2. TEORI PENUNJANG

2.1. HTML

HTML (*Hypertext Markup Language*) merupakan *plain-text* (atau lebih dikenal sebagai ASCII) yang digunakan untuk membangun suatu halaman *web* dan dapat dibuat menggunakan beberapa *text editor*, seperti *Emacs* atau *Vi* pada UNIX, *Simple text* pada *Macintosh* atau *Notepad* pada *Windows*. Dengan HTML, teks, gambar, suara serta *link* dapat digabungkan menjadi satu.

HTML bukan merupakan bahasa pemrograman, karena seperti tercermin dari namanya, HTML adalah suatu bahasa *mark-up* yang digunakan untuk melakukan *mark-up* atau penandaan terhadap sebuah dokumen teks. Tanda tersebut digunakan untuk menentukan format atau *style* dari teks yang ditandai.

Ciri utama dari *file* HTML adalah ekstensi .htm, .html, atau .shtml. halaman *web* yang dibuat dengan skrip HTML murni, tanpa ditambah skrip lain atau bahasa lain seperti VB*Script*, akan bersifat statis. Halaman *web* tersebut hanya dapat dibaca, tidak dapat diubah maupun dieksekusi oleh *user* lain. Selain itu, HTML juga memiliki sifat fleksibel karena dapat dapat dikombinasikan dengan *skript* atau bahasa pemrograman lainnya.

File HTML merupakan *file* teks biasa yang mengandung *tag* dengan ekstensi .htm atau .html. *Tag* adalah penandaan yang digunakan dalam HTML. *Tag* HTML ini terdiri dari tanda lebih kecil (<), nama *tag* dan tanda lebih besar (>). Dalam sebuah *file* HTML harus mengandung sebuah struktur sebagai berikut:

```
<HTML>
..
</HTML>
```

Tag <HTML> tersebut harus diletakan pada bagian paling awal. *Tag* </HTML> harus diletakan pada bagian paling akhir dan penulisannya tidak bersifat *case sensitive*. Penulisan tag dengan huruf kapital hanya untuk mempermudah perbedaan antara teks biasa dengan *tag*.

Secara lengkap, *file* HTML biasanya mempunyai bagian *head* dan bagian *body*, jadi struktur lengkapnya adalah sebagai berikut:

```
<HTML>
<HEAD>
...
</HEAD>
<BODY>
...
</BODY>
</HTML>
```

Bagian *head* umumnya berisi informasi mangenai dokumen tersebut, misalnya judul dokumen, versi HTML yang digunakan, dan lain sebagainya. Sedangkan bagian *body* berisi *layout* atau desain halaman *web*.

2.2. PHP

PHP dikenal sebagai bahasa scripting yang menyatu dengan *tag-tag* HTML, dieksekusi di *server*, dan digunakan untuk membuat halaman *web* yang dinamis seperti halnya *Active Server Pages* (ASP) atau *Java Server Pages* (JSP).

Lain halnya dengan HTML atau *JavaScript*, bahasa pemrograman PHP tidak menampilkan *script* dari program saat suatu halaman HTML ditampilkan, melainkan hanya menampilkan hasil pengeksekusian *file* PHP-nya saja.

PHP diciptakan oleh Rasmus Lerdorf pada tahun 1994 dan bersifat *open source*. Sampai bulan Januari 2001, PHP sudah digunakan oleh kurang lebih 5 juta *domain* dan terus berkembang hingga saat ini. Jumlah *domain* sampai saat ini yang menggunakan PHP dapat dilihat pada http://www.php.net/usage.php.

Oleh karena bersifat *open source*, *source code* PHP dapat digunakan, diganti atau diedit tanpa keharusan membayar biaya tertentu. Keunggulan PHP selain bersifat *open source* adalah sifat *cross platform*-nya, yaitu dapat dipakai dihampir semua *web server* yang ada di pasaran (*Apache*, AOL*Server*, fhttpd, dan lain-lain) yang dapat dijalankan pada berbagai system operasi (*Linux*, *Unix*, *Solaris*, *Windows*), seperti pada *platform Windows* dengan menggunakan *software* "PHP for *Windows*" dengan *web server* IIS pada *Windows* NT/2000 atau PWS pada *Windows* 98.

2.2.1. Sintaks PHP

Script PHP dapat terletak di dalam file HTML, WML, ataupun di *file* tersendiri. Perintah PHP hanya ditandai dengan *tag* pembuka `<? php` dan *tag* penutup `?>`. jadi perintah-perintah yang berada diantara *tag* tersebut akan dijalankan oleh PHP. Berikut ini adalah contoh sederhana aplikasi PHP.

Contoh aplikasi PHP:

```
<HTML>
<HEAD>
<TITLE> Pemrograman PHP </TITLE>
</HEAD>
<CENTER>
<?
Echo "Selamat menggunakan PHP";
?>
</CENTER>
</BODY>
</HTML>
```

Apabila *script* ini disimpan dalam format PHP dan kemudian dibuka menggunakan *browser* HTML, maka akan tampak dilayar tulisan "Selamat menggunakan PHP".

2.2.2. Perintah-perintah PHP

Struktur kontrol dari PHP terdiri dari berbagai macam perintah. Berikut ini adalah penjelasan mengenai perintah-perintah kontrol tersebut.

2.2.2.1. Perintah if/ if else

Perintah ini digunakan jika satu atau lebih operasi akan dilaksanakan jika syaratnya terpenuhi. Sintaks dasar dari perintah if adalah sebagai berikut:

```
If  (persyaratan) {
      Operasi program;
}
```

Atau

```
If  (persyaratan) {
      Operasi 1;
} else {
```

```
        Operasi 2;
}
```

Pada PHP juga terdapat beberapa operator perbandingan yang dapat dilihat pada table 2.1.

Tabel 2.1. Operator Perbandingan

Operator	Operasi	Deskripsi
`$a = = $b`	Sama dengan	Benar jika `$a` sama dengan `$b`
`$a = = = $b`	Identik	Benar jika `$a` sama dengan `$b`, dan memiliki tipe yang sama
`$a != $b`	Tidak sama dengan	Benar jika `$a` tidak sama dengan `$b`
`$a < $b`	Lebih kecil dari	Benar jika `$a` lebih kecil dari `$b`
`$a > $b`	Lebih besar dari	Benar jka `$a` lebih besar dari `$b`
`$a <= $b`	Lebih kecil atau sama dengan	Benar jika `$a` lebih kecil atau sama dengan `$b`
`$a >= $b`	Lebih besar atau sama dengan	Benar jika `$a` lebih besar atau sama dengan `$b`

PHP juga mempunyai operator logika seperti tampak pada tabel 2.2. berikut.

Tabel 2.2. Operator Logika

Operator	Operasi	Deskripsi
`$a` and `$b`	And	Benar jika `$a` dan `$b` keduanya benar
`$a` or `$b`	Or	Benar jika salah satu atau keduanya `$a`, `$b` bernilai benar
`$a` xor `$b`	Xor	Benar jika salah satu `$a`, `$b` bernilai benar, tetapi salah jika keduanya bernilai benar.
`!$a`	Not	Benar jika `$a` bernilai tidak benar

2.2.2.2. Perintah while

Perintah ini digunakan untuk melakukan *proses looping*. Ada dua jenis, yaitu *top tested* yaitu kondisi diperiksa terlebih dahulu setelah benar barulah perintah didalam dilakukan dan *bottom tested* yaitu perintah didalamnya dilakukan terlebih dahulu barulah kondisi diperiksa.

Sintaks untuk while top tested adalah sebagai berikut:

```
While (pernyataan) {
      Operasi program;
}
```

Sedangkan untuk while bottom tested adalah sebagai berikut:

```
Do{
      Operasi program;
} while (kondisi)
```

2.3 MySQL

MySQL adalah sebuah aplikasi *Relational Database Management Server* (RDBMS) bersifat open source yang memungkinkan data diakses dengan cepat oleh banyak pemakai secara bersamaan dan juga memungkinkan membatasi akses pemakai berdasarkan *privilege* (hak akses) yang diberikan. MySQL menggunakan bahasa SQL (*Structured Query Language*) yang merupakan bahasa standar pemrograman *database*.

MySQL dipublikasikan sejak tahun 1996, akan tetapi sebenarnya dikembangan sejak 1979. MySQL telah memenangkan penghargaan *Linux Journal Reader's Choice Award* selama tiga tahun. MySQL sekarang tersedia dibawah lisensi *open source*, tetapi ada juga lisensi untuk penggunaan MySQL yang bersifat komersial. Keunggulan dari MySQL ini adalah:

- Bersifat *open source*
- System software-nya tidak memberatkan kerja *server* atau computer karena dapat bekerja di *background*.

2.3.1 Perintah-perintah MySQL

Pada MySQL terdapat beberapa perintah. Perintah-perintah pada MySQL ini hamper sama dengan *database server* yang lain. Perintah-perintah pada MySQL itu antara lain adalah sebagai berikut:

1. *Create database* digunakan untuk membuat *database* pada *database server.* Sintaksnya adalah:

   ```
   Create database database_name
   ```

   `Database_name` adalah nama *database* yang akan dibuat

2. *Use database* digunakan untuk menunjuk pada *database* yang akan digunakan. Sintaksnya adalah:

   ```
   Use database_name
   ```

   `Database_name` adalah nama *database* yang akan digunakan

3. *Create table* digunakan untuk membuat tabel pada suatu *database.* Sintaksnya adalah:

   ```
   create table table_name
   (
      Column_1 column_type column_attributs,
      Column_2 column_type column_attributs,
      Primary key (column_name),
      Index index_name (column_name)
   )
   ```

   `Table_name` adalah nama tabel yang akan dibuat.

   `Column_1` adalah nama kolom yang akan dibuat dalam tabel.

   `Column_type` adalah tipe dari kolom tersebut, dapat berupa: *char, varchar, tinytext, text, mediumtext, longtext, enum, int, tinyint, mediumint, bigint, float, double, decimal, date, datetime, timestamp, time, year.*

   `Column_attributes` adalah atribut dari kolom tersebut, dapat berisi: *null, not null.*

   Tipe kolom yang digunakan adalah:

   - *Varchar(length)* yaitu digunakan untuk menyimpan karakter dengan maksimum panjang 255. variable length digunakan untuk membatasi panjang karakter.
   - *Int* yaitu digunakan untuk menyimpan numeric.

- *Date* yaitu digunakan untuk menyimpan tanggal dengan format YYYY-MM-DD.
- *Enum* yaitu digunakan untuk menyimpan karakter tertentu saja, misalnya: *enum* ('T','F') berarti hanya dapat menyimpan karakter T dan F saja.

4. *Insert* digunakan untuk menambahkan *record* pada tabel. Sintaksnya adalah :

   ```
   insert into table_name (column1, column2,...)values(value1, value2,..)
   ```

   Table_name adalah nama tabel yang akan ditambahkan *record*-nya.

   Column1, column2 adalah kolom yang akan ditambahkan.

   Value1, value2 adalah data yang akan ditambahkan.

5. *Update* digunakan untuk mengubah *record* yang sudah ada pada tabel. Sintaksnya adalah:

   ```
   Update table_name set column1=value1, column2=value2 where column=value
   ```

   Table_name adalah nama tabel yang akan diubah *record*-nya.

   Column1,column2 adalah kolom yang akan ditambahkan.

   Value1, value2 adalah data yang akan digantikan.

6. *Drop table* digunakan untuk menghapus tabel. Sintaksnya adalah:

   ```
   Drop table table_name
   ```

   Table_name adalah nama tabel yang akan dihapus.

7. *Show table* digunakan untuk menampilkan tabel yang telah dibuat dalam *database* yang aktif. Sintaksnya adalah:

   ```
   show tables
   ```

8. *Show field* digunakan untuk menampilakan seluruh *field* dalam suatu tabel. Sintaksnya adalah:

   ```
   Show field from table_name
   ```

   Table_name adalah nama tabel yang akan ditampilkan *field*-nya.

9. *Alter table* digunakan untuk menambahkan, mengubah, dan menghapus *field* dalam suatu tabel. Sintaknya adalah:

   - Untuk menambahkan:

     ```
     Alter table table_name add column column1 column_type column_attributes
     ```

     Table_name adalah nama tabel yang akan ditambahkan *field*-nya.

`Column1` adalah nama *field* baru, *column_type* adalah tipe kolom dan `column_attributes` adalah atribut kolom

- Untuk mengubah:

```
Alter table table_name change column1 column2 column_type
column_attributes
```

Table_name adalah nama tabel yang akan diubah *field*-nya. `Column1` adalah nama *field* yang akan diubah,`column2` adalah nama *field* baru. *column_type* adalah tipe kolom dan *column_attributes* adalah atribut kolom dari *field* yang baru.

- Untuk menghapus:

```
Alter table table_name drop column column1
```

Table_name adalah nama tabel yang akan dihapus *field*-nya. `Column1` adalah nama *field* yang akan dihapus.

10. *Select* digunakan untuk mengambil data dari suatu tabel. Sintaksnya adalah:

```
Select {*namafield} from namatabel [into tabeltujuan] [where
kondisi]
```

Tanda bintang (*) menunjukkan bahwa semua *field* yang akan dipilih.

11. *Delete* digunakan untuk menghapus sebuah *record* dari tabel. Sintaksnya adalah :

```
Delete from namatabel where kondisi
```

2.3.2. Koneksi PHP dengan MySQL

Untuk menghubungkan bahasa pemrograman PHP dengan MySQL dibutuhkan beberapa perintah khusus, yaitu:

a. Pembuatan koneksi antara *server* dari MySQL dengan *web server* tempat menyimpan halaman *web*. Perintahnya:

```
<?
Mysql_connect ("nama server mysql",login,password);
?>
```

b. Setelah terbentuk koneksi, selanjutnya dilakukan pemilihan *database* yang akan digunakan. Perintahnya:

```
<?
Mysql_select_db ("nama database");
?>
```

Jika belum pernah dibentuk suatu *database*, maka harus dibuat *database* baru dengan perintah sebagai berikut:

```
<?
Mysql_create_db ("nama database baru");
?>
```

c. Barulah kemudian dapat dilakukan perintah-perintah SQL yang lain, seperti: *Select, Update, Insert*, dan sebagainya. Contoh:

```
<?
Mysql query("perintah query");
?>
```

2.4. Server-side dan Client-side

Bahasa-bahasa yang dapat digunakan untuk membangun suatu *web* dapat dibedakan menjadi dua, yaitu *client-side* dan *server-side*. *Client-side* memiliki arti bahwa informasi yang disampaikan akan dieksekusi di *client* atau *browser*. Contoh bahasa yang bersifat *client-side* adalah HTML dan *JavaScript*. Sedangkan pada bahasa yang bersifat *server-side*, yaitu proses pengerjaan informasi akan dikirim dan diproses di server dari *website* tersebut. Contoh bahasa yang bersifat *server-side* adalah PHP, ASP (*Active Server Pages*), *Perl* dan JSP (*Java Server Pages*).

Penggunaan *client-side* bukan merupakan sesuatu yang mutlak dalam membangun aplikasi *web*, melainkan hanya dibutuhkan untuk menjalankan proses-proses yang dapat dilakukan pada *browser*. Tidak semua proses dapat dijalankan pada sisi *client*. Proses-proses yang membutuhkan akses ke *database* harus dilakukan di *server*.

Penggunaan *server-side* dan *client-side* harus disesuaikan dengan keperluan dari aplikasi *web* itu sendiri. *Server-side* digunakan untuk memproses segala sesuatu yang berhubungan dengan *server*, seperti *environment* dari server atau manipulasi data *database*.

Salah satu bahasa yang digunakan untuk membuat program di sisi *client* adalah *JavaScript* yang merupakan bahasa skrip berbasis objek yang bersifat *crossplatform*, disisipkan pada dokumen HTML dan dijalankan oleh *browser* pada saat dibaca. Penggunaan *client-side* tidak dianjurkan pada aplikasi *web* yang

membutuhkan data dalam jumlah banyak dan selalu berubah-ubah, atau menggunakannya untuk keperluan verifikasi *user* dan *password*.

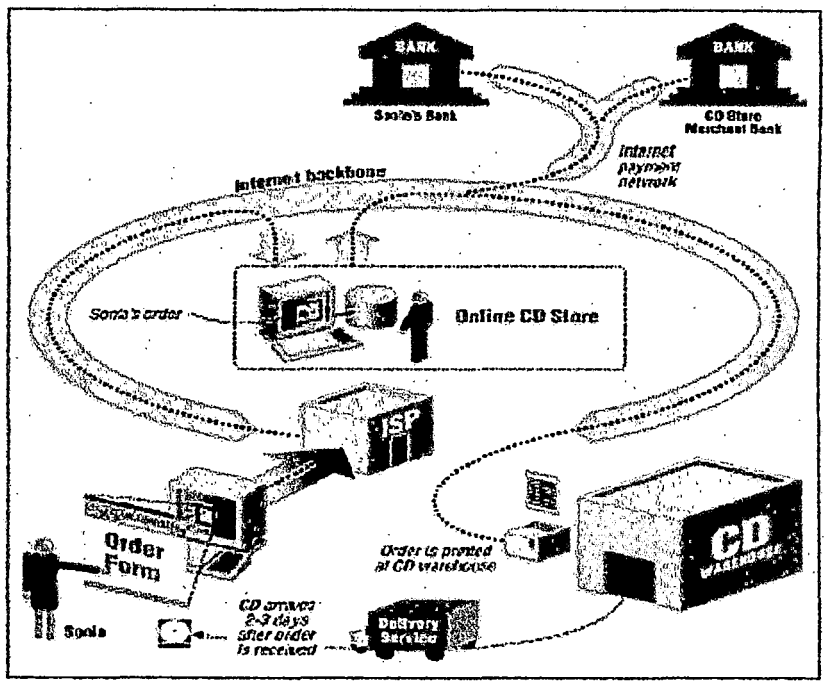
2.5. E-commerce

Definisi *E-Commerce* (*Electronic Commerce*) : *E-commerce* merupakan suatu cara berbelanja atau berdagang secara online atau direct selling yang memanfaatkan fasilitas Internet dimana terdapat website yang dapat menyediakan layanan "*get and deliver*". *E-commerce* akan merubah semua kegiatan marketing dan juga sekaligus memangkas biaya-biaya operasional untuk kegiatan trading (perdagangan). Proses yang ada dalam *E-commerce* adalah sebagai berikut :

- Presentasi electronis (Pembuatan Web site) untuk produk dan layanan.
- Pemesanan secara langsung dan tersedianya tagihan.
- Otomasi account Pelanggan secara aman (baik nomor rekening maupun nomor Kartu Kredit).
- Pembayaran yang dilakukan secara Langsung (online) dan penanganan transaksi

Keuntungan yang diperoleh dengan menggunakan transaksi melalui *E-commerce* bagi suatu perusahaan adalah sebagai berikut :

- Meningkatkan pendapatan dengan menggunakan online channel yang biayanya lebih murah.
- Mengurangi biaya-biaya yang berhubungan dengan kertas, seperti biaya pos surat,
  pencetakan, report, dan sebagainya.
- Mengurangi keterlambatan dengan mengunakan transfer elektronik / pembayaran
  yang tepat waktu dan dapat langsung dicek.
- Mempercepat pelayanan ke pelanggan, dan pelayanan lebih responsif.

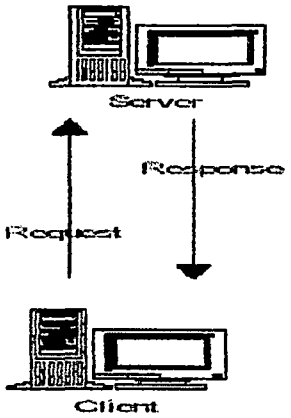


Gambar 2.1. Contoh aplikasi *E-commerce*: Pembelian CD Dengan Kartu Kredit

Sumber: Dian Adriana. *Pengenalan Pemrograman E-Commerce Dengan PHP dan MySQL*, 2003. <http://ilmukomputer.com/dian_ecommerce.pdf>

2.5.1. Arsitektur dan Konfigurasi Sistem

Arsitektur dasar dari aplikasi *web* ini adalah arsitektur *clientIserver*. Artinya pemrosesan aplikasi ini dijalankan melibatkan kedua sisi yakni sisi mesin *server* pusat dan sisi *client*. Hal ini berbeda dengan misalnya aplikasi *Microsoft Word* yang hanya melibatkan satu sisi saja yaitu sisi *client*. Atau bagi pengguna mesin VAX yang hanya menggunakan sisi *server* saja sedangkan sisi *client* hanya *dumb terminal* saja yang tidak melakukan pemrosesan apapun di sisi *client*.

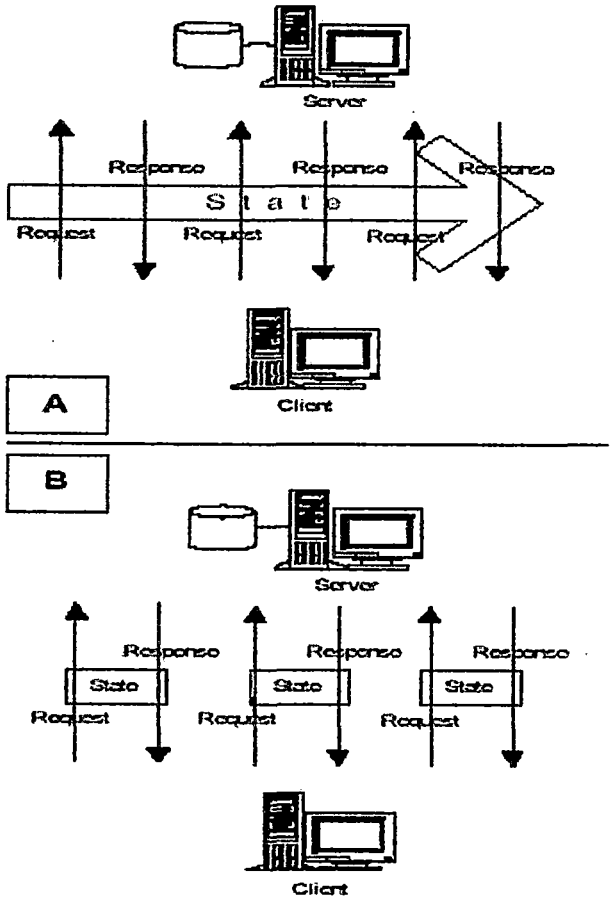


Gambar 2.2. Arsitektur *Client-Server*

Sumber: Dian Adriana. *Pengenalan Pemrograman E-Commerce Dengan PHP dan MySQL*,2003. <http://ilmukomputer.com/dian_ecommerce.pdf>

2.5.2. Stateless Web Server

Untuk aplikasi *E-Commerce* ini *web* server harus dapat mengingat siapa / identitas pengguna yang sedang melakukan browsing setiap halaman. Pada dasarnya aplikasi *web* dan *protocol* HTTP adalah *stateless.* Artinya setiap merespon sebuah *request* HTTP, server akan selesai bekerja (*complete*) dan tidak melakukan pencatatan apa yang telah dilakukan oleh pengguna sebelumnya dan terhadap siapa identitas pengguna. *Server* memperlakukan informasi permintaan (*request*) secara serial, satu persatu pada saat *request* masuk. Tidak ada koneksi permanen (*persistence*) yang berjalan setelah sebuah halaman telah selesai dilayani /dikerjakan.



Gambar 2.3. Perbandingan *State* dalam Sistem Aplikasi: A.*State* yang kontinyu dalam aplikasi *desktop*, dan B.*Stateless protokol* dalam aplikasi *web*

Sumber: Dian Adriana. *Pengenalan Pemrograman E-Commerce Dengan PHP dan MySQL*,2003. <http://ilmukomputer.com/dian_ecommerce.pdf>

Agar sebuah *situs web* mempunyai *memori / state*, dalam hal ini aplikasi ini mampu mengingat 'siapa memesan apa', beberapa informasi yang mengidentifikasi pengguna harus dikirim dengan setiap *request* halaman *web*. Informasi tersebut disimpan dengan menggunakan *session*.

Session tersebut dipergunakan untuk merekam / *tracking* aktivitas pengguna yang melalui sejumlah halaman pada *website*, misalnya pada jenis aplikasi *Shopping Cart* (kereta belanja). Direkam pula informasi identitas pengguna yang memiliki kereta belanja tersebut.

Dengan PHP, untuk penggunaan *session* ini mula-mula dilakukan pengaturan pada *file* php.ini yang menunjukkan *session* dimulai (*start*). Dengan ini PHP akan membuat suatu identifier unik dan *file* yang berkaitan, yang disimpan di *server* (lokasinya di atur di php.ini dan nilai *default*nya di direktori

/tmp). Kemudian pada saat pengguna berkunjung pada halaman-halaman *situs web*, semua informasi variabel yang dipilih oleh pengguna akan disimpan dalam *file* pada *server*, dan semua *script* yang dibutuhkan untuk melacak sebagai identifier unik.

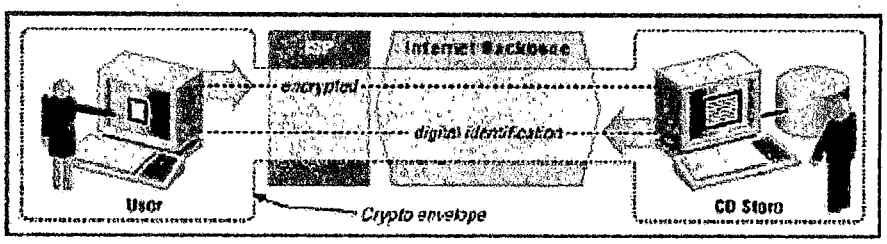
Implementasi *session* dapat mempergunakan *cookie* yang disimpan pada sisi *client*, atau dipropagasikan melalui alamat URL. Untuk penggunaan *cookie*, yaitu dengan *passing* variabel melalui *cookie* yang menyimpan informasi semua elemen barang belanja dan harganya. Jika menggunakan *cookie* untuk fungsi penyimpan informasi tadi, diperlukan membuat *string* unik yang akan diletakkan dalam *cookie*, dalam direktori di *server* akan terdapat sebuah *file* yang memiliki nama yang sama sebagai ID pengguna yang unik. Dalam *file* tersebut dapat disimpan semua *variabel* yang berkaitan dengan pengguna. Contohnya terdapat *array* berisi item-item barang yang ditambahkan oleh seorang pengguna ke dalam kereta belanjanya.

Terdapat keterbatasan penggunaan *cookie*, yakni bila *browser* pengguna di atur untuk menolak (*reject*) *cookie*. Metode lain yang dapat digunakan adalah dengan propagasi URL, yaitu dengan mengaktifkan *flag* –enable-trans-sid dalam konfigurasi PHP, hal ini berguna agar *session id* akan secara otomatis ditambahkan ke setiap *relative link* pada halaman-halaman *web* setiap kali *session* telah dimulai.

2.5.3. Enkripsi Public-Key/ Private-Key

Mesin di *web* menggunakan skema keamanan *Public-key/Private-key*. Artinya komputer yang akan berkomunikasi menggunakan data terenkripsi harus memiliki dua buah kunci untuk mengenkripsi data dan mendekripsinya. Pertama, *public-key* tersedia bagi siapa saja yang ingin melakukan komunikasi terhadapnya. Sehingga siapapun yang ingin melakukan komunikasi terhadap sebuah mesin secara *secure* akan memiliki salinan dari *Public key* mesin tersebut. Namun *public key* ini tidak cukup untuk dapat mendekripsi data, masih dibutuhkan *Private key* yang bersifat rahasia. Misalnya pada pemrosesan kartu kredit dengan sebuah bank, nasabah memiliki *Public key* bank tersebut dimana ia dapat melakukan

dekripsi informasi, namun masih diperlukan *Private key* yang disimpan oleh bank tersebut, untuk dapat melakukan dekripsi data.



Gambar 2.4. Pengiriman Data Terenkripsi antara Pengguna dengan *Server E-Commerce*

Sumber: Dian Adriana. *Pengenalan Pemrograman E-Commerce Dengan PHP dan MySQL,2003.* <http://ilmukomputer.com/dian_ecommerce.pdf>

2.5.4. Secure Protocol

Protokol HTTP secara alamiah bersifat terbuka terhadap penyusupan. Paket-paket data yang melintas melalui *router Internet* dapat disadap dan dibaca. Namun informasi kartu kredit diinginkan agar tidak mudah terbaca. Untuk itu dibutuhkan penggunaan *Secure Socket Layer* atau SSL. SSL adalah *protokol* tambahan dimana *key* dan sertifikat dari suatu situs *e-commerce* akan ditransfer ke *browser* atau ke *server* lain.

Melalui SSL, *browser* akan dapat memverifikasi sertifikat dari *situs* tersebut sehingga dapat mengetahui identitas pengirim sebenarnya. Tata cara enkripsi ini masih mengandung kelemahan yakni pada aspek sumber daya manusia apabila kurang jujur, yakni apabila terjadi akses tidak sah dilakukan oleh orang yang sudah berada dalam sistem.

2.5.5. Pemrosesan Kartu Kredit

Pemrosesan kartu kredit dilakukan oleh perusahaan yang khusus untuk itu, terdapat beberapa nama perusahaan yang cukup dikenal, namun semuanya memiliki kesamaan cara kerja. Mula-mula dikirim permintaan / *request* dengan informasi kartu kredit: nomor, tanggal kadaluarsa, alamat, dan sebagainya, dan kemudian perusahaan tersebut akan mengirimkan kode kembalian sebagai respon. Kode PHP di sini adalah berfungsi untuk membandingkan kode yang diterima dengan nilai yang didapat sebelumnya dari agen pemrosesan. Untuk aplikasi ini dipergunakan Simulasi Kartu Kredit sebagai pemroses kartu kredit.

2.6.*Entity Relationship Diagram* (ERD)

Entity Relationship Diagram (ERD) berfungsi untuk menggambarkan hubungan antar *entitas* dalam suatu sistem. Dalam ERD, *entitas* disimbolkan dengan kotak segiempat dan diberi nama dengan menggunakan kata benda tunggal. Contoh *entitas* seperti pada gambar 2.5.

Gambar 2.5. Simbol *Entitas* Dalam ERD

Suatu *entitas* memiliki atribut-atribut dengan satu atribut unik yang menjadi *primary key*, yaitu atribut unik yang membedakan satu individu dengan individu lainnya dalam satu *entitas* yang sama.

Relationship menjelaskan hubungan antara dua *entitas* dan diberi nama dengan menggunakan kata kerja. ERD juga harus dilengkapi dengan tanda yang menunjukan *conditionaly*. *Conditionaly* ada empat, yaitu *one-to-one, one-to-many, many-to-one, many-to-many.*