BAB 2

LANDASAN TEORI

2.1 Teori-Teori Basis Data

2.1.1 Data

Menurut Whitten, Bentley, dan Dittman (2004, p23), data adalah fakta mentah mengenai orang, tempat, kejadian, dan hal-hal penting dalam organisasi.Tiap fakta, dengan sendirinya ,secara relatif tidak ada artinya.

Menurut Turban (2003, p17), data adalah suatu fakta atau deksripsi dasar dari sesuatu, kejadian, aktivitas, dan transaksi yang diperoleh, disimpan, direkam, diklasifikasikan, tetapi belum memberikan manfaat khusus bagi penggunanya.

Jadi dapat disimpulkan bahwa data adalah suatu fakta dari suatu, kejadian, aktivitas, dan transaksi yang diciptakan, disimpan, diklasifikasikan oleh sistem informasi tetapi belum memberikan manfaat khusus bagi penggunanya.

2.1.2 Definisi *Database*

Menurut Atzeni, Ceri, Paraboschi, dan Torlone *Database* (2005, p2) adalah kumpulan data yang digunakan untuk menggambarkan informasi penting menjadi sistem informasi.

Menurut Kroenke *Database* (2007, p11) adalah menggabarkan kumpulan dari table-table yang terintegrasi. *Table-table* yang terintegrasi adalah *table* yang menyimpan data dan hubungan-hubungan antar data.

Menurut Hoffer, Prescott, dan Mc Fadden *database* adalah kumpulan-kumpulan data yang terorganisir yang terhubung secara logis. *Database* dapat berbeda-beda bentuk dan kompleksitas.

Menurut Connolly dan Begg (2010, p65), pengertian *database* adalah kumpulan data yang terhubung secara logis yang dipakai bersama dan

deskripsi dari data ini dirancang untuk memenuhi kebutuhan informasi sebuah organisasi.

Menurut O'Brien (2003, p145) *database* adalah sebuah kumpulan yang terintegrasi dari elemen data yang terhubung secara logikal. Elemen data mendeskripsikan entiti-entiti dan hubungan antara entiti-entiti.

Jadi *database* adalah suatu sistem penyimpanan data yang tersusun atas sekumpulan data yang secara logika saling terkait yang dirancang untuk memenuhi kebutuhan informasi perusahaan. Model *database* relasional adalah sistem yang banyak digunakan karena struktur logikalnya yang sederhana. Pada model relasional seluruh data disusun secara logikal dalam relasi-relasi atau tabel. Setiap relasi terdiri dari baris, dan kolom dari relasi yang diberi nama tertentu disebut atribut. Sedangkan baris dari relasi disebut *tuple* dan setiap *tuple* (baris) memiliki satu nilai untuk setiap attribut.

2.1.3 *Database Management System* (DBMS)

2.1.3.1 Definisi DBMS

Menurut Connolly dan Begg (2010, p66), pengertian DBMS adalah sebuah sistem piranti lunak yang memungkinkan user untuk mendefinisikan, membuat, menjaga, dan mengontrol akses ke dalam basis data.

2.1.3.2 Tujuan DBMS

Tujuan utama pengolahan data dalam basis data adalah agar dapat memperoleh data yang dicari dengan mudah dan cepat. Pemanfaatan basis data dilakukan untuk memenuhi sejumlah tujuan seperti berikut ini :

1. Kecepatan dan kemudahan (*speed*)
2. Efisiensi ruang penyimpanan (*space*)
3. Keakuratan (*accuracy*)
4. Ketersediaan (*availability*)

5. Kelengkapan (*completeness*)
6. Keamanan (*security*)
7. Kebersamaan pemakai (*sharebility*)

2.1.3.3 Komponen – Komponen DBMS

Menurut Connolly dan Begg (2010, p68), *Database Management System* (DBMS) memiliki 5 komponen penting, yaitu:

1. *Hardware* (Perangkat Keras)

   Dalam menjalankan aplikasi dan DBMS diperlukan perangkat keras. Perangkat keras dapat berupa*a single personal computer, single mainframe*, sampai jaringan komputer. Perangkat keras yang digunakan bergantung pada persyaratan dari organisasi dan DBMS yang digunakan.

2. *Software* (Perangkat Lunak)

   Komponen perangkat lunak meliputi DBMS *software* dan program aplikasi beserta Sistem Operasi, termasuk perangkat lunak tentang jaringan bila DBMS digunakan dalam jaringan seperti LAN (*Local Area Network).*

3. Data

   Data merupakan komponen terpenting dari DBMS dan juga merupakan komponen penghubung antara komponen mesin (*Hardware* dan *Software*) dan komponen *human* (*Procedures* dan *People*).

4. Prosedur

   Prosedur merupakan panduan dan instruksi dalam membuat desain dan menggunakan basis data Penggunaan dari sistem dan staf dalam mengelola basis data membutuhkan prosedur dalam menjalankan sistem dan mengelola basis data itu sendiri. Prosedur di dalam basis data dapat berupa: *login* di dalam basis data, penggunaan sebagian fasilitas DBMS, cara menjalankan dan

memberhentikan DBMS, membuat salinan *backup database*, memeriksa *hardware* dan *software* yang sedang berjalan, mengubah struktur basis data, meningkatkan kinerja atau membuat arsip data pada media penyimpanan sekunder.

5. Manusia

Komponen terakhir yaitu manusia sendiri yang terlibat dalam sistem tersebut. Komponen ini meliputi data dan *database administrator, database designers, application developers*, dan *end-users.*

2.1.3.4 Keuntungan dan Kerugian DBMS

Menurut Connolly dan Begg (2010, p77), keuntungan DBMS adalah sebagai berikut:

1. Mengontrol redudansi data
2. Mendapat informasi yang lebih dari jumlah data yang sama
3. Peningkatan integritas data
4. Peningkatan produktifitas
5. Peningkatan keamanan serta layanan *backup* dan *recovery*

Menurut Connolly dan Begg (2010, p80), kerugian DBMS adalah sebagai berikut :

1. Kompleksitas
2. Ukuran
3. Biaya dari DBMS
4. Biaya tambahan perangkat keras
5. Biaya proses konversi
6. Performa
7. Pengaruh kegagalan yang lebih tinggi

2.1.4 *Database Language*

2.1.4.1 *Data Definition Language* (DDL)

Menurut Connolly dan Begg (2010, p92), pengertian *Data Definition Language* adalah suatu bahasa yang memperbolehkan *Database Administrator* (DBA) atau pengguna untuk mendeskripsikan dan memberi nama suatu entitas, atribut, dan relasi data yang dibutuhkan untuk aplikasi, bersama dengan integritas data yang diasosiasikan dan batasan (*constraint*) keamanan data.

2.1.4.2 *Data Manipulation Language* (DML)

Menurut Connolly dan Begg (2010, p92), pengertian *Data Manipulation Language* adalah suatu bahasa yang menyediakan seperangkat operasi untuk mendukung manipulasi data yang berada pada basis data.

Pengoperasian data yang akan dimanipulasi biasanya meliputi :

1. Penambahan data baru ke dalam basis data.
2. Modifikasi data yang disimpan ke dalam basis data.
3. Pengembalian data yang terdapat di dalam basis data.
4. Penghapusan data dari basis data.

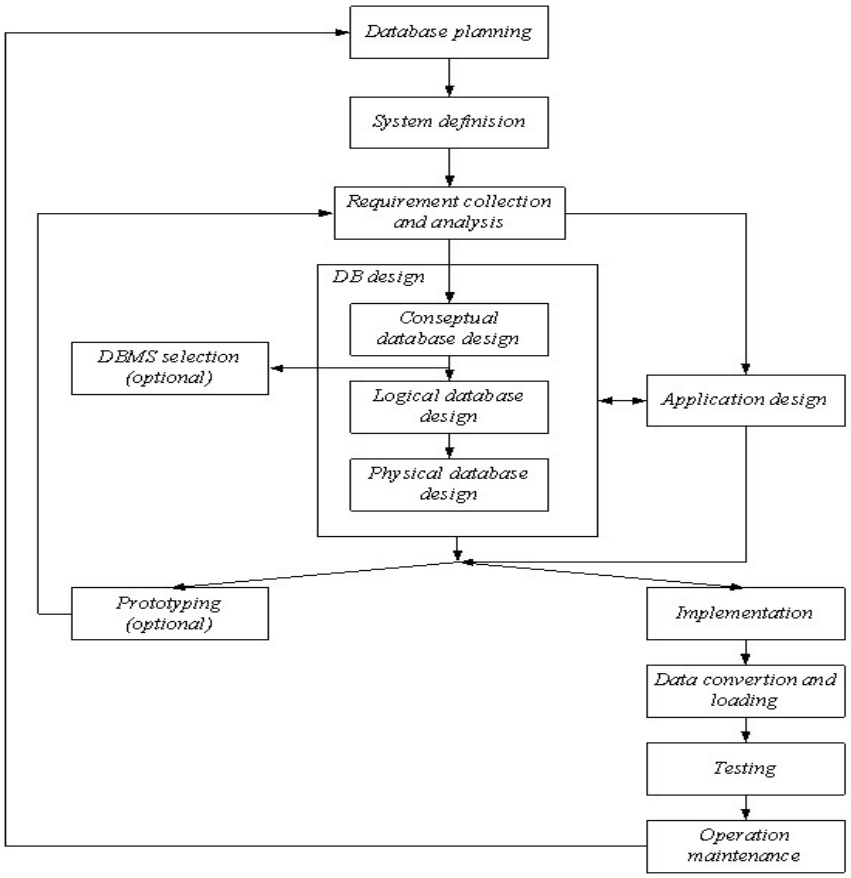
DML dibagi menjadi 2 jenis yaitu *Procedural* dan *Non-procedural.* Menurut Connolly dan Begg (2010, p92), pengertian *Procedural DML* adalah suatu bahasa yang memperbolehkan pengguna untuk mendeskripsikan ke sistem data apa yang dibutuhkan dan bagaimana mendapatkan data tersebut secara tepat, sedangkan *Non-procedural DML* adalah sebuah bahasa yang mengizinkan pengguna untuk menentukan data apa yang dibutuhkan tanpa memperhatikan bagaimana data diperoleh.

2.1.5 *Database Lifecycle*

Menurut Connoly dan Begg (2010, p313), sebuah sistem *database* merupakan komponen dasar sistem informasi organisasi yang lebih besar sehingga siklus hidup aplikasi *database* berhubungan dengan

siklus hidup sistem informasi. Tahapan-tahapan siklus hidup aplikasi adalah sebagaimana terlihat pada gambar berikut :

Gambar 2.1 *Database Lifecycle*
(Connoly, 2010, p314)



2.1.5.1 Definisi Sistem

Menurut Connolly dan Begg (2010,p316), sistem adalah menggambarkan lingkup dan batasan-batasan dari aplikasi basis data dan *user view* yang utama. Sebelum mencoba merancang suatu aplikasi basis data diperlukan untuk mengenali batasan sistem dan bagaimana antarmuka dengan bagian sistem informasi lainnya dalam organisasi.

Menurut O'Brien (2003, p8) sistem adalah kumpulan elemen yang saling terhubung atau berinteraksi membentuk suatu kesatuan atau sekumpulan komponen yang saling terhubung dan bekerja sama

untuk mencapai sasaran dengan menerima *input* dan menghasilkan *output* dalam sebuah proses transformasi yang teroganisir.

Dari pendapat-pendapat di atas dapat disimpulkan sistem adalah kumpulan unsur-unsur yang berhubungan untuk melaksanakan kegiatan- kegiatan perusahaan dalam mencapai suatu tujuan tertentu.

2.1.5.2 Pengumpulan dan Analisis Kebutuhan

Menurut Connolly dan Begg (2010, p316), pengumpulan dan analisis kebutuhan adalah proses dari analisis dan pengumpulan informasi tentang bagian organisasi yang didukung oleh sistem aplikasi basis data dan menggunakan informasi ini untuk mengenali kebutuhan-kebutuhan untuk sistem baru. Pengumpulan dan analisis kebutuhan adalah tahapan persiapan merancang basis data. Jumlah data yang dikumpulkan tergantung pada masalah alamiah dan kebijakan suatu perusahaan. Beberapa teknik atau cara untuk mendapatkan informasi adalah dengan teknik *Fact Findng*. *Fact Finding* adalah teknik yang digunakan untuk mengidentifikasikan kebutuhan.

2.1.5.3 Metodologi Perancangan Basis Data

Menurut Connolly dan Begg (2010, p466), metodologi perancangan basis data adalah suatu pendekatan terstruktur yang menggunakan prosedur, teknik, alat-alat, dan bantuan dokumentasi untuk mendukung dan memfasilitasi proses perancangan. Menurut Connolly dan Begg (2010, p467) proses perancangan terdiri dari tiga bagian, yaitu:

1. Perancangan Basis Data Konseptual

   Perancangan basis data konseptual adalah proses membangun suatu model informasi yang digunakan suatu perusahaan, yang berdiri sendiri terhadap semua pertimbangan fisikal.

2. Perancangan Basis Data Logikal

   Perancangan basis data logikal adalah proses membangun model informasi yang digunakan dalam suatu perusahaan

berdasarkan pada spesifik data model, tetapi berdiri sendiri terhadap semua fakta-fakta DBMS dan pertimbangan fisikal lainnya.

3. Perancangan Basis Data Fisikal

Perancangan basis data fisikal adalah proses menghasilkan satu deskripsi mengenai implementasi basis data pada media penyimpanan sekunder, menggambarkan dasar relasi, *file* organisasi, dan *indeks-indeks* yang digunakan untuk mencapai efisiensi akses terhadap data, dan semua integritas *constraint* dan pengukuran keamanan

2.1.5.4 Seleksi DBMS

Menurut Connolly dan Begg (2010, p325), pengertian seleksi DBMS adalah menyeleksi DBMS yang tepat untuk mendukung aplikasi basis data. Seleksi DBMS dilakukan antara tahapan perancangan database logikal dan perancangan database fisikal. Tujuannya untuk kecukupan sekarang dan kebutuhan masa mendatang pada perusahaan, membuat keseimbangan biaya termasuk pembelian produk DBMS, piranti lunak untuk mendukung aplikasi basis data, biaya yang berhubungan dengan perubahan dan pelatihan pegawai.

2.1.5.5 Perancangan Aplikasi

Menurut Connolly dan Begg (2010, p329), pengertian perancangan aplikasi adalah merancang antarmuka pemakai dan program aplikasi, yang akan memproses basis data. Perancangan basis data dan aplikasi merupakan aktivitas yang dilakukan secara bersamaan pada *database application life cycle*.

2.1.5.6 *Prototyping*

Menurut Connolly dan Begg (2010, p333), pengertian *prototyping* adalah membuat model kerja dari aplikasi basis data. Tujuannya adalah untuk memungkinkan pemakai menggunakan *prototype* untuk mengidentifikasikan fitur-fitur sistem berjalan dengan baik atau tidak,

dan bila memungkinkan untuk menyarankan peningkatan atau bahkan penambahan fitur-fitur baru ke dalam sistem database.

Ada dua macam strategi *prototyping* yang digunakan sekarang :

1. *Prototyping* Kebutuhan (*Requirement Prototyping*)

   Menggunakan suatu *prototype* untuk menetapkan kebutuhan dari tujuan aplikasi basis data dan ketika kebutuhan sudah terpenuhi, *prototype* tidak digunakan lagi atau dibuang.

2. *Prototyping Evolusioner* (*Evolutionary Prototyping*)

   Prototype Evolusioner digunakan dengan tujuan yang sama. Perbedaan yang penting adalah bahwa *prototype* tidak dibuang tetapi dengan mengembangkan lebih lanjut menjadi aplikasi basis data yang dikerjakan.

2.1.5.7 Implementasi

Menurut Connolly dan Begg (2010, p333), pengertian implementasi adalah realisasi fisik suatu basis data dan perancangan aplikasi. Implementasi basis data dapat dicapai menggunakan *Data Definition Language* (DDL) dari DBMS yang dipilih atau *Graphical User Interface* (GUI). Pernyataan DDL digunakan untuk menciptakan struktur-struktur basis data dan *file-file* basis data yang kosong. Semua spesifikasi *user view* juga diimplementasikan pada tahap ini.

2.1.5.8 *Data Conversion And Loading*

Menurut Connolly dan Begg (2010, p334), pengertian *data convertion and loading* adalah mentransfer semua data yang telah ada ke dalam basis data yang baru dan mengkonversi semua aplikasi yang ada untuk dijalankan pada basis data yang baru. Tahap ini hanya dibutuhkan ketika sistem basis data yang baru menggantikan sistem basis data yang lama.

2.1.5.9 Pengujian

Menurut Connolly dan Begg (2010, p334), pengertian pengujian adalah proses menjalankan program aplikasi dengan maksud untuk mencari kesalahan. Sebelum digunakan, aplikasi basis data yang baru dikembangkan harus diuji secara menyeluruh. Untuk mencapainya harus hati-hati dalam menggunakan perencanaan strategi uji dan menggunakan data asli untuk semua proses penguji.

.

2.1.5.10 Operasional dan Pemeliharaan

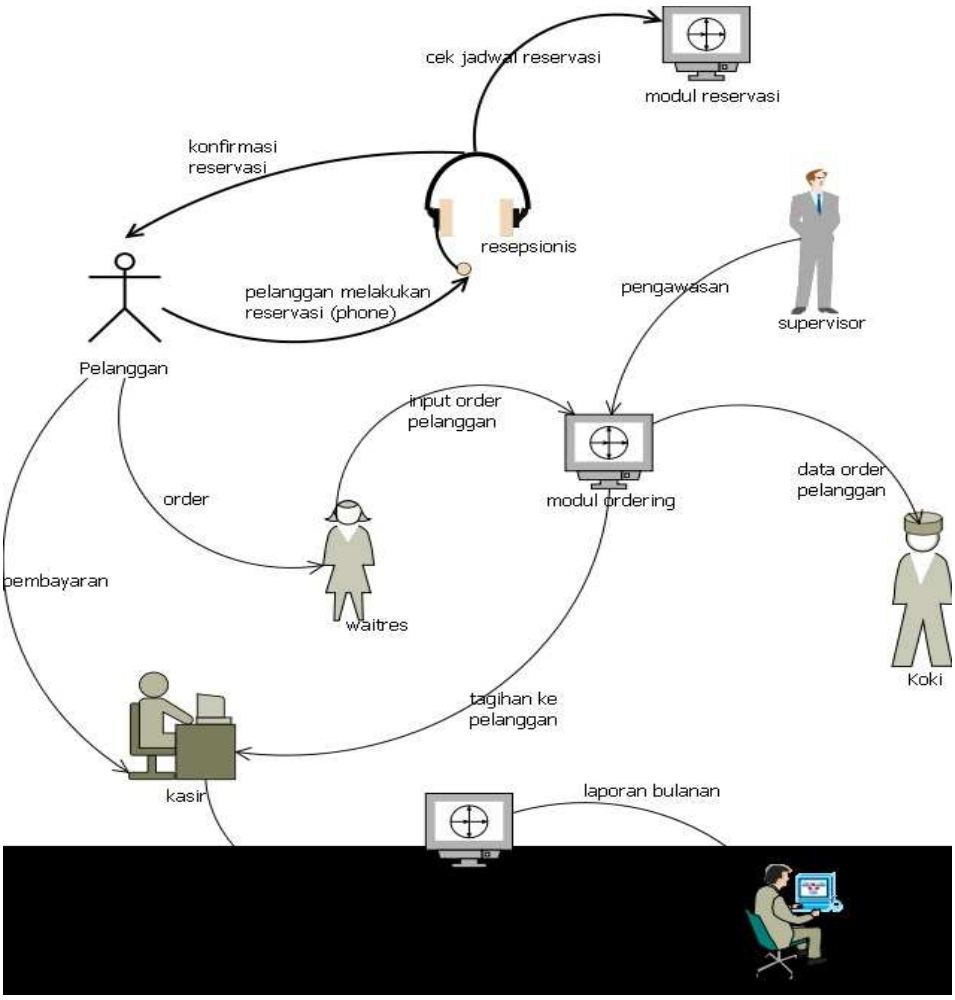
Menurut Connolly dan Begg (2010, p335), pengertian operasional dan pemeliharaan adalah proses memonitor dan memelihara sistem yang telah di-*install*.

2.2 *Rich Picture*

Rich picture adalah penggambaran sistem atau situasi dengan menggunakan gambar-gambar. Gambaran keseluruhan dari orang, objek, proses, struktur, dan masalah pada keseluruhan proses bisnis yang ada di perusahaan

Rich picture digunakan untuk menggambarkan keseluruhan proses bisnis secara jelas dengan gambar dan hubugan antar gambar tersebut dengan penjelasan singkat agar orang yang melihat dapat dengan mudah untuk mengerti dan memahami maksud dari gambar tersebut

. **Gambar 2.2** Contoh *Rich picture*



2.3 Diagram Aliran Data (DFD)

DFD adalah suatu teknik analisa struktur dimana *system analyst* bisa meletakkan semua representasi grafis dari proses data yang melalui organisasi (Kendall,2005, p192). DFD mampu menggambarkan sistem informasi secara logikal maupun fisikal. DFD memiliki empat jenis simbol, yaitu : aliran data (*data flow*),penyimpanan data (*data store*), proses (*proccess*), sumber (*source*).

Menurut Whitten, et.al (2004, p334*), "Data Flow Diagram* (DFD) *is a tool that depicts the flow of data through a system and the work or processing performed by that system"* , dapat diartikan sebagai DFD adalah sebuah alat yang menggambarkan aliran data sampai sebuah sistem selesai dan kerja atau proses dan dilakukan dalam sistem tersebut.

Empat Komponen DFD,antara lain :

1. *External agent*

Menurut Whitten, et.al (2004 , p363), *"External agents define a person, an organization unit, another system or another organization that lies outside to scope of the project but that interacts with the system being studied"*, dapat diartikan sebagai *external agent* mendefinisikan orang, sebuah unit organisasi, sistem lain atau organisasi lain yang berada di luar sistem projek tetapi yang mempengaruhi kerja sistem.

2. *Process*

Menurut Whitten, et.al (2004, p347), proses adalah penyelenggaraan kerja atau jawaban, datang nya aliran data atau kondisinya. Menurut Whitten, et.al (2004, p347) ada beberapa bentuk proses diantaranya :

a. Bentuk *Gane dan Sarson*

b. Bentuk *De Marco/Yourdon*

c. Bentuk *SSADM/IDEFO*

3. *Data stores*

Menurut Whitten, et.al (2004, p366) *Data stores* adalah tempat penyimpanan data.

2.3.1 Jenis-jenis DFD

Jenis-jenis DFD antara lain:

1. Level 0 (Diagram konteks) Level ini terdapat sebuah proses yang berada di posisi pusat.
2. Level 1 (Diagram Nol) Level ini merupakan sebuah proses yang terdapat di level nol yang dipacahkan menjadi beberapa proses lainnya.
3. Level 2 (Diagram Rinci) Pada level ini merupakan diagram yang merincikan diagram dari level 1

- Tanda ‘*’ digunakan hanya jika proses tersebut tidak bisa dirincikan lagi.

  Contoh : 2.0*, artinya proses level rendah tidak bisa ririncikan lagi.

- Penomoran yang dilakukan berdasarkan urutan proses.

2.4 Konsep *Object-Oriented Analysis and Design* (OOAD)

Menurut Satzinger, Jackson, dan Burd (2009, p60), *Object Oriented Analysis* (OOA) adalah suatu cara untuk menentukan semua tipe *object* yang bekerja di dalam sebuah sistem dan menggambarkan interaksi yang dibutuhkan *user* untuk menyelesaikan tugas-tugasnya.

Menurut Satzinger, Jackson, dan Burd (2009, p60), *Object Oriented Design* (OOD) adalah suatu cara untuk menentukan semua tipe *object* yang harus berkomunikasi dengan orang-orang dan perangkat di dalam sistem, menggambarkan bagaimana *object-object* berinteraksi untuk menyelesaikan tugasnya dan memperbaiki definisi masing-masing tipe dari *object* sehingga dapat diimplementasikan dengan sebuah bahasa atau lingkungan khusus.

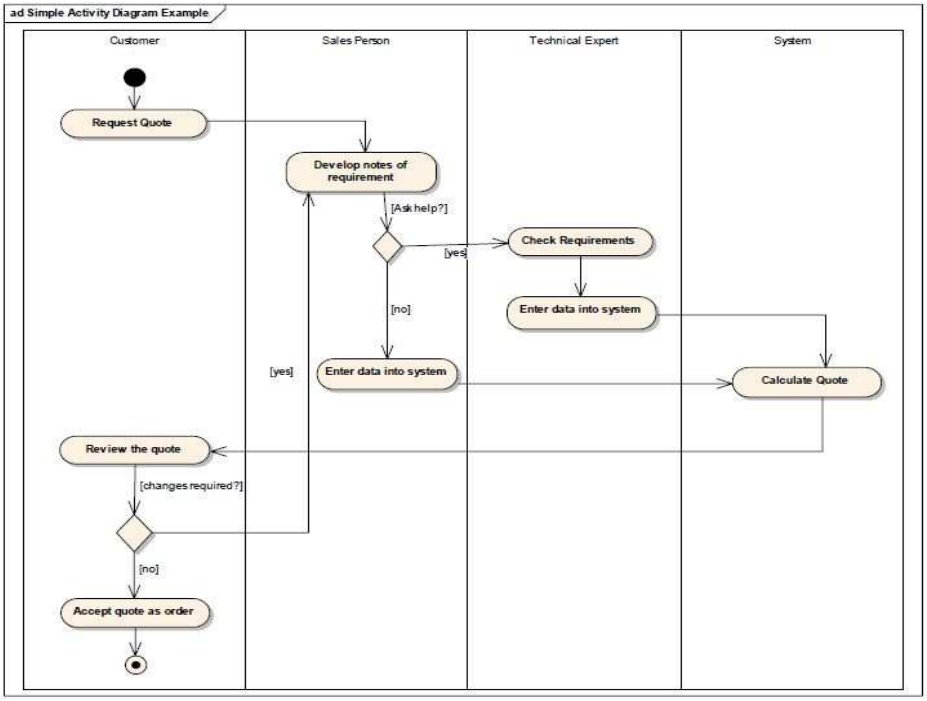
Dari definisi-definisi di atas dapat disimpulkan bahwa *object oriented analysis and design* (OOAD) adalah suatu cara untuk menentukan semua tipe *object* yang bekerja di dalam sebuah sistem dengan menggambarkan interaksi yang dibutuhkan *user* untuk menyelesaikan tugas-tugasnya, dan memperbaiki definisi masing-masing tipe dari *object* sehingga dapat diimplementasikan dengan sebuah bahasa atau lingkungan khusus.

2.4.1 *Activity Diagram*

Menurut Satzinger, Jackson, dan Burd (2009), *Activity Diagram* adalah sebuah diagram alur kerja sederhana yang menjelaskan tentang aktivitas sejumlah *user* (atau *system*), orang yang melakukan setiap aktivitas, dan alur sekuensial dari aktivitas-aktivitas tersebut.

Gambar 2.3 *Activity Diagram*

(Satzinger, p146)



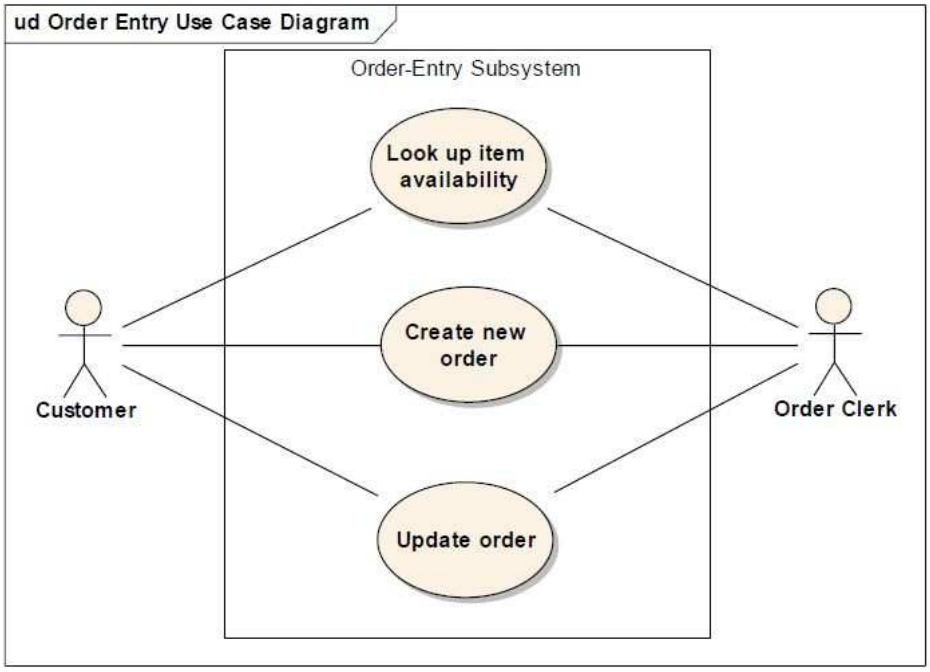
2.4.2 *Use Case Diagram*

Menurut Satzinger, Jackson, dan Burd (2009, p213), *Use Cbase Diagram* adalah diagram yang menggambarkan berbagai peran *user* dan cara *user-user* tersebut berinteraksi dengan sistem. *Use Case Diagram* adalah diagram yang digunakan untuk menggambarkan orang-orang yang menggunakan sistem serta bentuk interaksi antara orang-orang tersebut dengan sistem yang digunakannya.

Hubungan pada *Use Case Diagram* terbagi menjadi:

1. <<*include*>> *relationship* atau disebut juga <<*uses*>> *relationship a*dalah hubungan antar *use case* yang memungkinkan satu *use case* menggunakan fungsionalitas yang disediakan oleh *use case* lain.
2. <<*extends*>> *relationship* merupakan hubungan antar *use case* yang memungkinkan satu *use case* secara *optional* menggunakan fungsionalitas yang disediakan oleh *use case* lain.

Gambar 2.4 *Use Case Diagram*
(Satzinger, p216)



2.4.3 *Use Case Description*

Menurut Satzinger, Jackson, dan Burd (2009, p220), *use case diagram* membantu untuk mengidentifikasi beberapa proses yang dilakukan oleh *user* dan yang harus didukung oleh sistem. Untuk pengembangan sistem yang lebih baik, kita harus lebih masuk ke level detil dengan pendeskripsian. Ada tiga level pendeskripsian dari use case, yaitu *brief description*, *intermediate description*, dan *fully developed description*.

1. *Brief description*

   Brief description biasanya digunakan untuk *use case* yang sederhana, khususnya ketika ruang lingkup sistem yang akan dikembangkan itu masih kecil. *Brief description* (deskripsi ringkas) dapat disebut sebagai ringkasan mengenai apa yang dilakukan oleh sistem untuk merespon aksi dari pengguna.

2. *Intermediate description*

   Intermediate description merupakan deskripsi yang lebih detail dan merupakan perluasan dari sebuah *brief description* untuk memasukkan arus aktivitas-aktivitas internal untuk suatu *use case*. Jika terdapat *multiple scenarios*, maka tiap arus

aktivitas-aktivitas dideskripsikan secara masing-masing. Selain itu, dokumentasi mengenai kondisi-kondisi pengecualian juga dapat didokumentasikan jika diperlukan.

3. *Fully developed description*

Fully developed description merupakan metode yang paling formal untuk mendokumentasikan sebuah *use case*. Dengan menggunakan deskripsi jenis ini kita dapat mengetahui secara *detail* dan meningkatkan pemahaman kita mengenai proses bisnis dan bagaimana sistem harus mendukung proses bisnis tersebut.

Gambar 2.5 *Use case Description*
(Satzinger, p221)

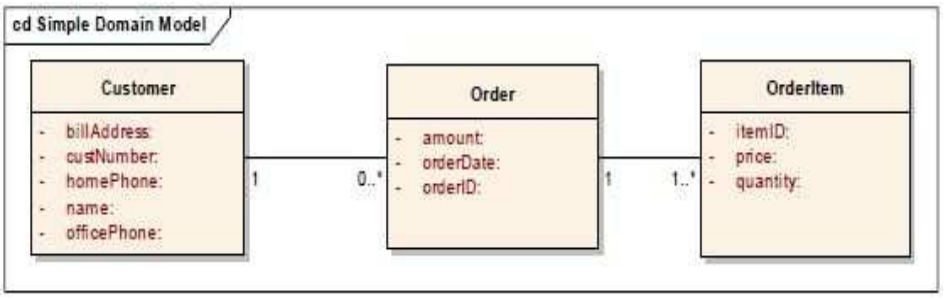
Flow Aktivitas untuk *scenario Order Clerk* membuat *Telephone Order*
Main Flow: 1. Pelanggan menelepon untuk melakukan pemesanan 2. *Order Clerk* memverifikasi informasi pelanggan. Jika pelanggannya adalah pelanggan baru, maka akan melibatkan *use case* "*Maintain Customer Account*" untuk menambahkan data pelanggan baru. 3. *Order Clerk* memulai proses pembuatan pesanan baru. 4. Pelanggan meminta *item* tertentu untuk ditambahkan ke pesanan. 5. *Order Clerk* memverfikasi *item* tersebut dan menambahkannya ke pesanan. 6. Ulangi langkah 4 dan 5 sampai semua *item* sudah dimasukkan ke dalam pesanan. 7. Pelanggan mengindikasikan akhir pemesanan, *Clerk* mengakhiri pembuatan pesanan, *system* menghitung total pemesanan. 8. Pelanggan melakukan pembayaran, *Clerk* memasukkan jumlah pembayaran, *system* memverifikasi pembayaran. 9. *System* mengakhiri proses pemesanan.
Exception Conditions: 1. Jika *item* tidak tersedia, maka pelanggan bisa: a. Memilih untuk tidak membeli *item* tersebut, atau b. Meminta *item* tersebut ditambahkan sebagai *back-ordered item*. 2. Jika pembayaran pelanggan ditolak karena hasil validasi tidak valid, maka a. Pemesanan dibatalkan, atau b. Pemesanan ditunda sampai pengecekan diterima.

2.4.4 *Domain Model Class Diagram*

Menurut Satzinger, Jackson, dan Burd (2009, p184), *Domain Model Class Diagram* adalah sebuah *UML Class Diagram* yang menggambarkan benda-benda yang penting dalam pelaksanaan tugas para pengguna, seperti *class-class problem domain*, hubungan antar *class-class* tersebut, dan atribut-atributnya. *Domain Model Class Diagram* adalah diagram yang digunakan

untuk menggambarkan *class-class* yang terlibat, hubungan antar *class-class* tersebut serta atribut-atributnya.

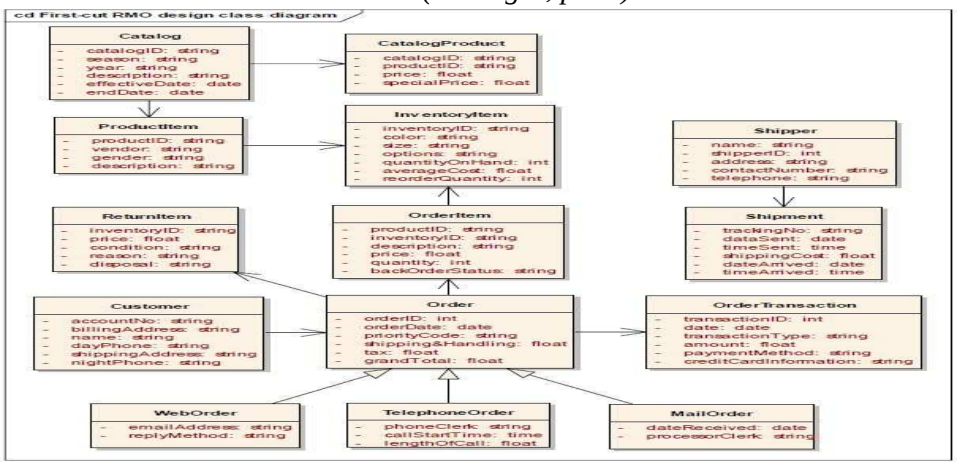
Gambar 2.6 *Domain Model Class Diagram*
(Satzinger, p187)



2.4.5 *First Cut Design Class Diagram*

Menurut Satzinger, Jackson, dan Burd (2009, p309), *first-cut design class diagram* dikembangkan dengan memperluas *domain model class diagram.* Hal ini dilakukan dalam dua tahap, yaitu mengelaborasi atribut dengan jenis dan informasi tentang nilai awal serta menambahkan panah *navigation visibility*. *Navigation visibility* adalah prinsip perancangan dimana sebuah objek dapat melihat atau berinteraksi dengan objek lain. Dalam memulai proses perancangan, pengembangan sebuah *first-cut design class diagram* berdasarkan *domain model class diagram* yang telah dibuat pada tahap sebelumnya.

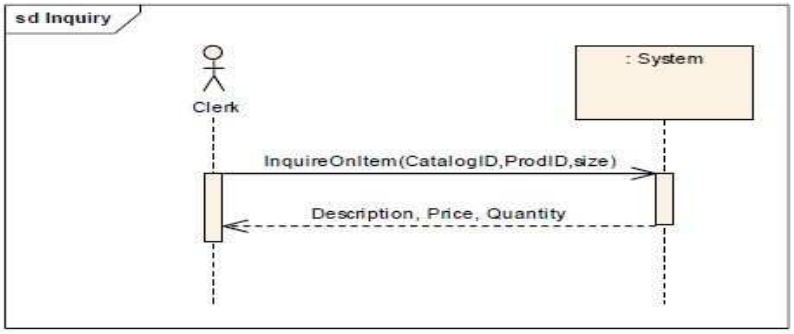
Gambar 2.7 *First Cut Design Class Diagram*
(Satzinger, p311)



2.4.6 *System Sequence Diagram*

System sequence diagram adalah sebuah diagram yang menunjukkan urutan pesan antara aktor eksternal dan sistem selama berjalannya *use case* atau *scenario* (Satzinger, Jackson, dan Burd, 2009, p213).

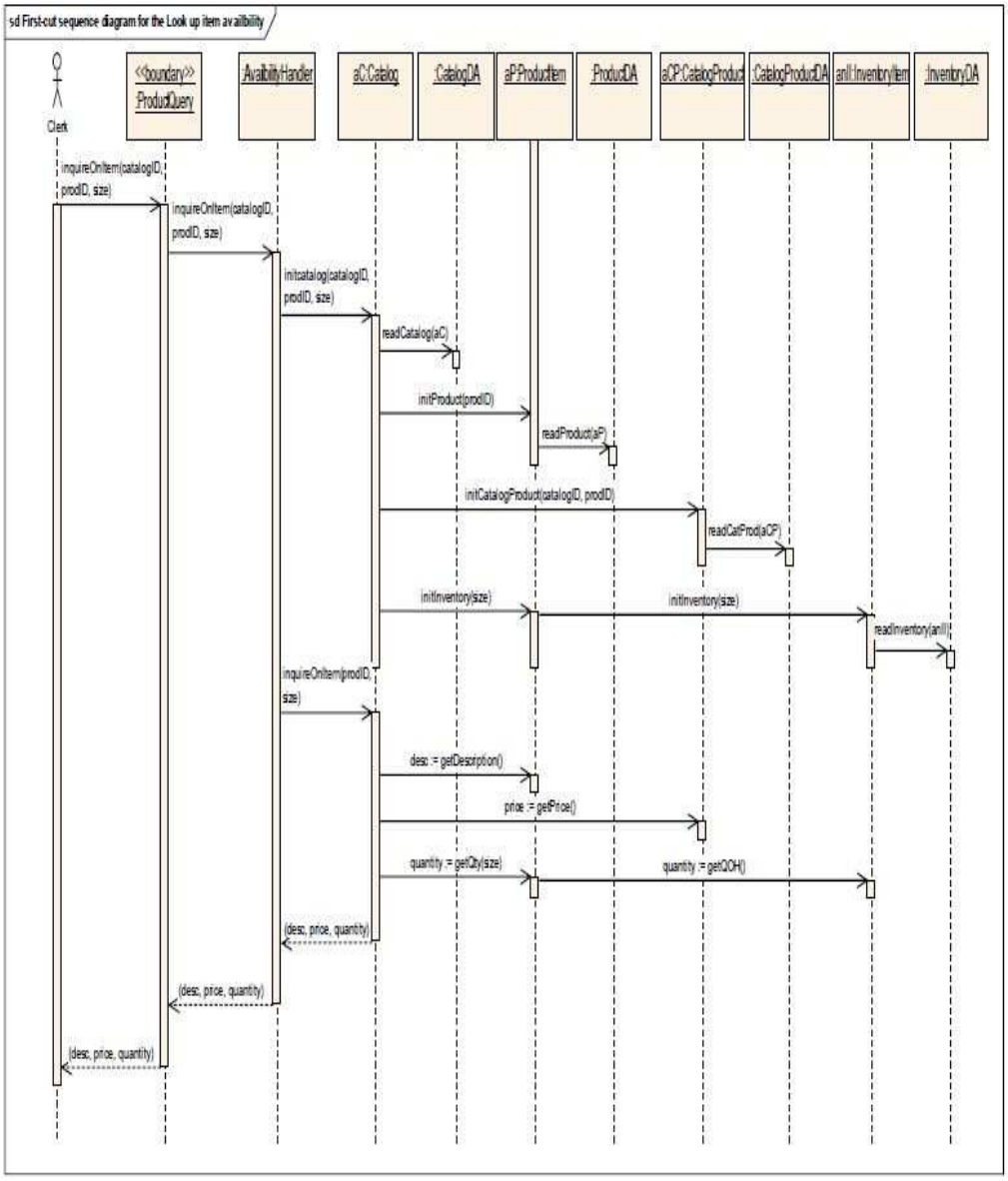
Gambar 2.8 *System Sequence Diagram*
(Satzinger, p229)



2.4.7 *Multilayer Design Sequence Diagram*

First-cut sequence diagram hanya berfokus pada *class* yang ada di *domain layer*, tahap selanjutnya adalah pengembangan *sequence diagram* tersebut dengan memperluas objek-objek yang terlibat dengan membuat *multilayer design*, termasuk *view layer* dan *data access layer* (Satzinger, Jackson, dan Burd, 2009, pp329-333).

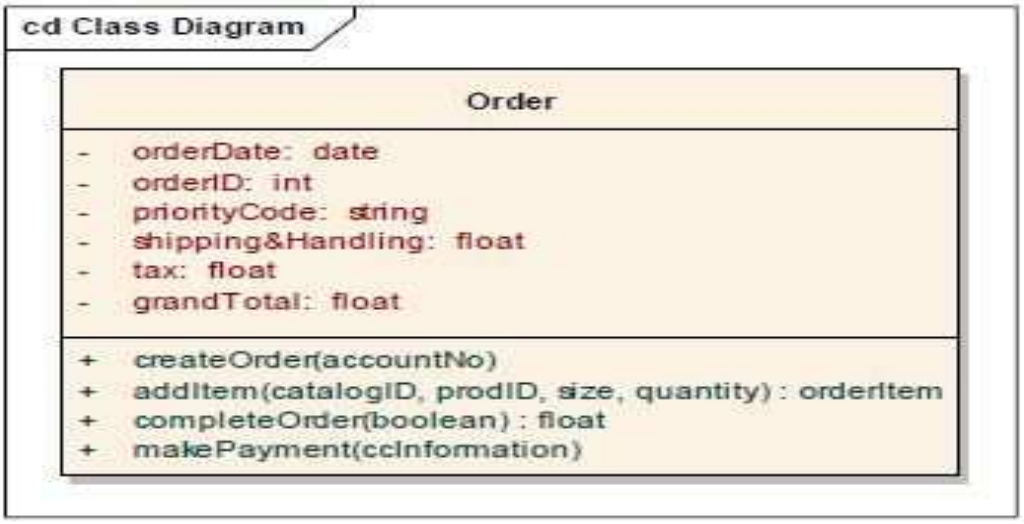
Gambar 2.9 *Multilayer Design Sequence Diagram (Satzinger, p325)*



2.4.8 *Updated Class Diagram*

Menurut Satzinger, Jackson, dan Burd (2009, p337), *Updated Design Class Diagram* adalah *Design Class Diagram* yang dikembangkan untuk tiap layer. Dalam *view layer* dan *data access layer*, beberapa *class* baru harus dispesifik. *Domain layer* juga mempunyai beberapa *class* baru yang ditambahkan untuk *use case controller*.

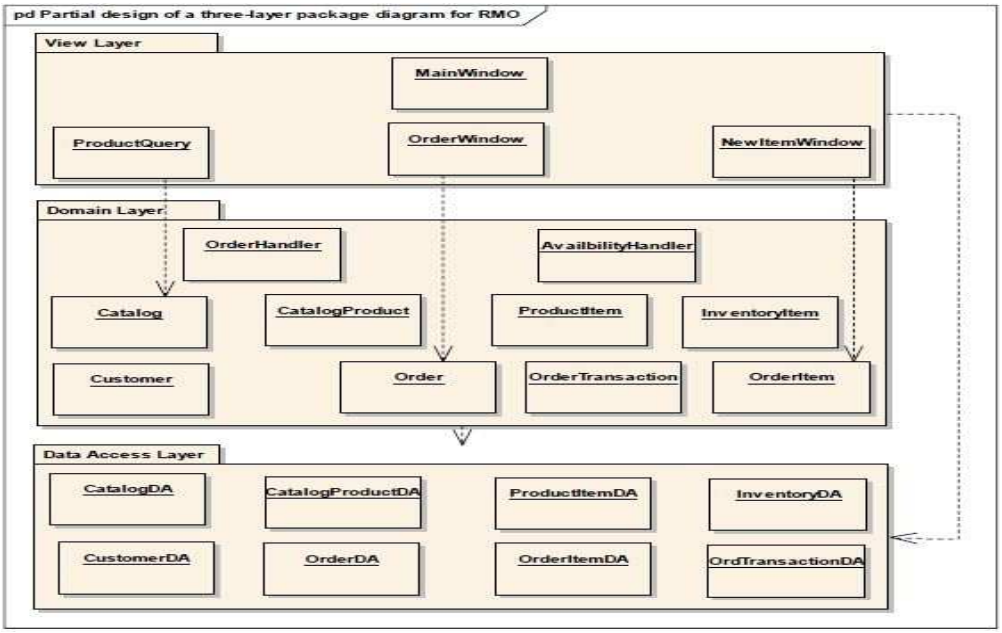
Gambar 2.10 *Updated Class Diagram*
(Satzinger, p339)



2.4.9 *Package Diagram*

Menurut Satzinger, Jackson, dan Burd (2009), Package Diagram adalah diagram tingkat lanjut yang memungkinkan designer mengasosiasikan beberapa *class* didalam kelompok yang berhubungan.

Gambar 2.11 *Package Diagram*
(Satzinger,p341)



2.4.10 *User Interface*

Menurut Satzinger, Jackson, dan Burd (2009, p442), *User Interface* adalah bagian dari sistem informasi yang membutuhkan interaksi dengan *user* untuk menghasilkan input dan output. *User interface* memungkinkan pengguna sistem berinteraksi dengan komputer untuk melakukan suatu proses seperti mencatat transaksi. Terkadang output dihasilkan setelah pengguna berinteraksi dengan sistem, seperti informasi yang ditampilkan setelah *query* mengenai status dari suatu pesanan.

2.5 *Web*

Menurut Williams & Sawyer (2004, p6) tentang *World Wide Web (www)*, biasa disebut dengan *web*, adalah sistem yang saling berhubungan pada komputer *internet* (disebut juga *server*) yang mana mendukung dokumen dengan format spesial dalam bentuk *multimedia*.

Menurut Hongjun Lu (2008), *Web* menyediakan ligkungan yang *user-friendly* untuk penghasil informasi pada *web*. *Web* merupakan CGI *script* yang dapat diprogram yang dapat ditambahkan pada *web server* yang sudah di-*instal*.

2.6 PHP

PHP singkatan dari *PHP Hypertext Preprocessor* yang digunakan sebagai bahasa script *server-side* dalam pengembangan *web* yang disisipkan dalam dokumen HTML. Penggunaan PHP memungkinkan web dapat dibuat dinamis sehingga pemeliharaan situs web tersebut menjadi lebih mudah dan efisien. PHP merupakan *software Open-Source* yang disebarkan dan dilisensikan secara gratis serta dapat di-*download* secara bebas dari situs resminya (*www.php.net*). PHP ditulis dengan menggunakan bahasa C.

Saat ini PHP amat populer dan menggantikan Perl yang sebelumnya juga populer sebagai bahasa *scripting web*. PHP telah menjadi modul Apache terpopuler (menurut *www.securityspace.com*), melebihi FrontPage dan Mod Perl. Dan menurut hasil survei *www.netcraft.co.uk*, PHP terus meningkat penggunaannya dan telah digunakan pada jutaan domain dan jutaan alamat *IP*. PHP telah digunakan oleh berbagai situs populer baik luar negeri maupun situs dalam negeri.

2.7 MySQL

MySQL merupakan bahasa pemrograman *open-source* paling populer dan paling banyak digunakan di lingkungan Linux. Kepopuleran ini karena ditunjang oleh performansi *query* dari *database*-nya yang jarang bermasalah. MySQL (*My Structure Query Language*) adalah sebuah program pembuat *database* yang bersifat *open-source*, artinya siapa saja dapat menggunakannya secara bebas.

MySQL sebenarnya produk yang berjalan pada *platform* Linux. Karena sifatnya yang *open-source*, MySQL dapat berjalan pada semua *platform* baik Windows maupun Linux. Selain itu, MySQL juga merupakan program pengakses *database* yang bersifat jaringan sehingga dapat digunakan untuk aplikasi *multiuser* (banyak pengguna). Saat ini *database* MySQL telah digunakan hampir oleh semua pemrogram *database*, terlebih dalam pemrograman *web*.

Kelebihan lain dari MySQL adalah penggunaan bahasa *query* yang dimiliki SQL (*Structured Query Language*). *SQL* adalah suatu bahasa permintaan yang terstruktur dan telah terstandarisasi untuk semua program pengakses *database* seperti Oracle, PosgreSQL, SQL Server, dan lain-lain.

Menurut Arlita, Ismail, dan Putro (2010), MySQL adalah *Relational Database Management System* (RDMS) yang didistribusikan secara gratis di bawah *General Public Lisence* (GPL). MySQL merupakan turunan dari salah satu konsep utama dalam *database*, yaitu *Structured Query Language* (SQL).

Sebagai sebuah program penghasil *database*, MySQL tidak dapat berjalan sendiri tanpa adanya sebuah aplikasi lain (*interface*). MySQL dapat didukung oleh hampir semua program aplikasi baik yang *open-source* seperti PHP maupun yang tidak, yang ada pada *platform* Windows seperti Visual Basic, Delphi, dan lainnya.

2.8 Interaksi Manusia dan Komputer

Menurut Sneiderman (2010, p32), ada lima faktor manusia terukur yang dapat dijadikan sebagai pusat evaluasi, yaitu :

1. Waktu belajar waktu yang dibutuhkan oleh *user* untuk mempelajari cara relevan dalam mengerjakan tugas dengan lancar.

2. Kecepatan kinerja waktu yang diperlukan untuk mengerjakan suatu tugas yang diberikan.

3. Tingkat kesalahan berapa banyak kesalahan yang dilakukan oleh *user* dan kesalahan-kesalahan seperti apa yang bisa terjadi saat *user* mengerjakan tugas tersebut.

4. Daya ingat kemampuan *user* mempertahankan pengetahuannya setelah jangka waktu tertentu.

5. Kepuasan subjektif kepuasan *user* terhadap berbagai aspek dari sistem.

Menurut Shneiderman (2010, p88), terdapat delapan aturan emas dalam desain antarmuka yaitu:

1. Berusaha untuk konsisten

   Aturan ini merupakan aturan yang paling sering dilanggar, karena terdapat banyak bentuk konsistensi. Konsisten dalam hal urutan aksi, istilah yang digunakan, menu, *layout*, penggunaan warna, tata letak, kapitalisasi, *font*, dan sebagainya.

2. Menyediakan kebutuhan *universal*

   Dengan memahami kebutuhan *user* yang bermacam-macam dan membuat desain fleksibel yang mendukung perubahan dalam konten.

3. Memberikan umpan balik yang informatif

   Untuk setiap aksi *user*, harus ada sistem umpan balik. Untuk aksi kecil yang sering dilakukan, respon dapat dibuat dengan sederhana. Sedangkan untuk aksi yang besar dan jarang dilakukan, respon harus dibuat lebih tegas dan jelas.

4. Desain dialog untuk menghasilkan penutupan atau keadaan akhir

   Urutan aksi hendaknya disusun ke dalam kelompok kategori awal, tengah, dan akhir. Umpan balik yang informatif dapat memberikan kepuasan pencapaian, rasa lega, sinyal untuk mempersiapkan diri memasuki kelompok kategori aksi selanjutnya.

5. Penawaran pencegahan dan penanganan kesalahan sederhana

   Usahakan dalam mendesain suatu sistem, diarahkan agar *user* tidak membuat kesalahan yang serius. Misalnya menyediakan pilihan menu, tidak mengizinkan karakter alfabet pada kotak entri numerik. Jika *user* melakukan kesalahan, sistem harus dapat mendeteksi kesalahan dan menawarkan instruksi yang sederhana, konstruktif, dan spesifik untuk perbaikan. Contoh, *user* tidak perlu mengetik ulang seluruh perintah, melainkan hanya memperbaiki bagian yang salah saja.

6. Mengizinkan pembalikan aksi yang mudah

   Pada suatu sistem aplikasi harus terdapat pembalikan aksi. Fitur ini dapat memperkecil kesalahan, selama *user* tahu bahwa aksi dapat dibatalkan. Pembalikan aksi dapat berupa tindakan tunggal, tugas data entri, atau serangkaian aksi seperti entri nama dan alamat.

7. Mendukung pusat kendali internal

   User yang sudah terbiasa dengan suatu aplikasi, biasanya ingin memiliki kendali atas antarmuka dan tanggapan dari aksinya. Aksi antarmuka yang tidak umum, urutan entri data yang membosankan, kesulitan dalam memperoleh informasi yang dibutuhkan, serta ketidakmampuan menghasilkan aksi yang diinginkan dapat menimbulkan keresahan dan ketidakpuasan pada *user*.

8. Mengurangi beban ingatan jangka pendek

   Manusia mempunyai keterbatasan dalam memproses informasi dengan waktu yang singkat. Oleh karena itu diperlukan tampilan yang sederhana, pengurangan jendela-gerak frekuensi, pemberian waktu pelatihan yang cukup untuk kode-kode, hafalan dan rangkaian aksi. Jika diperlukan, akses *online* untuk sintaks, singkatan, kode, dan informasi yang terkait harus disediakan.

2.9 *Web Database*

Web database merupakan aplikasi penggunaan *web* sebagai *platform* yang menghubungkan pengguna dengan antarmuka satu atau lebih basis data.

Menurut Abdullat (2004), Penggabungan teknologi *internet* dan aplikasi *database* yang membuat komputasi lingkungan baru yang dapat digambarkan sebagai kaya dan kompleks.

Keuntungan dari penggunaan *web database* adalah sebagai berikut (Connolly dan Begg, 2010, p1036-p1038) :

- Kesederhanaan (*Simplicity*)

  Dalam bentuk asli, HTML sebagai *markup language* sangat mudah untuk di pelajari oleh *developer* maupun pengguna akhir

- Tidak bergantung pada *platform* (*Platform independence*)

  Salah satu alasan dibuatnya aplikasi basis data berbasis *web* adalah karena pada umumnya *web-client* (*browser*) tidak bergantung pada *platform* sehingga jika suatu aplikasi akan di jalankan pada sistem operasi yang berbeda tidak memerlukan modifikasi.

- Grafis antarmuka pengguna (*Graphical User Interface* / GUI)

  Menyederhanakan dan meningkatkan pengaksesan basisdata, tetapi GUI memerlukan progam tambahan yang akan menyebabkan ketergantungan pada *platform*. *Web browser* menyediakan GUI yang mudah digunakan untuk mengakses banyak hal. Memiliki GUI yang baik akan mengurangi biaya pelatihan untuk pengguna akhir

- Standarisasi (*Standardization*)

  HTML standar secara *de facto* dimana seluruh *web* browser berada, memungkinkan sebuah dokumen HTML pada satumesin dibaca oleh pengguna pada mesin lainnya di bagian lain dunia melalui koneksi di *internet* dan *web browser*.

- Dukungan lintas *platform* (*Cross-platform Support*)

  Web browser ada secara *virtual* untuk setiap tipe dari *platform* komputer memungkinkan pengguna pada sebagian besar jenis komputer mengakses basis data dari manapun diseluruh bagian dunia.

- Memungkinkan Akses jaringan yang transparan(*Transparent Network Access*)

  Akses jaringan terlihat transparan bagi pengguna, kecuali untuk spesifikasi URL, ditangani sepenuhnya oleh *web browser* dan *web server*. Dukungan *built-in* untuk jaringan akan menyederhanakan akses basis data dan mengeliminasi kebutuhan perangkat lunak jaringan dan kompleksitas dari *platform* yang berbeda untuk saling berkomunikasi.

- Penyebaran bisa diukur (*Scalable Deployment*)

  Solusi berbasis *web* mampu membuat arsitektur *three-tier* yang menyediakan dasar untuk *scalability*. Dengan menyimpan aplikasi di *server* terpisah, maka *web* mengeliminasi waktu dan biaya yang berhubungan dengan aplikasi penyebaran. Hal ini akan menyederhanakan penanganan *upgrade* dan *Admini*strasi pengaturan dari banyak *platform* yang melalui banyak kantor.

- Inovasi (*Innovation*)

  Web memampukan organisasi untuk menyediakan jasa barudan mencapai konsumen baru melalui aplikasi yang dapat diakses secara global.

  Selain memiliki kelebihan-kelebihan yang telah dijelaskan diatas, *web database* juga tidak lepas dari beberapa kelemahan sebagai berikut (Conolly dan Begg, 2010, p1038-p1040):

- Kehandalan (*Reliability)*

  Ketika sebuah perintah diberikan melalui *internet*, tidak dapat dipastikan perintah tersebut pasti akan dikirim.

- Keamanan (*Security*)

Autentifikasi pengguna dan transmisi data sangat kritis karena banyak pengguna yang tidak dikenal bisa mengakses data.

- Biaya (*Cost*)

Biaya perawatan bisa menjadi semakin mahal bersamaan dengan meningkatnya permintaan dari pengguna.

- *Skalabilitas (Scalability)*

Aplikasi *web* bisa menghadapi *load* data yang sangat banyak yang datang secara di terduga. Hal ini memerlukan pembangunan performa yang kuat dari arsitektur *server* yang sangat sulit diukur.

- Fungsi terbatas dari HTML (*Limited Functionality of HTML)*

Beberapa aplikasi basis data interaktif ada yang tidak terkonversi secara mudah ke aplikasi berbasis *web* selama masih menyediakan kemudahan yang sama bagi pengguna.

- *Statelessness*

Statelessness dari lingkungan *web* membuat pengaturan dari koneksi basis data dan transaksi pengguna menjadi sulit dan membutuhkan aplikasi untuk merawat informasi tambahan.

- *Bandwidth*

Kegiatan di dalam *internet* memerlukan *bandwidth* bahkan untuk tugas yang sangat sederhana sekalipun sehingga masalah *bandwidth* menjadi penting.

- Kinerja (*Performance*)

Banyak bagian dari *web database* yang kompleks berpusat pada interpretasi bahasa yang akhirnya membuatnya lebih lambat dari pada basis data tradisional.

- Perangkat pembangunan yang belum siap (*Imaturity ofdevelopmet tools*)

Pembangun aplikasi basis data untuk *web* secara cepat bisa mengidentifikasi

ketidaksiapan perangkat pembangunan yang ada. Teknologi *web* masih belum matang dan pembangunan lingkungan yang lebih baik dibutuhkan untuk meringankan beban dari *programmer* aplikasi.

2.10 Teori Khusus

2.10.1 Persediaan

Menurut Nasution (2003, p103), persediaan adalah sumber daya yang menganggur yang menunggu proses lebih lanjut. Proses lebih lanjut terserbut berupa kegiatan produksi pada sistem manufaktur, kegiatan pemasaran pada sistem distribusi ataupun kegiatan konsumsi pangan pada sistem rumah tangga.

Menurut Freddy Rangkuti (2004, p1), persediaan adalah sebagai berikut. ”Persediaan merupakan bahan-bahan, bagian yang disediakan, dan bahan-bahan dalam proses yang terdapat dalam perusahaan untuk proses produksi, serta barang- barang jadi atau produk yang disediakan untuk memenuhi permintaan dari konsumen atau pelanggan setiap waktu.”

Jadi dapat disimpulkan bahwa persediaan adalah bahan-bahan, bagian yang disediakan, dan bahan-bahan dalam proses yang terdapat dalam perusahaan untuk proses produksi, serta barang-barang jadi atau produk yang disediakan untuk memenuhi permintaan dari konsumen atau pelanggan setiap waktu yang disimpan dan dirawat menurut aturan tertentu dalam tempat persediaan agar selalu dalam keadaan siap pakai dan dicatat dalam bentuk buku perusahaan.

2.10.2 Fungsi dan Tujuan Persediaan

Fungsi persediaan menurut Freddy Rangkuti (2004, p15) adalah sebagai berikut:

1. Fungsi *Batch Stock* atau *Lot Size Inventory*

   Penyimpanan persediaan dalam jumlah besar dengan pertimbangan adanya potongan harga pada harga pembelian,

efisiensi produksi karena psoses produksi yang lama, dan adanya penghematan di biaya angkutan.

2. Fungsi *Decoupling*

Merupakan fungsi perusahaan untuk mengadakan persediaan *decouple*, dengan mengadakan pengelompokan operasional secara terpisah-pisah.

3. Fungsi Antisipasi

Merupakan penyimpanan persediaan bahan yang fungsinya untuk penyelamatan jika sampai terjadi keterlambatan datangnya pesanan bahan dari pemasok. Tujuan utama adalah untuk menjaga proses konversi agar tetap berjalan dengan lancar.

2.10.3 Pembelian

Siklus pembelian mencakup perolehan barang tunai untuk dijual kembali atau untuk digunakan dalam produksi, perolehan jasa perorangan, perolehan aktiva dan peralatan. Transaksi pembelian dapat digolongkan menjadi dua : pembelian lokal dan *import*.

Menurut Mulyadi (2001, P301) pembelian adalah suatu usaha digunakan dalam perusahaan untuk pengadaan barang yang diperlukan bagi perusahaan.

Fungsi yang terkait dalam sistem pembelian adalah :

2.10.3.1 Bagian Gudang

Bertanggung jawab untuk mengajukan permintaan pembelian sesuai dengan persediaan yang ada di gudang dan untuk menyimpan barang yang telah diterima untuk tugas penerima.

2.10.3.2 Bagian Pembelian

Bertanggung jawab untuk memperoleh informasi mengenai harga barang, penentuan pembelian yang dipilih dalam pengadaan barang dan mengeluarkan order beli kepada pemasok yang dituju.

2.10.3.3 Bagian Penerimaan

Bertanggungjawab dalam melakukan pemeriksaan terhadap jenis, mutu dan kualitas barang yang diterima dari pemasok guna menentukan dapat atau tidaknya barang tersebut diterima oleh perusahaan

2.10.3.4 Bagian Akuntansi

Bagian akuntansi yang terkait transaksi pembelian adalah bagian penvatatan persediaan dan bagian pencatatan pembayaran. Bagian pencatatan persediaan mencatat harga pokok persediaan yang dibeli kedalam surat persediaan. Bagian pencatatan pembayaran mencatat transaksi pembelian kedalam register bukti kas keluar yang berfungsi sebagai pencatat surat kas keluar.

2.10.4 Penyewaan

Layanan utama bisnis rental adalah menyediakan akses kendaraan atau peralatan yang dibutuhkan. Ketika mengembangkan ide bisnis penyewaan kendaraan atau peralatan ada beberapa aspek yang harus diperhatikan, aspek ini mencakup:

1. **Lokasi yang diusulkan**, mengevaluasi lokasi dan kemampuan lokasi tersebut dalam mendukung bisnis penyewaan. Lokasi terbaik akan tergantung pada ruang lingkup bisnis sewa dan tata letak komunitas, mudah diakses orang melakukan hal-hal lain dalam komunitas atau dekat ke bandara atau akomodasi pengunjung utama.
2. **Fasilitas**, kebutuhan fasilitas akan bervariasi sesuai dengan jenis usaha sewa yang ingin dibuka. Dengan cukup uang dan renovasi untuk membangun, membeli, atau menyewa ruang usaha sewa guna menemukan fasilitas terbaik sesuai dengan kebutuhan. Tetapi semua tergantung pada ketersediaan ruang dan preferensi pribadi untuk kebutuhan manajemen kantor dan penyimpanan peralatan sewa dan peralatan untuk fasilitas yang lebih besar dengan ruang untuk menyimpan kendaraan di dalam.

3. **Barang dan jasa yang tersedia**, ada dua jenis penyewaan yang dapat dilakukan, yaitu penyewaan barang dan juga jasa. Kita harus dapat menganalisa kebutuhan-kebutuhan konsumen saat ini agar dapat menawarkan barang ataupun jasa yang sesuai dengan apa yang konsumen butuhkan.
4. **Potensial pelanggan**, ketika melakukan penilaian pelanggan, ingat bahwa tujuan utama adalah untuk memahami potensial pelanggan lebih baik. Pada akhir proses, harus mampu menjawab pertanyaan-pertanyaan sebagai berikut:
    - Potensial pelanggan apa yang sudah ada di wilayah tersebut atau masyarakat?
    - Apakah pelanggan ini bersedia untuk menghabiskan uang?
    - Sensitivitas harga apa yang pelanggan memiliki?
    - Siapa pelanggan (menurut usia, jenis kelamin, pendidikan, pendapatan, pekerjaan, dll)

2.10.5 Perbaikan

Perencanaan perbaikan kendaraan harus mencakup inventarisasi semua kendaraan yang dicakup oleh rencana organisasi. Inventarisasi harus diserahkan pada *form* yang tersedia . Ini adalah bentuk yang sama digunakan untuk laporan inventaris tahunan minimal harus menyertakan semua kendaraan yang didanai .

Informasi mengenai detail setiap pelayanan kendaraan yang direncanakan:

1. Tahun / Membuat / Model .

a . Tahun - Tahun kendaraan itu diproduksi .

b . Membuat - Nama dan produsen

(yaitu, Ford Eldorado , dll) .

c . Model - nama model atau nomor yang ditetapkan oleh produsen (kafilah , E - 350 , dll) .

2. Kode Kendaraan - Jenis kendaraan

3. Nomor Identifikasi Kendaraan (VIN) - Nomor seri yang ditugaskan oleh produsen.

4. Badan Nomor Kendaraan - Jumlah ditugaskan untuk kendaraan dengan organisasi Anda .

5. Kondisi - Titik yang menggambarkan kondisi mekanik / jasa kendaraan :

a. 100 - Hanya perawatan pencegahan rutin diperlukan .

b. 80 sampai 90 - rangka kerja yang baik , membutuhkan perbaikan kecil hanya jarang (lebih dari enam bulan antara perbaikan) .

c. 50 sampai 70 - Membutuhkan sering perbaikan kecil atau perbaikan besar jarang terjadi.

d. 20 sampai 40 - Memerlukan perbaikan besar sering (kurang dari enam bulan antara perbaikan) .

e. 10 - Penggunaan Lanjutan menyajikan biaya perbaikan yang berlebihan dan potensi gangguan layanan .

6. Usia - Usia kendaraan tahun.

7. Sisa Masa Manfaat - Perkiraan jumlah tahun kendaraan akan dapat melaksanakan tujuan yang telah ditetapkan sebelum digantikan .

8. Biaya Penggantian - saat tahun perkiraan harga pembelian kendaraan baru dari jenis ini .

9. Kapasitas tempat duduk - Jumlah kursi yang tersedia untuk stasiun baik rawat jalan dan kursi publik (termasuk supir untuk kendaraan vanpool) .

10. Jenis BBM - Surat singkatan dari jenis bahan bakar yang digunakan oleh kendaraan . (Lihat bentuk persediaan untuk daftar jenis bahan bakar.)

11. WSDOT Jabatan - Jika kendaraan itu dibeli dengan dana hibah melalui WSDOT , adalah judul masih dipegang oleh WSDOT.