

2. TINJAUAN PUSTAKA

2.1 Sistem *Presensi*

Sistem merupakan bagian-bagian atau prosedur-prosedur yang saling berhubungan, berkumpul bersama-sama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu tujuan tertentu. Sistem adalah kumpulan elemen-elemen yang saling terkait dan bekerja sama untuk memproses masukan (input) yang ditujukan kepada sistem tersebut dan memproses masukkan tersebut untuk menjadi keluaran (output) yang diinginkan. (Kristanto, 2007)

Presensi adalah pencatatan dan pengolahan data presensi yang dilakukan secara terus-menerus. Pencatatan dilakukan setiap hari kerja dan dilakukan pelaporan. Presensi pegawai merupakan salah satu tolak ukur metode pengembangan pegawai. Jika presensi pegawai setelah mengikuti pengembangan meningkat, maka metode pengembangan yang dilakukan baik, sebaliknya jika presensi pegawai tetap berarti metode pengembangan yang diterapkan kurang baik.

Program aplikasi presensi yaitu suatu aplikasi yang bergerak di bidang proses pendataan karyawan, yang terdiri dari penginputan data karyawan, data presensi karyawan dan pencetakan laporan. Pendataan karyawan bertujuan untuk memasukkan data karyawan. Data presensi karyawan bertujuan untuk memasukkan data presensi karyawan yang dasarnya menyatakan karyawan itu bekerja pada hari saat dia bekerja atau tidak.

2.2 Agen Lapangan

Agen lapangan adalah seorang pegawai yang turun ke lapangan, dan mengerjakan tugasnya yang ada di luar (lapangan).

Bagi perusahaan yang bergerak di bidang jasa, keberadaan agen-agen yang turun ke lapangan sangatlah penting. Agen lapangan ini yang akan menjadi kekuatan perusahaan dalam memaksimalkan pendapatan dengan turun langsung ke lapangan dan menghadapi masalah yang ada di lingkungan pekerjaannya.

2.3 Website

Ada beberapa ahli yang mengemukakan beberapa pengertian dan juga definisi mendasar mengenai apa itu sebuah *website*. Berikut ini adalah beberapa pengertian *website* menurut para ahli:

1. Suwanto Raharjo

Menurut Suwanto Raharjo, *Web* merupakan salah satu layanan internet yang paling banyak digunakan oleh *user*nya dibandingkan dengan layanan lain seperti, seperti layanan internet *ftp*, *gopher*, *news* dan juga layanan surat elektronik atau *email*.

2. Wahana Komputer

Menurut Wahana Komputer, *Web* merupakan sebuah formulir komunikasi interaktif yang digunakan pada suatu jaringan komputer, salah satunya adalah penggunaan pada jaringan internet.

3. Taufiq Hidayatullah

Menurut A. Taufiq Hidayatullah, *Web* merupakan bagian dari sebuah aplikasi di dalam internet paling terlihat, dan juga paling banyak digunakan oleh *user* atau *client*nya.

4. Haertalib

Menurut Haertalib, *Web* merupakan sebuah lokasi di dalam jaringan internet, dimana lokasi tersebut memiliki tempat dan juga alamat tertentu, sehingga bisa diakses siapa saja.

5. Boone

Menurut Boone, *Web* merupakan sebuah koleksi sumber informasi yang kaya akan desan dan juga grafis yang mana dapat saling berhubungan satu sama lain dalam suatu jaringan internet yang lebih besar.

6. Feri Indayudha

Menurut Feri Indayudha, *Web* adalah suatu program yang dapat memuat banyak hal, seperti film, gambar, suara, serta musik yang dapat ditampilkan dan juga dilihat, dengan menggunakan teknologi jaringan internet yang ada.

7. Yuhefizar

Menurut Yuhefizar, *Web* merupakan suatu metode untuk menampilkan informasi di dalam jaringan internet, baik berupa teks, gambar, suara, maupun video yang interaktif. Lebih lanjut disebutkan pula bahwa *web* mempunyai kelebihan untuk menghubungkan (*link*) satu dokumen dengan dokumen lainnya (*hypertext*) yang dapat diakses melalui sebuah *browser*.

Dari penjabaran diatas, maka dapat dikatakan bahwa yang dimaksud dengan *web* merupakan suatu metode menampilkan informasi di dalam *browser*. Penampilan informasi ini membutuhkan *user* untuk melakukan *request*, dan juga *software* yang digunakan untuk menampilkan informasi yang direquest tersebut. *Software* yang biasa digunakan, seperti *web browser* yang biasa kita gunakan. (www.dosenit.com)

2.4 Database

Database adalah sekumpulan tabel-tabel yang berisi data dan merupakan kumpulan dari *field* atau kolom. Struktur file yang menyusun sebuah *database* adalah sebagai berikut:

1. Data

Data adalah satu satuan informasi yang akan diolah. Sebelum diolah, data dikumpulkan di dalam suatu file *database*.

2. Record

Record adalah data yang isinya merupakan satu kesatuan seperti *NamaUser* dan *Password*. Setiap keterangan yang mencakup *NamaUser* dan *Password* dinamakan satu *record*. Setiap *record* diberi nomor urut yang disebut nomor *record* (*Record Number*).

3. Field

Field adalah sub bagian dari *record*. Dari contoh isi *record* di atas, maka terdiri dari 2 *field*, yaitu: *Field NamaUser* dan *Password*. (Anhar-2010)

Dalam pemrosesan nya, *database* memiliki berbagai macam perkembangan dari periode ke periode. Berikut rangkuman sejarah pemrosesan *database* yang disajikan pada tabel 2.1

Tabel 2.1 Rangkuman Sejarah Pemrosesan *Database*

Timeframe	Teknologi	Remarks
Sebelum -1968	Pemrosesan file	Pendahulu pemrosesan database. Data disimpan dalam daftar. Karakteristik pemrosesan ditentukan oleh penggunaan umum media pita magnetik.
1968 – 1980	Hierarkis dan model network	Era pemrosesan <i>database</i> non-relasional. Model data hierarkis yang terkemuka adalah DL/I, yaitu versi pertama DBMS IBM yang disebut IMS. Model data network terkemuka adalah model CODASYL DBTG. IDMS network yang paling populer.
1980 - hingga kini	Model data relasional	Model data relasional, yang dipublikasikan pertama kali pada tahun 1970. Di aplikasikan secara komersial pada tahun 1980. IBM menyebutkan dengan DB2, vendor lainnya mengikuti dengan memodifikasi produk DBMS-nya atau dengan menciptakan produk baru. Oracle mencapai puncaknya, SQL menjadi bahasa relasional standar.
1982	Produik DBMS mikrokomputer pertama	Ashton-Tate mengembangkan produk dBase. Microrim menciptakan R:Base. Borland membuat Paradox.

Timeframe	Teknologi	Remarks
1985	Berkepentingan dalam pengembangan DBMS (OODBMS) yang berorientasi objek	Dengan ditemukannya pemrograman berorientasi objek, diusulkan OODBMS. Produk ini hanya meraih kesuksesan yang kecil secara komersial, terutama karena keunggulannya tidak menjustifikasi biaya mengkonversi miliaran byte data organisasi ke format baru. Saat ini masih dalam tahap pengembangan.
1991	Microsoft ships Access	DBMS personal diciptakan sebagai unsu Windows. Secara bertahap menggantikan semua produk DBMS personal lainnya.
1995	Aplikasi pertama <i>database</i> Internet	<i>Database</i> menjadi komponen kunci dari aplikasi Internet. Popularitas Internet meningkatkan kebutuhan dan permintaan akan keahlian <i>database</i> .
1997	Penerapan XML	Penggunaan XML memecahkan masalah pemahaman <i>database</i> jangka panjang. Vendor utama mulai mengintegrasikan XML ke produk DBMS

Dengan suksesnya media penyimpan disk pada tahun 1960-an, kita dapat memiliki akses non sekuensial, atau langsung, ke *records*. Dalam hal ini, *database* dirancang untuk menghilangkan masalah pemrosesan file sekuensial. Ada dua arsitektur atau model yang awalnya sukses. IBM mengembangkan dan mempromosikan DL/I atau *Data Language One*, yang membuat model *database* dalam bentuk pohon atau hirarki. Model ini, yang dikembangkan dalam kaitannya dengan industri manufaktur, mudah digunakan untuk menyimpan data seperti daftar *bill of material* dan suku cadang, tetapi sebenarnya bukan untuk tujuan umum. Menyajikan data network non-hierarkis ternyata cukup sulit.

Karena itu, CODASYL, kelompok yang mengembangkan standar untuk bahasa COBOL menciptakan sebuah model DBTG (*Data Base Task Group*) pada tahun 1970-an. Model DBTG dapat mewakili sistem hierarki dan *network*. Model ini pernah diajukan sebagai standar nasional, tetapi tidak pernah dipilih karena rumit. Akan tetapi, model ini telah menjadi dasar dari sejumlah produk DBMS yang sukses pada tahun 1970an dan 1980an. Produk IDMS buatan Cullinane Corporation adalah yang paling sukses. (Kroenke, David M, 2015)

2.4 MySQL

MySQL adalah sebuah sistem manajemen *database* relasi (*relational database management system*) yang bersifat *open source*. (Arbie, 2004) MySQL merupakan buah pikiran dari Michael “Monty” Widenius, David Axmark dan Allan Larson yang di mulai tahun 1995. Mereka bertiga kemudian mendirikan perusahaan bernama MySQL AB di Swedia.

Software database MYSQL kini dilepas sebagai software manajemen database yang *open source*, sebelumnya merupakan software database yang *shareware*. Database MYSQL tersedia secara bebas cuma-cuma dan boleh digunakan oleh setiap orang, dengan lisensi *open source* GNU *General Public Lisence* (GPL) ataupun lisensi komersil non GPL.

2.5 HTML

HTML adalah kepanjangan dari *HyperText Markup Language*, merupakan bahasa interpretasi yang digunakan pada sebuah halaman *web*. *HTML* mendeskripsikan struktur halaman *web* yang ditulis dengan elemen atau *tag* yang yang mengapit konten atau teks didalamnya.

Penjelasan lebih rinci mengenai arti kata-perkata dari *HTML* adalah sebagai berikut:

1. *HyperText*: adalah istilah teks aktif, yang apabila diklik akan meloncat atau menuju halaman lain. Ini merupakan kemampuan dari sebuah halaman *web* yang dapat saling berhubungan antara halaman satu dengan lainnya.
2. *Markup*: Merupakan *tag-tag* yang biasanya diawali dengan *tag* pembuka (*opening tag*) dan *tag* penutup (*closing tag*) yang memberi kemampuan untuk menata *layout* atau memformat struktur halaman *web* pada sebuah konten teks sederhana didalam *file HTML* itu sendiri.
3. *Language*: yaitu bahasa yang digunakan oleh *HTML* itu sendiri. Perintah-perintah *tag* menggunakan bahasa yang dapat dimengerti oleh *browser* atau *interpreter* lainnya.

HTML bukanlah sebuah bahasa pemrograman pada umumnya, seperti Java, C, C++, Visual Basic dan sejenisnya, melainkan bahasa *markup* / markah yang ditulis dengan perintah *tag-tag* atau elemen yang mengapit konten didalamnya yang akan ditampilkan pada sebuah halaman *web* oleh *browser* atau *HTML interpreter* (penerjemah *HTML*) lainnya.

HTML berguna untuk menampilkan konten, menghubungkan (*link*) antar halaman, memberi struktur dan informasi terkait dengan sebuah halaman *web*. Konten sebuah *web* tidak hanya terbatas pada teks saja, melainkan konten interaktif lainnya seperti video, audio, gambar dan animasi dapat disisipkan dan ditampilkan pada halaman *web*. (Betha Sidik, Pohan, Husni Iskandar, 2014)

2.6 CSS

CSS merupakan singkatan dari *cascade style sheet*, merupakan fitur baru dari HTML 4.0. hal ini diperlukan setelah melihat perkembangan HTML menjadi kurang praktis karena *web pages* terlalu banyak dibebani, hal-hal ini yang berkaitan dengan faktor tampilan seperti *font* dan lain-lain.

Untuk itu kumpulan *style* dikelola secara terpisah maka manajemen *pages* menjadi lebih mudah dan efisien, pada prakteknya penggunaan CSS ini didukung oleh *Explorer* dan *Navigator* browser terpopuler pada internet. (Betha Sidik, Pohan, Husni Iskandar, 2014)

2.7 Bootstrap

Bootstrap adalah framewok bahasa pemrograman Cascade Style Sheet (CSS), Hyper Text Markup Language (HTML), dan JavaScript yang ditujukan untuk membuat tampilan aplikasi berbasis web menjadi responsif. Maksud responsif adalah tampilan aplikasi web akan menyesuaikan dengan ukuran layar dari perangkat yang mengaksesnya. Framework ini dibuat oleh Mark Otto dan Jacob Thornton. Bootstrap pertama kali dirilis pada tanggal 19 Agustus 2011 dan berlisensi open source yang artinya bebas digunakan tanpa harus melakukan pembayaran. Alamat website resmi dari framework Bootstrap adalah <http://getbootstrap.com>. Untuk mengunduh framework Bootstrap dapat di lakukan melalui website tersebut atau dapat melalui GitHub dengan alamat <https://github.com/twbs/bootstrap/>.

2.8 Backend

Backend adalah sebuah istilah dalam pengembangan *website* di sisi *server*. Teknologi *backend* adalah segala macam teknologi yang ada di sisi *server* dari sebuah *website*. Ada banyak teknologi *backend* yang berhubungan dengan pengembangan *website*. Secara umum setidaknya ada 4 kategori teknologi backend, yaitu:

- a. Bahasa pemrograman, contoh: PHP, Python, Ruby
- b. Database, contoh: MySQL, SQL Server, Oracle
- c. Web server, contoh: Apache, Nginx, Lighttpd
- d. Web service / API, contoh: RESTful API, SOAP

2.9 PHP

PHP merupakan salah satu bahasa pemrograman berbasis *website* yang ditulis oleh dan untuk pengembangan *website*. PHP pertama kali dikembangkan oleh Rasmus Lerdorf, seorang pengembang perangkat lunak dan anggota tim Apache, dan dirilis pada akhir tahun 1994. PHP dikembangkan dengan tujuan awal hanya untuk mencatat pengunjung pada *website* pribadi Rasmus Lerdorf. Pada rilis keduanya, ditambahkan *Form Interpreter*, sebuah *tools* untuk melakukan penerjemahan perintah SQL. Rilis kedua disebut dengan PHP/FI. Sejak itu, PHP mulai diterima sebagai sebuah bahasa pemrograman baru yang sangat diminati. Terbukti pada pertengahan tahun 1997, tercatat sekitar 50.000 situs diseluruh dunia telah menggunakan PHP.

Dengan bertambah banyaknya pengguna PHP di seluruh dunia, maka PHP tidak memungkinkan lagi untuk dikelola oleh satu orang saja. Sehingga dibentuk sebuah tim pengembangan proyek *open source* "*benevolent junta*". Tim tersebut dipimpin oleh dua, Zeev Suraski dan Andi Gutmans. Keduanya lalu mendirikan sebuah perusahaan PHP dengan nama Zend (akronim dari Zeev Suraski dan Andi Gutmans). Selanjutnya Zend merilis versi PHP3 dan PHP4.

Tahun 1998 terjadi peningkatan penggunaan PHP yang sangat besar, bersamaan dengan naiknya popularitas penggunaan teknologi *open source*. Berdasarkan survei yang dilakukan Netcraft10, pada bulan Januari 2013 situs yang menggunakan PHP sudah mencapai 244 juta situs. (Ahmad Solichin, 2015)

2.10 Framework

Tahun 1998 terjadi peningkatan penggunaan PHP yang sangat besar, bersamaan dengan naiknya popularitas penggunaan teknologi *open source*. Berdasarkan survei yang dilakukan Netcraft10, pada bulan Januari 2013 situs yang menggunakan PHP sudah mencapai 244 juta situs. (Ahmad Solichin, 2015)

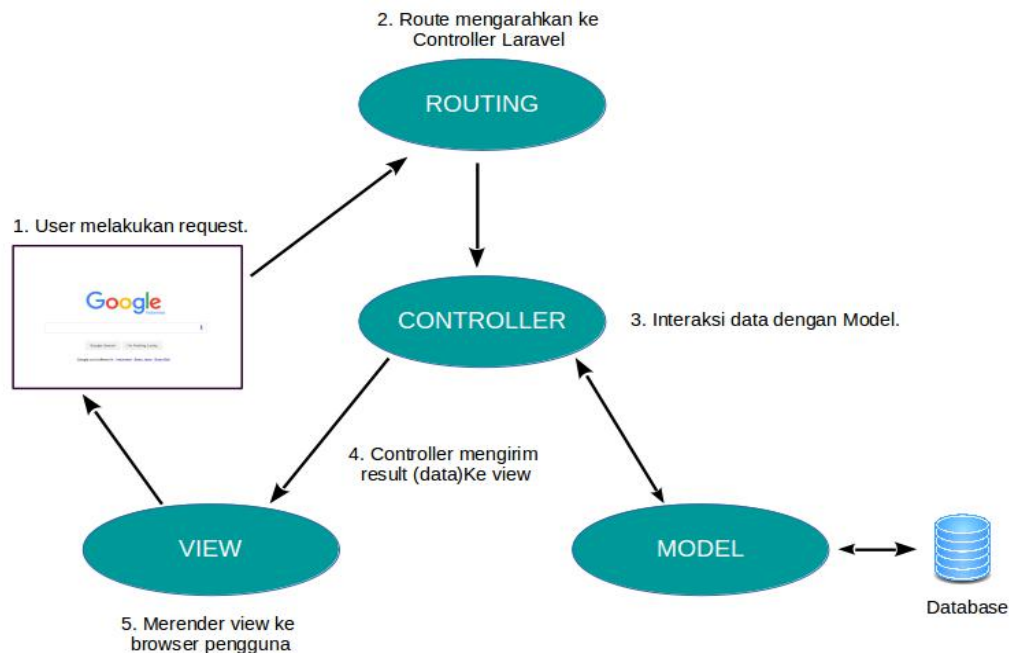
Framework adalah struktur konseptual dasar yang berisi kumpulan fungsi untuk tujuan tertentu yang sudah siap untuk digunakan, sehingga pembuatan aplikasi dapat dilakukan dengan lebih cepat karena kode programnya tidak di buat dari awal. Beberapa alasan dari digunakannya *framework* dalam membuat aplikasi adalah sebagai berikut :

1. Aplikasi akan memiliki standar pemrograman yang universal.
2. Menghindari repetitive work.
3. Memudahkan dalam team work.
4. Memudahkan dalam maintenance dan pengembangan aplikasi di masa mendatang.
5. Hemat waktu dan biaya.

2.11 Laravel

Laravel adalah *framework* bahasa pemrograman PHP yang ditujukan untuk pengembangan aplikasi berbasis *web* dengan menerapkan konsep *Model-View-Controller* (MVC). Framework ini dibuat oleh Taylor Otwell dan pertama kali dirilis pada tanggal 9 Juni 2011. Laravel berlisensi open source yang artinya bebas digunakan tanpa harus melakukan pembayaran. Alamat website resmi dari framework Laravel adalah <https://laravel.com>.

Ketika berinteraksi dengan laravel, browser akan mengirimkan *request* kepada *web server* kemudian akan diteruskan ke sistem routing Laravel. Router laravel akan memproses request kemudian mengalihkannya ke masing-masing class dan method sesuai dengan request url yang telah didefinisikan sebelumnya. Oleh Controller, terjadi komunikasi dengan model jika diperlukan data yang berhubungan dengan database. Dalam beberapa kasus, controller akan melakukan render view yang nantinya akan dikonversi menjadi HTML dan dikirim kembali ke browser.



Gambar 2.1 Cara kerja Laravel

Fitur-fitur modern Laravel yang sangat membantu developer dalam membuat aplikasi adalah Bundles, Eloquent ORM (Object-Relational Mapping), Query Builder, Application Logic, Reverse Routing, Resource Controller, Class Auto Loading, View Composers, Blade, IoC Containers, Migration, Database Seeding, Unit Testing, Automatic Pagination, Form request, dan Middleware.

Framework Laravel juga memiliki beberapa keunggulan sebagai berikut.

1. Menggunakan Command Line Interface (CLI) Artisan.
2. Menggunakan package manager PHP Composer.
3. Penulisan kode program lebih singkat, mudah dimengerti, dan ekspresif.

Fitur framework Laravel yang ditekankan pada penelitian ini adalah *Blade*, *Migration*, *Eloquent ORM* dan *Middleware*. Berikut adalah penjelasan mengenai empat fitur tersebut.

1. Blade

Blade adalah template engine. Pada dasarnya Blade adalah view namun dengan menggunakan Blade akan mempermudah untuk mengatur tampilan website dan menampilkan data. Cara untuk membuat file view menjadi file Blade adalah dengan menambahkan ekstensi `.blade.php` pada file view. Dan cara untuk memanggil file Blade sama dengan cara untuk memanggil file view biasa.

2. Migration

Migration adalah fitur yang menyediakan cara baru untuk membuat database. Dengan menggunakan migration cara membuat database melalui Command Line Interface (CLI) database atau dengan menggunakan aplikasi database manager digantikan dengan menggunakan class.

Tahapan menggunakan migration adalah membuat class kemudian melakukan perintah migrate melalui Command Line Interface (CLI) artisan.

Keuntungan menggunakan migration adalah class yang dibuat bisa dipakai

untuk membuat database pada berbagai macam Relation Database Management System (RDBMS) yang didukung oleh Laravel.

Sebagai contoh misalnya aplikasi yang digunakan selama ini menggunakan database MySQL, kemudian karena alasan pengembangan aplikasi maka akan dilakukan penggantian database ke PostgreSQL.

Dalam proses penggantian tersebut tidak perlu membuat class lagi, tinggal melakukan perintah migrate melalui Command Line Interface (CLI) artisan. Keuntungan lain dari menggunakan migration adalah semua perubahan yang dilakukan pada database akan disimpan pada suatu tabel. Sehingga bisa dilakukan pembatalan (rollback) pada database jika melakukan perubahan yang tidak benar.

3. Eloquent Object Relational Mapping (ORM)

Eloquent ORM adalah implementasi dari ActiveRecord yang digunakan untuk mengatur relasi antar tabel di database. Pada Eloquent ORM tabel direpresentasikan dalam bentuk kelas dan data yang tersimpan didalam tabel direpresentasikan dalam bentuk objek.

Relasi yang dapat diatur menggunakan Eloquent ORM adalah sebagai berikut.

A. One-to-One yaitu relasi satu ke satu.

Pada relasi ini digunakan method `hasOne` dan `belongsTo`.

B. One-to-Many yaitu relasi satu ke banyak.

Pada relasi ini digunakan method `hasMany` dan `belongsTo`.

c. Many-to-One yaitu relasi banyak ke satu.

Pada relasi ini digunakan method `belongsTo` dan `hasMany`.

d. Many-to-Many yaitu relasi banyak ke banyak.

Pada relasi ini digunakan method `belongsToMany`.

4. Middleware

Middleware adalah fitur yang menyediakan mekanisme untuk memfilter HTTP request yang masuk ke aplikasi. Laravel memiliki beberapa Middleware yaitu `Authenticate`, `EncryptCookies`, `RedirectIfAuthenticated`, dan `VerifyCsrfToken`.

Sebagai pembahasan akan dibahas `Middleware Authenticate`. Middleware tersebut akan memeriksa apakah user sudah login atau belum. Jika user sudah login maka request akan dilanjutkan ke halaman yang dikehendaki oleh user. Tetapi jika user belum login maka `Middleware Authenticate` akan mengarahkan user ke halaman login.

Jika Middleware yang sudah ada pada Laravel kurang sesuai dengan kebutuhan ataupun tidak sesuai dengan kebutuhan maka dapat dibuat sendiri Middleware yang sesuai dengan kebutuhan.

2.12 Apache

Apache adalah server web yang dapat dijalankan di banyak sistem operasi (Unix, BSD, Linux, Microsoft Windows dan Novell Netware serta platform lainnya) yang berguna untuk melayani dan memfungsikan situs web. Protokol yang digunakan untuk melayani fasilitas web/www ini menggunakan HTTP. Apache memiliki fitur-fitur canggih seperti pesan kesalahan yang dapat diatur, autentikasi berbasis basis data dan lain-lain.

2.13 Rest (Representational State Transfer)

REST adalah filosofi desain yang mendorong kita untuk menggunakan protokol dan fitur yang sudah ada pada Web untuk memetakan permintaan terhadap sumber daya pada berbagai macam representasi dan manipulasi data di Internet (Scribner, 2009).

REST adalah gaya arsitektural yang memiliki aturan seperti antar muka yang seragam, sehingga jika aturan tersebut diterapkan pada web services akan dapat memaksimalkan kinerja web services terutama pada performa, skalabilitas, dan kemudahan untuk dimodifikasi. Pada arsitektur REST data dan fungsi dianggap sebagai sumber daya yang dapat diakses lewat Uniform Resource Identifier (URI), biasanya berupa tautan pada web.

REST menggunakan protokol HTTP yang bersifat stateless. Perintah HTTP yang bisa digunakan adalah fungsi GET, POST, PUT atau DELETE. Hasil yang dikirimkan dari server biasanya dalam bentuk format XML atau JSON sederhana tanpa ada protokol pemaketan data, sehingga informasi yang diterima lebih mudah dibaca dan diparsing disisi client.

Dalam penerapannya, REST lebih banyak digunakan untuk web service yang berorientasi pada resource. Maksud orientasi pada sumber daya adalah orientasi yang menyediakan sumber daya sebagai layanannya dan bukan kumpulan-kumpulan dari aktifitas yang mengolah sumber daya itu. Bentuk web service menggunakan REST style sangat cocok digunakan sebagai backend dari aplikasi berbasis mobile karena cara aksesnya yang mudah dan hasil data yang dikirimkan berformat JSON sehingga ukuran file menjadi lebih kecil.

2.14 API (Application Programming Interface)

Web API adalah antar muka program dari sistem yang dapat diakses lewat method dan header pada protokol HTTP yang standar. Web API dapat diakses dari berbagai macam HTTP client seperti browser dan perangkat mobile. Web API juga memiliki keuntungan karena menggunakan infrastruktur yang juga digunakan oleh web terutama untuk penggunaan caching dan concurrency (Block, 2014).

2.15 Postman

Postman adalah sebuah aplikasi yang digunakan untuk melakukan uji coba REST API yang telah dibuat. Fungsi Postman adalah untuk pengecekan web service. Postman dapat menampilkan hasil dari HTTP request yang kompleks sekalipun dengan cepat. (Arianto, 2016)

2.16 Composer

Composer adalah dependency manager untuk bahasa PHP. Fungsinya agar dapat menginstall suatu library. Composer akan secara otomatis menginstall library lain yang dibutuhkan, tanpa perlu mendownload satu persatu.

Berikut perintah instalasi Laravel versi 5.5 di Command Line Interface :

```
“composer create-project --prefer-dist laravel/laravel namaprojek "5.5.*”
```

2.17 Node Package Manager (NPM)

NPM, adalah singkatan dari *node package manager*, yang mana merupakan sebuah *online repository*, untuk mempublikasikan dan mendistribusikan kode-kode Node.js dari sebuah proyek. Selain itu npm adalah sebuah command-line utility yang digunakan oleh pengembang, untuk berinteraksi dengan repository. Command-line juga biasa digunakan untuk instalasi, mengelola versi, dan manajemen dependensi paket-paket kode dari repository secara langsung. Kebanyakan library dan aplikasi yang terbuat dari Node.js dipublikasikan di npm. Namun sayangnya untuk mengunduh modul-modul yang ada di online repository, dibutuhkan akses internet.

2.18 Atom Text Editor

Atom adalah sebuah text editor yang memiliki lisensi open source yang tersedia untuk platform OS X, Linux dan Windows. Atom ini dibuat oleh GitHub dan di klaim sebagai text editor yang bisa di custom dengan merubah file configurasinya. Atom ini bersifat modular yang dimana dapat menginstall dan melakukan konfigurasi pada sebuah plugins tambahan.

Terdapat beberapa plugins yang penulis gunakan antara lain : language-blade, blade-snippets, language-babel, Atom Beautify, Minimap, color-picker, laravel.

2.19 JavaScript

JavaScript merupakan Bahasa pemrograman berbasis *Script*. *JavaScript* memiliki kemampuan untuk menciptakan kemampuan untuk menciptakan halaman *Web* yang dinamis serta didukung oleh banyak *Web Browser*. Hal ini menjadikan *JavaScript* sebagai Bahasa *Script* yang paling populer dan banyak digunakan oleh para programmer Web dalam pengembangan Web.

JavaScript berjalan di dalam kode HTML (*HyperText Markup Language*). Dengan menggunakan *JavaScript*, dapat membuat aplikasi yang interaktif pada halaman Web.

2.20 jQuery

jQuery adalah JavaScript library yang dirancang untuk meringkas kode-kode JavaScript, sehingga dapat menyederhanakan penulisan skrip program, sesuai dengan slogan “write less, do more”. jQuery pertama kali dirilis oleh John Resig pada tahun 2006, pada perkembangannya jQuery tidak hanya sebagai framework JavaScript, namun memiliki kelebihan antara lain:

1. Kemudahan mengakses dan memanipulasi elemen-elemen HTML.
2. Manipulasi CSS.
3. Penanganan event HTML.
4. Efek-efek JavaScript dan animasi.
5. Memodifikasi elemen HTML DOM.

Sintak dasar jQuery `$(selector).action()`, tanda `$` untuk mendefinisikan jQuery, jQuery selector digunakan untuk mendapatkan elemen HTML, action adalah tindakan yang dilakukan jQuery pada elemen (). Contoh penggunaan jQuery untuk menyembunyikan elemen dengan id “test” sebagai berikut. `$("#test").hide()` Semua metode jQuery berada di dalam fungsi `document.ready()` yaitu perintah inisialisasi yang menunjukkan dokumen telah siap ditampilkan dan sekaligus menjalankan perintah yang terdapat didalam fungsi.

2.21 Ajax

Teknologi AJAX (Asynchronous Javascript and XML) diperkenalkan oleh Jesse James Garret dari Adaptive Path tahun 2005. Ia mendiskripsikan bagaimana mengembangkan Web yang berbeda dengan metode tradisional melalui artikelnya yang berjudul “Ajax : A new Approach to Web Applications”. Dalam artikel ini ia yakin bahwa aplikasi Web dapat menutup jurang pemisah antara Web dan aplikasi desktop.

Andi Sunyoto, M.Kom (2007) lebih jauh menuliskan bahwa pengembangan Web secara tradisional bekerja secara synchronously antara aplikasi dan server, setiap kali melakukan link atau melakukan operasi submit pada form. Caranya browser mengirim data ke server, server merespons dan seluruh halaman akan di refresh.

Aplikasi Web yang bekerja dengan AJAX bekerja secara Asynchronously yang berarti mengirim dan menerima data dari user ke server tanpa perlu memuat kembali seluruh halaman, melainkan hanya melakukan pergantian pada bagian web yang hendak diubah. Penggunaan AJAX mulai populer ketika digunakan oleh Google pada tahun 2005.

AJAX menggunakan Asynchronous data transfer (pada HTTP request) antara browser dan web server, yang memperbolehkan halaman web me-request bit yang kecil atau seluruh informasi dari server. Teknik AJAX membuat aplikasi internet menjadi kecil, cepat dan lebih user-friendly. AJAX adalah aplikasi web yang lebih baik dan menambah keuntungan dibanding aplikasi desktop seperti dapat menjangkau pengguna yang luas, mudah diinstal, mudah dikembangkan dan efisien.

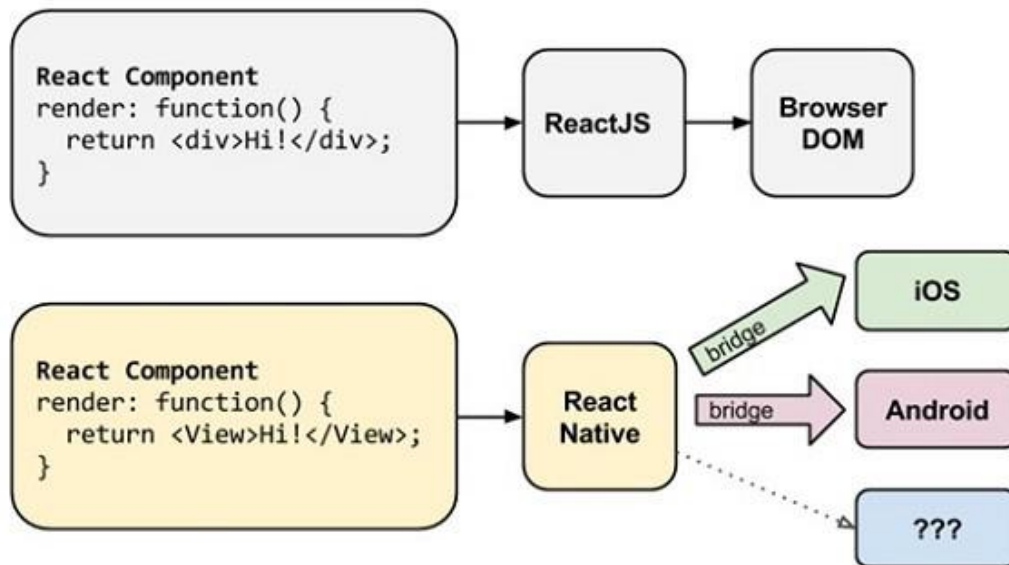
2.22 Frontend

Frontend adalah pengembangan di sisi client termasuk membuat user interface dan interaksi data dengan menggunakan api yang telah dibuat oleh sisi backend. Teknologi *frontend* adalah segala macam teknologi yang ada di sisi *client* dari sebuah sistem interaktif. Penulis menggunakan teknologi frontend untuk mengembangkan versi android dari sistem presensi dengan menggunakan pustaka React-Native.

2.23 React-Native

React Native adalah suatu framework yang diprakarsai oleh Facebook yang berguna dalam pengembangan aplikasi *mobile android* atau *ios* menggunakan teknologi web. Bahasa dari react native sendiri adalah java script dengan mengikuti tata cara penulisan ES-2016.

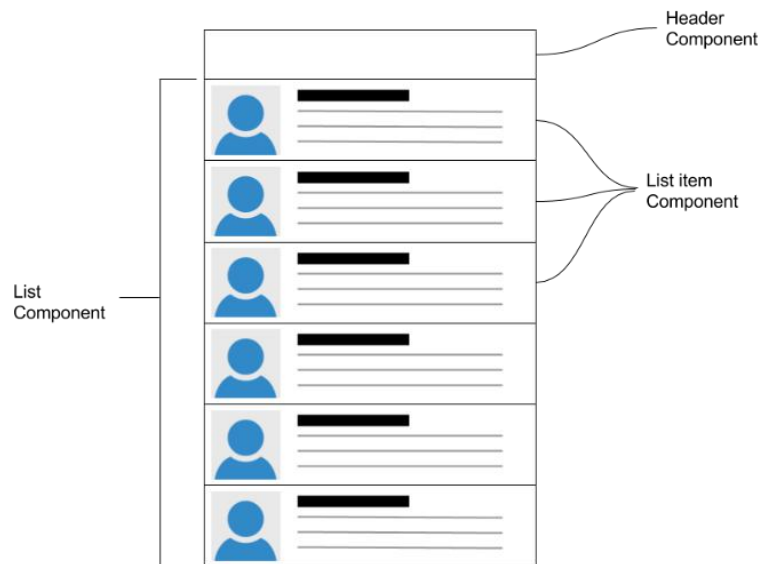
React Native bekerja dengan menanamkan file Javascript yang sudah di-bundle didalam aplikasi, dan menjalankan mereka secara local dari aplikasi yang telah dibuat. Namun dapat juga meletakkan file Javascript kita didalam server dan diambil ketika ada koneksi hal ini memungkinkan untuk melakukan update aplikasi secara cepat tanpa melalui proses submit ke Google Playstore ataupun iOS Appstore. Untuk UI dan UX, React Native juga menggunakan Javascript untuk styling hampir mirip dengan CSS diweb namun dengan CamelCase.



Gambar 2.2 Cara kerja React.JS dan React Native

React Native terdiri dari Component, Props, State, dan Lifecycle Method. Yang masing masing dari bagian mempunyai tugas tersendiri.

Component merupakan bagian utama dari sebuah program react native, dapat disebut sebagai kerangka program namun tidak seperti html, kerangka ini dapat dipisah dan dijadikan kelas-kelas kecil, hingga dapat digunakan kembali.



Gambar 2.3 Contoh Pengaplikasian Component di Dalam React Native

Props adalah singkatan dari properties, tugas utama dari properties adalah mengoper data dari satu component ke component lainnya. Namun karena react-native sama seperti dengan react.js, Hanya mendukung *one way data binding*, maka nilai dari props hanya dapat diturunkan ke kelas anak saja.

State adalah variabel global seperti di dalam bahasa pemrograman lainnya namun state bersifat dinamis dan real time. Contohnya ketika pengguna mengubah data dari state, jika component merender data dari state tersebut maka data akan langsung di ubah pada saat itu juga.

Lifecycle Method adalah sebuah metode dalam react native agar memudahkan dalam pendinamisan aplikasi yang akan dibuat. terdapat tiga lifecycle utama yaitu:

- Inisialisasi / mounting
ketika komponen dibuat/ditambahkan pertama kali pada DOM
- Update / rerender
ketika terdapat perubahan state/prop yang mengakibatkan perubahan pada DOM
- Unmounting
ketika komponen akan dihapus dari DOM

Pada setiap lifecycle tersebut, komponen react akan mengeksekusi method/fungsi yang berbeda yang kita sebut sebagai lifecycle methods. Beberapa method mempunyai prefix will dan did yang menunjukkan kapan method tersebut akan dieksekusi.

2.7 UML

Menurut Nugroho (2010:6), UML (Unified Modeling Language) adalah bahasa pemodelan untuk sistem atau perangkat lunak yang berparadigma “berorientasi objek”. Pemodelan (modeling) sesungguhnya digunakan untuk penyederhanaan permasalahan-permasalahan yang kompleks sedemikian rupa sehingga lebih mudah dipelajari dan dipahami. UML tidak hanya merupakan sebuah bahasa pemograman visual saja, namun juga dapat secara langsung dihubungkan ke berbagai bahasa pemograman, seperti JAVA, C++, Visual Basic, atau bahkan dihubungkan secara langsung ke dalam sebuah object-oriented database. Begitu juga mengenai pendokumentasian dapat dilakukan seperti, requirements, arsitektur, design, source code, project plan, tests, dan prototypes. UML terdiri atas beberapa diagram, yaitu :

- Diagram Use Case
- Diagram Class
- Diagram Package
- Diagram Sequence
- Diagram Collaboration
- Diagram StateChart
- Diagram Activity
- Diagram Deployment

Semakin kompleks bentukan sistem yang akan dibuat, maka semakin sulit komunikasi antara orang-orang yang saling terkait dalam pembuatan dan pengembangan software yang akan dibuat. Pada masa lalu, UML mempunyai peranan sebagai software blueprint (gambaran) language untuk analisis sistem, designer, dan programmer. Sedangkan pada saat ini, merupakan bagian dari software trade (bisnis software). UML memberikan jalur komunikasi dari sistem analis kemudian designer, lalu programmer mengenai rancangan software yang akan dikerjakan.

Salah satu pemecahan masalah *object oriented* adalah dengan menggunakan UML. Oleh karena itu orang-orang yang berminat dalam mempelajari UML harus mengetahui dasar-dasar mengenai *Object Oriented Solving* (pemecahan masalah OO). Tahap pertama, pembentukan model. Model adalah gambaran abstrak dari suatu dasar masalah. Dan dunia nyata atau tempat dimana masalah itu timbul bisa disebut dengan domain. Model mengandung objek-objek yang beraktifitas dengan saling mengirimkan *messages* (pesan-pesan). Objek mempunyai sesuatu yang diketahui (atribut /*attributes*) dan sesuatu yang dilakukan (*behaviors* atau *operations*). *Attributes* hanya berlaku dalam ruang lingkup objek itu sendiri (state). Lalu “blue print” dari suatu objek adalah *Classes* (kelas). Objek merupakan bagian-bagian dari kelas.

2.7.1 Diagram Use Case

Diagram *Use Case* menggambarkan apa saja aktivitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar. yang menjadi persoalan itu apa yang dilakukan bukan bagaimana melakukannya. Diagram *Use Case* dekat kaitannya dengan kejadian-kejadian. kejadian (*scenario*) merupakan contoh apa yang terjadi ketika seseorang berinteraksi dengan sistem. Diagram *Use Case* berguna dalam tiga hal, yaitu:

1. Menjelaskan fasilitas yang ada (*requirements*)

Use Case baru selalu menghasilkan fasilitas baru ketika sistem di analisa, dan design menjadi lebih jelas.

2. Komunikasi dengan klien

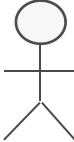


Penggunaan notasi dan simbol dalam diagram *Use Case* membuat pengembang lebih mudah berkomunikasi dengan klien-kliennya.

3. Membuat test dari kasus-kasus secara umum

Kumpulan dari kejadian-kejadian untuk *Use Case* bisa dilakukan *test* kasus layak untuk kejadian-kejadian tersebut.

Simbol yang digunakan dalam Diagram *Use Case* diantaranya terdapat pada Tabel 2.1.

Tabel 2.1 Diagram *Use Case*

Simbol	Nama	Keterangan
	Aktor	Digunakan untuk menyatakan pelaku yang berperan dari luar sistem
	<i>Use Case</i>	Digunakan untuk menyatakan fungsionalitas dari sistem
	<i>Communication</i>	Digunakan untuk menyatakan spesifikasi peran aktor atau use case terhadap use case lain

2.7.2 Diagram *Class*

Diagram *Class* memberikan pandangan secara luas dari suatu sistem dengan menunjukkan kelas-kelasnya dan hubungan mereka. Diagram *Class* bersifat statis, menggambarkan hubungan apa yang terjadi bukan apa yang terjadi jika mereka berhubungan. Diagram *Class* mempunyai 3 macam *relationships* (hubungan), sebagai berikut.

1. *Association*

Suatu hubungan antara bagian dari dua kelas. Terjadi *association* antara dua kelas jika salah satu bagian dari kelas mengetahui yang lainnya dalam melakukan suatu kegiatan. Di dalam diagram, sebuah *association* adalah penghubung yang menghubungkan dua kelas.



2. Aggregation

Suatu *association* dimana salah satu kelasnya merupakan bagian dari suatu kumpulan. *Aggregation* memiliki titik pusat yang mencakup keseluruhan bagian. Sebagai contoh : *OrderDetail* merupakan kumpulan dari *Order*.

3. Generalization

Suatu hubungan turunan dengan mengasumsikan satu kelas merupakan suatu *superClass* (kelas super) dari kelas yang lain. *Generalization* memiliki tingkatan yang berpusat pada *superClass*. Contoh : *Payment* adalah *superClass* dari *Cash*, *Check*, dan *Credit*. Tabel 2.2 memperlihatkan simbol yang digunakan dalam Diagram *Class*.

Tabel 2.2 Diagram *Class*




Simbol	Nama	Keterangan
	Class	Menggambarkan seperangkat objek dengan atribut, perilaku dan hubungan
	<i>Generalization</i>	Digunakan untuk menyatakan keterhubungan antar <i>class</i>

2.7.3 Diagram *Sequence*

Diagram *sequence* merupakan salah satu diagram *Interaction* yang menjelaskan bagaimana suatu operasi itu dilakukan, *message* (pesan) apa yang dikirim dan kapan pelaksanaannya. Diagram ini diatur berdasarkan waktu. Objek-objek yang berkaitan dengan proses berjalannya operasi diurutkan dari kiri ke kanan berdasarkan waktu terjadinya dalam pesan yang terurut.

Lifeline adalah garis *dot* (putus-putus) vertikal pada gambar, menerangkan waktu terjadinya suatu objek. Setiap panah yang ada adalah pemanggilan suatu pesan. Panah berasal dari pengirim ke bagian paling atas dari batang kegiatan (*activation bar*) dari suatu pesan pada *lifeline* penerima. *Activation bar* menerangkan lamanya suatu pesan diproses. Simbol yang digunakan dalam Diagram *Sequence* disajikan pada tabel 2.3.

Tabel 2.3 Diagram *Sequence*

Simbol	Nama	Keterangan
	<i>Object</i>	Objek yang digunakan
	<i>Stimulus</i>	Aliran pesan
	<i>Frame</i>	<i>Frame</i> objek



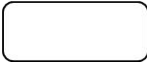
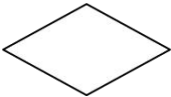


2.7.4 Diagram *Activity*

Pada dasarnya diagram *Activity* sering digunakan oleh *flowchart*. Diagram ini berhubungan dengan diagram *Statechart*. Diagram *Statechart* berfokus pada objek yang dalam suatu proses (atau proses menjadi suatu objek), diagram *Activity* berfokus pada aktifitas-aktifitas yang terjadi yang terkait dalam suatu proses tunggal. Jadi dengan kata lain, diagram ini menunjukkan bagaimana aktifitas-aktifitas tersebut bergantung satu sama lain.

Diagram *Activity* dapat dibagi menjadi beberapa jalur kelompok yang menunjukkan objek yang mana yang bertanggung jawab untuk suatu aktifitas. Peralihan tunggal (*single transition*) timbul dari setiap adanya *activity* (aktifitas), yang saling menghubungkan pada aktifitas berikutnya.

Tabel 2.4 memperlihatkan simbol yang digunakan dalam Diagram *Activity*.

Tabel 2.4 Diagram *Activity*

Simbol	Nama	Keterangan
	<i>Initial State</i>	Titik Awal
	<i>Final State</i>	Titik Akhir
	<i>Action State</i>	Aktifitas
	<i>Decision</i>	Pilihan untuk mengambil keputusan
	<i>Fork</i>	Menunjukkan kegiatan yang dilakukan secara paralel atau untuk menggabungkan dua kegiatan paralel menjadi satu
	<i>Rake</i>	Menunjukkan adanya dekomposisi

2.8 Struktur Navigasi

Struktur navigasi adalah struktur bagaimana suatu halaman di hubungkan dengan halaman lain. Terdapat empat macam struktur navigasi, yaitu linear navigation model, hierarchical model, spoke-and-hub model, dan full web model. (Iwan Binanto,2010)

2.8.1 Struktur Navigasi Linier

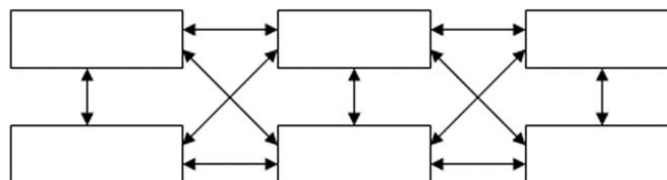
Struktur navigasi linear pada Gambar 2.3 merupakan struktur yang mempunyai satu rangkaian cerita berurutan. Pengguna akan melakukan navigasi secara berurutan, dari *frame* atau byte informasi yang satu ke yang lain.



Gambar 2.2 Struktur Navigasi Linier

2.8.2 Struktur Navigasi Non Linier

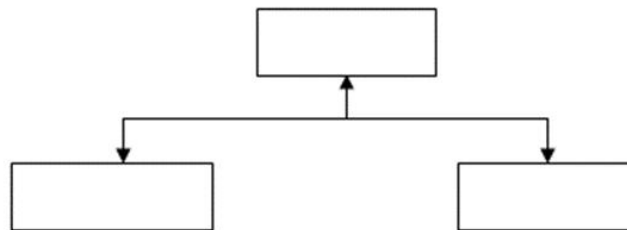
Struktur navigasi non linier atau struktur tidak berurut merupakan pengembangan dari struktur navigasi linier. Pada struktur ini diperkenankan membuat navigasi bercabang. Percabangan yang dibuat pada struktur non linier ini berbeda dengan percabangan pada struktur hirarki, karena pada percabangan non linier ini walaupun terdapat percabangan, tetapi tiap-tiap tampilan mempunyai kedudukan yang sama, yaitu tidak ada *Master Page* dan *Slave Page*. Struktur Navigasi Non Linier disajikan pada Gambar 2.3.



Gambar 2.3 Struktur Navigasi Non Linier

2.8.3 Struktur Navigasi Hirarki

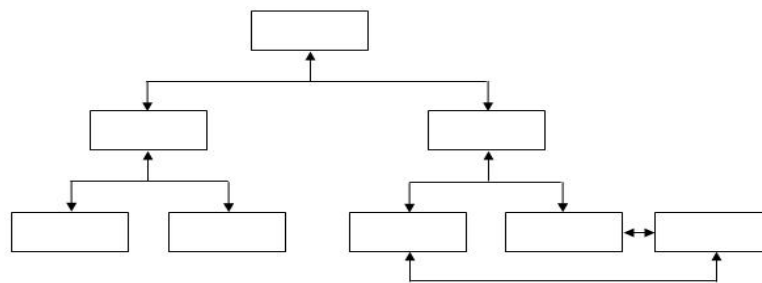
Struktur Navigasi Hirarki pada Gambar 2.5 biasa disebut struktur bercabang, merupakan suatu struktur yang mengandalkan percabangan untuk menampilkan data berdasarkan kriteria tertentu. Tampilan pada menu pertama disebut sebagai *Master Page* (halaman utama pertama), halaman utama ini mempunyai halaman percabangan yang disebut *Slave Page* (halaman pendukung). Jika salah satu halaman pendukung dipilih atau diaktifkan, maka tampilan tersebut akan bernama *Master Page* (halaman utama kedua), dan seterusnya. Pada struktur navigasi ini tidak diperkenankan adanya tampilan secara linier.



Gambar 2.4 Struktur Navigasi Hirarki

2.8.4 Struktur Navigasi Komposit

Struktur Navigasi Komposit merupakan gabungan dari ketiga struktur sebelumnya yaitu Linier, Non-Linier, dan Hiraki. Struktur navigasi ini juga bisa disebut dengan Struktur Navigasi Bebas. Struktur navigasi ini banyak digunakan dalam pembuatan multimedia sehingga dapat memberikan ke-interaksian yang lebih tinggi. Struktur navigasi komposit ditunjukkan pada Gambar 2.6.



Gambar 2.5 Struktur Navigasi Komposit