BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Sistem

Untuk mengawali pembahasan tentang analisis dan perancangan sistem informasi, pemahaman akan sistem terlebih dahulu harus ditekankan. Definisi sistem berkembang sesuai dengan konteks di mana pengertian sistem itu digunakan. Berikut akan diberikan beberapa definisi sistem secara umum :

1. Kumpulan dari bagian-bagian yang bekerja sama untuk mencapai tujuan yang sama. Contoh :
   a. Sistem Tatasurya
   b. Sistem Pencernaan
   c. Sistem Transportasi Umum
   d. Sistem Otomatif
   e. Sistem Komputer
   f. Sistem Informasi
2. Sekumpulan objek-objek yang saling berelasi dan berinteraksi serta hubungan antar objek bisa dilihat sebagai satu kesatuan yang dirancang untuk mencapai satu tujuan.

Dengan demikian, secara sederhana sistem dapat diartikan sebagai suatu kumpulan atau himpunan dari unsur atau variable-variabel yang saling terogranisasi, saling berinteraksi, dan saling bergantung sama lain. *Murdick* dan *Ross* (1993) mendefinisikan sistem sebagai seperangkat elemen yang digabungkan

satu dengan lainnya untuk satu tujuan bersama. Sementara definisi sistem dalam kamus *Webster's Unbriged* adalah elemen-elemen yang saling berhubungan dan membentuk satu kesatuan atau organisasi (Hanif Al Fatta ; 2007 : 3).

II.2. Pengertian Sistem Informasi

Sistem informasi bukan merupakan hal yang baru, yang baru adalah komputerisasinya. Sebelum ada komputer, teknik penyaluran informasi yang memungkinkan manajer merencanakan serta mengendalikan operasi telah ada. Komputer menambahkan satu atau dua dimensi, seperti kecepatan, ketelitian dan penyediaan data dengan volume yang lebih besar yang memberikan bahan pertimbangan yang lebih banyak untuk mengambil keputusan.

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi untuk dapat menyediakan laporan-laporan yang diperlukan oleh pihak luar tertentu (Tata Sutabri ; 2012 : 38).

II.3. Akuntansi

Menurut Abubakar (2007 : 1) dalam bukunya *Akuntansi Untuk Bisnis Usaha Kecil dan Menengah.* Akuntansi adalah merupakan proses identifikasi, pencatatan dan komunikasi terhadap transaksi ekonomi dari suatu entitas (perusahaan). Jadi secara umum terdapat tiga aktivitas dalam akuntansi yaitu :

1. Aktivitas identifikasi (*Identifying*), dalam aktivitas ini akan dilakukan identifikasi terhadap transaksi yang terjadi dalam suatu entitas (perusahaan),

2. Aktivitas pencatatan (*recording*), dalam aktivitas ini semua transaksi ekonomi atau transaksi keuangan yang telah diidentifikasi pada tahap pertama akan dicatat secara kronologis dan sistematis dengan ukuran nilai moneter tertentu, dan
3. Aktivitas komunikasi (*external user*), misalnya kreditur, investor dan fiskus.

II.4. Sistem Informasi Akuntansi

Menurut Hall (2007 : 10) dalam bukunya *Sistem Informasi Akuntansi*, Sistem Informasi Akuntansi merupakan sistem yang memproses berbagai transaksi keuangan dan transaksi nonkeuangan yang secara langsung mempengaruhi pemrosesan transaksi keuangan.

Sistem Informasi Akuntansi mempunyai tiga subsistem, yaitu:

a. Sistem pemrosesan transaksi yang mendukung operasi bisnis harian melalui berbagai dokumen serta pesan untuk para pengguna di seluruh perusahaan.
b. Sistem buku besar/pelaporan keuangan yang menghasilkan laporan keuangan, seperti laporan laba rugi, neraca, arus kas, dan sebagainya.
c. Sistem pelaporan manajemen yang menyediakan pihak mananjemen internal berbagai laporan keuangan bertujuan khusus serta informasi yang dibutuhkan untuk pengambilan keputusan, seperti anggaran, laporan kinerja, serta laporan pertanggungjawaban.

II.5. Pengertian Java

Java Language Spesification adalah definisi teknis dari bahasa pemrograman *Java* yang di dalamnya terdapat aturan penulisan sintaks dan

semantik *Java*. Referensi lengkap tentang *Java Language Spesification* dapat anda temui pada website resmi *Java*, yaitu http://java.sun.com/docs/books/jl (Wahana Komputer ; 2010 : 3).

Java Memiliki banyak kelebihan, antara lain :

a. Sederhana

   Untuk mempelajari Java tidaklah sulit. Anda bisa mempelajari dan membuat program secara cepat dengan menggunakan Java.

b. Berorientasi Objek

   Meskipun banyak bahasa pemrograman yang menyebut dirinya berorientasi objek, Java bukan merupakan turunan dari bahasa pemrograman manapun dan sama sekali tidak kompatibel dengan source-code bahasa apapun.

c. Aman

   Banyak bahasa yang pada awalnya dirancang dengan tingkat keamanan yang hampir tidak ada. Fasilitas-fasilitas yang diberikan seringkali dapat dimanfaatkan untuk disusupi oleh berbagai virus. Berbeda dengan Java yang disusun sejak awal dengan prinsip keamanan.

d. Interaktif

   Java dirancang memenuhi kebutuhan dunia nyata untuk menciptakan program jaringan yang interaktif.

e. Kokoh

   Java membatasi anda dari berbagai hal kunci supaya dapat menemukan kesalahan lebih cepat saat mengembangkan program.

f. Terinterprestasi dan Berkinerja Tinggi

Java dapat diterjemahkan oleh sistem apapun yang memiliki program Java di dalamnya. Java dirancang dengan hati-hati sehingga mudah diterjemahkan ke dalam bahasa asli suatu sistem untuk menghasilkan kinerja yang tinggi.

g. Terdistribusi

Java dapat dengan mudah bekerja dalam lingkungan terdistribusi seperti internet/intranet karena kemampuannya menangani protokol TCP/IP.

h. Dinamis

Karena berorientasi objek, Java mudah dalam hal pemeliharaan dan pengembangan, karena kita tidak harus membedah isi program untuk mengubah dan mengembangkan program dengan skala lebih besar (Ridwan Sanjaya; 2006 : 2-4).

II.6. Pengertian NetBeans

NetBeans merupakan salah satu proyek *open source* yang disponsori oleh *Sun Microsystem*. Proyek ini bediri pada tahun 2000 dan telah menghasilkan 2 produk, yaitu NetBeanss IDE dan NetBeans Platform. NetBeans IDE merupakan produk yang digunakan untuk melakukan pemrograman baik menulis kode, meng-*compile*, mencari kesalahan dan mendistribusikan program. Sedangkan NetBeans Platform adalah sebuah modul yang merupakan kerangka awal / pondasi dalam bangun aplikasi desktop yang besar.

NetBeans juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi *desktop*), pemrograman enterprise, dan

pemrograman perangkat mobile. Saat ini NetBeans telah mencapai versi 6.8 (Wahana Komputer ; 2010 : 15).

II.7. Pengertian Database

Database adalah sekumpulan *file* data yang saling berhubungan dan diorganisasi sedemikian rupa sehingga memudahkan untuk mendapat dan memproses data. Lingkungan sistem *database* menekankan data yang tidak tergantung (*idenpendent data*) pada aplikasi yang akan menggunakan data. Data adalah kumpulan fakta dasar (mentah) yang terpisah.

Sebuah *database* harus dibuat dengan rapi agar data yang dimasukkan sesuai dengan tempatnya. Sebagai contoh, di sebuah perpustakaan, penyimpanan buku dikelompokkan berdasar jenis atau kategori-kategori tertentu, misalnya kategori buku komputer, buku pertanian, dan lain-lain. Kemudian dikelompokkan lagi berdasarkan abjad judul buku, ini dilakukan agar setiap pengunjung dapat dengan mudah mencari dan mendapatkan buku yang dimaksud (Wahana Komputer ; 2006 ; 1).

II.8. Pengertian MySQL

MySQL pertama kali dirintis oleh seorang programmer database bernama Michael Widenius, yang dapat anda hubungi di emailnya monty@analytikerna.

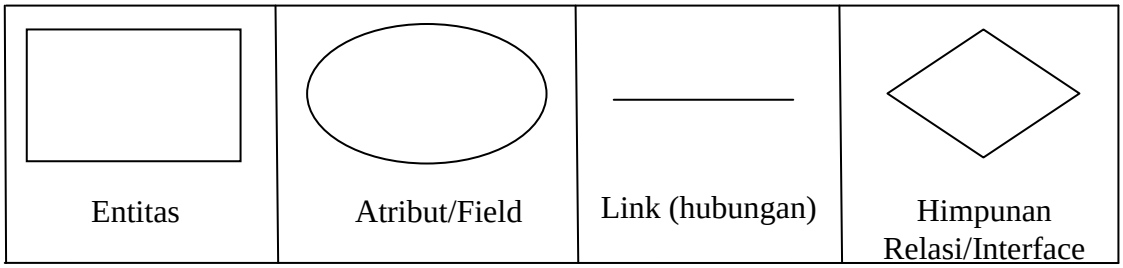
MySQL *database server* adalah RDBMS (*Relasional Database Management System*) yang dapat menangani data yang bervolume besar. meskipun begitu, tidak menuntut *resource* yang besar. MySQL adalah *database* yang paling populer di antara *database* yang lain.

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 *database* ini untuk keperluan pribadi atau usaha tanpa harus membeli atau membayar lisensi, yang berada di bawah lisensi GNU/GPL (*General Public License*), yang dapat anda download pada alamat resminya http://www.mysql.com. MySQL sudah cukup lama dikembangkan, beberapa *fase* penting dalam pengembangan MySQL adalah sebagai berikut :

a. MySQL dirilis pertama kali secara internal pada 23 Mei 1995.
b. Versi *windows* dirilis pada 8 Januari 1998 untuk *windows* 95 dan *windows* NT.
c. Versi 3.23 : beta dari Juni 2000, dan dirilis pada Januari 2001.
d. Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (*unions*) (Wahana ; 2010 : 5).

II.9. *Entity Relationship Diagram* (ERD)

Entity Relationship Diagram atau ERD merupakan salah satu alat (tool) berbentuk grafis yang populer 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).



Gambar. II.1 Bentuk Simbol ERD
(Sumber : Yuniar Supardi ; 2010 : 448)

II.10. 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).

II.11. 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 :

1. Berisi data yang diperlukan.

2. Memiliki sesedikit mungkin redundansi.
3. Mengakomodasi banyak nilai untuk tipe data yang diperlukan.
4. Mengefisienkan update.
5. 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.

II.11.1.Bentuk-bentuk Normalisasi

a. Bentuk 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.

b. Bentuk normal tahap pertama (1" Normal Form)

Definisi :

Sebuah table disebut 1NF jika :

1. Tidak ada baris yang duplikat dalam tabel tersebut.
2. Masing-masing cell bernilai tunggal

Catatan: Permintaan yang menyatakan tidak ada baris yang duplikat dalam sebuah tabel berarti tabel tersebut memiliki sebuah kunci, meskipun kunci tersebut dibuat dari kombinasi lebih dari satu kolom atau bahkan kunci tersebut merupakan kombinasi dari semua kolom.

c. Bentuk 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.

d. Bentuk normal tahap ketiga (3rd normal form)

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

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

e. Bentuk 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).

f. Boyce Code Normal Form (BCNF)

a. Memenuhi 1st NF

b. Relasi harus bergantung fungsi pada atribut superkey (Kusrini ; 2007 : 39-43).

II.12. UML (*Unified Modeling Language*)

Menurut Windu Gata (2013 : 4) 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.

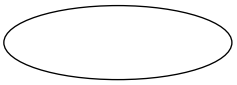
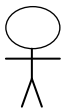
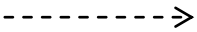
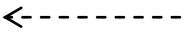
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem.UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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

a. *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.1. 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.

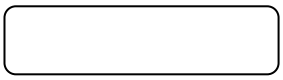
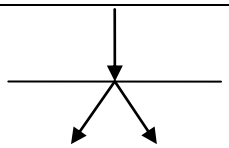
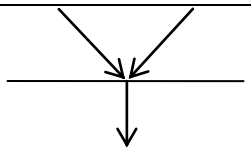
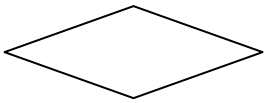
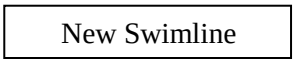
(Sumber : Windu Gata ; 2013 : 4)

b. 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.2. Simbol *Activity Diagram*

Gambar	Keterangan

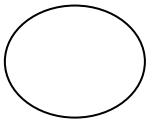
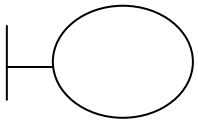
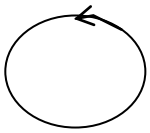
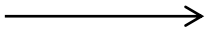
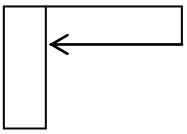
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel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

(Sumber : Windu Gata ; 2013 : 6)

c. 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.3. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass,* 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 : 7)

d. *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.4. *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)

II.13. Daftar Pustaka

Fatta, Hanif Al, 2007. ***Analisis & Perancangan Sistem Informasi Untuk Keunggulan Bersaing Perusahaan & Organisasi Modern***. Andi, Yogyakarta.

Sutabri, Tata, 2012. ***Analisis Sitem Informasi***. Andi, Yogyakarta.

Arif, Abubakar, 2007. ***Akuntansi Untuk Bisnis Usaha Kecil dan Menengah***. Grasindo, Jakarta.

Hall, James A, 2007. ***Accounting Information System***. Salemba Empat, Jakarta.

Komputer, Wahana, 2010. ***Membangun GUI dengan JAVA Netbeans 6.5***. Andi, Yogyakarta.

Sanjanya, Ridwan, 2005. ***Pengolahan Database MySQL dengan Java 2***. Andi, Yogyakarta.

Komputer, Wahana, 2010. ***Membuat Aplikasi facebook dengan Platform NetBeans***. Elex Media Komputindo, Jakarta.

Komputer, Wahana, 2006. ***Seri Panduan Aplikasi Membuat APlikasi Database dengan Java 2***. Andi, Yogyakarta.

Komputer, Wahana, 2010. ***Panduan Belajar MySQL Database Server***. TransMedia, Jakarta.

Supardi, Yuniar, 2010. ***Semua Bisa Menjadi Programmer Java Basic Programming***. Elex Media, Jakarta.

McLeod, Raymond, 2008. ***Sistem Informasi Manajemen***. Salemba Empat, Jakarta.

Kusrini, 2007. ***Strategi Perancangan dan Pengolaan Basis Data***. Andi, Yogyakarta.

Gata, Windu, 2013. ***Sukses Membangun Aplikasi Penjualan Dengan Java***. Elex Media, Jakarta.