RANCANG BANGUN APLIKASI *MOBILE* VISUALISASI JALUR PENDAKIAN GUNUNG MENGGUNAKAN LUNGO JS DAN APACHE CORDOVA

SKRIPSI

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun Oleh :

Moh Sofa

105090607111020

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2014

LEMBAR PERSETUJUAN

RANCANG BANGUN APLIKASI *MOBILE*
VISUALISASI JALUR PENDAKIAN GUNUNG MENGGUNAKAN
LUNGO JS DAN APACHE CORDOVA

SKRIPSI

LABORATORIUM MOBILE

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun Oleh :

Moh Sofa

105090607111020

Telah diperiksa dan disetujui oleh dosen pembimbing

Pada tanggal 16 Juni 2014:

Dosen Pembimbing I	Dosen Pembimbing II
Aryo Pinandito, S.T., M.MT.	**Agi Putra Kharisma, S.T., M.T.**
NIK. 83051916110374	**NIK. -**

LEMBAR PENGESAHAN

RANCANG BANGUN APLIKASI *MOBILE*
VISUALISASI JALUR PENDAKIAN GUNUNG MENGGUNAKAN
LUNGO JS DAN APACHE CORDOVA

SKRIPSI

LABORATORIUM MOBILE

Untuk memenuhi sebagian persyaratan untuk mencapai gelar Sarjana Komputer

Disusun Oleh :

Moh Sofa
NIM. 105090607111020

Setelah dipertahankan di depan Majelis Penguji
pada tanggal 2 Juli 2014
dan dinyatakan memenuhi syarat untuk memperoleh gelar Sarjana dalam bidang Ilmu Komputer

Penguji I	Penguji II
Fajar Pradana, S.ST,.M.Eng	**Denny Sagita Rusdianto, S.Kom., M.Kom**
NIK. 87112116110371	**NIK.85112406110250**

Penguji III

Wibisono Sukmo Wardhono, ST,.MT
NIK. 82040406110091

Mengetahui
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, MT.
NIP. 196708011992031001

LEMBAR PERNYATAAN

Saya yang bertanda tangan di bawah ini :

Nama	:	Moh Sofa
NIM	:	105090607111020
Program Studi	:	Informatika / Ilmu Komputer
Jurusan	:	Ilmu Komputer
Fakultas	:	Program Teknologi Informasi Ilmu Komputer
Penulis skripsi berjudul	:	Rancang Bangun Aplikasi *Mobile* Visualisasi Jalur Pendakian Gunung Menggunakan Lungo Js Dan Apache Cordova

Dengan ini menyatakan bahwa :

1. Isi dari skripsi yang saya buat adalah benar-benar karya sendiri dan tidak menjiplak karya orang lain, selain nama-nama yang termaktub di isi dan tertulis di daftar pustaka skripsi ini.
2. Apabila di kemudian hari ternyata skripsi yang saya tulis terbukti hasil jiplakan, maka saya bersedia menanggung segala resiko yang akan saya terima.

Demikian pernyataan ini dibuat dengan segala kesadaran dan penuh tanggung jawab dan digunakan sebagaimana mestinya.

Malang, 16 Juni 2014

Yang menyatakan,

Moh Sofa

NIM. 105090607111020

KATA PENGANTAR

Puji syukur penulis panjatkan kehadirat Allah SWT, atas segala rahmat dan karunia-Nya, penulis dapat menyelesaikan skripsi dengan judul "Rancang Bangun Aplikasi *Mobile* Visualisasi Jalur Pendakian Gunung Menggunakan Lungo Js dan Apache Cordova". Skripsi ini merupakan salah satu syarat untuk memenuhi persyaratan akademis menyelesaikan studi di program Sarjana Ilmu Komputer Universitas Brawijaya. Selama melaksanakan skripsi ini, penulis mendapat bantuan dan dukungan dari banyak pihak. Untuk itu, penulis ingin mengucapkan terima kasih kepada :

1. Ayahanda Mukhtar Fauzi, Ibunda Ima atas segala nasehat, kasih sayang, perhatian dan kesabarannya di dalam membesarkan dan mendidik penulis, serta yang senantiasa tiada henti – hentinya memberikan doa dan semangat demi terselesaikannya skripsi ini.
2. Aryo Pinandito, ST., M.MT selaku dosen pembimbing I dan Agi Putra Kharisma, S.T., M.T sebagai dosen pembimbing II yang telah bijaksana dan sabar dalam membimbing dan menyalurkan ilmu kepada penulis serta semua waktu dan nasehat yang telah diberikan dalam proses penyelesaian skripsi ini.
3. Bapak Ir. Sutrisno, M.T, Bapak Ir. Heru Nurwasito, M.Kom, Bapak Himawat Aryadita, S.T, M.Sc, dan Bapak Eddy Santoso, S.Kom selaku Ketua, Wakil Ketua 1, Wakil Ketua 2 dan Wakil Ketua 3 Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya.
4. Bapak Drs. Marji, M.T dan Bapak Issa Arwani, S.Kom, M.Sc selaku Ketua dan Sekretaris Program Studi Informatika/Ilmu Komputer Universitas Brawijaya.
5. Seluruh Dosen Informatika/Ilmu Komputer Universitas Brawijaya atas kesediaan membagi ilmunya kepada penulis.
6. Seluruh Civitas Akademika Informatika/Ilmu Komputer Universitas Brawijaya yang telah banyak memberi bantuan dan dukungan selama

penulis menempuh studi di PTIIK Universitas Brawijaya dan selama penyelesaian skripsi ini.

7. Keluarga besar penulis, Kakek Sutomo, Kakak Abdullah Mukhtar, Cholila yang selalu memberi dorongan, semangat dan motivasi untuk keberhasilan penulis dalam menyelesaikan perkuliahan.
8. Teman-temanku Bayu, Afrizal, Ras Fahmi, Ine, Fira yang telah menjadi keluarga mini selama penulis menempuh studi di universitas brawijaya, you are everything brooh.
9. Teman-teman pendaki amatir Taufik, Silvi, Tia, Rendy, Kebo dan anak MANDAPALA, nikmati keEPICan negeri ini teman dan saya tunggu kabar kelulusan kalian.
10. Teman-teman Unit Kegiatan Mahasiswa Tegazs, UABT yang memberikan bumbu manis pengalaman serta canda tawa selama penulis menempuh studi di Universitas Brawijaya.
11. Teman-teman seperjuangan Angkatan 2010 Ilmu Komputer dan Informatika, terimakasih atas segala bantuannya selama menempuh studi di Teknik Informatika Universitas Brawijaya.
12. Semua pihak yang tidak dapat penulis sebutkan satu per satu yang terlibat baik secara langsung maupun yang tidak langsung demi terselesaikannya skripsi ini.

Hanya doa yang bisa penulis berikan semoga Allah SWT memberikan pahala serta balasan kebaikan yang berlipat. Penulis menyadari bahwa skripsi ini masih banyak kekurangan dan masih jauh dari sempurna. Untuk itu, saran dan kritik yang membangun sangat penulis harapkan. Semoga skripsi ini membawa manfaat bagi penyusun maupun pihak lain yang menggunakannya.

Malang, 7 Juli 2014

Penulis

ABSTRAK

Moh Sofa. 2014. Rancang Bangun Aplikasi Mobile Visualisasi Jalur Pendakian Gunung Menggunakan Lungo Js dan Apache Cordova. Skripsi Program Studi Informatika/Ilmu Komputer, Program Teknologi Informasi dan Komputer, Universitas Brawijaya.

Pembimbing: Aryo Pinandito, ST., M.MT dan Agi Putra Kharisma, S.T., M.T.

Mendaki gunung merupakan kegiatan kombinasi antara olahraga berjalan dengan rekreasi. Dalam kegiatan mendaki gunung banyak tantangan dan masalah yang harus dihadapi oleh pendaki, salah satu masalah serius yang dihadapi pendaki adalah tersesat. Sebagai solusi terhadap masalah tersebut yaitu dengan membangun sebuah aplikasi *mobile* yang dapat memvisualisasikan jalur pendakian dengan memanfaatkan GPS (Global Positioning System) dan kompas digital yang terdapat pada *smartphone*. Saat ini terdapat berbagai *platform* yang digunakan pada perangkat *mobile* seperti Android, iOS, Windows Phone dan lain lain. Perbedaan *platform* akan menyulitkan dalam pengembangan aplikasi yang dapat berjalan pada semua *platform*. Lungo Js yang merupakan *mobile framework* menggunakan fitur terbaru dari HTML 5 dan CSS 3 bisa menjadi pilihan untuk membuat aplikasi *mobile cross platform*. Dengan kebutuhan aplikasi untuk mengakses fitur *native* seperti *Geolocation*, *Storage* dan *Compass* maka digunakan Apache Cordova. Dari hasil pengujian dapat disimpulkan bahwa Apache Cordova dapat mengakses *hardware smartphone* yang dibutuhkan pada aplikasi visualiasi jalur pendakian dengan javascript dan implementasi Lungo Js yang menjadi antarmuka sebuah aplikasi *native* telah memenuhi kebutuhan dan fungsionalitas yang telah dirancang sebelumnya. Pada pengujian perhitungan manual dan aplikasi dapat disimpulkan bahwa implementasi formula Haversine dan *Forward* Azimuth yang digunakan untuk menghitung informasi jarak dan penunjuk arah telah sesuai dengan perhitungan yang dilakukan dengan manual.

Kata kunci: Jalur Pendakian, Lungo Js, Apache Cordova, Haversine, *Forward* Azimuth

ABSTRACT

Moh Sofa. 2014. ***Developing The Building Of Mobile Application In Hiking Traffic Lane Visualization Used Lungo Js And Apache Cordova*****.** ***Adviser*** **: Aryo Pinandito, ST., M.MT dan Agi Putra Kharisma, S.T., M.T.**

Hiking is a combination activity between tracking sport and recreation. In hiking activity there are a lot of challenges and problems should be faced by the mountain hiker, one of the serious matters is loosing the way. The solution of that problem is building a mobile application that is able to visualize the traffic lane of hiking by GPS utilization and digital compass in smarthphone. Nowadays, there are some platforms used in mobile sets of equipment such as Android, iOS, Windows Phone etc. The difference of platform will complicate in developing application that is able to be operated in all of the platforms. Lungo Js as the mobile framework uses the newest feature of HTML 5 and CSS 3 that can become the options to make a mobile cross platform application. The requirement of application to acces the native feature, such as; Geolocation, Storage and Compass, therefore it is used Apache Cordova can access the hardware smartphone that is needed on the visual traffic lane hiking application by javascript and the implementation of lungo as the interface of a native application that have complied the requirement and functionality that have been designed before. On that manual computation trial and application can be concluded that the formula implementation of haversine and forward azimuth that used to calculate the distance in formation and instruction of direction that is appropriate with the calculation used manually.

Key Terms : Hiking Traffic Lane, Lungo Js, Apache Cordova, Haversine, Forward Azimuth

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR LAMPIRAN

BAB I

PENDAHULUAN

1.1 Latar Belakang

Mendaki gunung merupakan kegiatan kombinasi antara olahraga berjalan dengan rekreasi. Berjalan saat mendaki gunung memiliki teknik tersendiri, karena pendaki harus berjalan dengan memikul beban dipunggung, melintasi lembah, mendaki tebing, menuruni lereng-lereng atau meniti punggungan-punggungan kecil yang tipis. Dalam kegiatan mendaki gunung banyak tantangan dan masalah yang harus dihadapi oleh pendaki, salah satu masalah serius yang dihadapi pendaki adalah tersesat. Contoh kejadian tersesatnya pendaki pada akhir tahun 2013 terjadi digunung tertinggi pulau Jawa yaitu Gunung Semeru, dua pendaki yang berasal dari perguruan tinggi diJakarta hilang selama lima hari. Kemudian satu pendaki ditemukan di *blank* 75 arah Tawon Songo dengan kondisi patah kaki dan satu pendaki lainya ditemukan di Gunung Buto yang lokasinya melenceng 4 jam dari Tawon Songo [TAU-2013].

Pendaki yang tersesat akan kesulitan untuk mengetahui jalur kembali yang benar. Sebagai solusi terhadap masalah tersebut yaitu dengan membangun sebuah aplikasi *mobile* yang dapat memvisualisasikan jalur pendakian dengan memanfaatkan GPS (*Global Positioning System*) dan kompas digital yang terdapat pada *smartphone*. Dengan menggunakan perangkat *mobile* yang penggunaannya praktis serta memanfaatkan GPS sebagai penanda jalur pendakian serta kompas digital untuk penunjuk arah berdasarkan koordinat GPS, maka aplikasi *mobile* ini diharapkan dapat membantu pendaki dalam menentukan arah kembali yang benar sehingga menghindari terjadinya pendaki tersesat.

Perhitungan untuk menentukan arah serta informasi jarak berdasarkan kordinat *latitude longitude* dari GPS, dapat dilakukan dengan menggunakan formula Haversine dan Forward Azimuth. Dengan menggunakan kedua formula tersebut dalam menentukan informasi jarak dan penentuan arah diharapkan dapat memberikan perhitungan yang dapat digunakan oleh pendaki untuk mengetahui arah jalur kembali pendakian serta informasi jarak terhadap suatu pos pendakian.

Saat ini terdapat berbagai *platform* yang digunakan pada perangkat *mobile* seperti Android, iOS, Windows Phone dan lain lain. Perbedaan *platform* akan menyulitkan dalam pengembangan aplikasi yang dapat berjalan pada semua *platform*. Penggunaan HTML 5 bisa menjadi solusi untuk membangun sebuah aplikasi *cross platform*. Dengan Lungo Js yang merupakan *mobile framework* menggunakan fitur terbaru dari HTML 5 dan CSS 3 bisa menjadi pilihan untuk membuat aplikasi *mobile cross platform*. Keunggulan Lungo Js adalah kecepatan eksekusi dengan UI (*User Interface*) kompleks, serta kustomisasi UI dan penggunaan yang mudah [GAI-13].

Dengan kebutuhan aplikasi untuk mengakses fitur *native* seperti *Geolocation, Storage* dan *Compass* maka digunakan Apache Cordova. Apache Cordova merupakan satu set API (*Application Program Interface*) javascript yang memungkinkan pengembang aplikasi *mobile* untuk mengakses fungsi perangkat seperti kamera atau *accelerometer* dengan javascript [COR-14].

Berdasarkan permasalahan yang sudah dipaparkan diatas, maka penulisan skripsi ini akan membangun aplikasi *mobile* dalam memvisualisasikan jalur pendakian gunung memanfaatkan GPS (*Global Positioning System*) dan kompas digital sehingga dapat menandai jalur pendakian dan dibangun menggunakan Lungos Js dan Apache Cordova agar dapat berjalan dibeda *platform*.

1.2 Rumusan Masalah

Berdasarkan permasalahan yang di angkat pada bagian latar belakang, adapun rumusan masalah dari skripsi ini adalah sebagai berikut:

1. Bagaimana menganalisis dan merancang aplikasi *mobile* yang dapat memvisualisasikan jalur pendakian gunung ?
2. Bagaimana implementasi Lungo Js dalam memvisualisasikan jalur pendakian gunung pada perangkat bergerak dengan memanfaatkan GPS (*Global Positioning System*) dan kompas digital ?
3. Bagaimana mengimplementasikan aplikasi *mobile* visualisasi jalur pendakian gunung dalam mengakses fitur *native* pada perangkat bergerak menggunakan Apache Cordova sehingga fitur *native* tersebut dapat diakses menggunakan HTML 5 dan Javascript ?

4. Bagaimana mengimplementasikan formula Haversine dan *Forward* Azimuth dalam aplikasi *mobile* visualisasi jalur pendakian gunung sehingga dapat melakukan perhitungan informasi jarak dan arah berdasarkan kordinat *latitude longitude* ?

1.3 Batasan Masalah

Agar permasalahan yang di rumuskan lebih terfokus dan tidak terjadi pelebaran topik, maka penelitian skripsi ini di batasi hal berikut :

1. Pengambilan titik koordinat dengan titik koordinat berikutnya untuk penanda jalur pendakian minimal 10 meter karena tingkat akurasi GPS pada *smartphone* 3-6 meter.
2. Perhitungan jarak dan arah dalam derajat berdasarkan kordinat *latitude* dan *longtitude* menggunakan formula Haversine dan *Forward* Azimuth.

1.4 Tujuan

Tujuan dari skripsi ini adalah membangun aplikasi pada perangkat bergerak yang dapat memvisualisasikan jalur pendakian gunung sehingga membantu pendaki dalam menggambarkan jalur kembali pendakian yang benar.

1.5 Manfaat

Manfaat yang dapat diperoleh dari skripsi ini antara lain:

1. Menjadi media bagi pendaki dalam memvisualisasikan jalur pendakian yang dapat menjadi acuan informasi terhadap jalur kembali yang benar.
2. Membantu pendaki yang tersesat untuk kembali kejalur atau *track* yang benar.
3. Memudahkan pengelola pos pendakian untuk menggambarkan jalur pendakian kepada para pendaki.

1.6 Sistematika Penulisan

Sistematika penulisan laporan skripsi ini adalah:

Bab I Pendahuluan

Menguraikan mengenai latar belakang, rumusan masalah, batasan masalah, tujuan, manfaat dan sistematika penulisan.

Bab II Dasar Teori

Menguraikan dasar teori dan referensi yang mendasari pengembangan aplikasi *mobile* visualisasi jalur pendakian gunung.

Bab III Metode Penelitian

Menguraikan tentang metode dan langkah kerja yang di lakukan dalam penulisan skripsi yang terdiri dari studi literatur, perancangan perangkat lunak, implementasi perangkat lunak, pengujian dan analisis serta pengambilan kesimpulan dan saran.

Bab IV Perancangan

Membahas analisis kebutuhan dan perancangan aplikasi *mobile* visualisasi jalur pendakian gunung sesuai dengan dasar teori dan literatur yang sudah ada.

Bab V Implementasi

Membahas implementasi dari aplikasi *mobile* visualisasi jalur pendakian gunung sesuai dengan perancangan yang telah dibuat.

Bab VI Pengujian dan Analisis

Memuat hasil pengujian dan analisis terhadap aplikasi yang telah di realisasikan.

Bab VII Penutup

Memuat kesimpulan yang di peroleh dari pembuatan dan pengujian aplikasi yang dikembangkan dalam skripsi ini serta saran-saran untuk pengembangan lebih lanjut.

BAB II
DASAR TEORI

Pada bab ini dijelaskan mengenai dasar teori yang digunakan untuk menunjang penulisan skripsi "Rancang Bangun Aplikasi *Mobile* Visualisasi Jalur Pendakian Gunung Menggunakan Lungo Js dan Apache Cordova". Beberapa dasar teori yang dimaksud diantaranya adalah Apache Cordova, Lungo Js, Quo Js, Google *Maps* API, Haversine dan *Forward* Azimuth.

2.1 Apache Cordova

Apache Cordova adalah satu set API perangkat yang memungkinkan pengembang aplikasi *mobile* untuk mengakses fungsi perangkat asli seperti kamera atau *accelerometer* dari JavaScript. Dikombinasikan dengan kerangka UI seperti jQuery Mobile atau Dojo Mobile atau Sencha Touch, ini memungkinkan aplikasi *smartphone* untuk dikembangkan hanya dengan HTML, CSS, dan JavaScript.

Bila menggunakan API Cordova, sebuah aplikasi dapat dibangun tanpa kode asli (Java, Objective-C, dll). Sebaliknya, teknologi web yang digunakan, dan berjalan secara lokal (umumnya tidak di server). Oleh karena itu, API JavaScript konsisten di beberapa platform perangkat dan dibangun pada standar web, aplikasi menjadi *portabel* untuk *platform* lain dengan minimal atau tanpa perubahan kode.

Cordova menyediakan satu set perpustakaan JavaScript seragam yang dapat dipanggil, dengan perangkat-spesifik dukungan kode asli bagi perpustakaan JavaScript. Cordova yang tersedia untuk platform berikut: iOS, Android, Blackberry, Windows Phone, Palm WebOS, Bada, dan Symbian.

Apache Cordova lulus pada Oktober 2012 sebagai proyek tingkat atas dalam Apache *Software Foundation* (ASF). Melalui ASF, pengembangan Cordova masa depan akan memastikan kepengurusan proyek terbuka. Ini akan selalu tetap bebas dan *open source* di bawah lisensi Apache. [COR-14]

2.1.1 API *Geolocation*

Geolocation merupakan objek yang menyediakan akses data berbasis lokasi pada perangkat sensor GPS. *Geolocation* menyediakan informasi tentang lokasi perangkat, seperti garis lintang dan bujur [PHO-14].

Dengan menggunakan API *Geolocation* memungkinkan dengan javascript dapat mengakses *hardware* geolocation dari sebuah perangkat bergerak. Contoh *source code* untuk menggunakan API *Geolocation* dalam mengakses *hardware* dan mendapatkan sebuah nilai *latitude longitude* ditunjukkan dalam Gambar 2.1.

```
<script type="text/javascript" charset="utf-8">
    // Wait for device API libraries to load
    document.addEventListener("deviceready",  onDeviceReady,
false);
    // device APIs are available
    function onDeviceReady() {
        navigator.geolocation.getCurrentPosition(onSuccess,
onError);
    }
    // onSuccess Geolocation
    function onSuccess(position) {
    var element = document.getElementById('geolocation');
    element.innerHTML = 'Latitude:'+
    position.coords.latitude+ '<br />'+
    'Longitude: '+ position.coords.longitude + '<br />';
    }
    // onError Callback receives a PositionError object
    function onError(error) {
        alert('code: '    + error.code    + '\n' +
              'message: ' + error.message + '\n');
    }
    </script>
```

Gambar 2.1 Contoh *Source Code* Penggunaan API *Geolocation*

Sumber : [PHO-14]

2.1.2 API *Compass*

Compass merupakan sebuah fitur *native* yang terdapat dalam sebuah *smartphone* yang dapat memberikan informasi penunjuk arah. Untuk menggunakan fitur *native* compass menggunakan API *Compass* dari Apache Cordova ditunjukkan dalam Gambar 2.2.

```
<script type="text/javascript" charset="utf-8">
    // Wait for device API libraries to load
    document.addEventListener("deviceready",  onDeviceReady,
false);
    // device APIs are available
    function onDeviceReady() {
     navigator.compass.getCurrentHeading(onSuccess,
onError);
    }
    // onSuccess: Get the current heading
    function onSuccess(heading) {
        alert('Heading: ' + heading.magneticHeading);
    }
    // onError: Failed to get the heading
    function onError(compassError) {
        alert('Compass Error: ' + compassError.code);
    }
    </script>
```

Gambar 2.2 Contoh *Source Code* API *Compass*

Sumber : [PHN-14]

2.1.3 API *Storage*

API *Storage* menyediakan akses untuk media penyimpanan secara lokal dalam perangkat bergerak. Contoh penggunaan *database* menggunakan API *Storage* ditunjukkan dalam Gambar 2.3.

```
<script type="text/javascript" charset="utf-8">
    // Wait for device API libraries to load
    document.addEventListener("deviceready",  onDeviceReady,
```

```
false);
    // device APIs are available
    function onDeviceReady() {
        var db = window.openDatabase("Database", "1.0",
"Cordova Demo", 200000);
        db.transaction(populateDB, errorCB, successCB);
    }
    // Populate the database
    function populateDB(tx) {
        tx.executeSql('DROP TABLE IF EXISTS DEMO');
        tx.executeSql('CREATE TABLE IF NOT EXISTS DEMO (id
unique, data)');
        tx.executeSql('INSERT INTO DEMO (id, data) VALUES
(1, "First row")');
        tx.executeSql('INSERT INTO DEMO (id, data) VALUES
(2, "Second row")');}
    // Transaction error callback
    function errorCB(tx, err) {
        alert("Error processing SQL: "+err);
    }
    // Transaction success callback
    function successCB() {
        alert("success!")}
    </script>
```

Gambar 2.3 Contoh *Source Code* API *Storage*

Sumber : [PHE-14]

2.2 Lungo Js

Lungo Js merupakan *framework* HTML5 untuk *developer* yang ingin merancang, membangun dan berbagi aplikasi beda *platform*. Berikut beberapa kelebihan dari Lungo Js :

- *HTML5 Optimized Apps*: Mendukung standar web terbuka, seperti HTML5, CSS3 dan JavaScript. Ini membawa lingkungan yang konsisten di ponsel, TV dan perangkat komputer.

- *Open Source Project*: Setiap kode baris baru di Lungo akan disambut baik, pengembang beakan membantu untuk meningkatkan hari demi hari proyek ini.
- *Powerfull JavaScript API*: Lungo menawarkan API yang kuat sehingga dapat memiliki kontrol penuh terhadap segala sesuatu yang terjadi di aplikasi yang dikembangakan.
- *Cross-Device*: Lungo akan sesuai dengan semua dari mereka menciptakan UX yang unik dan menakjubkan. [SOJ-14].

Berikut struktur yang digunakan untuk mengembangkan sebuah aplikasi menggunakan Lungo Js.

```
<link rel="stylesheet"
href="components/lungo.brownie/lungo.css">
<link rel="stylesheet"
href="components/lungo.icon/lungo.icon.css">
<link rel="stylesheet"
href="components/lungo.brownie/lungo.theme.css">
<script src="components/quojs/quo.js"></script>
<script src="components/lungo/lungo.js"></script>
```

Gambar 2.4 Struktur Menggunakan Lungo Js

Sumber : [SOJ-14]

`<section>` merupakan wadah utama komponen UI di aplikasi dan `<article>` harus ditempatkan di dalam bagian `<section>`. Setiap bagian dan artikel harus berisi ID unik.

```
<section id= "main">
     <article id= "main-article">
         Konten Anda
     </ Article>
 </ Section>
```

Gambar 2.5 Struktur Isi di Lungo Js

Sumber : [SOJ-14]

Fungsi JavaScript yang menginisialisasi Lungo :

```
.Lungo init ({});
```

Gambar 2.6 Inisialisasi Lungo Js

Sumber : [SOJ-14]

2.3 Quo Js

Quo Js adalah mikro, modular, *Object-Oriented* dari JavaScript *library* yang menyederhanakan HTML *document traversing*, penanganan *event*, dan interaksi Ajax untuk pengembangan web *mobile* yang cepat. Hal ini memungkinkan untuk menulis kode yang fleksibel dan kode *cross-browser* yang elegan. Dirancang untuk mengubah cara menulis JavaScript dengan ukuran kecil: 5-6K gzip yang dapat menangani kerja-kerja dasar dengan API yang bagus sehingga dapat berkonsentrasi menyelesaikan aplikasi yang dikembangkan. Dirilis di bawah lisensi *Open Source* MIT, yang memberikan kemungkinan untuk menggunakannya dan memodifikasinya dalam setiap keadaan.

JavaScript saat ini dibangun berdasarkan kebutuhan perangkat desktop sehingga kinerja pada ponsel tidak optimal. Maka dari itu Quo Js dikembangkan utnuk membantu pengembang untuk menciptakan aplikasi menggunakan JavaScript yang baik dan bagus.

Quo Js mendukung beberapa *gesture* pada *smartphone* layar sentuh yaitu *Tap, Single Tap, Double-Tap, Hold, 2xFingers Tap, 2xFingers Double-Tap, Swipe Up, Swipe Right, Swipe Down, Swipe Left, Swipe, Drag, Rotate Left, Rotate Right, Rotate, Pinch Out, Pinch, Fingers* [SOY-14].

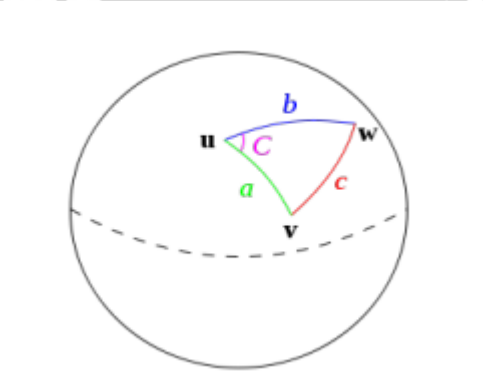
2.4 Google *Map* API

Google *Map* API merupakan aplikasi *interface* yang dapat diakses menggunakan javascript yang bertujuan untuk menggunakan Google *map* sehingga dapat ditampilkan pada aplikasi yang dibangun. Menggunakan Google *Map* API haru mendapatkan *Api Key* terlebih dahulu. Dengan Google *Map* API ini kita juga bisa mendapatkan banyak fitur yang dapat membantu kita dalam mengembangkan sebuah aplikasi. Beberapa fitur dari Google *Map* API seperti

marker sebuah tempat, garis pengubung antar tempat satu dengan yang lain, *polygon* dan lain-lain.

2.5 Haversine

Haversine adalah persamaan penting dalam navigasi untuk menghitung jarak lingkaran besar terpendek atas permukaan bumi untuk bujur tertentu dan lintang [ALG-13]. Rumus Haversine mengasumsikan bulat bumi dan tidak mencakup dampak ellipsoidal [ESS-11]. Untuk menghitung jarak antara dua terrestrial koordinat seperti ditunjukkan pada Gambar 2.7.



Gambar 2.7 Ilustrasi Segitiga dalam Haversine

Sumber : [ESS-11]

Untuk menghitung jarak antara dua *GeoPoints* rumus Haversine sebagai berikut :

$$a = \sin2 (\Delta\text{latitude} / 2) + \cos (\varphi1) * \cos (\varphi2) * \sin2 (\Delta \lambda / 2) \quad (2.1)$$

$$c = 2 * \text{atan2} (\sqrt{a} , \sqrt{(1 - a)}) \quad (2.2)$$

$$d = R * c \quad (2.3)$$

Dimana :

R = radius bumi dan itu sama dengan 6.371 km.

$\Delta\varphi$ = perbedaan bujur antara lokasi pengguna dan tujuan (φ 2 - φ 1) dalam radian.

$\Delta\lambda$ = perbedaan lintang antara lokasi pengguna dan tujuan (λ 2 - λ 1) dalam radian.

2.6 *Forward* Azimuth

Forward Azimuth berfungsi untuk menghitung sudut antara dua *GeoPoints*. Secara umum, nilai posisi bervariasi karena mengikuti lingkaran jalan besar (orthodrome); posisi akhir akan berbeda dari posisi awal dengan derajat yang bervariasi sesuai dengan jarak dan lintang. Formula ini adalah untuk bantalan awal yang jika diikuti dalam garis lurus sepanjang busur lingkaran besar akan memberikan nilai dari titik awal ke titik akhir [VEN-10]. Berikut rumus dari *forward* azimuth :

$$\theta = \operatorname{atan2}(\sin \Delta\lambda \cdot \cos \varphi 2 , \cos \varphi 1 \cdot \sin \varphi 2 - \sin \varphi 1 \cdot \cos \varphi 2 \cdot \cos \Delta\lambda) \quad (2.4)$$

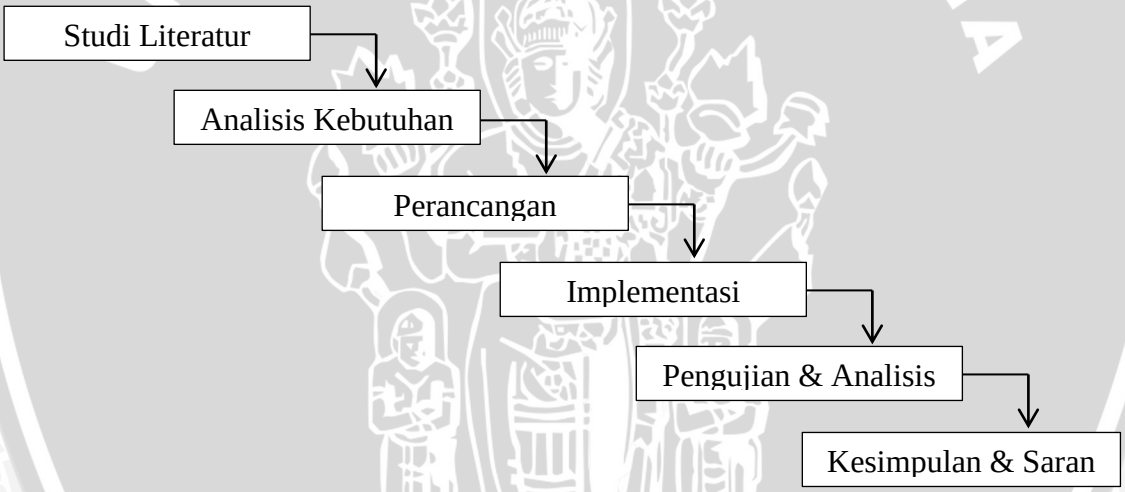
Dimana :

$\Delta\varphi$ = perbedaan bujur antara lokasi pengguna dan tujuan (φ 2 - φ 1) dalam radian.

$\Delta\lambda$ = perbedaan lintang antara lokasi pengguna dan tujuan (λ 2 - λ 1) dalam radian.

BAB III
METODE PENELITIAN

Pada bab ini menjelaskan tentang langkah-langkah dalam perancangan, implementasi dan pengujian dari aplikasi *mobile* visualisasi jalur pendakian gunung yang akan dibuat. Langkah pertama adalah mengumpulkan teori-teori pendukung dan mengemasnya ke dalam studi literatur. Kemudian dilanjutkan dengan proses analisis kebutuhan serta perancangan aplikasi. Langkah berikutnya adalah implementasi sesuai dengan perancangan yang telah dibuat. Kemudian dilakukan pengujian dan analisis terhadap aplikasi. Kesimpulan dan saran disertakan sebagai catatan atas aplikasi dan kemungkinan arah pengembangan aplikasi selanjutnya.



Gambar 3.1 Alur Penelitian dengan model *Waterfall*

3.1 Studi Literatur

Studi literatur menjelaskan dasar teori yang digunakan sebagai penunjang dan acuan penulisan skripsi dan pengembangan aplikasi. Teori dan pustaka pendukung tersebut meliputi:

1. Apache Cordova
   a. API *Geolocation*
   b. API *Compass*
   c. API *Storage*

2. Lungo Js
3. Quo Js
4. Google *Maps* API
5. Haversine
6. *Forward* Azimuth

3.2 Analisis Kebutuhan

Analisis kebutuhan bertujuan untuk mengetahui semua kebutuhan yang diperlukan dari pengembangan perangkat lunak yang akan dibangun. Metode analisis yang digunakan adalah *object-oriented analysis* dengan bahasa permodelan 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 (*requirements*) sistem aplikasi visualisasi jalur pendakian gunung yang kemudian dimodelkan dalam diagram *use case*.

3.3 Perancangan

Pada tahap perancangan perangkat lunak akan dilakukan setelah semua kebutuhan sudah didapatkan pada tahap analisis kebutuhan. Pada tahap pertama dilakukan perancangan arsitektural, kemudian dilakukan perancangan sistem yang dimodelkan dalam EPC (*Event-driven Process Chain*). Tahap ketiga yaitu dilakukan perancangan *database* untuk menyimpan peta dan tahap perancangan selanjutnya adalah antarmuka pengguna atau *user interface* aplikasi.

3.4 Implementasi

Implementasi merupakan tahapan pengembangan perangkat lunak yang mengacu pada perancangan yang telah dibuat sebelumnya. Pada tahap implementasi diawali dengan penjabaran spesifikasi lingkungan perancangan perangkat lunak. Implementasi pengembangan perangkat lunak menggunakan bahasa standar web yaitu HTML 5, CSS 3 dan Javascript. Proses *porting* aplikasi web menjadi aplikasi *native* pada android menggunakan Apache Cordova. Pada tahap akhir dilakukan implementasi simulasi pada *smartphone* android secara langsung.

3.5 Pengujian dan Analisis

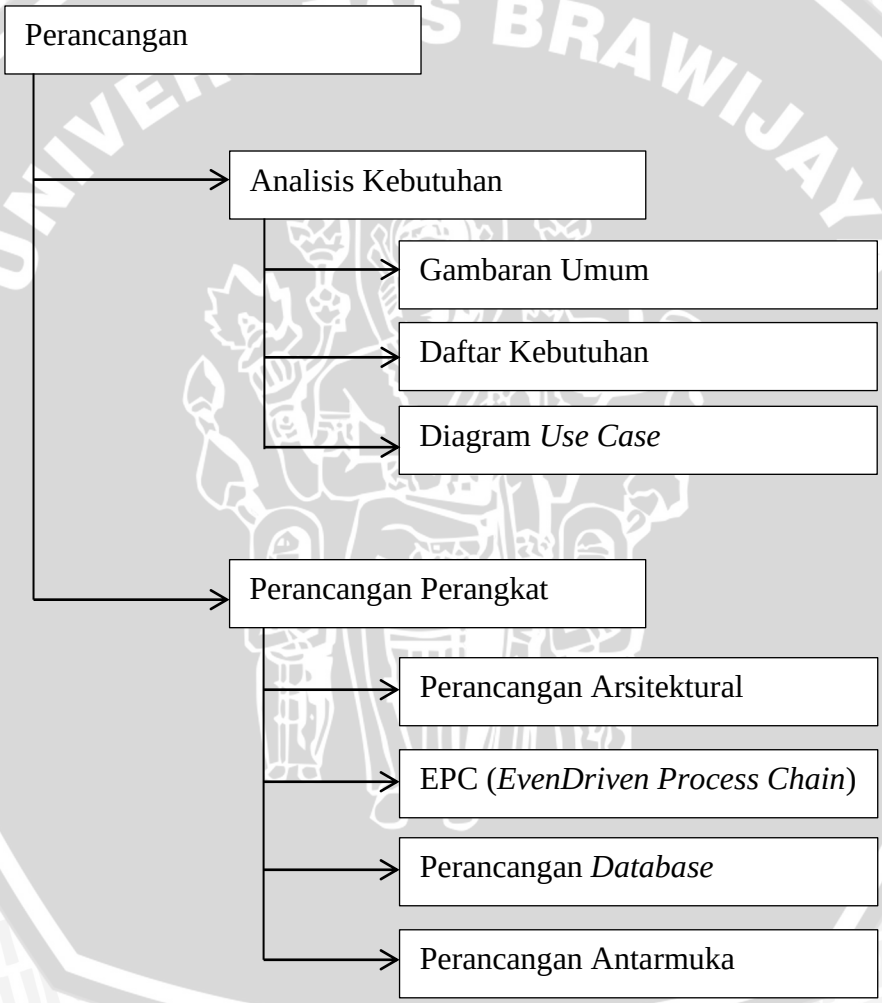
Pengujian perangkat lunak merupakan tahap untuk mengetahui apakah kinerja dan peforma aplikasi visualiasi jalur pendakian gunung telah sesuai dengan spesifikasi kebutuhan yang melandasinya. Dilakukan tiga tahapan dalam pengujian pada aplikasi ini yaitu pengujian Apache Cordova, pengujian perbandingan perhitungan manual dan aplikasi kemudian pengujian validasi. Pengujian Apache Cordova dilakukan untuk mengetahui apakah Apache Cordova dapat mengaskses *hardware* yaitu *Geolocation*, *Compass* dan *Storage* pada perangkat bergerak yang menjadi kebutuhkan aplikasi visualisasi jalur pendakian dengan menggunakan javascript. Pengujian perbandingan perhitungan manual dan aplikasi dilakukan untuk mengetahui apakah implementasi perhitungan formula Haversine dan *Forward* Azimuth dalam aplikasi sudah memiliki ketelitian yang sama dengan perhitungan yang dilakukan secara manual. Pengujian validasi dilakukan untuk menguji kinerja sistem apakah sesuai dengan kebutuhan fungsionalitas yang sudah dibuat sebelumnya serta merupakan pengujian terhadap implementasi Lungo Js yang menjadi antarmuka pada aplikasi *native* dengan menggunakan HTML 5 dan CSS 3. Setelah tahap pengujian, dilakukan analisis untuk mengetahui hasil dari pengujian perangkat lunak sehingga mendapatkan kesimpulan dari rancang bangun aplikasi yang telah dibuat.

3.6 Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan, implementasi dan pengujian perangkat lunak telah selesai dilakukan. Kesimpulan diambil berdasarkan hasil pengujian dan analisis terhadap sistem yang dibangun. Setelah itu penulisan saran yang dimaksudkan untuk memberikan pertimbangan atas pengembangan perangkat lunak lebih lanjut.

BAB IV
ANALISIS DAN PERANCANGAN

Pada bab ini dijelaskan mengenai perancangan aplikasi visualisasi jalur pendakian gunung pada perangkat bergerak. Perancangan yang dilakukan meliputi dua tahap. Tahap pertama dilakukan analisis kebutuhan dilanjutkan tahap kedua yaitu perancangan perangkat lunak secara lebih spesifik. Perancangan perangkat lunak sistem ini ditunjukan dengan diagram blok dalam Gambar 4.1.



Gambar 4.1 Diagram Blok Perancangan

4.1 Analisis Kebutuhan

Pada tahap analisis kebutuhan ini diawali dengan gambaran umum perangkat lunak yang akan dibangun. Kemudian dilanjutkan dengan penjabaran kebutuhan perangkat lunak yang dimodelkan dalam diagram *use case*. Analisis kebutuhan ini ditunjukan untuk menggambarkan kebutuhan-kebutuhan yang harus disediakan oleh sistem agar dapat memenuhi kebutuhan pengguna.

4.1.1 Gambaran Umum

Aplikasi visualisasi jalur pendakian gunung adalah aplikasi pada perangkat bergerak yang dapat memvisualisasikan jalur pendakian dengan memanfaatkan GPS. Aplikasi ini bisa menjadi media bagi para pendaki dalam menggambarkan jalur pendakian dengan mudah dan praktis, sehingga berguna untuk memberikan gambaran jalur kembali yang benar dan dapat menghindari kejadian tersesatnya para pendaki. Pada aplikasi visualisasi jalur pendakian ini, disediakan tanda berupa gambar yang dapat digunakan untuk memberikan tanda pada setiap titik jalur pendakian yang diambil berdasarkan kordinat GPS, hal ini bertujuan untuk memudahkan pendaki dalam mengenal kembali jalur yang telah dilewati sebelumnya. Selain penanda, juga terdapat penunjuk arah kembali yang memanfaatkan kompas digital dan nilai *latitude longitude* yang didapatkan dari GPS. Fitur penunjuk arah kembali ini merupakan pengganti dari proses navigasi darat yang berguna untuk menentukan jalur pendakian secara manual. Fitur lainnya adalah informasi jarak dari posisi terbaru terhadap titik jalur pendakian yang sudah diambil sebelumnya sehingga memberikan informasi kepada pendaki seberapa jauh dari posisi sebelumnya. Jalur pendakian yang sudah digambarkan juga bisa divisualisasikan dalam bentuk *map*. Hal ini terpenuhi jika terdapat jaringan internet ditempat pendakian. Maka dengan itu, hasil dari visualisasi jalur pendakian yang sudah dibuat bisa digunakan oleh pendaki untuk memberikan gambaran terhadap jalur kembali yang benar.

4.1.2 Daftar Kebutuhan

Dalam mengidentifikasi kebutuhan yang harus dipenuhi dalam pengembangan perangkat lunak, maka identifikasi kebutuhan menjadi dua hal yaitu kebutuhan fungsional dan kebutuhan non-fungsional. Kebutuhan fungsional merupakan kebutuhan utama yang dibutuhkan dalam menjalankan sistem. Sedangkan kebutuhan non-fungsional merupakan pendukung sistem yang akan dijalankan. Kebutuhan fungsional ditunjukkan pada Table 4.1 dan kebutuhan non-fungsional ditunjukkan pada Table 4.2.

Tabel 4.1 Daftar Kebutuhan Fungsional

Nomor	Kebutuhan	*Use Case*
FR _001	Pengguna dapat melihat halaman peta.	Melihat halaman Peta.
FR _002	Pengguna dapat melihat titik jalur pendakian.	Melihat jalur pendakian.
FR _003	Pengguna dapat menambahkan tanda pada titik jalur pendakian dengan objek gambar atau teks.	Menambahkan tanda.
FR _004	Pengguna dapat menghapus tanda gambar atau teks.	Menghapus tanda.
FR _005	Pengguna dapat melihat arah kembali dengan kompas sesuai kordinat.	Melihat arah kembali.
FR _006	Pengguna dapat melihat informasi jarak posisi terbaru dengan titik sebelumnya.	Melihat informasi jarak.
FR _007	Pengguna dapat melihat jalur pendakian dalam *map*.	Melihat jalur pendakian dalam *map*.
FR _008	Pengguna dapat menyimpan data jalur pendakian dalam *database*.	Menyimpan jalur pendakian.
FR _009	Pengguna dapat membuat peta baru.	Membuat peta baru.

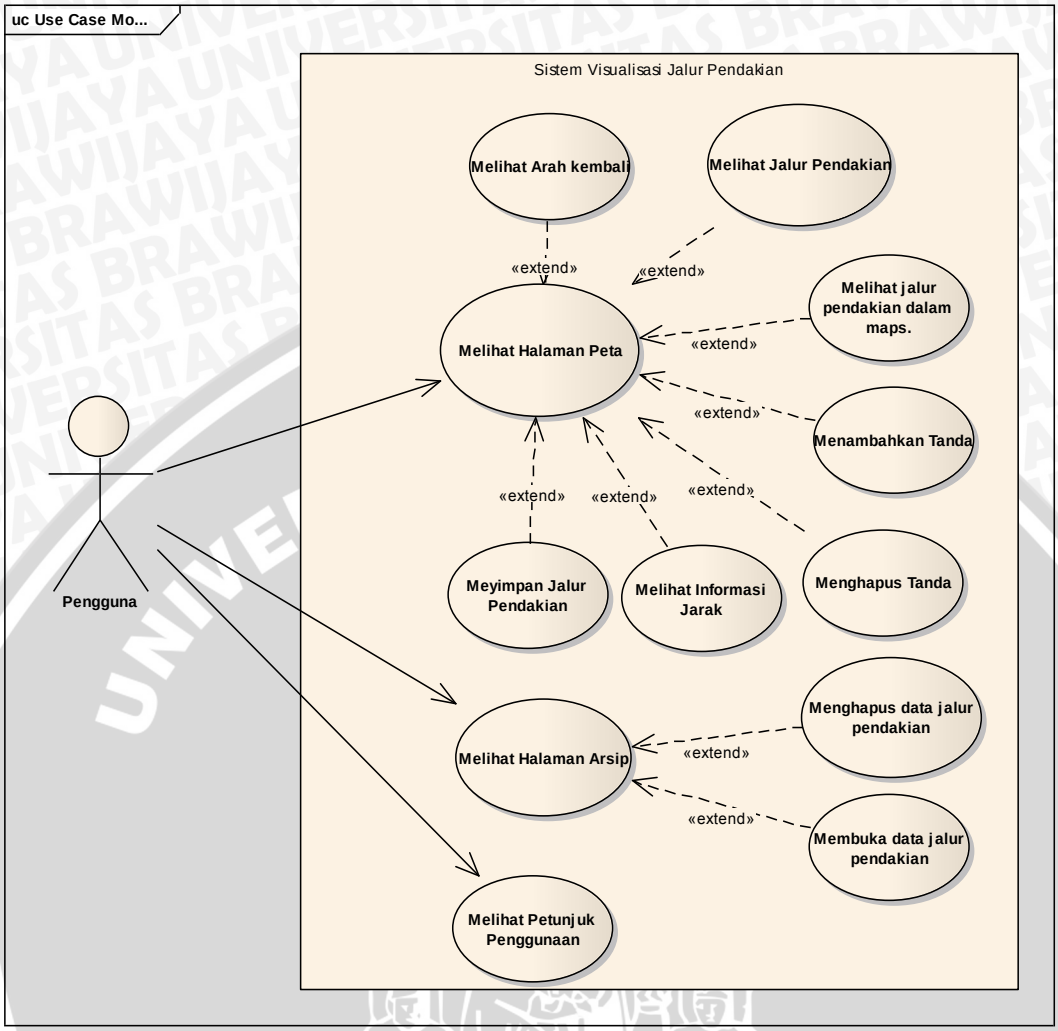
FR _010	Pengguna dapat melihat halaman arsip.	Melihat halaman arsip.
FR_011	Pengguna dapat membuka data jalur pendakian dari *database*.	Membuka jalur pendakian.
FR_012	Pengguna dapat menghapus data jalur pendakian dari *database*.	Menghapus jalur pendakian.
FR_013	Pengguna dapat melihat halaman petunjuk penggunaan.	Melihat halaman petunjuk penggunaan.

Tabel 4.2 Spesifikasi Kebutuhan Non-Fungsional

Parameter	Deskripsi kebutuhan
Reliability	Aplikasi dapat menyimpan data jalur pendakian ketika perangkat yang digunakan mati secara tiba-tiba sehingga data jalur pendakian bisa digunakan kembali saat perangkat dihidupkan.

4.1.3 Diagram *Use Case*

Diagram *use case* merupakan diagram yang memodelkan interaksi antar pengguna dengan sistem. *Use case* sendiri merupakan sebuah pekerjaan yang dilakukan oleh pengguna terhadap sistem. Satu buah *use case* akan disertai dengan skenario *use case* untuk menjelaskan rangkaian aktivitas yang terjadi di *use case* tersebut. Diagram *use case* dari aplikasi visualisasi jalur pendakian ditunjukkan dalam Gambar 4.2.



Gambar 4.2 Diagram *Use Case* Sistem Visualisasi Jalur Pendakian

Gambar 4.2 diatas merupakan diagram *use case* yang menggambarkan interaksi antara pengguna dengan sistem pada aplikasi visualisasi jalur pendakian gunung. Pada diagram tersebut terlihat bahwa sistem memiliki beberapa menu yang dapat diakses seperti menu peta, arsip dan petunjuk penggunaan. Dari setiap menu memiliki fungsi tertentu seperti menu peta yang dapat digunakan oleh pengguna untuk memvisualisasikan jalur pendakian dari sebuah gunung serta dilengkapi dengan fitur penunjuk arah kembali, informasi jarak, penambahan sebuah tanda, dan melihat dalam bentuk *map*. Sedangkan menu arsip merupakan menu yang menampilkan daftar peta jalur pendakian yang tersimpan dalam *database local* perangkat bergerak pengguna. Jalur pendakian yang tersimpan bisa digunakan kembali dan juga dihapus secara permanen dari *database*.

Tabel 4.3 Skenario Use Case Melihat Jalur Pendakian

Nama Use Case	Melihat Jalur Pendakian.
Aktor	Pengguna.
Tujuan	Membuat jalur pendakian.
Deskripsi	*Use case* ini memodelkan kegiatan dalam menggambarkan titik jalur pendakian memanfaatkan GPS.
Kondisi Awal	Titik dari jalur pendakian belum digambarkan.
Kondisi Akhir	Titik dari jalur pendakian telah digambar.
Main Flow	1. Pengguna menekan tombol gambar titik jalur pendakian. 2. Sistem mengambil nilai *latitude longitude* menggunakan GPS. 3. Sistem menggambarkan titik jalur pendakian dan menyimpan nilai *latitude longitude* dari titik jalur pendakian tersebut.
Alternatif Flow	1.1 Eksepsi jika kordinat GPS tidak didapatkan. Menampilkan *alert* tentang kesalahan bahwa pengambilan kordinat GPS gagal. Kegagalan ini bisa terjadi karena GPS belum diaktifkan atau satelit GPS terlalu lama merespon permintaan pengguna.

Pada Tabel 4.3 menjelaskan dalam proses menggambarkan titik jalur pendakian sehingga menjadi jalur pendakian dari sebuah gunung. Hal ini merupakan langkah pertama dalam memvisualisasikan jalur pendakian, pengguna harus berada dalam menu peta terlebih dahulu kemudian menekan tombol gambar titik jalur pendakian. Setelah ditekan sistem akan mengambil nilai *latitude longitude* dengan GPS dan jika berhasil maka titik jalur pendakian digambarkan pada perangkat bergerak pengguna. Jika nilai *latitude longitude* tidak didapatkan maka akan muncul sebuah *alert* yang memberikan informasi bahwa *request*

latitude longitude GPS tidak bisa didapatkan. Hasil akhir dalam proses ini adalah penggambaran sebuah titik jalur pendakian pada *canvas* peta.

Tabel 4.4 Skenario Use Case Menyimpan Jalur Pendakian

Nama Use Case	Menyimpan jalur pendakian
Aktor	Pengguna.
Tujuan	Menyimpan data jalur pendakian ke *database*.
Deskripsi	Menyimpan data jalur pendakian yang telah dibuat.
Kondisi Awal	Data jalur pendakian belum tersimpan dalam *database*.
Kondisi Akhir	Data jalur pendakian telah tersimpan dalam *database*.
Main Flow	1. Pengguna membuat sebuah peta jalur pendakian. 2. Menekan *icon* tanda pilihan. 3. Menekan tombol simpan. 4. Mengisi nama jalur pendakian. 5. Menekan tombol simpan.
Alternatif Flow	-

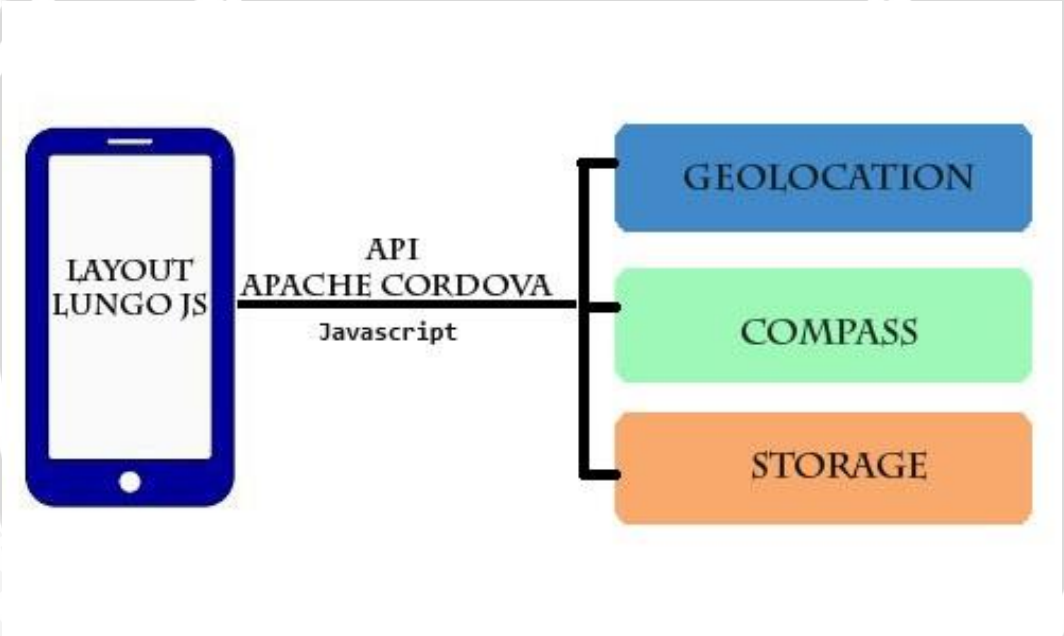
Pada Tabel 4.4 menjelaskan sebuah proses dalam menyimpan data jalur pendakian yang telah dibuat oleh pengguna kedalam *database local* pada perangkat yang digunakan. Data jalur pendakian yang disimpan bisa digunakan kembali dengan cara membukanya pada menu arsip. Langkah yang sama dilakukan untuk menjelaskan tabel skenario yang dapat dilihat pada lampiran.

4.2 Perancangan Perangkat Lunak

Perancangan perangkat lunak dilakukan dalam empat tahap, yaitu perancangan arsitektural, permodelan EPC (*Event-Driven Process Chain*) untuk menggambarkan struktur fungsi-fungsi yang menyusun aplikasi, perancangan *database* dan perancangan antarmuka pengguna.

4.2.1 Perancangan Arsitektural

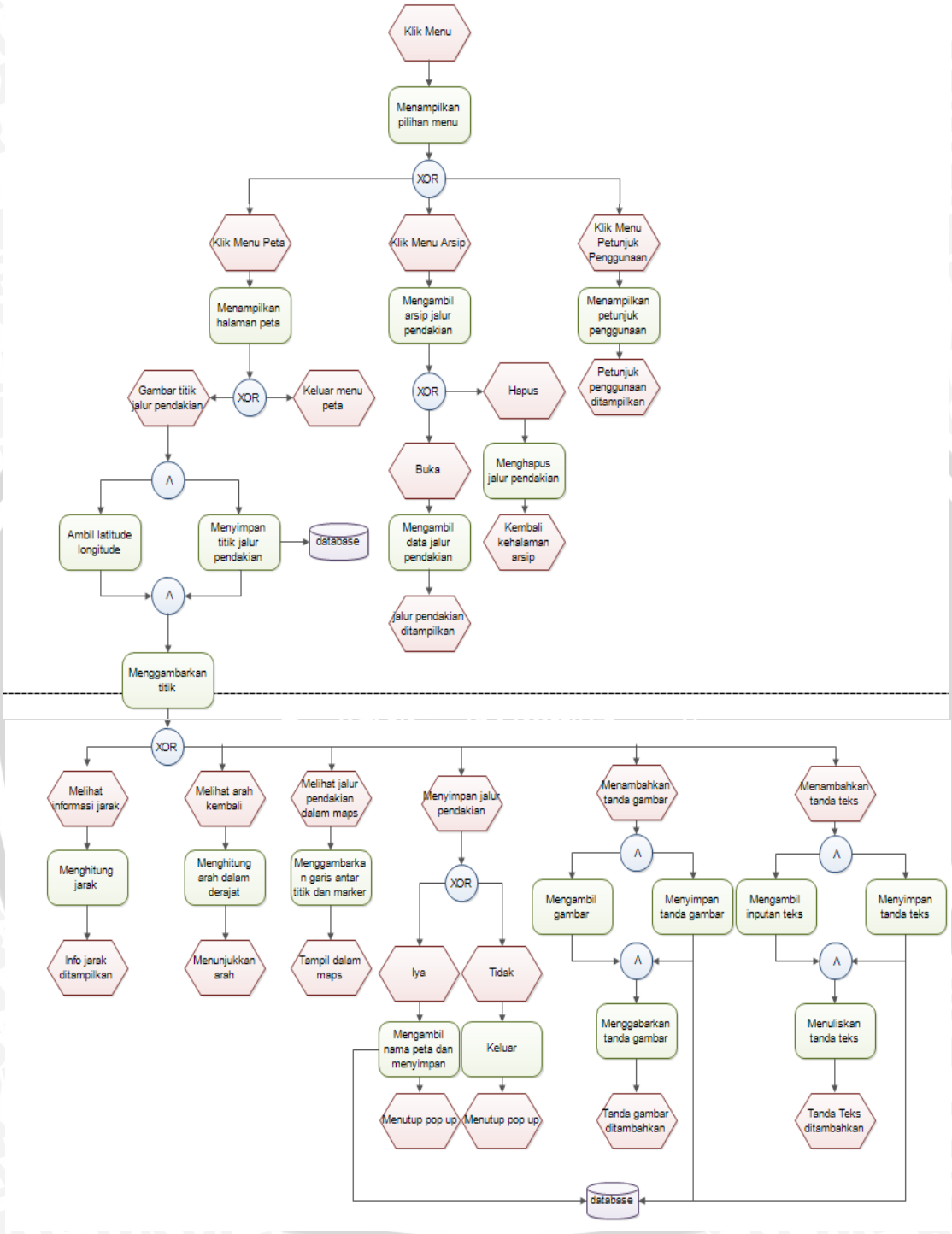
Aplikasi visualisasi jalur pendakian gunung menggunakan bahasa standar web yang nantinya di *porting* menjadi aplikasi *native* menggunakan Apache Cordova. Penggunaan Apache Cordova juga untuk mengakses fitur *native* seperti *Geolocation*, *Compass* dan *Database* pada *smartphone* menggunakan javascript. Pada bagian antarmuka menggunakan Lungo Js yang merupakan *mobile framework* HTML 5 dan CSS 3. *Framework* Lungo Js juga terdapat *micro javascript library* yang bernama Quo Js. Quo Js berguna untuk *event handling* pada perangkat *mobile touchscreen*. Desain arsitektural sistem ditunjukkan dalam Gambar 4.3.



Gambar 4.3 Perancangan Arsitektur Sistem

4.2.2 EPC (*Event-Driven Process Chain*)

Perancangan diagram EPC merupakan permodelan fungsi-fungsi penyusun sistem yang menggunakan metode programing *event-driven*. Rancangan diagram EPC ditunjukkan dalam Gambar 4.4.



Gambar 4.4 Rancangan Diagram EPC Sistem

Pada Gambar 4.4 diatas merupakan diagram EPC dari sistem visualisasi jalur pendakian gunung. Penggunaan objek segilima menggambarkan sebuah *event*, sedangkan gambar segiempat merupakan sebuah fungsi yang di *trigger* ketika *event* diberikan oleh pengguna. Penggunaan konektor yaitu XOR dan OR mempunyai arti seperti logika matematika secara umum. Alur dari *event* yang digambarkan dalam diagram EPC diatas adalah pengguna memberikan sebuah *event* untuk memilih menu, dari *event* tersebut sistem men*trigger* fungsi untuk menampilkan semua menu dari aplikasi, dan penghubung XOR berarti pengguna hanya dapat menjalan satu buah *event* dalam satu waktu. Seperti pengguna memilih menu peta maka sistem mengartikan bahwa terjadi sebuah *event* klik menu peta kemudian sistem akan men*trigger* fungsi untuk menampilkan menu peta. Dalam menu peta pengguna kemudian memberikan *event* untuk menggambarkan sebuah titik jalur pendakian maka sistem men*trigger* dua fungsi secara bersamaan yang ditandai dengan konektor AND. Fungsi yang dijalankan adalah mengambil nilai *latitude longitude* serta menyimpan titik tersebut dalam database. Setelah fungsi tersebut dijalankan maka terdapat *join* yang ditandai dengan dipanggilnya fungsi untuk menggambarkan titik jalur pendakian.

4.2.3 Perancangan *Database*

Perancangan *database* diperlukan dalam aplikasi ini agar data bisa digunakan kembali (*reusable*). Data yang akan disimpan diubah menjadi tipe JSON. Pada saat diambil kembali dari *database*, data dikembalikan kedalam bentuk *array*. Perubahan bentuk data dilakukan untuk memudahkan dalam penyimpanan *database*. Perancangan *database* ditunjukkan pada tabel 4.5.

Tabel 4.5 Percangan *Database* Sistem

No	Nama Kolom	Tipe Data	Keterangan	Contoh data
1	Nama (*Unique*)	*String, Primery key*	Nama dari jalur pendakian yang disimpan.	Gunung Semeru
2	keteranganTitik	*String*	Keterangan dari setiap titik yang digambarkan	Pos 2, jarak 120 m

3	sumbux	*String*	Posisi titik jalur pada sumbu x layar *smartphone*.	60
4	sumbuy	*String*	Posisi titik jalur pada sumbu y layar *smartphone*.	100
5	gambarTitik	*String*	*Path* gambar yang digunakan sebagai titik jalur pendakian.	Image/titik.png
6	sumbuxGambar	*String*	Posisi gambar pada sumbu x layar *smartphone*.	60
7	sumbuyGambar	*String*	Posisi gambar pada sumbu y layar *smartphone*.	Pos Pertama
8	*latitude*	*String*	Nilai *latitude* dari titik jalur pendakian yang digambarkan	-34.397
9	*longitude*	*String*	Nilai *longitude* dari titik jalur pendakian yang digambarkan.	150.644
10	gambarTanda	*String*	*Path* gambar yang digunakan sebagai penanda.	Image/pohon.png
11	sumbuxTanda	*String*	Posisi gambar pada sumbu x layar *smartphone*.	100
12	sumbuytanda	*String*	Posisi gambar pada sumbu y layar *smartphone*.	90

13	teksTanda	*String*	Teks yang digunakan sebagai penanda.	Ambil jalur kiri.
14	sumbuxTandaTeks	*String*	Posisi teks pada sumbu x layar *smartphone*.	60
15	sumbuyTandaTeks	*String*	Posisi teks pada sumbu y layar *smartphone*.	50

Pada Tabel 4.5 ditunjukkan rancangan database yang akan digunakan untuk menyimpan data jalur pendakian yang telah dibuat oleh pengguna. *Database* yang dipakai merupakan *database local* pada perangkat bergerak yang digunakan pengguna. DBMS (*Database Management System*) yang digunakan yaitu SQL lite. Data yang disimpan dalam *database* merupakan data dari jalur pendakian seperti nama dari jalur pendakian, kordinat titik pada sumbu x dan y pada perangkat, *path* gambar, nilai *latitude longitude* dan lain lain.

4.2.4 Perancangan Antarmuka Pengguna

Perancangan antarmuka pengguna dilakukan untuk memberikan gambaran bagaimana tampilan aplikasi yang akan dibuat. Antarmuka aplikasi ini akan menjadi jembatan antara pengguna untuk berinteraksi dengan sistem.

4.2.4.1 Antarmuka Menu

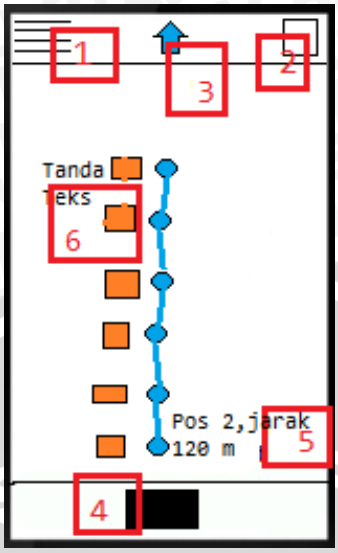
Antarmuka menu merupakan tampilan yang menggambarkan menu-menu yang dapat dipilih oleh pengguna. Pada bagian ini terdapat empat pilihan menu yaitu beranda, menu peta, menu arsip, menu petunjuk penggunaan. Perancangan antarmuka menu ditunjukkan dalam Gambar 4.5.

Gambar 4.5 Perancangan Antarmuka Menu

Pada Gambar 4.5 ditunjukkan perancangan antarmuka pengguna dalam memilih menu yang ada pada aplikasi visualisasi jalur pendakian gunung. Menu beranda merupakan menu yang akan tampil pertama kali saat aplikasi dijalankan. Dan menu peta merupakan menu yang digunakan pengguna untuk membuat dan melihat jalur pendakian gunung. Sedangkan menu arsip merupakan menu yang menampilkan data jalur pendakian yang tersimpan dalam perangkat pengguna. Dan menu petunjuk merupakan menu yang berisi mengenai informasi penggunaan aplikasi visualisasi jalur pendakian.

4.2.4.2 Antarmuka Jalur Pendakian

Antarmuka dari jalur pendakian merupakan antarmuka ketika jalur pendakian telah dibuat oleh pengguna. Perancangan antarmuka visualisasi jalur pendakian ditunjukkan dalam Gambar 4.6.



Gambar 4.6 Perancangan Antarmuka Jalur Pendakian

Dalam Gambar 4.6 ditunjukkan perancangan antarmuka ketika visualisasi jalur pendakian dibuat. Berikut keterangan dari perancangan antarmuka diatas :

1. Tombol untuk membuka pilihan menu.
2. Tombol untuk membuka pilihan tanda berupa gambar/teks, tombol simpan, tombol penunjuk arah, tombol untuk melihat versi *map*, tombol melihat informasi jarak.
3. *Icon* yang menunjukkan informasi arah kembali dari jalur pendakian.
4. Tombol untuk menggambarkan titik jalur pendakian.
5. Keterangan dari titik jalur pendakian berupa nomor pos dan informasi jarak terhadap titik sebelumnya.
6. Tanda gambar dan tanda teks dari titik jalur pendakian.

4.2.4.3 Antarmuka Jalur Pendakian Dalam *Map*

Antarmuka jalur pendakian dalam *map* merupakan visualisasi jalur pendakian yang memanfaatkan API google *map*. Dalam antarmuka jalur pendakian dalam *map* akan seperti jalur pendakian *offline*, yaitu adanya informasi urutan pos dan informasi jarak. Perancangan antarmuka jalur pendakian dalam *map* ditunjukkan dalam Gambar 4.7.

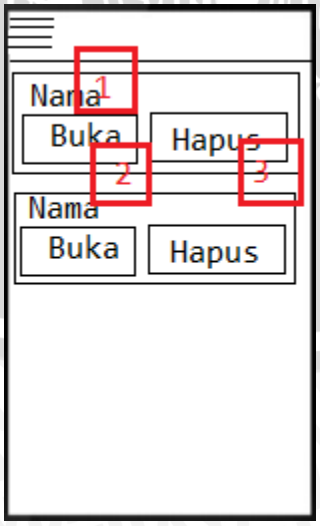
Gambar 4.7 Perancangan Antarmuka Jalur Pendakian Dalam *Map*

Dalam Gambar 4.7 ditunjukkan perancangan antarmuka jalur pendakian dalam *map*. Berikut keterangan dari perancangan antarmuka diatas :

1. *Marker* atau Titik dari sebuah jalur pendakian.
2. Keterangan dari masing-masing titik jalur pendakian.

4.2.4.4 Antarmuka Menu Arsip

Menu arsip merupakan menu yang berfungsi untuk menampilkan data jalur pendakian yang telah disimpan pengguna sebelumnya. Perancangan antarmuka menu arsip ditunjukkan dalam Gambar 4.8.



Gambar 4.8 Perancangan Antarmuka Menu Arsip

Dalam Gambar 4.8 ditunjukkan perancangan antarmuka menu arsip. Berikut keterangan dari perancangan antarmuka diatas :

1. Nama dari jalur pendakian.
2. Tombol buka untuk membuka data jalur pendakian.
3. Tombol hapus untuk menghapus data jalur pendakian dari *database*.

Untuk perancangan antarmuka bagian menu petunjuk dan lain lain dapat dilihat pada lampiran.

BAB V
IMPLEMENTASI

Pada 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 fungsi dan *layout*, implementasi *database*, implementasi algoritma dan implementasi antarmuka perangkat lunak.

5. 1 Spesifikasi Sistem

Hasil dari analisis kebutuhan dan perancangan perangkat lunak yang telah dijelaskan pada BAB IV menjadi acuan untuk melakukan implementasi menjadi sebuah aplikasi perangkat bergerak yang dapat berfungsi sesuai dengan kebutuhan. Spesifikasi sistem di implementasikan pada spesifikasi perangkat keras dan perangkat lunak.

5.1.1 Spesifikasi Perangkat Keras

Spesifikasi perangkat keras yang digunakan untuk implementasi aplikasi visualiasi jalur pendakian ditunjukkan pada Tabel 5.1.

Tabel 5.1 Spesifikasi Perangkat Keras *Smartphone*

Nama Komponen	Deskripsi
Touch Screen	Sebuah media *input* pada perangkat bergerak yang menggunakan sentuhan.
Geolocation	Sebuah *hardware include* pada perangkat bergerak sebagai *receiver* data GPS.
Compass	Sebuah *hardware include* pada perangkat bergerak yang dapat menunjukkan arah utara, selatan, timur dan barat memanfaatkan magnetik kutub utara dan selatan

Storage	Sebuah *hardware include* pada perangkat bergerak yang berfungsi sebagai media penyimpanan.

5.1.2 Spesifikasi Perangkat Lunak

Spesifikasi perangkat lunak yang digunakan dalam proses pembuatan aplikasi visualisasi jalur pendakian ditunjukkan pada Tabel 5.2.

Tabel 5.2 Spesifikasi Perangkat Lunak Komputer

Nama Komponen	Spesifikasi
Sistem Operasi	Windows 8 pro 64-bit (6.2, build 9200)
Application Progammable Interface	Apache Cordova 3.3.0, Lungo Js v.2.2, Quo Js 2.3.6
IDE (*Integrated Development Environment*)	NetBeans IDE 7.3 dan Android *Developer Tools Bundle*

Spesifikasi perangkat lunak yang digunakan untuk implementasi aplikasi visualisasi jalur pendakian ditunjukkan pada Tabel 5.3.

Tabel 5.3 Spesifikasi Perangkat Lunak *Smartphone*

Nama Komponen	Spesifikasi
Sistem Operasi	Android
DBMS *(Database Management System)*	SQL Lite

5.2 Batasan-Batasan Implementasi

Beberapa batasan dalam mengimplementasikan aplikasi visualisasi jalur pendakian gunung adalah sebagai berikut:

1. Implementasi aplikasi visualisasi jalur pendakian pada perangkat bergerak yang menggunakan sistem operasi Android.

2. Implementasi aplikasi visualisasi jalur pendakian pada *smartphone* Android yang mempunyai *hardware Geolocation, Compass, Storage (DBMS SQl lite)* untuk kebutuhan pengambilan koordinat pengguna, kompas dan media penyimpanan.
3. Pengunaan aplikasi tidak harus terhubung dengan jaringan internet kecuali untuk penggunaan fitur visualiasi dalam bentuk *Maps*.
4. Pada aplikasi visualisasi jalur pendakian menggunakan satuan meter (m) dan melakukan pembulatan 2 digit dibelakang koma (2 desimal) pada informasi jarak sehingga memberikan informasi lebih jelas.
5. *Developer* mengabaikan ketelitian akurasi pada *receiver* GPS perangkat yang digunakan karena penulisan skripsi ini tidak melakukan penelitian dalam hal tersebut.
6. Penggunaan fitur penunjuk arah dan informasi jarak bisa dilakukan ketika *receive*r GPS mendapatkan nilai *latitude longitude* terbaru dari posisi pengguna.

5.3 Implementasi Fungsi dan *Layout*

Dengan menggunakan metode programming *event-drivent* maka perancangan direalisasikan dengan membuat fungsi dari setiap *event* yang terjadi. Pada Tabel 5.4 menjelaskan mengenai pasangan antara sebuah *event* dengan fungsi beserta file *layout* yang digunakan untuk *output*nya.

Tabel 5.4 Implementasi Fungsi Dan *Layout*

No	Event	Nama Fungsi	File Layout
1	Klik menu	*Function anonymous*()	Features.html
2	Klik menu peta	Map ()	Map.html
3	Klik menu arsip	Data()	Data.html
4	Klik menu petunjuk	*Function anonymous()*	Guide.html
5	Gambar titik jalur pendakian	gambarTitikPendakian()	Map.html
6	Menambahkan tanda teks	posisiImage()	Map.html
7	Menambahkan tanda gambar	posisiTeks()	Map.html
8	Menghapus Tanda	*Function anonymous()*	Map.html

9	Menyimpan jalur pendakian	simpanTitik(), updateDataTitik(), updateDatatanda(), updateNama()	Map.html
10	Melihat informasi jarak	hitungJarak()	Map.html
11	Melihat petunjuk arah	koversiDerajat(), arahPetunjuk()	Map.html
12	Melihat dalam bentuk *Maps*	googleMaps()	Map.html
13	Buka data jalur pendakian	*Function anonymous()*	Data.html
14	Hapus data jalur pendakian	Hapus()	Data.html

5.4 Implementasi *Database*

Pada aplikasi visualisasi jalur pendakian gunung menggunakan *database* untuk menyimpan data jalur pendakian yang telah dibuat oleh pengguna. Implementasi *database* ditunjukkan pada Tabel 5.5.

Tabel 5.5 Implementasi *Database*

Nama Field	***Key***	***Type***	**Deskripsi**
Nama	PK	*String*	*Unique*
keteranganTitik		*String*	*Null*
sumbux		*String*	*Null*
sumbuy		*String*	*Null*
gambarTitik		*String*	*Null*
sumbuxGambar		*String*	*Null*
sumbuyGambar		*String*	*Null*
latitude		*String*	*Null*
longitude		*String*	*Null*
gambarTanda		*String*	*Null*
sumbuxTanda		*String*	*Null*
sumbuytanda		*String*	*Null*

teksTanda		*String*	*Null*
sumbuxTandaTeks		*String*	*Null*
sumbuyTandaTeks		*String*	*Null*

5.5 Implementasi Algoritma

Dalam aplikasi visualisai jalur pendakian terdapat fungsi-fungsi yang di *trigger* ketika adanya sebuah *event* yang diberikan oleh pengguna. Pada penulisan implementasi algoritma ini hanya dicantumkan algoritma dari beberapa proses saja sehingga tidak semua algoritma dicantumkan. Algoritma proses yang dicantumkan antara lain proses menggambarkan titik jalur pendakian, menambahkan tanda teks, menambahkan tanda gambar, menyimpan jalur pendakian, melihat informasi jarak, melihat petunjuk arah, melihat dalam bentuk *maps.*

5.5.1 Implementasi Algoritma Proses Menggambarkan Titik Jalur Pendakian

Menggambarkan titik jalur pendakian dilakukan dengan menekan tombol yang berada dibawah pada halaman menu peta. Penggambaran titik jalur pendakian di gambarkan ketika *receiver* GPS mendapatkan nilai *latitude longitude* dari posisi pengguna. Implementasi algoritma proses menggambarkan titik jalur pendakian ditunjukkan dalam Gambar 5.1

```
1  gambarTitikPendakian = function(pointKe) {
2  document.addEventListener("deviceready", onDeviceReady,
3  false);
4     function onDeviceReady() {
5  var options = {maximumAge: 3000, enableHighAccuracy:
6  true};
7  navigator.geolocation.getCurrentPosition(onSuccess,
8  onError, options);
9          }
10   // onSuccess Geolocation
11 function onSuccess(position) {
```

```
12      var c = document.getElementById("can");
13      var ctx = c.getContext("2d");
14      if (pointKe == 0) {
15      var img = new Image();
16      img.src = "image/titik.png";
17      img.onload = function() {
18      ctx.drawImage(img, (screen.availWidth / 2) - 35,
19  (2000 - 30));};
20      ctx.beginPath();
21      ctx.font = "18px Arial";
22      ctx.fillText('Pos Pemberangkatan', (screen.availWidth
23  / 2) - 10, (2000 - 12));
24      ctx.closePath();
25      kordinatx = screen.availWidth / 2;
26      kordinaty = 2000 - 29;
27  latTitik1 = position.coords.latitude;
28  longTitik1 = position.coords.longitude;
29    informasiJarak = '<option value="' + (pointKe + 1) +
30  '" lat="' + latTitik1 + '" long="' + longTitik1 +
31  '">Jarak Ke Pos ' + (pointKe + 1) + '</option>';
32    informasiArah = '<option value="' + (pointKe + 1) + '"
33  lat="' + latTitik1 + '" long="' + longTitik1 + '">Tunjuk
34  Arah Ke Pos ' + (pointKe + 1) + '</option>';
35    keteranganTitik[counterTitik] = 'Pos Pemberangkatan';
36    sumbux[counterTitik] = (screen.availWidth / 2) - 10;
37    sumbuy[counterTitik] = 2000 - 12;
38    gambarTitik[counterTitik] = "image/titik.png";
39      sumbuxGambar[counterTitik] = (screen.availWidth / 2)
40  - 35;
41      sumbuyGambar[counterTitik] = 2000 - 30;
42      lati[counterTitik] = latTitik1;
43      longi[counterTitik] = longTitik1;
44      simpanTitik(namaPeta, keteranganTitik, sumbux,
45  sumbuy, gambarTitik, sumbuxGambar, sumbuyGambar, lati,
46  longi);
```

```
47      counterTitik++;
48      } else {
49      latTitik2 = position.coords.latitude;
50      longTitik2 = position.coords.longitude;
51      jarak = hitungJarak(latTitik1, latTitik2, longTitik1,
52  longTitik2); //hitung jarak
53      arah = konversiDerajat(latTitik1, latTitik2,
54  longTitik1, longTitik2); //hitung arah dalam derajat
55      if (arah < 0) { arah = 360 + arah;}
56     //code untuk menggambarkan titik
57      var imageObj = new Image();
58      imageObj.src = "image/titik2.png";
59      imageObj.onload = function() {
60      ctx.drawImage(imageObj, kordinatx - 35, kordinaty);};
61     // menambahkan list jarak dan arah
62      informasiJarak += '<option value="' + (pointKe + 1) +
63  '" lat="' + latTitik2 + '" long="' + longTitik2 +
64  '">Jarak Ke Pos ' + (pointKe + 1) + '</option>';
65      informasiArah += '<option value="' + (pointKe + 1) +
66  '" lat="' + latTitik2 + '" long="' + longTitik2 +
67  '">Tunjuk Arah Ke Pos ' + (pointKe + 1) + '</option>';
68      ctx.beginPath();
69      ctx.font = "18px Arial";
70      ctx.fillText('Pos ' + (pointKe + 1) + ', jarak : ' +
71  Number(jarak).toFixed(2) + ' m', kordinatx - 10,
72  (kordinaty - 50));
73      ctx.closePath();
74      keteranganTitik[counterTitik] = "Pos " + (pointKe +
75  1) + ", jarak : " + Number(jarak).toFixed(2) + " m";
76      sumbux[counterTitik] = kordinatx - 10;
77      sumbuy[counterTitik] = kordinaty - 50;
78      gambarTitik[counterTitik] = "image/titik2.png";
79      sumbuxGambar[counterTitik] = kordinatx - 35;
80      sumbuyGambar[counterTitik] = kordinaty - 68;
81      lati[counterTitik] = latTitik2;
```

```
82      longi[counterTitik] = longTitik2;
83      updateDataTitik(namaPeta, keteranganTitik, sumbux,
84  sumbuy, gambarTitik, sumbuxGambar, sumbuyGambar, lati,
85  longi);
86      counterTitik++;
87  tandaSumbux = kordinatx - 60;
88  tandaSumbuy = kordinaty - 68;
89      tandaTeksSumbux = kordinatx - 130;
90      tandaTeksSumbuy = kordinaty - 50
91      kordinaty = kordinaty - 69;
92      latTitik1 = latTitik2;
93      longTitik1 = longTitik2;}}}
94  function onError(error) {
95    alert('code: ' + error.code + '\n' +'message: ' +
96  error.message + '\n');
97          }
```

Gambar 5.1 Algoritma Proses Menggambarkan Titik Jalur Pendakian

Keterangan *source code* implementasi algoritma proses menggambarkan titik jalur pendakian adalah sebagai berikut :

1. Baris 7 merupakan fungsi untuk mengambil nilai *latitude longitude*, jika berhasil maka akan mengeksekusi fungsi `onSucces();`
2. Baris 15 - 23 dan 57 - 73 implementasi untuk menggambarkan titik dan keterangan informasi jarak.
3. Baris 44 - 46 dan 83 - 85 implementasi untuk menyimpan data titik jalur pendakian dalam *database*.
4. Baris 94-97 yaitu fungsi `onError()` untuk menangani kesalahan ketika pengambilan nilai *latitude longitude* tidak berhasil.

5.5.2 Implementasi Algoritma Proses Menambahkan Tanda Teks

Menambahkan tanda teks bisa dilakukan setelah menggambarkan sebuah titik jalur pendakian. Teks akan diletakkan disebelah kiri dari titik jalur pendakian. Implementasi algoritma menambahkan tanda teks ditunjukkan dalam Gambar 5.2.

```
1       posisiTeks = function(namaPeta, teks, sumbux,
2   sumbuy) {
3           var c = document.getElementById("can");
4           var ctx = c.getContext("2d");
5           if (sumbux) {
6               ctx.beginPath();
7               ctx.font = "18px Arial";
8               ctx.fillText(teks, sumbux, sumbuy);
9               ctx.closePath();
10              tandaTeks[counterTandaTeks] = teks;
11              sumbuxTandaTeks[counterTandaTeks] = sumbux;
12              sumbuyTandaTeks[counterTandaTeks] = sumbuy;
13              updateDataTanda(namaPeta,          tandaTeks,
14  sumbuxTandaTeks, sumbuyTandaTeks, 'teks');
15              counterTandaTeks++;
16          } else {
17              Lungo.Notification.show();
18              Lungo.Notification.show('info-sign',
19  'Gambarkan Pos Dahulu', 2);
20          }
21      };
```

Gambar 5.2 Algoritma Proses Menambahkan Tanda Teks

Keterangan dari *source code* implementasi algoritma proses menambahkan tanda teks adalah sebagai berikut :

1. Baris 6 - 9 implementasi untuk menambahkan tanda teks disamping kiri titik jalur pendakian.
2. Baris 13 - 12 implementasi untuk menyimpan data teks kedalam *database*.
3. Baris 17 – 19 implementasi untuk memberikan *alert* kesalahan ketika titik jalur pendakian belum digambarkan.

5.5.3 Implementasi Algoritma Proses Menambahkan Tanda Gambar

Menambahkan tanda gambar dilakukan sama seperti menambahkan tanda teks yang telah dijelaskan diatas. Hanya saja tanda gambar sudah disediakan

secara langsung meskipun terbatas. Pengguna langsung menekan *icon* gambar yang disediakan untuk dijadikan sebuah tanda pada sebuah titik jalur pendakian. Implementasi algoritma menambahkan tanda gambar ditunjukkan dalam Gambar 5.3.

```
1   posisiImage = function(namaPeta, gambar, sumbux,
2   sumbuy) {
3      var c = document.getElementById("can");
4      var ctx = c.getContext("2d");
5      var img = new Image();
6      img.src = gambar;
7      if (sumbux) {
8      img.onload = function() {
9        ctx.drawImage(img, sumbux, sumbuy);
10    };
11     tandaGambar[counterTandaGambar] = gambar;
12     sumbuxTandaGambar[counterTandaGambar] = sumbux;
13     sumbuytandaGambar[counterTandaGambar] = sumbuy;
14     updateDataTanda(namaPeta, tandaGambar,
15  sumbuxTandaGambar, sumbuytandaGambar, 'gambar');
16     counterTandaGambar++;
17     } else {
18      Lungo.Notification.show();
19      Lungo.Notification.show('info-sign', 'Gambarkan Pos
20  Dahulu', 2);
21          };
22  };
```

Gambar 5.3 Algoritma Proses Menambahkan Tanda Gambar

Keterangan *source code* implementasi algoritma proses menambahkan tanda gambar adalah sebagai berikut :

1. Baris 5 – 6 inisialisasi objek gambar dan memasukkan data *path* gambar yang akan ditambahkan disamping titik jalur pendakian.
2. Baris 8 – 10 implementasi untuk menggambarkan tanda gambar disamping kiri titik jalur pendakian.

3. Baris 14 – 15 implementasi untuk menyimpan data tanda gambar kedalam *database*.

5.5.4 Implementasi Algoritma Proses Menyimpan Jalur Pendakian

Menyimpan jalur pendakian merupakan proses untuk menyimpan data jalur pendakian yang sudah di buat oleh pengguna. Proses yang dilakukan untuk menyimpan jalur pendakian dengan menekan tombol simpan yang kemudian memasukkan nama yang di inginkan dan dilanjutkan dengan menekan tombol simpan. Implementasi algoritma proses menyimpan jalur pendakian ditunjukkan dalam Gambar 5.4.

```
1   simpanTitik = function(namaPeta, keteranganTitik,
2   sumbux, sumbuy, gambarTitik, sumbuxGambar,
3   sumbuyGambar, latitude, longitude) {
4     document.addEventListener("deviceready",
5   onDeviceReady, false);
6     function populateDB(tx) {
7     var Keterangan = JSON.stringify(keteranganTitik);
8     var posisiSumbux = JSON.stringify(sumbux);
9     var posisiSumbuy = JSON.stringify(sumbuy);
10    var pathGambar = JSON.stringify(gambarTitik);
11    var posisiGambarx = JSON.stringify(sumbuxGambar);
12    var posisiGambary = JSON.stringify(sumbuyGambar);
13    var lat = JSON.stringify(latitude);
14    var long = JSON.stringify(longitude);
15    tx.executeSql('CREATE TABLE IF NOT EXISTS MAP (nama
16  unique,keteranganTitik,sumbux,sumbuy,gambarTitik,sumbux
17  Gambar,sumbuyGambar,latitude,longitude,gambarTanda,sumb
18  uxTanda,sumbuytanda,teksTanda,sumbuxTandaTeks,sumbuyTan
19  daTeks)');
20     tx.executeSql("INSERT INTO MAP (nama,
21  keteranganTitik, sumbux, sumbuy, gambarTitik,
22  sumbuxGambar, sumbuyGambar, latitude, longitude) VALUES
23  ('" + namaPeta + "', '" + Keterangan + "','" +
24  posisiSumbux + "','" + posisiSumbuy + "','" +
```

```
25  pathGambar + "','" + posisiGambarx + "','" +
26  posisiGambary + "','" + lat + "','" + long + "')");}
27          function errorCB(err) {
28              Lungo.Notification.error('Error', '' +
29  err.message + '', 'remove', 1);}
30          function successCB() {
31              Lungo.Notification.success('Info', 'Peta
32  berhasil di simpan', 'ok', 1);}
33          function onDeviceReady() {
34              var db = window.openDatabase("hiking",
35  "1.0", "Cordova Demo", 200000);
36              db.transaction(populateDB, errorCB,
37  successCB);}
38      };
```

Gambar 5.4 Algoritma Proses Menyimpan Jalur Pendakian

Keterangan *source code* implementasi algoritma proses menyimpan jalur pendakian adalah sebagai berikut :

1. Baris 7 – 14 implementasi untuk merubah format *array* menjadi format JSON.
2. Baris 15 – 19 implementasi *query* untuk membuat tabel bernama MAP.
3. Baris 10 - 26 implementasi *query* untuk memasukkan data jalur pendakian sesuai nama kolom dalam tabel MAP.
4. Baris 27 – 29 implementasi untuk memberi pesan peringatan kesalahan dalam proses penyimpanan.
5. Baris 30 -32 implementasi untuk memberi pesan sukses dalam proses penyimpanan.
6. Baris 33 – 38 implementasi untuk mengeksekusi fungsi untuk memasukkan data kedalam *database*.

5.5.5 Implementasi Algoritma Proses Melihat Informasi Jarak

Melihat informasi jarak dilakukan dengan menekan tombol info jarak kemudian menentukan informasi jarak ke pos berapa yang diinginkan. Dalam melihat informasi jarak maka sistem harus mendapatkan nilai *latitude longitude* terbaru dari posisi pengguna. Implementasi algoritma melihat informasi jarak ditunjukkan dalam Gambar 5.5.

```
1   hitungJarak = function(lat1, lat2, long1, long2) {
2           var R = 6371; // km
3           var dLat = (lat2 - lat1) * Math.PI / 180;
4           var dLon = (long2 - long1) * Math.PI / 180;
5           var lati1 = lat1 * Math.PI / 180;
6           var lati2 = lat2 * Math.PI / 180;
7           var a = (Math.sin(dLat / 2) * Math.sin(dLat /
8   2)) +
9                   Math.sin(dLon / 2) * Math.sin(dLon / 2)
10  * Math.cos(lati1) * Math.cos(lati2);
11          var c = 2 * Math.atan2(Math.sqrt(a),
12  Math.sqrt(1 - a));
13          var jarak = R * c;
14          return jarak * 1000;
15      };
```

Gambar 5.5 Algoritma Proses Melihat Informasi Jarak

Keterangan *source code* implementasi algoritma proses melihat informasi jarak adalah sebagai berikut :

1. Baris 2 merupakan nilai dari rata-rata radius bumi dalam kilometer (km).
2. Baris 3 – 6 implementasi untuk mengubah nilai *latitude longitude* dalam radian.
3. Baris 7 – 14 merupakan implementasi formula Haversine dalam menghitung jarak berdasarkan nilai *latitude longitude*.

5.5.6 Implementasi Algoritma Proses Melihat Petunjuk Arah

Melihat petunjuk arah dilakukan sama seperti dengan melihat informasi jarak. Pengguna memilih ke pos berapa arah yang ingin ditunjukkan. Dalam menentukan arah maka juga harus mendapatkan nilai *latitude longitude* terbaru dari posisi pengguna. Implementasi algoritma proses melihat petunjuk arah ditunjukkan dalam Gambar 5.6.

```
1  konversiDerajat = function(lat1, lat2, long1, long2) {
2    var dLon = (long2 - long1) * (Math.PI / 180);
3    var lati1 = lat1 * Math.PI / 180;
4    var lati2 = lat2 * Math.PI / 180;
5    var y = Math.sin(dLon) * Math.cos(lati2);
6    var x = Math.cos(lati1) * Math.sin(lati2) -
7          Math.sin(lati1) * Math.cos(lati2) *
8  Math.cos(dLon);
9    var derajat = Math.atan2(y, x) / (Math.PI / 180);
10         return derajat;
11     };
12 arahPetunjuk = function(arah) {
13         var watchID = null;
14         document.addEventListener("deviceready",
15 onDeviceReady, false);
16         function onDeviceReady() {
17             startWatch();
18         }
19         function startWatch() {
20             // Update compass every 500 miliseconds
21             var options = {frequency: 500};
22             petunjuk = arah;
23             watchID =
24 navigator.compass.watchHeading(onSuccess, onError,
25 options);}
26  function onSuccess(heading) {
27    var newHeading = Math.round(heading.magneticHeading);
```

```
28      var arah0 = 360 - newHeading;
29      if (petunjuk < 0) {
30      petunjuk = 360 + petunjuk;
31      var magnetik = 360 - arah;}
32      if (petunjuk > arah0) {
33        posisiArahKompas = (petunjuk + arah0);
34        $("#arah").css('-webkit-transform', 'rotate(' +
35  posisiArahKompas + 'deg)');
36      }else if(petunjuk < arah0){
37         posisiArahKompas = (arah0- petunjuk);
38         $("#arah").css('-webkit-transform', 'rotate(' +
39  posisiArahKompas + 'deg)');
40      }else {
41      $("#arah").css('-webkit-transform', 'rotate(' + 0
42  + 'deg)'); }
43          }
44     function onError(compassError) {
45        alert('Compass error: ' + compassError.code);}
46      };
```

Gambar 5.6 Algoritma Proses Melihat Petunjuk Arah

Keterangan *source code* dari implementasi algoritmas proses melihat petunjuk arah adalah sebagai berikut :

1. Baris 1 – 10 implementasi dalam menghitung arah dalam nilai derajat menggunakan formula *forward* azimuth.
2. Baris 19 – 25 implementasi fungsi untuk menggunakan *hardware compass* perangkat dan mengeksekusi fungsi `onSucces()` ketika berhasil dan `onError()` ketika gagal.
3. Baris 26 – 43 implementasi fungsi `onSucces()` untuk menunjukkan arah dengan sebuah icon panah dalam halaman menu peta.
4. Baris 44 – 46 implementasi fungsi `onError()` untuk memberikan pesan peringatan kesalahan dalam menggunakan *hardware compass*.

5.5.7 Implementasi Algoritma Proses Melihat Jalur Pendakian Bentuk *Map*

Melihat jalur pendakian bentuk *map* dilakukan dengan cara menekan tombol *Map*. Penggunaan fitur ini bisa dilakukan ketika terdapat koneksi internet. Implementasi algoritma proses melihat jalur pendakian bentuk *Map* ditunjukkan dalam Gambar 5.7.

```
1   googlemaps = function() {
2           document.addEventListener("deviceready",
3   onDeviceReady, false);
4       function onDeviceReady() {
5       var path = [];var marker = [];var markers = [];
6       var infowindow = [];
7
8       var mapOptions = {
9         center: new google.maps.LatLng(lati[0], longi[0]),
10        zoom: 15,
11        mapTypeId: google.maps.MapTypeId.ROADMAP
12      };
13      var map = new
14  google.maps.Map(document.getElementById("petagoogle"),
15  mapOptions);
16      for (var i = 0; i < lati.length; i++) {
17         path.push(new google.maps.LatLng(lati[i],
18  longi[i]));
19          markers[i] = new google.maps.Marker({
20                        position: new
21  google.maps.LatLng(lati[i], longi[i]),
22                        map: map,
23                        title: keteranganTitik[i],
24                        infowindowindex: i
25                    });
26          var info = new google.maps.InfoWindow({
27                        content: keteranganTitik[i]
28                    });
29          marker.push(markers[i]);
```

```
30          infowindow.push(info);
31  google.maps.event.addListener(markers[i], 'click',
32  function() {
33  infowindow[this.infowindowindex].open(map, this);});
34           }
35     var flightPath = new google.maps.Polyline({
36          path: path,
37          geodesic: true,
38          strokeColor: '#FF0000',
39          strokeOpacity: 1.0,
40          strokeWeight: 2
41               });
42    flightPath.setMap(map);
43     }
44  };
```

Gambar 5.7 Algoritma Proses Melihat Jalur Pendakian Bentuk Maps

Keterangan *source code* implementasi algoritma proses melihat jalur pendakian bentuk *map* adalah sebagai berikut :

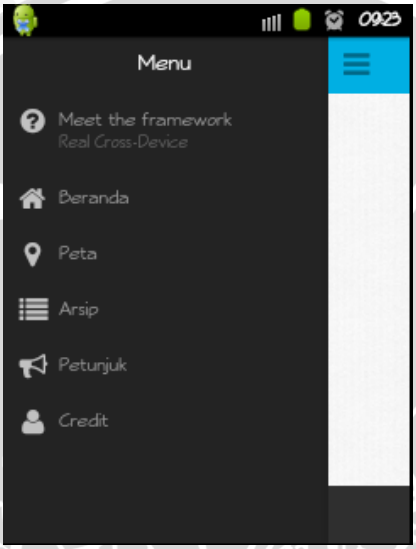
1. Baris 8 – 15 implementasi untuk membuat sebuah peta dari tempat berdasarkan nilai *latitude longitude* yang dimasukkan.
2. Baris 16 – 33 implementasi untuk menggambarkan *marker* (titik) dan informasi tiap-tiap *marker* pada *map*.
3. Baris 35 - 42 implementasi untuk menggambarkan garis penghubung dari titik-titik yang sudah digambarkan.

5.6 Implementasi Antarmuka

Implementasi antarmuka aplikasi visualisasi jalur pendakian terdiri dari halaman beranda, halaman peta, halaman arsip dan halaman petunjuk penggunaan.

5.6.1 Antarmuka Pilihan Menu

Antarmuka pilihan menu merupakan antarmuka untuk menampilkan menu yang terdapat dalam aplikasi visualisasi jalur pendakian. Terdapat lima menu yaitu halaman beranda, peta, arsip, petunjuk dan *credit*. Implementasi antarmuka pilihan menu ditunjukkan dalam Gambar 5.8.



Gambar 5.8 Antarmuka Pilihan Menu.

5.6.2 Antarmuka Menu Peta

Halaman peta merupakan halaman untuk memvisualisasikan jalur pendakian yang di buat oleh pengguna. Berikut antarmuka dalam halaman peta.

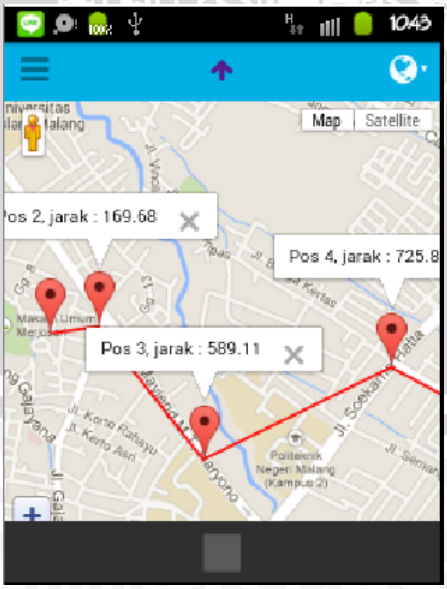
5.6.2.1 Antarmuka Jalur Pendakian

Antarmuka jalur pendakian merupakan antarmuka ketika pengguna membuat sebuah jalur pendakian. Pada antarmuka jalur pendakian terdapat titik dari pos-pos pendakian yang terdapat keterangan disebelah kanan berupa informasi jarak terhadap titik sebelumnya dan sebuah tanda berupa gambar atau teks disebelah kiri titik yang menjadi patokan pendaki dari masing-masing pos. Implementasi antarmuka Jalur pendakian ditunjukkan dalam Gambar 5.9.

Gambar 5.9 Antarmuka Jalur Pendakian

5.6.2.2 Antarmuka Jalur Pendakian Dalam *Map*

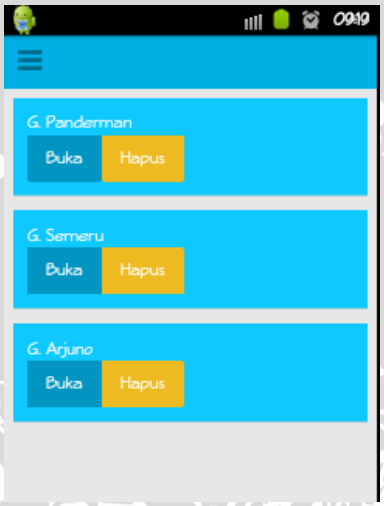
Antarmuka jalur pendakian dalam *map* merupakan antarmuka dari jalur pendakian yang menggunakan API google *map* dalam memvisualisasikan jalur pendakian. Fitur ini bisa digunakan jika terdapat jaringan internet di tempat pendakian. Implementasi dari antarmuka jalur pendakian dalam maps ditunjukkan dalam Gambar 5.10.



Gambar 5.10 Antarmuka Jalur Pendakian Dalam *Map*

5.6.3 Antarmuka Halaman Arsip

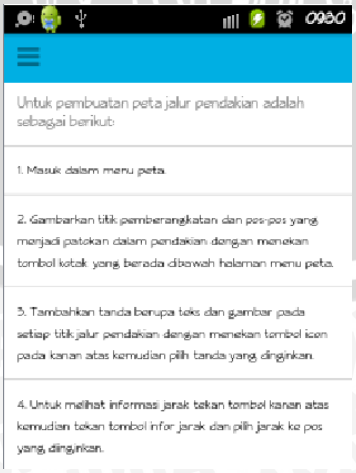
Halaman arsip merupakan antarmuka untuk menampilkan daftar jalur pendakian yang tersimpan dalam *database* perangkat pengguna. Dalam antarmuka halaman arsip terdapat nama dari jalur pendakian serta dua tombol yaitu tombol buka dan tombol hapus dimasing-masing jalur pendakian. Implementasi antarmuka halaman arsip ditunjukkan dalam Gambar 5.11.



Gambar 5.11 Antarmuka Halaman Arsip

5.6.5 Antarmuka Halaman Petunjuk

Halaman petunjuk merupakan antarmuka yang berisi cara penggunaan aplikasi jalur pendakian gunung. Implementasi antarmuka halaman petunjuk ditunjukkan dalam Gambar 5.12.



Gambar 5.12 Antarmuka Halaman Petunjuk

BAB VI

PENGUJIAN DAN ANALISIS

Bab ini membahas mengenai tahapan pengujian dan analisis perangkat lunak visualisasi jalur pendakian yang telah di implementasikan sebelumnya. Pengujian dilakukan melalui tiga tahapan yaitu pengujian Apache Cordova, pengujian perbandingan perhitungan manual dengan aplikasi serta pengujian validasi. Setelah tahapan pengujian selesai maka dilanjutkan dengan analisis terhadap hasil pengujian di setiap tahap pengujian.

6.1 Pengujian

Proses pengujian dilakukan dalam tiga tahapan yaitu pengujian Apache Cordova dalam mengakses *hardware* perangkat bergerak, pengujian perbandingan perhitungan manual dengan aplikasi serta pengujian validasi.

6.1.1 Pengujian Apache Cordova

Pengujian Apache Cordova dilakukan untuk mengetahui apakah API Apache Cordova dapat menggakses *hardware* dari perangkat bergerak yang dibutuhkan aplikasi visualisasi jalur pendakian dengan menggunakan javascript. Pengujian Apache Cordova ini menggunakan strategi pegujian validasi.

6.1.1.1 Kasus Uji Mengakses *Hardware Geolocation*

Tabel 6.1 Kasus Uji Mengakses *Hardware Geolocation*

Nama Kasus Uji	Kasus Uji Mengakses *Hardware Geolocation*
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui apakah Apache Cordova dapat mengakses *hardware geolocation* dengan menggunakan javascript dan mendapatkan nilai *latitude longitude*.
Prosedur Uji	1. Menjalankan fungsi untuk mengakses *geolocation* dan menampilkan nilai *latitude longitude* posisi pengguna.

Hasil yang diharapkan	API Apache Cordova dapat mengakses *geolocation* dan memberikan nilai *latitude longitude*.

6.1.1.2 Kasus Uji Mengakses *Hardware Compass*

Tabel 6.2 Kasus Uji Mengakses *Hardware Compass*

Nama Kasus Uji	Kasus Uji Mengakses *Hardware Compass*
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui apakah Apache Cordova dapat mengakses *hardware compass* dengan menggunakan javascript.
Prosedur Uji	1. Menjalankan fungsi untuk mengakses *compass* dan memberikan nilai *magnetic heading* dari perangkat.
Hasil yang diharapkan	API Apache cordova dapat mengakses *compass* dan memberikan nilai *magnetic heading* perangkat.

6.1.1.3 Kasus Uji Mengakses *Hardware Storage*

Tabel 6.3 Kasus Uji Mengakses *Hardware Storage*

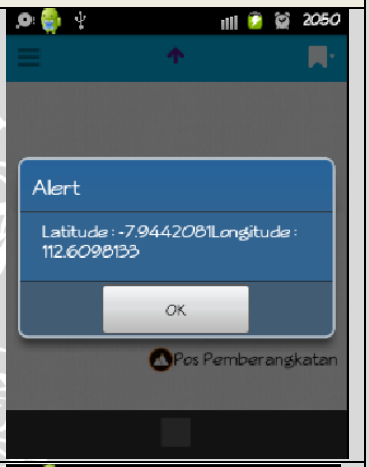
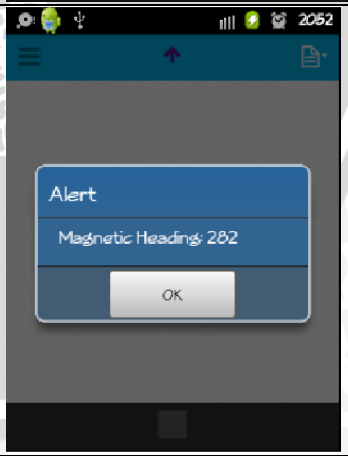
Nama Kasus Uji	Kasus Uji Mengakses *Hardware Storage*
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui apakah Apache Cordova dapat mengakses *hardware storage* dengan menggunakan javascript sehingga dapat menyimpan data dalam perangkat bergerak.
Prosedur Uji	1. Menjalankan fungsi untuk mengakses *storage* dan menyimpan sebuah data. 2. Keluar dari aplikasi yang dijalankan. 3. Menjalankan aplikasi kembali dan menampilkan data yang telah disimpan.

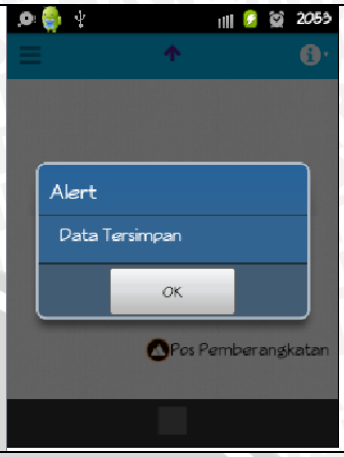
Hasil yang diharapkan	API Apache Cordova dapat mengakses *storage* dan dapat menyimpan data.

6.1.1.4 Hasil Pengujian Apache Cordova

Hasil pengujian Apache Cordova pada setiap kasus uji akan dijabarkan pada Tabel 6.4.

Tabel 6.4 Hasil Pengujian Apache Cordova

Nama Kasus Uji	Hasil yang diharapkan	Hasil yang didapatkan	Tampilan Validitas
Kasus Uji Mengakses *Hardware Geolocation*	API Apache Cordova dapat mengakses *geolocation* dan memberikan nilai *latitude longitude*.	API Apache Cordova dapat mengakses *geolocation* dan memberikan nilai *latitude longitude*.	
Kasus Uji Mengakses *Hardware Compass*	API Apache cordova dapat mengakses *compass* dan memberikan nilai *magnetic heading* perangkat.	API Apache cordova dapat mengakses *compass* dan memberikan nilai *magnetic heading* perangkat.	

Kasus Uji Mengakses *Hardware Storage*	API Apache Cordova dapat mengakses *storage* dan dapat menyimpan data.	API Apache Cordova dapat mengakses *storage* dan dapat menyimpan data.	

6.1.2 Pengujian Perbandingan Perhitungan Manual dengan Aplikasi

Pengujian perbandingan perhitungan merupakan pengujian fungsionalitas untuk mengetahui apakah aplikasi sudah dapat memenuhi kebutuhan untuk melakukan perhitungan informasi jarak dengan menerapkan formula Haversine dan perhitungan arah dalam derajat dengan menerapkan formula *Forward* Azimuth berdasarkan nilai *latitude longitude*. Hal ini dilakukan untuk melihat kecocokan hasil perhitungan secara manual dengan perhitungan aplikasi.

Tabel 6.5 Kasus Uji Perhitungan Manual dan Perhitungan Aplikasi

Nama Kasus Uji	Kasus Uji Perhitungan Manual dan Perhitungan Aplikasi
Tujuan Pengujian	Pengujian dilakukan untuk mengetahui apakah aplikasi mampu melakukan proses perhitungan seperti perhitungan manual dengan hasil yang sama.
Prosedur Uji	1. Menjalankan aplikasi untuk melakukan proses komputasi perhitungan. 2. Melakukan proses perhitungan secara manual. 3. Membandingkan hasil perhitungan manual dengan aplikasi.
Hasil yang diharapkan	Hasil perhitungan manual sama dengan perhitungan yang di lakukan oleh aplikasi.

6.1.2.1 Perhitungan Aplikasi

Proses perhitungan aplikasi dilakukan dengan memasukkan nilai latitude longitude yang juga digunakan nantinya dalam perhitungan manual.

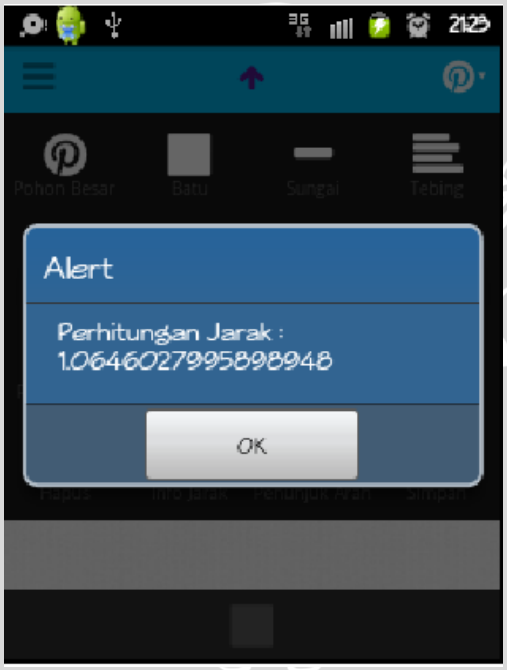
1. **Perhitungan Jarak**

   Perhitungan jarak oleh aplikasi dengan nilai masukan *latitude longitude* sebagai berikut :

   Nilai *latitude longitude* 1 : 40.7486, -73.9864

   Nilai *latitude longitude* 2 : 40.7580, -73.9840

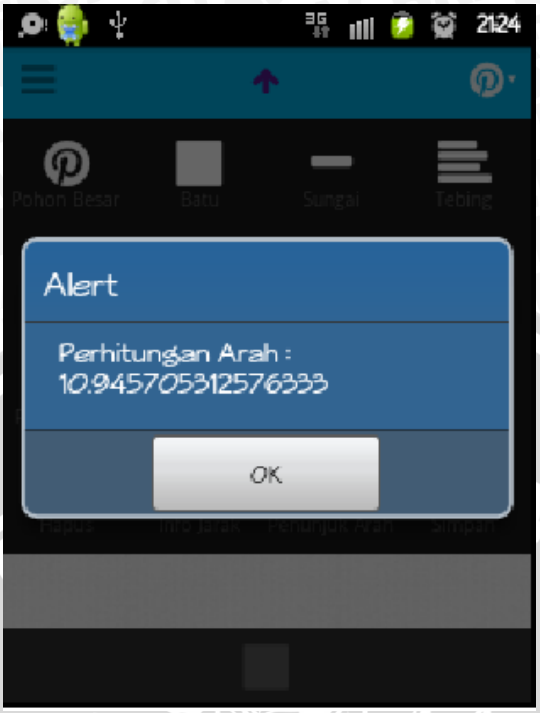
   Hasil proses komputasi dari aplikasi ditunjukkan dalam Gambar 6.1



Gambar 6.1 Hasil Perhitungan Jarak Aplikasi

2. **Perhitungan Arah**

   Perhitungan arah oleh aplikasi juga menggunakan nilai masukkan *latitude longitude* yang sama dengan perhitungan jarak. Hasil komputasi dari aplikasi ditunjukkan dalam Gambar 6.2.



Gambar 6.2 Hasil Perhitungan Arah Aplikasi

6.1.2.2 Perhitungan Manual

Perhitungan manual yang dilakukan dalam menghitung jarak dan arah dalam derajat berdasarkan nilai *latitude longitude* adalah sebagai berikut :

1. **Perhitungan Jarak**

   Contoh kasus penerapan formula Haversine :

   Nilai *latitude longitude* 1 : 40.7486, -73.9864

   Nilai *latitude longitude* 2 : 40.7580, -73.9840

   Perhitungan formula Haversine sebagai berikut :

   $a = \sin^2(\Delta\varphi/2) + \cos(\varphi_1).\cos(\varphi_2).\sin^2(\Delta\lambda/2)$

   $a = \sin^2(0.0094/2) + \cos(40.7486).\cos(40.7580).\sin^2(0.0024/2)$

   $a = 0.00000000672899882872562 5 + 0.757580934088 \text{ x } 0.757473834545 \text{ x } 0.000000000438649083490401$

   $a = 0.000000006980716611799625$

$c = 2.\text{atan2}(\sqrt{a}, \sqrt{(1-a)})$

$c = 2.\text{atan2}(\sqrt{0.00000000698071661179962 5}, \sqrt{(1-0.00000000698071661179962 5)})$

$c = 0.00016710136597937306$

$d = R.c$

$d = 6371 \times 0.00016710136597937306$

$d = 1.06460280265458576526$ km

2. **Perhitungan Arah**

Contoh kasus penerapan formula *forward* Azimuth :

Nilai *latitude longitude* 1 : 40.7486, -73.9864

Nilai *latitude longitude* 2 : 40.7580, -73.9840

Perhitungan formula *forward* Azimuth sebagai berikut :

$\theta = \text{atan2}(\sin(\Delta\lambda).\cos(\varphi 2), \cos(\varphi 1).\sin(\varphi 2) - \sin(\varphi 1).\cos(\varphi 2).\cos(\Delta\lambda))$

$\theta = \text{atan2}(\sin(0.0024).\cos(40.7580), \cos(40.7486).\sin(40.7580) - \sin(40.7486).\cos(40.7580).\cos(0.0024))$

$\theta = 10°$

6.1.2.3 Hasil Pengujian Perbandingan Perhitungan Manual dan Aplikasi

Hasil pengujian yang dilakukan antara perhitungan manual dan aplikasi di sajikan dalam bentuk tabel perbandingan dengan mengukur selisih hasil perhitungan dari keduanya. Tabel perbandingan perhitungan manual dengan aplikasi ditunjukkan pada Tabel 6.6.

Tabel 6.6 Hasil Pengujian Perhitungan Manual dan Aplikasi

No	Perhitungan	Manual	Aplikasi	Selisih
1	Jarak	1.064602802654585 76526 km	1.0646027995898 948 km	0.00000000030 646 km
2	Arah	10°	10.945705312576 333°	0.94570531257 6333°

Tabel 6.6 diatas merupakan hasil dari pengujian perbandingan perhitungan manual dengan aplikasi. Pada tabel diatas ditunjukkan hasil perhitungan secara manual serta perhitungan dari aplikasi serta selisih dari perhitungan manual dengan perhitungan aplikasi. Selisih antara perhitungan manual dan aplikasi digunakan untuk mengetahui tingkat ketelitian perhitungan aplikasi terhadap perhitungan yang dilakukan secara manual.

6.1.3 Pengujian Validasi

Pengujian validasi digunakan untuk mengetahui apakah sistem yang di bangun sudah benar sesuai dengan kebutuhan. Item-item yang telah di rumuskan dalam daftar kebutuhan akan menjadi acuan untuk melakukan pengujian validasi. Pengujian validasi juga merupakan pengujian dari Lungo Js yang menggunakan HTML 5 dan CSS 3 dalam mengimplementasikan antarmuka aplikasi pada sebuah aplikasi *native*. Pengujian validasi menggunakan metode *black-box* , karena tidak diperlukan konsentrasi terhadap alur jalannya algoritma program dan lebih di tekankan untuk menemukan konformitas antara kinerja sistem dengan daftar kebutuhan.

6.1.3.1 Kasus Uji Validasi

Untuk mengetahui kesesuaian antara kebutuhan dan kinerja sistem , pada setiap kebutuhan sistem dilakukan proses pengujian dengan kasus uji masing-masing yang ditunjukkan pada tabel-tabel dibawah ini :

Tabel 6.7 Kasus Uji Melihat Halaman Peta

Nama Kasus Uji	Kasus Uji Melihat Halaman Peta
Objek Uji	Kebutuhan Fungsional FR _001
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional untuk membuka halaman peta.
Prosesur Uji	1. Pengguna menekan tomblol menu peta.
Hasil yang Diharapkan	Aplikasi dapat menampilkan halaman peta.
Hasil yang Didapatkan	Aplikasi dapat menampilkan halaman peta.
Status Validasi	Valid

Tabel 6.7 menjelaskan tahap pengujian melihat halaman peta yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.8 Kasus Uji Melihat Titik Jalur Pendakian

Nama Kasus Uji	Kasus Uji Melihat Titik Jalur Pendakian
Objek Uji	Kebutuhan Fungsional FR _002
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional untuk menggambarkan titik jalur pendakian.
Prosesur Uji	1. Pengguna menekan tombol gambarkan titik jalur pendakian.
Hasil yang Diharapkan	Aplikasi dapat menggambarkan titik jalur pendakian.
Hasil yang Didapatkan	Aplikasi dapat menggambarkan titik jalur pendakian.
Status Validasi	Valid

Tabel 6.8 menjelaskan tahap pengujian melihat titk jalur pendakian yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.9 Kasus Uji Menambahkan Tanda

Nama Kasus Uji	Kasus Uji Menambahkan Tanda
Objek Uji	Kebutuhan Fungsional FR _003
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional untuk menambahkan tanda.
Prosesur Uji	1. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 2. Penguna menambahkan tanda gambar atau teks.
Hasil yang Diharapkan	Aplikasi dapat menambahkan tanda gambar atau teks.
Hasil yang Didapatkan	Aplikasi dapat menambahkan tanda gambar atau teks.
Status Validasi	Valid

Tabel 6.9 menjelaskan tahap pengujian menambahkan tanda yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.10 Kasus Uji Menghapus Tanda

Nama Kasus Uji	Kasus Uji Menghapus Tanda
Objek Uji	Kebutuhan Fungsional FR _004
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam

	menghapus tanda.
Prosesur Uji	1. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 2. Pengguna menekan tombol hapus.
Hasil yang Diharapkan	Aplikasi dapat menghapus tanda.
Hasil yang Didapatkan	Aplikasi dapat menghapus tanda.
Status Validasi	Valid

Tabel 6.10 menjelaskan tahap pengujian menghapus tanda yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.11 Kasus Uji Melihat Arah Kembali

Nama Kasus Uji	Kasus Uji Melihat Arah Kembali
Objek Uji	Kebutuhan Fungsional FR _005
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam menunjukkan arah kembali.
Prosesur Uji	1. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 2. Pengguna menekan tombol arah 3. Pengguna menentukan arah ke sebuah titik jalur pendakian. 4. Pengguna menekan tombol simpan.
Hasil yang Diharapkan	Aplikasi dapat menampilkan menunjukkan arah.
Hasil yang Didapatkan	Aplikasi dapat menampilkan menunjukkan arah.
Status Validasi	Valid

Tabel 6.11 menjelaskan tahap pengujian melihat arah kembali yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung.

Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.12 Kasus Uji Melihat Informasi Jarak

Nama Kasus Uji	Kasus Uji Melihat Informasi Jarak
Objek Uji	Kebutuhan Fungsional FR _006
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam menunjukkan informasi jarak.
Prosesur Uji	1. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 2. Pengguna menekan tombol info jarak 3. Pengguna menentukan jarak ke sebuah titik jalur pendakian. 4. Pengguna menekan tombol lihat.
Hasil yang Diharapkan	Aplikasi dapat menampilkan informasi jarak.
Hasil yang Didapatkan	Aplikasi dapat menampilkan informasi jarak.
Status Validasi	Valid

Tabel 6.12 menjelaskan tahap pengujian melihat informasi jarak yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.13 Kasus Uji Melihat Jalur Pendakian Dalam *Maps*

Nama Kasus Uji	Kasus Uji Melihat Jalur Pendakian Dalam *Maps*
Objek Uji	Kebutuhan Fungsional FR _007
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam melihatkan jalur pendakian dalam maps.
Prosesur Uji	1. Pengguna menekan tombol pilihan diatas

	kanan halaman menu peta. 2. Pengguna menekan tombol maps.
Hasil yang Diharapkan	Aplikasi dapat menampilkan jalur pendakian dalam bentuk maps.
Hasil yang Didapatkan	Aplikasi dapat menampilkan jalur pendakian dalam bentuk maps.
Status Validasi	Valid

Tabel 6.13 menjelaskan tahap pengujian melihat jalur pendakian dalam *maps* yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.14 Kasus Uji Menyimpan Data

Nama Kasus Uji	Kasus Uji Menyimpan Data
Objek Uji	Kebutuhan Fungsional FR _008
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam menyimpan data.
Prosesur Uji	1. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 2. Pengguna menekan tombol simpan. 3. Pengguna memasukkan nama jalur pendakian. 4. Pengguna menekan tombol simpan.
Hasil yang Diharapkan	Aplikasi dapat menyimpan data jalur pendakian.
Hasil yang Didapatkan	Aplikasi dapat menyimpan data jalur pendakian.
Status Validasi	Valid

Tabel 6.14 menjelaskan tahap pengujian menyimpan data yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian

yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.15 Kasus Uji Membuat Peta Baru

Nama Kasus Uji	Kasus Uji Membuat Peta Baru
Objek Uji	Kebutuhan Fungsional FR _009
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam membuat peta baru.
Prosesur Uji	1. Pengguna menggambarkan sebuah titik jalur pendakian. 2. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 3. Pengguna menekan tombol baru.
Hasil yang Diharapkan	Aplikasi dapat membuat jalur pendakian baru.
Hasil yang Didapatkan	Aplikasi dapat membuat jalur pendakian baru.
Status Validasi	Valid

Tabel 6.15 menjelaskan tahap pengujian membuat peta baru yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.16 Kasus Uji Melihat Halaman Arsip

Nama Kasus Uji	Kasus Uji Melihat Halaman Arsip
Objek Uji	Kebutuhan Fungsional FR _010
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam melihat halaman arsip.
Prosesur Uji	1. Pengguna menekan tombol menu arsip
Hasil yang Diharapkan	Aplikasi dapat menampilkan halaman arsip.

Hasil yang Didapatkan	Aplikasi dapat menampilkan halaman arsip.
Status Validasi	Valid

Tabel 6.16 menjelaskan tahap pengujian melihat halaman arsip yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.17 Kasus Uji Membuka Data Jalur Pendakian

Nama Kasus Uji	Kasus Uji Membuka Data Jalur Pendakian
Objek Uji	Kebutuhan Fungsional FR _011
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam membuka data jalur pendakian.
Prosesur Uji	1. Pengguna menekan tombol menu arsip. 2. Penguna menekan tombol buka
Hasil yang Diharapkan	Aplikasi dapat membuka data jalur pendakian.
Hasil yang Didapatkan	Aplikasi dapat membuka data jalur pendakian.
Status Validasi	Valid

Tabel 6.17 menjelaskan tahap pengujian membuka data jalur pendakian yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.18 Kasus Uji Menghapus Data Jalur Pendakian

Nama Kasus Uji	Kasus Uji Menghapus Data Jalur Pendakian
Objek Uji	Kebutuhan Fungsional FR _012
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam menghapus data jalur pendakian.

Prosesur Uji	1. Pengguna menekan tombol menu arsip 2. Pengguna menekan tombol hapus 3. Pengguna menekan tombol hapus sebagai konfirmasi.
Hasil yang Diharapkan	Aplikasi dapat menghapus data jalur pendakian
Hasil yang Didapatkan	Aplikasi dapat menghapus data jalur pendakian
Status Validasi	Valid

Tabel 6.18 menjelaskan tahap pengujian menghapus data jalur pendakian yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.19 Kasus Uji Melihat Halaman Petunjuk Penggunaan

Nama Kasus Uji	Kasus Uji Melihat Halaman Petunjuk Pengguna
Objek Uji	Kebutuhan Fungsional FR _013
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat memenuhi kebutuhan fungsional dalam melihat halaman petunjuk pengguaan
Prosesur Uji	1. Pengguna menekan tombol menu petunjuk
Hasil yang Diharapkan	Aplikasi dapat menampilkan halaman petunjuk
Hasil yang Didapatkan	Aplikasi dapat menampilkan halaman petunjuk
Status Validasi	Valid

Tabel 6.19 menjelaskan tahap pengujian melihat halaman petunjuk pengguna yang merupakan kebutuhan fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan fungsional yang berarti status validasi pengujian valid.

Tabel 6.20 Kasus Uji *Reliability* Jalur Pendakian

Nama Kasus Uji	Kasus Uji *Reliability* Jalur Pendakian

Objek Uji	Kebutuhan Non-Fungsional
Tujuan Pengujian	Pengujian dilakukan untuk memastikan aplikasi dapat menyimpan data jalur pendakian ketika perangkat yang digunakan mati secara tiba-tiba.
Prosesur Uji	1. Pengguna menggambarkan sebuah jalur pendakian 2. Pengguna mematikan perangkat yang digunakan
Hasil yang Diharapkan	Aplikasi dapat menyimpan data jalur pendakian ketika perangkat mati tiba-tiba dan data belum disimpan oleh pengguna.
Hasil yang Didapatkan	Aplikasi dapat menyimpan data jalur pendakian ketika perangkat mati tiba-tiba dan data belum disimpan oleh pengguna.
Status Validasi	Valid

Tabel 6.20 menjelaskan tahap pengujian *Reliability* Jalur Pendakian yang merupakan kebutuhan non-fungsional dari aplikasi visualisasi jalur pendakian gunung. Pengujian yang dilakukan mendapatkan hasil sesuai dengan kebutuhan non-fungsional yang berarti status validasi pengujian valid.

6.2 Analisis

Proses analisis bertujuan untuk mendapatkan kesimpulan dari hasil pengujian aplikasi visualisasi jalur pendakian gunung yang telah dilakukan. Analisis dilakukan terhadap hasil pengujian disetiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian Apache Cordova, Pengujian perbandingan perhitungan manual dengan aplikasi dan pengujian validasi.

6.2.1 Analisis Pengujian Apache Cordova

Proses analisis terhadap pengujian Apache Cordova dilakukan dengan melihat kemampuan Apache Cordova dalam mengakses *hardware* perangkat

bergerak. Dari hasil pengujian maka dapat disimpulkan bahwa Apache Cordova dapat mengakses *hardware* dengan menggunakan javascript. Dengan kemampuan ini maka Apache Cordova dapat digunakan dalam memvisualisasikan jalur pendakian yang membutuhkan fungsionalitas dari *hardware* perangkat bergerak seperti *Geolocation, Compass* dan *Storage*.

6.2.2 Analisis Pegujian Perhitungan Manual dan Aplikasi

Proses analisis terhadap hasil pengujian perhitungan manual dan aplikasi dilakukan dengan melihat selisih dari hasil perhitungan keduanya. Dari hasil pengujian didapatkan selisih dari perhitungan jarak adalah sebesar 0.00000000030646 dan selisih perhitungan arah sebesar 0.945705312576333°. Nilai Selisih ini disebabkan perbedaan dalam pembulatan dalam perhitungan yang dihasilkan dalam aplikasi dan manual. Selisih ini tidak memberikan dampak yang signifikan dalam pemberian informasi kepada pengguna karena informasi yang akan diberikan kepada pengguna menggunakan angka 2 desimal atau 2 angka dibelakang koma untuk informasi jarak dan informasi arah menggunakan angkat bulat.

6.2.3 Analisis Pengujian Validasi

Proses analisis terhadap pengujian validasi dilakukan dengan melihat hasil kecocokan antara hasil kinerja sistem dengan daftar kebutuhan. Pegujian validasi juga merupakan pengujian dari implementasi Lungo Js dalam memvisuliasikan jalur pendakian. Dengan menggunakan HTML 5 dan CSS 3, Lungo Js dapat menjadi antarmuka seperti bahasa *native* yang digunakan dalam Android yaitu XML. Maka, berdasarkan hasil pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas perangkat lunak visualisasi jalur pendakian telah memenuhi kebutuhan yang telah dijabarkan pada tahap analisis kebutuhan.

6.2.4 Analisis Kelemahan Sistem

Proses analisis terhadap kelemahan sistem dilakukan berdasarkan kekurangan dari sistem terhadap hal-hal yang tidak dapat diselesaikan secara langsung. Kelemahan sistem yang terdapat pada aplikasi visualisasi jalur pendakian yaitu sistem tidak dapat memberikan sebuah peringatan ketika

pengguna melakukan pengambilan titik jalur pendakian kurang dari 10 meter terhadap titik jalur pendakian sebelumnya sehingga dapat mempengaruhi ketelitian akurasi dari nilai *latitude longitude* GPS.

BAB VII
PENUTUP

7.1 Kesimpulan

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

1. Perancangan dalam aplikasi visualisasi jalur pendakian dapat menggunakan UML (*Unified Model Language*) dan menggunakan diagram EPC (*Event-Drivent Process Chain*).
2. Berdasarkan hasil pengujian yang dilakukan, menjelaskan bahwa implementasi Lungo Js yang merupakan *framework mobile* HTML 5 dan CSS 3 dapat memvisualisasikan jalur pendakian pada aplikasi *mobile* visualisasi jalur pendakian yang merupakan sebuah aplikasi *native* pada perangkat bergerak.
3. Berdasarkan hasil pengujian yang dilakukan, membuktikan bahwa API Apache Cordova dapat mengakses *hardware* perangkat bergerak yang dibutuhkan dalam implementasi aplikasi visualisasi jalur pendakian seperti *Geolocation, Compass* dan *Storage*.
4. Berdasarkan hasil pengujian perhitungan yang dilakukan dengan menggunakan formula Haversine dan *forward* Azimuth yang diterapkan pada aplikasi mampu melakukan perhitungan dengan hasil yang sama dengan dengan perhitungan secara manual sehingga dapat memberikan informasi jarak serta menunjukkan arah dalam derajat.

7.2 Saran

1. Untuk penelitian lebih lanjut, pengembang aplikasi dapat melakukan penelitian terhadap akurasi nilai *latitude longitude* yang didapatkan dari *receiver* GPS sehingga hasil dari perhitungan jarak dan perhitungan arah menjadi lebih sesuai dengan lapangan.
2. Untuk penelitian lebih lanjut, pengembang bisa menggunakan server untuk menyimpan data jalur pendakian sehingga pendaki bisa *sharing* data jalur pendakian antar pendaki.

DAFTAR PUSTAKA

[ALG-13] Alghasra, M Dalal dan Saeed, Y Hesham. *Guiding Visually Impaired People with NXT Robot through an Android Mobile Application.* International Journal of Computing and Digital Systems, Vol 2, No. 3, hal. 129-134

[COR-14] Cordova. 2013. *About Apache Cordova™.* [terhubung berkala]. http://cordova.apache.org/ [4 Juni 2014].

[ESS-11] Essayad, Abdesslam. 2011. *Design and implementation of a platform or locationbased services: a case study of GIS of archaeological and handicraft of ez Medina*. International Journal of Computer Science Issues, Vol. 8, Issue 5, No 3.

[GAI-13] Gaic, Dragan. 2013 . *TOP 7 NOTABLE LESS KNOWN MOBILE HTML5 FRAMEWORKS.* [terhubung berkala]. http://www.gajotres.net/top-7-notable-less-known-mobile-html5-frameworks/ [4 Maret 2014].

[PHE-14] Phonegap. 2014. *Storage.* [terhubung berkala]. http://docs.phonegap.com/en/3.3.0/cordova_storage_storage.md.html#Storage [4 Juni 2014]

[PHN-14] Phonegap. 2014. *Compass.* [terhubung berkala]. http://docs.phonegap.com/en/3.3.0/cordova_compass_compass.md.html#Compass [4 Juni 2014]

[PHO-14] Phonegap. 2014. *Geolocation.* [terhubung berkala]. http://docs.phonegap.com/en/3.3.0/cordova_geolocation_geolocation.md.html#Geolocation [4 Juni 2014]

[SOJ-14] Soyjavi. 2014. *LungoJS.* [terhubung berkala]. https://github.com/TapQuo/Lungo.js/ [4 juni 2014].

[SOY-14] Soyjavi. 2014. *QuoJS.* [terhubung berkala]. https://github.com/soyjavi/QuoJS/blob/master/README.md [4 juni 2014].

[TAU-2013] Taufik, Mohammad. 2013. *Lima hari hilang, dua pendaki Semeru ditemukan.* [terhubung berkala].

http://www.merdeka.com/peristiwa/lima-hari-hilang-dua-pendaki-semeru-ditemukan.html [4 Maret 2014].

[VEN-14] Vennes, Chris. 2010. *Calculate distance, bearing and more between Latitude/Longitude points*. [terhubung berkala]. http://www.movable-type.co.uk/scripts/latlong.html [4 Juni 2014].

Lampiran

1. Skenario *Use Case*

Tabel 1.1 Skenario *Use Case* Melihat Halaman Peta

Nama Use Case	Melihat Halaman Peta.
Aktor	Pengguna.
Tujuan	Melihat menu peta aplikasi.
Deskripsi	*Use case* ini memodelkan kegiatan melihat menu peta aplikasi.
Kondisi Awal	Pengguna sudah membuka aplikasi.
Kondisi Akhir	Aplikasi menampilkan menu peta.
Main Flow	1. Pengguna menekan tombol pilihan menu. 2. Pengguna memilih menu peta. 3. Sistem menampilkan menu peta.
Alternatif Flow	-

Tabel 1.2 Skenario *Use Case* Menambahkan Tanda

Nama Use Case	Menambahkan Tanda
Aktor	Pengguna.
Tujuan	Untuk menambahkan tanda gambar/teks.
Deskripsi	Menambahkan sebuah tanda pada titik yang telah diambil dari GPS.
Kondisi Awal	Tanda gambar/teks belum ditambahkan.
Kondisi Akhir	Tanda gambar/teks telah ditambahkan.
Main Flow	1. Pengguna menekan tombol pilihan diatas kanan halaman menu peta 2. Pengguna memilih tanda gambar atau menambahkan teks. 3. Tanda digambakan disamping titik kordinat GPS.

Alternatif Flow	1.1 Eksepsi jika titik belum ada Menampilkan *alert* kesalahan karena titik belum digambarkan.

Tabel 1.3 Skenario *Use Case* Melihat Jalur Pendakian Dalam *Map*

Nama Use Case	Melihat jalur pendakian dalam *Map*
Aktor	Pengguna
Tujuan	Melihat jalur pendakian dalam bentuk *map*
Deskripsi	Melihat jalur dalam bentuk *map* dan bisa digunakan jika terdapat koneksi internet.
Kondisi Awal	Jalur pendakian ditampilkan secara *offline*.
Kondisi Akhir	Jalur pendakian ditampilkan dalam bentuk *map*.
Main Flow	1. Pengguna menggambarkan sebuah jalur pendakian. 2. Pengguna menekan tombol pilihan diatas kanan halaman menu peta. 3. Pengguna menekan tombol maps.
Alternatif Flow	1.1 Eksepsi jika internet tidak tersedia. Halaman akan kosong dan tidak menampilkan jalur pendakian.

Tabel 1.4 Skenario *Use Case* Menghapus Tanda

Nama Use Case	Menghapus Tanda
Aktor	Pengguna.
Tujuan	Untuk menghapus tanda gambar/teks.
Deskripsi	Menghapus tanda yang sudah ditambahkan
Kondisi Awal	Tanda masih ada.
Kondisi Akhir	Tanda telah dihapus
Main Flow	1. Pengguna menekan tombol pilihan diatas kana halaman menu peta.

	2. Pengguna menekan tombol hapus.
Alternatif Flow	-

Tabel 1.5 Skenario *Use Case* Menunjukkan Arah

Nama Use Case	Menunjukkan Arah
Aktor	Pengguna.
Tujuan	Memberikan petunjuk arah.
Deskripsi	Menunjukkan arah berdasarkan posisi terbaru dengan titik yang diambil sebelumnya.
Kondisi Awal	Icon arah belum menunjukkan arah.
Kondisi Akhir	*Icon* arah menunjukkan arah.
Main Flow	1. Pengguna menekan tombol pilihan di kanan atas halaman menu peta. 2. Pengguna menekan tombol tunjukkan arah. 3. Pengguna memilih arah kesebuah titik. 4. *Icon* arah bergerak sesuai dengan arah posisi terbaru dengan titik kordinat GPS.
Alternatif Flow	1.1 Eksepsi jika pengambilan kordinat *latitude longitude* gagal Akan menampilkan sebuah *alert* bahwa pengambilan nilai *latitude longitude* terbaru gagal.

Tabel 1.6 Skenario *Use Case* Menampilkan Info Jarak

Nama Use Case	Menampilkan Info Jarak
Aktor	Pengguna.
Tujuan	Memberikan informasi jarak.
Deskripsi	Informasi jarak pada posisi terbaru dengan titik yang diambil dari GPS sebelumnya.
Kondisi Awal	Informasi jarak belum ditampilkan

Kondisi Akhir	Informasi jarak ditampilkan.
Main Flow	1. Pengguna menekan tombol pilihan dikanan atas halaman menu peta. 2. Pengguna menekan tombol info jarak 3. Pengguna memilih titik yang ingin diketahui jaraknya. 4. Pengguna menekan tombol simpan.
Alternatif Flow	1.1 Eksepsi pengambilan kordinat *latitude longitude* gagal Akan menampilkan sebuah *alert* bahwa pengambilan nilai latitude longitude terbaru gagal.

Tabel 1.7 Skenario *Use Case* Melihat Menu Arsip

Nama Use Case	Melihat Menu Arsip
Aktor	Pengguna.
Tujuan	Melihat daftar arsip peta.
Deskripsi	Menampilkan daftar peta yang telah tersimpan.
Kondisi Awal	Menu arsip belum ditampilkan.
Kondisi Akhir	Menampilkan menu arsip.
Main Flow	1. Menekan *icon* menu. 2. Menekan *icon* menu arsip. 3. Sistem menampilkan daftar peta yang tersimpan.
Alternatif Flow	-

Tabel 1.8 Skenario *Use Case* Membuka Data

Nama Use Case	Membuka Data
Aktor	Pengguna.
Tujuan	Membuka jalur pendakian.
Deskripsi	Membuka jalur pendakian yang telah dibuat dan disimpan

	sebelumnya.
Kondisi Awal	Peta belum terbuka.
Kondisi Akhir	Peta telah terbuka.
Main Flow	1. Pengguna sudah berada dimenu arsip. 2. Pengguna menekan tombol buka. 3. Peta telah dibuka dan berada dimenu peta
Alternatif Flow	-

Tabel 1.9 Skenario *Use Case* Menghapus Data

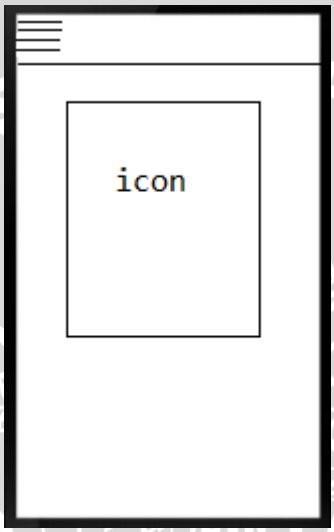
Nama Use Case	Menghapus Data
Aktor	Pengguna.
Tujuan	Menghapus data jalur pendakian.
Deskripsi	Menghapus data jalur pendakian yang tersimpan didalam database.
Kondisi Awal	Peta berada di *database*.
Kondisi Akhir	Peta telah terhapus dari *database*.
Main Flow	1. Pengguna sudah berada dimenu arsip 2. Pengguna menekan tombol hapus. 3. Peta telah terhapus dari *database*.
Alternatif Flow	-

Tabel 1.10 Skenario *Use Case* Melihat Menu Petunjuk

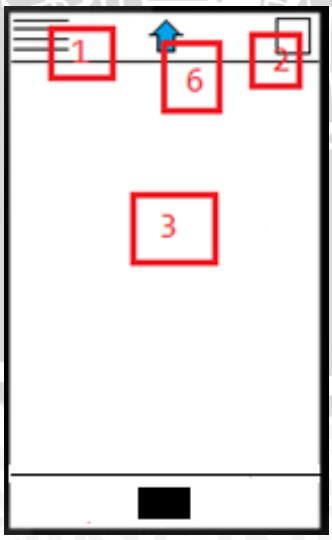
Nama Use Case	Melihat Menu Petunjuk
Aktor	Pengguna.
Tujuan	Melihat pentunjuk.
Deskripsi	Menampilkan informasi penggunaan aplikasi.
Kondisi Awal	Menu petunjuk belum ditampilkan.
Kondisi Akhir	Menampilkan menu petunjuk.
Main Flow	1. Pengguna menekan *icon* menu.

	2. Pengguna menekan *icon* menu petunjuk. 3. Menu petunjuk terbuka.
Alternatif Flow	-

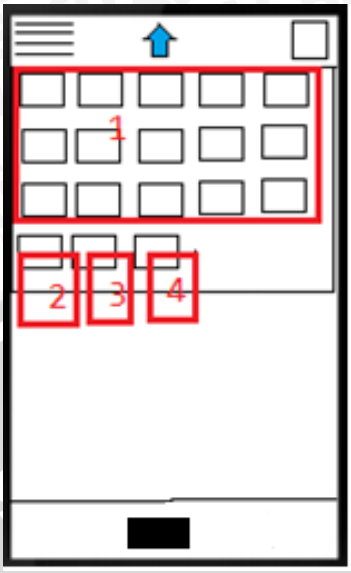
2. Perancangan Antarmuka



Gambar 2.1 Perancangan Antarmuka Menu Halaman Utama

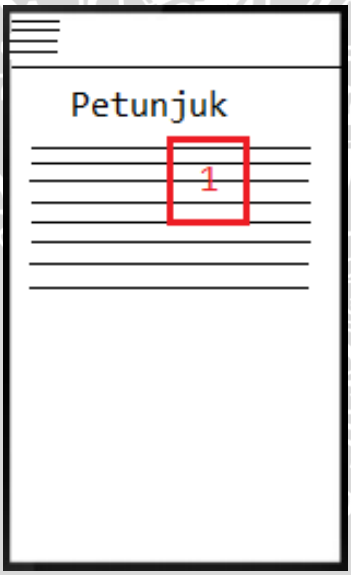


Gambar 2.2 Perancangan Antarmuka Menu Halaman Peta



Gambar 2.3 Perancangan Antarmuka Menu Pilihan Halaman Peta

\



Gambar 2.4 Perancangan Antarmuka Menu Petunjuk