BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem adalah sesuatu yang memiliki bagian – bagian yang saling berinteraksi untuk mencapai tujuan tertentu melalui tiga tahapan, yaitu input, proses, dan output. Sistem selalu terdiri dari beberapa subsistem kecil, yang masing – masing melaksanakan fungsi – fungsi khusus yang mendukung sistem. Sebagai sistem, setiap perusahaan menerima masukan – masukan dan mengubah menjadi keluaran – keluaran dalam bentuk produk atau jasa. (*Agustina Mela Sari ; 2013 : 1*).

II.2. Informasi

Informasi adalah salah satu jenis sumberdaya yang tersedia bagi manajer, yang dapat dikelola seperti halnya sumberdaya yang lain. Informasi dari komputer dapat digunakan oleh para manajer,non manajer, serta orang-orang dan organisasi-organisasi dalam lingkungan perusahaan. (*Agustina Mela Sari ; 2013 : 1*).

II.3. Sistem Informasi

Sistem informasi adalah kombinasi antara prosedur kerja, informasi, orang dan teknologi informasi yang diorganisasikan untuk mencapai tujuan dalam sebuah organisasi. Sistem informasi dapat didefinisikan sebagai sekumpulan komponen yang saling berhubungan, mengumpulkan, memproses, menyimpan dan mendistribusikan informasi untuk menunjang pengambilan keputusan dan

pengawasan dalam suatu organisasi. (*Dessi Tri Santi ; 2014 : 7-8*).

II.4. Sistem Informasi Geografis

Sistem Informasi Geografis (SIG) atau *Geographic Information System* (GIS) adalah sebuah sistem yang didesain utnuk menangkap, menyimpan, memanipulasi, menganalisa, mengatur dan menampilkan seluruh jenis data geografis.

Akronim GIS terkadang dipakai sebagai istilah untuk *geographical information science* atau *geospatial information studies* yang merupakan ilmu studi atau pekerjaan yang berhubungan dengan *Geographic Information System*. Dalam artian sederhana sistem informasi geografis dapat kita simpulkan sebagai gabungan kartografi, analisis statistik dan teknologi sistem basis data (*database*). *(Edy Irwansyah ; 2013 : 1).*

II.4.1. Bentuk Dasar Model GIS

Bentuk dari model SIG memiliki empat komponen utama, yaitu:

1. *Collection, input and correction* adalah operasi yang menekankan pada penerimaan / pengumpulan data dalam sistem, termasuk digitasi manual, *scanning, keyboard entry,* dan penarikan *online* dari sistem *database* lain. Pada tahap ini peta digital pertama kali dibangun .
2. *Mekanisme Storage and retrieval* termasuk kontrol fasilitas penyimpanan data dalam memori, disket, dan mekanisme penarikan untuk melayani kebutuhan ketiga komponen berikutnya.
3. *Manipulation and analysis* menampilkan keseluruhan teknik yang tersedia

dalam transformasi model digital menggunakan *mathematical medan.* Ini merupakan inti dari GIS, dan yang membedakannya dengan *Computer Assisted Cartography.* Sekumpulan algoritma data processing tersedia untuk transformasi data spasial, dan hasil dari manipulasi data dapat ditambahkan pada *database* digital dan dihubungkan dengan visualisasi baru dari sebuah peta

4. *Output and reporting* meliputi proses mengeluarkan data dari sistem dalam komputer atau bentuk lain yang dapat dibaca. Ini merupakan tahap di mana pengguna *database* digital dapat selektif membuat peta analog baru. *(Edy Irwansyah ; 2013 : 65-66)*

II.4.2. Komponen Sistem Informasi Geografis

Komponen-komponen yang membangun sebuah sistem informasi geografis adalah :

1. *Computer System and Software*

   Merupakan sistem komputer dan kumpulan pirakti lunak yang digunakan untuk mengolah data.

2. *Spatial Data*

   Merupakan data spasial (bereferensi keruangan dan kebumian) yang akan di olah. *(Edy Irwansyah ; 2013 : 11-12).*

II.4.3. Model Data Spasial

Pada pemanfaatannya data spasial yang diolah dengan menggunakan komputer (data spasial digital) menggunakan model sebagai pendekatannya. *Economic and Social Commission for Asia and the Pasific* (1996), mendefinisikan

model data sebagai suatu set logika atau aturan dan karateristik dari suatu data spasial. Model data merupakan representasi hubungan atara duania nyata dengan data spasial.

Terdapat dua model dalam data spasial, yaitu *model data raster* dan *model data vektor*. Keduanya memiliki karakteristik yang berbeda, selain itu dalam pemanfaatannya tergantung dari masukkan data dan hasil akhir yang akan dihasilkan. Model data tersebut merupakan representasi dari obyek-obyek geografi yang terekan sehingga dapat dikenali dan diproses oleh komputer. *(Edy Irwansyah ; 2013 : 22).*

II.4.4. Data *Raster*

Data raster adalah data yang disimpan dalam bentuk kotan segi empat (*grid)* / sel sehinggal terbentuk suatu ruang yang teratur. Foto digital seperti areal forografi atau foto satelit merupakan bagian dari data *raster* pada peta. *Raster* mewakili data grid *continue*. Nilainya menggunakan gambar berwarna seperti fotografi, yang ditampilkan dengan *level* merah, hijau, dan biru pada sel. Pada data raster, obyek geografis dipresentasikan sebagai struktur sel *grid* yang disebut sebagai *pixel* (*picture element*). Resolusi (definisi visual) tergantung pada ukuran fixel-nya, semakin kecil ukuran permukaan bumi yang direpresentasikan oleh sel, semakin tinggi resolusinya. *(Edy Irwansyah ; 2013 : 40).*

II.4.5. Data *Vektor*

Data vektor adalah data yang direkan dalam bentuk koordinat titik yang menampilkan, menempatkan, dan menyimpan data spasial dengan menggunakan

titik, garis, atau area (poligon). Ada tiga tipe data vektor (titik, garis, dan poligon) yang bisa digunakan untuk menampilkan informasi pada peta. Titik bisa digunakan sebagai lokasi sebuah kota atau posisi tower radio. Garis bisa digunakan untuk menunjukkan *route* suatu perjalanan atau menggambarkan *boundary*. Poligon bisa digunakan untuk menggambarkan sebuah danau atau sebuag Negara pada peta dunia. *(Edy Irwansyah ; 2013 : 40).*

II.5. UML *(Unified Modelling Language)*

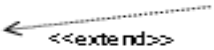
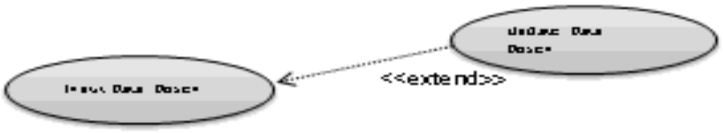
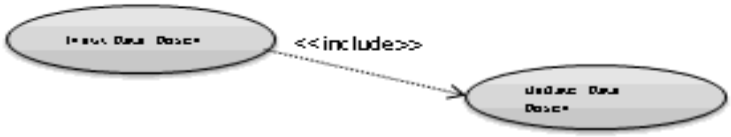
Unified Modelling Language (UML) adalah sebuah "bahasa" yg telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. Dengan menggunakan UML kita dapat membuat model untuk semua jenis aplikasi piranti lunak, dimana aplikasi tersebut dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Tetapi karena UML juga menggunakan *class* dan *operation* dalam konsep 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 mendefinisikan 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*

Design), Jim Rumbaugh OMT (*Object Modeling Technique*), dan Ivar Jacobson OOSE (*Object-Oriented Software Engineering*). Sejarah UML sendiri cukup panjang. Sampai era tahun 1990 seperti kita ketahui puluhan metodologi pemodelan berorientasi objek telah bermunculan di dunia. Diantaranya adalah: *metodologi booch, metodologi coad, metodologi OOSE, metodologi OMT, metodologi shlaer-mellor, metodologi wirfs-brock,* dsb. Masa itu terkenal dengan masa perang metodologi (*method war*) dalam pendesainan berorientasi objek. Masing-masing metodologi membawa notasi sendiri-sendiri, yang mengakibatkan timbul masalah baru apabila kita bekerjasama dengan group/perusahaan lain yang menggunakan metodologi yang berlainan. Dimulai pada bulan Oktober 1994 *Booch, Rumbaugh dan Jacobson,* yang merupakan tiga tokoh yang boleh dikata metodologinya banyak digunakan mempelopori usaha untuk penyatuan metodologi pendesainan berorientasi objek. Pada tahun 1995 direlease draft pertama dari UML (versi 0.8). Sejak tahun 1996 pengembangan tersebut dikoordinasikan oleh Object Management Group (OMG – http://www.omg.org). Tahun 1997 UML versi 1.1 muncul, dan saat ini versi terbaru adalah versi 1.5 yang dirilis bulan Maret 2003. Booch, Rumbaugh dan Jacobson menyusun tiga buku serial tentang UML pada tahun 1999. Sejak saat itulah UML telah menjelma menjadi standar bahasa pemodelan untuk aplikasi berorientasi objek. *(Yuni Sugiarti ; 2013 : 33)*

Dalam pembuatan skripsi ini penulis menggunakan diagram Use Case yang terdapat di dalam UML. Adapun maksud dari Use Case Diagram diterangkan dibawah ini.

1. *Use Case Diagram*

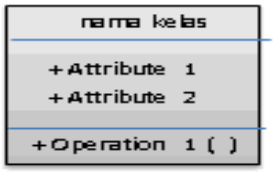
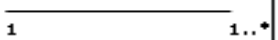
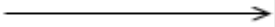
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain. *(Yuni Sugiarti ; 2013 : 41)*

Simbol	Deskripsi
Use Case 	fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit dan aktor; biasanya dinyatakan dengan menggunakan kata kerja di awal frase nama use case
Aktor  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 aktor adalah gambar orang, tapi aktor belum tentu merupakan orang; biasanya dinyatakan menggunakan kata benda di awal frase nama aktor
Asosiasi / association	komunikasi antara aktor dan use case yang berpartisipasi pada use case atau use case memiliki interaksi dengan aktor
Extend  <<extend>>	relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan dapat berdiri sendiri walaupun tanpa use case tambahan itu; mirip dengan prinsip inheritance pada pemrograman berorientasi objek; biasanya use case tambahan memiliki nama depan yang sama dengan use case yang ditambahkan, arah panah menunjukan pada use case yang dituju contoh : 
Include <<include>> 	relasi use case tambahan ke sebuah use case dimana use case yang ditambahkan memerlukan use case ini untuk menjalankan fungsinya atau sebagai syarat dijalankan 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, contoh : 

Gambar 2.1. *Use Case* Diagram
Sumber : (Yuni Sugiarti ; 2013 ; 42)

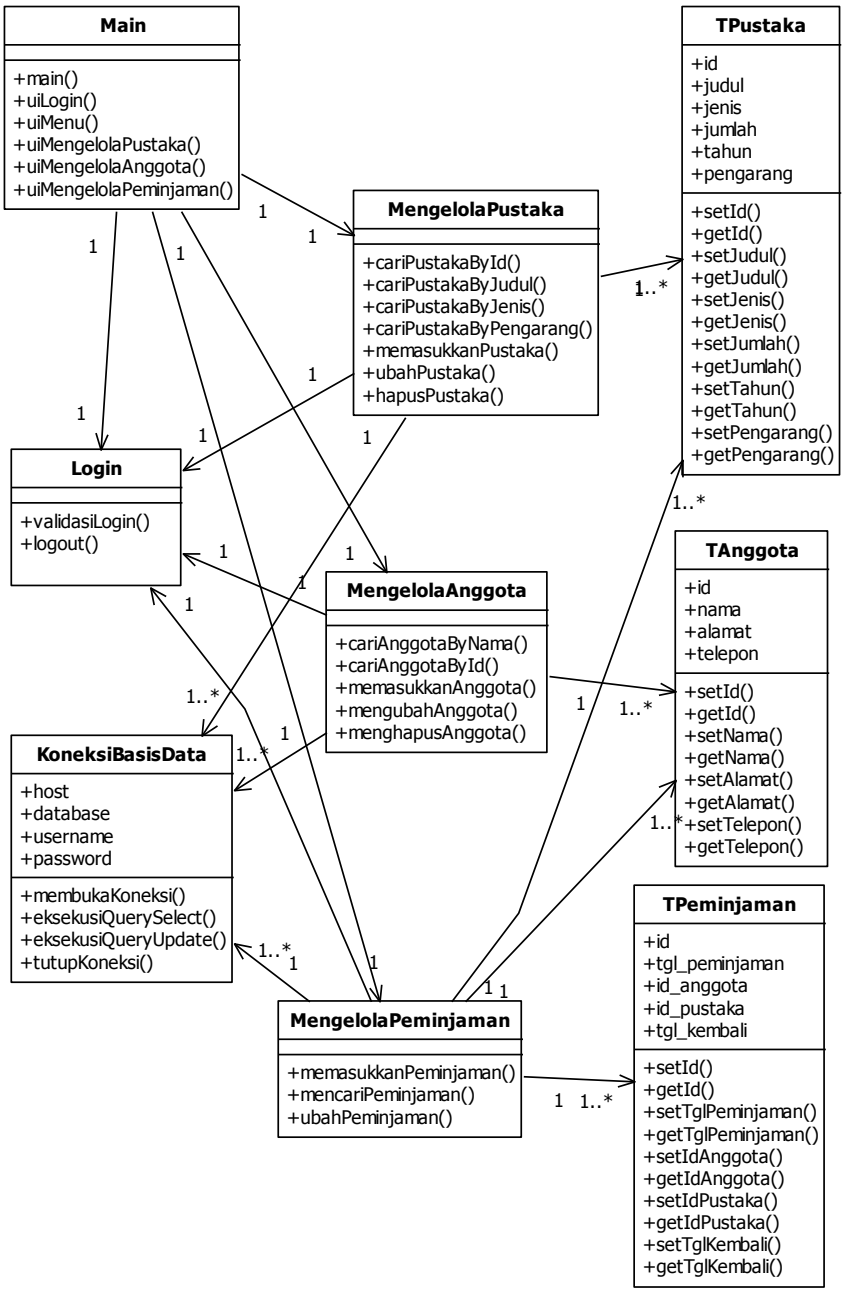
2. *Class Diagram*

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. Berikut adalah simbol-simbol pada diagram kelas :

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antarmuka / interface  interface	sama dengan konsep interface dalam pemrograman berorientasi objek
Asosiasi 	relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah/directed asosiasi 	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)
Kebergantungan / defedency	relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi 	relasi antar kelas dengan makna semua-bagian (whole-part)

Gambar 2.2. *Class* Diagram

Sumber : (Yuni Sugiarti ; 2013 : 59)



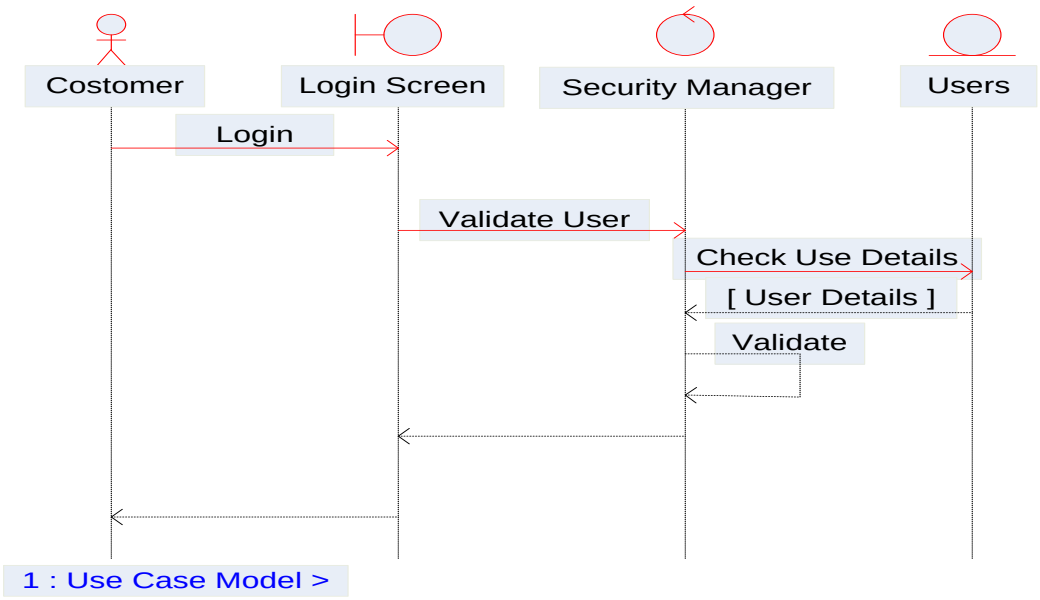
Gambar 2.3. Contoh Class Diagram
Sumber : (Yuni Sugiarti ; 2013 : 63)

3. *Sequence Diagram*

Diagram *Sequence* menggambarkan kelakuan/prilaku objek pada *use case* dengan mendeskripsikan waktu hidup objek dan *message* yang dikirimkan dan

diterima antar objek. Oleh karena itu untuk menggambarkan diagram *sequence* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* beserta metode-metode yang dimiliki kelas yang diinstansiasi menjadi objek itu.

Banyaknya diagram *sequence* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri atau yang penting semua *use case* yang telah didefinisikan interaksi jalannya pesan sudah dicakup pada diagram *sequence* sehingga semakin banyak *use case* yang didefinisikan maka diagram *sequence* yang harus dibuat juga semakin banyak.



Gambar 2.4. Contoh *Sequence* Diagram
Sumber : (Yuni Sugiarti ; 2013 : 63)

4. *Activity Diagram*

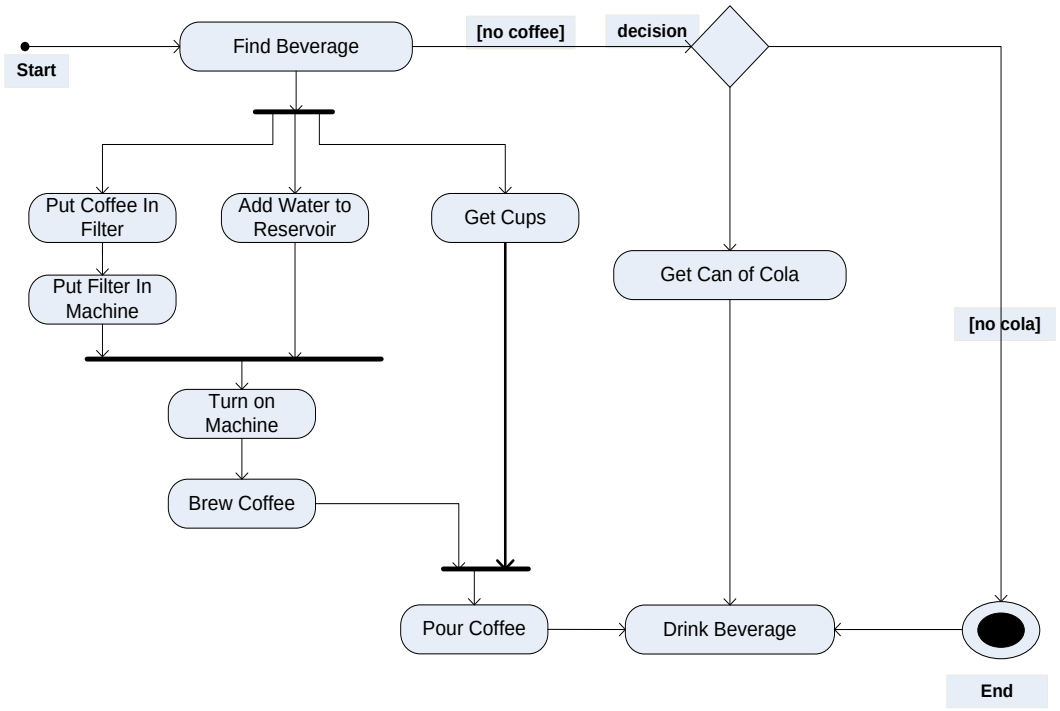
Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

Activity diagram merupakan *state diagram* khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu *activity diagram* tidak menggambarkan behaviour internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas.

Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan behaviour pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal.

Activity diagram dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu.



Gambar 2.5. Activity Diagram
Sumber : (Yuni Sugiarti ; 2013 : 76)

II.6. PHP

PHP singkatan dari PHP : *Hypertext Preprocessor* yaitu bahasa pemrograman web server-side yang bersifat open source. PHP merupakan script yang terintegrasi dengan HTML dan berada pada server (Server side HTML embedded scripting). PHP adalah script yang digunakan untuk membuat halaman website yang dinamis. Dinamis berarti halaman yang akan ditampilkan dibuat saat halaman itu diminta oleh client. Mekanisme ini menyebabkan informasi yang diterima client selalu yang terbaru/ up to date. Semua script PHP dieksekusi pada server dimana script tersebut dijalankan. *(Anhar,ST ; 2013 : 3).*

II.6.1. *Sintaks* dasar PHP

PHP merupakan sebuah bahasa pemrograman web yang memiliki sintak atau aturan dalam menuliskan script atau kode-kodenya. Untuk menjelaskan cara penulisan kode PHP, bia kita lihat pada empat macam cara penulisan kode PHP, yaitu :

1. <? echo (“ini adalah script PHP\n”); ?>
2. <? php echo (“ini juga script PHP\n”); ?>
3. <script language=”php>

   echo (“LAtihan menulis script PHP”);

   </script>
4. <% echo (”kalau yang ini mirip ASP”); %>. *(Anhar,ST ; 2013 : 23-24).*

II.7. Basis Data

Basis Data adalah sekumpulan tabel-tabel yang saling berelasi, relasi tersebut bisa ditunjukkan dengan kunci dari tabel yang ada. Basis data juga bisa didefinisikan sebagai suatu kumpulan dari data yang tersimpan dan diatur atau diorganisasikan sehingga data tersebut bisa diambil atau dicari dengan mudah dan efisien.

Menurut Bambang Hariyanto,2004 Basis data adalah kumpulan data (elementer) yang secara logik berkaitan dalam merepresentasikan fenomena atau fakta secara terstruktur dalam domain tertentu untuk mendukung aplikasi pada sistem tertentu. Basis data adalah kumpulan data yang saling terkait digunakan untuk memenuhi kebutuhan tertentu (prahasta, E. 2012).

Basis data adalah suatu kumpulan data yang disusun dalam bentuk tabel-

tabel yang saling berkaitan maupun berdiri sendiri dan disimpan secara bersama-sama pada suatu media. Basis data dapat digunakan oleh satu atau lebih program aplikasi secara optimal, data disimpan tanpa mengalami ketergantungan pada program yang akan menggunakannya. (*Annisaa Cahyaningsih Rangkuti ; 2014 : 94*).