BAB 2
LANDASAN TEORI

2.1 *Cloud Computing*

2.1.1 Sejarah Perkembangan *Cloud Computing*

Cloud computing adalah hasil dari evolusi bertahap di mana sebelumnya terjadi fenomena *grid computing*, virtualisasi, *application service provision* (ASP) dan *Software as a Service (SaaS).* Konsep penyatuan *computing resources* melalui jaringan global sendiri dimulai pada tahun 60-an. Saat itu muncul "*Intergalactic computer network*" oleh J.C.R. Licklider, yang bertanggung jawab atas pembangunan ARPANET (*Advanced Research Projects Agency Network*) di tahun 1969. J.C.R. Licklider memiliki sebuah cita-cita di mana setiap manusia di dunia ini dapat terhubung dan bisa mengakses program dan data dari situs manapun, di manapun. Menurut Margaret Lewis, Direktur Marketing Produk AMD. "Cita-cita itu terdengar mirip dengan apa yang kini disebut dengan *cloud computing*". Para pakar komputasi lainnya juga memberikan penambahan terhadap konsep ini, di antaranya John McCarthy yang menawarkan ide mengenai jaringan komputasi yang akan menjadi infrastruktur publik, sama seperti *the service bureaus* yang sudah ada sejak tahun '60-an.

Semenjak tahun '60-an, *cloud computing* telah berkembang berdampingan dengan perkembangan Internet dan Web. Namun karena terjadi perubahan teknologi *bandwidth* yang cukup besar pada tahun 1990-an, maka Internet lebih dulu berkembang dibanding *cloud computing*. Dan kini teryata terlihat bahwa pendorong utama *cloud computing* adalah karena adanya revolusi Internet. Salah satu batu loncatan yang cukup drastis adalah dengan adanya Salesforce.com di tahun 1999, yang merupakan pencetus pertama aplikasi perusahaan dijalankan melalui Internet. Perkembangan berikutnya adalah adanya Amazon *Web Services* di tahun 2006, di mana

dengan teknologi *Elastic Compute Cloud* (EC2), terdapat situs layanan web yang di komersialkan yang memungkinkan perusahaan kecil dan individu untuk menyewa komputer atau server, agar dapat menjalankan aplikasi komputer mereka.

Batu lompatan besar lainnya datang di tahun 2009 dengan Web 2.0 mencapai puncaknya. Google dan lainnya memulai untuk menawarkan aplikasi *browser-based* untuk perusahaan besar, seperti Google Apps. "Kontribusi yang paling penting dari komputasi *cloud* adalah munculnya "*killer apps*" dari penguasa teknologi seperti Microsoft dan Google. Ketika perusahaan tersebut mengirimkan layanan dalam bentuk yang mudah untuk di konsumsi, efek penerimaannya menjadi sangat luas", menurut Dan Germain, *Chief Technology IT provider Cobweb Solution*. "Faktor utama lainnya yang mempengaruhi berkembangnya komputasi *cloud* antara lain matangnya teknologi visual, perkembangan universal *banwidth* berkecepatan tinggi, dan perangkat lunak universal", menurut Jamie Turner pelopor komputasi *cloud*. Turner menambahkan, "*cloud computing* sudah menyebar luas hingga kepada para pengguna Google Doc. Kita hanya dapat membayangkan betapa besarnya ruang lingkup yang sudah di capai. Apa saja dapat di lakukan dan dikirimkan melalui *cloud"* [2].

2.1.2 Definisi *Cloud Computing*

Cloud Computing adalah model untuk menciptakan kenyamanan dalam mengakses jaringan sesuai keperluan ke dalam wadah bersama sumber daya komputasi yang dapat dikonfigurasi dengan cepat dan dirilis dengan upaya manajemen yang minimal atau minimal interaksi antar penyedia jasa manajemen [3].

2.1.3 Model Pengembangan *Cloud Computing*

Dalam penerapannya, ada beberapa tipe model pengembangan cloud computing. Model pengembangan ini ditinjau dari cakupan dan ruang lingkupnya

1. Public Cloud

Merupakan infrastruktur layanan *cloud computing* yang dibuat untuk melayani berbagai macam pengguna, di mana mereka menggunakan secara bersama / digunakan untuk umum, yang mungkin dimiliki, dikelola, dan dioperasikan oleh bisnis, akademik, ataupun organisasi pemerintah [3].

2. Private Cloud

Merupakan infrastruktur layanan *cloud computing*, yang dioperasikan hanya untuk sebuah organisasi tertentu / digunakan untuk pribadi. Biasanya dimiliki, dikelola, dan dioperasikan oleh organisasi atau pihak ketiga [3].

3. Community Cloud

Dalam model ini, sebuah infrastruktur *cloud* digunakan bersama-sama oleh beberapa organisasi yang memiliki kesamaan kepentingan, misalnya dari sisi misinya, atau tingkat keamanan yang dibutuhkan, dan lainnya [3].

4. Hybrid Cloud

Untuk jenis ini, infrastruktur *Cloud* yang tersedia merupakan komposisi dari dua atau lebih infrastruktur *Cloud* (*private*, *community*, atau *public*). meskipun secara entitas mereka tetap berdiri sendiri, tapi dihubungkan oleh suatu teknologi / mekanisme yang memungkinkan portabilitas data dan aplikasi antar *Cloud* itu. Misalnya, mekanisme *load balancing* yang antar *Cloud*, sehingga

alokasi sumber daya bisa dipertahankan pada level yang optimal Secara umum *Cloud* ini mirip dengan cairan yang mudah dapat memperluas dan mengisi. Bersifat Elastisitas, yang berarti bahwa pengguna dapat meminta sumber daya tambahan pada permintaannya dan dengan mudah melepaskan sumber daya ketika mereka tidak lagi diperlukan. Elastisitas ini salah satu alasan utama individu, bisnis, dan pengguna IT berpindah ke awan (*Cloud*). Pada data center tradisional itu selalu mungkin untuk menambah dan melepaskan sumber daya. Untuk evolusi *Cloud Computing* ini sudah berlangsung dan sepenuhnya dapat mengubah cara perusahaan menggunakan teknologi untuk layanan pelanggan, mitra, dan pemasok. Beberapa perusahaan sudah memiliki sumber daya IT hampir seluruhnya di *Cloud* [3].

2.1.4 Karakteristik *Cloud Computing*

1. On Demand

Pengguna bisa mengkonfigurasikan secara mandiri layanan yang ingin dipakai melalui sebuah sistem, tanpa perlu interaksi langsung dengan pihak *cloud provider*. Konfigurasi layanan yang dipilih ini harus tersedia saat itu juga secara otomatis. [3].

2. Rapid elasticity

Kapabilitas dari layanan *cloud provider* bisa dipakai oleh *cloud consumer* secara dinamis berdasarkan kebutuhan. *Cloud consumer* bisa menaikkan atau menurunkan kapasitas layanan. Kapasitas layanan yang disediakan ini biasanya tidak terbatas, dan *service consumer* bisa dengan bebas dan mudah memilih kapasitas yang diinginkan setiap saat [3].

3. Broadband Network Access

Kapabilitas layanan dari *cloud provider* tersedia lewat jaringan dan bisa diakses oleh berbagai jenis perangkat, seperti *smartphone*, tablet, laptop dan *workstation* [3].

4. Resources Pooling

Sumber daya komputasi (*storage, CPU, memory, network bandwidth*) yang dikumpulkan oleh penyedia layanan (*service provider*) untuk memenuhi kebutuhan banyak pelanggan (*service consumers*) dengan model *multi-tenant*. Sumber daya komputasi ini bisa berupa sumber daya fisik ataupun virtual dan juga bisa dipakai secara dinamis oleh para pelanggan untuk mencukupi kebutuhannya [3].

5. Measured services

Tersedia layanan untuk mengoptimasi dan memonitor layanan yang dipakai secara otomatis. Dengan monitoring sistem ini, kita bisa melihat berapa *resources* komputasi yang telah dipakai, seperti: *bandwidth , storage, processing* dan jumlah pengguna aktif.. Layanan monitoring ini sebagai bentuk transparansi antara *cloud provider* dan *cloud consumer* [3].

2.1.5 Model Layanan *Cloud Computing*

Layanan pada *Cloud Computing* dibagi menjadi tiga model layanan, yaitu *Software as a Service, Platform as a Service, dan Infrastructure as a Service.* Berikut penjelasan dari ketiga jenis layanan :

1. Software As A Service (SAAS)

Software as a Service (SaaS) adalah layanan dari *Cloud Computing* dimana pelanggan dapat menggunakan *software* (perangkat lunak) yang telah

disediakan oleh *cloud provider*. Pelanggan cukup tahu bahwa perangkat lunak bisa berjalan dan bisa digunakan dengan baik [3].

2. Platform Aas A Service (PAAS)

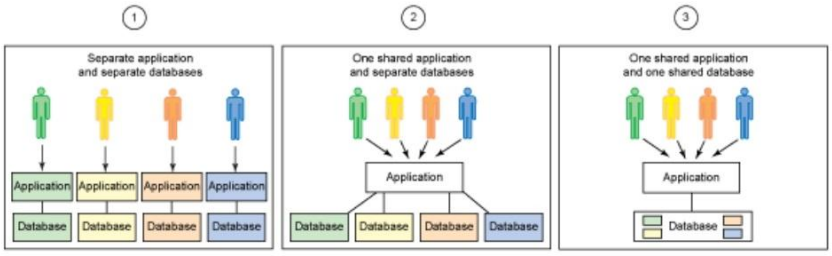
Pada *platform as a service* konsumen / pengguna diberikan sebuah *platform* untuk mengembangkan sampai implementasi sistem. Konsumen / pengguna harus untuk membuat dan mengimplementasikan sistemnya sendiri. umumnya *tools* untuk *development* disediakan dalam bentuk *web application* [3].

3. Infrastructure As A Service (IAAS)

Layanan ini menyediakan akses virtualisasi computer, sumber daya dari hardware termasuk mesin, jaringan maupun penyimpanan data [3].

2.1.6 *Multi Tenant Models*

Multi tenant models adalah arsitektur atau perancangan perangkat lunak di dalam *software as a service,* berikut beberapa cara *multi tenant models* :



Gambar 2.1 *Multi-tenant models* [4]

1. *Separate application and separate databases* (aplikasi dan database terpisah)
   Keuntungan: mudah untuk memastikan data klien/pengguna tidak tercampur dengan klien/pengguna lain, mudah untuk

melakukan modifikasi aplikasi ke klien/pengguna tertentu, dari sisi keamanan lebih baik.
Kekurangan: untuk update *database* harus memperbarui secara manual untuk setiap klien/pengguna [4].

2. *One shared application and separate databases* (satu aplikasi bersama dan *database* terpisah)
Keuntungan: *update* aplikasi dan memastikan data klien/pengguna tidak tercampur dengan klien/pengguna yang lain
Kekurangan: untuk *mengupdate databse* harus memperbaharui setiap *database* [4].

3. *One shared application and one shared databases* (aplikasi dan database digunakan bersama-sama)
Keuntungan : mudah untuk mengelola *database* & *update*, tidak ada duplikasi.
Kekurangan : jika melewatkan sebuah "ClientID" dalam *query* SQL, dapat menyebabkan masalah dengan klien lain [4].

2.2 *E-Commerce*

Electronic Commerce atau yang lazim dikenal dengan *e-commece* didiefenisikan sebagai cara untuk menjual dan membeli barang-barang (dan jasa) melalui sistem elektronik seperti jaringan internet. Sebagian besar kegiatan *e-commerce* dilakukan sepenuhnya scara elektronik atau item virtual seperti premi ataupun untuk mengakses konten pada situs web, namun sebagian besar perdagangan elektronik melibatkan transportasi fisik item dalam berbagai cara. Pengecer online kadang-kadang dikenal sebagai *e-tailers line ritel* dan kadang-kadang dikenal sebagai *e-tail*. Hampir semua pengecer besar telah memanfaatkan *e-commerce* pada *World Wide Web.Elecronic Commerce* pada umumnya dianggap sebagai aspek

penjualan e-bisnis. Ia juga terdiri dari pertukaran data untuk menfasilitasi pembiayaan dan pembayaran aspek dari transaksi bisnis [5].

2.3 *Content Management System* (CMS)

CMS secara sederhana dapat diartikan sebagai berikut :
sebuah sistem yang memberikan kemudahan kepada para penggunanya dalam mengelola dan mangadakan perubahan isi sebuah website dinamis tanpa sebelumnya dibekali pengetahuan tentang hal-hal yang bersifat teknis. Dengan demikian, setiap orang, penulis maupun editor, setiap saat dapat menggunakannya secara leluasa untuk membuat, menghapus atau bahkan memperbaharui isi website tanpa campur tangan langsung dari pihak *webmaster* [6].

2.4 Model Analisis

2.4.1 Entity Relationship Diagram (ERD)

Entity Relationship Diagram atau biasa dikenal dengan diagram E-R secara grafis menggambarkan isi sebuah *database.* ERD terbagi atas tiga komponen, yaitu entitas (*entity*), atribut (*atribute*), dan relasi atau hubungan (*relation*). Secara garis besar entitas merupakan dasar yang terlibat dala sistem. Atribut atau field berperan sebagai penjelas dari entitas, dan relasi atau hubungan menunjukkan hubungan yang terjadi antara dua entitas [7].

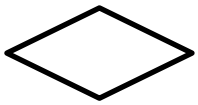
1. Entitas (*Entity*)
   Entitas (*entity*) menunjukkan objek-objek dasar yang terkait di dalam sistem. Objek dasar dapat berupa orang, benda atau hal lain yang keterangannya perlu disimpan dalam basis data [7].
2. Atribut (*Atribute*)
   Atribut sering juga disebut sebagai properti (*property*) merupakan keterangan-keterangan yang terkait pada sebuah entitas yang perlu disimpan sebagai basis data. Atribut berfungsi sebagai penjelas sebuah entitas [7].

3. Relasi (*Relation*)

Relasi atau hubungan adalah kejadian atau transaksi yang terjadi di antara dua entitas yang keterangannya perlu disimpan dalam basis data [7]

Notasi-notasi simbolik di dalam diagram E-R yang digunakan sebagai berikut [8] :

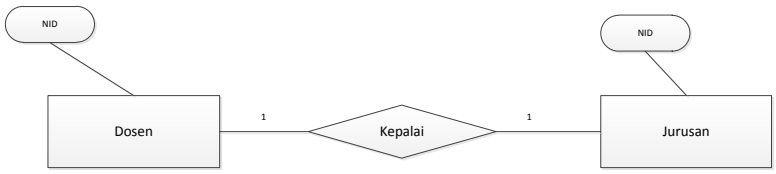
Tabel 2.1 Notasi yang digunakan pada *Entity Relational Diagram* (ERD)

No	Nama Simbol	Fungsi	Simbol
1	Persegi panjang	Menyatakan suatu himpunan entitas	
2	Lingkaran/Elip	Menyatakan atribut (atribut yang berfungsi sebagai key digaris bawahi)	
3	Belah ketupat	Menyatakan himpunan relasi	
4	Garis	Sebagai penghubung antara himpunan relasi dengan himpunan entitas dan himpunan entitas dengan atributnya	

Di dalam *Entity Relational Diagram* (ERD) terdapat kardinalitas (*cardinality*) kardinalitas relasi menunjukan jumlah maksimun tupel yang dapat berelasi dengan entitas pada entitas lain, terdapat 3 macam kardinalitas relasi, yaitu [8] :

1. *One to One*

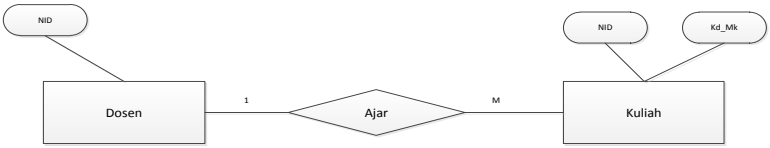
   Tingkat hubungan satu ke satu, dinyatakan dengan satu kejadian pada entitas pertama, hanya mempunyai satu hubungan dengan satu kejadian pada entitas yang kedua dan sebaliknya [8]. Berikut contoh *one to one*



Gambar 2.2 Diagram kardinalitas *One to One*

2. *One to Many* atau *Many to One*

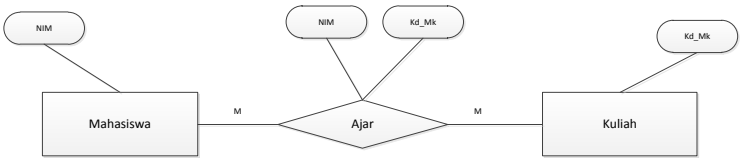
   Tingkat hubungan satu ke banyak adalah sama dengan banyak ke satu. Tergantung dari arah mana hubungan tersebut dilihat [8]. Berikut contoh *one to many* atau *Many to One*



Gambar 2.3 Diagram kardinalitas *One to many*

3. Many to Many

   Tingkat hubungan banyak ke banyak terjadi jika tiap kejadian pada sebuah entitas akan mempunyai banyak hubungan dengan kejadian pada entitas lainnya [8] . Berikut contoh *many to many*



Gambar 2.4 Diagram kardinalitas *many to many*

2.4.2 Diagram Konteks

Diagram konteks adalah diagram yang terdiri dari suatu proses dan menggambarkan ruang lingkup suatu sistem. Diagram konteks merupakan level tertinggi dari DFD yang menggambarkan seluruh input ke sistem atau output dari sistem. Diagram konteks akan memberi gambaran tentang keseluruhan sistem. Sistem dibatasi oleh Boundary (dapat digambarkan dengan garis putus). Dalam diagram konteks hanya ada satu proses. Tidak boleh ada *store* dalam diagram konteks [8].

2.4.3 Data Flow Diagram

DFD merupakan alat perancangan sistem yang berorientasi pada alur data dengan konsep dekomposisi yang dapat digunakan untuk penggambaran analisis maupun rancangan sistem yang mudah dikomunikasikan kepada pemakai maupun pembuat program [8]. Bagian dari DFD adalah:

1. Arus Data (*Data Flow*)

   Arus data merupakan tempat mengalirnya informasi dan digambarkan dengan garis yang menghubungkan komponen dari sistem [8].

2. Proses (*process*)

   Proses merupakan apa yang dikerjakan oleh sistem. Proses dapat mengolah data atau aliran data masuk menjadi aliran data ke luar. Proses berfungsi mentransformasikan satu atau beberapa data masukan menjadi satu atau beberapa data keluaran sesuai dengan spesifikasi yang diinginkan. Setiap proses memiliki satu atau beberapa masukan serta menghasilkan satu atau beberapa data keluaran [8].

3. Kesatuan Luar (*External Entity*)

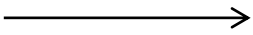
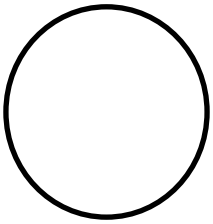
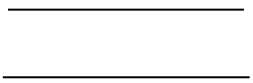
   Kesatuan luar merupakan kesatuan di lingkungan luar sistem yang dapat berupa orang, organisasi atau sistem lain yang akan memberikan masukan (*input*) atau menerima keluaran (*output*) dari sistem [8].

4. Simpanan Data (*Data Store*)

Simpanan data (*data store*) merupakan tempat penyimpanan data pengikat data yang ada di dalam sistem. *Data store* dapat disimbolkan dengan sepasang dua garis sejajar atau dua garis dengan salah satu sisi samping terbuka. Proses dapat mengambil data dari atau memberikan data ke *database*. [8].

Berikut tabel simbol *data flow diagram* (DFD) yang sering digunakan

Tabel 2.2 simbol *data flow diagram* (DFD)

No	Nama SImbol	Simbol
1	Arus Data	
2	Proses	
3	Simpanan Data (*Data Store*)	
4	Entitas Luar	

2.4.4 Kamus Data

Kamus data atau data *dictionary* adalah katalog fakta tentang data dan kebutuhan-kebutuhan informasi dari suatu sistem informasi. Dengan menggunakan kamus data, analisis sistem dapat mendefinisikan data yang mengalir di sistem dengan lengkap. Pada tahap perancangan sistem, kamus

data dapat digunakan untuk merancang *input*, *output*, dan merancang *database program*. Kamus data dibuat berdasarkan arus data yang ada [8].

2.5 *Database* (Basis Data)

Beberapa definisi basis data sebagai berikut :

1. *Database* adalah sekumpulan data *store* (bias dalam jumlah yang besar) yang tersimpan dalam magnetic disk, optical disk, magnetic drum atau media penyimpanan [8].
2. *Database* adalah sekumpulan program-program aplikasi umum yang bersifat "batch" yang mengeksekusi dan memproses data secara umum (seperti pencarian,peremajaan,penambahan, dan penghapusan terhadap data) [8].
3. Database terdiri dari data yang akan digunakan atau diperuntukan terhadap banyak 'user' (baik menggunakan teknik pemrosesan yang bersifat batch atau on-line) akan menggunakan data tersebut sesuai dengan tudas dan fungsinya, dan 'user' lain dapat juga menggunakan data tersebut dalam waktu yang bersamaan. [8].

2.5.1 MySQL

MySQL adalah sebuah program *database server* yang mampu menerima dan mengirimkan datanya dengan sangat cepat, *multi user* serta menggunakan perintah standar SQL (*Structured Query Language*).

MySQL dapat juga berperan sebagai *client* sehingga sering disebut database *client/server*, yang *open source* dengan kemampuan dapat beroperasi di OS (*operating system*) manapun [9].

Contoh :

1. DDL (*Data Definition language*)
    a. *Create* : mysql > CREARE DATABASE nama_*database*;
    b. *Drop* : mysql > DROP DATABASE nama_*database*;
    c. *Alter* : mysql > ALTER nama_tabel alter_spesifikasi kondisi;

2. DML (*Data Manipulation Language*)
    a. *INSERT* : mysql > INSERT INTO nama_tabel VALUES (isi_field);
    b. SELECT :mysql> SELECT * from nama_tabel ;
    c. *UPDATE* : mysql > UPDATE nama_tabel SET isi_field=’data_baru’;
    d. *DELETE* : *DELETE* nama_tabel *WHERE* kondisi;

2.5.2 SQL (Structure Query Language)

SQL adalah sebuah bahasa permintaan *database* yang terstruktur. Bahasa SQL dibuat sebagai bahasa yang dapat merelasikan beberapa tabel dalam *database* maupun merelasikan antar *database*. Bahasa SQL ditulis langsung dalam sebuah program *databases* sehingga pengguna dapat melihat langsung permintaan yang diinginkan sekaligus melihat hasilnya.
SQL dibagi menjadi 2 bentuk query, yaitu [9]:

1. **DDL (*Data Definition Language*)**

   DDL adalah sebuah metode *query* sql yang berguna untuk mendefinisikan data pada sebuah *database*, adapun *query* yang dimiliki adalah [9]:

    a. CREATE : Digunakan untuk melakukan pembuatan tabel dan *database* [9].
    b. DROP : Digunakan untuk melakukan penghapusan tabel maupun *database* [9].
    c. ALTER : Digunakan untuk melakukan perubahan struktur tabel yang telah dibuat, baik menambah *field* (*add*), mengganti nama *field* (*change*) ataupun menamakannya kembali (*rename*) serta menghapus (*drop*) [9].

2. **DML (*Data Manipulation Language*)**

   DML adalah sebuah metode *query* yang dapat digunakan apabila DDL telah terjadi, sehingga fungsi dari *query* ini adalah untuk

melaukan pemanipulasian *database* yang telah ada atau telah dibuat sebelumnya. Adapun *query* yang termasuk di dalamnya adalah [9]:

a. INSERT : Digunakan untuk melakukan penginputan pada tabel di *database* [9].
b. UPDATE : Digunakan untuk melakukan pengubahan atau peremajaan terhadap data yang ada pada tabel [9].
c. DELETE : Digunakan untuk melakukan penghapusan data pada tabel. Penghapusan ini dapat dilakukan secara sekaligus (seluruh isi tabel) maupun hanya beberapa *Recordset* [9].

2.6 Bahasa Pemrograman

2.6.1 HTML (*Hypertext Markup Language*)

Hypertext Markup Language adalah bahasa markup yang umum digunakan untuk membuat halaman web. Sebenarnya HTML bukanlah sebuah bahasa pemrograman. Apabila di tinjau dari namanya, HTML merupakan bahasa markup atau penandaan terhadap sebuah dokumen teks. Tanda tersebut di gunakan untuk menentukan format atau style dari teks yang di tandai. HTML dibuat oleh Tim Berners-Lee ketika masih bekerja untuk CERN dan dipopulerkan pertama kali oleh browser Mosaic. Selama awal tahun 1990 HTML mengalami perkembangan yang sangat pesat. Setiap pengembangan HTML pasti akan menambahkan kemampuan dan fasilitas yang lebih baik dari versi sebelumnya. Sebelum suatu HTML disahkan sebagai suatu dokumen HTML standar, ia harus disetujui dulu oleh W3C untuk dievaluasi secara ketat. Setiap terjadi perkembangan suatu versi HTML, maka mau tak mau browser pun harus memperbaiki diri agar bisa mendukung kode-kode HTML yang baru tersebut. Sebab jika tidak, browser tak akan bisa menampilkan HTML tersebut. [10].

Contoh penulisan:

```
<html>
<head>
<title> Ini Html </title>
</head>
<body>
<center> Hello word </center>
```

2.6.2 *Hypertext Preprocessor* (PHP)

PHP adalah akronim dari *Hypertext Preprocessor*, yaitu suatu bahasa pemrograman berbasiskan kode-kode (*script*) yang digunakan untuk mengolah suatu data dan mengirimkannya kembali ke web browser menjadi kode HTML Adapun karakteristik script PHP dapat diuraikan sebagai berikut [11]:

1. File PHP disimpan dengan ekstensi file seperti :*.php3,*.php4,*.php.
2. Script PHP biasanya diawali dengan tag <?php dan ditutup dengan tag ?>.
3. File PHP dapat menginduk atau disisipkan pada bahasa script lainnya atau dapat berdiri sendiri.

Contoh penulisan:

```
<?php
Echo” Hi, I'm a PHP script! ”;
?>
```

2.6.3 Javascript

JavaScript adalah bahasa *scripting* yang popular di internet dan dapat bekerja di sebagian besar browser popular seperti internet Explorer (IE). Mozilla Firefox, Netscape dan Opera. Kode JavaScript dapat disisipkan

dalam halaman web menggunakan tag SCRIPT. beberapa hal tentang JavaScript [12] :

1. JavaScript didesain untuk menambah interaktif suatu web
2. JavaScript merupakan sebuah bahasa scriptting.
3. Bahasa scripting merupakan bahasa pemrograman yang ringan.
4. JavaScript berisi baris kode yang dijalankan di komputer (web browser).
5. Javascript biasanya disisipkan (embedded) dalam halaman HTML.
6. JavaScript adalah bahasa interpreter (yang berarti skrip dieksekusi tanpa proses kompilasi).
7. Setiap orang dapat menggunakan javascript tanpa membayar lisensi.

Contoh penulisan :

```
<script language="javascript" type="text/javascript">
document.write("Hello World");
</script>
```

2.6.4 CSS (*Cascading Style Sheet*)

Cascading Style Sheet (CSS) adalah salah satu bahasa pemrograman web yang digunakan untuk mempercantik halaman web dan mengendalikan beberapa komponen dalam sebuah web sehingga akan lebih terstruktur dan seragam. CSS dapat kita gunakan dalam mengendalikan ukuran gambar, warna teks, warna tabel, ukuran border, warna border, warna hyperlink, warna mouse over, spasi antar paragraf, spasi antar teks, margin kiri, kanan, atas, bawah, dan parameter lainnya. CSS bisa juga diartikan sebagai bahasa style sheet yang digunakan untuk mengatur tampilan dokumen. Dengan adanya CSS memungkinkan kita untuk menampilkan halaman yang sama dengan format yang berbeda. [10].

Contoh penulisan :

```
Selector { Properties : Value; }
```

```
//* contoh */
body {
        padding-top:5px;
        height: 100%;
        color: #151515;
        background: url(../img/bg.jpg) repeat #efefef;
        padding-bottom:0;
    }
```

2.7 *Library* dan *Tools* Pendukung

2.7.1 *Twitter Bootstrap*

Twitter Bootstrap adalah sebuah alat bantu untuk membuat sebuah tampilan halaman website yang dapat mempercepat pekerjaan seorang pengembang website ataupun pendesain halaman website. Sesuai namanya, website yang dibuat dengan alat bantu ini memiliki tampilan halaman yang sama / mirip dengan tampilan halaman Twitter atau desainer juga dapat mengubah tampilan halaman website sesuai dengan kebutuhan.

Twitter Bootstrap dibangun dengan teknologi HTML dan CSS yang dapat membuat layout halaman website, tabel, tombol, form, navigasi, dan komponen lainnya dalam sebuah website hanya dengan memanggil fungsi CSS (*class*) dalam berkas HTML yang telah didefinisikan. Selain itu juga terdapat komponen-komponen lainnya yang dibangun menggunakan JavaScript [13].

2.7.2 Virtualmin

Virtualmin adalah salah satu layanan *Control panel* yang digunakan untuk mengelola web hosting dengan lisensi gratis. Dengan virtualmin akan memudahkan untuk melakukan berbagai pekerjaan dalam mengelola sebuah

website seperti membuat domain baru, membuat sub-domain, email, statistik dan sebagainya.

Virtualmin sangat cocok digunakan untuk proses pengembangan karena dilengkapi dengan API (*Application Programming Interface*) yang cukup kompleks, dibanding control panel yang berlisensi gratis lainnya [14].

Contoh API *Remote* Virtualmin PHP :

```
<?php
$result = shell_exec("wget -O - --quiet --http-user=root --http-passwd=pass --no-check-certificate 'https://localhost:10000/virtual-server/remote.cgi?program=list-domains'");
echo $result;
```