# **DEFINISI-DEFINISI**

## **Pengertian Sistem**

Istilah sistem merupakan istilah dari bahasa yunani “*system*” yang artinya adalah himpunan bagian atau unsur yang saling berhubungan secara teratur untuk mencapai tujuan. Berikut ini pengertian sistem dari beberapa ahli :

Sistem menurut pendapat Satzinger, Jackson, & Burd (2010, p6) adalah kumpulan komponen-komponen yang saling berkaitan yang berfungsi bersama untuk mencapai beberapa hasil.

Sedangkan sistem menurut pendapat O'Brien, & Marakas (2009, p24) adalah sekelompok komponen yang saling berkaitan dan bekerja sama kearah tujuan bersama dengan menerima masukan-masukan dan menghasilkan keluaran dalam proses pengelolaan transformasi atau perubahan.

Menurut Romney dan Steinbart (2006, p4), sistem adalah kumpulan dari dua atau lebih komponen yang berinteraksi untuk mencapai tujuan. Sistem terdiri dari subsistem yang lebih kecil, masing-masing melaksanakan fungsi penting dan mendukung sistem yang lebih besar.

Berdasarkan pendapat-pendapat di atas, maka dapat disimpulkan bahwa definisi dari sistem adalah suatu koleksi dari komponen – komponen yang memiliki suatu kesatuan dan berhubungan dengan satu dengan yang lain dan memiliki fungsi yang sama.

**2.1.1 Pengertian Informasi**

Menurut Rainer & Cegielski (2010, p10), informasi mengacu pada data yang telah diorganisasikan sehingga mempunyai arti dan nilai bagi penerimanya.

Menurut O’Brien & Marakas (2009, p38), informasi merupakan data yang telah diubah menjadi konteks yang berarti dan berguna untuk penerima akhir tertentu.

Berdasarkan definisi diatas dapat disimpulkan bahwa informasi adalah data yang telah diolah untuk menghasilkan suatu hasil (*output*) dimana hasil dari proses tersebut bermanfaat bagi penerimanya.

**2.1.2 Pengertian Sistem Informasi**

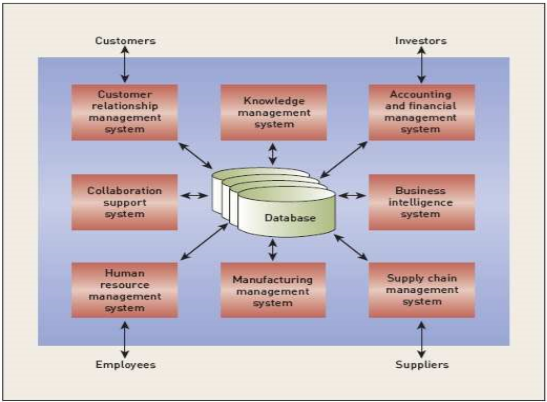
Menurut O’Brien & Marakas (2009, p4), Sistem Informasi adalah suatu kombinasi yang teratur baik dari orang – orang, *hardware*, *software*, jaringan komunikasi, dan sumber daya data dimana mencakup kegiatan mengumpulkan, mengubah, dan menyebarkan informasi dalam sebuah organisasi.

Menurut Satzinger, Jackson, & Burd (2010, p6), Sistem Informasi adalah kumpulan komponen yang saling berhubungan dimana mengumpulkan, memproses, menyimpan, dan menyajikan output dari informasi yang dibutuhkan untuk menyelesaikan tugas bisnis.

Setiap sistem mempunyai sebuah batas antara lingkungan sistem dan sistem itu sendiri. Setiap *input* maupun *output* harus melewati *system* *boundary*. *System* *boundary* adalah batas antara sebuah sistem dan lingkungannya dimana harus dilewati oleh *input* dan *output*. Dalam sebuah sistem informasi, manusia juga merupakan sebuah kunci komponen dan manusia tersebut melakukan sebuah kerja untuk diselesaikan oleh sebuah sistem. Jadi selain terdapat *system* *boundary*, juga terdapat batas lainnya yang dinamakan *automation* *boundary* yang merupakan hal penting untuk seorang sistem analis. *Automation* *boundary* adalah batas antara bagian sistem yang terotomatisasi dengan bagian sistem yang bersifat manual.

Jenis-jenis Sistem Informasi menurut Satzinger, Jackson, & Burd (2010, p9) sebagai berikut :

1. *Customer Relationship Management System*
2. *Supply Chain Management System*
3. *Accounting and Financial Management System*
4. *Human Resource Management System*
5. *Manufacturing Management System*
6. *Knowledge Management System*
7. *Collaboration Support System*
8. *Business Intelligence System*



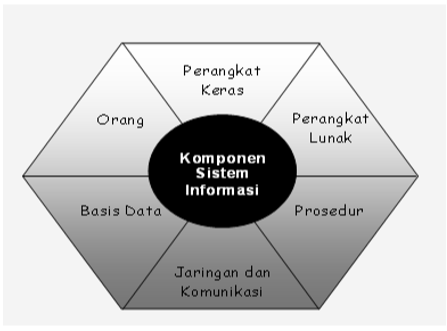
Gambar 2.1 Jenis-jenis Sistem Informasi

Sumber : (Satzinger, Jackson, Burd, 2010, P9)

**2.1.3 Komponen-Komponen Sistem Informasi**

Sistem informasi terdiri dari sekelompok komponen yang saling berhubungan, bekerja sama untuk mencapai tujuan bersama menerima input serta menghasilkan output dalam proses transformasi yang teratur (Ladjamudin, 2005, p14).

Kerangka konsep berbagai komponen dan aktivitas sistem informasi dapat dilihat pada gambar di bawah ini :



Gambar 2.2 Komponen-Komponen Sistem Informasi

Sumber : (Ladjamudin, 2005,p15)

Berdasarkan gambar di atas, komponen-komponen sistem informasi terdiri dari :

1. **Manusia/*User***

manusia diperlukan dalam operasi sistem informasi. Sumber daya manusia meliputi pemakai akhir dan pakar sistem. Pemakai akhir adalah orang yang menggunakan informasi yang dihasilkan sistem informasi, misalnya pelanggan, pemasok, teknisi, mahasiswa, dosen, dan orang-orang yang berkepentingan. Sedangkan pakar sistem informasi adalah orang yang mengembangkan dan mengoperasikan sistem informasi, misalnya system, analyst, developer, operator sistem dan staf administrasi lainnya (Mulyanto, 2009,p31).

1. **Perangkat Keras/*Hardware***

Sumber daya *hardware* adalah semua peralatan yang digunakan dalam memproses informasi, misalnya komputer dan periferalnya, lembar kertas, *disk* *magnetic* atau optik dan *flash* *disk* (Mulyanto, 2009, p31).

1. **Perangkat Lunak/*Software***

*Software* merupakan sekumpulan perintah/fungsi yang ditulis dengan aturan tertentu untuk memerintahkan komputer agar melaksanakan sesuatu (Ladjamudin, 2005, p15).

1. **Data**

Data merupakan dasar sumber daya organisasi yang diperlukan untuk memproses informasi. Data dapat berbentuk teks, gambar, audio maupun video. Sumber daya informasi umumnya diatur, disimpan dan diakses oleh berbagai pengelolaan sumber daya data ke dalam database dan dasar pengetahuan (Ladjamudin, 2005, p15).

1. **Jaringan/*Network***

Sumber daya jaringan merupakan media komunikasi yang menghubungkan komputer, pemroses komunikasi dan peralatan lainnya dengan kendali *software* komunikasi. Jaringan dapat berupa kabel, satelit, seluler dan pendukung jaringan seperti modem, *software* pengendali serta prosesor antar jaringan (Ladjamudin, 2005, p15).

Keseluruhan komponen sistem informasi tersebut saling terkait satu sama lain dalam sistem informasi. Sistem informasi dibangun menggunakan teknologi komunikasi dan informasi yaitu *hardware*, *software* dan jaringan. Ketiga komponen tersebut dipakai untuk mengolah data yang diperoleh untuk menghasilkan informasi yang lebih bermanfaat. Keseluruhan proses pengolahan informasi tidak lepas dari komponen manusia. Manusia adalah komponen penting sistem informasi karena sistem informasi adalah benda yang tidak bermanfaat bila tidak digunakan oleh manusia.

## **Pengertian Database**

Menurut O’ Brien dan Marakas (2009, p196) database adalah kumpulan terintegrasi dari elemen data yang secara logika saling berhubungan. Berbeda dengan Connolly & Begg (2005, p15) yang mengatakan bahwa database adalah kumpulan relasi-relasi logikal dari data, dan deskripsi dari data yang dapat digunakan bersama dan dibuat untuk memperoleh informasi yang dibutuhkan oleh perusahaan.

## **Pengertian Aplikasi**

Aplikasi merupakan suatu subkelas perangkat lunak komputer yang memanfaatkan kemampuan komputer langsung untuk melakukan suatu tugas yang diinginkan pengguna. Biasanya dibandingkan dengan perangkat lunak sistem yang meng-integrasikan berbagai kemampuan komputer.

Terdapat beberapa teori yang mendefinisikan aplikasi yang dikemukakan oleh beberapa para ahli, diantaranya adalah :

1. Menurut Hengky W. Pramana (2012, p17) Aplikasi adalah satu unit perangkat lunak yang dibuat untuk melayani kebutuhan akan beberapa aktivitas seperti sistem perniagaan, *game*, pelayanan masyarakat, periklanan, atau semua proses yang hampir dilakukan manusia.
2. Menurut Shelly, Cashman, Verman (2009, p57) Aplikasi adalah seperangkat instruksi khusus dalam komputer yang dirancang agar kita menyelesaikan tugas-tugas tertentu.
3. Menurut Dhanta (2009, p32), aplikasi adalah *software* yang dibuat oleh suatu perusahaan komputer untuk mengerjakan tugas – tugas tertentu.

Jadi Aplikasi merupakan sebuah program yang dibuat dalam sebuah perangkat lunak dengan komputer untuk memudahkan pekerjaan atau tugastugas tertentu seperti penerapan, penggunaan, dan penambahan data yang dibutuhkan.

Menurut O’brien (2010, p124) software komputer terbagi menjadi 2 yaitu :

1. *Application* *Software*

Mengarahkan kinerja penggunaan tertentu atau aplikasi komputer untuk memenuhi kebutuhan pengolahan informasi dari pengguna

Contoh : *word* *processing*.

1. *System* *software*

Mengendalikan dan mendukung operasi dari sistem komputer karena melakukan berbagai tugas pengolahan informasi.

Contoh : *operating* *system*, *network* *management*, *database*

*Management*.

## **Pengertian Aplikasi Berbasis Web**

Menurut O’Brien (2010, p157), *Web* *service* merupakan komponen *software* yang berbasis *framework* *web* dan standar *object*-*oriented* dan teknologi untuk penggunaan web yang secara elektronik menghubungkan aplikasi *user* yang berbeda dan *platform* yang berbeda. *Web* *service* dapat menghubungkan fungsi bisnis untuk pertukaran data secara *real* *time* dalam aplikasi berbasis web.

Banyak dari perusahaan - perusahaan berkembang yang menggunakan Aplikasi berbasis Web dalam merencanakan sumber daya mereka dan untuk mengelola perusahaan mereka. Aplikasi berbasis Web ini menggunakan protokol *HTTP*, aplikasi di sisi server berkomunikasi dengan *client* melalui *Web* *server*. Aplikasi di sisi *client* umumnya berupa *Web* *browser* jadi. Aplikasi berbasis *Web* (*client* / *server*-*side* *script*) berjalan di atas aplikasi berbasis internet.

Menurut Simarmata (2010, p185), aplikasi berbasis web adalah sistem perangkat lunak yang berdasarkan pada teknologi dan standar *World Wide Web Consortium* (*W3C*). Mereka menyediakan sumber daya web spesifik seperti konten dan layanan melalui sebuah antarmuka pengguna dan *browser* *web*.

1. **Tujuan Aplikasi Berbasis Web**

Tujuan aplikasi berbasis web, yaitu :

1. Aplikasi berbasis web dapat digunakan untuk membatu operasional perusahaan seperti membuat *invoice*, sistem informasi persediaan.
2. Memudahkan dalam penyimpanan data di *database*.
3. Aplikasi berbasis web juga dapat bekerja memonitoring sistem dalam hal tampilan, dapat didesain dan disesuaikan untuk berbagai jenis industri.
4. **Karakteristik Aplikasi Web**

Karakteristik aplikasi web antara lain :

1. **Kepadatan Jaringan**

Aplikasi web umumnya berada pada suatu jaringan komputer dan harus bisa melayani kebutuhan komunitas klien yang beragam.

1. **Keserempakan**

Sejumlah besar pengguna mungkin akan mengakses aplikasi secara serempak.

1. **Jumlah pengguna yang tidak dapat diprediksi**

Jumlah pengguna mungkin beragam dari hari ke hari. Pengguna di hari libur mungkin berada di hari kerja.

1. **Kinerja**

Pengguna aplikasi web akan pergi jika pengguna terlalu lama menunngu aplikasi web.

1. **Ketersediaan**

Menyediakan akses layanan 24 jam

1. **Digerakkan oleh data**

Beberapa aplikasi web bergantung pada layanan data, contoh : *onlineshop*.

1. **Peka terhadap isi**

Kualitas isi dan keindahan tetap merupakan faktor penting kualitas web.

1. **Evolusi yang berkesinambungan**

Merupakan hal yang biasa untuk aplikasi web jika ada pembaharuan setiap saat.

1. **Keamanan**

Terhubung ke internet, menyebabkan aplikasi web rentan serangan dari pihak lain.

1. **Estetika**

Salah satu daya tarik aplikasi web adalah tampilan dan nuansanya.

1. **Kesegaraan**

Aplikasi web sering memerlukan kedisiplinan waktu untuk merilis produk ke pasar dalam waktu beberapa hari atau beberapa minggu.

1. **Perbandingan Web dan Aplikasi Desktop**

*Tabel 2.1 Perbandingan Web dan Aplikasi Desktop*

|  |  |  |
| --- | --- | --- |
| Aspek | Aplikasi Web | Aplikasi Desktop |
| Kemudahan akses | Mudah diakses dari komputer atau perangkat *portable* yang memiliki akses internet. | tidak mudah diakses, karena harus mendatangi komputer yang terinstal program tersebut. |
| Instalasi | Aplikasi web memerlukan instalasi tunggal (yang diletakkan di *hosting*. | Aplikasi desktop harus diinstal secara individual pada setiap komputer. |
| *Maintenance* | Biasanya dimaintence oleh penyedia hosting yang bertanggung jawab terhadap web klien (tergantung penyedia hosting) | Dimaintence secara individu pada komputer yang terinstal program. |
| Keamanan | Lebih rentan terhadap gangguan keamanan seperti *hacking*, *cracking*, dll. | Gangguan keamanan rendah |
| Biaya | Membutuhkan biaya seperti berlanggan *domain*, *hosting*. Jika tidak diperbaharui, aplikasi tidak dapat diakses. | Tidak memerlukan biaya berlangganan, dikarena-kan sekali program dibuat hanya memerlukan biaya *updating*. |
| Koneksi | Bergantung pada *bandwidth* untuk mentransfer data ke user, jika *bandwidth* tidak terpenuhi web akan *blank*. | Tidak bergantung pada *bandwidth* yang disediakan. |

1. **Manfaat dari pengembangan aplikasi web**
2. **Kenyamanan**

Manfaat yang paling penting dari aplikasi berbasis web adalah bahwa yang paling nyaman digunakan. Satu dapat menggunakan aplikasi ini setiap saat dari lokasi manapun di seluruh dunia baik dengan menggunakan komputer atau telepon untuk mengakses data yang dibutuhkan sebagai seluruh database tersedia sepanjang waktu.

1. **Biaya yang lebih rendah**

Aplikasi berbasis web biaya murah dari pada aplikasi dekstop karena dukungan yang berkurang dan pemeliharaan, persyaratan lebih rendah pada sistem pengguna akhir dan arsitektur sederhana. Ini juga tidak menyia-nyiakan ruang komputer.

1. **Beberapa *Platform***

Biasanya semua aplikasi berbasis web jauh lebi harmonis dari seluruh platform perangkat lunak yang diinstal tradisional. Umumnya ada kebutuhan dari web browser seperti Internet Explorer, Firefox, Netscape. Mereka juga kompatibel dengan sebagian besar sistem operasi komputer (Windows, Linux, atau Mac). Material dari browser atau sistem operasi, tidak ada perbedaan dalam cara aplikasi bekerja. Kualitas pekerjaan tetap sepanjang masa yang sama.

1. ***Download* *software***

Selalu *up-to-date*, aplikasi berbasis web biasanya *up-to-date* tidak perlu menjadi di jalankan selalu, sebagai gradasi. Pengguna tidak harus menggagalkan tentang apakah aplikasi *up-to-date* atau tidak. Ini adalah salah satu manfaat dari aplikasi web *open* *source*.

1. **Kelebihan dan Kelemahan Aplikasi berbasis web**

Kelebihan kompetitif dari aplikasi berbasis web :

1. Aplikasi tersebut ringan dan dapat diakses selama ada koneksi internet atau intranet ke *server*.
2. Dapat diakses dengan menggunakan *browser* tanpa harus

menginstall aplikasi tersebut.

Kekurangan menggunakan aplikasi berbasis web :

1. Antarmuka yang dapat dibuat terbatas sesuai spesifikasi standar untuk membuat dokumen *Web* dan keterbatasan kemampuan *Web* *browser* untuk menampilkannya**.**
2. Terbatasnya kecepatan internet mungkin membuat respon aplikasi menjadi lambat**.**
3. Tingkat keamanan yang lebih rentan untuk diakses oleh orang lain atau pihak yang tidak berhak**.**

Oleh karena itu dapat disimpulkan, user dapat mengakses data atau informasi perusahaan mereka melalui laptop, smartphone, atau bahkan komputer PC dengan mudah tanpa harus menginstal terlebih dahulu aplikasi tersebut.

## **Pengenalan PHP (***Hypertext Prerocessing)*

PHP pertami kali dibuat oleh Rasmus Lerdford, seorang programmer C. Pada waktu itu PHP masih bernama *Form Interpreted* (FI), yang wujudnya berupa sekumpulan skrip yang digunakan untuk mengolah data *form* dari *web*. Jadi, awal mula PHP digunakannya untuk menghitung jumlah pengunjung didalam *web*-nya.

Dengan alasan untuk meningkatkan performa, Rasmus Lerdorf membuat ulang kode program tersebut dalam bahasaC. Lerdord menyebut kode program ini sebagai *Personal Home Page*. Versi ini pertama kali keluar pada tahun 1995. Isinya adalah sekumpulan skrip PERL yang dibuatnya untuk membuat halaman *web*-nya menjadi dinamis. Selanjutnya Rasmus merilis kode sumber tersebut untuk umum dan menamakannya PHP/FI, kedependekan dari *Hypertext Prerocessing / Form Interpreter*.

Dengan perilisan kode sumber ini menjadi open source, maka banyak programmer yang tertarik untuk ikut mengembangkan PHP. Kemudian pada tahun 1996 ia mengeluarkan PHP versi 2.0 yang kemampuannya telah dapat mengakses database dan dapat terintegrasi dengan *Hypertext* *Markup Language* (HTML). Pada rilis ini interpreter PHP sudah diimplementasikan dalam program C. Dalam rilis ini disertakan juga modul-modul ekstensi yang meningkatkan kemampuan PHP / FI secara signifikan. PHP versi 2.0 ini telah menarik banyak perahtian programmer, namun bahasa ini memiliki masalah dengan kestabilan yang kurang bisa diandalkan. Hal ini lebih dikarenakan Lerdorf hanya bekerja sendiri untuk mengembangkan PHP.

Pada saat itulah Zeev Suraski dan Andi Gutmans, ikut mengambil bagian dan membuat ulang parsing engine yang menjadi dasar dari PHP agar lebih stabil. Dengan dukungan dari banyak programmer lainnya, proyek PHP secara perlahan beralih dari proyek satu orang menjadi proyek masal yang lebih akrab kita kenal *open-source project*. PHP selanjutnya dikembangkan oleh The PHP Group yang merupakan kumpulan banyak programmer dari seluruh dunia.

Pada tahun 1998 tepatnya pada tanggal 6 Juni 1998 keluarlah PHP versi 3.0 yang dikeluarkan oleh Rasmus sendiri bersama kelompok pengembang *software*-nya.

PHP versi 4.0 keluar pada tanggal 22 Mei 2000 merupakan versi yang lebih lengkap lagi dibandingkan dengan versi sebelumnya. Perubahan yang paling mendasar pada PHP 4.0 adalah terintegrasinya Zend Engine yang dibuat oleh Zeev Suraski dan Andi Gutmans yang merupakan penyempurnaan dari PHP *scripting engine*, yang lainnya adalah build in HTTP *session*, tidak lagi menggunakan library tambahan seperti pada PHP. Tujuan dari bahasa *scripting* ini adalah untuk membuat aplikasi-aplikasi yang dijalankan di atas teknologi web. Dalam hal ini, aplikasi pada umumnya akan memberikan hasil pada *web browser*, tetapi prosesnya secara keseluruhan dijalankan *web server*.

PHP 4.0 adalah versi PHP yang paling banyak dipakai pada awal abad ke-21. Versi ini banyak dipakai disebabkan kemampuannya untuk membangun aplikasi web kompleks tetapi tetap memiliki kecepatan dan stabilitas yang tinggi. Pada Juni 2004, Zend merilis PHP 5.0. Dalam versi ini, inti dari interpreter PHP mengalami perubahan besar. Versi ini juga memasukkan model pemrograman berorientasi objek ke dalam PHP untuk menjawab perkembangan bahasa pemrograman ke arah paradigma berorientasi objek. Beberapa penambahan fitur meliputi PHP *Data* *Objects* (PDO) untuk mengakses *database*, *closures*, *trait*, dan *namespaces*.

Versi lanjutan dari PHP, yakni PHP 6.x sebenarnya telah lama dikembangkan, bahkan sejak tahun 2005. Fokus pengembangan PHP 6 terutama dalam mendukung *unicode*, agar PHP bisa mendukung berbagai jenis karakter bahasa *non-latin*.

Namun dikarenakan beberapa alasan seperti kurangnya programmer, dan performa yang tidak memuaskan, pengembangan PHP 6 dihentikan dan fitur yang ada dimasukkan kedalama PHP 5.

Pada tahun 2014, sebuah proyek lanjutan PHP mulai mengemuka, yakni PHP 7 yang berkembang dari banyak eksperimen yang dinamakan PHP *Next* *Generation* (PHPNG), yang dikembangkan Dmitry Stogov, Xinchen Hui, dan Nikita Popov. Proyek ini menggunakan pendekatan modern agar PHP diproses lebih cepat seperti memakai teknik just-in-time (JIT) compiler. Proyek PHPNG bertujuan untuk menyusun ulang kode PHP untuk meningkatkan performa. Selain performa yang meningkat, terdapat beberapa fitur baru pada PHP 7, seperti *combined* *comparison* *operator* atau dikenal dengan *spaceship* *operator* “”, *anonymous* *classes*, dan dukungan yang lebih stabil untuk server 64-bit.

Beberapa fitur yang sudah usang (*deprecated*) juga dihapus, seperti penulisan PHP dengan ASP style <% %> dan tag <script language=php></script>. Kedua cara ini sudah tidak bisa digunakan lagi. Modul *mysql* *extension* juga dihapus karena sudah digantin dengan mysqli extension. Penghapusan modul mysql *extension* ini sering menjadi masalah karena banyak kode program atau buku PHP lama yang masih menggunakannya.

1. **Pengertian PHP**

Menurut Doyle (2009, p3), PHP adalah bahasa pemrograman untuk membuat situs web dinamis dan interaktif. PHP berjalan di *server* *web* dan melayani pengunjung dengan halaman web sesuai permintaan. Artinya adalah sintaks dan perintah-perintah yang kita berikan akan sepenuhnya dijalankan di server tetapi disertakan HTML biasa.

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

Tujuan dari bahasa *scripting* ini adalah untuk membuat aplikasi aplikasi yang dijalankan diatas teknologi web. Dalam hal ini, aplikasi pada umumnya akan memberikan hasil pada web *browser*, tetapi prosesnya secara keseluruhan dijalankan *web* *server*.

Kelebihan PHP Ketika *e*-*commerce* semakin berkembang, situs-situs yang statispun semakin ditinggalkan, karena dianggap sudah tidak memenuhi keinginan pasar, padahal situs tersebut harus tetap dinamis. Pada saat ini bahasa PERL dan CGI sudah jauh ketinggalan jaman sehingga sebagian besar *designer* *web* banyak beralih ke bahasa *server-side scripting* yang lebih dinamis seperti PHP. Seluruh aplikasi berbasis web dapat dibuat dengan PHP. Namun kekuatan yang paling utama PHP adalah pada konektivitasnya dengan system database didalam web.

1. **Fungsi PHP**

Salah satu fungsi dari PHP ini dapat disisipkan pada dokumen HTML. Karena kemampuan inilah PHP juga sering disebut sebagai bahasa pemrograman script atau scripting language. Berikut ini merupakan sintaksis dasar pada PHP.

1. Pembatas

PHP hanya mengeksekusi kode yang ditulis dalam pembatas sebagaimana ditentukan oleh dasar sintaks PHP. Apapun di luar pembatas tidak diproses oleh PHP. Pembatas paling umum adalah “<?php” untuk membuka dan “?>” untuk menutup kode PHP. Tujuan dari pembatas ini adalah untuk memisahkan kode PHP dari kode di luar PHP, seperti HTML, dan Javascript.

1. Variabel

Variabel dalam PHP diawali dengan simbol dolar “$”. Pada versi PHP 5 diperkenalkan jenis isyarat yang memungkinkan fungsi untuk memaksa mereka menjadi parameter objek dari *class* tertentu, *array*, atau fungsi. Namun, jenis petenjuk tidak dapat digunakan dengan jenis skalar seperti angka atau *string*. Contoh variabel dapat ditulis sebagai $*nama\_variabel*.

1. Komentar

PHP memiliki 3 jenis sintaks sebagai komentar pada kode yaitu blok “/ \* \* /”, komentar 2 baris “//”, serta tanda “#” digunakan untuk komentar 1 baris. Komentar bertujuan untuk meninggalkan catatan pada kode PHP dan tidak akan diterjemahkan ke program.

1. **Keunggulan & Kekurangan**
2. Keunggulan bahasa pemrograman PHP, antara lain :
3. Bahasa pemrograman PHP adalah sebuah bahasa *script* yang tidak melakukan sebuah kompilasi dalam penggunaannya.
4. *Web* *server* yang mendukung PHP dapat ditemukan di mana-mana dari mulai *apache, IIS, Lightpad*, hingga *xitami* dengan konfirgurasi yang relatif mudah.
5. Dalam sisi pengembangan lebih mudah, karena banyaknya *developer* yang siap membantu dalam pengembangan.
6. Dalam sisi pemahaman, PHP adalah bahasa *scripting* yang paling mudah karena memiliki referensi yang banyak.
7. PHP adalah bahasa *open source* yang digunakan di berbagai mesin (Linux, Unix, Macintosh, Windows) dan dapat dijalankan secara *runtime* melalui *console* serta dapat menjalankan perintahperintah sistem.
8. Kekurangan bahasa pemrograman PHP, antara lain :
9. Tidak ideal jika untuk pengembangan skala besar.
10. Tidak memiliki sistem pemrograman berorientasi objek yang sesungguhnya.
11. Tidak dapat memisahkan antara tampilan dengan logik dengan baik.
12. PHP mempunyai kelemahan *security* tertentu yang mana jika *programmer* tidak jeli dalam melakukan pemrograman dan kurang memperhatikan isu dan konfigurasi PHP.

## **Faktor Yang Mempengaruhi Perilaku Pelanggan**

Konsep *Object Oriented Programming* (OOP) pertama kali muncul di MIT (*Massachusetts Institute of Technology*) pada era 1960-an. Sekitar beberapa tahun kemudian antara 1962-1965, sebuah basaha pemrograman yang mendasari konsep OOP diperkenalkan dengan nama bahasa pemrograman SIMULA 1, dikembangkan oleh Kristen Nygaard dan OleJohan yang merupakan warga negara Norwegia. Setelah itu pada tahun 1967 keluarlah SIMULA 67.

Bahasa SIMULA dengan OOP menginspirasi banyak pengembang, sehingga pada tahun 70-an tercipta bahasa SMALTALK, bahasa yang pertama kali disebut oleh dunia sebagai *object oriented programming*, dengan membawa fitur barunya yaitu *inheritance*.

Semenjak itu konsep OOP semakin dikenal, pada tahun 1980-an banyak bahasa pemrograman menggunakan konsep OOP, diperkenalkan kepada dunia, tetapi hanya empat yang cukup populer, diantaranya ADA (US Department of Defense), PROLOG (The Japanese “Fifth Generation Computer Project), Eifle dan C++. Bahasa pemrograman ADA dan PROLOG dipercaya akan bersaing ketat sebagai bahasa pemrograman yang paling dominan.

Namun pada tahun 1980-an bahasa pemrograman C++ mematahkan kepercayaan tersebut. Bahasa pemrograman C++ menjadi bahasa pemrograman yang populer dan mendominasi hingga sekarang. Bahasa pemrograman C++ yang merupakan gabungan dari 2 konsep bahasa pemrograman, yakni C dan SIMULA.

Semenjak C++ terkenal, banyak sekali pengembang yang terinspirasi oleh C++ dan pada tahun 1990-an, bahasa pemrograman Java diperkenalkan yang mengaku terinspirasi oleh C++, dan tahun 2002 perusahaan Microsoft juga mengeluarkan bahasa turunan dari C++ yaitu C# (C-Sharp), disusul dengan VB.Net dengan fitur OOP yang merupakan penyempurnaan dari bahasa VB 0.6 yang tidak mendukung fitur OOP.

1. **Pengertian OOP**

*Object Oriented Programming* atau disingkat menjadi OOP adalah paradigma pemrograman dalam melakukan pemrograman yang berorientasi kepada objek, semua fungsi, data dan pengolahan data akan dibungkus dalam kelas-kelas dan objek-objek.

Masing-masing objek dapat memiliki sifat dan tugasnya. Pada paradigma ini, objek-objek tersebut dapat bekerja sendiri dan juga dapat saling bekerja sama dengan kemungkinan untuk saling berhubungan, seperti menerima, mengirim data kepada objek lainnya dan memproses data.

Paradigma OOP dapat dilihat sebagai interaksi dari objek yang saling berhubungan satu sama lain untuk melakukan tugasnya. Pengembangan program OOP dilakukan dengan cara mengikuti model yang telah ada di dalam dunia nyata.

OOP bertujuan untuk memberikan pola pikir dalam mengembangkan program, pola pikir tersebut dipercaya dapat memberikan kemudahan, fleksibilitas, kemudahan pembuatan, pengembangan program, dan perawatan program.

1. **Jenis-jenis OOP pada bahasa pemrograman**

Karena kemudahan yang diberikan oleh konsep OOP, banyak bahasa yang membawa dukungan fitur OOP, karena hal itu OOP menjadi memiliki dan dibagi menjadi 3 jenis, yaitu :

1. Bahasa OOP murni, merupakan sebuah bahasa yang mengharuskan program ditulis hanya berupa objek saja. Contoh – Eifel, Smaltalk, Ruby, Jade, dan lain-lain.
2. Bahasa OOP *hybrid*, merupakan bahasa yang dirancang untuk pemrograman objek dengan beberapa elemen prosedural.
3. Bahasa OOP *hybrid* dalam *web*, salah seperti bahasa OOP *hybrid*, yang membedakan hanya konsep yang digunakan dalam pemrograman *web*.
4. **Konsep Object Oriented Programming**
5. *Class*, adalah sebuah rancangan untuk mendefinisikan karakter dan perilaku dari objek, yang merupakan kumpulan atas definisi dan fungsi-fungsi dalam suatu unit, untuk suatu tujuan tertentu.
6. *Object*, adalah dasar dari modularitas dan struktur pada OOP, dan merupakan representasi dari class, objek akan memiliki sifat dan perilaku dari class yang digunakan.
7. *Encapsulation*, adalah konsep dalam implementasi untuk membungkus data dan fungsi menjadi satu entitas, dan membatasi akses dari luar *class*.
8. *Inheritance*, adalah konsep pewarisan *class*. *Class* juga dapat menuruni dan memiliki apa yang dimiliki oleh *class* lainnya.
9. *Abstraction*, adalah konsep untuk mendesain sebuah objek, teknik dalam menyembunyikan detail suatu proses dalam objek tersebut, dengan tujuan untuk memfokuskan pengguna pada fungsi inti objek.
10. *Polymorphism*, adalah kemampuan dalam menyampaikan pesan tertentu keluar dari hierarki objeknya, dimana objek yang berbeda memberikan tanggapan atau respon terhadap pesan yang sama sesuai dengan sifat masing-masing objek.
11. **Keunggulan & Kekurangan OOP**
12. Keunggulan OOP, diantaranya :
13. OOP menyediakan struktur modular yang jelas untuk program sehingga OOP sangat bagus digunakan untuk mendefinisikan tipe data abstrak dimana detail implementasinya tersembunyi.
14. OOP akan mempermudah dalam memaintain dan memodifikasi kode yang sudah ada. Objek yang baru dapat dibuat tanpa mengubah kode yang sudah ada.
15. OOP menyediakan *framework* untuk *library* kode dimana komponen *software* yang tersedia dapat dengan mudah diadaptasi dan dimodifikasi oleh *programmer*. Hal ini sangat berguna untuk mengembangkan *Graphical User Interfaces* (GUI).
16. Kekurangan OOP, diantaranya :
17. Tidak memperbolehkan implementasi yang kuat pada *reuse*.
18. *Property* *software* tidak terikat dalam satu unit fungsional sehingga harus *crosscut* di antara komponennya.
19. *Crosscut* tersebut mengakibatkan sulitnya pengembangan dan pemeliharaan.

## **Pengertian *Framework***

*Framework* adalah kerangka kerja. *Framework* merupakan sekumpulan *script* *class* dan *function* yang dapat memudahkan *developer*/*programmer* dalam menangani suatu permasalahan dalam pembuatan aplikasi seperti pemanggilan *variable*, koneksi ke *database*, dll. sehingga *developer* bisa lebih fokus dan lebih cepat dalam membangun sebuah aplikasi.

*Framework* bisa disebut komponen pemrograman yang siap digunakan kembali kapan saja, sehingga *programmer* tidak harus membuat *script* yang sama secara berulang-ulang.

Secara sederhana dapat dijelaskan bahwa *framework* merupakan sekumpulan fungsi (*libraries*), maka seorang *programmer* tidak perlu lagi membuat fungsi-fungsi (biasanya disebut kumpulan *Library*) dari awal, *programmer* tinggal memanggil kumpulan *library* atau fungsi yang sudah ada didalam *framework*, tentunya cara menggunakan fungsi-fungsi itu sudah ditentukan oleh *framework*. Berikut merupakan beberapa contoh fungsi standar *framework* diantaranya fungsi grafik, tabel bergaya zebra, validasi, *upload*, *captcha*, proteksi terhadap XSS (XSS *filtering*), *template*, kompresi, XML, *paging*, enkripsi, email, SEO, *session*, *security*, kalender, bahasa, manipulasi gambar, dan lain-lain.

1. **Fungsi *Framework***
2. Dapat membantu kerja developer dalam membangun aplikasi sehingga aplikasi bisa diselesaikan dalam waktu yang singkat**.**
3. Penerapan *design* *patterns* memudahkan dalam rancangan, pengembangan dan pemeliharaan sistem.
4. *Stability* dan *Reability* yang dibangun didalam sistem lebih stabil dan handal karena berbasis pada *framework* yang sudah teruji stabilitas dan kehandalannya.
5. *Coding Style* konsisten, memudahkan dalam membaca kode dalam menemukan *bugs*.
6. *Security* *Concern* dan *framework* akan mengantisipasi dengan memasang perisai terhadap adanya berbagai masalah keamanan yang mungkin terjadi.
7. Dokumentasi, *framework* dapat mendisiplinkan kita untuk menulis dokumen apa yang kita tulis.
8. **Kelebihan dan Kelemahan *Framework***

Kelebihan Framework antara lain :

1. Lebih cepat dan efisien

Jika mengerjakan proyek besar, maka penggunaan *framework* dapat membantu mempercepat proses pengembangan. Pada umumnya, *framework* memiliki beragam fungsi dan plugin yang bisa dimanfaatkan. Dalam kerangka kerja ini, maka proses pengembangan proyek jauh lebih cepat daripada harus menulis kode dari awal. Selain itu, tidak perlu menulis berulangulang untuk kode yang bersifat repetitif.

1. Menghemat biaya

Sebagian besar bersifat *open* *source* dan gratis untuk bisa digunakan. Biaya yang harus dikeluarkan oleh *client* juga akan menjadi lebih kecil karena proses pengerjaan yang lebih simple dan lebih cepat.

1. Memperhatikan faktor keamanan

*Framework* telah banyak digunakan oleh *developer*, dan kemungkinan adanya masalah keamanan maupun *bug* yang telah diperbaiki. Selain itu, *framework* biasanya memiliki komunitas dalam jumlah cukup banyak yang dapat berperan dalam jangka panjang. Setiap kali pengguna menemukan celah keamanan, maka mereka dapat memberi tahu tim untuk segera memperbaikinya.

Kelemahan *Framework* antara lain :

1. Kurangnya pemahaman bahasa pemrograman

Jika bekerja menggunakan kerangka kerja kemudian hanya mengetahui sedikit tentang bahasa pemrograman yang digunakan, maka developer hanya mempelajari mengenai kerangka tersebut. Sehingga pemahaman mengenai bahasa pemrograman menjadi tidak berkembang.

1. Memiliki batasan

Dalam penggunaannya, kerangka ini juga memiliki beberapa batasan yang tidak dapat Anda modifikasi. Sehingga Anda harus bekerja sesuai dengan standar yang digunakan di dalamnya. Oleh karena itu, ketika mengembangkan sebuah aplikasi, maka harus menggunakan kerangka kerja yang sesuai dengan kebutuhan *developer*.

1. Kode Publik

Karena bersifat publik, maka kode dapat digunakan siapa saja termasuk pihak-pihak yang mempunyai niat buruk.

## **Pengenalan Codeigniter**

Kelahiran codeignier adalah bermula dari kegalauan Rick Ellis atas banyaknya kode PHP yang harus ditulis ketika membangun salah satu CMS kesayangannya, *expression engine*. Rick Ellis ingin mempermudah penulisan kode-kode program PHP dan membuatnya lebih singkat dan cepat. Akhirnya Rick Ellis mengambil inisiatif dengan membuat sendiri kode singkat / *shortcode* dari fungsi-fungsi yang ada di PHP.

Pembuatan *shortcode* tersebut membuatnya mampu membangun *expression engine* dengan sangat bagus, efisien dan cepat. Selain itu, performanya juga sangat bagus.

Setelah tidak berapa lama, Rick Ellis melalui situsnya ellislab(dot)com membagikan *shortcode* yang dibuat sendiri itu untuk digunakan oleh *developer* lainnya. Tujuannya yaitu membantu *developer* lain dalam menangani masalah dalam *framework* yang terkenal. Banyak *developer* turun tangan membantu perkembangan CodeIgniter dan puncaknya, CI menjadi *framework* terpopuler tahun 2006.



Gambar 2.1 Logo CodeIgniter

Sumber :

1. **Pengertian CodeIgniter**

Codeigniter merupakan sebuah *web framework* yang dikembangkan oleh Rick Ellis dari Ellis Lab. Codeigniter dirancang untuk menjadi sebuah web framework yang ringan dan mudah untuk digunakan. Bahkan pengakuan dari Rasmus Lerdorf, sang pencipta bahasa pemrograman PHP mengatakan bahwa Codeigniter merupakan web framework yang mudah, cepat dan handal.

Codeigniter dirilis pertama kali pada tanggal 28 Februari 2006, namun pada bulan Juli 2013 Ellis Lab mengumumkan bahwa mereka mencari pemilik baru untuk Codeigniter karena pada lingkup internal sendiri tidak memiliki cukup fokus untuk mengembangkan Codeigniter, dan pada akhirnya pada bulan Oktober 2014, kepemilikan Codeigniter berpindah ke tangan British Columbia Institute of Technology, yakni salah satu sekolah tinggi teknologi di Kanada.

Setelah hampir lima bulan lamanya sejak peralihan kepemilikan, BCIT akhirnnya merilis Codeigniter 3.0. Dibanding versi sebelumnya tentunya Codeigniter 3 mempunya fitur yang lebih kaya seperti pengembangan *Database Driver*, terdapat pustaka yang baru dan juga PDO Codeigniter kini telah berfungsi secara penuh dengan subdriver.

Sebelum mencoba Codeigniter, perlu kita ketahui istilah *web framework* itu sendiri. Menurut *Microsoft Computer Dictionary*, *web* adalah sekumpulan dokumen yang saling terhubung dalam sistem *hypertext* yang penggunanya akan menjelajahi web melalui halaman beranda. Sedangkan *framework* adalah desain struktur dasar yang dapat digunakan kembali (*reusable*) yang terdiri dari *abstract class* dan *concrete class* di pemrograman yang berorientasi objek.

Dirilis pada dokumentasi Codeigniter, Codeigniter merupakan toolkit bagi orang yang ingin membangun aplikasi *web* menggunakan PHP. Tujuannya adalah membuat pengembangan proyek menjadi lebih cepat dibandingkan dengan menulis kode dari awal (*stracth*). Codeigniter menyediakan kumpulan *library* untuk tugas-tugas yang sering dilakukan dan sangat mudah untuk mengakses *library* yang tersedia di Codeigniter. Dengan Codeigniter, kita cukup fokus pada pengembangan *project* dan meminimalisir kode yang akan ditulis.

1. **Fitur-fitur CodeIgniter**

Menilai baik dan buruknya sebuah aplikasi dari fiturnya,bukanlah sebuah cara yang tepat. Karena ini tidak menggambarkan bagaimana sebuah aplikasi dirancang. Fitur-fitur tersebut tidak menggambarkan kualitas kode, *performance* dan keamanan (*security*). Satu-satunya cara untuk mengetahui yaitu dengan melihat kodenya.

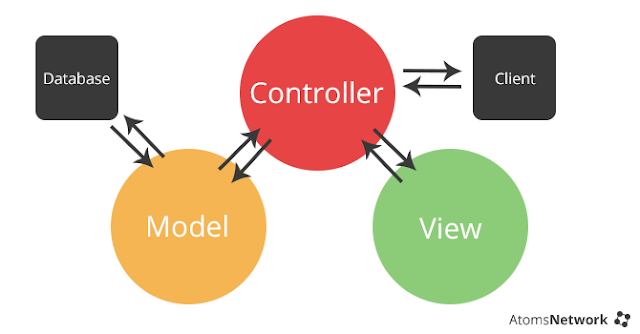
Berikut ini adalah fitur-fitur yang didukung oleh Codeigniter :

* *Model-View-Controller Based*
* PHP 7 *Compatible*
* *Extremely Light Weight*
* *Full Featured database classes with support for several platform*
* *Active Record Databse Support*
* *Forms and Data Validation*
* *Security* and XSS *Filtering*
* *Session* *Management*
* Email *Sending* *Class*, *Support* *Attachment*, HTML/Text email, *multiple* protocols (sendemail, SMTP, and Mail)
* *Image* *Manipulation* *Library* (*cropping*, *resizing*, *rotation*) *Support* GD, *Image* *Magic*, and NetPBM
* File *Uploading* Class
* FTP *Class*
* Localization
* PAGINATION
* Data Encryption

1. **Konsep MVC**

Codeigniter menggunakan pendekatan Model-View-Controller, yang bertujuan untuk memisahkan logika dan presentasi. Konsep ini mempunyai keunggulan dimana desainer dapat bekerja pada template file, sehingga redudansi kode presentasi dapat diperkecil. Berikut adalah konsep Model-View-Controller yang diterapkan pada Codeigniter.

1. Model, menggambarkan struktur data. Biasanya kelas model akan berisi fungsi yang digunakan untuk mengambil, menambah dan memperbaharui informasi yang ada didatabase.
2. View, informasi yang diperlihatkan kepada user. View adalah halaman web yang terdiri dari HTML, CSS dan Javascript, akan tetapi pada codeigniter, view dapat juga sebagai potongan halaman seperti header atau footer. Bahkan dapat juga halaman RSS atau tipe halaman lainnya.
3. Controller, perantara Model, View, dan *resource* lainnya yang dibutuhkan untuk menangani HTTP *request* dan menghasilkan halaman web.



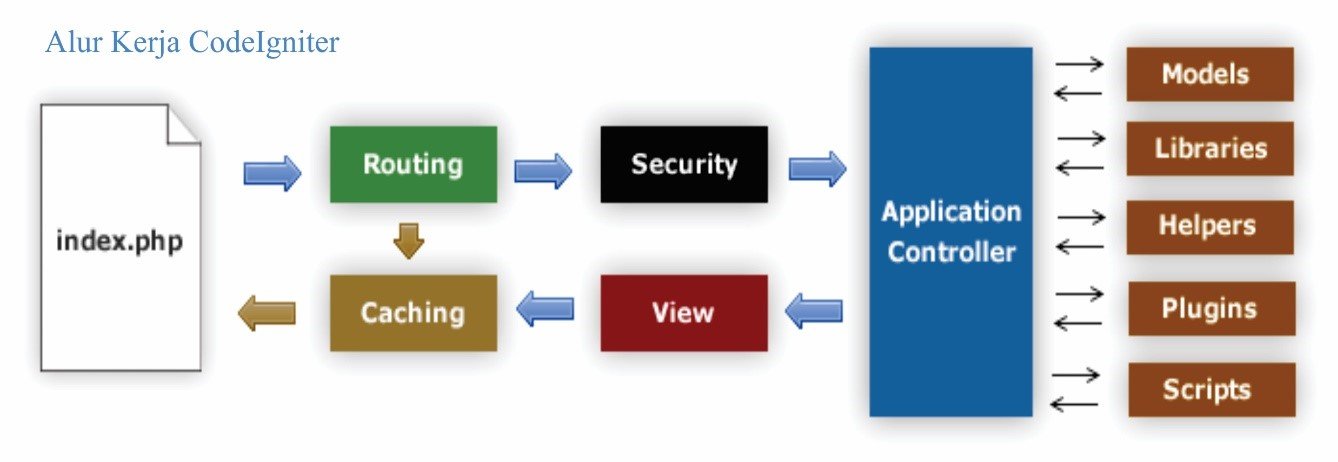
Gambar 2.2 Cara kerja MVC

Sumber :

Pada Codeigniter, programmer juga dapat mengabaikan pemakaian Model dan cukup menggunakan Controller dan View. Cara tersebut sah-sah saja dipakai, tetapi akan lebih baik jika kita memakai Model juga agar hierarki MVC tercapai dan disamping itu beban di Controller akan berkurang.

1. **Sistem Kerja CodeIgniter**

Untuk melengkapi pemahaman mengenai Codeigniter, berikut adalah gambaran sederhana yang menjelaskan bagaimana Codeigniter bekerja :

Gambar 2.3 Alur kerja CodeIgniter

Sumber :

1. Index.php: berfungsi sebagai file pertama dalam program yang akan dibaca oleh program.
2. Router : router akan memeriksa HTTP request untuk menentukan hal apa yang harus dilakukan oleh program.
3. Cache File : apabila dalam program sudah terdapat “*cache file*” maka file tersebut akan langsung dikirim ke browser. File cache inilah yang dapat membuat sebuah *website* dapat dibuka dengan lebih cepat. *Cache* file dapat melewati proses yang sebenarnya harus dilakukan oleh program codeigniter.
4. *Security* : Sebelum *controller* di *load* secara keseluruhan , maka data yang disubmit oleh *user* dalam bentuk *request* HTTP akan di periksa terlebih dahulu melalui *security* yang dimiliki oleh *Codeigniter*.
5. Controller : controller akan membuka file bagian model, core *libraries*, *helper* dan semua *resources* yang dibutuhkan dalam program tersebut.
6. *View*: bagia akhir akan dilakukan pengecekan semua program yang ada dalam *view* kemudian file akan mengirimkannya ke *browser* supaya dapat dilihat. Apabila file *view* sudah ada yang di “*cache*” maka file *view* baru yang belum ter-*cache* akan *update* file *view* yang sudah ada.
7. **Fungsi CodeIgniter**
8. Mempercepat dan mempermudah kita dalam pembuaan *website***.**
9. Menghasilkan struktur pemrograman yang sangat rapi, baik dari segi kode maupun struktur file phpnya**.**
10. Memberikan standar coding sehingga memudahkan kita atau orang lain untuk mempelajari kembali sistem aplikasi yang dibangun**.**
11. **Keuntungan CodeIgniter**

Sebagai *web framework* populer yang menggunakan bahasa pemrograman PHP, Codeigniter mempunyai beberapa keunggulan seperti yang telah disebutkan di dokumentasinya, keunggulan-keunggulan tersebut meliputi :

1. *Free*, karena berada di bawah lisensi open source, kita dapat melakukan apapun dengan Codeigniter. Lisensi lengkapnya dapat dokumentasi.
2. *Light weight*, sistem inti Codeigniter memerlukan library yang sedikit. Sangat berbeda dengan *framework* lainnya yang membutuhkan banyak sumber daya tambahan. *Library* tambahan akan digunakan ketika *request* secara dinamis, membuat *system* yang dibangun menjadi efisien dan cukup cepat.
3. *Fast*, menurut dokumentasi, performa yang dimiliki Codeigniter terbukti cepat setelah dibandingkan dengan framework lainnya.
4. Menggunakan kaidah MVC, dengan menggunakan Model-View-Controller, kita dapat memisahkan bagian logic dan presentasion dari aplikasi yang kita bangun. Hal ini tentu sangat cocok dan bagus untuk proyek yang memfokuskan. Desainer fokus pada template file dan programmer fokus pada pembangunan logic dari aplikasi yang dibangun.
5. Menghasilkan URL yang bersih, URL yang dihasilkan oleh Codeigniter bersih dan ramah terhadap *search engine*. Codeigniter menggunakan pendekatan *segment-based* dibandingkan dengan query string yang biasa digunakan oleh programmer yang tidak menggunakan *web framework*.
6. *Packs a Punch*, Codeigniter hadir dengan berbagai *library* yang akan membantu tugas-tugas di pengembangan web yang sudah umum dan sering dilakukan seperti mengakses database, mengirim email, validasi data dari form, mengelola *session*, memanipulasi gambar, dan masih banyak lagi.
7. *Extensible*, kita dapat menambahkan library atau helper yang kita ciptakan sendiri ke dalam Codeigniter. Selain itu kita dapat juga menambahkan fitur lewat class *extensions* atau system hooks.
8. *Thoroughly Documented*, hampir semua fitur, *library* dan helper yang ada di Codeigniter telah terdokumentasi dengan lengkap dan tersusun dengan baik. Ketika mendapatkan unduhan Codeigniter, dokumentasinya sudah tersedia dan siap digunakan.
9. Mempunyai komunitas yang ramah, bergabung di komunitas Codeigniter tentunya akan membantu sekali para pengguna Codeigniter yang masih pemula atau yang sudah mahir untuk saling berbagai ilmu pengetahuan. Komunitas ini dapat ditemui di *codeigniter.com/forums*.
10. **Kekurangan CodeIgniter**
11. Codeigniter tidak ditujukan untuk pembuatan *web* dengan skala besar.
12. *Library* yang sanga terbatas. Hal ini dikarenakan sangat sulit mencari plugin tambahan yang terverifikasi secara resmi, karena pada situsnya Codeigniter tidak menyediakan *plugin*-*plugin* tambahan untuk mendukung pengembangan aplikasi dengan CI.
13. Belum adanya *editor* khusus Codeigniter, sehingga dalam melakukan *create project* dan modul-modulnya harus berpindah-pindah folder.