# BAB II

# LANDASAN TEORI

## II.1. Konsep Dasar

### II.1.1. Sistem

Gordon B. Davis dalam bukunya menyatakan bahwa sistem bisa berupa abstrak atau fisik. Sistem yang abstrak adalah susunan gagasan-gagasan atau konsepsi yang teratur yang saling bergantung. Sedangkan sistem yang bersifat fisik adalah serangkaian unsur yang bekerja sama untuk mencapai suatu tujuan.

Norman L. Enger menyatakan bahwa suatu sistem dapat terdiri atas kegiatan-kegiatan yang berhubungan guna mencapai tujuan-tujuan perusahaan seperti pengendalian inventaris atau penjadwalan produksi. Sedangkan Prof. Dr. Mr. S. Prajudi Atmosudirdjo menyatakan bahwa suatu sistem terdiri atas objek-objek atau unsur-unsur atau komponen-komponen yang berkaitan dan berhubungan satu sama lainnya sedemikian rupa sehingga unsur-unsur tersebut merupakan suatu kesatuan pemrosesan atau pengolahan yang tertentu (Tata Sutabri, 2012: 6-7).

Secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur, komponen, atau variabel yang terorganisasi, saling berinteraksi, saling tergantung satu sama lain dan terpadu (Tata Sutabri, 2012: 3).

### II.1.2. Informasi

Informasi merupakan proses lebih lanjut dari data yang sudah memiliki nilai tambah, informasi dapat dikelompokkan menjadi 3 bagian, yaitu:

1. Informasi Strategis. Informasi ini digunakan untuk mengambil keputusan jangka panjang, yang mencakup informasi eksternal, rencana perluasan perusahaan, dan sebagainya.
2. Informasi Taktis. Informasi ini dibutuhkan untuk mengambil keputusan jangka menengah, seperti informasi tren penjualan yang dapat dimanfaatkan untuk menyusun rencana penjualan.
3. Informasi Teknis. Informasi ini dibutuhkan untuk keperluan operasional sehari-hari, seperti informasi persediaan stock, retur penjualan, dan laporan kas harian.

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan. Sistem pengolahan informasi akan mengolah data menjadi informasi atau mengolah data dari bentuk tak berguna menjadi berguna bagi yang menerimanya. Nilai informasi berhubungan dengan keputusan. Bila tidak ada pilihan atau keputusan maka informasi tidak diperlukan. Keputusan dapat berkisar dari keputusan berulang sederhana sampai keputusan strategis jangka panjang. Nilai informasi dilukiskan paling berarti dalam konteks pengambilan keputusan (Tata Sutabri, 2012: 21-22).

### II.1.3. Sistem Informasi

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan laporan-laporan yang diperlukan oleh pihak luar tertentu.

Tipe sistem informasi adalah sebagai berikut:

1. Sistem informasi Akuntansi
2. Sistem informasi Pemasaran
3. Sistem informasi Manajemen Persediaan
4. Sistem informasi Personalia
5. Sistem informasi Distribusi
6. Sistem informasi Pembelian
7. Sistem informasi Kekayaan
8. Sistem informasi Analisis Kredit
9. Sistem informasi Penelitian dan Pengembangan
10. Sistem informasi Teknik

Semua sistem informasi tersebut dimaksudkan untuk memberikan informasi kepada semua tingkat manajemen, mulai manajemen tingkat bawah, manajemen tingkat menengah, hingga manajemen tingkat atas (Tata Sutabri, 2012: 38-41).

## II.2. Sistem Pendukung Keputusan

Pengambilan keputusan merupakan proses pemilihan alternative tindakan untuk mencapai tujuan atau sasaran tertentu. Pengambilan keputusan dilakukan dengan pendekatan sistematis terhadap permasalahan melalui proses pengumpulan data menjadi informasi serta ditambah dengan faktor - faktor yang perlu dipertimbangkan dalam pengambilan keputusan. Menurut Keen dan Scoot Morton: "Sistem Pendukung Keputusan merupakan penggabungan sumber - sumber kecerdasan individu dengan kemampuan komponen untuk memperbaiki kualitas keputusan. Sistem Pendukung Keputusan juga merupakan sistem informasi berbasis komputer untuk manajemen pengambilan keputusan yang menangani masalah - masalah semi struktur ".

Dengan pengertian diatas dapat dijelaskan bahwa sistem pendukung keputusan bukan merupakan alat pengambilan keputusan, melainkan merupakan sistem yang membantu pengambil keputusan dengan melengkapi mereka dengan informasi dari data yang telah diolah dengan relevan dan diperlukan untuk membuat keputusan tentang suatu masalah dengan lebih cepat dan akurat. Sehingga sistem ini tidak dimaksudkan untuk menggantikan pengambilan keputusan dalam proses pembuatan keputusan.

Alter (2002:38) mendefinisikan Sistem pendukung keputusan atau *Decision Support Systems* (DSS) adalah sistem informasi interkatif yang menyediakan informasi, pemodelan, dan pemanipulasian data yang digunakan untuk membantu pengambilan keputusan pada siuasi yang semiterstruktur dan

situasi yang tidak terstruktur di mana tak seorang pun tahu secara pasti bagaimana keputusan seharusnya dibuat. Konsep DSS dikemukaakan pertama kali oleh Scoot Morton pada tahun 1971 (Turban, McLean, dan Wetherbe, 1999). Beliau mendefinisikan cikal bakal DSS tersebut sebagai : "sistem berbasis komputer yang interaktif, yang membantu pengambil keputusan dengan menggunakan data dan model untuk memecahkan persoalan-persoalan tak terstruktur (Ahmad Khaidir, 2014).

### II.2.1. Karakteristik Sistem Pendukung Keputusan

Karakteristik dalam sistem pendukung keputusan, antara lain (Erliza Septia Nagara dan Rini Nurhayati, 2015):

1. Sistem Pendukung Keputusan dirancang untuk membantu pengambil keputusan dalam memecahkan masalah yang sifatnya semi terstruktur ataupun tidak terstruktur dengan menambahkan kebijaksanaan manusia dan informasi komputerisasi.
2. Dalam proses pengolahannya, sistem pendukung keputusan mengkombinasikan penggunaan model-model analisis dengan teknik pemasukan data konvensional serta fungsi-fungsi pencari/interogasi informasi.
3. Sistem Pendukung Keputusan, dirancang sedemikian rupa sehingga dapat digunakan/dioperasikan dengan mudah.
4. Sistem Pendukung Keputusan dirancang dengan menekankan pada aspek fleksibilitas serta kemampuan adaptasi yang tinggi.

### II.2.2. Keuntungan dan Keterbatasan Sistem Pendukung Keputusan

Sistem pendukung keputusan dapat memberikan berbagai manfaat atau keuntungan bagi pemakainya, antara lain (Nungsiati, 2013) :

1. Memperluas kemampuan pengambil keputusan dalam memproses data/informasi bagi pemakainya.
2. Membantu pengambilan keputusan dalam hal penghematan waktu yang dibutuhkan untuk memecahkan masalah terutama berbagai masalah yang sangat kompleks dan tidak terstruktur.
3. Dapat menghasilkan solusi dengan lebih cepat serta hasilnya dapat diandalkan.
4. Walaupun suatu Sistem Pendukung Keputusan, mungkin saja tidak mampu memecahkan masalah yang dihadapi oleh pengambil keputusan, namun dapat menjadi stimulant bagi pengambil keputusan dalam memahami persoalannya, Karena sistem pendukung keputusan mampu menyajikan berbagai alternatif

### II.2.3. Komponen Sistem Pendukung Keputusan

Untuk dapat menerapkan SPK, ada 4 komponen subsistem yang harus disediakan yaitu (Syukron Hidayat dan Imam Mukhlash, 2015) :

1. Subsistem manajemen data

   Subsistem ini menyediakan data bagi sistem, termasuk didalamnya basis data. Berisi data yang relevan untuk situasi dan diatur oleh perangkat lunak yang disebut *Database Management System* (DBMS).
2. Subsistem manajemen model

Subsistem ini berfungsi sebagai pengelola berbagai model, mulai dari model keuangan, statistik, matematik, atau model kuantitatif lainnya yang memiliki kemampuan analisis dan manajemen perangkat lunak yang sesuai. Perangkat lunak ini sering disebut *Model Base Management System* (MBMS).

3. Subsistem manajemen pengetahuan

   Subsistem ini mendukung berbagai subsistem lainnya, atau dapat dikatakan berperan sebagai komponen yang independen. Subsistem ini menyediakan intelegensi untuk menambah pertimbangan pengambil keputusan.

4. Subsistem manajemen antar muka pengguna

   Subsistem ini berupa tampilan yang disediakan yang mampu mengintegrasikan sistem terpasang dengan pengguna secara interaktif. Melalui subsistem ini pengguna dapat berkomunikasi dengan sistem pendukung keputusan serta memerintah sistem pendukung keputusan.

## II.3. Metode *Multi Factor Evaluation Procces* (MFEP)

*Multifactor Evaluation Process* (MFEP) adalah metode kuantitatif yang menggunakan. '*weighting system'*. Dalam pengambilan keputusan multifaktor, pengambil keputusan secara subyektif dan intuitif menimbang berbagai faktor yang mempunyai pengaruh penting terhadap alternatif pilihan mereka. Untuk keputusan yang berpengaruh secara strategis, lebih dianjurkan menggunakan sebuah pendekatan kuantitatif seperti MFEP. Dalam MFEP pertama-tama seluruh kriteria yang menjadi faktor penting dalam melakukan pertimbangan diberikan pembobotan *(weighting*) yang sesuai. Langkah yang sama juga dilakukan terhadap

alternatif-alternatif yang akan dipilih, yang kemudian dapat dievaluasi berkaitan dengan faktor-faktor pertimbangan tersebut. Metode MFEP menentukan bahwa alternatif dengan nilai tertinggi adalah solusi terbaik berdasarkan kriteria yang telah dipilih (Ahmad Khaidir, 2014).

Langkah-langkah proses perhitungan menggunakan metode MFEP, yaitu:

1. Menentukan faktor dan bobot faktor dimana total pembobotan harus sama dengan 1 (∑ pembobotan = 1), yaitu factor *weight*.
2. Mengisikan nilai untuk setiap faktor yang mempengaruhi dalam pengambilan keputusan dari data-data yang akan diproses, nilai yang dimasukkan dalam proses pengambilan keputusan merupakan nilai objektif, yaitu sudah pasti yaitu *factor evaluation* yang nilaianya antara 0 -1.
3. Proses perhitungan *weight evaluation* yang merupakan proses perhitungan bobot antara *factor weight* dan *factor evaluation* dengan serta penjumlahan seluruh hasil *weight evaluations* untuk memperoleh total hasil evaluasi.

Penggunaan model *Multifactor Evaluation Process* (MFEP) dapat direalisasikan dengan contoh berikut:

$$WE = FW \times E$$

$$\sum WE = \sum (FW \times E) \quad \text{.....(II.1)}$$

Keterangan :

WE = *Weighted Evaluation*

FW = *Factor Weight*

E = *Evaluation*

∑WE = *Total Weighted Evaluation*

## II.4. *Microsoft Visual Basic 2010*

Pada akhir tahun 1999, Teknologi .NET diumumkan. Microsoft memosisikan teknologi tersebut sebagai *platform* untuk membangun XML Web *Services*. XML Web *services* memungkinkan aplikasi tipe manapun dan dapat mengambil data yang tersimpan pada server dengan tipe apapun melalui internet.

Visual Basic.NET adalah Visual Basic yang direkayasa kembali untuk digunakan pada *platform* .NET sehingga aplikasi yang dibuat menggunakan Visual Basic .NET dapat berjalan pada sistem komputer apa pun, dan dapat mengambil data dari server dengan tipe apa pun asalkan terinstal .NET Framework. (Priyanto Hidayatullah, 2012 ; 5).

## II.5. *SQL Server 2008*

SQL *(Structured Query Language)* adalah sebuah bahasa yang dipergunakan umtuk mengakses data dalam basis data relasional. Bahasa ini secara *de facto* merupakan bahasa standar yang dipergunakan dalam manajemen basis data relasional. Saat ini hampir semua server basis data yang ada mendukung bahasa ini untuk melakukan manajemen datanya.

*SQL* terdiri dari dua bahasa, yaitu *Data Definition Language Manipulation Language (DML)*. Implementasi DDL dan DML sistem manajemen basis data *(SMBD)*,

namun secara umum implemen bahasa ini memiliki bentuk standar yang ditetapkan oleh *ANSI*. (Adelia, Jimmy Setiawan : 2011 ; 115 ).

1. *Data Defenition Language (DDL)*

   *DDL* digunakan untuk mendefinisikan, mengubah, serta menghapus basis data dan objek-objek yang diperlukan dalam basis data, misalnya tabel, *view*, *user*, dan sebagainya. *DDL* biasanya digunakan oleh administrator basis data dalam pembuatan sebuah aplikasi basis data. Secara umum *DDL* yang digunakan adalah:

   a. *CREATE* untuk membuat objek baru.
   b. *USE* untuk menggunakan objek yang sudah ada.
   c. *ALTER* untuk mengubah objek yang sudah ada.
   d. *DROP* untuk menghapus objek.

2. *Data Manipulation Language (DML)*

   *DML* digunakan untuk memanipulasi data yang ada dalam suatu tabel. Perintah-perintah yang umum dilakukan adalah:

   a. *SELECT* untuk menampilkan data.
   b. *INSERT* untuk menambahkan data baru.
   c. *UPDATE* untuk mengubah data yang sudah ada.
   d. *DELETE* untuk menghapus data.

## II.6. Pengertian Basis Data

Basis data dapat dipahami sebagai suatu kumpulan data terhubung (*interrelated data*) yang disimpan secara bersama-sama pada suatu media, tanpa *mengatap* satu sama lain atau tidak perlu suatu kerangkapan data (kalaupun ada

maka kerangkapan data tersebut harus seminimal mungkin dan terkontrol [*controlled redudancy*]), data disimpan dengan cara-cara tertentu sehingga mudah digunakan/atau ditampilkan kembali; data dapat digunakan oleh satu atau lebih program-program aplikasi secara optimal; data disimpan tanpa mengalami ketergantungan dengan program yang akan menggunakannya; data disimpan sedemikian rupa sehingga proses penambahan, pengambilan, dan modifikasi data dapat dilakukan dengan mudah dan terkontrol (Edy Sutanta, 2011 : 29-30).

## II.7. Normalisasi

Menurut Martin (1995), Normalisasi diartikan sebagai suatu teknik yang menstrukurkan/ mendekomposisi data dalam cara-cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data. Permasalahan yang dimaksud adalah berkaitan dengan penyimpangan-penyimpangan *(anomalies)* yang terjadi akibat adanya kerangkapan data dalam relasi dan in-efisiensi pengolahan (Edy Sutanta ; 2011 : 174).

Proses normalisasi menghasilkan relasi yang optimal, yaitu (Edy Sutanta ; 2011 : 175) :

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.

Secara berturut-turut masing-masing level normal tersebut dibahas berikut ini, dimulai dari bentuk tidak normal. (Edy Sutanta ; 2011 : 176-179)

1. Relasi bentuk tidak normal (*Un Normalized Form* / UNF)

   Relasi-relasi yang dirancang tanpa mengindahkan batasan dalam defisi basis data dan karakteristik *Relational Database Management System* (RDBM) menghasilkan relasi *Un Normalized Form* (UNF). Bentuk ini harus di hindari dalam perancangan relasi dalam basis data. Relasi *Un Normalized Form* (UNF) mempunyai kriteria sebagai berikut.

   a. Jika relasi mempunyai bentuk *non flat file* (dapat terjadi akibat data disimpan sesuai dengan kedatangannya, tidak memiliki struktur tertentu, terjadi duplikasi atau tidak lengkap)
   b. Jika relasi membuat *set atribut* berulang *(non single values)*
   c. Jika relasi membuat *atribut non atomic value*

2. Relasi bentuk normal pertama (*First Norm Form* / 1NF)

   Relasi disebut juga *First Norm Form* (1NF) jika memenuhi kriteria sebagai berikut.

   a. Jika seluruh atribut dalam relasi bernilai *atomic ( atomic value)*
   b. Jika seluruh atribut dalam relasi bernilai tunggal *(single value)*
   c. Jika relasi tidak memuat set atribut berulang
   d. Jika semua record mempunyai sejumlah atribut yang sama.

   Permasalahan dalam *First Norm Form* (1NF) adalah sebagai berikut.

   a. Tidak dapat menyisipkan informasi parsial

b. Terhapusnya informasi ketika menghapus sebuah *record*

3. Bentuk normal kedua *(Second Normal Form /* 2NF)

Relasi disebut sebagai *Second Normal Form (*2NF) jika memenuhi kriteria sebagai berikut

a. Jika memenuhi kriteria *First Norm Form (*1NF).
b. Jika semua atribut nonkunci *Functional Dependence* (FD) pada *Primary Key* (PK).

Permasalahan dalam *Second Normal Form /* 2NF adalah sebagai berikut

a. Kerangkapan data *(data redundancy)*
b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data *(data inconsistency)*
c. Proses pembaharuan data tidak efisien

Kriteria tersebut mengidentifikasikan bahwa antara atribut dalam *Second Normal Form* masih mungkin mengalami *Third Norm Form*. Selain itu, relasi *Second Normal Form* (2NF) menuntut telah didefinisikan atribut *Primary Key* (PK) dalam relasi. Mengubah relasi *First Norm Form* (1NF) menjadi bentuk *Second Normal Form* (2NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

a. Identifikasikan *Functional Dependence* (FD) relasi *First Norm Form* (1NF)
b. Berdasarkan informasi tersebut, dekomposisi relasi *First Norm Form* (1NF) menjadi relasi-relasi baru sesuai *Functional Dependence* nya. Jika menggunakan diagram maka simpul-simpul yang berada pada puncak

diagram ketergantungan data bertindak *Primary Key* (PK) pada relasi baru

4. Bentuk normal ketiga (*Third Norm Form* / 3NF)

   Suatu relasi disebut sebagai *Third Norm Form* jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Second Normal Form* (2NF)
   b. Jika setiap atribut nonkunci tidak (TDF) *(Non Transitive Dependeny)* terhadap *Primary Key* (PK)

   Permasalahan dalam *Third Norm Form* (3NF) adalah keberadaan penentu yang tidak merupakan bagian dari *Primary Key* (PK) menghasilkan duplikasi rinci data pada atribut yang berfungi sebagai *Foreign Key* (FK) (duplikasi berbeda dengan keterangan data).

   Mengubah relasi *Second Normal Form* (2NF) menjadi bentuk *Third Norm Form* (3NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

   a. Identifikasi TDF relasi *Second Normal Form* (2NF)
   b. Berdasarkan informasi tersebut, dekomposisi relasi *Second Normal Form* (2NF) menjadi relasi-relasi baru sesuai TDF-nya.

5. Bentuk normal *Boyce-Cood (Boyce-Codd Norm Form* / BCNF)

   Bentuk normal *Boyce-Codd Norm Form* (BCNF) dikemukakan oleh R.F. Boyce dan E.F. Codd. Suatu relasi disebut sebagai *Boyce-Codd Norm Form* (BCNF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Third Norm Form* (3NF)
   b. Jika semua atribut penentu (determinan) merupakan CK

6. Bentuk normal keempat (*Forth Norm Form* / 4NF)

   Relasi disebut sebagi *Forth Norm Form* (4NF) jika memenuhi kriteria sebagai berikut.

   a. Jika memenuhi kriteria *Boyce-Codd Norm Form.*
   b. Jika setiap atribut didalamnya tidak mengalami ketergantungan pada banyak nilai.

7. Bentuk normal kelima (*Fifth Norm Form* / 5NF)

   Suatu relasi memenuhi kriteria *Fifth Norm Form* (5NF) jika kerelasian antar data dalam relasi tersebut tidak dapat direkonstruksi dari struktur relasi yang sederhana.

8. Bentuk normal kunci domain (*Domain Key Norm Form* / DKNF)

   Relasi disebut sebagai *Domain Key Norm Form* (DKNF) jika setiap batasan dapat disimpulkan secara sederhana dengan mengetahui sekumpulan nama atribut dan domainnya selama menggunkan sekumpulan atribut pada kuncinya.

## II.8. *Unified Modeling Language* (UML)

Menurut Henderi (2012:152), *Unified Modelling Language* (UML) merupakan satu kumpulan konvensi pemodelan yang digunakan untuk menentukan atau menggambarkan sebuah sistem software yang terkait dengan objek. 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 mengkomunikasikan beberapa aspek dalam sistem melalui jumlah elemen grafis yang bisa dikombinasikan menjadi diagram (Rosana Junita Sirait, et al., 2015).

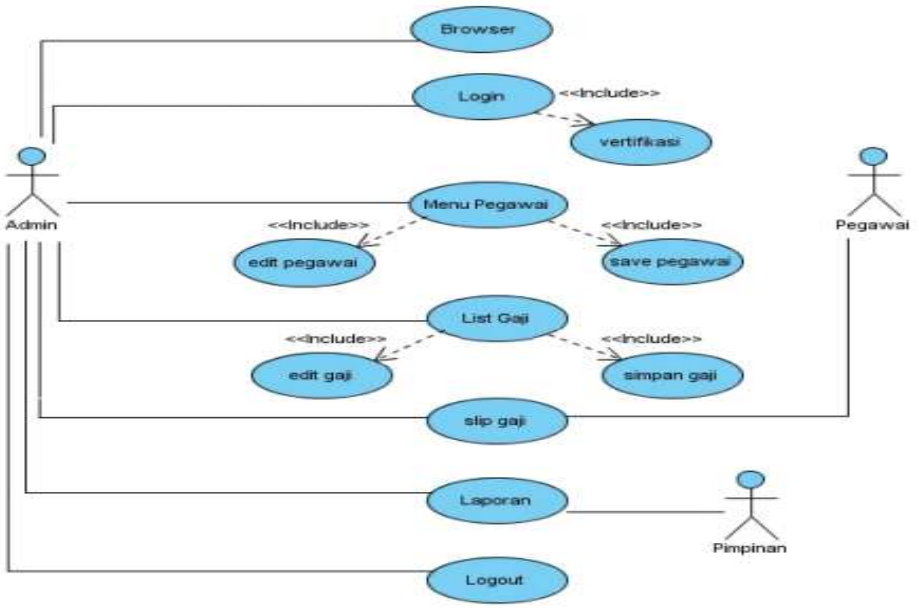
Menurut Rama (2008:111), "*Unified Modeling Language* (UML) adalah suatu bahasa permodelan untuk menyebutkan, memvisualisasikan, membuat dan mendokumentasikan sistem informasi." Menurut Henderi (2007:4), "*Unified Modeling Language* (UML) adalah sebuah bahasa pemodelan yang telah menjadi standar dalam industri *software* untuk visualisasi, merancang, dan mendokumentasikan sistem perangkat lunak." Bahasa pemodelan UML lebih cocok untuk pembuatan perangkat lunak dalam bahasa pemrograman berorientasi objek (C+, Java, VB.NET), namun demikian tetap dapat digunakan pada bahasa pemrograman prosedural.

Berdasarkan beberapa pendapat dikemukakan diatas dapat ditarik kesimpulan bahwa "*Unified Modeling Language* (UML) adalah sebuah bahasa yang berdasarkan grafik atau gambar untuk menvisualisasikan, menspesifikasikan, membangun dan pendokumentasian dari sebuah sistem pengembangan perangkat lunak berbasis OO (*Object Oriented*) (Aris, et al., 2015).

### II.8.1. *Use Case* Diagram

*Use case* adalah deskripsi dari sebuah sistem dari perspektif pengguna. *Use case* bekerja dengan cara mendeskripsikan tipikal interaksi antara *user* (pengguna) sebuah sistem dengan sistemnya sendiri melalui sebuah cerita bagaimana sebuah sistem dipakai (Oktafiansyah, 2012).

Suatu *use case* diagram menampilkan sekumpulan *use case* dan aktor (pelaku) dan hubungan diantara *use case* dan aktor tersebut. *Use case* diagram digunakan untuk penggambaran *use case* statik dari suatu sistem. *Use case* diagram penting dalam mengatur dan memodelkan kelakuan dari suatu sistem. *Use case* menjelaskan apa yang dilakukan sistem (atau subsistem) tetapi tidak menspesifikasi cara kerjanya. *Flow of event* digunakan untuk menspesifikasi cara kerjanya kelakuan dari *use case*. *Flow of event* menjelaskan *use case* dalam bentuk tulisan dengan sejelas-jelasnya, dintaranya bagaimana, kapan *use case* dimulai dan berakhir, ketika *use case* berinteraksi dengan aktor, objek apa yang digunakan, alur dasar dan alur alternatif. Terdapat beberapa simbol dalam menggambarkan diagram *use case*, yaitu *use cases*, aktor dan relasi (Achmad Hamzah Nasrullah dan Dadang Sudrajat, 2015).

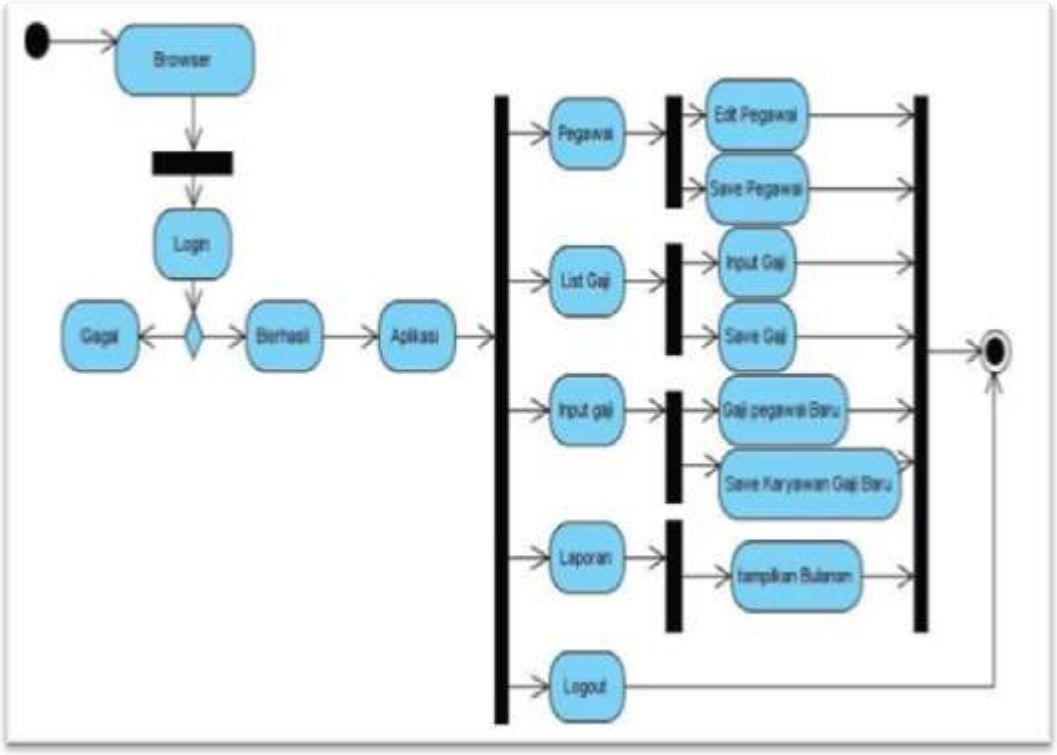



**Gambar II.1. *Use Case* Diagram**
**(Sumber : Aris, et al., 2015)**

### II.8.2. *Activity* Diagram

*Activity* diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus (Oktafiansyah, 2012).

*Activity* diagram memperlihatkan alur langkah demi langkah dalam suatu proses. Suatu aktivitas menunjukkan sekumpulan aksi (secara sekuensial atau bercabang dari satu aksi ke aksi lain), dan nilai yang dihasilkan atau digunakan oleh aksi-aksi yang terjadi. Activity diagram ditunjukkan untuk memodelkan fungsi dari suatu sistem dan menekankan pada alur dari kontrol didalam pelaksanaan dari suatu tindakan (Achmad Hamzah Nasrullah dan Dadang Sudrajat, 2015).

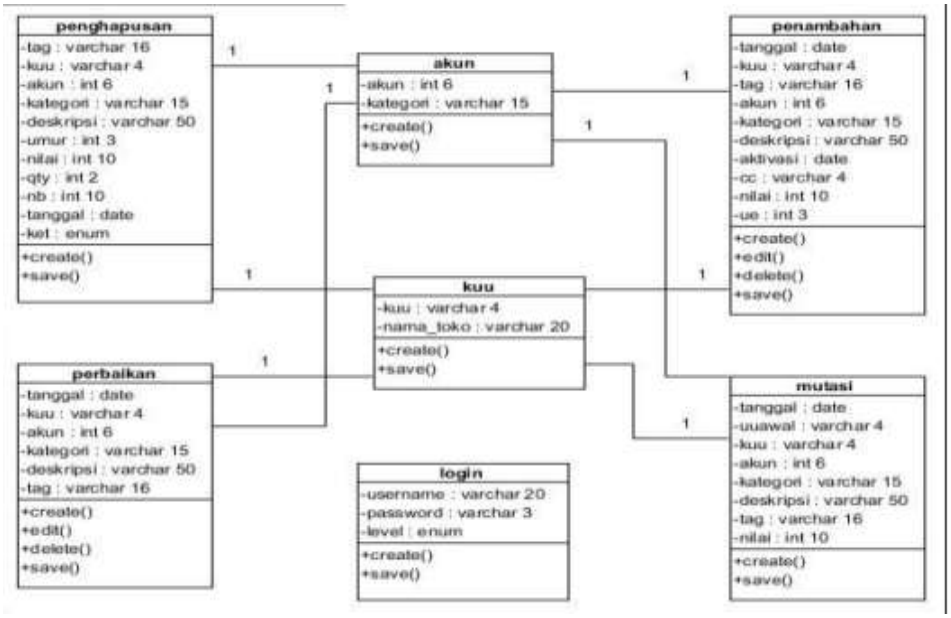



**Gambar II.2. *Activity* Diagram**
**(Sumber : Aris, et al., 2015)**

### II.8.3. *Class* Diagram

*Class* diagram menunjukan sekumpulan kelas, antarmuka, dan kerjasama serta hubungannya. *Class* diagram digunakan untuk memodelkan perancangan statik dari gambaran sistem. Biasanya meliputi pemodelan *vocabulary* dari sistem, pemodelan kerjasama, atau pemodelan skema. *Class* diagram dapat digunakan untuk membangun sistem yang dapat dieksekusi melalui teknik *forward and reverse*, selain untuk penggambaran, menspesifikasikan, dan pendokumentasian struktur model (Achmad Hamzah Nasrullah dan Dadang Sudrajat, 2015).

*Class diagram* menggambarkan struktur statis dari kelas dalam sistem anda dan menggambarkan atribut, operasi dan hubungan antara kelas. *Class diagram* membantu dalam memvisualisasikan struktur kelas-kelas dari suatu sistem dan merupakan tipe diagram yang paling banyak dipakai. Selama tahap desain, *class diagram* berperan dalam menangkap struktur dari semua kelas yang membentuk arsitektur sistem yang dibuat (Haviluddin, 2011).



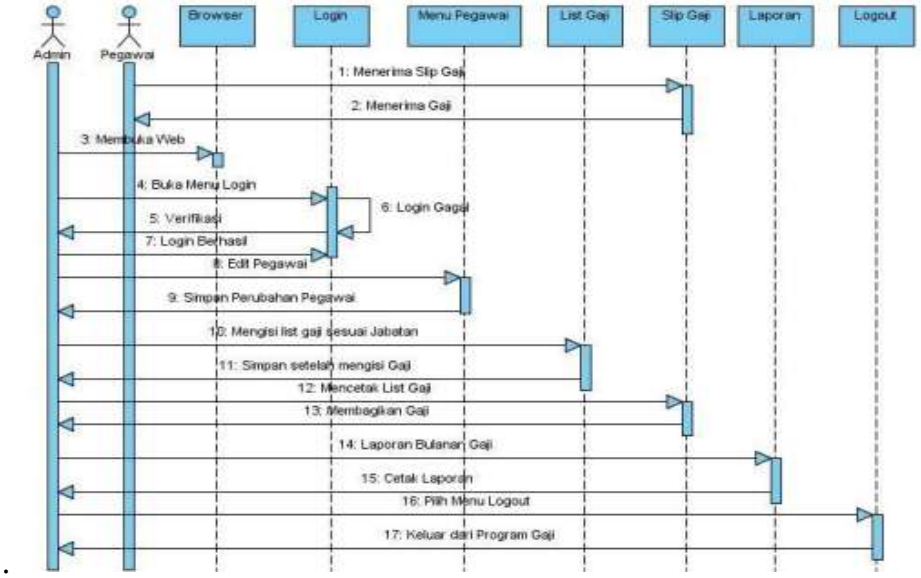

**Gambar II.3. *Class* Diagram**

**(Sumber : Rosana Junita Sirait, et al., 2015)**

### II.8.4. *Sequence* Diagram

*Sequence* Diagram digunakan untuk menggambarkan perilaku pada sebuah skenario. Diagram ini menunjukkan jumlah contoh obyek dan *message* (pesan) yang diletakkan diantara obyek-obyek ini di dalam *use case* (Oktafiansyah, 2012).

*Sequence diagram* menjelaskan interaksi objek yang disusun berdasarkan urutan waktu. Secara mudahnya *sequence diagram* adalah gambaran tahap demi tahap, termasuk kronologi (urutan) perubahan secara logis yang seharusnya dilakukan untuk menghasilkan sesuatu sesuai dengan *use case diagram* (Haviluddin, 2011).




**Gambar II.4. *Sequence* Diagram**

**(Sumber : Aris, et al., 2015)**