SISTEM INFORMASI PRAKTIK DOKTER BERBASIS PERANGKAT BERGERAK

SKRIPSI

Untuk Memenuhi Sebagian Persyaratan Memperoleh Gelar Sarjana Komputer

Disusun Oleh :

REEZA AULYA AKBAR

NIM. 105060807111074

PROGRAM STUDI INFORMATIKA / ILMU KOMPUTER

PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER

UNIVERSITAS BRAWIJAYA

MALANG

2014

LEMBAR PERSETUJUAN

SISTEM INFORMASI PRAKTIK DOKTER BERBASIS PERANGKAT BERGERAK

SKRIPSI

Untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer

Disusun Oleh :

REEZA AULYA AKBAR

NIM. 105060807111074

Skripsi ini telah disetujui oleh dosen pembimbing pada tanggal 30 Mei 2014

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

LEMBAR PENGESAHAN

SISTEM INFORMASI PRAKTIK DOKTER BERBASIS PERANGKAT BERGERAK

SKRIPSI

Disusun Oleh :

REEZA AULYA AKBAR

NIM. 105060807111074

Skripsi ini telah diuji dan dinyatakan lulus pada tanggal 27 Juni 2014

Penguji I,	**Penguji II,**
Dr. Eng Herman Tolle, ST., MT	**Denny Sagita Rusdianto, S.Kom., M.Kom**
NIP. 19740823 200012 1 001	**NIK. 851124 06 1 1 0250**

Penguji III,

Wibisono Sukmo Wardhono, ST., MT
NIK. 820404 06 1 1 0091

Mengetahui,
Ketua Program Studi Informatika / Ilmu Komputer

Drs. Marji, MT
NIP. 19670801 199203 1 001

ORISINALITAS SKRIPSI

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

Apabila ternyata di dalam naskah SKRIPSI ini dapat dibuktikan terdapat unsur-unsur PLAGIASI, saya bersedia SKRIPSI ini digugurkan dan gelar akademik yang telah saya peroleh (SARJANA) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku yaitu UU No. 20 Tahun 2003, Pasal 25 Ayat (2) dan Pasal 70.

Undang-Undang No. 20 Tahun 2003 Pasal 25 Ayat (2) yang berisi "Lulusan perguruan tinggi yang karya ilmiahnya digunakan untuk memperoleh gelar akademik, profesi, atau vokasi terbukti merupakan jiplakan dicabut gelarnya" dan Pasal 70 yang berisi "Lulusan yang karya ilmiah yang digunakannya untuk mendapatkan gelar akademik, profesi, atau vokasi sebagaimana dimaksud dalam Pasal 25 Ayat (2) terbukti merupakan jiplakan dipidana dengan pidana penjara paling lama dua tahun dan/atau pidana denda paling banyak Rp. 200.000.000,00 (dua ratus juta rupiah)".

Malang, 30 MEI 2014

Mahasiswa,

Reeza Aulya Akbar

NIM. 105060807111074

KATA PENGANTAR

Puji dan syukur Penulis panjatkan kehadirat Allah SWT, karena hanya dengan rahmat dan bimbingannya Penulis dapat menyelesaikan tugas akhir dengan judul "Sistem Informsi Praktik Dokter Berbasis Perangkat Bergerak" dengan baik. Tanpa rahmat dan bimbingan dari Tuhan Yang Maha Esa, maka niscaya Penulis tidak akan dapat menyelesaikan tugas akhir ini dengan baik dan tepat waktu.

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

1. Orang tua Penulis, Drs Moh Marsuki . M.Si(Bapak) dan R Endang Agustina (Ibu) yang telah memberi motivasi, kasih sayang serta dukungan moril dan materil kepada Penulis. MbakYesy Widyusita, STdan Mas Chairil Iskandar, ST yang telah memberikan semangat dari awal sampai akhir pengerjaan tugas akhir ini.
2. Bapak Drs. Marji, MT. dan Bapak Issa Arwani, ST., MT. selaku Ketua dan Sekretaris Program Studi Informatika / Ilmu Komputer serta segenap Bapak / Ibu Dosen, Staff Administrasi dan Perpustakaan Program Studi Informatika / Ilmu Komputer Universitas Brawijaya.
3. Bapak Aryo Pinandito, ST., M.MT. dan Bapak Agi Putra Kharisma, ST., MT. selaku dosen pembimbing I dan dosen pembimbing II yang telah banyak memberikan ide, pemikiran, bimbingan, ilmu, dan saran dalam penyusunan tugas akhir ini.
4. Bapak Ir Sutrisno, ST, M.T selaku dosen pembimbing akademik yang telah memberikan bimbingan, ilmu, dan saran selama Penulis menuntut ilmu.
5. Semua teman-teman di PTIIK, terima kasih atas segala bantuannya selama menjadi mahasiswa.

6. Semua pihak yang tidak dapat Penulis sebutkan satu per satu yang terlibat baik secara langsung maupun tidak langsung demi terselesaikannya tugas akhir ini.

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

Akhirnya penulis berharap agar skripsi ini dapat memberikan sumbangan dan manfaat bagi semua pihak yang berkepentingan.

Malang, 30 MEI 2014

Penulis

ABSTRAK

Reeza Aulya Akbar. 2014. Sistem Informasi Praktik Dokter Berbasis Perangkat Bergerak. Program Teknologi Informasi dan Ilmu Komputer, Universitas Brawijaya, Malang. Dosen Pembimbing : Aryo Pinandito, ST., M.MT. dan Agi Putra Kharisma, ST., MT.

Permintaan pasar global untuk mengakses informasi semakin bertambah, salah satu informasi yang sangat dibutuhkan adalah informasi layanan kesehatan, seperti Malang ini yang sangat padat dan luas. Ketersedian sumber informasi yang dapat diakses dimana saja dan kapan saja menjadi kebutuhan utama para warga Malang maupun luar Malang.kita masih sering mengalami masalah, masalah tersebut adalah pada saat kita ingin mencari informasi dokter umum dan dokter spesialis yang berada di kota malang. Kurangnya informasi yang diberikan pemerintah untuk mendapatkan informasi daftar dokter, informasi jadwal dokter dan rute menuju lokasi dokter sering kali membuat masyarakat kesulitan dalam menemukan informasi tersebut yang membuat masyarakat memilih bertanya langsung ke tempat praktik dokter atau bertanya kepada saudara, kerabat, ataupun masyarakat lainnya untuk mendapatkan informasi tersebut.. Untuk mendukung kebutuhan akses informasi tersebut, dibutuhkan pengembangan suatu aplikasi yang kaya dan inovatif yang dapat membantu masyarakat sehingga masyarakat tidak kesulitan lagi dalam mendapatkan informasi tersebut .

Hasil pengujian yang dilakukan menunjukkan bahwa pengujian validasi sistem telah memenuhi kebutuhan fungsional yang di spesifikasikan , dan pengujian UAT (*User Acceptance Test*) dengan menggunakan kuisioner didapatkan bahwa aplikasi praktik dokter dapat membantu pengguna dalam mendapatkan daftar dokter, jadwal dokter dan rute menuju lokasi praktek dokter.

Kata Kunci :informasi dokter, lokasi dokter , perangkat bergerak

ABSTRACT

Reeza Aulya Akbar. 2014. *Information Systems Practice Physician Based Mobile Devices.* *Information Technology and Computer Science Program,* Brawijaya *University,* Malang. *Advisors* : Aryo Pinandito, ST., M.MT. dan Agi Putra Kharisma, ST., MT.

Global market demand for access to information is increasing, one of the information that is needed is a health care information, such as Malang is very dense and wide. Availability of resources that can be accessed anywhere and at any time become the primary needs of residents and outside Malang Malang. we still experience the problem, the problem is when we want to find information general practitioners and specialist physicians who are in the unfortunate city. The lack of information provided by the government to obtain a list of physician information, schedule information and directions to the location of physician clinicians often make public the difficulty in finding the information that makes people choose to ask directly to the doctor's office or ask the brothers, relatives, or other community to get The information . To support the information access needs, required the development of a rich and innovative applications that can help the community so that people no longer difficulty in obtaining such information.

Test results have shown that the system has met the validation testing functional requirements in specified, and UAT testing (User Acceptance Test) using the questionnaire showed that the application practice physician can help the user in getting the list of physicians, physician schedules and directions to the doctor's office locations.

Keywords: doctor information, , the device mobile, the location of the doctor

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

DAFTAR KODE

DAFTAR LAMPIRAN

BAB I
PENDAHULUAN

1.1 Latar Belakang

Permintaan pasar global untuk mengakses informasi semakin bertambah, termasuk masalah informasi kesehatan, masalah kesehatan menjadi sisi yang sangat penting untuk dibahas kepermukaan. Khususnya dalam informasi yang menghubungkan dua pelaku yaitu pemerintah dengan masyarakat sebagai pelaku utama sosial dalam kehidupan berbangsa dan bernegara. Secara konstitusional Negara telah menjelaskan secara tegas melalui Undang-Undang Nomor 36 tahun 2009 bahwa pemerintah bertanggung jawab atas ketersediaan akses terhadap informasi, edukasi, dan fasilitas pelayanan kesehatan untuk meningkatkan dan memelihara derajat kesehatan yang setinggi-tingginya [PRE-09].

Pada kenyataanya informasi yang didapat oleh masyarakat berbanding terbalik dengan kepastian konstitusi yang didengungkan oleh pemerintah selama ini, yang berarti masyarakat mengalami krisis informasi mengenai kesehatan. Informasi yang dominan dibutuhkan oleh masyarakat khususnya mengenai lingkungan yang berkenaan dengan dokter spesialis maupun dokter umum. Sejauh ini informasi yang diberikan belum mendukung mengenai kapan dan dimana dokter berada, dan hanya memberikan informasi hanya pada satu tempat praktik. Ini membuat masyarakat kesulitan untuk mendapatkan informasi praktik dokter sesuai dengan kebutuhannya.

Berkembangnya teknologi informasi khususnya alat komunikasi seperti *smartphone* yang dapat diakses dimana saja dan kapan saja, seharusnya menjadi proses penawaran untuk menghubungkan masalah komunikasi masyarakat mengenai informasi yang berkaitan dengan pelayanan kesehatan. Saat ini *smartphone* telah banyak digunakan untuk membantu mengatasi masalah sehari-hari. Dalam bidang pelayanan kesehatan *smartphone* dapat digunakan untuk memudahkan dalam mendapatkan informasi layanan kesehatan yang dibutuhkan. Pada *smartphone* terdapat berbagai jenis platform seperti Android, iOS, Blackberry, Windows dan yang lainnya . Menurut hasil survei dari Gartner, sebuah perusahaan riset dan konsultan

Teknologi Informasi (TI) ternama, terlihat adanya peningkatan pangsa pasar Android lebih dari 700 % dalam tahun 2010 yang memiliki nominal smartphone terjual sebanyak 67.224.500. Hal ini menunjukkan besarnya potensi Android di masa depan [NUR-13]. Dari hasil survey tersebut sebenarnya keadaan ini berhubungan dengan kemungkinan bahwa *smartphone* khususnya Android dapat dimanfaatkan sebagai alat untuk mendapatkan akses informasi yang berkaitan dengan praktik dokter di tiap-tiap daerah.

Berdasarkan uraian tersebut maka dibutuhkan pengembangan suatu sistem informasi pada perangkat bergerak berbasis Android menggunakan bahasa pemrograman Java dengan pendekatan MVC untuk aplikasi perangkat bergerak dan menggunakan bahasa pemprograman PHP untuk *web service* sebagai penghubung antara aplikasi perangkat bergerak dan server. *Web service* inilah yang akan menyediakan dan mengelolah data untuk ditampilkan di aplikasi perangkat berberak. Dengan adanya suatu pengembangan suatu sistem informasi diharapkan dapat membantu masyarakat sehingga masyarakat tidak kesulitan lagi dalam mendapatkan informasi tersebut.

1.2 Perumusan Masalah

Berdasarkan latar belakang yang telah dijelaskan maka permasalahan dapat dirumuskan bagaimana menganalisis, merancang dan mengimplementasi sistem yang dapat membantu pengguna dalam mendapatkan informasi praktik dokter umum dan dokter spesialis pada perangkat bergerak berbasis android.

1.3 Tujuan

Tujuan dari penelitian ini adalah untuk merancang dan membangun sistem informasi praktik dokter pada perangkat bergerak untuk membatu pengguna dalam mendapatkan informasi praktik dokter umum dan dokter spesialis.

1.4 Batasan Masalah

Berdasarkan latar belakang dan rumusan yang telah diuraikan maka penelitian ini mempunyai batasan masalah sebagai berikut :

1. Lingkungan sistem informasi praktik dokter dibatasi di kota Malang
2. Studi kasus implementasi sistem informasi praktik dokter pada perangkat bergerak dengan sistem operasi Android.
3. Pengumpulan *requirement* sistem dilakukan dengan metode wawancara kepada masyarakat di kota Malang.
4. Analisis kebutuhan sistem dilakukan dengan metode *object-oriented analysis* dengan menggunakan bahasa pemodelan UML (*Unified Modeling Language*)

1.5 Manfaat Penelitian

1.5.1 Pemerintah

Mendukung salah satu tugas Pemerintah yaitu program mencerdaskan bangsa dan bertanggung jawab atas ketersediaan akses terhadap informasi kesehatan sebagai amanat dalam Undang-undang

1.5.2 Masyarakat

Mempermudah masyarakat untuk mengetahui informasi praktik dokter kota Malang dengan menggunakan sistem informasi praktik dokter.

1.6 Sistematika Pembahasan

Untuk mencapai tujuan yang diharapkan, maka sistematika penulisan yang disusun dalam laporan tugas akhir ini adalah sebagai berikut :

BAB I Pendahuluan

Berisi tentang latar belakang masalah, rumusan masalah, batasan masalah, tujuan dan manfaat dari penelitian serta sistematika penulisan.

BAB II Dasar Teori

Menguraikan tentang dasar teori dan referensi secara luas mengenai *software* maupun *hardware* serta informasi yang diperlukan dalam

pengembangan, perancangan, implementasi, dan pengujian perangkat lunak dari permasalahan yang dibahas.

BAB III Metode Penelitian

Membahas tentang metode yang digunakan dalam penelitian dan langkah-langkah yang akan dilakukan dalam penulisan yang terdiri dari studi literatur, perancangan sistem perangkat lunak, implementasi sistem perangkat lunak, pengujian dan analisis, serta pengambilan kesimpulan.

BAB IV Analisis dan Perancangan

Membahas analisis kebutuhan dan perancangan sesuai dengan teori yang ada.

BAB V Implementasi dan Pengujian

Membahas tentang implementasi dari sistem berdasarkan metodologi dan perancangan serta memuat proses dan hasil pengujian terhadap sistem yang telah diimplementasi.

BAB VI Penutup

Memuat kesimpulan yang diperoleh dari pembuatan dan pengujian perangkat lunak yang dikembangkan serta saran-saran untuk pengembangan lebih lanjut.

BAB II
DASAR TEORI

Bab ini berisi dasar teori yang diperlukan untuk penelitian. Dasar teori adalah membahas teori yang diperlukan untuk menyusun penelitian yang diusulkan.

Teori dasar yang akan dibahas pada bab ini yaitu konsep dasar *Web service*, Format data, JSON, *Global Positioning Sistem, Google Maps API* yang trdiri dari *MapView, MapActivity, OverlayItem,* dan *GeoPoint. Android Location API* yang terdiri dari *Geocoder, Location, LocationManager, LocationListener* , dan Skala Likert

2.1 *Web Service*

Web service adalah sebuah software yang dirancang untuk mendukung interoperabilitas interaksi mesin-ke-mesin melalui sebuah jaringan. *Web service* secara teknis memiliki mekanisme interaksi antar sistem sebagai penunjang interoperabilitas, baik berupa agregasi (pengumpulan) maupun sindikasi (penyatuan). *Web service* memiliki layanan terbuka untuk kepentingan integrasi data dan kolaborasi informasi yang bisa diakses melalui internet oleh berbagai pihak menggunakan teknologi yang dimiliki oleh masing-masing pengguna [SUT-12].

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

Alasan menggunakan *web service* adalah kemudahan dalam penggunaan kembali (*reuse*) dan berbagi (*share*) logika yang sama dengan klien yang beragam

seperti *mobile*, desktop, dan aplikasi web. Jangkauan *web service* yang luas karena *web service* bergantung pada standar yang terbuka, dapat beroperasi pada plaftorm yang berbeda, serta tidak bergantung pada teknologi eksekusi yang mendasarinya. Semua *web service* setidaknya menggunakan HTTP dan format pertukaran data standar berupa XML, JSON, atau media lain. Selain itu, *web service* menggunakan HTTP dalam dua cara yang berbeda yaitu sebagai protokol standar untuk menentukan perilaku standar pelayanan serta sebagai media transportasi untuk menyampaikan data [DAI-12].

Ide untuk memanfaatkan teknologi web mulai terealisasi secara nyata ketika pada tahun 1990-an akhir dan salah satu pendekatan implementasi web service yang paling banyak digunakan disebut dengan *Simple Access Object Protocol* (SOAP) yang memanfaatkan *Extensible Markup Language* (XML), yang belum lama muncul saat itu sebagai paket format untuk mekanisme *Remote Procedure Call* (RPC). Beberapa tahun setelah kepopuleran *web service* berbasis SOAP muncul model implementasi baru yang dinamakan *REpresentational State Transfer* (REST) dan karena prinsip tersebut menggunakan model REST maka *web service* yang menerapkannya disebut *RESTful Web Services*. Dari uraian tersebut menyatakan bahwa *web service* dapat dibagi menjadi dua jenis, yaitu SOAP dan REST.

2.2.1 SOAP *Web Service*

SOAP *(Simple Object Access Protocol)* merupakan protokol komunikasi berbasis XML yang memperbolehkan aplikasi saling bertukar informasi melalui HTTP. SOAP merupakan format untuk mengirimkan *message* melalui Internet, bersifat platform independent, language independent, dan merupakan standar W3C. Sebuah pesan SOAP adalah sebuah dokumen XML yang berisi elemen-elemen berikut [JGN-11] :

1. *Envelope element* yang mengidentifikasi dokumen XML sebagai sebuah pesan SOAP.
2. Elemen *header* yang berisi informasi *header*. (bersifat opsional)
3. Elemen *body* yang berisi panggilan dan merespon informasi.

4. *Fault element* yang berisi pesan kesalahan yang terjadi pada waktu proses. (bersifat opsional)

Dibawah ini adalah contoh kerangka pesan SOAP.

```
<?xml version="1.0"?>
<soap:Envelope
xmlns:soap="http://www.w3.org/2001/12/soap-envelope"
soap:encodingStyle="http://www.w3.org/2001/12/soap-encoding">
<soap:Header>
...
</soap:Header>
            <soap:Body">
            ...
                        <soap:Fault>
                        ...
                        </soap:Fault>
            </soap:Body>
</soap:Envelope>
```

Gambar 2. 1 Kerangka pesan SOAP

Sumber : [JGN-11]

2.2.2 REST *Web Service*

REST *(Representational State Transfer)* merupakan sebuah model arsitektur yang menghubungkan jalur komunikasi antar mesin/aplikasi melalui protokol HTTP yang berbasis *synchronous request/response*. Proses transfer data dimulai ketika *client* mengirimkan sebuah *request message* yang mencakup HTTP method yang akan dipanggil, lokasi *resource* dalam format URI, serta pilihan format pesan. Kemudian *server* memproses permintaan dari *client* dan meresponse balik dengan mengirimkan data yang sesuai dengan permintaan *client* dalam format apapun. Namun sekarang ini format yang banyak dikenali oleh *client* adalah JSON.

Berikut ini adalah beberapa ciri dari REST [SSA-13]*:*

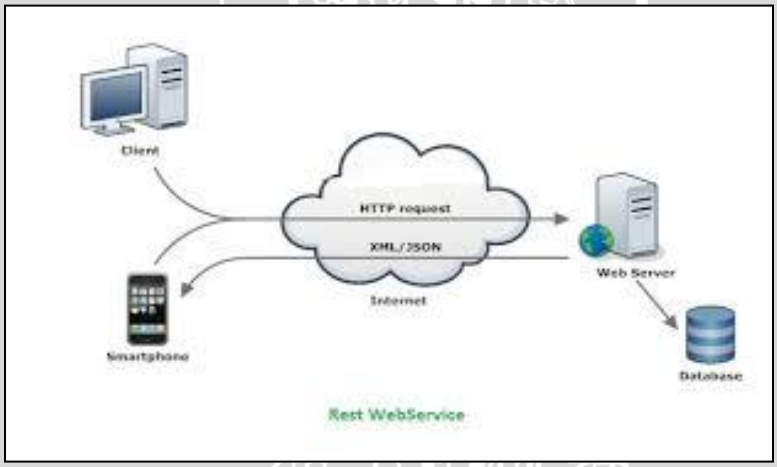
- Penggunaan HTTP *method* yang tepat untuk memanipulasi resource.

- *Request*-URI tidak mengandung *query string* dan hanya terdiri dari element resource yang dipisahkan dengan karakter " / ". Berikut ini contoh pengaksesan resource melalui URI :

  ```
  https://example.com/customers?lastName=Ghifar&zipcode=40293
  ```

Jadi pada *REST* tidak ada peraturan *envelope element, header, body, etc* seperti yang ada di SOAP, sehingga REST dapat dikatakan lebih sederhana dan lebih mudah untuk dikembangkan. Yang harus dipahami dalam konsep REST yaitu pada dasarnya setiap URL itu adalah unik yang merupakan representasi dari objek. Memperoleh konten-konten objek tersebut dengan menggunakan HTTP *method* yang sesuai, yaitu dengan menggunakan fungsi GET, POST, PUT atau DELETE.

Dibawah ini adalah REST *Web Service* arsiktektur



Gambar 2. 2 REST *Web Service* arsiktektur

Sumber : [RHL-14]

2.2.3 Format Data

Pada *web service*, tidak ada standarisasi yang mengatur format data yang akan digunakan [CHO-07]. Format data adalah jenis aturan penulisan dari sebuah pesan yang tergantung dari dukungan server terhadap tipe format data dalam pengiriman pesan kepada client. Berikut penjelasan beberapa format data yang dipakai dalam penelitian ini.

2.2.3.2 JSON

JavaScript Object Notation (JSON) adalah standar berbasis teks untuk pertukaran data. Format JSON dikenal ringan (berukuran kecil), mudah dibaca, ditulis, dan dipahami manusia serta mudah untuk diuraikan dan dibuat oleh mesin. Format ini dibuat berdasarkan bahasa pemprograman *JavaScript*, standar ECMA-262 edisi ketiga - Desember 1999. JSON bersifat *independent language* namun menggunakan kaidah penulisan yang dikenal luas oleh programmer dari keluarga bahasa C (C, C++, *C#*, *Java*, *JavaScript*, *Perl*, *Phyton*, dan lain-lain), hal tersebut menjadikan JSON sangat ideal sebagai bahasa dalam pengiriman data [WIY-12].

JSON terbuat dari dua struktur [JSO-13] :

1. Kumpulan pasangan nama atau nilai. Pada beberapa bahasa, hal ini dinyatakan sebagai objek (*object*), rekaman (*record*), struktur (*struct*), kamus (*dictionary*), tabel *hash* (*hashtable*), daftar berkunci (*keyedlist*), atau *associativearray*.
2. Daftar nilai terurutkan (*an ordered list of values*). Pada kebanyakan bahasa, hal ini dinyatakan sebagai larik (*array*), vektor (*vector*), daftar (*list*), atau urutan (*sequence*).

Penggunaan JSON secara umum terdiri dari fungsi *encode* dan *decode*. Fungsi *encode* untuk mengubah data menjadi JSON *array* dan fungsi *decode* untuk mengubah JSON *array* menjadi suatu nilai kembalian. Contoh penggunaan sintaks fungsi *encode* pada metode JSON dapat dilihat pada kode 2.3

Gambar 2. 3 Sintaks JSON *encode*

```
<?php
$arr = array('a' => 1, 'b' => 2, 'c' => 3, 'd' => 4, 'e' =>
5);
echo json_encode($arr);
-
```

Sumber : [PHP-14]

Method json_encode ($arr) akan mengubah nilai dari variabel $arr menjadi suatu format yang dapat dibaca sebagai JSON *array*. Hasil keluaran dari sintaks tersebut dapat dilihat pada gambar 2.4

Gambar 2. 4 Hasil JSON *encode*

```
{"a" : 1, "b" : 2, "c" : 3, "d" : 4, "e" : 5}
```

Sumber : [PHP-14]

JSON *decode* adalah proses memperoleh nilai dari suatu variabel dengan format JSON *arary*. Sintaks JSON pada proses *decode* dapat dilihat pada gambar 2.5

Gambar 2. 5 Sintaks JSON pada proses decode

```
<?php
$json = '{"foo-bar" :  12345}';
$obj = json_decode($json);
print $obj->{'foo-bar'};
?>
```

Sumber : [PHP-14]

Pada sintaks di atas, *method*json_decode($json) akan mengekstrak nilai dari variabel JSON *array* $json menjadi suatu nilai *output*. Hasil keluaran dari proses *decode* tersebut dapat dilihat pada Gambar 2.6

Gambar 2. 6 Hasil JSON *decode*

```
1 2 3 4 5
```

Sumber : [PHP-14]

Penggunaan JSON secara umum terdiri dari fungsi *encode* dan *decode*. Fungsi *encode* untuk mengubah data menjadi JSON *array* dan fungsi *decode* untuk mengubah JSON *array* menjadi suatu nilai kembalian. Dalam penelitian ini, data yang didapat dari server yang dilakukan oleh *web service* akan dikirimkan menggunakan format JSON kepada perangkat bergerak, sehingga dapat memudahkan proses yang terjadi pada perangkat bergerak untuk memproses dan menampilkan data tersebut sesuai dengan permintaan pengguna.

2.2 *Global Positioning Sistem (GPS)*

Global Positioning Sistem (GPS) adalah sistem untuk menentukan letak di permukaan bumi dengan bantuan penyelarasan (*synchronization*) sinyal satelit. Sistem ini menggunakan 24 satelit yang mengirimkan sinyal gelombang mikro ke Bumi. Sinyal ini diterima oleh alat penerima di permukaan, dan digunakan untuk menentukan letak, kecepatan, arah, dan waktu. Sistem navigasi berbasis satelit ditempatkan ke orbit oleh Departemen Pertahanan Amerika Serikat.

GPS pada awalnya ditujukan untuk aplikasi militer, namun pada 1980-an, pemerintah membuat sistem yang tersedia untuk penggunaan sipil. GPS bekerja di semua kondisi cuaca, di mana saja di dunia, 24 jam sehari. Tidak ada biaya langganan atau biaya setup untuk menggunakan GPS. GPS juga dapat diakses dengan bebas oleh siapa saja asalkan mempunyai sebuah alat penerima GPS. Data lokasi yang dimaksudkan di sini adalah berupa posisi geografis, yaitu latitude (lintang), longitude (bujur), dan altitude(ketinggian). GPS dapat menghitung koordinat lintang dan bujur suatu titik dengan mengunci minimal 3 sinyal satelit yang berbeda.[RRD-11:6]

Informasi GPS ditransmisikan oleh beberapa satelit (tiga satelit misalnya) sehingga GPS receiver mampu mengkalkulasi dan menampilkan seakurat mungkin posisi, kecepatan dan informasi waktu kepada pengguna GPS.

2.3 *Google Maps API*

Google Maps merupakan sebuah layanan peta dunia virtual berbasis web yang disediakan oleh Google Inc. Google Maps dapat dilekatkan sebagai elemen dalam tata letak antarmuka pengguna yang dirancang oleh pemrogram. Untuk menghubungkan aplikasi perangkat lunak dengan Google Maps diperlukan sebuah kunci yang diistilahkan sebagai API Key.

Google Maps adalah suatu peta dunia yang dapat kita gunakan untuk melihat suatu daerah. Dengan kata lain, Google Maps merupakan suatu peta yang dapat dilihat dengan menggunakan suatu browser. Kita dapat menambahkan fitur Google Maps dalam web yang telah kita buat atau pada blog kita yang berbayar maupun

gratis sekalipun dengan Google Maps API. Google Maps API adalah suatu library yang berbentuk JavaScript.Pustaka eksternal yang ditawarkan oleh Google untuk menghubungkan aplikasi Android dengan Google Maps adalah com.google.android.maps. Untuk memanfaatkan pustaka ini, diperlukan Google API's add-on sehingga dapat dibuat aplikasi berbasis Google Maps pada Android SDK dengan aksesterhadap data Google Maps. Pustaka Google Maps API terdiri atas beberapa kelas. Berikut merupakan kelas-kelas tersebut [AFD-13:05].

2.3.1 *MapActivity*

MapActivity merupakan kelas abstrak yang digunakan untuk menampilkan sebuah *MapView*. Dalam pemrograman android, untuk bisa membuat aplikasi yang mengakses layanan Google Maps, harus dilakukan pembuatan kelas yang merupakan kelas anak dari *MapActivity*. Di dalam kelas itu nantinya baru dilakukan pengkodean untuk menampilkan *MapView* dan melakukan pengaturan-pengaturan yang berkaitan. Berikut merupakan contoh kode program*MapActivity*:

kode 2. 1 Kodeprogram*MapActivity*

```
public kelas TampilkanMap extends MapActivity
        implements OnCheckedChangeListener {
                MapView mp=null;
                MapController mc;
                RadioGroup rg;

                /** Called when thr activity is first created. */
                @Override
                public void onCreate (Bundle savedIstanceState) {
                super.onCreate (Bundle savedIstanceState);
                setContentView(R.layout.main);

                mp=(MapView)findViewById(R.Id.mapView);
                mp.setBuiltInZoomControls(true);
                rg=(RadioGroup)findViewById(R.Id.viewRG);
                rg.setOnCheckedChangeListener(this);
```

```
            }

            @Override
            protected boolean isRouteDisplayed(){
                  return false;
            }
            @Override
            public void onCheckedChange(RadioGroup group, int
checkId){
                  //TODO Auto-generate method stub
                  switch(checkId){
                  case R.Id.sateliteRB:
                              mp.setStreetView(false);
                              mp.setSatellite(true);
                              break;
                  case R.Id.streetRB:
                              mp.setSatellite(false);
                              mp.setStreetView(true);
                              break;
                              }
                        }
                  }
```

Sumber : [AHA-12]

2.3.2 *MapView*

MapView merupakan kelas berupa tampilan yang digunakan untuk menampilkan peta dunia digital, yang mana data-datanya didapatkan dari layanan Google Maps. Google Map API merupakan aplikasi interface yang dapat diakses dengan javascript agar Google Map dapat ditampilkan pada halaman web yang dibangung. Untuk dapat mengakses Google Map, harus melakukan pendaftaran Api Key terlebih dahulu dengan data pendaftaran berupa nama domain web yang dibangun. Tampilan ini dapat menangkap *touch gesture* sehingga dapat dilakukan penggeseran dan perbesaran pada peta tersebut. Tampilan peta dapatdiubah-ubah ke dalam tiga mode tampilan, yaitu tampilan *Street View*, *Satellite View*, dan *Traffic View*.

kode 2. 2 Contoh kode program *MapView*

```
private void setupMapIfNeeded()
{
if (map == null)        {
FragmentManager fragmentManager =
getSupportFragmentManager();
 SupportMapFragment supportMapFragment = (SupportMapFragment)
        fragmentManager.findFragmentById(R.id.maps);
map = supportMapFragment.getMap();
if (map != null)
{
setupMap();
  }
 }
}
```

Gambar 2. 7 Tampilan Google Maps

Sumber : [Google Maps]

2.3.3 *OverlayItem*

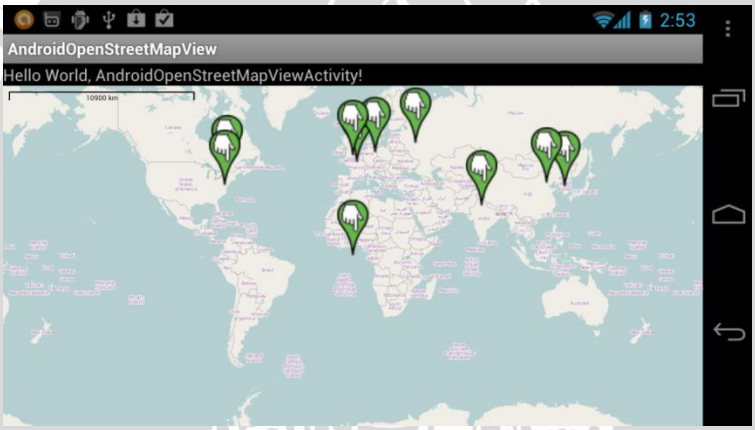
Kelas ini merupakan kelas abstrak yang digunakan sebagai kelas dasar bagi sebuah Overlay yang terdiri atas daftar-daftar OverlayItems. Sesuai dengan namanya, kelas ini memungkinkan untuk mendaftarkan point of interest yang akan ditampilkan di peta [MML-09:05]. Kelas ini menangani beberapa hal seperti proses penggambaran penanda (marker) untuk setiap titik dan menangani pula proses untuk menindaklanjuti adanya tap yang dilakukan pengguna terhadap item.

kode 2. 3 Contoh kode program *OverlayItem*

```
public kelas CustomItemizedOverlay extends
ItemizedOverlay<OverlayItem> {
      private ArrayList<OverlayItem> mapOverlays = new
ArrayList<OverlayItem>();
      private Context context;
      public CustomItemizedOverlay(Drawable defaultMarker) {
                  super(boundCenterBottom(defaultMarker));
                  //TODO Auto-generated coonstructor stub
            }
      public CustomItemizedOverlay(Drawable defaultMarker, Context
context){
            this(defaultMarker);
            this.context = context;
            }
            @Override
      protected OverlayItem createItem(int i){
                  //TODO Auto-generated method stub
                  return mapOverlays.get(i);
                  }
                  @Override
            public int size(){
                  //TODO Auto-generated method stub
                  }
            @Override
      protected boolean onTap(int index){
      OverlayItem item = mapOverlays.get(index);
```

```
AlertDialog.Builder ad = new AlertDialog.Builder(context);
ad.setMessage(Item.getSnippet());
ad.show();
return true;
}
public void addOverlay(OverlayItem overlay){
      mapOverlays.add(overlay);
      this.populate();
      }
}
```

Sumber : [AHA-12]



Gambar 2. 8 *ViewMarker*

Sumber : [JKM-13]

2.4.4 GeoPoint

Kelas *GeoPoint* merupakan kelas yang merepresentasikan sepasang titik lokasi geografis yaitu lintang dan bujur. Titik tersebut disimpan sebagaiangka integer dalam satuan mikroderajat. Konstruktor kelas ini adalah GeoPoint(int latitudeE6, int longitudeE6).GeoPoint juga perangkat lunak yang ideal untuk pemetaan geografis, mampu menyimpan lintang, bujur, ketinggian dan presisi, memperbarui tanda dalam real time pada peta agar mudah dilihat.

kode 2. 4 Contoh kode program *GeoPoint*

```
        private void getGeoPointUser(double lat, double lon){
        //TODO Auto-generated method stub
        GeoPoint point = new GeoPoint((int)(lat * 1E6),(int)(lon *
1E6));
        OverlayItem overlayitem = new OverlayItem(point,"haii..",
"saya omayib")'
        itemizedOverlay = new
        CustomItemizedOverlay(this.getResources()
                .getDrawable(R.drawable.marker),
        MapMarker.this);
                itemizedOverlay.addOverlay(overlayitem);
                mapOverlays.add(itemizedOverlay);
                mapController.animateTo(point);
                mapController.seZoom(6);
            }
```

Sumber [AHA-12]

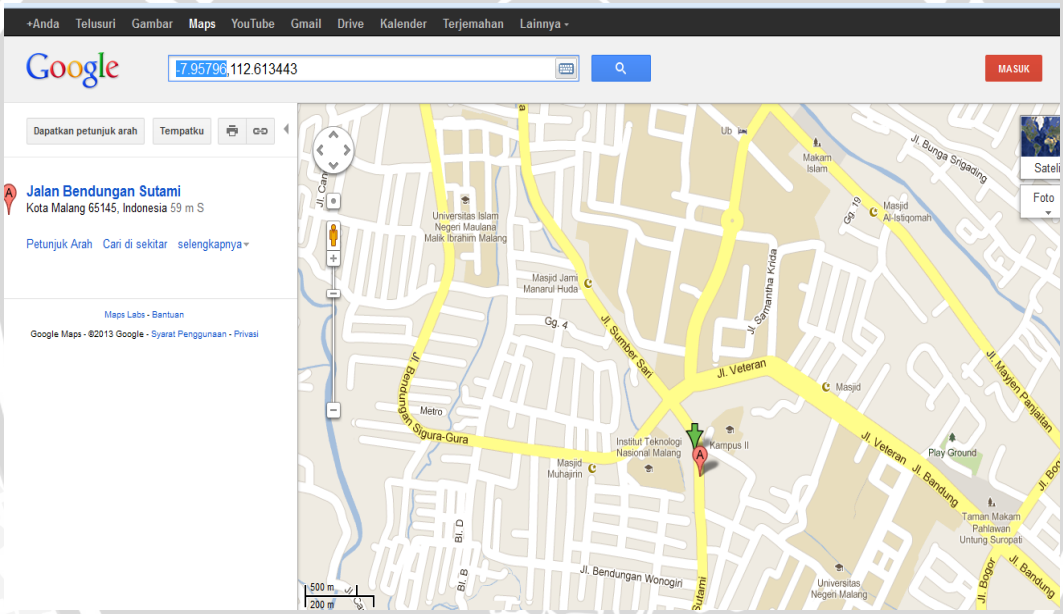
Gambar 2. 9 Tampilan *Google Maps*pada *smartphone android*

Sumber : [JKM-13:03]

2.4 Android Location API

2.4.1 Geocoder

Kelas Geocoder merupakan kelas yang dipergunakan untuk menangani geocoding dan reverse geocoding [JKM-13:03]. Geocoding sendiri merupakan proses konversi dari data geografis (misalnya alamat atau kode pos) menjadi koordinat geografis dalam bentuk lintang dan bujur (misalnya -7.057006, 110.432798) dan hasil angka tersebut digunakan untuk penentuan titik lokasi-lokasi yang dibutuhkan oleh pengguna untuk mencari suatu tempat tertentu. Sementara Reverse Geocoding sendiri merupakan kebalikan dari Geocoding.



Gambar 2. 10 Contoh marker suatu lokasi

Sumber : [Google Maps]

2.4.2 Location

Kelas *Location* merupakan kelas yang merepresentasikan suatu lokasi geografis tertentu pada map. Dalam kelas ini terdapat method-method yang berguna yang berhubungan dengan suatu lokasi. Untuk mencari jarak tertentu dapat digunakan

method *distanceTo (Location dest)* dengan memberikan parameter lokasi tujuan tertentu yang diinginkan.

2.4.3 Location Manager

Kelas ini memberikan akses pada layanan lokasi sistem. Hal ini memungkinkan aplikasi untuk mendapatkan pembaharuan dari lokasi geografis perangkat Android. Pada kelas ini, terdapat atribut-atribut yang berguna dalam proses pembaharuan lokasi geografis perangkat. Dua di antaranya adalah GPS_PROVIDER dan NETWORK_PROVIDER. Kedua atribut tersebut bertipe String.

2.4.4 Location Listener

LocationListener merupakan antarmuka yang digunakan untuk menerima pemberitahuan dari *LocationManager* ketika lokasi perangkat Android berubah [WNE-12:02].*Location Listener* dipanggil dari method *request Location Updates*(*String provider, long minTime, float minDistance, LocationListener listener*) pada *LocationManager*.

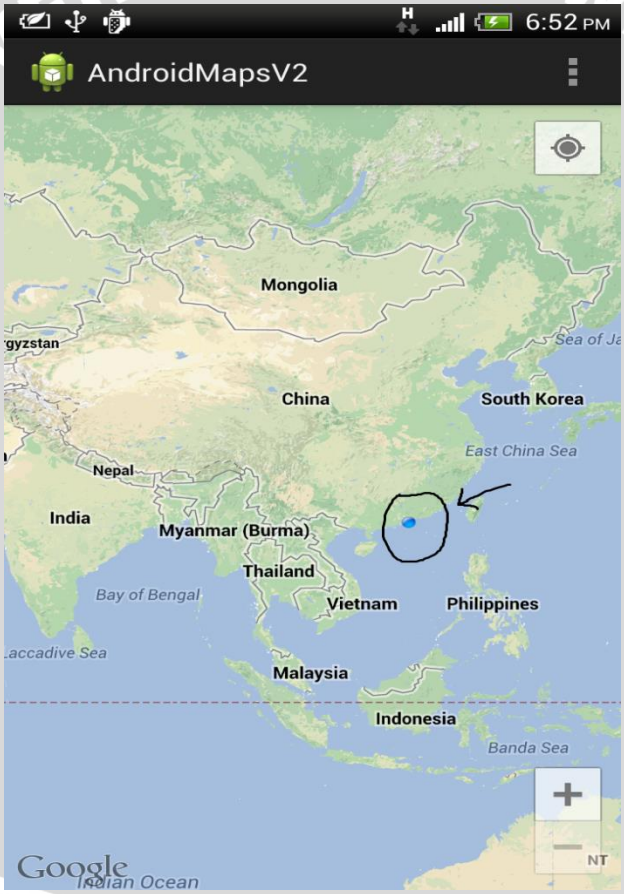
a. Berikut merupakan contoh kode program dari *location*, *locationmanager*, dan *locationlistener* :

kode 2. 5 Contoh kode program *Location*

```
public void onCreate(Bundle savedInstanceState){
    super.onCreate(savedInstanceState);
    setContentView(R.layout.main);

    mapView = (MapView) findViewById(R.id.map_view);
    mapView.setBuiltInZoomControls(true);
    mapOverlays = mapView.getController();
    pos = new userPosition();

    LocationManager mLOcationManager = (LocationManager)
    getSystemService(context.LOCATION_SERVICE);
    locationListener = new MyLOcationListener();
    mLocationManager.requestLocationUpdate
    (LocationManager.GPS_PROVIDER,0,0, locationListener);
```

```
if (pos.getLatitude() > 0){
getGeoPointUser(pos.getLatitude(),
pos.getLongitude());

}else{
getGeoPointUser(-7.801037, 110.364756);
}
                                    }
```

Sumber : [AHA-12]



Gambar 2. 11 *Location*

Sumber : [*Google Maps Location*]

2.5 Skala Likert

Skala *Likert* digunakan untuk mengukur sikap, pendapat, dan persepsi seseorang atau sekelompok orang tentang fenomena sosial. Dalam penelitian, fenomena sosial ini telah ditetapkan secara spesifik oleh peneliti, yang selanjutnya disebut sebagai variabel penelitian.

Dengan skala *Likert*, maka variabel yang akan diukur dijabarkan menjadi indikator variabel. Kemudian indikator tersebut dijadikan sebagai titik tolak untuk menyusun item-item instrumen yang dapat berupa pernyataan atau pertanyaan.

Jawaban setiap item instrumen yang menggunakan skala *likert* mempunyai gradasi dari sangat positif sampai sangat negatif, yang dapat berupa kata-kata antara lain:

a. Sangat setuju	a. Selalu
b. Setuju	b. Sering
c. Ragu-ragu	c. Kadang-kadang
d. Tidak setuju	d. Tidak pernah
e. Sangat tidak setuju	

a. Sangat positif	a. Sangat baik
b. Positif	b. Baik
c. Negatif	c. Tidak baik
d. Sangat negatif	d. Sangat tidak baik

Untuk keperluan analisis kuantitatif, maka jawaban itu dapat deberi skor, misalnya :

1. Setuju/selalu/sangat positif diberi skor	5
2. Setuju/sering/positif diberi skor	4
3. Ragu-ragu/kadang-kadang/netral diberi skor	3
4. Tidak setuju/hamper tidak pernah/negatif diberi skor	2

5. Sangat tidak setuju/tidak pernah/ diberi skor 1 [SUG-10]

Contoh penggunaan skala likert :

Terdapat 70 panelis atau audien untuk pengukuran aspek tertentu dalam sebuah penelitian. Berikut rangkuman hasil penilaian panelis atau audien ditunjukkan pada tabel 2.1

Tabel 2. 1 Rangkuman hasil penilaian panelis atau audien

No	Hasil penilaian panelis atau audien
1	Panelis/audien yang menjawab sangat tidak suka (1) berjumlah 20 orang
2	Panelis/audien yang menjawab tidak suka (2) berjumlah 25 orang
3	Panelis/audien yang menjawab netral (3) berjumlah 15 orang
4	Panelis/audien yang menjawab suka (4) berjumlah 8 orang
5	Panelis/audien yang menjawab sangat suka (5) berjumlah 2 orang

Kemudian menghitung total nilai yang dihasilkan dari keseluruhan jawaban panelis pada setiap skor Likert dengan menggunakan rumus Total nilai per skor Likert (1).

Total nilai per skor Likert : **T x Pn** ……1)

T = Total jumlah panelis/audien yang memilih

Pn = Pilihan angka Skor Likert.

Hasil Perhitungan dengan menggunakan Rumus (1) ditunjukkan pada Tabel 2.2:

Tabel 2. 2 Hasil Perhitungan Total nilai per skor Likert

No	Keterangan	Perhitungan
1	Panelis/audien yang menjawab sangat tidak suka (1)	20 x 1 = 20

2	Panelis/audien yang menjawab tidak suka (2)	25 x 2 = 50
3	Panelis/audien yang menjawab netral (3)	15 x 3 = 45
4	Panelis/audien yang menjawab suka (4)	8 x 4 = 32
5	Panelis/audien yang menjawab sangat suka (5)	2 x 5 = 10

Dari Tabel 2.3 dilakukan perhitungan total skor dengan menggunakan Rumus Total skor (2).

Total skor = Σ (T x Pn) .....2)

Perhitungan Total skor = 10 + 32 + 45 + 50 + 20 = 157

Perhitungan interval (jarak) dan interpretasi persen untuk mengetahui penilaian dengan metode mencari interval skor persen (I) dengan rumus Interval (3).

Interval (**I**) = 100 / Jumlah Skor .....3)

Perhitungan Interval = 100 / 5 = 20 (interval jarak dari jarak terendah 0% hingga tertinggi 100%).

Kriteria interpretasi skor berdasarkan interval ditunjukkan pada Tabel 2.3:

Tabel 2. 3 Kriteria interpretasi skor

No	Interval (I)	Keterangan
1	0% – 19,99%	Sangat (tidak setuju/buruk/kurang sekali)
2	20% – 39,99%	(Tidak setuju / Kurang baik)
3	40% – 59,99%	Cukup / Netral
4	60% – 79,99%	(Setuju/Baik/suka)
5	80% – 100%	Sangat (setuju/Baik/Suka)

Untuk mendapatkan hasil interpretasi, harus diketahui skor tertinggi (Y) dan terendah (X) untuk item penilaian dengan rumus (4) dan rumus (5):

Y = Skor tertinggi likert x jumlah panelis/audien ……..4)

X = Skor terendah likert x jumlah panelis/audien ……...5)

Perhitungan :

Y = 5 x 70 = 350

X = 1 x 70 = 70

Kemudian dilakukan perhitungan untuk persentase keseluruhan jawaban atau index % dengan rumus (6).

index % = Total skor / Y x 100 ……..6)

Dan didapatkan hasil akhir yang diperoleh ialah:

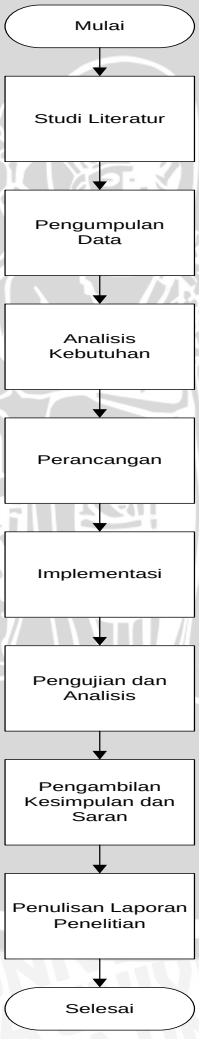
= Total skor / Y x 100

= 157 / 350 x 100

= 44.86 %, masuk dalam kategori Cukup / Netral [FMN-13]

BAB III
METODE PENELITIAN

Pada bab ini dijelaskan langkah-langkah yang akan dilakukan dalam pengerjaan tugas akhir, yaitu studi literatur dan penyusunan dasar teori, pengumpulan data, analisa dan perancangan, implementasi, pengujian dari aplikasi perangkat lunak yang akan dibuat, hingga penulisan laporan. Kesimpulan dan saran disertakan sebagai catatan atas aplikasi dan kemungkinan arah pengembangan aplikasi selanjutnya. Gambar 3.1 menunjukkan tahapan penelitian secara umum.



Gambar 3.1 Tahapan Penelitian

3.1 Studi Literatur

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

1. *Web Service*
2. GPS
3. Google Maps API
4. Android *Location API*
5. Skala Likert

3.2 Pengumpulan Data

Perangkat lunak yang dikembangkan dalam tugas akhir ini adalah Sistem informasi praktik dokter berbasis Perangkat Bergerak. Adapun teknik yang digunakan dalam pengumpulan data yaitu:

1) Observasi

Penelitian lapangan dilakukan secara langsung agar dapat melihat kondisi lapangan secara langsung serta memperoleh data-data yang dibutuhkan dalam pengembangan Sistem informasi praktik dokter berbasis perangkat bergerak. Observasi dilakukan dengan mengunjungi lokasi praktik dokter yang ada di Kota Malang.

2) Riset Pustaka

Riset Pustaka dilakukan dengan cara mengumpulkan data-data atau sumber yang diperoleh dari berbagai referensi yang berfungsi untuk menyelesaikan permasalahan yang ada sehingga dapat membantu dalam menyelesaikan pembuatan aplikasi.

3.3 Tahap Analisis Kebutuhan Sistem

Sebelum melakukan analisis kebutuhan perlu dilakukan pengumpulan *requirement*. Metode pengumpulan *requirement* yang digunakan adalah wawancara . Wawancara di lakukan untuk mengetahui apa saja yang dibutuhkan pengguna sehingga sistem sesuai dengan diharapkan pengguna. Setelah melakukan

pengumpulan *requirement* dengan metode wawancara selanjutnya dilakukan analisis kebutuhan sistem.

Analisis sistem merupakan tahap yang paling penting dalam suatu pengembangan sebuah aplikasi, karena kesalahan pada tahap analisis sistem akan menyebabkan kesalahan pada tahap selanjutnya. Dengan adanya proses ini, diharapkan dapat menentukan sejauh mana aplikasi yang dibuat tersebut dapat mencapai target. Dari proses tersebut akan dihasilkan suatu gambaran sistem yang kemungkinan memiliki kesalahan-kesalahan ataupun kelemahan-kelemahan sehingga dimungkinkan dilakukan perbaikan.

Dalam sebuah sistem pasti sering terjadi suatu kendala untuk mencapai suatu tujuan dan sering kali kita menyadari bahwa masalah itu terjadi setelah sistem yang berjalan tidak benar. Jika terjadi masalah kita harus bisa menemukan penyebabnya, karena masalah timbul tidak dengan sendirinya melainkan ada sebab yang menimbulkan masalah tersebut. Tujuan dilakukannya analisis sistem yaitu untuk memahami sistem yang sedang berjalan.

Analisis kebutuhan digunakan untuk mendapatkan kebutuhan aplikasi praktik dokter yang akan dibangun. Metode analisis yang digunakan adalah *object-oriented analysis* dengan menggunakan bahasa pemodelan UML (*Unified Modeling Language*). Diagram *use case* digunakan untuk mendeskripsikan kebutuhan-kebutuhan dan fungsionalitas perangkat lunak dari sudut pandang pengguna. Analisis kebutuhan dilakukan dengan mengidentifikasi semua kebutuhan aplikasi praktik dokter yang kemudian akan dimodelkan dalam diagram *use case*.

3.3 Perancangan

Perancangan perangkat lunak dilakukan setelah semua kebutuhan perangkat lunak didapatkan melalui tahap analisis kebutuhan. Perancangan perangkat lunak berdasarkan *object-oriented analysis* dan *object-oriented design* yaitu menggunakan pemodelan UML (*Unified Modeling Language*). Perancangan dimulai dari perancangan alur atau aktifitas yang dilakukan pengguna secara prosedural yang

dimodelkan dalam *activity diagram*. Interaksi antar objek yang telah diidentifikasi, dimodelkan dalam *sequence diagram*. Selanjutnya, dilakukan perancangan sistem aplikasi praktik dokter dengan mengidentifikasi *class* yang dibutuhkan, serta kemudian dimodelkan dalam *class diagram*. Kemudian tahap perancangan dilanjutkan dengan perancangan antarmuka pengguna.

3.4 Implementasi

Implementasi perangkat lunak mengacu kepada perancangan perangkat lunak. Implementasi perangkat lunak diawali dengan penjabaran spesifikasi lingkungan perancangan perangkat lunak. Implementasi perangkat lunak dilakukan dengan menggunakan bahasa pemrograman Java pada aplikasi pengguna. Implementasi antarmuka berdasarkan perancangan yang telah dilakukan.

3.5 Pengujian dan Analisis

Pegujian yang dilakukan pada aplikasi ini yaitu pengujian *validasi*, pengujian UAT(*User Acceptance Test*). Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah berjalan seperti yang diharapkan. Pengujian validasi ini dilakukan berdasarkan *use case* yang ada pada tahap perancangan. Pengujian UAT dilakukan dengan melibatkan pengguna, Pengujian UAT bertujuan untuk melihat tingkat penerimaan pengguna terhadap aplikasi dengan variable penilaian pencapaian kegunaan *(usefulness)*.

3.6 Pengambilan Kesimpulan dan Saran

Pengambilan kesimpulan dilakukan setelah semua tahapan perancangan perangkat lunak, implementasi perangkat lunak, dan pengujian perangkat lunak telah selesai dilakukan. Kesimpulan diambil dari hasil pengujian dan analisis terhadap sistem yang dibangun. Tahap terakhir dari penulisan adalah saran yang dimaksudkan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan perangkat lunak lebih lanjut.

BAB IV
ANALISIS DAN PERANCANGAN

Pada bab ini menjelaskan tentang analisis sistem dan desain dari aplikasi praktik dokter pada perangkat bergerak Android. Analisis sistem dan desain sistem memiliki dua tahap. Tahap pertama adalah proses analisis kebutuhan kemudian pada tahap kedua proses perancangan perangkat lunak. Tahap analisis kebutuhan terdiri dari gambaran umum alur aplikasi, identifikasi aktor, mendaftar kebutuhan yang diperlukan, memodelkan ke dalam *use case* dan diagram *activity*. Pada tahap perancangan terdiri dari perancangan arsitektur, perancangan basis data, pemodelan diagram *sequence* untuk penggambaran interaksi antar objek di dalam aplikasi praktik dokter, pemodelan diagram class untuk menggambarkan perancangan struktur class-class yang menyusun aplikasi praktik dokter. Dan perancangan antarmuka pengguna.

4.1. Analisis Kebutuhan Sistem

Analisis kebutuhan diawali dengan melakukan pengumpulan *requirement* dengan menggunakan metode wawancara untuk mengetahui apa saja yang dibutuhkan, Sehingga sistem sesuai dengan yang di harapkan pengguna. Setelah mendapatkan *requirement* maka dilakukan pengumpulan data ke beberapa tempat praktik dokter di kota Malang seperti RS Saiful Anwar, RS lavallete, RS Unisma dan dilakukan pencatatan data temtang praktik dokter untuk memenuhi *requirement* yang dibutuhkan oleh pengguna. Selanjutnya dilakukan analisis dengan menggunakan metode *object-oriented analysis* dengan menggunakan bahasa pemodelan UML (*Unified Modeling Language*) yang akan di bahas dibawah berikut.

4.1.1. Gambaran Umum Aplikasi

Aplikasi praktik dokter ini merupakan aplikasi yang dirancang untuk membantu pengguna dalam mencari informasi daftar dokter, jadwal dokter dan lokasi praktik dokter. Aplikasi ini berjalan dengan menggunakan koneksi internet, ketika aplikasi dijalankan pengguna akan mendapatkan tampilan daftar spesialisasi dokter yang akan di pilih oleh pengguna , aplikasi akan mencari daftar dokter yang sesuai

dengan spesialisasi yang telah di pilih, selanjutnya penggunaakan memilih dokter yang sesuai dengan keinginannya yang akan menampilkan informasi jadwal dokter tersebut, dan pengguna juga dapat melihat rute menuju lokasi dokter tersebut.

4.1.2. Identifikasi Aktor

Identifikasi aktor dilakukan untuk menentukan aktor yang terlibat dengan sistem. Aktor merupakan segala sesuatu yang berada di luar sistem yang akan menggunakan aplikasi praktik dokter. Pada aplikasi praktik dokter terdapat satu aktor yang memiliki peran dalam menjalankan aplikasi ini yaitu pengguna. Peran dari aktor dapat dilihat pada tabel 4.1.

Tabel 4. 1 Tabel aktor

No	Aktor	Deskripsi
1	Pengguna (*User*)	Merupakan pengguna yang menggunakan aplikasi praktik dokter sebagai media untuk mencari informasi mengenai praktik dokter.
2	Admin	Merupakan aktor yang mengelola data dokter. Admin akan mengelola data melalui web.

4.1.3. Daftar Kebutuhan

Daftar kebutuhan terdiri dari kebutuhan fungsional dan non-fungsional. Pada tabel 4.2 kebutuhan fungsional ditunjukan dengan kode penomoran FXX.

Tabel 4. 2 Tabel kebutuhan fungsional

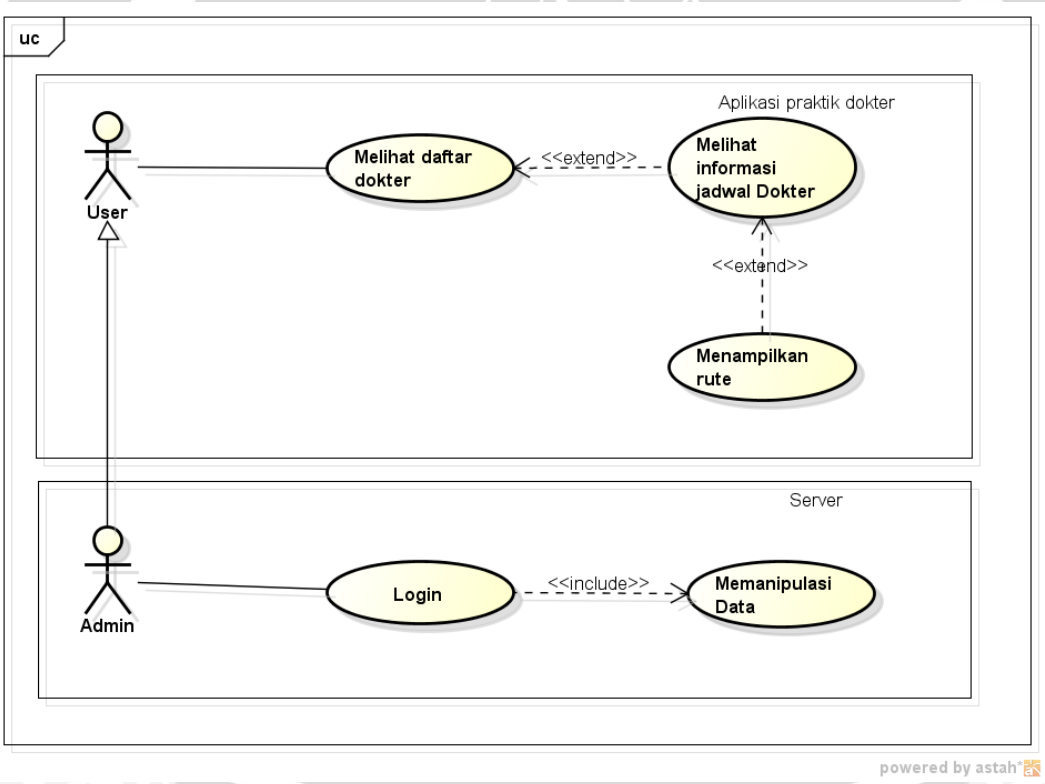
Kode	Kebutuhan	Use case
F01	Aplikasi harus bisa menampilkan daftar dokter sesuai dengan spesialisasi yang dipilih pengguna	Melihat daftar dokter
F02	Aplikasi harus bisa menampilkan informasi hari, jam, tempat praktik, dan alamat praktik	Melihat informasi dokter
F03	Aplikasi harus bisa menampilkan rute menuju lokasi dokter yang dipilih pengguna	Melihat rute lokasi

F04	Aplikasi harus bisa mengelola data dokter	Memanipulasi data

Tabel 4.2 merupakan tabel kebutuhan fungsional yang terdiri dari empat kebutuhan sistem yang harus dapat dipenuhi oleh sistem. Setiap kebutuhan funsional sistem dimodelkan dalam bentuk *use case*, dan setiap kebutuhan fungsional harus dilakukan pengujian terlebih dahulu sebelum sistem di publikasikan

4.1.4. Diagram *Use Case*

Pemodelan *use case* sistem diperoleh dari kebutuhan funsional yang digunakan untuk menggambarkan interaksi antara satu atau lebih aktor dengan sistem yang akan dibuat. Gambar 4.1 menunjukan use case aplikasi praktik dokter.



Gambar 4. 1 *Use Case* sistem informasi praktik dokter

Gambar 4.1 merupakan *use case* sistem praktik dokter yang terdiri dari 2 aktor yaitu pengguna (*user*) dan admin, dimana admin dapat memanipulasi data dokter dan

mengakses aplikasi praktik dokter sedangkan *user* dapat melakukan akses aplikasi praktik dokter dan tidak dapat memanipulasi data praktik dokter.

4.1.5. Skenario Use Case

Tabel 4. 3 Skenario Use Case Melihat daftar dokter

Kode *Use Case*	F01
Nama *Use Case*	Melihat daftar dokter
Persyaratan Kontek	Aplikasi telah terpasang pada telephone pintar Android pengguna
Tujuan Dalam Kontek	Mempermudah pengguna untuk mengetahui daftar dokter yang sesuai dengan spesialisasi yang dipilih.
Prakondisi	Aplikasi sudah terpasang dan pengguna membuka aplikasi praktik dokter
Kondisi Akhir Sukses	Aplikasi menampilkan form spesialisasi, dan aplikasi akan mencarikan dokter yang sesuai dengan spesialisasi yang dipilih.
Kondisi Akhir Failed	-
Aktor	Pengguna
Alur Utama	Aktifitas
	1. Pengguna membuka aplikasi praktik dokter 2. Sistem menampilkan daftar spesialisasi 3. Pengguna memilih spesialisasi 4. Aplikasi akan menampilkan dokter yang sesuai dengan spesialisasi yang dipilih pengguna.

Tabel 4.3 menjelaskan kegiatan yang dilakukan pada saat melihat daftar dokter pada gambar 4.1 *use case* praktikdokter. Pertama pengguna menjalankan aplikasi yang telah terpasang pada telephone pintar Android lalu aplikasi menampilkan daftar spesialisasi dokter, dan aplikasi akan mencarikan dokter yang sesuai dengan spesialisasi yang dipilih.

Tabel 4. 4 Skenario Use Case Melihat Informasi dokter

Kode *Use Case*	F02
Nama *Use Case*	Melihat informasi dokter
Persyaratan Kontek	Aplikasi telah terpasang pada telephone pintar Android pengguna
Tujuan Dalam Kontek	Mempermudah pengguna dalam mengetahui informasi hari praktik, jam praktik, tempat praktik, alamat praktik dokter.
Prakondisi	Pengguna sudah memilih dokter berdasarkan spesialisasinya.
Kondisi Akhir Sukses	Aplikasi menampilkan informasi hari, jam, tempat praktik, dan alamat praktik dokter
Kondisi Akhir Failed	-
Aktor	Pengguna
Alur Utama	Aktifitas
	1. Pengguna membuka aplikasi praktik dokter 2. Sistem menampilkan form spesialisasi dokter, 3. Pengguna memilih spesialisasi 4. Aplikasi akan menampilkan dokter yang sesuai dengan spesialisasi yang dipilih 5. Pengguna memilih dokter yang sudah di tampilkan oleh aplikasi 6. Aplikasi menampilkan informasi hari, jam, tempat praktik, dan alamat praktik

Tabel 4.4 menjelaskan kegiatan yang dilakukan pada saat melihat informasi dokter pada gambar 4.1 *use case* layanan kesehatan. Pertama pengguna menjalankan

aplikasi yang telah terpasang pada telephone pintar Android lalu aplikasi menampilkan daftar spesialisasi dokter, dan aplikasi akan mencarikan dokter yang sesuai dengan spesialisasi yang dipilih.Setelah itu pengguna akan memilih dokter yang telah ditampilkan aplikasi, dan pengguna akan memilih salah satu untuk mendapatkan informasi dari dokter yang telah di pilih.

Tabel 4. 5 Skenario Use Case Melihat rute lokasi dokter

Kode *Use Case*	F03
Nama *Use Case*	Melihat rute lokasi
Persyaratan Kontek	Aplikasi telah terpasang pada telephone pintar Android pengguna
Tujuan Dalam Kontek	Menampilkan rute menuju lokasi dokter yang dipilih pengguna.
Prakondisi	Pengguna melihat informasi dokter
Kondisi Akhir Sukses	Aplikasi menampilkan rute menuju lokasi dokter
Kondisi Akhir Failed	-
Aktor	Pengguna
Alur Utama	Aktifitas
	1. Pengguna membuka aplikasi praktik dokter 2. Sistem menampilkan form spesialisasi dokter, 3. Pengguna memilih spesialisasi 4. Aplikasi akan mencarikan dokter yang sesuai dengan spesialisasi yang dipilih. 5. Pengguna memilih dokter yang sudah di telah di tampilkan oleh aplikasi 6. Aplikasi menampilkan informasi hari, jam, tempat praktik, dan alamat praktik 7. Pengguna memilih jadwal, sesuai dengan jadwal

	yang di pilih. 8. Aplikasi akan menampilkan rute menuju lokasi dokter yang di maksud.

Tabel 4.5 menjelaskan kegiatan yang dilakukan pada saat melihat rute lokasi dokter pada gambar 4.1 *use case* praktik dokter. Pertama pengguna menjalankan aplikasi yang telah terpasang pada telephone pintar Android lalu aplikasi menampilkan daftar spesialisasi dokter, dan aplikasi akan mencarikan dokter yang sesuai dengan spesialisasi yang dipilih.Setelah itu pengguna akan memilih dokter yang telah di tampilkan oleh aplikasi, dan pengguna akan memilih salah satu untuk mendapatkan informasi dari dokter yang telah di pilih. Dan terakhir pengguna bisa melihat rute menuju dokter yang di maksud.

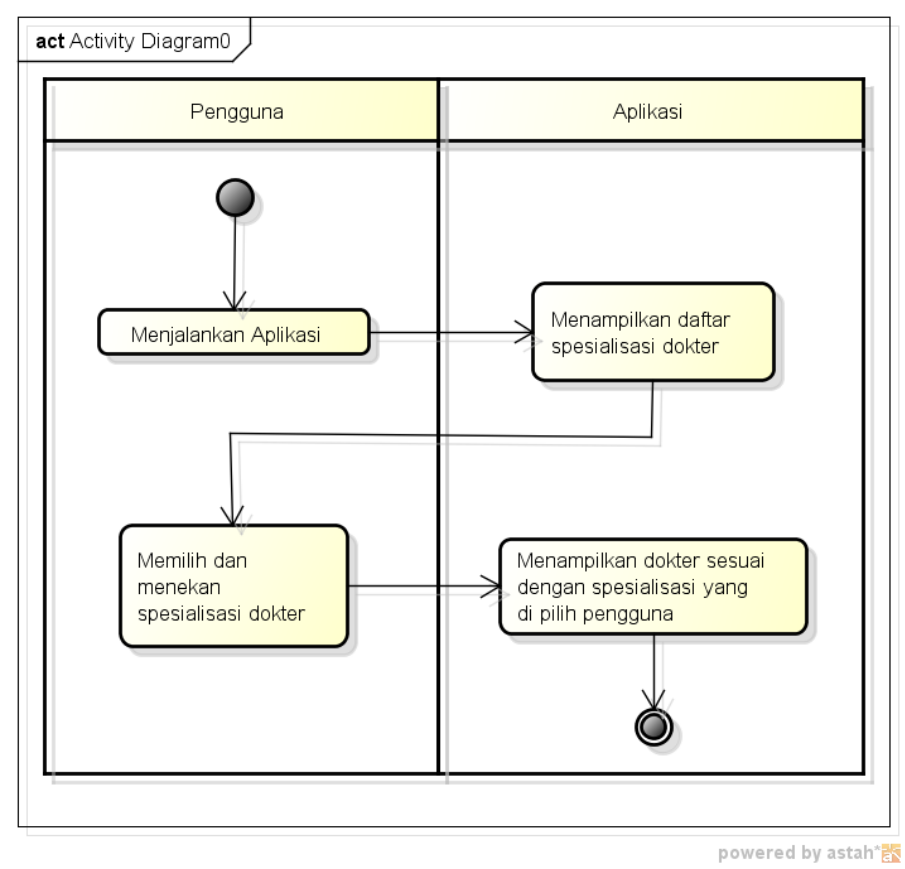
Tabel 4. 6 Skenario Use Case Memanipulasi data

Kode *Use Case*	F04
Nama *Use Case*	Memanipulasi data
Persyaratan Kontek	Aplikasi telah berhasil di pasang di server.
Tujuan Dalam Kontek	Untuk mengelola data praktik dokter
Prakondisi	Aplikasi sudah terpasang dan admin membuka aplikasi server.
Kondisi Akhir Sukses	Admin dapat memanipulasi data
Kondisi Akhir Failed	-
Aktor	Admin
Alur Utama	Aktifitas
	1. Admin Login 2. Admin Memanipulasi data

Tabel 4.6 menjelaskan kegiatan yang dilakukan pada saat memanipulasi data dokter pada gambar 4.1 *use case* praktik dokter. Pertama admin melakukan login ke aplikasi , dan setelah berhasil login admin dapat melalukan manipulasi data dokter.

4.1.6. Diagram Activity

Diagram *activity* adalah diagram untuk memodelkan aktifitas antara pengguna dan sistem yang berjalan berdasarkan pada sekenario *use case*. Skenario *use case* yang digambarkan pada tabel 4.3 dapat dimodelkan ke dalam diagram *activity* pada gambar 4.2 dengan diagram tersebut dapat terlihat aktor yang melakukan proses tiap langkahnya.



Gambar 4. 2 Aktivitas diagram melihat daftar dokter

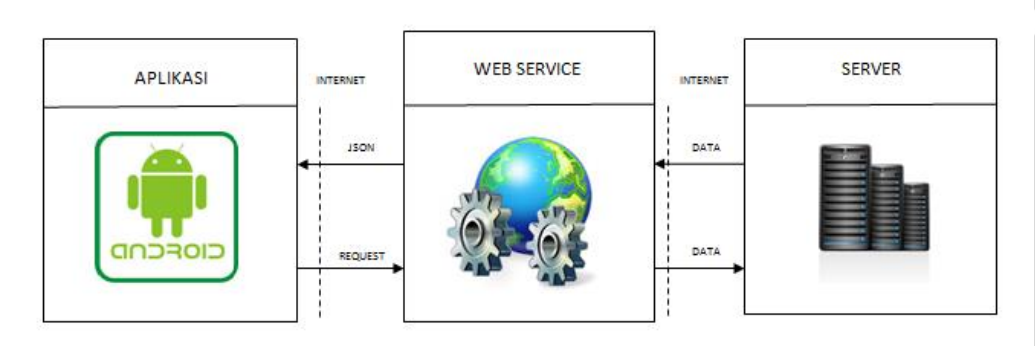
Gambar 4.2 merupakan aktifitas diagram melihat daftar dokter yang digunakan untuk menjelaskan interaksi antara pengguna dengan sistem. Pertama

pengguna menjalankan aplikasi praktik dokter lalu aplikasi menampilkan daftar spesialisasi dokter, dan aplikasi akan mencarikan dokter yang sesuai dengan spesialisasi yang dipilih. Aktifitas diagram untuk aktifitas lainnya bisa di lihat di lampiran.

4.2. Perancangan Perangkat Lunak

4.2.1. Perancangan Arsitektur

Gambar 4.3 merupakan gambaran arsitektur dari aplikasi praktik dokter

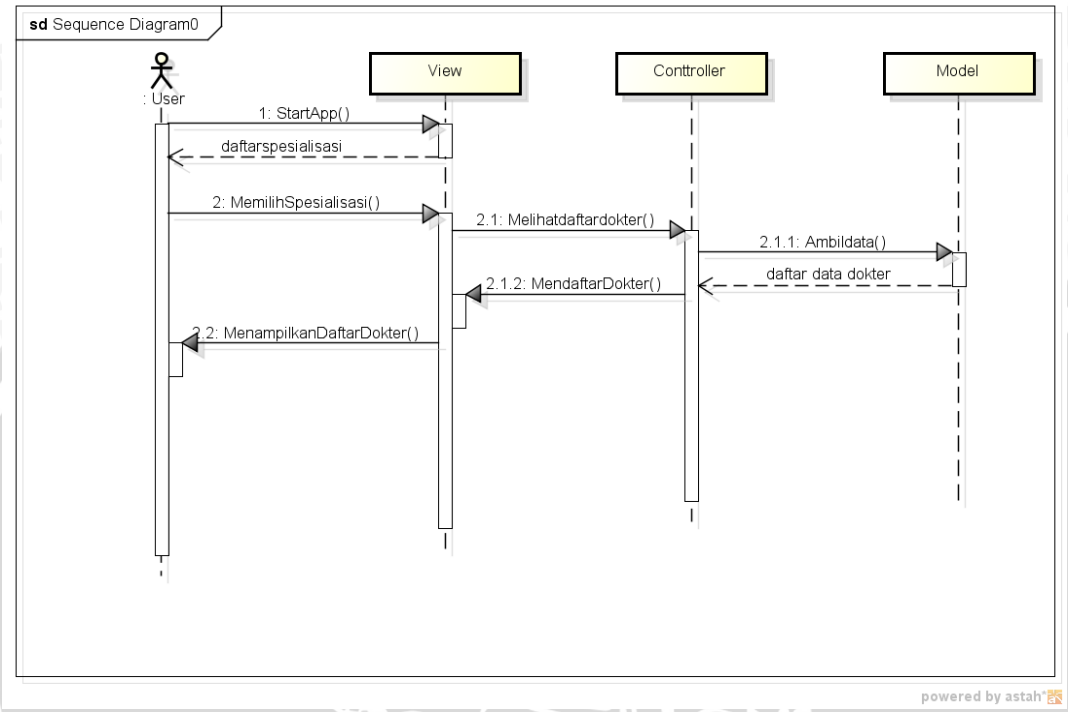


Gambar 4. 3 Arsitektur sistem aplikasi sistem informasi praktik dokter

Gambar 4.3 Proses kerja sistem dimulai dari aplikasi klien yang melakukan *request* yang dilakukan oleh pengguna. *Request* tersebut akan dikirimkan ke *web service* yang berfungsi sebagai penghubung antara klien dan server. *Web service* ini akan melakukan proses terhadap server untuk mendapatkan data sesuai dengan *request* dari aplikasi klien. Data yang ditemukan selanjutnya akan diproses terlebih dahulu untuk selanjutnya dikirimkan kembali ke aplikasi klien.

4.2.2. Perancangan Diagram *Sequence*

Diagram *sequence* digunakan untuk menggambarkan interaksi antar objek di dalam aplikasi praktik dokter.

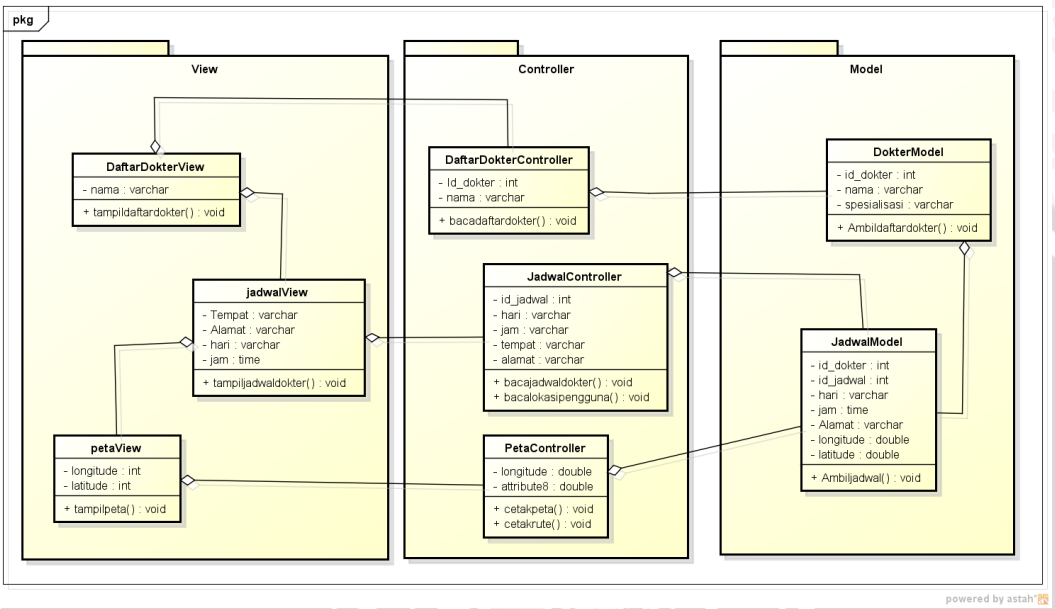


Gambar 4. 4 Diagram sequence melihat daftar dokter

Pada gambar 4.4 dijelaskan pada saat pengguna menjalankan aplikasi praktik dokter maka aplikasi akan menjalankan method `StartApp()` untuk menampilkan daftar spesialisasi dokter. Selanjutnya aplikasi akan melakukan pengambilan data dokter sesuai dengan spesialisasinya yang telah tersimpan di *database*. Setelah data selesai diambil selanjutnya menampilkan daftar dokter sesuai spesialisasinya. Diagram sequence yang lain dapat dilihat pada lampiran.

4.2.3. Perancangan Diagram *Class*

Berdasarkan gambar 4.4, gambar 4.10, dan gambar 4.11 maka di dapatkan pemodelan diagram *class* untuk menggambarkan perancangan struktur *class-class* yang menyusun aplikasi .

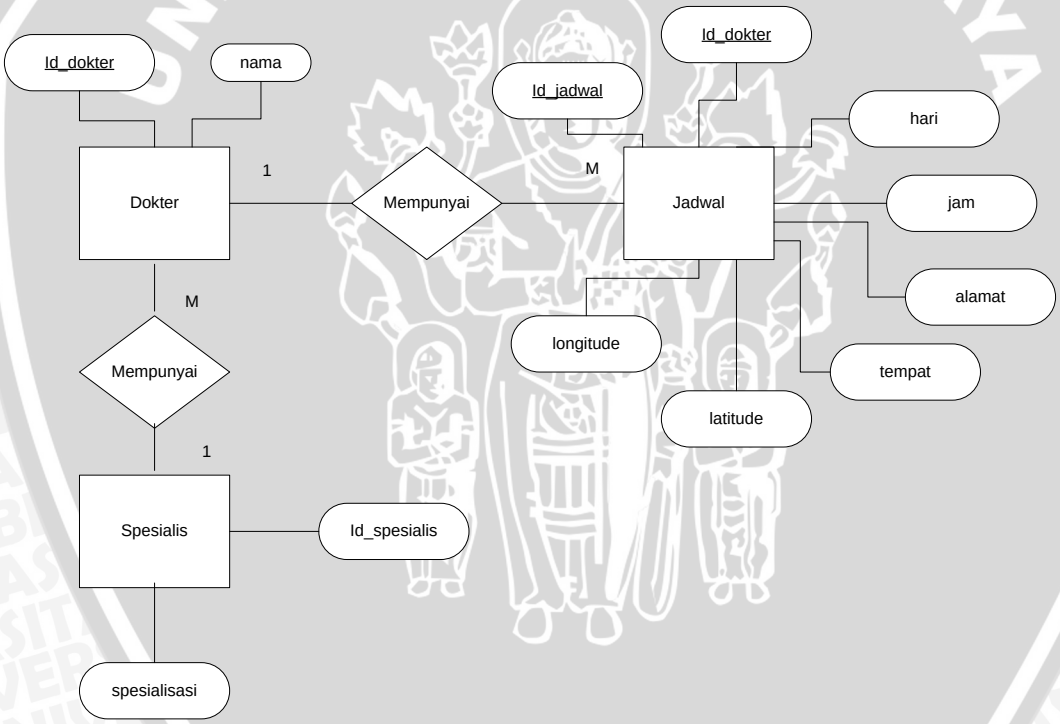


Gambar 4. 5 Diagram Class aplikasi praktik dokter

Gambar 4.5 diatas menjelaskan tentang *class-class* yang terdapat pada aplikasi praktik dokter, terdapat 3 *package* pada *class diagram* aplikasi dokter yaitu *package view*, *controller*, dan *model*. *Package view* berisi *class* `datadokterview` untuk menampilkan daftar dokter dan mempunyai relasi *aggregation* dengan *class* `jadwaldokter` untuk menampilkan informasi dokter yang berarti merupakan bagian dari class `daftadokterview`. *Class peta* berfungsi untuk menampilkan peta dan rute menuju lokasi dokter.Selanjutnya *package controller* yang berisi *class* `daftardoktercontroller`, `jadwalcontroller`, dan `petacontroller` yang berfungsi untuk proses yang akan dilakukan oleh aplikasi praktik dokter .dan terakhir adalah *package model* yang berisi *class* `doktermodel` dan `jadwalmodel` yang berisi data.

4.2.4. Perancangan Basis Data

Basis data berfungsi sebagai tempat menyimpan data. Pada penelitian ini perancangan basis data direpresentasikan dalam bentuk *Entity Relationship Diagram* (ERD). ERD menunjukkan hubungan yang terjadi diantara objek(entitas) yang terlibat dalam suatu database.ERD berisi komponen-komponen himpunan entitas dan himpunan relasi yang masing-masing dilengkapi dengan beberapa atribut yang mempresentasikan seluruh fakta yang ditinjau dari keadaan yang nyata.Pada perancangan basis data sistem ini terdapat dua buah tabel yaitu tabel dokter dan tabel jadwal. ERD sistem ini dapat dilihat dalam Gambar 4.6



Gambar 4. 6 *Entity Relational Diagram*

Berikut ini merupakan struktur tabel serta keterangan masing-masing tabel dan field yang ada pada database.

Tabel 4. 7 Rancangan tabel spesialis untuk menyimpan data spesialis dokter

Tabel Dokter	
Kolom	**Tipe Data**
id_spesialis	int
Spesialisasi	varchar

Tabel 4. 8 Rancangan tabel dokter untuk menyimpan data dokter

Tabel Dokter	
Kolom	**Tipe Data**
id_dokter	int
Nama	varchar
Spesialisasi	varchar

Tabel 4. 9 Rancangan tabel jadwal untuk menyimpan jadwal dokter

Tabel Jadwal	
Kolom	**Tipe Data**
id_dokter	int
id_jadwal	int
Tempat	varchar
Alamat praktik	varchar
Hari	varchar
jam	varchar
longitude	double
latitude	double

4.2.5. Perancangan Antarmuka

Perancangan antarmuka merupakan rancangan untuk implementasi antarmuka pengguna pada aplikasi praktik dokter. Berikut merupakan tampilan antarmuka aplikasi praktik dokter.

Gambar 4. 7 Rancangan antar muka aplikasi praktek dokter

Untuk rancangan antarmuka untuk melihat daftar dokter, melihat informasi dokter dan penunjuk rute jalan dapat dilihat pada lampiran .

BAB V
IMPLEMENTASI DAN PENGUJIAN

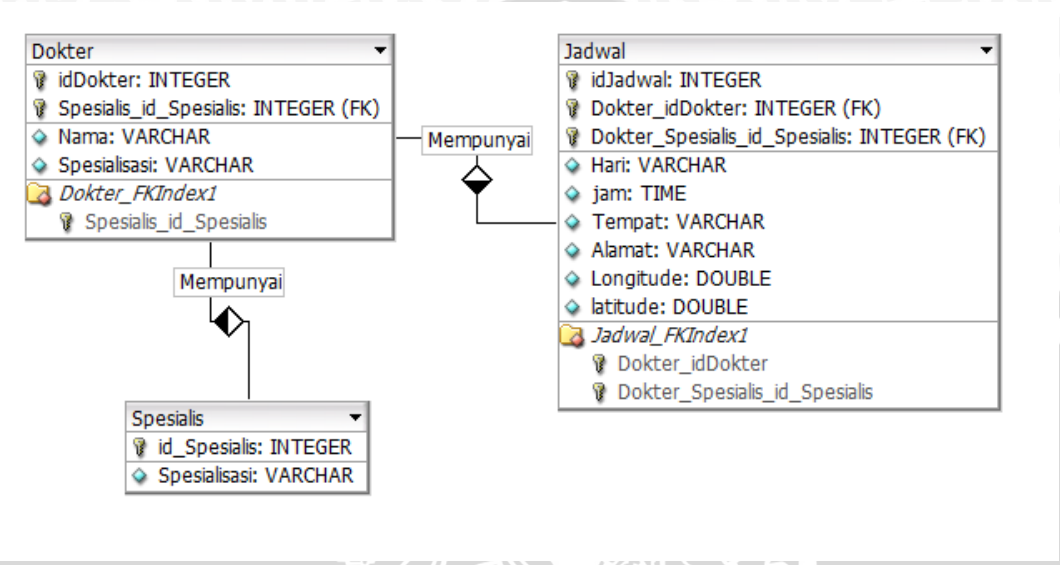
Bab ini membahas mengenai tahapan implementasi dan pengujian aplikasi sistem informasi praktik dokter berbasis perangkat bergerak. berdasarkan dari hasil analisis kebutuhan dan perancangan aplikasi. Pembahasan terdiri dari spesifikasi perangkat keras dan perangkat lunak, implementasi basis data, implementasi *class*, implementasi kode program dan implementasi antarmuka. Kemudian dilanjutkan dengan pengujian validasi, dan pengujian UAT (*User Acceptance Test*) pada aplikasi sistem informasi praktik dokter berbasis perangkat bergerak.

5.1 Spesifikasi Perangkat Keras dan Perangkat Lunak

Dalam pengembangan aplikasi sistem informasi praktik dokter berbasis perangkat bergerak ini menggunakan sebuah komputer dengan operasi sistem windows 7 dan perangkat lunak yang digunakan untuk membuat aplikasi tersebut yaitu Eclipse Juno, ADT (Android Developer Tools) serta untuk uji coba aplikasi sistem informasi praktik dokter ini menggunakan Samsung galaxy tab 3 dengan operasi sistem android 4.2.2 (Jelly Bean).

5.2 Implementasi Basis Data

Implementasi pada penyimpanan data dilakukan dengan MySQL. Hasil implementasi penyimpanan data ini berupa file basis data dengan format ekstensi file sql. Hasil implementasi MySQL pada basis data ini dimodelkan dalam *physical diagram*. Pada *physical diagram* terdapat hubungan relasi antar tabel. Dalam Gambar 5.1 menggambarkan *physical diagram* dari aplikasi sistem informasi praktik dokter berbasis perangkat bergerak.



Gambar 5. 1 *physical diagram*

5.3 Implementasi *Class*

Setiap *class* yang telah dirancang pada proses perancangan direalisasikan pada sebuah *file* program dengan ekstensi *.java dan *file layout* dengan ekstensi *.xml sebagai tampilan dari *class* tersebut. *Class* dalam perancangan yang hanya berisi *method* tidak memiliki *layout* untuk menampilkan isinya. Tabel 5.1 menjelaskan mengenai pasangan antara *class* dengan *file* program dan *layout* yang digunakan untuk mengimplementasikannya.

Tabel 5. 1 Implementasi *class* dan *layout* untuk program

No	Package	Objek	Nama Class	Nama File Program	Nama File Layout
1	cob.a.petax	View	DaftarDokter	DaftarDokter.Java	Activity_main_dokter.xml
2	cob.a.petax	View	Jadwal	Jadwal.Java	Activity_main_jadwal.xml

3	cob.a.petax	View	Peta	Peta.Java	Peta.xml
4	cob.a.petax	Controller	DaftarDokterController	DaftarDokter.Java	-
5	cob.a.petax	Controller	JadwalController	Jadwal.Java	-
6	cob.a.petax	Controller	PetaController	Peta.Java	-
7	cob.a.petax	Model	DokterModel	DokterModel.java	-
8	cob.a.petax	Model	JadwalModel	JadwalModel.java	-

5.4 Implementasi kode

Implementasi kode program JSON parser terdapat beberapa bagian penting. kode 5.1 merupakan potongan kode untuk pengambilan data JSON dari *web service*, data hasil pengambilan tersebut lalu akan dimasukkan ke objek `jObj`.

Kode 5. 1 Potongan kode untuk pengambilan JSON dari *web service*

```
package cob.a.petax;
. . .
public class JSONParser {
. . .
}
  public JSONObject AmbilJson(String url) {
  try {
  DefaultHttpClient httpClient = new DefaultHttpClient(); HttpPost
httpPost = new HttpPost(url);
  HttpResponse httpResponse = httpClient.execute(httpPost);
  HttpEntity httpEntity = httpResponse.getEntity();
  is = httpEntity.getContent();
  } catch (UnsupportedEncodingException e) {
  e.printStackTrace();
  } catch (ClientProtocolException e) {
  e.printStackTrace();
  } catch (IOException e) {
  e.printStackTrace();
  }
  try {
```

```
  BufferedReader reader = new BufferedReader(new
InputStreamReader(
  is, "iso-8859-1"), 8);
  StringBuilder sb = new StringBuilder();
  String line = null;
  while ((line = reader.readLine()) != null) {
  sb.append(line + "\n");
  }
  is.close();
  json = sb.toString();
  } catch (Exception e) {
  Log.e("Buffer Error", "Error converting result " +
e.toString());
          }
  try {
  jObj = new JSONObject(json);
  } catch (JSONException e) {
  Log.e("JSON Parser", "Error parsing data " + e.toString());
  }
  return jObj;
  }
}
```

Kode 5.1 merupakan potongan kode untuk pengambilan JSON dari *web serivce* . ketika pengguna meminta data kepada server maka *web service* akan memproses data dan akan mengirimkan data dengan format JSON kepada aplikasi pengguna . Gambar 5. 2 dan 5. 3 merupakan contoh data yang dikirimkan dengan format JSON dari server.

```
{dokter:[{"id_dokter":"1","nama":"Prof. DR . dr . Djanggan S.,SP,
PD, SP, JP"}]}
```

Gambar 5. 2 Contoh respon JSON daftar dokter dari server

```
{jadwal:[{"id_jadwal":"1","id_dokter":"1","hari":"Senin","jam":"07.0
0 - 09.00","Tempat":"Rs saiful anwar","alamat":" Jl. Jaksa Agung
Suprapto No. 2, Malang","longitude":"112.631334","latitude":"-
7.971764"}]}
```

Gambar 5. 3 Contoh respon JSON jadwal dokter dari server

Implementasi kode program untuk Gambar 4.5 *class diagram* terdapat beberapa *class* yaitu *view*, *controler* dan *model*. Dari masing-masing *class* terdapat beberapa yang akan diimplementasikan yaitu untuk menampilkan daftar dokter untuk *classview*, untuk membaca daftar dokter, membaca lokasi pengguna, mencetak peta, mencetak rute untuk *class controller*, *method* dan untuk mengambil daftar dokter untuk *class model.*

Implementasi kode program untuk *use case* melihat daftar dokter terdapat beberapa bagian penting. Kode 5.2 merupakan potongan kode untuk menampilkan daftar dokter dari *class* `DaftarDokter`

Kode 5. 2 Potongan kode untuk menampilkan daftar dokter

```
package cob.a.petax;
. . .
 setContentView(R.layout.activity_main_dokter);
 Bundle b = getIntent().getExtras();
 spesialis = b.getString("spesialis");
. . .
}
```

Kode 5.2 merupakan potongan kode implementasi dari *class view* untuk menampilkan daftar dokter dengan menggunakan *method* `setContentView(R.layout.activity_main_dokter)`. Ketika aplikasi berjalan, aplikasi memanggil *layout* `activity_main_dokter.xml` sebagai antarmuka untuk aplikasi yang di dalam *layout* tersebut terdapat daftar dokter yang sudah di isi oleh controller .

Implementasi kode program untuk *use case* melihat daftar dokter terdapat beberapa bagian penting . Kode 5.3 merupakan potongan kode untuk membaca daftar dokter dari *class* `DokterController`

Kode 5.3 Potongan kode untuk membaca daftar dokter

```
package cob.a.petax;
. . .
  protected String doInBackground(String... args) {
. . .
```

```
  try {
  str_json = json.getJSONArray("daftar_dokter");

  for(int i = 0; i < str_json.length(); i++){
  JSONObject ar = str_json.getJSONObject(i);

  HashMap<String, String> map = new HashMap<String, String>();

  String id_dokter = ar.getString("id_dokter");
  String nama = ar.getString("Nama");

  map.put(in_id_dokter, id_dokter);
  map.put(in_nama, nama);

  data_map.add(map);
  }
. . .
}
```

Kode 5.3 merupakan potongan kode implementasi dari *class controller* untuk membaca daftar dokter dengan menggunakan *method* `str_json = json.getJSONArray("daftar_dokter")`. Ketika pengguna sudah memilih spesialisasi di halama utama, maka controller akan membaca daftar dokter di *class* `doktermodel()` sesuai dengan spesialisasi pilihan pengguna. Selanjutnya *controller* akan memproses daftar dokter yang sudah didapatkan yang akan di tampilkan di *class view*.

Implementasi kode program untuk *use case* melihat daftar dokter terdapat beberapa bagian penting . Kode 5.4 merupakan potongan kode untuk mengambil daftar dokter dari *class* `DokterModel`

Kode 5.4 Potongan kode untuk mengambil daftar dokter

```
package cob.a.petax;
. . .
 String link_url = isi+spesialis;
 JSONParser jParser = new JSONParser();
 JSONObject json = jParser.AmbilJson(link_url);
 return json;
. . .
}
```

Kode 5.4 merupakan potongan kode implementasi dari *class model* untuk mengambil daftar dokter dengan menggunakan *method* `JSONObject json = jParser.AmbilJson(link_url)`, ketika *class controller* memanggil *class* ini maka *class* ini akan mengambil data daftar dokter yang berasal dari *web service* yang kemudian akan di kembalikan kepada *class controller* dan akan di tampilkan di *class view*.

Implementasi kode program untuk mengambil koordinat lokasi pengguna terdapat beberapa bagian penting . Kode 5.5 merupakan potongan kode untuk mengambil lokasi pengguna dari *class* `JadwalController` untuk mengambil koordinat lokasi pengguna

Kode 5. 5 Potongan kode untuk mengambil lokasi pengguna

```
package cob.a.petax;
. . .
 Location location = locationmanager.getLastKnownLocation
(LocationManager.GPS_PROVIDER)
. . .
 Longitude = location.getLongitude();
 Latitude = location.getLatitude();
.
.
 public void onLocationChanged(Location location) {
 String message = String.format(
 "New Location \n Longitude: %1$s \n Latitude: %2$s",
 location.getLongitude(), location.getLatitude()
  );
. . .
}
```

Kode 5.5 merupakan kode implementasi dari *class controller* untuk mengambil lokasi pengguna dengan menggunakan *method* `Location location=locationmanager.getLastKnownLocation(LocationManager.GPS_PROVIDER)`, *method* ini untuk pengambilan titik lokasi dari pengguna yang nantinya akan ditampilkan pada peta dan digunakan untuk dasar petunjuk rute.

Implementasi kode program untuk mencetak peta terdapat beberapa bagian penting. Kode 5.6 merupakan potongan kode untuk mencetak peta dari *class* `petacontroller` untuk mencetak peta

Kode 5. 6 Potongan kode untuk mencetak peta

```
package cob.a.petax;
. . .
 mMap                          =                          ((MapFragment)
getFragmentManager().findFragmentById(R.id.map))
                              .getMap();
. . .
}
```

Kode 5.6 merupakan potongan kode implementasi dari *class* `PetaController` untuk mencetak peta dengan menggunakan *method* mMap = `((MapFragment)getFragmentManager().findFragmentById(R.id.map)).getMap()`, *Method* ini digunakan untuk mengambil peta *Google Maps*. *method* bawaan dari *Google Maps* yaitu getMap().

Implementasi kode program untuk melihat rute terdapat beberapa bagian penting. Kode 5.7 merupakan potongan kode untuk mencetak rute dari *class* `petacontroller` untuk mencetak rute.

Kode 5.7 Potongan kode untuk mencetak rute

```
package cob.a.petax;
. . .
 PolylineOptions rectLine = new PolylineOptions().width(3).color(
Color.RED);

 for (int i = 0; i < directionPoint.size(); i++) {
 rectLine.add(directionPoint.get(i));
}

 mMap.addPolyline(rectLine);
. . .
}
```

Kode 5.7 merupakan kode implementasi dari *class* `PetaController` untuk mencetak rute dengan menggunakan *method* `mMap.addPolyline(rectLine),` *Method* ini digunakan untuk mencetak rute.

Kode 5. 8 Kode untuk menambah data dokter

```
<?php
include "koneksi.php" ;

$simpan = "insert into dokter set id_dokter='$_POST[id_dokter]' ,
nama='$_POST[Nama]' , spesialisasi='$_POST[Spesialisasi]' " ;

mysql_query ($simpan) ;

header ('location:updatedokter.php') ;

?>
```

Kode 5.8 merupakan kode implementasi untuk menambah data dokter di web yang dilakukan oleh admin.

Kode 5. 9 Kode untuk menambah data jadwal dokter

```
<?php
include "koneksi.php" ;

$simpan = "insert into jadwal set id_jadwal='$_POST[id_jadwal]' ,
id_dokter='$_POST[id_dokter]'          ,          hari='$_POST[hari]',
jam='$_POST[jam]',                    alamat='$_POST[alamat]',
longitude='$_POST[longitude]', latitude='$_POST[latitude]'" ;

mysql_query ($simpan) ;

header ('location:updatejadwal.php') ;

?>
```

Kode 5.9 merupakan kode implementasi untuk menambah data jadwal dokter di web yang dilakukan oleh admin.

Kode 5. 10 Kode untuk menambah data spesialisasi dokter

```
<?php
include "koneksi.php" ;

$simpan = "insert into spesialis set
id_spesialis='$_POST[id_spesialis]' ,
spesialisasi='$_POST[spesialisasi]'" ;

mysql_query ($simpan) ;

header ('location:spesialisasi.php') ;

?>
```

Kode 5.10 merupakan kode implementasi untuk menambah data spesialisasi dokter di web yang dilakukan oleh admin.

5.5 Implementasi Antarmuka

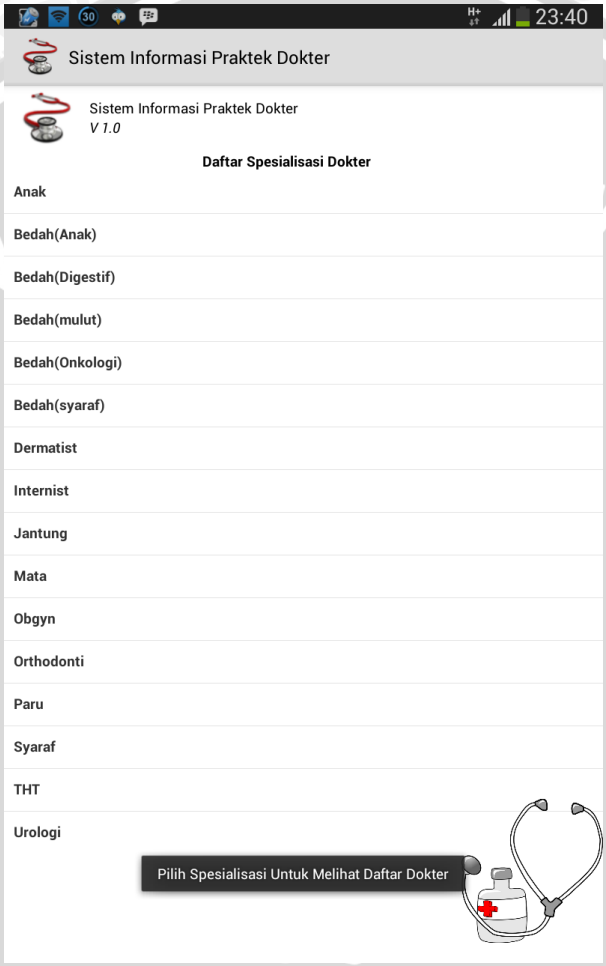
Implemtasi antar muka pada aplikasi praktik dokter ini terdiri dari halaman spesialisasi, halaman daftar dokter, halaman daftar jadwal dokter, dan peta untuk melihat rute

Halaman spesialisasi merupakan implementasi yang berguna untuk menampilkan daftar spesialisasi dokter. Kode halaman dimplentasikan pada activity_main.xml. berikut adalah kode pada halaman spesialisasi ditunjukkan pada kode 5.11

Kode 5. 11 Potongan kode *layout* activity_main.xml

```
<?xml version="1.0" encoding="utf-8"?>
. . .
<ListView
 android:id="@android:id/list"
 android:layout_width="fill_parent"
 android:layout_height="wrap_content"
 android:layout_below="@+id/widget35"
 android:layout_alignParentLeft="true" />
. . .
</RelativeLayout>
```

Halaman spesialisasi berguna untuk menampilkan daftar spesilaisasi dokter yang nantinya akan di pilih pengguna untuk memilih spesialisasi sesuai dengan informasi yang di butuhkan. Tampilan halaman spesialisasi di tunjukkan pada gambar 5.4



Gambar 5. 4 Tampilan halaman spesialisasi

Halaman daftar dokter merupakan implementasi yang berguna untuk menampilkan daftar dokter. Kode halaman dimplentasikan pada activity_main_dokter.xml. berikut adalah kode pada halaman daftar dokter ditunjukkan pada kode 5.12

Kode 5. 12 Potongan kode *layout* activity_main_dokter.xml

```
<?xml version="1.0" encoding="utf-8"?>
. . .
<ListView
 android:id="@android:id/list"
 android:layout_width="fill_parent"
 android:layout_height="wrap_content"
 android:layout_below="@+id/widget35"
 android:layout_alignParentLeft="true" />
. . .
</RelativeLayout>
```

Halaman daftar dokter berguna untuk menampilkan daftar dokter yang nantinya akan di pilih pengguna untuk memilih dokter sesuai dengan informasi yang di butuhkan. Tampilan halaman spesialisasi di tunjukkan pada gambar 5.5

Gambar 5. 5 Tampilan halaman daftar dokter

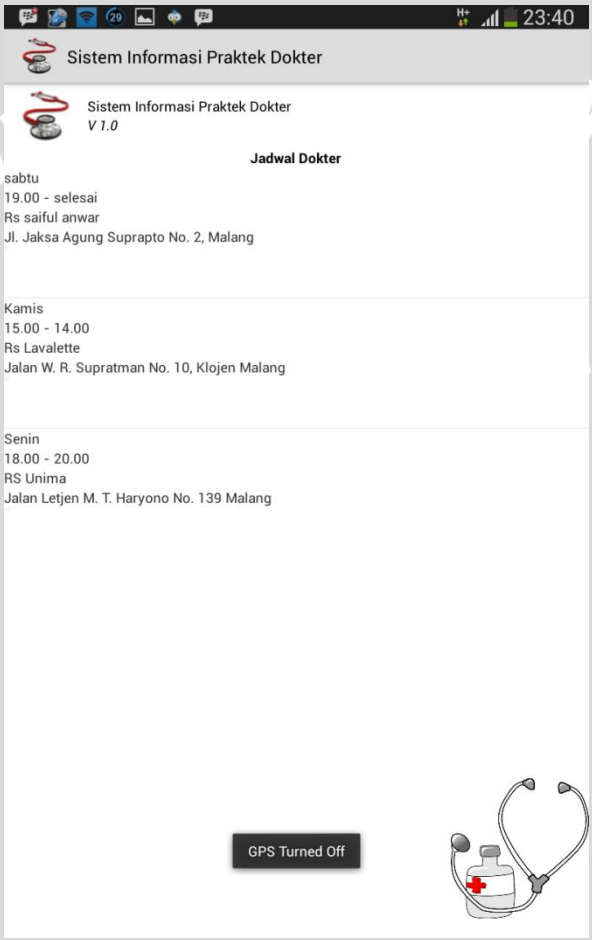
Halaman jadwal dokter merupakan implementasi yang berguna untuk menampilkan jadwal dokter. Kode halaman dimplentasikan pada activity_main_jadwal.xml. berikut adalah kode pada halaman jadwal dokter ditunjukkan pada kode 5.13

Kode 5 13 Implementasi kode *layout* activity_main_jadwal.xml

```
<?xml version="1.0" encoding="utf-8"?>
. . .
<ListView
 android:id="@android:id/list"
 android:layout_width="fill_parent"
 android:layout_height="wrap_content"
 android:layout_below="@+id/widget35"
 android:layout_alignParentLeft="true" />
```

```
. . .
</RelativeLayout>
```

Halaman informasi jadwal dokter berguna untuk menampilkan jadwak dokter yang nantinya akan di pilih pengguna untuk memilih jadwal dokter sesuai dengan informasi yang di butuhkan. Tampilan halaman spesialisasi di tunjukkan pada gambar 5.6



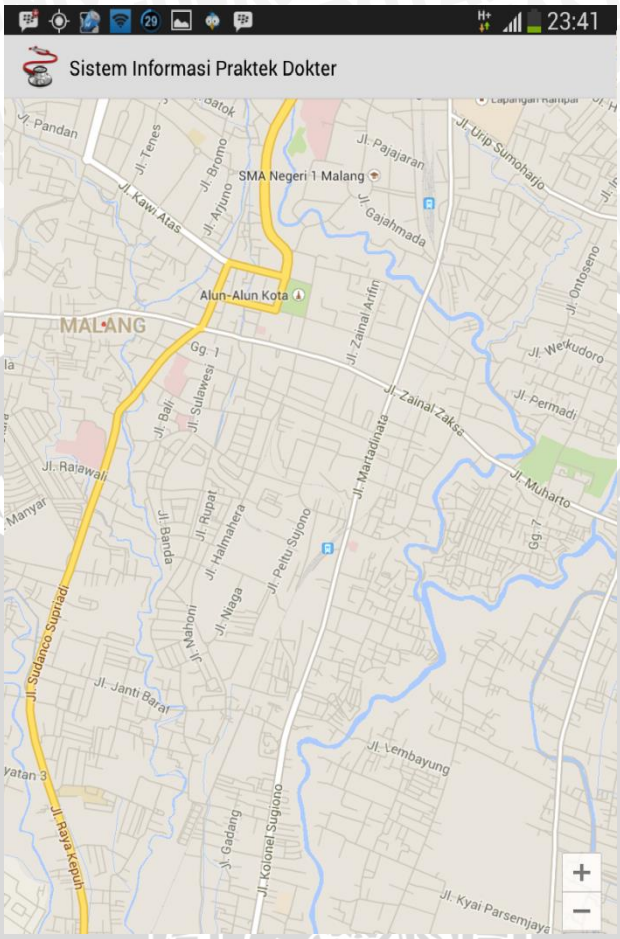
Gambar 5. 6 Tampilan halaman informasi jadwal dokter

Halaman peta merupakan implementasi yang berguna untuk menampilkan rute dari lokasi pengguna ke lokasi dokter. Kode halaman dimplentasikan pada peta.xml. berikut adalah kode pada halaman peta ditunjukkan pada kode 5.14

Kode 5 14 Implementasi kode *layout* peta.xml

```
<RelativeLayout
. . .
 <fragment
  android:id="@+id/map"
  android:layout_width="match_parent"
  android:layout_height="match_parent"
  class="com.google.android.gms.maps.MapFragment" />
. . .
</RelativeLayout>
```

Halaman peta berguna untuk menampilkan rute yang nantinya akan digunakan pengguna untuk navigasi menuju lokasi dokter. Tampilan halaman peta di tunjukkan pada gambar 5.7



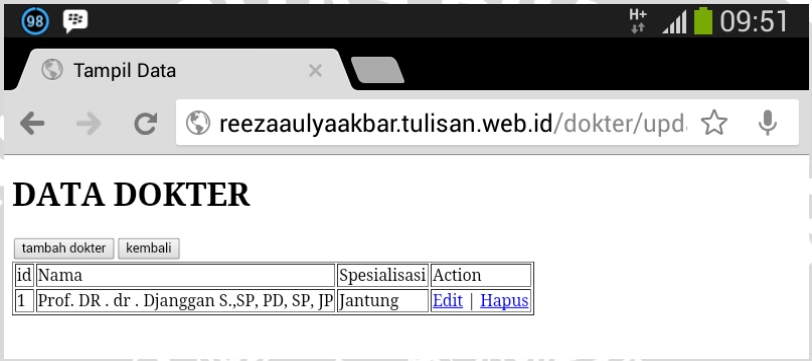
Gambar 5. 7 Tampilan halaman peta

Kode 5. 15 Implementasi kode untuk memanipulasi data dokter

```
<head><title>Tampil Data</title>
<body>
<h1>DATA DOKTER</h1>
<table border='1'>
<tr>
<td>id</td>
<td>Nama</td>
<td>Spesialisasi</td>
<td>Action</td>
</tr>
<input type="button" value="tambah dokter"
onclick="top.location='forminputdata.php'" />
<input type="button" value="kembali"
onclick="top.location='index.php'" />
```

```
. . .
</table>
<br>
</center>
</body>
</html>
```

Halaman update dokter berguna untuk memanipulasi data dokter. Tampilan halaman update dokter di tunjukkan pada gambar 5.8



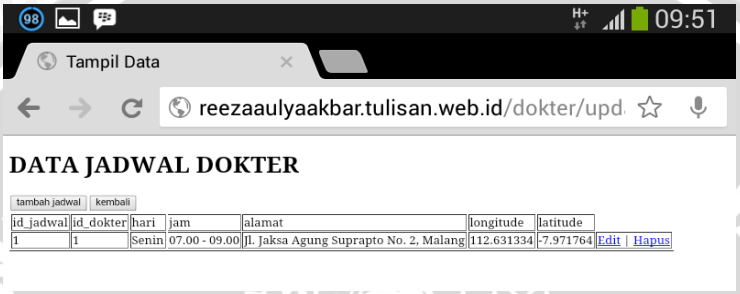
Gambar 5. 8 Tampilan halaman update dokter

Kode 5. 16 Implementasi kode untuk memanipulasi data jadwal dokter

```
<!DOCTYPE html>
<head><title>Tampil Data</title>
<body>
<h1>DATA JADWAL DOKTER</h1>
<table border='1'>
<tr>
<td>id_jadwal</td>
<td>id_dokter</td>
<td>hari</td>
<td>jam</td>
<td>alamat</td>
<td>longitude</td>
<td>latitude</td>
</tr>
<input type="button" value="tambah jadwal"
onclick="top.location='forminputdatajadwal.php'" />
<input type="button" value="kembali"
onclick="top.location='index.php'" />
. . .
</table>
<br>
```

```
</center>
</body>
</html>
```

Halaman update jadwal berguna untuk memanipulasi data jadwal dokter. Tampilan halaman update jadwal di tunjukkan pada gambar 5.9

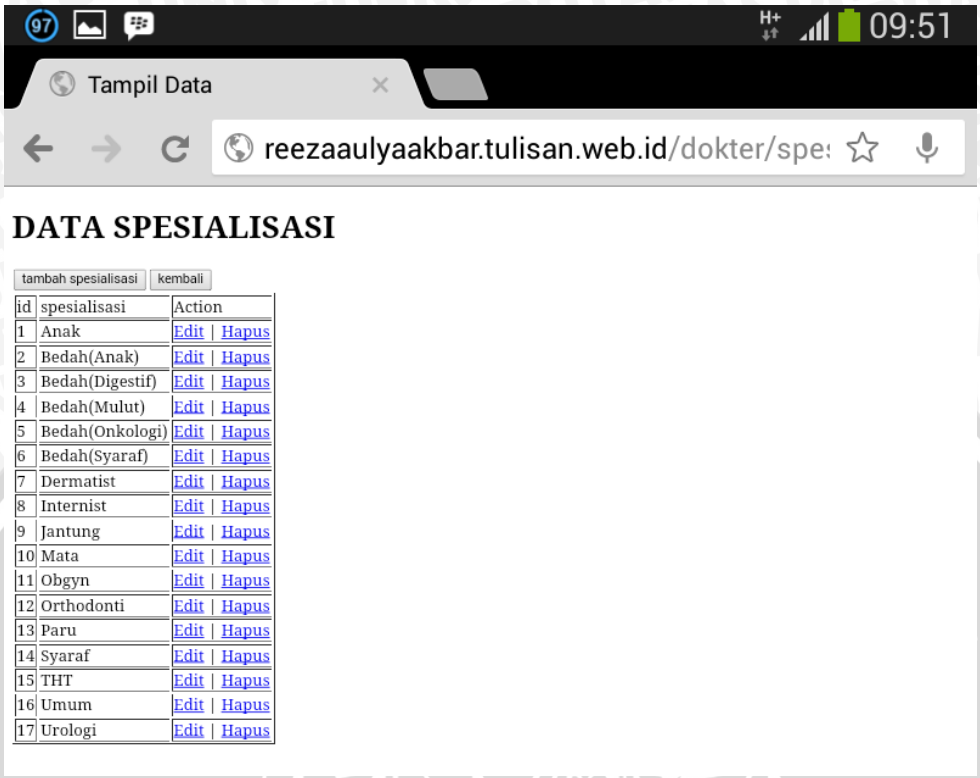


Gambar 5. 9 Tampilan halaman update jadwal

Kode 5 17 Implementasi kode untuk memanipulasi spesialisasi dokter

```
<!DOCTYPE html>
<head><title>Tampil Data</title>
<body>
<h1>DATA SPESIALISASI</h1>
<table border='1'>
<tr>
<td>id</td>
<td>spesialisasi</td>
<td>Action</td>
</tr>
<input       type="button"       value="tambah       spesialisasi"
onclick="top.location='forminputSpesialisasi.php'" />
<input                type="button"                value="kembali"
onclick="top.location='index.php'" />
. . .
</table>
<br>
</center>
</body>
</html>
```

Halaman update spesialis berguna untuk memanipulasi data spesialis dokter. Tampilan halaman update spesialis di tunjukkan pada gambar 5.10



Gambar 5. 10 Tampilan halaman update spesialis

5.6 Pengujian

Pada aplikasi ini yaitu pengujian *validasi* , pengujian non fungsional (*compatibility*) dan pengujian UAT (*User Acceptance Test*) . Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah berjalan seperti yang diharapkan, pengujian validasi ini dilakukan berdasarkan *use case* yang ada pada tahap perancangan. Pengujian UAT dilakukan dengan melibatkan pengguna, Pengujian UAT bertujuan untuk melihat tingkat penerimaan pengguna terhadap aplikasi dengan variable penilaian pencapaian kegunaan *(usefulness)*.

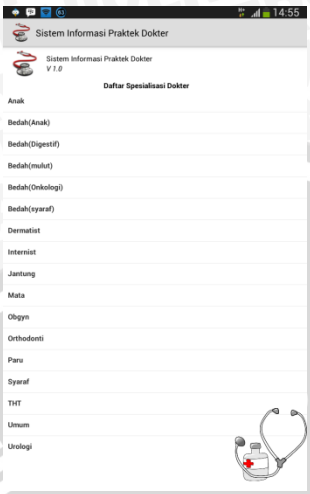
5.6.1 Pengujian Validasi

Pengujian validasi digunakan untuk mengetahui apakah sistem yang dibangun sudah berjalan seperti yang diharapkan. Pengujian validasi ini dilakukan berdasarkan *use case* yang ada pada tahap perancangan. Pengujian ini menggunakan metode pengujian *black-box testing*, karena tidak diperlukan konsentrasi terhadap alur jalannya algoritma program dan lebih ditekankan untuk menemukan kecocokan antara kinerja sistem dengan daftar kebutuhan (*use case*).

1. **Kasus uji lihat daftar dokter**

Tabel 5. 2 Kasus uji lihat daftar dokter

Nama Kasus Uji	Kasus uji lihat daftar dokter
Objek Uji	Kebutuhan Fungsional (F01)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi yang dibuat dapat menampilkan daftar dokter sesuai dengan pilihan pengguna.
Data Masukan	Spesialis Anak
Prosedur Uji	5. Membuka aplikasi praktik dokter 6. Memilih spesialisasi anak
Hasil Yang Diharapkan	1. Aplikasi praktik dokter terbuka 2. Aplikasi menampilkan daftar spesialis dokter anak yaitu : a. dr Masdar, SP .A b. dr Diana Anggraini, Sp.A c. dr Prasodjo, Sp. A
Hasil	1. Aplikasi praktik dokter terbuka

<table>
<tr>
<td></td>
<td>


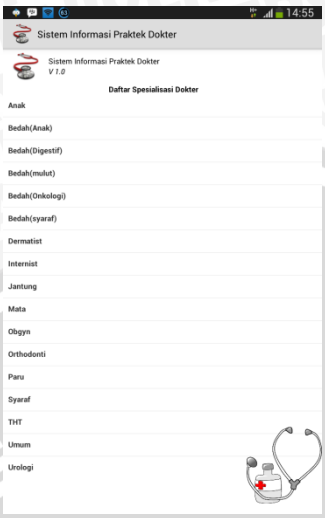
2. Menampilkan daftar spesialis dokter anak


</td>
</tr>
<tr>
<td>Status Validitas</td>
<td>Valid</td>
</tr>
</table>

2. Kasus uji lihat informasi dokter

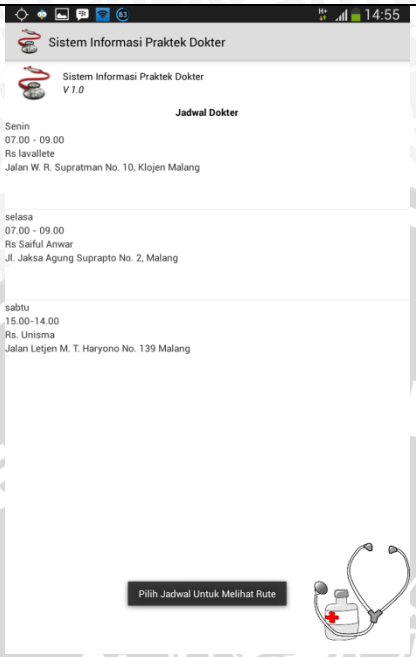
Tabel 5. 3 Kasus uji lihat informasi dokter

Nama Kasus Uji	Kasus uji lihat informasi dokter
Objek Uji	Kebutuhan Fungsional (F02)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat menampilkan informasi dokter sesuai dengan pilihan pengguna.
Data Masukan	dr Masdar, SP .A
Prosedur Uji	7. Membuka aplikasi praktik dokter 8. Memilih spesialisasi anak 9. Memilih dr Masdar, SP .A
Hasil Yang Diharapkan	1. Aplikasi praktik dokter terbuka 2. Aplikasi menampilkan daftar spesialis dokter anak yaitu : a. dr Masdar, SP .A b. dr Diana Anggraini, Sp.A c. dr Prasodjo, Sp. A 3. Aplikasi menampilkan jadwal dr Masdar, SP. A yaitu : a. Senin 07.00 – 09.00 RS lavallete jln W.R Supratman no 10, klojen Malang b. Selasa 07.00 – 09. 00 RS Saiful Anwar jln jaksa agung suprapto no .2, Malang c. Sabtu 15.00 – 14.00 Rs Unisma jln M.T haryono no 139.Malang
Hasil	1. Aplikasi praktik dokter terbuka



2. Menampilkan daftar spesialis dokter anak

3. Menampilkan jadwal dr Masdar, SP .A

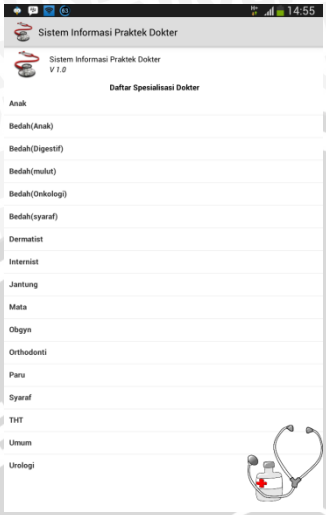
	
Status Validitas	Valid

3. Kasus uji Lihat Rute

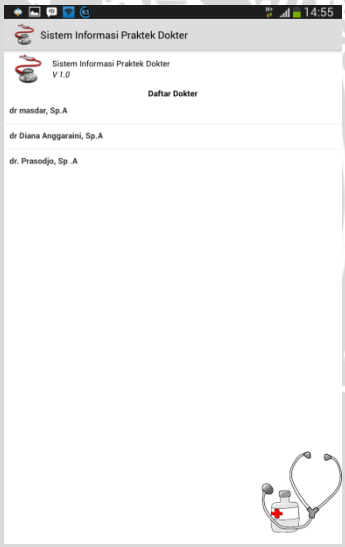
Tabel 5. 4 Kasus uji Lihat Rute

Nama Kasus Uji	Kasus uji Lihat Rute
Objek Uji	Kebutuhan Fungsional (F03)
Tujuan Pengujian	Pengujian dilakukan untuk memastikan bahwa aplikasi dapat menampilkan rute dari lokasi pengguna menuju tempat lokasi dokter yang dipilh pengguna.
Data Masukan	Koordiat lokasi pengguna di universitas brawijaya (longitude : 112.615892 dan latitude : -7.955732.) dan lokasi dokter di RS saiful anwar (longitude : 112.63143 dan latitude : -7.971786)

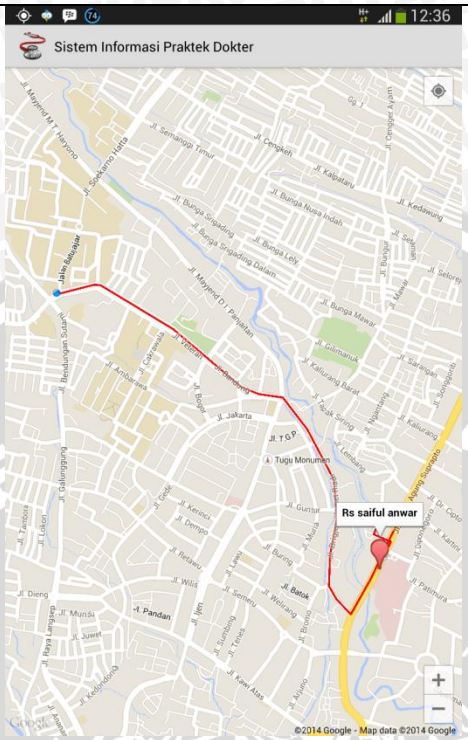
Prosedur Uji	1. Membuka aplikasi praktik dokter 2. Memilih spesialisasi anak 3. Memilih dr Masdar, SP .A 4. Memilih Selasa 07.00 – 09. 00 RS Saiful Anwar jln jaksa agung suprapto no .2, Malang
Hasil Yang Diharapkan	1. Aplikasi praktik dokter terbuka 2. Aplikasi menampilkan daftar spesialis dokter anak yaitu : a. dr Masdar, SP .A b. dr Diana Anggraini, Sp.A c. dr Prasodjo, Sp. A 3. Aplikasi menampilkan jadwal dr Masdar, SP. A yaitu : a. Senin 07.00 – 09.00 RS lavallete jln W.R Supratman no 10, klojen Malang b. Selasa 07.00 – 09. 00 RS Saiful Anwar jln jaksa agung suprapto no .2, Malang c. Sabtu 15.00 – 14.00 Rs Unisma jln M.T haryono no 139.Malang 4. Aplikasi menampilkan rute dari universitas brawijaya menuju RS saiful anwar
Hasil	1. Aplikasi praktik dokter terbuka



2. Menampilkan daftar spesialis dokter anak



	3. Menampilkan jadwal dr Masdar, SP .A  4. Aplikasi menampilkan rute dari universitas brawijaya menuju RS saiful anwar

	
Status Validitas	Valid

5.6.2 Pengujian UAT (*User Acceptance Test*)

. Pengujian UAT dilakukan dengan melibatkan pengguna, Pengujian UAT bertujuan untuk melihat tingkat penerimaan pengguna terhadap aplikasi dengan variable penilaian pencapaian kegunaan *(usefulness)*. Teknik kuisioner dilakukan untuk mendukung pengujian ini. Sampel yang digunakan dalam pengujian ini sebanyak 20 sampel secara acak. Pertanyaan yang diajukan dalam kuisioner ini adalah apakah sistem telah membantu pengguna dalam mendapatkan informasi yang diinginkan. Pernyataan-pernyataan tersebut akan disertakan sebagai lampiran. Hasil dari pengujian UAT (*User Acceptance Test*) ditunjukkan pada Tabel 5.5

Tabel 5. 5 Hasil Pengujian UAT (*User Acceptance Test*)

No	Pernyataan	STS	TS	N	S	SS	Total
1	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi daftar dokter	0	1	6	9	4	20
2	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi jadwal dokter	0	2	3	10	5	20
3	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan rute menuju lokasi praktik dokter	0	0	6	9	5	20
4	Saya merasa aplikasi ini sangat membantu saya ketika mengalami masalah tentang informasi praktik dokter	0	3	5	7	5	20
5	Saya akan menggunakan aplikasi ini ketika saya ingin mencari informasi praktik dokter	0	5	2	11	2	20

Keterangan:

STS : Sangat Tidak Setuju **S** : Setuju

TS : Tidak Setuju **SS** : Sangat Setuju

N : Netral

5.7 Analisis Hasil Pengujian

Proses analisis terhadap hasil pengujian dilakukan untuk mendapatkan kesimpulan dari hasil pengujian aplikasi sistem informasi praktik dokter yang telah dilakukan. Proses analisis mengacu pada hasil pengujian yang didapatkan. Analisis dilakukan terhadap hasil pengujian di setiap tahap pengujian. Proses analisis yang dilakukan meliputi analisis hasil pengujian validasi .

5.7.1 Analisis Hasil Pengujian Validasi

Proses analisis terhadap hasil pengujian validasi dilakukan dengan melihat kecocokan antara hasil kinerja sistem dengan daftar kebutuhan. Berdasarkan hasil pengujian validasi dapat disimpulkan bahwa implementasi dan fungsionalitas aplikasi sistem informasi praktik dokter telah memenuhi kebutuhan yang dijabarkan pada tahap analisis kebutuhan. Status validitas yang dihasilkan ialah Valid.

5.7.2 Analisis Hasil Pengujian UAT (*User Acceptance Test*)

Pengujian UAT dilakukan dengan melibatkan pengguna, Pengujian UAT bertujuan untuk melihat tingkat penerimaan pengguna terhadap aplikasi dengan variable penilaian pencapaian kegunaan *(usefulness)*. Untuk mendapatkan nilai dari hasil pencapaian maka digunakan skala likert sebagai skala penilaian pendapat pengguna tentang kegunaan aplikasi. Interpretasi skor Likert atau persentase dari setiap skor Likert ditunjukkan pada Tabel 5.6. Hasil perhitungan index persentase dari pernyataan ditunjukkan pada Tabel 5.7 dan hasil status pengujian UAT (*User Acceptance Test*) ditunjukkan pada Tabel 5.8

Tabel 5. 6 Interpretasi Skor Likert

Skor Likert	Interpretasi skor dengan interval = 20	Pilihan
1	0% - 19.99%	Sangat Tidak Setuju
2	20% - 39.99%	Tidak Setuju
3	40% - 59.99%	Netral
4	60% - 79.99%	Setuju
5	80% - 100%	Sangat Setuju

Keterangan:

Interval = 20 didapatkan dari pembagian nilai 100 dengan jumlah skor Likert

Tabel 5. 7 Hasil Persentase keseluruhan

No	Pernyataan	STS	TS	N	S	SS	Total Skor	Index (%)
1	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi daftar dokter	0	1	6	9	4	79	79%
2	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi jadwal dokter	0	2	3	10	5	75	75%
3	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan rute menuju lokasi praktik dokter	0	0	6	9	5	79	79%
4	Saya merasa aplikasi ini sangat membantu saya ketika mengalami masalah tentang informasi praktik dokter	0	3	5	7	5	74	74%
5	Saya akan menggunakan aplikasi ini ketika saya ingin mencari informasi praktik dokter	0	5	2	11	2	70	70%

Keterangan:

STS : Sangat Tidak Setuju **S** : Setuju

TS : Tidak Setuju **SS** : Sangat Setuju

N : Netral

Total Skor = S_{STS} x 1 + S_{TS} x 2 + S_N x 3 + S_S x 4 + S_{SS} x 5 …… 1)

Index (%) = Total Skor / Y x 100 …… 2)

Y : Skor Likert tertinggi x Jumlah audien

Berdasarkan tabel 5.8 pada pertanyaan 1 didapatkan hasil presentase keseluruhan sebesar 79%. Karena banyaknya audien yang merasa sangat terbantu dengan adanya aplikasi ini untuk mendapatkan informasi daftar dokter. Pada pertanyaan 2 didapatkan hasil presentase keseluruhan sebesar 75%. Karena banyaknya audien yang merasa sangat terbantu dengan adanya aplikasi ini untuk mendapatkan informasi jadwal dokter. Pada pertanyaan 3 didapatkan hasil presentase keseluruhan sebesar 79%. Karena banyaknya audien yang merasa sangat terbantu dengan adanya aplikasi ini untuk mendapatkan informasi rute menuju lokasi praktik dokter. Pada pertanyaan 4 didapatkan hasil presentase keseluruhan sebesar 74%. Karena banyaknya audien yang merasa sangat terbantu dengan adanya aplikasi ini ketika mengalami masalah tentang informasi praktik dokter. Pada pertanyaan 5 didapatkan hasil presentase keseluruhan sebesar 70%. Karena banyaknya audien yang merasa akan menggunakan aplikasi ini ketika ingin mencari informasi praktik dokter.

Tabel 5. 8 Status pengujian UAT (*User Acceptance Test*)

Asepek Penilaian	Persentase (%)	Status
Dapat membantu untuk mendapatkan informasi daftar dokter	79%	Accepted
Dapat membantu untuk mendapatkan informasi jadwal dokter	75%	Accepted
Dapat membantu untuk mendapatkan rute menuju lokasi praktik dokter	79%	Accepted
Dangat membantu ketika mengalami masalah tentang informasi praktik dokter	74%	Accepted
Akan menggunakan aplikasi ini ketika ingin mencari informasi praktik dokter	70%	Accepted

Perhitungan persentase rata-rata dari semua penilaian

= jumlah total presentase (%) / jumlah pernyataan

= 377/ 5 = 75.4% = 75%

Kesimpulan yang dapat diambil dari hasil perhitungan persentase keseluruhan jawaban pada setiap pernyataan dan status pengujian UAT (*User Acceptance Test*) ialah aplikasi dapat membantu pengguna dengan persentase rata-rata penilaian sebesar 75%.

BAB VI
PENUTUP

6.1 Kesimpulan

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

1. Analisis dapat dilakukan dengan menggunakan metode *object-oriented analysis* untuk mendapatkan kebutuhan aplikasi yang akan di bangun.
2. Perancangan dapat dilakukan dengan menggunakan bahasa pemodelan UML (*Unified Modeling Language*) sebagai acuan di tahap implementasi.
3. Implementasi dapat dilakukan menggunakan bahasa pemrograman Java pada Android sesuai dengan tahap perancangan.
4. Berdasarkan hasil pengujian validasi sistem telah memenuhi kebutuhan fungsional yang di spesifikasikan.
5. Berdasarkan hasil pengujian UAT (*User Acceptance Test*) dengan menggunakan kuisioner didapatkan bahwa aplikasi praktik dokter dapat membantu pengguna dalam mendapatkan daftar dokter, jadwal dokter dan rute menuju lokasi praktek dokter dengan nilai rata-rata 75% yang artinya aplikasi dapat diterima oleh pengguna.

6.2 Saran

1. Untuk pengembang versi lebih lanjut sebaiknya tidak hanya diimplementasikan pada *platform* Android.
2. Untuk pengembangan versi lebih lanjut sebaiknya tidak hanya menggunakan pencarian rute dengan fungsi Google tetapi menggunakan algoritma untuk penentuan rute terdekat
3. Untuk pengembangan versi lebih lanjut sebaiknya rute di lengkapi dengan informasi jarak dan estimasi waktu untuk mencapai lokasi tujuan.

DAFTAR PUSTAKA

[AFD-13] Anton Firdaus, Dokumentasi Google Maps API, https://developers.google.com/maps/documentation/android/, Mei 2013.

[AHA-12] Akbarul Huda, Arif. 2012. 24 Jam Pintar Pemprograman Android. PT. Andi Yogyakarta. Juni 2013

[CHO-07] Chow, Shu-Wai., 2007, "PHP Web 2.0 Mashup Project", PACKT Publisihing, Birmingham.

[DAI-12] Daigneau, Robert. 2012. "Service Design Patterns: Fundamental Design Solutions for SOAP/WSDL and RESTful Web Services". Boston: Pearson Education, Inc

[FMN-13] M. Fathir Natsir. "Cara menghitung Skala Likert". URL: http://fathirphoto.wordpress.com/2013/09/24/cara-menghitung-skala-likert/, diakses tanggal 22 Mei 2014.

[JKM-13] Joko, Munif.2013. Dokumentasi Android http://developer.android.com Mei 2013.

[JGN-11] Jeevan, Gnana. 2011. *SOAP (Simple Object Access Protocol).* MAMCE, New Delhi.

[JSO-13] JSON. "Pengenalan JSON". URL : http://json.org, diakses tanggal 16 Desember 2013.

[MML-09] Murphy M.L.2009.*Beginning Android*. Apress. New York. Mei 2013

R Radifan .2011.Perancangan dan Pembuatan Sistem Informasi

[NUR-13] Nurochman, Yusuf Mufti, 2013, Rancang Bangun Sistem Pemanggil Darurat pada situasi perampokan berbasis android.

[PEM-09] Presiden Republik Indonesia, 2009, UNDANG-UNDANG REPUBLIK INDONESIA NOMOR 36 TAHUN 2009

[PHP-14] PHP team. 2014. JSON Book Manual. URL : http://php.net/manual, diakses tanggal 6 Januari 2014

[RRD-11] Lokasi Friend Finder Berbasis GPS pada Sistem Operasi Android.Skripsi S-1.Institut Teknologi Sepuluh November. Surabaya. Mei 2013.

[SSA-13] Sena , Samuel Aji. 2013. *Perancangan dan Pembuatan Application Programing Interface Server untuk Arduino.*Fakultas Teknik, Universitas Brawijaya, Malang.

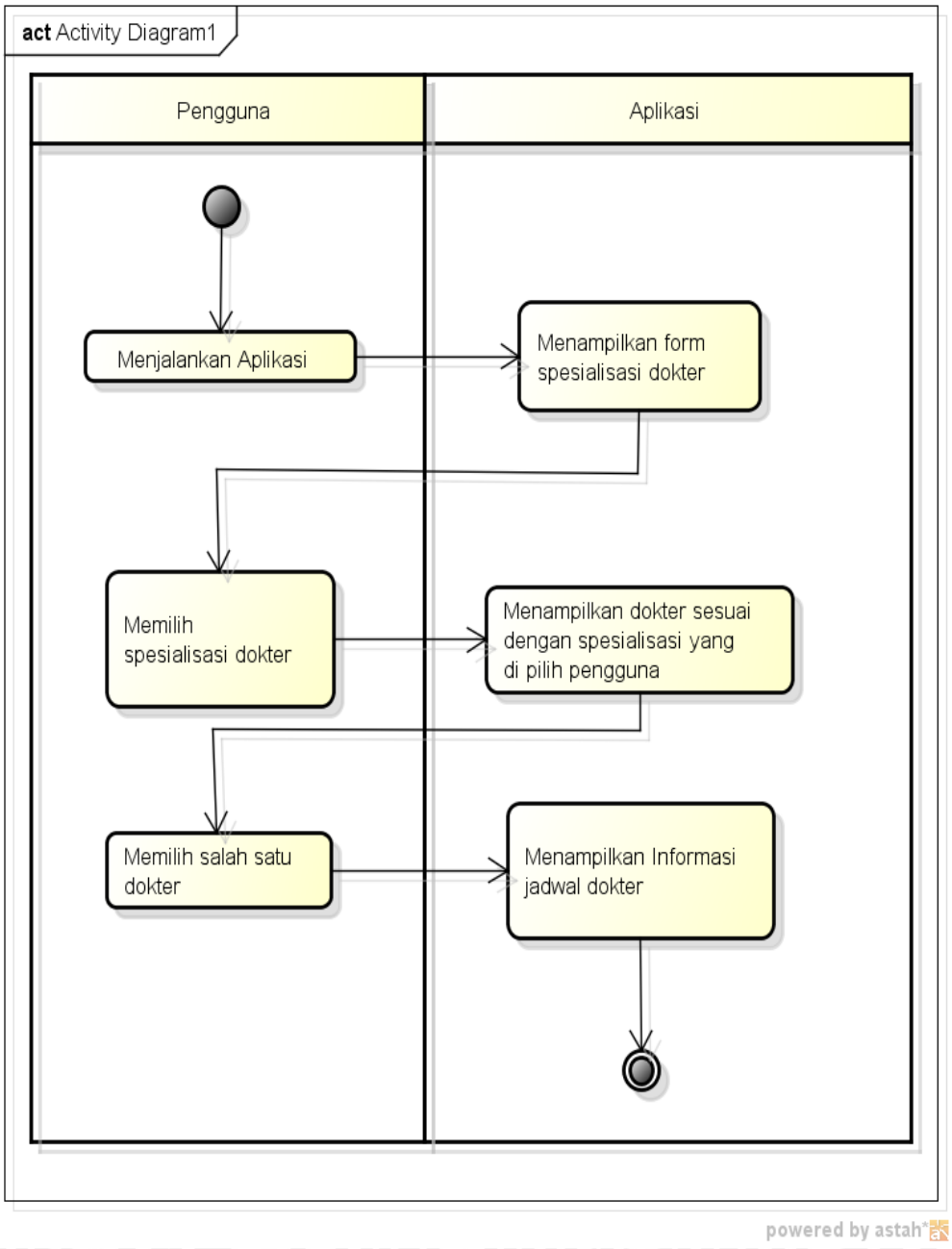
[SUG-10] Sugiyono. 2010. "Metode Penelitian Bisnis". Bandung: ALFABETA.

[SUT-12] Sutanta, Edhy et.al. 2012. "Kebutuhan Web Service untuk Sinkronisasi Data Antar Sistem Informasi dalam E-Gov di Pemkab Bantul Yogyakarta". JURTIK - STMIK BANDUNG (edisi Mei 2012).

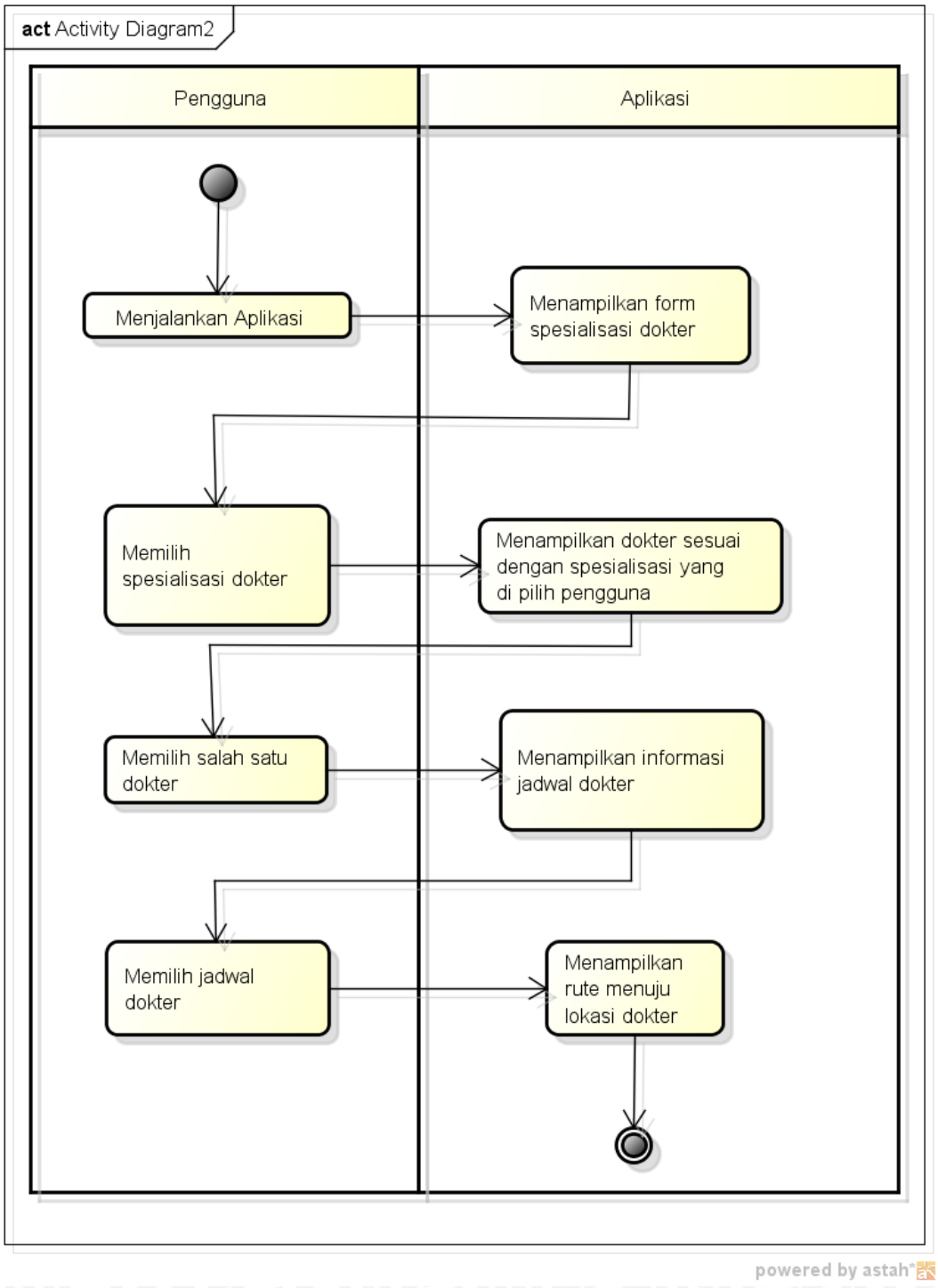
[WIY-12] Wiyono, Didiek S. Dan Wijayanto, Ardhi, 2012, "Impementasi Rest Web Service Dengan Menggunakan JSON Pada Aplikasi Mobile Enterprise Resource Planning", Performa, Vol. 11, No. 2.

LAMPIRAN

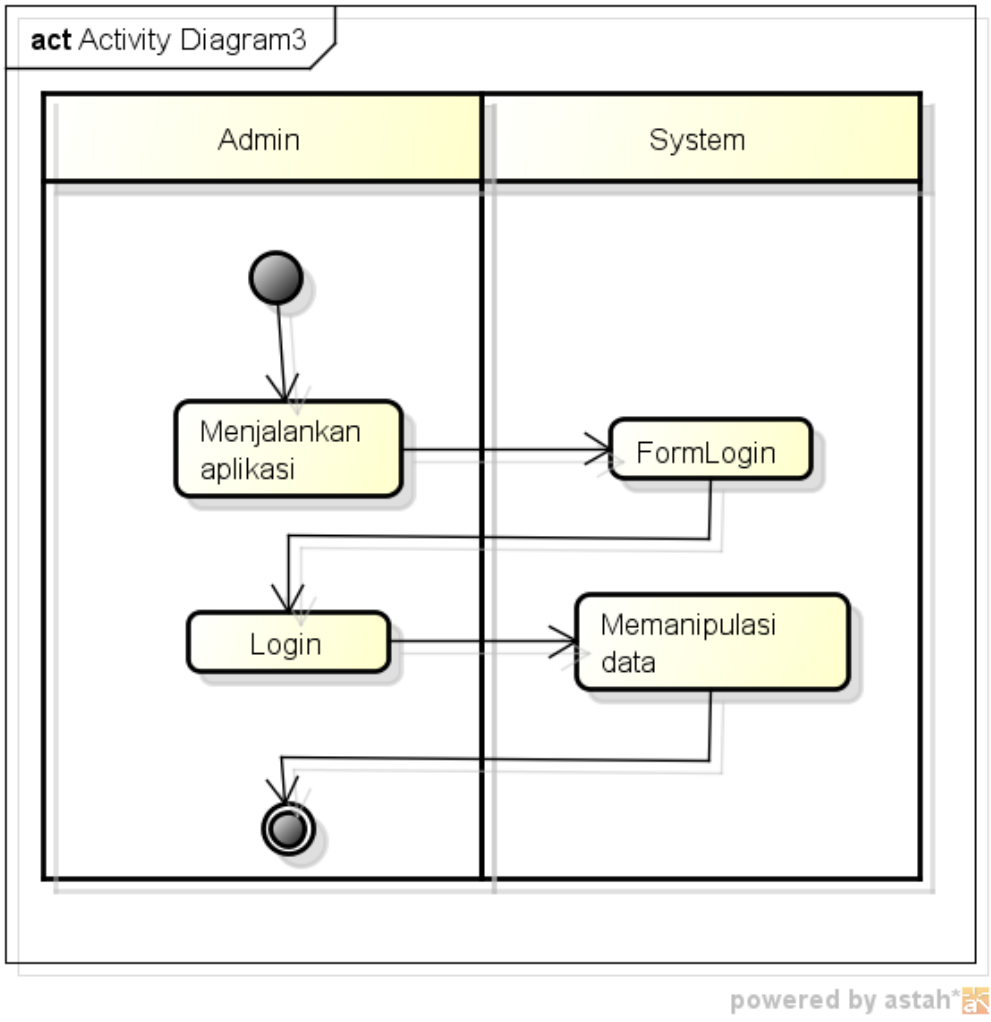
Lampiran 1 : Aktifitas diagram



Gambar L.1. 1 Aktifitas diagram melihat informasi jadwal dokter

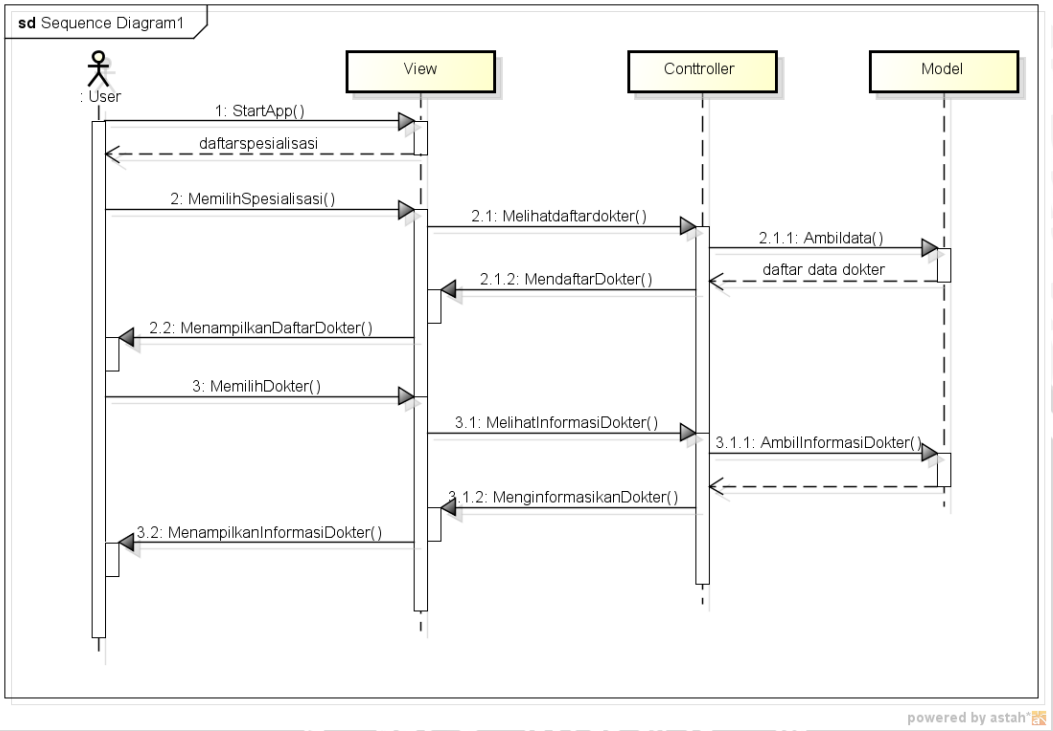


Gambar L.1. 2 Aktifitas diagram melihat rute menuju lokasi dokter

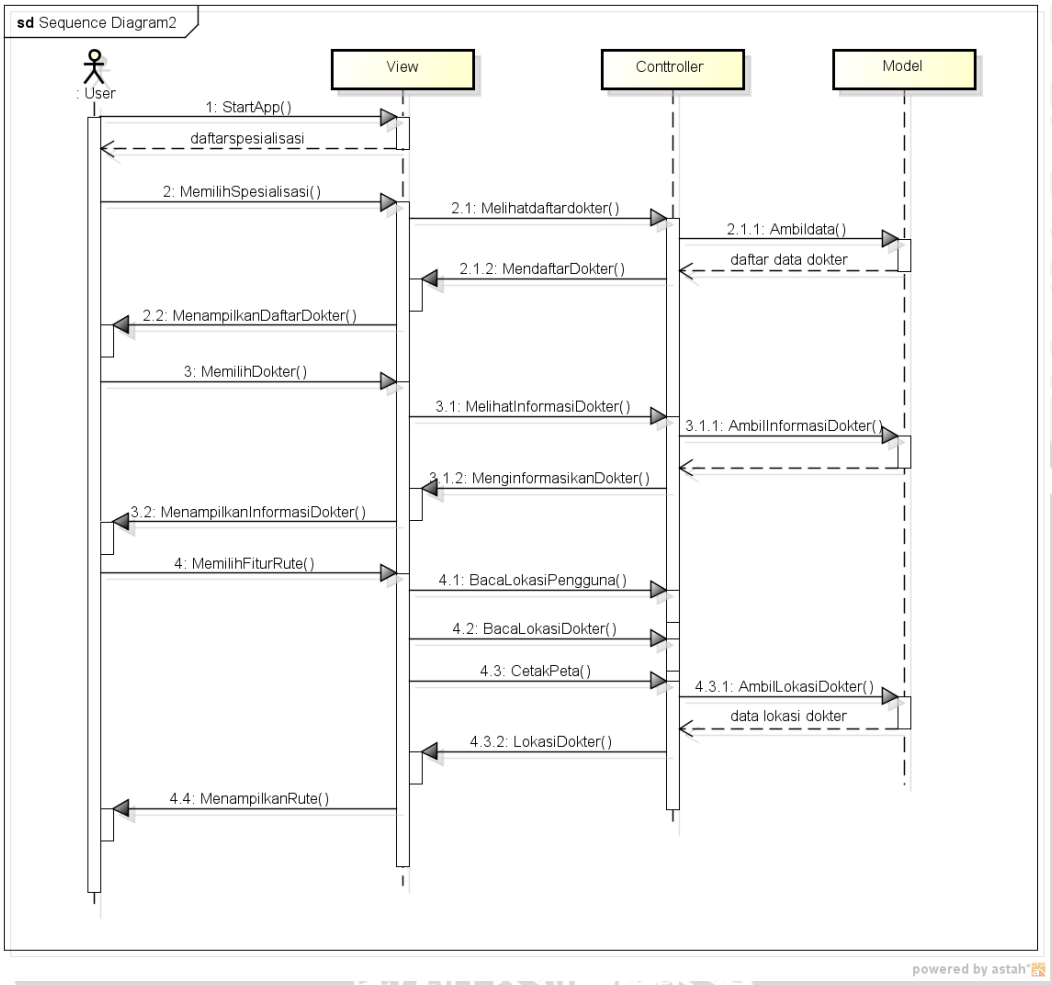


Gambar L.1. 3 Aktifitas diagram melihat rute menuju lokasi dokter

Lampiran 2: Sequence diagram

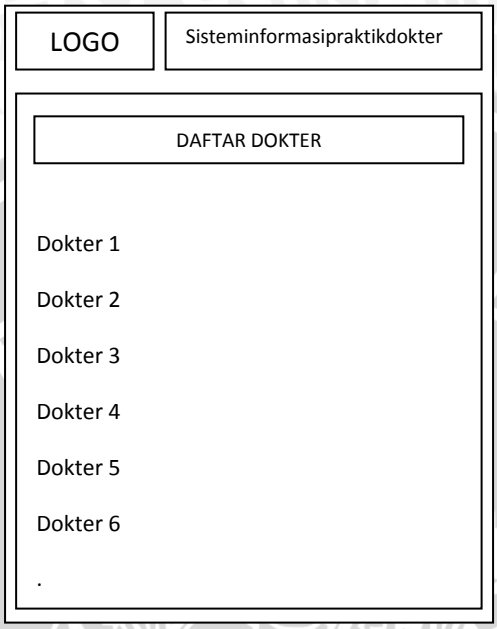


Gambar L.2.1 Sequence diagram untuk melihat informasi jadwal dokter

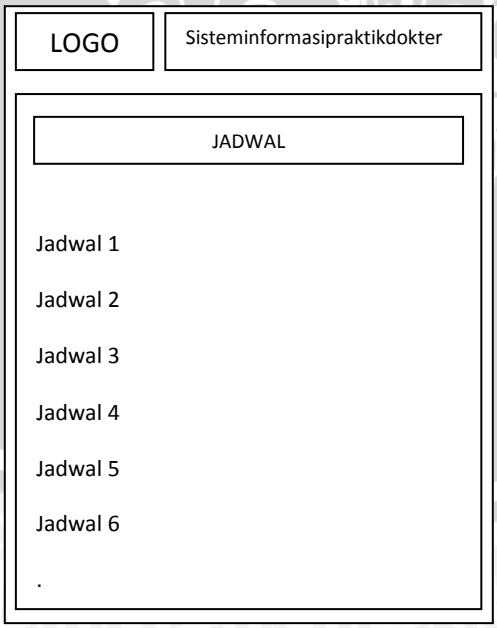


Gambar L.2. 2 Sequence diagram untuk melihat rute menuju dokter

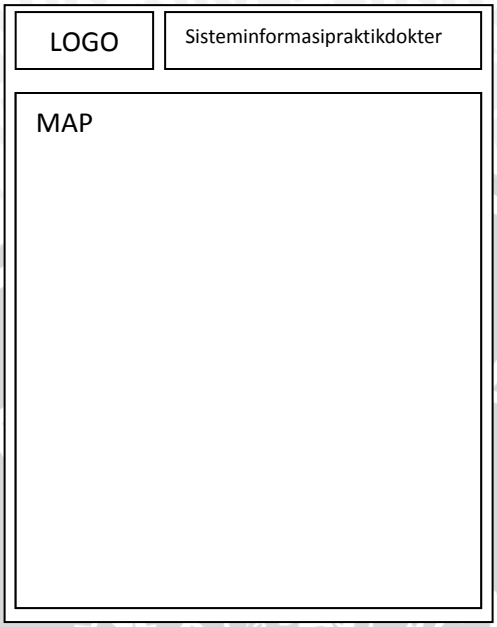
Lampiran 3: Rancangan Antar Muka



Gambar L.3.1 Rancangan antar muka untuk melihat daftar dokter



Gambar L.3. 2 Rancangan antar muka untuk melihat jadwal dokter



Gambar L.3. 3 Rancangan antar muka peta

Lampiran 4: kuisioner

Nama : .....

Jenis Kelamin* : Laki-laki / Perempuan *(*coret salah satu)*

No. Identitas : .....

(KTP/SIM/NIM) .....

.....

Alamat : .....

.....

No. HP : .....

-kuesioner-

Berilah tanda silang (X) pada jawaban yang menurut Anda paling sesuai!

1 = Sangat Tidak Setuju **2 = Tidak Setuju**

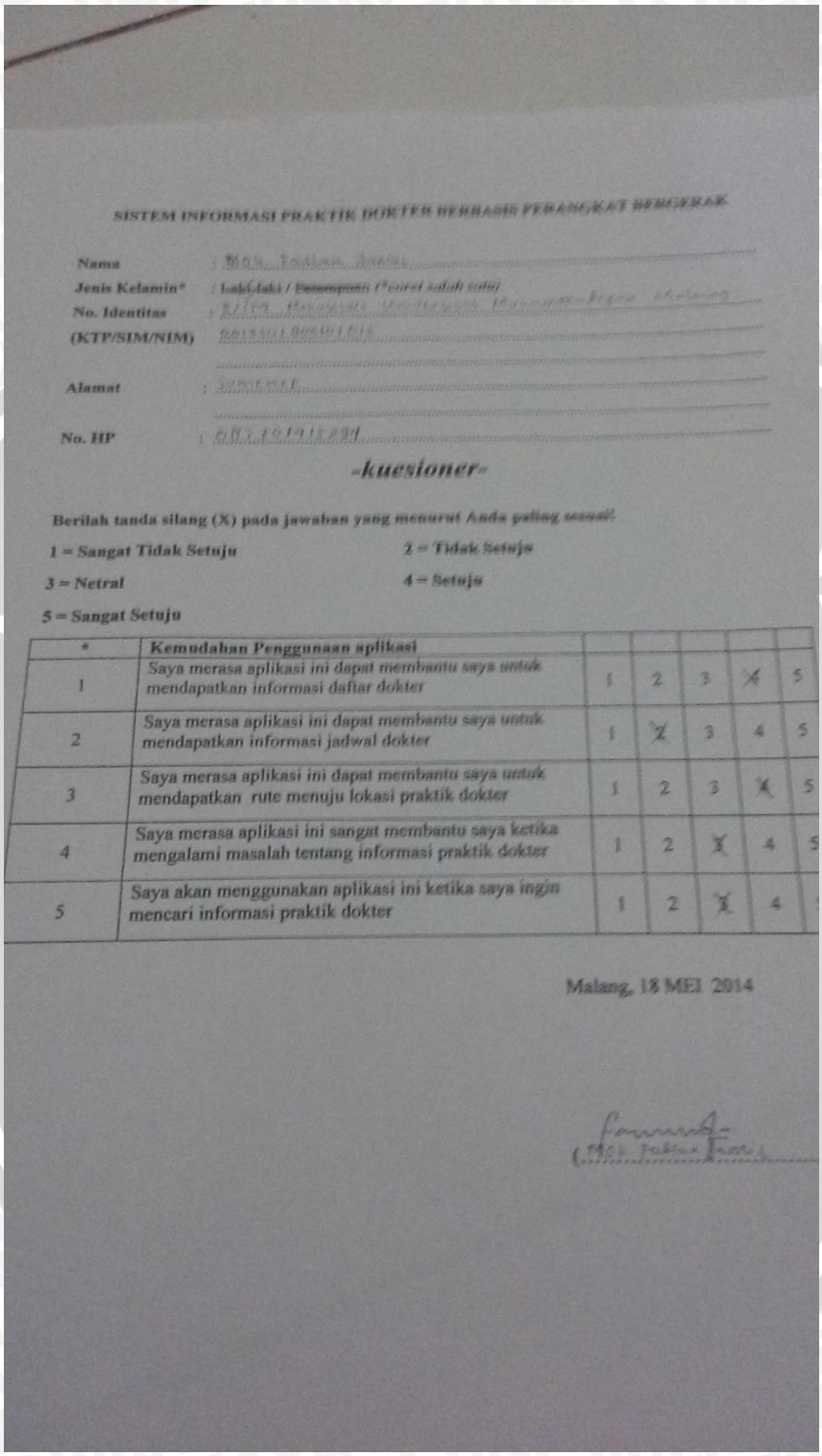
3 = Netral **4 = Setuju**

5 = Sangat Setuju

*	**Kemudahan Penggunaan aplikasi**					
1	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi daftar dokter	1	2	3	4	5
2	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi jadwal dokter	1	2	3	4	5
3	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan rute menuju lokasi praktik dokter	1	2	3	4	5
4	Saya merasa aplikasi ini sangat membantu saya ketika mengalami masalah tentang informasi praktik dokter	1	2	3	4	5
5	Saya akan menggunakan aplikasi ini ketika saya ingin mencari informasi praktik dokter	1	2	3	4	5

Malang, 18 MEI 2014

(.....)



SISTEM INFORMASI PRAKTIK DOKTER BERBASIS PERANGKAT BERGERAK

Nama : [illegible]
Jenis Kelamin* : Laki-laki / ~~Perempuan~~ (*coret salah satu)
No. Identitas (KTP/SIM/NIM) : [illegible]
Alamat : [illegible]
No. HP : [illegible]

-kuesioner-

Berilah tanda silang (X) pada jawaban yang menurut Anda paling sesuai:

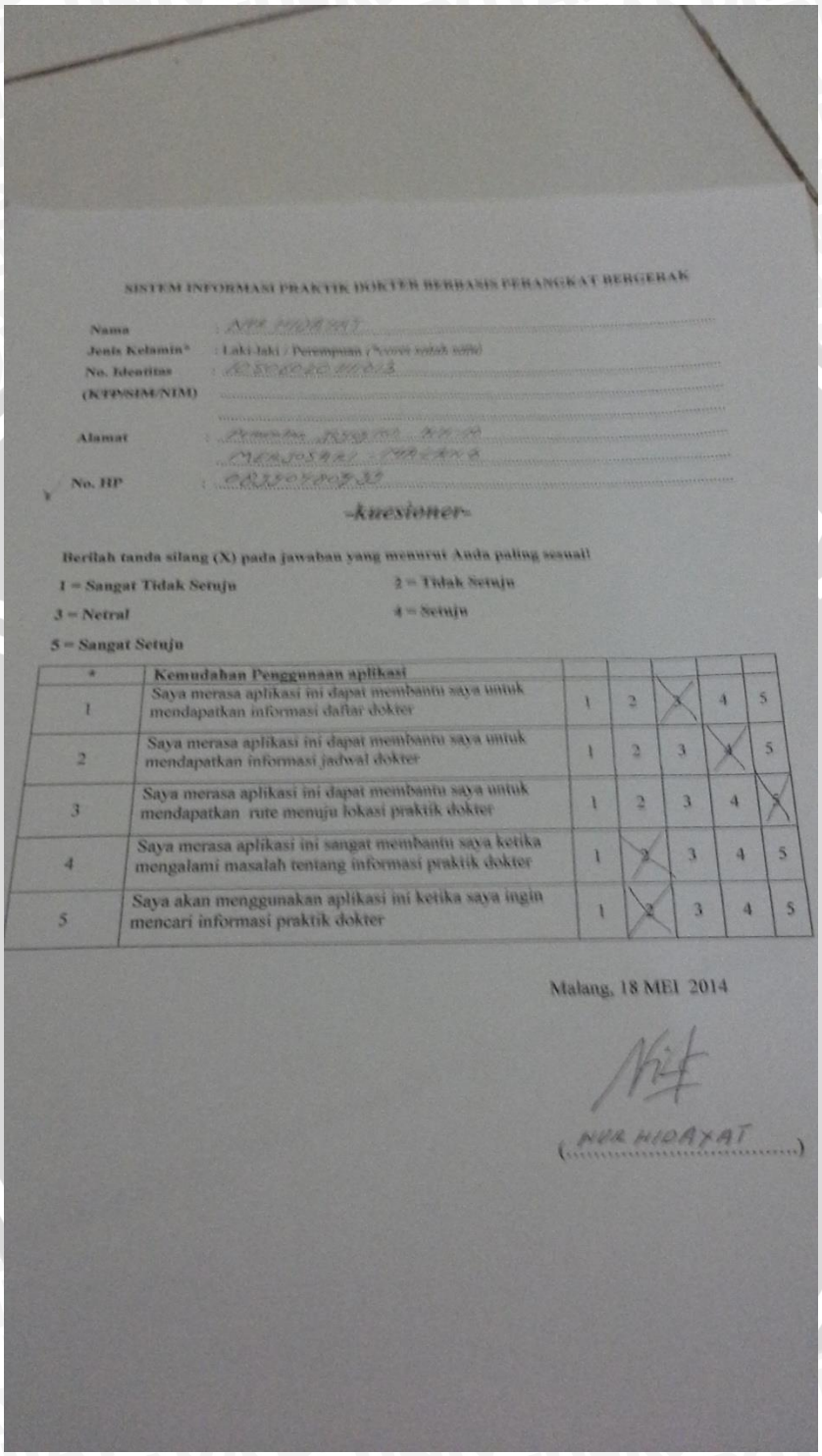
1 = Sangat Tidak Setuju
2 = Tidak Setuju
3 = Netral
4 = Setuju
5 = Sangat Setuju

*	Kemudahan Penggunaan aplikasi					
1	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi daftar dokter	1	2	3	X	5
2	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi jadwal dokter	1	X	3	4	5
3	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan rute menuju lokasi praktik dokter	1	2	3	X	5
4	Saya merasa aplikasi ini sangat membantu saya ketika mengalami masalah tentang informasi praktik dokter	1	2	X	4	5
5	Saya akan menggunakan aplikasi ini ketika saya ingin mencari informasi praktik dokter	1	2	X	4	

Malang, 18 MEI 2014

([illegible])

Gambar L.4.1 Hasil kuisioner



SISTEM INFORMASI PRAKTIK DOKTER BERBASIS PERANGKAT BERGERAK

Nama : NUR HIDAYAT
Jenis Kelamin* : Laki-laki / Perempuan (*coret salah satu)
No. Identitas : 0530040003
(KTP/SIM/NIM)
Alamat : Perumahan Joyo Mas Asri 20
Merjosari - Malang
No. HP : 085707078535

-kuesioner-

Berilah tanda silang (X) pada jawaban yang menurut Anda paling sesuai!

1 = Sangat Tidak Setuju 2 = Tidak Setuju
3 = Netral 4 = Setuju
5 = Sangat Setuju

*	Kemudahan Penggunaan aplikasi					
1	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi daftar dokter	1	2	X	4	5
2	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan informasi jadwal dokter	1	2	3	X	5
3	Saya merasa aplikasi ini dapat membantu saya untuk mendapatkan rute menuju lokasi praktik dokter	1	2	3	4	X
4	Saya merasa aplikasi ini sangat membantu saya ketika mengalami masalah tentang informasi praktik dokter	1	X	3	4	5
5	Saya akan menggunakan aplikasi ini ketika saya ingin mencari informasi praktik dokter	1	X	3	4	5

Malang, 18 MEI 2014

(NUR HIDAYAT)

Gambar L.4.2 Hasil Kuisioner

Tabel L.4. 1 Rekapitulasi hasil kuisioner

No	Nama	JK	1	2	3	4	5
1	Nur Hidayat	L	3	4	5	2	2
2	Moh Fadlan Jamil	L	4	2	4	3	3
3	Miftah Nurul alim	L	3	4	5	4	4
4	Rendi Danifi	L	5	3	3	5	2
5	Moh Fharhan Dhoyfur	L	4	4	4	2	5
6	Abd Rahman	L	3	2	4	4	4
7	Virzannida Abwa	P	5	5	3	3	2
8	Ariestiowati Wulandari	P	4	4	4	5	4
9	Andreyanto	L	3	4	5	4	3
10	Findr Adi As'Ari	L	4	5	4	4	4
11	Mahfud Junaidi	L	3	4	3	3	2
12	R Firman Agung	L	4	5	4	5	4
13	Nur Layla	P	3	3	4	4	4
14	Nur Qomari	P	4	4	5	2	4
15	Anwar Pratama Putra	L	2	4	3	5	4
16	Sitti Sholeha	P	4	4	5	4	4
17	Risma Nur Aini	P	5	5	4	3	2
18	Dedy Ahmadi	L	4	4	3	5	5
19	Panji Putra	L	5	5	4	4	4
20	Aulia Afafun Nisa	P	4	3	3	3	4