

**PENGEMBANGAN SISTEM INFORMASI MANAJEMEN
KEUANGAN PETERNAKAN KAMBING
(STUDI KASUS: YOGA'S FARM KABUPATEN TULUNGAGUNG)**

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Rizki Indra Fanani
NIM: 135150401111104



**PROGRAM STUDI SISTEM INFORMASI
JURUSAN SISTEM INFORMASI
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2019**

PENGESAHAN

Pengembangan Sistem Informasi Manajemen Keuangan Peternakan Kambing
(Studi Kasus: Yoga's Farm Kabupaten Tulungagung)

SKRIPSI

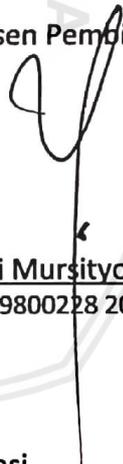
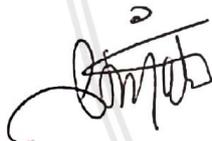
Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Rizki Indra Fanani
NIM: 135150401111104

Skripsi ini telah diuji dan dinyatakan lulus pada
3 Januari 2019
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II



Ismiarta Aknuranda, S.T, M.Sc, Ph.D
NIK: 201006 740719 1 001

Yusi Tyroni Mursityo, S.Kom., M.AB.
NIP: 19800228 200604 1 001

Mengetahui
Ketua Jurusan Sistem Informasi



Dr. Eng. Herman Tolle, S.T., M.T
NIP: 19740823 200012 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini yang berjudul "PENGEMBANGAN SISTEM INFORMASI MANAJEMEN KEUANGAN PETERNAKAN KAMBING (STUDI KASUS: YOGA'S FARM KABUPATEN TULUNGAGUNG)" tidak terdapat karya ilmiah yang pernah diajukan orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya ataupun pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Demikian pernyataan ini saya buat dengan sesungguhnya dan apabila ternyata di dalam naskah skripsi ini dapat dibuktikan terdapat unsur plagiasi, maka saya bersedia skripsi ini diundurkan dan gelar akademik yang telah saya peroleh dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No.20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 3 Januari 2019



Rizki Indra Fanani

Nim: 135150401111104

KATA PENGANTAR

Segala puji bagi Allah SWT atas limpahan rahmat yang diberikan sehingga penulis dapat menyelesaikan skripsi berjudul “Pengembangan Sistem Informasi Manajemen Keuangan Peternakan Kambing (Studi Kasus: Yoga’s Farm Kabupaten Tulungagung)”.

Dari aspek akademis, penelitian ini mengajarkan banyak ilmu yang belum saya dapatkan di kelas perkuliahan. Selain itu, penelitian ini juga mengajarkan bagaimana penulis dapat berkomunikasi dengan banyak orang, menyamakan persepsi yang saya miliki dengan pemangku kepentingan dan dosen pembimbing. Proses penelitian yang dilakukan juga mengajarkan untuk menjadi pribadi yang lebih sabar, menurunkan ego, dan senantiasa berbagi hal-hal yang bermanfaat yang berkaitan dengan kebutuhan pembelajaran. Penelitian ini tidak dapat diselesaikan tanpa dukungan dan bantuan dari orang-orang terdekat, sehingga saya ingin menyampaikan rasa terima kasih kepada:

1. Bapak Ismiarta Aknuranda, S.T, M.Sc, Ph.D, selaku pembimbing satu yang telah memberikan kesempatan untuk bergabung dalam kelompok riset serta menjadi pembimbing yang sabar, solutif, dan memberikan contoh sebagai pribadi yang profesional dalam memberikan bimbingan kepada mahasiswa.
2. Bapak Himawat Aryadita, S.T, M.Sc dan Bapak Yusi Tyroni Mursityo, S.Kom., M.AB., selaku pembimbing dua yang mengajarkan ketegasan dan kedisiplinan dalam menyelesaikan kewajiban. Memberikan contoh sebagai pribadi yang sabar, solutif, dan profesional dalam memberikan bimbingan kepada mahasiswa.
3. Bapak Yoga Wahyu Pratama, S.Pd., selaku pemilik peternakan yang telah memberikan ijin untuk melakukan penelitian di peternakan beliau.
4. Ibu saya, Umi Badriah yang selalu memberikan dukungan doa, moril, materiil, mengajarkan ketegasan, kedisiplinan dalam menyelesaikan tugas, dan senantiasa memberikan contoh yang baik untuk menjadi pribadi yang sabar dan pekerja keras. Ayah saya, Sujoto yang sangat saya yakini senantiasa memberikan dukungan doa dan senantiasa menjadi pribadi yang bersahabat dengan saya.
5. Pendamping tercinta Dinda Ayu Rudyana Putri yang selalu memberi dukungan do’a, nasehat, dan motivasi, serta selalu memberi semangat ketika proses pengerjaan skripsi.
6. Saudara Arik Harianto, S.T., Redi Sucipto, S.T., Ja’far Sodiq, S.T., dan drh. Nawawi yang telah memberikan motivasi dalam pengerjaan skripsi.
7. Teman-teman Sistem Informasi tercinta yang selalu memberikan semangat, dorongan, dan bantuan pikiran.
8. Seluruh pihak yang telah membantu kelancaran penulisan skripsi yang tidak dapat disebutkan satu persatu.
9. Penulis menyadari bahwa dalam penyusunan dan penulisan naskah skripsi ini masih banyak kekurangan baik format laporan maupun isinya. Untuk itu penulis sangat mengharapkan kritik dan saran yang membangun. Semoga



naskah skripsi ini dapat bermanfaat baik bagi pembaca maupun penulis, lebih-lebih bisa dikembangkan menjadi penelitian yang baru, amin.

Malang, 3 Januari 2019

Rizki Indra Fanani

Kang.indroo@gmail.com



ABSTRAK

Rizki Indra Fanani, Pengembangan Sistem Informasi Manajemen Keuangan Peternakan Kambing (Studi Kasus: Yoga's Farm Kabupaten Tulungagung)

Pembimbing: Ismiarta Aknuranda, S.T., M.Sc, Ph.D dan Yusi Tyroni Mursityo, S.Kom., M.AB

Kebutuhan akan informasi pada saat ini merupakan hal yang sangat penting dalam berbagai aspek kehidupan. Yoga's Farm adalah sebuah peternakan yang menjalankan proses penggemukan kambing, penjualan kambing, penjualan obat ternak, dan penjualan berbagai macam jenis pakan ternak. Peternakan ini memiliki cukup keuntungan namun semua pelaporan keuangan masih tidak tertata dengan rapi. Pelaporan keuangan hanya dilakukan dengan manual, hal ini menyebabkan banyaknya keuangan yang tidak terdefinisi kegunaannya dan menimbulkan keuntungan dari peternakan tidak bisa teridentifikasi. Dari masalah yang muncul maka dikembangkan sebuah sistem informasi yang akan membantu dalam pengolahan data keuangan yang terstruktur. Sistem pelaporan keuangan juga bisa dicetak dan menampilkan keuntungan dari peternakan kambing. Metode yang digunakan dalam pengembangan ini adalah waterfall.

Sistem yang dikembangkan dengan metode *waterfall* akan memiliki hasil yang baik karena pengembangan sistem sangat terorganisir dengan baik. Tahapan yang ada pada waterfall adalah *Requirement Defintion, System and Software Design, Implementation and Unit Testing, Integration and System testing, dan Operation and Maintenance*. Sedangkan kerangka kerja yang digunakan dalam pengembangan sistem informasi ini adalah laravel. Laravel adalah suatu framework PHP yang dirilis dibawah lisensi MIT, dibangun dngan konsep MVC (model view controller). Pengujian untuk fungsional sistem adalah pengujian validasi yang digunakan untuk pengujian semua fungsi yang ada pada sistem, hasil dari pengujian validasi menyatakan bahwa sistem 100% valid yang menandakan berfungsi dengan baik dan sesuai dengan ekspektasi pengguna. Pengujian yang digunakan untuk memeriksa nonfungsional dari sistem adalah pengujian kompatibilitas, pengujian kompatibilitas yang dilakukan berfokus pada peramban yang digunakan oleh sistem, hasil menyatakan bahwa sistem bisa dijalankan pada 9 browser yang berbeda. Namun, pada IE (*Internet Explorer*) versi 11 terdapat *critical issue, 9 major issue* dari 16 versi peramban dan 5 *minor issue* dari 10 versi peramban.

Kata kunci: Sistem Informasi Manajemen Keuangan, Waterfall, Laravel, Pengujian Validasi, Kompatibilitas Peramban

ABSTRACT

Rizki Indra Fanani, Development of Goat Livestock Financial Management Information System (Case Study: Yoga's Farm, Tulungagung Regency)

Supervisor: Ismiarta Aknuranda, S.T., M.Sc, Ph.D and Yusi Tyrone Mursityo, S.Kom., M.AB

The need for information at this time is very important in various aspects of life. Yoga's Farm is a farm that runs the process of fattening goats, selling goats, selling cattle medicine, and selling various types of animal feed. This farm has enough benefits but all financial reporting is still not neatly arranged. Financial reporting is only done manually, this results in a large number of finances that are not defined as being useful and which can lead to profits from livestock cannot be identified. From the problems that arise, an information system is developed that will assist in processing structured financial data. Financial reporting systems can also be printed and display the benefits of goat farming. The method used in this development is waterfall.

Systems developed using the waterfall method will have good results because the development of the system is very well organized. The stages in the waterfall are Requirement Definition, System and Software Design, Implementation and Unit Testing, Integration and System testing, and Operation and Maintenance. Whereas the framework used in developing this information system is Laravel. Laravel is a PHP framework released under the MIT license, built on the MVC concept (model view controller). Functional testing system is a validation test used to test all functions that exist on the system, the results of validation testing state that the system is 100% valid which indicates that it is functioning properly and in accordance with user expectations. The test used to check the nonfunctionality of the system is compatibility testing, compatibility testing carried out focuses on the browser used by the system, the results state that the system can be run on 9 different browsers. However, in IE (Internet Explorer) version 11 there are critical issues, 9 major issues from 16 browser versions and 5 minor issues from 10 browser versions.

Keyword: Financial Management Information System, Waterfall, Laravel, Validation Testing, Browser Compatibility

DAFTAR ISI

LEMBAR PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR.....	iv
ABSTRAK.....	vi
ABSTRACT	vii
DAFTAR ISI.....	viii
DAFTAR TABEL.....	xii
DAFTAR GAMBAR.....	xiv
DAFTAR LAMPIRAN	xvi
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 Tentang Peternakan Yoga's Farm	6
2.3 Pengembangan dengan Pendekatan Model <i>Waterfall</i>	6
2.4 Sistem Informasi	8
2.5 Manajemen Keuangan.....	8
2.6 <i>Bussiness Process Managemenet Notation (BPMN)</i>	8
2.6.1 <i>Flow Object</i>	9
2.6.2 <i>Connections</i>	10
2.6.3 <i>Swimlanes</i>	11
2.6.4 <i>Artifacts</i>	12
2.6.5 <i>Data</i>	12
2.7 <i>Unified Modelling Language (UML)</i>	13
2.7.1 <i>Use Case Diagram</i>	13

2.7.2	<i>Sequence Diagram</i>	14
2.7.3	<i>Class Diagram</i>	15
2.8	<i>Laravel</i>	15
2.9	Pengujian Perangkat Lunak	17
2.9.1	<i>Validation Testing</i>	17
2.9.2	<i>Browser Compatibility Testing</i>	17
BAB 3	METODOLOGI	19
3.1	Studi Literatur	19
3.2	Penggalian dan Analisis Kebutuhan Pengguna	19
3.3	Perancangan Sistem	20
3.4	Implementasi Sistem	20
3.5	Pengujian	20
BAB 4	Penggalian dan Analisis Kebutuhan sistem	21
4.1	Proses Bisnis AS-IS	21
4.2	Proses Bisnis TO-BE	22
4.3	Analisis Persyaratan	28
4.3.1	Analisis Masalah	28
4.3.2	Identifikasi Pengguna	29
4.3.3	Identifikasi Fitur	30
4.3.4	Persyaratan Fungsional	31
4.3.5	Persyaratan Nonfungsional	32
4.4	Pemodelan <i>Use Case</i>	33
4.5	<i>Use Case Specification</i>	34
4.5.1	<i>Use Case Specification</i> Kelola Produk Jual	34
4.5.2	<i>Use Case Specification</i> Kelola Produk Beli	37
4.5.3	<i>Use Case Specification</i> Kelola Data Maintenance	39
4.5.4	<i>Use Case Specification</i> Kelola Data Pemasukan	41
4.5.5	<i>Use Case Specification</i> Kelola Data Pengeluaran	44
4.5.6	<i>Use Case Specification</i> Melihat Laporan	46
4.5.7	<i>Use Case Specification</i> Mencetak Laporan	47
BAB 5	PERANCANGAN	49
5.1	Perancangan Arsitektur	49



5.2	Pemodelan Interaksi Objek.....	49
5.2.2	Class Diagram	55
5.3	Perancangan Database	56
5.4	Perancangan Antarmuka	61
5.4.1	Antarmuka Login	61
5.4.2	Antarmuka Halaman Utama	61
5.4.3	Antarmuka Produk Jual	62
5.4.4	Antarmuka Produk Beli	63
5.4.5	Antarmuka Maintenance	64
5.4.6	Antarmuka Pemasukan	65
5.4.7	Antarmuka Pengeluaran	66
5.4.8	Antarmuka Laporan	67
BAB 6	IMPLEMENTASI	68
6.1	Lingkungan Sistem	68
6.1.1	Lingkungan Perangkat Keras	68
6.1.2	Lingkungan Perangkat Lunak	68
6.2	Implementasi Program	69
6.3	Implementasi Antarmuka	70
6.3.1	Halaman Login.....	70
6.3.2	Halaman <i>Dashboard</i>	71
6.3.3	Halaman Kelola Produk Jual.....	71
6.3.4	Halaman Kelola Produk Beli	72
6.3.5	Halaman Kelola Data Maintenance	72
6.3.6	Halaman Kelola Data Pemasukkan	73
6.3.7	Halaman Kelola Data Pengeluaran.....	74
6.3.8	Halaman Laporan	74
BAB 7	PENGUJIAN	76
7.1	Pengujian Fungsional	76
7.1.1	<i>Validation Testing</i> Kelola Produk Jual	76
7.1.2	<i>Validation Testing</i> Kelola Produk Beli	77
7.1.3	<i>Validation Testing</i> Kelola Data Maintenance.....	79
7.1.4	<i>Validation Testing</i> Kelola Data Pemasukkan.....	80



7.1.5 <i>Validation Testing</i> Kelola Data Pengeluaran.....	81
7.1.6 <i>Validation Testing</i> Melihat Laporan	82
7.1.7 <i>Validation Testing</i> Mencetak Laporan	82
7.2 Pengujian Nonfungsional	83
7.2.1 Browser Compatibility Testing	83
BAB 8 PENUTUP	85
8.1 Kesimpulan.....	85
8.2 Saran	86
DAFTAR PUSTAKA.....	87
LAMPIRAN A HASIL WAWANCARA.....	89
A.1 Hasil Wawancara dengan Pemilik Peternakan Kambing.....	89



DAFTAR TABEL

Tabel 2.1 Notasi <i>Events</i>	9
Tabel 2.2 Notasi <i>Activity</i>	9
Tabel 2.3 Notasi <i>Gateways</i>	10
Tabel 2.4 Notasi <i>Connections</i>	11
Tabel 2.5 Notasi <i>Swimlanes</i>	11
Tabel 2.6 Notasi <i>Artifacts</i>	12
Tabel 2.7 Notasi <i>Data</i>	12
Tabel 4.2 Analisis Masalah	28
Tabel 4.3 Identifikasi Pengguna	29
Tabel 4.4 Fitur	30
Tabel 4.5 Persyaratan Fungsional	31
Tabel 4.6 Persyaratan Nonfungsional	33
Tabel 4.7 Hubungan <i>Use Case</i> dengan Fitur	34
Tabel 4.8 <i>Use Case Specification</i> Kelola Produk Jual	34
Tabel 4.9 <i>Use Case Specification</i> Kelola Produk Beli.....	37
Tabel 4.10 <i>Use Case Specification</i> Kelola Data Maintenance	39
Tabel 4.11 <i>Use Case Specification</i> Kelola Data Pemasukkan	41
Tabel 4.12 <i>Use Case Specification</i> Kelola Data Pengeluaran	44
Tabel 4.13 <i>Use Case Specification</i> Melihat Laporan	46
Tabel 4.14 <i>Use Case Specification</i> Mencetak Laporan.....	47
Tabel 5.1 Struktur Tabel Produk.....	57
Tabel 5.2 Struktur Tabel produk_jual.....	57
Tabel 5.3 Struktur Tabel Produk_belis.....	58
Tabel 5.4 Struktur Tabel Pemasukkans	58
Tabel 5.5 Struktur Tabel Pengeluarans	59
Tabel 5.6 Struktur Tabel <i>Maintenances</i>	59
Tabel 5.7 Struktur Tabel Saldos.....	60
Tabel 5.8 Struktur Tabek <i>Password_reset</i>	60
Tabel 5.9 Struktur Tabel <i>Users</i>	60
Tabel 5.10 Struktur Tabel <i>Migrations</i>	61



Tabel 6.1 Lingkungan Perangkat Keras	68
Tabel 6.2 Lingkungan Perangkat Lunak.....	68
Tabel 6.3 Kode Program Simpan Produk Beli	69
Tabel 6.4 Kode Program Lihat Laporan.....	69
Tabel 6.5 Kode Program Hapus Data Pemasukkan.....	70
Tabel 7.1 Rencana dan Kasus Uji <i>Validation Testing</i> Kelola Produk Jual.....	76
Tabel 7.2 Rencana dan Kasus Uji <i>Validation Testing</i> Kelola Produk Beli.....	77
Tabel 7.3 Rencana dan Kasus Uji <i>Validation Testing</i> Kelola Data Maintenance...	79
Tabel 7.4 Rencana dan Kasus Uji <i>Validation Testing</i> Data Pemasukkan	80
Tabel 7.5 Rencana dan Kasus Uji <i>Validation Testing</i> Kelola Data Pengeluaran....	81
Tabel 7.6 Rencana dan Kasus Uji <i>Validation Testing</i> Melihat Laporan.....	82
Tabel 7.7 <i>Validation Testing</i> Mencetak Laporan	82
Tabel 7.8 <i>Browser</i> Yang Digunakan Pada <i>Browser Compatibility Testing</i>	83



DAFTAR GAMBAR

Gambar 2.1 Model <i>Waterfall</i>	7
Gambar 2.2 <i>Use Case Diagram</i>	14
Gambar 2.3 <i>Sequence Diagram</i>	14
Gambar 2.4 <i>Class Diagram</i>	15
Gambar 2.5 <i>Validation Testing</i>	17
Gambar 3.1 Kerangka Kerja Penelitian	19
Gambar 4.1 Proses Bisnis AS-IS Pembelian.....	24
Gambar 4.2 Proses Bisnis AS-IS Penjualan.....	25
Gambar 4.3 Proses Bisnis TO-BE Pembelian	26
Gambar 4.4 Proses Bisnis TO-BE Penjualan	27
Gambar 4.5 Kodifikasi Persyaratan Fungsional.....	31
Gambar 4.6 Use Case Sistem Informasi Manajemen Keuangan Yoga's Farm	33
Gambar 5.1 Perancangan Arsitektur Sistem Informasi Manajemen Keuangan ...	49
Gambar 5.2 <i>Sequence Diagram</i> Kelola Produk Jual	50
Gambar 5.3 <i>Sequence Diagram</i> Kelola Produk Beli	52
Gambar 5.4 <i>Sequence Diagram</i> Kelola Data Maintenance.....	52
Gambar 5.5 <i>Sequence Diagram</i> Kelola Data Pemasukkan.....	53
Gambar 5.6 <i>Sequence Diagram</i> Melihat Laporan.....	55
Gambar 5.7 <i>Sequence Diagram</i> Kelola Data Pengeluaran.....	55
Gambar 5.8 <i>Class Diagram</i> Sistem Informasi Manajemen Keuangan.....	56
Gambar 5.9 Rancangan <i>Database</i>	57
Gambar 5.10 Antarmuka Login	61
Gambar 5.11 Antarmuka Halaman Utama	62
Gambar 5.12 Antarmuka Tambah Data Produk Jual	62
Gambar 5.13 Antarmuka Rekap Penjualan.....	63
Gambar 5.14 Antarmuka Tambah Data Produk Beli.....	63
Gambar 5.15 Antarmuka Rekap Pembelian.....	64
Gambar 5.16 Antarmuka Tambah Data Maintenance.....	64
Gambar 5.17 Antarmuka Rekap Data Maintenance	65
Gambar 5.18 Antarmuka Tambah Data Pemasukkan.....	65



Gambar 5.19 Antarmuka Rekap Pemasukkan	66
Gambar 5.20 Antarmuka Tambah Data Pengeluaran	66
Gambar 5.21 Antarmuka Rekap Data Pengeluaran	67
Gambar 5.22 Antarmuka Laporan.....	67
Gambar 6.1 Halaman Login.....	70
Gambar 6.2 Halaman Dashboard	71
Gambar 6.3 Halaman Kelola Produk Jual (Rekap Penjualan).....	71
Gambar 6.4 Halaman Kelola Produk Jual (Tambah Data)	72
Gambar 6.5 Halaman Kelola Produk Beli (Rekap Pembelian).....	72
Gambar 6.6 Halaman Kelola Produk Beli (Tambah Data)	72
Gambar 6.7 Halaman Kelola Data Maintenance (Rekap Maintenance).....	73
Gambar 6.8 Halaman Kelola Data Maintenance (Tambah Data)	73
Gambar 6.9 Halaman Kelola Data Pemasukkan (Rekap Pemasukkan).....	73
Gambar 6.10 Halaman Kelola Data Pemasukkan (Tambah Data)	74
Gambar 6.11 Halaman Kelola Data Pengeluaran (Rekap Pengeluaran)	74
Gambar 6.12 Halaman Kelola Data Pengeluaran (Tambah Data).....	74
Gambar 6.13 Halaman Laporan	75
Gambar 7.1 Kodifikasi <i>Validation Testing</i>	76
Gambar 7.2 <i>Browser Compatibility Testing</i> dengan <i>SortSite</i>	84

DAFTAR LAMPIRAN

LAMPIRAN A HASIL WAWANCARA.....	89
A.1 Hasil Wawancara dengan Pemilik Peternakan Kambing.....	89



BAB 1 PENDAHULUAN

1.1 Latar belakang

Kebutuhan akan informasi merupakan hal yang penting saat ini, dalam berbagai aspek kehidupan. Informasi merupakan data yang sangat penting yang dapat memberikan pengetahuan yang berguna atau bermanfaat bagi penerimanya (George, 2011). Perkembangan teknologi yang sangat pesat membuat penyebaran informasi semakin cepat dan kompleks. Hal ini menyebabkan banyak produsen untuk perangkat teknologi informasi demi melayani kebutuhan konsumen. Kebutuhan akan teknologi informasi ini seolah menjadikan teknologi informasi menjadi kebutuhan pokok dan mempunyai peran penting pada kehidupan masyarakat modern. Teknologi informasi menawarkan banyak kemudahan pada kehidupan masyarakat yang menjadikannya semakin banyak yang menggunakannya (Arif, 2015).

Sistem Informasi merupakan sistem yang mempunyai kemampuan untuk mengumpulkan informasi dari semua sumber dan menggunakan berbagai media untuk memberikan informasi (Mc Leod, 2001). Oleh karena itu, dapat disimpulkan bahwa sistem informasi merupakan sistem yang dapat mengumpulkan data serta mengolah data tersebut menjadi sebuah informasi yang dapat digunakan untuk mendapat pengetahuan atau menentukan keputusan. Dengan menggunakan sistem informasi kinerja yang dilakukan pada suatu perusahaan akan efektif dan efisien. Jenis dari sistem informasi juga bermacam-macam seperti sistem informasi akademik, sistem informasi penjualan, sistem informasi keuangan, sistem informasi kepegawaian, dan lainnya.

Peternakan adalah segala urusan yang berkaitan dengan sumber daya fisik, benih, bibit dan/atau bakalan, pakan, alat dan mesin peternakan, budi daya ternak, panen, pascapanen, pengolahan, pemasaran, dan pengusahanya (Undang-Undang, 2009). Yoga's Farm adalah sebuah peternakan yang menjalankan proses penggemukan kambing, penjualan kambing, penjualan obat ternak, dan penjualan berbagai macam jenis pakan ternak. Peternakan ini merupakan sebuah peternakan yang besar karena memiliki kapasitas sekitar 2000 ekor kambing dari kandang dan gudang yang cukup luas. Sebagian besar proses penjualan pada peternakan ini adalah pemesanan, namun juga melayani pembeli yang datang langsung ke peternakan. Penjualan kambing dilakukan dengan cara timbang hidup, jadi harga kambing ditentukan dari berat badan.

Meskipun peternakan ini memiliki omset yang cukup besar, namun pemilik peternakan masih kesulitan mendeteksi keuntungan. Pelaporan keuangan masih dilakukan dengan sederhana, penghitungan keuntungan dengan melihat nota transaksi yang dilakukan pada peternakan dan dihitung secara manual. Laporan keuangan dicatat dalam sebuah buku yang isinya tidak tertata dengan rapi. Semua produk yang dihasilkan dan produk yang dibeli oleh peternakan dicatat pada satu buku yang sama. Hal ini menyebabkan banyak dana peternakan yang tidak terdefinisi kegunaannya dengan jelas. Oleh karena itu, keuntungan dari peternakan tidak bisa diidentifikasi secara jelas dalam laporan tersebut. Pelaporan keuangan

tidak dilakukan secara intensif yang berakibat laporan data penjualan dan pembelian juga tidak teridentifikasi dengan baik. Dengan adanya masalah tersebut pengembangan sistem informasi merupakan salah satu solusi yang sesuai dengan kondisi yang ada.

Dalam pengembangan sistem informasi terdapat beragam model untuk pengembangan. *Waterfall Model* adalah salah satu model yang bersifat sistematis dalam pengembangan perangkat lunak. Pengembangan perangkat lunak sistem informasi yang baik, *Waterfall Model* memiliki 5 fase yaitu *Requirement Defintion, System and Software Design, Implementation and Unit Testing, Integration and System testing*, dan *Operation and Maintenance* (Sommerville, 2010). Hal pertama yang harus dilakukan dalam model ini adalah mendefinisikan kebutuhan pengguna sistem informasi, setelah mendapatkan kebutuhan dari pengguna dapat dibuat perancangan sistem informasi berdasarkan hasil analisis kebutuhan tersebut. Jika perancangan sudah ditentukan maka dapat dilakukan implementasi yang menghasilkan kode program tertentu yang meyesuaikan dengan hasil perancangan. Setelah perancangan dan imnplementasi selesai perlu dilakukan pengujian untuk mengukur kelayakan perangkat lunak yang telah dikembangkan.

Dari penjelasan masalah di atas untuk meningkatkan kinerja manajemen keuangan dan pelaporan data penjualan serta pembelian peternakan, dibutuhkan sebuah sistem informasi yang digunakan untuk sistem pelaporan dengan menggunakan metode *Waterfall Model*. Penggunaan *waterfall* ini karena masalah yang diperoleh dari narasumber telah dikonfirmasi kebenarannya dan telah terjadi kesepakatan antara peneliti dengan *stakeholder*. Metode *waterfall* sesuai dengan tahapan dalam pengembangan sistem secara sistematis. Sistem yang dikembangkan dengan metode *waterfall* akan memiliki hasil yang baik karena pengembangan sistem sangat terorganisir. Selain itu, *waterfall model* konsisten dengan fase yang dimilikinya dan dokumentasi dihasilkan pada setiap fase. Masalah utamanya adalah partisi yang tidak fleksibel dari proyek menjadi tahap-tahap yang berbeda. Komitmen harus dibuat terlebih dahulu pada tahap awal dalam proses yang membuatnya sulit untuk menanggapi perubahan persyaratan dari *stakeholde*. Pada prinsipnya, *waterfall model* seharusnya digunakan ketika persyaratan kebutuhannya dipahami dengan baik dan tidak mungkin berubah (Sommerville, 2011).

1.2 Rumusan masalah

Berdasarkan latar belakang di atas, dapat dirumuskan masalah penelitian ini yaitu:

1. Bagaimana hasil analisis kebutuhan sistem informasi manajemen keuangan pada Yoga's farm?
2. Bagaimana hasil perancangan sistem informasi manajemen keuangan pada Yoga's farm?
3. Bagaimana hasil implementasi sistem informasi manajemen keuangan pada Yoga's farm?

4. Bagaimana hasil pengujian sistem informasi manajemen keuangan pada Yoga's farm?

1.3 Tujuan

Berikut tujuan penelitian yang telah dilakukan peneliti:

1. Membuat sistem informasi manajemen keuangan berdasarkan analisis kebutuhan pengguna pada peternakan kambing Yoga's Farm.
2. Mengetahui hasil perancangan sistem informasi manajemen keuangan pada Yoga's Farm.
3. Mengembangkan sistem informasi berdasarkan data yang diperoleh dari analisis kebutuhan pengguna dan perancangan sistem.
4. Mengetahui hasil kelayakan sistem yang telah dibuat dengan melakukan pengujian pada sistem informasi manajemen keuangan Yoga's Farm

1.4 Manfaat

Berikut manfaat penelitian yang telah dilakukan peneliti:

1. Bagi Ilmu Pengetahuan
Menambah referensi tentang penggunaan metode pengembangan sistem informasi, prancangan sistem informasi serta analisis dari pengguna yang digunakan untuk pembuatan sistem informasi.
2. Bagi Penulis
Menambah wawasan dan pengalaman penulis tentang analisis data dari pengguna, perancangan, implementasi, dan pengujian sistem informasi
3. Bagi Yoga's Farm
Memiliki sistem informasi manajemen keuangan yang memadai untuk meningkatkan sistem pelaporan peternakan menjadi efektif dan efisien.

1.5 Batasan masalah

Berikut batasan masalah untuk menyelesaikan penelitian yang dilakukan oleh peneliti:

1. Penelitian ini dilaksanakan di peternakan Yoga's Farm Kabupaten Tulungagung
2. Analisis proses bisnis menggunakan *Business Process Management Notation* (BPMN)
3. Perancangan dipenelitian ini dilakukan analisis kebutuhan fungsional dan nonfungsional untuk membuat *Use Case Diagram* dan *Use Case Spesification*.
4. Pengembangan sistem menggunakan pendekatan *Software Developmet Life Cycle* model *Waterfall*.
5. Pengujian sistem informasi menggunakan *Validation Testing* dan *Browser Compatibility Testing*.

1.6 Sistematika pembahasan

Berikut garis besar sistematika pembahasan dalam penelitian ini yang dimulai dari pendahuluan sampai penutup:

BAB 1 PENDAHULUAN

Bab ini menjelaskan tentang latar belakang dari penelitian yang dilakukan, rumusan masalah yang akan diselesaikan peneliti, tujuan dari penelitian, manfaat penelitian, batasan masalah yang digunakan dalam penelitiannya, dan sistematika pembahasan.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisi penjelasan teori yang digunakan peneliti untuk menjalankan penelitian, model yang digunakan dalam penelitian, metode yang digunakan dalam penelitian, atau dari literatur yang berkaitan dengan tema penelitian.

BAB 3 METODOLOGI

Bab ini menjelaskan langkah-langkah dalam melaksanakan penelitian. Mulai dari kegiatan pertama yang dilakukan dalam penelitian sampai penelitian selesai.

BAB 4 PENGGALIAN DAN ANALISIS KEBUTUHAN

Bab ini menguraikan tentang analisis data hasil observasi dengan *stakeholder* yang akan digunakan untuk kebutuhan sistem informasi manajemen keuangan yang terdiri dari kebutuhan fungsional dan nonfungsional.

BAB 5 PERANCANGAN

Bab ini menjelaskan tentang perancangan sistem informasi keuangan berbasis *website* untuk pengelolaan manajemen keuangan Yoga's Farm.

BAB 6 IMPLEMENTASI

Bab ini menguraikan tentang implementasi sistem informasi manajemen keuangan Yoga's Farm

BAB 7 PENGUJIAN

Bab ini menguraikan tentang proses dan hasil pengujian dari sistem informasi manajemen keuangan Yoga's Farm

BAB 8 PENUTUP

Bab ini berisi kesimpulan dari penelitian yang sudah dilaksanakan sesuai dengan masalah yang ada. Serta memaparkan saran yang mungkin bisa dilakukan dalam penelitian selanjutnya.

BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Kajian pustaka pada penelitian ini membahas mengenai penelitian terdahulu yang berkaitan dengan perancangan perangkat lunak dan pengembangan perangkat lunak atau sistem yang menggunakan *Software Development Life Cycle* (SDLC). Penelitian yang pertama mengenai pengembangan sistem informasi akuntansi syariah. Berdasarkan penelitian tersebut pada studi kasus mengalami banyak masalah tentang pelaporan keuangan dan hilangnya data transaksi. Semua pelaporan keuangan masih dilakukan secara manual menggunakan *Microsoft Excel 2017*. Dalam penelitian tersebut pengembangan sistem informasi akuntansi mampu menangani dan memudahkan pengumpulan data serta mempercepat pelaporan keuangan (Ratnawati, 2014)

Yang kedua mengenai perancangan dan pengembangan sistem informasi persediaan barang, harga pokok produk, dan transaksi penjualan pada Son Screen Printing Sidoarjo. Berdasarkan hasil penelitian dari Adytria di perusahaan tersebut terdapat beberapa masalah yaitu belum adanya pencatatan yang lengkap mengenai persediaan barang dan transaksi penjualan, semua kegiatan tersebut masih dilakukan dengan cara manual. Oleh karena itu, perusahaan tidak memiliki sistem pelaporan kegiatan tahunan yang baik. Sistem informasi yang dikembangkan oleh Adytria menghasilkan data persediaan barang, harga pokok produksi, dan transaksi penjualan (Pamungkas, 2017).

Perbandingan antara tiga *platform* model pengembangan perangkat lunak yang dilakukan oleh Adel Alshamrani dan Abdullah Bahattab dengan judul “*A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model*” *Waterfall Model* memiliki beberapa keuntungan yang diantaranya (Alshamrani & Bahattab, 2015) (1) mudah dipahami dan mudah melakukan implementasi; (2) diketahui dan banyak digunakan; (3) desain sistem jelas sebelum dibuat perangkat lunak; (4) penyusunan yang sistematis sehingga mudah untuk melakukan implementasikan; (5) meminimalisir *overhead* perencanaan; (6) dan fase diproses dan diselesaikan satu per satu. Menurut (Alshamrani & Bahattab, 2015) *Waterfall Model* baik digunakan ketika kebutuhan sudah sangat diketahui, jelas, dan tetap. Pada penelitian yang dilakukan peneliti telah didapatkan bahwa kebutuhan dari pengguna sudah sangat jelas sehingga penulis akan menggunakan model *Waterfall* untuk pengembangan sistem informasi.

Pemanfaatan kerangka laravel untuk pengembangan alat perekrutan berbasis web yang dilakukan oleh Vishal V. Parkar dengan judul “*Utilization o Laravel Framework for Development of Web Based Recruitment Tool*” Laravel adalah sebuah kerangka kerja pengembangan web MVC yang ditulis dalam PHP. Laravel dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan berkelanjutan serta untuk meningkatkan pengalaman bekerja dengan menggunakan aplikasi. Efisiensi

pengembangan metode desain web berdasarkan kerangka laravel lebih tinggi dibandingkan dengan metode desain web tradisional berdasarkan kerangka *CodeIgniter*. Jika dibandingkan dengan kerangka kerja tradisional untuk merancang aplikasi web menghasilkan banyak keterbatasan, misal peningkatan waktu pengembangan. Disamping itu, desain web berdasarkan kerangka laravel memiliki skalabilitas, sehingga dapat meningkatkan efisiensi pengembangan (Parker, 2016).

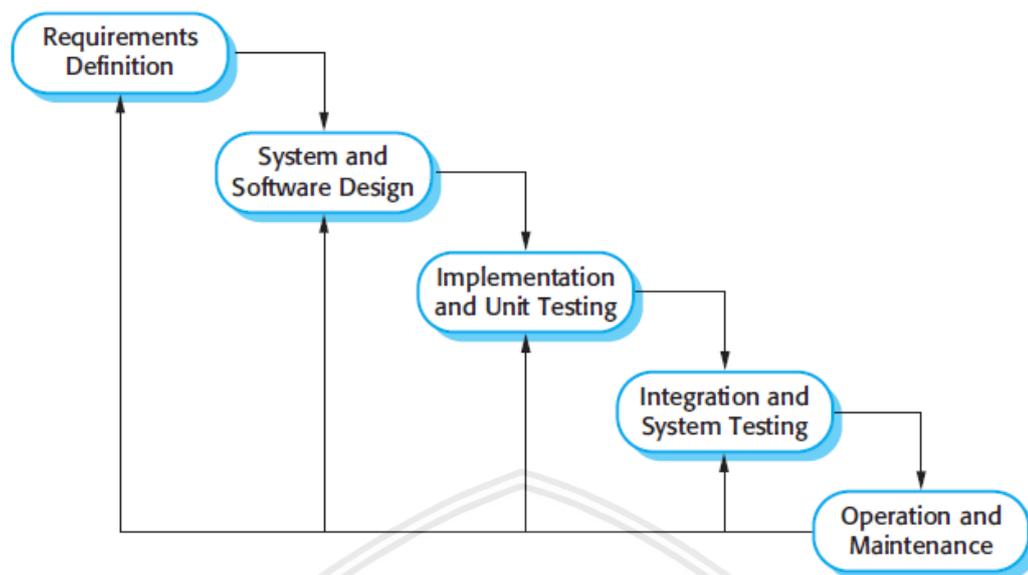
Berdasarkan penelitian-penelitian yang dijabarkan diatas, dapat disimpulkan bahwa pengembangan aplikasi harus sesuai dengan kebutuhan *stakeholder*. Dengan perancangan kebutuhan sistem yang diperoleh dari pengguna dan dijadikan acuan dalam pengembangan sistem informasi.

2.2 Tentang Peternakan Yoga's Farm

Yoga's Farm berdiri sejak tahun 2012, berdasarkan wawancara kepada pemilik peternakan (*Owner*) bernama Yoga Wahyu P, S.Pd. peternakan memiliki 3 karyawan tetap yang langsung dibawah kendali pemilik. Yoga's Farm merupakan sebuah peternakan yang milik perseorangan yang berada di desa Tegalrejo, Rejotangan, Tulungagung. Yoga's farm memiliki sekitar 2000 ekor kambing yang dipelihara untuk diambil hasil dagingnya. Peternakan ini menyediakan kambing dengan ukuran siap konsumsi dan benih, pakan ternak, obat ternak, dan alat peternakan. Penjualan kambing dilakukan dengan cara timbang hidup, jadi harga kambing ditentukan dari berat badan. Yoga's farm juga menjual pakan hasil olahan sendiri dan juga bahan baku dari pakan tersebut. Selain itu, peternakan juga menyediakan beberapa macam obat untuk penyakit ternak, kesehatan ternak, dan kebersihan kandang. Peralatan peternakan kambing juga disediakan untuk peternak baru yang belum mempunyai alat yang memadai, seperti alat suntik hewan, pompa semprot kandang, pencampur pakan ternak dan lainnya. Dengan proses kegiatan yang berjalan setiap hari, peternakan ini mempunyai banyak suplier dan beberapa pelanggan. Banyak peternak lain yang membeli pakan dan obat ternak di Yoga's Farm. Selain membeli produk yang dihasilkan Yoga's farm, sistem kemitraan juga dijalankan. Sistem kemitraan merupakan bentuk kerjasama antar peternak besar dan peternak kecil dengan sistem bagi hasil.

2.3 Pengembangan dengan Pendekatan Model *Waterfall*

Pengembangan perangkat lunak sistem informasi yang baik harus dilakukan dengan teliti dan terorganisasi, *Waterfall Model* merupakan salah satu metode pengembangan perangkat lunak yang terorganisasi dengan baik. *Waterfall Model* memiliki 5 fase yaitu *Requirement Definition, System and Software Design, Implementation and Unit Testing, Integration and System testing*, dan *Operation and Maintenance* (Sommerville, 2010).



Gambar 2.1 Model Waterfall

Sumber: Sommerville (2010)

Menurut Pressman (2010), *Waterfall Model* merupakan salah satu model dalam pengembangan perangkat lunak dimana semua proses kegiatan harus direncanakan dan dijadwalkan sebelum dikerjakan. Proses dari metode *Waterfall model* (Sommerville, 2011) antara lain:

1. *Requirement Definition*
Langkah ini merupakan tahapan untuk analisis kebutuhan sistem dengan cara pengumpulan data dari *stackholder* yang berguna untuk kebutuhan sistem maupun data tambahan dari jurnal, artikel dan lain-lain.
2. *System and Software Design*
Langkah ini merupakan tahapan yang akan menghasilkan dokumen perancangan dari sistem. Perancangan sistem ini berasal dari data yang telah diperoleh dari identifikasi kebutuhan sistem.
3. *Implementation and Unit Testing*
Langkah ini merupakan tahap mengubah hasil perancangan menjadi kode program sesuai dengan apa yang telah dirancang. Semua algoritma diterapkan pada bagian ini untuk menyesuaikan dengan kebutuhan pengguna.
4. *Integration and System Testing*
Langkah ini merupakan tahap pengujian pada semua fungsi pada sistem untuk memeriksa apakah semua berjalan sesuai dengan kebutuhan pengguna dan perancangan sistem.
5. *Operation and Maintenance*
Langkah ini bisa dikatakan tahapan final dalam pembuatan sebuah sistem, setelah tahap 1 sampai 4 kemudian sistem akan digunakan oleh

pengguna. Serta perbaikan dan pengembangan sistem juga dilakukan untuk memantau kinerja dari sistem tersebut.

2.4 Sistem Informasi

Sistem Informasi merupakan sistem yang mempunyai kemampuan untuk mengumpulkan informasi dari semua sumber dan menggunakan berbagai media untuk menunjukkan informasi (Mc Leod, 2001). Sistem informasi merupakan alat untuk menyajikan informasi demikian rupa sehingga bermanfaat bagi penerimanya. Tujuannya adalah untuk memberikan informasi dalam perencanaan, memulai, pengorganisasian, oprasional sebuah perusahaan yang melayani sinergi organisasi dalam proses mengendalikan pengambilan keputusan (Kertahadi, 2007). Sistem informasi yang diselenggarakan cara untuk mengumpulkan, memasukkan, mengolah, dan menyimpan data dan terorganisasi cara untuk meyimpan, mengelola, mengendalikan dan melaporkan informasi dengan cara yang suatu orgnisasi dapat mencapai tujuan yang telah ditetapkan (Rommey, 1997).

Oleh karena itu, dapat disimpulkan bahwa sistem informasi merupakan sistem yang dapat mengumpulkan data serta mengolah data tersebut menjadi sebuah informasi yang dapat digunakan untuk mendapat pengetahuan atau menentukan keputusan.

2.5 Manajemen Keuangan

Manajemen keuangan dapat diartikan sebagai manajemen dana baik yang berkaitan dengan pengalokasian dana dalam berbagai bentuk investasi secara efektif maupun usaha pengumpulan dana untuk pembiyaan investasi atau pembelanjaan secara efisien. Pelaksana dari manajemen keuangan adalah manajer keuangan. Meskipun fungsi seorang manajer keuangan setiap organisasi belum tentu sama, namun pada prinsipnya fungsi utama seorang manager keuangan adalah merencanakan, mencari, dan memanfaatkan dengan berbagai cara untuk memaksimalkan efisiensi (daya guna) dari operasi-operasi perusahaan (Sartono, 2011).

Manajemen keuangan merupakan aktivitas pemilik dan meminjam perusahaan untuk memperoleh sumber modal yang semurah-murahnya dan menggunakan seefektif, seefisien, dan seekonomis mungkin untuk menghasilkan laba (Darsono, 2011). Oleh karena itu, dapat disimpulkan bahwa manajemen keuangan adalah pengalokasian atau pengelolaan dana dengan efektif dan efisien untuk menghasilkan keuntungan yang maksimal.

2.6 *Bussiness Process Managemenet Notation (BPMN)*

BPMN adalah notasi untuk menggambarkan proses bisnis yang memungkinkan sebuah perusahaan memahami proses bisnis internal. BPMN dibatasi hanya untuk mendukung konsep permodelan yang berlaku untuk proses bisnis (Object Management Group, 2011). BPMN memilik 5 kategori elemen

sebagai penunjang dalam pembuatan BPMN (Object Management Group, 2011), antara lain:

2.6.1 Flow Object

Flow Object dibagi menjadi 3 bagian yaitu *event*, *activity*, dan *gateway*:

1. Events

Event merupakan sesuatu yang terjadi ketika proses bisnis dijalankan serta mempengaruhi alur dari proses. *Events* pada umumnya adalah notasi yang merepresentasikan *trigger* dan *results* pada sebuah proses bisnis. Tipe notasi pada *events* terdiri dari *start event*, *intermediate event*, dan *end* yang akan dijelaskan pada Tabel 2.1.

Tabel 2.1 Notasi Events

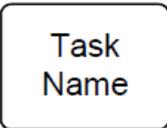
Nama	Notasi	Deskripsi
<i>Start Event</i>		Notasi yang merepresentasikan sebuah proses dimulai.
<i>Intermediate Event</i>		Notasi ini muncul di antara <i>start event</i> dan <i>end</i> , serta akan memengaruhi alur proses bisnis. Notasi ini tidak dapat digunakan untuk memulai dan mengakhiri sebuah proses secara langsung. Notasi ini memiliki beberapa macam varian untuk merepresentasikan event yang berbeda-beda. Setiap varian diindikasikan dengan penggunaan simbol-simbol yang berbeda di tengah notasi ini.
<i>End</i>		Notasi yang merepresentasikan sebuah proses telah selesai.

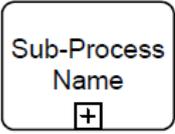
Sumber: *Object Management Group* (2011)

2. Activity

Activity merupakan tugas yang dijalankan dalam proses bisnis. *Activity* ini juga menjelaskan nama dari tugas yang dilakukan dalam proses bisnis. Tipe notasi pada *activity* terdiri dari *task* dan *sub-process* yang akan dijelaskan pada Tabel 2.2.

Tabel 2.2 Notasi Activity

Nama	Notasi	Deskripsi
<i>Task</i>		Notasi yang merepresentasikan aktivitas yang bersifat <i>atomic</i> , yang berarti aktivitas tersebut tidak dipecah ke tingkat yang lebih detail.

Nama	Notasi	Deskripsi
Sub-process		Notasi yang digunakan untuk merepresentasikan sekumpulan aktivitas. <i>Sub-process</i> memiliki detail yang berisi sekumpulan aktivitas di dalamnya.

Sumber: *Object Management Group* (2011)

3. Gateways

Gateways dapat mendefinisikan dan mengontrol arus proses bisnis. Tipe notasi pada *gateways* terdiri dari *exclusive*, *parallel*, *event-based*, *complex*, dan *inclusive* yang akan dijelaskan pada Tabel 2.3.

Tabel 2.3 Notasi Gateways

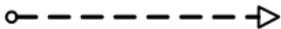
Notasi	Nama	Deskripsi
<i>Exclusive</i>		Digunakan untuk membuat alur alternatif pada proses bisnis. Dari beberapa alur alternatif yang dibuat, hanya 1 alur alternatif yang dapat dijalankan sesuai dengan kondisi yang terpenuhi.
<i>Parallel</i>		Digunakan untuk membuat alur paralel dan menyinkronkan alur paralel yang telah dibuat.
<i>Event-based</i>		Digunakan untuk membuat alur alternatif pada proses bisnis. Alur yang berjalan dipicu oleh sebuah <i>events</i> tertentu yang terjadi.
<i>Complex</i>		Digunakan untuk memvisualisasikan alur kompleks dalam proses bisnis.
<i>Inclusive</i>		Digunakan untuk membuat alur alternatif sekaligus dapat berjalan secara paralel pada proses bisnis. Kondisi pada setiap alur diperiksa, kemudian alur yang memenuhi kondisi dijalankan.

Sumber: *Object Management Group* (2011)

2.6.2 Connections

Connection merupakan garis yang menghubungkan antar *flow object* dengan notasi BPMN lainnya. Tipe notasi pada *connections* terdiri dari *sequence flow*, *message flow*, *associations*, dan *data association* yang akan dijelaskan pada Tabel 2.4.

Tabel 2.4 Notasi *Connections*

Nama	Notasi	Deskripsi
<i>Sequence Flow</i>		Notasi yang digunakan untuk menunjukkan urutan aktivitas yang akan dijalankan di dalam proses bisnis.
<i>Message Flow</i>		Notasi yang digunakan untuk menunjukkan aliran yang dikirimkan dan diterima sebuah aktivitas yang dimiliki oleh partisipan yang berbeda.
<i>Associations</i>		Notasi yang digunakan untuk menghubungkan notasi-notasi <i>artifacts</i> yang berisi informasi dengan notasi-notasi BPMN lainnya.
<i>Data Associations</i>		Notasi yang digunakan untuk menghubungkan aktivitas BPMN dengan notasi data.

Sumber: *Object Management Group* (2011)

2.6.3 *Swimlanes*

Swimlanes adalah jenis notasi yang digunakan untuk mengelompokkan aktivitas berdasarkan unit bisnis yang menjalankan aktivitas-aktivitas tertentu. *Swimlanes* pada umumnya digunakan untuk memberikan informasi yang jelas untuk aktivitas *business-to-business*. Tipe notasi pada *swimlanes* terdiri dari *pool* dan *lane* yang akan dijelaskan pada Tabel 2.5.

Tabel 2.5 Notasi *Swimlanes*

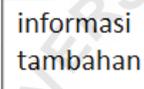
Nama	Notasi	Deskripsi
<i>Pool</i>		Notasi yang digunakan untuk merepresentasikan partisipan sebuah proses bisnis. <i>Pool</i> dapat memiliki detail di dalamnya, sehingga seluruh aktivitas yang dimiliki oleh sebuah partisipan dapat dilihat pada model proses bisnis. Namun <i>pool</i> juga dapat bersifat <i>black-box</i> .
<i>Lane</i>		<i>Lane</i> merupakan sub-partisi dari sebuah <i>pool</i> , notasi ini merepresentasikan unit-unit yang dimiliki oleh sebuah partisipan proses bisnis.



2.6.4 Artifacts

Artifact dapat mendefinisikan informasi tambahan yang berada di luar proses. Tipe notasi pada *artifacts* terdiri dari *group* dan *text annotation* yang akan dijelaskan pada Tabel 2.6.

Tabel 2.6 Notasi *Artifacts*

Nama	Notasi	Deskripsi
<i>Group</i>		Notasi yang digunakan untuk memberikan informasi pengelompokan notasi-notasi kedalam sebuah kategori. Notasi ini tidak memengaruhi alur proses bisnis dan pada umumnya digunakan untuk kepentingan dokumentasi dan analisis. Nama notasi group diletakkan di dalam notasi ini, umumnya diletakkan di sisi bawah notasi.
<i>Text Annotation</i>		Notasi ini digunakan untuk memberikan informasi dalam bentuk teks untuk memudahkan pembaca BPMN memahami proses bisnis.

Sumber: *Object Management Group* (2011)

2.6.5 Data

Data digunakan untuk visualisasi dokumen, data, dan objek lain digunakan dan diperbarui pada sebuah alur proses. Tipe notasi pada *data* terdiri dari *data object*, *data object(collection)*, *data input*, *data output* dan *data store* yang akan dijelaskan pada Tabel 2.7.

Tabel 2.7 Notasi *Data*

Nama	Notasi	Deskripsi
<i>Data Object</i>		Notasi yang digunakan untuk memberikan informasi tambahan mengenai data atau dokumen atau objek lain yang berkaitan dengan aktivitas atau proses tertentu.
<i>Data Object (Collection)</i>		Memvisualisasikan <i>data object</i> yang berjumlah lebih dari 1.
<i>Data Input</i>		Notasi yang digunakan untuk memvisualisasikan data atau dokumen atau objek lain yang menjadi <i>input</i> sebuah atau serangkaian aktivitas.

Nama	Notasi	Deskripsi
<i>Data Output</i>		Notasi yang digunakan untuk memvisualisasikan data atau dokumen atau objek lain yang menjadi <i>output</i> sebuah atau serangkaian aktivitas.
<i>Data Store</i>		Notasi yang digunakan untuk memberikan informasi mengenai sebuah lokasi penyimpanan dan pembaruan data.

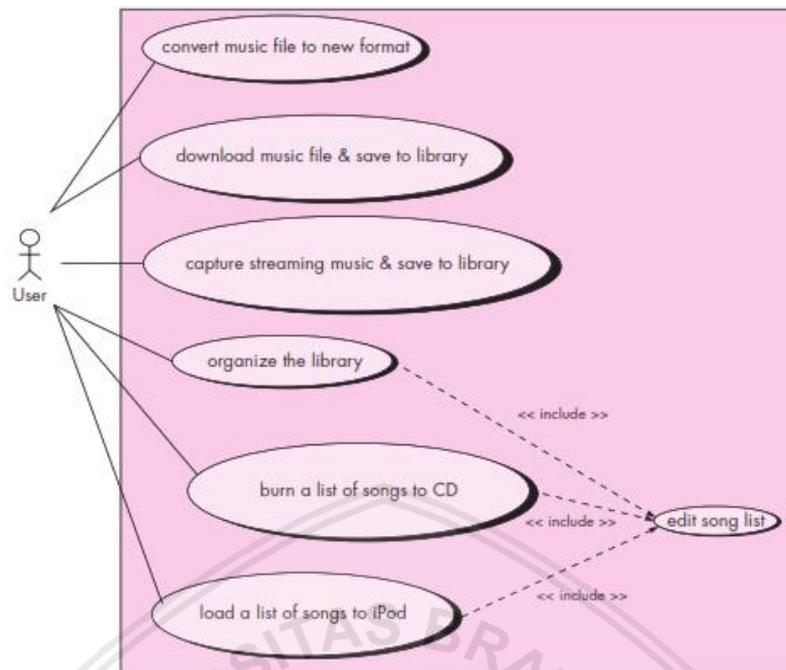
Sumber: *Object Management Group* (2011)

2.7 Unified Modelling Language (UML)

UML merupakan bahasa untuk membuat perancangan sistem. UML dapat digunakan untuk membangun dan mendokumentasikan gambaran sistem. Dengan membuat rancangan arsitektur, pembuatan perangkat lunak dapat lebih terbantu dengan adanya UML. Oleh karena itu, mudah untuk dipahami, mudah untuk menentukan sistem dan mudah untuk menjelaskan desain sistem (Pressman, 2010).

2.7.1 Use Case Diagram

Pemodelan use case dilakukan untuk mengetahui informasi mengenai pemangku kepentingan yang akan terlibat dan terkena dampak dari sebuah sistem, mengetahui siapa saja calon pengguna sistem, mengetahui kebutuhan pemangku kepentingan dan pengguna sistem berdasarkan masalah yang mereka hadapi, serta untuk mengetahui kemungkinan solusi yang dapat ditawarkan kepada pemangku kepentingan dan pengguna untuk memenuhi kebutuhannya. Berikut contohnya pada Gambar 2.2.



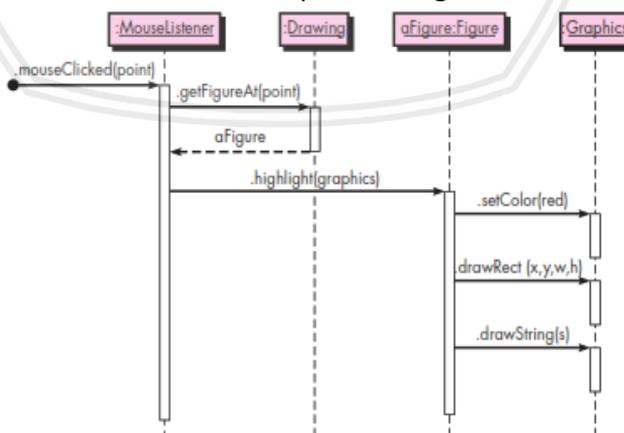
Gambar 2.2 Use Case Diagram

Sumber: Pressman (2010)

Semua hubungan antar *use case* berbentuk oval dan dihubungkan oleh garis, hal ini bisa digunakan untuk memastikan semua fungsi dalam sistem sudah teridentifikasi (Pressman, 2010)

2.7.2 Sequence Diagram

Sequence diagram menampilkan struktur dinamis antar objek dari sebuah sistem selama fungsi dijalankan. *Sequence diagram* menunjukkan proses pengiriman pesan antar objek untuk menyelesaikan fungsi tertentu (Pressman, 2010). Gambar 2.3 adalah contoh dari *sequence* diagram.



Gambar 2.3 Sequence Diagram

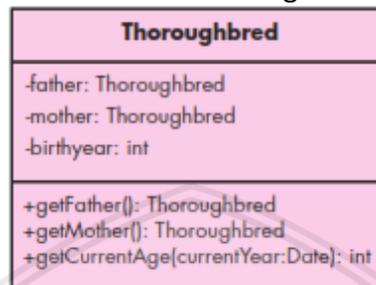
Sumber: Pressman (2010)



Sequence diagram berisi notasi-notasi yang merepresentasikan *instances* dari sebuah kelas, antarmuka, komponen, dan *nodes*, beserta pertukaran pesan di antara *instances* tersebut.

2.7.3 Class Diagram

Class diagram menampilkan struktur statis dari sebuah sistem, *class diagram* tidak menampilkan struktur dinamis dari komunikasi antar objek setiap kelas dalam diagram. Berikut contoh *class diagram*:



Gambar 2.4 Class Diagram

Sumber: Pressman (2010)

Masing-masing atribut dan operasi yang dimiliki oleh kelas dapat ditentukan visibilitasnya dan digambarkan dengan notasi yang diletakkan sebelum nama atribut atau nama operasi.

2.8 Laravel

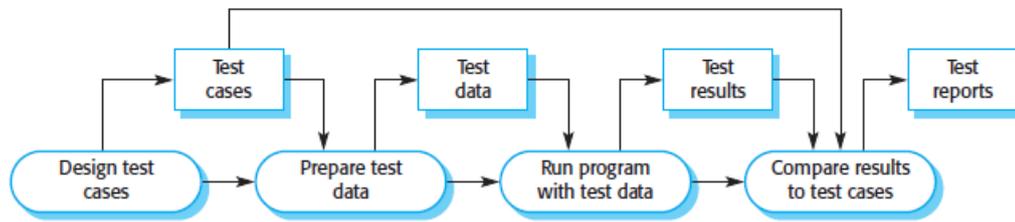
Framework Laravel dibuat oleh Taylor Otwell, proyek *Laravel* dimulai pada April 2011. Awal mula proyek ini dibuat karena Otwell sendiri tidak menemukan *framework* yang up-to-date dengan versi PHP. Mengembangkan *framework* yang sudah ada juga bukan merupakan ide yang bagus karena keterbatasan sumber daya. Dikarenakan beberapa keterbatasan tersebut, Otwell membuat sendiri *framework* dengan nama *Laravel*. *Laravel* adalah suatu *framework* PHP yang dirilis dibawah lisensi MIT, dibangun dengan konsep MVC (*Model View Controller*) (Rohman, 2014).

Laravel adalah pengembangan *website* berbasis MVP yang ditulis dalam PHP yang dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya pengembangan awal dan biaya pemeliharaan, dan untuk meningkatkan pengalaman bekerja dengan aplikasi dengan menyediakan sintak yang ekspresif, jelas dan menghemat waktu. *Laravel* merupakan *framework* yang mudah diakses, *powerful* dan menyediakan *tools* yang diperlukan untuk skala aplikasi besar. *Laravel* juga merupakan sebuah aplikasi luar biasa dari sebuah kumpulan program kontrol, sistem migrasi yang ekspresif dan dukungan *tools* yang dibutuhkan dalam menguji aplikasi yang terintegrasi dengan beberapa aplikasi lainnya (Rohman, 2014). Beberapa fitur yang ada pada *Laravel* (*Laravel Documentation*, 2018):

1. *Bundles* yaitu sebuah fitur dengan sistem pengemasan modular dan berbagai *bundle* telah tersedia untuk digunakan dalam aplikasi.
2. *Eloquent ORM* merupakan penerapan PHP lanjutan dari pola “*active record*”, menyediakan metode internal untuk mengatasi kendala hubungan antara objek *database*.
3. *Application logic* merupakan bagian dari aplikasi yang dikembangkan baik menggunakan *Controllers* atau sebagai bagian dari deklarasi *Route*. Sintak yang digunakan untuk mendefinisikannya mirip dengan yang digunakan oleh *framework Sinatra*.
4. *Reverse routing*, mendefinisikan hubungan antara *Link* dan *Route*, sehingga jika suatu saat ada perubahan pada *route* secara otomatis akan tersambung dengan *link* yang relevan. Ketika *Link* yang dibuat dengan menggunakan nama-nama dari *Route* yang ada, secara otomatis *Laravel* akan membuat URI yang sesuai.
5. *Restful controllers*, memberikan sebuah pilihan untuk memisahkan logika dalam melayani HTTP GET dan permintaan POST.
6. *Class auto loading*, menyediakan otomatis pemuatan untuk kelas PHP, tanpa membutuhkan pemeriksaan manual terhadap jalur masuknya. Fitur ini mencegah muatan yang tidak perlu.
7. *View composers* adalah kode unit logical yang dapat dieksekusi ketika sebuah *View* dimuat.
8. *IoC Container* memungkinkan untuk objek baru yang dihasilkan dengan mengikuti prinsip control pembalik, dengan pilihan contoh dan referensi dari objek baru sebagai *singletons*.
9. *Migrations* menyediakan versi sistem kontrol untuk skema *database*, sehingga memungkinkan untuk menghubungkan perubahan dalam basis kode aplikasi dan keperluan yang dibutuhkan dalam merubah tata letak *database*, mempermudah dalam penempatan dan memperbarui aplikasi.
10. *Unit Testing* mempunyai peran penting dalam *framework Laravel*, dimana *unit testing* ini mempunyai banyak tes untuk mendeteksi dan mencegah regresi. Unit testing ini dapat dijalankan melalui utilitas “*artisan command-line*”
11. *Automatic pagination* menyederhanakan tugas dari penerapan halaman, menggantikan penerapan yang manual dengan metode otomatis yang terintegrasi ke *Laravel*.

2.9 Pengujian Perangkat Lunak

2.9.1 Validation Testing



Gambar 2.5 Validation Testing

Pengujian ini berfokus pada fungsional sistem, pengujian penting dilakukan karena yang dilihat dari sistem pertama kali adalah fungsionalitas sistem seperti *inputan* pengguna, aksi pengguna, dan lainnya. *Validation testing* ini dilakukan untuk mengetahui apakah fungsionalitas dari sistem informasi sudah sesuai dengan kebutuhan. Oleh karena itu, tujuan pengujian perangkat lunak adalah untuk meyakinkan pengembang sistem dan *stakeholder* bahwa perangkat lunaknya cukup baik untuk penggunaan operasional.

Kasus uji adalah spesifikasi masukkan untuk pengujian dan output yang diharapkan dari sistem sekaligus pernyataan tentang apa yang sedang diuji. Data uji adalah input yang telah dirancang untuk menguji sistem. Data uji terkadang dapat dihasilkan secara langsung. Keluaran dari tes hanya dapat diprediksi oleh orang yang mengerti apa yang harus dilakukan sistem. Kebijakan pengujian mungkin didasarkan pada pengalaman penggunaan sistem dan dapat fokus pada pengujian fitur sistem operasional. Seperti semua fungsi sistem harus diuji dan masukkan harus diuji dengan benar (Sommerville, 2010).

2.9.2 Browser Compatibility Testing

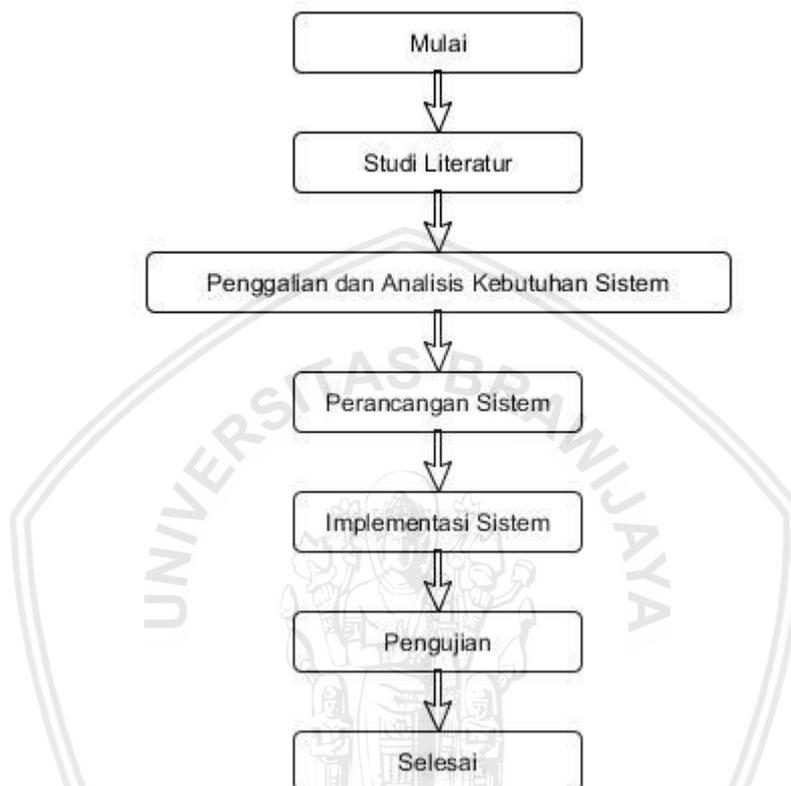
Komputer, layar, sistem operasi, *browser*, dan kecepatan koneksi yang berbeda dapat memiliki pengaruh signifikan pada operasi *WebApp*. Setiap konfigurasi komputasi dapat menghasilkan perbedaan dalam kecepatan proses pada sisi klien, resolusi layar, dan kecepatan koneksi. Sistem operasi juga dapat menyebabkan masalah saat proses *WebApp*. *Browser* yang berbeda terkadang menghasilkan yang sedikit perbedaan hasil, terlepas dari tingkat standarisasi HTML dalam *WebApp*. Dalam beberapa kasus, masalah kompatibilitas kecil tidak menimbulkan masalah berarti, tetapi dalam kasus lain ditemukan kesalahan serius. Seperti kecepatan unduhan menjadi tidak stabil dan kurangnya plug-in yang diperlukan dapat membuat konten tidak tersedia. Perbedaan browser dapat mengubah tata letak halaman secara dramatis, gaya font dapat diubah dan menjadi tidak terbaca, atau formulir mungkin tidak diorganisasi dengan benar. Pengujian kompatibilitas berusaha untuk mengungkap masalah ini sebelum *WebApp* digunakan masyarakat luas. Langkah pertama dalam pengujian kompatibilitas adalah menentukan serangkaian konfigurasi komputasi sisi klien

dan variannya. Mengidentifikasi setiap platform komputasi, perangkat tampilan tipikal, sistem operasi yang didukung pada platform, *browser* yang tersedia, kemungkinan kecepatan koneksi internet, dan informasi serupa. Selanjutnya, serangkaian validasi kompatibilitas tes diturunkan, sering diadaptasi dari tes antarmuka yang ada, tes navigasi, tes kinerja, dan tes keamanan. Maksud dari tes ini adalah untuk mengungkap kesalahan atau masalah eksekusi yang dapat ditelusuri ke perbedaan konfigurasi. (Pressman, 2010).



BAB 3 METODOLOGI

Pada bab ini membahas mengenai alur penelitian atau kerangka kerja dalam pengembangan sistem informasi manajemen keuangan pada peternakan Yoga's farm. Berikut kerangka kerja penelitian ini bisa dilihat pada Gambar 3.1.



Gambar 3.1 Kerangka Kerja Penelitian

3.1 Studi Literatur

Studi literatur dilakukan untuk menggali pengetahuan detail tentang teori yang digunakan dengan penelitian ini. Sumber yang digunakan dalam studi literatur ini berasal dari jurnal, buku, skripsi dan artikel yang berkaitan dengan penelitian serta mendukung konsep dari penelitian.

3.2 Penggalian dan Analisis Kebutuhan Pengguna

Pada tahap ini dilaksanakan observasi dan wawancara untuk memperoleh data kebutuhan sistem yang dilakukan pada peternakan Yoga's Farm. Kebutuhan sistem yang diobservasi didapat langsung dari pemilik peternakan yang berfokus pada manajemen keuangan. Analisis kebutuhan merupakan tahap dimana kebutuhan yang telah didapat dari hasil wawancara dan observasi ditinjau kembali. Hal ini dilakukan untuk memastikan kebutuhan *stakeholder* sudah terpenuhi. Kemudian dilakukan analisis proses bisnis untuk mendefinisikan

masalah dan solusinya. Dalam analisis ini akan menghasilkan kebutuhan fungsional dan nonfungsional. Jika kebutuhan fungsional dan nonfungsional sudah terdefinisi maka akan dimodelkan menggunakan *use case diagram* untuk mengetahui apa saja yang bisa dilakukan pengguna di dalam sisten dan *use case spesififikasi* untuk menjelaskan proses berjalannya setiap *use case*.

3.3 Perancangan Sistem

Pada tahapan ini persyaratan fungsional dan nonfungsional sudah diidentifikasi. Perancangan sistem informasi dilakukan untuk memodelkan sistem ke dalam notasi-notasi pada *Unified Modeling Language (UML)* berdasarkan hasil tahap analisis persyaratan. Perancangan sistem berisi hasil pemodelan interaksi antar objek ke dalam *sequence diagram*, pemodelan objek ke dalam *class diagram*, perancangan *database*, perancangan algoritma ke dalam *pseudocode*, dan perancangan antarmuka pengguna sistem. Semua kebutuhan untuk implimentasi sistem dijelaskan pada bagian ini.

3.4 Implementasi Sistem

Pada bagian implementasi dilakukan proses pembuatan sistem informasi manajemen keuangan peternakan(SMIKP) Yoga's Farm sesuai dengan hasil perancangan pada bab sebelumnya. SMIKP dikembangkan dengan menggunakan bahasa pemrograman PHP dan *Javascript*, dengan kerangka kerja Laravel. Hasil dari tahap pembuatan SMIKP menyediakan fitur yang sesuai dengan yang sudah teridentifikasi pada tahap analisis kebutuhan. Kegiatan dokumentasi juga dilakukan untuk tahap implementasi. Dokumentasi tahap implementasi berisi informasi mengenai perangkat keras dan perangkat lunak yang digunakan untuk mengembangkan sistem informasi. Selain itu, dokumentasi juga berisi hasil implementasi algoritma dalam bentuk kode program dan antarmuka pengguna.

3.5 Pengujian

Pada tahap ini perancangan dan implementasi sudah selesai dilaksanakan. Dari hasil perancangan dan implementasi sistem perlu dilaksanakan pengujian terhadap sistem yang bertujuan menguji tingkat keberhasilan sistem. Pengujian dilakukan dengan menyesuaikan perancangan awal dan implemtasi, apakah sudah sesuai dengan kebutuhan *stakeholder*. Dalam pengujian ini digunakan 2 pengujian yaitu (1) *Validation testing* yang digunakan untuk mengetahui kesesuaian fungsi sistem dengan yang diharapkan, kasus uji diperoleh berdasarkan penggunaan sistem oleh aktor yang didapat berdasarkan alur *use case* pada spesifikasi *use case*; (2) *Browser compatibilty testing* dilakukan dengan menggunakan aplikasi SortSite untuk menguji kompatibilitas sistem informasi dengan beberapa aplikasi peramban. Jika terdapat hasil pengujian yang menyatakan sistem membutuhkan perbaikan, maka dilakukan kegiatan perbaikan sistem informasi pada tahap implementasi.

BAB 4 PENGGALIAN DAN ANALISIS KEBUTUHAN SISTEM

Pada bagian analisis kebutuhan sistem ini akan menjelaskan analisis kebutuhan dari sistem informasi manajemen keuangan peternakan kambing. Pembahasan analisis yang pertama menjelaskan hasil wawancara dari *stakeholder* dengan membuat tabel untuk memetakan informasi yang diperoleh. Informasi yang diperoleh tersebut merupakan proses bisnis dari peternakan yang akan digambarkan dalam diagram BPMN. Setelah proses bisnis digambarkan maka analisis masalah yang ada pada peternakan sudah terdefinisi. Hasil dari analisis masalah ini berguna untuk menentukan saran tentang rincian sistem informasi. Berdasarkan saran yang sudah diperoleh, dilanjutkan untuk pembuatan analisis persyaratan fungsional dan nonfungsional. Persyaratan ini akan dimanfaatkan dalam membuat *Use Case Diagram* dan *Use Case Specification*.

4.1 Proses Bisnis AS-IS

Dari hasil wawancara yang didapat dari pemilik peternakan, proses bisnis yang didapatkan dalam manajemen keuangan dimulai dari melihat ketersediaan barang yang ada sampai barang selesai dijual atau dibeli. Gambar 4.1 dan Gambar 4.2 akan menjelaskan notasi dari proses bisnis yang ada saat ini.

Pada Gambar 4.1 dipetakan proses bisnis AS-IS pembelian, berikut penjelasan urutan proses bisnis pembelian:

1. Pemilik peternakan menghubungi *supplier*
2. *Supplier* memeriksa ketersediaan barang, jika barang tidak tersedia maka *supplier* memberi informasi pada pemilik peternakan bahwa barang tidak ada dan proses berhenti. Jika barang tersedia *supplier* memberi informasi barang tersedia dan jumlah yang harus dibayarkan.
3. Pemilik peternakan menerima informasi barang tidak tersedia maka berhenti dan jika menerima informasi barang tersedia dari *supplier* pemilik peternakan langsung memesan barang pada *supplier*.
4. *Supplier* menerima pesanan dari pemilik peternakan dan melakukan pengiriman barang ke peternakan
5. Barang yang sudah dikirim dimasukkan dalam gudang, kemudian pemilik peternakan melakukan pembayaran pada *supplier* dan *supplier* memeriksa pembayaran, proses berhenti
6. Pemilik memberi informasi pada petugas keuangan tentang pembayaran yang telah dilakukan.
7. Petugas keuangan menulis data transaksi, menghitung keuntungan peternakan, kemudian menyimpan laporan, dan proses berhenti.

Pada Gambar 4.2 dipetakan proses bisnis AS-IS penjualan, berikut penjelasan urutan proses bisnis pembelian:

1. *Customer* datang ke peternakan dan melihat langsung informasi ketersediaan barang
2. Pemilik peternakan memeriksa ketersediaan barang, jika barang tersedia maka pemilik memberi informasi barang tersedia pada *customer* dan jika

barang tidak tersedia pemilik juga memberi informasi barang tidak tersedia.

3. *Customer* menerima informasi barang tidak tersedia maka berhenti dan jika menerima informasi barang tersedia dari pemilik peternakan maka *customer* melakukan pembelian dan pembayaran.
4. Kemudian pemilik peternakan memeriksa pembayaran dan menyediakan barang yang akan diambil oleh *customer* dan proses berhenti.
5. Pemilik peternakan menyerahkan hasil transaksi pada petugas keuangan.
6. Petugas keuangan menerima pembayaran dari pemilik peternakan dan menulis data transaksi
7. Petugas keuangan menghitung keuntungan dan kemudian disimpan dalam bentuk laporan keuangan, proses selesai.

4.2 Proses Bisnis TO-BE

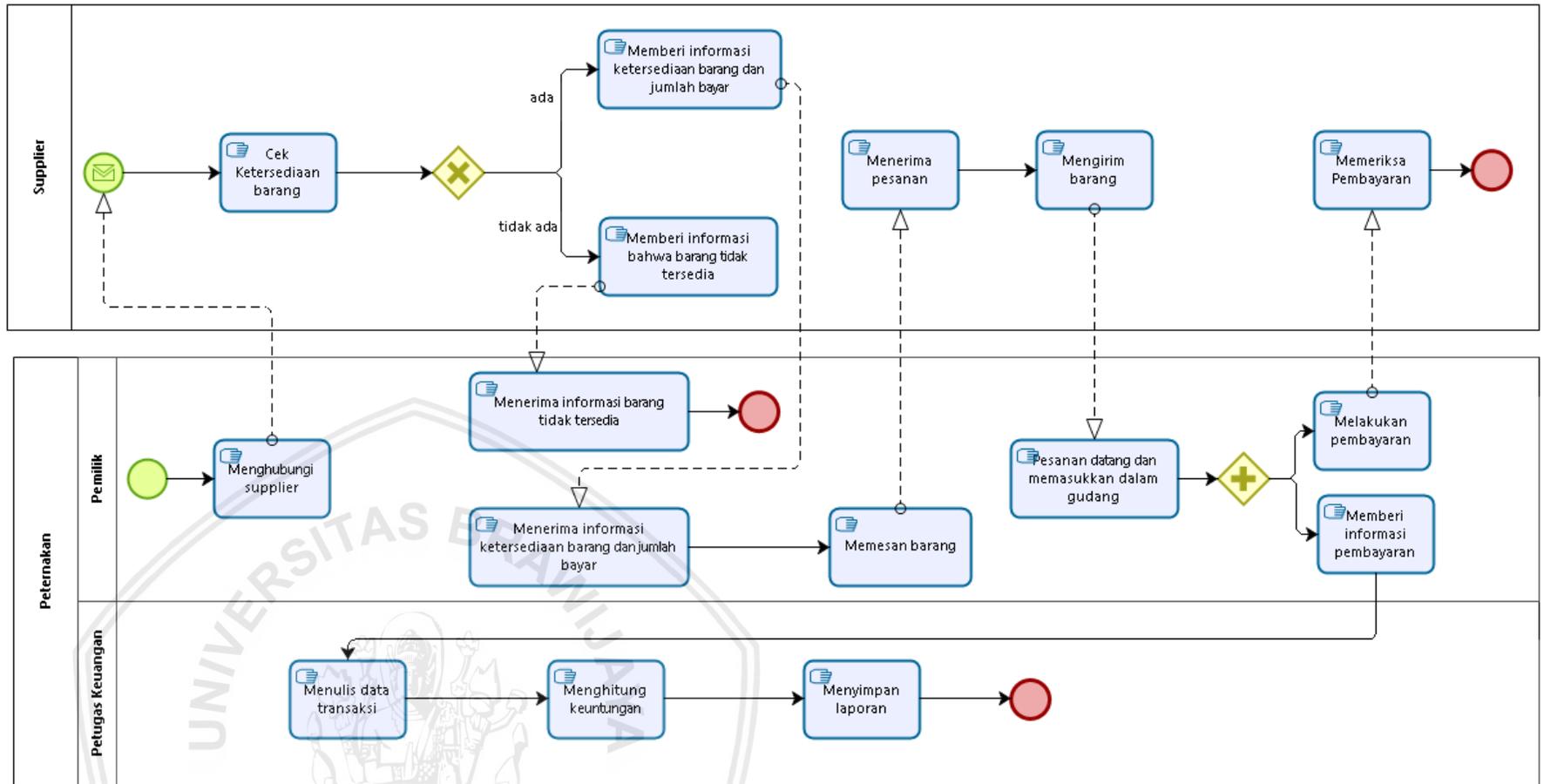
Berdasarkan proses bisnis pada Gambar 4.1 dan Gambar 4.2. Kemudian dilakukan analisis berikutnya yaitu analisis proses bisnis rekomendasi untuk menentukan proses bisnis rekomendasi yang sesuai. Berikut proses bisnis TO-BE dijelaskan pada Tabel 4.1.

Tabel 4.1 Analisis Proses Bisnis TO-BE

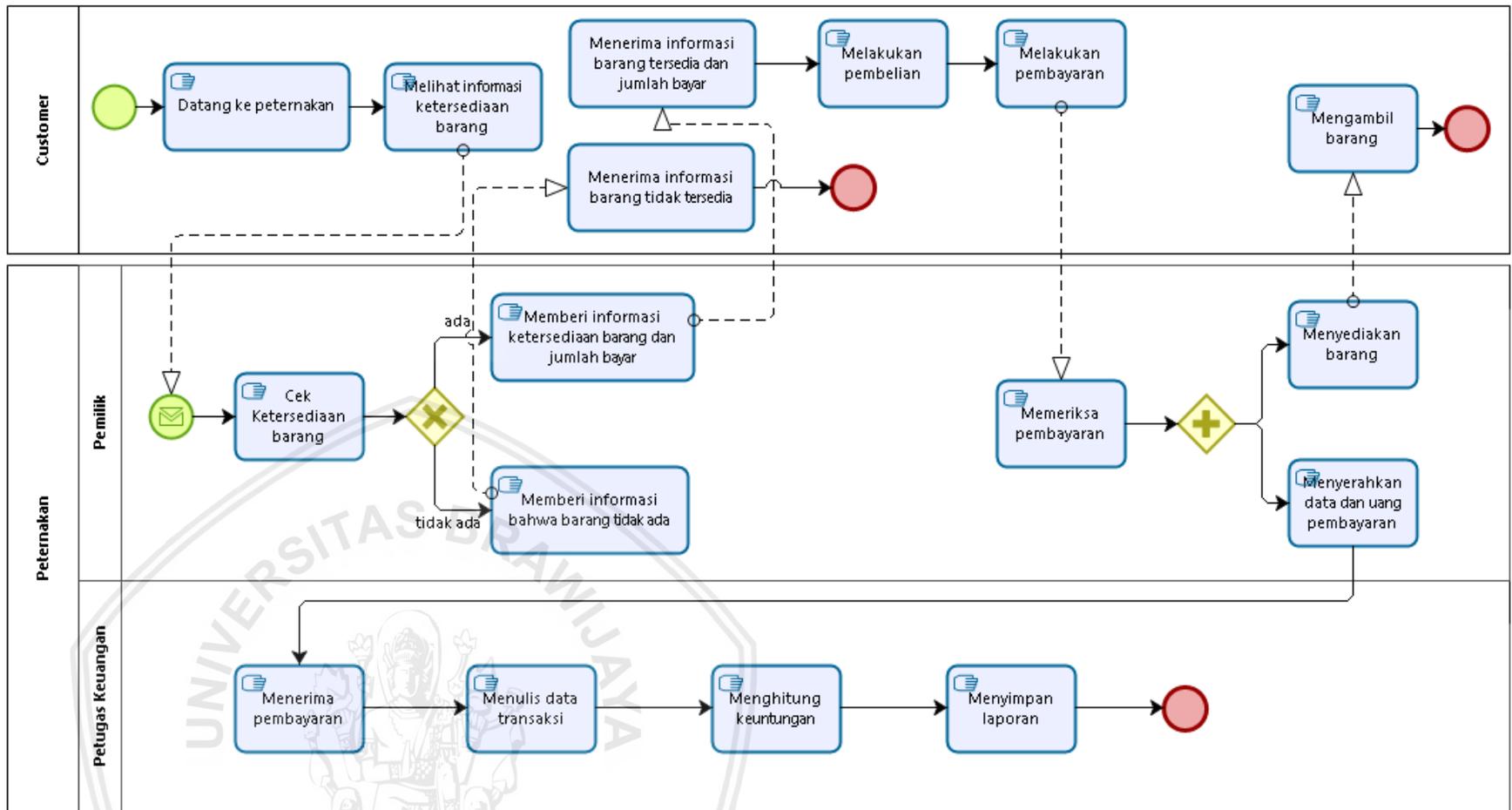
No.	Masalah	Proses Bisnis AS-IS	Proses Bisnis TO-BE
1.	Pelaporan keuangan tidak tertata dengan rapi	Menulis data transaksi	<ol style="list-style-type: none"> 1. Memasukkan data dalam sistem 2. Melihat rekap data 3. Menampilkan rekap data oleh sistem
2.	Keuntungan dari peternakan tidak bisa diidentifikasi secara jelas	Menghitung keuntungan	<ol style="list-style-type: none"> 1. Melihat keuntungan 2. Menghitung keuntungan dan menampilkan keuntungan oleh sistem
3.	Laporan data penjualan dan pembelian juga tidak teridentifikasi dengan baik	Menyimpan laporan	<ol style="list-style-type: none"> 1. Menyimpan data oleh sistem 2. Melakukan cetak laporan 3. Mencetak laporan oleh sistem

Pada Gambar 4.3 dipetakan proses bisnis TO-BE pembelian, berikut penjelasan urutan proses bisnis pembelian:

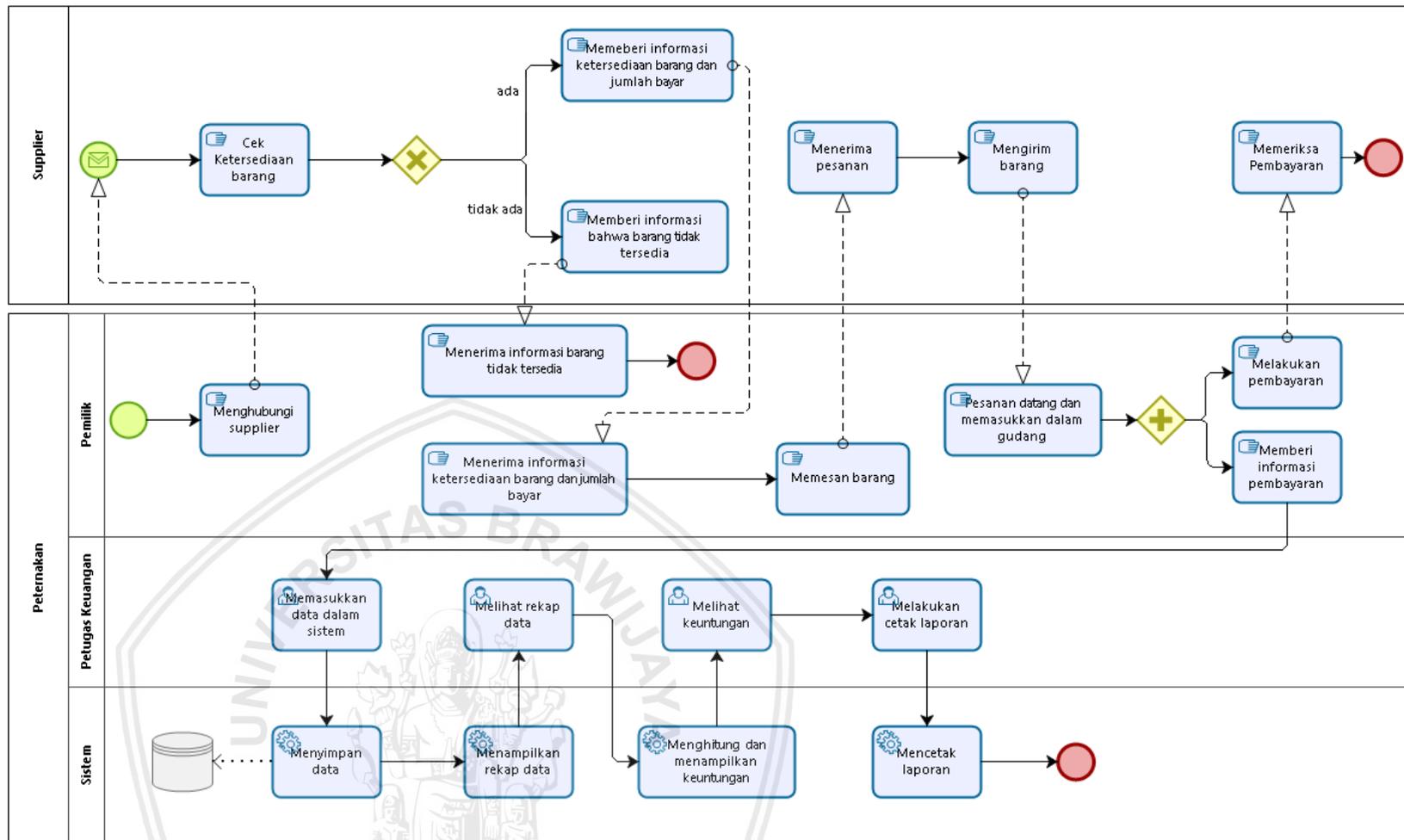
1. Pemilik peternakan menghubungi *supplier*.
2. *Supplier* memeriksa ketersediaan barang, jika barang ada maka *supplier* memberi informasi pada pemilik peternakan bahwa barang tersedia dan jumlah yang harus dibayar. Jika barang tidak tersedia maka memberi informasi barang tidak tersedia.
3. Pemilik peternakan menerima informasi barang tidak tersedia dan proses berhenti, jika pemilik peternakan menerima informasi barang tersedia dan jumlah bayar maka pemilik peternakan melakukan pemesanan barang pada *supplier*.
4. *Supplier* menerima pesanan dan mengirim barang ke peternakan.
5. Pesanan yang sudah datang langsung dimasukkan dalam gudang dan pemilik peternakan melakukan pembayaran, yang kemudian pembayaran diperiksa oleh *supplier* dan proses berhenti.
6. Pemilik peternakan memberi informasi pada petugas keuangan dan petugas keuangan memasukkan data dalam sistem.
7. Sistem menyimpan data dalam *database* dan menampilkannya pada petugas keuangan.
8. Petugas keuangan melihat data rekap yang telah ditampilkan oleh sistem, kemudian sistem menghitung dan menampilkan keuntungan dari peternakan.
9. Petugas keuangan melihat keuntungan yang diperoleh peternakan dan melakukan pencetakan laporan keuangan.
10. Sistem mencetak laporan keuangan peternakan dan proses berhenti.



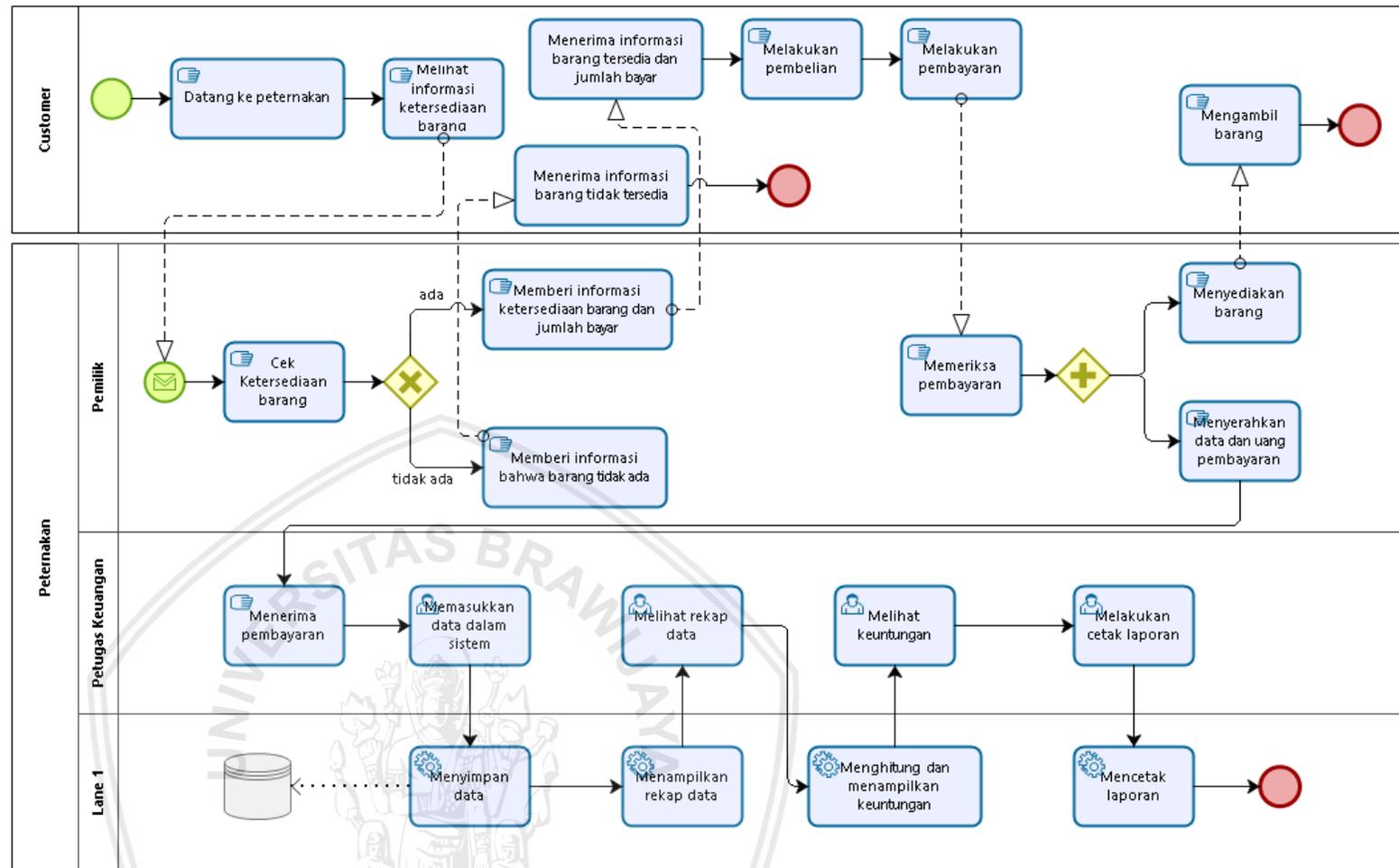
Gambar 4.1 Proses Bisnis AS-IS Pembelian



Gambar 4.2 Proses Bisnis AS-IS Penjualan



Gambar 4.3 Proses Bisnis TO-BE Pembelian



Gambar 4.4 Proses Bisnis TO-BE Penjualan

Pada Gambar 4.4 dipetakan proses bisnis TO-BE penjualan, berikut penjelasan urutan proses bisnis pembelian:

1. *Customer* datang ke peternakan dan melihat langsung informasi ketersediaan barang
2. Pemilik peternakan memeriksa ketersediaan barang, jika barang tersedia maka pemilik memberi informasi barang tersedia pada *customer* dan jika barang tidak tersedia pemilik juga memberi informasi barang tidak tersedia.
3. *Customer* menerima informasi barang tidak tersedia maka berhenti dan jika menerima informasi barang tersedia dari pemilik peternakan maka *customer* melakukan pembelian dan pembayaran.
4. Kemudian pemilik peternakan memeriksa pembayaran dan menyediakan barang yang akan diambil oleh *customer* dan proses berhenti.
5. Pemilik peternakan menyerahkan hasil transaksi pada petugas keuangan.
6. Petugas keuangan menerima pembayaran dari pemilik peternakan dan memasukkan data dalam sistem.
7. Sistem menyimpan data dalam *database* dan menampilkannya pada petugas keuangan.
8. Petugas keuangan melihat data rekap yang telah ditampilkan oleh sistem, kemudian sistem menghitung dan menampilkan keuntungan dari peternakan.
9. Petugas keuangan melihat keuntungan yang diperoleh peternakan dan melakukan pencetakan laporan keuangan.
10. Sistem mencetak laporan keuangan peternakan dan proses berhenti.

4.3 Analisis Persyaratan

4.3.1 Analisis Masalah

Pada sub bagian berikut dilakukan analisis masalah yang ada pada manajemen keuangan yoga's farm. Berikut Tabel 4.2 yang menjelaskan masalah yang ada.

Tabel 4.1 Analisis Masalah

<i>The problem of</i>	<ol style="list-style-type: none"> 1. Pelaporan keuangan masih tidak tertata dengan rapi 2. Keuntungan dari peternakan tidak bisa diidentifikasi secara jelas 3. Laporan data penjualan dan pembelian juga tidak teridentifikasi dengan baik
<i>Affects</i>	Keuntungan peternakan
<i>The impact of which is</i>	<ol style="list-style-type: none"> 1. Data keuangan tidak teridentifikasi dengan baik 2. Data penjualan dan pembelian tidak teridentifikasi dengan baik

A successful solution would	Sistem yang menyediakan layanan untuk mengelola manajemen keuangan.
------------------------------------	---

Dari Tabel 4.2 di atas, proses analisis masalah dilakukan. Hal pertama yang adalah menentukan poin-poin masalah yang ada dalam peternakan kambing, kedua mengidentifikasi dampak dari masalah tersebut, kemudian jika dampak itu sudah terjadi apa yang akan terjadi pada peternakan. Setelah semua itu teridentifikasi, maka bisa ditentukan solusi untuk menyelesaikan masalah tersebut.

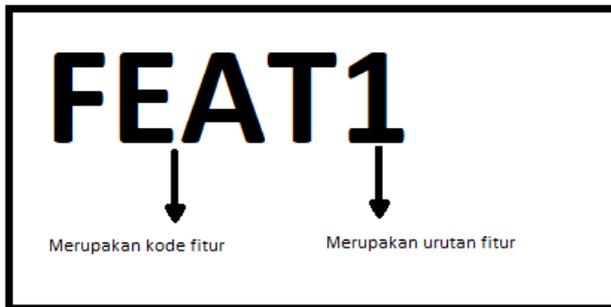
4.3.2 Identifikasi Pengguna

Pada sub bagian ini menjelaskan tentang identifikasi pengguna yang akan menggunakan sistem manajemen keuangan secara langsung. Identifikasi pengguna sistem dijelaskan pada Tabel 4.3.

Tabel 4.2 Identifikasi Pengguna

Tipe Pemangku Kepentingan	Tipe Pengguna	Deskripsi	Kebutuhan Pengguna	Solusi Yang Ditawarkan
Pengguna	Pemilik peternakan	Pemilik peternakan yang berprofesi menjadi peternak dan memiliki semua hak pada peternakan	Sistem harus bisa menampilkan keuntungan peternakan	Sistem informasi yang bisa menghitung keuntungan peternakan
	Petugas keuangan	Petugas peternakan yang mengelola seluruh manajemen keuangan dan data transaksi pada peternakan	Sistem harus bisa mengelola semua data transaksi peternakan	Sistem informasi yang bisa melakukan pengelolaan semua transaksi peternakan

4.3.3 Identifikasi Fitur

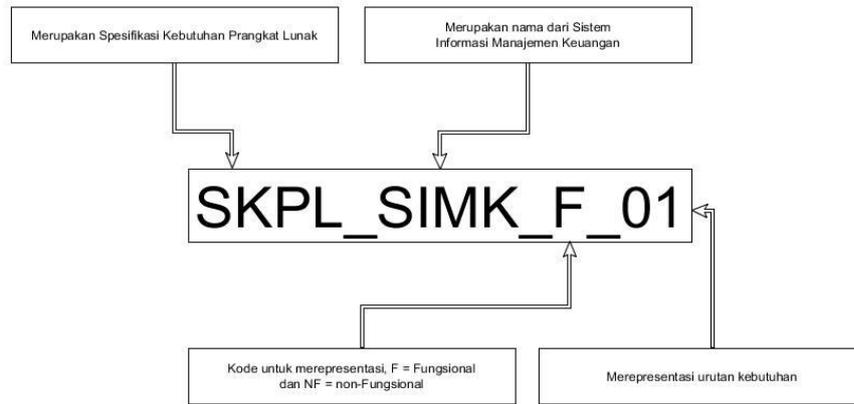


Identifikasi fitur pada penelitian ini memperlihatkan beberapa solusi yang ditawarkan untuk memenuhi kebutuhan pengguna, sehingga masalah dapat diselesaikan. Fitur yang telah diidentifikasi merupakan keterangan singkat dari layanan yang disediakan oleh sistem. Hasil identifikasi ini akan digunakan kembali untuk menjadi dasar melakukan identifikasi persyaratan fungsional dan nonfungsional. Berikut hasil dari identifikasi fitur dijelaskan pada Tabel 4.4.

Tabel 4.3 Fitur

Kode Fitur	Deskripsi
FEAT1	Sistem dapat digunakan untuk mengelola data transaksi yang terdiri dari penambahan, penyajian, pengubahan dan penghapusan data transaksi produk peternakan.
FEAT2	Sistem dapat digunakan untuk mengelola data maintenance yang terdiri dari penambahan, penyajian, pengubahan dan penghapusan data maintenance peternakan.
FEAT3	Sistem dapat digunakan untuk mengelola data keuangan lain yang terdiri dari penambahan, penyajian, pengubahan dan penghapusan data pemasukkan keuangan peternakan.
FEAT4	Sistem dapat digunakan untuk memperoleh hasil perhitungan keuntungan dalam bentuk laporan setiap bulan. Sistem juga memungkinkan untuk mencetak laporan keuangan setiap bulan dalam format pdf.

4.3.4 Persyaratan Fungsional



Gambar 4.5 Kodifikasi Persyaratan Fungsional

Pada sub bagian ini dilakukan analisis persyaratan fungsional yang ada pada sistem informasi manajemen keuangan pada Yoga’s Farm berbasis *website*. Analisis ini dilakukan dengan cara observasi langsung di peternakan serta wawancara *owner* peternakan. Berikut hasil analisis kebutuhan fungsional yang dijelaskan pada Tabel 4.5.

Tabel 4.4 Persyaratan Fungsional

Kode Fitur	Kode	Fungsi	Pengguna	Keterangan
FEAT1	SKPL_SIMK_F_01	Kelola produk jual	Keuangan	Berfungsi untuk memasukkan data dan mengelola produk jual
	SKPL_SIMK_F_02	Melihat data produk jual	Keuangan	Berfungsi untuk melihat data pada produk jual
	SKPL_SIMK_F_03	Kelola produk beli	Keuangan	Berfungsi untuk memasukkan data dan mengelola data produk beli
	SKPL_SIMK_F_04	Melihat data produk beli	Keuangan	Berfungsi untuk melihat data yang ada pada produk beli
FEAT2	SKPL_SIMK_F_05	Kelola data <i>maintenance</i>	Keuangan	Berfungsi untuk memasukkan data dan mengelola data <i>maintenance</i>

	SKPL_SIMK_F_06	Melihat data <i>maintenance</i>	Keuangan	Berfungsi untuk melihat data yang ada pada maintenance
FEAT3	SKPL_SIMK_F_07	Kelola data pemasukan	Keuangan	Berfungsi untuk memasukkan data dan mengelola data pemasukan
	SKPL_SIMK_F_08	Melihat data pemasukan	keuangan	Berfungsi untuk melihat data yang ada pada pemasukan
	SKPL_SIMK_F_09	Kelola data pengeluaran	keuangan	Berfungsi untuk memasukkan data dan mengelola data pengeluaran
	SKPL_SIMK_F_10	Melihat data pengeluaran	keuangan	Berfungsi untuk melihat data yang ada pada pengeluaran
FEAT4	SKPL_SIMK_F_11	Melihat rekap transaksi dan keuntungan	keuangan	Berfungsi untuk melihat rekap transaksi dan keuntungan dari peternakan
	SKPL_SIMK_F_12	Mencetak laporan	keuangan	Berfungsi untuk mencetak laporan dari data yang sudah dikelola

Berdasarkan empat fitur yang sudah diidentifikasi pada Tabel 4.4, bisa dikelompokkan persyaratan fungsional dari sistem. Tabel di atas menjelaskan hubungan antara fitur dengan persyaratan fungsional sistem. Penjelasan dari setiap persyaratan fungsional dan aktor yang menggunakan fungsi tersebut juga dimasukkan dalam tabel untuk lebih jelas dipahami.

4.3.5 Persyaratan Nonfungsional

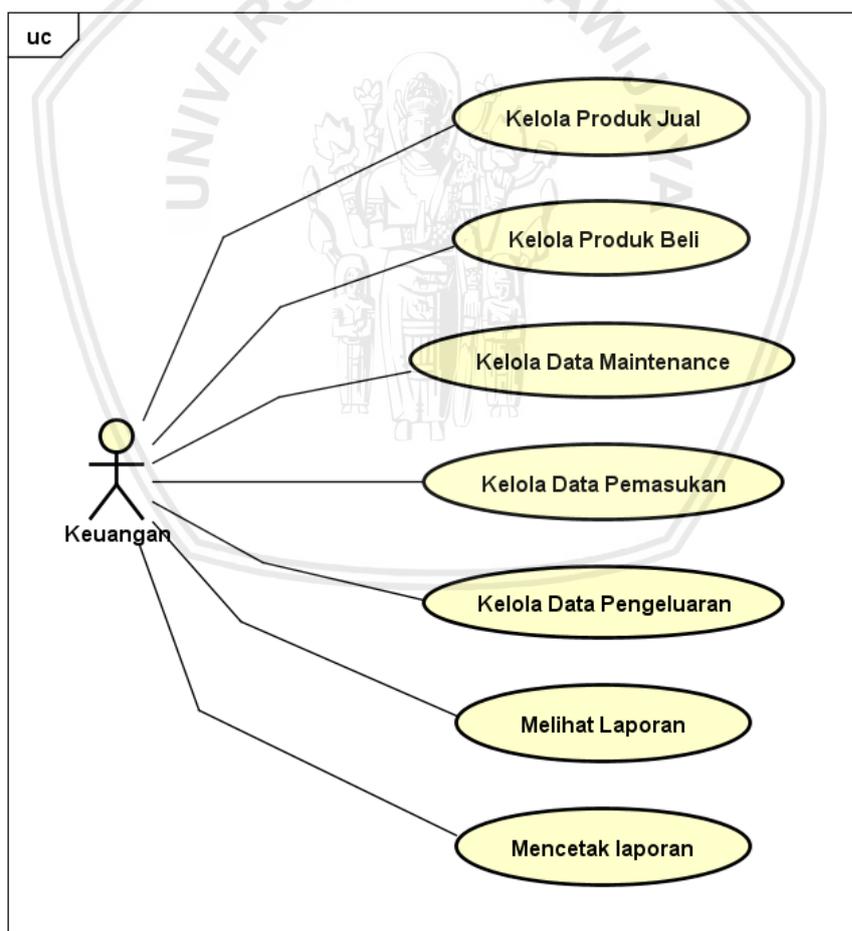
Pada bagian ini menjelaskan analisis kebutuhan nonfungsional yang merupakan batasan layanan yang akan ditawarkan oleh sistem. Semua kebutuhan nonfungsional akan dijelaskan pada Tabel 4.6.

Tabel 4.5 Persyaratan Nonfungsional

Kode Fitur	Kode	Parameter	Deskripsi
FEAT1	SKPL_SIMK_NF_01	<i>Compatibility</i>	Kemampuan sistem untuk bisa diakses pada berbagai <i>browser</i>
FEAT2			
FEAT3			
FEAT4			

Tabel di atas merupakan tabel yang menjelaskan hubungan antara fitur dengan persyaratan nonfungsional. Dalam penelitian ini persyaratan nonfungsional yang digunakan hanya satu yaitu *compatibility*. Dengan mengkodekan persyaratan nonfungsional ini lebih mudah untuk dicari, deskripsi dari persyaratan tersebut juga dimasukkan dengan jelas.

4.4 Pemodelan Use Case



Gambar 4.6 Use Case Sistem Informasi Manajemen Keuangan Yoga’s Farm



Pada Gambar 4.6 menampilkan *use case* sistem informasi manajemen keuangan Yoga's Farm. Pengguna dari sistem tersebut hanya satu yaitu petugas keuangan, petugas keuangan bertugas untuk menjalankan semua fungsi yang ada pada bidang keuangan, mulai dari mengelola data sampai mencetak dan menyimpan data transaksi yang ada pada peternakan.

Pada bagian ini setiap *use case* yang telah teridentifikasi dihubungkan dengan fitur yang sudah teridentifikasi sebelumnya. Hal ini membuktikan bahwa permodelan *use case* sesuai dengan hasil analisis persyaratan. Tabel 4.7 akan menjelaskan hubungan antar *use case* dengan fitur.

Tabel 4.6 Hubungan Use Case dengan Fitur

Use Case	Kode Fitur
Kelola produk jual	FEAT1
Kelola produk beli	
Kelola data maintenance	FEAT2
Kelola data pemasukkan	FEAT3
Kelola data pengeluaran	
Melihat laporan	FEAT4
Mencetak laporan	

Use case dipetakan pada Gambar 4.6 yang menghubungkan antara aktor atau pengguna sistem. Sistem yang sudah jelas fungsinya akan kembali dihubungkan dengan fitur yang sudah diidentifikasi. Penghubungan ini untuk mengetahui fungsi sudah sesuai dengan fitur yang ada.

4.5 Use Case Specification

Pada bagian ini dijelaskan persyaratan fungsional yang lebih terperinci dari persyaratan fungsional yang telah didefinisikan pada sub bab 4.4. Rincian dari persyaratan fungsional akan dijelaskan dalam *use case specification* sebagai berikut.

4.5.1 Use Case Specification Kelola Produk Jual

Berikut penjelasan dari spesifikasi persyaratan fungsional kelola produk jual akan dirinci pada Tabel 4.8.

Tabel 4.7 Use Case Specification Kelola Produk Jual

Kelola Produk Jual (SKPL_SIMK_F_01)	
Brief Description	Berfungsi untuk mengelola data produk jual yaitu memasukkan data penjualan, melihat data penjualan, mengubah data penjualan, dan menghapus data penjualan.
Actor	Petugas Keuangan
Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.

Post-condition	<ol style="list-style-type: none"> 1. Petugas Keuangan berhasil menyimpan hasil pengelolaan data produk jual. 2. Petugas Keuangan melihat hasil pengelolaan data produk jual yang sudah tersimpan pada sistem.
Basic Flow	<p>{use case dimulai}</p> <ol style="list-style-type: none"> 1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk menambahkan data penjualan 2. Sistem menampilkan <i>form</i> data produk jual 3. Lakukan <i>subflow</i> Menambahkan Nama Item Produk Jual 4. Petugas Keuangan memasukkan data produk jual 5. Petugas Keuangan memilih untuk menyimpan data produk jual 6. Sistem menyimpan data produk jual. <p>{menyimpan data produk jual}</p> <ol style="list-style-type: none"> 7. Pesan berhasil disimpan disampaikan oleh sistem. <p>{use case selesai}</p> <ol style="list-style-type: none"> 8. <i>Use case</i> selesai
Alternative Flow	<p>A1. Melihat Rekap Data Produk Jual Pada {melihat rekap data produk jual} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk melihat rekap data produk jual yang tersimpan pada sistem, maka:</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk melihat rekap data produk jual 2. Sistem memuat rekap data produk jual. <p>{memuat rekap data produk jual}</p> <ol style="list-style-type: none"> 3. Sistem menampilkan data rekap produk jual <p>{menampilkan rekap data produk jual}</p> <ol style="list-style-type: none"> 4. <i>Use case</i> selesai <p>A2. Mengubah Data Produk Jual Pada {menampilkan rekap data produk jual} di <i>alternative flow</i>, jika Petugas Keuangan memilih mengubah data produk jual, maka dilakukan <i>subflow</i> Mengubah Data Produk Jual.</p> <p>A3. Menghapus Data Produk Jual Pada {menampilkan rekap data produk jual} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk menghapus data, maka dilakukan <i>subflow</i> Menghapus Data Produk Jual.</p> <p>A4. Menangani Kegagalan Penyimpanan Pada {menyimpan data produk jual} di <i>basic flow</i> dan pada {menyimpan data item} di <i>subflow</i>, jika sistem gagal menyimpan data, maka ditampilkan pesan gagal menyimpan data oleh sistem, kemudian <i>use case</i> selesai.</p>



	<p>A5. Menangani Kegagalan Penghapusan Data Produk Jual Pada {penghapusan data produk jual} di <i>subflow</i>, jika sistem gagal melakukan penghapusan, maka sistem menampilkan pesan gagal hapus data, kemudian <i>use case</i> selesai.</p> <p>A6. Menangani Kegagalan Memuat Data Pada {memuat rekap data produk jual} di <i>alternative flow</i> dan pada {memuat data produk jual} di <i>subflow</i>, jika sistem gagal memuat data, maka sistem menunjukkan pesan gagal memuat data, kemdian <i>use case</i> selesai.</p> <p>A7. Menangani Kegagalan Pengubahan Data Pada {pengubahan data produk jual} di <i>subflow</i>, jika sistem gagal mengubah data, maka sistem menunjukkan pesan data gagal diubah, kemudian <i>use case</i> selesai.</p>
<p>Subflow</p>	<p>S1. Menambah Nama <i>Item</i> Produk Jual</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memasukkan data <i>item</i>. 2. Petugas Keuangan memilih menyimpan data <i>item</i>. 3. Sistem menyimpan data <i>item</i>. <p>{menyimpan data <i>item</i>}</p> <ol style="list-style-type: none"> 4. Sistem menunjukkan pesan data berhasil disimpan. <p>{melihat nama <i>item</i>}</p> <ol style="list-style-type: none"> 5. <i>Use case</i> selsai. <p>S2. Mengubah Data Produk Jual</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk mengubah data produk jual. 2. Sistem memuat data produk jual yang akan diubah. <p>{memuat data produk jual}</p> <ol style="list-style-type: none"> 3. Sistem menunjukkan data produk jual yang akan diubah. 4. Petugas Keuangan mengubah data produk jual. 5. Petugas Keuangan memilih untuk menyimpan data yang sudah diubah. 6. Sistem melakukan pengubahan pada data. <p>{pengubahan data produk jual}</p> <ol style="list-style-type: none"> 7. Sistem menunjukkan pesan data berhasil diubah. 8. <i>Use case</i> selesai. <p>S3. Menghapus Data Produk Beli</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk menghapus data. 2. Sistem melakukan penghapusan data produk jual. <p>{penghapusan data produk jual}</p> <ol style="list-style-type: none"> 3. Sistem menampilkan berhasil menghapus data. 4. <i>Use case</i> selesai.



Use case specification kelola produk jual merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* kelola produk jual, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sub proses yang dilakukan pada sistem, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A" dan sub proses menggunakan kode "S".

4.5.2 Use Case Specification Kelola Produk Beli

Berikut penjelasan dari spesifikasi persyaratan fungsional kelola produk beli akan dirinci pada Tabel 4.9.

Tabel 4.8 Use Case Specification Kelola Produk Beli

Kelola Produk Beli (SKPL_SIMK_F_02)	
Brief Description	Berfungsi untuk mengelola data produk beli yaitu memasukkan data pembelian, melihat data pembelian, mengubah data pembelian, dan menghapus data pembelian.
Actor	Petugas Keuangan
Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.
Post-condition	1. Petugas Keuangan berhasil menyimpan hasil pengelolaan data produk beli. 2. Petugas Keuangan melihat hasil pengelolaan data produk beli yang sudah tersimpan pada sistem.
Basic Flow	<p>{use case dimulai}</p> <ol style="list-style-type: none"> 1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk menambahkan data produk beli 2. Sistem menampilkan <i>form</i> data produk beli 3. Lakukan <i>subflow</i> Menambahkan Nama Item Produk Beli 4. Petugas Keuangan memasukkan data produk Beli 5. Petugas Keuangan memilih untuk menyimpan data produk Beli 6. Sistem menyimpan data produk beli. <p>{menyimpan data produk beli}</p> <ol style="list-style-type: none"> 7. Pesan berhasil disimpan disampaikan oleh sistem. <p>{use case selesai}</p> <ol style="list-style-type: none"> 8. <i>Use case</i> selesai
Alternative Flow	<p>A1. Melihat Rekap Data Produk Beli</p> <p>Pada {melihat rekap data produk beli} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk melihat rekap data produk beli yang tersimpan pada sistem, maka:</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk melihat rekap data produk beli

	<p>2. Sistem memuat rekap data produk beli. {memuat rekap data produk beli}</p> <p>3. Sistem menampilkan data rekap produk beli. {menampilkan rekap data produk beli}</p> <p>4. <i>Use case</i> selesai</p> <p>A2. Mengubah Data Produk Beli Pada {menampilkan rekap data produk beli} di <i>alternative flow</i>, jika Petugas Keuangan memilih mengubah data produk beli, maka dilakukan <i>subflow</i> Mengubah Data Produk Beli.</p> <p>A3. Menghapus Data Produk Beli Pada {menampilkan rekap data produk beli} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk menghapus data, maka dilakukan <i>subflow</i> Menghapus Data Produk Beli.</p> <p>A4. Menangani Kegagalan Penyimpanan Pada {menyimpan data produk beli} di <i>basic flow</i> dan pada {menyimpan data item} di <i>subflow</i>, jika sistem gagal menyimpan data, maka ditampilkan pesan gagal menyimpan data oleh sistem, kemudian <i>use case</i> selesai.</p> <p>A5. Menangani Kegagalan Penghapusan Data Produk Beli Pada {penghapusan data produk beli} di <i>subflow</i>, jika sistem gagal melakukan penghapusan, maka sistem menunjukkan pesan gagal hapus data, kemudian <i>use case</i> selesai.</p> <p>A6. Menangani Kegagalan Memuat Data Pada {memuat rekap data produk beli} di <i>alternative flow</i> dan pada {memuat data produk beli} di <i>subflow</i>, jika sistem gagal memuat data, maka sistem menunjukkan pesan gagal memuat data, kemudian <i>use case</i> selesai.</p> <p>A7. Menangani Kegagalan Pengubahan Data Pada {pengubahan data produk beli} di <i>subflow</i>, jika sistem gagal mengubah data, maka sistem menunjukkan pesan data gagal diubah, kemudian <i>use case</i> selesai.</p>
<p>Subflow</p>	<p>S1. Menambah Nama <i>Item</i> Produk Beli</p> <ol style="list-style-type: none"> Petugas Keuangan memasukkan data <i>item</i>. Petugas Keuangan memilih menyimpan data <i>item</i>. Sistem menyimpan data <i>item</i>. {menyimpan data item} Sistem menunjukkan pesan data berhasil disimpan. {melihat nama item} <i>Use case</i> selsai. <p>S2. Mengubah Data Produk Beli</p>

	<ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk mengubah data produk beli. 2. Sistem memuat data produk beli yang akan diubah. {memuat data produk beli} 3. Sistem menampilkan data produk beli yang akan diubah. 4. Petugas Keuangan mengubah data produk beli. 5. Petugas Keuangan memilih untuk menyimpan data yang sudah diubah. 6. Sistem melakukan perubahan pada data. {pengubahan data produk beli} 7. Sistem menunjukkan pesan data berhasil diubah. 8. <i>Use case</i> selesai. <p>S3. Menghapus Data Produk Beli</p> <ol style="list-style-type: none"> 5. Petugas Keuangan memilih untuk menghapus data. 6. Sistem melakukan penghapusan data produk beli. {penghapusan data produk beli} 7. Sistem menampilkan berhasil menghapus data. 8. <i>Use case</i> selesai.
--	---

Use case specification kelola produk beli merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* kelola produk beli, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sub proses yang dilakukan pada sistem, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A" dan sub proses menggunakan kode "S".

4.5.3 Use Case Specification Kelola Data Maintenance

Berikut penjelasan dari spesifikasi persyaratan fungsional kelola data maintenance akan dirinci pada Tabel 4.10.

Tabel 4.9 Use Case Specification Kelola Data Maintenance

Kelola Data Maintenance (SKPL_SIMK_F_03)	
Brief Description	Berfungsi untuk mengelola data maintenance yaitu memasukkan data maintenance, melihat data maintenance, mengubah data maintenance, dan mengahapus data maintenance.
Actor	Petugas Keuangan
Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.
Post-condition	<ol style="list-style-type: none"> 1. Petugas Keuangan berhasil menyimpan hasil pengelolaan data maintenance. 2. Petugas Keuangan melihat hasil pengelolaan data maintenance yang sudah tersimpan pada sistem.
Basic Flow	{use case dimulai}



	<ol style="list-style-type: none"> 1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk menambahkan data maintenance 2. Sistem menunjukkan <i>form</i> data maintenance 3. Petugas keuangan memasukkan data maintenance 4. Petugas keuangan memilih untuk menyimpan data maintenance 5. Sistem menyimpan data maintenance. {menyimpan data maintenance} 6. Pesan berhasil disimpan disampaikan oleh sistem. {use case selesai} 7. <i>Use case</i> selesai
<p>Alternative Flow</p>	<p>A1. Melihat Rekap Maintenance Pada {melihat rekap maintenance} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk melihat rekap maintenance yang tersimpan pada sistem, maka:</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk melihat rekap maintenance 2. Sistem memuat rekap maintenance. {memuat rekap maintenance} 3. Sistem menampilkan data rekap maintenance. {menampilkan rekap maintenance} 4. <i>Use case</i> selesai <p>A2. Mengubah Data Maintenance Pada {menampilkan rekap maintenance} di <i>alternative flow</i>, jika Petugas Keuangan memilih mengubah data maintenance, maka dilakukan <i>subflows</i> Mengubah Data Maintenance.</p> <p>A3. Menghapus Data Maintenance Pada {menampilkan rekap maintenance} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk menghapus data maintenance, maka dilakukan <i>subflow</i> Menghapus Data Maintenance.</p> <p>A4. Menangani Kegagalan Penyimpanan Pada {menyimpan data maintenance} di <i>basic flow</i>, jika sistem gagal menyimpan data, maka ditampilkan pesan gagal menyimpan data oleh sistem, kemudian <i>use case</i> selesai.</p> <p>A5. Menangani Kegagalan Penghapusan Data Maintenance Pada {penghapusan data maintenance} di <i>subflow</i>, jika sistem gagal melakukan penghapusan, maka sistem menunjukkan pesan gagal menghapus data, kemudian <i>use case</i> selesai.</p> <p>A6. Menangani Kegagalan Memuat Data</p>



	<p>Pada {memuat rekap data maintenance} di <i>alternative flow</i> dan pada {memuat data maintenance} di <i>subflow</i>, jika sistem gagal memuat data, maka sistem menunjukkan pesan gagal memuat data, kemudian <i>use case</i> selesai.</p> <p>A7. Menangani Kegagalan Pengubahan Data</p> <p>Pada {pengubahan data maintenance} di <i>subflow</i>, jika sistem gagal mengubah data, maka sistem menunjukkan pesan data gagal diubah, kemudian <i>use case</i> selesai.</p>
Subflow	<p>S1. Mengubah Data Maintenance</p> <ol style="list-style-type: none"> 1. Petugasa Keuangan memilih untuk megubah data maintenance. 2. Sistem memuat data maintenance yang akan diubah. {memuat data maintenance} 3. Sistem menunjukkan data maintenance yang akan diubah. 4. Petugas Keuangan mengubah data maintenance. 5. Petugas Keuangan memilih untuk menyimpan data yang sudah diubah. 6. Sistem melakukan pengubahan pada data. {pengubahan data produk beli} 7. Sistem menunjukkan pesan data berhasil diubah. 8. <i>Use case</i> selesai. <p>S2. Menghapus Data Maintenance</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk menghapus data. 2. Sistem melakukan penghapusan data maintenance. {penghapusan data maintenance} 3. Sistem menampilkan pesan berhasil menghapus data. 4. <i>Use case</i> selesai.

Use case specification kelola data *maintenance* merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* kelola data *maintenance*, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sub proses yang dilakukan pada sistem, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A" dan sub proses menggunakan kode "S".

4.5.4 Use Case Specification Kelola Data Pemasukan

Berikut penjelasan dari spesifikasi persyaratan fungsional kelola data pemasukan akan dirinci pada Tabel 4.11.

Tabel 4.10 Use Case Specification Kelola Data Pemasukkan

Kelola Data Pemasukkan (SKPL_SIMK_F_04)	
Brief Description	Berfungsi untuk mengelola data maintenance yaitu memasukkan data pemasukkan, melihat data

	pemasukkan, mengubah data pemasukkan, dan menghapus data pemasukkan.
Actor	Petugas Keuangan
Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.
Post-condition	1. Petugas Keuangan berhasil menyimpan hasil pengelolaan data pemasukkan. 2. Petugas Keuangan melihat hasil pengelolaan data pemasukkan yang sudah tersimpan pada sistem.
Basic Flow	<p>{use case dimulai}</p> <ol style="list-style-type: none"> 1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk menambahkan data pemasukkan 2. Sistem menampilkan <i>form</i> data pemasukkan 3. Petugas keuangan memasukkan data pemasukkan 4. Petugas keuangan memilih untuk menyimpan data pemasukkan 5. Sistem menyimpan data pemasukkan. <p>{menyimpan data pemasukkan}</p> <ol style="list-style-type: none"> 6. Pesan berhasil disimpan disampaikan oleh sistem. <p>{use case selesai}</p> <ol style="list-style-type: none"> 7. <i>Use case</i> selesai
Alternative Flow	<p>A1. Melihat Rekap Pemasukkan</p> <p>Pada {melihat rekap pemasukkan} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk melihat rekap pemasukkan yang tersimpan pada sistem, maka:</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk melihat rekap pemasukkan 2. Sistem memuat rekap pemasukkan. <p>{memuat rekap pemasukkan}</p> <ol style="list-style-type: none"> 3. Sistem menampilkan data rekap pemasukkan. <p>{menampilkan rekap pemasukkan}</p> <ol style="list-style-type: none"> 4. <i>Use case</i> selesai <p>A2. Mengubah Data Pemasukkan</p> <p>Pada {menampilkan rekap pemasukkan} di <i>alternative flow</i>, jika Petugas Keuangan memilih mengubah data</p>



	<p>pemasukkan, maka dilakukan <i>subflow</i> Mengubah Data Pemasukkan.</p> <p>A3. Menghapus Data Pemasukkan</p> <p>Pada {menampilkan rekap pemasukkan} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk menghapus data pemasukkan, maka dilakukan <i>subflow</i> Menghapus Data Pemasukkan.</p> <p>A4. Menangani Kegagalan Penyimpanan</p> <p>Pada {menyimpan data pemasukkan} di <i>basic flow</i>, jika sistem gagal menyimpan data, maka ditampilkan pesan gagal menyimpan data oleh sistem, kemudian <i>use case</i> selesai.</p> <p>A5. Menangani Kegagalan Penghapusan Data Pemasukkan</p> <p>Pada {penghapusan data pemasukkan} di <i>subflow</i>, jika sistem gagal melakukan penghapusan, maka sistem menampilkan pesan gagal menghapus data, kemudian <i>use case</i> selesai.</p> <p>A6. Menangani Kegagalan Memuat Data</p> <p>Pada {memuat rekap pemasukkan} di <i>alternative flow</i> dan pada {memuat data pemasukkan} di <i>subflow</i>, jika sistem gagal memuat data, maka sistem menunjukkan pesan gagal memuat data, kemudian <i>use case</i> selesai.</p> <p>A7. Menangani Kegagalan Perubahan Data</p> <p>Pada {perubahan data pemasukkan} di <i>subflow</i>, jika sistem gagal mengubah data, maka sistem menunjukkan pesan data gagal diubah, kemudian <i>use case</i> selesai.</p>
<p>Subflow</p>	<p>S1. Mengubah Data Pemasukkan</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk mengubah data pemasukkan. 2. Sistem memuat data pemasukkan yang akan diubah. {memuat data pemasukkan} 3. Sistem menunjukkan data pemasukkan yang akan diubah. 4. Petugas Keuangan mengubah data pemasukkan. 5. Petugas Keuangan memilih untuk menyimpan data yang sudah diubah.



	<p>6. Sistem melakukan perubahan pada data. {perubahan data pemasukkan}</p> <p>7. Sistem menunjukkan pesan data berhasil diubah.</p> <p>8. <i>Use case</i> selesai.</p> <p>S2. Menghapus Data Pemasukkan</p> <p>1. Petugas Keuangan memilih untuk menghapus data.</p> <p>2. Sistem melakukan penghapusan data pemasukkan. {penghapusan data pemasukkan}</p> <p>3. Sistem menampilkan pesan berhasil menghapus data.</p> <p>4. <i>Use case</i> selesai.</p>
--	---

Use case specification kelola data pemasukkan merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* kelola data pemasukkan, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sub proses yang dilakukan pada sistem, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A" dan sub proses menggunakan kode "S".

4.5.5 Use Case Specification Kelola Data Pengeluaran

Berikut penjelasan dari spesifikasi persyaratan fungsional kelola data pengeluaran akan dirinci pada Tabel 4.12.

Tabel 4.11 Use Case Specification Kelola Data Pengeluaran

Kelola Data Pengeluaran (SKPL_SIMK_F_05)	
Brief Description	Berfungsi untuk mengelola data maintenance yaitu memasukkan data pengeluaran, melihat data pengeluaran, mengubah data pengeluaran, dan menghapus data pengeluaran.
Actor	Petugas Keuangan
Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.
Post-condition	1. Petugas Keuangan berhasil menyimpan hasil pengelolaan data pengeluaran. 2. Petugas Keuangan melihat hasil pengelolaan data pengeluaran yang sudah tersimpan pada sistem.
Basic Flow	<p>{use case dimulai}</p> <p>1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk menambahkan data pengeluaran</p> <p>2. Sistem menampilkan <i>form</i> data pengeluaran</p> <p>3. Petugas keuangan memasukkan data pengeluaran</p>



	<ol style="list-style-type: none"> 4. Petugas keuangan memilih untuk meyimpan data pengeluaran 5. Sistem menyimpan data pemasukkan. {menyimpan data pengeluaran} 6. Pesan berhasil disimpan disampaikan oleh sistem. {use case selesai} 7. <i>Use case</i> selesai
<p>Alternative Flow</p>	<p>A1. Melihat Rekap Pengeluaran Pada {melihat rekap pengeluaran} di <i>alternative flow</i>, jika Petugas Keuangan mmilih untuk melihat rekap pemasukkan yang tersimpan pada sistem, maka:</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk melihat rekap pengeluaran 2. Sistem memuat rekap pengeluaran. {memuat rekap pengaluaran} 3. Sistem menampilkan data rekap pengeluaran. {menampilkan rekap pengeluaran} 4. <i>Use case</i> selesai <p>A2. Mengubah Data Pengeluaran Pada {menampilkan rekap pengeluaran} di <i>alternative flow</i>, jika Petugas Keuangan memilih mengubah data pengeluaran, maka dilakukan <i>subflow</i> Mengubah Data Pengeluaran.</p> <p>A3. Menghapus Data Pengeluaran Padaa {menampilkan rekap pengeluaran} di <i>alternative flow</i>, jika Petugas Keuangan memilih untuk menghapus data pengeluaran, maka dilakukan <i>subflow</i> Menghapus Data Pengeluaran.</p> <p>A4. Menangani Kegagalan Penyimpanan Pada {menyimpan data pengeluaran} di <i>basic flow</i>, jika sistem gagal menyimpan data, maka ditampilkan pesan gagal menyimpan data oleh sistem, kemudian <i>use case</i> selesai.</p> <p>A5. Menangani Kegagalan Penghapusan Data Pemasukkan Pada {penghapusan data pengeluaran} di <i>subflow</i>, jika sistem gagal melakukan penghapusan, maka sistem menunjukkan pesan gagal menghapus data, kemudian <i>use case</i> selesai.</p> <p>A6. Menangani Kegagalan Memuat Data Pada {memuat rekap pengeluaran} di <i>alternative flows</i> dan pada {memuat data pengeluaran} di <i>subflow</i>, jika sistem gagal memuat data, maka sistem menunjukkan pesan gagal memuat data, kemudian <i>use case</i> selesai.</p>



	<p>A7. Menangani Kegagalan Pengubahan Data Pada {pengubahan data pengeluaran} di <i>subflow</i>, jika sistem gagal mengubah data, maka sistem menunjukkan pesan data gagal diubah, kemudian <i>use case</i> selesai.</p>
Subflow	<p>S1. Mengubah Data Pengeluaran</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk megubah data pengeluaran. 2. Sistem memuat data pengeluaran yang akan diubah. {memuat data pengeluaran} 3. Sistem menunjukkan data pengeluaran yang akan diubah. 4. Petugas Keuangan megubah data pengeluaran. 5. Petugas Keuangan memilih untuk menyimpan data yang sudah diubah. 6. Sistem melakukan pengubahan pada data. {pengubahan data pengeluaran} 7. Sistem menunjukkan pesan data berrhasil diubah. 8. <i>Use case</i> selesai. <p>S2. Menghapus Data Pemasukkan</p> <ol style="list-style-type: none"> 1. Petugas Keuangan memilih untuk menghapus data. 2. Sistem melakukan penghapusan data pengeluaran. {penghapusan data pemasukkan} 3. Sistem menunjukkan pesan berhasil menghapus data. 4. <i>Use case</i> selesai.

Use case specification kelola data pengeluaran merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* kelola data pengeluaran, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sub proses yang dilakukan pada sistem, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A" dan sub proses menggunakan kode "S".

4.5.6 Use Case Specification Melihat Laporan

Berikut penjelasan dari spesifikasi persyaratan fungsional melihat laporan akan dirinci pada Tabel 4.13.

Tabel 4.12 Use Case Specification Melihat Laporan

Melihat Laporan (SKPL_SIMK_F_06)	
Brief Description	Berfungsi untuk melihat laporan yang berisi data hasil rekap dari semua kegiatan keuangan (penjualan, pembelian, pemasukkan, pengeluaran, dan maintenance) serta menunjukkan keuntungan yang didapatkan dalam setiap bulan.
Actor	Petugas Keuangan



Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.
Post-condition	1. Petugas Keuangan melihat hasil pengelolaan data yang sudah dijadikan dalam bentuk laporan keuangan
Basic Flow	<p>{use case dimulai}</p> <ol style="list-style-type: none"> 1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk melihat laporan keuangan 2. Sistem memuat data laporan <p>{memuat laporan}</p> <ol style="list-style-type: none"> 3. Sistem menampilkan rekap keuangan dari penjualan, pembelian, maintenance, pemasukkan, pengeluaran, dan keuntungan 4. <i>Use case</i> selesai
Alternative Flow	<p>A1. Menangani Kegagalan Memuat Data</p> <p>Pada {memuat laporan} di <i>basic flow</i>, jika sistem gagal memuat laporan, maka sistem menunjukkan pesan gagal memuat laporan, kemudian <i>use case</i> selesai.</p>

Use case specification melihat laporan merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* melihat laporan, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A".

4.5.7 Use Case Specification Mencetak Laporan

Berikut penjelasan dari spesifikasi persyaratan fungsional kelola produk jual akan dirinci pada Tabel 4.14.

Tabel 4.13 Use Case Specification Mencetak Laporan

Mencetak Laporan (SKPL_SIMK_F_07)	
Brief Description	Berfungsi untuk mencetak semua data rekap (penjualan, pembelian, maintenance, pemasukkan, dan pengeluaran) dalam bentuk laporan rapi.
Actor	Petugas Keuangan
Pre-condition	1. Petugas Keuangan berhasil masuk ke dalam sistem.
Post-condition	1. Petugas Keuangan mencetak laporan secara lengkap dari semua kegiatan keuangan
Basic Flow	<p>{use case dimulai}</p> <ol style="list-style-type: none"> 1. <i>Use case</i> dimulai pada saat Petugas Keuangan memilih untuk mencetak laporan 2. Sistem memuat <i>review</i> dari laporan yang akan dicetak <p>{memuat review}</p> <ol style="list-style-type: none"> 3. Petugas Keuangan memilih untuk mencetak laporan 4. Sistem mencetak laporan 5. <i>Use case</i> selesai

Alternative Flow	A1. Menangani Kegagalan Memuat Review Pada {memuat <i>review</i> } di <i>basic flow</i> , jika sistem gagal memuat laporan, maka sistem menampilkan pesan gagal memuat laporan, kemudian <i>use case</i> selesai.
-------------------------	---

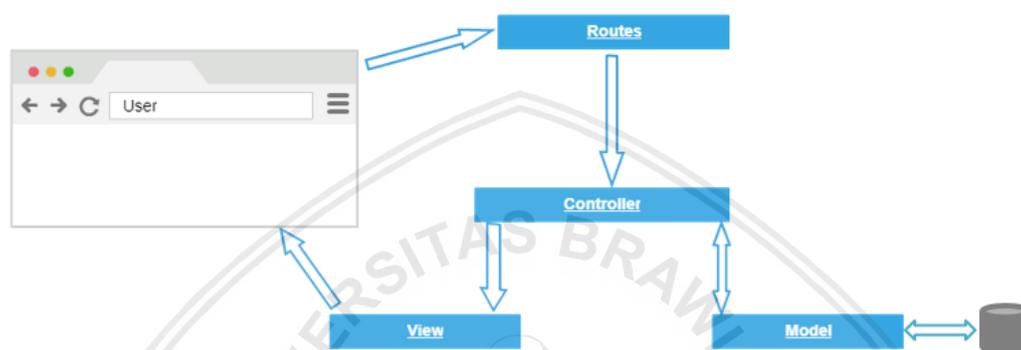
Use case specification mencetak laporan merupakan penjelasan dari pemetaan *use case*. Penjelasan ini lebih terperinci secara proses alur berjalannya sistem. Mulai dari penjelasan tentang *use case* mencetak laporan, kondisi awal sistem saat mulai digunakan, kondisi yang diharapkan, alur dasar yang dilakukan dalam satu *use case*, sampai alternatif yang disediakan sistem untuk menangani semua masalah kesalahan baik dari pengguna maupun internal sistem. Semua dijelaskan dengan terperinci dengan kode alternatif menggunakan "A".



BAB 5 PERANCANGAN

5.1 Perancangan Arsitektur

Perancangan arsitektur sistem informasi manajemen keuangan peternakan kambing. Pada Gambar 5.1 digambarkan arsitektur berjalannya sistem informasi manajemen keuangan, dari *User* menuju pada *Routes*, *Routes* memanggil *Controller* yang berasangkutan dan menghubungkan dengan *Model* yang mengacu pada *database*, dan *Controller* menghubungkan dengan *View* yang akan menampilkan pada *User*.



Gambar 5.1 Perancangan Arsitektur Sistem Informasi Manajemen Keuangan

5.2 Pemodelan Interaksi Objek

Pada sub bagian ini akan dilakukan pemodelan interaksi yang terjadi antar objek dalam sistem. Interaksi sistem terjadi pertukaran pesan antara entitas aktor, *routes*, entitas *boundary*, objek *control*, dan objek *model* yang berinteraksi satu dengan yang lain untuk memenuhi kebutuhan pengguna sistem.

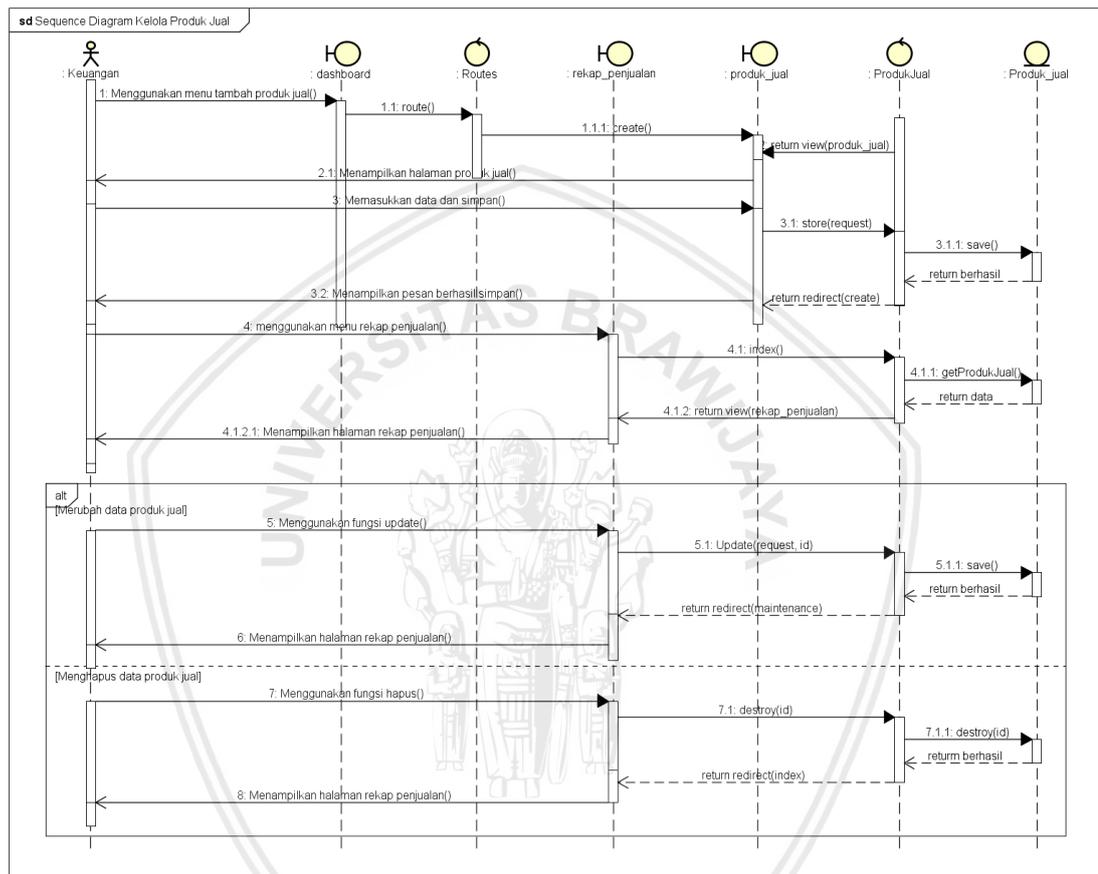
5.2.1.1 Sequence Diagram Kelola Produk Jual

Sequences diagram pada Gambar 5.2 merupakan gambaran dari interaksi antara objek ketika aktor keuangan menggunakan sistem kelola produk jual. Objek yang terlibat pada interaksi untuk mengelola produk jual adalah aktor keuangan, *dashboard*, *rekap_penjualan*, dan *produk_jual* sebagai objek *boundary*, *routes* sebagai objek rute pada laravel, kelas *ProdukJual* sebagai objek *control*, dan kelas *Produk_jual* sebagai objek *model*.

Interaksi dimulai ketika aktor keuangan mengirim pesan pada sistem melalui objek *boundary dashboard*. Kemudian objek *boundary* meneruskan pesan pada *routes* yang mengarahkan pada kelas *ProdukJual* sebagai objek *control*. Selanjutnya, objek *control* mengembalikan pada objek *boundary produk_jual* untuk menampilkan halaman produk jual.

Pada halaman kelola produk jual terdapat 4 fungsi yang dapat dilakukan oleh pengguna sistem yaitu menambahkan, melihat, mengubah, dan menghapus data produk jual. Jika pengguna memasukkan produk jual melalui *boundary produk_jual* dan diteruskan ke objek *control ProdukJual*, maka *ProdukJual* sebagai *control* akan meneruskan ke objek *model* untuk menyimpan data. Jika pengguna

memilih melihat rekap penjualan melalui *boundary* rekap_penjualan, ProdukJual sebagai *control* akan mengambil dari *database* melalui *model* Produk_jual untuk ditampilkan kepada pengguna. Jika pengguna memilih mengubah data melalui *boundary* rekap_penjualan, ProdukJual sebagai *control* akan mengubah data melalui dari *database* melalui *model* Produk_jual. Dan jika pengguna memilih menghapus data melalui *boundary* rekap_penjualan, ProdukJual sebagai *control* akan menghapus data dari *database* melalui *model* Produk_jual. Berikut *sequence diagram* kelola produk jual dapat dilihat pada Gambar 5.2.



Gambar 5.2 Sequence Diagram Kelola Produk Jual

5.2.1.2 Sequence Diagram Kelola Produk Beli

Sequence diagram pada Gambar 5.3 merupakan gambarn dari interaksi antar objek ketika aktor keuangan menggunakan sistem kelola produk beli. Objek yang terlibat pada interaksi untuk mengelola produk beli adalah aktor keuangan, *dashboard*, rekap_pembelian, dan produk_beli sebagai objek *boundary*, *routes* sebagai objek rute pada laravel, kelas ProdukBeli sebagai objek *control*, dan kelas Produk_beli sebagai objek model.

Interaksi dimulai ketika aktor keuangan mengirim pesan pada sistem melalui objek *boundary dashboard*. Kemudian objek *boundary* meneruskan pesan pada *routes* yang mengarahkan pada kelas ProdukBeli sebagai objek *control*.

Selanjutnya, objek *control* mengembalikan pada objek *boundary* produk_beli untuk menampilkan halaman produk beli.

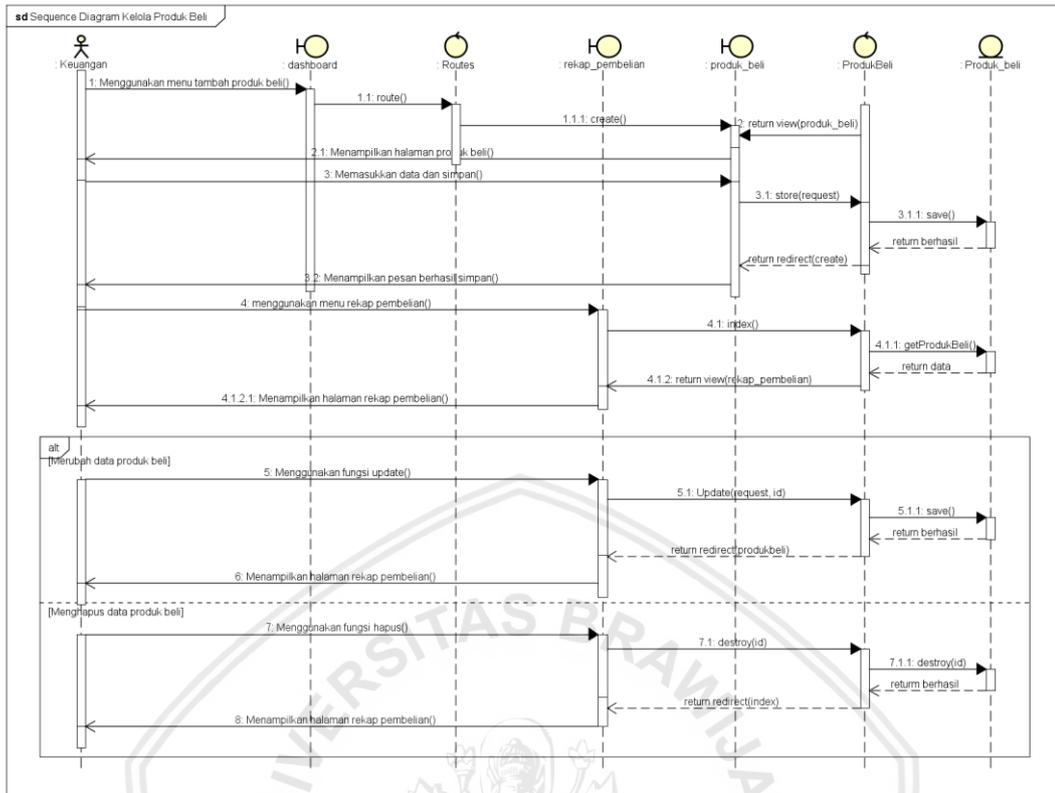
Pada halaman kelola produk beli terdapat 4 fungsi yang dapat dilakukan oleh pengguna sistem yaitu menambahkan, melihat, mengubah, dan menghapus data produk beli. Jika pengguna memasukkan produk beli melalui *boundary* produk_beli dan diteruskan ke objek *control* ProdukBeli, maka ProdukBeli sebagai *control* akan meneruskan ke objek *model* untuk menyimpan data. Jika pengguna memilih melihat rekap pembelian melalui *boundary* rekap_pembelian, ProdukBeli sebagai *control* akan mengambil dari *database* melalui *model* Produk_beli untuk ditampilkan kepada pengguna. Jika pengguna memilih mengubah data melalui *boundary* rekap_pembelian, ProdukBeli sebagai *control* akan mengubah data melalui dari *database* melalui *model* Produk_beli. Dan jika pengguna memilih menghapus data melalui *boundary* rekap_pembelian, ProdukBeli sebagai *control* akan menghapus data dari *database* melalui *model* Produk_beli. Berikut *sequence diagram* kelola produk beli dapat dilihat pada Gambar 5.3.

5.2.1.3 Sequence Diagram Kelola Data Maintenance

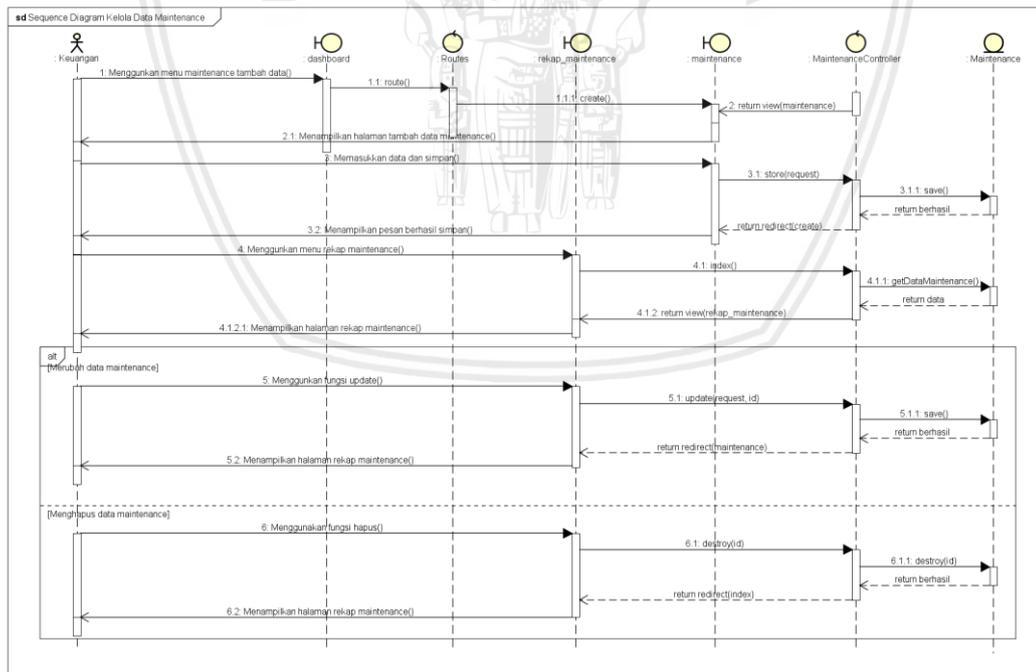
Sequence diagram pada Gambar 5.4 merupakan gambaran dari interaksi antar objek ketika aktor keuangan menggunakan sistem kelola data maintenance. Objek yang terlibat pada interaksi untuk mengelola data maintenance adalah aktor keuangan, *dashboard*, rekap_maintenance, dan maintenance sebagai objek *boundary*, *routes* sebagai objek rute pada laravel, kelas MaintenanceController sebagai objek *control*, dan kelas Maintenance sebagai objek model.

Interaksi dimulai ketika aktor keuangan mengirim pesan pada sistem melalui objek *boundary dashboard*. Kemudian objek *boundary* meneruskan pesan pada *routes* yang mengarahkan pada kelas MaintenanceController sebagai objek *control*. Selanjutnya, objek *control* mengembalikan pada objek *boundary* maintenance untuk menampilkan halaman maintenance.

Pada halaman kelola data maintenance terdapat 4 fungsi yang dapat dilakukan oleh pengguna sistem yaitu menambahkan, melihat, mengubah, dan menghapus data maintenance. Jika pengguna memasukkan maintenance melalui *boundary* maintenance dan diteruskan ke objek *control* MaintenanceController, maka MaintenanceController sebagai *control* akan meneruskan ke objek *model* untuk menyimpan data. Jika pengguna memilih melihat rekap pembelian melalui *boundary* rekap_maintenance, MaintenanceController sebagai *control* akan mengambil dari *database* melalui *model* Maintenance untuk ditampilkan kepada pengguna. Jika pengguna memilih mengubah data melalui *boundary* rekap_maintenance, MaintenanceController sebagai *control* akan mengubah data melalui dari *database* melalui *model* Maintenance. Dan jika pengguna memilih menghapus data melalui *boundary* rekap_maintenance, MaintenanceController sebagai *control* akan menghapus data dari *database* melalui *model* Maintenance. Berikut *sequence diagram* kelola data maintenance dapat dilihat pada Gambar 5.4.



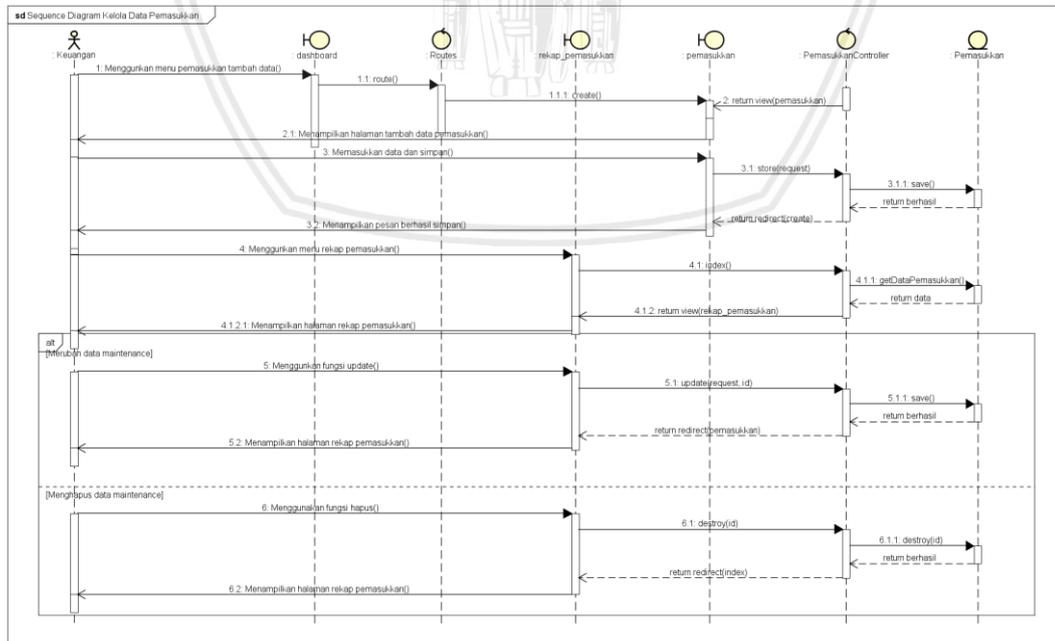
Gambar 5.3 Sequence Diagram Kelola Produk Beli



Gambar 5.4 Sequence Diagram Kelola Data Maintenance

5.2.1.4 Sequence Diagram Kelola Data Pemasukkan

Sequences diagram pada Gambar 5.5 merupakan gambaran dari interaksi antar obyek ketika aktor keuangan menggunakan sistem kelola data pemasukkan. Obyek yang terlibat pada interaksi untuk mengelola data pemasukkan adalah aktor keuangan, *dashboard*, *rekap_pemasukkan*, dan *pemasukkan* sebagai objek *boundary*, *routes* sebagai objek rute pada laravel, kelas *PemasukkanController* sebagai objek *control*, dan kelas *Pemasukkan* sebagai objek model. Interaksi dimulai ketika aktor keuangan mengirim pesan pada sistem melalui objek *boundary dashboard*. Kemudian objek *boundary* meneruskan pesan pada *routes* yang mengarahkan pada kelas *PemasukkanController* sebagai objek *control*. Selanjutnya, objek *control* mengembalikan pada objek *boundary* pemasukkan untuk menampilkan halaman pemasukkan. Pada halaman kelola data pemasukkan terdapat 4 fungsi yang dapat dilakukan oleh pengguna sistem yaitu menambahkan, melihat, mengubah, dan menghapus data pemasukkan. Jika pengguna memasukkan data pemasukkan melalui *boundary* pemasukkan dan diteruskan ke objek *control* *PemasukkanController*, maka *PemasukkanController* sebagai *control* akan meneruskan ke objek *model* untuk menyimpan data. Jika pengguna memilih melihat rekap pemasukkan melalui *boundary* *rekap_pemasukkan*, *PemasukkanController* sebagai *control* akan mengambil dari *database* melalui *model* *Pemasukkan* untuk ditampilkan kepada pengguna. Jika pengguna memilih mengubah data melalui *boundary* *rekap_pemasukkan*, *PemasukkanController* sebagai *control* akan mengubah data melalui dari *database* melalui *model* *Pemasukkan*. Dan jika pengguna memilih menghapus data melalui *boundary* *rekap_pemasukkan*, *PemasukkanController* sebagai *control* akan menghapus data dari *database* melalui *model* *Pemasukkan*. Berikut *sequence diagram* kelola data pemasukkan dapat dilihat pada Gambar 5.5.



Gambar 5.5 Sequence Diagram Kelola Data Pemasukkan

5.2.1.5 Sequence Diagram Kelola Data Pengeluaran

Sequences diagram pada Gambar 5.6 merupakan gambaran dari interaksi antara objek ketika aktor keuangan menggunakan sistem kelola data pengeluaran. Objek yang terlibat pada interaksi untuk mengelola data pengeluaran adalah aktor keuangan, *dashboard*, *rekap_pengeluaran*, dan *pengeluaran* sebagai objek *boundary*, *routes* sebagai objek rute pada laravel, kelas *PengeluaranController* sebagai objek *control*, dan kelas *Pengeluaran* sebagai objek model.

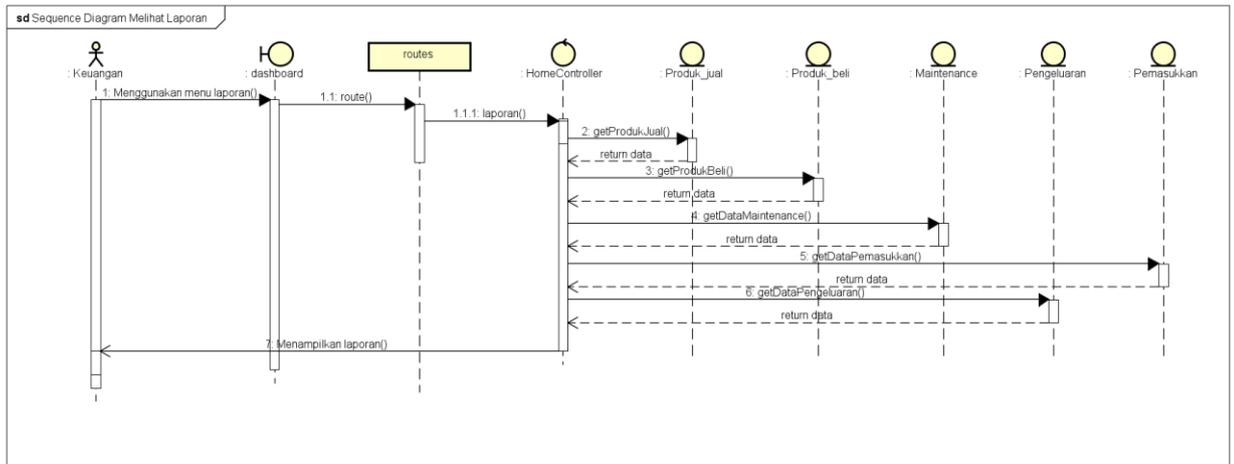
Interaksi dimulai ketika aktor keuangan mengirim pesan pada sistem melalui objek *boundary dashboard*. Kemudian objek *boundary* meneruskan pesan pada *routes* yang mengarahkan pada kelas *PengeluaranController* sebagai objek *control*. Selanjutnya, objek *control* mengembalikan pada objek *boundary* pengeluaran untuk menampilkan halaman pengeluaran.

Pada halaman kelola data pengeluaran terdapat 4 fungsi yang dapat dilakukan oleh pengguna sistem yaitu menambahkan, melihat, mengubah, dan menghapus data pengeluaran. Jika pengguna memasukkan data pemasukkan melalui *boundary* pengeluaran dan diteruskan ke objek *control* *PengeluaranController*, maka *PengeluaranController* sebagai *control* akan meneruskan ke objek *model* untuk menyimpan data. Jika pengguna memilih melihat rekap pengeluaran melalui *boundary* *rekap_pengeluaran*, *PengeluaranController* sebagai *control* akan mengambil dari *database* melalui *model* *Pengeluaran* untuk ditampilkan kepada pengguna. Jika pengguna memilih mengubah data melalui *boundary* *rekap_pengeluaran*, *PengeluaranController* sebagai *control* akan mengubah data melalui dari *database* melalui *model* *Pengeluaran*. Dan jika pengguna memilih menghapus data melalui *boundary* *rekap_pengeluaran*, *PengeluaranController* sebagai *control* akan menghapus data dari *database* melalui *model* *Pengeluaran*. Berikut *sequence diagram* kelola data pengeluaran dapat dilihat pada Gambar 5.6.

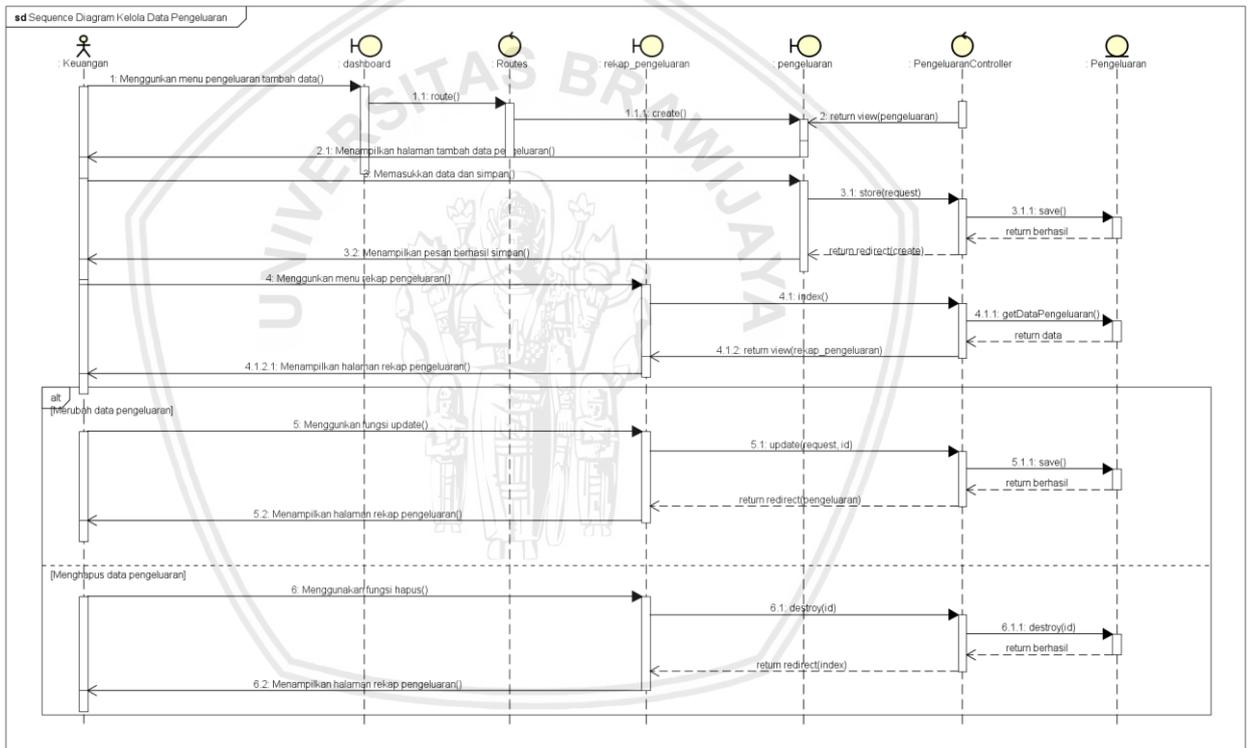
5.2.1.6 Sequence Diagram Melihat Laporan

Sequences diagram pada Gambar 5.7 merupakan gambaran dari interaksi antara objek ketika aktor keuangan menggunakan sistem melihat laporan. Objek yang terlibat pada interaksi untuk melihat laporan adalah aktor keuangan, *dashboard*, *routes* sebagai objek rute pada laravel, kelas *HomeController* sebagai objek *control*, dan kelas *Produk_jual*, *Produk_beli*, *Maintenance*, *Pemasukkan*, dan *Pengeluaran* sebagai objek model.

Interaksi dimulai ketika aktor keuangan mengirim pesan pada sistem melalui objek *boundary dashboard*. Kemudian objek *boundary* meneruskan pesan pada *routes* yang mengarahkan pada kelas *HomeController* sebagai objek *control*. Selanjutnya, objek *control* mengambil data dari *database* melalui beberapa *model* yaitu *Produk_jual*, *Produk_beli*, *Maintenance*, *Pemasukkan*, dan *Pengeluaran*. Data yang diambil dikembalikan pada *HomeController* sebagai *control* dan ditampilkan pada pengguna. Berikut *sequence diagram* melihat laporan dapat dilihat pada Gambar 5.7.



Gambar 5.6 Sequence Diagram Melihat Laporan

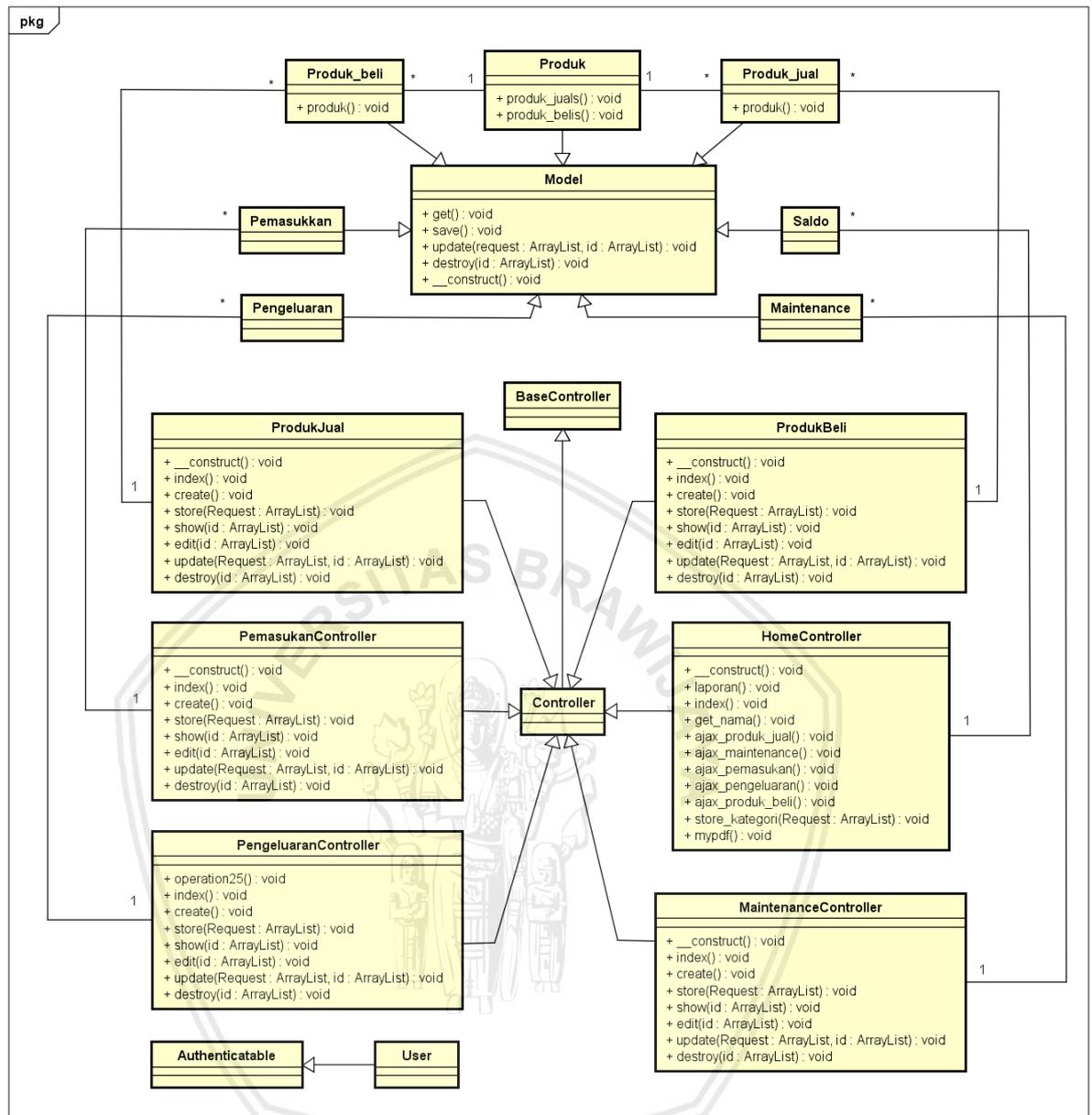


Gambar 5.7 Sequence Diagram Kelola Data Pengeluaran

5.2.2 Class Diagram

Pada Gambar 5.8 digambarkan hubungan antar setiap kelas yang disertai dengan fungsi-fungsi yang berada pada kelas tersebut. Dalam kelas diagram ini dijelaskan bahwa kelas pada sistem informasi manajemen keuangan peternakan kambing menggunakan abstraksi dari kelas *Model*, *BaseController*, dan *Authenticatable*. Kelas tersebut merupakan kelas abstraksi dari *Framework Laravel* yang digunakan pada sistem informasi manajemen keuangan peternakan kambing. Semua kelas abstraksi tersebut saling berhubungan satu dengan yang lainnya, sehingga dapat digambarkan pada Gambar 5.8 berikut.

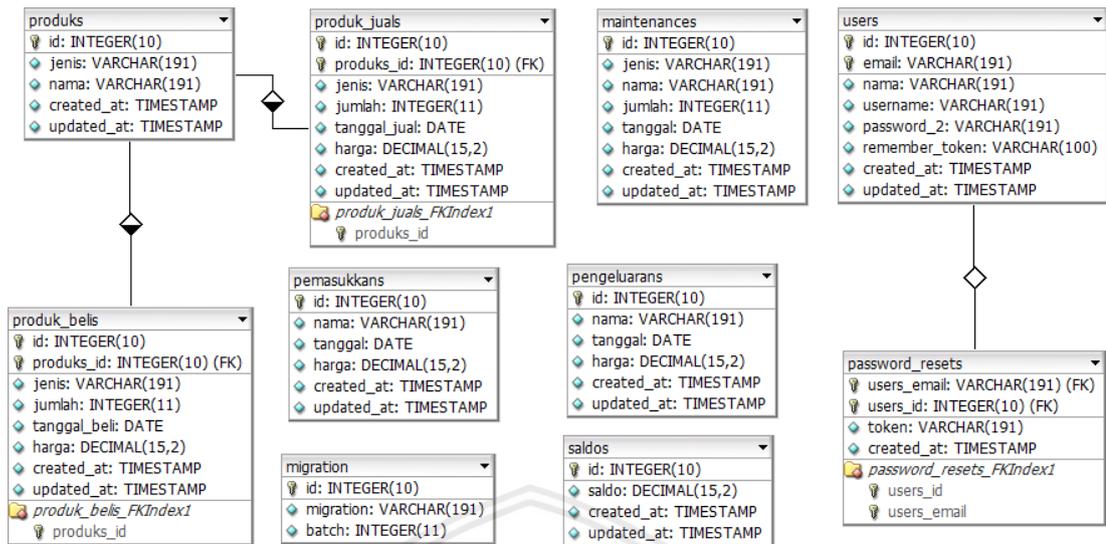




Gambar 5.8 Class Diagram Sistem Informasi Manajemen Keuangan

5.3 Perancangan Database

Database merupakan sebuah tempat yang berguna untuk penyimpanan data. *Database* yang digunakan pada sistem manajemen keuangan kambing menggunakan MySQL. Perancangan *database* ini digunakan merancang *database* data dalam manajemen keuangan agar masukan dan keluaran data bisa sesuai dengan kebutuhan pengguna. Gambar 5.9 menggambarkan rancangan *database* pada sistem informasi manajemen keuangan peternakan kambing.



Gambar 5.9 Rancangan Database

Berikut masing-masing struktur data pada database yang berupa tabel akan dijelaskan pada tabel-tabel berikut.

Tabel 5.1 Struktur Tabel Produk

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel produk
2.	jenis	varchar	191	Jenis barang yang meliputi pakan, obat, dan kambing
3.	nama	varchar	191	Nama barang dari setiap jenis
4.	Created_at	timestamp	-	Waktu pemasukkan data
5.	Updated_at	timestamp	-	Waktu pengubahan data

Tabel 5.2 Struktur Tabel produk_juals

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel produk jual
2.	jenis	varchar	191	Jenis barang yang meliputi pakan, obat, dan kambing



3.	Produk_id	integer	11	Produk_id yang menjadi id nama pada produk jual
4.	Jumlah	integer	11	Jumlah barang produk jual
5.	Tanggal_jual	date	-	Tanggal penjualan produk
6.	Harga	decimal	15,2	Harga produk
7.	Created_at	timestamp	-	Waktu pemasukkan data
8.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.3 Struktur Tabel Produk_belis

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel produk beli
2.	jenis	varchar	191	Jenis barang yang meliputi pakan, obat, dan kambing
3.	Produk_id	integer	11	Produk_id yang menjadi id nama pada produk beli
4.	Jumlah	integer	11	Jumlah barang produk beli
5.	Tanggal_jual	date	-	Tanggal penjualan produk
6.	Harga	decimal	15,2	Harga produk
7.	Created_at	timestamp	-	Waktu pemasukkan data
8.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.4 Struktur Tabel Pemasukkans

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel pemasukkans
2.	Nama	varchar	191	Nama pemasukkan
3.	Tanggal	date	-	Tanggal pemasukkan



4.	Harga	decimal	15,2	Harga/jumlah pemasukkan
5.	Created_at	timestamp	-	Waktu pemasukkan data
6.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.5 Struktur Tabel Pengeluaran

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel pengeluaran
2.	Nama	varchar	191	Nama pengeluaran
3.	Tanggal	date	-	Tanggal pengeluaran
4.	Harga	decimal	15,2	Harga/jumlah pengeluaran
5.	Created_at	timestamp	-	Waktu pemasukkan data
6.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.6 Struktur Tabel *Maintenances*

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel maintenance
2.	jenis	varchar	191	Jenis barang yang meliputi alat, kandang dll.
3.	Nama	varchar	191	Nama dari kebutuhan maintenance
4.	Jumlah	integer	11	Jumlah barang produk beli
5.	Tanggal	date	-	Tanggal maintenance
6.	Harga	decimal	15,2	Harga produk
7.	Created_at	timestamp	-	Waktu pemasukkan data
8.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.7 Struktur Tabel Saldos

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel saldus
2.	saldo	decimal	15,2	Saldo dari keuangan peterakan
3.	Created_at	timestamp	-	Waktu pemasukkan data
4.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.8 Struktur Tabek *Password_reset*

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	Email	Varchar	191	Email dari pengguna sistem
2.	Token	Varchar	191	Peringatan jika melakukan kesalahan ketika masuk dalam sistem
3.	Created_at	timestamp	-	Waktu pemasukkan data

Tabel 5.9 Struktur Tabel *Users*

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel users
2.	Name	Varchar	191	Nama dari pengguna sistem
3.	Email	Varchar	191	Email dari pengguna sistem
4.	Username	Varchar	191	Username yang digunakan pengguna sistem untuk masuk dalam sistem
5.	Password	Varchar	191	Password pengguna untuk masuk dalam sistem
6.	Remember_token	Varchar	191	Peringatan jika melakukan kesalahan ketika

				masuk dalam sistem
7.	Created_at	timestamp	-	Waktu pemasukkan data
8.	Updated_at	timestamp	-	Waktu perubahan data

Tabel 5.10 Struktur Tabel *Migrations*

No.	Nama Fields	Tipe	Panjang	Keterangan
1.	id	integer(PK)	10	Id sebagai <i>primary key</i> dari tabel <i>migrations</i>
2.	Migration	Varchar	191	Data perpindahan
3.	Batch	Int	11	Antrian dari perpindahan

5.4 Perancangan Antarmuka

Perancangna antarmuka merupakan perancangan yang menggambarkan tamplan sistem informasi atau *layout* dri sistem informasi. Antarmuka ini akna memudahkan paada saat pengimplementasian. Bagian ini mengagambarkan tata letak menu, halamman *forms* masukkan, halaman laporrn dan lainnya.

5.4.1 Antarmuka Login

Halaman *login* merupakan halaman pertama yang dilihat oleh pengguna sisstem. Dalam tampilan antarmuka login dibuat sederhana untuk memudahkan pengguna dalam memasukkan data *username* dan *password*. Fungsi untuk login juga ditmpilkan dibawah masukkan untuk memudahkan pengguna.

Login

Username

Password

Remember me

[Forgot you Password?](#)

Gambar 5.10 Antarmuka Login

5.4.2 Antarmuka Halaman Utama

Halaman utama atau *dashboard* menampilkan informasi pengguna dari sistem yang dan ucapan selamat datang pada pengguna. Pada sisi sebelah kiri ditampilkan menu-menu yang disediakan sistem sehingga pengguna dapat melihat jelas semua pilihan menu yang dibutuhkan.

Nama Tabel Menu		Informasi Admin
Admin	Dashboard	
Tampilan Menu		

Gambar 5.11 Antarmuka Halaman Utama

5.4.3 Antarmuka Produk Jual

5.4.3.1 Tambah Data

Halaman tambah data pada antarmuka produk jual merupakan halaman yang menampilkan *form* masukkan data produk jual. Dengan pilihan menu disebelah kiri dan *form* masukkan disebelah kanan. Halaman ini mempunyai fungsi menyimpan data yang telah dimasukkan kedalam sistem.

Nama Tabel Menu		Informasi Admin
Admin	Produk Jual	Tambah
Tampilan Menu	Form inputan	
	Submit	

Gambar 5.12 Antarmuka Tambah Data Produk Jual

5.4.3.2 Rekap Penjualan

Halaman rekap penjualan memiliki fungsi pencarian data penjualan serta menampilkan semua data yang ada pada produk jual. Rekap data tersebut bisa dikelola dengan menggunakan fungsi edit dan hapus. Fungsi untuk menampilkan halaman selanjutnya juga ada dibagian bawah.

Nama Tabel Menu	Informasi Admin
Admin	Produk Jual
Tampilan Menu	<div style="border: 1px solid black; padding: 5px;"> Inputan pencarian </div> <div style="border: 1px solid black; padding: 5px; margin-top: 5px;"> Menampilkan Table Rekap Q Search... </div> <div style="text-align: right; margin-top: 5px;"> Previous Next </div>

Gambar 5.13 Antarmuka Rekap Penjualan

5.4.4 Antarmuka Produk Beli

5.4.4.1 Tambah Data

Halaman tambah data pada antarmuka produk beli merupakan halaman yang menampilkan *form* masukkan data produk beli. Dengan pilihan menu disebelah kiri dan *form* masukkan disebelah kanan. Halaman ini mempunyai fungsi menyimpan data yang telah dimasukkan kedalam sistem.

Nama Tabel Menu	Informasi Admin
Admin	Produk Beli Tambah
Tampilan Menu	<div style="border: 1px solid black; padding: 5px; height: 150px;"> Form inputan </div> <div style="text-align: center; margin-top: 10px;"> Submit </div>

Gambar 5.14 Antarmuka Tambah Data Produk Beli

5.4.4.2 Rekap Pembelian

Halaman rekap pembelian memiliki fungsi pencarian data pembelian serta menampilkan semua data yang ada pada produk beli. Rekap data tersebut bisa dikelola dengan menggunakan fungsi edit dan hapus. Fungsi untuk menampilkan halaman selanjutnya juga ada dibagian bawah.

Nama Tabel Menu	Informasi Admin
Admin	Produk Beli
Tampilan Menu	<div style="border: 1px solid black; padding: 5px;"> Inputan pencarian </div> <div style="border: 1px solid black; padding: 5px;"> Menampilkan Table Rekap🔍 Search... </div> <div style="text-align: right; margin-top: 5px;"> Previous Next </div>

Gambar 5.15 Antarmuka Rekap Pembelian

5.4.5 Antarmuka Maintenance

5.4.5.1 Tambah Data

Halaman tambah data pada antarmuka maintenance merupakan halaman yang menampilkan *form* masukkan data maintenance. Dengan pilihan menu disebelah kiri dan *form* masukkan disebelah kanan. Halaman ini mempunyai fungsi menyimpan data yang telah dimasukkan kedalam sistem.

Nama Tabel Menu	Informasi Admin
Admin	Maintenance
Tampilan Menu	<div style="border: 1px solid black; padding: 5px;"> Form inputan </div> <div style="text-align: center; margin-top: 10px;"> <input type="submit" value="Submit"/> </div>

Gambar 5.16 Antarmuka Tambah Data Maintenance

5.4.5.2 Rekap Maintenance

Halaman rekap maintenance memiliki fungsi pencarian data maintenance serta menampilkan semua data yang ada pada maintenance. Rekap data tersebut bisa dikelola dengan menggunakan fungsi edit dan hapus. Fungsi untuk menampilkan halaman selanjutnya juga ada dibagian bawah.

Nama Tabel Menu	Informasi Admin
Admin	Maintenance
Tampilan Menu	<div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> Inputan pencarian </div> <div style="border: 1px solid black; padding: 5px;"> Menampilkan Table Rekapian Q Search... </div> <div style="text-align: right; margin-top: 5px;"> Previous Next </div>

Gambar 5.17 Antarmuka Rekap Data Maintenance

5.4.6 Antarmuka Pemasukan

5.4.6.1 Tambah Data

Halaman tambah data pada antarmuka pemasukkan merupakan halaman yang menampilkan *form* masukkan data pemasukkan. Dengan pilihan menu disebelah kiri dan *form* masukkan disebelah kanan. Halaman ini mempunyai fungsi menyimpan data yang telah dimasukkan kedalam sistem.

Nama Tabel Menu	Informasi Admin
Admin	Pemasukan
Tampilan Menu	<div style="border: 1px solid black; padding: 5px; margin-bottom: 5px;"> Form inputan </div> <div style="text-align: center; margin-top: 10px;"> Submit </div>

Gambar 5.18 Antarmuka Tambah Data Pemasukkan

5.4.6.2 Rekap Pemasukan

Halaman rekap pemasukkan memiliki fungsi pencarian data pemasukkan serta menampilkan semua data yang ada pada pemasukkan. Rekap data tersebut bisa dikelola dengan menggunakan fungsi edit dan hapus. Fungsi untuk menampilkan halaman selanjutnya juga ada dibagian bawah.

Nama Tabel Menu	Informasi Admin
Admin	Pemasukan
Tampilan Menu	<div style="border: 1px solid black; padding: 5px;"> Inputan pencarian </div> <div style="border: 1px solid black; padding: 5px;"> Menampilkan Table Rekap <input type="text" value="Search..."/> </div> <div style="text-align: right; margin-top: 5px;"> <input type="button" value="Previous"/> <input type="button" value="Next"/> </div>

Gambar 5.19 Antarmuka Rekap Pemasukkan

5.4.7 Antarmuka Pengeluaran

5.4.7.1 Tambah Data

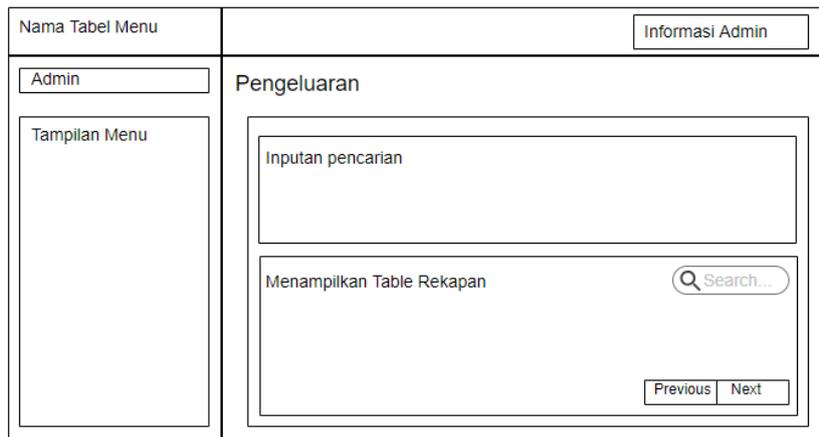
Halaman tambah data pada antarmuka pengeluaran merupakan halaman yang menampilkan *form* masukkan data pengeluaran. Dengan pilihan menu disebelah kiri dan *form* masukkan disebelah kanan. Halaman ini mempunyai fungsi menyimpan data yang telah dimasukkan kedalam sistem.

Nama Tabel Menu	Informasi Admin
Admin	Pengeluaran
Tampilan Menu	<div style="border: 1px solid black; padding: 5px;"> Form inputan </div> <div style="text-align: center; margin-top: 20px;"> <input type="button" value="Submit"/> </div>

Gambar 5.20 Antarmuka Tambah Data Pengeluaran

5.4.7.2 Rekap Data Pengeluaran

Halaman rekap pengeluaran memiliki fungsi pencarian data pengeluaran serta menampilkan semua data yang ada pada pengeluaran. Rekap data tersebut bisa dikelola dengan menggunakan fungsi edit dan hapus. Fungsi untuk menampilkan halaman selanjutnya juga ada dibagian bawah.



Gambar 5.21 Antarmuka Rekap Data Pengeluaran

5.4.8 Antarmuka Laporan

Pada halaman laporan ditampilkan semua data rekap dari produk jual, produk beli, maintenance, pemasukkan dan pengeluaran. Dalam halaman laporan juga ditampilkan saldo keuangan dan keuntungan peternakan setiap bulannya. Pada sisi kanan atas terdapat fungsi cetak laporan yang nantinya akan terhubung dan tersimpan sebagai format pdf.



Gambar 5.22 Antarmuka Laporan

BAB 6 IMPLEMENTASI

Pada bagian ini membahas tentang hasil implementasi yang diperoleh dari penggalan dan analisis kebutuhan serta perancangan. Hasil dari penggalan dan analisis kebutuhan digunakan untuk mengidentifikasi kebutuhan dari sistem agr lebih mudah dalam implementasi sistem informasi yang sudah dijelaskan pada bab 4. Hasil dari perancangan sistem informasi digunakan untuk mendapatkan alur kinerja sistem informasi yang akan dibuat dan sudah dibahas pada bab 5. Berikut hasil dari implementasi sistem informasi manajemen keuangan peternakan kambing.

6.1 Lingkungan Sistem

Pada sub bagian ini menjelaskan tentang lingkungan aplikasi sistem manajemen keuangan peternakan kambing. Hal yang akan dijelaskan meliputi lingkungan perangkat keras dan perangkat lunak yang digunakan oleh pengguna sistem.

6.1.1 Lingkungan Perangkat Keras

Bagian ini menjelaskan tentang perangkat keras yang digunakan pada sistem manajemen keuangan peternakan kambing akan dirinci pada Tabel 6.1.

Tabel 6.1 Lingkungan Perangkat Keras

No.	Nama Perangkat Keras
1.	<i>Processor AMD A4-3305M</i>
2.	<i>RAM 4 GB</i>
3.	<i>HDD 500 GB</i>
4.	<i>Monitor 14"</i>
5.	<i>Display AMD Radeon(TM) HD6480G</i>

6.1.2 Lingkungan Perangkat Lunak

Bagian ini menjelaskan tentang perangkat lunak yang digunakan pada sistem manajemen keuangan peternakan kambing akan dirinci pada Tabel 6.2.

Tabel 6.2 Lingkungan Perangkat Lunak

No.	Nama Perangkat Lunak	Deskripsi
1.	Sistem operasi <i>Windows 10 Pro 64 bit</i>	Sistem operasi yang dikembangkan oleh perusahaan <i>windows</i>
2.	XAMPP v3.2.2	Perangkat lunak untuk membantu pengembangan <i>website</i> dengan menggunakan bahasa pemrograman PHP dan juga membantu pembuatan <i>database</i> dengan MySQL yang berguna sebagai <i>server local</i> .

3.	Sublime version 3.1.1	Perangkat lunak untuk membantu penulisan kode program dalam pembuatan sistem berbasis <i>website</i> .
4.	Google chrome	Perangkat lunak yang dikembangkan oleh Google yang digunakan untuk implementasi sistem informasi berbasis <i>website</i> .

6.2 Implementasi Program

Pada sub bagian ini disertakan kode program yang sudah diimplementasikan pada sistem manajemen keuangan peternakan kambing yang menggunakan *framework Laravel*. Berikut disertakan kode program simpan produk beli dari *controlller* ProdukBeli dengan fungsi *store()*, dapat dilihat pada Tabel 6.3.

Tabel 6.3 Kode Program Simpan Produk Beli

1	<code>public function store(Request \$request)</code>
2	<code>{</code>
3	<code> \$produk beli = new Produk beli();</code>
4	<code> \$produk beli->jenis = \$request->jenis;</code>
5	<code> \$produk beli->produk id = \$request->nama;</code>
6	<code> \$produk beli->jumlah = \$request->jumlah;</code>
7	<code> \$produk beli->tanggal beli = date('Y-m-d',</code> <code> strtotime(\$request->tanggal));</code>
8	<code> \$produk beli->harga = \$request->harga;</code>
9	<code> \$produk beli->save();</code>
10	<code> \$request->session()->flash('status', '<div class="box box-</code> <code>success box-solid"><div class="box-header"><h3 class="box-</code> <code>title">Data Berhasil Disimpan</h3><div class="box-tools pull-</code> <code>right"><button type="button" class="btn btn-box-tool" data-</code> <code>widget="remove"><i class="fa fa-times"></i></button></div></div></div>');</code>
11	<code> return redirect()->route('produkbeli.create');</code>
12	<code>}</code>

Berikut disertakan kode program lihat laporan beli dari *controller* HomeController dengan fungsi *laporan()*, dapat dilihat pada Tabel 6.4.

Tabel 6.4 Kode Program Lihat Laporan

1	<code>public function laporan()</code>
2	<code>{</code>
3	<code> \$jual = Produk_jual::selectRaw('harga * jumlah as total')-></code> <code>get()->sum('total');</code>
4	<code> \$beli = Produk_beli::selectRaw('harga * jumlah as total')-></code> <code>get()->sum('total');</code>
5	<code> \$main = Maintenance::selectRaw('harga * jumlah as total')-></code> <code>get()->sum('total');</code>
6	<code> \$masuk = Pemasukan::get(['harga'])->sum('harga');</code>
7	<code> \$keluar = Pengeluaran::get(['harga'])->sum('harga');</code>
8	
9	<code> \$saldo = number_format(((\$jual+\$masuk) -</code> <code>(\$beli+\$main+\$keluar));</code>
10	<code> \$totJual = Produk_jual::selectRaw('harga * jumlah as</code> <code>total')</code>
11	<code> ->whereMonth('tanggal jual', '=', date('m'))</code>
12	<code> ->whereYear('tanggal jual', '=', date('Y'))</code>
13	<code> ->get()->sum('total');</code>



14	<code>\$totBeli = Produk_beli::selectRaw('harga * jumlah as total')</code>
15	<code>->whereMonth('tanggal_beli', '=', date('m'))</code>
16	<code>->whereYear('tanggal_beli', '=', date('Y'))</code>
17	<code>->get()->sum('total');</code>
18	<code>\$totMain = Maintenance::selectRaw('harga * jumlah as total')</code>
19	<code>->whereMonth('tanggal', '=', date('m'))</code>
20	<code>->whereYear('tanggal', '=', date('Y'))</code>
21	<code>->get()->sum('total');</code>
22	<code>\$totKel = Pengeluaran::whereMonth('tanggal', '=', date('m'))</code>
23	<code>->whereYear('tanggal', '=', date('Y'))</code>
24	<code>->get()->sum('harga');</code>
25	<code>\$totMas = Pemasukan::whereMonth('tanggal', '=', date('m'))</code>
26	<code>->whereYear('tanggal', '=', date('Y'))</code>
27	<code>->get()->sum('harga');</code>
28	<code>return view('home.laporan')->with(['saldo' => \$saldo, 'totJual' => number_format(\$totJual),</code>
29	<code>'totMas' => number_format(\$totMas), 'totMain' => number_format(\$totMain), 'totBeli' => number_format(\$totBeli),</code>
30	<code>'totKel' => number_format(\$totKel)</code>
31	<code>]);</code>
32	<code>}</code>

Berikut disertakan kode program hapus data pemasukkan dari *controlller* PemasukkanController dengan fungsi *destroy()*, dapat dilihat pada Tabel 6.5.

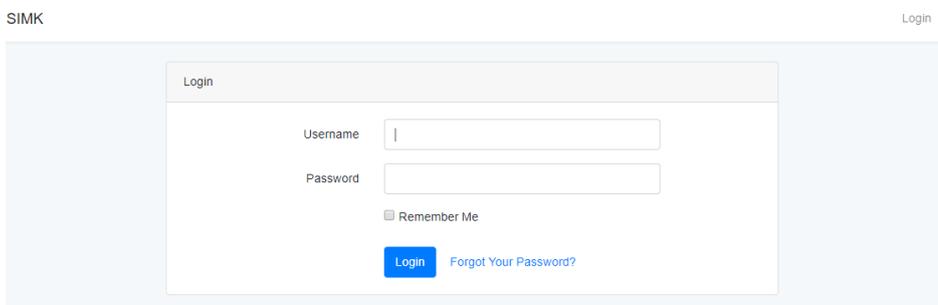
Tabel 6.5 Kode Program Hapus Data Pemasukkan

1	<code>public function destroy(\$id)</code>
2	<code>{</code>
3	<code> Pemasukan::destroy(\$id);</code>
4	<code> return redirect()->route('pemasukanlain.index')->with('status', '<div class="box box-danger box-solid"><div class="box-header"><h3 clas="box-title">Data dihapus.</h3><div class="box-tools pull-right"><butttion type="button" class="btn btn-box-tool" data-widget="remove"><i class="fa fa-times"></i></butttion></div></div></div>');</code>
5	<code>}</code>

6.3 Implementasi Antarmuka

6.3.1 Halaman Login

Pada Gambar 6.1 implemmentasi antarmukaa pada halaman login terdapat *form* untuk memasukkan *username* dan *password* serta fungsi login.



Gambar 6.1 Halaman Login



6.3.2 Halaman *Dashboard*

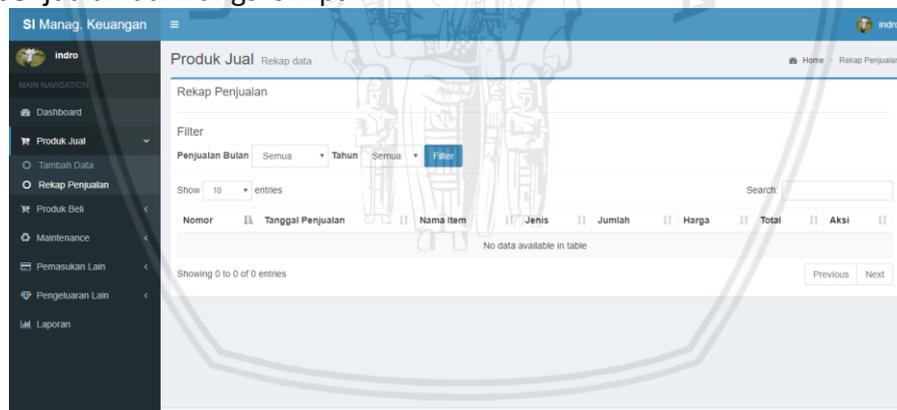
Pada Gambar 6.2 implementasi antarmuka pada halaman *dashboard* menampilkan ucapan selamat datang kepada pengguna dan semua menu yang terdapat pada sistem.



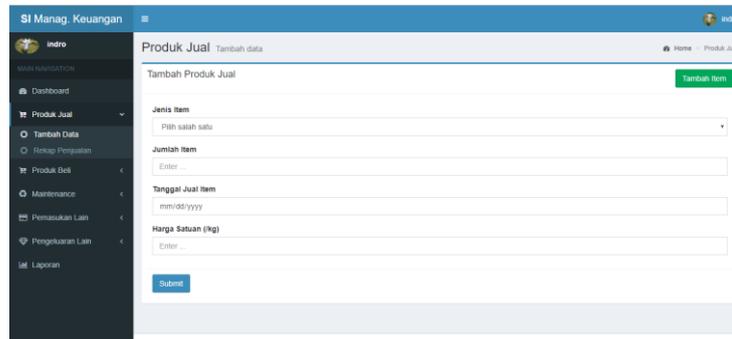
Gambar 6.2 Halaman Dasboard

6.3.3 Halaman Kelola Produk Jual

Pada Gambar 6.3 dan Gambar 6.4 implementasi antarmuka pada halaman kelola produk jual mempunyai 2 tampilan yaitu (1) rekap penjualan; (2) tambah data. Pada tampilan (1) menampilkan semua data dari rekap penjualan serta fungsi edit dan hapus. Pada tampilan (2) menampilkan *form* untuk memasukkan data penjualan dan fungsi simpan.



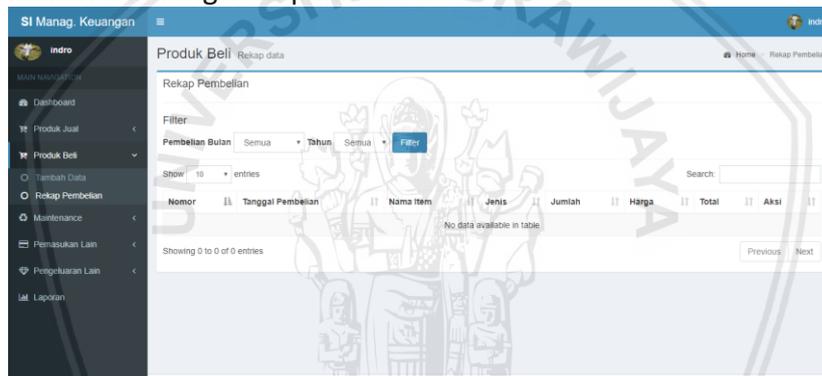
Gambar 6.3 Halaman Kelola Produk Jual (Rekap Penjualan)



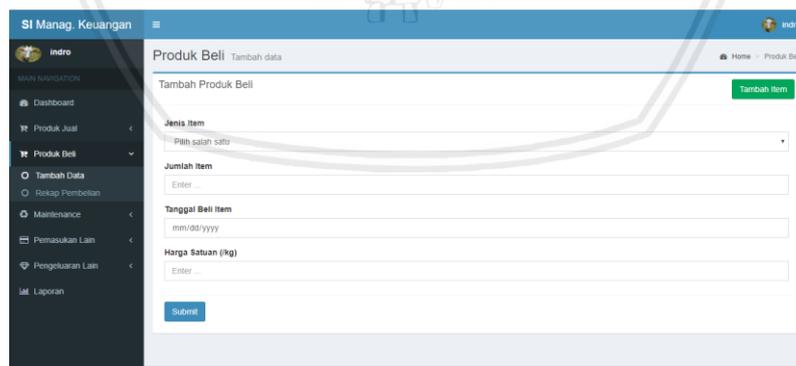
Gambar 6.4 Halaman Kelola Produk Jual (Tambah Data)

6.3.4 Halaman Kelola Produk Beli

Pada Gambar 6.5 dan Gambar 6.6 implementasi antarmuka pada halaman kelola produk beli mempunyai 2 tampilan yaitu (1) rekap pembelian; (2) tambah data. Pada tampilan (1) menampilkan semua data dari rekap pembelian serta fungsi edit dan hapus. Pada tampilan (2) menampilkan *form* untuk memasukkan data pembelian dan fungsi simpan.



Gambar 6.5 Halaman Kelola Produk Beli (Rekap Pembelian)



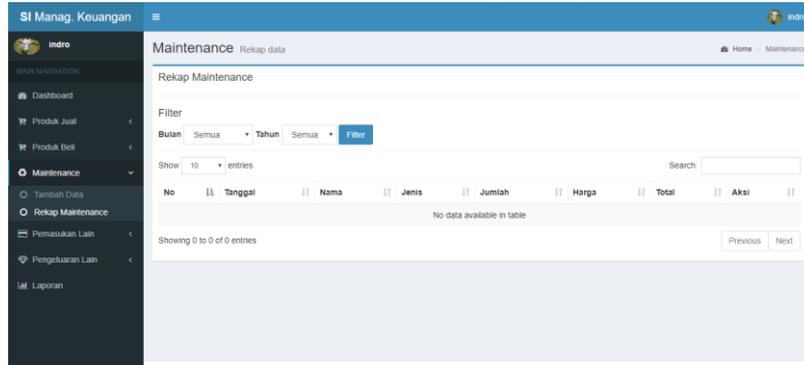
Gambar 6.6 Halaman Kelola Produk Beli (Tambah Data)

6.3.5 Halaman Kelola Data Maintenance

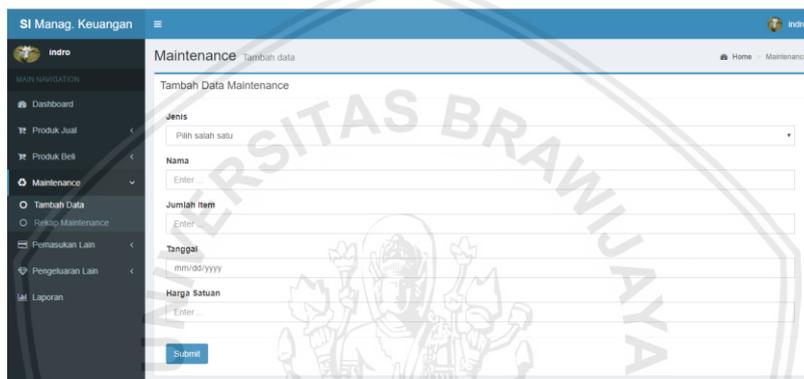
Pada Gambar 6.7 dan Gambar 6.8 implementasi antarmuka pada halaman kelola data maintenance mempunyai 2 tampilan yaitu (1) rekap maintenance; (2) tambah data. Pada tampilan (1) menampilkan semua data dari rekap data



maintenance serta fungsi edit dan hapus. Pada tampilan (2) menampilkan *form* untuk memasukkan data maintenance dan fungsi simpan.



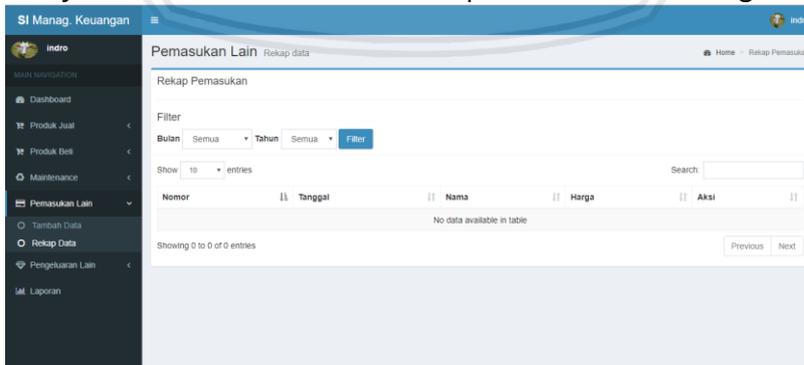
Gambar 6.7 Halaman Kelola Data Maintenance (Rekap Maintenance)



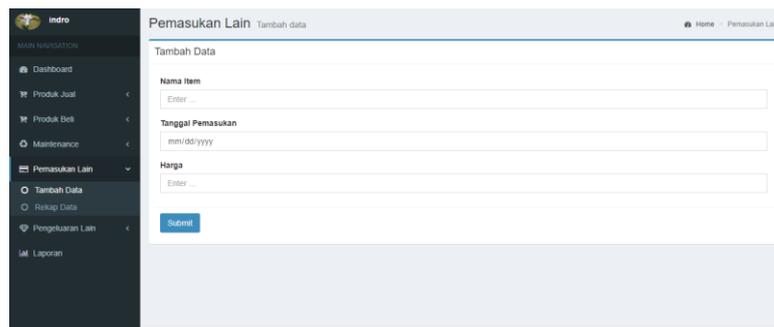
Gambar 6.8 Halaman Kelola Data Maintenance (Tambah Data)

6.3.6 Halaman Kelola Data Pemasukkan

Pada Gambar 6.9 dan Gambar 6.10 implementasi antarmuka pada halaman kelola data pemasukkan mempunyai 2 tampilan yaitu (1) rekap pemasukkan; (2) tambah data. Pada tampilan (1) menampilkan semua data dari rekap data pemasukkan serta fungsi edit dan hapus. Pada tampilan (2) menampilkan *form* untuk memasukkan data pemasukkan dan fungsi simpan.



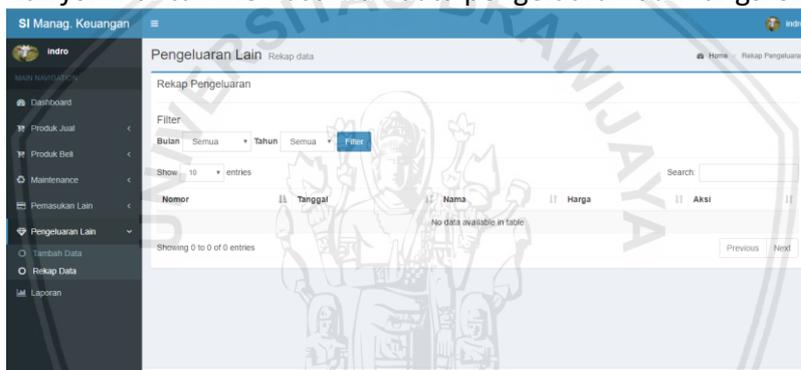
Gambar 6.9 Halaman Kelola Data Pemasukkan (Rekap Pemasukkan)



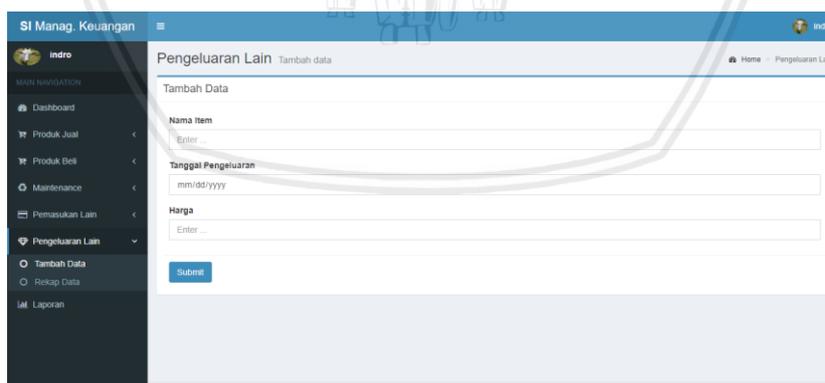
Gambar 6.10 Halaman Kelola Data Pemasukkan (Tambah Data)

6.3.7 Halaman Kelola Data Pengeluaran

Pada Gambar 6.11 dan Gambar 6.12 implementasi antarmuka pada halaman kelola data pengeluaran mempunyai 2 tampilan yaitu (1) rekap pengeluaran; (2) tambah data. Pada tampilan (1) menampilkan semua data dari rekap data pengeluaran serta fungsi edit dan hapus. Pada tampilan (2) menampilkan *form* untuk memasukkan data pengeluaran dan fungsi simpan.



Gambar 6.11 Halaman Kelola Data Pengeluaran (Rekap Pengeluaran)



Gambar 6.12 Halaman Kelola Data Pengeluaran (Tambah Data)

6.3.8 Halaman Laporan

Pada Gambar 6.13 implementasi antarmuka pada halaman laporan menampilkan informasi keuangan data dari produk jual, produk beli, maintenance, pemasukkan, dan pengeluaran. Juga ditampilkan saldo keuangan dan keuntungan, serta terdapat fungsi untuk cetak laporan.



SI Manag. Keuangan

indro

Laporan Bulan Juli

Laporan

Bulan: Januari Tahun: 2018

Sisa saldo = Rp 0

Nomor	Kategori	Total
1	Penjualan Produk Bulan Ini	Rp 0
2	Pembelian Produk Bulan Ini	Rp 0
3	Biaya Maintenance Bulan Ini	Rp 0
4	Pemasukan Lain Bulan Ini	Rp 0
5	Pengeturan Lain Bulan Ini	Rp 0

Gambar 6.13 Halaman Laporan

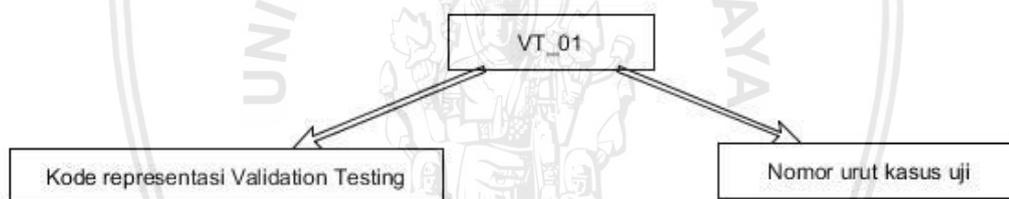


BAB 7 PENGUJIAN

Pada bab ini akan membahas tentang pengujian sistem informasi manajemen keuangan peternakan kambing Yoga's Farm. Pengujian sistem dibagi menjadi 2 bagian pokok. (1) Pengujian Fungsional sistem yang menggunakan *validation testing*; (2) Pengujian nonfungsional sistem yang menggunakan *browser compatibility testing*.

7.1 Pengujian Fungsional

Pada sub bagian ini akan dilaksanakan pengujian dari kebutuhan fungsional sistem informasi manajemen keuangan peternakan kambing yang akan menggunakan *validation testing*. Pengujian ini berfokus pada fungsionalitas sistem, pengujian penting dilakukan karena yang dilihat dari sistem pertama kali adalah fungsionalitas sistem seperti memasukkan pengguna, aksi pengguna, dan lainnya. *Validation testing* dilakukan untuk mengetahui apakah fungsionalitas dari sistem informasi manajemen keuangan peternakan kambing ini sudah sesuai dengan kebutuhan dari peternakan kambing. Oleh karena itu, pada sistem informasi manajemen keuangan peternakan kambing Yoga's Farm membutuhkan pengujian ini.



Gambar 7.1 Kodifikasi *Validation Testing*

Berikut *Validation Testing* akan dijelaskan pada sub-sub bagian di bawah ini:

7.1.1 *Validation Testing* Kelola Produk Jual

Pada Tabel 7.1 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_01 dilakukan untuk memastikan persyaratan dengan kode SKPL_SIMK_F_01 bisa berjalan dengan semestinya (valid).

Tabel 7.1 Rencana dan Kasus Uji *Validation Testing* Kelola Produk Jual

Kode Pengujian	VT_01
Kode Persyaratan	SKPL_SIMK_F_01
Tujuan Pengujian	Pengujian untuk memastikan bahwa sistem dapat mengelola data produk jual
Prosedur Uji	<ol style="list-style-type: none"> 1. Penguji adalah petugas keuangan 2. Penguji memilih menu tambah data 3. Penguji memilih fungsi simpan

	<ol style="list-style-type: none"> 4. Penguji memilih menu rekap data 5. Penguji memilih fungsi edit 6. Penguji memilih fungsi hapus
	<ol style="list-style-type: none"> 1. Penguji memasukkan angka pada form nama 2. Penguji memasukkan huruf pada form tanggal 3. Penguji memasukkan huruf pada form jumlah dan harga
Hasil yang diharapkan	<ol style="list-style-type: none"> 1. Sistem menampilkan <i>form</i> masukkan data 2. Sistem berhasil menyimpan data 3. Sistem menampilkan rekap data produk jual 4. Sistem berhasil mengubah data 5. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Hasil pengujian	<ol style="list-style-type: none"> 1. Sistem menampilkan <i>form</i> masukkan data 2. Sistem berhasil menyimpan data 3. Sistem menampilkan rekap data produk jual 4. Sistem berhasil mengubah data 5. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Status uji	Valid

7.1.2 Validation Testing Kelola Produk Beli

Pada Tabel 7.2 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_02 dilakukan untuk memastikan persyaratan dengan kode SKPL_SIMK_F_02 bisa berjalan dengan semestinya (valid).

Tabel 7.2 Rencana dan Kasus Uji Validation Testing Kelola Produk Beli

Kode Pengujian	VT_02
Kode Persyaratan	SKPL_SIMK_F_02

Tujuan Pengujian	Pengujian untuk memastikan bahwa sistem dapat mengelola data produk beli
Prosedur Uji	<ol style="list-style-type: none"> 1. Penguji adalah petugas keuangan 2. Penguji memilih menu tambah data 3. Penguji memilih fungsi simpan 4. Penguji memilih menu rekap data 5. Penguji memilih fungsi edit 6. Penguji memilih fungsi hapus
	<ol style="list-style-type: none"> 1. Penguji memasukkan angka pada form nama 2. Penguji memasukkan huruf pada form tanggal 3. Penguji memasukkan huruf pada form jumlah dan harga
Hasil yang diharapkan	<ol style="list-style-type: none"> 4. Sistem menampilkan <i>form</i> masukkan data 5. Sistem berhasil menyimpan data 6. Sistem menampilkan rekap data produk beli 7. Sistem berhasil mengubah data 8. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Hasil pengujian	<ol style="list-style-type: none"> 1. Sistem menampilkan <i>form</i> masukkan data 2. Sistem berhasil menyimpan data 3. Sistem menampilkan rekap data produk beli 4. Sistem berhasil mengubah data 5. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Status uji	Valid

7.1.3 Validation Testing Kelola Data Maintenance

Pada Tabel 7.3 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_03 dilakukan untuk memastikan persyaratan dengan kode SKPL_SIMK_F_03 bisa berjalan dengan semestinya (valid).

Tabel 7.3 Rencana dan Kasus Uji Validation Testing Kelola Data Maintenance

Kode Pengujian	VT_03
Kode Persyaratan	SKPL_SIMK_F_03
Tujuan Pengujian	Pengujian untuk memastikan bahwa sistem dapat mengelola data maintenance
Prosedur Uji	<ol style="list-style-type: none"> 1. Penguji adalah petugas keuangan 2. Penguji memilih menu tambah data 3. Penguji memilih fungsi simpan 4. Penguji memilih menu rekap data 5. Penguji memilih fungsi edit 6. Penguji memilih fungsi hapus
	<ol style="list-style-type: none"> 1. Penguji memasukkan angka pada form nama 2. Penguji memasukkan huruf pada form tanggal 3. Penguji memasukkan huruf pada form jumlah dan harga
Hasil yang diharapkan	<ol style="list-style-type: none"> 4. Sistem menampilkan <i>form</i> masukkan data 5. Sistem berhasil menyimpan data 6. Sistem menampilkan rekap data maintenance 7. Sistem berhasil mengubah data 8. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Hasil pengujian	<ol style="list-style-type: none"> 1. Sistem menampilkan <i>form</i> masukkan data 2. Sistem berhasil menyimpan data 3. Sistem menampilkan rekap data maintenance 4. Sistem berhasil mengubah data 5. Sistem berhasil menghapus data

	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Status uji	Valid

7.1.4 Validation Testing Kelola Data Pemasukkan

Pada Tabel 7.4 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_04 dilakukna untuk memastikan persyaratan dengan kode SKPL_SIMK_F_04 bisa berjalan dengan semestinya (valid).

Tabel 7.4 Rencana dan Kasus Uji Validation Testing Data Pemasukkan

Kode Pengujian	VT_04
Kode Persyaratan	SKPL_SIMK_F_04
Tujuan Pengujian	Pengujian untuk memastikan bahwa sisstem dapat mengelola data pemasukkan
Prosedur Uji	<ol style="list-style-type: none"> 1. Penguji adalah petugas keuangan 2. Penguji memilih menu tambah data 3. Penguji memilih fungsi simpan 4. Penguji memilih menu rekap data 5. Penguji memilih fungsi edit 6. Penguji memilih fungsi hapus
	<ol style="list-style-type: none"> 1. Penguji memasukkan angka pada form nama 2. Penguji memasukkan huruf pada form tanggal 3. Penguji memasukkan huruf pada form jumlah dan harga
Hasil yang diharapkan	<ol style="list-style-type: none"> 4. Sistem menampilkan <i>form</i> masukkan data 5. Sistem berhasil menyimpan data 6. Sistem menampilkan rekap data pemasukkan 7. Sistem berhasi mengubah data 8. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Hasil pengujian	1. Sistem menampilkan <i>form</i> masukkan data



	<ol style="list-style-type: none"> 2. Sistem berhasil menyimpan data 3. Sistem menampilkan rekap data pemasukkan 4. Sistem berhasil mengubah data 5. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Status uji	Valid

7.1.5 Validation Testing Kelola Data Pengeluaran

Pada Tabel 7.5 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_05 dilakukan untuk memastikan persyaratan dengan kode SKPL_SIMK_F_05 bisa berjalan dengan semestinya (valid).

Tabel 7.5 Rencana dan Kasus Uji Validation Testing Kelola Data Pengeluaran

Kode Pengujian	VT_05
Kode Persyaratan	SKPL_SIMK_F_05
Tujuan Pengujian	Pengujian untuk memastikan bahwa sistem dapat mengelola data pengeluaran
Prosedur Uji	<ol style="list-style-type: none"> 1. Penguji adalah petugas keuangan 2. Penguji memilih menu tambah data 3. Penguji memilih fungsi simpan 4. Penguji memilih menu rekap data 5. Penguji memilih fungsi edit 6. Penguji memilih fungsi hapus
	<ol style="list-style-type: none"> 1. Penguji memasukkan angka pada form nama 2. Penguji memasukkan huruf pada form tanggal 3. Penguji memasukkan huruf pada form jumlah dan harga
Hasil yang diharapkan	<ol style="list-style-type: none"> 4. Sistem menampilkan <i>form</i> masukkan data 5. Sistem berhasil menyimpan data 6. Sistem menampilkan rekap data pengeluaran 7. Sistem berhasil mengubah data 8. Sistem berhasil menghapus data



	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Hasil pengujian	<ol style="list-style-type: none"> 1. Sistem menampilkan <i>form</i> masukkan data 2. Sistem berhasil menyimpan data 3. Sistem menampilkan rekap data pengeluaran 4. Sistem berhasil mengubah data 5. Sistem berhasil menghapus data
	<ol style="list-style-type: none"> 1. Sistem menolak masukkan angka 2. Sistem menolak masukkan huruf 3. Sistem menolak masukkan huruf
Status uji	Valid

7.1.6 Validation Testing Melihat Laporan

Pada Tabel 7.6 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_06 dilakukan untuk memastikan persyaratan dengan kode SKPL_SIMK_F_06 bisa berjalan dengan semestinya (valid).

Tabel 7.6 Rencana dan Kasus Uji *Validation Testing* Melihat Laporan

Kode Pengujian	VT_06
Kode Persyaratan	SKPL_SIMK_F_06
Tujuan Pengujian	Pengujian untuk memastikan bahwa sistem bisa menampilkan laporan keuangan
Prosedur Uji	<ol style="list-style-type: none"> 4. Penguji adalah petugas keuangan 5. Penguji memilih menu laporan
Hasil yang diharapkan	Sistem menampilkan laporan keuangan
Hasil pengujian	Sistem menampilkan laporan keuangan
Status uji	Valid

7.1.7 Validation Testing Mencetak Laporan

Pada Tabel 7.7 berisi informasi tentang pengujian validasi dengan kode VT_01. Kasus uji VT_07 dilakukan untuk memastikan persyaratan dengan kode SKPL_SIMK_F_07 bisa berjalan dengan semestinya (valid).

Tabel 7.7 *Validation Testing* Mencetak Laporan

Kode Pengujian	VT_07
Kode Persyaratan	SKPL_SIMK_F_07



Tujuan Pengujian	Pengujian untuk memastikan bahwa sistem dapat mencetak laporan keuangan
Prosedur Uji	6. Penguji adalah petugas keuangan 7. Penguji memilih fungsi cetak laporan
Hasil yang diharapkan	Sistem berhasil mencetak laporan keuangan
Hasil pengujian	Sistem berhasil mencetak laporan keuangan
Status uji	Valid

7.2 Pengujian Nonfungsional

Pada sub bagian ini dibahas pengujian non-fungsional untuk mengetahui semua layanan yang ada pada sistem dan kelayakan sistem. Pengujian non-fungsional yang digunakan meliputi *Browser Compatibility Testing*.

7.2.1 Browser Compatibility Testing

Pada sistem informasi berbasis *website*, ada beberapa hal yang mempengaruhi kinerja dari sistem. Hal tersebut adalah perangkat komputer, perangkat *display*, koneksi internet, *browser*, dan sistem operasi. Dalam penggunaan sistem informasi berbasis *website* biasa terjadi beberapa masalah dalam kompatibilitas, meskipun tidak menjadi masalah yang signifikan. Namun di sisi lain juga ditemukan masalah yang serius seperti ukuran atau gaya huruf berubah, tampilan yang menjadi berantakan, dan lain sebagainya. Dalam pengujian kompatibilitas ini dilakukan pemeriksaan masalah-masalah tersebut.

Browser Compability Testing dilakukan pada sistem yang dilihat dari sisi penggunaan pada *web browser*. Pada pengujian ini digunakan sebuah perangkat lunak bantuan yaitu *SortSite* versi 5.29.818.0. Alat ini membantu pemeriksaan kompatibilitas sistem pada beberapa *web browser* yang ada. *Browser* yang diperiksa kompatibilitasnya terhadap sistem dapat dilihat pada Tabel 7.2.

Tabel 7.8 Browser Yang Digunakan Pada *Browser Compatibility Testing*

No	Nama Browser	Versi Browser
1	Internet Explorer	9, 10, dan 11
2	Microsoft Edge	16
3	Firefox	60
4	Safari	<=10 dan 11
5	Opera	51
6	Chrome	66
7	iOS	<=9, 10 dan 11
8	Android	<=3 dan 4
9	BlackBerry	<=7.1 dan 10.0

Pada Gambar 7.1 digambarkan hasil dari pengujian kompatibilitas yang dijelaskan sebagai berikut:

1. **Browser** merupakan nama *browser* yang digunakan dalam pengujian kompatibilitas
2. **Version** merupakan versi dari *browser* yang digunakan dalam pengujian kompatibilitas.
3. **Critical issue** merupakan pengindikasian sebuah masalah yang terjadi pada fitur atau konten yang tidak bisa dijalankan atau tidak didukung pada suatu *browser*.
4. **Major issue** merupakan pengindikasian sebuah masalah utama dalam performa pada suatu *browser*.
5. **Minor issue** merupakan pengindikasian sebuah masalah minor dalam performa pada suatu *browser*.

Browser	IE			Edge	Firefox	Safari		Opera	Chrome	iOS			Android		BlackBerry	
	9	10	11	16	60	≤ 10	11	51	66	≤ 9	10	11	≤ 3	4*	≤ 7.1	10.0
Critical Issues	✓	✓	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓	✓
Major Issues	⚠	⚠	⚠	⚠	✓	⚠	✓	✓	✓	⚠	✓	✓	⚠	⚠	⚠	⚠
Minor Issues	⚠	⚠	⚠	⚠	⚠	✓	✓	✓	✓	✓	✓	✓	✓	✓	⚠	✓

Gambar 7.2 Browser Compatibility Testing dengan SortSite

Hasil dari pengujian kompatibilitas pada Gambar 7.1 dapat membantu pemeriksaan masalah yang ada pada suatu *browser*. Hasil pengujian menunjukkan sistem dapat dijalankan pada 9 jenis *browser*. Berdasarkan pengujian kompatibilitas *browser* didapatkan hasil menunjukkan sistem dapat dijalankan pada 9 jenis *browser*. Namun, pada IE (*Internet Explorer*) versi 11 terdapat *critical issue* yang menandakan fitur atau konten tidak bisa dijalankan pada peramban tersebut. Terdapat 9 *major issue* dari 16 versi peramban yang menunjukkan bahwa ada masalah performa utama pada 9 versi peramban tersebut. Dan ada 5 *minor issue* dari 10 versi peramban yang menunjukkan bahwa ada masalah kecil yang tidak berpengaruh pada sistem. Jadi, secara keseluruhan sistem manajemen keuangan ini masih kompatibel dengan 9 macam *browser* dengan berbagai versi.

BAB 8 PENUTUP

8.1 Kesimpulan

Berdasarkan hasil dari penelitian yang sudah dilakukan, dapat diambil kesimpulan dari hasil pengembangan sistem informasi manajemen keuangan peternakan kambing, yaitu sebagai berikut:

1. Analisis dari kebutuhan sistem mendapatkan hasil identifikasi proses bisnis AS-IS dan proses bisnis TO-BE pada pembelian dan penjualan. Analisis persyaratan sistem menganalisis masalah yang ada pada peternakan, hasil identifikasi pengguna terdapat 1 pengguna yaitu petugas keuangan, serta terdapat 4 fitur yang telah teridentifikasi dari kebutuhan pengguna. Dalam persyaratan sistem dilakukan analisis persyaratan fungsional dan nonfungsional, ditemukan 12 fungsional dan 1 nonfungsional. Persyaratan fungsionalnya adalah kelola produk jual, melihat data produk jual, kelola produk beli, melihat data produk beli, kelola data maintenance, melihat data maintenance, kelola data pemasukkan, melihat data pemasukkan, kelola data pengeluaran, melihat data pengeluaran, melihat rekap transaksi dan keuntungan, dan mencetak laporan. Kemudian dimodelkan pada *use case diagram* dan dijelaskan dengan *use case specification*.
2. Perancangan sistem informasi dilakukan untuk memodelkan sistem ke dalam notasi-notasi pada *Unified Modeling Language* (UML) berdasarkan hasil tahap analisis persyaratan. Perancangan sistem berisi hasil pemodelan interaksi antar objek ke dalam *sequence diagram*, pemodelan objek ke dalam *class diagram*, perancangan *database*, perancangan algoritma ke dalam *pseudocode*, dan perancangan antarmuka pengguna sistem. Perancangan pada penelitian ini menghasilkan 6 *sequence diagram*, 7 *class model*, 6 *class controller*, dan 1 *class authentication* pada *class diagram*, 10 tabel pada rancangan *database*, dan 12 rancangan antarmuka. Perancangan dibuat sesuai dengan analisis kebutuhan yang telah didefinisikan pada bab penggalan data dan analisis kebutuhan sistem.
3. Implementasi pada penelitian ini menggunakan perangkat lunak pendukung XAMPP yang berguna untuk membantu pengembangan *website* dengan bahasa pemrograman PHP dan pembuatan *database*, serta untuk penulisan program menggunakan *Sublime*. Penelitian ini juga menghasilkan kode program yang menggunakan kerangka kerja *laravel*. Implementasi ini juga menghasilkan 12 tampilan antarmuka yang sesuai dengan rancangan pada bab perancangan. Perangkat lunak pembantu dalam tes implementasi menggunakan Google Chrome.
4. Dari hasil *validation testing* yang telah dilakukan oleh peneliti, hasil yang didapatkan adalah 100% valid dari 12 fungsional. Hal tersebut menandakan bahwa sistem telah berjalan dengan semestinya dan sesuai dengan ekspektasi pengguna. Pengujian validasi juga memasukkan kasus uji kesalahan yang mungkin dilakukan oleh pengguna. Berdasarkan pengujian kompatibilitas

browser didapatkan hasil menunjukkan sistem dapat dijalankan pada 9 jenis *browser*. Namun, pada IE (*Internet Explorer*) versi 11 terdapat *critical issue* yang menandakan fitur atau konten tidak bisa dijalankan pada peramban tersebut. Terdapat 9 *major issue* dari 16 versi peramban yang menunjukkan bahwa ada masalah performa utama pada 9 versi peramban tersebut. Dan ada 5 *minor issue* dari 10 versi peramban yang menunjukkan bahwa ada masalah kecil yang tidak berpengaruh pada sistem. Jadi, secara keseluruhan sistem manajemen keuangan ini masih kompatibel dengan 9 macam *browser* dengan berbagai versi.

8.2 Saran

Saran yang dapat peneliti berikan untuk bahan pertimbangan perbaikan untuk pengembangan selanjutnya pada sistem infomasi manajemen keuangan peternakan kambing ini adalah:

1. Diperlukan adanya evaluasi dari segi antarmuka pada *website* untuk mengetahui perlu dilakukan perbaikan atau pengembangan antarmuka lebih lanjut.
2. Diperlukan analisis persyaratan pengguna yang berkala untuk mengetahui kebutuhan pengguna terhadap sistem yang suatu saat mungkin bisa berubah.
3. Pengembangan berikutnya, dapat menambahkan fitur untuk kelola data produk dalam gudang untuk mengelola data produk yang ada dalam gudang.

DAFTAR PUSTAKA

- Alshamrani, A. & Bahattab, A., 2015. *A Comparison Between Three SDLC Models Waterfall Model, Spiral Model, and Incremental/Iterative Model*. Jeddah: *IJCSI International Journal of Computer Science*, XII(1), pp. 106-111.
- Arif, Utama, B., 2015. *Pemanfaatan Teknologi Informasi di Kalangan Mahasiswa Universitas Negeri Surabaya*. Surabaya: Universitas Airlangga.
- Darsono, 2011. *MANAJEMEN KEUANGAN: Pengertian, Fungsi, Tujuan Manajememen Keuangan*. [Online] <<http://www.contohsurat.co.id/2017/02/manajemen-keuangan.html>> [Diakses 17 Februari 2018]
- Group Object Management, 2011. *About The Business Process Model And Notation Specification Version 2.0*. [Online] <<https://www.omg.org/spec/BPMN/2.0/About-BPMN/>> [Diakses 20 Februari 2018]
- Kertahadi, 2007. 15 Pengertian Sistem Informasi Menurut Ahli Terlengkap dan Terbaru 2017. [Online] Tersedia di <<http://www.pelajaran.co.id/2017/29/pengertian-sistem-informasi-menurut-ahli.html>> [Diakses 17 Februari 2018]
- Laravel., 2018. *Documentation*. [Online] <<https://laravel.com/docs/5.7>> [Diakses 9 November 2018]
- McLeod, Raymond. 2001. *Sistem Informasi Manajemen*. Jilid 1. Prehallindo: Jakarta.
- Pamungkas, A. F., 2017. *Pengembangan Sistem Infromasu Persediaan Barang, Harga Pokok Produksi, dan Transaksi Penjualan Berbasis Web Pada Son Screen Printing Sidoarjo*. Malang: Universitas Brawijaya.
- Parkar, Vishal, V., 2016. *Utilization of Laravel Framework for Development of Web Based Recruitment Tool*. India: *Department of Computer Engineering*.
- Pressman, R. S., 2010. *Software Engineering A Practitioner's Approach 7th Ed - Roger S. Pressman*. 7th penyunt. New York: McGraw-Hill.
- Qisthi, F. I., 2018. *Pengembangan Sistem Informasi Kesehatan Promer Untuk Mendukung Penerapan Model Organisasi dan Manajemen Kesehatan Primer ANDAL Pada Poliklinik Pabrik Gula Kebon Agung*. Malang: Universitas Brawijaya.
- Ratnawati, S., 2014. *Pengembangan Sistem Informasi Akuntansi Syariah (Studi Kasus: BMT Bintaro)*. *Jurnal Sistem Informasi*.
- Republik Indonesia. 2009. *Keputusan Dewan Perwakilan Rakyat dan Presiden 18 Tahun 2009 tentang Peternakan dan Kesehatan Hewan*. Sekretariat Negara. Jakarta.
- Rohman, A., 2014. *Mengenal Framework "Laravel" (Best PHP Framework 2014)*. Indonesia: Ilmu Teknologi Informasi.
- Romney, 1997. 15 Pengertian Sistem Informasi Menurut Ahli Terlengkap dan Terbaru 2017. [Online] Tersedia di <<http://www.pelajaran.co.id/2017/29/pengertian-sistem-informasi-menurut-ahli.html>> [Diakses 17 Februari 2018]

Sommerville, I., 2011. *Software Engineering 9th ed.* Ninth Edition. United States of Amerika.

Terry, George, R., 2011. *Dasar-Dasar Manajemen.* Jakarta: Bumi Aksara.



LAMPIRAN A HASIL WAWANCARA

A.1 Hasil Wawancara dengan Pemilik Peternakan Kambing

Berikut hasil wawancara tidak terstruktur yang dilakukan peneliti untuk mengetahui kebutuhan dari sistem manajemen keuangan peternakan kambing yang dilaksanakan pada:

Tanggal	: 3 Januari 2018
Waktu	: 10.00 – 11.00 WIB
Tempat	: Yoga's Farm Tulungagung
Narasumber	: Yoga Wahyu Pratama, S.Pd
Jabatan	: Pemilik Peternakan

Pertanyaan: Sejak kapan Bapak memulai usaha peternakan kambing ini?

Jawaban: Dulu usaha ini dimulai waktu saya kuliah, sekitar tahun 2008. Tapi masih belum bentuk usaha peternakan seperti sekarang. Nah, kalo peternakan seperti sekarang ini mulai dirintis tahun 2012.

Pertanyaan: Berapa karyawan yang bekerja disini dan apa saja tugasnya?

Jawaban: Karyawan disini ada 3 orang, satu orang di bagian pemberian pakan dan kesehatan ternak, satu orang di bagian pencampuran pakan dan kesehatan kandang, dan satu orang dibagian keuangan.

Pertanyaan: Apa saja kegiatan yang dikerjakan di peternakan?

Jawaban: Di peternakan ada beberapa kegiatan yang pasti. Pertama penjualan kambing dan domba yang sudah gemuk, kedua penjualan pakan mentah atau sudah jadi, ketiga penjualan obat ternak, keempat penjualan alat peternakan, dan kelima penjualan bibit ternak.

Pertanyaan: Bagaimana cara penjualan kambing di peternakan?

Jawaban: Kambing hidup ditimbang, nanti harganya sesuai dengan hasil timbangan, jadi dijualnya per kilo bukan perekor.

Pertanyaan: Bagaimana peternakan bisa memperoleh semua kebutuhan ternak, pakan, dan lainnya?

Jawaban: Ya tinggal pesan pada suplier.

Pertanyaan: Bagaimana cara pelaporan keuangan di peternakan?

Jawaban: Selama ini pelaporan ya direkap bagian keuangan setiap bulan sekali. Nota pembelian maupun penjualan dan semua kegiatan yang membutuhkan dana dikumpulkan setiap bulan untuk direkap. Kalo tidak setiap bulan notanya banyak yang hilang. Itupun kadang keuntungannya masih berantakan, gak tau itu uang kemana larinya.

Pertanyaan: Apakah pembuatan laporan membutuhkan waktu lama?

Jawaban: Ya kalo pas rekap itu ya lumayan lama, soalnya data nota perbulannya itu juga banyak jadi ya kalo nulis satu persatu pasti lama.

Pertanyaan: Jika saya menawarkan sebuah sistem untuk pembuatan laporan apakah bapak bersedia?

Jawaban: Ya jika itu bisa mempermudah laporan keuangan setiap bulannya malah bagus itu mas.

Pertanyaan: Apa saja yang peternakan butuhkan dalam pengolahan data keuangan?

Jawaban: Yang jelas semua data keuangan keluar masuk itu. Seperti yang sudah saya jelaskan tadi, semua penjualan dan pembelian produk dan kegiatan yang ada di peternakan. Termasuk juga gaji karyawan, kebutuhan listrik, kebutuhan pangan karyawan. Dan jika bisa sekalian data itu sudah dirapikan sistem, jadi nanti petugas keuangan tinggal cetak, tidak perlu edit lagi.

