penyesuaian pada tahap pemeliharaan ini. Peningkatan kehandalan, kinerja dan efisiensi dari sistem atau perangkat lunak dapat dilakukan dengan melakukan perbaikan unit-unit yang ada.

Keunggulan atau beberapa poin yang dapat menjadi alasan dalam memilih metode *waterfall* ini dalam sebuah pengembangan perangkat lunak ialah:

1. Penerapan dan pengaplikasian mudah dan sederhana.
2. Pada awal proyek atau pengembangan dapat dilakukan pendefinisian kebutuhan sistem dengan mudah.
3. Penghematan dana, usaha dan waktu dapat dilakukan karena pendefinisian kebutuhan wajib diselesaikan diawal fase atau proyek.
4. Cocok digunakan pada sebuah proyek yang memiliki pengguna atau *stakeholder* yang tidak dapat ditemui setiap saat karena adanya tuntutan pendifinisian kebutuhan pada awal proyek.
5. Pengembangan dilakukan dengan proses fase satu demi satu, sehingga dapat mengurangi kesalahan yang bisa saja terjadi karena dikerjakan secara berurutan.
6. Tuntutan untuk mendefinisikan kebutuhan selengkap dan sebaik mungkin diawal cocok untuk proyek yang dikerjakan cukup jauh atau diluar kota dengan dana perjalanan yang terbatas.
7. Cocok digunakan apabila kebutuhan perangkat lunak sudah jelas dan dapat diperhitungkan di awal pembuatan, sehingga kesalahan dapat dihindari.

Kekurangan atau beberapa poin yang dapat menjadi alasan untk tidak memilih metode *waterfall* ialah:

1. Jika terjadi kesalahan pada suatu tahap atau fase maka akan berpengaruh langsung pada tahap atau fase setelahnya karena ketergantungan cukup tinggi pada tahap sebelumnya dan bersifat mengalir proses akan terus berjalan pada tahap selanjutnya.
2. Fase cukup kaku karena semua harus dikerjakan secara berurutan.

2.4 Unified Modeling Language (UML)

Salah satu arsitektur sistem yang menerapkan konsep OOAD (*Object-Oriented Design/Analysis*) menggunakan suatu bahasa yang konsisten yang digunakan untuk mengkontruksi, menentukan, mendokumentasikan, dan memvisualisasi *artifact* (beberapa informasi yang dipergunakan maupun dihasilkan dalam sebuah proses RPL. Dapat berupa deskripsi, perangkat lunak, maupun model) yang masuk dalam suatu sistem perangkat lunak.

UML merupakan bahasa pemodelan yang paling banyak dipakai dari tiga metode OO yang sebelumnya telah ada, yaitu Booch, OMT (*Object Modeling Technique*), dan OOSE (*Object-Oriented Software Engineering*).

Beberapa tujuan UML antara lain:

1. Menjadi media bahasa untuk memodelkan yang terbebas dari berbagai bahasa pemrograman yang ada maupun proses rekayasa.

2. Satu model yang siap digunakan untuk pengembangan dan menukarkan beberapa model dengan mudah dan dapat dimengerti secara luas. UML cukup ekspresif sebagai bahasa permodelan visual.
3. Beberapa diagram grafis dari UML yang merupakan praktek permodelan terbaik yang dapat digunakan membuat suatu model ialah sebagai berikut:
   a) Model bisnis *use case*
   b) Model *use case*
   c) Diagram *behavioral* yaitu diagram *sequence*
   d) Diagram implementasi yaitu diagram komponen, diagram *deployment*.
   e) Generate code.

Beberapa diagram tersebut memiliki kemampuan dokumentasi yang merupakan artifak utama dari sebuah UML. Beberapa diagram lainnya yang dianggap tidak termasuk dalam paradigma OO tidak termasuk dalam bagian dari UML. Beberapa diagram tersebut ialah *data-flow diagram* yang digantikan oleh *activity diagram* dan *collaboration diagram*.

2.4.1 Diagram *Use Case*

Untuk mempresentasikan intraksi dari aktor yang telah terdefinisi dalam sistem dan fungsional apa saja yang telah terdapat didalam sistem maupun siapa saja yang dapat menggunakanannya digunakanlah diagram use case. Diagram use case ialah salah satu dari beberapa permodelan yang digunakan untuk sistem *behavior* yang akan dikembangkan (Rosa dan Shalahuddin, 2018).

Beberapa hal utama dalam sebuah use case ialah tindakan pendefinisian aktor dan use case. Adapun beberapa syarat penamaan dalam sbeuah use case ialah nama haruslah dapat didefiniskan dengan sesimpel mungkin dan haruslah dapat dengan mudah dipahami (Rosa dan Shalahuddin, 2018). Komponen - komponen notasi diagram *use case* disajikan pada tabel Tabel 2.1:

Tabel 2.1 Notasi Diagram Use Case

NO	GAMBAR	NAMA	KETERANGAN
1		Aktor	Manusia atau sebuah proses dari sistem lain yang berada diluar sistem yang dapat terhubung/berinteraksi dengan sistem yang akan dikembangkan. Walaupun simbol aktor digambarkan dengan gambar manusia tapi sebuah aktor belum tentu merupakan manusia. Pendefinisian nama aktor dapat menggunakan kata benda dari frase awal nama aktor.

2	<<include>> ------->	*Include*	Sebuah tambahan hubungan dari sebuah use case yang pendefinisiannya ialah use case yang ditunjuk panah adalah syarat atau harus terpenuhi agar use case yang menunjuk dapat berjalan. Singkatnya use case yang ditunjuk panah diperlukan untuk menjalankan use case yang menunjuk.
3	<<extend>> <-------	*Extend*	Sebuah tambahan hubungan dari sebuah use case yang didefinisikan dengan arah panah yang menunjuk sebuah base use case yang dapat diperluas fungsinya kepada use case yang menunjuk tapi tidak merupakan sebuah syarat dijalankannya sebuah use case seperti halnya include lebih kepada sebuah pilihan (optional) untuk memperluas sebuah use case tersebut.
4		*Association*	Merupakan simbol komunikasi/interaksi antara aktor dengan use case begitupula sebaliknya.
5		*Use Case*	Merupakan lambang atau simbol sebuah fungsionalitas dari sistem yang akan dikembangkan. Nantinya akan dipresentasikan hubungan komunikasi/interkasi dengan aktor. Pendefinisian use case disarankan menggunakan kata kerja.

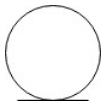
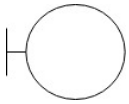
Sumber: Rosa dan Shalahuddin (2018)

2.4.2 Diagram Sekuen

Diagram sekuen (Sequence Diagram) merupakan cara untuk mendeskripsikan/mempresentasikan kelakuan/interaksi antar objek pada *use case*. Beberapa hal yang dideskripsikan dalam diagram sekuen adalah waktu hidup objek dan pesan yang dikirimkan maupun diterima dari atau ke objek lainnya. Adalah wajib untuk mengetahui objek-objek apa saja yang terlibat dalam sebuah

use case dan kapan dia terlibat, untuk itu diperlukannya untuk melihat use case skenario dalam membuat diagram sekuen (Rosa dan Shalahuddin, 2018). Komponen-komponen yang ada pada sekuen diagram dijabarkan pada tabel 2.2.

Tabel 2.2 Komponen Diagram Sekuen

Gambar	Keterangan
	Kelas *Entity* menggambarkan sebuah komponen yang menjadi database dari sistem yang akan dibangun.
	Kelas *Boundary* merupakan komponen yang merupakan jembatan yang menghubungkan antar aktor dengan sistem menggunakan tampilan visual.
	Kelas *Control* berisi *logic* atau logika sebagai pengatur anatara kelas *entity* dan kelas *boundary*.
	Syncronus Message direpresentasikan dengan panah dengan ujung penuh yang merupakan sebuah simbol untuk pengiriman pesan antar objek.
	Recursive direpresentasikan dengan panah ujung penuh yang menunjuk ke dirinya sendiri yang memiliki arti pengiriman pesan untuk dirinya sendiri.
	Activation direpresentasikan seperti persegi panjang vertikal yang menggambarkan objek yang dihidupakan dan akan melakukan eksekusi operasi pada suatu objek yang lain.
	Asyncronus Message direpresentasikan dengan panah dengan ujung tidak penuh yang merupakan symbol untuk mengirimkan pesan secara *Asyncronus* seperti ajax.
	Callback direpresentasikan dengan panah ke arah kiri yang menunjukkan *callback* dari objek yang dipanggil dengan menggunakan ajax.
	Return Message direpresentasikan dengan panah ke arah kiri yang merupakan sebuah symbol yang menunjukan balikan atau *return message* dari objek yang telah dipanggil melalui *message*.

Sumber: www.uml-diagrams.org

2.4.3 Kelas Diagram

Kelas diagram (class diagram) berguna untuk mempresentasikan struktur dari sebuah sistem yang akan dikembangkan begitupula dalam kelas-kelas yang

diartikan/didefinisikan yang nantinya dibuat dalam pengkodean. Method atau operasi dan attribut haruslah didefinisikan dalam kelas diagram. Method atau operasi ialah fungsi yang ada dalam suatu kelas dan attribut ialah variable yang didefinisikan dalam sebuah kelas.

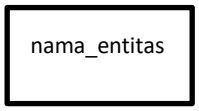
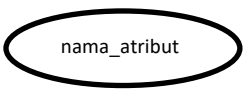
Setiap kelas didalam struktur sistem haruslah bisa menjalankan fungsinya sesuai dengan kebutuhan sistem sehingga perancangan kelas diagram dapat diimplementasikan dalam kode dengan baik dan benar (Rosa dan Shalahuddin, 2018). Beberapa hal yang haruslah ada pada sebuah susunan struktur kelas yang baik dan benar adalah:

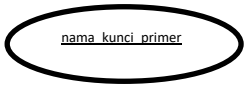
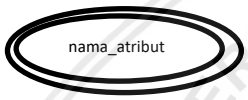
1. Kelas main, kelas yang akan dijalankan pada saat awal sistem dijalankan
2. Kelas view, kelas yang memiliki peran sebagai kelas yang mengatur tampilan untuk pengguna sebagai media interaksi dan eksekusi perintah.
3. Kelas controller, kelas yang memiliki peran atau fungsi sebagai tempat logika sistem dan penanganan sebuah proses bisnis yang dijalankan sistem
4. Kelas model, kelas yang memiliki peran atau fungsionalitasnya meliputi pengaturan data atau penghubung antara data perangkat lunak dengan basis data.

2.5 Entity Relationship Diagram (ERD)

Dalam memodelkan basis data relational kebanyakan pengembang menggunakan ERD. Teori himpunan matematika digunakan sebagai dasar pengembangan ERD. Notasi barker, notasi crow's foot, dan notasi chen adalah beberapa aliran notasi yang ada pada ERD. Notasi yang disebutkan terakhir yaitu notasi chen adalah notasi yang paling populer dan paling sering dipakai (Rosa dan Shalahuddin, 2018). Pada penelitian ini akan menggunakan notasi chen pada ERD, berikut adalah notasi chen yang bisa dibaca pada Tabel 2.3.

Tabel 2.3 ERD dengan Notasi Chen

NO	GAMBAR	NAMA	KETERANGAN
1		Entitas	Entitas ialah sebuah bagian dari ERD yang nantinya akan dijadikan tabel pada basis data yang digunakan. Entitias memiliki data-data yang disebut atribut nantinya akan menjadi field/kolom pada basis data.
2		Atribut	Atribut ialah data yang disimpan dalam sebuat entitas, pada implementasinya nantinya atribut akan menjadi kolom data pada sebuah tabel basis data.

3		Atribut kunci primer (primary key)	Atribut kunci primer ialah sebuah data atau kolom data yang dimasukan dalam sebuah entitas atau tabel yang menjadi penanti kunci akses record data yang diingkan agar memudahkan pengaksesan dan pembuatan relasi. Atribut kunci primer biasanya adalah sebuah id atau nomor dokumen. Atribut kunci primer haruslah sebuah data yang unik dengan kata lain tidak memiliki kesamaan dengan data lain yang ada dalam tabel.
4		Atribut multinilai (multivalue)	Atribut multinilai atau bisa disebut multivalue adalah suatu kolom data atau data yang memiliki nilai lebih dari satu yang disimpan dalam sebuah entitas.
5		Relasi	Relasi ialah simbol yang menjalaskan sebuah hubungan antar entitas. Pendefinisiannya biasanya menggunakan kata kerja.
6		Asosiasi	Asosiasi ialah sebuah simbol yang menandakan hubungan antar entitas dan relasi dengan memiliki multiplicity pada ujungnya yang memiliki makna jumlah maksimum ketehubungan antar entitas satu dengan entitas lainnya yang disebut kardinalitas.

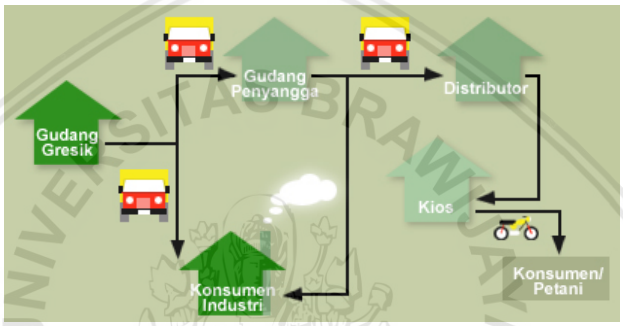
2.6 PT Petrokimia Gresik

Salah satu perusahaan pupuk terkomplet di Indonesia bahkan Asia Tenggara adalah PT Petrokimia Gresik. Perusahaan badan usaha milik negara (BUMN) ini pada saat ini tidak hanya memproduksi pupuk, tapi juga memproduksi produk non pupuk yaitu Asam Sulfat, Asam Fosfat, Amoniak, Dry Ice, Aluminum Fluoride, Cement Retarder, dan lain sebagainya. Perusahaan yang biasa disebut dengan nama petro saja oleh warga sekitar ini juga memproduksi pupuk subsidi pemerintah untuk petani antara lain PHONSKA, ZA, Urea, SP-36, dan Petroganik. Pada awal pendiriannya pada tanggal 10 Agustus 1964 PT Petrokimia Gresik

dimaksudkan untuk mendukung program pemerintah yaitu swasembada pangan nasional dengan membantu peningkatan produksi pertanian. Pada saat awal pendiriannya perusahaan ini bernama proyek petrokimia surabaya dan pada tanggal 10 Juli 1972 proyek ini diresmikan oleh Bapak Soeharto Presiden NKRI dengan nama PT Petrokimia Gresik (PT Petrokimia Gresik, 2018).

Penggunaan Teknologi Informasi dalam setiap proses bisnis yang dijalankan perusahaan bukanlah hal baru, setidaknya pada sekarang ini PT Petrokimia Gresik sedang mencanangkan penggunaaan teknologi informasi pada setiap proses bisnis perusahaan maka dari itu adanya Departemen Teknologi Informasi cukup sentral perannya pada saat ini. Pada saat ini PT Petrokimia Gresik tergabung dalam *holding company* dengan PT Pupuk Indonesia.

2.6.1 Distribusi Pupuk PT Petrokimia Gresik



Gambar 2.2 Alur Distribusi PT Petrokimia Gresik

Sumber: http://www.petrokimia-gresik.com/Pupuk/Sistem.Distribusi (2018)

Seperti dalam Gambar 2.2 alur distribusi pupuk PT. Petrokimia Gresik. Setiap gudang penyangga tidak memproduksi pupuk sendiri melainkan disediakan oleh gudang gresik (gudang pusat). Distributor mengambil pupuk pada gudang penyangga daerah domisili atau daerah edar distributor, selanjutnya distributor akan mendistribusikan pupuk ke kios dan petani.

2.7 PHP

PHP (*PHP Hypertext Preprocessing*) ialah salah satu bahasa *scripting* yang penerapannya dilakukan pada platform web, PHP sangatlah populer dikalangan pengembang perangkat lunak pada platform web atau yang biasa disebut *web developer*. Untuk membuat web secara dinamis dapat dilakukan menggunakan PHP dengan menyelipkannya pada kode HTM yang merupakan bahasa *markup* standar untuk platform web (Tim EMS, 2016).

Maksud dari bahasa *scripting* ialah ditanamkan atau disisipkannya bahsa ke dalam HTML. Untuk membedakan kode PHP dan kode HTML sebagai wadahnya, digunakan tag-tag PHP. PHP sangat populer dan dapat dipakai untuk memprogram situs web dinamis tipe apa pun, bahkan PHP dapat digunakan untk membangun CMS.Pembuatan halaman web yang interaktif dan dinamis banyak menggunakan PHP sebagai toolnya karena PHP memiliki performa yang cukup

powerful. PHP banyak digunakan dan merupakan alternatif untuk menggantikan bahasa pemrograman lain, seperti ASP dari Microsoft (Tim EMS, 2016).

2.8 HTML

Jubilee Enterprise (2014) mengatakan dalam bukunya bahwa HTML merupakan bahasa *markup* standar untuk dunia web, HTML mengatur penyajian informasi dalam dunia internet dan juga dapat menjadi sarana pengguna berpindah tempat dari satu tempat ke tempat lainnya dalam dunia internet. HTML merupakan singkatan dari Hypertext Markup Language yang berarti suatu text yang jika diklik akan membawa pengguna berpindah dari suatu dokumen atau tempat ke tempat lainnya. Dalam penerapan HTML memilki markup atau tag yang berfungsi mengatur layout dan tampilan-tampilan visual yang kita lihat di sebuah website, termasuk font, warna teks, gambar, dan sebagainya.

2.9 JavaScript

Salah satu bahasa *scripting* untuk platform web yang sangat populer selain PHP adalah JavaScript. Untuk menyisipkan javascript didalam HTML pengembang dapat menggunakan tag script. JavaScript adalah bahasa pemrograman yang ringan selain itu juga dapat menambah interaktif sebuah web yang dibangun. JavaScript biasanya disisipkan dalam halaman HTML dan dapat diekseuksi tanpa proses kompilasi. Untuk menulis sebuat kode javascript ada beberapa cara yang dapat digunakan yaitu dibagian <head>, <body> atau menempatkannya dalam file eksternal kemudian dipanggil dalam kode. Pemanggilan kode javascript dilakukan saat halaman diload atau dieksekusi oleh web peramban namun kadang juga bisa diload atau dieksekusi saat terjadi pemanggilan event tertentu (Sunyoto, 2007).

2.10 MySQL

Salah satu perangkat lunak yang paling populer dalam membantu pengembang untuk memanajemen basis data berbasis SQL adalah MySQL. Perangkat lunak ini biasanya disebut dengan *Database Management System* (DBMS). MySQL tersedia sebagai perangkat lunak dengan lisensi GNU *General Public License* (GPL) yang berarti perangkat lunak ini dapat dipakai secara gratis, tetapi dalam kasus-kasus tertentu pengguna harus memakai lisensi komersial.

MySQL menggunakan konsep database yang sudah sejak lama dipakai, yaitu *Structured Query Language* (SQL). MySQL merupakan server basis data yang termasuk cepat dan mudah dioperasikan, untuk sekarang ini banyak sekali pengembang aplikasi web yang menggunakan mysql sebagai server basis data mereka. Beberapa hal yang dapat dijadikan alasan untuk memilih mysql sebagai server basis data yang akan dipakai adalah:

1. *Multiuser*, Memiliki kemampuan untuk digunakan dengan lebih dari satu user tanpa mengalami konflik atau pun masalah.
2. *Portability*, Memiliki kemampuan untuk berjalan dengan lancar pada berbagai basis sistem operasi.

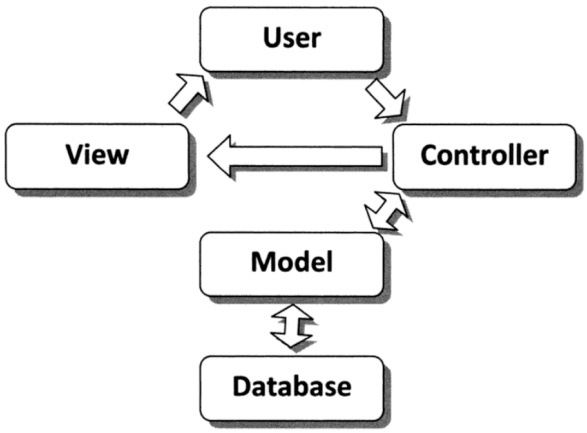
3. *Scalability*, Memiliki kemampuan yang baik dalam menangani data yang banyak, dapat menangani lebih dari 50 juta *records*, 60 ribu tabel, 5 miliyar baris, serta 32 indeks dalam satu tabel.
4. *Security*, Kemampuan sekuritasnya tidak diragukan lagi. Memiliki beberapa lapis keamanan seperti level *subnetmask*, *hostname*, dan akses izin *user*, serta *password* yang dienkripsi.

2.11 MVC (Model View Controller)

Salah satu pola pemrograman yang memisahkan atau membagi antara fungsi logika atau proses, data dan tampilan adalah MVC. Dengan membagi tiga inti dari sebuah perangkat lunak memudahkan pengembangan untuk melakukan maintenance atau perbaikan dalam aplikasi yang sudah selesai. Tidak hanya itu pembagian ini juga memudahkan pengembang untuk melanjut pengembangan suatu perangkat lunak yang sebelumnya dikembangkan oleh pengembang lain (Wardana, 2010). Model hubungan MVC dijelaskan di Gambar 2.3.

Bagian terdepan dari sebuah perangkat lunak yang menggunakan arsitektur MVC adalah *view*. *View* memiliki fungsi seperti namanya yaitu sebagai sarana pengguna melihat atau berinteraksi dengan sebuah perangkat lunak. Pengguna akan memberikan perintah dalam *view* untuk kemudian dilanjutkan kepada *controller* untuk diproses oleh perangkat lunak. Sederhananya view memiliki tugas untuk mengatur segala sesuatu yang berhubungan dengan tampilan sebuah perangkat lunak maupun bagaimana cara perangkat lunak berinteraksi ataupun memberikan informasi kepada penggunanya.

Bagian selanjutnya adalah model, komponen atau bagian ini adalah komponen yang berfungsi atau bertugas dalam mengelola data yang dihasilkan maupun diperlukan oleh sebuah perangkat lunak. Komponen ini yang berhubungan langsung dengan basis data. Bagian paling sentral dari arsitektur MVC adalah *controller*, *controller* memiliki banyak tugas salah satunya adalah menghubungkan *view* dan *model*. Selain itu, logika bisnis dan logika program dari sebuah perangkat lunak MVC ditempatkan disini.



Gambar 2.3 Model Hubungan MVC

2.12 CodeIgniter

CodeIgniter adalah salah satu framework berbasis web yang menyediakan fungsi-fungsi kode yang dapat digunakan kembali (reusable) dalam pembuatan website dan ini mempercepat efisiensi dalam mengembangkan aplikasi web. Banyak sekali kelebihan menggunakan codeigniter, antara lain penghematan waktu, script website menjadi lebih mudah dibaca dan diperbarui, mempermduah pembuatan sistem besar dengan terstruktur, kode script kita menjadi lebih mudah di kelola dan karena adanya pengelompokan kode maka akan mempermudah pembagian kerja dalam tim jika kita membuat website secara tim (Wardana, 2010).

2.13 Google Maps API

Penggunaan Google Maps API adalah salah satu cara menerapkan GIS dalam platform web. Google Maps API sendiri yakni metode yang disediakan oleh Google untuk mempermudah pengembang dalam mengaplikasikan fitur yang ada didalam Google Maps. Fitur pada Google Maps yang diaplikasikan ialah menampakkan peta, memasang *marker* untuk memperlihatkan posisi, dan lain-lain. Google Maps webservice mempunyai fitur Google Maps Geocoding API. Google Maps Geocoding API ialah layanan yang menyediakan fitur untuk geocoding dan geocoding terbalik.

Geocoding ialah cara kerja dalam mengkonversi domisili (nama jalan) menjadi wujud koordinat geografis yang mempunyai lintang dan bujur, kemudian spot koordinat akan diposisikan pada peta dan diberikan pertanda marker pada peta. Geocoding terbalik ialah cara kerja dalam mengkonversi koordinat geografis menjadi format domisili (nama jalan).

2.14 Pengujian Perangkat Lunak

Untuk menjaga dan memastikan bahwa perangkat lunak sudah sesuai dengan apa yang diharapkan pada saat proyek dimulai perlu dilakukan pengujian pada perangkat lunak. Selain itu, untuk menghindari adanya error atau bug saat dalam kondisi-kondisi tertentu perlunya dilakukan pengujian terhadap logika yang

ada pada sistem bisa dengan menggunakan pengujian unit pada pengujian *blackbox*. Memang sangat sulit membuat sebuah perangkat lunak terbebas bersih dari bug namun yang bisa dilakukan oleh pengembang hanya meminimalisir bug yang ada dan menghindari adanya bug yang sangat sentral dalam perangkat lunak yang dapat merubah alur proses bisnis suatu perangkat lunak. Maka, sebelum dikembangkan lebih lanjut diperlukan sebuah perangkat lunak diuji terlebih dahulu.

Pengujian PL (Perangkat Lunak) merupakan sebuah kesatuan aktifitas yang dirancangan secara sistematis untuk menguji atau menilai suatu perangkat lunak sesuai dengan hasil yang diharapkan. Pengujian PL bisa diawali dengan pengujian unit, unit disini dapat berupa kelas atau method dalam pemrograman berorientasi objek (OOP). Sesudah unit-unit selesai diuji maka dapat dilaksanakan pengujian integrasi. Pengujian integrasi sebaiknya dikerjakan secara berjenjang, tak dijalankan secara satu tahap lantas diakhir untuk menghindari kesusah penyelidikan bug yang ada. Berbeda dengan pengujian unit, pengujian integrasi lebih kepada menguji hubungan antara satu atau lebih kelas pada perangkat lunak yang dikembangkan (Rosa dan Shalahuddin, 2018).

2.14.1 Blackbox Testing

Pengujian perangkat lunak dari sektor spesifikasi fungsional tanpa menguji desain dan kode program disebut *Blackbox testing* (pengujian kotak hitam). Pengujian dilakukan dengan melihat apakah masukan dan keluaran sudah sesuai dengan apa yang diharapkan oleh pengembangkan tanpa melihat kode sumber perangkat lunak. Dalam hal ini pengujian kotak hitam cenderung dapat menemukan hal-hal berikut:

a. Fungsi yang tak sesuai atau tak ada.
b. Kekeliruan antarmuka (*interface errors*).
c. Kekeliruan pada struktur data dan akses basis data.
d. Kekeliruan performansi (*performance errors*).
e. Kekeliruan inisialisasi dan terminasi.

Ada banyak jenis metode pengujian blackbox, namun untuk penelitian ini digunakan metode pengujian berbasis skenario berdasarkan pada skenario use case yang dibuat. Pengujian berbasis skenario ini berfokus pada apa yang dikerjakan pengguna, bukan apa yang dikerjakan produk (Pressman, 2010).

2.14.2 Whitebox Testing

Berbeda dengan pengujian kotak hitam yang menguji tanpa melihat kode sumber dari perangkat lunak, whitebox testing (pengujian kotak putih) selain menguji kesesuaian antara masukan dan keluaran namun juga melihat apakah lojik atau logika dari kode program berjalan sesuai dengan keinginan. Dalam pengujian kotak putih pengembang harus mengikuti standar yang ada dalam membuat *test-case* seperti halnya dalam kasus *looping*, seleksi kondisi dll. Berikut beberap hal yang menjadi alasan penggunaan metode pengujian kotak putih:

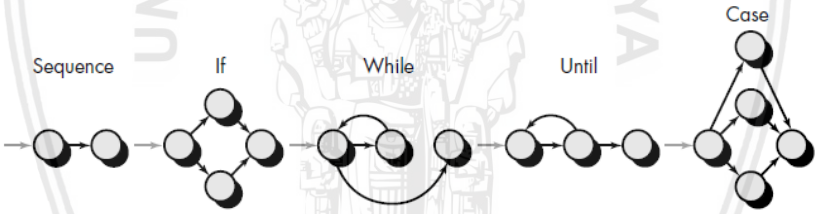
a. Menjamin segala jalur independen dalam suatu method telah digunakan minimal sekali dan berjalan secara semestinya.
b. Mengaplikasikan selutuh keputusan lojik dari segala keadaan *true* atau *false*.
c. Menjalankan segala perulangan dengan batasan nilai yang ada pada kondisi perulangan
d. Menerapkan struktur data internal untuk menjamin bahwa setiap jalur keputusan berjalan dengan baik.

2.14.2.1 Basic Path dan Cyclomatic Complexity

Basic path testing adalah sebuah metode dalam *white-box testing*. Metode ini memungkinkan perancang kasus uji untuk mendapatkan kompleksitas dari *algoritme*. Mengukur desain prosedural dan menggunakan urutan ini sebagai panduan untuk menentukan sebuah basis set dari jalur yang diesekusi (Pressman, 2001).

A. Flow Graph

Sebelum metode basis path dilakukan maka sebuah notasi sederhana dari representasi control flow yang disebut Flow Graph harus diperkenalkan. Flow Graph mengambarkan aliran kontrol logis dengan menggunakan notasi pada gambar 2.4



Gambar 2.4 Flow Graph

Sumber: Pressman (2011)

B. Cyclomatic Complexity

Cyclomatic Complexity merupakan suatu ukuran yang digunakan untuk dapat mengetahui kompleksitas logika dari suatu program. Ada tiga cara yang dapat dilakukan dalam menghitung cyclomatic complexity V(G):

1. V(G) = E – N + 2

2. V(G) = Region

3. V(G) = P + 1

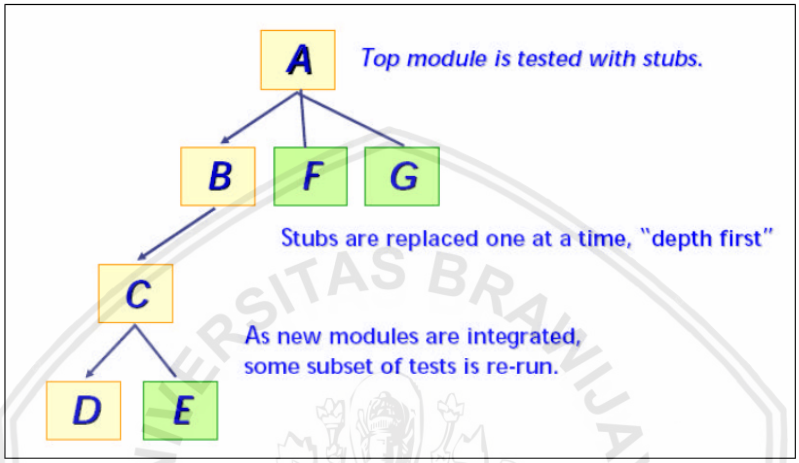
Dimana V(G) = Flow Graph, E = Edges dan N = Node (Pressman, 2001).

2.14.2.2 Pengujian Integrasi Top Down

Integrasi Top Down adalah pendekatan *incremental* dengan menggerakkan ke bawah melalui hirarki control, dimulai dengan control utama. Strategi intergrasi top-down memeriksa control mayor atau keputusan pada saat awal di dalam proses pengujian. Pada struktur program yang difaktorkan dengan baik, penarikan

keputusan terjadi pada tingkat hirarki yang lebih tinggi sehingga terjadi lebih dulu. Seperti dapat dilihat pada gambar 2.5.

1. Munguji lapisan atas atau sub sistem pemanggil terlebih dahulu.
2. Mengkombinasikan semua sub sistem yang dipanggil oleh sub sistem yang telah diuji dan melakukan pengujian terhadap sub sistem yang hasil kombinasi tadi.
3. Dilakukan secara berulang sampai semua sub sistem di uji.



Gambar 2.5 Integrasi Top Down

2.14.2.3 Pengujian Kompatibilitas

Untuk pengujian kebutuhan non-fungsional menggunakan pengujian compatibility. Pengujian ini bertujuan untuk memeriksa apakah sistem yang dibangun dapat digunakan di berbagai perangkat browser. Perbedaan browser terkadang memberikan hasil yang sedikit berbeda tanpa memperhatikan tingkat standarisasi HTML dalam aplikasi web (Pressman, 2010).

BAB 3 METODOLOGI PENELITIAN

Bab atau tahap ini dijabarkan setiap langkah yang dilaksanakan pada proses penyelesaian penelitian ini. Pada bagian ini akan diaplikasikan sebagai petunjuk dalam pelaksanaan penelitian. Setiap langkah yang dipergunakan seperti yang nampak pada Gambar 3.1 yakni studi literatur, analisis kebutuhan, perancangan sistem, implementasi, pengujian dan penarikan kesimpuan dan saran.

Gambar 3.1 Diagram Alir Penelitian

3.1 Studi Literatur

Subab ini dijabarkan pengambilan studi literatir yang dapat mendukung pengerjaan dan penyelesaian penelitian ini. Pemilihan rujukan dapat berasal dari berbagai jurnal ilmiah yang ada, buku, website, dan setiap tulisan ilmiah yang berhubungan maupun dapat dipergunakan dalam mendukung penyelesaian permasalahan penelitian. Beberapa poin teori yang dapat dijadikan rujukan adalah:

1. Rekayasa Perangkat Lunak (RPL)
2. SDLC (Sofware Development Life Cycle)
3. Unifield Modeling Language (UML)
4. PT Petrokimia Gresik
5. PHP
6. HTML
7. JavaScript
8. MySQL
9. MVC (Model View Controller)
10. CodeIgniter
11. Google Maps API
12. Pengujian Perangkat Lunak

3.2 Analisis Kebutuhan

Dalam fase ini dilakukan analisa terhadap kondisi yang terjadi saat ini. Tahap ini merupakan tahap terpenting dalam mengembangkan suatu sistem. Pada tahap ditujukan untuk mengetahui kebutuhan fungsional dan non-fungsional. Pada tahap ini akan dilakukan wawancara yang akan dilakukan kepada Departemen Distribusi Wilayah 1 PT Petrokimia Gresik. Dari situ bisa mendapatkan gambaran umum sistem, identifikasi aktor yang terlibat, daftar kebutuhan yang nantinya dibuat modelnya pada diagram use case dan skenario use case. Selain melakukan wawancara juga dilakukan observasi untuk mengidentifikasi hal yang tidak

didapatkan dalam wawancara. Hasil dari proses wawancara dan observasi dilampirkan pada bagian lampiran.

3.3 Perancangan dan Implementasi Sistem

Tahapan selanjutnya yaitu melakukan perancangan sistem. Tahap ini dilakukan ketika semua kebutuhan sistem telah tersedia. Kebutuhan sistem yang sudah diperoleh melalui tahap analisis kebutuhan kemudian akan menghasilkan perancangan awal, dari perancangan awal akan dimodelkan. Tahap ini merupakan proses menterjemahkan kebutuhan yang didapat sebelumnya menjadi model perangkat lunak menggunakan pemodelan UML (Unified Modeling Language). Permodelann UML yang akan dilakukan adalah perancangan diagram kelas, perancangan siagram sekuen, perancangann *database*, dan perancangan *interface*.

Perancangan yang selesai dibuat fase sebelumnya kemudian diterapkan kedalam kode-kode pemrograman. Membuat fitur-fitur yang dibutuhkan untuk digabungkan dan dintergrasikan untuk menjadi suatu Aplikasi. Implementasi dilakukan memakai bahasa pemograman *PHP, Javascript, HTML, CSS, Ajax dan framework CodeIgniter*.

3.4 Pengujian

Pada fase ini difokuskan untuk mencari kesalahan-kesalahan yang ada pada perangkat lunak entah itu kesalahan logika, alur program maupun ketidaksesuaian antara masukan dan keluaran yang diharapkan. Dalam tahap ini digunakan metode pengujian kotak hitam dan kotak putih. Secara garis besar pengujian dalam tahap ini difokuskan untuk menemukan ketidaksesuaian antara perangkat lunak yang telah dibuat dengan analisis kebutuhan yang dilakukan maupun dengan perancangan yang telah dibuat. Selain itu, alur logika program akan dilihat dan dicek dengan metode pengujian kotak putih serta kesesuian perangkat lunak dengan kebutuhan yang telah dirumuskan dapat dilihat dengan metode pengujian kotak hitam.

Pengujian kotak putih menggunakan metode *basic path testing* dan pengujian integrasi menggunakan pendekatan *top down*. Pada pengujian kotak hitam menggunakan teknik *scenario-based testing*. Serta untuk kebutuhan non-fungsional dilakukan pengujian kompatibilitas.

3.5 Penarikan Kesimpulan dan Saran

Fase yang terakhir di penelitian ini difokuskan dalam merumuskan kesimpulan dari penelitian ini maupun saran yang dapat dipergunakan untuk pengembangan selanjutnya dari penelitian ini ataupun perangkat lunak yang dihasilkan. Pengembang seharusnya mempertimbangkan saran yang nantinya akan dijabarkan dalam melakukan pengembangan lanjut dari hasil penelitian ini.

BAB 4 ANALISIS KEBUTUHAN

Analisis kebutuhan merupakan *first step* untuk mengembangkan *software*/sistem. Dari analisis kebutuhan ini didapatkan kebutuhan-kebutuhan yang ada pada sistem ini. didapatkannya hasil yang dibutuhkan pada fase ini dari hasil studi lapangan di Departemen Distribusi Wilayah 1 PT Petrokimia Gresik dengan cara wawancara dan observasi. Dalam tahap ini, didapatkan aktor-aktor yang ada didalam sistem dan bisa melakukan apa saja, serta dapat didefinisikan kebutuhan fungsional dan non-fungsional.

4.1 Gambaran Umum Sistem

Sistem Aplikasi Manajemen Distribusi Pupuk adalah sistem yang diwujudkan dengan tujuan untuk menjadi media berbagi data atau informasi antar stakeholder distribusi dari Gudang pusat ke Gudang penyangga untuk memastikan proses distribusi berjalan baik dan data dapat terekam dengan baik. Selain itu, dapat menjadi media pemantauan data stok di gudang penyangga secara realtime. Dilengkapi juga dengan validasi keakuratan data dilapangan dengan melakukan stok opname yang datanya dapat langsung diproses dalam sistem. Disediakan juga media untuk melaporkan pupuk rusak dan memantau pengiriman pupuk rusak dari gudang penyangga ke gudang pusat.

4.2 Proses Rekayasa Kebutuhan

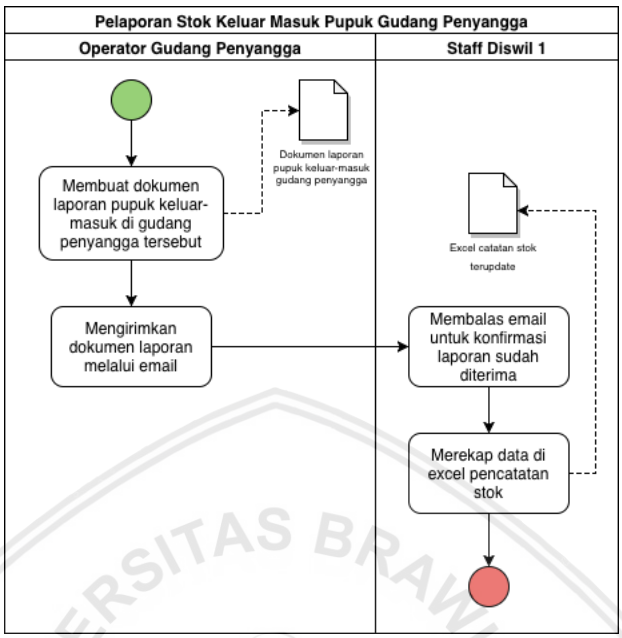
Proses rekayasan kebutuhan merupakan langkah atau cara yang digunakan dalam menemukan persyaratan dari pengguna duntuk mendapatkan kebutuhan sistem yang tepat.

4.2.1 Elisitasi Kebutuhan

Elisitasi merupakan tahapan penggalaian kebutuhan yang bertujuan membantu pengembang dalam memahami sistem yang akan dispesifikasikan. Dari hasil wawancara yang dilakukan dengan staf departemen distribusi wilayah 1 yang bertugas dalam bagian data stok atas nama Edwyk Sony Udaieby dan staf bagian stok opname atas nama Rachmad Edy Rafianto. Selain melakukan wawancara juga dilakukan observasi dalam proses bisnis yang berhubungan dengan sistem selama kurang lebih satu bulan. Hasil wawancara dapat dilihat pada Lampiran B.

Pada Gambar 4.1 merupakan proses bisnis dari pelaporan pupuk keluar-masuk di gudang penyangga. Proses pelaporan dimulai dengan membuat dokumen laporan pupuk keluar-masuk di gudang penyangga oleh operator gudang penyangga pada gudang tersebut. Kemudian laporan dikirimkan melalui email kepada staf departemen diswil 1. Lalu staf departemen diswil 1 membalas email untuk melakukan konfirmasi bahwa laporan telah diterima untuk selanjutnya dilakukan rekap data pada data pencatatan stok pupuk. Pada proses bisnis ini ditemukan kelemahan yaitu seringnya terjadi data yang terlewat tidak terekap karena prose perekapan masih *manual*. Pada sistem nanti tidak diperlukan melakukan pelaporan stok keluar-masuk pupuk gudang penyangga karena data

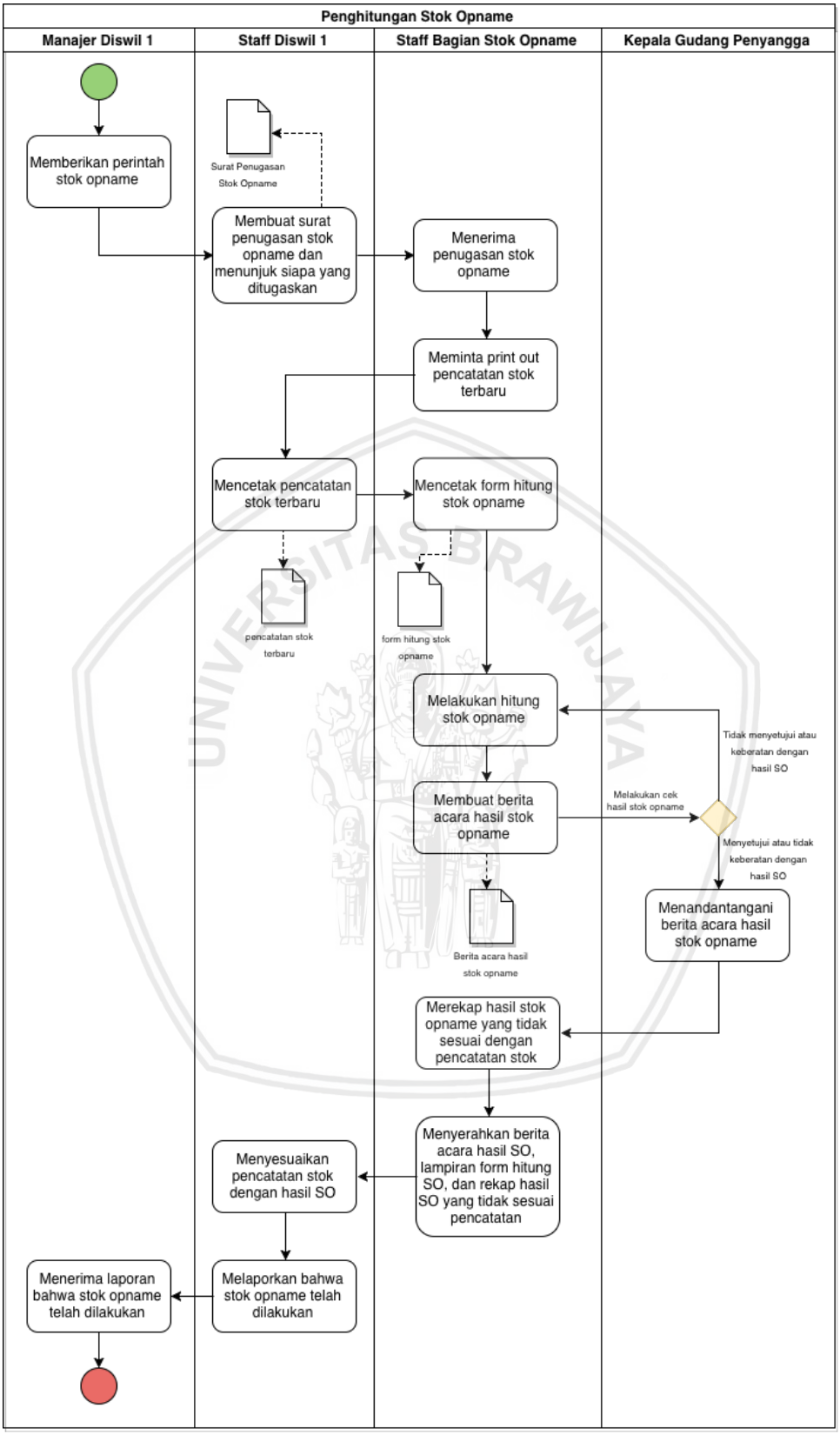
stok pupuk akan secara otomatis didapatkan dari proses pengiriman pupuk yang dilakukan juga pada sistem.



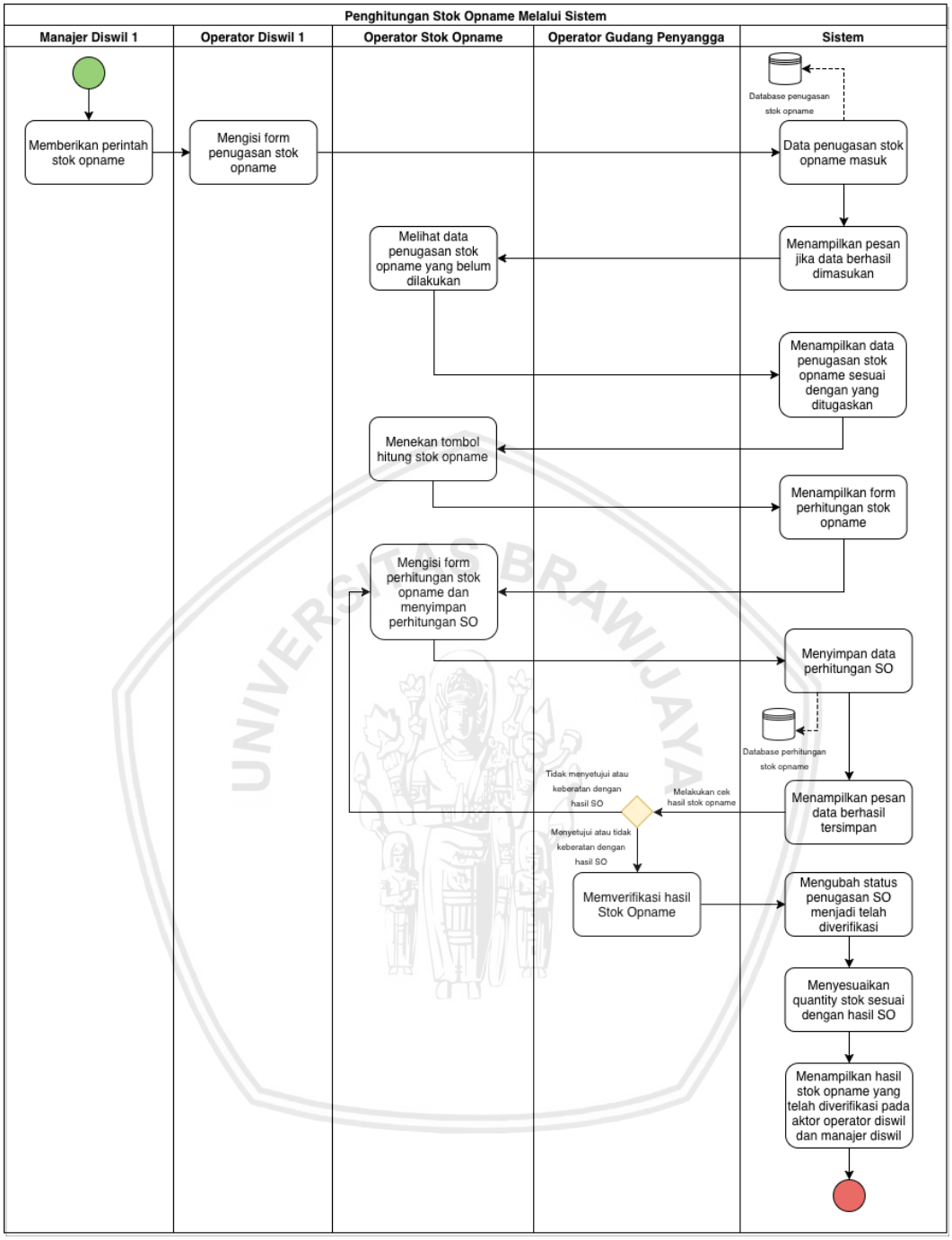
Gambar 4.1 Proses Bisnis Pelaporan Pupuk Keluar-Masuk Gudang Penyangga Yang Sedang Digunakan

Pada gambar 4.2 merupakan proses bisnis stok opname yang sedang dilakukan oleh PT Petrokimia Gresik. Pada proses bisnis ini ditemukan kelemahan yaitu pada proses setelah hasil stok opname diverifikasi oleh kepala gudang penyangga karena hasil stok opname masih berupa *print out* dokumen dalam beberapa kasus terjadi kehilangan dokumen berita acara hasil stok opname ini menyebabkan harus kembali ke gudang penyangga untuk dilakukan hitung ulang. Selain itu, masih harus dilakukan penyesuaian pencatatan stok setelah sampai di kantor pusat, proses ini dianggap tidak efisien karena tidak dapat langsung otomatis disesuaikan dengan hasil stok opname. Hal ini dapat terselesaikan dengan sistem yang akan dikembangkan, yang proses bisnis nya dapat dilihat pada gambar 4.3. Proses perhitungan stok opname dapat langsung dilakukan didalam sistem kemudian ketika selesai dapat langsung disesuaikan pada database stok pupuk.

Kemudian pada gambar 4.3 adalah proses bisnis stok opname dengan menggunakan sistem, dengan proses bisnis ini dapat menyelesaikan atau memperbaiki kelemahan yang ada pada proses yang lama. Salah satunya adalah hasil stok opname langsung tersimpan dalam sistem jadi data akan aman begitu juga untuk penyesuaian pencatatan stok akan otomatis langsung ter-*update* pada *database*.

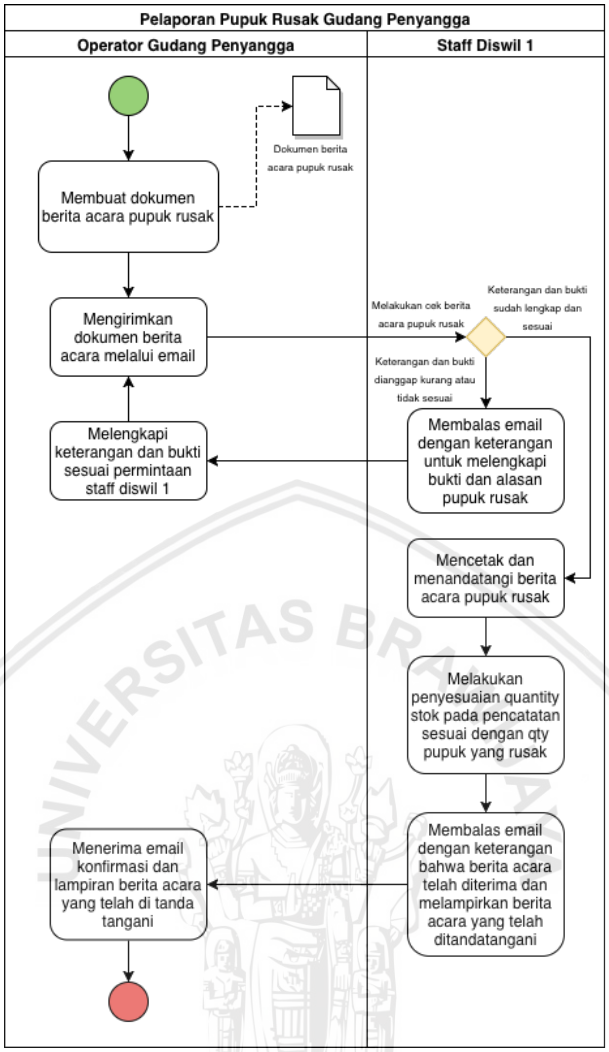


Gambar 4.2 Proses Bisnis Stok Opname Yang Sedang Digunakan

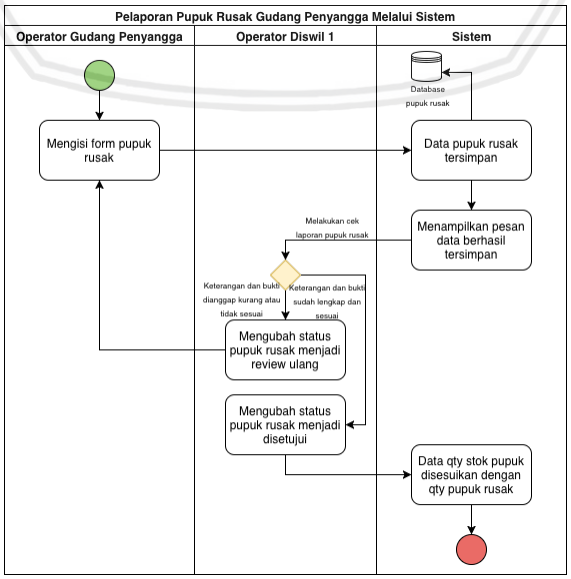


Gambar 4.3 Proses Bisnis Stok Opname Dengan Menggunakan Sistem

Selanjutnya adalah proses bisnis pelaporan pupuk rusak yang sedang digunakan dapat dilihat pada gambar 4.4. Seperti halnya pada kasus pelaporan pupuk keluar-masuk gudang penyangga, pada pelaporan pupuk rusak ini menggunakan cara yaitu dokumen yang dikirimkan melalui *email* untuk kemudian direkap jumlah pupuk yang rusak untuk disesuaikan pada pencatatan stok. Masalah yang dihadapi pun sama adanya data yang tidak terekap. Masalah ini dapat diselesaikan dengan sistem dengan proses bisnis yang dijelaskan pada gambar 4.5 yaitu data pelaporan langsung otomatis ter-*update* dengan data stok.



Gambar 4.4 Proses Bisnis Pelaporan Pupuk Rusak Yang Sedang Digunakan



Gambar 4.5 Proses Bisnis Pelaporan Pupuk Rusak Dengan Menggunakan Sistem

4.2.2 Identifikasi Aktor

Pada tahap pengidentifikasian aktor, dapat dirumuskan aktor-aktor serta peran apa saja yang dapat dilakukan terhadap sistem yang ada. Semua aktor didapatkan dari hasil observasi maupun wawancara yang telah dilakukan. Menurut hasil wawancara dan observasi, pada sistem ini terdapat delapan peran aktor yaitu Operator Diswil 1, Operator Gudang Penyangga, Operator Rekanan Transportir, Operator Gudang Pusat, Manager Diswil 1, Operator Stok Opname, Operator Rekanan Gudang Penyangga, dan *Guest*. Dijelaskna pada Tabel 4.1 hasil dari penemuan aktor yang terdiri dari kolom no, nama aktor, dan deskripsi yang menjelaskan kegiatan dari aktor.

Tabel 4.1 Identifikasi Aktor

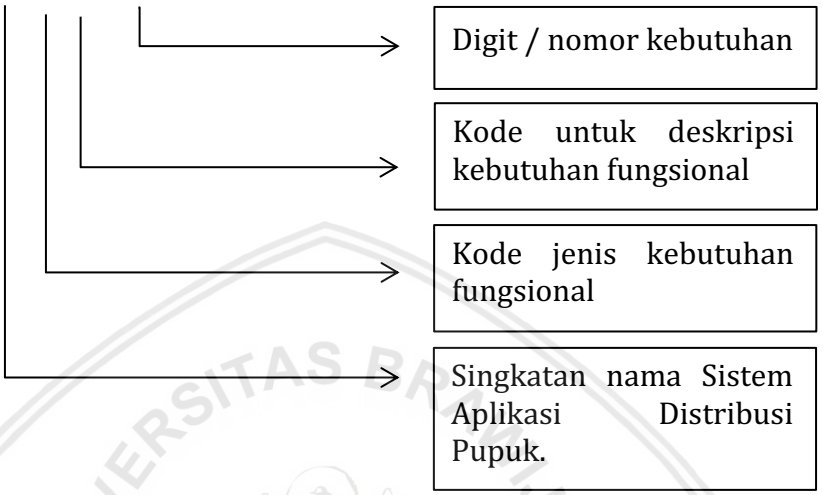
No.	Nama Aktor	Deskripsi
1.	Operator Diswil 1	Operator Diswil 1 adalah aktor yang merupakan karyawan Departemen Distribusi Wilayah 1 yang memiliki tanggung jawab penuh atas proses distribusi dan manajemen stok pupuk.
2.	Operator Gudang Penyangga	Operator Gudang Penyangga adalah aktor yang merupakan rekanan dari PT Petrokimia Gresik yang bertugas dan bertanggung jawab mengelola suatu Gudang penyangga.
3.	Operator Rekanan Transportir	Operator Rekanan Transportir adalah aktor yang bertanggung jawab untuk menyediakan transportasi guna mengambil pupuk dari gudang pusat untuk diantarkan ke gudang penyangga.
4.	Operator Gudang Pusat	Operator Gudang Pusat adalah aktor yang bertanggung jawab untuk menyiapkan pupuk yang akan diambil oleh Rekanan Transportir dan menerima pupuk rusak yang dikirim dari gudang penyangga.
5.	Manajer Diswil 1	Manajer Diswil 1 adalah aktor yang bertanggung jawab untuk memantau proses distribusi.
6.	Operator Stok Opname	Operator Stok Opname adalah aktor yang bertanggung jawab melakukan proses stok opname.
7.	Operator Rekanan Pengelola Gudang Penyangga	Operator Rekanan Pengelola Gudang Penyangga adalah aktor yang bertanggung jawab mengawasi kinerja operator gudang penyangga yang mereka kelola.
8.	*Guest*	Aktor yang mengakses sistem tetapi belum melakukan proses login.

4.3 Aturan Penomoran

Kebutuhan Fungsional:

Contoh: SADP_F_101: Representasi kebutuhan fungsional sistem aplikasi SADP dengan nomor 101.

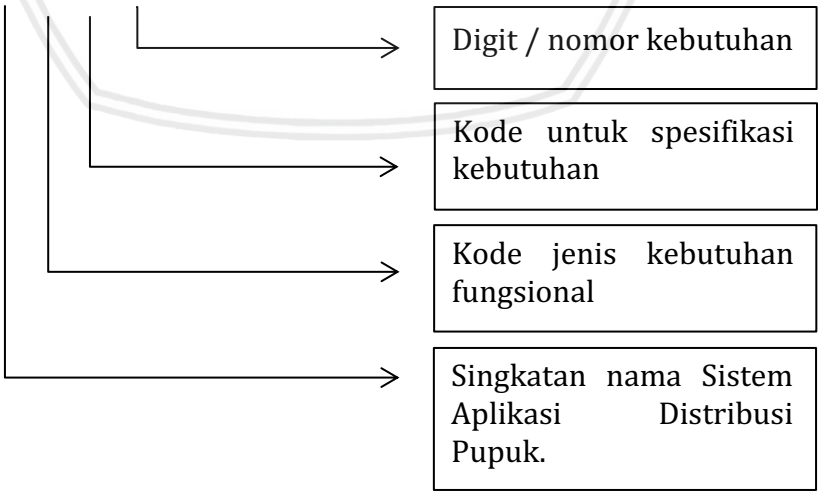
Kode: SADP _F_D _101



Spesifikasi Kebutuhan:

Contoh: SADP_F_S_101: Representasi spesifikasi kebutuhan fungsional sistem aplikasi SADP dengan nomor 101.

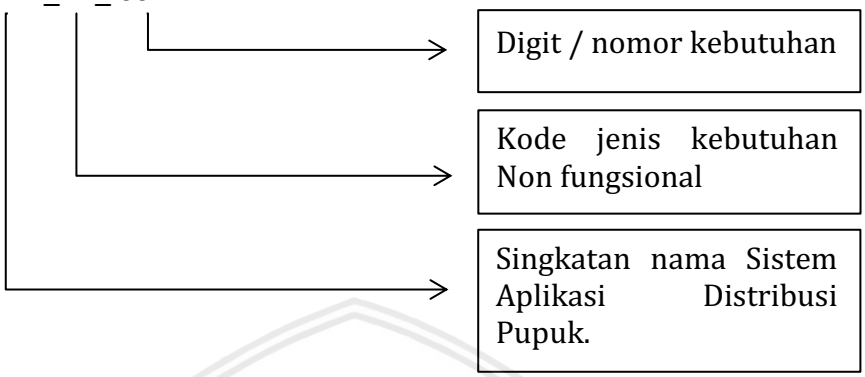
Kode: SADP_F_S _101



Kebutuhan Non Fungsional :

Contoh: SADP_NF_001: Representasi kebutuhan non fungsional sistem aplikasi SADP dengan nomor 001.

Kode: SADP_NF_ 001



4.4 Kebutuhan Fungsional

Kebutuhan fungsional yakni kebutuhan primer yang wajib disediakan dalam sistem sehingga dapat diakses oleh pengguna. Dari hasil elisitasi kebutuhan yang dilakukan pada tahap analisis kebutuhan, diperoleh sebanyakan 62 kebutuhan fungsional yang dibutuhkan oleh pengguna. Kebutuhan fungsional dari sistem ini dijabarkan dalam tabel 4.2 yang terdiri dari aktor yang bersangkutan, nama fungsi, kode kebutuhan dan deskripsi kebutuhan (Deskripsi dan Spesifikasi).

Tabel 4.2 Kebutuhan Fungsional

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
1.	Operator Diswil 1	Melihat Jenis Pupuk	SADP_F_D_001	Sistem sepatutnya bisa menyediakan fungsi untuk aktor dapat melihat data jenis pupuk pada sistem.
			SADP_F_S_001	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data jenis pupuk yaitu ID, nama pupuk dan kemasan (kg).
2.	Operator Diswil 1	Menambah Jenis Pupuk	SADP_F_D_002	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah jenis pupuk.
			SADP_F_S_002	Sistem sepatutnya bisa menyediakan fungsi

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				untuk aktor menambah jenis pupuk dengan data ID, nama pupuk dan kemasan pupuk (kg)
3.	Operator Diswil 1	Mengubah Jenis Pupuk	SADP_F_D_003	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data jenis pupuk
			SADP_F_S_003	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data jenis pupuk yaitu ID, nama pupuk dan kemasan pupuk (kg)
4.	Operator Diswil 1	Melihat Rekanan Transportir	SADP_F_D_004	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat rekanan transportir
			SADP_F_S_004	Sistem harus dapat menyediakan layanan untuk aktor melihat rekanan transportir yaitu ID, nama perusahaan, username, daftar nomor polisi yang telah didaftarkan.
5.	Operator Diswil 1	Menambah Rekanan Transportir	SADP_F_D_005	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah rekanan transportir.
			SADP_F_S_005	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah rekanan transportir yaitu ID, nama perusahaan, username, password, konfirmasi password,

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				dan daftar nomor polisi yang telah didaftarkan.
6.	Operator Diswil 1	Mengubah Rekanan Transportir	SADP_F_D_006	Sistem sepatutnya bisa menyediakan fungsi untuk aktor data rekanan transportir
			SADP_F_S_006	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data rekanan transportir yaitu ID, nama perusahaan, username, password, konfirmasi password, dan daftar nomor polisi yang telah didaftarkan.
7.	Operator Diswil 1	Melihat Batas Stok Kritis	SADP_F_D_007	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data batas stok kritis.
			SADP_F_S_007	Sistem sepatutnya bisa menyediakan fungsi untuk aktor dapat melihat data batas stok kritis yaitu ID, jenis pupuk, daerah dan qty (ton).
8.	Operator Diswil 1	Menambah Batas Stok Kritis	SADP_F_D_008	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data batas stok kritis.
			SADP_F_S_008	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data batas stok kritis yang terdiri dari ID, jenis pupuk, daerah dan qty (ton).
9.			SADP_F_D_009	Sistem sepatutnya bisa menyediakan fungsi

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
	Operator Diswil 1	Mengubah Batas Stok Kritis		untuk aktor mengubah data batas stok kritis.
			SADP_F_S_009	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data batas stok kritis. Yang terdiri dari ID, jenis pupuk, daerah dan qty (ton).
10.	Operator Diswil 1	Melihat Rekanan Pengelola Gudang Penyangga	SADP_F_D_010	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat rekanan pengelola Gudang penyangga.
			SADP_F_S_010	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat rekanan pengelola Gudang penyangga yang terdiri dari ID, nama perusahaan, alamat, username
11.	Operator Diswil 1	Menambah Rekanan Pengelola Gudang Penyangga	SADP_F_D_011	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data rekanan pengelola Gudang penyangga.
			SADP_F_S_011	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data rekanan pengelola Gudang penyangga yang terdiri dari ID, nama perusahaan, alamat kantor, username, password dan konfirmasi password.

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
12.	Operator Diswil 1	Mengubah Rekanan Pengelola Gudang Penyangga	SADP_F_D_012	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data rekanan pengelola Gudang penyangga.
			SADP_F_S_012	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data rekanan pengelola Gudang penyangga yang terdiri dari ID, nama perusahaan, alamat kantor, username, password dan konfirmasi password.
13.	Operator Diswil 1	Melihat Gudang Penyangga	SADP_F_D_013	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data Gudang penyangga.
			SADP_F_S_013	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data Gudang penyangga yang terdiri dari ID, nama perusahaan, nama gudang, nama transportir, kepala gudang, daerah, alamat, kontak, daya tampung, username.
14.	Operator Diswil 1	Menambah Gudang Penyangga	SADP_F_D_014	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data Gudang penyangga.
			SADP_F_S_014	Sistem sepatutnya bisa menyediakan fungsi untuk aktor

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				menambah data Gudang penyangga yang terdiri dari ID, nama perusahaan, nama gudang, nama transportir, kepala gudang, daerah, alamat, kontak, daya tampung, username, password dan konfirmasi password.
15.	Operator Diswil 1	Mengubah Gudang Penyangga	SADP_F_D_015	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data Gudang penyangga.
			SADP_F_S_015	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data Gudang penyangga yang terdiri dari ID, nama perusahaan, nama gudang, nama transportir, kepala gudang, daerah, alamat, kontak, daya tampung, username, password dan konfirmasi password.
16.	Operator Diswil 1	Melihat Petugas Stok Opname	SADP_F_D_016	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data Petugas Stok Opname.
			SADP_F_S_016	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data Petugas Stok Opname yang terdiri dari ID, nama tim, username, nama

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				petugas 1, nik petugas 1, nama petugas 2, nik petugas 2.
17.	Operator Diswil 1	Menambah Petugas Stok Opname	SADP_F_D_017	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data Petugas Stok Opname.
			SADP_F_S_017	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data Petugas Stok Opname yang terdiri dari ID, nama tim, username, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2, password dan konfirmasi password.
18.	Operator Diswil 1	Mengubah Data Petugas Stok Opname	SADP_F_D_018	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data Petugas Stok Opname.
			SADP_F_S_018	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengubah data Petugas Stok Opname yang terdiri dari ID, nama tim, username, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2, password dan konfirmasi password.
19.	Operator Diswil 1		SADP_F_D_019	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
		Membuat Penugasan Stok Opname		penugasan stok opname.
			SADP_F_S_019	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat penugasan stok opname dengan rincian data yaitu ID, nama tim, administrasi, deadline, daerah, pilihan gudang sesuai pilihan daerah.
20.	Operator Diswil 1	Melihat Penugasan Stok Opname (Operator Diswil 1)	SADP_F_D_020	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname.
			SADP_F_S_020	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname dengan rincian data yaitu ID, nama tim, administrasi, gudang, deadline, dan status
21.	Operator Diswil 1	Membuat Permintaan Pengiriman Pupuk	SADP_F_D_021	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat permintaan pengiriman pupuk.
			SADP_F_S_021	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat permintaan pengiriman pupuk dengan rincian data yaitu ID, jenis pupuk, deadline berangkat, tujuan, transportir, dan qty (ton).

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
22.	Operator Diswil 1	Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1)	SADP_F_D_022	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat permintaan pengiriman pupuk.
			SADP_F_S_022	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat permintaan pengiriman pupuk dengan rincian data yaitu id, pupuk, kemasan(kg), tujuan, transportir, deadline berangkat, quantity (ton), status.
23.	Operator Diswil 1	Melihat Detail Permintaan Pengiriman Pupuk	SADP_F_D_023	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat detail permintaan pupuk
			SADP_F_S_023	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat detail permintaan pupuk dengan rincian data yaitu ID, jenis pupuk, deadline berangkat, qty (ton), transportir, tujuan. serta dengan rincian data pengiriman perkendaraan yang terdiri dari jenis pupuk, deadline berangkat, qty(ton), transportir, tujuan, ID, nomor plat, supir, qty (ton), qty berangkat (ton), qty sampai (ton), tanggal berangkat, tanggal sampai, status.

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
24.	Operator Rekanan Transportir	Melihat Status Pengiriman Pupuk	SADP_F_D_024	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat status pengiriman pupuk.
			SADP_F_D_024	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat status pengiriman pupuk dengan rincian data yaitu ID, tgl berangkat, tgl sampai, supir, nopol, pupuk, qty rencana(ton), qty berangkat(ton), qty sampai(ton), tujuan, deadline berangkat, status.
25.	Operator Rekanan Transportir	Menanggapi Permintaan Pengiriman Pupuk	SADP_F_D_025	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menanggapi permintaan pengiriman pupuk.
			SADP_F_S_025	Sistem sepatutnya bisa menyediakan fungsi untuk menanggapi permintaan pengiriman pupuk dengan data yang harus dimasukan adalah nama supir, nomor polisi dan qty angkut tiap kendaraan.
26.	Operator Rekanan Gudang Penyangga	Melihat Selisih Stok Opname di Gudang Penyangga	SADP_F_D_026	Sistem sepatutnya bisa menyediakan fungsi untuk aktor Melihat Selisih Stok Opname di Gudang Penyangga.
			SADP_F_S_026	Sistem sepatutnya bisa menyediakan fungsi untuk aktor Melihat

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				Selisih Stok Opname di Gudang Penyangga dengan rincian data yaitu ID stok opname, gudang penyangga, tanggal, jenis pupuk, kemasan, qty(ton).
27.	Operator Rekanan Gudang Penyangga	Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga)	SADP_F_D_027	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname yang sudah divalidasi.
			SADP_F_S_027	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname yang sudah divalidasi dengan rincian data yaitu kepala gudang, petugas 1, petugas 2, administrasi, tanggal, jam mulai, jam selesai, status, pupuk, hasil ton fisik, data admin gudang, selisih(ton).
28.	Operator Stok Opname Diswil 1	Membuat Perhitungan Stok Opname	SADP_F_D_028	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat perhitungan stok opname.
			SADP_F_S_028	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat perhitungan stok opname dengan rincian data pada daftar stok opname yaitu id, id gudang, administrasi, gudang penyangga, kepala gudang, deadline, dikerjakan, jam mulai,

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				jam selesai, status. rincian data untuk hitungan perpupuk yaitu area, kunci, panjang, lebar, tinggi, jumlah bag, jumlah ton, lebihan bag, lebihan ton, total bag, total ton.
29.	Operator Stok Opname Diswil 1	Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1)	SADP_F_D_029	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname.
			SADP_F_S_029	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname dengan rincian data yaitu kepala gudang, petugas 1, petugas 2, administrasi, tanggal, jam mulai, jam selesai, status, pupuk, hasil ton fisik, data admin gudang, selisih(ton).
30.	Operator Diswil 1, Manager Diswil 1	Melihat Peta Kondisi Stok Daerah	SADP_F_D_030	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat peta kondisi stok daerah.
			SADP_F_S_030	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat peta kondisi stok daerah dengan rincian data yaitu Peta Google Maps yang dilengkapi dengan layering warna setiap daerah yang mengindikasikan kondisi stok (aman/kurang)

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				dilengkapi dengan detail data meliputi pupuk, kemasan (kg), qty(ton), kritis(ton) dan detail pergudang yang terdiri dari gudang, pupuk, kemasan(kg), qty(ton).
31.	Operator Gudang Penyangga	Membuat Laporan Pupuk Rusak	SADP_F_D_030	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat laporan pupuk rusak.
			SADP_F_S_030	Sistem sepatutnya bisa menyediakan fungsi untuk aktor membuat laporan pupuk rusak dengan rincian data yaitu pupuk, qty rusak (ton), foto pupuk rusak, dan alasan rusak.
32.	Operator Diswil 1	Melihat Laporan Pupuk Rusak (Operator Diswil 1)	SADP_F_D_032	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat laporan pupuk rusak.
			SADP_F_S_032	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat laporan pupuk rusak dengan rincian data yaitu id, tanggal, jenis pupuk, kemasan(kg), gudang penyangga, quantity (ton), alasan, status dan foto pupuk rusak.
33.	Operator Gudang Penyangga.	Menerima Pupuk Masuk Gudang Penyangga	SADP_F_D_033	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menerima pupuk masuk gudang penyangga.

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
			SADP_F_S_033	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menerima pupuk masuk dengan rincian data yaitu qty dari pusat (ton), qty diterima (ton).
34.	Operator Gudang Penyangga.	Menambah Pupuk Keluar Gudang Penyangga.	SADP_F_D_034	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data pupuk keluar Gudang penyangga.
			SADP_F_S_034	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menambah data pupuk keluar Gudang penyangga dengan rincian data yaitu pupuk, distributor, nomor polisi, qty keluar (ton).
35.	Operator Gudang Penyangga	Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga)	SADP_F_D_035	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data pupuk keluar Gudang penyangga.
			SADP_F_S_035	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data pupuk keluar Gudang penyangga dengan rincian data yaitu id, id stok, distributor, nomor polisi, pupuk, kemasan (kg), quantity (ton), tanggal dan jam pupuk keluar.
36.		Melihat Pupuk Masuk Gudang	SADP_F_D_036	Sistem sepatutnya bisa menyediakan fungsi

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
	Operator Gudang Penyangga	Penyangga (Operator Gudang Penyangga)		untuk aktor melihat pupuk masuk Gudang penyengga
			SADP_F_S_036	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk masuk Gudang penyengga dengan rincian data yaitu id, tanggal berangkat, tanggal sampai, supir, nopol, pupuk, quantity rencana (ton), quantity berangkat (ton), quantity sampai (ton), tujuan, deadline, status
37.	Operator Gudang Penyangga	Validasi Hasil Stok Opname	SADP_F_D_037	Sistem sepatutnya bisa menyediakan fungsi untuk aktor validasi hasil stok opname.
			SADP_F_S_037	Sistem sepatutnya bisa menyediakan fungsi untuk aktor validasi hasil stok opname dengan menyediakan *button* untuk melihat hasil stok opname, validasi dan proses kembali.
38.	Operator Gudang Pusat	Memproses Pemberian Pupuk	SADP_F_D_038	Sistem sepatutnya bisa menyediakan fungsi untuk aktor memproses pemberian pupuk.
			SADP_F_S_038	Sistem sepatutnya bisa menyediakan fungsi untuk aktor memproses pemberian pupuk melalui *button* proses dengan rincian

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				data qty ton yang harus diberikan.
39.	Operator Rekanan Transportir	Melihat Pupuk Hilang Saat Pengiriman	SADP_F_D_039	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk hilang saat pengiriman.
			SADP_F_S_039	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk hilang saat pengiriman dengan rincian data yaitu ID, jenis pupuk, kemasan (kg), qty yang hilang (ton), nama supir, nopol, tujuan, tanggal berangkat, tanggal sampai.
40.	Operator Gudang Pusat	Menerima Pengembalian Pupuk Rusak	SADP_F_D_040	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menerima pengembalian pupuk saat dikembalikan dari gudang penyangga
			SADP_F_S_040	Sistem sepatutnya bisa menyediakan fungsi untuk aktor menerima pengembalian pupuk rusak dengan menyediakan terima untuk melakukan proses menerima pengembalian pupuk dari gudang penyangga.
41.	Operator Rekanan Transportir	Mencetak Surat Jalan	SADP_F_D_041	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mencetak surat jalan untuk pengantar pupuk.

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
			SADP_F_S_041	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mencetak surat jalan untuk pengantar pupuk dengan rincian data yaitu nomor surat jalan, tujuan, tanggal kirim, pupuk yang diantar dengan jumlah ton yang diangkut.
42.	Operator Diswil 1	Validasi Pupuk Rusak	SADP_F_D_042	Sistem sepatutnya bisa menyediakan fungsi untuk aktor validasi laporan pupuk rusak dari gudang penyangga.
			SADP_F_S_042	Sistem sepatutnya bisa menyediakan fungsi untuk aktor validasi laporan pupuk rusak dari gudang penyangga dengan menyediakan dua pilihan yaitu setujui dan review ulang.
43.	Operator Gudang Penyangga	Mengirim Pupuk Rusak	SADP_F_D_043	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengirim pupuk rusak ke gudang pusat setelah mendapat validasi dari operator diswil 1
			SADP_F_S_043	Sistem sepatutnya bisa menyediakan fungsi untuk aktor mengirim pupuk rusak ke gudang pusat dengan rincian data yitu nama supir, plat nomor, daftar pupuk rusak yang dikirim.

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
44.	Guest	Login	SADP_F_D_044	Sistem sepatutnya bisa menyediakan fasilitas login untuk *guest* sehingga dapat dikenali sistem sebagai manajer diswil 1, operator diswil 1, operator stok opname, operator gudang pusat, operator gudang penyangga, operator rekanan gudang penyangga, dan operator transportir.
			SADP_F_S_044	Sistem harus dapat menyediakan formulir login berupa username dan password.
45.	Guest	Logout	SADP_F_D_045	Sistem sepatutnya bisa menyediakan fungsi logout kepada aktor untuk dapat keluar dari sistem.
			SADP_F_S_045	Sistem sepatutnya bisa menyediakan fasilitas logout kepada aktor untuk dapat keluar dari sistem.
46.	Operator Gudang Pusat	Melihat Daftar Pemberian Pupuk	SADP_F_D_046	Sistem sepatutnya bisa menyediakan fungsi untuk aktor Melihat Daftar Pemberian Pupuk.
			SADP_F_S_046	Sistem sepatutnya bisa menyediakan fungsi untuk aktor Melihat Daftar Pemberian Pupuk dengan rincian data yaitu id, tanggal berangkat, tanggal sampai, supir, nomor polisi, pupuk, qty

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				rencana (ton), qty berangkat (ton), qty sampai (ton), tujuan, deadline, status.
47.	Operator Gudang Pusat	Melihat Pengiriman Pupuk Rusak	SADP_F_D_047	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data pengiriman pupuk rusak.
			SADP_F_S_047	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data pengiriman pupuk rusak dengan rincian data yaitu id, nomor polisi, supir, tanggal berangkat, tanggal sampai, status, dan daftar pupuk yang dikirim.
48.	Operator Rekanan Gudang Penyangga	Melihat Gudang Penyangga yang dikelola	SADP_F_D_048	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data Gudang penyangga yang dikelola sesuai dengan aktor operator rekanan gudang penyangga tersebut.
			SADP_F_S_048	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data Gudang penyangga yang dikelola dengan terdiri dari ID, nama gudang, nama transportir, kepala gudang, daerah, alamat, kontak, daya tampung, username.
49.	Operator Rekanan	Melihat Penugasan Stok	SADP_F_D_049	Sistem sepatutnya bisa menyediakan fungsi

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
	Gudang Penyangga	Opname (Operator Rekanan Gudang Penyangga)		untuk aktor melihat penugasan stok opname dengan tujuan gudang penyangga yang dikelola oleh aktor.
			SADP_F_S_049	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname dengan tujuan gudang penyangga yang dikelola oleh aktor dengan rincian data yaitu ID, administrasi, gudang penyangga, kepala gudang, deadline, tgl dikerjakan, jam mulai, jam selesai, dan status.
50.	Manajer Diswil 1	Melihat Penugasan Stok Opname (Manajer Diswil 1)	SADP_F_D_050	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname.
			SADP_F_S_050	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname dengan rincian data yaitu ID, ID gudang, administrasi, gudang penyangga, kepala gudang, deadline, tgl dikerjakan, dan status
51.	Operator Stok Opname Diswil 1	Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1)	SADP_F_D_051	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname yang ditugaskan pada aktor.

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
			SADP_F_S_051	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat penugasan stok opname yang ditugaskan pada aktor dengan rincian data yaitu ID, ID gudang, administrasi, gudang penyangga, kepala gudang, deadline, dikerjakan, jam mulai, jam selesai, dan status
52.	Operator Stok Opname Diswil 1	Melihat Riwayat Penugasan Stok Opname	SADP_F_D_052	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat riwayat penugasan stok opname yang ditugaskan pada aktor.
			SADP_F_S_052	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat riwayat penugasan stok opname yang ditugaskan pada aktor dengan rincian data yaitu ID, administrasi, gudang penyangga, deadline, status, dan masa berlaku.
53.	Operator Diswil 1	Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1)	SADP_F_D_053	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk masuk ke gudang penyangga dalam sistem.
			SADP_F_S_054	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk masuk ke gudang penyangga dalam sistem dengan

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				rincian data yaitu id, gudang penyangga, transportir, nomor polisi, pupuk, kemasan (kg), quantity (ton), tanggal sampai.
54.	Operator Diswil 1	Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1)	SADP_F_D_054	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk keluar dari gudang penyangga.
			SADP_F_S_054	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat pupuk keluar dari gudang penyangga dengan rincian data yaitu id, id stok, gudang penyangga, distributor, nomor polisi, pupuk, kemasan, qty (ton), tanggal dan jam pupuk keluar.
55.	Operator Gudang Penyangga	Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga)	SADP_F_D_055	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat laporan pupuk rusak di gudang penyangga yang dikelola aktor pada sistem.
			SADP_F_S_055	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat laporan pupuk rusak di gudang penyangga yang dikelola aktor pada sistem dengan rincian data yaitu id, jenis pupuk, kemasan (kg), alasan, quantity (ton), tanggal, status dan foto pupuk rusak.

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
56.	Manajer Diswil 1	Melihat Laporan Pupuk Rusak (Manajer Diswil 1)	SADP_F_D_056	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat laporan pupuk rusak pada sistem.
			SADP_F_S_056	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat laporan pupuk rusak pada sistem dengan rincian data yaitu id, tanggal, jenis pupuk, kemasan, gudang penyangga, quantity (ton), alasan dan foto pupuk rusak.
57.	Operator Gudang Penyangga	Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga)	SADP_F_D_057	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data pengiriman pupuk rusak.
			SADP_F_S_057	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat data pengiriman pupuk rusak dengan rincian data yaitu id, nomor polisi, supir, tanggal berangkat, tanggal sampai, status, dan daftar pupuk yang dikirim.
58.	Operator Rekanan Transportir	Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir)	SADP_F_D_058	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat permintaan pengiriman pupuk.
			SADP_F_S_058	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat permintaan pengiriman pupuk

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
				dengan rincian data yaitu id, pupuk, kemasan(kg), tujuan, deadline berangkat, quantity (ton)
59.	Operator Gudang Penyangga	Melihat Hasil Stok Opname (Operator Gudang Penyangga)	SADP_F_D_059	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname.
			SADP_F_S_059	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname dengan rincian data yaitu kepala gudang, petugas 1, petugas 2, administrasi, tanggal, jam mulai, jam selesai, status, pupuk, hasil ton fisik, data admin gudang, selisih(ton).
60.	Manajer Diswil 1	Melihat Hasil Stok Opname yang Sudah Divalidasi (Manajer Diswil 1)	SADP_F_D_060	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname yang sudah divalidasi.
			SADP_F_S_060	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname yang sudah divalidasi dengan rincian data yaitu kepala gudang, petugas 1, petugas 2, administrasi, tanggal, jam mulai, jam selesai, status, pupuk, hasil ton fisik, data admin gudang, selisih(ton).
61.	Operator Diswil 1	Melihat Hasil Stok Opname yang Sudah	SADP_F_D_061	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat

No	Aktor	Nama Fungsi	Kode	Deskripsi Kebutuhan
		Divalidasi (Operator Diswil 1)		hasil stok opname yang sudah divalidasi.
			SADP_F_S_061	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat hasil stok opname yang sudah divalidasi dengan rincian data yaitu kepala gudang, petugas 1, petugas 2, administrasi, tanggal, jam mulai, jam selesai, status, pupuk, hasil ton fisik, data admin gudang, selisih(ton).
62.	Operator Gudang Penyangga	Melihat Stok Pupuk	SADP_F_D_062	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat stok pupuk yang tersedia di gudang penyangga yang dikelola oleh aktor
			SADP_F_S_062	Sistem sepatutnya bisa menyediakan fungsi untuk aktor melihat stok pupuk yang tersedia di gudang penyangga yang dikelola oleh aktor dengan rincian data yaitu id, pupuk, kemasan (kg), quantity (ton).

4.5 Kebutuhan Non-Fungsional

Kebutuhaan yang bersifat non-fungsional yakni kebutuhan yang tidak ada hubungannya dengan fungsi atau manfaat secara langsung dari sebuah perangkat lunak. Kebutuhan ini lebih pada aspek kemudahan, kecepatan dan efisiensi sumber daya.

Kebutuhan kompatibilitas ini didapatkan dari hasil wawancara dan observasi, didapatkan di lapangan bahwa aktor yang terlibat dalam sistem ini menggunakan

web browser yang berbeda-beda untuk itu sistem juga harus dapat berjalan dengan baik pada peramban yang digunakan oleh aktor-aktor yang ada. Beberapa peramban yang digunakan aktor dan harus dapat digunakan untuk menjalankan sistem ini adalah *Google Chrome, Mozila Firefox*, dan *Microsoft Edge*. Untuk melihat kebutuhan non-fungsional yang sepatutnya dipunyai Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web dijelaskan dalam Tabel 4.3

Tabel 4.3 Kebutuhan Non-Fungsional

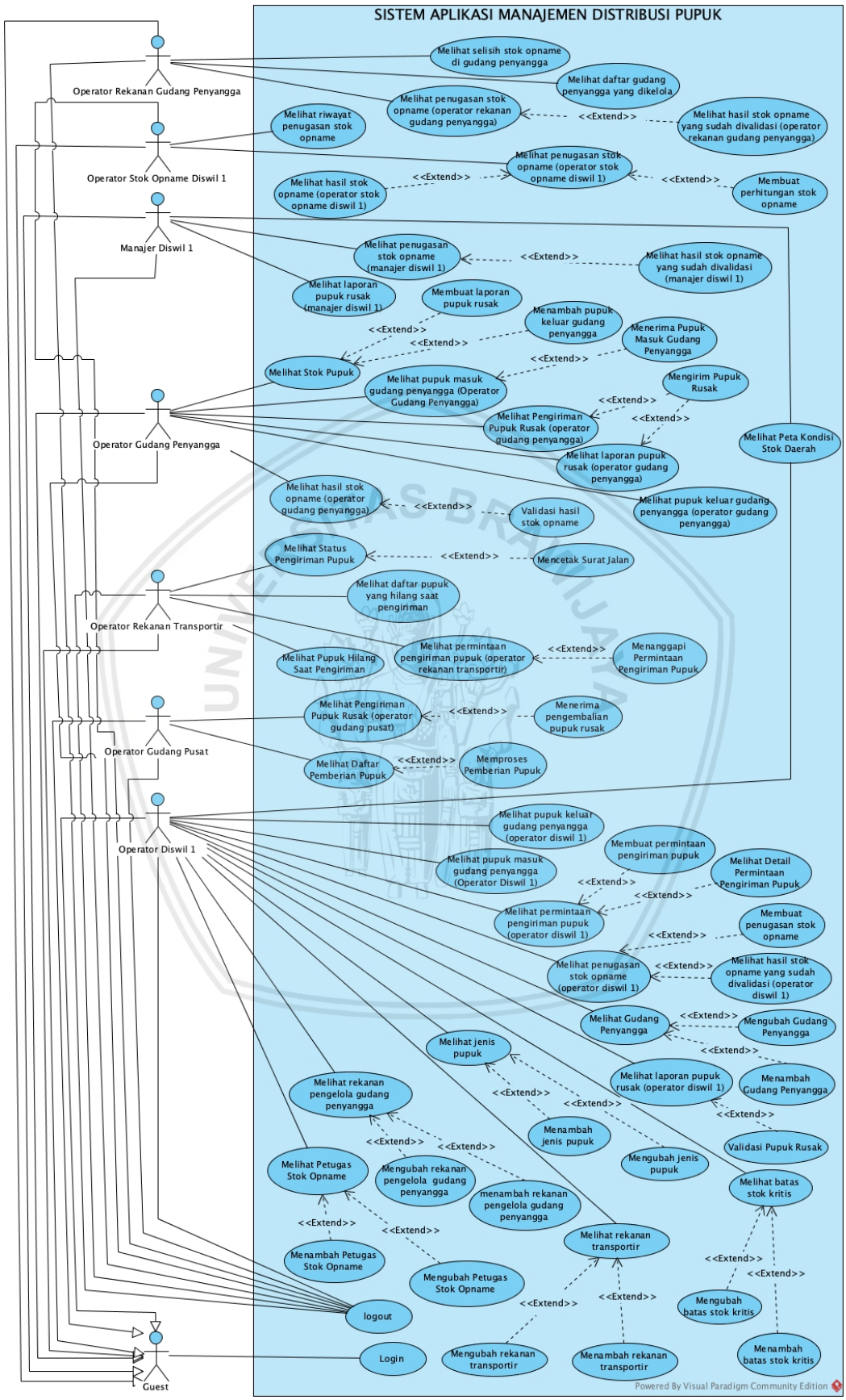
No	Kode	Parameter	Deskripsi Kebutuhan
1.	SADP_NF_01	Kompatibilitas	Sistem harus bisa dijalankan di web peramban yang biasa digunakan oleh aktor-aktor yang ada yaitu Google Chrome, Mozilla Firefox, dan Microsoft Edge.

4.6 Diagram Use Case

Diagram *use case* yakni diagram yang dapat dipergunakan untuk membuat model *behavior*/perilaku dari aktor terhadap *software*/sistem, mengambarkan apa saja yang dapat dilakukan oleh aktor terhadapat sistem. Aktor yang ada pada Sistem Aplikasi Manajemen Distribusi Pupuk ini ada delapan aktor yaitu operator rekanan gudang penyangga, operator stok opname diswil 1, manajer diswil 1, operator gudang penyangga, operator rekanan transportir, operator gudang pusat, operator diswil 1, dan *guest*. Kemampuan aktor untuk bisa apa saja dengan sistem dijabarkan pada Gambar 4.6.

4.7 Skenario Use Case

Pada subbab skenario *use case* dijabarkan cara dalam mempergunakan sistem. Jumlah skenario yang ada sama dengan jumlah kebutuhan yang telah didapatkan yaitu 62. Bagian-bagian yang akan dijelaskan pada sebuah skenario use case ialah Kolom aktor digunakan untuk menuliskan aktor apa saja yang terdapat pada *use case* tersebut, kolom *objective* adalah untuk mendeskripsikan tujuan dari *use case*, kolom *pre-conditiion* untuk mendeskripsikan syarat untuk dimulainya *use case*, kolom *main flow* untuk menjelaskan alur utama kegiatan atau langkah dalam *use case*, kolom *alternative flow* digunakan untuk menjelaskan alur alternatif dalam *use case*, dan kolom *post-condition* digunakan untuk mendeskripsikan alur atau hasil akhir pada *use case*. Penjabarannya dapat merujuk pada tabel 4.4 sampai dengan tabel 4.65.



Gambar 4.6 Diagram Use Case

Tabel 4.4 Skenario Use Case Melihat Jenis Pupuk

Aktor	Operator Diswil 1
Objective	Aktor melihat jenis pupuk yang telah terdapat didalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor menekan kategori menu pupuk. 2. Sistem memperlihatkan isi kategori menu pupuk. 3. Aktor menekan menu jenis pupuk.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu jenis pupuk dan memperlihatkan data tabel jenis pupuk.

Tabel 4.5 Skenario Use Case Manambah Jenis Pupuk

Aktor	Operator Diswil 1
Objective	Aktor menambah jenis pupuk yang telah terdapat didalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu jenis pupuk.
Main Flow	1. Aktor menekan *button* tambah 2. Sistem memperlihatkan form tambah pupuk. 3. Aktor mengisi data tambah pupuk yaitu nama pupuk, dan kemasan pupuk (kg). 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data jenis pupuk ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka sistem akan memperlihatkan pesan "Semua field harus disini".
Post-Condition	Sistem memperlihatkan pesan data berhasil tesimpan dan jenis pupuk telah tertambah pada tabel jenis pupuk.

Tabel 4.6 Skenario Use Case Mengubah Jenis Pupuk

Aktor	Operator Diswil 1

Objective	Aktor menambah mengubah data jenis pupuk yang telah terdapat didalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu jenis pupuk.
Main Flow	1. Aktor menekan *button* edit pada data pupuk yang ingin diubah. 2. Sistem memperlihatkan form ubah pupuk. 3. Aktor mengisi data pupuk yang ingin diubah yaitu nama pupuk, dan kemasan pupuk (kg). 4. Aktor menekan *button* simpan. 5. Sistem mengubah data jenis pupuk ke dalam basis data aplikasi/sistem.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil disimpan, dan data jenis pupuk telah terubah pada tabel jenis pupuk.

Tabel 4.7 Skenario Use Case Melihat Rekanan Transportir

Aktor	Operator Diswil 1
Objective	Aktor melihat rekanan transportir yang telah terdapat didalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor menekan menu rekanan transportir.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu rekanan transportir dan tabel rekanan transportir.

Tabel 4.8 Skenario Use Case Manambah Rekanan Transportir

Aktor	Operator Diswil 1
Objective	Aktor menambah rekanan transportir yang telah terdapat didalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu rekanan transportir.
Main Flow	1. Aktor menekan *button* tambah pada halaman rekanan transportir.

	2. Sistem memperlihatkan form tambah rekanan transportir. 3. Aktor mengisi data yang ada pada form yaitu nama perusahaan, username, password, konfirmasi password, nomor polisi. 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data rekanan transportir ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka sistem akan memperlihatkan pesan "Semua field harus disini".
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan, dan data rekanan transportir telah tertambahkan pada tabel.

Tabel 4.9 Skenario Use Case Mengubah Rekanan Transportir

Aktor	Operator Diswil 1
Objective	Aktor mengubah data rekanan transportir yang telah terdapat didalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu rekanan transportir
Main Flow	1. Aktor menekan *button* edit pada data yang ingin diubah. 2. Sistem memperlihatkan form ubah rekanan transportir. 3. Aktor mengisi data rekanan transportir yang ingin diubah yaitu nama perusahaan, username, password, konfirmasi password, nomor polisi. 4. Aktor menekan *button* simpan. 5. Sistem mengubah data rekanan transportir ke dalam basis data aplikasi/sistem.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan, dan data rekanan transportir telah terubah.

Tabel 4.10 Skenario Use Case Melihat Batas Stok Kritis

Aktor	Operator Diswil 1
Objective	Aktor melihat data batas stok kritis dalam sistem.

Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk. 2. Sistem memperlihatkan daftar menu didalam ketegori menu pupuk. 3. Aktor memilih menu batas stok kritis.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu batas stok kritis dan tabel batas stok kritis.

Tabel 4.11 Skenario Use Case Menambah Batas Stok Kritis

Aktor	Operator Diswil 1
Objective	Aktor menambah data batas stok kritis dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu batas stok kritis.
Main Flow	1. Aktor memilih *button* tambah. 2. Sistem memperlihatkan form tambah stok kritis. 3. Aktor mengisi data tambah batas stok kritis yaitu nama pupuk, daerah, dan qty (ton). 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data batas stok kritis ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka sistem akan memperlihatkan pesan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan data batas stok kritis telah tertambah.

Tabel 4.12 Skenario Use Case Mengubah Batas Stok Kritis

Aktor	Operator Diswil 1
Objective	Aktor mengubah data batas stok kritis dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu batas stok kritis
Main Flow	1. Aktor memilih *button* edit pada data yang ingin diubah. 2. Sistem memperlihatkan form ubah stok kritis. 3. Aktor mengisi data batas stok kritis yang ingin diubah yaitu nama pupuk, daerah, dan qty (ton).

	4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data batas stok kritis ke dalam database.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan, dan data batas stok kritis telah terubah.

Tabel 4.13 Skenario Use Case Melihat Rekanan Pengelola Gudang Penyangga

Aktor	Operator Diswil 1
Objective	Aktor melihat rekanan pengelola gudang penyangga dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu gudang penyangga. 2. Sistem memperlihatkan daftar menu didalam ketegori menu gudang penyangga. 3. Aktor memilih menu rekanan g. penyangga.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu rekanan g. penyangga dan tabel rekanan gudang penyangga.

Tabel 4.14 Skenario Use Case Menambah Rekanan Pengelola Gudang Penyangga

Aktor	Operator Diswil 1
Objective	Aktor menambah rekanan pengelola gudang penyangga dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu rekanan g. penyangga.
Main Flow	1. Aktor memilih *button* tambah 2. Sistem memperlihatkan form tambah rekanan gudang. 3. Aktor mengisi data tambah rekanan pengelola gudang penyangga yaitu nama perusahaan, alamat kantor, username, password dan konfirmasi password. 4. Aktor menekan *button* simpan.

	5. Sistem merekam dan memasukan data rekanan pengelola gudang penyangga ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka akan memperlihatkan pesan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan, dan data rekanan pengelola gudang penyangga telah tertambah.

Tabel 4.15 Skenario Use Case Mengubah Rekanan Pengelola Gudang Penyangga

Aktor	Operator Diswil 1
Objective	Aktor mengubah data rekanan pengelola gudang penyangga dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu rekanan g. penyangga
Main Flow	1. Aktor memilih *button* edit pada data yang ingin diubah. 2. Sistem memperlihatkan form ubah rekanan gudang. 3. Aktor mengisi data pupuk yang ingin diubah yaitu nama perusahaan, alamat kantor, username, password, dan konfirmasi password. 4. Aktor menekan *button* simpan. 5. Sistem mengubah data rekanan pengelola gudang penyangga ke dalam basis data aplikasi/sistem.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tesimpan dan data rekanan pengelola gudang penyangga telah terubah.

Tabel 4.16 Skenario Use Case Melihat Gudang Penyangga

Aktor	Operator Diswil 1
Objective	Aktor melihat data gudang penyangga dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login

Main Flow	1. Aktor memilih kategori menu gudang penyangga. 2. Sistem memperlihatkan daftar menu didalam ketegori menu gudang penyangga. 3. Aktor memilih menu daftar g. penyangga.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman daftar Data Gudang Penyangga dan tabel daftar gudang penyangga.

Tabel 4.17 Skenario Use Case Menambah Gudang Penyangga

Aktor	Operator Diswil 1
Objective	Aktor menambah data gudang penyangga dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu daftar g. penyangga
Main Flow	1. Aktor memilih *button* tambah 2. Sistem memperlihatkan form tambah gudang penyangga. 3. Aktor mengisi data tambah pupuk yaitu nama perusahaan, nama transportir, nama gudang, username, password, konfirmasi password, kepala gudang, daerah, daya tampung, kontak, alamat. 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data gudang penyangga ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka akan memperlihatkan pemberitahuan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan, dan data gudang penyangga telah tertambah.

Tabel 4.18 Skenario Use Case Mengubah Gudang Penyangga

Aktor	Operator Diswil 1
Objective	Aktor mengubah data gudang penyangga pada sistem
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu daftar g. penyangga.
Main Flow	1. Aktor memilih *button* edit pada data yang ingin diubah.

	2. Sistem memperlihatkan form gudang penyangga. 3. Aktor mengisi data pupuk yang ingin diubah yaitu nama perusahaan, nama transportir, nama gudang, username, password, konfirmasi password, kepala gudang, daerah, daya tampung, kontak, alamat. 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data gudang penyangga yang diubah ke dalam basis data aplikasi/sistem.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan, dan data Gudang penyangga telah berhasil diubah

Tabel 4.19 Skenario Use Case Melihat Petugas Stok Opname

Aktor	Operator Diswil 1
Objective	Aktor melihat data petugas stok opname pada sistem
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu stok opname 2. Sistem memperlihatkan daftar menu didalam ketegori menu stok opname 3. Aktor memilih menu petugas stok opname
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu petugas stok opname dan tabel petugas stok opname.

Tabel 4.20 Skenario Use Case Menambah Petugas Stok Opname

Aktor	Operator Diswil 1
Objective	Aktor menambah data petugas stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu petugas stok opname.
Main Flow	1. Aktor memilih *button* tambah 2. Sistem memperlihatkan form tambah petugas stok opname.

	3. Aktor mengisi data tambah stok opname yaitu nama tim, username, password, konfirmasi password, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data petugas stok opname ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka sistem akan memperlihatkan pesan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan data petugas stok opname berhasil ditambahkan.

Tabel 4.21 Skenario Use Case Mengubah Petugas Stok Opname

Aktor	Operator Diswil 1
Objective	Aktor mengubah data petugas stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu petugas stok opname
Main Flow	1. Aktor memilih *button* edit pada data yang ingin diubah. 2. Sistem memperlihatkan form ubah petugas stok opname. 3. Aktor mengisi data petugas stok opname yang ingin diubah yaitu nama tim, username, password, konfirmasi password, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2. 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data petugas stok opname yang diubah ke dalam basis data aplikasi/sistem.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan data petugas stok opname berhasil terubah.

Tabel 4.22 Skenario Use Case Membuat Penugasan Stok Opname

Aktor	Operator Diswil 1
Objective	Aktor membuat penugasan stok opname dalam sistem.

Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu penugasan stok opname.
Main Flow	1. Aktor memilih *button* tambah 2. Sistem memperlihatkan form tambah penugasan 3. Aktor mengisi data tambah pupuk yaitu nama tim, administrasi, deadline, daerah dan pilih gudang. 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data penugasan stok opname ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka akan memperlihatkan pesan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan data penugasan stok opname berhasil dibuat.

Tabel 4.23 Skenario Use Case Melihat Penugasan Stok Opname (Operator Diswil 1)

Aktor	Operator Diswil 1
Objective	Aktor melihat penugasan stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih kategori menu stok oname. 2. Sistem memperlihatkan daftar menu didalam ketegori menu stok opname. 3. Aktor memilih menu penugasan stok opname.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu penugasan stok opname.

Tabel 4.24 Skenario Use Case Membuat Permintaan Pengiriman Pupuk

Aktor	Operator Diswil 1
Objective	Aktor membuat permintaan pengiriman pupuk dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu permintaan pupuk.
Main Flow	1. Aktor memilih *button* tambah

	2. Sistem memperlihatkan form tambah permintaan pupuk. 3. Aktor mengisi data tambah pupuk yaitu jenis pupuk, deadline berangkat, tujuan, transportir, dan qty (ton) 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data permintaan pupuk ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka akan memperlihatkan pemberitahuan untuk mengisi semua form.
Post-Condition	Sisitem memperlihatkan pesan data berhasil tersimpan dan data permintaan pengiriman pupuk berhasil dibuat.

Tabel 4.25 Skenario Use Case Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1)

Aktor	Operator Diswil 1.
Objective	Aktor melihat permintaan pengiriman pupuk dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih kategori menu pupuk. 2. Sistem memperlihatkan daftar menu didalam ketegori menu pupuk. 3. Aktor memilih menu permintaan pupuk.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan daftar permintaan pengiriman pupuk di halaman menu permintaan pupuk.

Tabel 4.26 Skenario Use Case Melihat Detail Permintaan Pengiriman Pupuk

Aktor	Operator Diswil 1.
Objective	Aktor melihat detail permintaan pengiriman pupuk dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu permintaan pupuk.
Main Flow	1. Aktor memilih *button* detail pada permintaan pupuk yang diinginkan.

Alternatif Flow	-
Post-Condition	Sistem memperlihatkan data detail permintaan pupuk yang dipilih pada form detail permintaan pupuk.

Tabel 4.27 Skenario Use Case Melihat Status Pengiriman Pupuk

Aktor	Operator Rekanan Transportir.
Objective	Aktor melihat status pengiriman pupuk pada sistem
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih menu permintaan pupuk.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan tabel status permintaan pupuk pada menu permintaan pupuk.

Tabel 4.28 Skenario Use Case Menanggapi Permintaan Pengiriman Pupuk

Aktor	Operator Rekanan Transportir
Objective	Aktor dapat menanggapi permintaan pengiriman pupuk pada sistem
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu permintaan pupuk.
Main Flow	1. Aktor menekan *button* antar pada tabel permintaan pupuk yang ingin diproses. 2. Sistem memperlihatkan form respon permintaan pupuk. 3. Aktor mengisi data yang diperlukan untuk mananggapi permintaan pupuk yaitu nama supir, nomor polisi, qty. 4. Aktor menekan *button* antar. 5. Sistem mengubah status permintaan pupuk pada database
Alternatif Flow	1.1. Apabila jumlah quantity muatan kendaraan yang dimasukan aktor tidak sama dengan quantity permintaan, maka sistem akan manampilkan pesan gagal, qty tidak sesuai.

	1.2. Apabila plat nomor yang dipilih masih dalam proses pengiriman pupuk, maka sistem akan memperlihatkan pesan kendaraan masih mengantar pupuk. 3. Apabila plat nomor yang dipilih sama dengan yang dipilih sebelumnya, maka sistem akan memperlihatkan pesan kendaraan yang dipilih tidak boleh sama.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan permintaan pupuk telah berhasil ditanggapi.

Tabel 4.29 Skenario Use Case Melihat Selisih Stok Opname di Gudang Penyangga

Aktor	Operator Rekanan Gudang Penyangga.
Objective	Aktor Melihat Selisih Stok Opname di Gudang Penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih menu pupuk hilang
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan data pupuk yang hilang digudang penyangga pada halaman pupuk hilang.

Tabel 4.30 Skenario Use Case Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga)

Aktor	Operator Rekanan Gudang Penyangga,
Objective	Aktor melihat hasil stok opname yang sudah divalidasi oleh operator gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu stok opname 2. Sistem memperlihatkan halaman menu stok opname dan tabel daftar tugas stok opname yang dikerjakan. 3. Aktor memilih *button* hasil pada tabel daftar tugas stok opname yang dikerjakan.
Alternatif Flow	-

Post-Condition	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.

Tabel 4.31 Skenario Use Case Membuat Perhitungan Stok Opname

Aktor	Operator Stok Opname Diswil 1
Objective	Membuat perhitungan stok opname
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu stok opname.
Main Flow	1. Aktor menekan *button* hitung pada penugasan stok opname yang ingin di ekseskusi. 2.Sistem memperlihatkan halaman hitungan stok opname perpupuk. 3. Aktor memilih pupuk yang dihitung. 4. Aktor memilih *button* tambah. 5. Sistem menambah pupuk pada tabel hitung stok opname perpupuk. 6. Aktor memilih *button* proses 7. Sistem memperlihatkan modal perhitungan per pupuk. 8. Aktor memilih *button* tambah untuk menambah perhitungan pupuk. 9. Sistem menambah field perhitung pada tabel hitungan perpupuk. 10. Aktor mengisi data yang dibutuhkan untuk menghitung stok opname yaitu area, kunci, panjang, lebar, tinggi, lebihan bag. 11. Sistem akan otomatis memunculkan hasil perhitungan pada jumlah bag, jumlah ton, lebihan ton, total bag, total ton. 12. Aktor milih *button* simpan. 13. Sistem menyimpan perhitungan stok opname pada databse dan menutup modal perhitungan perpupuk. 14. Aktor memilih *button* selesai 15. Sistem memunculkan modal konfirmasi. 16. Aktor memilih *button* ok.

Alternatif Flow	1. Apabila aktor tidak menambah perhitungan pupuk sama sekali, sistem akan memunculkan pemberitahuan "data isian tidak bisa kosong" 2. Apabila aktor tidak menghitung semua pupuk yang ada di gudang, sistem akan memunculkan pemberitahuan "Data Gagal Tersimpan, Ada Pupuk Yang Belum Dihitung "
Post-Condition	Sistem memperlihatkan pesan berhasil, dan data perhitungan stok opname berhasil dibuat.

Tabel 4.32 Skenario Use Case Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1)

Aktor	Operator Stok Opname Diswil 1
Objective	Aktor melihat hasil stok opname pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu stok opname. 2. Sistem memunculkan ketegori menu stok opname 3. Aktor memilih menu penugasan stok opname. 4. Sistem memunculkan halaman menu stok opname 5. Aktor memilih *button* hasil pada stok opname yang diinginkan.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan modal hasil stok opname.

Tabel 4.33 Skenario Use Case Melihat Peta Kondisi Stok Daerah

Aktor	Operator Diswil 1, Manajer Diswil 1
Objective	Aktor melihat peta kondisi stok daerah pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih ketegori menu pupuk. 2. Sistem memperlihatkan isi kategori menu pupuk. 3. Aktor memilih menu kondisi stok daerah. 4. Sistem memperlihatkan peta kondisi stok.

	5. Aktor menekan pada peta daerah yang diinginkan
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan data total stok berbagai pupuk dan detail pergudang dari kabupaten/kota yang dipilih.

Tabel 4.34 Skenario Use Case Membuat Laporan Pupuk Rusak

Aktor	Operator Gudang Penyangga
Objective	Aktor membuat laporan pupuk rusak pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu stok pupuk.
Main Flow	1. Aktor menekan *button* pupuk rusak pada jenis pupuk yang ingin dilaporkan 2. Sistem memperlihatkan form tambah Pupuk Rusak 3. Aktor menambahkan data laporan pupuk rusak yaitu qty rusak (ton), foto bukti, dan alasan rusak. 4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data laporan pupuk rusak ke dalam basis data aplikasi/sistem.
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu data maka akan memperlihatkan pemberitahuan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan data laporan pupuk rusak berhasil dibuat.

Tabel 4.35 Skenario Use Case Melihat Laporan Pupuk Rusak (Operator Diswil 1)

Aktor	Operator Diswil 1
Objective	Aktor melihat laporan pupuk rusak pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk 2. Sistem manampilkan isi kategori menu pupuk 3. Aktor milih menu pupuk rusak.
Alternatif Flow	-

Post-Condition	Sistem memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.

Tabel 4.36 Skenario Menerima Pupuk Masuk Gudang Penyangga

Aktor	Operator Gudang Penyangga.
Objective	Aktor menerima pupuk masuk gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu pupuk masuk.
Main Flow	1. Aktor memilih *button* terima. 2. Sistem manampilkan form tambah pupuk masuk. 3. Aktor mengisi data pupuk masuk yang diperlukan yaitu qty diterima (ton) 4. Aktor menekan *button* simpan 5. Sistem manampilkan modal konfirmasi. 6. Aktor memilih *button* ya
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu data maka akan memperlihatkan pemberitahuan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan data berhasil tesimpan dan status pengiriman pupuk yang diinginkan berubah menjadi selesai

Tabel 4.37 Skenario Use Case Menambah Pupuk Keluar Gudang Penyangga

Aktor	Operator Gudang Penyangga
Objective	Aktor menambah pupuk keluar gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login, dan sudah pada menu stok pupuk.
Main Flow	1. Aktor menekan *button* pupuk keluar pada pupuk yang diinginkan. 2. Sistem memperlihatkan modal tambah pupuk keluar. 3. Aktor mengisi data yang diperlukan untuk menambah pupuk keluar yaitu distributor, nomor polisi, qty keluar (ton)

	4. Aktor menekan *button* simpan. 5. Sistem merekam dan memasukan data pupuk keluar gudang penyangga ke database
Alternatif Flow	1. Apabila aktor tidak mengisi salah satu atau semua data maka akan memperlihatkan pemberitahuan untuk mengisi semua form.
Post-Condition	Sistem memperlihatkan data berhasil tersimpan dan data pupuk keluar berhasil ditambahkan.

Tabel 4.38 Skenario Use Case Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga)

Aktor	Operator Gudang Penyangga
Objective	Aktor melihat pupuk keluar gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk. 2. Sistem memperlihatkan isi kategori menu pupuk 3. Aktor memilih menu pupuk keluar
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan menu pupuk keluar dan memperlihatkan data pupuk keluar pada tabel pupuk keluar.

Tabel 4.39 Skenario Use Case Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga)

Aktor	Operator Gudang Penyangga
Objective	Aktor melihat data pupuk masuk ke gudang penyangga.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk. 2. Sistem memperlihatkan isi kategori menu pupuk 3. Aktor memilih menu pupuk masuk penyangga

Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu pupuk masuk dan memperlihatkan data pupuk masuk pada tabel pupuk masuk.

Tabel 4.40 Skenario Use Case Validasi Hasil Stok Opname

Aktor	Operator Gudang Penyangga.
Objective	Aktor memvalidasi hasil stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu hasil stok opname.
Main Flow	1. Aktor memilih *button* validasi pada hasil stok opname yang ingin divalidasi 2. Sistem memperlihatkan modal konfirmasi 3. Aktor memilih *button* ya pada modal konfirmasi.
Alternatif Flow	1. Aktor memilih *button* proses kembali, maka proses stok opname akan dihitung kembali.
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan status stok opname yang diinginkan berubah menjadi tervalidasi.

Tabel 4.41 Skenario Use Case Memproses Pemberian Pupuk

Aktor	Operator Gudang Pusat.
Objective	Aktor memproses pemberian pupuk dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu pemberian pupuk.
Main Flow	1. Aktor menekan *button* proses pada permintaan pupuk yang diinginkan. 2. Sistem menampikan form modal proses keberangkatan pengiriman pupuk. 3. Aktor memilih *button* simpan
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan status permintaan pengiriman pupuk yang diproses berubah menjadi pusat --> gudang

Tabel 4.42 Skenario Use Case Melihat Pupuk Hilang Saat Pengiriman

Aktor	Operator Rekanan Transportir.
Objective	Aktor melihat pupuk hilang saat pengiriman dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu pupuk hilang.
Alternatif Flow	-
Post-Condition	Sistem berhasil memperlihatkan data dan halaman pupuk hilang

Tabel 4.43 Skenario Use Case Menerima Pengembalian Pupuk Rusak

Aktor	Operator Gudang Pusat
Objective	Aktor menerima pengembalian pupuk rusak dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu pupuk rusak.
Main Flow	1. Aktor menakan *button* proses pada pupuk rusak yang diterima 2. Sistem memperlihatkan modal konfirmasi 3. Aktor menekan *button* terima. 4. Sistem mengubah status pengembalian pupuk rusak.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan berhasil, dan data pupuk rusak telah terverifikasi diterima gudang pusat.

Tabel 4.44 Skenario Use Case Mencetak Surat Jalan

Aktor	Operator Rekanan Transportir
Objective	Aktor mencetak surat jalan dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu permintaan pupuk.
Main Flow	1. Aktor memilih *button* surat jalan pada tabel status permintaan pupuk.

Alternatif Flow	-
Post-Condition	Sistem memperlihatkan surat jalan pengiriman yang dipilih.

Tabel 4.45 Skenario Use Case Validasi Pupuk Rusak

Aktor	Operator Diswil 1
Objective	Aktor memvalidasi pupuk rusak dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu pupuk rusak.
Main Flow	1. Aktor memilih *button* ubah status pada laporan pupuk rusak yang diinginkan 2. Sistem memperlihatkan modal ubah status 3. Aktor mengubah status menjadi setujui. 4. Aktor menekan *button* simpan.
Alternatif Flow	1. Aktor mengubah status menjadi review ulang, Sistem akan mengubah status laporan pupuk rusak menjadi review ulang.
Post-Condition	Sistem memperlihatkan pesan berhasil, dan status laporan pupuk rusak telah berubah valid.

Tabel 4.46 Skenario Use Case Mengirim Pupuk Rusak

Aktor	Operator Gudang Penyangga
Objective	Aktor dapat mengirim pupuk rusak dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login dan sudah pada menu pupuk rusak.
Main Flow	1. Aktor menekan *button* kirim pupuk rusak. 2. Sistem memperlihatkan form kirim pupuk rusak. 3. Aktor melengkapi data yang diperlukan untuk mengirim pupuk rusak ke gudang pusat yaitu nama supir, plat nomor, pupuk rusak yang dikirim. 4. Aktor memilih *button* simpan.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan pesan data berhasil tersimpan dan data pengiriman pupuk rusak berhasil tertambah.

Tabel 4.47 Skenario Use Case Login

Aktor	*Guest*
Objective	Aktor melakukan login pada sistem untuk dikenali hak aksesnya oleh sistem.
Pre-Condition	Aktor telah membuka halaman login.
Main Flow	1. Aktor mengisikan data username dan password yang sesuai dan benar. 2. Aktor menekan *button* login pada halam login. 3. Sistem memeriksa dan validitasi data yang diisikan aktor pada halaman login.
Alternatif Flow	1. Jika data username atau password tidak valid, maka sistem memperlihatkan pesan error bahwa login gagal dan memperlihatkan halaman login. 2. Jika terdapat kolom yang tidak terisi, maka sistem memperlihatkan pesan gagal login dan memperlihatkan halaman login.
Post-Condition	Sistem memperlihatkan *home page* sesuai dengan aktor yang berhasil dikenali.

Tabel 4.48 Skenario Use Case Logout

Aktor	*Guest*
Objective	Aktor melakukan logout untuk keluar dari sistem dan menghapus hak akses yang ada pada sistem.
Pre-Condition	Aktor telah login ke dalam sistem.
Main Flow	1. Aktor menekan *button* menu logout. 2. Sistem menghapus hak akses akun dari sistem.
Alternatif Flow	-
Post-Condition	Sistem menghilangkan hak akses aktor dan memperlihatkan halaman login.

Tabel 4.49 Skenario Use Case Melihat Daftar Pemberian Pupuk

Aktor	Operator Gudang Pusat.
Objective	Aktor Melihat Daftar Pemberian Pupuk dalam sistem.
Pre-Condition	Aktor harus masuk ke dalam sistem dengan cara login.
Main Flow	1. Aktor memilih menu pemberian pupuk.

Alternatif Flow	1. Apabila aktor tidak valid, sistem akan memperlihatkan halaman home.
Post-Condition	Sistem berhasil memperlihatkan data dan halaman pemberian pupuk.

Tabel 4.50 Skenario Use Case Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat)

Aktor	Operator Gudang Pusat
Objective	Aktor melihat data pengiriman pupuk rusak dalam sistem.
Pre-Condition	Aktor harus masuk ke dalam sistem dengan cara login.
Main Flow	1. Aktor memilih menu pupuk rusak.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu pupuk rusak dan tabel pengiriman pupuk rusak.

Tabel 4.51 Skenario Use Case Melihat Gudang Penyangga yang Dikelola

Aktor	Operator Rekanan Gudang Penyangga
Objective	Aktor melihat data gudang penyangga yang dikelola dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu daftar gudang penyangga.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman daftar Data Gudang Penyangga dan tabel daftar gudang penyangga yang dikelola oleh aktor tersebut.

Tabel 4.52 Skenario Use Case Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga)

Aktor	Operator Rekanan Gudang Penyangga
Objective	Aktor melihat penugasan stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login

Main Flow	1. Aktor memilih menu stok oname.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu penugasan stok opname.

Tabel 4.53 Skenario Use Case Melihat Penugasan Stok Opname (Manajer Diswil 1)

Aktor	Manajer Diswil 1
Objective	Aktor melihat penugasan stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu stok oname.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu penugasan stok opname.

Tabel 4.54 Skenario Use Case Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1)

Aktor	Operator Stok Opname Diswil 1
Objective	Aktor melihat penugasan stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu penugasan stok oname.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu penugasan stok opname.

Tabel 4.55 Skenario Use Case Melihat Riwayat Penugasan Stok Opname

Aktor	Operator Stok Opname Diswil 1
Objective	Aktor melihat riwayat penugasan stok opname dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login

Main Flow	1. Aktor memilih menu riwayat stok oname.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu riwayat stok opname.

Tabel 4.56 Skenario Use Case Melihat Pupuk masuk Gudang Penyangga (Operator Diswil 1)

Aktor	Operator Diswil 1
Objective	Aktor melihat data pupuk masuk ke gudang penyangga.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk. 2. Sistem memperlihatkan isi kategori menu pupuk 3. Aktor memilih menu pupuk masuk penyangga
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu pupuk masuk dan memperlihatkan data pupuk masuk pada tabel pupuk masuk.

Tabel 4.57 Skenario Use Case Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1)

Aktor	Operator Diswil 1
Objective	Aktor melihat pupuk keluar gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk. 2. Sistem memperlihatkan isi kategori menu pupuk 3. Aktor memilih menu pupuk keluar penyangga
Alternatif Flow	-

Post-Condition	Sistem memperlihatkan menu pupuk keluar dan memperlihatkan data pupuk keluar pada tabel pupuk keluar.

Tabel 4.58 Skenario Use Case Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga)

Aktor	Operator Gudang Penyangga
Objective	Aktor melihat laporan pupuk rusak pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu pupuk 2. Sistem manampilkan isi kategori menu pupuk 3. Aktor milih menu pupuk rusak.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.

Tabel 4.59 Skenario Use Case Melihat Laporan Pupuk Rusak (Manajer Diswil 1)

Aktor	Manajer Diswil 1.
Objective	Aktor melihat laporan pupuk rusak pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor milih menu pupuk rusak.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.

Tabel 4.60 Skenario Use Case Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga)

Aktor	Operator Gudang Penyangga.
Objective	Aktor melihat data pengiriman pupuk rusak dalam sistem.
Pre-Condition	Aktor harus masuk ke dalam sistem dengan cara login.

Main Flow	1. Aktor memilih menu pupuk rusak.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu pupuk rusak dan tabel pengiriman pupuk rusak.

Tabel 4.61 Skenario Use Case Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir)

Aktor	Operator Rekanan Transportir.
Objective	Aktor melihat permintaan pengiriman pupuk dalam sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu permintaan pupuk.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan daftar permintaan pengiriman pupuk di halaman menu permintaan pupuk.

Tabel 4.62 Skenario Use Case Melihat Hasil Stok Opname (Operator Gudang Penyangga)

Aktor	Operator Gudang Penyangga
Objective	Aktor melihat hasil stok opname pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login.
Main Flow	1. Aktor memilih kategori menu stok opname. 2. Sistem memunculkan ketegori menu stok opname 3. Aktor memilih menu hasil stok opname. 4. Sistem memunculkan halaman menu hasil stok opname 5. Aktor memilih *button* hasil pada stok opname yang diinginkan.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan modal hasil stok opname.

Tabel 4.63 Skenario Use Case Melihat Hasil Stok Opname Yang Sudah Divalidasi (Manajer Diswil 1)

Aktor	Manajer Diswil 1
Objective	Aktor melihat hasil stok opname yang sudah divalidasi oleh operator gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu stok opname 2. Sistem memperlihatkan halaman menu stok opname dan tabel daftar tugas stok opname yang dikerjakan. 3. Aktor memilih *button* hasil pada tabel daftar tugas stok opname yang telah divalidasi.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.

Tabel 4.64 Skenario Use Case Melihat Hasil Stok Opname Yang Sudah Divalidasi (Operator Diswil 1)

Aktor	Operator Diswil 1
Objective	Aktor melihat hasil stok opname yang sudah divalidasi oleh operator gudang penyangga pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih kategori menu stok opname 2. Sistem manampilkan menu dalam kategori menu stok opname 3. Aktor memilih menu penugasan stok opname 4. Sistem memperlihatkan halaman menu penugasan stok opname dan tabel daftar tugas stok opname yang dikerjakan. 5. Aktor memilih *button* hasil pada tabel daftar tugas stok opname yang telah divalidasi.
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.

Tabel 4.65 Skenario Use Case Melihat Stok Pupuk

Aktor	Operator Gudang Penyangga
Objective	Aktor dapat melihat stok pupuk yang ada pada gudang penyangga yang dikelola aktor pada sistem.
Pre-Condition	Aktor wajib telah berhasil memasuki sistem dengan cara login
Main Flow	1. Aktor memilih menu stok pupuk pada kategori menu pupuk
Alternatif Flow	-
Post-Condition	Sistem memperlihatkan halaman menu stok pupuk.

BAB 5 PERANCANGAN DAN IMPLEMENTASI

Pada tahap/fase ini dijelaskan mengenai fase perancangan dan implementasi darii sistem aplikasi manajemen distribusi pupuk. Sebelum dilakukan implementasi sistem, dilakukan perancangan sistem yang digunakan sebagai dasar untuk mengimplementasikan sistem, dalam tahap ini perancangan yang dilakukan berdasarkan analisis kebutuhan yang dilakukan. Sedangkan implementasi dilakukan dengan menerapkan hasil perancangan sistem untuk membangun sistem secara keseluruhan yang terbagi dalam implementasi database, implementasi kode program dan implementasi antarmuka sistem.

5.1 Perancangan

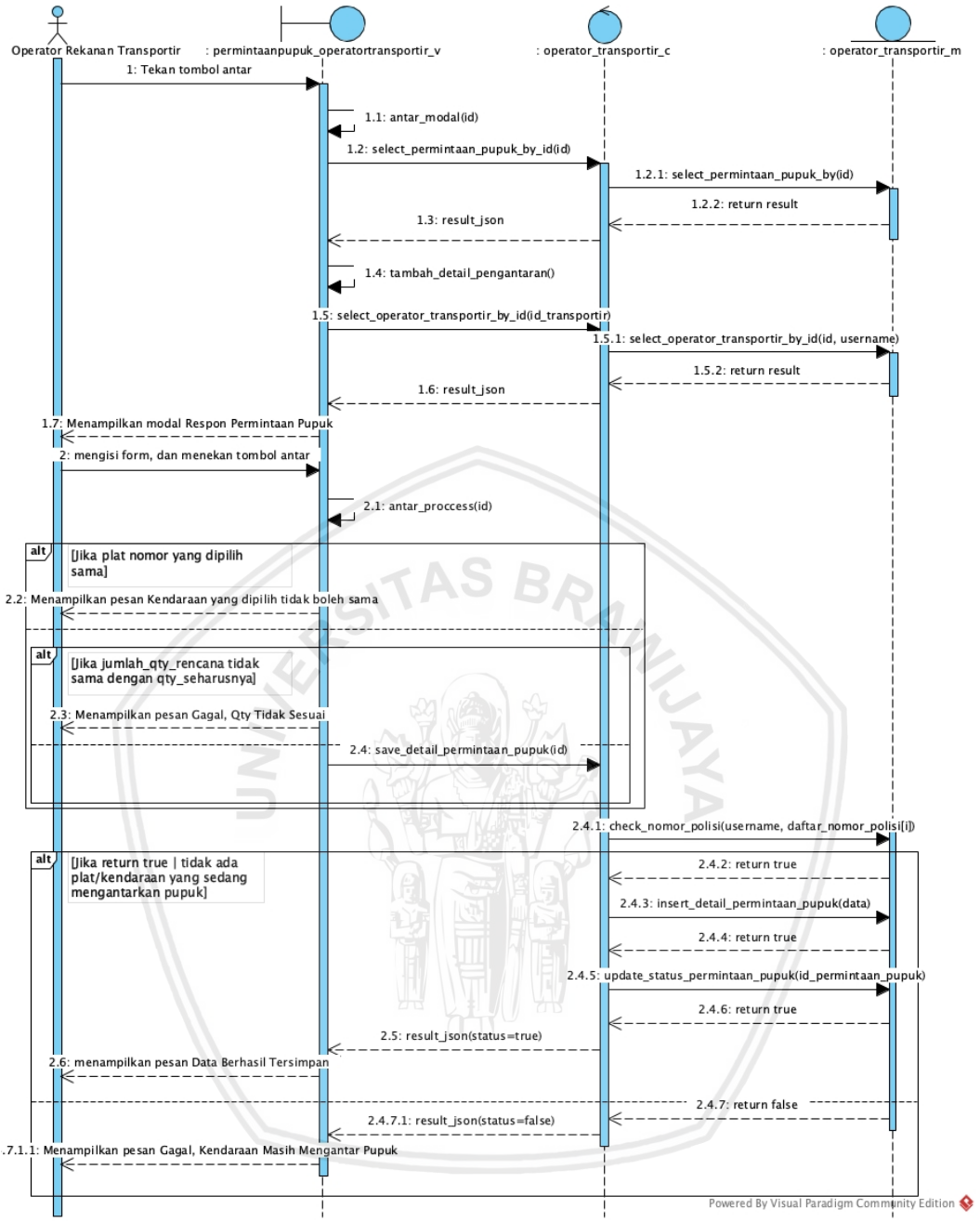
Setelah melakukan analisis kebutuhan, fase selanjutnya yakni perancangan. Total tahapan yang akan dikerjakan pada fase perancangan yakti lima tahap. Lima tahapan itu yakni perancangan diagram sekuen, perancangan diagram kelas, perancangan basis data, perancangan komponen, hingga perancangan antarmuka.

5.1.1 Perancangan *Sequence* Diagram

Diagram sekuen merupakan cara untuk mendeskripsikan/mempresentasikan kelakuan/interaksi antar objek pada use case. Beberapa hal yang dideskripsikan dalam diagram sekuen adalah waktu hidup objek dan pesan yang dikirimkan maupun diterima dari atau ke objek lainnya. Berikut merupakan tiga sequence diagram yang digunakan sebagai sampel.

5.1.1.1 *Sequence* Diagram Menanggapi Permintaan Pengiriman Pupuk

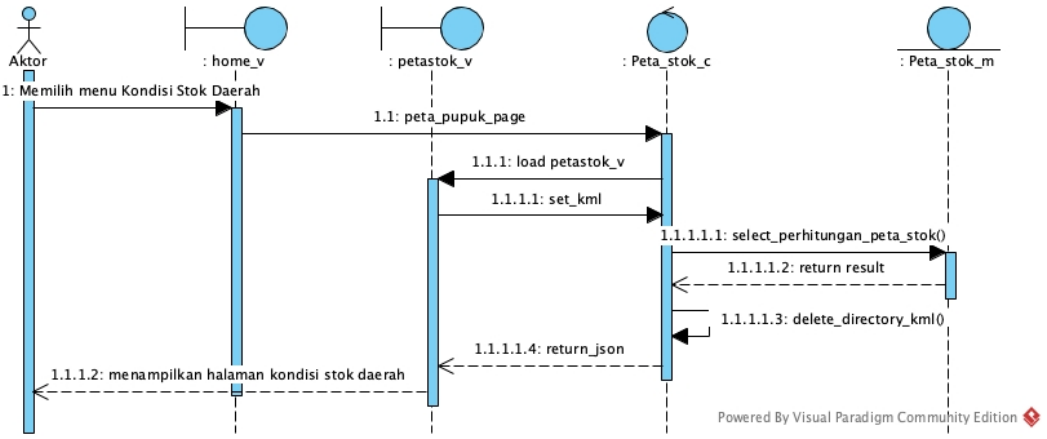
Sequence diagram menanggapi permintaan pengiriman pupuk terdapat pada Gambar 5.1 menjelaskan alur interaksi antar objek pada sistem saat proses menanggapi permintaan pengiriman pupuk. Pada *Sequence diagram* menanggapi permintaan pengiriman pupuk terdapat satu *boundary* yaitu permintaanpupuk_operatortransportir_v.php, kemudian satu controller yaitu operator_transportir_c.php, dan satu entity yaitu operator_transportir_m.php. Terdapat tiga alternatif yaitu jika plat nomor yang dipilih sama, jika qty rencana tidak sama dengan qty seharusnya dan jika tidak ada kendaraan yang dipilih yang sedang mengantarkan pupuk.



Gambar 5.1 Sequence Diagram Menanggapi Permintaan Pengiriman Pupuk

5.1.1.2 *Sequence* Diagram Melihat Peta Kondisi Stok Daerah

Sequence diagram melihat peta kondisi stok daerah terdapat pada Gambar 5.2 menjelaskan alur interaksi antar objek pada sistem saat proses melihat peta kondisi stok daerah. Pada *Sequence diagram* melihat peta kondisi stok daerah terdapat dua *boundary* yaitu home_v.php dan petastok_v.php, kemudian satu controller yaitu Peta_stok_c.php, dan satu entity yaitu Peta_stok_m.php.



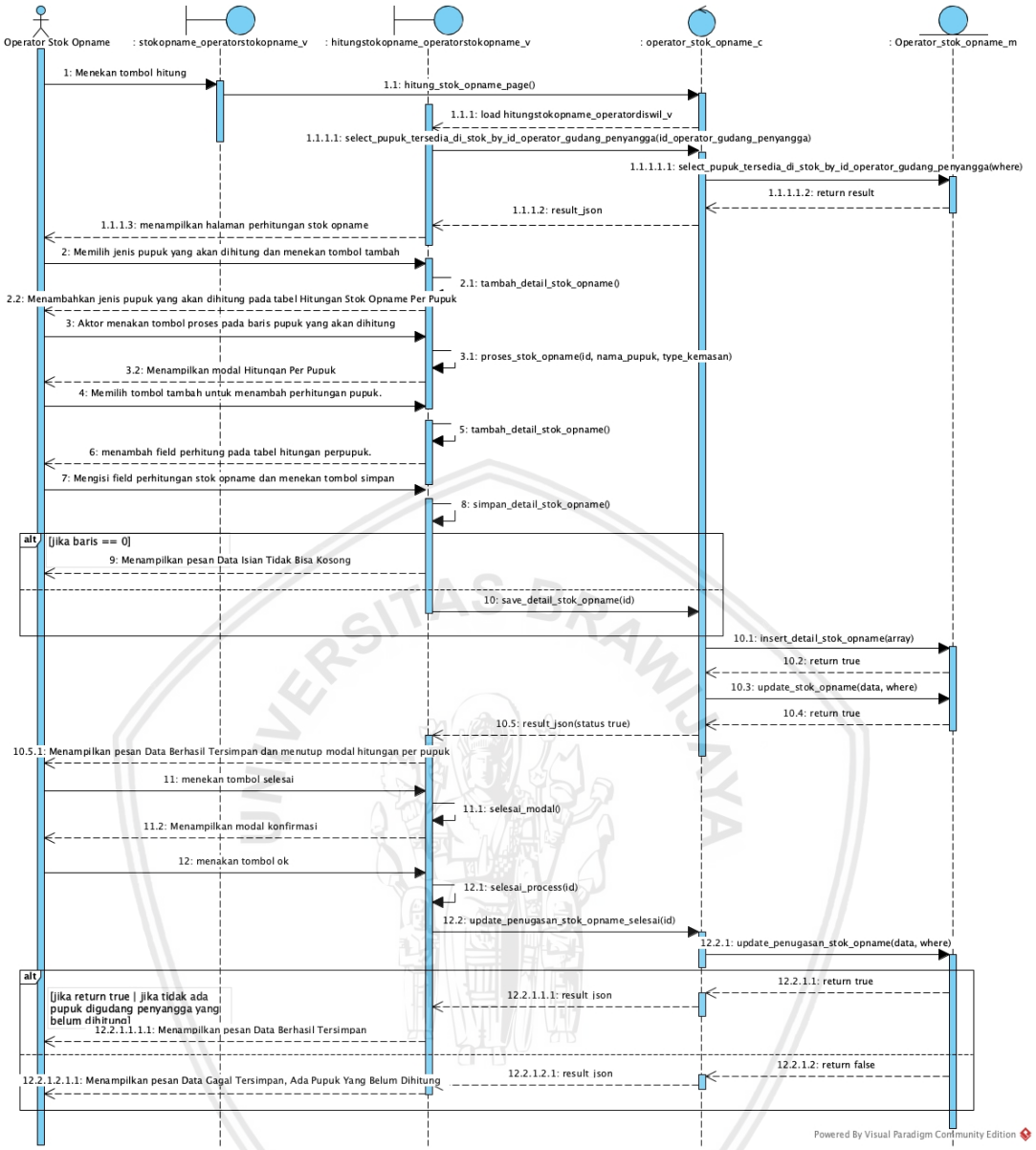
Gambar 5.2 Sequence Diagram Melihat Peta Kondisi Stok Daerah

5.1.1.3 *Sequence* Diagram Membuat Perhitungan Stok Opname

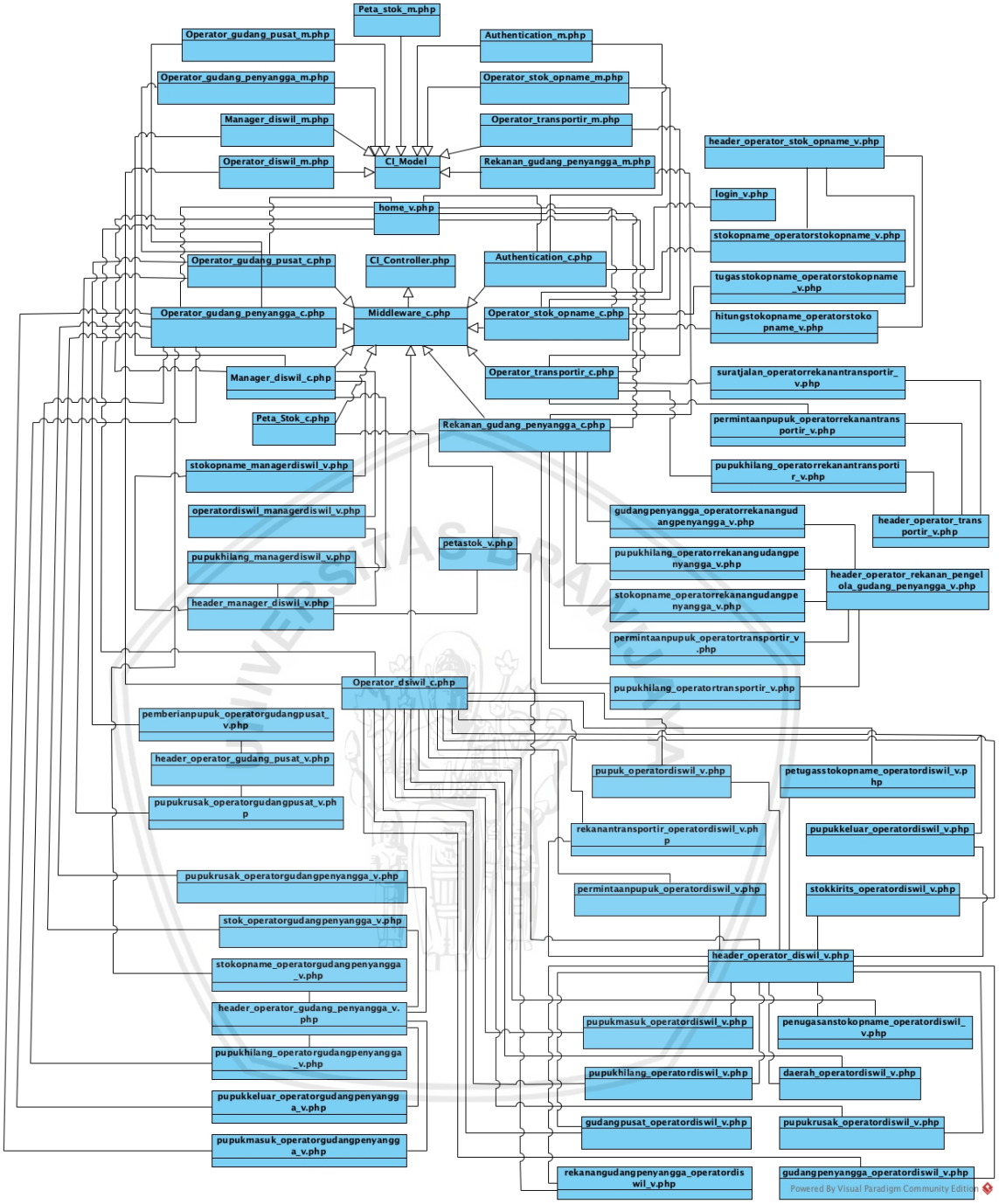
Sequence diagram membuat perhitungan stok opname terdapat pada Gambar 5.3 menjelaskan alur interaksi antar objek pada sistem saat proses membuat perhitungan stok opname. Pada *Sequence diagram* membuat perhitungan stok opname terdapat dua *boundary* yaitu stokopname_operatorstokopname_v.php dan hitungstokopname_operatorstokopname_v.php, kemudian satu controller yaitu operator_stok_opname_c.php, dan satu entity yaitu operator_stok_opname_m.php. Terdapat dua alternatif yaitu jika baris == 0 dan jika tidak ada pupuk digudang penyangga yang belum dihitung.

5.1.2 Perancangan *Class* Diagram

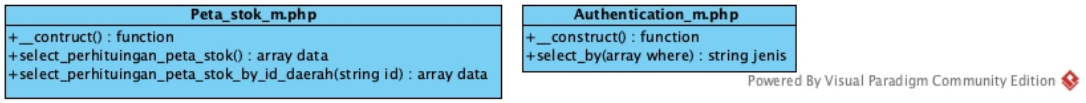
Perancangan *class* diagram berguna untuk mempresentasikan struktur dari sebuah sistem yang akan dikembangkan begitupula dalam pendefinisian kelas-kelas yang nantinya akan dibuat dalam pengkodean. Method atau operasi dan attribut haruslah didefinisikan dalam kelas diagram. Method atau operasi ialah fungsi yang ada dalam suatu kelas dan attribut ialah variable yang didefinisikan dalam sebuah kelas. *Class* diagram dari Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web terdiri dari 10 kelas controller, 9 kelas model dan 34 kelas view. *Class* Diagram dari Sistem Aplikasi Manajemen Distribusi Pupuk diilustrasikan pada Gambar 5.4. Sedangkan untuk detail dari kelas model dijelaskan pada Gambar 5.5, Gambar 5.6 dan Gambar 5.7. Untuk detail kelas controller pada Gambar 5.8 dan Gambar 5.9.



Gambar 5.3 Sequence Diagram Membuat Perhitungan Stok Opname

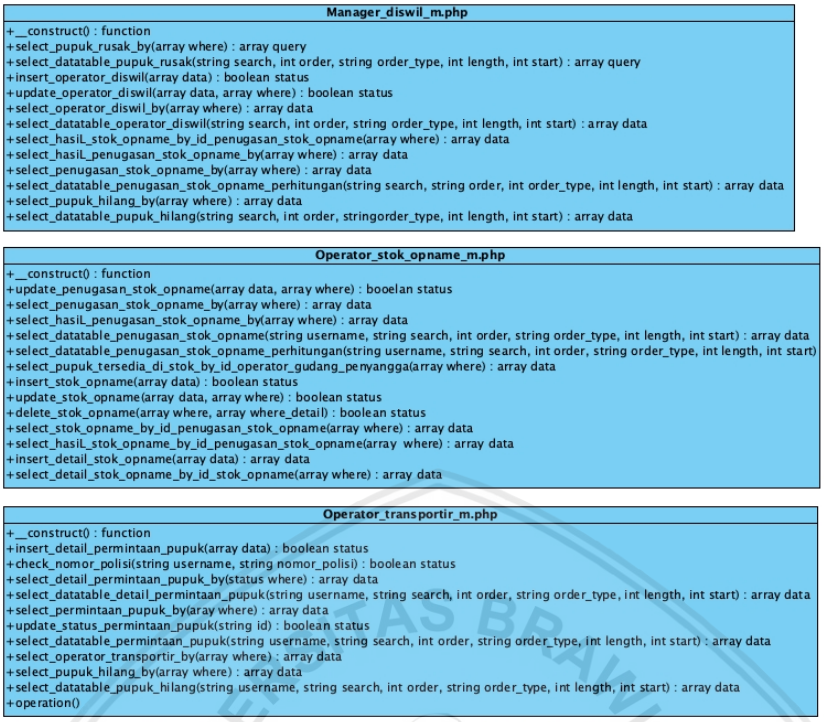


Gambar 5.4 Diagram Class

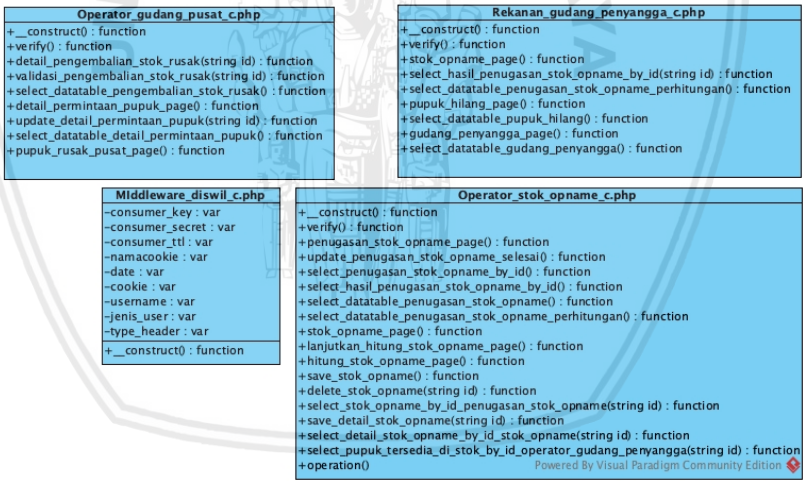


Gambar 5.5 Detail Class Diagram Model

Gambar 5.6 Lanjutan Detail Class Diagram Model



Gambar 5.7 Lanjutan Detail Class Diagram Model



Gambar 5.8 Detail Class Diagram Controller

5.1.3 Perancangan Komponen

Komponen perlu dirancangan agar dapat menjelaskan bagaimana algoritme dari sebuah proses yang terjadi dalam sistem. Pendefinisian algoritme fungsi direpresentasikan menggunakan pseudocode. Pada tahapan/fase ini terdapat 3 algoritme fungsi yang digunakan sebagai sampel, yaitu algoritme fungsi save_detail_permintaan_pupuk pada kelas Operator_transportir_c, set_kml pada kelas Peta_stok_c, dan save_detail_stok_opname pada kelas Operator_stok_opname_c.

Gambar 5.9 Lanjutan Detail Class Diagram Controller

5.1.3.1 Algoritme Fungsi save_detail_permintaan_pupuk

Algoritme fungsi save_detail_permintaan_pupuk dapat dijabarkan didalam Kode Program 5.1 yang merupakan algortime yang yang menjelaskan proses operator rekanan transportir ketika sudah mengisi semua data detail permintaan pupuk untuk selanjutnya disimpan dalam basis data.

Nama Class: Operator_transportir_c
Nama Method: save_detail_permintaan_pupuk
Algoritme:

```
1  Mulai
2  Mengecek autentikasi jenis user operator transportir
```

3	Inialisasi variable daftar_nama_supir dengan mengambil post input nama_supir
4	Inialisasi variable daftar_nomor_polisi dengan mengambil post input nomor_polisi
5	Inialisasi variable daftar_qty_rencana dengan mengambil post input qty_rencana
6	Inialisasi variable id_permintaan_pupuk dengan mengambil nilai id
7	Inialisasi variable panjang_data dengan mengambil jumlah banyak data variable daftar_nama_supir
8	Perulangan for i sampai kurang dari panjang_data
9	Inialisasi variable status dengan mengirimkan username dan daftar_nomor_polisi[array indeks i] kepada method check_nomor_polisi dari kelas model operator_transportir lalu mengambil hasilnya
10	Jika status sama dengan false maka
11	Inialisasi variable result_json = array status false
12	Print variable result_json yang diencode ke json
13	Break
14	Jika panjang_data sama dengan i + 1 maka
15	Perualangan i sampai kurang dari panjang_data
16	Inialisasi variable id_detail_permintaan_pupuk dengan nilai 'dpp' dan waktu saat itu dengan format (tahun-bulan-tanggal jam-menit-detik)
17	Inialisasi variable nama_supir dengan nilai dari variable array daftar_nama_supir[array index i]
18	Inialisasi variable nomor_polisi dengan nilai dari variable array daftar_nomor_polisi[array index i]
19	Inialisasi variable qty_rencana dengan nilai dari variable array daftar_qty_rencana[array index i]
20	Inialisasi variable status dengan nilai 'transportir --> pusat'
21	Inialisasi variable tanggal_berangkat dengan nilai waktu saat itu dengan format (year-month-date)
22	Inialisasi variable created_at dengan nilai waktu saat itu dengan format (tahun-bulan-tanggal jam-menit-detik)
23	Inialisasi variable updated_at dengan nilai waktu saat itu dengan format (tahun-bulan-tanggal jam-menit-detik)
24	Inialisasi variable associative array data dengan berisi
25	Array key id dengan value nilai dari variable id_detail_permintaan_pupuk
26	Array key id_permintaan_pupuk dengan value nilai dari variable id_permintaan_pupuk
27	Array key qty_rencana dengan value nilai dari variable qty_rencana
28	Array key nama_supir dengan value nilai dari variable nama_supir
29	Array key nomor_polisi dengan value nilai dari variable nomor_polisi
30	Array key tanggal_berangkat dengan value nilai dari variable tanggal_berangkat
31	Array key status dengan value nilai dari variable status
32	Array key created_at dengan value nilai dari variable created_at
33	Array key updated_at dengan value nilai dari variable updated_at
34	Delay eksekusi 1 detik
35	Mengirim dan memanggil variable data ke method insert_detail_permintaan_pupuk

```
End Perulangan
Mengirim dan memanggil variable id_permintaan_pupuk ke
method update_status_permintaan_pupuk
Inialisasi variable result_json = array status true
Print variable result_json yang diencode ke json
End seleksi kondisi
End Perulangan
Selesai
```

Kode Program 5.1 Algoritme Fungsi save_detail_permintaan_pupuk

5.1.3.2 Algoritme Fungsi set_kml

Algoritme fungsi set_kml dapat dijabarkan didalam Kode Program 5.2 yang merupakan algortime yang yang menjelaskan proses manajer diswil 1 atau operator diswil 1 ketika mengakses halaman menu kondisi stok daerah dan memperlihatkan layering peta kondisi stok daerah.

Nama Class: Peta_stok_c
Nama Method: set_kml
Algoritme:

```
Mulai
Inialisasi variable warna dengan nilai kode warna default
kml
Inialisasi variable peta_stok dengan nilai dari hasil
method select_perhitungan_peta_stok()
Inialisasi variable data dengan nilai tag awal file kml
Inialisasi variable id_daerah dengan nilai string kosong
Inialisasi variable status_per_daerah dengan nilai true
Inialisasi variable count dengan nilai 0
Perulangan i sampai kurang dari banyak data variable
peta_stok
Inialisasi variable warna dengan nilai warna default kml
Jika variable count sama dengan 0 maka
Inialisasi variable id_daerah dengan nilai id_daerah dari
array peta_stok[indeks count]
End seleksi kondisi
Jika qty dari array peta_stok[indeks count] kurang dari
kritis dari array peta_stok[indeks count]
Inialisasi variable status_per_daerah dengan nilai false
End seleksi kondisi
Jika variable count kurang dari banyak data dari variable
peta_stok - 1 maka
Jika variable id_daerah tidak sama dengan id_daerah dari
array peta_stok[indeks count+1] maka
Jika status_per_daerah adalah true maka
Inialisasi variable warna dengan nilai kode warna hijau
Else
Inialisasi variable warna sama dengan nilai kode warna
merah
End seleksi kondisi
Menambahkan nilai variable data dengan style id dan
polystyle color kml
Inialisasi variable id_daerah dengan nilai id_daerah dari
array peta_stok[indeks count+1]
Inialisasi variable status_per_daerah dengan nilai true
End seleksi kondisi
```

```
28  Jika variable count sama dengan banyak data variable
29  peta_stok - 1 maka
    Jika variable status_per_daerah adalah true maka
30  Inialisasi variable warna dengan nilai kode warna hijau
31  Else
32  Inialisasi variable warna dengan nilai kode warna merah
33  End seleksi kondisi
34  Menambahkan nilai variable data dengan style id dan
35  polystyle color kml
    Inialisasi variable status_per_daerah dengan nilai true
36  End seleksi kondisi
37  Tambah nilai variable count
38  End perulangan
39  Inialisasi variable status_delete nilai dari hasil method
    delete_directory_kml()
40  Inialisasi variable nama_kml dengan nilai peta dan waktu
    saat itu dengan format tahun-bulan-hari-menit-detik lalu
    dienkripsi dengan md5
41  Inialisasi variable data_lokasi dengan isi dari file pada
    direktori "./assets/uploads/datalayer.txt"
42  Inialisasi variable status_write dengan nilai balikan
    boolan dari aksi write_file di direktori "
    ./assets/uploads/kml/".nilai variable nama_kml.".kml",
    menulis nilai dari variable data dengan mode write binary
43  Inialisasi variable status_write_append dengan nilai
    balikan boolan dari aksi write_file di direktori "
    ./assets/uploads/kml/".nilai variable nama_kml.".kml",
    menulis nilai dari variable data_lokasi dengan mode
    append
44  Inialisasi variable return_json
45  Inialisasi variable associative array return_json dengan
    berisi
46  Array key status dengan value nilai true
47  Array key src dengan value nilai direktori file kml
48  Print variable result_json yang diencode ke json
49  Selesai
```

Kode Program 5.2 Algoritme Fungsi set_kml

5.1.3.3 Algoritme Fungsi save_detail_stok_opname

Algoritme fungsi save_detail_stok_opname dapat dijabarkan didalam Kode Program 5.3 yang merupakan algortime yang yang menjelaskan proses operator stok opname ketika sudah melakukan perhitungan stok opname untuk selanjutnya data perhitungan akan disimpan dalam basis data.

Nama Class: operator_stok_opname_c
Nama Method: save_detail_stok_opname
Algoritme:

```
1  Mulai
2  Mengecek autentikasi jenis user operator stok opname
3  Inialisasi variable detail dengan mengambil post input
   detail_stok_opname
4  Inialisasi variable total_hitung_kemasan dengan mengambil
   post input total_hitung_kemasan
```

5	Inialisasi variable total_hitung_ton dengan mengambil post input total_hitung_ton
6	Perulangan i sampai kuarang dari banyak data dari variable count
7	Inialisasi variable id dengan nilai "dso" dan waktu saat itu dengan format tahun-bulan-tanggal-jam-menit-detik
8	Inialisasi variable zona dengan nilai array detail index i dan 0
9	Inialisasi variable kunci dengan nilai array detail index i dan 1
10	Inialisasi variable panjang dengan nilai array detail index i dan 2
11	Inialisasi variable lebar dengan nilai array detail index i dan 3
12	Inialisasi variable tinggi dengan nilai array detail index i dan 4
13	Inialisasi variable jumlah_kemasan dengan nilai array detail index i dan 5
14	Inialisasi variable jumlah_ton dengan nilai array detail index i dan 6
15	Inialisasi variable lebihan_kemasan dengan nilai array detail index i dan 7
16	Inialisasi variable lebihan_ton dengan nilai array detail index i dan 8
17	Inialisasi variable total_kemasan dengan nilai array detail index i dan 9
18	Inialisasi variable total_ton dengan nilai array detail index i dan 10
19	Inialisasi variable created_at dengan nilai waktu saat itu dengan format tahun-bulan-tanggal-jam-menit-detik
20	Inialisasi variable updated_at dengan nilai waktu saat itu dengan format tahun-bulan-tanggal-jam-menit-detik
21	Memanggil method insert_detail_stok_opname dengan mengirim associative array dengan berisi
22	Array key id dengan value nilai dari variable id
23	Array key id_stok_opname dengan value nilai dari variable id_stok_opname
24	Array key zona dengan value nilai dari variable zona
25	Array key kunci dengan value nilai dari variable kunci
26	Array key panjang dengan value nilai dari variable panjang
27	Array key lebar dengan value nilai dari variable lebar
28	Array key tinggi dengan value nilai dari variable tinggi
29	Array key lebihan_kemasan dengan value nilai dari variable lebihan_kemasan
30	Array key lebihan_ton dengan value nilai dari variable lebihan_ton
31	Array key jumlah_kemasan dengan value nilai dari variable jumlah_kemasan
32	Array key jumlah_ton dengan value nilai dari variable jumlah_ton
33	Array key total_kemasan dengan value nilai dari variable total_kemasan
34	Array key total_ton dengan value nilai dari variable total_ton
35	Array key created_at dengan value nilai dari variable created_at
36	Array key updated_at dengan value nilai dari variable updated_at
37	Delay eksekusi 1 detik

```
38  End perulangan
39  Inialisasi variable associative array data dengan berisi
40  Array key total_hitung_kemasan dengan value nilai dari
    variable total_hitung_kemasan
41  Array key total_hitung_ton dengan value nilai dari variable
    total_hitung_ton
42  Array key status dengan value selesai
43  Array key updated_at dengan value waktu saat itu dengan
    format tahun-bulan-tanggal-jam-menit-detik
44  Inialisasi variable associative array where dengan berisi
45  Array key id dengan value nilai dari variable
    id_stok_opname
46  Memanggil dan mengiriman variable data dan where ke method
    update_stok_opname
47  Inialisasi variable associative array return_json dengan
    berisi
48  Array key status dengan value nilai true
49  Array key src dengan value nilai direktori file kml
50  Print variable result_json yang diencode ke json
51  Selesai
```

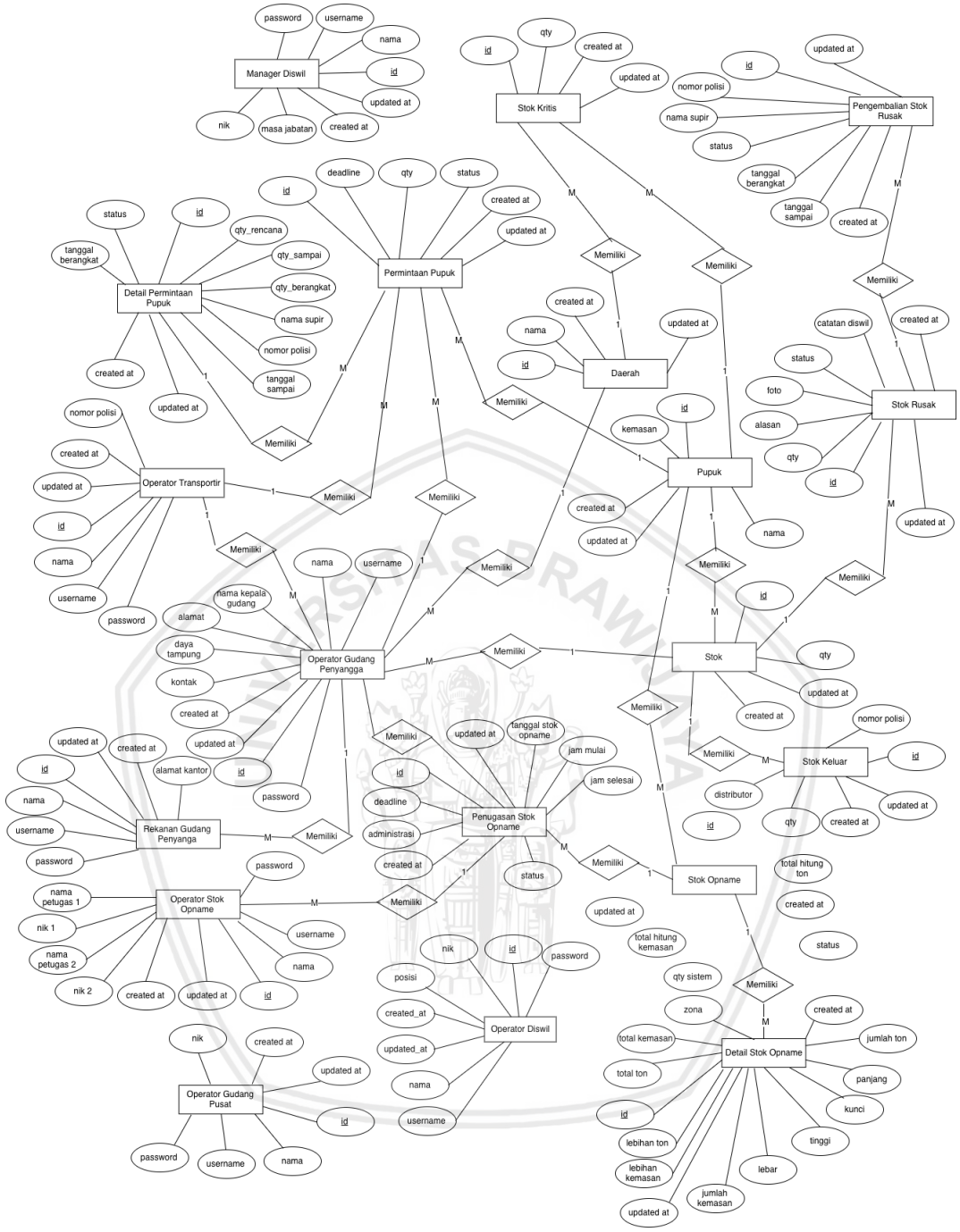
Kode Program 5.3 Algoritme Fungsi select_datatable_pupuk_rusak

5.1.4 Perancangan Basis Data

Perancangan basis data atau *database* dirancang secara konseptual dengan menggunakn Entity Relationship Diagram (ERD). Ada 19 entitas yaitu manajer diswil, stok kritis, pengembalian stok rusak, stok rusak, pupuk, daerah, permintaan pupuk, detail permintaan pupuk, operator transportir, operator gudang penyangga, stok, stok keluar, stok opname, penugasan stok opname, rekanan gudang penyangga, operator stok opname, operator gudang pusat, operator gudang diswil, dan detail stok opname. Perancangan basis data dari sistem aplikasi manajemen distribusi pupuk diilustrasikan dengan diagram pada Gambar 5.10.

5.1.5 Perancangan Antarmuka

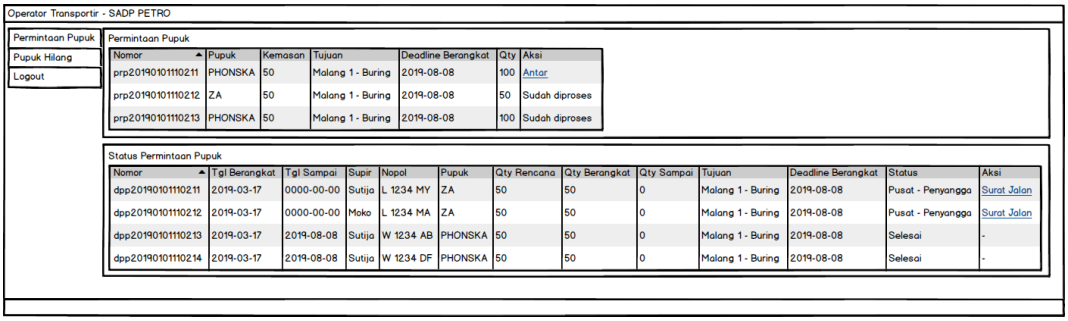
Pada fase/tahap ini akan dijabarkan hasil dari sebuah rancangan antarmuka tau tampilan dari sistem aplikasi manajemen distribusi pupuk yang telah dikembangkan. Tidak semua perancangan yang dimasukan. Hanya ada tiga sampel yang akan dijabarkan pada subbab ini yakni halaman menu permintaan pupuk, halaman menu kondisi stok daerah, dan halaman menu hitung stok opname. Sampel kemudian akan dijabarkan pada Gambar 5.11, Gambar 5.12, dan Gambar 5.13.



Gambar 5.10 Entity Relationship Diagram

5.1.5.1 Perancangan Antarmuka Halaman Menu Permintaan Pupuk

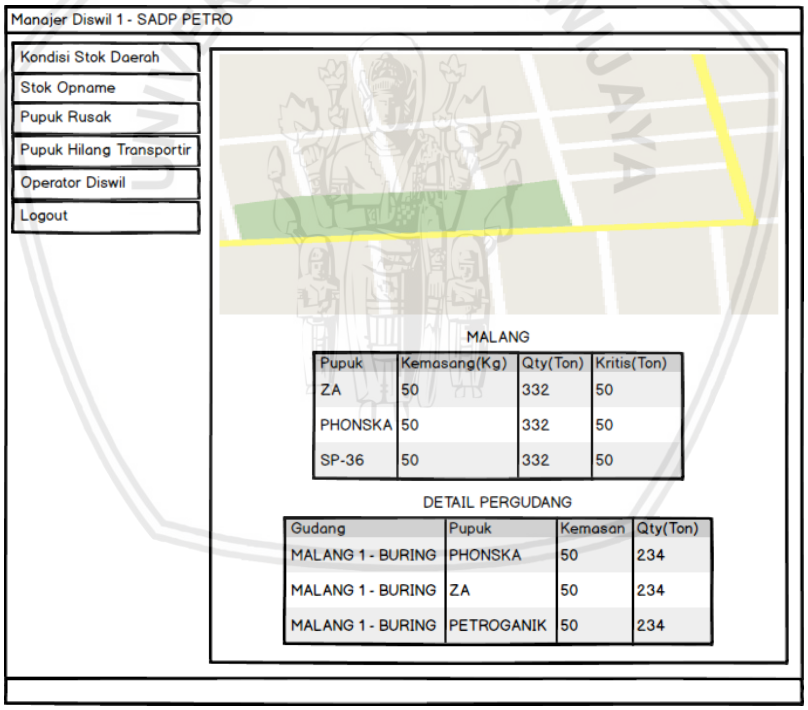
Halaman menu permintaan pupuk tersedia untuk beberapa aktor namu dari setiap aktor memiliki antarmuka dan penampilan data yang berbeda. Pada perancangan antarmuka ini ditujukan untuk halaman menu permintaan pupuk untuk aktor Operator Rekanan Transportir. Halaman ini berfungsi untuk memperlihatkan data permintaang pupuk dan menanggapi permintaan pupuk dari operator diswil 1. Hasil perancangan antarmuka halaman menu permintaan pupuk dipresentasikan dalam Gambar 5.11.



Gambar 5.11 Perancangan Antarmuka Halaman Menu Permintaan Pupuk

5.1.5.2 Perancangan Antarmuka Halaman Menu Kondisi Stok Daerah

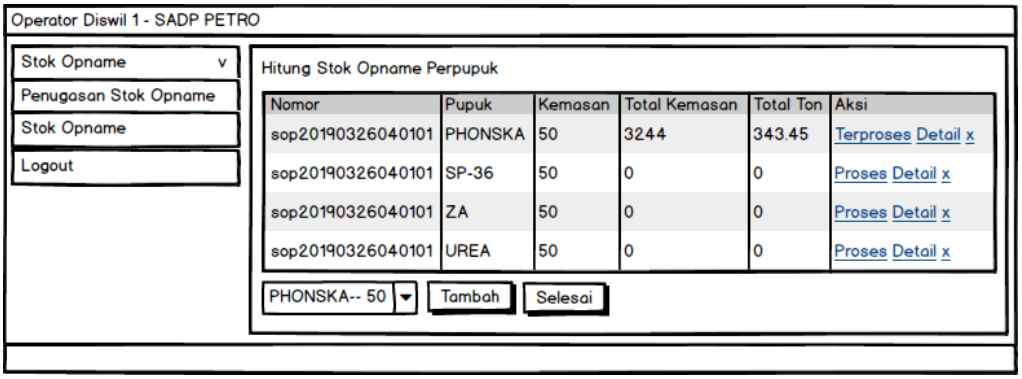
Antarmuka hitung stok opname merupakan halaman untuk operator stok opname yang berfungsi untuk menghitung stok opname. Hasil perancangan antarmuka hitung stok opname direpresentasikan dalam Gambar 5.12.



Gambar 5.12 Perancangan Antarmuka Halaman Kondisi Stok Daerah

5.1.5.3 Perancangan Antarmuka Halaman Hitung Stok Opname

Tampilan hitung stok opname yakni *page* untuk operator stok opname yang berfungsi untuk menghitung stok opname. Hasil perancangan antarmuka hitung stok opname direpresentasikan dalam Gambar 5.13.



Gambar 5.13 Perancangan Antarmuka Halaman Hitung Stok Opname

5.2 Implementasi

Pada fase ini, perancangan yang berhasil dirancang pada tahap sebelumnya kemudian diimplementasikan kedalam kode sumber pemrograman. Pengkodean dapat diawali dengan pembuatan unit-unit lalu selanjutnya digabungkan/diintegrasikan satu sama lain.

5.2.1 Spesifikasi Sistem

Spesifikasi sistem dari sistem aplikasi manajemen distribusi pupuk berbasis web yang digunakan dalam mengembangkan sistem dibagi menjadi dua sub bab, yaitu spesifikasi perangkat keras dan spesifikasi perangkat lunak.

5.2.1.1 Spesifikasi Perangkat Keras

Pengembangan sistem aplikasi manajemen distribusi berbasis web menggunakan perangkat keras dengan spesifikasi yang dijabarkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras

Nama Komponen	Spesifikasi
System Model	Apple Macbook Pro A1502
Processor	Intel Core i5 2,9 Ghz
Memory (RAM)	8GB Dual-Slot DDR3 1867 MHz
GPU	Intel Iris Graphics 6100
Storage	Apple SSD SM0512G

5.2.1.2 Spesifikasi Perangkat Lunak

Dalam Pengembang yang dilakukan terdapat beberapa perangkat lunak yang digunakan untuk mendukung pengembangan sistem ini. Adapun perangkat lunak pendukung tersebut akan disebutkan dalam spesifikasi perangkat lunak yang tertera pada Tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak

Nama Komponen	Spesifikasi
Sistem Operasi	MacOS Mojave 10.14.3
Bahasa Pemrograman	PHP, Javascript, CSS, HTML, Ajax
Perkakas Bantu	Visual Studio Code Visual Paradigm Community Edition Draw.io Microsoft Office 365

5.2.1.3 Batasan Implementasi

Berikut merupakan beberapa batasan dalam tahap implementasi sistem aplikasi manajemen distribusi pupuk berbasis web. Batasan-batasan tersebut diantaranya adalah:

1. Sistem yang dikembangkan berbasis web dengan menggunakan beberapa bahasa pemrograman antara lain PHP, CSS, HTML, AJAX, dan Javascript.
2. *Databasa Management System* yang digunakan adalah MySQL.
3. Sistem dikembangkan dengan menggunakan framework Codeigniter versi 3.17.

5.2.2 Implementasi Basis Data

Implementasi basis data dibuat berdasarkan hasil perancangan basis data yang telah dibuat pada tahap perancangan sistem. Implementasi basis data menggunakan PHPMyAdmin. Dari hasil implementasi basis data diperoleh 19 tabel yaitu manajer diswil, stok kritis, pengembalian stok rusak, stok rusak, pupuk, daerah, permintaan pupuk, detail permintaan pupuk, operator transportir, operator gudang penyangga, stok, stok keluar, stok opname, penugasan stok opname, rekanan gudang penyangga, operator stok opname, operator gudang pusat, operator gudang diswil, dan detail stok opname.

5.2.2.1 Data Definition Language (DDL)

Pada tabel 5.3 merupakan penggambaran bagaimana pembuatan sebuah basis data beserta tabel-tabel yang ada dengan menggunakan DDL.

Tabel 5.3 Implementasi Basis Data Dengan DDL

No.	Data Definition Language

1.	`CREATE TABLE `daerah` (`
	`  `id` varchar(255) NOT NULL,`
	`  `nama` varchar(255) NOT NULL,`
	`  `created_at` datetime NOT NULL,`
	`  `updated_at` datetime NOT NULL`
	`)`

2.	`CREATE TABLE `detail_permintaan_pupuk` (`

	`id` varchar(255) NOT NULL, `qty_rencana` double NOT NULL, `qty_sampai` double NOT NULL, `qty_berangkat` double NOT NULL, `nama_supir` varchar(120) NOT NULL, `nomor_polisi` varchar(120) NOT NULL, `status` varchar(120) NOT NULL, `tanggal_berangkat` date NOT NULL, `tanggal_sampai` date NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id_permintaan_pupuk` varchar(255) DEFAULT NULL)
3.	CREATE TABLE `detail_stok_opname` (`id` varchar(20) NOT NULL, `lebihan_ton` double NOT NULL, `lebihan_kemasan` int(11) NOT NULL, `jumlah_kemasan` int(11) NOT NULL, `lebar` int(11) NOT NULL, `tinggi` int(11) NOT NULL, `kunci` int(11) NOT NULL, `panjang` int(11) NOT NULL, `zona` varchar(120) NOT NULL, `total_kemasan` int(11) NOT NULL, `total_ton` double NOT NULL, `jumlah_ton` double NOT NULL, `id_stok_opname` varchar(120) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL)
4.	CREATE TABLE `manager_diswil` (`nik` int(11) NOT NULL, `masa_jebatan` varchar(120) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL)
5.	CREATE TABLE `operator_diswil` (`nik` int(11) NOT NULL, `posisi` varchar(120) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL)
6.	CREATE TABLE `operator_gudang_penyangga` (`nama_kepala_gudang` varchar(120) NOT NULL, `alamat` varchar(200) NOT NULL, `id_daerah` varchar(255) NOT NULL, `daya_tampung` double NOT NULL, `kontak` varchar(60) NOT NULL,

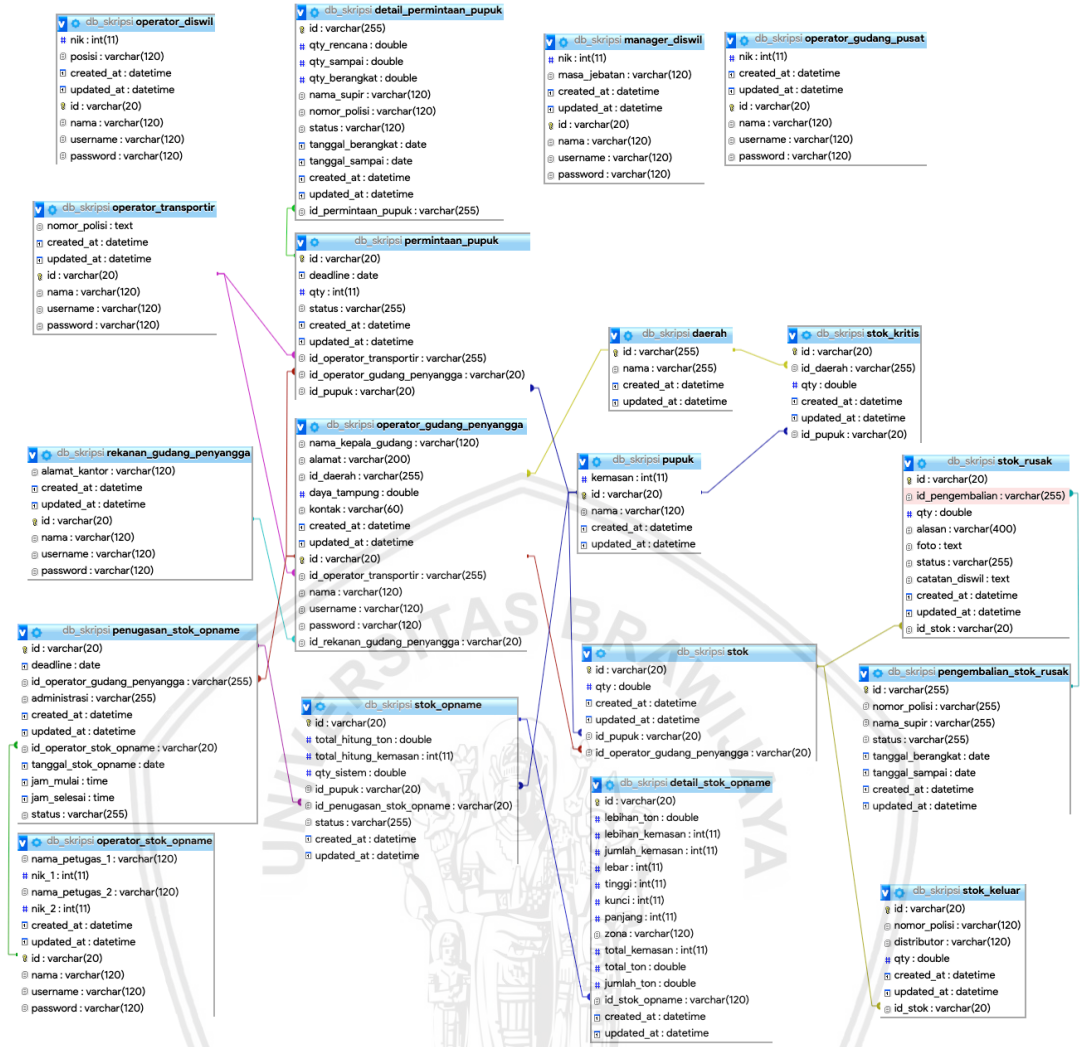
	`created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `id_operator_transportir` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL, `id_rekanan_gudang_penyangga` varchar(20) DEFAULT NULL)
7.	CREATE TABLE `operator_gudang_pusat` (`nik` int(11) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL)
8.	CREATE TABLE `operator_stok_opname` (`nama_petugas_1` varchar(120) NOT NULL, `nik_1` int(11) NOT NULL, `nama_petugas_2` varchar(120) NOT NULL, `nik_2` int(11) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL)
9.	CREATE TABLE `operator_transportir` (`nomor_polisi` text NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL)
10.	CREATE TABLE `pengembalian_stok_rusak` (`id` varchar(255) NOT NULL, `nomor_polisi` varchar(255) NOT NULL, `nama_supir` varchar(255) NOT NULL, `status` varchar(255) NOT NULL, `tanggal_berangkat` date NOT NULL, `tanggal_sampai` date DEFAULT NULL, `created_at` datetime NOT NULL, `updated_at` datetime NOT NULL)
11.	CREATE TABLE `penugasan_stok_opname` (`id` varchar(20) NOT NULL, `deadline` date NOT NULL, `id_operator_gudang_penyangga` varchar(255) NOT NULL, `administrasi` varchar(255) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL,

	`id_operator_stok_opname` varchar(20) DEFAULT NULL, `tanggal_stok_opname` date DEFAULT NULL, `jam_mulai` time DEFAULT NULL, `jam_selesai` time DEFAULT NULL, `status` varchar(255) NOT NULL)
12.	CREATE TABLE `permintaan_pupuk` (`id` varchar(20) NOT NULL, `deadline` date NOT NULL, `qty` int(11) NOT NULL, `status` varchar(255) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id_operator_transportir` varchar(255) NOT NULL, `id_operator_gudang_penyangga` varchar(20) DEFAULT NULL, `id_pupuk` varchar(20) DEFAULT NULL)
13.	CREATE TABLE `pupuk` (`kemasan` int(11) NOT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL)
14.	CREATE TABLE `rekanan_gudang_penyangga` (`alamat_kantor` varchar(120) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id` varchar(20) NOT NULL, `nama` varchar(120) DEFAULT NULL, `username` varchar(120) DEFAULT NULL, `password` varchar(120) DEFAULT NULL)
15.	CREATE TABLE `stok` (`id` varchar(20) NOT NULL, `qty` double NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id_pupuk` varchar(20) DEFAULT NULL, `id_operator_gudang_penyangga` varchar(20) DEFAULT NULL)
16.	CREATE TABLE `stok_keluar` (`id` varchar(20) NOT NULL, `nomor_polisi` varchar(120) NOT NULL, `distributor` varchar(120) NOT NULL, `qty` double NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id_stok` varchar(20) DEFAULT NULL)
17.	CREATE TABLE `stok_kritis` (`id` varchar(20) NOT NULL, `id_daerah` varchar(255) NOT NULL,

	`qty` double NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id_pupuk` varchar(20) DEFAULT NULL)
18.	CREATE TABLE `stok_opname` (`id` varchar(20) NOT NULL, `total_hitung_ton` double NOT NULL, `total_hitung_kemasan` int(11) NOT NULL, `qty_sistem` double NOT NULL, `id_pupuk` varchar(20) NOT NULL, `id_penugasan_stok_opname` varchar(20) NOT NULL, `status` varchar(255) NOT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL)
19.	CREATE TABLE `stok_rusak` (`id` varchar(20) NOT NULL, `id_pengembalian` varchar(255) DEFAULT NULL, `qty` double NOT NULL, `alasan` varchar(400) NOT NULL, `foto` text NOT NULL, `status` varchar(255) DEFAULT NULL, `catatan_diswil` text DEFAULT NULL, `created_at` datetime DEFAULT NULL, `updated_at` datetime DEFAULT NULL, `id_stok` varchar(20) DEFAULT NULL)

5.2.2.2 Physical Data Model (PDM)

Gambar 5.14 menggambarkan hasil implementasi basis data yang telah dibuat dan diilustrasikan dalam bentuk *Physical Data Model.* Dari hasil implementasi basis data diperoleh 19 tabel yaitu manajer diswil, stok kritis, pengembalian stok rusak, stok rusak, pupuk, daerah, permintaan pupuk, detail permintaan pupuk, operator transportir, operator gudang penyangga, stok, stok keluar, stok opname, penugasan stok opname, rekanan gudang penyangga, operator stok opname, operator gudang pusat, operator gudang diswil, dan detail stok opname



Gambar 5.14 Physical Data Model

5.2.3 Implementasi Kode Program

Pada fase implementasi kode program adalah mengubah hasil perancangan algoritme ke dalam bahasa komputer dengan menggunakan beberapa bahasa pemrograman yaitu PHP, HTML, CSS, Javascript dan Ajax serta menggunakan *framework CodeIgniter*.

5.2.3.1 Implementasi Fungsi save_detail_permintaan_pupuk

Fungsi save_detail_permintaan_pupuk adalah fungsi yang terdapat pada kelas Operator_transportir_c yang dipanggil ketika Operator Rekanan Transportir menyimpan data detail permintaan pupuk sebagai tanggapan atas permintaan penggiriman dari Operator Diswil 1. Fungsi save_detail_permintaan_pupuk dapat dilihat pada Kode Program 5.4.

Nama Class: Operator_transportir_c
Nama Method: save_detail_permintaan_pupuk
Kode Program:

```
1   public function save_detail_permintaan_pupuk($id){

2       $this->verify();
3       $daftar_nama_supir = $this->input->post('nama_supir');
4       $daftar_nomor_polisi = $this->input-
    >post('nomor_polisi');
5       $daftar_qty_rencana = $this->input-
    >post('qty_rencana');
6       $id_permintaan_pupuk = $id;

7       $panjang_data = count($daftar_nama_supir);

8       for($i = 0; $i < $panjang_data; $i++){
9           $status = $this->operator_transportir-
    >check_nomor_polisi($this->username,
    $daftar_nomor_polisi[$i]);
10          if($status == false){
11              $result_json = array('status' => false);
12              echo json_encode($result_json);
13              break;
14          }else if($panjang_data == $i + 1){
15              for ($i = 0; $i < $panjang_data; $i++) {
16                  $id_detail_permintaan_pupuk = 'dpp' .
    date('Ymdhis');
17                  $nama_supir = $daftar_nama_supir[$i];
18                  $nomor_polisi = $daftar_nomor_polisi[$i];
19                  $qty_rencana = $daftar_qty_rencana[$i];
20                  $status = 'transportir ---> pusat';
21                  $tanggal_berangkat = date('Y-m-d');
22                  $created_at = date('Y-m-d H:i:s');
23                  $updated_at = date('Y-m-d H:i:s');
24                  $data = array('id' =>
    $id_detail_permintaan_pupuk, 'id_permintaan_pupuk' =>
    $id_permintaan_pupuk, 'qty_rencana' => $qty_rencana,
    'nama_supir' => $nama_supir, 'nomor_polisi' =>
    $nomor_polisi, 'tanggal_berangkat' => $tanggal_berangkat,
    'status' => $status, 'created_at' => $created_at,
    'updated_at' => $updated_at);
25                  sleep(1);
26                  $this->operator_transportir-
    >insert_detail_permintaan_pupuk($data);
27              }
28                  $this->operator_transportir-
    >update_status_permintaan_pupuk($id_permintaan_pupuk);
29                  $result_json = array('status' => true);
30                  echo json_encode($result_json);
31          }
32      }
33  }
34
```

Kode Program 5.4 *Source Code* Fungsi save_detail_permintaan_pupuk

5.2.3.2 Implementasi Fungsi set_kml

Fungsi set_kml adalah fungsi yang terdapat pada kelas Peta_stok_c fungsi ini digunakan saat Aktor (Manajer Diswil 1 dan Operator Diswil 1) membuka menu halaman kondisi stok daerah dan sistem memperlihatkan peta kondisi stok daerah. Fungsi set_kml dapat dilihat pada Kode Program 5.5.

Nama Class: Peta_stok_c
Nama Method: set_kml
Kode Program:

```
1  public function set_kml(){
2      $warna = 'ffbdbdbd';
3      $peta_stok = $this->peta_stok-
   >select_perhituingan_peta_stok();
4      $data = '<?xml version="1.0" encoding="UTF-8"?>
         <kml xmlns="http://www.opengis.net/kml/2.2">
         <Document>
               <name>Tempatku.kml</name>
               ';
5      $id_daerah = '';
6      $status_per_daerah = true;
7      $count = 0;
8        for($i = 0; $i < count($peta_stok); $i++){
9          $warna = 'ffbdbdbd';
10         if($count == 0){
11             $id_daerah = $peta_stok[$count]->id_daerah;
12         }
13         if($peta_stok[$count]->qty < $peta_stok[$count]-
   >kritis){
14             $status_per_daerah = false;
15         }
16         if($count < count($peta_stok) - 1){
17             if($id_daerah != $peta_stok[$count+1]-
   >id_daerah){
18                 if($status_per_daerah){
19                     $warna = 'ff589d0f';
20                 }else{
21                     $warna = 'ff1427a5';
22                 }
23                 $data .=
                   '<Style id="'.$peta_stok[$count]-
   >id_daerah.'">
                       <LineStyle>
                           <color>ff000000</color>
                           <width>1.5</width>
                       </LineStyle>
                       <PolyStyle>
                           <color>'.$warna.'</color>
                       </PolyStyle>
                   </Style>';
24                 $id_daerah = $peta_stok[$count+1]-
   >id_daerah;
25                 $status_per_daerah = true;
26             }
27         }else if($count == count($peta_stok) - 1){
```

```
28              if($status_per_daerah){
29                  $warna = 'ff589d0f';
30              }else{
31                  $warna = 'ff1427a5';
32              }
33              $data .=
                '<Style id="'.$peta_stok[$count]->id_daerah.'">
                    <LineStyle>
                        <color>ff000000</color>
                        <width>1.5</width>
                    </LineStyle>
                    <PolyStyle>
                        <color>'.$warna.'</color>
                    </PolyStyle>
                </Style>';
34              $status_per_daerah = true;
35          }
36          $count++;
37      }
38        $status_delete = $this->delete_directory_kml();
39        $nama_kml = md5('peta'.date('Ymdhis'));
40        $data_lokasi =
    read_file("./assets/uploads/datalayer.txt");
41        $status_write =
    write_file("./assets/uploads/kml/".$nama_kml.".kml", $data,
    "wb");
42        $status_write_append =
    write_file("./assets/uploads/kml/".$nama_kml.".kml",
    $data_lokasi, "a");
43      $return_json = array(
44              'status' => true,
45              'src' =>
    "https://skripsi.ridiansyah.com/assets/uploads/kml/".$nama_k
    ml.".kml"
46        );
47        echo json_encode($return_json);
48  }
```

Kode Program 5.5 Source Code Fungsi set_kml

5.2.3.3 Implementasi Fungsi save_detail_stok_opname

Fungsi save_detail_stok_opname adalah fungsi yang terdapat pada kelas Operator_stok_opname_c yang dipanggil ketika Operator Stok Opname menyimpan data hasil perhitungan stok opname perpupuk setelah melakukan perhitungan. Fungsi save_detail_stok_opname dapat dilihat pada Kode Program 5.6.

Nama Class: Operator_stok_opname_c
Nama Method: save_detail_stok_opname
Kode Program:

```
1   public function save_detail_stok_opname($id_stok_opname){
2       $this->verify();
3       $detail = $this->input->post('detail_stok_opname');
4       $total_hitung_kemasan          =          $this->input-
    >post('total_hitung_kemasan');
```

```
5       $total_hitung_ton = $this->input->post('total_hitung_ton');

6           for($i = 0; $i < count($detail); $i++){
7               $id = 'dso'.date('Ymdhis');
8               $zona = $detail[$i][0];
9               $kunci = $detail[$i][1];
10              $panjang = $detail[$i][2];
11              $lebar = $detail[$i][3];
12              $tinggi = $detail[$i][4];
13              $jumlah_kemasan = $detail[$i][5];
14              $jumlah_ton = $detail[$i][6];
15              $lebihan_kemasan = $detail[$i][7];
16              $lebihan_ton = $detail[$i][8];;
17              $total_kemasan = $detail[$i][9];
18              $total_ton = $detail[$i][10];

19              $created_at = date('Y-m-d H:i:s');
20              $updated_at = date('Y-m-d H:i:s');

21              $this->operator_stok_opname->insert_detail_stok_opname(
22                  array(
23                      'id' => $id,
24                      'id_stok_opname' => $id_stok_opname,
25                      'zona' => $zona,
26                      'kunci' => $kunci,
27                      'panjang' => $panjang,
28                      'lebar' => $lebar,
29                      'tinggi' => $tinggi,
30                      'lebihan_kemasan' => $lebihan_kemasan,
31                      'lebihan_ton' => $lebihan_ton,
32                      'jumlah_kemasan' => $jumlah_kemasan,
33                      'jumlah_ton' => $jumlah_ton,
34                      'total_kemasan' => $total_kemasan,
35                      'total_ton' => $total_ton,
36                      'created_at' => $created_at,
37                      'updated_at' => $updated_at
38                  ));
39                  sleep(1);
40          }
41          $data = array(
42              'total_hitung_kemasan' => $total_hitung_kemasan,
43              'total_hitung_ton' => $total_hitung_ton,
44              'status' => 'selesai',
45              'updated_at' => date('Y-m-d H:i:s')
46          );
47          $where = array(
48              'id' => $id_stok_opname
49          );
50          $this->operator_stok_opname->update_stok_opname($data, $where);
51          $result_json = array('status' => true);
52          echo json_encode($result_json);
53  }
```

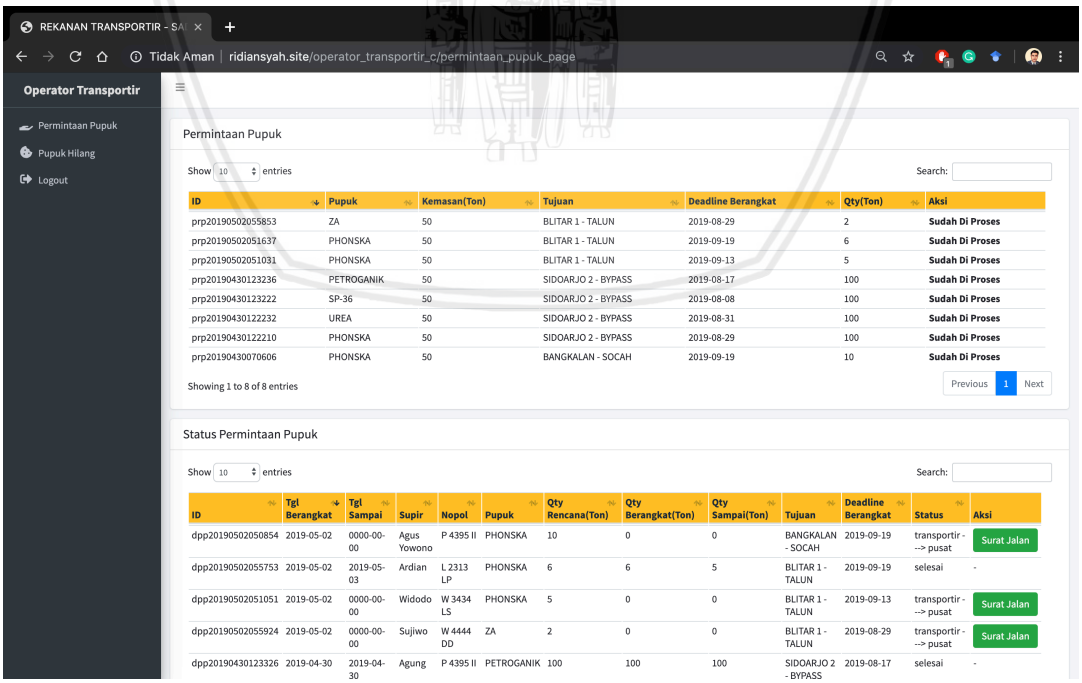
Kode Program 5.6 Source Code Fungsi save_detail_stok_opname

5.2.4 Implementasi Antarmuka

Pada tahap implementasi antarmuka ini dibuat antarmuka sebagai media interaksi antara pengguna dan sistem. Implementasi antarmuka dibuat dengan mangacu pada perancangan antaramuka yang telah dibuat pada tahap perancangan antarmuka.

5.2.4.1 Implementasi Antarmuka Halaman Menu Permintaan Pupuk

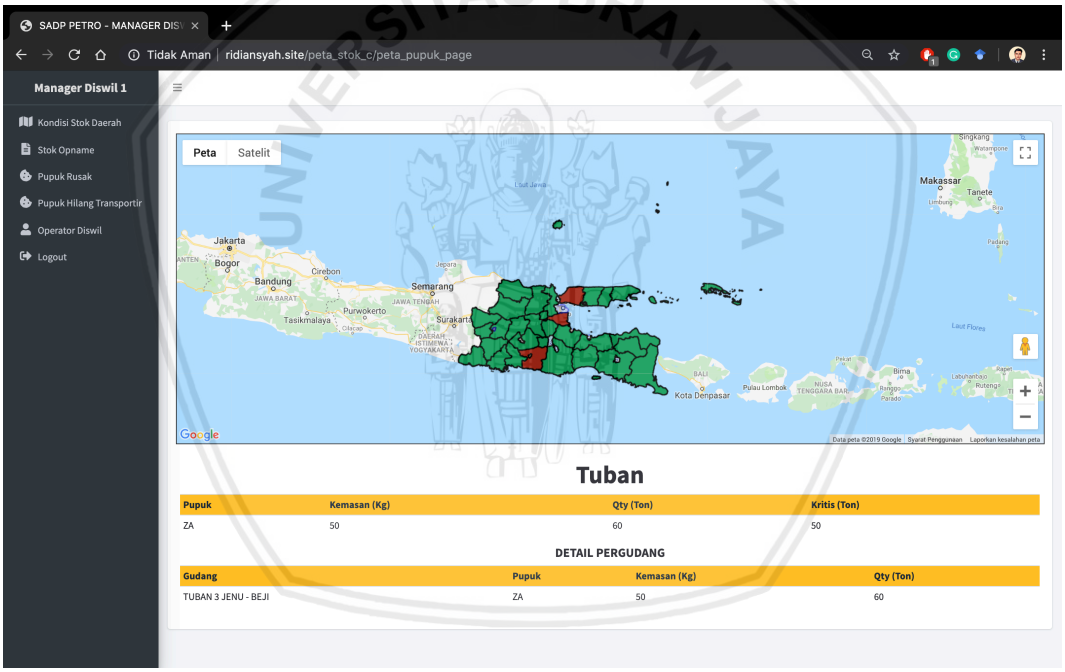
Halaman menu permintaan pupuk tersedia untuk beberapa aktor namun dari setiap aktor memiliki antarmuka dan penampilan data yang berbeda. Pada implementasi antarmuka ini ditujukan untuk halaman menu permintaan pupuk untuk aktor Operator Transportir. Halaman ini berfungsi untuk memperlihatkan data permintaan pupuk dan menanggapi permintaan pupuk dari operator diswil 1. Terdapat 2 tabel yaitu tabel permintaan pupuk dan tabel Status permintaan pupuk. Tabel permintaan pupuk menyajikan data permintaan pupuk oleh operator diswil 1 kepada operator transportir tersebut, data yang ditampilkan pada tabel permintaan pupuk adalah nomor, pupuk, kemasan, tujuan, deadline berangkat, qty (ton) dan aksi. Sedangkan untuk tabel status permintaan pupuk menyajikan data status kendaraan atau truk yang telah ditugaskan untuk mengangkut pupuk ke gudang penyangga, data yang ditampilkan pada tabel status permintaan pupuk adalah nomor, tanggal berangkat, tanggal sampai, supir, nomor polisi, pupuk, qty rencana qty berangkat, qty sampai, tujuan, status dan aksi. Hasil perancangan antarmuka halaman menu permintaan pupuk dipresentasikan pada Gambar 5.15.



Gambar 5.15 Implementasi Antarmuka Permintaan Pupuk

5.2.4.2 Impelementasi Antarmuka Halaman Menu Kondisi Stok Daerah

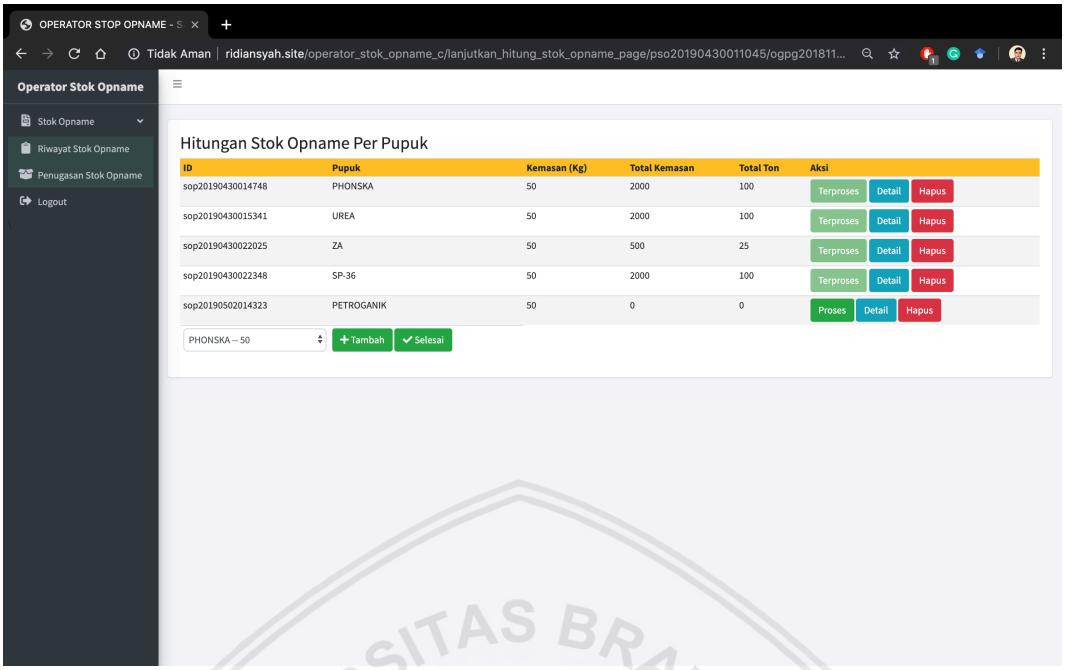
Halaman menu kondisi stok daerah ini tersedia untuk 2 aktor yaitu operator diswil 1 dan manajer diswil 1. Halaman ini berfungsi untuk memperlihatkan data kondisi stok daerah dengan menggunakan google maps API dilengkapi dengan layering KML. Data informasi yang disediakan dalam halaman ini adalah indikator warna pada wilayah kabupaten kota di Jawa Timur yang mengambarkan kondisi stok daerah, warna merah menggambarkan ada jumlah stok pupuk tersedia di daerah tersebut dibawah batas stok kritis yang telah ditentukan, sedangkan warna hijau mengambarkan bahwa jumlah stok pupuk yang tersedia didaerah tersebut diatas batas stok kritis. Selain itu dilengkapi juga dengan 2 tabel yaitu tabel yang berisi informasi mengenai jumlah total masing-masing jenis pupuk yang tersedia di daerah yang dipilih dan tabel yang berisi detail jumlah stok tiap jenis pupuk ditiap gudang penyangga. Hasil perancangan antarmuka halaman menu kondisi stok daerah dipresentasikan pada Gambar 5.16.



Gambar 5.16 Implementasi Antarmuka Kondisi Stok Daerah

5.2.4.3 Implementasi Antarmuka Halaman Hitung Stok Opname

Halaman hitung stok opname ini berfungsi untuk menambah pupuk yang akan dihitung oleh operator stok opname. Tersedia tabel yang berisi data pupuk yang telah ditambahkan, data yang ditampilkan yaitu nomor, pupuk, kemasan, total kemasan, total ton, dan beberapa *button* aksi. Hasil perancangan antarmuka halaman hitung stok opname dipresentasikan pada Gambar 5.17.



Gambar 5.17 Implementasi Antarmuka Hitung Stok Opname

BAB 6 PENGUJIAN

Fase pengujian ini akan dijabarkan pengujian perangkat lunak yang telah dilakukan terhadap Sistem Aplikasi Manajemen Distribusi Pupuk Berbasis Web begitu juga dengan analisis pengujian yang dilakukan setelah semua pengujian sistem telah berhasil diselesaikan. Fase ini dikerjakan pada saat telah menyelesaikan fase implementasi dari sistem. Sistem akan dilakukan beberapa pengujian dengan menggunakan metode kotak putih dan kotak hitam.

6.1 Pengujian Unit

Pengujian unit dilakukan untuk menguji unit seperti method dari suatu kelas dari hasil perancangan sistem. Pengujian dilakukan menggunakan salah satu metode pada *White Box Testing* yaitu *basis path testing*. Pengujian unit dilakukan dalam tiga sampel uji yaitu pada klas Operator_transportir_c, Peta_stok_c dan operator_stok_opname_c.

6.1.1 Pengujian Unit Klas Operator_transportir_c pada fungsi save_detail_permintaan_pupuk

1. Pseudocode

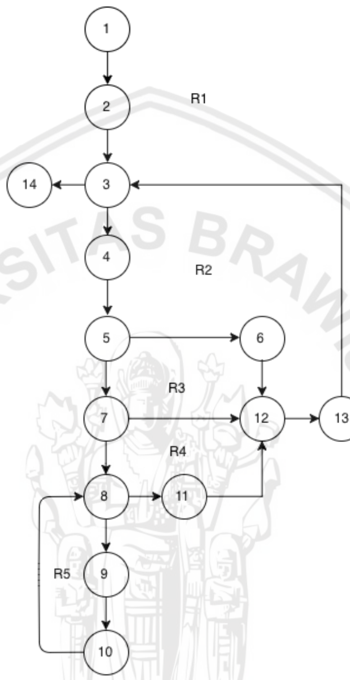
```
START
CALL: function verify
INIT daftar_nama_supir = POST[nama_supir]
INIT daftar_nomor_polisi = POST[nomor_polisi]
INIT daftar_qty_rencana = POST[qty_rencana]
INIT id_permintaan_pupuk = id
INIT panjang_data = COUNT[daftar_nama_supir]
LOOP (i sampai panjang_data)
    INIT status = hasil function check nomor_polisi
    IF (status == false) THEN
        INIT array result_json
        ECHO result_json
        BREAK
    ELSE IF (panjang_data == i+1) THEN
        LOOP (i sampai panjang_data)
            INIT id_detail_permintaan_pupuk =
"dpp"+date(ymdhis)
            INIT nama_supir = daftar_nama_supir[array
index i]
            INIT nomor_polisi = daftar_nomor_polisi[array
index i]
            INIT qty_rencana = daftar_qty_rencana[array
index i]
            INIT status = "transportir --> pusat"
            INIT tanggal_berangkat = date(year-month-day)
            INIT created_at = date(y-m-d h:i:s)
            INIT updated_at = date(y-m-d h:i:s)
            INIT array data
            sleep(1 detik)
            CALL: insert_detail_permintaan_pupuk(data)
        END LOOP
```

```
                CALL: update_status_permintaan_pupuk
(id_permintaan_pupuk)                                   11
                INIT array result_json
                ECHO result_json
     END IF                                             12
END LOOP                                                13
END                                                     14
```

Kode Program 6.1 Pseudocode save_detail_permintaan_pupuk

2. Basic Path Testing

A. Flow Graph



Gambar 6.1 Flow Graph save_detail_permintaan_pupuk

B. Cyclomatic Complexity

1. Region : 5
2. Node : 14
3. Edge : 17
   i. V(G) = 5, ada 5 Region = R1, R2, R3, R4, R5
   ii. V(G) = Edges – Node + 2 = 17 – 14 + 2 = 5
   iii. V(G) = Predicates nodes + 1 = 4 + 1 = 5

C. Independent Path

1. Path 1 = 1 – 2 – 3 – 14
2. Path 2 = 1 – 2 – 3 – **4 – 5 – 6 – 12 – 13** – 3 – 13
3. Path 3 = 1 – 2 – 3 – 4 – 5 – **7** – 12 – 13 – 3 – 13
4. Path 4 = 1 – 2 – 3 – 4 – 5 – 7 – **8 – 11** – 12 – 13 – 3 – 13

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

5. Path 5 = 1 – 2 – 3 – 4 – 5 – 7 – 8 – **9 – 10 – 8** – 11 – 12 – 13 – 3 – 13

Tabel 6.1 Hasil Pengujian Unit pada Fungsi save_detail_permintaan_pupuk

No.	No. Jalur	Prosedur Uji	Expected Result	Result	Status
1	Path 1	Inialisasi var panjang_data = 0	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil *null*	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil *null*	Valid
2	Path 2	Inialisasi var panjang_data = 0 Inialisasi var status = false	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = false	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = false	Valid
3	Path 3	Inialisasi var panjang_data = 1 Inialisasi var status = true Inialisasi var panjang_data = 0 dibawah var status	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil *null*	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil *null*	Valid
4	Path 4	Inialisasi var panjang_data = 1 Inialisasi var status = true Inialisasi var i = 1	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid
5	Path 5	Inialisasi var panjang_data = 1 Inialisasi var status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan	*Coverage* sesuai dengan *independent path* dan	Valid

			hasil json status = true	menghasilkan hasil json status = true	

6.1.2 Pengujian Unit Klas Peta_stok_c pada fungsi set_kml

1. Pseudocode

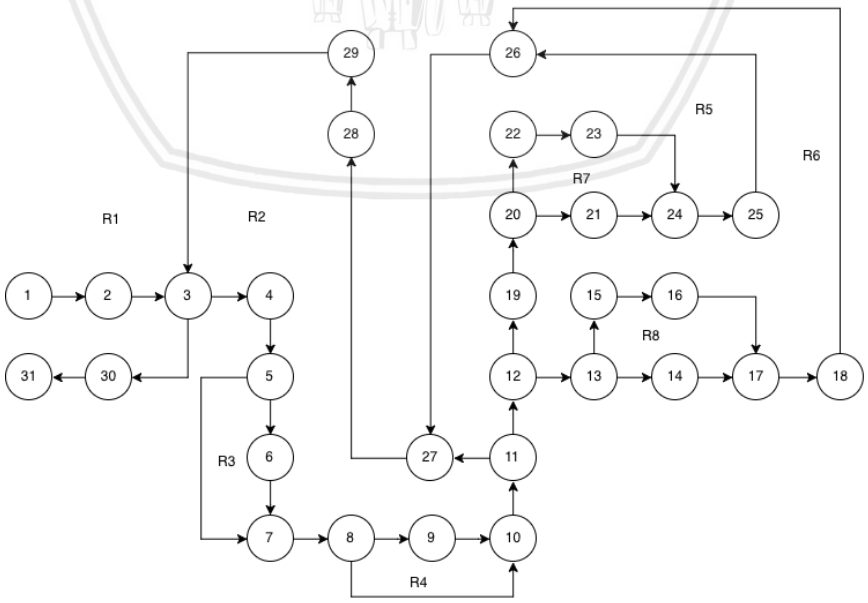
```
1   START
2   INIT warna = 'ffbdbdbd'
3   INIT peta_stok= hasil function select_perhitungan_peta_stok
4   INIT data = '<?xml version="1.0" encoding="UTF-8"?>
                <kml xmlns="http://www.opengis.net/kml/2.2">
                <Document>
                      <name>Tempatku.kml</name>'
5   INIT id_daerah = ''
6   INIT status_per_daerah = true
7   INIT count = 0
8   LOOP (i sampai count(peta_stok)
9       INIT warna = 'ffbdbdbd'
10      IF (count == 0) THEN
11         INIT id_daerah = id_daerah[array peta_stok index
    count]
12      END IF
13      IF (qty[array peta_stok index count] < kritis[array
    peta_stok index count]) THEN
14         INIT status_per_daerah = false
15      END IF
16      IF (count < count(peta_stok) - 1) THEN
           IF (id_daerah != id_deerah[array peta_stok index
17  count+1]
              IF (status_per_daerah) THEN
18               INIT warna = 'ff589d0f'
19            ELSE
20               INIT warna = 'ff1427a5'
21            END IF
22            INIT data .= '<Style id="'.$peta_stok[$count]-
23  >id_daerah.'">
                            <LineStyle>
                                <color>ff000000</color>
                                <width>1.5</width>
                            </LineStyle>
                            <PolyStyle>
                                <color>'.$warna.'</color>
                            </PolyStyle>
                        </Style>'
              INIT id_daerah = id_daerah[array peta_stok index
24  count+1]
              INIT status_per_daerah = true
25        ELSE IF (count == count(peta_stok)-1) THEN
26           IF (status_per_daerah) THEN
27              INIT warna = 'ff589d0f'
28           ELSE
29              INIT warna = 'ff1427a5'
30           END IF
31
```

```
32              INIT data .= '<Style id="'.$peta_stok[$count]-
    >id_daerah.'">
33                          <LineStyle>
34                              <color>ff000000</color>
35                              <width>1.5</width>
36                          </LineStyle>
37                          <PolyStyle>
38                              <color>'.$warna.'</color>
39                          </PolyStyle>
40                      </Style>'
41              INIT status_per_daerah = true
42          END IF
43      END IF
44          count++
45  END LOOP
46  INIT status_delete = call dan get return function
    delete_directory_kml
47  INIT nama_kml = md5(peta.date(ymdhis))
48  INIT data_lokasi = read_file
    ("./assets/uploads/datalayer.txt")
49  INIT status_write = write_file ("./assets/uploads/kml/".
    variable nama_kml.".kml", $data, "wb")
50  INIT status_write_append = write_file
    ("./assets/uploads/kml/".variable nama_kml.".kml", variable
    data_lokasi, "a")
51  INIT array return_json
    variable nama_kml.".kml)
52  ECHO return_json
53  END
```

Kode Program 6.2 Pseudocode set_kml

2. Basic Path Testing

A. Flow Graph



Gambar 6.2 Flow Graph set_kml

B. Cyclomatic Complexity

1. Region : 8
2. Node : 31
3. Edge : 37
    i. V(G) = 8, ada 8 Region = R1, R2, R3, R4, R5, R6, R7, R8
    ii. V(G) = Edges – Node + 2 = 37 – 31 + 2 = 8
    iii. V(G) = Predicates nodes + 1 = 7 + 1 = 8

C. Independent Path

1. Path 1 = 1 – 2 – 3 – 30 – 31
2. Path 2 = 1 – 2 – **3** – **4** – **5** – **7** – **8** – **9** – **10** – **11** – **27** – **28** – **29** – **3** – 30 – 31
3. Path 3 = 1 - 2 - 3 - 4 - 5 - **6** - 7 - 8 - 10 - 11 - 27 - 28 - 29 - 3 - 30 - 31
4. Path 4 = 1 - 2 - 3 - 4 - 5 – 6 – 7 - 8 **- 10 - 11 - 27 - 28 - 29 - 3 - 30 - 31**
5. Path 5 = 1 - 2 - 3 - 4 - 5 – 6 - 7 - 8 **– 10 – 11 – 12 – 13 – 14 – 17 – 18 – 26 – 27 – 28 – 29 – 3 – 30 – 31**
6. Path 6 = 1 - 2 - 3 - 4 - 5 – 6 - 7 - 8 – 10 – 11 – 12 – 13 – **15 – 16** – 17 – 18 – 26 – 27 – 28 – 29 – 3 – 30 – 31
7. Path 7 = 1 - 2 - 3 - 4 - 5 – 6 - 7 - 8 – 10 – 11 – 12 – **19 – 20 – 21 – 24 – 25** – 26 – 27 – 28 – 29 – 3 – 30 – 31
8. Path 8 = 1 - 2 - 3 - 4 - 5 – 6 - 7 - 8 – 10 – 11 – 12 – 19 – 20 – **22 – 23** – 24 – 25 – 26 – 27 – 28 – 29 – 3 – 30 – 31

Tabel 6.2 Hasil Pengujian Unit pada Fungsi set_kml

No.	No. Jalur	Prosesur Uji	Expected Result	Result	Status
1	Path 1	Inialisasi var peta_stok = []	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid
2	Path 2	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang',	*Coverage* sesuai dengan *independent path* dan menghasilkan	*Coverage* sesuai dengan *independent path* dan menghasilkan	Valid

		'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 10, 'kritis' => 50)); Inialisasi var count = 1 Menambah -1 pada array indeks count	hasil json status = true	hasil json status = true	
3	Path 3	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang', 'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 10, 'kritis' => 50)); Menambah – 1 pada count	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid
4	Path 4	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang', 'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 100, 'kritis' => 50)); Menghapus -1 pada count(peta_stok)	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid
5	Path 5	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang', 'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 100,	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid

		'kritis' => 50) (object) array('id_daerah' => ‘drh2018120403275’, 'nama_daerah' => 'Gresik', 'nama' => 'ZA', 'kemasan' => 50, 'qty' => 100, 'kritis' => 50)); Menghapus -1 pada count(peta_stok)			
6	Path 6	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang', 'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 10, 'kritis' => 50) (object) array('id_daerah' => ‘drh2018120403275’, 'nama_daerah' => 'Gresik', 'nama' => 'ZA', 'kemasan' => 50, 'qty' => 10, 'kritis' => 50)); Menghapus -1 pada count(peta_stok)	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid
7	Path 7	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang', 'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 100,	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid

		'kritis' => 50) (object) array('id_daerah' => 'drh2018120403275', 'nama_daerah' => 'Gresik', 'nama' => 'ZA', 'kemasan' => 50, 'qty' => 100, 'kritis' => 50)); Menambah Inialisasi var count=1			
8	Path 8	Inialisasi var peta_stok = array((object) array('id_daerah' => 'drh20181211060154', 'nama_daerah' => 'Malang', 'nama' => 'PHONSKA', 'kemasan' => 50, 'qty' => 10, 'kritis' => 50) (object) array('id_daerah' => 'drh2018120403275', 'nama_daerah' => 'Gresik', 'nama' => 'ZA', 'kemasan' => 50, 'qty' => 10, 'kritis' => 50)); Menambah Inialisasi var count=1	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid

6.1.3 Pengujian Unit Klas Operator_stok_opname_c pada fungsi save_detail_stok_opname

1. Pseudocode

```
1  START                                                    1
2  CALL: function verify
3  INIT detail = post[detail_stok_opname]
4  INIT total_hitung_kemasan = post[total_hitung_kemasan]   2
5
6  INIT total_hitung_ton = post[total_hitung_ton]
   LOOP (i sampai count(detail))                            3
```

```
7       INIT id = 'dso'.date(ymdhis)
8       INIT zona = array detail index i dan 0
9       INIT kunci = array detail index i dan 1
10      INIT panjang = array detail index i dan 2
11      INIT lebar = array detail index i dan 3
12      INIT tinggi = array detail index i dan 4
13      INIT jumlah_kemasan = array detail index i dan 5
14      INIT jumlah_ton = array detail index i dan 6
15      INIT lebihan_kemasan = array detail index i dan 7
16      INIT lebihan_ton = 0
17      INIT total_kemasan = array detail index i dan 9
18      INIT total_ton = array detail index i dan 10
19      INIT created_at = date(y-m-d h:i:s)
20      INIT updated_at = date(y-m-d h:i:s)
21      CALL: insert_detail_stok_opname(array(id,
    id_stok_opname, zona, kunci, panjang, lebar, tinggi,
    lebihan_kemasan, lebihan_ton, jumlah_kemasan, jumlah_ton,
    total_kemasan, total_ton, created_at, update_at))
22      sleep(1 detik)
23  END LOOP
24  INIT array data
25  INIT where = array(id)
26  CALL: update_stok_opname(data, where)
27  INIT result_json = array(status=true)
28  ECHO result_json encode ke json
29  END
```

4

5

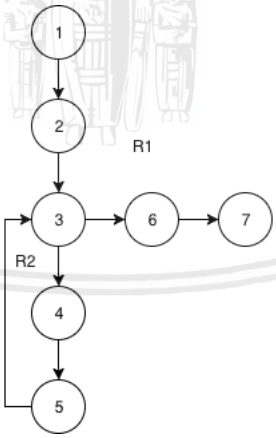
6

7

Kode Program 6.3 Pseudocode save_detail_stok_opname

2. Basic Path Testing

A. Flow Graph



Gambar 6.3 Flow Graph save_detail_stok_opname

B. Cyclomatic Complexity

1. Region : 2
2. Node : 7
3. Edge : 7
   i. V(G) = 2, ada 2 Region = R1, R2
   ii. V(G) = Edges – Node + 2 = 7 – 7 + 2 = 2

iii. V(G) = Predicates nodes + 1 = 1 + 1 = 2

C. Independent Path

1. Path 1 = 1 - 2 - 3 - 6 - 7
2. Path 2 = 1 - 2 - 3 - **4 - 5 - 3** - 6 - 7

Tabel 6.3 Hasil Pengujian Unit pada Fungsi save_detail_stok_opname

No.	No. Jalur	Prosesur Uji	Expected Result	Result	Status
1	Path 1	Inialisasi var detail = [] Inialisasi var id_stok_opname = 'pso20190430125136'	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid
2	Path 2	Inialisasi var detail = [['1', '1', '1', '1', '1', '1', '1', '1', '1', '1','1']] Inialisasi var id_stok_opname = 'pso20190430125136'	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	*Coverage* sesuai dengan *independent path* dan menghasilkan hasil json status = true	Valid

6.2 Pengujian Integrasi

Pengujian integrasi berada pada level high-level design, yaitu relasi antar klas dalam sistem. Teknik yang dipergunakan pada pengujian integrasi ini yakni basis path testing dengan pendekatan top-down. Pengujian integrasi dilakukan pada method update_rekanan_gudang_penyangga pada klas Operator_diswil_c yang di dalamnya memanggil method update_rekanan_gudang_penyangga dari klas Operator_diswil_m.

1. Pseudocode

```
1  START
2  CALL: function verify()
3  INIT nama = POST[nama]
4  INIT alamat_kantor = POST[alamat_kantor]
5  INIT username = POST[username]
6  INIT password = md5 POST[password]
7  INIT update_at = date(tahun-bulan-tanggal jam:menit:detik)
8  INIT array data
9  IF POST[password] == '' THEN
```

```
10  INIT array assosiative data['nama' => nama, 'alamat_kantor'
    => alamat_kantor, 'username' => username, 'updated_at' =>
    updated_at]
11  ELSE
12  INIT array assosiative data['nama' => nama, 'alamat_kantor'
    => alamat_kantor, 'username' => username, 'password' =>
    password, 'updated_at' => updated_at]
13  END IF
14  INIT array assosiative where['id' => id]
15  CALL: update_rekanan_gudang_penyangga(data, where)
16  INIT array assosiative result_json['status' => status]
17  ECHO json_encode(result_json)
18  END
```

Kode Program 6.4 Pseudocode update_rekanan_gudang_penyangga

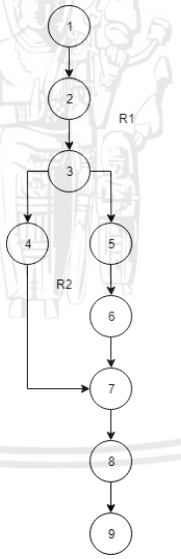
2. Method Stub

```
1  public function stub_updatepenyangga($data, $where){
2          return true;
3  }
```

Kode Program 6.5 Method Stub Yang Digunakan

3. Basic Path Testing

A. Flow Graph



Gambar 6.4 Flow Graph update_rekanan_gudang_penyangga

B. Cyclomatic Complexity

1. Region: 3
2. Node: 9
3. Edge: 10

i. V(G) = 2, ada 2 Region = R1, R2

ii. V(G) = Edges – Node + 2 = 9 - 9 + 2 = 2

iii. V(G) = Predicates nodes + 1 = 1 + 1 = 2

C. Independent Path

1. Path 1 = 1 – 2 – 3 – 4 – 7 – 8 – 9
2. Path 2 = 1 – 2 – 3 – **5 – 6** – 7 – 8 – 9

Tabel 6.4 Hasil Pengujian Unit pada Fungsi update_rekanan_gudang_penyangga

No. Jalur	Prosedur Uji	Expected Result	Result	Status
Path 1	Inialisasi var password = " " Membuat method stub_updatepenyangga dengan return true	Menghasilkan hasil json status = true	Menghasilkan hasil json status = true	Valid
Path 2	Inialisasi var password = " password" Membuat_method stub_updatepenyangga dengan return true	Menghasilkan hasil json status = true	Menghasilkan hasil json status = true	Valid

6.3 Pengujian Validasi

Dalam rangka memvalidasi fungsi-fungsi yang telah terdapat didalam sistem aplikasi manajemen distribusi pupuk dilakukan pengujian validasi dengan menggunakan metode *scenario-based testing* dengan prosedur pengujian yang didapatkan dari use case skenario lalu kemudian dilihat apakah hasil atau keluran sesuai dengan yang diharapkan atau tidak. Pengujian validasi ini termasuk dalam pengujian kotak hitam. Pengujian tentunya dilakukan terhadap semua fungsi atau kebutuhan yang ada yaitu 62 kebutuhan.

6.3.1 Pengujian Validasi Melihat Jenis Pupuk

Dalam *test-case* melihat jenis pupuk terdiri dari satu *test-case* yakni, *test-case* berhasil melihat jenis pupuk yang dapat diperlihatkan didalam tabel 6.5.

A. *Test-case* Berhasil Melihat Jenis Pupuk

Tabel 6.5 *Test-case* Berhasil Melihat Jenis Pupuk

Nama *Test-case*	Berhasil Melihat Jenis Pupuk
Prosedur	1. Mengklik kategori menu pupuk. 2. Mengklik menu jenis pupuk.

Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu jenis pupuk dan memperlihatkan data tabel jenis pupuk.
Hasil	Sistem memperlihatkan halaman menu jenis pupuk dan memperlihatkan data tabel jenis pupuk.
Status	Valid

6.3.2 Pengujian Validasi Menambah Jenis Pupuk

Dalam test-case menambah jenis pupuk terdapat dua *test-case* yaitu *test-case* berhasil menambah jenis pupuk yang dapat diperlihatkan didalam tabel 6.6, dan *test-case* menambah jenis pupuk alternatif 1: jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.7.

A. *Test-case* Berhasil Menambah Jenis Pupuk

Tabel 6.6 *Test-case* Berhasil Menambah Jenis Pupuk

Nama *Test-case*	Berhasil Menambah Jenis Pupuk
Prosedur	1. Mengklik *button* tambah 2. Mengisi data tambah pupuk yaitu nama pupuk, dan kemasan pupuk (kg). 3. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan jenis pupuk telah tertambah pada tabel jenis pupuk.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan jenis pupuk telah tertambah pada tabel jenis pupuk.
Status	Valid

B. *Test-case* Menambah Jenis Pupuk Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.7 *Test-case* Menambah Jenis Pupuk Alternatif 1

Nama *Test-case*	Menambah Jenis Pupuk Alternatif 3: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1 2. Mengklik menu jenis pupuk. 3. Mengklik *button* tambah 4. Tidak mengisi salah satu atau semua data tambah pupuk yaitu nama pupuk, dan kemasan pupuk (kg). 5. Mengklik *button* simpan.

Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.3 Pengujian Validasi Mengubah Jenis Pupuk

Dalam test-case mengubah jenis pupuk terdapat satu *test-case* yaitu *test-case* berhasil mengubah jenis pupuk yang dapat diperlihatkan didalam tabel 6.8.

A. *Test-case* Berhasil Mengubah Jenis Pupuk

Tabel 6.8 *Test-case* Berhasil Mengubah Jenis Pupuk

Nama *Test-case*	Berhasil Mengubah Jenis Pupuk
Prosedur	1. Telah login sebagai operator diswil 1 2. Mengklik menu jenis pupuk. 3. Mengklik *button* edit pada data pupuk yang ingin diubah. 4. Mengisi data pupuk yang ingin diubah yaitu nama pupuk, dan kemasan pupuk (kg). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil disimpan dan data jenis pupuk telah terubah pada tabel jenis pupuk.
Hasil	Sistem memperlihatkan pesan data berhasil disimpan dan data jenis pupuk telah terubah pada tabel jenis pupuk.
Status	Valid

6.3.4 Pengujian Validasi Melihat Rekanan Transportir

Dalam test-case melihat rekanan transportir terdapat satu *test-case* yaitu *test-case* berhasil melihat rekanan transportir yang dapat diperlihatkan didalam tabel 6.9.

A. *Test-case* Berhasil Melihat Rekanan Transportir

Tabel 6.9 *Test-case* Berhasil Melihat Rekanan Transportir

Nama *Test-case*	Berhasil Melihat Rekanan Transportir.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan transportir.

Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu rekanan transportir dan tabel rekanan transportir.
Hasil	Sistem memperlihatkan halaman menu rekanan transportir dan tabel rekanan transportir.
Status	Valid

6.3.5 Pengujian Validasi Menambah Rekanan Transportir

Dalam test-case menambah rekanan transportir terdapat dua *test-case* yaitu *test-case* berhasil menambah rekanan transportir yang dapat diperlihatkan didalam tabel 6.10, dan *test-case* menambah rekanan transportir alternatif 1: jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.11.

A. *Test-case* Berhasil Menambah Rekanan Transportir

Tabel 6.10 *Test-case* Berhasil Menambah Rekanan Transportir

Nama *Test-case*	Berhasil Menambah Rekanan Transportir.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan transportir. 3. Mengklik *button* tambah. 4. Mengisi data yang ada pada form yaitu nama perusahaan, username, password, konfirmasi password, nomor polisi. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan, dan data rekanan transportir telah tertambahkan pada tabel.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan, dan data rekanan transportir telah tertambahkan pada tabel.
Status	Valid

B. *Test-case* Menambah Rekanan Transportir Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.11 *Test-case* Menambah Rekanan Transportir Alternatif 1

Nama *Test-case*	Menambah Rekanan Transportir Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1

	2. Mengklik menu rekanan transportir 3. Mengklik *button* tambah. 4. Tidak mengisi data yang ada pada form yaitu nama perusahaan, username, password, konfirmasi password, nomor polisi. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.6 Pengujian Validasi Mengubah Rekanan Transportir

Dalam test-case mengubah rekanan transportir terdapat satu *test-case* yaitu *test-case* berhasil mengubah rekanan transportir yang dapat diperlihatkan didalam tabel 6.12.

A. *Test-case* Berhasil Mengubah Rekanan Transportir

Tabel 6.12 *Test-case* Berhasil Mengubah Rekanan Transportir

Nama *Test-case*	Berhasil Mengubah Rekanan Transportir
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan transportir. 3. Mengklik *button* edit pada data yang ingin diubah. 4. Mengisi data rekanan transportir yang ingin diubah yaitu nama perusahaan, username, password, konfirmasi password, nomor polisi. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan, dan data rekanan transportir telah terubah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan, dan data rekanan transportir telah terubah.
Status	Valid

6.3.7 Pengujian Validasi Melihat Batas Stok Kritis.

Dalam test-case melihat batas stok kritis terdapat satu *test-case* yaitu *test-case* berhasil melihat batas stok kritis yang dapat diperlihatkan didalam tabel 6.13.

A. *Test-case* Berhasil Melihat Batas Stok Kritis.

Tabel 6.13 *Test-case* Berhasil Melihat Batas Stok Kritis

Nama *Test-case*	Berhasil Melihat Batas Stok Kritis.
Prosedur	1. Telah login sebagai operator diswil 1 2. Mengklik menu batas stok kritis.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu batas stok kritis dan tabel batas stok kritis.
Hasil	Sistem memperlihatkan halaman menu batas stok kritis dan tabel batas stok kritis.
Status	Valid

6.3.8 Pengujian Validasi Menambah Batas Stok Kritis

Dalam test-case menambah batas stok kritis terdapat dua *test-case* yaitu *test-case* berhasil menambah stok kritis yang dapat diperlihatkan didalam tabel 6.14, dan *test-case* menambah stok kritis alternatif 1: jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.15.

A. *Test-case* Berhasil Menambah Stok Kritis.

Tabel 6.14 *Test-case* Berhasil Menambah Stok Kritis

Nama *Test-case*	Berhasil Menambah Stok Kritis
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu batas stok kritis. 3. Mengklik *button* tambah. 4. Mengisi data tambah batas stok kritis yaitu nama pupuk, daerah, dan qty (ton). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data batas stok kritis telah tertambah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data batas stok kritis telah tertambah.
Status	Valid

B. *Test-case* Menambah Stok Kritis Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.15 *Test-case* Menambah Stok Kritis Alternatif 1

Nama *Test-case*	Menambah Stok Kritis Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.

Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu batas stok kritis. 3. Mengklik *button* tambah. 4. Tidak mengisi data tambah batas stok kritis yaitu nama pupuk, daerah, dan qty (ton). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.9 Pengujian Validasi Mengubah Batas Stok Kritis.

Dalam test-case mengubah batas stok kritis terdapat satu *test-case* yaitu *test-case* berhasil mengubah batas stok kritis yang dapat diperlihatkan didalam tabel 6.16.

A. *Test-case* Berhasil Mengubah Batas Stok Kritis.

Tabel 6.16 *Test-case* Berhasil Mengubah Batas Stok Kritis

Nama *Test-case*	Berhasil Mengubah Batas Stok Kritis.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu batas stok kritis. 3. Mengklik *button* edit pada data yang ingin diubah. 4. Mengisi data batas stok kritis yang ingin diubah yaitu nama pupuk, daerah, dan qty (ton). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan, dan data batas stok kritis telah terubah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan, dan data batas stok kritis telah terubah.
Status	Valid

6.3.10 Pengujian Validasi Melihat Rekanan Pengelola Gudang Penyangga.

Dalam test-case melihat rekanan pengelola Gudang penyangga terdapat satu *test-case* yaitu *test-case* berhasil melihat rekanan pengelola Gudang penyangga yanf dapat dilihat pada tabel 6.17.

A. *Test-case* Berhasil Melihat Rekanan Pengelola Gudang Penyangga.

Tabel 6.17 *Test-case* Berhasil Melihat Rekanan Pengelola Gudang Penyangga

Nama *Test-case*	Berhasil Melihat Rekanan Pengelola Gudang Penyangga.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan g. penyangga.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu rekanan g. penyangga dan tabel rekanan gudang penyangga.
Hasil	Sistem memperlihatkan halaman menu rekanan g. penyangga dan tabel rekanan gudang penyangga.
Status	Valid

6.3.11 Pengujian Validasi Menambah Rekanan Pengelola Gudang Penyangga.

Dalam test-case menambah rekanan pengelola Gudang penyangga terdapat dua *test-case* yaitu *test-case* berhasil menambah rekanan pengelola Gudang penyangga yang dapat diperlihatkan didalam tabel 6.18 dan *test-case* menambah rekanan pengelola Gudang penyangga alternatif 1: jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.19.

A. *Test-case* Berhasil Menambah Rekanan Pengelola Gudang Penyangga.

Tabel 6.18 *Test-case* Berhasil Menambah Rekanan Pengelola Gudang Penyangga

Nama *Test-case*	Berhasil Menambah Rekanan Pengelola Gudang Penyangga.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan g. penyangga. 3. Mengklik *button* tambah 4. Memperlihatkan form tambah rekanan gudang. 5. Mengisi data tambah rekanan pengelola gudang penyangga yaitu nama perusahaan, alamat kantor, username, password dan konfirmasi password. 6. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan, dan data rekanan pengelola gudang penyangga telah tertambah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan, dan data rekanan pengelola gudang penyangga telah tertambah.
Status	Valid

B. *Test-case* Menambah Rekanan Pengelola Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.19 *Test-case* Menambah Rekanan Pengelola Gudang Penyangga Alternatif 1

Nama *Test-case*	Menambah Rekanan Pengelola Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan g. penyangga. 3. Mengklik *button* tambah. 4. Tidak mengisi data rekanan pengelola gudang penyangga yaitu nama perusahaan, alamat kantor, username, password dan konfirmasi password. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.12 Pengujian Validasi Mengubah Rekanan Pengelola Gudang Penyangga.

Dalam test-case mengubah rekanan pengelola Gudang penyangga terdapat satu *test-case* yaitu *test-case* berhasil mengubah rekanan pengelola gudang penyangga yang dapat diperlihatkan didalam tabel 6.20.

A. *Test-case* Berhasil Mengubah Rekanan Pengelola Gudang Penyangga.

Tabel 6.20 *Test-case* Berhasil Mengubah Rekanan Pengelola Gudang Penyangga

Nama *Test-case*	Berhasil Mengubah Rekanan Pengelola Gudang Penyangga.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu rekanan g. penyangga. 3. Mengklik *button* edit pada data yang ingin diubah. 4. Mengisi data pupuk yang ingin diubah yaitu nama perusahaan, alamat kantor, username, password, dan konfirmasi password.

	5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tesimpan dan data rekanan pengelola gudang penyangga telah terubah.
Hasil	Sistem memperlihatkan pesan data berhasil tesimpan dan data rekanan pengelola gudang penyangga telah terubah.
Status	Valid

6.3.13 Pengujian Validasi Melihat Gudang Penyangga.

Dalam test-case melihat gudang penyangga terdapat satu *test-case* yaitu *test-case* berhasil melihat gudang penyangga yang dapat diperlihatkan didalam tabel 6.21.

A. *Test-case* Berhasil Melihat Gudang Penyangga.

Tabel 6.21 *Test-case* Berhasil Melihat Gudang Penyangga

Nama *Test-case*	Berhasil Melihat Gudang Penyangga.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu daftar g. penyangga.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman daftar Data Gudang Penyangga dan tabel daftar gudang penyangga.
Hasil	Sistem memperlihatkan halaman daftar Data Gudang Penyangga dan tabel daftar gudang penyangga.
Status	Valid

6.3.14 Pengujian Validasi Menambah Gudang Penyangga.

Dalam test-case menambah gudang penyangga terdapat empat *test-case* yaitu *test-case* berhasil menambah gudang penyangga yang dapat diperlihatkan didalam tabel 6.22 dan *test-case* menambah gudang penyangga alternatif 1: jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.23.

A. *Test-case* Berhasil Menambah Gudang Penyangga.

Tabel 6.22 *Test-case* Berhasil Menambah Gudang Penyangga.

Nama *Test-case*	Berhasil Menambah Gudang Penyangga
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu daftar g. penyangga. 3. Mengklik *button* tambah

	4. Mengisi data tambah pupuk yaitu nama perusahaan, nama transportir, nama gudang, username, password, konfirmasi password, kepala gudang, daerah, daya tampung, kontak, alamat. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan, dan data gudang penyangga telah tertambah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan, dan data gudang penyangga telah tertambah.
Status	Valid

B. *Test-case* Menambah Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.23 *Test-case* Menambah Gudang Penyangga Alternatif 1

Nama *Test-case*	Menambah Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu daftar g. penyangga. 3. Mengklik *button* tambah. 4. Tidak mengisi data gudang penyangga yaitu nama perusahaan, nama transportir, nama gudang, username, password, konfirmasi password, kepala gudang, daerah, daya tampung, kontak, alamat. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.15 Pengujian Validasi Mengubah Gudang Penyangga.

Dalam test-case mengubah gudang penyangga terdapat satu *test-case* yaitu *test-case* berhasil mengubah gudang penyangga yang dapat diperlihatkan didalam tabel 6.24

A. *Test-case* Berhasil Mengubah Gudang Penyangga.

Tabel 6.24 *Test-case* Berhasil Mengubah Gudang Penyangga

Nama *Test-case*	Berhasil Mengubah Gudang Penyangga.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu daftar g. penyangga. 3. Mengklik *button* edit pada data yang ingin diubah. 4. Mengisi data pupuk yang ingin diubah yaitu nama perusahaan, nama transportir, nama gudang, username, password, konfirmasi password, kepala gudang, daerah, daya tampung, kontak, alamat. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan, dan data Gudang penyangga telah berhasil diubah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan, dan data Gudang penyangga telah berhasil diubah.
Status	Valid

6.3.16 Pengujian Validasi Melihat Petugas Stok Opname

Dalam test-case melihat petugas stok opname terdapat satu *test-case* yaitu *test-case test-case* berhasil melihat petugas stok opname yang dapat diperlihatkan didalam tabel 6.25.

A. *Test-case* Berhasil Melihat Petugas Stok Opname.

Tabel 6.25 *Test-case* Berhasil Melihat Petugas Stok Opname

Nama *Test-case*	Berhasil Melihat Petugas Stok Opname.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu petugas stok opname.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu petugas stok opname dan tabel petugas stok opname.
Hasil	Sistem memperlihatkan halaman menu petugas stok opname dan tabel petugas stok opname.
Status	Valid

6.3.17 Pengujian Validasi Menambah Petugas Stok Opname.

Dalam test-case menambah petugas stok opname terdapat dua *test-case* yaitu *test-case* berhasil menambah petugas stok opname yang dapat diliahat pada tabel 6.26 dan *test-case* menambah petugas stok opname alternatif 1: jika aktor

tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.27.

A. *Test-case* Berhasil Menambah Petugas Stok Opname.

Tabel 6.26 *Test-case* Berhasil Menambah Petugas Stok Opname

Nama *Test-case*	Berhasil Menambah Petugas Stok Opname
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu petugas stok opname. 3. Mengklik *button* tambah 4. Mengisi data tambah stok opname yaitu nama tim, username, password, konfirmasi password, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data petugas stok opname berhasil ditambahkan.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data petugas stok opname berhasil ditambahkan.
Status	Valid

B. *Test-case* Menambah Petugas Stok Opname Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.27 *Test-case* Menambah Petugas Stok Opname Alternatif 1

Nama *Test-case*	Menambah Petugas Stok Opname Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu petugas stok opname 3. Mengklik *button* tambah. 4. Tidak mengisi data petugas stok opname yaitu nama tim, username, password, konfirmasi password, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".

Status	Valid

6.3.18 Pengujian Validasi Mengubah Petugas Stok Opname.

Dalam test-case mengubah petugas stok opname terdapat satu *test-case* yaitu *test-case* berhasil mengubah petugas stok opname yang dapat diperlihatkan didalam tabel 6.28.

A. *Test-case* Berhasil Mengubah Petugas Stok Opname.

Tabel 6.28 *Test-case* Berhasil Mengubah Petugas Stok Opname

Nama *Test-case*	Berhasil Mengubah Petugas Stok Opname.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu petugas stok opname. 3. Mengklik *button* edit pada data yang ingin diubah. 4. Mengisi data petugas stok opname yang ingin diubah yaitu nama tim, username, password, konfirmasi password, nama petugas 1, nik petugas 1, nama petugas 2, nik petugas 2. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data petugas stok opname berhasil terubah.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data petugas stok opname berhasil terubah.
Status	Valid

6.3.19 Pengujian Validasi Membuat Penugasan Stok Opname.

Dalam test-case membuat penugasan stok opname terdapat dua *test-case* yaitu *test-case* berhasil membuat penugasan stok opname yang dapat diperlihatkan didalam tabel 6.29 dan *test-case* membuat penugasan stok opname alternatif 1: Jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.30.

A. *Test-case* Berhasil Membuat Penugasan Stok Opname.

Tabel 6.29 *Test-case* Berhasil Membuat Penugasan Stok Opname

Nama *Test-case*	Berhasil Membuat Penugasan Stok Opname.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu penugasan stok opname. 3. Mengklik *button* tambah.

	4. Mengisi data tambah pupuk yaitu nama tim, administrasi, deadline, daerah dan pilih gudang. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data penugasan stok opname berhasil dibuat.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data penugasan stok opname berhasil dibuat.
Status	Valid

B. *Test-case* Membuat Penugasan Stok Opname Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.30 *Test-case* Membuat Penugasan Stok Opname Alternatif 1

Nama *Test-case*	Membuat Penugasan Stok Opname Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu penugasan stok opname. 3. Mengklik *button* tambah. 4. Tidak mengisi data penugasan stok opname yaitu nama tim, administrasi, deadline, daerah dan pilih gudang. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.20 Pengujian Validasi Melihat Penugasan Stok Opname (Operator Diswil 1).

Dalam test-case melihat penugasan stok opname (operator diswil 1) terdapat satu *test-case* yaitu *test-case* berhasil melihat penugasan stok opname (operator diswil 1) yang dapat diperlihatkan didalam tabel 6.31.

A. *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Diswil 1).

Tabel 6.31 *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Diswil 1)

Nama *Test-case*	Berhasil Melihat Penugasan Stok Opname (Operator Diswil 1).
Prosedur	1. Telah login sebagai Operator Diswil 1. 2. Mengklik menu penugasan stok opname.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu penugasan stok opname.
Hasil	Sistem memperlihatkan halaman menu penugasan stok opname.
Status	Valid

6.3.21 Pengujian Validasi Membuat Permintaan Pengiriman Pupuk.

Dalam test-case membuat permintaan pengiriman pupuk terdapat dua *test-case* yaitu *Test-case* Berhasil Membuat Permintaan Pengiriman Pupuk yang dapat diperlihatkan didalam tabel 6.32 dan *Test-case* Membuat Permintaan Pengiriman Pupuk Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data yang dapat diperlihatkan didalam tabel 6.33.

A. *Test-case* Berhasil Membuat Permintaan Pengiriman Pupuk.

Tabel 6.32 *Test-case* Berhasil Membuat Permintaan Pengiriman Pupuk

Nama *Test-case*	Berhasil Membuat Permintaan Pengiriman Pupuk.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* tambah. 4. Mengisi data tambah pupuk yaitu jenis pupuk, deadline berangkat, tujuan, transportir, dan qty (ton) 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data permintaan pengiriman pupuk berhasil dibuat.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data permintaan pengiriman pupuk berhasil dibuat.
Status	Valid

B. *Test-case* Membuat Permintaan Pengiriman Pupuk Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.33 *Test-case* Membuat Permintaan Pengiriman Pupuk Alternatif 1

Nama *Test-case*	Membuat Permintaan Pengiriman Pupuk Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* tambah. 4. Tidak mengisi data permintaan pupuk yaitu jenis pupuk, deadline berangkat, tujuan, transportir, dan qty (ton). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Semua field harus disini".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

6.3.22 Pengujian Validasi Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1).

Dalam test-case melihat Permintaan Pengiriman Pupuk (Operator Diswil 1) terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1) yang dapat dilihat poada tabel 6.34.

A. *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1).

Tabel 6.34 *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1)

Nama *Test-case*	Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Diswil 1).
Prosedur	1. Telah login sebagai Operator Diswil 1. 2. Mengklik menu permintaan pupuk.
Hasil yang diharapkan	Sistem akan memperlihatkan daftar permintaan pengiriman pupuk di halaman menu permintaan pupuk.
Hasil	Sistem memperlihatkan daftar permintaan pengiriman pupuk di halaman menu permintaan pupuk.
Status	Valid

6.3.23 Pengujian Validasi Melihat Detail Permintaan Pengiriman Pupuk.

Dalam test-case melihat melihat detail permintaan pengiriman pupuk terdapat satu uji kasus *Test-case* Berhasil Melihat Detail Permintaan Pengiriman Pupuk yang dapat diperlihatkan didalam tabel 6.35.

A. *Test-case* Berhasil Melihat Detail Permintaan Pengiriman Pupuk.

Tabel 6.35 *Test-case* Berhasil Melihat Detail Permintaan Pengiriman Pupuk

Nama *Test-case*	Berhasil Melihat Detail Permintaan Pengiriman Pupuk
Prosedur	1. Telah login sebagai Operator Diswil 1. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* detail pada permintaan pupuk yang diinginkan.
Hasil yang diharapkan	Sistem akan memperlihatkan data detail permintaan pupuk yang dipilih pada form detail permintaan pupuk.
Hasil	Sistem memperlihatkan data detail permintaan pupuk yang dipilih pada form detail permintaan pupuk.
Status	Valid

6.3.24 Pengujian Validasi Melihat Status Pengiriman Pupuk.

Dalam test-case melihat status pengiriman pupuk terdapat satu *test-case* yaitu *test-case Test-case* Berhasil Melihat Status Pengiriman Pupuk yang dapat diperlihatkan didalam tabel 6.36.

A. *Test-case* Berhasil Melihat Status Pengiriman Pupuk.

Tabel 6.36 *Test-case* Berhasil Melihat Status Pengiriman Pupuk

Nama *Test-case*	Berhasil Melihat Status Pengiriman Pupuk.
Prosedur	1. Telah login sebagai Operator Rekanan Transportir. 2. Mengklik menu permintaan pupuk.
Hasil yang diharapkan	Sistem akan memperlihatkan tabel status permintaan pupuk pada menu permintaan pupuk.
Hasil	Sistem memperlihatkan tabel status permintaan pupuk pada menu permintaan pupuk.
Status	Valid

6.3.25 Pengujian Validasi Menanggapi Permintaan Pengiriman Pupuk.

Dalam test-case menanggapi permintaan pengiriman pupuk terdapat empat *test-case* yaitu *Test-case* Berhasil Menanggapi Permintaan Pengiriman Pupuk yang dapat diperlihatkan didalam tabel 6.37, *Test-case* Menanggapi

Permintaan Pengiriman Pupuk Alternatif 1: Jika total *quantity* muat kendaraan yang ditugaskan tidak sama dengan *quantity* permintaan yang dapat diperlihatkan didalam tabel 6.38, *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 2: Jika plat nomor yang dipilih masih dalam proses pengiriman pupuk yang dapat diperlihatkan didalam tabel 6.39, dan *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 3: Jika plat nomor yang dipilih sama dengan yang dipilih sebelumnya yang dapat diperlihatkan didalam tabel 6.40.

A. *Test-case* Berhasil Menanggapi Permintaan Pengiriman Pupuk.

Tabel 6.37 *Test-case* Berhasil Menanggapi Permintaan Pengiriman Pupuk

Nama *Test-case*	Berhasil Menanggapi Permintaan Pengiriman Pupuk
Prosedur	1. Telah login sebagai Operator Rekanan Transportir. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* antar pada tabel permintaan pupuk yang ingin diproses. 4. Mengisi data yang diperlukan untuk mengisi detail permintaan pupuk yaitu nama supir, nomor polisi, qty. 5. Mengklik *button* antar.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan permintaan pupuk telah berhasil ditanggapi.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan permintaan pupuk telah berhasil ditanggapi.
Status	Valid

B. *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 1: Jika total *quantity* muat kendaraan yang ditugaskan tidak sama dengan *quantity* permintaan.

Tabel 6.38 *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 1

Nama *Test-case*	Menanggapi Permintaan Pengiriman Pupuk Alternatif 1: Jika total *quantity* muat kendaraan yang ditugaskan tidak sama dengan *quantity* permintaan.
Prosedur	1. Telah login sebagai Operator Rekanan Transportir. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* antar. 4. Mengisi qty kendaraan yang ditugaskan tidak sama dengan qty permintaan.

	5. Mengklik *button* antar.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan "Gagal, Qty tidak sesuai".
Hasil	Sistem memperlihatkan pesan "Semua field harus disini".
Status	Valid

C. *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 2: Jika plat nomor yang dipilih masih dalam proses pengiriman pupuk.

Tabel 6.39 *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 2

Nama *Test-case*	Menanggapi Permintaan Pengiriman Pupuk Alternatif 2 Jika plat nomor yang dipilih masih dalam proses pengiriman pupuk.
Prosedur	1. Telah login sebagai Operator Rekanan Transportir. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* antar. 4. Tidak mengisi data yang diperlukan untuk mananggapi permintaan pupuk yaitu nama supir, qty. 5. Mengklik plat nomor yang dipilih masih dalam proses pengiriman pupuk. 6. Mengklik *button* antar.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan kendaraan masih mengantar pupuk.
Hasil	Sistem memperlihatkan pesan kendaraan masih mengantar pupuk.
Status	Valid

D. *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 3: Jika plat nomor yang dipilih sama dengan yang dipilih sebelumnya.

Tabel 6.40 *Test-case* Menanggapi Permintaan Pengiriman Pupuk Alternatif 3

Nama *Test-case*	Menanggapi Permintaan Pengiriman Pupuk Alternatif 3: Jika plat nomor yang dipilih sama dengan yang dipilih sebelumnya.
Prosedur	1. Telah login sebagai Operator Rekanan Transportir. 2. Mengklik menu permintaan pupuk.

	3. Mengklik *button* antar. 4. Tidak mengisi data yang diperlukan untuk mananggapi permintaan pupuk yaitu nama supir, qty. 5. Mengklik plat nomor yang dipilih sama dengan yang dipilih sebelumnya. 6. Mengklik *button* antar.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan kendaraan yang dipilih tidak boleh sama.
Hasil	Sistem memperlihatkan pesan kendaraan yang dipilih tidak boleh sama.
Status	Valid

6.3.26 Pengujian Validasi Melihat Selisih Stok Opname di Gudang Penyangga.

Dalam test-case Melihat Selisih Stok Opname di Gudang Penyangga terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Selisih Stok Opname di Gudang Penyangga yang dapat diperlihatkan didalam tabel 6.41.

A. *Test-case* Berhasil Melihat Selisih Stok Opname di Gudang Penyangga.

Tabel 6.41 *Test-case* Berhasil Melihat Selisih Stok Opname di Gudang Penyangga

Nama *Test-case*	Berhasil Melihat Selisih Stok Opname di Gudang Penyangga.
Prosedur	1. Telah login sebagai Operator Rekanan Gudang Penyangga. 2. Mengklik menu pupuk hilang.
Hasil yang diharapkan	Sistem akan memperlihatkan data pupuk yang hilang digudang penyangga pada halaman pupuk hilang.
Hasil	Sistem memperlihatkan data pupuk yang hilang digudang penyangga pada halaman pupuk hilang.
Status	Valid

6.3.27 Pengujian Validasi Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga).

Dalam test-case melihat hasil stok opname yang sudah divalidasi (Operator Rekanan Gudang Penyangga) terdapat satu *test-case* yaitu *test-case Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga) yang dapat diperlihatkan didalam tabel 6.42.

A. *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga).

Tabel 6.42 *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Rekanan Gudang Penyangga).
Prosedur	1. Telah login sebagai Operator Rekanan Gudang Penyangga. 2. Mengklik menu stok opname. 3. Mengklik *button* hasil pada tabel daftar tugas stok opname yang dikerjakan
Hasil yang diharapkan	Sistem akan memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Hasil	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Status	Valid

6.3.28 Pengujian Validasi Membuat Perhitungan Stok Opname.

Dalam test-case membuat perhitungan stok opname terdapat tiga *test-case* yaitu *test-case* Berhasil Membuat Perhitungan Stok Opname yang dapat diperlihatkan didalam tabel 6.43, *Test-case* Membuat Perhitungan Stok Opname Alternatif 1: Jika aktor tidak menambah perhitungan pupuk sama sekali yang dapat diperlihatkan didalam tabel 6.44, dan *Test-case* Membuat Perhitungan Stok Opname Alternatif 2: Jika aktor tidak menghitung semua pupuk yang ada di gudang yang dapat diperlihatkan didalam tabel 6.45.

A. *Test-case* Berhasil Membuat Perhitungan Stok Opname.

Tabel 6.43 *Test-case* Berhasil Membuat Perhitungan Stok Opname

Nama *Test-case*	Berhasil Membuat Perhitunga Stok Opname.
Prosedur	1. Telah login sebagai Operator Stok Opname Diswil 1. 2. Mengklik menu stok opname. 3. Mengklik *button* hitung pada penugasan stok opname yang ingin di ekseskusi. 4. Mengklik pupuk yang dihitung. 5. Mengklik *button* tambah. 6. Mengklik *button* proses pada pupuk yang ingin dihitung.

	7. Mengklik *button* tambah untuk menambah perhitungan pupuk. 8. Mengisi data yang dibutuhkan untuk menghitung stok opname yaitu area, kunci, panjang, lebar, tinggi, lebihan bag. 9. Mengklik *button* simpan. 10. Mengklik *button* selesai 11. Mengklik *button* ok.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan berhasil, dan data perhitungan stok opname berhasil dibuat.
Hasil	Sistem memperlihatkan pesan berhasil, dan data perhitungan stok opname berhasil dibuat.
Status	Valid

B. *Test-case* Membuat Perhitungan Stok Opname Alternatif 1: Jika aktor tidak menambah perhitungan pupuk sama sekali.

Tabel 6.44 *Test-case* Membuat Perhitungan Stok Opname Alternatif 1

Nama *Test-case*	Membuat Perhitungan Stok Opname Alternatif 1: Jika aktor tidak menambah perhitungan pupuk sama sekali.
Prosedur	1. Telah login sebagai Operator Stok Opname Diswil 1 2. Mengklik menu stok opname. 3. Mengklik *button* hitung pada penugasan stok opname yang ingin di ekseskusi. 4. Mengklik pupuk yang dihitung. 5. Mengklik *button* tambah. 6. Mengklik *button* proses pada pupuk yang ingin dihitung. 7. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memunculkan pemberitahuan "data isian tidak bisa kosong".
Hasil	Sistem memunculkan pemberitahuan "data isian tidak bisa kosong".
Status	Valid

C. *Test-case* Membuat Perhitungan Stok Opname Alternatif 2: Jika aktor tidak menghitung semua pupuk yang ada di gudang.

Tabel 6.45 *Test-case* Membuat Perhitungan Stok Opname Alternatif 2

Nama *Test-case*	Membuat Perhitungan Stok Opname Alternatif 2: Jika aktor tidak menghitung semua pupuk yang ada di gudang.
Prosedur	1. Telah login sebagai Operator Stok Opname Diswil 1 2. Mengklik menu stok opname. 3. Mengklik *button* hitung pada penugasan stok opname yang ingin di ekseskusi. 4. Mengklik *button* selesai 5. Mengklik *button* ok pada modal konfirmasi.
Hasil yang diharapkan	Sistem akan memunculkan pemberitahuan "Data Gagal Tersimpan, Ada Pupuk Yang Belum Dihitung".
Hasil	Sistem memunculkan pemberitahuan "Data Gagal Tersimpan, Ada Pupuk Yang Belum Dihitung".
Status	Valid

6.3.29 Pengujian Validasi Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1).

Dalam test-case melihat melihat hasil stok opname (Operator Stok Opname Diswil 1) terdapat satu *test-case* yaitu *test-case* berhasil melihat hasil stok opname (Operator Stok Opname Diswil 1) yang dapat diperlihatkan didalam tabel 6.46.

A. *Test-case* Berhasil Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1).

Tabel 6.46 *Test-case* Berhasil Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1)

Nama *Test-case*	Berhasil Melihat Hasil Stok Opname (Operator Stok Opname Diswil 1)
Prosedur	1. Telah login sebagai Operator Stok Opname Diswil 1 2. Mengklik menu stok opname. 3. Mengklik *button* hasil pada stok opname yang diinginkan.
Hasil yang diharapkan	Sistem akan memperlihatkan modal hasil stok opname.
Hasil	Sistem memperlihatkan modal hasil stok opname.
Status	Valid

6.3.30 Pengujian Validasi Melihat Peta Kondisi Stok Daerah.

Dalam test-case melihat peta kondisi stok daerah terdapat satu *test-case* *test-case* berhasil melihat peta kondisi stok daerah yang dapat diperlihatkan didalam tabel 6.47.

A. *Test-case* Berhasil Melihat Peta Kondisi Stok Daerah.

Tabel 6.47 *Test-case* Berhasil Melihat Peta Kondisi Stok Daerah

Nama *Test-case*	Berhasil Melihat Peta Kondisi Stok Daerah.
Prosedur	1. Telah login sebagai Operator Diswil 1, Manajer Diswil 2. Mengklik menu kondisi stok daerah. 3. Mengklik pada peta daerah yang diinginkan.
Hasil yang diharapkan	Sistem akan memperlihatkan data total stok berbagai pupuk dan detail pergudang dari kabupaten/kota yang dipilih.
Hasil	Sistem memperlihatkan data total stok berbagai pupuk dan detail pergudang dari kabupaten/kota yang dipilih.
Status	Valid

6.3.31 Pengujian Validasi Membuat Laporan Pupuk Rusak.

Pada kasus pengujian Membuat Laporan Pupuk Rusak terdapat dua *test-case* yaitu *Test-case* Berhasil Membuat Laporan Pupuk Rusak yang ditunjukkan pada Tabel 6.48 dan *Test-case* Membuat Laporan Pupuk Rusak Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data yang ditunjukan pada tabel 6.49.

A. *Test-case* Berhasil Membuat Laporan Pupuk Rusak.

Tabel 6.48 *Test-case* Berhasil Membuat Laporan Pupuk Rusak

Nama *Test-case*	Berhasil Membuat Laporan Pupuk Rusak
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu stok pupuk. 3. Mengklik *button* pupuk rusak pada jenis pupuk yang ingin dilaporkan. 3. Menambahkan data laporan pupuk rusak yaitu quantity rusak, foto bukti, dan alasan rusak. 4. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data laporan pupuk rusak berhasil dibuat.
Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data laporan pupuk rusak berhasil dibuat.

Status	Valid

B. *Test-case* Membuat Laporan Pupuk Rusak Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.49 *Test-case* Membuat Laporan Pupuk Rusak Alternatif 1

Nama *Test-case*	Membuat Laporan Pupuk Rusak Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu stok pupuk 3. Mengklik *button* pupuk rusak pada jenis pupuk yang ingin dilaporkan. 4. Tidak mengisi data pupuk rusak yaitu qty rusak (ton), foto bukti, dan alasan rusak. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pemberitahuan untuk mengisi semua form.
Hasil	Sistem memperlihatkan pemberitahuan untuk mengisi semua form.
Status	Valid

6.3.32 Pengujian Validasi Melihat Laporan Pupuk Rusak (Operator Diswil 1).

Pada kasus pengujian Melihat Laporan Pupuk Rusak (Operator Diswil 1) terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Diswil 1) yang ditunjukkan pada Tabel 6.50.

A. *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Diswil 1).

Tabel 6.50 *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Diswil 1)

Nama *Test-case*	Berhasil Melihat Laporan Pupuk Rusak (Operator Diswil 1).
Prosedur	1. Telah login sebagai Operator Diswil 1 2. Mengklik menu pupuk rusak.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.

Hasil	Sistem memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.
Status	Valid

6.3.33 Pengujian Validasi Menerima Pupuk Masuk Gudang Penyangga

Pada kasus pengujian Menerima Pupuk Masuk Gudang Penyangga terdapat dua *test-case* yaitu *Test-case* Berhasil Menerima Pupuk Masuk Gudang Penyangga yang ditunjukkan pada Tabel 6.51 dan *Test-case* Menerima Pupuk Masuk Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data yang ditunjukan pada tabel 6.52.

A. *Test-case* Berhasil Menerima Pupuk Masuk Gudang Penyangga.

Tabel 6.51 *Test-case* Berhasil Menerima Pupuk Masuk Gudang Penyangga

Nama *Test-case*	Berhasil Menerima Pupuk Masuk Gudang Penyangga
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu pupuk masuk. 3. Mengklik *button* terima pada pupuk yang diterima. 4. Mengisi data pupuk masuk yang diperlukan yaitu qty diterima (ton) 5. Mengklik *button* simpan. 6. Mengklik *button* ya pada modal konfirmasi.
Hasil yang diharapkan	Sistem akan memperlihatkan data berhasil tesimpan dan status pengiriman pupuk yang diinginkan berubah menjadi selesai.
Hasil	Sistem memperlihatkan data berhasil tesimpan dan status pengiriman pupuk yang diinginkan berubah menjadi selesai.
Status	Valid

B. *Test-case* Menerima Pupuk Masuk Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.52 *Test-case* Menerima Pupuk Masuk Gudang Penyangga Alternatif 1

Nama *Test-case*	Menerima Pupuk Masuk Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu pupuk masuk.

	3. Mengklik *button* terima pada pupuk yang diterima. 4. Tidak mengisi data pupuk masuk yaitu qty diterima (ton). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pemberitahuan untuk mengisi semua form.
Hasil	Sistem memperlihatkan pemberitahuan untuk mengisi semua form.
Status	Valid

6.3.34 Pengujian Validasi Menambah Pupuk Keluar Gudang Penyangga.

Pada kasus pengujian menambah pupuk keluar gudang penyangga terdapat dua *test-case* yaitu *Test-case* Berhasil menambah Pupuk Keluar Gudang Penyangga yang ditunjukkan pada Tabel 6.53 dan *Test-case* Menambah Pupuk Keluar Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data yang ditunjukan pada tabel 6.54.

A. *Test-case* Berhasil Menambah Pupuk Keluar Gudang Penyangga.

Tabel 6.53 *Test-case* Berhasil Menambah Pupuk Keluar Gudang Penyangga

Nama *Test-case*	Berhasil Menambah Pupuk Keluar Gudang Penyangga
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu stok pupuk. 3. Mengklik *button* pupuk keluar pada pupuk yang diinginkan. 3. Mengisi data yang diperlukan untuk menambah pupuk keluar yaitu distributor, nomor polisi, qty keluar (ton). 4. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan data berhasil tersimpan dan data pupuk keluar berhasil ditambahkan.
Hasil	Sistem memperlihatkan data berhasil tersimpan dan data pupuk keluar berhasil ditambahkan.
Status	Valid

B. *Test-case* Menambah Pupuk Keluar Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data

Tabel 6.54 *Test-case* Menambah Pupuk Keluar Gudang Penyangga Alternatif 1

Nama *Test-case*	Menambah Pupuk Keluar Gudang Penyangga Alternatif 1: Jika aktor tidak mengisi salah satu atau semua data.
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu stok pupuk 3. Mengklik *button* pupuk keluar pada pupuk yang diinginkan. 4. Tidak mengisi data pupuk keluar yaitu distributor, nomor polisi, qty keluar (ton). 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pemberitahuan untuk mengisi semua form.
Hasil	Sistem memperlihatkan pemberitahuan untuk mengisi semua form.
Status	Valid

6.3.35 Pengujian Validasi Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga)

Pada kasus pengujian melihat pupuk keluar gudang penyangga terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (operator gudang penyangga) yang ditunjukkan pada Tabel 6.55.

A. *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga).

Tabel 6.55 *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Gudang Penyangga)
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu pupuk keluar.
Hasil yang diharapkan	Sistem akan memperlihatkan menu pupuk keluar dan memperlihatkan data pupuk keluar pada tabel pupuk keluar.
Hasil	Sistem memperlihatkan menu pupuk keluar dan memperlihatkan data pupuk keluar pada tabel pupuk keluar.
Status	Valid

6.3.36 Pengujian Validasi Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga)

Pada kasus pengujian melihat pupuk masuk gudang penyangga terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (operator gudang penyangga) yang ditunjukkan pada Tabel 6.56.

A. *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga).

Tabel 6.56 *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Gudang Penyangga)
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu pupuk masuk penyangga.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk masuk dan memperlihatkan data pupuk masuk pada tabel pupuk masuk.
Hasil	Sistem memperlihatkan halaman menu pupuk masuk dan memperlihatkan data pupuk masuk pada tabel pupuk masuk.
Status	Valid

6.3.37 Pengujian Validasi Validasi Hasil Stok Opname.

Dalam test-case Berhasil Validasi Hasil Stok Opname terdapat dua *test-case* yaitu *Test-case* Berhasil Validasi Hasil Stok Opname yang ditunjukkan pada Tabel 6.57 dan *Test-case* Validasi Hasil Stok Opname Alternatif 1: Jika aktor memilih *button* proses kembali yang ditunjukkan pada Tabel 6.58.

A. *Test-case* Berhasil Validasi Hasil Stok Opname.

Tabel 6.57 *Test-case* Berhasil Validasi Hasil Stok Opname

Nama *Test-case*	Berhasil Validasi Hasil Stok Opname.
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu hasil stok opname. 3. Mengklik *button* validasi pada hasil stok opname yang ingin divalidasi. 3. Mengklik *button* ya pada modal konfirmasi.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan status stok opname yang diinginkan berubah menjadi tervalidasi.

Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan status stok opname yang diinginkan berubah menjadi tervalidasi.
Status	Valid

B. *Test-case* Validasi Hasil Stok Opname Alternatif 1: Jika aktor memilih *button* proses kembali

Tabel 6.58 *Test-case* Validasi Hasil Stok Opname Alternatif 1

Nama *Test-case*	Validasi Hasil Stok Opname Alternatif 1: Jika aktor memilih *button* proses kembali.
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu hasil stok opname. 3. Mengklik *button* proses kembali pada hasil stok opname yang ingin diproses kembali. 3. Mengklik *button* ya pada modal konfirmasi.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan status stok opname berubah menjadi sedang dikerjakan.
Hasil	Sistem memperlihatkan pesan gagal.
Status	Valid

6.3.38 Pengujian Validasi Memproses Pemberian Pupuk

Dalam test-case Memproses Pemberian Pupuk terdapat satu *test-case* yaitu *Test-case* Berhasil Memproses Pemberian Pupuk yang ditunjukkan pada Tabel 6.59.

A. *Test-case* Berhasil Memproses Pemberian Pupuk.

Tabel 6.59 *Test-case* Berhasil Memproses Pemberian Pupuk

Nama *Test-case*	Berhasil Memproses Pemberian Pupuk
Prosedur	1. Telah login sebagai Operator Gudang Pusat. 2. Mengklik menu pemberian pupuk. 3. Mengklik *button* proses pada permintaan pupuk yang diinginkan. 4. Mengklik *button* simpan
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan status permintaan pengiriman pupuk yang diproses berubah menjadi pusat --> gudang.

Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan status permintaan pengiriman pupuk yang diproses berubah menjadi pusat --> gudang.
Status	Valid

6.3.39 Pengujian Validasi Melihat Pupuk Hilang Saat Pengiriman.

Dalam test-case Melihat Pupuk Hilang Saat Pengiriman terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Pupuk Hilang Saat Pengiriman yang ditunjukkan pada Tabel 6.60.

A. *Test-case* Berhasil Melihat Pupuk Hilang Saat Pengiriman.

Tabel 6.60 *Test-case* Berhasil Melihat Pupuk Hilang Saat Pengiriman

Nama *Test-case*	Berhasil Melihat Pupuk Hilang Saat Pengiriman.
Prosedur	1. Telah login sebagai operator rekanan transportir. 2. Mengklik menu pupuk hilang.
Hasil yang diharapkan	Sistem akan memperlihatkan data dan halaman pupuk hilang.
Hasil	Sistem memperlihatkan data dan halaman pupuk hilang.
Status	Valid

6.3.40 Pengujian Validasi Menerima Pengembalian Pupuk Rusak.

Dalam test-case menerima pengembalian pupuk rusak terdapat satu *test-case* yaitu *Test-case* Berhasil Menerima Pengembalian Pupuk Rusak yang ditunjukkan pada Tabel 6.61.

A. *Test-case* Berhasil Menerima Pengembalian Pupuk Rusak.

Tabel 6.61 *Test-case* Berhasil Menerima Pengembalian Pupuk Rusak

Nama *Test-case*	Berhasil Menerima Pengembalian Pupuk Rusak.
Prosedur	1. Telah login sebagai Operator Gudang Pusat. 2. Mengklik menu pupuk rusak. 3. Mengklik *button* proses pada pupuk rusak yang diterima. 4. Mengklik *button* terima.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan berhasil, dan data pupuk rusak telah terverifikasi diterima gudang pusat.
Hasil	Sistem memperlihatkan pesan berhasil, dan data pupuk rusak telah terverifikasi diterima gudang pusat.
Status	Valid

6.3.41 Pengujian Validasi Mencetak Surat Jalan

Dalam test-case mencetak surat jalan terdapat 1 *test-case* yaitu *Test-case* Berhasil Mencetak Surat Jalan yang ditunjukkan pada Tabel 6.62.

A. *Test-case* Berhasil Mencetak Surat Jalan.

Tabel 6.62 *Test-case* Berhasil Mencetak Surat Jalan

Nama *Test-case*	Berhasil Mencetak Surat Jalan.
Prosedur	1. Telah login sebagai operator rekanan transportir. 2. Mengklik menu permintaan pupuk. 3. Mengklik *button* surat jalan pada tabel status permintaan pupuk.
Hasil yang diharapkan	Sistem akan memperlihatkan surat jalan pengiriman yang dipilih.
Hasil	Sistem memperlihatkan surat jalan pengiriman yang dipilih.
Status	Valid

6.3.42 Pengujian Validasi Validasi Pupuk Rusak

Dalam test-case validasi pupuk rusak terdapat dua *test-case* yaitu *Test-case* Berhasil Validasi Pupuk Rusak yang ditunjukkan pada Tabel 6.63 dan *Test-case* Validasi Hasil Stok Opname Alternatif 1: Jika aktor memilih *button* review ulang yang ditunjukkan pada Tabel 6.64.

A. *Test-case* Berhasil Validasi Pupuk Rusak

Tabel 6.63 *Test-case* Berhasil Validasi Pupuk Rusak

Nama *Test-case*	Berhasil Validasi Pupuk Rusak.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu pupuk rusak. 3. Mengklik *button* ubah status pada laporan pupuk rusak yang diinginkan 4. Mengubah status menjadi setujui pada modal ubah status. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan berhasil, dan status laporan pupuk rusak telah berubah valid.
Hasil	Sistem memperlihatkan pesan berhasil, dan status laporan pupuk rusak telah berubah valid.

Status	Valid

B. *Test-case* Validasi Hasil Stok Opname Alternatif 1: Jika aktor memilih *button* review ulang

Tabel 6.64 *Test-case* Validasi Hasil Stok Opname Alternatif 1

Nama *Test-case*	Validasi Hasil Stok Opname Alternatif 1: Jika aktor memilih *button* review ulang.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu pupuk rusak. 3. Mengklik *button* ubah status pada laporan pupuk rusak yang diinginkan 4. Mengubah status menjadi review ulang. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan mengubah status laporan pupuk rusak menjadi review ulang
Hasil	Sistem mengubah status laporan pupuk rusak menjadi review ulang
Status	Valid

6.3.43 Pengujian Validasi Mengirim Pupuk Rusak

Dalam test-case Mengirim Pupuk Rusak terdapat satu *test-case* yaitu *Test-case* Berhasil Mengirim Pupuk Rusak yang ditunjukkan pada Tabel 6.65.

A. *Test-case* Berhasil Mengirim Pupuk Rusak

Tabel 6.65 *Test-case* Berhasil Mengirim Pupuk Rusak

Nama *Test-case*	Berhasil Mengirim Pupuk Rusak
Prosedur	1. Telah login sebagai Operator Gudang Penyangga. 2. Mengklik menu pupuk rusak. 3. Mengklik *button* kirim pupuk rusak. 4. Melengkapi data yang diperlukan untuk mengirim pupuk rusak ke gudang pusat yaitu nama supir, plat nomor, pupuk rusak yang dikirim. 5. Mengklik *button* simpan.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan data berhasil tersimpan dan data pengiriman pupuk rusak berhasil tertambah.

Hasil	Sistem memperlihatkan pesan data berhasil tersimpan dan data pengiriman pupuk rusak berhasil tertambah.
Status	Valid

6.3.44 Pengujian Validasi Login

Dalam test-case login terdapat 3 *test-case* yaitu *Test-case* Berhasil Login yang ditunjukkan pada Tabel 6.66, *Test-case* Login Alternatif 1: Jika data username atau password tidak valid yang ditunjukan pada Tabel 6.67, dan *Test-case* Login Alternatif 2: Jika data username atau password tidak valid yang ditunjukan pada Tabel 6.68.

A. *Test-case* Berhasil Login

Tabel 6.66 *Test-case* Berhasil Login

Nama *Test-case*	Berhasil Login
Prosedur	1. Berada pada halaman login. 2. Memasukan username dan password yang sesuai. 3. Mengklik *button* login.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman *home* sesuai dengan aktor yang berhasil dikenali.
Hasil	Sistem memperlihatkan halaman *home* sesuai dengan aktor yang berhasil dikenali.
Status	Valid

B. *Test-case* Login Alternatif 1: Jika data username atau password tidak valid.

Tabel 6.67 *Test-case* Login Alternatif 1

Nama *Test-case*	Login Alternatif 1: Jika data username atau password tidak valid.
Prosedur	1. Berada pada halaman login. 2. Memasukan username dan password yang tidak sesuai. 3. Mengklik *button* login pada form.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan error bahwa login gagal dan memperlihatkan halaman login.
Hasil	Sistem memperlihatkan pesan error bahwa login gagal dan memperlihatkan halaman login.
Status	Valid

C. *Test-case* Login Alternatif 2: Jika data username atau password tidak valid.

Tabel 6.68 *Test-case* Login Alternatif 2

Nama *Test-case*	Login Alternatif 2: Jika data username atau password tidak valid.
Prosedur	1. berada pada halaman login. 2. mengisi data username dan password yang tidak valid. 3. Mengklik *button* login.
Hasil yang diharapkan	Sistem akan memperlihatkan pesan gagal login dan memperlihatkan halaman login.
Hasil	Sistem memperlihatkan pesan gagal login dan memperlihatkan halaman login.
Status	Valid

6.3.45 Pengujian Validasi Logout

Dalam test-case logout terdapat 1 *test-case* yaitu, *Test-case* Berhasil logout yang ditunjukkan pada Tabel 6.69.

A. *Test-case* Berhasil Logout

Tabel 6.69 *Test-case* Berhasil Logout

Nama *Test-case*	Berhasil Logout
Prosedur	1. Telah login. 2. Mengklik menu logout
Hasil yang diharapkan	Sistem akan menghapus hak akses aktor dan memperlihatkan halaman login.
Hasil	Sistem menghapus hak akses aktor memperlihatkan halaman login.
Status	Valid

6.3.46 Pengujian Validasi Melihat Daftar Pemberian Pupuk

Dalam test-case Melihat Daftar Pemberian Pupuk terdapat satu *test-case* yaitu, *Test-case* Berhasil Melihat Daftar Pemberian Pupuk yang ditunjukkan pada Tabel 6.70.

A. *Test-case* Berhasil Melihat Daftar Pemberian Pupuk.

Tabel 6.70 *Test-case* Berhasil Melihat Daftar Pemberian Pupuk

Nama *Test-case*	Berhasil Melihat Daftar Pemberian Pupuk
Prosedur	1. Telah login sebagai Operator Gudang Pusat.

	2. Mengklik menu pemberian pupuk.
Hasil yang diharapkan	Sistem akan berhasil memperlihatkan data dan halaman pemberian pupuk.
Hasil	Sistem berhasil memperlihatkan data dan halaman pemberian pupuk.
Status	Valid

6.3.47 Pengujian Validasi Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat)

Dalam test-case Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat) terdapat satu *test-case* yaitu, *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat) yang ditunjukkan pada Tabel 6.71.

A. *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat).

Tabel 6.71 *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat)

Nama *Test-case*	Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Pusat)
Prosedur	1. Telah login sebagai Operator Gudang Pusat 2. Mengklik menu pupuk rusak.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk rusak dan tabel pengiriman pupuk rusak.
Hasil	Sistem memperlihatkan halaman menu pupuk rusak dan tabel pengiriman pupuk rusak.
Status	Valid

6.3.48 Pengujian Validasi Melihat Gudang Penyangga yang Dikelola.

Dalam test-case melihat gudang penyangga yang dikelola terdapat satu *test-case* yaitu *test-case* berhasil melihat gudang penyangga yang dikelola dapat dilihat pada tabel 6.72.

A. *Test-case* Berhasil Melihat Gudang Penyangga yang Dikelola.

Tabel 6.72 *Test-case* Berhasil Melihat Gudang Penyangga yang Dikelola

Nama *Test-case*	Berhasil Melihat Gudang Penyangga.
Prosedur	1. Telah login sebagai operator diswil 1. 2. Mengklik menu daftar g. penyangga.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman daftar Data Gudang Penyangga dan tabel daftar gudang penyangga.

Hasil	Sistem memperlihatkan halaman daftar Data Gudang Penyangga dan tabel daftar gudang penyangga.
Status	Valid

6.3.49 Pengujian Validasi Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga).

Dalam test-case melihat penugasan stok opname (operator rekanan gudang penyangga) terdapat satu *test-case* yaitu *test-case* berhasil melihat penugasan stok opname (operator rekanan gudang penyangga) yang dapat diperlihatkan didalam tabel 6.73.

A. *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga).

Tabel 6.73 *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Penugasan Stok Opname (Operator Rekanan Gudang Penyangga).
Prosedur	1. Telah login sebagai Operator Rekanan Gudang Penyangga. 2. Mengklik menu stok opname.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu penugasan stok opname.
Hasil	Sistem memperlihatkan halaman menu penugasan stok opname.
Status	Valid

6.3.50 Pengujian Validasi Melihat Penugasan Stok Opname (Manajer Diswil 1).

Dalam test-case melihat penugasan stok opname (manajer diswil 1) terdapat satu *test-case* yaitu *test-case* berhasil melihat penugasan stok opname (manajer diswil 1) yang dapat diperlihatkan didalam tabel 6.74.

A. *Test-case* Berhasil Melihat Penugasan Stok Opname (Manajer Diswil 1).

Tabel 6.74 *Test-case* Berhasil Melihat Penugasan Stok Opname (Manajer Diswil 1)

Nama *Test-case*	Berhasil Melihat Penugasan Stok Opname (Manajer Diswil 1).
Prosedur	1. Telah login sebagai Manajer Diswil 1.

	2. Mengklik menu stok opname.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu penugasan stok opname.
Hasil	Sistem memperlihatkan halaman menu penugasan stok opname.
Status	Valid

6.3.51 Pengujian Validasi Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1).

Dalam test-case melihat penugasan stok opname (operator stok opname diswil 1) terdapat satu *test-case* yaitu *test-case* berhasil melihat penugasan stok opname (operator stok opname diswil 1) yang dapat diperlihatkan didalam tabel 6.75.

A. *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1).

Tabel 6.75 *Test-case* Berhasil Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1)

Nama *Test-case*	Berhasil Melihat Penugasan Stok Opname (Operator Stok Opname Diswil 1).
Prosedur	1. Telah login sebagai Operator Stok Opname Diswil 1. 2. Mengklik menu penugasan stok opname.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu penugasan stok opname.
Hasil	Sistem memperlihatkan halaman menu penugasan stok opname.
Status	Valid

6.3.52 Pengujian Validasi Melihat Riwayat Penugasan Stok Opname

Dalam test-case melihat riwayat penugasan stok opname terdapat satu *test-case* yaitu *test-case* berhasil melihat riwayat penugasan stok opname yang dapat diperlihatkan didalam tabel 6.76.

A. *Test-case* Berhasil Melihat Riwayat Penugasan Stok Opname.

Tabel 6.76 *Test-case* Berhasil Melihat Riwayat Penugasan Stok Opname

Nama *Test-case*	Berhasil Melihat Riwayat Penugasan Stok Opname.
Prosedur	1. Telah login sebagai Operator Stok Opname Diswil 1.

	2. Mengklik menu riwayat stok opname.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu riwayat stok opname.
Hasil	Sistem memperlihatkan halaman menu riwayat stok opname.
Status	Valid

6.3.53 Pengujian Validasi Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1)

Pada kasus pengujian melihat pupuk masuk gudang penyangga terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (operator diswil 1) yang ditunjukkan pada Tabel 6.77.

A. *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1).

Tabel 6.77 *Test-case* Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1)

Nama *Test-case*	Berhasil Melihat Pupuk Masuk Gudang Penyangga (Operator Diswil 1)
Prosedur	1. Telah login sebagai Operator Diswil 1. 2. Mengklik menu pupuk masuk penyangga.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk masuk dan memperlihatkan data pupuk masuk pada tabel pupuk masuk.
Hasil	Sistem memperlihatkan halaman menu pupuk masuk dan memperlihatkan data pupuk masuk pada tabel pupuk masuk.
Status	Valid

6.3.54 Pengujian Validasi Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1)

Pada kasus pengujian melihat pupuk keluar gudang penyangga terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (operator diswil 1) yang ditunjukkan pada Tabel 6.78.

A. *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1).

Tabel 6.78 *Test-case* Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1)

Nama *Test-case*	Berhasil Melihat Pupuk Keluar Gudang Penyangga (Operator Diswil 1)
Prosedur	1. Telah login sebagai Operator Diswil 1 2. Mengklik menu pupuk keluar penyangga.
Hasil yang diharapkan	Sistem akan memperlihatkan menu pupuk keluar dan memperlihatkan data pupuk keluar pada tabel pupuk keluar.
Hasil	Sistem memperlihatkan menu pupuk keluar dan memperlihatkan data pupuk keluar pada tabel pupuk keluar.
Status	Valid

6.3.55 Pengujian Validasi Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga).

Pada kasus pengujian Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga) terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga) yang ditunjukkan pada Tabel 6.79.

A. *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga).

Tabel 6.79 *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Laporan Pupuk Rusak (Operator Gudang Penyangga).
Prosedur	1. Telah login sebagai Operator Gudang Penyangga 2. Mengklik menu pupuk rusak.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.
Hasil	Sistem memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.
Status	Valid

6.3.56 Pengujian Validasi Melihat Laporan Pupuk Rusak (Manajer Diswil 1).

Pada kasus pengujian Melihat Laporan Pupuk Rusak (Manajer Diswil 1) terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Manajer Diswil 1) yang ditunjukkan pada Tabel 6.80.

A. *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Manajer Diswil 1).

Tabel 6.80 *Test-case* Berhasil Melihat Laporan Pupuk Rusak (Manajer Diswil 1)

Nama *Test-case*	Berhasil Melihat Laporan Pupuk Rusak (Manajer Diswil 1).
Prosedur	1. Telah login sebagai Manajer Diswil 1 2. Mengklik menu pupuk rusak.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.
Hasil	Sistem memperlihatkan halaman menu pupuk rusak dan memperlihatkan daftar laporan pupuk rusak pada tabel pupuk rusak.
Status	Valid

6.3.57 Pengujian Validasi Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga)

Dalam test-case Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga) terdapat satu *test-case* yaitu, *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga) yang ditunjukkan pada Tabel 6.81

A. *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga).

Tabel 6.81 *Test-case* Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Pengiriman Pupuk Rusak (Operator Gudang Penyangga)
Prosedur	1. Telah login sebagai Operator Gudang Penyangga 2. Mengklik menu pupuk rusak.
Hasil yang diharapkan	Sistem akan memperlihatkan halaman menu pupuk rusak dan tabel pengiriman pupuk rusak.

Hasil	Sistem memperlihatkan halaman menu pupuk rusak dan tabel pengiriman pupuk rusak.
Status	Valid

6.3.58 Pengujian Validasi Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir).

Dalam test-case melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir) terdapat satu *test-case* yaitu *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir) yang dapat diperlihatkan didalam tabel 6.82.

A. *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir).

Tabel 6.82 *Test-case* Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir)

Nama *Test-case*	Berhasil Melihat Permintaan Pengiriman Pupuk (Operator Rekanan Transportir).
Prosedur	1. Telah login sebagai Operator Rekanan Transportir. 2. Mengklik menu permintaan pupuk.
Hasil yang diharapkan	Sistem akan memperlihatkan daftar permintaan pengiriman pupuk di halaman menu permintaan pupuk.
Hasil	Sistem memperlihatkan daftar permintaan pengiriman pupuk di halaman menu permintaan pupuk.
Status	Valid

6.3.59 Pengujian Validasi Melihat Hasil Stok Opname (Operator Gudang Penyangga)

Dalam test-case melihat melihat hasil stok opname (Operator Gudang Penyangga) terdapat satu *test-case* yaitu *test-case* berhasil melihat hasil stok opname (Operator Gudang Penyangga) yang dapat diperlihatkan didalam tabel 6.83.

A. *Test-case* Berhasil Melihat Hasil Stok Opname (Operator Gudang Penyangga).

Tabel 6.83 *Test-case* Berhasil Melihat Hasil Stok Opname (Operator Gudang Penyangga)

Nama *Test-case*	Berhasil Melihat Hasil Stok Opname (Operator Gudang Penyangga)
Prosedur	1. Telah login sebagai Operator Gudang Penyangga 2. Mengklik menu stok opname.

	3. Mengklik *button* hasil pada stok opname yang diinginkan.
Hasil yang diharapkan	Sistem akan memperlihatkan modal hasil stok opname.
Hasil	Sistem memperlihatkan modal hasil stok opname.
Status	Valid

6.3.60 Pengujian Validasi Melihat Hasil Stok Opname yang Sudah Divalidasi (Manajer Diswil 1).

Dalam test-case melihat hasil stok opname yang sudah divalidasi (Manajer Diswil 1) terdapat satu *test-case* yaitu *test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Manajer Diswil 1) yang dapat diperlihatkan didalam tabel 6.84.

A. *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Manajer Diswil 1).

Tabel 6.84 *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Manajer Diswil 1)

Nama *Test-case*	Berhasil Melihat Stok Opname yang Sudah Divalidasi (Manajer Diswil 1).
Prosedur	1. Telah login sebagai Manajer Diswil 1. 2. Mengklik menu stok opname. 3. Mengklik *button* hasil pada tabel daftar tugas stok opname yang dikerjakan
Hasil yang diharapkan	Sistem akan memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Hasil	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Status	Valid

6.3.61 Pengujian Validasi Melihat Hasil Stok Opname yang Sudah Divalidasi (Operator Diswil 1).

Dalam test-case melihat hasil stok opname yang sudah divalidasi (Operator Diswil 1) terdapat satu *test-case* yaitu *test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Diswil 1) yang dapat diperlihatkan didalam tabel 6.85.

A. *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Diswil 1).

Tabel 6.85 *Test-case* Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Diswil 1)

Nama *Test-case*	Berhasil Melihat Stok Opname yang Sudah Divalidasi (Operator Diswil 1).
Prosedur	1. Telah login sebagai Operator Diswil 1 2. Mengklik menu stok opname. 3. Mengklik *button* hasil pada tabel daftar tugas stok opname yang dikerjakan
Hasil yang diharapkan	Sistem akan memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Hasil	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Status	Valid

6.3.62 Pengujian Validasi Melihat Stok Pupuk.

Dalam test-case Melihat Stok Pupuk terdapat satu *test-case* yaitu *test-case* Berhasil Melihat Stok Pupuk yang dapat diperlihatkan didalam tabel 6.86.

A. *Test-case* Berhasil Melihat Stok Pupuk.

Tabel 6.86 *Test-case* Berhasil Melihat Stok Pupuk

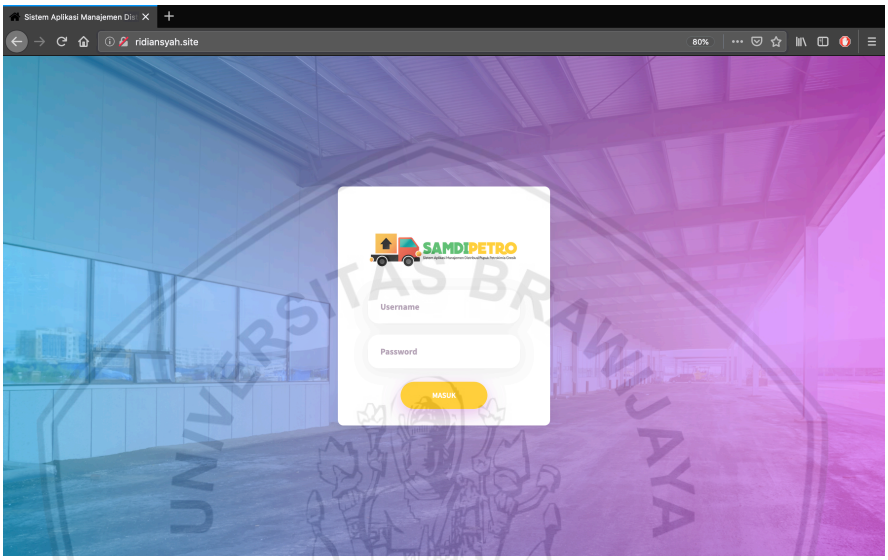
Nama *Test-case*	Berhasil Melihat Stok Pupuk
Prosedur	1. Telah login sebagai Operator Gudang Penyangga 2. Mengklik menu stok pupuk pada kategori menu pupuk
Hasil yang diharapkan	Sistem akan memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Hasil	Sistem memperlihatkan data hasil stok opname yang sudah divalidasi pada form hasil stok opname.
Status	Valid

6.4 Pengujian Kompatibilitas

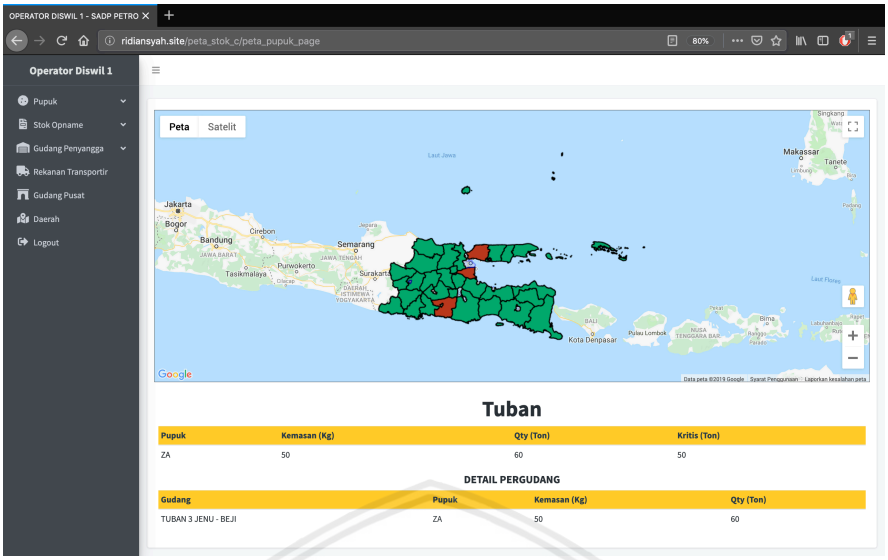
Pengujian kompatibilitas dikerjakan dengan menjalankan semua fungsi sistem sesuai dengan skenario pada pengujian validasi melalui tiga peramban yang berbeda agar dapat memperlihatkan apakah sistem dapat digunakan dengan baik pada masing-masing peramban. Hasil pengujian kompatibilitas dapat dilihat dari Gambar 6.5 sampai 6.22. Setelah menjalankan sistem pada peramban Chrome, Edge, dan Firefox sistem berjalan dengan baik tanpa menemukan kendala.

6.4.1 Pengujian Kompatibilitas Peramban Mozilla Firefox

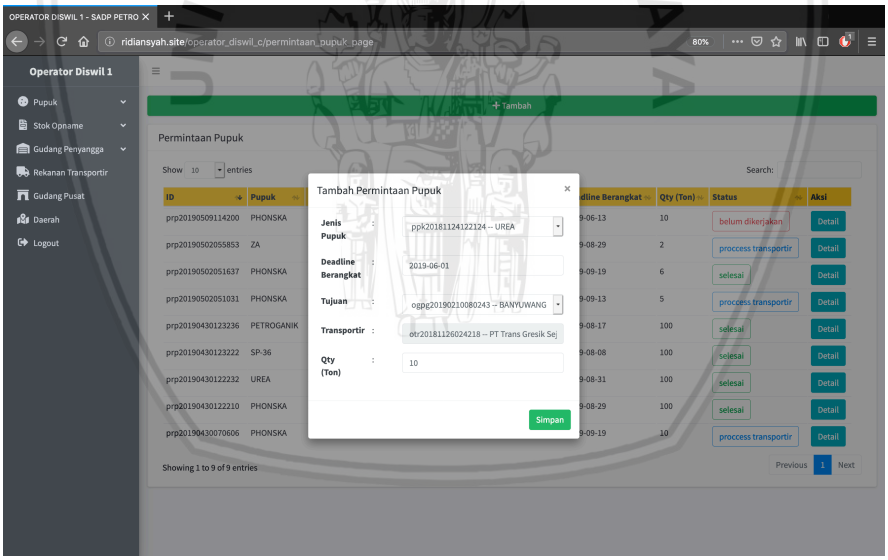
Pengujian kompatibilitas untuk peramban Mozilla Firefox dilakukan dengan menjalankan sistem melalui peramban tersebut untuk melihat apakah sistem mampu berjalan dengan baik. Hasil pengujian kompatibilitas dari peramban Mozilla Firefox dapat dilihat pada Gambar 6.5 sampai 6.10.



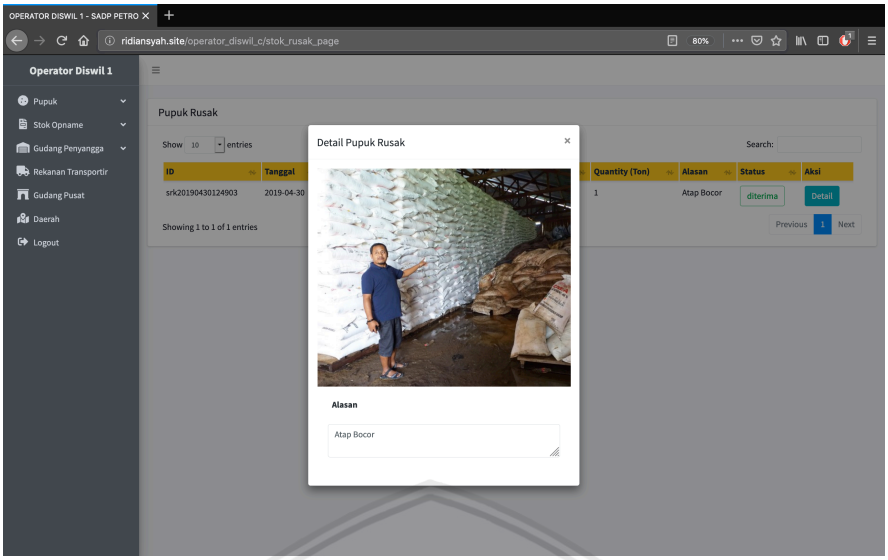
Gambar 6.5 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 1



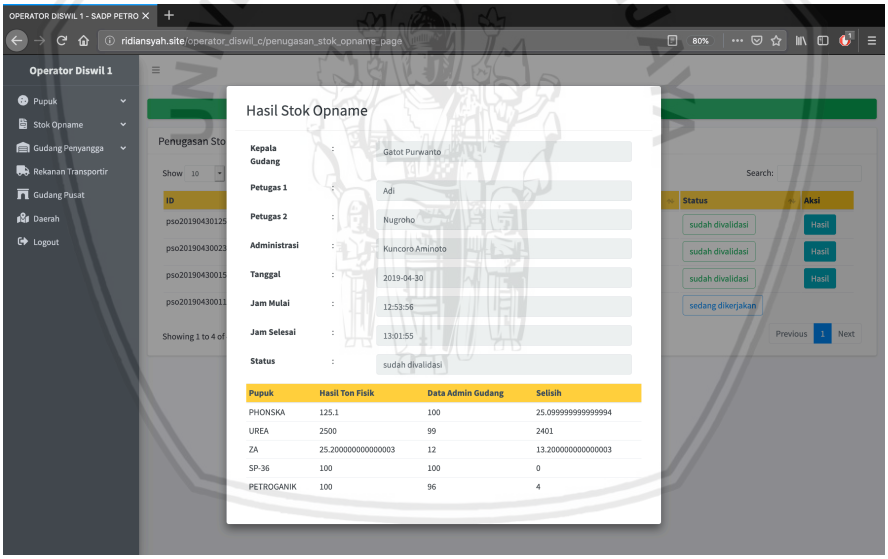
Gambar 6.6 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 2



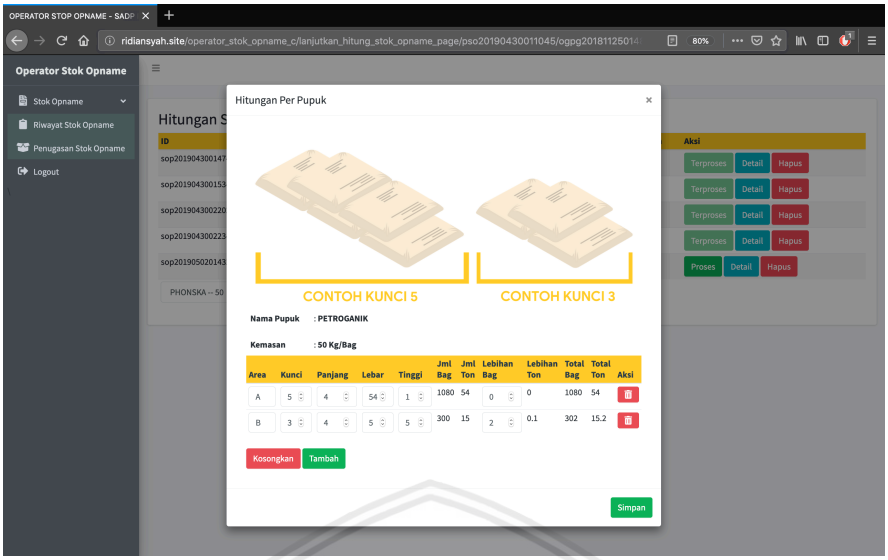
Gambar 6.7 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 3



Gambar 6.8 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 4



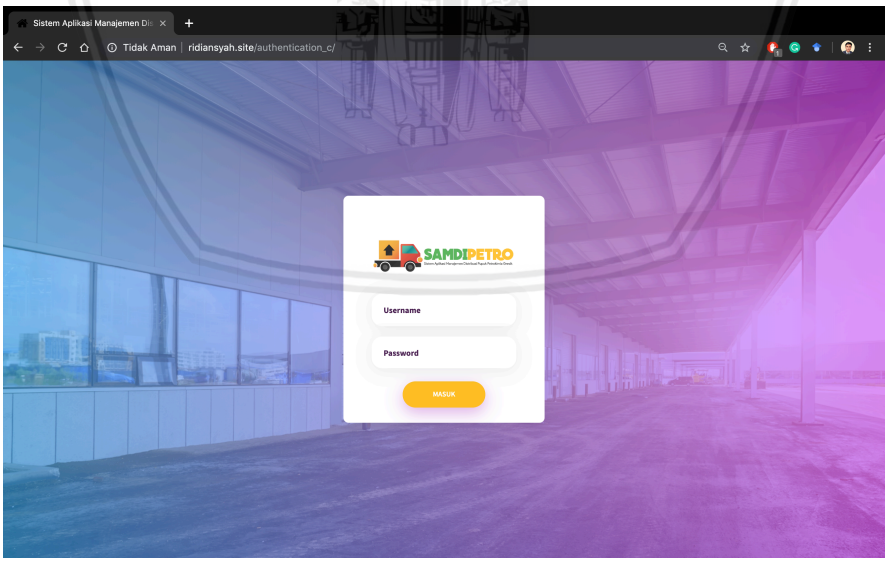
Gambar 6.9 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 5



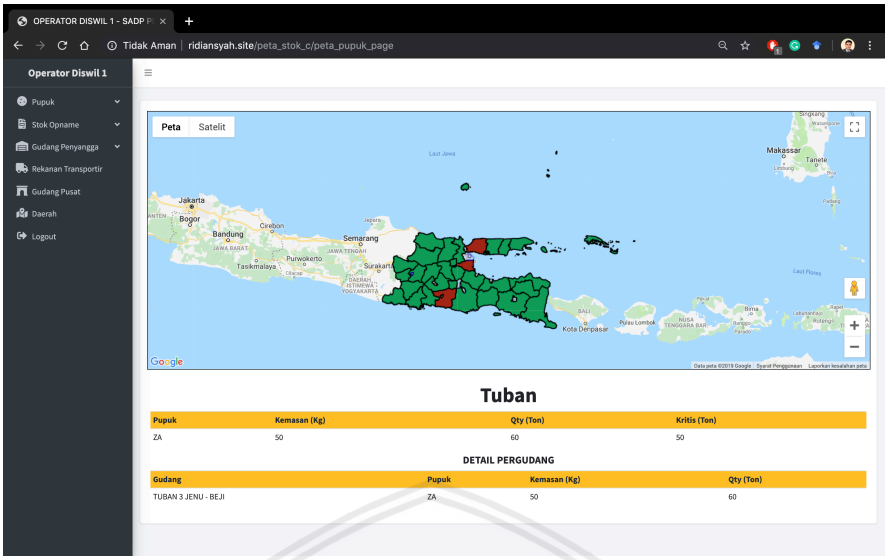
Gambar 6.10 Pengujian Kompatibilitas pada Peramban Mozilla Firefox 6

6.4.2 Pengujian Kompatibilitas Peramban Google Chrome

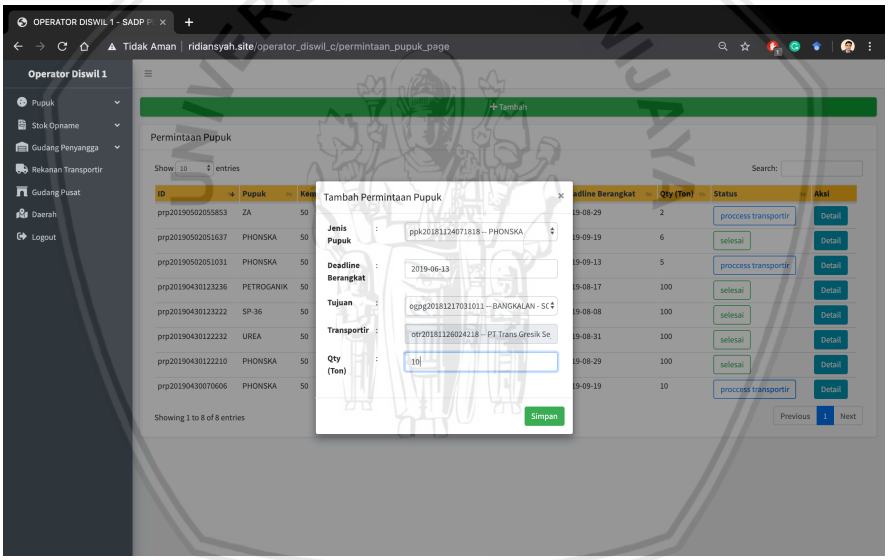
Pengujian kompatibilitas untuk peramban Google Chrome dilakukan dengan menjalankan sistem melalui peramban tersebut untuk melihat apakah sistem mampu berjalan dengan baik. Hasil pengujian kompatibilitas dari peramban Google Chrome dapat dilihat pada Gambar 6.11 sampai 6.16.



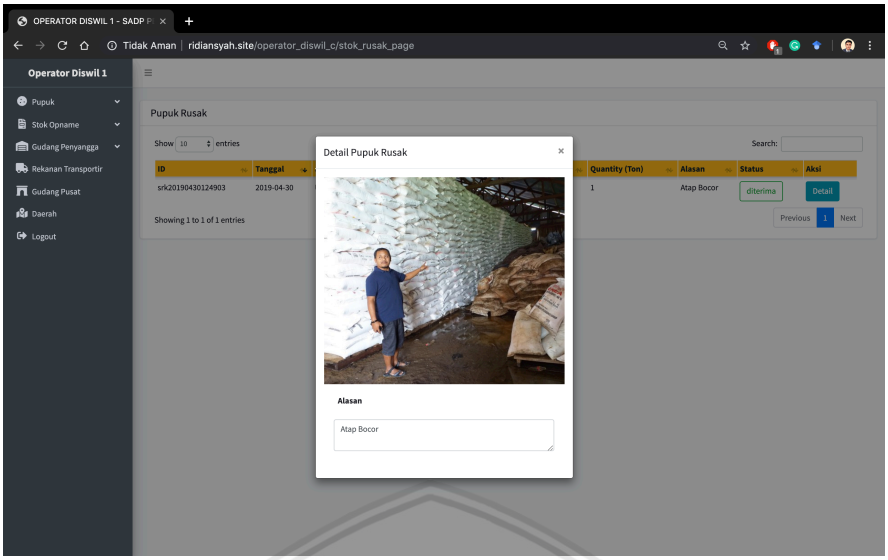
Gambar 6.11 Pengujian Kompatibilitas pada Peramban Google Chrome 1



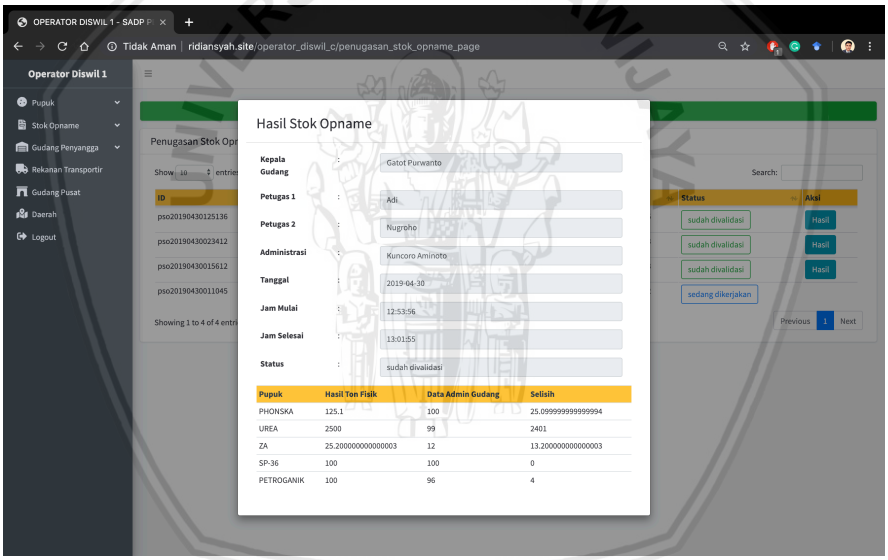
Gambar 6.12 Pengujian Kompatibilitas pada Peramban Google Chrome 2



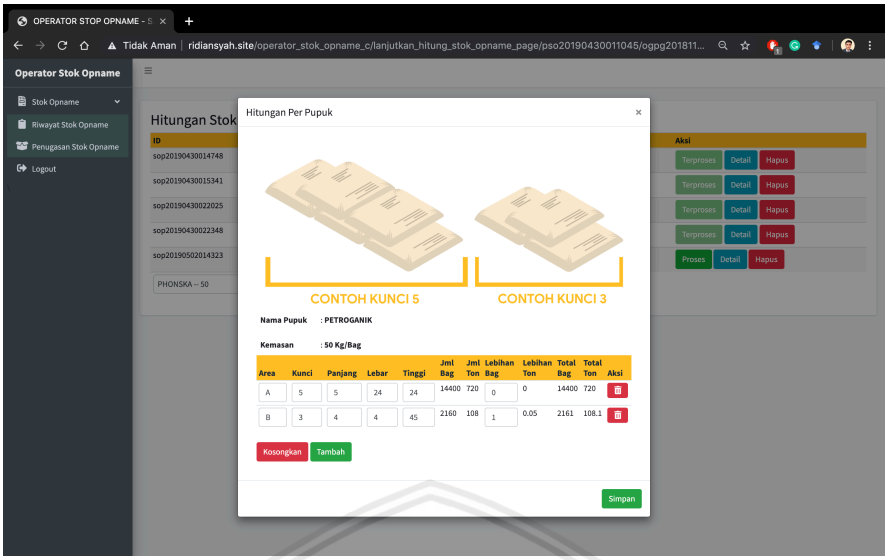
Gambar 6.13 Pengujian Kompatibilitas pada Peramban Google Chrome 3



Gambar 6.14 Pengujian Kompatibilitas pada Peramban Google Chrome 4



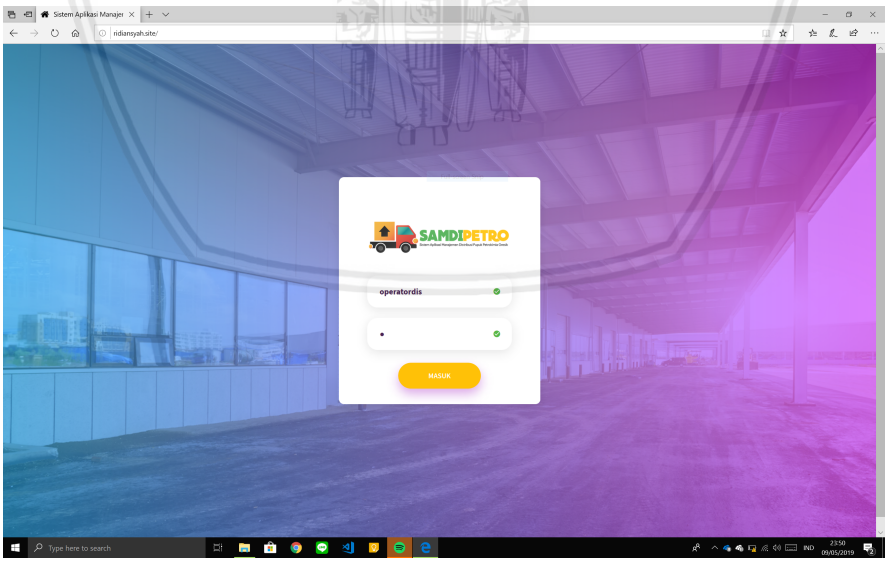
Gambar 6.15 Pengujian Kompatibilitas pada Peramban Google Chrome 5



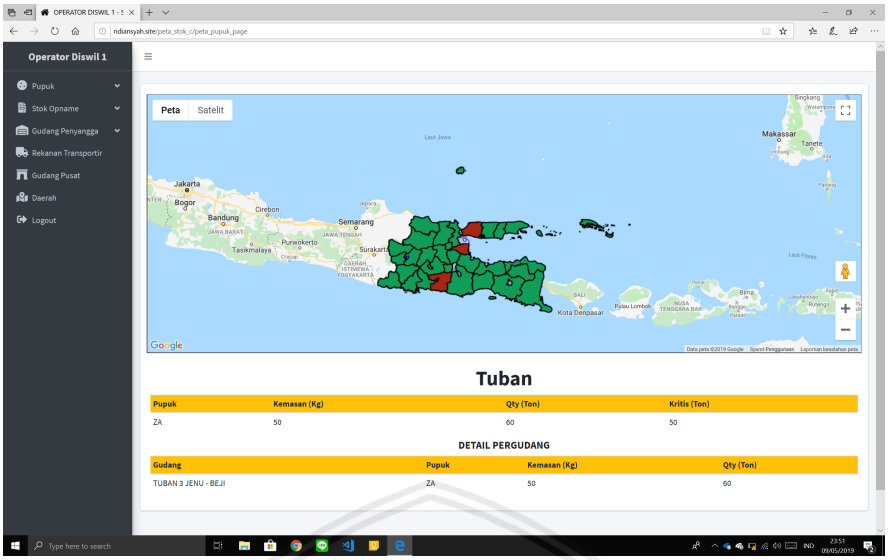
Gambar 6.16 Pengujian Kompatibilitas pada Peramban Google Chrome 6

6.4.3 Pengujian Kompatibilitas Peramban Microsoft Edge

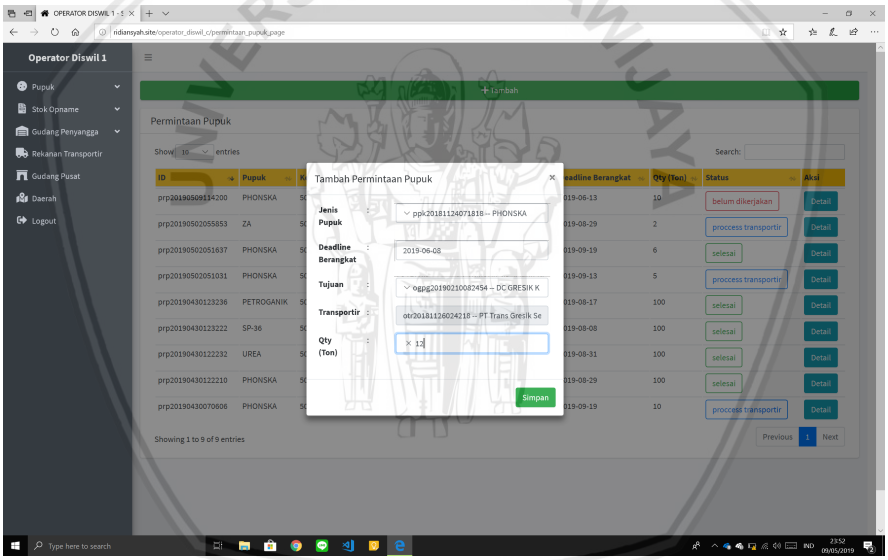
Pengujian kompatibilitas untuk peramban Microsoft Edge dilakukan dengan menjalankan sistem melalui peramban tersebut untuk melihat apakah sistem mampu berjalan dengan baik. Hasil pengujian kompatibilitas dari peramban Microsoft Edge dapat dilihat pada Gambar 6.17 sampai 6.22.



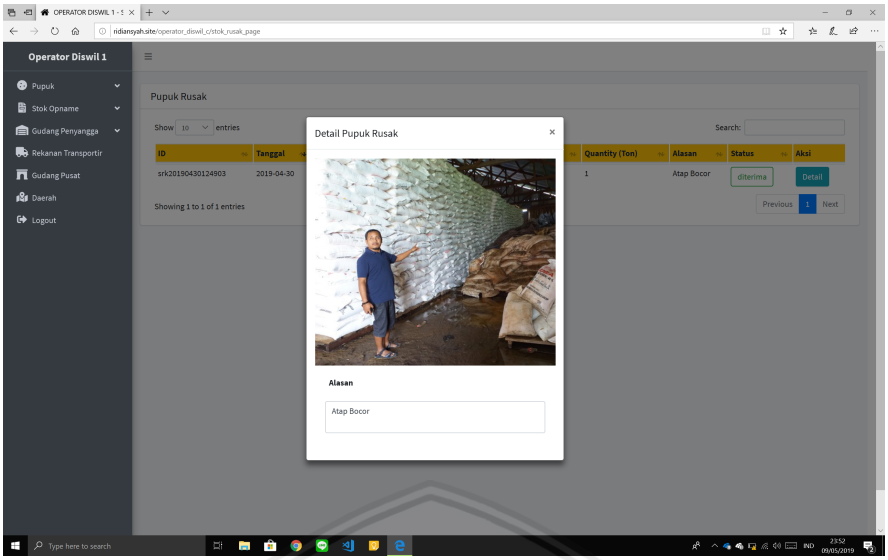
Gambar 6.17 Pengujian Kompatibilitas pada Peramban Microsoft Edge 1



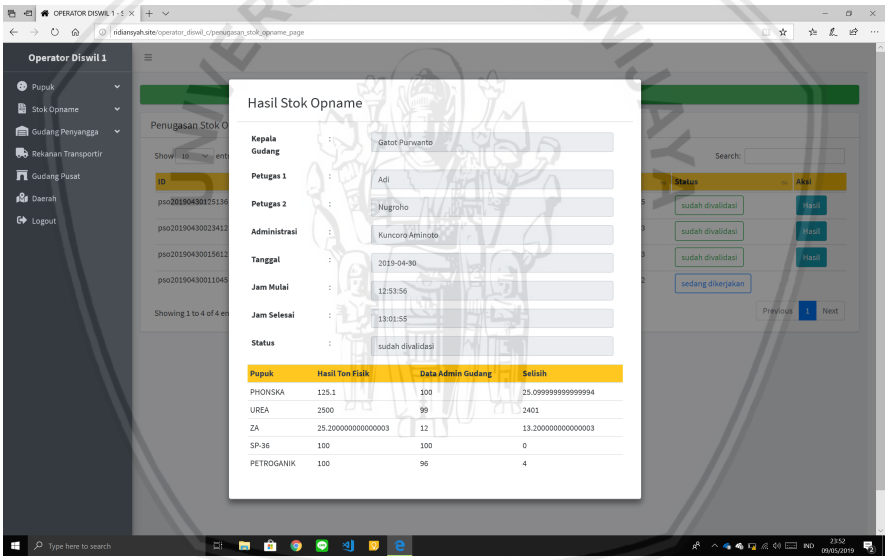
Gambar 6.18 Pengujian Kompatibilitas pada Peramban Microsoft Edge 2



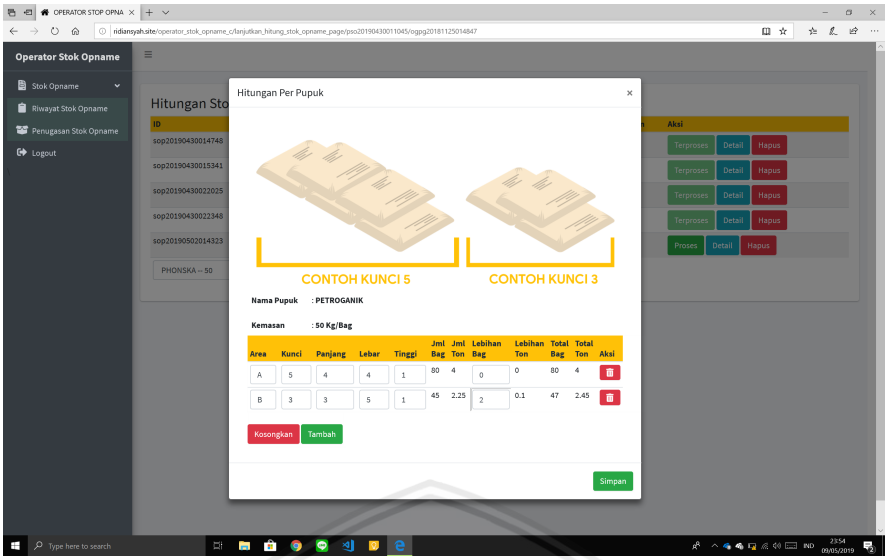
Gambar 6.19 Pengujian Kompatibilitas pada Peramban Microsoft Edge 3



Gambar 6.20 Pengujian Kompatibilitas pada Peramban Microsoft Edge 4



Gambar 6.21 Pengujian Kompatibilitas pada Peramban Microsoft Edge 5



Gambar 6.22 Pengujian Kompatibilitas pada Peramban Microsoft Edge 6

6.5 Hasil Analisis Pengujian

Pada bagian/fase ini dijabarkan hasil dari pada analisis pengujian dari tiga pengujian yang dilakukan, yakni pengujian unit, pengujian integrasi dan pengujian validasi. Dalam pengujian unit dilakukan pengujian kepada tiga sampel uji. Pada save_detail_permintaan_pupuk () nilai *cyclomatic complexity* yang didapatkan yaitu 5, pada set_kml() nilai cyclomatic complexity yang didapatkan yaitu 8, dan pada save_detail_stok_opname() nilai *cyclomatic complexity* yang didapatkan yaitu 2 dengan hasil 100% valid. Pada pengujian integrasi pada fungsi update_rekanan_gudang_penyangga nilai *cyclomatic complexity* yang didapatkan yaitu 2 dengan hasil 100% valid. Pada hasil pengujian validasi yang dilakukan pada semua kebutuhan yaitu 62. 100% menghasilkan status valid dan tidak ditemukan kesalahan dalam sistem. Sedangkan pada hasil pengujian kompatibilitas, didapatkan sistem dapat berjalan dengan baik pada tiga jenis peramban yang berbeda. Hal ini dapat disimpulkan bahwa Sistem Aplikasi Manajemen Distribusi Pupuk dapat digunakan pengguna dengan baik pada jenis peramban yang dipakai.

BAB 7 PENUTUP

Tahap terakhir dalam penelitian ini yaitu bab penutup difokuskan dalam merumuskan kesimpulan dari penelitian ini maupun saran yang dapat dipergunakan untuk pengembangan selanjutnya dari penelitian ini ataupun perangkat lunak yang dihasilkan.

7.1 Kesimpulan

Setelah setiap fase dapat atau telah dilewati yaitu analisis kebutuhan, perancangan sistem, implementasi perangkat lunak dan pengujian terhadap sistem yang dibuat, untuk itu telah disimpukan beberapa poin yaitu:

1. Menurut hasil dalam fase analisis kebutuhan dapat dirumuskan terdapat enam puluh dua kebutuhan fungsional dan satu kebutuhan non fungsional. Aktor yang ada dalam sistem berjumlah delapan ialah Operator Diswil 1, Operator Gudang Penyangga, Operator Rekanan Transportir, Operator Gudang Pusat, Manajer Diswil 1, Operator Stok Opname, Operator Rekanan Pengelola Gudang Penyangga, dan *Guest*.
2. Menurut hasil pada tahap perancangan sistem dapat disusun perancangan dengan berisikan perancangan *sequence* diagram, *class* diagram, perancangan *database*, perancangan komponen, dan perancangan antarmuka yang merupakan hasil dari merumuskan hasil analisis kebutuhan.
3. Menurut hasil dalam fase implementasi yang telah dilakukan, diperoleh hasil sistem aplikasi manajemen distribusi pupuk yang memiliki fitur-fitur penting yakni melihat kondisi stok daerah, menghitung stok opname, membuat laporan pupuk rusak, mengirim pupuk ke gudang penyangga, dan mengirim pupuk rusak. Sistem dibangun dengan mempergunakan bahasa pemrograman *PHP, Ajax, JavaScript, HTML,* dan *CSS* yang diterapkan pada *framework codeigniter*.
4. Menurut hasil dalam fase pengujian siistem yang mampu dikerjakan dengan menggunakan cara pengujian kotak putih yang digunakan pada pengujian unit dan pengujian integrasi. Sedangkan untuk, pengujian kotak hitam pada pengujian validasi. Pengujian unit dilakukan terhadap tiga fungsi yaitu save_detail_permintaan_pupuk () nilai *cyclomatic complexity* yang didapatkan adalah lima, pada set_kml() nilai *cyclomatic complexity* yang didapatkan adalah delapan, dan pada save_detail_stok_opname() nilai *cyclomatic complexity* yang didapatkan adalah dua. Pengujian validasi dilakukan terhadap enam puluh dua kebutuhan fungsional. Kemudian dilakukan pengujian nonfungsional yaitu kompatibilitas *testing* yang diuji pada *web* peramban *Chrome, Firefox,* dan *Edge*. Pada fase saat semua pengujian telah dilakukan dapat memproleh hasil akhir 100% semuanya valid dengan tidak ditemukannya kesalahan.

7.2 Saran

Beberapa saran diperuntukan kepada pengembangan selanjutnya dari aplikasi manajemen distribusi pupuk berbasis web untuk pengembangan setelah penelitian ini ialah:

1. Menambahkan fungsi agar dapat membantu proses klaim tahunan kepada rekanan pengelola gudang penyangga dan rekanan transportir.
2. Menambah kebutuhan untuk proses pendaftaran kendaraan atau truk milik rekanan transportir.
3. Menambah aktor distributor untuk melakukan *booking* pengambilan pupuk digudang penyangga.

DAFTAR REFERENSI

Huda, Miftahul dan Bunafit Komputer, 2010. *Membuat Aplikasi Database dengan Java, MySQL dan Netbeans*. [e-book] Jakarta: PT Elex Media Komputindo. Tersedia di: Google Books <https://books.google.co.id/books?isbn=9792773150> [Diakses 25 September 2018]

Irnawati, O., 2018. *Implementasi Metode Waterfall Pada Sistem Informasi Stock Opname*. IJSE - Indonesian Journal on Software Engineering. Tersedia di: < http://ijse.web.id/jurnal/index.php/ijse/article/view/71> [Diakses 25 September 2018]

Jubilee Enterprise, 2014. *HTML 5 Manual Book*. [e-book] Jakarta: PT Elex Media Komputindo. Tersedia di: Google Books <https://books.google.co.id/books?isbn=9786020253572> [Diakses 25 September 2018]

Juniardi, F. dan Azwansyah, H., 2014. *Penyusunan Sistem Informasi Geografis Infrastruktur Transportasi Kabupaten Kapuas Hulu Berbasis WEB*. Jurnal ELKHA, Vol. 6, No. 1. Tersedia di: <http://jurnal.untan.ac.id/index.php/Elkha/article/view/5181/5334> [Diakses 25 September 2018]

Pressman, R. S., 2012. *Rekayasa Perangkat lunak - Buku Satu, Pendekatan Praktisi (Edisi 7)*. Yogyakarta: ANDI.

PT Petrokimia Gresik. *Sistem Distribusi*. [online] Tersedia di: <http://www.petrokimia-gresik.com/Pupuk/Sistem.Distribusi> [Diakses 25 September 2018]

PT Petrokimia Gresik. *Profile Perusahaan.* [online] Tersedia di: <http://www.petrokimia-gresik.com/Resources/Docs/CP%20Petrokimia%20Gresik%20small.pdf> [Diakses 25 September 2018]

Rosa, A. S. & Shalahuddin, M., 2018. Rekayasa Perangkat Lunak Terstuktur dan Berorientasi Objek. Bandung: Informatika.

Sommerville, I., 2011. Software Engineering: Ninth Edition. Boston: Pearson Education.

Sunyoto, A., 2007. *Ajax Memabangun Web dengan Teknologi Asynchronouse JavaScript dan XML*. [e-book] Yogyakarta: ANDI. Tersedia di: Google Books <https://books.google.co.id/books?isbn=9789792902228> [Diakses 25 September 2018]

Susilo, E. B. dan Listyorini, T., 2015. *Perancangan Sistem Informasi Distribusi Obat Pasien Rawat Inap*. Universitas Muria Kudus. Tersedia di: <https://jurnal.umk.ac.id/index.php/SNA/article/view/360/370> [Diakses 25 September 2018]

Tim EMS, 2016. *PHP 5 dari Nol*. [e-book] Jakarta: PT Elex Media Komputindo. Tersedia di: Google Books <https://books.google.co.id/books?isbn=6020283607> [Diakses 25 September 2018]

Tone, K., 2016. *Rancang Bangun Sistem Informasi Distribusi Bantuan Sosial Beras Miskin (Studi Kasus Kecamatan Binamu Kabupaten Jeneponto).* UIN Alauddin Makassar. Tersedia di: <http://journal.uin-alauddin.ac.id/index.php/instek/article/view/2547> [Diakses 25 September 2018]

Wardana, 2010. *Menjadi Master PHP dengan Framework Codeigniter*. [e-book] Jakarta: PT Elex Media Komputindo. Tersedia di: Google Books <https://books.google.co.id/books?isbn=9792781897> [Diakses 25 September 2018]

LAMPIRAN A SURAT PERNYATAAN PENELITIAN

PETROKIMIA
GRESIK

SURAT KETERANGAN
No : 4192/K.02.02/03/MKP/2018

Dengan ini kami menerangkan bahwa mahasiswa tersebut dibawah ini :

- o Nama : Moh. Wahyu Dwi Ridiansyah
- o Nomor Induk : 155150200111232
- o Program Studi : T. Informatika - FILKOM - Universitas Brawijaya

Telah menyelesaikan kegiatan Penelitian di PT Petrokimia Gresik pada tanggal 01 - 31 Oktober 2018.
Selama kegiatan Penelitian tersebut tidak pernah melanggar peraturan yang berlaku dan telah melaksanakan tugasnya dengan baik.

Demikian surat keterangan ini kami buat untuk dapat dipergunakan sebagaimana mestinya.

Gresik, 31 Oktober 2018
PT Petrokimia Gresik

PETROKIMIA GRESIK
DEP. PENGEMBANGAN SDM

Tjaturtjitra Suhitarini, SE. MM
Manager Pengembangan SDM

PT Petrokimia Gresik
Petrokimia Gresik Building
Jln. Jenderal A. Yani - Gresik 61119 - Indonesia
P. +62 31 3982100, 3982200
F. +62 31 3981722, 3982272
E. pkg@petrokimia-gresik.com
www.petrokimia-gresik.com

LAMPIRAN B HASIL WAWANCARA

A. WAWANCARA TAHAP PERTAMA

Wawancara tahap pertama difokuskan pada aspek identifikasi masalah.

1. Apakah ada kendala yang anda dihadapi dalam pelaksanaan tanggung jawab dan tugas Departemen Distribusi Wilayah PT Petrokimia Gresik?
   Jawaban:
   Ada mas. Mengenai stok opname, pelaporan pupuk rusak dan pencatatan laporan stok gudang penyangga.

2. Bisa dijelaskan masalah yang dihadapi sekarang mengenai stok opname itu seperti apa?
   Jawaban:
   Secara periodik kami melakukan stok opname disetiap Gudang Penyangga, selanjutnya kami akan memasukan data stok yang tercatat dikantor kedalam file *document* untuk menjadi bahan stok opname nanti. Pada saat penghitungan stok opname masih menghitung manual menggunakan kertas untuk nanti hasilnya dimasukan kedalam file dokumen dan diprint untuk meminta tanda tangan persetujuan dari kepala gudang. Karena hasil yang kami bawa berupa *hardcopy* beberapa kali terjadi kehilangan dokumen sehingga kami harus melakukan stok opname lagi. Jadi dana perjalanan kita habis banyak mas. Belum lagi waktu sampai di kantor kita harus menyerahkan hasil stok opname untuk disesuaikan pencatatan jumlah stoknya dikantor dua kali kerja tidak efisien mas. Padahal, dalam sekali penugasan stok opname staff distribusi ditugaskan bisa untuk melakukan stok opname untuk Gudang Penyangga dalam satu provinsi.

3. Secara periodik itu berapa lama sekali ya mas?
   Jawaban:
   Tidak pasti mas tergantung penugasan dari kantor

4. Kenapa kegiatan stok opname ini harus dilakukan?
   Jawaban:
   Untuk memastikan stok yang ada di gudang penyangga sesuai dengan stok yang tercatat. Dari stok opname juga dapat diketahui adanya ketidaksesuaian jumlah stok pupuk atau kehilangan pupuk di Gudang penyangga. Karena Gudang penyangga ini tidak langsung dikelola oleh pegawai PT Petrokimia Gresik melainkan bekerja sama dengan rekanan pengelola.

5. Apakah ada keluhan dari perusahaan rekanan pengelola gudang penyangga?
   Jawaban:
   Yang saya tahu mereka sering mengeluh tidak langsung mengetahui hasil stok opname ketika ada perbedaan hasil stok opname dan stok yang

tercatat. Jadi, mereka tahunya setelah pihak petro mengirimkan daftar selisih stok opname pada saat sebelum klaim tahunan. Padahal menurut mereka jika mereka langsung mengetahui adanya kehilangan sesaat setelah proses stok opname selesai bisa menjadi bahan pertimbangan untuk melakukan suatu tindakan ke kepala gudang penyangganya, entah itu teguran atau apa.

6. Ketika ada selisih stok tercatat dengan hasil stok opname, apa tindakan selanjutnya yang dilakukan?
Jawaban:
Mengenai itu nanti dilakukan pada saat klaim tahunan mas. Misal kasus A itu alasannya apa biasanya rekanan membawa bukti. nah nanti disitu ditentukan berapa yang ganti rugi berapa yang tidak. Biasanya yang tidak itu karena kesalahan *input* mas. Kalau masalah itu nanti urusan atasan waktu klaim tahunan mas bukan di stok opname, intinya stok opname disini untuk memeriksa pencatatan dengan stok *real* agar akurat.

7 Bisa dijelaskan seperti apa permasalahan dan dampak yang ada mengenai masalah pelaporan pupuk rusak?
Jawaban:
Ya yang sekarang ini pelaporannya melalui *email* mas. Jadi, dikirimkan fotonya lalu alasannya nanti disini terima apa tidak kalau terima baru kita melakukan pencatatan dan penyesuaian data jumlah stok pupuk, agak kurang *efisien* mas tidak bisa langsung otomatis tercatat, padahal kadang data yang diterima juga cukup banyak belum lagi sama data pelaporan yang lain. Selain itu juga, belum ada pemantauan pengiriman pupuk rusak dari gudang penyangga ke pusat untuk didaur ulang, jadi selama ini kita agak kesulitan memantau pengiriman pupuk rusak dari gudang penyangga sudah dikirimkan ke gudang pusat apa belum. Sehingga pengelolaan pupuk rusak kurang terawasi.

8. Bisa dijelaskan seperti apa permasalahan dan dampak yang ada mengenai pencatatan laporan stok gudang penyangga?
Jawaban:
Selama ini untuk pemantauan data setiap gudang penyangga kami hanya menggunakan data pelaporan berupa file dokumen yang dikirim oleh kepala Gudang penyangga melalui email ke Departemen Distribusi Wilayah 1 untuk selanjutnya nanti dimasukan kedalam file excel oleh staff diswil 1. Staff diswil 1 sering mengalami kendala dalam mengelola data yang dikirimkan karena banyaknya *email* yang dikirim sehingga sering terjadi data yang terlewat tidak terolah dan itu membuat proses pengeloaan data menjadi lama atau tidak *efektif*. Pemantauan data stok di gudang penyangga juga masih menggunakan *table data* excel yang didalamnya berisi transaksi keluar dan masuk stok. Sehingga berakibat sulitnya memantau stok disetiap gudang penyangga secara cepat dan mudah. Padahal, kecepatan dan

kemudahan dalam memantau stok disetiap Gudang akan berpengaruh pada proses pengawasan ketersediaan pupuk disetiap daerah untuk menghindari adanya kekurangan stok pupuk. Karena jika terjadi kekurangan stok maka kebutuhan petani didaerah tersebut tidak dapat terpenuhi dan akan berpengaruh pada hasil pertaniannya. Padahal ketika data stok dapat dilihat secara *real time* dan bisa dilihat oleh diswil bisa langsung ditindak lanjuti dengan melakukan pengiriman dari Gudang pusat ke Gudang Penyangga. Selain itu, Manager juga sulit memantau keadaan stok yang ada digudang-gudang penyangga. untuk menjalankan tugasnya mengawasi pengendalian stok yang ada dengan mudah, melakukan pengecekan melalui daftar table excel sangatlah tidak efisien. Padahal, manager memiliki banyak sekali kegiatan

Gresik, 9 Oktober 2018
Staff Distribusi Wilayah 1
PT Petrokimia Gresik

Rachmad Edy Raflanto

Staff Distribusi Wilayah 1
PT Petrokimia Gresik

Edwyk Sony Udaleby

B. WAWANCARA TAHAP KEDUA

Wawancara tahap kedua di fokuskan pada aspek pengenalan alur proses yang ada dan pengumpulan data.

1. Bagaimana alur proses stok opname dari penugasan sampai selesai?
   Jawaban:
   Pertama manajer melalui staffnya diswil 1 akan menugaskan tim untuk melakukan stok opname satu tim terdiri dari 2 orang staff diswil dan 1 staff administrasi. Penugasannya sesuai area tanggung jawab tim tersebut. Setelah itu tim tersebut mempersiapkan data-data keluar-masuk pupuk sesuai yang tercatat laporan yang ada di kantor Setelah itu berangkat menuju gudang-gudang penyangga. Disana tim petugas stok opname akan menghiturig stok fisik pupuk yang benar-benar ada disana. Apakah benar jumlah pupuk yang ada di gudang penyangga sesuai dengan yang tercatat dikantor, setelah itu petugas akan melaporkan dalam dokumen hasil stok opname. Dalam dokumen hasil stok opname tersebut juga perlu persetujuan kepala gudang penyangga mas meskipun pada perhitungannya sesuai ataupun tidak. Kalau tidak nanti kita lakukan hitung ulang lagi mana yang dianggap tidak sesuai, sampai kepala gudang menyetujui. Setelah semua selesai petugas akan pindah ke gudang berikutnya dan seterusnya. Setelah selesai semua nanti petugas akan serahkan hasil stok opname yang telah petugas lakukan ke kantor PT Petrokimia Gresik. Lalu dilakukan juga menyesuaian catatan stok dari hasil stok opname. Setelah itu data-data hasil final dari perhitungan stok opname per gudang penyangga akan di serahkan pada Manajer Diswil 1.

2. Apa data yang harus dimasukkan ke perhitungan stok opname mas ?
   Jawaban:
   Area, Perhitungan dengan metode kunci, hasil, lebihan, total. Nanti saya berikan filenya untuk lebih jelasnya.

3. Bisa dijelaskan apa itu perhitungan dengan metode kunci?
   Jawaban:
   Jadi pupuk itukan tidak kotak ya mas? jadi ketika *dijejer* itu ada cela diantara 2 pupuk itu, nah itu ditaruh pupuk disitu. Nah maksudnya misal kunci 5 ya itu berarti ada 3 pupuk dibawah diatasnya ada 2 pupuk yang mengisi sela diantara pupuk itu. Jadi nanti hitungnya tiap itu jadi cepat mas gak perlu satu-satu.

4. Apa saja yang isi dokumen laporan hasil stok opname?
   Jawaban:
   Nama kepala gudang penyangga, petugas 1 dan 2, petugas adminitrasi, tanggal stok opname, jam mulai dan jam selesai, lalu hasil stok fisik dari setiap pupuk, data stok tercatat, lalu ada selisih dari stok fisik dan stok tercatat.

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

5. Untuk penjemputan pupuk dari Gudang Pusat dan diantarkan ke Gudang Penyangga bagaimana mas alurnya ?
   Jawaban:
   Nah disini kita ada sistem mas, tapi cuma sampai ambil pupuk di gudang pusat. Jadi prosesnya diswil membuat permintaan penjemputan pupuk ke Gudang Pusat untuk dikirimkan ke Gudang Penyangga yang diinginkan melalui sistem lalu Rekanan Transportir yang bertanggung jawab di gudang penyangga tersebut mengisi berapa kendaraan yang ditugaskan untuk mengantar pupuk plat nomornya berapa, nama supirnya siapa, dan berapa ton yang akan dimuat kendaraan tersebut. Setelah truk sampai di gudang pusat, admin gudang pusat akan mengkonfirmasi jika truk sudah dimuat dan berangkat ke gudang penyangga.

6. Setelah berangkat proses selanjutnya bagaimana mas?
   Jawaban:
   Ohh iya, ini diluar sistem ya. Disana nanti setelah sampai digudang penyangga ditimbang ulang muatannya. apabila ada selisih berarti dinyatakan hilang diperjalanan. Setelah ditimbang dan supir setuju dengan hasilnya admin gudang penyangga akan mencatat quantity hasil timbangan sebagai jumlah quantity pupuk yang diterima.

7 Data apa yang diisikan oleh staff diswil di permintaan pengiriman pupuk?
   Jawaban:
   Pupuk yang mau dikirim, deadline berangkat, quantity (ton), gudang penyangga tujuan.

8. Data apa saja yang dilaporkan kepala gudang penyangga kepada departemen distribusi?
   Jawaban:
   Pupuk masuk meliputi Jenis pupuk, berat pupuk yang diterima, transportir yang ngirim pupuk, nomor polisi truk, berat per kemasan, tanggal sampai. Pupuk keluar meliputi nama distributor yang mengambil, nomor polisi, jenis pupuk, berat perkemasan, quantity yang diambil, tanggal dan jam pengambilan.

9. Data apa saja yang diperlukan Departemen Distribusi Wilayah 1 untuk diketahui secara cepat dan realtime?
   Jawaban:
   Jumlah stok pupuk tiap jenis perdaerah apakah aman atau kurang.

10. Apa indikator bahwa stok didaerah dikatakan aman dan kurang?
    Jawaban:
    Ada namanya *safety stock* atau orang disini nyebutnya batas stok kritis mas dan disetiap daerah beda-beda tiap pupuk juga beda-beda. Tidak baku bisa

berubah setiap saat. Nanti kita yang menentukan. Kalau dibawah itu artinya kurang aman.

11. Apa saja jenis pupuk yang dikirimkan di gudang penyangga?
    Jawaban:
    Pupuk-pupuk bersubsidi yaitu UREA, ZA, SP-36, PHONSKA dan PETROGANIK.

12. Data apa saja yang perlu dicatat dalam proses pupuk keluar dari gudang penyangga ke distributor?
    Jawaban:
    Nama distributor, pupuk yang diambil, nomor polisi kendaraan, quantity yang diambil, tanggal dan jam pengambilan.

13. Bagaimana proses pelaporan pupuk rusak?
    Jawaban:
    Gudang penyangga melaporkan pupuk rusak dengan melampirkan foto dan alasan pupuk rusak. Selanjutnya operator diswil 1 membalas bisa menyetujui bisa meminta review ulang entah itu karena foto kurang jelas atau alasan kurang jelas. Apabila disetujui nantinya pupuk rusak apabila sudah cukup banyakan akan dikirimkan ke gudang pusat menggunakan kendaraan gudang penyangga. Di pusat nanti pupuk rusak akan didaur ulang.

14. Data apa saja yang sekiranya diperlukan dalam pengiriman pupuk rusak ke pusat?
    Jawaban:
    Nama supir, plat nomor, dengan muatan pupuk rusaknya apa saja.

Gresik, 10 Oktober 2018

Staff Distribusi Wilayah 1	Staff Distribusi Wilayah 1
PT Petrokimia Gresik	PT Petrokimia Gresik
Rachmad Edy Rafianto	Edwyk Sony Udaieby

LAMPIRAN C DATA GUDANG PENYANGGA PROVINSI JAWA TIMUR PT PETROKIMIA GRESIK

Nama Gudang	PROPINSI	Pengelola	Telepon	Kepala Gudang	Telepon	Kapasitas (Ton)
BANGKALAN - SOCAH	JAWA TIMUR	PT Bhanda Ghara Reksa	Ds. Kelehan	Agung Firmansyah	081703391992	1700
BANYUWANGI 1 - SINGOJURUH	JAWA TIMUR	PT Graha Sarana Gresik	Desa Singorujuh	Andri Kurniawan	081216047070	3750
BANYUWANGI 4 - ROGOJAMPI	JAWA TIMUR	PT WRPP	Ds Singojuruh	Sugeng Pramono	08123043337	5000
BANYUWANGI 5 - KALIPURO	JAWA TIMUR	PT Graha Sarana Gresik	106 Jl. Argopuro Klatak	Virdiantoro.s.w	081335494595	2700
BLITAR 1 - TALUN	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Kendalrejo	Beny TS	081357769091	1750
BLITAR 2 - WLINGI	JAWA TIMUR	PT Gresik Cipta Sejahtera	Ds Lengkungan Burit Babatan	Tejo Wahyono	081331335344	0
BOJONEGORO - TEMAYANG	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Bojonegoro, Nganjuk KM 23	Totok Irianto	081230473368	2000
BOJONEGORO 1 - BAURENO	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Gajah	Mukrim Harjono	081341178579	5000
BOJONEGORO 2 - SUMBEREJO	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Sumberejo	K Munawar	081359620175	5000
BOJONEGORO 3 - PADANGAN	JAWA TIMUR	PT Aneka Jasa Grhadika	Desa Bandar Rejo	Bagoes Wisda M P	085647656667	1000
BOJONEGORO 4 - KALITIDU	JAWA TIMUR	PT Aneka Jasa Grhadika	Ds Talok Kec Kalitidu (Depan E	Andi Ahmad	085337008084	6000
BONDOWOSO - TANGSIL	JAWA TIMUR	PT Bhanda Ghara Reksa	Ex.Gudang Puskud, Tangsil	Bambang Sumbodo	081334542271	2750
DC BANYUWANGI	JAWA TIMUR	PT Graha Sarana Gresik	Jl Lingkar Ketapang Akaba Bany	Nurchoiri	082143255447	18500
DC GRESIK KIG I	JAWA TIMUR	PT Petrokopinpo Cipta Selaras	3 Jl Tridarma Komplek KIG Gres	Ferri	081235167938	11104
DC GRESIK KIG Q	JAWA TIMUR	PT Graha Sarana Gresik	Jl Tridarma Komplek KIG Gresik	Heru	081217187148	24400
GRESIK - TRI DHARMA	JAWA TIMUR	PT Graha Sarana Gresik	Jl Darmo Sugondo Karang Kiring	Antonius Sudarto	081332207968	0
JEMBER 1 - RAMBIPUJI	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Stasiun Rambipuji	Djati Trismono	081335777797	2500
JEMBER 2 - AMBULU	JAWA TIMUR	PT Aneka Jasa Grhadika	99 Jl. Ambulu	Nurkhan	081216008013	4000
JEMBER 3 - PUGER	JAWA TIMUR	PT Aneka Jasa Grhadika	Ds Jambe Arum	Hery Murjito	085330882737	5000
JOMBANG - GATSU	JAWA TIMUR	PT Bhanda Ghara Reksa	18 Jl. Gatot Subroto	Sugeng Riyanto	081233969281	5000
KEDIRI 1 - KAJEN	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Sambirobyong	Haris Prayudi	085257774115	3000
KEDIRI 2 - RINGINREJO	JAWA TIMUR	PT Aneka Jasa Grhadika	Ds Srikaton	Didit Lesmono	081332864295	0
KEDIRI 3 GURAH	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Kediri - Kertosono	Herwanto	081252244406	6900
LAMONGAN 1	JAWA TIMUR	PT Graha Sarana Gresik	74 Jl. Jaksa Agung Suprapto	Dimas Yoga Saputra	082140659055	5000
LAMONGAN 3 - PUCUK	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Babat - Lamongan Ds K	SUGIANTO	081230576554	3200
LUMAJANG 2 - TEMPEH	JAWA TIMUR	PT Bhanda Ghara Reksa	Ds Besuk	Arief Anwar H. C	082330400900	3500
LUMAJANG 3 - ROWOKANGKUNG	JAWA TIMUR	PT Aneka Jasa Grhadika	Ds Nogosari	Farid Romawan	082334823888	7000
MADIUN - KALIGUNTING	JAWA TIMUR	PT DAT	Desa Kaligunting, Kec. Caruban	Kusnandar	081357311806	3000

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

MADIUN - SUMBERBENING	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Sumberbening	Hadi Sukmono	081216 45518	5000
MAGETAN - MAOSPATI	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Sugih Waras	Lugito	081259 18078	7000
MALANG - PAKISAJI	JAWA TIMUR	PT Graha Sarana Gresik	120 Jl. Kebun Agung, Ds. Kebun	Pranoto	082143 222523	6345
MALANG 1 - BURING	JAWA TIMUR	PT Bhanda Ghara Reksa	6 Jl Majen Sungkono	Faried Fatriadi	081252 928900	2000
MALANG 5 - BAKALAN	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Mboro Sentulan	Hendrop Liberto Harianja	081332 226849	7300
MOJOKERTO - SOOKO	JAWA TIMUR	PT Bhanda Ghara Reksa	Jl. R.A. Basuni	M. Khanif	082133 593914	10000
MOJOKERTO 2 - TROWULAN	JAWA TIMUR	PT Graha Sarana Gresik	Ds Jatipasar	Ispriagus	081332 841520	13000
MOJOKERTO 3 - KEMLAGI	JAWA TIMUR	PT Graha Sarana Gresik	Ds Kemlagi	Galih Prastowo	081331 338266	2400
NGANJUK 2 - LOCERET	JAWA TIMUR	PT Aneka Jasa Grhadika	Ds Loceret	Fritzsman Robinson	082134 021472	7000
NGANJUK 3 - LOCERET	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Loceret Km 1 Ds Tempe	Aris Wijanarko	081216 085100	5850
NGAWI - GENENG	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Geneng Km. 10	Syaifuddin Zuhri	081572 99404	0
NGAWI 2 - KARANGJATI	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Kedungmiri	Aji Ismu H	085257 410252	10750
NGAWI 3 - PARON	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Ngawi-Maospati Km 4 Ds Tem	Feri Panjaitan	081259 671858	10000
PACITAN - MENADI	JAWA TIMUR	PT Bhanda Ghara Reksa	7 Jl. Adisucipto (Jl. Raya Saw	Rubiyanto	081556 426835	1750
PAMEKASAN - LARANGAN	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Nyalaran Ds Blumbunga	Tedi Sampurna	081328 304034	4000
PAMEKASAN 2 - GALIS	JAWA TIMUR	PT Graha Sarana Gresik	101 Jl. Raya Galis Ds. Pagendi	Ahpan Mujianto	081330 590206	1822
PASURUAN 2 - KEJAYAN	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Kejayan Pasuruan, Kan	Yoyok Suryadi	081230 181443	2000
PASURUAN 3 - PELABUHAN	JAWA TIMUR	PT Graha Sarana Gresik	1A Jl Letjen S Parman Pasuruan	Bambang Tri Aries S	081334 256144	3250
PONOROGO 1 - COKRO	JAWA TIMUR	PT Bhanda Ghara Reksa	Jl. Raya Ponorogo - Madiun Km.	Supriyanto L	082143 346234	2700
PONOROGO 2 - BALONG	JAWA TIMUR	PT Graha Sarana Gresik	Jl Ponorogo Pacitan Km 13 Ds S	Ardy Purwanto	081232 79977	3500
PROBOLINGGO 1 - PAITON	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Surabaya - Paiton Km.	Maryono	081331 555359	2300
PROBOLINGGO 2 - WONOMERTO	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl Raya Sempuh Gembol	Moch.Suyuti	081939 597129	2500
SAMPANG - TORJUN	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Torjun	Budi Santoso	082142 659629	1000
SIDOARJO 2 - BYPASS	JAWA TIMUR	CV Mulia P	Jl. Raya By-Pas Km 12	Gatot Purwanto	856483 84983	5000
SITUBONDO 1 - ARJASA	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Situbondo-Asembagus K	Deny Apriyanto	081222 118169	5500
SUMENEP - SARONGGI	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Tanah Merah	Moh. Ishamuddin	085257 799939	1500
SUMENEP 2 - BLUTO	JAWA TIMUR	PT Graha Sarana Gresik	Ds. Pakandangan Tengah	Lutfi Utomo	082316 287849	2194
TRENGGALEK	JAWA TIMUR	PT Bhanda Ghara Reksa	Desa Kendalrejo	Suyadi	081136 3476	1500
TUBAN 2 - PALANG	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl Raya Palang Tuban	Khasan Kholiq	082332 841107	5000
TUBAN 3 JENU - BEJI	JAWA TIMUR	PT Aneka Jasa Grhadika	Jl. Raya Jenu, Desa Sugihwaras	Masrukhin	081331 702276	8000
TULUNGAGUNG 1 - NGUNUT	JAWA TIMUR	PT Bhanda Ghara Reksa	Jl. Raya - Ngunut Km.2	Yuni Swinarko	081234 176778	2000
TULUNGAGUNG 2 - NGANTRU	JAWA TIMUR	PT Graha Sarana Gresik	Jl. Raya Ngantru - Tulungagung	Misdianto	082141 198774	5000

repository.ub.ac.id
UNIVERSITAS BRAWIJAYA

LAMPIRAN D FORM HITUNG STOK OPNAME

Perhitungan fisik Stock Opname Pupuk

Tanggal :

Jam :

Lokasi :

Jenis Pupuk :

Kemasan :

No.	Area	Hasil Penghitungan Kunci				Jumlah		+/		Total	
		Kunci	Panjang	Lebar	Tinggi	Bag	Ton	(Bag)	(Ton)	(Bag)	(Ton)
		1	2	3	4	5=(1x2x3x4)	6	7	8	9=(5+7)	10=(6+8)
	JUMLAH										

Kepala Gudang Penyangga	Staft Dept. Distribusi Wil-I
Gede Apriana	Rachmad Edy Rafianto Bryan