BAB II

TINJAUAN PUSTAKA

II.1. Travel Agent

Travel Agent merupakan usaha yang bergerak dibidang jasa yang memiliki tujuan untuk menyiapkan suatu perjalanan bagi seseorang yang merencanakan untuk mengadakannya. *(Anisya ; 2013 : 49)*

II.2. Defenisi Sistem Informasi

Sistem Informasi (SI) adalah kombinasi dari teknologi informasi dan aktivitas orang yang menggunakan teknologi itu untuk mendukung operasi dan manajemen. Dalam arti yang sangat luas, istilah sistem informasi yang sering digunakan merujuk kepada interaksi antara orang, proses algoritmik, data, dan teknologi. Dalam pengertian ini, istilah ini digunakan untuk merujuk tidak hanya pada penggunaan organisasi Teknologi Informasi dan Komunikasi (TIK), tetapi juga untuk cara di mana orang berinteraksi dengan teknologi ini dalam mendukung proses bisnis.

Komponen Sistem Informasi

Sistem informasi terdiri dari komponen-kompponen yang disebut dengan istilah blok bangunan (Building Block), dimana masing-masing blok ini saling berintegrasi satu sama lainya membentuk satu kesatuan untuk mencapai tujuannya. (*Anisya ; 2013 : 50*)

Adapun blok-blok tersebut adalah sebagai berikut :

1. **Blok masukan (*Input Blok*)**

Meliputi metode-metode dan media untuk menangkap data yang akan dimasukan, dapat berupa dokumen-dokumen dasar.

2. **Blok Model (*Model Block*)**

Terdiri dari kombinasi prosedur, logika dan model matematik yang berfungsi memanipulasi data untuk keluaran tertentu.

3. **Blok Keluaran (*Output Block*)**

Berupa keluaran dokumen dan informasi yang berkualitas

4. **Blok Teknologi**

Untuk menerima input, menjalankan model, menyimpan dan mengakses data, menghasilkan dan mengirimkan keluaran serta membantu pengendalian dari sistem secara keseluruhan.

5. **Blok Basis Data (*Database Block*)**

Merupakan kumpulan data yang saling berhubungan satu dengan yang lainnya, tersimpan didalam perangkat keras komputer dan perangkat lunak untuk memanipulasi.

6. **Blok Kendali (*Controls Block*)**

Meliputi masalah pengendalian yang berfungsi mencegah dan menangani kesalahan atau kegagalan sistem.

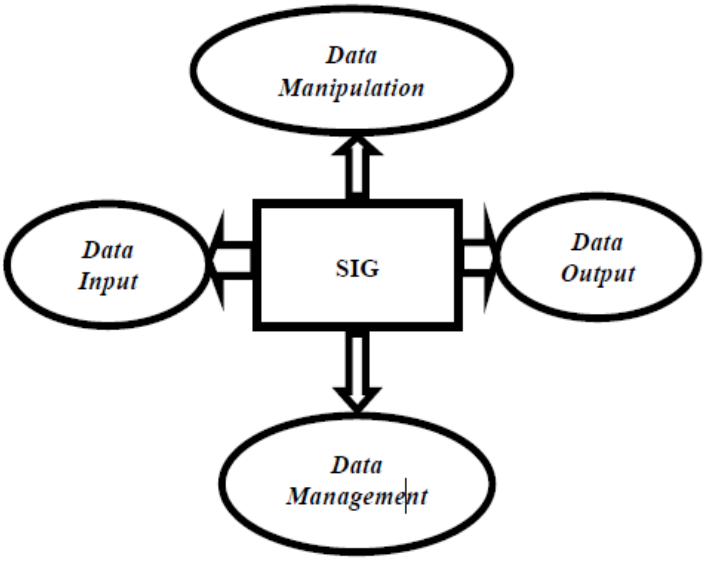
II.3. Sistem Informasi Geografis

Geografi adalah ilmu yang mempelajari bumi dengan menggunakan pendekatan keruangan, ekologi, dan kompleks wilayah. Fenomena yang diamati merupakan dinamika perkembangan dan pengembangan wilayah yang ada dalam kehidupan sehari-hari, misalnya informasi mengenai letak dan persebaran dari kejadian-kejadian alamiah maupun fenomena terdapatnya sumber daya ketersediaan data yang bersifat geografi. Sumber daya yang memiliki atribut utama keruangan akan mempermudah penyelesaian banyak kepentingan Secara umum, sistem informasi geografis atau geographic information system (GIS), merupakan suatu sistem (berbasiskan komputer) yang digunakan untuk menyimpan, dan menganalisis objek-objek dan fenomena-fenomena lokasi geografis merupakan karakteristik yang penting atau kritis untuk dianalisis. (*Muhammad Sholeh ; 2013 : 169*)

Meskipun SIG mampu membuat dan menampilkan peta, tetapi masih banyak hal lain yang biasa dikerjakannya. SIG sebagai himpunan terpadu dari hardware, software, data, livewire. Aplikasi SIG yang baik adalah apabila aplikasi tersebut dapat menjawab salah satu atau lebih dari 3 (tiga) pertanyaan dasar berikut:

1. Lokasi, dapat digunakan untuk menjawab pertanyaan mengenai lokasi tertentu.
2. Kondisi, dapat digunakan untuk menjawab pertanyaan mengenai kondisi dari suatu lokasi.
3. Pola, dapat digunakan untuk membaca gejala gejala alam dan mempelajarinya.

Sistem Informasi Geografis pada dasarnya dapat dirinci menjadi empat subsistem seperti pada Gambar II.1, yaitu: subsistem pemasukan dan pengkodean data (data *input*), subsistem penyimpanan, pengambilan dan pengolahan data (data *management*), subsistem manipulasi dan analisa data (data *manipulation and analysis*), serta subsistem penyajian data (data *ouput*).



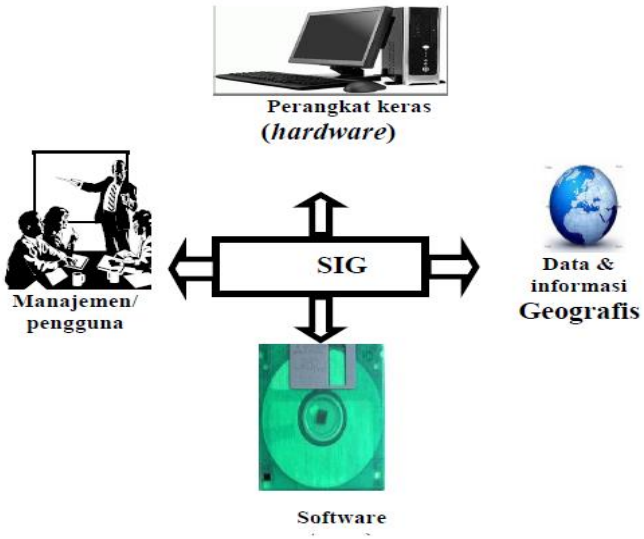
Gambar II.1. Subsistem Dalam Perangkat Lunak SIG
Sumber : Muhammad Sholeh (2013 : 169)

Secara garis besar komponen subsistem tersebut dapat diuraikan sebagai berikut:

1. Subsistem pemasukan dan pengkodean data (data *input*) Subsistem ini bertugas untuk mengumpulkan dan mempersiapkan data spasial dan data atribut dari berbagai sumber, subsistem ini pula yang bertanggung jawab dalam mengonversi atau mentransformasikan format-format data aslinya ke dalam format yang dapat digunakan oleh Sistem Informasi Geografis.

2. Subsistem penyimpanan, pengambilan dan pengolahan data (data *management*) Subsistem ini mengorganisasikan data spasial maupun atribut ke dalam sebuah *database* sehingga mudah dipanggil, di *update* dan di *edit*.
3. Subsistem manipulasi dan analisis data (data *manipulation and analyst*) Subsistem ini menentukan informasi yang dapat dihasilkan oleh sistem informasi geografis. Selain itu, subsistem ini juga melakukan manipulasi dan pemodelan data untuk menghasilkan informasi yang diharapkan
4. Subsistem penyajian data (data *output*) Subsistem ini menampilkan atau menghasilkan keluaran seluruh atau sebagian database baik dalam bentuk *softcopy* maupun bentuk *hardcopy* seperti tabel, grafik, peta dan lainnya.

Selain keempat subsistem di atas, SIG sebagai sistem yang kompleks dan terintegrasi dengan lingkungan sistem-sistem yang lain di tingkat fungsional dan jaringan terdiri dari beberapa komponen. Komponen-komponen Sistem Informasi Geografis tersebut dapat dilihat pada gambar berikut ini:



Gambar II.2. Komponen Utama SIG
Sumber : (Muhammad Sholeh ; 2013 : 169)

Adapun penjelasan dari masing-masing komponen SIG tersebut adalah:

1. Perangkat keras (*Hardware*)

Pada saat ini SIG tersedia untuk berbagai platform perangkat keras mulai dari PC d*e*sktop, *workstation*, hingga *multi-user host* yang dapat digunakan oleh banyak orang secara bersamaan dalam jaringan computer yang luas, berkemampuan tinggi, memiliki ruang penyimpanan (*Hard disk*) yang besar dan mempunyai kapasitas *memory* (RAM) yang besar.

2. Perangkat lunak (*Software*)

Beberapa perangkat lunak yang diperlukan untuk mendukung penyajian SIG yang lebih baik, misalnya *Xing Mpeg Player*, Media *Player*, *Adobe Photoshop*, *Macromedia Flash*. Pemilihan perangkat lunak SIG sangat bergantung pada sejumlah faktor, yaitu tujuan aplikasi, biaya, serta kemampuan user dalam menggunakan perangkat lunak SIG tersebut.

3. Data dan Informasi Geografis

Sistem informasi geografis dapat mengumpulkan, mengolah dan menyimpan data dan informasi yang diperlukan baik secara tidak langsung dengan cara mengimpornya dari perangkat-perangkat lunak SIG yang lain atau perangkat lunak yang mendukungnya maupun secara langsung dengan memasukkan data atributnya atau dengan cara mendigitasi data spasialnya dari peta yang telah ada sebelumnya.

4. Manajemen / Pengguna

Fungsi pengguna atau manajemen adalah untuk memilih informasi yang diperlukan, membuat jadwal *updating* yang efisien, merencanakan aplikasi dan

menganalisis hasil yang dikeluarkan untuk kegunaan yang diinginkan, sehingga hasil akhir yang diperoleh dapat mencapai tujuan pembuatan aplikasi serta sesuai dengan kebutuhan pemakai khususnya pada tingkat *end user*.

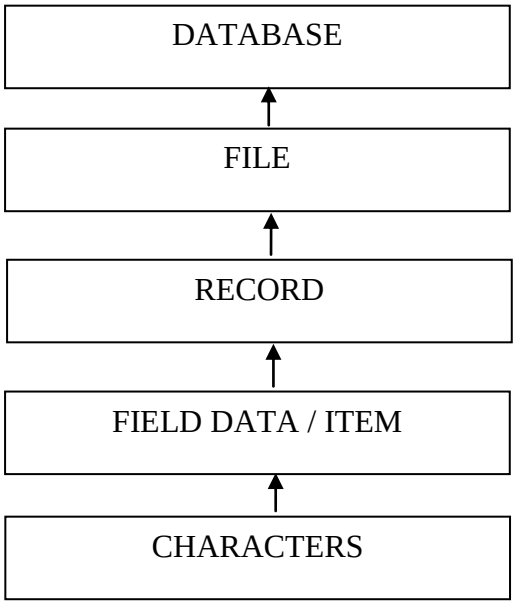
II.4. *Database*

Pengertian *database* adalah :*"kumpulan data (elementer) yang secara logic berkaitan dalam mempresentasikan fenomena/fakta secara terstruktur dalam domain tertentu untuk mendukung aplikasi dalam system tertentu".* Dari definisi diatas maka dapat disimpulkan bahwa database adalah kumpulan dari item data yang saling berhubungan satu dengan yang lainnya yang diorganisasikan berdasarkan sebuah skema atau struktur tertentu, yang kelak dapat dimanfaatkan kembali dengan cepat dan mudah. (*Minarni ; 2014 : 105*)

Alasan diperlukan *Database*

1. Salah satu komponen penting dalam sistem informasi, karena merupakan dasar dalam menyediakan informasi
2. Menentukan kualitas informasi : akurat, tepat pada waktunya dan relevan. Informasi dapat dikatakan bernilai bila manfaatnya lebih efektif dibandingkan dengan biaya mendapatkannya.
3. Mengurangi duplikasi data (*data redudancy*)
4. Hubungan data dapat ditingkatkan (*data relatability*)
5. Mengurangi pemborosan tempat simpanan luar

Untuk lebih jelasnya dapat dilihat pada gambar II.3 jenjang data :



Gambar II.3. Jenjang Data
(Sumber : Minarni 2014 : 105)

Dimana:

1. *Characters* : merupakan bagian data yang terkecil, dapat berupa karakter numerik, huruf ataupun karakterkarakter khusus (*special characters*) yang membentuk suatu item data/*field*.
2. *Field* : merepresentasikan suatu atribut dari *record* yang menunjukkan suatu item dari data, seperti misalnya nama, alamat dan lain sebagainya. Kumpulan dari *field* membentuk suatu *record*.
   a. *field* name: harus diberi nama untuk membedakan field yang satu dengan lainnya
   b. *field* representation: tipe *field* (karakter, teks, tanggal, angka, dsb), lebar field (ruang maksimum yang dapat diisi dengan karakterkarakter data).
   c. *field value*: isi dari *field* untuk masing-masing *record*.

3. *Record* : Kumpulan dari *field* membentuk suatu *record*. *Record* menggambarkan suatu unit data individu yang tertentu. Kumpulan dari record membentuk suatu *file*. Misalnya *file* personalia, tiap-tiap *record* dapat mewakili data tiap-tiap karyawan.
4. *File*: *File* terdiri dari *record-record* yang menggambarkan satu kesatuan data yang sejenis. Misalnya *file* mata pelajaran berisi data tentang semua mata pelajaran yang ada.
5. Database : Kumpulan dari *file* / *table* membentuk suatu database.

II.5. *Internet*

Internet (*Interconnected networks*) adalah kumpulan jaringan-jaringan komputer (*networks*) sedunia yang saling berhubungan satu sama lain. Agar bisa berhubungan, *Internet* menggunakan bahasa yang sama yang disebut TCP/IP (*Transmission Control Protocol* / *Internet Protocol*). TCP/IP memberikan sebuah alamat (*address*) dan identitas (identity : disingkat ID) yang unik (tidak boleh sama) pada setiap komputer diseluruh dunia untuk menghindari adanya kesalahan pengiriman data. Sebagai sebuah jaringan komputer dunia, *Internet* dapat dikatakan sebagai jalur transportasi segala informasi yang berbentuk *file* atau data pada komputer lain. Dengan demikian, *Internet* sendiri tidak mengandung informasi. Lebih tepat dikatakan, bahwa informasi dapat ditemukan melalui atau menggunakan *internet*. (*Uswatun Hasanah ; 2013 : 41*)

II.6. MySQL

SQL penting sekali di pelajari, karena dalam setiap kasus pemrograman database, perintah SQL selalu digunakan untuk membuat *query* data didalam *database*. Jadi, *MySQL* itu adalah adalah software atu program *database server*, sedangkan SQL adalah bahasa pemrogramannya, dia itu bahasa permintaan (*query*) dalam database server ternasuk dalam *MySQL* itu sendiri.

Database MySql adalah program berbasis DOS, perintah dasarnya adalah SQL (*Structured Query Language),* anda dapat mengaksesnya dari jendela DOS Prompt atau Command prompt. (*Bunafit Nugroho;2013 : 26*)

II.7. *Website*

Website (Situs *Web*) merupakan kumpulan dari halaman-halaman *web* yang berhubungan dengan *file-file* lain yang terkait. Dalam sebuah *website* terdapat suatu halaman yang dikenal dengan sebutan *home page*. *Home page* adalah sebuah halaman yang pertama kali dilihat ketika seseorang mengunjungi website. Dari *home page*, pengujung dapat mengklik *hyperlink* untuk pindah kehalaman lain yang terdapat dalam website tersebut. (*Yoni Widhiarso ; 2013 : 3*)

Secara umum website mempunyai fungsi sebagai berikut :

1. Fungsi Komunikasi

Beberapa fasilitas yang memberikan fungsi komunikasi, seperti : *chatting*, *web base email* dan lain-lain.

2. Fungsi Informasi

Fungsi informasi *website* seperti : *News*, *Profile*, *Library*, referensi dan lain-lain.

3. Fungsi *Intertainment*

Website mempunyai fungsi hiburan. Misalnya *web-web* yang menyediakan *game on-line, music on-line* dan lain-lain.

II.8. Pemograman *Web*

II.8.1. *HTML*

HTML (*Hypertext Markup Language*) adalah sebuah bahasa pemrograman yang berbentuk skrip skrip yang berguna untuk membuat sebuah halaman web. *HTML* dapat dibaca oleh berbagai platform seperti : *Windows, Linux, Macintosh*. Kata ”*Markup Language*” pada *HTML* menunjukkan fasilitas yang berupa tanda tertentu dalam skrip *HTML* dimana kita bias mengatur judul, garis, tabel, gambar, dan lainlain dengan perintah yang telah ditentukan pada elemen *HTML*. *HTML* sendiri dikeluarkan oleh W3C (*Word Wide Web Consortin*), setiap terjadi perkembangan level *HTML* harus dievakuasi ketat dan disetujui oleh W3C. (*Uswatun Hasanah ; 2013 : 41*)

II.8.2. 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).*

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 :

<? echo (“ini adalah script PHP\n”); ?>

<? php echo (“ini juga script PHP\n”); ?>

<script language=”php>

echo (“LAtihan menulis script PHP”);

</script>

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

II.9. *ArcView*

ArcView merupakan salah satu perangkat lunak Sistem Informasi Geografis dan pemetaan yang dikembangkan oleh ESRI (*Envirnmental System Research Institute*, inc) yaitu salah satu perusahaan yang menghasilkan produk SIG yang handal dan juga merupakan provider yang terdepan dan terbesar perangkat lunak SIG sejak tahun 1992 yang saat ini telah menjadi *software* Sistem Informasi Geografis ternama di dunia. *ArcView* memiliki tampilan yang menarik, interaktif, memiliki tingkat kemudahan yang tinggi hingga terkenal dan sering digunakan dewasa ini. Hampir semua pengguna aplikasi SIG mengenal *ArcView*.

Dengan *ArcView*, pengguna dapat memiliki kemampuan-kemampuan untuk melakukan visualisasi, explore dan menjawab query (baik basis data spasial maupun non-spasial), menganalisa data secara geografis dan sebagainya.

ArcView adalah dalam pengolahan atau *editing*, menerima dan mengkonversi dari data digital lain seperti CAD, atau dihubungkan dengan data image seperti format .JPG, .TIFF, atau image bergerak (.GIF). Input data spasial sering disebut dengan digitasi. *ArcView* juga memiliki kemampuan untuk melakukan digitasi. Data hasil digitasi yang berasal dari proses input data disimpan dalam sebuah *Theme* yang selanjutnya dapat diolah atau ditransfer ke *software* lain untuk pengolahan selanjutnya.

Beberapa fungsi utama *ArcView* GIS adalah pertukaran data, membaca dan menuliskan data dari dan ke dalam format perangkat lunak SIG lainya, melakukan analisis statistik dan operasi-operasi matematis, menampilkan informasi spasial dengan atribut-atributnya yang terdapat dalam (disimpan) basis data atribut, melakukan fungsi-fungsi dasar SIG seperti analisis sederhana spasial dan membuat peta tematik serta Meng-*customize* aplikasi dengan menggunakan bahasa *script* atau bahasa pemrograman sederhana. Selain itu *ArcView* juga mempunyai kemampuan Tracking Analyst, yang diirancang untuk organisasi-organisasi yang memonitor obyek-obyek fenomena yang begerak atau berubah sesuai dengan perubahan waktu dan Internet mapserver yang digunakan untuk mempublikasikan peta-peta dinamis melalui internet dengan menggunakan *ArcView* standar. (*Uning Lestari ; 2012 : 118*)

II.10. 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*)*

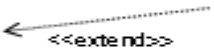
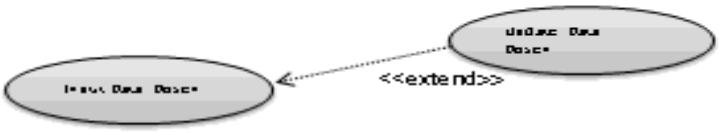
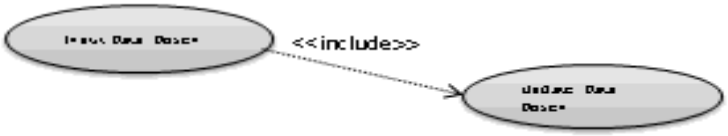
Dalam pembuatan skripsi ini penulis menggunakan diagram Use Case yang terdapat di dalam UML. Adapun maksud dari Use Case Diagram diterangkan dibawah ini.

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*)*

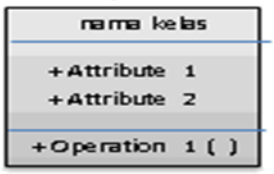
Use case dapat dilihat pada gambar II.4 :

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 : 

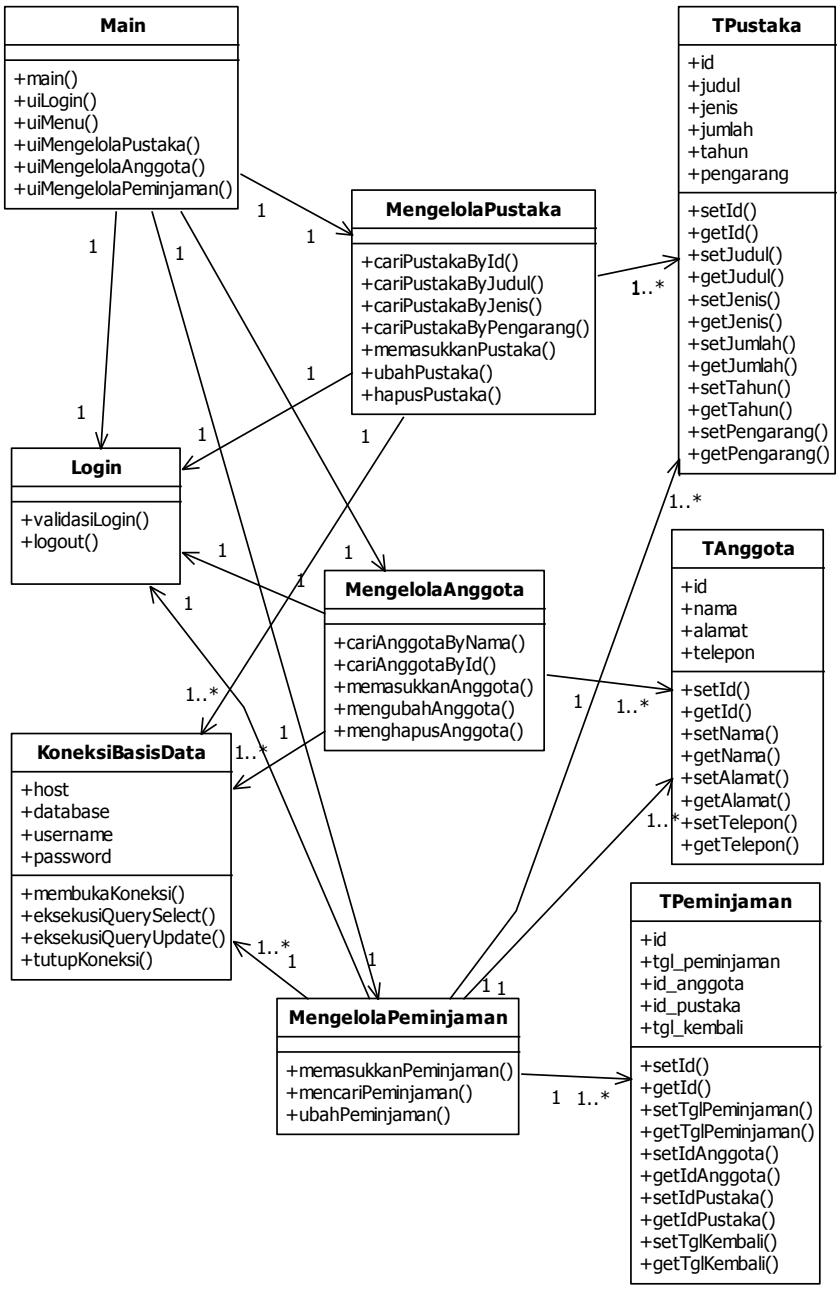
Gambar II.4. *Use Case* Diagram
*Sumber : (*Yuni Sugiarti ; 2013 : 42*)*

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 dapat dilihat gambar II.5 :

Simbol	Deskripsi
Package 	Package merupakan sebuah bungkusan dari satu atau lebih kelas
Operasi 	Kelas pada struktur sistem
Antarmuka / 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 II.5. *Class* Diagram
*Sumber : (*Yuni Sugiarti ; 2013 : 59*)*



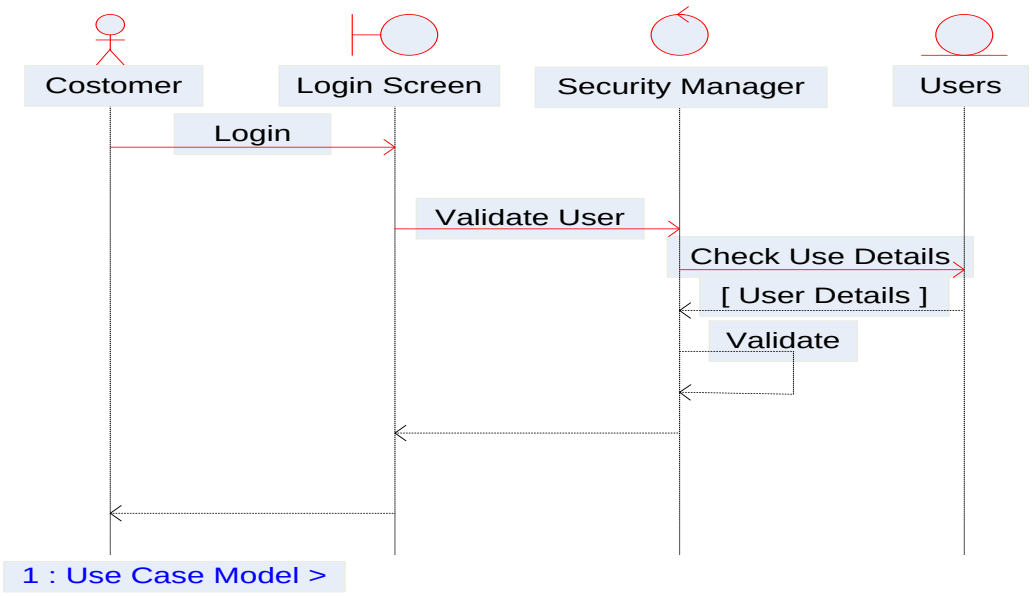
Gambar II.6. Contoh Class Diagram
*Sumber : (*Yuni Sugiarti ; 2013 : 63*)*

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 II.7. Contoh *Sequence* Diagram
*Sumber : (*Yuni Sugiarti ; 2013 : 63*)*

4. *Activity Diagram*

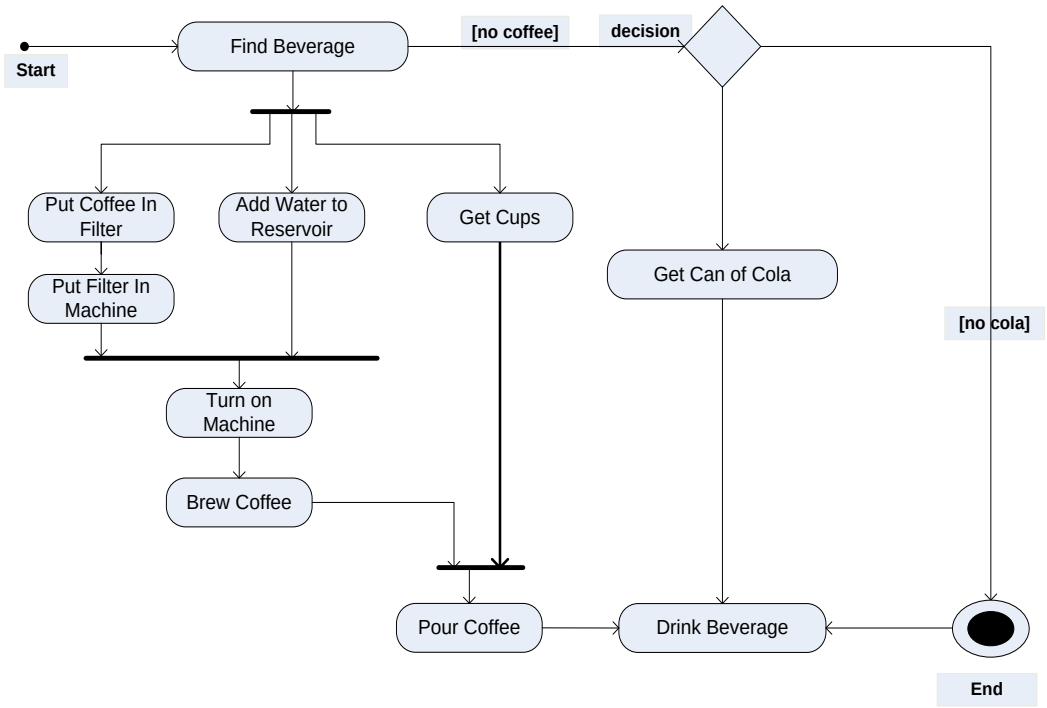
Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi.

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 II.8. Activity Diagram
*Sumber : (*Yuni Sugiarti ; 2013 : 76*)*