BAB II

LANDASAN TEORI

2.1 Data Mining

Data Mining adalah suatu istilah yang digunakan untuk menguraikan penemuan pengetahuan di dalam basis data. *Data Mining* adalah proses yang menggunakan teknik statistik, matematika, kecerdasan buatan, *machine learning* untuk mengekstraksi dan mengidentifikasi informasi yang bermanfaat dan pengetahuan yang terkait dari berbagai basis data besar (Kusrini & Emha Taufiq Luthfi, 2009:25).

Data Mining adalah salah satu bidang yang berkembang pesat karenabesarnya kebutuhan akan nilai tambah dari *database* skala besar yang makin banyak terakumulasi sejalan dengan pertumbuhan teknologi informasi. (Iko Pramudiono, 2003:1).*Data mining*, sering juga disebut *Knowledge Discovery in Database* (KDD),adalah kegiatan yang meliputi pengumpulan, pemakaian data historis untuk menemukan keteraturan, pola atau hubungan dalam set data berukuran besar. Keluaran dari *data mining* ini bisa dipakai untuk memperbaiki pengambilan keputusan di masa depan. Sehingga istilah *pattern recognition* jarang digunakan karena termasuk bagian dari *data mining* (Azevedo, A. Santos & Manuel F,2008:30).

Menurut Gartner Group, *data mining* adalah suatu proses menemukan hubungan yang berarti, pola, dan kecenderungan dengan memeriksa dalam sekumpulan besar data yang tersimpan dalam penyimpanan dengan menggunakan teknik pengenalan pola seperti teknik statistik dan matematika (Larose, 2005).

2.1.1 Teknik Data Mining Klasifikasi

Teknik Klasifikasi dalam *data mining* dikelompokkan ke dalam Teknik Pohon Keputusan, Bayesian (Naïve Bayesian dan Bayesian Belief Networks), Jaringan Saraf Tiruan (*Backpropagation*), Teknik yang berbasis konsep dari penambangan aturan-aturan asosiasi, dan teknik lain (K-Nearest Neighboor, algoritma genetik, teknik dengan pendekatan himpunan *rough* dan *fuzzy*)

Setiap teknik memiliki kelebihan dan kekurangannya sendiri. Data dengan profil tertentu mungkin paling optimal jika diklasifikasi dengan teknik tertentu, atau dengan kata lain, profil data tertentu dapat mendukung termanfaatkannya kelebihan dari teknik ini.

Secara umum, Proses Klasifikasi dapat dilakukan dalam dua tahap, yaitu proses belajar dari *data training* dan klasifikasi kasus. Pada proses belajar, Algoritma Klasifikasi mengolah data training untuk menghasilkan sebuah model. Setelah model diuji dan dapat diterima, pada tahap klasifikasi, model tersebut digunakan untuk memprediksi kelas dari kasus baru untuk membantu proses pengambilan keputusan (Han dkk.,2012).

2.1.2 Pohon Keputusan

Pohon Keputusan atau *Decision Tree* merupakan representasi sederhana dari teknik klasifikasi untuk sejumlah kelas berhingga, dimana simpul internal maupun simpul akar ditandai dengan nama atribut, rusuk-rusuknnya diberi label nilai atribut yang mungkin dan simpul daun ditandai dengan kelas-kelas yang berbeda.(Fajar Astuti, 2013).

2.1.3 Algoritma C4.5

Secara umum algorima C4.5 untuk membangun pohon keputusan adalah sebagai berikut :

1. Pilih atribut sebagai *node* akar.

2. Buat cabang untuk tiap-tiap nilai.

3. Bagi kasus dalam cabang.

4. Ulangi proses untuk setiap cabang sampai semua kasus pada cabang memiliki kelas yang sama (Kusrini & Luthfi, 2011)

Untuk memilih atribut sebagai *node* akar, didasarkan pada nilai Gain tertinggi dari atribut-atribut yang ada. Untuk menghitung Gain digunakan rumus seperti tertera dalam persamaan berikut :

$$Gain(S, A) = Entrotopy(S) - \sum_{i=1}^{n} * Entropy(Si)$$

Keterangan :

S : himpunan kasus

A: Atribut

n : jumlah partisi atribut

|Si| : jumlah kasus pada partisi ke –i

|S| : jumlah kasus dalam S

Setelah mendapatkan nilai Gain, ada satu hal lagi yang perlu kita lakukan perhitungan, yaitu mencari nilai Entropy. Entropy digunakan untuk menentukan seberapa informatif sebuah masukan atribut untuk menghasilkan keluaran atribut. Rumus dasar dari Entropy tersebut adalah sebagai berikut :

$$Entropy(S) = \sum_{i-1}^{n} - pi * \log 2\, pi$$

Keterangan :

S : Himpunan kasus

A : Fitur

n : Jumlah partisi S

pi : Proporsi dari Si terhadap S

Untuk memudahkan penjelasan mengenai algoritma C4.5, berikut ini disertakan contoh kasus yang dituangkan dalam Tabel 2.1.

Tabel 2.1 Keputusan Bermain Tenis (Kusrini & Luthfi, 2011)

NO	OUTLOOK	TEMPERATUR	HUMIDITY	WINDY	PLAY
1.	Sunny	Hot	High	False	No
2.	Sunny	Hot	High	True	No
3.	Cloudy	Hot	High	False	Yes
4.	Rainy	Mid	High	False	Yes
5.	Rainy	Cool	Normal	False	Yes
6.	Rainy	Cool	Normal	True	Yes
7.	Cloudy	Cool	Normal	True	Yes
8.	Sunny	Mild	High	False	Yes
9.	Sunny	Cool	Normal	False	Yes
10.	Rainy	Mild	Normal	False	Yes
11.	Sunny	Mild	Normal	True	No
12.	Cloudy	Mild	High	True	Yes
13.	Cloudy	Hot	Normal	False	Yes
14.	Rainy	Mild	High	True	No

Dalam kasus yang tertera pada Tabel 2.1 akan dibuat pohon keputusan untuk menentukan main tenis atau tidak dengan melihat keadaan cuaca, temparatur, kelembaban dan keadaan angin.

Selanjutnya data tersebut akan diproses sesuai langkah-langkah membentuk pohon keputusan. Berikut ini adalah penjelasan lebih terperinci:

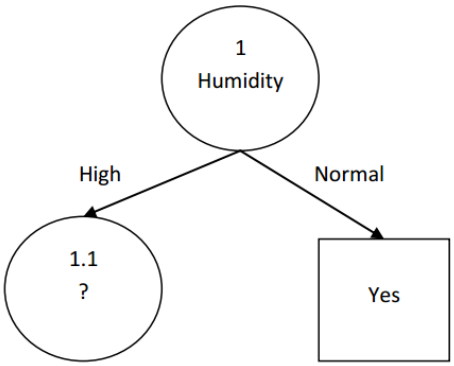
Hitung jumlah kasus, yakni jumlah kasus untuk keputusan Yes dan jumlah keputusan No, dan Entropy dari semua kasus dan kasus yang dibagi berdasarkan atribut-atribut yang digunakan. Setelah itu lakukan perhitungan Gain untuk setiap atribut. Hasil perhitungan dapat dilihat pada Tabel 2.2.

Tabel 2.2 Perhitungan Node 1 (Kusrini & Luthfi, 2011)

Node			Jumlah Kasus	No	Yes	Entropy	Gain
1	Total		14	4	10	0.863	
	Outlook						0.258
		Cloudy	4	0	4	0	
		Rainy	5	1	4	0.721	
		Sunny	5	3	2	0.97	
	Temperature						0.183
		Cool	4	0	4	0	
		Hot	4	2	2	1	
		Mild	6	2	4	0.918	
							0.370
	Humidity						
		High	7	4	3	0.985	
		Normal	7	0	7	0	
	Windy						0.005
		False	8	2	6	0.811	
		True	6	4	2	0.918	

Dari hasil pada Tabel 2.2 dapat diketahui bahwa atribut dengan Gain tertinggi adalah Humidity yaitu sebesar 0.37. Dengan demikian Humidity dapat menjadi node akar. Ada dua nilai atribut dari Huminity yaitu High dan Normal. Dari kedua nilai atribut tersebut, nilai atribut Normal sudah mengklasifikasikan kasus menjadi satu keputusan Yes, sehingga tidak perlu dilakukan perhitungan lenih lanjut, tetapi untuk nilai atribut High masih perlu dilakukan perhitungan lagi.

Setelah dilakukan perhitungan , maka terbentuklah pohon keputusan sementara seperti pada Gambar 2.1.



Gambar 2.1 Pohon Keputusan Hasil Perhitungan Node 1(Kusrini & Luthfi, 2011)

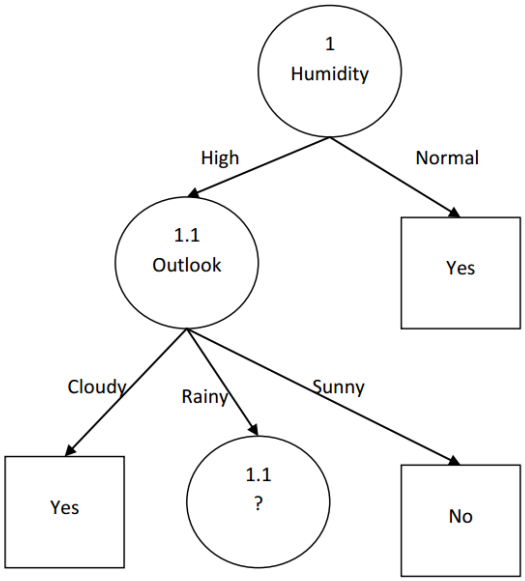
Hitung kembali jumlah kasus, Entropy dari semua kasus yang dibagi berdasarkan atribut yang dapat menjadi *node* akar dari nilai atribut Humidity-High. Setelah itu lakukan perhitungan Gain untuk masing-masing atribut. Hasil perhitungan ditunjukkan pada Tabel 2.3.

Tabel 2.3 Perhitungan Node 1.1 (Kusrini & Luthfi, 2011)

Node			Jumlah Kasus	No	Yes	Entropy	Gain
1.1	Humidity-Gain		7	4	3	0.965	
	Outlook						0.699
		Cloudy	2	0	2	0	
		Rainy	2	2	1	1	
		Sunny	3	3	0	0	
	Temperature						0.02
		Cool	0	0	0	0	
		Hot	3	2	1	0.918	
		Mild	4	2	2	1	
	Windy						0.02
		False	4	2	2	1	
		True	3	2	1	0.918	

Dari Tabel 2.3 dapat diketahui bahwa atribut dengan Gain tertinggi adalah Outlook yaitu sebesar 0,67. Dengan demikian Outlook dapat menjadi node cabang dari nilai atribut Humidiy –

High . Dari ketiga nilai atribut Outlook, nilai atribut Cloudy sudah mengklasifikasi kasus 1 yaitu keputusan Yes dan nilai atribut Sunny sudah mengklasifikasi kasus 1 yaitu keputusan No. Pohon keputusan yang dihasilkan sampai tahap ini ditunjukkan pada Gambar 2.2.



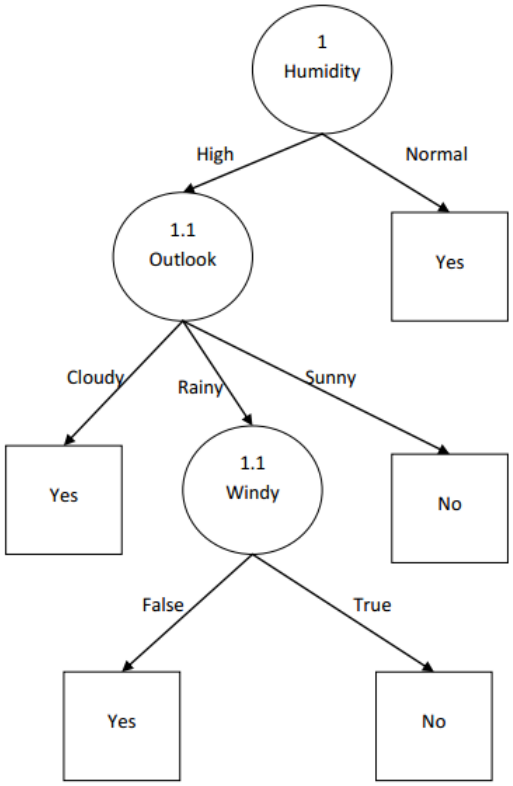
Gambar 2.2 Pohon Keputusan Hasil Perhitungan Node 1.1 (Kusrini & Luthfi, 2011)

Gambar 2.2 menunjukkan hasil perhitungan node 1.1 dimana nilai atribut Outlook yang belum mengklasifikasi kasus menjadi 1 yaitu Rainy. Dengan demikian nilai atribut Outlook – Rainy menjadi node akar untuk perhitungan Node 1.1.2. Hitung jumlah kasus, jumlah kasus untuk keputusan Yes, jumlah kasus untuk keputusan No, dan Entropy dari semua kasus dan kasus yang dibagi berdasarkan atribut Temperature dan Windy yang dapat menjadi node cabang dari nilai atribut Outlook, yakni Rainy. Setelah itu hitung nilai Gain untuk masing-masing atribut. Hasil perhitungan ditunjukkan pada Tabel 2.4.

Tabel 2.4 Perhitungan Node 1.1.2 (Kusrini & Luthfi, 2011)

Node			Jumlah Kasus	No	Yes	Entropy	Gain
1.1.2	Humidity-High DAN Outlook-Rainy		2	1	1	1	
	Temperature						0
		Cool	0	0	0	0	
		Hot	0	0	0	0	
		Mild	2	1	1	1	
	Windy						1
		False	1	0	1	0	
		True	1	1	0	0	

Dari hasil pada Tabel 2.3 dapat diketahui bahwa atribut dengan Gain tertinggi adalah Windy dengan nilai 1. Dengan demikian Windy dapat menjadi node cabang dari nilai atribut Rainy. Ada dua nilai atribut dari Windy yaitu True dan False. Dari kedua nilai atribut tersebut, nilai atribut False sudah mengklasifikasi kasus menjadi 1 yaitu keputusan Yes dan nilai atribut True sudah mengklasifikasikan kasus menjadi satu dengan keputusan No, sehingga tidak perlu lagi perhitungan lebih lanjut untuk nilai atribut ini. Pohon keputusan yang terbentuk sampai tahap ini ditunjukkan pada Gambar 2.3.



Gambar 2.3 Pohon Keputusan Hasil Perhitugan Node 1.1.2 (Kusrini & Luthfi, 2011)

Dengan memperhatikan pohon keputusan pada Gambar 2.3, diketahui bahwa semua kasus sudah masuk dalam kelas. Dengan demikian, pohon keputusan pada Gambar 2.3 merupakan pohon keputusan terakhir yang terbentuk.

2.2 Definisi Sistem

Menurut Bambang Hariyanto (2008) Sistem adalah kumpulan elemen yang saling berinteraksi untuk mencapai satu tujuan tertentu.

Beberapa prinsip umum sistem adalah sebgai berikut :

1. Sistem selalu merupakan bagian sistem lebih besar. Sistem dapat dipartisi menjadi subsistem-subsistem yang lebih kecil.
2. Sistem lebih terspesialisasi akan kurang dapat beradaptasi untuk menghadapi keadaan-keadaan berbeda.

3. Lebih besar ukuran sistem maka akan memerlukan lebih banyak sumber daya untuk operasi dan pemeliharaan.
4. Sistem senantiasa mengalami perubahan, tumbuh dan berkembang.

2.3 Perangkat Lunak

Menurut Rosa dan Salahuddin (2011), Perangkat lunak adalah program komputer yang terasosiasi dengan dokumentasi perangkat lunak seperti dokumentasi kebutuhan model desain, dan cara penggunaan. Sebuah program komputer tanpa terasosiasi dengan dokumentasinya maka belum dapat disebut perangkat lunak. Sebuah perangkat lunak juga sering disebut dengan sistem perangkat lunak, sehingga dapat diartikan sebuah sistem yang memiliki hubungan satu sama lain untuk memenuhi program pelanggan. Pelanggan adalah orang atau organisasi yang membeli atau memesan perangkat lunak dari pengembang perangkat lunak. Sedangkan *user* atau pengguna perangkat lunak adalah orang yang memiliki kepentingan utuk memakai atau menggunakan perangkat lunak untuk memudahkan pekerjaannya.

2.4 Internet

2.4.1 Sejarah Internet

Asal usul internet berasal dari jaringan komputer yang dibentuk pada tahun 1970-an. Jaringan komputer tersebut disebut dengan ARPAnet, (*US Defense Advanced Research Projects Agency*). yaitu jaringan komputer yang dibentuk oleh departement pertahanan Amerika Serikat. ARPAnet dibangun dengan sasaran untuk membuat suatu jaringan komputer yang tersebar untuk menghindari pemusatan informasi di satu titik yang dipandang rawan untuk dihancurkan apabila terjadi peperangan. Dengan cara ini diharapkan apabila satu bagian dari jaringan terputus, maka jalur yang melalui jaringan tersebut dapat secara otomatis dipindahkan ke saluran lainnya. Selanjutnya, jaringan komputer tersebut diperbarui dan dikembangkan, dan sekarang penerusnya

menjadi tulang punggung global untuk sumber daya informasi yang disebut dengan internet. (Irwansyah Edy, 2014).

2.4.2 Web

Perkembangan dunia internet telah melahirkan suatu fasilitas layan baru, yaitu web, yang merupaka layanan terpenting internet. Dewasa ini, fasilitas web mengizinkan pengakses untul mengakses dan berinteraksi dengan teks, grafik, animasi, foto, suara dan video. Web secara fisik adalah kumpulan komputer pribadi, web browser, koneksi ke ISP, komputer server, router, dan switch yang digunakan untuk mengalirkan informasi dan menjadi wahana pertama sebagai pihak terkait. Web dibagi menjadi beberapa jenis, yaitu:

a. **Web Search Engine:** adalah web yang memiliki kemampuan untuk melakukan pencarian dokumen berdasarkankata kunci tertentu. Contoh: Google dan Yahoo
b. **Web Portal:** adalah web yang berisi kumpulan link, search engine, dan informasi. Contoh: Yahoo dan AOL.
c. **Web Perusahaan:** adalah web yang mendeskripsikan suatu perusahaan, layanan fasilitas, dan segal sesuatu tentang perusahaan. Contoh: indosat.
d. **Web Pribadi:** adalah web yang memberikan profil pemilik web.

Web dikenal dengan sistem client-server. Komputer pengguna disebut komputer client, sedangka komputer yang di akses disebut server. Web menggunakan protokol yang disebut HTTP (*HyperText Transfer Protocol*) yang berjalan pada TCP/IP. Perekat antar halaman web disebut *Hyperlink* dan *Hypertext.* Agar dapat menjelajahi web, kita membutuhkan perangkat lunak yang disebut web browser. Kemudian, membangun sebuah web ditulis pada bahasa komputer yang dikenal *Hypertext Markup Languange (*HTML).

Pada prakteknya, bahasa komputer berbasis web tidak hanya HTML, melainkan juga melibatkan bahasa pemrograman lain seperti PHP atau Perl. Tujuannya adalah untuk membentuk halaman yang bersifat dinamis. Perlu diketahui, aplikasi web itu sendiri dibagi menjadi Web statis dan Web Dinamis.

Web statis dibentuk dengan menggunakan HTML saja. Kekurangan aplikasi seperti ini terletak pada keharusan untuk memelihara program secara terus-menerus untuk mengikuti setiap perubahan yang terjadi. Kelemahan ini diatasi dengan model aplikasi web dinamis.

Dengan memperluas kemampuan HTML, yakni dengan menggunakan perangkat lunak tambahan, perubahan informasi dalam halaman-halaman web dapat ditangani melalui perubahan data, bukan melalui perubahan program. Sebagai implementasinya, aplikasi dapat dikoneksikan ke basis data. Dengan demikian, perubahan informasi dapat dilakukan olehoperator atau yang bertanggung jawab terhadap kemuktakiran data, dan tidak menjadi tanggung jawab pemrograman atau Webmaster. Pemrograman aplikasi web terdiri dari :

2.4.3 HTML

HTML kependekan dari *Hyper Text Markup Languange*. Dokumen HTML adalah file teks murni yang dapat di buat dengan editor teks sembarang. Dokumen

ini dikenal sebagai *web page*. Dokumen HTML merupakan dokumen yang disajikan dalam browser *web surfer*. Dokumen ini umumnya berisi informasi atau *interface* aplikasi di dalam internet. Ada dua cara untuk membuat sebuah *web page*: dengan HTML *editor* atau dengan *editor* teks biasa (misalnya notepad).

Dokumen HTML disusun oleh elemen-elemen. “Elemen” merupakan istilah bagi komponen-komponen dasar pembentuk dokumen HTML. Beberapa contoh elemen adalah: head,

body, table, paragraf, dan list. Elemen dapat berupa teks murni, atau bukan teks, atau keduanya.(Abdul Kadir, 2008)

2.4.4 CSS

CSS merupakan singkatan dari *Cascading Style Sheet*, merupakan fitur baru dari HTML 4.0. hal ini diperlukan setelah melihat perkembangan HTML menjadi kurang praktis karena *web pages* terlalu banyak dibebani hal-hal yag berkaitan dengan faktor tampilan seperti *font* dan lain-lain. Untuk itu jika kumpulan isi gaya (*style*) tersebut dikelolah secara terpisah maka manajemen *pages* menjadi lebih mudah dan efisien. (Bambang Hariyanto, 2008)

2.4.5 PHP

Menurut Budi Raharjo (2011), PHP merupakan salah satu bahasa pemrograman berbentuk skrip yang ditempatkan dalam server dan diproses di server. Hasilnyalah yang dikirm ke klien, tempat pemakai menggunakan *browser*. Secara khusus, PHP dirancang untuk membentuk aplikasi web dinamis. Maksudnya, PHP mampu menghasilkana *website* yang secara terus-menerus hasilnya bisa berubah-ubah sesuai dengan pola yang diberikan. Hal tersebut tergantung pada permintaan *client browser*-nya (bisa menggunakan *browser* Opera, Internet Explorer, Mozzila,, dan lain-lain). Umumnya, pembuatan web dinamis berhubungan erat dengan *Databse* sebagai sumber data.

PHP mempunyai fungsi yang sama dengan skrip-skrip seperti ASP *(Active Server Page)*, Cold Fusion, ataupun Perl. Namu, Perlu diketahui bahwa PHP sebenarnya bisa dipakai secara *Command Line*. Artinya, skrip PHP dapat dijalankan tanpa melibatkan *web server* maupun *browser.*

Model kerja HTML diawali dengan permintaan suatu halaman web oleh *browser.* Berdasarkan URL (*Uniform Resource Locator*) atau dikenal dengan sebutan alamat internet,

browser mendaptkan alamat dari *web browser*, mengidentifikasi alamat yang dikehendaki, dan menyampaikan segala informasi yang dibutuhkan oleh *web server*. Selanjutnya, *web server* akan mencarikan *file* yang diminta dan memberikan isinya ke *web browser* (atau yang biasa disebut browser saja). *Browser* yang mendapatkan isinya segera melakukan proses penerjemah kode HTML dan menampilkan ke layar pemakai.

Bagaimana halnya yang diminta adalah sebuah halaman PHP? Prinsipnya serupa dengan kode HTML. Hanya saja, ketika berkas PHP yang diminta didapatkan oleh web server, isinya segera dikirimkan ke mesin PHP dan mesininilah yang memproses dan memberikan hasilnya (berupa kode HTML) ke *Web Server.* Selanjutnya, *web server* menyampaikan ke klien.

2.5 Basis Data MySQL

Menutut Bambang Hariyanto (2008), data adalah rekaman mengenai fenomena/fakta yang ada atau yang terjadi. Data pada pokonya adalah refleksi fakta yang ada. Data mengenai fakta-fakta penting organisasi harus direkam dan dikelola secara baik sehingga dapat dipakai/diakses secara efisien sehingga efektif mendukung operasi dan pengendalian organisasi. Data merupakan sumber daya penting pada manajemen modern. Untuk itu, organisasi perlu melakukan penataan dan manajemen data yang baik agara data yang dimiliki organisasi dapat berdaya guna secara maksimal.

Basisdata adalah kumpulan data (elementer) yang secara logic berkaitan dalam merepresentasikan fenomena/fakta secara terstruktur dalam domain tertentu untuk mendukung aplikasi pada sistem tertentu. Basisdata adalah kumpulan data yang saling berhubungan yang merefleksikan fakta-fakta yang terdapat di organisasi. Basisdata mendeskripsikan *state* organisasi/perusahaan/sistem. Saat satu kejadian muncul si dunia nyata mengubah *state* organisasi/perusahaan/sistem maka satu perubahan pun harus dilakukan terhdadap data yang

disimpan di basisdata. Basisdata merupakan komponen utama sistem informasi karena semua informasi untuk pengambilan keputusan berasal dari data di basisdata. Pengelolaaan basisdata yang buruk dapat mengakibatkan ketidaktersediaan data penting yang digunakan untuk menghasilkan informasi yang diperlukan dalam pengambilan keputusan.

Sistem manajemen basisdata atau DBMS (*Database Management System*) adalah perangkat lunak untuk mendefinisikan, menciptakan, mengelola, dan mengendalikan pengsaksesan basisdata. Fungsi sistem manajemen basisdata saat ini yang penting adalah menyediakan basis untuk sistem informasi manjemen.

MySQL adalah salah satu jenis *database server* yang sangat terkenal kepopulerannya disebabkan MySQL menggunakan SQL sebagai bahasa dasar untuk mengakses *database*nya. Selain itu ia bersifat *open source* (tidak perlu membayar untuk menggunakannya) pada pelbagai platform (kecuali untuk jenis Enterprise, yang bersifat komersial).

MySQL termasuk jenis RDBMS (*Relational Database Management System*). Itulah sebabnya, istilah seperti tabel, baris, dan kolom digunakan pada MySQL. Pada MySQL, sebuah *database* mengandung satu atau beberapa kolom. (Bambang Hariyanto, 2008)

2.6 UML

UML itu singkatan dari Unified Modelling Language. Sesuai dengan kata terakhir dari kepanjangannya, UML merupakan salah satu bentuk language atau bahasa. Menurut pencetusnya, UML didefinisikan sebagai bahasa visual untuk menjelaskan, memberikan spesifikasi, merancang, membuat model, dan mendokumentasikan aspek-aspek dari sebuah sistem.

Karena tergolong bahasa visual, UML lebih mengedepankan penggunaan diagram untuk menggambarkan aspek dari sistem yang sedang dimodelkan. Memahami UML itu sebagai bahasa visual itu penting, karena penekanan tersebut membedakannya dengan bahasa pemrograman yang

lebih dekat ke mesin. Bahasa visual lebih dekat ke mental model pikiran kita, sehingga pemodelan menggunakan bahasa visual bisa lebih mudah dan lebih cepat dipahami dibandingkan apabila dituliskan dalam sebuah bahasa pemrograman.

UML adalah salah satu bentuk notasi atau bahasa yang sama yang digunakan oleh professional dibidang software untuk menggambarkan atau memodelkan sebuah system software. Sebelumnya ada banyak notasi atau bahasa lain untuk mencapai keperluan yang sama misalnya DFD (Data Flow Diagram). Tetapi sejak matang dan populernya teknologi pemrograman, perancangan, dan analisis berorientasi objek, UML telah menjadi *de facto standard language*.

Ada tiga cara dalam memakai UML dalam melakukan pemodelan system:

1. UML sebagai sketsa

UML digambarkan dalam sketsa coretan-coretan dalam kertas atau whitboard secara tidak formal. Biasanya digunakan dalam sesi diskusi tim untuk membahas aspek tertentu dalam tahap analisis dan perancangan.

2. UML sebagai *blueprint system*

Seperti diagram kelistrikan adalah blueprint dari komponen atau produk yang akan dihasilkan, UML juga bisa menggambarkan blueprint yang identik untuk sebuah system software.

3. UML sebagai bahasa pemrograman

UML berfungsi sebagai bahasa pemrograman mencoba melakukan semuanya dengan UML sampai kepada produk jadinya. Analisis dan perancangan dilakukan dengan diagram-diagram yang ada dalam UML, sementara sebuah tool atau generator bisa menghasilkan produk akhir dari diagram-diagram ini. Diagram-diagram yang terdapat dalam UML antara lain:

• *use case diagram*

• *class diagram*

• *statechart diagram*

• *activity diagram*

• *sequence diagram*

• *collaboration diagram*

• *component diagram*

• *deployment diagram* (Munawar, 2005).

2.6.1 Use Case

Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah "apa" yang diperbuat sistem, dan bukan "bagaimana". Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, meng-*create* sebuah daftar belanja, dan sebagainya.

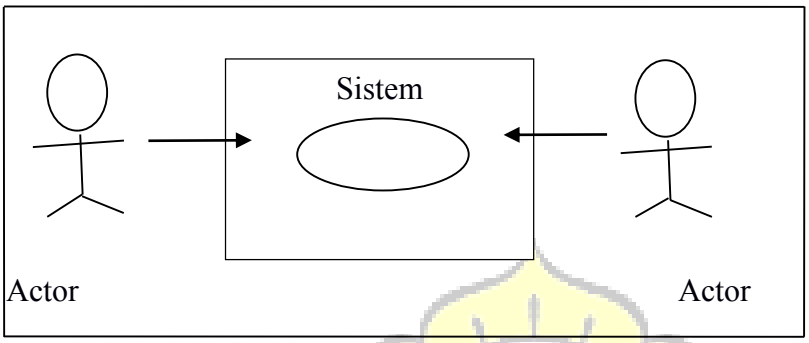
Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu.

Use case diagram dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem.Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya. Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal.

Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extenduse case* lain dengan *behaviour*-nya sendiri. Sementara hubungan

generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Diagram use case menunjukan 3 aspek dari sistem yaitu *Actor*, *use case* dan *system/sub system boundary*. *Actor* mewakili peran orang, *system* yang lain atau alat ketika berkomunikasi dengan *use case*.

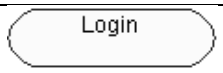
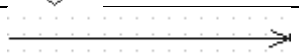


Gambar 2.4 *Use Case Model (Munawar, 2005)*

2.6.2 Activity Diagram

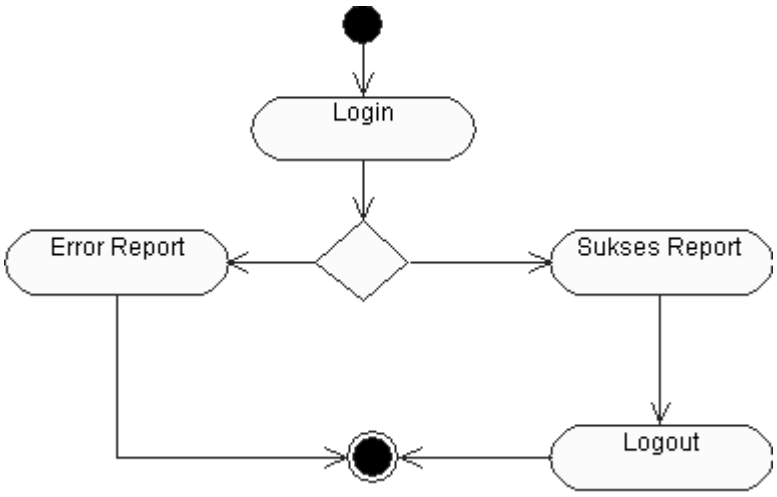
Menurut (Romi Satria Wahono, 2003) Pada dasarnya Diagram aktivitas adalah Diagram *flowchart* yang diperluas yang menunjukkan aliran kendali satu aktivitas ke aktivitas lain. Kegunaan diagram ini adalah untuk memodelkan workflow atau jalur kerja, memodelkan operasi, bagaimana objek-objek bekerja, aksi-aksi dan pengaruh terhadap objek. Simbol-simbol yang terdapat dalam *Activity* Diagram, diantaranya sebagai berikut :

Tabel 2.5 Simbol Activity Diagram (Romi Satrio Wahono, 2003)

Keterangan	**Simbol**
Titik Awal atau permulaan.	
Titik Akhir atau akhir dari aktivitas.	
Aktiviti, atau aktivitas yang dilakukan oleh aktor.	
Decision, atau pilihan untuk mengambil keputusan.	
Arah tanda panah alur proses.	

Activity diagram menunjukkan apa yang terjadi, tetapi tidak menunjukkan siapa yang melakukan apa. Dalam pemprograman hal tersebut tidak menunjukkan *class* mana yang bertanggungjawab atas setiap *action*. Pada pemodelan bisnis, hal tersebut tidak bisa menunjukkan organisasi mana yang menjalankan sebuah *action*. *Swimlane* adalah sebuah cara untuk mengelompokan *activity* berdasarkan *actor* (mengelompokkan *activity* dari sebuah urutan yang sama). *Actor* bisa ditulis nama *actor* ataupun sekaligus dengan lambang *actor* (*stick figure*) pada *usecase diagram*. *Swimlane* digambarkan secara vertikal, walaupun terkadang digambarkan secara horizontal.

Activity diagram merupakan salah satu diagram yang umum digunakan dalam UML untuk menjabarkan proses atau aktivitas dari aktor. Sebagai contoh, pelanggan melakukan *login* (masuk) pada halaman *website* untuk bergabung, jika pelanggan belum terdaftar, maka akan ditolak oleh sistem dan dikembalikan. Proses penjabarannya adalah sebagai berikut :



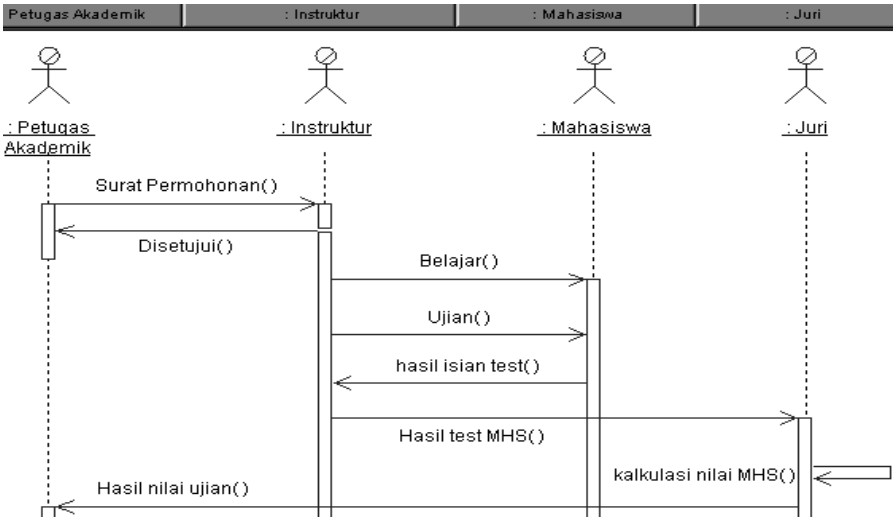
Gambar 2.5 Activity Diagram (Romi Satrio Wahono, 2003)

Di dalam *Activity* diagram tersebut dijelaskan bahwa *user* melakukan proses *login* untuk dapat memasuki area sistem, jika proses *login* dan/atau *user* belum teregistrasi, maka *user* akan ditolak oleh sistem tersebut dan diberi pesan *error*. Selain itu, bila *user* telah teregistrasi dan memasukkan kode *login* dengan benar maka akan diberi akses untuk masuk ke sistem, dan diberikan pesan sukses. *User* dapat *logout* (keluar) untuk mengakhiri sesi.

2.6.3 Sequence Diagram

Menurut Sri Dharwiyanti (2003) *Sequence Diagram* adalah suatu diagram yang digunakan untuk memodelkan skenario penggunaan. Sekenario penggunaan adalah barisan kejadian yang terjadi selama satu kesekusi sistem. *Sequence* diagram digunakan untuk :

1. *Overview* perilaku sistem.
2. Menunjukkan objek-objek yang diperlukan.
3. Mendokumentasikan skenario dari suatu *use-case*
4. Memeriksa jalur-jalur pengaksesan



Gambar 2.6 contoh Sequence Diagram (Sri Dharwiyanti, 2003)

UNIVERSITAS
DARMA PERSADA