BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem merupakan serangkaian bagian yang saling tergantung dan bekerja sama untuk mencapai tujuan tertentu. Suatu sistem pasti tersusun dari sub-sub sistem yang lebih kecil yang juga saling tergantung dan bekerja sama untuk mencapau tujuan. Sebagai contoh, sistem administrasi universitas terdiri dari sub-sub sistem administrasi fakultas dan sub-sistem fakultas terdiri dari sub-sub sistem administrasi jurusan.

Tujuan dasar suatu sistem tergantung pada jenis sistem itu sendiri. Sebagai contoh, sistem peredaran darah manusia merupakan sistem biologi yang memiliki tujuan untuk mengedarkan darah yang mengandung oksigen dan sari makanan ke seluruh tubuh. Sedangkan sistem buatan manusia seperti sistem yang terdapat di sekolah, organisasi bisnis, atau instansi pemerintah juga mempunyai tujuan yang berbeda. Organisasi bisnis biasanya memiliki tujuan yang lebih jelas, seperti yang telah disebutkan pada bagian sebelumnya, yaitu mendapatkan laba.

Sistem informasi yang kadang kala disebut sebagai sistem pemrosesan data, merupakan sistem buatan manusia yang biasanya terdiri dari sekumpulan komponen, baik manual ataupun berbasis komputer yang terintegrasi untuk mengumpulkan, menyimpan dan mengelola data serta menyediakan informasi kepada pihak yang berkepentingan sebagai pemakai informasi tersebut (Anastasia Diana ; 2011 : 3).

II.2. Informasi

Informasi adalah data yang berguna yang telah diolah sehingga dapat dijadikan dasar untuk mengambil keputusan yang tepat. Informasi sangat penting bagi organisasi. Pada dasarnya informasi adalah penting seperti sumber daya yang lain, misalnya peralatan, bahan, tenaga, dsb.

Informasi yang berkualitas dapat mendukung keunggulan kompetitif suatu organisasi. Dalam sistem informasi akuntansi, kualitas dari informasi yang disediakan merupakan hal penting dalam kesuksesan sistem.

Secara konseptual seluruh sistem organisasional mencapai tujuannya melalui proses alokasi sumber daya, yang diwujudkan melalui proses pengambilan keputusan manajerial. Informasi memiliki nilai ekonomik pada saat ia mendukung keputusan alokasi sumber daya, sehingga dengan demikian mendukung sistem untuk mencapai tujuan.

Pemakai informasi akuntansi dapat dibagi dalam dua kelompok besar: ekstern dan intern. Pemakai ekstern mencakup pemegang saham, investor, kreditor, pemerintah, pelanggan, pemasok, pesaing, serikat pekerja, dan masyarakat. Pemakai intern terutama para manajer, kebutuhannya bervariasi tergantung pada tingkatannya (Agustinus ; 2012 : 1).

II.3. Akuntansi

Akuntansi merupakan bahasa bisnis. Sebagai bahasa bisnis akuntansi menyediakancara untuk menyajikan dan meringkas kejadian-kejadian bisnis dalam bentuk informasi keuangan kepada pemakainya. Informasi akuntansi merupakan bagian terpenting dari seluruh informasi yang diperlukan oleh

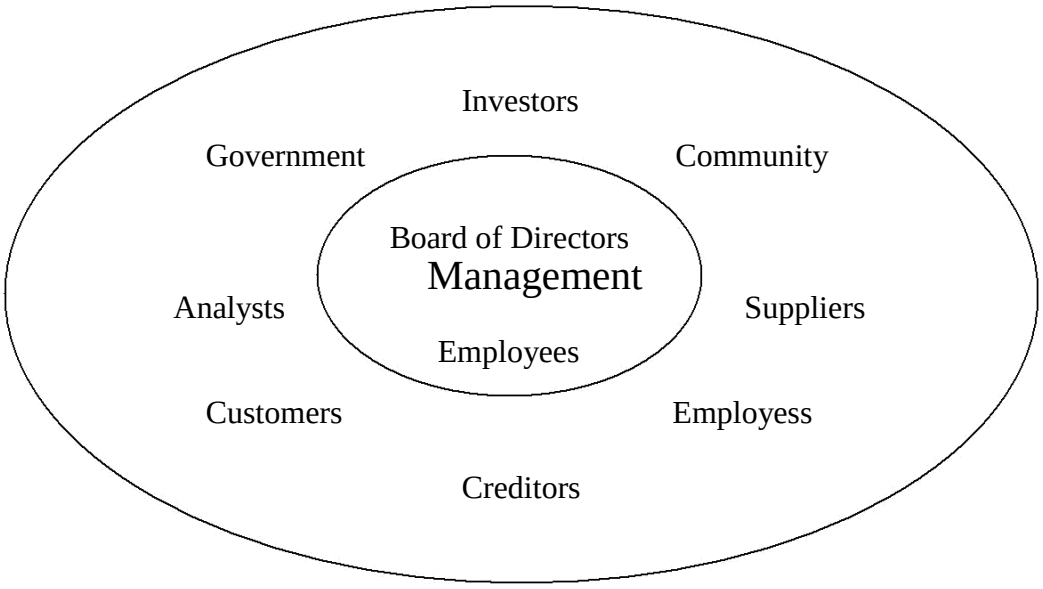
manajemen. Informasi akuntansi yang dihasilkan oleh suatu sistemdibedakan menjadi dua, yaitu informasi akuntansi keuangan dan informasi akuntansimanajemen.

Pemakai informasi akuntansi pun terdiri dari dua kelompok, yaitu pemakai eksternaldan pemakai internal. Yang dimaksud dengan pemakai ekseternal mencakuppemegang saham, investor, kreditor, pemerintah, pelanggan, pemasok, pesaing,serikat kerja dan masyarakat. Sedangkan pemakai internal adalah pihak manajer dariberbagai tingkatan dalam organisasi bersangkutan (Kusrini ; 2012 : 1).

1. **Pengguna Informasi Akuntansi**

   Pengguna informasi akuntansi dikelompokan sebagai berikut :

a. Intern → Pihak intern menggunakan informasi akuntansi untuk menyamai operasi perusahaan kelompok ini disebut pelaksana manajemen perusahaan
b. Ekstern →Kelompok ini adalah perorangan dan organisasi yang mempunyai kepentingan ekonomi dalam perusahaan tetapi bukan pihak manajemen.



Gambar II.1. Informasi Akuntansi
(Sumber : Eka ; 2012)

Proses Akuntansi :

Informasi akuntansi dihasilkan melalui proses akuntansi. Proses akuntansi dalam satu siklus meliputi langkah-langkah sebagai berikut :

Langkah 1 : Mengidentifikasi Transaksi atau Kejadian untuk dicatat

Tujuan : Untuk mendapatkan informasi, biasanya dalam bentuk dokumen sumber mengenai transaksi atau kejadian.

Langkah 2 : Menjurnal Transaksi dan Kejadian

Tujuan : Untuk mengidentifikasi, menilai, dan mencatat dampak ekonomi dari transaksi terhadap perusahaan secara kronologis, untuk mempermudah pemindahan ke dalam perkiraan.

Langkah 3 : Memposting dari Jurnal ke Buku Besar

Tujuan : Untuk memindahkan informasi dari jurnal ke buku besar yang berfungsi menyimpan perkiraan.

Langkah 4 : Mempersiapkan Neraca Saldo yang Belum Disesuaikan

Tujuan : Untuk memberikan daftar yang memudahkan dalam pengecekan keseimbangan debet dan kredit serta memberikan titik awal dalam pembuatan ayat jurnal penyesuaian.

Langkah 5 : Menjurnal dan Memposting Ayat Jurnal Penyesuaian

Tujuan : Untuk mencatat akrual, jatuh tempo deferal, estimasi, dan kejadian lainnya yang tidak disertai dengan dokumen sumber baru

Langkah 6 : Menyusun Neraca Saldo yang disesuaikan

Tujuan : Untuk mencocokkan keseimbangan debet- kredit dan untuk mempermudah penyusunan laporan keuangan

Langkah 7 : Membuat Laporan Keuangan

Tujuan : Untuk memberikan informasi yang berguna bagi pengambilan keputusan eksternal

Langkah 8 : Menjurnal dan Memposting Ayat Jurnal Penutup

Tujuan : Untuk menutup perkiraan sementara dan memindahkan jumlah laba bersih ke laba ditahan

Langkah 9 : Menyusun Neraca Saldo setelah Penutupan

Tujuan : Untuk memeriksa keseimbangan debet kredit setelah ayat jurnal penutup.

Langkah 10 : Menjurnal dan Memposting Ayat Jurnal Pembalik

Tujuan : untuk mempermudah pembuatan ayat jurnal berikutnya dan mengurangi biaya akuntansi (ini adalah langkah opsional).

II.4. Sistem Informasi Akuntansi

Akuntansi merupakan bahasa bisnis. Sebagai bahasa bisnis akuntansi menyediakan cara untuk menyajikan dan meringkas kejadian-kejadian bisnis dalam bentuk informasi keuangan kepada pemakainya. Informasi akuntansi merupakan bagian terpenting dari seluruh informasi yang diperlukan oleh manajemen. Informasi akuntansi yang dihasilkan oleh suatu sistem dibedakan menjadi dua, yaitu informasi akuntansi keuangan dan informasi akuntansi manajemen.

Pemakai informasi akuntansi pun terdiri dari dua kelompok, yaitu pemakai eksternal dan pemakai internal. Yang dimaksud dengan pemakai ekseternal mencakup pemegang saham, investor, kreditor, pemerintah, pelanggan, pemasok, pesaing, serikat kerja dan masyarakat. Sedangkan pemakai internal adalah pihak manajer dari berbagai tingkatan dalam organisasi bersangkutan. Sistem Informasi Akuntansi(SIA) dapat didefinisikan sebagai sebuah sistem informasi yang merubah data transaksi bisnis menjadi informasi keuangan yang berguna bagi Pemakainya (Kusrini ; 2012 : 1).

Sistem informasi akuntansi adalah sistem yang bertujuan untuk mengumpulkan dan memproses data serta melaporkan informasi yang berkaitan dengan saksi keuangan. Lingkup sistem informasi akuntansi dapat dijelaskan dari manfaat yang didapat dari informasi akuntansi. Manfaat atau tujuan sistem informasi akuntansi tersebut adalah sebagai berikut :

1. Mengamankan harta / kekayaan perusahaan. Harta / kekayaan di sini meliputi kas perusahaan, persediaan barang dagangan, termasuk aset tetap perusahaan.
2. Menghasilkan beragam informasi untuk pengambilan keputusan. misal, pengelola toko swalayan memerlukan informasi mengenai barang apa saja yang diminati oleh konsumen. Membeli barang yang kurang laku berarti kas akan terjebak dalam persediaan dan berarti kehilangan kesempatan untuk membeli barang dagangan yang laku.
3. Menghasilkan informasi untuk pihak eksternal. Setiap pengelola usaha memiliki kewajiban untuk membayar pajak. Besarnya pajak yang dibayar tergantung pada omset penjualan (jika pengelola memilih menggunakan norma dalam perhitungan pajaknya) atau tergantung pada laba rugi usaha (jika pengelola memilih untuk tidak menggunakan norma dalam perhitungan pajaknya).
4. Menghasilkan informasi untuk penilaian kinerja karyawan atau divisi. Sistem informasi dapat juga dimanfaatkan untuk penilaian kinerja karyawan atau divisi.
5. Menyediakan data masa lalu untuk kepentingan audit (pemeriksaan). Data yang tersimpan dengan baik sangat memudahkan proses audit (pemeriksaan).
6. Menghasilkan informasi untuk penyusunan dan evaluasi anggaran perusahaan. Anggaran merupakan alat yang sering digunakan perusahaan untuk mengendalikan pengeluaran kas.

7. Menghasilkan informasi yang diperlukan dalam kegiatan perencanaan dan pengendalian. Selain berguna untuk membandingkan informasi yang berkaitan dengan anggaran dan biaya standar dengan kenyataan seperti yang telah dikemukakan (Anastasia Diana ; 2011 : 6).

II.5. Bentuk Multiple Step

Bentuk *multiple step* adalah bentuk laporan laba rugi di mana dilakukan beberapa pengelompokkan terhadap pendapatan-pendapatan dan biaya-biaya yang disusun dalam urut-urutan tertentu sehingga bisa dihitung penghasilan-penghasilan sebagai berikut:

1. Laba bruto, yaitu hasil penjualan dikurangi harga pokok penjualan.
2. Penghasilan usaha bersih, yaitu laba bruto dikurangi biaya-biaya usaha.
3. Penghasilan bersih sebelum pajak, yaitu penghasilan usaha bersih ditambah dan dikurangi dengan pendapatan-pendapatan dan biaya-biaya di luar usaha.
4. Penghasilan bersih sesudah pajak, yaitu penghasilan bersih sebelum pajak dikurangi pajak penghasilan.

Penghasilan bersih dari elemen-elemen luar biasa, yaitu penghasilan bersih sesudah pajak ditambah dan/atau dikurangi dengan elemen-elemen yang tidak biasa sesudah diperhitungkan pajak penghasilan untuk pos luar biasa (Nelsi Wisama ; 2009 : 6).

II.6. Pengertian Java

Bahasa pemrograman Java merupakan karya SunsMicrosystemInc. Rilis resmi level *beta* dilakukan pada November 1995. Dua bulan berikutnya Netscape menjadi perusahaan pertama yang memperoleh lisensi bahasa Java dari Sun. Java adalah bahasa pemrograman berorientasi objek yang berukuran kecil, sederhana dan aman, diinterpretasi atau dioptimalisasi secara dinamis, ber-*bytecode*, arsitektur yang netral, mempunyai *garbage-collector, multithreading*, mempunyai mekanisme penanganan pengecualian, berbasis tipe untuk penulisan program mudah diperluas dinamis serta telah diperuntukkan sistem tersebar.

Pada awal pembuatannya, Java dinamakan Oak, kemudian nama Oak dinilai kurang menjual sehingga pada Januari 1995 nama Oak diubah menjadi Java. Sebagai bahasa yang bersifat terbuka, Java didukung oleh banyak *programmer* dari seluruh dunia yang memberikan kontribusinya untuk mengembangkan bahasa Java (Didik Dwi Prasetyo ; 2010 : 1).

Pada pengembangan *enterpriseapplications*, kita menggunakan sejumlah besar paket. Pada *consumerelectronicproduct*, hanya sejumlah kecil bagian bahasa yang digunakan. Masing-masing edisi berisi *Java 2 Software Development KIT* (SDK) untuk mengembangkan aplikasi dan *Java 2 Runtime Environment* (JRE) untuk menjalankan aplikasi.

a. *Standard Edition*(J2SE)

   The Java 2 Platform, Standard Edition(J2SE) menyediakan lingkungan pengembangan yang kaya fitur, stabil, aman dan *cross-platform*. Edisi ini mendukung konektivitas basis data, rancangan antarmuka pemakai,

masukan/keluaran dan pemrograman jaringan dan termasuk sebagai paket-paket dasar bahasa Java.

b. *EnterpriseEdition*(J2EE)

The Java 2, EnterpriseEdition(J2EE) menyediakan kakas untuk membangun dan menjalankan *multitierenterpriseapplications*. J2EE berisi paket-paket di J2SE ditambah paket-paket untuk mendukung pengembangan *EnterpriseJavaBeans, Java Servlets, JavaServer, Pages, XML,* dan kendali transaksi yang fleksibel.

c. *Micro Edition*(J2ME)

The Java 2, Micro Edition(J2ME) untuk beragam *consumerelectronicproduct,* seperti *pager, smartcard, cellphone, handheld*PDA, dan *set-topbox.* J2ME sembari menyediakan bahasa Java yang sama, unggul dalam portabilitas kemampuan dijalankan di mana pun dan *safenetworkdelivery*seperti J2SE dan J2EE (Didik Dwi Prasetyo ; 2010 : 1).

II.7. 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.8. Pengertian Database

Database merupakan kumpulan data yang saling berhubungan, hubungan antar data dapat ditunjukkan dengan adanya *field* kunci dari setiap tabel yang beda.

Dalam satu *file* atau tabel terdapat *record-record* yang sejenis, sama besar, sama bentuk, yang merupakan satu kumpulan entitas yang seragam. Satu *record* terdiri dari *field* yang saling berhubungan menunjukkan bahwa *fiel* tersebut satu pengertian yang lengkap dan disimpan dalam satu *record*. Basis data mempunyai beberapa kriteria penting yaitu :

1. Bersifat data oriented dan bukan program oriented.
2. Dapat digunakan oleh beberapa program aplikasi tanpa perlu mengubah basis datanya.
3. Dapat dikembangkan dengan mudah, baik *volume* maupun strukturnya.
4. Dapat memenuhi kebutuhan sistem-sistem baru secara mudah.
5. Dapat digunakan dengan cara-cara yang berbeda.

Prinsip utama *database* adalah pengaturan data dengan tujuan utama fleksibel dan kecepatan pada saat pengambilan data kembali. Adapun ciri-ciri basis data di antaranya adalah sebagai berikut :

1. Efesiensi meliputi kecepatan, ukuran dan ketepatan.
2. Data dalam jumlah besar.
3. Berbagi pakai (dipakai bersama-sama atau *sharebility*).
4. Mengurangi bahkan menghilangkan terjadinya duplikasi dan data yang tidak konsisten (Windu Gata ; 2013 : 19).

II.8.1. Hierarki Database

Data diorganisasikan ke dalam bentuk elemen data (*field*), rekaman (*record*), dan berkas (*file*). Definisi dari ketiganya adalah sebagai berikut:

Elemen data adalah satuan data terkecil yang tidak dapat dipecah lagi menjadi unit lain yang bermakna. Misalnya data siswa terdiri dari NIS, Nama, Alamat, Telepon atau Jenis Kelamin. Rekaman merupakan gabungan sejumlah elemen data yang saling terkait. Istilah lain dari rekaman adalah baris atau tupel. Berkas adalah himpunan seluruh rekaman yang bertipe sama (Haidar Dzacko ; 2007 : 1).

II.8.2. Model Database

Model data dapat dikelompokkan berdasarkan konsep pembuatan deskripsi struktur basis data, yaitu:

1. Model data konseptual (*high level*) menyajikan konsep tentang bagaimana *user* memandang atau memperlakukan data. Dalam model ini dikenalkan tiga konsep penyajian data yaitu:

a. *Entity* (entitas) merupakan penyajian obyek, kejadian atau konsep dunia nyata yang keberadaannya secara eksplisit didefinisikan dan disimpan dalam basis data, contohnya Mahasiswa, Matakuliah, Dosen, Nilai dan lain sebagainya.

b. *Atribute* (atribut) adalah keterangan-keterangan yang menjelaskan karakteristik dari suatu entitas seperti NIM, Nama, Fakultas, Jurusan untuk entitas Mahasiswa.

c. *Relationship* (hubungan) merupakan hubungan atau interaksi antara satu entitas dengan yang lainnya, misalnya entitas pelanggan berhubungan dengan entitas barang yang dibelinya.

2. Model data fiskal (*low level*) merupakan konsep bagaimana deskripsi detail data disimpan ke dalam komputer dengan menyajikan informasi tentang format rekaman, urutan rekaman, dan jalur pengaksesan data yang dapat membuat pencarian rekaman data lebih efisien.
3. Model data implementasi (*representational*) merupakan konsep deskripsi data disimpan dalam komputer dengan menyembunyikan sebagian detail deskripsi data sehingga para *user* mendapat gambaran global bagaimana data disimpan dalam komputer. Model ini merupakan konsep model data yang digunakan oleh model hierarki, jaringan dan relasional (Haidar Dzacko ; 2007 : 3).

II.9. Pengertian MySQL

Menurut Supardi (2007:97), perangkat lunak MySQL adalah perangkat lunak basis data *server* yang terkenal dan bersifat open-source dengan dukungan

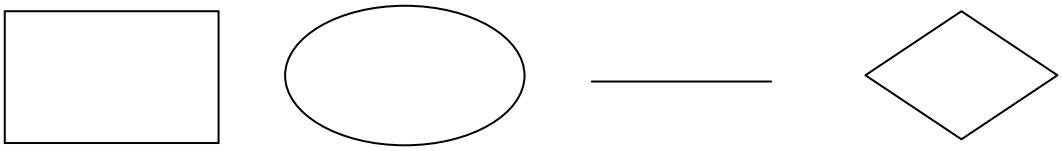
driver yang luas dari berbagai vendor. MySQL adalah seakuntansi implementasi dari sistem manajemen basis data relasional (RDBMS) yang didistribusikan secara gratis di bawah lisensi GPL (*General Public License*). Setiap pengguna dapat secara bebas menggunakan MySQL, namun dengan batasan perangkat lunak tersebut tidak boleh dijadikan produk turunan yang bersifat komersial. MySQL sebenarnya merupakan turunan salah satu konsep utama dalam basis data yang telah ada sebelumnya. SQL (*Structured Query Language*). SQL adalah seakuntansi konsep pengoperasian basisdata, terutama untuk pemilihan atau seleksi dan pemasukan data, yang memungkinkan pengoperasian data dikerjakan dengan mudah secara otomatis.

Kehandalan suatu sistem basisdata (DBMS) dapat diketahui dari cara kerja pengoptimasi-nya dalam melakukan proses perintah-perintah SQL yang dibuat oleh pengguna maupun program-program aplikasi yang memanfaatkannya. Sebagai peladen basis data, MySQL mendukung operasi basis data transaksional maupun operasi basisdata *non*-transaksional. Pada modus operasi *non*-transaksional, MySQL dapat dikatakan unggul dalam hal unjuk kerja dibandingkan perangkat lunak pengelolah basis data kompetitor lainnya. Namun demikian pada modus *non*-transaksional tidak ada jaminan atas reliabilitas terhadap data yang tersimpan, karenanya modus *non*-transaksional hanya cocok untuk jenis aplikasi yang tidak membutuhkan reliabilitas data seperti aplikasi *blogging* berbasis *web*, CMS, dan sejenisnya. Untuk kebutuhan sistem yang ditujukan untuk bisnis sangat disarankan untuk menggunakan modus basis data

transaksional, hanya saja sebagai konsekuensinya unjuk kerja MySQL pada modus transaksional tidak secepat unjuk kerja pada modus *non*-transaksional.

II.10. *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).



Entitas | Atribut/Field | Link (hubungan) | Himpunan Relasi/Interface

Gambar. II.2. Bentuk Simbol ERD
(Sumber : Ir. Yuniar Supardi ; 2010 : 448)

II.11. 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.12. Teknik Normalisasi

Normalisasi adalah teknik perancangan yang banyak digunakan sebagai pemandu dalam merancang basis data relasional. Pada dasarnya, normalisasi adalah proses dua langkah yang meletakkan data dalam bentuk tabulasi dengan menghilangkan kelompok berulang lalu menghilangkan data yang terduplikasi dari tabel rasional.

Teori normalisasi didasarkan pada konsep bentuk normal. Sebuah tabel relasional dikatakan berada pada bentuk normal tertentu jika tabel memenuhi himpunan batasan tertentu. Ada lima bentuk normal yang tekah ditemukan

II.12.1.Bentuk-bentuk Normalisasi

1. 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.

2. Bentuk normal tahap pertama (1” Normal Form)

   Definisi :

   Sebuah table disebut 1NF jika :

   - Tidak ada baris yang duplikat dalam tabel tersebut.
   - 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.

3. Bentuk normal tahap kedua (2^{nd} 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.

4. Bentuk normal tahap ketiga (3^{rd} 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 :

   - X haruslah *superkey* pada tabel tersebut.
   - Atau A merupakan bagian dari *primary key* pada tabel tersebut.

5. 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).

6. Boyce Code Normal Form (BCNF)
   - Memenuhi 1st NF
   - Relasi harus bergantung fungsi pada atribut superkey (Janner Simarmata ; 2010 : 76)..

II.13. 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.

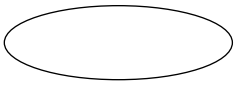
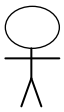
Alat bantu yang digunakan dalam perancangan berorientasi objek berbasiskan UML adalah sebagai berikut :

1. *Use case* Diagram

Use case diagram merupakan pemodelan untuk kelakukan (*behavior*) sistem informasi yang akan dibuat. *Use case* mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem informasi yang akan dibuat. Dapat dikatakan

use case digunakan untuk mengetahui fungsi apa saja yang ada di dalam sistem informasi dan siapa saja yang berhak menggunakan fungsi-fungsi tersebut. Simbol-simbol yang digunakan dalam *use case* diagram, yaitu :

Tabel II.1. Simbol *Use Case*

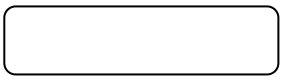
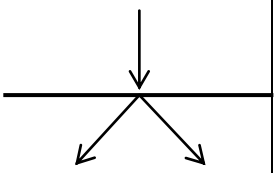
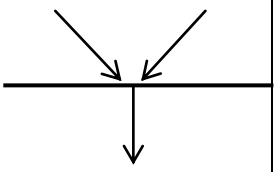
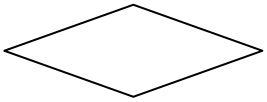
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi 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)

2. 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)

3. *Class Diagram* (Diagram Kelas)

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

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

Tabel II.3. *Multiplicity Class Diagram*

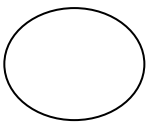
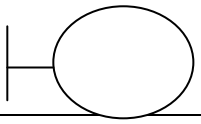
Multiplicity	Penjelasan
1	Satu dan hanya satu
0..*	Boleh tidak ada atau 1 atau lebih
1..*	1 atau lebih
0..1	Boleh tidak ada, maksimal 1
n..n	Batasan antara. Contoh 2..4 mempunyai arti minimal 2 maksimum 4

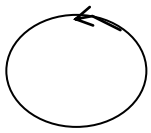
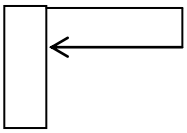
(Sumber : Windu Gata ; 2013 : 9)

4. Diagram Urutan (*Sequence Diagram*)

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

Tabel II.4. Simbol *Sequence Diagram*

Gambar	Keterangan
	*Entity Class*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.

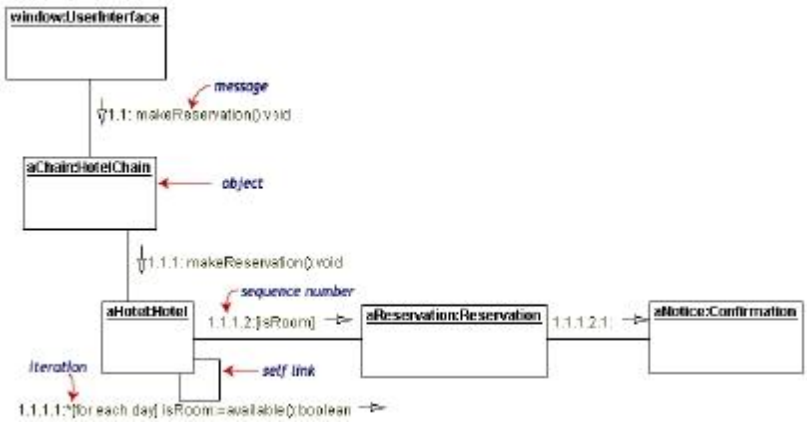
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation*, *activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata ; 2013 : 7)

5. *Collaboration* Diagram

Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message*.

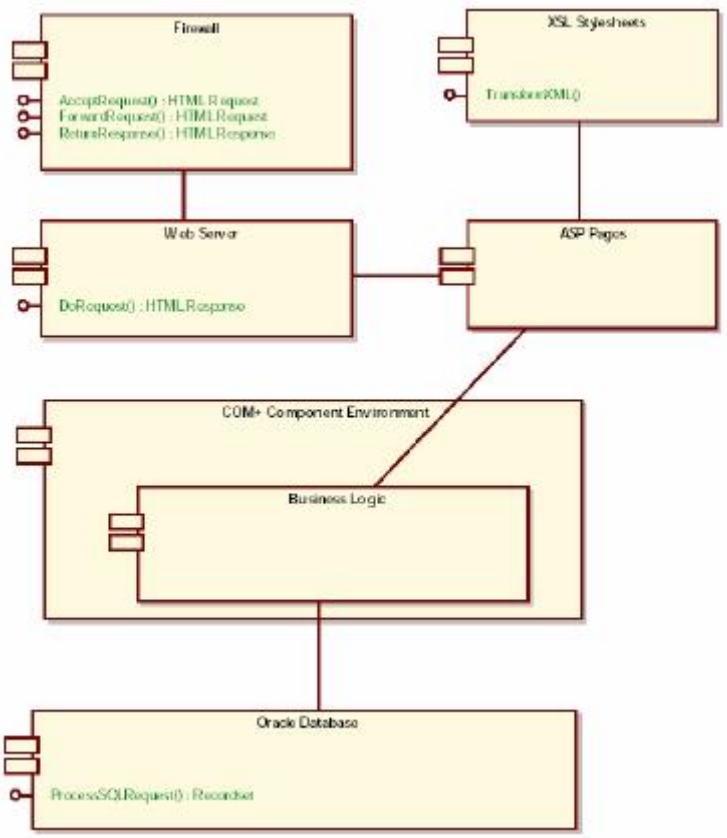
Setiap *message* memiliki *sequence number*, di mana *message* dari level tertinggi memiliki nomor 1. Messages dari level yang sama memiliki prefiks yang sama.



Gambar II.3. *Collaboration* Diagram
(Sumber : Sri Dharwiyanti ; 2013 : 9)

6. *Component* Diagram

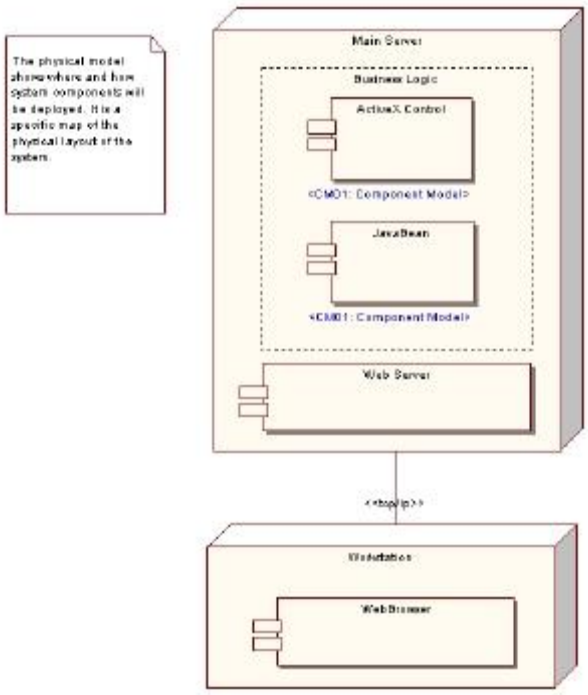
Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya. Komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time, link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain.



Gambar II.4. *Component* Diagram
(Sumber : Sri Dharwiyanti ; 2013 : 10)

7. *Deployment* Diagram

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras apa), bagaimana kemampuan jaringan pada lokasi tersebut, spesifikasi server, dan hal-hal lain yang bersifat fisikal Sebuah *node* adalah server, *workstation*, atau piranti keras lain yang digunakan untuk men-*deploy* komponen dalam lingkungan sebenarnya.

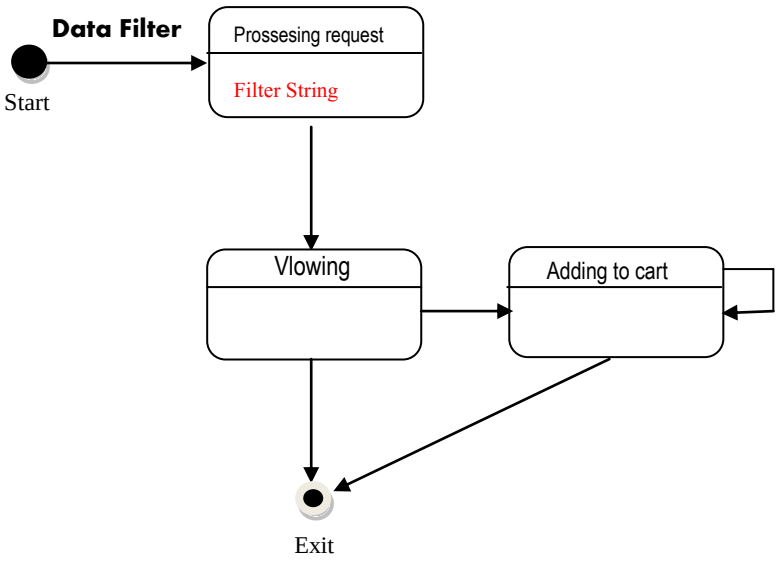


Gambar II.5. *Deployment* Diagram
(Sumber : Sri Dharwiyanti ; 2013 : 11)

8. *Statechart* Diagram

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*).

Dalam UML, *state* digambarkan berbentuk segiempat dengan sudut membulat dan memiliki nama sesuai kondisinya saat itu. Transisi antar *state* umumnya memiliki kondisi *guard* yang merupakan syarat terjadinya transisi yang bersangkutan, dituliskan dalam kurung siku. *Action* yang dilakukan sebagai akibat dari *event* tertentu dituliskan dengan diawali garis miring. Titik awal dan akhir digambarkan berbentuk lingkaran berwarna penuh dan berwarna setengah.



Gambar II.6. *Statechart* Diagram
(Sumber : Sri Dharwiyanti ; 2013 : 7)