BAB II

TINJAUAN PUSTAKA

II.1. Sistem

Sistem merupakan sekumpulan elemen – elemen yang saling terintegrasi serta melaksanakan fungsinya masing – masing untuk mencapai tujuan yang telah ditetapkan. Karakteristik sistem terdiri dari :

1. Komponen Sistem

   Suatu sistem teridiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk suatu kesatuan. Komponen – komponen sistem atau elemen – elemen sistem dapat berupa suatu subsistem atau bagian – bagian dari sistem.

2. Batasan Sistem

   Batasan merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batasan sistem ini memungkinkan suatu sistem dipandang suatu kesatuan. Batasan suatu sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

3. Lingkungan Luar Sistem

   Lingkungan luar dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut.

4. Penghubung Sistem

   Penghubung merupakan media penghubung antara satu subsistem dengan subsistem lainnya. Melalui penghubung ini memungkinkan sumber – sumber daya mengalir dari satu subsistem ke subsistem lainnya.

5. Masukan Sistem

   Masukkan sistem adalah energi yang di masukkan ke dalam sistem. Masukkan dapat berupa masukan perawatan (*maintenance input*) dan masukkan sinyal (*signal input*). *Maintance input* adalah energi yang dimasukkan supaya sistem tersebut dapat beroperasi. Signal imput adalah energi yang diproses untuk mendapatkan keluaran.

6. Keluaran Sistem

   Keluaran sistem adalah hasil energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan.

7. Pengolahan Sistem

   Suatu sistem dapat mempunyai suatu bagian pengolah atau sistem itu sendiri sebagai pengolahnya. Pengolah akan mengubah masukkan menjadi keluaran.

8. Sasaran Sistem

   Suatu sistem dapat mempunyai tujuan (goal) atau sasaran (objective). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya. Sasaran dari sistem sangat menentukan sekali masukkan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya. (Jurnal SAINTIKOM vol.9 no.2 : 2010 ; 1)

II.1.1. Klasifikasi Sistem

Suatu sistem dapat diklasifikasikan menjadi sebagai berikut dan didefenisikan:

1. Sistem abstrak dan sistem fisik

   Sistem abstrak adalah sistem yang berupa pemikiran atau ide - ide yang tidak tampak secara fisik, sedangkan sistem fisik merupakan sistem yang ada secara fisik.

2. Sistem alamiah dan sistem buatan manusia

   Sistem alamiah adalah sistem yang terjadi melalui proses alam. Sedangkan sistem buatan manusia merupakan sistem yang melibatkan interaksi manusia.

3. Sistem tertentu dan sistem tak tentu

   Sistem tertentu adalah suatu sistem yang operasinya dapat diprediksi secara tepat sedangakan sistem tak tertentu adalah sistem dengan perilaku ke depan yang tidak diprediksi.

4. Sistem terbuka dan sistem tertutup

   Sistem tertutup adalah sistem yang tidak terpengaruh oleh lingkungan luar atau otomatis, sedangkan sistem terbuka adalah sistem yang berhubungan dan terpengaruh oleh lingkungan luar. (Jurnal SAINTIKOM vol.9 no.2 : 2010 ; 2).

II.2. Informasi

Informasi adalah data yang diolah menjadi suatu bentuk yang berarti bagi penerimanya dan bermanfaat dalam pengambilan keputusan pada saat sekarang atau yang akan datang. Informasi juga merupakan fakta – fakta atau data yang

telah diproses sedemikian rupa atau mengalami proses transformasi data sehingga berubah bentuk menjadi informasi.

Kualiatas dari suatu informasi tiga hal yaitu ;

1. Akurat, berarti informasi harus bebas dari kesalahan – kesalahan dan tidak bias atau menyesatkan. Akurat juga berarti informasi harus jelas mencerminkan maksudnya. Informasi harus akurat karena dari sumber informasi sampai ke penerima informasi kemungkinan banyak terjadi gangguan (*noise*) yang dapat merubah atau merusak informasi tersebut.
2. Tepat pada waktunya, berarti informasi yang pada penerima tidak boleh terlambat. Informasiyang sudah usang tidak akan mempunyai nilai lagi. Karena informasi merupakan landasan di dalam pengambilan keputusan. Bila pengambilan keputusan terlambat, maka dapat berakibat fatal untuk organisasi.
3. Relevan, berarti informasi tersebut mempunyai manfaat untuk pemaikainya. Relevansi informasi untuk tiap – tiap orang satu dengan lainnya berbeda. (Jurnal SAINTIKOM vol.9 no.2 : 2010 ; 3)

II.3. Sistem Informasi

Berikut ini adalah defenisi sistem informasi menurut beberapa ahli:

Tabel II.1. Defenisi Sistem Informasi

Sumber	Defenisi
Alter (1992)	Sistem informasi adalah kombinasi antar prosedur kerja, informasi, orang, dan teknologi informasi yang diorganisasikan untuk mencapai tujuan dalam sebuah organisasi
Bodnar Dan	Sistem informasi adalah sekumpulan perangkat keras

Hopwood (1993)	dan perangkat lunak yang dirancang untuk mentransformasikan data kedalam bentuk informasi yang berguna
Gelinas, Oram Dan Wiggins (1990)	Sistem informasi adalah suatu sistem buatan manusia yang secara umum terdiri atas kumpulan komponen berbasis komputer dan manual yang dibuat untuk menghimpun menyimapan, dan mengelola data ke serta menyediakan informasi keluaran pada para pemakai
Hall (2001)	Sistem informasi adalah sebuah rangkaian prosedur formal dimana data dikelompokkan, diproses menjadi informasi dan didistribusikan kepada pemakai
Turban , Mclean dan Wetherbe (1999)	Sebuah sistem informasi mengumpulkan, memproses, menyimpan, menganalisis dan menyeabarkan informasi untuk tujuan spesifik
Wilkinson (1992)	Sebuah sistem informasi adalah kerangka kerja yang mengkoordinasikan suber daya (manusia, komputer) untuk mengubah masukan (input) menjadi keluaran (informasi), guna mencapai sasaran perusahaan

(Sumber: Abdul Kadir, 2012).

II.4. Mutasi Karyawan

Mutasi dalam manajemen sumber daya manusia disebut pergeseran tugas. Siagian menjelaskan mutasi dapat dilihat dalam dua perspektif. Pertama, dianggap sebagai bentuk penempatan untuk seseorang pada tugas dengan tanggung jawab baru, perubahan posisi hirarkis dan pendapatan relatif dari posisi yang lama, sedangkan bentuk kedua mengacu pada pergeseran dari tempat kerja ke tempat yang sama dengan tanggung jawab yang relatif sama dan tanpa perubahan pendapatan. Selain itu, ia berpendapat bahwa mutasi pada umumnya dilakukan setidaknya setiap 2 tahun dan maksimal 14 tahun berdasarkan usulan dari Kepala Unit. Proses mutasi pekerjaan termasuk dalam penilaian kinerja dan berhubungan dengan kegiatan promosi dan pelatihan. Siagian menambahkan mutasi pekerjaan membawa beberapa manfaat seperti memiliki perspektif baru

pada kehidupan organisasi, memiliki pandangan yang lebih luas, memperoleh pengetahuan dan keterampilan baru, memiliki motivasi kerja dan kepuasan kerja. Selain itu, mutasi pekerjaan dapat memberikan pengalaman baru kepada pegawai melalui peningkatan cara berpikir yang akan mempengaruhi prestasi kerja. Pegawai dapat menjalani berbagai agenda organisasi ketika terlibat dalam mutasi, sehingga berguna untuk mencegah karyawan penurunan semangat kerja yang berdampak buruk terhadap kinerja.

II.5. Normalisasi

Normalisasi adalah suatu teknik untuk memecah/ mendekomposisi data dalam cara – cara tertentu untuk mencegah timbulnya permasalahan pengolahan data dalam basis data dalam bisnis data. Proses normalisasi terdiri dari beberapa level, yaitu :

1. Bentuk Tidak Normal (Un Normalized Form/UNF). Kriteria dari bentuk ini adalah :
   a. Jika relasi mempunyai bentuk *nonflat file*.
   b. Jika relasi memuat set attribut berulang.
   c. Jika relasi memuat *attribut non atomic value*.
2. Bentuk Normal Pertama (First Normal Form/ 1NF). Kriteria dari bentuk ini adalah :
   a. Jika seluruh attribut dalam relasi bernilai atomaik.
   b. Jika seluruh attribut dalam relasi bernilai tunggal.
   c. Jika relasi tidak memuat set attribut berulang.

d. Jika semua record mempunyai sejumlah attribut yang sama.

Permasalahan dalam adalah :

a. Tidak dapat menyisipkan informasi parsial.
b. Terhapusnya informasi ketika menghapus sebuah record.
c. Pembaharuan attribut non kunci mengakibatkan sejumlah record harus diperbaharui.

Untuk mengubah relasi UNF menjadi bentuk 1NF dilakukan dengan cara :

a. Melengkapi nilai – nilai dalam attribut.
b. Mengubah struktur relasi.

3. Bentuk Normal Kedua (Second Normal Form / 2NF). Kriteria dari bentuk ini adalah :
   a. Jika memenuhi kriteria 1NF
   b. Jika semua attribut non kunci memiliki ketergantungan fungsional terhadap attribut kunci.

Permasalahan dalam 2NF adalah :

a. Kerangkapan data.
b. Pembaharuan yang tidak benar dapat menimbulkan inkonsistensi data.
c. Proses pembaharuan data tidak efisien.
d. Penyimpanan pada saat penyisipan, penghapusan dan pembaharuan.

Untuk mengubah relasi 1NF menjadi bentuk 2NF dilakukan dengan cara :

a. Identifikasi ketergantungan fungsional pada relasi 1NF.

b. Berdasarkan ketergantungan fumgsional, dekomposisi relasi 1NF menjadi relasi – relasi baru sesuai dengan ketergantungan fungsionalnya.

4. Bentuk normal Ketiga (Third Normal Form / 3NF). Kriteria dari bentuk ini adalah :
   a. Jika memenuhi kriteria 3NF.
   b. Jika semua attribut non kunci tidak memiliki ketergantungan transitif kunci. (Janer Simarmata : 2010)

II.5.1. Basis Data (*Database*)

Secara sederhana database (basis data/ pangkalan data) dapat diungkapkan sebagai suatu pengorganisasian data dengan bantuan komputer yang memungkinkan data dapat diakses dengan mudah dan cepat (Kadir, 2004). Pengertian akses dapat mencakup pemerolehan data maupun pemanipulasian data seperti menambah serta menghapus data. Dengan memanfaatkan komputer, data dapat disimpan dalam media pengingat yang disebut *hard disk*. Dengan menggunakan media ini, keperluan kertas untuk menyimpan data dapat dikurangi. Selain itu, data menjadi lebih cepat untuk diakses terutama jika dikemas dalam bentuk database.

Pengaplikasian database dapat kita lihat dan rasakan dalam keseharian kita. Database ini menjadi penting untuk mengelola data dari berbagai kegiatan. Misalnya, kita bisa menggunakan mesin ATM (anjungan tunai mandiri/ *automatic teller machine*) bank karena bank telah mempunyai database tentang nasabah dan

rekening nasabah. Kemudian data tersebut dapat diakses melalui mesin ATM ketika bertransaksi melalui ATM. Pada saat melakukan transaksi, dalam konteks database sebenarnya kita sudah melakukan perubahan (*update*) data pada database di bank.

Ketika kita menyimpan alamat dan nomor telepon di HP, sebenarnya juga telah menggunakan konsep database. Data yang kita simpan di HP juga mempunyai struktur yang diisi melalui formulir (*form*) yang disediakan. Pengguna dimungkinkan menambahkan nomor HP, nama pemegang, bahkan kemudian dapat ditambah dengan alamat *email*, alamat *web*, nama kantor, dan sebagainya.

Pemahaman tentang database ini dapat didekatkan pada konsep akuntansi. Kita bisa umpamakan bahwa ketika kita melakukan proses akuntansi secara manual, kita menuliskan suatu catatan ke dalam lajur dan kolom buku. Mulai dari jurnal, buku besar, buku pembantu kita memasukkan catatan satu demi satu. Melihat buku akuntansi tersebut, sebenarnya kita sudah melihat konsep database, yang jika dikelola dengan komputer masih diperlukan penyesuaian dalam membentuk kolom-kolomnya. (Mujilan, 2012)

II.5.2. Model Database

Model database yang saat ini banyak digunakan adalah model database relational. Imam (2008) menyebutkan "Model database ini disusun dalam bentuk tabel dua dimensi yang terdiri dari baris (*record*) dan (*field*), pertemuan antara baris dengan kolom disebut item data (*data value*). Tabel-tabel yang ada dihubungkan (*relationship*) sedemikian rupa menggunakan *field-field* kunci (*key field*) sehingga dapat meminimalkan duplikasi data."

Model database relational ini dapat kita kenal konsepnya mulai dari yang paling sederhana misalnya dengan penerapan program aplikasi *excel*. Meskipun untuk pengelolaan database secara luas *excel* jarang digunakan dan kurang mencukupi, namun untuk melihat konsep database dan konsep membangunnya program ini dapat dimanfaatkan. *Excel* mempunyai baris yang disebut *raw* dan mempunyai kolom. Kemudian item data merupakan sel atau pertemuan antara baris dan kolom. Tabel-tabel dapat diumpakan apabila kita menggunakan tabel dalam suatu *sheet* tertentu. Data dari berbagai tabel dapat diambil dari tabel lain menggunakan perintah *look up* yang berdasarkan kode kunci tertentu. Kode kunci tersebut berada pada suatu kolom tertentu, yang dalam konsep database relational disebut sebagai *key field* tadi.

II.5.3. Struktur *Database*

Untuk memahami konteks *database* kita perlu memahami istilah dan hal-hal yang terkait dengan database. Dalam berbagai program aplikasi database terdapat kesamaan ataupun sedikit perbedaan di dalamnya. Seseorang yang mempelajari database dengan program aplikasi tertentu harus memperhatikan struktur dan karakteristik sesuai dengan bahasa dalam aplikasi tersebut. Namun demikian, secara umum terdapat karakteristik sebagai berikut:

1. Nama *file*

Nama *file* adalah nama yang digunakan untuk mengidentifikasi adanya data yang disimpan dalam komputer dan digunakan untuk pemanggilan data. *File* yang dikelola akan muncul dalam komputer dengan ekstensi sesuai dengan program aplikasinya. *File* tersebut dapat digunakan untuk menandakan adanya *file*

database, ataupun *file table*. Database dan table akan saling terkait, meskipun cara menyimpan dalam komputer akan mengalami sedikit perbedaan pada beberapa aplikasi. Misalnya Ms. *Access* akan menyimpan dengan *file* yang dapat kita lihat adalah *file* databasenya. Di program *MySql* nama database ini akan menjadi *folder*. Sementara di *FoxPro* nama database dapat menjadi *file* tersendiri. *Table* cara menyimpannya juga berbeda, dalam *Ms. Access* mungkin kita tidak melihat nama *table* secara kasat mata karena akan dikelola di dalam *file* database. Di dalam *MySql* kita bisa melihat beberapa nama *file* terkait dengan pengelolaan *table*. Dan di dalam *FoxPro table* ini dapat menjadi nama *file* terpisah dan dapat dikenali pula sebagai *free table*.

2. Database

Database sebenarnya merupakan nama untuk menampung berbagai *table* di dalamnya. Konsep ini akan sama dalam berbagai program aplikasi. Misalnya kita membangun database akuntansi dengan nama database "*akun_base*". Di dalam *akun_base* akan diorganisasi berbagai table yang terkait dengan kegiatan akuntansi misalnya tabel: rekening, pelanggan, jurnal, buku induk, dan administrator program, dan sebagainya. Setiap data yang masuk tidaklah dicatat dalam database, namun di dalam masing-masing *table* yang sesuai.

3. *Table*

Table merupakan tempat untuk menyimpan data sesuai dengan kelompok data. Setiap isi table mengandung data yang mempunyai karakteristik dalam penggunaannya. Untuk mempermudah pengolahan biasanya pembangun database mengkategorikan *table* sesuai dengan data isinya sebagai berikut :

a. *Master table*

Master table berisi data tentang hal-hal utama dalam kegiatan database. Table ini berisi record yang relatif permanen atau seringkali menjadi acuan ketika mengoperasikan transaksi. Dalam master tabel identitas record menjadi penting dan diusahakan merupakan data atau kode yang bersifat unik. Unik dapat diartikan bahwa tidak ada dalam satu table berisi kode yang sama. Disain kode menjadi penting di sini. Misalnya dalam mendisain nama akun dalam database akuntansi, maka kode akun menjadi sangat penting artinya. Dalam *table* berisi nama barang, maka kode barang menjadi hal penting. Contoh lain dalam database akademik, tabel *master* dapat berupa : mahasiswa, daftar dosen, daftar kurikulum.

b. *Transaction table*

Tabel transaksi digunakan untuk menyimpan data dalam menjalankan suatu kegiatan atau bisnis. Data ini seringkali akan bertambah dalam kesehariannya ketika terjadi transaksi yang sesuai dengannya. Secara lebih mudah dapat dipahami dalam akuntansi seringkali mencatat transaksi dalam jurnal. Terkait hal tersebut, transaksi ini dicatat dalam tabel jurnal. Dalam mencatat transaksi ini, kita harus menyesuaikan kode data tertentu dengan kode yang terdapat dalam *master table.*

c. *Tabulation table*

Tabulasi data dapat digunakan untuk menyimpan data seperti halnya master data namun bersifat sebagai data pembantu ketika menginput formulir baik untuk data master maupun transaksi. Misalnya untuk memetakan

keterangan hobi, jenis kelamin, nama golongan, nama level manajemen, dan sebagainya. Dengan konsep penamaan field yang baik mungkin saja table tabulasi ini dapat digunakan untuk memuat berbagai kelompok data. Misalnya fieldnya berupa kode dan keterangan. Contoh kelompok gender dengan L = laki-laki; P = perempuan. Kelompok level dengan M = Manajer, O = operator, S = seller

d. *Temporary table*

Temporary adalah data sementara yang digunakan untuk membantu ketika terjadi proses transaksi. Data ini dapat saja langsung dihapus ketika transaksi selesai terproses. Misalnya digunakan untuk mempermudah perhitungan, penyimpanan data sementara sebelum diproses setuju ke database. Misalnya: ketika terjadi transaksi di depan kasir, data-data pertama akan ditangkap dan dimasukan dalam file temporary sebelum akhirnya kasir melakukan perintah "ok" yang menandakan data transaksi siap untuk disimpan atau diproses dalam komputer. Ketika masa tunggu ini, data masih dapat diedit, dibatalkan, ataupun ditambah. Sementara ketika sudah masuk ke sistem, edit atau penambahan akan membutuhkan prosedur tertentu. Seandainya dianalogikan dengan sistem akuntansi maka proses edit data yang telah masuk ke sistem dapat digunakan prosedur seperti halnya melakukan jurnal koreksi.

4. *Field*

Field adalah penanda untuk kolom data. Jika dalam *excel* penanda tersebut adalah kolom A, B, dan seterusnya, sementara dalam konsep *table* dalam database

maka nama *field* memegang peranan penting. Dalam konsep table dalam database, ketika memanggil dengan nama *field* tertentu maka data-data di dalamnya akan muncul. Pengolahan dapat dilakukan dengan membuat *filter*, misalnya berdasarkan kode tertentu, berdasarkan *record* tertentu. Misalnya kita ingin memanggil record terkait nama karyawan Andi. Dalam database Andi ini diberi ID: 11001. Sehingga kita bisa menggunakan konsep filtrasi untuk memanggil personalia dengan kode ID 11001. Apabila data ketemu, maka kita dapat menggunakan data berdasarkan *field-field* yang ada, misalnya nama, tempat lahir, tanggal lahir, alamat, dan sebagainya yang mengacu pada ID 11001.

Dalam mengatur setting *field*, biasanya akan terkait hal-hal sebagai berikut :

a. *Field type* : tipe *field* ini dapat terkait apakah *field* tersebut akan berisi data berupa *key field* (*primary key*, *secondary key*), atau *descriptor*. *Primary key* akan berisi ID atau kode pokok yang akan digunakan dalam mengidentifikasi *record*, sehingga data di dalam *field* tersebut tidak diijinkan untuk memiliki lebih dari satu data yang sama. *Secondary* adalah subset dari *key* utama. Misalnya saja kode mata kuliah dalam satu semester tidak boleh terdapat lebih dari satu pada ID atau NIM yang sama. *Descriptor* adalah *field* berisi data yang akan merupakan satu kesatuan dengan yang lainnya sebagai penjelasan akan adanya *record* atau ID tertentu.

b. Data *type*: tipe data merupakan jenis data yang dapat dimasukkan dalam *field*. Hal ini dapat dibagi secara umum sebagai karakter/ *text*, numerik, tanggal dan sebagainya.

c. *Field Size*. Penting untuk memahami ukuran field yang akan digunakan dalam menampung data. Dalam pengembangan sistem harus dapat memperkirakan berapa lebar ukuran *field* yang efektif. Apabila terlalu lebar akan terjadi banyak spasi kosong dan berpengaruh pada ukuran *file* yang disimpan. Sementara apabila terlalu sempit akan terdapat data yang tidak tersimpan. Misalnya ketika kita menghitung bahwa nama

menggunakan ukuran 40 karakter sudah memenuhi untuk *field* data kita. Konsekuensinya, apabila terdapat nama di atas 40 karakter maka akan terpotong menjadi 40 karakter. Konsekuensi lain adalah nama tersebut diinput hingga memuat maksimal 40 karakter yaitu dengan mengadakan singkatan nama.

5. *Records*

Records merupakan baris data. Karena satu baris data biasanya mengindikasikan satu kesatuan data tertentu, maka satu *record* ada yang menyebut satu data. Misalnya keterangan mengenai biodata Andi disimpan dalam satu record beridentitas ID 11001, maka untuk menyebutkan satu kesatuan data seputar Andi dalam baris tertentu ada yang menyebutkan sebagai *record* data Andi. Dalam konsep database masing-masing *record* memiliki nomor identitas tersendiri baik itu identitas yang diberikan komputer

ataupun yang diinputkan secara manual. Sehingga dalam konteks tertentu dapat digunakan konsep nomor *record*, ID otomatis, ID *primary key*. (Mujilan, 2012)

II.6. *Unified Modeling Language* (UML)

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 industry perangkat lunak dan pengembangan sistem. (Windu Gata : 2013)

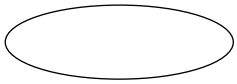
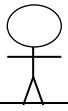
II.6.1. Diagram-*Diagram UML*

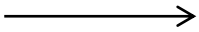
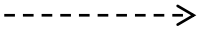
1. *Diagram Use Case (Use Case Diagram)*

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 Diagram*

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

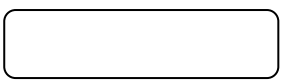
	pembagian tenaga kerja dan tugas-tugas yang berkaitan dengan peran pada konteks target sistem. Orang atau sistem bisa muncul dalam beberapa peran. Perlu dicatat bahwa aktor berinteraksi dengan use case, tetapi tidak memiliki control terhadap use case.
	Asosiasi antara aktor dan use case, digambarkan 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.

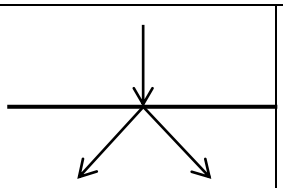
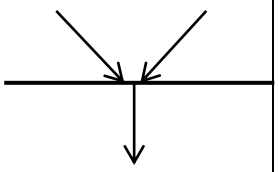
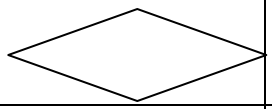
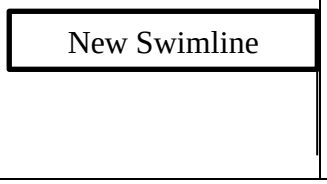
Sumber : (Windu Gata : 2013)

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 2. Simbol Diagram Aktivitas

Gambar	Keterangan
	*Start point,* diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point,* akhir aktifitas.
	*Activites,* menggambarkan suatu proses/kegiatan bisnis.

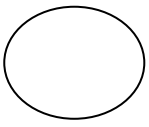
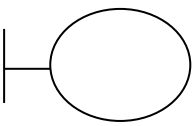
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan dua kegiatan pararel menjadi satu.
	*Join* (penggabungan) atau rake, digunakan untuk menunjukkan adanya dekomposisi.
	*Decision Points*, menggambarkan pilihan untuk pengambilan keputusan, *true*, *false.*
	*Swimlane*, pembagian *activity* diagram untuk menunjukkan siapa melakukan apa.

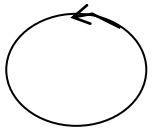
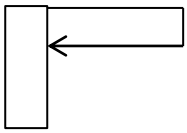
Sumber : (Windu Gata : 2014)

3. *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)

4. 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. Simbol 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

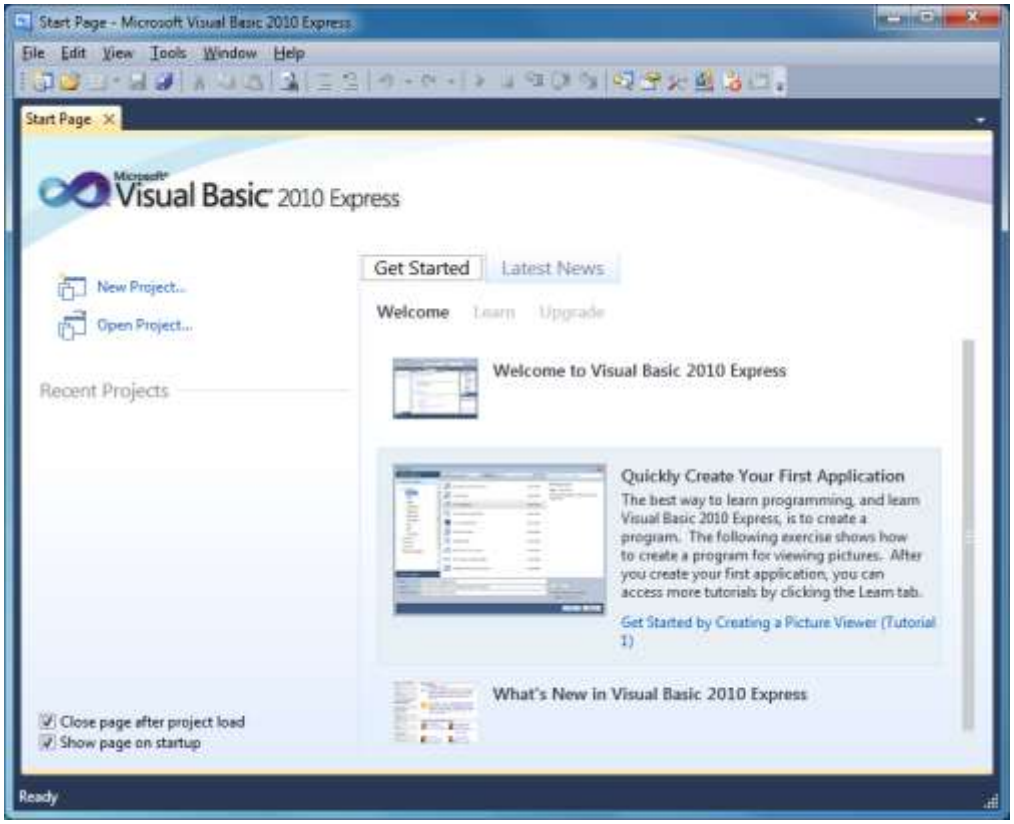
Sumber : (Windu Gata : 2013)

II.7. Bahasa Pemograman *Microsoft Visual Studio 2010*

Microsoft Visual Studio 2008 merupakan kelanjutan dari *Microsoft Visual Studio* sebelumnya, yaitu *Visual Studio. Net* 2003 yang diproduksi oleh Microsoft. Pada bulan Februari 2002 *Microsoft* memproduksi teknologi. *Net Framework* versi 1.0, teknologi. *Net* ini didasarkan atas susunan berupa *Net Framework*, sehingga setiap produk baru yang terkait dengan teknologi. Net akan selalu berkembang mengikuti perkembangan. *Net Frameworknya*. Pada perkembangannya nantinya mungkin untuk membuat program dengan teknologi. *Net* memungkinkan para pengembang perangkat lunak akan dapat menggunakan lintas sistem operasi, yaitu dapat dikembangkan di sistem operasi windows juga dapat dijalankan pada sistem operasi lain, misalkan pada sistem operasi *Linux*, seperti yang telah dilakukan pada pemograman *Java* oleh *Sun Microsystem*. Pada saat ini perusahaan-perusahaan sudah banyak meng*update* aplikasi lama yang dibuat *Microsoft Visual Basic* 6.0 ke teknologi. *Net* karena kelebihan-kelebihan

yang ditawarkan, terutama memungkinkan pengembang perngkat lunak secara cepat mampu membuat program *robust*, serta berbasiskan integrasi ke internet yang dikenal dengan *XML Web Service* (A.M.Hirin, 2010)

Untuk melihat tampilan visual basic 2010 dapat dilihat pada gambar II.5. sebagai berikut :



Gambar II.1. Tampilan Utama Visual Basic 2010

Sumber : (A. M. Hirin, 2011)

II.8. *SQL Server 2008*

SQL Server 2008 adalah sebuah RDBMS (*Relational Database Management System*) yang di-develop oleh Microsoft, yang digunakan untuk menyimpan dan mengolah data. Pada *SQL Server 2008*, kita bisa melakukan pengambilan dan modifikasi data yang ada dengan cepat dan efisien. Pada *SQL*

Server 2008, kita bisa membuat objek-objek yang sering digunakan pada aplikasi bisnis, seperti membuat database, *table, function, stored database, trigger* dan *view*. Selain objek, kita juga menjalankan perintah Sql (*Structured Query Language*) untuk mengambil data. (Elex Media Competindo;2010:101)