

**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**



# DAFTAR ISI

<b>DAFTAR ISI</b>	<b>iii</b>
<b>DAFTAR GAMBAR</b>	<b>v</b>
<b>DAFTAR TABEL</b>	<b>vi</b>
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	1
1.3 Tujuan . . . . .	2
1.4 Batasan Masalah . . . . .	2
1.5 Metode Penelitian . . . . .	2
1.6 Sistematika Penulisan . . . . .	2
<b>2 DASAR TEORI</b>	<b>5</b>
2.1 Windows Phone . . . . .	5
2.1.1 Lingkungan Kerja . . . . .	5
2.1.2 XAML . . . . .	5
2.1.3 Kontrol terhadap Ponsel . . . . .	6
2.1.4 Siklus Hidup Aplikasi . . . . .	10
2.1.5 Peta di Windows Phone . . . . .	11
2.1.6 Lokasi . . . . .	17
2.1.7 Memanfaatkan Sumber Data . . . . .	20
2.2 Kiri API . . . . .	22
2.2.1 <i>Web Service</i> Penentuan Rute . . . . .	23
2.2.2 <i>Web Service</i> Pencarian Lokasi . . . . .	24
2.2.3 <i>Web Service</i> Menemukan Transportasi Terdekat . . . . .	25
<b>3 ANALISIS</b>	<b>27</b>
3.1 Analisis Aplikasi Sejenis . . . . .	27
3.2 Analisis Aplikasi . . . . .	30
3.2.1 Kebutuhan Aplikasi . . . . .	30
3.2.2 Analisis Kontrol yang Dipakai . . . . .	30
3.2.3 Analisis Terhadap Siklus Hidup Aplikasi . . . . .	31
3.2.4 Analisis Peta . . . . .	32
3.2.5 Analisis Pemanfaatan Sumber Data . . . . .	33
3.2.6 Analisis Kiri API . . . . .	33
3.2.7 Diagram <i>Use Case</i> dan Scenario . . . . .	36
3.2.8 Kelas Diagram . . . . .	39
<b>4 PERANCANGAN</b>	<b>41</b>
4.1 Diagram <i>Sequence</i> . . . . .	41
4.2 Perancangan Kelas . . . . .	43

4.2.1	Kelas <i>PhoneApplicationPage</i>	44
4.2.2	Kelas <i>MainPage</i>	44
4.2.3	Kelas <i>City</i>	46
4.2.4	Kelas <i>BackgroundWorker</i>	46
4.2.5	Kelas <i>Geocoordinate</i>	46
4.2.6	Kelas <i>Geolocator</i>	47
4.2.7	Kelas <i>Geoposition</i>	47
4.2.8	Kelas <i>LocationFinder</i>	47
4.2.9	Kelas <i>Map</i>	48
4.2.10	Kelas <i>HttpClient</i>	49
4.2.11	Kelas <i>JsonConvert</i>	49
4.2.12	Kelas <i>Protocol</i>	49
4.2.13	Kelas <i>RootObjectSearchPlace</i>	50
4.2.14	Kelas <i>SearchResult</i>	51
4.2.15	Kelas <i>RootObjectFindRoute</i>	51
4.2.16	Kelas <i>RoutingResult</i>	51
4.2.17	Kelas <i>Route</i>	51
4.3	Perancangan Antar Muka	53
4.3.1	Antarmuka Kelas <i>MainPage</i>	53
4.3.2	Antarmuka Kelas <i>Map</i>	54
4.3.3	Antarmuka Kelas <i>Route</i>	55
<b>5</b>	<b>IMPLEMENTASI DAN PENGUJIAN APLIKASI</b>	<b>57</b>
5.1	Implementasi	57
5.1.1	Perangkat Keras untuk Implementasi	57
5.1.2	Perangkat Lunak untuk Implementasi	57
5.1.3	Hasil Implementasi	58
5.2	Pengujian	60
5.2.1	Lingkungan Pengujian	61
5.2.2	Pengujian Fungsional	62
5.2.3	Pengujian Experimental	62
5.2.4	Perbandingan Aplikasi Pencari Rute di Android dengan Aplikasi Pencari Rute di Windows Phone	63
<b>6</b>	<b>KESIMPULAN DAN SARAN</b>	<b>67</b>
6.1	Kesimpulan	67
6.2	Saran	67
<b>BIBLIOGRAFI</b>		<b>69</b>

## DAFTAR GAMBAR

2.1	Hirarki navigasi . . . . .	7
2.2	<i>TextBlock</i> , <i>TextBox</i> dan <i>PasswordBox</i> . . . . .	8
2.3	Gambar kontrol pada Windows Phone . . . . .	9
2.4	Antarmuka <i>ListBox</i> . . . . .	10
2.5	Gambar siklus hidup aplikasi[1] . . . . .	10
2.6	Tampilan peta pada Windows Phone . . . . .	12
2.7	<i>cartographic</i> . . . . .	13
2.8	Keluaran Toolkit Pushpin pada peta [2] . . . . .	14
2.9	Tampilan polyline pada Peta . . . . .	14
3.1	Tampilan utama aplikasi Public Transport . . . . .	27
3.2	Menunjuk lokasi pada peta . . . . .	28
3.3	Memberikan daftar nama tempat dan nama jalan terkait . . . . .	28
3.4	Tampilan rute kendaraan umum dalam bentuk daftar . . . . .	29
3.5	Tampilan rute kendaraan umum di peta . . . . .	29
3.6	Tampilan <i>pushpin</i> untuk angkot . . . . .	33
3.7	Tampilan <i>pushpin</i> untuk jalan kaki . . . . .	33
3.8	Diagram <i>use case</i> . . . . .	37
3.9	Diagram Kelas . . . . .	39
4.1	Diagram <i>sequence</i> Mendapatkan Kordinat perangkat . . . . .	41
4.2	Diagram <i>sequence</i> Mendapatkan Kordinat pada Peta . . . . .	42
4.3	Diagram <i>sequence</i> Mendapatkan Rute . . . . .	43
4.4	Diagram Kelas . . . . .	44
4.5	Antarmuka <i>MainPage</i> . . . . .	53
4.6	Antarmuka Daftar Tempat . . . . .	54
4.7	Antarmuka <i>Map</i> . . . . .	54
4.8	Antarmuka <i>Route</i> . . . . .	55
4.9	Antarmuka Rute dalam bentuk Daftar . . . . .	55
5.1	Gambar antarmuka kelas <i>MainPage</i> . . . . .	58
5.2	Gambar antarmuka <i>Splash</i> di kelas <i>MainPage</i> . . . . .	59
5.3	Gambar antarmuka <i>list</i> asal dan <i>list</i> tujuan di kelas <i>MainPage</i> . . . . .	59
5.4	Gambar antarmuka kelas <i>Map</i> . . . . .	60
5.5	Gambar antarmuka menunggu di kelas <i>Route</i> . . . . .	60
5.6	Gambar antarmuka kelas <i>Route</i> . . . . .	61
5.7	Gambar antarmuka <i>list</i> di kelas <i>Route</i> . . . . .	61

## DAFTAR TABEL

2.1	Properti kelas Map . . . . .	15
2.2	Properti <i>Polyline Class</i> . . . . .	17
2.3	Kelas pada <i>Namespace Geolocator</i> . . . . .	18
2.4	Properti pada <i>Geocoordinate</i> . . . . .	19
2.5	Tabel parameter layanan penentuan rute . . . . .	23
2.6	Tabel parameter layanan pencarian lokasi . . . . .	24
2.7	Tabel parameter layanan menemukan transportasi terdekat . . . . .	25
3.1	Skenario mandapatkan lokasi untuk masukan lokasi asal dan lokasi tujuan . . . . .	37
3.2	Skenario memasukan lokasi asal dan lokasi tujuan pada <i>TextBox</i> . . . . .	38
3.3	Skenario menunjuk lokasi asal dan lokasi tujuan pada peta . . . . .	38
3.4	Skenario memilih alamat . . . . .	38
3.5	Skenario menampilkan rute kendaraan umum . . . . .	39
5.1	Tabel Hasil pengujian fungsionalitas terhadap pengguna . . . . .	62
5.2	Tabel Hasil pengujian fungsionalitas . . . . .	64
5.3	Tabel perbedaan . . . . .	65

# 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 kendaraan 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<sup>1</sup>. 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 menggunakan 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.

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

### 1.2 Rumusan Masalah

Sehubung dengan latar belakang diatas timbul permasalahan sebagai berikut:

- Bagaimana membuat aplikasi di Windows Phone?

<sup>1</sup><http://static.kiri.travel/en-about/>

- 1     ● Bagaimana mengintegrasikan Kiri API dengan aplikasi pencari rute kendaraan umum  
2       di Windows Phone?

3     **1.3 Tujuan**

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

- 6     ● Mempelajari cara pembuatan perangkat linak di Windows Phone lalu mengembangkan  
7       aplikasi yang akan dibuat.
- 8     ● Membuat aplikasi di di Windows Phone yang memanfaatkan Kiri API.

9     **1.4 Batasan Masalah**

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

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

16     **1.5 Metode Penelitian**

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

- 19     ● Melakukan studi pustaka mengenai XAML, kontrol dan navigasi di Windows Phone,  
20       Peta di Windows Phone, GPS di Windows Phone dan Kiri API.
- 21     ● Melakukan analisis terhadap aplikasi lain yang menggunakan Kiri API.
- 22     ● Melakukan analisis terhadap dasar teori untuk pembangunan aplikasi Pencarian Rute  
23       Kendaraan Umum untuk Windows Phone.
- 24     ● Melakukan perancangan aplikasi Pencarian Rute Kendaraan Umum untuk Windows  
25       Phone.
- 26     ● Implementasi dari aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- 27     ● Menguji aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- 28     ● Membuat kesimpulan.

29     **1.6 Sistematika Penulisan**

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

33    Bab 2 membahas tentang teori-teori yang akan digunakan dalam tugas akhir ini. Bahasan  
34       yang dijelasakan pada bab ini adalah Windows Phone dan Kiri API. Teori Windows Phone  
35       yang dijelaskan meliputi lingkungan kerja, XAML, kontrol terhadap ponsel, siklus hidup  
36       aplikasi, peta di Windows Phone, lokasi, dan memanfaatkan sumber data. Teori Kiri API

- 1 yang dijelaskan meliputi *web service* penentuan rute, *web service* pencarian lokasi, dan *web service* menemukan transportasi terdekat.
- 3 Bab 3 membahas tentang analisis pembangunan aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone. Pada Bab 3 akan 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.



<sup>1</sup> **BAB 2**

<sup>2</sup> **DASAR TEORI**

<sup>3</sup> Bab ini berisi dasar teori dari pembangunan Aplikasi Pencarian Rute Kendaraan Umum  
<sup>4</sup> untuk Windows Phone. Beberapa teori yang dibahas dalam bab ini adalah XAML, kontrol  
<sup>5</sup> terhadap ponsel, siklus hidup Windows Phone, peta, lokasi , pemanfaatan sumber data, dan  
<sup>6</sup> Kiri API.

<sup>7</sup> **2.1 Windows Phone**

<sup>8</sup> Windows Phone merupakan sistem operasi untuk perangkat bergerak yang dikembangkan  
<sup>9</sup> Microsoft. Pengembangan aplikasi Windows Phone membutuhkan Windows Desktop  
<sup>10</sup> sebagai media pengembangan. Bahasa pemrograman yang digunakan untuk membuat  
<sup>11</sup> perangkat lunak di Windows Phone yaitu C# atau Visual Basic<sup>[2]</sup>.

<sup>12</sup> Pada sub bab [2.1.1](#) sampai [2.1.7](#) akan membahas pemrograman di Windows Phone.  
<sup>13</sup> Pembahasan akan dimulai dengan apa itu Windows Phone dan fitur di Windows Phone  
<sup>14</sup> yang akan digunakan dalam pembangunan perangkat lunak Pencarian Rute Kendaraan di  
<sup>15</sup> Windows Phone.

<sup>16</sup> **2.1.1 Lingkungan Kerja**

<sup>17</sup> Microsoft .NET framework merupakan sebuah perangkat lunak yang dibangun untuk  
<sup>18</sup> membantu dalam pembangunan aplikasi di Windows, Windows Phone, Windows Server,  
<sup>19</sup> and Microsoft Azure<sup>[1]</sup>. Microsoft .NET framework terdiri dari runtime bahasa umum dan  
<sup>20</sup> perpustakaan kelas .NET Framework, yang meliputi kelas, interface, dan jenis nilai yang  
<sup>21</sup> mendukung berbagai teknologi. Microsoft .NET Framework menyediakan lingkungan yang  
<sup>22</sup> mudah dikelola, pengembangan yang disederhanakan, dan integrasi dengan berbagai bahasa  
<sup>23</sup> pemrograman, termasuk Visual Basic dan Visual C#.

<sup>24</sup> Seperti yang telah disebutkan ada dua bahasa pemrograman dalam .NET Framework  
<sup>25</sup> yang dipakai dalam pembangunan aplikasi di Windows Phone 8 yaitu Visual Basic dan  
<sup>26</sup> Visual C#.

<sup>27</sup> Untuk masalah kehandalan keduanya menawarkan kehandalan yang baik. Kelebihan dari  
<sup>28</sup> Visual Basic adalah bahasa pemrograman berorientasi objek yang kuat dan memiliki banyak  
<sup>29</sup> pengembangan fitur di inheritance, polymorphism, interfaces, and overloading<sup>[1]</sup>. Kelebihan  
<sup>30</sup> dari C# yang merupakan pengembangan dari C/C++ adalah sederhana, modern, aman dan  
<sup>31</sup> berorientasi objek<sup>[1]</sup>. Satu hal yang dirasakan penulis adalah kenyamanan ketika memilih  
<sup>32</sup> bahasa .NET tersebut. Akan lebih mudah bagi developer yang menggunakan Visual Basic  
<sup>33</sup> 6.0 untuk menggunakan Visual  
<sup>34</sup> Basic .NET. Tetapi bagi developer yang menggunakan C++ atau java sebelumnya akan  
<sup>35</sup> lebih mudah menggunakan C#.

<sup>36</sup> **2.1.2 XAML**

<sup>37</sup> *Extensible Application Markup Language* (XAML) merupakan bahasa deklaratif yang  
<sup>38</sup> dipakai untuk membuat antarmuka aplikasi. XAML merupakan bahasa yang digunakan

untuk membuat antarmuka di Windows Phone 8[2]. Pada dasarnya penggunaan XAML sama dengan HTML pada pembuatan antarmuka web. XAML dapat menginisialisasi objek dan mengatur properti untuk menunjukkan 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 :

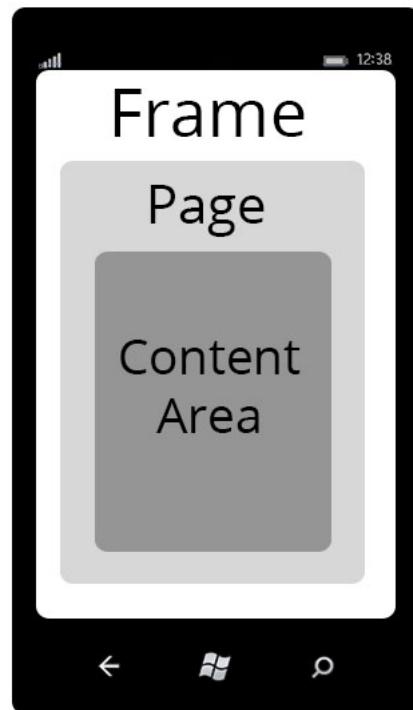
- x:Key : sebuah nama unik untuk menunjuk referensi ke suatu resource atau berkas lain. Nilai ini dapat dipanggil kembali untuk menggunakan resource tersebut.
- x:Class : menunjukkan nama kelas.
- x:Name : menunjukkan nama sebuah obyek dan untuk membedakan antar obyek yang satu dengan obyek yang lain.
- x:Uid : mengidentifikasi elemen objek dalam XAML. Elemen objek 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[1]. 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. Model halaman adalah pengguna berpindah dari satu halaman ke halaman lain dengan konten yang berbeda dan *frame* sebagai pengontrolnya. Setiap antarmuka aplikasi dibungkus dengan *frame*. *Frame* ini yang akan melakukan kontrol terhadap aplikasi dan memungkinkan perpindahan 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 lain-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.

Terdapat 3 jenis panel yang digunakan untuk menangani tata letak yaitu *Grid*, *StackPanel*, dan *Canvas*. Perlu diperhatikan bahwa setiap halaman hanya memiliki satu jenis panel. Berikut adalah 3 jenis panel pada 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* memposisikan 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

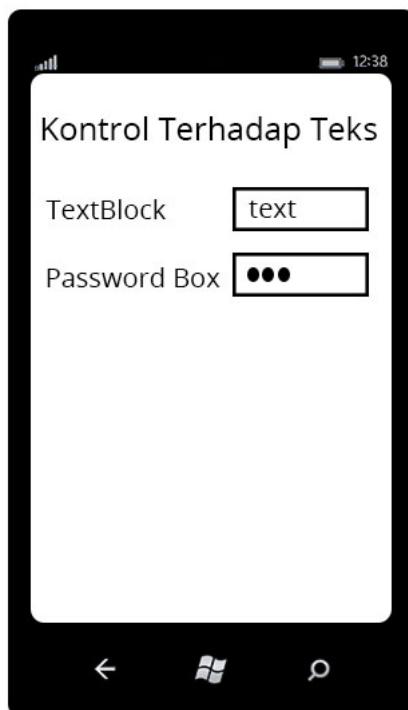
```
18 |     <Grid x:Name="ContentPanel">
19 |     </Grid>
```

### **2.1.3.3 Kontrol Terhadap Teks**

Kontrol terhadap teks akan menampilkan konten yang memiliki tipe *String*. Terdapat berbagai macam kontrol terhadap teks di Windows Phone yaitu *TextBlock*, *TextBox* dan

<sup>1</sup> *PasswordBox*. Ketiga macam kontrol tersebut dibedakan berdasarkan tujuannya. Berikut  
<sup>2</sup> keterangan, gambar 2.2, dan kode pada listing 2.2 kontrol teks.

- <sup>3</sup> • *TextBlock* merupakan tempat menaruh potongan teks yang hanya bisa dilihat.
- <sup>4</sup> • *TextBox* biasanya digunakan untuk teks masukan yang pendek. Tapi bisa juga dipakai  
<sup>5</sup> untuk masukan yang banyak dan beberapa baris.
- <sup>6</sup> • *PasswordBox* biasanya digunakan untuk masukan yang bersifat rahasia. Karakter  
<sup>7</sup> yang dimasukan langsung disamarkan menjadi bentuk titik.



Gambar 2.2: *TextBlock*, *TextBox* dan *PasswordBox*

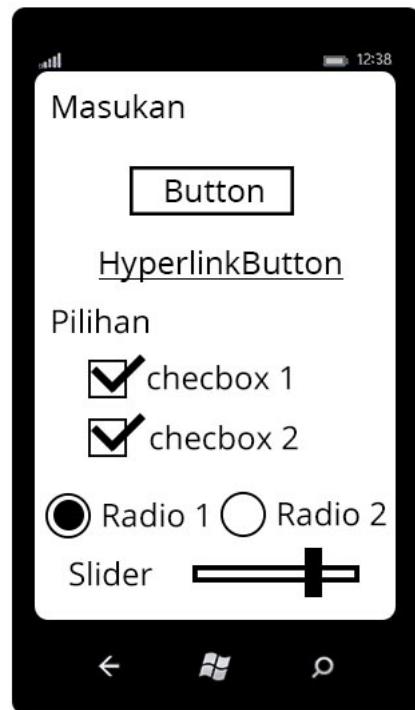
Listing 2.2: Kode untuk menampilkan *TextBlock* dan *TextBox*

```
8 | <TextBlock x:Name="TextBlock1" Text="TextBlock"/>
9 | <TextBox x:Name="TextBox1" Text="TextBox"/>
```

#### <sup>10</sup> 2.1.3.4 Tombol dan Kontrol Pilihan

<sup>11</sup> Tombol memungkinkan pengguna untuk berpindah halaman. Sedangkan kontrol pilihan  
<sup>12</sup> memudahkan dalam memilih. Berikut keterangan, gambar 2.3, dan kode pada listing 2.3  
<sup>13</sup> tombol dan kontrol pilihan.

- <sup>14</sup> • *Button* merupakan kontrol yang dipakai pengguna untuk mengaktifkan *event* tekan.
- <sup>15</sup> • *HyperlinkButton* merupakan kontrol yang menampilkan tautan. Jika *HyperlinkButton*  
<sup>16</sup> ditekan maka akan berpindah ke halaman yang akan dituju.
- <sup>17</sup> • *CheckBox* merupakan kontrol yang memungkinkan pengguna memilih beberapa item.
- <sup>18</sup> • *RadioButton* merupakan kontrol yang memungkinkan pengguna memilih satu pilihan  
<sup>19</sup> dari beberapa pilihan.



Gambar 2.3: Gambar kontrol pada Windows Phone

- 1     • *Slider* merupakan kontrol yang memungkinkan pengguna memilih nilai kisaran dari  
2     jalur yang sudah disediakan.

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

3 |    <Button x:Name="find" Content="Button"/>

4   **2.1.3.5 Kontrol Daftar**

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

- 7     • *ListBox* akan menampilkan daftar *item*. Daftar ini dapat dipilih dengan cara ditekan.  
8     • *LongListSelector* dipakai untuk mengelompokan, menampilkan, dan melakukan peng-  
9     gulungan terhadap daftar yang panjang.



Gambar 2.4: Antarmuka *ListBox*

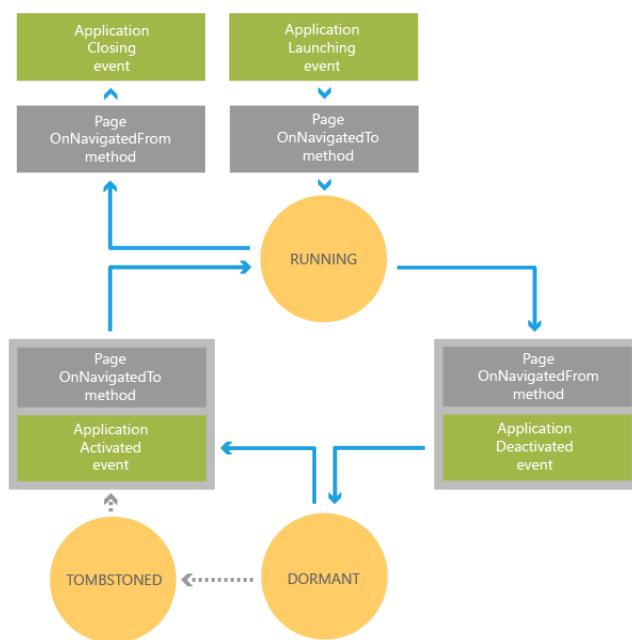
Listing 2.4: Kode untuk menampilkan *listBox*

```

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>
```

#### 8 2.1.4 Siklus Hidup Aplikasi

9 Siklus hidup aplikasi merupakan waktu mulai dari aplikasi dijalankan sampai dengan  
10 aplikasi diberhentikan dari memori. Siklus hidup aplikasi penting diketahui agar pengguna  
11 tidak kecewa (dalam aplikasi ini yaitu kecewa karena aplikasi keluar saat layar perangkat  
12 dimatikan) menggunakan aplikasi serta memastikan sumber daya tersedia (dalam aplikasi ini  
13 yaitu sumber daya GPS). Seringkali pengguna tidak berhati-hati dalam menggunakan aplikasi  
14 kasi, maka dari itu penulis harus memahami kapan aplikasi harus diaktifkan, ditangguhkan,  
15 atau bahkan dinonaktifkan karena sudah tidak digunakan. Gambar 2.5 adalah ilustrasi dari  
16 siklus hidup pada Windows Phone.



Gambar 2.5: Gambar siklus hidup aplikasi[1]

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

##### 20 • The Launching Event

21 Merupakan tampilan awal saat aplikasi dipilih yang memberitahukan pengguna bahwa  
22 aplikasi sedang dijalankan. *Event* ini akan dipanggil ketika aplikasi di jalankan  
23 pertama kali. *Event* ini akan berjalan di belakang (*background processing*) ketika aplikasi  
24 ditutup sementara atau sedang berada pada keadaan *Dormant* atau *Tombstoned*  
25 menjadi *running*.

1     ● *Running*

2       Setelah dijalankan, aplikasi akan masuk ke keadaan *running*. Hal ini akan terus ber-  
3       langsung sampai pengguna berpindah ke halaman depan, atau mundur melewati ha-  
4       laman utama aplikasi. Aplikasi keluar dari keadaan *running* jika perangkat di kunci.  
5       Keadaan *running* masih dapat terjadi saat perangkat di kunci dengan menonaktifkan  
6       *idle detection* pada aplikasi.

7     ● *method OnNavigatedFrom*

8       Merupakan *method* yang dipanggil ketika berpindah ke halaman lain aplikasi. Ketika  
9       *method* ini dipanggil maka aplikasi akan menyimpan keadaan dari halaman sebelum  
10      ditinggalkan. Hal tersebut dibutuhkan agar halaman tersebut bisa dikembalikan ke  
11      keadaan sebelum ditinggalkan saat pengguna ingin kembali ke halaman tersebut. Pe-  
12      manggilan dilakukan ketika berpindah antara halaman di aplikasi atau ketika berpin-  
13      dah aplikasi.

14     ● *The Deactivated Event*

15      *Event* ini akan terjadi ketika pengguna berpindah aplikasi dan menekan tombol "start"  
16      atau menjalankan aplikasi lain. Untuk penanganan *deactivated event*, aplikasi harus  
17      menyimpan data sebelumnya, sehingga data sebelumnya dapat dikembalikan suatu  
18      saat. Windows Phone 8 juga mendukung sistem pengembalian data dengan *State*  
19      *Object*. *State Object* akan digunakan untuk menyimpan keadaan aplikasi sebelum  
20      aplikasi dinonaktifkan.

21     ● *Dormant*

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

27     ● *Tombstoned*

28      Aplikasi yang masuk ke keadaan *Tombstoned* akan dihentikan, namun sistem operasi  
29      akan menyimpan informasi aplikasi pada saat aplikasi berada di keadaan *deactivated*.

30     ● *The Activated Event*

31      *Event* ini dipanggil ketika aplikasi meninggalkan keadaan *Dormant* atau *Tombstoned*.  
32      Operasi ini dilakukan pada latar belakang.

33     ● *The OnNavigatedTo Method*

34      *method* ini dipanggil ketika pengguna berpindah ke halaman yang sebelumnya diting-  
35      galkan. *method* ini akan memeriksa keadaan aplikasi dan memulihkannya jika keadaan  
36      sebelumnya pernah disimpan.

37     ● *The Closing Event*

38      *Event* ini akan tercapai ketika pengguna berpindah mundur keluar dari halaman uta-  
39      ma. Pada kasus ini, aplikasi akan dihentikan dan tidak ada keadaan yang disimpan.

40     **2.1.5 Peta di Windows Phone**

41      Peta yang dipakai di Windows Phone adalah *Windows Phone Maps*. Windows Phone  
42      menawarkan beberapa pilihan dalam tampilan peta mulai dari *cartographic*, pencahayaan  
43      dan pandangan. Selain tampilan pada sub bab ini akan dibahas mengenai mendapatkan  
44      lokasi, petunjuk arah, *MapPolyline* dan *Pushpin*[1].

### 1 2.1.5.1 Penambahan Peta Ke Aplikasi

2 Untuk penambahan peta pada Windows Phone menggunakan kontrol peta. Kontrol peta  
 3 merupakan bagian dari *library* Windows Phone. Dengan begitu untuk dapat menggunakan-  
 4 nya perlu direferensikan. Untuk dapat menggunakannya juga harus ditambah *capability*  
 5 ID\_CAP\_MAP. Setelah hal tersebut dilakukan barulah peta dapat ditampilkan. Berikut  
 6 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

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

Listing 2.6: Menampilkan peta dengan nama MyMap dari kode program

```
8 | { public MapFrom()
9 | { InitializeComponent();
10 | Map MyMap = new Map();
11 | ContentPanel.Children.Add(MyMap);
12 |
13 }
```

### 14 2.1.5.2 Tampilan Peta di Windows Phone

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

- 19 • Menentukan pusat peta berarti menentukan titik tengah sebagai pandangan awal di  
 20 peta. Untuk penentuan titik tengah dibutuhkan 2 nilai yaitu *latitude* dan *longitude*.  
 21 Sedangkan *zoom* merupakan properti untuk mengatur seberapa dekat atau jauh pan-  
 22 dangan yang akan ditampilkan di peta. *Zoom* memiliki nilai yang bisa diatur dari satu  
 23 hingga dua puluh. Kode untuk mengatur titik tengah peta dan tingkat *zoom* dapat  
 24 dilihat pada *listing 2.7* dan *listing 2.8*.

Listing 2.7: Mengatur tingkat zoom dari XAML

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

Listing 2.8: Mengatur tingkat zoom dari dari kode program

```
27 |
28 | { public MapFrom()
29 | { InitializeComponent();
30 | Map MyMap = new Map();
31 |
32 | //Mengatur titik tengah peta
```

```

1     MyMap.Center = new GeoCoordinate(47.6097, -122.3331);
2
3     //mengatur tingkat zoom
4     MyMap.ZoomLevel = 10;
5     ContentPanel.Children.Add(MyMap);
6 }

```

- 7 • *Cartographic* peta di Windows Phone merupakan cara pandang dalam melihat dan  
8 menerjemahkan peta. Terdapat empat jenis *cartographic*, yaitu:

- 9 – *Road*: Tampilan normal 2 dimensi.  
10 – *Aerial*: Tampilan peta yang diambil dari foto di udara.  
11 – *Hybrid*: Tampilan Aerial yang digabung dengan jalan dan label.  
12 – *Terrain*: Menampilkan gambar fisik bumi termasuk ketinggian dan air.



Gambar 2.7: *cartographic*

- 13 • Mode warna yang disediakan Windows Phone ada dua yaitu terang dan gelap. Secara  
14 *default* mode pada peta di Windows Phone adalah terang.  
15 • Tampilan pada Peta di Windows Phone dapat berubah karena hasil diputar, dimi-  
16 ringkan, ditarik, dan diturunkan. Berikut beberapa hal yang dapat diatur sebagai  
17 tampilan di peta.  
18 – *Heading* merupakan representasi dari derajat secara geometri. Derajat ini dide-  
19 finisikan dalam 0 sampai 360 yang dipakai untuk memutar peta. Contoh, 0 atau  
20 360 ke arah utara, 90 ke arah barat, 180 ke arah selatan, dan 270 derajat ke arah  
21 timur.  
22 – *Pitch* merupakan derajat kemiringan dari peta dari sudut pengguna. Contoh,  
23 *Pitch* = 0 berarti melihat dari atas ke bawah sedangkan *Pitch* = 45 berarti  
24 melihat dari samping ke bawah dengan sudut 45 derajat.

#### 25 2.1.5.3 *Pushpin* ke Peta

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

Listing 2.9: Kode untuk menampilkan pushpin

```

32 MapOverlay overlay = new MapOverlay
33 {
34     GeoCoordinate = map.Center,
35     Content = new Border
36     {
37         BorderBrush = new SolidColorBrush(Color.FromArgb(120, 255, 0, 0)),
38         Child = new TextBlock(){Text="Pushpin"},
39         BorderThickness = new Thickness(1),
40         Background = new SolidColorBrush(Color.FromArgb(120, 255, 0, 0)),
41         Width = 80,
42         Height = 60
43     };
44 }

```



Gambar 2.8: Keluaran Toolkit Pushpin pada peta [2]

```

1 MapLayer layer = new MapLayer();
2 layer.Add(overlay);
3
4 map.Layers.Add(layer);

```

#### 5 2.1.5.4 *Polyline* pada Peta

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



Gambar 2.9: Tampilan polyline pada Peta

#### Listing 2.10: Kode untuk menampilkan polyline

```

11 MapPolyline line = new MapPolyline();
12 line.StrokeColor = Colors.Red;
13 line.StrokeThickness = 10;
14 line.Path.Add(new GeoCoordinate(-6.8619546, 107.614441));
15 line.Path.Add(new GeoCoordinate(-6.908693, 107.611185));

```

#### 16 2.1.5.5 *Namespace Control Map*

17 *Namespace* merupakan nama yang dipakai untuk mengatur kelas-kelas. Windows Phone  
18 8 sudah menyediakan *namespace* bawaan untuk mengatur peta. *Namespace* yang disediakan  
19 adalah *Maps.Controls*. *Namespace* ini yang berisi kelas-kelas yang paling sering digunakan  
20 untuk mengatur peta pada Windows Phone. Agar dapat menggunakan kelas pada *na-*  
21 *mespace* tersebut perlu ditambahkan *namespace* dan *capabilities*. *Namespace* yang harus  
22 ditambahkan pada baris awal XAML adalah *Microsoft.Phone.Maps.Controls*. Selanjutnya  
23 ada penambahan *capabilities ID\_CAP\_MAP*. Penambahan *capabilities* ditambahkan pada  
24 *WMAppManifest.xml*.

### 2.1.5.6 Kelas Map

2 Merupakan kelas yang mewakili kontrol map.

Berikut properti yang dapat digunakan pada kelas ini.

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

Tabel 2.1: Properti kelas Map

3

4 Berikut *method* yang dapat digunakan pada kelas ini.

5 • *SetView(LocationRectangle)*

6 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai wilayah geogra-  
7 fisis. *method* ini tidak mengembalikan nilai.

8 • *SetView(GeoCoordinate, Double)*

9 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah dan  
10 tingkat zoom. *method* ini tidak mengembalikan nilai.

11 • *SetView(LocationRectangle, MapAnimationKind)*

12 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis  
13 dan animasi. *method* ini tidak mengembalikan nilai.

14 • *SetView(LocationRectangle, Thickness)*

15 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis  
16 dengan batas tertentu. *method* ini tidak mengembalikan nilai.

17 • *SetView(GeoCoordinate, Double, MapAnimationKind)*

18 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah,  
19 tingkat zoom, dan animasi. *method* ini tidak mengembalikan nilai.

20 • *SetView(GeoCoordinate, Double, Double)*

21 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah,  
22 tingkat zoom, dan *heading*. *method* ini tidak mengembalikan nilai.

23 • *SetView(LocationRectangle, Thickness, MapAnimationKind)*

24 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai wilayah geografis  
25 dengan batas tertentu, dan animasi. *method* ini tidak mengembalikan nilai.

26 • *SetView(GeoCoordinate, Double, Double, MapAnimationKind)*

27 *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah,  
28 tingkat zoom, *heading*, dan animasi. *method* ini tidak mengembalikan nilai.

- 1     ● *SetView(GeoCoordinate, Double, Double)*  
2         method untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah,  
3         tingkat zoom, *heading*, *pitch*. method ini tidak mengembalikan nilai.
- 4     ● *SetView(GeoCoordinate, Double, Double, Double, MapAnimationKind)*  
5         method untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah,  
6         tingkat zoom, *heading*, *pitch*, dan animasi. method ini tidak mengembalikan nilai.
- 7     ● *UpdateLayout*  
8         method yang memastikan semua posisi objek turunan mengikuti tata letak.

#### 9     **2.1.5.7    *Polyline Class***

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

Berikut properti yang dapat digunakan pada kelas ini.

Nama	Deskripsi
<i>Dispatcher</i>	Mendapatkan objek yang terkait.
<i>Path</i>	Mengatur dan mendapatkan kumpulan nilai <i>GeoCoordinates</i> yang membuat <i>polyline</i> .
<i>StrokeColor</i>	Mengatur dan mendapatkan warna garis.
<i>StrokeDashed</i>	Mengatur dan mendapatkan nilai untuk menggambar <i>polyline</i> pustus-pitus.
<i>StrokeThickness</i>	Mengatur dan mendapatkan lebar garis untuk menggambar <i>polyline</i> .

Tabel 2.2: Properti *Polyline Class*

1 Berikut *method* yang dapat digunakan pada kelas ini.

3 • *CheckAccess*

4 *method* yang menentukan bisa atau tidaknya pemanggilan *thread* untuk mengakses  
5 objek.

6 • *ClearValue*

7 *method* yang akan membersihkan nilai lokal

8 • *Finalize*

9 *method* yang dipakai untuk melakukan pembersihan pada sumber daya yang tidak  
10 terpakai sebelum objek dihancurkan.

11 **2.1.5.8 Pushpin Class**

12 Merupakan kelas yang dipakai untuk menggambarkan elemen terpisah diatas peta. Meskipun pushpin merupakan bawaan pada peta untuk menunjuk suatu lokasi tetapi *pushpin*  
13 dari peta tidak dapat diubah-ubah. *Pushpin* pada Windows Phone 8 dapat dibuat sesuai  
14 kebutuhan. Namun ada cara lain dengan menambahkan Windows Phone Toolkit. Windows  
15 Phone Toolkit mempunyai komponen untuk menggambar pushpin diatas peta.  
16

17 **2.1.6 Lokasi**

18 Aplikasi di Windows Phone 8 dapat memanfaatkan lokasi di mana perangkat berada.  
19 Aplikasi dapat melacak lokasi sesaat pengguna atau pelacakan selama periode tertentu.  
20 Data lokasi perangkat berasal dari berbagai sumber termasuk *Global Positioning System*  
21 (*GPS*), *Wireless Fidelity* (*Wi-Fi*), dan jaringan seluler. Ada 2 set API berbeda yang dapat  
22 dimanfaatkan di Windows Phone yaitu *Runtime Location API* dan *.NET Location API*.  
23 Windows Phone *Runtime Location* memiliki keunggulan fitur yang banyak sedangkan *.NET*  
24 *Location* direkomendasikan jika aplikasi ditargetkan pada Windows Phone 7.1 dan Windows  
25 Phone 8[1].

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

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

1 antarmuka. Jika pelacakan lokasi di belakang aplikasi maka tidak ada perubahan pada an-  
 2 tarmuka namun pelacakan dilakukan secara terus menerus. Pelacakan yang terus menerus di  
 3 belakang aplikasi akan membuat keadaan aplikasi cepat dipulihkan dari keadaan *Dormant*.

#### 4 2.1.6.1 Mendapatkan Posisi Pengguna

5 Di Windows Phone 8 telah ada *GeoCoordinate class* yang dapat digunakan untuk menge-  
 6 tahui posisi pengguna. *Geolocator class* dari *Windows.Devices.Geolocation* akan mengemba-  
 7 likan posisi saat ini. Untuk menggunakan *Geolocator*, perlu menghidupkan *ID\_CAP\_LOCATION*  
 8 di *|properties| WMAppManifest.xml*. Dalam mendapatkan posisi perlu diperhatikan status  
 9 dari GPS karena membutuhkan waktu dari awal pengaktifan hingga mendapatkan lokasi  
 10 pengguna secara akurat. Untuk lebih jelas mengenai status posisi dapat dilihat pada nilai  
 11 status dibawah ini.

- 12 • *Ready* : Jika lokasi tersedia.
- 13 • *Initializing* : Jika status penangkap GPS belum memiliki cukup satelit untuk menda-  
 14 patkan posisi yang akurat.
- 15 • *NoData* : Data lokasi belum tersedia. Status ini muncul jika aplikasi sedang mamanggil  
 16 *GetGeopositionAsync()* atau *register*.
- 17 • *Disable* : Status mengindikasikan tidak diperbolehkannya pengaksesan lokasi.
- 18 • *NotInitialized* : Data lokasi belum tersedia. Status ini muncul jika aplikasi belum  
 19 mamanggil *GetGeopositionAsync()* atau *register*.
- 20 • *NotAvailable* : Jika sensor arah mata angin dan lokasi tidak tersedia.

#### 21 2.1.6.2 Namespace Geolocator

22 *Namespace* merupakan nama yang dipakai untuk mengatur kelas-kelas. Windows Phone  
 23 8 sudah menyediakan *namespace* bawaan untuk mengakses lokasi. *Namespace* yang disedi-  
 24 akan adalah *namespace geolocator*. *Namespace* ini akan mengakses lokasi geografis dari  
 25 perangkat dan mendukung pelacakan lokasi dari waktu ke waktu. Agar dapat menggunakan  
 26 kelas pada *namespace* tersebut perlu ditambahkan *namespace* dan *capabilities*. *Namespace*  
 27 yang harus ditambahkan pada baris awal XAML adalah **Windows.Device.Geolocator**.  
 28 Selanjutnya ada penambahan *capabilities ID\_CAP\_LOCATION*. Penambahan *capabiliti-  
 29 es* ditambahkan pada *WMAppManifest.xml*. Kelas yang diatur oleh *namespace geolocator*  
 dapat di lihat pada tabel 2.3[3].

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

Tabel 2.3: Kelas pada *Namespace Geolocator*

30

#### 31 2.1.6.3 Geocoordinate

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

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

Tabel 2.4: Properti pada *Geocoordinate***2.1.6.4 Geolocator**

*Geolocator* merupakan kelas yang mendukung pengaksesan terhadap lokasi. Berikut *method* yang disediakan *Geolocator*:

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

*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.

### 1 2.1.7 Memanfaatkan Sumber Data

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

6 Sebelum data dapat dikirim antar perangkat perlu dilakukan *Serialization*. *Serialization*  
7 disini merupakan proses mentransformasikan objek ke format yang bisa dengan mudah di-  
8 kirim melewati jaringan atau disimpan di database. Formatnya disini berupa string yang  
9 direpresentasikan sebagai objek di XML atau JSON(Javascript Object Notation). Ada beberapa  
10 objek yang dapat melakukan serialisasi, tetapi yang akan dibahas penulis disini hanya  
11 serialisasi JSON[2].

12 Banyak *web service* yang mengembalikan data dalam format JSON. JSON memiliki  
13 format *data-interchange* yang ringan [4]. Karena hal tersebut JSON mudah diurai dan di-  
14 hasilkan oleh mesin. Kurung kurawal mengindikasikan objek, kurung siku berarti array, dan  
15 properti berupa nama dan nilai pasangan yang dipisahkan oleh titik dua. JSON format  
16 memiliki ukuran data yang kecil dan baik untuk penggunaan perangkat bergerak. Untuk  
17 contoh format JSON dapat dilihat di bagian Kiri API pada Bab dua ini karena Kiri API  
18 menggunakan format JSON. Serialisasi menggunakan *DataContractJsonSerializer* membuat  
19 serialisasi mudah untuk menerjemahkan form String JSON ke objek yang dapat langsung  
20 digunakan. *DataContractJsonSerializer* memakai *WriteObject()* untuk serialisasi and *Read-  
21 Object()* untuk de-serialisasi.

#### 22 2.1.7.1 *HttpClient*

23 Merupakan Kelas yang dipakai untuk mengirim permintaan HTTP dan menerima kembali  
24 HTTP dari *Uniform Resource Identifier*(URI) yang dapat diidentifikasi. Berikut  
25 *method* yang disediakan kelas *HttpClient*.

- 26 • *DeleteAsync(Uri)*

27 *method* yang dipakai untuk mengirimkan permintaan DELETE ke URI yang spesifik  
28 sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
29 aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini  
30 membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembalinya  
31 berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut  
32 memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan  
33 dari data yang dikirim.

- 34 • *GetAsync(Uri)*

35 *method* yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai  
36 operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
37 aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan  
38 parameter URI sebagai tujuan dari permintaan. Sedangkan kembalinya berupa objek yang  
39 mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter  
40 yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

- 42 • *GetAsync(Uri,HttpCompletionOption)*

43 *method* yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik sebagai  
44 operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
45 aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan  
46 parameter URI sebagai tujuan dari permintaan dan nilai tambahan yang dimaksukan  
47 sebagai indikasi operasi dianggap selesai. Sedangkan kembalinya berupa objek yang  
48 mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut memiliki 2 parameter  
49 yaitu hasil berupa pesan kembalian dari http dan kemajuan dari data yang dikirim.

1     ● *GetBufferAsync(Uri)*

2       method yang dipakai untuk mengirimkan permintaan *GET* ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembalinya 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.

9     ● *GetInputStreamAsync(Uri)*

10      method yang dipakai untuk mengirimkan permintaan *GET* ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembalinya 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.

17     ● *GetStringAsync(Uri)*

18      method yang dipakai untuk mengirimkan permintaan *GET* ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembalinya 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.

25     ● *PostAsync(Uri)*

26      method yang dipakai untuk mengirimkan permintaan *POST* ke URI yang spesifik sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembalinya 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.

33     ● *SendRequestAsync(HttpRequestMessage)*

34      method yang dipakai untuk mengirimkan permintaan HTTP sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan parameter pesan dari permintaan. Sedangkan kembalinya 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.

40     ● *SendRequestAsync(HttpRequestMessage, HttpCompletionOption)*

41      method yang dipakai untuk mengirimkan permintaan HTTP sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini membutuhkan parameter pesan dari permintaan dan nilai tambahan yang dimaksukan sebagai indikasi operasi dianggap selesai. Sedangkan kembalinya 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.

---

<sup>2</sup><http://msdn.microsoft.com/en-us/library/ms734701%28v=vs.110%29.aspx>

### 1 2.1.7.2 Json.NET

2 Json.NET merupakan *library* untuk melakukan *serialize* dan *deserialize* terhadap objek  
3 dalam .NET. *Library* ini memiliki 2 kelas yaitu kelas JsonConvert dan kelas JsonSerializer.  
4 Berikut merupakan keterangan dari dua kelas tersebut.

- 5     ● JsonConvert merupakan kelas yang berfungsi untuk konversi dari JSON String ke  
6     objek dan sebaliknya. Kelas ini memiliki dua buah *method* yaitu SerializeObject()  
7     dan DeserializeObject().  
  
8     ● JsonSerializer merupakan kelas yang berfungsi untuk konversi dari JSON String ke  
9     objek dan sebaliknya. Kelebihan yang dimiliki kelas ini adalah dapat membaca dan  
10    menulis JSON dari *file text*.

## 11 2.2 Kiri API

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

16    Pemanfaatan Kiri API dengan melakukan permintaan dengan parameter *POST* atau *GET*  
17    lalu Kiri akan mengembalikan hasil dalam format JSON. Permintaan tersebut dikirimkan  
18    ke URL atau *Uniform Resource Locator*. Berikut URL yang disediakan Kiri Api.

- 19     ● <http://preview.kiri.travel/handle.php>  
20       Merupakan URL untuk uji coba. Untuk kemampuannya juga menurut dokumentasi  
21       Kiri API masih tidak stabil.  
  
22     ● <http://kiri.travel/handle.php>  
23       Merupakan URL produksi. Ini merupakan URL yang direkomendasikan untuk mena-  
24       ngani permintaan pengguna.

25    Untuk setiap permintaan membutuhkan *API key* yang didapat dengan mendaftar[5]. Peng-  
26    gunaan API memungkinkan pengaksesan di mana saja dengan menggunakan koneksi inter-  
27    net. Pada sub bab 2.2.1 sampai sub bab 2.2.3 penulis akan membahas beberapa layanan  
28    Kiri API.

29    Berikut langkah-langkah untuk mendapatkan *API key*.

- 30     ● Masuk ke situs [dev.kiri.travel](http://dev.kiri.travel).  
  
31     ● Register dengan memasukan alamat email, nama, dan nama perusahaan.  
  
32     ● Password akan dikirimkan ke alamat email. Tentunya password akan dibuat otomatis  
33       oleh pihak Kiri.  
  
34     ● Login dengan menggunakan password yang dikirim ke alamat email.  
  
35     ● Setelah berhasil login, di menu utama pilih *API Keys Managements*.  
  
36     ● Pilih tombol Add lalu masukan deskripsi penggunaan *API key*.  
  
37     ● *API key* didapat dan dapat digunakan.

### 2.2.1 Web Service Penentuan Rute

Web service penentuan rute merupakan layanan 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.

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

Tabel 2.5: Tabel parameter layanan penentuan rute

Format kembalian layanan penentuan rute dapat dilihat pada *listing 2.11*:

Listing 2.11: Kode kembalian pencarian rute

```

6 | {
7 |     "status": "ok" or "error"
8 |     "routingresults": [
9 |         {
10|             "steps": [
11|                 [
12|                     "walk" or "none" or others,
13|                     "walk" or vehicle_id or "none",
14|                     ["lat_1,lon_1", "lan_2,lon_2", ... "lat_n,lon_n"],
15|                     "human readable description", dependant on locale",
16|                     URL for ticket booking or null (future)
17|                 ],
18|                 [
19|                     "walk" or "none" or others,
20|                     "walk" or vehicle_id or "none",
21|                     ["lat_1,lon_1", "lan_2,lon_2", ... "lat_n,lon_n"],
22|                     "human readable description", dependant on locale",
23|                     URL for ticket booking or null (future)
24|                 ],
25|                 ...
26|             ],
27|             "traveltimes": any text string, null if and only if route is not found.
28|         },
29|         {
30|             "steps": [...],
31|             "traveltimes": "..."
32|         },
33|         {
34|             "steps": [...],
35|             "traveltimes": "..."
36|         },
37|         ...
38|     ]
39| }

```

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 elemen *array* untuk menampung langkah. Berikut keterangan dari setiap *array* yang menampung langkah.

- Indeks 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 kendaraan.
- Indeks ke 1 atau baris 8 pada *listing 2.11* merupakan detail dari indeks 0. Artinya jika indeks 0 menyatakan "walk" berarti indeks 1 harus "walk", "none" berarti indeks 1 harus "none", dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk ditampilkan gambarnya.

- Indeks ke 2 atau baris 9 pada *listing 2.11* adalah deretan nilai tipe *String* yang berisi jalur dalam format "lat,lng". Maksud dari "lat,lng" disini adalah titik awal dan titik akhir dari setiap jalur yang dilewati.
- Indeks ke 3 atau baris 10 pada *listing 2.11* berisi bentuk yang akan ditampilkan kepada pengguna. Informasi yang disampaikan dapat berupa:
  - %fromicon = untuk menunjukkan ikon "from". Biasanya untuk mode presentasi di perangkat bergerak.
  - %toicon = untuk menunjukkan ikon "to". Biasanya untuk mode presentasi di perangkat bergerak.
- Indeks 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]/[means_details].png)
- [http://kiri.travel/images/means/\[means\]/baloon/\[means\\_details\].png](http://kiri.travel/images/means/[means]/baloon/[means_details].png)

Nilai [means] dapat diambil dari indeks 0 nilai kembalian dan nilai [means\_details] dapat diambil dari indeks 1 nilai kembalian.

### 2.2.2 Web Service Pencarian Lokasi

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

<i>version</i>	2	Memberitahukan bahwa layanan yang dipakai adalah protokol veris 2
<i>mode</i>	"searchplace"	Mengintruksikan layanan untuk mencari tempat
<i>region</i>	"cgk" or "bdo" or "sub"	Kota yang akan dicari tempatnya
<i>querystring</i>	teks apa saja dengan minimum text satu karakter	<i>Query string</i> yang akan dicari menggunakan layanan ini
<i>apikey</i>	16-digit heksadesimal	<i>API key</i> 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

```

23 {
24   "status": "ok" or "error"
25   "searchresult": [
26     {
27       "placename": "place_name"
28       "location": "lat,lon"
29     },
30     {
31       "placename": "place_name"
32       "location": "lat,lon"
33     },
34     ...
35   ],
36   "attribution": [
37     "attribution_1", "attribution_2", ...
38   ]
39 }
```

- 1 Berikut maksud dari *listing 2.12*:
- 2 Ketika pencarian lokasi sukses dilakukan maka status akan memberitahukan "ok" seperti di  
3 baris 2. Selanjutnya akan ditampilkan hasil dari lokasi yang ada beserta atributnya. Berikut  
4 keterangan dari format dari pencarian lokasi:
- 5 • "searchresult" (pada baris 4 sampai 7, 8 sampai 11, dan seterusnya) berisi array dari  
6 tempat:
- 7 – placename: nama tempat
- 8 – location: latitude dan longitude dari tempat
- 9 • "attributions" berisi kumpulan nilai yang berisikan atribut tambahan untuk dimun-  
10 culkan.

### 11 2.2.3 Web Service Menemukan Transportasi Terdekat

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

<i>version</i>	2	Memberitahukan bahwa layanan yang dipakai adalah protokol versi 2
<i>mode</i>	"nearbytransports"	Menginstruksikan layanan untuk mencari rute transportasi terdekat
<i>start</i>	<i>latitude</i> dan <i>longitude</i>	Kota yang akan dicari tempatnya
<i>apikey</i>	16-digit hexadesimal	<i>API key</i> yang digunakan

Tabel 2.7: Tabel parameter layanan menemukan transportasi terdekat

15 Format kembalian layanan menemukan transportasi terdekat dapat dilihat pada *lis-*  
16 *ting 2.13*.

Listing 2.13: Kode kembalian menemukan lokasi terdekat

```

17 {
18   "status": "ok" or "error"
19   "nearbytransports": [
20     [
21       "walk" or "none" or others,
22       "walk" or vehicle_id or "none",
23       text string,
24       decimal value
25     ],
26     [
27       "walk" or "none" or others,
28       "walk" or vehicle_id or "none",
29       text string,
30       decimal value
31     ],
32     ...
33   ],
34 }
```

35 Berikut maksud dari *listing 2.13*:

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

- 39 • Indeks ke 0 atau baris 5 pada *listing 2.13* dapat berisi "walk" atau "none" atau  
40 "others". Artinya jika "walk" berarti berjalan kaki, "none" jika rute tidak ditemukan  
41 dan "others" berarti menggunakan kendaraan.
- 42 • Indeks ke 1 atau baris 6 pada *listing 2.13* merupakan detail dari indeks 0. Artinya jika  
43 indeks 0 "walk" berarti indeks 1 harus "walk", "none" berarti indeks 1 harus "none"  
44 dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk ditampilkan  
45 gambarnya.

- <sup>1</sup> • Indeks ke 2 atau baris 7 pada *listing 2.13* berisi nama kendaraan.
- <sup>2</sup> • Indeks ke 3 atau baris 8 pada *listing 2.13* berisi jarak dengan satuan kilometer.

1

## BAB 3

2

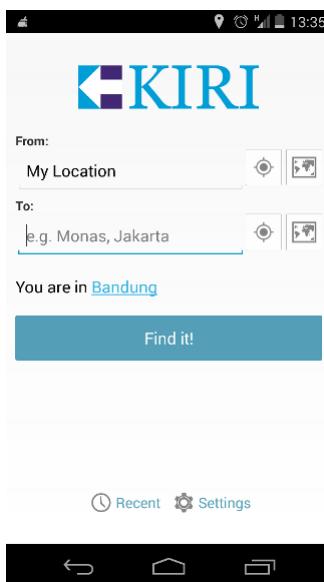
## ANALISIS

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

6 **3.1 Analisis Aplikasi Sejenis**

7 Aplikasi sejenis yang penulis temui bernama Public Transport<sup>1</sup>. Namun aplikasi Public  
8 Transport tersebut hanya dapat dijalankan di sistem aplikasi android. Aplikasi Public  
9 Transport ini memanfaatkan Kiri API. Aplikasi tersebut penggunaannya sederhana. Di halaman awal pengguna dapat mengetikkan lokasi awal dan tujuan. Selain dengan mengetik  
10 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  
11 sistem akan mengeluarkan hasil pencarian rute.  
12

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



Gambar 3.1: Tampilan utama aplikasi Public Transport

15 Gambar 3.1 menunjukkan halaman utama aplikasi Public Transport. Di halaman ini pengguna dapat memasukan lokasi asal dan lokasi tujuan. Cara memasukan lokasi ada 2 macam yaitu dengan mengetik dan menunjuk pada peta dengan mengetuk tombol peta. Bila pengguna ingin menunjuk lokasi pengguna berada dapat dilakukan dengan mengetuk tombol kordinat. Tersedia juga pelihan kota yang dapat dipilih oleh pengguna.

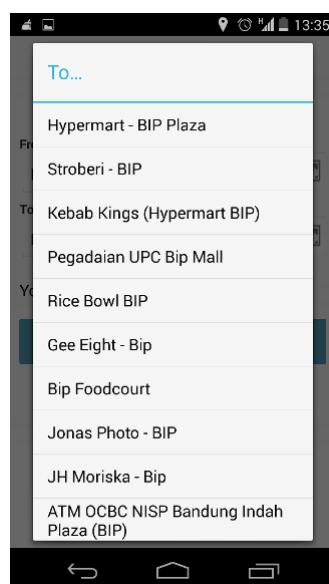
20 Gambar 3.2 jika pengguna sudah mengetahui lokasi namun tidak tahu nama lokasi. Pada halaman ini pengguna diarahkan untuk menemukan lokasi pada peta dan mengetuk

<sup>1</sup><https://play.google.com/store/apps/details?id=travel.kiri.smarttransportapp>



Gambar 3.2: Menunjuk lokasi pada peta

- 1 lokasi tersebut untuk memilihnya.



Gambar 3.3: Memberikan daftar nama tempat dan nama jalan terkait

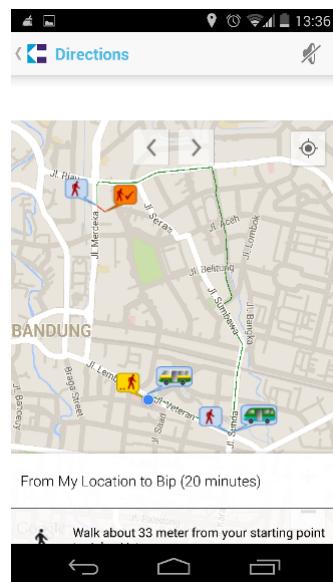
- 2 Pada gambar 3.3 pengguna dapat memilih nama tempat terkait. Pemilihan didasarkan
- 3 sesuai masukan pengguna untuk memastikan tempat asal maupun tempat tujuan. Jika
- 4 nama tempat sudah jelas maka tidak akan ada halaman ini.

5 Pada gambar 3.4 menampilkan daftar rute kendaraan umum yang harus dinaiki beserta  
6 gambar untuk mempermudah pengguna. Selain itu disertakan juga jarak dan perkiraan  
7 waktu sampai di lokasi tujuan.

8 Pada gambar 3.5 menampilkan rute kendaraan umum dan jalur yang harus dilalui pada  
9 peta. Dengan cara ini pengguna dapat mengetahui posisi dan jalur yang harus dilalui.



Gambar 3.4: Tampilan rute kendaraan umum dalam bentuk daftar



Gambar 3.5: Tampilan rute kendaraan umum di peta

## 1 3.2 Analisis Aplikasi

2 Aplikasi akan dibuat menggunakan bahasa pemograman C#. Aplikasi yang digunakan  
3 untuk membangun Aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone adalah  
4 Visual Studio Express 2012. Pada sub bab ini akan dibahas kebutuhan aplikasi, analisis  
5 kontrol yang dipakai, analisis terhadap siklus hidup aplikasi, analisis peta, analisis pemanfa-  
6 atan sumber data, analisa Kiri API, diagram *use case*, dan diagram kelas dari aplikasi yang  
7 akan dibangun.

### 8 3.2.1 Kebutuhan Aplikasi

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

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

21 Untuk hasil keluaran ada dua tipe seperti aplikasi peta lainnya. Kedua tipe tersebut ada-  
22 lah bentuk daftar dan bentuk peta. Bentuk daftar memudahkan dalam melihat tiap langkah  
23 rute. bentuk daftar memudahkan pengguna dalam melihat arah dan posisi lingkungan pada  
24 rute yang dipilih.

25 Aplikasi yang penulis bangun didasarkan pada kebutuhan sebagai berikut.

- 26 1. Pengguna dapat memasukan lokasi asal dan lokasi tujuan pada *TextBox* yang disedi-  
27 akan atau menunjuk langsung lokasi pada peta.
- 28 2. Mendapatkan lokasi terkait menurut lokasi yang dimasukan pengguna.
- 29 3. Menampilkan hasil rute angkutan umum dari lokasi asal ke lokasi tujuan.

### 30 3.2.2 Analisis Kontrol yang Dipakai

31 Dari kebutuhan yang telah disebutkan diatas penulis menyadari pentingnya kontrol yang  
32 harus dipakai. Kontrol yang dimaksud termasuk tata letak, teks, pilihan, dan daftar. Ke-  
33 butuhan akan kontrol penting bukan hanya untuk kebutuhan tapi memudahkan pengguna.

34 Untuk kontrol tata letak penulis membayangkan pengaturan yang tertata rapih dan  
35 beberapa elemen dalam satu baris atau kolom. Tetapi juga penulis tidak mengharapkan  
36 penggunaan kontrol tata letak yang rumit. Dari hasil pengamatan penulis kontrol tata  
37 letak yang cocok adalah Grid. Kontrol tata letak ini penulis pilih karena mudah diposisikan  
38 sesuai baris dan kolom. Selain itu tampilan Grid akan menyesuaikan jika layar diputar dari  
39 posisi pemandangan ke posisi potret dan sebaliknya.

40 Kontrol terhadap teks juga diperlukan untuk aplikasi. Kebutuhan yang diperlukan ada-  
41 lah mengeluarkan potongan teks yang dapat dibaca dan tempat pengguna memasukan teks.  
42 Untuk mengeluarkan teks yang dapat dilihat penulis akan menggunakan *TextBlock*. Te-  
43 xtBlock digunakan untuk menampilkan tulisan "from" dan "to" pada halaman utama apli-  
44 kasi. Untuk masukan pengguna terhadap aplikasi penulis akan menyediakan *TextBox* sebagai  
45 tempat teks. *TextBox* digunakan sebagai masukan untuk lokasi asal dan lokasi tujuan.

46 Suatu aplikasi tentunya tidak hanya mempunyai satu halaman. Sama hal dengan aplikasi  
47 yang penulis buat memiliki beberapa halaman yang mempunyai tugas berbeda. Karena hal

1 tersebut dibutuhkan kontrol untuk berpindah dari satu halaman ke halaman lain. Kontrol  
2 yang dibutuhkan yaitu kontrol tombol. Kontrol tombol akan mengeksekusi *event click* yang  
3 memungkinkan pindah halaman dan melakukan perintah. Kontrol tombol akan penulis  
4 gunakan untuk berpindah ke halaman peta, menemukan lokasi pengguna, dan pencarian  
5 rute. Pada Gambar ?? terdapat 5 tombol yaitu tombol map pada bagian from, tombol here  
6 pada bagian from, tombol map pada bagian to, tombol here pada bagian to, dan tombol  
7 find. Berikut kegunaan dari tombol-tombol tersebut.

- 8     ● Tombol map pada bagian from

9         Tombol untuk berpindah dari halaman utama menuju halaman peta. Pada halaman  
10      peta pengguna dapat menunjuk lokasi asal dan kembali lagi ke halaman utama. Saat  
11      kembali ke halaman utama lokasi yang dipilih akan disimpan dan pada *TextBox* bagian  
12      from akan tertulis "lokasi dari peta".

- 13     ● Tombol map pada bagian from

14         Tombol untuk mencari lokasi pengguna. Setelah tombol ini di tekan maka lokasi  
15      pengguna akan disimpan dan pada bagian *TextBox* bagian from akan tertulis "here".

- 16     ● Tombol map pada bagian to

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

- 21     ● Tombol map pada bagian to

22         Tombol untuk mencari lokasi pengguna. Setelah tombol ini di tekan maka lokasi  
23      pengguna akan disimpan dan pada bagian *TextBox* bagian to akan tertulis "here".

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

26         Pada aplikasi ini penulis akan menampilkan daftar tempat dan daftar rute angkutan  
27      umum yang dipakai. Bentuk daftar digunakan penulis karena hasil tempat dan rute ang-  
28      kutan umum akan banyak. Bentuk daftar yang dapat dipakai di Windows Phone adalah  
29      menggunakan *ListBox*. *ListBox* akan menampilkan daftar tempat dan daftar rute satu per  
30      satu menurun ke bawah.

### 31     3.2.3 Analisis Terhadap Siklus Hidup Aplikasi

32         Aplikasi pada Windows Phone memiliki siklus hidup yang dijelaskan pada bab 2.1.4.  
33      Pengaturan aplikasi ini diatur sesuai konfigurasi awal sistem operasi Windows Phone. Tetapi  
34      pengaturan ini dapat diatur sesuai kebutuhan aplikasi. Karena di aplikasi ini terdapat  
35      keadaan yang berbeda dengan konfigurasi awal sistem operasi Windows Phone maka perlu  
36      dilakukan pengaturan ulang siklus hidup.

37         Saat aplikasi dijalankan, pengguna memasukan lokasi asal dan lokasi tujuan. Setelah  
38      memasukan lokasi pengguna akan mencari rute. Ketika rute berhasil ditemukan aplikasi akan  
39      berada di keadaan Running. Tetapi ada kemungkinan pengguna berpindah aplikasi atau  
40      mematikan layar untuk menghemat daya. Dalam kasus tersebut sistem operasi Windows  
41      Phone akan menganggap aplikasi tidak aktif dan aplikasi akan masuk pada keadaan *dormant*.  
42      Untuk menangani kasus tersebut maka penulis harus menyimpan keadaan dan informasi  
43      saat sebelum aplikasi menjadi tidak aktif. Penanganan yang penulis akan lakukan adalah  
44      menggunakan *method OnNavigatedFrom()*. Dengan *method* tersebut keadaan aplikasi akan  
45      disimpan di memori.

46         Pada saat aplikasi masuk keadaan Dormant semua *thread* dan proses akan dihentikan.  
47      Pada saat tersebut juga GPS Windows Phone akan terhenti dan tidak akan mengetahui  
48      posisi pengguna. GPS akan kembali aktif mengetahui posisi pengguna jika pengguna masuk

1 ke aplikasi dan tentunya membutuhkan waktu untuk pelacakan lokasi. Tetapi Windows  
2 Phone mendukung proses di belakang untuk pelacakan GPS selama keluar dari aplikasi  
3 atau layar perangkat dimatikan. Maka dari itu aplikasi yang penulis buat akan mendukung  
4 pengaksesan lokasi meskipun layar perangkat dimatikan atau berpindah aplikasi.

5 Ketika aplikasi sudah berada pada keadaan *Dormant* atau *Tombstoned*, pengguna masih  
6 dapat memulihkan keadaan aplikasi saat aplikasi berada di keadaan *Running* sebelumnya.  
7 Penanganan yang penulis akan lakukan untuk hal tersebut adalah menggunakan *method*  
8 *OnNavigatedTo()*. Menggunakan *method* tersebut akan memulihkan informasi halaman pada  
9 keadaan *Running* sebelumnya.

10 **3.2.4 Analisis Peta**

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

- 13 • Pemetaan terhadap peta atau *cartographic* dan mode warna  
14 • Tingkat *zoom*  
15 • Menampilkan gambar dan keterangan angkutan umum menggunakan *pushpin*  
16 • Menggambar rute pada peta menggunakan *polyline*

17 Untuk cara pandang peta terdapat 4 pandangan yang disediakan peta di Windows Phone  
18 yaitu *Road*, *Aerial*, *Hybrid*, dan *Terrain*. Aplikasi ini adalah aplikasi pencari rute dan pan-  
19 dangan lebih banyak diarahkan ke jalanan perkotaan. Kebutuhan pengguna adalah nama  
20 jalan, kondisi jalan, dan kondisi sekitar. Dari dasar pandangan tersebut pandangan yang  
21 penulis pilih untuk aplikasi ini adalah *Road*. Tambahan setelah mengatur pandangan peta  
22 yaitu mengatur warna dan penulis akan menggunakan mode warna terang sesuai bawaan  
23 peta di Windows Phone.

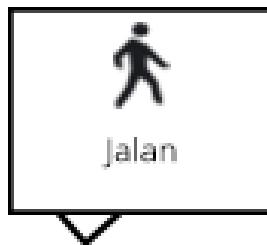
24 Tampilan awal peta di Windows Phone akan mengeluarkan peta dengan pandangan du-  
25 nia. Karena aplikasi pencarian rute ini masih terbatas di Pulau Jawa, Indonesia terutama  
26 Jawa Barat maka tingkat zoom harus diatur agar mengikuti lokasi pengguna dan di satu  
27 daerah saja. Jika pengguna berada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

31 Di setiap kota ada satu angkutan umum yang banyak dipakai yaitu angkutan kota(angkot).  
32 Namun bagi yang baru pertama mengunjungi suatu daerah dan mencari angkot mungkin  
33 akan kesulitan membaca trayek dari angkot tersebut. Namun ada satu cara yang mudah  
34 untuk membedakan angkot di setiap rute yaitu dari warna dan coraknya. Agar dapat me-  
35 mudahkan pengguna dan menghindari pengguna dari kesalahan naik angkot maka Kiri API  
36 sudah menyediakan gambar angkot yang seusai dengan setiap rute. Gambar angkot terse-  
37 but akan ditempatkan di peta dengan suatu penampung beserta keterangannya. Salah satu  
38 teknik untuk menempatkan gambar tersebut adalah dengan membuat lapisan terpisah di  
39 atas peta tempat gambar tersebut. Untuk hal tersebut penulis akan memanfaatkan Pushpin  
40 sebagai lapisan terpisah untuk menaruh gambar dan keterangan angkutan umum. Berikut  
41 tampilan *pushpin* untuk angkot [3.6](#) dan *pushpin* untuk jalan kaki [3.7](#).

42 Pencarian rute yang penulis gunakan untuk aplikasi yaitu dengan memakai Kiri API.  
43 Kiri API akan memberikan kembalian berupa titik-titik rute perjalanan dari lokasi asal ke  
44 lokasi tujuan. Karena hal itu penulis harus menggambar rute tersebut sesuai jalan pada  
45 peta. Untuk hal tersebut penulis akan menggunakan *Polyline* pada Windows Phone untuk  
46 menggambarnya. *Polyline* yang digambar harus terlihat dengan jelas dan diberi warna yang  
47 kontras dengan tampilan peta. Warna polyline yang penulis akan pilih adalah merah dengan  
48 ketebalan 2.



Gambar 3.6: Tampilan *pushpin* untuk angkot



Gambar 3.7: Tampilan *pushpin* untuk jalan kaki

### <sup>1</sup> 3.2.5 Analisis Pemanfaatan Sumber Data

<sup>2</sup> Aplikasi yang penulis buat memanfaatkan sumber data dari luar. Sumber data yang pe-  
<sup>3</sup> nulis dapatkan dalam format JSON (*Javascript object Notation*). Pengambilan sumber data  
<sup>4</sup> tersebut dilakukan dengan melakukan permintaan HTTP dari *Uniform Resource Identifier*  
<sup>5</sup> / URI. Pemanfaatan sumber data yang penulis gunakan adalah kelas *HttpClient*.

<sup>6</sup> *method* yang penulis gunakan adalah *GetStringAsync()*. *method* ini akan mengirimkan  
<sup>7</sup> permintaan melalui URI dan mengembalikan hasilnya dalam tipe data *String* dan kemajuan  
<sup>8</sup> data. Karena *method* ini mengembalikan hasil dalam tipe data *String* maka mudah disesua-  
<sup>9</sup> ikan dengan kebutuhan tugas akhir ini. Selanjutnya penulis harus membuat pengurai untuk  
<sup>10</sup> keluaran untuk diolah menjadi informasi yang dibutuhkan.

### <sup>11</sup> 3.2.6 Analisis Kiri API

<sup>12</sup> Kiri API menyediakan 2 parameter untuk permintaan yaitu *POST* dan *GET*. Da-  
<sup>13</sup> lam tugas akhir ini penulis akan menggunakan parameter *GET*. Parameter **GET** penu-  
<sup>14</sup> lis pilih karena dalam tugas akhir ini penulis akan banyak mendapatkan data dan tidak  
<sup>15</sup> ada data sensitif yang dikirimkan. Untuk hal ini penulis akan mengirim ke URL <http://kiri.travel/handle.php>.

<sup>16</sup> Untuk setiap permintaan terhadap Kiri API dibutuhkan *API key*. Kegunaan *API key*  
<sup>17</sup> adalah password untuk mengakses Kiri API. *API key* dapat didapatkan di <dev.kiri>.  
<sup>18</sup> *travel*. *API key* yang penulis gunakan pada tugas akhir ini adalah 97A7A1157A05ED6F.

<sup>19</sup> Untuk tugas akhir ini penulis akan menggunakan 2 layanan yang ada pada Kiri API.  
<sup>20</sup> Layanan yang digunakan adalah pencarian lokasi dan penentuan rute. Pencarian lokasi  
<sup>21</sup> adalah layanan untuk menemukan tempat atau nama jalan yang terkait dengan masukan  
<sup>22</sup> pengguna. Penentuan rute adalah layanan untuk menemukan langkah yang harus ditempuh  
<sup>23</sup> pengguna untuk sampai ke lokasi tujuan dari lokasi asal.

<sup>24</sup> Pemanfaatan layanan pencarian lokasi yaitu dengan parameter *GET* melalui protokol  
<sup>25</sup> HTTP. Berikut parameter yang harus dikirimkan beserta keterangannya.

- <sup>26</sup> • *version: 2*

<sup>27</sup> Karena acuan penulis adalah Kiri API versi maka di parameter *version* penulis akan

1 menggunakan 2.

- 2 • *mode*: "searchplace"

3 Mode "searchplace" digunakan untuk mencari lokasi terkait.

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

- 13 • *querystring*: merupakan kata kunci lokasi

- 14 • *apikey*: 16 digit heksadesimal

15 Format layanan yang dikirim melalui URL adalah [kiri.travel/handle.php?version=2&mode=searchplace&region=cgk/bdo/sub&querystring=string&apikey=97A7A1157A05ED6F](http://kiri.travel/handle.php?version=2&mode=searchplace&region=cgk/bdo/sub&querystring=string&apikey=97A7A1157A05ED6F).

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

19 <http://kiri.travel/handle.php?version=2&mode=searchplace&region=bdo&querystring=bip&apikey=97A7A1157A05ED6F>

20 Berikut hasil kembalian dari Kiri API:

Listing 3.1: Kode kembalian dari pencarian rute

```

25 {
26     "status ":"ok",
27     "searchresult ":[
28         {
29             "placename ":"Hypermart - BIP Plaza",
30             "location ":"-6.90864,107.61108"
31         },
32         {
33             "placename ":"Stroberi - BIP",
34             "location ":"-6.90834,107.61115"
35         },
36         {
37             "placename ":"Kebab Kings (Hypermart BIP)",
38             "location ":"-6.91503,107.61017"
39         },
40         {
41             "placename ":"Pegadaian UPC Bip Mall",
42             "location ":"-6.90916,107.61052"
43         },
44         {
45             "placename ":"Rice Bowl BIP",
46             "location ":"-6.90873,107.61088"
47         },
48         {
49             "placename ":"Gee Eight - Bip",
50             "location ":"-6.90817,107.61080"
51         },
52         {
53             "placename ":"Jonas Photo - BIP",
54             "location ":"-6.91066,107.61016"
55         },
56         {
57             "placename ":"Bip Foodcourt",
58             "location ":"-6.91081,107.61015"
59         },
60         {
61             "placename ":"Mister Baso BIP",
62             "location ":"-6.90348,107.61709"
63         },
64         {
65             "placename ":"JH Moriska - Bip",
66             "location ":"-6.90868,107.61070"
67         }
68     ],
69     "attribution " : null
70 }
```

1 Hasil dari kembalian berupa kumpulan *placename* dan *location*. Hasil tersebut akan  
 2 aplikasi tampung namun yang akan ditampilkan ke pengguna hanya *placename*. Menam-  
 3 pilkan *location* tidak efektif menurut penulis karena akan membingungkan pengguna. Dari  
 4 percobaan yang penulis lakukan, nilai dari *attributions* selalu bernilai "null". Karena hal  
 5 tersebut maka nilai *attributions* akan penulis abaikan.

6 Pemanfaatan layanan penentuan rute untuk mendapatkan langkah yang harus ditempuh  
 7 pengguna untuk mencapai lokasi tujuan dari lokasi asal. Pemanfaatan layanan ini yaitu  
 8 dengan parameter *GET* melalui protokol HTTP. Berikut parameter yang harus dikirim:

- 9     ● *version*: 2

10    Karena acuan penulis adalah Kiri API versi maka di parameter *version* penulis akan  
 11    menggunakan 2.

- 12     ● *mode*: "findroute"

13    Mode "findroute" digunakan untuk mendapatkan langkah yang harus ditempuh me-  
 14    nuju lokasi tujuan.

- 15     ● *locale*: "en" untuk bahasa Inggris dan "id" untuk bahasa Indonesia.

16    Karena aplikasi ini memungkinkan dipakai orang banyak maka penulis putuskan untuk  
 17    menggunakan bahasa Inggris.

- 18     ● *start*: koordinat lokasi awal dalam berupa latitude dan longitude.

19    Masukan untuk lokasi awal harus dalam bentuk koordinat. Jika masukan dari peng-  
 20    guna adalah alamat atau tempat maka perlu dicari kordinatnya dahulu.

- 21     ● *finish*: koordinat lokasi tujuan dalam berupa latitude dan longitude.

22    Masukan untuk lokasi tujuan harus dalam bentuk koordinat. Jika masukan dari peng-  
 23    guna adalah alamat atau tempat maka perlu dicari kordinatnya dahulu.

- 24     ● *presentation*: "mobile" untuk perangkat bergerak dan "desktop" untuk komputer.

25    Karena aplikasi ini dirancang untuk Windows Phone 8, presentasi yang penulis pilih  
 26    adalah "mobile".

- 27     ● *apikey*: 16 digit heksadesimal.

28    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](http://kiri.travel/handle.php?version=2&mode=findroute&locale=en/id&start=lat,lng&finish=lat,lng&presentation=mobile/desktop&apikey=97A7A1157A05ED6)

31    Penulis mencoba menuju jalan merdeka dari jalan ciumbuleuit. Layanan dikirimkan ke

32    URL [kiri.travel/handle.php](http://kiri.travel/handle.php). Berikut format layanan yang penulis kirim [http://kiri.](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)

33    |  |  |
| --- | --- |
| 33 | travel/handle.php?version=2&mode=findroute&locale=en&start=-6.8747337,107.6048829&finish= |

34    |  |  |
| --- | --- |
| 34 | -6.9114646,107.6104887&presentation=mobile&apikey=97A7A1157A05ED6F. |

35

36 Berikut hasil kembalian dari Kiri API:

Listing 3.2: Kode kembalian pencarian rute

```
37 {
38   "status ":"ok",
39   "routingresults ":[
40     {
41       "steps ":[
42         [
43           "walk",
44           "walk",
45           ["-6.8747337,107.6048829","-6.87445,107.60465"],
46           "Walk about 41 meter from your starting point \%fromicon to Jalan Ciumbuleuit
47           \%toicon.",
48           null
49         ],
50       [
51         "angkot",
52         "ciumbuleuitsthallurus",
53         ["-6.87445,107.60465","-6.87541,107.60443","-6.87637,107.60421","-6.87734,107.60400",
54         "-6.87830,107.60378",
55         "-6.87926,107.60356","-6.87926,107.60356","-6.87963,107.60352",
56       ]
57     ]
58   }
59 }
```

```

1      " -6.87978,107.60352 "," -6.88093,107.60392 "," -6.88209,107.60433 "," -6.88209,107.60433",
2      " -6.88328,107.60490 "," -6.88328,107.60490 "," -6.88347,107.60481 "," -6.88452,107.60459",
3      " -6.88556,107.60436 "," -6.88660,107.60413 "," -6.88764,107.60390 "," -6.88764,107.60391",
4      " -6.88782,107.60392 "," -6.88887,107.60404 "," -6.88991,107.60416 "," -6.88991,107.60416",
5      " -6.89161,107.60428 "," -6.89161,107.60428 "," -6.89166,107.60421 "," -6.89275,107.60424",
6      " -6.89275,107.60424 "," -6.89405,107.60408 "," -6.89405,107.60408 "," -6.89496,107.60400"],
7      "Take angkot Ciumbuleuit – St. Hall (lurus) at Jalan Ciumbuleuit \%fromicon,
8      and alight at Jalan Cihampelas
9      \%toicon about 2.3 kilometer later.", null
10     ],
11     [
12       "angkot",
13       "kalapaledeng",
14       [" -6.89501,107.60403 "," -6.89562,107.60398 "," -6.89623,107.60395 "," -6.89732,107.60401",
15       " -6.89732,107.60401 "," -6.89882,107.60414 "," -6.89882,107.60414 "," -6.89969,107.60418",
16       " -6.90071,107.60426 "," -6.90173,107.60433 "," -6.90173,107.60433 "," -6.90297,107.60437",
17       " -6.90420,107.60440 "," -6.90420,107.60440 "," -6.90426,107.60456 "," -6.90422,107.60481",
18       " -6.90399,107.60546 "," -6.90406,107.60617 "," -6.90454,107.60697 "," -6.90454,107.60697",
19       " -6.90512,107.60745 "," -6.90618,107.60778 "," -6.90618,107.60778 "," -6.90643,107.60787",
20       " -6.90651,107.60807 "," -6.90675,107.60914 "," -6.90675,107.60914 "," -6.90694,107.60939",
21       " -6.90723,107.60939 "," -6.90891,107.60943 "," -6.90891,107.60943 "," -6.90909,107.60934",
22       " -6.90914,107.60857 "," -6.90933,107.60846 "," -6.91021,107.60887 "," -6.91021,107.60887",
23       " -6.91030,107.60897 "," -6.91028,107.60927 "," -6.90986,107.61040 "," -6.90986,107.61040"],
24       "Take angkot Kalapa – Ledeng at Jalan Cihampelas \%fromicon, and alight at
25       Jalan Aceh
26       \%toicon about 2.3 kilometer later.", null
27     ],
28     [
29       "walk",
30       "walk",
31       [" -6.90986,107.61040 "," -6.9114646,107.6104887"],
32       "Walk about 178 meter from Jalan Aceh \%fromicon to your destination \%toicon
33       .",
34       null
35     ],
36     [
37       "traveltime": "30 minutes"
38     }
39   ]
40 }

```

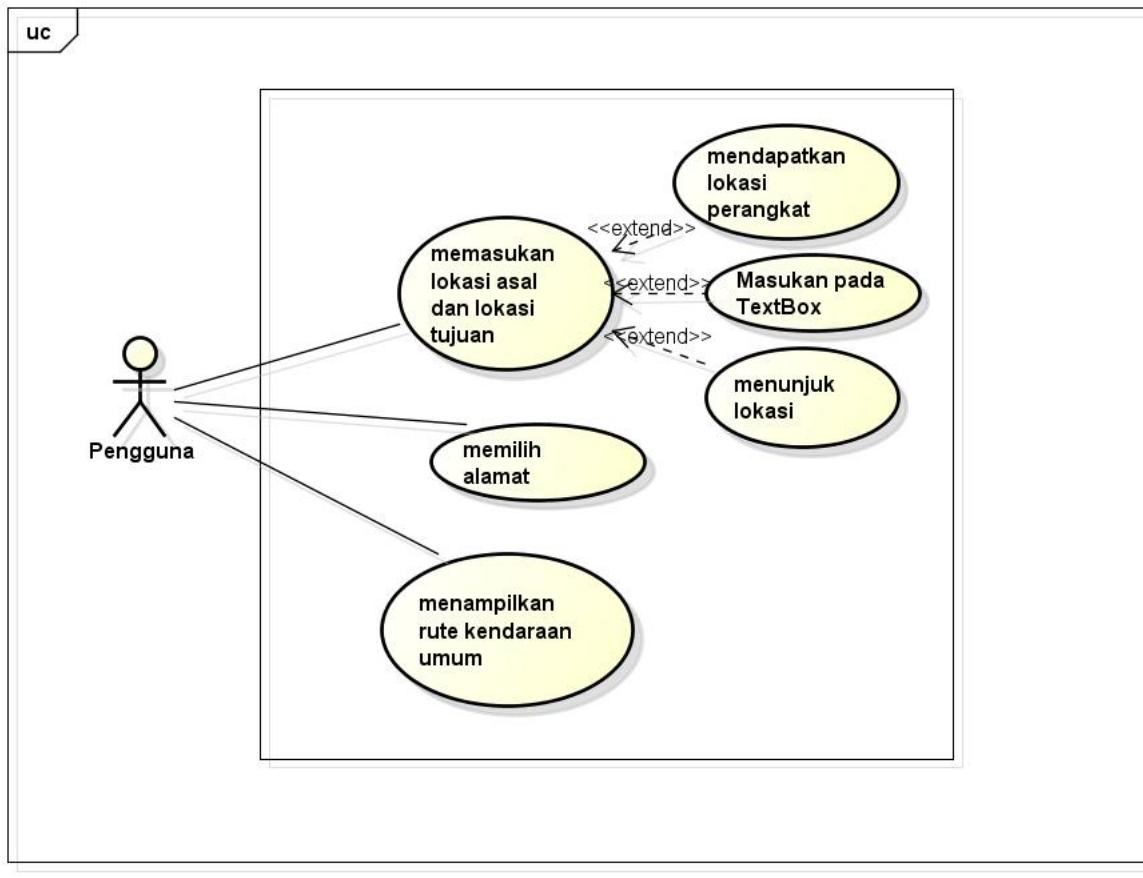
Setiap langkah akan aplikasi tampung dalam elemen *array*. Untuk keterangan dan jenis angkutan umum akan aplikasi tampilkan dalam bentuk *pushpin* pada peta atau daftar. Sedangkan untuk titik-titik kordinat akan digambarkan pada peta. Dari analisa penulis setiap langkah menunjukan perpindahan angkutan 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.

- 1 Diagram *use case* saat pengguna mencari rute kendaraan umum dapat dilihat pada gambar (Gambar: 3.8):



powered by Astah

Gambar 3.8: Diagram *use case*

2

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

Nama	Mendapatkan lokasi perangkat
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 mendapatkan lokasi untuk masukan lokasi asal dan lokasi tujuan

Nama	Masukan pada <i>TextBox</i>
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 pada *TextBox*

Nama	Menunjuk 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 alamat
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

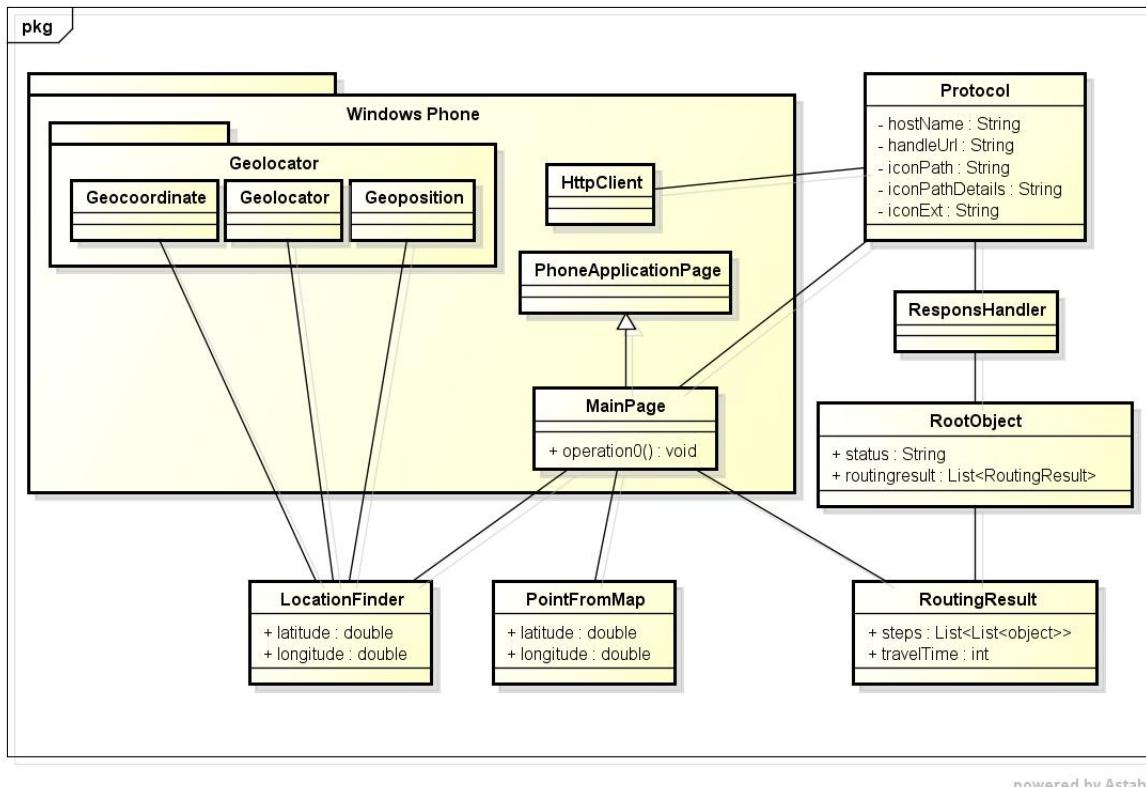
Tabel 3.4: Skenario memilih alamat

Nama	Menampilkan rute kendaraan umum
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 Kelas Diagram

Pembuatan kelas diagram didasarkan pada skenario pada sub bab 3.2.7. Kelas diagram dapat dilihat pada gambar 4.4.



Gambar 3.9: Diagram Kelas

Berikut deskripsi kelas pada gambar 4.4.

- 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.

1        Hasil kembalian akan dipisahkan di kelas ini untuk selanjutnya ditampung di kelas  
2        *RoutingResult*.

3        • Kelas *RoutingResult*  
4        Merupakan kelas untuk menampung setiap langkah dari rute sesuai masukan pengguna.  
5        Pada kelas ini juga rute akan digambarkan pada peta.

6        • Kelas *PointFromMap*  
7        Merupakan kelas yang dapat mengetahui lokasi yang ditunjuk pengguna pada peta.  
8        Kelas ini akan menyimpan lokasi yang ditunjuk pengguna dalam bentuk *latitude* dan  
9        *longitude*.

10        • Kelas *LocationFinder*  
11        Merupakan kelas yang digunakan untuk mencari lokasi. kelas ini akan memanfaatkan  
12        kelas *Geocoordinate* untuk mendapatkan lokasi. Setelah lokasi didapatkan dalam  
13        bentuk kelas *Geoposition* maka akan diubah ke *latitude* dan *longitude*.

1

## BAB 4

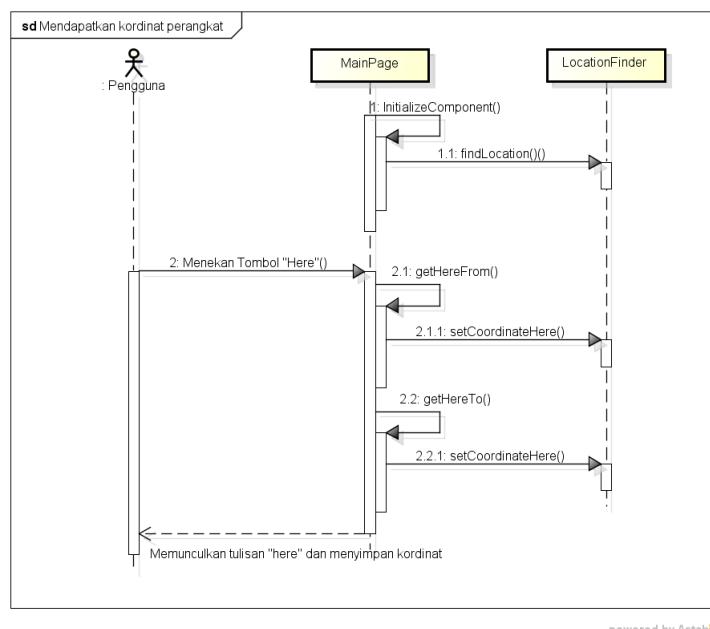
2

### PERANCANGAN

- 3 Pada bab 4 akan dibahas mengenai perancangan seperti diagram *sequence*, diagram kelas  
4 secara rinci, deskripsi atribut dan *method* dari setiap kelas, dan perancangan antarmuka.

5 **4.1 Diagram Sequence**

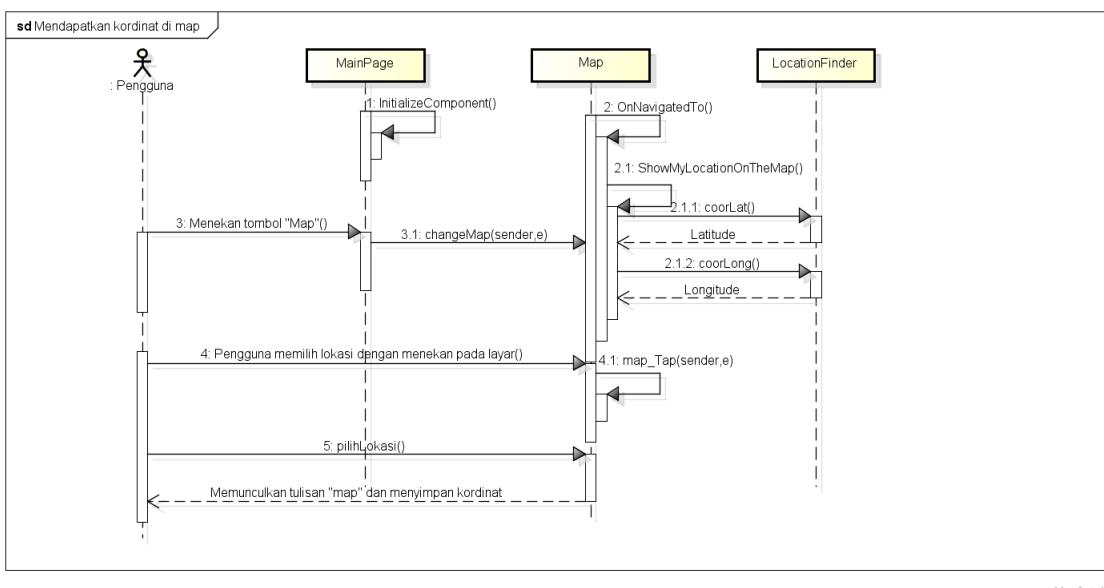
- 6 Diagram *sequence* merupakan diagram yang menggambarkan interaksi antar objek dalam  
7 suatu skenario. Gambar diagram *sequence* dapat dilihat pada gambar reffig:sequence lokasi  
8 perangkat sampai reffig:sequence rule.



Gambar 4.1: Diagram *sequence* Mendapatkan Kordinat perangkat

9 Diagram 4.1 merupakan diagram *sequence* untuk memilih lokasi dengan lokasi per-  
10 angkat berada. Diagram menunjukkan bahwa setelah aplikasi dibuka maka aplikasi akan  
11 mencari dahulu lokasi perangkat dengan memanfaatkan kelas LocationFinder. Lalu setelah  
12 aplikasi terbuka jika pengguna ingin memilih lokasi tersebut sebagai lokasi asal maka peng-  
13 guna harus menekan tombol "here". Setelah tombol "here" ditekan maka kelas MainPage  
14 akan mengambil nilai *Latitude* dan nilai *Longitude* dari kelas LocatonFinder. Setelah lokasi  
15 didapatkan maka akan muncul tulisan "here" pada masukan di kelas MainPage.

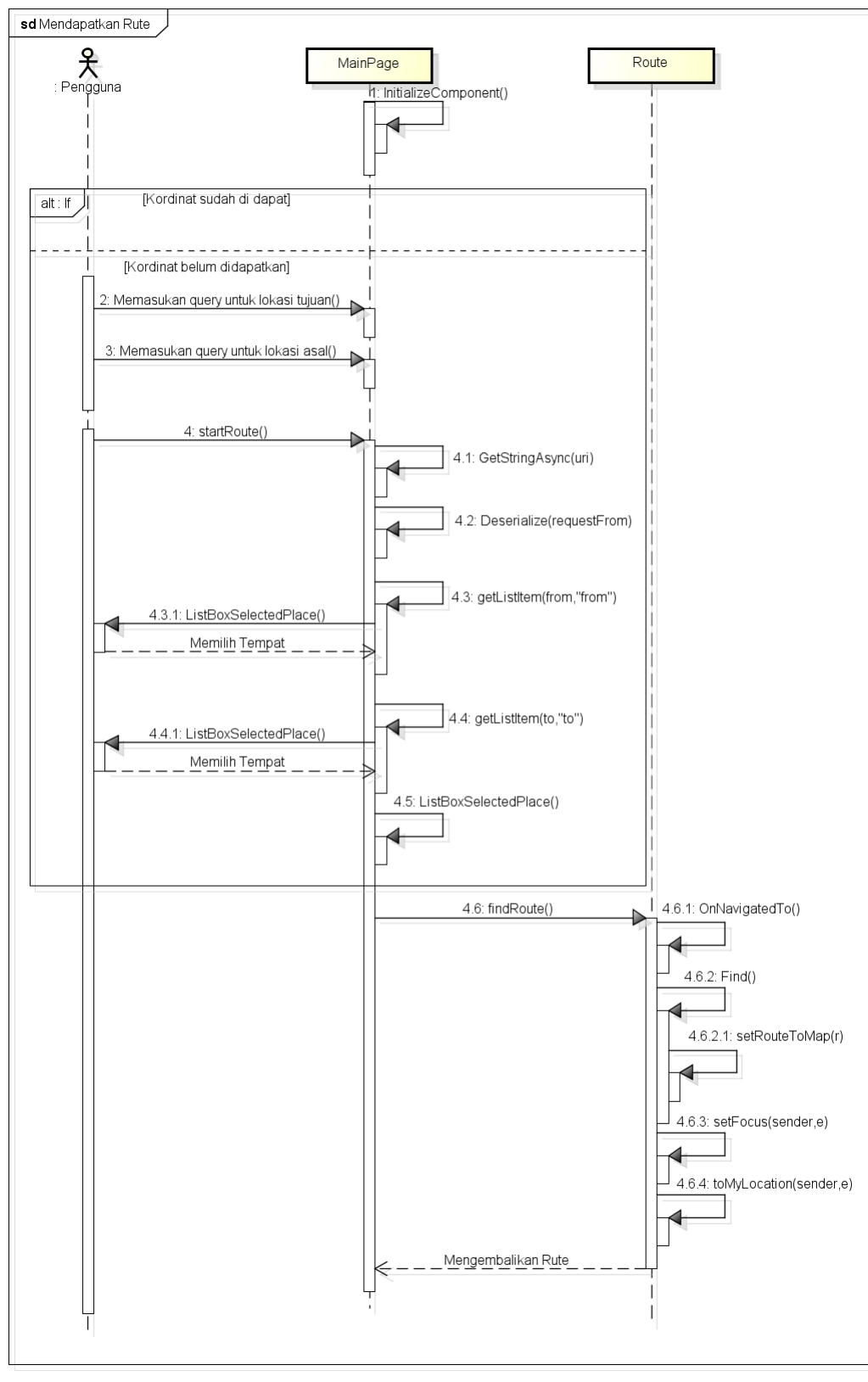
16 Diagram 4.2 merupakan diagram *sequence* untuk memilih lokasi. Diagram menun-  
17 jukkan bahwa setelah aplikasi dibuka maka pengguna dapat menekan tombol "map". Setelah  
18 tombol "map" ditekan maka halaman akan dialihkan ke kelas Map. Saat kelas Map terbuka  
19 maka lokasi yang ditunjukkan adalah lokasi dimana perangkat berada. Untuk mengetahui  
20 lokasi kelas Map mengambil kordinat *latitude* dan *longitude* dari kelas LocationFinder. Di



Gambar 4.2: Diagram *sequence* Mendapatkan Kordinat pada Peta

- 1 kelas "Map" pengguna dapat memilih lokasi dengan memilih lokasi pada peta dan memang-
- 2 gil *method map\_tap*, lalu setelah pengguna memilih tempat pengguna akan menekan tombol
- 3 Pilih Lokasi yang akan memanggil *method pilihLokasi*. Lokasi yang dipilih pengguna akan
- 4 disimpan di kelas MainPage dan pada masukan akan tertulis "map".

Diagram 4.3 merupakan diagram *sequence* untuk mencari rute. Diagram menunjukkan bahwa setelah aplikasi dibuka maka aplikasi akan melakukan inisialisasi. Untuk mencari rute dari lokasi asal ke lokasi tujuan dibutuhkan kordinat *latitude* lokasi asal, *longitude* lokasi asal, *latitude* lokasi tujuan, dan *longitude* lokasi tujuan. Jika pengguna mendapatkan lokasi dari peta atau sesuai lokasi maka yang didapatkan sudah pasti kordinat, namun jika pengguna memasukan kata kunci perlu didapat kordinat *latitude* dan *longitude* dari kata kunci tersebut. Jika masukan yang didapat berupa kata kunci maka akan dilakukan pemeriksaan apakah kordinat untuk kata kunci tersebut tersedia. Pemeriksaan dilakukan dengan melakukan pemanggilan Kiri API. Tahap pemanggilan meliputi pemanggilan *method GetStringAsync* lalu mengjadikan objek kembalinya dengan *method Deserialize*. Jika sudah didapat dan hasilnya lebih dari satu maka akan dipanggil *method getListItem* yang akan menampilkan daftar pilihan ke pengguna untuk dipilih. Pengguna dapat memilih tempat sesuai tempat asal maupun tujuan yang diinginkan. Setelah lokasi asal dan lokasi tujuan dapat maka kelas MainPage akan mengarahkan ke kelas Route untuk menampilkan hasilnya. Kelas Route akan memanggil *method OnNavigatedTo* yang bertujuan untuk mendapatkan lokasi asal dan lokasi tujuan. Setelah itu akan memanggil *method Find* lalu mengembalikan rute yang ditemukan kepada pengguna.



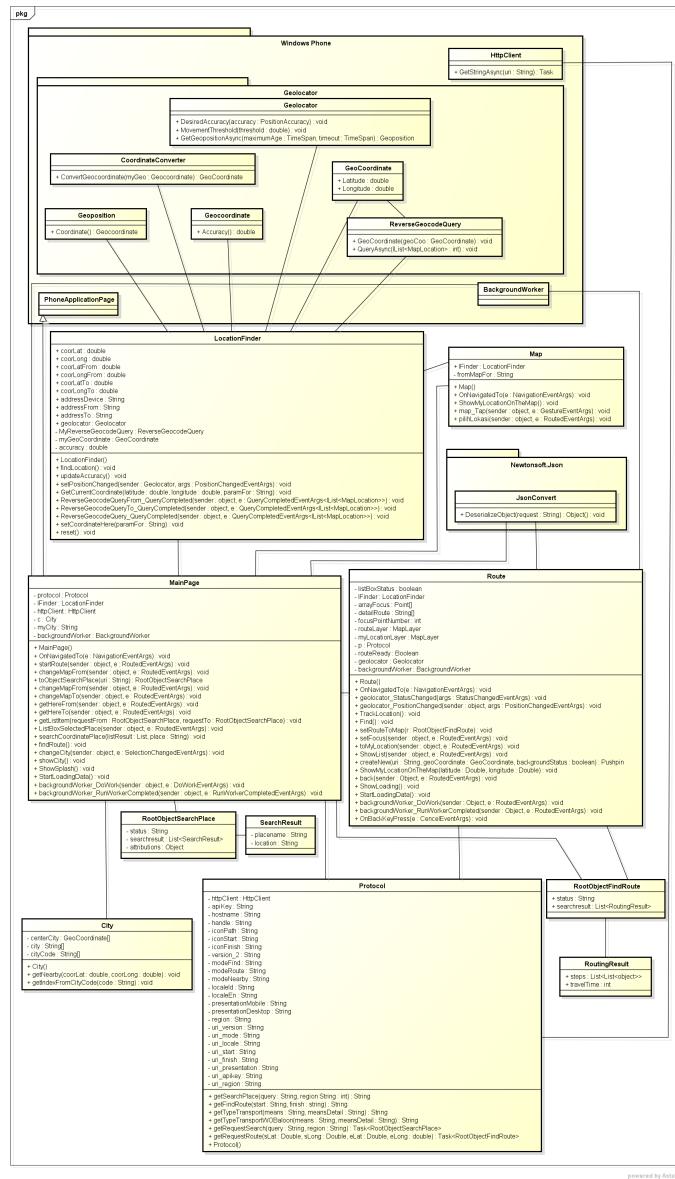
powered by Astah

Gambar 4.3: Diagram *sequence* Mendapatkan Rute

## 4.2 Perancangan Kelas

- Pada sub bab ini akan dibahas mengenai deskripsi kelas secara rinci pada aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone. Untuk lebih jelas mengenai kelas yang ada

- 1 pada aplikasi ini, penulis menyajikan gambar diagram kelas yang dapat dilihat pada gambar  
 2 **4.4.**



Gambar 4.4: Diagram Kelas

### 3 4.2.1 Kelas *PhoneApplicationPage*

- 4 *PhoneApplicationPage* merupakan kelas bawaan Windows Phone yang menangani in-  
 5 terksi pengguna dengan aplikasi dan siklus hidup aplikasi.

### 6 4.2.2 Kelas *MainPage*

- 7 *MainPage* merupakan kelas turunan dari kelas *PhoneApplicationPage* yang menangani in-  
 8 teraksi langsung antara halaman aplikasi dengan pengguna. Pada kelas ini akan ditaruh  
 9 kontrol yang diperlukan. Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 10 1. protocol bertipe Protocol untuk mendapatkan URL yang digunakan dalam permintaan  
 11 ke Kiri API.  
 12 2. IFinder bertipe LocationFinder digunakan untuk menampung semua informasi meng-  
 13 enai lokasi.

- 1     3. httpClient bertipe HttpClient merupakan objek yang akan mengurus permintaan dan  
2     kembalian dari Kiri API.
  - 3     4. city bertipe kumpulan String untuk menampung kota yang didukung oleh layanan  
4     Kiri.
  - 5     5. myCity bertipe String untuk menampung kode kota sesuai Kode Penerbangan IATA.
  - 6     6. backgroundWorker bertipe BackgroundWorker untuk mengurus pencarian lokasi di  
7     belakang layar.
- 8 Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:
- 9     1. Konstruktor MainPage digunakan untuk memuat komponen yang ada di halaman  
10    MainPage dan mendapatkan kota terdekat dari lokasi perangkat.
  - 11    2. *method* OnNavigatedTo digunakan untuk mendapatkan lokasi dari peta dan mendapat-  
12    kan objek LocationFinder. *method* ini memiliki parameter NavigationEventArgs.
  - 13    3. *method* OnNavigatedFrom digunakan untuk mengirimkan *state* objek LocationFinder  
14    saat berpindah kelas. *method* ini memiliki parameter NavigationEventArgs.
  - 15    4. *method* startRoute digunakan untuk mendapatkan masukan pengguna. Jika masukan  
16    didapat dari peta atau lokasi perangkat berarti sudah dalam kordinat, namun jika  
17    masukan didapat dari *query* maka akan dicari lokasi yang terkait sesuai *query* tersebut.  
18    Lokasi terkait yang didapatkan akan dikembalikan ke pengguna untuk dipilih. Setelah  
19    pengguna lokasi dalam bentuk kordinat didapatkan maka kelas akan diarahkan ke kelas  
20    Route. *method* ini memiliki parameter objek tombol dan event.
  - 21    5. *method* changeMapFrom digunakan untuk berpindah ke halaman mapFrom. *method*  
22    ini memiliki parameter objek tombol dan event.
  - 23    6. *method* changeMapTo digunakan untuk berpindah ke halaman mapTo. *method* ini  
24    memiliki parameter objek tombol dan event.
  - 25    7. *method* getHereFrom digunakan untuk mendapatkan kordinat perangkat lalu menyimpan  
26    nilainya di kordinat asal di kelas LocationFinder dan menulisakan "Here" pada  
27    TextBox lokasi asal. *method* ini memiliki parameter objek tombol dan event.
  - 28    8. *method* getHereTo digunakan untuk mendapatkan kordinat perangkat lalu menyimpan  
29    nilainya di kordinat tujuan di kelas LocationFinder dan menulisakan "Here" pada  
30    TextBox lokasi tujuan. *method* ini memiliki parameter objek tombol dan event.
  - 31    9. *method* getListItem digunakan untuk membuat listBox lalu menampilkan ke pengguna.  
32    *method* ini memiliki parameter RootObjectSearchPlace dan string yang menunjukkan  
33    list yang ditampilkan untuk lokasi asal dan lokasi tujuan.
  - 34    10. *method* ListBoxSelectedPlace digunakan untuk mendapatkan tempat asal yang dipilih  
35      pengguna. *method* ini memiliki parameter objek dan *event* SelectionChangedEventArgs.
  - 36    11. *method* searchCoordinatePlace digunakan untuk mencari kordinat dari tempat pilihan  
37      pengguna. *method* ini memiliki parameter ListBox dan tempat yang dipilih dalam  
38      bentuk string.
  - 39    12. *method* findRoute digunakan untuk berpindah ke kelas Route jika lokasi asal dan lokasi  
40      tujuan sudah ditentukan.
  - 41    13. *method* changeCity digunakan untuk mengubah kota tujuan dari pencarian. *method*  
42      ini memiliki parameter objek dan *event* SelectionChangedEventArgs.

- 1    14. *method* showCity digunakan untuk mencari kota yang paling dekat dengan lokasi per-
- 2    angkat.
- 3    15. *method* ShowSplash digunakan untuk menampilkan tampilan awal untuk proses inisi-
- 4    alisasi aplikasi.
- 5    16. *method* StartLoadingData digunakan untuk memanggil BackgroundWorker. Backgro-
- 6    undWorker digunakan untuk melakukan aksi di belakang layar.
- 7    17. *method* backgroundWorker\_DoWork digunakan untuk melakukan pemanggilan aksi
- 8    di belakang layar. *method* ini memiliki parameter objek dan DoWorkEventArgs.
- 9    18. *method* backgroundWorker\_RunWorkerCompleted digunakan untuk melakukan pe-
- 10    manggilan saat BackgroundWorker selesai melakukan tugasnya. *method* ini memiliki
- 11    parameter objek dan RunWorkerCompletedEventArgs.

#### 12    4.2.3 Kelas *City*

13    *City* merupakan kelas yang menyimpan kota-kota yang mendukung pencarian rute ken-

14    daraam umum dengan bantuan Kiri. Berikut adalah penjelasan atribut-atribut yang dimiliki

15    kelas ini:

- 16    1. centerCity bertipe *array of GeoCoordinate* untuk menyimpan kordinat pusat dari kota.
- 17    2. city bertipe *array of String* untuk menyimpan nama kota.
- 18    3. cityCode bertipe *array of String* untuk menyimpan kode kota dalam huruf kecil sesuai
- 19    aturan IATA Airport Code.

20    Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:

- 21    1. Konstruktor City digunakan untuk untukinisialisasi nilai atribut.
- 22    2. *method* getNearby digunakan untuk mencari kota terdekat dengan lokasi perangkat.
- 23    *method* ini mengembalikan interger yang merupakan index kota pada atribut city.
- 24    *method* ini memiliki 2 buah parameter yaitu *latitude* dan *longitude* yang bertipe double.
- 25    3. *method* getIndexFromCityCode digunakan untuk mencari index pada *array* sesuai kode
- 26    kota. *method* ini memiliki parameter bertipe String yang merupakan kode kota.

#### 27    4.2.4 Kelas *BackgroundWorker*

28    *BackgroundWorker* merupakan kelas yang dipakai untuk mengeksekusi operasi pada

29    *thread* terpisah. Berikut adalah penjelasan *event* yang dimiliki kelas ini dan dipakai untuk

30    perancangan aplikasi:

- 31    1. *Event* DoWork
- 32    2. *Event* RunWorkerCompleted

33    Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:

- 34    1. *method* RunWorkerAsync digunakan untuk memulai operasi di belakang layar.

#### 35    4.2.5 Kelas *Geocoordinate*

36    *Geocoordinate* merupakan kelas bawaan dari Windows Phone yang akan dimanfaatkan

37    untuk membaca *latitude* dan *longitude*.

1    **4.2.6 Kelas *Geolocator***

2       *Geolocator* merupakan kelas bawaan Windows Phone untuk mengkases lokasi. Dengan  
3       bantuan kelas ini maka dapat mengetahui status lokasi dari perangkat dan menemukan lokasi  
4       secara akurat.

5    **4.2.7 Kelas *Geoposition***

6       *Geoposition* merupakan kelas yang menampung lokasi sesuak kembalian *Geolocator*.

7    **4.2.8 Kelas *LocationFinder***

8       *LocationFinder* merupakan kelas yang akan menampung lokasi dan pencarian lokasi.  
9       Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 10      1. coorLat bertipe Double untuk menampung kordinat latitude pengguna.
- 11      2. coorLong bertipe Double untuk menampung kordinat longitude pengguna.
- 12      3. coorLatFrom bertipe Double untuk menampung kordinat latitude lokasi asal yang  
13        diinginkan pengguna.
- 14      4. coorLongFrom bertipe Double untuk menampung kordinat longitude lokasi asal yang  
15        diinginkan pengguna.
- 16      5. coorLatTo bertipe Double untuk menampung kordinat latitude lokasi tujuan yang  
17        diinginkan pengguna.
- 18      6. coorLongTo bertipe Double untuk menampung kordinat longitude lokasi tujuan yang  
19        diinginkan pengguna.
- 20      7. addressDevice bertipe String untuk menyimpan alamat perangkat berada.
- 21      8. addressDeviceFrom bertipe String untuk menyimpan alamat berdasarkan lokasi lokasi  
22        asal yang diinginkan pengguna.
- 23      9. addressDeviceTo bertipe String untuk menyimpan alamat berdasarkan lokasi tujuan  
24        yang diinginkan pengguna.
- 25      10. geolocator bertipe Geolocator untuk menampung pengaturan mendapatkan lokasi.
- 26      11. myReverseGeocodeQuery bertipe ReverseGeocodeQuery untuk konversi dari alamat  
27        ke lokasi dan sebaliknya.
- 28      12. myCoordinate bertipe GeoCoordinate untuk menampung kordinat geografis.
- 29      13. accuracy bertipe double untuk menampung akurasi perangkat mendapatkan lokasi.

30       Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:

- 31      1. Konstruktor LocationFinder berfungsi mengatur atribut geolocator dan mencari lokasi  
32        perangkat.
- 33      2. *method* findLocation berfungsi inisialisasi GPS lalu mendapat kordinat dan menam-  
34        pungnya di atribut.
- 35      3. *method* updateAccuracy berfungsi untuk mengubah nilai akurasi dari perangkat.
- 36      4. *method* setPositionChanged berfungsi mengubah atribut coorLat, atribut coorLong,  
37        dan akurasi jika terdapat perubahan lokasi. *method* ini memiliki parameter Geolo-  
38        cator dan PositionChangedEventArgs yang akan menjalankan *method* jika terdapat  
39        perubahan yang diberitahukan melalui kelas Geolocator.

- 1       5. *method* GetCurrentCoordinate berfungsi mengubah posisi saat ini, posisi lokasi asal, dan lokasi tujuan. *method* ini memiliki tiga buah parameter *latitude* bertipe Double, *longitude* bertipe Double, dan *paramFor* bertipe String. Parameter *latitude* dan *longitude* merupakan lokasi sedangkan parameter *paramFor* digunakan sebagai tujuan perubahan lokasi.
- 6       6. *method* ReverseGeocodeQueryFrom\_QueryCompleted berfungsi untuk mencari alamat lokasi asal. *method* ini memiliki parameter objek dan QueryCompletedEventArgs<IList<MapLocation>>.
- 7       7. *method* ReverseGeocodeQueryTo\_QueryCompleted berfungsi untuk mencari alamat lokasi tujuan. *method* ini memiliki parameter objek dan QueryCompletedEventArgs<IList<MapLocation>>.
- 8       8. *method* ReverseGeocodeQuery\_QueryCompleted berfungsi untuk mencari alamat lokasi perangkat. *method* ini memiliki parameter objek dan QueryCompletedEventArgs<IList<MapLocation>>.
- 9       9. *method* setCoordinateHere berfungsi untuk menyimpan kordinat dan alamat perangkat ke kordinat dan alamat lokasi asal dan lokasi tujuan. *method* ini memiliki parameter *paramFor* bertipe String yang akan digunakan sebagai masukan disimpannya lokasi perangkat.
- 10      10. *method* reset berfungsi untuk memasang kembali lokasi asal dan lokasi tujuan.

#### 20     **4.2.9 Kelas Map**

21     *Map* merupakan kelas yang akan mendapatkan titik yang ditunjuk pengguna pada peta  
22     lalu menerjemahkannya dalam bentuk titik kordinat. Berikut adalah penjelasan atribut-  
23     atribut yang dimiliki kelas ini:

- 24     1. *IFinder* bertipe LocationFinder digunakan untuk menampung semua informasi meng-  
25     enai lokasi.
- 26     2. *fromMapFor* bertipe string digunakan sebagai indikator lokasi asal atau lokasi tujuan  
27     yang didapatkan dari map.

28     Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:

- 29     1. Konstruktor Map untuk inisialisasi dan penambahan *event* mengetuk pada peta.
- 30     2. *method* OnNavigatedTo berfungsi untuk mendapatkan masukan lokasi asal atau lokasi  
31     tujuan yang akan ditentukan dari map untuk kemudian ditampung di Objek Location-  
32     Finder. *method* ini memiliki sebuah parameter NavigationEventArgs.
- 33     3. *method* OnNavigatedFrom digunakan untuk mengirimkan *state* objek LocationFinder  
34     saat berpindah kelas. *method* ini memiliki parameter NavigationEventArgs.
- 35     4. *method* ShowMyLocationOnTheMap digunakan untuk memberitahu dan menandai  
36     lokasi perangkat.
- 37     5. *method* map\_Tap berfungsi untuk menandai lokasi yang ditunjuk pengguna lalu me-  
38     nerjemahkan lokasi yang ditunjuk pengguna pada peta dan mengirimnya ke kelas Lo-  
39     cationFinder.
- 40     6. *method* pilihLokasi berfungsi berpindah ke kelas MainPage dan memberitahu kelas  
41     MainPage bahwa lokasi sudah dipilih. *method* ini memiliki parameter objek tombol  
42     dan event.

1   **4.2.10 Kelas *HttpClient***

2       *HttpClient* merupakan kelas bawaan Windows Phone untuk mengatur pengiriman dan  
3       kembalian menggunakan protokol HTTP. Berikut adalah penjelasan *method* kelas *HttpClient*  
4       yang dipakai untuk perancangan aplikasi ini:

- 5       1. *method* GetStringAsync membutuhkan parameter alamat bertipe string dan mengem-  
6       balikan kembalian dari Kiri dalam bentuk Task<string>.

7   **4.2.11 Kelas *JsonConvert***

8       *JsonConvert* merupakan kelas yang menyediakan *method* untuk mengonversi berbagai  
9       jenis komponen *common language runtime* dan *JSON*. Kelas ini merupakan bagian *namespa-*  
10      ce *Newtonsoft*. Berikut adalah penjelasan *method* yang dipakai untuk perancangan aplikasi:

- 11      1. *method* DeserializeObject berfungsi untuk konversi dari bentuk string menjadi objek.  
12      *method* ini memiliki satu parameter bertipe string lalu mengembalikan string tersebut  
13      dalam bentuk objek.

14   **4.2.12 Kelas *Protocol***

15       *Protocol* merupakan kelas untuk menampung semua alamat dalam pengiriman menggu-  
16       nakan protokol HTTP. Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 17      1. uri\_version bertipe string digunakan untuk menyimpan nama dari parameter uri.  
18      2. uri\_mode bertipe string digunakan untuk menyimpan nama dari parameter mode.  
19      3. uri\_locale bertipe string digunakan untuk menyimpan nama dari parameter locale.  
20      4. uri\_start bertipe string digunakan untuk menyimpan nama dari parameter start.  
21      5. uri\_finish bertipe string digunakan untuk menyimpan nama dari parameter finish.  
22      6. uri\_presentation bertipe string digunakan untuk menyimpan nama dari parameter  
23       presentation.  
24      7. uri\_apikey bertipe string digunakan untuk menyimpan nama dari parameter apikey.  
25      8. uri\_region bertipe string digunakan untuk menyimpan nama dari parameter region.  
26      9. uri\_query bertipe string digunakan untuk menyimpan nama dari parameter query.  
27     10. apiKey bertipe string digunakan untuk menyimpan nilai kunci API untuk mengirim  
28       permintaan ke Kiri.  
29     11. hostname bertipe string digunakan untuk digunakan untuk menyimpan alamat host  
30       dari Kiri.  
31     12. handle bertipe string digunakan untuk menyimpan alamat host ditambah "handle.php".  
32     13. iconPath bertipe string digunakan untuk menyimpan lokasi gambar yang dibutuhkan.  
33     14. iconStart bertipe string digunakan untuk menyimpan lokasi gambar awal perjalanan  
34       dari lokasi awal.  
35     15. iconFinish bertipe string digunakan untuk menyimpan lokasi gambar akhir perjalanan  
36       ke lokasi tujuan.  
37     16. version\_2 bertipe string digunakan untuk menyimpan nilai versi dari API yang di-  
38       gunaikan (saat pembuatan penelitian ini versi Kiri API yang digunakan adalah versi  
39       2).

- 1 17. modeFind bertipe string yang digunakan untuk menyimpan nilai "searchplace" yang  
2 merupakan mode mencari lokasi terkait pada Kiri API.
- 3 18. modeRoute bertipe string yang digunakan untuk menyimpan nilai "findroute" yang  
4 merupakan mode mencari rute pada Kiri API.
- 5 19. modeNearby bertipe string yang digunakan untuk menyimpan nilai "nearbytransport"  
6 " yang merupakan mode mencari lokasi terdekat pada Kiri API.
- 7 20. localeId bertipe string yang digunakan untuk menyimpan nilai bahasa jika kembalian  
8 yang diinginkan ingin berbahasa Indonesia.
- 9 21. localeEn bertipe string yang digunakan untuk menyimpan nilai bahasa jika kembalian  
10 yang diinginkan ingin berbahasa Inggris.
- 11 22. presentationMobile bertipe string yang digunakan untuk menyimpan nilai penyajian  
12 untuk perangkat *mobile*.
- 13 23. presentationDesktop bertipe string yang digunakan untuk menyimpan nilai penyajian  
14 untuk perangkat *desktop*.

15 Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:

- 16 1. *method* getTypeTransport merupakan *method* yang akan mengembalikan alamat dari  
17 gambar transportasi dengan bingkai. *method* ini memiliki 2 parameter yaitu means  
18 sebagai tipe transportasi dan meansDetail sebagai nama kendaraan.
- 19 2. *method* getTypeTransportWOBaloon merupakan *method* yang akan mengembalikan  
20 alamat dari gambar transportasi tanpa bingkai tambahan. *method* ini memiliki 2 par-  
21 meter yaitu means sebagai tipe transportasi dan meansDetail sebagai nama kendaraan.
- 22 3. getSearchPlace merupakan *method* yang akan mengembalikan URI pencarian lokasi  
23 sesuai paramater. Parameter yang dimaksud adalah kata kunci masukan pengguna.
- 24 4. *method* getFindRoute merupakan *method* yang akan mengembalikan URI pencarian  
25 rute sesuai parameter. Parameter yang dimaksud adalah kordinat lokasi asal dan  
26 kordinat lokasi tujuan yang bertipe string.
- 27 5. *method* getRequestSearch digunakan untuk mendapatkan lokasi terkait sesuai masuk-  
28 an pengguna. *method* ini akan mengembalikan Task<RootObjectSearchPlace> karena  
29 menggunakan operasi *asynchronous*. *method* ini memiliki parameter kata kunci ma-  
30 sukan pengguna dan kota yang masing-masing parameter bertipe String.
- 31 6. *method* getRequestRoute digunakan untuk mendapatkan rute sesuai lokasi asal dan  
32 lokasi tujuan. *method* ini akan mengembalikan Task<RootObjectFindRoute> karena  
33 menggunakan operasi *asynchronous*. *method* ini memiliki parameter *latitude* lokasi  
34 asal, *longitude* lokasi asal, *latitude* lokasi tujuan, dan *longitude* lokasi tujuan yang  
35 masing-masing bertipe double.

#### 36 4.2.13 Kelas *RootObjectSearchPlace*

37 *RootObjectSearchPlace* merupakan kelas untuk menampung objek hasil pencarian lokasi.  
38 Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 39 1. status bertipe *string* digunakan untuk menampung hasil kembalian status dari Kiri.
- 40 2. searchresult bertipe *list* dan menampung banyak objek *SearchResult*.
- 41 3. attributions bertipe objek untuk menampung *attributions*.

1   **4.2.14 Kelas *SearchResult***

2       *SearchResult* merupakan kelas untuk menampung nama tempat dan kordinat dari nama  
3       tempat tersebut. Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 4       1. placename bertipe *string* digunakan untuk menampung nama tempat.  
5       2. location bertipe *string* digunakan untuk menampung nama tempat.

6   **4.2.15 Kelas *RootObjectFindRoute***

7       *RootObjectFindRoute* merupakan kelas untuk menampung hasil pencarian rute. Berikut  
8       adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 9       1. status  
10      2. routingresults

11   **4.2.16 Kelas *RoutingResult***

12       *RoutingResult* merupakan kelas untuk menampung langkah menuju tempat tujuan dan  
13       waktu yang dibutuhkan. Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 14      1. steps  
15      2. traveltimes

16   **4.2.17 Kelas *Route***

17       *Route* merupakan kelas untuk pencarian rute dan menampilkannya kepada pengguna.  
18       Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 19      1. listBoxStatus bertipe boolean digunakan untuk status rute dalam bentuk daftar sedang  
20       tertutup atau terbuka.
- 21      2. lFinder bertipe LocationFinder digunakan untuk menampung semua informasi meng-  
22       enai lokasi.
- 23      3. arrayFocus bertipe *array of Point* digunakan untuk menampung titik fokus perubahan  
24       jenis transportasi yang digunakan pengguna.
- 25      4. detailRoute bertipe *array of String* digunakan untuk menampung keterangan yang  
26       dibutuhkan pengguna dari Kiri API.
- 27      5. focusPointNumber bertipe *integer* digunakan untuk menentukan *index of* dari atribut  
28       arrayFocus dan detailRoute.
- 29      6. routeLayer bertipe MapLayer digunakan untuk menampung *Polyline* dan *Pushpin*
- 30      7. myLocationLayer bertipe MapLayer digunakan untuk menampung titik dari lokasi  
31       perangakat berada.
- 32      8. p bertipe Protocol digunakan untuk menampung permintaan data dengan Kiri API.
- 33      9. geolocator bertipe Geolocator untuk menangani perubahan lokasi yang terjadi.
- 34      10. backgroundWorker bertipe BackgroundWorker digunakan untuk menangani proses di  
35       belakang layar.
- 36       Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:

1. Konstruktor Route digunakan untuk memuat komponen yang ada di halaman Route, inisialisasi atribut, dan pemanggilan backgroundWorker.
2. *method* OnNavigatedTo digunakan untuk mendapatkan objek LocationFinder dan memanggil *method* TrackLocation. *method* ini memiliki parameter NavigationEventArgs.
3. *method* OnNavigatedFrom digunakan untuk mengirimkan *state* objek LocationFinder saat berpindah kelas. *method* ini memiliki parameter NavigationEventArgs.
4. *method* TrackLocation digunakan untuk inisialisasi GeoLocator dan memulai pelacakan lokasi terus menerus.
5. *method* geolocator\_StatusChanged digunakan untuk mengetahui status GeoLocator.
6. *method* geolocator\_PositionChanged digunakan untuk mengetahui perubahan lokasi yang terjadi dan menyimpannya di kelas LocationFinder.
7. *method* Find digunakan untuk mencari rute yang dicari pengguna.
8. *method* setRouteToMap digunakan untuk menggambar rute dan lokasi pada *layer* peta. *method* ini memiliki parameter RootObjectFindRoute yang merupakan objek untuk pencarian rute.
9. *method* setFocus digunakan untuk mengarahkan pusat pandangan ke titik lokasi pergantian jenis transportasi. *method* ini memiliki parameter objek dan RoutedEventArgs.
10. *method* toMyLocation digunakan untuk mengarahkan pusat pandangan ke titik lokasi perangkat berada. *method* ini memiliki parameter objek dan RoutedEventArgs.
11. *method* ShowList digunakan untuk membuka dan menutup rute dalam bentuk daftar. *method* ini memiliki parameter objek dan RoutedEventArgs.
12. *method* createNew digunakan untuk membuat objek Pushpins. *method* ini memiliki parameter uri bertipe String, transport bertipe String yang menandakan jenis transportasi, dan geoCoordinate bertipe GeoCoordinate sebagai lokasi dari Pushpins.
13. *method* drawMyLocationOnTheMap digunakan untuk membuat penanda lokasi di lapisan myLocationLayer pada peta. *method* ini memiliki parameter latitude bertipe double dan longitude bertipe double.
14. *method* back digunakan untuk konfirmasi ke pengguna jika pengguna ingin meninggalkan aplikasi. *method* ini memiliki dua buah parameter sender bertipe objek dan e bertipe RoutedEventArgs.
15. *method* ShowLoading digunakan untuk memunculkan *popup* menunggu.
16. *method* StartLoadingData digunakan untuk pemanggilan BackgroundWorker.
17. *method* backgroundWorker\_DoWork digunakan untuk eksekusi *method* dengan BackgroundWorker. *method* ini memiliki dua buah parameter sender bertipe objek dan e bertipe DoWorkEventArgs.
18. *method* backgroundWorker\_RunWorkerCompleted digunakan untuk menutup *popup* menunggu jika semua *method* sudah selesai dijalankan. *method* ini memiliki dua buah parameter sender bertipe objek dan e bertipe RunWorkerCompletedEventArgs.
19. *method* OnBackKeyPress digunakan untuk konfirmasi ke pengguna jika pengguna ingin meninggalkan aplikasi dengan menekan tombol "back". *method* ini memiliki parameter e bertipe CancelEventArgs.

## <sup>1</sup> 4.3 Perancangan Antar Muka

<sup>2</sup> Pada sub bab ini akan dibahas mengenai antarmuka pada aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone. Antarmuka berfungsi sebagai jembatan yang menghubungkan antara aplikasi dengan pengguna. Berikut ini akan dijelaskan mengenai rancangan antarmuka aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone.

### <sup>6</sup> 4.3.1 Antarmuka Kelas *MainPage*



Gambar 4.5: Antarmuka *MainPage*

<sup>7</sup> Antarmuka Kelas Map pada gambar [4.5](#) merupakan tampilan awal saat aplikasi <sup>8</sup> dijalankan. Antarmuka Kelas Map memiliki dua buah masukan, lima buah tombol, dan <sup>9</sup> satu menu daftar. Berikut adalah detailnya.

<sup>10</sup> Dua buah masukan yaitu.

- <sup>11</sup> • Masukan lokasi asal  
Merupakan masukan lokasi asal mula pengguna ingin melakukan perjalanan.
- <sup>13</sup> • Masukan lokasi tujuan  
Merupakan masukan lokasi tujuan berhentinya perjalanan.

<sup>15</sup> Lima buah tombol yaitu.

- <sup>16</sup> • Tombol map untuk lokasi asal  
Jika tombol ditekan maka akan berpindah ke kelas map untuk memilih lokasi asal di peta. Jika di kelas Map pengguna memilih lokasi maka pada masukan lokasi asal terdapat tulisan "Maps".
- <sup>20</sup> • Tombol here untuk lokasi asal  
Jika tombol ditekan maka lokasi asal adalah lokasi perangkat saat tombol ditekan dan masukan lokasi asal menjadi "here".
- <sup>23</sup> • Tombol map untuk lokasi tujuan  
Jika tombol ditekan maka akan berpindah ke kelas map untuk memilih lokasi tujuan di peta. Jika di kelas Map pengguna memilih lokasi maka pada masukan lokasi tujuan terdapat tulisan "Maps".
- <sup>27</sup> • Tombol here untuk lokasi tujuan  
Jika tombol ditekan maka lokasi tujuan adalah lokasi perangkat saat tombol ditekan dan masukan lokasi tujuan menjadi "here".

- 1     ● Tombol find
- 2         Jika tombol ditekan maka akan menampilkan daftar tempat asal dan tempat tujuan
- 3         lalu mengarahkan ke Kelas Route.
  
- 4     Satu buah daftar yaitu.
  
- 5     ● Daftar kota yang tersedia
- 6         Merupakan daftar kota yang tersedia (kota yang rute angkutan umumnya dapat ditemukan dengan aplikasi ini). Disaat aplikasi dijalankan maka daftar akan menunjuk ke
- 7         kota terdekat tempat perangkat berada.

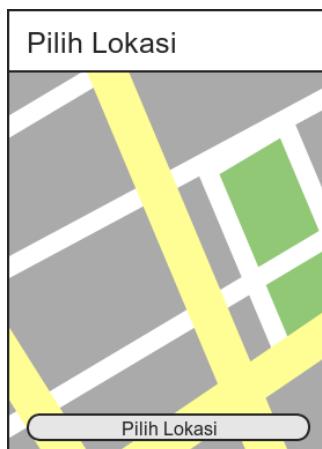
Pilih Tempat

Kota 1  
Kota 2  
Kota 3  
Kota 4  
Kota 5

Gambar 4.6: Antarmuka Daftar Tempat

9             Antarmuka Daftar Tempat pada gambar 4.6 merupakan daftar yang akan dimunculkan jika pengguna memasukan kata kunci pada pencarian tempat asal dan tempat tujuan.  
10  
11          Daftar akan muncul jika didapat kembalian hasil pencarian lebih dari satu.

#### 12     4.3.2    Antarmuka Kelas *Map*

Gambar 4.7: Antarmuka *Map*

13             Antarmuka Kelas Map pada gambar 4.7 merupakan antarmuka untuk menunjuk  
14         lokasi pada peta. Terdapat satu buah tombol yang akan dimunculkan jika pengguna sudah  
15         memilih lokasi. Jika tombol ditekan maka kordinat lokasi akan disimpan dan dikirim pada  
16         kelas MainPage dan halaman akan diarahkan ke kelas MainPage.

Gambar 4.8: Antarmuka *Route*

#### **4.3.3 Antarmuka Kelas *Route***

Antarmuka Kelas *Route* pada gambar 4.8 merupakan antarmuka untuk melihat rute dari lokasi asal ke lokasi tujuan dalam bentuk daftar maupun peta. Terdapat empat buah tombol pada antarmuka Kelas *Route*. Berikut tombol yang terdapat pada Kelas *Route*.

- Tombol prev  
Jika tombol ditekan maka akan menunjuk titik sebelumnya pada rute peta.
- Tombol next  
Jika tombol ditekan maka akan menunjuk titik setelahnya pada rute peta.
- Tombol here  
Jika tombol ditekan maka akan menunjuk lokasi perangkat berada pada peta.
- Tombol Show List  
Jika tombol ditekan maka akan menunjuk atau menyembunyikan daftar rute.



Gambar 4.9: Antarmuka Rute dalam bentuk Daftar

Antarmuka rute dalam bentuk daftar pada gambar 4.9 merupakan antarmuka untuk melihat rute secara lebih jelas dengan keterangan tahap demi tahap disertai jarak dan waktu perjalanan. Antarmuka daftar dapat dilihat atau disembunyikan sesuai keinginan pengguna namun saat kelas Rute dibuka antarmuka daftar rute akan disembunyikan.



## BAB 5

### IMPLEMENTASI DAN PENGUJIAN APLIKASI

Pada bab 5 akan dibahas implementasi dan pengujian aplikasi pencari rute kendaraan umum untuk Windows Phone.

#### 5.1 Implementasi

Pada sub bab ini akan dijelaskan mengenai lingkungan yang digunakan untuk membangun aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone. Pada lingkungan yang akan dibahas juga penulis membangun aplikasi sesuai rancangan yang telah dibahas pada bab 4 dan mengujinya.

##### 5.1.1 Perangkat Keras untuk Implementasi

Dalam membangun aplikasi ini perangkat keras yang digunakan adalah sebagai berikut:

1. Komputer

- (a) Processor: intel Core i7-2620M CPU 2,7 GHz
- (b) RAM: 4 GB
- (c) Hardisk: 640 GB
- (d) VGA: Intel HD 3000

2. Perangkat Bergerak

- (a) Processor: 1,2 GHz
- (b) RAM: 1 GB
- (c) ROM: 8 GB
- (d) Layar: 720 x 1280 pixel, 4,7 inch
- (e) GPS
- (f) Sensor: kompas, *accelerometer*

##### 5.1.2 Perangkat Lunak untuk Implementasi

Dalam membangun aplikasi ini perangkat lunak yang digunakan adalah sebagai berikut:

1. Komputer

- (a) Sistem Operasi Windows 8.1
- (b) IDE Visual Studio Express 2012
- (c) Bahasa Pemrograman C#
- (d) Library .Net Framework 4.5

2. Perangkat Bergerak

- (a) Sistem Operasi Windows Phone 8.1

### **5.1.3 Hasil Implementasi**

Hasil implementasi dari perangkat lunak ini terbagi dalam tiga bagian, yaitu:

1. Kode Program

Kode Program pada perangkat lunak ditulis dengan menggunakan bahasa c#. Bahasa C# dipilih berdasarkan analisa pada bab 3 dan kemampuan penulis.

2. Hasil kompilasi program

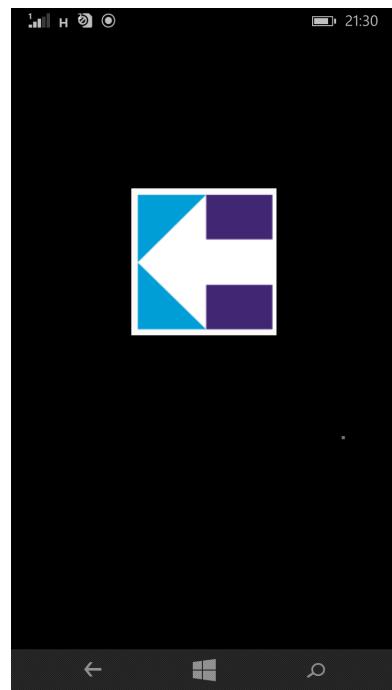
Hasil dari kompilasi program berupa file Kiri\_Debug\_AnyCPU.xap. File ini dapat dipasang pada perangkat dengan sistem operasi Windows Phone versi 8 atau lebih tinggi.

3. Antarmuka Aplikasi

Berikut merupakan hasil implementasi antarmuka aplikasi Pencari Rute Kendaraan Umum untuk Windows phone.



Gambar 5.1: Gambar antarmuka kelas MainPage



Gambar 5.2: Gambar antarmuka Splash di kelas MainPage



Gambar 5.3: Gambar antarmuka *list* asal dan *list* tujuan di kelas MainPage



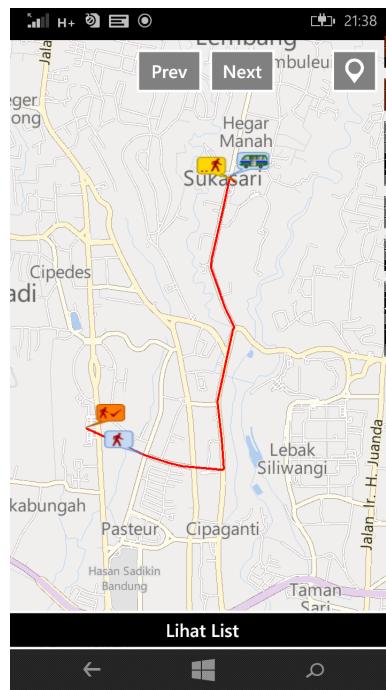
Gambar 5.4: Gambar antarmuka kelas Map



Gambar 5.5: Gambar antarmuka menunggu di kelas Route

## <sup>1</sup> 5.2 Pengujian

<sup>2</sup> Pada bagian ini akan dibahas mengenai hasil pengujian yang telah dilakukan terhadap  
<sup>3</sup> aplikasi yang telah dibangun penulis. Pengujian yang dilakukan terdiri dari dua bagian  
<sup>4</sup> yaitu pengujian fungsional dan pengujian experimental. Pengujian fungsional bertujuan  
<sup>5</sup> untuk memastikan semua fungsi aplikasi berjalan sesuai harapan. Sementara pengujian  
<sup>6</sup> eksperimental bertujuan untuk mengetahui keberhasilan proses kerja dari aplikasi.



Gambar 5.6: Gambar antarmuka kelas Route



Gambar 5.7: Gambar antarmuka *list* di kelas Route

### <sup>1</sup> 5.2.1 Lingkungan Pengujian

- <sup>2</sup> Dalam proses pengujian perangkat lunak penulis menggunakan sistem operasi Windows Phone 8.1 dengan spesifikasi perangkat keras sebagai berikut.
- <sup>4</sup> 1. Processor : 1.2 Ghz Quad Core
  - <sup>5</sup> 2. RAM : 1 GB
  - <sup>6</sup> 3. Layar : 1280 x 720 pixels, 4,7 inch
  - <sup>7</sup> 4. GPS : A-GPS, GLONASS, Beidou

### **5.2.2 Pengujian Fungsional**

Pengujian fungsional dilakukan untuk menguji kesesuaian reaksi yang terjadi dengan reaksi yang diharapkan. Untuk menguji kesesuaian reaksi yang terjadi dengan reaksi yang diharapkan penulis menguji 5 orang pengguna dengan perintah sebagai berikut.

1. Carilah rute dari sini ke BIP!
  2. Carilah rute dari BIP ke Ciwalk!
  3. Carilah rute dari sini ke ITB pintu jalan Ganesa!
- Dari tiga perintah berikut didapat hasil sebagai berikut.

Pengguna	Perintah	Aksi Pengguna	Reaksi yang Diharapkan
1	1	Menekan tombol "Here" dan menulis "BIP" pada TextBox	sesuai
	2	Menulis "BIP" pada TextBox dan menulis "Ciwalk" pada TextBox	sesuai
	3	Menekan tombol "Here" dan menulis "jl Ganesa" pada TextBox	tidak sesuai
2	1	Menekan tombol "Here" dan menulis "BIP" pada TextBox	sesuai
	2	Menulis "BIP" pada TextBox dan menulis "Ciwalk" pada TextBox	sesuai
	3	Menekan tombol "Here" dan menunjuk pada peta	sesuai
3	1	Menekan tombol "Here" dan menulis "BIP" pada TextBox	sesuai
	2	Menulis "BIP" pada TextBox dan menulis "Ciwalk" pada TextBox	sesuai
	3	Menekan tombol "Here" dan menulis "ITB Ganeca" pada TextBox lalu menyadari tombol map dan menggunakannya	sesuai
4	1	Menekan tombol "Here" dan menulis "BIP" pada TextBox	sesuai
	2	Menulis "BIP" pada TextBox dan menulis "Ciwalk" pada TextBox	sesuai
	3	Menekan tombol "Here" dan menulis "Institut Teknologi Bandung" pada TextBox	tidak sesuai

Tabel 5.1: Tabel Hasil pengujian fungsionalitas terhadap pengguna

Hasil pengujian tersebut ditunjukkan pada tabel [5.2](#).

### **5.2.3 Pengujian Experimental**

Pengujian eksperimental dilakukan untuk mengetahui keberhasilan aplikasi yang dibangun penulis. Berikut pengujian eksperimental yang dilakukan penulis.

1        1. Pengujian 1

- 2            • Tempat : Rumah(Jalan Kejaksaan menuju Unpar)
- 3            • Waktu : Pukul 9 pada tanggal 7 April 2014 Pada pengujian 1 penulis memasukan  
4            alamat di dalam rumah untuk menuju Unpar. Saat penulis memasukan kata  
5            Unpar muncul daftar tempat sesuai kata kunci "Unpar", lalu penulis memilih  
6            "Universitas Katolik Parahyangan".

7   **5.2.4 Perbandingan Aplikasi Pencari Rute di Android dengan Aplikasi  
8        Pencari Rute di Windows Phone**

9        Aplikasi Pencari rute yang penulis buat dengan yang ada di Android memiliki beberapa  
10      perbedaan. Berikut perbedaan antara aplikasi pencari rute di Android dengan Windwows  
11      Phone.

No	Aksi Pengguna	Reaksi yang Diharapkan	Reaksi Perangkat lunak
1	Menjalankan aplikasi	Aplikasi menampilkan SplashScreen selama beberapa saat dan tampilan awal halaman MainPage ditampilkan	sesuai
2	Memasukan lokasi asal dan lokasi tujuan dari <i>textbox</i>	<i>Textbox</i> terisi sesuai masukan	sesuai
3	Memasukan lokasi asal atau lokasi tujuan berdasarkan lokasi perangkat dengan menekan tombol "here"	<i>Textbox</i> lokasi asal atau lokasi tujuan terisi "here"	sesuai
4	Menekan tombol "maps" untuk memilih lokasi pada peta	Pindah halaman ke kelas Map	sesuai
5	Menekan lokasi pada peta lalu menekan tombol "pilih lokasi"	Lokasi ditandai dan pengguna diarahkan kembali ke kelas Main Page	sesuai
6	Memilih kota	Kota berubah sesuai yang dipilih pengguna	sesuai
7	Menekan tombol "Find"	Bila lokasi diambil dari peta dan lokasi perangkat maka tidak akan tampil <i>ListBox</i> dan antarmuka dialihkan ke kelas Route	sesuai
8		Bila input masukan berupa kata kunci tempat maka akan tampil <i>ListBox</i> untuk dipilih, lalu setelah dipilih antarmuka dialihkan ke kelas Route	sesuai
9	Bila <i>ListBox</i> ditampilkan dan pengguna memilih	<i>ListBox</i> akan tertutup	sesuai
10	Pada kelas Route pengguna menekan tombol "here"	Pusat peta akan diarahkan ke lokasi pengguna berada	sesuai
11	Pada kelas Route pengguna menekan tombol "Tunjukan Daftar"	Jika daftar tertutup maka daftar akan terbuka	sesuai
12		Jika daftar terbuka maka daftar akan tertutup	sesuai
13	Pada kelas Route pengguna menekan tombol "Next"	Pusat peta akan diarahkan ke pertantian transportasi berikutnya sesua kembalian Kiri disertai keterangan	sesuai
14	Pada kelas Route pengguna menekan tombol "Prev"	Pusat peta akan diarahkan ke pertantian transportasi sebelumnya sesua kembalian Kiri disertai keterangan	sesuai

Tabel 5.2: Tabel Hasil pengujian fungsionalitas

Perbedaan	Android	Windows Phone
1	1	1
2	2	2
3	3	3

Tabel 5.3: Tabel perbedaan



<sup>1</sup> **BAB 6**

<sup>2</sup> **KESIMPULAN DAN SARAN**

<sup>3</sup> **6.1 Kesimpulan**

<sup>4</sup> Berikut adalah kesimpulan yang didapat berdasarkan penelitian penulis.

- <sup>5</sup> 1. Pengguna dapat memasukan lokasi berdasarkan peta, lokasi perangkat, dan dengan  
<sup>6</sup> kata kunci.

<sup>7</sup> **6.2 Saran**

<sup>8</sup> Berdasarkan hasil kesimpulan yang telah dipaparkan, penulis memberi saran sebagai  
<sup>9</sup> berikut.

- <sup>10</sup> 1. Memanfaatkan tombol "enter" untuk memanggil *method startRoute*. Masukan dari  
<sup>11</sup> pengguna karena setelah pengguna menentukan tujuan tombol "Find" terhalang *ke-*  
<sup>12</sup> *yboard* maka pengguna harus menekan tombol "back" terlebih dahulu untuk menekan  
<sup>13</sup> tombol "Find".



1

## BIBLIOGRAFI

- 2 [1] “Windows phone silverlight development.” <http://msdn.microsoft.com/library/windows/apps/ff402535.aspx>, 2014. Accessed: 2014-8-17.
- 3
- 4 [2] P. Manning, *Pro Windows Phone App Development*. Apress, 2013.
- 5 [3] A. Whitechapel and S. McKenna, *Windows Phone 8 Development Internals*. Microsoft,
- 6 2013.
- 7 [4] T. Bray, “The JavaScript Object Notation (JSON) Data Interchange Format.” RFC 7159
- 8 (Proposed Standard), Mar. 2014.
- 9 [5] “Kiri api v2 documentation.” [https://bitbucket.org/projectkiri/kiri\\_api/wiki/KIRI%20API%20v2%20Documentation](https://bitbucket.org/projectkiri/kiri_api/wiki/KIRI%20API%20v2%20Documentation), 2014. Accessed: 2014-8-17.
- 10