

**SKRIPSI**

**PENGEMBANGAN APLIKASI PENCARI  
RUTE KENDARAAN UMUM  
UNTUK WINDOWS PHONE**



**YOHAN**

**NPM: 2011730048**

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



**UNDERGRADUATE THESIS**

**DEVELOPMENT APPLICATION PUBLIC TRANSPORT  
ROUTE SEARCH FOR WINDOWS PHONE**



**YOHAN**

**NPM: 2011730048**

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

**2014**



## **ABSTRAK**

Transportasi menjadi bagian yang tidak terpisahkan dalam kehidupan manusia saat penelitian ini dibuat. Namun saat ini banyak orang yang lebih memilih menggunakan kendaraan pribadi dibanding kendaraan umum. Maraknya penggunaan kendaraan pribadi menimbulkan masalah pemanasan global, kemacetan, dan harga bahan bakar minyak yang tinggi.

Pesatnya perkembangan teknologi membuat perangkat bergerak kian digemari di Indonesia. Salah satu sistem operasi yang meramaikan industri perangkat bergerak adalah Windows Phone. Windows Phone merupakan sistem operasi buatan Microsoft. Salah satu yang membedakan sistem operasi Windows Phone dengan sistem operasi lain adalah antarmuka yang Microsoft sebut Metro. Saat penelitian ini dilakukan sistem operasi Windows Phone adalah versi 8.

Perangkat lunak yang berhasil dibangun pada penelitian ini menggunakan bahasa C# untuk Windows Phone. Untuk mencari rute angkutan umum penulis memanfaatkan Kiri API dan memanfaatkan *web service* untuk mendapatkan rute yang diinginkan pengguna.

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



## **ABSTRACT**

Under Construction.

**Keywords:** Route, Public Transport, Windows Phone



# DAFTAR ISI

<b>DAFTAR ISI</b>	<b>ix</b>
<b>DAFTAR GAMBAR</b>	<b>xi</b>
<b>DAFTAR TABEL</b>	<b>xii</b>
<b>1 PENDAHULUAN</b>	<b>1</b>
1.1 Latar Belakang . . . . .	1
1.2 Rumusan Masalah . . . . .	2
1.3 Tujuan . . . . .	2
1.4 Batasan Masalah . . . . .	2
1.5 Metode Penelitian . . . . .	2
1.6 Sistematika Penulisan . . . . .	3
<b>2 DASAR TEORI</b>	<b>5</b>
2.1 Windows Phone . . . . .	5
2.1.1 Lingkungan Kerja . . . . .	5
2.1.2 XAML . . . . .	6
2.1.3 Kontrol terhadap Ponsel . . . . .	6
2.1.4 Siklus Hidup Aplikasi . . . . .	10
2.1.5 Peta di Windows Phone . . . . .	12
2.1.6 Lokasi . . . . .	17
2.1.7 Memanfaatkan Sumber Data . . . . .	20
2.2 Kiri API . . . . .	23
2.2.1 <i>Web Service</i> Penentuan Rute . . . . .	23
2.2.2 <i>Web Service</i> Pencarian Lokasi . . . . .	25
2.2.3 <i>Web Service</i> Menemukan Transportasi Terdekat . . . . .	26
<b>3 ANALISIS</b>	<b>29</b>
3.1 Analisis Aplikasi Sejenis . . . . .	29
3.2 Analisis Aplikasi . . . . .	32
3.2.1 Kebutuhan Aplikasi . . . . .	32
3.2.2 Analisis Kontrol yang Dipakai . . . . .	32
3.2.3 Analisis Terhadap Siklus Hidup Aplikasi . . . . .	33
3.2.4 Analisis Peta . . . . .	34
3.2.5 Analisis Pemanfaatan Sumber Data . . . . .	36
3.2.6 Analisis Kiri API . . . . .	36
3.2.7 Diagram <i>Use Case</i> dan Scenario . . . . .	39
3.2.8 Kelas Diagram . . . . .	43
<b>4 PERANCANGAN</b>	<b>45</b>
4.1 Diagram <i>Sequence</i> . . . . .	45
4.2 Perancangan Kelas . . . . .	48

4.2.1	Kelas <i>PhoneApplicationPage</i>	48
4.2.2	Kelas <i>MainPage</i>	48
4.2.3	Kelas <i>City</i>	50
4.2.4	Kelas <i>BackgroundWorker</i>	51
4.2.5	Kelas <i>Geocoordinate</i>	51
4.2.6	Kelas <i>Geolocator</i>	51
4.2.7	Kelas <i>Geoposition</i>	51
4.2.8	Kelas <i>LocationFinder</i>	51
4.2.9	Kelas <i>Map</i>	53
4.2.10	Kelas <i>HttpClient</i>	53
4.2.11	Kelas <i>JsonConvert</i>	54
4.2.12	Kelas <i>Protocol</i>	54
4.2.13	Kelas <i>RootObjectSearchPlace</i>	56
4.2.14	Kelas <i>SearchResult</i>	56
4.2.15	Kelas <i>RootObjectFindRoute</i>	56
4.2.16	Kelas <i>RoutingResult</i>	56
4.2.17	Kelas <i>Route</i>	56
4.3	Perancangan Antar Muka	58
4.3.1	Antarmuka Kelas <i>MainPage</i>	58
4.3.2	Antarmuka Kelas <i>Map</i>	59
4.3.3	Antarmuka Kelas <i>Route</i>	60
<b>5</b>	<b>IMPLEMENTASI DAN PENGUJIAN APLIKASI</b>	<b>63</b>
5.1	Implementasi	63
5.1.1	Perangkat Keras untuk Implementasi	63
5.1.2	Perangkat Lunak untuk Implementasi	63
5.1.3	Hasil Implementasi	64
5.2	Pengujian	66
5.2.1	Lingkungan Pengujian	67
5.2.2	Pengujian Fungsional	68
5.2.3	Pengujian Experimental	69
5.2.4	Perbandingan Aplikasi Pencari Rute di Android dengan Aplikasi Pencari Rute di Windows Phone	69
<b>6</b>	<b>KESIMPULAN DAN SARAN</b>	<b>73</b>
6.1	Kesimpulan	73
6.2	Saran	73
<b>DAFTAR REFERENSI</b>		<b>75</b>
<b>DAFTAR REFERENSI</b>		<b>77</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 [3] . . . . .	14
2.9	Tampilan polyline pada Peta . . . . .	15
3.1	Tampilan utama aplikasi Public Transport . . . . .	29
3.2	Menunjuk lokasi pada peta . . . . .	30
3.3	Memberikan daftar nama tempat dan nama jalan terkait . . . . .	30
3.4	Tampilan rute kendaraan umum dalam bentuk daftar . . . . .	31
3.5	Tampilan rute kendaraan umum di peta . . . . .	31
3.6	Tampilan <i>pushpin</i> untuk angkot . . . . .	35
3.7	Tampilan <i>pushpin</i> untuk jalan kaki . . . . .	35
3.8	Diagram <i>use case</i> . . . . .	41
3.9	Diagram Kelas . . . . .	43
4.1	Diagram <i>sequence</i> Mendapatkan Kordinat perangkat . . . . .	45
4.2	Diagram <i>sequence</i> Mendapatkan Kordinat pada Peta . . . . .	46
4.3	Diagram <i>sequence</i> Mendapatkan Rute . . . . .	47
4.4	Diagram Kelas . . . . .	48
4.5	Antarmuka <i>MainPage</i> . . . . .	58
4.6	Antarmuka Daftar Tempat . . . . .	60
4.7	Antarmuka <i>Map</i> . . . . .	60
4.8	Antarmuka <i>Route</i> . . . . .	61
4.9	Antarmuka Rute dalam bentuk Daftar . . . . .	61
5.1	Gambar antarmuka kelas <i>MainPage</i> . . . . .	64
5.2	Gambar antarmuka <i>Splash</i> di kelas <i>MainPage</i> . . . . .	65
5.3	Gambar antarmuka <i>list</i> asal dan <i>list</i> tujuan di kelas <i>MainPage</i> . . . . .	65
5.4	Gambar antarmuka kelas <i>Map</i> . . . . .	66
5.5	Gambar antarmuka menunggu di kelas <i>Route</i> . . . . .	66
5.6	Gambar antarmuka kelas <i>Route</i> . . . . .	67
5.7	Gambar antarmuka <i>list</i> di kelas <i>Route</i> . . . . .	67

## DAFTAR TABEL

2.1	Properti kelas Map . . . . .	15
2.2	Properti <i>Polyline Class</i> . . . . .	17
2.3	Kelas pada <i>Namespace Geolocator</i> . . . . .	19
2.4	Properti pada <i>Geocoordinate</i> . . . . .	19
2.5	Tabel parameter layanan penentuan rute . . . . .	24
2.6	Tabel parameter layanan pencarian lokasi . . . . .	25
2.7	Tabel parameter layanan menemukan transportasi terdekat . . . . .	26
3.1	Skenario mandapatkan lokasi untuk masukan lokasi asal dan lokasi tujuan . . . . .	41
3.2	Skenario memasukan lokasi asal dan lokasi tujuan pada <i>TextBox</i> . . . . .	42
3.3	Skenario menunjuk lokasi asal dan lokasi tujuan pada peta . . . . .	42
3.4	Skenario memilih alamat . . . . .	42
3.5	Skenario menampilkan rute kendaraan umum . . . . .	43
5.1	Tabel Hasil pengujian fungsionalitas terhadap pengguna . . . . .	68
5.2	Tabel Hasil pengujian fungsionalitas . . . . .	70
5.3	Tabel perbedaan . . . . .	71

<sup>1</sup>

## BAB 1

<sup>2</sup>

### PENDAHULUAN

<sup>3</sup> Pada Bab satu akan dibahas pendahuluan dari penelitian yang dilakukan. Bab satu terbagi  
<sup>4</sup> dalam delapan *sub bab*, yaitu *latar belakang, rumusan masalah, tujuan, batasan masalah,*  
<sup>5</sup> *ruang lingkup masalah, metode penelitian, teknik pengumpulan data, dan sistematika penu-*  
<sup>6</sup> *lisan.*

<sup>7</sup> **1.1 Latar Belakang**

<sup>8</sup> Transportasi menjadi bagian yang penting bagi manusia di saat penelitian ini dilakukan.  
<sup>9</sup> Ada dua jenis transportasi bagi seseorang yaitu kendaraan umum dan kendaraan pribadi.  
<sup>10</sup> Kenyataannya pada saat penelitian ini dilakukan banyak yang lebih memilih kendaraan  
<sup>11</sup> pribadi dibanding kendaraan umum. Maraknya penggunaan kendaraan pribadi dan penam-  
<sup>12</sup> bahan jalur kendaraan yang tidak sebanding banyaknya kendaraan menimbulkan kemacetan.  
<sup>13</sup> Maraknya penggunaan kendaraan pribadi dikarenakan kurang nyamannya kendaraan umum  
<sup>14</sup> dan kesulitan dalam menentukan kendaraan umum yang harus dinaiki. Banyaknya rute ken-  
<sup>15</sup> daraan umum membuat orang kebingungan dalam memilih kendaraan umum menuju lokasi  
<sup>16</sup> yang diinginkan. Seseorang cenderung malas untuk bertanya dan mencari rute yang efisien.  
<sup>17</sup> Karena hal tersebut, seseorang lebih memilih menggunakan kendaraan pribadi ketimbang  
<sup>18</sup> kendaraan umum.

<sup>19</sup> Ide pembuatan aplikasi yang memudahkan seseorang dalam menentukan rute kendara-  
<sup>20</sup> an umum sudah lebih dulu ada yang dikenal dengan nama Kiri. Kiri dibuat dengan latar  
<sup>21</sup> belakang

<sup>22</sup> tiga masalah besar yaitu pemanasan global, kemacetan, dan harga bahan bakar minyak yang  
<sup>23</sup> tinggi<sup>1</sup>. Meskipun Kiri pertama dibuat di web tetapi Kiri dapat dimanfaatkan untuk pen-  
<sup>24</sup> carian kendaraan selain di web. Pemanfaatan Kiri tersebut dalam mencari rute kendaraan  
<sup>25</sup> umum dengan menggunakan Kiri API.

<sup>26</sup> Pesatnya perkembangan teknologi sekarang ini mendorong perkembangan perangkat ber-  
<sup>27</sup> gerak (*mobile*). Perangkat bergerak kian digemari orang-orang terutama di Indonesia. Salah  
<sup>28</sup> satu yang menarik perhatian adalah Windows Phone 8 yang dibuat Microsoft. Antarmuka  
<sup>29</sup> Windows Phone 8 yang disebut *Metro* cukup menarik dan mudah digunakan.

<sup>30</sup> Berdasarkan hal tersebut, penulis mencoba mengembangkan aplikasi Pencarian Rute  
<sup>31</sup> Kendaraan Umum di Windows Phone dalam tugas akhir ini. Aplikasi yang penulis kem-  
<sup>32</sup> bangan akan memungkinkan pengguna menemukan rute kendaraan umum untuk sampai di  
<sup>33</sup> tujuan. Untuk memudahkan pengguna, penulis akan menampilkan dalam dua bentuk yaitu

---

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

1 peta dan daftar.

## 2 **1.2 Rumusan Masalah**

3 Sehubung dengan latar belakang diatas timbul permasalahan sebagai berikut:

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

## 7 **1.3 Tujuan**

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

- 10    ● Mempelajari cara pembuatan perangkat lunak di Windows Phone lalu mengembangkan  
11      aplikasi yang akan dibuat.
- 12    ● Membuat aplikasi di Windows Phone yang memanfaatkan Kiri API.

## 13 **1.4 Batasan Masalah**

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

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

## 20 **1.5 Metode Penelitian**

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

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

- 1     • Menguji aplikasi Pencarian Rute Kendaraan Umum untuk Windows Phone.
- 2     • Membuat kesimpulan.

### **3 1.6 Sistematika Penulisan**

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

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

13    Bab 3 membahas tentang analisis pembangunan aplikasi Pencarian Rute Kendaraan  
14    Umum untuk Windows Phone. Pada Bab 3 akan dibahas mengenai analisis kebutuhan apli-  
15    kasi, analisis kontrol yang dipakai, analisis terhadap siklus hidup aplikasi, analisis terhadap  
16    siklus hidup aplikasi, analisis peta, analisis memanfaatkan sumber data, analisis Kiri API,  
17    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>[3]</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

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

### 3 2.1.2 XAML

4       *Extensible Application Markup Language* (XAML) merupakan bahasa deklaratif yang  
5 dipakai untuk membuat antarmuka aplikasi. XAML merupakan bahasa yang digunakan  
6 untuk membuat antarmuka di Windows Phone 8[3]. Pada dasarnya penggunaan XAML  
7 sama dengan HTML pada pembuatan antarmuka web. XAML dapat menginisialisasi objek  
8 dan mengatur properti untuk menunjukkan hubungan antar objek.

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

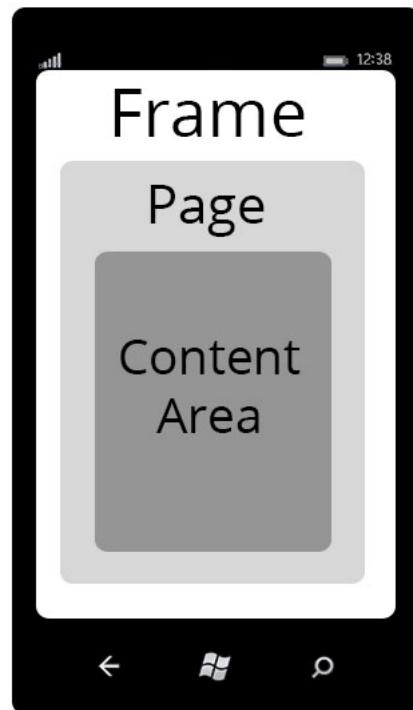
- 15       • x:Key : sebuah nama unik untuk menunjuk referensi ke suatu resource atau berkas  
16 lain. Nilai ini dapat dipanggil kembali untuk menggunakan resource tersebut.
- 17       • x:Class : menunjukkan nama kelas.
- 18       • x:Name : menunjukkan nama sebuah obyek dan untuk membedakan antar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

### 23 2.1.3 Kontrol terhadap Ponsel

24       Maksud dari kontrol terhadap ponsel adalah pengaturan tata letak terhadap antarmuka  
25 di Windows Phone[1]. Windows Phone 8 menyediakan banyak set kontrol yaitu tata letak,  
26 tombol, kontrol masukan untuk mendapatkan informasi sampai ke menu.

#### 27 2.1.3.1 Navigasi

28       Aplikasi yang dibuat di Windows Phone didasarkan pada model halaman. Model ha-  
29 laman adalah pengguna berpindah dari satu halaman ke halaman lain dengan konten yang  
30 berbeda dan *frame* sebagai pengontrolnya. Setiap antarmuka aplikasi dibungkus dengan  
31 *frame*. *Frame* ini yang akan melakukan kontrol terhadap aplikasi dan memungkinkan per-  
32 pindahan dari satu halaman ke halaman lain. Sedangkan halaman merupakan pembungkus  
33 dari elemen di dalamnya saja. Untuk lebih jelas mengenai *frame*, halaman, dan area konten  
34 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>
```

### 1 2.1.3.3 Kontrol Terhadap Teks

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

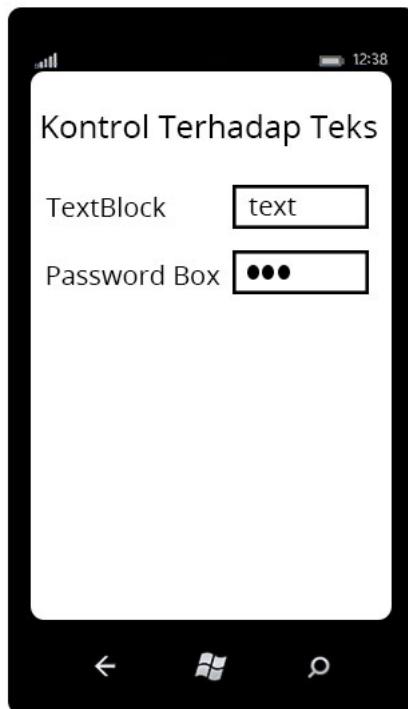
- 6
- *TextBlock* merupakan tempat menaruh potongan teks yang hanya bisa dilihat.

7

  - *TextBox* biasanya digunakan untuk teks masukan yang pendek. Tapi bisa juga dipakai  
 8 untuk masukan yang banyak dan beberapa baris.

9

  - *PasswordBox* biasanya digunakan untuk masukan yang bersifat rahasia. Karakter  
 10 yang dimasukan langsung disamarkan menjadi bentuk titik.



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

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

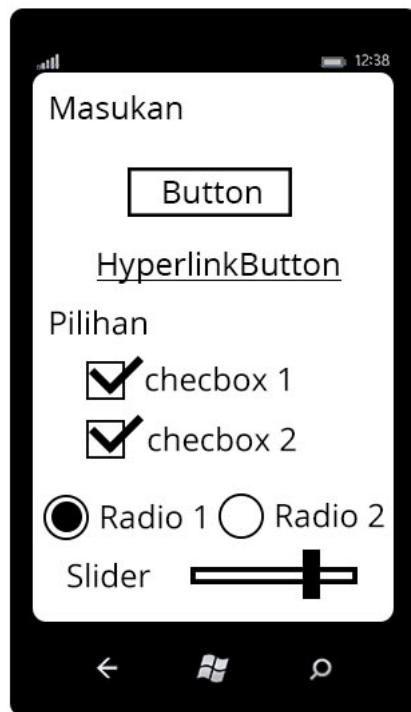
```
11 | <TextBlock x:Name="TextBlock1" Text="TextBlock"/>
12 | <TextBox x:Name="TextBox1" Text="TextBox"/>
```

### 13 2.1.3.4 Tombol dan Kontrol Pilihan

14 Tombol memungkinkan pengguna untuk berpindah halaman. Sedangkan kontrol pilihan  
 15 memudahkan dalam memilih. Berikut keterangan, gambar 2.3, dan kode pada *listing 2.3*  
 16 tombol dan kontrol pilihan.

- 17
- *Button* merupakan kontrol yang dipakai pengguna untuk mengaktifkan *event* tekan.

- 1     • *HyperlinkButton* merupakan kontrol yang menampilkan tautan. Jika *HyperlinkButton* ditekan maka akan berpindah ke halaman yang akan dituju.
- 2
- 3     • *CheckBox* merupakan kontrol yang memungkinkan pengguna memilih beberapa item.
- 4
- 5     • *RadioButton* merupakan kontrol yang memungkinkan pengguna memilih satu pilihan dari beberapa pilihan.
- 6
- 7     • *Slider* merupakan kontrol yang memungkinkan pengguna memilih nilai kisaran dari jalur yang sudah disediakan.



Gambar 2.3: Gambar kontrol pada Windows Phone

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

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

#### 9 2.1.3.5 Kontrol Daftar

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

- 12     • *ListBox* akan menampilkan daftar *item*. Daftar ini dapat dipilih dengan cara ditekan.
- 13     • *LongListSelector* dipakai untuk mengelompokan, menampilkan, dan melakukan penggulungan terhadap daftar yang panjang.
- 14



Gambar 2.4: Antarmuka *ListBox*

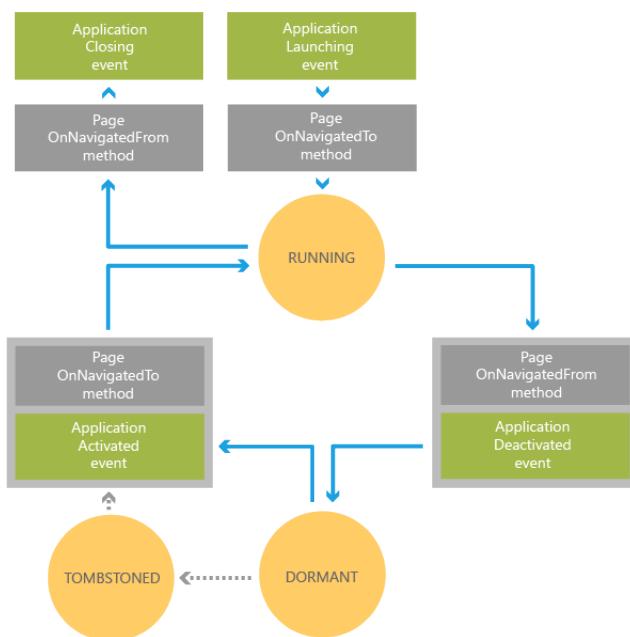
Listing 2.4: Kode untuk menampilkan *listBox*

```

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 apli-  
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 menun-  
18 jukan peristiwa aplikasi atau tingkat peristiwa di halaman. Berikut ini adalah keterangan  
19 untuk siklus hidup Windows Phone pada gambar 2.5.

20 • *The Launching Event*

21 Merupakan tampilan awal saat aplikasi dipilih yang memberitahukan pengguna bah-

1 wa aplikasi sedang dijalankan. *Event* ini akan dipanggil ketika aplikasi di jalankan  
2 pertama kali. *Event* ini akan berjalan di belakang (*background processing*) ketika apli-  
3 kasi ditutup sementara atau sedang berada pada keadaan *Dormant* atau *Tombstoned*  
4 menjadi *running*.

5 ● *Running*

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

11 ● *method OnNavigatedFrom*

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

18 ● *The Deactivated Event*

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

25 ● *Dormant*

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

31 ● *Tombstoned*

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

34 ● *The Activated Event*

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

37 ● *The OnNavigatedTo Method*

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

1     ● *The Closing Event*

2     Event ini akan tercapai ketika pengguna berpindah mundur keluar dari halaman utama.  
3     Pada kasus ini, aplikasi akan dihentikan dan tidak ada keadaan yang disimpan.

4     **2.1.5 Peta di Windows Phone**

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

9     **2.1.5.1 Penambahan Peta Ke Aplikasi**

10    Untuk penambahan peta pada Windows Phone menggunakan kontrol peta. Kontrol peta  
11    merupakan bagian dari *library* Windows Phone. Dengan begitu untuk dapat menggunakan-  
12    nya perlu direferensikan. Untuk dapat menggunakannya juga harus ditambah *capability*  
13    ID\_CAP\_MAP. Setelah hal tersebut dilakukan barulah peta dapat ditampilkan. Berikut  
14    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

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

Listing 2.6: Menampilkan peta dengan nama MyMap dari kode program

```
16 | public void mapFrom()
17 | {
18 |     InitializeComponent();
19 |     Map MyMap = new Map();
20 |     ContentPanel.Children.Add(MyMap);
21 | }
```

22    **2.1.5.2 Tampilan Peta di Windows Phone**

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

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

**Listing 2.7:** Mengatur tingkat zoom dari XAML

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

**Listing 2.8:** Mengatur tingkat zoom dari dari kode program

```
9 | public mapFrom()
10| {
11|     InitializeComponent();
12|     Map MyMap = new Map();
13|
14|     //Mengatur titik tengah peta
15|     MyMap.Center = new GeoCoordinate(47.6097, -122.3331);
16|
17|     //mengatur tingkat zoom
18|     MyMap.ZoomLevel = 10;
19|     ContentPanel.Children.Add(MyMap);
20| }
```

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

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



**Gambar 2.7:** *cartographic*

- Mode warna yang disediakan Windows Phone ada dua yaitu terang dan gelap. Secara *default* mode pada peta di Windows Phone adalah terang.
- Tampilan pada Peta di Windows Phone dapat berubah karena hasil diputar, dimiringkan, ditarik, dan diturunkan. Berikut beberapa hal yang dapat diatur sebagai tampilan di peta.
  - *Heading* merupakan representasi dari derajat secara geometri. Derajat ini didefinisikan dalam 0 sampai 360 yang dipakai untuk memutar peta. Contoh, 0 atau 360 ke arah utara, 90 ke arah barat, 180 ke arah selatan, dan 270 derajat ke arah timur.

- 1 – *Pitch* merupakan derajat kemiringan dari peta dari sudut pengguna. Contoh,  
 2 *Pitch* = 0 berarti melihat dari atas ke bawah sedangkan *Pitch* = 45 berarti  
 3 melihat dari samping ke bawah dengan sudut 45 derajat.

#### 4 2.1.5.3 Pushpin ke Peta

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



Gambar 2.8: Keluaran Toolkit Pushpin pada peta [3]

Listing 2.9: Kode untuk menampilkan pushpin

```
11 MapOverlay overlay = new MapOverlay
12 {
13     GeoCoordinate = map.Center,
14     Content = new Border
15     {
16         BorderBrush = new SolidColorBrush(Colors.FromArgb(120, 255, 0, 0)),
17         Child = new TextBlock(){Text="Pushpin"},
18         BorderThickness = new Thickness(1),
19         Background = new SolidColorBrush(Colors.FromArgb(120, 255, 0, 0)),
20         Width = 80,
21         Height = 60
22     }
23 };
24 MapLayer layer = new MapLayer();
25 layer.Add(overlay);
26
27 map.Layers.Add(layer);
```

#### 28 2.1.5.4 Polyline pada Peta

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

Listing 2.10: Kode untuk menampilkan polyline

```
34 MapPolyline line = new MapPolyline();
35 line.StrokeColor = Colors.Red;
```



Gambar 2.9: Tampilan polyline pada Peta

```

1   line.StrokeThickness = 10;
2   line.Path.Add(new GeoCoordinate(-6.8619546, 107.614441));
3   line.Path.Add(new GeoCoordinate(-6.908693, 107.611185));

```

#### 2.1.5.5 Namespace Control Map

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

#### 2.1.5.6 Kelas Map

Merupakan kelas yang mewakili kontrol map.

Berikut properti yang dapat digunakan pada kelas ini.

Nama	Deskripsi
<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

15

Berikut *method* yang dapat digunakan pada kelas ini.

16

- 1     ● *SetView(LocationRectangle)*
- 2         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai wilayah geografis. *method* ini tidak mengembalikan nilai.
- 4     ● *SetView(GeoCoordinate, Double)*
- 5         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah dan tingkat zoom. *method* ini tidak mengembalikan nilai.
- 7     ● *SetView(LocationRectangle, MapAnimationKind)*
- 8         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis dan animasi. *method* ini tidak mengembalikan nilai.
- 10    ● *SetView(LocationRectangle, Thickness)*
- 11         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai region geografis dengan batas tertentu. *method* ini tidak mengembalikan nilai.
- 13    ● *SetView(GeoCoordinate, Double, MapAnimationKind)*
- 14         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, dan animasi. *method* ini tidak mengembalikan nilai.
- 16    ● *SetView(GeoCoordinate, Double, Double)*
- 17         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, dan *heading*. *method* ini tidak mengembalikan nilai.
- 19    ● *SetView(LocationRectangle, Thickness, MapAnimationKind)*
- 20         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai wilayah geografis dengan batas tertentu, dan animasi. *method* ini tidak mengembalikan nilai.
- 22    ● *SetView(GeoCoordinate, Double, Double, MapAnimationKind)*
- 23         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, *heading*, dan animasi. *method* ini tidak mengembalikan nilai.
- 25    ● *SetView(GeoCoordinate, Double, Double, Double)*
- 26         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, *heading*, *pitch*. *method* ini tidak mengembalikan nilai.
- 28    ● *SetView(GeoCoordinate, Double, Double, Double, MapAnimationKind)*
- 29         *method* untuk mengatur pandangan di atas peta secara spesifik sesuai titik tengah, tingkat zoom, *heading*, *pitch*, dan animasi. *method* ini tidak mengembalikan nilai.
- 31    ● *UpdateLayout*
- 32         *method* yang memastikan semua posisi objek turunan mengikuti tata letak.

---

### 33    **2.1.5.7    Polyline Class**

34         Merupakan kelas yang dipakai untuk menggambarkan garis lurus yang saling terhubung.  
35         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-putus.
<i>StrokeThickness</i>	Mengatur dan mendapatkan lebar garis untuk menggambar <i>polyline</i> .

Tabel 2.2: Properti *Polyline Class*

- 1
- 2 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. Mes-  
13 kipun pushpin merupakan bawaan pada peta untuk menunjuk suatu lokasi tetapi *pushpin*  
14 dari peta tidak dapat diubah-ubah. *Pushpin* pada Windows Phone 8 dapat dibuat sesuai  
15 kebutuhan. Namun ada cara lain dengan menambahkan Windows Phone Toolkit. Windows  
16 Phone Toolkit mempunyai komponen untuk menggambar pushpin diatas peta.

#### 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 de-  
28 ngan berbagai tingkat akurasi dan konsumsi daya. Perangkat diatas juga berkomunikasi  
29 langsung untuk memutuskan sumber mana yang digunakan untuk menentukan lokasi per-  
30 angkat berdasarkan ketersediaan data lokasi dan prasyarat yang ditentukan aplikasi. Lapisan

1 diatas penyedia data lokasi tersebut adalah pengelola antarmuka. Aplikasi akan mengunakan  
2 antarmuka tersebut untuk memulai dan menghentikan layanan lokasi, mengatur tingkat  
3 akurasi, dan menerima data lokasi.

4 Karena pengguna dapat berpindah tempat untuk menuju tempat yang lain, maka pelacakan  
5 lokasi harus dilakukan terus menerus. Pelacakan lokasi secara terus menerus ini dapat  
6 dilakukan di depan maupun di belakang aplikasi Windows Phone 8. Pelacakan aplikasi di  
7 depan akan memungkinkan aplikasi melacak lokasi pengguna sekaligus melakukan perbaruan  
8 antarmuka. Jika pelacakan lokasi di belakang aplikasi maka tidak ada perubahan pada antarmuka  
9 namun pelacakan dilakukan secara terus menerus. Pelacakan yang terus menerus di  
10 belakang aplikasi akan membuat keadaan aplikasi cepat dipulihkan dari keadaan *Dormant*.

#### 11 2.1.6.1 Mendapatkan Posisi Pengguna

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

- 19 • *Ready* : Jika lokasi tersedia.
- 20 • *Initializing* : Jika status penangkap GPS belum memiliki cukup satelit untuk mendapatkan posisi yang akurat.
- 22 • *NoData* : Data lokasi belum tersedia. Status ini muncul jika aplikasi sedang memanggil *GetGeopositionAsync()* atau *register*.
- 24 • *Disable* : Status mengindikasikan tidak diperbolehkannya pengaksesan lokasi.
- 25 • *NotInitialized* : Data lokasi belum tersedia. Status ini muncul jika aplikasi belum memanggil *GetGeopositionAsync()* atau *register*.
- 27 • *NotAvailable* : Jika sensor arah mata angin dan lokasi tidak tersedia.

#### 28 2.1.6.2 Namespace Geolocator

29 *Namespace* merupakan nama yang dipakai untuk mengatur kelas-kelas. Windows Phone  
30 8 sudah menyediakan *namespace* bawaan untuk mengakses lokasi. *Namespace* yang disediakan  
31 adalah *namespace geolocator*. *Namespace* ini akan mengakses lokasi geografis dari  
32 perangkat dan mendukung pelacakan lokasi dari waktu ke waktu. Agar dapat menggunakan  
33 kelas pada *namespace* tersebut perlu ditambahkan *namespace* dan *capabilities*. *Namespace* yang  
34 harus ditambahkan pada baris awal XAML adalah **Windows.Device.Geolocator**.  
35 Selanjutnya ada penambahan *capabilities* *ID\_CAP\_LOCATION*. Penambahan *capabilities*  
36 ditambahkan pada *WMAppManifest.xml*. Kelas yang diatur oleh *namespace geolocator*  
37 dapat dilihat 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*

#### 1 2.1.6.3 *Geocoordinate*

2 Geocoordinate adalah kelas yang menunjukkan lokasi sebagai kordinat geografis. Kelas  
 3 ini hanya menyediakan properti yang hanya bisa dibaca. Kelas ini menyediakan properti  
 4 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*

#### 5 2.1.6.4 *Geolocator*

6 *Geolocator* merupakan kelas yang mendukung pengaksesan terhadap lokasi.

7 Berikut *method* yang disediakan *Geolocator*:

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

14 Berikut Properti yang disediakan kelas Geolocator:

- 15 • *public PositionStatus LocationStatus { get; }*  
 16 Merupakan properti dari kelas *geolocator* untuk mendapatkan status posisi dengan  
 17 mengembalikan kelas *PositionStatus*. Status pada kelas *PositionStatus* adalah *Ready*,  
 18 *Initializing*, *NoData*, *Disable*, *NotInitialized*, dan *NotAvailable*.

- 19 • *public PositionAccuracy DesiredAccuracy { get; set; }*  
 20 Properti yang digunakan untuk mengatur dan mendapatkan tingkat akurasi. Untuk  
 21 tingkat akurasi dapat dipilih tingkat *High* untuk tingkat akurasi tinggi dan dipilih  
 22 tingkat *Default* untuk menghemat daya. Keluaran dari properti ini adalah tipe data  
*PositionAccuracy*.

- 1     • *public Nullable<uint> DesiredAccuracyInMeters { get; set; }*

2         Sama seperti properti *DesiredAccuracy* diatas tetapi dalam satuan meter. Keluaran  
3         dari properti ini adalah tipe data *uint*.

- 4     • *public uint ReportInterval { get; set; }*

5         Merupakan properti untuk mendapatkan selang waktu pembaruan lokasi. Properti ini  
6         mengeluarkan tipe data unit.

7     **2.1.6.5 Geoposition**

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

- 10     • *public CivicAddress CivicAddress { get; }*

11         Data alamat sipil yang terkait dengan lokasi geografis.

- 12     • *public Geocoordinate Coordinate { get; }*

13         Data latitude dan longitude yang terkait lokasi geografis.

14     **2.1.7 Memanfaatkan Sumber Data**

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

19         Sebelum data dapat dikirim antar perangkat perlu dilakukan *Serialization*. *Serialization*  
20         disini merupakan proses mentransformasikan objek ke format yang bisa dengan mudah di-  
21         kirim melewati jaringan atau disimpan di database. Formatnya disini berupa string yang  
22         direpresentasikan sebagai objek di XML atau JSON(Javascript Object Notation). Ada beberapa  
23         objek yang dapat melakukan serialisasi, tetapi yang akan dibahas penulis disini hanya  
24         serialisasi JSON[3].

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

35     **2.1.7.1 HttpClient**

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

1     ● *DeleteAsync(Uri)*

2       *method* yang dipakai untuk mengirimkan permintaan DELETE ke URI yang spesifik  
3       sebagai operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memung-  
4       kinkan aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini  
5       membutuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembali-  
6       annya berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek  
7       tersebut memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan ke-  
8       majuan dari data yang dikirim.

9     ● *GetAsync(Uri)*

10      *method* yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik seba-  
11       gi operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
12       aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini mem-  
13       butuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya  
14       berupa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut  
15       memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari  
16       data yang dikirim.

17     ● *GetAsync(Uri,HttpCompletionOption)*

18      *method* yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik seba-  
19       gi operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
20       aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini mem-  
21       butuhkan parameter URI sebagai tujuan dari permintaan dan nilai tambahan yang  
22       dimaksukan sebagai indikasi operasi dianggap selesai. Sedangkan kembaliannya ber-  
23       upa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut  
24       memiliki 2 parameter yaitu hasil berupa pesan kembalian dari http dan kemajuan dari  
25       data yang dikirim.

26     ● *GetBufferAsync(Uri)*

27      *method* yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik seba-  
28       gi operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
29       aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini mem-  
30       butuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya ber-  
31       upa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut  
32       memiliki 2 parameter yaitu hasil berupa pesan kembalian yang dikirimkan secara  
33       buffer(disimpan dalam memori) dan kemajuan dari data yang dikirim.

34     ● *GetInputStreamAsync(Uri)*

35      *method* yang dipakai untuk mengirimkan permintaan GET ke URI yang spesifik seba-  
36       gi operasi *asynchronous*. Maksud dari operasi *asynchronous* adalah memungkinkan  
37       aplikasi untuk melanjutkan pekerjaan selagi *method* ini dipanggil<sup>2</sup>. *method* ini mem-  
38       butuhkan parameter URI sebagai tujuan dari permintaan. Sedangkan kembaliannya ber-  
39       upa objek yang mewakili operasi *asynchronous* disertai kemajuan. Objek tersebut  
40       memiliki 2 parameter yaitu hasil berupa pesan kembalian yang dikirimkan secara  
41       stream(langsung sesuai waktu) dan kemajuan dari data yang dikirim.

42     ● *GetStringAsync(Uri)*

1        *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.

8        • *PostAsync(Uri)*

9        *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.

16      • *SendRequestAsync(HttpRequestMessage)*

17      *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.

23      • *SendRequestAsync(HttpRequestMessage, HttpCompletionOption)*

24      *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.

31      **2.1.7.2 Json.NET**

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

- 35      • JsonConvert merupakan kelas yang berfungsi untuk konversi dari JSON String ke objek dan sebaliknya. Kelas ini memiliki dua buah *method* yaitu SerializeObject() dan DeserializeObject().

- 38      • JsonSerializer merupakan kelas yang berfungsi untuk konversi dari JSON String ke objek dan sebaliknya. Kelebihan yang dimiliki kelas ini adalah dapat membaca dan menulis JSON dari *file text*.

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

## 2 2.2 Kiri API

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

6 Pemanfaatan Kiri API dengan melakuan permintaan dengan parameter *POST* atau *GET*  
7 lalu Kiri akan mengembalikan hasil dalam format JSON. Permintaan tersebut dikirimkan  
8 ke URL atau *Uniform Resource Locator*. Berikut URL yang disediakan Kiri Api.

- 9 • <http://preview.kiri.travel/handle.php>

10 Merupakan URL untuk uji coba. Untuk kemampuannya juga menurut dokumentasi  
11 Kiri API masih tidak stabil.

- 12 • <http://kiri.travel/handle.php>

13 Merupakan URL produksi. Ini merupakan URL yang direkomendasikan untuk mena-  
14 ngani permintaan pengguna.

15 Untuk setiap permintaan membutuhkan *API key* yang didapat dengan mendaftar[2]. Peng-  
16 gunaan API memungkinkan pengaksesan di mana saja dengan menggunakan koneksi inter-  
17 net. Pada sub bab 2.2.1 sampai sub bab 2.2.3 penulis akan membahas beberapa layanan  
18 Kiri API.

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

- 20 • Masuk ke situs [dev.kiri.travel](http://dev.kiri.travel).

- 21 • Register dengan memasukan alamat email, nama, dan nama perusahaan.

- 22 • Password akan dikirimkan ke alamat email. Tentunya password akan dibuat otomatis  
23 oleh pihak Kiri.

- 24 • Login dengan menggunakan password yang dikirim ke alamat email.

- 25 • Setelah berhasil login, di menu utama pilih *API Keys Managements*.

- 26 • Pilih tombol Add lalu masukan deskripsi penggunaan *API key*.

- 27 • *API key* didapat dan dapat digunakan.

### 28 2.2.1 Web Service Penentuan Rute

29 *Web service* penentuan rute merupakan layanan Kiri API yang digunakan untuk men-  
30 dapatkan langkah perjalanan dari lokasi asal ke lokasi tujuan. Parameter dan keterangan  
31 untuk layanan ini dapat dilihat pada tabel 2.5.

<i>version</i>	2	Memberitahukan bahwa layanan yang dipakai adalah protokol versi 2
<i>mode</i>	"findroute"	Menginstruksikan 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

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

Listing 2.11: Kode kembalian pencarian rute

```

2 {
3     "status": "ok" or "error"
4     "routingresults": [
5         {
6             "steps": [
7                 [
8                     "walk" or "none" or others,
9                     "walk" or vehicle_id or "none",
10                    ["lat_1,lon_1", "lat_2,lon_2", ... "lat_n,lon_n"],
11                    "human readable description, dependant on locale",
12                    URL for ticket booking or null (future)
13                ],
14                [
15                    "walk" or "none" or others,
16                    "walk" or vehicle_id or "none",
17                    ["lat_1,lon_1", "lat_2,lon_2", ... "lat_n,lon_n"],
18                    "human readable description, dependant on locale",
19                    URL for ticket booking or null (future)
20                ]
21            ],
22            "traveltime": any text string, null if and only if route is not found.
23        },
24        {
25            "steps": [ ... ],
26            "traveltime": "..."
27        },
28        {
29            "steps": [ ... ],
30            "traveltime": "..."
31        },
32        ...
33    ]
34 }

```

35 Berikut maksud dari *listing 2.11*:

36 Ketika pencarian rute sukses dilakukan maka status akan memberitahukan "ok" seperti  
37 di baris 2. Selanjutnya setiap langkah dari posisi awal ke posisi tujuan akan ditampung  
38 di elemen *array* untuk menampung langkah. Berikut keterangan dari setiap *array* yang  
39 menampung langkah.

- 40 • Indeks ke 0 atau baris 7 pada *listing 2.11* dapat berisi "walk" atau "none" atau  
41 "others". Baris tersebut berarti jika "walk" untuk berjalan kaki, "none" jika rute  
42 tidak ditemukan dan "others" untuk menggunakan kendaraan.
- 43 • Indeks ke 1 atau baris 8 pada *listing 2.11* merupakan detail dari indeks 0. Artinya jika  
44 indeks 0 menyatakan "walk" berarti indeks 1 harus "walk", "none" berarti indeks 1  
45 harus "none", dan selain itu menyatakan id kendaraan yang mana bisa dipakai untuk  
46 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 menunjukan ikon "from". Biasanya untuk mode presentasi di perangkat bergerak.
    - %toicon = untuk menunjukan 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"

```

```

1      },
2      ...
3  ]
4  "attributions": [
5    "attribution_1", "attribution_2", ...
6  ]
7 }

```

8 Berikut maksud dari *listing 2.12*:

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

12 • "searchresult" (pada baris 4 sampai 7, 8 sampai 11, dan seterusnya) berisi array dari  
13 tempat:

14 – placename: nama tempat  
15 – location: latitude dan longitude dari tempat

16 • "attributions" berisi kumpulan nilai yang berisikan atribut tambahan untuk dimun-  
17 culkan.

### 18 2.2.3 Web Service Menemukan Transportasi Terdekat

19 Merupakan Kiri API yang digunakan untuk menemukan rute transportasi terdekat sesuai  
20 titik yang diinginkan pengguna. Parameter dan keterangan untuk layanan ini dapat dilihat  
21 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

22 Format kembalian layanan menemukan transportasi terdekat dapat dilihat pada *lis-*  
23 *ting 2.13*.

*Listing 2.13: Kode kembalian menemukan lokasi terdekat*

```

24 {
25   "status": "ok" or "error"
26   "nearbytransports": [
27     [
28       "walk" or "none" or others,
29       "walk" or vehicle_id or "none",
30       text string,
31       decimal value
32     ],
33     [
34       "walk" or "none" or others,
35       "walk" or vehicle_id or "none",
36       text string,
37       decimal value
38     ],
39     ...
40   ]
41 }

```

42 Berikut maksud dari *listing 2.13*:

43 Ketika pencarian rute sukses dilakukan maka status akan memberitahukan "ok" seperti di

1 baris 2. Selanjutnya akan diberikan array yang berisi transportasi terdekat yang diurutkan  
2 dari yang terdekat ke yang terjauh. Berikut keterangan dari setiap array tersebut:

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



1

## BAB 3

2

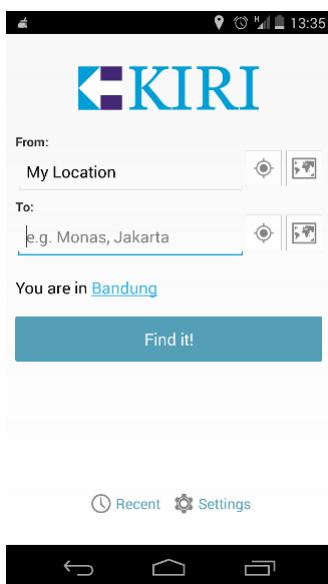
### ANALISIS

- 3 Pada bab ini akan dibahas mengenai analisis aplikasi sejenis, analisis kebutuhan aplikasi,  
4 analisi 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 ha-  
10 laman awal pengguna dapat mengetikan lokasi awal dan tujuan. Selain dengan mengetik  
11 pengguna juga dapat menunjuk lokasi pada peta. Setelah lokasi dipilih sistem akan me-  
12 mastikan dengan memberi daftar nama jalan dan tempat terkait. Jika sudah memilih maka  
13 sistem akan mengeluarkan hasil pencarian rute.

14 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  
16 pengguna dapat memasukan lokasi asal dan lokasi tujuan. Cara memasukan lokasi ada 2  
17 macam yaitu dengan mengetik dan menunjuk pada peta dengan mengetuk tombol peta.

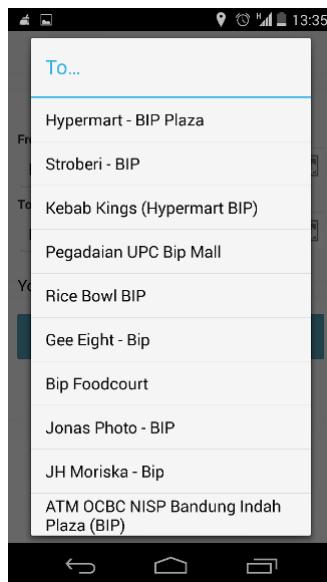
<sup>1</sup><https://play.google.com/store/apps/details?id=travel.kiri.smarttransportapp>

- 1 Bila pengguna ingin menunjuk lokasi pengguna berada dapat dilakukan dengan mengetuk
- 2 tombol kordinat. Tersedia juga pelihan kota yang dapat dipilih oleh pengguna.



Gambar 3.2: Menunjuk lokasi pada peta

- 3 Gambar 3.2 jika pengguna sudah mengetahui lokasi namun tidak tahu nama lokasi.
- 4 Pada halaman ini pengguna diarahkan untuk menemukan lokasi pada peta dan mengetuk lokasi tersebut untuk memilihnya.



Gambar 3.3: Memberikan daftar nama tempat dan nama jalan terkait

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

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

- 12 Pada gambar 3.5 menampilkan rute kendaraan umum dan jalur yang harus dilalui pada



Gambar 3.4: Tampilan rute kendaraan umum dalam bentuk daftar



Gambar 3.5: Tampilan rute kendaraan umum di peta

- 1 peta. Dengan cara ini pengguna dapat mengetahui posisi dan jalur yang harus dilalui.

## **3.2 Analisis Aplikasi**

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

### **3.2.1 Kebutuhan Aplikasi**

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

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

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

Aplikasi yang penulis bangun didasarkan pada kebutuhan sebagai berikut.

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

### **3.2.2 Analisis Kontrol yang Dipakai**

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

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

1 Kontrol terhadap teks juga diperlukan untuk aplikasi. Kebutuhan yang diperlukan adalah mengeluarkan potongan teks yang dapat dibaca dan tempat pengguna memasukan teks.  
2 Untuk mengeluarkan teks yang dapat dilihat penulis akan menggunakan *TextBlock*. Te-  
3 xtBlock digunakan untuk menampilkan tulisan "from" dan "to" pada halaman utama apli-  
4 kasi. Untuk masukan pengguna terhadap aplikasi penulis akan menyediakan *TextBox* sebagai  
5 tempat teks. *TextBox* digunakan sebagai masukan untuk lokasi asal dan lokasi tujuan.

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

- 16 • Tombol map pada bagian from  
17 Tombol untuk berpindah dari halaman utama menuju halaman peta. Pada halaman  
18 peta pengguna dapat menunjuk lokasi asal dan kembali lagi ke halaman utama. Saat  
19 kembali ke halaman utama lokasi yang dipilih akan disimpan dan pada *TextBox* bagian  
20 from akan tertulis "lokasi dari peta".
  - 21 • Tombol map pada bagian from  
22 Tombol untuk mencari lokasi pengguna. Setelah tombol ini di tekan maka lokasi  
23 pengguna akan disimpan dan pada bagian *TextBox* bagian from akan tertulis "here".
  - 24 • Tombol map pada bagian to  
25 Tombol untuk berpindah dari halaman utama menuju halaman peta. Pada halaman  
26 peta pengguna dapat menunjuk lokasi tujuan dan kembali lagi ke halaman utama.  
27 Saat kembali ke halaman utama lokasi yang dipilih akan disimpan dan pada *TextBox*  
28 bagian to akan tertulis "lokasi dari peta".
  - 29 • Tombol map pada bagian to  
30 Tombol untuk mencari lokasi pengguna. Setelah tombol ini di tekan maka lokasi  
31 pengguna akan disimpan dan pada bagian *TextBox* bagian to akan tertulis "here".
  - 32 • Tombol find Tombol ini akan mencari rute angkutan umum dan menampilkannya  
33 pada halaman peta.
- 34 Pada aplikasi ini penulis akan menampilkan daftar tempat dan daftar rute angkutan  
35 umum yang dipakai. Bentuk daftar digunakan penulis karena hasil tempat dan rute ang-  
36 kutan umum akan banyak. Bentuk daftar yang dapat dipakai di Windows Phone adalah  
37 menggunakan *ListBox*. *ListBox* akan menampilkan daftar tempat dan daftar rute satu per  
38 satu menurun ke bawah.

39 **3.2.3 Analisis Terhadap Siklus Hidup Aplikasi**

40 Aplikasi pada Windows Phone memiliki siklus hidup yang dijelaskan pada bab 2.1.4.  
41 Pengaturan aplikasi ini diatur sesuai konfigurasi awal sistem operasi Windows Phone. Tetapi

1 pengaturan ini dapat diatur sesuai kebutuhan aplikasi. Karena di aplikasi ini terdapat  
2 keadaan yang berbeda dengan konfigurasi awal sistem operasi Windows Phone maka perlu  
3 dilakukan pengaturan ulang siklus hidup.

4 Saat aplikasi dijalankan, pengguna memasukan lokasi asal dan lokasi tujuan. Setelah  
5 memasukan lokasi pengguna akan mencari rute. Ketika rute berhasil ditemukan aplikasi akan  
6 berada di keadaan Running. Tetapi ada kemungkinan pengguna berpindah aplikasi atau  
7 mematikan layar untuk menghemat daya. Dalam kasus tersebut sistem operasi Windows  
8 Phone akan menganggap aplikasi tidak aktif dan aplikasi akan masuk pada keadaan *dormant*.  
9 Untuk menangani kasus tersebut maka penulis harus menyimpan keadaan dan informasi  
10 saat sebelum aplikasi menjadi tidak aktif. Penanganan yang penulis akan lakukan adalah  
11 menggunakan *method OnNavigatedFrom()*. Dengan *method* tersebut keadaan aplikasi akan  
12 disimpan di memori.

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

20 Ketika aplikasi sudah berada pada keadaan *Dormant* atau *Tombstoned*, pengguna masih  
21 dapat memulihkan keadaan aplikasi saat aplikasi berada di keadaan *Running* sebelumnya.  
22 Penanganan yang penulis akan lakukan untuk hal tersebut adalah menggunakan *method*  
23 *OnNavigatedTo()*. Menggunakan *method* tersebut akan memulihkan informasi halaman pada  
24 keadaan *Running* sebelumnya.

#### 25 **3.2.4 Analisis Peta**

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

- 28 • Pemetaan terhadap peta atau *cartographic* dan mode warna
- 29 • Tingkat *zoom*
- 30 • Menampilkan gambar dan keterangan angkutan umum menggunakan *pushpin*
- 31 • Menggambar rute pada peta menggunakan *polyline*

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

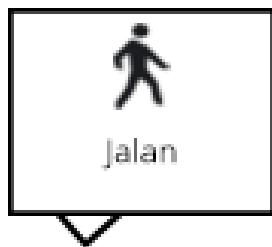
39 Tampilan awal peta di Windows Phone akan mengeluarkan peta dengan pandangan du-  
40 nia. Karena aplikasi pencarian rute ini masih terbatas di Pulau Jawa, Indonesia terutama

1 Jawa Barat maka tingkat zoom harus diatur agar mengikuti lokasi pengguna dan di satu  
 2 daerah saja. Jika pengguna berada di daerah Bandung maka tingkat zoom pada peta di-  
 3 sesuaikan pada daerah tersebut. Tingkat zoom dapat dapat diatur dari kode dan XAML.  
 4 Tingkat zoom yang penulis akan gunakan adalah 14. Tingkat zoom 14 akan menampilkan  
 5 satu kota dengan jelas.

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



Gambar 3.6: Tampilan *pushpin* untuk angkot



Gambar 3.7: Tampilan *pushpin* untuk jalan kaki

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

### 1 3.2.5 Analisis Pemanfaatan Sumber Data

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

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

### 11 3.2.6 Analisis Kiri API

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

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

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

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

- 27 • *version*: 2

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

- 30 • *mode*: "searchplace"

31 Mode "searchplace" digunakan untuk mencari lokasi terkait.

- 32 • *region*: "cgk" untuk Jakarta, "bdo" untuk Bandung, dan "sub" untuk Surabaya

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

- 41 • *querystring*: merupakan kata kunci lokasi

1       • *apikey*: 16 digit heksadesimal

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

4

5 Penulis mencoba mencari lokasi bip dari kata kunci "bip" yang berada di bandung. La-

6 yan dan dikirimkan ke URL [kiri.travel/handle.php](http://kiri.travel/handle.php). Berikut format layanan yang penulis

7 kirim:

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

10

11 Berikut hasil kembalian dari Kiri API:

Listing 3.1: Kode kembalian dari pencarian rute

```

12 {
13     "status ":"ok",
14     "searchresult ":[
15         {
16             "placename ":"Hypermart - BIP Plaza",
17             "location ":"-6.90864,107.61108"
18         },
19         {
20             "placename ":"Stroberi - BIP",
21             "location ":"-6.90834,107.61115"
22         },
23         {
24             "placename ":"Kebab Kings (Hypermart BIP)",
25             "location ":"-6.91503,107.61017"
26         },
27         {
28             "placename ":"Pegadaian UPC Bip Mall",
29             "location ":"-6.90916,107.61052"
30         },
31         {
32             "placename ":"Rice Bowl BIP",
33             "location ":"-6.90873,107.61088"
34         },
35         {
36             "placename ":"Gee Eight - Bip",
37             "location ":"-6.90817,107.61080"
38         },
39         {
40             "placename ":"Jonas Photo - BIP",
41             "location ":"-6.91066,107.61016"
42         },
43         {
44             "placename ":"Bip Foodcourt",
45             "location ":"-6.91081,107.61015"
46         },
47         {
48             "placename ":"Mister Baso BIP",
49             "location ":"-6.90348,107.61709"
50         },
51         {
52             "placename ":"JH Moriska - Bip",
53             "location ":"-6.90868,107.61070"
54         }
55     ],
56     "attributions":null
57 }
```

58 Hasil dari kembalian berupa kumpulan *placename* dan *location*. Hasil tersebut akan

59 aplikasi tampung namun yang akan ditampilkan ke pengguna hanya *placename*. Menam-

60 pilkan *location* tidak efektif menurut penulis karena akan membingungkan pengguna. Dari

61 percobaan yang penulis lakukan, nilai dari *attributions* selalu bernilai "null". Karena hal

62 tersebut maka nilai *attributions* akan penulis abaikan.

63 Pemanfaatan layanan penentuan rute untuk mendapatkan langkah yang harus ditempuh

64 pengguna untuk mencapai lokasi tujuan dari lokasi asal. Pemanfaatan layanan ini yaitu

65 dengan parameter *GET* melalui protokol HTTP. Berikut parameter yang harus dikirim:

- 1     ● *version*: 2

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

- 4     ● *mode*: "findroute"

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

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

8       Karena aplikasi ini memungkinkan dipakai orang banyak maka penulis putuskan untuk  
9       menggunakan bahasa Inggris.

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

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

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

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

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

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

- 19    ● *apikey*: 16 digit heksadesimal.

20      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)

21      Penulis mencoba menuju jalan merdeka dari jalan ciumbuleuit. Layanan dikirimkan ke  
22      URL <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>.

23

24      Berikut hasil kembalian dari Kiri API:

Listing 3.2: Kode kembalian pencarian rute

```

29 {
30   "status ":"ok",
31   "routingresults ":[
32     {
33       "steps ":[
34         [
35           "walk",
36           "walk",
37           ["-6.8747337,107.6048829","-6.87445,107.60465"],
38           "Walk about 41 meter from your starting point \%fromicon to Jalan Ciumbuleuit
39           \%toicon.",
40           null
41         ],
42         [
43           "angkot",
44           "ciumbuleuitsthallurus",
45           ["-6.87445,107.60465","-6.87541,107.60443","-6.87637,107.60421","-6.87734,107.60400",
46           "-6.87830,107.60378",
47             "-6.87926,107.60356","-6.87926,107.60356","-6.87963,107.60352",
48             "-6.87978,107.60352","-6.88093,107.60392","-6.88209,107.60433","-6.88209,107.60433",
49           50
50

```

```

1      " -6.88328,107.60490", " -6.88328,107.60490", " -6.88347,107.60481", " -6.88452,107.60459",
2      " -6.88556,107.60436", " -6.88660,107.60413", " -6.88764,107.60390", " -6.88764,107.60391",
3      " -6.88782,107.60392", " -6.88887,107.60404", " -6.88991,107.60416", " -6.88991,107.60416",
4      " -6.89161,107.60428", " -6.89161,107.60428", " -6.89166,107.60421", " -6.89275,107.60424",
5      " -6.89275,107.60424", " -6.89405,107.60408", " -6.89405,107.60408", " -6.89496,107.60400"],
6
7      "Take angkot Ciumbuleuit – St. Hall (lurus) at Jalan Ciumbuleuit \%fromicon,
8      and alight at Jalan Cihampelas
9      \%toicon about 2.3 kilometer later.", ,
10     null
11   ],
12   [
13     "angkot",
14     "kalapaledeng",
15     [" -6.89501,107.60403", " -6.89562,107.60398", " -6.89623,107.60395", " -6.89732,107.60401",
16       " -6.89732,107.60401", " -6.89882,107.60414", " -6.89882,107.60414", " -6.89969,107.60418",
17       " -6.90071,107.60426", " -6.90173,107.60433", " -6.90173,107.60433", " -6.90297,107.60437",
18       " -6.90420,107.60440", " -6.90420,107.60440", " -6.90426,107.60456", " -6.90422,107.60481",
19       " -6.90399,107.60546", " -6.90406,107.60617", " -6.90454,107.60697", " -6.90454,107.60697",
20       " -6.90512,107.60745", " -6.90618,107.60778", " -6.90618,107.60778", " -6.90643,107.60787",
21       " -6.90651,107.60807", " -6.90675,107.60914", " -6.90675,107.60914", " -6.90694,107.60939",
22       " -6.90723,107.60939", " -6.90891,107.60943", " -6.90891,107.60943", " -6.90909,107.60934",
23       " -6.90914,107.60857", " -6.90933,107.60846", " -6.91021,107.60887", " -6.91021,107.60887",
24       " -6.91030,107.60897", " -6.91028,107.60927", " -6.90986,107.61040", " -6.90986,107.61040"],
25
26     "Take angkot Kalapa – Ledeng at Jalan Cihampelas \%fromicon, and alight at
27     Jalan Aceh
28     \%toicon about 2.3 kilometer later.", ,
29     null
30   ],
31   [
32     "walk",
33     "walk",
34     [" -6.90986,107.61040", " -6.9114646,107.6104887"],
35     "Walk about 178 meter from Jalan Aceh \%fromicon to your destination \%toicon
36     .",
37     null
38   ],
39   ],
40   [
41     "traveltime": "30 minutes"
42   }
43 ]
44 ]
45 ]
46 ]
47 ]
48 ]
49 ]
50 ]
51 ]
52 ]
53 ]
54 ]
55 ]
56 ]
}

```

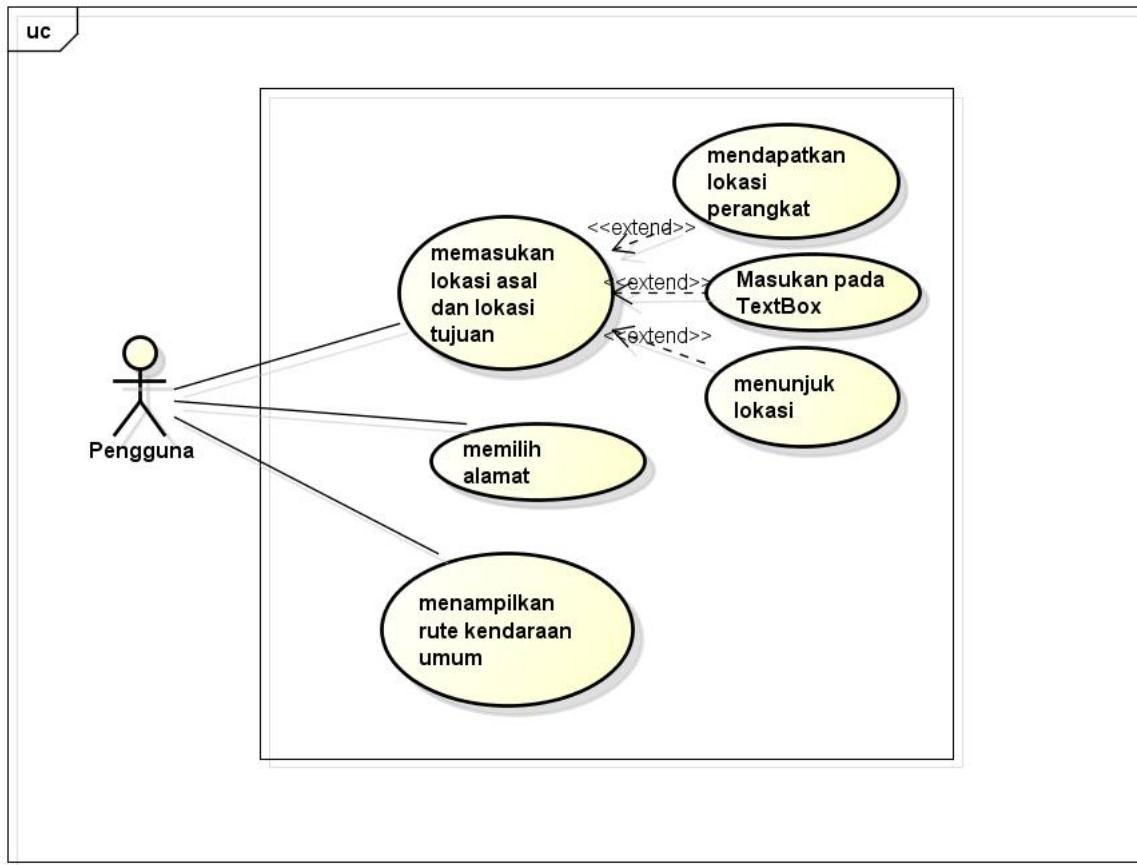
Setiap langkah akan aplikasi tampung dalam elemen *array*. Untuk keterangan dan jenis angkutan umum akan aplikasi tampilkan dalam bentuk *pushpin* pada peta atau daftar. Sedangkan untuk titik-titik kordinat akan digambarkan pada peta. Dari analisa penulis setiap langkah menunjukkan perpindahan angkutan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

- 1     ● Mendapatkan lokasi pengguna berada.
- 2     ● Memasukan lokasi asal dan lokasi tujuan.
- 3     ● Menunjuk langsung lokasi asal dan tujuan pada peta.
- 4     ● Memilih alamat atau tempat dari pilihan yang disediakan.
- 5     ● Menampilkan rute kendaraan umum dalam bentuk titik dan *pushpin* pada peta atau bentuk daftar dari tempat asal ke tempat tujuan.
- 6

- 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

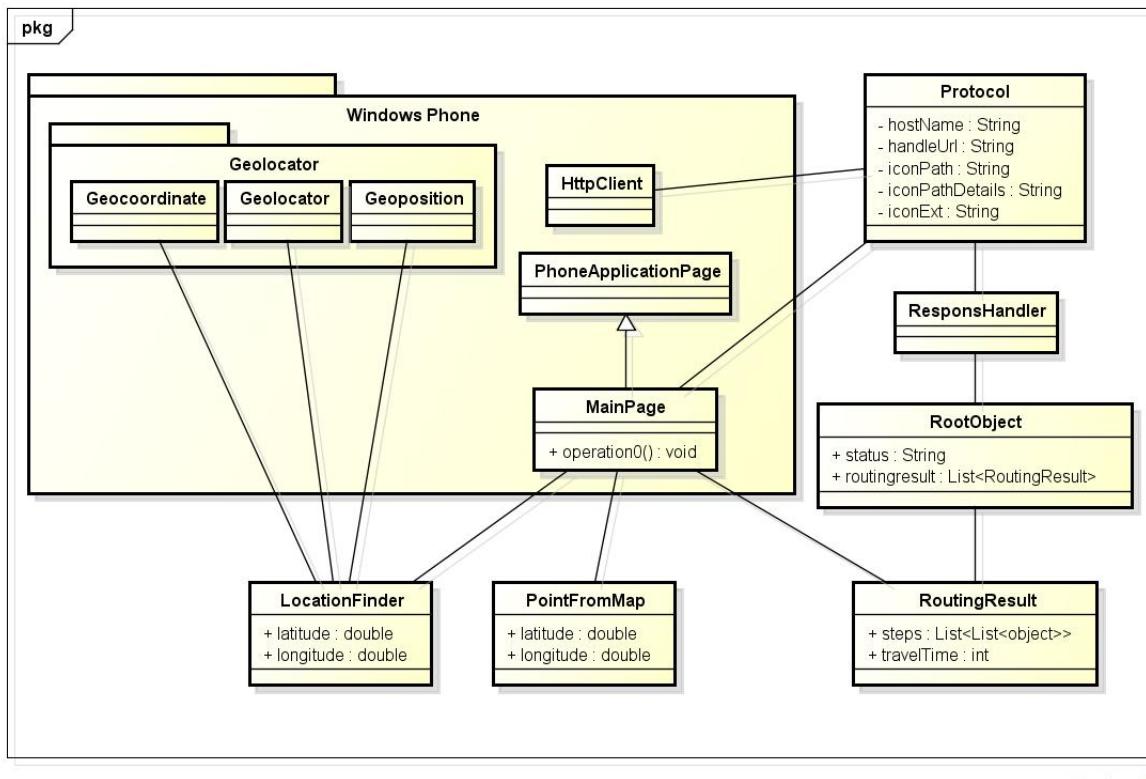
Tabel 3.4: Skenario memilih alamat

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.

1     ● Kelas *RootObject*

2       Merupakan kelas untuk menampung status dan daftar dari layanan *routing* Kiri API.  
3       Hasil kembalian akan dipisahkan di kelas ini untuk selanjutnya ditambahkan di kelas  
4       *RoutingResult*.

5     ● Kelas *RoutingResult*

6       Merupakan kelas untuk menampung setiap langkah dari rute sesuai masukan pengguna.  
7       Pada kelas ini juga rute akan digambarkan pada peta.

8     ● Kelas *PointFromMap*

9       Merupakan kelas yang dapat mengetahui lokasi yang ditunjuk pengguna pada peta.  
10      Kelas ini akan menyimpan lokasi yang ditunjuk pengguna dalam bentuk *latitude* dan  
11      *longitude*.

12    ● Kelas *LocationFinder*

13      Merupakan kelas yang digunakan untuk mencari lokasi. kelas ini akan memanfaatkan  
14      kelas *Geocoordinate* untuk mendapatkan lokasi. Setelah lokasi didapatkan dalam  
15      bentuk kelas *Geoposition* maka akan diubah ke *latitude* dan *longitude*.

1

## BAB 4

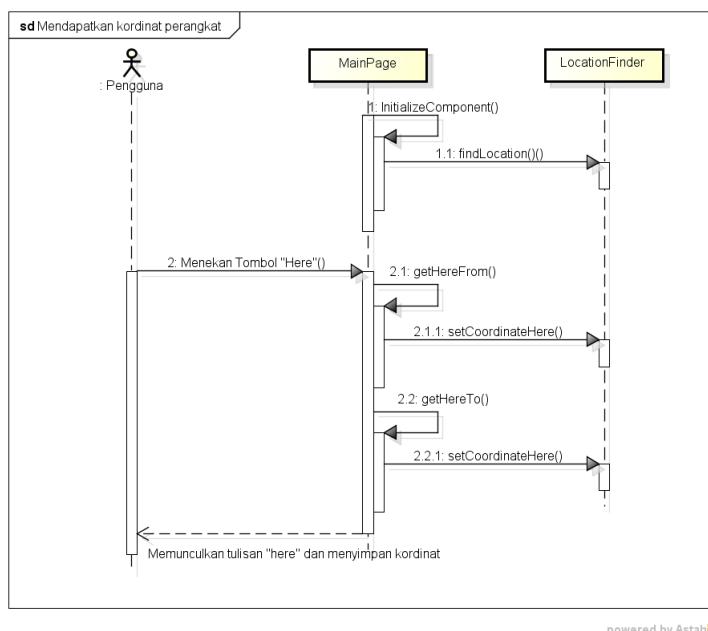
2

### 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**

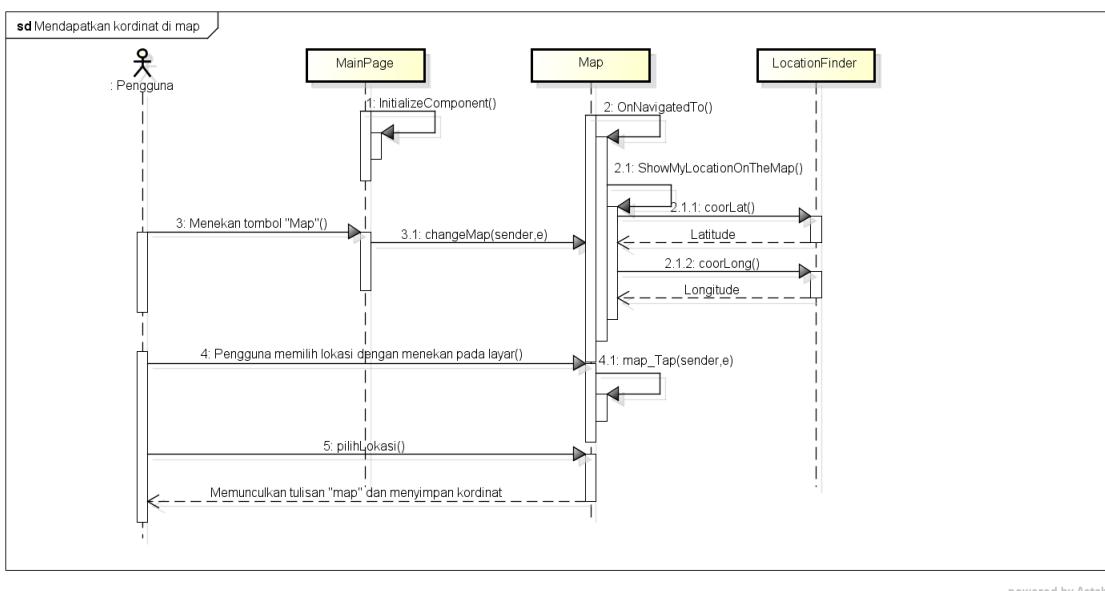
- 6 Diagram *sequence* merupakan diagram yang menggambarkan interaksi antar objek dalam  
7 suatu skenario. Gambar diagram *sequence* dapat dilihat pada gambar reffig:sequence lokasi  
8 perangkat sampai reffig:sequence route.



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 LocationFinder. Setelah lokasi  
15 didapatkan maka akan muncul tulisan "here" pada masukan di kelas MainPage.

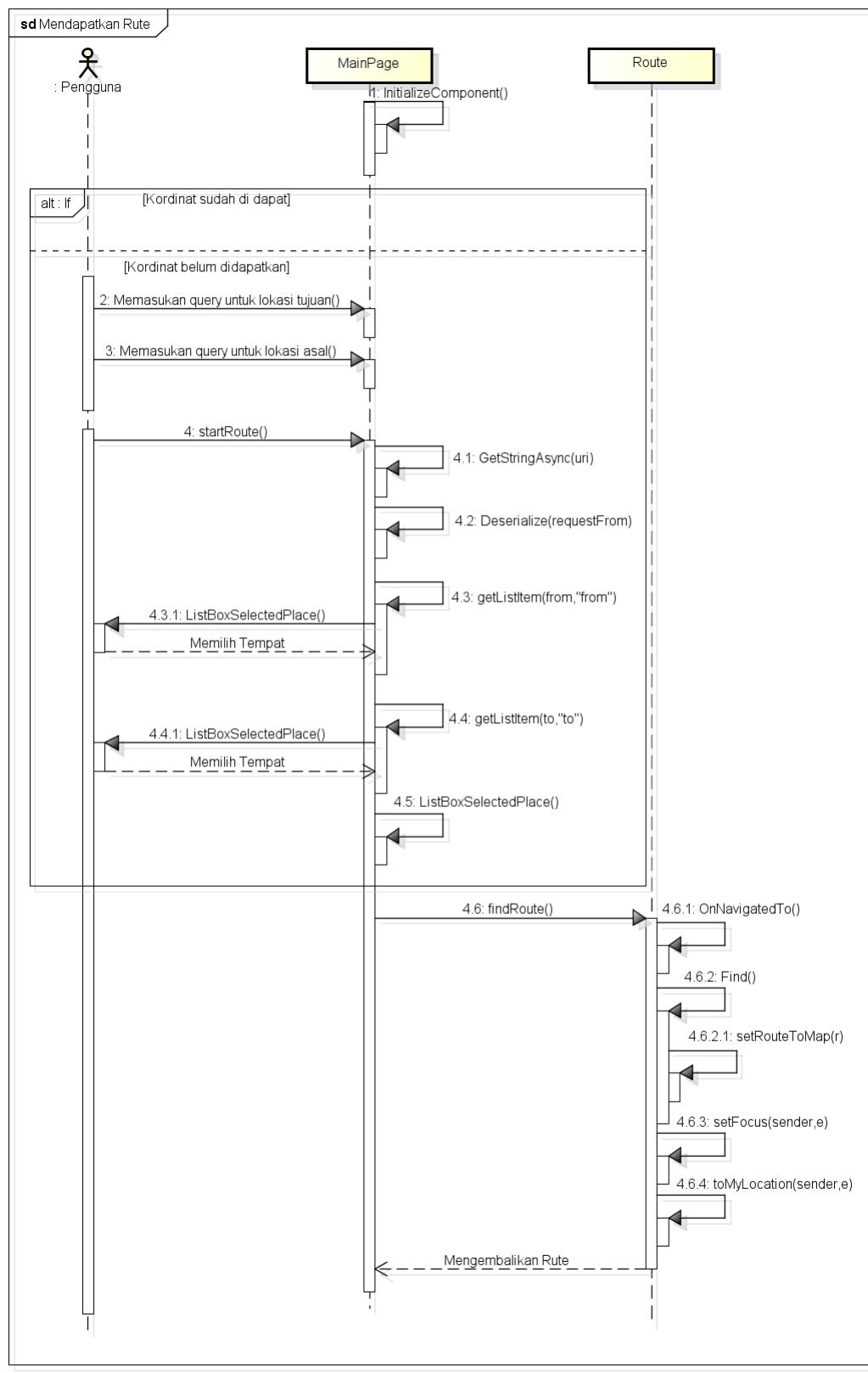
16 Diagram 4.2 merupakan diagram *sequence* untuk memilih lokasi. Diagram menun-  
17 jukan bahwa setelah aplikasi dibuka maka pengguna dapat menekan tombol "map". Setelah



Gambar 4.2: Diagram *sequence* Mendapatkan Kordinat pada Peta

- 1 tombol "map" ditekan maka halaman akan dialihkan ke kelas Map. Saat kelas Map terbuka
- 2 maka lokasi yang ditunjukan adalah lokasi dimana perangkat berada. Untuk mengetahui
- 3 lokasi kelas Map mengambil kordinat *latitude* dan *longitude* dari kelas LocationFinder. Di
- 4 kelas "Map" pengguna dapat memilih lokasi dengan memilih lokasi pada peta dan memang-
- 5 gil *method map\_tap*, lalu setelah pengguna memilih tempat pengguna akan menekan tombol
- 6 Pilih Lokasi yang akan memanggil *method pilihLokasi*. Lokasi yang dipilih pengguna akan
- 7 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 kembaliannya 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 didapat 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.

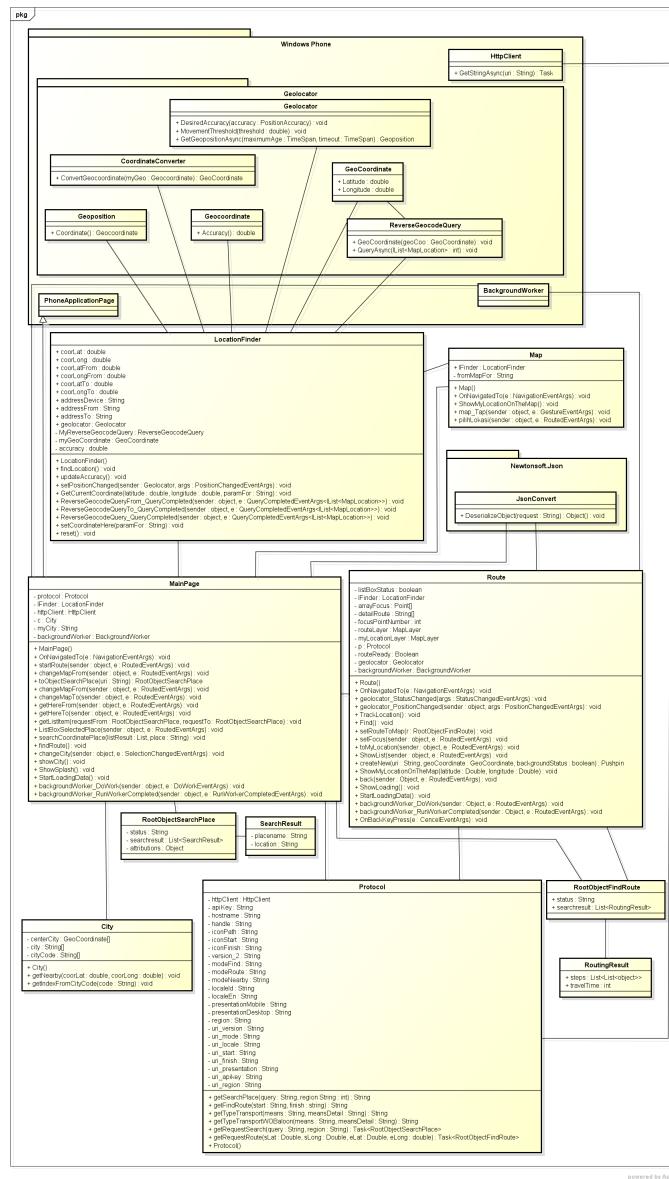


powered by Astah

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 pada aplikasi ini, penulis menyajikan gambar diagram kelas yang dapat dilihat pada gambar 4.4.



Gambar 4.4: Diagram Kelas

### 4.2.1 Kelas *PhoneApplicationPage*

- PhoneApplicationPage* merupakan kelas bawaan Windows Phone yang menangani interaksi pengguna dengan aplikasi dan siklus hidup aplikasi.

### 4.2.2 Kelas *MainPage*

- MainPage* merupakan kelas turunan dari kelas *PhoneApplicationPage* yang menangani interaksi langsung antara halaman aplikasi dengan pengguna. Pada kelas ini akan ditaruh

- 1 kontrol yang diperlukan. Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:
- 2     1. protocol bertipe Protocol untuk mendapatkan URL yang digunakan dalam permintaan  
3       ke Kiri API.
- 4     2. IFinder bertipe LocationFinder digunakan untuk menampung semua informasi meng-  
5       enai lokasi.
- 6     3. httpClient bertipe HttpClient merupakan objek yang akan mengurus permintaan dan  
7       kembalian dari Kiri API.
- 8     4. city bertipe kumpulan String untuk menampung kota yang didukung oleh layanan  
9       Kiri.
- 10    5. myCity bertipe String untuk menampung kode kota sesuai Kode Penerbangan IATA.
- 11    6. backgroundWorker bertipe BackgroundWorker untuk mengurus pencarian lokasi di  
12       belakang layar.

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

- 14    1. Konstruktor MainPage digunakan untuk memuat komponen yang ada di halaman  
15       MainPage dan mendapatkan kota terdekat dari lokasi perangkat.
- 16    2. *method* OnNavigatedTo digunakan untuk mendapatkan lokasi dari peta dan menda-  
17       patkan objek LocationFinder. *method* ini memiliki parameter NavigationEventArgs.
- 18    3. *method* OnNavigatedFrom digunakan untuk mengirimkan *state* objek LocationFinder  
19       saat berpindah kelas. *method* ini memiliki parameter NavigationEventArgs.
- 20    4. *method* startRoute digunakan untuk mendapatkan masukan pengguna. Jika masukan  
21       didapat dari peta atau lokasi perangkat berarti sudah dalam kordinat, namun jika  
22       masukan didapat dari *query* maka akan dicari lokasi yang terkait sesuai *query* tersebut.  
23       Lokasi terkait yang didapatkan akan dikembalikan ke pengguna untuk dipilih. Setelah  
24       pengguna lokasi dalam bentuk kordinat didapatkan maka kelas akan diarahkan ke kelas  
25       Route. *method* ini memiliki parameter objek tombol dan event.
- 26    5. *method* changeMapFrom digunakan untuk berpindah ke halaman mapFrom. *method*  
27       ini memiliki parameter objek tombol dan event.
- 28    6. *method* changeMapTo digunakan untuk berpindah ke halaman mapTo. *method* ini  
29       memiliki parameter objek tombol dan event.
- 30    7. *method* getHereFrom digunakan untuk mendapatkan kordinat perangkat lalu menyimpan  
31       nilainya di kordinat asal di kelas LocationFinder dan menulisakan "Here" pada  
32       TextBox lokasi asal. *method* ini memiliki parameter objek tombol dan event.
- 33    8. *method* getHereTo digunakan untuk mendapatkan kordinat perangkat lalu menyimpan  
34       nilainya di kordinat tujuan di kelas LocationFinder dan menulisakan "Here" pada  
35       TextBox lokasi tujuan. *method* ini memiliki parameter objek tombol dan event.

- 1        9. *method* getListItem digunakan untuk membuat listBox lalu menampilkan ke pengguna.  
2        *method* ini memiliki parameter RootObjectSearchPlace dan string yang menunjukan  
3        list yang ditampilkan untuk lokasi asal dan lokasi tujuan.
- 4        10. *method* ListBoxSelectedPlace digunakan untuk mendapatkan tempat asal yang dipilih  
5        pengguna. *method* ini memiliki parameter objek dan *event SelectionChangedEventArgs*.  
6        *gs*.
- 7        11. *method* searchCoordinatePlace digunakan untuk mencari kordinat dari tempat pilihan  
8        pengguna. *method* ini memiliki parameter ListBox dan tempat yang dipilih dalam  
9        bentuk string.
- 10      12. *method* findRoute digunakan untuk berpindah ke kelas Route jika lokasi asal dan lokasi  
11      tujuan sudah ditentukan.
- 12      13. *method* changeCity digunakan untuk mengubah kota tujuan dari pencarian. *method*  
13      ini memiliki parameter objek dan *event SelectionChangedEventArgs*.
- 14      14. *method* showCity digunakan untuk mencari kota yang paling dekat dengan lokasi per-  
15      angkat.
- 16      15. *method* ShowSplash digunakan untuk menampilkan tampilan awal untuk proses inisi-  
17      alisasi aplikasi.
- 18      16. *method* StartLoadingData digunakan untuk memanggil BackgroundWorker. Backgro-  
19      undWorker digunakan untuk melakukan aksi di belakang layar.
- 20      17. *method* backgroundWorker\_DoWork digunakan untuk melakukan pemanggilan aksi  
21      di belakang layar. *method* ini memiliki parameter objek dan DoWorkEventArgs.
- 22      18. *method* backgroundWorker\_RunWorkerCompleted digunakan untuk melakukan pe-  
23      manggilan saat BackgroundWorker selesai melakukan tugasnya. *method* ini memiliki  
24      parameter objek dan RunWorkerCompletedEventArgs.

#### 25      4.2.3 Kelas City

26      *City* merupakan kelas yang menyimpan kota-kota yang mendukung pencarian rute ken-  
27      daraam umum dengan bantuan Kiri. Berikut adalah penjelasan atribut-atribut yang dimiliki  
28      kelas ini:

- 29        1. centerCity bertipe *array of GeoCoordinate* untuk menyimpan kordinat pusat dari kota.
- 30        2. city bertipe *array of String* untuk menyimpan nama kota.
- 31        3. cityCode bertipe *array of String* untuk menyimpan kode kota dalam huruf kecil sesuai  
32        aturan IATA Airport Code.

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

- 34        1. Konstruktor City digunakan untuk untuk inisialisasi nilai atribut.

- 1     2. *method* getNearby digunakan untuk mencari kota terdekat dengan lokasi perangkat.
- 2     *method* ini mengembalikan interger yang merupakan index kota pada atribut city.
- 3     *method* ini memiliki 2 buah parameter yaitu *latitude* dan *longitude* yang bertipe double.
- 4     3. *method* getIndexFromCityCode digunakan untuk mencari index pada *array* sesuai kode kota. *method* ini memiliki parameter bertipe String yang merupakan kode kota.

#### 6 4.2.4 Kelas *BackgroundWorker*

7     *BackgroundWorker* merupakan kelas yang dipakai untuk mengeksekusi operasi pada  
8 *thread* terpisah. Berikut adalah penjelasan *event* yang dimiliki kelas ini dan dipakai untuk  
9 perancangan aplikasi:

- 10    1. *Event* DoWork
- 11    2. *Event* RunWorkerCompleted

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

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

#### 14 4.2.5 Kelas *Geocoordinate*

15     *Geocoordinate* merupakan kelas bawaan dari Windows Phone yang akan dimanfaatkan  
16 untuk membaca *latitude* dan *longitude*.

#### 17 4.2.6 Kelas *Geolocator*

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

#### 21 4.2.7 Kelas *Geoposition*

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

#### 23 4.2.8 Kelas *LocationFinder*

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

- 26    1. coorLat bertipe Double untuk menampung kordinat latitude pengguna.
- 27    2. coorLong bertipe Double untuk menampung kordinat longitude pengguna.
- 28    3. coorLatFrom bertipe Double untuk menampung kordinat latitude lokasi asal yang  
29       diinginkan pengguna.
- 30    4. coorLongFrom bertipe Double untuk menampung kordinat longitude lokasi asal yang  
31       diinginkan pengguna.
- 32    5. coorLatTo bertipe Double untuk menampung kordinat latitude lokasi tujuan yang  
33       diinginkan pengguna.

- 1       6. coorLongTo bertipe Double untuk menampung kordinat longitude lokasi tujuan yang  
2       diinginkan pengguna.
- 3       7. addressDevice bertipe String untuk menyimpan alamat perangkat berada.
- 4       8. addressDeviceFrom bertipe String untuk menyimpan alamat berdasarkan lokasi lokasi  
5       asal yang diinginkan pengguna.
- 6       9. addressDeviceTo bertipe String untuk menyimpan alamat berdasarkan lokasi tujuan  
7       yang diinginkan pengguna.
- 8       10. geolocator bertipe Geolocator untuk menampung pengaturan mendapatkan lokasi.
- 9       11. myReverseGeocodeQuery bertipe ReverseGeocodeQuery untuk konversi dari alamat  
10      ke lokasi dan sebaliknya.
- 11      12. myCoordinate bertipe GeoCoordinate untuk menampung kordinat geografis.
- 12      13. accuracy bertipe double untuk menampung akurasi perangkat mendapatkan lokasi.

- 13      Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:
- 14      1. Konstruktor LocationFinder berfungsi mengatur atribut geolocator dan mencari lokasi  
15      perangkat.
  - 16      2. *method* findLocation berfungsi inisialisasi GPS lalu mendapat kordinat dan menam-  
17      pungnya di atribut.
  - 18      3. *method* updateAccuracy berfungsi untuk mengubah nilai akurasi dari perangkat.
  - 19      4. *method* setPositionChanged berfungsi mengubah atribut coorLat, atribut coorLong,  
20      dan akurasi jika terdapat perubahan lokasi. *method* ini memiliki parameter Geolo-  
21      cator dan PositionChangedEventArgs yang akan menjalankan *method* jika terdapat  
22      perubahan yang diberitahukan melalui kelas Geolocator.
  - 23      5. *method* GetCurrentCoordinate berfungsi mengubah posisi saat ini, posisi lokasi asal,  
24      dan lokasi tujuan. *method* ini memiliki tiga buah parameter *latitude* bertipe Double,  
25      *longitude* bertipe Double, dan paramFor bertipe String. Parameter *latitude* dan *lo-  
26      ngitude* merupakan lokasi sedangkan parameter paramFor digunakan sebagai tujuan  
27      perubahan lokasi.
  - 28      6. *method* ReverseGeocodeQueryFrom\_QueryCompleted berfungsi untuk mencari ala-  
29      mat lokasi asal. *method* ini memiliki parameter objek dan QueryCompletedEventArgs<IList<MapLocation>>.
  - 30      7. *method* ReverseGeocodeQueryTo\_QueryCompleted berfungsi untuk mencari alamat  
31      lokasi tujuan. *method* ini memiliki parameter objek dan QueryCompletedEventArgs<IList<MapLocation>>.
  - 32      8. *method* ReverseGeocodeQuery\_QueryCompleted berfungsi untuk mencari alamat lo-  
33      kasi perangkat. *method* ini memiliki parameter objek dan QueryCompletedEventArgs<IList<MapLocation>>.

- 1     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.
- 5     10. *method* reset berfungsi untuk memasang kembali lokasi asal dan lokasi tujuan.

#### 6   **4.2.9   Kelas Map**

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

- 10    1. IFinder bertipe LocationFinder digunakan untuk menampung semua informasi mengenai lokasi.
- 12    2. fromMapFor bertipe string digunakan sebagai indikator lokasi asal atau lokasi tujuan yang didapatkan dari map.

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

- 15    1. Konstruktor Map untuk inisialisasi dan penambahan *event* mengetuk pada peta.
- 16    2. *method* OnNavigatedTo berfungsi untuk mendapatkan masukan lokasi asal atau lokasi tujuan yang akan ditentukan dari map untuk kemudian ditampung di Objek LocationFinder. *method* ini memiliki sebuah parameter NavigationEventArgs.
- 19    3. *method* OnNavigatedFrom digunakan untuk mengirimkan *state* objek LocationFinder saat berpindah kelas. *method* ini memiliki parameter NavigationEventArgs.
- 21    4. *method* ShowMyLocationOnTheMap digunakan untuk memberitahu dan menandai lokasi perangkat.
- 23    5. *method* map\_Tap berfungsi untuk menandai lokasi yang ditunjuk pengguna lalu menerjemahkan lokasi yang ditunjuk pengguna pada peta dan mengirimnya ke kelas LocationFinder.
- 26    6. *method* pilihLokasi berfungsi berpindah ke kelas MainPage dan memberitahu kelas MainPage bahwa lokasi sudah dipilih. *method* ini memiliki parameter objek tombol dan event.

#### 29   **4.2.10   Kelas HttpClient**

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

- 33    1. *method* GetStringAsync membutuhkan parameter alamat bertipe string dan mengembalikan kembalian dari Kiri dalam bentuk Task<string>.

#### **4.2.11 Kelas *JsonConvert***

*JsonConvert* merupakan kelas yang menyediakan *method* untuk mengkonversi berbagai jenis komponen *common language runtime* dan *JSON*. Kelas ini merupakan bagian *namespace* *Newtonsoft*. Berikut adalah penjelasan *method* yang dipakai untuk perancangan aplikasi:

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

#### **4.2.12 Kelas *Protocol***

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

1. *uri\_version* bertipe string digunakan untuk menyimpan nama dari parameter *uri*.
2. *uri\_mode* bertipe string digunakan untuk menyimpan nama dari parameter *mode*.
3. *uri\_locale* bertipe string digunakan untuk menyimpan nama dari parameter *locale*.
4. *uri\_start* bertipe string digunakan untuk menyimpan nama dari parameter *start*.
5. *uri\_finish* bertipe string digunakan untuk menyimpan nama dari parameter *finish*.
6. *uri\_presentation* bertipe string digunakan untuk menyimpan nama dari parameter *presentation*.
7. *uri\_apikey* bertipe string digunakan untuk menyimpan nama dari parameter *apikey*.
8. *uri\_region* bertipe string digunakan untuk menyimpan nama dari parameter *region*.
9. *uri\_query* bertipe string digunakan untuk menyimpan nama dari parameter *query*.
10. *apiKey* bertipe string digunakan untuk menyimpan nilai kunci API untuk mengirim permintaan ke Kiri.
11. *hostname* bertipe string digunakan untuk digunakan untuk menyimpan alamat host dari Kiri.
12. *handle* bertipe string digunakan untuk menyimpan alamat host ditambah "handle.php".
13. *iconPath* bertipe string digunakan untuk menyimpan lokasi gambar yang dibutuhkan.
14. *iconStart* bertipe string digunakan untuk menyimpan lokasi gambar awal perjalanan dari lokasi awal.
15. *iconFinish* bertipe string digunakan untuk menyimpan lokasi gambar akhir perjalanan ke lokasi tujuan.
16. *version\_2* bertipe string digunakan untuk menyimpan nilai versi dari API yang digunakan (saat pembuatan penelitian ini versi Kiri API yang digunakan adalah versi 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.

1 **4.2.13 Kelas *RootObjectSearchPlace***

2 *RootObjectSearchPlace* merupakan kelas untuk menampung objek hasil pencarian lokasi.

3 Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 4 1. status bertipe *string* digunakan untuk menampung hasil kembalian status dari Kiri.
- 5 2. searchresult bertipe *list* dan menampung banyak objek *SearchResult*.
- 6 3. attributions bertipe objek untuk menampung *attributions*.

7 **4.2.14 Kelas *SearchResult***

8 *SearchResult* merupakan kelas untuk menampung nama tempat dan kordinat dari nama

9 tempat tersebut. Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 10 1. placename bertipe *string* digunakan untuk menampung nama tempat.
- 11 2. location bertipe *string* digunakan untuk menampung nama tempat.

12 **4.2.15 Kelas *RootObjectFindRoute***

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

- 15 1. status
- 16 2. routingresults

17 **4.2.16 Kelas *RoutingResult***

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

- 20 1. steps
- 21 2. traveltime

22 **4.2.17 Kelas *Route***

23 *Route* merupakan kelas untuk pencarian rute dan menampilkannya kepada pengguna.

24 Berikut adalah penjelasan atribut-atribut yang dimiliki kelas ini:

- 25 1. listBoxStatus bertipe boolean digunakan untuk status rute dalam bentuk daftar sedang  
26 tertutup atau terbuka.
- 27 2. IFinder bertipe LocationFinder digunakan untuk menampung semua informasi meng-  
enai lokasi.
- 29 3. arrayFocus bertipe *array of Point* digunakan untuk menampung titik fokus perubahan  
30 jenis transportasi yang digunakan pengguna.
- 31 4. detailRoute bertipe *array of String* digunakan untuk menampung keterangan yang  
32 dibutuhkan pengguna dari Kiri API.

- 1       5. focusPointNumber bertipe *integer* digunakan untuk menentukan *index of* dari atribut  
2       arrayFocus dan detailRoute.
  - 3       6. routeLayer bertipe MapLayer digunakan untuk menampung *Polyline* dan *Pushpin*
  - 4       7. myLocationLayer bertipe MapLayer digunakan untuk menampung titik dari lokasi  
5       perangakt berada.
  - 6       8. p bertipe Protocol digunakan untuk menampung permintaan data dengan Kiri API.
  - 7       9. geolocator bertipe Geolocator untuk menangani perubahan lokasi yang terjadi.
  - 8       10. backgroundWorker bertipe BackgroundWorker digunakan untuk menangain proses di  
9       belakang layar.
- 10     Berikut adalah penjelasan beberapa *method* yang dimiliki kelas ini:
- 11     1. Konstruktor Route digunakan untuk memuat komponen yang ada di halaman Route,  
12       inisialisasi atribu, dan pemanggilan backgroundWorker.
  - 13     2. *method* OnNavigatedTo digunakan untuk mendapatkan objek LocationFinder dan me-  
14       manggil *method* TrackLocation. *method* ini memiliki parameter NavigationEventArgs.
  - 15     3. *method* OnNavigatedFrom digunakan untuk mengirimkan *state* objek LocationFinder  
16       saat berpindah kelas. *method* ini memiliki parameter NavigationEventArgs.
  - 17     4. *method* TrackLocation digunakan untuk inisislisasi GeoLocator dan dan memulai pe-  
18       lacakkan lokasi terus menerus.
  - 19     5. *method* geolocator\_StatusChanged digunakan untuk mengetahui status GeoLocator.
  - 20     6. *method* geolocator\_PositionChanged digunakan untuk mengetahui perubahan lokasi  
21       yang terjadi dan menyimpannya di kelas LocationFinder.
  - 22     7. *method* Find digunakan untuk mencari rute yang dicari pengguna.
  - 23     8. *method* setRouteToMap digunakan untuk menggambar rute dan lokasi pada *layer* pe-  
24       ta. *method* ini memiliki parameter RootObjectFindRoute yang merupakan objek un-  
25       tuk pencarian rute.
  - 26     9. *method* setFocus digunakan untuk mengarahkan pusat pandangan ke titik lokasi per-  
27       gantian jenis transportasi. *method* ini memiliki parameter objek dan RoutedEventArgs.
  - 29     10. *method* toMyLocation digunakan untuk mengarahkan pusat pandangan ke titik lokasi  
30       perangakt berada. *method* ini memiliki parameter objek dan RoutedEventArgs.
  - 31     11. *method* ShowList digunakan untuk membuka dan menutup rute dalam bentuk daftar.  
32       *method* ini memiliki parameter objek dan RoutedEventArgs.
  - 33     12. *method* createNew digunakan untuk membuat objek Pushpins. *method* ini memiliki  
34       parameter uri bertipe String, transport bertipe String yang menandakan jenis trans-  
35       portasi, 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.

### 18 4.3 Perancangan Antar Muka

19 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.

#### 23 4.3.1 Antarmuka Kelas MainPage



Gambar 4.5: Antarmuka *MainPage*

1 Antarmuka Kelas Map pada gambar 4.5 merupakan tampilan awal saat aplikasi  
2 dijalankan. Antarmuka Kelas Map memiliki dua buah masukan, lima buah tombol, dan  
3 satu menu daftar. Berikut adalah detailnya.

4 Dua buah masukan yaitu.

- 5 • Masukan lokasi asal

6 Merupakan masukan lokasi asal mula pengguna ingin melakukan perjalanan.

- 7 • Masukan lokasi tujuan

8 Merupakan masukan lokasi tujuan berhentinya perjalanan.

9 Lima buah tombol yaitu.

- 10 • Tombol map untuk lokasi asal

11 Jika tombol ditekan maka akan berpindah ke kelas map untuk memilih lokasi asal  
12 di peta. Jika di kelas Map pengguna memilih lokasi maka pada masukan lokasi asal  
13 terdapat tulisan "Maps".

- 14 • Tombol here untuk lokasi asal

15 Jika tombol ditekan maka lokasi asal adalah lokasi perangkat saat tombol ditekan dan  
16 masukan lokasi asal menjadi "here".

- 17 • Tombol map untuk lokasi tujuan

18 Jika tombol ditekan maka akan berpindah ke kelas map untuk memilih lokasi tujuan  
19 di peta. Jika di kelas Map pengguna memilih lokasi maka pada masukan lokasi tujuan  
20 terdapat tulisan "Maps".

- 21 • Tombol here untuk lokasi tujuan

22 Jika tombol ditekan maka lokasi tujuan adalah lokasi perangkat saat tombol ditekan  
23 dan masukan lokasi tujuan menjadi "here".

- 24 • Tombol find

25 Jika tombol ditekan maka akan menampilkan daftar tempat asal dan tempat tujuan  
26 lalu mengarahkan ke Kelas Route.

27 Satu buah daftar yaitu.

- 28 • Daftar kota yang tersedia

29 Merupakan daftar kota yang tersedia (kota yang rute angkutan umumnya dapat ditemukan  
30 dengan aplikasi ini). Disaat aplikasi dijalankan maka daftar akan menunjuk ke  
31 kota terdekat tempat perangkat berada.

32 Antarmuka Daftar Tempat pada gambar 4.6 merupakan daftar yang akan dimunculkan jika pengguna memasukan kata kunci pada pencarian tempat asal dan tempat tujuan.

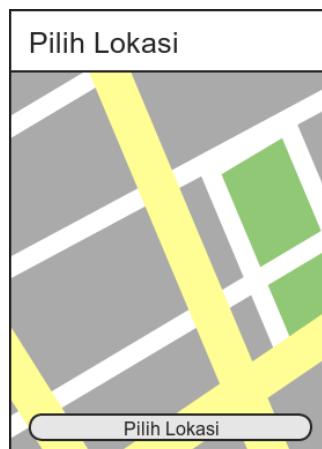
34 Daftar akan muncul jika didapat kembalian hasil pencarian lebih dari satu.

### 35 4.3.2 Antarmuka Kelas Map

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



Gambar 4.6: Antarmuka Daftar Tempat



Gambar 4.7: Antarmuka Map

#### **1 4.3.3 Antarmuka Kelas Route**

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

- 5 • Tombol prev  
6 Jika tombol ditekan maka akan menunjuk titik sebelumnya pada rute peta.
- 7 • Tombol next  
8 Jika tombol ditekan maka akan menunjuk titik setelahnya pada rute peta.
- 9 • Tombol here  
10 Jika tombol ditekan maka akan menunjuk lokasi perangkat berada pada peta.
- 11 • Tombol Show List  
12 Jika tombol ditekan maka akan menunjuk atau menyembunyikan daftar rute.

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



Gambar 4.8: Antarmuka *Route*



Gambar 4.9: Antarmuka Rute dalam bentuk Daftar



## **BAB 5**

### **IMPLEMENTASI DAN PENGUJIAN APLIKASI**

<sup>3</sup> Pada bab 5 akan dibahas implementasi dan pengujian aplikasi pencari rute kendaraan umum  
<sup>4</sup> untuk Windows Phone.

#### **5.1 Implementasi**

<sup>6</sup> Pada sub bab ini akan dijelaskan mengenai lingkungan yang digunakan untuk membangun  
<sup>7</sup> aplikasi Pencari Rute Kendaraan Umum untuk Windows Phone. Pada lingkungan yang akan  
<sup>8</sup> dibahas juga penulis membangun aplikasi sesuai rancangan yang telah dibahas pada bab 4  
<sup>9</sup> dan mengujinya.

##### **5.1.1 Perangkat Keras untuk Implementasi**

<sup>10</sup> Dalam membangun aplikasi ini perangkat keras yang digunakan adalah sebagai berikut:

<sup>11</sup> 1. Komputer

- <sup>12</sup> (a) Processor: intel Core i7-2620M CPU 2,7 GHz
- <sup>13</sup> (b) RAM: 4 GB
- <sup>14</sup> (c) Hardisk: 640 GB
- <sup>15</sup> (d) VGA: Intel HD 3000

<sup>16</sup> 2. Perangkat Bergerak

- <sup>17</sup> (a) Processor: 1,2 GHz
- <sup>18</sup> (b) RAM: 1 GB
- <sup>19</sup> (c) ROM: 8 GB
- <sup>20</sup> (d) Layar: 720 x 1280 pixel, 4,7 inch
- <sup>21</sup> (e) GPS
- <sup>22</sup> (f) Sensor: kompas, *accelerometer*

##### **5.1.2 Perangkat Lunak untuk Implementasi**

<sup>23</sup> Dalam membangun aplikasi ini perangkat lunak yang digunakan adalah sebagai berikut:

<sup>24</sup> 1. Komputer

- <sup>25</sup> (a) Sistem Operasi Windows 8.1

- 1        (b) IDE Visual Studio Express 2012
  - 2        (c) Bahasa Pemrograman C#
  - 3        (d) Library .Net Framework 4.5
- 4        2. Perangkat Bergerak
- 5              (a) Sistem Operasi Windows Phone 8.1

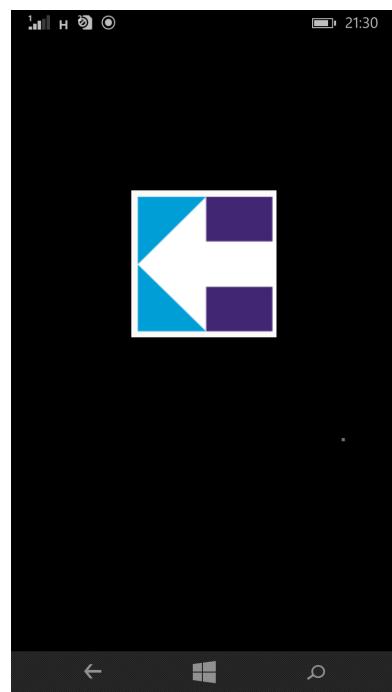
### 6    5.1.3   Hasil Implementasi

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

- 8        1. Kode Program  
9              Kode Program pada perangkat lunak ditulis dengan menggunakan bahasa c#. Bahasa  
10          C# dipilih berdasarkan analisa pada bab 3 dan kemampuan penulis.
- 11          2. Hasil kompilasi program  
12              Hasil dari kompilasi program berupa file Kiri\_Debug\_AnyCPU.xap. File ini dapat  
13              dipasang pada perangkat dengan sistem operasi Windows Phone versi 8 atau lebih  
14          tinggi.
- 15          3. Antarmuka Aplikasi  
16              Berikut merupakan hasil implementasi antarmuka aplikasi Pencari Rute Kendaraan  
17          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



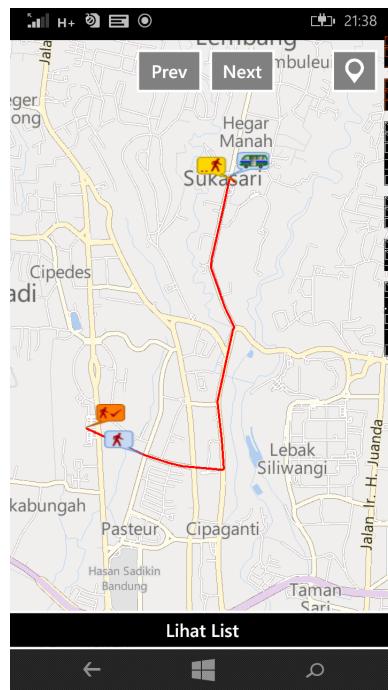
Gambar 5.4: Gambar antarmuka kelas Map



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

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 menggunakan	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](#).

<sup>1</sup> **5.2.3 Pengujian Experimental**

<sup>2</sup> Pengujian eksperimental dilakukan untuk mengetahui keberhasilan aplikasi yang dibuat  
<sup>3</sup> penulis. Berikut pengujian eksperimental yang dilakukan penulis.

<sup>4</sup> 1. Pengujian 1

- <sup>5</sup> • Tempat : Rumah(Jalan Kejaksaan menuju Unpar)  
<sup>6</sup> • Waktu : Pukul 9 pada tanggal 7 April 2014 Pada pengujian 1 penulis memasukan  
<sup>7</sup> alamat di dalam rumah untuk menuju Unpar. Saat penulis memasukan kata  
<sup>8</sup> Unpar muncul daftar tempat sesuai kata kunci "Unpar", lalu penulis memilih  
<sup>9</sup> "Universitas Katolik Parahyangan".

<sup>10</sup> **5.2.4 Perbandingan Aplikasi Pencari Rute di Android dengan Aplikasi  
11 Pencari Rute di Windows Phone**

<sup>12</sup> Aplikasi Pencari rute yang penulis buat dengan yang ada di Android memiliki beberapa  
<sup>13</sup> perbedaan. Berikut perbedaan antara aplikasi pencari rute di Android dengan Windwows  
<sup>14</sup> 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> Bab ini berisi kesimpulan dari pembangunan aplikasi beserta saran untuk pengembangan  
<sup>4</sup> selanjutnya.

#### <sup>5</sup> 6.1 Kesimpulan

<sup>6</sup> Dari hasil penelitian penulis terhadap perancangan aplikasi pencari rute kendaraan  
<sup>7</sup> umum didapat kesimpulan sebagai berikut.

- <sup>8</sup> 1. Pengguna dapat memasukan lokasi berdasarkan peta, lokasi perangkat, dan dengan  
<sup>9</sup> kata kunci.
- <sup>10</sup> 2. Penggunaan Kiri API sudah dapat digunakan untuk mencari rute angkutan umum.
- <sup>11</sup> 3. Akses internet mempengaruhi performansi mendapatkan rute.

#### <sup>12</sup> 6.2 Saran

<sup>13</sup> Berdasarkan hasil kesimpulan yang telah dipaparkan, penulis memberi saran sebagai  
<sup>1</sup> berikut.

- <sup>2</sup> 1. Memanfaatkan tombol "enter" untuk memanggil *method startRoute*. Masukan dari  
<sup>3</sup> pengguna karena setelah pengguna menentukan tujuan tombol "Find" terhalang *ke-  
4* *yboard* maka pengguna harus menekan tombol "back" terlebih dahulu untuk menekan  
<sup>5</sup> tombol "Find".
- <sup>6</sup> 2. Memanfaatkan pencarian lokasi pada peta agar pengguna dapat memilih lokasi dengan  
<sup>7</sup> lebih mudah.
- <sup>8</sup> 3. Menambahkan *gesture* dan animasi yang akan meningkatkan interaksi pengguna ter-  
<sup>9</sup> hadap aplikasi.



## DAFTAR REFERENSI

- [1] "Windows phone silverlight development." <http://msdn.microsoft.com/library/windows/apps/ff402535.aspx>, 2014. Accessed: 2014-8-17.
- [2] P. Manning, *Pro Windows Phone App Development*. Apress, 2013.
- [3] A. Whitechapel and S. McKenna, *Windows Phone 8 Development Internals*. Microsoft, 2013.
- [4] T. Bray, "The JavaScript Object Notation (JSON) Data Interchange Format." RFC 7159 (Proposed Standard), Mar. 2014.
- [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.



## DAFTAR REFERENSI

- 1760 [1] Microsoft *Windows Phone Silverlight development* 2014 : <http://msdn.microsoft.com/library/windows/apps/ff402535.aspx>.
- 1761  
1762 [2] Kiri Team *KIRI API v2 Documentation* 2014 : [https://bitbucket.org/projectkiri/kiri\\_api/wiki/KIRI%20API%20v2%20Documentation](https://bitbucket.org/projectkiri/kiri_api/wiki/KIRI%20API%20v2%20Documentation).
- 1763  
1764 [3] Manning, Paul *Pro Windows Phone App Development* 2013: Apress.
- 1765 [4] Szostak, Tomasz *Windows Phone 8 Application Development Essentials* 2013: PACKT.