BAB II

LANDASAN TEORI

II.1. Sistem Pendukung Keputusan

Konsep Sistem Pendukung Keputusan pertama kali diperkenalkan pada awal tahun 1970-an oleh Michael S. Scott Morton dengan istilah *Management Decision System* (Sprague, 1982). Konsep pendukung keputusan ditandai dengan sistem interaktif berbasis komputer yang membantu pengambil keputusan memanfaatkan data dan model untuk menyelesaikan masalah-masalah yang tidak terstruktur. Pada dasarnya SPK dirancang untuk mendukung seluruh tahap pengambilan keputusan mulai dari mengidentifikasi masalah, memilih data yang relevan, menentukan pendekatan yang digunakan dalam proses pengambilan keputusan, sampai mengevaluasi pemilihan alternatif. (Hilya Magdalena, 2012 : 50).

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 mengindentifikasi 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).

II.1.1. Karakteristik dan Keterbatasan Sistem Pendukung Keputusan

SPK, menurut tinjaun 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 pasa 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.2. *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 menjad 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 berubahubah, 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 pembandingan antar sepasang objek, metode AHP memberikan sebuah standar nilai pembandingan antar dua objek seperti dituangkan pada tabel II.1.

Tabel II.1. Nilai Perbandingan

Pembanding	**Nilai**
Sangat diutamakan	9
Lebih diutamakan menuju sangat diutamakan	8
Lebih diutamakn	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.2.1. 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 x ((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.

   $CI = (\lambda maks-n)/n$ (1)

   Dimana n = banyaknya elemen
8. Memeriksa konsistensi hirarki. Jika nilainya lebih dari 10 persen maka penilaian data *judgment* harus diperbaiki.

   $CR = CI/IR$ (2)

   Dimana CR = Consistency Ratio

   CI = Consistency Index

IR = Index Random Consistency

II.3. *TOPSIS*

Menurut Tzeng dan Huang TOPSIS (*Technique for Order Preference by Similarity to Ideal Solution*) adalah salah satu metode pengambilan keputusan multikriteria yang pertama kali diperkenalkan oleh Yoon dan Hwang pada tahun 1981. TOPSIS memberikan sebuah solusi dari sejumlah alternatif yang mungkin dengan cara membandingkan setiap alternatif dengan alternatif terbaik dan alternatif terburuk yang ada di antara alternatif-alternatif masalah. Metode ini menggunakan jarak untuk melakukan perbandingan tersebut. Pada sistem ini, metode TOPSIS digunakan dalam meranking alternatif kost sebagai solusi alternatif kost terbaik untuk pengguna. Secara umum langkah-langkah yang harus dilakukan dalam menggunakan TOPSIS untuk memecahkan suatu masalah menurut Tzeng dan Huang adalah sebagai berikut (Herik Sugianto, dkk, 2016 : 2) :

1. Membangun Sebuah Matriks Keputusan

   Matriks keputusan X mengacu terhadap m alternatif yang akan dievaluasi berdasarkan n kriteria. Matriks keputusan X dapat dilihat pada Gambar 1 berikut :

$$X = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ . \\ . \\ . \\ a_m \end{array} \begin{array}{c} \begin{array}{cccccccc} x_1 & x_2 & x_3 & . & . & . & x_n \end{array} \\ \begin{bmatrix} x_{11} & x_{12} & x_{13} & . & . & . & x_{1n} \\ x_{21} & x_{22} & x_{23} & . & . & . & x_{2n} \\ x_{31} & x_{32} & x_{33} & . & . & . & x_{3n} \\ . & . & . & & & & . \\ . & . & . & & & & . \\ . & . & . & & & & . \\ x_{m1} & x_{m2} & x_{m3} & . & . & . & x_{mn} \end{bmatrix} \end{array}$$

Gambar II.1. Matriks Keputusan

Dimana a_i (i = 1, 2, 3, …, m) adalah alternatif-alternatif yang mungkin, xj(j = 1, 2, 3, …, n) adalah atribut dimana performansi alternatif diukur, dan xij adalah performansi alternatif a_i dengan acuan atribut xj.

2. Membuat Matriks Keputusan yang Ternormalisasi

   Persamaan yang digunakan untuk mentransformasikan setiap elemen xij adalah :

$$rij = \frac{xij}{\sqrt{\sum_{i=1}^{m} xij^2}} \qquad (3)$$

3. Membuat Matriks Keputusan yang Ternormalisasi Terbobot

   Dengan bobot wj = (w1, w2, w3, …, wn), dimana wj adalah bobot dari kriteria ke-j dan $\sum_{i=1}^{m} w_j = 1$, maka normalisasi bobot matriks V adalah : $v_{ij} = w_j r_{ij}$ (4)

   Dengan i = 1, 2, 3, … , m; dan j = 1, 2, 3, …, n; dimana vij adalah elemen dari matriks keputusan yang ternormalisasi terbobot V, wj adalah bobot dari

kriteria ke-j, dan rij adalah elemen dari matriks keputusan yang ternormalisasi R.

4. Menentukan Matriks Solusi Ideal Positif dan Solusi Ideal Negatif

Solusi ideal positif dinotasikan A^+, sedangkan solusi ideal negatif dinotasikan A^-. Berikut ini adalah persamaan dari A^+ dan A^- :

$A^+ = \{(\max vij \mid j \in J), (\min vij \mid j \in J'), i=1,2,3, \ldots ,m\}$ (5)

$= \{v_1^+, v_2^+, v_3^+, \ldots , v_n^+\}$

$A^- = \{(\max vij \mid j \in J), (\min vij \mid j \in J'), i=1,2,3, \ldots ,m\}$ (6)

$= \{v_1^-, v_2^-, v_3^-, \ldots , v_n^-\}$

J = { j = 1, 2, 3, … , n dan J merupakan himpunan kriteria keuntungan (*benefit criteria*)}. J' = { j = 1, 2, 3, …, n dan J' merupakan himpunan kriteria biaya (*cost criteria*)}. Dimana vij adalah elemen dari matriks keputusan yang ternormalisasi terbobot V,v_1^+ (j = 1, 2, 3, … , n) adalah elemen matriks solusi ideal positif, dan v_1^- (j = 1, 2, 3, … , n) adalah elemen matriks solusi ideal negatif.

5. Menghitung Separasi

S^+ adalah jarak alternatif dari solusi ideal positif yang didefenisikan sebagai :

$S_i^+ = \sqrt{\sum_{j=1}^{n}(vij - vj^-)^2}$, dengan i = 1,2,3,.....,m (7)

S^- adalah jarak alternatif dari solusi ideal negatif yang didefenisikan sebagai :

$S_i^- = \sqrt{\sum_{j=1}^{n}(vij - vj^-)^2}$, dengan i = 1,2,3,.....,m (8)

Dimana S_i^+ adalah jarak alternatif ke-i dari solusi ideal positif, S_i^- adalah jarak alternatif ke-i dari solusi ideal negatif, vij adalah elemen dari matriks

keputusan yang ternormalisasi terbobot V, v_j^+ adalah elemen matriks solusi ideal positif, dan v_j^- adalah elemen matriks solusi ideal negatif.

6. Menghitung Kedekatan Relatif Terhadap Solusi Ideal Positif

   Kedekatan relatif dari setiap alternatif terhadap solusi ideal positif dapat dihitung dengan menggunakan persamaan berikut:

   $$C_i^+ = \frac{s_i^-}{(s_i^- + s_i^+)}, 0 \leq C_i^+ \leq 1 \quad (9)$$

   Dengan i = 1, 2, 3, …, m, dimana C_i^+ adalah kedekatan relatif dari alternatif ke-i terhadap solusi ideal positif, s_i^+ adalah jarak alternatif ke-i dari solusi ideal positif, dan s_i^- adalah jarak alternatif ke-i dari solusi ideal negatif.

7. Merangking lternatif

   Alternatif diurutkan dari nilai C_i^+ terbesar ke nilai terkecil. Alternatif dengan nilai C_i^+ terbesar merupakan solusi yang terbaik.

II.4. Basis Data

Basis data sendiri dapat didefinisikan dalam sejumlah sudut pandang seperti :

1. Himpunan kelompok data (arsip) yang saling lberhubungan 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 elektronis. (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. Sisem 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 agregate,* 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.5. Normalisasi

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 (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 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.6. *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

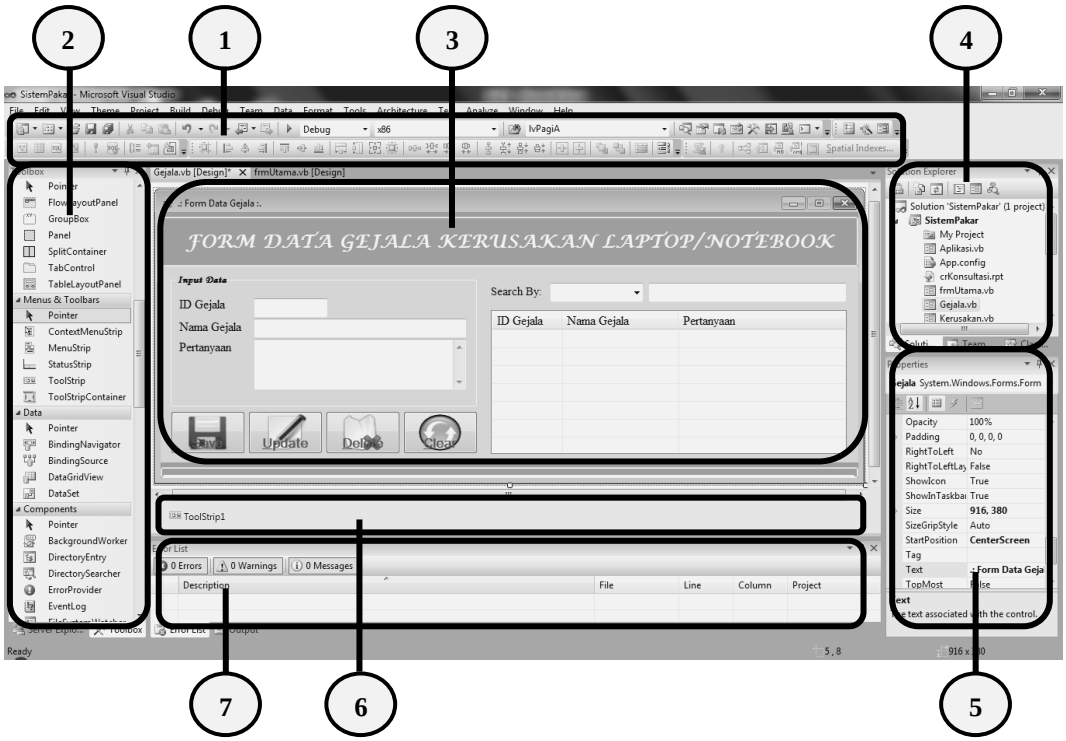
(SMBD), namun secara umum implemen bahasa ini memiliki bentuk standar yang ditetapkan oleh *ANSI*. (Adelia, Setiawan, 2011: 115).

II.7. *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. Intalasi 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.7.1. 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.2. *Integrated Developing Environment of Visual Basic 2010*
(Sumber: Dodit Suprianto, 2010 : 15)

Dari Gambar II.2. 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 *pulldown*. 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.8. *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.8.1. *Artifact* dan Notasi UML

1. Diagram Use Case

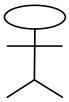
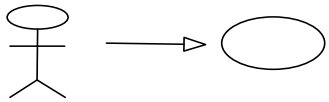
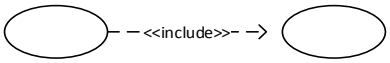
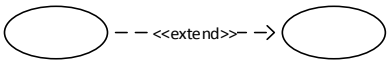
   Use case menjelaskan urutan kegiatan yang dilakukan actor dan sistem untuk mencapai suatu tujuan tertentu. Walaupun menjelaskan kegiatan, namun use case hanya menjelaskan apa yang dilakukan oleh actor dan sistem bukan bagaimana actor dan sistem melakukan kegiatan tersebut.

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

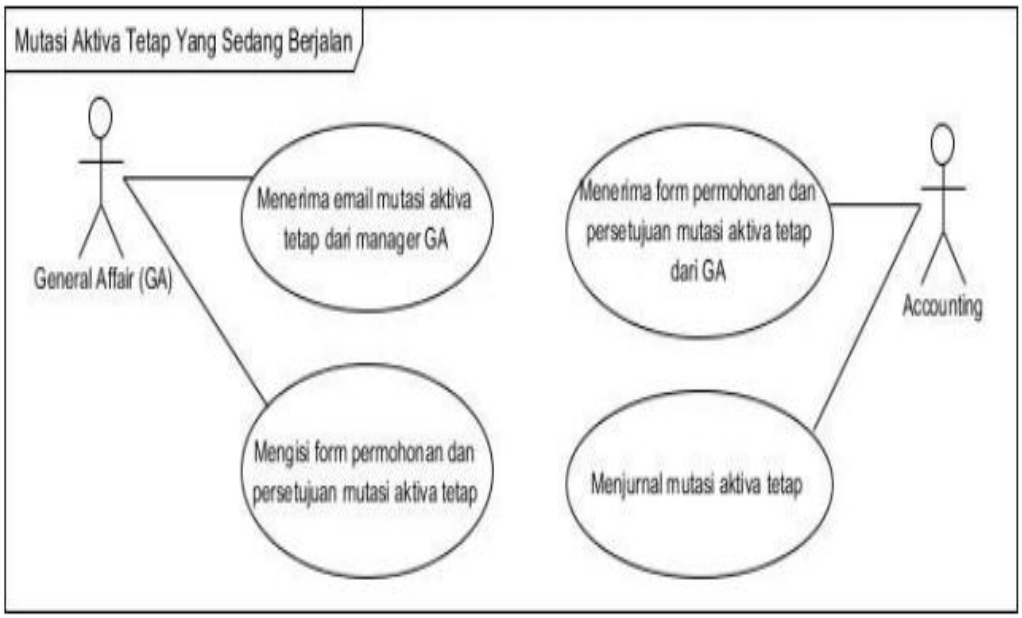
   a. *Use Case*, untuk mengetahui action atau prosedur apa yang ada didalam sistem.
   b. *Actor*, siapa saja yang terlibat dalam action tersebut.
   c. *Relationship*, bagaimana actions saling berelasi satu sama lain didalam sistem (Edgar Winata dan Johan Setiawan, 2013).

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 sisitem yang sedang kita kembangkan**
	Relasi: sebagai penghubung antara aktorusecase, usecase-usecase dll.
	**Relasi Asosiasi: relasi terjadi antara aktor dengan usecase biasanya berupa garis lurus dengan kepala panah dis alah satu ujung nya**
	**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.**

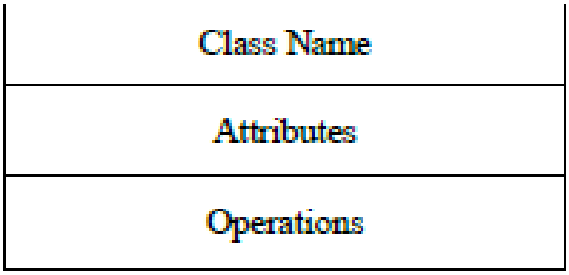
(Sumber : Oktafiansyah, 2012 : 5)



Gambar II.3. *Use Case* Diagram
(Sumber : Rosana Junita Sirait, dkk., 2015)

2. *Class Diagram*

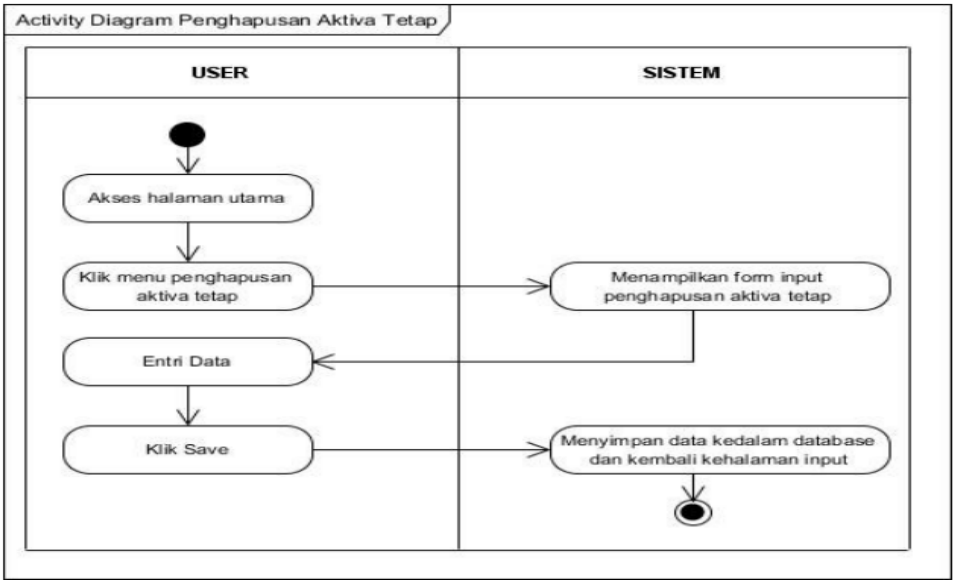
   Menggambarkan struktur statis *class* di dalam sistem. *Class* merepresentasikan sesuatu yang ditangani oleh sistem. *Class* dapat berhubungan dengan yang lain melalui berbagai cara: *associated* (terhubung satu sama lain), *dependent* (satu *class* tergantung/menggunakan *class* yang lain), *specialed* (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.4. 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.

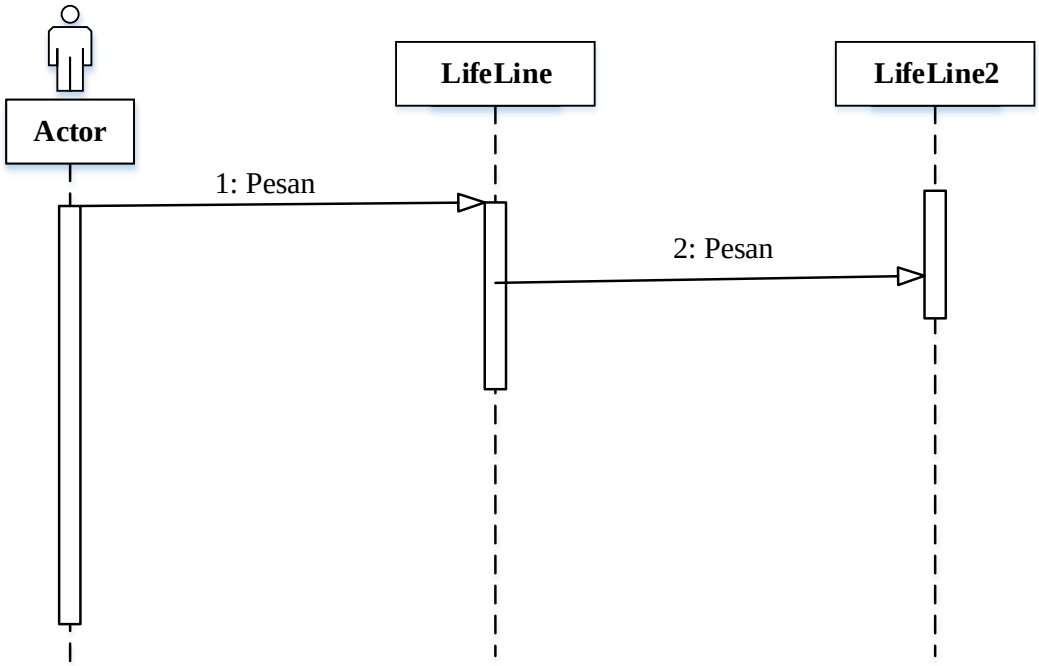


Gambar II.5. *Activity* Diagram
(Sumber : Rosana Junita Sirait, dkk., 2015)

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 atar dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait).



Gambar II.6. Bentuk Umum *Sequence* diagram
(Sumber : Sentosa Pohan, 2015 : 14)