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. Pengeritan Aplikasi

Program aplikasi adalah program siap pakai atau program yang direka untuk melaksanakan suatu fungsi bagi pengguna atau aplikasi yang lain. Aplikasi juga diartikan sebagai penggunaanatau penerapan suatu konsep yang menjadi pokok pembahasan atau sebagai program komputer yang dibuat untuk menolong manusia dalam melaksanakan tugas tertentu. Aplikasi *software* yang dirancang untuk penggunaan praktisi khusus, klasifikasi luas ini dapat dibagi menjadi 2 (dua), yaitu :

1. Aplikasi *software* spesialis, program dengan dokumentasi tergabung yang dirancang untuk menjalankan tugas tertentu.
2. Aplikasi paket, suatu program dengan dokumentasi tergabung yang dirancang untuk jenis masalah tertentu (Rahmatillah ; 2011: 3).

II.3. Pengertian Sistem

Sistem informasi berbasis komputer merupakan sekelompok perangkat keras dan perangkat lunak yang dirancang untuk mengubah data menjadi informasi yang bermanfaat. Jenis sistem informasi berbasis komputer, yaitu :

1. Pengolahan Data. Pengolahan data elektronik – *electronic dataprocessing (EDP)* adalah pemanfaatan teknologi komputeruntuk melakukan pengolahan data transaksi-transaksi dalamsuatu organisasi. EDP adalah aplikasi sistem informasi akuntansi paling dasar dalam setiap organisasi. Sehubungan dengan perkembangan teknologi komputer, istilah pengolahandata mulai dikenal dan mempunyai arti yang sama denganistilah EDP.
2. Sistem Informasi Manajemen (SIM), menguraikan penggunaanteknologi komputer untuk menyediakan informasi bagipengambilan keputusan para manajer.
3. Sistem Pendukung Keputusan – *Decision Support Systems(*DSS). DSS diarahkan untuk melayani permintaan informasitertentu, khusus, dan tidak rutin dari manajemen. Contohadalah penggunaan *spreadsheet* untuk melakukan analisis“*what if*” dari data operasi atau anggaran.

4. Sistem Pakar – *expert systems* (ES) adalah sistem informasiberbasis pengetahuan yang memanfaatkan pengetahuannyatentang bidang aplikasi tertentu untuk bertindak sepertiseorang konsultan ahli bagi pemakainya.
5. Sistem Informasi Eksekutif – *executive information systems*(EIS). EIS dibuat bagi kebutuhan informasi stratejik manajementingkat puncak.
6. Sistem Informasi Akuntansi – sistem berbasis komputer yangdirancang untuk mengubah data akuntansi menjadi informasi (Agustinus ; 2012 : 2).

II.3.1. Karekteristik Sistem

Supaya sistem itu dikatakan sistem yang baik memiliki karakteristik sistem yaitu :

1. Komponen Sistem

Suatu sistem terdiri 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 (*boundry*)

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 (*environment*)

Lingkungan luar sistem (*environment*) adalah diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkung luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut.

4. Penghubung Sistem (*interface*)

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

5. Masukan Sistem (*input*)

Masukan sistem adalah energi yang dimasukkan ke dalam sistem. Masukan dapat berupa masukan perawatan (*maintanceinput*) dan masukan sinyal (*signalinput*).

6. Keluaran Sistem (*output*)

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

7. Pengolah Sistem

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

8. Sasaran Sistem

Suatu sistem mempunyai tujuan (*goal*) atau sasaran (*objective*). Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak ada gunanya (Jeperson Hutahaean ; 2014 : 2).

II.3.2. Klasifikasi Sistem

Sistem dapat diklasifikasikan dalam beberapa sudut pandang seperti berikut :

1. Sistem abstrak (*abstract system*). Sistem abstrak adalah sistem yang berupa pemikiran-pemikiran atau ide-ide yang tidak tampak secara fisik.
2. Sistem fisik (*physical system*). Sistem fisik adalah sistem yang ada secara fisik.
3. Sistem alamiyah (*natural system*). Sistem alamiyah adalah sistem yang terjadi melalui proses alam, tidak dibuat oleh manusia.
4. Sistem buatan manusia (*human made system*). Sistem buatan manusia adalah sistem yang dibuat oleh manusia yang melibatkan interaksi antara manusia dengan mesin.
5. Sistem tertentu (*deterministicl system*). Sistem tertentu adalah sistem yang beroperasi dengan tingkah laku yang sudah dapat diprediksi, sebagai keluaran sistem yang dapat diramalkan.
6. Sistem tak tentu (*probalistic system*). Sistem tak tentu adalah sistem yang kondisi masa depannya tidak dapat diprediksi karena mengandung unsur probabilistik.
7. Sistem tertutup (*close system*). Sistem tertutup adalah sistem yang baik terpengaruh dan tidak berhubungan dengan lingkungan luar, sistem bekerja otomatis tanpa ada turut campur lingkungan luar.

8. Sistem terbuka (*open system*). Sistem terbuka adalah sistem yang berhubungan dan terpengaruh dengan lingkungan luarnya. Sistem ini menerima *input* dan *output* dari lingkungan luar atau subsistem lainnya (Jeperson Hutahaean ; 2014 : 6).

II.4. Pengertian Payroll

Payroll sering diartikan sebagai jumlah total yang dibayarkan kepada karyawan untuk suatu periode tertentu. Alasan-alasan pentingnya gaji dan upah adalah sebagai berikut :

1. Karyawan sangat sensitif terhadap kesalahan-kesalahan dalam penggajian atau terhadap hal-hal yang tidak wajar.
2. Pengeluaran gaji dan upah adalah hal yang diatur oleh peraturan pemerintah daerah dan pemerintah pusat.
3. Jumlah pengeluaran gaji serta pajak gaji dan upah yang berkaitan akan memberikan pengaruh penting terhadap laba bersih untuk kebanyakan perusahaan.

Gaji biasanya digunakan untuk pembayaran atas jasa manajemen, administratif, atau jasa-jasa yang serupa, tingkat gaji biasanya dinyatakan dalam satuan bulan atau tahun sedangkan upah biasanya digunakan untuk imbalan karyawan lapangan (pekerja kasar), baik yang terdidik, maupun yang tak terdidik, dan didasarkan pada jam kerja, mingguan, atau borongan (Wibowo ; 2010 : 3).

II.4.1. Pengertian Gaji

Gaji adalah bentuk balas jasa atau penghargaan yang diberikan secara teratur kepada seorang pegawai atas jasa dan hasil kerjanya. Gaji seraing juga disebut sebagai upah, di mana keduanya merupakan suatu bentuk kompensasi, yakni imbalan jasa yang diberikan secara teratus atas prestasi kerja seorang pegawai. Perbedaan gaji dan upan hanya terletak pada kuatnya ikatan kerja dan jangka waktu penerimaannya.

Berdasarkan Kepmenakertrans No. Kep.49/MEN/2004 tentang Ketentuan Struktur dan Skala Upah, Menyatakan bahwa struktur upah dapat didasarkan hal-hal seperti berikut :

1. Struktur organisasi
2. Rasio perbedaan bobot pekerjaan antar jabatan
3. Kemampuan perusahaan
4. Upah minuman
5. Kondisi pasar

Keputusan tersebut harus menjadi dasar utama dalam menentukan upah oleh perusahaan. Berikut beberapa tip mempertimbangkan penentuan upah pekerja.

1. Perhatikan standar hidup minimal (sandang, pangan dan papan). Sebagai pengusaha Anda wajib memenuhi standar tersebut. standar tersebut berbeda di tiap wilayah, jadi sebaiknya anda teliti biaya hidup standar minimal di tempat usaha anda didirikan, sesuaikan dengan upah minimun yanng berlaku di tempat anda.

2. Bobot pekerjaan bisa menjadi komponen pengukuran ketika anda akan menentukan gaji. Bobot ini diukur berdasarkan *know how*, *problem solving* dan *accountability*.
3. Gaji bisa juga ditentukan dari kemampuan perusahaan untuk memberikan gaji kepada karyawannya. Hal ini penting supaya gaji karyawan tidak membuat beban yang terlalu berat.
4. Tingkat *competitiveness* dapat menjadi ukuran dalam menentukan besaran gaji. Anda bisa bertanya kepada orang lain atau kepada perusahaan yang sejenis dalam memberikan gaji kepada karyawannya.
5. Pengalaman kerja seseorang harus dihargai secara wajar dan proporsional karena pengalaman kerja akan membantu produktivitas anda dalam bekerja.
6. Sejauh mana daya tawar pegawai terhadap perusahaan harus dipertimbangkan, begitu pula sebaliknya. Sejauh mana perusahaan mempunyai daya tawar terhadap pegawai (Eka An Aqinuddin ; 2010 : 174).

II.5. Pengertian Java

Java Language Spesification adalah definisi teknis dari bahasa pemrograman *Java* yang di dalamnya terdapat aturan penulisan sintaks dan semantik *Java.* Api adalah *application programming interface* yaitu sebuah *layer* yang berisi *class-class* yang sudah didefinisikan dan antarmuka pemogramam yang akan membantu para pengembang aplikasi dalam perancangan sebuah aplikasi. API memampukan para pengembang untuk dapat mengakses fungsi-

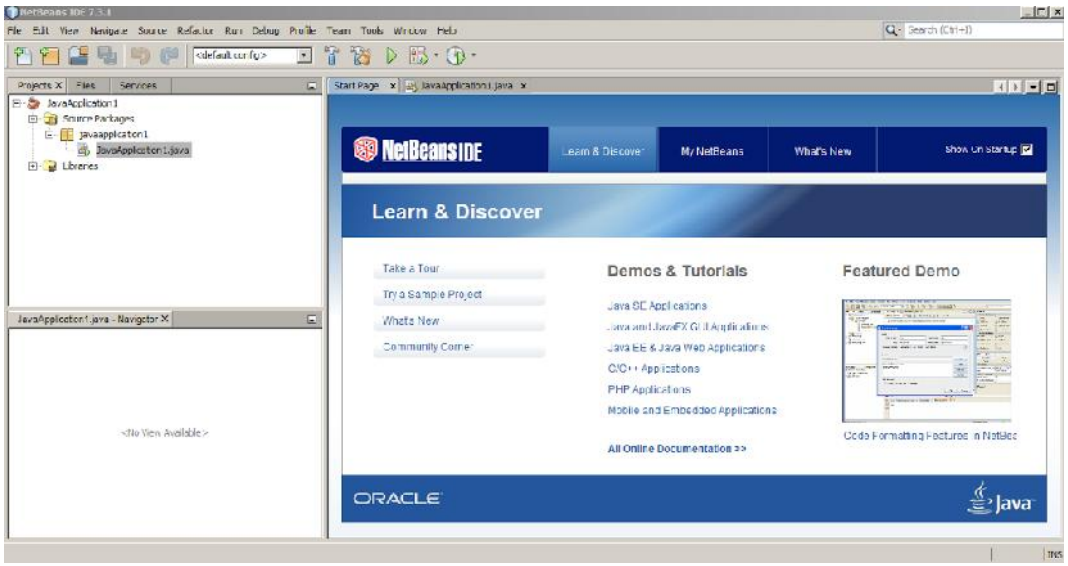
fungsi sistem operasi yang diizinkan melalui bahasa Java. Pada saat ini dikenal ada tiga buah API dari Java, yaitu :

- 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).

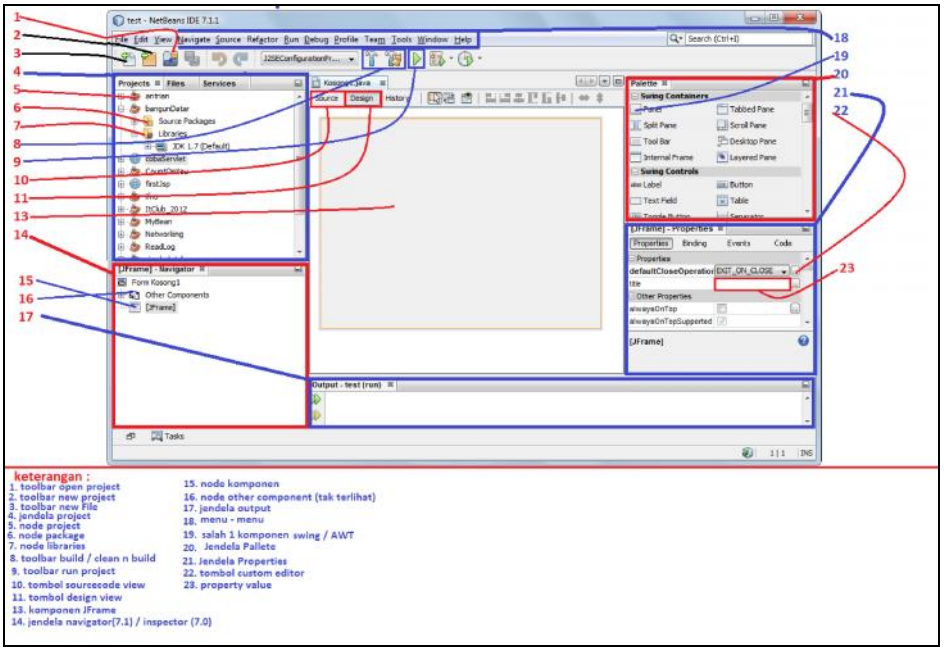
II.6. Pengertian NetBeans

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

NetBeans juga menyediakan paket yang lengkap dalam pemrograman dari pemrograman standar (aplikasi deskitop), pemrograman *enterprise*, dan pemrograman perangkat *mobile*. Saat ini NetBeans telah mencapai versi 6.8. (Wahana Komputer ; 2010 : 15). Tampilan Netbeans dan tools pada netbeans dapat dilihat pada gambar II.1 dan II.2 berikut :



Gambar II.1. Tampilan Netbeans
(Sumber : Wahana Komputer ; 2010 : 15)



Gambar II.2. Tampilan Tools Netbeans
(Sumber : Wahana Komputer ; 2010 : 15)

II.7. Pengertian *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. 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 *harddisk.* 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 banktelah 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 (Agustinus Mujilan ; 2012 : 23).

II.8. Pengertian MySQL

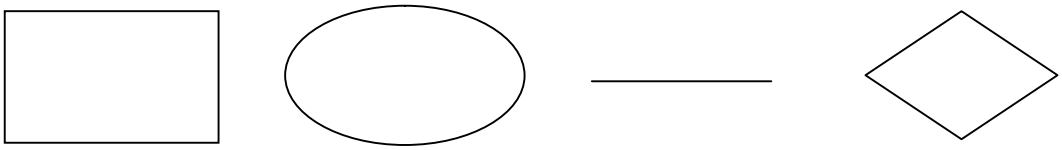
Mysql pertama kali dirintis oleh seorang *programmerdatabase* bernama Michael Widenius, yang dapat anda hubungi di emailnya monty@analytikerna Mysql *databaseserver* adalah RDBMS (*Relasional Database Management System*) yang dapat menangani data yang bervolume besar. meskipun begitu, tidak menuntut *resource* yang besar. Mysql adalah *database* yang paling popular diantara *database* yang lain.

MySQL adalah program *database* yang mampu mengirim dan menerima data dengan sangat cepat dan *multiuser*. MySQL memiliki dua bentuk lisensi, yaitu *freesoftware* 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 dan windows NT.
- Versi 3.23 : beta dari Juni 2000, dan dirilis pada January 2001.
- Versi 4.0 : beta dari Agustus 2002, dan dirilis pada Maret 2003 (unions) (Wahana Komputer ; 2010).

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

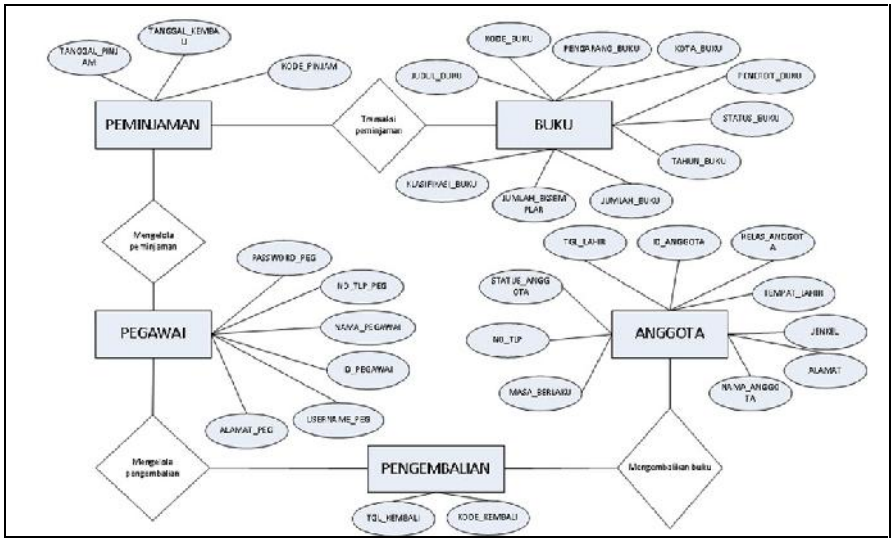
Entity Relationship Diagram atau ERD merupakan salah satu alat (tool) berbentuk grafis yang populer untuk *desain database.* Tool ini relatif lebih mudah dibandingkan dengan Normalisasi. Kebanyakan sistem analis memakai alat ini, tetapi yang jadi masalah, kalau kita cermati secara seksama, tool ini mencapai 2NF (Yuniar Supardi ; 2010 : 448).



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

Gambar. II.3. Bentuk Simbol ERD
(Sumber : Yuniar Supardi ; 2010 : 448)

Adapun contoh dari perancangan *Entity Relationship Diagram* dapat dilihat pada gambar II.4 berikut :



Gambar II.4. Contoh ERD
(Sumber : Yuniar Supardi ; 2010 : 448)

II.10. Kamus Data

Kamus data (data *dictionary*) mencakup definisi-definisi dari data yang disimpan di dalam basis data dan dikendalikan oleh sistem manajemen basis data. Figur 6.5 menunjukkan hanya satu tabel dalam basis data jadwal. Struktur basis data yang dimuat dalam kamus data adalah kumpulan dari seluruh definisi *field*, definisi tabel, relasi tabel, dan hal-hal lainnya. Nama *field* data, jenis data (seperti teks atau angka atau tanggal), nilai-nilai yang valid untuk data, dan karakteristik-karakteristik lainnya akan disimpan dalam kamus data. Perubahan-perubahan pada struktur data hanya dilakukan satu kali di dalam kamus data, program-program palikasi yang mempergunakan data tidak akan ikut terpengaruh (D. Tri Octafian ; 2011:154).

Tabel II.1. Contoh Kamus Data

No	Nama Field	Tipe Data	Panjang
1	id_pelanggan	CHAR	5
2	nama	VARCHAR	30
3	alamat	VARCHAR	60
4	telp	VARCHAR	15

(Sumber : Jurnal Teknologi dan Informatika ; D. Tri Octafian ; 2011:154)

II.11. 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 yaitu sebagai berikut :

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

Contoh yang kita gunakan di sini adalah sebuah perusahaan yang mendapatkan barang dari sejumlah pemasok. Masing-masing pemasok berada pada satu kota. Sebuah kota dapat mempunyai lebih dari satu pemasok dan masing-masing kota mempunyai kode status tersendiri. Contoh normalisasi 1NF adalah seperti pada table berikut :

Tabel 2. Tabel Bentuk Normal Pertama (1NF)

p#	Status	kota	b#	qty
p1	20	Yogyakarta	b1	300
p1	20	Yogyakarta	b2	200
p1	20	Yogyakarta	b3	400
p1	20	Yogyakarta	b4	200
p1	20	Yogyakarta	b5	100
p1	20	Yogyakarta	b6	100
p2	10	Medan	b1	300
p2	10	Medan	b2	400
p3	10	Medan	b2	200
p4	20	Yogyakarta	b2	200
p4	20	Yogyakarta	b4	300

(Sumber : Janner Simarmata ; 2010 : 78)

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

Definisi bentuk normal kedua menyatakan bahwa tabel dengan kunci utama gabungan hanya dapat berada pada 1NF, tetapi tidak pada 2NF. Sebuah tabel relasional berada pada bentuk normal kedua jika dia berada pada bentuk normal

kedua jika dia berada pada 1NF dan setiap kolom bukan kunci yang sepenuhnya tergantung pada seluruh kolom yang membentuk kunci utama.

Tabel 3. Tabel Bentuk Normal Kedua (2NF)

Pemasok2

p#	status	Kota
P1	20	Yogyakarta
P2	10	Medan
P3	10	Medan
P4	20	Yogyakarta
P5	30	Bandung

Barang

p#	b#	qty
p1	b1	300
p1	b2	200
p1	b3	400
p1	b4	200
p1	b5	100
p1	b6	100
p2	b1	300
p2	b2	400
p3	b2	200
p4	b2	200
p4	b4	300
p4	b5	400

(Sumber : Janner Simarmata ; 2010 : 78)

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

Bentuk normal ketiga mengharuskan semua kolom pada tabel relasional tergantung hanya pada kunci utama. Secara definisi, sebuah tabel berada pada bentuk normal ketiga (3NF) jika tabel sudah berada pada 2NF dan setiap kolom yang bukan kunci tidak tergantung secara transitif pada kunci utamanya.

Tabel 4. Tabel Bentuk Normal Ketiga (3NF)

Pemasok Kota

p#	Kota
P1	Yogyakarta
P2	Medan
P3	Medan
P4	Yogyakarta
P5	Bandung

Kota Status

Kota	status
Yogyakarta	20
Medan	10
Yogyakarta	20
Bandung	30

(Sumber : Janner Simarmata ; 2010 : 78)

4. *Boyce Code Normal Form* (BCNF)

Setelah 3NF, semua masalah normalisasi hanya melibatkan tabel yang mempunyai tiga kolom atau lebih dan semua kolom adalah kunci. Banyak praktisi berpendapat bahwa menempatkan entitas pada 3NF sudah cukup karena sangat jarang entitas yang berada pada 3NF bukan merupakan 4NF dan 5NF.

5. Bentuk Normal Keempat (4NF)

Sebuah tabel rasional berada pada bentuk normal keempat (4NF) jika dia dalam BCNF dan semua ketergantungan multivalue merupakan ketergantungan fungsional. Bentuk normal keempat (4NF) didasarkan pada konsep ketergantungan multivalue (MVD). Sebuah ketergantungan multivalue tiga kolom, satu kolom mempunyai banyak baris bernilai sama, tetapi kolom lain bernilai berbeda.

Tabel 5. Tabel Bentuk Normal Keempat (4NF)

Pegawai Proyek

peg#	Pry#
1211	P1
1211	P3

Pegawai Ahli

Peg#	Ahli
1211	Analisis
1211	Perancangan
1211	Pemrograman

(Sumber : Janner Simarmata ; 2010 : 78)

6. Bentuk Normal Kelima

Sebuah tabel berada pada bentuk normal kelima (5NF) jika ia tidak dapat mempunyai dekomposisi lossless menjadi sejumlah tabel lebih kecil. Empat bentuk normal pertama berdasarkan pada konsep ketergantungan fungsional, sedangkan bentuk normal kelima berdasarkan pada konsep ketergantungan gabungan (*join dependence*) (Janner Simarmata ; 2010 : 78).

Tabel 6. Tabel Bentuk Normal Kelima (5NF)

peg#	Pry#	Ahli
1211	11	Perancangan
1211	28	Pemrograman

(Sumber : Janner Simarmata ; 2010 : 78)

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

Menurut Windu Gata (2013 : 4) Hasil pemodelan pada OOAD terdokumentasikan dalam bentuk *Unified Modeling Language* (UML). UML adalah bahasa spesifikasi standar yang dipergunakan untuk mendokumentasikan, menspesifikasikan dan membangun perangkat lunak.

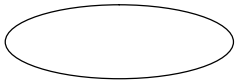
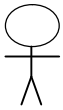
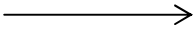
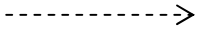
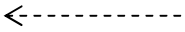
UML merupakan metodologi dalam mengembangkan sistem berorientasi objek dan juga merupakan alat untuk mendukung pengembangan sistem. UML saat ini sangat banyak dipergunakan dalam dunia industri yang merupakan standar bahasa pemodelan umum dalam industri perangkat lunak dan pengembangan sistem.

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

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.7. Simbol *Use Case*

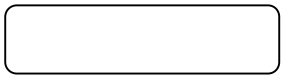
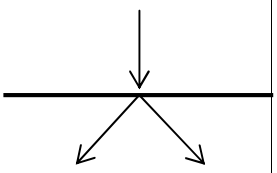
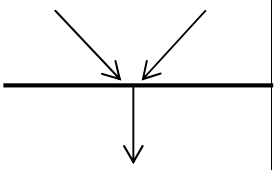
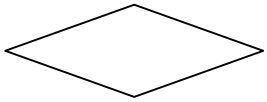
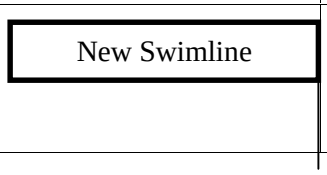
Gambar	Keterangan
	*Use case* menggambarkan fungsionalitas yang disediakan sistem sebagai unit-unit yang bertukan pesan antar unit dengan aktor, biasanya dinyatakan dengan menggunakan kata kerja di awal nama *use case*.
	Aktor adalah *abstraction* dari orang atau sistem yang lain yang mengaktifkan fungsi 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.8. Simbol *Activity Diagram*

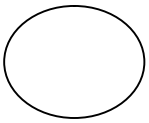
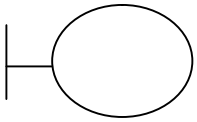
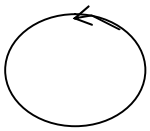
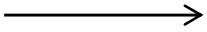
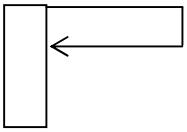
Gambar	Keterangan
	*Start point*, diletakkan pada pojok kiri atas dan merupakan awal aktifitas.
	*End point*, akhir aktifitas.
	*Activites*, menggambarkan suatu proses/kegiatan bisnis.
	*Fork* (Percabangan), digunakan untuk menunjukkan kegiatan yang dilakukan secara parallel atau untuk menggabungkan 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. 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.9. Simbol *Sequence Diagram*

Gambar	Keterangan
	*EntityClass*, merupakan bagian dari sistem yang berisi kumpulan kelas berupa entitas-entitas yang membentuk gambaran awal sistem dan menjadi landasan untuk menyusun basis data.
	*Boundary Class*, berisi kumpulan kelas yang menjadi *interface* atau interaksi antara satu atau lebih aktor dengan sistem, seperti tampilan formentry dan *form* cetak.
	*Control class*, suatu objek yang berisi logika aplikasi yang tidak memiliki tanggung jawab kepada entitas, contohnya adalah kalkulasi dan aturan bisnis yang melibatkan berbagai objek.
	*Message*, simbol mengirim pesan antar *class*.
	*Recursive*, menggambarkan pengiriman pesan yang dikirim untuk dirinya sendiri.
	*Activation, activation* mewakili sebuah eksekusi operasi dari objek, panjang kotak ini berbanding lurus dengan durasi aktivitas sebuah operasi.
	Lifeline, garis titik-titik yang terhubung dengan objek, sepanjang *lifeline* terdapat *activation*.

(Sumber : Windu Gata ; 2013 : 7)

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 II.10. *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)