BAB II

TINJAUAN PUSTAKA

2.1 Profil Tempat Kerja Praktek

2.1.1 Sejarah Instansi

Science Group (SG) adalah lembaga pendidikan yang fokus dan komitmen untuk membangun kualitas sumber daya manusia (SDM) Indonesia yang unggul. SDM yang unggul harus dibentuk melalui sebuah rangkaian proses pendidikan berkualitas yang terstruktur, sejak dari tingkat dasar, menengah, hingga pendidikan tinggi.

Science Group Indonesia Berdiri dan dikembangkan sejak 1 Januari 2004 di Bandung-Indonesia, bermula dari sebuah kantor di jalan Buton No. 1 Pav Bandung 40112. Hingga saat ini (2012) SG telah mengoperasikan 1 kantor pusat, 2 gerai cabang dan 1 gerai kemitraan di Bandung, yaitu :

1. **Kantor Pusat Science Group Indonesia.**
   Jalan Buton No. 1 Pav - Bandung 40112.
2. **Kantor Cabang Batu Nunggal.**
   Jalan Batu Nunggal 2 Bandung.
3. **Kantor Cabang Laurentius.**
   Komplek SMA St. Aloysius III. Jalan Sukajadi 487, Bandung.
4. **Gerai kemitraan GEMA.**
   Komplek SMA St. Aloysius I. Jalan Sultan Agung No. 2, Bandung.

Sistem kemitraan bisnis (Business Partner) Science Group dimulai sejak Januari 2011, untuk memberikan peluang bisnis kepada masyarakat dan menjadikan Science Group sebagai lembaga pendidikan terbaik di tingkat nasional. Berikut dibawah ini adalah program belajar yang diselenggarakan oleh Science-Group Indonesia.

1. **Bimbingan Belajar.**
   Bimbingan belajar ini meliputi les prifat ataupun reguler untuk mata pelajaran Matematika, Fisika, dan Kimia untuk tingkat pendidikan SMA, SMP, dan SD.

2. HOME-Schooling.

HOME-Schooling adalah program khusus yang dimiliki oleh Science-Group Indonesia. Kegiatan belajar-mengajar yang dilaksanakan secara kolektif dengan jumlah siswa maksimal 10 orang. Kurikulum yang digunakan mengacu kepada kurikulum nasional yang dikeluarkan oleh pemerintah, dengan peminatan khusus. Program ini teraffiliasi dengan Yayasan St. Aloysius Bandung. Siswa SMP HOME-Schooling akan menjadi siswa yang terdaftar di SMP St. Aloysius I Bandung, sehingga jika lulus SMP akan mendapatkan Ijazah SMA yang dikeluarkan oleh pemerintah.

3. Language Center.

Program bimbingan belajar bahasa inggris secara reguler ataupun prifat.

4. Science Club.

Program bimbingan belajar khusus yang dimiliki oleh Science-Group Indonesia yang mengajarkan mata-pelajaran Fisika dan Kimia melalui praktek eksperimen kepada siswa.

5. Science Robotic

Program belajar khusus siswa SMP yang dimiliki oleh Science-Group Indonesia, yang mengajarkan proses perakitan serta perancangan robot sederhana untuk siswa SMP.

2.1.2 Visi dan Misi

1. **Visi**

i. Membangun sumber daya manusia Indonesia yang berkualitas dan memiliki kompetensi.

ii. Menjadi merek nasional dan pemimpin bisnis bidang pendidikan di Indonesia.

2. **Misi**

1. Mengembangkan kurikulum dan menyelenggarakan Program-program Pendidikan:
   i. Setara Formal.
   ii. Ilmu Pengetahuan Alam (IPA/Science).
   iii. Teknologi (Technology).
   iv. Ilmu Pengetahuan Sosial (IPS).

v. Bahasa (Language).

2. Mengembangkan kurikulum pembelajaran berstandar nasional dan internasional.
3. Menyelenggarakan program-program pendidikan unggulan.
4. Mengembangkan jaringan nasional bisnis pendidikan dengan sistem kemitraan dan waralaba.
5. Pendukung percepatan belajar bagi siswa-siswa berbakat.
6. Membuka kantor wilayah dan gerai-gerai mitra usaha di Ibukota Propinsi dan kota-kota besar di seluruh Indonesia.

2.1.3 Logo Instansi.



Gambar 2.1 Logo Perusahaan

1. Logo perusahaan Science-Group Indonesia terdiri dari 2 (dua) kalimat, atau 4 (empat) kata, yaitu kalimat ***Science Group*** dengan simbol pensil yang menggantikan huruf (i) dalam kalimat tersebut, serta kalimat ***Edu – Center.***

2. Jenis huruf yang digunakan untuk kalimat ***Science Group*** adalah Tahoma dengan warna gradasi oranye tua – oranye muda. Hal ini menggambarkan komitmen yang tegas dan mendalam yang dimiliki oleh Science Group Indonesia untuk berperan aktif dalam kemajuan pendidikan di Indonesia.

3. Simbol pensil berwarna oranye yang menghadap keatas dengan mata pensil berwarna merah serta penghapus berwarna abu-abu, menggambarkan perkembangan kualitas pendidikan di Indonesia yang selalu meningkat.

4. Kalimat ***Edu-center*** menggambarkan visi perusahaan untuk menjadi salah satu pusat pendidikan terbesar di Indonesia.

Logo perusahaan Science-Group Indonesia melambangkan komitmen Science-Group Indonesia dalam meningkatkan kualitas pendidikan Indonesia menuju ke arah yang lebih baik, dengan partisipasi aktif dari segenap komponen perusahaan melalui dasar yang mendalam yaitu visi dan misi perusahaan.

2.1.4 Badan Hukum Instansi

Sebagai salah satu penyelenggara pendidikan di Kota Bandung yang diawasi oleh Dinas Pendidikan Pemerintah Kota Bandung, Science-Group Indonesia memiliki surat hukum sebagai berikut:

1. **Surat Keputusan Nomor: 421.10/2184-PNFI/2010.**

   Perihal Izin Menyelenggarakan Pendidikan Non Formal dan Informal yang Diselenggarakan Masyarakat.

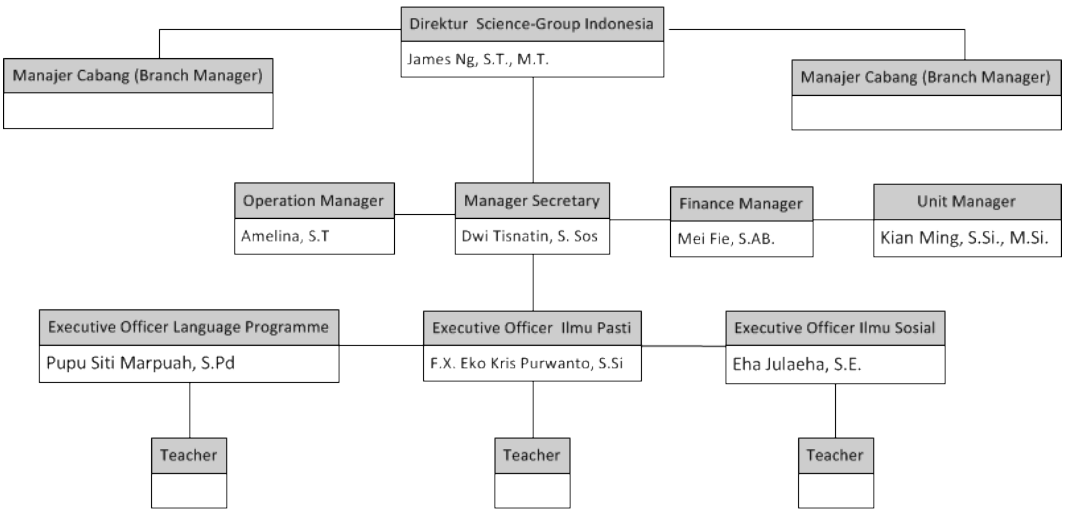
2. **Surat Keterangan Terdaftar Nomor: 421.10/234-PNFI/2011**

   Perihal Ijin Operasional Pusat Kegiatan Belajar Masyarakat (PKBM).

2.1.5 Struktur Organisasi

Dalam mewujudkan visi dan misi perusahaan, Science-Group Indonesia didukung oleh sumber daya manusia yang berkompetensi di bidangnya. Berikut adalah struktur organisasi di dalam Science Group Indonesia.

Berikut adalah struktur perusahaan Science Group Indonesia.



Gambar 2.2 Struktur Organisasi Perusahaan

Setiap komponen perusahaan memiliki tanggung-jawab sebagai berikut:

a) **Direktur**

Bertanggung jawab untuk mengawasi dan mengembangkan kegiatan operasional perusahaan, sehingga dapat mencapai visi dan misi perusahan.

b) **Manajer Cabang** (***Branch Manager***)

Bertanggung-jawab untuk mengawasi dan mengembangkan kegiatan operasional kantor cabang, sehingga dapat memberikan kontribusi yang baik bagi perusahaan.

c) **Manajer Sekretaris (*Manager Secretary)***

Mengkoordinasikan kegiatan operasional masing-masing komponen perusahaan. Selain itu Manajer Sekretaris bertugas untuk memberi laporan perusahaan secara periodik kepada Direktur.

d) **Manajer Keuangan (*Finance Manager*)**

Mengatur dan mengolah keuangan perusahaan secara efisien, serta melaporkan keuangan perusahaan secara periodik kepada Direktur.

e) **Manajer Operasional (*Operation Manager*)**

Manajer Operasional bertanggung-jawab dalam ketersediaan alat tulis kantor (ATK) berupa spidol, pulpen, kertas, serta melakukan duplikasi surat atau dokumen mata pelajaran di Science-Group Indonesia.

f) **Manajer Unit (*Unit Manager*)**

Manajer Unit bertanggung-jawab dalam pemeliharaan perangkat mekanis dan elektronik di Science-Group Indonesia, seperti perangkat komputer, printer, serta perangkat praktikum mata pelajaran fisika dan kimia.

g) **Executive Officer**

Executive Officer bertugas mengatur para Guru mata pelajaran dan bertanggung-jawab dalam pelaksanaan kegiatan belajar-mengajar di Science Group Indonesia.

h) **Guru (Teacher)**

Bertanggung-jawab akan kelas les yang diambil, dan mengajar dengan kurikulum serta jadwal di Science-Group Indonesia.

2.2 LANDASAN TEORI

2.2.1 Konsep Dasar Sistem

Sistem merupakan kumpulan dari user/elemen-elemen yang saling berinteraksi dan saling mempengaruhi dalam melakukan kegiatan bersama untuk mencapai suatu tujuan.

Contoh :

1. Sistem komputer terdiri dari : *Software*,*Hardware*,*Brainware*.
2. Sistem akuntansi, perbankan, kesehatan, dll

a) **Menurut Jerry FithGerald**

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

b) **Menurut Ludwig Von Bartalanfy**

"Sistem merupakan seperangkat unsur yang saling terkait dalam suatu antar relasi diantara unsur-unsur tersebut dengan lingkungan."

c) **Menurut Anatol Raporot**

"Sistem adalah suatu kumpulan kesatuan dan perangkat hubungan satu sama lain."

d) **Menurut L. Ackof**

"Sistem adalah setiap kesatuan secara konseptual atau fisik yang terdiri dari bagian-bagian dalam keadaan saling tergantung satu sama lainnya."

A. **Syarat Sistem**

a. Sistem harus dibentuk untuk menyelesaikan tujuan.

b. Elemen sistem harus mempunyai rencana yang ditetapkan.

c. Adanya hubungan diantara elemen sistem.

d. Unsur dasar dari proses (arus informasi, energi dan materal) lebih penting dari pada elemen sistem.

e. Tujuan organisasi lebih penting dari pada tujuan elemen.

B. **Karakteristik Sistem**

a. Komponen

Suatu sistem terdiri dari sejumlah komponen yang berinteraksi, bekerja sama membentuk suatu kesatuan. Komponen-komponen sistem dapat berupa suatu subsistem atau bagian-bagian sistem. Setiap sistem selalu mengandung komponen-komponen atau **subsistem-subsistem**. Setiap subsistem 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 yang disebut **Supra Sistem,** misalnya suatu perusahaan dapat disebut dengan suatu sistem dan indusri yang merupakan sistem lebih besar dapat disebut dengan supra sistem. Kalau dipandang industri sebagai suatu sistem, maka perusahaan dapat disebut sebagai subsistem. Demikian juga bila perusahaan dipandang sebagai suatu sistem, maka sistem akutansi adalah subsistemnya.

b. Batas

Batas sistem merupakan daerah yang membatasi antara suatu sistem dengan sistem yang lainnya atau dengan lingkungan luarnya. Batas sistem ini memungkinkan suatu sistem dipandang sebagai suatu kesatuan, karena dengan batas sistem ini fungsi dan tugas dari subsistem yang satu dengan lainnya berbeda tetapi tetap saling berinteraksi. Batas suatu sistem menunjukkan ruang lingkup (***scope***) dari sistem tersebut.

c. Lingkungan Luar

Segala sesuatu diluar dari batas sistem yang mempengaruhi operasi dari suatu sistem. Lingkungan luar sistem ini dapat bersifat menguntungkan atau merugikan. Lingkungan luar yang menguntungkan harus dipelihara dan dijaga agar tidak hilang pengaruhnya, sedangkan lingkungan luar yang merugikan harus dimusnahkan dikendalikan agar tidak mengganggu operasi sistem.

d. Penghubung *(Interface)*

Merupakan media penghubung antara suatu subsistem dengan subsistem yang lainnya. Untuk membentuk satu kesatuan, sehingga sumber-sumber daya mengalir dari subsistem yang satu ke subsistem yang lainnya. Dengan kata lain dari suatu subsistem akan menjadi input subsistem yang lain.

e. *Input*

Merupakan energi yang dimasukan kedalam sistem. Masukan dapat berupa *Maintenance Input*, yaitu **energi yang dimasukan supaya sistem tersebut dapat beroperasi**. *Signal Input* adalah energi yang diproses untuk didapatkan keluaran.

Sebagai contoh didalam sistem komputer, program adalah *maintenance input* yang digunakan untuk mengoperasikan komputernya dan data adalah *signal input* untuk diolah menjadi informasi.

f. Output

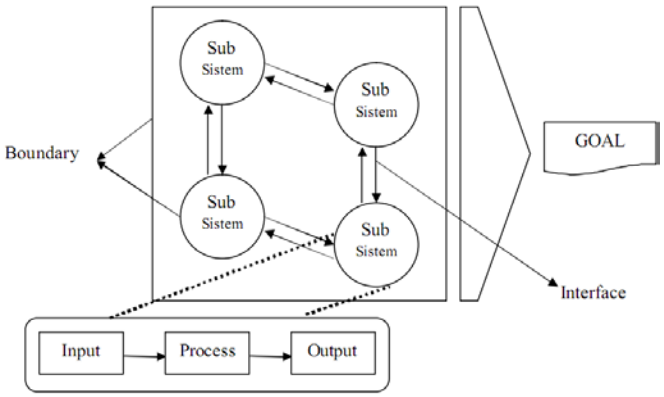
Merupakan hasil dari energy yang diolah oleh sistem. Meliputi : keluaran yang berguna, contohnya Informasi yang dikeluarkan oleh komputer. Dan keluaran yang tidak berguna yang dikenal sebagai sisa pembuangan, contohnya panas yang dikeluarkan komputer.

g. Process

Merupakan bagian yang memproses masukan untuk menjadi keluaran yang diinginkan. Contoh CPU pada Komputer, Bagian Produksi yang mengubah bahan baku menjadi barang jadi, Bagian akuntansi yang mengolah data transaksi menjadi laporan keuangan.

h. Goal

Setiap sistem pasti mempunyai tujuan ataupun sasaran yang mempengaruhi input yang dibutuhkan dan output yang dihasilkan. Dengan kata lain suatu sistem akan dikatakan berhasil kalau pengoperasian sistem itu mengenai sasaran atau tujuannya. Sistem yang tidak mempunyai sasaran, maka operasi sistem tidak akan berjalan dengan baik.



Gambar 2.3 Karakteristik Suatu Sistem

2.2.2 Konsep Dasar Informasi

a) **Informasi :** Data yang telah diproses menjadi bentuk yang memiliki arti bagi penerima dan berupa fakta, suatu nilai yang bermanfaat. Jadi ada suatu proses transfirmasi data menjadi suatu informasi → input-proses-output.
b) **Data :** Merupakan *raw* material untuk suatu informasi. Perbedaan informasi dan data sangat relative tergantung pada nilai gunanya bagi manajemen yang memerlukan. Suatu informasi bagi level manajemen tertentu bisa menjadi data bagi manajemen diatasnya, atau sebaliknya.
c) **Representasi Informasi :** Pelambangan informasi, misalnya: Representasi biner.
d) **Kuantitas Informasi :** Suatu ukuran informasi. Tergantung representasi. Untuk representasi biner satuannya: bit, byte, word dll.
e) **Kualitas Informasi :** bias terhadap error, karena: kesalahan cara pengukuran dan pengumpulan, kegagalan mengikuti prosedur pemrosesan, kehilangan atau data tidak terproses, kesalahan perekaman atau koreksi data, kesalahan file histori/master, kesalahan prosedur pemrosesan ketidak berfungsian sistem.
f) **Umur Informasi :** kapan atau sampai kapan sebuah informasi memiliki nilai/arti bagi penggunanya. Ada condition information (mengacu pada titik waktu tertentu) dan operating information (menyatakan suatu perubahan pada suatu range waktu).
g) **Kualitas Informasi,** tergantung dari 3 hal, yaitu informasi harus :

1. Akurat, berarti informasi harus bebas dari kesalahan-kesalahan dan tidak bias atau menyesatkan. Akurat berarti juga harus jelas mencerminkan maksudnya.
2. Tetap pada waktunya, berarti informasi yang datang pada penerima tidak boleh terlambat.
3. Relevan, berarti informasi tersebut mempunyai manfaat untuk pemakainya. Relevansi informasi untuk tiap-tiap orang satu dengan yang lainnya berbeda.

h) **Nilai Informasi :** ditentukan dari dua hal, yaitu manfaat dan biaya memandapatkannya. Suatu informasi dikatakan bernilai bila manfaat lebih efektif debanding biaya mendapatkannya. Pengukuran nilai informasi biasanya dihubungkan dengan analsis cost effectiveness atau cost benefit.

2.2.3 Unified Modeling Language

Unified Modeling Language (UML) adalah bahasa standar untuk melakukan spesifikasi, visualisasi, konstruksi, dan dokumentasi dari komponen-komponen perangkat lunak, dan digunakan untuk pemodelan bisnis dan sistem perangkat lunak lainnya.

Dengan menggunakan UML dapat memodelkan aplikasi piranti lunak yang dapat berjalan pada piranti keras, sistem operasi dan jaringan apapun, serta ditulis dalam bahasa pemrograman apapun. Namun, UML lebih cocok digunakan untuk penulisan piranti lunak dalam bahasa-bahasa berorientasi objek seperti C++, Java, C# atau VB.NE, sebab UML menggunakan konsep *class* dan *operation*. Walaupun demikian, UML tetap dapat digunakan untuk modeling aplikasi prosedural dalam VB atau C.

Seperti bahasa-bahasa lainnya, UML mendefinisikan notasi dan syntax/semantik. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML syntax mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan.

Tipe diagram UML dibagi menjadi 2, yaitu:

a. *Structural diagrams*
b. *Behavior diagrams*.

I. Structural Diagrams

Structural diagrams digunakan untuk mendeskripsikan relasi antar kelas, sedangkan behavior diagrams digunakan untuk mendeskripsikan interaksi antara aktor dan sebuah *use case* (bagaimana seorang aktor menggunakan sistem).

Structural diagram memiliki 4 diagram, yaitu:

1. Class Diagram

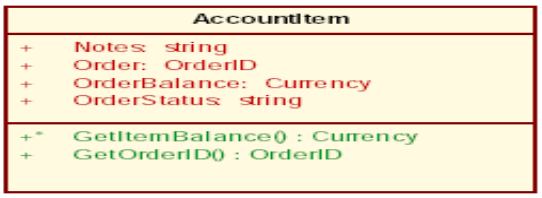
Class adalah sebuah spesifikasi yang jika diinstansiasi akan menghasilkan sebuah objek dan merupakan inti dari pengembangan dan desain berorientasi objek. *Class* menggambarkan keadaan (atribut/properti) suatu sistem, sekaligus menawarkan layanan untuk memanipulasi keadaan tersebut (metoda/fungsi). *Class diagram* menggambarkan struktur dan deskripsi *class*, *package* dan *objek*

beserta hubungan satu sama lain seperti containment, pewarisan, asosiasi, dan lain-lain. *Class* memiliki tiga area pokok:

1. Nama(dan *stereotype*)
2. Atribut
3. Metoda

Atribut dan metoda dapat memiliki salah satu sifat berikut :

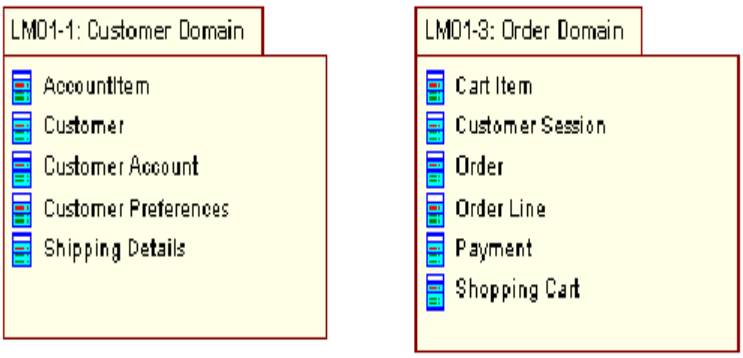
1. *Private* , tidak dapat dipanggil dari luar class yang bersangkutan.
2. *Protected* , hanya dapat dipanggil oleh class yang bersangkutan dan anak-anak yang mewarisinya.
3. *Public* , dapat dipanggil oleh siapa saja.



Gambar 2.4 Struktur Class Diagram

Class dapat merupakan implementasi dari sebuah *interface* , yaitu *class* abstrak *yang* hanya memiliki metoda. *Interface* tidak dapat langsung diinstansiasikan, tetapi harus diimplementasikan dahulu menjadi sebuah *class.* Dengan demikian *interface* mendukung resolusi metoda pada saat *run-time.*

Sesuai dengan perkembangan *class* model, *class* dapat dikelompokkan menjadi *package* . Kita juga dapat membuat diagram yang terdiri atas *package*.



Gambar 2.5 Contoh Class Package Diagram

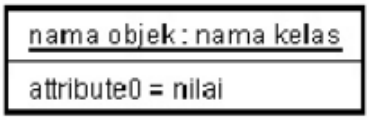
2. Object Diagram

Diagram obyek ialah merupakan suatu diagram yang mengambarkan objek-objek dalam suatu system, dalam kejadian nyatanya. Bila pada kelas diagram terdapat banyak relasi antar kelas. Diagram obyek menggambarkan

representasi dari diagram kelas dalam sistem. Selain itu, Diagram obyek menggambarkan gambaran keadaan sistem dalam waktu tertentu.

Pada diagram objek, hanya terdapat satu relasi antar kelas, yaitu asosiasi. Tidak hanya itu, pada diagram objek yang didefinisikan hanya terdapat nama objek suatu kelas beserta atributnya saja, tanpa operasi yang dilakukan oleh objek tersebut.

Contoh penulisan diagram objek :

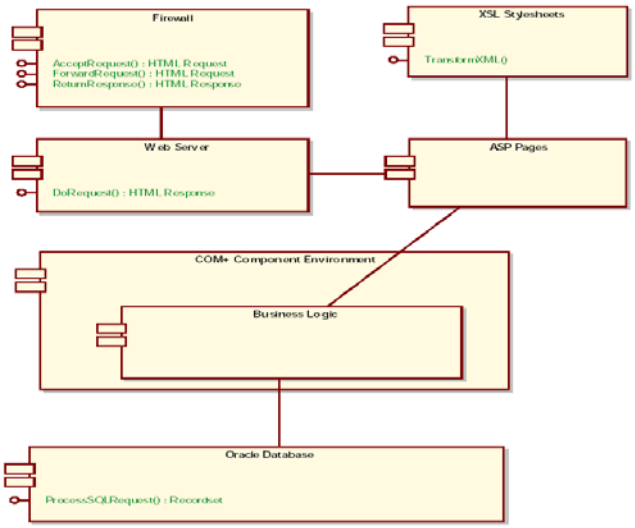


Gambar 2.6 Struktur Diagram Objek

Penulisan nama pada *object diagram*, bisa kita tulis nama kelasnya saja, tanpa diikut nama obyeknya. Dengan penulisan nama kelas awalnya harus ditulis dengan huruf kapital.

3. Component Diagram

Component diagram menggambarkan struktur dan hubungan antar komponen piranti lunak, termasuk ketergantungan (*dependency*) di antaranya. Komponen piranti lunak adalah modul berisi *code*, baik berisi *source code* maupun *binary code*, baik *library* maupun *executable*, baik yang muncul pada *compile time*, *link time*, maupun *run time*. Umumnya komponen terbentuk dari beberapa *class* dan/atau *package*, tapi dapat juga dari komponen-komponen yang lebih kecil. Komponen dapat juga berupa *interface*, yaitu kumpulan layanan yang disediakan sebuah komponen untuk komponen lain. Contoh dari component diagram dapat dilihat di bawah ini.

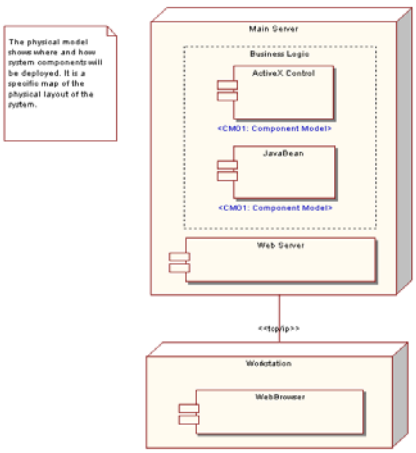


Gambar 2.7 Contoh *Component Diagram*

II. Deployment Diagram

Deployment/physical diagram menggambarkan detail bagaimana komponen di-*deploy* dalam infrastruktur sistem, di mana komponen akan terletak (pada mesin, server atau piranti keras yang digunakan). Hubungan antar *node* (misalnya TCP/IP) dan *requirement* dapat juga didefinisikan dalam diagram ini.

Contoh *deployment diagram adalah sebagai berikut*:



Gambar 2.8 Contoh *Deployment Diagram*

Behavior Diagram memiliki 5 diagram, yaitu:

1. Use Case Diagram

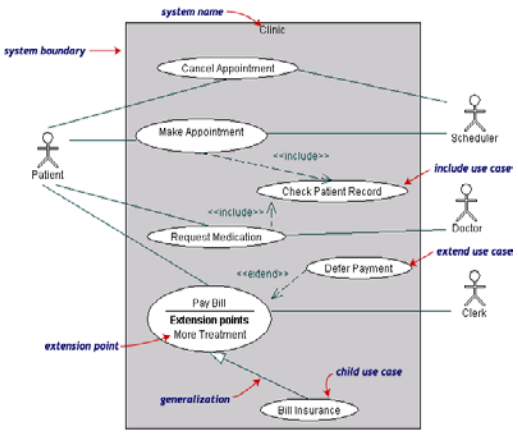
Use case diagram menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor

dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke sistem, membuat (*create)* sebuah daftar belanja, dan sebagainya.

Seorang/sebuah aktor adalah ***sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu***. *Use case diagram* dapat sangat membantu bila kita sedang menyusun *requirement* sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua *feature* yang ada pada sistem. Sebuah *use case* dapat meng-*include* fungsionalitas *use case* lain sebagai bagian dari proses dalam dirinya.

Secara umum diasumsikan bahwa *use case* yang di-*include* akan dipanggil setiap kali *use case* yang meng-*include* dieksekusi secara normal. Sebuah *use case* dapat di-*include* oleh lebih dari satu *use case* lain, sehingga duplikasi fungsionalitas dapat dihindari dengan cara menarik keluar fungsionalitas yang *common*. Sebuah *use case* juga dapat meng-*extend use case* lain dengan *behaviour*-nya sendiri. Sementara hubungan generalisasi antar *use case* menunjukkan bahwa *use case* yang satu merupakan spesialisasi dari yang lain.

Contoh use case diagram :



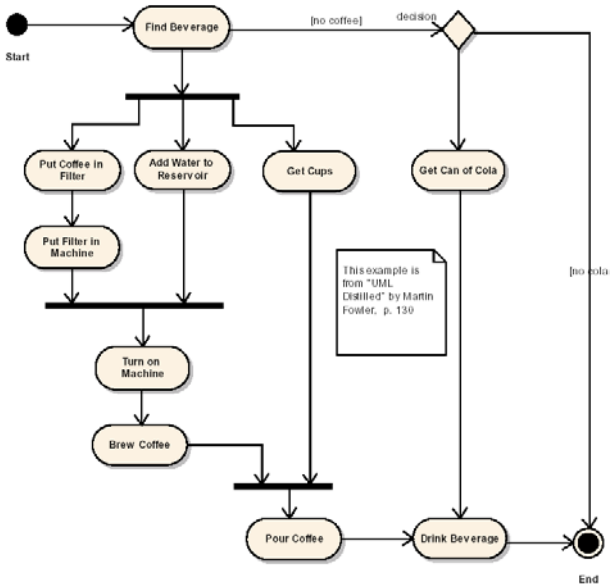
Gambar 2.9 Contoh *Use-Case Diagram*

2. Activity Diagram

Activity diagrams menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity diagram* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. *Activity diagram* merupakan *state diagram* khusus, di mana sebagian besar *state*

adalah *action* dan sebagian besar transisi di-*trigger* oleh selesainya *state* sebelumnya (*internal processing*).

Contoh *activity* diagram :

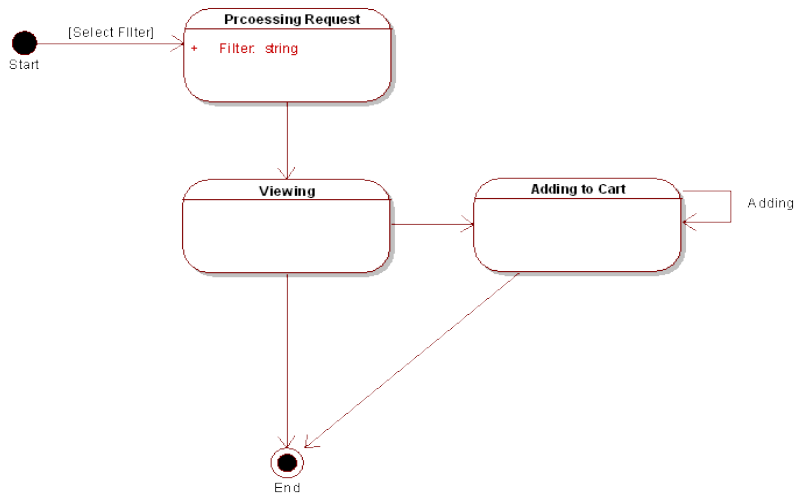


Gambar 2.10 Contoh *Activity Diagram*

3. State Chart Diagram

Statechart diagram menggambarkan transisi dan perubahan keadaan (dari satu *state* ke *state* lainnya) suatu objek pada sistem sebagai akibat dari *stimuli* yang diterima. Pada umumnya *statechart diagram* menggambarkan *class* tertentu (satu *class* dapat memiliki lebih dari satu *statechart diagram*).

Contoh *statechart diagram:*



:

Gambar 2.11 Contoh Statechart Diagram

4. Collaboration Diagram

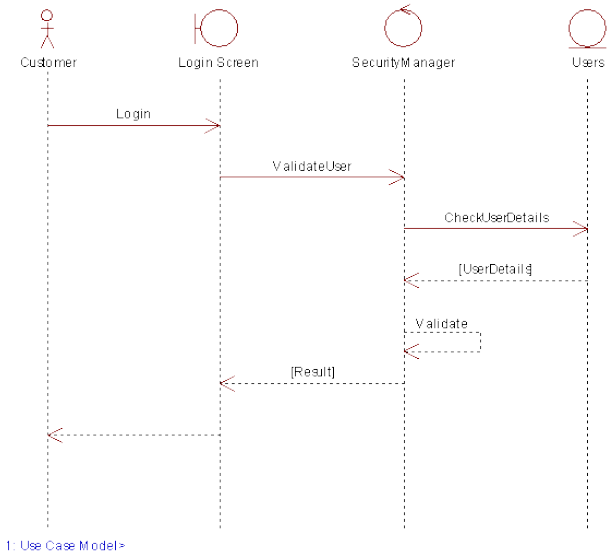
Collaboration diagram juga menggambarkan interaksi antar objek seperti *sequence diagram*, tetapi lebih menekankan pada peran masing-masing objek dan bukan pada waktu penyampaian *message* . Setiap *message* memiliki *sequence number*, di mana *message* dari level tertinggi memiliki nomor 1. Messages dari level yang sama memiliki prefiks yang sama.

5. Sequence Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display* , dan sebagainya) berupa *message* yang digambarkan terhadap waktu. *Sequence diagram* biasa digunakan untuk menggambarkan skenario atau rangkaian langkah-langkah yang dilakukan sebagai respons dari sebuah *event* untuk menghasilkan *output* tertentu.

Untuk objek-objek yang memiliki sifat khusus, standar UML mendefinisikan ikon khusus untuk objek *boundary, controller* dan *persistent entity*.

Contoh sequence diagram :



Gambar 2.12 Contoh Sequence Diagram

2.2.4 Basis Data

Basis data adalah suatu kumpulan data yang saling berhubungan antara satu dengan yang lainnya, yang tersimpan dalam perangkat keras komputer (*hardware*) dan digunakan perangkat lunak (*software*) untuk memanipulasinya. Penerapan *database* didalam sistem informasi disebut database sistem.

Sistem data adalah suatu sistem informasi yang mengintegrasikan kumpulan dari data yang saling berhubungan satu dengan yang lainnya dan membuat tersedia untuk beberapa aplikasi yang bermacam-macam didalam suatu organisasi.

Oleh sebab itu basis data dapat didefinisikan menjadi :

"Basis data merupakan data tertentu yang secara bersama-sama dapat digunakan untuk aplikasi-aplikasi yang berbeda." Sebelum membentuk suatu *database*, data melalui tingkatan dalam struktur data yaitu, mulai dari karakter-karakter, data *item* (*field*), *records* dan kemudian *database*.

1. Karakter

   Karakter merupakan bagian data yang terkecil, karakter dapat berupa angka, huruf, abjad atau symbol-simbol khusus yang membentuk suatu data item.

2. Field

   Sebuah *field* menggambarkan satu atribut dari *record* yang menunjukan suatu item dari data tertentu.

3. Records

   Records merupakan kumpulan dari *field* dan *record* menggambarkan suatu unit dan individu yang tertentu.

4. File

   File terdiri dari *record-record* yang menggambarkan satu kesatuan data yang sejenis.

5. Database

   Database terdiri dari kumpulan *file*. Tujuan dari penggunaan database diantaranya :

   1. Kecepatan dan Kemudahan (*speed*)
   2. Efisiensi Ruang Penyimpanan (*space*)
   3. Keakuratan (*accuracy*)

4. Ketersediaan (*availability*)
5. Kelengkapan (*completeness*)
6. Keamanan (*security*)
7. Kebersamaan Pemakaian (*sharability*)

2.3 Software Pendukung

2.3.1 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.

Salah satu software yang biasa digunakan oleh banyak web master di dunia adalah Apache. Software tersebut dapat didownload secara gratis dari web resmi apache, yaitu http://www.apache.org. Web server ini fleksibel terhadap berbagai sistem operasi seperti Windows 9x/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.

2.3.2 Sejarah PHP

Kelahiran PHP bermula saat Rasmus Lerdorf membuat sejumlah skrip Perl yang dapat mengamati siapa saja yang melihat-lihat daftar riwayat hidupnya yakni pada tahun 1994. Skrip-skrip ini selanjutnya dikemas menjadi tool yang disebut dengan “Personal Home Page”. Sedangkan menurut dokumen resmi PHP, PHP merupakan singkatan dari PHP Hypertext Prepocessor. Ia merupakan bahasa berbentuk skrip yang ditempatkan dalam server dan diproses diserver. Hasilnyalah yang dikirimkan ke klien, tempat pemakai menggunakan browser .

Pada saat ini PHP sudah cukup popular sebagai peranti pemrograman web. PHP bersifat bebas dipakai, sehingga tidak perlu membayar apapun untuk menggunakan perangkat lunak ini. Jika ingin mencoba tidak perlu menggunakan komputer berkelas server. Hanya dengan sebuah komputer biasa, kita bisa mempelajari dan mempraktekan PHP. Konsep kerja PHP diawali dengan permintaan suatu halaman web oleh browse.

Berdasarkan URL (Uniform Resource Locator) atau dikenal dengan sebutan alamat Internet. Browser mendapatkan alamat dari web server, mengidentifikasi halaman yang dikehendaki dan menyampaikan segala informasi yang dibutuhkan oleh web server. Selanjutnya web server akan mencarikan file yang diminta dam memberikan isinya ke web browser. Kemudian diterjemahkan kedalam kode HTML dan menampilkannya ke layar pengguna. Adapun kelebihan-kelebihan dari PHP adalah:

1. PHP mudah dibuat dan memiliki kecepatan akses tinggi.
2. PHP dapat berjalan dalam web server yang berbeda dan dalam sistem operasi yang berbeda pula.
3. PHP diterbitkan secara gratis.
4. PHP adalah termasuk bahasa yang embedded (bisa ditempel atau diletakkan dalam tag HTML).
5. PHP termasuk server-side programming.

2.3.3 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 [Ignore] 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

2.3.4 Cascading Style Sheet (CSS)

Cascading Style Sheet (CSS) merupakan salah satu bahasa pemrograman web untuk mengendalikan beberapa komponen dalam sebuah web sehingga akan lebih terstruktur dan seragam. Sama halnya style dalam aplikasi pengolahan kata pada Microsoft Word yang dapat mengatur beberapa style, misalnya heading, bodytext, footer, dan lainnya untuk dapat digunakan bersama-sama dalam beberapa berkas.

Pada umumnya CSS dipakai untuk memformat tampilan halaman web yang dibuat dengan bahasa HTML dan XHTML. CSS dapat mengendalikan ukuran gambar, warna bagian tubuh pada text, warna, ukuran, paragraf, dan parameter lainnya. CSS juga bahasa style sheet yang digunakan untuk mengatur tampilan halaman web dengan format yang berbeda.

CSS merupakan salah satu fasilitas Adobe Dreamweaver CS 6 yang mendukung HTML dalam mendesain tampilan suatu web. Perintah-perintah yang sering digunakan dapat dikumpulkan dalam CSS sehingga dapat digunakan

kembali dalam mendesain web sehingga tidak perlu mengulang perintah dari awal. Pada panel CSS kita bisa membuat, mengedit, atau menampung seluruh perintah dari CSS.

2.3.5 Framework Code Igniter

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* hanya 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.

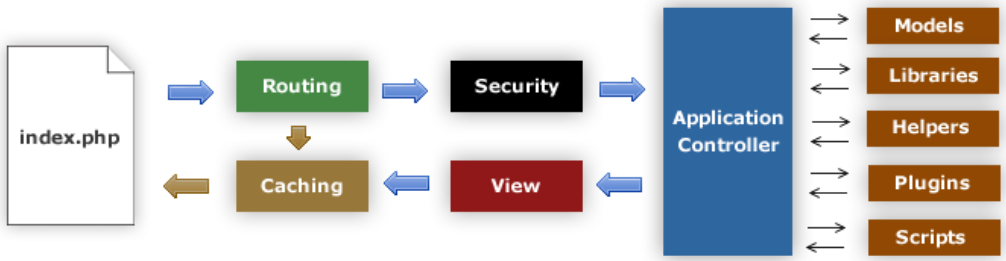
Berikut fitur-fitur yang didukung oleh CodeIgniter :

1. Sistem berbasis *Model View Controller*
2. Kompatibel dengan PHP versi 4.
3. Ringan dan Cepat.
4. Terdapat dukungan untuk berbagai basis data.
5. Mendukung Active Record *Database*, *form* dan validasi data masukan, pencatatan *error* yang terjadi, lokalisasi bahasa.
6. Tersedia pengaturan session, pustaka untuk bekerja dengan XMP-RPC.
7. 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.*

8. Mendukung enkripsi data, benchmarking, caching, hooks, ekstensi *class* dan plugin, XSS filtering.
9. Menghasilkan *clean* URL, URI routing yang felksibel.
10. Memiliki *helper* yang sangat banyak jumlahnya

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



Gambar 2.13. 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.