



Hak cipta dan penggunaan kembali:

Lisensi ini mengizinkan setiap orang untuk menggubah, memperbaiki, dan membuat ciptaan turunan bukan untuk kepentingan komersial, selama anda mencantumkan nama penulis dan melisensikan ciptaan turunan dengan syarat yang serupa dengan ciptaan asli.

Copyright and reuse:

This license lets you remix, tweak, and build upon work non-commercially, as long as you credit the origin creator and license it on your new creations under the identical terms.

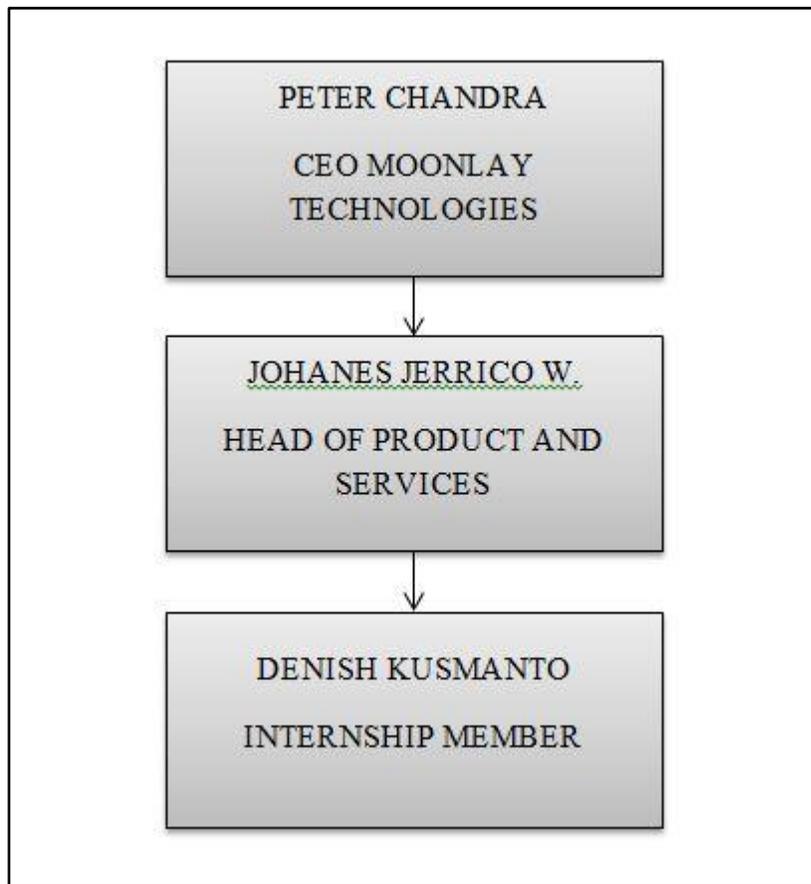
BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan dan Koordinasi

Selama bekerja di Moonlay Technologies, penulis bersama dengan para peserta magang lainnya mengerjakan sebuah proyek di bawah divisi *Product and Services* Moonlay Technologies. Di Moonlay Serpong terdapat 10 mahasiswa magang UMN yang terdiri dari sembilan mahasiswa TI dan satu mahasiswi DKV. Sembilan mahasiswa UMN mengerjakan satu proyek yang sama. Sembilan mahasiswa TI mengerjakan *web* dan *mobile application* dan mahasiswi DKV yang membuat desain layout. Para mahasiswa magang di beri tugas untuk mendesain poster sebuah perusahaan (sebagai contoh poster McD, Bank Mega, dsb). Setelah tugas awal telah selesai, Bapak Johannes Jerrico Wijaya selaku pembina lapangan selama magang dan *Head of Product and Service Division* memberikan dua pilihan kepada peserta magang TI, ingin bekerja sebagai *web programmer* (ASP.NET & MVC) atau ingin bekerja sebagai *mobile programmer*. *Web programmer* bertanggung jawab akan desain dan fitur – fitur web sementara *mobile programmer* bertanggung jawab dengan desain, penampilan, fitur – fitur dalam tampilan *mobile device* (smartphone, tablet). Penulis memilih sebagai *web programmer* sampai kegiatan magang berakhir.

U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A



Gambar 3.3 Koordinasi Kerja Magang

Saat bekerja magang di Moonlay Technologies, peserta mengembangkan sebuah produk berbasis *web*. Dalam pengembangannya, peserta magang TI di bagi ke dalam tim – tim kecil beranggotakan tiga orang di bawah bimbingan langsung dari Johannes Jerrico W. Terdapat tiga kelompok yaitu kelompok Peggy, Joe Shiendy, dan Destia kemudian kelompok Wawan, Adrian, dan Joshua dan kelompok peserta sendiri bersama Zaty Shamarkasih, dan Meylinda Sandra. Kelompok Peggy, Joe Shiendy, Destia dan kelompok Denish, Zaty, Meylinda mengerjakan *web application* sedangkan kelompok Wawan, Adrian, dan Joshua mengerjakan *mobile application*.

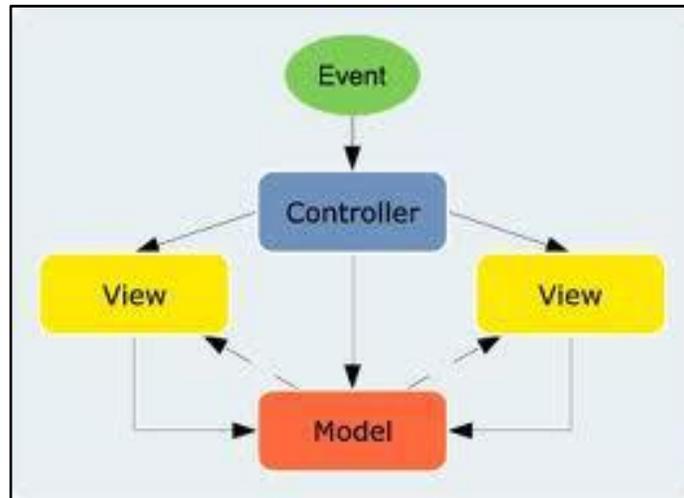
Beliau memberikan *assignment* kepada masing – masing kelompok dan kemudian menjelaskan maksud dari *assignment* tersebut. Kemudian membagi *assignment* tersebut ke dalam sub modul yang langsung diberikan kepada kelompok – kelompok peserta magang untuk dikerjakan. Sub modul yang telah selesai dikerjakan,

dapat langsung di nilai oleh beliau untuk di tinjau kembali. Jika menurut beliau belum sesuai dengan spesifikasi dan kriteria yang diinginkan, peserta magang harus segera merubah dan melaporkannya kembali untuk kembali di tinjau.

3.2 Tugas yang Dilakukan

Tugas yang dilakukan selama dua bulan kerja magang yaitu merancang produk *web*. Awalnya peserta di beri tugas untuk mendesain sebuah *web page* dan *email page* di mana *layouting web page* menggunakan perangkat lunak Adobe Dreamweaver. Setelah tugas pertama selesai, tugas kedua yaitu mendesain *email page* dimana sebelumnya peserta di minta untuk mempelajari MVC 4 (Model, View, Controller) dan ASP.NET dimana penggunaannya membutuhkan perangkat lunak Microsoft Visual Studio 2012. *Model-View-Controller* atau MVC sendiri merupakan suatu metode yang memisahkan tampilan antarmuka (*user interface*), proses pengolahan data dan data yang akan digunakan. Metode ini digunakan dalam *web-based application*. MVC dalam penerapannya menggunakan pemrograman berorientasi objek (*Object Oriented Programming*). Komponen-komponen dalam MVC :

1. *Model* merupakan bagian yang berisi data, mewakili struktur data serta berisi fungsi-fungsi yang akan digunakan berulang dalam pengolahan data. Contohnya, memasukan data ke dalam *database*, merubah isi *database*, atau menghapus data.
2. *View* merupakan tampilan antarmuka atau *interface* yang akan di lihat oleh pengguna.
3. *Controller* pada MVC berfungsi untuk memanipulasi *model*. Berisi barisan kode yang akan menjadi perantara antara *view* dengan *model*.



Gambar 3.4 Struktur MVC (<http://www.tutorialeonline.net/>)

Setelah berkulat dengan pembuatan *email page*, peserta di minta untuk mempelajari mengenai implementasi *javascript Carousel*, *infinite scroll*, *javascript zoom-in zoom-out*. Setelah mempelajarinya, peserta di minta untuk menggabungkan *JavaScript Carousel* dan *infinite scroll* dengan *virtual cards* yang telah di buat teman kelompok untuk disatukan.

3.3 Uraian Pelaksanaan Kerja Magang

3.3.1 Proses Pelaksanaan

Berikut merupakan tabel pelaksanaan kerja magang selama dua bulan di Moonlay Technologies :

UNIVERSITAS
MULTIMEDIA
NUSANTARA

No	Kegiatan	Minggu ke -							
		1	2	3	4	5	6	7	8
1	Pengenalan ASP.NET dan MVC 4	■							
2	Analisa <i>User Requirement</i>		■						
3	Membangun Tampilan Sistem		■	■					
4	Membangun Sistem			■	■	■			
5	Menguji Sistem					■	■		
6	Perbaiki Sistem					■	■		
7	Penyempurnaan Sistem							■	■
8	Konsultasi dengan Pembimbing Lapangan	■	■	■	■	■	■	■	■

Table 1.1 Pelaksanaan Kerja Magang

Pelaksanaan kerja magang diawali dengan pengenalan perangkat lunak yang akan digunakan dan melakukan instalasi perangkat tersebut. Berikut alur kerja pelaksanaan kerja magang peserta di PT. Moonlay Technologies.

Minggu ke-1

1. Pengenalan MVC 4 dan ASP.NET dengan membaca dan mencoba tutorial di internet.
2. Instalasi Microsoft Visual Studio 2012.

Minggu ke-2

1. Menganalisa *User Requirement* yang ditentukan.
2. Mencoba membangun tampilan *User Interface*.

Minggu ke-3

1. Membuat desain *User Interface* menggunakan HTML + CSS.
2. Pengenalan JavaScript *Carousel*, *infinite scroll* dengan membaca dan mencari tutorial di internet.

Minggu ke-4

1. Implementasi JavaScript *Carousel, infinite scroll* ke dalam *virtual cards*.

Minggu ke-5

1. Menggabungkan JavaScript *Carousel, infinite scroll* dengan *tipsy* ke dalam *virtual cards*.
2. Menguji sistem apakah berjalan lancar atau tidak.
3. Analisa apakah terdapat kesalahan,error pada sistem.

Minggu ke-6

1. Memperbaiki setiap kesalahan, kekurangan, error yang terjadi.
2. Menguji kembali sistem.

Minggu ke-7 dan 8

1. Memastikan kembali sistem yang dibuat telah sesuai dengan *User Requirement*.
2. Konsultasi dengan pembimbing lapangan apakah sudah sesuai atau belum.

3.3.2 Perancangan Sistem

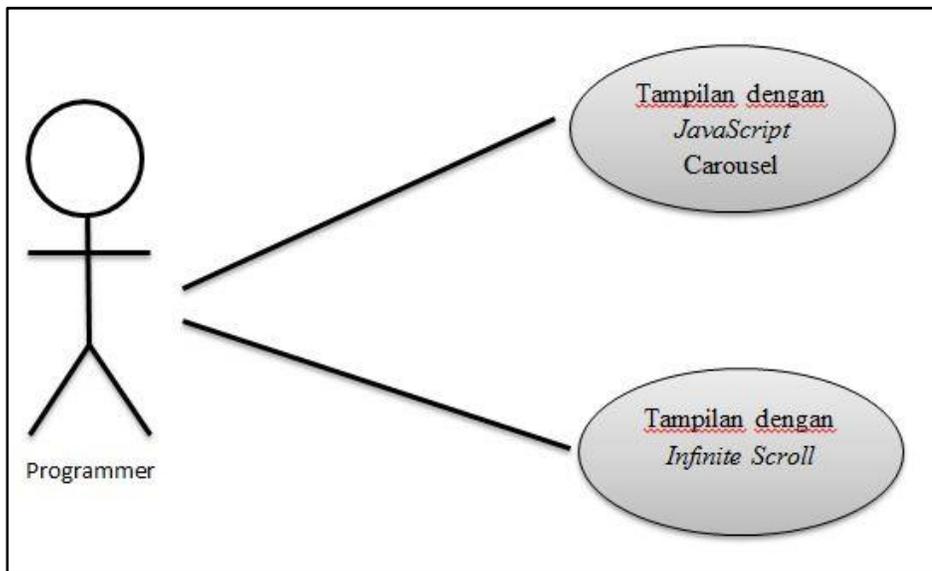
Dalam perancangan sebuah sistem membutuhkan suatu rancangan sehingga alur sistem tersebut jelas. Proses perancangan sistem ini terdiri atas *Use Case Diagram, Activity Diagram, dan Sequence Diagram*. Berikut analisa perancangan sistem

a. Metode Pengembangan Program

Metode pengembangan yang dilakukan menggunakan SDLC (*System Development Life Cycle*). Keunggulan SDLC yaitu meningkatkan mutu, kualitas, dan produktivitas.

b. Use Case Diagram

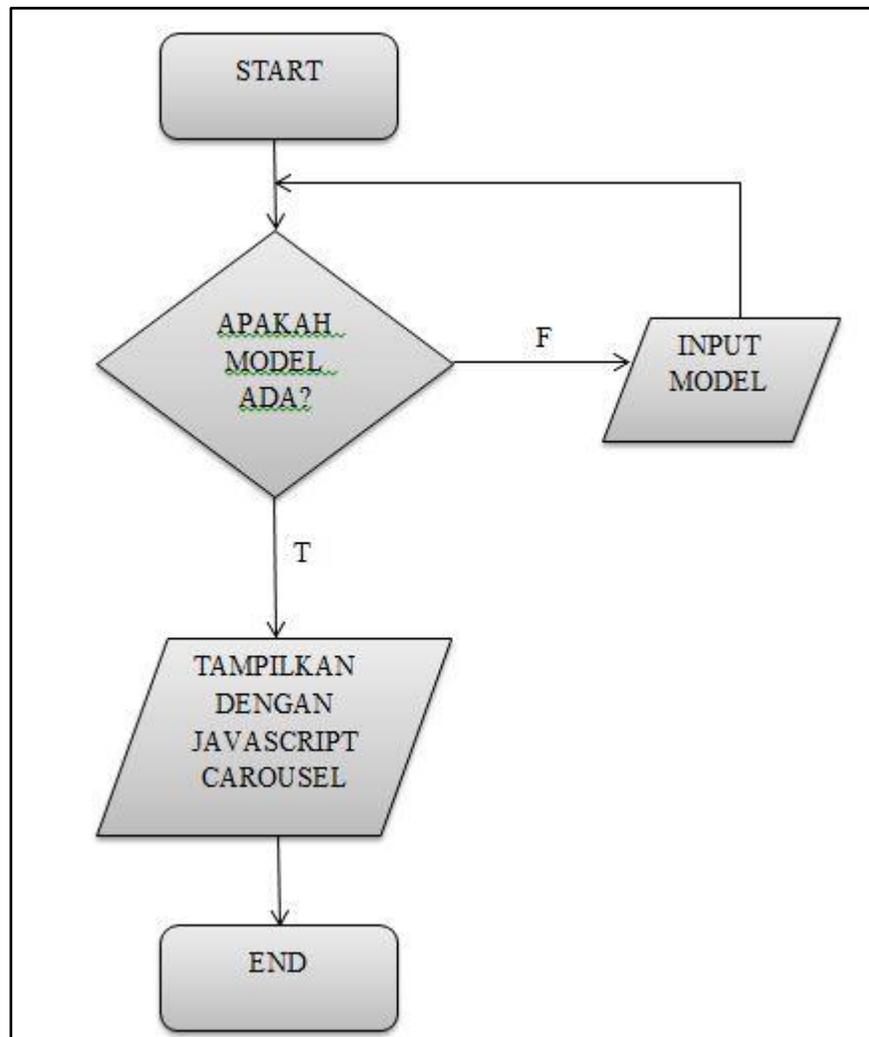
Use Case diagram menggambarkan proses sistem dari sudut pandang user.



Gambar 3.5 Use Case Diagram

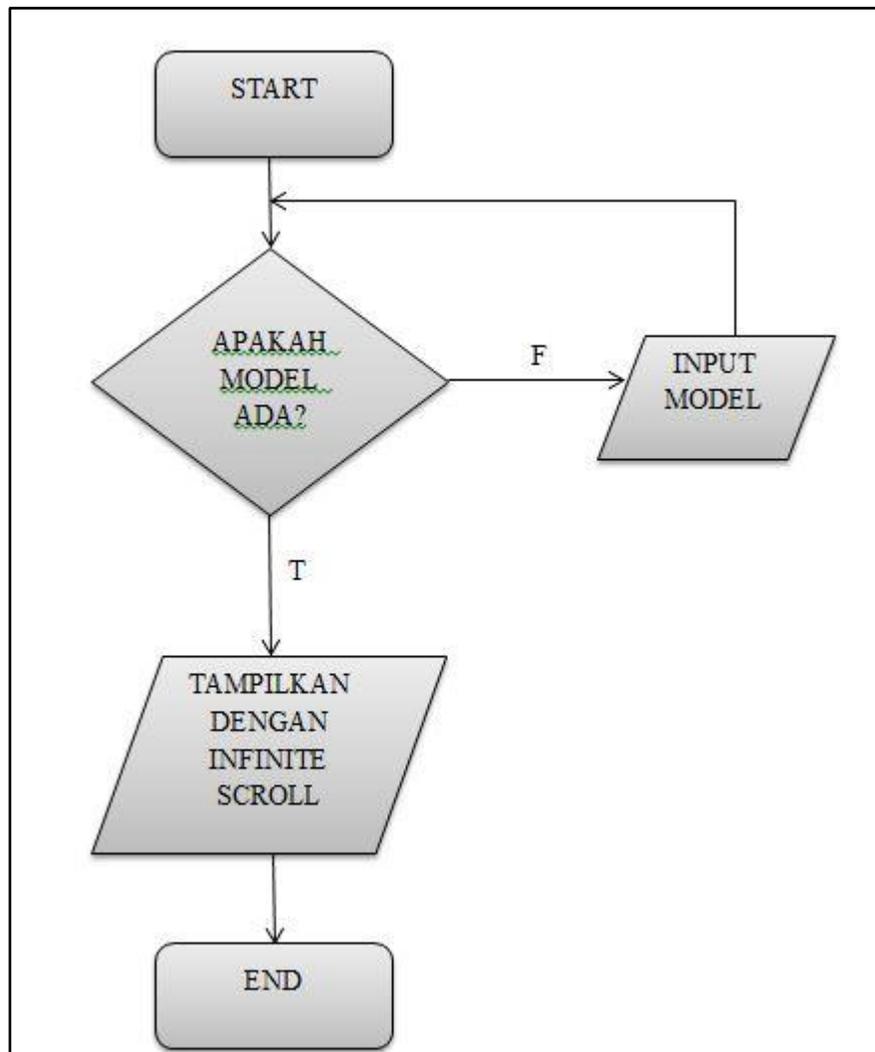
c. ***Flowchart Diaram***

Flowchart (bagan alir) merupakan bagan yang menunjukkan alir di dalam program atau prosedur sistem secara logika.



Gambar 3.6 Flowchart Diagram *JavaScript* Carousel

Pada gambar 3.6 menunjukkan bahwa saat mulai, sistem akan cek dahulu apakah model sudah ada atau belum. Jika belum maka programmer harus menginput model terlebih dahulu ke dalam sistem kemudian saat dijalankan, sistem akan kembali cek apakah model sudah ada. Jika sudah ada, maka model tersebut akan ditampilkan dengan *JavaScript* Carousel.

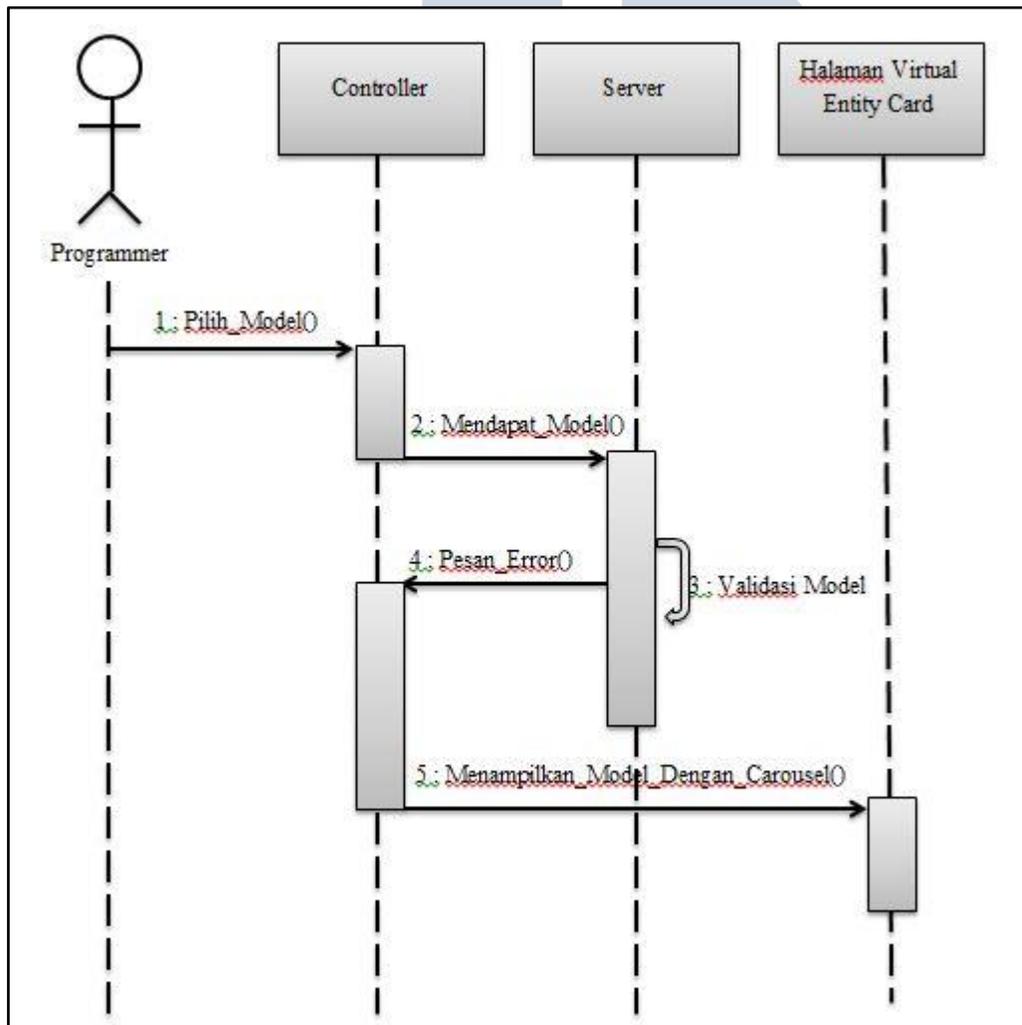


Gambar 3.7 Flowchart Diagram *Infinite Scroll*

Pada gambar 3.7 sama dengan gambar 3.6 bahwa saat mulai, sistem akan cek dahulu apakah model sudah ada atau belum. Jika belum maka programmer harus menginput model terlebih dahulu ke dalam sistem kemudian saat dijalankan, sistem akan kembali cek apakah model sudah ada. Jika sudah ada, maka model tersebut akan ditampilkan dengan *Infinite Scroll*.

d. Sequence Diagram

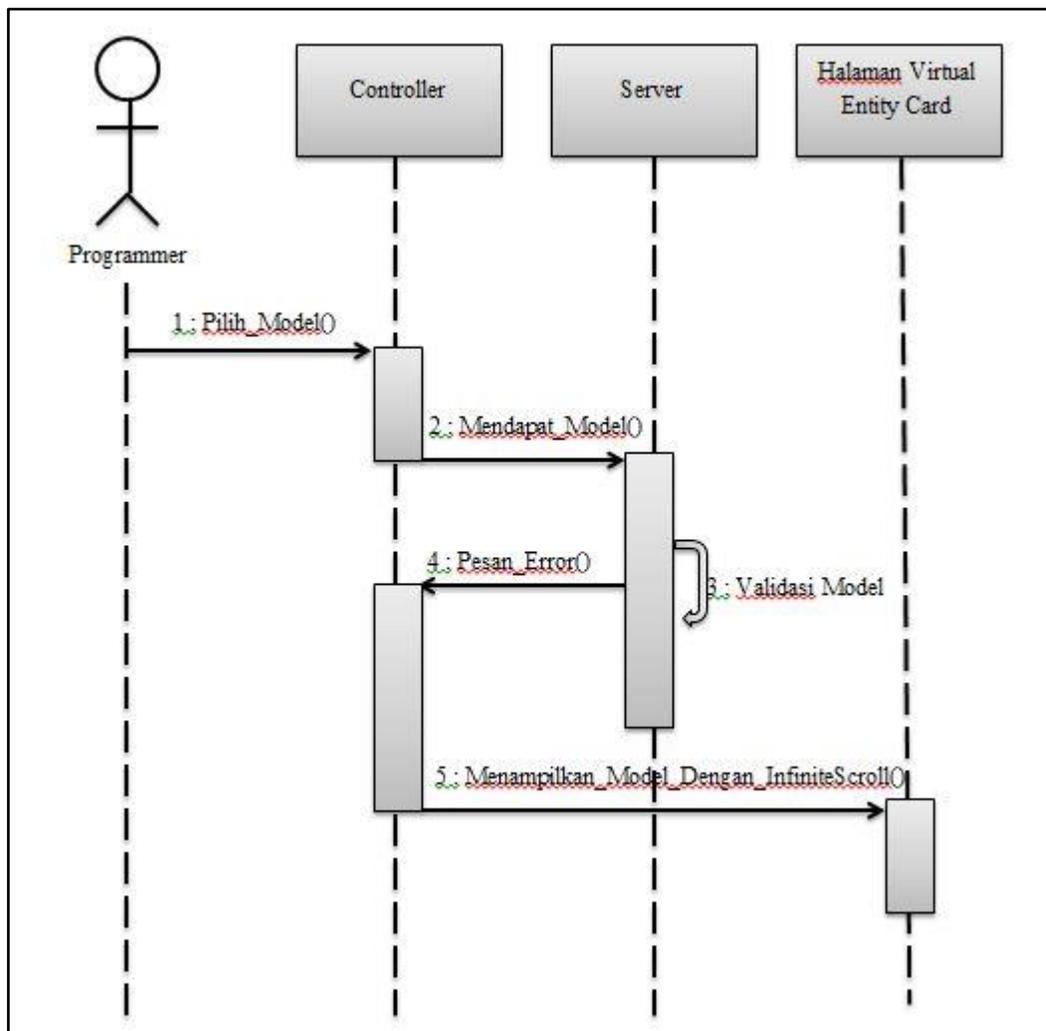
Sequence diagram (diagram urutan) merupakan diagram yang memperlihatkan / menampilkan interaksi antar objek di dalam sistem yang disusun pada suatu urutan. *Sequence diagram* digunakan untuk menggambarkan skenario atau rangkaian langkah – langkah yang dilakukan sebagai suatu respon dari suatu kejadian untuk menghasilkan *output* tertentu.



Gambar 3.8 Sequence Diagram *JavaScript* Carousel

Pada gambar 3.8 menunjukkan sequence diagram saat programmer memilih model *card* untuk digunakan ke dalam tampilan *JavaScript* Carousel. Setelah user

memilih, maka pilihan tersebut langsung di kirim oleh controller ke server. Kemudian server akan validasi dan memberikan pesan error ke controller. Jika model yang akan digunakan sudah ada dan benar, maka controller akan menggunakan model tersebut untuk ditampilkan dengan *JavaScript Carousel*.



Gambar 3.9 Sequence Diagram *Infinite Scroll*

Pada gambar 3.9 menunjukkan sequence diagram saat programmer memilih model *card* untuk digunakan ke dalam tampilan *Infinite Scroll*. Setelah user memilih, maka pilihan tersebut langsung dikirim ke server oleh controller. Kemudian server akan validasi dan memberikan pesan error ke controller. Jika model yang akan

digunakan sudah ada dan benar, maka controller akan menggunakan model tersebut untuk ditampilkan dengan *Infinite Scroll*.

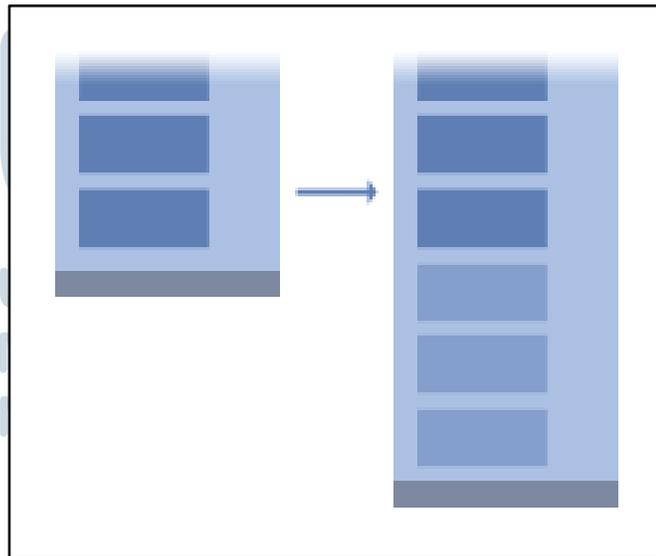
e. Rancang Tampilan User Interface

1. Contoh rancang tampilan user interface untuk *JavaScript Carousel* :



Gambar 3.10 Contoh Rancang Tampilan JCarousel

2. Contoh rancang tampilan user interface untuk *Infinite Scroll* :



Gambar 3.11 Contoh Rancang Tampilan *Infinite Scroll*

3.3.3 Implementasi

Implementasi *JavaScript* Carousel dan *Infinite Scroll* dalam proyek ini sebagai berikut.

1. *JavaScript* Carousel

Untuk mendapatkan tampilan *coverflow slider* seperti yang dipakai Apple pada iTunesnya, maka digunakan *jquery* Carousel. *Jquery* Carousel sendiri terdiri dari bermacam – macam tampilan namun yang dipakai oleh peserta yaitu jenis *waterwheel*. Cara pemakaiannya yaitu dengan menginclude *library jquery.waterwheelCarousel.js* yang telah disediakan. Contoh kode *snippet* dapat dilihat di gambar 3.12.

```
1
2 <script type="text/javascript" src="~/Scripts/jquery-ui-1.9.1.js"></script>
3 <script type="text/javascript" src="~/Scripts/jquery.waterwheelCarousel.js"></script>
4 <link rel="stylesheet" href="~/Content/styles.css" type="text/css" />
5 <link rel="stylesheet" href="~/Content/Site.css" type="text/css" />
6
7 <script type="text/javascript">
8
9 $(document).ready(function () {
10     $('#stage').load('Home/Content #carousel', function (response, status, xhr) {
11
12         var carousel = $('#carousel').waterwheelCarousel({
13             flankingItems: 3,
14             movingToCenter: function ($item) {
15                 $('#callback-output').prepend('movingToCenter: ' + $item.attr('id') + '<br/>');
16             },
17             movedToCenter: function ($item) {
18                 $('#callback-output').prepend('movedToCenter: ' + $item.attr('id') + '<br/>');
19             },
20             movingFromCenter: function ($item) {
21                 $('#callback-output').prepend('movingFromCenter: ' + $item.attr('id') + '<br/>');
```

Gambar 3.12 Contoh *snippet code* Index.cshtml

Supaya *jquery waterwheel* dapat berjalan, harus menambahkan *jquery-ui 1.9.1*. *Stage* merupakan sebuah nama *div* dimana di dalam *div* tersebut proses *jquery*

akan berlangsung. *Div* tersebut akan me-load *div* carousel yang berada di 'Home/Content'. Isi *div* carousel adalah sebagai berikut.

```
1 <div id="carousel">
2     @if (Model != null){
3         foreach(coverflowslider.Models.Gambar temp in Model){
4             
5         }
6     }
7     <a href="#" id="prev" >Prev</a> | <a href="#" id="next">Next</a>
8 </div>
```

Gambar 3.13 Contoh *snippet code div* Carousel

Pada gambar 3.13, *div* Carousel digunakan untuk mengambil gambar kupon. *Gambar* merupakan *constructor* yang berisi *setter* *getter* untuk kupon yang akan digunakan. *Temp* merupakan *variable*. *Imgcarousel* merupakan sebuah *variable* yang berada di *controller*.

Balik ke gambar no 3.12, *variable* carousel di buat untuk menampung *div* carousel yang akan di buat *coveflow slider*. Hasilnya kurang lebih akan sebagai gambar 3.14 di bawah ini.



Gambar 3.14 Implementasi JCarousel

2. *Infinite Scroll*

Penggunaan *infinite scroll* tidak jauh beda dengan penggunaan *jquery* Carousel karena sudah ada *library* khusus yaitu *jquery.infinitemscroll.js*. Contoh kode *snippet* dapat di lihat di gambar 3.15.

```
<div id="next">
  <a href="@Url.Action("Page2", "Home", new { page = 2 })"></a>
</div>
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.6.1/jquery.js"></script>
<script>window.jQuery || document.write("<script src='~/Scripts/jquery-1.6.1.js'></script>")</script>
<script src="~/Scripts/debug.js"></script>
<script src="~/Scripts/jquery.infinitemscroll.js"></script>
<script src="~/Scripts/manual-trigger.js"></script>
```

Gambar 3.15 Contoh *snippet code* pada *Index.cshtml*

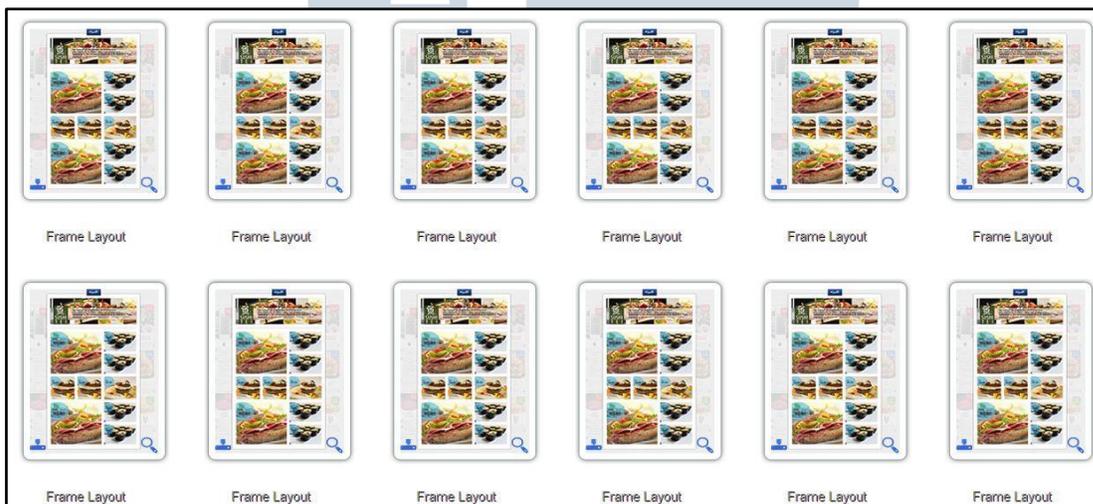
Terdapat suatu *script* seperti gambar 3.16. *PortfolioContainer* yang ada di dalam *script* ini merupakan suatu *div class* yang akan mengalami *infinite scroll*. Contoh *snippet code* dapat di lihat di gambar 3.17. Hasilnya kurang lebih akan seperti gambar 3.18.

```
91 <script>
92     $(' .portfolioContainer').infinitemscroll({
93         navSelector: "#next",
94         nextSelector: "#next a:first",
95         itemSelector: "div.portfolioContainer",
96         maxPage: 3,
97         debug: false
98     });
99 </script>
100
```

Gambar 3.16 Contoh *script* pada *Index.cshtml*

```
56 <body>
57 <div class="portfolioContainer">
58   @if (Model != null)
59   {
60     foreach(Templatess.Models.Gambar temp in Model){
61       <div class="objects">...</div>
62     }
63   }
64 </div>
65 </body>
```

Gambar 3.17 Contoh *snippet code*



Gambar 3.18 Implementasi *Infinite Scroll*

3.3.4 Kendala yang Ditemukan

Selama kerja magang di Moonlay Technologies, peserta magang menghadapi beberapa kendala dalam pengerjaan proyek. Beberapa kendala yang ditemukan selama proses kerja magang :

1. Terjadinya *miss* komunikasi dan *miss understanding* antara peserta dengan pembina kerja magang sehingga pada saat proses pengerjaan hingga akhir pengerjaan hasil tidak cocok yang berakibat peserta magang harus mulai mengerjakan tugas yang sama kembali.
2. Setelah pengerjaan suatu sistem kemudian terjadi penambahan dan pengurangan sehingga berakibat sistem harus mengalami perubahan.
3. Penggunaan *web browser* yang berbeda – beda menyebabkan tampilan yang tidak sama antar *browser* karena setiap *browser* memiliki algoritma yang berbeda – beda. Bagaimana memberikan tampilan yang sama di setiap browser yang berbeda (Internet Explorer versi 7 – 10, Mozilla FireFox, Google Chrome).

3.3.5 Solusi atas Kendala yang Ditemukan

Untuk mengatasi kendala – kendala yang dihadapi selama kerja magang, peserta menemukan beberapa solusi, diantaranya :

1. Peserta mengerjakan setiap tugas yang diberikan sebaik, sebisa, dan secepat mungkin walaupun sering terjadi revisi dan perbaikan karena adanya perubahan.
2. Penganturan CSS yang lebih teliti dan presisi serta *testing* berulang kali untuk memastikan bahwa hasilnya sama meskipun *cross browser*.

U M N
U N I V E R S I T A S
M U L T I M E D I A
N U S A N T A R A