BAB II

TINJAUAN PUSTAKA

II.1. Sistem Informasi

II.1.1. Sistem

Perancangan suatu program aplikasi terdiri dari satu kesatuan sistem. Terdapat dua kelompok pendekatan di dalam mendefinisikan sistem, yaitu yang menekankan pada prosedur dan yang menekankan pada komponen. Pendekatan sistem yang lebih menekankan pada prosedur mendefinisikan sistem sebagai berikut.

"Sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau menyelesaikan suatu sasaran tertentu". (Jogiyanto HM ; 2005 : 1)

Prosedur didefinisikan oleh Ricard F. Neuschel sebagai suatu urut-urutan operasi klerikal (tulis-menulis), biasanya melibatkan beberapa orang di dalam satu atau lebih departemen, yang diterapkan untuk menjamin penanganan yang seragam dari transaksi-transaksi bisnis yang terjadi.

Pendekatan yang lebih menekankan pada elemen atau komponennya didefinisikan sistem sebagai kumpulan dari elemen-elemen yang berinteraksi untuk mencapai tujuan tertentu. Suatu sistem yang baik harus mempunyai tujuan

dan sasaran yang tepat karena hal ini akan sangat menentukan dalam mendefinisikan masukan yang dibutuhkan sistem dan juga keluaran yang dihasilkan.

Dapat disimpulkan sistem adalah kegiatan-kegiatan yang saling berhubungan antara satu sama yang lainnya yang terdiri dari objek-objek, unsur-unsur atau komponen-komponen sehingga membentuk suatu kesatuan pemrosesan untuk mencapai tujuan tertentu.

II.1.2. Informasi

Informasi adalah data yang telah diklasifikasikan atau diolah atau diinterpretasikan untuk digunakan dalam proses pengambilan keputusan (Tata Sutabri ; 2004: 18).

Adapun kualitas dari informasi adalah sebagai berikut :

1. Akurat *(accurate)*
2. Tepat waktu *(timelines)*
3. Relevan *(relevance)*

II.1.3. Sistem Informasi

Sistem Informasi adalah suatu sistem didalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian yang mendukung fungsi operasi organisasi yang bersifat manajerial dengan kegiatan strategi dari suatu organisasi untuk dapat menyediakan kepada pihak luar dengan laporan-laporan yang diperlukan (Tata Sutabri ; 2004: 36).

Sistem informasi adalah suatu sistem di dalam suatu organisasi yang mempertemukan kebutuhan pengolahan transaksi harian, mendukung operasi, bersifat manajerial dan kegiatan strategi dari suatu organisasi dan menyediakan pihak luar tertentu dengan laporan-laporan yang diperlukan.” (Jogiyanto HM ; 2005 : 11)

Sistem informasi disebut sebagai sistem buatan manusia yang biasanya terdiri dari sekumpulan komponen baik manual ataupun berbasis komputer yang terintegrasi untuk mengumpulkan, menyimpan, dan mengelola data serta menyediakan informasi kepada pihak-pihak yang berkepentingan sebagai pemakai informasi tersebut.” (Anastasia Diana & Lilis Setiawati ; 2011 : 4)

II.1.4. Konsep Dasar Sistem Informasi

Komponen-komponen sistem informasi adalah sebagai berikut:

1. Perangkat keras *(hardware)*

   Perangkat keras *(hardware)* terdiri dari monitor, CPU, keyboard, mouse dan harddisk.

2. Perangkat lunak *(software)*

   Perangkat lunak *(software)* berupa program-program aplikasi yang akan digunakan, yaitu merupakan kumpulan dari perintah atau fungsi yang ditulis dengan aturan tertentu untuk memerintahkan komputer melaksanakan tugas tertentu.

3. Data

   Data merupakan komponen dasar dari informasi yang akan diproses lebih lanjut untuk menghasilkan informasi.

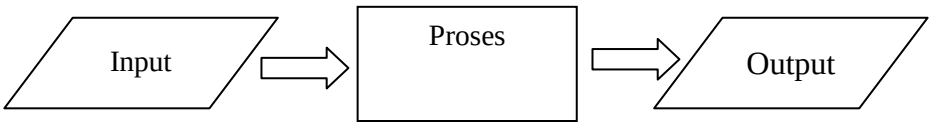
4. Prosedur

   Prosedur merupakan dokumentasi prosedur atau proses sistem, tata cara atau penuntun operasional (aplikasi) dan teknis

5. Manusia

   Manusia adalah pengguna dari sistem informasi.

Sedangkan komponen utama suatu sistem informasi terdiri dari *input*, proses dan *output*. Adapun komponen utama sistem informasi dapat dilihat pada gambar II.1. dibawah ini :



Gambar II.1. Komponen sistem informasi

Sumber : Anastasia Diana & Lilis Setiawati (2011 : 4)

II.2. Sistem Informasi Akuntansi

Sistem Informasi Akuntansi (SIA) adalah sistem yang bertujuan untuk mengumpulkan data serta melaporkan informasi yang berkaitan dengan transaksi keuangan (Anastasia Diana & Lilis Setiawati ; 2011 : 4)

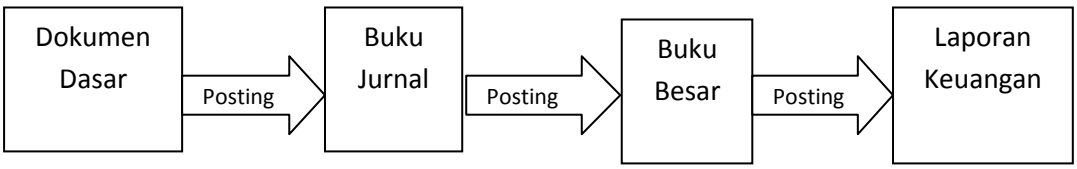
Lingkup sistem informasi akuntansi dapat dijelaskan dari manfaat yang didapat dari informasi akuntansi. Manfaat atau tujuan sistem informasi akuntansi tersebut adalah sebagai berikut :

1. Mengamankan harta/kekayaan perusahaan.
2. Menghasilkan beragam informasi untuk mengambil keputusan.
3. Menghasilkan informasi untuk pihak eksternal.
4. Menghasilkan informasi untuk penilaian kinerja karyawan atau divisi.
5. Menyediakan data masa lalu untuk kepentingan audit (pemeriksaan).
6. Menghasilkan informasi untuk penyusunan dan evaluasi anggaran perusahaan.
7. Menghasilkan informasi yang diperlukan dalam kegiatan perencanaan dan pengendalian.

II.3. Akuntansi

Akuntansi adalah aktivitas mengumpulkan, menganalisis, menyajikan dalam bentuk angka, mengklasifikasikan, mencatat, meringkas, dan melaporkan aktivitas / transaksi perusahaan dalam bentuk informasi keuangan. **(**Rudianto ; 2009 : 14)

Dalam proses menghasilkan informasi yang dibutuhkan oleh berbagai pihak yang berkepentingan, akuntansi harus melewati beberapa tahapan proses, untuk sampai pada penyajian informasi keuangan yang dibutuhkan berbagai pihak. Proses akuntansi itu disebut dengan Siklus Akuntansi. Siklus Akuntansi adalah urutan kerja yang harus dibuat oleh akuntan, sejak awal hingga menghasilkan laporan keuangan suatu perusahaan. Adapun siklus akuntansi dapat dilihat pada gambar II.2. dibawah ini :



Gambar II.2. Siklus Akuntansi

Sumber : Pengantar Akuntansi (Rudianto : 2009:14)

Keterangan :

a. Dokumen Dasar adalah bukti transaksi yang dijadikan dasar oleh akuntan untuk mencatat, seperti: faktur, kuitansi, nota penjualan,dll.

b. Jurnal *(Journal)* adalah aktivitas meringkas dan mencatat transaksi perusahaan berdasarkan dokumen dasar. Tempat untuk mencatat dan meringkas tersebut disebut dengan buku jurnal.

c. *Posting* adalah aktivitas memindahkan catatan di buku jurnal kedalam buku besar sesuai dengan jenis transaksi dan nama perkiraan masing-masing.

d. Buku Besar *(General Ledger)* adalah kumpulan dari semua akun/perkiraan yang dimiliki suatu perusahaan yang saling berhubungan satu dengan lainnya dan merupakan suatu kesatuan.

e. Akun/perkiraan *(Account)* adalah suatu kelas informasi di dalam suatu sistem akuntansi. Atau suatu media yang digunakan untuk mencatat informasi sumber daya perusahaan dan informasi lainnya berdasarkan jenisnya. Misalnya perkiraan kas, perkiraan piutang, akun modal, dan sebagainya.

II.4. Pengenalan UML

Unified Modelling Language (UML) adalah salah satu standar bahasa yang banyak digunakan di dunia industri untuk mendefinisikan *requirement*, membuat analisis & desain, serta menggambarkan arsitektur dalam pemrograman berorientasi objek. (Rosa A.S - M.Shalahuddin ; 2011 : 113)

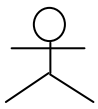
UML dikembangkan oleh 3 pendekar ‘berorientasi objek’, yaitu *Grady Booch, Jim Rumbaugh*, dan *Ivar Jacobson*. UML menjadi bahasa yang bisa digunakan untuk berkomunikasi dalam perspektif obyek antara *user* dengan *developer*, antara *developer* dengan *developer*, antara *developer* analisis dengan

developer disain, dan antara *developer* disain dengan *developer* pemrograman. UML memungkinkan *developer* melakukan permodelan secara *visual*, yaitu penekanan pada penggambaran, bukan didominasi oleh narasi. Permodelan *visual* membantu untuk menangkap struktur dan kelakuan dari obyek, mempermudah penggambaran interaksi antara elemen dalam sistem, dan mempertahankan konsistensi antara disain dan implementasi dalam pemrograman.

UML menyediakan standar pada notasi dan artifak (*diagram*) yang bisa digunakan untuk memodelkan suatu sistem. Berikut ini adalah notasi yang ada pada UML, yaitu sebagai berikut:

1. *Actor*

   Actor adalah orang, proses, atau sistem lain yang berinteraksi dengan sistem informasi yang akan dibuat. Jadi walaupun simbol dari *actor* adalah gambar orang, tapi *actor* belum tentu merupakan orang.



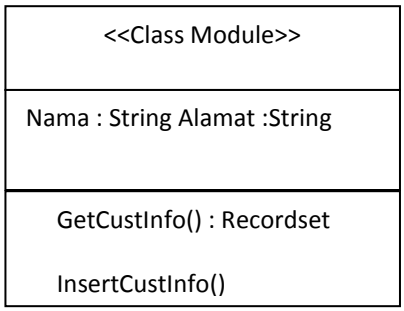
Gambar II.3. Notasi *Actor*

Sumber : Rosa A.S - M.Shalahuddin (2011 : 131)

2. Class

Class merupakan pembentuk utama dari sistem berorientasi objek karena class menunjukkan kumpulan objek yang memiliki atribut dan operasi yang sama. Notasi *class* berbentuk persegi panjang berisi 3 bagian : persegi paling atas untuk

nama *class*, persegi panjang tengah untuk atribut, dan persegi panjang paling bawah untuk operasi. Seperti yang ditunjukkan pada gambar II.6. sebagai berikut :

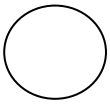


Gambar II.4. Notasi *Class*

Sumber : Rosa A.S - M.Shalahuddin (2011 : 122)

3. Interface

Interface merupakan kumpulan operasi tanpa implementasi dari suatu *class*.

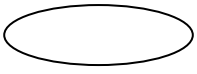


Gambar II.5. Notasi Interface

Sumber : Rosa A.S - M.Shalahuddin (2011 : 122)

4. Use Case

Use case merupakan fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antarunit atau *actor*.

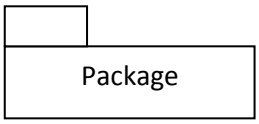


Gambar II.6. Notasi *Use Case*

Sumber : Rosa A.S - M.Shalahuddin (2011 : 131)

5. Package

Package merupakan sebuah bungkusan dari satu atau lebih kelas atau elemen diagram UML lainnya. Tujuannya adalah untuk mempermudah penglihatan (*visibility*) dari model yang sedang dibangun.



Gambar II.7. Notasi *Package*

Sumber : Rosa A.S - M.Shalahuddin (2011 : 130)

6. Note

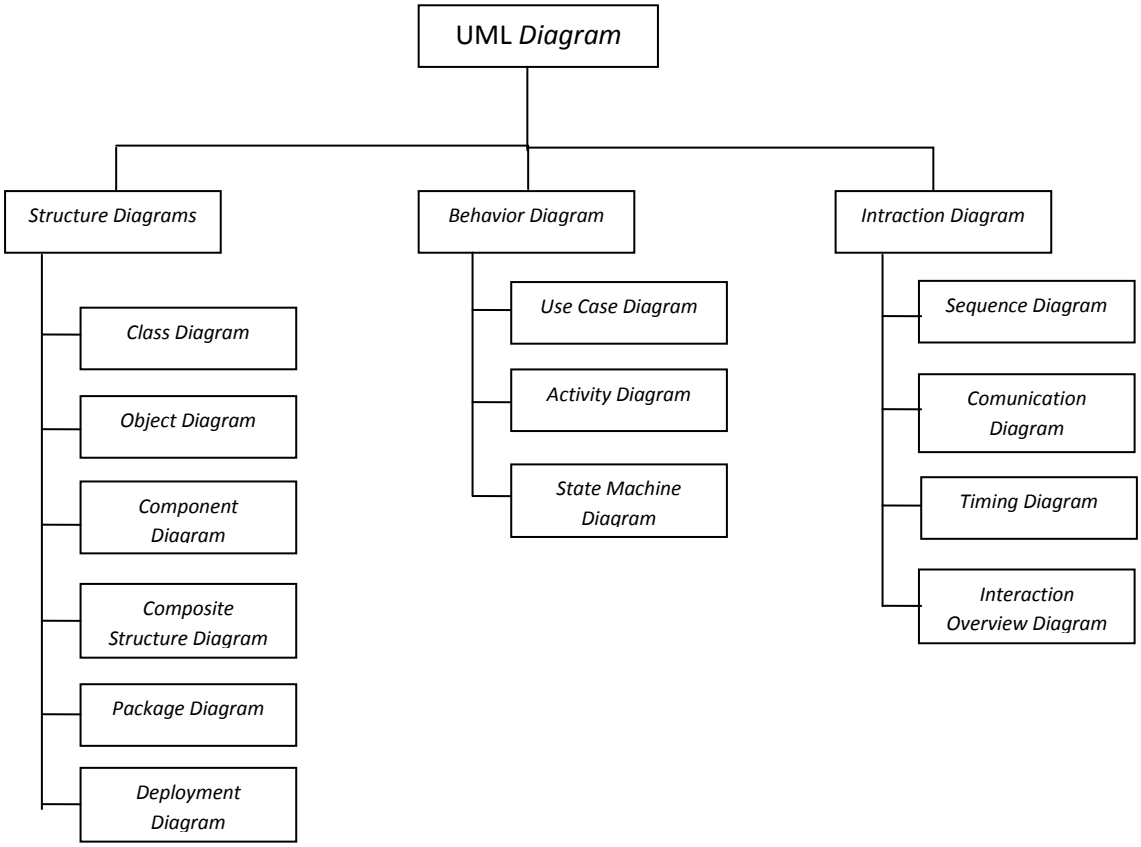
Note digunakan untuk memberikan keterangan dan komentar tambahan dari suatu elemen sehingga bisa langsung terlampir dalam model.

Gambar II.8. Notasi *Note*

Sumber : Rosa A.S - M.Shalahuddin (2011 : 130)

II.5. Diagram UML

Pembagian kategori dan macam-macam diagram tersebut dapat dilihat pada gambar II.9. dibawah ini :



Gambar II.9. *Diagram* UML

Sumber : Rosa A.S - M.Shalahuddin (2011 : 121)

Berikut ini penjelasan singkat dari pembagian kategori tersebut :

1. *Structure Diagrams*

Yaitu kumpulan *diagram* yang digunakan untuk menggambarkan suatu struktur statis dari sistem yang dimodelkan.

2. *Behavior Diagrams*

Yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.

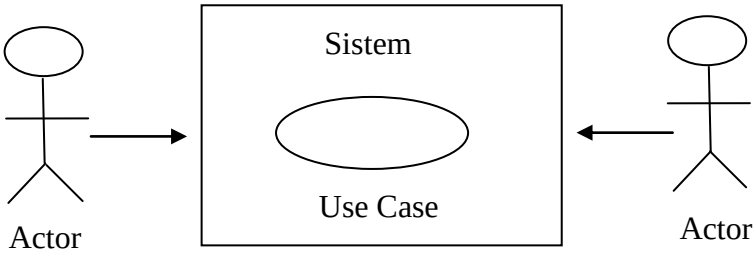
3. *Interaction Diagrams*

Yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antar subsistem pada suatu sistem.

Diagram – diagram pada metode UML (*Unified Modelling Language*) adalah sebagai berikut :

1. *Use Case Diagram*

Use case adalah alat bantu terbaik gune menstimulasikan pengguna potensial untuk mengatakan tentang suatu sistem dari sudut pandangnya. Tidak selalu mudah bagi pengguna untuk menyatakan bagaimana mereka bermaksud menggunakan sebuah sistem. Karena sistem pengembangan tradisional sering ceroboh dalam melakukan analisis, akibatnya pengguna seringkali susah menjawabnya tatkala dimintai masukan tentang sesuatu. Ide dasarnya adalah bagaimana melibatkan penggunaan sistem di fase – fase awal analisis dan perancangan sistem. Diagram *use case* menunjukkan 3 aspek dari sistem yaitu *actor, use case* dan sistem / sub sistem *boundary. Actor* mewakili peran orang, sistem yang lain atau alat ketika berkomunikasi dengan *use case*. Gambar II.10. mengilustrasikan *actor, use case* dan *boundary.*

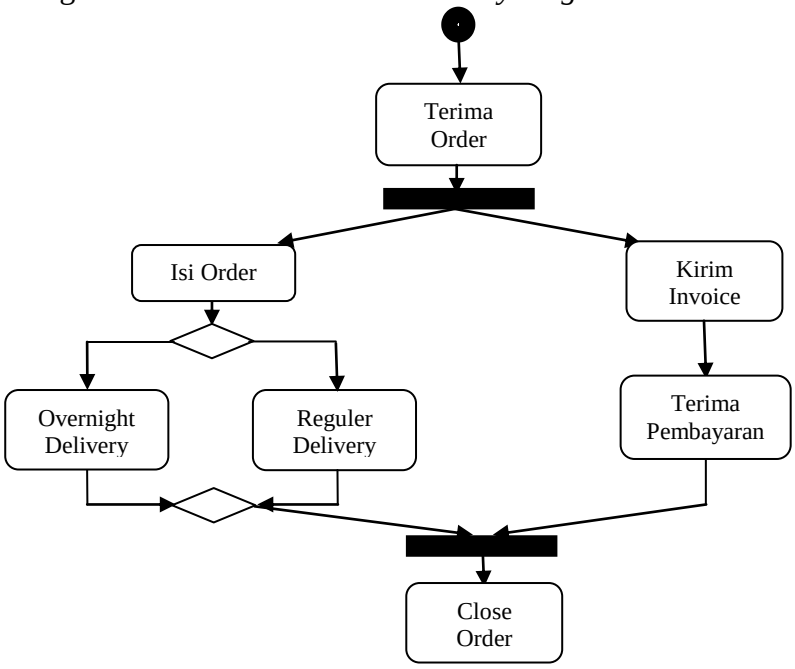


Gambar II.10. Use Case Model

Sumber : Munawar (2005 : 64)

2. *Activity Diagram*

Activity diagram adalah teknik untuk mendeskripsikan logika prosedural, proses bisnis dan aliran kerja dalam banyak kasus. *Activity diagram* mempunyai peran seperti halnya *flowchart*, akan tetapi perbedaannya dengan *flowchart* adalah *activity diagram* bisa mendukung perilaku paralel sedangkan *flowchart* tidak bisa. Berikut gambar dari sederhana dari *Activity diagram.*



Gambar II.11. Contoh Activity Diagram Sederhana

Sumber : Munawar (2005 : 111)

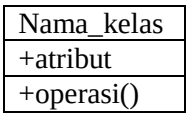
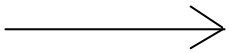
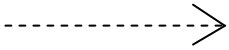
3. *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang akan dibuat untuk membangun sistem. Kelas memiliki apa yang disebut atribut dan metode atau operasi.

a. Atribut merupakan varabel-variabel yang dimiliki oleh suatu kelas.

b. Operasi atau metode adalah fungsi-fungsi yang dimiliki oleh suatu kelas.

Tabel II.1 Simbol-Simbol pada Class Diagram

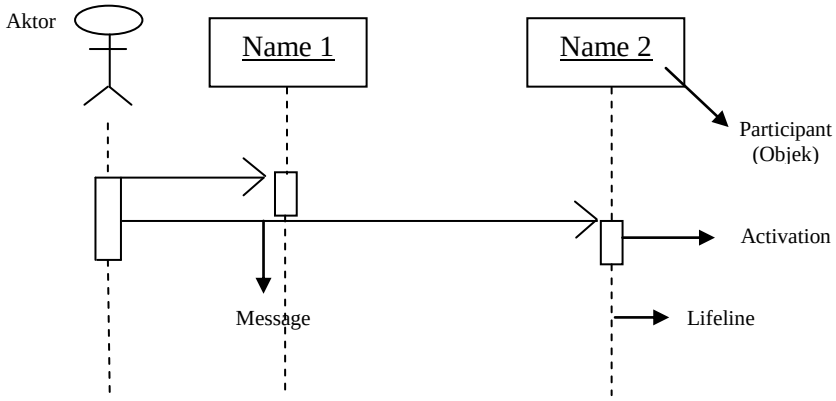
Simbol	Deskripsi
Kelas 	kelas pada struktur sistem
Antarmuka / interface  Nama_interface	sama dengan konsep interface dalam pemrograman berorientasi objek
Asosiasi / association 	relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan multiplicity
Asosiasi berarah / directed association 	relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan multiplicity
Generalisasi 	relasi antar kelas dengan makna generalisasi-spesialisasi (umum khusus)
Kebergantungan / dependency 	relasi antar kelas dengan makna kebergantungan antar kelas
Agregasi / aggregation	relasi antar kelas dengan makna

Sumber : Rosa A.S.M.Shalahuddin (2011 : 123)

4. *Sequence Diagram*

Sequence Diagram digunakan untuk menggambarkan prilaku pada sebuah skenario. Diagram ini menunjukan sejumlah contoh obyek dan pesan yang diletakan diantara obyek – obyek ini di dalam *use case*.

Komponen utama *sequence diagram* terdiri atas obyek yang dituliskan dengan kotak segiempat bernama. *Message* diwakili oleh garis dengan tanda panah dan waktu yang ditunjukan dengan *progress vertical* (Munawar, 2005 : 87).



Gambar II.12. Simbol-Simbol pada Sequence Diagram

Sumber : Munawar (2005 : 89)

II.6. Sekilas Tentang PHP

PHP adalah akronim dari *Hypertext Preprocessor*, yaitu suatu bahasa pemrograman yang berjalan dalam sebuah *web server* dan berfungsi sebagai pengolah data pada sebuah *server*. (Madcoms Madiun ; 2011 : 11)

Kode PHP mempunyai ciri-ciri khusus, yaitu sebagai berikut :

a. Hanya dapat dijalankan menggunakan *web server*, misal : Apache.
b. Kode PHP diletakkan dan dijalankan di *web browser*.
c. Kode PHP dapat digunakan untuk mengakses *database*, seperti: MySQL, PostgreSQL, *Oracle*, dan lain-lain.
d. Merupakan *software* yang bersifat *open source*.
e. Gratis untuk di-*download* dan digunakan.
f. Memiliki sifat *multiflatform*, artinya dapat dijalankan menggunakan sistem operasi apapun, seperti: Linux, Unix, *Windows*, dan lain-lain.

PHP dijalankan dalam *file* berekstensi .php, .php3 atau .phtml, itu tergantung dengan *setting*an PHP anda, tetapi secara umum ekstensi *file* PHP adalah .php. Kode PHP menyatu dengan tag – tag HTML dalam satu *file*. Kode PHP diawali dengan tag <? atau <?php dan ditutup dengan ?>.

Struktur penulisan dalam PHP, sama seperti dalam C++, yaitu setiap pernyataan diakhiri oleh *semicolon* (;) dan bersifat *case sensitive untuk penulisan nama variabel.* Cara penulisan komentar dalam PHP juga sama dengan C++.

Beberapa hal yang harus diperhatikan dalam pemrograman PHP adalah :

1. Tipe Data

PHP mengenal 5 tipe data yaitu *integer, floating point, string, array* dan *object*. Penggunaan tipe data tidak secara ekspilisit dideklarasikan seperti dalam C++.

2. Operator

Dalam PHP terdapat operator aritmatika, *assignment*, *bitwise*, perbandingan, logika, *increment*/*decrement* yang kesemuanya sama dengan C++ dalam cara penggunaannya.

3. Pernyataan

Dalam PHP juga terdapat *conditional statement* yang cara penggunaannya sama seperti dalam C++.

4. Fungsi

Dalam PHP, tipe data balikan sebuah fungsi tidak di deklarasikan secara eksplisit seperti dalam C++. Dalam PHP, fungsi tidak perlu dideklarasikan, cukup di definisikan saja. Pendefinisian fungsi dapat diletakkan di awal, tengah, akhir maupun di *file* lain.

5. Operasi *File*

Membuka *File* → f*open* (nama_*file*, mode_akses);

Menutup *File* → f*close*(*file_pointer*)

Membaca Isi *File* → f*gets*(*file_pointer*, panjang_*string*)

Tag HTML tidak diabaikan

f*gets*(*file_pointer*, panjang_*string*)

Mengabaikan tag HTML

Menulis ke *File* → f*puts*(*file_pointer*,*string*)

Memeriksa apakah *pointer* telah berada di akhir *file* → feof(*file_pointer*)

Ket : → Gunakan fungsi

mode_akes pada PHP sama dengan mode akses pada C++.

$file = f*open*(“coba.txt”,”r+w”). $*file* disebut sebagai *file_pointer*.

II.7. Sekilas Tentang MySQL

Menurut Madcoms (2011:140) MySQL adalah salah satu pogram yang dapat digunakan sebagai *database*, dan merupakan salah satu *software* untuk *database server* yang banyak digunakan. MySQL bersifat *Open Source* dan menggunakan SQL. MySQL bisa dijalankan diberbagai *platfrom* misalnya *Windows*, Linux, dan lain sebagainya.

MySQL adalah suatu perangkat lunak *database* relasi (*Relational Database Management System* atau RDBMS) yang paling banyak digunakan oleh *programmer* dalam pengolahan *database*. MySQL memiliki beberapa keunggulan dari perangkat lunak yang lain dalam mengolah *database*, yaitu :

1. MySQL dapat digunakan oleh beberapa *user* dalam waktu yang bersamaan tanpa mengalami masalah.
2. MySQL memiliki kecepatan yang bagus dalam menangani *query* sederhana.
3. MySQL memiliki *operator* dan fungsi secara penuh dan mendukug perintah Select dan Where dalam perintah *query*
4. MySQL memiliki keamanan yang bagus karena beberapa lapisan sekuritas seperti level subnetmask, nama *host*, dan izin akses *user* dengan sistem perijinan yang mendetail serta sandi terenkripsi.

5. MySQL mampu menangani basis data dalam skala besar, dengan jumlah rekaman (*records*) lebih dari 50 juta dan 60 ribu tabel serta kurang lebih 5 milyar baris.
6. MySQL dapat melakukan koneksi dengan *client* menggunakan protokol TCP/IP, *Unix soket* (UNIX), atau *Named Pipes* (NT).
7. MySQL dapat mendeteksi pesan kesalahan pada *client* dengan menggunakan lebih dari dua puluh bahasa.
8. MySQL dapat berjalan stabil pada berbagai sistem *operasi* seperti *Windows*, Linux, FreeBSD Mac Os X Server, Solaris, Amiga, dan masih banyak lagi.
9. MySQL distribusikan secara *open source*, di bawah lisensi GPL sehingga dapat digunakan secara gratis.

Koneksi ke *database* digunakan untuk mengakses data-data yang ada dalam *database* tersebut. Berikut ini script untuk koneksi ke *database* : (Madcoms Madiun ; 2011 : 141)

```
Mysql_Connect(nama host, nama user, password);
```

Keterangan :

1. Nama *host* adalah lokasi tempat MySQL dipublikasikan. Pada contoh nama *host* diisi dengan *localhost*.
2. Nama *user* yaitu nama *user* yang terdaftar dalam MySQL yang digunakan untuk mengakses data yang ada dalam MySQL. Pada contoh nama *user* diisi dengan *root*.

3. *Password* adalah *password* yang digunakan untuk membuka *database* (PHPMyAdmin). Isi dengan *password* yang anda buat waktu instalasi AppServ.

Sebagai contoh koneksi ke database pada Dreamweaver sebagai berikut :

(Madcoms Madiun ; 2011 : 141)

```
<?
//mysql_connect(“localhost”,”user”,”password”)
$koneksi=mysql_connect(“localhost”,”root”,”1234”);
//untuk membuat koneksi ke server
If($koneksi){
        echo”Koneksi ke database berhasil”;
}else{
        echo”Koneksi ke database gagal”;
}
?>
```

II.8. *Entity Relationship Diagram* (ERD)

Merupakan suatu model untuk menjelaskan hubungan antar dua dalam basis data berdasarkan suatu persepsi bahwa *real word* terdiri dari *object-object* dasar yang memmpunyai hubungan atau antar *object-object* tersebut. Relasi antar *object* dengan menggunakan symbol-simbol grafis tertentu.

Model *entity relationship* adalah suatu penyajian dengan menggunakan *entity* dan *relationship*. Diperkenalkan pada tahun 1976 oleh P.P. Chen.

II.8.1. Komponen-komponen yang terdapat didalam *Entity Relationship* Model.

1. *Entity*

a. Adalah sesuatu yang dapat dibedakan dalam dunia nyata dimana informasi yang berkaitan dengannya dikumpulkan.

b. *Entity* set adalah kumpulan *entity* yang sejenis.

c. Symbol yang digunakan untuk *entity* adalah persegi panjang.

d. *Entity* set dapat berupa :

1. *Entity* yang bersifat fisik, yaitu *entity* yang dapat dilihat.

   Contohnya : rumah, kendaraan, mahasiswa, dosen, dan lain-lain.

2. *Entity* yang bersifat konsep atau logic, yaitu *entity* yang tidak dapat dilihat.

   Contohnya : pekerjaan, perusahaan, rencana, mata kuliah, dan lain-lain.

1. Simbol yang digunakan untuk *entity* adalah persegi panjang.

   Untuk melihat gambar *entity* ini, lihat pada gambar II.13. sebagai berikut :

Gambar II.13. *Entity*

(Sumber : Linda Marlinda, S. Kom; 2004:17)

2. *Relationship*

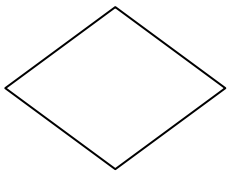
   a. Adalah hubungan yang terjadi antara satu atau lebih *entity*.

   b. *Relationship* tidak mempunyai keberadaan fisik, kecuali yang mewarisi hubungan antara *entity* tersebut.

   c. *Relationship* set adalah kumpulan relationship yang sejenis.

   d. Simbol yang digunakan adalah bentuk belah ketupat, *diamond* atau *rectangle.*

   Untuk melihat gambar *relationship* ini, lihat pada gambar II.14. sebagai berikut :



Gambar II.14. *Relationship*

(Sumber : Linda Marlinda, S. Kom; 2004:18)

3. Attribute

a. Adalah karakteristik dari *entity* atau *relationship* yang menyediakan penjelasan detail tentang atau *relationship* tersebut.

b. Attribute value (nilai atribute) adalah suatu data aktual atau informasi yang disimpan di suatu atribute di dalam suatu *entity* atau *relationship*.

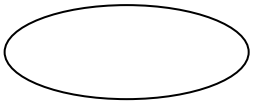
c. Terdapat dua jenis atribute, yaitu :

   1. *Indetifer (key)*, untuk menentukan suatu *entity* secara unik.

   2. *Descriptor (nonkey attribute)*, untuk menentukan karakteristik dari suatu *entity* yang tidak unik.

d. Simbol yang digunakan adalah bentuk oval

Untuk melihat gambar *attribute* ini, lihat pada gambar II.15. sebagai berikut :



Gambar II.15. *Attribute*

(Sumber : Linda Marlinda, S. Kom; 2004:18)

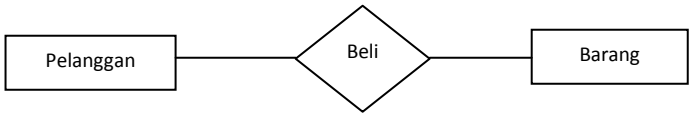
4. Indicator Tipe

a. *Indicator tipe associative object*

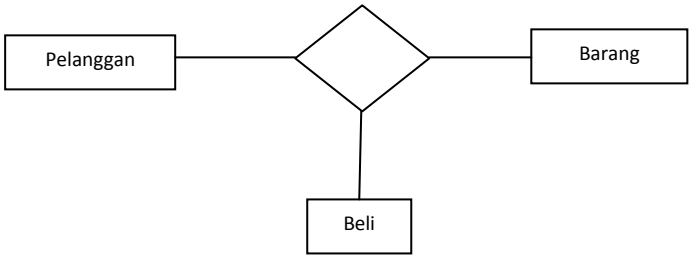
Berfungsi sebagai suatu objek dan suatu *relationship*

Untuk melihat gambar *indicator type* ini, lihat pada gambar II.16. sebagai berikut :

Contoh :



Menjadi :



Gambar II.16. *Indicator Tipe*

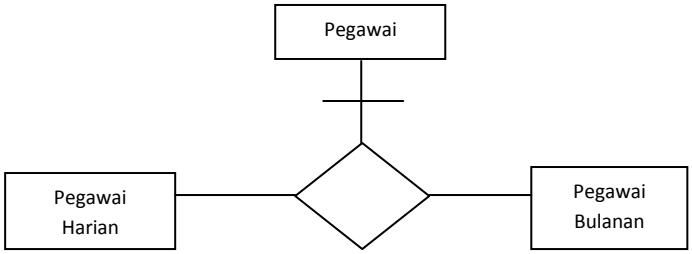
(Sumber : Linda Marlinda, S. Kom; 2004:19

b. Indicator tipe supertipe

Terdiri dari suatu object dan sub kategori atau lebih yang dihubungkan dengan dihubungkan dengan relationship yang tidak bernama (Linda Marlinda, S. Kom 2004:17-19).

Untuk melihat gambar *indicator tipe supertipe* ini, lihat pada gambar II.17. sebagai berikut :

Contoh :



Gambar II.17. *Indicator Tipe SuperTipe*

(Sumber : Linda Marlinda, S. Kom; 2004:19)

II.9. Normalisasi

Normalisasi adalah proses pengkelompokkan attribute-attribute dan suatu relasi sehingga membentuk *Well- Structure Relation.* Normalisasi merupakan proses pengkelompokkan elemen data menjadi suatu tabel-tabel menunjukan entity dan relasinya. Normalisasi ditemukan pada tahun 1970 oleh E. F. CODD.

II.9.1. Well-Structure Relation

Well- Structure Relation adalah sebuah *relation* dengan jumlah kerangkapan datanya sedikit (*Minimum amount of redundancy*), serta memberikan kemungkinan bagi user untuk melakukan *Insert, Delete,* dan *Modify* terhadap baris-baris data pada *relation* tersebut, yang tidak berakibat terjadinya *Error* atau INKONSENTENSI DATA, yang disebabkan oleh operasi-operasi tersebut.

Contoh :

Terdapat sebuah *Relation Course* dengan ketentuan sebagai berikut :

a. Setiap mahasiswa hanya boleh mengambil satu mata kuliah saja.
b. Setiap mata kuliah mempunyai uang kuliah yang standar (tidak tergantung pada mahasiswa yang mengambil mata kuliah tersebut).

Tabel II.2. *Relation Course*

STUDENT-ID	KODE-MTK	BIAYA
92130	CS-200	75
92200	CS-300	100
99250	CS-300	75
92425	CS-400	150
92500	CS-300	100
92575	CS-500	50

(Sumber : Linda Marlinda, S. Kom; 2004 :115)

Relation Course diatas merupakan sebuah *relation* yang sederhana dan terdiri dari 3 kolom/attribute (Linda Marlinda; 2004 : 115).

a. Bentuk tidak normal (*Unnormalized Form*)

Bentuk ini merupakan kumpulan data yang akan direkam, tidak ada keharusan mengikuti suatu format tertentu. Dapat saja data tidak lengkap atau terduplikasi. Data dikumpulkan apa adanya dengan saat menginput.

Contoh data :

Tabel II.3. Bentuk tidak normal (*Unnormalized Form*)

No_Siswa	Nama	Pa	Kelas 1	Kelas 2	Kelas 3
22890100	Shandy	Linda	1234	1543	1543
22890101	Susi	Riska	1234	1775	

(Sumber : Linda Marlinda, S. Kom; 2004 : 122)

Siswa yang punya nomor siswa, nama, dan pa mengikuti 3 mata pelajaran/kelas. Di sini ada perulangan kelas 3 kali ini bukan bentuk tidak 1 NF.

a. Bentuk Normal Ke Satu (1 NF/ *Fisrt Normal Form*)

Suatu relasi 1 NF dan hanya jika sifat dan setiap relasi attributenya bersifat *atomic*. Atom adalah zat terkecil yang masih memiliki sifat induknya. Bila dipecah lagi maka ia tidak memiliki sifat induknya.

Ciri-ciri 1 NF :

1. Setiap data dibentuk kedalam *flat file* data terbentuk per satu *record* nilai dan field berupa "*atomic value*".

2. Tidak ada *set attribute* yang berulang atau bernilai ganda,
3. Tiap *field*
4. hanya satu pengertian.

Tabel II.4. Bentuk Normal Ke Satu (1 NF/ *Fisrt Normal Form*)

No_Siswa	Nama	Pa	Kelas 1
22890100	Shandy	Linda	1234
22890100	Shandy	Linda	1543
22890101	Susi	Riska	1234
22890101	Susi	Riska	1775
22890101	Susi	Riska	1543

(Sumber : Linda Marlinda, S. Kom; 2004 : 122)

b. Bentuk Normal Ke Dua (2 NF/ *Second Normal Form*)

Bentuk normal kedua mempunyai syarat yaitu bentuk data telah memenuhi kriteria bentuk normal kesatu. *Attribute* bukan kunci haruslah bergantung secara fungsi pada *primary key*. Jadi, untuk membentuk normal kedua haruslah sudah ditentukan kunci-kunci *field.* Kunci *field* haruslah unik dan dapat mewakili *attribute* lain yang menjadi anggotanya. Misal : dari contoh relasi siswa pada 1 NF terlihat bahwa *primary key* adalah nomor siswa. Nama siswa dan pa bergantung fungsi pada no_siswa, tetapi kode_kelas bukanlah fungsi dan siswa dipecah menjadi 2 relasi :

Relasi siswa :

Tabel II.5. Bentuk Normal Kedua Relasi Siswa (2 NF/ *Second Normal Form*)

No_Siswa	Nama	Pa
22890100	Shandy	Linda
22890101	Susi	Riska

(Sumber : Linda Marlinda, S. Kom; 2004 : 123)

Relasi ambil_Kelas

Tabel II.6. Bentuk Normal Kedua Relasi ambil_Kelas (2 NF/ *Second Normal Form*)

No_Siswa	Kode Kelas
22890100	1234
22890100	1543
22890101	1234
22890101	1775
22890101	1543

(Sumber : Linda Marlinda, S. Kom; 2004 : 123)

c. Bentuk Normal Ketiga (3 NF/*Third Normal Form*)

Untuk menjadi bentuk normal ketiga maka relasi dalam bentuk normal kedua dan semua *attribute* bukan primer tidak punya hubungan yang transistif. Dengan kata lain, setiap *attribute* bukan kunci haruslah bergantung hanya pada *primary key* dan *primary key* secara menyeluruh.

Contoh pada bentuk normal kedua diatas termasuk juga bentuk normal ketiga karena seluruh attribute yang ada bergantung penuh pada kunci primernya.

d. *Boyee-Cood Normal Form* (BCNF)

BNCF mempunyai paksaan lebih kuat dan bentuk normal ketiga untuk menjadi BCNF, relasi harus dalam bentuk normal kesatu dan setiap *attribute* harus bergantung fungsi pada *attribute superkey.*

Pada contoh di bawah ini terdapat relasi seminar dengan ketentuan sebagai berikut :

1. Kunci primer adalah no_siswa +seminar
2. Siswa bopleh mengambil satu atau dua seminar.
3. Setiap siswa dibimbing oleh salah satu di antara 2 instruktur seminar tersebut.
4. Setiap instruktur boleh hanya mengambil satu seminar saja.

Pada contoh ini no_siswa dan seminar menunjuk seorang instruktur :

Relasi seminar

Tabel II.7. *Boyee-Cood Normal Form* (BCNF)

No_Siswa	Seminar	Instruktur
22890100	2281	Si doel
22890101	2281	Pak tile
22890102	2291	Mandra
22890101	2291	Basuki

22890109	2291	Basuki

(Sumber : Linda Marlinda, S. Kom; 2004 : 124)

Bentuk relasi seminar adalah bentuk normal ketiga, tetapi tidak BCNF karena nomor seminar masih tergantung fungsi pada instruktur. Jika setiap instruktur dapat mengajar hanya pada satu seminar saja, maka seminar bergantung fungsi pada satu *attribute* bukan *superkey* seperti disyaratkan oleh BCNF. Maka relasi seminar haruslah dipecah menjadi dua yaitu :

Tabel II.7. *Boyee-Cood Normal Form* (BCNF)

Relasi Pengajar

Instruktur	**Seminar**
Si doel	2281
Pak tile	2281
Mandra	2291
Basuki	2291

No_Siswa	**Instruktur**
22890100	Si doel
22890101	Pak tile
22890102	Mandra
22890101	Basuki
22890109	Basuki

(Sumber : Linda Marlinda, S. Kom; 2004 : 124)

e. Bentuk Normal Ke Empat (4NF)

Relasi R adalah bentuk 4 NF jika dan hanya jika relasi tersebut juga termasuk BCNF dan semua ketergantungan *multivaluei* adalah juga ketergantungan fungsional.

f. Bentuk Normal Kelima (5 NF)

Disebut juga PINF (*Projection Join Normal Form*) dan 4 NF dilakukan dengan menghilangakan ketergantungan *join* yang merupakan kunci kandidat (Linda Marlinda, S. Kom ; 2004 : 122-125).