

**ANALISIS PERFORMANSI PENERAPAN PENGGUNAAN PETA
OFFLINE PADA APLIKASI *LOCATION BASED SERVICE*
BERBASIS *MOBILE WEBSITE* STUDI KASUS PENCARIAN
LOKASI TAMBAL BAN DI KOTA MALANG**

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:

Alfa Yazid

NIM: 135150109111022



PROGRAM STUDI TEKNIK INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

ANALISIS PERFORMANSI PENERAPAN PENGGUNAAN PETA *OFFLINE* PADA
APLIKASI *LOCATION BASED SERVICE* BERBASIS *MOBILE WEBSITE* STUDI KASUS
PENCARIAN LOKASI TAMBAL BAN DI KOTA MALANG

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :

Alfa Yazid

NIM: 135150109111022

Skripsi ini telah diuji dan dinyatakan lulus pada
10 Agustus 2016

Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I

Dosen Pembimbing II

Dr. Eng. Herman Tolle, S.T, M.T

NIP: 197408232000121001

Drs. Marji, M.T

NIP: 196708011992031001

Mengetahui

Ketua Jurusan Teknik Informatika

Tri Astoto Kurniawan, S.T, M.T, Ph.D

NIP: 197105182003121001

PERNYATAAN ORISINALITAS

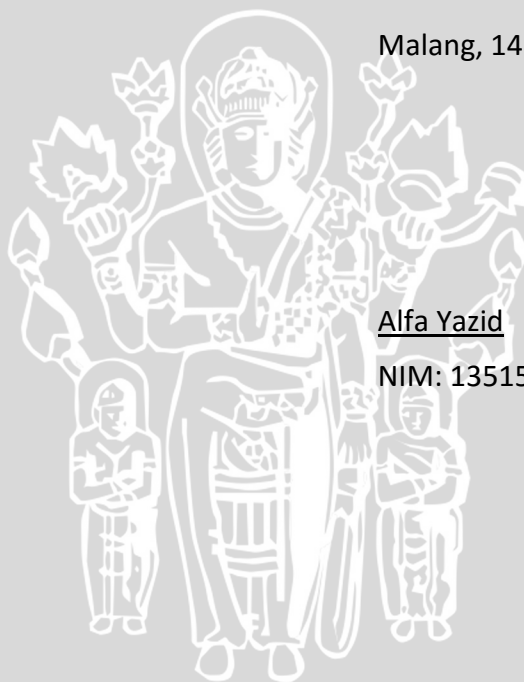
Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 14 Agustus 2016

Alfa Yazid

NIM: 135150109111022



KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadirat Allah SWT, karena hanya dengan rahmat bimbingannya Penulis dapat menyelesaikan skripsi dengan judul "Analisis Performansi Penerapan Penggunaan Peta *Offline* Pada Aplikasi *Location Based Service* Berbasis *Website* Studi Kasus Pencarian Lokasi Tambal Ban di Kota Malang" dengan baik. Tanpa rahmat dan bimbingan dari Allah SWT, maka niscaya Penulis tidak akan dapat menyelesaikan skripsi ini dengan baik dan tepat waktu.

Terima kasih pula Penulis sampaikan kepada pihak-pihak yang telah membantu Penulis dalam penyelesaian skripsi ini. Pihak-pihak tersebut yaitu antara lain :

1. Bapak Agus Wahyu Widodo, S.T, M.Cs selaku Kepala Program Studi Teknik Informatika serta Bapak / Ibu Dosen, Staff Administrasi dan Perpustakaan Program Studi Teknik Informatika Universitas Brawijaya.
2. Bapak Dr. Eng. Herman Tolle, S.T, M.T dan Bapak Drs. Marji, M.T selaku dosen pembimbing I dan dosen pembimbing II yang telah banyak memberikan ide, pemikiran, bimbingan, ilmu, dan saran dalam penyusunan skripsi ini.
3. Orang tua Penulis, Bapak Miftahul Ulum dan Ibu Neneng Salmia yang telah memberi motivasi, kasih sayang serta dukungan moril dan materil kepada Penulis.
4. Semua teman-teman SAP di Fakultas Ilmu Komputer Universitas Brawijaya, terima kasih atas segala bantuannya selama menjadi mahasiswa.
5. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya skripsi ini.

Semoga jasa dan amal baik mendapatkan balasan dari Allah SWT. Ibarat tak ada gading yang tak retak, dengan segala kerendahan diri, penulis menyadari sepenuhnya bahwa skripsi ini masih belum sempurna. Oleh karena itu kritik dan saran untuk kesempurnaan skripsi ini senantiasa penulis harapkan dari berbagai pihak.

Malang, 3 Agustus 2016

Penulis

ABSTRAK

Alfa Yazid. 2016. Analisis Performansi Penerapan Penggunaan Peta *Offline* Pada Aplikasi *Location Based Service* Berbasis *Mobile Website* Studi Kasus Pencarian Lokasi Tambal Ban Di Kota Malang. Fakultas Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing : Dr. Eng. Herman Tolle, S.T, M.T. dan Drs. Marji, S.T, M.T.

Tambal Ban merupakan salah satu fasilitas umum yang menyediakan layanan bagi pengendara kendaraan bermotor maupun non-motor yang berjalan menggunakan roda dan ban yang mengalami kebocoran pada ban-nya kini telah tersebar di berbagai daerah dalam jumlah relatif banyak, tak terkecuali di Kota Malang karena populasi masyarakat di kota Malang berbanding lurus dengan kendaraan yang dimiliki begitu pula dengan tersebarnya fasilitas Tambal Ban di berbagai daerah di Kota Malang. Dengan adanya *Open Street Map* yang merupakan peta gratis dan terbuka yang dapat digunakan di seluruh dunia, maka dari itu *Open Street Map* dapat digunakan di negara berkembang dan di zona bencana di mana internet tidak tersedia. Berdasarkan hasil pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas aplikasi *mobile website* pencarian lokasi tambal ban yang dibangun telah memenuhi kebutuhan fungsional yang dijabarkan pada tahap analisis kebutuhan. Dari hasil pengujian yang dilakukan dengan cara menjalankan aplikasi, kemudian memilih satu lokasi tambal ban dengan lokasi asal dan tujuan yang sama. Peta yang menggunakan Google Maps menghasilkan ukuran *file* hanya sebesar 172 Kb sedangkan peta yang menggunakan Open Street Map menghasilkan ukuran *file* 1.52 Mb. Pengujian kecepatan akses juga terkait menjelaskan sesuai apa yang dijelaskan dalam pengujian ukuran *file*, semakin besar ukuran *file* dari sebuah halaman *website* maka membutuhkan pula waktu yang cukup lama untuk dapat mengakses halaman tersebut. Tetapi kelebihan peta Open Street Map yang menggunakan teknologi *cache* akan jauh lebih cepat di akses jika petanya sudah pernah diakses, karena peta tersebut sudah tersimpan dalam *storage smartphone* meskipun koneksi *internet* telah terputus.

Kata kunci : Malang, Tambal Ban, peta *offline*, LBS, Google Maps, Open Street Map.

ABSTRACT

Alfa Yazid. 2016. *Application Performance Analysis Using Offline Map On Application Location Based Service-Based Mobile Website Case Study Search Location Tambal Ban On Malang city.* Faculty of Computer Science, University of Brawijaya, Malang. Advisor: Dr. Eng. Herman Tolle, S.T, M.T. and Drs.Marji, S.T, M.T.

Tambal Ban is one of the public facilities that provide services to all vehicle which using tire and leaking tire now have been provided in various areas, not least in Malang since the population of the city of Malang directly proportional to the vehicle owned facilities as well as the spread of Tambal Ban in various areas in the city of Malang. Open Street Map is a free and open forum that can be used around the world, and therefore Open Street Map can be used in developing countries and in disaster zones where the internet is not available. Based on the results of the validation test can be concluded that the implementation and mobile application functionality of Tambal Ban location searching website has been built according to the functional requirements outlined in the requirements analysis phase. The results testing is done by running the application, then select one Tambal Ban with the location of the same origin and destination. Map using Google Maps generate a file size of only 172 Kb, while maps that use Open Street Map generates 1.52 Mb file size. Testing access speed is also linked as explained as described in the test file size, the larger the file size of a web page it requires too long time to be able to access the page. But the advantages of the map that uses Open Street Map cache technology will be much faster I n access if the map had been accessed, because the map in the smartphone is already stored in storage even though the Internet connection has been disconnected.

Keywords: Malang, Tambal Ban, offline maps, LBS, Google Maps, Open Street Map.

DAFTAR ISI

PENGESAHAN	ii
PERNYATAAN ORISINALITAS	iii
KATA PENGANTAR.....	iv
ABSRAK.....	v
ABSTRACT	vi
DAFTAR ISI	vii
DAFTAR TABEL.....	x
DAFTAR KODE.....	xi
DAFTAR GAMBAR.....	xii
BAB 1 PENDAHULUAN.....	1
1.1 Latar Belakang.....	1
1.2 Rumusan Masalah.....	2
1.3 Tujuan	2
1.4 Manfaat.....	2
1.5 Batasan Masalah	2
1.6 Sistematika Pembahasan	2
BAB 2 LANDASAN KEPUSTAKAAN	4
2.1 Kajian Pustaka	4
2.2 LBS (Location Based Service)	5
2.2.1 Unsur Utama LBS.....	5
2.2.2 Komponen LBS	5
2.2.3 Cara Kerja LBS.....	6
2.3 <i>Web Service</i>	7
2.4 <i>Hypertext Markup Language (HTML)</i>	7
2.5 <i>Global Positioning System (GPS)</i>	8
2.6 JSON	8
2.7 <i>Unified Modeling Language (UML)</i>	9
2.8 Pengujian Perangkat Lunak.....	11
2.8.1 Pengujian Validasi	11
BAB 3 METODOLOGI	13

3.1 Studi Literatur	13
3.2 Metode Pengambilan Data	14
3.3 Analisis Kebutuhan Sistem.....	14
3.4 Perancangan	14
3.5 Implementasi	14
3.6 Pengujian dan Analisis	14
BAB 4 PERANCANGAN DAN IMPLEMENTASI	16
4.1 Gambaran Umum Sistem.....	16
4.2 Spesifikasi Sistem	16
4.3 Spesifikasi Kebutuhan Sistem	17
4.3.1 Identifikasi Aktor	17
4.3.2 Kebutuhan Fungsional.....	18
4.3.3 Kebutuhan Non Fungsional.....	18
4.4 Perancangan	19
4.4.1 Perancangan Use Case Diagram	19
4.4.2 Skenario <i>Use Case</i>	20
4.4.3 Perancangan Activity Diagram	21
4.4.4 Perancangan Penerapan Penggunaan Peta <i>Offline</i>	23
4.4.5 Perancangan <i>Database</i>	25
4.4.6 Perancangan Antarmuka.....	25
4.4.7 Perancangan Uji Coba	27
4.5 Implementasi Penggunaan Peta <i>Offline</i>	28
4.5.1 Spesifikasi Sistem	28
4.5.2 Batasan-batasan Implementasi.....	29
4.5.3 Implementasi Kode Program	30
4.5.4 Implementasi Antarmuka	36
BAB 5 HASIL DAN PENGUJIAN	39
5.1 Pengujian Validasi	39
5.1.1 Kasus Uji Pengujian Validasi.....	39
5.1.2 Hasil Uji Pengujian Validasi	41
5.2 Pengujian Non Fungsional	42
5.2.1 Pengujian Ukuran <i>File</i>	42

5.2.2 Pengujian Kecepatan Akses.....	43
5.2.3 Pengujian Kecepatan Akses Peta <i>Offline</i> dengan <i>cache</i>	44
5.3 Analisis Hasil Pengujian.....	44
5.3.1 Analisis Hasil Pengujian Validasi	45
5.3.2 Analisis Hasil Pengujian Ukuran <i>File</i>	45
5.3.3 Analisis Hasil Pengujian Kecepatan Akses.....	45
5.3.4 Analisis Hasil Pengujian Kecepatan Akses Peta <i>Offline</i> dengan <i>cache</i>	46
BAB 6 Penutup	48
6.1 Kesimpulan.....	48
6.2 Saran	48
DAFTAR PUSTAKA.....	49

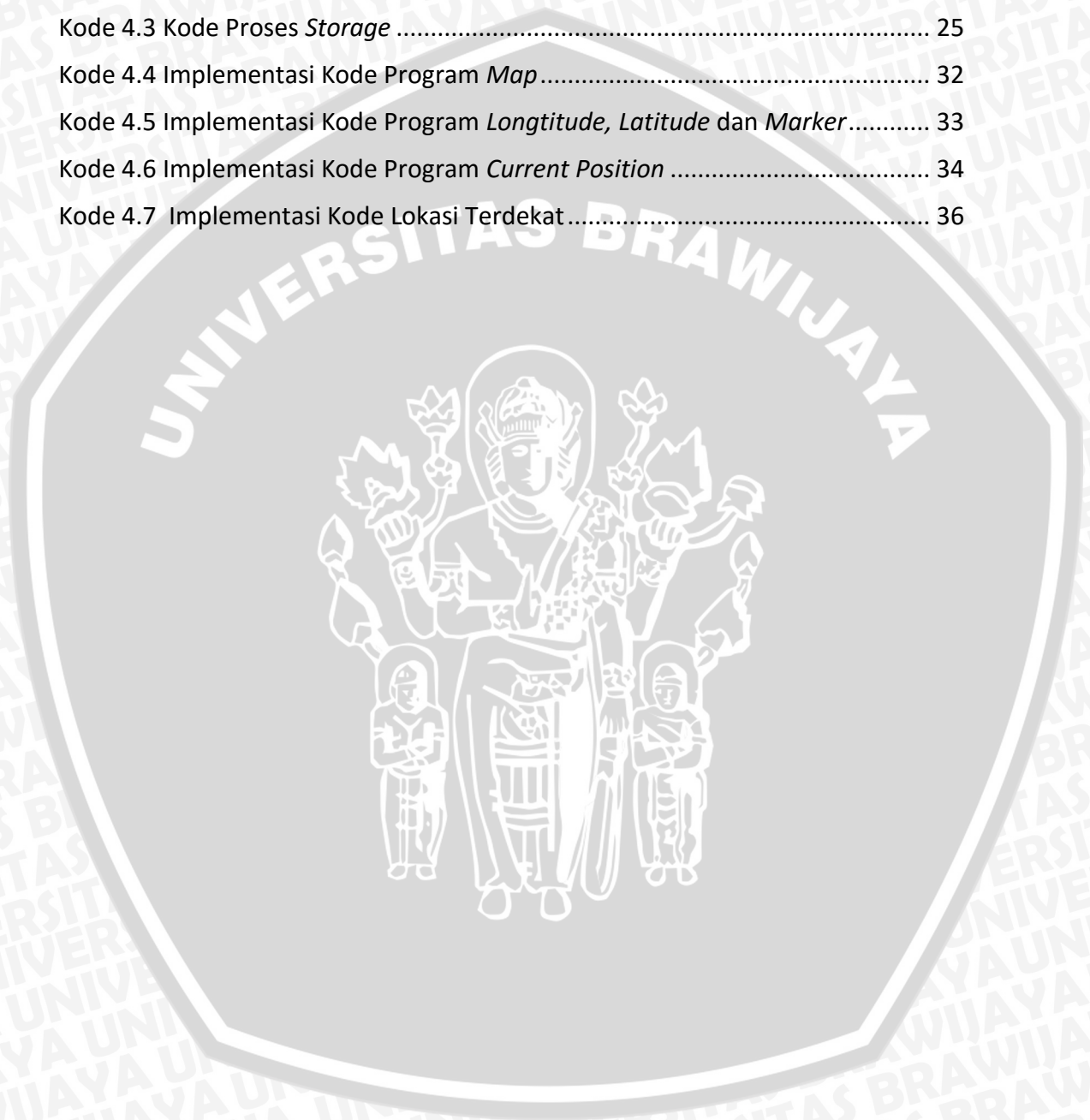


DAFTAR TABEL

Tabel 4.1 Identifikasi Aktor	18
Tabel 4.2 Spesifikasi Kebutuhan Fungsional Sistem	18
Tabel 4.3 Spesifikasi Kebutuhan Non Fungsional Sistem	18
Tabel 4.3 Skenario <i>use case</i> mengelola peta	20
Tabel 4.4 Skenario <i>use case</i> rute perjalanan	21
Tabel 4.5 Perancangan Tabel <i>Location</i>	25
Tabel 4.6 Spesifikasi Perangkat Keras Komputer	28
Tabel 4.7 Spesifikasi Perangkat Keras Smartphone	28
Tabel 4.8 Spesifikasi Perangkat Lunak Komputer	29
Tabel 4.9 Spesifikasi Perangkat Lunak Smartphone	29
Tabel 5.1 Kasus Uji Lihat Peta Tambal Ban Google Maps	39
Tabel 5.2 Kasus Uji Lihat Lokasi Sekarang <i>User</i>	40
Tabel 5.3 Kasus Uji Lihat Peta Tambal Ban Open Street Map	40
Tabel 5.4 Hasil Pengujian Validasi	41
Tabel 5.5 Hasil Pengujian Ukuran <i>File</i>	42
Tabel 5.6 Hasil Pengujian Kecepatan Akses	43
Tabel 5.7 Hasil Pengujian Kecepatan Akses Peta <i>Offline</i> dengan <i>cache</i>	44

DAFTAR KODE

Kode 4.1 Kode <i>Event Cache</i>	24
Kode 4.2 Kode Inisialisasi Leaflet	24
Kode 4.3 Kode Proses <i>Storage</i>	25
Kode 4.4 Implementasi Kode Program <i>Map</i>	32
Kode 4.5 Implementasi Kode Program <i>Longitude, Latitude</i> dan <i>Marker</i>	33
Kode 4.6 Implementasi Kode Program <i>Current Position</i>	34
Kode 4.7 Implementasi Kode Lokasi Terdekat	36



DAFTAR GAMBAR

Gambar 2.1 Alur location based service (Markkula, 2001).....	6
Gambar 2.2 Contoh <i>Use Case</i> (Pressman, 2012)	9
Gambar 2.3 Contoh Activity Diagram (Pressman, 2012)	10
Gambar 3.1 Diagram Blok Metodologi Penelitian	13
Gambar 4.1 Perancangan Umum Sistem	16
Gambar 4.2 Perancangan Web <i>Client</i>	17
Gambar 4.3 Use Case Diagram.....	19
Gambar 4.4 <i>Activity Diagram</i> Admin dan Sistem	22
Gambar 4.5 <i>Activity Diagram</i> User dan Sistem.....	22
Gambar 4.6 Flow Diagram Cache	23
Gambar 4.7 Halaman Beranda User	26
Gambar 4.8 Halaman Peta OSM	26
Gambar 4.9 Halaman <i>Contacts</i>	27
Gambar 4.10 Halaman Peta Google Maps.....	27
Gambar 4.11 Halaman Beranda <i>User</i>	37
Gambar 4.12 Halaman OSM (OpenStreetMap)	37
Gambar 4.13 Halaman <i>Contacts</i>	38
Gambar 4.14 Halaman Google Maps	38
Gambar 5.1 Grafik Perbandingan Kecepatan Akses	46

BAB 1 PENDAHULUAN

1.1 Latar Belakang

Perkembangan yang sangat pesat saat ini membuat arus kebutuhan dalam dunia teknologi informasi turut berkembang cepat. Kebutuhan untuk memperoleh informasi telah menjadi kebutuhan pokok masyarakat. Salah satu informasi yang dibutuhkan masyarakat adalah kebutuhan informasi yang berkaitan dengan tempat-tempat fasilitas umum yang dibutuhkan.

Tambal Ban merupakan salah satu fasilitas umum yang menyediakan layanan bagi pengendara kendaraan bermotor maupun non-motor yang berjalan menggunakan roda dan ban yang mengalami kebocoran pada ban-nya kini telah tersebar di berbagai daerah dalam jumlah realatif banyak, tak terkecuali di kota Malang karena populasi masyarakat di kota Malang berbanding lurus dengan kendaraan yang dimiliki begitu pula dengan tersebarnya fasilitas Tambal Ban di berbagai daerah di kota Malang. (Guru Otomotif, 2011)

Perkembangan teknologi informasi khususnya pada penerapan peta dengan menggunakan teknologi LBS (*Location Based Service*) sudah sangat berkembang dan cukup banyak dikembangkan, akan tetapi para pengembang aplikasi menggunakan peta *online* untuk mengembangkannya dengan menggunakan peta yang langsung di dapat antara lain dari *Google Maps* yang kemudian petanya langsung dapat diolah menjadi sebuah sistem. LBS memiliki kemampuan untuk mencari lokasi geografis dari sistem dan menyediakan layanan berdasarkan lokasi yang diperolehnya, kemudian dapat disajikan dalam bentuk aplikasi desktop, mobile maupun aplikasi berbasis *web*.

Melalui Open Data Commons Open Database License 1.0, kontributor OSM dapat memiliki, memodifikasi, dan membagikan data peta secara luas. Terdapat beragam jenis peta digital yang tersedia di internet, namun sebagian besar memiliki keterbatasan secara legal maupun teknis. Hal ini membuat masyarakat, pemerintah, peneliti dan akademisi, inovator, dan banyak pihak lainnya tidak dapat menggunakan data yang tersedia di dalam peta tersebut secara bebas. Di sisi lain, baik peta dasar OSM maupun data yang tersedia di dalamnya dapat diunduh secara gratis dan terbuka, untuk kemudian digunakan dan didistribusikan kembali. (Open Street Map, 2011)

Munculnya teknologi LBS yang dapat dikembangkan dengan teknologi pembuatan *website* dengan menggunakan peta *offline*, mendorong penulis untuk merancang dan menganalisis suatu sistem pengolahan data dan informasi yang meliputi penempatan lokasi Tambal Ban di kota Malang. Sistem juga mencakup penerapan seperti *point location* dan *routing* yang diterapkan pada teknologi LBS.

1.2 Rumusan Masalah

Melalui latar belakang yang telah diuraikan sebelumnya, maka dapat dirumuskan permasalahan sebagai berikut :

1. Bagaimana rancangan dan implementasi penggunaan peta *offline* pada aplikasi LBS (*Location Based Service*) berbasis *website* ?
2. Bagaimana sistem yang dirancang dengan menggunakan peta *offline* dapat menentukan lokasi Tambal Ban di Kota Malang ?
3. Bagaimana perbandingan performansi antara peta Google Maps dan peta OpenStreetMap dilihat dari parameter ukuran file dan kecepatan akses?

1.3 Tujuan

Menganalisis dan mengetahui perbandingan antara peta *online* yang dikembangkan dengan teknologi Google Maps dan peta *offline* yang menggunakan teknologi *cache* di dapat dari Open Street Map (OSM), khususnya dalam aspek :

1. Ukuran file yang dihasilkan.
2. Kecepatan akses diantara dua peta.

1.4 Manfaat

Penerapan analisis peta *offline* pada aplikasi *Location Based Service* pencarian Tambal Ban di Kota Malang berbasis *mobile website* bermanfaat untuk memudahkan dalam memperoleh informasi letak Tambal Ban di Kota Malang, mengetahui perbedaan antara peta *online* dan *offline*, dan menambah wawasan bagi penulis dalam mengembangkan peta *offline* dengan menggunakan teknologi LBS yang selama ini banyak para pengembang menggunakan peta *online* untuk membuat sebuah aplikasi LBS.

1.5 Batasan Masalah

Adapun batasan-batasan masalah dalam penelitian ini adalah :

1. Peta *offline* yang digunakan yaitu di dapat dari OpenStreetMap (OSM) dan peta *online* dari Google Maps.
2. Peta *offline* menggunakan teknologi *cache* sehingga saat pertama kali mengakses peta harus terhubung dengan internet.

1.6 Sistematika Pembahasan

Pembuatan tugas akhir ini dilakukan dengan sistematika sebagai berikut :

1. BAB I PENDAHULUAN

Bab ini berisi latar belakang penelitian, rumusan masalah, batasan masalah, tujuan penelitian, manfaat penelitian, dan sistematika penulisan.

2. BAB II LANDASAN KEPUSTAKAAN

Pada bab ini dibahas mengenai pustaka yang digunakan dalam pengerjaan skripsi. Teori-teori yang terdapat pada bab ini mencakup semua aspek yang terdapat dalam sistem.

3. BAB III METODOLOGI

Metodologi penelitian membahas metode yang digunakan dalam penelitian yang terdiri dari studi literatur, wawancara, analisis kebutuhan, perancangan, implementasi, pengujian dan pengambilan kesimpulan.

4. BAB IV. PERANCANGAN DAN IMPLEMENTASI

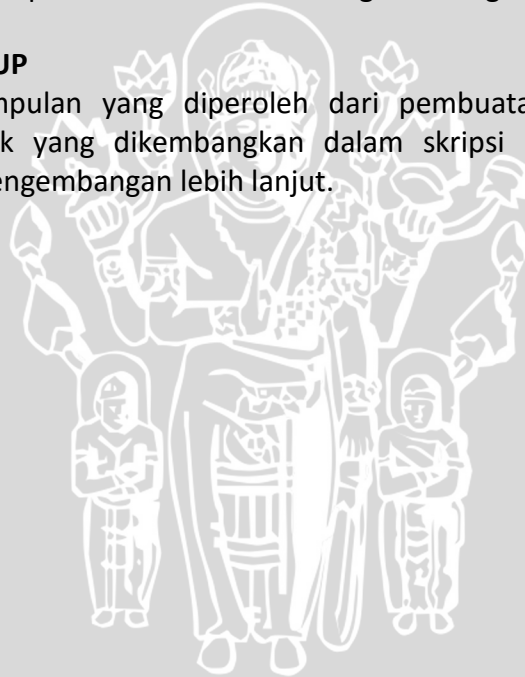
Bab ini berisi tentang alur data aplikasi, perancangan desain aplikasi, pembuatan aplikasi dan implementasi analisis penerapan penggunaan peta *offline* pada aplikasi LBS.

5. BAB V. HASIL DAN PENGUJIAN

Memuat tentang hasil pengujian dan pembahasan terhadap sistem yang telah direalisasikan. Setelah pengujian telah seluraian pendek dilakukan maka perlu untuk membandingkan dengan penilaian secara manual.

6. BAB VI PENUTUP

Memuat kesimpulan yang diperoleh dari pembuatan dan pengujian perangkat lunak yang dikembangkan dalam skripsi ini serta saran – saran untuk pengembangan lebih lanjut.



BAB 2 LANDASAN KEPUSTAKAAN

2.1 Kajian Pustaka

Dalam penulisan skripsi ini peneliti menggali informasi dari penelitian-penelitian sebelumnya sebagai bahan perbandingan, baik mengenai kekurangan atau kelebihan yang sudah ada. Selain itu, peneliti juga menggali informasi dari buku-buku maupun skripsi dalam rangka mendapatkan suatu informasi yang ada sebelumnya tentang teori yang berkaitan dengan judul yang digunakan untuk memperoleh landasan teori ilmiah.

Pertama Skripsi Win Anggita Cinantiasari mahasiswa Universitas Brawijaya dengan judul Rancang Bangun Aplikasi Mobile Pemantau Kepadatan Lalu Lintas Kota Surabaya menggunakan LBS.

Penelitian ini merupakan jenis penelitian implementatif *mobile*, yang merancang keseluruhan sistem termasuk bisnis proses dan alur data.

Kedua Skripsi Yustian Mantjoro, mahasiswa Politeknik Elektronika ITS Surabaya dengan judul “Penggunaan Teknologi OpenStreetMap Untuk GIS Fasilitas Pelayanan Umum Berbasis Android Studi Kasus Kota Palu”

Hasil penelitian ini menunjukkan beberapa komponen yang dibutuhkan dalam merancang peta yang menggunakan OpenStreetMap antara lain :

a. Memory

Tempat penyimpanan sementara data cache geospasial seperti *tiles* dan basis data fasilitas umum. Data akan terisi setelah dilakukannya pengunduhan dan sinkronisasi dari server.

b. OpenStreetMap API

Suatu layanan web yang menyediakan cara bagaimana sisi client dapat mengakses langsung data geospasial di OSM WMS server melalui antarmuka HTTP. OSM API digunakan untuk sinkronisasi data antara sisi client dan sesi server seperti saat melakukan permintaan atau update data dari sisi client ke sisi server, kemudian server merespon dan mengirim setiap permintaan dari sisi client. Sinkronisasi ke server terjadi apabila permintaan data geospasial dari Mobile Android tidak terdapat pada data cache geospasial di memori. Setiap data geospasial yang diterima dari server akan disimpan dimemori sebagai data cache geospasial.

c. Routing Service Server

Layanan penghitungan rute berserta informasinya berdasarkan data spasial OSM melalui media web atau antarmuka HTTP dengan menggunakan API penghitungan rute. Routing Service Server yang diberi nama YOURS menghasilkan pencarian berdasarkan jalur tercepat atau terpendek dan yang digunakan pada aplikasi SIG ini yaitu pencarian jalur terpendek.

2.2 LBS (Location Based Service)

Location Based Service (LBS) atau layanan berbasis lokasi adalah sebuah layanan informasi yang dapat diakses dengan perangkat bergerak melalui jaringan dan mampu menampilkan posisi secara geografis keberadaan perangkat bergerak tersebut. *Location Based Service* dapat berfungsi sebagai layanan untuk mengidentifikasi lokasi dari seseorang atau suatu objek tertentu, seperti menemukan lokasi SPBU terdekat atau lokasi lainnya.

Hal paling penting dari *Location Based Service* (LBS) dapat bekerja sesuai yang diinginkan oleh pengembang aplikasi android. Androidpun memungkinkan pengembang menentukan metode pencarian lokasi yang dibutuhkan dan juga dapat mengatur kebutuhan daya, biaya dan akurasi berdasarkan spesifik yang akan dibuat untuk aplikasi tersebut (Virrantaus et al., 2001).

2.2.1 Unsur Utama LBS

Dua unsur utama dari *Location Based Service* adalah :

a. *Location Manager (API Maps)* : Menyediakan perangkat bagi sumber atau *source* untuk LBS, *Application Programming Interface (API) maps* menyediakan fasilitas untuk menampilkan atau memanipulasi peta.

b. *Location Providers (API Location)* : Menyediakan teknologi pencarian lokasi yang digunakan oleh perangkat. *API Location* berhubungan dengan data GPS (*Global Positioning System*) dan data lokasi *real-time*. *API Location* berada pada paket Android yaitu dalam "paket android.location". Lokasi, perpindahan, serta kedekatan dengan lokasi tertentu dapat ditentukan melalui *Location Manager*.

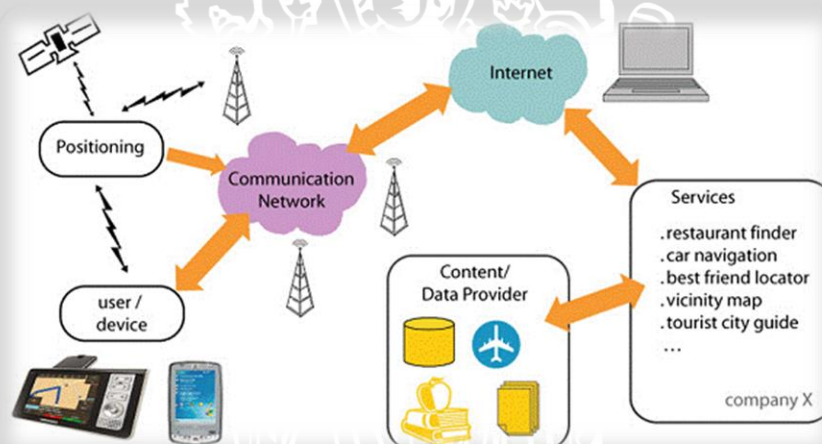
2.2.2 Komponen LBS

Terdapat lima komponen pendukung utama dalam teknologi Layanan Berbasis Lokasi antara lain :

1. Piranti *Mobile*, adalah satu komponen penting dalam LBS. Piranti ini berfungsi sebagai alat bantu bagi pengguna untuk meminta informasi. Hasil dari informasi yang diminta dapat berupa teks, suara, gambar dan lain sebagainya. Piranti *mobile* yang dapat digunakan bisa berupa PDA, *smartphone*, laptop. Selain itu, piranti *mobile* dapat juga berfungsi sebagai alat navigasi di kendaraan seperti halnya alat navigasi berbasis GPS.
2. Jaringan Komunikasi, komponen ini berfungsi sebagai jalur penghubung yang dapat mengirimkan data-data yang dikirim oleh pengguna dari piranti *mobile*-nya untuk kemudian dikirimkan ke penyedia layanan dan kemudian hasil permintaan tersebut dikirimkan kembali oleh penyedia layanan kepada pengguna.
3. Komponen *Positioning* (Petunjuk Posisi/Lokasi), Setiap layanan yang diberikan oleh penyedia layanan biasanya akan berdasarkan pada posisi pengguna yang meminta layanan tersebut. Oleh karena itu diperlukan komponen yang berfungsi sebagai pengolah/pemroses yang akan

menentukan posisi pengguna layanan saat itu. Posisi pengguna tersebut bisa didapatkan melalui jaringan komunikasi mobile atau juga menggunakan *Global Positioning System* (GPS).

4. Penyedia layanan dan aplikasi, merupakan komponen LBS yang memberikan berbagai macam layanan yang bisa digunakan oleh pengguna. Sebagai contoh ketika pengguna meminta layanan agar bisa tahu posisinya saat itu, maka aplikasi dan penyedia layanan langsung memproses permintaan tersebut, mulai dari menghitung dan menentukan posisi pengguna, menemukan rute jalan, mencari data di *Yellow Pages* sesuai dengan permintaan dan masih banyak lagi yang lainnya.
5. Penyedia data dan konten, penyedia layanan tidak selalu menyimpan seluruh data dan informasi yang diolahnya. Karena bisa jadi berbagai macam data dan informasi yang diolah tersebut berasal dari pengembang/pihak ketiga yang memang memiliki otoritas untuk menyimpannya. Sebagai contoh basis data geografis dan lokasi bisa saja berasal dari badan-badan milik pemerintah atau juga data-data perusahaan/bisnis/industri bisa saja berasal dari *Yellow pages*, maupun perusahaan penyedia data lainnya. Secara lengkap lima komponen pendukung LBS tersebut dapat dilihat pada Gambar 2.1 (Steiniger et al., 2006)



Gambar 2.1 Alur location based service (Markkula, 2001)

2.2.3 Cara Kerja LBS

Untuk menggambarkan cara kerja LBS, aplikasi LBS akan mencarikan informasi mengenai lokasi SPBU yang berada di sekitar posisi pengguna.

1. Fungsi pencarian telah diaktifkan, posisi pengguna sebenarnya dari perangkat *mobile* diperoleh dari *Positioning Service*. Hal ini dapat dilakukan baik oleh perangkat menggunakan GPS sendiri atau layanan posisi jaringan yang berasal dari *provider* (Cell Tower). Setelah itu perangkat *mobile* pengguna mengirimkan permintaan informasi, yang berisi tujuan untuk mencari dan mengirimkan posisi melalui jaringan komunikasi ke *gateway* telekomunikasi.

2. *Gateway* memiliki tugas untuk bertukar pesan di antara jaringan komunikasi selular dan internet. Oleh karena itu dia mengetahui alamat *web* dari beberapa aplikasi *server* dan rute permintaan ke spesifik *server* tertentu. *Gateway* akan menyimpan juga informasi tentang perangkat *mobile* yang telah meminta informasi.
3. Aplikasi *server* membaca permintaan dan mengaktifkan layanan yang terkait.
4. Kemudian, *service* menganalisis lagi pesan dan memutuskan mana informasi dan posisi pengguna diperlukan untuk menjawab permintaan pengguna.
5. Selanjutnya *service* akan menemukan bahwa informasi lokasi SPBU.
6. Setelah sekarang semua informasi *service* akan melakukan *buffer* spasial dan *query routing* untuk mendapatkan beberapa SPBU terdekat. Setelah itu hasil dikirim kembali ke pengguna melalui *internet*, *gateway* dan jaringan *mobile*.
7. Kemudian, informasi mengenai SPBU akan disampaikan kepada pengguna dalam bentuk peta *digital*.

2.3 Web Service

Web service adalah sebuah software yang dirancang untuk mendukung interoperabilitas interaksi mesin-ke-mesin melalui sebuah jaringan. *Web service* secara teknis memiliki mekanisme interaksi antar sistem sebagai penunjang interoperabilitas, baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). *Web service* memiliki layanan terbuka untuk kepentingan integrasi data dan kolaborasi informasi yang bisa diakses melalui internet oleh berbagai pihak menggunakan teknologi yang dimiliki oleh masing-masing pengguna (Sutanta, 2012).

Teknologi *web service* ini sudah banyak di implementasikan oleh organisasi-organisasi bisnis untuk mengkolaborasikan sistem-sistem di dalam internal organisasi. *Web service* menjanjikan banyak keuntungan seperti peningkatan produktivitas, efisiensi, dan akurasi. Fokus atau tujuan utama dari *web service* ini adalah untuk membantu aplikasi agar dapat saling berkomunikasi satu sama lain. Semua protokol transpor yang berbasiskan internet seperti HTTP, SMTP (*Simple Mail Transfer Protocol*), dan FTP (*File Transfer Protocol*) bisa diimplementasikan untuk saling bertukar pesan melalui *web service*.

2.4 Hypertext Markup Language (HTML)

Hypertext Markup Language (HTML) merupakan sebuah bahasa yang digunakan untuk mendefinisikan struktur konten dari dokumen / laman berbasis web. Kode-kode HTML tersebut kemudian diterjemahkan oleh *web browser* menjadi sebuah tampilan laman web. HTML merupakan W3C *Recommendation*,

yang mana versi stabil terakhir adalah XHTML 1.1, adapun HTML 5 masih dalam tahap pengembangan.

HTML berupa kode-kode *tag* yang menginstruksikan *browser* untuk menghasilkan tampilan sesuai dengan yang diinginkan. Sebuah *file* yang merupakan *file* HTML dapat dibuka dengan menggunakan *browser web* seperti Mozilla Firefox atau Microsoft Internet Explorer. HTML juga dapat dikenali oleh aplikasi pembuka *email* ataupun dari PDA dan program lain yang memiliki kemampuan *browser* (Mukhlisin, 2013).

2.5 Global Positioning System (GPS)

GPS (*Global Positioning System*) adalah sistem satelit navigasi dan penentuan posisi yang dimiliki dan dikelola oleh Amerika Serikat. Sistem ini didesain untuk memberikan posisi dan kecepatan tiga dimensi serta informasi mengenai waktu, secara kontinyu di seluruh dunia tanpa bergantung waktu dan cuaca, kepada banyak orang secara simultan. Pada dasarnya GPS terdiri dari 3 segmen utama yaitu : (Abidin, 2007)

1. Segmen angkasa (*space segment*)

Satelit GPS bisa dianalogikan sebagai stasiun radio di angkasa, yang dilengkapi dengan antena-antena untuk mengirim dan menerima sinyal-sinyal gelombang. Sinyal ini selanjutnya diterima oleh *receiver* GPS di/dekat permukaan bumi, dan digunakan untuk menentukan informasi posisi, kecepatan, maupun waktu.

2. Segmen system kontrol (*control system segment*)

Disamping memonitor dan mengontrol kesehatan seluruh satelit beserta seluruh komponennya, segmen kontrol ini juga berfungsi menentukan orbit dari seluruh satelit GPS yang merupakan informasi *vital* untuk penentuan posisi dengan satelit.

3. Segmen pengguna (*user segment*)

Segmen pengguna terdiri dari para pengguna satelit GPS. Dalam hal ini alat penerima sinyal GPS (*GPS receiver*) diperlukan untuk menerima dan memproses sinyal dari satelit GPS untuk digunakan dalam penentuan posisi, kecepatan dan waktu.

2.6 JSON

JSON (*JavaScript Object Nation*) adalah format pertukaran data yang ringan, mudah dibaca dan ditulis oleh manusia, serta mudah diterjemahkan dan dibuat (*generate*) oleh computer. JSON merupakan format teks yang tidak bergantung pada bahasa pemrograman apapun karena menggunakan gaya bahasa yang umum digunakan oleh *programmer* keluarga C termasuk C, C++, C#, Java, JavaScript, Perl, Python dll. Oleh karena sifat-sifat tersebut, menjadikan JSON ideal sebagai bahasa pertukaran data. JSON terbuat dari dua struktur :

1. Kumpulan pasangan nama/nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), table hash (*hash table*), daftar berkunci (*keyed list*), atau *associative array*.

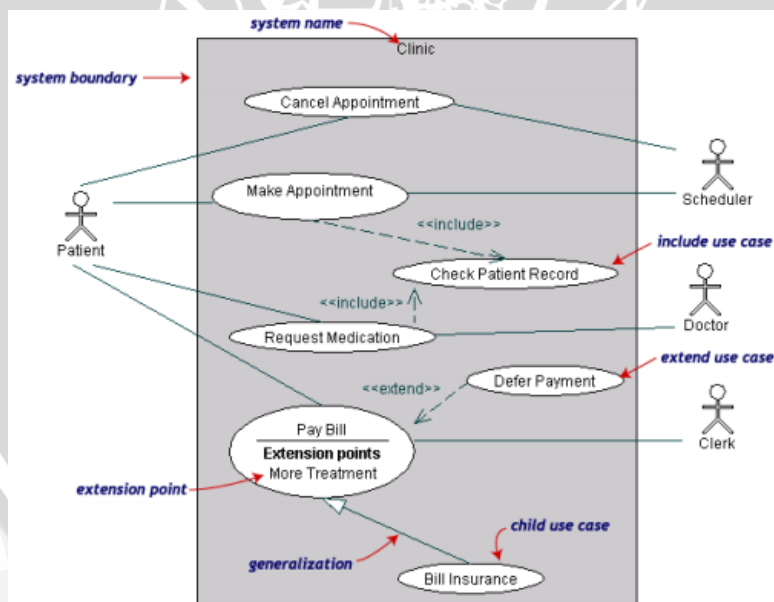
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vector (*vector*), daftar (*list*), atau urutan (*sequence*).

Struktur-struktur data ini disebut sebagai struktur data universal. Pada dasarnya, semua bahasa pemrograman *modern* mendukung struktur data ini dalam bentuk yang sama maupun berlainan. Hal ini pantas disebut demikian karena format data mudah dipertukarkan dengan bahasa-bahasa pemrograman yang juga berdasarkan pada struktur data ini (Json, 2014).

2.7 Unified Modeling Language (UML)

Unified Modeling Language adalah sebuah bahasa yang telah menjadi standar dalam industri untuk visualisasi, merancang dan mendokumentasikan sistem piranti lunak. UML menawarkan sebuah standar untuk merancang model sebuah sistem. UML mendefinisikan notasi dan *syntaxis/semantic*. Notasi UML merupakan sekumpulan bentuk khusus untuk menggambarkan berbagai diagram piranti lunak. Setiap bentuk memiliki makna tertentu, dan UML *syntaxis* mendefinisikan bagaimana bentuk-bentuk tersebut dapat dikombinasikan UML mendefinisikan diagram-diagram sebagai berikut (Dharwiyanti, 2003):

1. Diagram Use Case

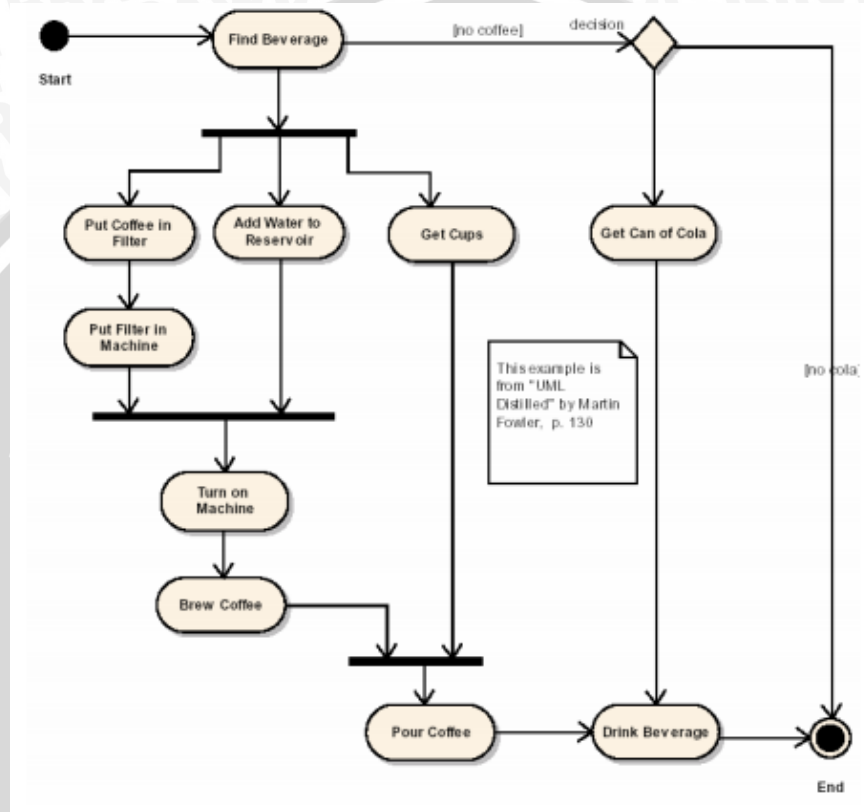


Gambar 2.2 Contoh Use Case (Pressman, 2012)

Diagram use case menggambarkan fungsionalitas yang diharapkan dari sebuah sistem. Yang ditekankan adalah “apa” yang diperbuat sistem, dan bukan “bagaimana”. Sebuah *use case* merepresentasikan sebuah interaksi antara aktor dengan sistem. *Use case* merupakan sebuah pekerjaan tertentu, misalnya login ke

sistem, meng-*create* sebuah daftar belanja, dan sebagainya. Seorang/sebuah aktor adalah sebuah entitas manusia atau mesin yang berinteraksi dengan sistem untuk melakukan pekerjaan-pekerjaan tertentu. *Use case* diagram dapat sangat membantu bila kita sedang menyusun kebutuhan sebuah sistem, mengkomunikasikan rancangan dengan klien, dan merancang *test case* untuk semua fitur yang ada pada sistem.

2. Diagram Activity



Gambar 2.3 Contoh Activity Diagram (Pressman, 2012)

Diagram *activity* menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. Diagram *activity* juga dapat menggambarkan proses paralel yang mungkin terjadi pada beberapa eksekusi. Diagram *activity* merupakan *state* diagram khusus, di mana sebagian besar *state* adalah *action* dan sebagian besar transisi di-trigger oleh selesainya *state* sebelumnya (*internal processing*). Oleh karena itu diagram *activity* tidak menggambarkan *behaviour* internal sebuah sistem (dan interaksi antar subsistem) secara eksak, tetapi lebih menggambarkan proses-proses dan jalur-jalur aktivitas dari level atas secara umum.

Sebuah aktivitas dapat direalisasikan oleh satu *use case* atau lebih. Aktivitas menggambarkan proses yang berjalan, sementara *use case* menggambarkan bagaimana aktor menggunakan sistem untuk melakukan aktivitas. Sama seperti *state*, standar UML menggunakan segiempat dengan sudut membulat untuk menggambarkan aktivitas. *Decision* digunakan untuk menggambarkan *behaviour*

pada kondisi tertentu. Untuk mengilustrasikan proses-proses paralel (*fork* dan *join*) digunakan titik sinkronisasi yang dapat berupa titik, garis horizontal atau vertikal. Diagram *activity* dapat dibagi menjadi beberapa *object swimlane* untuk menggambarkan objek mana yang bertanggung jawab untuk aktivitas tertentu

2.8 Pengujian Perangkat Lunak

Pengujian perangkat lunak memerlukan perancangan kasus uji (*test case*) agar dapat menemukan kesalahan dalam waktu singkat dan usaha minimum. Berbagai macam metode perancangan kasus uji telah berevolusi. Metode-metode ini menyediakan mekanisme yang dapat membantu memastikan kelengkapan dari pengujian dan menyediakan kemungkinan tertinggi untuk menemukan kesalahan-kesalahan dalam perangkat lunak (Sommerville, 2011).

Pada skripsi ini teknik atau metode perancangan kasus uji yang digunakan adalah *black-box testing*. *Black-box testing* berfokus pada persyaratan fungsional perangkat lunak. Dengan demikian, pengujian *black-box* memungkinkan perekayasa perangkat lunak mendapatkan serangkaian kondisi *input* yang sepenuhnya menggunakan semua persyaratan fungsional untuk semua program.

Pengujian *black-box* merupakan pendekatan komplementer yang kemungkinan besar mampu mengungkap kelas kesalahan. Pengujian *black-box* berusaha menemukan kesalahan dalam kategori berikut :

1. Fungsi-fungsi yang tidak benar atau hilang.
2. Kesalahan antarmuka.
3. Kesalahan dalam struktur data atau akses basis data eksternal.
4. Kesalahan kinerja.
5. Inisialisasi dan kesalahan terminasi.

Pengujian *black-box* cenderung diaplikasikan selama tahap akhir pengujian. Pengujian *black-box* memperhatikan struktur kontrol, maka perhatian berfokus pada domain informasi. Strategi untuk pengujian perangkat lunak mengintegrasikan metode desain *test case* perangkat lunak ke dalam sederetan langkah yang direncanakan dengan baik.

Pada skripsi ini proses pengujian dilakukan dengan menggunakan pengujian validasi (*validation testing*). Sedangkan pengujian non-fungsional menggunakan pengujian ukuran *file* dan pengujian kecepatan akses.

2.8.1 Pengujian Validasi

Validasi dapat ditentukan dengan berbagai cara, tetapi definisi yang sederhana adalah bahwa validasi berhasil bila perangkat lunak berfungsi dengan cara yang dapat diharapkan secara bertanggung jawab oleh pelanggan. Validasi perangkat lunak dicapai melalui sederetan pengujian *black-box* yang memperlihatkan konformitas dengan persyaratan. Baik rencana maupun prosedur didesain untuk memastikan apakah semua persyaratan fungsional terpenuhi, semua persyaratan kinerja dicapai, dokumentasi dilakukan dengan benar dan

direkayasa oleh manusia, dan persyaratan lainnya dipenuhi (transportabilitas, kompatibilitas, pembetulan kesalahan, maintainabilitas).



BAB 3 METODOLOGI

Pada bab ini dijelaskan metodologi yang digunakan dalam penelitian. Metode penelitian lebih difokuskan pada analisis peta *offline*. Langkah-langkah yang dilakukan adalah identifikasi permasalahan, identifikasi solusi, realisasi solusi dan analisa hasil realisasi solusi. Diagram blok metode penelitian ditunjukkan pada Gambar berikut :



Gambar 3.1 Diagram Blok Metodologi Penelitian

3.1 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan untuk menunjang penulisan skripsi. Teori-teori pendukung tersebut meliputi :

1. Google Maps API
2. Geolocation API
3. Leaflet Open Street Map API
4. *Responsive Web Design*
5. HTML 5
6. *JavaScript*
7. CSS3
8. *Location Based Service*
9. JSON

3.2 Metode Pengambilan Data

Data yang diperlukan untuk penelitian ini berupa data lokasi tambal ban di kota Malang yang diperoleh berdasarkan survei langsung ke lapangan. Data tambal ban yang didapat berupa *longitude* dan *latitude* lokasi tambal ban tersebut. Data lokasi Tambal Ban yang digunakan adalah hanya khusus untuk lokasi Tambal Ban Nitrogen yang terdapat di Kota Malang sebanyak 10 data lokasi Tambal Ban Nitrogen.

3.3 Analisis Kebutuhan Sistem

Analisis kebutuhan digunakan untuk mendapatkan kebutuhan aplikasi tambal ban yang akan dibangun. Metode analisis menggunakan bahasa pemodelan UML (*Unified Modelling Language*). Diagram *use case* digunakan untuk mendeskripsikan kebutuhan-kebutuhan dan fungsionalitas perangkat lunak dari sudut pandang pengguna. Analisis kebutuhan dilakukan dengan mengidentifikasi semua kebutuhan aplikasi tambal ban yang akan dimodelkan dalam diagram *use case*. Tiap *use case* dalam diagram tersebut juga akan dijelaskan lebih rinci dalam skenario *use case*.

3.4 Perancangan

Perancangan perangkat lunak dilakukan setelah semua kebutuhan perangkat lunak didapatkan melalui tahap analisis kebutuhan. Perancangan perangkat lunak dilakukan berdasarkan pemodelan UML (*Unified Modeling Language*). Perancangan dimulai dari perancangan alur atau aktifitas yang dilakukan pengguna secara prosedural yang dimodelkan dalam *activity diagram*. Interaksi antar objek yang telah diidentifikasi dan dimodelkan dalam *sequence diagram*. Di dalam proses ini peta yang dibutuhkan adalah peta dari Google dan OpenStreetMap dimana Google untuk membuat peta yang berjalan dengan koneksi internet sedangkan OSM berjalan tanpa koneksi internet.

3.5 Implementasi

Implementasi aplikasi pencarian lokasi Tambal Ban Proses ini mengimplementasikan rancangan sistem ke dalam kode-kode HTML untuk sebuah aplikasi web. Untuk membantu implementasi aplikasi web digunakan piranti Adobe Dreamweaver. Pada tahap akhir dilakukan implementasi simulasi pada hardware secara langsung menggunakan Android mobile device.

3.6 Pengujian dan Analisis

Pengujian ini memastikan bahwa aplikasi telah berjalan sesuai yang diharapkan. Metode pengujian yang akan digunakan adalah pengujian unit, pengujian black-box. Proses pengujian aplikasi dimulai dari pengujian unit, pengujian validasi, dan pengujian kompatibilitas. Pengujian kompatibilitas dilakukan untuk mengetahui bagaimana kompatibilitas aplikasi ketika berjalan dengan beberapa jenis mobile device.

Kemudian pengujian kedua peta yaitu Google Map dan OpenStreetMap untuk mengetahui hasil dari analisis perbandingan sesuai parameter yang digunakan dalam tujuan penelitian yaitu, ukuran file dan kecepatan akses dari kedua peta. Analisis juga dilakukan untuk mengetahui hasil dari pengujian aplikasi sehingga didapatkan kesimpulan dari pengujian yang dilakukan.



BAB 4 PERANCANGAN DAN IMPLEMENTASI

4.1 Gambaran Umum Sistem

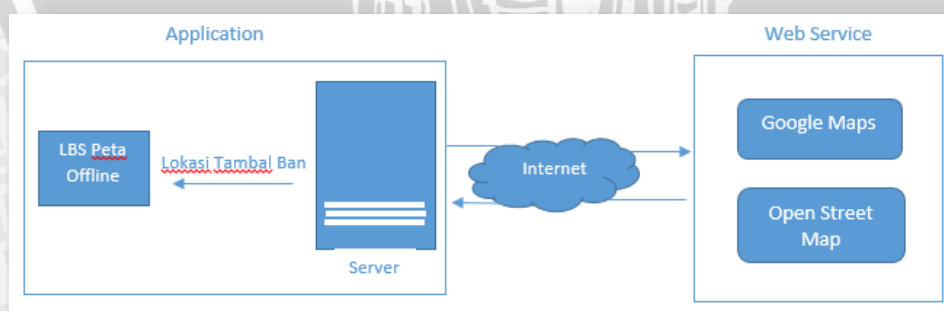
Pada bagian ini dibahas mengenai gambaran analisis 16system secara umum. Aplikasi penerapan peta *offline* ini dirancang untuk menerapkan peta *offline* dan membandingkannya dengan peta *online*. Parameter yang digunakan dalam membedakan kedua peta yaitu dalam hal ukuran *file* yang dihasilkan, kecepatan akses diantara dua peta dan akurasi penempatan sebuah lokasi.

Peta *online* yang digunakan dalam penerapan ini yaitu peta GoogleMaps yang akan dibandingkan dengan peta *offline* yang menggunakan OpenStreetMap. Peta *online* yang dirancang dengan GoogleMaps akan berjalan seperti biasa layaknya aplikasi LBS yang sudah banyak diterapkan, sedangkan peta *offline* yang berjalan dengan peta OpenStreetMap menggunakan teknologi *cache* yaitu untuk membuat peta dapat tetap diakses walau jaringan internet telah terputus.

Pada dasarnya, teknik *cache* bertujuan untuk menampilkan sebuah halaman *website* lebih cepat di waktu berikutnya apabila *user* membuka kembali situs-situs *website* tersebut.

4.2 Spesifikasi Sistem

Aplikasi analisis penerapan peta *offline* Location Based Service berbasis *website* studi kasus pencarian tambal ban ini dibagi atas dua sistem yaitu *web service* dan *web client*. Kedua sistem tersebut berkomunikasi dengan menggunakan *Web Service*. Admin dapat mengatur semua daftar lokasi tambal ban yang terdapat di kota Malang yang nantinya dapat di akses oleh User. Semua proses data yang dilakukan pada *web client* nantinya akan dikirim ke *web service* dan disimpan ke *database*. Perancangan umum sistem ditunjukkan pada Gambar 4.1.



Gambar 4.1 Perancangan Umum Sistem

Gambar 4.1 menunjukkan terdapat dua sistem utama yaitu *application* dan *web service*. *Web service* berguna mengambil data lokasi tambal ban pada *database* kemudian dikirim ke Google Maps dan OSM. Setelah data lokasi diterima

kemudian User akan mengakses data lokasi yang ingin dituju. Untuk gambar perancangan dari *client* ditunjukkan pada Gambar 4.2.



Gambar 4.2 Perancangan Web Client

Gambar 4.2 merupakan perancangan *web client* berdasarkan aktor yang mengakses sistem. Untuk penjelasan dari setiap aktor akan lebih dijelaskan pada bagian identifikasi aktor. Peta yang terdapat dalam sistem terbagi menjadi dua proses.

Proses pertama adalah peta *online* yang petanya didapat dari Google Maps dan semua sistemnya berjalan dengan membutuhkan koneksi internet, kemudian lokasi yang di ambil untuk studi kasus yaitu tambal ban di kota Malang.

Proses kedua adalah peta *offline* yang dimana petanya didapat dari OpenStreetMap yang menggunakan sistem proses *cache*, yaitu peta yang datanya harus di unduh terlebih dahulu sebelum menentukan lokasi kemudian jika peta sudah di unduh maka peta tidak lagi membutuhkan koneksi internet.

4.3 Spesifikasi Kebutuhan Sistem

Berikut ini adalah spesifikasi kebutuhan pada sistem penerapan analisis performansi penggunaan peta *offline* pada aplikasi LBS berbasis web studi kasus pencarian lokasi tambal ban di Kota Malang.

4.3.1 Identifikasi Aktor

Identifikasi Aktor sistem yang terdapat dalam sistem masing-masing akan dideskripsikan sebagai berikut :

Tabel 4.1 Identifikasi Aktor

No	Aktor	Deskripsi
1	Admin	Admin adalah sebuah aktor yang dapat mengelola fitur atau fungsionalitas sistem, seperti melihat semua data lokasi tambal ban, kelola data lokasi tambal ban dan mengelola semua fitur yang ada dalam sistem.
2	User	User adalah sebuah aktor yang dapat melihat lokasi tambal ban terdekat dari posisinya sekarang maupun yang ada di daerah Kota Malang dan melihat rute yang digunakan untuk menuju ke suatu lokasi.

4.3.2 Kebutuhan Fungsional

Kebutuhan fungsional adalah semua kebutuhan yang berhubungan dengan sistem yang masing-masing akan dispesifikasikan sebagai berikut :

Tabel 4.2 Spesifikasi Kebutuhan Fungsional Sistem

No	Kebutuhan	Aktor
1	Melihat Daftar Lokasi Tambal Ban	Admin, User
2	Menentukan rute lokasi Tambal Ban	User
3	Menambah Data lokasi Tambal Ban	Admin
4	Mengubah Data lokasi Tambal Ban	Admin
5	Menghapus Data lokasi Tambal Ban	Admin

4.3.3 Kebutuhan Non Fungsional

Kebutuhan non fungsional untuk penerapan analisis peta *offline* berbasis web ini yaitu menganalisis dari tiga parameter antara lain ukuran file yang dihasilkan dari sebuah proses rute lokasi tambal ban, kecepatan akses diantara dua peta antara Google Maps yang berjalan dengan *online* dan OpenStreetMap yang berjalan dengan *offline*, dan akurasi penempatan sebuah lokasi User saat itu.

Tabel 4.3 Spesifikasi Kebutuhan Non Fungsional Sistem

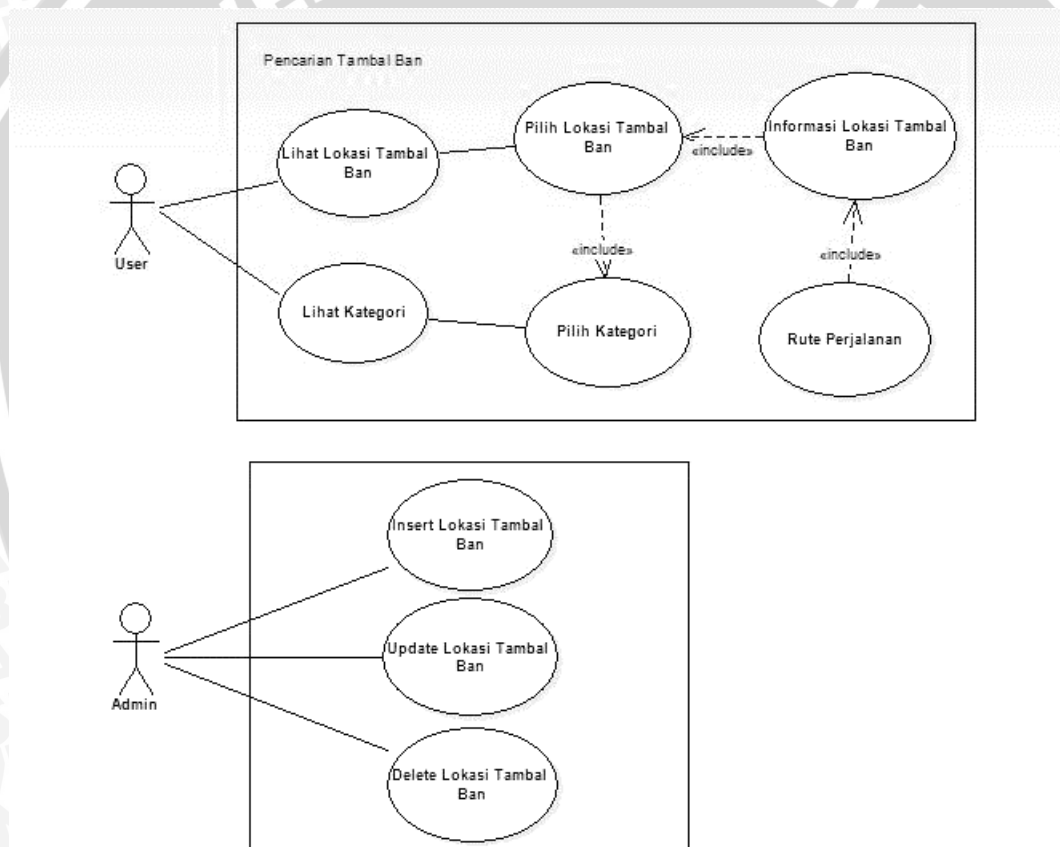
No	Parameter	Deskripsi Kebutuhan
1	<i>Realibility</i>	Aplikasi berjalan secara <i>offline</i> dengan menggunakan teknik <i>cache</i> .
2	<i>Response Time</i>	Peta <i>offline</i> yang menggunakan teknik <i>cache</i> berdasarkan kecepatan akses, dan efisiensi data.

4.4 Perancangan

Perancangan adalah tahap ketika semua kebutuhan sistem dan kebutuhan data sudah terpenuhi sehingga bisa memulai perancangan berdasarkan analisa-analisa tersebut. Teori-teori dari pustaka dan data digabungkan dengan ilmu yang didapat diimplementasikan untuk merancang aplikasi penerapan peta *offline*. Perancangan sistem berdasarkan *Object Oriented Analysis* yaitu menggunakan pemodelan UML (*Unified Modeling Language*).

4.4.1 Perancangan Use Case Diagram

Use case adalah sebuah diagram yang menggambarkan fungsionalitas dari suatu sistem, sehingga pengguna akan paham dan mengerti kegunaan sistem yang akan dibangun. *Use case* meliputi semua hal yang ada pada sistem sedangkan aktor meliputi hal yang ada di luar sistem. Untuk *use case diagram* ditunjukkan pada Gambar 4.3.



Gambar 4.3 Use Case Diagram

Gambar 4.3 merupakan perancangan *use case diagram*. *Use case diagram* terbagi menjadi dua subsistem yaitu sistem yang dijalankan *User* dan *Admin* yang masing-masing memiliki tugas yang berbeda.

4.4.2 Skenario *Use Case*

Skenario *use case* bertujuan untuk menjelaskan detail masing-masing *use case* yang terdapat dalam diagram *use case*. Pada skenario *use case*, akan diberikan uraian nama *use case*, deskripsi global tentang *use case*, kondisi awal yang harus dipenuhi dan kondisi akhir yang diharapkan setelah berjalannya fungsional *use case*. Skenario *use case* yang digunakan adalah sistem pada User dan Admin yang bertujuan untuk sistem yang menganalisis peta *offline* dapat dipahami dari mulai berjalan sampai akhir.

1. Skenario *use case* mengelola data peta

Kebutuhan fungsional yang harus disediakan oleh sistem adalah kebutuhan untuk dapat mengelola lokasi peta mulai dari tambah, ubah dan hapus. Skenario *use case* untuk tambah, ubah dan hapus lokasi peta ditunjukkan pada Tabel 4.3.

Tabel 4.3 Skenario *use case* mengelola peta

Nama	Tambah, ubah, hapus daftar lokasi tambal ban.
Tujuan	Admin dapat menambah, mengubah dan menghapus lokasi tambal ban.
Deskripsi	<i>Use case</i> ini menjelaskan proses untuk dapat menambah, mengubah, dan menghapus daftar lokasi tambal ban.
Aktor	Admin
Alur Utama	
Aksi Aktor	Reaksi Sistem
Admin masuk ke halaman tambah lokasi tambal ban.	1. <i>Server</i> menerima kiriman dari Admin. 2. <i>Server</i> melakukan validasi data lokasi yang dikirim. 3. <i>Server</i> melakukan proses penyimpanan data lokasi pada <i>database</i> .
Alur alternatif : Jika data tidak valid maka data tidak akan tersimpan.	
Kondisi Akhir Sukses	Lokasi tambal ban yang baru akan muncul berdasarkan data yang dibuat sebelumnya

2. Skenario *use case* rute perjalanan

Kebutuhan fungsional yang harus disediakan oleh sistem adalah kebutuhan untuk mendapatkan sebuah rute perjalanan menuju lokasi tambal ban tertentu. Skenario *use case* untuk rute perjalanan terdapat pada Tabel 4.4.

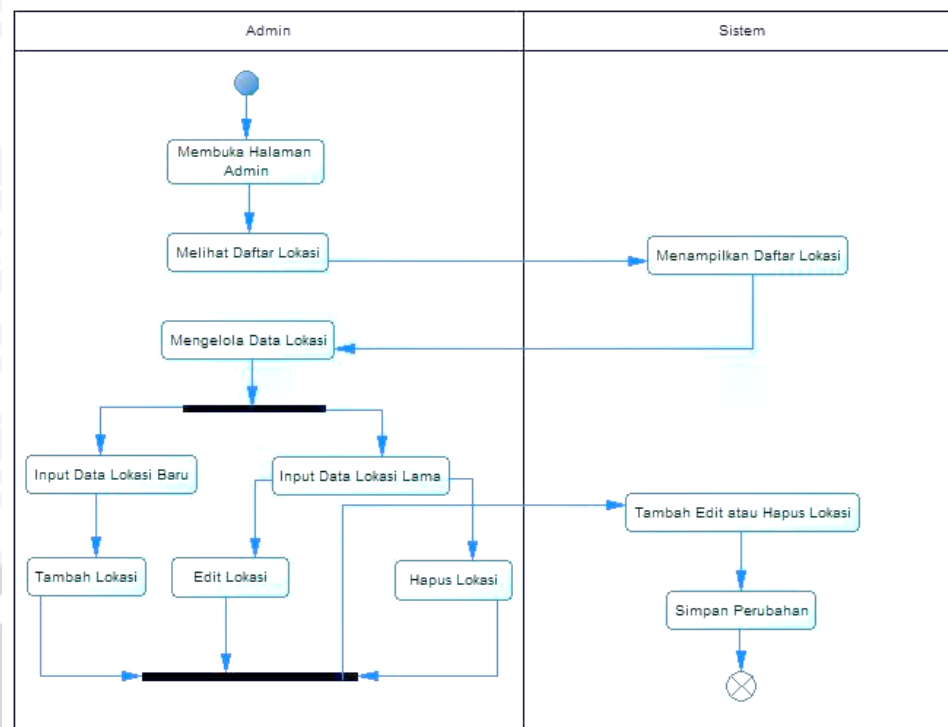
Tabel 4.4 Skenario *use case* rute perjalanan

Nama	Rute perjalanan
Tujuan	Aktor mendapatkan rute perjalanan ke suatu lokasi tambal ban tertentu.
Deskripsi	<i>Use case</i> ini menjelaskan proses bagaimana aktor mendapatkan rute perjalanan ke lokasi tambal ban yang sudah ditentukan sebelumnya.
Aktor	User
Alur Utama	
Aksi Aktor	Reaksi Sistem
1. Aktor masuk ke dalam menu lokasi tambal ban.	1. Sistem melihatkan semua lokasi tambal ban yang terdapat di Kota Malang.
2. Aktor memilih kategori tambal ban.	2. Sistem memberikan sebuah rute perjalanan lokasi tambal ban yang telah ditentukan User.
3. Aktor memilih tujuan lokasi tambal ban.	
Alur alternatif: -	
Kondisi Akhir Sukses	Aktor mendapatkan rute perjalanan ke suatu lokasi tambal ban yang dipilih.

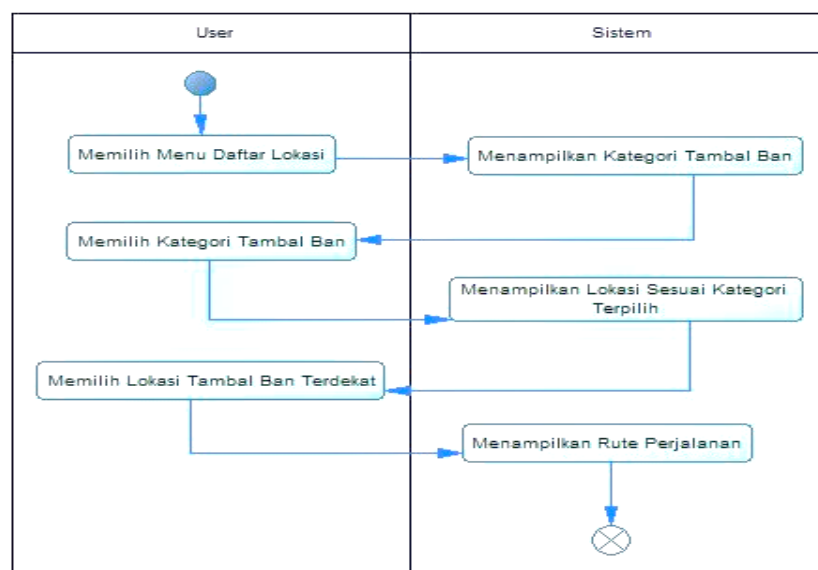
4.4.3 Perancangan Activity Diagram

Activity diagram menggambarkan bagaimana alur aktivitas dari sistem yang sedang dirancang. Gambar 4.4 merupakan *activity diagram* dari alur aktivitas antara admin dan sistem.

Gambar 4.4 menunjukkan *activity diagram* pada admin. Admin membuka halaman Admin kemudian melihat daftar lokasi pada halaman Admin lalu sistem menampilkan daftar lokasi yang akan di kelola oleh Admin seperti menambahkan data lokasi tambal ban yang baru, mengubah dan menghapus data lokasi tambal ban yang sudah ada sebelumnya. Untuk *activity diagram* User ditunjukkan pada Gambar 4.5.



Gambar 4.4 Activity Diagram Admin dan Sistem



Gambar 4.5 Activity Diagram User dan Sistem

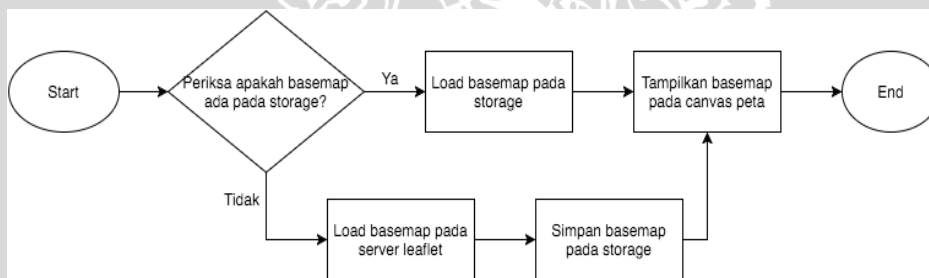
Gambar 4.5 menunjukkan *activity diagram* dari User. User dapat melihat daftar lokasi tambal ban berdasarkan kategori yang dipilih. User dapat melihat rute perjalanan setelah menentukan lokasi tambal ban terpilih baik lokasi tambal ban terdekat maupun yang lainnya.

4.4.4 Perancangan Penerapan Penggunaan Peta *Offline*

Perancangan penggunaan peta *offline* pada aplikasi LBS berbasis *website* studi kasus tambal ban di Kota Malang ini menggunakan peta OpenStreetMap yang didapat dari Leaflet. Leaflet adalah *open source* JavaScript untuk peta interaktif yang memiliki banyak fitur pemetaan untuk para pengembang.

Pada aplikasi penerapan peta *offline* ini menggunakan teknik *cache* yaitu untuk membuat peta dapat tetap diakses walau jaringan internet telah terputus. Pada dasarnya, teknik *cache* bertujuan untuk menampilkan sebuah halaman *website* lebih cepat di waktu berikutnya apabila *user* membuka kembali situs-situs *website* tersebut. Alasannya karena *file* yang terdapat pada halaman *website* bukan diambil dari *server* secara *online* tetapi diambil dari *cache browser* di komputer pengguna sehingga halaman *website* dapat tampil lebih cepat.

Dalam aplikasi penerapan peta *offline* ini sistem berupaya memerintahkan *browser* untuk melakukan *cache* pada peta dasar Leaflet. Tujuannya untuk dapat menyimpan seluruh gambar peta dasar Leaflet dengan tingkat skala berapapun yang pernah dilakukan oleh *user*. Sehingga ketika sambungan internet *user* telah di nonaktifkan maka *user* tetap dapat melakukan akses pada peta dasar Leaflet dengan skala yang sudah ada pada *cache*. Pada dasarnya, *flow* pada *cache* ini dapat dilihat pada Gambar 4.6.



Gambar 4.6 Flow Diagram Cache

Gambar 4.6 menunjukkan alur dari proses kerja *cache* pada peta *offline*. Terdapat proses dalam *cache* yaitu pengecekan *basemap* peta ,jika *basemap* sudah ada dalam *storage* maka peta akan langsung muncul dalam sebuah *canvas* tanpa koneksi internet, sebaliknya jika *basemap* tidak tersedia dalam *storage* maka harus mengambil *basemap* pada *server* Leaflet dengan koneksi internet. Adapun *event* untuk melakukan *cache* pada aplikasi penerapan peta *offline* menggunakan kode yang dapat dilihat pada Kode 4.1.

```

1  'use strict';
2  window.offlineMaps = {};
3
4  window.offlineMaps.eventManager = {
5
6    _events: {},
7
8    on: function (event, action) {
9
10     console.log('event.on: ' + event);
11     if (!(event in this._events)) {
12
13       this._events[event] = [];

```

```

14     }
15     this._events[event].push(action);
16     return this;
17
18 },
19
20 off: function (event) {
21     console.log('event.off: ' + event);
22     delete this._events[event];
23     return this;
24 },
25
26 fire: function (event) {
27     console.log('event.fire: ' + event);
28     var events = this._events;
29     if (event in events) {
30         var actions = events[event];
31         var args = Array.prototype.slice.call(arguments, 1);
32         for (var i = 0, l = actions.length; i < l; i++) {
33             var action = actions[i];
34             if (action instanceof Function) {
35                 action.apply(null, args);
36             } else {
37                 this.fire.apply(this, [action].concat(args));
38             }
39         }
40     }
41     return this;
42 }
43
44 };

```

Kode 4.1 Kode Event Cache

Sedangkan untuk inisialisasi pada Leaflet, menggunakan kode seperti pada Kode 4.2.

```

1  emr.on('mapLoad', function (storage) {
2
3      var map = L.map('map').setView([53.902254, 27.561850], 13);
4      new StorageTileLayer('http://{s}.tile.osm.org/{z}/{x}/{y}.png',
5      {storage: storage}).addTo(map);
6
7      emr.fire('mapLoaded'); });

```

Kode 4.2 Kode Inisialisasi Leaflet

Pada *storage* juga memiliki proses sendiri yang dapat dilihat pada Kode 4.3.

```

1  emr.on('storageLoad', function () {
2
3      var storage = getIndexedDBStorage() || getWebSqlStorage() || null;
4      if (!storage) {
5          emr.fire('storageLoaded', null);
6      }
7
8  }
9

```

10	});
11	

Kode 4.3 Kode Proses *Storage*

Pada awalnya pengguna tetap membutuhkan koneksi internet untuk mendapatkan lokasi tambal ban yang ada pada *database* aplikasi, setelah mendapatkan titik lokasi dari *database*, *user* dapat menggunakan melakukan skala pada peta tanpa butuh koneksi internet.

Keuntungannya adalah pengguna dapat menghemat kuota internet untuk melakukan *load* gambar peta dengan skala tertentu jika pengguna melakukan *zoom in* dan *zoom out* pada peta. Selain itu, pengaksesan *zoom in* dan *zoom out* juga akan lebih cepat dari biasanya.

4.4.5 Perancangan *Database*

Perancangan *database* pada sistem penerapan analisis peta *offline* pada aplikasi LBS berbasis web studi kasus pencarian lokasi tambal ban di Kota Malang ini berfungsi sebagai penyimpan semua data yang dibutuhkan seperti *longitude*, *latitude*, *place* dan *icon*.

Tabel *Location* berfungsi sebagai penyimpan data dari semua lokasi tambal ban di Kota Malang yang sudah di masukkan daftarnya ke dalam sistem oleh admin. Struktur tabel *location* dapat dilihat pada Tabel 4.5.

Tabel 4.5 Perancangan Tabel *Location*

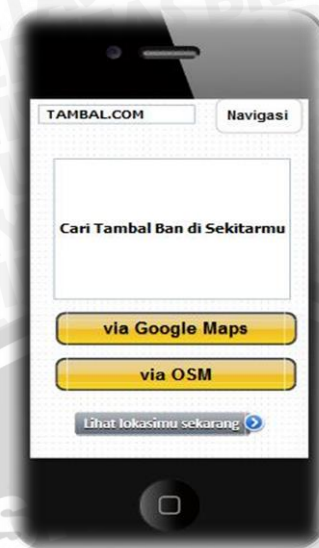
Field
Place
Icon
Title
Lon
Lat

4.4.6 Perancangan Antarmuka

Perancangan antarmuka berisi tentang perancangan dari antarmuka user. Dari perancangan ini nantinya akan diimplementasi menjadi sebuah sistem yang siap digunakan untuk di analisis.

1. Halaman Beranda User

Pada halaman beranda user terdapat menu-menu yang dapat di pilih untuk mengakses peta lokasi tambal ban terdekat dan mengetahui posisinya saat itu juga. Halaman beranda user ditunjukkan pada Gambar 4.7.



Gambar 4.7 Halaman Beranda User

2. Halaman Peta Open Street Map

Pada halaman peta OSM terdapat pilihan lokasi tambah ban yang bisa diatur oleh *user*. Untuk rute perjalanan *user* dapat melihat informasi perjalanan yang harus dilewati untuk menuju suatu lokasi tambah ban tertentu. Halaman peta OSM ditunjukkan pada Gambar 4.8.



Gambar 4.8 Halaman Peta OSM

3. Halaman *Contacts*

Halaman ini berfungsi untuk mengirimkan kritik dan saran kepada peneliti sehingga adanya pengembangan selanjutnya. Halaman *contacts* ditunjukkan pada Gambar 4.9.



Gambar 4.9 Halaman *Contacts*

4. Halaman Peta Google Maps

Pada halaman peta Google Maps kurang lebih sama seperti halaman Open Street Map tetapi yang membedakan yaitu peta Google Maps diakses secara penuh dengan koneksi *internet*. Halaman peta Google Maps ditunjukkan pada Gambar 4.10.



Gambar 4.10 Halaman Peta Google Maps

4.4.7 Perancangan Uji Coba

Dalam perancangan uji coba terdapat tiga parameter utama yang akan diuji. Pertama adalah pengujian berdasarkan ukuran file yang dihasilkan diantara kedua peta yaitu Google Maps yang berjalan *online* dan OpenStreet Map yang berjalan menggunakan *cache* sehingga sistem dapat berjalan *offline*. Kedua adalah pengujian berdasarkan parameter kecepatan akses diantara kedua peta. Ketiga

berdasarkan akurasi penempatan sebuah lokasi yang ditentukan oleh peta itu sendiri.

4.5 Implementasi Penggunaan Peta Offline

Pada sub bab ini dibahas mengenai implementasi perangkat lunak berdasarkan hasil yang telah didapatkan dari analisis kebutuhan dan proses perancangan perangkat lunak. Pembahasan terdiri dari penjelasan tentang spesifikasi sistem, batasan-batasan dalam implementasi, implementasi *database*, implementasi kode program, dan implementasi antarmuka perangkat lunak.

4.5.1 Spesifikasi Sistem

Hasil dari analisis kebutuhan dan perancangan perangkat lunak yang telah dijelaskan, menjadi acuan untuk melakukan implementasi menjadi sebuah aplikasi android yang dapat berfungsi sesuai dengan kebutuhan. Spesifikasi sistem diimplementasikan pada spesifikasi perangkat keras dan perangkat lunak.

4.5.1.1 Spesifikasi Perangkat Keras

Analisis performansi penerapan peta *offline* pada aplikasi *Location Based Service* berbasis *website* studi kasus tambal di Kota Malang ini menggunakan sebuah komputer dengan spesifikasi prosesor, *memory*, *harddisk*, *motherboard*, dan kartu grafis yang ditunjukkan pada Tabel 4.6 berikut.

Tabel 4.6 Spesifikasi Perangkat Keras Komputer

Nama Komponen	Spesifikasi
<i>System Model</i>	Asus N43SL
Prosesor	Intel(R) Core(TM) i5-2430M CPU @ 2.40GHz
Memori	8192 MB RAM
Grafis	Integrated Intel® HD Graphics 3000 NVIDIA® GeForce® GT 540M
<i>Display</i>	14 inc 16:9 HD(1366 x 768) LED Backlight

Adapun dalam proses instalasi dan pengujian, perangkat yang digunakan adalah *smartphone* android dengan spesifikasi perangkat keras yang ditunjukkan pada Tabel 4.7 berikut.

Tabel 4.7 Spesifikasi Perangkat Keras Smartphone

Nama Komponen	Spesifikasi
<i>System Model</i>	Samsung Galaxy Tab 7.0 Plus GT P6200
Prosesor	Dual-core 1.2 GHz Cortex-A9
Memori	microSD, up to 64 GB internal 16 GB

GPS	A-GPS, GLONASS
Display	IPS LCD capacitive touchscreen, 600 x 1024 pixels, 7.0 inc

4.5.1.2 Spesifikasi Perangkat Lunak

Analisis performansi penerapan peta *offline* pada aplikasi *Location Based Service* berbasis *website* studi kasus tambal di Kota Malang ini menggunakan perangkat lunak dengan spesifikasi yang ditunjukkan pada Tabel 4.8.

Tabel 4.8 Spesifikasi Perangkat Lunak Komputer

Nama Komponen	Spesifikasi
<i>Operating System</i>	Microsoft Windows 10
<i>Text Editor</i>	Sublime Text
Bahasa Pemrograman	JavaScript HTML CSS
<i>Database</i>	Json

Sedangkan dalam proses instalasi dan pengujian, perangkat yang digunakan adalah *smartphone* android dengan spesifikasi perangkat lunak yang ditunjukkan pada Tabel 4.9 berikut.

Tabel 4.9 Spesifikasi Perangkat Lunak Smartphone

Nama Komponen	Spesifikasi
<i>Operating System</i>	Android OS, v4.3 (Jelly Bean), Upgradable to v4.4.4 (KitKat), Planned upgrade to v5.0 (Lollipop)

4.5.2 Batasan-batasan Implementasi

Beberapa batasan dalam mengimplementasikan sistem adalah sebagai berikut :

1. Analisis performansi penerapan peta *offline* pada aplikasi *Location Based Service* berbasis *website* studi kasus tambal di Kota Malang ini dirancang untuk dapat berjalan pada perangkat *mobile* dengan sistem operasi Android, yaitu minimal versi 4.0.3 *Ice Cream Sandwich* (API 15) sampai dengan versi 4.4 *Kitkat* (API 19).
2. Aplikasi menggunakan peta OpenStreetMap yang didapat dari *offlinemapbox-leaflet*.
3. Peta *Offline* yang digunakan pada OpenStreetMap menggunakan teknologi *cache*.

4. Implementasi data jalan menyesuaikan data jalan yang tersedia pada OpenStreetMap.
6. Aplikasi dikembangkan dengan menggunakan bahasa pemrograman HTML dan JavaScript.
7. Komunikasi data antara aplikasi *client* yang berjalan pada perangkat *mobile* Android dengan basis data pada server menggunakan *web service*.
8. Data yang digunakan dalam sistem yaitu data JSON.

4.5.3 Implementasi Kode Program

Berdasarkan perancangan sistem yang telah dijelaskan sebelumnya, berikut ini merupakan implementasi kode program.

4.5.3.1 Implementasi Kode Program *Map* (Peta *Offline* OpenStreetMap)

Pada implementasi kode program *Map* terdapat beberapa *function* seperti *mapbox*, *value*, *cache*, dan *storage*. *Function* *mapbox* merupakan bagian dimana pemanggilan peta lokasi Kota Malang untuk ditampilkan dalam sistem sedangkan *function* *value* dan *cache* yaitu fungsi yang digunakan untuk menyimpan *cache* suatu lokasi yang telah ditentukan sehingga peta dapat bekerja secara *offline* tanpa koneksi *internet*.

```

1  (function (window, emr, mapbox, MM, undefined) {
2      var total = get_total();
3      var StorageRequestManager = function (storage) {
4          MM.RequestManager.apply(this, []);
5          this._storage = storage;
6      };
7
8      StorageRequestManager.prototype._imageToDataUri = function (image) {
9          var canvas = window.document.createElement('canvas');
10         canvas.width = image.width;
11         canvas.height = image.height;
12
13         var context = canvas.getContext('2d');
14         context.drawImage(image, 0, 0);
15
16         return canvas.toDataURL('image/png');
17     };
18
19     StorageRequestManager.prototype._createTileImage = function (id,
20 coord, value, cache) {
21     var img = window.document.createElement('img');
22     img.id = id;
23     img.style.position = 'absolute';
24     img.coord = coord;
25     this.loadingBay.appendChild(img);
26     if (cache) {
27         img.onload = this.getLoadCompleteWithCache();
28         img.crossOrigin = 'Anonymous';
29     } else {
30         img.onload = this.getLoadComplete();
31     }
32     img.onerror = this.getLoadComplete();
33     img.src = value;
34 };
35
36     StorageRequestManager.prototype._loadTile = function (id, coord, url)
37 {
38     var self = this;
39     if (this._storage) {
40         this._storage.get(id, function (value) {
41             if (value) {

```

```

42         self._createTileImage(id, coord, value, false);
43     } else {
44         self._createTileImage(id, coord, url, true);
45     }
46     }, function () {
47         self._createTileImage(id, coord, url, true);
48     });
49 } else {
50     self._createTileImage(id, coord, url, false);
51 }
52 };
53
54 StorageRequestManager.prototype.processQueue = function (sortFunc) {
55     if (sortFunc && this.requestQueue.length > 8) {
56         this.requestQueue.sort(sortFunc);
57     }
58     while (this.openRequestCount < this.maxOpenRequests &&
59 this.requestQueue.length > 0) {
60         var request = this.requestQueue.pop();
61         if (request) {
62             this.openRequestCount++;
63             this._loadTile(request.id, request.coord, request.url);
64             request = request.id = request.coord = request.url = null;
65         }
66     }
67 };
68
69 StorageRequestManager.prototype.getLoadCompleteWithCache = function ()
70 {
71     if (!this._loadComplete) {
72         var theManager = this;
73         this._loadComplete = function(e) {
74             e = e || window.event;
75
76             var img = e.srcElement || e.target;
77             img.onload = img.onerror = null;
78
79             if (theManager._storage) {
80                 theManager._storage.add(this.id,
81 theManager._imageToDataUri(this));
82             }
83
84             theManager.loadingBay.removeChild(img);
85             theManager.openRequestCount--;
86             delete theManager.requestsById[img.id];
87
88             if (e.type === 'load' && (img.complete ||
89 (img.readyState && img.readyState === 'complete')) {
90                 theManager.dispatchCallback('requestcomplete', img);
91             } else {
92                 theManager.dispatchCallback('requesterror', {
93                     element: img,
94                     url: ('' + img.src)
95                 });
96                 img.src = null;
97             }
98
99             setTimeout(theManager.getProcessQueue(), 0);
100         };
101     }
102     return this._loadComplete;
103 };
104
105 MM.extend(StorageRequestManager, MM.RequestManager);
106
107 var StorageLayer = function(provider, parent, name, storage) {
108     this.parent = parent || document.createElement('div');
109     this.parent.style.cssText = 'position: absolute; top: 0px; left:
110 0px;' +
111     'width: 100%; height: 100%; margin: 0; padding: 0; z-index:
112 0';
113     this.name = name;
114     this.levels = {};
115     this.requestManager = new StorageRequestManager(storage);

```

```

116     this.requestManager.addCallback('requestcomplete',
117     this.getTileComplete());
118     this.requestManager.addCallback('requesterror',
119     this.getTileError());
120     if (provider) {
121         this.setProvider(provider);
122     }
123 };
124
125     MM.extend(StorageLayer, MM.Layer);
126
127     var StorageTemplatedLayer = function(template, subdomains, name,
128     storage) {
129         return new StorageLayer(new MM.Template(template, subdomains),
130         null, name, storage);
131     };
132
133     emr.on('mapLoad', function (storage) {
134         var map = mapbox.map('map');
135         map.addLayer(new
136         StorageTemplatedLayer('http://{S}.tile.osm.org/{Z}/{X}/{Y}.png',
137         ['a', 'b', 'c'], undefined, storage));
138         map.ui.zoomer.add();
139         map.ui.zoombox.add();
140         map.centerzoom({lat: -7.9666204, lon: 112.6326321}, 14);
141
142
143         var markerLayer = mapbox.markers.layer();
144
145         // Replace marker factory function with our version
146         markerLayer.factory(function(m) {
147             // Create a marker using the simplestyle factory
148             var elem = mapbox.markers.simplestyle_factory(m);
149
150             // Add function that centers marker on click
151             // MM.addEvent(elem, 'click', function(e) {
152                 //     map.ease.location({
153                     //         lat: -7.9666204,
154                     //         lon: 112.6326321
155                     //     }).zoom(map.zoom()).optimal();
156             // });
157
158             return elem;
159         });

```

Kode 4.4 Implementasi Kode Program Map

4.5.3.2 Implementasi Kode Program *Longitude*, *Latitude* dan *Marker*

Pada implementasi kode program *marker* terdapat *variabel* seperti *lat*, *lon* dan *title* yang berguna untuk menandakan *Latitude* dan *Longitude* sebuah daerah lokasi tertentu. Terdapat juga fungsi *marker* yang berfungsi untuk memberikan tanda seperti warna dan *icon* pada suatu lokasi.

```

1 //MARKER
2
3 for(var i=0;i<=total;i++){
4     var lat = get_request_lat(i);
5     var lon = get_request_lon(i);
6     var title = get_request_title(i);
7     mapbox.markers.interaction(markerLayer);
8     map.addLayer(markerLayer);
9     markerLayer.add_feature({
10         geometry: {

```

```

17 // The order of coordinates here is lon, lat. This is because
18 // we use the GeoJSON specification for all marker features.
19 // (lon, lat is also the internal order of KML and other geographic formats)
20
21 coordinates: [lon, lat]
22
23 },
24
25 properties: {
26     // these properties customize the look of the marker
27     // see the simplestyle-spec for a full reference:
28     // https://github.com/mapbox/simplestyle-spec star-
29     stroked
30
31     'marker-color': '#28C9FA',
32     'marker-symbol': 'circle-stroked',
33
34     title: '',
35     description: title
36 }
37 });
38
39 }
40
41 //MARKER

```

Kode 4.5 Implementasi Kode Program *Longitude, Latitude* dan *Marker*

4.5.3.3 Implementasi Kode Program *Current Position*

Pada implementasi kode program *current position* terdapat *variabel* curlat dan curlon yang berguna untuk menentukan koordinat posisi yang sedang ditempati *user*. Fungsi *marker* yang terdapat dalam kode program *current position* berfungsi untuk membedakan tanda antara lokasi tambal ban dan lokasi sekarang.

```

1 //CURENT POSITION
2
3 setTimeout(function(){
4     var curlat = get_current_lat();
5     var curlon = get_current_lon();
6     console.log(curlat);
7     console.log(curlon);
8     mapbox.markers.interaction(markerLayer);
9     map.addLayer(markerLayer);
10    markerLayer.add_feature({
11        geometry: {
12            coordinates: [curlon, curlat]
13        },
14        properties: {
15            // these properties customize the look of the marker
16            // see the simplestyle-spec for a full reference:
17            // https://github.com/mapbox/simplestyle-spec star-
18            stroked
19
20            'marker-color': '#10E305',
21            'marker-symbol': 'star-stroked',
22

```

```

32         title: '',
33         description: 'Posisi Kamu Sekarang.'
34     }
35     });
36     }, 2000);
37     //END CURRENT POS
38

```

Kode 4.6 Implementasi Kode Program *Current Position*

4.5.3.4 Implementasi Kode Program Lokasi Terdekat

Pada implementasi kode program lokasi terdekat berfungsi untuk menampilkan suatu lokasi tambal yang terdekat dengan posisi *user* saat itu.

```

1  var latlng = new google.maps.LatLng( locations[ closest].lat,
2  locations[ closest].lon );
3
4  // Place a Google Marker at the closest location as the map center
5  // When you hover over the marker, it will display the title
6  for(var i = 0; i < locations.length; i++)
7  {
8
9      // get the distance between user's location and this point
10     var dist = Haversine( locations[ i ].lat, locations[ i ].lon,
11     lat, lon );
12     if(locations[ closest].lat != locations[i].lat && locations[
13     closest].lon != locations[i].lon){
14         var marker = new google.maps.Marker({
15             position: new
16             google.maps.LatLng(locations[i].lat, locations[i].lon),
17             icon:
18             'ASSET/images/'+locations[i].icon+'.png',
19             map: map
20         });
21         var content1 = locations[i].place+" . Distance is " +
22         dist + " km";
23         var latlngsend = new google.maps.LatLng( locations[
24         i ].lat, locations[ i ].lon );
25         addInfoWindow(marker, content1, latlngsend);
26     }
27 }
28
29 var marker2 = new google.maps.Marker( {
30     position: latlng,
31     map: map,
32     icon: 'ASSET/images/'+locations[closest].icon+'-near.png',
33     title: 'Closest Location to User: '+locations[closest].place+'
34     adn the distance is ' + mindist + ' km'
35 });
36
37 // Create an InfoWindow for the marker
38

```

```

54     var contentString = "" + "Tambal Ban Terdekat: " + locations[
55     closest ].place + ". Jarak " + mindist + " ";    // HTML text to display
56     in the InfoWindow
57
58     infowindow = new google.maps.InfoWindow( { content: contentString
59     } );
60
61
62     // Set event to display the InfoWindow anchored to the marker when
63     the marker is clicked.
64
65     Google.maps.event.addListener( marker2, 'click', function() {
66
67
68         infowindow.open( map, marker2 );
69         directionsDisplay.setMap(null);
70
71         var start = curentpos;
72         var end = latlng;
73
74         //var directionsDisplay = new
75         google.maps.DirectionsRenderer();// also, constructor can get
76         "DirectionsRendererOptions" object
77
78         directionsDisplay.setMap(map); // map should be already
79         initialized.
80
81
82         directionsDisplay.setPanel( document.getElementById( 'right-panel' ) );
83
84         var request = {
85             origin : start,
86             destination : end,
87             travelMode : google.maps.TravelMode.DRIVING
88         };
89
90         var directionsService = new
91         google.maps.DirectionsService();
92
93         directionsService.route(request, function(response,
94         status) {
95
96             if (status == google.maps.DirectionsStatus.OK) {
97                 directionsDisplay.setDirections(response);
98             }
99         });
100
101         });
102
103
104
105         map.setCenter( latlng );
106     }
107
108     // Convert Degress to Radians
109     function Deg2Rad( deg ) {
110
111         return deg * Math.PI / 180;
112     }
113
114
115
116
117     // Get Distance between two lat/lng points using the Haversine function
118     // First published by Roger Sinnott in Sky & Telescope magazine in 1984
119     ("Virtues of the Haversine")
120
121     //
122     function Haversine( lat1, lon1, lat2, lon2 )
123
124
125
126

```

```

127 {
    var R = 6372.8; // Earth Radius in Kilometers

    var dLat = Deg2Rad(lat2-lat1);
    var dLon = Deg2Rad(lon2-lon1);

    var a = Math.sin(dLat/2) * Math.sin(dLat/2) +
        *
        Math.cos(Deg2Rad(lat1)) * Math.cos(Deg2Rad(lat2))
        *
        Math.sin(dLon/2) * Math.sin(dLon/2);

    var c = 2 * Math.atan2(Math.sqrt(a), Math.sqrt(1-a));
    var d = R * c;

    // Return Distance in Kilometers
    return d;
}

// Call the method 'Init()' to display the google map when the web page
is displayed ( load event )

google.maps.event.addDomListener( window, 'load', Init );
function change(){
    document.getElementById("myForm").submit();
}

```

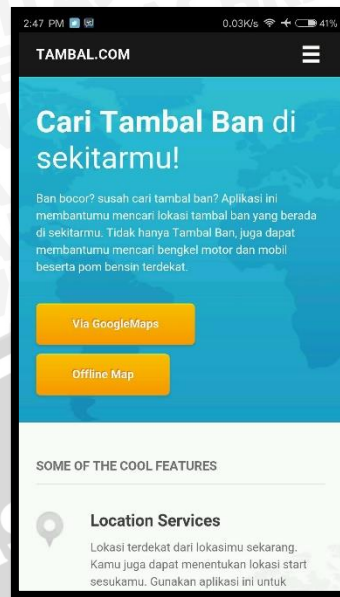
Kode 4.7 Implementasi Kode Lokasi Terdekat

4.5.4 Implementasi Antarmuka

Implementasi antarmuka ini terdiri dari perancangan antarmuka untuk aplikasi *user*. Seperti yang telah dirancang dalam perancangan terdiri dari tiga halaman yaitu halaman beranda *user*, halaman peta OSM, halaman *contacts*.

4.5.4.1 Implementasi Antarmuka Halaman Beranda *User*

Halaman beranda *user* ditunjukkan pada Gambar 4.11 berikut ini menunjukkan halaman utama dalam aplikasi *mobile website* pencarian lokasi tambal ban di Kota Malang.



Gambar 4.11 Halaman Beranda User

4.5.4.2 Implementasi Antarmuka Halaman Peta OSM (OpenStreetMap)

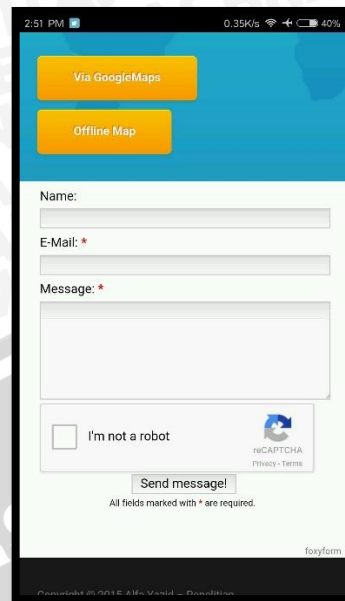
Implementasi untuk halaman peta OSM (OpenStreetMap) yang berisikan segala informasi dan rute perjalanan lokasi tambal ban terdekat ditunjukkan pada Gambar 4.12 berikut ini.



Gambar 4.12 Halaman OSM (OpenStreetMap)

4.5.4.3 Implementasi Antarmuka Halaman Contacts

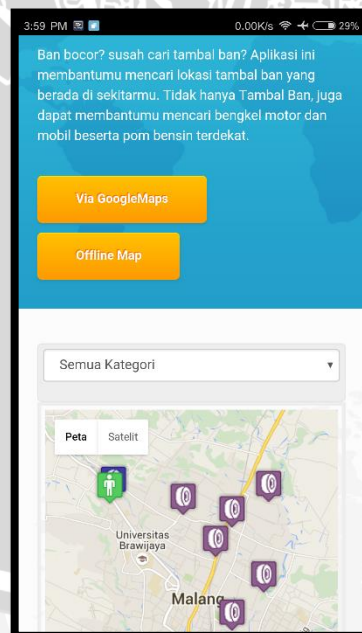
Implementasi untuk halaman *contacts* yang berisikan *form* untuk memberikan kritik dan saran bagi peneliti untuk pengembangan selanjutnya tentang analisis peta *offline* ditunjukkan pada Gambar 4.13.



Gambar 4.13 Halaman Contacts

4.5.4.4 Implementasi Antarmuka Halaman Google Maps

Halaman Google Maps yang ditunjukkan pada Gambar 4.14 berikut ini menunjukkan halaman pencarian lokasi tambal di Kota Malang yang menggunakan peta Google Maps.



Gambar 4.14 Halaman Google Maps

BAB 5 HASIL DAN PENGUJIAN

Pada bab ini dilakukan proses pengujian dan analisis terhadap aplikasi yang telah dibangun. Adapun tujuan dilakukan pengujian ini adalah untuk memastikan semua kebutuhan fungsional maupun non fungsional yang dirancang sebelumnya telah terpenuhi. Pengujian yang akan dilakukan meliputi pengujian validitas, *realibility* dan *response time*. Untuk menguji kebutuhan fungsional, maka dilakukan pengujian validasi dimana menggunakan metode pengujian *black-box*. Sedangkan untuk menguji kebutuhan non fungsional dilakukan pengujian *realibility* dan pengujian *response time*. Selanjutnya juga dilakukan pengujian berdasarkan dari tujuan analisis system seperti ukuran *file* dan kecepatan akses.

5.1 Pengujian Validasi

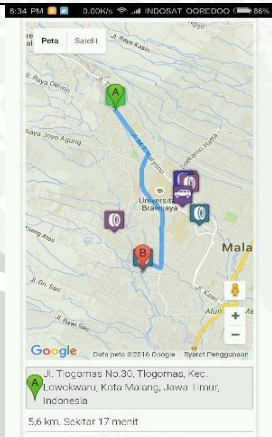
Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah benar sesuai dengan yang dibutuhkan. *Item-item* yang telah dirumuskan dalam daftar kebutuhan fungsional akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi ini menggunakan metode pengujian *black-box*, karena tidak diperlukan konsentrasi terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan konformitas antara kinerja sistem dengan daftar kebutuhan.

Pada skripsi ini dilakukan pengujian validasi terhadap aplikasi *mobile website* pencarian lokasi tambal ban di kota Malang untuk sistem operasi Android. Untuk mengetahui kesesuaian antara kebutuhan dengan kinerja sistem, pada setiap kebutuhan sistem dilakukan proses pengujian dengan kasus uji masing-masing yang ditunjukkan dalam tabel-tabel di bawah ini.

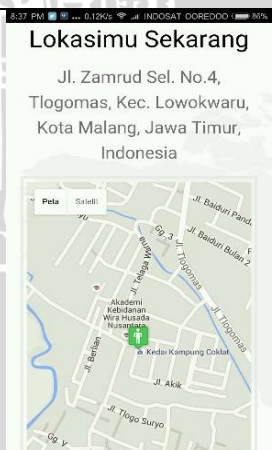
5.1.1 Kasus Uji Pengujian Validasi

Tabel 5.1 Kasus Uji Lihat Peta Tambal Ban Google Maps

Nama Kasus Uji	Lihat Lokasi Tambal Ban dengan Peta Google Maps
Objek Uji	Kebutuhan Fungsional
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi yang dibuat telah memenuhi kebutuhan fungsional untuk menyediakan fasilitas menampilkan lokasi Tambal Ban dengan peta Google Maps kepada user.
Prosedur Uji	1. Membuka aplikasi. 2. Membuka <i>slide menu</i>. 3. Memilih menu "Google Maps".
Hasil Yang Diharapkan	Aplikasi akan menampilkan halaman untuk melihat lokasi Tambal Ban dengan peta Google Maps.

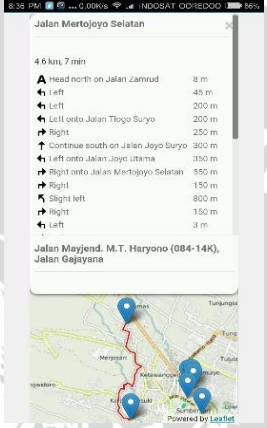
Hasil uji	
-----------	--

Tabel 5.2 Kasus Uji Lihat Lokasi Sekarang User

Nama Kasus Uji	Lihat Lokasi Sekarang User
Objek Uji	Kebutuhan Fungsional
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi yang dibuat telah memenuhi kebutuhan fungsional untuk menyediakan fasilitas menampilkan lokasi sekarang user.
Prosedur Uji	1. Membuka aplikasi. 2. Membuka <i>slide menu</i>. 3. Memilih menu "Lokasi Saya".
Hasil Yang Diharapkan	Aplikasi akan menampilkan halaman untuk melihat lokasi sekarang user.
Hasil uji	

Tabel 5.3 Kasus Uji Lihat Peta Tambal Ban Open Street Map

Nama Kasus Uji	Lihat Lokasi Tambal Ban dengan Peta Open Street Map
Objek Uji	Kebutuhan Fungsional

Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi yang dibuat telah memenuhi kebutuhan fungsional untuk menyediakan fasilitas menampilkan lokasi Tambal Ban dengan peta Open Street Map kepada user.
Prosedur Uji	1. Membuka aplikasi. 2. Membuka <i>slide menu</i>. 3. Memilih menu "<i>Offline Map</i>".
Hasil Yang Diharapkan	Aplikasi akan menampilkan halaman untuk melihat lokasi Tambal Ban dengan peta Open Street Map.
Hasil uji	

5.1.2 Hasil Uji Pengujian Validasi

Berdasarkan kasus uji yang telah dilaksanakan sesuai dengan prosedur pengujian validasi pada sub bab sebelumnya, maka didapatkan hasil seperti yang ditunjukkan pada tabel 5.4 berikut.

Tabel 5.4 Hasil Pengujian Validasi

No.	Nama Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan	Status Validitas
1.	Lihat Peta Tambal Ban Google Maps	Aplikasi dapat menampilkan halaman untuk melihat lokasi tambal ban dengan Google Maps kepada <i>user</i> .	Aplikasi menampilkan halaman untuk melihat lokasi tambal ban dengan Google Maps kepada <i>user</i> .	100% Valid
2.	Lihat Lokasi Sekarang <i>User</i>	Aplikasi dapat menampilkan halaman untuk melihat peta lokasi sekarang <i>user</i> .	Aplikasi menampilkan halaman untuk melihat lokasi sekarang <i>user</i> .	100% Valid
3.	Lihat Peta Tambal Ban Open Street Map	Aplikasi dapat menampilkan halaman untuk	Aplikasi menampilkan halaman untuk melihat lokasi tambal ban	100% Valid

		melihat lokasi tambal ban dengan Open Street Map kepada user.	dengan Open Street Map kepada user.	
--	--	---	-------------------------------------	--

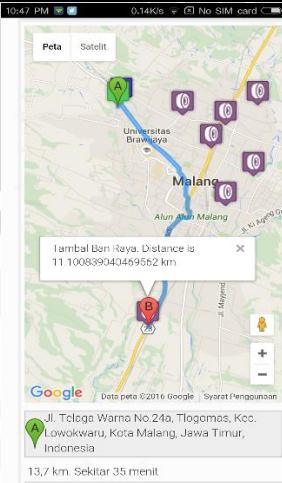
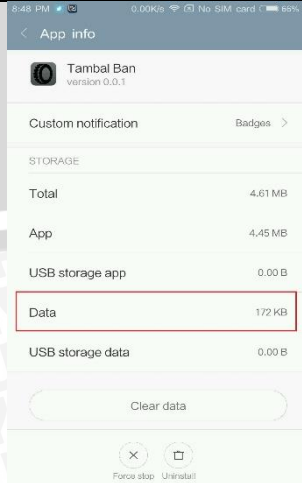
5.2 Pengujian Non Fungsional

Tahap pengujian non fungsional ini akan menjelaskan mengenai pengujian aplikasi pencarian lokasi tambal ban Kota Malang berdasarkan kebutuhan non fungsional yang dijelaskan pada tahap perancangan. Pengujian ini dilakukan untuk mengetahui hasil dari analisis seperti ukuran *file* dan kecepatan akses dari kedua peta yaitu Google Maps dan Open Street Map.

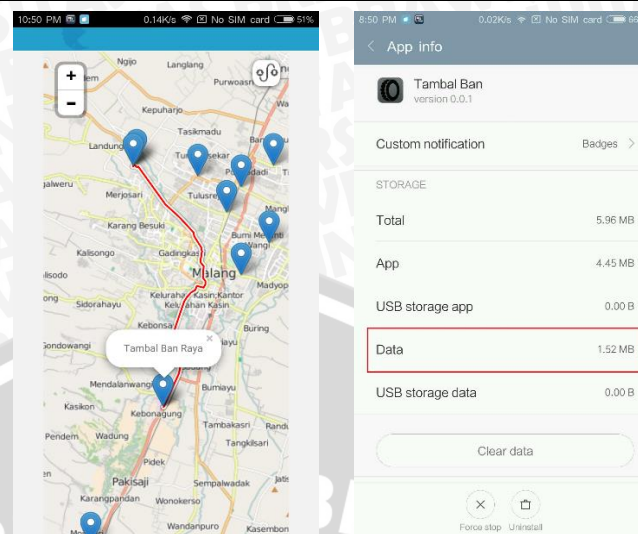
5.2.1 Pengujian Ukuran File

Aplikasi pencarian lokasi tambal ban yang memanfaatkan layanan LBS ini menggunakan dua peta sebagai media pencarian lokasinya yaitu Google Maps dan Open Street Map. Namun pada bagian ini akan dilakukan pengujian untuk membandingkan ukuran *file* yang dihasilkan antara peta Google Maps dan Open Street Map yang menggunakan teknologi *cache*, dengan cara mengambil sebuah data sampel proses pencarian sebuah lokasi tambal ban. Tabel 5.5 menunjukkan hasil pengujian, dimana dilakukan sebuah pencarian lokasi tambal ban dengan letak asal dan tujuan yang sama.

Tabel 5.5 Hasil Pengujian Ukuran File

Nama Kasus Uji	Pengujian Ukuran File
Objek Uji	Kebutuhan Non Fungsional
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui perbedaan ukuran <i>file</i> yang dihasilkan dari kedua peta yaitu Google Maps dan Open Street Map.
Prosedur Uji	1. Memilih satu lokasi tambal ban dari lokasi asal dan tujuan yang sama yaitu Tambal Ban Raya. 2. Membuka <i>tools task manager apps</i> dari <i>smartphone</i> .
Hasil Uji Ukuran File (Google Maps)	 

Hasil Uji Ukuran File
(Open Street Map)



5.2.2 Pengujian Kecepatan Akses

Pengujian kecepatan akses adalah pengujian yang dilakukan untuk menilai tingkat kecepatan akses dari kedua peta, antara Google Maps dan Open Street Map yang menggunakan teknologi *cache*. Pengujian kecepatan akses ini menggunakan fasilitas dari *website* <http://www.websiteoptimization.com/> yang berguna untuk menganalisa suatu halaman dari sebuah *website* baik dari kecepatan akses maupun performa *website* yang di analisis. Hasil dari pengujian kecepatan akses antara kedua peta ditunjukkan pada Tabel 5.6.

Tabel 5.6 Hasil Pengujian Kecepatan Akses

Nama Kasus Uji	Pengujian Kecepatan Akses	
Objek Uji	Kebutuhan Non Fungsional	
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui kecepatan akses dari kedua peta yaitu Google Maps dan Open Street Map.	
Prosedur Uji	Upload konten kedua halaman peta Google Maps dan Open Street Map ke http://www.websiteoptimization.com/	
Hasil Uji Kecepatan Akses (Google Maps)		

Hasil Uji Kecepatan Akses (Open Street Map)	Connection Rate		Download Time	
	14.4 Kbps		327.74 seconds	
	28.8 Kbps		165.37 seconds	
	33.6 Kbps		142.17 seconds	
	56 Kbps		86.50 seconds	
	128 Kbps		28.57 seconds	
	1.44 Mbps		5.22 seconds	

5.2.3 Pengujian Kecepatan Akses Peta *Offline* dengan *cache*

Pengujian kecepatan akses peta *offline* dengan *cache* adalah pengujian yang dilakukan untuk mengukur kecepatan akses dari halaman peta Open Street Map yang sudah menggunakan *cache*. Pengujian kecepatan akses ini juga menggunakan fasilitas dari website <http://www.websiteoptimization.com/>. Hasil dari pengujian kecepatan akses peta *offline* yang menggunakan *cache* ditunjukkan pada Tabel 5.7.

Tabel 5.7 Hasil Pengujian Kecepatan Akses Peta *Offline* dengan *cache*

Nama Kasus Uji	Pengujian Kecepatan Akses Peta <i>Offline</i> dengan <i>cache</i>		
Objek Uji	Kebutuhan Non Fungsional		
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui kecepatan akses dari peta yang menggunakan <i>cache</i> .		
Prosedur Uji	<i>Upload</i> konten peta Open Street Map yang menggunakan teknologi <i>cache</i> http://www.websiteoptimization.com/		
Hasil Uji Kecepatan Akses (Open Street Map <i>Offline</i>)			
	Connection Rate		Download Time
	14.4 Kbps	49.67 seconds	
	28.8 Kbps	26.43 seconds	
	33.6 Kbps	23.12 seconds	
	56 Kbps	15.15 seconds	
	128 Kbps	6.86 seconds	
	1.44 Mbps	1.52 seconds	

5.3 Analisis Hasil Pengujian

Proses analisis terhadap hasil pengujian dilakukan untuk mendapatkan kesimpulan dari hasil pengujian aplikasi *mobile website* pencarian lokasi tambalan yang telah dibuat. Proses analisis mengacu pada hasil pengujian yang

didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahapan pengujian. Proses analisis yang dilakukan meliputi hasil pengujian validasi dan analisis hasil pengujian non-fungsional.

5.3.1 Analisis Hasil Pengujian Validasi

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kecocokan antara hasil kinerja sistem dengan daftar kebutuhan. Berdasarkan hasil pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas aplikasi *mobile website* pencarian lokasi tambal ban yang dibangun telah memenuhi kebutuhan fungsional yang dijabarkan pada tahap analisis kebutuhan.

5.3.2 Analisis Hasil Pengujian Ukuran File

Proses analisis terhadap hasil pengujian ukuran file dilakukan dengan melihat hasil dari pengujian yang didapatkan. Berdasarkan hasil pengujian ukuran *file* yang dilakukan didapatkan bahwa aplikasi pencarian lokasi tambal ban yang dibuat dengan menggunakan dua peta berbeda yaitu Google Maps dan Open Street Map menghasilkan ukuran *file* yang cukup signifikan. Dari hasil pengujian yang dilakukan dengan cara menjalankan aplikasi, kemudian memilih satu lokasi tambal ban dengan lokasi asal dan tujuan yang sama. Peta yang menggunakan Google Maps menghasilkan ukuran *file* hanya sebesar 172 Kb sedangkan peta yang menggunakan Open Street Map menghasilkan ukuran *file* 1.52 Mb.

Berdasarkan hasil tersebut dapat disimpulkan bahwa peta yang menggunakan Open Street Map memiliki *file* yang lebih besar karena peta Open Street Map tersebut dibuat menggunakan teknologi *cache* sehingga ketika pertama kali peta diakses menggunakan koneksi *internet* maka peta yang dilihat akan di *download* kemudian disimpan dalam *storage smartphone*. Setelah peta tersimpan dalam skala tertentu, maka saat lokasi yang telah tersimpan dibuka kembali tidak membutuhkan koneksi *internet* sama sekali kecuali skala peta tertentu yang dilihat belum terdapat dalam *storage*. Sebaliknya peta yang menggunakan Google Maps memiliki ukuran *file* yang relatif cukup kecil dibandingkan peta Open Street Map, karena cara kerja dari peta Google Maps yang digunakan sama seperti layaknya aplikasi *maps* yang digunakan oleh banyak orang dan membutuhkan layanan penuh koneksi *internet*.

5.3.3 Analisis Hasil Pengujian Kecepatan Akses

Proses analisis terhadap hasil pengujian kecepatan akses dilakukan dengan menggunakan *tools* dari *website* <http://www.websiteoptimization.com/> yang dapat memeriksa kecepatan suatu halaman *website*, baik berdasarkan waktu dalam mengakses sebuah halaman *website* dan menganalisis *error* dari bahasa pemrograman yang digunakan dalam *website* yang diperiksa. Merujuk dari satu sampel hasil pengukuran yang didapatkan, maka kecepatan akses halaman peta yang menggunakan Google Maps jika kecepatan koneksi internetnya 1.44 Mb maka membutuhkan waktu untuk membuka halaman tersebut adalah 2.6 detik sedangkan peta yang menggunakan Open Street Map dengan kecepatan koneksi

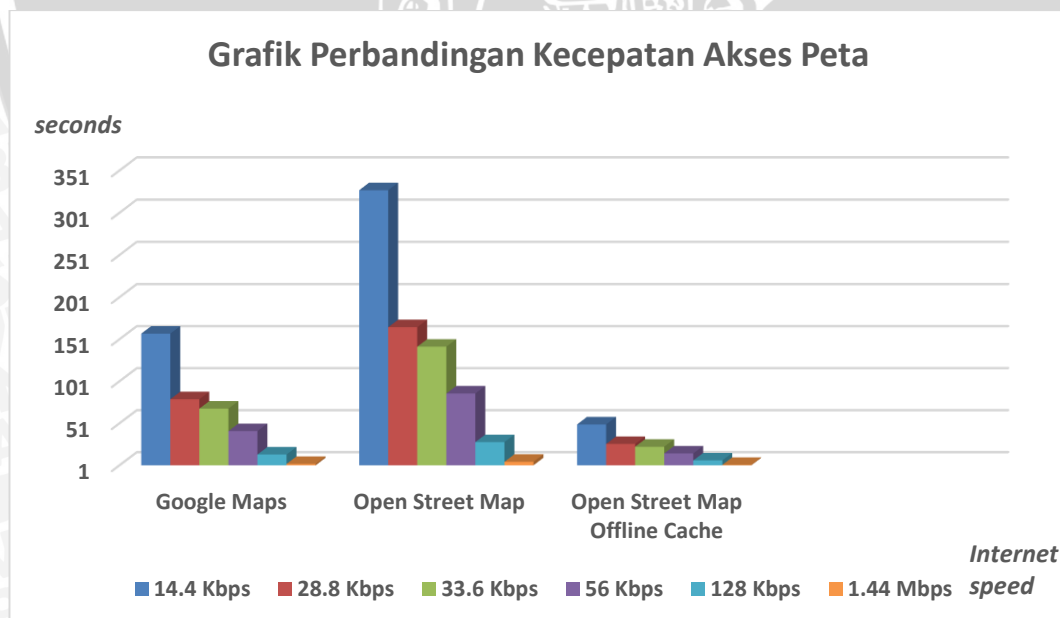
internet yang sama yaitu 1.44 Mb membutuhkan waktu untuk membuka halaman tersebut yaitu selama 5.22 detik.

Hasil dari pengukuran ini juga terkait menjelaskan sesuai apa yang dijelaskan dalam pengujian ukuran *file*, karena semakin besar ukuran *file* dari sebuah halaman *website* maka membutuhkan pula waktu yang cukup lama untuk dapat mengakses halaman tersebut. Tetapi kelebihan peta Open Street Map yang menggunakan teknologi *cache* akan jauh lebih cepat di akses jika petanya sudah pernah diakses, karena peta tersebut sudah tersimpan dalam *storage smartphone* meskipun koneksi *internet* telah terputus.

5.3.4 Analisis Hasil Pengujian Kecepatan Akses Peta *Offline* dengan *cache*

Proses analisis hasil pengujian kecepatan akses peta *offline* dengan *cache* sama seperti menguji kecepatan kedua peta antara Google Maps dan Open Street Map yaitu dengan menggunakan *tools* dari *website* <http://www.websiteoptimization.com/>. Berdasarkan hasil pengujian yang didapatkan, halaman peta *offline* yang menggunakan *cache* hanya membutuhkan waktu yang sangat singkat seperti data yang didapat jika kecepatan internet sebesar 1.44 Mbps maka halaman peta *offline* hanya membutuhkan waktu akses sekitar 1.52 detik.

Hasil dari pengujian ini juga menunjukkan bahwa peta yang menggunakan *cache* hanya membutuhkan waktu yang singkat tetapi dengan kondisi peta halaman yang diakses sudah pernah dibuka sebelumnya sesuai skala yang telah ditentukan dan peta tersebut akan tersimpan dalam *memory smartphone* sehingga halaman peta yang diakses akan dengan cepat terbuka. Grafik dari hasil ketiga pengujian peta dapat dilihat pada Gambar 5.1.



Gambar 5.1 Grafik Perbandingan Kecepatan Akses

Dari grafik pada Gambar 5.1 menunjukkan bahwa peta Open Street Map saat pertama kali di akses memiliki rataan waktu yang sangat tinggi karena skala peta yang sudah diakses kemudian di *download* terlebih dahulu ke dalam *memory smartphone*. Pada data peta Open Street Map yang sudah tersimpan dalam *memory smartphone* terlihat bahwa halaman peta yang di akses hanya membutuhkan rataan waktu yang cukup singkat. Sedangkan halaman peta yang menggunakan Google Maps dapat diakses hanya dengan koneksi *internet*.



BAB 6 PENUTUP

6.1 Kesimpulan

Berdasarkan hasil perancangan, implementasi dan pengujian yang telah dilakukan maka dapat diambil kesimpulan sebagai berikut :

1. Perancangan aplikasi *mobile website* pencarian lokasi tambal ban di Kota Malang yang memanfaatkan layanan LBS (*Location Based Service*) ini telah dibuat sesuai dengan spesifikasi kebutuhan yang telah dianalisis dengan menggunakan pemodelan UML (*Unified Modeling Language*).
2. Implementasi dari peta *offline* yang dirancang menggunakan teknologi *cache* telah berjalan dengan baik berdasarkan hasil pengujian validasi dengan validitas 100%.
3. Implementasi dari kebutuhan non-fungsional realibilitas telah dipenuhi berdasarkan hasil pengujian ukuran *file* dimana peta yang menggunakan Google Maps ukuran *file* nya jauh lebih kecil dibandingkan peta yang menggunakan OpenStreetMap dengan teknologi *cache*.
4. Berdasarkan data hasil uji coba ukuran *file* maka kecepatan akses peta yang di uji juga akan sangat mempengaruhi sehingga halaman peta yang menggunakan Google Maps akan lebih cepat diakses dibandingkan halaman peta yang menggunakan Open Street Map.
5. Kelebihan peta Open Street Map yang menggunakan teknologi *cache* meskipun lebih lama waktu mengakses, tetapi halamannya akan jauh lebih cepat diakses ulang jika peta yang diakses sudah pernah dibuka sesuai skalanya tanpa membutuhkan koneksi *internet*.

6.2 Saran

Saran yang dapat diberikan untuk aplikasi analisis penerapan peta *offline* pencarian lokasi tambal ban di Kota Malang ini antara lain :

1. Dapat dilakukan pengembangan yang menggunakan peta yang sepenuhnya diakses *offline* tanpa membutuhkan koneksi *internet* sehingga lebih dapat bermanfaat dan digunakan oleh banyak *user*.
2. Melakukan *routing* lokasi secara *online* pada aplikasi agar dapat dilakukan pengujian akurasi penempatan sebuah lokasi.

DAFTAR PUSTAKA

Abidin, H.Z. 2007 . Penentuan Posisi Dengan GPS dan Aplikasinya. *Third edition*. ISBN 978-979-408-377-2. 398 pp. P.T. Pradnya Paramita, Jakarta.

Surya Wulan Riyanti. 2012. Manfaat Dalam Penggunaan Tambal Ban. Wulanliyanti.blogspot.com/2012/manfaat-dalam-penggunaan-atm.html

Steiniger, Stefan; Moritz Neuni; dan Alistair Edwardes. (2006). *Foundations of Location Based Services*.

Virrantaus, K; Markkula, J; Garmash, A; Terziyan, Y. 2001 . *Location Based Service*

Sommerville, Ian. 2011. *Software Engineering* (Rekayasa Perangkat Lunak). Jakarta: Erlangga.

Dharwiyanti, Sri dan Wahono, Satria. 2003 . Pengantar *Unified Modelling Language* (UML). Dikutip dari <http://ilmukomputer.com> diakses pada 23 Juli 2014.

Json. 2014. Pengenalan JSON. <http://www.json.org/json-id.html> diakses pada 11 Juli 2014.

Sutanta, E. 2012 . Kebutuhan *Web Service* untuk Sinkronisasi Data Antar Sistem Informasi dalam *E-Gov* di Pemkab Bantul Yogyakarta. JURTIK - STMIK BANDUNG.

Mukhlisin, H. 2013 . Kupas Tuntas *HTML*. Ilmu Komputer.

Open Street Map. 2011. Apa itu Open Street Map. <http://openstreetmap.id/about/tentang-openstreetmap/> diakses pada 14 Agustus 2016.

Guru Otomotif. 2016. Pengertian Tambal Ban Tubeless, Ban Biasa, dan Ban Radial. <http://www.guruotomotif.com/2016/03/pengertian-ban-tubeless-ban-biasa-radial-dan.html> diakses pada 14 Agustus 2016.