BAB II

TINJAUAN PUSTAKA

II.1. Sistem Pendukung Keputusan

Menurut Jurnal Hilyah M. (2012) 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 prosen pengambilan keputusan, sampai mengevaluasi pemilihan alternatif.

Menurut Jurnal Iskandar Z. Nasibu Vol. 2 No. 5 (2009) Sistem Pendukung Keputusan (*Decision Support System/DSS*) adalah sebuah sistem yang memberikan dukungan kepada seorang manajer, atau kepada sekelompok manajer yang relatif yang bekerja sebagai tim pemecah masalah, dalam memecahkan masalah semi terstruktur dengan memberikan informasi atau saran mengenai keputusan tertentu. Informasi tersebut dapat diberikan dalam bentuk laporan berkala, laporan khusus, maupun output dalam model matematis. Model tersebut juga mempunyai kemampuan untuk memberikan saran dalam tingkat yang bervariasi.

II.1.1. Pengertian Sistem

Secara Leksikal, sistem berarti: susunan yang teratur dari pandangan, teori, asas, dan sebagainya. Dengan kata lain, system adalah suatu kesatuan usaha yang terdiri dari bagian-bagin 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 sistem yang meliputi kerjasama antara bagian yang interdependen satu sama yang 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. Defenisi tersebut menunujukkan bahwa sistem sebagai gugus dari elemen-elemen yang saling berinteraksi secara teratur dalam rangka mencapai tujuan atau subtujuan. (Marimin ; 2008 : 1)

Menurut Jurnal Dina Andhayati (2010) Suatu sistem didefenisikan sebagai suatu kesatuan yang terdiri dari dua atau lebih komponen atau subsistem yang berintegrasi untuk mencapai suatu tujuan. Suatu system mempunyai karateristik yaitu mempunyai komponen, batas system, lingkungan luar sistem, penghubung, masukan, keluaran, pengolah dan sasaran.

Menurut Kusrini (2007) Sistem merupakan kumpulan elemen yang saling berkaitan yang bertanggung jawab memproses masukan (*input*) sehingga menghasilkan keluaran (*output*).

II.1.2. Karakterisitik Sistem

Model Umum sebuah sistem terdiri dari input, proses, output. Hal ini merupakan konsep sebuah sistem yang sangat sederhana mengingat sebuah sistem dapat mempunyai beberapa masukan dan keluaran sekaligus. Selain itu sebuah sistem juga memiliki karakteristik atau sifat-sifat tertentu, yang mencirikan bahwa hal tersebut bisa dikatakan sebagai suatu sistem. Adapun karakteristik yang dimaksud adalah sebagai berikut: (Tata Sutabri ; 2012 : 13)

1. Komponen Sistem (*Components*)

   Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang bekerja sama membentuk satu kesatuan. Komponen-komponen tersebut dapat berupa suatu bentuk subsistem. Setiap subsistem memiliki sifat-sifat sistem yang menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai sistem lebih besar yang disebut dengan supra sistem.

2. Batasan Sistem (*Boundary*)

   Ruang lingkup sistem merupakan daerah yang membatasi antara sistem dengan sistem lainnya atau sistem dengan lengkungan luar. Batasan sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan yang tidak dapat dipisah-pisahkan.

3. Lingkungan Luar Sistem (*Environtment)*

   Bentuk apapun yang ada di luar ruang lingkup atau batasan sistem yang mempengaruhi operasi sistem tersebut disebut dengan lingkungan luar sistem ini dapat menguntungkan dan dapat juga merugikan sistem tersebut.

Lingkungan luar yang menguntungkan merupakan energy bagi setiap sistem tersebut, yang dengan demikian lingkungan luar tersebut harus selalu dijaga dan dipelihara. Sedangkan lingkungan luar yang merugikan harus dikendalikan. Kalau tidak maka akan mengganggu kelangsungan hidup sistem tersebut.

4. Penghubung Sistem (*Interface*)

   Media yang menghubungkan sistem dengan subsistem yang lain disebut dengan penghubung sistem atau interface. Penghubung ini memungkinkan sumber-sumber daya mengalir dari suatu subsistem ke subsistem yang lain. Keluaran suatu subsistem akan menjadi masukan untuk subsistem yang lain dengan melewati penguhubung. Dengan demikian terjadi suatu integrasi sistem yang membentuk suatu kesatuan. (Tata Sutabri ; 2012 : 13)

II.1.3. Pengertian Keputusan

Menurut Jurnal Ellya Sestri (2013) pengambilan keputusan merupakan hal vital dalam menentukan kebijakan yang harus diambil dalam menghadapi persaingan di dunia bisnis. Pengambilan keputusan dapat dipengaruhi oleh beberapa aspek, dan hal ini dapat mempengaruhi kecepatan dalam mengambil keputusan dimana pengambilan keputusan harus cepat dan akurat.

II.1.4. Pengertian Sistem Pendukung Keputusan

Dalam Jurnal Nila Susanti dan Sri Winiarti (2013) Sistem pendukung keputusan menurut *Gorry Dan Scout Morton* adalah sistem berbasis komputer

interaktif, yang membantu para pengambil keputusan untuk menggunakan data dan berbagai model untuk memecahkan masalah-masalah tidak terstruktur.

Dalam Jurnal Dita Monita (2013) menjelaskan Sistem pendukung keputusan adalah bagian dari sistem informasi berbasis komputer (termasuk sistem berbasis pengetahuan (manajemen pengetahuan)) yang dipakai untuk mendukung pengambilan keputusan dalam suatu organisasi atau perusahaan. Dapat juga dikatakan sebagai sistem komputer yang mengolah datamenjadi informasi untuk mengambil keputusan dari masalah semiterstruktur yang spesifik.

II.1.5. Tahap-Tahap Pengambilan Keputusan

Dalam Jurnal Hilyah M. (2012), Menurut Herbert A. Simon (Kadarsyah Suryadi dan Ali Ramdhani, 2002, 15-16) model yang menggambarkan proses pengambilan keputusan. Proses Pengambilan Keputusan melibatkan 4 tahapan, yaitu:

a. Tahap *Intelligence*

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

b. Tahap *Design*

   Dalam tahap ini merupakan proses menemukan, mengembangkan dan menganalisis alternatif tindakan yang bisa dilakukan.

c. Tahap *Choice*

Dalam tahap ini dilakukan proses pemilihan diantara berbagai alternatif tindakan yang mungkin dijalankan.

d. Tahap *Implementation*

Dalam tahap ini pengambil keputusan menjalankan rangkaian aksi pemecahan yang dipilih di tahap *choice.* Implementasi yang sukses ditandai dengan terjawabnya masalah yang dihadapi, sementara kegagalan ditandai dengan tetap adanya masalah yang sedang dicoba untuk diatasi.

II.1.6. Komponen Sistem Pendukung Keputusan

Dalam Jurnal Nila Susanti dan Sri Winiarti (2013) Sistem Pendukung Keputusan terdiri atas empat komponen utama, yaitu:

1. Subsistem manajemen data berfungsi sebagai memasukkan suatu *database* yang berisi data yang relevan untuk situasi dan dikelola oleh perangkat lunak yang disebut yang disebut sistem manajemen *database* (DBMS). *Knowledge Base* berisi semua fakta, ide, hubungan dan interaksi suatu domain tertentu.
2. Subsistem manajemen basis pengetahuan bertugas untuk mendukung semua subsistem lain atau bertindak sebagai suatu komponen independen. Ia memberikan intelegensi untuk memperbesar pengetahuan pengambil keputusan.
3. Subsistem manajemen model merupakan paket perangkat lunak yang memasukkan model keuangan statistik, ilmu manajemen atau model

kuantitatif lainnya yang memberikan kapabilitas analitik dan manajemen perangkat lunak yang tepat.

4. Subsistem antarmuka pengguna (dialog) untuk mengimplementasikan sistem kedalam program aplikasi sehinggga pengguna atau pemakai dapat berkomunikasi dengan sistem yang dirancang.

II.I.7. Kriteria Sistem Pendukung Keputusan

Dalam Jurnal Dita Monita (2013) Sistem pendukung keputusan dirancang secara khusus untuk mendukung seseorang yang harus mengambil keputusan-keputusan tertentu [1]. Berikut ini beberapa criteria sistem pendukung keputusan, yaitu:

1. Interaktif

   Sistem pendukung keputusan memiliki *user interface* yang komunikatif sehingga pemakai dapat melakukan akses secara cepat ke data dan memperoleh informasiyang dibutuhkan.

2. Fleksibel

   Sistem pendukung keputusan memiliki sebanyak mungkin variable masukan, kemampuan untuk mengolah dan memberikan keluaran yang menyajikan alternatif-alternatif keputusan kepada pemakai.

3. Data Kualitas

   Sistem pendukung keputusan memiliki kemampuan untuk menerima data kualitas yang dikuantitaskan yang sifatnya subyektif dari pemakainya, sebagai data masukan untuk pengolahan data. Misalnya terhadap kecantikan

yang bersifat kualitas, dapat dikuantitaskan dengan pemberian bobot nilai seperti 75 atau 90.

4. Prosedur Pakar

   Sistem pendukung keputusan mengandung suatu prosedur yang dirancang berdasarkan rumusan formal atau juga berupa prosedur kepakaran seseorang atau kelompok dalam menyelesaikan suatu bidang masalah dengan fenomena tertentu.

II.2. Metode *Analytical Hierarchy Process* (AHP)

Menurut Jurnal Hilyah M. (2012), *Analytical Hierarchy Process* (AHP) dikembangkan oleh Thomas L. Saaty pada tahun 1970-an. Metode ini merupakan salah satu model pengambilan keputusan multikriteria yang dapat membantu kerangka berpikir manusia dimana faktor logika, pengalaman pengetahuan, emosi dan rasa dioptimasikan ke dalam suatu proses sistematis. Pada dasarnya, AHP merupakan metode yang digunakan untuk memecahkan masalah yang kompleks dan tidak terstruktur ke dalam kelompok–kelompoknya, dengan mengatur kelompok tersebut ke dalam suatu hierarki, kemudian memasukkan nilai numerik sebagai pengganti persepsi manusia dalam melakukan perbandingan relatif. Dengan suatu hipotesa maka akan dapat ditentukan elemen mana yang mempunyai prioritas tertinggi.

II.2.1. Pengertian *Analytical Hierarchy Process* (AHP)

Berdasarkan Jurnal (Krupesh A Chauhan,et al. (2008*)* *Analytical Hierarchy Process* adalah teknik pengambilan keputusan yang dikembangkan pada 1970-an oleh L. Saaty dengan melakukan perhitungan untuk keputusan yang lebih kompleks, tidak terstruktur dan mempunyai banyak kriteria.

Menurut Jurnal Dita Monita Vol. III No. 2 (2013) *Analytical Hierarchy Process* (AHP) merupakan salah satu metode untuk membantu menyusun suatu pprioritas dari berbagai pilihan dengan menggunakan berbagai kriteria. Karena sifatnya yang multikriteria, *Analytical Hierarchy Process* cukup banyak digunakan dalam penyusunan prioritas. Disamping bersifat multikriteria, *Analytical Hierarchy Process* juga didasarkan pada suatu proses yang terstruktur dan logis. Pemilihan atau penyusunan prioritas dilakukan dengan suatu prosedur yang logis dan terstruktur.

II.2.2. Tahapan–Tahapan Proses Metode *Analytic Hierarchy Process* (AHP)

Berdasarkan jurnal Yusuf Anshori (2012) menjelaskan bahwa secara umum, tahapan-tahapan proses yang harus dilakukan dalam menggunakan AHP untuk memecahkan suatu masalah adalah sebagai berikut :

1. Mendefenisikan permasalahan dan menentukan tujuan. Bila AHP digunakan untuk memilih alternatif atau menyusun prioritas alternatif, maka tahap ini dilakukan pengembangan alternatif.
2. Menyusun masalah ke dalam suatu struktur hierarki sehingga permasalahan yang kompleks dapat ditinjau dari sisi yang detail dan terukur .

3. Menyusun prioritas untuk tiap elemen masalah pada setiap hierarki. Prioritas ini dihasilkan dari suatu matriks perbandingan berpasangan antara seluruh elemen pada tingkat hierarki yang sama.
4. Melakukan pengujian konsistensi terhadap perbandingan antar elemen yang didapatkan pada tiap tingkat hierarki. Thomas L. Saaty membuktikan bahwa Indeks Konsistensi dari matriks berordo n diperoleh rumus sebagai berikut:

$$CI = \frac{\lambda_{max} - n}{n - 1} \qquad \text{……..Pers(2.1)}$$

Dimana :

CI = *Consistency Index* (Rasio penyimpangan konsistensi)

λ_{max} = Nilai eigen terbesar dari matriks berordo *n*

n = jumlah elemen yang dibandingkan

Nilai *CI* bernilai nol apabila terdapat standar untuk menyatakan apakah *CI* menunjukkan matriks yang konsisten. Saaty berpendapat bahwa suatu matriks yang dihasilkan dari perbandingan yang dilakukan secara acak merupakan suatu matriks yang tidak konsisten. Dari matriks acak didapatkan juga nilai *Consistency Index* yang disebut dengan *Random Index* (*RI*).

Dengan membandingkan *CI* dengan *RI* maka didapatkan patokan untuk menentukan tingkat konsistensi suatu matriks yang disebut dengan *Consistency Ratio* (CR) dengan rumus :

$$CR = CI / RI$$

.....Pers(2.2)

Dimana :

CR = Consistency Ratio

RI = *Random Index*

II.2.3. Prinsip Dasar *Analytic Hierarchy Process* (AHP)

Berdasarkan Jurnal Ellya Sestri (2013) menjelaskan prinsip dasar AHP (*Analytic Hierarchy Process*) yang harus dipahami diantaranya adalah sebagai berikut :

1. *Reciprocal Comparison*

   Artinya pengambilan keputusan harus dapat memuat perbandingan dan menyatakan preferensinya. Preferensi tersebut harus memenuhi syarat resiprokal yaitu apabila A lebih disukai daripada B dengan skala x, maka B lebih disukai daripada A dengan skala 1/x.

2. *Homogenity*

   Artinya preferensi seseorang harus dapat dinyatakan dalam skala terbatas atau dengan kata lain elemen-elemennya dapat dibandingkan satu sama lainnya. Kalau aksioma ini tidak dipenuhi maka elemen-elemen yang dibandingkan tersebut tidak homogeny dan harus dibentuk *cluster* (kelompok elemen) yang baru.

3. *Independence*

   Artinya preferensi dinyatakan dengan mengasumsikan bahwa criteria tidak dipengaruhi oleh alternatif-alternatif yang ada melainkan oleh objektif keseluruhan. Ini menunjukkan bahwa pola ketergantungan dalam AHP

adalah searah, maksudnya perbandingan antara elemen-elemen dalam satu tingkat dipemngaruhi atau tergantung oleh elemen-elemen pada tingkat di atasnya.

4. *Expectation*

   Artinya untuk tujuan mengambil keputusan. Struktur hirarki diasumsikan lengkap. Apabila asumsi ini tidak dipenuhi maka pengambilan keputusan tidak memakai seluruh criteria atau objektif yang tersedia atau diperlukan sehingga keputusan yang diambil dianggap tidak lengkap.

Metode AHP Merupakan sebuah hirarki fungsional dengan input utama yang berupa persepsi manusia. Suatu masalah yang kompleks dan tidak terstruktur dipecah kedalam kelompok-kelompok yang kemudian diatur menjadi suatu bentuk hirarki. Konsep utama dalam AHP adalah *preference.* Supermatriks dalam AHP terdiri dari 3 tahap, yaitu:

1. Tahap supermatriks tanpa bobot (*unweighted supermatrix*), adalah supermatrix yang diperoleh dari bobot yang didapat dari matriks perbandingan berpasangan.
2. Tahap supermatriks terbobot (*weighted supermatrix*) adalah supermatriks yang didapat dari perkalian semua elemen didalam komponen dari *unweighted supermatrix* dengan bobot *cluster* yang sesuai sehingga setiap kolom pada *weighted supermatrix* memilki jumlah 1. Jika kolom pada *unweighted supermatrix* sudah memiliki jumlah 1, maka tidak perlu membobot komponen tersebut pada *weighted supermatrix.*

3. Tahap supermatriks batas (*limit supermatrix)*, adalah supermatriks yang diperoleh dengan menaikkan bobot dari *weighted supermatrix*. Menaikkan bobot *weighted supermatrix* dilakukan dengan cara mengalihkan supermatriks itu dengan dirirnya sendirir secara berulang-ulang. Ketika bobot pada setiap kolom memiliki nilai yang sama, maka limit matrix telah stabil dan proses perkalian matrix dihentikan. Hasil akhir perhitungan memberikan bobot prioritas dan sintesis.

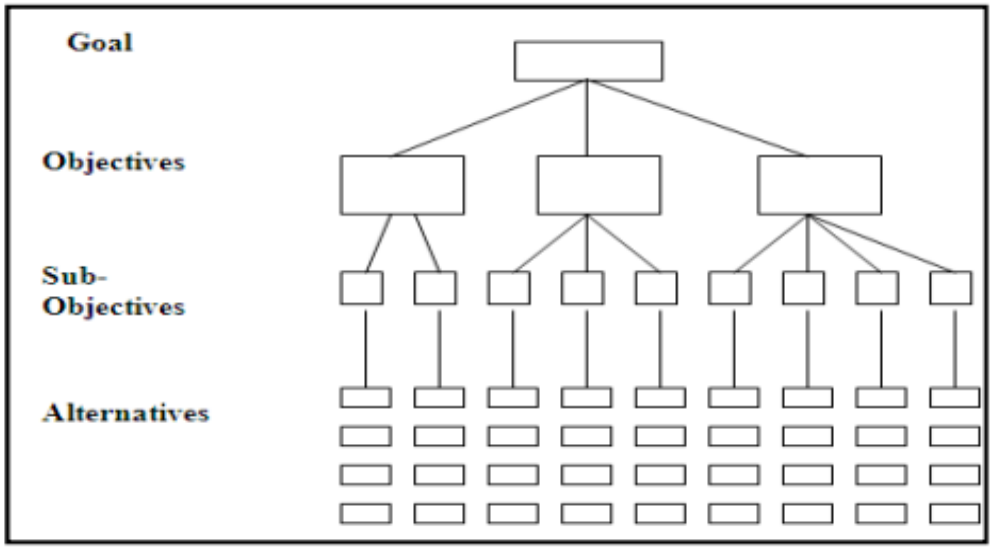
II.2.4 Prinsip Kerja *Analytic Hierarchy Process* (AHP)

Menurut Jurnal Iskandar Z. Nasibu prinsip kerja AHP adalah penyederhanaan suatu persoalan kompleks, yang tidak terstruktur, stratejik, dan dinamik menjadi bagian-bagiannya, serta menata dalam suatu hirarki. Penggunaan AHP dimulai dengan membuat struktur hirarki dari permasalahan (dekomposisi), melakukan perbandingan berpasangan antar variable, melakukan analisis/evaluasi, dan menentukan alternative terbaik (Saaty, 1993).

Lebih lanjut, Suryadi dan Ramadhani (2000) mengemukakan bahwa pada dasarnya langkah-langkah dalam metode AHP diuraikan sebagai berikut :

1. Menyusun Hirarki dari permasalahan yang dihadapi

Persoalan yang akan diselesaikan menjadi unsure-unsurnya, yaitu criteria dan alternative, kemudian disusun menjadi struktur hirarki seperti Gambar II.1. dibawah ini:



Gambar II.1. Struktur Hierarki AHP

Sumber : Jurnal Iskandar Z. Nasibu Volume 2 No. 5 (2009)

2. Penilaian Kriteria dan Alternatif

Kriteria dan alternative dinilai melalui perbandingan berpasangan. Menurut Saaty (1988), untuk berbagai persoalan, skala 1 samapai 9 adalah skala terbaik dalam mengekspresikan pendapat. Nilai dan defenisi pendapat kualitatif dari skala perbandingan Saaty dapat dilihat pada tabel II.1

Tabel II.1. Skala Penilaian Perbandingan Berpasangan

Intensitas Kepentingan	Keterangan
1	Kedua elemen sama pentingnya
3	Elemen yang satu sedikit lebih penting daripada elemen yang lainnya
5	Elemen yang satu lebih penting daripada elemen lainnya
7	Satu elemen mutlak penting daripada elemen lainnya
9	Satu elemen mutlak penting daripada elemen lainnya
2,4,6,8	Nilai-nilai antara dua nilai pertimbangan yang berdekatan
Kebalikan	Jika aktivitas i mendapat satu angka dibandingkan dengan aktivitas j, maka j memiliki nilai kebalikannya dibandingkan dengan i

Sumber : Jurnal Iskandar Z. Nasibu Volume 2 No. 5 (2009)

Perbandingan dilakukan berdasarkan kebijakan pembuat keputusan \dengan menilai tingkat kepentingan satu elemen terhadap elemen lainnya. Proses perbandingan berpasangan, dimulai dari level hirarki paling atas yang ditujukan untuk memilik criteria, Misalnya A, kemudian diambil elemen yang akan dibandingkan, missal A1, A2, dan A3. Selanjutnya susunan elemen-elemen yang dibandingkan tersebut akan tampak seperti pada tabel matriks dibawah ini:

Tabel II.2. Contoh Matriks Perbandingan Berpasangan

	AI	A2	A3
A1	1		
A2		1	
A3			1

Sumber : Jurnal Iskandar Z. Nasibu Volume 2 No. 5 (2009)

3. Penentuan Prioritas

Untuk setiap criteria dan alternative, perlu dilakukan perbandingan berpasangan (pairwise comparison). Nilai-nilai perbandingan relative kemudian diolah untuk menentukan peringkat alternative dari seluruh alternatif. Pertimbangan-pertimbangan terhadap perbandingan berpasangan disintesis untuk memperoleh keseluruhan prioritas melalui tahapan-tahapan berikut:

a. Kuadratkan matriks hasil perbandingan berpasangan.
b. Hitung jumlah nilai dari setiap baris, kemudian lakukan normalisasi matriks.

4. Konsistensi Logis

Semua elemen dikelompokkan secara logis dan diperingatkan secara konsisten sesuai dengan sutau criteria yang logis.

II.3. Pengertian PHP

PHP merupakan bahasa skrip yang digunakan untuk membuat halaman web yang dinamis. PHP bersifat *open product.* Penggunaan dapat mengubah *source code* dan mendistribusikannya secara bebas serta diedarkan secara gratis . PHP bersifat *server scripting* yang dapat ditambhakan kedalam HTML, sehingga suatu halaman web tidak lagi bersifat statis, namun bersifat dinamis. Sifat *server-side* berarti pengerjaan skrip PHP akan dilakukan di sebuah *web server*, kemudian hasilnya akan dikirimkan ke *browser.* Salah satu *web server* yang paling umum digunakan untuk PHP adalah Apache. PHP dapat dijalankan pada sistem operasi *Unix, Windows,* dan Mac OS X.

PHP Hypertext Preprocessor atau sering disebut PHP merupakan bahasa pemrograman berbasis server-side yang dapat melakukan parsing script php menjadi script web sehingga dari sisi client menghasilkan suatu tampilan yang menarik. PHP merupakan pengembangan dari FI atau Form Interface yang dibuat oleh Rasmus Lerdoff pada tahun 1995. (YM Kusuma Ardhana ; 2012 ; 88)

PHP (*PHP Hypertext Preprocessor)* adalah kode/skrip yang akan dieksekusi pada server side. Skrip PHP akan membuat suatu aplikasi dapat di-integrasikan ke dalam HTML, sehingga suatu halaman web tidak lagi bersifat statis, namun menjadi bersifat dinamis. Sifat server-side berarti pengerjaan skrip dilakukan di server, baru kemudian hasilnya dikirimkan ke browser. (Deni Sutaji ; 2012 ; 2)

II.3.1. Aturan PHP

Adapun aturan penulisan skrip PHP ada dua cara, yaitu:

1. *Embedded Script*

   Dengan cara meletakkan tag PHP diantara tag-tag HTML, contohnya:

   <html>

   <body>

   <?php echo ”Belajar”;?>

   </body>

   </html>

2. *Non Embedded Script*

   Dengan cara ini, semua script HTML diletakkan didalam skrip PHP. Contohnya:

   <?php

   Echo ”<html>”;

   Echo ”<body>”;

   Echo ”Belajar PHP”;

   Echo ”</body>”;

   Echo ”</html>”;

   ?> (Deni Sutaji ; 2012 ; 2)

II.4. Pengertian MySQL

MySQL *database server* adalah RDBMS (*Relasional Database Management Sistem*) yang dapat menangani data bervolume besar. Meskipun

begitu, tidak menuntut *resource* yang besar. MySQL adalah *database* yang paling populer diantara *database* yang lain.

MySQL adalah program database yang mampu mengirim dan menerima data dengan sangat cepat dan multi user. MySQL memiliki dua bentuk lisensi, yaitu free software dan shareware. Penulis sendiri dalam menjelaskan buku ini menggunakan MySQL yang free software karena bebas menggunakan databse ini untuk keperluan probadi atau usaha tanpa harus membeli atau membayar lisensi yang berada di bawah lisensi GNU/GPL (General Public License), yang dapat di download pada alamat resminya *http://www.mysql.com*. MySQL sudah cukup lama dikembangkan, beberapa *fase* penting dalam pengembangan MySQL adalah sebagai berikut:

- MySQL dirilis pertama kali secara secara internal pada 23 Mei 1995
- Versi *windows* dirilis pada 8 Januari 1998 untuk *windows* 95 dan *windows* NT
- Versi 3.23 : beta dari Juni 2000, dan dirilis pada Januari 2001.
- Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (unions)

(Wahana Komputer ; 2010 ; 5)

II.5. *Unified Modelling Languag* (UML)

UML singkatan dari Unified Modeling Language yang berarti bahasa pemodelan standar. Chonoles mengatakan sebagai bahasa, berarti UML memiliki sintaks dan semantic. Ketika kita membuat model menggunakan konsep UML ada aturan-aturan yang harus diikuti. Bagaimana elemen pada model-model yang kita

buat berhubungan satu dengan lainnya harus mengikuti standar yang ada. UML bukan hanya sekedar diagram, tetapi juga menceritakan konteksnya. (Prabowo et al; 2011 : 6)

UML diaplikasikan untuk maksud tertentu, biasanya antara lain:

1. Merancang perangkat lunak
2. Sarana komunikasi antara perangkat lunak dengan proses bisnis.
3. Menjabarkan sistem secara rinci untuk analisa dan mencari apa yang diperlukan sistem.
4. Mendokumentasi sistem yang ada, proses-proses dan organisasinya.

Blok pembangunan utama UML adalah diagram. Beberapa diagram ada yang rinci (jenis *timing diagram*) dan lainnya ada yang bersifat umum (misalnya diagram kelas). Para pengembangan sistem berorientasi objek menggunakan bahasa model untuk menggambar, membangun dan mendokumentasikan sistem yang mereka rancang. UML memungkinkan para anggota team untuk bekerja sama dengan bahasa model yang sama dengan mengaplikasikan beragam sistem. Intinya, UML merupakan alat komunikasi yang konsisten dalam mensupport para pengembangan sistem saat ini. (Prabowo Pudjo Widodo, Herlawati; 2011 : 6-7)

II.5.1. Diagram-Diagram UML

Beberapa leteratur menyebutkan bahwa UML menyediakan Sembilan jenis diagram, yang lain menyebutkan delapan karena ada beberapa yang digabung, misalnya diagram komunikasi, diagram urutan, dan diagram pewaktuan digabung menjadi diagram interaksi. Namun demikian model-model itu dapat

dikelompokkan berdasarkan sifatnya yaitu statis atau dinamis. Jenis diagram itu antara lain:

1. Diagram Kelas, bersifat statis. Diagram ini memperlihatkan himpunan kelas-kelas, antarmuka-antarmuka, kolaborasi-kolaborasi, serta relasi-relasi. Diagram ini umum dijumpai pada pemodela sistem berorientasi objek . Meskipun bersifat statis, sering pula diagram kelas mmemuat kelas-kelas aktif.
2. Diagram paket (Package Diagram), bersifat statis. Diagram ini memperlihatkan kumpulan kelas-kelas, merupakan bagian dari diagram komponen.
3. Diagram *use case*, bersifat statis. Diagram ini memperlihatkan himpunan *use case* dan aktor-aktor (suatu jenis khusus dari kelas). Diagram ini terutama sangat penting untuk mengorganisasi dan memodelkan perilaku suatu sistem yang dibutuhkan serta diharapkan pengguna.
4. Diagram interaksi dan *sequence* (urutan), bersifat dinamis. Diagram urutan adalah diagram interaksi yang menekankan pada pengiriman pesan dalam suatu waktu tertentu.
5. Diagram komunikasi (*Communication Diagram*), bersifat dinamis. Diagram sebagai pengganti diagram kolaborasi UML 14 yang menekankan organisasi structural dari objek-objek yang menerima seta mengirim pesan.
6. Diagram *Statechart* (*Statechart Diagram*), bersifat dinamis. Diagram status memperlihatkan keadaan-keadaan pada sistem, memuat status (*state*), transisi, kejadian serta aktifitas. Diagram ini teruama penting untuk

memperlihatkan sifat dinamis dari antarmuka (*interface*), kelas, kolaborasi, dan terutama penting pada pemodelan sistem-sisten yang reaktif.

7. Diagram aktivitas (*Aktivity Diagram*), bersifat dinamis. Diagram aktivitas adalah tipe khusus dari diagram status yang memperlihatkan aliran dari suatu aktivitas lainnya dalam suatu sistem. Diagram ini terutama penting dalam pemodelan fungsi-fungsi sistem dan member tekanan pada aliran kendali antar objek.
8. Diagram komponen (*Component Diagram*), bersifat statis. Diagram komponen ini memeperlihatkan organisasi serta keberuntungan sistem / perangkat lunak pada komponen-komponen yang telah ada sebelumnya. Diagram ini berhubungan dengan diagram kelas dimana komponen secara tipikal dipetakan ke dalam satu atau lebih kelas-kelas, antaarmuka-antramuka serta kolaborasi-kolaborasi.
9. Diagram *Deployment (Deployment Diagram)*, bersifat statis. Diagram ini memperlihatkan konfigurasi saat aplikasi dijalankan (*run-time*). Memuat simpul-simpul beserta komponen-komponen yang ada di dalamnya. Diagram *deployment* berhubungan erat dengan diagram komponen dimana diagram ini memuat satu atau lebih komponen-komponen. Diagram ini sangat berguna saat aplikasi kita berlaku sebagai aplikasi yang dijalankan pada banyak mesin (*Distributed Computing*)

Kesembilan diagram ini tidak mutlak harus digunakan dalam pengembangan perangkat lunak, semuanya dibuat sesuai dengan kebutuhan. Pada UML dimungkinkan kita menggunakan diagram-diagram lainnya (misalnya *Data Flow*

Diagram, *Entity Relationship Diagram* dan sebagainya). (Prabowo Pudjo Widodo, Herlawati; 2011 : 10-12)

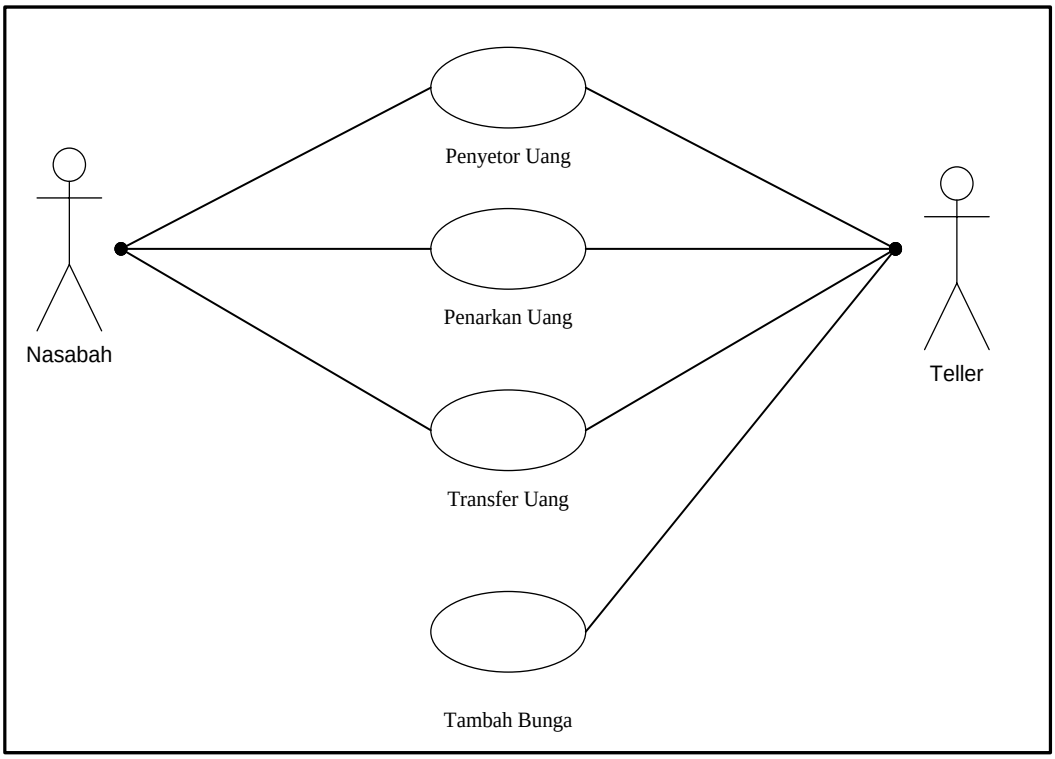
a. Diagram *Use Case* (*Use Case Diagram*)

Use Case menggambarkan *external view* dari sistem yang akan kita buat modelnya. MenurutPooley (2005:15) mengatakan bahwa model *use case* dapat dijabarkan dalam diagram, tetapi yang perlu diingat, diagram tidak identik dengan model karena model lebih luas dari diagram.

Komponen pembentuk diagram *use case* adalah:

1) Aktor (*actor*), menggambarkan pihak-pihak yang berperan dalam sistem.
2) *Use Case*, aktifitas/sarana yang disiapkan oleh bisnis / sistem.
3) Hubungan (*Link*), actor mana saja yang terlibat dalam *use case* ini.

Gambar dibawah ini merupakan salah satu contoh bentuk diagram *use case* yaitu :

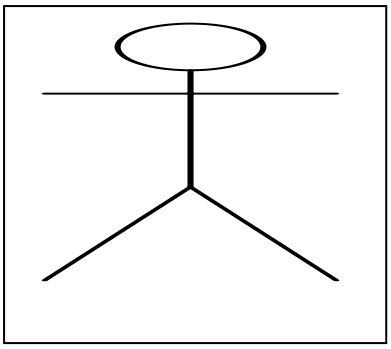


Gambar II.2. Diagram *Use Case*

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 17)

a) Aktor

Menurut Chonoles (2003 : 17) menyarankan sebelum membuat *use case* dan menentukan aktornya, agar mengidentifikasikan siapa saja pihak yang terlibat dalam sistem kita. Pihak yang terlibat biasanya dinamakan *stakeholder*.

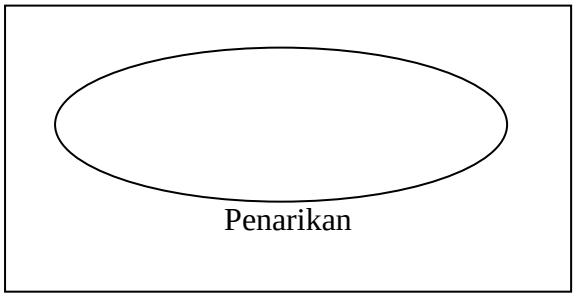


Gambar II.3. Aktor

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 17)

b) Use Case

Menurut Pilone (2005 : 21) use case menggambarkan fungsi tertentu dalam suatu sistem berupa komponen, kejadian atau kelas. Sedangkan menurut Whitten (2004 : 258) mengartikan use case sebagai urutan langkah-langkah yang secara tindakan saling terkait (scenario), baik terotomatisasi maupun secara manual, untuk tujuan melengkapi suatu tugas bisnis tunggal. Use case digambarkan dalam bentuk ellips/oval pada gambar II.3 sebagai berikut :



Gambar II.4. Simbol Use Case

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 22)

Use case sangat menentukan karakteristik sistem yang kita buat, oleh karena itu (Chonoles, 2003: 22) menawarkan cara untuk mengahasilkan use case yang baik, yakni:

1) Pilihlah nama yang baik

   Use case adalah sebuah *behavior* (perilaku), jadi seharusnya dalam frase kata kerja. Untuk membuat namanya lebih detil, tambahkan kata benda yang mengidentifikasikan dampak aksinya terhadap suatu kelas objek. Oleh karena itu diagram *use case* seharusnya berhubungan dengan diagram kelas.

2) Ilustrasikan perilaku dengan lengkap

   Use case dimulai dari inisiasi oleh aktor primer dan berakhir pada actor dan menghasilkan tujuan. Jangan membuat *use case* kecuali Anda mengetahui tujuannya. Sebagai contoh, memilih jenis tempat tidur (*king size*, *queen size* atau dobel) saat tamu memesan tidak dapat dijadikan use case karena merupakan bagian dari use case pemesanan kamar dan tidak dapat berdiri sendiri (tidak mungkin tamu memesan kamar tidur jenis king tapi tidak memesan kamar hotel).

3) Identifikasi perilaku dengan lengkap

   Untuk mencapai tujuan dan menghasilkan nilai tertentu dari aktor, use case harus lengkap. Ketika member nama pada *use case*, plihlah frasa kata kerja yang implikasinya hingga selesai. Misalnya gunakan frasa *reserve a room* (pemesanan kamar) dan jangan *reserving a room* (memesan kamar) karena memesan kamar menggambarkan perilaku yang belum selesai.

4) Menyediakan *Use case* lawan (inverse)

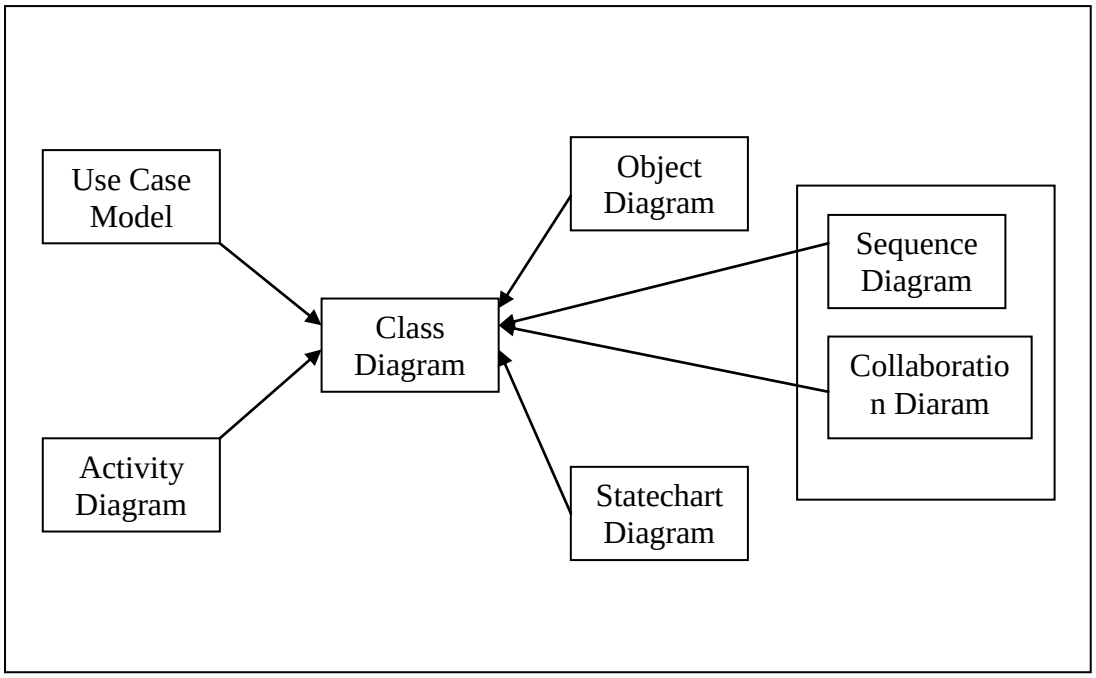
Kita biasanya membutuhkan *use case* yang membatalkan tujuan misalnya pada *use case* pemesanan kamar, dibutuhkan pada *use case* pembatalan pesanan kamar.

5) Batasi Use case hingga satu perilaku saja

Kadang kita cenderung membuat *use case* yang menghasilkan lebih dari satu tujuan aktivitas. Guna menghindari kerancuan, jagalah *use case* kita hanya fokus pada satu hal. Misalnya, pengguna *use case* menghasilkan ketidakfokusan, karena memiliki dua perilaku yang berbeda. Untuk menyediakan penjelasan detail terhadap segala kemungkinan yang terjadi pada *use case*, apa yang terjadi dan apa respon sistem.

b. Diagram kelas (Class Diagram)

Diagram kelas adalah inti dasri proses pemodelan objek, baik forward engineering maupun reverse engineering memanfaatkan diagram ini. Forward engineering adalah proses perubahan model menjadi kode program dsedangkan reserve engineering sebaliknya merubah kode program menjadi model.



Gambar II.5. Hubungan diagram kelas dengan diagram UML lainnya
Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 38)

c. Diagram Aktivitas (Activity Diagram)

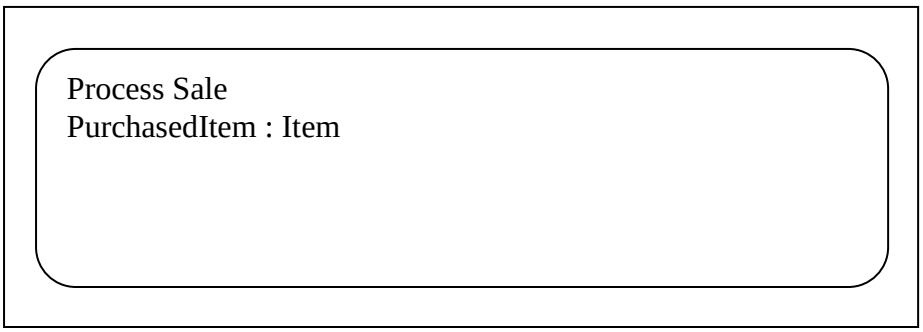
Diagram aktivitas lebih memfokuskan dari pada eksekusi dan alur sistem dari pada bagaimana sistem itu dirakit. Diagram ini tidak hanya memodelkan model bisnis juaga. Diagram aktivitas menunjukkan aktivitas sistem dalam bentuk kumpulan aksi-aksi. Ketika digunakan dalam pemodelan software, diagram aktivitas mempresentasikan pemanggilan suatu fungsi tertentu misalnya *call*. Sedangkan bila digunakan dalam pemodelan bisnis, diagram ini menggambarkan aktivitas yang dipicu oleh kejadian-kejadian di luar seperti pemesanan atau kejadian-kejadian internal misalnya proses penggajian tiap jumat sore.

Aktivitas merupakan kumpulan aksi-aksi. Aksi-aksi melakukan langkah sekali saja tidak boleh dipecah menjadi beberapa langkah lagi. Contoh akasi yaitu:

1) Fungsi matematika
2) Pemanggilan perilaku
3) Pemrosesan data

Ketika kita menggunakan diagram aktivitas untuk memodelkan perilaku suatu classifier, classifier dikatakan konteks dari aktivitas. Aktivitas dapat mengakses atribut dan operasi classifier, tiap objek yang terhubung dan parameter-parameter jika aktivitas memiliki hubungan dengan perilaku. Ketika digunakan untuk model proses bisnis, informasi itu biasanya disebut process-relevant data. Aktivitas diharapkan dapat digunakan ulang dalam suatu aplikasi, sedangkan aksi biasanya specific dan digunakan hanya untuk aktivitas tertentu.

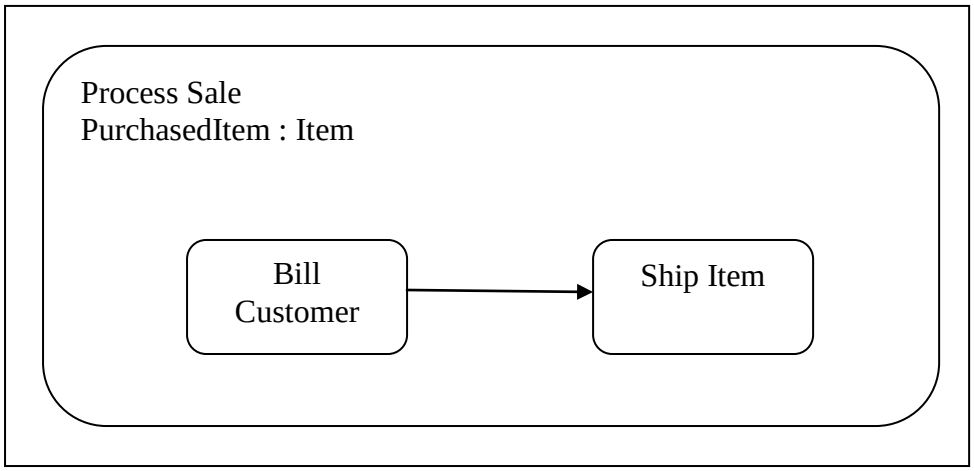
Aktivitas digambarkan dengan persegi panjang tumpu, namanya ditulis di kiri atas. Parameter yang terlibat dalam aktivitas ditulis dibawahnya.



Gambar II.6. Aktivitas sederhana tanpa rincian

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 145)

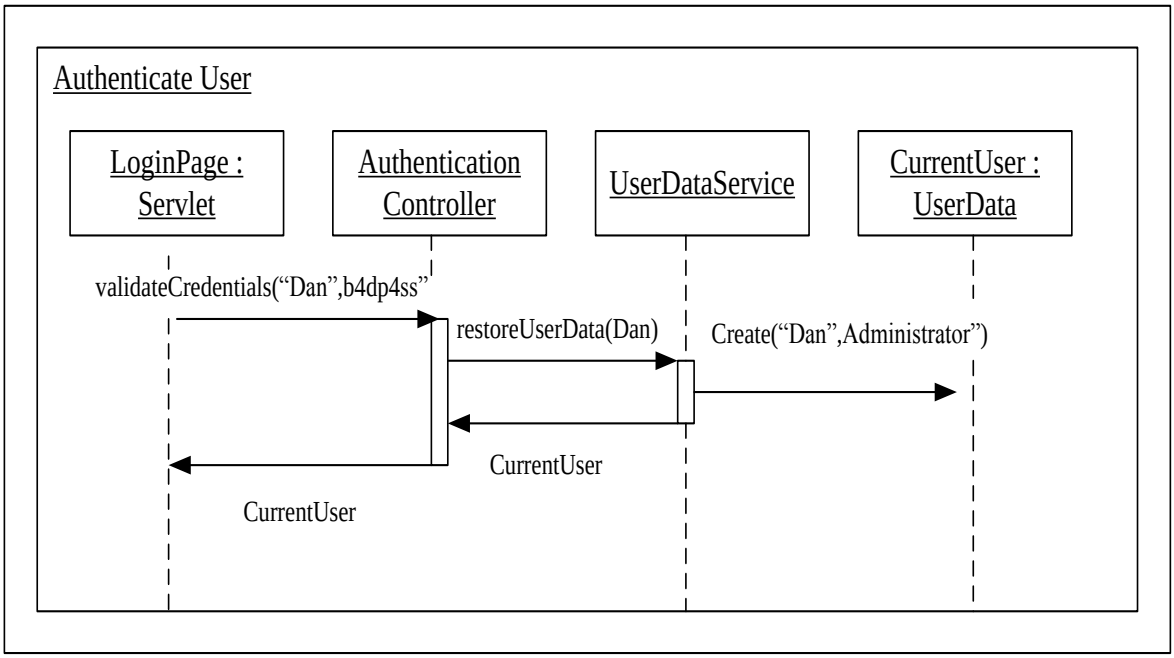
Detail aktivitas dapat dimasukkan di dalam kotak. Akan diperlihatkan dengan symbol yang sama dengan aktivitas dan namanya diletakkan didalam persegi panjang.



Gambar II.7. Aktivitas dengan detail sederhana
Sumber :Prabowo Pudjo Widodo, Herlawati (2011 : 145)

d. Sequence Diagram

Menurut (Douglas, 2004 : 174) menyebutkan ada tiga diagram primer UML dalam memodelkan scenario interaksi, yaitu: diagram urutan (*sequence diagram),* diagram waktu (*timing diagram*) dan diagram komunikasi (*communication diagram*). Namun demikian (Pilone, 2005 : 174) menyatakan bahwa diagram yang paling banyak dipakai adalah diagram urutan. Gambar II.6. memperlihatkan contoh diagram urutan dengan notasi-notasinya yang akan dijelaskan nanti.



Gambar II.8. Contoh diagram urutan

Sumber : Prabowo Pudjo Widodo, Herlawati (2011 : 175)

II.6. ERD (*Entity Relationship Diagram*)

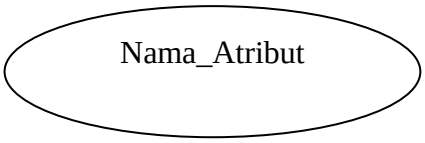
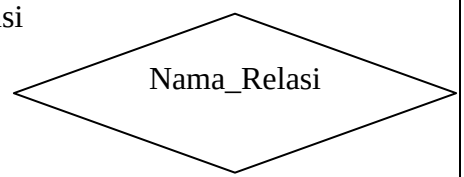
ERD (*Entity Relationship Diagram*) data model didasarkan pada persepsi terhadap dunia nyata yang tersusun atas kumpulan objek-objek dasar yang disebut entitas dan hubungan antar objek. Entitas adalah sesuatu atau objek dalam dunia nyata yang dapat dibedakan dari objek lain.

Sebagai tambahan, model ER menyajikan pula batasan dimana isi basis data harus menyesuaikan dengan batasan. Salah satu batasan yang penting adalah pemetaan kardinalitas (*mapping cardinalities*), yang menggambarkan jumlah entitas yang berhubungan dengan entitas lain melalui suatu relasi. (Janner Si ; 2010 : 60-61)

II.6.1. Simbol-simbol ERD (*Entity Relationship Diagram*)

Adapun symbol-simbol ERD (*Entity Relationship Diagram*) ditunjukkan pada table II.3.

Tabel II.3. Simbol-simbol ERD (*Entity Relationship Diagram*)

Simbol	Deskripsi
Entitas / *entity* 	Entitas merupakan data inti yang akan disimpan; bakal table pada basis data
Atribut 	Fiel atau kolom data yang perlu disimpan dalam suatu entitas
Relasi 	Relasi yang menghubungkan antara entitas; relasi biasanya diawali dengan kata kerja
Asosiasi / *Association* **1** **0..***	Penghubung antar relasi dan entitas dimana di kedua ujungnya memiliki *multiplicity* kemungkinan jumlah

	pemakaian

Sumber : (Janner Si ; 2010 : 59-60)

II.7. Normalisai

Normalisasi adalah bagian perancangan basisdata. Tanpa Normalisasi, sistem basisdata menjadi tidak akurat, lambat, tidak efisien, serta tidak memberikan data yang diharapkan.

Pada waktu menormalisasi basisdata, ada empat tujuan yang harus dicapai, yaitu:

1. Mengatur data dalam kelompok-kelompok sehingga masing-masing kelompok hanya menangani bagian kecil sistem.
2. Meminimal jumlah data berulang dalam basis data.
3. Membuat basisdata yang datanya diakses dan manipulasi secara cepat dan efisien tanpa melupakan integritas data.
4. Mengatur data sedemikian rupa sehingga ketika memodifikasi data, hanya mengubah satu tempat.

Tujuan Normalisai adalah membuat kumpulan table relasional yang bebas dari data berulang dan dapat dimodifikasi secara benar dan konsisten. Ini berarti bahwa semua table pada basisdata relasional harus berada pada bentuk normal ketiga (3NF). Sebuah table relasional berada pada 3NF jika dan hanya jika semua kolom bukan kunci adalah saling independen berarti bahwa tidak ada kolom bukan kunci yang tergantung pada sembarang kombinasi kolom lainnya. Dua

bentuk normal pertama aadalah langkah antara untuk mencapai tujuan, yaitu mempunyai semua table dalam 3NF. (Janner Si; 2010 : 77-78)

Tahapan normalisai terdiri dari beberapa bentuk yaitu sebagai berikut:

1. Bentuk normal pertama (1NF / *First Normal Form*)

Bentuk normal pertama memiliki ciri yaitu data berbentuk *flat file* (file datar), *record* disusun sesuai kedatangan, masih mungkin terjadi penyimpangan data (*anomaly* data). *Anomali* data dapat berupa *insert anomaly*, *delete anomaly*, *update anomaly* dan *redundancy* data (data duplikat). (Janner Si; 2010 : 79)

2. Bentuk Normal Kedua (2NF/ *Second Normal Form*)

Bentuk normal kedua memiliki ciri yaitu tidak terjadi *anomali* data, setiap *field / attribute* bukan kunci harus tergantung fungsi (*functional dependency*) terhadap *field / attribute* kunci, masih mungkin terjadi *transitive dependency* (*field* bukan kunci tergantung pada *field* bukan kunci dalam satu table). (Janner Si; 2010 : 81)

3. Bentuk Normal Ketiga (3NF / *Third Normal Form*)

Bentuk normal ketiga memiliki syarat yaitu table harus tidak terdapat transitive dependency (*field* bukan kunci tergantung pada *field* bukan kunci dalam satu tabel). (Janner Si; 2010 : 82)

4. Bentuk Normal *Boyce Codd* (BCNF / *Boyce Codd Normal Form*)

Pada tahap ini menghilangkan ketergantungan *field* bukan kunci secara persial (bagian) kunci dalam satu tabel. Apabila pada normal ketiga tidak lagi ditemukan *field* bukan kunci tergantung secara persial dalam satu tabel, maka normal ketiga juga merupakan bentuk BCNF. (Janner Si; 2010 : 84)

5. Bentuk Normal Kelima (5NF / *Five Normal Form*)

Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan *(join dependence*). Ketergantungan gabungan berarti bahwa sebuah tabel., setelah didekomposisi menjadi tiga atau lebih tabel yang lebih kecil, harus dapat digabungkan kembali untuk membentuk tabel asal. Dengan kata lain, 5NF menunjukkan ketika sebuah tabel tidak dapat didekomposisi lagi. (Janner Si ; 2010 : 85)

II.8. *Dreamweaver*

Dreamweaver MX (MX 6, MX 7, MX 2004 dan MX 8) adalah suatu bentuk program editor web yang dibuat oleh Macromedia dengan alamat Web site www.macromedia.com. Dengan menggunakan program ini, seorang programmer web dapat dengan mudah membuat dan mendesain webnya, karena bersifat WYSIWYG (*What You Seee Is What You Get*). (Bunafit Nugroho ; 2009 : 1)

Dreamweaver MX dan 8 selain sebagai editor yang komplet juga dapat digunakan untuk membuat animasi sederhana yang berbentuk layer dengan bantuan JavaScript yang didukungnya. Dengan adanya program ini kita tidak akan susah-susah untuk mengetik skrip-skrip format HTML, PHP, JSP, ASP, JavaScript, CSS maupun bentuk program yang lainnya.

Sebagai editor, *Dreamweaver* mempunyai sifat yang WYSIWYG Dibaca (wai-si-wig) yang artinya apa yang kita lihat pada halaman desain, maka

semuanya itu akan Kita peroleh pada *browser*. Dengan kelebihan ini sehingga seorang *programmer* (pembuat program) atau *desainer* (pembuat desain web) dapat langsung melihat hasil buatannya tanpa harus membukanya pada *browser* (aplikasi pengakses web seperti Internet Explorer, Mozila, dll). (Bunafit Nugroho ; 2009 : 2)