Implementasi Teknologi Web Services Pada Aplikasi Pencarian Taksi

Debby E. Sondakh*[1], Stenly R. Pungus[2], Prizilia Runtukahu[3], Rizky Saroinsong[3]
Universitas Klabat; Jl. A. Mononutu – Airmadidi – Minahasa Utara, 0431-891035
Teknik informatika, Fakultas Ilmu Komputer
e-mail: ***[1]debby.sondakh@unklab.ac.id**, [2]stenly.pungus@unklab.ac.id,
[3]prayziliaruntukahu@gmail.com, [4]rizkysaroinsong@gmail.com

Abstrak

Taksi merupakan layanan transportasi yang umum ditemui di daerah perkotaan. Untuk menggunakan layanan taksi pengguna dapat menunggu taksi yang lewat atau memesan melalui telepon. Kendala yang terjadi dari metode ini adalah panggilan telepon tidak tersambung karena operator sedang melayani konsumen lain, posisi taksi yang berada jauh dari konsumen, atau konsumen tidak mengetahui posisinya saat ini dengan tepat. Ketika menunggu taksi di jalan, pencarian taksi dapat memakan waktu lama. Makalah ini memaparkan tentang pengembangan aplikasi pencarian taksi berbasis teknologi Android, menggunakan model proses Rational Unified Process dari pendekatan rekayasa perangkat lunak. Implementasi teknologi web service menghasilkan aplikasi pencarian taksi terdekat yang dapat mengintegrasikan aplikasi berbasis web dengan platform berbeda, dengan aplikasi mobile berbasis sistem operasi Android. Pada aplikasi mobile, pengguna dapat mencari taksi terdekat sesuai perusahaan terdaftar yang dilacak menggunakan GPS dan melihat posisi taksi pada peta dari Google Maps Application Programming Interface, melihat informasi taksi, serta memesan taksi atau membatalkan pemesanan.

Kata kunci- Aplikasi Pencarian Taksi, Web Services, Android

Abstract

Taxi is a common transportation services in urban areas. In order to use taxi services customers may wait for a taxicab passing by or make a request by phone. Constraints that may arise from this method are unconnected phone call when the operator is currently serving others, far away taxicab position from the customer, customer knows not about his current position. Moreover, cab searching may take some time. This paper presents the development of Android-based taxicab searching application, using Rational Unified Process model. Implementation of web services technology resulting a taxicab search application that could integrate different platforms web-based applications, with Android-based mobile application. Using mobile application, user can search the nearest cabs tracked by GPS as well as see their positions on a map using Google Maps Application Programing Interface, view some taxicab information, request or cancel reservation.

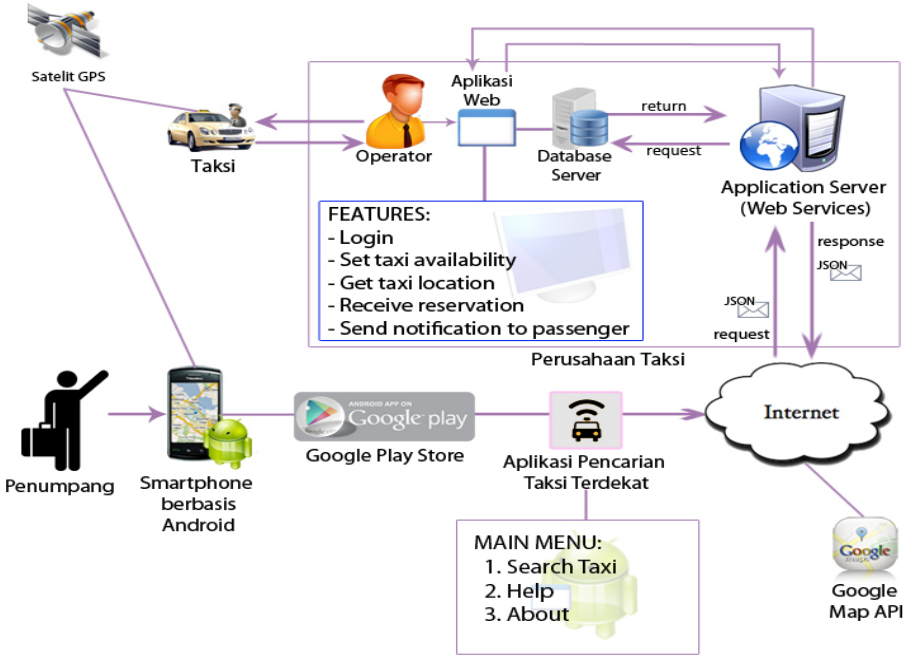
Keywords- *Taxi Search Application, Web Services, Android*

1. PENDAHULUAN

Taksi adalah media transportasi umum penumpang yang dapat melayani perjalanan dengan rute dan jadwal yang fleksibel, mendekati fungsi kendaraan pribadi [1]. Jangkauan wilayah operasi taksi ditentukan oleh perusahaan taksi, dengan tariff yang sesuai dengan argometer. Jenis angkutan ini dapat ditemukan di daerah perkotaan. Untuk menggunakan jasa taksi, konsumen (calon penumpang) dapat menghubungi operator perusahaan taksi melalui telepon atau menunggu taksi yang lewat di jalan. Kendala yang dapat terjadi ketika menghubungi operator adalah panggilan telepon tidak tersambung karena operator sedang melayani konsumen

lain, posisi taksi yang berada jauh dari konsumen, atau konsumen tidak mengetahui posisinya saat ini dengan tepat. Demikian juga ketika menunggu taksi di jalan, pencarian taksi dapat memakan waktu lama. Saat ini aplikasi taksi (call taxi application) sudah digunakan di kota-kota besar, diantaranya adalah Uber [2]. Uber menggunakan sistem 'ridesharing' dimana pengendara Uber menggunakan kendaraan mereka masing-masing untuk mengantarkan penumpang. Penumpang mengirimkan permintaan taksi (*pick-up*) langsung melalui aplikasi. Makalah ini menjelaskan tentang aplikasi berbasis *platform* Android untuk pencarian dan reservasi taksi, yang merupakan pengembangan dari penelitian sebelumnya yang telah dilakukan di [3].

Aplikasi yang dibangun menghubungkan lebih dari satu perusahaan taksi dan pengguna (calon penumpang) dapat mencari taksi terdekat melalui perangkat *mobile*, ditampilkan pada Gambar 1. Terdapat dua aplikasi saling terhubung yang dibangun yaitu aplikasi berbasis *web* untuk perusahaan taksi dan aplikasi *mobile* untuk pengguna.



Gambar 1. Kerangka Aplikasi

Pengguna melakukan pencarian taksi berdasarkan perusahaan taksi yang dipilih melalui menu *Search Taxi*. Selanjutnya akan tampil peta serta taksi-taksi terdekat dan *available* dari perusahaan taksi yang terdaftar. Pengguna dapat melihat informasi taksi (nomor taksi, nama pengendara, dan nomor telepon pengendara) dengan menekan *marker* pada peta. Pengguna juga dapat melakukan pemesanan taksi dengan cara meng-*input* nomor taksi. Aplikasi kemudian mengirim *request* pemesanan tersebut ke perusahaan melalui *application server* perusahaan dan disimpan di basis data. *Request* dari aplikasi *mobile* akan dikirim ke *application server* dengan teknologi *web services* dalam bentuk *JavaScript Object Notation (*JSON*)* menggunakan layanan Internet. *Web services* merupakan sistem perangkat lunak yang terdiri atas kumpulan fungsi atau *methods* yang tersimpan pada sebuah *server* dan dapat dipanggil oleh *client*. Dengan *web services* layanan dan aplikasi dapat digunakan ulang, dan memungkinkan terjadinya pertukaran data antar *platform* yang berbeda. Teknologi *web services* telah dimanfaatkan dalam bidang kesehatan [4], bisnis [5], integrasi data di kepolisian [6], *e-government* [7], dan sebagainya. Antarmuka *web services* dideskripsikan menggunakan format yang mampu diproses oleh mesin, khususnya *Web Services Description Language* (WSDL). Sistem-sistem lainnya berinteraksi dengan *web service* menggunakan pesan SOAP yang umumnya dikirim melalui *Hypertext Transfer Protocol* (HTTP) dalam bentuk *eXtensible Markup Language* (XML) sehingga

memiliki korelasi dengan standar *Web*. SOAP melakukan proses *request* dan *response* antar *web services* dengan aplikasi yang memanggilnya. Sedangkan WSDL berisi informasi *detail* sebuah *web services*, di dalamnya dijelaskan fungsi-fungsi atau *methods* yang tersedia dalam *web services*, parameter yang diperlukan untuk memanggil sebuah *method*, dan hasil atau tipe data yang dikembalikan oleh *method* yang dipanggil, dan *Universal Description, Discovery and Intergration* (UDDI) yang digunakan untuk mengintegrasikan dan mencari *web services* [8].

Aplikasi *web* pada perusahaan mengambil posisi taksi-taksi terdekat yang dideteksi oleh GPS pada *database* perusahaan beserta informasi taksi, kemudian mengirimkan kembali informasi tersebut ke aplikasi *mobile* melalui *application server,* dan ditampilkan pada peta yang diambil dari *service Google Maps Application Programming Interface* (Google Maps API) dengan *marker* di setiap posisi taksi. *Google Maps API* menyediakan kelas, tipe, dan fungsi yang dapat digunakan untuk membangun aplikasi peta *Google Maps.* Pembangunan aplikasi peta dengan *Google Maps API* menggunakan bahasa *Javascript*. Untuk membangun aplikasi *Android* menggunakan *Google Maps API* dibutuhkan *Google Play Service* sebagai *library* dalam proyek. Beberapa konfigurasi yang perlu dilakukan pada *file* `Android Manifest.xml` yaitu menambah *permission* untuk cek koneksi dan akses internet dan untuk deteksi lokasi; menentukan spesifikasi OpenGL; dan menambahkan meta data berupa *key* yang dibuat dengan menambahkan *tag* meta pada *application tree* [9].

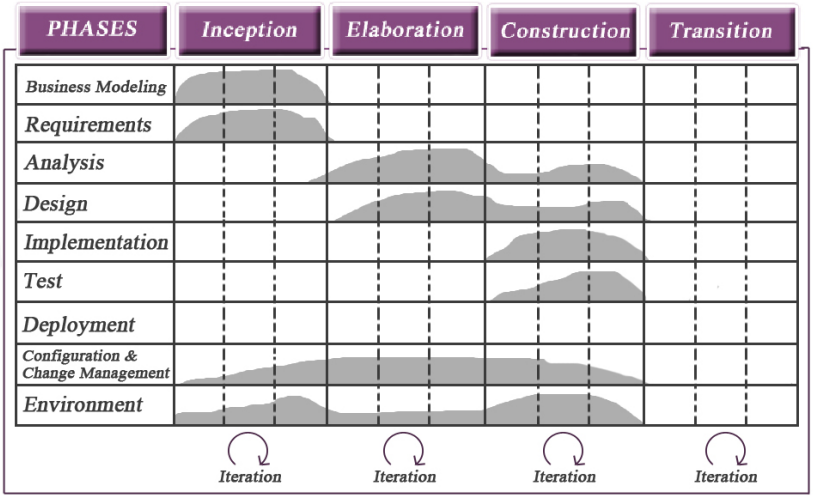
Pada saat pengguna melakukan pemesanan taksi, aplikasi *web* pada perusahaan menampilkan pemberitahuan adanya pesanan taksi dengan *detail* informasi pemesanan. *Operator* kemudian menginformasikan pemesanan kepada pengendara taksi. menggunakan *short message service* (SMS). SMS adalah mekanisme pengiriman pesan singkat melalui dari dan ke perangkat komunikasi *mobile* menggunakan jaringan telekomunikasi selular. Pengiriman pesan dilakukan melalui perangkat yang disebut SMS *gateway*. [10] mendefinisikan SMS *gateway* sebagai alat atau layanan yang mengatur transit SMS, termasuk mengubah pesan pada lalu lintas jaringan selular dari media lain, atau sebaliknya. Peneliti menggunakan aplikasi Gammu untuk menghubungkan basis data SMS *gateway* dengan SMS *device*. Gammu dapat mengontrol pemanggilan kembali pesan yang disimpan, melakukan *backup*, serta mengirimkan pesan [11] Saat ada SMS masuk ke SMS *device*, maka gammu akan memindahkannya ke dalam *inbox* dalam *database* SMS *Gateway.* Sebaliknya saat aplikasi pengirim SMS memasukkan SMS ke dalam *outbox* dalam *database* SMS *gateway*, maka gammu mengirimkannya melalui SMS *device*, dan memindahkan SMS ke *sent item* dalam *database*. Dalam *database* SMS *Gateway* yang di-*generate* otomatis oleh Gammu, seperti SMS yang ada dalam *handphone* yaitu *inbox, outbox,* serta *sent item*. Setelah penjemputan dilakukan pengendara memberikan konfirmasi kepada pihak perusahaan, yang akan mengubah status ketersediaan taksi.

2. METODE PENELITIAN

Pengembangan aplikasi dilakukan menggunakan pendekatan rekayasa perangkat lunak, dengan model proses *rational unified process* (RUP), seperti ditampilkan pada Gambar 2. RUP terdiri atas empat fase yaitu *inception, elaboration, construction,* dan *transition*. Setiap proses kerja (pada kolom paling kiri) dapat berjalan bersamaan dalam setiap fase. Langkah-langkah pengembangan aplikasi dilakukan menurut langkah-langkah sebagai berikut:

1. Inception
   Pada fase ini dilakukan penentuan cakupan dan batasan penelitian, identifikasi kebutuhan fungsional aplikasi, melakukan perancangan awal aplikasi, serta menentukan *tools* yang akan digunakan untuk pengembangan aplikasi
2. Elaboration
   Pada fase ini, dilakukan analisis dan perancangan arsitektur, basis data, dan antarmuka aplikasi Android dan web berdasarkan kebutuhan yang diidentifikasi pada fase *inception*.
3. Construction
   Fase ini merupakan pengimplementasian (pengkodean) rancangan yang telah dibuat pada fase sebelumnya untuk menghasilkan aplikasi dengan spesifikasi yang ditentukan.

4. Transition
   Pada fase ini dilakukan instalasi dan simulasi dari aplikasi yang telah dibuat, termasuk melakukan pengujian terhadap aplikasi yang dibangun dengan metode blackbox untuk menguji apakah aplikasi yang dibuat sudah memenuhi kebutuhan fungsional yang ditentukan pada perencanaan awal.

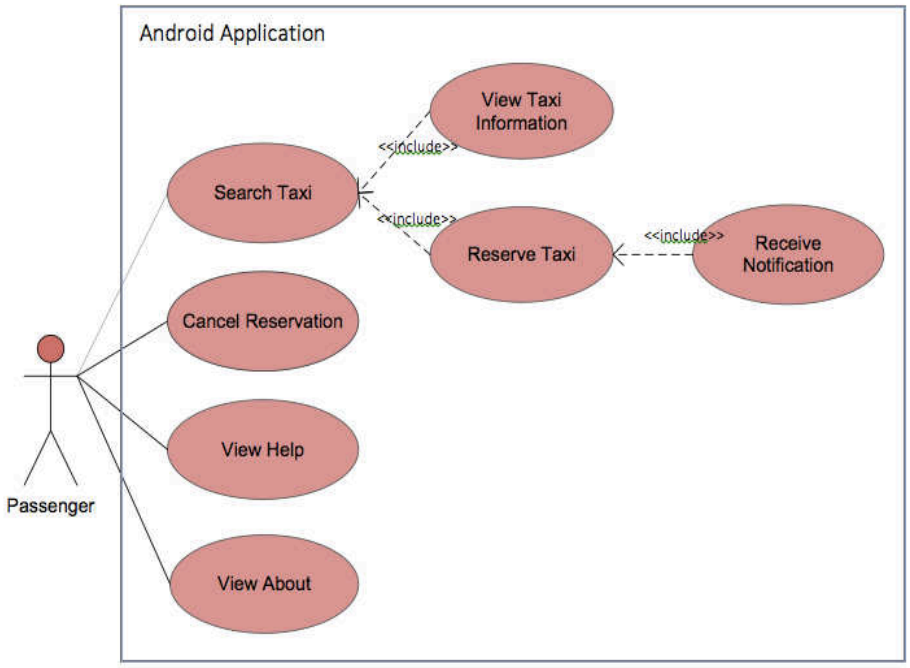


Gambar 2. Metode Penelitian berdasarkan Model Proses RUP

3. HASIL DAN PEMBAHASAN

Bagian ini memamparkan analisis dan rancangan arsitektur aplikasi yang direpresentasikan dalam bentuk diagram *use case* dan *class*, dan implementasi rancangan yaitu aplikasi berbasis *web* dan aplikasi *mobile*.

3.1 Spesifikasi Fungsionalitas Aplikasi

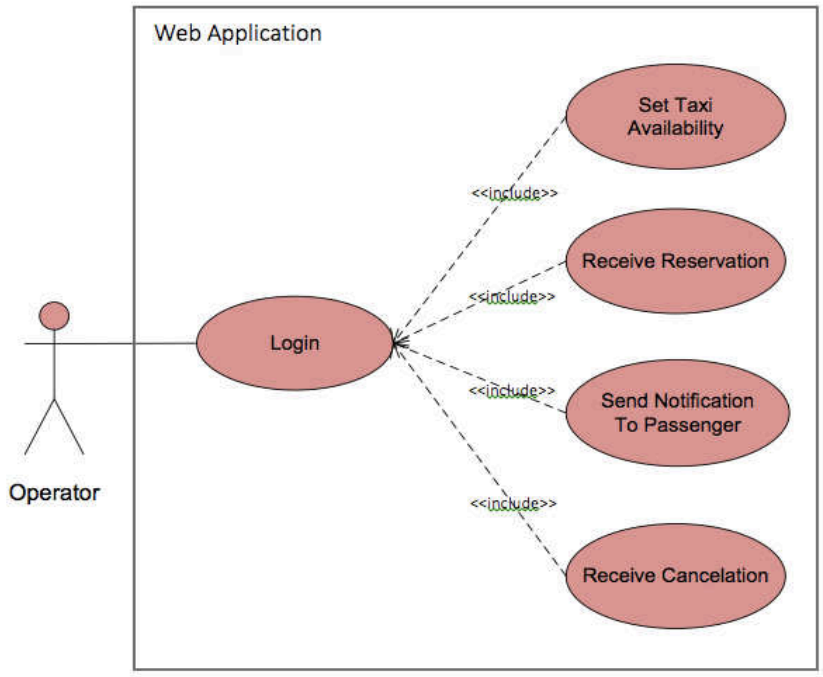


Gambar 3. Diagram *Use Case* Aplikasi Pencarian Taksi - Berbasis Android

Diagram *use case* dipakai untuk merepresentasikan kebutuhan fungsional suatu sistem atau aplikasi. Gambar 3 menampilkan diagram *use case* yang menggambarkan interaksi antara pengguna (*passanger*) dengan aplikasi *mobile*, sedangkan diagram *use case* yang menggambarkan interaksi antara operator dengan aplikasi berbasis web pada perusahaan taksi ditunjukkan pada Gambar 4. Masing-masing diagram dijelaskan sebagai berikut:

Pada Gambar 3 terdapat 5 *use case* utama, dijelaskan sebagai berikut:

1. *Search Taxi*. Pengguna dapat mencari taksi terdekat dan aplikasi akan menampilkan peta dengan *marker* pada setiap posisi taksi terdekat menurut perusahaan taksi yang dipilih pengguna.
2. *View Taxi Information*. Pengguna melihat informasi tentang taksi sesuai dengan *marker* yang dipilih dari hasil pencarian.
3. *Reserve Taxi*. Pengguna melakukan reservasi taksi dan menunggu konfirmasi pesanan dari perusahaan taksi
4. *Receive Notification*. Pengguna menerima pesan konfirmasi dari perusahaan taksi terkait taksi yang diminta.
5. *Cancel Reservation*. Pengguna dapat membatalkan reservasi.

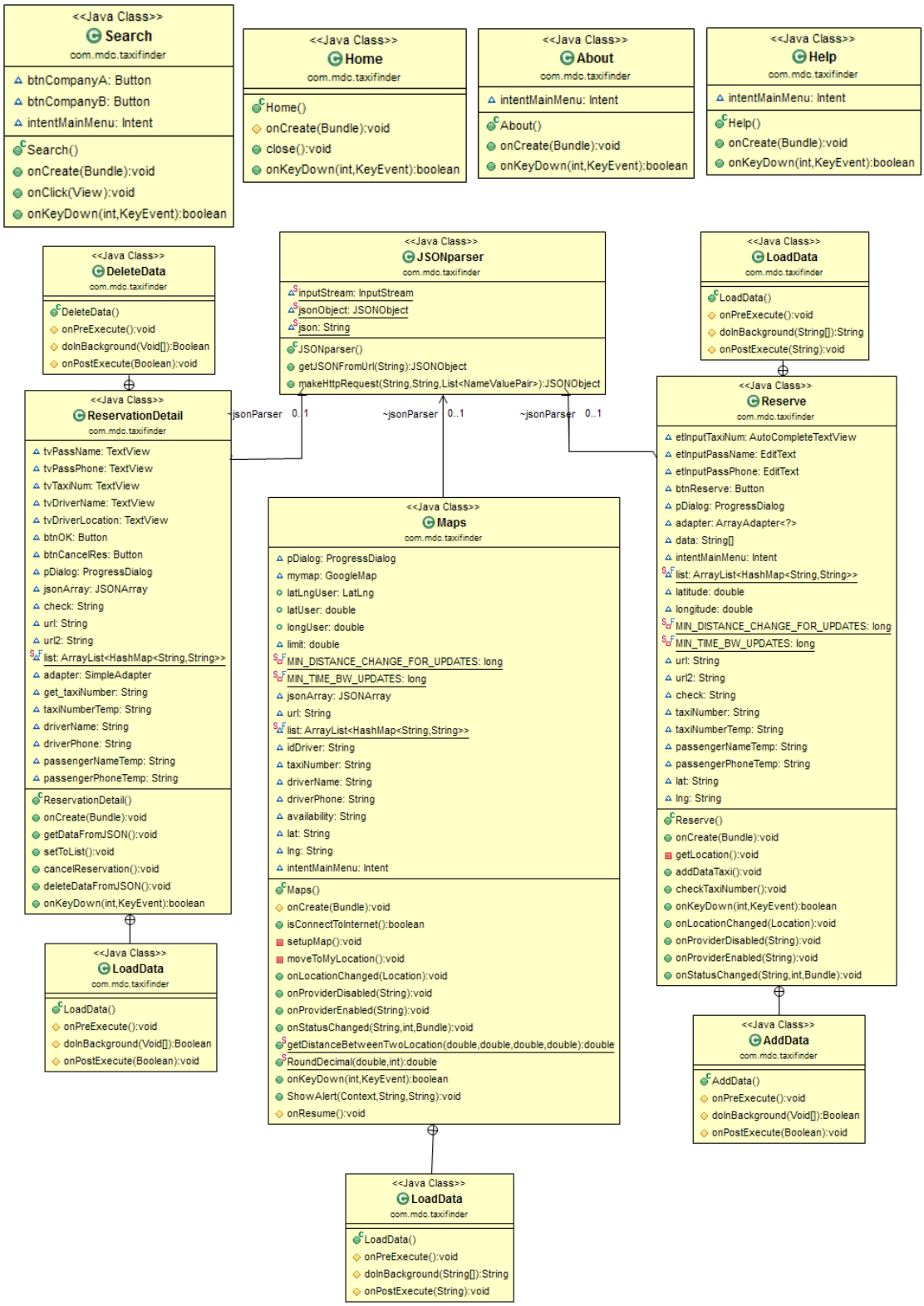


Gambar 4. Diagram *Use Case* Aplikasi Pencarian Taksi - Berbasis Web

Pada Gambar 4 terdapat 5 *use case*, menjelaskan fungsi-fungsi yang disediakan aplikasi web pada perusahaan taksi dengan aktor operator:

1. Login. Fungsi yang memungkinkan operator untuk masuk ke dalam aplikasi web, kemudian mengakses fungsi lainnya.
2. *Set Taxi Availability*. Operator dapat mengubah status ketersediaan taksi *available* atau *unavailable*, setelah mendapat konfirmasi dari pengendara taksi.
3. *Receive Reservation*. Operator mendapat notifikasi adanya pesanan taksi dan dapat melihat daftar pesanan.
4. *Send Notification*. Operator mengirimkan notifikasi kepada pengendara taksi tentang informasi pemesanan.
5. *Receive Cancelation*. Operator mendapat notifikasi adanya pembatalan pesanan taksi, kemudian memberikan notifikasi kepada pengendara taksi.

3.2 Pemodelan

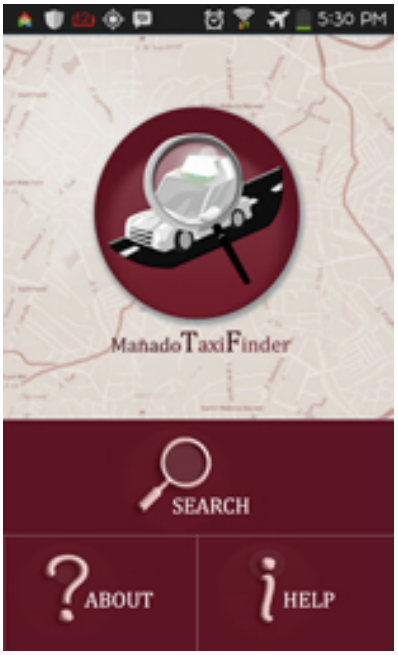


Gambar 5. Diagram *Class* Aplikasi Pemesanan Taksi

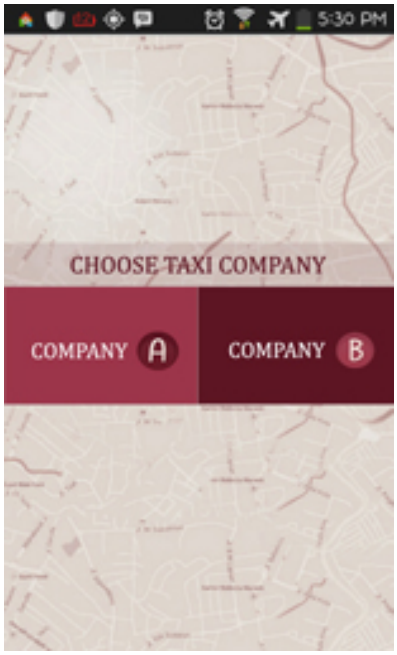
Gambar 5 menjelaskan hubungan antar *class*, atribut-atribut, dan *methods* yang terdapat pada aplikasi. *Class-class* utama dalam aplikasi ini antara lain adalah *Maps* yang berfungsi untuk menampilkan lokasi dan informasi taksi pada *map*, *JSONparser* sebagai tempat berkomunikasi antara aplikasi dengan *web service* perusahaan, *Reserve* yang merupakan *class* untuk menangani informasi pemesanan, *ReservationDetail* untuk menampilkan informasi detail pemesanan taksi.

3.3 Implementasi

Bagian ini membahas implementasi rancangan untuk Aplikasi Pencarian Taksi. Gambar 6 sampai Gambar 10 menampilkan beberapa halaman dari aplikasi *mobile*.



Gambar 6. Halaman Utama



Gambar 7. Halaman *Search*

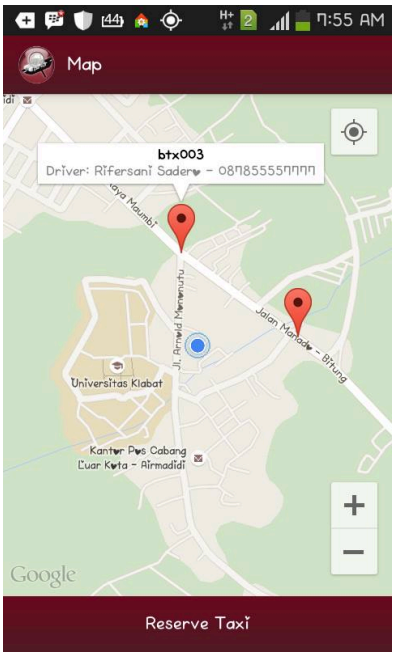
Gambar 6 menampilkan tampilan utama antarmuka dari aplikasi dimana terdapat *button search* untuk proses pencarian taksi, *button help* untuk melihat bantuan penggunaan aplikasi serta *button about* untuk melihat informasi tentang aplikasi. Gambar 7 menampilkan halaman *search* pada aplikasi yang terdiri dari *button Taxi Company A* yang akan menampilkan *marker-marker* taksi dari perusahaan A, dan *button Taxi Company B* akan menampilkan *marker-marker* taksi dari perusahaan B pada peta. Kode program halaman *search*, sebagai berikut:

```
//listener untuk button company A
btnCompanyA.setOnClickListener(new OnClickListener(

     public void onClick(View v){
         Intent iCompanyA = new Intent(Search.this,Maps.class);
         startActivity(iCompanyA);
         }
});
      //listener untuk button company B
btnCompanyB.setOnClickListener(new OnClickListener(){

     public void onClick(View v){
```

```
            Intent iCompanyB = new Intent(Search.this,Reserve.class);
            startActivity(iCompanyB);
        }
});
```



Gambar 8. Tampilan Peta dengan *Marker* Taksi Terdekat

Gambar 8 tampilan aplikasi yang berupa peta yang menunjukkan *current position* dari *user* beserta *marker* berwarna merah yang menunjukkan posisi taksi-taksi terdekat dari perusahaan taksi yang dipilih oleh pengguna. Lokasi taksi dideteksi oleh GPS dan otomatis tersimpan pada basis data. Untuk menampilkan *marker* dari taksi-taksi terdekat pada peta, aplikasi menggunakan teknologi *web services* untuk mengambil data lokasi taksi yang ada dalam format JSON. Penggalan kode program untuk menentukan *current position* dan *marker*, sebagai berikut:

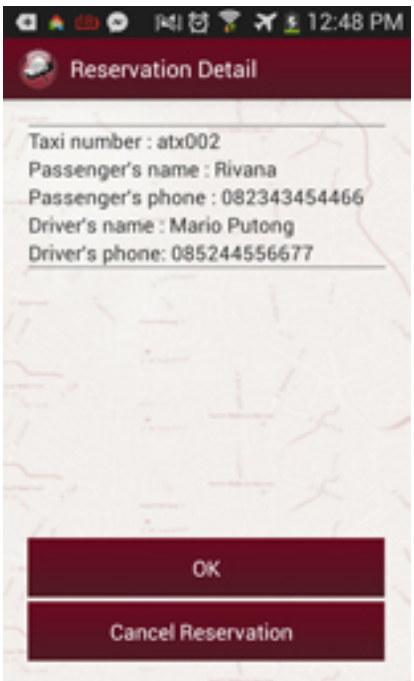
```
// menampilkan lokasi user
private void moveToMyLocation() {
      LocationManager locationManager = (LocationManager)
getSystemService(LOCATION_SERVICE);
                  Criteria criteria = new Criteria();
      String provider = locationManager.getBestProvider(criteria,
true);
      Location location =
locationManager.getLastKnownLocation(provider);
      locationManager.requestLocationUpdates(provider,
MIN_TIME_BW_UPDATES, MIN_DISTANCE_CHANGE_FOR_UPDATES, this);
      …
}
// mengganti lokasi ke current location
public void onLocationChanged(Location location) {

latUser = location.getLatitude();
longUser = location.getLongitude();

latLngUser = new LatLng(latUser, longUser);
```

```
      // menampilkan current location pada map
mymap.moveCamera(CameraUpdateFactory.newLatLng(latLngUser));
mymap.animateCamera(CameraUpdateFactory.zoomTo(15));
}
      …

protected void onPostExecute(String file_url) {
    for (int i = 0; i < list.size(); i++) {
          // ambil data JSON dari list
    HashMap<String, String> map = new HashMap<String, String>();
    map = list.get(i);
    String detail = "Driver: " + map.get("driverName") + " - "
    + map.get("driverPhone");
          // menghitung jarak taksi dari posisi user
    double latTaxi = Double.parseDouble(map.get("lat"));
    double lonTaxi = Double.parseDouble(map.get("lng"));
    double latU = Double.valueOf(latUser);
    double longU = Double.valueOf(longUser);
    double jarak = getDistanceBetweenTwoLocation(latU, longU,
        latTaxi, lonTaxi);
    jarak = RoundDecimal(jarak, 2);
               if (jarak <= limit){
           // menampilkan marker pada map
           mymap.addMarker(new MarkerOptions().position(new
LatLng(latTaxi,lonTaxi))
           .title(map.get("taxiNumber")).snippet(detail));
         }
    }
    pDialog.dismiss();    }
```



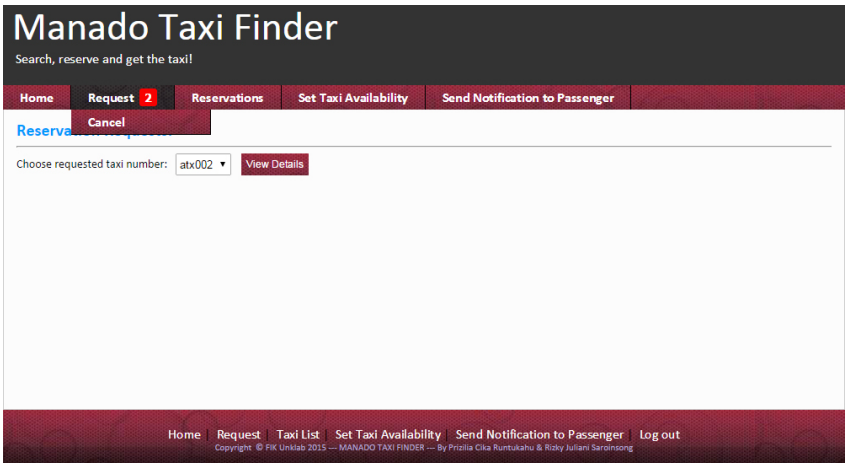
Gambar 9. Halaman *Reserve Taxi* **Gambar 10. Halaman *Reservation Detail***

Gambar 9 tampilan halaman *reserve taxi* yang berisi *field* yang harus diisi oleh *user* yaitu nomor taksi, nama dan nomor telepon. Informasi pemesanan yang diisi akan dikirim

menggunakan teknologi *web service* dan ditangani oleh *operator* perusahaan dari taksi yang dipilih. Gambar 10 merupakan tampilan halaman *reservation detail* yang menampilkan detail pemesanan taksi yaitu nomor taksi, nama, nomor telepon, nama pengendara, dan nomor telepon pengendara yang diambil pada *database* perusahaan dalam format JSON menggunakan teknologi *web services*. Penggalan kode program untuk *Reserve*, sebagai berikut:

```
// listener untuk button Reserve
btnReserve.setOnClickListener(new View.OnClickListener() {
@Override
public void onClick(View v) {
// TODO Auto-generated method stub
      // mendapatkan String yang diinput oleh user
taxiNumberTemp = etInputTaxiNum.getText().toString();
passengerNameTemp = etInputPassName.getText().toString();
passengerPhoneTemp = etInputPassPhone.getText().toString();
      // validasi jika field kosong
if (taxiNumberTemp.equals("") || passengerNameTemp.equals("")
|| passengerPhoneTemp.equals("")) {
      Toast.makeText(getApplicationContext(),
"Please fill the form!", Toast.LENGTH_LONG).show();
} else {
// mendapatkan lokasi user
getLocation();
lat = Double.toString(latitude);
lng = Double.toString(longitude);

// input data pemesanan menggunakan JSON
AddData add = new AddData();
add.execute();}
}
});
…
```



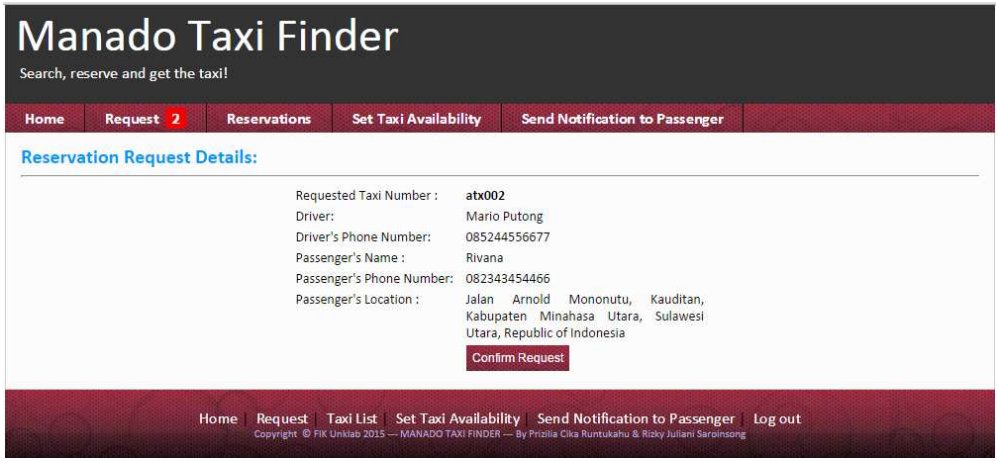
Gambar 11. Halaman *Request* pada Aplikasi Web

Gambar 11 menampilkan halaman *Request* yang berisi daftar pemberitahuan permintaan pemesanaan taksi dari penumpang. Kode program untuk menampilkan daftar pesanan taksi dari pengguna, sebagai berikut:

```
//ambil data request dari database
```

```
$query_notification = "SELECT * FROM tbl_request";
$notification = $mysqli->query($query_notification);
$row_notification = $notification->fetch_assoc();
$totalRows = mysqli_num_rows($notification);

//tampilkan request
<?php do { ?>
<option value="<?php echo $row_notification["taxiNumberTemp"]?>"
selected="selected">
<?php echo $row_notification["taxiNumberTemp"]?>
</option>
<?php } while ($row_notification = $notification->fetch_assoc());
$rows = mysql_num_rows($notification);
if($rows > 0) {
mysqli_data_seek($notification, 0);
$row_notification = $notification->fetch_assoc();
} ?>
```



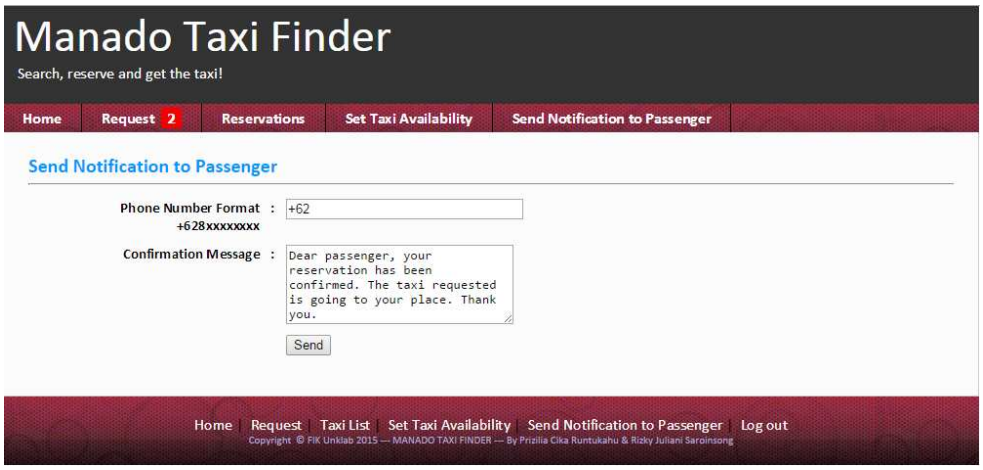
Gambar 12. Tampilan Halaman *Reservation Request Details* pada Aplikasi Web

Gambar 12 menampilkan halaman *Reservation Request Detail* yang tabel rincian informasi permintaan pemesanan dari penumpang, yaitu nomor taksi, nama pengendara, nomor telepon pengendara, nama penumpang, nomor telepon calon penumpang, lokasi penumpang. Sebagian kode program untuk menampilkan detail reservasi, sebagai berikut:

```
//ambil data dari tabel request
$colname_request = "-1";
if (isset($_POST['lst_taxiNumber']))
{
  $colname_request = (get_magic_quotes_gpc()) ?
  $_POST['lst_taxiNumber'] : addslashes($_POST['lst_taxiNumber']);
}
$mysqli->select_db($db_name) or die ("no database");
$query_request = sprintf("SELECT * FROM tbl_request WHERE
taxiNumberTemp = '%s'", $colname_request);
$request = $mysqli->query($query_request) or die(mysql_error());
$row_request = $request->fetch_assoc();

//tampilkan detail pemesanan
echo $row_request['taxiNumberTemp'];
echo $row_taxi['driverName'];
echo $row_taxi['driverPhone'];
```

```
echo $row_request['passengerNameTemp'];
echo $row_request['passengerPhoneTemp'];
$address = getaddress($lat,$lng);
if($address){ echo $address;
}else {
echo "Address not found.";
}
```



Gambar 13. Tampilan Halaman *Send Notification to Passenger* pada Aplikasi Web

Gambar 13 menampilkan antarmuka halaman *Send Notification to Passenger* yang berisi nomor telepon penumpang dan pesan konfirmasi yang akan dikirimkan ke penumpang. Kode program untuk mengirim notifikasi kepada pengguna, menggunakan Gammu, sebagai berikut:

```
//ambil data yang diinput
$noTujuan = $_POST['nohp'];
$message = $_POST ['msg'];

//kirim pesan dengan sms gateway
exec('D:\Gammu\bin\gammu-smsd-inject.exe -c D:\Gammu\bin\smsdrc EMS
'.$noTujuan.' -text "'.$message.'"');
header("Location: logReservation.php");
```

4. KESIMPULAN

Dari uraian tentang pengembangan aplikasi pencarian taksi yang dijelaskan pada makalah ini, maka dapat ditarik kesimpulan sebagai berikut:

1. Implementasi teknologi *web service* telah menghasilkan aplikasi pencarian taksi terdekat yang dapat mengintegrasikan aplikasi berbasis *web* dengan *platform* berbeda, dengan aplikasi *mobile* berbasis sistem operasi Android.
2. Pemanfaatan *Google Maps API* memungkinkan aplikasi menampilkan *marker* posisi taksi-taksi terdekat pada peta, sehingga pengguna dapat melihat dan memilih taksi secara langsung.
3. Teknologi *web services* dimanfaatkan untuk menampilkan informasi taksi pada peta, proses pemesanan taksi, dan pembatalan pemesanan.

Untuk pengembangan lanjutan dari aplikasi ini, dapat ditambahkan fitur sebagai berikut:

1. Menampilkan rute dari *current position user* ke arah taksi yang dipilih.

2. Membuat aplikasi tambahan bagi pengendara taksi agar status ketersediaan taksi dapat di-*update* secara otomatis melalui aplikasi tersebut.
3. Menambahkan fitur *tracking* bagi taksi agar *user* dapat melihat posisi taksi yang sedang menuju ke lokasi dari *user*.

DAFTAR PUSTAKA

[1] S. Warpani, *Merencanakan Sistem Perangkutan*. Bandung: ITB, 1990.

[2] Rempel, J. A Review of Uber, the Gworing Alternative to Traditional Taxi Service. AFB AccessWorld Magazine, Vol. 15, No. 6, June 2014. http://www.afb.org/afbpress/Pub.asp?DocID=aw150602, Diakses 7 Juli 2014.

[3] Runtukahu, P., Saroinsong, R., dan Sondakh, D, 2015, Desain Aplikasi Pencarian Taksi Menggunakan Teknologi Web Services Berbasis Android, Prosiding Konferensi Nasional Sistem Informasi 2015, Manado, 26-28 Februari 2015.

[4] Hidayat, R. dan Azhari. A., 2013, Penerapan Teknologi Web Service Untuk Integrasi Layanan Puskemas dan Rumah Sakit, *Berkala MIPA*, vol. 23, hal. 64-77.

[5] Marthasari, G.I., Aminudin, dan Munarko, Y., 2010, Implementasi *Web Service* Untuk Mendukung Interoperabilitas pada Aplikasi *E-Commerce*, The 12th Industrial Electronics Seminar, Indonesia 3 Nopember 2010.

[6] Kenali, E. W., 2010, Implementasi Web Service untuk Integrasi Data Satuan Reserse Kriminal: Studi Kasus di Polda Lampung, *Tesis*, Program Pasca Sarjana Ilmu Komputer, Univ. Gadjah Mada,Yogyakarta.

[7] Sutanta, E. dan Mustofa, K., 2012, Kebutuhan Web Service Untuk Sinkronisasi Data Antar Sistem Informasi Dalam *E-Gov* di Pemkab Bantul Yogyakarta, *JURTIK*, vol. 1.

[8] W3C, *Web Services Architecture* (2004,Februari), http://www.w3.org/TR/ws-arch/#whatis, Diakses 18 Agustus 2014.

[9] Elfarqy, Menggunakan *Google Maps* API pada *Android*, http://www.bisakomputer.com/menggunakan-*Google*-*Maps*-api-pada-*Android*/, Diakses: 29 September 2014.

[10] Katankar, V. K. dan Thakare, V. M., 2010, Short Message Service using SMS Gateway, *International Journal on Computer Science and Engineering*, vol. 02, hal. 1487-1491.

[11] Gammu, http://wammu.eu/gammu/, diakses 3 Oktober 2014.