SKRIPSI

PENGEMBANGAN APLIKASI PENCARI RUTE KENDARAAN UMUM UNTUK WINDOWS PHONE

YOHAN

NPM: 2011730048

PROGRAM STUDI TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INFORMASI DAN SAINS
UNIVERSITAS KATOLIK PARAHYANGAN
2014

UNDERGRADUATE THESIS

DEVELOPMENT APPLICATION PUBLIC TRANSPORT ROUTE SEARCH FOR WINDOWS PHONE

YOHAN

NPM: 2011730048

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY AND SCIENCES
PARAHYANGAN CATHOLIC UNIVERSITY
2014

LEMBAR PENGESAHAN

PENGEMBANGAN APLIKASI PENCARI RUTE KENDARAAN UMUM UNTUK WINDOWS PHONE

YOHAN

NPM: 2011730048

Bandung, 1 Juli 2014

Menyetujui,

Pembimbing Tunggal

Pascal Alfadian, M.Com.

Ketua Tim Penguji	Anggota Tim Penguji
Thomas Anung Basuki, Ph.D.	Dr. rer. nat. Cecilia Esti Nugraheni

Mengetahui,

Ketua Program Studi

Thomas Anung Basuki, Ph.D.

PERNYATAAN

Dengan ini saya yang bertandatangan di bawah ini menyatakan bahwa skripsi dengan judul:

PENGEMBANGAN APLIKASI PENCARI RUTE KENDARAAN UMUM UNTUK WINDOWS PHONE

adalah benar-benar karya saya sendiri, dan saya tidak melakukan penjiplakan atau pengutipan dengan cara-cara yang tidak sesuai dengan etika keilmuan yang berlaku dalam masyarakat keilmuan.

Atas pernyataan ini, saya siap menanggung segala risiko dan sanksi yang dijatuhkan kepada saya, apabila di kemudian hari ditemukan adanya pelanggaran terhadap etika keilmuan dalam karya saya, atau jika ada tuntutan formal atau non-formal dari pihak lain berkaitan dengan keaslian karya saya ini.

Dinyatakan di Bandung,
Tanggal 1 Juli 2014

Meterai

Yohan
NPM: 2011730048

ABSTRAK

Sedang Dalam Pembuatan.

Kata-kata kunci: Rute, Kendaraan Umum, Windows Phone

ABSTRACT

Under Construction.

Keywords: Route, Public Transport, Windows Phone

Dipersembahkan untuk diri sendiri

KATA PENGANTAR

Puji Syukur penulis kepada Tuhan yang telah memberikan rahmatnya sehingga penulis dapat menyelesaikan skripsi yang berjudul "Pencari Rute Kendaraan Untuk Windows Phone"

Bandung, Juli 2014

Penulis

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

BAB 1

PENDAHULUAN

Pada Bab satu akan dibahas pendahuluan dari penelitian yang dilakukan. Bab satu terbagi dalam delapan *sub bab*, yaitu *latar belakang*, *rumusan masalah*, *tujuan*, *batasan masalah*, *ruang lingkup masalah*, *metode penelitian*, *teknik pengumpulan data*, dan *sistematika penulisan*.

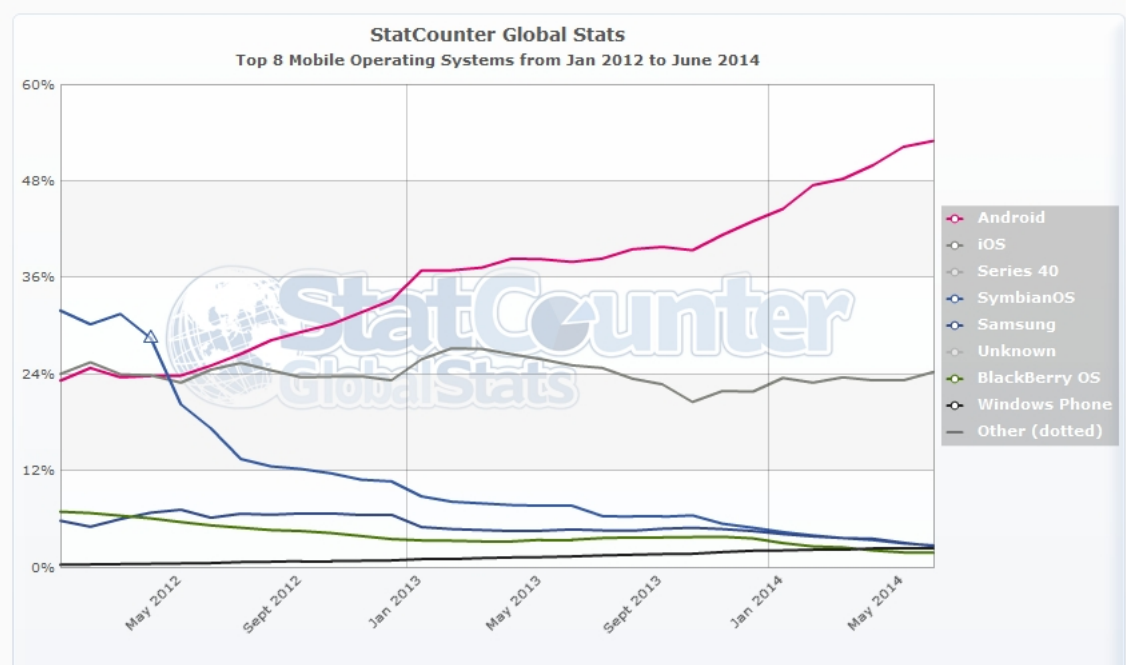
1.1 Latar Belakang

Transportasi menjadi bagian yang penting bagi manusia di saat penelitian ini dilakukan. Ada dua jenis transportasi bagi seseorang yaitu kendaraan umum dan kendaraan pribadi. Kenyataannya pada saat penelitian ini dilakukan banyak yang lebih memilih kendaraan pribadi dibanding kendaraan umum. Maraknya penggunaan kendaraan pribadi dan penambahan jalur kendaraan yang tidak sebanding banyaknya kendaraan menimbulkan kemacetan. Maraknya penggunaan kendaraan pribadi dikarenakan kurang nyamannya kendaraan umum dan kesulitan dalam menentukan kendaraan umum yang harus dinaiki. Banyaknya rute kendaran umum membuat orang kebingungan dalam memilih kendaraan umum menuju lokasi yang diinginkan. Seseorang cenderung malas untuk bertanya dan mencari rute yang efisien. Karena hal tersebut, seseorang lebih memilih menggunakan kendaraan pribadi ketimbang kendaraan umum.

Ide pembuatan aplikasi yang memudahkan seseorang dalam menentukan rute kendaraan umum sudah lebih dulu ada yang dikenal dengan nama Kiri. Kiri dibuat dengan latar belakang tiga masalah besar yaitu pemanasan global, kemacetan, dan harga bahan bakar minyak yang tinggi[1]. Meskipun Kiri pertama dibuat di web tetapi Kiri dapat dimanfaatkan untuk pencarian kendaraan selain di web. Pemanfaatan Kiri tersebut dalam mencari rute kendaraan umum dengan mengggunakan Kiri API.

Pesatnya perkembangan teknologi sekarang ini mendorong perkembangan perangkat bergerak (*mobile*). Perangkat bergerak kian digemari orang-orang terutama di Indonesia. Salah satu yang menarik perhatian adalah Windows Phone 8 yang dibuat Microsoft. Antarmuka Windows Phone 8 yang disebut *Metro* cukup menarik dan mudah digunakan. Meskipun jumlah penggunanya masih belum sebanyak pengguna Android dan IOS tapi jumlah penggunanya terus naik di tahun 2014 ini. Statistik peningkatan jumlah pengguna di Windows Phone dari tahun 2012 hingga 2014 dapat dilihat pada gambar 1.1.

[1] http://static.kiri.travel/en-about/



Gambar 1.1:
Statistik Pengguna Windows Phone[2]

Berdasarkan hal tersebut, penulis mencoba mengembangakan aplikasi Pencarian Rute Kendaraan Umum di Windows Phone dalam tugas akhir ini. Aplikasi yang penulis kembangan akan memungkinkan pengguna menemukan rute kendaraan umum untuk sampai di tujuan. Untuk memudahkan pengguna, penulis akan menampilkan dalam 2 bentuk yaitu peta dan daftar.

1.2 Rumusan Masalah

Sehubung dengan latar belakang diatas timbul permasalahan sebagai berikut:

- Bagaimana membuat aplikasi di Windows Phone?
- Bagaimana mengintegrasikan Kiri API dengan aplikasi pencari rute kendaraan umum di Windows Phone?

1.3 Tujuan

Berdasarkan rumusan masalah pada sub bab 1.2, maka tujuan dari pembuatan tugas akhir ini adalah:

- Mempelajari cara pembuatan perangkat linak di Windows Phone lalu mengembagan aplikasi yang akan dibuat.
- Membuat aplikasi di di Windows Phone yang memanfaatkan Kiri API.

[1] http://gs.statcounter.com/#mobile_os-ww-monthly-201201-201406

1.4 Batasan Masalah

Ruang lingkup pengembangan aplikasi Pencari Rute Kendaraan untuk Windows Phone ini dibatasi hal berikut:

- Aplikasi ini akan berjalan di sistem operasi Windows Phone 8.
- Aplikasi ini membutuhkan koneksi internet.
- Aplikasi ini akan menampilkan rute jalur angkot, bus umum dan travel di tiga kota besar yaitu Bandung, Jakarta, dan Surabaya.

1.5 Metode Penelitian

Metode Penelitian yang penulis gunakan dalam membuat tugas akhir ini adalah sebagai berikut:

- Melakukan studi pustaka mengenai XAML, kontrol dan navigasi di Windows Phone, Peta di Windows Phone, GPS di Windows Phone dan Kiri API.
- Melakukan analisis terhadap aplikasi lain yang menggunakan Kiri API.
- Melakukan analisis terhadap dasar teori untuk pembangunan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Melakukan perancangan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Implementasi dari aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Menguji aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- Membuat kesimpulan.

1.6 Sistematika Penulisan

Bab 1 membahas latar belakang masalah, rumusan masalah, tujuan penulisan tugas akhir, batasan masalah, ruang lingkup masalah, metode penelitian, dan teknik pengumpulan data tugas akhir ini.

Bab 2 membahas tentang teori-teori yang akan digunakan dalam tugas akhir ini. Bahasan yang dijelasakan pada bab ini adalah Windows Phone dan Kiri API. Teori Windows Phone yang dijelaskan meliputi lingkungan kerja, xaml, kontrol terhadap ponsel, siklus hidup aplikasi, peta di Windows Phone, lokasi, dan memanfaatkan sumber data. Teori Kiri API yang dijelaskan meliputi *routing web service*, *web service* pencarian lokasi, dan *web service* menemukan transportasi terdekat.

Bab 3 membahas tentang analisis pembangunan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone. Pada Bab 3akan dibahas mengenai analisis kebutuhan aplikasi, analisis kontrol yang dipakai, analisis terhadap siklus hidup aplikasi, analisis terhadap siklus hidup aplikasi, analisis peta, analisis memanfaatkan sumber data, analisis Kiri API, diagram use case, dan diagram kelas.

BAB 2

DASAR TEORI

Bab ini berisi dasar teori dari pembangunan Aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone. Beberapa teori yang dibahas dalam bab ini adalah XAML, kontrol terhadap ponsel, siklus hidup Windows Phone, peta, *Global Positioning System* di Windows Phone, pemanfaatan sumber data, dan Kiri API.

2.1 Windows Phone

Windows Phone merupakan sistem operasi untuk perangkat bergerak yang dikembangkan Microsoft[1]. Pengembangan aplikasi Windows Phone membutuhkan Windows Desktop 8 sebagai media pengembangan. Bahasa pemrograman yang digunakan untuk membuat perangkat lunak di Windows Phone yaitu C# atau Visual Basic.

Pada sub bab 2.1.1 sampai 2.1.7 akan membahas pemrograman di Windows Phone. Pembahasan akan dimulai dengan apa itu Windows Phone dan fitur di Windows Phone yang akan digunakan dalam pembangunan perangkat lunak Pencarian Rute Kendaraan di Windows Phone.

2.1.1 Lingkungan Kerja

Microsoft .NET framework merupakan sebuah perangkat lunak yang dibangun untuk membantu dalam pembangunan aplikasi di Windows, Windows Phone, Windows Server, and Microsoft Azure[1]. Microsoft .NET framework terdiri dari runtime bahasa umum dan perpustakaan kelas .NET Framework, yang meliputi kelas, interface, dan jenis nilai yang mendukung berbagai teknologi. Microsoft .NET Framework menyediakan lingkungan yang mudah dikelola, pengembangan yang disederhanakan, dan integrasi dengan berbagai bahasa pemrograman, termasuk Visual Basic dan Visual C#.

Seperti yang telah disebutkan ada dua bahasa pemrograman dalam .NET Framework yang dipakai dalam pembangunan aplikasi di Windows Phone 8 yaitu Visual Basic dan Visual C#. Untuk masalah kehandalan keduanya menawarkan kehandalan yang baik. Kelebihan dari Visual Basic adalah bahasa pemrograman berorientasi objek yang kuat dan memiliki banyak pengenbangan fitur di inheritance, polymorphism, interfaces, and overloading[1].Kelebihan dari C# yang merupakan pengembangan dari C/C++ adalah sederhana, modern, aman dan berorientasi objek[1]. Satu hal yang dirasakan penulis adalah kenyamanan ketika memilih bahasa .NET tersebut. Akan lebih mudah bagi developer yang menggunakan Visual Basic 6.0 untuk menggunakan Visual

[1] en.wikipedia.org/wiki/Windows_Phone

Basic .NET. Tetapi bagi developer yang menggunakan C++ atau java sebelumnya akan lebih mudah menggunakan C#.

2.1.2 XAML

Extensible Application Markup Language (XAML) merupakan bahasa deklaratif yang dipakai untuk membuat antarmuka aplikasi. XAML merupakan bahasa yang digunakan untuk membuat antarmuka di Windows Phone 8. Pada dasarnya penggunaan XAML sama dengan HTML pada pembuatan antarmuka web. XAML dapat menginisialisasi objek dan mengatur properti untuk menunjukan hubungan antar objek.

Untuk aturan penulisan sintak XAML didasarkan pada XML. Setiap XAML Windows Runtime menggunakan konvensi bahasa XAML dan ditulis pada *namespace* yang ditandai dengan prefix x sebagai elemen paling atas. Setelah itu di baris ke dua dimulai dengan xmlns diikuti titik dua, lalu nama dari *namespace*, diikuti tanda sama dengan dan *path* yang merepresentasikan *namespace*. Prefix x pada XAML mengandung beberapa struktur program yang sering kita gunakan yaitu :

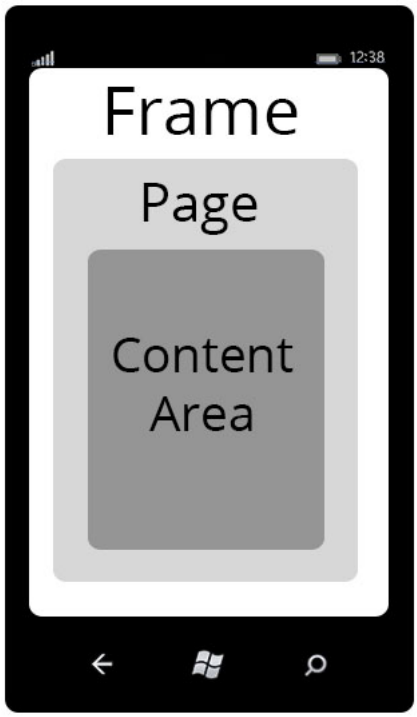
- x:Key : sebuah nama unik untuk menunjuk referensi ke suatu resource atau berkas lain. Nilai ini dapat dipanggil kembali untuk menggunakan resource tersebut.
- x:Class : menunjukkan nama kelas.
- x:Name : menunjukkan nama sebuah obyek dan untuk membedakan antar obyek yang satu dengan obyek yang lain.
- x:Uid : mengidentifikasi elemen objek dalam XAML. Objek elemen merupakan objek yang dapat melakukan kontrol terhadap kelas atau elemen lain yang ditampilkan di desain antarmuka.

2.1.3 Kontrol terhadap Ponsel

Maksud dari kontrol terhadap ponsel adalah pengaturan tata letak terhadap antarmuka di Windows Phone. Windows Phone 8 menyediakan banyak set kontrol yaitu tata letak, tombol, kontrol masukan untuk mendapatkan informasi sampai ke menu.

2.1.3.1 Navigasi

Aplikasi yang dibuat di Windows Phone didasarkan pada model halaman. Maksud dari model halaman adalah pengguna berpindah dari satu halaman ke halaman lain dengan konten yang berbeda-beda dengan frame sebagai pengontrolnya. Setiap antarmuka aplikasi dibungkus dengan frame. Frame inilah yang melakukan kontrol terhadap aplikasi dan memungkinkan berpindah dari satu halaman ke halaman lain. Sedangkan halaman merupakan pembungkus dari elemen di dalamnya saja. Untuk lebih jelas mengenai frame, halaman dan area konten dapat dilihat pada gambar 2.1.



Gambar 2.1: Hirarki navigasi

2.1.3.2 Kontrol Tata Letak

Kontrol Tata Letak merupakan penampung pada antarmuka Windows Phone untuk objek di antarmuka dan kontrol yang lain (tombol radio, textbox, dan lai-lain). Kontrol tata letak digunakan untuk meletakan objek-objek di layar. Ketika pertama membuat aplikasi Windows Phone maka tata letak dasar sebagai penampung akan langsung dibuat berikut panel judul dan panel konten. Selanjutnya untuk penambahan kontrol tata letak yang lain dapat ditambahkan di panel konten.

Ada 3 macam panel yang dipakai untuk menangani tata Letak yaitu Grid, StackPane, dan Canvas. Perlu diperhatikan bahwa setiap halaman hanya memiliki satu macam panel. Berikut 3 macam panel di Windows Phone:

- StackPanel merupakan panel yang memposisikan element menjadi 1 baris dan beberapa elemen di setiap halaman diposisikan horizontal atau vertical saja.
- Grid merupakan panel yang mendukung tata letak yang rumit. Panel ini memposisikan elemen di baris dan kolom mana saja di setiap halaman.
- Canvas memposiskan elemen sebagai absolut kordinat. Jadi setiap elemen di dalam Canvas dapat diposisikan spesifik sesuai kordinat x dan y.

Kode untuk mengatur jenis panel pada Windows Phone dapat dilihat pada *listing* 2.1 .

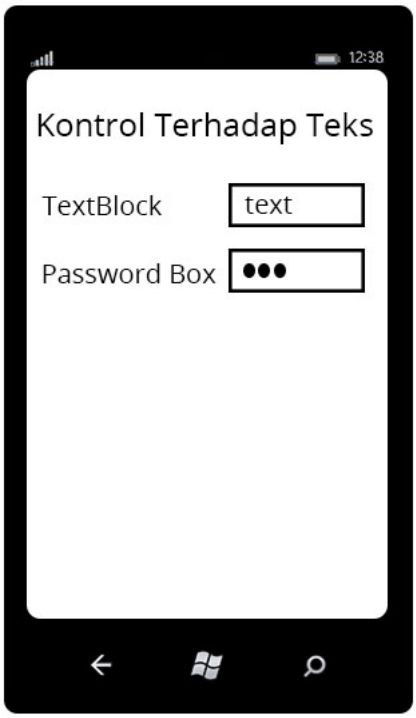
Listing 2.1: Kode tata letak grid

```
1 <Grid x:Name="ContentPanel">
2 </Grid>
```

2.1.3.3 Kontrol Terhadap Teks

Kontrol terhadap teks akan menampilkan menampilkan konten yang memiliki tipe String. Ada berbagai macam kontrol terhadap teks di Windows Phone yaitu TextBlock, TextBox dan PasswordBox. Ketiga macam kontrol tersebut dibedakan menurut tujuannya. Berikut keterangan, gambar 2.2, dan kode pada *listing* 2.2 kontrol teks.

- TextBlock merupakan tempat menaruh potongan teks yang hanya bisa dilihat.
- TextBox biasanya digunakan untuk teks masukan yang pendek. Tapi bisa juga dipakai untuk masukan yang banyak dan beberapa baris.
- PasswordBox biasanya digunakan untuk masukan yang bersifat rahasia. Karakter yang dimasukan langsung disamarkan menjadi bentuk titik.



Gambar 2.2: TextBlock, TextBox dan PasswordBox

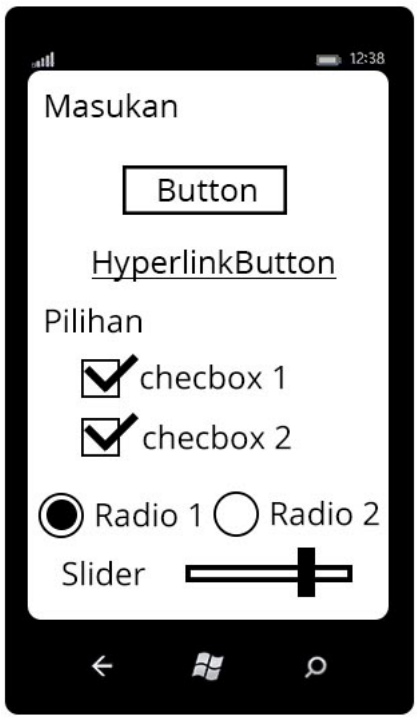
Listing 2.2: Kode untuk menampilkan TextBlock dan TextBox

```
1 <TextBlock x:Name="TextBlock1" Text="TextBlock"/>
2 <TextBox x:Name="TextBox1"Text="TextBox"/>
```

2.1.3.4 Tombol dan Kontrol Pilihan

Tombol memungkinkan pengguna untuk bernavigasi. Sedangkan kontrol pilihan memudahkan dalam memilih. Berikut keterangan, gambar 2.3, dan kode pada *listing* 2.3 tombol dan kontrol pilihan.

- Button merupakan kontrol yang dipakai pengguna untuk mengaktifkan *event* klik.
- HyperlinkButton merupakan kontrol yang menampilkan hyperlink. Jika hyperlink di tekan maka akan berpindah ke halaman yang dituju.
- CheckBox merupakan kontrol yang memungkinkan pengguna memilih beberapa item.
- RadioButton merupakan kontrol yang memungkinkan pengguna memilih satu pilihan dari beberapa pilihan.
- Slider merupakan kontrol yang memungkinkan user memilih nilai kisaran dari jalur yang sudah disediakan.



Gambar 2.3: Gambar kontrol pada Windows Phone

Listing 2.3: Kode untuk menampilkan TextBlock, TextBox, dan PasswordBox

```
1    <Button x:Name="find" Content="Button"/>
```

2.1.3.5 Kontrol Daftar

Kontrol yang dipakai untuk menampilkan daftar dari beberapa item. Berikut keterangan, gambar 2.4, dan kode pada *listing* 2.4 kontrol daftar.

- ListBox akan menampilkan daftar item. Daftar ini dapat dipilih dengan cara di klik.
- LongListSelector dipakai untuk mengelompokan, menampilkan, dan melakukan penggulungan terhadap daftar yang panjang.

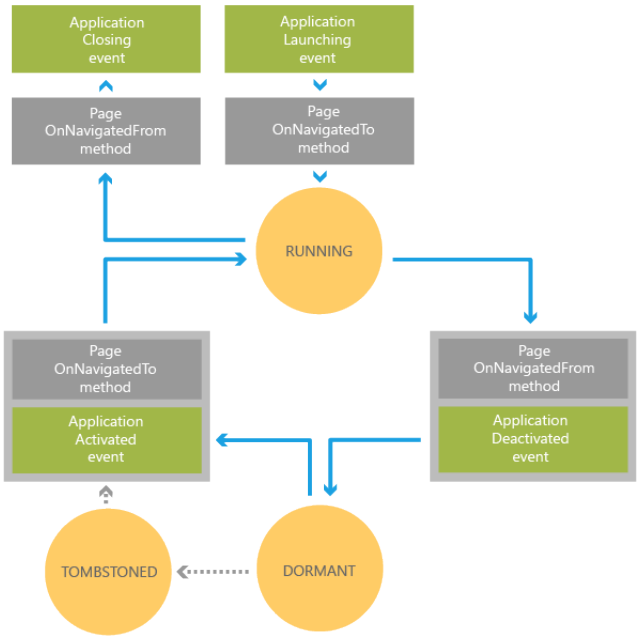
Gambar 2.4: Antarmuka ListBox

Listing 2.4: Kode untuk menampilkan listBox

```
1    <ListBox>
2        <ListBoxItem Content="Item 1" />
3        <ListBoxItem Content="Item 2" />
4        <ListBoxItem Content="Item 3" />
5        <ListBoxItem Content="Item 4" />
6        <ListBoxItem Content="Item 5" />
7    </ListBox>
```

2.1.4 Siklus Hidup Aplikasi

Siklus hidup aplikasi merupakan waktu mulai dari aplikasi dijalankan sampai aplikasi dibuang dari memori. Siklus hidup aplikasi penting diketahui agar pengguna tidak kecewa menggunakan aplikasi serta memastikan sumber daya tersedia (dalam aplikasi ini yaitu sumber daya GPS). Seringkali pengguna tidak berhati-hati dalam menggunakan aplikasi, maka dari itu penulis harus paham kapan aplikasi harus diaktifkan, ditangguhkan atau bahkan di hapus karena sudah tidak digunakan. Gambar 2.5 akan mengilustrasikan siklus hidup pada Windows Phone.



Gambar 2.5: Gambar siklus hidup aplikasi[1]

Sesuai gambar 2.5 lingkaran melambangkan keadaan aplikasi, persegi panjang menunjukan peristiwa aplikasi atau tingkat peristiwa di halaman. Berikut keterangan untuk siklus hidup Windows Phone pada gambar 2.5.

- The Launching Event
  Merupakan tampilan awal saat aplikasi dipilih yang memberitahukan pengguna bahwa aplikasi

sedang dijalankan. *Event* ini akan dijalankan ketika pertama kali melakukan permintaan terhadap aplikasi. *Event* ini akan berjalan di belakang ketika aplikasi ditutup sementara atau sedang berada pada keadaan Dormant atau Tombstoned menjadi Running.

- Running
  Setelah diluncurkan aplikasi akan masuk keadaan Running. Hal ini akan terus berlangsung sampai pengguna benavigasi ke depan, atau mundur melewati halaman utama aplikasi. Aplikasi keluar dari keadaan Running jika telefon di kunci. Keadaan Running masih dapat terjadi saat telefon di kunci dengan menonaktifkan *idle detection* pada aplikasi.

- Metode OnNavigatedFrom
  Merupakan metode yang dipanggil ketika bernavigasi ke halaman lain aplikasi. Ketika metode ini dipanggil maka aplikasi akan menyimpan keadaan dari halaman sebelum ditinggalkan. Hal tersebut dibutuhkan sehingga halaman tersebut bisa dikembalikan ke keadaan sebelum ditinggalkan saat pengguna ingin kembali ke halaman tersebut. Pemanggilan dilakukan ketika bernavigasi antara halaman di aplikasi atau ketika berpindah aplikasi.

- The Deactivated Event
  Event ini akan terjadi ketika pengguna berpindah aplikasi dan menekan tombol "start" atau menjalankan aplikasi lain. Untuk penanganan *deactivated event*, aplikasi harus menyimpan data sebelumnya sehingga data sebelum bisa dikembalikan suatu saat. Windows Phone 8 juga memdukung sistem pengembalian data dengan State Object. State Object akan untuk menyimpan keadaan aplikasi sebelum aplikasi dinonaktifkan.

- Dormant
  Keadaan setelah *deactivated event* terjadi. Pada keadaan ini, semua thread aplikasi akan dihentikan dan tidak ada proses yang terjadi, tetapi aplikasi tetap utuh di memori. Tetapi jika operating sistem membutuhkan memori yang lebih besar maka aplikasi yang dalam keadaan Dormant akan menjadi Tombstone untuk membebaskan memori.

- Tombstoned
  Aplikasi yang masuk ke keadaan Tombstoned akan dihentikan, namun sistem operasi akan menyimpan informasi aplikasi pada saat aplikasi berada di keadaan Deactivated.

- The Activated Event
  Event ini dipanggil ketika aplikasi meninggalkan keadaan Dormant atau Tombstoned. Operasi ini dilakukan pada latar belakang.

- The OnNavigatedTo Method
  Metode ini dipanggil ketika pengguna bernavigasi ke halaman yang sebelumya ditinggalkan. Metode ini akan memeriksa keadaan aplikasi dan memulihkannya jika keadaan sebelumnya pernah disimpan.

- The Closing Event
  Event ini akan tercapai ketika pengguna bernavigasi mundur keluar dari halaman utama. Pada kasus ini, aplikasi akan dihentikan dan tidak ada keadaan yang disimpan.

2.1.5 Peta di Windows Phone

Peta yang dipakai di Windows Phone adalah Windows Phone Maps. Windows Phone menawarkan beberapa pilihan dalam tampilan peta mulai dari kartografi, pencahayaan dan pandangan. Selain tampilan pada Sub Bab ini akan dibahas mengenai mendapatkan lokasi, petunjuk arah, MapPolyline dan Pushpins.

2.1.5.1 Penambahan Peta Ke Aplikasi

Untuk penambahan Peta pada Windows Phone menggunakan kontrol peta. Kontrol peta merupakan bagian dari perpustakaan Windows Phone. Dengan begitu untuk dapat menggunakannya perlu direferensikan. Untuk dapat menggunakannya juga harus ditambah *capability* ID_CAP_MAP. Selanjutnya barulah peta dapat ditampilkan. Berikut gambar 3.5, kode XAML pada *listing* 2.5, dan kode program pada *listing* 2.6 peta.

Gambar 2.6: Tampilan peta pada Windows Phone

Listing 2.5: Menampilkan peta dengan nama MyMap dari XAML

```
1     <Controls:Map x:Name="MyMap"/>
```

Listing 2.6: Menampilkan peta dengan nama MyMap dari kode program

```
1     public mapFrom()
2   {
3           InitializeComponent();
4           Map MyMap = new Map();
5           ContentPanel.Children.Add(MyMap);
6   }
```

2.1.5.2 Tampilan Peta di Windows Phone

Dalam tampilannya ada beberapa hal yang perlu diperhatikan agar pengguna merasa nyaman saat melihat peta di Windows Phone. Beberapa tampilan yang bisa ditampilkan dibuat untuk hal yang berbeda-beda. Berikut akan dibahas menentukan pusat dan tingkat zoom, kartografi, warna dan tampilan peta.

- Menentukan pusat peta berarti menentukan titik tengah sebagai pandangan awal di peta. Untuk penentuan titik tengah dibutuhkan 2 nilai yaitu latitude dan longitude. Sedangkan

tingkat zoom merupakan properti untuk mengatur seberapa dekat atau jauh pandangan yang akan ditampilkan di peta. Tingkat zoom memiliki nilai yang bisa diatur dari satu hingga 20. Kode untuk mengatur titik tengah peta dan tingkat *zoom* dapat dilihat pada *listing* 2.7 dan *listing* 2.8.

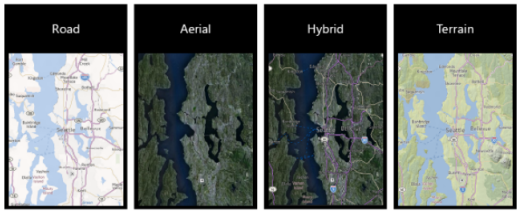
Listing 2.7: Mengatur tingkat zoom dari XAML

```
<Controls:Map x:Name="MyMap" ZoomLevel="10" Margin="-25,0,-16,0"/>
```

Listing 2.8: Mengatur tingkat zoom dari dari kode program

```
public mapFrom()
{
    InitializeComponent();
    Map MyMap = new Map();

    //Mengatur titik tengah peta
    MyMap.Center = new GeoCoordinate(47.6097, -122.3331);

    //mengatur tingkat zoom
    MyMap.ZoomLevel = 10;
    ContentPanel.Children.Add(MyMap);
}
```

- Kartografi peta di Windows Phone merupakan cara pandang dalam melihat dan menerjemahkan peta. Ada beberapa 4 jenis kartografi, yaitu:
  - Road: Tampilan normal 2 dimensi.
  - Aerial: Tampilan peta yang diambil dari foto di udara.
  - Hybrid: Tampilan Aerial yang digabung dengan jalan dan label.
  - Terrain: Menampilkan gambar fisik bumi termasuk ketinggian dan air.



Gambar 2.7: Kartografi

- Mode warna yang disediakan Windows Phone ada 2 yaitu terang dan gelap. Secara bawaan mode pada peta di Windows Phone adalah terang.
- Tampilan pada Peta di Windows Phone dapat berubah karena hasil diputar, dimiringkan, ditarik, dan diturunkan. Berikut beberapa hal yang dapat diatur sebagai tampilan di peta.
  - *Heading* merupakan representasi dari derajat secara geometri. Derajat ini didefinisikan dalam 0 sampai 360 yang dipakai untuk memutar peta. Contoh, 0 atau 360 ke arah utara, 90 ke arah barat, 180 ke arah selatan, dan 270 derajat ke arah timur.
  - *Pitch* merupakan derajat kemiringan dari peta dari sudut pengguna. Contoh, $Pitch = 0$ berarti melihat dari atas ke bawah sedangkan $Pitch = 45$ berarti melihat dari samping ke bawah dengan sudut 45 derajat.

2.1.5.3 Pushpins ke Peta

Pushpin merupakan elemen yang dapat ditempatkan pada peta secara spesifik dan bisa dipakai untuk interaksi pada peta. Peta tidak mendukung langsung penggunaan pushpin karena pushpin merupakan elemen MapOverlay (bagian/lapisan terpisah dari peta). Untungnya di Windows Phone memiliki Windows Phone 8 Toolkit yang memiliki set objek agar dapat menggunakan pushpin pada peta di Windows Phone. Contoh keluaran pushpin dapat dilihat pada Gambar 2.8 dan kode untuk menampilkannya dapat dilihat pada *listing* 2.9.

Gambar 2.8: Keluaran Toolkit Pushpin pada peta [3]

Listing 2.9: Kode untuk menampilkan pushpin

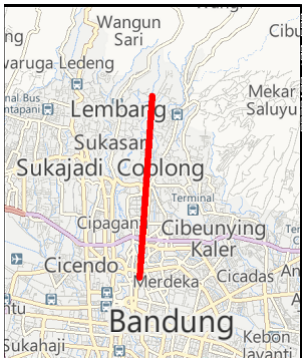
```
MapOverlay overlay = new MapOverlay
{
    GeoCoordinate = map.Center,
    Content = new Border
    {
    BorderBrush = new SolidColorBrush(Color.FromArgb(120, 255, 0, 0)),
    Child = new TextBlock(){Text="Pushpin"},
    BorderThickness = new Thickness(1),
    Background = new SolidColorBrush(Color.FromArgb(120,255,0,0)),
    Width = 80,
    Height = 60
    }
};
MapLayer layer = new MapLayer();
layer.Add(overlay);

map.Layers.Add(layer);
```

2.1.5.4 Polyline pada Peta

Dalam menentukan arah dibutuhkan 2 titik yaitu titk awal dan titik tujuan. Tentu saja arah tersebut butuh ditandai dengan garis. Polyline merupakan tentetan garis lurus yang saling terhubung satu sama lain. Dengan polyline arah pada peta dapat ditandai dengan warna maupin tebal atau tipisnya garis. Contoh keluaran polyline dapat dilihat pada Gambar 2.9 dan kode untuk menampilkannya dapat dilihat pada *listing* 2.10.

Listing 2.10: Kode untuk menampilkan polyline

```
MapPolyline line = new MapPolyline();
line.StrokeColor = Colors.Red;
line.StrokeThickness = 10;
line.Path.Add(new GeoCoordinate(-6.8619546, 107.614441));
line.Path.Add(new GeoCoordinate(-6.908693, 107.611185));
```



Gambar 2.9: Tampilan polyline pada Peta

2.1.5.5 Namespace Kontrol Map

Namespace merupakan nama yang dipakai untuk mengatur kelas-kelas. Windows Phone 8 sudah menyediakan namespace bawaan untuk mengatur peta. Namespace yang disediakan adalah Maps.Controls. Namespace ini yang berisi kelas-kelas yang paling sering digunakan untuk mengatur peta pada Windwows Phone. Agar dapat menggunakan kelas pada namespace tersebut perlu ditambahkan namespace dan capabilities. Namespace yang harus ditambahkan pada baris awal XAML adalah Microsoft.Phone.Maps.Controls. Selanjutnya ada penambahan capabilities ID_CAP_MAP. Penambahan capabilities ditambahkan pada WMAppManifest.xml.

2.1.5.6 Kelas Map

Merupakan kelas yang mewakili kontrol map.
Berikut properti yang dapat digunakan pada kelas ini.

Nama	Deskripsi
CartographicMode	Mengatur dan mendapatkan tipe dari peta.
Center	Mengatur dan mendapatkan lokasi tengah pada peta.
ColorMode	Mengatur dan mendapatkan mode warna peta
Heading	Mengatur dan mendapatkan arah pandang peta.
Height	Mengatur dan mendapatkan tinggi.
LandmarksEnabled	Mengindikasikan apakah bangunan 3D ditampilkan.
Name	Mengatur dan mendapatkan nama identifikasi objek.
PedestrianFeaturesEnabled	Mengindikasikan fitur pejalan kaki ditampilkan.
Pitch	Mengatur dan mendapatkan derajat kemiringan peta.
Tag	Mengatur dan mendapatkan nilai objek.
TileSources	Mendapatkan koleksi lapisan lantai.
Width	Mengatur dan mendapatkan lebar.
ZoomLevel	Mengatur dan mendapatkan tingkat zoom pada peta.

Tabel 2.1: Properti kelas Map

Berikut metode yang dapat digunakan pada kelas ini.

- SetView(LocationRectangle)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis. Metode ini tidak mengambalikan nilai.

- SetView(GeoCoordinate, Double)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah dan tingkat zoom. Metode ini tidak mengambalikan nilai.

- SetView(LocationRectangle, MapAnimationKind)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis dan animasi. Metode ini tidak mengambalikan nilai.

- SetView(LocationRectangle, Thickness)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis dengan batas tertentu. Metode ini tidak mengambalikan nilai.

- SetView(GeoCoordinate, Double, MapAnimationKind)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, dan animasi. Metode ini tidak mengambalikan nilai.

- SetView(GeoCoordinate, Double, Double)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, dan heading. Metode ini tidak mengambalikan nilai.

- SetView(LocationRectangle, Thickness, MapAnimationKind)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis dengan batas tertentu, dan animasi. Metode ini tidak mengambalikan nilai.

- SetView(GeoCoordinate, Double, Double, MapAnimationKind)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, heading, dan animasi. Metode ini tidak mengambalikan nilai.

- SetView(GeoCoordinate, Double, Double, Double)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, heading, pitch. Metode ini tidak mengambalikan nilai.

- SetView(GeoCoordinate, Double, Double, Double, MapAnimationKind)
  Metode untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, heading, pitch, dan animasi. Metode ini tidak mengambalikan nilai.

- UpdateLayout
  Metode yang memastikan semua posisi objek turunan mengikuti tata letak.

2.1.5.7 Kelas Polyline

Merupakan kelas yang dipakai untuk menggambarkan garis lurus yang saling terhubung. Kelas ini tergabung ke dalam namespace Microsoft.Phone.Maps.Controls.

Berikut properti yang dapat digunakan pada kelas ini.

Berikut metode yang dapat digunakan pada kelas ini.

- CheckAccess
  Metode yang menentukan bisa atau tidaknya pemanggilan thread untuk mengakses objek ini.

Nama	Deskripsi
Dispacher	Mendapatkan objek yang terkait.
Path	Mengatur dan mendapatkan kumpulan GeoCoordinates yang membuat polyline.
StrokeColor	Mengatur dan mendapatkan warna garis.
StrokeDashed	Mengatur dan mendapatkan nilai untuk mengetahui jika polyline dihancurkan.
StrokeThickness	Mengatur dan mendapatkan lebar garis untuk menggambar polyline.

Tabel 2.2: Properti Kelas Polyline

- ClearValue
  Metode yang akan membersihkan nilai lokal

- Finalize
  Metode yang dipakai untuk melakukan pembersihan pada sumber daya yang tidak terpakai sebelum objek dihancurkan.

2.1.5.8 Kelas Pushpin

Merupakan kelas yang dipakai untuk menggambarkan elemen terpisah diatas peta. Meskipun pushpin merupakan bawaan pada peta untuk menunjuk suatu lokasi tetapi pushpin dari peta tidak dapat diubah-ubah. Pushpin pada Windows Phone 8 dapat dibuat sesuai ketertarikan[1]. Namun ada cara lain dengan menambahkan Windows Phone Toolkit. Windows Phone Toolkit mempunyai komponan untuk menggambar pushpin diatas peta.

2.1.6 Lokasi

Aplikasi di Windows Phone 8 dapat memanfaatkan lokasi dimana perangkat berada. Aplikasi dapat melacak lokasi sesaat pengguna atau pelacakan selama periode tertentu. Data lokasi perangkat berasal dari berbagai sumber termasuk *Global Positioning System* atau GPS, Wi-Fi, dan seluler. Ada 2 set API berbeda yang dapat dimanfaatkan di Windows Phone yaitu *Runtime Location* API dan .NET *Location* API. Windows Phone *Runtime Location* memiliki keunggulan fitur yang banyak sedangkan .NET *Location* direkomendasikan jika aplikasi ditargetkan pada Windows Phone 7.1 dan Windows Phone 8.

Hal yang perlu diperhatikan dalam menentukan layanan lokasi adalah penangkap GPS, Wi-Fi, dan radio seluler. Perangkat tersebut berfungsi sebagai penyedia data lokasi dengan berbagai tingkat akurasi dan konsumsi daya. Perangkat diatas juga berkomunikasi langsung untuk memutuskan sumber mana yang digunakan untuk menentukan lokasi perangkat berdasarkan ketersediaan data lokasi dan prasyarat yang ditentukan aplikasi. Lapisan diatas penyedia data lokasi tersebut adalah pengelola antarmuka. Aplikasi akan mengunakan antarmuka tersebut untuk memulai dan menghentikan layanan lokasi, mengatur tingkat akurasi, dan menerima data lokasi.

Karena pengguna dapat berpindah tempat untuk menuju tempat yang lain, maka pelacakan lokasi harus dilakukan terus menerus. Pelacakan lokasi secara terus menerus ini dapat dilakukan di depan maupun di belakang aplikasi Windows Phone 8. Pelacakan aplikasi di depan akan memungkinkan aplikasi melacak lokasi pengguna sekaligus melakukan perbaruan antarmuka. Jika pelacakan

[1] http://developer.nokia.com/resources/library/Lumia/change-history/archived-content/maps-and-navigation/guide-to-the-wp8-maps-api.html

lokasi di belakang aplikasi maka tidak ada perubahan pada antarmuka namun pelacakan terus dilakukan. Pelacakan yang terus menerus di belakang aplikasi akan membuat keadaan aplikasi cepat dipulihkan dari keadaan Dormant.

2.1.6.1 Mendapatkan Posisi Pengguna

Di Windows Phone 8 telah ada GeoCoordinate yang dapat digunakan untuk mengetahui posisi pengguna. Geolocator dari Windows.Devices.Geolocation akan mengembalikan posisi saat ini. Untuk menggunakan Geolocator, perlu menghidupkan ID_CAP_LOCATION di \properties\WMAppManifest.xml. Dalam mendapatkan posisi juga perlu diperhatikan status dari GPS karena GPS membutuhkan waktu dari mengaktifkan GPS hingga mendapatkan lokasi pengguna secara akurat. Untuk lebih jelas mengenai status dalat dilihat pada nilai status dibawah ini.

Berikut nilai yang mungkin dari Status Posisi:

- *Ready* : Jika lokasi tersedia.
- *Initializing* : Status jika penangkap GPS belum memiliki cukup satelit untuk mendapatkan posisi yang akurat.
- *NoData* : Data lokasi belum tersedia. Status ini muncul jika aplikasi sedang mamanggil GetGeopositionAsync atau register.
- *Disable* : Status mengindikasikan tidak diperbolehkannya pengaksesan lokasi.
- *NotInitialized* : Data lokasi belum tersedia. Status ini muncul jika aplikasi belum mamanggil GetGeopositionAsync atau register.
- *NotAvailable* : Jika Windows sensor dan lokasi tidak tersedia.

2.1.6.2 Namespace Geolocator

Namespace merupakan nama yang dipakai untuk mengatur kelas-kelas. Windows Phone 8 sudah menyediakan namespace bawaan untuk mengakses lokasi. Namespace yang disediakan adalah namespace Geolocator. Namespace ini akan mengakses lokasi geografis dari perangkat dan mendukung pelacakan lokasi dari waktu ke waktu. Agar dapat menggunakan kelas pada namespace tersebut perlu ditambahkan namespace dan *capabilities*. Namespace yang harus ditambahkan pada baris awal XAML adalah **Windows.Device.Geolocator**. Selanjutnya ada penambahan *capabilities* ID_CAP_LOCATION. Penambahan *capabilities* ditambahkan pada WMAppManifest.xml. Kelas yang diatur oleh namespace geolocator dapat di lihat pada tabel 2.3.

Kelas	Deskripsi
Geocoordinate	Berisi informasi untuk mengidentifikasi lokasi geografis.
Geolocator	Mendukung dalam pengaksesan lokasi perangkat.
Geoposition	Memberikan data lokasi beserta latitude dan longitude atau data alamat.

Tabel 2.3: Kelas pada Namespace Geolocator

2.1.6.3 Kelas Geocoordinate

Kelas Geocoordinate adalah kelas yang menunjukan lokasi sebagai kordinat geografis. Kelas ini hanya menyediakan properti yang hanya bisa dibaca. Kelas ini menyediakan properti yang ditunjukan pada tabel 2.4.

Properti	Deskripsi
Altitude	Ketinggian lokasi dalam satuan meter.
Heading	Arah menghadap perangkat dalam satuan derajat yang relative terhadap mata angin utara.
Latitude	Garis lintang dalam satuan derajat.
Longitude	Garis bujur dalam satuan derajat.
Point	Lokasi dari Geocoordinate.
Speed	Kecepatan dalam satuan meter per detik.

Tabel 2.4: Properti pada Kelas Geocoordinate

2.1.6.4 Kelas Geolocator

Kelas Geolocator merupakan kelas yang mendukung pengaksesan terhadap lokasi.
Berikut metode yang disediakan kelas Geolocator:

- public IAsyncOperation<Geoposition> GetGeopositionAsync()
  Operator await diatas dimaksudkan untuk meminta posisi lokasi terus menerus sampai selesai dan menunda tugas yang lain.
  Metode GetGeopositionAsync yang merupakan bawaan Kelas Geolocator akan meminta data lokasi dan menanganinya sampai selesai. Kembalian dari metode GetGeopositionAsync adalah Geoposition.

Berikut Properti yang disediakan kelas Geolocator:

- public PositionStatus LocationStatus get;
  Merupakan properti dari kelas geolocator untuk mendapatkan status posisi dengan mengembalikan kelas PositionStatus. Status pada kelas PositionStatus adalah Ready, Initializing, NoData, Disable, NotInitialized, dan NotAvailable.

- public PositionAccuracy DesiredAccuracy get; set;
  Properti yang digunakan untuk mengatur dan mendapatkan tingkat akurasi. Untuk tingkat akurasi dapat dipilih tingkat *High* untuk tingkat akurasi tinggi dan dipilih tingkatDefault untuk menghemat daya. Keluaran dari properti ini adalah tipe data PositionAccuracy.

- public Nullable<uint> DesiredAccuracyInMeters get; set;
  Sama seperti properti DesiredAccuracy diatas tetapi dalam satuan meter. Keluaran dari properti ini adalah tipe data uint.

- public uint ReportInterval get; set;
  Merupakan properti untuk mendapatkan selang waktu pembaruan lokasi. Properti ini mengeluarkan tipe data unit.

2.1.6.5 Kelas Geoposition

Kelas Geoposition merupakan kelas yang memuat lokasi (latitude dan longitude). Berikut Properti yang disediakan kelas Geoposition:

- public CivicAddress CivicAddress get;
  Data alamat sipil yang terkait dengan lokasi geografis.

- public Geocoordinate Coordinate get;
  Data latitude dan longitude yang terkait lokasi geografis.

2.1.7 Memanfaatkan Sumber Data

Hal yang penting dari sebuah aplikasi adalah informasi. Windows Phone 8 memiliki kemampuan dalam menghubungkan aplikasi dengan sumber data lainnya. Memanfaatkan sumber data ada 2 cara yaitu yang lokal atau berada di perangkat dan web service. Web Service merupakan metode komunikasi antara 2 perangkat melalui jaringan.

Sebelum data dapat dikirim antar perangkat perlu dilakukan *Serialization*. *Serialization* disini merupakan proses mentransformasikan objek ke format yang bisa dengan mudah dikirim melewati jaringan atau disimpan di database. Formatnya disini berupa string yang direpresentasikan sebagai objek di XML atau JSON(Javascript Object Notation). Ada beberapa objek yang dapat melakukan serialisasi, tetapi yang akan dibahas penulis disini hanya serialisasi JSON.

Banyak *web service* yang mengembalikan data dalam format JSON. JSON memiliki struktur yang mudah dipahami dimana kurung kurawal mengindikasikan objek, kurung siku berarti array, dan properti berupa nama dan nilai pasangan yang dipisahkan oleh titik dua. JSON format memiliki ukuran data yang kecil dan baik untuk penggunaan perangkat bergerak. Untuk contoh format JSON dapat dilihat di bagian Kiri API pada Bab 2 ini karena Kiri API menggunakan format JSON. Serialisasi menggunakan DataContractJsonSerializer membuat serialisasi mudah untuk menerjemahkan form String JSON ke objek yang dapat langsung digunakan. DataContractJsonSerializer memakai WriteObject() untuk serialisasi and ReadObject() untuk de-serialisasi.

2.1.7.1 Kelas HttpClient

Merupakan Kelas yang dipakai untuk mengirim permintaan HTTP dan menerima kembalian HTTP dari *Uniform Resource Identifier*(URI) yang dapat diidentifikasi. Berikut metode yang disediakan kelas HttpClient.

- DeleteAsync
  Metode yang dipakai untuk mengirimkan permintaan DELETE ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

- GetAsync(Uri)
  Metode yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

- GetAsync(Uri,HttpCompletionOption)
  Metode yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan dan nilai tambahan yang dimaksukan sebagai indikasi operasi dianggap selesai. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

- GetBufferAsync
  Metode yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian yang dikirimkan secara buffer(disimpan dalam memori) dan kemajuan dari data yang dikirim.

- GetInputStreamAsync
  Metode yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian yang dikirimkan secara stream(langsung sesuai waktu) dan kemajuan dari data yang dikirim.

- GetStringAsync
  Metode yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dalam bentuk string dan kemajuan dari data yang dikirim.

- PostAsync
  Metode yang dipakai untuk mengirimkan permintaan POST ke URI yang spesifik sebagai

operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

- SendRequestAsync(HttpRequestMessage)
  Metode yang dipakai untuk mengirimkan permintaan HTTP sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter pesan dari permintaan. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

- SendRequestAsync(HttpRequestMessage, HttpCompletionOption)
  Metode yang dipakai untuk mengirimkan permintaan HTTP sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi metode ini dipanggil[2]. Metode ini membutuhkan parameter pesan dari permintaan dan nilai tambahan yang dimaksukan sebagai indikasi operasi dianggap selesai. Sedangkan kembaliannya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

2.2 Kiri API

API atau *Application Programming Interface* merupakan aturan yang dikodekan secara spesifik yang dapat digunakan untuk komunikasi antar aplikasi. Jadi API disini memfasilitasi untuk pemanggilan fungsi-fungsi tertentu diluar aplikasi itu sendiri. Pemanfaatan Kiri API adalah JSON atau *JavaScript Object Notation* format.

Pemanfaatan Kiri API dengan melakuan permintaan dengan parameter POST atau GET dan Kiri akan mengembalikan hasil dalam format JSON. Permintaan tersebut dikirimkan ke URL. Ada URL yang disediakan Kiri Api, yaitu:

- `http://preview.kiri.travel/handle.php`
  Merupakan URL untuk uji coba. Untuk kemampuannya juga menurut dokumentasi Kiri API masih tidak stabil.

- `http://kiri.travel/handle.php`
  Merupakan URL produksi. Ini merupakan URL rekomendasi terhadap permintaan.

Untuk setiap permintaan membutuhkan *API key* yang didapat dengan mendaftar[2]. Penggunaan API memungkinkan pengaksesan dimana saja dengan menggunakan koneksi internet. Pada Sub Bab dibawah penulis akan membahas beberapa pemanggilan pesan yang terdapat pada Kiri API.

Berikut langkah-langkah dari membuka `dev.kiri.travel` sampai mendapatkan *API key*.

[2] `http://msdn.microsoft.com/en-us/library/ms734701%28v=vs.110%29.aspx`

- Masuk ke situs dev.kiri.travel.
- Register dengan memasukan alamat email, nama, dan nama perusahaan.
- Password akan dikirimkan ke alamat email. Tentunya password akan dibuat otomatis oleh pihak Kiri.
- Login setelah mengetahui password yang diterima dari alamat email.
- Setelah berhasil login, di menu utama terdapat API Keys Managements yang dapat dipilih.
- Pilih tombol Add lalu masukan deskripsi penggunaan *API key* yaitu untuk Skripsi.
- *API key* didapat dan dapat digunakan.

2.2.1 *Routing Web Service*

Routing Web Service merupakan Kiri API yang digunakan untuk mendapatkan langkah perjalanan dari lokasi asal ke lokasi tujuan. Parameter dan keterangan untuk layanan ini dapat dilihat pada tabel 2.5.

version	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
mode	"findroute"	mengintruksikan layanan untuk mencari rute
locale	"en" or "id"	bahasa yang digunakan untuk balasan
start	lat,lng	titik awal *latitude* dan *longitude*
finish	lat,lng	titik akhir *latitude* dan *longitude*
presentation	"mobile" or "desktop"	Menentukan tipe prensentasi untuk keluaran. Contoh, jika tipe presentasi "mobile", maka link "tel:" akan ditambahkan di hasil.
apikey	16-digit hexadecimals	API key yang digunakan

Tabel 2.5: Tabel parameter layanan *routing web service*

Format kembalian layanan *routing web service* dapat dilihat pada *listing* 2.11:

Listing 2.11: Kode kembalian pencarian rute

```
{
    "status": "ok" or "error"
    "routingresults": [
        {
            "steps": [
                [
                    "walk" or "none" or others,
                    "walk" or vehicle_id or "none",
                    ["lat_1,lon_1", "lan_2,lon_2", ... "lat_n,lon_n"],
                    "human readable description, dependant on locale",
                    URL for ticket booking or null (future)
                ],
                [
                    "walk" or "none" or others,
                    "walk" or vehicle_id or "none",
                    ["lat_1,lon_1", "lan_2,lon_2", ... "lat_n,lon_n"],
                    "human readable description, dependant on locale",
                    URL for ticket booking or null (future)
                ]
            ],
            "traveltime": any text string, null if and only if route is not found.
        },
        {
```

```
24              "steps": [ ... ],
25              "traveltime": "..."
26          } ,
27          {
28              "steps": [ ... ],
29              "traveltime": "..."
30          } ,
31          ...
32      ]
33  }
```

Berikut maksud dari *listing* 2.11:
Ketika pencarian rute sukses dilakukan maka status akan memberitahukan "ok" seperti di baris 2. Selanjutnya setiap langkah dari posisi awal ke posisi tujuan akan ditampung di array dari langkah. Berikut keterangan dari setiap array tersebut:

- Index ke 0 atau baris 7 pada *listing* 2.11 dapat berisi "walk" atau "none" atau "others". Baris tersebut berarti jika "walk" untuk berjalan kaki, "none" jika rute tidak ditemukan dan "others" untuk menggunakan kendaran.
- Index ke 1 atau baris 8 pada *listing* 2.11 merupakan detail dari index 0. Artinya jika index 0 menyatakan "walk" berarti index 1 harus "walk", "none" berarti index 1 harus "none", dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk ditampilkan gambarnya.
- Index ke 2 atau baris 9 pada *listing* 2.11 adalah array string yang berisi jalur dalam format "lat,lng". Maksud dari "lat,lng" disini adalah titik awal dan titik akhir dari setiap jalur yang dilewati.
- Index ke 3 atau baris 10 pada *listing* 2.11 berisi bentuk yang akan ditampilkan kepada pengguna. Informasi yang disampaikan dapat berupa:
  - %fromicon = untuk menunjukan ikon "from". Biasanya untuk mode presentasi di perangkat bergerak.
  - %toicon = untuk menunjukan ikon "to". Biasanya untuk mode presentasi di perangkat bergerak.
- Index ke 4 atau bari 11 pada *listing* 2.11 berisi URL untuk pemesanan tiket jika tersedia. Jika tidak tersedia akan bernilai null.

Kiri telah menyediakan gambar untuk setiap angkutan umum. Gambar tersebut dapat di akses di URL:

- http://kiri.travel/images/means/[means]/[means_details].png
- http://kiri.travel/images/means/[means]/baloon/[means_details].png

Dimana

$$means$$

dapat diambil dari index 0 nilai kembalian dan

$$means_details$$

dapat diambil dari index 1 nilai kembalian.

2.2.2 Web Service Pencarian Lokasi

Merupakan Kiri API yang digunakan untuk mencari lokasi beserta kordinat *latitude* dan *longitude*. Parameter dan keterangan untuk layanan ini dapat dilihat pada tabel 2.6.

version	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
mode	"searchplace"	mengintruksikan layanan untuk mencari tempat
region	"cgk" or "bdo" or "sub"	kota yang akan dicari tempatnya
querystring	text apa saja dengan minimum text satu karakter	query string yang akan dicari menggunakan layanan ini
apikey	16-digit hexadecimals	API key yang digunakan

Tabel 2.6: Tabel parameter layanan pencarian lokasi

Format kembalian dari layanan pencarian lokasi dapat dilihat pada *listing* 2.12.

Listing 2.12: Kode kembalian pencarian lokasi

```
1  {
2      "status": "ok" or "error"
3      "searchresult": [
4          {
5              "placename": "place name"
6              "location": "lat,lon"
7          },
8          {
9              "placename": "place name"
10             "location": "lat,lon"
11         },
12         ...
13     ]
14     "attributions": [
15         "attribution_1", "attribution_2", ...
16     ]
17 }
```

Berikut maksud dari *listing* 2.12:
Ketika pencarian lokasi sukses dilakukan maka status akan memberitahukan "ok" seperti di baris 2. Selanjutnya akan ditampilkan hasil dari lokasi yang ada beserta atributnya. Berikut keterangan dari format dari pencarian lokasi:

- Searchresult (pada bari 4 sampai 7, 8 sampai 11, dan seterusnya) berisi array dari tempat:
  - placename: nama tempat.
  - location: latitude dan longitude dari tempat.
- Attributions berisi array string yang berisikan atribut tambahan untuk dimunculkan.

2.2.3 Web Service Menemukan Transportasi Terdekat

Merupakan Kiri API yang digunakan untuk menemukan rute transportasi terdekat sesuai titik yang diinginkan pengguna. Parameter dan keterangan untuk layanan ini dapat dilihat pada tabel 2.7.

version	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
mode	"nearbytransports"	mengintruksikan layanan untuk mencari rute transportasi terdekat
start	latitude dan longitude	kota yang akan dicari tempatnya
apikey	16-digit hexadecimals	API key yang digunakan

Tabel 2.7: Tabel parameter layanan menemukan transportasi terdekat

Format kembalian layanan menemukan transportasi terdekati dapat dilihat pada *listing* 2.13.

Listing 2.13: Kode kembalian menemukan lokasi terdekat

```
1  {
2      "status": "ok" or "error"
3      "nearbytransports": [
4          [
5              "walk" or "none" or others,
6              "walk" or vehicle_id or "none",
7              text string,
8              decimal value
9          ],
10         [
11             "walk" or "none" or others,
12             "walk" or vehicle_id or "none",
13             text string,
14             decimal value
15         ],
16         ...
17     ]
18 }
```

Berikut maksud dari *listing* 2.13:
Ketika pencarian rute sukses dilakukan maka status akan memberitahukan "ok" seperti di baris 2. Selanjutnya akan diberikan array yang berisi transportasi terdekat yang diurutkan dari yang terdekat ke yang terjauh. Berikut keterangan dari setiap array tersebut:

- Index ke 0 atau baris 5 pada *listing* 2.13 dapat berisi "walk" atau "none" atau "others". Artinya jika "walk" berarti berjalan kaki, "none" jika rute tidak ditemukan dan "others" berarti menggunakan kendaran.
- Index ke 1 atau baris 6 pada *listing* 2.13 merupakan detail dari index 0. Artinya jika index 0 "walk" berarti index 1 harus "walk", "none" berarti index 1 harus "none" dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk ditampilkan gambarnya.
- Index ke 2 atau baris 7 pada *listing* 2.13 berisi nama kendaraan.
- Index ke 3 atau bari 8 pada *listing* 2.13 berisi jarak dengan satuan kilometer.

BAB 3

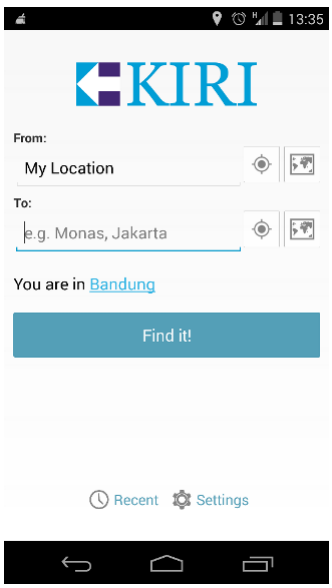
ANALISIS

Pada bab ini akan dibahas mengenai analisis aplikasi sejenis, analisis kebutuhan aplikasi, analisi kontrol yang dipakai, analisis terhadap siklus hidup aplikasi, analisis peta, analisis pemanfaatan sumber data, analisis Kiri API, diagram *Use Case*, dan diagram kelas.

3.1 Analisis Aplikasi Sejenis

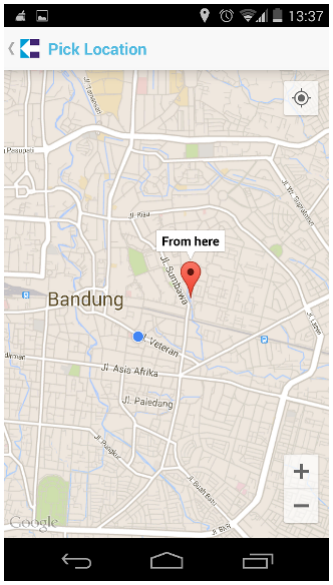
Aplikasi sejenis yang penulis temui bernama "Public Transport"[1]. Namun aplikasi "Public Transport" tersebut hanya dapat dijalankan di sistem aplikasi android. Aplikasi "Public Transport" ini memanfaatkan Kiri API. Aplikasi tersebut penggunaannya sederhana. Di halaman awal pengguna dapat mengetikan lokasi awal dan tujuan. Selain dengan mengetik pengguna juga dapat menunjuk lokasi pada peta. Setelah lokasi dipilih sistem akan memastikan dengan memberi daftar nama jalan dan tempat terkait. Jika sudah memilih maka sistem akan mengeluarkan hasil pencarian rute.

Berikut adalah tampilan dari aplikasi "Public Transport" (Gambar 3.1 sampai 3.5):

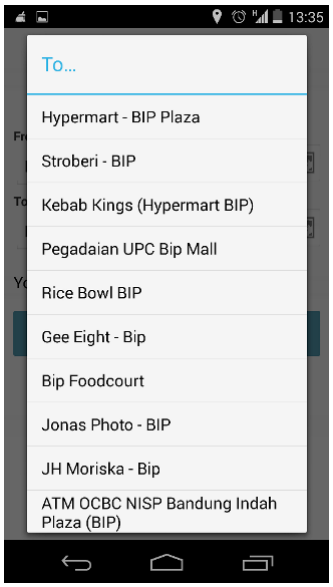


Gambar 3.1: Tampilan awal aplikasi Public Transport

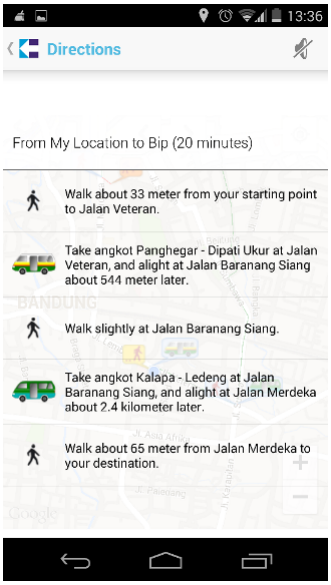
[1] https://play.google.com/store/apps/details?id=travel.kiri.smarttransportapp



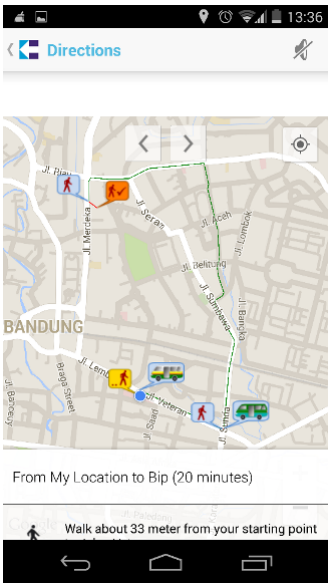
Gambar 3.2: Menunjuk lokasi pada peta



Gambar 3.3: Memberikan daftar nama tempat dan nama jalan terkait



Gambar 3.4: Tampilan rute kendaraan umum dalam bentuk daftar



Gambar 3.5: Tampilan rute kendaraan umum di peta

3.2 Analisis Aplikasi

Aplikasi akan dibuat mengguakan bahasa pemograman C#. Aplikasi yang digunakan untuk membangun Aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone adalah Visual Studio Express 2012. Pada sub bab ini akan dibahas kebutuhan aplikasi, anaslisis kontrol yang dipakai, analisis terhadap siklus hidup aplikasi, analisis peta, analisis pemanfaatan sumber data, analisa Kiri API, diagram use case, dan diagram kelas dari aplikasi yang akan dibangun.

3.2.1 Kebutuhan Aplikasi

Dari hasil observasi penulis dalam menentukan lokasi asal dan lokasi tujuan ada 2 cara. 2 cara tersebut yaitu dengan menulis alamat atau tempat dan dengan menunjuk pada peta. Cara menuliskan alamat atau tempat yaitu dengan menuliskan alamat atau tempat pada tempat yang disediakan untuk asal dan tujuan. Cara menunjuk pada peta yaitu dengan mengetuk layar di posisi yang diinginkan. Kedua hal tersebut pada dasarnya sama saja tetapi ada faktor kemudahan pengguna dalam pemakaiannya. Jadi penulis menyediakan 2 cara tersebut pada aplikasi yang penulis buat agar pengguna dapat memilih salah satunya.

Pada saat menuliskan lokasi atau tempat atau menunjuk langsung pada peta mungkin saja terjadi kesalahan. Kesalahan tersebut bisa saja salah pengetikan atau nama tempat yang tidak ada. Maka dari itu dibutuhkan pemeriksaan terhadap masukan pengguna. Pemeriksaan tersebut dilakukan setelah pengguna memulai pencarian dengan menekan tombol "FIND".

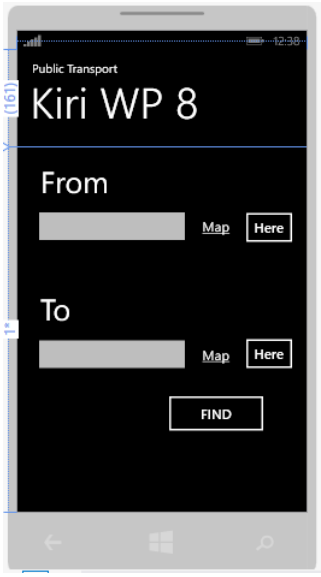
Untuk hasil keluaran ada 2 tipe seperti aplikasi peta lainnya. Kedua tipe tersebut adalah bentuk daftar dan bentuk peta. Bentuk daftar memudahkan dalam melihat tiap langkah rute. bentuk daftar memudahkan pengguna dalam melihat arah dan posisi lingkungan pada rute yang dipilih.

Aplikasi yang penulis bangun didasarkan pada kebutuhan adalah sebagai berikut:

1. Pengguna dapat memasukan lokasi asal dan lokasi tujuan pada bidang yang disediakan atau menunjuk langsung lokasi pada peta.
2. Mendapatkan lokasi terkait menurut lokasi yang dimasukan pengguna.
3. Menampilkan hasil rute angkutan umum dari lokasi asal ke lokasi tujuan.

3.2.2 Analisis Kontrol yang Dipakai

Dari kebutuhan yang telah disebutkan diatas penulis menyadari pentingnya kontrol yang harus dipakai. Kontrol yang dimaksud termasuk tata letak, teks, pilihan, dan daftar. Kebutuhan akan kontrol penting bukan hanya untuk kebutuhan tapi memudahkan pengguna. Antarmuka halaman utama yang penulis bangun dapat dilihat pada Gambar 3.6.



Gambar 3.6: Antarmuka pencari rute kendaraan umum di Windows Phone

Untuk kontrol tata letak penulis membayangkan pengaturan yang tertata rapih dan beberapa elemen dalam satu baris atau kolom. Tetapi juga penulis tidak mengharapkan penggunaan kontrol tata letak yang rumit. Dari hasil pengamatan penulis kontrol tata letak yang cocok adalah Grid. Kontrol tata letak ini penulis pilih karena mudah diposisiskan sesuai baris dan kolom. Selain itu tampilan Grid akan menyesuaikan jika layar diputar dari posisi pemandangan ke posisi potret dan sebaliknya. Berikut kode untuk membuat tata letak pada halaman Windows Phone menjadi tata letak Grid.

Kontrol terhadap teks juga diperlukan untuk aplikasi ini. Kebutuhan yang diperlukan adalah mengeluarkan potongan teks yang dapat dibaca dan tempat pengguna memasukan teks. Untuk mengeluarkan teks yang dapat dilihat penulis akan menggunakan TextBlock. TextBlock digunakan untuk menampilkan tulisan "from" dan "to" pada aplikasi. Untuk masukan pengguna terhadap aplikasi penulis akan menyediakan TextBox sebagai tempat teks. TextBox digunakan sebagai masukan untuk lokasi asal dan lokasi tujuan. Berikut kode dan hasil yang ditampilkan pada Windows Phone.

Suatu aplikasi tentunya tidak hanya mempunyai satu halaman. Sama hal dengan aplikasi yang penulis buat memiliki beberapa halaman yang mempunyai tugas berbeda. Karena hal tersebut dibutuhkan kontrol untuk berpindah dari satu halaman ke halaman lain. Kontrol yang dibutuhkan yaitu kontrol tombol. Kontrol tombol akan mengeksekusi *event click* yang memungkinkan pindah halaman dan melakukan perintah. Kontrol tombol akan penulis gunakan untuk berpindah ke halaman peta, menemukan lokasi pengguna, dan pencarian rute. Pada Gambar 3.6 terdapat 5 tombol yaitu tombol map pada bagian from, tombol here pada bagian from, tombol map pada bagian to, tombol here pada bagian to, dan tombol find. Berikut kegunaan dari tombol-tombol tersebut.

- Tombol map pada bagian from
  Tombol untuk berpindah dari halaman utama menuju halaman peta. Pada halaman peta pengguna dapat menunjuk lokasi asal dan kembali lagi ke halaman utama. Saat kembali ke halaman utama lokasi yang dipilih akan disimpan dan pada TextBox bagian from akan tertulis

"lokasi dari peta".

- Tombol map pada bagian from
  Tombol untuk mancari lokasi pengguna. Setelah tombol ini di tekan maka lokasi pengguna akan disimpan dan pada bagian TextBox bagian from akan tertulis "here".

- Tombol map pada bagian to
  Tombol untuk berpindah dari halaman utama menuju halaman peta. Pada halaman peta pengguna dapat menunjuk lokasi tujuan dan kembali lagi ke halaman utama. Saat kembali ke halaman utama lokasi yang dipilih akan disimpan dan pada TextBox bagian to akan tertulis "lokasi dari peta".

- Tombol map pada bagian to
  Tombol untuk mancari lokasi pengguna. Setelah tombol ini di tekan maka lokasi pengguna akan disimpan dan pada bagian TextBox bagian to akan tertulis "here".

- Tombol find Tombol ini akam mencari rute angkutan umum dan menampilkannya pada halaman peta.

Pada aplikasi ini penulis akan menampilkan daftar tempat dan daftar rute angkutan umum yang dipakai. Bentuk daftar digunakan penulis karena hasil tempat dan rute angkutan umum banyak. Bentuk daftar yang dapat dipakai di Windows Phone adalah menggunakan ListBox. ListBox akan menampilkan daftar tempat dan daftar rute satu per satu menurun ke bawah.

3.2.3 Analisis Terhadap Siklus Hidup Aplikasi

Aplikasi pada Windows Phone memiliki siklus hidup yang dijelaskan pada bab 2. Pengaturan aplikasi ini diatur sesuai konfigurasi awal sistem operasi Windows Phone. Tetapi pengaturan ini dapat diatur sesuai kebutuhan aplikasi. Karena di aplikasi ini terdapat keadaan yang berbeda dengan konfigurasi awal sistem operasi Windows Phone maka perlu adanya pengaturan ulang.

Saat aplikasi dijalankan, pengguna memasukan lokasi asal dan lokasi tujuan dimasukan sampai pencarian rute dilakukan aplikasi akan berada di keadaan Running. Tetapi ada kemungkinan pengguna berpindah aplikasi atau mematikan layar untuk menghemat daya. Dalam kasus tersebut sistem operasi Windows Phone akan menganggap aplikasi tidak aktif dan aplikasi akan masuk pada keadaan Dormant. Untuk menangani kasus tersebut maka penulis harus menyimpan keadaan dan informasi saat sebelum aplikasi menjadi tidak aktif. Penanganan yang penulis akan lakukan adalah menggunakan metode OnNavigatedFrom. Dengan metode tersebut keadaan aplikasi akan disimpan di memori.

Pada saat aplikasi masuk keadaan Dormant semua thread dan proses akan dihentikan. Pada saat tersebut juga GPS Windows Phone akan terhenti dan tidak akan mengetahui posisi pengguna. GPS akan kembali aktif mengetahui posisi pengguna jika pengguna masuk ke aplikasi dan tentunya membutuhkan waktu untuk pelacakan lokasi. Tetapi Windows Phone mendukung proses di belakang untuk pelacakan GPS selama keluar dari aplikasi atau layar perangkat dimatikan. Maka dari itu aplikasi yang penulis buat akan mendukung pengaksesan lokasi meskipun layar perangkat dimatikan atau berpindah aplikasi.

Ketika aplikasi sudah berada pada keadaan Dormant atau Tombstoned, pengguna masih dapat memulihkan keadaan aplikasi saat aplikasi berada di keadaan Running sebelumnya. Penanganan yang penulis akan lakukan untuk hal tersebut adalah menggunakan metode OnNavigatedTo. Menggunakan metode tersebut akan memulihkan informasi halaman pada keadaan Running sebelumnya.

3.2.4 Analisis Peta

Untuk tampilan peta ada beberapa aspek yang perlu diperhatikan untuk memudahkan pengguna. Aspek yang perlu diperhatikan adalah sebagai berikut.

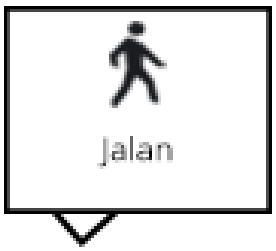
- Pandangan terhadap peta atau kartografi dan mode warna
- Tingkat zoom
- Menampilkan gambar dan keterangan angkutan umum menggunakan pushpin
- Menggambar rute pada peta menggunakan polyline.

Untuk cara pandang peta terdapat 4 pandangan yang disediakan Peta Windows Phone yaitu Road, Aerial, Hybrid, dan Terrain. Aplikasi ini adalah aplikasi pencari rute dan pandangan lebih banyak diarahkan ke jalan. Kebutuhan pengguna adalah nama jalan, kondisi jalan, dan kondisi sekitar. Untuk dasar pandangan tersebut pandangan yang penulis pilih untuk aplikasi ini adalah Road. Tambahan setelah mengatur pandangan peta yaitu mengatur warna dan penulis akan menggunakan mode warna terang sesuai bawaan.

Tampilan awal peta di Windows Phone akan mengeluarkan peta dengan pandangan dunia. Karena aplikasi pencarian rute ini masih terbatas di Pulau Jawa, Indonesia terutama Jawa Barat maka tingkat zoom harus diatur agar mengikuti lokasi pengguna dan di satu daerah saja. Jika pengguna berada di daerah Bandung maka tingkat zoom pada peta disesuaikan pada daerah tersebut. Tingkat zoom dapat dapat diatur dari kode dan XAML. Tingkat zoom yang penulis akan gunakan adalah 14. Tingkat zoom 14 akan menampilkan kota dengan jelas.

Untuk di setiap kota ada satu angkutan umum yang banyak dipakai yaitu angkutan kota(angkot). Namun bagi yang baru pertama mengunjungi suatu daerah dan mencari angkot mungkin akan kesulitan membaca trayek dari angkot tersebut. Namun ada satu cara yang mudah untuk membedakan angkot di setiap rute yaitu dari warna dan coraknya. Agar dapat memudahkan pengguna dan menghindari pengguna saat naik angkot maka Kiri API sudah menyediakan gambar angkot yang seusai dengan setiap rute. Gambar angkot tersebut akan ditepatkan di peta dengan suatu penampung beserta keterangan. Salah satu teknik untuk menempatkan gambar tersebut adalah dengan membuat lapisan terpisah di atas peta tempat gambar tersebut. Untuk hal tersebut penulis akan memanfaatkan Pushpin sebagai lapisan terpisah untuk menaruh gambar dan keterangan angkutan umum. Berikut tampilan pushpin untuk angkot **??** dan pushpin untuk jalan kaki 3.8.

Gambar 3.7: Tampilan pushpin untuk angkot



Gambar 3.8: Tampilan pushpin untuk jalan kaki

Pencarian rute yang penulis gunakan untuk aplikasi yaitu dengan memakai Kiri API. Kiri API akan memberikan kembalian berupa titik-titik rute. Karena hal itu penulis harus menggambar rute tersebut sesuai jalan pada peta. Untuk hal tersebut penulis akan menggunakan Kelas Polyline pada Windows Phone untuk menggambarnya. Polyline yang digambar harus terlihat dengan jelas dan diberi warna yang kontras dengan warna peta. Warna polyline yang penulis akan pilih adalah merah dengan ketebalan 2.

3.2.5 Analisis Pemanfaatan Sumber Data

Aplikasi yang penulis buat memanfaatkan sumber data dari luar. Sumber data yang penulis dapatkan dalam format JSON (Javascript object Notation). Pengambilan sumber data tersebut dilakukan dengan melakukan permintaan HTTP dari *(*Uniform Resource Identifier) (URI). Pemanfaatan sumber data yang penulis gunakan adalah kelas HttpClient.

Metode yang penulis gunakan adalah GetStringAsync. Metode ini akan mengirimkan permintaan melalui URI dan mengembalikan hasilnya dalam bentuk String dan kemajuan data. Karena metode ini mengembalikan hasil dalam bentuk String maka mudah disesuaikan dengan kebutuhan tugas akhir ini. Selanjutnya penulis harus membuat pengurai untuk Sting yang dikeluarkan untuk diolah menjadi informasi yang dibutuhkan

3.2.6 Analisis Kiri API

Kiri API menyediakan 2 parameter untuk permintaan yaitu POST dan GET. Dalam tugas akhir ini penulis akan menggunakan parameter GET. Parameter GET penulis pilih karena dalam tugas akhir ini penulis akan banyak mendapatkan data dan tidak ada data sensitif yang dikirimkan. Untuk hal ini penulis akan mengirim ke URL `http://kiri.travel/handle.php`.

Untuk setiap permintaan terhadap Kiri API dibutuhkan *API key*. Kegunaan *API key* adalah password untuk mengakses Kiri API. *API key* dapat didapatkan di `dev.kiri.travel`. *API key* yang penulis gunakan pada tugas akhir ini adalah 97A7A1157A05ED6F.

Untuk tugas akhir ini penulis akan menggunakan 2 layanan yang ada pada Kiri API. Layanan yang digunakan adalah pencarian lokasi dan routing. Pencarian lokasi adalah layanan untuk menemukan tempat atau nama jalan yang terkait dengan masukan pengguna. Routing adalah layanan untuk menemukan langkah yang harus ditempuh pengguna untuk sampai ke lokasi tujuan dari lokasi asal.

Pemanfaatan layanan pencarian lokasi yaitu dengan parameter GET melalui protokol HTTP. Berikut parameter yang harus dikirimkan beserta keterangannya.

- version: 2
  Karena acuan penulis adalah Kiri API versi maka di parameter version penulis akan menggunakan 2.

- mode: "searchplace"
  Mode "searchplace" digunakan untuk mencari lokasi terkait.

- region: "cgk" untuk Jakarta, "bdo" untuk Bandung, dan "sub" untuk Surabaya
  Karena Kiri API baru tersedia di 3 kota yaitu Jakarta, Bandung, dan Surabaya maka region harus dimasukan untuk pencarian. Region harus dipilih antara "cgk"/"bdo"/"sub" sebagai parameter. Pengguna dapat menentukan masukan region jika menuliskannya pada lokasi asal atau lokasi tujuan. Tetapi, jika pengguna tidak menuliskannya maka sistem yang akan menentukan. Cara penentuan region oleh sistem adalah sistem akan menampung titik tengah dari ketiga region tersebut lalu membandingkannya dengan lokasi pengguna berada. Jarak terdekat antara lokasi pengguna dan salah satu region menandakan pengguna berada di region tersebut.

- querystring: merupakan kata kunci lokasi
- apikey: 16 digit hexadesimal

Format layanan yang dikirim melalui URL adalah kiri.travel/handle.php?version=2&mode=searchplace&region=cgk/bdo/sub&querystring="string"&apikey=97A7A1157A05ED6F.

Penulis mencoba mencari lokasi bip dari kata kunci "bip" yang berada di bandung. Layanan dikirimkan ke URL kiri.travel/handle.php. Berikut format layanan yang penulis kirim: http://kiri.travel/handle.php?version=2&mode=searchplace&region=bdo&querystring=bip&apikey=97A7A1157A05ED6F

Berikut hasil kembalian dari Kiri API:

Listing 3.1: kode kembalian dari pencarian rute


```
{
    "status":"ok",
    "searchresult":[
        {
            "placename":"Hypermart - BIP Plaza",
            "location":"-6.90864,107.61108"
        },
        {
            "placename":"Stroberi - BIP",
            "location":"-6.90834,107.61115"
        },
        {
            "placename":"Kebab Kings (Hypermart BIP)",
            "location":"-6.91503,107.61017"
        },
        {
            "placename":"Pegadaian UPC Bip Mall",
            "location":"-6.90916,107.61052"
        },
        {
            "placename":"Rice Bowl BIP",
            "location":"-6.90873,107.61088"
        },
        {
            "placename":"Gee Eight - Bip",
            "location":"-6.90817,107.61080"
        },
        {
            "placename":"Jonas Photo - BIP",
            "location":"-6.91066,107.61016"
        },
        {
            "placename":"Bip Foodcourt",
            "location":"-6.91081,107.61015"
        },
        {
            "placename":"Mister Baso BIP",
            "location":"-6.90348,107.61709"
        },
        {
            "placename":"JH Moriska - Bip",
            "location":"-6.90868,107.61070"
        }
    ],
    "attributions":null
}
```


Hasil dari kembalian berupa kumpulan *placename* dan *location*. Hasil tersebut akan aplikasi tampung namun yang akan ditampilkan ke pengguna hanya *placename*. Menampilkan *location* tidak efektif menurut penulis karena akan membingungkan pengguna. Dari percobaan yang penulis lakukan, nilai dari *attributions* selalu "null". Karena hal tersebut *attributions* akan penulis abaikan.

Pemanfaatan layanan routing untuk mendapatkan langkah yang harus ditempuh pengguna

untuk mencapai lokasi tujuan dari lokasi asal. Pemanfaatan layanan ini yaitu dengan parameter GET melalui protokol HTTP. Berikut parameter yang harus dikirim:

- version: 2
  Karena acuan penulis adalah Kiri API versi maka di parameter version penulis akan menggunakan 2.
- mode: "findroute"
  Mode "findroute" digunakan untuk mendapatkan langkah yang harus ditempuh menuju lokasi tujuan.
- locale: "en" untuk bahasa Inggris dan "id" untuk bahasa Indonesia.
  Karena aplikasi ini memungkinkan dipakai orang banyak maka penulis putuskan untuk menggunakan bahasa Inggris.
- start: koordinat lokasi awal dalam berupa latitude dan longitude.
  Masukan untuk lokasi awal harus dalam bentuk koordinat. Jika masukan dari pengguna adalah alamat atau tempat maka perlu dicari kordinatnya dahulu.
- finish: koordinat lokasi tujuan dalam berupa latitude dan longitude.
  Masukan untuk lokasi tujuan harus dalam bentuk koordinat. Jika masukan dari pengguna adalah alamat atau tempat maka perlu dicari kordinatnya dahulu.
- presentation: "mobile" untuk perangkat bergerak dan "desktop" untuk komputer.
  Karena aplikasi ini dirancang untuk Windows Phone 8, presentasi yang penulis pilih adalah "mobile".
- apikey: 16 digit hexadesimal

Format layanan yang dikirim melalui URL adalah kiri.travel/handle.php?version=2&mode=findroute&locale=en/id&start=lat,lng&finish=lat,lng&presentation=mobile/desktop&apikey=97A7A1157A05ED6

Penulis mencoba menuju jalan merdeka dari jalan ciumbuleuit. Layanan dikirimkan ke URL kiri.travel/handle.php. Berikut format layanan yang penulis kirim http://kiri.travel/handle.php?version=2&mode=findroute&locale=en&start=-6.8747337,107.6048829&finish=-6.9114646,107.6104887&presentation=mobile&apikey=97A7A1157A05ED6F.

Berikut hasil kembalian dari Kiri API:

Listing 3.2: code *respond* routing

```
{
    "status":"ok",
    "routingresults":[
        {
            "steps":[
                [
                    "walk",
                    "walk",
                    ["-6.8747337,107.6048829","-6.87445,107.60465"],
                    "Walk about 41 meter from your starting point \%fromicon to Jalan Ciumbuleuit \%toicon
                        .",
                    null
                ],
```

```
13                  [
14                      "angkot",
15                      "ciumbuleuitsthalllurus",
16                      ["-6.87445,107.60465","-6.87541,107.60443","-6.87637,107.60421","-6.87734,107.60400",
17                      "-6.87830,107.60378", "-6.87926,107.60356","-6.87926,107.60356","-6.87963,107.60352",
18                      "-6.87978,107.60352","-6.88093,107.60392","-6.88209,107.60433","-6.88209,107.60433",
19                      "-6.88328,107.60490","-6.88328,107.60490","-6.88347,107.60481","-6.88452,107.60459",
20                      "-6.88556,107.60436","-6.88660,107.60413","-6.88764,107.60390","-6.88764,107.60391",
21                      "-6.88782,107.60392","-6.88887,107.60404","-6.88991,107.60416","-6.88991,107.60416",
22                      "-6.89161,107.60428","-6.89161,107.60428","-6.89166,107.60421","-6.89275,107.60424",
23                      "-6.89275,107.60424","-6.89405,107.60408","-6.89405,107.60408","-6.89496,107.60400"],
24                      "Take angkot Ciumbuleuit - St. Hall (lurus) at Jalan Ciumbuleuit \%fromicon, and
                            alight at Jalan Cihampelas
25                      \%toicon about 2.3 kilometer later.",
26                      null
27                  ],
28                  [
29                      "angkot",
30                      "kalapaledeng",
31                      ["-6.89501,107.60403","-6.89562,107.60398","-6.89623,107.60395","-6.89732,107.60401",
32                      "-6.89732,107.60401","-6.89882,107.60414","-6.89882,107.60414","-6.89969,107.60418",
33                      "-6.90071,107.60426","-6.90173,107.60433","-6.90173,107.60433","-6.90297,107.60437",
34                      "-6.90420,107.60440","-6.90420,107.60440","-6.90426,107.60456","-6.90422,107.60481",
35                      "-6.90399,107.60546","-6.90406,107.60617","-6.90454,107.60697","-6.90454,107.60697",
36                      "-6.90512,107.60745","-6.90618,107.60778","-6.90618,107.60778","-6.90643,107.60787",
37                      "-6.90651,107.60807","-6.90675,107.60914","-6.90675,107.60914","-6.90694,107.60939",
38                      "-6.90723,107.60939","-6.90891,107.60943","-6.90891,107.60943","-6.90909,107.60934",
39                      "-6.90914,107.60857","-6.90933,107.60846","-6.91021,107.60887","-6.91021,107.60887",
40                      "-6.91030,107.60897","-6.91028,107.60927","-6.90986,107.61040","-6.90986,107.61040"],
41                      "Take angkot Kalapa - Ledeng at Jalan Cihampelas \%fromicon, and alight at Jalan Aceh
42                      \%toicon about 2.3 kilometer later.",
43                      null
44                  ],
45                  [
46                      "walk",
47                      "walk",
48                      ["-6.90986,107.61040","-6.9114646,107.6104887"],
49                      "Walk about 178 meter from Jalan Aceh \%fromicon to your destination \%toicon.",
50                      null
51                  ]
52                  ],
53                      "traveltime":"30 minutes"
54                  }
55              ]
56  }
```

Setiap langkah akan aplikasi tampung dalam bentuk *array*. Untuk keterangan dan jenis angkutan umum akan penulis keluarkan dalam bentuk pushpin atau daftar. Sedangkan untuk titik-titik kordinat akan digambarkan pada peta. Dari analisa penulis *steps* menunjukan perpindahan angkutan umum yang dipakai, berpindah dari angkutan umum atau jalan, dan dari jalan untuk menaiki angkutan umum. Keterangan yang penulis akan tambahkan harus berada antara setiap *steps* tersebut. Dari analisa penulis juga terdapat kata "%fromicon" dan "%toicon" yang tidak menunjukan sesuatu. Karena itu kedua kata tersebut akan penulis hilangkan agar tidak mengganggu pengguna. Penulis juga akan mengambil gambar angkutan kota dan gambar jalan yang sudah disediakan dari Kiri dengan memanfaatkan URL yang disediakan.

3.2.7 Diagram Use-Case dan Scenario

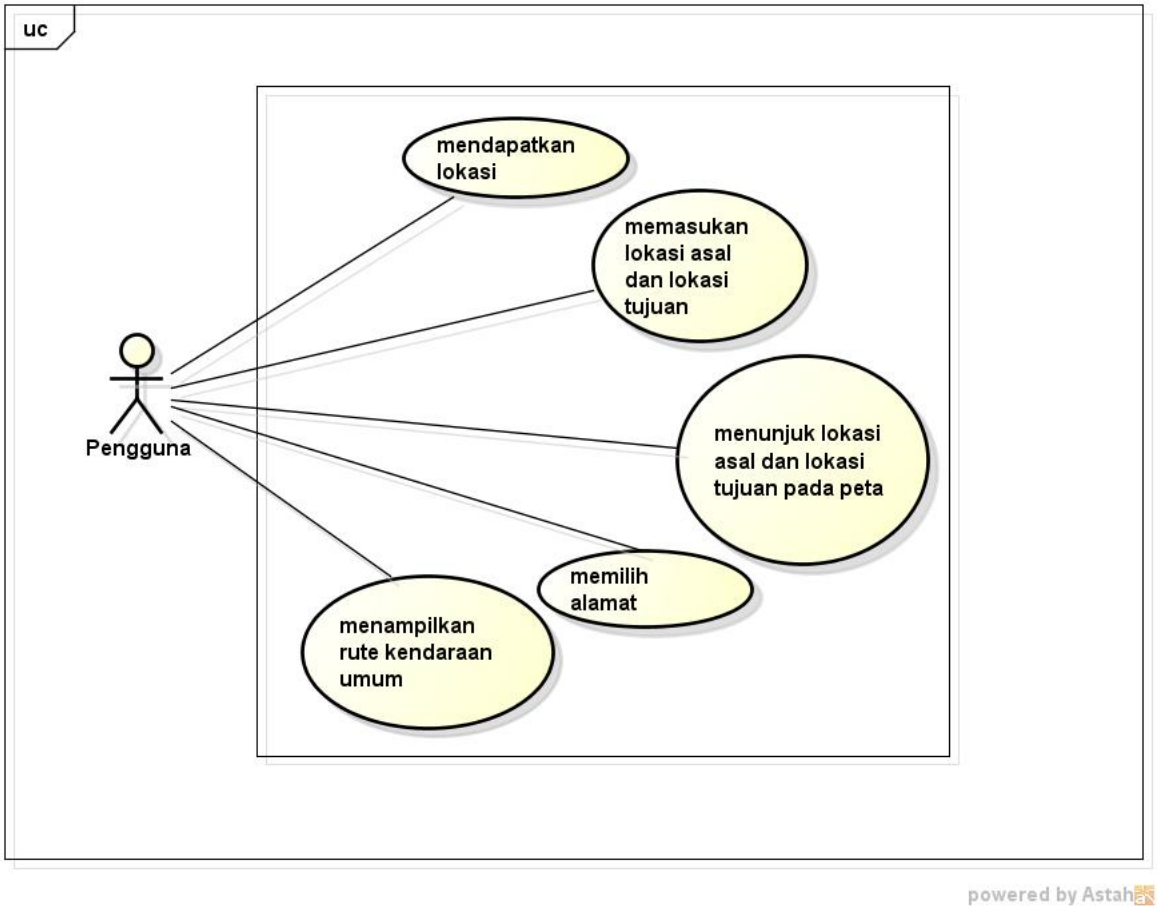
Diagram use-case adalah diagram yang menjelaskan interaksi sistem dengan lingkungan (contoh: pengguna). Berdasarkan analisa di atas maka pengguna dapat:

- Mendapatkan lokasi pengguna berada.
- Memasukan lokasi asal dan lokasi tujuan.
- Menunjuk langsung lokasi asal dan tujuan pada peta.

- Memilih alamat atau tempat dari pilihan yang disediakan.

- Menampilkan rute kendaraan umum dalam bentuk titik dan *pushpin* pada peta atau bentuk daftar dari tempat asal ke tempat tujuan.

Diaram use case saat pengguna mencari rute kendaraan umum dapat dilihat pada gambar (Gambar: 3.9):



Gambar 3.9: Diagram Use Case

Skenario pencarian rute kendaraan umum dapat dilihat pada tabel 3.1 sampai tabel 3.5.

Nama	Mendapatkan lokasi
Aktor	Pengguna
Deskripsi	Mendapatkan lokasi perangkat berada
Kondisi awal	TextBox masih kosong dan pengguna menekan tombol lokasi
Kondisi akhir	Lokasi ditemukan dan TextBox berisi "here"
Skenario utama	Pengguna menekan tombol lalu perangkat akan mencari lokasi perangkat dan TextBox berisi "here"
Eksespsi	lokasi tidak ditemukan jika GPS perangkat tidak aktif

Tabel 3.1: Skenario mandapatkan lokasi

Nama	Masukan lokasi
Aktor	Pengguna
Deskripsi	Memasukan lokasi asal pengguna dan tujuan pengguna (masukan dapat berupa alamat, kordinat, atau tempat)
Kondisi awal	TextBox masih dalam keadaan belum terisi
Kondisi akhir	Lokasi awal dan tujuan sudah dimasukan
Skenario utama	Pengguna mengetikan lokasi awal dan tujuan pada TextBox yang sudah disediakan
Eksespsi	tidak ada

Tabel 3.2: Skenario memasukan lokasi asal dan lokasi tujuan

Nama	Masukan lokasi
Aktor	Pengguna
Deskripsi	Memasukan lokasi asal pengguna dan tujuan pengguna dengan menunjuk pada peta
Kondisi awal	TextBox masih dalam keadaan belum terisi
Kondisi akhir	TextBox terisi dengan "lokasi dari peta"
Skenario utama	Pengguna menunjuk lokasi pada peta dan TextBox terisi dengan "lokasi dari peta"
Eksespsi	tidak ada

Tabel 3.3: Skenario menunjuk lokasi asal dan lokasi tujuan pada peta

Nama	Memilih
Aktor	Pengguna
Deskripsi	Pengguna memilih alamat atau lokasi yang terkait masukan pengguna
Kondisi awal	Lokasi awal dan lokasi tujuan terisi dan pengguna menekan tombol "Find"
Kondisi akhir	Pengguna sudah memilih dan lokasi sudah dapat dipastikan
Skenario utama	Pengguna menekan tombol "Find". Sistem mengembalikan daftar yang berisi alamat atau tempat terkait masukan pengguna
Eksespsi	Lokasi masukan pengguna tidak ditemukan

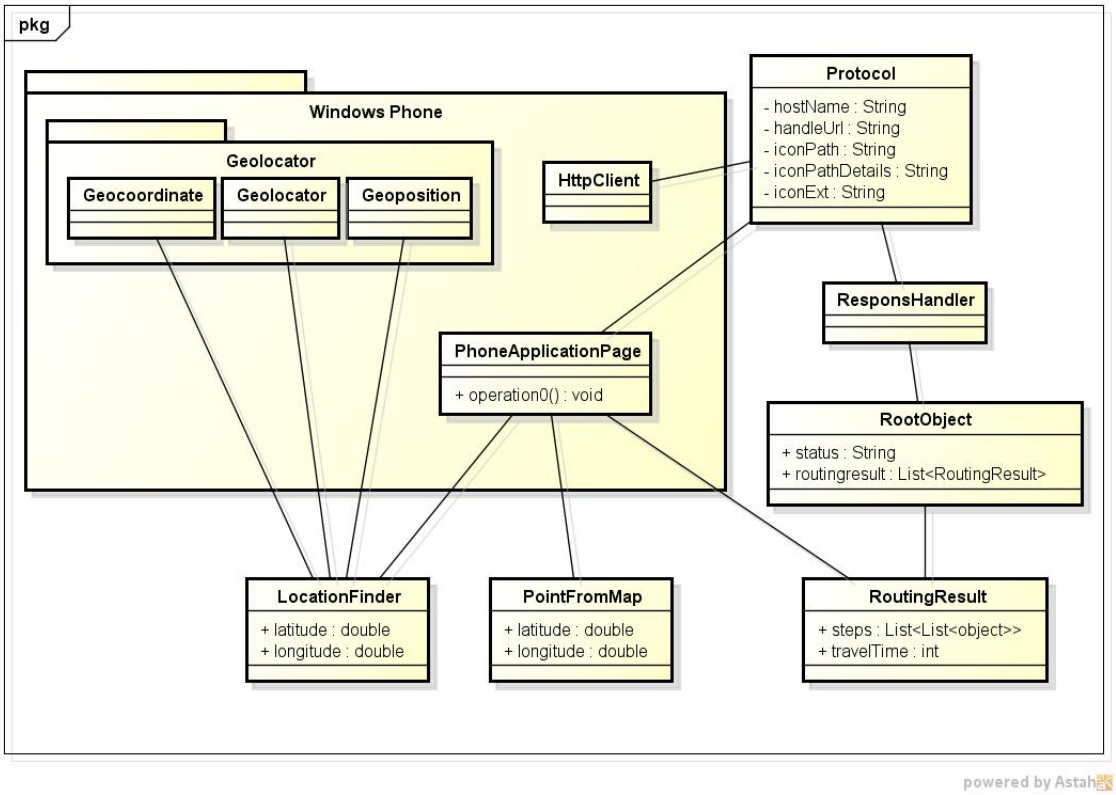
Tabel 3.4: Skenario memilih alamat

Nama	Menampilkan
Aktor	Pengguna
Deskripsi	Lokasi dari pengguna diolah menjadi rute kendaraan umum dari lokasi asal dan lokasi tujuan
Kondisi awal	Lokasi sudah dapat dipastikan
Kondisi akhir	Rute kendaraan umum dimunculkan pada peta dan dalam bentuk daftar
Skenario utama	Lokasi dapat dipastikan sistem. Sistem lalu akan memproses data masukan. Sistem akan mengembalikan hasil rute kendaraan umum pada peta dan dalam bentuk daftar
Eksespsi	Rute kendaraan umum tidak ditemukan

Tabel 3.5: Skenario menampilkan rute kendaraan umum

3.2.8 Class Diagram

Pembuatan kelas diagram didasarkan pada skenario pada sub bab 3.2.7. Kelas diagram dapat dilihat pada gambar 3.10.



Gambar 3.10: Diagram Kelas

Berikut deskripsi kelas pada gambar 3.10.

- Kelas Protocol
  Merupakan kelas yang menampung semua alamat URL yang berhubungan dengan Kiri API. Semua pemanggilan akan ditangani oleh kelas ini.

- Kelas ResponsHandler
  Merupakan kelas yang menangani masukan dari pemanggilan layanan.

- Kelas RootObject
  Merupakan kelas untuk menampung status dan daftar dari layanan *routing* Kiri API. Hasil kembalian akan dipisahkan di kelas ini untuk selanjutnya ditampung di kelas RoutingResult.

- Kelas RoutingResult
  Merupakan kelas untuk menampung setiap langkah dari rute sesuai masukan pengguna. Pada kelas ini juga rute akan digambarkan pada peta.

- Kelas PointFromMap
  Merupakan kelas yang dapat mengetahui lokasi yang ditunjuk pengguna pada peta. Kelas ini akan menyimpan lokasi yang ditunjuk pengguna dalam bentuk latitude dan longitude.

- Kelas LocationFinder
  Merupakan kelas yang digunakan untuk mencari lokasi. kelas ini akan memanfaatkan kelas Geocoordinate untuk mendapatkan lokasi. Setelah lokasi didapatkan dalam bentuk kelas Geoposition maka akan diubah ke latitude dan longitude.

DAFTAR REFERENSI

[1] Microsoft *Windows Phone Silverlight development* 2014 : http://msdn.microsoft.com/library/windows/apps/ff402535.aspx.

[2] Kiri Team *KIRI API v2 Documentation* 2014 : https://bitbucket.org/projectkiri/kiri_api/wiki/KIRI%20API%20v2%20Documentation.

[3] Manning, Paul *Pro Windows Phone App Development* 2013: Apress.

[4] Szostak, Tomasz *Windows Phone 8 Application Development Essentials* 2013: PACKT.