BAB II

TINJAUAN PUSTAKA

II.1. Sistem Informasi

Sistem merupakan serangkai bagian yang saling tergantung dan bekerja sama untuk mencapai tujuan tertentu. Suatu sistem pasti tersusun dari sub-sub sistem yang lebih kecil dan juga saling tergantung dan bekerja sama untuk mencapai tujuan. Sebagai contoh, sistem administrasi universitas terdiri dari sistem dari sub-sub sistem administrasi fakultas dan sub-sub sistem administrasi jurusan fakultas.

Sistem informasi yang kadang kala disebut sebagai sistem pemrosesan data, merupakan sistem buatan manusia yang biasanya terdiri dari sekumpulan komponem, baik manual ataupun berbasis komputer yang terintegrasi untuk mengumpulkan, menyimpan dan mengelola data serta menyediakan informasi kepada pihak yang berkepentingan sebagai pemakai informasi tersebut (Anastasia Diana; 2011 : 3).

II.2. Sistem Informasi Geografis

Menurut Hadi Muhammad (2013 : 2) Sistem Informasi Geografis **(***Geographic Information System*) adalah sistem informasi khusus yang mengelola data yang memiliki informasi spasial (bereferensi keruangan). Atau dalam arti yang lebih sempit, adalah sistem komputer yang memiliki kemampuan untuk membangun, menyimpan, mengelola dan menampilkan informasi berefrensi

geografis, misalnya data yang diidentifikasi menurut lokasinya, dalam sebuah database. SIG menghubungkan sekumpulan unsur-unsur peta dengan atribut-atributnya di dalam satuan yang dikenal sebagai "*layers*". Contoh *layers* misalnya sungai, bangunan, jalan, laut, batas-batas administrasi, hutan dan lain-lain. Kumpulan dari *layers* ini membentuk basis data SIG. Dengan demikian, perancangan basisdata merupakan hal yang penting dalam SIG untuk menentukan efektifitas dan efisiensi proses-proses masukan, pengelolaan, dan keluaran SIG.

Menurut Hersa Farida Qoriani (2012 : 3) GIS atau sistem informasi berbasis pemetaan dan geografi adalah sebuah alat bantu manajemen berupa informasi berbantuan komputer yang terkait dengan sistem pemetaan dan analisis terhadap segala sesuatu, serta peristiwa-peristiwa yang terjadi dimuka bumi. Teknologi GIS mengintegrasikan operasi pengolahan data berbasis database yang biasa digunakan, seperti pengambilan data berdasarkan kebutuhan serta analisis statistic dengan menggunakan visualisasi yang khas serta berbagai keuntungan yang mampu ditawarkan melalui analisis geografis melalui gambar-gambar tertentu.

Konsep GIS telah diperkenalkan di Indonesia sejak pertengahan tahun 1980-an., dan kini telah dimanfaatkan di berbagai bidang baik negeri maupun swasta. Kemampuan dasar dari GIS adalah mengintegrasikan berbagai operasi basis data seperti *query*, menganalisisnya, dan menyimpan serta menampilkannya dalam bentuk pemetaan berdasarkan letak geografisnya. Inilah yang membedakan GIS dengan sistem informasi lain. Komponen GIS terdiri atas *hardware*, *software*,

data, dan user. Dengan adanya GIS diharapkan tersedia informasi yang cepat, benar dan akurat tantang keadaan di lingkungannya.

II.3. Xampp

Xampp merupakan paket *Php* berbasis *open source*. *Xampp* membantu memudahkan dalam mengembangkan aplikasi berbasis *Php*. *Xampp* mengkombinasikan beberapa paket *software* berbeda kedalam satu paket. Adapun lisensi masing-masing paket *software* tersebut dapat ditemukan didirektori *\xampp\licence*. *Xampp* menyediakan antar muka *control panel* tersendiri yang dapat digunakan untuk menjalankan semua *service* (paket *software* pendukung) yang telah terinstal. Pada sistem operasi windows, *control panel* dapat diakses melalui menu [*Start*] [Program] [*Apachefriends*] [*xampp*] [*control xampp server panel*]. Pada *web server* (lokal komputer, tidak di *server internet* sesungguhnya) pada *Xampp*, akan menyediakan satu *folder* kerja yang bernama htdocs. Pada paket ini, *folder* kerja tersebut dapat ditemukan pada sub*folder* C:\..*Xampp* (sesuai lokasi dimana menyimpan hasil instalasinya). (Meiska Firstiara Maudi; 2014:102)

II.4. Java Script

JavaScript merupakan bahasa pemrograman *web client side*. Kalau *html* digunakan untuk membuat halaman *web* statis, maka JavaScript digunakan untuk membuat halaman *web* yang interaktif dan dinamis. Karena sebagai bahasa pemrograman, *JavaScript* dapat digunakan untuk membuat aplikasi matematis,

efek animasi sederhana, bahkan juga untuk membuat game. Hampir *browser* yang ada saat ini sudah *support JavaScript*. Dokumen *JavaScript* dapat dibuat dengan *text editor* biasa, seperti: *Notepad, Wordpad, Notepad++*, dll, yaitu dengan menyimpannya kedalam format *.js. (Meiska Firstiara Maudi; 2014:103)

II.5 Adobe Photoshop

Adobe Photoshop adalah suatu perangkat lunak yang canggih yang dapat digunakan untuk membuat, menyunting dan memanipulasi tampilan termasuk mengoreksi warna dan memberi efek tampilan atas sebuah gambar atau photo, hasil dari program ini merupakan sebuah gambar atau *image*, didalam komputer grafis terbagi menjadi dua kelompok yaitu Gambar *Bitmap* dan Gambar *Vektor*.

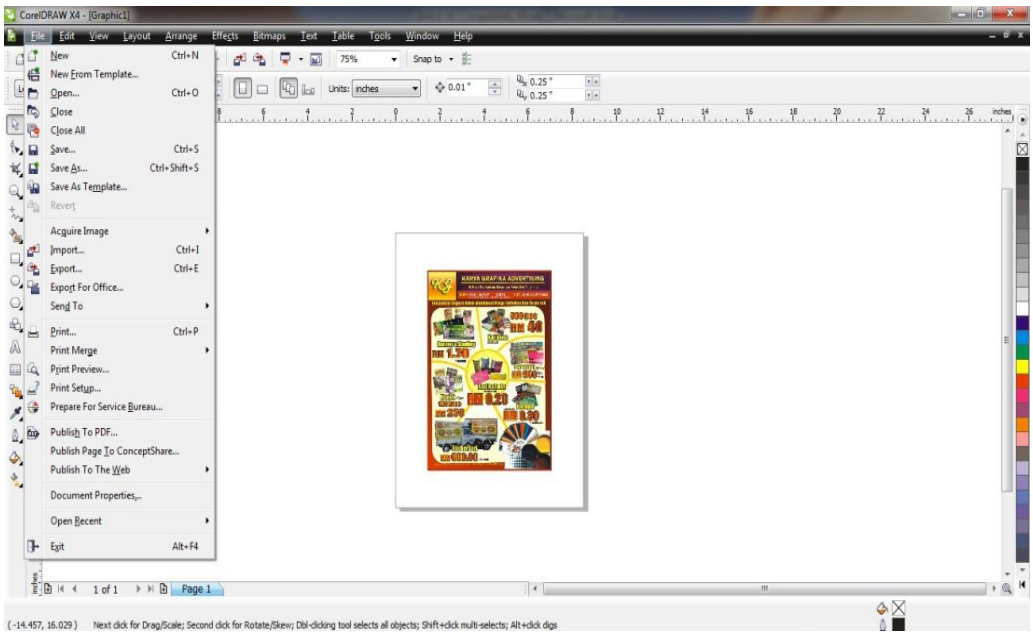
Dengan kemampuan pengolahan bitmap yang sangat baik, menjadikan *Adobe Photoshop* menjadi standar yang umum digunakan didalam pengolahan objek bitmap. *Adobe Photoshop* menyimpan beberapa kemampuan yang sangat baik untuk membuat gambar selayaknya menggunakan aplikasi berbasis vektor. Akan tetapi hal tersebut membutuhkan pemahaman konsep dasar pembentukan kurva vektor yang tidak dapat ditinggalkan oleh aplikasi dalam mengolah bitmap seperti *photoshop*. Konsep dasar yang harus dipahami adalah : manajemen *layer*, pembuatan *path*, dan seleksi. *Toolbox* berfungsi sebagai tombol pengganti perintah yang dipergunakan untuk mempercepat pekerjaan. Nama-nama *toolbox terdiri atas Marquee tools, Lasso tools, Magic Wand tool, Move tool, Crop tool, Slice tool, Healing brush tool, Pencil tool, Clone Stamp tool, History Brush tool, Eraser tool, Paint Bucket tool, Blur tool, Path Component Selection tool, Type*

tool, Pen tool, Zoom tool, Eyedroper Hand tool, dan sebagainya. (Ridwan Sukmana; 2015; 35-36)

II.6. Coreldraw X4

Coreldraw adalah aplikasi design grafis yang digunakan untuk membuat berbagai macam design seperti logo, kartu nama, kalender, poster, stiker dan lain-lain yang terkenal dalam dunia digital dan menyediakan berbagai fasilitas yang dapat memudahkan dalam proses desain. *Coreldraw* juga seperti *Adobe Photoshop* yang memiliki *Toolbox* yang berfungsi sebagai tombol pengganti perintah yang dipergunakan untuk mempercepat pekerjaan

Nama-nama tersebut *Shape Tool, Crop tool, Zoom tool, Freehand tool, Smart Fill tool, Rectangel tool, Elipse tool, Rectangle tool, Polygon tool, Basic Shapes tool, Text tool, Table tool, Pararel Dimension tool, .Straight Line Connector tool, Blend tool, Color Eyedropper tool, Gradient tool, Eyedropper tool, Outline tool, Fill tool, Interactive Fill tool*, dan sebagainya. Di bawah ini adalah gambar jendela halaman kerja program *CorelDraw X4*. (Ridwan Sukmana; 2015; 37-38)



Gambar II.1 Gambar Jendela Halaman Kerja Program *CorelDraw X4*

Sumber : (Ridwan Sukmana; 2015; 37-38)

Keterangan :

1. *Title Bar*

   Baris judul berisi nama untuk judul program yang sedang aktif.

2. *Menu Bar*

   Baris menu berisi barisan perintah berupa menu yang terdiri dari *file, edit*, *object, type* dan lain-lain.

3. Lembar kanvas

   Berisi piranti untuk mempermudah dalam pengerjaan memanipulasi gambar.

4. *Tool box*

   Digunakan sebagai lembar kerja atau penempatan obyek teks dan gambar.

II.7. *Unified Modeling Language* (UML)

Unified Modeling Language (UML) adalah suatu alat untuk memvisualisasikan dan mendokumentasikan hasil analisa dan desain yang berisi sintak dalam memodelkan sistem secara *visual* (Braun*, et. al*.2011). Juga merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem *software* yang terkait dengan objek (Whitten, *et. al*. 2004)

UML merupakan salah satu alat bantu yang sangat handal dalam bidang pengembangan sistem berorientasi objek karena UML menyediakan bahasa pemodelan *visual* yang memungkinkan pengembang sistem membuat *blue print* atas visinya dalam bentuk yang baku. UML berfungsi sebagai jembatan dalam berkomunikasikan beberapa aspek dalam sistem melalui sejumlah elemen grafis yang bisa dikombinasikan menjadi diagram. UML mempunyai banyak diagram yang dapat mengakomodasi berbagai sudut pandang dari suatu perangkat lunak yang akan dibangun.(Yuni Sugiarti; 2013:36-37)

Dengan menggunakan UML, kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemograman apapun. Tetapi karena UML juga menggunakan dan operation dalam konssep dasarnya, maka ia lebih cocok untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NET. Walaupun demikian UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C.

Seperti bahasa-bahasa lainnya, UML mengidentifikasikan notasi dan syntax/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan. Notasi UML terutama diturunkan dari 3 notasi yang telah ada sebelumnya: Grady Booch OOD (*Object-Oriented Software Design*), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*) (Yuni Sugiarti;2013:34).

Blok pembangunan utama UML adalah diagram. Beberapa diagram ada yang rinci (jenis *timming diagram*) dan lainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembang sistem berorientasi objek menggunakan bahasa model untuk menggambarkan, membangundan mendokumentasikan sistem yang mereka rancang. UML memungkinkan para anggota team untuk bekerja sama dengan bahasa model yang sama dengan mengaplikasikan beragam sistem. Intinya UML merupakan alat komunikasi yang konsisten dalam mendukung para pengembang sistem saat ini. Di bawah ini adalah komponen-komponen pada UML (*Unified Modeling Language*), yaitu:

II.7.1. *Use Case* Diagram

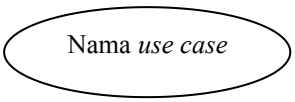
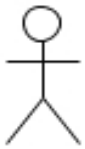
Dalam membuat sebuah sistem, langkah awal yang perlu dilakukan adalah menentukan kebutuhan. Terdapat dua jenis kebutuhan, yaitu kebutuhan fungsional dan kebutuhan nonfungsional. Kebutuhan fungsional adalah kebutuhan pengguna dan stakeholder sehari-hari yang akan dimiliki oleh sistem, dimana kebutuhan ini akan digunakan oleh pengguna *stakeholder*. Sedangkan kebutuhan nonfungsional

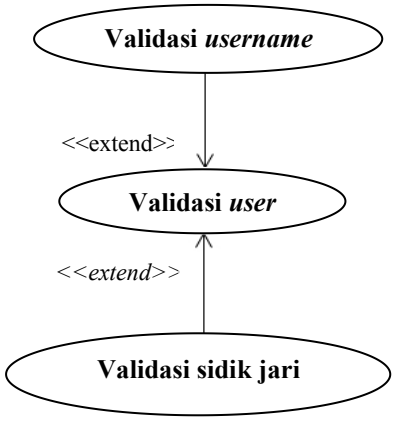
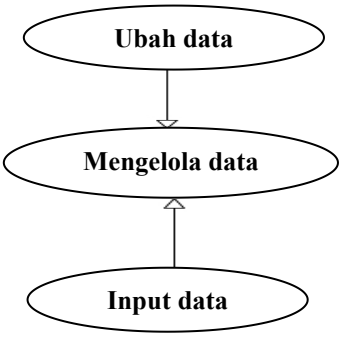
adalah kebutuhan yang memperhatikan hal-hal berikut yaitu performansi, kemudahan dalam menggunakan sistem, kehandalan sistem, keamanan sistem, keuangan, legalitas, dan operasional.

Kebutuhan fungsi akan digambarkan melalui sebuah diagram yang dinamakan diagram use case. *Use case* diagram atau diagram use case merupakan pemodelan untuk menggambarkan kelakuan (*behavior*) sistem yang akan dibuat. Diagram *use case* mendeskripsikan sebuah interaksi antar satu atau lebih *actor* dengan sistem yang akan dibuat. Dengan pengertian yang cepat, diagram *use case* digunakan untuk mengetahui fungsi apa saja yang ada didalam sebuah sistem dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Terdapat beberapa simbol dalam menggambarkan diagram *use case*, yaitu *use case*, *actor* dan relasi. Hal perlu diingat mengenai diagram *use case* adalah diagram *use case* bukan menggambarkan tampilan antarmuka (*user interface*), arsitektur dari sistem, kebutuhan nonfungsional, dan tujuan performansi. Sedangkan untuk penamaan *use case* adalah nama didefinisikan sesimpel mungkin, dapat dipahami dan menggunakan kata kerja (Yuni Sugiarti; 2013:41).

Berikut ini pada tabel II.1. adalah simbol-simbol yang ada pada diagram *use case*:

Tabel II.1. Simbol – Simbol *Use Case* Diagram

Simbol	Deskripsi
Use case  	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor, biasanya dinyatakan dengan menggunakan kata kerja diawal diawal frase nama *use case*
Aktor / *actor*  **nama aktor**	orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari *actor* adalah gambar orang, tapi *aktor* belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal nama *aktor*.
Asosiasi/ *association* ———	Komunikasi antara *actor* dan *use case* yang berpartisipasi pada *use case* atau *use case* memiliki interaksi dengan aktor
Ekstensi / *extend* <<*extend*>> ——→	Relasi *use case* tambahan kesebuah *use case* dimana *use case* yang ditambahkan dapat berdiri sendiri walau tanpa *use case* tambahan itu: mirip dengan prinsip *inheritance* pada pemograman berorientasi objek; biasanya *use*

	case tambahan memiliki nama depan yang sama dengan *use case* yang ditambahkan, misal  Arah panah mengarah pada *use case* yang ditambahkan; biasanya *use case* yang menjadi *extend*-nya merupakan jenis yang sama dengan *use case* yang menjadi induknya.
Generalisasi/*generalization* 	Hubungan generalisasi dan spesialisasi (umum – khusus) antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari lainnya, misalnya:  Arah panah mengarah pada *use case* yang menjadi generalisasinya (umum).

Menggunakan / *include* / *uses*  <<*include*>> <<*uses*>> 	Relasi *use case* tambahan ke sebuah *use case* dimana *use case* yang ditambahkan memerlukan *use case* ini untuk menjalankan fungsinya atau sebagai syarat *use case* ini. Ada dua sudut pandang yang cukup besar mengenai *include* di *use case* : *include* berarti *use case* yang ditambahkan akan selalu dipanggil saat *use case* tambahan dijalankan. *include* berarti *use case* yang tambahan akan selalu melakukan pengecekan apakah *use case* yang ditambahkan telah dijalankan sebelum *use case* tambahan dijalankan.

Sumber : (Rosa A.S, M.Shalahuddin; 2013; 155-158).

II.7.2. *Class Diagram* (Diagram Kelas)

Diagram kelas atau *class diagram* menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. kelas memiliki apa yang disebut atribut dan metode atau operasi.

1. Atribut merupakan variabel-variabel yang dimiliki oleh suatu kelas.
2. Atribut mendeskripsikan property dengan sebaris teks didalam kotak kelas tersebut.
3. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

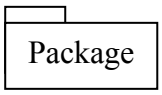
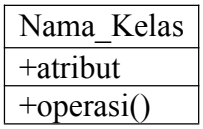
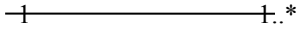
Diagram kelas mendeskripsikan jenis-jenis objek dalam sistem dan berbagai hubungan statis yang terdapat diantara mereka. Diagram kelas juga menunjukkan properti dan operasi sebuah kelas dan batasan-batasan yang terdapat dalam hubungan-hubungan objek. Diagram kelas menggambarkan struktur dan deskripsi class, package dan objek beserta hubungan satu sama lain seperti containment, perwarisan, asosiasi, dan lain-lain (Yuni Sugiarti;2013:57).

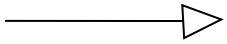
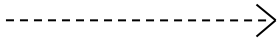
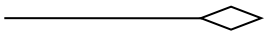
Kelas memiliki tiga area pokok :

1. Nama
2. Atribut
3. Operasi

Berikut adalah simbol-simbol yang ada pada diagram kelas :

Tabel II.2. Simbol-Simbol Diagram Kelas

Simbol	**Deskripsi**
Package 	Package merupakan bungkusan dari satu kelas atau lebih
Operasi 	Kelas pada struktur system
Antarmuka / *interface*  Nama *interface*	Sama dengan konsep *interface* dalam pemrograman berorientasi objek
Asosiasi / *association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*

Asosiasi berarah/ *Directed association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *multiplicity*
Generalisasi 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
Ketergantungan 	Relasi antar kelas dengan makna Kebergantungan antar kelas
Agregasi / *aggregation* 	Relasi antar kelas dengan makna Semua bagian (*whole part*)

Sumber : (Yuni Sugiarti;2013:59).

II.7.3. *Activity Diagram* (Aktivitas)

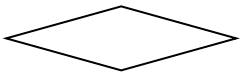
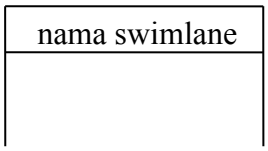
Diagram aktivitas atau *activity diagram* menggambarkan *workflow*(aliran kerja) atau aktivias dari sebuah sistem atau proses bisnis. *Activity diagram* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, decision yang mungkin terjadi, dan bagaimana mereka berakhir.

Activity Diagram merupakan *state* diagram khusus, dimana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tdak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem)

secara eksak, tetapi lebih menggambarka proses- proses dan jalur-jalur aktivitas dari level atas secara umum (Yuni Sugiarti;2013:75).

Berikut adalah tabel II.3. yang menggambarkan simbol-simbol yang digunakan pada diagram aktivitas :

Tabel II.3. Simbol *Activity Diagram*

Simbol	Deskripsi
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas memiliki sebuah status awal.
Aktivitas 	Aktivitas yang dilakukan sistem, aktivitas biasanya diawali dengan kata kerja.
Percabangan / *decision* 	Asosiasi percabangan dimana jika ada pilihan aktivitas lebih dari satu.
Penggabungan / *join*	Asosiasi penggabungan dimana lebih dari satu aktivitas digabungkan menjadi satu.
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas memiliki sebuah status akhir.
Swimlane 	Memisahkan organisasi bisnis yang bertanggungjawab terhadap aktivitas yang terjadi.

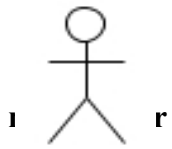
Sumber : (Rosa A.S, M.Shalahuddin; 2013; 161-163).

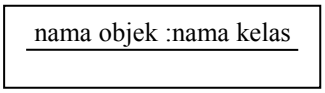
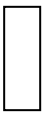
II.7.4. *Sequence* Diagram

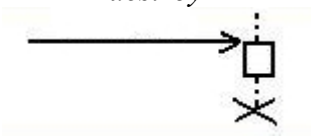
Diagram sekuen menggambarkan kelakuan/perilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan message yang dikirimkan dan diterima antar objek.

Banyaknya diagram *sequence* yang digambarkan adalah sebanyak pendefenisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefenisikan interaksi jalanya pesan sudah dicakup pada diagram sekuen sehingga semakin banyak *use case* didefenisikan maka diagram sekuen yang harus dibuat juga semakin banyak (Yuni Sugiarti;2013:69). Berikut adalah tabel II.4. yang menerangkan simbol-simbol yang ada pada diagram sekuen:

Tabel II.4. Diagram *Sequence*

Simbol	Deskripsi
Aktor 	orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat di luar sistem informasi yang akan dibuat itu sendiri, jadi walaupun simbol dari *actor* adalah gambar orang, tapi *actor* belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal nama *actor*.

Garis hidup / *lifeline* 	Menyatakan kehidupan suatu objek.
Objek 	Menyatakan objek yang berinteraksi pesan.
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi, semua yang terhubung dengan waktu aktif ini adalah tahapan yang dilakukan didalamnya.
Pesan tipe create <<*create*>> 	Menyatakan suatu objek membuat objek yang lain, arah panah mengarah pada objek yang dibuat
Pesan tipe call 1 : nama_metode{} 	Menyatakan suatu objek memanggil operasi / metode yang ada pada objek lain atau dirinya sendiri.
Pesan tipe send 1 : masukan 	Menyatakan bahwa suatu objek mengirimkan data / masukan / informasi ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
Pesan tipe return 1 : keluaran 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode menghasilkan suatu kembalian ke objek tertentu, arah panah mengarah pada objek yang

	menerima kembalian.
Pesan tipe destroy <<*destroy*>> 	Menyatakan suatu objek mengakhiri hidup objek yang lain, arah panah mengarah pada objek yang diakhiri, sebaiknya jika ada create maka ada *destroy*.

Sumber : (Rosa A.S, M.Shalahuddin; 2013; 165-167).

II.8. Kamus Data

Menurut Edhy Sutanta (2011 : 25) Sebelum memperoleh defenisi formal basis data, kita akan mencoba memahaminya secara sederhana terlebih dahulu. Istilah basis data tersusun atas dua suku kata, yaitu basis dan data (basis data = basis + data). Dalam sistem bilangan biner, kita dapat menuliskan beberapa contoh bilangan sebagai berikut.

0 → sama dengan 0 dalam sistem bilangan desimal

1 → sama dengan 1 dalam sistem bilangan desimal

10 → sama dengan 2 dalam sistem bilangan desimal

11 → sama dengan 3 dalam sistem bilangan desimal

100 → sama dengan 4 dalam sistem bilangan desimal

II.9. Entity Relationship Diagram

Menurut Edhy Sutanta (2011 : 91) *Entity Relationship Diagram/ER_M* merupakan suatu model data yang dikembangkan berdasarkan objek. ER_M digunakan untuk menjelaskan hubungan antara data dalam basis data kepada

pengguna secara logik. ER_M didasarkan pada suatu persepsi bahwa *real world* terdi atas objek-objek dasar yang mempunyai hubungan/kerelasian antar objek-objek dasar tersebut. ER_M digambarkan dalam bentuk diagram yang disebut dengan ER *(ER_Diagram/ER_D)*. Untuk menggambarkan ER_D diguna-kan simbol-simbol grafis tertentu.

II.10. Normalisasi

Menurut Edhy Sutanta (2011 : 174) Normalisasi diartikan sebagai suatu teknik yang menstrukturkan atau mendekomposisikan data dalam cara-cara tertentu untuk mencegah timbulnya pemasalahan dalam pengolahan data dalam basis data. Permsalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan *(anomalies)* yang terjadi akibat adanya kerangkapan data dalam relasi dan in-efisiensi pengolahan. Proses normalisasi menghasilkan relasi yang optimal yaitu :

1. Memiliki struktur record yang konsisten secara logik
2. Memiliki struktur record yang mudah untuk dimengerti
3. Memiliki struktur record yang sederhana dalam pemeliharaan
4. Memiliki struktur record yang mudah ditampilkan kembali untuk memenuhi kebutuhan pengguna
5. Minimalisasi kerangkapan data guna meningkatkan kinerja sistem.

II.11 PHP & My SQL

PHP : Hypertext Processor adalah bahasa skrip yang dapat ditanamkan atau disisipkan ke dalam HTML. PHP banyak dipakai untuk mengiperasikan sutus *web* dinamis. PHP dapat digunakan untuk mebangun sebuah CMS. Pada awalnya PHP merupakan kependekatan dari personal home page (situs personal). PHP pertama kali dibuat oleh rumus lerdorf pada tahun 1995. Pada waktu itu PHP masih bernama form interorented (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data formulir web (Alan Nur Aditya : 2011 : 1).

Setelah belajar teknik dasar html kini kita belajar dasar My SQL, My SQL adalah subuah perangkat lunak sistem berbasis data SQL (bahasa inggirs : *database management system*) atau DBMS yang *multithread, multiuser*, dengan sekitar 6 juta instalasi di seluruh dunia. My SQL AB membuat My SQL tersedia sebagai perangkat lunak gratis dibawa lisensi GNU general *public license* (GPL), tetapi mereka juga menjuala dibawah *licensi* komersial untuk kasis-kasus dimana penggunanya tidak cocok dengan penggunaan GPL (Alan Nur Aditya : 2011 : 61).