



**PENERAPAN *FLOWTHING MODEL* PADA SISTEM *MOBILE VOUCHER*  
*FOOD DELIVERY***

**SKRIPSI**

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk menyelesaikan pendidikan di Program Studi Sistem Informasi Universitas Jember dan mendapat gelar Sarjana Sistem Informasi

**oleh**

**Novado Caesar**

**NIM. 102410101006**

**PROGRAM STUDI SISTEM INFORMASI  
UNIVERSITAS JEMBER**

**2015**



**PENERAPAN *FLOWTHING MODEL* PADA SISTEM *MOBILE VOUCHER*  
*FOOD DELIVERY***

**SKRIPSI**

diajukan guna melengkapi tugas akhir dan memenuhi salah satu syarat untuk menyelesaikan pendidikan di Program Studi Sistem Informasi Universitas Jember dan mendapat gelar Sarjana Sistem Informasi

**oleh**

**Novado Caesar**

**NIM. 102410101006**

**PROGRAM STUDI SISTEM INFORMASI  
UNIVERSITAS JEMBER**

**2015**

## PERSEMBAHAN

Skripsi ini saya persembahkan untuk:

1. Seluruh keluarga saya yang sangat saya cintai. Terimakasih atas dukungan dan doanya.
2. Seluruh tenaga pendidik yang saya hormati. Terimakasih atas pelajaran yang diberikan.
3. Seluruh sahabat dan pendukung yang menemani saya. Terimakasih atas kerjasamanya.
4. Almamater Program Studi Sistem Informasi Universitas Jember.
5. Seluruh kerabat otaku, *gamer*, *cosplayer*, *gemstone lover*, dan mitra bisnis. Terimakasih atas waktu yang diberikan.
6. Ifitah Adi S.KM yang paling saya cintai dan saya banggakan.

**MOTTO**

***“Everything changes and nothing stands still”***

*(Heraclitus of Ephesus. 535 BC – 475 BC)*

***“Kosong adalah isi, isi adalah kosong”***

*(Siddhartha Gautama Buddha)*



**PERNYATAAN**

Saya yang bertanda tangan di bawah ini:

Nama : Novado Caesar

NIM : 102410101006

menyatakan dengan sesungguhnya bahwa karya ilmiah yang berjudul “ Penerapan *Flowthing Model* pada Sistem *Mobile Voucher Food Delivery*” adalah benar-benar hasil karya sendiri, kecuali jika dalam pengutipan substansi disebutkan sumbernya, belum pernah diajukan pada institusi mana pun, dan bukan karya jiplakan. Saya bertanggung jawab atas keabsahan dan kebenaran isinya sesuai dengan sikap ilmiah yang harus dijunjung tinggi.

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

Jember, 2 Desember 2015

Yang menyatakan,

Novado Caesar

NIM. 102410101006

**PENERAPAN *FLOWTHING MODEL* PADA SISTEM *MOBILE VOUCHER*  
*FOOD DELIVERY***

Oleh :

Novado Caesar

102410101006

Pembimbing

Dosen Pembimbing Utama : Anang Andrianto, S.T., M.T.

Dosen Pembimbing Anggota : Nelly Oktavia A, S.Si, MT

**PENGESAHAN**

Skripsi berjudul “**Penerapan *Flowthing Model* pada Sistem *Mobile Voucher Food Delivery*” telah diuji dan disahkan pada:**

hari, tanggal : Kamis, 12 November 2015

tempat : Program Studi Sistem Informasi Universitas Jember.

Tim Penguji:

Ketua,

Anggota I,

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

Windy Eka Yulia R, S. Kom,  
MT

NIP. 196909281993021001

NIP. 198403052010122002

Mengesahkan  
Ketua Program Studi,

Prof. Drs. Slamini, M.Comp.Sc.,Ph.D

NIP. 196704201992011001

**PENGESAHAN PEMBIMBING**

Skripsi berjudul” **Penerapan *Flowthing Model* pada Sistem *Mobile Voucher Food Delivery***”, telah diuji dan disahkan pada:

hari, tanggal : Kamis, 12 November 2015

tempat : Program Studi Sistem Informasi Universitas Jember.

Dosen Pembimbing Utama,

Dosen Pembimbing Anggota,

Anang Andrianto, S.T., M.T.

NIP. 196906151997021002

Nelly Oktavia A,S.Si, MT

NIP. 19841024209122008



## RINGKASAN

**Penerapan *Flowthing Model* pada Sistem *Mobile Voucher Food Delivery*;** Novado Caesar, 102410101006; 2015: 227 halaman; Program Studi Sistem Informasi Universitas Jember.

Perkembangan *internet* tidak dapat terhindarkan lagi. Mulai dari penggunaannya sehari-hari sampai pada dunia bisnis. *Internet* dijadikan pembantu usaha bisnis bagi para usahawan. Kenyamanan yang diberikan adalah koneksi jarak jauh tanpa harus menggunakan kabel seperti komputer di rumah.

Seiring perkembangan teknologi, *internet* merambah ke perangkat *mobile phone*. *Internet* dapat di jalankan pada *mobile phone*, dan dijadikan sebagai peluang bisnis. Banyak pengusaha menggunakan media *internet* via *mobile*, sebagai usaha untuk menarik minat para *customer*.

Salah satu teknologi yang tercipta dari perkembangan *internet* via *mobile* yaitu *mobile voucher*. *Mobile voucher* adalah sistem yang menyediakan nilai rupiah pada *mobile*. Nilai tersebut dapat di tambah dan di kurangi nilainya hanya dengan mengakses sesama *mobile*. Nilai tersebut dapat menggantikan nilai rupiah atau nilai mata uang.

Seiring perkembangan teknologi bisnis yang menjangkau di dunia *internet*, para pengembang teknologi menemukan sebuah teknologi baru. Teknologi itu adalah *service oriented computing*. *Service oriented computing* adalah, cara merancang sistem yang menghasilkan sistem berbasis pelayanan.

*Service oriented computing* tidak dapat mengalamatkan kunci-kunci element pada aplikasi yang berbeda tanpa sebuah metode. Metode yang digunakan agar kunci-kunci *element* tersebut dapat dialamatkan adalah metode *flowthing model*. Metode *flowthing model* dapat di terapkan pada perancangan sistem agar sistem, tersebut menjadi sistem yang berbasis pada pelayanan.

Permasalahan yang muncul adalah bagaimana merancang sistem yang menggunakan *mobile voucher* di dalamnya, agar menjadi sistem berbasis *service oriented*. Sistem tersebut menerapkan metode *flowthing model* guna menciptakan sistem berbasis *service oriented*. Permasalahan yang muncul kemudian adalah bagaimana metode tersebut dapat diterapkan pada sistem *mobile voucher*.

Penerapan metode *flowthing model* pada sistem *mobile voucher* adalah, dengan cara merancang *flow diagram* yang didasarkan pada *user requirement*, pada bagian analisis desain. Jadi kunci-kunci element yang teralamatkan, sesuai dengan apa yang dibutuhkan oleh sistem.

## PRAKATA

Puji syukur ke hadirat Allah SWT atas segala rahmat dan karunia-Nya, sehingga penulis dapat menyelesaikan skripsi dengan judul “Penerapan *Flowthing Model* pada Sistem *Mobile Voucher Food Delivery*”. Skripsi ini disusun untuk memenuhi salah satu syarat menyelesaikan pendidikan Strata Satu (S1) pada Program Studi Sistem Informasi Universitas Jember.

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

1. Anang Andrianto, S.T., M.T. selaku Dosen Pembimbing Utama dan Nelly Oktavia A,S.Si, MT selaku Dosen Pembimbing Anggota yang telah meluangkan waktu, pikiran dan perhatian dalam penulisan skripsi ini;
2. Prof. Drs. Slamini, M.Comp.Sc.,Ph.D selaku Dosen Pembimbing Akademik yang telah membimbing selama penulis menjadi mahasiswa;
3. Seluruh Bapak dan Ibu dosen beserta staf karyawan di Program Studi Sistem Informasi Universitas Jember;
4. Ayah Yonnova Effendy dan Ibu Cicik Suprihatin yang telah memberikan semangat, motivasi, kasih sayang dan doa yang selalu mengalir untuk penulis;
5. Kakakku Gema Budiarto dan Tunanganku Iftitah Adi, yang selalu memberi semangat untuk penulis;

6. Teman – teman PSSI, KKN, SMAN 1 Genteng, *cosplay*, *otaku*, *gemstone lover* yang selalu memberi semangat kepada penulis;
7. Semua pihak yang tidak dapat disebut satu – persatu.

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

Jember, 2 Desember 2015

Penulis

## DAFTAR ISI

DAFTAR GAMBAR.....	xiv
BAB 1. PENDAHULUAN.....	1
1.1 Latar Belakang .....	1
1.2 Rumusan Masalah.....	3
1.3 Batasan Masalah .....	4
1.4 Tujuan .....	4
BAB 2. TINJAUAN PUSTAKA.....	5
2.1 Definisi Sistem Informasi .....	5
2.2 <i>E-commerce</i> .....	6
2.3 <i>M-Commerce</i> .....	7
2.4 <i>User Interface UML</i> .....	8
2.5 <i>Flowthing Model</i> .....	11
2.6 <i>Angular Javascript</i> .....	16
BAB 3. METODE PENELITIAN .....	17
3.1 Objek Penelitian.....	17
3.2 Jenis dan Sumber Data.....	17
3.3 Tahap Perancangan Sistem .....	17
3.3.1 <i>User Requirements</i> .....	18
3.3.2 <i>Specification Design</i> .....	18
3.3.3 <i>Demonstration to Client</i> .....	18
3.3.4 <i>Refine SRS</i> .....	18
3.4 Penyusunan Skripsi.....	19
3.5 Penerapan metode <i>flowthing model</i> .....	19
BAB 4. DESAIN DAN PERANCANGAN SISTEM .....	43
4.1 <i>Use Case Diagram</i> .....	43

4.2 User Interaction Diagram.....	44
4.3 <i>User Interface Diagram</i> .....	60
4.4 <i>Class Diagram</i> .....	61
4.5 <i>Entity Relationships Diagram</i> .....	62
4.6 Implementasi Perancangan .....	63
4.7 Pengujian .....	64
BAB 5. HASIL DAN PEMBAHASAN.....	73
5.1 Penerapan <i>Flowthing Model</i> pada Sistem <i>Mobile Voucher Food Delivery</i> .....	73
5.2 Sistem <i>Mobile Voucher Food Delivery</i> Beserta Fitur-Fiturnya.....	75
BAB 6. PENUTUP .....	82
5.1 Kesimpulan .....	82
5.2 Saran .....	82
DAFTAR PUSTAKA.....	83
LAMPIRAN .....	85



**DAFTAR GAMBAR**

Gambar 2.1 *Use case diagram* ..... 8

Gambar 2.2 *User interaction diagram* ..... 9

Gambar 2.3 *User interface diagram*..... 10

Gambar 2.4 *Class diagram*..... 11

Gambar 2.5 *Diagram Flowthing*..... 12

Gambar 2.6 *Macam-macam flow* ..... 13

Gambar 2.7 *Contoh flow yang berbeda* ..... 13

Gambar 2.8 *Interaksi antar flow* ..... 14

Gambar 2.9 *Screen pada flow of Order*..... 14

Gambar 2.10 *Stakeholder Requirement Specification*..... 15

Gambar 2.11 *Flowthing model pada security system.* ..... 16

Gambar 3.1 *Model prototype* ..... 18

Gambar 3.2 *SRS ordering system* ..... 22

Gambar 3.3 *SRS log-in(mobile) system* ..... 22

Gambar 3.4 *SRS order data system* ..... 22

Gambar 3.5 *SRS product data system*..... 22

Gambar 3.6 *SRS member data system* ..... 22

Gambar 3.7 *SRS top-up data system*..... 22

Gambar 3.8 *Ordering system flow*..... 22

Gambar 3.9 *Log-in(mobile) system flow* ..... 22

Gambar 3.10 *Order data system flow*..... 22

Gambar 3.11 *Product data system flow*..... 22

Gambar 3.12 *Member data system flow* ..... 22

Gambar 3.13 *Top-up data system flow* ..... 31

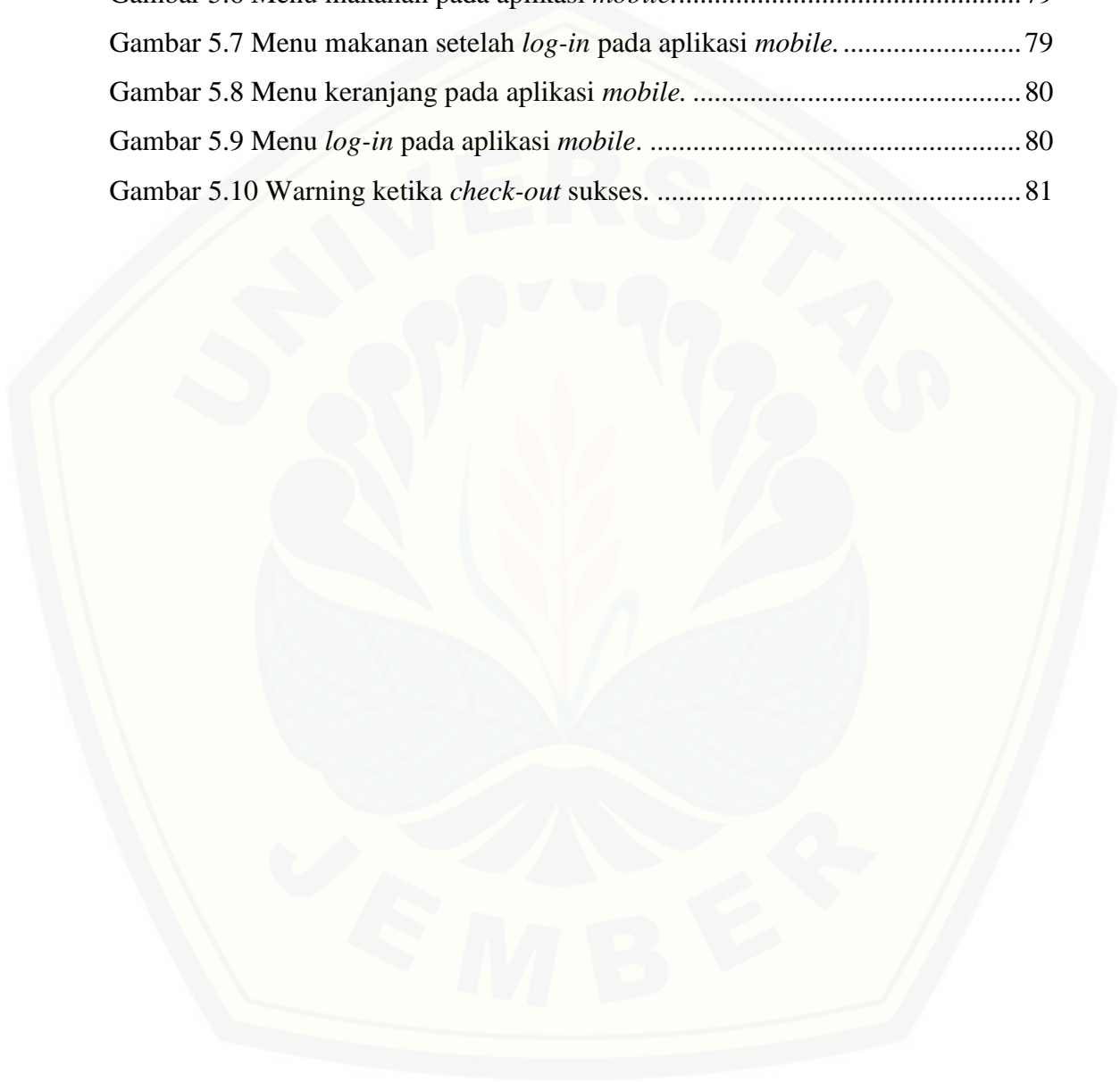
Gambar 3.14 *Ordering system flow diagram* ..... 33

Gambar 3.15 <i>Log-in(mobile ) system flow diagram</i> .....	34
Gambar 3.16 <i>Order data system flow diagram</i> .....	35
Gambar 3.17 <i>Product data system flow diagram</i> .....	36
Gambar 3.18 <i>Member data system flow diagram</i> .....	37
Gambar 3.19 <i>Top-up data system flow diagram</i> .....	38
Gambar 3.20 <i>Mobile Voucher Food Delivery System Flow Diagram</i> .....	40
Gambar 4.1 <i>Use case diagram sistem mobile voucher food delivery.</i> .....	44
Gambar 4.2 <i>Data pesanan user interaction diagram.</i> .....	45
Gambar 4.3 <i>Pesanan deatil user interaction diagram</i> .....	46
Gambar 4.4 <i>Data produk user interaction diagram</i> .....	46
Gambar 4.5 <i>Add produk user interaction diagram</i> .....	47
Gambar 4.6 <i>Cancel add produk user interaction diagram</i> .....	47
Gambar 4.7 <i>Delete produk user interaction diagram</i> .....	48
Gambar 4.8 <i>Produk detail user interaction diagram</i> .....	48
Gambar 4.9 <i>Edit produk user interaction diagram</i> .....	49
Gambar 4.10 <i>Cancel edit produk user interaction diagram</i> .....	49
Gambar 4.11 <i>Data member user interaction diagram</i> .....	50
Gambar 4.12 <i>Add member user interaction diagram</i> .....	50
Gambar 4.13 <i>Cancel add member user interaction diagram</i> .....	51
Gambar 4.14 <i>Delete memnber user interaction diagram</i> .....	51
Gambar 4.15 <i>Member detail user interaction diagram.</i> .....	52
Gambar 4.16 <i>Edit member user interaction diagram</i> .....	52
Gambar 4.17 <i>Cancel edit member user interaction diagram</i> .....	53
Gambar 4.18 <i>Data saldo user interaction diagram.</i> .....	53
Gambar 4.19 <i>Add saldo user interaction diagram</i> .....	54
Gambar 4.20 <i>Cancel add saldo user interaction diagram.</i> .....	54
Gambar 4.21 <i>Delete saldo user interaction diagram</i> .....	55
Gambar 4.22 <i>Saldo detail user interaction diagram</i> .....	55



Gambar 4.23 <i>Edit saldo user interaction diagram</i> .....	56
Gambar 4.24 <i>Cancel edit saldo user interaction diagram.</i> ....	56
Gambar 4.25 <i>Menu mobile user interaction diagram.</i> .....	57
Gambar 4.26 <i>Login mobile user interaction diagram</i> .....	58
Gambar 4.27 <i>Produk mobile user interaction diagram</i> .....	58
Gambar 4.28 <i>Chart data user interaction diagram</i> .....	59
Gambar 4.29 <i>User interface diagram sistem mobile vocuher food delivery.</i> .....	60
Gambar 4.30 <i>Class diagram aplikasi web sistem mobile voucher food delivery.</i> .....	61
Gambar 4.31 <i>Class diagram aplikasi mobile sistem mobile voucher food delivery.</i> .....	62
Gambar 4.32 <i>Entity Relationships Diagram sistem mobile voucher food delivery.</i> .....	63
Gambar 4.33 <i>Coding fungsi menu aplikasi web.</i> .....	64
Gambar 4.34 <i>Complexity dari coding pada gambar 4.35</i> .....	65
Gambar 4.35 <i>Potongan kode aplikasi mobile.</i> .....	66
Gambar 4.36 <i>Complexity dari coding pada gambar 4.37</i> .....	67
Gambar 4.37 <i>Pengujian pada kesalahan log-in (sukses).</i> .....	67
Gambar 4.38 <i>Pengujian pada penambahan menu (sukses).</i> .....	68
Gambar 4.39 <i>Pengujian pada update data makanan.</i> .....	69
Gambar 4.40 <i>Pengujian pada update dan hapus data member.</i> .....	70
Gambar 4.41 <i>Pengujian pada aplikasi mobile (log-in dan menu keranjang sukses).</i> .....	71
Gambar 4.42 <i>Pengujian pada log-in pasca mengisi daftar keranjang dan check-out item.</i> .....	72
Gambar 5.1 <i>Log-in pada aplikasi web</i> .....	76
Gambar 5.2 <i>Menu makanan pada aplikasi web.</i> .....	76
Gambar 5.3 <i>Menu member pada aplikasi web.</i> .....	77

Gambar 5.4 Menu <i>top-up</i> pada aplikasi <i>web</i> . .....	77
Gambar 5.5 Menu pesanan pada aplikasi <i>web</i> . .....	78
Gambar 5.6 Menu makanan pada aplikasi <i>mobile</i> . .....	79
Gambar 5.7 Menu makanan setelah <i>log-in</i> pada aplikasi <i>mobile</i> . .....	79
Gambar 5.8 Menu keranjang pada aplikasi <i>mobile</i> . .....	80
Gambar 5.9 Menu <i>log-in</i> pada aplikasi <i>mobile</i> . .....	80
Gambar 5.10 Warning ketika <i>check-out</i> sukses. ....	81



## BAB 1. PENDAHULUAN

Bab 1 merupakan langkah awal dari penulisan tugas akhir ini. Bab ini berisi latar belakang, perumusan masalah, tujuan dan manfaat, batasan masalah, metodologi penelitian dan sistematika penulisan.

### 1.1 Latar Belakang

Semakin pesatnya perkembangan *internet* dan teknologi komputer, semakin pesat pula peluang untuk mengembangkan bisnis. Mulai dari penyajian produk, cara bertukar informasi, dan cara berkomunikasi dengan *customer*. Oleh karena itu, teknologi dari *mobile commerce* tidak dapat terhindarkan lagi. Terkait dengan penggunaannya yang mudah, dan tanpa harus memakai kabel seperti komputer di rumah. Sejak ada teknologi *mobile commerce*, penjual dapat melebarkan jangkauan pasarnya, menghindari biaya yang mahal, dan memberikan *customer* pelayanan yang lebih baik .

Dunia perdagangan semakin berkembang secara kualitas, seiring dengan berkembangnya teknologi komputer. Tahun 2010, *mobile commerce* meningkatkan jumlah pengguna pada perdagangan online dari *commerce* elektronik kurang lebih 50 persen, dan akan terprediksi untuk tiga tahun ke depan, jumlah pengguna akan terus bertambah. sejak ditemukannya *mobile commerce*, kebiasaan dan cara berinteraksi masyarakat, telah mengalami perubahan (Budde, 2011).

Teknologi komputer yang menjadi perhatian saat ini adalah teknologi komputer tanpa kabel (*wireless*). Melalui teknologi seperti ini, informasi dapat diakses dimana saja dan kapan saja. Contohnya adalah *mobile phone* yang dapat melakukan *browsing*. *Mobile phone* dapat mengakses informasi dari dunia *internet* kapan saja.

*Mobile voucher* merupakan metode transaksi dengan cara menyimpan nilai dalam bentuk nominal, untuk menggantikan nilai uang, dan melakukan pembayaran dengan cara memotong nilai tersebut. Masing-masing konsumen akan didata alamatnya, dan berbagai macam data lainnya yang dibutuhkan bagi pihak *administrator*. Produk-produk makanan dapat diakses via *mobile*, dengan melihat dan memilih menu-menu yang konsumen sukai. Setelah itu konsumen hanya tinggal memesan makanan, dan setelah makanan tersedia di pihak *food court*, makanan akan langsung dikirim ke pihak konsumen. Teknologi *mobile voucher* sudah menjadi salah satu cara bagi perusahaan untuk menarik perhatian para konsumen untuk membeli produk-produknya (Riyan, 2015).

*Service oriented computing* menggunakan pelayanan sebagai dasar untuk membuat sistem berbasis pelayanan yang dapat dikembangkan secara cepat dan beraneka ragam, serta menekan biaya yang serendah-rendahnya. Pelayanan pada sistem, sering dirancang menurut penggunaannya.

*Web service* adalah teknologi yang digunakan menurut konsep *service oriented computing*. *Web service* mengintegrasikan aplikasi berbasis *web*, dengan cara berbagi bisnis proses dan mengkoneksikannya melalui *internet*. Aplikasi ini ditujukan agar dapat digunakan di berbagai *vendor* dan *platform*.

Pelayanan membuat transaksi informasi menjadi lebih mudah, terkait dengan konsep mengakses, memrogram, dan mengintegrasikan aplikasi berbasis pelayanan. Hasil yang diharapkan adalah, kemudahan dalam membuat komposisi aplikasi yang baru, menggunakan logika dari sistem yang telah tersedia sebelumnya. Hal ini merepresentasikan perubahan yang fundamental pada komunitas pengembang *software*, secara efektif dan efisien pada perkembangan *software* di masa depan.

Komponen-komponen pengembangan sistem tidak dapat diaplikasikan begitu saja karena, *Web service* secara teknis tidak mengalamatkan kunci *element* dari pelayanan yang ada. Salah satu masalah terbesar dari pengembangan sistem

*oriented computing* adalah menyediakan metodologi, untuk mendukung spesifikasi dan desain dari komposisi pelayanan yang ada.

Metodologi yang digunakan adalah *flowthing model*, yang dapat merefleksi komponen-komponen dari *web service* yang diproduksi sebelumnya pada *software development lifecycle*. *Flow model* dapat digunakan oleh manajer bisnis dan *system analyst* untuk melacak transformasi ke model yang digunakan *software developer*. Hal ini dilakukan untuk menciptakan sistem berbasis *service oriented*. Kegiatan ini juga menyediakan kesempatan untuk berbisnis dan menentukan standarisasi pada perkembangan aplikasi selanjutnya (Al-Fedaghi, 2014).

Penelitian ini akan mengembangkan *mobile voucher* karena teknologi tersebut dapat menarik minat-minat para konsumen dalam membeli produk yang ditawarkan oleh pihak penjual. Penelitian terkait *mobile commerce* ini akan menggunakan *flowthing model* karena metode ini belum diterapkan pada sistem ini sehingga kunci-kunci element yang akan dirancang belum teralamatkan. Sistem ini juga menekankan pada pelayanan, guna meningkatkan efektifitas sistem dalam dunia bisnis, yang menawarkan berbagai kebutuhan yang berbeda, dan menghindari pembuatan sistem yang dikerjakan hanya berdasarkan kebutuhan. Pengembangan sistem ini akan menggunakan *flowthing model* agar sistem berorientasi pada pelayanan (*service oriented*).

## 1.2 Rumusan Masalah

Berdasarkan uraian diatas, permasalahan yang muncul adalah:

1. Bagaimana penerapan *flowthing model* pada perancangan sistem *mobile voucher food delivery* ?
2. Bagaimana cara membangun sistem *mobile voucher food delivery* yang berbasis pada *service oriented* ?



## 1.3 Batasan Masalah

Batasan masalah dalam penelitian ini serta pada penerapan *flowthing model* di dalam perancangan sistem *mobile voucher food delivery* adalah :

1. Top-up pertama (mendaftar) untuk member hanya dapat dilakukan dengan transaksi manual bersama administrator.
2. Penelitian ini menggunakan 1 user bagi aplikasi *web* dari sistem *mobile voucher food delivery*.
3. Seluruh kebutuhan yang diperlukan oleh sistem pada penelitian ini, dirancang dan diputuskan oleh peneliti dan didasarkan pada penelitian terdahulu.
4. Data pesanan pada aplikasi *web* tidak dapat diubah.

## 1.4 Tujuan

Tujuan dari penulis dalam menulis penelitian ini adalah :

1. Menerapkan *flowthing model* pada perancangan sistem *mobile voucher food delivery*.
2. Membangun sistem *mobile voucher food delivery* yang berbasis pada *service oriented*.

## **BAB 2. TINJAUAN PUSTAKA**

Bab 2 merupakan bab yang berisi paparan mengenai disiplin ilmu yang digunakan dalam penelitian ini. Hal ini diperlukan agar penelitian yang dilakukan dapat dipertanggungjawabkan kebenarannya.

### **2.1 Definisi Sistem Informasi**

Pengertian sistem informasi secara umum merupakan rumusan para definisi sistem informasi yang dikemukakan oleh para ahli pada bidang ini. Pada umumnya dalam setiap kegiatan yang kita lakukan, membutuhkan informasi untuk melakukan hal yang kita lakukan. Seperti yang terjadi sebuah perusahaan, divisi HRD mengelola data para karyawan beserta semua informasi yang ada di dalamnya. Hal ini perlu dikelola karena informasi mengenai karyawan akan sangat dibutuhkan dalam banyak hal lain seperti untuk penggajian.

Menurut Gordon B. Davis “Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel-variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu. Sistem bisa berupa abstraksi atau fisis.”

Sedangkan definisi sistem informasi menurut Tata Sutabri mengemukakan, “Sistem yang abstrak adalah susunan yang teratur dari gagasan-gagasan atau konsepsi yang saling tergantung. Sedangkan sistem yang bersifat fisis adalah serangkaian unsur yang bekerjasama untuk mencapai suatu tujuan.”

Pengertian informasi sendiri, menurut Tata Sutabri “Informasi adalah hasil pengolahan data yang diperoleh dari setiap elemen sistem menjadi bentuk yang mudah dipahami oleh penerimanya dan informasi ini menggambarkan kejadian-kejadian nyata untuk menambah pemahamannya terhadap fakta-fakta yang ada, sehingga dapat digunakan untuk pengambilan suatu keputusan.”

Menurut John F. Nash “Sistem Informasi adalah kombinasi dari manusia, fasilitas atau alat teknologi, media, prosedur dan pengendalian yang bermaksud menata jaringan komunikasi yang penting, proses atas transaksi-transaksi tertentu dan rutin, membantu manajemen dan pemakai intern dan ekstern dan menyediakan dasar pengambilan keputusan yang tepat.”

Definisi-definisi para ahli di atas, kemudian dapat kita rangkum tentang definisi sistem informasi, merupakan kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Penggunaan teknologi disini merujuk pada istilah yang digunakan Teknik Informasi dan Komunikasi (TIK) pada penggunaan database sebagai basis data.

Kemudian jika kita melihat pengertian sistem informasi yang berinteraksi dengan proses bisnis dan melihat pada pengertian organisasi sendiri, sistem informasi merupakan sekumpulan informasi pada sebuah basis data, yang menggunakan model dan media teknologi informasi yang berguna dalam pengambilan keputusan bisnis, pada sebuah organisasi (Ross, 2011).

## **2.2 E-commerce**

Menurut (Hardcastle, 2014), *E-commerce* dapat diibaratkan dengan penggunaan teknologi untuk melakukan kegiatan berbisnis. Seperti membeli suatu barang, atau menjual suatu barang dan melakukan servis-servis yang ada. *E-commerce* tidak hanya sebatas melakukan transaksi dengan menggunakan teknologi yang ada, tetapi juga ada kegiatan dengan wilayah dengan jangkauan yang luas di dalamnya.

*E-commerce* dapat dibagi ke dalam lima macam :

- a. Bisnis ke bisnis atau b2b. Kegiatan ini sangat umum. Hampir 80 persen E-commerce digolongkan dalam kegiatan seperti ini. Kegiatan ini bekerja antara perusahaan satu dengan perusahaan lain.



- b. Bisnis ke Konsumen. Kegiatan ini berlangsung antara perusahaan dengan konsumen. Perusahaan menjual barang dagangannya lewat teknologi yang nantinya di beli oleh konsumen. Konsumen akan menghubungi pihak perusahaan atau sebaliknya.
- c. Bisnis ke pemerintah. Kegiatan ini berlangsung antara perusahaan dengan kantor-kantor pemerintah atau perusahaan milik pemerintah.
- d. Konsumen dengan konsumen. Kegiatan ini berlangsung door to door melewati teknologi.
- e. Mobile *e-commerce* adalah penjualan dengan teknik melewati teknologi telepon genggam. Kegiatan ini menjadikan penjual lebih praktis.

### **2.3 M-Commerce**

Manurut (Anckar, 2002), *m-commerce* adalah transaksi yang terpercaya melalui *mobile*, bertukar benda atau pelayanan antar penjual, pedagang, dan perusahaan. *Mobile commerce* merupakan transaksi yang sempurna, yang dapat dilakukan antara siang dan malam.

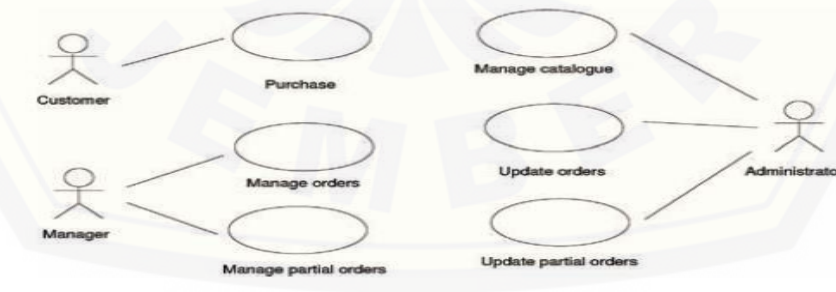
Menurut (Tarasewich, 2003), Penggunaan *mobile* dapat dilakukan dimana saja, kapan saja bagi semua orang. Akan tetapi layar di *mobile* tidak sebesar di *dekstop*, maka dari itu diperlukan kehati-hatian dalam membuat program dagang yang berbasis *mobile*. Dalam melakukan transaksi berbasis *mobile*, tidak diperlukan bantuan *internet* secara fisik. Seperti cara lama yang digunakan pada komputer pribadi.

## 2.4 User Interface UML

Menurut (Hennicker, 2001), mendesain *interface* pada aplikasi *web* termasuk rumit, karena dibutuhkan aspek-aspek metode dan model untuk menggambarannya. Hennicker mempresentasikan metode untuk aplikasi *web* secara konseptual, dinavigasikan, dan dapat dipresentasikan.

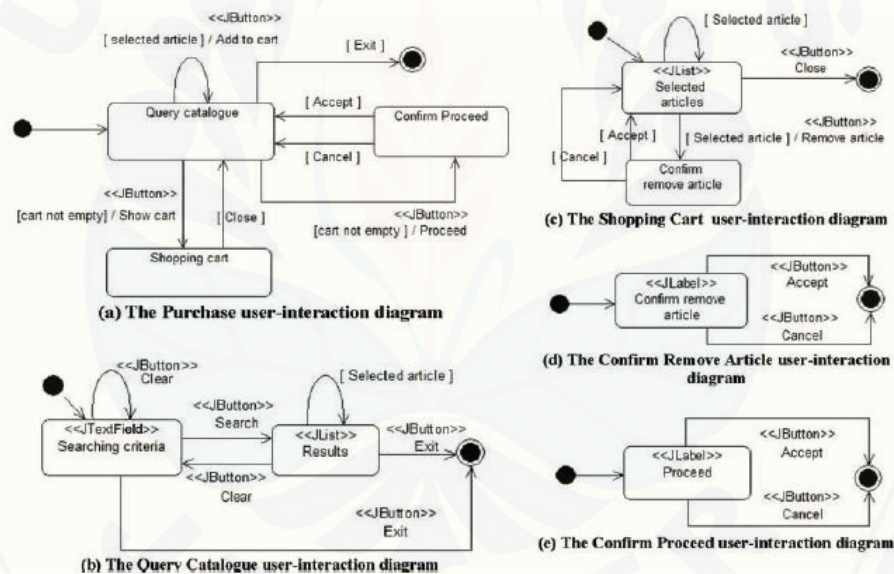
Desain navigasi adalah desain abstrak dari desain *user interface*. Desain navigasi dapat dirancang, setelah desain konsep dibangun. Setelah desain navigasi dirancang, maka tahap selanjutnya membuat desain presentasi. Pada tahap ini diperlukan desain untuk menentukan profil pada aplikasi *web* yang akan di bangun. Pada tahap presentasi ini akan ditentukan bagaimana *user* akan berinteraksi dengan *user interface*.

Menurut (Jesus, 2009), untuk merancang desain pada *user interface*, dapat dimulai dengan merancang *use case*. *Use case* menunjukkan kebutuhan-kebutuhan yang dibutuhkan oleh sistem. *Actor* adalah pihak yang berwenang untuk mengakses sistem, atau berhubungan langsung dengan sistem. *Use case diagram* akan direprestasikan oleh gambar 2.1.



Gambar 2.1 *Use case diagram*.

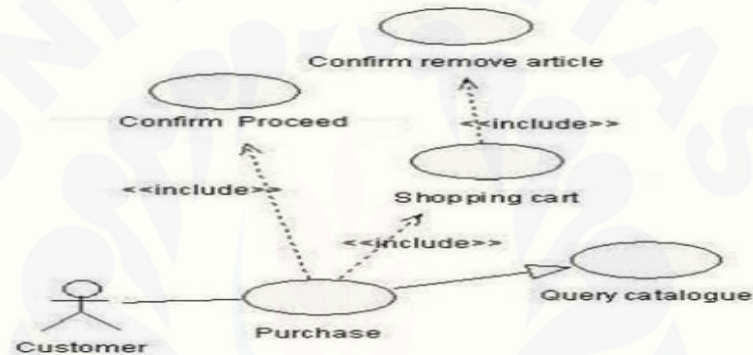
Menurut (Jesus, 2009), desain yang harus dirancang untuk membuat desain *user interface* setelah *use case*, adalah *activity diagram*. *Activity diagram* yang dimaksud adalah, diagram yang dispesialisasikan untuk keperluan *user interface*. Jadi *activity diagram* pada *framework* ini dirancang untuk keperluan *user interface*. *Activity diagram* untuk *user interface* terdiri dari *state* dan *transision*. *State* adalah aksi dari *output data*, yaitu bagaimana sistem bekerja atau merespon dari interaksi *user*. *State* digambarkan oleh kotak. *Transision* adalah kondisi *state* pada saat *state* bekerja. *Transision* digambarkan oleh tanda panah. *Activity diagram* untuk *user interface* dinamakan *user interaction diagram*. *User interaction diagram* akan direpresentasikan pada gambar 2.2.



Gambar 2.2 User interaction diagram.

Setelah *user interaction diagram* dirancang, *user interface diagram* dapat dibangun. *User interface diagram* adalah gabungan dari *use case diagram* dengan *user interaction diagram*. *User interface diagram* dapat disebut juga versi baru dari *use case diagram*. *User interface diagram* menggunakan *actor* yang sama dengan

*use case diagram*, dan menggunakan *state*, yang digunakan oleh *user interaction diagram*. Untuk *relationship* dari *actor* dan *state*, *user interface diagram* menggunakan *include*. Setiap *use case* yang terhubung dengan *include*, akan ikut termodifikasi oleh *use case* induknya. *Use case* yang tidak terhubung dengan *include* oleh *use case* lain, tidak akan termodifikasi. Berikut ini adalah contoh *user interface diagram*.

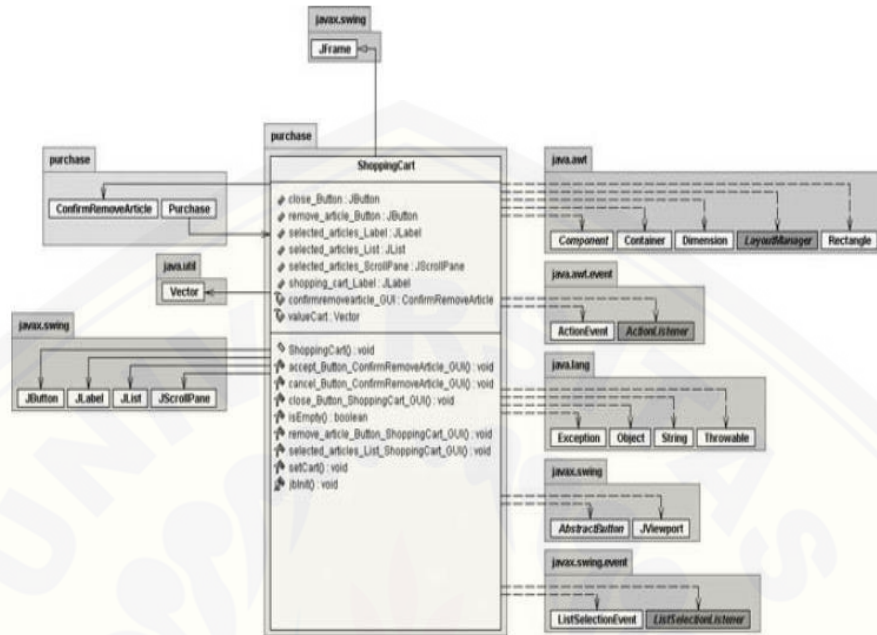


Gambar 2.3 *User interface diagram*.

Setelah *user interface diagram* dibangun, komponen *graphical user interface* akan direpresentasikan dengan *class diagram*. Setiap *use case* yang terhubung dengan *actor* akan dapat di representasikan dengan *window* sebagai komponen dalam *user interface*.

*Class diagram* dibangun dari *window use case diagram* dan *user interaction diagram*. Jika *user interface diagram* dan *user interaction diagram* telah dirancang, maka *class diagram* dapat dibangun pula.

*Class diagram* dibuat dari kelas-kelas *Java swing*. *Use case* direpresentasikan oleh kelas, dan *user interaction diagram* direpresentasikan oleh *frame-frame* pada kelas tersebut. Berikut ini adalah contoh dari *class diagram*.



Gambar 2.4 Class diagram.

## 2.5 Flowthing Model

Model *flowthing* adalah metode yang berorientasi pada pelayanan. Model berorientasi pelayanan ini bertujuan untuk mendukung perkembangan sistem, menekan harga serendah-rendahnya, dan memudahkan distribusi aplikasi pada sistem. Metode pelayanan juga digunakan untuk mempererat hubungan antara penyedia layanan dan pelanggannya.

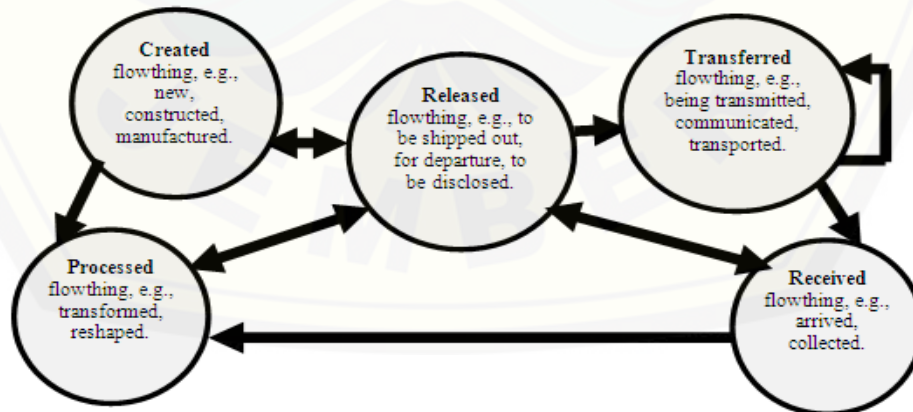
Pada *flowthing model*, *flow* berfungsi sebagai dasar konsep untuk memecahkan masalah yang ada pada bisnis *high level* aplikasi berbasis *web*. Beberapa model lain seperti *workflow*, *flowchart*, *data flow*, *flow diagram*, *cash circular flow*, terikat kepada satu pola. *Flow* adalah konsep dasar untuk memodelkan sistem (Al-Fedaghi, 2014).

Pembuatan desain sistem yang berbasis pada model, sangatlah kompleks. Diperlukan bahasa pemodelan sistem (*system modelling language*(SysML)) untuk

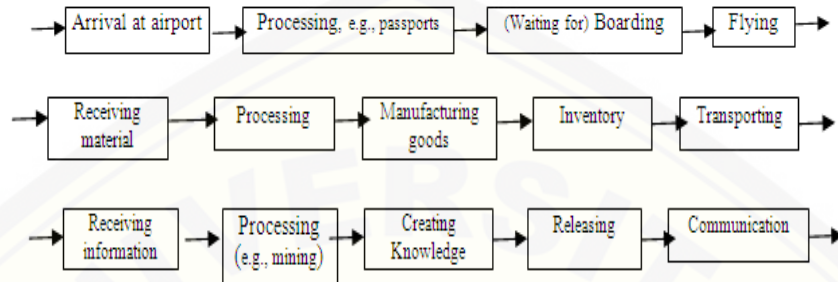


mendukung perkembangan perancangan aplikasi. Pada pembahasan bahasa pemodelan sistem, terdapat spesifikasi, analisis, desain, dan validasi. Daripada menggunakan (*UML*), *SysML* jauh lebih mudah untuk dipelajari dan diterapkan. Sangat dimungkinkan untuk menerjemahkan spesifikasi dalam bahasa tunggal, pada berbagai *teamwork* yang berbeda-beda. Penelitian ini menggunakan metode perancangan sistem klasik, dimulai dengan *requirement analysis*, dan dilanjutkan ke tahap desain sintesis, *software design*, dan implementasi. *Flowthing model (FM)* masuk ke dalam tahap *requirements* dan *functional analysis*.

Model *flowthing* adalah metode yang menggambarkan semua data yang mengalir di dalam *flow*. Segala sesuatu yang mengalir dapat diterima, diproses, dibuat, dilepas, dan ditransfer akan digambarkan di model ini. Segala sesuatu yang mengalir, seperti informasi, material, barang-barang, uang, dan lain-lain dapat dimasukkan ke dalam model ini. Aliran data dapat berupa material, barang, dan informasi ini dapat di proses, diterima, atau dibuat. Satu aktifitas yang dilakukan oleh satu pasang stakeholder dinamakan *flow*. Pada suatu *flow*, dapat terjadi berbagai aktifitas, dan dapat terjadi berbagai aliran data atau informasi. Diagram *flowthing* akan digambarkan pada gambar 2.5. Macam-macam *flow* akan direpresentasikan pada gambar 2.6.

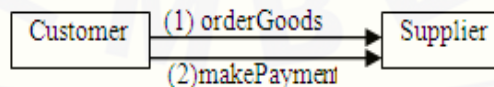


Gambar 2.5 Diagram Flowthing.



Gambar 2.6 Macam-macam *flow*.

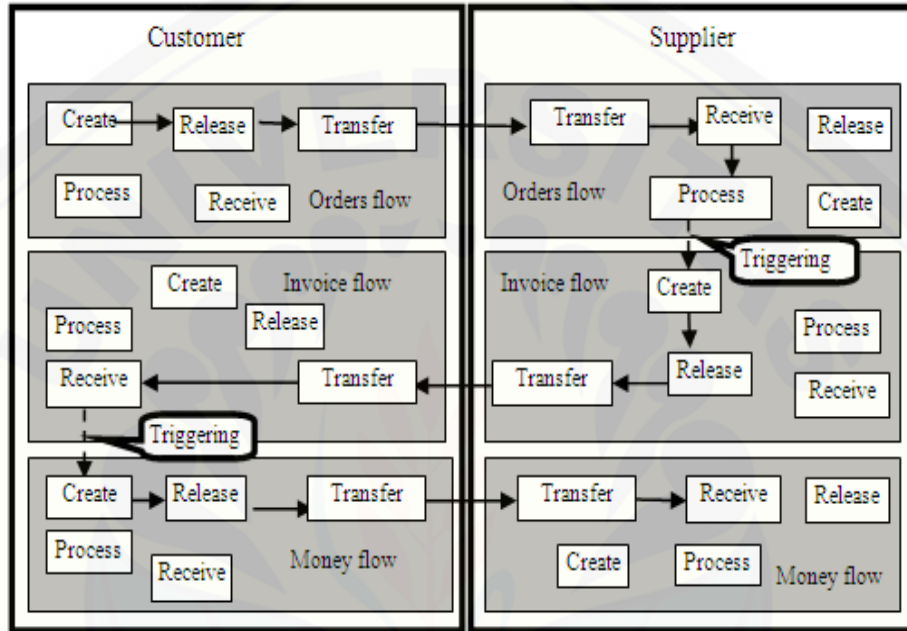
*Flow* bagian atas menggambarkan aliran sistem penerbangan. *Flow* bagian tengah menggambarkan aliran sistem *manufacutring*. *Flow* bagian bawah menggambarkan aliran data. *Flow* dapat berhubungan satu sama lain. Setiap hubungan memiliki *flow of orders*, yaitu tanda hubungan antar *flow*. Berikut ini adalah contoh konteks *flow* berupa penyuplai dan pelanggan. Tanda panah menunjukkan *flow of orders*, yaitu sebagai pemicu antar *flow* yang berbeda. Gambar berikut ini memiliki arti pelanggan dapat memesan barang ke penyuplai, dan memberi pembayaran kepada penyuplai. Dua *flow* yang berbeda digambarkan pada gambar 2.7.



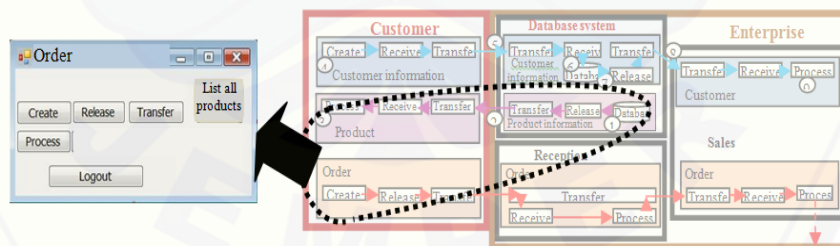
Gambar 2.7 Contoh *flow* yang berbeda.

Setiap *flow* dapat berinteraksi satu sama lain. Satu *flow* dapat mempengaruhi atau ,memicu *flow* lainnya untuk melakukan aktifitas. Terdapat 3 *flow* yang berbeda

yaitu *Orders flow* untuk sistem pemesanan, *Invoice flow* untuk sistem pengadaan produk, dan *money flow* untuk sistem pembayaran. Berikut ini adalah contoh interaksi antar flow dan screen pada *Orders flow*.



Gambar 2.8 Interaksi antar *flow*.



Gambar 2.9 Screen pada *Flow of Order*.

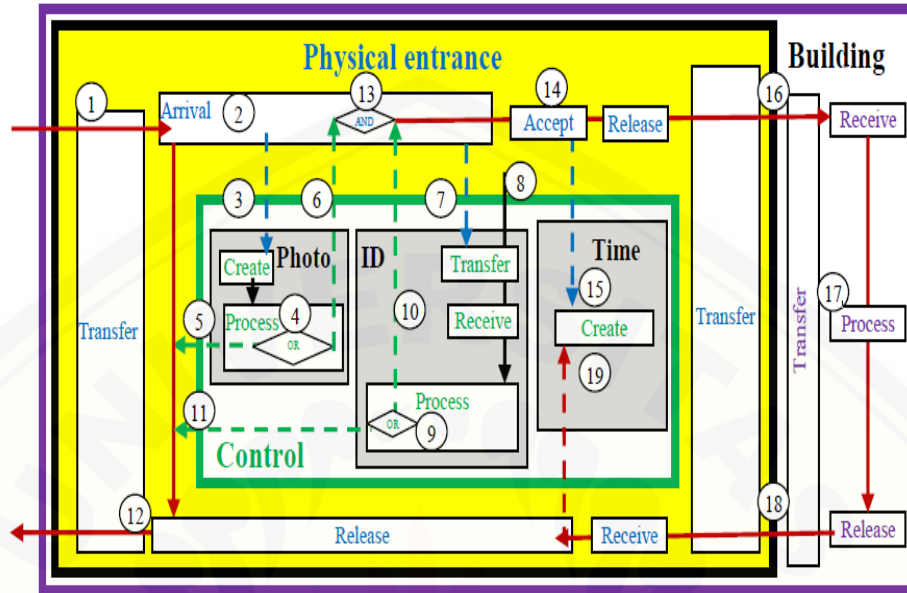
*Requirements analisis* dimulai dengan membuat *SRS(Stakeholder Requirement Spesification)*. Pada bahasan kali ini, akan dicontohkan melalui study



kasus *security system*. Berikut ini adalah contoh dari SRS pada *security system* dan FM pada *security system*.

<i>Security System – Stakeholder Requirements Specification</i>
<p><i>System Overview</i></p> <p><b>SS1: System Summary</b> A security system is to be developed that controls entry and exit to a building through a single point of entry. Identification of personnel will be made by two independent checks. Each person will be photographed upon entry and their time in the building monitored.</p> <p><i>Nominal System Specification</i></p>
<p><b>SS11: Security Checks</b> Secure areas are to be protected by two independent security checks, one based upon an employee ID and one based upon biometric data. Access to secure areas will be unavailable until the users ID is confirmed. The time between the two independent security checks will not exceed a configurable period. The user is allowed three attempts at biometric and / or card identification before access is completely disabled.</p> <p><i>Identification Requirements</i></p> <p><b>SS111: Security Card</b> Access will be denied to any user unless he has a valid Security Card. Security cards only contain the employee name and ID. Out of date security cards cause a denial of access.</p> <p><b>SS112: Biometric Scan</b> Access will be denied to any user unless their biometric data is recognized. The biometric data is to be stored in the system database and not on the security card.</p> <p><b>SS12: Access Priority and Time</b> The system will only process one user at a time, giving them sufficient time to enter and exit the area before automatically securing itself.</p> <p><b>SS13: Exit requirements</b> The user is not allowed to exit until the security card has been successfully authorized.</p> <p><i>Personnel Monitoring</i></p> <p><b>MA1: Image Capture</b> An image is to be taken of any person, at the first attempt, when trying to access a secure area for logging time and employee ID.</p> <p><b>MA2: Time monitoring</b> The time a user spends in a secure area is to be recorded. An alarm will notify administration/security if a person stays longer than 10 hours in the secure area.</p> <p><i>Emergency Requirements</i></p> <p><b>E1: Emergency Exit</b> In the event of an emergency the administrator can invoke a "Free Exit Mode". All security checks for exiting the area are to be disabled until the administrator returns the system to normal working.</p> <p><b>E2: Security Lockdown</b> The administrator can invoke a security lockdown mode - in this event the system will lock all access points until the administrator returns the system to normal working.</p>

Gambar 2.10 Stakeholder Requirement Specification.



Gambar 2.11 Flowthing Model pada Security System.

## 2.6 Angular Javascript

Menurut (Minar, 2014), pada pengembangan aplikasi perangkat lunak, *angularJS* (*angular javascript*) adalah *open source framework* pada aplikasi berbasis web, yang dipelihara oleh *google*, dan dikembangkan oleh pengembang perangkat lunak, untuk mengatasi masalah-masalah yang terkait dengan *single page application*, dengan menggunakan bahasa pemrograman *javascript*.

*Single web application* ini dirancang untuk mempermudah pengembangan aplikasi itu sendiri dan untuk mempermudah *testing* pada program. Kemudahan ini dapat dirasakan bagi pihak *developer* untuk mengembangkan aplikasinya.

## BAB 3. METODE PENELITIAN

Bab 3 berisi tentang penerapan metode *flowthing model* pada perancangan sistem *mobile voucher food delivery*. Penerapan *flowthing model* akan dijelaskan secara sistematis pada bab ini.

### 3.1 Objek Penelitian

Objek dari penelitian ini adalah sistem *mobile voucher food delivery*.

### 3.2 Jenis dan Sumber Data

Jenis sumber data yang diperoleh untuk memperkuat argumentasi dari penelitian ini adalah data sekunder, yang diperoleh dari jurnal-jurnal, artikel, penelitian terdahulu, dan bahan-bahan lainnya yang berhubungan dengan penelitian.

### 3.3 Tahap Perancangan Sistem

Perancangan sistem *mobile voucher food delivery* ini menggunakan model *prototype*. *Flowthing model* akan diterapkan pada sistem ini di bagian analisis spesifikasi desain.

Menurut (Rosa, 2011), model *prototype* cocok untuk analisis kebutuhan yang lebih detail, karena sebagian pelanggan tidak dapat mendeskripsikan kebutuhannya. Dari Penjelasan diatas maka, sesuai dengan sistem yang akan saya buat, maka model yang paling cocok untuk merancang sistem saya adalah model *prototype* karena lebih mengacu pada kebutuhan pelanggan.

Perangkat lunak akan dibangun dengan menggunakan model *prototype*. Fase dari *prototype model* dapat dilihat pada gambar 3.1 (Sabale, 2014).



Gambar 3.1 Model *prototype*.

### 3.3.1 *User Requirements*

Dalam tahapan ini segala kebutuhan mengenai sistem yang akan dirancang oleh programmer, di kumpulkan terlebih dahulu, yang kemudian akan digunakan untuk membuat desain sistem.

### 3.3.2 *Specification Design*

Pada tahap ini desain dari sistem akan di rancang dan di analisis. *Flowthing model* diterapkan pada analisis desain, khususnya pada tahap ini. *Flowthing model* dapat merefleksikan komponen *web service*, dan jika sistem ini akan dikembangkan, *analyst* dari sistem yang baru akan melacak logika dari model ini dan akan menerapkannya pada sistem yang baru.

### 3.3.3 *Demonstration to Client*

Pada saat desain sudah selesai dikerjakan maka, pada tahap ini *prototype* dari sistem siap untuk didemonstrasikan ke pihak *user*. Bila hasil demonstrasi tidak sesuai dengan keinginan user maka, alur perancangan akan kembali ke tahap sebelumnya, yaitu perancangan desain.

### 3.3.4 *Refine SRS*

Pada tahap ini segala kebutuhan user telah *fix* diterapkan pada sistem. Desain telah *fix* dan akan langsung memasuki ke tahap pembuatan program atau software yang sebenarnya.

### 3.4 Penyusunan Skripsi

Setelah desain sistem dibuat, metode flowthing telah di terapkan dan perancangan telah selesai, maka akan disusun laporan skripsi dan sistem informasi *mobile voucher food delivery*.

### 3.5 Penerapan metode *flowthing model*

Metode *flowthing* pada penelitian ini terdapat pada bagian desain. Metode ini terdiri dari *flow* atau aliran informasi yang saling berkaitan. Fungsi dari desain ini adalah untuk mendeteksi aliran informasi yang diperlukan bagi sistem *mobile voucher food delivery*.

Penerapan metode flowthing model diawali dari user requirement. User requirement dibutuhkan agar seluruh kebutuhan pada sistem yang akan dirancang, dapat diketahui dengan valid. User requirement didefinisikan di dalam *stakeholder requirement spesification(SRS)*. SRS didefinisikan sebagai kebutuhan dari sistem yang akan dirancang.. Berikut ini adalah SRS pada sistem mobile voucher food delivery.



<p style="text-align: center;"><i>Ordering System– Stakeholder Requirements Specification</i></p> <p><i>System Overview</i></p> <p>Sistem <i>order</i> terjadi pada aplikasi <i>mobile</i>. <i>Customer</i> dapat melihat produk pada menu makanan, dan <i>customer</i> dapat menentukan jumlah dan pembatalan produk pada menu keranjang. <i>Customer</i> dapat melakukan <i>order</i> ketika telah melakukan <i>log-in</i>. Data order dari <i>customer</i> akan terlihat pada menu <i>order</i> pada aplikasi <i>web</i>.</p>
<p><i>System specification</i></p> <ol style="list-style-type: none"><li>1. <i>Check-out</i>: <i>Check-out</i> terjadi ketika <i>customer</i> mengakses tombol <i>check-out</i>.</li><li>2. <i>Check-out validation</i>: Validasi terjadi ketika <i>customer</i> sedang melakukan <i>check-out</i>. <i>Check-out</i> akan berhasil ketika <i>customer</i> dalam kondisi <i>log-in</i>.</li><li>3. <i>Check-out result</i>: Data saldo dan produk pada aplikasi <i>mobile</i> akan berubah, ketika data <i>update</i> dari <i>database</i> telah diterima oleh aplikasi <i>mobile</i>.</li></ol>

Gambar 3.2 SRS ordering system.

Gambar 3.2 adalah SRS ordering system pada sistem mobile voucher food delivery. Gambar 3.2 menjelaskan kebutuhan ordering system. Sistem ini menjelaskan kebutuhan ordering bagi customer untuk melakukan pemesanan produk pada aplikasi mobile. Terdapat tiga item pada system specification yang akan menjadi acuan untuk menentukan pemetaan flow pada sistem ini.

<p style="text-align: center;"><i>Log-in(mobile) System– Stakeholder Requirements Spesification</i></p> <p><i>System Overview</i></p> <p><i>Customer</i> perlu melakukan <i>log-in</i> pada aplikasi <i>mobile</i> untuk melakukan <i>order</i>. Data saldo masing-masing <i>customer</i> akan terlihat apabila <i>customer</i> telah melakukan <i>log-in</i>. <i>Customer</i> tidak perlu terlebih dahulu melakukan <i>log-in</i>, untuk melihat produk, dan meletakkan produk pada menu keranjang. Sistem <i>log-in</i> membutuhkan <i>password</i> dan nama <i>member</i> yang valid agar <i>log-in</i> berhasil. <i>Log-in</i> tidak berhasil apabila nama dan <i>password</i> <i>member</i> tidak valid.</p>
<p><i>System spesification</i></p> <ol style="list-style-type: none"><li>4. <i>Log-in: Form log-in</i> akan ditampilkan ketika <i>customer</i> mengakses tombol <i>log-in</i>.</li><li>5. <i>Log-in proccess</i> : <i>Customer</i> dapat mengisi form <i>log-in</i> ketika form <i>log-in</i> telah ditampilkan.</li><li>6. <i>Log-in validation: Form log-in</i> akan tampil kembali ketika, data <i>log-in</i> tidak sesuai <i>database</i>. <i>Warning</i> berhasil <i>login</i> akan ditampilkan ketika data <i>login</i> sesuai dengan <i>database</i>.</li><li>7. <i>Saldo's customer</i>: Data saldo pada aplikasi <i>mobile</i> akan tampak ketika data form <i>login customer valid</i> dan data telah terdistribusi dari <i>database</i> ke aplikasi <i>mobile</i>.</li></ol>

Gambar 3.3 SRS *log-in(mobile) system*.

Gambar 3.3 adalah SRS *log-in(mobile)system* pada sistem *mobile voucher food delivery*. Gambar 3.3 menjelaskan kebutuhan *log-in* bagi *customer* pada aplikasi *mobile*. *Log-in* dibutuhkan oleh *customer* untuk melakukan *check-out* pada aplikasi *mobile*.

<p style="text-align: center;"><i>Order Data System– Stakeholder Requirements Specification</i></p> <p><i>System Overview</i></p> <p><i>Order data</i> terletak pada aplikasi <i>web</i>. Data <i>order</i> yang terlihat pada <i>order menu</i> adalah data dari <i>customer</i> pada saat melakukan <i>check-out</i> pada aplikasi <i>mobile</i>. Data <i>order menu</i> tidak dapat diubah. Data <i>order menu</i> hanya memiliki fitur <i>menu detail</i> atau <i>product detail</i>.</p>
<p><i>System specification</i></p> <ol style="list-style-type: none"><li>8. <i>Order request</i>: Data <i>order request</i> akan dikirim ke <i>database</i> ketika <i>administrator</i> mengakses menu <i>order data</i>.</li><li>9. <i>Order replies</i>: <i>Database</i> akan mengirimkan data tabel <i>order</i> ketika, <i>request</i> dari aplikasi <i>web</i> telah sampai pada <i>database</i>.</li><li>10. <i>Order table</i>: Tabel <i>order</i> akan ditampilkan ketika <i>order data</i> dari <i>database</i> telah sampai ke aplikasi <i>web</i>.</li><li>11. <i>Update form order</i>: Data <i>update order</i> menggantikan data <i>order</i> lama ketika <i>order data</i> telah ditampilkan pada aplikasi <i>web</i>.</li></ol>

Gambar 3.4 SRS *order data system*.

Gambar 3.4 adalah SRS *order data system* pada sistem *mobile voucher food delivery*. Gambar 3.4 menjelaskan kebutuhan *order data system*. *Order data sistem* merupakan sistem yang berhubungan dengan data pesanan pada aplikasi *web*.



<p style="text-align: center;"><i>Product Data System– Stakeholder Requirements Specification</i></p> <p><i>System Overview</i></p> <p><i>Product menu</i> terdapat pada aplikasi <i>web</i>. <i>Product menu</i> memiliki fitur <i>add, delete, edit, dan detail</i>. Data yang ada pada <i>product menu</i> dapat diakses melalui <i>mobile</i> oleh <i>customer</i> sebagai menu pilihan yang akan di <i>order</i>.</p>
<p><i>System specification</i></p> <ol style="list-style-type: none"><li>12. <i>Product request</i>: Data <i>product request</i> akan dikirim ke <i>database</i> ketika <i>administrator</i> mengakses menu <i>product data</i>.</li><li>13. <i>Product replies</i>: <i>Database</i> akan mengirimkan data tabel produk ketika, <i>request</i> dari aplikasi <i>web</i> telah sampai pada <i>database</i>.</li><li>14. <i>Product table</i>: Tabel produk akan ditampilkan ketika <i>product data</i> dari <i>database</i> telah sampai ke aplikasi <i>web</i>.</li><li>15. <i>Update form product</i>: Data <i>update</i> produk menggantikan data produk lama ketika produk data telah ditampilkan pada aplikasi <i>web</i>.</li></ol>

Gambar 3.5 SRS *product data system*.

Gambar 3.5 adalah *SRS product data system* pada sistem *mobile voucher food delivery*. Gambar 3.5 menjelaskan kebutuhan produk pada sistem *mobile voucher food delivery*. Sistem ini berhubungan dengan perlakuan *administrator* pada menu produk, pada aplikasi *web*.

<p style="text-align: center;"><i>Member Data System– Stakeholder Requirements Specification</i></p> <p><i>System Overview</i></p> <p><i>Member data system</i> tersedia pada aplikasi <i>web</i>. Sistem ini memiliki fitur <i>add, delete, edit dan detail</i>. Sistem ini adalah sistem yang berhubungan dengan data <i>member</i>. Data saldo yang terdapat di dalam sistem ini dapat terlihat pada aplikasi <i>mobile</i>, pada saat <i>customer</i> berada pada kondisi <i>log-in</i>.</p>
<p><i>System specification</i></p> <ol style="list-style-type: none"><li>16. <i>Member request</i>: Data <i>member request</i> akan dikirim ke <i>database</i> ketika <i>administrator</i> mengakses menu <i>member data</i>.</li><li>17. <i>Member replies</i>: <i>Database</i> akan mengirimkan data tabel <i>member</i> ketika, <i>request</i> dari aplikasi <i>web</i> telah sampai pada <i>database</i>.</li><li>18. <i>Member table</i>: Tabel <i>member</i> akan ditampilkan ketika <i>member data</i> dari <i>database</i> telah sampai ke aplikasi <i>web</i>.</li><li>19. <i>Update form member</i>: Data <i>update member</i> menggantikan data <i>member</i> lama ketika <i>member data</i> telah ditampilkan pada aplikasi <i>web</i>.</li></ol>

Gambar 3.6 SRS *member data system*.

Gambar 3.6 adalah SRS *member data system* pada sistem *mobile voucher food delivery*. Gambar 3.6 menjelaskan kebutuhan *member* pada sistem *mobile voucher food delivery*. Sistem ini berhubungan dengan perlakuan *administrator* pada menu *member*, pada aplikasi *web*.

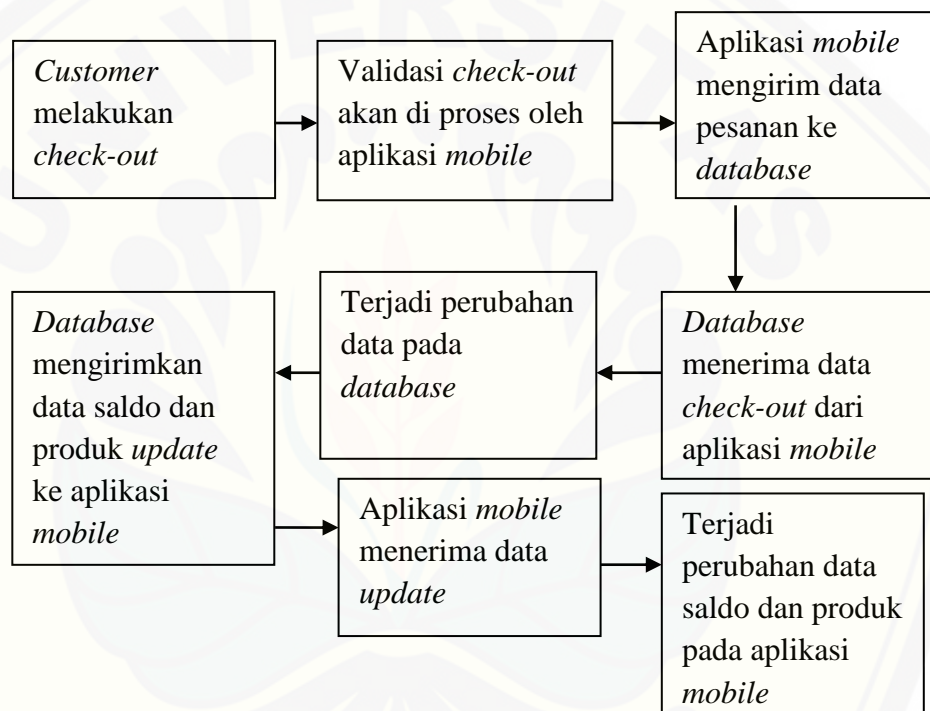
<p style="text-align: center;"><i>Top-up Data System– Stakeholder Requirements Spesification</i></p> <p><i>System Overview</i></p> <p><i>Top-up data system</i> terletak pada aplikasi <i>web</i>. <i>Top-up system</i> berfungsi untuk mengelola data penambahan saldo pada masing-masing <i>member</i>. Perubahan data saldo dapat mempengaruhi tampilan jumlah saldo pada aplikasi <i>mobile</i>.</p>
<p><i>System spesification</i></p> <ol style="list-style-type: none"><li>20. <i>Top-up request</i>: Data <i>top-up request</i> akan dikirim ke <i>database</i> ketika <i>administrator</i> mengakses menu <i>top-up data</i>.</li><li>21. <i>Top-up replies</i>: <i>Database</i> akan mengirimkan data tabel <i>top-up</i> ketika, <i>request</i> dari aplikasi <i>web</i> telah sampai pada <i>database</i>.</li><li>22. <i>Top-up table</i>: Tabel <i>top-up</i> akan ditampilkan ketika <i>top-up data</i> dari <i>database</i> telah sampai ke aplikasi <i>web</i>.</li><li>23. <i>Update form top-up</i>: Data update <i>top-up</i> menggantikan data <i>top-up</i> lama ketika <i>top-up data</i> telah ditampilkan pada aplikasi <i>web</i>.</li></ol>

Gambar 3.7 SRS top-up data system.

Gambar 3.7 adalah SRS top-up data system pada sistem mobile voucher food delivery. Gambar 3.7 menjelaskan kebutuhan penambahan saldo pada sistem mobile voucher food delivery. Sistem ini berhubungan dengan perlakuan administrator pada penambahan saldo member, pada aplikasi web.

SRS pada sistem mobile voucher food delivery telah dirancang. Langkah selanjutnya adalah merancang model flow pada masing-masing SRS yang telah tersedia. Model flow dirancang untuk menggambarkan data yang mengalir pada masing-masing sistem.

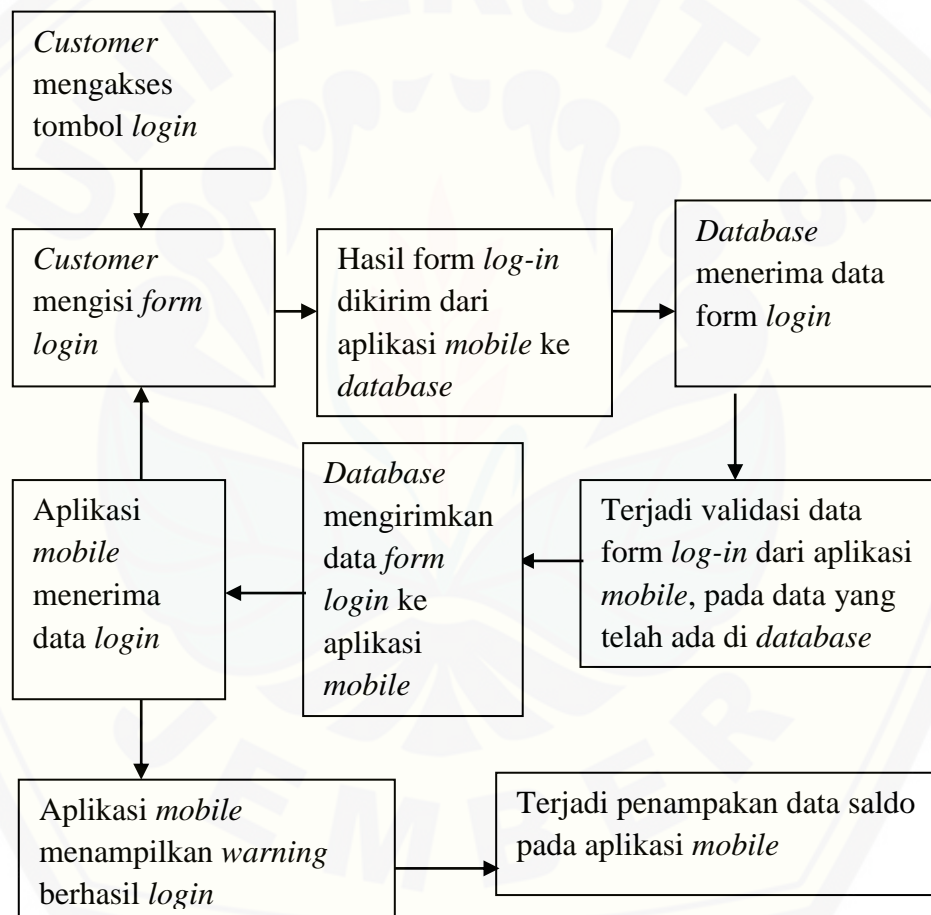
Data yang saling terdistribusi antar aplikasi, tentu saja memiliki sumber data. Fungsi sumber data adalah menyediakan tempat bagi data, untuk disimpan, dan akan dipanggil ketika salah satu aplikasi, atau kedua aplikasi membutuhkannya. Sumber data yang dikenal untuk menyimpan data dikenal dengan sebutan *database*. Berikut ini adalah *flow* yang dirancang sesuai dengan *SRS* yang tersedia.



Gambar 3.8 *Ordering system flow*.

Gambar 3.8 adalah *flow* pada *ordering system*. *Stage* pertama saat *customer* melakukan *check-out* dinamakan *create* karena tidak ada data *check-out* yang disimpan dan ditampilkan sebelumnya. *Stage* kedua adalah termasuk *stage process*. Hal ini dikarenakan validasi merupakan proses pembenaran. Jadi validasi termasuk *stage process*. *Stage* tiga merupakan *stage release*, karena aplikasi *mobile* mengirim data ke luar aplikasi. *Stage* empat merupakan *stage receive*, karena aplikasi menerima data dari aplikasi lain. Antara *stage release* dan *receive* pasti ada

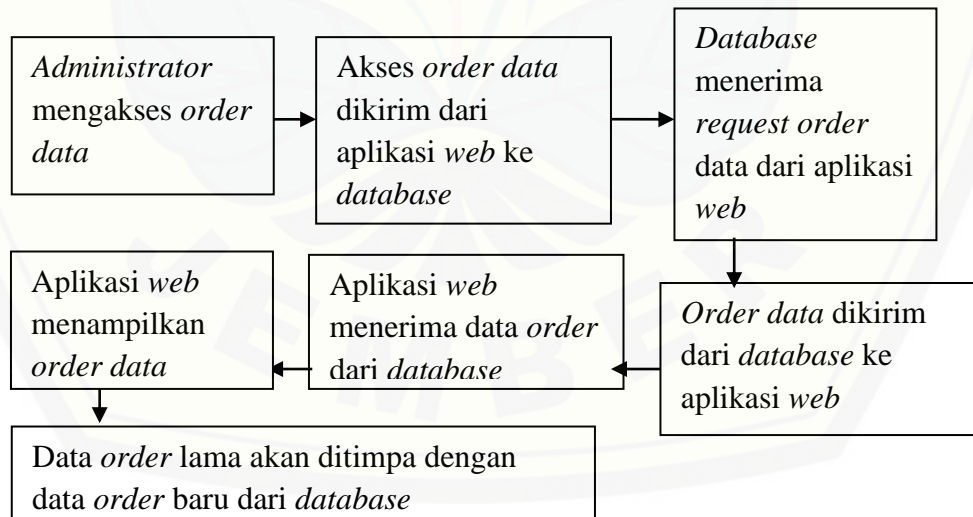
*stage transfer*. Hal ini dikarenakan, data yang mengalir melewati aplikasi yang berbeda. *Stage lima* merupakan *stage process*. hal ini dikarenakan terjadi perubahan data pada *database*. *Stage enam* merupakan *stage release*, karena *database* mengirimkan data menuju ke aplikasi lain. *Stage tujuh* merupakan *stage receive*, karena aplikasi menerima data dari aplikasi lain. *Stage delapan* adalah *stage process*. Hal ini dikarenakan terjadi perubahan nilai pada aplikasi.



Gambar 3.9 Log-in system flow.



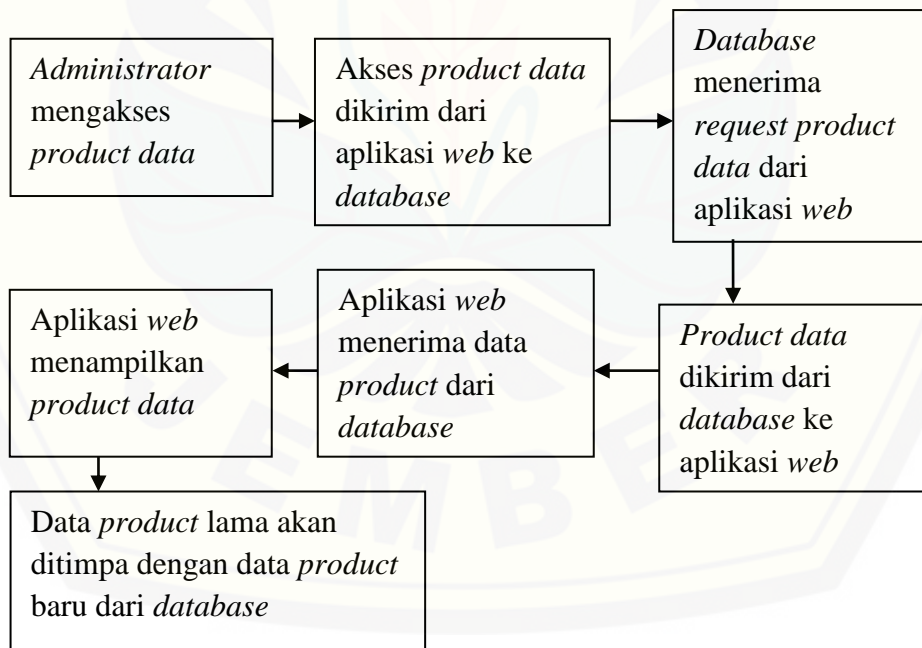
Gambar 3.9 adalah *flow* pada *log-in system*. *Stage* pertama saat *customer* mengakses tombol *log-in* dinamakan *create* karena tidak ada data akses tombol *log-in* yang disimpan dan ditampilkan sebelumnya. *Stage* kedua adalah termasuk *stage process*. Hal ini dikarenakan pengisian *form* merupakan proses. Jadi pengisian *form log-in* termasuk *stage process*. *Stage* tiga merupakan *stage release*, karena aplikasi *mobile* mengirim data ke luar aplikasi. *Stage* empat merupakan *stage receive*, karena aplikasi menerima data dari aplikasi lain. Antara *stage release* dan *receive* pasti ada *stage transfer*. Hal ini dikarenakan, data yang mengalir melewati aplikasi yang berbeda. *Stage* lima merupakan *stage process*. hal ini dikarenakan terjadi validasi data pada *database*. *Stage* enam merupakan *stage release*, karena *database* mengirimkan data menuju ke aplikasi lain. *Stage* tujuh merupakan *stage receive*, karena aplikasi menerima data dari aplikasi lain. *Stage* delapan adalah *stage create* karena tidak ada data yang disimpan terkait tampilan *warning* sebelumnya. *Stage* sembilan adalah *stage process*, karena terdapat perubahan tampilan yang sudah pernah ada sebelumnya.



Gambar 3.10 Order data system flow.

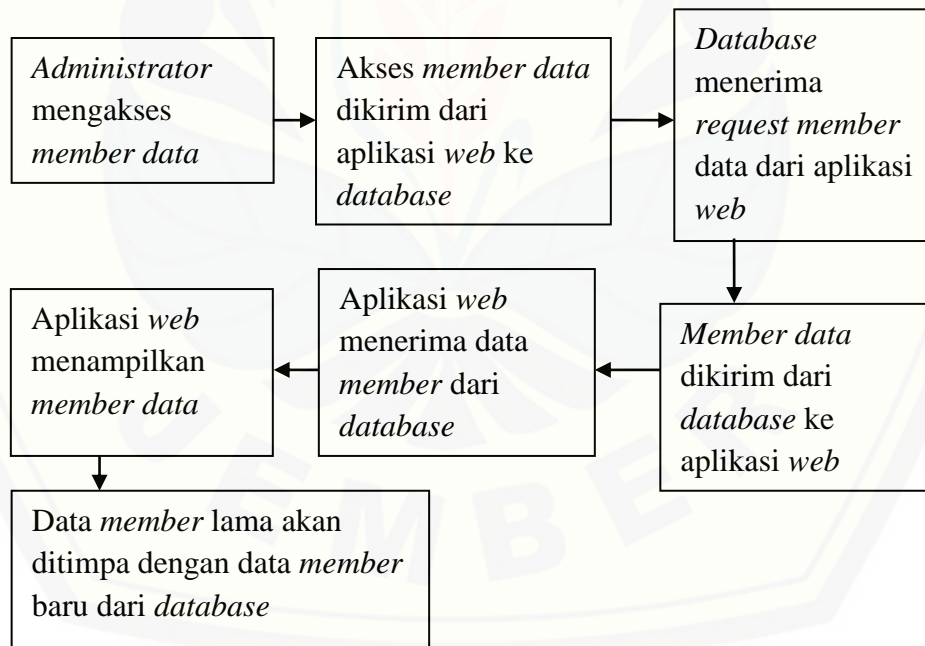


Gambar 3.10 adalah *flow* pada *order data system*. *Stage* pertama saat *administrator* mengakses tombol *order data* dinamakan *create* karena tidak ada akses tombol yang disimpan dan ditampilkan sebelumnya. *Stage* dua merupakan *stage release*, karena aplikasi *web* mengirim data ke luar aplikasi. *Stage* tiga merupakan *stage receive*, karena *database* menerima data dari aplikasi lain. Antara *stage release* dan *receive* pasti ada *stage transfer*. Hal ini dikarenakan, data yang mengalir melewati aplikasi yang berbeda. *Stage* empat merupakan *stage release*, karena *database* mengirimkan data menuju ke aplikasi lain. *Stage* lima merupakan *stage receive*, karena aplikasi menerima data dari *database*. *Stage* enam adalah *stage create*. Hal ini dikarenakan pembuatan *form* data yang diterima dari *database*. *Stage* tujuh merupakan *stage process*, karena terjadi penggantian nilai dari data form lama, dengan data *form* yang baru.



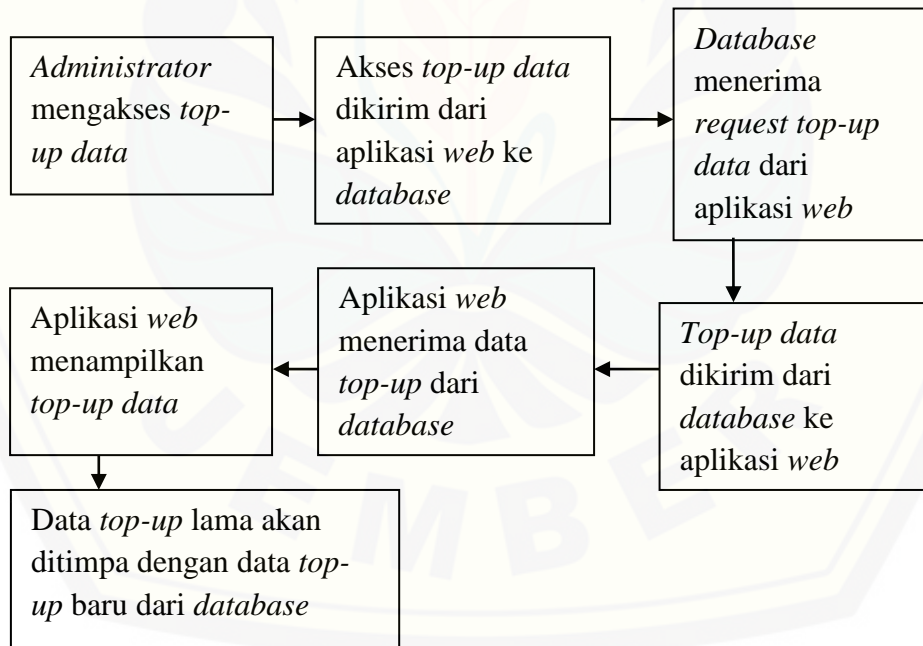
Gambar 3.11 *Product data system flow*.

Gambar 3.11 adalah *flow* pada *product data system*. *Stage* pertama saat *administrator* mengakses tombol *product data* dinamakan *create* karena tidak ada akses tombol yang disimpan dan ditampilkan sebelumnya. *Stage* dua merupakan *stage release*, karena aplikasi *web* mengirim data ke luar aplikasi. *Stage* tiga merupakan *stage receive*, karena *database* menerima data dari aplikasi lain. Antara *stage release* dan *receive* pasti ada *stage transfer*. Hal ini dikarenakan, data yang mengalir melewati aplikasi yang berbeda. *Stage* empat merupakan *stage release*, karena *database* mengirimkan data menuju ke aplikasi lain. *Stage* lima merupakan *stage receive*, karena aplikasi menerima data dari *database*. *Stage* enam adalah *stage create*. Hal ini dikarenakan pembuatan *form* data yang diterima dari *database*. *Stage* tujuh merupakan *stage process*, karena terjadi penggantian nilai dari data *form* lama, dengan data *form* yang baru.



Gambar 3.12 *Member data system flow*.

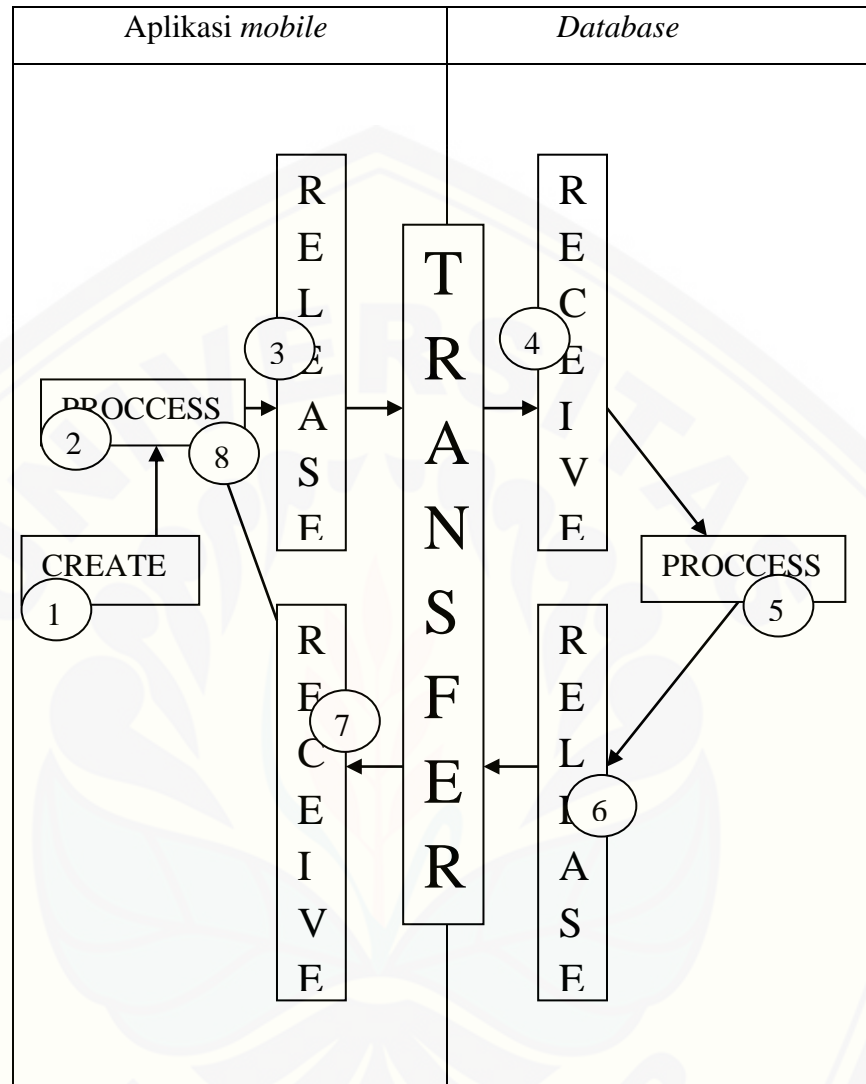
Gambar 3.12 adalah *flow* pada *member data system*. *Stage* pertama saat *administrator* mengakses tombol *member data* dinamakan *create* karena tidak ada akses tombol yang disimpan dan ditampilkan sebelumnya. *Stage* dua merupakan *stage release*, karena aplikasi *web* mengirim data ke luar aplikasi. *Stage* tiga merupakan *stage receive*, karena *database* menerima data dari aplikasi lain. Antara *stage release* dan *receive* pasti ada *stage transfer*. Hal ini dikarenakan, data yang mengalir melewati aplikasi yang berbeda. *Stage* empat merupakan *stage release*, karena *database* mengirimkan data menuju ke aplikasi lain. *Stage* lima merupakan *stage receive*, karena aplikasi menerima data dari *database*. *Stage* enam adalah *stage create*. Hal ini dikarenakan pembuatan *form* data yang diterima dari *database*. *Stage* tujuh merupakan *stage process*, karena terjadi penggantian nilai dari data *form* lama, dengan data *form* yang baru.



Gambar 3.13 *Top-up data system flow*.

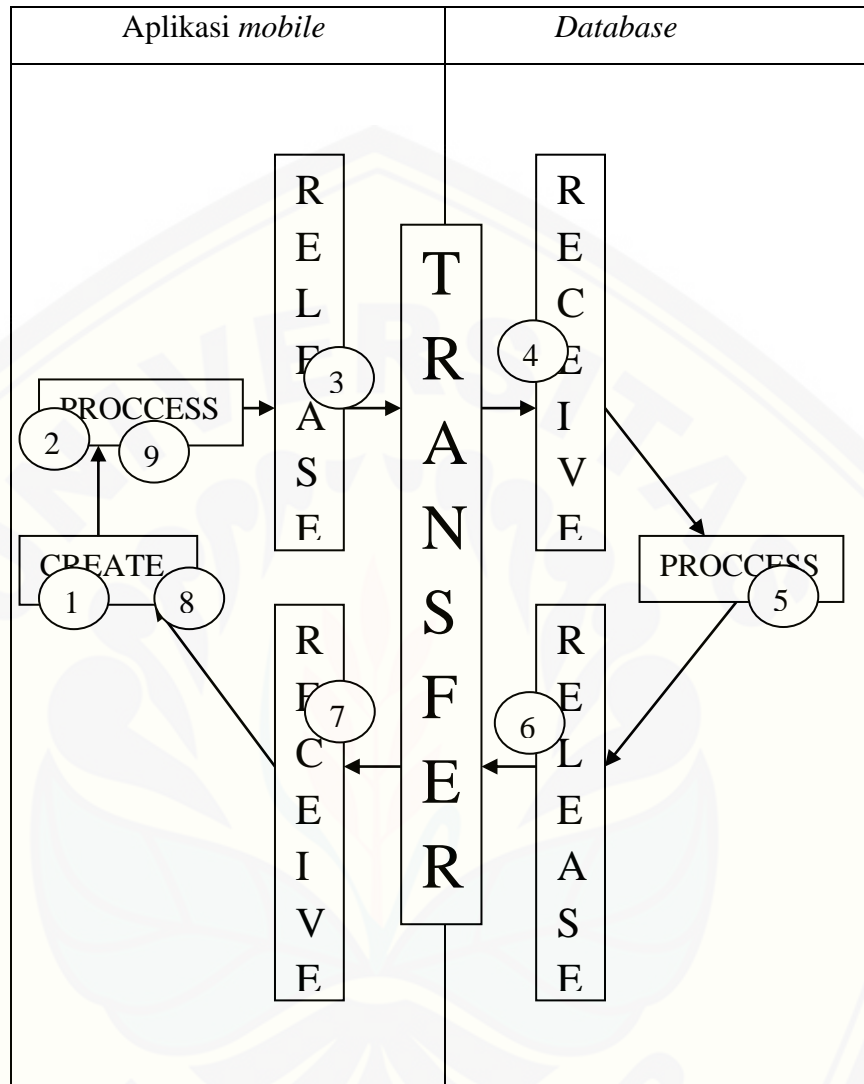
Gambar 3.13 adalah *flow* pada *top-up data system*. *Stage* pertama saat *administrator* mengakses tombol *top-up data* dinamakan *create* karena tidak ada akses tombol yang disimpan dan ditampilkan sebelumnya. *Stage* dua merupakan *stage release*, karena aplikasi *web* mengirim data ke luar aplikasi. *Stage* tiga merupakan *stage receive*, karena *database* menerima data dari aplikasi lain. Antara *stage release* dan *receive* pasti ada *stage transfer*. Hal ini dikarenakan, data yang mengalir melewati aplikasi yang berbeda. *Stage* empat merupakan *stage release*, karena *database* mengirimkan data menuju ke aplikasi lain. *Stage* lima merupakan *stage receive*, karena aplikasi menerima data dari *database*. *Stage* enam adalah *stage create*. Hal ini dikarenakan pembuatan form data yang diterima dari *database*. *Stage* tujuh merupakan *stage process*, karena terjadi penggantian nilai dari data *form* lama, dengan data *form* yang baru.

*Flow* masing-masing sistem telah dirancang. Tahap berikutnya adalah merancang *flow diagram* dari masing-masing sistem. *Flow diagram* ini menyesuaikan *flow* yang telah dirancang sebelumnya, atau yang telah terdefiniskan sebelumnya. Berikut ini adalah *flow diagram* dari masing-masing sistem.



Gambar 3.14 *Ordering system flow diagram.*

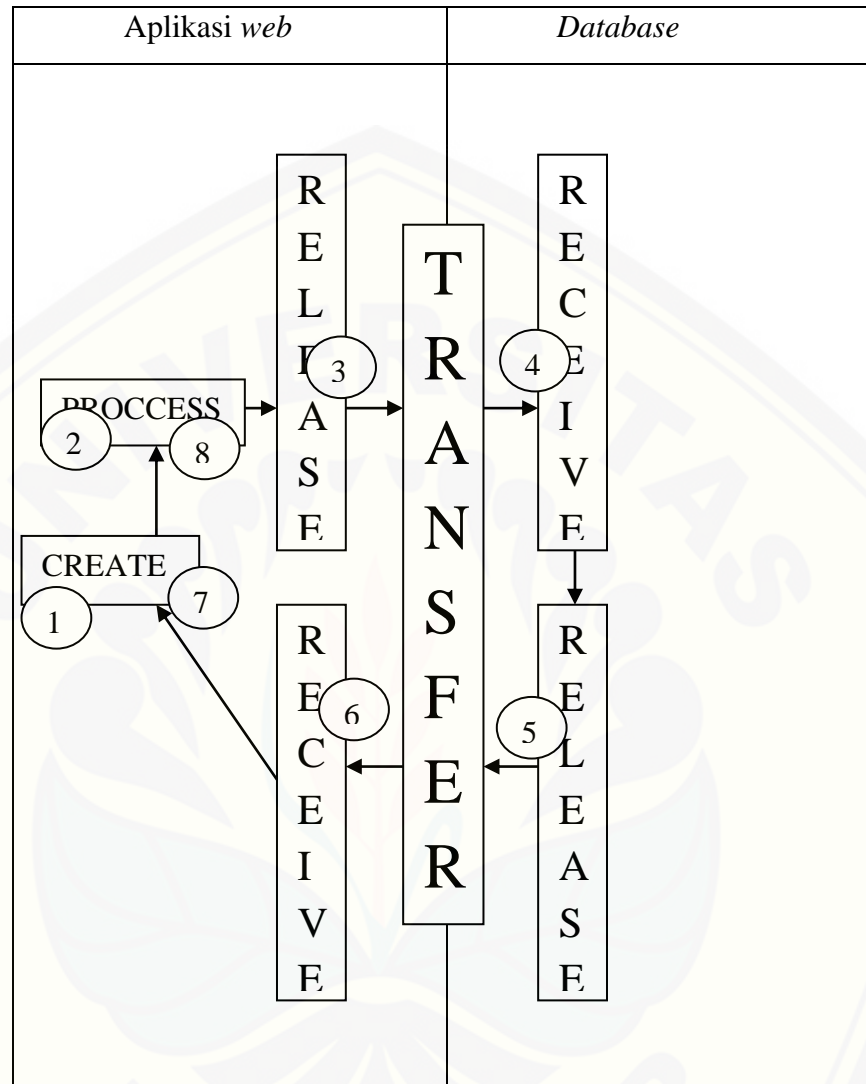
Gambar 3.14 adalah *flow diagram* dari *ordering system*. Nomor yang ditampilkan pada diagram tersebut adalah *stage* dari *flow* yang mengalir pada diagram tersebut. *Stage-stage* yang terjadi pada diagram tersebut telah dijelaskan pada desain *flow* sebelumnya.



Gambar 3.15 Log-in system flow diagram.

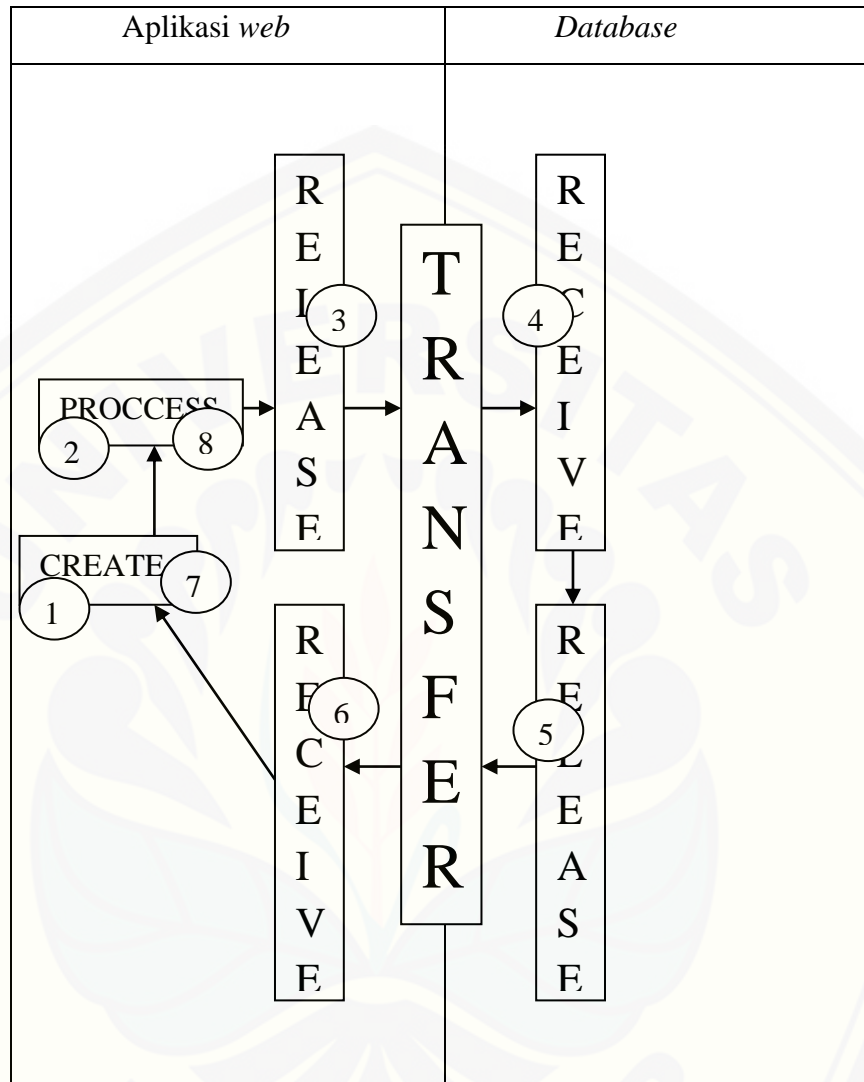
Gambar 3.15 adalah *flow diagram* dari *log-in(mobile) system*. Nomor yang ditampilkan pada diagram tersebut adalah *stage* dari *flow* yang mengalir pada diagram tersebut. *Stage-stage* yang terjadi pada diagram tersebut telah dijelaskan pada desain *flow* sebelumnya.





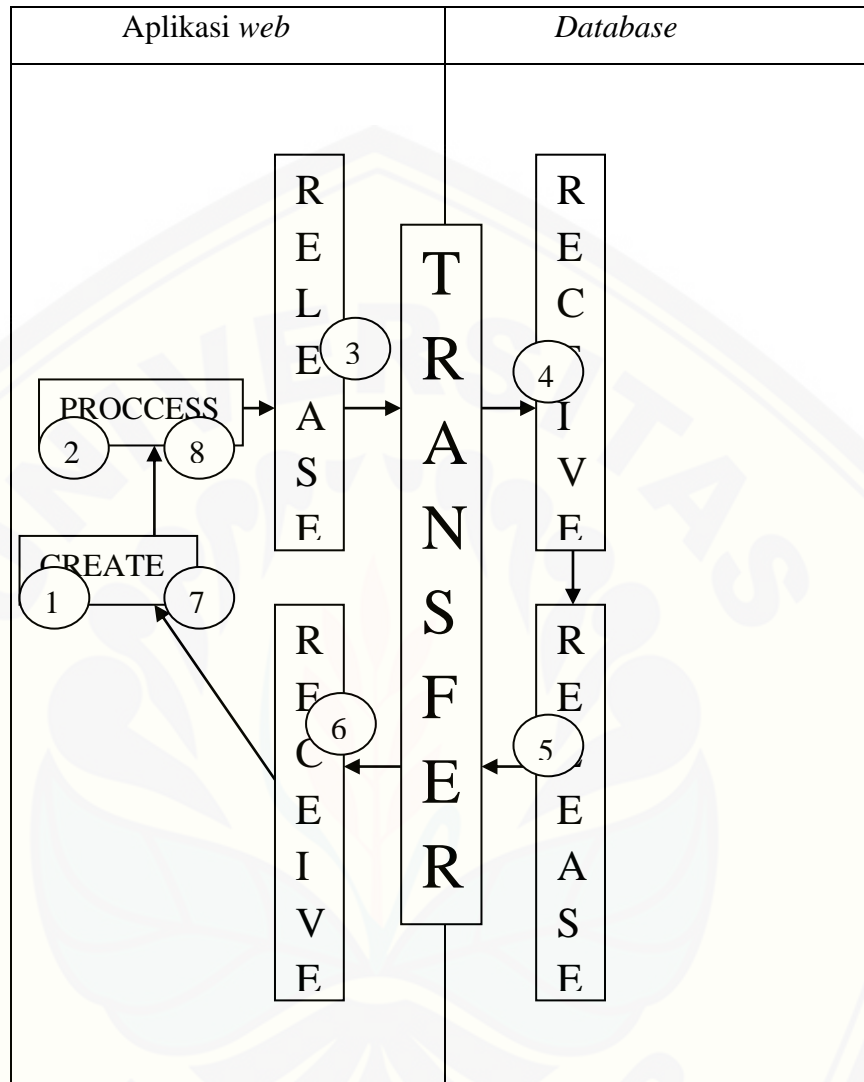
Gambar 3.16 Order data system flow diagram.

Gambar 3.16 adalah *flow diagram* dari *order data system*. Nomor yang ditampilkan pada diagram tersebut adalah *stage* dari *flow* yang mengalir pada diagram tersebut. *Stage-stage* yang terjadi pada diagram tersebut telah dijelaskan pada desain *flow* sebelumnya.



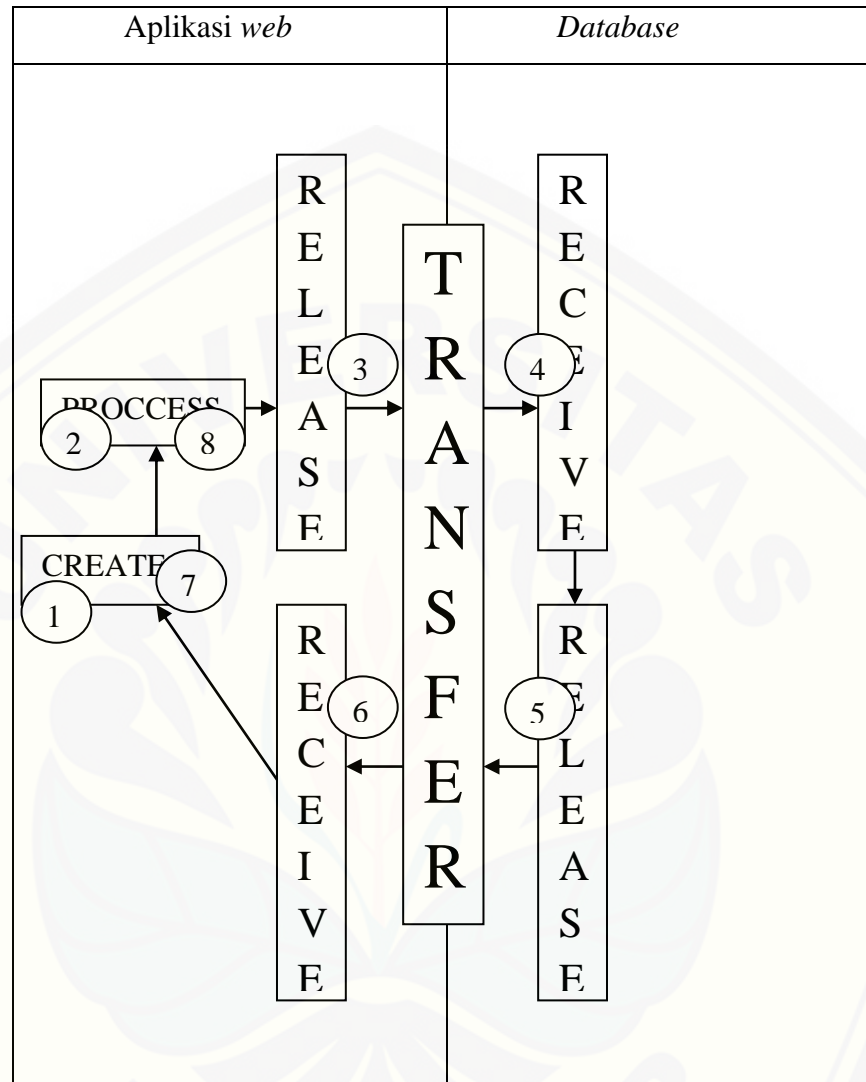
Gambar 3.17 Product data system flow diagram.

Gambar 3.17 adalah *flow diagram* dari *product data system*. Nomor yang ditampilkan pada diagram tersebut adalah *stage* dari *flow* yang mengalir pada diagram tersebut. *Stage-stage* yang terjadi pada diagram tersebut telah dijelaskan pada desain *flow* sebelumnya.



Gambar 3.18 Membervdata system flow diagram.

Gambar 3.18 adalah *flow diagram* dari *member data system*. Nomor yang ditampilkan pada diagram tersebut adalah *stage* dari *flow* yang mengalir pada diagram tersebut. *Stage-stage* yang terjadi pada diagram tersebut telah dijelaskan pada desain *flow* sebelumnya.



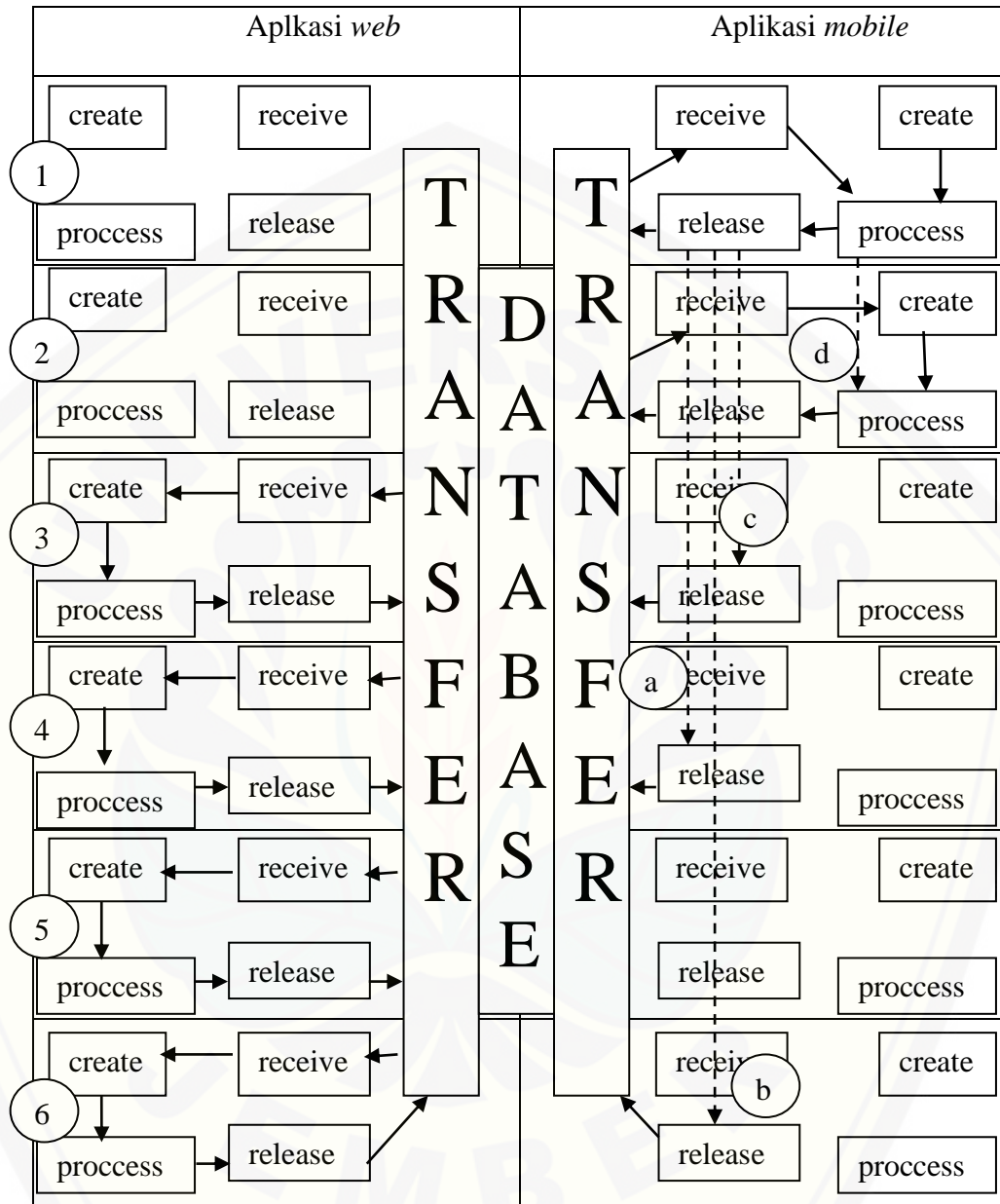
Gambar 3.19 Top-up data system flow diagram.

Gambar 3.19 adalah *flow diagram* dari *top-up data system*. Nomor yang ditampilkan pada diagram tersebut adalah *stage* dari *flow* yang mengalir pada diagram tersebut. *Stage-stage* yang terjadi pada diagram tersebut telah dijelaskan pada desain *flow* sebelumnya.

Sistem *mobile voucher food delivery* adalah sistem yang terdiri dari aplikasi *mobile* dan aplikasi *web*. Aplikasi yang saling berkaitan, tentu memiliki distribusi data di dalamnya. Aliran data(*flow*) juga dapat terjadi antara kedua aplikasi tersebut.

Rancangan *flow diagram* pada masing-masing *system* yang di desain menurut *SRS* pada sistem *mobile voucher food delivery*, telah ditetapkan. Hal selanjutnya yang di rancang adalah *flowthing diagram* pada keseluruhan sistem *mobile voucher food delivery*. *Flowthing diagram* ini dirancang agar distribusi data yang terjadi pada aplikasi *web* ke aplikasi *mobile* atau sebaliknya, dapat terdeteksi.

*Flowthing diagram* sistem *mobile voucher food delivery* juga akan mendeskripsikan hubungan antara masing-masing subsistem yang dirancang sebelumnya sesuai dengan *SRS* yang tersedia. Berikut ini adalah *flowthing diagram* pada sistem *mobile voucher food delivery*.



Gambar 3.20 Mobile voucher food delivery system flow diagram.

Gambar 3.20 adalah *flowthing diagram* pada sistem *mobile voucher food delivery*. Terdapat enam *flow* yang di desain pada *flowthing diagram* tersebut dan ditandai dengan lingkaran dengan nominal di dalamnya. Hal ini dikarenakan



*flowthing diagram* ini dirancang dari enam *flow*, pada masing-masing subsistem sebelumnya.

Lingkaran dengan nomor satu adalah *flow of ordering* yang disesuaikan dari *ordering system*. Lingkaran dengan nomor dua adalah *flow of log-in* yang disesuaikan menurut *log-in(mobile) system*. Lingkaran dengan nomor tiga adalah *flow of order data* yang disesuaikan oleh *order data system*. Lingkaran nomor empat adalah *flow of product data* yang disesuaikan dari *product data system*. Lingkaran nomor lima adalah *flow of member data* yang disesuaikan dari *member data system*. Lingkaran nomor enam adalah *flow of top-up data* yang disesuaikan dari *top-up data system*.

*Flowthing diagram* pada sistem *mobile voucher food delivery* diatas menggambarkan hubungan antar *flow*. Hubungan ini terlihat pada *flow-flow* yang terhubung oleh garis putus-putus yang disebut pemicu, atau *trigger*. Jadi *flow* awal yang ditarik oleh garis putus-putus adalah *flow* pemicu Bergeraknya aliran data pada *flow* lain.

Pada *flowthing diagram* diatas terdapat lingkaran dengan huruf alfabet. Lingkaran tersebut menandakan *trigger* yang terjadi pada *flowthing diagram* tersebut.

Lingkaran dengan alfabet a menerangkan hubungan antara *flow of product data* dengan *flow of ordering*. Hal ini dikarenakan *flow of ordering* memberikan data *update product* ke *database*, yang akan di terima oleh *flow of product data* dari *database*. Hal yang terjadi adalah perubahan data pada *update product data* yang terlihat pada *stage process* pada *product data flow*.

Lingkaran dengan alfabet b menerangkan hubungan antara *flow of top-up data* dengan *flow of ordering*. Hal ini dikarenakan *flow of ordering* memberikan data *update saldo* ke *database*, yang akan di terima oleh *flow of top-up data* dari *database*. Hal yang terjadi adalah perubahan data pada *update to-up data* yang terlihat pada *stage process* pada *top-up data flow*.

Lingkaran dengan alfabet c menerangkan hubungan antara *flow of order data* dengan *flow of ordering*. Hal ini dikarenakan *flow of ordering* memberikan data *order* ke *database*, yang akan di terima oleh *flow of order data* dari *database*. Hal yang terjadi adalah perubahan data pada *update order data* yang terlihat pada *stage process* pada *order data flow*.

Lingkaran dengan alfabet d menjelaskan hubungan antara *flow of ordering* dan *flow of log-in*. Hubungan yang terjadi adalah, ketika validasi pada *flow of ordering* pada *stage process* menyebutkan bahwa *customer* belum melakukan *log-in*, maka sistem akan masuk ke *stage process* pada *flow of log-in*. Sistem akan menampilkan *form log-in*, dan akan di isi oleh *customer*. Jadi pada saat *customer* belum melakukan *log-in* ketika melakukan *check-out*, maka sistem akan menampilkan *form log-in*.

## BAB 4. DESAIN DAN PERANCANGAN SISTEM

Pada bab ini akan diuraikan mengenai desain apa saja yang diperlukan untuk merancang sistem *mobile voucher food delivery*, dan sistem perancangan *mobile voucher food delivery* itu sendiri, terkait dengan metode penelitian dan dibatasi oleh batasan masalah yang ada.

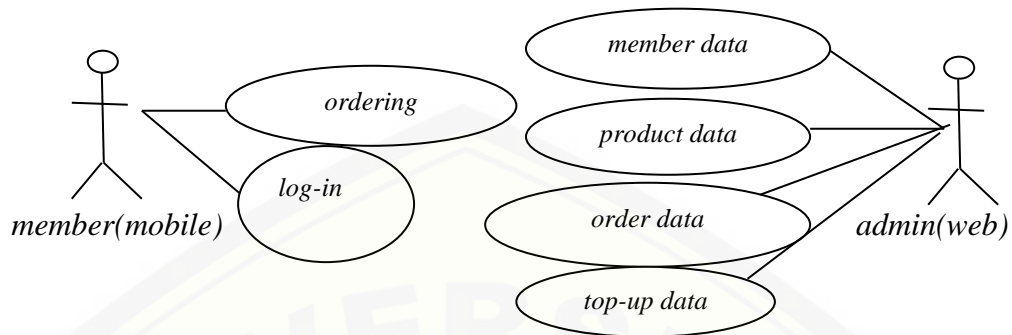
Desain perancangan sistem *mobile voucher food delivery* terdiri dari :

- a. *Use case diagram.*
- b. *User interaction diagram.*
- c. *User interface diagram.*
- d. *Class diagram.*
- e. *Entity relationship diagram.*

### 4.1 Use Case Diagram

Berikut ini adalah *use case diagram* dari sistem *mobile voucher food delivery*. Member dapat melakukan pemesanan pada pihak admin melalui *mobile*, jika saldo yang tersedia masih cukup.

Pihak *admin* dapat mengelola data menu makanan, data *member*, data pesanan, dan data saldo dari para *member*. Status pada menu makanan dapat di ganti menjadi tersedia atau tidak tersedia, sesuai dengan ketersediaannya di pihak perusahaan.



Gambar 4.1 Use case diagram sistem mobile voucher food delivery.

Gambar 4.1 adalah *use case* dari sistem *mobile voucher food delivery*. Aplikasi *mobile* berhubungan dengan pihak *customer*. *Customer* dapat memesan menu makanan pada aplikasi *mobile*. Aplikasi *web* berhubungan dengan pihak *administrator*. Pihak *administrator* dapat melihat data pesanan, mengubah menu, mengisi saldo *customer*, serta menambah *member* baru.

## 4.2 User Interaction Diagram

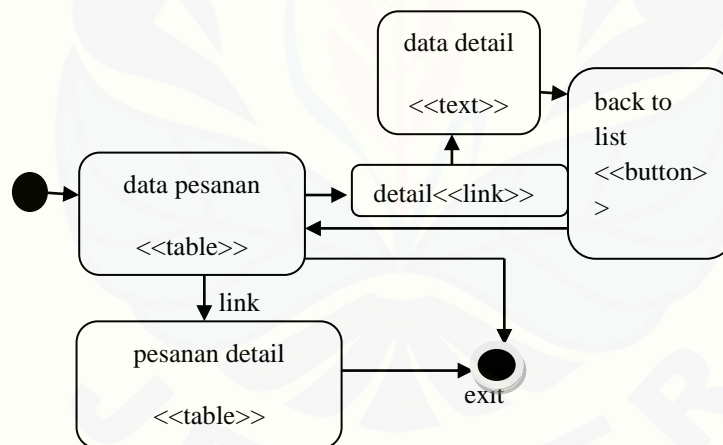
Desain yang dirancang setelah *use case* adalah *user interaction diagram*. Diagram ini menjelaskan bagaimana interaksi antara pengguna sistem dan *software* itu berkomunikasi.

Gambar 4.2 sampai dengan 4.25 adalah *user interaction diagram* bagi aplikasi *web* pada sistem *mobile voucher food delivery*. Gambar 4.2 menjelaskan data pesanan pada aplikasi *web*. data pesanan hanya memiliki fitur detail. data pesanan tentunya tidak dapat diubah oleh administrasi. Data pesanan memiliki *link* ke pesanan *detail* yang berfungsi untuk melihat semua menu yang dipesan oleh *customer*. Gambar 4.3 menjelaskan tentang pesanan detail, yang terhubung dengan *link* dari data pesanan.

Gambar 4.4 sampai dengan 4.10 menjelaskan tentang fitur menu atau produk. fitur produk memiliki fitur utama yaitu *add, detail, edit, dan delete*. produk dapat ditambah jumlahnya, dihapus, di rubah datanya, dan dilihat secara rinci.

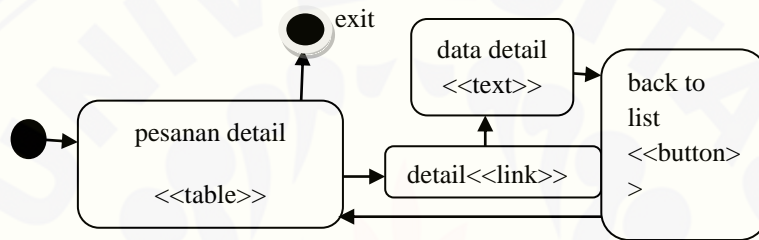
Gambar 4.11 sampai dengan 4.17 menjelaskan tentang fitur member. *User* mendaftar langsung kepada pihak *administrator*, dengan mengisi saldo sejumlah yang disepakati. Setelah user memiliki saldo, maka *user* telah terdaftar sebagai *member*.

Gambar 4.18 sampai dengan 4.24 menjelaskan tentang saldo *member*. *Member* dapat menambah saldo mereka sesuai dengan keinginan. Gambar 4.25 adalah *user interaction diagram* dari fitur *login*. Pihak *administrator* harus *login* terlebih dahulu sebelum memasuki aplikasi *web*. Gambar-gambar di bawah ini adalah *user interaction* pada aplikasi *web*.



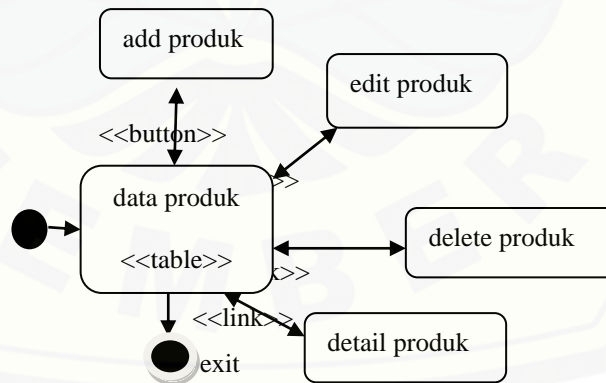
Gambar 4.2 Data pesanan *user interaction diagram*.

Gambar 4.2 adalah *user interaction* pada padat pesanan aplikasi berbasis *web*. Data pesanan digambarkan melalui sebuah tabel. Data pesanan memiliki fitur data detail dan pesanan detail. Data detail berisi penjelasan pada masing-masing baris data pesanan. Pesanan detail berisi penjelasan menu pada tabel pesanan.



Gambar 4.3 Pesanan detail *user interaction diagram*.

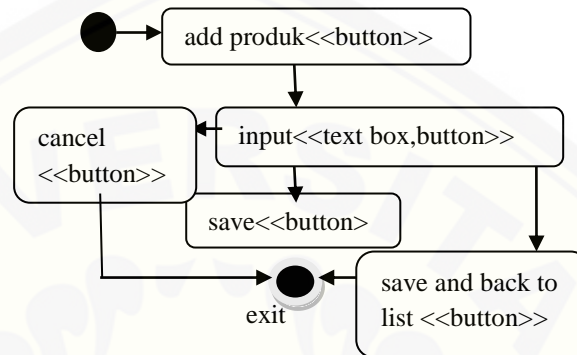
Gambar 4.3 adalah *user interaction* dari bentuk data pesanan yang lebih detail. Sama dengan data pesanan, pesanan detail juga memiliki fitur detail pada masing-masing baris pada tabel pesanan detail.



Gambar 4.4 Data produk *user interaction diagram*.

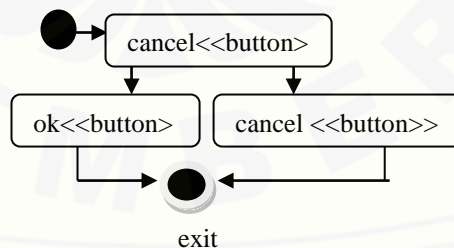


Gambar 4.4 adalah *user interaction* dari menu data produk. Data produk memiliki empat fitur. Diantaranya adalah *add product*, *edit product*, *delete product* dan *detail product*.



Gambar 4.5 Add produk user interaction diagram.

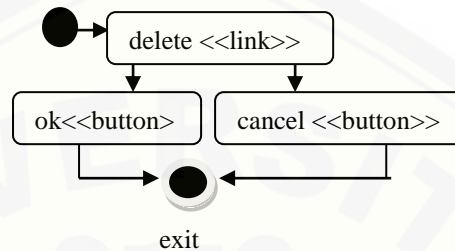
Gambar 4.5 adalah *user intraction* pada *submenu add product*. *Submenu add product* memiliki komponen kolom teks dan tombol. Tombol yang bekerja diantaranya adalah tombol *save* yang berfungsi untuk menyimpan data, tombol *cancel* yang berfungsi untuk kembali pada menu produk, dan tombol *save and back to list* yang berfungsi untuk menyimpan dan kembali kepada menu produk. kolom teks digunakan untuk menyediakan ruang untuk pengisian data produk baru.



Gambar 4.6 Cancel add produk user interaction diagram.

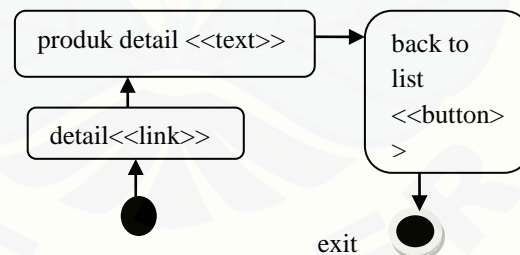
Gambar 4.6 adalah *user interaction* pada *cancel add* pada penambahan produk. Tombol *cancel* mengarah kepada *warning*, dan memiliki dua tombol yaitu

tombol *ok* dan *cancel*. Tombol *ok* mengarah kepada pembatalan penambahan produk. Tombol *cancel* mengarah kepada penghentian pada pembatalan produk.



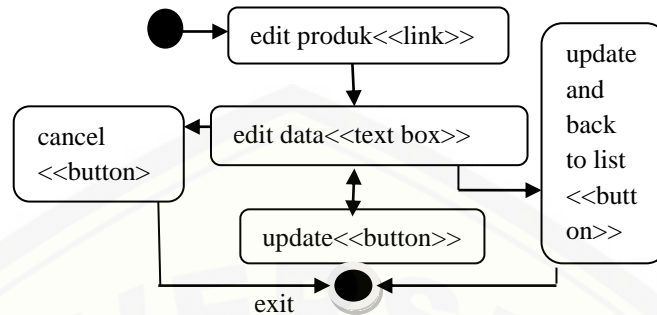
Gambar 4.7 Delete produk user interaction diagram.

Gambar 4.7 adalah *user interaction* pada *delete* produk. Terdapat dua tombol pada *delete* produk yaitu tombol *ok* dan *cancel*. Tombol *ok* berfungsi untuk melanjutkan penghapusan produk. Tombol *cancel* berfungsi untuk membatalkan penghapusan produk.



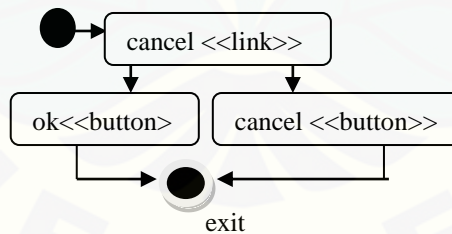
Gambar 4.8 Produk detail user interaction diagram.

Gambar 4.8 adalah *user interaction* pada detail produk. Detail produk berisi penjelasan pada menu produk. Detail produk memiliki satu fitur yaitu fitur *back to list* yang berfungsi untuk kembali pada menu produk.



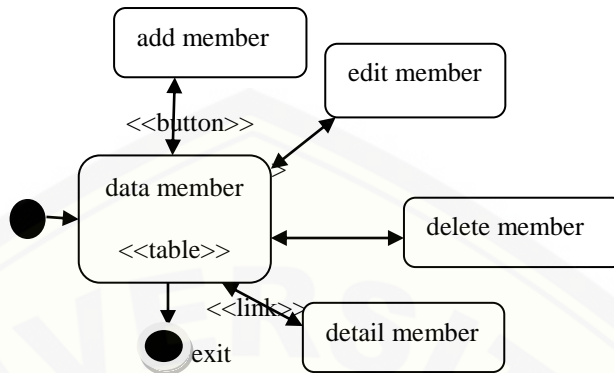
Gambar 4.9 Edit produk user interaction diagram.

Gambar 4.9 adalah *user interaction* pada *edit produk*. *Edit produk* berisi kolom-kolom text yang berfungsi untuk mengisi nilai produk yang akan di *editing*. Terdapat tombol *update* yang berfungsi untuk menyimpan data *update*. Terdapat pula tombol *cancel* yang berfungsi untuk membatalkan *update* dan tombol *update and go back to list* yang berfungsi untuk menyimpan data *update* dan kembali pada menu produk.



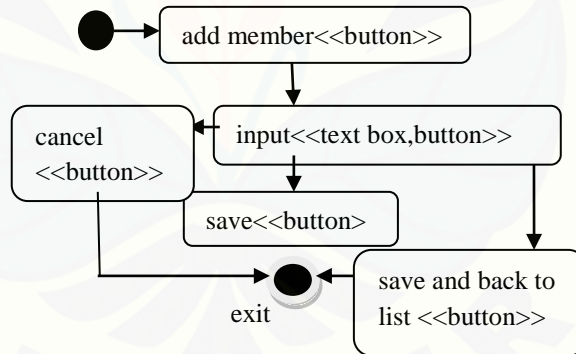
Gambar 4.10 Cancel edit produk user interaction diagram.

Gambar 4.10 adalah *user interaction cancel edit produk*. *Cancel edit produk* berbentuk *warning* yang memiliki dua tombol yaitu tombol *ok* yang berfungsi untuk memberhentikan *edit produk* dan tombol *cancel* yang berfungsi untuk membatalkan pemberhentian *edit* pada produk.



Gambar 4.11 Data member user interaction diagram.

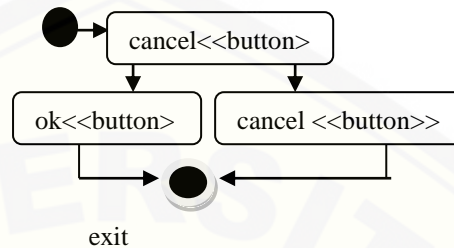
Gambar 4.11 adalah *user interaction* dari menu data member. Data member memiliki empat fitur. Diantaranya adalah *add member*, *edit member*, *delete member* dan *detail member*.



Gambar 4.12 Add member user interaction diagram.

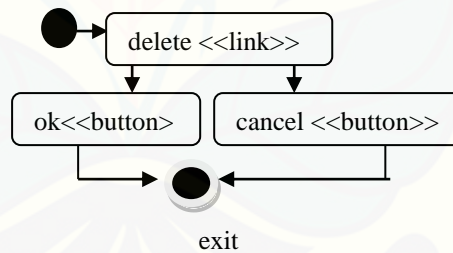
Gambar 4.12 adalah *user intraction* pada *submenu add* produk. *Submenu add* produk memiliki komponen kolom teks dan tombol. Tombol yang bekerja diantaranya adalah tombol *save* yang berfungsi untuk menyimpan data, tombol *cancel* yang berfungsi untuk kembali pada menu produk, dan tombol *save and back*

*to list* yang berfungsi untuk menyimpan dan kembali kepada menu produk. kolom teks digunakan untuk menyediakan ruang untuk pengisian data produk baru.



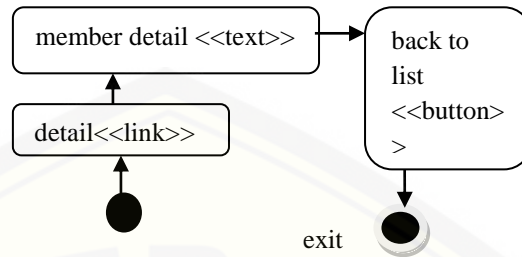
Gambar 4.13 *Cancel add member user interaction diagram.*

Gambar 4.13 adalah *user interaction* pada *cancel add* pada penambahan *member*. Tombol *cancel* mengarah kepada *warning*, dan memiliki dua tombol yaitu tombol *ok* dan *cancel*. Tombol *ok* mengarah kepada pembatalan penambahan *member*. Tombol *cancel* mengarah kepada penghentian pada pembatalan *member*.



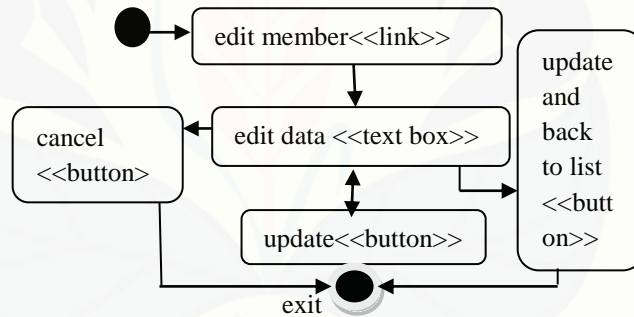
Gambar 4.14 *Delete memnber user interaction diagram.*

Gambar 4.14 adalah *user interaction* pada *delete member*. Terdapat dua tombol pada *delete member* yaitu tombol *ok* dan *cancel*. Tombol *ok* berfungsi untuk melanjutkan penghapusan *member*. Tombol *cancel* berfungsi untuk membatalkan penghapusan *member*.



Gambar 4.15 Member detail user interaction diagram.

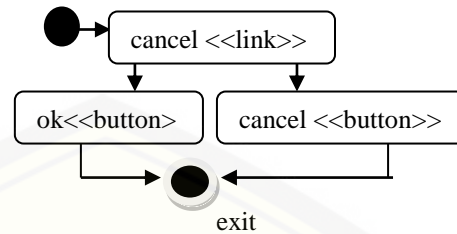
Gambar 4.15 adalah *user interaction* pada detail member. Detail member berisi penjelasan pada menu member. Detail member memiliki satu fitur yaitu fitur *back to list* yang berfungsi untuk kembali pada menu member.



Gambar 4.16 Edit member user interaction diagram.

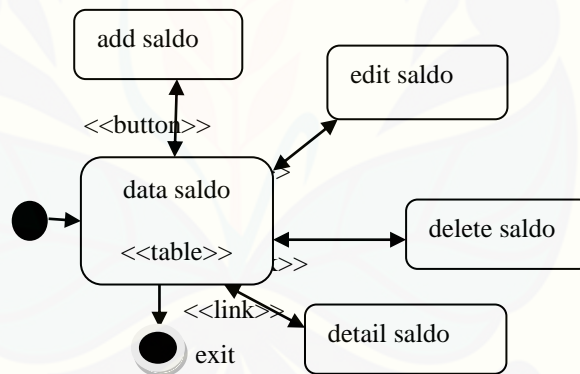
Gambar 4.16 adalah *user interaction* pada *edit member*. *Edit member* berisi kolom-kolom text yang berfungsi untuk mengisi nilai member yang akan di *editing*. Terdapat tombol *update* yang berfungsi untuk menyimpan data *update*. Terdapat pula tombol *cancel* yang berfungsi untuk membatalkan *update* dan tombol *update and go back to list* yang berfungsi untuk menyimpan data *update* dan kembali pada menu member.





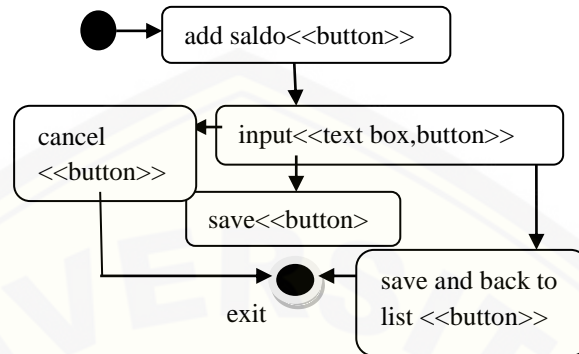
Gambar 4.17 *Cancel edit member user interaction diagram.*

Gambar 4.17 adalah *user interaction cancel edit member*. *Cancel edit member* berbentuk *warning* yang memiliki dua tombol yaitu tombol *ok* yang berfungsi untuk memberhentikan *edit member* dan tombol *cancel* yang berfungsi untuk membatalkan pemberhentian *edit* pada member.



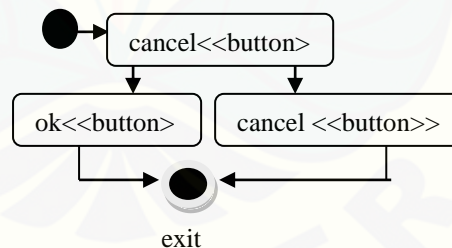
Gambar 4.18 *Data saldo user interaction diagram.*

Gambar 4.18 adalah *user interaction* dari menu data saldo. Data saldo memiliki empat fitur. Diantaranya adalah *add saldo*, *edit saldo*, *delete saldo* dan *detail saldo*.



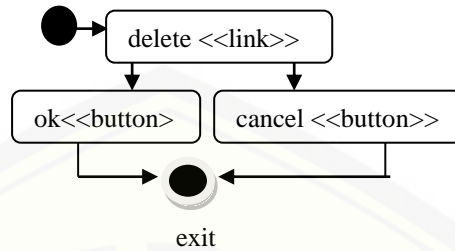
Gambar 4.19 Add saldo user interaction diagram.

Gambar 4.19 adalah *user intraction* pada *submenu add saldo*. *Submenu add saldo* memiliki komponen kolom teks dan tombol. Tombol yang bekerja diantaranya adalah tombol *save* yang berfungsi untuk menyimpan data, tombol *cancel* yang berfungsi untuk kembali pada menu saldo, dan tombol *save and back to list* yang berfungsi untuk menyimpan dan kembali kepada menu saldo. kolom teks digunakan untuk menyediakan ruang untuk pengisian data saldo baru.



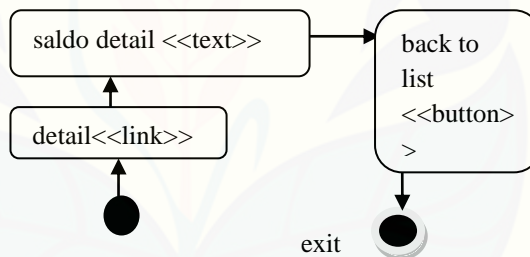
Gambar 4.20 Cancel add saldo user interaction diagram.

Gambar 4.20 adalah *user interaction* pada *cancel add* pada penambahan saldo. Tombol *cancel* mengarah kepada *warning*, dan memiliki dua tombol yaitu tombol *ok* dan *cancel*. Tombol *ok* mengarah kepada pembatalan penambahan saldo. Tombol *cancel* mengarah kepada penghentian pada pembatalan saldo.



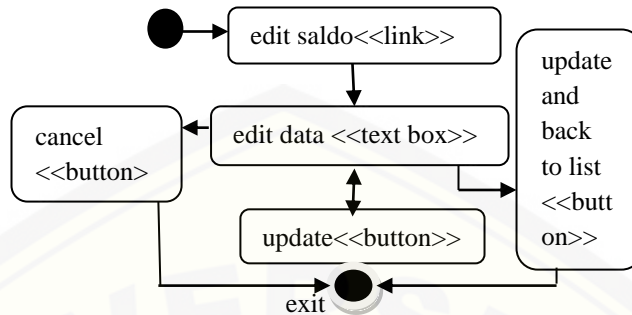
Gambar 4.21 Delete saldo user interaction diagram.

Gambar 4.21 adalah *user interaction* pada *delete* saldo. Terdapat dua tombol pada *delete* saldo yaitu tombol *ok* dan *cancel*. Tombol *ok* berfungsi untuk melanjutkan penghapusan saldo. Tombol *cancel* berfungsi untuk membatalkan penghapusan saldo.



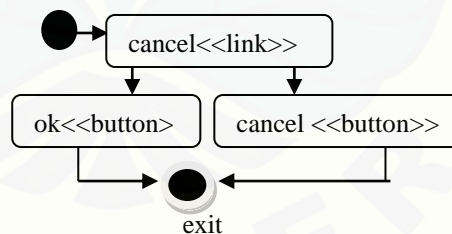
Gambar 4.22 Saldo detail user interaction diagram.

Gambar 4.22 adalah *user interaction* pada detail saldo. Detail saldo berisi penjelasan pada menu saldo. Detail saldo memiliki satu fitur yaitu fitur *back to list* yang berfungsi untuk kembali pada menu saldo.



Gambar 4.23 Edit saldo user interaction diagram.

Gambar 4.23 adalah *user interaction* pada edit saldo. Edit saldo berisi kolom-kolom text yang berfungsi untuk mengisi nilai saldo yang akan di *editing*. Terdapat tombol *update* yang berfungsi untuk menyimpan data *update*. Terdapat pula tombol *cancel* yang berfungsi untuk membatalkan *update* dan tombol *update and go back to list* yang berfungsi untuk menyimpan data *update* dan kembali pada menu saldo.

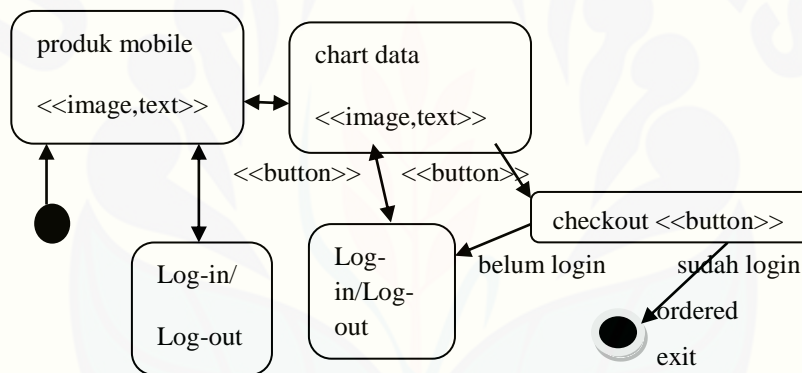


Gambar 4.24 Cancel edit saldo user interaction diagram.

Gambar 4.24 adalah *user interaction cancel edit saldo*. *Cancel edit saldo* berbentuk *warning* yang memiliki dua tombol yaitu tombol *ok* yang berfungsi untuk memberhentikan *edit saldo* dan tombol *cancel* yang berfungsi untuk membatalkan pemberhentian *edit* pada saldo.

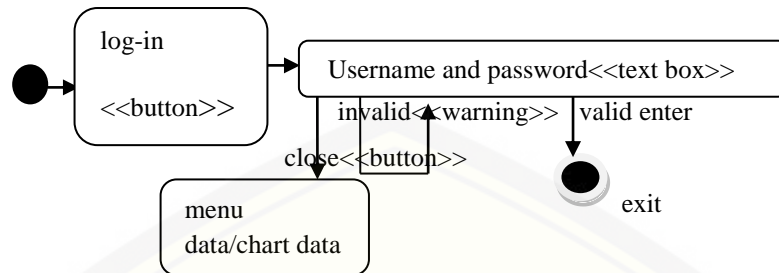
Berikut ini adalah *user interaction diagram* pada aplikasi *mobile*. Terdapat 4 *user interaction* pada aplikasi *mobile*. Pada aplikasi *mobile member* tidak perlu melakukan *login* terlebih dahulu untuk melihat menu. *Member* perlu melakukan *login* untuk melakukan *order*. Halaman keranjang terdapat fitur hapus dan mengubah jumlah produk yang akan dipesan. fitur menu terdapat tombol pesan yang berfungsi untuk menaruh menu yang di klik, di keranjang belanjaan.

Gambar 4.26 sampai dengan 4.29 adalah *user interaction diagram* pada aplikasi *mobile*, pada sistem *mobile voucher food delivery*.



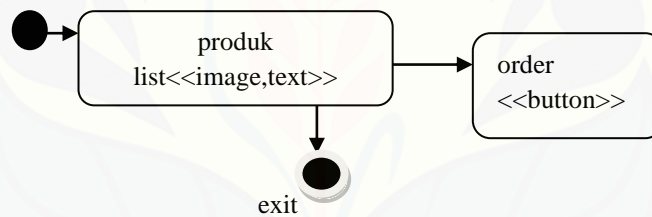
Gambar 4.25 Menu *mobile user interaction diagram*.

Gambar 4.25 adalah *user interaction* pada menu pada aplikasi *mobile*. Pihak *customer* tidak harus *log-in* terlebih dahulu untuk melihat menu yang tersedia. Pihak *customer* harus melakukan *log-in* terlebih dahulu untuk memesan menu yang telah disediakan.



Gambar 4.26 Log-in mobile user interaction diagram.

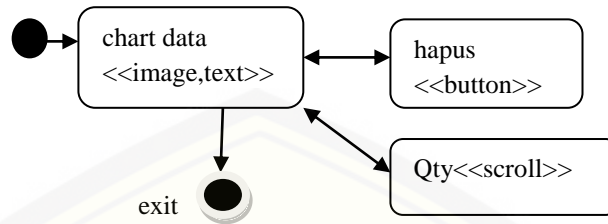
Gambar 4.26 adalah *user interaction* pada fitur *log-in* pada aplikasi *mobile*. Sama dengan aplikasi *web*, *username* dan *password* harus cocok atau benar, apabila ingin melakukan *log-in* dengan benar.



Gambar 4.27 Produk mobile user interaction diagram.

Gambar 4.27 adalah *user interaction* pada menu produk pada aplikasi *mobile*. Produk memiliki komponen gambar dan teks. Teks berfungsi untuk menerangkan gambar yang ada. Gambar berfungsi untuk memperjelas mengenai gambaran aktual produk pada menu.





Gambar 4.28 Chart data user interaction diagram.

Gambar 4.28 adalah *user interaction* pada menu keranjang. Pada menu keranjang item yang telah dipilih dapat diubah jumlahnya, dan dihapus apabila customer hendak membatalkan item yang telah dipilihnya.

### 4.3 User Interface Diagram

Berikut ini akan dibahas mengenai *User interface diagram* dari sistem *mobile voucher food delivery*. Sistem ini memiliki 2 diagram. Diagram pertama untuk pihak *admin*, dan yang kedua untuk pihak *member* atau *customer*. Sistem *mobile voucher food delivery* memiliki bentuk *single page web application*. Jadi beberapa fitur terdapat dalam satu halaman.

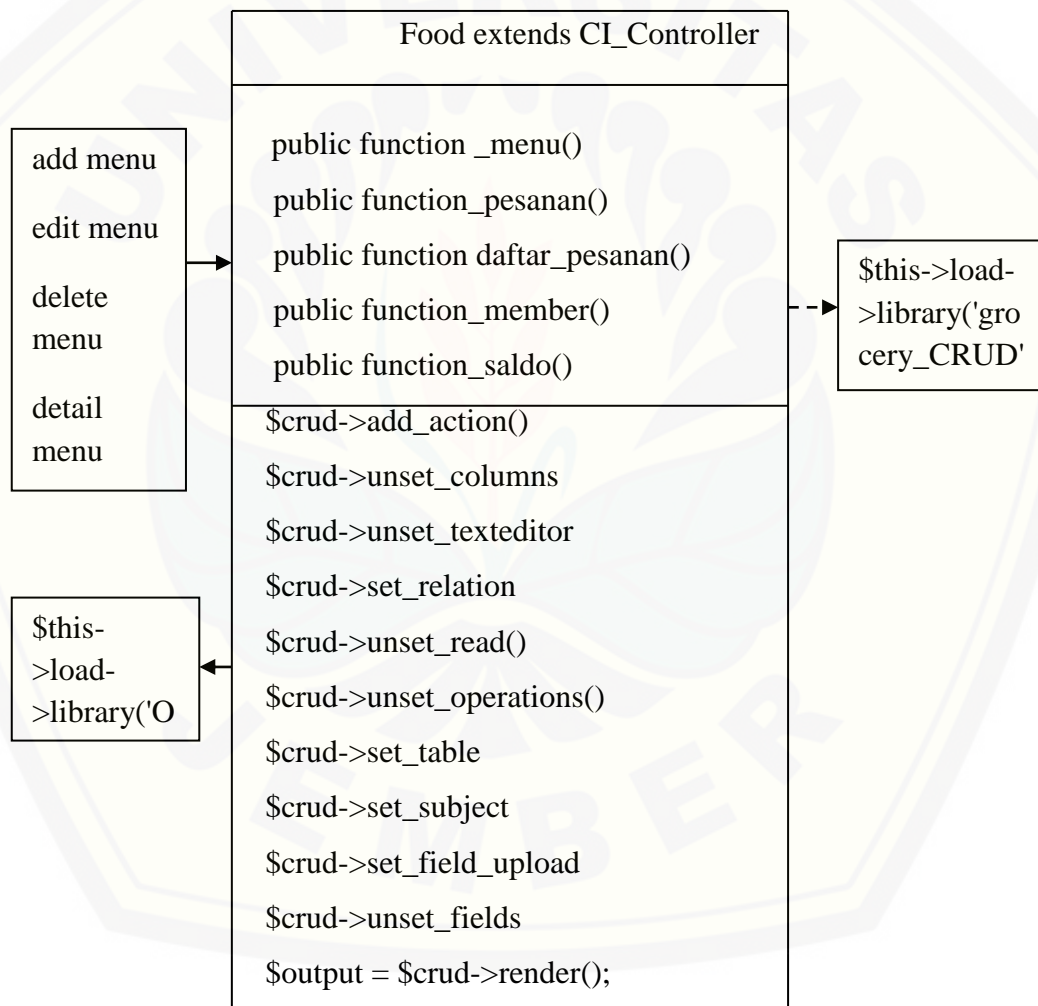
Aplikasi berbasis *web* bagi *admin* memiliki fitur *menu edit*, *member edit*, *order edit*, dan *top-up edit* yang terdapat pada satu halaman. Masing-masing fitur terdapat beberapa persamaan sub-fitur. Sub-fitur seperti *edit*, *delete*, *adding*, dan *detail*.

Aplikasi *mobile* yang digunakan oleh *member* atau *customer* memiliki fitur *log-in* yang optional. Jadi *log-in* hanya berfungsi untuk melihat saldo dan memesan menu makanan yang tersedia. Jadi *member* tidak perlu *log-in* untuk melihat menu makanan yang tersedia. Berikut ini adalah *user interface diagram* dari sistem *mobile voucher food delivery*.

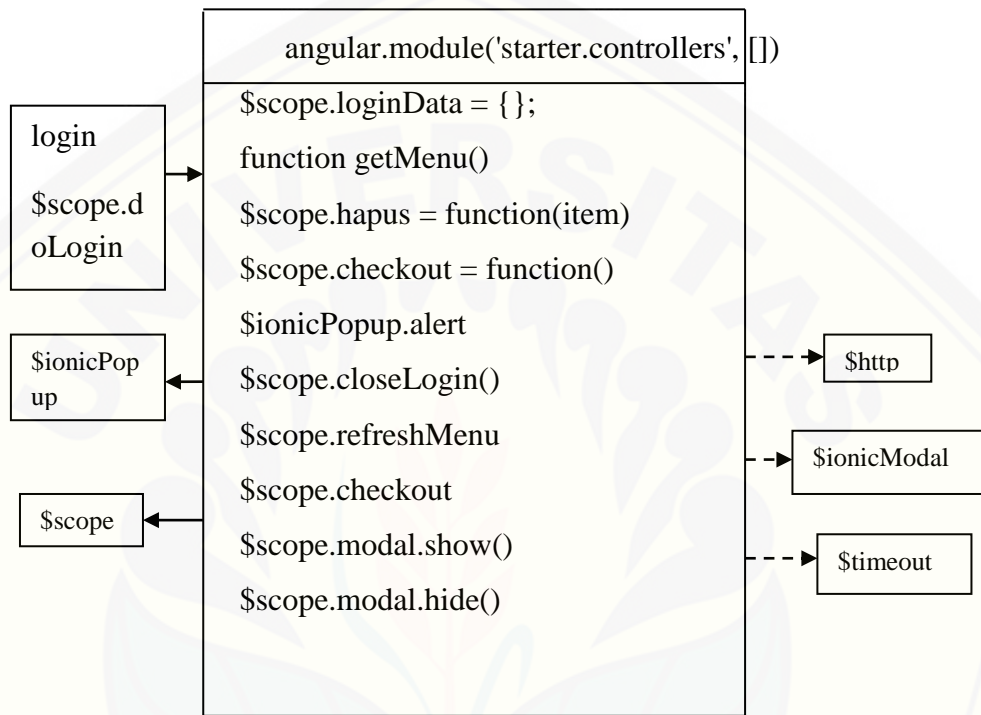


## 4.4 Class Diagram

Berikut ini adalah *class diagram* dari sistem *mobile voucher food delivery*. Bahasa pemrograman yang digunakan adalah javascript, yang mengembangkan *single page application*. Jadi beberapa fitur terdiri dari *function-function* yang saling terhubung di dalam satu kelas. Berikut ini adalah *class diagram* pada sistem *mobile voucher food delivery*.



Gambar 4.30 *Class diagram* aplikasi web sistem *mobile voucher food delivery*.



Gambar 4.31 Class diagram aplikasi mobile sistem mobile voucher food delivery.

#### 4.5 Entity Relationships Diagram

Pemodelan paling awal untuk basis data, yang paling banyak digunakan adalah ERD. Berikut ini adalah *entity relationship diagram* dari sistem *mobile voucher food delivery*.



Gambar 4.32 Entity Relationships Diagram sistem mobile voucher food delivery.

Gambar diatas menjelaskan tentang *entity relationship diagram* pada sistem *mobile voucher food delivery*. Hubungan antara entitas member dengan pesanan adalah *one-to-many* dengan *foreign key* id-member. Satu data member dapat digunakan pada lebih dari satu data pesanan. Hubungan antara entitas member dengan saldo adalah *one-to-many* dengan *foreign key* id-member. Satu data member dapat digunakan pada lebih dari satu data saldo. Hubungan antara entitas pesanan dengan pesanan-menu adalah *one-to-many* dengan *foreign key* id-pesanan. Satu data pesanan dapat digunakan pada lebih dari satu data pesanan-menu. Hubungan antara entitas menu-food dengan pesanan-menu adalah *one-to-many* dengan *foreign key* id-menu-food. Satu data menu-food dapat digunakan pada lebih dari satu data pesanan-menu.

## 4.6 Implementasi Perancangan

Setelah tahap desain perancangan selesai, tahap selanjutnya dalam penelitian ini yaitu tahap pengimplementasian desain perancangan ke dalam bahasa pemrograman. Bahasa pemrograman yang dipakai adalah *PHP* dan *javascript*. Kerangka kerja yang digunakan untuk membantu bahasa pemrograman ini adalah *codeigniter* dan *angular javascript*.

## 4.7 Pengujian

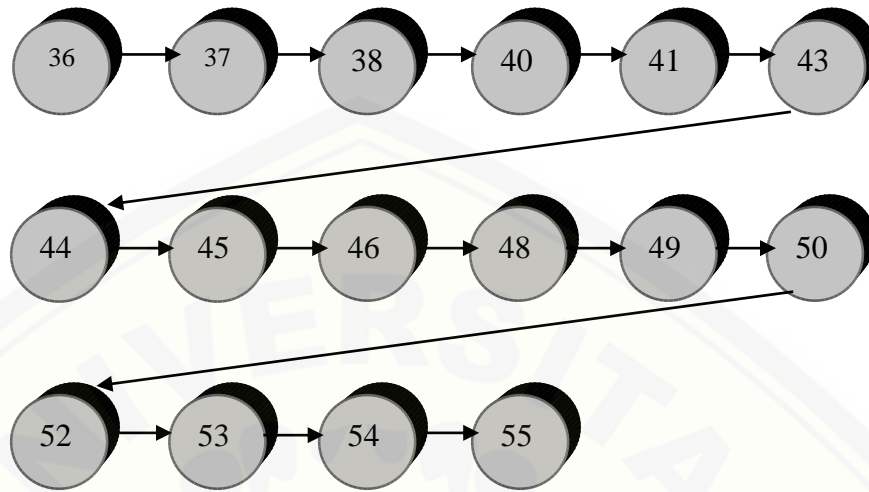
Menurut (Arthur, 1996), *White box testing* adalah pengujian program dengan melihat isi atau kode program. Dalam penelitian ini penulis menggunakan *cyclomatic complexity* untuk mengetahui suatu kompleksitas dari program yang dirancang. Berikut ini adalah kompleksitas dari sistem *mobile voucher food delivery*.

```
35
36 public function menu()
37 {
38     $crud = new grocery_CRUD();
39
40     $crud->set_table('menu_food');
41     $crud->set_subject('Menu');
42
43     $crud->set_field_upload('logo','assets/uploads/foto_menu');
44     $crud->set_field_upload('foto','assets/uploads/foto_menu');
45     $crud->unset_fields('tgl');
46     $crud->unset_texteditor('keterangan','full_text');
47
48     $output = $crud->render();
49     $data['judul'] = "Menu";
50     $data['crumb'] = array( 'Menu' => '' );
51
52     $template = 'admin_template';
53     $view = 'grocery';
54     $this->outputview->output_admin($view, $template, $data, $output);
55 }
56
```

Gambar 4.33 Coding fungsi menu aplikasi web.

Potongan kode diatas adalah potongan kode dari aplikasi web pada sistem *mobile voucher food delivery*. Kode diatas merupakan kode pada bagian menu makanan. Berikut ini adalah *Complexity* dari kode tersebut.





Gambar 4.34 Complexity dari coding pada gambar 4.33.

Hasil kompleksitas dari alur diatas bernilai 1. Nominal tersebut didapatkan dari mengurangi jumlah tanda panah, dengan jumlah lingkaran, dijumlahkan dengan 2. Jadi jumlah tanda panah dikurangi dengan jumlah lingkaran bernilai -1. Kemudian nilai tersebut dijumlahkan dengan 2, maka akan memiliki hasil akhir yang bernilai 1.

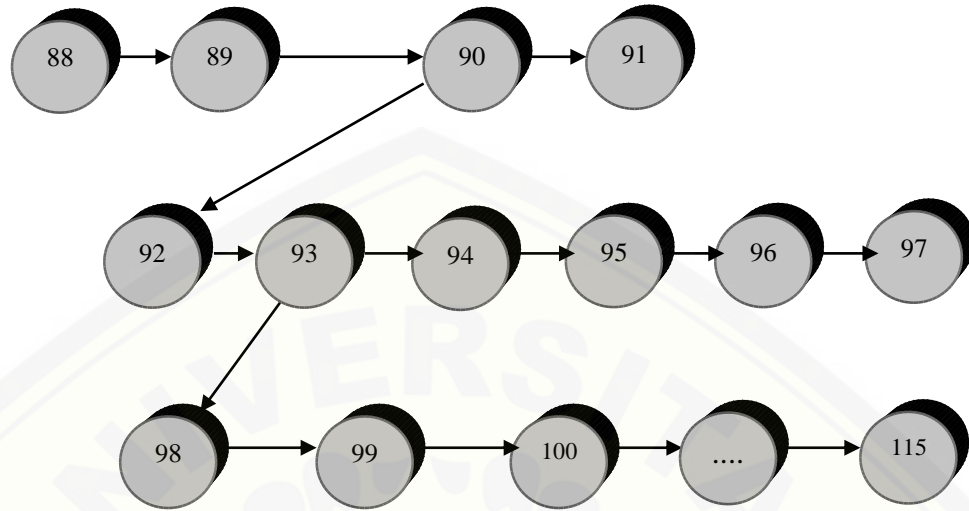
```

88     $scope.checkout = function() {
89         keranjang = $scope.keranjang.length;
90         if (!$scope.islogin) {
91             $scope.modal.show();
92         }else{
93             if (keranjang == 0) {
94                 $ionicPopup.alert({
95                     title: 'Keranjang kosong!',
96                     template: 'Silakan pesan dahulu'
97                 });
98             }else{
99                 var idPemesanan = new Date().getUTCMilliseconds();
100                var idMember = $scope.sisa.id_member;
101                var menu = $scope.keranjang;
102
103                $http.get("http://localhost/food/food_delivery_web/in
104                    .success(function(selesai){
105                        angular.forEach(menu, function(value, key) {
106                            qty = '1'; //Tahap Pengembangan
107                            $http.get("http://localhost/food/food_delivery_
108                                });
109                            $ionicPopup.alert({
110                                title: 'Terima Kasih',
111                                template: 'Pesanan Akan Kami Proses'
112                            });
113                            $scope.keranjang = [];
114                        });
115                    });

```

Gambar 4.35 Potongan kode aplikasi *mobile*.

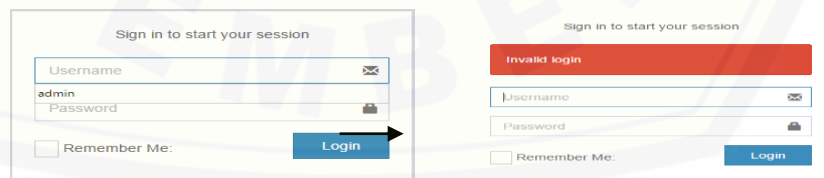
Potongan kode diatas adalah potongan kode dari aplikasi *mobile* pada sistem *mobile voucher food delivery*. Kode diatas adalah kode pada bagian *checkout menu* atau *order menu*. Keranjang pemesanan tidak dapat dikosongkan dan diperlukan *login* untuk memesan menu makanan yang tersedia. Berikut ini adalah kompleksitas dari kode-kode tersebut.



Gambar 4.36 Complexity dari coding pada gambar 4.35.

Hasil kompleksitas dari alur diatas juga bernilai 1. Alur program yang memiliki nilai kompleksitas 1, maka program tersebut dapat dikatakan sederhana.

Menurut (Rosa, 2011) , *black box testing* menguji perangkat lunak dari segi spesifikasi fungsional, tanpa menguji desain dan kode program. Pengujian dimaksudkan untuk mengetahui apakah fungsi-fungsi, masukan, dan keluaran dari perangkat lunak, berjalan sesuai dengan spesifikasi yang dibutuhkan. Berikut ini adalah gambar pengujian sistem *mobile voucher food delivery* dengan *black box testing*.



Gambar 4.37 Pengujian pada kesalahan *log-in* (sukses).

[+ Add Menu](#)

Show  entries

No. ↓	Nama menu ↑	Logo ↑	Foto ↑	Keterangan ↑	Harga ↑	Status ↑	Tgl
1	Rendang			Masakan kas padang	10000	TERSEDIA	08/06/2015 - 14:14
2	Bakso Goreng			Camilan bakso	0	TERSEDIA	21/06/2015 - 09:15
3	Ramen			Masakan jepang	15000	TIDAK TERSEDIA	08/06/2015 - 14:14
4	bakpia Pecah Seribu			Pkoe maknyus westo	10000000	TERSEDIA	21/06/2015 - 09:45

penambahan data makanan pada menu.

[+ Add Menu](#)

Nama menu

Logo [Upload a file](#)

Foto [Upload a file](#)

Keterangan

Harga

Status

[Save](#) [Save and go back to list](#) [Cancel](#)


penambahan menu sukses.

Your data has been successfully stored into the database. [Edit Menu](#) or [Go back to list](#)


Gambar 4.38 Pengujian pada penambahan menu (sukses).


No.	Nama menu	Logo	Foto	Keterangan	Harga	Status	Tgl	Actions
1	Rendang			Masakan kas padang	10000	TERSEDIA	08/06/2015 - 14:14	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Delete</a>
2	Bakso Goreng			Camilan bakso	1000	TERSEDIA	17/06/2015 - 07:15	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Delete</a>
3	Ramen			Masakan jepang	15000	TIDAK TERSEDIA	08/06/2015 - 14:14	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Delete</a>
4	Menu Baru			Keterangan makanan	9000	TERSEDIA	09/06/2015 - 11:13	<a href="#">Detail</a> <a href="#">Edit</a> <a href="#">Delete</a>

mengklik edit pada menu.

 **Edit Menu**

**Nama menu**

**Logo**  [delete](#)

**Foto**  [delete](#)

**Keterangan**

**Harga**

**Status**

update data makanan sukses.

Your data has been successfully updated. [Go back to list](#)

Gambar 4.39 Pengujian pada *update* data makanan.

Member

Export Print

Show 10 entries Search:

No.	Uername	Nama lengkap	No hp	Alamat	Password	Actions
1	member	Member	85	Jember	*****	History Edit Delete
2	edo	edo	89	jember	*****	History Edit Delete

Showing 1 to 2 of 2 entries Previous 1 Next

Edit Member

Username member

Nama lengkap Member

No hp 85

Alamat Jember

Password .....

Update changes Update and go back to list Cancel

update data member. menghapus data member.

update data member.

Your data has been successfully updated. [Go back to list](#)

menghapus data member sukses.

Your data has been successfully deleted from the database.

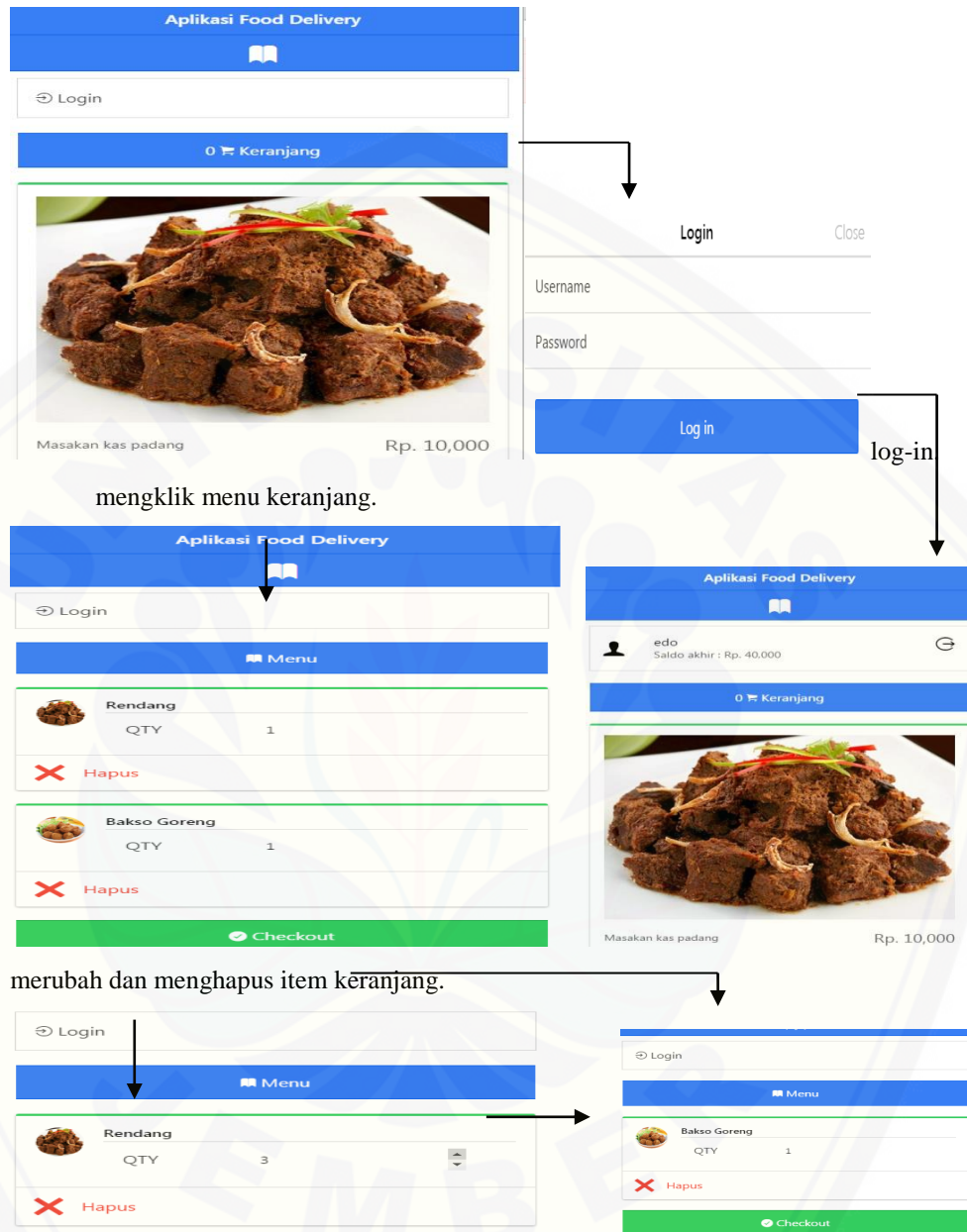
AlertifyJS

Are you sure that you want to delete this record?

OK CANCEL

Gambar 4.40 Pengujian pada *update* dan hapus data *member*.





Gambar 4.41 Pengujian pada aplikasi *mobile* (log-in dan menu keranjang sukses).