

**PEMBANGUNAN SISTEM *BOOKING ONLINE* DALAM PENYEWAAN
JASA *EVENT ORGANIZER* BERBASIS *WEB* PADA**

CV. V-PRO KOTA PADANG

TUGAS AKHIR

Diajukan sebagai Salah Satu Syarat untuk Menyelesaikan Program Strata-1 pada
Program Studi Sistem Informasi Fakultas Teknologi Informasi



**PROGRAM STUDI SISTEM INFORMASI
FAKULTAS TEKNOLOGI INFORMASI
UNIVERSITAS ANDALAS**

2019

HALAMAN PENGESAHAN

PEMBANGUNAN SISTEM *BOOKING* ONLINE DALAM PENYEWAAAN
JASA EVENT ORGANIZER BERBASIS WEB PADA CV. V-PRO KOTA
PADANG

Oleh

Maghfirah Maulani

1411521003



LULUS SIDANG TUGAS AKHIR
23 Juli 2019

Padang, 25 Juli 2019
Telah diperiksa dan disetujui oleh,
Pembimbing Tugas Akhir

Ricky Akbar, M.Kom
NIP. 198410062012121001

Mengetahui,

Ketua Jurusan Sistem Informasi



Hasdi Putra, M.T
NIP. 198307272008121003

KATA PENGANTAR

Puji dan syukur penulis ucapkan kehadirat Allah SWT, atas segala rahmat dan karunia-Nya sehingga penulis dapat menyelesaikan proposal tugas akhir ini. Proposal tugas akhir ini disusun berdasarkan penelitian yang penulis lakukan pada CV. V-Pro Kota Padang yang berjudul “Pembangunan Sistem *Booking Online* Dalam Penyewaan Jasa *Event Organizer* Berbasis *Web* Pada CV. V-Pro Kota Padang”.

Dalam penyusunan proposal tugas akhir ini penulis banyak mendapatkan bantuan dari berbagai pihak. Oleh sebab itu, penulis mengucapkan terima kasih atas bimbingan dan bantuan kepada:

1. Bapak Ricky Akbar, M.Kom sebagai dosen pembimbing dalam tugas akhir ini,
2. Bapak Agung Wijaya, selaku pemilik penyewa jasa *event organizer* pada CV. V-Pro Kota Padang,
3. Seluruh pihak yang terlibat dan yang telah membantu dan memberikan dukungan kepada penulis baik secara moril maupun materil dalam pembuatan proposal tugas akhir ini.

Penulis menyadari bahwa penulisan laporan ini jauh dari kata sempurna. Kritik dan saran dari pembaca melalui *email* maghfirahmaulani123@gmail.com yang bersifat membangun sangatlah diharapkan. Semoga dengan pembuatan laporan ini dapat memberikan ilmu dan manfaat bagi pembaca dan terutama bagi penulis. Semoga Allah memberikan rahmat dan berkahnya kepada kita semua, Amin.

Padang, 07 Mei 2019

Penulis,

Maghfirah Maulani

ABSTRAK

Event Organizer merupakan pengelola suatu kegiatan atau pengorganisir acara berupa jasa, untuk membantu pelanggan agar dapat menyelenggarakan acara yang diinginkan dan dilakukan dengan profesional sehingga hasilnya lebih. Saat ini banyak perusahaan yang bergerak dibidang penyediaan jasa *event organizer* yang menyediakan berbagai macam pilihan item yang akan disewakan salah satunya adalah CV. V-Pro. Proses *booking* dalam penyewaan jasa *event organizer* CV. V-Pro masih menggunakan cara manual yaitu *customer* datang langsung ke perusahaan untuk penyewaan serta memilih item yang diinginkan dan pengelolaan data *customer* pada CV. V-Pro masih menggunakan kertas. Hal ini menjadikan kurang efisiennya proses *booking* dalam penyewaan pada *event organizer* ini. Maka dibuatlah sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web*, sehingga memudahkan *customer* dalam melakukan *booking* maupun penyewaan jasa yang dibutuhkan dan juga memudahkan penyedia jasa dalam mengelola data *customer*. Tujuan dari penelitian ini yaitu untuk menganalisis, merancang, membangun, serta menguji dengan memanfaatkan teknologi informasi berbasis *web*. Metodologi penelitian yang digunakan adalah studi literatur sebagai landasan teori dari data penelitian sebelumnya, pengumpulan data berupa observasi, wawancara dan pengumpulan dokumen, serta menggunakan metode pengembangan sistem waterfall. Tahapan metode waterfall meliputi analisis, design, coding, dan testing terhadap aplikasi yang dibuat. Dari penelitian ini dihasilkan sebuah aplikasi berbasis *web* untuk sistem penyewaan jasa *event organizer*, dan dari hasil pengujian sistem tersebut diperoleh bahwa perancangan dan pembangunan sistem telah sesuai dengan kebutuhan.

Kata Kunci: Sistem *booking online* dalam penyewaan jasa *event organizer*, aplikasi, web.

DAFTAR ISI

HALAMAN PENGESAHAN	ii
KATA PENGANTAR	iii
ABSTRAK	iv
DAFTAR ISI	v
DAFTAR GAMBAR	ix
DAFTAR TABEL	xi
BAB I PENDAHULUAN	1
1.1 Latar Belakang	1
1.2 Rumusan Masalah	3
1.3 Batasan Masalah.....	3
1.4 Tujuan Penelitian.....	3
1.5 Manfaat Penelitian.....	3
1.6 Sistematika Penulisan.....	4
1.7 Analisis Aplikasi Sejenis.....	4
BAB II TINJAUAN PUSTAKA	6
2.1 Penyewaan Jasa	6
2.2 <i>Event Organizer</i> (EO).....	6
2.3 <i>Booking Online</i>	7
2.4 Perangkat Lunak Pendukung.....	8
2.4.1 <i>Website</i>	8
2.4.2 <i>PHP (PHP Hypertext Preprocessor)</i>	8
2.4.3 <i>Framework</i>	9
2.4.4 <i>Laravel</i>	9
2.4.4.1 <i>Kelebihan Laravel</i>	10
2.4.4.2 <i>Fitur Laravel</i>	11

2.4.5	MySQL.....	12
2.5	<i>Tools</i> Pemodelan Sistem	13
2.5.1	<i>Business Process Modelling Notation</i> (BPMN).....	13
2.5.2	UML (<i>Unified Modeling Language</i>)	16
2.5.3	<i>Entity Relationship Diagram</i>	18
BAB III METODE PENELITIAN		19
3.1	Objek Kajian.....	19
3.2	Metode Pengumpulan Data	19
3.3	Metode Pengembangan Sistem	20
3.4	Flowchart Penelitian.....	21
BAB IV ANALISIS DAN PERANCANGAN SISTEM.....		25
4.1.	Analisis Sistem.....	25
4.1.1.	Kedudukan Sistem.....	25
4.1.2.	Sistem Berjalan	25
4.1.3.	Sistem Diusulkan.....	26
4.1.4.	Kebutuhan Fungsional Aplikasi.....	28
4.1.5.	Use Case Diagram Aplikasi	28
4.1.6.	Deskripsi Tugas Aktor	30
4.1.7.	<i>Use Case Scenario</i> Aplikasi.....	31
4.1.7.1.	<i>Use Case Scenario</i> Registrasi	31
4.1.7.2.	<i>Use Case Scenario</i> Tambah Transaksi.....	32
4.1.7.3.	<i>Use Case Scenario</i> Cetak Transaksi.....	33
4.1.8.	<i>Sequence Diagram</i>	33
4.1.8.1.	<i>Sequence Diagram</i> Registrasi	33
4.1.8.2.	<i>Sequence Diagram</i> Tambah Transaksi.....	34
4.1.8.3.	<i>Sequence Diagram</i> Cetak Transaksi	35
4.1.9.	<i>Class Analysis</i>	36
4.1.9.1.	<i>Class Analysis</i> Registrasi	36
4.1.9.2.	<i>Class Analysis</i> Tambah Transaksi.....	37
4.1.9.3.	<i>Class Analysis</i> Cetak Transaksi.....	37

4.2.	Perancangan Sistem.....	38
4.2.1.	Perancangan Basis Data	38
4.2.2.	Struktur Tabel dan Basis Data.....	39
4.2.3.	Arsitektur Aplikasi	40
4.2.4.	<i>Class Diagram</i>	41
4.2.5.	Perancangan Antarmuka	43
4.2.5.1.	Perancangan Antarmuka Registrasi.....	43
4.2.5.2.	Perancangan Antarmuka Tambah Transaksi.....	44
4.2.5.3.	Perancangan Antarmuka Cetak Transaksi.....	44
BAB V IMPLEMENTASI DAN PENGUJIAN SISTEM.....		46
5.1.	Implementasi Sistem	46
5.1.1.	Pengkodean Program.....	47
5.1.1.1.	Kode Program <i>Routing</i> Aplikasi	47
5.1.1.2.	Kode Program <i>Controller</i>	48
5.1.1.3.	Kode Program <i>Model</i>	51
5.1.1.4.	Kode Program <i>View</i>	52
5.1.2.	Implementasi Antar Muka Aplikasi	55
5.1.2.1.	Halaman Login.....	55
5.1.2.2.	Halaman Admin	56
5.1.2.3.	Halaman <i>Customer</i>	57
5.2.	Pengujian Sistem	59
5.2.1.	Fokus Pengujian	59
5.2.2.	Kasus Hasil Pengujian.....	60
5.2.2.1.	Pengujian Registrasi Melalui Aplikasi <i>Web</i>	60
5.2.2.2.	Pengujian Tambah Transaksi	63
5.2.2.3.	Pengujian Cetak Transaksi	65
5.3.	Hasil Pengujian	67
BAB VI PENUTUP.....		70
6.1	Kesimpulan.....	70
6.2	Saran.....	71
DAFTAR PUSTAKA		72
LAMPIRAN A.....		75

LAMPIRAN B	77
LAMPIRAN C	85
LAMPIRAN D	95
LAMPIRAN E	101
LAMPIRAN F	103
LAMPIRAN G	111
LAMPIRAN H	124
LAMPIRAN I	131



DAFTAR GAMBAR

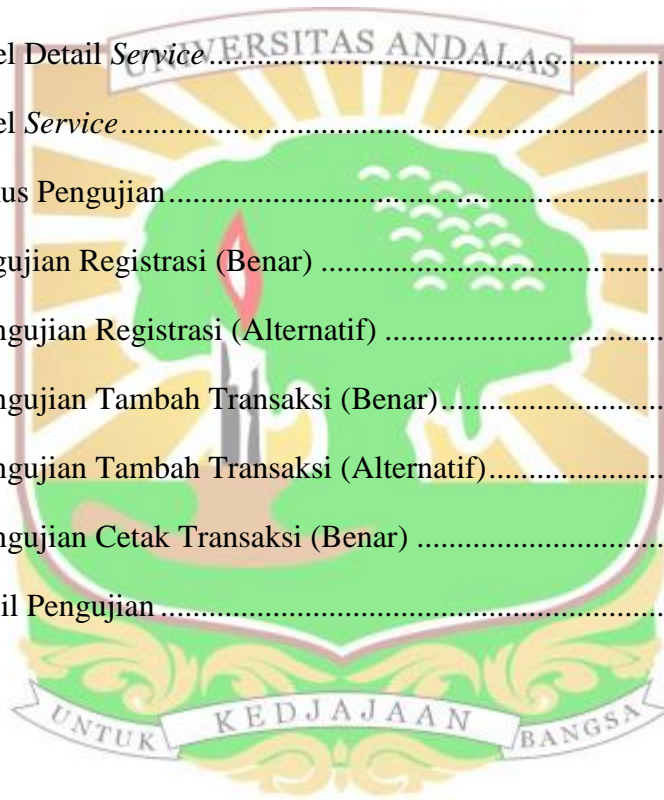
Gambar 2.1 Simbol <i>Event</i>	14
Gambar 2.2 Simbol <i>Activity</i> dan <i>gateway</i>	14
Gambar 2.3 Tipe-tipe <i>Connecting Object</i>	15
Gambar 2.4 Simbol <i>Pool</i> dan <i>Lane</i> dalam <i>Swimlane</i>	15
Gambar 3.1 Metode <i>Waterfall</i>	20
Gambar 3.2 <i>Flow chart</i> <i>Penelitian</i>	22
Gambar 4.1 BPMN Penyewaan Jasa <i>EO</i> yang Sedang Berjalan	26
Gambar 4.2 BPMN Register yang di Usulkan	27
Gambar 4.3 BPMN Penyewaan Jasa <i>EO</i> yang di Usulkan	28
Gambar 4.4 <i>Use Case Diagram</i> Perancangan Aplikasi	29
Gambar 4.5 <i>Sequence Diagram</i> Registrasi	34
Gambar 4.6 <i>Sequence Diagram</i> Tambah Transaksi	35
Gambar 4.7 <i>Sequence Diagram</i> Cetak Transaksi	36
Gambar 4.8 <i>Class Analysis</i> Registrasi	36
Gambar 4.9 <i>Class Analysis</i> Tambah Transaksi	37
Gambar 4.10 <i>Class Analysis</i> Cetak Transaksi	38
Gambar 4.11 Rancangan ERD Sistem	39
Gambar 4.12 Arsitektur Aplikasi	41
Gambar 4.13 <i>Class Diagram</i> Aplikasi	42
Gambar 4.14 Perancangan antarmuka halaman registrasi	43
Gambar 4.15 Perancangan antarmuka halaman tambah transaksi	44
Gambar 4.16 Perancangan antarmuka halaman cetak transaksi	45
Gambar 5.1 Kode Program Roating Aplikasi	47
Gambar 5.2 Kode Program TransaksiController Aplikasi	49

Gambar 5.3 Kode Program Model Transaksi	52
Gambar 5.4 Kode Program View List Member	53
Gambar 5.5 Tampilan Halaman <i>Login</i>	55
Gambar 5.6 Tampilan Halaman Utama Pemilik	56
Gambar 5.7 Tampilan Halaman <i>Customer</i>	58
Gambar 5.8 <i>Input</i> Registrasi	61
Gambar 5.9 <i>Output</i> Pengujian Registrasi (Benar).....	61
Gambar 5.10 Pengecekan Pengujian Registrasi	62
Gambar 5.11 <i>Output</i> Pengujian Registrasi (Alternatif)	62
Gambar 5.12 <i>Input</i> Pengujian Registrasi (Benar)	63
Gambar 5.13 <i>Output</i> Pengujian Registrasi <i>Member</i> Baru (Benar)	64
Gambar 5.14 Pengecekan Pengujian Transaksi (Benar).....	64
Gambar 5.15 <i>Input</i> Pengujian Tambah Transaksi (Alternatif).....	65
Gambar 5.16 Halaman <i>Detail</i> Transaksi.....	66
Gambar 5.17 Tampilan Cetak Transaksi.....	66



DAFTAR TABEL

Tabel 2.1 Notasi-notasi pada Diagram Use Case.....	16
Tabel 4.1 Deskripsi Tugas Aktor	30
Tabel 4.2 <i>Use Case Scenario</i> Registrasi	31
Tabel 4.3 <i>Use Case Scenario</i> Tambah Transaksi.....	32
Tabel 4.4 <i>Use Case Scenario</i> Cetak Transaksi	33
Tabel 4.5 Tabel Transaksi	39
Tabel 4.6 Tabel Detail <i>Service</i>	40
Tabel 4.7 Tabel <i>Service</i>	40
Tabel 5.1. Fokus Pengujian.....	59
Tabel 5.2 Pengujian Registrasi (Benar)	60
Tabel 5.3. Pengujian Registrasi (Alternatif)	62
Tabel 5.4. Pengujian Tambah Transaksi (Benar).....	63
Tabel 5.5. Pengujian Tambah Transaksi (Alternatif).....	64
Tabel 5.6. Pengujian Cetak Transaksi (Benar)	65
Tabel 5.7. Hasil Pengujian	67





BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada era globalisasi sekarang ini teknologi dan komunikasi semakin berkembang dimana semua orang mengharapkan sesuatu yang serba praktis dan serba cepat. Begitu pula dengan teknologi yang sudah berkembang pesat saat ini dengan berbagai variasi tampilan dan manfaat yang disajikan secara berbeda. Salah satu contoh teknologinya yaitu sistem *booking* secara *online*. *Booking online* merupakan cara yang mudah untuk mencari dan memesan jasa atau barang melalui internet. Semua informasi mengenai *booking online* disimpan secara digital dalam sistem komputer milik perusahaan (Sentosa, 2015). Penggunaan *booking online* dapat mempermudah kegiatan operasional pada suatu perusahaan, tidak terkecuali pada penyewaan jasa *event organizer*.

Event Organizer merupakan suatu perusahaan yang bergerak di bidang jasa yang membantu *client* untuk menyelenggarakan suatu acara agar berjalan sesuai yang diinginkan oleh *client*. Salah satu penyewa jasa dibidang ini adalah CV. V-Pro. CV. V-Pro beralamat di Jalan Kemayoran No.01, Kel. Air Tawar Timur, Kec. Padang Utara, Kota Padang, Sumatera Barat dan berdiri sejak tahun 2017 yang dipimpin oleh Agung Wijaya. CV.V-Pro ini merupakan salah satu penyewaan jasa yang cukup terkenal, akan tetapi dalam kegiatan operasionalnya, termasuk dalam hal sewa CV. V-Pro *event organizer* masih menggunakan sistem *manual* yaitu *customer* datang langsung ke perusahaan untuk penyewaan serta memilih item yang diinginkan dan pihak perusahaan mencatat pilihan sewa yang dipilih oleh *customer* ke buku. Akibatnya banyak mengalami kendala seperti pelanggan yang sudah datang ke CV. V-Pro tidak dapat menyewa dikarenakan jadwal sewa bentrok dengan pelanggan lain, maupun pelanggan yang sudah memastikan jadwal melalui media komunikasi seperti via telepon, tetapi tetap tidak mendapatkan jadwal yang diinginkan. Penggunaan kertas atau buku juga menjadi kendala bagi pemilik sewa karena bertambahnya biaya yang harus dikeluarkan. Selain itu, dengan sistem yang lama *customer* diminta langsung datang ketempat penyewaan, sehingga menjadi salah satu kendala bagi *customer* dalam melakukan penyewaan. Oleh sebab itu

untuk mengatasi permasalahan yang telah disebutkan sebelumnya, maka perlu dibangun sebuah aplikasi *booking online* penyewaan jasa *event organizer* pada Cv. V-Pro yang sekiranya mampu menggantikan sistem penyewaan melalui pertemuan empat mata maupun telepon.

Adapun penelitian yang pernah dilakukan terkait dengan pembangunan aplikasi *event organizer* ini adalah penelitian oleh Eka Chandra Ramdhani, dkk pada tahun 2018 yang berjudul “Sistem Informasi Penyewaan Peralatan *Event Organizer* Berbasis *Web* pada PT. Adecon Jakarta”. Tujuan dilakukan penelitian tersebut adalah agar mempermudah konsumen dalam melakukan pemesanan peralatan event organizer serta memudahkan dalam memberikan pelayanan yang optimal terhadap konsumen (Eka, 2018). Kedua penelitian yang dilakukan oleh Fahrudin Prasetyo, dkk pada tahun 2017 yang berjudul “Perancangan Sistem Informasi Pemesanan Event Organizer Berbasis Website Menggunakan Framework CodeIgniter (Studi Kasus : PT. Munji Organizer Semarang)”. Tujuan dilakukan penelitian ini adalah untuk mempermudah dalam pengelolaan data pemesanan maupun keuangan (Fahrudin, 2017). Ketiga penelitian yang dilakukan oleh Romaida Yuliana Sagala, pada tahun 2017 yang berjudul “Sistem Informasi Pengolahan Jasa Event Organizer”. Tujuan dilakukan penelitian ini adalah untuk mempermudah klien dalam menciptakan ide suatu *event* yang diadakan, dengan mengkustomisasikan kebutuhan *event* (Romaida, 2017)

Berdasarkan uraian yang telah dipaparkan, maka penulis mengusulkan perbedaan sistem dengan penambahan sistem *budget*, karena penelitian sebelumnya tidak menggunakan sistem *budget*. Maka penulis memutuskan untuk memberi judul penelitian tugas akhir ini dengan judul “**Pembangunan Sistem *Booking Online* Dalam Penyewaan Jasa *Event Organizer* Berbasis *Web* Pada CV. V-Pro Kota Padang**”. Dengan aplikasi berbasis *web* yang dibangun, diharapkan mampu mempermudah kegiatan operasional yang ada pada CV. V-Pro dan juga membantu customer melalui kemudahan dalam penyewaan jasa *event organizer*.

1.2 Rumusan Masalah

Berdasarkan uraian latar belakang diatas, maka rumusan masalah pada laporan ini adalah bagaimana membangun sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV. V-Pro.

1.3 Batasan Masalah

Berdasarkan rumusan masalah seperti yang dijelaskan sebelumnya, maka batasan masalah dalam tugas akhir ini adalah:

1. Aplikasi hanya bisa diakses melalui *web*.
2. Aplikasi menggunakan model waterfall.
3. Aplikasi *web* dibangun dengan menggunakan *framework* Laravel dan menggunakan *database* MySQL.
4. Pengujian aplikasi hanya sebatas memeriksa ketersediaan fungsional dan kesesuaian dengan rancangan sistem yang diusulkan.
5. Aplikasi ini memiliki lima belas fungsional.

1.4 Tujuan Penelitian

Adapun tujuan penelitian ini dilakukan Antara lain:

1. Menganalisis kebutuhan dan mengetahui gambaran pembangunan sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV.V-Pro.
2. Membangun sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV.V-Pro.
3. Melakukan pengujian terhadap sistem yang dibangun, sehingga menghasilkan sistem yang sesuai dengan kebutuhan.

1.5 Manfaat Penelitian

Manfaat dari penelitian ini adalah untuk mempermudah costumer dalam melakukan *booking online* dalam penyewaan jasa *event organizer* pada CV. V-Pro.

1.6 Sistematika Penulisan

Sistematika penulisan yang digunakan untuk penyusunan tugas akhir ini adalah sebagai berikut:

BAB I PENDAHULUAN

Bab ini menjelaskan tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, serta sistematika penulisan laporan.

BAB II TINJAUAN PUSTAKA

Bab ini berisi tentang landasan teori dan informasi pendukung yang digunakan untuk penelitian ini.

BAB III METODE PENELITIAN

Bab ini menjelaskan tentang objek penelitian, metode pengumpulan data, metode pengembangan sistem yang digunakan, dan *flow chart* penelitian.

BAB IV ANALISIS DAN PERANCANGAN SISTEM

Bab ini menjelaskan tentang pemodelan analisis sistem menggunakan *tools* seperti, *Business Process Model Notation* (BPMN), *use case diagram*, *scenario diagram*, *sequence diagram*, *class analysis*, perancangan basis data, struktur basis data dan tabel, *class diagram*, arsitektur aplikasi dan perancangan antarmuka.

BAB V IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini menjelaskan tentang pengimplementasian aplikasi ke dalam bahasa pemrograman berdasarkan analisis dan perancangan, serta pengujian terhadap hasil implementasi sistem.

BAB VI PENUTUP

Berisi tentang kesimpulan terhadap hasil penelitian dan saran untuk pengembangan sistem kedepannya.

1.7 Analisis Aplikasi Sejenis

Berisi tentang fungsional penelitian yang pernah dilakukan terkait dengan pembangunan aplikasi *event organizer*. Yang pertama, fungsional penelitian yang dilakukan oleh Eka Chandra Ramdhani yaitu mengelola data customer, mengelola

transaksi, mengelola pembayaran dan konfirmasi pembayaran. Kedua, fungsional penelitian yang dilakukan oleh Fahrudin Prasetyo yaitu administrasi dapat melakukan pengelolaan data berita, data paket, data pemesanan, data user, data pelanggan dan mengkonfirmasi data pemesanan dari pelanggan. Ketiga, fungsional penelitian yang dilakukan oleh Romaida Yuliana yaitu sistem menampilkan tampilan home bagian *clien*, *clien* melakukan *booking* pada sistem, *clien* melakukan konfirmasi pembayaran. Sistem menampilkan tampilan pembayaran bagian *clien*. *clien* meng-input kode *booking*. Kemudian administrator me-review kelengkapan dokumen pembookingan selanjutnya menerima data konfirmasi pembayaran *event*. *Clien* dan Administrator masing-masing memvalidasi kontrak kerjasama secara bertemu langsung. Selanjutnya *event* dilaksanakan.



BAB II

TINJAUAN PUSTAKA

Bab ini menjelaskan mengenai landasan teori, perangkat lunak pendukung, alat pemodelan sistem yang digunakan, dan alur sistem *booking online* dalam penyewaan jasa *event organizer* yang sedang berlangsung.

2.1 *Booking Online*

Booking berasal dari kata *book* yang artinya adalah pemesanan atau pembukuan. *Booking* dilakukan dengan alasan agar tempat yang dikehendaki tidak diambil oleh orang lain. Sedangkan *online* adalah tersambung ke suatu jaringan komputer atau dapat diakses oleh komputer. *Online* juga dapat diartikan sebagai sedang terhubung (*in progress*) (Online, 2011) Tujuan dari reservasi atau *booking online* adalah (Sulistiyanto, 2012)

1. Orang yang ingin memesan tempat lewat *internet* hanya membutuhkan akses *internet* dan interfacenya menggunakan *web browser*.
2. Menjadikan *website* reservasi tidak hanya untuk wadah pemesanan semata tetapi menjadi tempat berkumpulnya komunitas dengan membangun basis komunikasi.
3. Pengelolaan yang berorientasi pada pelayanan, kombinasi konsepsi pelayanan konvensional dan virtual: *responsive* (respon yang cepat dan ramah), dinamis, informatif dan komunikatif.
4. Informasi yang *up to date*, komunikasi multi-arah yang dinamis
5. Model: kartu kredit atau transfer.

2.2 *Penyewaan Jasa*

Penyewaan adalah proses, cara perbuatan menyewa atau menyewakan (KBBI, 2017). Penyewaan adalah pemindahan hak guna pakai suatu barang, benda atau jasa dari pihak pemilik barang atau benda kepada pihak penyewa dalam jangka waktu tertentu dengan uang oleh pihak penyewa kepada pihak pemilik barang atau benda sesuai dengan perjanjian kedua belah pihak (Subekti, 2008).

Unsur-unsur sewa-menyewa (Wirjono, 2009):

1. Adanya pihak yang menyewa dan pihak penyewa

2. Adanya konsensus antara kedua belah pihak yang melakukan sewa
3. Adanya objek sewa menyewa, yaitu barang atau tempat
4. Adanya kewajiban dari pihak yang menyewakan untuk menyerahkan kenikmatan kepada pihak penyewa atas suatu benda
5. Adanya kewajiban dari penyewa untuk menyerahkan uang kepada yang menyewakan

Jasa adalah perbuatan yang memberikan segala sesuatu yang diperlukan orang lain (KBBI, 2017). Maka dapat disimpulkan, penyewaan jasa merupakan proses peminjaman yang memberikan segala sesuatu yang diperlukan orang lain.

2.3 *Event Organizer (EO)*

Menurut Kennedy, *event* dalam arti sempit dapat diartikan sebagai pameran, pertunjukan, festival dengan syarat ada penyelenggara, peserta dan pengunjung. Dalam arti luas, *even*) dapat diartikan sebagai suatu tempat agar mereka memperoleh informasi atau pengalaman penting serta tujuan lain yang diharapkan oleh penyelenggara. (Kennedy, 2009)

Menurut (Ruslan, 2007), tujuan dari diadakannya suatu *event* adalah:

1. Meningkatkan pengetahuan atau kesadaran khalayak terhadap perusahaan atau produk yang ditampilkan.
2. Memperoleh publikasi positif melalui komunikasi timbal balik.
3. Menunjukkan niat baik dari perusahaan atau produk yang diwakilinya dan sekaligus memberikan citra positif pada masyarakat sebagai publik sasarannya.
4. Mempertahankan penerimaan masyarakat.
5. Memperoleh rekanan baru melalui *event* yang dirancang secara menarik dan kreatif. (Pudjiastuti, 2010)

Event pada dasarnya merupakan suatu perkumpulan orang untuk merayakan sesuatu yang bersifat pribadi atau umum, ritual atau pengingatan. *Event* telah menjadi fitur utama dari kampanye politik dan bisnis. *Event* digunakan sebagai alat pemasaran untuk menciptakan kesadaran dan daya tarik yang lebih besar atau instan melalui iklan, kampanye dan saluran komunikasi (Glenn McCartney; 2010).

EO adalah Usaha dalam bidang jasa yang ditunjuk secara resmi oleh *client* untuk mengorganisasikan rangkaian acara, dimulai dari proses pembuatan konsep, perencanaan, persiapan, eksekusi hingga rangkaian acara selesai dalam rangka membantu *client* mewujudkan tujuan yang diharapkan melalui rangkaian acara yang diadakan (Setyawan, 2009) . Peran *event organizer* yaitu membantu mewujudkan maksud dan tujuan acara penyandang dana dalam bentuk kreatif acara serta eksekusinya (Megananda, 2009)

2.4 Perangkat Lunak Pendukung

Berikut merupakan penjelasan tentang perangkat lunak pendukung dan *tools* yang digunakan dalam penelitian ini.

2.4.1 Website

Website atau lazim disingkat *web* adalah halaman informasi yang disediakan melalui jalur *internet* sehingga bisa diakses di seluruh dunia selama terkoneksi dengan jaringan *internet*. *Web* adalah salah satu alat komunikasi *online* yang menggunakan media *internet* dalam pendistribusiannya. Apapun bahasanya, yang pasti kita semua setuju bahwa *website* merupakan kumpulan halaman yang menampilkan informasi data teks, data gambar diam atau gerak, data animasi, suara, video dan gabungan dari semuanya, baik yang bersifat statis maupun dinamis yang membentuk satu rangkaian yang saling terkait dimana masing-masing dihubungkan dengan jaringan-jaringan halaman *hyperlink* (Yoshiko, 2015)

2.4.2 PHP (*PHP Hypertext Preprocessor*)

PHP adalah sebuah bahasa interpreter yang memiliki kemiripan dengan bahasa C dan Perl, mempunyai kesederhanaan dalam perintah dan digunakan untuk pembuatan aplikasi web. PHP dibuat oleh Rasmus Lerdoff pada musim gugur tahun 1994. Pada awalnya, PHP digunakan pertama kali pada situsnya untuk mencatat siapa saja yang berkunjung untuk melihat biodatanya. Dengan menggunakan PHP, maka *maintenance* situsnya menjadi lebih mudah. Versi pertama yang dirilis pada tahun 1995 dikenal dengan tool *Personal Home Page*, terdiri atas *engine parser* yang sangat sederhana, hanya mengerti beberapa *makro* khusus dan beberapa

utilitas yang sering digunakan pada halaman-halaman situs, seperti buku tamu, *hit counter*, dan lainnya menurut (Fitriyani, 2017)

Menurut dokumen resmi PHP, PHP merupakan singkatan dari *PHP Hypertext Preprocessor*. PHP merupakan bahasa berbentuk skrip yang ditempatkan dalam *server* dan diproses di server. Hasilnya yang dikirimkan ke *clien*, tempat pemakai menggunakan *browser*. Artinya semua sintaks yang kita berikan akan sepenuhnya dijalankan pada server sedangkan yang dikirimkan ke *browser* hanya hasilnya saja. PHP dapat digunakan bersama dengan HTML sehingga memudahkan dalam membuat aplikasi *web* dengan cepat. PHP dapat digunakan untuk menciptakan *dynamic website* baik itu yang memerlukan penggunaan *database* ataupun tidak. Seperti halnya dengan program *open source* lainnya, PHP di buat di bawah *General Public License (GNU)* menurut (Kadir, 2008) dalam (Hardyanti, 2011) (Kamil, 2013)

2.4.3 Framework

Definisi *framework* bervariasi menurut literatur yang ada. Namun definisi *framework* tersebut mengacu kepada sebuah kerangka aplikasi yang dapat diperluas dan digunaulang. *Framework* didefinisikan sebagai sebuah aplikasi semi *complete* yang dapat digunakan kembali dan dapat dimodifikasi untuk menghasilkan aplikasi tertentu (Kamil, 2013). *Framework* juga didefinisikan sebagai kerangka sebuah aplikasi yang dapat dimodifikasi dan disesuaikan oleh pengembang aplikasi (Kamil, 2013). Penggunaan *framework* untuk pengembangan aplikasi ini bertujuan untuk mempercepat proses pengembangan aplikasi.

2.4.4 Laravel

Laravel adalah sebuah *framework* PHP yang dirilis dibawah lisensi MIT, dibangun dengan konsep MVC (*model view controller*). *Laravel* adalah pengembangan *website* berbasis MVP yang ditulis dalam PHP yang telah dirancang untuk meningkatkan kualitas perangkat lunak dengan mengurangi biaya, baik biaya pengembangan awal dan biaya pemeliharaan, serta memberikan *sintaks* yang ekspresif dan jelas, serta menyediakan fungsi inti yang akan menghemat waktu pengembangan. *Laravel* memudahkan pengembang dalam memahami basis kode yang lebih kecil, serta menerapkan solusi dengan cara yang bersih, sederhana, dan

elegan. *Laravel* adalah salah satu dari beberapa *framework* PHP yang menawarkan *code* modular, saat ini *laravel* menjadi tren *framework* yang banyak digunakan oleh pengembang *web* di dunia (Pertiwi, 2011) (McCool, 2012). *Model View Controller* pertama kali dipublikasikan oleh peneliti XEROX PARAC yang bekerja dalam pembuatan bahasa pemrograman Smalltalk sekitar tahun 1970-1980. MVC adalah sebuah metode untuk membuat sebuah aplikasi dengan memisahkan data (Model) dari tampilan (View) dan cara bagaimana memprosesnya (Controller). Dalam implementasinya kebanyakan *framework* dalam aplikasi *website* adalah berbasis arsitektur MVC. MVC memisahkan pengembangan aplikasi berdasarkan komponen utama yang membangun sebuah aplikasi seperti manipulasi data, antarmuka pengguna, dan bagian yang menjadi kontrol dalam sebuah aplikasi web. Bagian Model View Controller (Pertiwi, 2011)

1. Model: mewakili struktur data, biasanya model berisi fungsi-fungsi yang membantu seseorang dalam pengelolaan basis data seperti memasukkan data ke basis data, pembaruan data dan lain-lain.
2. View: adalah bagian yang mengatur tampilan ke pengguna, bisa di katakan berupa halaman *web*.
3. Controller: merupakan bagian yang menjembatani *model* dan *view*, *controller* berisi perintah-perintah yang berfungsi untuk memproses suatu data dan mengirimkannya ke halaman *web*.

2.4.4.1 Kelebihan *Laravel*

Meskipun terhitung masih baru tapi *laravel* menjadi salah satu *PHP framework* yang sedang populer pada saat ini. Seperti yang dituliskan Taylor Otwell, *laravel* sendiri merupakan sebuah *framework PHP* yang mempunyai sifat *open source*. Lebih lanjut Taylor Otwell menyatakan bahwa *laravel* didesain khusus dengan maksud untuk membantu *developer* terutama dalam membuat web dengan sintak sederhana dan ekspresif. Kelebihan dari *laravel* adalah sebagai berikut (McCool, 2012):

1. Sintak lebih mudah dipahami karena lebih simple.
2. Full MVC.

3. *Composer support* adalah sebuah *dependency manager* untuk PHP dimana anda bisa menginstall suatu *library* melalui *composer*. Sehingga *composer* secara otomatis menginstall *library* lain yang dibutuhkan, tanpa perlu mendownload *library* satu persatu. Jadi dengan *composer* ini anda dapat mengupdate aplikasi yang anda kembangkan dengan mudah.
4. *Blade templating* memungkinkan *developer* untuk membuat template yang lebih terstruktur dan dinamis.

2.4.4.2 Fitur *Laravel*

Laravel juga menyediakan beberapa fitur yang bisa untuk mempermudah dalam penggunaannya, beberapa fitur tersebut adalah sebagai berikut (McCool, 2012):

1. *Bundles*, yaitu sebuah fitur dengan sistem pengemasan modular dan tersedia beragam di aplikasi.
2. *Eloquent ORM*, merupakan penerapan PHP lanjutan menyediakan metode internal dari pola *active record* yang mengatasi masalah pada hubungan objek *database*.
3. *Application Logic*, merupakan bagian dari aplikasi, menggunakan *controller* atau bagian *Route*.
4. *Reverse Routing*, mendefinisikan relasi atau hubungan antara *Link* dan *Route*.
5. *Restful Controllers*, memisahkan logika dalam melayani HTTP GET and POST.
6. *Class Auto Loading*, menyediakan *loading* otomatis untuk *class* PHP.
7. *View Composer*, adalah kode unit logikal yang dapat dieksekusi ketika *view* sedang *loading*.
8. *IoC Container*, memungkinkan obyek baru dihasilkan dengan pembalikan *controller*.
9. *Migration*, menyediakan sistem kontrol untuk skema *database*.
10. *Unit Testing*, banyak tes untuk mendeteksi dan mencegah regresi.

2.4.5 MySQL

MySQL adalah sebuah implementasi dari sistem manajemen basisdata relasional (RDBMS) yang didistribusikan secara gratis dibawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basisdata yang telah ada sebelumnya SQL (*Structured Query Language*). SQL adalah sebuah konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis. Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya (Hardyanti, 2011).

MySQL mendukung operasi basisdata *transaksional* maupun operasi basisdata *non transaksional*. Pada modus operasi nontransaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak peladen basisdata kompetitor lainnya. Namun demikian pada modus *non-transaksional* tidak ada jaminan atas *reliabilitas* terhadap data yang tersimpan, karenanya modus *non-transaksional* hanya cocok untuk jenis aplikasi yang tidak membutuhkan *reliabilitas* data seperti aplikasi *blogging* berbasis *web* (*wordpress*), CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basisdata *transaksional*, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus *transaksional* tidak secepat unjuk kerja pada modus *non-transaksional*.

2.5 Tools Pemodelan Sistem

Tools pemodelan sistem merupakan alat yang digunakan dalam menganalisis dan merancang sistem. Adapun alat yang digunakan untuk pemodelan dalam penelitian ini *Business Process Modelling Notation* (BPMN) untuk menggambarkan logika dari proses bisnis, *Unified Modelling Language* (UML) untuk memodelkan *software* yang dibangun, dan *Entity Relationship Diagram*

(ERD) dalam menggambarkan struktur *database* yang dibangun. Berikut merupakan penjelasan dari masing-masing alat pemodelan tersebut:

2.5.1 *Business Process Modelling Notation (BPMN)*

BPMN adalah standar untuk memodelkan proses bisnis dan proses-proses *web services*. BPMN dirancang agar mudah digunakan dan dipahami serta memiliki kemampuan untuk memodelkan proses bisnis yang kompleks dan secara spesifik dirancang dengan mempertimbangkan *web services*. Notasi yang digunakan dalam BPMN mudah dipahami oleh semua pengguna bisnis. BPMN lebih cenderung menggambarkan proses dari pandangan seorang analisis bisnis yang lebih mengarah penyampaian kebutuhan yang akan dideskripsikan kepada analisis IT dan pengembang *software*. BPMN menyediakan BPD (*Business Process Diagram*), yang berlandaskan pada teknik *flowchart* yang digunakan untuk membuat model proses bisnis (Aini, 2014).

Sebuah BPD terdiri dari sekumpulan elemen grafis, dan kategori dasar dari elemen BPD adalah (Dewi, Indahyanti, & S, 2012):

a) *Flow Objects*

Flow objects dibagi menjadi 3, yaitu *event*, *activity* dan *gateway*.

- *Event*, digambarkan dengan sebuah lingkaran dan merupakan sesuatu yang “terjadi” selama berlangsungnya proses bisnis. *Event* mempengaruhi aliran proses dan memiliki penyebab (*trigger*) atau hasil (*result*). *Event* terbagi atas 3 tipe yaitu *start*, *intermediate*, dan *end*. Simbol *event* ini dapat dilihat pada gambar 2.1.



Gambar 2.1 Simbol *Event* (Dewi, Indahyanti, & S, 2012)

- *Activity*, ditunjukkan dengan persegi panjang dengan ujung-ujung bulat dan merupakan bentuk umum untuk pekerjaan yang dilakukan

oleh perusahaan. Tipe dari aktivitas adalah *task* dan *sub process* yang dibedakan dengan tanda + pada bagian tengah bawah dari bentuk tersebut.

- *Gateway*, digambarkan dengan bentuk seperti belah ketupat dan digunakan untuk mengontrol percabangan dan penggabungan *Sequence Flow*. Jadi, *gateway* menentukan keputusan tradisional, penggabungan, dan penggabungan aliran. Internal Markers akan menentukan perilaku dari kontrol. Simbol *activity* dan *gateway* ini dapat dilihat pada gambar 2.2.



Gambar 2.2 Simbol *Activity* dan *Gateway* (Dewi, Indahyanti, & S, 2012)

b) *Connecting Objects*

Connecting Objects adalah elemen yang menghubungkan *flow object*. *Connecting Objects* memiliki 3 jenis elemen, yaitu:

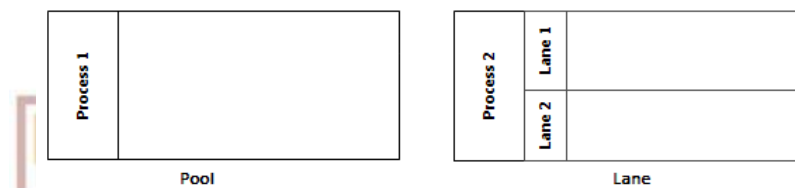
- *Alur Sequence (Sequence Flow)*
Digunakan untuk menunjukkan urutan yang kegiatannya dilakukan dalam sebuah proses.
- *Alur Pesan (Message Flow)*
Digunakan untuk menunjukkan aliran pesan antara dua entitas yang siap untuk mengirim dan menerima.
- *Asosiasi (Association)*
Digunakan untuk asosiasi data, informasi dan artefak dengan aliran benda. Tipe-tipe *connecting object* ini dapat dilihat pada gambar 2.3.



Gambar 2.3 Tipe-Tipe *Connecting Object* (Dewi, Indahyanti, & S, 2012)

c) *Swimlanes*

Swimlanes digambarkan dengan bentuk garis yang memisahkan dan mengelompokkan aktor. *Swimlanes* digunakan sebagai mekanisme untuk membagi kategori visual yang menggambarkan kemampuan fungsional atau tanggung jawab yang berbeda. Lanes digunakan untuk mengatur dan mengkategorikan aktivitas. Simbol *pool* dan *lane* dalam *swimlane* dapat dilihat pada gambar 2.4.



Gambar 2.4 Simbol *Pool* dan *Lane* dalam *Swimlane* (Dewi, Indahyanti, & S, 2012)

d) *Artifacts*

Artifacts adalah elemen yang digunakan untuk memberikan informasi tambahan dari sebuah proses. Versi BPMN saat ini memiliki 3 tipe *artifacts*, yaitu:

- *Data object*: mekanisme untuk menunjukkan bagaimana data dibutuhkan atau diproduksi oleh aktivitas. *Data object* dihubungkan dengan aktivitas melalui *associations*.
- *Group*: diwakili dengan persegi panjang dengan ujung bulat yang digambarkan dengan garis putus-putus. *Group* dapat digunakan untuk tujuan dokumentasi atau analisis, tetapi tidak mempengaruhi *Sequence Flow*.
- *Annotation*: mekanisme untuk pemodelan memberikan informasi teks tambahan untuk pembaca dari diagram BPMN (Dewi, Indahyanti, & S, 2012).

2.5.2 UML (*Unified Modeling Language*)

UML adalah keluarga notasi grafis yang didukung oleh meta-model tunggal, yang membantu pendeskripsian dan desain sistem perangkat lunak, khususnya sistem yang dibangun menggunakan pemrograman berorientasi objek

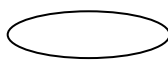
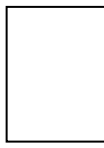
(OOP). UML juga memberikan standar penulisan sebuah sistem blueprint, yang meliputi konsep bisnis proses, penulisan kelas-kelas dalam bahasa program yang spesifik, skema *database* dan komponen-komponen yang diperlukan dalam sistem *software* menurut (Fowler, 2005) dalam (Haryani, 2015)


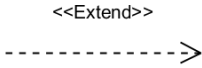
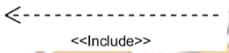
Pada pembangunan aplikasi ini UML yang digunakan adalah *use case diagram*, *sequence diagram* dan *class diagram*. Berikut merupakan penjelasan dari diagram-diagram yang digunakan:

a. *Use Case Diagram*

Use case merupakan diagram (UML) yang menggambarkan interaksi yang terjadi antara sistem, sistem eksternal dan para penggunanya (*user*). Bisa dikatakan juga bahwa *use case diagram* menggambarkan pengguna-pengguna yang menggunakan sistem beserta caranya berinteraksi dengan (Riadi, 2013). Notasi-notasi yang digunakan untuk membuat *use case diagram* dapat dilihat pada Tabel 2.1.

Tabel 2.1. Notasi-notasi pada Diagram *Use Case* (Riadi, 2013)

No	Gambar	Nama	Keterangan
1.		<i>Actor</i>	Menspesifikasikan himpunan peran yang pengguna mainkan ketika berinteraksi dengan use case.
2.		<i>Use case</i>	Deskripsi dari aksi-aksi yang ditampilkan sistem yang menghasilkan suatu hasil yang terukur bagi suatu aktor
3		<i>System</i>	Menspesifikasikan paket yang menampilkan sistem secara terbatas.

4		<i>Association</i>	Garis yang menghubungkan antara aktor dengan <i>use case</i> atau objek yang satu dengan objek lainnya.
5		<i>Extend</i>	Menspesifikasikan bahwa <i>use case</i> target memperluas perilaku dari <i>use case</i> sumber pada suatu titik yang diberikan.
6		<i>Include</i>	Menspesifikasikan bahwa <i>use case</i> sumber secara eksplisit.

Tabel 2.1. Notasi-notasi pada Diagram *Use Case* (Riadi, 2013)

b. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa message yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).

Sequence diagram biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah event untuk menghasilkan *output* tertentu. Diawali dari apa yang men-*trigger* aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan (Dharwiyanti, 2003)

c. *Use Case Scenario*

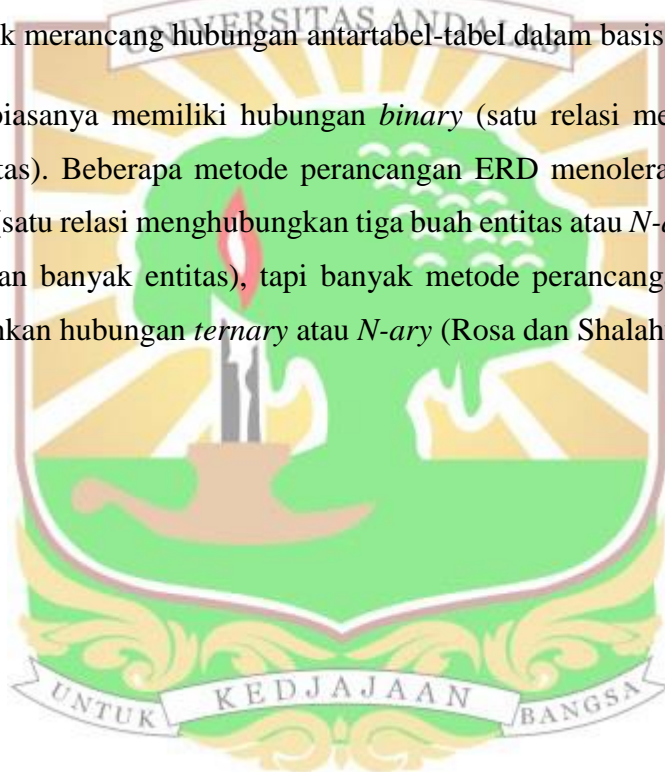
Use case scenario merupakan uraian deskripsi dari *use case diagram* sehingga pengguna UML bias mengetahui detail dari proses yang ada pada *use case diagram* (Mulyani, 2016). Di dalam *use case scenario*, akan diberi uraian nama aktor yang berhubungan dengan *use case* tersebut, tujuan dari *use case*, deskripsi global tentang *use case*, pra-kondisi yang harus dipenuhi, dan pasca-kondisi yang diharapkan setelah berjalannya fungsional *use case* (Stone, 2006).

Dalam *Use Case scenario* terdapat skenario normal dan skenario alternatif. Skenario normal adalah skenario bila sistem berjalan dengan normal tanpa terjadi kesalahan. Skenario alternatif adalah skenario bila sistem tidak berjalan dengan normal. *Use Case scenario* merupakan dasar dari pembuatan *Sequence Diagram*.

2.5.3 *Entity Relationship Diagram*

Pemodelan awal basis data yang paling banyak digunakan adalah menggunakan ERD. ERD dikembangkan berdasarkan teori himpunan dalam bidang matematika (Rizqi, 2017). ERD digunakan untuk pemodelan basis data relational untuk merancang hubungan antartabel-tabel dalam basis data.

ERD biasanya memiliki hubungan *binary* (satu relasi menghubungkan dua buah entitas). Beberapa metode perancangan ERD menoleransi hubungan relasi *ternary* (satu relasi menghubungkan tiga buah entitas atau *N-ary*, satu relasi menghubungkan banyak entitas), tapi banyak metode perancangan ERD yang tidak mengizinkan hubungan *ternary* atau *N-ary* (Rosa dan Shalahudin, 2013)



BAB III

METODOLOGI PENELITIAN

Pada bab ini dijelaskan mengenai objek kajian, metode pengumpulan data, dan metode pembangunan aplikasi. Kemudian juga dipaparkan *flowchart* penelitian yang digunakan dalam pembangunan sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV. V-Pro Kota Padang.

3.1 Objek Kajian

Objek kajian dalam penelitian ini adalah pembangunan sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV. V-Pro Kota Padang yang beralamat di Jalan Kemayoran No.01, Kel. Air Tawar Timur, Kec. Padang Utara, Kota Padang, Sumatera Barat. Pemilihan objek penelitian ini dikarenakan masih belum terkomputerisasi yaitu *customer* harus pergi ke lokasi untuk melakukan penyewaan jasa *event organizer* dan proses pencatatan penyewaan masih manual menggunakan kertas hal ini dirasakan tidak efektif dan efisien.

3.2 Metode Pengumpulan Data

Pada tahap ini dilakukan pengumpulan data yang bertujuan untuk memperoleh seluruh informasi terkait dengan aplikasi yang akan dibangun. Metode ini dilakukan dengan mempelajari dan mengumpulkan data langsung ke lapangan. Studi lapangan terdiri dari observasi, wawancara, analisis dokumen dan studi literatur:

- a. Observasi atau survey dilakukan dengan cara pengamatan langsung terhadap objek penelitian yang telah ditetapkan. Seperti alur *booking* dalam penyewaan jasa *event organizer* pada CV. V-Pro.
- b. Wawancara
Pengumpulan data dengan melakukan wawancara bertujuan untuk mendapatkan informasi lebih rinci dengan mencari tahu kebutuhan fungsional yang dibutuhkan pada aplikasi yang akan dibangun.

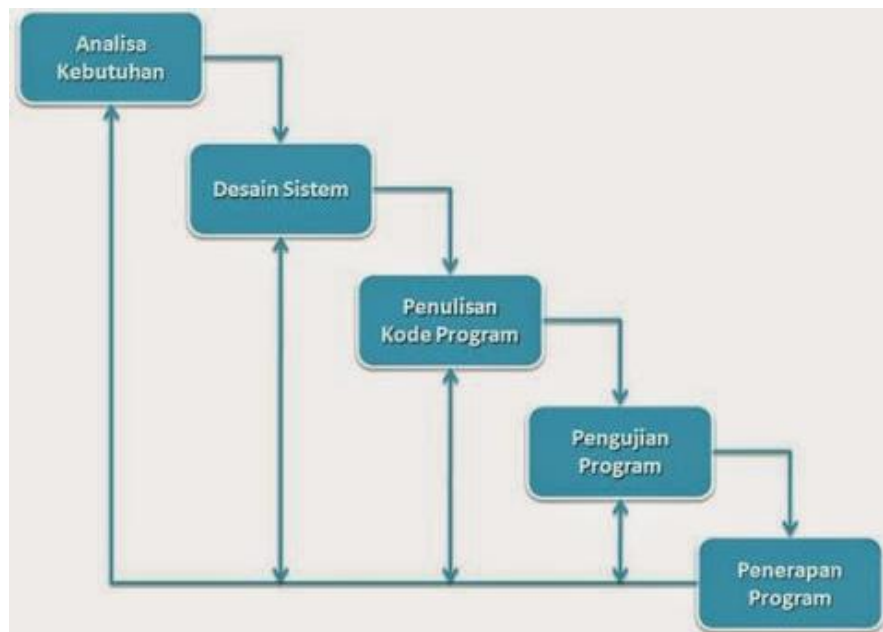
- c. Studi Literatur

Pengumpulan data dilakukan dengan mempelajari literatur terkait dengan penelitian yang bersumber dari situs internet, jurnal ilmiah, dan sumber lainnya.

3.3 Metode Pengembangan Sistem

Pada penelitian ini, penulis menggunakan metode *waterfall*. *Waterfall* adalah sebuah model perkembangan perangkat lunak yang dilakukan secara

sekuensial, dimana satu tahap dilakukan setelah tahap sebelumnya selesai dilaksanakan (Pressman, 2001). Metode *waterfall* dapat dilihat pada gambar 3.1.



Gambar 3.1 Metode *Waterfall* (Pressman, 2001).

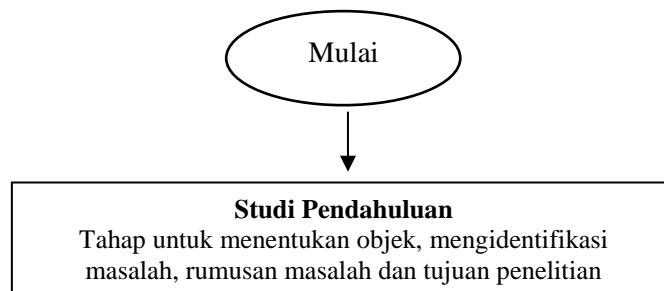
Berikut adalah penjelasan dari tahap-tahap yang dilakukan di dalam model ini menurut:

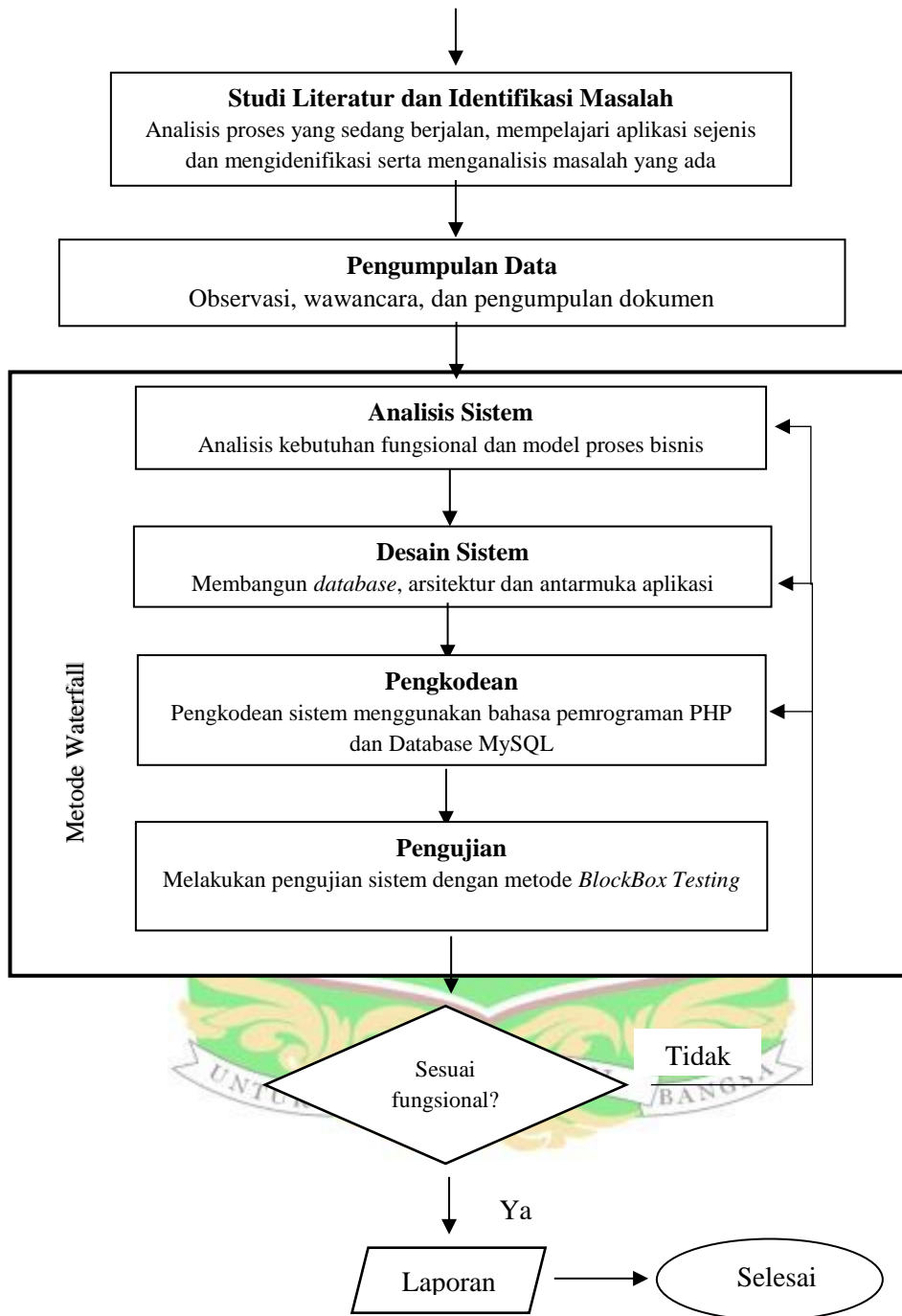
- a. Analisa kebutuhan, tahap ini merupakan proses identifikasi terhadap permasalahan yang terjadi pada CV. V-Pro, lalu dilakukan analisis dan evaluasi. Selanjutnya, dilakukan analisis terhadap kebutuhan yang diperlukan untuk pengembangan aplikasi. Proses analisis ini digambarkan dengan BPMN (*Business Process Model Notation*).
- b. Desain sistem, tahap ini dilakukan perancangan semua kebutuhan yang telah dianalisa pada tahapan sebelumnya. Perancangan yang dilakukan terdiri dari perancangan ERD (*Entity Relational Diagram*), arsitektur perangkat lunak, perancangan antarmuka dengan menggunakan *tools* UML.
- c. Penulisan kode program, hasil dari tahap *design* harus diubah kedalam bahasa pemrograman melalui proses *coding* dan menjadi satu program. Implementasi dilakukan menggunakan bahasa pemrograman PHP, HTML, *javascript*, dan *database MySQL*.

- d. Pengujian program, pengujian dilakukan untuk memastikan sistem telah terpenuhi dan sesuai dengan kebutuhan yang diharapkan.

3.4 *Flowchart* Metodologi Penelitian

Bagian ini akan dijelaskan langkah-langkah yang dilakukan untuk pembangunan sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV. V-Pro. Ada beberapa tahapan pada *flow chart* penelitian ini, yaitu studi pendahuluan, studi literatur, pengumpulan data, analisis sistem, pembuatan desain sistem, pengkodean, pengujian sistem, dan pembuatan laporan. *Flow chart* penelitian ini dapat dilihat pada Gambar 3.2.





Gambar 3.2. Flowchart Penelitian

Adapun penjelasan dari *Flow Chart* penelitian yang dilakukan adalah sebagai berikut:

1. Studi pendahuluan

Studi pendahuluan berupa penetapan objek dari penelitian dan pengidentifikasian masalah-masalah yang akan dijadikan topik dalam penelitian. Dalam penelitian ini objek yang ditetapkan adalah *event organizer* pada CV. V-Pro yang berada di Jalan Kemayoran No.01, Kel. Air Tawar Timur, Kec. Padang Utara, Kota Padang, Sumatera Barat, dengan pokok utama permasalahan yaitu bagaimana menerapkan pembangunan sistem *booking online* dalam penyewaan *event organizer* pada objek yang dikaji.

2. Studi literatur

Pada tahap ini dilakukan pengumpulan informasi dan penelitian yang telah dilakukan sebelumnya sebagai data acuan dalam melakukan penelitian. Didapatkan beberapa studi literatur yang digunakan dalam penelitian ini, diantaranya penelitian yang dilakukan oleh Eka Chandra Ramdhani, dkk pada tahun 2018 yang berjudul *Sistem Informasi Penyewaan Peralatan Event Organizer Berbasis Web* pada PT. Adecon Jakarta dan penelitian yang dilakukan oleh Fahrudin Prasetyo, dkk pada tahun 2017 yang berjudul *Perancangan Sistem Informasi Pemesanan Event Organizer Berbasis Website Menggunakan Framework CodeIgniter (Studi Kasus : PT. Munji Organizer Semarang)*.

3. Pengumpulan data

Pada tahap ini berupa pengumpulan data-data yang dibutuhkan baik data utama maupun data penunjang yang berguna dalam penelitian. Teknik pengumpulan data yang digunakan adalah dengan cara observasi langsung ke lokasi objek penelitian, wawancara dengan pemilik CV. V-Pro dan pengumpulan dokumen.

4. Analisis sistem

Pada tahapan ini yang dilakukan adalah menganalisis proses yang sedang berjalan pada objek yang telah ditentukan sebelumnya, dan dalam tahapan ini juga mendefinisikan kebutuhan fungsional usulan yang akan

diimplementasikan pada objek yang ada. Pada tahap ini ditentukan kebutuhan fungsional atau fitur sistem, batasan sistem dan tujuan sistem yang ditentukan melalui wawancara sebelumnya. Analisis fungsional sistem digambarkan dengan beberapa model yaitu *use case diagram*, *use case scenario*, *sequence diagram*, dan *class analysis*.

5. Desain Sistem

Tahap desain ini berupa perancangan sistem. Perancangan sistem meliputi perancangan *database*, perancangan *user Interface*, dan arsitektur IT untuk pembangunan sistem *booking online* dalam penyewaan jasa *event organizer* di CV. V-Pro. Tools yang digunakan adalah *Balsamiq Mockups* untuk merancang *User Interface* dan *MySQL* untuk basis data atau *database*.

6. Pengkodean

Pada tahapan ini dibuat kode program dalam membangun sistem *booking online* dalam penyewaan jasa *event organizer*. Program komputer yang dibangun yaitu aplikasi berbasis *web*. Model aplikasi berbasis *web* dibangun dengan menggunakan bahasa pemrograman HTML, PHP, dan Javascript, serta menggunakan *framework* Laravel sebagai kerangka kerja.

7. Pengujian sistem

Pengujian dilakukan untuk melihat kesesuaian kebutuhan fungsional dengan aplikasi yang dirancang menggunakan *blockbox testing*. Jika sesuai dengan kebutuhan, maka dilanjutkan ketahap berikutnya, jika tidak sesuai maka kembali ke desain sistem.

8. Laporan

Pada tahapan ini dilakukan pembuatan laporan tentang analisis, perancangan dan pengujian sistem yang dibangun. Laporan ini dapat digunakan sebagai data acuan penelitian selanjutnya karena tahapan laporan ini menghasilkan informasi mengenai analisis, pengembangan hingga implementasi dan pengujian sistem yang dibangun pada penelitian ini.

BAB IV

ANALISIS DAN PERANCANGAN SISTEM

Pada bab ini dijelaskan hasil dari analisis dan perancangan aplikasi pemesanan layanan *event organizer* berbasis *web*. Analisis sistem ini menggunakan *tools Business Process Model and Notation* (BPMN), analisis kebutuhan fungsional sistem, *use case diagram*, *use case scenario*, *sequence diagram*, dan *class analysis*. Sedangkan pada perancangan sistem terdiri dari perancangan *database*, struktur tabel dan basis data, arsitektur aplikasi, *class diagram*, dan *user interface*.

4.1. Analisis Sistem

Pada tahap analisis sistem dilakukan analisis terhadap sistem yang sedang berjalan serta merancang sistem yang diusulkan untuk memenuhi kebutuhan fungsional dari sistem yang dibangun. Pemodelan analisis sistem dilakukan dengan menggunakan UML. UML yang digunakan dalam analisis sistem ini adalah *use case diagram*, *use case scenario*, *sequence diagram*, dan *class analysis*.

4.1.1. Kedudukan Sistem

Proses bisnis yang sedang berjalan pada V-Pro Event Organizer dalam melakukan transaksi sampai saat ini masih dilakukan secara manual dan belum terkomputerisasi. Akibatnya aliran informasi sering berjalan lambat dan proses transaksi kurang berjalan optimal karena pelanggan harus bertemu karyawan secara langsung untuk mendapatkan informasi. Oleh sebab itu, pada penelitian ini dirancanglah suatu aplikasi pemesanan layanan *event organizer* berbasis *web* agar pelanggan bisa mendapatkan informasi secara cepat dan memesan layanan sesuai *budget*.

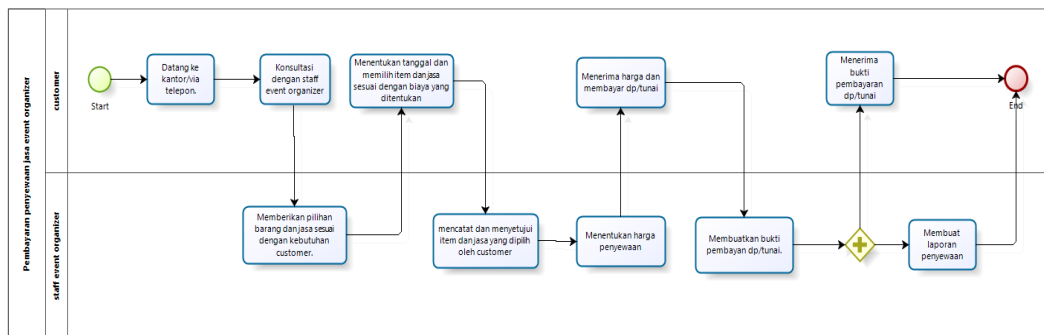
4.1.2. Sistem Berjalan

Pada bagian ini dijelaskan proses bisnis yang sedang berjalan pada CV. V-Pro. Proses pemesanan layanan *event organizer* pada CV. V-Pro yang sedang berjalan sebagai berikut:

1. Untuk melakukan *booking* dalam penyewaan *customer* datang langsung ke kantor/via telepon.
2. *Customer* konsultasi dengan staff *event organizer*.
3. *Staff event organizer* memberikan pilihan barang dan jasa sesuai dengan kebutuhan *customer*.

4. *Customer* menentukan tanggal dan memilih item dan jasa yang dibutuhkan.
5. *Staff event organizer* mencatat dan menyetujui item dan jasa yang dipilih oleh *customer*.
6. *Staff event organizer* menentukan harga penyewaan.
7. *Customer* membayar dp/tunai penyewaan.
8. *Staff event organizer* memberikan bukti pembayaran dp/tunai kepada *customer*.
9. *Customer* menerima bukti.
10. *Staff event organizer* membuat laporan penyewaan.

Untuk proses bisnis yang sedang berjalan penyewaan jasa *event organizer* pada CV. V-Pro Kota Padang ini digambarkan dengan BPMN seperti yang digambarkan pada Gambar 4.1



Gambar 4.1 BPMN Penyewaan Jasa *Event Organizer* yang Sedang Berjalan.

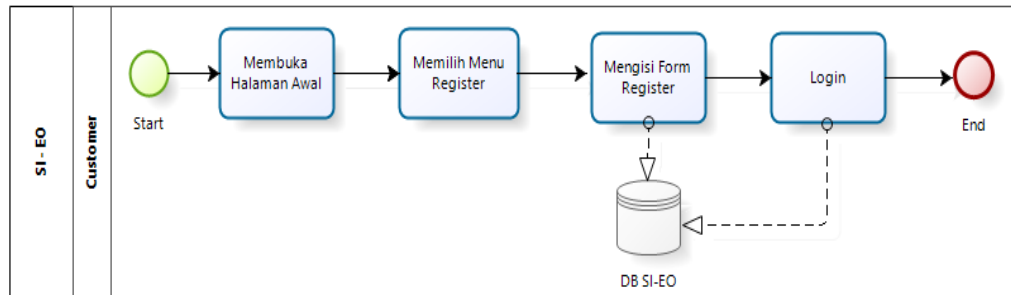
4.1.3. Sistem Diusulkan

Sistem penyewaan jasa *event organizer* yang diusulkan dalam analisis sistem ini, dimodelkan dengan menggunakan BPMN. Terdapat dua proses bisnis yang diusulkan yaitu proses registrasi dan cara penyewaan jasa *event organizer*. Berikut merupakan BPMN yang diusulkan pada pembangunan sistem *booking online* dalam penyewaan jasa *event organizer* berbasis *web* pada CV. V-Pro.

- a. Pada bagian ini dijelaskan proses bisnis yang diusulkan pada CV. V-Pro. Aktor yang terlibat untuk sistem register adalah customer. Proses register pada layanan event organizer pada CV. V-pro adalah sebagai berikut:
 1. *Customer* membuka halaman awal.
 2. *Customer* memilih menu register
 3. *Customer* mengisi form register.

4. *Customer* melakukan login.

Untuk proses bisnis yang diusulkan dari penyewaan jasa *event organizer* pada CV. V-Pro Kota Padang ini digambarkan dengan BPMN seperti yang terdapat pada Gambar 4.2

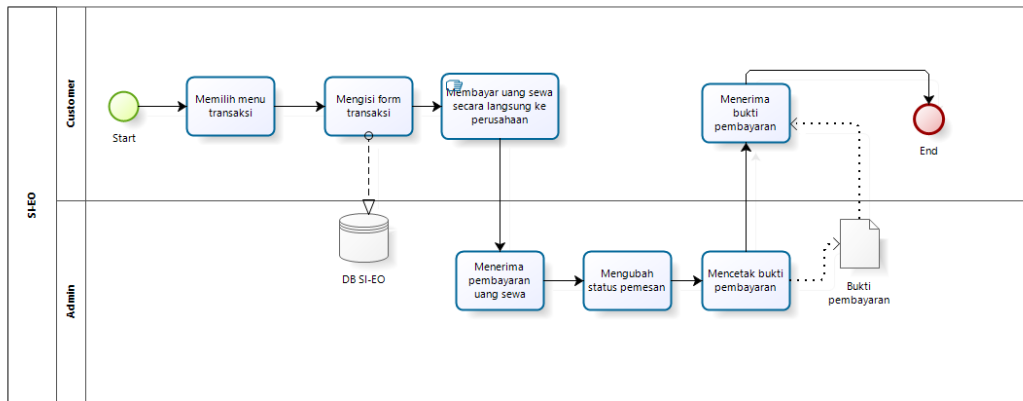


Gambar 4.2. BPMN Register yang diusulkan

b. Pada bagian penyewaan, *Staff event organizer* berperan sebagai admin. Proses pemesanan layanan *event organizer* pada CV. V-Pro yang diusulkan adalah sebagai berikut:

1. *Customer* memilih menu tambah transaksi
2. *Customer* mengisikan form transaksi
3. *Customer* membayar uang sewa secara langsung ke perusahaan
4. Admin menerima uang sewa
5. Admin mengubah status pemesanan
6. Admin mencetak bukti
7. *Customer* menerima bukti

Untuk proses bisnis yang diusulkan dari penyewaan jasa *event organizer* pada CV. V-Pro Kota Padang ini digambarkan dengan BPMN seperti yang terdapat pada Gambar 4.3



Gambar 4.3. BPMN Penyewaan Jasa *Event Organizer* yang diusulkan

4.1.4. Kebutuhan Fungsional Aplikasi

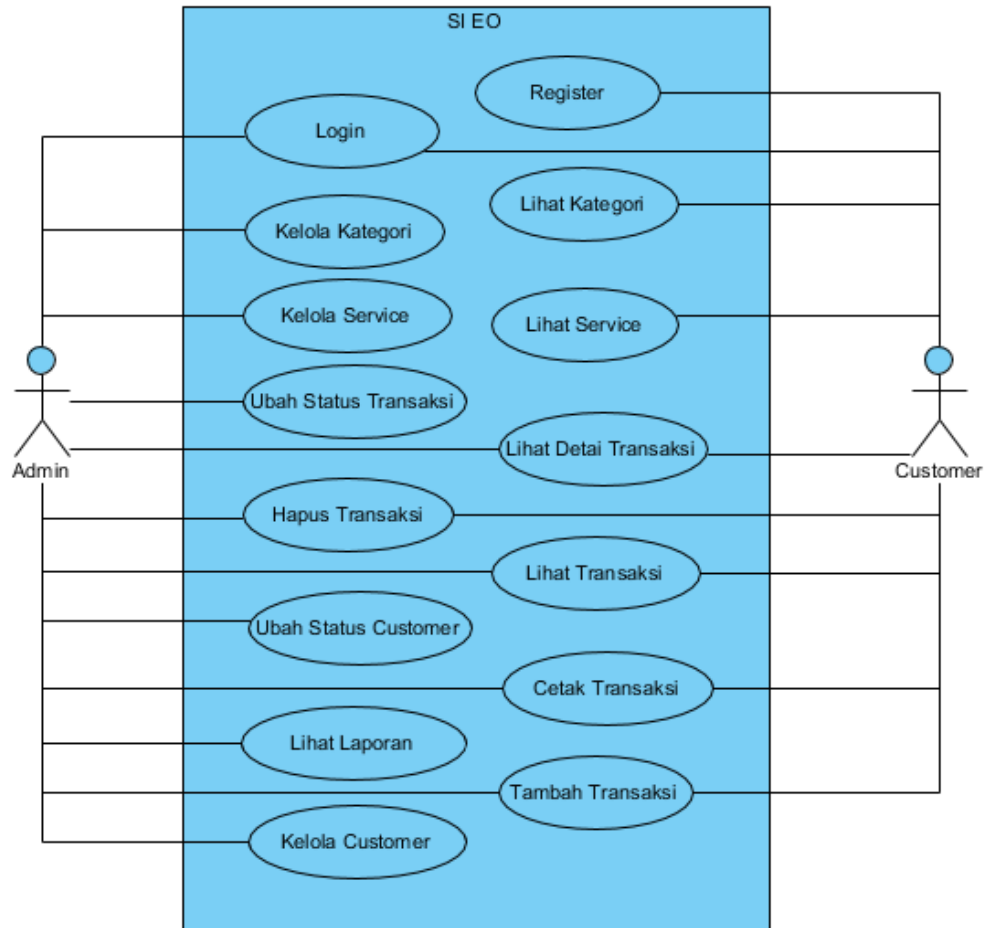
Kebutuhan fungsional dihasilkan dari proses pengumpulan data dan proses bisnis yang diusulkan. Berdasarkan analisis alur proses pada layanan *event organizer* yang diusulkan, maka dirumuskan kebutuhan fungsionalnya yaitu:

1. *Customer* dapat melakukan register.
2. Admin dan *customer* dapat melakukan login.
3. Admin mengelola kategori.
4. *Customer* dapat melihat kategori
5. Admin mengelola data service.
6. *Customer* dapat melihat data service.
7. Admin dan *customer* dapat menambah transaksi.
8. Admin dan *customer* dapat melihat data transaksi.
9. Admin dan *customer* dapat menghapus data transaksi
10. Admin dapat mengubah status transaksi.
11. Admin dan customer dapat melihat data detail transaksi.
12. Admin dan *customer* dapat mencetak laporan transaksi.
13. Admin dapat melihat laporan transaksi.
14. Admin dapat mengelola data *customer*.
15. Admin dapat mengubah status *customer*.

4.1.5. Use Case Diagram Aplikasi

Use case diagram menggambarkan hubungan antara fungsional dengan aktor yang terlibat di dalam sistem. Dengan menggunakan *Use Case Diagram*, kita

dapat mengetahui apa saja yang dapat dilakukan oleh aktor di dalam sistem yang dibuat. *Use case diagram* pembangunan aplikasi pemesanan layanan *event organizer* dapat dilihat pada gambar 4.4 berikut.



Gambar 4.4 *Use Case Diagram* Perancangan Aplikasi

Berdasarkan *use case diagram* perancangan aplikasi pada gambar 4.4, didapatkan dua aktor yang terlibat dalam aplikasi penyewaan layanan *event organizer*. Dua aktor yang terlibat dalam aplikasi ini adalah admin dan *customer*. Terdapat lima belas fungsional pada aplikasi memiliki hubungan sesuai dengan fungsi dari masing-masing aktor. Fungsional yang berjumlah lima belas itu antara lain login, register, kelola kategori, lihat kategori, kelola *service*, lihat *service*, tambah transaksi, lihat transaksi, ubah status transaksi, lihat detail transaksi, hapus transaksi, lihat laporan, cetak laporan, kelola *customer* dan ubah status *customer*. Aktor *customer* dapat melakukan register, login, lihat kategori, lihat *service*, tambah transaksi, lihat transaksi, hapus transaksi, lihat detail *service* dan cetak laporan. Aktor admin dapat melakukan login, kelola kategori, kelola *service*,

tambah transaksi, lihat transaksi, ubah status transaksi, lihat detail transaksi, hapus transaksi, lihat laporan, cetak laporan, kelola *customer* dan ubah status *customer*. Fungsional dipilih berdasarkan kebutuhan dan efektivitas pada sistem.

4.1.6. Deskripsi Tugas Aktor

Deskripsi tugas aktor merupakan penjelasan dari *Use Case Diagram*. Berdasarkan dengan *use case diagram* yang telah dibuat, terdapat dua aktor yang terlibat dalam aplikasi pemesanan layanan event organizer, yaitu admin dan *customer*. Masing-masing aktor memiliki tugasnya masing-masing. Deskripsi peran atau tugas dari masing-masing aktor dapat dilihat pada tabel 4.1.

Tabel 4.1 Deskripsi Tugas Aktor

No.	Aktor	Deskripsi Tugas
1	Admin	<ul style="list-style-type: none"> • Login admin dapat melakukan login. • kelola kategori admin dapat menambah, mengubah, melihat dan menghapus data kategori. • kelola <i>service</i>, admin dapat menambah, mengubah, melihat, dan menghapus data layanan. • tambah transaksi admin dapat menambahkan transaksi dan memilih <i>customer</i> yang bertransaksi. • lihat transaksi admin dapat melihat transaksi. • hapus transaksi Admin dapat menghapus transaksi • ubah status transaksi admin dapat mengubah status transaksi. • lihat detail transaksi admin dapat melihat detail transaksi • lihat laporan admin dapat melihat laporan transaksi yang dilakukan oleh <i>customer</i> • cetak laporan admin dapat mencetak laporan transaksi • kelola <i>customer</i> admin dapat menambah, mengubah, dan menghapus data <i>customer</i>.

Tabel 4. 1 Deskripsi Tugas Aktor (Lanjutan)

		<ul style="list-style-type: none"> • ubah status <i>customer</i> admin dapat mengubah status <i>customer</i>.
2.	<i>Customer</i>	<ul style="list-style-type: none"> • login <i>customer</i> dapat melakukan register. • registrasi <i>customer</i> dapat melakukan login. • lihat kategori, <i>customer</i> dapat melihat data kategori. • lihat <i>service</i>, <i>customer</i> dapat melihat semua layanan. • lihat transaksi <i>customer</i> dapat melihat transaksi yang telah dilakukannya. • lihat detail transaksi <i>customer</i> dapat melihat detail. • hapus transaksi Admin dapat menghapus transaksi • cetak laporan admin dapat mencetak laporan transaksi

4.1.7. Use Case Scenario

Use case scenario digunakan untuk menjelaskan langkah-langkah tentang bagaimana menggunakan fungsional yang ada pada sistem. *Use Case Scenario* yang dibahas pada bagian ini mencakup register, tambah transaksi, dan cetak status transaksi. Skenario *use case* lainnya dijelaskan pada bagian lampiran B.

4.1.7.1. Use Case Scenario Register

Skenario registrasi dapat dilakukan oleh *customer*. Kondisi awal dari skenario registrasi adalah aktor telah membuka *guest*. *Use Case Scenario* dari registrasi dapat dilihat pada Tabel 4.2.

Tabel 4.2 *Use Case Scenario Register*

<i>Use Case</i>	Register
<i>Actor</i>	Customer
<i>Entry Condition</i>	<i>Actor</i> telah membuka halaman awal
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> memilih menu “Sign Up” 2. Sistem menampilkan Halaman sign up. 3. <i>Actor</i> mengisi form register 4. <i>Actor</i> menekan tombol register 5. Sistem menyimpan data user 6. Sistem menampilkan halaman awal

<i>Scenario Alternatif</i>	7. <i>Actor</i> menekan tombol submit saat data kosong atau ada kesalahan. 8. Sistem menampilkan notifikasi error
<i>Exit Condition</i>	Sistem menampilkan notifikasi error

4.1.7.2. Use Case Scenario Tambah Transaksi

Use case scenario tambah transaksi dapat dilakukan oleh aktor admin dan *customer*. Pada bagian ini dijelaskan langkah - langkah yang dilakukan oleh admin dan *customer* untuk membuat transaksi baru. Skenario *use case* dari tambah transaksi yang dilakukan oleh admin dan *customer* dapat dilihat pada tabel 4.3.

Tabel 4.3 Use Case Scenario Tambah Transaksi

<i>Use Case</i>	Tambah Transaksi
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	Aktor telah login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> Memilih menu Transaksi 2. Sistem menampilkan halaman Transaksi 3. <i>Actor</i> menekan tombol <i>new transaction</i> 4. Sistem menampilkan halaman tambah transaksi 5. <i>Actor</i> mengisi data tanggal transaksi 6. <i>Actor</i> mengisi data budget yang dimiliki 7. <i>Actor</i> menekan tombol tambah 8. Sistem menampilkan data kategori 9. <i>Actor</i> memilih kategori service. 10. Sistem Menampilkan data service 11. <i>Actor</i> memilih layanan yang diinginkan 12. Sistem menampilkan harga layanan dan total harga layanan 13. <i>Actor</i> menekan tombol simpan 14. Sistem menyimpan data 15. Sistem menampilkan halaman lihat transaksi
<i>Scenario Alternatif</i>	<ol style="list-style-type: none"> 16. <i>Actor</i> menekan tombol simpan saat inputan data salah atau kosong 17. Sistem menampilkan notifikasi error
<i>Exit Condition</i>	Sistem menampilkan notifikasi error

4.1.7.3. Use Case Scenario Cetak Transaksi

Proses cetak transaksi hanya dapat dilakukan oleh admin. Pada proses ini admin dapat mencetak transaksi yang telah dilakukan oleh *customer*. Skenario *use case* dari cetak transaksi dapat dilihat pada Tabel 4.4.

Tabel 4.4 *Use Case Scenario* Cetak Transaksi

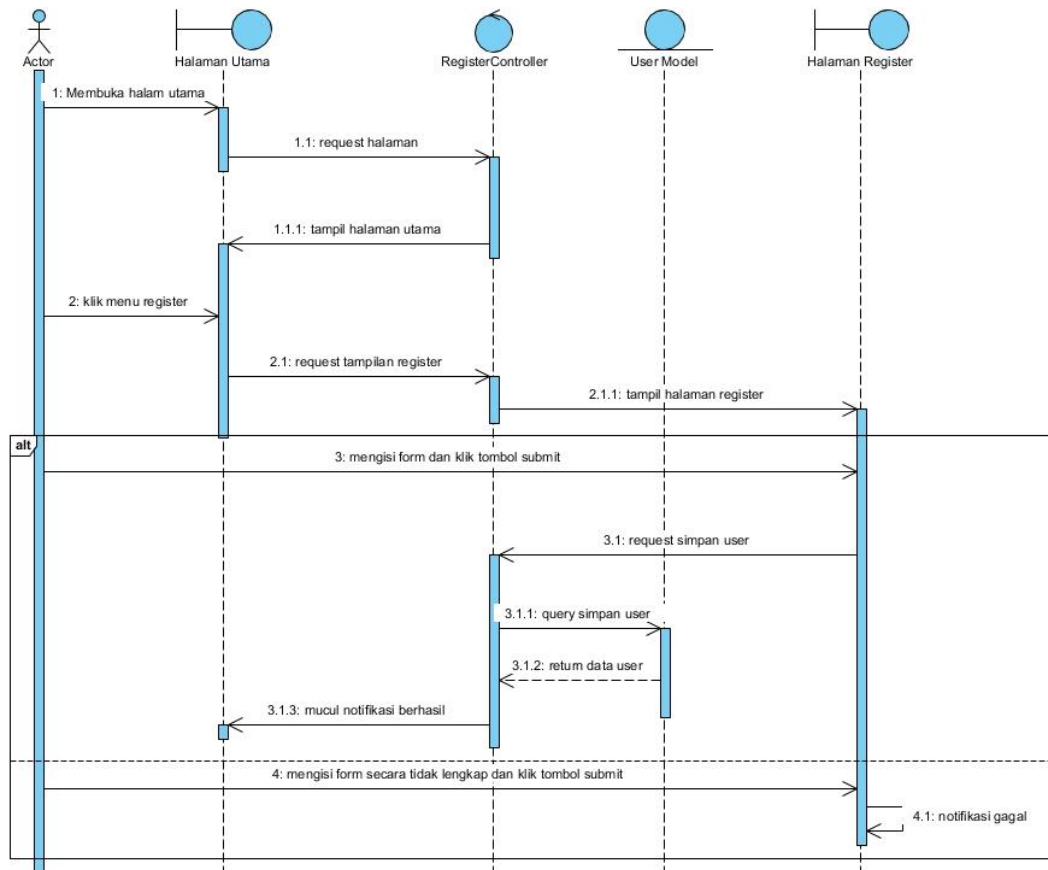
<i>Use Case</i>	Cetak Transaksi
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman awal 2. Sistem menampilkan halaman awal. 3. <i>Actor</i> memilih menu Transaksi 4. Sistem menampilkan halaman Transaksi 5. <i>Actor</i> menekan tombol cetak 6. Sistem mengarahkan halaman ke halaman cetak transaksi 7. Sistem Mencetak transaksi
<i>Exit Condition</i>	Sistem menampilkan halaman cetak Transaksi dan mencetak transaksi yang ditampilkan

4.1.8. Sequence Diagram

Bagian ini menjelaskan tentang *sequence diagram* dari setiap proses yang terjadi di dalam sistem. *Sequence diagram* berpedoman pada analisis *use case diagram*, dan kemudian dikembangkan sampai dengan proses terkecil yang ada pada setiap *use case*. Pada *sequence diagram* ini setiap *actor* di asumsikan telah melakukan *login* terlebih dahulu sebelum masuk ke proses berikutnya. *Sequence Diagram* yang ditampilkan pada bagian ini adalah *sequence diagram* registrasi, tambah transaksi dan cetak laporan. *Sequence diagram* untuk proses lainnya ditampilkan pada bagian lampiran C.

4.1.8.1. Sequence Diagram Register

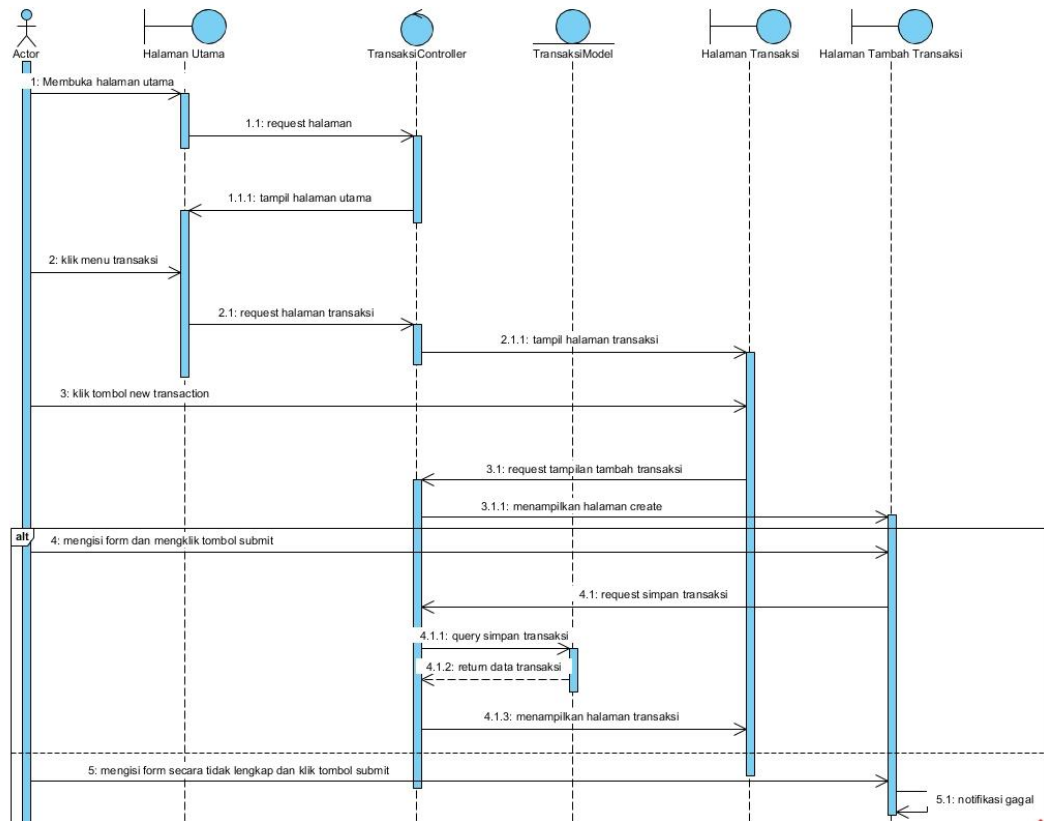
Proses register dapat dilakukan oleh *customer*. Proses ini dimulai dari saat aktor membuka halaman awal. Saat aktor memilih menu register pada halaman utama. Setelah itu system menampilkan halaman register. Aktor mengisi form register dan menekan tombol submit. *Request* dikirim ke RegisterController. RegisterController melakukan *request* data transaksi ke model User. Model transaksi memberikan *return* berupa data *query* register. Kemudian User menampilkan halaman home. Untuk lebih jelasnya dapat dilihat pada gambar 4.5.



Gambar 4.5 *Sequence Diagram Register*

4.1.8.2. *Sequence Diagram Tambah Transaksi*

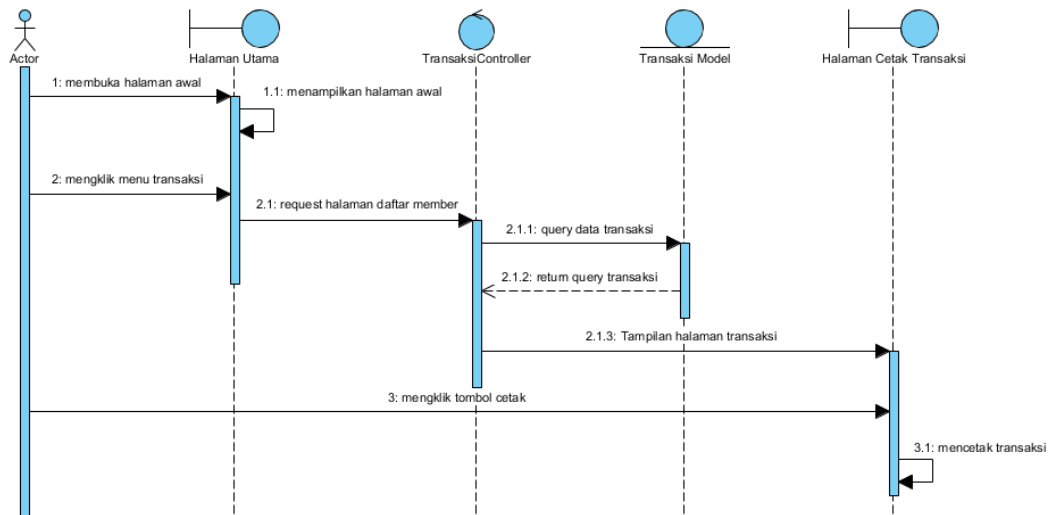
Proses tambah transaksi dilakukan oleh admin dan *customer*. Proses ini dimulai dari membuka halaman utama. Aktor menekan menu *transaction* untuk menampilkan halaman transaksi. Kemudian aktor menekan tombol *new transaction*. *TransaksiController* melakukan *request* untuk menampilkan halaman *create* transaksi. User mengisi *form* dan menekan tombol *save*. Kemudian *request* dikirim ke *TransaksiController* untuk menyimpan data transaksi. *Transaksi* melakukan *query* penyimpanan data ke model *Transaksi*. Model transaksi memberikan *return* berupa data *query* transaksi. Kemudian *TransaksiController* mengirimkan *request* untuk menampilkan halaman transaksi. Untuk lebih jelasnya dapat dilihat pada gambar 4.6.



Gambar 4.6 Sequence Diagram Tambah Transaksi

4.1.8.3. Sequence Diagram Cetak Transaksi

Proses cetak transaksi hanya dapat dilakukan oleh admin. Proses ini dimulai dari saat aktor membuka halaman awal menampilkan halaman utama. Aktor menekan menu *transaction* pada halaman awal. *Request* dikirim ke *view* untuk menampilkan halaman transaksi. Pada halaman transaksi terdapat tombol cetak. Aktor menekan tombol cetak tersebut. *Request* dikirimkan ke *TransaksiController* untuk menampilkan halaman cetak. Halaman Cetak kemudian mencetak transaksi. Untuk lebih jelasnya dapat dilihat pada gambar 4.7.



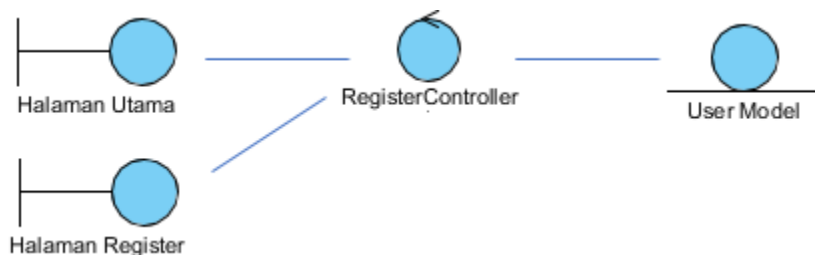
Gambar 4.7 Sequence Diagram Cetak Transaksi

4.1.9. Class Analysis

Class analysis berguna untuk menggambarkan kelas-kelas yang terlibat dalam setiap proses. *Class analysis* dibuat berdasarkan *sequence diagram* yang telah dibuat sebelumnya. Pada bagian ini ditampilkan *class analysis* pada sistem yang dibangun. *Class analysis* yang ditampilkan ada tiga, yaitu *class analysis* Registrasi, tambah transaksi dan cetak transaksi. *Class analysis* lainnya dicantumkan pada lampiran D.

4.1.9.1. Class Analysis Register

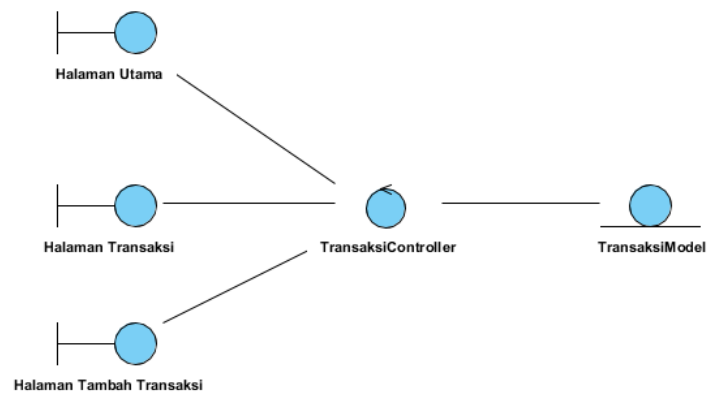
Class analysis register menjelaskan semua kelas yang terlibat dalam proses register. *Class analysis* register transaksi terdiri dari dua halaman, yaitu halaman utama dan halaman register. Pengelolaan data diproses oleh TransaksiController untuk kemudian diambil dari database menggunakan user model. Untuk lebih jelasnya dapat dilihat pada gambar 4.8.



Gambar 4.8 Class Analysis Register

4.1.9.2. *Class Analysis* Tambah Transaksi

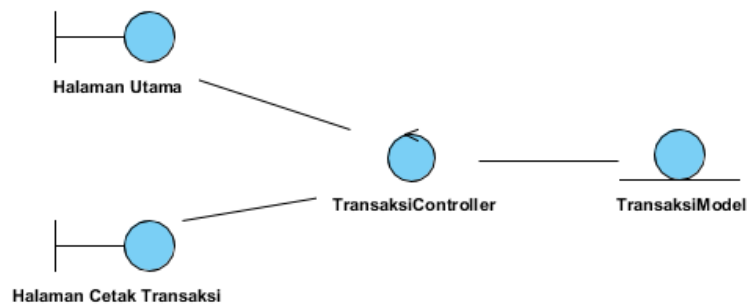
Class analysis tambah transaksi menjelaskan semua kelas yang terlibat dalam proses penambahan transaksi baru. *Class analysis* tambah transaksi terdiri dari tiga halaman, yaitu halaman utama, halaman transaksi dan halaman tambah transaksi. Pengelolaan data diproses oleh TransaksiController untuk kemudian dikirim dan disimpan ke database menggunakan model transaksi. Untuk lebih jelasnya dapat dilihat pada gambar 4.9.



Gambar 4.9 *Class Analysis* Tambah Transaksi

4.1.9.3. *Class Analysis* Cetak Transaksi

Class analysis cetak transaksi menjelaskan semua kelas yang terlibat dalam proses cetak transaksi. *Class analysis* cetak transaksi terdiri dari dua halaman, yaitu halaman utama dan halaman cetak transaksi. Proses cetak transaksi data diproses oleh TransaksiController. Model yang digunakan untuk penghapusan data dari database adalah model user. Untuk lebih jelasnya dapat dilihat pada gambar 4.10.



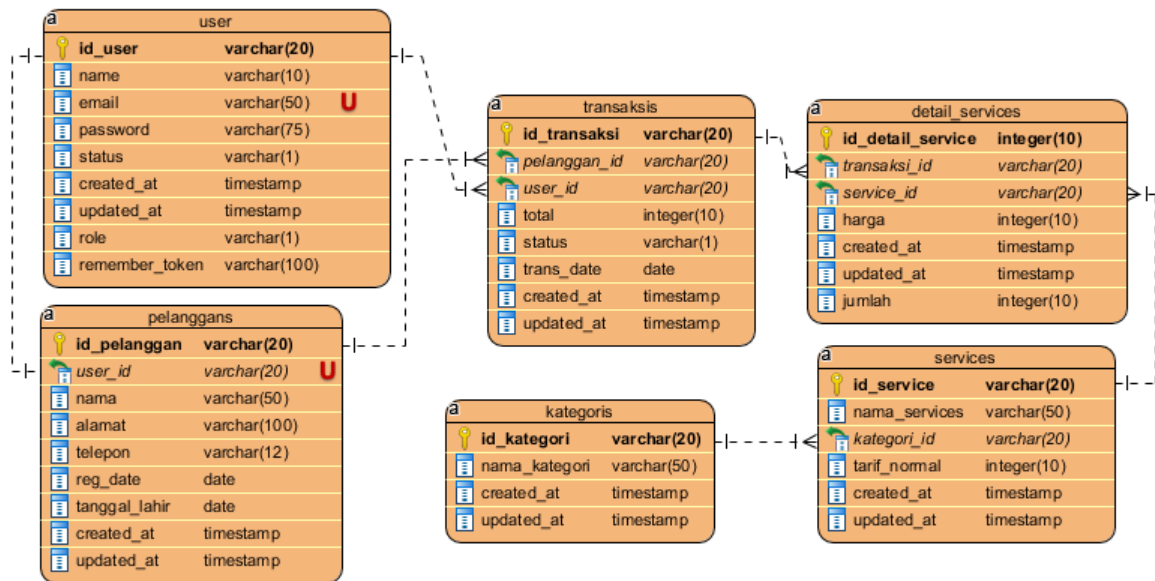
Gambar 4.10 *Class Analysis* Cetak Transaksi

4.2. Perancangan Sistem

Berdasarkan tahapan analisis proses bisnis yang sedang berjalan, alur sistem, dan kebutuhan fungsional sistem yang dibangun maka diperoleh hasil yang menjadi dasar dan tolak ukur untuk melakukan perancangan sistem. Perancangan sistem ini meliputi perancangan *database*, struktur tabel dan basis data, arsitektur aplikasi antarmuka (*user interface*), dan diagram kelas (*class diagram*).

4.2.1. Perancangan Basis Data

Perancangan basis data dimulai dengan menyusun struktur basis data berdasarkan entitas yang digunakan dan relasi dengan entitas lainnya. Entitas dan hubungan antar entitas digambarkan dengan ERD. Pada aplikasi ini, basis data yang dirancang memiliki enam tabel. Tabel-tabel tersebut terdiri dari empat tabel master dan dua tabel transaksi. Tabel master terdiri dari tabel tabel users, tabel pelanggan, tabel kategoris dan tabel services. Sedangkan tabel transaksi terdiri dari tabel transaksis dan tabel detail_services. Rancangan ERD sistem dapat dilihat pada gambar 4.11



Gambar 4.11 Rancangan ERD Sistem

4.2.2. Struktur Tabel dan Basis Data

Struktur tabel dan basis data adalah representasi tabel pada masing-masing relasi beserta status atributnya. Status tersebut berupa *primary key*, *foreign key*, tipe data, nama atribut, dan nama tabel. Berikut merupakan uraian dari tabel Transaksi, tabel Detail_Services, dan tabel Services yang dijelaskan pada Tabel 4.5, 4.6, dan 4.7. Untuk uraian tabel lainnya dapat dilihat pada bagian lampiran E.

Tabel 4.5 Tabel Transaksi

Nama atribut	Tipe data	Keterangan
Id_transaksi	varchar (20)	PK
pelanggan_id	varchar (20)	FK
user_id	varchar (20)	FK
Total	Integer (10)	
Status	Integer (1)	
trans_date	Date	
created_at	Timestamps	
updated_at	Timestamps	

Tabel 4.6 Tabel Detail_Services

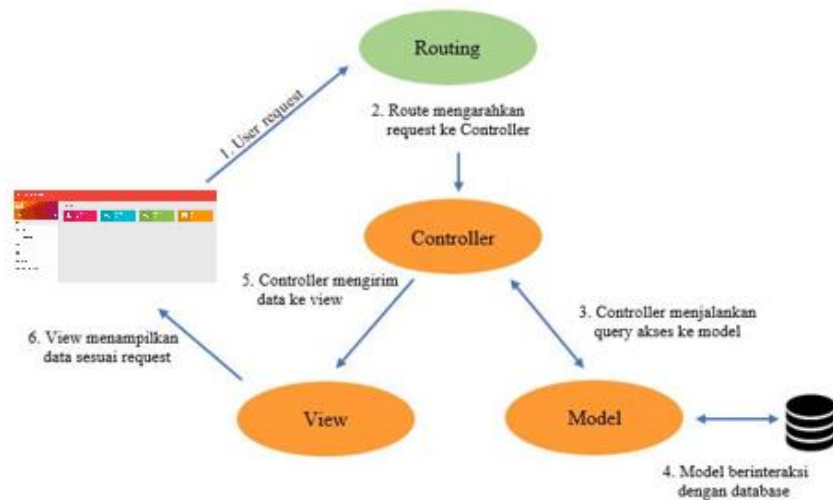
Nama atribut	Tipe data	Keterangan
Id_detail_service	Integer (10)	PK
transaksi_id	varchar (20)	FK
service_id	varchar (20)	FK
Harga	Integer (10)	
created_at	Timestamps	
updated_at	Timestamps	
Jumlah	Integer (10)	

Tabel 4.7 Tabel Services

Nama atribut	Tipe data	Keterangan
Id_services	varchar (20)	PK
nama_service	vanchar (50)	FK
tarif_normal	Integer (10)	FK
kategori_id	varchar (20)	
created_at	Timestamps	
updated_at	Timestamps	

4.2.3. Arsitektur Aplikasi

Perancangan dan pembangunan aplikasi pemesanan layanan *event organizer* berbasis *web* ini menggunakan arsitektur aplikasi MVC (*Model View Controller*) dengan metode Pemrograman Berbasis Objek (PBO). Arsitektur ini menggunakan arsitektur yang digunakan oleh *framework* Laravel Arsitektur pembangunan aplikasi ini dapat dilihat pada gambar 4.12.

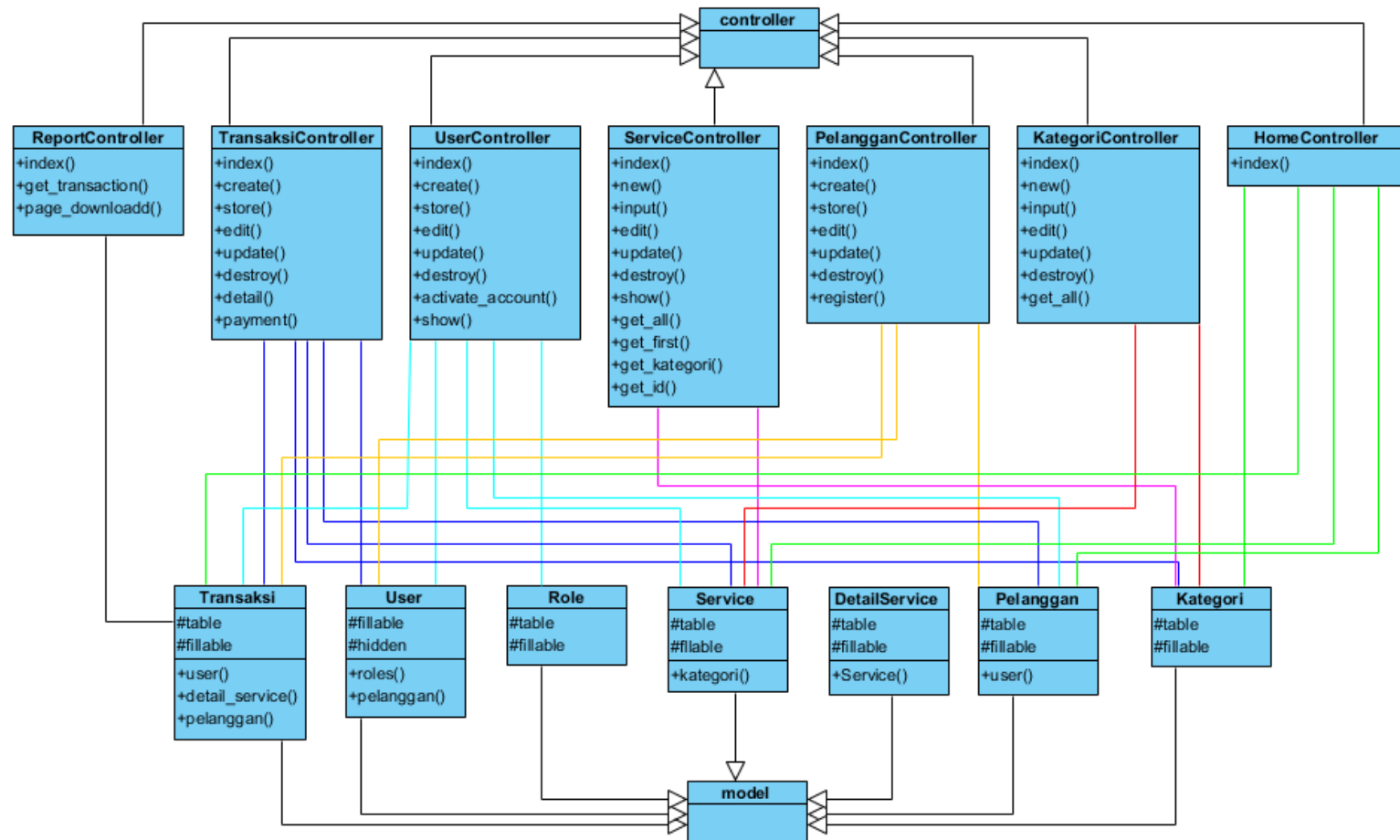


Gambar 4.12. Arsitektur Aplikasi

Berdasarkan gambar 4.12 mengenai arsitektur aplikasi terdapat empat elemen penting yang terlibat, yaitu *routing*, *controller*, *model*, dan *view*. Saat *user* melakukan *request* terhadap sistem, maka *routing* mengarahkan kepada *controller* dengan *method* yang dibutuhkan. *Controller* kemudian memanggil data melalui *model*. *Model* mengambil data dari *database*. Data yang didapatkan *model* dikirim kembali ke *controller* untuk kemudian ditampilkan oleh *view* ke halaman *user*.

4.2.4. Class Diagram

Class diagram adalah gambaran tentang struktur kelas-kelas dari suatu sistem. Dalam *class diagram* terdapat kelas *view*, *model*, dan *controller*. *Class diagram* juga menampilkan atribut dari masing-masing kelas beserta *method-method* yang terkandung didalamnya. *Method* yang ada pada masing-masing kelas memiliki tugas sesuai kebutuhan sistem. Dalam membangun sistem yang terintegrasi, setiap kelas memiliki relasi dengan kelas lainnya. *Class diagram* dari aplikasi ini dapat dilihat pada gambar 4.13.



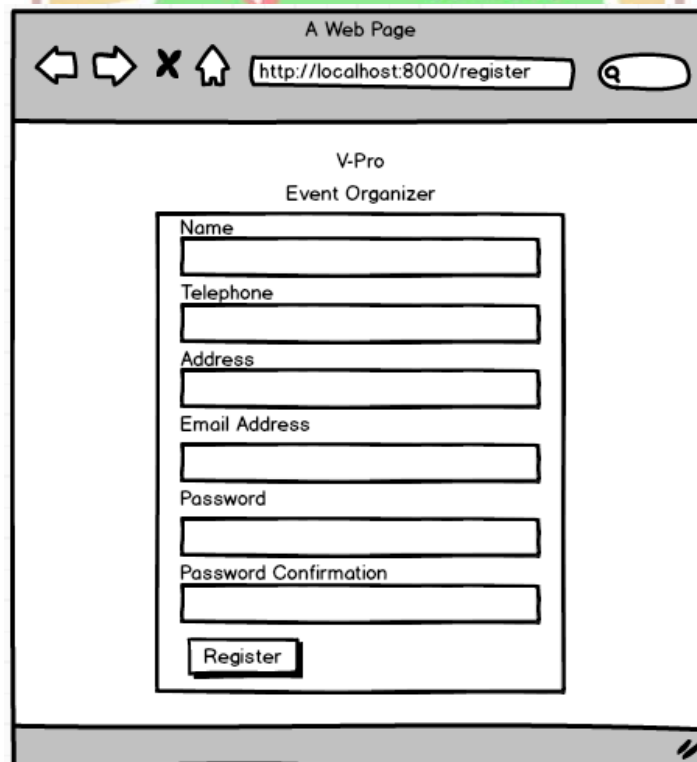
Gambar 4.13 Class Diagram Aplikasi

4.2.5. Perancangan Antarmuka

Antarmuka adalah mekanisme komunikasi antara pengguna (*user*) dengan sistem. Antarmuka pengguna (*user interface*) dapat menerima dan memberikan informasi kepada pengguna untuk membantu mengarahkan alur penelusuran masalah sampai ditemukan suatu solusi. User Interface yang dijelaskan pada bagian ini adalah registrasi, tambah transaksi dan cetak transaksi. *User interface* lainnya dijelaskan pada lampiran F.

4.2.5.1. Perancangan Anatarmuka Register

User Interface yang dijelaskan pada bagian ini adalah antarmuka halaman register. Halaman register merupakan halaman yang digunakan untuk *customer* melakukan pendaftaran awal sebelum membuat akun. Pada halaman register, terdapat enam input dan satu tombol register. Untuk gambaran perancangan antarmuka register tersebut dapat dilihat pada gambar 4.14.



The image shows a web browser window with the address bar displaying 'http://localhost:8000/register'. The page content includes the title 'V-Pro Event Organizer' and a registration form with the following fields: Name, Telephone, Address, Email Address, Password, and Password Confirmation. A 'Register' button is located at the bottom of the form.

Gambar 4.14. Perancangan antarmuka halaman Registrasi

4.2.5.2. Perancangan Anatarmuka Tambah Transaksi

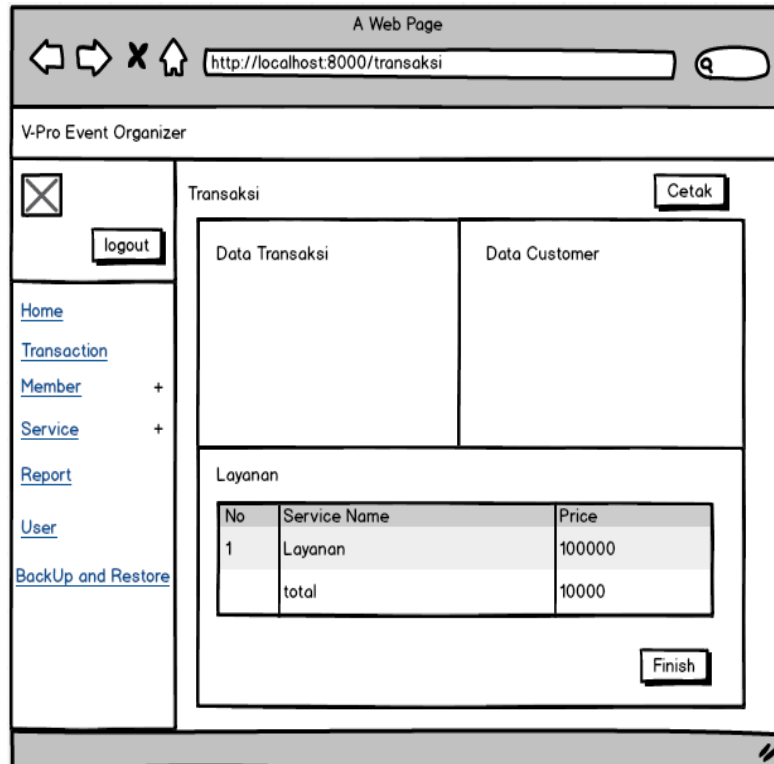
User Interface yang dijelaskan pada bagian ini adalah antarmuka halaman tambah transaksi. Halaman Tambah transaksi merupakan halaman yang digunakan untuk menambahkan data transaksi yang baru. Pada halaman tambah transaksi, terdapat *sidebar*, *form* data transaksi, dan *form* order detail. Untuk gambaran perancangan antarmuka aplikasi *web* tersebut dapat dilihat pada gambar 4.15.

The screenshot displays a web browser window titled "A Web Page" with the URL "http://localhost:8000/transaksi". The application is "V-Pro Event Organizer". On the left is a sidebar with a "logout" button and navigation links: Home, Transaction, Member, Service, Report, User, and BackUp and Restore. The main area is titled "Transaksi" and contains a "customer type" dropdown (set to "Customer"), a "transaction type" dropdown (set to "Transaksi"), a "customer name" text input, a "Date" field with a calendar icon, and an "Order Detail" section. The "Order Detail" section includes a "Set Budget" area with "budget" and "total belanja" inputs and a "+" button, and a "Pilih Service" area with "Kategori" and "Service" dropdowns, a "total belanja" input, and a "+" button. A "Save" button is at the bottom.

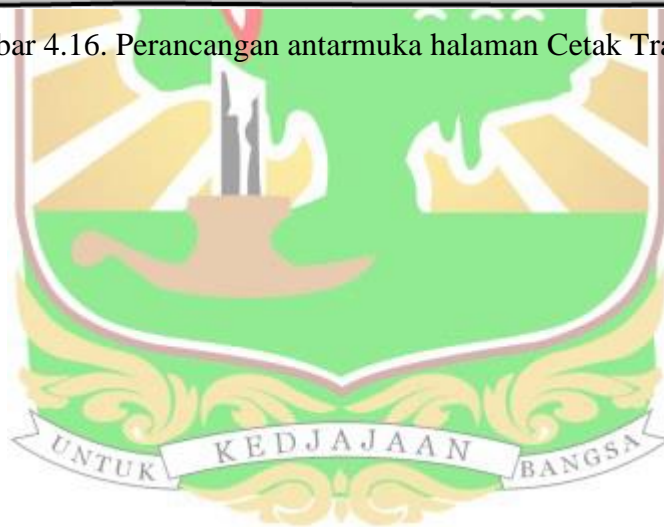
Gambar 4.15. Perancangan antarmuka halaman Tambah Transaksi

4.2.5.3. Perancangan Anatarmuka Cetak Transaksi

User Interface yang dijelaskan pada bagian ini adalah antarmuka cetak transaksi. Halaman cetak transaksi merupakan halaman yang digunakan untuk mencetak transaksi. Pada halaman cetak transaksi, terdapat *sidebar*, data transaksi, data *customer*, data detail order transaksi, tombol finisih dan tombo cetak untuk mencetak transaksi. Untuk gambaran perancangan antarmuka aplikasi *web* tersebut dapat dilihat pada gambar 4.16.



Gambar 4.16. Perancangan antarmuka halaman Cetak Transaksi



BAB V

IMPLEMENTASI DAN PENGUJIAN SISTEM

Bab ini membahas tentang implementasi kebutuhan pembangunan aplikasi dari hasil perancangan aplikasi yang dibahas pada bab sebelumnya. Setelah dilakukan tahap implementasi, maka dilanjutkan ke tahap pengujian sistem untuk meneliti kekurangan dan kesalahan pada sistem. Implementasi dan pengujian dilakukan pada aplikasi *web*.

5.1. Implementasi Sistem

Aplikasi Sistem Informasi Pelayanan *Event Organizer* ini difungsikan sebagai sistem yang mengelola dan menyajikan data dan informasi yang berkaitan dengan pelayanan dan pemesanan *event* pada CV. V-Pro *Event Organizer*. Pada sistem ini terdapat dua aktor yang terlibat dan lima belas fungsional yang dapat dijalankan. Fungsional tersebut mewakili setiap tugas dari *user* dan diharapkan dapat memberikan efisiensi dan efektifitas terhadap semua proses pada sistem. Pada aplikasi pembangunan sistem *Booking Online* dalam penyewaan Jasa EO Berbasis *Web* ini menggunakan sistem *Budget* dimana *customer* dapat mengisikan langsung jumlah *budget* yang diinginkan.

Implementasi aplikasi ini dilakukan dengan menggunakan perangkat keras komputer dengan spesifikasi sebagai berikut:

1. Komputer dengan *processor* Intel(R) Core(TM) i5-7200U CPU 2.50 GHz.
2. *Random Acces Memory* (RAM) 8 GB.
3. *Harddisk* Komputer dengan kapasitas 1 TB.

Spesifikasi perangkat lunak yang digunakan dalam implementasi sistem ini adalah sebagai berikut:

1. Sistem operasi Windows 10 Home
2. *Web browser* yang digunakan adalah google chrome version 66.0

Berikut ini spesifikasi *system requirement* untuk implementasi sistem:

1. Sistem operasi Windows 7, 8, 8.1, 10, Mac OS X 10.5 dan 10.6+, Linux, dan Ubuntu 8+.
2. *Web server* dengan modul PHP 7.0.13 atau setelahnya
3. *Web server* yang menggunakan PHPmyadmin.

4. *Web browser* yang mendukung HTML 5

Aplikasi *web* dibangun menggunakan bahasa pemrograman PHP (*Pearl Hypertext Preprocessor*) dan *framework* Laravel. *Framework* Laravel yang digunakan adalah Laravel versi 5.3, dan *web server apache* XAMPP versi 3.2.2. *Web server* berfungsi untuk menjalankan bahasa pemrograman PHP pada *browser* dan mengolah data pada *database*. *Database* yang digunakan sebagai tempat penyimpanan data adalah *database* MySQL.

5.1.1. Pengkodean Program

Berdasarkan arsitektur aplikasi MVC yang digunakan, pada bagian ini dijelaskan kode program yang dibuat. Bagian *view*, *model*, dan *controller* dijelaskan secara terpisah berdasarkan fungsional masing-masing. Berikut dijelaskan fungsi dan kegunaan masing-masing bagian pada aplikasi. Untuk penjelasan kode program lebih lanjut dapat dilihat pada lampiran G.

5.1.1.1. Kode Program Routing Aplikasi

Router berfungsi menangani request aplikasi kemudian mengarahkan aplikasi untuk memanggil halaman/resouce tertentu. *Routing* adalah bagian yang mengatur arah perpindahan permintaan (*request*) dari *user*. *Route* dapat *handle* semua perintah yang dideklarasikan dan mengirimkannya kepada *controller* serta *method* yang sesuai. Pada pembangunan aplikasi ini kode program *routing* menggunakan aplikasi *web*. Kode program *routing* aplikasi dapat dilihat pada Gambar 5.1.

```
Route::get('/login', function () {
    return view('auth.login');
});

Route::get('/', function () {
    return view('auth.index');
});

Auth::routes();

Route::get('/member-form', function () {
    return view('auth.member_form');
});

Route::post('/register-member', 'PelangganController@register');
```

Gambar 5.1. Kode Program Routing Aplikasi

```

Route::group(['middleware'=>'auth'], function()
{
    Route::get('/home', 'HomeController@index')->name('home');
    Route::resource('/pelanggan', 'PelangganController');
    Route::post('/pelanggan/update', 'PelangganController@update');
    Route::get('/service', 'ServiceController@index');
    Route::get('/service/destroy/{id}', 'ServiceController@destroy');
    Route::get('/service/show', 'ServiceController@show');
    Route::post('/service/update', 'ServiceController@update');
    Route::get('/service/new', 'ServiceController@new');
    Route::post('/service/input', 'ServiceController@input');

    Route::get('/kategori', 'KategoriController@index');
    Route::get('/kategori/destroy/{id}', 'KategoriController@destroy');
    Route::get('/kategori/show', 'KategoriController@show');
    Route::post('/kategori/update', 'KategoriController@update');
    Route::get('/kategori/new', 'KategoriController@new');
    Route::post('/kategori/input', 'KategoriController@input');

    Route::resource('/transaksi', 'TransaksiController');
    Route::get('/transaksi/detail/{id}', 'TransaksiController@detail');
    Route::get('/transaksi/destroy/{id}',
    'TransaksiController@destroy');
    Route::get('/transaksi/edit/{id}', 'TransaksiController@edit');
    Route::get('/transaksi/delete/service/{pesanan}/{service}',
    'TransaksiController@delete_service');
    Route::post('/transaksi/update', 'TransaksiController@update');

    Route::resource('/user', 'UserController');
    Route::get('/user/activated/{id}/{status}',
    'UserController@activated_account');
    Route::get('/user/destroy/{id}', 'UserController@destroy');
    Route::get('/user/show/{id}', 'UserController@show');

    Route::get('/report', 'ReportController@index');
});

Route::get('/service/get_all', 'ServiceController@get_all');
Route::get('/service/{kategori_id}/get_kategori',
'ServiceController@get_kategori');
Route::get('/service/get_first', 'ServiceController@get_first');
Route::get('/service/{id}/search', 'ServiceController@get_id');

Route::get('/report/download', 'ReportController@page_download');

```

Gambar 5.1. Kode Program Routing Aplikasi (Lanjutan)

5.1.1.2. Kode Program Controller

Controller merupakan bagian yang mengatur seluruh alur dan *method* pada aplikasi sistem informasi dengan menggunakan arsitektur MVC. Salah satu *controller* yang dibangun adalah *TransaksiController*. *TransaksiController* berisi

kode program untuk mengolah data Transaksi. Kode program *TransaksiController* dapat dilihat pada Gambar 5.2.

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Transaksi;
use App\User;
use App\Pelanggan;
use App\Service;
use App\Kategori;
use App\DetailService;
use Illuminate\Support\Facades\Validator;
use Illuminate\Support\Facades\Auth;

class TransaksiController extends Controller
{
    public $successStatus = 200;

    public function index()
    {
        $transaksis = Transaksi::with('detail_service',
'pelanggan','user')->orderBy('transaksis.trans_date', 'desc')->get();
        return view('pages.transaksi.index',compact('transaksis'));
    }

    public function create()
    {
        $data['pelanggan'] = Pelanggan::with('user')->where('user_id',
'!=', 1)->get();
        $data['kategori'] = Kategori::all();

        return view('pages.transaksi.create', $data);
    }

    public function store(Request $request)
    {
        $tanggal = date('Y-m-d');
        $arrTanggal= explode('-', $tanggal);
        $kode = 'F-'. $arrTanggal[0]. $arrTanggal[1]. $arrTanggal[2];
        $cari_kode = Transaksi::where('id_transaksi','like',$kode.'%')-
>orderBy('id_transaksi','desc')->first();

        if(empty($cari_kode)){
            $new_no = 1;
        } else {
            $arrNo = explode($kode, $cari_kode->id_transaksi);
            $new_no = $arrNo[1] + 1;
        }
        $new_kode = $kode.$new_no;
        $arrInput = array();
        $arrInput['id_transaksi'] = $new_kode;
        $arrInput['pelanggan_id'] = $request->pelanggan_id;
        if(Auth::user()->role == 1){
```

```
$arrInput['user_id'] = Auth::user()->id;
```

Gambar 5.2. Kode Program *TransaksiController* Aplikasi

```
    } else {  
        $arrInput['user_id'] = null;  
    }  
    $arrInput['total'] = $request->biaya;  
    $arrInput['status'] = 0;  
    $arrInput['trans_date'] = $request->trans_date;  
    $arrInput['diskon'] = 0;  
  
    $transaksi = Transaksi::insert($arrInput);  
    $service_id = $request->service_id;  
    $harga_pesanan = $request->harga_pesanan;  
    $arrInsert = array();  
    for($i = 0; $i < count($service_id); $i++ ){  
        $arr = array();  
        $arr['service_id'] = $service_id[$i];  
        $arr['harga'] = $harga_pesanan[$i];  
        $arr['transaksi_id'] = $new_kode;  
        array_push($arrInsert, $arr);  
    }  
  
    $insertService = DetailService::insert($arrInsert);  
    return redirect('transaksi');  
}  
  
public function destroy ($id)  
{  
    DetailService::where('transaksi_id', $id)->delete();  
    Transaksi::where('id_transaksi', $id)->delete();  
  
    return redirect(route('transaksi.index'));  
}  
  
public function edit ($id)  
{  
    $data['transaksi'] = Transaksi::with('user', 'pelanggan',  
'pelanggan.user')->where('id_transaksi', $id)->first();  
    $data['save'] = '';  
    return view('pages.transaksi.edit', $data);  
}  
  
public function update(Request $request)  
{  
    $user = Auth::user();  
    $service = $request->input('service');  
    $package = $request->input('package');  
    $coustumer = $request->input('pelanggan_id');  
    $transaksi_id = $request->input('transaksi_id');  
    $trans_date = $request->input('trans_date');  
    $fee = 0;
```

```

    $transaksi = Transaksi::where('id_transaksi', $transaksi_id)-
    >first();

```

Gambar 5.2. Kode Program *TransaksiController* Aplikasi (Lanjutan)

```

    $fee += $transaksi->total;

    Transaksi::where('id_transaksi', $transaksi_id)->update([
    'pelanggan_id' => $transaksi->pelanggan_id, 'user_id' => $user-
    >id, 'total' => $fee, 'status' => $request->input('status'), 'trans_date'
    => $trans_date, 'updated_at' => now()]);

    return redirect(route('transaksi.index'));
}

public function detail($id)
{
    $data['transaksi'] = Transaksi::where('id_transaksi', $id)-
    >first();

    return view('pages.transaksi.detail', $data);
}

public function payment (Request $request)
{
    $transaksi_id = $request->input('transaksi_id');

    if ($transaksi_id != null) {
        Transaksi::where('id_transaksi', $transaksi_id)-
        >update(['status' => 2]);
        $data['message'] = 'success';
        return response()->json($data, $this->successStatus);
    }
    else {
        $data['message'] = 'error';
        return response()->json($data, 400);
    }
}
}

```

Gambar 5.2. Kode Program *TransaksiController* Aplikasi (Lanjutan)

5.1.1.3. Kode Program *Model*

Model adalah bagian yang berguna memanggil data dari *database* serta mengirimkan data ke *controller* dengan menggunakan sintax SQL (*Structured Query Language*). Salah satu *model* yang dibangun adalah *model* transaksi untuk memanggil entitas transaksi pada *database*. Kode program *model* transaksi dapat dilihat pada Gambar 5.3.

```

namespace App;
use Illuminate\Database\Eloquent\Model;

class Transaksi extends Model
{
    protected $table = 'transaksis';
    protected $primaryKey = 'id_transaksi';
    public $incrementing = false;

    protected $fillable = ['id_transaksi', 'pelanggan_id', 'user_id',
'diskon', 'service', 'budget', 'total'];

    public function user()
    {
        return $this->belongsTo('App\User', 'id_user');
    }

    public function detail_service()
    {
        return $this->hasMany('App\DetailService', 'transaksi_id');
    }

    public function pelanggan()
    {
        return $this->belongsTo('App\Pelanggan', 'pelanggan_id');
    }
}

```

Gambar 5.3. Kode Program *Model* Transaksi

Berdasarkan Gambar 5.3 terdapat *class* transaksi yang mewakili entitas transaksi. Didalamnya terdapat fungsi yang menggambarkan relasi antara *class* transaksi dengan *class* yang lain.

5.1.1.4. Kode Program *View*

View merupakan antar muka aplikasi yang berinteraksi langsung dengan *user*. Data dari *database* dan perintah dari *controller* kemudian ditampilkan pada *view*. Salah satu *view* yang dibangun adalah *view* Transaksi Index yang berfungsi untuk menampilkan halaman “Daftar Transaksi”. *View* Transaksi Index ini berisi *script* untuk memanggil daftar transaksi yang ditampilkan dalam bentuk tabel. Kode program *list member* dapat dilihat pada Gambar 5.4.


```

@extends('layouts.app')
@section('content')
<div class="container-fluid">
  <div class="block-header">
    <h2>Transaction Data</h2>
  </div>
  <div class="row clearfix">
    <div class="col-lg-12 col-md-12 col-sm-12 col-xs-12">
      <div class="card">
        <div class="header">
          <ul class="header-dropdown m-r--5"></ul>
        </div>
        <div class="body">
          <div class="text-right">
            <a href="{!! route('transaksi.create') !!}" class="btn btn-
primary">
              <i class="material-icons">add</i>
              <span>New Transaction</span></a>
          </div><br>
          <div class="table-responsive">
            <table class="table table-bordered table-striped table-
hover js-basic-example dataTable">
              <thead>
                <tr>
                  <th class="text-center" width="1%">No</th>
                  <th class="text-center">ID Transaksi</th>
                  <th class="text-center">Customer</th>
                  <th class="text-center">Tanggal</th>
                  <th class="text-center">Total</th>
                  <th class="text-center">Status</th>
                  <th class="text-center" width="20%">Aksi</th>
                </tr>
              </thead>
              <tbody>
                <?php $no = 1; ?>
                @foreach($transaksis as $transaksi)
                  @if($user->role == 3 && isset($user->pelanggan-
>id_pelanggan) && $user->pelanggan->id_pelanggan == $transaksi-
>pelanggan_id)
                    <tr>
                      <td class="text-center" width="1%">{{ $no }}</td>
                      <td class="text-center">
                        {!! $transaksi->id_transaksi !!}
                      </td>
                      <td class="text-center">
                        {!! $transaksi->pelanggan->nama !!}
                      </td>
                      <td class="text-center">
                        {!! $transaksi->trans_date !!}
                      </td>
                      <td class="text-center">
                        {{number_format($transaksi->total)}}
                      </td>
                      <td class="text-center">
                        @if($transaksi->status == 0) Booking
                        @elseif($transaksi->status == 1) Pending
                        @else Paid
                        @endif
                      </td>
                    </tr>
                  @endif
                @endforeach
              </tbody>
            </table>
          </div>
        </div>
      </div>
    </div>
  </div>
</section>

```

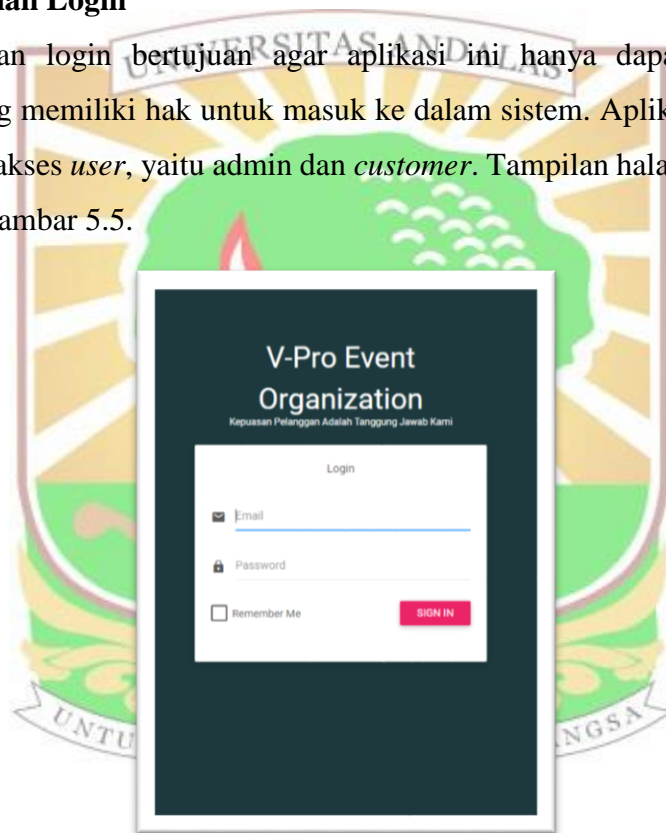
Gambar 5.4. Kode Program View List Member

5.1.2. Implementasi Antar Muka Aplikasi

Pada bagian ini dijelaskan tampilan aplikasi web dalam pembangunan sistem informasi aplikasi pemesanan layanan *event organizer* pada CV. V-Pro Event Organizer. Aplikasi *web* ini dapat diakses oleh admin, Halaman awal yang ditampilkan saat mengakses aplikasi *web* ini adalah halaman *guest*. Jika *user* berhasil *login*, maka akan ditampilkan halaman utama. Berikut dijelaskan tampilan aplikasi *web* dari sistem yang dibangun sesuai dengan implementasi sistem. Untuk penjelasan antar muka lain dapat dilihat pada lampiran H.

5.1.2.1. Halaman Login

Halaman login bertujuan agar aplikasi ini hanya dapat diakses oleh pengguna yang memiliki hak untuk masuk ke dalam sistem. Aplikasi ini memiliki dua level hak akses *user*, yaitu *admin* dan *customer*. Tampilan halaman *login* dapat dilihat pada Gambar 5.5.

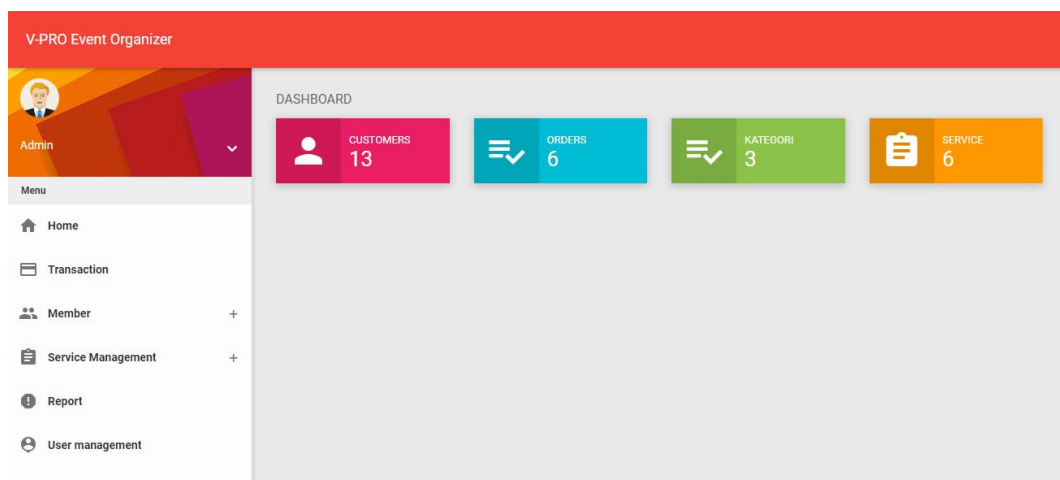


Gambar 5.5 Tampilan Halaman *Login*

Berdasarkan gambar 5.5 *user* diharuskan untuk *login* agar dapat masuk ke dalam sistem dengan men-input-kan *username* dan *password* yang telah didaftarkan pada *database* lalu menekan tombol “*Sign In*”. Jika *user* berhasil *login* maka sistem menampilkan halaman utama aplikasi.

5.1.2.2. Halaman Admin

Halaman admin dapat diakses oleh *user* dengan level hak akses admin. Admin memiliki wewenang dalam mengelola data Transaksi, data *service*, data kategori, manajemen user, data pelanggan, dan melihat laporan transaksi. Pada halaman ini *user* juga dapat melihat informasi-informasi tentang *event organizer*, seperti jumlah customer, jumlah order, jumlah *service* berjalan dan jumlah kategori. Tampilan halaman admin dapat dilihat pada Gambar 5.6.



Gambar 5.6 Tampilan Halaman Utama Pemilik

Berdasarkan Gambar 5.6 dapat dijelaskan bahwa halaman pemilik CV. V-Pro memiliki 7 menu yaitu enam menu utama dan satu menu *sign out* yang dapat diuraikan sebagai berikut:

1. Menu *Home*

Menu beranda adalah halaman utama yang menampilkan informasi dan pemberitahuan tentang *event organizer*. Informasi tersebut berupa jumlah customer, jumlah order, jumlah *service* berjalan dan jumlah kategori.

2. Menu *Transaction*

Menu *Transaction* menampilkan halaman pengelolaan data *Transaction*. Pada menu ini *user* dapat melakukan beberapa fungsional seperti tambah transaksi, edit transaksi, lihat data transaksi, hapus data transaksi, dan cetak data transaksi.

3. Menu *Member*

Menu *Member* menampilkan halaman pengelolaan data *customer*. Pada menu ini *user* dapat melakukan beberapa fungsional seperti registrasi *customer*, perbarui data *customer*, lihat data *customer*, dan hapus data *customer*.

4. Menu *Service Management*

Menu data *Service Management* terdiri dari dua submenu, yaitu menu kategori dan *service*. Masing-masing submenu menampilkan data terkait yang dapat dikelola oleh *user*.

5. Menu *Report*

Menu *Report* merupakan menu yang berfungsi untuk melihat laporan transaksi. Menu *report* menampilkan data laporan terkait dan dapat di-*convert* kedalam Microsoft Excel, CSV, dan PDF serta dapat melakukan pencetakan.

6. Menu *User Manajemen*

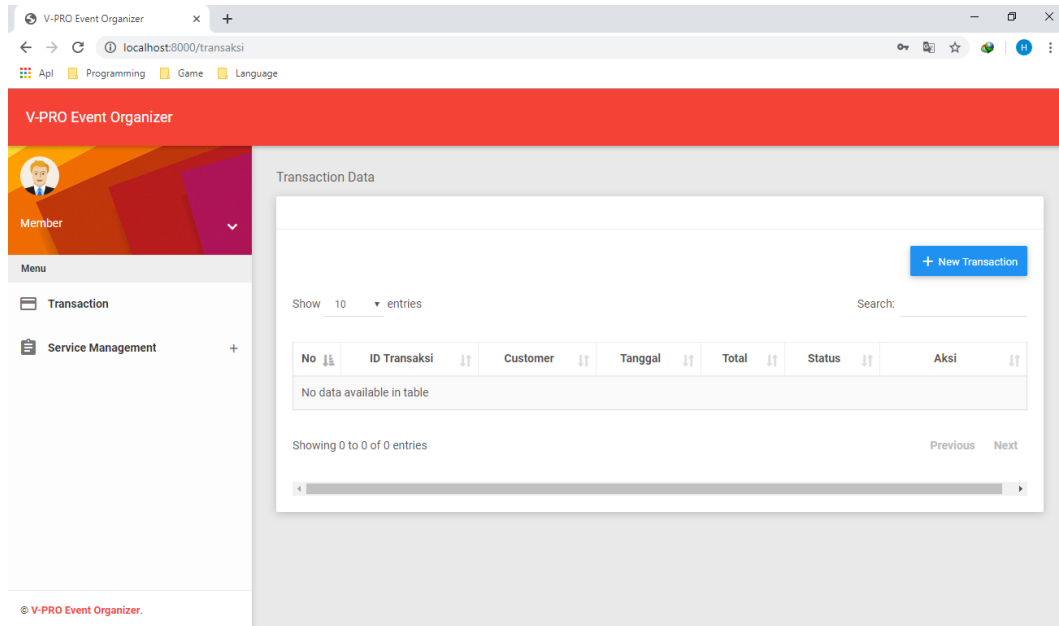
Menu manajemen *user* menampilkan halaman informasi seluruh data *user* yang memiliki hak akses kedalam aplikasi. *User* dapat mengelola setiap data *user* seperti menambah, meng*update* atau menghapus data *user*.

7. Menu *Sign Out*

Menu *logout* berfungsi untuk menutup halaman utama dan menampilkan halaman *guest*. Dengan mengakses menu *logout*, maka *session* *user* yang sedang aktif dihapus. Menu ini dapat diakses *user* pada bagian *navbar* di aplikasi.

5.1.2.3. Halaman *Customer*

Halaman *customer* dapat diakses oleh *user* dengan level hak akses *customer*. *Customer* masuk ke dalam sistem dan menggunakan fungsional yang tersedia. Halaman pertama yang diakses oleh *customer* setelah login adalah halaman *Transaksi*. Tampilan halaman dapat dilihat pada Gambar 5.7.



Gambar 5.7 Tampilan Halaman *Customer*

Berdasarkan Gambar 5.7 dapat dijelaskan bahwa halaman *customer* memiliki dua menu utama dan satu menu *sign out* yang dapat diuraikan sebagai berikut:

1. Menu *Transaction*

Menu *Transaction* menampilkan halaman lihat *Transaction*. Pada menu ini juga terdapat tombol transaksi yang digunakan oleh user untuk dapat menambah transaksi baru.

2. Menu *Service Management*

Menu data *Service Management* terdiri dari dua submenu, yaitu menu kategori dan *service*. Masing-masing submenu menampilkan data terkait yang dapat dilihat oleh *user*.

3. Menu *Sign Out*

Menu *logout* berfungsi untuk menutup halaman utama dan menampilkan halaman *guest*. Dengan mengakses menu *logout*, maka *session* user yang sedang aktif dihapus. Menu ini dapat diakses *user* pada bagian *navbar* di aplikasi.

5.2. Pengujian Sistem

Tahap pengujian sistem merupakan proses untuk memeriksa apakah aplikasi yang dihasilkan telah berjalan sesuai dengan perancangan sistem yang telah ditetapkan. Pengujian ini dilakukan dengan menggunakan metode *black box testing*, yaitu metode pengujian yang berfokus pada pemeriksaan ketersediaan fungsional yang sudah dirancang pada aplikasi. Kelebihan metode ini dibandingkan metode *white box testing* adalah pada metode *black box testing* tidak diperlukan pengetahuan khusus tentang kode program aplikasi, sedangkan pada metode *white box testing* pengujian diharuskan untuk memeriksa komponen internal kode program aplikasi. Pengujian ini dilakukan oleh penulis, pemilik dan pegawai serta salah satu pelanggan dari CV. V-pro Padang.

5.2.1. Fokus Pengujian

Fokus pengujian aplikasi ini menggunakan data uji berdasarkan data yang telah didapat dari aplikasi *web*. Pada pengujian ini terdapat lima belas item uji aplikasi web. Fokus pengujian dapat dilihat pada tabel 5.1 berikut.

Tabel 5.1. Fokus Pengujian

No.	Item Uji	Jenis Pengujian	Keterangan
1	Melakukan registrasi	Tambah	Sesuai
2	Melakukan login	Tambah	Sesuai
3	Mengelola data kategori	Lihat, tambah, edit, hapus	Sesuai
4	Melihat data kategori	Lihat	Sesuai
5	Mengelola data <i>service</i>	Lihat, tambah, edit, hapus	Sesuai
6	Melihat data <i>service</i>	Lihat	Sesuai
7	Tambah transaksi	Tambah	Sesuai
8	Melihat transaksi	Lihat	Sesuai
9	Hapus Transaksi	Hapus	Sesuai
10	Melihat Detail Transaksi	Lihat	Sesuai
11	Ubah Status Transaksi	Edit	Sesuai
12	Lihat Laporan	Lihat	Sesuai
13	Cetak Laporan	Cetak	Sesuai
14	Kelola <i>Customer</i>	Lihat, tambah, edit, hapus	Sesuai
15	Ubah Status <i>Customer</i>	Edit	Sesuai

5.2.2. Kasus Hasil Pengujian

Pada bagian ini ditampilkan hasil dari pengujian sistem. Pengujian yang dilakukan berpedoman pada fokus pengujian yang telah ditentukan sebelumnya. Pengujian dilakukan berdasarkan fitur-fitur pada sistem yang dibangun dengan memperhatikan masukan ke sistem dan keluaran yang dihasilkan sistem. Di bagian ini dijelaskan tiga kasus pengujian, yaitu pengujian registrasi, tambah transaksi dan cetak laporan.

5.2.2.1. Pengujian Registrasi Melalui Aplikasi Web

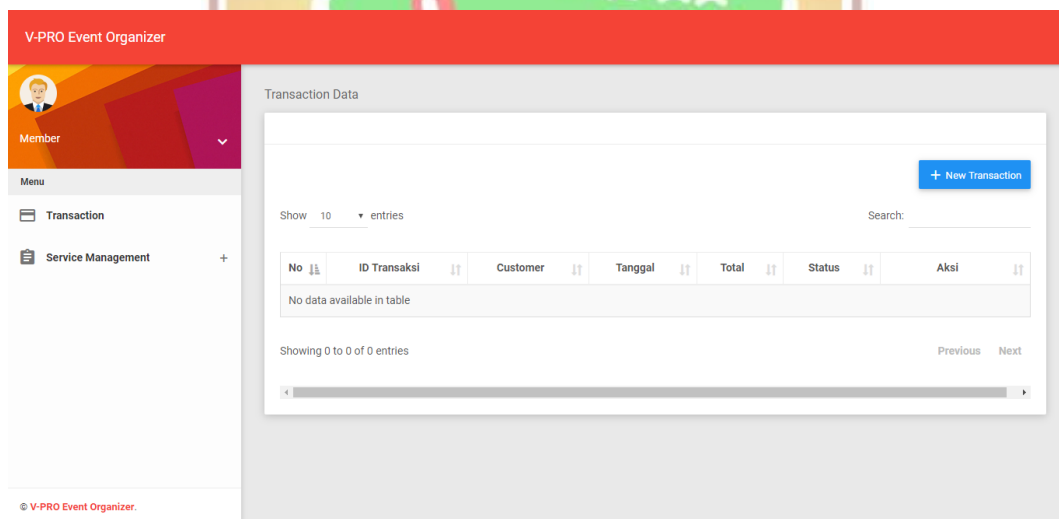
Pada pengujian ini dilakukan *input* data pelanggan pada aplikasi *web*. Hasil pengujian registrasi dapat dilihat pada Tabel 5.2 dan Tabel 5.3.

Tabel 5. 2 Pengujian Registrasi (Benar)

Kasus dan Hasil Uji (Benar)	
Data masukan	Semua data yang dibutuhkan dalam form registrasi
Yang diharapkan	Data tersimpan dan sistem menampilkan halaman <i>home</i>
Pengamatan	Data tersimpan dan sistem menampilkan halaman <i>home</i>
Hasil	Sesuai

Pengujian dilakukan dengan menekan tombol “*Sign Up*” pada halaman *register*, lalu muncul form “*Member Baru*”. Setelah itu masukkan data sesuai dengan form dan menekan tombol “*Register*”. Jika berhasil, sistem menampilkan halaman *home*. Tampilan hasil pengujian registrasi *member* (benar) dapat dilihat pada Gambar 5.8 dan 5.9.

Gambar 5.8 *Input* Registrasi (Benar)



Gambar 5.9 *Output* Pengujian Registrasi (Benar)

Berdasarkan gambar 5.8 dan gambar 5.9 dapat dilihat bahwa registrasi *member* baru berhasil dilakukan. Data tersimpan ke database dan sistem menampilkan notifikasi berhasil menambahkan *member*.

Untuk melakukan pembuktian bahwa data sudah diinputkan sesuai dengan keinginan, maka pengujian dilanjutkan dengan mengakses langsung ke *database* dan melakukan *query select* dengan *field* email pada tabel Users. Pembuktian dengan *query database* dapat dilihat pada gambar 5.10.

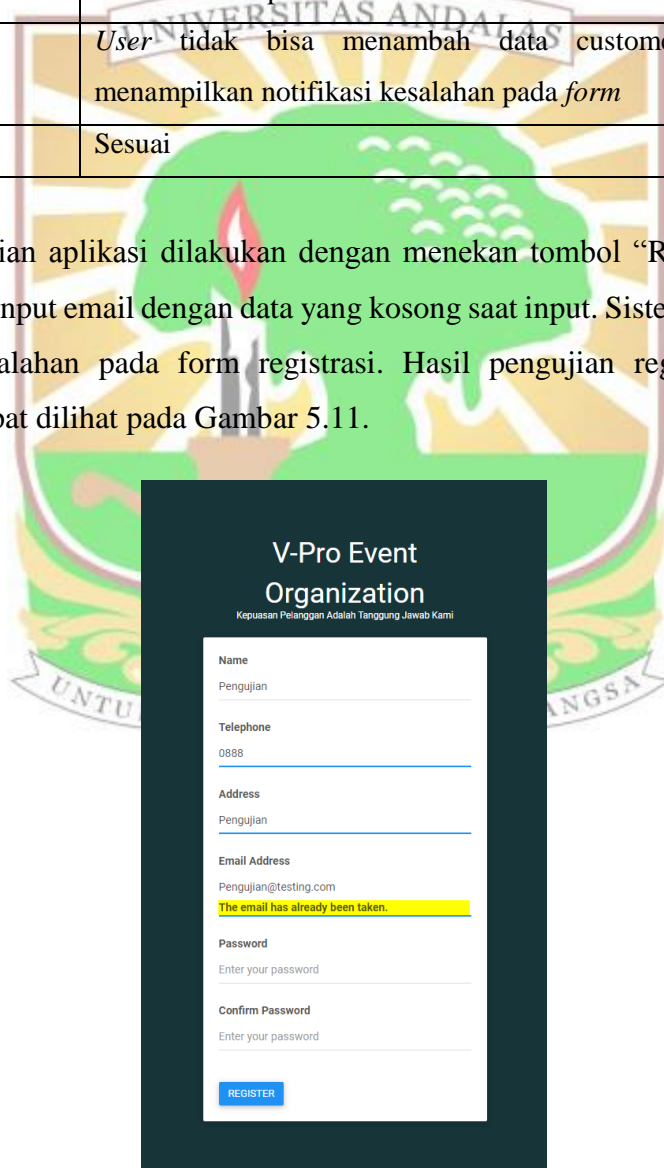
id_user	name	email	password	role	status	remember_token
U-201906221	Pengujian	Pengujian@testing.com	\$2y\$10\$Tc2FTm8qYCJE/rAkMeobe92DNdh7zHjy7EaqMonN65...	3	0	k7AODx06ahRS9QP1

Gambar 5.10 Pengecekan Pengujian Registrasi (Benar)

Tabel 5.3. Pengujian Registrasi (Alternatif)

Kasus dan Hasil Uji (Alternatif)	
Data masukan	<i>User</i> menekan tombol “Register” saat <i>form</i> registrasi <i>member</i> masih ada yang kosong
Yang diharapkan	Sistem menampilkan notifikasi kesalahan
Pengamatan	<i>User</i> tidak bisa menambah data customer dan sistem menampilkan notifikasi kesalahan pada <i>form</i>
Hasil	Sesuai

Pengujian aplikasi dilakukan dengan menekan tombol “Register” dengan mengisi form input email dengan data yang kosong saat input. Sistem menampilkan notifikasi kesalahan pada form registrasi. Hasil pengujian registrasi *member* (alternatif) dapat dilihat pada Gambar 5.11.



Gambar 5.11 Output Pengujian Registrasi (Alternatif)

5.2.2.2. Pengujian Tambah Transaksi

Pada pengujian ini dilakukan dengan membuka halaman tambah transaksi dan mengisi form transaksi. Setelah itu *user* menekan tombol tambah pada aplikasi untuk mengecek budget dan tambah item transaksi. Hasil pengujian tambah transaksi dapat dilihat pada tabel 5.4 dan tabel 5.5.

Tabel 5. 4. Pengujian Tambah Transaksi (Benar)

Kasus dan Hasil Uji (Benar)	
Data masukan	Semua data yang dibutuhkan dalam form registrasi “Tambah Transaksi”
Yang diharapkan	Transaksi sukses dan sistem menampilkan notifikasi transaksi berhasil pada halaman lihat transaksi
Pengamatan	Transaksi berhasil dan muncul notifikasi transaksi berhasil dan data tersimpan di <i>database</i>
Hasil	Sesuai

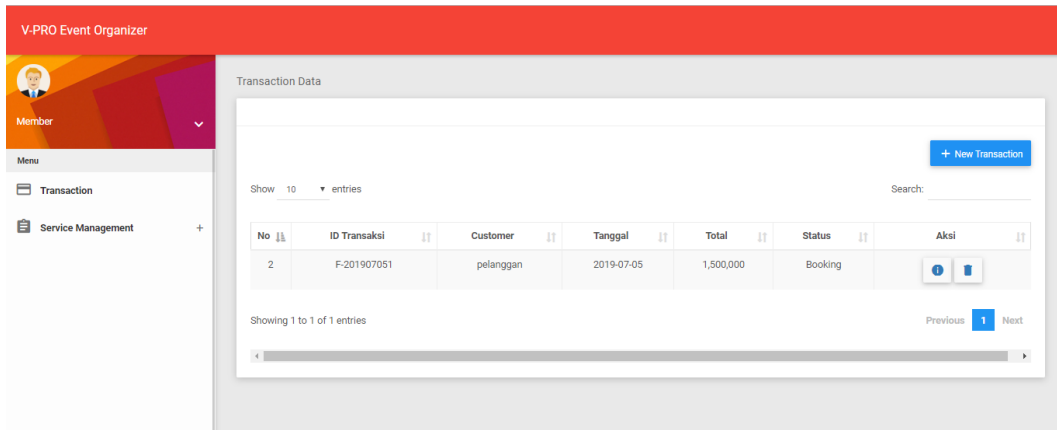
Pengujian dilakukan dengan membuka menu “Transaksi” pada halaman utama aplikasi, dan menekan submenu “*New Transaction*”. Selanjutnya menekan tombol tambah yang ada budget, maka muncul form tambah item transaksi. Selanjutnya tombol tambah di item transaksi ditekan untuk menambahkan item detail transaksi. Selanjutnya menekan tombol *save* untuk menyimpan transaksi ke *database*. Tampilan hasil pengujian tambah transaksi (benar) dapat dilihat pada gambar 5.12 dan 5.13.

The screenshot displays the 'Transaction Management' interface. On the left, there is a sidebar with a 'Member' profile and a 'Menu' containing 'Transaction' and 'Service Management'. The main content area shows the following details:

- Customer Type:** Member
- Customer Name:** U-201907051 - pelanggan - 0888
- Date:** 05/07/2019
- Order Detail:**
 - Set Budget:** Input field with value 50000000 and a plus button.
 - Pilih Service:** Dropdown menu with 'Multimedia' selected, followed by a plus button.

A 'Save' button is located at the bottom left of the form area.

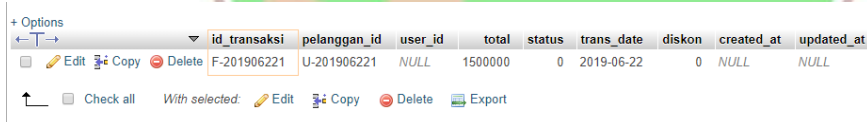
Gambar 5.12 *Input* Pengujian Tambah Transaksi (Benar)



Gambar 5.13 *Output* Pengujian Tambah Transaksi (Benar)

Berdasarkan gambar 5.12 dan gambar 5.13 dapat dilihat bahwa transaksi berhasil dilakukan. Sistem menampilkan notifikasi transaksi berhasil dan sistem menampilkan halaman daftar transaksi berlangsung.

Untuk melakukan pembuktian bahwa transaksi sesuai dengan keinginan, maka pengujian dilanjutkan dengan mengakses langsung ke *database* dan melakukan *query select* dengan *field* dengan id transaksi pada tabel transaksi. Jika data ditemukan, maka transaksi berhasil dilakukan. Pembuktian dengan *query database* dapat dilihat pada gambar 5.14.



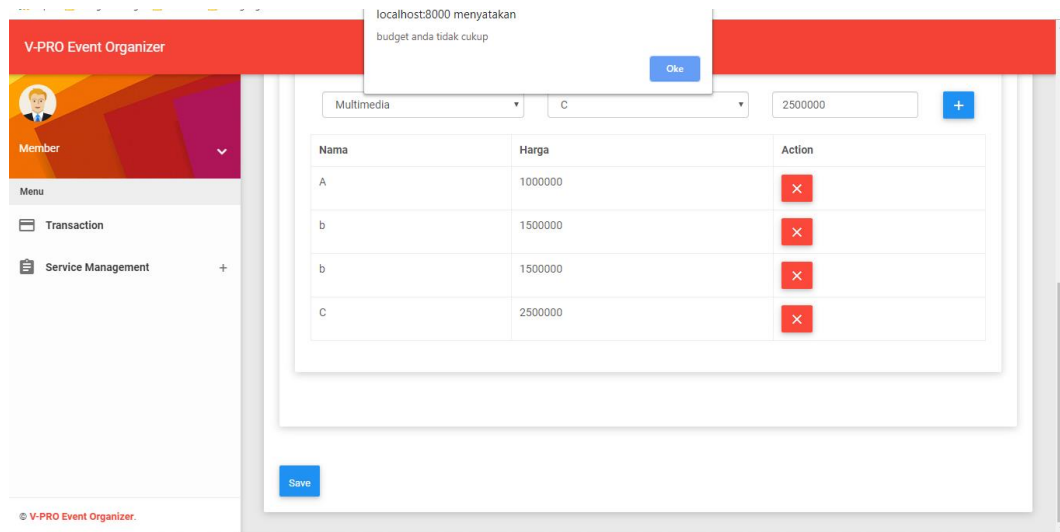
Gambar 5.14 Pengecekan Pengujian Transaksi (Benar)

Tabel 5. 5. Pengujian Tambah Transaksi (Alternatif)

Kasus dan Hasil Uji (Benar)	
Data masukan	Total harga melebihi data budget yang dimiliki
Yang diharapkan	Transaksi gagal dan sistem menampilkan notifikasi kesalahan
Pengamatan	sistem menampilkan notifikasi kesalahan
Hasil	Sesuai

Pengujian aplikasi dilakukan dengan menekan tombol *save* saat item total harga melebihi budget yang diinputkan. Kemudian sistem menampilkan notifikasi

kesalahan. Hasil pengujian tambah transaksi (alternatif) dapat dilihat pada gambar 5.15, dan 5.16.



Gambar 5.15 *Input Pengujian Tambah Transaksi (Alternatif).*

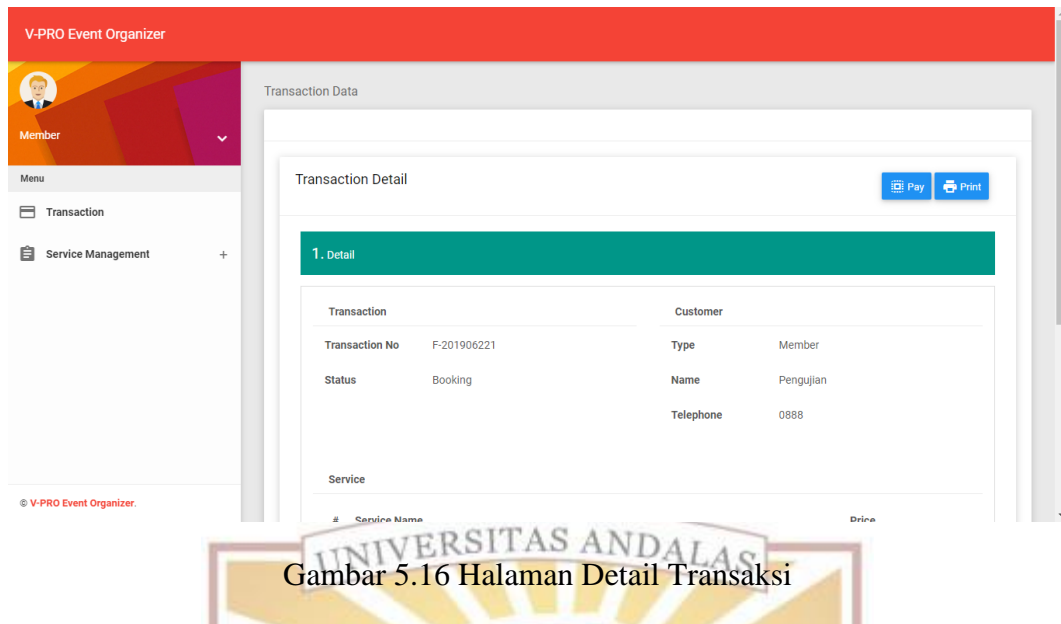
5.2.2.3. Pengujian Cetak Transaksi

Pada pengujian ini mengakses halaman detail transaksi. Pada halaman detail transaksi terdapat tombol cetak. Kemudian tombol cetak ditekan dan sistem mencetak laporan transaksi. Hasil pengujian cetak transaksi dapat dilihat pada tabel 5.6 dan 5.7.

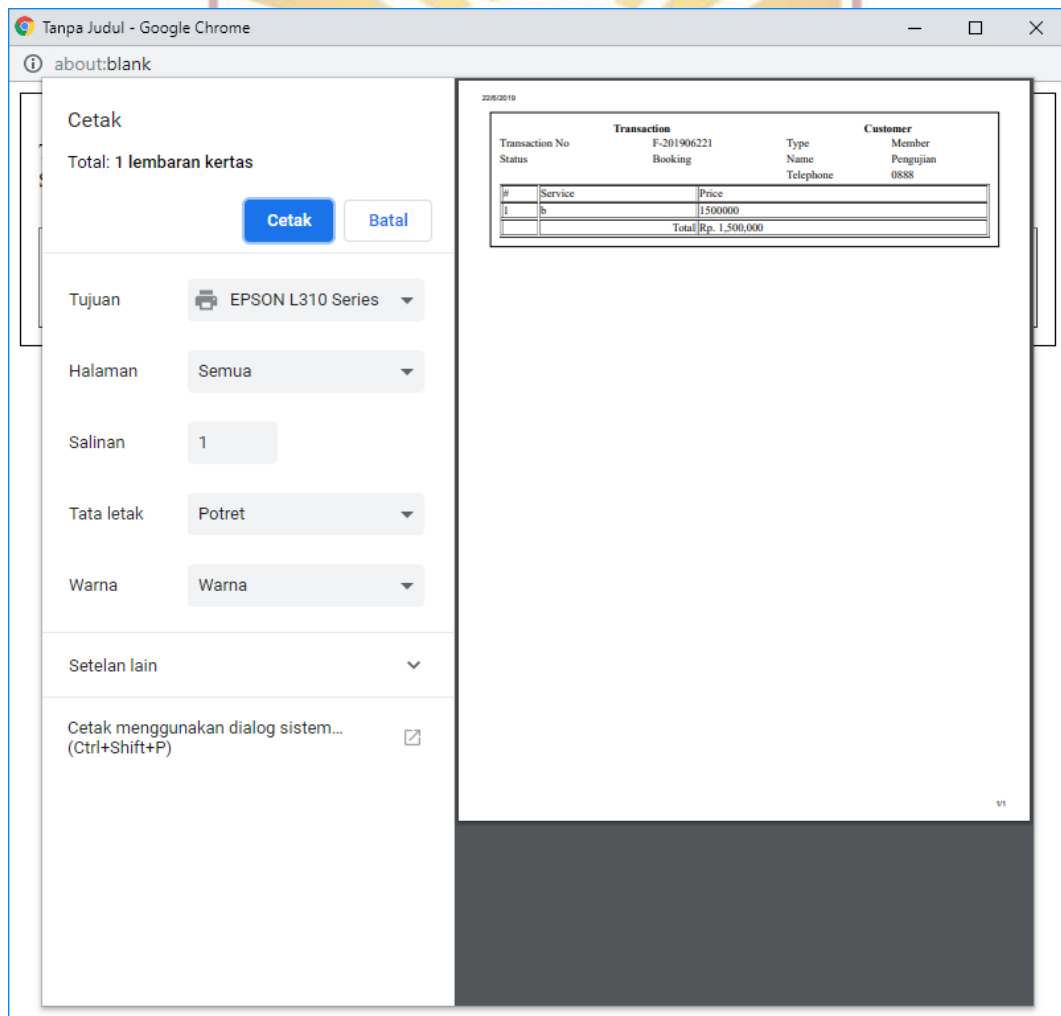
Tabel 5. 6. Pengujian Cetak Transaksi (Benar)

Kasus dan Hasil Uji (Benar)	
Data masukan	User menekan tombol lihat laporan
Yang diharapkan	Data didapatkan, dan muncul laporan transaksi berdasarkan rentang tanggal yang dipilih
Pengamatan	Muncul laporan transaksi yang akan dicetak
Hasil	Sesuai

Pengujian aplikasi dilakukan dengan menekan tombol “print” pada halaman detail. Kemudian sistem mencetak laporan dari transaksi yang dipilih. Hasil pengujian lihat laporan transaksi dapat dilihat pada Gambar 5.16.



Gambar 5.16 Halaman Detail Transaksi



Gambar 5.17 Tampilan Cetak Transaksi

5.3. Hasil Pengujian

Tahap pengujian yang dilakukan terhadap aplikasi yang dibangun, fokus pada ketersediaan dan kesesuaian fungsional sistem yang diuji secara manual. Setelah melakukan pengujian, didapatkan kesesuaian hasil antara perancangan dengan keluaran sistem, selain itu selama pengujian, juga tidak ditemukan kegagalan dalam setiap proses pada masing-masing fungsional. Dapat disimpulkan bahwa pembangunan aplikasi Pemesanan Layanan Event Organization pada CV V-Pro Event Organizer telah berjalan sesuai dengan fungsional yang dirancang. Hasil pengujian secara lengkap dijelaskan pada lampiran I. Kesimpulan dari hasil pengujian sistem dapat dilihat pada Tabel 5.7.

Tabel 5.7. Hasil Pengujian

No.	Fungsional	Skenario Pengujian		Hasil yang Diharapkan		Hasil
		Benar	Alternatif	Benar	Alternatif	
1	Melakukan Registrasi	Membuka Halaman <i>Sign Up</i> , Melakukan register, Input data benar	Melakukan <i>input</i> data salah pada setiap fungsi	Sistem menyimpan data user dan menampilkan halaman home	Sistem menampilkan notifikasi kesalahan	Sesuai
2	Melakukan <i>Login</i>	Membuka Halaman <i>Login</i> , Melakukan <i>Login</i>	Melakukan <i>input</i> data email atau password salah	Sistem melakukan autentifikasi dan menampilkan halaman home	Sistem menampilkan notifikasi kesalahan	Sesuai
3	Mengelola data Kategori	Membuka menu Kategori, menjalankan fungsi tambah, edit, cari, dan hapus data kategori, <i>input</i> data benar	Melakukan <i>input</i> data salah pada setiap fungsi	Sistem menampilkan, menyimpan, dan menghapus data kategori serta menampilkan notifikasi berhasil untuk setiap fungsi	Sistem menampilkan notifikasi kesalahan	Sesuai

Tabel 5.7. Hasil Pengujian (Lanjutan)

4	Melihat data Kategori	Membuka halaman kategori, menampilkan data kategori	-	Sistem menampilkan halaman kategori	Sistem menampilkan halaman kategori	Sesuai
5	Mengelola data <i>Service</i>	Membuka menu <i>Service</i> , menjalankan fungsi tambah, edit, cari, dan hapus data kategori, <i>input data</i> benar	Melakukan <i>input data</i> salah pada setiap fungsi	Sistem menampilkan, menyimpan, dan menghapus data <i>service</i> serta menampilkan notifikasi berhasil untuk setiap fungsi	Sistem menampilkan notifikasi kesalahan	Sesuai
6	Melihat data <i>Service</i>	Membuka halaman <i>Service</i> , menampilkan data kategori	-	Sistem menampilkan halaman <i>Service</i> ,	-	Sesuai
7	Tambah Transaksi	Membuka menu tambah transaksi, <i>input</i> transaksi baru, <i>input data</i> benar	Melakukan <i>input data</i> salah pada setiap fungsi	Sistem menyimpan data kategori serta menampilkan notifikasi berhasil untuk setiap fungsi	Sistem menampilkan notifikasi kesalahan	Sesuai
8	Melihat data Transaksi	Membuka halaman Transaksi, menampilkan data Transaksi	-	Sistem menampilkan halaman Transaksi	-	Sesuai
9	Menghapus Transaksi	Menekan tombol hapus, Menghapus data transaksi	-	Sistem menghapus data transaksi dari <i>database</i>	-	Sesuai

Tabel 5.7. Hasil Pengujian (Lanjutan)

10	Melihat data Detail Transaksi	Membuka halaman Detail Transaksi, menampilkan data Detail Transaksi	-	Sistem menampilkan halaman Detail Transaksi,	-	Sesuai
11	Ubah Status Transaksi	Membuka halaman transaksi, ubah status transaksi	-	Sistem mengubah data status transaksi	-	Sesuai
12	Cetak Laporan	Membuka halaman Detail Transaksi, Cetak	-	Sistem menampilkan jendela cetak transaksi,	-	Sesuai
13	Melihat Laporan	Membuka halaman Laporan, menampilkan Laporan Transaksi	-	Sistem menampilkan halaman Laporan,	-	Sesuai
14	Ubah Status Customer	Membuka halaman <i>Customer</i> , ubah status <i>Customer</i>	-	Sistem mengubah data status <i>customer</i>	-	Sesuai
15	Mengelola data <i>Customer</i>	Membuka menu <i>Customer</i> , menjalankan fungsi tambah, edit, dan hapus data kategori, <i>input data</i> benar	Melakukan <i>input data</i> salah pada setiap fungsi	Sistem menyimpan, dan hapus data <i>customer</i> serta menampilkan notifikasi	Sistem menampilkan notifikasi kesalahan	Sesuai

BAB VI PENUTUP

Bab ini berisi tentang kesimpulan dan saran pada tugas akhir ini. Kesimpulan merupakan hasil akhir atau pencapaian dari penelitian berdasarkan tujuan yang telah ditetapkan sebelumnya, sedangkan saran merupakan harapan yang diinginkan kedepannya terkait penelitian yang telah dilakukan.

6.1 Kesimpulan

Aplikasi pemesanan layanan *Event Organizer* Berbasis *Web* pada CV. V-Pro *Event Organizer* telah berhasil dibangun dengan metode *waterfall*. Dari setiap tahap yang dilakukan dalam metode *waterfall* yaitu tahap analisis, *design*, *coding*, dan implementasi dapat diperoleh kesimpulan sebagai berikut.

1. Dalam tahapan analisis diperoleh data dan informasi berupa kedudukan sistem, dimana proses bisnis yang sedang berjalan sebelumnya pada CV. V-Pro *Event Organizer* yang masih manual dan belum terkomputerisasi perlu diubah menjadi proses yang terkomputerisasi dan terintegrasi. Pada tahapan ini juga dianalisis alur proses bisnis dari CV. V-Pro *Event Organizer* yang digambarkan menggunakan *tool Business Process Modelling Notation* (BPMN). Proses bisnis yang digambarkan adalah proses registrasi dan pemesanan layanan *event organizer*. Terdapat lima belas kebutuhan fungsional yang didapatkan dari analisis BPMN. Kebutuhan fungsional yang telah dianalisis, kemudian diuraikan dan digambarkan dengan *use case diagram* sehingga menghasilkan dua aktor yaitu admin dan *customer*. Selanjutnya seluruh fungsional digambarkan dengan *use case scenario*, *sequence diagram*, dan *class analysis*.
2. Dalam tahapan *design* dilakukan penggambaran aplikasi yang dibangun menggunakan *entity relationship diagram* (ERD) yang terdiri dari enam tabel beserta struktur tabel dan basis data. Pada *class diagram* digambarkan kelas-kelas yang digunakan pada aplikasi, dimana kelas *controller* terbagi atas tujuh kelas *controller*, dan kelas *model* terbagi atas tujuh kelas *model*. Tahapan *design* selanjutnya menggambarkan antar muka (*user interface*)

dari aplikasi *web*, serta arsitektur aplikasi yang menjelaskan alur sistem dan hubungan antar komponen aplikasi *web*.

3. Sistem ini sudah bisa diakses oleh admin dan *customer* untuk melakukan pengelolaan dan penyewaan jasa *event organizer*, yaitu dengan cara aktor *login* pada aplikasi lalu melakukan transaksi.

6.2 Saran

Aplikasi pemesanan layanan *event organizer* pada CV V-Pro Event Organizer masih membutuhkan pengembangan lebih lanjut yang sesuai dengan kebutuhan fungsional. Dalam pengembangan aplikasi diharapkan adanya kerjasama dengan pihak - pihak terkait seperti layanan fotografi, media percetakan, bank, atau pihak-pihak lainnya sehingga semua yang berkaitan dapat terintegrasi dengan baik. Selain menggunakan aplikasi *web*, seharusnya juga menggunakan aplikasi *mobile*. Dengan demikian penyewaan *event* dapat dilakukan oleh *customer* dimana saja dan kapan saja hanya dengan mengakses satu aplikasi saja.



DAFTAR PUSTAKA

- Aini, F. N. (2014). *Pemodelan Arsitektur Enterprise Menggunakan TOGAF ADM untuk mendukung Sistem Informasi Promosi pada Perguruan Tinggi (Studi Kasus Universitas Respati Yogyakarta)*.
- Dewi, L. P., Indahyanti, U., & S, Y. H. (2012). *Pemodelan Proses Bisnis Menggunakan Activity Diagram UML DAN BPMN (Studi Kasus FRS ONLINE)*.
- Dharwiyanti, S. &. (2003). *Pengantar Unified Modeling Language (UML)*. May 2017. Diambil kembali dari www.IlmuKomputer.com
- Eka. (2018). *sistem informasi penyewaan peralatan event organizer berbasis web pada PT. adecon Jakarta. JURNAL ABDIMAS BSI*.
- Fahrudin. (2017). *Perancangan sistem informasi pemesanan event organizer berbasis website menggunakan framework codeigniter (studi kasus PT. Munji Organizer Semarang). Artikel ilmiah*.
- Fitriyani, M. R. (2017). *Sistem Pendukung Keputusan Untuk Kenaikan Jabatan Aparatufitrr Sipil Negara Struktural Pada Badan Pengelolaan Keuangan Dan Aset Daerah (BPKAD) Kabupaten Lampung Tengah Dengan Metode Simple Additive Weighting”*,. Lampung.
- Haryani. (2015). *Pemanfaatan Web Sistem Informasi Akademik Sebagai Pengontrol Nilai Siswa (Studi Kasus: MTS Pondok Moderen AL-Islam Nganjuk. Paradigma* . Volume 18.
- Kamil, H. (2013). *Pengembangan Aplikasi Distribusi Surat Di Fakultas Teknologi Informasi Dengan Notifikasi SMS Menggunakan Framework Yii dan GAMM*.
- Kennedy, J. E. (2009). *Manajemen Event. PT Bhuana Ilmu Populer*. Jakarta.
- Marselia, M. (2012). *Pembuatan Media Pembelajaran Berbasis Film Animasi Kartun Pada Pengenalan Perangkat Keras Komputer Dalam Pembelajaran TIK*. Dipetik May 2017, dari repository.upi.edu

- McCool, S. (2012). *Laravel Starter Birmingham, UK ed. s.l.* Packt Publishing.
- Megananda, Y. d. (2009). *7 Langkah Jitu Membangun Bisnis Event Organizer*?. Jakarta: PT. Bhuana Ilmu Populer.
- Mulyani, S. (2016). *Sistem Informasi Manajemen Rumah Sakit: Analisis dan Perancangan*. Bandung: Abdi Sistematika.
- Online. (2011, Februari 7). Diambil kembali dari Freedictionary.com: <http://www.thefreedictionary.com/online>.
- Pertiwi, D. H. (2011). *Desain Dan Implementasi Sistem Informasi Perpustakaan Berbasis WEB Dengan MVC (Model View Controller)*. Volume 2.
- Pressman, R. S. (2001). *Software Engineering. Dalam A Practitioner's Approach*. Mc-Graw Hill.
- Pudjiastuti, W. (2010). *Special Event Alternatif Jitu Membidik Pasar*. Dalam P. E. Komputindo.. Jakarta.
- Riadi, M. (2013, 12). *Use Case Diagram*. Diambil kembali dari comuse-case-diagram.html : <http://www.kajianpustaka>.
- Rizqi, M. H. (2017). *Aplikasi Pengaduan, Perizinan, Dan Perkembangan Nilai Siswa Berbasis Web di Sma Negeri 8 Bandung. E Proceeding of Applied Science*.
- Romaida. (2017). *Sistem Infomasi Pengolahan Jasa Event Organizer*?. Universitas Komputer Indonesia.
- Rosa dan Shalahudin, M. (2013). *Rekayasa Perangkat Lunak Terstruktur dan Berorientasi Objek*. Bandung: Informatika.
- Ruslan, R. (2007). *Manajemen Public Relations & Media komunikasi*. Jakarta: PT Raja Grafindo Persada.
- Sentosa, A. &. (2015). *Aplikasi e-Booking Rumah Makan Berbasis Web Dengan Penerapan Arsitektur Model View Controller (Studi Kasus: Rumah Makan kampoeng bamboe)*. *Jurnal Manajemen Sistem Informasi Dan Teknologi*, p. 61.

Setyawan, S. (2009). Diambil kembali dari Akibat Hukum Hubungan Kerjasama Antara EO (event organizer) Dan Manajemen Band Di Surakarta (Studi Kasus Di PT. Musikita Di Surakarta: <http://eprints.ums.ac.id/6006/1/C100020249.pdf>

Stone, R. a. (2006). The Case Study Approach to Scenario Planning. *Journal of Practical Consulting*, pp. 7-18.

Subekti, R. (2008). *Kitab Undang-Undang Hukum Perdata (Burgerlijk Wetboek voor Indonesie)*. Jakarta: PT. Pratnya Paramita.

Sulistiyanto. (2012). Aplikasi Sistem Informasi Paiton Resort Hotel Online. *Jurnal Cyber-Techn*, Volume: 6.

wijaya, a. (2017). buku profil cv.v-pro. Dalam a. wijaya. padang.

Wirjono, P. (2009). *Asas-Asas Hukum Pidana di Indonesia*, . Bandung : Refika.

Yoshiko, A. d. (2015). Rancang Bangun WEB Service Sistem Informasi Terintegrasi Gereja Masehi Injili Di Minahasa” (Studi Kasus : Gereja Gmim Getsemani Lansot Tomohon). Volume 5.







LAMPIRAN A

SURAT KETERANGAN

Yang bertanda tangan dibawah ini:

Nama : Agung Wijaya

Jabatan : Pemilik

Menyatakan Bahwa Siswa Perguruan Tinggi Negeri Yang Bernama:

Nama : Maghfirah Maulani

Nim : 1411521003

Jurusan/Fakultas : Sistem Informasi/Teknologi Informasi

PTN : Universitas Andalas Padang

Benar telah melakukan penelitian atau pengambilan data pada CV V-Pro dalam rangka penyusunan tugas akhir dengan judul **“Pembangunan Sistem *Booking Online* dan *Penyewaan Jasa Event Organizer* Berbasis *Web* Pada CV. V-Pro Kota Padang”**.

Demikian surat keterangan ini kami berikan untuk dapat digunakan sebagaimana mestinya.

Padang, 23 April 2019

Pemilik CV. V-Pro



(Agung Wijaya)



LAMPIRAN B

Use Case Scenario Aplikasi

1. Use Case Scenario Melakukan Login

<i>Use Case</i>	Login
<i>Actor</i>	Customer, Admin
<i>Entry Condition</i>	Actor telah membuka halaman awal
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor memilih menu "Login" 2. Sistem menampilkan Halaman Login. 3. Actor mengisi form Login 4. Actor menekan tombol "Sign in" 5. Sistem mencari username dan password 6. Sistem menampilkan halaman awal
<i>Scenario Alternatif</i>	<ol style="list-style-type: none"> 7. Actor menekan tombol "sign in" saat data kosong. 8. Sistem menampilkan halaman "login" dan pesan error
<i>Exit Condition</i>	Sistem menampilkan halaman login dan pesan error

2. Use Case Scenario Edit Kategori

<i>Use Case</i>	Edit Kategori
<i>Actor</i>	Admin
<i>Entry Condition</i>	Actor telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor membuka halaman "Service management" 2. Sistem menampilkan halaman "service management" 3. Actor memilih menu "Kategori" 4. Sistem menampilkan Halaman Kategori. 5. Actor menekan tombol "edit" 6. Sistem menampilkan halaman edit kategori 7. Actor mengisi form edit kategori 8. Actor menekan tombol submit 9. Sistem menyimpan data kategori 10. Sistem menampilkan halaman kategori
<i>Scenario Alternatif</i>	<ol style="list-style-type: none"> 11. Actor menekan tombol submit saat data kosong atau ada kesalahan. 12. Sistem menampilkan halaman kategori dan pesan error.
<i>Exit Condition</i>	Sistem menampilkan halaman kategori dan pesan error

3. Use Case Scenario Hapus Kategori

<i>Use Case</i>	Hapus Kategori
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman “Service management” 2. Sistem menampilkan halaman “service management”. 3. <i>Actor</i> memilih menu “Kategori” 4. Sistem menampilkan Halaman Kategori. 5. <i>Actor</i> menekan tombol “hapus” 6. Sistem menghapus data kategori 7. Sistem menampilkan halaman kategori
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman kategori

4. Use Case Scenario Lihat Kategori

<i>Use Case</i>	Lihat Kategori
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman “service management” 2. Sistem menampilkan halaman “service management”. 3. <i>Actor</i> memilih menu “Kategori” 4. Sistem menampilkan Halaman Kategori.
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman

5. Use Case Scenario Tambah Service

<i>Use Case</i>	Tambah Service
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman “Service management” 2. Sistem menampilkan halaman “Service management”. 3. <i>Actor</i> memilih menu “Service” 4. Sistem menampilkan Halaman Service. 5. <i>Actor</i> menekan tombol “new service” 6. Sistem menampilkan halaman tambah Service 7. <i>Actor</i> mengisi form tambah service 8. <i>Actor</i> menekan tombol submit 9. Sistem menyimpan data service 10. Sistem menampilkan halaman service

<i>Scenario Alternatif</i>	11. <i>Actor</i> menekan tombol submit saat data kosong atau ada kesalahan. 12. Sistem menampilkan halaman service dan pesan error.
<i>Exit Condition</i>	Sistem menampilkan halaman service dan pesan error

6. Use Case Scenerio Edit Service

<i>Use Case</i>	Edit Service
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman “Service management” 2. Sistem menampilkan Halaman “service management”. 3. <i>Actor</i> memilih menu “Service” 4. Sistem menampilkan Halaman Service. 5. <i>Actor</i> menekan tombol “edit” 6. Sistem menampilkan halaman edit Service 7. <i>Actor</i> mengisi form edit Service 8. <i>Actor</i> menekan tombol submit 9. Sistem menyimpan data Service 10. Sistem menampilkan halaman Service
<i>Scenario Alternatif</i>	11. <i>Actor</i> menekan tombol submit saat data kosong atau ada kesalahan. 12. Sistem menampilkan halaman Service dan pesan error.
<i>Exit Condition</i>	Sistem menampilkan halaman Service dan pesan error

7. Use Case Scenerio Hapus Service

<i>Use Case</i>	Hapus Service
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman “Service management” 2. Sistem menampilkan Halaman “service management”. 3. <i>Actor</i> memilih menu “Service” 4. Sistem menampilkan Halaman Service. 5. <i>Actor</i> menekan tombol “hapus” 6. Sistem menghapus data service 7. Sistem menampilkan halaman service

<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman service

8. Use Case Scenerio Lihat Service

<i>Use Case</i>	Lihat Service
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman "Service management" 2. Sistem menampilkan Halaman "service management". 3. <i>Actor</i> memilih menu "service" 4. Sistem menampilkan Halaman Service.
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman Service

9. Use Case Scenerio Lihat Transaksi

<i>Use Case</i>	Lihat Transaksi
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> memilih menu "transaction" 2. Sistem menampilkan Halaman transaksi.
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman transaksi

10. Use Case Scenerio Hapus Transaksi

<i>Use Case</i>	Hapus Transaksi
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman "Transaction" 2. Sistem menampilkan Halaman "transaction". 3. <i>Actor</i> menekan tombol "hapus" 4. Sistem menghapus data transaksi 5. Sistem menampilkan halaman transaksi
<i>Scenario Alternatif</i>	-

<i>Exit Condition</i>	Sistem menampilkan halaman transaksi
-----------------------	--------------------------------------

11. Use Case Scenario Lihat Detail Transaksi

<i>Use Case</i>	Lihat Detail Transaksi
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> memilih menu “transaction” 2. Sistem menampilkan Halaman transaksi. 3. <i>Actor</i> menekan tombol show 4. Sistem menampilkan halaman detail transaksi
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman detail transaksi

12. Use Case Scenario Ubah Status transaksi

<i>Use Case</i>	Lihat Transaksi
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> memilih menu “transaction” 2. Sistem menampilkan Halaman transaksi. 3. <i>Actor</i> menekan tombol show 4. Sistem menampilkan halaman detail transaksi 5. <i>Actor</i> menekan tombol Finish 6. Sistem mengubah status transaksi
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem mengubah status transaksi

13. Use Case Scenario Lihat Laporan

<i>Use Case</i>	Lihat Laporan
<i>Actor</i>	Admin
<i>Entry Condition</i>	<i>Actor</i> telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. <i>Actor</i> membuka halaman “report” 2. Sistem menampilkan Halaman laporan.
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman laporan

14. Use Case Scenario Ubah Status Member

<i>Use Case</i>	Ubah Status Member
<i>Actor</i>	Admin
<i>Entry Condition</i>	Actor telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor memilih menu “user management” 2. Sistem menampilkan Halaman “user management”. 3. Actor menekan tombol option 4. Sistem mengubah status member
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem mengubah status member

15. Use Case Scenario Lihat Member

<i>Use Case</i>	Lihat Member
<i>Actor</i>	Admin, Customer
<i>Entry Condition</i>	Actor telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor memilih menu “User management” 2. Sistem menampilkan Halaman Member.
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman member

16. Use Case Scenario Tambah Member

<i>Use Case</i>	Tambah Member
<i>Actor</i>	Admin
<i>Entry Condition</i>	Actor telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor memilih menu “Member” 2. Sistem menampilkan Halaman Member. 3. Actor menekan tombol “registration” 4. Sistem menampilkan halaman tambah Member 5. Actor mengisi form tambah Member 6. Actor menekan tombol submit 7. Sistem menyimpan Member 8. Sistem menampilkan halaman Member
<i>Scenario Alternatif</i>	<ol style="list-style-type: none"> 9. Actor menekan tombol submit saat data kosong. 10. Sistem menampilkan pesan error.
<i>Exit Condition</i>	Sistem menampilkan halaman pesan error

17. Use Case Scenario Edit Member

<i>Use Case</i>	Edit Member
<i>Actor</i>	Admin
<i>Entry Condition</i>	Actor telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor memilih menu “Member” 2. Sistem menampilkan Halaman Member. 3. Actor menekan tombol “edit” 4. Sistem menampilkan halaman edit Member 5. Actor mengisi form edit Member 6. Actor menekan tombol submit 7. Sistem menyimpan data Member 8. Sistem menampilkan halaman Member
<i>Scenario Alternatif</i>	<ol style="list-style-type: none"> 9. Actor menekan tombol submit 10. Sistem menampilkan halaman Member dan pesan error.
<i>Exit Condition</i>	Sistem menampilkan halaman Member dan pesan error

18. Use Case Scenario Hapus data Member

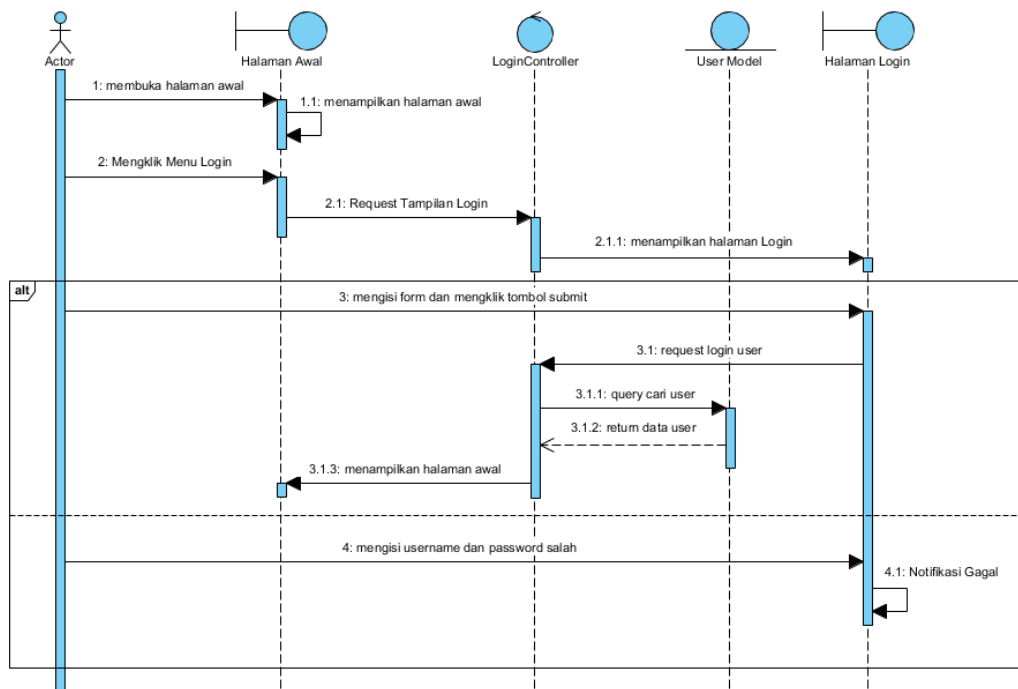
<i>Use Case</i>	Hapus Data Member
<i>Actor</i>	Admin
<i>Entry Condition</i>	Actor telah Login
<i>Flow of Event</i>	<ol style="list-style-type: none"> 1. Actor memilih menu “Member” 2. Sistem menampilkan Halaman Member. 3. Actor menekan tombol “hapus” 4. Sistem menghapus data member 5. Sistem menampilkan halaman member
<i>Scenario Alternatif</i>	-
<i>Exit Condition</i>	Sistem menampilkan halaman member

LAMPIRAN C

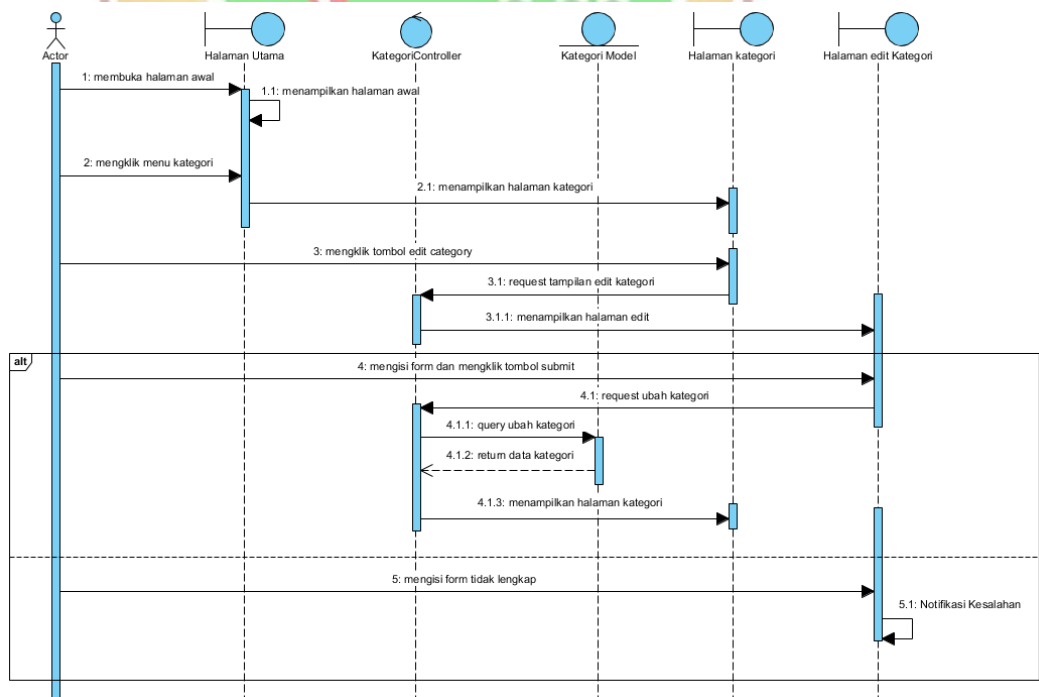


Sequence Diagram Aplikasi

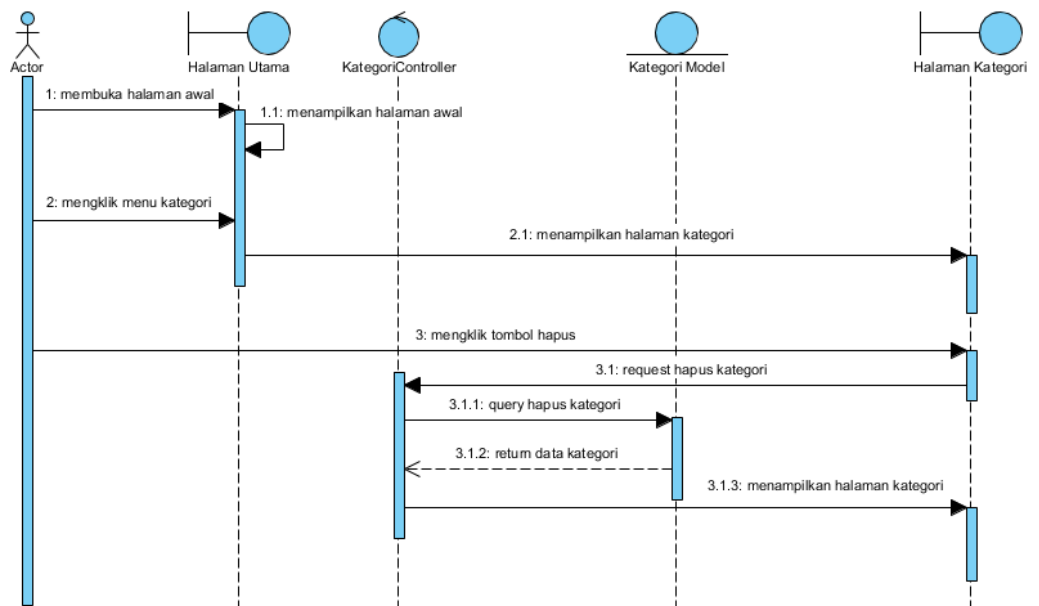
1. *Sequence Diagram Melakukan Login*



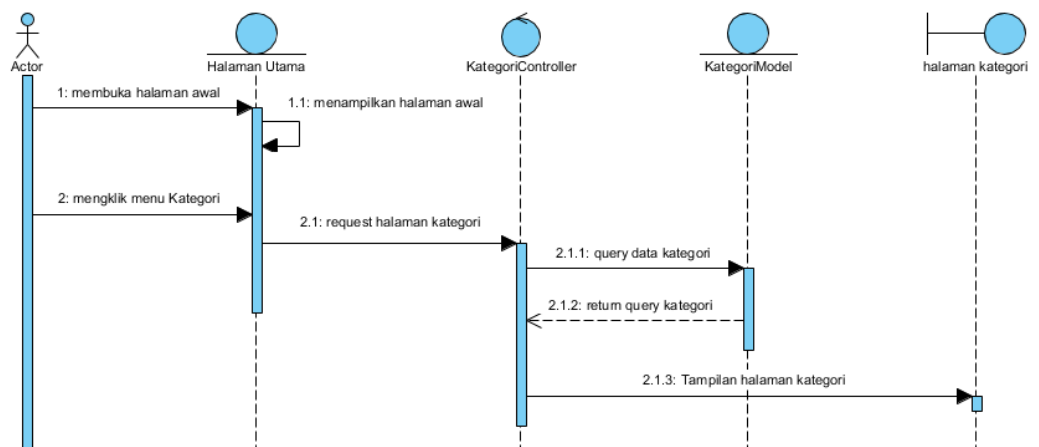
2. Sequence Diagram Edit Kategori



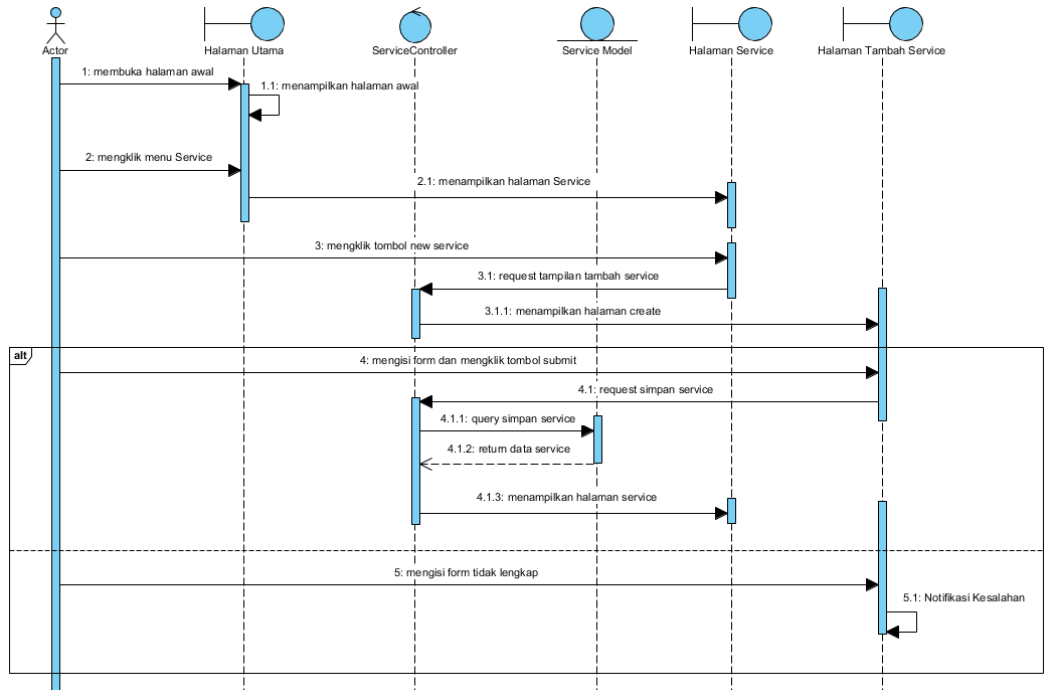
3. Sequence Diagram Hapus Kategori



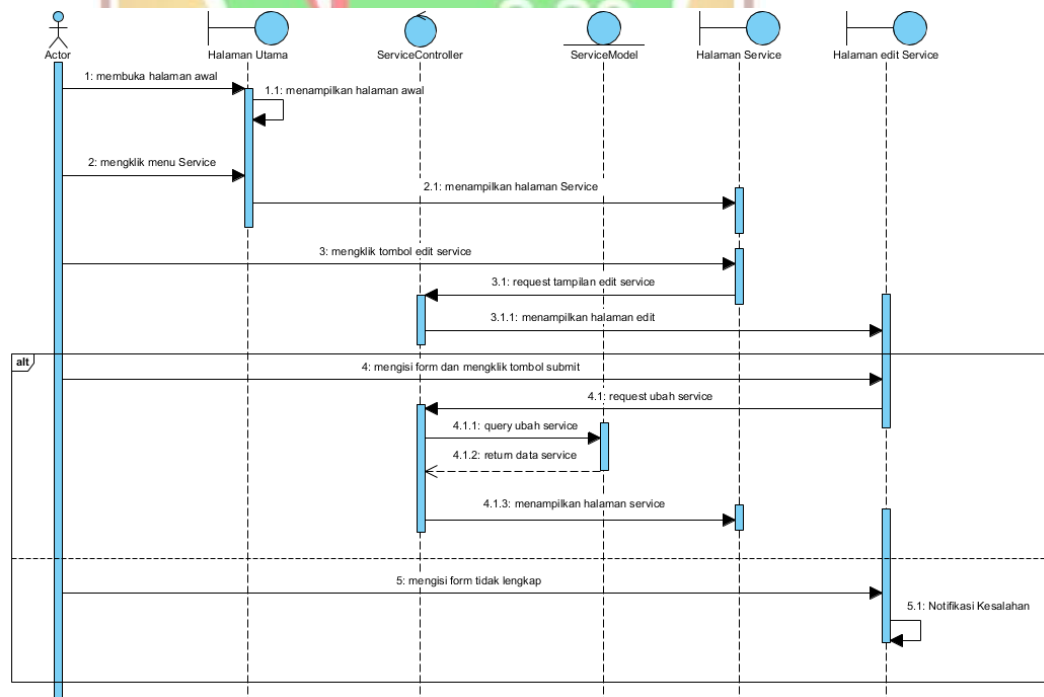
4. Sequence Diagram Lihat Kategori



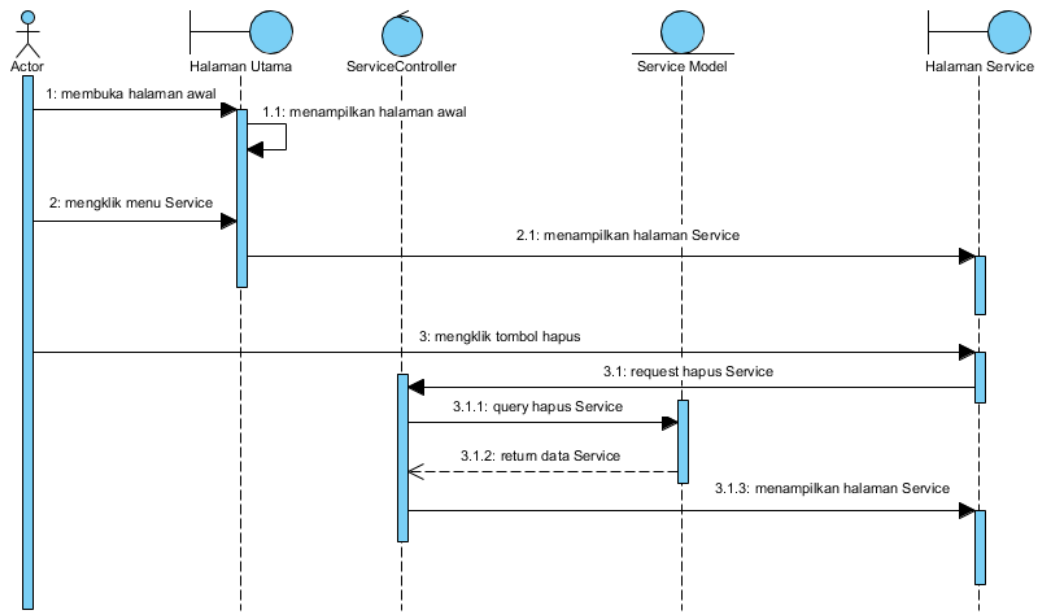
5. Sequence Diagram Tambah Service



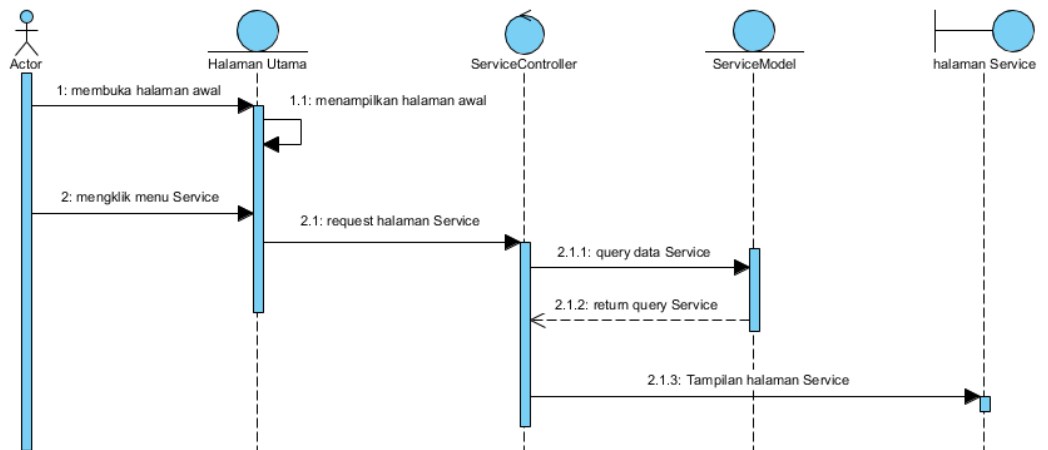
6. Sequence Diagram Edit service



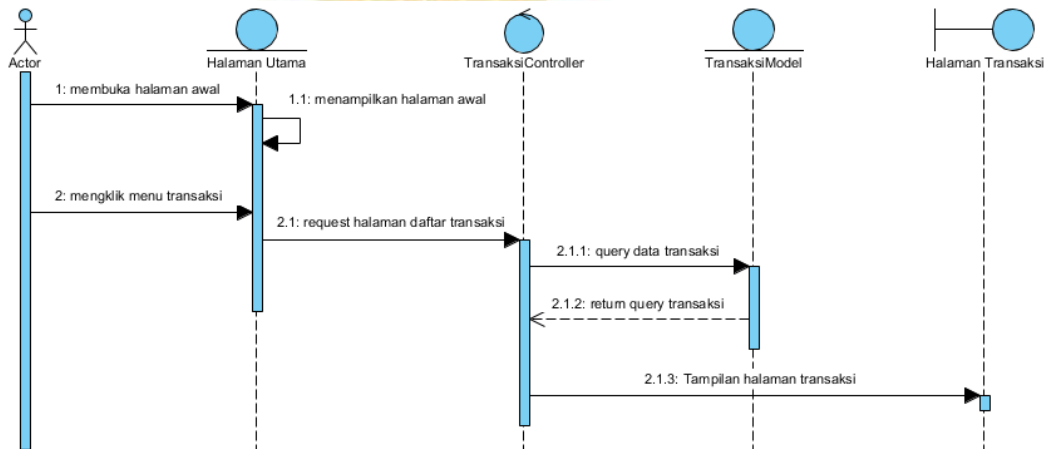
7. Sequence Diagram Hapus Service



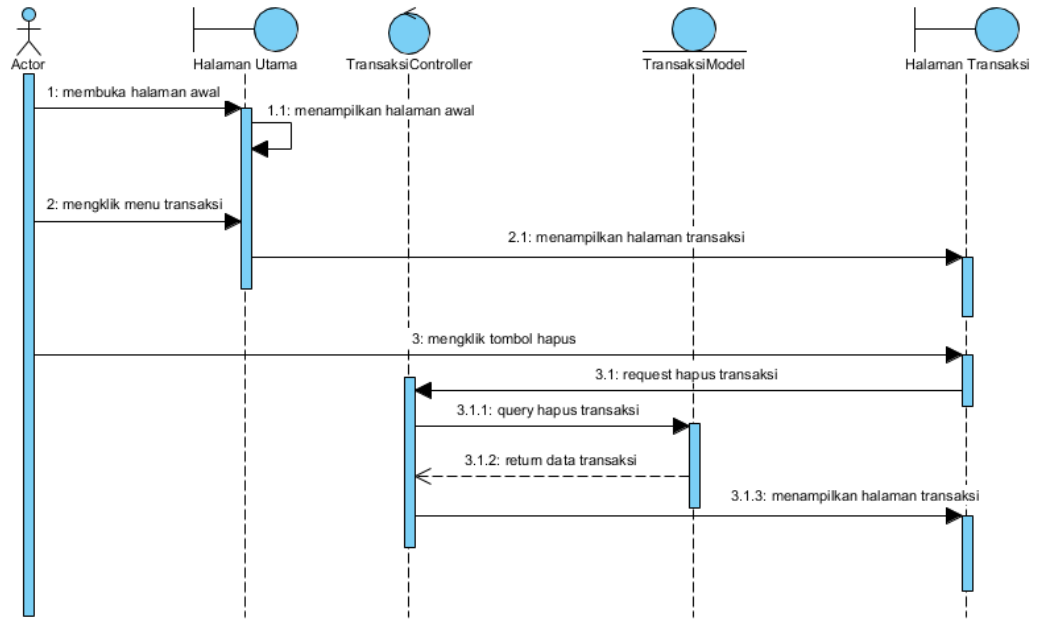
8. *Sequence Diagram* Lihat Service



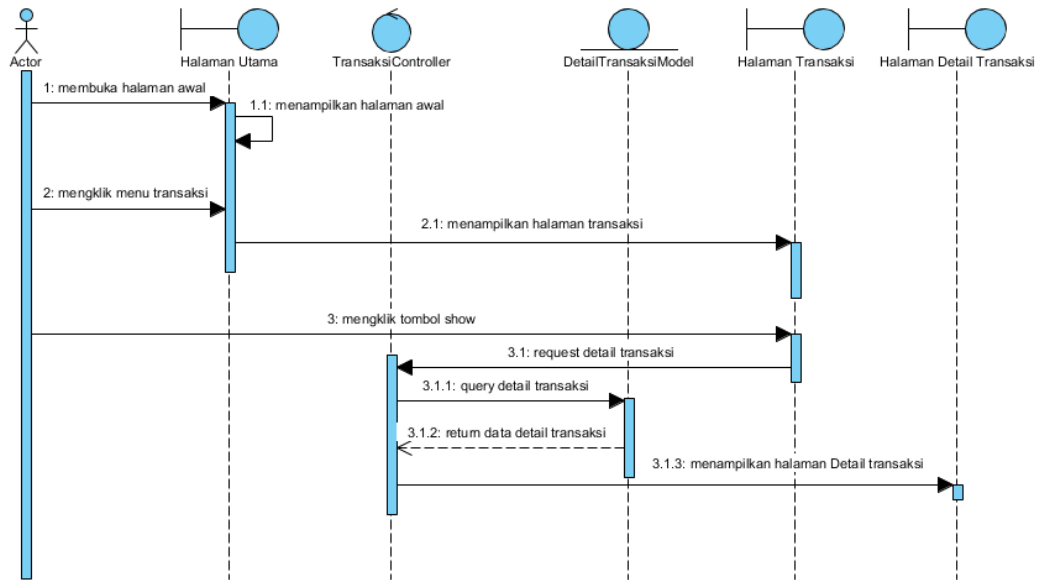
9. *Sequence Diagram* Lihat Transaksi



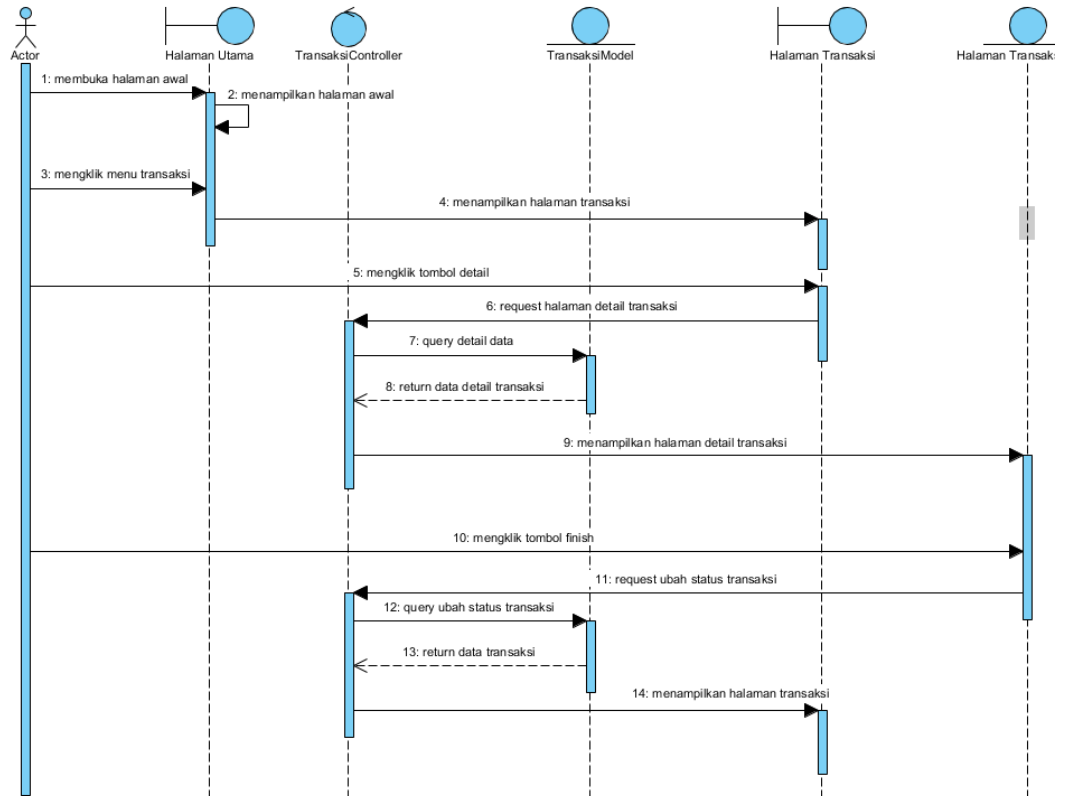
10. *Sequence Diagram* Hapus Transaksi



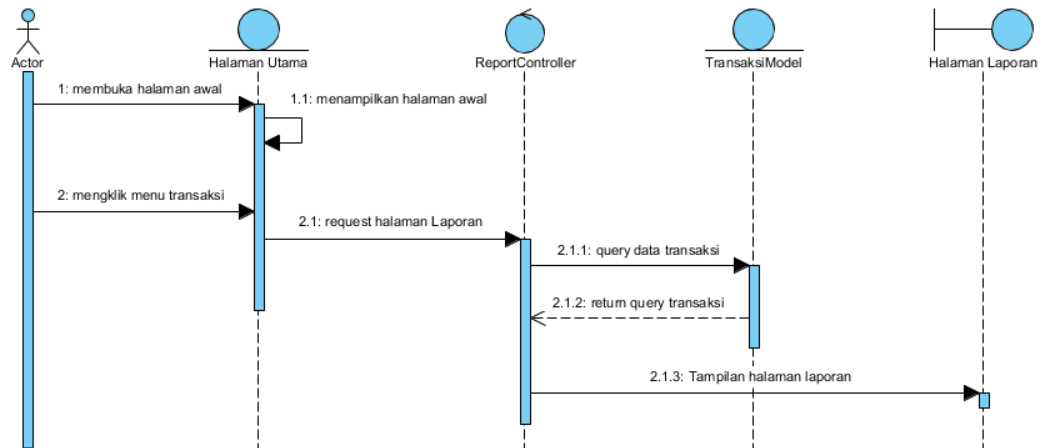
11. Sequence Diagram Lihat Detail Transaksi



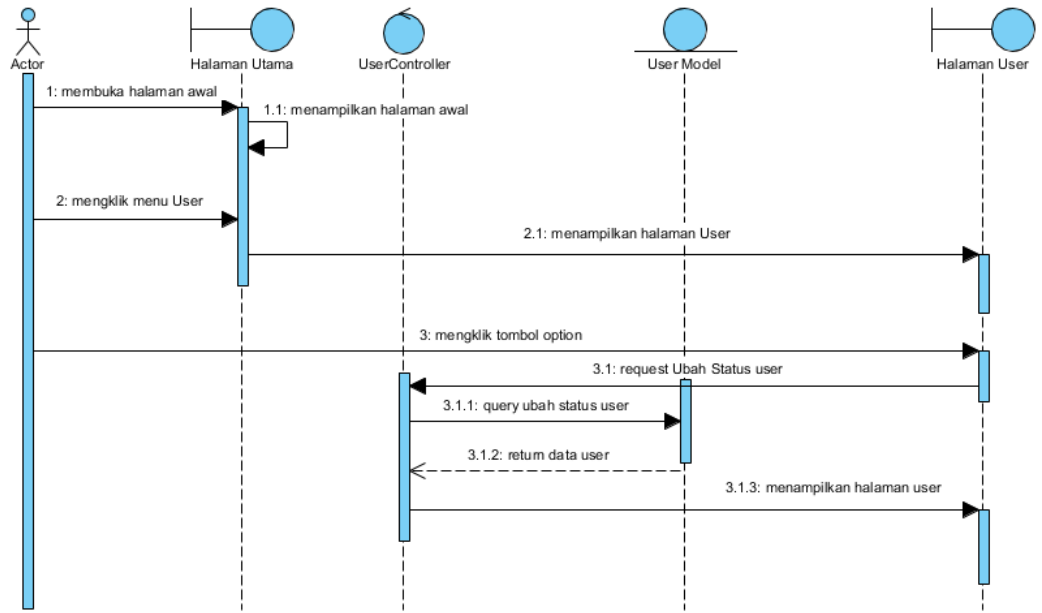
12. Sequence Diagram Ubah Status Transaksi



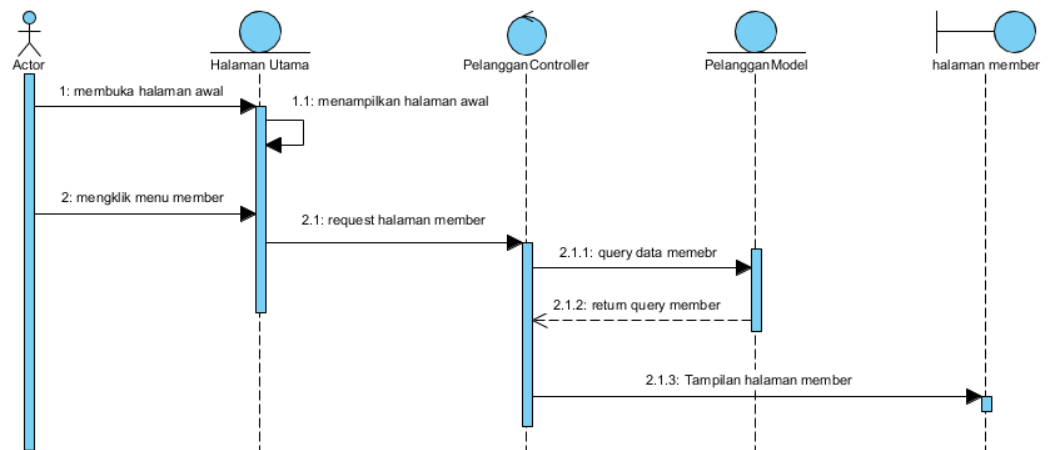
13. Sequence Diagram Lihat Laporan



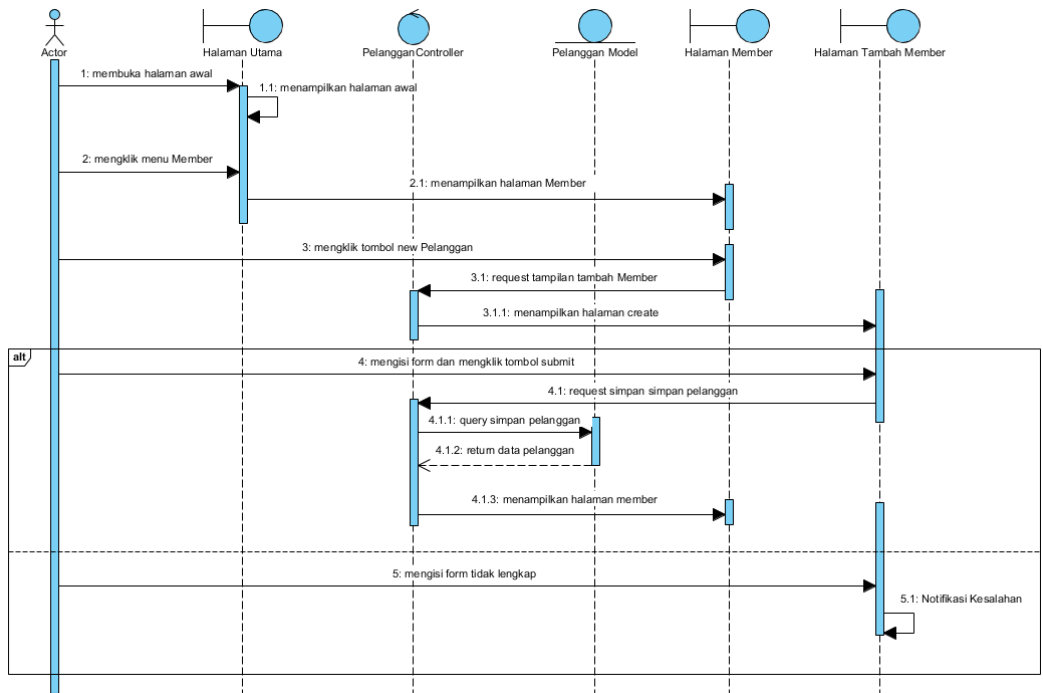
14. Sequence Diagram Ubah Status Member



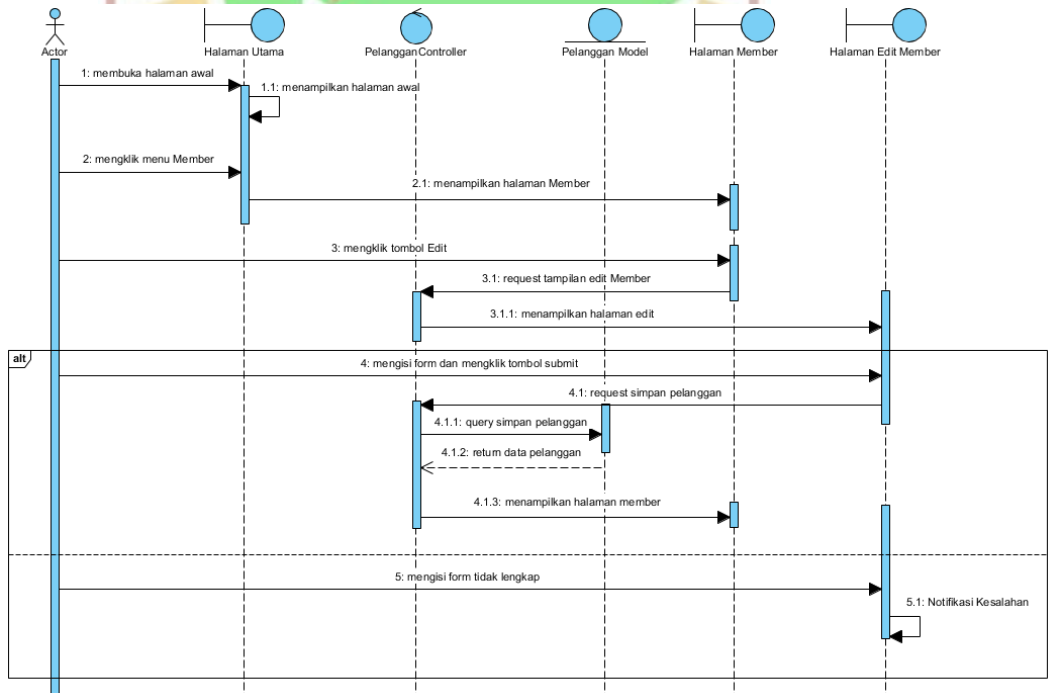
15. Sequence Diagram Lihat Member



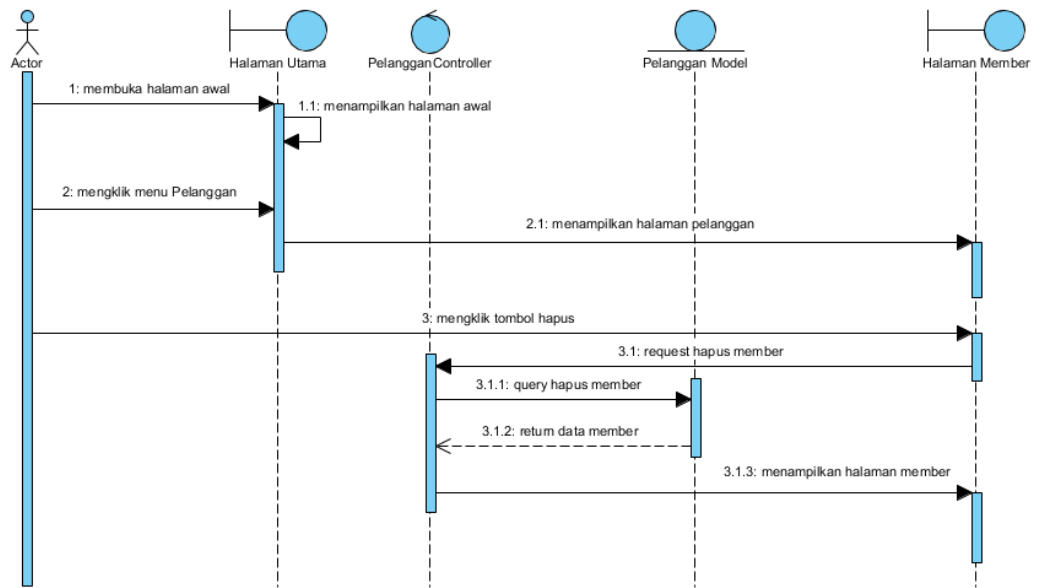
16. Sequence Diagram Tambah Member



17. Sequence Diagram Edit Member



18. Sequence Diagram Hapus Data Member

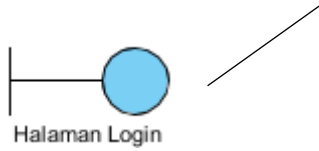


LAMPIRAN D

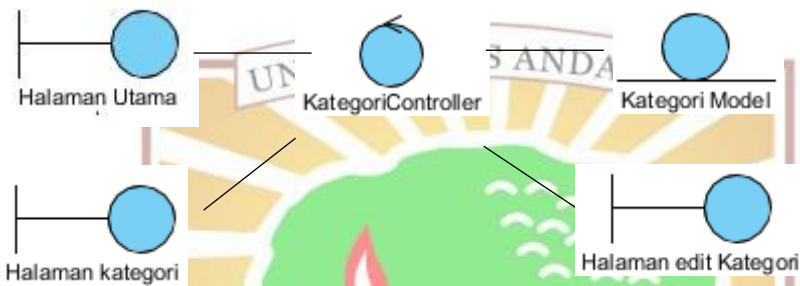


Class Analysis Aplikasi

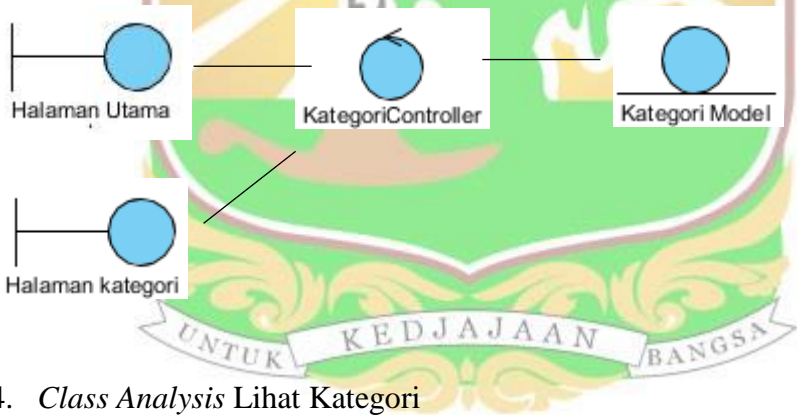
1. *Class Analysis Melakukan Login*



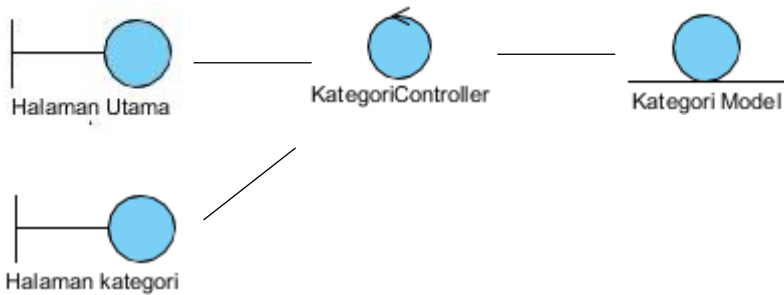
2. *Class Analysis* Edit Kategori



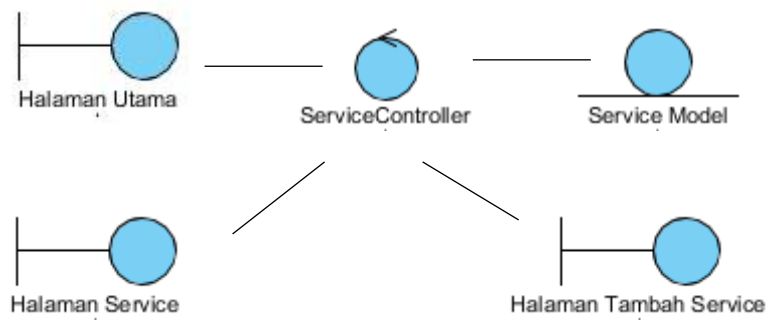
3. *Class Analysis* Hapus Kategori



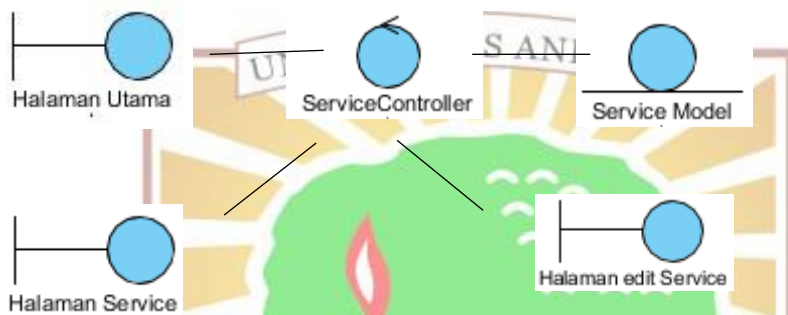
4. *Class Analysis* Lihat Kategori



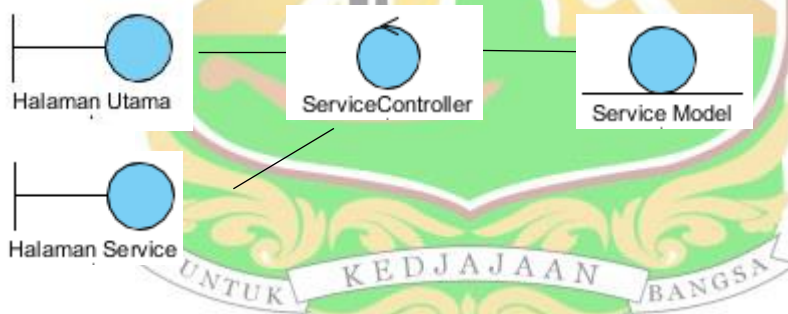
5. *Class Analysis* Tambah Service



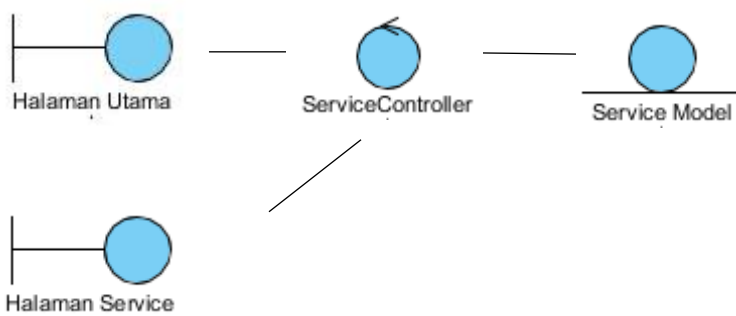
6. *Class Analysis* Edit Service



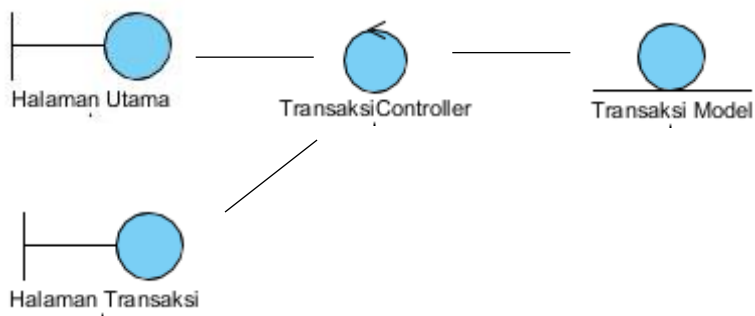
7. *Class Analysis* Hapus Service



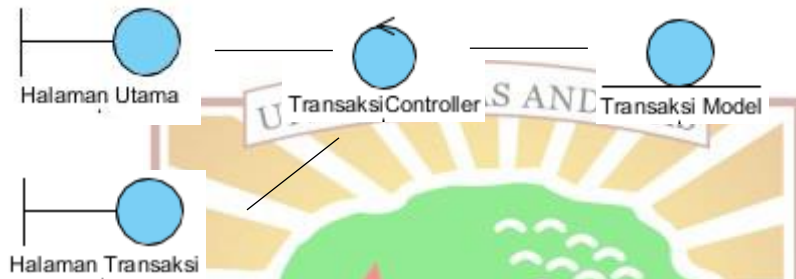
8. *Class Analysis* Lihat Service



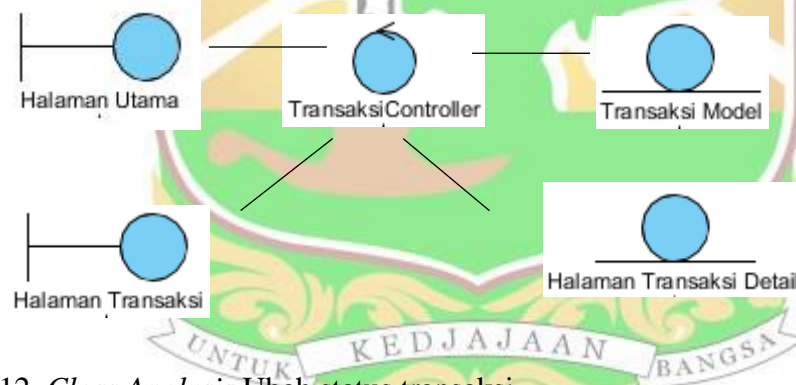
9. *Class Analysis* Lihat Transaksi



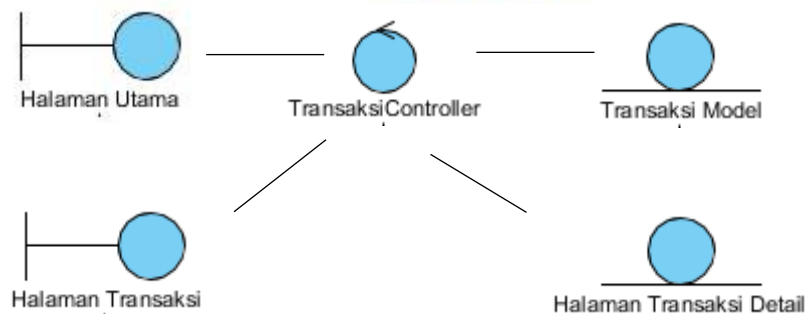
10. *Class Analysis* Hapus Transaksi



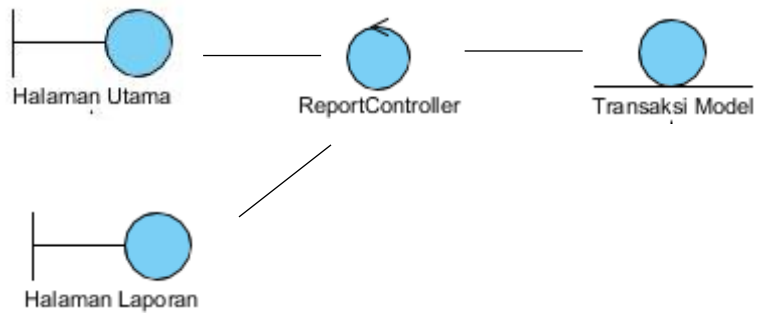
11. *Class Analysis* Lihat Detail Transaksi



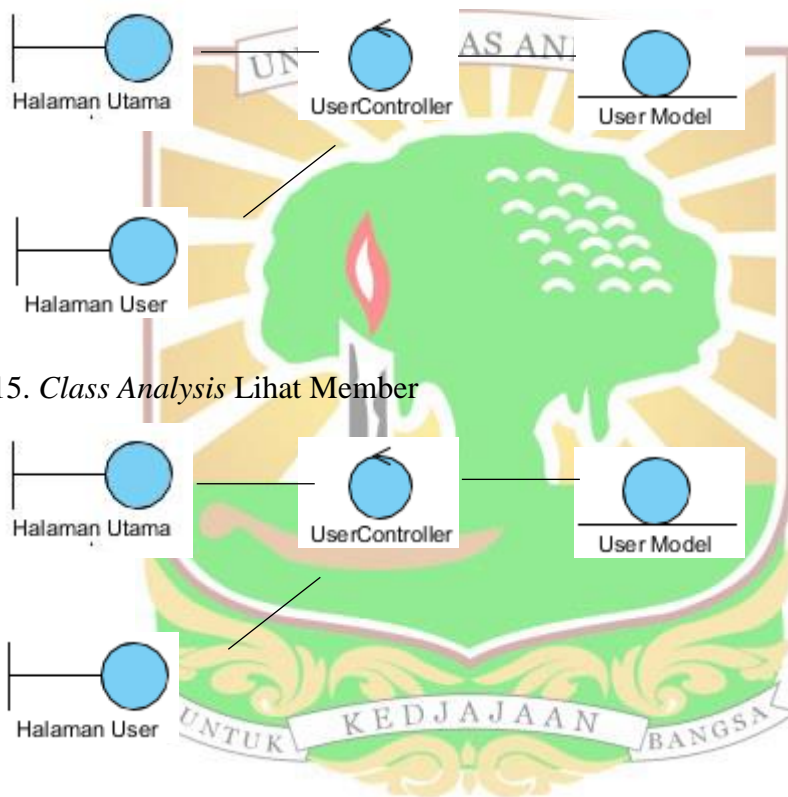
12. *Class Analysis* Ubah status transaksi



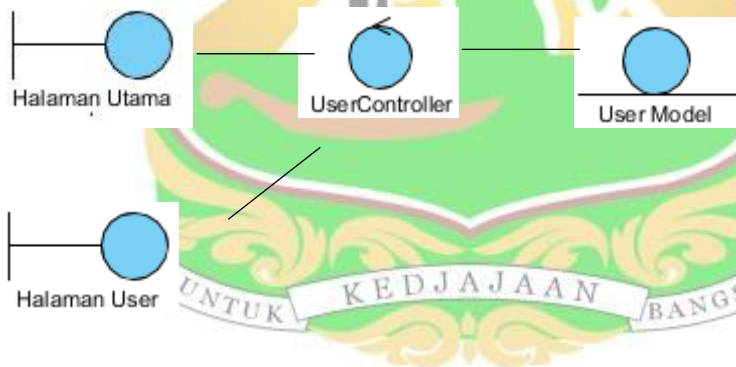
13. *Class Analysis* Lihat Laporan



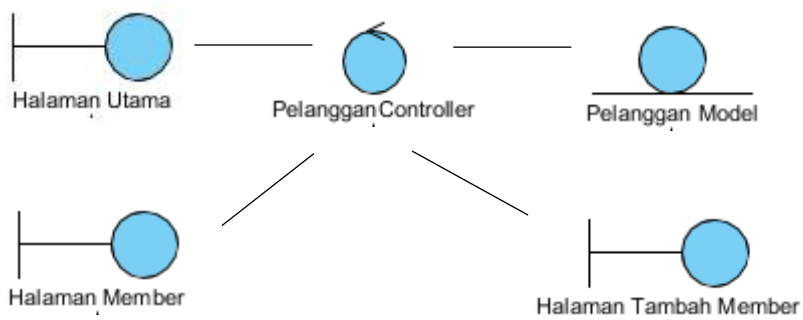
14. *Class Analysis* Ubah Status Member



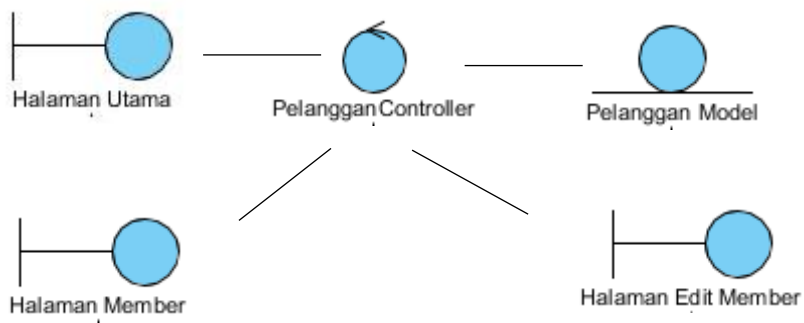
15. *Class Analysis* Lihat Member



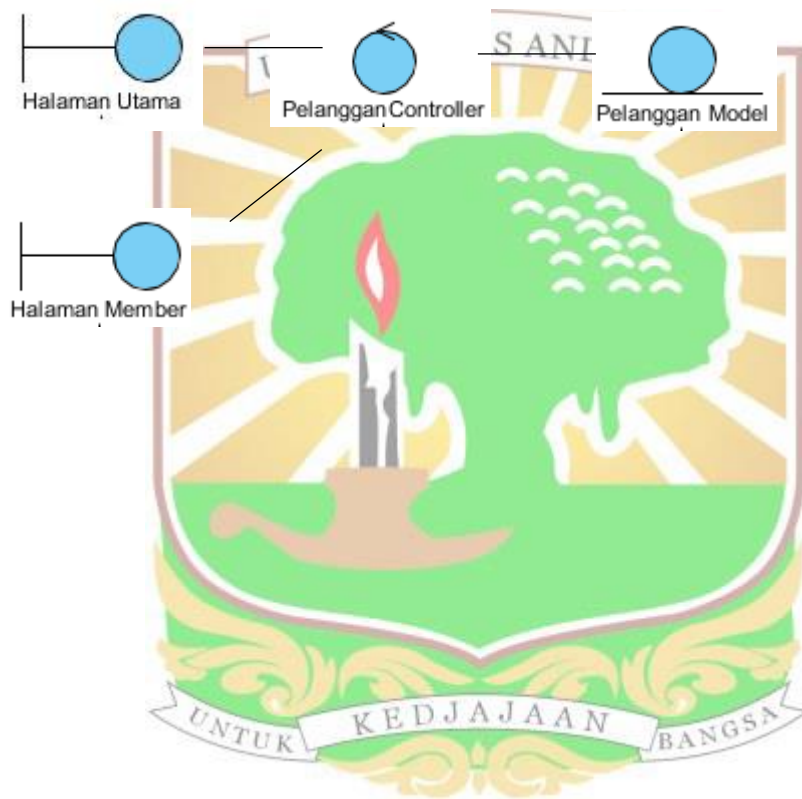
16. *Class Analysis* Tambah Member



17. *Class Analysis* Edit Member



18. *Class Analysis* Hapus Data Member





LAMPIRAN E

Struktur Tabel dan Basis Data Aplikasi

1. Tabel User

Nama atribut	Tipe data	Keterangan
Id_user	varchar (20)	PK
Name	varchar (50)	
email	varchar (50)	Unique
password	varchar (75)	
Status	varchar (1)	
created_at	Timestamps	
updated_at	Timestamps	
role	varchar (1)	
remember_token	varchar (100)	

2. Tabel Pelanggan

Nama atribut	Tipe data	Keterangan
Id_pelanggan	varchar (20)	PK
User_id	varchar (10)	FK
nama	varchar (50)	
alamat	varchar (100)	
telepon	Integer (12)	
reg_data	date	
Tanggal_lahir	date	
created_at	Timestamps	
updated_at	Timestamps	

3. Tabel Kategori

Nama atribut	Tipe data	Keterangan
Id_kategori	varchar (20)	PK
nama_kategori	varchar (50)	FK
created_at	Timestamps	
updated_at	Timestamps	



Perancangan Antar Muka Aplikasi

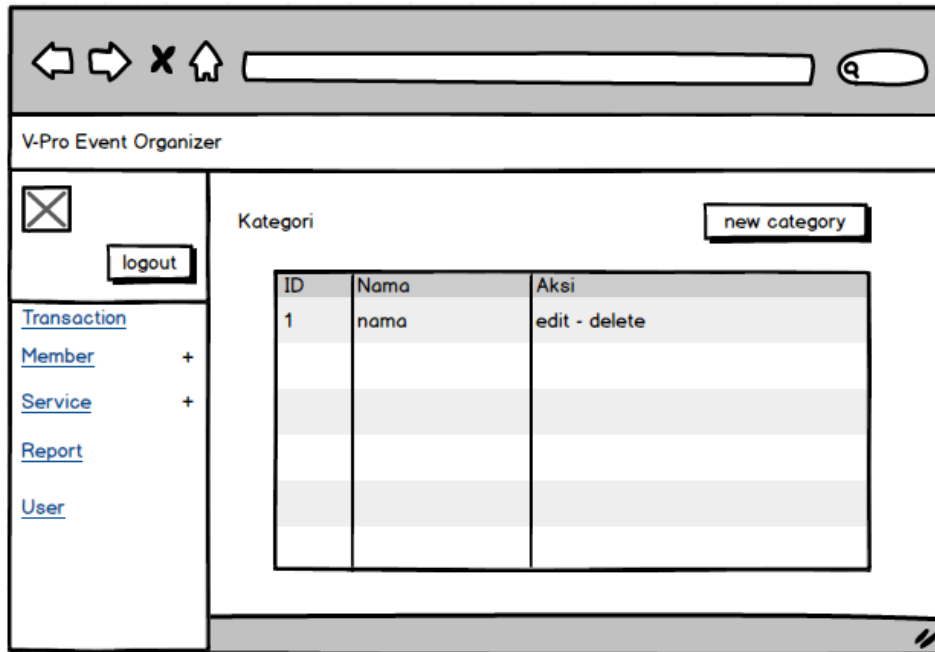
1. Perancangan Antar Muka Halaman Login

The wireframe shows a browser window with a navigation bar containing back, forward, close, and home icons, a search bar, and a search icon. The main content area is titled "V-Pro Event Organizer" and contains a login form. The form has two input fields labeled "Email" and "Password". Below the "Password" field is a radio button labeled "Remember Me" and a "Sign In" button. The browser window has a standard OS-style title bar and a close button in the bottom right corner.

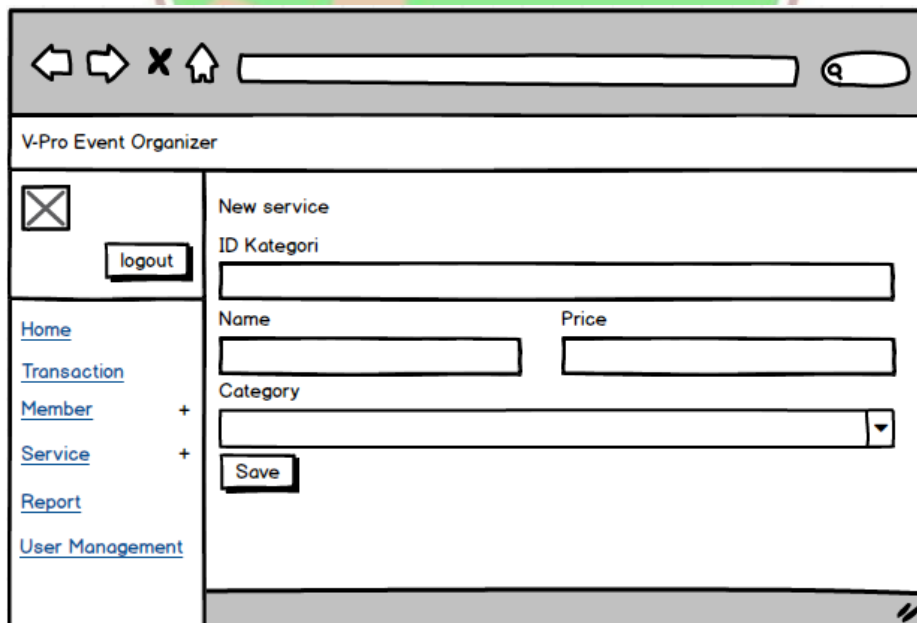
2. Perancangan Antar Muka Halaman Edit Kategori

The wireframe shows a browser window with a navigation bar containing back, forward, close, and home icons, a search bar, and a search icon. The main content area is titled "V-Pro Event Organizer" and contains a sidebar and a main content area. The sidebar has a "logout" button and a list of menu items: "Home", "Transaction", "Member +", "Service +", "Report", and "User Management". The main content area is titled "Edit Kategori" and has a "Name" input field and a "Save" button. The browser window has a standard OS-style title bar and a close button in the bottom right corner.

3. Perancangan Antar Muka Halaman Lihat Kategori



4. Perancangan Antar Muka Halaman Tambah Service



5. Perancangan Antar Muka Halaman Edit Service

V-Pro Event Organizer

logout

Home

Transaction

Member +

Service +

Report

User Management

Edit Service

Name

Price

Category

Save

6. Perancangan Antar Muka Halaman Lihat Service

V-Pro Event Organizer

logout

Transaction

Member +

Service +

Report

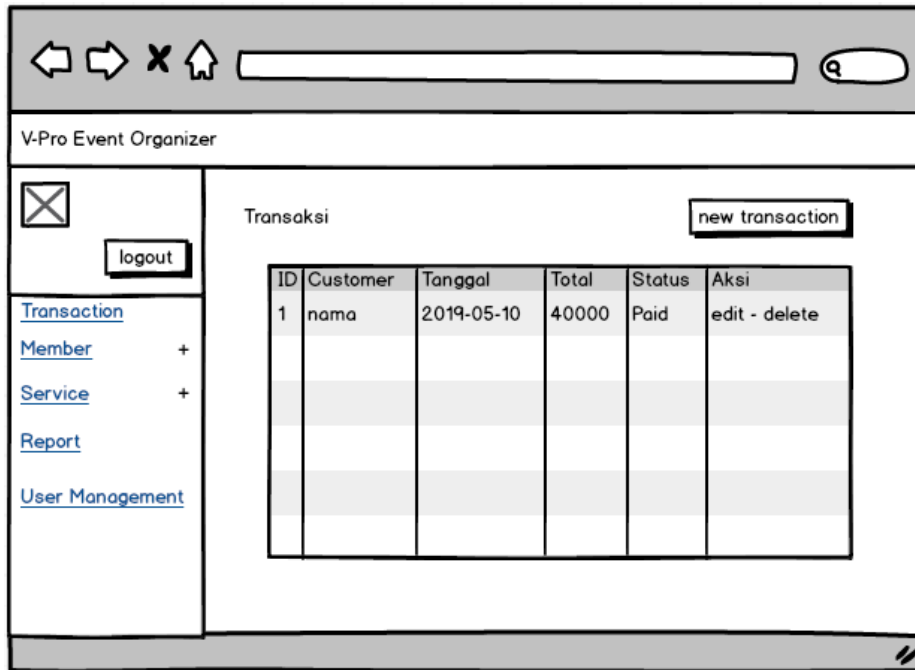
User

Service

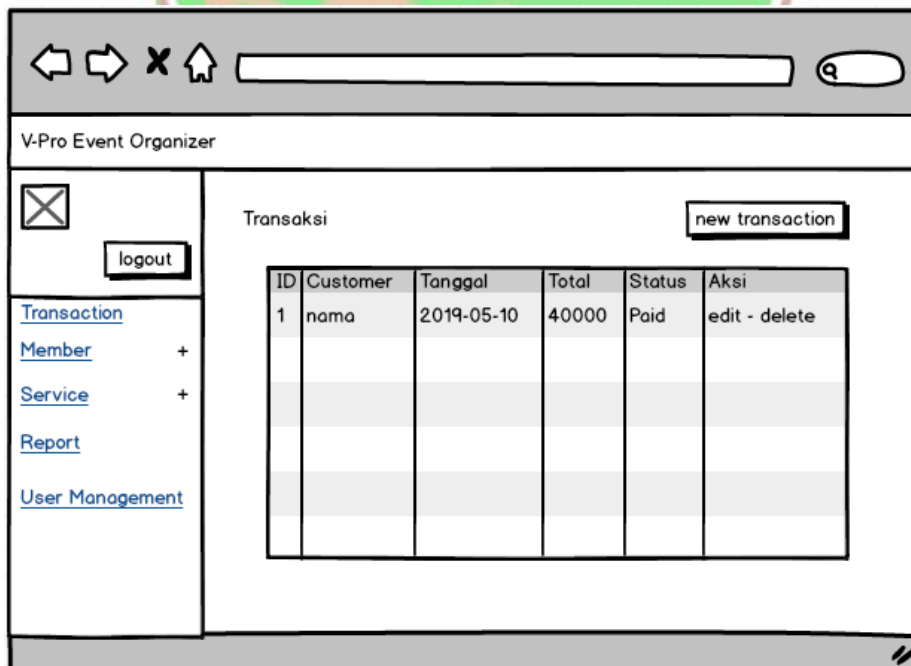
new service

ID	Nama	Harga	Kategori	Aksi
1	nama	Harga	Kategori	edit - delete

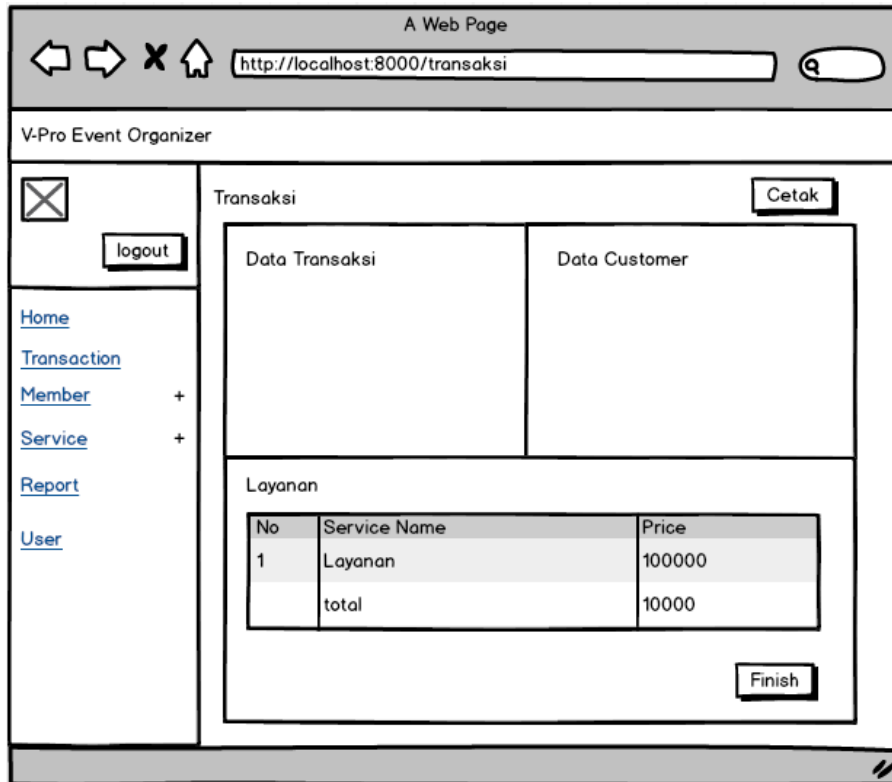
7. Perancangan Antar Muka Halaman Lihat Transaksi



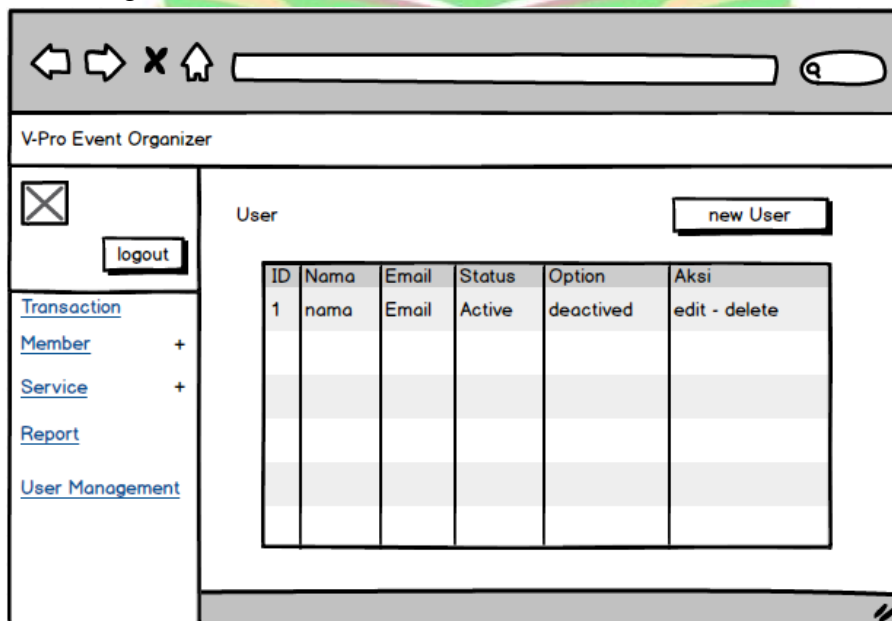
8. Perancangan Antar Muka Halaman Lihat Transaksi



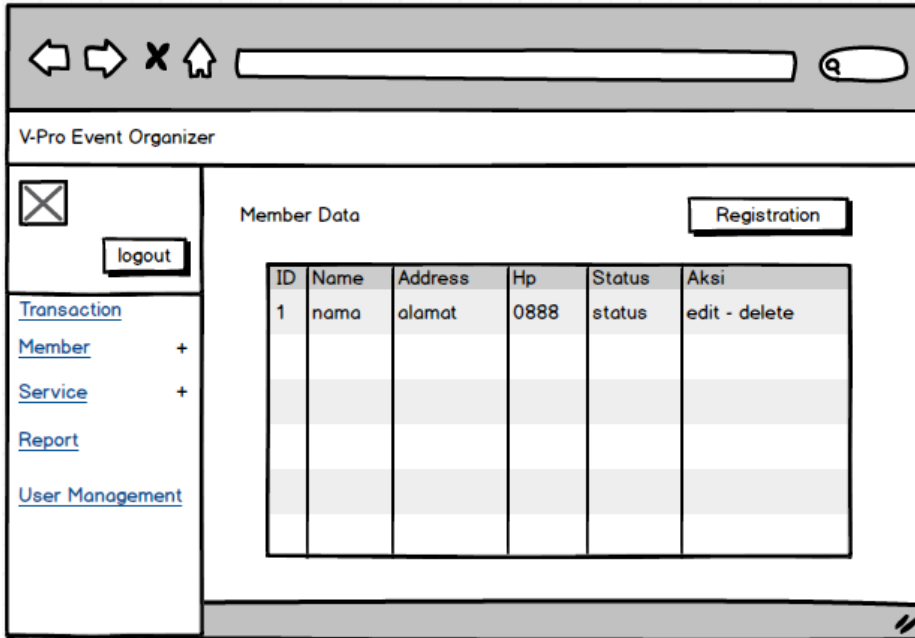
9. Perancangan Antar Muka Halaman Lihat Detail Transaksi



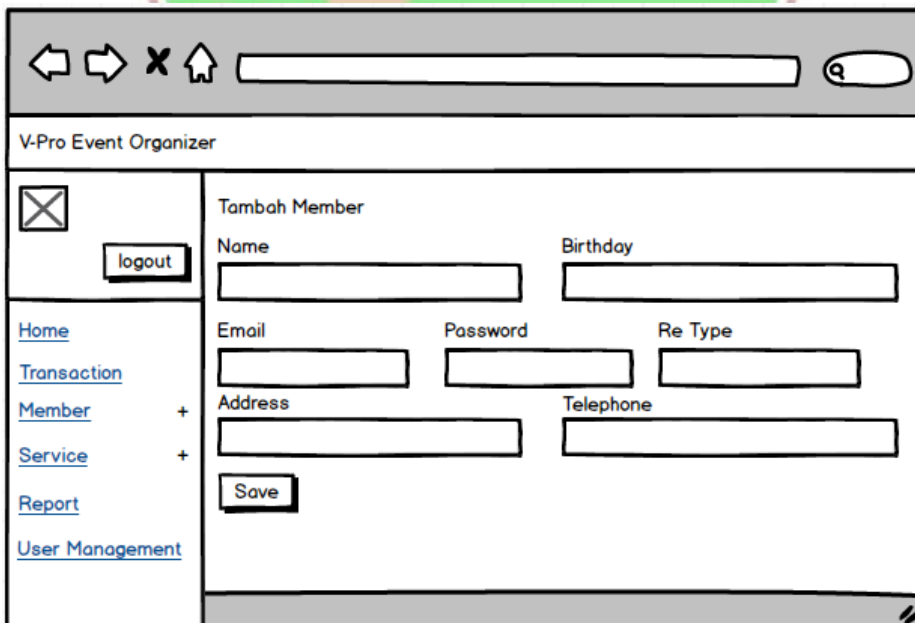
10. Perancangan Antar Muka Halaman Ubah Status Member



11. Perancangan Antar Muka Halaman Lihat Member



12. Perancangan Antar Muka Halaman Tambah Member



13. Perancangan Antar Muka Halaman Edit Member

The screenshot shows a web browser window with the following elements:

- Browser Address Bar:** Contains navigation icons (back, forward, stop, home) and a search icon.
- Page Title:** "V-Pro Event Organizer"
- Page Content:**
 - Header:** "Edit Member" with a close button (X).
 - Form Fields:** "Email", "Password", "Re Type", "Address", and "Telephone".
 - Buttons:** "logout" and "Save".
- Left Sidebar:** A menu with the following items:
 - [Home](#)
 - [Transaction](#)
 - [Member](#) +
 - [Service](#) +
 - [Report](#)
 - [User Management](#)





LAMPIRAN G

Kode Program Aplikasi Web

1. Controller

a. HomeController

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;

use App\Pelanggan;
use App\Transaksi;
use App\Kategori;
use App\Service;

class HomeController extends Controller
{
    /**
     * Create a new controller instance.
     *
     * @return void
     */
    // public function __construct()
    // {
    //     $this->middleware('auth');
    // }

    /**
     * Show the application dashboard.
     *
     * @return \Illuminate\Http\Response
     */
    public function index()
    {
        $data['pelanggan'] = count(Pelanggan::where('user_id', '!=',
1)->get());
        $data['transaksi'] = count(Transaksi::all());
        $data['service'] = count(Service::all());
        $data['kategori'] = count(Kategori::all());
        return view('pages.home', $data);
    }
}
```

b. KategoriController

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Kategori;
use App\Service;
use Auth;

class KategoriController extends Controller
{
    public function index ()
    {
        $data['kategori'] = Kategori::all();
        $data['save'] = '';
        return view('pages.kategori.index', $data);
    }

    public function destroy ($id)
    {
        $service = Service::where('kategori_id', $id)->get();

        if (count($service) > 0)
        {
            return redirect('/kategori');
        }
        else
        {
            Kategori::where('id_kategori', $id)->delete();
            return redirect(url('/kategori'));
        }
    }

    public function show ()
    {
        $id = request('id');
        $data['kategori'] = Kategori::find($id);
        return view('pages.kategori.edit', $data);
    }

    public function update (Request $request)
    {
        $id = $request->input('id');
        $name = $request->input('kategori_name');

        $array_to_input = [
            'nama_kategori' => $name,
        ];

        $update_kategori = Kategori::where('id_kategori', $id)-
>update($array_to_input);
        if ($update_kategori == 1) {
            return redirect(url('kategori/show?id='.$id));
        }
    }
}
```

```

public function new()
{
    return view('pages.kategori.new');
}

public function input(Request $request)
{
    $validatedData = $request->validate([
        'id_kategori' => 'required|unique:kategoris|max:20',
        'kategori_name' => 'required',
    ]);

    $id_kategori = $request->input('id_kategori');
    $name = $request->input('kategori_name');

    $array_to_input = [
        'nama_kategori' => $name,
        'id_kategori' => $id_kategori,
    ];

    $new_kategori = Kategori::InsertGetId($array_to_input);

    $data['kategori'] = Kategori::all();
    $data['save'] = 'success';
    return view('pages.kategori.index', $data);
}

public function get_all ()
{
    return Kategori::all();
}
}

```

c. PelangganController

```

<?php
namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Pelanggan;
use App\User;
use App\Transaksi;
use Validator;
use Auth;

class PelangganController extends Controller
{
    public function index()
    {
        $data['pelanggan'] = Pelanggan::where('user_id', '!=', 1)->get();
        $data['save'] = '';
        return view('pages.pelanggan.index', $data);
    }
}

```

```

public function create()
{
    $data['save'] = '';
    return view('pages.pelanggan.create', $data);
}

public function store(Request $request)
{
    $request->validate([
        "nama" => "required",
        "telepon" => "required",
        "email" => "required|email|unique:users,email",
        "password" => "required",
        "birth" => "required"
    ]);

    $tanggal = date('Y-m-d');
    $arrTanggal= explode('-', $tanggal);
    $kode = 'U-'. $arrTanggal[0]. $arrTanggal[1]. $arrTanggal[2];
    $cari_kode = User::where('id_user', 'like', $kode.'%')-
>orderBy('id_user', 'desc')->first();
    if(empty($cari_kode)){
        $new_no = 1;
    } else {
        $arrNo = explode($kode, $cari_kode->id_user);
        $new_no = $arrNo[1] + 1;
    }
    $new_kode = $kode.$new_no;

    $user = User::insert([
        "id_user" => $new_kode,
        "name" => $request->nama,
        "email" => $request->email,
        "password" => bcrypt($request->password),
        "role" => 3,
        "status" => 0,
        "created_at" => now()
    ]);

    $pelanggan = new Pelanggan();
    $pelanggan->user_id = $new_kode;
    $pelanggan->id_pelanggan = $new_kode;
    $pelanggan->nama = $request->nama;
    $pelanggan->alamat = $request->alamat;
    $pelanggan->telepon = $request->telepon;
    $pelanggan->tanggal_lahir = $request->birth;
    $pelanggan->reg_date = now()->toDateString();
    $pelanggan->save();

    $data['pelangans'] = Pelanggan::where('user_id', '!=', 1)-
>get();
    $data['save'] = 'success';
    return view('pages.pelanggan.index', $data);
}

public function register(Request $request)

```



```

{
    $request->validate([
        "nama" => "required",
        "telephone" => "required",
        "email" => "required|email|unique:users,email",
        "password" => "required",
        // "birth" => "required"
    ]);

    $tanggal = date('Y-m-d');
    $sarrTanggal= explode('-', $tanggal);
    $kode = 'U-'. $sarrTanggal[0]. $sarrTanggal[1]. $sarrTanggal[2];
    $cari_kode = User::where('id_user', 'like', $kode.'%')-
>orderBy('id_user', 'desc')->first();
    if(empty($cari_kode)){
        $new_no = 1;
    } else {
        $sarrNo = explode($kode, $cari_kode->id_user);
        $new_no = $sarrNo[1] + 1;
    }
    $new_kode = $kode.$new_no;

    $user = User::insert([
        "id_user" => $new_kode,
        "name" => $request->nama,
        "email" => $request->email,
        "password" => bcrypt($request->password),
        "role" => 3,
        "status" => 0,
        "created_at" => now()
    ]);

    $pelanggan = new Pelanggan();
    $pelanggan->user_id = $new_kode;
    $pelanggan->id_pelanggan = $new_kode;
    $pelanggan->nama = $request->nama;
    $pelanggan->alamat = $request->address;
    $pelanggan->telepon = $request->telephone;
    // $pelanggan->tanggal_lahir = $request->birth;
    $pelanggan->reg_date = now()->toDateString();
    $pelanggan->save();

    $search_user = User::find($new_kode);
    Auth::login($search_user);
    return redirect('transaksi');

    // $data['pelanggan'] = Pelanggan::where('user_id', '!=', 1)-
>get();
    // $data['save'] = 'success';
    // return redirect('/login');
}

public function edit ($id)
{
    $data['pelanggan'] = Pelanggan::with('user')-
>where('id_pelanggan', $id)->first();
    $data['save'] = '';
}

```

```

        return view('pages.pelanggan.edit', $data);
    }

    public function update (Request $request)
    {
        $pelanggan_id = $request->pelanggan_id;

        $request->validate([
            "nama" => "required",
            "telepon" => "required"
        ]);

        pelanggan::where('id_pelanggan', $pelanggan_id)->update([
            "nama" => $request->nama,
            "alamat" => $request->alamat,
            "telepon" => $request->telepon,
            "tanggal_lahir" => $request->birth
        ]);

        $data['pelanggan'] = Pelanggan::with('user')->where('id_pelanggan', $pelanggan_id)->first();
        $data['save'] = 'success';
        return view('pages.pelanggan.edit', $data);
    }

    public function delete($id){
        $check_trans = Transaksi::where('pelanggan_id', $id)->first();

        if ($check_trans == null) {
            $user = pelanggan::where('id_pelanggan', $id)->first();
            Pelanggan::where('id_pelanggan', $id)->delete();
            User::where('id_user', $user->user_id)->delete();
            $data['message'] = 0;
        }
        else{
            $data['message'] = 1;
        }
        return $data;
    }
}

```

d. ServiceController

```

<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\Service;
use App\DetailService;
use App\Kategori;

use Auth;

class ServiceController extends Controller
{

```

```

public function index ()
{
    $data['service'] = Service::all();
    $data['save'] = '';
    return view('pages.service.index', $data);
}

public function destroy ($id)
{
    $detail_service = DetailService::where('service_id', $id)-
>get();

    if (count($detail_service) > 0)
    {
        return redirect('/service');
    }
    else
    {
        Service::where('id_service', $id)->delete();
        return redirect(url('/service'));
    }
}

public function show ()
{
    $id = request('id');
    $data['service'] = Service::find($id);
    $data['kategori'] = Kategori::all();

    return view('pages.service.edit', $data);
}

public function update (Request $request)
{
    $id = $request->input('id');
    $name = $request->input('service_name');
    $tarif_normal = $request->input('tarif_normal');
    $kategori_id = $request->input('kategori_id');
    $array_to_input = [
        'nama_services' => $name,
        'kategori_id' => $kategori_id,
        'tarif_normal' => $tarif_normal,
    ];

    $update_service = Service::where('id_service', $id)-
>update($array_to_input);
    if ($update_service == 1) {
        return redirect(url('service/show?id='.$id));
    }
}

public function new()
{
    $data['kategori'] = Kategori::all();
    return view('pages.service.new', $data);
}

```

```

public function input(Request $request)
{
    $validatedData = $request->validate([
        'id_service' => 'required|unique:services|max:20',
        'service_name' => 'required',
        'tarif_normal' => 'required|numeric',
        'kategori_id' => 'required',
    ]);

    $id_service = $request->input('id_service');
    $name = $request->input('service_name');
    $tarif_normal = $request->input('tarif_normal');
    $kategori_id = $request->input('kategori_id');

    $array_to_input = [
        'id_service' => $id_service,
        'nama_services' => $name,
        'tarif_normal' => $tarif_normal,
        'kategori_id' => $kategori_id,
    ];

    $new_service = Service::InsertGetId($array_to_input);

    $data['service'] = Service::all();
    $data['save'] = 'success';
    return view('pages.service.index', $data);
}

public function get_all ()
{
    return Service::all();
}

public function get_kategori($kategori_id)
{
    return Service::where('kategori_id',$kategori_id)->get();
}

public function get_first()
{
    $kategori = Kategori::first();
    return Service::where('kategori_id',$kategori->id_kategori)->get();
}

public function get_id($id)
{
    return Service::find($id);
}
}

```

e. UserController

```
<?php

namespace App\Http\Controllers;

use Illuminate\Http\Request;
use App\User;
use Spatie\Permission\Models\Role;
use App\Transaksi;
use App\Pelanggan;

class UserController extends Controller
{
    public function index()
    {
        $data['users'] = User::where('role', '!=', 0)->get();
        $data['save'] = 'success';
        return view('pages.user.index', $data);
    }

    public function create()
    {
        $roles = Role::all()->pluck('name', 'id');
        return view('pages.user.create', compact('roles'));
    }

    public function store(Request $request)
    {
        $user = new User();
        $user->name = $request->name;
        $user->email = $request->email;
        $user->role = $request->role;
        $user->password = bcrypt($request->password);
        $user->status = 0;
        $user->save();

        $data['users'] = User::where('role', '!=', 0)->get();
        $data['save'] = 'success';
        return view('pages.user.index', $data);
    }

    public function edit($id)
    {
        $user = User::find($id);
        $roles = Role::all()->pluck('name', 'id');
        return view('pages.user.edit', compact('user', 'roles'));
    }

    public function update(Request $request, $id)
    {
        $user = User::find($id)->update(['name'=>$request->name,
        'email'=>$request->email, 'role'=>$request->role,
        'password'=>bcrypt($request->password)]);
    }
}
```

```

        return redirect(route('user.index'));
    }

    public function activated_account ($id, $status)
    {
        if ($status == 1) $user_status = User::where('id_user', $id)-
>update(['status' => 1]);
        else $user_status = User::where('id_user', $id)-
>update(['status' => 0]);

        if ($user_status == 1) {
            return redirect(route('user.index'));
        }
    }

    public function destroy ($id)
    {
        $pelanggan = count(Pelanggan::where('id_pelanggan', $id)-
>get());
        $transaksi = count(Transaksi::where('user_id', $id)->get());

        if ($pelanggan > 0 || $transaksi > 0) return
redirect(route('user.index'));
        else
        {
            User::where('id_user', $id)->delete();
            return redirect(route('user.index'));
        }
    }

    public function show ($id)
    {
        $user = User::with('roles')->where('id_user', $id)->first();
        return view('pages.user.show', compact('user'));
    }
}

```

2. Model

a. Detail Service

```

<?php

namespace App;

use Illuminate\Database\Eloquent\Model;

class DetailService extends Model
{
    protected $table = 'detail_services';
    protected $primaryKey = 'id_detail_service';
    protected $fillable = [
        'transaksi_id', 'service_id', 'harga',
    ];
}

```

```
public function Service()
{
    return $this->belongsTo('App\Service','service_id');
}

public function Transaksi()
{
    return $this->belongsTo('App\Transaksi','transaksi_id');
}
}
```

b. Kategori

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;

class Kategori extends Model
{
    protected $table = 'kategoris';
    protected $primaryKey = 'id_kategori';
    public $incrementing = false;

    protected $fillable = [
        'id_kategori', 'nama_kategori',
    ];
}
```

c. Pelanggan

```
<?php
namespace App;
use Illuminate\Database\Eloquent\Model;

class Service extends Model
{
    protected $table = 'services';
    protected $primaryKey = 'id_service';
    public $incrementing = false;

    protected $fillable = [
        'id_service', 'nama_services', 'tarif_normal', 'kategori_id'
    ];

    public function Kategori()
    {
        return $this->belongsTo('App\Kategori','kategori_id');
    }
}
```

d. User

```
<?php
namespace App;

use Illuminate\Notifications\Notifiable;
use Illuminate\Foundation\Auth\User as Authenticatable;
use Spatie\Permission\Traits\HasRoles;

class User extends Authenticatable
{
    use Notifiable;
    use HasRoles;

    protected $primaryKey = 'id_user';
    public $incrementing = false;

    /**
     * The attributes that are mass assignable.
     *
     * @var array
     */
    protected $fillable = [
        'id_user', 'name', 'email', 'password', 'role', 'status'
    ];

    /**
     * The attributes that should be hidden for arrays.
     *
     * @var array
     */
    protected $hidden = [
        'password', 'remember_token',
    ];

    public function roles ()
    {
        return $this->hasOne('App\Role', 'id', 'role');
    }

    public function pelanggan()
    {
        return $this->hasOne('App\Pelanggan', 'user_id');
    }
}
```

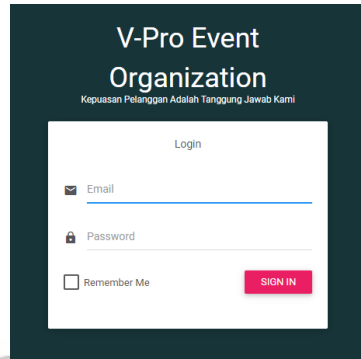




LAMPIRAN H

Implementasi Antar Muka Aplikasi

1. Halaman Login



V-Pro Event
Organization

Kepuasan Pelanggan Adalah Tanggung Jawab Kami

Login

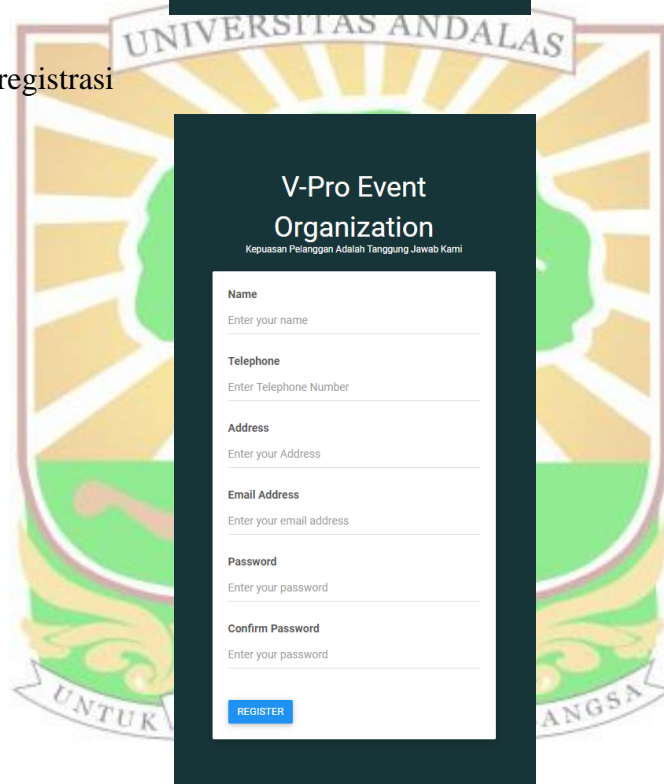
Email

Password

Remember Me

SIGN IN

2. Halaman registrasi



V-Pro Event
Organization

Kepuasan Pelanggan Adalah Tanggung Jawab Kami

Name

Enter your name

Telephone

Enter Telephone Number

Address

Enter your Address

Email Address

Enter your email address

Password

Enter your password

Confirm Password

Enter your password

REGISTER

3. Halaman Tambah Kategori

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

© V-PRO Event Organizer.

New Kategori

ID Kategori

Name

Save

4. Halaman Edit Kategori

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

© V-PRO Event Organizer.

Kategori Detail

Name

Multimedia

Save

5. Halaman Lihat Kategori

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

Kategori Data

+ New Kategori

Show 10 entries

Search:

No	ID Kategori	Name	Action
1	1	Multimedia	
2	2	Sound System	
3	4	Production	

Showing 1 to 3 of 3 entries

Previous 1 Next

© V-PRO Event Organizer.

6. Halaman Tambah Service

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

New Service

ID

Name Price

Category

Multimedia

Save

© V-PRO Event Organizer.

7. Halaman Edit Service

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

Service Detail

Name Price

A 1000000

Category

Multimedia

Save

© V-PRO Event Organizer.

8. Halaman Lihat Service

Service Data

+ New Service

Show 10 entries Search: pang

No	ID	Name	Price	Category	Action
5	5	panggung riging 8 * 6	2000000	Production	
6	6	panggung riging 4 * 6	1000000	Production	

Showing 1 to 2 of 2 entries (filtered from 6 total entries) Previous 1 Next

© V-PRO Event Organizer.

9. Halaman Lihat Transaksi

Transaction Data

+ New Transaction

Show 10 entries Search:

No	ID Transaksi	Customer	Tanggal	Total	Status	Aksi
1	F-201906221	Pengujian	2019-06-22	1,500,000	Booking	
2	F-201906081	user	2019-06-08	3,500,000	Paid	
3	F-201906082	5	2019-06-08	3,500,000	Booking	
4	F-201906041	rendy	2019-06-04	2,500,000	Paid	
5	7	qwerty	2019-05-20	1,000,000	Booking	
6	2	guest	2019-05-17	1,000,000	Paid	

© V-PRO Event Organizer.

10. Halaman Ubah Status Member

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

DATA USER

+ New User

Show 10 entries Search:

No	Name	Email	Status	Option	Action
1	Admin	admin@admin.com	Active	Deactivated	
2	5	5@5.com	Active	Deactivated	
3	6	6@6.com	Active	Deactivated	
4	7	7@7.com	Active	Deactivated	
5	a	a@1.com	Active	Deactivated	
6	a	a@12.com	Active	Deactivated	

© V-PRO Event Organizer.

11. Halaman Lihat Member

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

Member Data

+ Registration

Show 10 entries Search:

No	ID	Name	Address	Hp	Status	Aksi
15	U-201906221	Pengujian	Testing	0888	Member	
14	U-201906191	1	1	1	Member	
13	9	qwerty	qwerty	123123	Member	
12	8	guest again	1	1	Member	
11	6	Guest New	new	1234123	Member	
10	5	guest	Padang	129129	Member	

© V-PRO Event Organizer.

12. Halaman Tambah Member

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

© V-PRO Event Organizer.

Coustumer Registration

Name * Birthdate *
hh/bb/yyyy

Email * Password * Retype Password *

Address Telepon *

SAVE

13. Halaman Edit Member

V-PRO Event Organizer

Admin

Menu

- Home
- Transaction
- Member
- Service Management
- Report
- User management

© V-PRO Event Organizer.

Coustumer Registration

Name * Birthdate *
S 05/06/2019

Alamat Telepon *
S S

SAVE





LAMPIRAN I

Form Pengujian Sistem

Nama Aplikasi : Pembangunan Sistem *Booking Online* dan *Event Organizer* Berbasis *Web* Pada CV. V Pro Kota Padang

Nama Penguji :
Jabatan :

Pengujian untuk Fungsional *User Admin*

No.	Nama Fungsional	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
1.	Melakukan Login	Memilih menu <i>login</i> pada halaman <i>home</i>	Sistem menampilkan halaman <i>login</i> .	Sesuai / Tidak-Sesuai
2.	Mengelola Kategori	Menekan halaman Kategori dan menekan tombol edit	Sistem akan menampilkan halaman kategori	Sesuai / Tidak-Sesuai
	a. Edit Kategori	Menekan halaman kategori dan memilih tombol hapus.	Sistem akan menghapus kategori yang dipilih	Sesuai / Tidak-Sesuai
	b. Hapus Kategori	Menekan halaman kategori	Sistem akan menampilkan data pada halaman kategori	Sesuai / Tidak-Sesuai
	c. Lihat Kategori			
3.	Mengelola Service			
	a. Tambah Service	Memilih menu <i>service</i> dan mengisi form data <i>service</i>	Sistem akan menampilkan halaman <i>service</i>	Sesuai / Tidak-Sesuai
	b. Edit Service	Memilih menu <i>service</i> dan menekan tombol edit	Sistem akan menampilkan halaman <i>service</i>	Sesuai / Tidak-Sesuai
	c. Hapus Service	Memilih menu <i>service</i> dan menekan tombol hapus	Sistem akan menghapus <i>service</i> yang telah dipilih	Sesuai / Tidak-Sesuai
	d. Lihat Service	Menekan halaman <i>service</i>	Sistem akan menampilkan data pada halaman <i>service</i>	Sesuai / Tidak-Sesuai
4.	Mengelola Transaksi			
	a. Lihat Transaksi	Memilih menu <i>transaksi</i> pada halaman <i>home</i>	Sistem akan menampilkan data transaksi	Sesuai / Tidak-Sesuai

b. Hapus Transaksi	Memilih menu transaksi dan menekan tombol hapus	Sistem akan menghapus transaksi yang telah dipilih	Sesuai / Tidak-Sesuai-
c. Tambah Transaksi	Memilih menu <i>new transaksi</i> pada halaman transaksi	Sistem menampilkan form transaksi yang akan diisi	Sesuai / Tidak-Sesuai-
d. Lihat Detail Transaksi	Menekan tombol <i>show</i> yang ada pada menu transaksi	Sistem menampilkan semua <i>detail</i> transaksi	Sesuai / Tidak-Sesuai-
e. Ubah Status Transaksi	Memilih menu <i>show</i> yang ada pada menu transaksi lalu menekan tombol <i>finish</i>	Sistem akan mengubah status transaksi	Sesuai / Tidak-Sesuai-
5. Lihat Laporan	Admin membuka halaman <i>report</i>	Sistem menampilkan halaman laporan	Sesuai / Tidak-Sesuai-
6. Mengelola Member			
a. Ubah Status Member	Memilih halaman <i>user management</i> dan menekan tombol <i>option</i>	Sistem mengubah status member	Sesuai / Tidak-Sesuai-
b. Lihat Member	Memilih menu <i>user management</i>	Sistem menampilkan halaman lihat member	Sesuai / Tidak-Sesuai-
c. Tambah Member	Memilih halaman <i>member</i> dan menekan tombol <i>registration</i>	Sistem menampilkan halaman tambah member	Sesuai / Tidak-Sesuai-
d. Edit Member	Memilih halaman <i>member</i> dan menekan tombol edit	Sistem menampilkan halaman edit member	Sesuai / Tidak-Sesuai-
e. Hapus Data Member	Memilih halaman <i>member</i> dan menekan tombol hapus	Sistem menghapus data member	Sesuai / Tidak-Sesuai-
7. Cetak Laporan	Memilih menu transaksi dan menekan tombol cetak	Sistem menampilkan halaman cetak	Sesuai / Tidak-Sesuai-

Padang, 08 Juli 2019

Penguji

PRO
****E****

Form Pengujian Sistem

Nama Aplikasi : Pembangunan Sistem Booking Online dan Penjualan Jasa Event Organizer Berbasis Web Pada CV. V.Pro Kota Padang

Nama Penguji :
Jabatan :

Pengujian untuk Fungsional User Customer

No.	Nama Fungsional	Skenario Pengujian	Hasil yang Diharapkan	Hasil Pengujian
1.	Melakukan Login	Memilih menu <i>login</i> pada halaman <i>home</i>	Sistem menampilkan halaman <i>login</i> .	Sesuai / Tidak-Sesuai
2.	Melakukan Registrasi	Memilih menu <i>sign up</i> pada halaman <i>home</i>	Sistem menampilkan halaman <i>registrasi</i>	Sesuai / Tidak-Sesuai
3.	Lihat Kategori	Menekan halaman kategori	Sistem akan menampilkan data pada halaman kategori	Sesuai / Tidak-Sesuai
4.	Lihat Service	Menekan halaman service	Sistem akan menampilkan data pada halaman service	Sesuai / Tidak-Sesuai
5.	Mengelola transaksi			
	a. Lihat Transaksi	Memilih menu transaksi pada halaman <i>home</i>	Sistem akan menampilkan data transaksi	Sesuai / Tidak-Sesuai
	b. Hapus Transaksi	Memilih menu transaksi dan menekan tombol hapus	Sistem akan menghapus transaksi yang telah dipilih	Sesuai / Tidak-Sesuai
	c. Tambah Transaksi	Memilih menu <i>new transaksi</i> pada halaman transaksi	Sistem menampilkan form transaksi yang akan diisi	Sesuai / Tidak-Sesuai
	d. Lihat Detai Transaksi	Menekan tombol <i>show</i> yang ada pada menu transaksi	Sistem menampilkan semua <i>detail</i> transaksi	Sesuai / Tidak-Sesuai

Padang, 09 Mei 2019

Pengujian
V-PRO
**** Event ****

Nama Barang Event Organizer Pada CV. V-Pro Kota Padang

A. Multimedia

1. - Projector 1000	: Rp. 3.000.000
- Projector 3000	: Rp. 5.000.000
- Projector 6000	: Rp. 8.000.000
- Projector 10.000	: Rp. 10.000.000
2. - Screen Tembah Belakang 2*3	: Rp. 2.000.000
- Screen Tembah Belakang 3*4	: Rp. 3.000.000
- Screen Tembah Belakang 4*6	: Rp. 4.000.000
3. - Sceen Tembak Depan 2*3	: Rp. 2.000.000
- Sceen Tembak Depan 3*4	: Rp. 3.000.000
- Sceen Tembak Depan 4*6	: Rp. 4.000.000
4. - Video Mixer Channel 4	: Rp. 2.500.000
- Video Mixer Channel 8	: Rp. 3.000.000
- Video Mixer Channel 16	: Rp. 4.500.000
5. LightCam 5 buah	: Rp. 1.500.000
6. 6. LED Screen 5 buah	: Rp. 1.500.000

B. Man Power

1. TimeShow	: Rp. 2.000.000
2. TimeEvent	: Rp. 1.000.000
3. SPG	: Rp. 1.000.000

C. Makanan

1. Minang Food	: Rp. 4.000.000
2. Indonesian Food	: Rp. 5.000.000
3. Chinese Food	: Rp. 8.000.000

D. Production/Hand Production

1. - Panggung Ringing 4*6	: Rp. 2.500.000
- Panggung Ringing 6*8	: Rp. 4.500.000
- Panggung Ringing 8*10	: Rp. 5.500.000
- Panggung Ringing 8*12	: Rp. 7.000.000
2. Barikade 2 Meter	: Rp. 1.000.000
3. - Tenda Konvensional 3*4	: Rp. 1.000.000
- Tenda Konvensional 4*6	: Rp. 1.500.000

4. - WedBrod Modul	: Rp. 1.000.000
- WedBrod Cetak	: Rp. 1.000.000
5. Partisi R8 (3*3)	: Rp. 1.000.000
6. Mistiven	: Rp. 1.000.000
7. Standing AC	: Rp. 150.000
8. Kursi Plastik 20buah	: Rp. 1.500.000
9. Kursi Futural 20buah	: Rp. 1.500.000
10. Sofa VIP 1 set	: Rp. 1.000.000
11. Meja Biasa 1m * 60	: Rp. 1.000.000
12. Meja VIP	: Rp. 1.200.000
13. Runtable Kaca	: Rp. 250.000
14. Bastol/Kursi bar 5buah	: Rp. 1.500.000
15. - Tenda Krucut 3*3	: Rp. 1.500.000
- Tenda Krucut 5*5	: Rp. 2.500.000
16. Fluring/Lantai Kayu	: Rp. 350.000
E. Sound System and Lighting	
1. - Alat Band Electric	: Rp. 3.000.000
- Alat Band Akustik	: Rp. 3.500.000
2. - Lighting Parlet	: Rp. 200.000
- Lighting Moving Bean	: Rp. 500.000
- Lighting Minibrud	: Rp. 300.000
- Lighting Hologen	: Rp. 150.000
- Lighting Freshnel	: Rp. 200.000
- Lighting Skytread	: Rp. 2.000.000
- Lighting Followspot	: Rp. 1.500.000
F. Talent	
1. - Talent Lokal	: Rp. 2.500.000
- Talent Nasional	: Rp. 5.000.000

Padang, 8 Juli 2019

Pemilik
V-PRO
 ***** EVL *****

Agung Wijaya

Pencatatan Peynewaan yang sedang berjalan

NO.	EVENT LIST VPRO INDONESIA	DATE	STATUS	PIC / REMARKS
1.	LORONG BOONH BOONH KESEHATAN	17 MAR 2019	DONE	BUKOT PPA 23 - DCI
2.	FESTIVAL JANGGA PEMILU 2019 KPU Sumbang	4-19 MEI 2019	DONE PAYMENT	EVIDE PPA 23 - DCI Report & INVOICE
3.	BERGABUNG RAHMATUHAN FAIR 2019 - PKU	5-7 Juli 2019	DONE	
4.	GATHERING PIC BELAJAR II Jember - BBT	30 Juni 2019	DONE	
5.	GATHERING WJETH INDONESIA - PDG	SEPTMBER	FOLLOWUP	
6.	SUPPORT PEDUKSIAS EVENT GSK UNIBUD	6 Juli 2019	DONE	uploading 14 Juli 2019
7.	LORONG BOONH BOONH KESEHATAN - PALAMANU Expo 2019		PITCHING	PMB Samba 8 Juli 2019
8.	LAUNCHING HONDA GENIO - HIN/ATI MOTOR	5-8 Juli 2019	RUNING	
9.	YAMAHA CUSTOMER SURVEY			

Pencatatan pembayaran yang sedang berlangsung

V-PRO ***** Event Organizer
 Jl. Kemayoran Tunggal Hitam, Padang, Sumatera Barat, Indonesia
 Phone : 0751-8972927 | Website : www.vpro-id.com | Mail : vpo.oo@gmail.com

INVOICE

Date : 22 Maret 2019
 Invoice Number : VP-2203/2019
 Quotation Number : VP-05/160119

TO :
 Nama :
 Client : KPU Provinsi Sumatera Barat
 Alamat : Jl. Pramuka Raya No 9, Lolong Belanti,
 Padang Utara, Kota Padang, Sumatera Barat
 Phone : 0751) 446654

NO	ITEM / SERVICE	QTY	PRICE	LINE TOTAL
1	Jasa Even Organizer Kegiatan Lomba Band Pentas Musik Jingle Pemko 2019 di GOR Haji Agus Salim Padang pada tanggal 17 Maret 2019.	1 Event	Rp 58.887.000	Rp 58.887.000
SUB TOTAL				Rp 58.887.000
PPN				Rp 5.888.700
DISC				Rp
TOTAL				Rp 64.775.700

Bank Name : BNI (Bank Negara Indonesia)
 Account Name : VPRO INDONESIA
 Account Number : 0801.9315.04

SPECIAL NOTES & INSTRUCTIONS:

FINANCE : Des Anesia
DIRECTOR : Ageng Wajaya

