BAB II

TINJAUAN PUSTAKA

II.1. Pengertian Perancangan

Model perancangan sesungguhnya adalah modal objek yang mendeskripsikan realisasi fisik *use case* dengan cara berfokus pada bagaimana spesifikasi-spesifikasi kebutuhan fungsional dan *non*-fungsional, bersama dengan batasan-batasan lain yang berhubungan dengan lingkungan implemenatasi, memiliki imbas langsung pada pertimbangan-pertimbangan pada aktivitas-aktivitas yang dilakukan pada tahap implementasi. Tambahannya, model perancangan sesungguhnya secara langsung bertindak sebagai abstraksi implementasi sistem/perangkat lunak dan dengan sendirinya model perancangan suatu saat nanti akan menjadi asupan bagi aktivitas-aktivitas selanjutnya yang kelak akan terdefinisi pada tahap implementasi (Adi Nugroho ; 2010 : 212).

II.2. Pengertian Sistem

Secara leksikal, sistem berarti susunan yang teratur dari pandangan, teori, asas dan sebagainya. Dengan kata lain, sistem adalah suatu kesatuan usaha yang terdiri dari bagian-bagian yang berkaitan satu sama lain yang berusaha mencapai suatu tujuan dalam suatu lingkungan kompleks. Pengertian tersebut mencerminkan adanya beberapa bagian dan hubungan antara bagian, ini menunjukkan kompleksitas dari sitem yang meliputi kerja sama antara bagian yang interpenden satu sama lain. Selain itu dapat dilihat bahwa sistem berusaha

mencapai tujuan. Pencapaian tujuan ini menyebabkan timbulnya dinamika, perubahan-perubahan yang terus menerus perlu dikembangkan dan dikendalikan. Definisi tersebut menunjukkan bahwa sistem sebagai gugus dan elemen-elemen yang saling berinteraksi secara teratur dalam rangka mencapai tujuan atau subtujuan (Marimin ; 2008 : 1).

II.3. Pengertian 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.4. Akuntansi

Menurut Gade (2005:4) dalam bukunya *Teori Akuntansi*, akuntansi adalah istilah yang luas yang menunjukkan teori-teori tertentu, asumsi-asumsi mengenai cara bertindak, peraturan-peraturan cara mengukur dan prosedur untuk mengumpulkan dan melaporkan informasi yang berguna tentang kegiatan-kegiatan dan tujuan suatu organisasi.

Definisi spesifik akuntansi adalah ilmu pengetahuan terapan dan seni pencatatan yang dilakukan secara terus menerus menurut sistem tertentu, mengolah dan menganalisis catatan tersebut sehingga dapat disusun suatu laporan keuangan sebagai pertanggungjawaban pimpinan perusahaan atau lembaga terhadap kinerjanya.

Tujuan utama akuntansi adalah memberikan informasi yang diperlukan untuk mengambil putusan bagi manajemen, pemegang saham, pemerintah atau pihak-pihak lain yang berkepentingan sehingga keputusan-keputusan yang benar dapat diambil tentang apa yang sudah terjadi dalam organisasi atau apa yang harus diperbuat di kemudian hari.

II.5. Sistem Informasi Akuntansi

Menurut Hall (2007:10) dalam bukunya *Sistem Informasi Akuntansi*, Sistem Informasi Akuntansi merupakan sistem yang memproses berbagai transaksi keuangan dan transaksi *non*-keuangan yang secara langsung memengaruhi 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 manajemen internal berbagai laporan keuangan bertujuan khusus serta informasi yang dibutuhkan untuk pengambilan keputusan, seperti anggaran, laporan kinerja serta laporan pertanggungjawaban.

II.6. Pengertian Ledger

Ada begitu banyak transaksi yang di tulis setiap hari di jurnal umum. Karena itu, tentu kita ingin jumlah kas saat ini atau ingin melihat jumlah piutang, utang dan persediaan yang dimiliki saat ini. Untuk itu, pencatatan transaksi ke jurnal umum harus disertai dengan pencatatan ke buku besar. Jadi, setelah bagian akuntansi menulis akuntansi keuangan ke jurnal umum, pada hari dan saat itu juga, harus dilakukan pencatatan (*posting*) ke buku besar (Ryan Ariefiansyah : 2013 : 40). Berikut adalah contoh dari pembuatan laporan buku besar (*ledger*) yang dapat dilihat pada gambar II.1 berikut :

Buku Besar Kas

Saldo 82.160.000

Tanggal		No. rek	Nama Rekening	Index	Debet (Rp.)	Kredit (Rp.)	Saldo (Rp.)
Okt	1	1-101	Kas	3	150.000.000		150.000.000
Okt	1	1-101	Kas	1		120.000.000	30.000.000
Okt	2	1-101	Kas	3	50.000.000		80.000.000
Okt	3	1-101	Kas	1		4.500.000	75.500.000
Okt	5	1-101	Kas	1		100.000	75.400.000
Okt	6	1-101	Kas	1	2.500.000		77.900.000
Okt	15	1-101	Kas	2		2.700.000	75.200.000
Okt	17	1-101	Kas	1	4.500.000		79.700.000
Okt	25	1-101	Kas	1	3.600.000		83.300.000
Okt	26	1-101	Kas	1		2.000.000	81.300.000
Okt	28	1-101	Kas	1		300.000	81.000.000
Okt	29	1-101	Kas	3		2.140.000	78.860.000
Okt	30	1-101	Kas	1	6.000.000		84.860.000
Okt	31	1-101	Kas	2		2.700.000	82.160.000
Subtotal					216.600.000	134.440.000	82.160.000

Gambar II.1. Laporan Buku Besar
(Sumber : Ryan Ariefiansyah : 2013 : 40)

II.7. Pengertian Annual Statement

Satu kali setiap tahunnya perusahaan mengkomunikasikan kepada pemegang sahamnya dan pihak-pihak lainnya yang tertarik dengan menerbitkan laporan keuangan lengkap yang telah di audit. Laporan tahunan, sebutan dari bentuk komunikasi yang dimaksud, meringkas hasil keuangan operasi perusahaan untuk tahun tersebut dan rencana masa depannya. Banyak laporan tahunan yang telah menjadi sebuah bentuk hubungan masyarakat, yang berisi foto-foto staf dan direksi perusahaan berikut foto-foto dan gambaran mengenai produk baru dan kantor baru. fungsi dasar dari setiap laporan tahunan adalah untuk melaporkan informasi keuangan, yang hampir smuanya merupakan produk dari sistem akuntansi perusahaan (Jerry ; 2007 : 629).

PT. XYZ
NERACA TAHUN 2011
(dalam ribuan)

Aktiva			PASIVA		
Kas dan Bank	Rp 200.000		Hutang Bank	Rp 100.000	
Efek	Rp 200.000		Hutang Dagang	Rp 300.000	
Piutang	Rp 160.000		Hutang Pajak	Rp 160.000	
Persediaan	Rp 840.000				
Jumlah Aktiva Lancar		Rp 1.400.000	Jumlah Hutang Lancar		Rp 560.000
Tanah	Rp 100.000		Obligasi 5%		Rp 600.000
Bangunan	Rp 1.000.000				
Mesin	Rp 700.000		Modal Saham	Rp 1.200.000	
Intangible	Rp 100.000		Agio	Rp 200.000	
Akumulasi Penyusutan	Rp (300.000)		Laba Ditahan	Rp 440.000	
Jumlah Aktiva Tetap		Rp 1.600.000	Jumlah Modal		Rp 1.840.000
Jumlah Aktiva		Rp 3.000.000	Jumlah Pasiva		Rp 3.000.000

Gambar II.2. Laporan Neraca Tahunan
(Sumber : Jerry ; 2007 : 629**)**

II.8. 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.

Api adalah *application programming interface* yaitu sebuah *layer* yang berisi *class-class* yang sudah didefinisikan dan antarmuka pemrogramam yang akan membantu para pengembang aplikasi dalam perancangan sebuah aplikasi. API memampukan para pengembang untuk dapat mengakses fungsi-fungsi sistem operasi yang diizinkan melalui bahasa Java. Pada saat ini dikenal ada tiga buah API dari Java, yaitu :

- J2SE, Java 2 Standard Edition adalah sebuah API yang dapat digunakan untuk mengembangkan aplikasi-aplikasi yang bersifat *client-side standalone* atau *applet* .

- J2EE, Java 2 Enterprise Edition adalah API yang digunakan untuk melakukan pengembangan aplikasi-aplikasi yang bersifat *server-side* seperti Java Servlet, dan Java Server Pages (Wahana Komputer ; 2010 : 3).

II.9. 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.10. 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.11. 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.

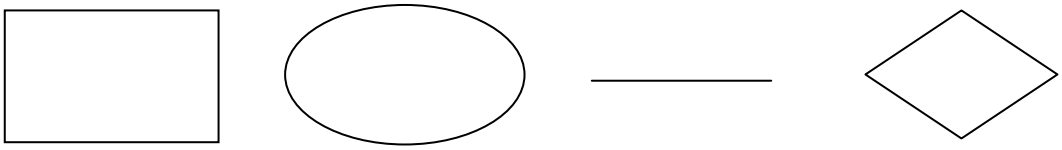
MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multi user*. MySQL memiliki dua bentuk lisensi, yaitu *free software* dan *shareware*. penulis sendiri dalam menjelaskan buku ini menggunakan database ini untuk keperluan pribadi atau usaha tanpa harus membeli 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 :

- MySQL dirilis pertama kali secara internal pada 23 Mei 1995

- Versi *windows* dirilis pada 8 Januari 1998 untuk *windows* 95.
- 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).

II.12. *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.3 Bentuk Simbol ERD
(Sumber : Ir. Yuniar Supardi ; 2010 : 448)

II.13. Kamus Data

Perancangan *database* digunakan dengan data *dictionary* (kamus data) yang merupakan daftar semua elemen/*field*. kamus data diperoleh pada saat analisis dengan diagram arus data. Pada perancangan ini dibuat dengan sekaligus contoh yaitu pada permasalahan komputerisasi karyawan yang mengerjakan proyek-proyek pada suatu perusahaan (Harianto Kristanto ; 2006 : 38).

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 aplikasi yang mempergunakan data tidak akan ikut terpengaruh (Raymond McLeod ; 2008 : 171).

II.14. Teknik Normalisasi

Proses normalisasi menyediakan cara sistematis untuk meminimalkan terjadinya kerangkapan data di antara relasi dalam perancangan logikal basis data. Format normalisasi terdiri dari lima bentuk, yaitu:

II.14.1.Bentuk-bentuk Normalisasi

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

Suatu tabel dikatakan sudah 1NF jika telah memenuhi ketentuan sebagai berikut:

- Tidak ada atribut mempunyai nilai berulang atau nilai array
- Tidak mempunyai baris yang rangkap

Bentuk unnormal mengijinkan nilai-nilai pada suatu atribut dapat berulang.

b. Bentuk normal tahap kedua (2nd normal form)

Relasi dapat dikatakan format normal kedua jika sudah dalam format normal pertama dan diikuti kondisi sebagai berikut:

- Key terdiri dari atribut tunggal
- Setiap atribut nonkey ketergantungan fungsional pada semua key atau tidak terjadinya ketergantungan pada key *composite*.

Misalnya tabel UNIV berada dalam normal kedua dengan mengasumsikan DNO sebagai key, kecuali CRSE. Jika ditentukan CNO dan SECNO sebagai key composite, atribut nonkey CNAME tergantung hanya pada CNO, bukan pada SECNO, sehingga CNAME tidak secara ketergantungan fungsional penuh terhadap key (CNO, SECNO).

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

Relasi dikatakan format normal ketiga jika sudah dalam format normal kedua dan tidak ada ketergantungan transitif di antara atribut. Misalnya tabel STUDNT mempunyai atribut SSNO sebagai *key* (2NF). Ketergantungan transitif terjadi di antara DNO dan COLREG. Saat DNO determinan COLREG tanpa melibatkan *key* SSNO. Contohnya, DNO=’CS’ termasuk COLREG=’Arts/Sc.’ tidak tergantung oleh atribut SSNO, sehingga STUDNT belum termasuk 3NF. Yang menjadi catatan, ketergantungan transitif tidak akan terjadi jika ada ketergantungan fungsional di antara atribut-atribut *non-key* yang melibatkan *key*.

d. Boyce Code Normal Form (BCNF)

BCNF menentukan setiap determinan adalah kunci kandidat (*candidate key*). Misalnya UNIV mempunyai dua determinan yaitu DNO dan DNAME yang merupakan *kunci kandidat* sehingga termasuk ke dalam BCNF. Di lain pihak CRSLST dalam 3NF tetapi tidak dalam BCNF. Atribut komposisinya (CNO, SECNO, SID, OFRNG) sebagai kunci-kunci kandidat dan tidak ada ketergantungan transitif, sehingga CRSLST termasuk ke dalam 3NF. Namun atribut CNO adalah determinan saat SECNO tergantung penuh secara fungsional terhadap CNO, walaupun CNO bukan kunci kandidat, sehingga CRSLST belum termasuk BCNF.

e. Bentuk Normal Tahap Keempat dan Kelima

Bentuk ini adalah bentuk normal ketiga atau BCNF dengan nilai atribut tidak tergantung pada nilai banyak (*multivalue dependency*). Konsep pada bentuk ini adalah ketergantungan pada gabungan beberapa atribut (*join dependency*) (Haidar Dzacko ; 2007 : 12).

II.15. 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

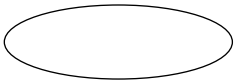
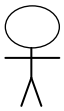
saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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

- *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 :

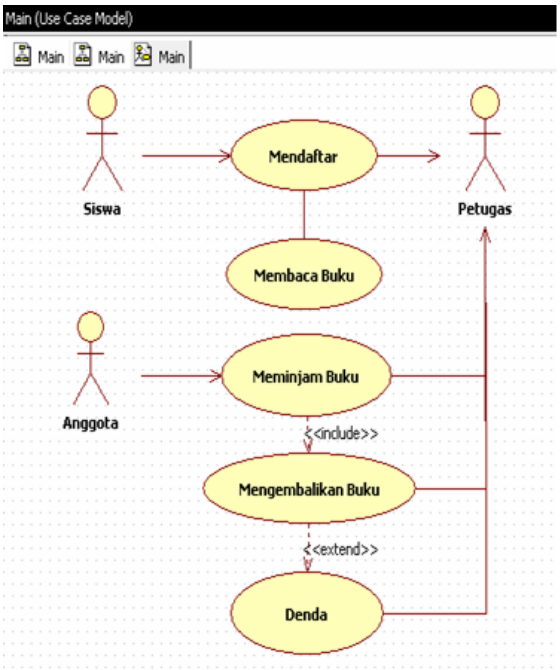
Tabel 1. Simbol *Use Case*

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)

Contoh dari pembuatan *use case diagram* dapat dilihat pada gambar II.4 berikut :

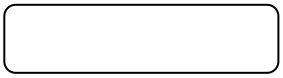
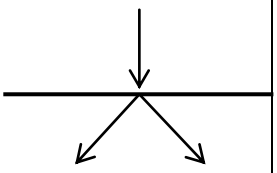
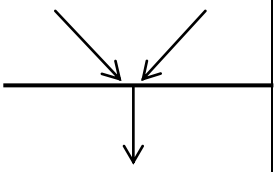
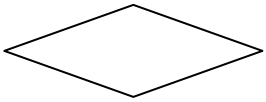
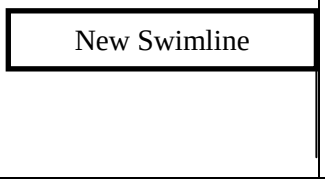


Gambar. II.4. *Use Case Diagaram*
(Sumber : Windu Gata ; 2013 : 4)

- Diagram Aktivitas (*Activity Diagram*)

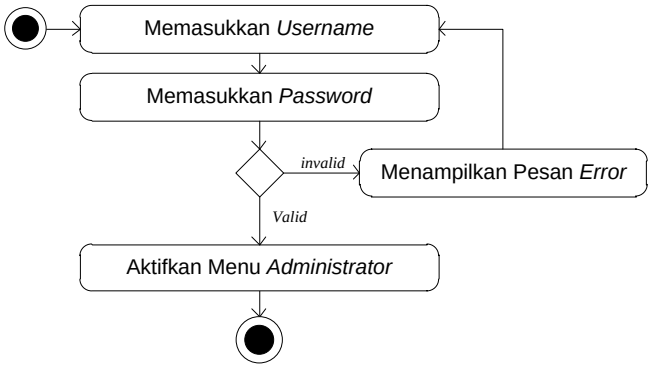
Activity Diagram menggambarkan *workflow* (aliran kerja) atau aktivitas dari sebuah sistem atau proses bisnis. Simbol-simbol yang digunakan dalam *activity diagram*, yaitu :

Tabel 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)

Contoh dari pembuatan *activity diagram* dapat dilihat pada gambar II.5 berikut :

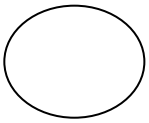
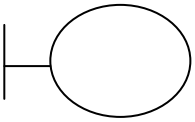
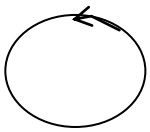
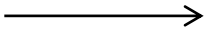


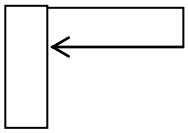
Gambar. II.5. *Activity Diagaram*
(Sumber : Windu Gata ; 2013 : 6)

- 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 :

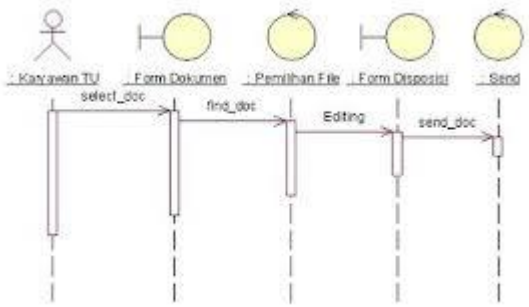
Tabel 3. Simbol *Sequence Diagram*

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)

Contoh dari pembuatan *sequence diagram* dapat dilihat pada gambar II.6 berikut :



Gambar. II.6. *Sequence Diagaram*
(Sumber : Windu Gata ; 2013 : 7)

- *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.

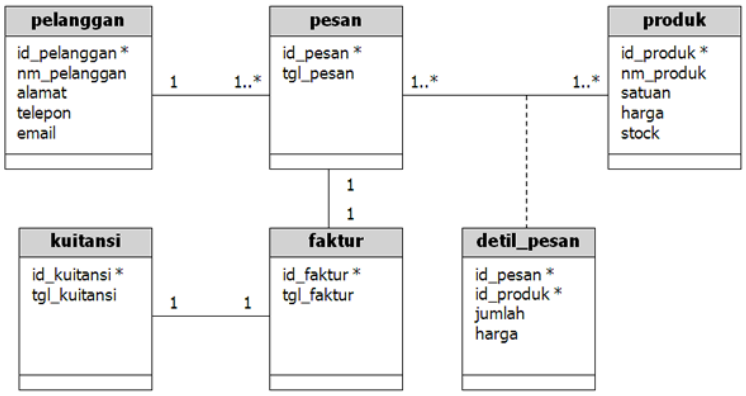
Class diagram secara khas meliputi: Kelas (*Class*), Relasi, *Associations*, *Generalization* dan *Aggregation*, Atribut (*Attributes*), Operasi (*Operations/Method*), *Visibility*, tingkat akses objek eksternal kepada suatu operasi atau atribut. Hubungan antar kelas mempunyai keterangan yang disebut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

Tabel 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 : 8)

Contoh dari pembuatan *use case diagram* dapat dilihat pada gambar II.7 berikut :



Gambar. II.7. *Class Diagaram*
(Sumber : Windu Gata ; 2013 : 8)