BAB II

LANDASAN TEORI

2.1. *Web Daily Deals*

Web daily deals merupakan salah satu tipe dari web *e-commerce* yang memberikan sebuah penawaran atas suatu produk yang dijual dalam periode waktu tertentu [16]. Member dari penawaran harian ini menerima penawaran secara *online* dan menerima undangan atau *subscribe* dari penyedia layanan melalui email, pos surat, dan jejaring sosial. Model dari bisnis penawaran harian ini memberikan keuntungan bagi *merchant* dan *customer* yang bisa membangun loyalitas terhadap suatu merek dan mendapatkan profit yang besar bagi *merchant*. Sebagian besar penyedia layanan penawaran ini bekerja sama dengan pembisnis lokal dan *reseller online* untuk mengembangkan penawaran-penawaran termasuk menawarkan diskon secara besar-besaran yang diberikan oleh para *merchant* kepada *customer* untuk terhadap produknya. Dengan menggunakan metodologi tersebut, sedikit atau banyaknya penawaran akan dijual berdasarkan periode waktu tertentu.

Pada awalnya periode waktu yang dipakai adalah periode 24 jam, tetapi beberapa penyedia *web daily deals* menawarkan alternatif lain yaitu penawaran dengan jangka waktu panjang yang bertujuan untuk memaksimalkan keuntungan dan mengijinkan *customer* untuk melakukan penawaran yang lebih dari satu dalam satu lokasi penawaran.

Ketika terdapat penawaran diskon yang sudah disepakati oleh pihak penyedia layanan dan *merchant*, penyedia layanan akan mempromosikan penawaran tersebut dengan cara mengirimkan email penawaran kepada *customer*. Sehingga saat *customer* tertarik pada suatu produk maka *customer* akan melakukan pembelian pada produk yang ditawarkan. Bagian terpenting dalam suatu penyedia layanan *daily deals* ini adalah data *customer*, yang akan digunakan untuk menjual produk milik *merchant*.

Setelah semua stok penawaran habis terjual, barulah *customer* melakukan pembayaran terhadap produk tersebut dan mendapatkan sebuah voucher digital yang akan ditebus kepada *merchant*. Voucher ini memiliki batas waktu pemakaian sesuai dengan kesepakatan yang telah dilakukan antara penyedia layanan *daily deals* dengan pihak *merchant*.

2.1.1. Sejarah *Daily Deals*

Web daily deals ini awalnya terinspirasi dari sebuah perusahaan bernama vente-privee.com milik seorang pengusaha bernama Jacques Antoine Granjon yang berasal dari Perancis pada tahun 2001. Perusahaan ini dikembangkan dalam lingkungan *online* untuk penjualan merek-merek dagang dari para membernya dengan harga yang lebih murah dibanding harga normal. Penjualan ini mencangkup pakaian, aksesoris, mainan, jam tangan, peralatan rumah tangga, peralatan olahraga, *gadget*, dan minuman keras.

Dengan meniru model penjualan yang dilakukan oleh vente-privee.com, bisnis serupa juga dilakukan oleh beberapa pengusaha di Amerika Serikat. Konsep

daily deals pertama kali diluncurkan oleh woot.com pada bulan Juli 2004. Pada tahun 2006, banyak industri bisnis *daily deals* yang mencapai kesuksesannya dengan lebih dari 100 penawaran yang ditawarkan. Pada tahun 2008, Groupon memasuki pasar *daily deals* dan menjadi perusahaan *online* tercepat kedua dalam menghasilkan keuntungan miliaran dolar [13].

Penawaran yang dilakukan oleh layanan *daily deals* akan bermunculan di Amerika Serikat dalam tahun mendatang. Menurut sebuah studi yang dirilis oleh BIA/Kelsey, pendapatan kotor dari layanan *daily deals* tersebut tumbuh dari $873juta dan saat ini mencapai $3,9miliar pada tahun 2015 [7]. Dengan memanfaatkan jejaring sosial telah mempercepat pertumbuhan *web daily deals*. Jejaring sosial dimanfaatkan sebagai sarana penawaran promosi yang dilakukan oleh penyedia layanan *web daily deals*. Penawaran demi penawaran disebarkan melalui jejaring sosial akan membuat penawaran tersebut menjadi lebih cepat diketahui oleh seluruh masyarakat.

Tahun 2010 merupakan tahun dimana perkembangan yang cepat terhadap *web daily deals*. Perkiraan beberapa pengamat terhadap industri *daily deals* ini memperkirakan industri *daily deals* akan tumbuh dengan kuat. Seorang analis memperkirakan bahwa pendapatan yang dihasilkan akan mencapai beberapa miliar dolar atau lebih dari 100% peningkatan dari tahun-tahun sebelumnya pada tahun 2011 [4]. Dengan pertumbuhan yang cepat dari *daily deals* ini, beberapa perusahaan internet terbesar juga meluncurkan web sejenis, seperti Google dengan Google Offersnya.

Meskipun angka pertumbuhan dari *daily deals* ini meningkat positif, ada sebuah penelitian yang menyatakan bahwa ada kelemahan struktural dari sebuah *web daily deals* yang harus diperbaiki dimasa yang akan datang. Kekurangan ini datang dari sisi *customer* dan *merchant*. Sebagai contoh, jika dilihat dari sisi *customer*, beberapa dari *customer* tersebut jarang ada yang kembali melakukan pembelian produk yang ada *web daily deals* dengan alasan beragam, dan dari sisi *merchant*, sebagian dari *merchant* menunjukkan ketidaktertarikan melakukan kerja sama dengan pihak penyedia layanan *daily deals* pada kesempatan berikutnya. Dengan alasan seperti ini, industri *daily deals* membutuhkan beberapa perubahan dari *web daily deals* yang sudah ada.

2.1.2. Pemasaran

Kebanyakan pembisnis menganggap promosi yang dilakukan di *web daily deals* bukanlah sarana dalam mendapatkan keuntungan. Padahal jika dilihat dari sisi positif dari bisnis di *web daily deals* ini adalah produk yang dimiliki oleh *merchant* akan dipromosikan kepada sejumlah *customer* dalam waktu singkat. Terdapat bukti dari nilai peningkatan penjualan yang signifikan yang sudah dilakukan oleh *web daily deals*. Salah satu pembisnis melaporkan peningkatan penjualan produk mereka sebesar 140% dibandingkan dengan penjualan mereka sebelumnya yang tanpa dibantu oleh *web daily deals* [4]. Diluar tujuan utama *web daily deals* yang ingin membantu memasarkan produk para *merchant*, *web daily deals* ini juga akan menggeser pemasaran yang dilakukan secara tradisional dan juga jasa periklanan [3].

2.1.3. Alat Berkomunikasi

Salah satu alat pemasaran dari *web daily deals* yang paling menonjol adalah pengiriman email untuk kegiatan promosinya. Pengiriman email promosi seperti ini telah banyak dikritik oleh pengguna email [13]. Namun, ini adalah bukti yang strategis untuk menghasilkan penjualan yang efektif. Pengunjung *web daily deals* bisa mendaftarkan dirinya melalui email *subscribe* dan menjadi *subscriber web daily deals* tersebut. Email akan dikirimkan kepada *subscriber* apabila ada penawaran-penawaran baru yang ada di *web daily deals* tersebut. Sebagian besar *web daily deals* mempublikasikan penawaran mereka melalui RSS *feed*, sehingga siapa saja yang ada bersama RSS *reader* bisa melihat penawaran terakhir dari sebuah *web daily deals*.

2.1.4. Afiliasi

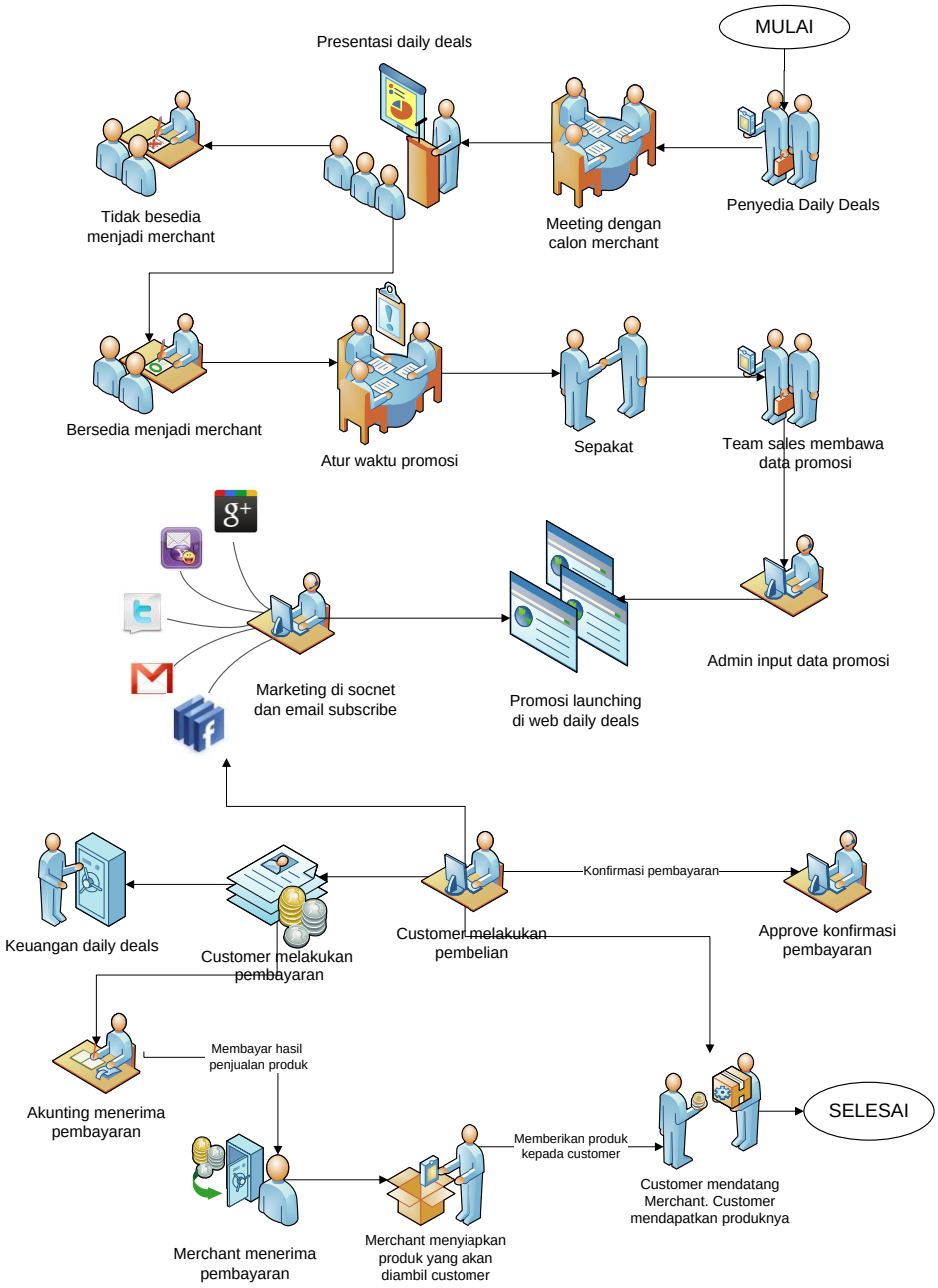
Sebagian besar *web daily deals* menggunakan web afiliasi untuk melakukan promosi terhadap penawaran-penawaran mereka. Web afiliasi ini biasa disebut dengan agregator. Web agregator yang sudah ada misalnya Yipit, web ini menampilkan beberapa penawaran dari sejumlah *web daily deals*. Layanan ini menawarkan email yang disesuaikan dengan lokasi dan kategori yang telah disepakati sebelumnya. Web agregator ini mengambil keuntungan dari setiap penjualan yang dilakukan oleh *web daily deals*. Tetapi tidak semua *web daily deals* menggunakan agregator, contohnya LivingSocial yang tidak berbagi keuntungan dengan web agregator.

2.1.5. Cara Kerja

Pada gambar II.1 akan dipaparkan bagaimana sebuah *web daily deals* bekerja. Penyedia layanan *daily deals* akan mendatangi calon *merchant* untuk melakukan penawaran kerja sama. Penyedia layanan *daily deals* akan melakukan presentasi mengenai aturan melakukan promosi di *web daily deals* dan menjelaskan keuntungan melakukan promosi produk di *web daily deals*.

Jika calon *merchant* tidak bersedia menjalin kerjasama maka pihak penyedia layanan *daily deals* akan mencari calon *merchant* lain. Jika calon *merchant* bersedia menjalin kerjasama dengan penyedia layanan *daily deals*, maka langkah selanjutnya adalah menentukan waktu promosi, besar *fee* yang didapat penyedia layanan *web daily deals*, besar diskon, dan pengumpulan data produk. Setelah mencapai suatu kesepakatan, selanjutnya admin *web daily deals* akan menginputkan data promosi produk, menaikkan promosi tersebut, dan melakukan pemasaran produk tersebut ke jejaring sosial dan mengirimkan email *subscribe* kepada member lainnya.

Ketika pengunjung *web daily deals* melihat promosi tersebut dan tertarik melakukan pembelian, maka pengunjung tersebut melakukan pemesanan produk, melakukan pembayaran, dan melakukan konfirmasi pembayaran. Jika telah melakukan konfirmasi pembayaran, pihak admin akan melakukan *approval* dan mengirimkan voucher digital yang akan di cetak oleh *customer*. Tahap terakhir adalah *customer* mendatangi *merchant* untuk menukarkan voucher dengan produk yang sebelumnya telah dibeli *customer*.



Gambar II.1 – Cara Kerja *Web Daily Deals*

2.2. Konsep Web 2.0

Web 2.0 merupakan revolusi bisnis dari sebuah industri komputer yang terbentuk karena adanya pergerakan internet menjadi sebuah *platform*. Diantara

aturan utama yang dimiliki Web 2.0 adalah membangun sebuah aplikasi yang memanfaatkan efek jaringan untuk membuat lebih banyak orang dapat menggunakan aplikasi tersebut. Beberapa orang berpikir bahwa Web 2.0 adalah sebagai perpaduan dari partisipasi orang-orang terhadap suatu arsitektur web. Ada beberapa pihak juga yang menyebutkan bahwa Web 2.0 adalah generasi pertama dari perkembangan sebuah internet. Sangat sulit untuk mendefinisikan apa itu Web 2.0. Internet berubah sangat cepat, seiring berjalannya waktu pengertian mengenai Web 2.0 akan berubah. Web 2.0 adalah tentang bagaimana lebih dari satu teknologi bekerja pada satu saat, dan juga bagaimana suatu teknologi digunakan dan memiliki efek yang nyata pada sebuah web.

Web 2.0 umumnya merujuk kepada kumpulan sosial, arsitektural, dan pola rancangan yang menghasilkan migrasi bisnis kepada internet yang akan dijadikan *platform*. Pola ini fokus kepada model interaksi antara komunitas, manusia, komputer, dan perangkat lunak. Interaksi manusia merupakan aspek yang sangat penting dari sebuah arsitektur perangkat lunak, dan spesifikasi dari sebuah web adalah untuk menyatukan pengalaman manusia dengan teknologi. Web 2.0 mewakili lebih dari sekedar perubahan pada teknologi internet. Perubahan global yang terjadi adalah terhadap cara pengguna sebuah web berinteraksi satu sama lain.

Web 2.0 memiliki beberapa pola, berikut ini adalah daftar pola Web 2.0 yang diurutkan mulai dari yang paling abstrak sampai yang paling konkret [6] :

1. *Service Oriented Architecture* (SOA)

   SOA adalah sebuah arsitektur, salah satu kerangka kerja untuk pencocokan dan kemampuan. Fitur utama dari SOA adalah dukungan yang mengintegrasi layanan yang dimiliki dan dikelola secara independen. SOA adalah sebuah pola yang mendasari Web 2.0, dan beberapa pola lain seperti *Mashup* dan *Software as a Service*. Contoh dari pola ini adalah aplikasi penawaran saham yang bisa dipanggil oleh konsumen.

2. *Software as a Service* (SaaS)

   SaaS memberikan fungsionalitas komputasi untuk penggunanya tanpa harus memiliki aplikasi tersebut. Ini berlaku untuk SOA yang berada pada bidang perangkat lunak. Contoh dari aplikasi yang memakai pola SaaS adalah web http://createpdf.adobe.com, pada web tersebut pengunjung bisa mengubah sebuah dokumen HTML kedalam bentuk PDF tanpa harus meng-*install* program tersebut pada komputer pengunjung. Biasanya pola SaaS digunakan pada aplikasi berbasis *cloud computing*.

3. *Participation-Collaboration*

   Pola *Participation-Collaboration* berfokus kepada cara mengorganisasi masyarakat dan interaksi sosial diantara pengguna Web 2.0. Pengorganisasian ini mencangkup penggunaan konten web, kontribusi untuk karya dari pengunjung, beta konstan, mempercayai pengunjung sebagai pengguna Web 2.0, dan membuat pengguna menjadi bagian dari arsitektur dan model Web 2.0. Contoh web yang menggunakan pola ini adalah Wikipedia. Karena

pengunjung mempunyai kontribusi dalam pengisian konten Wikipedia. Hal ini juga merupakan sebuah kecerdasan yang dimiliki oleh pengelola layanan.

4. *Asynchronous Particle Update*

   Pola ini ada dibalik AJAX, namun bisa juga dengan menggunakan teknologi lainnya. Daripada melakukan *reload* terhadap satu halaman web secara keseluruhan, lebih baik memperbarui sebagian kecil dari bagian halaman tersebut. Pola ini memiliki beberapa variasi yang dapat memicu proses *update*, termasuk *timeout*, aktivitas pengguna, dan parameter lainnya. Ini dapat menyebabkan sebuah aplikasi yang berbentuk *client-server* seperti *cloud computing*.

5. *Mashup*

   Pola *Mashup* bergantung pada sebuah layanan yang digunakan. Menggabungkan sebuah konten atau sumber daya komputasi dari berbagai sumber, dan pencampuran dari berbagai layanan untuk menciptakan sebuah layanan yang baru. Umumnya ada pada tampilan yang menghasilkan dua atau lebih aplikasi dalam waktu yang bersamaan. Contoh penggunaan pola *Mashup* adalah Google Map dengan memperlihatkan data keuangan diatasnya.

6. *Rich User Experience* (RUE)

   Pola *Rich User Experien*ce identik dengan pola *Rich Internet Applicatio*n (RIA). RUE adalah sebuah replika dari dunia nyata. Pola RUE menggabungkan beberapa aspek termasuk presentasi secara visual, informasi yang relevan, dan aplikasi yang dimodelkan untuk memahami ruang lingkup yang memungkinkan adanya interaksi antara pengguna dan sebuah perangkat

lunak. Contoh dari penerapan RUE ini adalah percakapan *online* yang dilakukan oleh satu pengguna dengan pengguna lainnya.

7. *The Synchronized Web*

   Dalam pola *The Synchronized Web* ini, pengguna dari berbagai negara yang sama atau negara yang berbeda menggunakan aplikasi secara bersamaan. Contoh dari pola ini adalah *game online,* yang bisa dimainkan secara bersamaan secara *online* diseluruh negara.

8. *Collaborative Tagging*

   Collaborative Tagging ini biasanya disebut dengan folksonomi, istilah yang diciptakan oleh Thomas Vander Wall. *Collaborative Tagging* mengacu pada kemampuan pengguna untuk menambahkan sebuah label atau *tag* untuk menghubungkan sumber daya dengan simbol semantik yang bersandar pada sebuah ontologi. Upaya dalam pembuatan web semantik telah gagal dilakukan, namun sementara itu munculnya *Collaborative Tagging* ini telah menambah aspek baru dalam penciptaan lapisan internet. Contoh implementasi dari web *Collaborative Tagging* adalah http://del.icio.us yang dimana pengguna dapat menerapkan label atau *tag* kepada suatu *bookmark* publik yang memiliki bilangan prima.

9. *Declarative Living and Tag Gardening*

   Dalam dunia nyata, orang membuat pernyataan tentang apa saja. *Declarative Living* adalah tindakan pengkodean dalam sebuah sintaksis dimana sebuah mesin dapat memproses, dan membuat kode-kode tersebut dapat terlihat pada sebuah halaman web. Sedangkan *Tag Gardening* adalah tindakan kolektif

yang dilakukan oleh beberapa pengguna. Contoh dari pengguna pola ini adalah Twitter (http://www.twitter.com) dan Facebook (http://www.facebook.com), dimana pengguna dapat membuat deklarasi atau ungkapan tentang kegiatan sehari-hari mereka.

10. *Semantik Web Grounding*

    Pola *Semantik Web Grounding* melakukan pantauan terhadap interaksi yang dilakukan antara sebuah deklarasi dan sumber daya, serta mengetahui bagaimana pengguna berinteraksi berdasarkan aspek tersebut. Pola ini memfasilitasi kemandirian dari sebuah perangkat lunak. Contoh dari pengimplementasian pola ini adalah Google Search.

11. *Persistent Rights Management*

    Persistent Rights Management merupakan sebuah pola dimana mempertahankan pengguna untuk melakukan *Create*, *Read*, *Update*, *Delete* (CRUD). Berbeda dengan hanya mengamankan hasil salinan yang asli, pola ini melimpahkan hak-hak penulis pada semua salinan karyanya. *Adobe LiveCycle Rights Management Server* dan *Rights Management Micros*oft adalah contoh dari pengimplementasian pola ini.

12. *Structured Information*

    Munculnya XML dan kemampuan untuk menerapkan *tagging* disesuaikan untuk elemen tertentu telah menyebabkan munculnya sintaksis yang disebut dengan Microformats. Microformats merupakan sebuah format kecil dengan kemampuan khusus dalam menandai informasi yang tepat dalam sebuah dokumen. Penggunaan format tersebut adalah XHTML yang memungkinkan

konten internet bisa lebih rinci dilakukan daripada menggunakan HTML. Contoh pengimplementasian pola ini adalah XML Friends Network (XFN).

2.3. PHP (*Hypertext Prepocessor*)

Asal usul PHP dimulai pada tahun 1995 ketika pengembang perangkat lunak yang bernama Rasmus Lerdorf mengembangkan skrip Perl/CGI yang dibuat dengan tujuan untuk mengetahui berapa banyak pengunjung yang membaca resume *online* miliknya. Skrip tersebut melakukan dua tugas, yaitu melihat informasi pengunjung dan menampilkan jumlah pengunjung web tersebut. Karena web pada saat itu merupakan hal yang masih baru, perangkat lunak yang dimiliki Lerdorf ini belum pernah diciptakan sebelumnya. Dengan demikian, banyak orang yang tertarik dengan skrip yang diciptakan Lerdorf. Kemudian Lerdorf memberikan nama untuk skrip yang diciptakannya, skrip tersebut diberi nama Personal Home Page (PHP).

Banyak orang yang meminta Lerdorf untuk mengembangkan skrip tersebut, dengan pengembangan pertama, Lerdorf membuat sebuah fitur baru untuk PHP yang disisipkan kedalam skrip HTML dalam bentuk variabel-variabel. Tetapi sayangnya Lerdorf lebih menyukai untuk melakukan pengembangan dibahasa C, bukan pada Perl. Dengan adanya tambahan skrip PHP dari Lerdorf, PHP mulai mengalami pengembangan lagi pada tahun 1997 dengan merilis PHP 2.0 atau Personal Home Page/Form Interpreter (PHP/FI). Rilisnya PHP 2.0 didampingi oleh sejumlah perangkat tambahan dan perbaikan dari *programmer* diseluruh dunia.

Perilisan PHP ini sangat popular, pengembang-pengembang tersebut lalu bergabung bersama Lerdorf. Penggabungan kode asli PHP dengan HTML dan *parsing*, melahirkan PHP 3.0. Dengan dirilisnya PHP 3.0 ini, lebih dari 50.000 pengguna menggunakan PHP untuk mengembangkan web-web mereka.

Pengembangan yang sangat pesat dilakukan terhadap PHP 3.0 dengan melakukan penambahan ratusan fungsi dasar PHP. Pada awal tahun 1999, Netcraft (http://www.netcraft.com) yaitu perusahaan penelitian dan analisa memperkirakan bahwa pada tahun tersebut pengguna PHP berjumlah 1juta pengguna, hal ini membuat PHP menjadi salah satu bahasa pemrograman paling populer sedunia.

Dua orang pengembang yaitu Zeev Suraski dan Andi Gutmans mengambil inisiatif untuk sepenuhnya mengembangkan pengoperasian PHP, puncaknya pengembangan pada PHP tersebut dijuluki Zend Scripting.

Pada bulan Mei 2000, sekitar 18 bulan lebih pengembangan dilakukan maka rilislah PHP 4.0. PHP 4.0 menawarkan kematangan dari sebuah bahasa pemrograman web, PHP 4.0 menyediakan fitur, daya, dan skalabilitas dari sejumlah ahli pengembangan bahasa pemrograman. Tim PHP masih merasa belum puas terhadap PHP 4.0 maka dirilislah PHP 5.0.

PHP 5.0 mempunyai kemampuan melakukan pemrograman berorientasi objek, mencoba mengatasai masalah eksepsi, peningkatan XML dan dukungan layanan web dan mendukung SQLite. *Upgrade* dilakukan untuk PHP 5.0 dengan merilis PHP 5.3 dengan menambahkan fitur array, *driver* MySQL yang baru, dan berbagai penambahan sintaksis seperti NOWDOC.

Sebelum mengenal lebih jauh mengenai PHP ada baiknya jika dasar-dasar PHP dipahami lebih dalam. Pada sub bab ini akan dijelaskan mengenai dasar-dasar PHP yang meliputi sintaksis PHP, memberikan baris komentar, menampilkan data pada *browser*, tipe data, *identifier*, *variable*, konstanta, ekspresi, struktur kontrol [5].

1. Sintaksis PHP

   Sebuah file PHP harus berekstensi .php dan biasanya berisi tag HTML dan beberapa sintaksis PHP. Dibawah ini akan dijelaskan beberapa penulisan sintaksis yang ada pada PHP.

   a. Sintaksis dasar

      Sintaksis standar PHP ini diawali dengan <?php , dan diakhiri dengan ?>, setiap akhir baris kode PHP harus diakhiri dengan titik koma. Titik koma adalah pemisah dan digunakan untuk membedakan satu set instruksi dari yang lain.

```
<html>
<body>
   <h2>Belajar PHP</h2>
   <?php
       echo "<p>Hello world.</p>";
   ?>
</body>
</html>
```

   b. Tag pendek

      Short tag ini memungkinkan penulisan sintaksis standar PHP lebih pendek dari aslinya.

```
<?=
      print "Rani akan belajar PHP.";
?>
```

Fungsi tersebut hampir sama dengan fungsi echo, seperti contoh :

```
<?php
      echo “Ratih ingin belajar PHP juga.”;
?>
```

c. Skrip

Beberapa *editor* tidak bisa secara langsung mengerti sintak PHP. Untuk menanggulangi hal tersebut, PHP memberikan dukungan penulisan sintak PHP dengan pembatas tag <script>, contohnya :

```
<script language=”php”>
      print “Penggunaan tag script untuk PHP.”;
</script>
```

d. ASP *Style*

Halaman dari sebuah Microsoft ASP menggunakan penulisan yang serupa dengan sintak PHP, pembatasan statis dan dinamis menggunakan karakter yang udah ditentukan. Pembukaan sintak diawali dengan <%, dan diakhiri dengan %>. PHP juga bisa menggunakan penulisan sintaksis seperti itu.

```
<%
      print ”Penulisan PHP dengan ASP Style.”;
%>
```

Perlu diingat bahwa penulisan *tag* PHP dengan menggunakan aturan penulisan sintak ASP perlu dihindari, kecuali memang ada alasan yang kuat dalam penggunaan sintaksis tersebut. Pada beberapa *editor*, *tag* seperti ini tidak bisa digunakan.

2. Memberikan baris komentar

PHP menawarkan beberapa sintaksis dimana *programmer* bisa menambahkan catatan kecil untuk menandai sintaksis yang dibuatnya kedalam bentuk

komentar. Sintaksis yang diberikan tanda komentar, tidak akan pernah di eksekusi. Berikut ini adalah jenis-jenis komentar yang ada di PHP.

a. *Single line c++ syntax*

Ada beberapa saat dimana seorang *programmer* ingin melakukan komentar terhadap satu baris saja, untuk mengefisiensikan sintaksis PHP, maka PHP mengijinkan *programmer* untuk melakukan komentar untuk satu baris sintaksis PHP dengan mengimplementasi komentar dari C++, yaitu dengan menggunakan tanda *slash* sebanyak dua kali "//".

```
<?php
      // Title : My First PHP script
      // Author : Ratih Kosaman
      echo "This is PHP program.";
?>
```

b. *Shell syntax*

Selain mendukung penulisan komentar yang mengimplementasikan komentar bawaan C++, PHP juga mendukung penulisan komentar dengan *shell sintaksis*. Yaitu dengan diawalli tanda pagar "#".

```
<?php
      # Title : My first PHP script
      # Author : Ratih Kosaman
      echo "This is another PHP script.";
?>
```

c. *Multiple line syntax*

Penulisan komentar ini mengijinkan *programmer* untuk melakukan komentar terhadap beberapa baris sintaksis yang diawali /* dan diakhiri */.

```
<?php
      /*    Processes PayPal payments
            This script is responsible for processing the
            customer's payment via PayPal.accepting the
            customer's
            credit card information and billing address.
            Copyright 2010 W.J. Gilmore, LLC.
      */
?>
```

3. Menampilkan data pada *browser*

Sebuah web pasti akan menampilkan hasil sintaksis pada sebuah layar *browser*, hal ini juga bisa dilakukan oleh PHP. PHP menawarkan beberapa alternatif untuk menampilkan data pada *browser*. Cara untuk menampilkan data pada *browser* adalah sebagai berikut :

a. Pernyataan print()

Pernyataan print() hanya akan membalikkan satu nilai keluaran saja. Pernyataaan print() memiliki aturan seperti dibawah ini.

```
<?php
      print(“<p>Baju saya berwarna kuning.</p>”);
?>
```

Pernyataan print() tersebut akan menghasilkan keluaran :

```
Baju saya berwarna kuning.
```

b. Pernyataan echo()

Pernyataan echo() memiliki tujuan yang sama dengan print() meskipun ada perbedaan teknis antara echo() dan print(). Echo() memiliki struktur seperti dibawah ini :

```
void echo(string argument1 [, … string argument])
```

Keluaran dari echo() mampu menghasilkan keluaran beberapa nilai string.

```
<?php
      $penyanyi1 = “Melly Goeslaw”;
      $penyanyi2 = “Krisdayanti”;
      echo “$penyanyi1 dan $penyanyi2, adalah penyanyi
      yang terkenal di Indonesia.”;
?>
```

Pernyataan diatas akan mengeluarkan hasil :

```
Melly Goeslaw dan Krisdayanti adalah penyanyi yang
terkenal di Indonesia.
```

c. Pernyataan printf()

Pernyataan printf() sangat cocok jika digunakan untuk menampilkan perpaduan antara teks statis dan informasi dinamis yang disimpan dalam satu atau beberapa variabel. Struktur dari printf() adalah :

```
integer printf(string format [,mixed args])
```

Sebagai contoh, misalkan anda ingin memasukkan nilai bilangan bulat dinamis menjadi string statis atau sebaliknya.

```
printf("Ayam itu bertelur sebanyak %d buah.",20);
```

Hasil eksekusi dari pernyataan tersebut adalah :

```
Ayam itu bertelur sebanyak 20 buah.
```

d. Pernyataan sprintf()

Pernyataan sprint() secara fungsional identik dengan printf() perbedaannya adalah bahwa keluaran dari telah ditetapkan dalam sebuah string. Sprint() memiliki struktur:

```
string sprintf(string format [,mixed arguments])
```

Contohnya adalah:

```
$cost = sprintf("$%.2f", 43.2); //$cost = 43,20
```

4. Tipe data

Tipe data adalah sebuah nama generik yang ditugaskan untuk memisahkan jenis data yang terdiri dari beberapa karakteristik. Umumnya tipe data terdiri dari boolean, integer, float, string, dan array. Tipe data ini dibagi menjadi dua bentuk, yaitu tipe data berbentuk skalar dan berbentuk *compound*.

a. Tipe data skalar

Tipe data skalar digunakan untuk mewakili nilai tunggal. Beberapa tipe data yang masuk dalam kategori ini adalah boolean, integer, float, dan string.

b. Tipe data *compound*

Tipe data *compound* memungkinkan beberapa item data yang sejenis dapat dikumpulkan dalam satu perwakilan entitas. Tipe data yang masuk dalam kategori *compound* adalah array dan objek.

5. *Identifier*

Identifier adalah istilah umum yang diterapkan untuk variabel, fungsi, dan beberapa pendefinisian objek. Beberapa aturan yang harus dipatuhi mengenai identifier ini adalah :

a. *Identifier* bisa terdiri dari lebih satu karakter dan harus dimulai dengan huruf atau *underscore*. Selanjutnya pendefinisian dapat terdiri dari huruf, angka, karakter yang digarisbawahi, dan karakter ASCII dari 127 sampai 255 karakter.

b. *Identifier* itu *case sensitive*, oleh karena itu variabel yang bernama $resep, akan berbeda arti dengan variabel $reSep, $Resep, ataupun $REsep.

c. *Identifier* dapat ditulis panjang. Hal ini menguntungkan karena memungkinkan *programmer* untuk secara akurat menggambarkan tujuan identifier dalam mengenali sebuah nama.

d. Sebuah *identifier* tidak dapat identik untuk setiap kunci standar PHP.

6. Variabel

Variabel adalah sebuah simbol yang dapat menyimpan nilai yang berbeda pada waktu yang berbeda pula.

a. Pendeklarasian variabel

Dalam mendeklarasikan sebuah variabel, dimulai dengan menulis tanda dolar $, yang kemudian diikuti oleh nama variabel. Nama variabel dapat dimulai dengan huruf atau garis bawah, karakter ASCII 127 sampai 255 karakter. Contoh penulisan variabel yang valid adalah $color, $model_, $operating_system, atau $_schema. Variabel itu *case sensitive*, jadi bila ada variabel yang bernama $color, akan berbeda arti dengan variabel yang bernama $coLor, $COLOR, atau $Color.

b. Lingkup variabel

Dalam mendeklarasikan sebuah variabel bisa dilakukan dimana saja pada script PHP. Lokasi deklarasi tersebut sangat berpengaruh terhadap sintaksis lainnya. lingkup variabel PHP dapat dibagi menjadi empat yaitu lokal variabel, fungsi parameter, global parameter, dan variabel statis.

7. Ekspresi

Ekspresi merupakan sebuah frasa yang mengekspresikan keterangan aksi dalam suatu program. Ekspresi terdiri dari *operand* dan *operator*. *Operand* merupakan masukan dari sebuah ekspresi. Perhatikan contoh dibawah ini untuk mengetahui apa itu *operand*.

```
$sum = $var1 + $var2; //$sum, $var1, $var2 adalah operand
$a++; //$a adalah operand
```

Sedangkan operator adalah simbol yang menspesifikasikan keterangan aksi dalam sebuah ekspresi. Contoh operasi adalah < , > , ( ) , [ ] , / , % , + , = , && , || , dll.

8. Struktur kontrol

   Struktur kontrol menerangkan alur kode yang terdapat dalam sebuah aplikasi. Mendefinisikan pengeksekusian karakteristik seperti bagaimana dan berapa kali kode tersebut di eksekusi.

   a. Pernyataan kondisi

      Pernyataan kondisi memungkinkan sebuah program untuk melakukan berbagai macam input, menggunakan bermacam kondisi yang didasari dengan sebuah nilai masukan.

      a) Pernyataan if
      b) Pernyataan else
      c) Pernyataan if-else

   b. Pernyataan perulangan

      Pernyataan perulangan memungkinkan sebuah program untuk melakukan sebuah instruksi yang berulang sampai kondisi yang diinginkan tercapai.

      a) Pernyataan while
      b) Pernyataan do-while
      c) Pernyataan for
      d) Pernyataan foreach

2.4. MySQL

MySQL merupakan sebuah relasional basis data, dimana sebuah tabel *database* relasional adalah sebuah struktur data yang digunakan untuk menyimpan dan mengatur informasi. Ilustrasi ini dapat dilakukan dengan membayangkan tabel sebagai *grid* yang terdiri dari kedua baris dan kolom seperti *spreadsheet*. Misalnya, baris dan kolom tersebut dimaksudkan untuk menyimpan informasi mengenai kontak karyawan, dan tabel ini ada lima kolom yang terdiri dari kolom ID karyawan, nama depan, nama belakang, alamat email, dan nomor telepon. Contoh ini merupakan penggambaran dari tujuan tabel tersebut yaitu untuk menyimpan data karyawan.

Untuk setiap kolom tersebut memiliki tipe data yang berbeda, isi dari kolom ini akan disimpan kedalam sebuah *database*. Setiap kolom tersebut harus menggunakan satu aturan yang telah ditetapkan, misalnya ukuran, tipe data (string, integer, atau desimal) dan format data (format tanggal, hari, tahun). Berikut ini keterangan lebih lengkap mengenai MySQL [5].

1. Jenis penyimpanan basis data MySQL terdiri dari MyISAM, IBMDB21, InnoDB, MEMORY, MERGE, FEDERATED, ARCHIVE, CSV, EXAMPLE, dan BLACKHOLE. Jenis mesin yang sering dipakai adalah MyISAM dan InnoDB.
2. Tipe Data

   Berikut ini adalah daftar tipe data yang paling sering digunakan pada MySQL.

   a. Tipe data DATE dan TIME

DATE

Tipe data DATE berguna untuk menyimpan informasi tanggal. Walaupun MySQL menampilkan tanggal dengan format YYYY-MM-DD. Nilai-nilai dapat dimasukkan baik menggunakan angka atau string. Sebagai contoh, 20100810 dan 2010-08-10 akan diterima sebagai masukan yang valid. Rentang format DATE adalah 1000/1/1 sampai 9999-12-31.0810.

DATETIME

Tipe data DATETIME berguna untuk menyimpan kombinasi informasi tanggal dan waktu. Seperti halnya DATE, nilai DATETIME disimpan dalam format standar YYYY-MM-DD HH:MM:SS. Nilai tersebut dapat dimasukkan baik menggunakan angka atau string. Sebagai contoh, 20100810153510 2010-08-10 15:35:10 akan diterima sebagai masukan yang valid. Rentang format DATETIME adalah 1000/01/01 00:00:00 9999-12-31 23:59:59.

TIME

Tipe data TIME berguna untuk menyimpan informasi waktu. Rentang dari TIME adalah -838:59:59 sampai 838:59:59.

TIMESTAMP [DEFAULT][ON UPDATE]

Tipe data TIMESTAMP berbeda dengan DATETIME dalam default yang dipakai oleh MySQL akan secara otomatis mengupdate kepada tanggal dan waktu yang beralangsung saat itu. Nilai TIMESTAMP memiliki format HH:MM:SS. Nilai-nilai dapat dimasukkan menggunakan angka atau string.

Rentang nilai yang dipakai TIMESTAMP adalah 1970-01-01 00:00:01 sampai 2037-12-31 23:59:59.

YEAR[(2|4)]

Tipe data YEAR digunakan untuk menyimpan informasi tahun. Tipe data YEAR memiliki rentang nilai dengan dua digit nomor bernilai 1-99, empat digit nomor bernilai 1901-2155, dua digit string bernilai "00" – "99" , dan empat digit string bernilai "1901" – "2155".

b. Tipe data numerik

BOOL, BOOLEAN

BOOL dan BOOLEAN hanya alias untuk TINYINT(1), dimaksudkan untuk menugaskan perintah *true* dan *false*. *True* akan bernilai 1 dan *false* akan bernilai 0.

INT [(M)] [UNSIGNED] [ZEROFILL]

Tipe data INT menyediakan rentang bilangan bulat yang kedua terbesar. Rentang nilai dari INT untuk signed adalah 2147483648 sampai 2147483647, sedangkan untuk unsigned bernilai dari 0 sampai 4294967295.

TINYINT [(M)] [UNSIGNED] [ZEROFILL]

Tipe data TINYINT menyediakan rentang nilai bilangan bulat terkecil. Untuk rentang nilai signed adalah -128 sampai 127, sedangkan untuk unsigned dari 0 sampai 255.

DECIMAL([M[,D]]) [UNSIGNED] [ZEROFILL]

Tipe data DECIMAL merupakan angka floating-point yang disimpan sebagai string. Rentang nilai untuk signed adalah

1.7976931348623157E+308 sampai -2.2250738585072014E-308, sedangkan untuk unsigned dari 2.2250738585072014E-308 sampai 1.7976931348623157E+308.

c. Tipe data string

TEXT

Tipe data TEXT merupakan tipe data nonbinari terbesar ketiga yang mendukung rentang nilai maksimum 65.535 karakter.

ENUM(“member1”,”member2”,…”member65,535”)

Tipe data ENUM memungkinkan untuk penyimpanan beberapa anggota yang berbeda dalam satu kolom. Jumlah anggota maksimal yang bisa ditampung oleh ENUM adalah 65.535 anggota. Jika deklarasi kolom meliputi atribut NULL maka NULL akan dianggap sebagai nilai yang valid dan akan menjadi default. Jika NOT NULL dideklarasikan, maka anggota pertama akan menjadi default.

d. Atribut tipe data

AUTO_INCREMENT

Atribut AUTO_INCREMENT menetapkan atribut kedalam kolom yang akan menghasilkan ID+1 untuk setiap baris yang akan dimasukkan. Hanya ada satu kolom AUTO_INCREMENT untuk setiap tabel.

NOT NULL

NOT NULL akan melarang setiap pemasukan nilai NULL pada kolom yang didefinisikan dengan NOT NULL.

NULL

NULL akan memperbolehkan pengisian nilai NULL pada kolom yang didefinisikan dengan NULL.

PRIMARY KEY

PRIMARY KEY merupakan atribut kunci dari setiap tabel. Atribut ini tidak boleh NULL dan bersifat UNIQUE yaitu tidak boleh ada PRIMARY KEY yang sama dalam sebuah tabel.

2.5. CodeIgniter

CodeIgniter merupakan *Application Development Framework*, sebuah alat untuk membangun web menggunakan bahasa pemrograman PHP. Tujuannya adalah memungkinkan pembangunan sebuah web lebih cepat daripada melakukan *coding* dari awal, dengan menyediakan banyak *library* yang dibutuhkan, *interface* dan struktur logika yang mudah untuk mengakses *library* tersebut. CodeIgniter memfokuskan pengkodean dengan meminimalisir jumlah kode yang dibutuhkan untuk mengerjakan sebuah pekerjaan [14].

2.5.1. Instalasi CodeIgniter

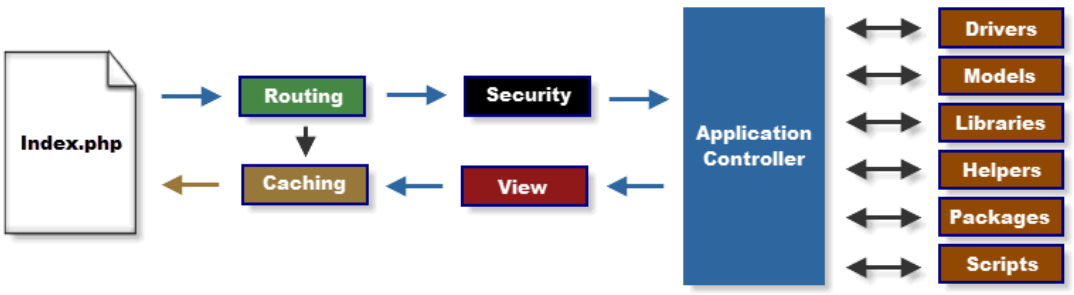
Sebelum meng-*install* CodeIgniter di web server, terlebih dahulu harus mengunduh versi CodeIgniter yang digunakan. Saat pertama rilis, CodeIgniter meluncurkan CodeIgniter 1.0. versi terakhir yang telah di rilis adalah CodeIgniter 2.1.0. Pengunduhan dapat dilakukan dengan mengunjungi web CodeIgniter

(http://www.codeigniter.com/downloads/). Setalah melakukan pengunduhan, lakukan proses installasi seperti langkah berikut ini [14] :

1. Ekstrak *package* CodeIgniter
2. Unggah *folder* dan *file-file* CodeIgniter ke web server.
3. Buka CodeIgniter/application/config/config.php dengan menggunakan sebuah *editor*, dan *setting base* URL. Jika menggunakan enkripsi atau *session*, atur kunci enkripsi pada *file* tersebut.
4. Jika menggunakan *database* dalam membangun web, panggil *database* dengan membuka application/config/database.php dan setting konfigurasi *database*.

2.5.2. Diagram Alir Aplikasi

Gambar dibawah ini akan mengilustrasikan bagaimana data mengalir di seluruh sistem [14].



Gambar II.2 – Ilustrasi Aliran Data di CodeIgniter

1. Index.php berfungsi sebagai pengontrol awal, menginisialisasikan sumber dasar yang dibutuhkan untuk menjalankan CodeIgniter.
2. *Router* akan mengkaji permintaan HTTP untuk menentukan apa yang akan dilakukan.

3. Jika ada *file cache*, *cache* akan dikirim langsung ke *browser* dengan melewati eksekusi sistem yang normal.
4. Keamanan. Sebelum aplikasi *controller* di-*load*, permintaan HTTP dan data apapun yang di masukkan *user* akan di filterisasi untuk keamanan.
5. *Controller* akan me-*load model*, *library*, *helper*, dan beberapa sumber daya yang dibutuhkan untuk permintaan yang spesifik.
6. *View* selesai diterjemahkan kemudian mengirimkan data ke web *browser* untuk dapat dilihat oleh *user*. Jika *caching* diaktifkan, akan menampilkan permintaan yang pertama sehingga permintaan selanjutnya bisa dilayani.

2.5.3. *Model-View-Controller*

CodeIgniter didasarkan pada pengembangan *Model-View-Controller*. MVC adalah sebuah pendekatan perangkat lunak yang memisahkan aplikasi logika dari presentasi. Dalam prakteknya memungkinkan halaman web mengandung *scripting* yang kecil karena *file* presentasi terpisah dengan skrip PHP.

Konsep *Model-View-Controller* pertama kali diperkenalkan oleh Xerox PARC, pada penelitian yang bekerja di sebuah bahasa pemrograman Smalltalk pada tahun 1970-1980. Smalltalk merupakan sebuah *object-oriented*, tipe yang dinamik, bahasa pemrograman yang *reflective*. Pada penggunaan pertamanya digunakan dalam sebuah pembelajaran pendidikan, dan berbeda dari sebuah data *mainframe* dan struktur lontrol dalam program Smalltalk yang meliputi [8]:

1. *Windowed user interface*
2. Konsep pemrograman yang berorientasi objek

3. Meninggalkan pesan diantara komponen objek
4. Kemampuan dalam memonitor dan mengubah struktur program

Model-View-Controller merupakan sebuah modular dari PHP *framework* yang berguna untuk membantu *web developer* dalam memisahkan kode-kode program ke dalam bagian yang jelas, membuat keseluruhan aplikasi mudah untuk dikembangkan, dirawat, dan diperluas. MVC *framework* mempunyai struktur yang baik, mengijinkan *developer* untuk berkonsentrasi terhadap apa yang *client* inginkan dan tidak perlu khawatir mengenai persoalan yang mempengaruhi setiap proyek. *Model-View-Controller* adalah sebuah nama yang menyatakan secara tidak langsung mengenai pola atau susunan desain yang mengijinkan *developer* untuk memisahkan kode-kode program kedalam tiga kategori, yaitu [14] :

a. *Model*

   Merepresentasikan struktur data. Biasanya kelas model akan berisi fungsi yang akan membantu *programmer* untuk mengambil, memasukkan, dan memperbaharui informasi didalam basis data.

b. *View*

   Memberikan informasi kepada pengguna web. *View* biasanya akan menjadi sebuah halaman web, tetapi dalam CodeIgniter, *view* bisa juga menjadi fragmen halaman seperti *header* atau *footer*.

c. *Controller*

   Berfungsi sebagai perantara antara *Model*, *View*, dan sumber daya lain yang dibutuhkan untuk memproses permintaan HTTP dan menghasilkan suatu halaman web.

Karena MVC dipisahkan kedalam tiga bagian maka *developer* dapat membuat *multiple views* dan *controllers* untuk beberapa pemberian *model* tanpa memaksakan perubahan pada desain model. Pemisahan ini mengijinkan perawatan, pemindahan dan pengorganisasian *file* yang lebih mudah daripada melakukan *coding* tanpa MVC. Tidak semua kondisi mengharuskan *model* berperan dalam semua *input*-an yang diberikan oleh *user*. *Model* digunakan untuk beberapa kondisi yang melibatkan pengambilan data dari basis data.

2.5.4. Pengkodean *Model-View-Controller*

Berikut ini adalah struktur dasar yang dimiliki model *Model-View-Controller* yang ada di CodeIgniter [14].

1. Membuat *Model*

   Model merupakan sebuah kelas php yang dirancang untuk mengelola informasi pada suatu basis data. Semua *model* mempunyai struktur antatomi seperti dibawah ini.

```
class Model_name extends CI_Model{
      function __construct(){
            parent::__construct();
      }
}
```

   Model_name merupakan sebuah nama kelas yang sama dengan nama *model* tersebut, penulisan Model_name diawali dengan huruf kapital di awal kata. File model ini akan disimpan kedalam *folder* model yang berada di application/model/file_model_anda. File model bisa disimpan dalam sebuah

sub folder. *Model* akan dipanggil oleh sebuah *controller*. Untuk memanggil sebuah *model*, kode yang digunakan adalah :

```
$this->load->model('Model_name');
```

Jika terdapat sebuah fungsi pada model, maka pemanggilan fungsi tersebut adalah :

```
$this->Model_name->function();
```

2. Membuat *View*

View merupakan sebuah kode HTML sederhana bisa berupa *header*, *footer*, *content, sidebar,* dll. Pembuat *view* sangat fleksibel, tergantung dari apa yang ingin pengguna tampilkan pada layar *browser*. Sebuah *view* tidak bisa dipanggil secara langsung, *view* harus dipanggil oleh sebuah *controller*. Perlu diingat bahwa dalam konsep MVC *controller* bertanggungjawab atas semua lalu lintas yang terjadi pada sebuah program. Dibawah ini adalah contoh penggunaan *view* pada CodeIgniter. Misal kode dibawah akan disimpan dengan nama file view_name.php.

```
<html>
  <head>
    <title>My First CodeIgniter Project</title>
  </head>
  <body>
    <h1>Hello World<h1>
  </body>
</html>
```

Semua file view akan disimpan kedalam application/views/view_name. Untuk melakukan panggilan view, kodenya adalah :

```
$this->load->view('view_name');
```

View dapat di simpan pada *sub folder*. Untuk memanggil *view* yang berada di *sub folder* bisa menggunakan kode dibawah ini.

```
$this->load->view('folder_name/view_name');
```

3. Membuat *C`ontroller*

Controller merupakan kelas sederhana yang akan muncul ketika sebuah URI dipanggil. *Controller default* akan digunakan sebagai *controller* yang pertama kali di panggil ketika sebuah program dijalankan di CodeIgniter. Telah dijelaskan diatas bahwa *controller* berperan sangat penting di CodeIgniter, *controller* merupakan perantara antara *view* dan *model*. Jika tidak ada sebuah *controller* maka *view* dan *model* tidak bisa dipanggil. Akibatnya tidak ada sebuah program yang bisa dijalankan. *Controller* akan selalu disimpan dalam application/controllers/controller_name. Huruf awal dari nama kelas *controller* harus diawali dengan huruf kapital. Dibawah ini merupakan anatomi pembuatan sebuah *controller*.

```
class Controller_name extends CI_Controller{

      public function __construct(){
            parent::construct();
            //kode contructor berada di sini
      }
}
```

Untuk mengatur *default controller* ada pada application/config/routes.php.

```
$route['default_controller'] = 'Controller_name';
```

Jika terdapat file *view*, *controller* akan memanggil terlebih dahulu *file view* tersebut, kemudian *view* akan ditampilkan pada *browser*. dibawah ini adalah contoh pemanggilan *view* menggunakan *controller*.

```
class Controller_name extends CI_Controller{
      public function __construct(){
            parent::construct();
            //kode contructor berada di sini
      }
      public function index(){
            $this->load->model('Model_name');
            $this->load->view('view_name');
      }
}
```

Controller tersebut akan memanggil *model* dengan nama Model_name dan akan memanggil *view* dengan nama view_name. Perintah yang disimpan dalam fungsi __construct() secara otomatis bisa digunakan oleh seluruh fungsi yang berada di kelas *controller* tersebut. Sedangkan perintah yang ada di fungsi index akan menjadi fungsi pertama yang dipanggil ketika kelas *controller* tersebut digunakan.

2.6. jQuery

jQuery memberikan kemudahan bagi *web designer* dan *web developer* untuk membuat efek-efek terhadap halaman web dengan tidak banyak melakukan *coding*. Karena jQuery mudah diimplementasikan maka jQuery sangat populer dalam pengembangannya. Salah satu contoh penggunaan jQuery adalah pada halaman Facebook dan Twitter. Ketika melakukan *posting* sebuat *tweet* baru pada Twitter, halaman tersebut tidak akan di *reload*. Kode program akan memanggil sebuah fungsi jQuery yang akan muncul pada barisan paling atas dan kemudian *tweet-tweet* yang lain akan turun kebawah. Penggunaan jQuery lainnya pada Twitter adalah ketika penerimaan notifikasi *tweet* baru, ketika ada *tweet* terbaru yang akan dimunculkan ada link aktif yang memberitahu ada *tweet* baru. Jika link

tersebut di klik maka *tweet-tweet* terbaru akan muncul dihalaman *home* kemudian *tweet-tweet* yang lama akan turun kebawah dan posisinya akan berada di bawah *tweet* yang baru. Halaman web tersebut tidak pernah me-*reload* dirinya sendiri, konten akan berubah secara dinamik.

Untuk mengetahui lebih lanjut mengenai jQuery, harus mengerti sedikit mengenai JavaScript. JavaScript merupakan sebuah bahasa pemrograman yang di mengerti oleh sebuah *web browser*. Kode JavaScript dapat berinteraksi dengan gambar dan teks pada halaman sebuah web, contohnya seperti menyembunyikan gambar, menggerakkan sebuah teks, atau mengubah isi halaman dalam periode waktu tertentu, atau ketika pengunjung web melakukan sebuah aksi seperti *scrolling* kursor. Kode JavaScript bisa membuat sebuah gambar menghilang ketika seseorang mengklik sebuah tombol pada halaman web, bisa membuat *window pop up* selama 30 detik. JavaScript biasanya digunakan untuk menambah keinteraktifan dan kedinamisan sebuah halaman web. Tetapi JavaScript adalah sebuah bahasa pemrograman yang utuh yang harus dipelajari. jQuery adalah sebuah *library* tambahan untuk JavaScript. Sebelum menggunakan jQuery, perlu men-*download file* jQuery. JQuery dapat diunduh di web resminya yaitu http://www.jquery.com.

Setelah menyimpan unduhan jQuery pada web server. Saatnya mencoba memanggil jQuery tersebut. Langkah-langkahnya adalah sebagai berikut [1] :

1. Buka sebuah teks *editor* untuk meng-*edit file* HTML dan *file* JavaScript.

2. Buat sebuah *file* dengan nama test.html yang isinya seperti sintaksis dibawah ini kemudian simpan *file* tersebut dalam satu *folder* bersama *file* jQuery yang telah diunduh sebelumnya.

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
    $(document).ready(function(){
       $("p").click(function(){
       $(this).hide();
    });
});
</script>
</head>
<body>
    <p>If you click on me, I will disappear.</p>
</body>
</html>
```

3. Panggil *file* test.html di *browser*. Hasilnya adalah jika kalimat yang berada dalam *tag* di klik, maka kalimat tersebut akan disembunyikan. Itu karena tag sudah menggunakan *event* hide() dari jQuery.

Sintaksis $(this).hide(); mendemostrasikan metode hide() yang dimiliki oleh jQuery yaitu untuk menyembunyikan elemen dari HTML. Contoh lainnya sintaksis $(“#test”).hide(); akan mendemonstrasikan metode hide() yang memiliki id=”test”. Sintaksis jQuery dibuat khusus untuk memilih elemen HTML dan melakukan beberapa tindakan pada elemen. Sintaksis dasarnya adalah $(selector).action(). Tanda dolar untuk mendefinisikan jQuery, (selector) untuk menemukan elemen HTML dan action() merupakan tindakan jQuery pada sebuah elemen. Pada semua metode jQuery berada dalam fungsi document.ready(). Hal ini digunakan untuk mencegah sintaksis jQuery dieksekusi sebelum dokumen

selesai di-*load*. Berikut ini akan dijelaskan lebih rinci mengenai beberapa sintak yang dimiliki jQuery [1].

1. jQuery *selector*

   jQuery *selector* berfungsi untuk memilih dan memanipulasi elemen HTML sebagai satu kesatuan atau sebagai elemen tunggal. jQuery *selector* diperlukan pada semua tahapan ketika menggunakan jQuery. *Selector* memungkinkan sebuah elemen dengan tepat dapat dipilih. jQuery mendukung pemilihan elemen pada CSS. Semua tipe *selector* dimulai dengan tanda dolar dan tanda kurung "$()".

2. jQuery fungsi *event*

   jQuery fungsi *event* ini melakukan penanganan terhadap inti fungsi di jQuery. *Event handler* adalah metode yang dipanggil ketika ada sesuatu terjadi pada HTML. Secara umum sintaksis jQuery di simpan di antara *tag* <head></head>.

```
<html>
<head>
<script type="text/javascript" src="jquery.js"></script>
<script type="text/javascript">
$(document).ready(function(){
  $("button").click(function(){
    $("p").hide();
  });
});
</script>
</head>
```

   Contoh diatas berarti fungsi ini dipanggil ketika sebuah button di klik dengan menggunakan metode hide().

3. jQuery *effect*

jQuery *effect* digunakan untuk memberikan efek *hide*, *show*, *toggle*, *slide*, *fade* dan *animate* terhadap suatu elemen. jQuery *effect* dibagi menjadi beberapa bagian, yaitu :

a. jQuery *hide* dan *show*

jQuery *hide* dan *show* digunakan untuk menyembunyikan dan menampilkan elemen yang tersembunyi sebelumnya.

```
$("#hide").click(function(){
  $("p").hide();
});
$("#show").click(function(){
  $("p").show();
});
```

Hide dan *show* dapat menggunakan dua parameter pilihan, yaitu *speed* dan *callback*. Sintaksisnya seperti $(selector).hide(speed,callback) dan $(selector).show(speed,callback). Contoh penggunaannya seperti ini :

```
$("button").click(function(){
  $("p").hide(1000);
});
```

b. jQuery *toggle*

Metode jQuery toggle() mengubah visibilitas dari elemen HTML menggunakan metode hide() dan show(). Elemen yang disembunyikan sebelumnya bisa ditampilkan menggunakan metode ini, begitu juga sebaliknya. Sintaksis *toggle* adalah seperti ini $(selector).toggle(speed,callback). Contoh penggunaannya seperti ini :

```
$("button").click(function(){
  $("p").toggle();
});
```

c. jQuery *slide*

Metode jQuery *slide* secara bertahap mengubah ketinggian untuk elemen yang dipilih. *Slide* yang bisa digunakan adalah slideDown, slideUp, dan slideToggle. Sintaksisnya adalah $(selector).slideDown(speed,callback) untuk slideDown. $(selector).slideUp(speed,callback) untuk slideUp. Dan $(selector).slideToggle(speed,callback) untuk slideToggle.

```
CONTOH slideDown :

$(".flip").click(function(){
  $(".panel").slideDown();
});
```

```
CONTOH slideUp :

$(".flip").click(function(){
  $(".panel").slideUp()
})
```

```
CONTOH slideToggle :

$(".flip").click(function(){
  $(".panel").slideToggle();
});
```

d. jQuery *fade*

Metode *fade* secara bertahap mengubah transparansi untuk elemen yang dipilih. *Fade* yang digunakan adalah fadeIn, fadeOut, dan fadeTo. Sintaksis untuk *fadeIn* adalah $(selector).fadeIn(speed,callback), untuk *fadeOut* adalah:

```
CONTOH fadeOut :

$("button").click(function(){
  $("div").fadeOut(4000);
});
```

$(selector).fadeOut(speed,callback), dan untuk fadeTo adalah $(selector).fadeTo(speed,callback).

```
CONTOH fadeTo :

$("button").click(function(){
  $("div").fadeTo("slow",0.25);
});
```

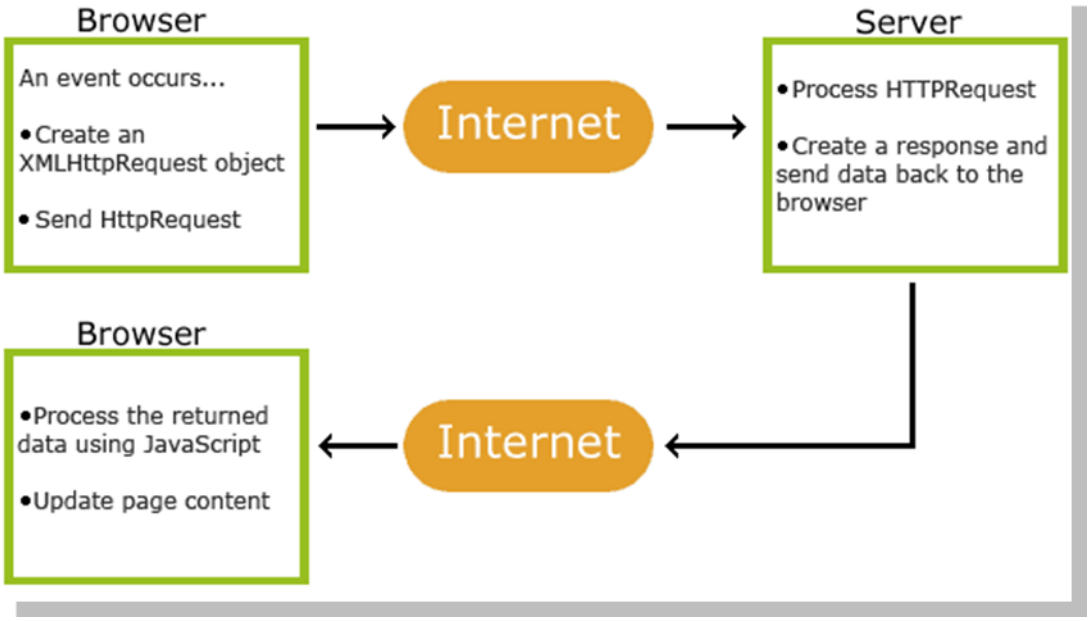
2.7. AJAX

Asynchronus JavaScript and XML atau biasa disebut dengan AJAX bukanlah bahasa pemrograman yang baru. AJAX adalah sebuah seni perubahan data dengan sebuah server, dan meng-*update* beberapa bagian dari halaman sebuah web tanpa melakukan *reload* kesemua halaman. Jika ada bagian dari sebuah halaman web menggunakan AJAX, jadi ketika halaman tersebut dibuka, maka bagian yang menggunakan AJAX tidak akan di-*reload*. Bagian yang menggunakan AJAX tersebut akan di-*reload* ketika bagian tersebut mendapatkan aksi dari pengguna atau bisa dibilang dipanggil jika dibutuhkan. Inti dari sebuah AJAX yaitu meng-*update* beberapa bagian dari halaman web tanpa melakukan *reload* satu halaman penuh web tersebut.

Untuk menggunakan AJAX, harus terlebih dahulu mengetahui tentang HTML/XHTML, CSS, dan JavaScript/DOM. AJAX adalah satu teknik dimana sebuah halaman web dinamis bisa diakses dengan cepat. AJAX mengijinkan sebuah halaman web di-*update* secara asinkronus dengan sedikit melakukan perubahan terhadap jumlah data pada server. Artinya, AJAX memungkinkan *update* bagian sebuah halaman web tanpa me-*reload* keseluruhan halaman web

tersebut. Pada sebuah halaman web yang klasik, harus melakukan *reload* kepada seluruh halaman web jika ada perubahan pada bagian halaman web tersebut. Contoh web yang menggunakan AJAX yaitu Google Maps, Gmail, Youtube, dan Facebook Tabs.

Untuk lebih jelas mengenai cara kerja AJAX, gambar dibawah ini akan mengilustrasikan apa yang dilakukan AJAX terhadab sebuah halaman web [15].



Gambar II.3 – Cara Kerja AJAX

Untuk lebih memperjelas pemahaman menganai AJAX. Berikut ini salah satu contoh halaman web yang menggunakan AJAX.

```
<html>
<head>
    <script type="text/javascript">
       function loadXMLDoc(){
          var xmlhttp;
          // code for IE7+, Firefox, Chrome, Opera, Safari
          if (window.XMLHttpRequest){
             xmlhttp=new XMLHttpRequest();
          }
          else {// code for IE6, IE5
             xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
          }
          xmlhttp.onreadystatechange=function(){
             if (xmlhttp.readyState==4 && xmlhttp.status==200){

document.getElementById("myDiv").innerHTML=xmlhttp.responseText;
             }
          }
          xmlhttp.open("GET","ajax_info.txt",true);
          xmlhttp.send();
       }
   </script>
</head>
<body>
    <div id="myDiv"><h2>Let AJAX change this text</h2></div>
    <button type="button" onclick="loadXMLDoc()">Change
Content</button>
</body>
</html>
```

Sintaksis AJAX diatas memiliki satu div dan satu *button*. Bagian div akan digunakan untuk menampilkan informasi kembali dari server dan *button* akan memanggil fungsi bernama loadXMLDoc().

Kunci dari berjalannya sebuah AJAX adalah adanya objek XMLHttpRequest. Semua *browser* modern mendukung objek XMLHttpRequest kecuali untuk IE 5 dan IE 6 *browser* tersebut menggunakan ActiveXObject. Objek XMLHttpRequest digunakan untuk bertukar data dengan server di belakang layar. Ini artinya memungkinkan memperbaharui halaman web tanpa me-*reload* keseluruhan halaman web tersebut.

1. Membuat XMLHttpRequest

   Semua *browser* modern seperti IE7, Firefox, Chrome, Safari, dan Opera memiliki *built-in* XMLHttpRequest. Sintaksis untuk menciptakan objek XMLHttpRequest ini adalah:

```
variable=new XMLHttpRequest();
```

   Sedangkan untuk membuat objek XMLHttpRequest untuk di *browser* IE 5 dan IE 6 adalah :

```
variable=new ActiveXObject("Microsoft.XMLHTTP");
```

   Untuk menangani semua web *browser* modern maupun IE 5 dan IE 6, periksa apakah *browser* tersebut mendukung objek XMLHttpRequest. Jika *browser* tersebut mendukung objek XMLHttpRequest, buatlah objek XMLHttpRequest, jika tidak maka buatlah ActiveXObject, seperti contoh dibawah ini ;

```
var xmlhttp;
if (window.XMLHttpRequest)
  {// code for IE7+, Firefox, Chrome, Opera, Safari
  xmlhttp=new XMLHttpRequest();
  }
else
  {// code for IE6, IE5
  xmlhttp=new ActiveXObject("Microsoft.XMLHTTP");
  }
```

2. Meminta XMLHttpRequest

   Objek XMLHttpRequest digunakan untuk pertukaran data dengan server. Untuk mengirim permintaan ke server, gunakan metode open() dan send() dari objek XMLHttpRequest. Sintaksisnya seperti dibawah ini:

```
xmlhttp.open("GET","ajax_info.txt",true);
xmlhttp.send();
```

Metode open(method,url,async) digunakan untuk menentukan jenis *request*, URL, dan menentukan apakah permintaan tersebut harus ditangani secara asinkronus atau tidak. Jenis permintaan yang digunakan adalah GET dan POST, URLnya berada di server, dan *async* bernilai TRUE (asynchronous) atau FALSE (synchronous). Metode send(string) digunakan untuk mengirimkan permintaan ke server. Parameter string hanya digunakan untuk permintaan POST.

3. Respon XMLHttpRequest

   Untuk mendapatkan respon dari server, gunakan responseText atau properti responseXML dari objek XMLHttpRequest. Properti responseText digunakan untuk mendapatkan respon data sebagai string. Sedangkan property responseXML digunakan untuk mendapatkan respon data sebagai data XML. Jika respon dari server bukan merupakan XML, gunakanlah property responseText. Jika respon dari server adalah XML, gunakanlah property responseXML. Contohnya adalah :

```
Contoh responseText :

document.getElementById("myDiv").innerHTML=xmlhttp.responseT
ext;
```

```
Contoh responseXML :

xmlDoc=xmlhttp.responseXML;
txt="";
x=xmlDoc.getElementsByTagName("ARTIST");
for (i=0;i<x.length;i++)
  {
  txt=txt + x[i].childNodes[0].nodeValue + "<br />";
  }
document.getElementById("myDiv").innerHTML=txt;
```

4. XMLHttpRequest readyState

   Ketika sebuah permintaan ke server dikirim, contoh yang ingin dilakukan adalah melakukan beberapa tindakan berdasarkan respon yang diterima. Event onreadystatechange dipicu setiap kali ada perubahan readyState. Property readyState mengandung status XMLHttpRequest. Tiga properti penting dari XMLHttpRequest adalah :

   a) onreadystatechange yaitu nama fungsi untuk dipanggil secara otomatis setiap kali ada perubahan terhadap readyState.
   b) readyState yaitu mengandung status XMLHttpRequest dengan perubahan dari 0 sampai 4. Nilai 0 jika permintaan tidak diinisialisasi. Nilai 1 jika koneksi server yang tidak dimungkinkan. Nilai 2 jika permintaan diterima. Nilai 3 untuk memproses permintaan. Nilai 4 kondisi dimana permintaan telah selesai dan respon siap diberikan.
   c) Status terdiri dari dua hasil. Hasil pertama bernilai 200 untuk OK. Hasil kedua 404 untuk *page not found*.

2.8. UML (*Undefine Modelling Language*)

UML (*Undefine Modeling Language*) merupakan sebuah standarisasi bahasa pemodelan untuk pembangunan perangkat lunak yang dibangun dengan menggunakan teknik pemrograman berorientasi objek. UML muncul karena adanya kebutuhan pemodelan visual untuk menspesifikasikan, menggambarkan, membangun, dan dokumentasi dari sistem perangkat lunak.

UML hanya berfungsi untuk melakukan pemodelan. Jadi penggunaan UML tidak terbatas pada metodologi tertentu, meskipun pada kenyataannya UML paling banyak digunakan pada metodologi berorientasi objek. UML memodelkan suatu sistem dengan menggunakan diagram dan teks-teks pendukung. Pada UML 2.3 ada 13 macam diagram yang dikelompokkan menjadi tiga kategori, yaitu :

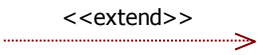
1. *Structure diagram*, yaitu kumpulan diagram yang digunakan untuk menggammbarkan suatu struktur statis dalam sistem yang dimodelkan.
2. *Behavior diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan kelakuan sistem atau rangkaian perubahan yang terjadi pada sebuah sistem.
3. *Interaction diagram*, yaitu kumpulan diagram yang digunakan untuk menggambarkan interaksi sistem dengan sistem lain maupun interaksi antarsubsistem pada suatu sistem.

Berikut ini akan dijelaskan beberapa diagram yang sering digunakan untuk memodelkan suatu analisis dan perancangan sistem yang berorientasi objek [10] :

1. *Use Case Diagram*

   Use case diagram mendeskripsikan sebuah interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. *Use case* digunakan untuk mengetahui fungsi apa saja yang ada di dalam sebuah sistem dan siapa saja yang berhak mengakses fungsi-fungsi itu. Berikut ini adalah simbol yang ada pada *use case diagram* :

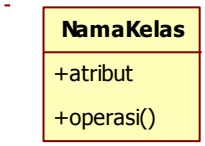
Tabel III.1 – Simbol *Use Case Diagram*

Simbol	Penjelasan
Use Case 	Fungsionalitas yang disediakan sistem sebagai unit-unit yang saling bertukar pesan antar unit atau aktor. Biasanya dinyatakan dengan menggunakan kata kerja di awal kata dalam penamaan *use case*.
Aktor 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem. Biasanya nama aktor dinyatakan dengan kata benda.
Asosiasi	Komunikasi antar aktor dan *use case* yang berpartisipasi pada *use case*, atau *use case* memiliki interaksi dengan aktor.
Extend 	Relasi antar *use case* yang menunjukkan bahwa satu *use case* merupakan fungsionalitas dari *use case* lain jika kondisi atau syarat tertentu terpenuhi.
Include 	Relasi antar *use case* yang menunjukkan bahwa *use case* yang di-*include*
Generalisasi 	Hubungan antara dua buah *use case* dimana fungsi yang satu adalah fungsi yang lebih umum dari fungsi lainnya. arah panah mengarah pada *use case* yang menjadi umumnya.

2. *Class Diagram*

Class diagram menggambarkan struktur sistem dari segi pendefinisian kelas-kelas yang dibuat untuk membangun sebuah sistem. Kelas-kelas pada sebuah sistem harus dapat melakukan fungsi sesuai dengan kebutuhan sistem. Berikut ini adalah simbol-simbol yang digunakan untuk menggambarkan sebuah *class diagram.*

Tabel III.2 – Simbol *Class Diagram*

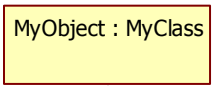
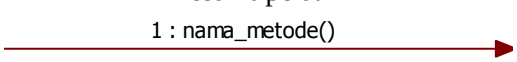
Simbol	Penjelasan
Kelas/*Class* 	Kelas yang berada pada struktur sistem.
Antarmuka/*Interface*  **nama_interface**	Sama dengan konsep *interface* pada pemrograman berorientasi objek.
Asosiasi/*Association* 	Relasi antar kelas dengan makna umum, asosiasi biasanya juga disertai dengan *multiplicity*.
Asosiasi berarah/*Directed Association* 	Relasi antar kelas dengan makna kelas yang satu digunakan oleh kelas yang lain, asosiasi biasanya juga disertai dengan *mupltiplicity*.
Generalisasi/*Generalization* 	Relasi antar kelas dengan makna generalisasi-spesialisasi (umum-khusus).
Kebergantungan/*Dependency* 	Relasi antar kelas dengan makna kebergantungan antar kelas.
Agregasi/*Agregation* 	Relasi antar kelas dengan makna semua-bagian.

3. *Sequence Diagram*

Sequence Diagram menggambarkan kelakuan objek pada *use case* dengan mendeskripsikan waktu hidup dan pesan yang dikirimkan dan diterima antarobjek. Untuk menggambarkan *sequence diagram* maka harus diketahui objek-objek yang terlibat dalam sebuah *use case* serta metode-metode yang dimiliki kelas yang diinisialisasi menjadi objek itu.

Banyaknya *sequence diagram* yang harus digambar adalah sebanyak pendefinisian *use case* yang memiliki proses sendiri. Berikut ini adalah simbol-simbol yang digunakan dalam *sequence diagram* :

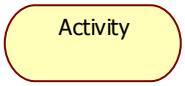
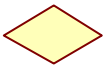
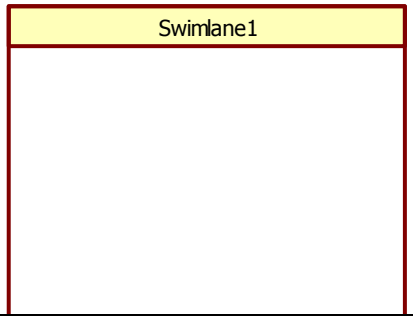
Tabel III.3 – Simbol *Sequence Diagram*

Simbol	Penjelasan
Aktor 	Orang, proses, atau sistem lain yang berinteraksi dengan sistem. Biasanya dinyatakan dengan menggunakan kata benda di awal kata nama aktor.
Garis hidup/ *lifeline*	Menyatakan kehidupan suatu objek.
Objek 	Menyatakan objek yang berinteraksi dengan pesan.
Waktu aktif 	Menyatakan objek dalam keadaan aktif dan berinteraksi dengan pesan.
Pesan tipe *create* 	Menyatakan suatu objek yang membuat objek lain, arah panah mengarah pada objek yang dibuat.
Pesan tipe *call* 	Menyatakan suatu objek memanggil operasi/metode yang ada pada objek lain atau dirinya sendiri. Arah panah mengarah pada objek yang memiliki operasi/metode, karena ini memanggil operasi/metode maka operasi/metode yang dipanggil harus ada pada diagram kelas sesuai dengan kelas objek yang berinteraksi.
Pesan tipe *send* 	Menyatakan bahwa suatu objek mengirimkan masukan/informasi/data ke objek lainnya, arah panah mengarah pada objek yang dikirimi.
Pesan tipe *return* 	Menyatakan bahwa suatu objek yang telah menjalankan suatu operasi atau metode yang menghasilkan suatu nilai balik ke objek tertentu. Arah panah mengarah pada objek yang menerima kembalian.

1. *Activity Diagram*

Activity diagram merupakan diagram yang menggambarkan aliran kerja atau aktivitas dari sebuah sistem atau proses bisnis. Berikut ini adalah simbol-simbol yang digunakan dalam *activity diagram*.

Tabel III.4 – Simbol *Activity Diagram*

Simbol	Penjelasan
Status awal 	Status awal aktivitas sistem, sebuah diagram aktivitas harus mempunyai sebuah status awal
Aktivitas 	Aktivitas yang dilakukan oleh sistem, aktivitas biasanya diawali dengan kata kerja
Percabangan 	Percabangan dilakukan ketika ada pilihan dari sebuah aktivitas
Penggabungan	Penggabungan dilakukan ketika ada lebih dari satu aktivitas yang digabungkan menjadi satu
Asosiasi 	Garis berarah yang menunjukkan arah aktivitas sistem
Status akhir 	Status akhir yang dilakukan sistem, sebuah diagram aktivitas harus mempunyai sebuah status akhir
	Memisahkan organisasi bisnis yang bertanggung jawab terhadap aktivitas yang terjadi