BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Secara leksikal, sistem berarti susunan yang teratur dari pandangan, teori, asas dan sebagainya. Dengan kata lain, sistem adalah suatu kesatuan usaha yang terdiri dari bagian-bagian yang berkaitan satu sama lain yang berusaha mencapai suatu tujuan dalam suatu lingkungan kompleks. Pengertian tersebut mencerminkan adanya beberapa bagian dan hubungan antara bagian, ini menunjukkan kompleksitas dari sitem yang meliputi kerja sama antara bagian yang interpenden satu sama lain. Selain itu dapat dilihat bahwa sistem berusaha mencapai tujuan. Pencapaian tujuan ini menyebabkan timbulnya dinamika, perubahan-perubahan yang terus menerus perlu dikembangkan dan dikendalikan. Definisi tersebut menunjukkan bahwa sistem sebagai gugus dan elemen-elemen yang saling berinteraksi secara teratur dalam rangka mencapai tujuan atau subtujuan (Marimin ; 2008 : 1-2).

II.2. Sistem Pakar

Sistem pakar menirukan perilaku seorang pakar dalam menangani suatu persoalan. Pada suatu kasus seorang pasien mendatangi dokter untuk memeriksa badannya yang mengalami gangguan kesehatan, maka dokter atau pakar kesehatan akan memeriksa dan melakukan diagnosa. Bila dokter cukup sibuk dan pelaksana diagnosa digantikan oleh sebuah sistem pakar, maka sistem pakar

diharapkan dapat membantu memahami dan menganalisa keadaan pasien dan menemukan penyakit yang diderita pasien itu. Sistem pakar diharapkan juga untuk menghasilkan dugaan atau hasil diagnosa yang sama dengan diagnosa yang dilakukan oleh seorang ahli. Tujuan utama sistem pakar bukan untuk menggantikan kedudukan seorang ahli maupun pakar, tetapi untuk memasyarakatkan pengetahuan dan pengalaman pakar-pakar yang ahli di bidangnya.

Tujuan utama sistem pakar bukan untuk menggantikan kedudukan seorang ahli maupun pakar, tetapi untuk memasyarakatkan pengetahuan dan pengalaman pakar-pakar yang ahli di bidangnya. Sistem pakar disusun oleh dua bagian utama, yaitu :

1. Lingkungan pengembangan (*development environment*), digunakan untuk memasukkan pengetahuan pakar ke dalam lingkungan sistem pakar.
2. Lingkungan konsultasi (*consultation environment*), digunakan oleh pengguna yang bukan pakar guna memperoleh pengetahuan pakar.

Sistem pakar memiliki 2 komponen utama yaitu basis pengetahuan dan mesin inferensi. Basis pengetahuan merupakan tempat penyimpanan pengetahuan dalam memori komputer, dimana pengetahuan ini diambil dari pengetahuan pakar. komponen-komponen sistem pakar adalah seperti di bawah ini :

1. Antarmuka (*User Interface*)

   User Interface merupakan mekanisme yang digunakan oleh pengguna dan sistem pakar untuk berkomunikasi. Antarmuka menerima informasi dari pemakai dan mengubahnya kedalam bentuk yang dapat diterima oleh sistem.

2. Basis Pengetahuan

   Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan.

3. Akuisisi Pengetahuan (*Knowledge Acquisition*)

   Akuisisi pengetahuan adalah akumulasi, transfer dan transformasi keahlian dalam menyelesaikan masalah dari sumber pengetahuan kedalam program komputer.

4. Mesin Inferensi

   Komponen ini mengandung mekanisme pola pikir dan penalaran yang digunakan oleh pakar dalam menyelesaikan suatu masalah.

5. *Workplace*

   Workplace merupakan area dari sekumpulan memori kerja (*working memory*). *Workplace* digunakan untuk merekam hasil-hasil antara dan kesimpulan yang dicapai.

6. Fasilitas Penjelasan

   Fasilitas penjelasan adalah komponen tambahan yang akan meningkatkan kemampuan sistem pakar.

7. Perbaikan Pengetahuan

   Pakar memiliki kemampuan untuk menganalisis dan meningkatkan kinerjanya serta kemampuan untuk belajar dari kinerjanya.

Ada banyak keuntungan bila menggunakan sistem pakar, diantaranya adalah :

1. Menjadikan pengetahuan dan nasehat mudah didapat.
2. Meningkatkan *output* dan produktivitas.
3. Menyimpan kemampuan dan keahlian pakar.
4. Meningkatkan penyelesaian masalah, menerusi paduan pakar, penerangan, sistem pakar khas.
5. Meningkatkan reliabilitas.
6. Memberikan *respons* (jawaban) yang cepat.
7. Merupakan penduan yang *inteligence* (cerdas).
8. Dapat bekerja dengan informasi yang lengkap dan mengandung ketidakpastian.
9. *Intelligence database* (basis data cerdas), bahwa sistem pakar dapat digunakan untuk mengakses basis data dengan cara cerdas.

Selain keuntungan-keuntungan di atas, sistem pakar seperti sistem lainnya juga memiliki kelemahan, diantaranya adalah:

1. Masalah dalam mendapatkan pengetahuan dimana pengetahuan tidak selalu bisa didapatkan dengan mudah, kadangkala pakar dari masalah yang kita buat tidak ada, dan kalaupun ada kadang-kadang pendekatan yang dimiliki pakar berbeda-beda.
2. Untuk membuat suatu sistem pakar yang benar-benar berkualitas tinggi sangatlah sulit dan memerlukan biaya yang sangat besar untuk pengembangan dan pemeliharaannya.
3. Boleh jadi sistem tak dapat membuat keputusan.

4. Sistem pakar tidaklah 100% menguntungkan, walaupun seorang tetap tidak sempurna atau tidak selalu benar. Oleh karena itu perlu diuji ulang secara teliti sebelum digunakan. Dalam hal ini peran manusia tetap merupakan faktor dominan.

Sistem pakar merupakan program-program praktis yang menggunakan strategi *heuristic* yang dikembangkan oleh manusia untuk menyelesaikan permasalahan-permasalahan yang *spesifik* (khusus). Disebabkan oleh *keheuristikannya* dan sifatnya yang berdasarkan pada pengetahuan, maka umumnya sistem pakar bersifat :

1. Memiliki informasi yang handal, baik dalam menampilkan langkah-langkah antara maupun dalam menjawab pertanyaan-pertanyaan tentang proses penyelesaian.
2. Mudah dimodifikasi, yaitu dengan menambah atau menghapus suatu kemampuan dari basis pengetahuan.
3. Heuristik dalam menggunakan pengetahuan (yang sering kali tidak sempurna) untuk mendapatkan penyelesaiannya.
4. Dapat digunakan dalam berbagai jenis computer.
5. Memiliki kemampuan untuk belajar beradaptasi**.**

Basis pengetahuan (*knowladge base)* sebagian sistem pakar yang berisikan fakta-fakta yang menggambarkan wilayah masalah dan teknik-teknik refresentasi pengetahuan yang menggambarkan bagaimana fakta-fakta saling bersesuaian secara logis.

Basis pengetahuan mengandung pengetahuan untuk pemahaman, formulasi, dan penyelesaian masalah. Komponen sistem pakar ini disusun atas dua elemen dasar, yaitu fakta dan aturan (*rule*). Fakta merupakan informasi tentang obyek dalam area permasalahan tertentu, sedangkan aturan merupakan informasi tentang cara bagaimana memperoleh fakta baru dari fakta yang telah diketahui.

Suatu perkalian inferensi yang menghubungkan suatu permasalahan dengan solusinya disebut rantai (*chain*). ada dua metode penalaran dengan *rules*, yaitu *forward chaining* atau *data-driven* dan *backward chaining* atau *goaldriven*.

1. *Forward Chaining*

   Suatu rantai yang dicari atau dilewati/dilintasi dari suatu permasalahan untuk memperoleh solusinya dengan penalaran dari fakta menuju konklusi yang terdapat dari fakta.

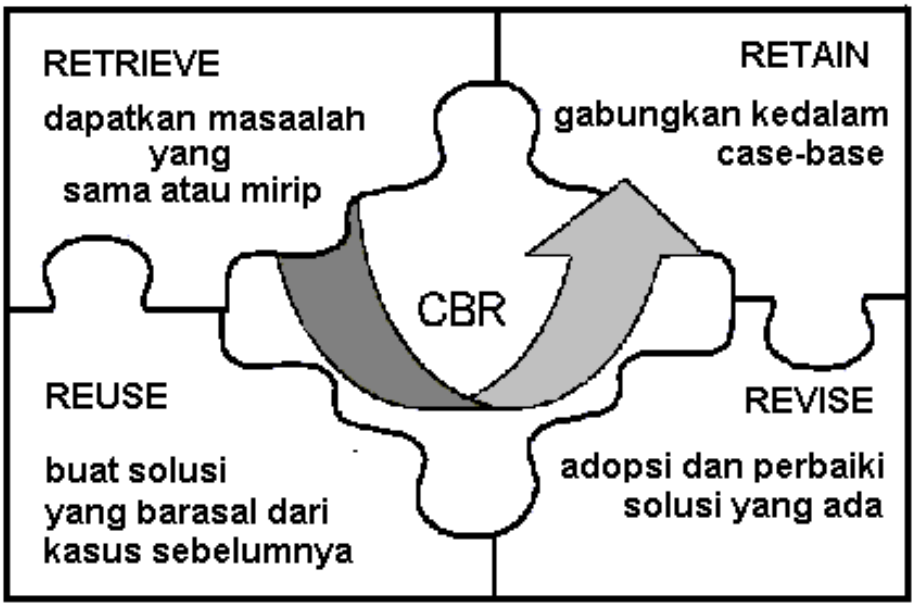
2. *Backward Chaining*

   a. Suatu rantai yang dilintasi dari suatu hipotesa kembali ke fakta yang mendukung hipotesa tersebut, dan dalam hal tujuan yang dapat dipenuhi dengan pemenuhan sub tujuannya. (Jurnal Teknologi dan Informatika : Andri Saputra ; 2011 : 202-206).

II.3. Metode *Case Based Reasoning* (CBR)

Case Based Reasoning (CBR) merupakan salah satu metode pemecahan masalah yang dalam mencari solusi dari suatu kasus yang baru, sistem akan melakukan pencarian terhadap solusi dari kasus lama yang memiliki permasalahan yang sama dan sudah pernah terjadi sebelumnya. Metode CBR dikembangkan

oleh Roger Schank dan rekannya di Universitas Yale pada awal tahun 1980. Terdapat dua prinsip dasar pada metode CBR, prinsip pertama adalah setiap permasalahan yang sama akan memiliki solusi yang sama pula. Oleh karena itu, solusi dari permasalahan yang sudah pernah terjadi dapat digunakan kembali untuk memecahkan masalah baru dengan permasalahan yang sama dengan masalah yang lama. Prinsip kedua adalah setiap permasalahan dapat terjadi berulang kali. Oleh karena itu, terdapat kemungkinan bahwa masalah yang akan muncul di masa yang akan datang memiliki kesamaan dengan masalah yang pernah terjadi sebelumnya.



Gambar II.1. Peroses CBR
Sumber : Syafiul Muzid; 2008 : 62

Terdapat empat proses yang terjadi pada metode CBR dalam menyelesaikan masalah yaitu:

1. *Retrivie*

Pada proses ini sistem akan melakukan identifikasi parameter pencocokan yang dapat dijadikan sebagai acuan lalu melakukan pencarian kasus lama yang memiliki kesamaan dengan kasus baru. Selanjutnya sistem akan melakukan pencocokan parameter antara kasus baru dengan kasus lama dan memilih kasus yang memiliki tingkat kecocokan tertinggi.

2. *Reuse*

Pada proses ini sistem akan menggunakan kembali informasi yang berasal dari kasus sebelumnya atau sistem akan melakukan adaptasi terlebih dahulu untuk memecahkan masalah pada kasus yang baru.

3. *Revise*

Pada proses ini sistem akan menggunakan kembali informasi yang berasal dari kasus sebelumnya atau sistem akan melakukan adaptasi terlebih dahulu untuk memecahkan masalah pada kasus yang baru.

4. *Retain*

Pada proses ini apabila ternyata ditemukan solusi baru yang lebih baik dari solusi yang telah ada sebelumnya maka solusi baru tersebut akan diberi indeks dan disimpan untuk kemudian digunakan kembali pada kasus serupa pada masa yang akan datang.

Adapun penerapan CBR banyak dilakukan untuk memodelkan domain permasalahan yaitu:

1. Sulit dalam melakukan elisitasi pengetahuan.

2. Sulit menerapkan aturan if-then, karena sangat bergantung pada tingkat kepakaran yang spesifik dan berkembang dalam waktu lama.
3. Ada kecenderungan sulit dalam pengelolaan data karena perkembangannya yang pesat.
4. Ada kecenderungan sulit dalam pengelolaan data karena perkembangannya yang pesat (Jurnal Informatika, Vol.4,No.2 : Hapnes Toba ; 2008 : 135-137).

Case-based Reasoning melakukan proses mengingat penyelesaian masalah sebelumnya. Kemudian ketika ada permasalahan baru, *Case-based Reasoning* melakukan perbandingan antara karakteristik permasalahan baru dengan permasalahan yang pernah diselesaikan sebelumnya, ketika permasalahan terbaru mirip dengan permasalahan sebelumnya, CBR melakukan proses ekstraksi solusi dari permasalahan yang relevan dengan permasalahan baru yang dihadapi, apabila solusi tersebut sesuai maka solusi tersebut dipergunakan untuk memecahkan permasalahan baru. Setelah itu, dilanjutkan dengan proses adaptasi, yakni memperbaiki pengetahuan lama agar sesuai untuk menyelesaikan permasalahan baru. Setelah melalui proses adaptasi, pengetahuan baru akan disimpan sebagai salah satu *case base*.

$$Similarity(f_j^I, f_{ij}^R) = \frac{\sum_{j=1}^{n} [w_j * s(f_j^I, f_{ij}^R)]}{\sum_{j=1}^{n} w_j}$$

(1)

J = atribut ke j
n = atribut ke n
Wj = bobot atribut ke j.
f_j^I = nilai dari atribut j pada *case* inputan I.
f_{ij}^R = atribut j pada *case* i yang tersimpan dalam *case memory* (R).
$S(f_j^I, f_{ij}^R)$ = hasil perbandingan f_j^I dengan f_{ij}^R, hasil $\in [0,1]$.

Pembobotan SWING merupakan metode yang memiliki kemampuan menggabungkan rentang atribut dari masing- masing input dari bobot dimana semua atribut didasarkan pada nilai terburuk dan nilai terbaik. Penentuan nilai dilakukan melalui penilaian yang dilakukan oleh para ahli yang diberikan dengan cara memberikan nilai maksimal pada atribut paling penting, lalu memberikan nilai dibawah nilai maksimal secara berurutan untuk atribut berikutnya. Nilai yang diberikan kepada seluruh atribut dinormalisasikan untuk menjadi nilai bobot yang memiliki rentang antara 0 hingga 1 [7]. Persamaan yang dipergunakan untuk menormalisasi nilai untuk dijadikan bobot dapat dilihat pada persamaan 2:

$$w_i = \frac{r_i}{\sum_{k=1}^{m} r_i}, i = 1, 2, \ldots, m,$$

r_i = nilai bagi atribut ke *i*
w_i=bobot ke *i*

Proses adaptasi *case* merupakan proses penyesuaian dari solusi yang terambil dimana solusi tersebut telah memiliki kemiripan paling tinggi.

Penyesuaian dilakukan agar solusi yang terambil tersebut sesuai dengan permasalahan yang sedang dialami dan membuat pengetahuan sistem cerdas menjadi berkembang. Langkah yang dapat diambil dalam proses adaptasi sebagai berikut :

1. Solusi yang dihasilkan dari *case* yang diambil dapat dipergunakan sebagai solusi dari permasalahan yang sedang dialami baik tanpa ataupun dengan melalui modifikasi.
2. Ketika ternyata dari proses *retrieval* mendapatkan lebih dari satu kasus yang paling relevan, solusi bisa didapatkan dari salah satu kasus diantara kasus yang paling relevan.

Cara Kerja CBR sama dengan proses penalaran masalah pada otak manusia. Ketika suatu masalah atau *case* yang ditangani ternyata gagal, maka kegagalan akan disimpan kedalam *case memory* sehingga kesalahan yang sama tidak terulang. Dan ketika *case* baru yang ditangani ternyata sukses CBR menyimpannya kedalam *case memory* untuk memecahkan permasalahan yang sama di kemudian hari. Untuk mengetahui *case* yang sukses atau gagal perlu dilakukan penilaian dalam dunia nyata untuk memastikan solusi dari CBR benar-benar sukses atau gagal.

Pemeliharaan dengan mengurangi duplikasi *case* dan menghapus atau memperbaiki *case* yang salah merupakan hal yang penting untuk mencegah level error terlalu tinggi. Pemeliharaan *case-based* dapat memberikan pengaruh positif pada kualitas *case* tersimpan untuk memberikan solusi pada permasalahan yang akan datang. Karena semakin banyaknya jumlah *case* yang tersimpan, performa

pencarian *case* juga semakin menurun (Jurnal Teknik ITS, Vol.1,No.1 : Irlando Moggi Prakoso; 2012 : A332-A353).

II.3.1. Studi Kasus dengan Metode Case-Based Reasoning

Setiap kasus yang disimpan pada basis kasus diformat seperti dibawah ini:

Tabel 1. Faktor bagian pada setiap kasus

Faktor bagian pada setiap kasus
Usia Ibu Hamil
Usia Kandungan
Gejala-gejala penyakit
Penyakit dan Solusi

Penjelasan:

Setiap kasus yang disimpan memiliki format empat bagian yang digunakan dalam memudahkan penyimpanan data kasus. Tetapi dari keempat faktor hanya tiga faktor yang digunakan dalam pencarian kasus yang mirip, sedangkan faktor penyakit dan solusi tidak diikutsertakan.

- Usia Ibu Hamil atau faktor A1, adalah data usia dari ibu yang sedang hamil. Pada bagian ini dibagi menjadi beberapa kategori, yaitu:

Tabel 2. Faktor A1 atau usia ibu hamil

Kode Usia Ibu	Usia Ibu
1	10 – 20 tahun
2	20 – 30 tahun

Kode Usia Ibu	Usia Ibu
3	30 – 40 tahun
4	40 – 50 tahun
5	> 50 tahun

- Usia Kandungan atau faktor A2, adalah data- data yang berisi usia kandungan dari ibu hamil tersebut. Bagian ini terdiri dari beberapa kategori, yaitu:

Tabel 3. Faktor A2 atau usia kandungan

KodeUsia	Usia
1	< 1 bulan
2	1 – 3 bulan
3	3 – 6 bulan
4	6 – 9 bulan
5	> 9 bulan

- Gejala-gejala penyakit atau faktor A3, bagian ini berisi gejala-gejala yang menyebabkan suatu penyakit pada kehamilan. Adapun dibawah ini hanya sebagai contoh gejala.

Tabel 4. Faktor A3 atau gejala penyakit

Kode Gejala	Gejala
G1	Gejala 1
G2	Gejala 2
G3	Gejala 3
G4	Gejala 4
G5	Gejala 5
G6	Gejala 6
G7	Gejala 7
G8	Gejala 8
G9	Gejala 9
G10	Gejala 10
G11	Gejala 11
G12	Gejala 12
G13	Gejala 13
G14	Gejala 14
G15	Gejala 15

- Penyakit dan solusi atau terapi suatu penyakit, dapat dilihat pada contoh data kasus yang telah disimpan.

II.3.1.1. Pengukuran kemiripan kasus (*similarity*)

Dalam mencari kasus yang memiliki kemiripan dengan kasus baru, setiap kasus baru akan disamakan dengan semua kasus yang ada pada basis kasus dengan faktor-faktor bagian diatas, namun hanya tiga faktor yang digunakan untuk mengukur kemiripan, yaitu usia ibu hamil, usia kandungan, serta gejala-gejala. Sedangkan faktor bagian penyakit dan solusi tidak diikutkan dalam pengukuran.

Misalnya ada kasus baru yang berisi data usia ibu hamil sekitar 20-30 tahun, usia kandungan anatar 1-3 bulan, dan gejala yang dialami yaitu G1, G3, G11. Maka untuk kasus baru ini akan dihitung kemiripannya dengan kasus – kasus yang ada dengan tiga faktor pengukur yaitu A1, A2, dan A3 dengan rumus sebagai berikut:

$$Stotal = \frac{A1 + A2 + A3}{Ntotal}$$

Penjelasan:

- A1 adalah faktor A1 yaitu faktor usia ibu hamil.
- A2 adalah faktor A2 yaitu usia kandungan.
- A3 adalah faktor A3 yaitu gejala-gejala.
- Ntotal adalah jumlah masukan, misalnya:
  - A1 diisi kode 3 yaitu usia ibu hamil antara 30-40 tahun maka N dihitung 1 masukan.
  - A2 diisi kode 2 yaitu usia kandungan antara 1-3 bulan maka N dihitung 1 masukan.
  - A3 diisi dengan kode gejala G1, G3, G11 maka N dihitung sebanyak 3

masukan.

- Sehingga Ntotal pada kasus baru diatas adalah 5.

- Stotal adalah jumlah nilai *similarity.*

Dari kasus baru diatas maka akan dihitung berdasarkan tabel 2, tabel 3, dan Tabel 4. Adapun perhitungannya sebagai berikut:

$$Stotal = \frac{3 + 2 + (G1 + G3 + G11)}{5}$$

Setelah dimasukan nilainya maka kasus baru tersebut akan dibandingkan dengan setiap kasus yang ada pada contoh yaitu Tabel 5. Hasil perhitunganya untuk kemiripan (*similarity*) setiap kasus yang tersimpan pada basis kasus dengan kasus baru adalah sebagai berikut :

Tabel 6. Jumlah nilai kemiripan dengan kasus baru.

Basis Kasus	**Nilai A1**	**Nilai A2**	**Nilai A3**	**Stotal**
K1	0	1	1	2/5
K2	1	0	0	1/5
K3	1	0	0	1/5
K4	***0***	***1***	***3***	***4/5***
K5	0	0	0	0/5
K6	0	1	0	1/5
K7	1	0	1	2/5
K8	1	0	0	1/5
K9	0	1	0	1/5
K10	0	0	0	0/5

Dari hasil perhitungan pada tabel 6, didapatkan satu kasus lama yang memiliki tingkat kemiripan paling tinggi dengan kasus yang baru daripada kasus-kasus lainnya, yaitu kasus K4 dengan nilai kemiripan sebesar 4/5 atau 80% kemiripan.

Tabel 5. Basis kasus yang tersimpan beserta data penyakit dan solusi atau terapinya.

Kode Kasus	Kode Usia Ibu Hamil (A1)	Kode Usia Kandungan (A2)	Gejala Penyebab (A3)	Penyakit	Solusi/terapi
K1	2	2	G1,G2, G5,G9	Penyakit A	Diberikan obat untuk penyakit A
K2	3	1	G6,G9, G12	Penyakit B	Diberikan obat untuk penyakit B
K3	3	3	G10, 13, 15	Penyakit C	Diberikan obat untuk penyakit C
K4	2	2	G1,G3,G5,G11,G12,G13	Penyakit A	Diberikan obat untuk penyakit A
K5	4	4	G5, G6, G9	Penyakit B	Diberikan obat untuk penyakit B
K6	2	2	G9, G14	Penyakit D	Diberikan obat untuk penyakit D
K7	3	3	G1, G2, G5	Penyakit A	Diberikan obat untuk penyakit A
K8	3	3	G4, G7, G13	Penyakit E	Diberikan obat untuk penyakit E
K9	5	2	G4, G7, G8,G13	Penyakit E	Diberikan obat untuk penyakit E
K10	4	4	G8, G9, G10	Penyakit F	Diberikan obat untuk penyakit F

II.3.1.2. Pengambilan atau pemilihan data

Kriteria untuk pemilihan kasus adalah kasus yang memiliki kemiripan paling tinggi dengan kasus baru yang akan disarankan sebagai solusi. Walaupun demikian, setiap kasus baru belum tentu memiliki nilai kemiripan yang lumayan tinggi dengan basis kasus. Maka perlu diberikan kriteria kemiripan dengan menghitung nilai desimal dari setiap Stotal atau nilai kemiripan. Adapun kriteria pembagian nilai Stotal adalah sebagai berikut:

Tabel 7. Kriteria kemiripan

Nilai Desimal Kemiripan	Kriteria Kemiripan
0,8 – 1	High
0,4 – 0,79	Medium
0 – 0,39	Low

Berdasarkan tabel kriteria kemiripan maka setiap kasus pada basis kasus memiliki kriteria kemiripan dengan kasus baru sebagai berikut:

Tabel 8. Hasil kriteria kemiripan setiap kasus dengan kasus baru

Basis Kasus	Stotal	Nilai Desimal Kemiripan	Kriteria Kemiripan
K1	2/5	0,4	Medium
K2	1/5	0,2	Low
K3	1/5	0,2	Low
K4	**4/5**	**0,8**	**High**
K5	0/5	0	Low
K6	1/5	0,2	Low
K7	2/5	0,4	Medium
K8	1/5	0,2	Low
K9	1/5	0,2	Low
K10	0/5	0	Low

Oleh karena itu kasus K4 akan dipilih menjadi solusi yang disarankan untuk kasus baru tersebut. Karena memiliki kriteria kemiripan HIGH seperti pada tabel diatas. Dengan kata lain, kasus baru tersebut kemungkinan adalah penyakit A dan disarankan diberi obat untuk penyakit A, seperti pada basis kasusnya (Syafiul Muzid; 2008 : 62-64).

II.4. Pengertian Macromedia Dreamweaver

Macromedia Dreamweaver adalah sebuah *software web design software web design* yang menawarkan cara mendesain *website* dengan dua langkah sekaligus dalam satu waktu, yaitu mendesain dan meprogram. Dreamweaver memiliki satu jendela mini yang disebut HTML *Source*, tempat kode-kode HTML tertulis. Setiap kali kita mendesain *web*, seperti menulis kata-kata, meletakkan gambar, membuat tabel dan proses lainnya, tag-tag HTML akan tertulis secara langsung mengiringi proses pengaturan *website*. Artinya kita memiliki kesempatan untuk mendesain. *Website* sekaligus mengenal tag-tag HTML yang membangun *website* itu. Di lain kesempatan kita juga dapat mendesain *website*

hanya dengan menulis tag-tag dan teks lain di jendela HTML *Source* dan hasilnya dapat dilihat langsung di layar (M. Suyanto ; 2009 : 244).

II.5. Pengertian PHP

PHP merupakan suatu bahasa pemrograman sisi server yang dapat anda gunakan untuk membuat halaman Web dinamis. Contoh bahasa yang lain adalah *Microsoft Active Server Page (*ASP) dan *Java Server Page* (JSP). Dalam suatu halaman HTML anda dapat menanamkan kode PHP yang akan dieksekusi setiap kali halaman tersebut dikunjungi. Karena kekayaannya akan fitur yang mempermudah perancangan dan pemrograman Web, PHP memiliki pepularitas yang tinggi. Anda dapat mengecek *survey* popularitas yang dilakukan *netcraft* di URL *www.php.net/usage.php.*

PHP adalah kependekan dari *HyperText Prepocessor* (suatu akronim rekursif) yang dibangun oleh Rasmus Lerdorf pada tahun 1994. Dahulu, pada awal pengembangnnya PHP disebut sebagai kependekan dari *Personal Home Page.* PHP merupakan produk *Open Source* sehingga anda dapat mengakses *source code*, menggunakan dan mengubahnya tanpa harus membayar sepeser pun. Gratis (Antonius Nugraha Widhi Pratama ; 2010 : 9).

II.6. Pengertian *Database*

Secara sederhana *database* (basis data/pangkalan data) dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan cepat. Pengertian akses dapat mencakup pemerolehan data maupun pemanipulasian data seperti

menambah serta menghapus data. Dengan memanfaatkan komputer, data dapat disimpan dalam media pengingat yang disebut *harddisk.* Dengan menggunakan media ini, keperluan kertas untuk menyimpan data dapat dikurangi. Selain itu, data menjadi lebih cepat untuk diakses terutama jika dikemas dalam bentuk *database* (Agustinus Mujilan ; 2012 : 23-24).

II.7. MySQL

MySQL adalah suatu sistem manajemen basis data relasional (RDBMS-*Relational Database Management System*) yang mampu bekerja dengan cepat, kokoh, dan mudah digunakan. Contoh RDBMS lain adalah *Oracle, Sybase*. Basis data memungkinkan anda untuk menyimpan, menelusuri, menurutkan dan mengambil data secara efesien. *Server* MySQL yang akan membantu melakukan fungsionaliitas tersebut. Bahasa yang digunakan oleh MySQL tentu saja adalah SQL-*standar* bahasa basis data relasional di seluruh dunia saat ini.

MySQL dikembangkan, dipasarkan dan disokong oleh sebuah perusahaan Swedia bernama MySQL AB. RDBMS ini berada di bawah bendera GNU GPL sehingga termasuk produk *Open Source* dan sekaligus memiliki lisensi komersial. Apabila menggunakan MySQL sebagai basis data dalam suatu situs Web. Anda tidak perlu membayar, akan tetapi jika ingin membuat produk RDBMS baru dengan basis MySQL dan kemudian mengualnua, anda wajib bertemu mudah dengan lisensi komersial (Antonius Nugraha Widhi Pratama ; 2010 : 10).

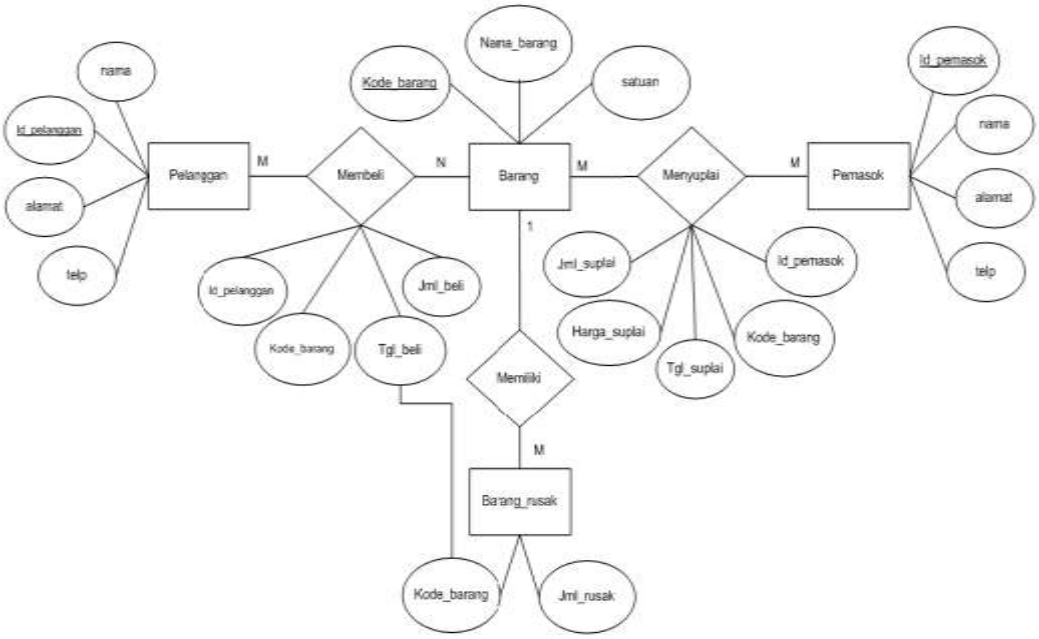
II.8. Entity Relationship Diagram (ERD)

Entity Relationship Diagram atau ERD merupakan salah satu alat (tool) berbentuk grafis yang opopuler untuk *desain database.* Tool ini relatif lebih mudah dibandingkan dengan Normalisasi. Kebanyakan sistem analis memakai alat ini, tetapi yang jadi masalah, kalau kita cermati secara seksama, tool ini mencapai 2NF (Yuniar Supardi ; 2010 : 448).

Notasi	Keterangan
	**Entitas**, adalah suatu objek yang dapat diidentifikasi dalam lingkungan pemakai.
	**Relasi**, menunjukkan adanya hubungan di antara sejumlah entitas yang berbeda.
	**Atribut**, berfungsi mendeskripsikan karakter entitas (atribut yg berfungsi sebagai key diberi garis bawah)
	Garis, sebagai penghubung antara relasi dengan entitas, relasi dan entitas dengan atribut.

Tabel II.1. Simbol ERD

(Sumber : Yuniar Supardi ; 2010 : 448)



Gambar II.2. ERD
Sumber : D Tri Octafian ; 2011 : 153

II.9. Kamus Data

Kamus data (data *dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Figur 6.5 menunjukkan hanya satu tabel dalam basis data jadwal. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel, dan hal-hal lainnya. Nama *field* data, jenis data (seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data, program-program palikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod ; 2008 : 171).

Nama Tabel : **pemasok**
Primary Key : id_pemasok
Foreign Key : -

No	Nama Field	Tipe Data	Panjang
1	Id_pemasok	CHAR	5
2	Nama	VARCHAR	30
3	Alamat	VARCHAR	60
4	Telp	VARCHAR	15

Nama Tabel : **pembelian**
Primary Key : no_beli
Foreign Key : id_pelanggan

No	Nama Field	Tipe Data	Panjang
1	No_beli	CHAR	6
2	Tgl_beli	DATE	-
3	Id_pelanggan	CHAR	5

Gambar II.3. Kamus Data
Sumber : D Tri Octafian ; 2011 : 155

II.10. Teknik Normalisasi

Salah satu topik yang cukup kompleks dalam dunia manajemen *database* adalah proses untuk menormalisasi tabel-tabel dalam *database relasional.* Dengan normalisasi kita ingin mendesain *database relasional* yang terdiri dari tabel-tabel berikut :

a. Berisi data yang diperlukan.
b. Memiliki sesedikit mungkin redundansi.
c. Mengakomodasi banyak nilai untuk tipe data yang diperlukan.
d. Mengefisienkan update.
e. Menghindari kemungkinan kehilangan data secara tidak disengaja/tidak diketahui.

Alasan utama dari normalisasi *database* minimal sampai dengan bentuk normal ketiga adalah menghilangkan kemungkinan adanya "*insertion anomalies"*, "*deletion anomalies",* dan "*update anomalies".* Tipe-tipe kesalahan tersebut sangat mungkin terjadi pada *database* yang tidak normal. Adapun tahap normalisasi sebagai berikut :

1. Tahap tidak normal

   Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti format tertentu, dapat saja tidak lengkap dan terduplikasi. Data dikumpulkan apa adanya sesuai keadaanya.

No_mhs	Nama	Prg_Studi	Kode_mk	Nama_mk	SKS	Kd_Dsn	Dosen
0231	Cahyo	I Komputer	PAM211	Kalkulus Lanjut I	3	MT002	Yasir
			PAAM261	Prg. Terstruktur I	3	IK003	Kamal

Gambar II.4. Tidak Normal
Sumber : Tawar ; 2011 : 7

2. Tahap normal tahap pertama (1" *Normal Form*)

   Sebuah table disebut 1NF jika :

   a. Tidak ada baris yang duplikat dalam tabel tersebut.
   b. Masing-masing cell bernilai tunggal

MHS(No_mhs, Nama, Prg_Studi)

No_mhs	Nama	Prg_Studi
0231	Cahyo	I Komp.
0232	Hoho	Statistik
0233	Budi	Matematika

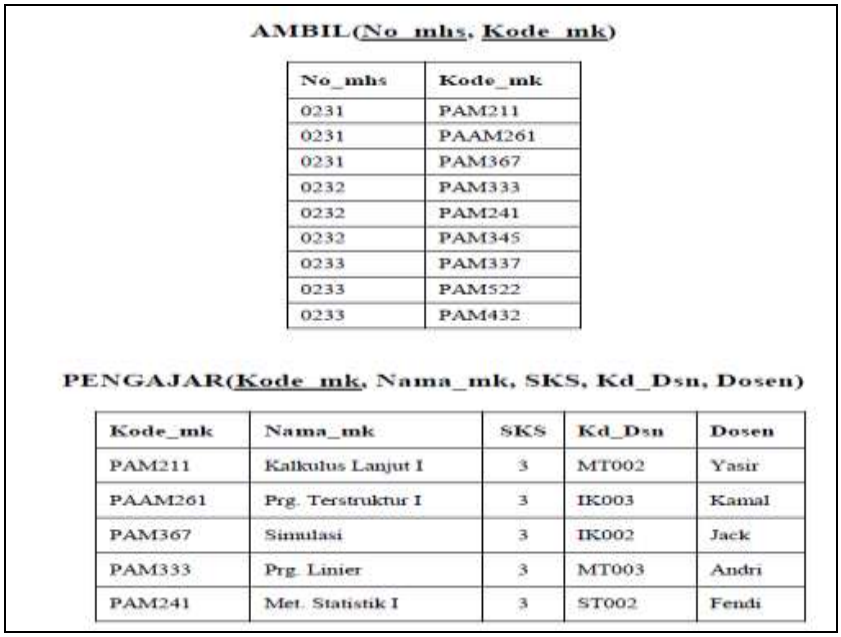
DAFTAR_MK(No_mhs, Kode_mk, Nama_Mk, SKS, Kd_Dsn, Dosen)

No_mhs	Kode_mk	Nama_mk	SKS	Kd_Dsn	Dosen
0231	PAM211	Kalkulus Lanjut I	3	MT002	Yasir
0231	PAAM261	Prg. Terstruktur I	3	IK003	Kamal
0231	PAM367	Simulasi	3	IK002	Jack
0232	PAM333	Prg. Linier	3	MT003	Andri
0232	PAM241	Met. Statistik I	3	ST002	Fendi
0232	PAM345	Analisis Data	3	ST003	Hasbi
0233	PAM337	Fungsi Khas	3	MT001	Jaya
0233	PAM522	Topologi	3	MT003	Andri

Gambar II.5. Normalisasi 1NF
Sumber : Tawar ; 2011 : 8

3. Tahap normal tahap kedua (2nd *normal form*)

   Bentuk normal kedua (2NF) terpenuhi jika pada sebuah tabel semua atribut yang tidak termasuk dalam primary key memiliki ketergantungan fungsional pada primary key secara utuh.



AMBIL(No_mhs, Kode_mk)

No_mhs	Kode_mk
0231	PAM211
0231	PAAM261
0231	PAM367
0232	PAM333
0232	PAM241
0232	PAM345
0233	PAM337
0233	PAM522
0233	PAM432

PENGAJAR(Kode_mk, Nama_mk, SKS, Kd_Dsn, Dosen)

Kode_mk	Nama_mk	SKS	Kd_Dsn	Dosen
PAM211	Kalkulus Lanjut I	3	MT002	Yasir
PAAM261	Prg. Terstruktur I	3	IK003	Kamal
PAM367	Simulasi	3	IK002	Jack
PAM333	Prg. Linier	3	MT003	Andri
PAM241	Met. Statistik I	3	ST002	Fendi

Gambar II.6. Normalisasi 2NF
Sumber : Tawar ; 2011 : 9

4. Tahap normal tahap ketiga (3rd *normal form*)

   Sebuah tabel dikatakan memenuhi bentuk normal ketiga (3NF), jika untuk setiap ketergantungan fungsional dengan notasi X Y Z, dimana Y mewakili semua atribut tunggal di dalam tabel yang tidak ada di dalam X, maka :

   a. X haruslah superkey pada tabel tersebut.
   b. Atau Y merupakan bagian dari primary key pada tabel tersebut.

KULIAH(Kode_mk, Nama_mk, SKS, Kd_Dsn)

Kode_mk	Nama_mk	SKS	Kd_Dsn
PAM211	Kalkulus Lanjut I	3	MT002
PAAM261	Prg. Terstruktur I	3	IK003
PAM367	Simulasi	3	IK002
PAM333	Prg. Linier	3	MT003
PAM241	Met. Statistik I	3	ST002
PAM345	Analisis Data	3	ST003
PAM337	Fungsi Khas	3	MT001
PAM522	Topologi	3	MT003
PAM432	Teori Optimasi	3	MT004

Gambar II.7. Normalisasi 3NF
Sumber : Tawar ; 2011 : 10

5. Boyce Code Normal Form (BCNF)

   a. Memenuhi 1st NF

   b. Relasi harus bergantung fungsi pada atribut superkey

NILAI(No_mhs, No_Rkng, Kd_Mk, Nilai)

No_mhs	Kode_mk	Nilai
0231	PAM211	A
0231	PAAM261	B
0231	PAM367	A
0232	PAM333	C
0232	PAM241	A
0233	PAM345	B
0233	PAM337	A
0235	PAM522	B
0237	PAM432	B

REKENING(No_mhs, No_Rkng)

No_mhs	No_Rkng
0231	88681
0232	88682
0233	88683
0235	88685
0237	88687

Gambar II.8. Normalisasi BCNF
Sumber : Tawar ; 2011 : 12

6. Tahap Normal Tahap Keempat dan Kelima

Penerapan aturan normalisasi sampai bentuk normal ketiga sudah memadai untuk menghasilkan tabel berkualitas baik. Namun demikian, terdapat pula bentuk normal keempat (4NF) dan kelima (5NF). Bentuk Normal keempat berkaitan dengan sifat ketergantungan banyak nilai (*multivalued dependency*) pada suatu tabel yang merupakan pengembangan dari ketergantungan fungsional. Adapun bentuk normal tahap kelima merupakan nama lain dari *Project Join Normal Form* (PJNF) (Kusrini ; 2007 : 39-43).

BAHASA(No_Mhs, Prg_Studi, Bhs_Asing)

No_mhs	Prg_Studi	Bhs_Asing
0232	Komputer	Inggris
0232	Akuntasnsi	Jerman
0232	Komputer	Jerman
0232	Akuntasnsi	Inggris
0236	Statistik	Perancis
0236	Hukum	Belanda
0236	Statistik	Belanda
0236	Hukum	Perancis

BAHASA2(No_Mhs, Prg_Studi, Bhs_Asing)

No_mhs	Prg_Studi	Bhs_Asing
0232	Komputer	Inggris
0232	Akuntasnsi	Jerman

Gambar II.9. Normalisasi 4NF
Sumber : Tawar ; 2011 : 13

II.11. *Unified Modeling Language* (UML)

Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

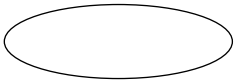
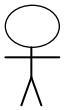
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem (Windu Gata ; 2013 : 4-9).

Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

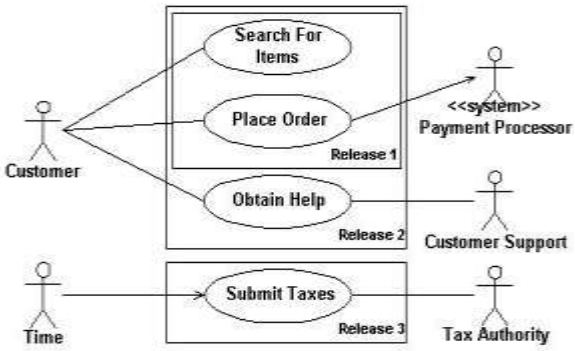
1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan *use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.2. Simbol *Use Case*

Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi dari target sistem. Untuk mengidentifikasikan aktor, harus ditentukan pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan *use case*, tetapi tidak memiliki control terhadap *use case*.
	Asosiasi antara aktor dan *use case*, digambarkan dengan garis tanpa panah yang mengindikasikan siapa atau apa yang meminta interaksi secara langsung dan bukannya mengidikasikan aliran data.
	Asosiasi antara aktor dan *use case* yang menggunakan panah terbuka untuk mengidinkasikan bila aktor berinteraksi secara pasif dengan sistem.
	Include, merupakan di dalam *use case* lain (*required*) atau pemanggilan *use case* oleh *use case* lain, contohnya adalah pemanggilan sebuah fungsi program.
	Extend, merupakan perluasan dari *use case* lain jika kondisi atau syarat terpenuhi.

(Sumber : Windu Gata ; 2013 : 4)



Gambar II.10. *Usecase* Diagram
(Sumber : Haviluddin, jurnal Informatika; 2011 : 04)

2. *Class Diagram* (Diagram Kelas)

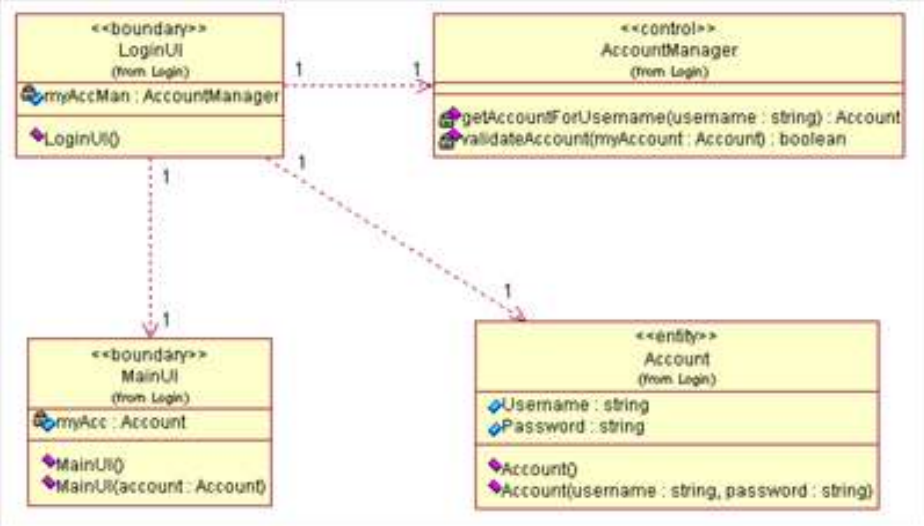
Merupakan hubungan antar kelas dan penjelasan detail tiap-tiap kelas di dalam model desain dari suatu sistem, juga memperlihatkan aturan-aturan dan tanggng jawab entitas yang menentukan perilaku sistem.

Class diagram juga menunjukkan atribut-atribut dan operasi-operasi dari sebuah kelas dan *constraint* yang berhubungan dengan objek yang dikoneksikan. *Class diagram* secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut dengan *multiplicity* atau kardinaliti.

Tabel II.3. *Multiplicity Class Diagram*

Multiplicity	**Penjelasan**
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

(Sumber : Windu Gata ; 2013 : 9)

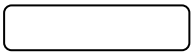
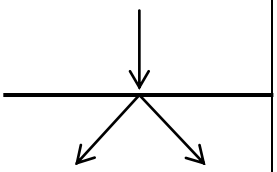
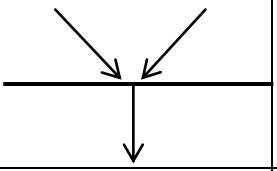
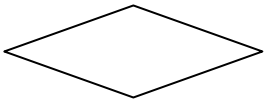


Gambar II.11. *Class* Diagram
(Sumber : Haviluddin, jurnal Informatika; 2011 : 03)

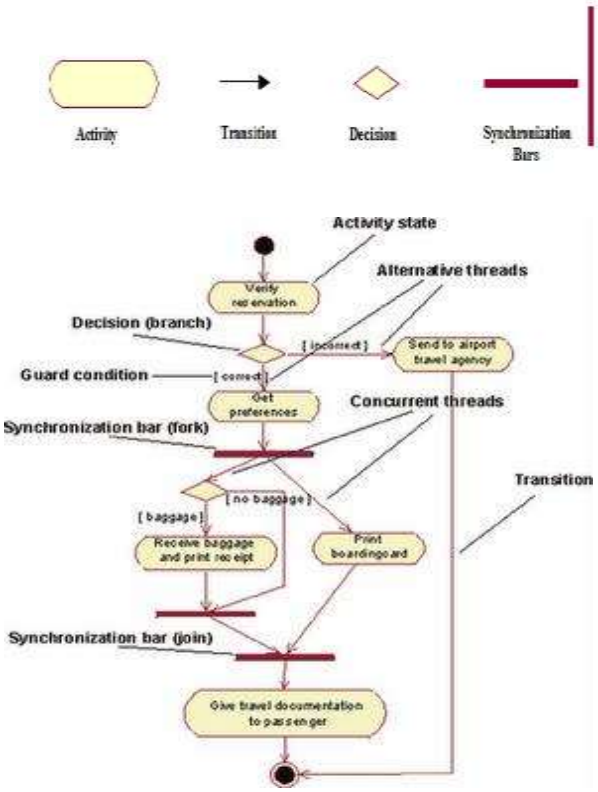
3. Diagram Aktivitas (*Activity Diagram*)

Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel II.4. Simbol *Activity Diagram*

Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false*.

(Sumber : Windu Gata ; 2013 : 6)

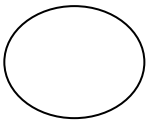
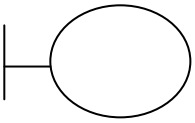
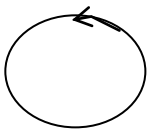
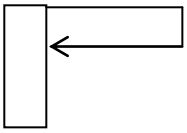


Gambar II.12. *Activity* Diagram
(Sumber : Haviluddin, jurnal Informatika; 2011 : 04)

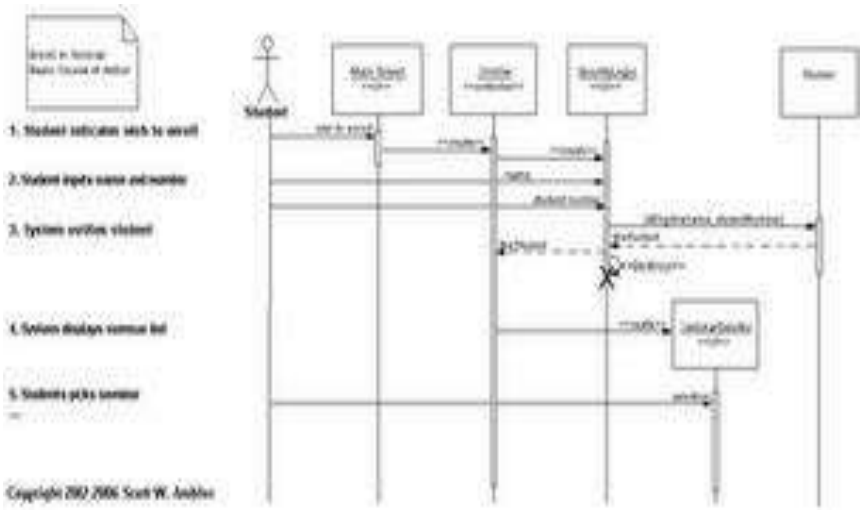
4. Diagram Urutan (*Sequence Diagram*)

Sequence diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup objek dan pesan yang dikirimkan dan diterima antar objek. Simbol-simbol yang digunakan dalam *sequence diagram*, yaitu :

Tabel II.5. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata ; 2013 : 8)



Gambar II.13. *Sequence* Diagram
(Sumber : Haviluddin, jurnal Informatika; 2011 : 05)