BAB II

TINJAUAN PUSTAKA

2.1. Tinjauan Perusahaan

PT. Combiphar merupakan Perusahaan Farmasi yang berbentuk PMDN (Penanaman Modal Dalam Negeri) dan masuk dalam 14 besar Perusahaan Farmasi di Indonesia. PT. Combiphar mempunyai sasaran sebagai perusahaan Farmasi yang terkemuka di negara-negara ASEAN, antara lain yaitu selalu meningkatkan kualitas dan mutu produk-produknya dengan cara melakukan inovasi-inovasi terhadap produk yang dihasilkannya.

2.1.1 Sejarah PT.Combiphar

PT. Combiphar didirikan di Jl. Sukabumi 61 Bandung pada tanggal 5 Januari 1971. Pada awalnya PT. Combiphar hanya merupakan home industry yaitu industri yang memproduksi obat-obatan sederhana yang berbentuk sirup (OBH), kapsul, tablet dan salep. Dalam mengembangkan usahanya pada tahun 1977, PT. Combiphar melakukan perluasan pabrik tempat produksi. Pada tanggal 27 Juli 1981, divisi produksi PT. Combiphar pindah ke Jl. Raya Simpang 383 Padalarang yang diresmikan oleh Direktur Jendral Pengawasan Obat Dan Makanan Departemen Kesehatan Republik Indonesia yaitu Dr. Midian Sirait. Sedangkan kantor pusatnya sendiri masih tetap di Jl. Sukabumi 61 Bandung.

Pada tahun 1985, terjadi pergantian kepemilikan perusahaan dimana PT. Combiphar dibeli oleh Gemala Group (PT. Kirana Guna Jaya). Kantor pusat PT. Combiphar pindah ke Jl. Pulolentut Kav. 11/E-4 Jawa Timur pada tahun 1987. Baru pada tanggal 8 April 1988, kantor pusat PT. Combiphar menetap di Jl. Tanah Abang II No. 19 Jakarta Pusat hingga sekarang. PT. Combiphar terus melakukan pengembangan dan meningkatkan kualitas produknya. Usaha yang terus dilakukan oleh PT. Combiphar tidak sia-sia dan membuahkan hasil yaitu diperolehnya sertifikat CPOB dari pemerintah pada tahun 1991 dan hingga saat ini PT. Combiphar telah memiliki 22 Sertifikat CPOB yang dikeluarkan oleh Departemen Kesehatan Republik Indonesia.

Pada tahun 1996, dilakukan renovasi terhadap bangunan dan didirikan gedung produksi sefalosporin yang terpisah dari gedung produksi lain yang diresmikan pada tahun 1997. Selain itu di tahun 1997, dibangun gedung induk produksi lensa mata dari Rohto, yang merupakan hasil join venture PT. Combiphar dengan Rohto Jepang dan kerjasama tersebut berakhir tahun 2000. Saat ini gedung Rohto digunakan oleh PT. Combiphar untuk bagian *Product Development* dan *Quality Assurance*.

Pada tahun 2002, PT. Combiphar mendirikan fasilitas gedung PT. Sanofi Synthelabo Combiphar (SSC) di lingkungan pabrik PT. Combiphar yang merupakan hasil *joint venture* antara PT. Combiphar dengan Sanofi-Synthelabo Perancis. Pada tahun 2002 juga, dibangun gedung khusus untuk produksi liquid (produksi OBH). Pada tahun 2003, PT. Combiphar mengupgrade *Waste Water Treatment Plant*. Pada tahun 2006, PT. Sanofi Synthelabo telah beralih

kepemilikannya menjadi anak perusahaan PT. Combiphar dengan nama Pharma Health Care (PHC). Pada bulan Maret 2006, keseluruhan gedung SSC dialihkan ke PT. Combiphar dan pada bulan September 2006 beralih jadi *Pharma Health Care* (PHC). Setelah diaudit oleh konsultan ISO yaitu AIMS, pada tanggal 9 Oktober 2006, PT. Combiphar diaudit juga oleh SGS yaitu badan yang berwenang memberikan sertifikat ISO. Hasilnya, PT. Combiphar dinyatakan lulus dan berhak mendapatkan sertifikat ISO 9001 : 2000.

2.1.2 Visi dan Misi Perusahaan

PT. Combiphar dalam mengembangkan dan membangun perusahaanya memiliki visi "menjadi tasalah satu perusahaan terkemuka dan disegani oleh Industri Farmasi di Indonesia" dengan menjalankan bisnis dan kegiatan sesuai aturan yang ada tanpa melakukan cara-cara di luar etika. Adapun misi PT. Combiphar adalah "menginginkan suatu kontribusi untuk memperbaiki kualitas hidup sehingga pasien merasa lebih baiohhk dari sebelumnya", melalui program "*Care, Optimize, Motivation, Be Defferent, Integrity, Pride, Harmony, Alert, Responsibility*".

Kebijakan Mutu PT. Combiphar (ISO 9001 : 2000) :

1. Memberikan kepuasan kepada pelanggan dengan menerapkan sistem manajemen mutu ISO 9001 : 2000.
2. Melayani pelanggan dengan cepat, tepat waktu, kenyamanan, ketelitian dan pelayanan yang efektif dan efisien.

3. Selalu berusaha untuk meningkatkan kualitas produk dan pelayanan untuk memberikan kepuasan kepada pelanggan.
4. Menjamin tersedianya sumber daya yang sesuai dan memadai.
5. Melakukan perbaikan terus menerus sistem manajemen mutu.

Sasaran Mutu PT. Combiphar (ISO 9001 : 2000) :

1. Delivery on time 80 %.
2. Keluhan pelanggan (dokter, outlet, pasien, distributor) ;
    a. Terkait distribusi ketersediaan produk harus diselesaikan dalam waktu 1 x 24 jam minimal 90 % (terselesaikan) dari total keluhan.
    b. Terkait mutu atau kualitas produk baik fisik maupun efikasi harus diselesaikan dalam waktu maksimal 3 minggu minimal 90 % dari total keluhan (konfirmasi dahulu ke *medical* dan juga *factory Quality Assurance* apakah mereka sanggup menyelesaikan keluhan pelanggan).
3. Indeks kepuasan pelanggan minimum 80 %.
4. Meningkatkan total nilai (*value*) penjualan minimal 15 % dari tahun sebelumnya (temasuk produk baru).
5. Menciptakan obat baru (produk sendiri) sebanyak minimal 5 (lima) jenis per tahun.

Adapun moto dari PT. Combiphar yaitu dalam mengembangkan produk harus aman, efektif dan memiliki kualitas yang tinggi (*Devoted to save, effective and high quality products*).

2.1.3 Lokasi, Sarana Fisik dan Sarana Penunjang

2.1.3.1 Lokasi

PT. Combiphar menempati 2 lokasi, yaitu di Jl. Tanah Abang II No. 19 Jakarta sebagai kantor pusat yang mengendalikan kegiatan perusahaan secara menyeluruh meliputi keuangan, pemasaran, registrasi, *Human Resources Development* dan lain-lain. Sedangkan pabriknya berada di Jl. Raya Simpang No. 383 Padalarang, Jawa Barat.

Divisi pabrik di Padalarang menempati areal tanah seluas 25.960 m^2, terdiri dari ruang bagian produksi, pengemasan, kantor dan gudang. Laboratorium *Quality Control* dan bagian *Product Development* menempati bangunan tersendiri.

2.1.3.2 Sarana Fisik

Divisi pabrik PT. Combiphar memiliki 5 bangunan pabrik yang terdiri dari bangunan utama dan bangunan di luar bangunan utama. Bangunan utama terdiri dari :

1. Kantor (Luas = 330 m^2) meliputi ruang tamu, ruang administrasi, ruang kepala pabrik, ruang bagian keuangan, ruang bagian logistik, ruang bagian elektronik data processing SAP (*System Application Program*), ruang bagian umum dan ruang pertemuan.
2. Gedung Produksi (Luas = 2.310 m^2) meliputi beberapa ruangan proses sediaan padat, semi padat dan cairan, pengujian *In Process*

Control dari sediaan padat (uji kekerasan, waktu hancur dan friabilitas), ruang pengemasan dan ruang karantina.

3. Gudang (Luas = 1.830 m^2) meliputi gudang bahan baku, gudang bahan pengemas dan gudang obat jadi.
4. Bangunan Sefalosporin (Luas = 100 m^2), terpisah dari ruang produksi umum untuk mencegah terjadinya kontaminasi ke produk lain.
5. Gedung *Pharma Health Care* (PHC) (Luas = 540 m^2), digunakan untuk produksi obat luar dan oral, pengemasan dan gudang.
6. Bangunan OBH yang terdiri dari ruang produksi (Luas =1.320 m^2) dan gudang (Luas = 1.464 m^2) yang terdiri dari gudang bahan baku, bahan pengemas dan obat jadi.
7. Gedung *Quality Assurance* dan *Product Development* yang terdiri dari 3 lantai (Luas = 665 m^2). Lantai bawah adalah laboratorium *form*ulasi dan dokumentasi. Lantai tengah adalah bagian *Quality Assurance* meliputi ruang tamu/lobi, laboratorium *Quality Control* yang terdiri dari ruang pengujian, ruang instrument, ruang mikrobiologi, ruang administrasi dan staff laboratorium *Quality Control* serta laboratorium *Product Development*. Pada lantai atas adalah bagian *Product Development* yang dilengkapi dengan ruang rapat dan perpustakaan.

Selain bangunan utama yang telah disebutkan diatas, terdapat juga fasilitas lain seperti kantin, mushola, gedung bagian teknik dan unit pengolahan limbah (Luas = 1100 m^2).

2.1.3.3 Sarana Penunjang

Untuk mendukung dan memperlancar semua aktivitas di pabrik, digunakan berbagai sarana penunjang, yaitu sebagai berikut : *Mechanical Workshop* (bengkel mekanik), generator dengan daya 157 KVA, *City Electricity* dengan daya 535 KVA, *Steam Boiler* (2 buah), *Central Air Conditioner* (AC), *Air Compressor* (2 buah), *Air Handling Unit* (AHU), Sumur air artesis, *Reverse Osmosis Water System* (2 m^3/ jam) dengan tangki penyimpanan 10.000 L, *Waste Water Treatment Plant* (tempat pengolahan limbah), *Deep Well* dengan kapasitas 100-200 m^3/bulan, *Lighting Protector*, *Telecomunication System* (faximile, telephone, *e-mail*), SAP In*form*ation *System*, *Local Area Network*.

2.2 Pengertian Dasar Sistem

Terdapat dua kelompok pendekatan didalam mendefinisikan sistem, yaitu yang menekankan pada prosedurnya dan yang menekankan pada komponen atau elemennya. Menurut Jerry FitzGerald [1] pendekatan sistem yang lebih menekankan pada prosedur mendefinisikan sistem sebagai berikut :

"*Suatu sistem adalah suatu jaringan kerja dari prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu*".

Pendekatan sistem yang merupakan jaringan jaringan kerja dari prosedur lebih menekankan urut-urutan operasi didalam sistem. Prosedur *(procedure)* didefinisikan oleh Richard F. Neuschel sebagai berikut :

"Suatu prosedur adalah urut-urutan operasi klerikal (tulis menulis) biasanya melibatkan beberapa orang didalam satu atau lebih departemen yang diterapkan untuk menjamin penanganan yang seragam dari transaksi-transaksi bisnis yang terjadi."

Lebih lanjut Jerry FitzGerald, Arda F. FitzGerald dan Waren D. Stallings, Jr., mendefinisikan prosedur sebagai berikut :

"Suatu prosedur adalah urut-urutan yang tepat dari tahapan-tahapan intruksi yang menerangkan apa (what) yang harus dikerjakan, siapa (who) yang mengerjakannya, kapan (when) dikerjakan dan bagaimana (how) mengerjakannya [3]".

Pendekatan sistem yang lebih menekankan pada elemen atau komponennya mendefinisikan sistem sebagai berikut :

"Sistem adalah kumpulan dari elemen-elemen yang berinteraksi untuk mencapai suatu tujuan tertentu".

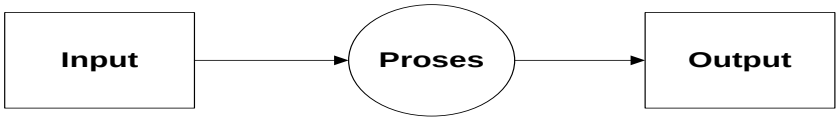
Kedua kelompok definisi ini adalah benar dan tidak bertentangan, yang berbeda adalah cara pendekatannya. Mempelajari suatu sistem akan lebih mengena bila mengetahui terlebih dahulu apakah suatu sistem itu. Lebih lanjut pengertian tentang sistem pertama kali dapat diperoleh dari definisinya. Dengan demikian definisi ini akan mempunyai peranan yang sangat penting didalam pendekatan untuk mempelajari suatu sistem. Pendekatan sistem yang merupakan

kumpulan dari elemen-elemen atau komponen-komponen atau subsistem-subsistem merupakan definisi yang lebih luas. Definisi ini akan lebih banyak diterima, karena kenyataannya suatu sistem dapat terdiri dari beberapa subsistem atau sistem-sistem bagian.

Komponen-komponen atau subsistem-subsistem dalam suatu sistem tidak dapat berdiri lepas sendiri-sendiri. Komponen-komponen atau subsistem-subsistem saling berinteraksi dan saling berhubungan membentuk satu kesatuan sehingga tujuan atau sasaran sistem tersebut dapat tercapai. Pendekatan sistem yang menekankan pada komponen akan lebih mudah didalam mempelajari suatu sistem untuk tujuan analisis dan perancangan suatu sistem. Suatu sistem mempunyai maksud tertentu. Ada yang menyebutnya maksud dari suatu sistem adalah untuk mencapai suatu tujuan *(goal)* dan ada yang menyebutkan untuk mencapai suatu sasaran *(objectives)*.

2.2.1 Bentuk Umum Sistem

Bentuk umum dari sistem terdiri atas masukan *(input)*, proses dan keluaran *(Output)*, dalam bentuk umum sistem ini terdapat satu atau lebih masukan yang akan diproses dan akan menghasilkan suatu keluaran.



Gambar II.1. Bentuk Umum Sistem

2.2.2 Karakteristik Sistem

Suatu sistem mempunyai karakteristik atau sifat-sifat tertentu yaitu :

a. Komponen Sistem *(Component)*

Suatu sistem terdiri dari sejumlah komponen yang saling berinteraksi, yang artinya saling bekerja sama membentuk satu kesatuan. Komponen atau elemen sistem dapat berupa suatu sub sistem atau bagian – bagian dari sistem. Setiap sistem tidak peduli berapapun kecilnya, selalu mengandung komponen atau sub sistem. Setiap sub sistem mempunyai sifat-sifat dari sistem untuk menjalankan suatu fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan. Suatu sistem dapat mempunyai suatu sistem yang lebih besar dapat disebut dengan *supra system*.

b. Batas Sistem *(Boundary)*

Merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai satu kesatuan. Batas sistem menunjukkan ruang lingkup (*scope*) dari sistem tersebut.

c. Lingkungan Luar Sistem (*Environment*)

Environment dari suatu sistem adalah apapun diluar batas dari sistem yang mempengaruhi operasi sistem. Lingkungan luar sistem dapat bersifat menguntungkan dan dapat juga bersifat merugikan sistem tersebut. Lingkungan luar yang menguntungkan merupakan energi dari sistem, dengan demikian harus tetap dijaga dan dipelihara. Sedangkan

lingkungan luar yang merugikan harus ditahan dan dikendalikan, kalau tidak maka akan mengganggu kelangsungan hidup sistem.

d. Penghubung Sistem (*Interface)*

Merupakan media penghubung antara satu sub sistem dengan sub sistem yang lainnya. Melalui penghubung ini memungkinkan sumber daya mengalir dari suatu sub sistem yang lainnya. Keluaran (*output*) dari suatu sub sistem akan menjadi masukkan (*input*) untuk sub sistem yang lainnya dengan melalui penghubung. Dengan penghubung satu sub sistem berintegrasi dengan sistem yang lainnya membentuk satu kesatuan.

e. Masukka n Sistem *(Input)*

Masukkan adalah energi yang dimasukkan ke dalam sistem. Masukkan dapat berupa masukan perawatan *(maintenance input)* dan masukkan sinyal *(signal input*). *Maintenance input* adalah energi yang dimasukkan supaya sistem dapat beroperasi. *Signal input* adalah energi yang diproses untuk didapat keluaran.

f. Pengolah Sistem (*Process*)

Suatu sistem dapat mempunyai bagian pengolah yang akan merubah masukka n menjadi keluaran.

g. Keluaran Sistem *(Output)*

Output adalah hasil dari energi yang diolah dan diklasifikasikan menjadi keluaran yang berguna dan sisa pembuangan. Keluaran dapat

merupakan masukkan untuk sub sistem yang lain atau kepada *supra system*

h. Sasaran Sistem *(Objective)*

Suatu sistem pasti mempunyai tujuan *(goal)* atau sasaran *(objective)*. Kalau suatu sistem tidak mempunyai sasaran, maka operasi sistem tidak akan ada gunanya. Sasaran dari sistem sangat menentukan sekali masukkan yang dibutuhkan sistem dan keluaran yang akan dihasilkan sistem. Suatu sistem dikatakan berhasil bila mengenai sasaran atau tujuannya.

2.3 Teknologi *Website* (WWW)

World Wide Web (WWW) merupakan suatu sistem penyedia in*form*asi dengan skala yang besar yang mengelola in*form*asi tersebut secara terdistribusi dalam internet dengan menggunakan teknologi hypermedia. WWW mempunyai karakteristik sebagai berikut :

a. Dukungan interface yang terintegrasi

WWW menyediakan suatu interface yang terintegrasi terhadap protokol, *form*at data, sistem pengalamatan, dan lain-lain. Hal ini menyebabkan berbagai macam layanan dan basis data yang ada di internet dapat diakses secara langsung.

b. Kemudahan pada sisi pengguna

WWW mendukung secara transparan sebagian besar aplikasi-aplikasi yang terdapat di internet seperti telnet, gopher, anonymous file tranfer protocol, finger, dan aplikasi lainnya.

c. Kemudahan dalam perkembangan

Kapabilitas server WWW dapat dikembangkan secara mudah dengan menggunakan standar PHP antara server WWW dengan alplikasi yang lain. Program PHP juga memungkinkan perubahan in*form*asi secara dinamis yang dapat diperoleh secara real time.

d. Tidak tergantung pada flat*form* tertentu

WWW memungkinkan seseorang dapat membangun server WWW di berbagai sistem yang berlainan dan memberikan in*form*asi dalam bentuk hypermedia. WWW tidak didesain untuk mendukung sistem tertentu.

WWW menggunakan *model client* server. WWW juga menggunakan ekspresi in*form*asi, transfer in*form*asi, metode penamaan in*form*asi yang standar sehingga dapat memproses dan mentransfer in*form*asi secara terdistribusi yang dilakukan secara sistematik.

In*form*asi yang terdistribusi disimpan di dalam server WWW dan *user* mengakses in*form*asi tersebut menggunakan suatu *software* yang disebut dengan *browser*. Server menyimpan in*form*asi di dalam file-file terstruktur menggunakan Hypertext Markup Language (HTML). Hypertext merupakan kumpulan teks-teks yang saling berhubungan satu sama lainnya. Hypermedia serupa dengan

hypertext, hanya saja media yang digunakan bukan hanya teks akan tetapi meliputi juga audio, image, dan video.

WWW menggunakan standar untuk mencapai konsistensi dalam menghasilkan dan melakukan transfer in*form*asi. WWW menggunakan HTML sebagai standar produksi in*form*asi dengan menggunakan Hypertext Transfer Protocol (HTTP) sebagai standar transfer in*form*asi. WWW menggunakan standar penamaan yang disebut dengan Universal Resource Locator (URL) untuk mendukung penyediaan in*form*asi secara universal dalam server yang terdistribusi.

2.3.1 Universal Resource Locator (URL)

URL digunakan untuk menentukan lokasi dari in*form*asi terdistribusi pada suatu server WWW. *User* dapat mengakses in*form*asi yang tersimpan di suatu server dengan menentukan lokasi server sesuai dengan ekspresi URL. Ekspresi URL mempunyai bentuk sebagai berikut :

Protocol ://Alamat Internet [: nomor port]/[directory]/[nama file]

Pada bagian protokol pada URL tersebut adalah protokol transfer data yang dapat berupa http, ftp, gopher, mailto, news, telnet, dan lain-lain.

2.3.2 Hypertext Transfer Protocol (HTTP)

HTTP merupakan sebuah protokol yang didesain untuk men-trasfer in*form*asi dalam bentuk hypermedia antara server dengan sebuah *client*. HTTP juga men-transfer data suatu in*form*asi melalui header-nya. Header HTTP ini merupakan bentuk pengembangan dari Multipurpose Internet Mail Extentions (MIMEs). Pengembangan ini memungkinkan HTTP untuk ment-ransfer in*form*asi dalam bentuk biner dan in*form*asi dalam bentuk yang tidak standar yang berhasil

dinegosiasi antara server dan *client*. Secara umum, delay akan terjadi pada saat melakukan suatu negosiasi sebelum proses transfer data. Karena lama dari delay yang disebabkan oleh overhead akan menjadi relative lebih besar pada transfer data yang sebetulnya singkat.

HTTP merupakan protokol yang bersifat stateless, sehingga server akan memproses setiap *request* dari *user* secara terpisah dari *request* yang lain, independen terhadap *request* yang sebelumnya. HTTP menggunakan 8 bit untuk mentransfer semua tipe data yang mungkin.

Mekanisme yang terjadi pada HTTP bila suatu *client* menginginkan layanan dari server dibagi dalam 4 langkah, yaitu :

a. Connection Setup

   Client mengakses sebuah server dengan menggunakan internet address dan port number. Default dari port number adalah 80.

b. *Request*

   Client mengirimkan message berupa in*form*asi dari metode transaksi dan kapabilitas *client*.

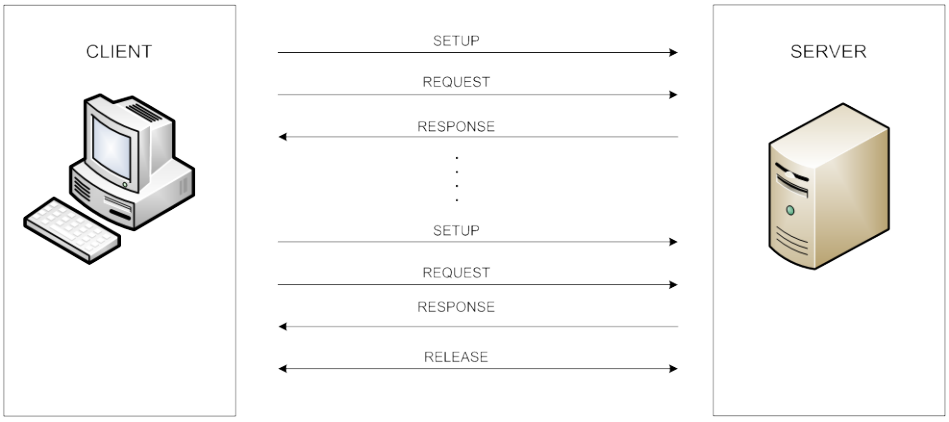
c. Response

   Server mengirimkan response kepada *client* sesudah *client* itu menyelesaikan *request*-nya. Response message meliputi in*form*asi dari transaksi dan data yang diminta.

d. Connection Release

   Client mengakhiri koneksi ke server.

Pada gambar 2.2 diperlihatkan mekanisme proses koneksi yang terjadi pada protokol HTTP antara *client* dan server. Proses koneksi dimulai dari *client* yang melakukan proses connection setup dan *request* ke server. Selanjutnya server akan mengirmkan response ke *client*. *Request* dari *client* ke server dapat terjadi lebih dari satu kali. Dari setiap *request* yang dilakukan oleh *client*, server akan mengirimkan response. Setelah *request*-response selesai, terjadi proses release antara *client* dan server.



Gambar II.2. Proses Koneksi Protokol HTTP antara *client* dan server

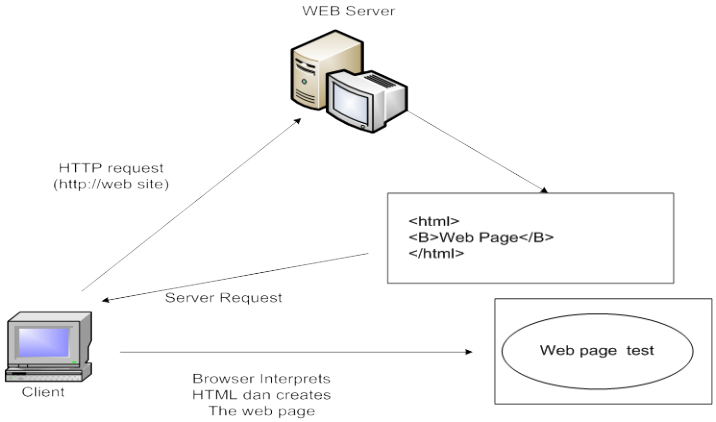
2.3.3 Aplikasi Program Berbasis Web

Banyak situs internet yang memiliki halaman dengan sifat statis seperti profil perusahaan, artikel, dan keterangan-keterangan lain. Situs ini mempunyai dokumen dengan teks yang sederhana, image dan hyperlinks ke dokumen yang dimilikinya. Untuk mengembangkan situs yang bersifat statis, digunakan teknologi *client* side. HTML dan Cascading *Style* Sheet (CSS) dapat digunakan untuk mengatur struktur dan menampilkan halaman isi. Seandainya ingin

diperindah dapat ditambahkan script yang sifatnya *client* side, seperti JavaScript, Jscript ataupun VBScript.

Dengan berkembangnya internet, situs yang ada di internet tidak hanya berfungsi untuk mempresentasikan content tetapi cenderung berupa aplikasi yang kebanyakan terhubung ke suatu basis data. Pada tahapan ini situs akan bersifat dinamis, karena content yang dipresentasikan akan bervariasi dan berubah-ubah sesuai dengan data yang diminta dan action dari *user*. Untuk mengembangkan situs yang dinamis diperlukan teknologi server side seperti PHP, ASP, Perl dan CGI yang lain. Dengan teknologi server side dapat dikembangkan suatu aplikasi berbasis internet yang dapat mengahsilkan dan menampilkan content secara dinamis [4].

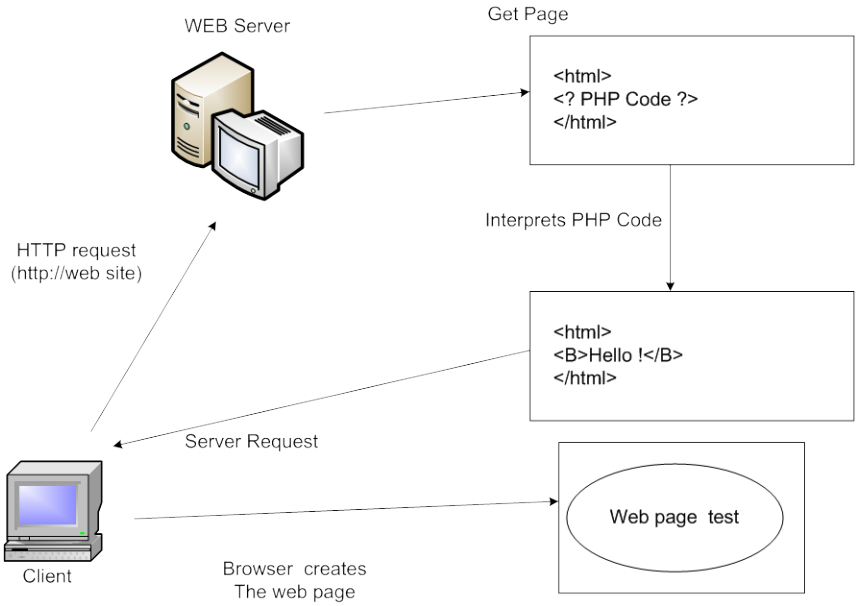
Pada gambar 2.3 diperlihatkan cara kerja situs dengan content yang statis. Pada saat terjadi suatu *request* dari *browser*, server web selanjutnya membaca *request* yang dikirim oleh *browser*, mencari dan menemukan halaman di server, dan mengirim halaman yang diminta melalui internet ke *browser*. Selanjutnya halaman tersebut ditampilkan di *browser*.



Gambar II.3. Web yang statis

Pada gambar 2.4 diperlihatkan cara kerja situs dengan content yang dinamis. Kronologis gambar adalah sebagai berikut :

a. *Browser* (*client*) melakukan *request* halaman web ke WEB Server.
b. WEB Server kemudian membaca *request* yang dikirim dari *browser*, mencari dan menemukan halaman di server.
c. Menterjemahkan perintah yang diberikan oleh bahasa program server-side menjadi halaman HTML.
d. Setelah konversi ke halaman HTML selesai kemudian mengirim halaman yang diminta ke *browser*.
e. Setelah halaman HTML yang dikirim dari WEB server sampai, selanjutnya *browser client* akan menampilkan halaman web.



Gambar II.4. Web yang dinamis

2.4 *Software* Pendukung

2.4.1 Web Server Apache

Web server merupakan server internet yang mampu melayani koneksi transfer data dalam protocol HTTP. Web server merupakan hal yang terpenting dari server di internet dibandingkan server lainnya seperti *e-mail* server, ftp server ataupun news server. Hal ini disebabkan web server telah dirancang untuk dapat melayani beragam jenis data, dari text sampai grafis 3 dimensi. Kemampuan ini telah menyebabkan berbagai institusi seperti universitas maupun perusahaan dapat menerima kehadirannya dan juga sekaligus menggunakannya sebagai sarana di internet.

Web server juga dapat menggabungkan dengan dunia mobile *wireless* internet atau yang sering disebut sebagai WAP (*Wireless Access Protocol*), yang banyak digunakan sebagai sarana handphone yang memiliki fitur WAP. Dalam kondisi ini, webserver tidak lagi melayani data file HTML tetapi telah melayani WML (*Wireless Markup Language*).

Salah satu *software* yang biasa digunakan oleh banyak web master di dunia adalah apache. *Software* tersebut dapat di*download* secara gratis dari web resmi apache, yaitu http://www.apache.org. Dalam Penggunaannya Apache merupakan*software open source* yang sekarang ini sudah merebut pasar dunia lebih dari 50%. Web server ini fleksibel terhadap berbagai *system* operasi seperti windows9x/NT ataupun unix/linux. Apache merupakan turunan dari webserver yang dikeluarkan oleh NCSA yaitu NCSA HTTPd pada sekitar tahun 1995.

Kelebihan web server Apache :

a. Freeware (*software* gratisan)

b. Mudah diinstal.

c. Mampu beroperasi pada berbagai *platform* sistem operasi .

d. Mudah mengkonfigurasinya.

Apache Web server mudah dalam menambahkan periferal lainnya ke dalam *platform* web servernya, misalnya : untuk menambahkan modul, cukup hanya menset file konfigurasinya agar mengikutsertakan modul itu ke dalam kumpulan modul lain yang sudah dioperasikan.

2.4.2 PHP

PHP/FI *(Personal Home Page/Form Interface*) merupakan nama awal dari PHP. Dibuat pertama kali oleh Rasmus Lerdoff. PHP, awalnya merupakan program CGI (*Common Gateway Interface*) yang dikhusukan untuk menerima input melalui *form* yang ditampilkan dalam *browser* web. *Software* ini disebarkan dan dilisensikan sebagai *software open source* (gratis). PHP secara resmi merupakan kepanjangan dari PHP: *Hypertext Prepocessor*, merupakan bahasa script serverside yang disisipkan pada halaman HTML.

2.4.2.2 Sejarah PHP

PHP pertama kali dibuat pada tahun 1994 oleh Rasmus Lerdoff, awalnya digunakan pada *website*-nya untuk mencatat siapa saja yang berkunjung dan melihat biodatanya. Versi pertama yang di release tersedia pada awal tahun 1995, dikenal sebagai *tool Personal Home Page*, yang terdiri atas *engine parser* yang sangat sederhana yang hanya mengerti beberapa makro khusus dan sejumlah utilitas yang sering digunakan pada halaman-halaman web, seperti buku tamu, hit

counter, dan lainnya. Parser diprogram ulang pada pertengahan tahun 1995 dan diberi nama PHP/FI versi 2.0. FI berasal dari paket Rasmus lainnya yang ditulis untuk menginterpretasikan data dari *form*, yang kemudian dikombinasikan dengan *tool Personal Home Page* dan ditambahkan dukungan untuk *database* mSQL (mini SQL).

Tahun 1995 ini dianggap sebagai tahun kelahiran dari PHP/FI yang kemudian membuat pertumbuhan aplikasi web yang sangat pesat, dan banyak orang kemudian berkontribusi mengembangkan PHP/FI. Diperkirakan pada akhir tahun 1996 telah digunakan oleh sedikitnya 15000 *website* di seluruh dunia. Dan pertengahan tahun 1997 mencapai 50.000 situs.

Pada pertengahan tahun 1997 ini terjadi perubahan pengembangan PHP. Pengembangan dilakukan oleh tim yang terorganisasi bukan oleh Rasmus sendiri saja lagi. *Parser* dikembangkan oleh Zeev Suraski dan Andi Gutmans yang kemudian menjadi dasar untuk versi 3, dan banyak utilitas tambahan yang diprogram umtuk menambah kemampuan dari versi 2. Setelah versi 3 lalu kemudian dikembangkan lagi menjadi versi 4. PHP versi 4 menggunakan *engine script* Zend untuk lebih meningkatkan kinerja (*performance*) dan mempunyai dukungan yang banyak berupa ekstensi dan fungsi.

PHP versi 4.2.0 direlease pada tanggal 22 April tahun 2002. Perbaikan pada *bugs* (kesalahan/*error*) terutama pada upload file melalui *browser* telah dibetulkan, dan banyak penambahan fungsi yang lebih memudahkan lagi bagi developer/pengembang aplikasi web untuk membuat program yang lebih baik. Jumlah fungsi yang ada sampai dengan versi ini mencapai ribuan fungsi dan

dikelompokan menjadi 109 fungsi, banyak bertambah dari versi sebelumnya. Sampai dengan versi 4.3.7 tercatat ada 125 kelompok fungsi yang dimiliki oleh PHP. Saat ini pengembangan PHP telah mulai memasuki versi 5.

2.4.2.3 Kemampuan PHP

PHP secara mendasar dapat mengerjakan semua yang dapat dikerjakan oleh program CGI, seperti mendapakan data dari *form*, menghasilkan isi halaman *web* yang dinamik, dan menerima *cookies*. Kemampuan (*feature*) PHP yang paling dihandalkan dan signifikan adalah dukungan kepada banyak *database*. Membuat halaman *web* yang menggunakan data dari *database* dengan sangat mudah dapat dilakukan. Tabel 2.1 berikut adalah daftar *database* yang didukung oleh PHP.

Tabel II.1. Daftar *Database* yang Didukung PHP

AdabasD	In*form*ix	MySQL	SQlite
Dbase	Ingres	ODBC	Sybase
Empress	IBM DB2	Oracle	Velocis
FilePro	Interbase	Ovrimos	Unix DBM
FrontBase	MySQL	PosgreSQL	
Hyperwave	Direct MS SQL	Solid	

PHP juga mendukung untuk berkomunikasi dengan layanan lain menggunakan protokol IMAP, SNMP, POP3, HTTP, dan lainnya yang tidak

terhitung. Untuk OS (*Operating Sistem*), PHP dapat berjalan di *platform* OS Windows dan *platform* OS Linux. Karena PHP merupakan bahasa *script serverside*, maka dibutuhkan sebuah *server web* yang berfungsi untuk memproses *script* PHP sebelum ditampilkan ke *browser*. Beberapa *server web* untuk PHP diantaranya Apache untuk Windows dan Linux, Oreily *Website* Pro untuk Windows, dan Xitami untuk Windows.

2.4.2.4 Kelebihan PHP

Kelebihan PHP dibandingkan dengan pemrograman lain :

a. Bahasa pemrograman PHP adalah sebuah bahasa script yang tidak melakukan sebuah kompilasi dalam penggunaanya.
b. *Web server* yang mendukung PHP dapat ditemukan dimana - mana dari mulai IIS sampai dengan apache, dengan configurasi yang relatif mudah.
c. Dalam sisi pengembangan lebih mudah, karena banyaknya milis - milis dan developer yang siap membantu dalam pengembangan.
d. Dalam sisi pemahamanan, PHP adalah bahasa *scripting* yang paling mudah karena referensi yang banyak.

   PHP adalah bahasa *open source* yang dapat digunakan di berbagai mesin (linux, unix, windows) dan dapat dijalankan secara *runtime* melalui *console* serta juga dapat menjalankan perintah-perintah sistem.

2.4.3 CodeIgniter

Dalam dunia pemrograman *website* ada banyak sekali bahasa pemrograman yang bisa digunakan. Salah satu bahasa pemrograman yang sangat

terkenal dan banyak sekali digunakan oleh para pembuat *website* adalah PHP. PHP merupakan bahasa pemograman yang mudah dipelajari dan andal untuk digunakan. Banyaknya komunitas PHP – dikenal dengan nama *PHP User Group* (PHPUG) – membuat banyak *class* maupun fungsi yang di*share* ke publik, sehingga membuat pembangunan sebuah *website* semakin mudah.

Dalam perkembangannya, semakin banyak bermunculan CMS (*Content Management System*) berbasis PHP. Dengan adanya CMS, seorang pengguna awam pun bisa membuat sebuah *website* tanpa harus menguasai PHP terlebih dahulu. Walaupun demikian, CMS tentu saja tidak bisa mengakomodir setiap kebutuhan pengguna, dikarenakan selalu ada kebutuhan spesifik dari pengguna yang tidak bisa dilakukan oleh CMS secara *default*.

Selain CMS, saat ini juga bermunculan berbagai macam *framework*. *Framework* merupakan alat yang digunakan untuk membantu pekerjaan. Karena untuk pembuatan *website* maka *framework* disini dapat diartikan sebagai alat yang dapat digunakan untuk mempermudah pembuatan *website*. Sebuah *framework* sudah menyediakan berbagai macam fungsi/*class* yang dibutuhkan dalam pembuatan *website*. *Framework* juga menyediakan lingkungan pengembangan yang harus diikuti dalam pembuatan *website* menggunakan *framework* tersebut.

Jika dengan CMS maka *user* tinggal menjalankan saja tidak perlu lagi memikirkan untuk menulis kode program sendiri, tetapi tidak demikian dengan *framework*. Menggunakan *framework user* masih harus menulis kode, bedanya kode-kode yang ditulis harus menyesuaikan dengan lingkungan *framework* yang digunakan. Memang konsekuensinya adalah *user* harus belajar lagi lingkungan

pengembangan berdasarkan *framework* yang digunakan, tetapi hal itu akan terbayar setelah bisa menguasai dan menggunakan *framework* tersebut.

Sebuah *framework* selain menyediakan lingkungan pengembangan sendiri-sendiri juga menyediakan berbagai macam fungsi siap pakai yang bisa digunakan dalam pembuatan *website*. Sehingga tidak perlu kaget jika akan banyak kode atau fungsi yang terlihat tidak seperti biasanya, karena fungsi-fungsi tersebut merupakan fungsi bawaan *framework* dan bukan fungsi asli dari PHP. Fungsi tersebut terkadang merupakan pengembangan atau penyesuaian fungsi asli PHP agar lebih mudah digunakan atau agar lebih sesuai dengan kebutuhan pengguna.

CodeIgniter adalah suatu *Application Development Framework* untuk pembangunan suatu *website* berbasis PHP. Tujuannya adalah agar pembangunan projek bisa lebih cepat dibandingkan dengan membangunnya dari awal, dengan menyediakan *library* yang kaya yang umum digunakan untuk pembangunan suatu *website*, dengan interface dan struktur logika yang sederhana untuk mengaksesnya. CodeIgniter akan membuat *developer* semakin kreatif.

Tujuan dari pembuatan *framework* CodeIgniter adalah untuk menghasilkan *framework* yang akan dapat digunakan untuk pengembangan proyek pembuatan *website* secara lebih cepat dibandingkan dengan pembuatan *website* dengan cara koding secara manual, dengan menyediakan banyak sekali pustaka yang dibutuhkan dalam pembuatan *website*, dengan antarmuka yang sederhana dan struktur logika untuk mengakses pustaka yang dibutuhkan. CodeIgniter membiarkan *user* untuk memfokuskan diri pada pembuatan *website* dengan meminimalkan pembuatan kode untuk berbagai tujuan pembuatan *website*.

Kelebihan CodeIgniter diantaranya adalah :

a. Gratis

   CodeIgniter dilisensikan dibawah lisensi Apache/BSD *style open source license*, ini berarti *user* dapat menggunakannya sesuai dengan keinginan.

b. Berjalan di PHP versi 4 dan 5

   Sekarang ini PHP sudah mencapai versi ke 5, meskipun begitu masih banyak orang yang tetap menggunakan PHP versi 4, oleh sebab itu CodeIgniter dikembangkan agar tetap kompatibel dengan PHP versi 4 dan dapat dijalankan pada PHP versi 5.

c. Ringan dan cepat

   Secara default CodeIgniter hanya berjalan dengan me-load beberapa pustaka saja, dengan demikian hanya membutuhkan resource yang sedikit sehingga ringan dan cepat dijalankan. Pustaka-pustaka lain yang nantinya akan digunakan bisa di load sesuai dengan kebutuhan.

d. Menggunakan MVC

   CodeIgniter menggunakan lingkungan pengembangan dengan metode *Model View Controller* (MVC) yang membedakan antara logika dan presentasi/tampilan, sehingga tugas bisa lebih mudah dipecah-pecah. Ada bagian yang khusus membuat tampilan dan bagian yang membuat core programnya.

e. Dokumentasi

Salah satu hal yang bisa dijadikan barometer apakah sebuah aplikasi benar-benar dikembangkan atau tidak bisa dilihat dari dokumentasinya. Dalam hal ini CodeIgniter sangat luar biasa, terdapat dokumentasi yang sangat lengkap tentang semua hal yang ada dalam CodeIgniter. Mulai dari langkah instalasi sampai dokumentasi fungsi-fungsi nya tersedia. Adanya dokumentasi sangat memudahkan bagi pemula dalam mempelajari lingkungan pengembangan *website* dengan CodeIgniter.

f. Pustaka yang lengkap

CodeIgniter dilengkapi dengan berbagai pustaka siap pakai untuk berbagai kebutuhan, misalnya saja koneksi *database*, email, session dan cookies, keamanan, manipulasi gambar dan banyak lagi.

Berikut fitur-fitur yang didukung oleh CodeIgniter :

a. Sistem berbasis *Model View Controller*
b. Kompatibel dengan PHP versi 4.
c. Ringan dan Cepat.
d. Terdapat dukungan untuk berbagai basis data.
e. Mendukung Active Record *Database*, *form* dan validasi data masukan, pencatatan *error* yang terjadi, lokalisasi bahasa.
f. Tersedia pengaturan session, pustaka untuk bekerja dengan XMP-RPC.
g. Tersedia *class* untuk upload file, transfer via FTP, pagination (membuat tampilan perhalaman), calendar, mengetahui *user* agent,

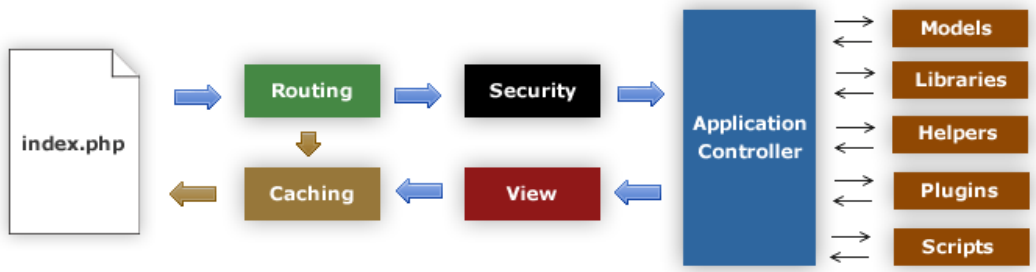
manipulasi gambar (*cropping*, *resizing*, *rotate* dan lain-lain), mengirim *email*, *template website*, *trackback*

h. Mendukung enkripsi data, benchmarking, caching, hooks, ekstensi *class* dan plugin, XSS filtering.

i. Menghasilkan *clean* URL, URI routing yang felksibel.

j. Memiliki *helper* yang sangat banyak jumlahnya

Berikut ini merupakan diagram dari aplikasi yang menggunakan CodeIgniter sebagai *framework*nya.



Gambar II.5. Diagram Aplikasi Menggunakan CodeIgniter

a. index.php berfungsi sebagai *controller* depan, mnginisialisasi basic resource yang dibutuhkah untuk menjalankan CI.

b. *Route* menganalisa HTTP *request* untuk menentukan apa yang harus dilakukan dengan HTTP *request* itu.

c. Jika file cache masih ada, maka akan dikirim langsung ke *browser*, tanpa melewati eksekusi normal sistem.

d. Keamanan, sebelum *controller* aplikasi di panggil, HTTP *request* dan data yang dikirim *user*, di *filter* untuk alasan keamanan.

e. *Controller* memanggil *model*, *library* inti, *plugin*, *helper*, dan *resource* lainnya yang di butuhkan untuk memroses *request* tertentu.

f. *View* yang sudah diproses, dikirim ke *browser* sebagai hasil yang terlihat. Jika status caching ON, *view* akan disimpan di cache, jadi jika ada *request* yang sama, *view* itu bisa ditampilkan lagi.

CodeIgniter menggunakan pola *Model-View-Controller* (MVC). MVC menggunakan pendekatan dimana terjadi pemisahan antara logika aplikasi dan tampilan.

a. *Model*, merepresentasikan struktur data. Umumnya *class model* mengandung fungsi untuk menampilkan, menambah, mengubah in*form*asi pada *database*

b. *View*, merupakan in*form*asi yang dipresentasikan kepada *user*. *View* biasanya merupakan sebuah halaman web, tapi pada CodeIgniter ini juga bisa merupakan fragment, seperti header atau footer. Atau bisa juga merupakan sebuah rss page ataupun jenis halaman lainnya.

c. *Controller* berfungsi sebagai penghubung antara *Model* dan *View* dan resource lainnya yang dibutuhkan untuk memproses HTTP *request* dan menghasillkan sebuah halaman web.

Walaupun begitu, konsep MVC pada CodeIgniter tidak begitu ketat, karena *Model* bisa tidak digunakan. Apabila pada saat pembuatan *website* tidak diperlukan pemisahan, atau ketika *user* memandang bahwa penggunaan *model* hanya akan menambah kompleksitas, maka cukup digunakan hanya *Controller* dan *View*. CodeIgniter juga mengijinkan programmer untuk menggabungkan

script yang telah dibuat sebelumnya walaupun tidak menggunakan kaidah/ aturan yang digunakan oleh CodeIgniter, ataupun membuat *library* baru, yang memungkinkan programmer untuk bekerja menggunakan caranya sendiri.

2.4.4 Pengenalan JavaScript

Javascript diperkenalkan pertama kali oleh Netscape pada tahun 1995. Pada awalnya bahasa ini dinamakan *"LiveScript"* yang berfungsi sebagai bahasa sederhana untuk *browser* Netscape Navigator 2. Pada masa itu bahasa ini banyak di kritik karena kurang aman, pengembangannya yang terkesan buru-buru dan tidak ada pesan kesalahan yang di tampilkan setiap kali *user* membuat kesalahan pada saat menyusun suatu program. Kemudian sejalan dengan sedang giatnya kerjasama antara Netscape dan Sun (pengembang bahasa pemrograman "Java") pada masa itu, maka Netscape memberikan nama "JavaScript" kepada bahasa tersebut pada tanggal 4 desember 1995. Pada saat yang bersamaan Microsoft sendiri mencoba untuk mengadaptasikan teknologi ini yang mereka sebut sebagai "Jscript" di *browser* Internet Explorer 3.

Javascript adalah bahasa yang berbentuk kumpulan skrip yang pada fungsinya berjalan pada suatu dokumen HTML, sepanjang sejarah internet bahasa ini adalah bahasa skrip pertama untuk web. Bahasa ini adalah bahasa pemrograman untuk memberikan kemampuan tambahan terhadap bahasa HTML dengan mengijinkan pengeksekusian perintah perintah di sisi *user*, yang artinya di sisi *browser* bukan di sisi server web. Javascript bergantung kepada *browser* (navigator) yang memanggil halaman web yang berisi skrip-skrip dari Javascript

dan tentu saja terselip di dalam dokumen HTML. Javascript juga tidak memerlukan kompilator atau penterjemah khusus untuk menjalankannya (pada kenyataannya kompilator Javascript sendiri sudah termasuk di dalam *browser* tersebut). Lain halnya dengan bahasa "Java" yang memerlukan kompilator khusus untuk menterjemahkannya di sisi *user* atau *client*.

Javascript merupakan suatu bahasa yang perkembangannya lambat di bandingkan dengan Java yang berkembang sangat cepat. Di Javascript *user* tidak mungkin menyembunyikan *script* yang *user* tulis, kode langsung di tulis di dalam dokumen HTML dan sangat mudah terlihat, sedangkan di Java, kode sudah berbentuk setengah terkompilasi (dalam bentuk applet) dan tidak mungkin terlihat dari dalam dokumen HTML, satu mesin *virtual* di sisi *user* yang bertanggung jawab untuk menterjemahkan program di dalam *applet* tersebut setiap kali halaman HTML yang memuat *applet* tersebut dipanggil oleh *browser*. Dibandingkan dengan *applet* java yang cukup lambat dibuka oleh *browser*, bisa dikatakan bahwa Javascript cukup cepat di panggil(di-*load*) oleh *navigator*.

JavaScript sendiri merupakan bahasa yang mudah dipahami, dalam artian diperlukan *skill novice* atau dasar untuk mengerti bahasa ini, jika anda sudah terbiasa dan mengenal konsep bahasa pemrograman visual, maupun Java ataupun C, akan sangat mudah untuk memahami konsep Javascript.

JavaScript adalah bahasa yang "*case sensitive*" artinya membedakan penamaan variabel dan fungsi yang menggunakan huruf besar dan huruf kecil, contoh variabel atau fungsi dengan nama TEST berbeda dengan variabel dengan

nama test. Dan yang terakhir seperti bahasa Java ataupun C, setiap instruksi diakhiri dengan karakter titik koma (;).

2.4.5 jQuery

JQuery merupakan suatu *framework* (*library*) Javascript yang menekankan bagaimana interaksi antara Javascript dan HTML. JQuery pertama kali dirilis pada tahun 2006 oleh John Resig. Pada perkembangannya JQuery tidak sekedar sebagai *framework* Javascript, namun memiliki kehandalan dan kelebihan yang cukup banyak. Hal tersebut menyebabkan banyak developer web menggunakannya. JQuery memiliki slogan "Write less, do more" yang kurang lebih maksudnya adalah kesederhanaan dalam penulisan code, tapi dengan hasil yang lebih banyak.

Library jQuery menyediakan berbagai macam keperluan untuk web *scripting*. Kemampuannya untuk dapat dikembangkan lebih lanjut, membuat banyaknya web developer untuk mengembangkan *plugin* tambahan. Fitur utama dari jQuery diantaranya :

a. Dapat mengakses elemen dalam dokumen

   Tanpa adanya *library* Javascript khusus, untuk mengakses suatu bagian tertentu dari halaman, harus mengikuti aturan Document *Object Model* (DOM) dan pengaksesan harus secara spesifik menyesuaikan dengan struktur HTML. Dengan kata lain, pengaksesan bagian tertentu dari halaman sangat tergantung pada struktur dari HTML. JQuery menawarkan cara yang mudah (bahkan sangat mudah) dalam

mengakses bagian tertentu dari halaman. Pengaksesan juga tidak terlalu bergantung pada struktur HTML.

b. Mengubah tampilan halaman *website*

CSS (Cascading *Style* Sheet) menawarkan metode yang cukup handal dalam mengatur dan mempercantik halaman web. Namun terkadang CSS punya kelemahan yang cukup mengganggu, yaitu beberapa perintah CSS tidak didukung oleh semua *browser*. Cukup merepotkan jika *user* harus mendesign halaman web dengan beberapa CSS sekaligus. Sekali lagi JQuery menawarkan solusi untuk mengatasi hal tersebut. Dengan JQuery, "kesenjangan" yang terjadi antara *browser* dalam urusan CSS akan tertutup dengan baik.

c. Mengubah isi dari dokumen

Tidak hanya memberikan "kosmetik" pada halaman web, jquery juga memberikan fasilitas untuk mengubah isi dari dokumen hanya dengan beberapa baris perintah. Mengubah disini dapat berarti mengganti teks, menambahkan teks atau gambar, mengurutkan suatu daftar (list), menghapus baris tabel dan sebagainya. Dengan JQuery, hal tersebut dapat dilakukan dengan hanya beberapa baris perintah.

d. Merespon interaksi *user*

Website yang baik tidak cukup digambarkan dengan *user*-interface dan tampilan yang memukau. Namun lebih dari itu, bagaimana pengunjung dapat berinteraksi dengan *website* dan dapat mengatur tampilannya sendiri. Interaktivitas sangat bergantung bagaimana pemrograman yang

dipakai dalam menangani event-handling. Javascript sendiri memiliki beberapa event-handling seperti onclick untuk menangani event saat terjadi click. Namun demikian, *event handling* pada Javascript terbatas pada *object-object* tertentu, dan jenisnya pun terbatas. JQuery melengkapi semuanya dengan tambahan penanganan event-handling yang semakin mudah.

e. Animasi pada dokumen

Animasi seringkali disertakan dalam suatu halaman web untuk menambah kecantikannya. Saat ini animasi masih cukup digemari oleh para peselancar situs. Animasi dapat dibuat dalam berbagai gaya, ada yang menggunakan Flash, gambar bergerak (GIF), video, dan sebagainya. Masing-masing tentu memiliki kelebihan dan kekurangan masing-masing. JQuery sendiri menawarkan konsep animasi (walaupun masih sederhana) yang cukup apik namun ramah *bandwidth* alias ringan. Salah satu animasi yang bisa dibuat dengan JQuery adalah *fading* jika terdapat suatu bagian dari halaman ditambahkan atau dihilangkan.

f. Mengambil in*form*asi dari server tanpa harus me-*refresh* halaman

Mengambil in*form*asi dari server tanpa *refresh* halaman merupakan salah satu konsep dasar yang dikenal dengan nama AJAX (Asynchronous Javascript and XML). jQuery akan memberikan kemudahan dalam mengembangkan *website* menggunakan AJAX,

karena *user* tidak dipusingkan lagi dengan perbedaan karakteristik pada masing-masing *browser*.

g. Menyederhanakan penulisan sintaks javascript

Semboyan JQuery adalah "*Write less, do more*" atau dengan kata lain kesederhanaan dalam penulisan code, tetapi menghasilnya tampilan yang lebih. Sebenarnya inilah yang menjadi daya tarik tersendiri buat para pengembang web untuk menggunakan JQuery.

2.4.6 MySQL

MySQL adalah sebuah perangkat lunak suatu sistem manajemen *database* yang *multithread* dan multi-*user*. MySQL dimiliki dan disponsori oleh sebuah perusahaan komersial Swedia MySQL AB, dimana memegang hak cipta hampir atas semua kode sumbernya. Kedua orang Swedia dan satu orang Finlandia yang mendirikan MySQL AB adalah David Axmark, Allan Larsson, dan Michael "Monty" Widenius.

MySQL merupakan susunan salah satu konsep utama dalam *database* sejak lama, yaitu SQL (*Structured Query Language*). Kendala dari suatu DBMS dapat diketahui dari cara kerja optimalisasinya dalam melakukan perintah-perintah SQL yang dibuat *user* ataupun program-program aplikasinya. Sebagai *database* server, MySQL dapat dikatakan lebih unggul dibanding *database* server lainnya dalam *query* data.

Suatu *database* relasional menyimpan data dalam tabel-tabel terpisah. Hal ini memungkinkan kecepatan dan fleksibilitas. Tabel-tabel yang dihubungkan

dengan relasi yang ditentukan membuatnya bisa mengkombinasikan data dari beberapa tabel pada suatu permintaan. Untuk menambahkan, mengakses, dan memproses data yang tersimpan pada suatu *database* komputer maka diperlukan sistem manajemen *database* seperti MySQL. Karena komputer sangat unggul dalam menangani sejumlah besar data, sistem manajemen *database* memainkan suatu peranan yang penting dalam komputasi, baik sebagai utility stand-alone maupun bagian dari aplikasi lainnya.

Beberapa perintah dasar SQL yang sering dipergunakan pada MySQL adalah sebagai berikut :

a. Create *Database*

   Yaitu perintah yang digunakan untuk membuat *database* baru.

   Sintaks : Create *database database*_nama *database*

b. Drop *Database*

   Yaitu perintah yang digunakan untuk menghapus *database*.

   Sintaks : Drop Tabel Tabel_name

c. Create Tabel

   Yaitu perintah yang digunakan untuk membuat tabel baru.

   Sintaks Create Tabel tabel_name (create_definition)

d. Describe

   Yaitu perintah yang digunakan untuk mendeskripsikan tabel

   Sintaks Describe (Desc) tabel [colum]

e. Alter Tabel

   Yaitu perintah yang digunakan untuk memodifikasi tabel

Sintaks Alter [Ignor] Tabel table_name

f. Drop Tabel

Yaitu perintah yang digunakan untuk menghapus tabel

Sintaks Drop Tabel tabel_name [tabel_name..]

g. Delete

Yaitu perintah yang digunakan untuk menghapus record dri tabel

Sintaks Delete From tabel_name Where Where_definiition

h. Select

Yaitu perintah yang digunakan untuk query ke *database*

Sintaks select * from tabel_name

Select Field from tabel_name

2.4.7 Konsep Dasar *Object Oriented Programming (OOP)*

2.4.7.1 Sejarah *Oriented Programming (OOP)*

Object-Oriented Programming (OOP) adalah sebuah pendekatan untuk pengembangan suatu *software* dimana dalam struktur *software* tersebut didasarkan kepada interaksi *object* dalam penyelesaian suatu proses/tugas.

Interaksi tersebut mengambil *form* dari pesan-pesan dan mengirimkannya kembali antar *object* tersebut. *Object* akan merespon pesan tersebut menjadi sebuah tindakan /*action* atau metode. Jika kita mencoba melihat bagaimana tugas disekitar kita diselesaikan, kita akan mengetahui bahwa kita berinteraksi dalam sebuah *object-oriented world*.

Object-oriented programs terdiri dari *object*s yang berinteraksi satu sama lainnya untuk menyelesaikan sebuah tugas. Seperti dunia nyata, *user*s dari *software* programs dilibatkan dari logika proses untuk menyelesaikan tugas. Contoh, ketika kamu mencetak sebuah halaman diword processor, kamu berarti melakukan inisialisasi tindakan dengan mengklik tombol printer. Kemudian kamu hanya menunggu respon apakah *job* tersebut sukses atau gagal, sedangkan proses terjadi internal tanpa kita ketahui. Tentunya setelah kamu menekan tombol printer, maka secara simultan *object* tombol tersebut berinteraksi dengan *object* printer untuk menyelesaikan job tersebut.

Konsep OOP dimulai pertengahan 1960-an dengan sebuah bahasa program **SIMULA** kemudian dilanjutkan di era 70-an dengan **SMALLTALK**. Meskipun developer *software* tidak secara intensif mengembangkan OOP, tetapi metodologi ***object*-oriented** tetap digunakan. Pada pertengahan 80-an, bahasa OOP seperti C++ dan Eifle menjadi popular diantara programmer computer. Popularitas OOP berlanjut pada tahun 90-an, banyak pengembang *software* menggunakan konsep OOP seperti Java dll. Di tahun 2002, versi terakhir dari Visual Studio, Microsoft memperkenalkan bahasa OOP baru yaitu C# (dibaca C-sharp) serta upgrade Visual Basic

2.4.7.2 Konsep Oriented Programming (OOP)

Konsep dari *object* oriented programming (OOP) adalah lebih dari sekedar sebuah konsep pemrograman, *Object* oriented programming adalah cara berpikir tentang aplikasi yang mempelajari untuk berpikir bahwa aplikasi bukan sekedar prosedur melainkan sebagai *object* dan real entity. *Object* yang dimaksud disini memiliki pengertian suatu modul yang mengkombinasikan antara data dan kode program yang bekerja sama dalam program dengan melewatkan proses satu sama lain. Jadi *object* oriented programming merupakan cara yang paling efisien untuk menulis program komputer yang sangat mudah untuk di kombinasikan dan untuk dipergunakan kembali.

Object oriented programming lebih memfokuskan kepada manipulasi *object*. Kenapa seorang *programmer* harus mempelajari *object* oriented *programming* bahkan seorang *programmer* yang tidak pernah bekerja dengan *object oriented programing* pun harus mempelajarinya juga. Hal ini dikarenakan pada suatu hari nanti semua bahasa pemrograman akan menambahkan kemampuan *object oriented programming* pada bahasanya.

Beberapa konsep dasar dan *term-term* yang umum untuk seluruh bahasa *Object Oriented Programing* :

1. Class

Class adalah kumpulan atas definisi data dan fungsi-fungsi dalam suatu unit untuk suatu tujuan tertentu. Sebuah *class* adalah dasar dari modularitas dan struktur dalam pemrograman berorientasi *object*. Sebuah *class* secara tipikal sebaiknya dapat dikenali oleh seorang *non-programmer* sekalipun terkait dengan

domain permasalahan yang ada, dan kode yang terdapat dalam sebuah *class* sebaiknya (relatif) bersifat mandiri dan independen (sebagaimana kode tersebut digunakan jika tidak menggunakan OOP). Dengan modularitas, struktur dari sebuah program akan terkait dengan aspek-aspek dalam masalah yang akan diselesaikan melalui program tersebut. Cara seperti ini akan menyederhanakan pemetaan dari masalah ke sebuah program ataupun sebaliknya.

2. *Object*s

Object adalah sebuah structure yang menggabungkan data dan prosedur untuk bekerja bersama-sama. Objek merupakan dasar dari modularitas dan struktur dalam sebuah program komputer berorientasi objek. Dalam term OOP, ***object*** adalah sebuah *structure* yang menggabungkan data dan prosedur untuk bekerja bersama-sama. Contoh, jika kamu tertarik dalam pelacakan data yang dihubungkan dengan produk, kamu akan menciptakan sebuah *object* produk yang bertanggung jawab untuk me-*maintenance* dan bekerja dengan data yang bersinggungan dengan produk. Jika kamu ingin kemampuan mencetak dalam aplikasi kamu, kamu harus bekerja dengan sebuah *object* printer yang bertanggung jawab untuk data serta metode yang digunakan untuk berinteraksi dengan printermu.

3. Abstraction

Ketika membangun *object*s dalam aplikasi OOP, adalah penting untuk menggabungkan konsep *abstraction* ini. Jika kamu membangun aplikasi *shipping*, kamu harus membangun ***object*** produk dengan atribut seperti ukuran dan berat. Warna adalah contoh in*form*asi yang tidak ada hubungannya dan harus dibuang.

Tetapi ketika kamu membangun *orderentry application*, warna menjadi penting dan harus termasuk atribut ***object*** produk.

4. *Encapsulation*

Ciri penting lainnya dari OOP adalah *encapsulation*. Encapsulation adalah sebuah proses dimana tidak ada akses langsung ke data yang diberikan, bahkan *hidden*. Jika kamu ingin mendapat data, kamu harus berinteraksi dengan *object* yang bertanggung jawab atas dara tersebut. Dalam contoh inventory, jika kita ingin melihat atau mengupdate in*form*asi atas produk, kita seharusnya bekerja melalui *object* produk. Untuk membaca data, kita mengirimkan pesan ke ***object*** produk, kemudian ***object*** produk akan membaca pesan dan mengirim pesan balik ke kamu.

Object produk mendefinisikan bahwa operasi dapat dilakukan pada data produk. Jika kamu mengirim pesan untuk memodifikasi data dan jika *object* produk menentukan bahwa permintaan itu valid maka permintaan tersebut akan dilakukan dan akan mengirimkan pesan balik bersama hasilnya. Dengan melakukan encapsulasi pada data, berarti kamu membuat data dari sistemmu lebih secure dan terpercaya. Karena kamu mengetahui bagaimana data diakses dan operasi apa yang sedang berjalan pada data. Hal ini membuat maintenance program menjadi lebih mudah. Kamu juga dapat memodifikasi metode yang digunakan untuk kerja pada data, dan jika kamu tidak merubah bagaimana metode itu diminta dan tipe response dikirim balik, maka kamu tidak dapat merubah *object* lain dengan menggunakan metode tersebut.

5. *Polymorphism*

*Polymorphism*s adalah kemampuan 2 buah *object* yang berbeda untuk merespon pesan permintaan yang sama dalam suatu cara yang unik.

Bagaimana hal ini berhubungan dengan OOP? Kamu dapat membuat *object*s yang dapat merespon pesan yang sama dalam implementasi yang unik. Contohnya, kamu dapat mengirim pesan print ke ***object*** printer yang akan mencetak pada printer, dan kamu juga dapat mengirim pesan yang sama ke *object* screen yang akan menuliskan pada screen monitor. Dalam OOP, kamu menerapkan tipe polymorphism melalui proses yang disebut *overloading*. Kamu dapat mengimplementasikan metode yang berbeda pada sebuah *object* yang mempunyai nama yang sama.

6. Inheritance

Menggunakan *inheritance* dalam OOP untuk mengklasifikasikan *object*s dalam program sesuai karakteristik umum dan fungsinya. Hal ini akan membuat pekerjaan bersama *object* lebih mudah dan lebih intuitif. Hal ini juga membuat programming lebih mudah karena memungkinkan kita untuk mengkombinasikan karakteristik umum kedalam *object* parent dan mewariskan karakteristik ini ke child *object*.

7. *Aggregation*

Aggregation adalah kondisi ketika ***object*** berisi gabungan dari *object*-*object* yang berbeda dan bekerja bersama. Contoh mesin pemotong rumput terdiri dari *object* roda, *object*s mesin, *object* pisau dan lain-lain. *Object* mesinpun merupakan gabungan dari berbagai *object*. Kemampuan untuk menggunakan

aggregation dalam OOP adalah satu feature yang powerful yang memungkinkan *model* menjadi akurat.

Pemrograman visual membawa OOP ke level berikutnya. Tujuan pemrograman visual adalah agar pemrograman menjadi lebih mudah bagi para programmer dan lebih mudah diakses oleh kalangan nonprogrammer dengan meminjam bahasa OOP, dan mempraktikanya secara grafis atau visual. Pemrograman visual memungkinkan pengguna untuk lebih focus pada pemecahan masalah ketimbang cara menangani bahasa pemrograman.

2.4.8 Object Relational Mapping (ORM)

ORM merupakan kependekan dari *Object Relational Mapping*, sebuah teknik pemrograman yang memetakan sebuah objek dengan *database*. ORM ini akan membantu menjadi 'jembatan' antara objek yang didefinisikan dalam kode program dengan *database*, bagaimana objek itu disimpan, diambil, dihapus dan sebagainya. Dengan ORM ini, programmer dibantu untuk melakukan aksi-aksi yang diperlukan terkait komunikasi objek ketika program dijalankan dengan *database* seperti menyimpan objek, mengambil data objek dari *database* kemudian ditampilkan, menghapus objek, mengubah objek dan sebagainya.

Sebagai contoh penggunaan adalah ketika ada definisi data berupa person yang memiliki atribut id dan nama. Di *database* didefinisikan sebuah tabel dengan kolom id dan nama. Dengan tanpa menggunakan ORM, untuk menyimpan objek itu ke *database* adalah dengan menjalan perintah SQL :

"INSERT INTO persons (id, nama) values ('1','wisnu manupraba')" .

Dengan menggunakan ORM, perintah SQL itu akan digantikan dengan fungsi misalkan *save* (tergantung fungsi yang disediakan oleh pustaka ORM yang digunakan), sehingga pemanggilannya adalah :

Person person = new Person (1, "Wisnu Manupraba");

person.insert();

ORM lah yang akan memetakan fungsi insert() sama dengan perintah SQL 'INSERT INTO….".

Ada banyak pustaka ORM yang tersedia saat ini, di berbagai bahasa. Di Java misalkan ada Hibernate dan MyIbatis (dulu namanya IBatis). Di PHP, berbagai *framework* seperti CakePHP, CodeIgniter, Yii dan sebagainya menyediakan ORMnya sendiri-sendiri.

Berikut adalah daftar ORM untuk masing-masing bahasa (tidak semuanya ada di sini dan yang ada di sini tidak semuanya masih bisa digunakan, beberapa sudah tidak digunakan lagi).

Tabel II.2. Daftar ORM untuk Masing-Masing Bahasa

Bahasa Programming	***Tools***	**Keterangan**
C++	ODB	*open source, compiler-based (no manual writing of the mapping code)*
	LiteSQL	*open source*
Coldfusion	Transfer ORM	*and object-relational mapper for CFML*
Cocoa	Core Data	*object graph management framework with several persistent stores. Ships with Mac OS X and iPhone OS*
Delphi	Bold for Delphi	
	ECO	*(by Capable Objects) available for Delphi 2007 and Delphi Prism*
	Synopse ORM	
Groovy	GORM	
Java	Cayenne	*Apache, open source for java*
	Kodo	*commercial implementation of both the JDO and JPA API*
	Torque	*an object-relational mapper for Java*
	Carbonado	*open source framework, backed by Berkeley DB or JDBC*
	Ebean	*open source ORM Framework*
	EclipseLink	*Eclipse Persistence Platform*
	Enterprise *Objects Framework*	*Mac OS X/Java, part of Apple*
.NET	ADO.NET Entity *Framework*	*Microsoft's ORM, part of .Net 4.0*
	AgileFx	*open source*

Bahasa Programming	***Tools***	**Keterangan**
	Devart	*LinqConnect, commercial, an ORM solution for Oracle, MySQL,*
Perl	DBIx::Class	
PHP	Axon	*ORM plug-in for the Fat-Free Framework. PDO-based and requires no configuration (GPL3).*
	CakePHP	*ORM and framework for PHP5, open source (scalars, arrays, objects). Based on database introspection. No class extending or code generation*
	Doctrine	*Open source ORM for PHP 5.2.3, free software (GNU LGPL)*
Python	ActiveRecord	*part of Ruby on Rails (open source)*
	Datamapper	*Free open source*
VB6	Database *Objec*ts	*Open source*

Kesimpulannya adalah salah satu kegunaan ORM mempermudah programmer agar tidak perlu lagi menuliskan perintah-perintah SQL ketika memerlukan koneksi dengan *database*, tetapi cukup dengan menggunakan fungsi-fungsi yang sudah disediakan oleh ORM tersebut.