BAB III

PELAKSANAAN KERJA MAGANG

3.1 Kedudukan Dan Koordinasi

Kerja magang selama 2 bulan dilaksanakan di RajaKamar yang berlokasi di Jl. Majapahit No. 16, Jakarta Pusat berkedudukan sebagai *Mobile Developer* untuk *Windows Phone* dibawah bagian IT yang khusus untuk membuat dan mengembangkan aplikasi RajaKamar pada *smartphone,* website, ataupun aplikasi desktop yang dikembangkan RajaKamar bersama dengan partner perusahaan. *Job description Mobile Developer* yaitu membuat aplikasi-aplikasi yang dibutuhkan RajaKamar, sampai dapat di-*submit* di *Play Store, BlackBerry World, Marketplace* dan juga memberikan *update* untuk versi-versi terbaru aplikasi, sehingga aplikasi mengikuti perkembangan tren pasar dan juga meminimalisir terjadinya *bug* pada aplikasi tersebut. Sebelum aplikasi tersebut di-*submit* harus dilakukan testing terelebih dahulu sehingga mengurangi tingkat kegagalan saat proses *submition,* karena banyak nya *error/bug* pada aplikasi tersebut.

3.2 Tugas Yang Dilakukan

Tugas yang dilakukan selama pelaksanaan kerja magang di RajaKamar, pada divisi *Mobile Application* ditulis penulis berdasarkan pengalaman penulis selama mengerjakan proyek pembuatan aplikasi, hal ini disebabkan pembimbing lapangan yang menginginkan aplikasi yang dibuat dapat diselesaikan dan di-*submit* ke

market sehingga dapat diunduh oleh para *costumer* RajaKamar yang menggunakan *Windows Phone*.

Tugas yang dilakukan selama kerja magan ini yaitu :

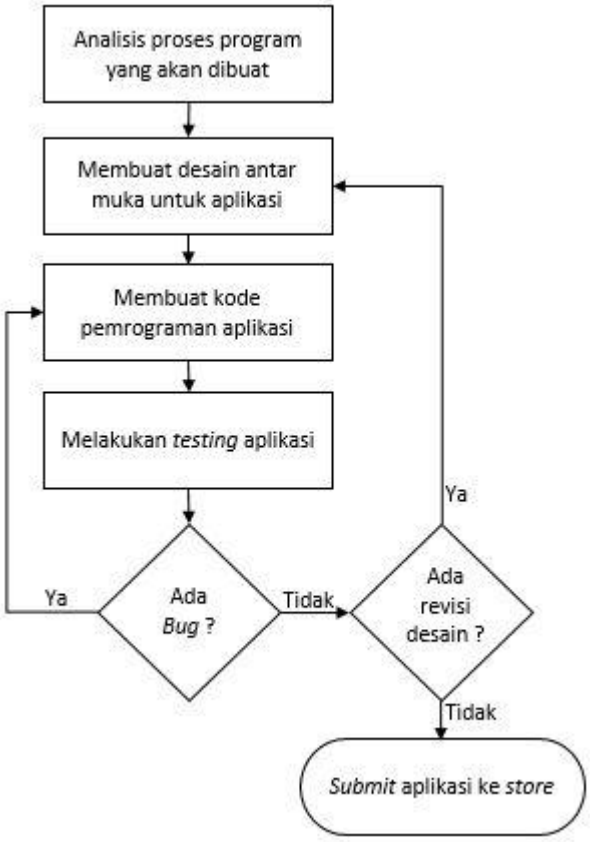
1. Melakukan analisis pada proses bisnis, dalam hal ini adalah proses registrasi, *Login*, hingga proses *booking* hotel.
2. Mempersiapkan perangkat-perangkat pembantu dalam proses pemrograman (*Windows 8* ,*Visual Studio 2012, Windows Phone SDK*)
3. Mempelajari penggunaan JSON.
4. Mempelajari bahasa pemrograman C#.
5. Mempelajari proses berjalannya aplikasi.
6. Memahami proses *Request* dan *Respone Server*.
7. Membuat rancangan antar muka (*interface*) aplikasi dan pemrograman apliaksi (*coding*).
8. Melakukan tes pada aplikasi yang sudah jadi.
9. *Submit* ke *Windows Store.*

3.3 Uraian Pelaksanaan Kerja Magang

3.3.1 Job Cycle

Dalam mengerjakan proyek seorang *mobile developerm* memiliki urutan hal-hal yang perlu dikerjakan sebelum memulai langsung tahap-tahap pengkodean, atau yang biasa disebut *job cycle*. Pada RajaKamar memiliki *job cycle* seperti pada perusahaan pada umumnya dalam hal pembuatan aplikasi, yaitu analisis lalu ke tahap desain. Hal ini dilakukan agar aplikasi yang dibuat memiliki standar yang

sama dengan aplikasi-aplikasi lain yang dibuat oleh RajaKamar, sehingga pengguna tidak perlu beradaptasi terlalu lama jika menggunakan aplikasi



Gambar 3.1 Job Cycle Developer

3.3.2 Proses Pelaksanaan Kerja Magang

a) Observasi berjalan

Kerja magang diawali dengan melakukan beberapa hal yang dibutuhkan untuk membuat aplikasi, yaitu :

1. uji coba melakukan proses pendaftaran, hingga melakukan proses *booking* pada *website* RajaKamar, sebagai pedoman untuk membuat proses-proses apa saja yang dibutuhkan dalam

aplikasi kedepannya. Sehingga memiliki keseragaman proses pada setiap aplikasi RajaKamar, baik *mobile* maupun *desktop*.

2. Aplikasi RajaKamar Menggunakan JSON dalam mengolah data-data yang diterima maupun dikirim. *JSON* merupakan format pertukaran data yang ringan mudah di baca ataupun di buat oleh manusia. Mempelajari cara membuat atau membaca *JSON Array* merupakan hal yang paling mendasar pada pembuatan aplikasi ini.

Gambar 3.2 Contoh JSON Array pada aplikasi

3. Langkah berikutnya untuk membuat aplikasi adalah mempelajari bahasa pemrograman C# , yaitu bahasa pemrograman yang berorientasi pada objek yang dikembangkan oleh Microsoft sebgai bagian dari *.NET Framework.* Penggunaan bahasa pemrograman ini mirip dengan bahasa pemrograman *Visual Basic,* hanya bedanya *C#* lebih berbasis pada bahas pemrograman *C++*.

```
[DataContract]
public class HotelRate
{
    [DataMember]
    public int HotelID { get; set; }
    [DataMember]
    public string HotelName { get; set; }
    [DataMember]
    public string Address1 { get; set; }
    [DataMember]
    public string Address2 { get; set; }
    [DataMember]
    public decimal Rating { get; set; }
    [DataMember]
    public string Curr { get; set; }
    [DataMember]
    public double LowestRate { get; set; }
    [DataMember]
    public string ImageURL { get; set; }
    [DataMember]
    public bool Allotment { get; set; }
    [DataMember]
    public double? Latitude { get; set; }
    [DataMember]
    public double? Longitude { get; set; }
}
```

Gambar 3.3 Penggunaan Class Pada C#

Sebenarnya aplikasi *Windows Phone* sendiri dapat dibuat dengan menggunakan 2 bahasa pemrograman standar yaitu

Visual Basic dan *C#*. Penggunaan *C#*, dikarenakan lebih banyak nya contoh-contoh dari forum *developer* menggunakan bahasa pemrograman *C#* dalam mengembangkan aplikasi, jadi penulis lebih memilih menggunakan bahasa pemrograman ini.

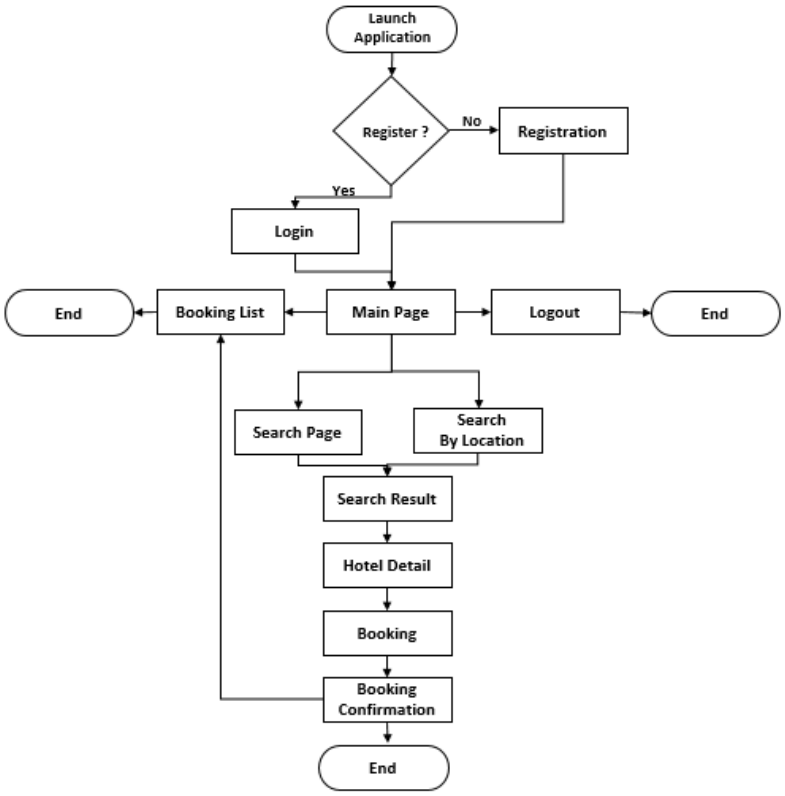
```
143 private void login_Click(object sender, RoutedEventArgs e)
144 {
145     string email = userbox.Text;
146     string pass = Passbox.Password;
147     pass = pass.Replace("\r\n", "");
148 
149     Log_in log = new Log_in();
150     log.CorporateCode = "0";
151     log.Email = email;
152     log.Password = pass;
153     string json = JsonConvert.SerializeObject(log);
154 
155     SendLogin(json);
156 }
```

Gambar 3.4 Contoh Penggunaan C#

4. Pemahaman terhadap proses-proses yang terjadi untuk aplikasi yang kita buat merupakan hal yang penting dan cukup vital. Sehingga apabila semua proses yang ada sudah diketahui maka dalam pembuatan aplikasi dapat lebih mudah dan efisien karena kita membuat aplikasi sesuai jalur proses yang ada dan mengurangi presentase terlewatnya sebuah proses dalam aplikasi. Aplikasi pada RajaKamar ini memang mengikuti proses atau langkah pada proses pemesanan kamar hotel di *website* : **www.rajakamar.com** untuk meningkatkan pengalaman pengguna pada setiap program atau aplikasi RajaKamar.

Setiap aplikasi RajaKamar harus memiliki aplikasi yang seragam, sehingga ada keseragaman di setiap aplikasi yang dikeluarkan oleh RajaKamar. Fitur-fitur utama yang harus dimiliki oleh setiap aplikasi meliputi :

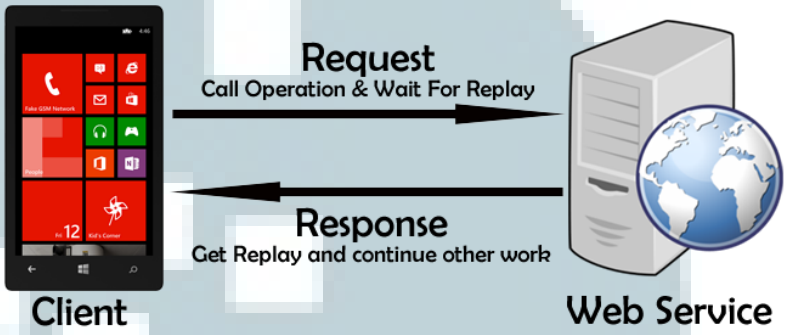
a) *Login, Register*
b) *Forgot Password*
c) *Search Hotel, Search Near By,*
d) *Booking List*
e) *Booking Hotel*. Dalam fitur *Booking Hotel* terdiri dari : *Booking Room* (jenis dan jumlah kamar yang dipesan), *Guest List* (pendaftaran tamu yang akan melakukan *checkin*), *Payment Method* (pemilihan metode pembayaran), dan *Booking Confirmation*.



Gambar 3.5 Proses Aplikasi

Dapat dilihat pada *Gambar 3.5* proses aplikasi dimulai dari menu *Login* yang disertai menu *Registration* dimana pengguna dapat langsung masuk kedalam aplikasi atau harus mendaftar terlebih dahulu. Setelah masuk ke halaman utama aplikasi baru pengguna dapat mulai mencari hotel sesuai kota yang menjadi tempat tujuan pengguna, dan melakukan proses-proses selanjutnya sampai, mendapat konfirmasi pemesanan kamar hotel dari RajaKamar. Setiap konfirmasi pemesanan kamar hotel dapat dilihat juga pada menu *Booking List*.

5. Aplikasi RajaKamar sudah menggunakan *Web Service*, sehingga proses *Request* dan *Response Server* menjadi hal yang harus dipahami terlebiuh dahulu sebelum membuat suatu aplikasi.



Gambar 3.6 Arsitektur Web Service

Pada *Gambar 3.6* dijelaskan proses *Request* dan *Respones Server*. Terlihat jelas bahwa ponsel mengirimkan permintaan kepada server yang akan mengirimkan data-data sesuai yang diminta oleh *client*, sehingga proses dapat segera dilanjutkan kembali.

Web Service yang digunakan dalam pembuatan aplikasi ini merupakan hasil karya dari Denny Wijaya, banyak membantu penulis dalam membuat aplikasi ini. *Web Service* ini sudah dibuat sebagai fondasi dari RajaKamar, sehingga nantinya semua aplikasi mengenai *booking hotel* dilakukan dengan *Web Service* ini. *Web Service* ini menggunakan bahasa pemrograman asp.net.

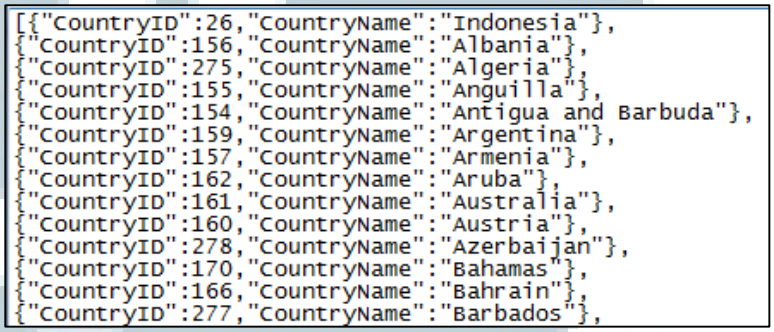
b) Pembuatan aplikasi

Setelah semua persiapan untuk pembuatan aplikasi siap dan penulis sudah memahami JSON dan bahasa pemrograman C#. Aplikasi sudah siap untuk dibuat, baik desain antar muka maupun tahap pemrograman aplikasi tersebut. Dalam pembuatan desain antar muka penulis dibantu oleh tim *designer* yang merupakan tim khusus dari RajaKamar dalam menangani desain. Hal ini diperlukan agar dalam tampilan antar muka aplikasi mengikuti *brand guidelide* dari RajaKamar.

1. Tahap awal dalam pembuatan aplikasi ini adalah, cara menggunakan JSON pada aplikasi berbasis *Windows Phone.* Untuk itu penulis memerlukan cara dasar dalam melakukan *parsing* dari *JSON Array* dan ditampilkan dalam bentuk *string* yang dapat dibaca langsung oleh pengguna dengan mudah dan cepat

Gambar 3.7 Tampilan Setelah Deserialize JSON

Proses penggunaan JSON ini dibagi menjadi 2 bagian yaitu *deserialize* dan *serialize*. Pada gambar diatas merupakan contoh dari penggunaan *deserialize* JSON. Karena masih tahap permulaan jadi penulis masih belajar menggunakan *JSON Array Country* yaitu kumpulan nama-nama negara beserta kode negara yang di-*deserialize* sehingga akan tampil seperti pada gambar diatas.



Gambar 3.8 JSON Array Country

Hasil dari *deserialize JSON* tersebut nantinya dimasukan kedalam sebuah *class* yang berfungsi untuk menampung sementara hasil dari *JSON Array Country* tersebut, jadi penulis menggunakan metode *View-Model* dalam menggunakan *JSON Array.* Penulis menggunakan fasilitas *AutoCompleteBox* yang sudah disediakan oleh *Microsoft* sebgai produsen *Windows Phone*, dan memnggunakan *Class Country* yang berisi *deserialize JSON Array Country* sebagai sumber data dari *AutoCompleteBox* tersebut dengan

menggunakan *DataBinding*, yaitu menggunakan *class* tertentu sebagai sumber utama dari suatu *TextBox*.

2. Setelah pembelajaran pertama dalam melakukan *deserialize JSON* sudah selesai tahap selanjut nya yaitu pembelajaran dalam melakukan *sereliaze JSON*, tahap ini berguna untuk mengirimkan permintaan ke *server*, karena setiap aplikasi di RajaKamar sudah menggunakan *Web Service*, sehingga semua proses di lakukan *server*. Proses *sereliaze* merupakan proses merubah variabel-variabel tertentu ke dalam *JSON Array*. Dalam penggunaan *JSON* baik *serialize* maupun *deserialize* menggunakan *package Newtonsoft.JSON* yang didapatkan dari *NuGet Package* di *Visual Studio 2012*. Paket ini dapat digunakan dengan cara menambahkan *Using Property* : ***using Newtonsoft.Json;***

```
143 private void login_Click(object sender, RoutedEventArgs e)
144 {
145     string email = userbox.Text;
146     string pass = Passbox.Password;
147     pass = pass.Replace("\r\n", "");
148 
149     Log_in log = new Log_in();
150     log.CorporateCode = "0";
151     log.Email = email;
152     log.Password = pass;
153     string json = JsonConvert.SerializeObject(log);
154 
155     SendLogin(json);
156 }
```

Gambar 3.9 Serialize JSON

Gambar di atas menunjukan cara *serialize JSON* pada halaman *Login*. Sama seperti proses *deserialize* pada tahap

ini juga menggunakan *class* sebagai tempat menampung sementara variabel sebelum di ubah menjadi *JSON Array*.

3. Pembuatan halaman *Login* yaitu halaman depan yang berguna bagi pengguna untuk memasukan *username* dan *password* yang sebelumnya sudah dibuat terlebih dahulu, atau apabila belum memiliki akun RajaKamar dapat membuatnya langsung dengan aplikasi ini

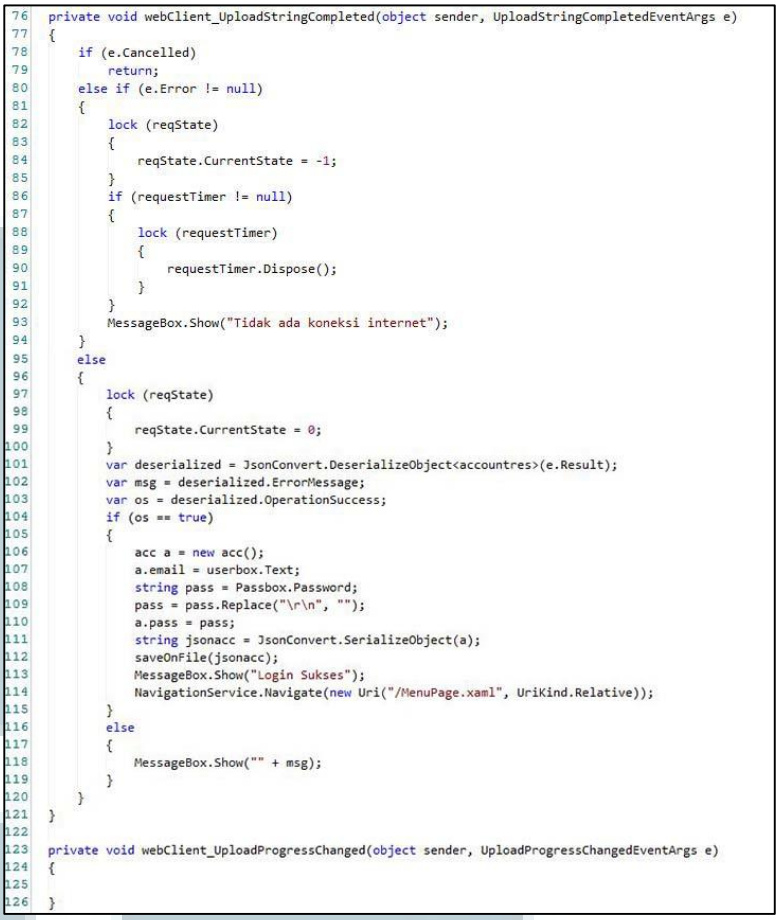
Gambar 3.10 Halaman Utama, Login, Registrasi

Ketiga halaman ini berada dalam *Class Account*. Dan setiap halaman menggunakan fungsi *Web Service* yang meminta dan menerima data-data dari *server* untuk melanjutkan prosesnya.

```
54  private void SendLogin(string json)
55  {
56      System.Threading.Timer requestTimer;
57      WebClient webClient = new WebClient();
58      webClient.Headers[HttpRequestHeader.ContentType] = "application/x-www-form-urlencoded";
59      //var uri = new Uri("[redacted]", UriKind.Absolute);//Test
60      var uri = new Uri("[redacted]", UriKind.Absolute);//Live
61      webClient.Headers[HttpRequestHeader.ContentLength] = json.Length.ToString();
62      webClient.UploadStringCompleted +=
63      new UploadStringCompletedEventHandler(webClient_UploadStringCompleted);
64      webClient.UploadProgressChanged += webClient_UploadProgressChanged;
65      lock (reqState)
66      {
67          reqState.CurrentState = 1;
68          reqState.CurrentWebRequest = webClient;
69      }
70      requestTimer = new System.Threading.Timer(RequestTimedOut, reqState, 60000,
71          System.Threading.Timeout.Infinite);
72      webClient.UploadStringAsync(uri, "POST", json);
73      requestTimer.Change(60000, 0);
74  }
```

Gambar 3.11 Request Server

Proses *request* diawali dengan menentukan metode *request* dan menentukan alamat dari *Web Service*, setelah metode dan alamat yang dituju sudah siap maka harus dibuatkan koneksi untuk menghubungkan antara *client* dan *server* sebgai *header* dari data yang akan dikirimkan ke *server*. Dan data yang dikirimkan ke server berbentuk *JSON Array* dari hasil *serialize JSON* dari variabel *username* dan *password*.



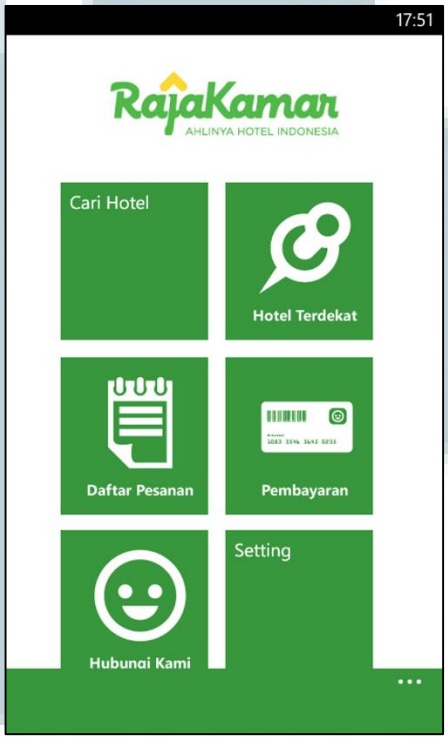
```
76  private void webClient_UploadStringCompleted(object sender, UploadStringCompletedEventArgs e)
77  {
78      if (e.Cancelled)
79          return;
80      else if (e.Error != null)
81      {
82          lock (reqState)
83          {
84              reqState.CurrentState = -1;
85          }
86          if (requestTimer != null)
87          {
88              lock (requestTimer)
89              {
90                  requestTimer.Dispose();
91              }
92          }
93          MessageBox.Show("Tidak ada koneksi internet");
94      }
95      else
96      {
97          lock (reqState)
98          {
99              reqState.CurrentState = 0;
100         }
101         var deserialized = JsonConvert.DeserializeObject<accountres>(e.Result);
102         var msg = deserialized.ErrorMessage;
103         var os = deserialized.OperationSuccess;
104         if (os == true)
105         {
106             acc a = new acc();
107             a.email = userbox.Text;
108             string pass = Passbox.Password;
109             pass = pass.Replace("\r\n", "");
110             a.pass = pass;
111             string jsonacc = JsonConvert.SerializeObject(a);
112             saveOnFile(jsonacc);
113             MessageBox.Show("Login Sukses");
114             NavigationService.Navigate(new Uri("/MenuPage.xaml", UriKind.Relative));
115         }
116         else
117         {
118             MessageBox.Show("" + msg);
119         }
120     }
121 }
122
123 private void webClient_UploadProgressChanged(object sender, UploadProgressChangedEventArgs e)
124 {
125
126 }
```

Gambar 3.12 Response Server

Data yang diterima dari server merupakan *JSON Array* yang harus diolah terlebih dahulu sebelum digunakan. Pengolahan data ini menggunakan fungsi *deserialize JSON* yang dimasukan ke dalam *Class AccountRes.* Apabila proses *Login* berhasil maka pengguna akan langsung diarahkan menuju halaman menu utama dari aplikasi ini dan data mengenai akun (*username* dan *passwor*) disimpan dalam memori ponsel agar saat suatu saat pengguna membuka aplikasi RajaKamar kembali dan belum melakukan *Logout* sebelumnya dapat

langsung melakukan proses *Login* tanpa harusa memasukan kembali *username* dan *password*.

4. Halaman menu utama (*MenuPage*) meruapakan halaman yang menjadi pusat dari aplikasi ini



Gambar 3.13 MenuPage

Isi dari halaman menu ini merupakan proses-proses yang ada dalam aplikasi, ditambah beberapa keterangan mengenai pembuat aplikasi dan tata cara pembayaran pemesanan kamar.

Pembuatan halaman ini menggunakan *HubTile Control*. Fasilitas ini merupakan bagian dari *Windows Phone Toolkit* yang berisi fasilitas-fasilitas tambahan yang disediakan

oleh *Microsofti* tetapi tidak termasuk dalam fasilitas yang ada di *Visual Studio 2012*. *HubTile Control* sendiri merupakan tampilan menu berbentuk kotak-kota seperti lantai 2D yang merupakan tampilan standar bagi *Windows Phone*, yang bernama *Metro User Interface*.

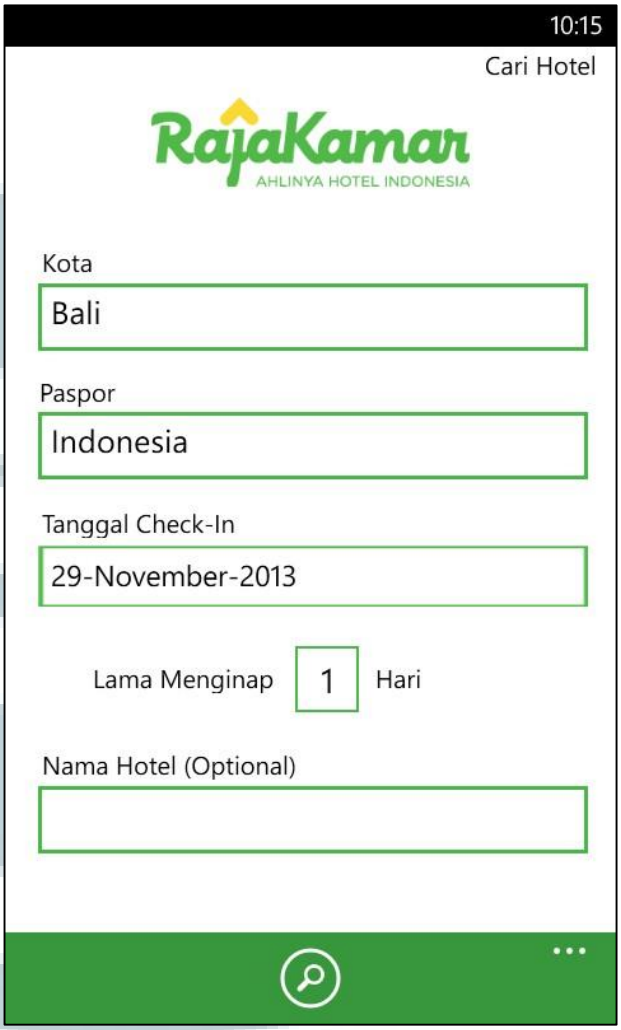
Fungsi-fungsi pada halaman ini hanya sebatas membuka halaman sesuai dengan menu yang dipilih dengan

```
private void search_tap(object sender, System.Windows.Input.GestureEventArgs e)
{
    NavigationService.Navigate(new Uri("/Menu/SearchPage.xaml", UriKind.Relative));
}

private void nearby_Tap(object sender, System.Windows.Input.GestureEventArgs e)
{
    NavigationService.Navigate(new Uri("/Menu/NearBy.xaml", UriKind.Relative));
}

private void BookingList_Tap(object sender, System.Windows.Input.GestureEventArgs e)
{
    NavigationService.Navigate(new Uri("/Menu/BookingList.xaml", UriKind.Relative));
}
```

Gambar 3.14 Fungsi pada MenuPage

Setiap fungsi harus di *declare* terlebih dahulu dan harus digunakan sebagai *Tap_Property* pada desain antar muka. *Tap_Property* merupakan *property* yang berguna untuk menjadikan suatu gambar, teks, bentuk-bentuk yang lain selain *button* memiliki fungsi seperti *button*.

5. Halaman pencarian hotel, adalah fasilitas utama yang harus ada pada aplikasi RajaKamar yang bergerak dalam bidang pemesanan kamar hotel



Gambar 3.15 Halaman Pencarian Hotel

Gambar 3.15 merupakan tampilan pencarian hotel RajaKamar. *Field* Kota merupakan kota yang menjadi destinasi, *field* Paspor merupakan paspor atau kewarganegaraan calon orang yang akan memesan kamar hotel. Kedua *field* tersebut menggunakan *AutoCompleteBox Control* yang merupakan salah satu bagian dari *Windows Phone Toolkit.* Field tanggal *check-in* merupakan tanggal pengguna sudah masuk ke hotel untuk

melakukan *chcek-in*, *field* ini menggunakan *DatePicker Control*, sebagai standar pemilihan tanggal pada *Windows Phone*. *Field* lama menginap merupakan lamanya tamu menginap di hotel dalam satuan malam/hari. Dan *field* terakhir adalah nama hotel, *field* ini diletakkan pada bagian akhir halaman karena merupakan isian opsional jadi tidak masalah apabila di kosongkan, dan *field* ini hanya menggunakan *TextBox* biasa.

Fungsi dibalik halaman ini adalah *Request* dan *Response Server* seperti yang digunakan pada halaman *Login*, namun dilakukan sedikit modifikasi agar proses yang dijalankan dapat sesuai kebutuhan. Modifikasi terletak pada bagian pengolahan *JSON Array* yang didapat dari *server*, pada halaman ini *JSON Array Tersebut* tidak langsung di-*deserialize* tetapi ditampung dulu di dalam *persistant*.

```
var deserialized = JsonConvert.DeserializeObject<HotelSearchResponse>(e.Result);
saveOnPage(e.Result);
var res = deserialized.ResultCode.ToString();
var hr = deserialized.HotelRates;
List<HotelRate> hotelRateList = new List<HotelRate>(hr);
string hrl = JsonConvert.SerializeObject(hotelRateList);

if (deserialized.OperationSuccess)
{
    saveOnFile(hrl);
    ApplicationBar.IsVisible = true;
    NavigationService.Navigate(new Uri("/Hotel/SearchResult.xaml", UriKind.Relative));
}
else
{
    MessageBox.Show("" + deserialized.ErrorMessage);
}
```

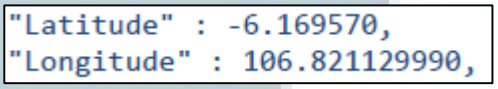
Gambar 3.16 Proses SearchHotel

Perbedaan pada proses ini adalah penggunaan *List<>* dalam proses penerimaan respon *server*. Karena hasil yang

diterima dari server dalam bentuk *JSON List Array* sehingga *class* harus dimasukan ke dalam *List<>*, untuk kemudian di simpan ke dalam *persistance.*

6. Halaman Pencarian hotel terdekat, salah satu keunggulan dari aplikasi *mobile* adalah *positioning,* yaitu koordinat letak dari suatu ponsel dalam satu daerah. Fitur ini dimiliki hampir seluruh *smartphone* pada saat ini, dan dimanfaatkan oleh pengembang aplikasi untuk membuat fitur pencarian berdasarkan posisi pengguna tersebut.

   Aplikasi RajaKamar juga memiliki fitur ini, dimana kita dapat mencari hotel-hotel yang berada di sekitar posisi kita pada saat melakukan pencarian.



Gambar 3.17 Contoh Koordinat Yang Dikirim ke Server

Pada dasarnya proses ini mirip dengan proses yang dilakukan oleh metode pencarian biasa tetapi perbedaannya hanya data yang dikirim ke server berupa koordinat posisi seperti pada *Gambar 3.17*, yang akan diolah secara otomatis oleh server sebagai bahan acuan untuk memberi balasan hotel yang kurang lebih berjarak 5 km dari posisi pengguna saat itu.

Koordinat yang didapat merupakan fasilitas yang ada pada ponsel dan masing-masing *operation system* memiliki cara yang berbeda dalam penggunannya

```
public async void GetLatitude()
{

    if ((bool)IsolatedStorageSettings.ApplicationSettings["LocationConsent"] != true)
    {
        // The user has opted out of Location.
        return;
    }

    Geolocator geolocator = new Geolocator();
    geolocator.DesiredAccuracyInMeters = 50;

        Geoposition geoposition = await geolocator.GetGeopositionAsync(
            maximumAge: TimeSpan.FromMinutes(5),
            timeout: TimeSpan.FromSeconds(10)
            );


        latitude = geoposition.Coordinate.Latitude.ToString();
        longitude = geoposition.Coordinate.Longitude.ToString();


}
```

Gambar 3.18 Cara Mendapatan Koordinat Posisi

Proses ini sudah diberikan di contoh yang di berikan dari *MSDN* yaitu situs jaringan yang dibuat *Microsoft* untuk para pengembang aplikasi-aplikasi *Windows* baik *Windows Phone, Windows 8, Windows Azure,* atau bahkan *Microsoft Office* yang membutuhkan pengembangan lagi.

Gambar 3.19 Tampilan Menu Search By Location

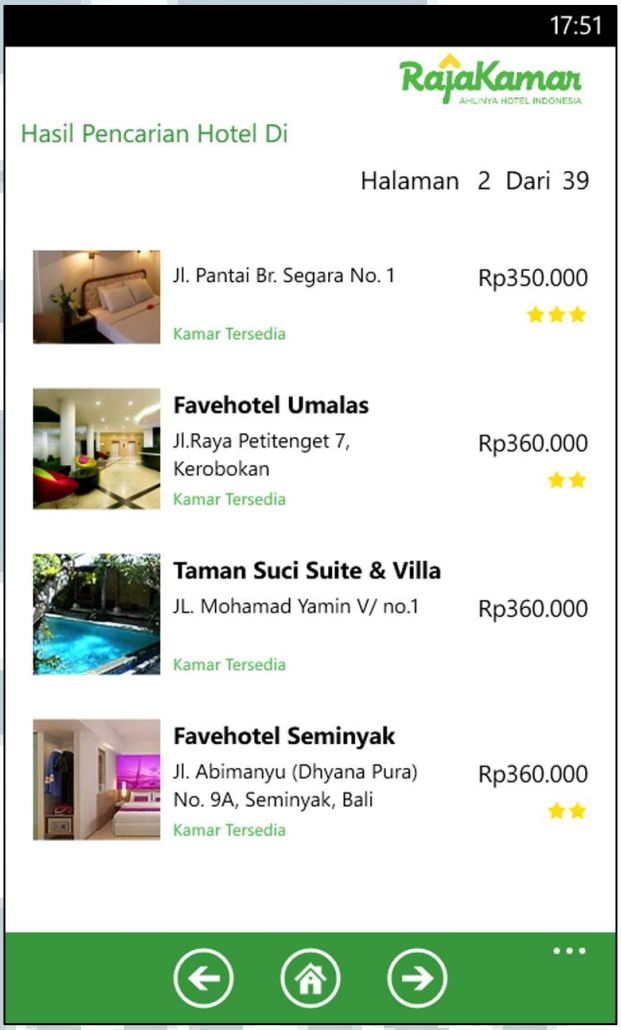
Pada halaman ini pengguna hanya perlu memasukan paspor, tanggal *check-in*, dan lama menginap, sama seperti menu pencarian biasa. Seperti yang sudah dijabarkan di atas jika halaman ini mengirimkan koordinat posisi yang didapat langsung pada saat aplikasi sudah berjalan, karena secara *default* aplikasi ini menggunakan *Location Positioning* pada ponsel pengguna, sehingga *latitude* dan *longitude* ponsel sudah langsung didapatkan.

7. Halaman *Booking List* merupakan halaman yang menampilkan daftar –daftar pesanan hotel yang pernah dilakukan. Halaman ini secara otomatis akan mengurutkan pesanan berdasarkan status pesanan (*confirm*, *cancel*).

   Data yang ditampilkan berasal dari *server* dan diterima dalam bentuk *JSON List.* Variabel yang ditampilkan adalah kode booking, rincian hotel, lama menginap, total biaya, dan status.

Gambar 3.20 Tampilan Menu Booking List

8. Halaman *SearchResult* merupakan kelanjutan dari tahap pencarian hotel baik dengan pencarian biasa ataupun pencarian berdasarkan koordinat, kedua proses tersebut akan menuju ke halaman *Search Result.*



Gambar 3.21 Tampilan SearchResult

Pada halaman proses *request* ataupun *response server*, terjadi saat perpindahan halaman pencarian. Sedangkan data pada saat halaman ini terbuka didapat dari hasil pencarian di menu sebelumnya, dan sudah disimpan dalam *persistance*. Data

daftar hotel tersebut masih dalam *JSON List* dan pada halaman ini data tersebut diolah untuk kemudian di-*parsing* hingga menjadi bagian-bagian yang dikelompokan berdasarkan hotel.

Ada 3 tombol utama yang berada dalam halaman ini. Fungsi dari tombol yang bergambar anak panah ke kiri dan ke kanan adalah bergeser halaman, panah ke kiri, ke halaman sebelumnya, sedangkan panah ke kanan, ke halaman berikutnya. Tombol bergambar rumah, berfungsi untuk kembali ke halaman menu utama.

Tombol bergambar anak panah tersebut memiliki fungsi ***Paging***, yaitu funsi untuk berpindah halaman, dan fungsi ini menggunakan *request* dan *response server* untuk mendapatkan hasilnya. Data yang dikirimkan ke *server* adalah kode pencarian dan juga halaman yang dituju.

```
private void Next_Click(object sender, EventArgs e)
{
        int page = Convert.ToInt32(pagenumber.Text);
        int total = Convert.ToInt32(totalpage.Text);
        if (page <= total)
        {
            page = page + 1;
            pagenumber.Text = page.ToString();

            userIdentity user = new userIdentity();
            user.CorporateCode = "0";
            user.Email = emailID;
            user.Password = passID;

            SearchPage sp = new SearchPage();
            sp.userIdentity = user;
            sp.ResultCode = resultcode;
            sp.Page = pagenumber.Text;
            string json = JsonConvert.SerializeObject(sp);
            SendPaging(json);
        }
```

Gambar 3.22 Fungsi Paging

Dan untuk fungsi memilih hotel terdapat fungsi **Pilih** yang mengambil variabel *HotelID* untuk melakukan permintaan ke *server*.

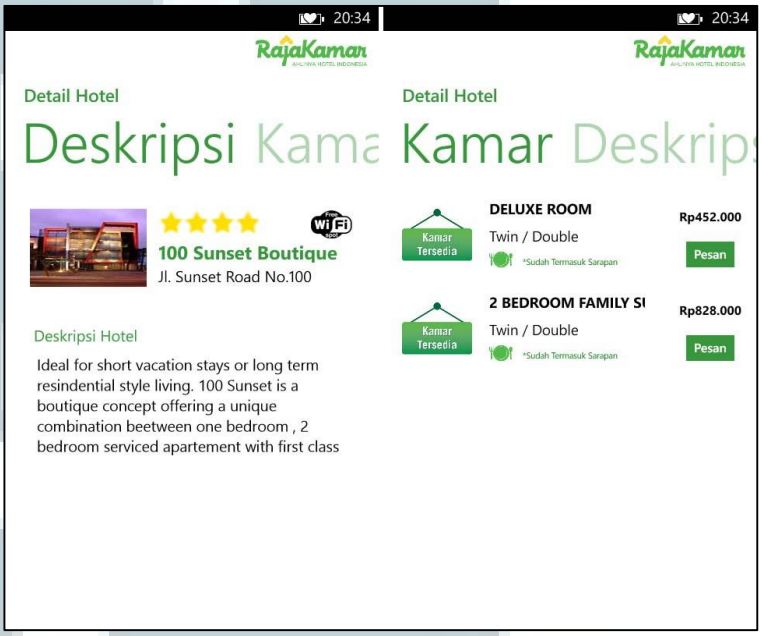
```
private void Pilih(object sender, System.Windows.Input.GestureEventArgs e)
{
    var selectedElement = e.OriginalSource as FrameworkElement;
    if (selectedElement != null)
    {
        var selectedData = selectedElement.DataContext as ShowHotelList;
        if (selectedData != null)
        {
            userIdentity user = new userIdentity();
            user.CorporateCode = "0";
            user.Email = emailID;
            user.Password = passID;

            HotelRatesPost HRP = new HotelRatesPost();
            HRP.CheckIn = checkin;
            HRP.HotelID = selectedData.HotelID;
            HRP.Nationality = nationality;
            HRP.Nite = nite;
            HRP.userIdentity = user;

            string json = JsonConvert.SerializeObject(HRP);
            saveOnFileRoomBooking(json);
            ApplicationBar.IsVisible = false;
            SendHotelRatePost(json);
            Searching();

            listHotel.IsEnabled = false;
        }
    }
}
```

Gambar 3.23 Fungsi Pilih

Pada gambar fungsi **Pilih** terlihat proses pembuatan *serialize JSON* dan proses pemanggilan fungsi *request* ke *server*. Hasil dari permintaan ke *server* tersebut akan disimpan ke dalam *persistance* untuk di jalankan pada langkah berikutnya.

9. Detail Hotel, setelah pengguna memilih hotel yang diinginkan akan diarahkan ke halaman *HotelDetail*. Pada menu ini pengguna akan diberikan keterangan-keterangan mengenai fasilitas-fasilitas hotel dan beberapa gambaran

mengenai hotel tersebut. Dan keterangan yang harus ada pada halaman ini mengenai tipe kamar dan keterangan harganya, untuk menjadi bahan acuan pengguna untuk menentukan pilihan kamar yang akan di pesan.



Gambar 3.24 Tampilan Detail Hotel

Dalam halaman detail hotel ini, terdapat 2 bagian utama, yaitu Deskripsi dan Kamar. Bagian Deskripsi adalah bagian yang mendeskripsikan hotel yang dipilih, dan fasilitas-fasilitas yang ada di hotel tersebut termasuk keterangan mengenai tingkat atau bintang hotel tersebut, jadi secara garis besar bagian ini adalah bagian yang menggambarkan mengenai hotel secara keseluruhan. Bagian Kamar merupakan bagian yang menampilkan tipe-tipe kamar pada hotel tersebut beserta harga dan paket sarapan atau tidak.

Proses ini tidak menggunakan fungsi *request* server hanya menampung data nya pada *persistance* untuk kemudian dipakai untuk melanjutkan proses selanjutnya.

10. Setelah pengguna memilih kamar, selanjutnya akan di arahkan ke halaman pemesanan jumlah kamar pada halman *BookingRoom*.

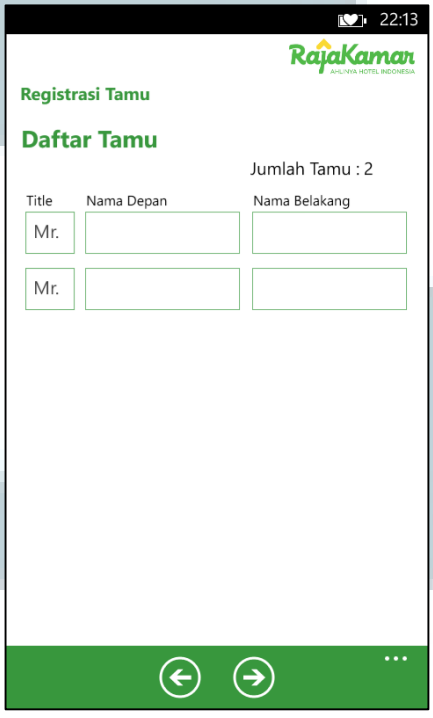
Gambar 3.25 Tampilan Booking Detail

Pada halaman ini akan di tampilkan rincian pesanan hotel dan kamar yang sudah dipilih pengguna sebelumnya. Dan pada

halaman ini pengguna diminta untuk memilih jumlah kamar yang ingin dipesan, dan juga pesanan khusus apabila dibutuhkan. Pada kotak jumlah pesanan kamar, penulis menggunakan *ListPicker Control*, perangkat ini berguna seperti *dropdown list* pada aplikasi-aplikasi lainnya, jadi pengguna dapat memilih daftar-daftar yang disediakan.

Setelah halaman ini selesai terisi, semua data-data akan disimpan di *persistance*, sebagai data yang nantinya akan dikirimkan ke *server*.

11. Setelah rincian pesanan kamar selesai, dilanjutkan dengan pengisian identitas tamu pada halaman *GuestDetail* yang nantinya akan melakukan *check-in* ke hotel tersebut.



Gambar 3.26 Tampilan Registrai Tamu

Halaman ini merupakan halaman yang harus diisi oleh tamu yang akan memesan kamar melalui RajaKamar, karena data yang diisikan akan disesuaikan dengan identitas yang nantinya melakukan proses *check-in* di hotel tujuan.

Jumlah tamu yang didaftarkan ini berdasarkan jumlah kamar yang di pesan, jadi apabila pengguna memesan 2 kamar, maka dia harus mengisikan 2 nama tamu untuk melakukan *check-in*.

12. Proses terakhir dalam tahap booking adalah pemilihan metode pembayaran, yang terdapat dalam halaman *PaymentDetail*. Pada tahap ini pengguna diwajibkan untuk memilih metode pembayarn *KlikBCA* atau *Credit Card*.

Gambar 3.27 Tampilan Metode Pembayaran

RajaKamar menggunakan 2 metode pembayaran untuk aplikasi *mobile*, yaitu *KlikBCA* dan *Credit Card*. Apbila pengguna memilih pilihan *KlikBCA* secara otomatis akan muncul kotak dialog untuk memasukan *KlikBCA ID*, sehingga pesanan dapat langsung dibayarkan di bagian *e-commerce* pada *KlikBCA*. Untuk pemilihan metode pembayaran *Credit Card* pengguna akan secara otomatis diarahkan ke halaman pembayaran menggunakan kartu kredit setelah mendapatkan konfirmasi pemesanan hotel dari RajaKamar.

13. Bagian terakhir dalam proses pemesanan adalah konfirmasi pemesanan kamar di halaman *ConfirmationPage*. Pada halaman ini semua rincian mengenai pemesanan hotel tertera, dan sudah masuk dalam reservasi kamar di hotel hanya tinggal melakukan pembayaran dan semua proses pemesanan selesai, jadi penggunan dapat langsung datang ke hotel pada tanggal yang sudah di pesan dan melakukan *check-in.*

c) Tahap *testing* aplikasi

Tahap ini merupakan tahap yang penting dalam pengembangan aplikasi sebelum dilakukan *submition* ke *store*. Uji coba aplikasi dengan menjalankan seluruh proses-prose yang ada di aplikasi bertujuan untuk mengetahui kekurangan-kekurangan dalam

pembuatan aplikasi, adanya *bugs/error,* dan juga untuk mendesain ulang tampilan antar muka aplikasi ini.

Kekurangan yang terjadi dalam aplikasi dalam berbentuk kesalahan kata-kata atau kurangnya panduan untuk pengguan dalam menjalankan aplikasi ini, sehingga membuat pengguna kesulitan menjalankan aplikasi. Karena terkadang *programmer* membuat aplikasi yang hanya dapat dimengerti oleh tetapi sulit dimengerti oleh orang lain, hal ini lah yang harus diperhatikan pada saat tahap uji coba.

Dengan menjalankan seluruh proses aplikasi, bertujuan untuk memeriksa kembali adanya *error/bugs* dalam aplikasi ini. Biarpun setiap satu proses selesai di program dan dapat berjalan lancar saat dilakukan *debug*, namun saat aplikasi dijalankan dalam satu kesatuan ada kemungkinan satu bagian proses tersebut mengalami *error/bugs*. Dalam pembuatan aplikasi ini penulis mengalami terjadi nya *error/bugs* yang ditandai dengan *force close* pada saat aplikasi dijalankan, karena terjadi hal seperti demikian maka harus dilakukan pemeriksaan langkah demi langkah setiap proses, sehingga dapat diketahui bagian yang mengalami *error/bugs* dan dapat segera dicari solusinya.

Tidak dapat disangkal lagi, banyak pengguna yang menilai suatu aplikasi dari tampilan antar muka yang menarik dan mudah dipahami oleh pengguna. Tahap uji coba merupakan salah satu tahap

penilaian dasar keberhasilan dari desain antar muka yang sudah dibuat pada aplikasi. Pada saat melakukan uji coba, aplikasi yang dibuat penulis dicoba oleh orang bagian *Marketing Communication*, untuk mendapat masukan-masukan terkait desain antar muka. Dengan dilakukan uji coba pada orang di luar bagian IT dapat menjadi tolak ukur keberhasilan desain antar muka, karena orang-orang tersebut tidak termasuk dalam tim pengembangan aplikasi, yang mengetahui secara langsung kegunaan pada tiap tombol atau fasilitas yang ada di aplikasi.

d) Tahap dokumentasi

Tahap ini berguna untuk kedepannya saat terjadi pengembangan aplikasi, namun pengembangan tersebut tidak dilakukan oleh pembuat aplikasi sebelumnya. Sehingga orang yang bertanggung jawab dalam pengembangan aplikasi dapat dengan mudah memahami arsitektur dari aplikasi tersebut, karena setiap *function* dan variabel diberikan keterangan agar dapat mudah dimengerti.

Dalam melakukan dokumentasi aplikasi penulis menggunakan *GitHub* untuk membantu pembuatan dokumentasi. Setiap perubahan-perubahan yang dilakukan selama pembuatan aplikasi dicatat dan diberikan keterangan mengenai perubahan tersebut dan orang yang bertanggung jawab atas perubahan aplikasi. Hal ini bertujuan agar

adanya pencatatan yang jelas setiap aplikasi memiliki penambahan atau pengurangan fitur. Sehingga saat mendapat *feedback* dari pengguna dari fitur-fitur tersebut ada pencatatannya dan mudah diperbaiki atau dikembangkan.

Aplikasi *GitHub* ini juga berguna sebagai alat kontrol. Karena setiap perubahan tercatat dan juga ditulis orang yang bertanggung jawab atas perubahan tersebut, sehingga perubahan tersebut dapat dilakukan pemantauan. Dalam proses pembuatan aplikasi, perkembangan pekerjaan yang penulis lakukan dapat langsung di kontrol oleh pembimbing dengan memeriksa *GitHub*, karena komputer penulis dan pembimbing sudah saling dihubungkan.

Selain menjadi alat bantu dalam hal kontrol, dan pengembangan, tahap dokumentasi dapat juga menjadi salah satu salat bantu dalam melakukan *backup* aplikasi, karena semua kegiatan yang dilakukan pada aplikasi dilakukan pencatatan.

e) Submition* ke *store

Setelah aplikasi siap dipakai dan siap untuk *publish* tahap selanjutnya adalah *submition* yaitu proses mendaftarkan aplikasi kita untuk masuk ke dalam *store*.

Format yang harus di upload ke dalam *Windows Store* untuk aplikasi *Windows Phone* adalah format XAP. File dengan format ini

didapatkan pada saat, apliaksi sedang di-*release*, maka secara otomotis *file* XAP ini akan muncul. Selain itu untuk masuk ke *store* setiap aplikasi harus memiliki logo aplikasi berukuran sedang dan kecil. Dan pada saat di upload aplikasi harus disertai dengan deskripsi aplikasi yang menggunakan bahasa, yang sesuai dengan regional aplikasi yang dipilih. Jadi apabila target *store* adalah Indonesia, maka deskripsi yang digunakan lebih baik menggunakan bahasa aplikasi.

Sebelum aplikasi dapat di download oleh para pengguna *Windows Phone,* aplikasi-aplikasi yang di unggah oleh pengembang harus melalui tahap uji coba terlebih dahulu dari pihak *Microsoft*, hal ini bertujuan untuk memberi standar aplikasi yang dapat masuk ke dalam ponsel *Windows Phone* pengguna. Apabila aplikasi belum lolos verifikasi dari pihak *Microsoft* pembuat aplikasi akan mendapat *email* yang berisi kekurangan-kekurangan dari aplikasi yang dibuat.

3.3.3 Kendala Dalam Pelaksanaan Kerja Magang

Dalam melakukan kerja magang di RajaKamar untuk membuat aplikasi berbasi *Windows Phone,* penulis menemukan beberapa kendala. Kendala yang penulis dapatkan selama kerja magang ini, yaitu :

1. Aplikasi yang dibuat menggunakan bahasa pemrograman C#, sedangkan penulis belum mendapatkan pelajaran dengan bahasa pemrograman ini di kampus.

2. Semua proses dalam aplikasi menggunakan JSON, dan di kampus belum ada mata kuliah yang menyinggung penggunaan JSON.
3. Aplikasi ini adalah aplikasi *Windows Phone* pertama yang penulis buat, sehingga masih harus belajar dari awal.
4. Pembuatan fungsi *request* dan *get response server*, penulis mengalami masalah dalam melakukan koneksi ke web service.

3.3.4 Solusi Atas Kenadala yang Ditemukan

Beberapa kendala yang penulis temukan dalam pembuatan program, menghambat penulis dalam pembuatan aplikasi ini. Solusi di dapat dari berbagai sumber, dari blog, forum, dan karyawan senior IT.

1. Masalah penggunaan bahasa pemrograman C#, menghambat penulis dalam membuat aplikasi ini dalam beberapa hari, tetapi penulis belajar secara otodidak dan membaca forum seperti *stackoverflow, msdn* untuk membantu pembelajaran.
2. JSON meruapakan hal yang asing bagi penulis. Kendala di JSON ini diselesaikan dengan adanya panduan dari Json.CodePlex.com sebagai pembuat *Newtonsoft.Json* yang penulis gunakan dalam melakukan proses terhadap JSON Array yang ada.
3. Sebagai aplikasi pertama yang dibuat penulis, tentunya membuat canggung dalam pengerjaan aplikasi. Hal ini dapat terselesaikan dengan sendirinya karena penulis sudah mulai biasa dalam membuat aplikasi dan penggunaan-penggunaan fasilitas pada *Visual Studio 2012*.

4. Kendala dalam membuat fungsi *request* dan *get response server*, dapat diselesaikan dengan bantuan dari karyawan senior IT, yang bernama Denny Wijaya. Dia merupakan *expert* dalam *framework .Net* di RajaKamar. Masalah terselesaikan saat fungsi *request* dan *get response server* dibuatkan oleh beliau.