

BAB II

LANDASAN TEORI

II.1. Sistem

Setiap sistem pasti terdiri dari struktur dan proses, struktur sistem merupakan unsur-unsur yang membentuk sistem tersebut, sedangkan proses sistem menjelaskan cara kerja dari setiap unsur sistem tersebut dalam mencapai tujuan sistem. Dan dapat disimpulkan bahwa suatu sistem pada dasarnya adalah sekelompok unsur yang erat berhubungan satu dengan yang lainnya, yang berfungsi bersama-sama untuk mencapai tujuan tertentu (Rosana Junita Sirait, et al : 2015 ; 2).

Di dalam mendefinisikan sistem pengertian sistem ada dua kelompok, pendekatan yang harus diperhatikan yang menekankan pada prosedur dan menekankan pada komponen atau elemennya.

Sistem adalah jaringan kerja prosedur – prosedur yang berhubungan, berkumpul bersama – sama untuk melakukan kegiatan dalam rangka mencapai tujuan tertentu. Sistem juga dapat didefinisikan sebagai suatu jaringan prosedur yang dibuat menurut pola yang terpadu untuk melaksanakan kegiatan pokok perusahaan. Sistem adalah suatu kerangka dari prosedur –prosedur yang saling berhubungan yang disusun dengan suatu skema yang menyeluruh untuk melaksanakan suatu kegiatan atau fungsi utama dari perusahaan. (Herry Langi, et al; 2016; 96).

II.2. Pengertian Keputusan

Keputusan adalah “hasil pemecahan masalah yang dihadapi dengan tegas”. Suatu keputusan merupakan jawaban terhadap suatu pertanyaan. Keputusan harus dapat menjawab pertanyaan: Tentang apa yang seharusnya dilakukan dan apa yang dibicarakan dalam hubungannya dengan perencanaan. Keputusan pun dapat merupakan tindakan terhadap pelaksanaan yang sangat menyimpang dari rencana semula. Keputusan yang baik pada dasarnya dapat digunakan untuk membuat rencana dengan baik pula. (Samsuri Ridwan, et al; 2014; 42).

II.3. Sistem Pendukung Keputusan

Menurut Bonczek dalam buku (Turban, et.al;2005) Sistem pendukung Keputusan sebagai sebuah sistem berbasis komputer yang terdiri atas komponen-komponen antara lain komponen sistem bahasa (language), komponen sistem pengetahuan (knowledge) dan komponen sistem pemrosesan masalah (problem processing) yang saling berinteraksi satu dengan yang lainnya. Menurut Alter dalam buku (Kusrini, 2007), Sistem pendukung keputusan merupakan sistem informasi interaktif yang menyediakan informasi, pemodelan, dan pemanipulasian data dimana sistem yang digunakan untuk membantu pengambilan keputusan dalam situasi yang semiterstruktur dan situasi yang tidak terstruktur. Hal yang perlu diperhatikan di sini adalah bahwa keberadaan SPK bukan untuk menggantikan tugas manajer, tetapi untuk menjadi sarana penunjang bagi mereka. SPK merupakan implementasi teori teori pengambilan keputusan yang telah diperkenalkan oleh ilmu ilmu seperti

operation research dan management science. Hanya bedanya adalah bahwa dahulu untuk mencari penyelesaian masalah yang dihadapi harus dilakukan perhitungan iterasi secara manual. Dalam kedua bidang ilmu di atas, dikenal istilah decision modeling, decision theory, decision analysis yang pada hakekatnya adalah merepresentasikan permasalahan manajemen yang dihadapi setiap hari ke dalam bentuk kuantitatif (Frans Ikorasaki , 2015 ; 105-16).

Model yang menggambarkan proses pengambilan keputusan. Proses ini terdiri dari tiga fase, yaitu sebagai berikut :

a. *Intelligence*

Tahap ini merupakan proses penelusuran dan pendeteksian dari lingkup problematika serta proses pengenalan masalah. Data masukan diperoleh, diproses, dan diuji dalam rangka mengidentifikasi masalah.

b. *Design*

Tahap ini merupakan proses menemukan, mengembangkan, dan menganalisis alternatif tindakan yang bisa dilakukan. Tahap ini meliputi proses untuk mengerti masalah, menurunkan solusi dan menguji kelayakan solusi.

c. *Choice*

Pada tahap ini dilakukan proses pemilihan diantara berbagai alternatif tindakan yang mungkin dijalankan. Hasil pemilihan tersebut kemudian diimplementasikan dalam proses pengambilan keputusan.

Meskipun implementasi termasuk tahap ketiga, namun ada beberapa pihak berpendapat bahwa tahap ini perlu dipandang sebagai bagian yang terpisah guna

menggambarkan hubungan antar fase secara lebih komprehensif. (Hilya Magdalena, 2012 : 50).

III.3.1.Karakteristik dan Keterbatasan Sistem Pendukung Keputusan

Suatu SPK memiliki tiga subsistem utama yang menentukan kapabilitas teknis SPK tersebut, yaitu: subsistem manajemen berbasis model, dan subsistem perangkat lunak penyelenggara dialog.(Hermawan Ardiyanto, 2013; 51).

1. Subsistem manajemen basis data

Subsistem ini berfungsi untuk menyimpan data-data yang dihasilkan dari *internal, eksternal* perumahan dan data *privat*.

2. Subsistem manajemen berbasis model

Subsistem ini berfungsi untuk menyederhanakan permasalahan, sehingga masalah lebih mudah dipahami.

3. Subsistem perangkat lunak penyelenggara dialog

Subsistem ini berfungsi untuk berkomunikasi antara pengguna dengan sistem.

SPK, menurut tinjauan konotatif, merupakan system yang ditujukan kepada tingkatan manajemen yang lebih tinggi, dengan penekanan karakteristik sebagai berikut:

1. Berfokus pada keputusan., ditujukan pada manajer puncak dan pengambil keputusan.
2. Menekankan pada fleksibilitas, adaptabilitas, dan respon yang cepat.

3. Mampu mendukung berbagai gaya pengambilan keputusan dan masing-masing pribadi manajer. (Hilya Magdalena, 2012 : 51).

Adapaun keterbatasan sistem pendukung keputusan adalah sebagai berikut:

1. Adanya gambaran bahwa SPK seakan-akan hanya dibutuhkan pada tingkat manajemen puncak. Pada kenyataannya, dukungan bagi pengambilan keputusan dibutuhkan pada semua tingkatan manajemen dalam suatu organisasi.
2. Pengambilan keputusan yang terjadi pada beberapa level harus dikoordinasikan. Jadi, dimensi dan pendukung keputusan adalah komunikasi dan koordinasi diantara pengambil keputusan antar level organisasi yang berbeda maupun pada level organisasi yang sama. (Hilya Magdalena, 2012 : 50).

II.4. *Analytical Hierarchy Process (AHP)*

AHP dikembangkan pada tahun 1970an oleh Dr Thomas L. Satty untuk menyediakan pendekatan sistematis untuk menentukan prioritas dan pengambilan keputusan dalam suatu kompleks lingkungan. AHP dirancang untuk mencerminkan cara berpikir orang sebenarnya. Metode ini memungkinkan aspek kuantitatif dan kualitatif keputusan yang akan dipertimbangkan. AHP mengurangi keputusan yang kompleks menjadi sebuah rangkaian satu-satu pada perbandingan yang kemudian memberikan hasil yang akurat. AHP juga menggunakan skala

rasio untuk bobot kriteria dan scoring alternatif yang menambahkan untuk pengukuran presisi. (Hilya Magdalena, 2012 : 51).

Karena sulitnya menentukan bobot-bobot ataupun prioritas-prioritas yang sering berubah-ubah, digunakan perbandingan berpasangan yang menggunakan data, pengetahuan, dan pengalaman membuat penilaian berkenaan dengan pertimbangan relatif pentingnya satu elemen terhadap yang lain. Untuk itu diperlukan suatu skala perbandingan antar dua elemen, baik secara kualitatif maupun kuantitatif. (Hilya Magdalena, 2012 : 52).

Untuk kegiatan perbandingan antar sepasang objek, metode AHP memberikan sebuah standar nilai perbandingan antar dua objek seperti dituangkan pada tabel II.1.

Tabel II.1. Nilai Perbandingan

Pembandingan	Nilai
Sangat diutamakan	9
Lebih diutamakan menuju sangat diutamakan	8
Lebih diutamakan	7
Diutamakan menuju lebih diutamakan	6
Diutamakan	5
Cukup diutamakan menuju diutamakan	4
Cukup diutamakan	3
Setara menuju cukup diutamakan	2
Setara	1

(Sumber : Hilya Magdalena, 2012 : 52)

II.4.1. Proses-proses dalam AHP

Adapun proses-proses yang harus dilakukan pada metode AHP adalah sebagai berikut (I Nyoman Yudha Astana, 2013 : 3) :

1. Mendefinisikan masalah dan menentukan solusi yang diinginkan.
2. Membuat struktur hirarki yang diawali tujuan umum dilanjutkan dengan kriteria dan kemungkinan alternatif pada tingkatan kriteria paling bawah.
3. Membuat matrik perbandingan berpasangan yang menggambarkan kontribusi relatif atau pengaruh setiap elemen terhadap kriteria yang setingkat di atasnya.
4. Melakukan perbandingan berpasangan sehingga diperoleh *judgment* (keputusan) sebanyak $n \times ((n-1)/2)$ bh, dengan n adalah banyaknya elemen yang dibandingkan.
5. Menghitung nilai *eigen* dan menguji konsistensinya jika tidak konsisten maka pengambilan data diulangi lagi.
6. Mengulangi langkah 3,4 dan 5 untuk setiap tingkatan hirarki.
7. Menghitung *vector eigen* dari setiap matrik perbandingan berpasangan.

$$CI = (\text{maks}-n)/n \quad (\text{.....1})$$

Dimana n = banyaknya elemen

Hitung Rasio Konsistensi atau Consistency Ratio (CR) dengan

rumus:

$$CR = CI/IR \quad (\text{.....2})$$

Dimana CR = Consistency Ratio

CI = Consistency Index

IR = Index Random Consistency

8. Memeriksa konsistensi hierarki. Jika nilainya lebih dari 10%, maka penilaian data judgement harus diperbaiki. Namun jika rasio konsistensi (CI/IR) kurang atau sama dengan 0,1, maka hasil perhitungan bisa dinyatakan benar. Berdasarkan perhitungan Thomas L. dengan menggunakan 500 sampel diperoleh nilai rata-rata indeks random (RI) untuk setiap ordo matrik tertentu.

Tabel II.2. Daftar Indeks Random Konsistensi

Ukuran Matriks	Nilai IR
1,2	0,00
3	0,58
4	0,90
5	1,12
6	1,24
7	1,32
8	1,41
9	1,45
10	1,49
11	1,51
12	1,48
13	1,56
14	1,57
15	1,59

Sumber : Syahrani Dhimas Prabowo, Eko Budi Setiawan (2013 : 29)

II.5. SMART

Pada hakekatnya SMART merupakan suatu model pengambil keputusan yang komprehensif dengan memperhitungkan hal-hal yang bersifat kualitatif dan kuantitatif. Dalam model pengambilan keputusan dengan SMART pada dasarnya berusaha menutupi setiap kekurangan dari model-model tanpa komputerisasi sebelumnya. SMART juga memungkinkan ke struktur suatu sistem dan lingkungan kedalam komponen saling berinteraksi dan kemudian menyatukan mereka dengan mengukur dan mengatur dampak dari komponen kesalahan sistem.

Peralatan utama dari model ini adalah sebuah hirarki fungsional dengan input utamanya adalah persepsi manusia. Jadi pada dasarnya tidak ada perbedaan yang mencolok pada model SMART dengan model lainnya sama sama terletak pada jenis inputnya, hanya saja terdapat persentase dari setiap pembobotan langsung yang ditentukan oleh hasil analisa permasalahan.

Metode pembobotan SMART merupakan metode pendukung keputusan yang paling sederhana. Dalam metode ini dilihat beberapa parameter yang menjadi penentu keputusan tersebut. Parameter tersebut mempunyai range nilai dan bobot yang berbeda-beda. Nilai tersebut nantinya akan menjadi penentu keputusan yang diambil (Eva Yulianti :2015).

II.5.1. Perhitungan Metode SMART

Adapun langkah-langkah dalam proses perhitungan metode *SMART* dapat ditunjukkan sebagai berikut (Kustiyahningsih & Syafa'ah, 2015, hal. 22-23).

1. Langkah 1: menentukan jumlah kriteria
2. Langkah 2: sistem secara default memberikan skala 0-100 berdasarkan prioritas yang telah diinputkan kemudian dilakukan normalisasi.

$$\text{Normalisasi} = \frac{w_j}{\sum w_j} \quad (1)$$

Keterangan : w_j : bobot suatu kriteria

$\sum w_j$: total bobot semua kriteria

- II.1. Langkah 3: memberikan nilai kriteria untuk setiap alternatif.
- II.2. Langkah 4: hitung nilai utility untuk setiap kriteria masing-masing.

$$u_i(a_i) = 100 \frac{(C_{out\ i} - C_{min})}{(C_{max} - C_{min})} \% \quad (2)$$

Keterangan :

$u_i(a_i)$: nilai utility kriteria ke-1 untuk kriteria ke-i

C_{max} : nilai kriteria maksimal

C_{min} : nilai kriteria minimal

$C_{out\ i}$: nilai kriteria ke-i

II.3. Langkah 5: hitung nilai akhir masing-masing. (3)

$$u(a_i) = \sum_{j=1}^m w_j u_j(a_i),$$

II.6. Basis Data

Basis data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti :

1. Himpunan kelompok data (arsip) yang saling berhubungan yang diorganisasikan sedemikian rupa agar kelak dapat dimanfaatkan kembali dengan cepat dan mudah.
2. Kumpulan data yang saling berhubungan disimpan secara bersama sedemikian rupa dan tanpa pengulangan yang yang perlu untuk memenuhi berbagai kebutuhan.
3. Kumpulan *file* atau tabel atau arsip yang saling berhubungan yang disimpan dalam media penyimpanan elektronik. (Akhmad Sholikhin, Kususma Riasti, April 2013 : 51).

Berdasarkan tingkat kompleksitas nilai data, tingkatan data dapat disusun dalam sebuah hierarki, mulai dari yang paling sederhana hingga paling sederhana hingga paling kompleks.

1. Sistem basis data, merupakan sekumpulan subsistem yang terdiri atas basis data dengan para pemakai yang menggunakan basis data secara bersama-sama, personal-personal yang merancang dan mengelola basis data, teknik-teknik untuk merancang dan mengelola basis data, serta sistem komputer untuk mendukungnya.
2. Basis data, merupakan sekumpulan dari bermacam-macam tipe *record* yang memiliki hubungan antar-*record* dan rincian data terhadap obyek tertentu.
3. File, merupakan sekumpulan *record* sejenis secara relasi yang tersimpan dalam media penyimpanan sekunder.
4. *Record*, merupakan *field*/atribut/data item yang saling berhubungan terhadap obyek tertentu.
5. *Data item/field*/atribut, merupakan unit terkecil yang disebut data, sekumpulan *byte* yang mempunyai makna.
6. *Data aggregate*, merupakan sekumpulan data item/*field*/atribut dengan ciri tertentu dan diberi nama.
7. *Byte*, adalah bagian terkecil yang dialamatkan dalam memori. *Byte* merupakan sekumpulan *bit* yang secara konvensional terdiri atas kombinasi 8 *bit* biner yang menyatakan sebuah karakter dalam memori (1 *byte* = 1 karakter).

8. *Bit*, adalah sistem biner yang terdiri atas dua macam nilai, yaitu 0 dan 1. Sistem biner merupakan dasar yang dapat digunakan untuk komunikasi antara manusia dan mesin (komputer). (Edy Sutanta, 2011 : 35-36).

II.7. Kamus Data

Kamus data adalah katalog fakta tentang data dan kebutuhan informasi suatu sistem informasi. Kamus data terdapat pada tahapan analisis dan perancangan. Pada tahap analisis, kamus data berfungsi untuk mendefinisikan data yang mengalir pada sistem. Sedangkan pada tahap perancangan, kamus data ini digunakan untuk merancang masukan dan keluaran seperti laporan serta basis data. Pada DFD aliran data memiliki sifat global, sedangkan pada kamus data dibuat berdasarkan aliran data yang terdapat pada DFD. (Indrajani : 2015 ; 30).

Sumber kamus data, yaitu:

1. Data Store (File-file)
2. Data Flow (Aliran data)
3. Data element yang dinyatakan dalam spesifikasi data dan berasal dari file.

Tabel II.2 Notasi Kamus Data

Notasi	Keterangan
=	Is Composed Of
+	And
()	Optional (May be present or absent)
{ }	Iteration
[]	Select one of several alternative choices

**	Comment
@	Identifier (key field) for a store
	Separates alternative choices in the construct

(Sumber : Indrajani ; 31)

Contoh kamus data, antara lain:

name = courtesy-title + first-name +(middle-name) + last-name

courtesy-title = [Mr. | Miss | Mrs. | Ms. | Dr. | Profesor]

first-name = {legal-character}

middle-name = {legal-character}

last-name = {legal-character}

legal-character = [A-Z|a-z|0-9|'|-'| |]

II.8. Normalisasi

Normalisasi diartikan sebagai suatu teknik yang menstrukturkan/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 (Martin, 1975)
: (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 definsi 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 Normal 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 Normal 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 mengidentifikasi bahwa antara atribut dalam *Second Normal Form* masih mungkin mengalami *Third Normal Form*. Selain itu, relasi *Second Normal Form* (2NF) menuntut telah didefinisikan atribut *Primary Key* (PK) dalam relasi. Mengubah relasi *First Normal Form* (1NF) menjadi bentuk *Second Normal Form* (2NF) dapat dilakukan dengan mengubah struktur relasi dengan cara :

- a. Identifikasikan *Functional Dependence* (FD) relasi *First Normal Form* (1NF)

- b. Berdasarkan informasi tersebut, dekomposisi relasi *First Normal 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 Normal Form* / 3NF)

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

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

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

Mengubah relasi *Second Normal Form* (2NF) menjadi bentuk *Third Normal 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-Codd* (*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 sebagai *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 menggunakan sekumpulan atribut pada kuncinya.

II.9. *SQL Server 2008*

SQL (*Structured Query Language*) adalah sebuah bahasa yang dipergunakan untuk 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 (DDL)* dan *Data 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, Setiawan, 2011: 115).

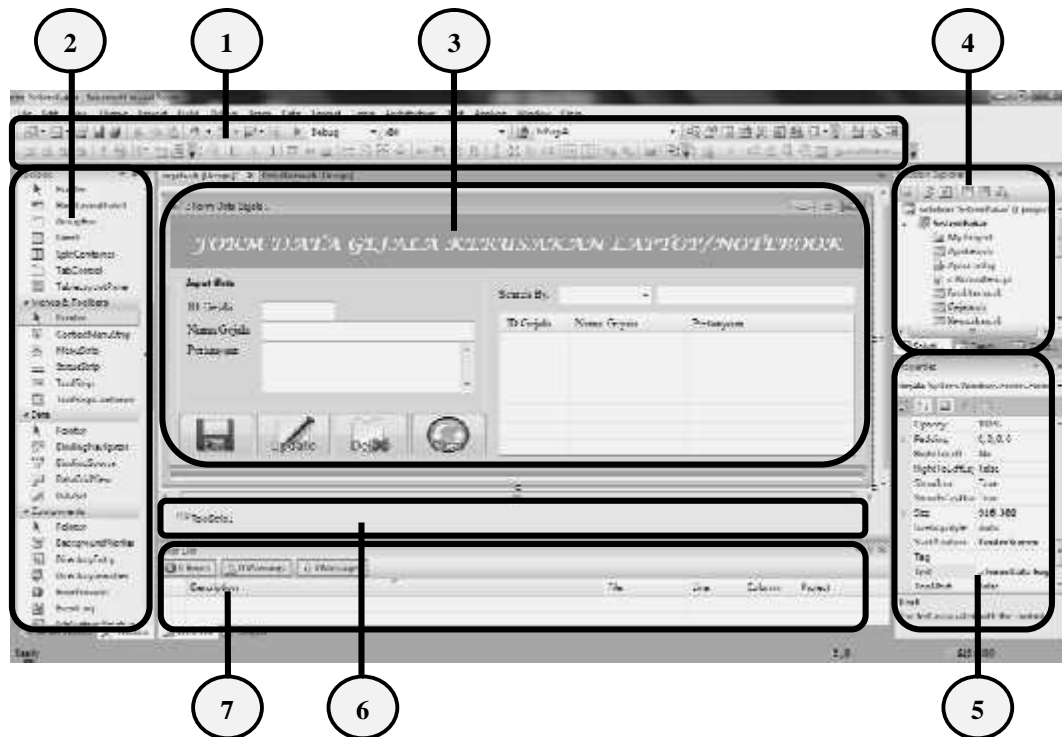
II.10. Microsoft Visual Basic 2010

Pemrograman Visual Basic Net adalah bahasa pemrograman terpopuler. Ini merupakan pemrograman yang berjalan di atas platform NET Framework. Karena itu setiap kali pemrograman VB.Net ini merilis versi barunya, tentu saja akan diikuti atau berbarengan dengan perkembangan Ner Framework terbaru. Instalasi Visual studio sebenarnya mudah pada prinsipnya. Hampir semua seragam di versi yang ada. Tapi yang harus dipersiapkan dulu adalah spek komputer yang harus memenuhi persyaratan untuk menjalankannya. (Edy Winarno, dkk., 2013 : 141).

II.11.1. Mengenal IDE Visual Studio 2010

Pada IDE (*Integrated Developing Environment*) *Visual Studio 2010* untuk *Windows Application* secara *default* telah terdapat sebuah *form*. *Form* tersebut bernama *Form1*. Pada *form* inilah tempat meletakkan kontrol-kontrol atau komponen-komponen untuk membuat sebuah aplikasi *Windows Form* dan kontrol-kontrol dari aplikasi inilah yang biasanya disebut dengan GUI (*Graphical*

User Interface). Jadi *user* akan berinteraksi dengan sebuah program aplikasi melalui GUI (Priyanto Hidayatullah, 2012 : 24). Adapun tampilan daripada *Visual Basic 2010* dapat dilihat pada Gambar II.14.



Gambar II.14 *Integrated Developing Environment of Visual Basic 2010*
(Sumber: Dodit Suprianto, 2010 : 15)

Dari Gambar II.14 dapat dilihat IDE daripada *Visual Basic 2010*, adapun keterangan dari gambar tersebut adalah sebagai berikut:

1. **Toolbar**, terdiri dari ikon-ikon sebagai jalan pintas (*shortcut*) pengganti menu *pull-down*. Ikon *toolbar* berisi perintah-perintah yang sering digunakan oleh *programmer* sehingga lebih cepat mengoperasikan perintah-perintah yang ada.
2. **Toolbox**, berisi kontrol *object* terutama pembentuk antarmuka (*interface*) *windows*, seperti *textbox*, *panel*, *button*, *groupbox*, dan lain sebagainya.

3. **Form Design**, sebuah *Form* akan terbentuk dengan sendirinya saat *project* dibuat pertama kali. *Form design* ini merupakan *Graphical User Interface* yang dirancang.
4. **Solution Explorer**, terdiri dari *object-object* pendukung *project*, seperti *form*, *module*, *report*, *data control*, dan lain sebagainya.
5. **Properties**, menampilkan *property* setiap *object* yang terpilih. Dari *Windows Property* dapat mengubah atau mengatur nilai masing-masing *properties object*.
6. **Status Object**, menampilkan daftar kontrol yang telah digunakan oleh *Form* bersangkutan. Dari kasus di atas, kontrol yang digunakan adalah *ToolStrip1*.
7. **Error List**, merupakan *window* yang akan menampilkan pesan tertentu jika terjadi kesalahan atau *error* pada saat proses pengembangan program atau kompilasi *program*.

II.12. Unified Modeling Language (UML)

UML adalah salah satu alat bantu yang sangat handal di dunia pengembangan sistem yang berorientasi objek. UML dapat menyediakan bahasa pemodelan yang mudah dimengerti oleh pengembang dan dapat dikomunikasikan dengan pemakai. Metode perancangan yang digunakan pada penelitian ini adalah metode berorientasi objek menggunakan UML. (Nyimas Sopiah, 2012 : 189).

UML (*Unified Modelling Language*) juga merupakan sebuah bahasa yang sudah menjadi standar di dunia industri untuk visualisasi, merancang dan

mendokumentasikan system piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah system.(Sentosa Pohan,2015 : 14).

II.12.1. *Artifact* dan Notasi UML

1. Diagram Use Case


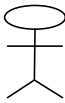
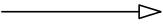
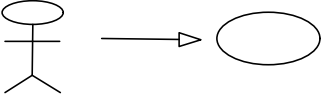
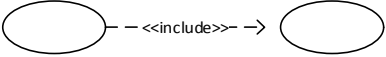
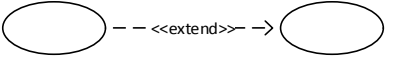
Use case menjelaskan urutan kegiatan yang dilakukan actor dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namun use case hanya menjelaskan apa yang dilakukan oleh actor dan sistem bukan bagaimana actor dan sistem melakukan kegiatan tersebut.

Berbagai simbol yang hadir didalam *use case* diagram antara lain adalah :

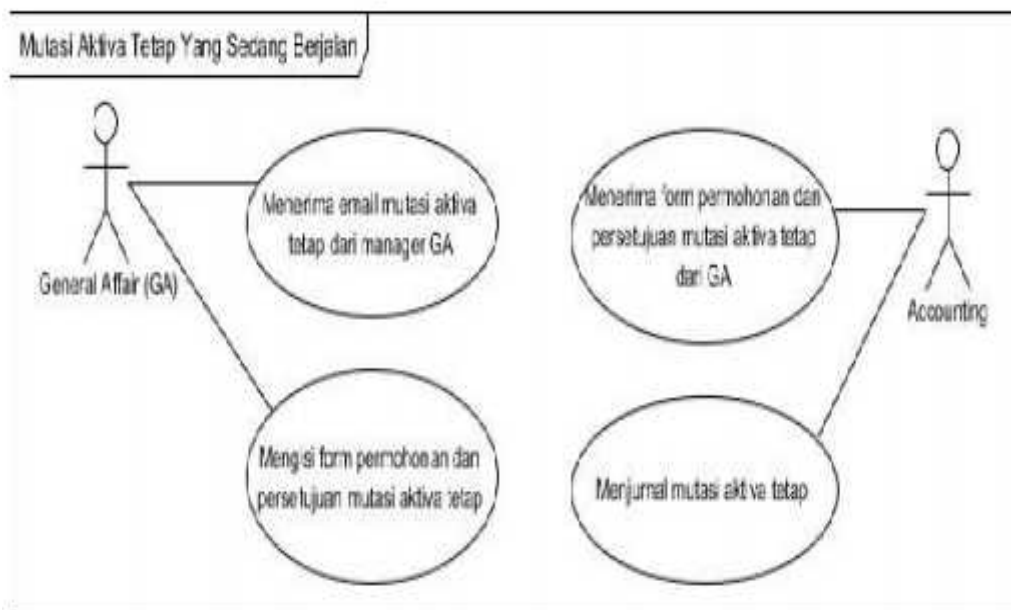
- a. *Use Case*, untuk mengetahui action atau prosedur apa yang ada didalam sistem.
- b. *Actor*, siapa saja yang terlibat dalam action tersebut.
- c. *Relationship*, bagaimana actions saling berelasi satu sama lain didalam sistem (Edgar Winata dan Johan Setiawan, 2013).

Adapun keterangan terhadap notasi / symbol dari gambar dibawah dapat dilihat pada tabel. II.2.

Tabel II.2. Simbol *Use Case Diagram*

Simbol	Keterangan
	Use Case: menggambarkan bagaimana seseorang akan menggunakan / memanfaatkan sistem
	Aktor: seseorang / sesuatu yang berinteraksi dengan sistem yang sedang kita kembangkan
	Relasi: sebagai penghubung antara aktor-usecase, usecase-usecase dll.
	Relasi Asosiasi: relasi terjadi antara aktor dengan usecase biasanya berupa garis lurus dengan kepala panah disalah satu ujungnya
	Include Relationship (relasi cakupan): memungkinkan suatu usecase untuk menggunakan fungsionalitas yang disediakan oleh usecase yang lainnya.
	Extend Relationship: memungkinkan usecase memiliki kemungkinan untuk memperluas fungsionalitas yang disediakan oleh usecase yang lainnya.

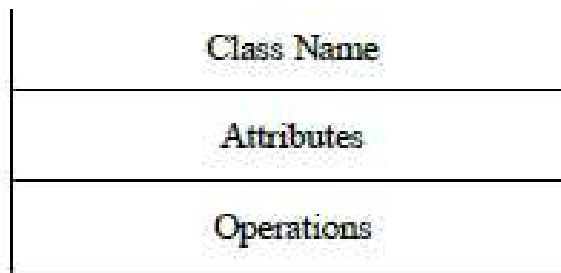
(Sumber : Oktafiansyah, 2012 : 5)



Gambar II.1. Use Case Diagram
(Sumber : Rosana Junita Sirait, dkk., 2015)

2. Class Diagram

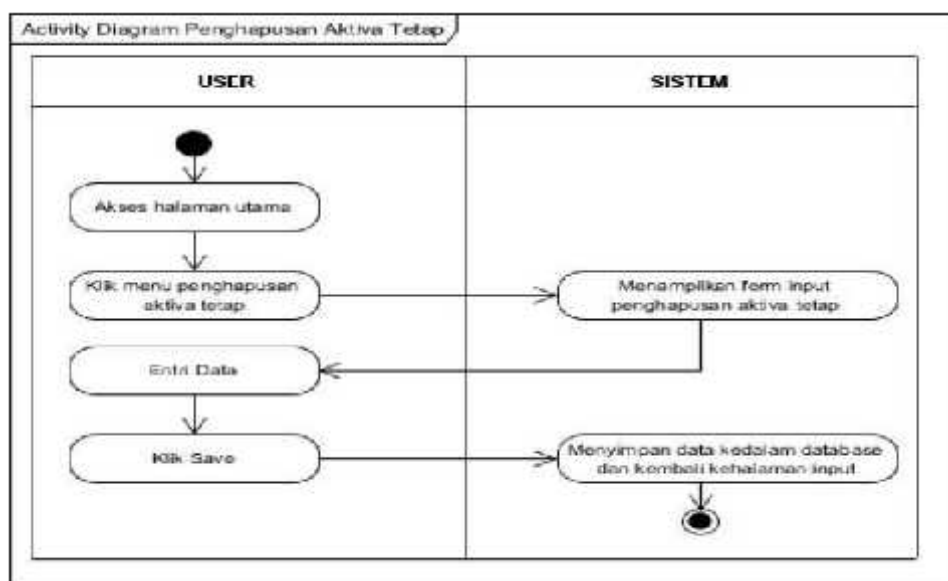
Menggambarakan struktur statis *class* di dalam sistem. *Class* merepresentasikan sesuatu yang ditangani oleh sistem. *Class* dapat berhubungan dengan yang lain melalui berbagai cara: *associated* (terhubung satu sama lain), *dependent* (satu *class* tergantung/menggunakan *class* yang lain), *specialized* (satu *class* merupakan spesialisasi dari *class* lainnya), atau *package* (grup bersama sebagai satu unit). Sebuah sistem biasanya mempunyai beberapa *class diagram*. Bentuk umum dari class diagram dapat dilihat pada gambar berikut :



Gambar II.2. Bentuk Umum *Class Diagram*
(Sumber : Sentosa Pohan, 2015 : 14)

3. *Activity Diagram*

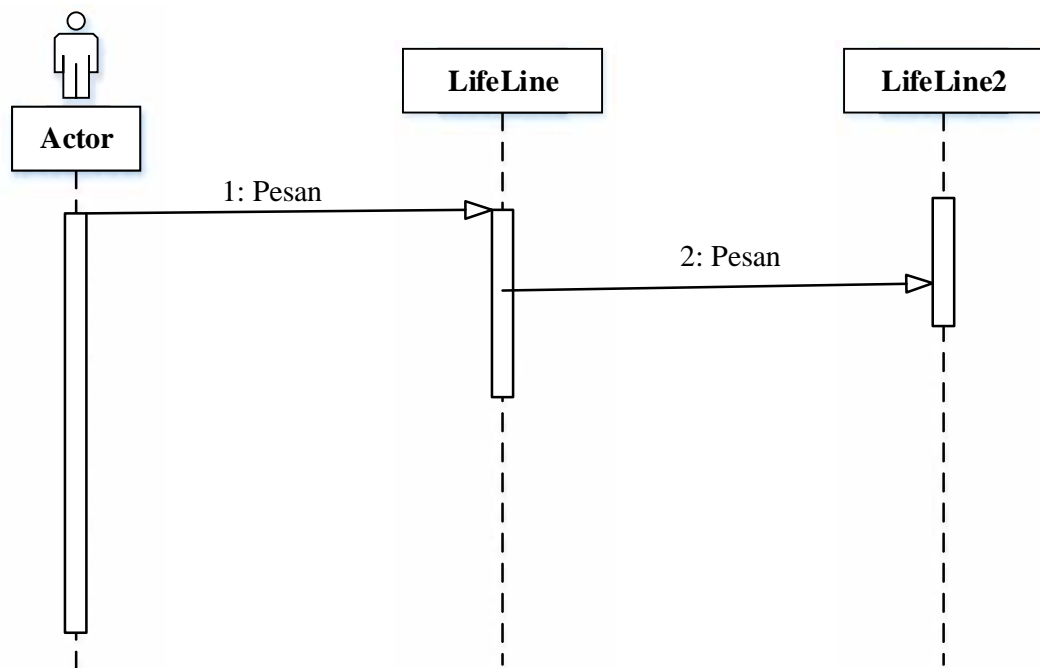
Menggambarkan rangkaian aliran dari aktivitas, digunakan untuk mendeskripsikan aktifitas yang dibentuk dalam suatu operasi sehingga dapat juga digunakan untuk aktifitas lainnya seperti *use case* atau interaksi.



Gambar II.4. *Activity Diagram*
(Sumber : Rosana Junita Sirait, dkk., 2015)

4. *Sequence diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).



Gambar II.3. Bentuk Umum *Sequence diagram*
(Sumber : Sentosa Pohan, 2015 : 14)