RANCANG BANGUN *WEB SERVICE* UNTUK PENJUALAN TIKET BUS DAMRI

[1)] Hamdani
[2)] Haviluddin
[3)] Ngurah Satria Darmawangsa

[1,2,3)] Pogram Studi Ilmu Komputer, FMIPA Universitas Mulawarman
Email : hamdani@unmul.ac.id [1)], haviluddin@yahoo.com [2)], ngurahsd@yahoo.com [3)]

ABSTRAK

Keberadaan jasa transportasi bus bagi masyarakat provinsi Kalimantan Timur sudah tidak asing lagi. Masyarakat Kalimantan Timur masih banyak yang memanfaatkan jasa transportasi bus untuk berpergian. DAMRI sebagai salah satu perusahaan jasa angkutan bus yang banyak diminati masyarakat, sebaiknya menyediakan suatu pelayanan terbaik dan maksimal kepada calon penumpang. Terutama dalam sistem penjualan tiket, baik melalui agen ataupun melalui *website*.

Berdasarkan permasalahan tersebut, maka dibutuhkan suatu sistem yang dapat membantu dalam pengelolaan dan pendistribusian data penjualan tiket bus dari agen ke PERUM DAMRI. Penelitian ini bertujuan untuk membangun aplikasi *web service* sebagai aplikasi *middleware* dan peyedia layanan penjualan tiket bus, sebagai *interface*-nya dibangun aplikasi antar muka berupa aplikasi *desktop* untuk agen dan PERUM DAMRI, serta dibangun aplikasi *website* untuk calon penumpang.

Perangcan aplikasi pada *desktop* dan *website* sebagai *interface* untuk komunikasi langsung ke *user*, serta aplikasi *web service* sebagai penyedia layanan dan *midlleware* antar kedua aplikasi tersebut. Dengan adanya *web service* ini diharapkan dapat memudahkan penjualan dan pengelolan data penjualan tiket bus dari agen pada PERUM DAMRI.

Kata Kunci : *Web service, Website, Middleware, DAMRI.*

PENDAHULUAN

Keberadaan jasa transportasi bus bagi masyarakat provinsi Kalimantan Timur sudah tidak asing lagi. Dapat kita lihat masih banyak masyarakat yang memanfaatkan jasa transportasi bus untuk berpergian, terutama pada hari-hari libur sekolah atau libur hari besar yang memang sudah menjadi tradisi masyarakat kita untuk bepergian keluar kota untuk mengisi waktu libur.

DAMRI sebagai salah satu perusahaan jasa angkutan yang banyak diminati masyarakat sebaiknya menyediakan suatu pelayanan terbaik dan maksimal kepada calon para penumpang. Terutama dalam sistem penjualan tiket kepada penumpang, karena hingga saat ini perusahaan-perusahaan bus di samarinda yang menyediakan jasa yang sama, dalam hal menangani reservasi dan penjualan tiketnya masih banyak yang menggunakan sistem tradisional, dimana para penumpang diharuskan membeli tiket langsung ditempat (terminal).

Berdasarkan latar belakang diatas, maka dibutuhkan suatu sistem yang mendukung komunikasi aplikasi *client* dalam hal ini adalah agen dan aplikasi *server* dalam hal ini adalah perusahaan bus DAMRI. Selain itu juga dibutuhkan aplikasi untuk pembelian online berupa website untuk mempermudah penumpang dalam reservasi tiket.

Untuk mengatasi masalah tersebut maka dibutuhkan sebuah aplikasi yang *multiplatform* agar pihak agen juga dapat mengembangkan aplikasi yang akan digunakan sesuai dengan kebutuhan masing-masing. Selain itu, aplikasi ini juga dapat menghubungkan *server* dengan *client-client* dalam proses pemesanan tiket. Dari semua yang dibutuhkan, *web service* adalah teknologi yang paling tepat. Karena *web service* bersifat *multiplatform* dan bisa menjembatani aplikasi-aplikasi meskipun berbeda sistem operasi dan bahasa pemrograman agar dapat melakukan pertukaran data dan informasi dengan mudah.

Dengan adanya *web service* yang mampu mengelola administrasi penjualan tiket bus ini, diharapkan dapat memudahkan agen dalam melakukan pekerjaannya. Karena para agen dari perusahaan bus cukup menyediakan *device* dan sebuah aplikasi *client* yang dapat mengolah transakasi yang dikirimkan oleh *web service* itu. Selebihnya agen cukup menangani masalah keuangan saja dan tidak perlu lagi untuk konfirmasi, pengecekan, dan pembuatan laporan dalam administrasi penjualan karena data sudah langsung tersimpan dalam *database server*.

TINJAUAN PUSTAKA

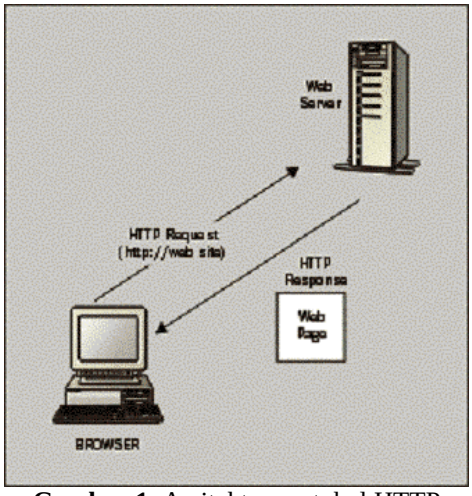
Web

Web adalah suatu ruang informasi dimana sumber-sumber daya berguna diidentifikasi oleh pengenal global yang disebut *Uniform Resource Identifier (URL)*. Sebuah halaman *Web* diakses dengan cara menuliskan URLnya atau mengikuti link yang menuju kepadanya, menggunakan *browser web*. Berikut cara kerja *web*, diantaranya adalah [2]:

1. URL menunjukan lokasi dokumen yang dikelola oleh sebuah *server web*.
2. URL diubah menjadi alamat *Internet Protocol* (IP) *server web* yang bersangkutan.
3. *Browser* kemudian mengirimkan *request* http ke *server web*.
4. *Server web* akan menjawab dengan memberikan dokumen yang diminta dalam format HTML.

HTTP

*Hypertext Transfer Proto*col (HTTP) adalah sebuah protokol jaringan lapisan aplikasi yang digunakan untuk sistem informasi terdistribusi, kolaboratif, dan menggunakan hipermedia. Penggunaannya banyak pada pengambilan sumber daya yang saling terhubung dengan tautan, yang disebut dengan dokumen hiperteks, yang kemudian membentuk *World Wide Web* pada tahun 1990 oleh fisikawan Inggris, Tim Berners-Lee.



Gambar 1. Arsitektur protokol HTTP

HTTP adalah sebuah protokol meminta/menjawab antara *client* dan *server*. Sebuah *client* HTTP (seperti *web browser* atau robot dan lain sebagainya), biasanya memulai permintaan dengan membuat hubungan ke *port* tertentu di sebuah *server Web hosting* tertentu (biasanya *port* 80). Klien yang mengirimkan permintaan HTTP juga dikenal dengan *user agent*. *Server* yang meresponsnya, yang menyimpan sumber daya seperti berkas HTML dan gambar, dikenal juga sebagai *origin server*. Di antara *user agent* dan juga *origin server*, bisa saja ada penghubung, seperti halnya *proxy*, *gateway*, dan juga *tunnel* [1,2].

WWW

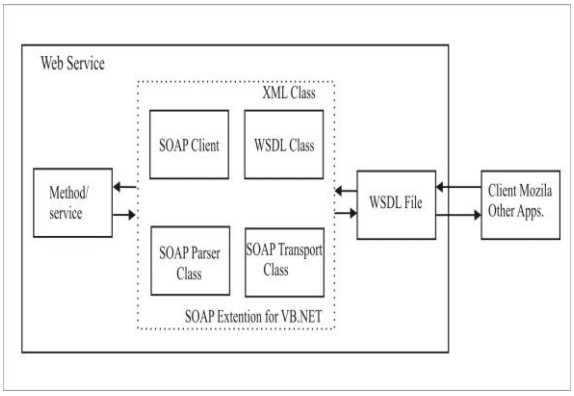
World Wide Web atau yang disingkat WWW, merupakan kumpulan *web server* dari seluruh dunia yang berfungsi menyediakan data dan informasi untuk dapat digunakan bersama. WWW atau biasa disebut *web* adalah bagian yang paling menarik dari Internet. Melalui *web*, dapat mengakses informasi-informasi yang tidak hanya berupa teks tetapi bisa juga berupa gambar, suara, video dan animasi.

Jadi dapat disimpulkan bahwa WWW adalah sekelompok dokumen multimedia yang saling terkoneksi menggunakan *hyperteks link.* Dengan mengklik *hyperlink*, maka bisa berpindah dari satu dokumen ke dokumen lainnya [2,3].

Web Service

Web service diartikan sebagai sebuah antar muka (*interface*) yang meng-gambarkan sekumpulan operasi-operasi yang dapat diakses melalui jaringan, misalnya internet, dalam bentuk pesan XML. *Web service* diartikan sebagai sepotong atau sebagian informasi atau proses yang dapat diakses oleh siapa saja, kapan saja dengan menggunakan piranti apa saja, tidak terikat dengan sistem operasi atau bahasa pemrograman yang digunakan.

Hal ini dimungkinkan karena *web service* berkomunikasi menggunakan sebuah standar format data yang universal yaitu XML dan menggunakan protokol SOAP. Karena *web service* menggunakan format data XML, maka *web service* juga mewariskan sifat *multi-tier* dari XML sehingga memungkinkan terjadinya integrasi antar *web service* atau aplikasi. Arsitektur *Web Service* yang terdiri dari beberapa teknologi yang saling berhubungan dan berlapis seperti digambarkan sebagai berikut [7].



Gambar 2. Arsitektur *Web Service*

Langkah yang dilakukan untuk mendefinisikan *web service* :

1. Pemanggilan *web browser* yang membuat *call procedure* pada file WSDL dan klien layanan SOAP (SOAP *Service Client*),
2. Klien SOAP *Service* mengambil method dan parameter untuk membangun kontainer XML, kontainer ini dikirim melalui HTTP sebagai SOAP *request*.
3. Server SOAP *Service* menerima SOAP *request*; SOAP *parser class* mengubah XML *container* dan menentukan *method* yang dipanggil serta parameter-parameter *method* pada *web service*.
4. *Method* kemudian dieksekusi pada *server* serta mengirimkan *output*,
5. Hasil/*output* kemudian dibungkus sebagai XML dan s*erver* mengirimkan XML *result container* sebagai respon untuk *request POST* oleh SOAP *transport http class*.
6. Klien mengubah XML *response container* dan mengirimkan hasil ke aplikasi yang memanggilnya.
7. Aplikasi kemudian memproses hasil.

Perbandingan Web Service dan Website

Web Services dapat di definisikan sebagai aplikasi yang diakses oleh aplikasi yang lain. Mungkin orang berpendapat itu semacam *website*, tetapi itu bukan demikian. Perbandingan tersebut dapat dilihat pada table 1 dibawah. [3,4]:

Tabel 1. Perbandingan *Website* dan *Webservice*

Website	Web Service
Memiliki *web interface*	Tidak memiliki *interface* yang bagus
Dibuat untuk langsung berinteraksi dengan *user*	Dibuat untuk langsung berinteraksi dengan aplikasi yang lain, baik beda OS/konsep sekalipun
Dibuat untuk bekerja pada *web browser*	Dibuat untuk bekerja disemua tipe *client*

Extensible Markup Language (XML)

Extensible Markup Language (XML) merupakan salah satu *metamarkup language* yang berupa teks biasa seperti dokumen HTML. Namun XML dapat menyediakan format *tag* yang dapat kita tentukan sendiri untuk menggambarkan data secara terstruktur. XML menyediakan fasilitas untuk pendeklarasian isi data yang dimuat dalam dokumen XML secara lebih tepat dan memberi hasil pencarian yang lebih baik untuk aplikasi dengan *platform* apa pun.Sebagai tambahan, XML dapat mendukung kelahiran aplikasi generasi baru dalam hal manipulasi data yang berbasis *web* [5,6].

Simple Object Access Protocol (SOAP)

SOAP (*Simple Object Access Protocol*) adalah standar untuk bertukar pesan-pesan berbasis XML melalui jaringan komputer atau sebuah jalan untuk program yang berjalan pada suatu sistem operasi (OS) untuk berkomunikasi dengan program pada OS yang sama maupun berbeda dengan menggunakan HTTP dan XML sebagai mekanisme untuk pertukaran data. SOAP menspesifikan secara jelas bagaimana cara untuk meng-*encode header* HTTP dan file XML sehingga program pada suatu komputer dapat memanggil program pada pada komputer lain dan mengirimkan informasi, dan bagaimana program yang dipanggil memberikan tanggapan.

Web Service Description Language (WSDL)

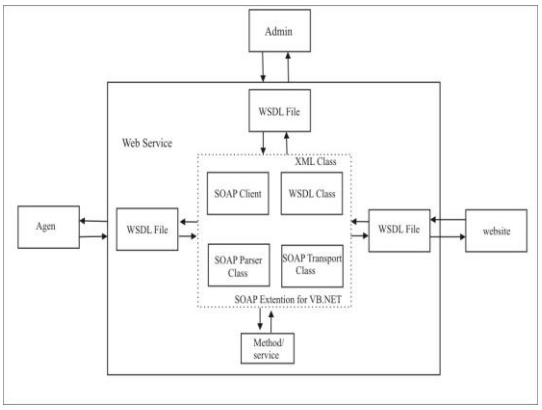
WSDL (*Web Services Description Language*) adalah format XML yang digunakan untuk menerangkan *web service*. WSDL menyediakan sebuah kamus XML untuk menjabarkan detil-detil ini. Definisi WSDL yang lengkap terdiri dari seluruh informasi yang dibutuhkan untuk meminta *web service*. Pengembang yang mau mempermudah yang lain untuk mengakses *service-service*nya harus menyediakan defisi-definisi WSDL. WSDL memainkan peranan penting pada seluruh arsitektur *web service* semenjak menjabarkan kontrak lengkap pada komunikasi aplikasi. Ini membantu untuk memastikan *interoperbilitas* pada *layer* deskripsi *service* [4].

Universal Description, Discovery, and Integration (UDDI)

UDDI merupakan suatu directory *service* untuk *web services*, dimana didalamnya kita bisa mencari *web services* berdasarkan *keyword* dan kategori tertentu. Kemampuan atribut metadata untuk *service-service* didaftarkan pada UDDI, dan lalu menjalankan *queri-queri* berdasarkan pada metadata tersebut yang menengahi secara mutlak menuju tujuan dari UDDI pada kedua waktu desain dan waktu pengeksekusian [6].

HASIL PENELITIAN

Web service penjualan tiket mempunyai fungsi sebagai *middleware* dan sebagai penyedia layanan/*service* yang nantinya akan dipanggil oleh aplikasi lain. Sebagai *interface* untuk memanggil layanan yang disediakan oleh *web service* maka dibutuhkan aplikasi antar muka, diantaranya adalah aplikasi *desktop* untuk agen dan admin, serta aplikasi *website* untuk calon penumpang yang ingin melakukan pemesanan tiket secara *online*.



Gambar 3. Arsitektur sistem penjualan tiket bus

Pada gambar 3 dapat dilihat arsitektur sistem yang akan dibangun, terdapat tiga aplikasi antar muka untuk mengakses layanan *web service* antara lain adalah aplikasi admin, aplikasi agen serta aplikasi *website*, *web service* ini berfungsi sebagai jembatan dan penyedia layanan untuk ketiga aplikasi tersebut.

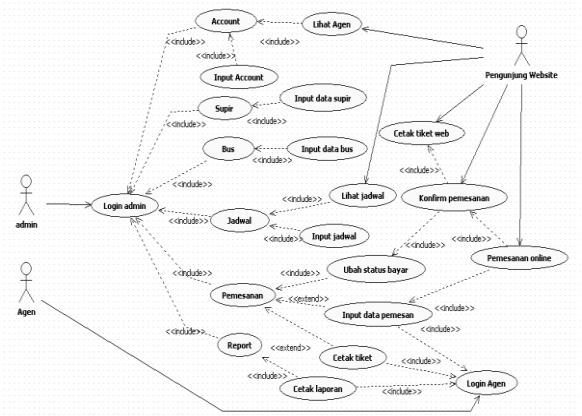
Untuk dapat mengakses *web service* setiap aplikasi terlebih dahulu melakukan *call prosedure* terhadap file WSDL yang berfungsi untuk mencari layanan atau method-method apa saja yang disediakan oleh *web service*. Jika layanan tersebut tersedia dalam *web service* maka *web service* akan mengirimkan *output*/hasil dalam format dokumen XML ke aplikasi antarmuka yang melakukan *call procedure* [6,7].

DESAIN SISTEM

Use Case Diagram

Diagram *use case* menggambarkan aktifitas yang dilakukan oleh suatu sistem dari sudut pandang pengamatan luar, dapat ditunjukkan seperti pada gambar 4. Dalam sistem penjualan tiket *web service* terdapat tiga aktor yang terlibat, antara lain:

1. Admin
   Admin adalah perusahan DAMRI. Admin memiliki hak akses penuh ke semua menu yang ada pada aplikasi *desktop*.
2. Agen
   Agen dalam sistem ini adalah agen perjalanan yang telah ditunjuk oleh perusahaan DAMRI untuk menjual tiket.
3. Pengunjung *website*
   Pengunjung atau pembeli adalah aktor yang membeli tiket baik yang melalui agen ataupun melalui pembelian secara *online*.



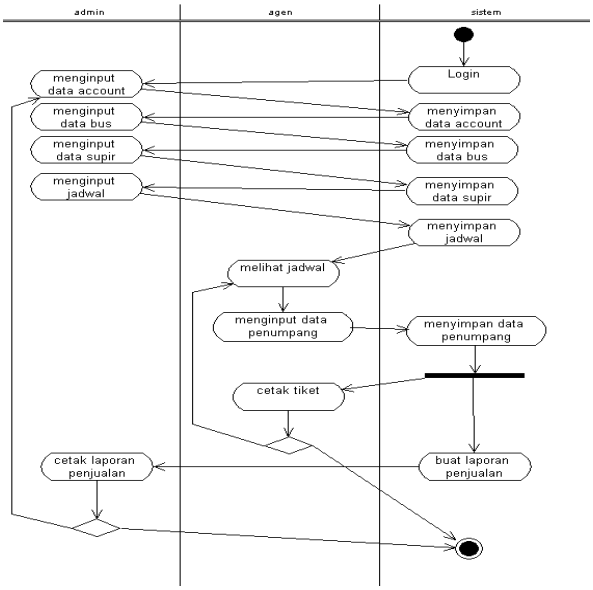
Gambar 4. *Use case diagram* penjualan tiket bus.

Activity Diagram

Sebuah diagram yang meng-gambarkan sifat dinamis dari sebuah sistem dengan permodelan aliran kontrol dari aktifitas ke aktifitas. Sebuah aktifitas merupakan operasi dari beberapa *class* pada sistem yang menghasilkan perubahan pada keadaan sistem.

a. *Activity Diagram* antara admin dan agen

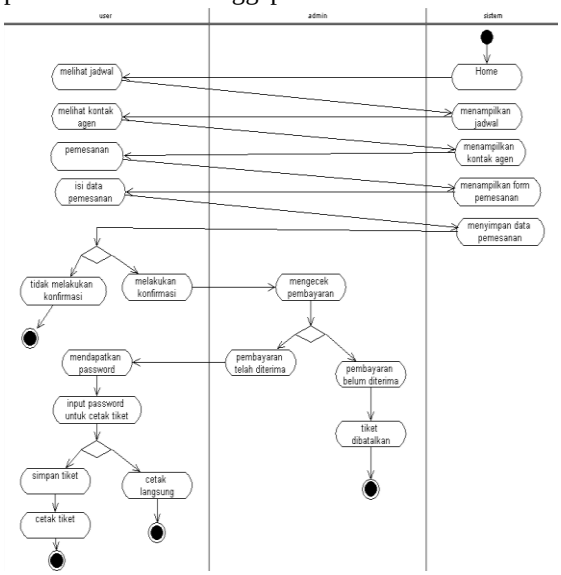
Dapat dilihat pada gambar 5 dimana admin dan agen melakukan *login* dan sistem yang memprosesnya, admin dan agen mempunyai aktifitas yang bebeda sesuai hak akses yang diberikan dimana admin melakukan aktifitas menginput data *account*, menginput data bus, menginput data supir, menginput jadwal yang nantinya dapat diakses oleh agen, admin juga dapat mencetak laporan. Sedangkan agen hanya dapat melakukan beberapa aktifitas diantaranya adalah melihat jadwal, menginput data penumpang dan mencetak tiket.



Gambar 5. *Activity diagram* antara admin dan agen pada aplikasi *desktop*

b. *Activity Diagram* antara pemesan dan dmin

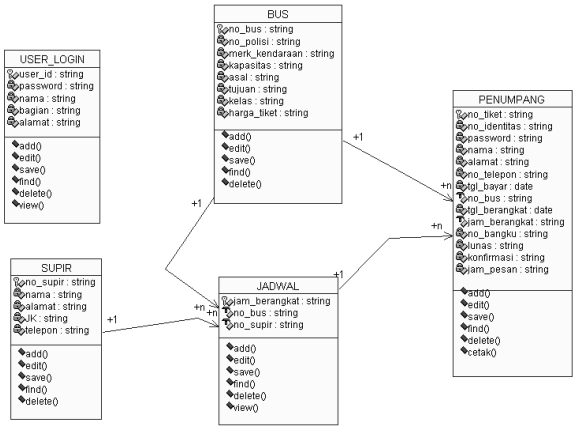
Alur *diagram activity* antara pemesan dan admin dapat dilihat pada gambar 4.4. Pada pemesan aktifitas yang dilakukan adalah melihat jadwal, melihat kontak agen, melakukan pemesanan melalui *website* dengan mengisi *form* pemesanan. Setelah itu pemesan dapat melakukan pembayaran dan konfirmasi pembayaran kepada admin untuk mendapatkan *password* untuk mencetak tiket *online*, dan jika tidak melakukan pembayaran pemesanan akan dianggap batal.



Gambar 6. *Activity* diagram antara admin dan pemesan pada *website*

Class Diagram

Pada sistem penjualan tiket bus ini terdapat lima buah *class*, dimana hanya ada satu *class* yang berdiri sendiri yaitu *class* USER_LOGIN dan empat *class* lainnya saling terkait diantaranya adalah: *class* BUS, *class* SUPIR, *class* JADWAL, *class* PENUMPANG. Relasi *class* tersebut dapat dilihat pada gambar 7.

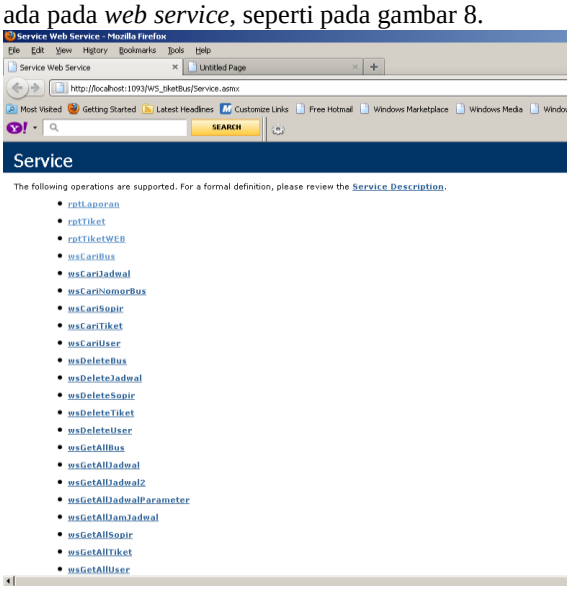


Gambar 7. *Class Diagram* Pemesanan Tiket

PENGUJIAN SISTEM

Layanan Web service

Aplikasi *web service* merupakan aplikasi penyedia layanan yang menjembatani aplikasi antar muka penjualan tiket bus, sehingga sistem penjualan tiket ini bisa berjalan. Aplikasi *web service* hanya menampilkan *method-method* yang ada pada *web service*, seperti pada gambar 8.



Gambar 8. Tampilan dari *web service*

Web Method wsGetAllUser

Web Method wsGetAllUser adalah *method* dari *web service* untuk melihat kontak agen, pada *web method* wsGetAllUser terdapat *button invoke* seperti pada gambar 9. *Button invoke* berfungsi menampilkan *file* XML yang berisikan data *user*. *File* ini nantinya dapat diakses oleh aplikasi *web*.

Gambar 9. Tampilan dari *Method* wsGetAllUser

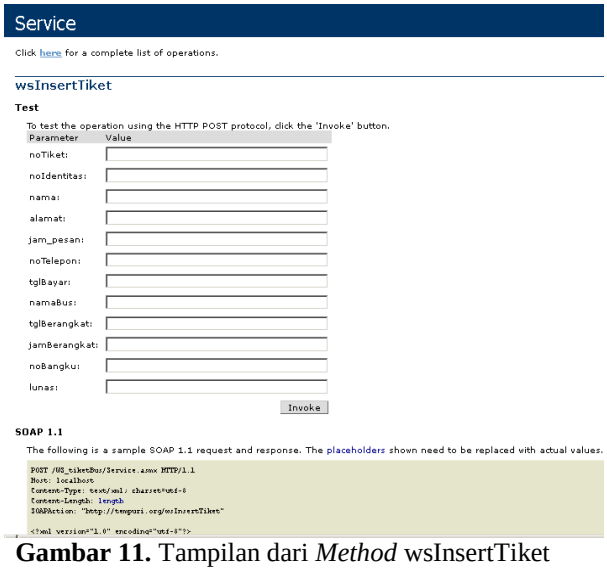
Untuk menguji *web method* dapat mengklik *invoke* dan akan menampilkan suatu *file* XML yang berisikan data user yang diinputkan pada aplikasi admin seperti gambar 10.


```
- <DataSet>
  - <xs:schema id="NewDataSet">
    - <xs:element name="NewDataSet" msdata:IsDataSet="true" msdata:UseCurrentLocale="true">
      - <xs:complexType>
        - <xs:choice minOccurs="0" maxOccurs="unbounded">
          - <xs:element name="USER">
            - <xs:complexType>
              - <xs:sequence>
                  <xs:element name="Nama_x0020_Agen" type="xs:string" minOccurs="0"/>
                  <xs:element name="Nama_x0020_Pemilik" type="xs:string" minOccurs="0"/>
                  <xs:element name="Alamat" type="xs:string" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  - <diffgr:diffgram>
    - <NewDataSet>
      - <USER diffgr:id="USER1" msdata:rowOrder="0">
          <Nama_x0020_Agen>admin</Nama_x0020_Agen>
          <Nama_x0020_Pemilik>suroso</Nama_x0020_Pemilik>
          <Alamat>jl wahab syharani</Alamat>
        </USER>
      - <USER diffgr:id="USER2" msdata:rowOrder="1">
          <Nama_x0020_Agen>delta</Nama_x0020_Agen>
          <Nama_x0020_Pemilik>Budi</Nama_x0020_Pemilik>
          <Alamat>Jl. pemuda 2</Alamat>
        </USER>
      </NewDataSet>
```


Gambar 10. Tampilan XML *file* wsGetAllUser

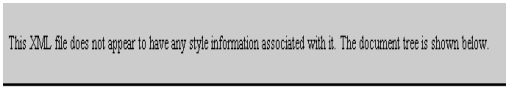
Web Method wsInsertTiket

Web method wsInsertTiket merupakan *method* dari *web service* untuk layanan pemesanan tiket dan penginputan datanya dilakukan di aplikasi yang memanggilnya. Parameter-parameter yang disediakan oleh *method* tersebut adalah: noTiket, noIdentitas, nama, alamat, jam_pesan, noTelepon, tglBayar, namaBus, tglBerangkat, jamBerangkat, noBangku, lunas. Parameter-parameternya dapat dilihat pada gambar 11.



Gambar 11. Tampilan dari *Method* wsInsertTiket

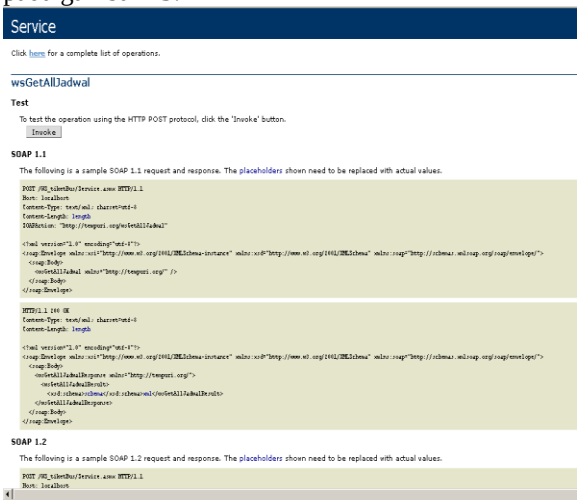
Button invoke berfungsi untuk melihat hasil data yang di-*input*, jika data yang diinputkan benar, maka akan tampil *file* XML seperti gambar 4.13,dan jika data yang diinputkan salah maka akan bernilai *false*.



Gambar 12. Tampilan dari XML *file* wsInsertTiket

Web Method wsGetAllJadwal

Web method wsGetAllJadwal merupakan *method* dari *web service* untuk layanan melihat jadwal keberangkatan. Pada *web method* wsGetAllJadwal hanya terdapat *button invoke* untuk menampilkan *file* XML dari *method* tersebut. Tampilan *method* wsGetAllJadwal dapat dilihat pada gambar 13.



Gambar 13. Tampilan dari *Method* wsGetAllJadwal

Untuk menguji *web method* dapat mengklik *invoke* dan akan menampilkan suatu *file* xml yang berisikan data jadwal keberangkatan yang diinputkan pada aplikasi admin seperti gambar 14.

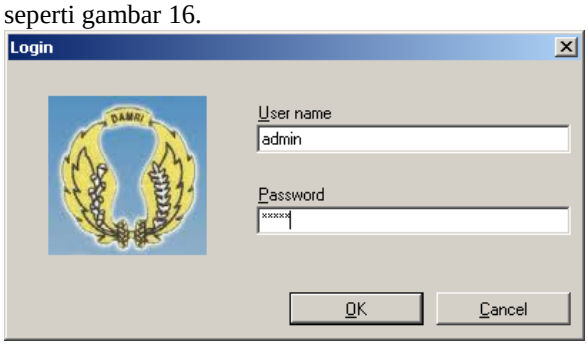

```
- <DataSet>
  - <xs:schema id="NewDataSet">
    - <xs:element name="NewDataSet" msdata:IsDataSet="true" msdata:UseCurrentLocale="true">
      - <xs:complexType>
        - <xs:choice minOccurs="0" maxOccurs="unbounded">
          - <xs:element name="JADWAL">
            - <xs:complexType>
              - <xs:sequence>
                  <xs:element name="jam_berangkat" type="xs:string" minOccurs="0"/>
                  <xs:element name="no_bus" type="xs:string" minOccurs="0"/>
                  <xs:element name="no_supir" type="xs:string" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
          </xs:choice>
        </xs:complexType>
      </xs:element>
    </xs:schema>
  - <diffgr:diffgram>
    - <NewDataSet>
      - <JADWAL diffgr:id="JADWAL1" msdata:rowOrder="0">
          <jam_berangkat>08:15</jam_berangkat>
          <no_bus>DAMRI1</no_bus>
          <no_supir>001</no_supir>
        </JADWAL>
      - <JADWAL diffgr:id="JADWAL2" msdata:rowOrder="1">
          <jam_berangkat>10:45</jam_berangkat>
          <no_bus>DAMRI2</no_bus>
          <no_supir>002</no_supir>
        </JADWAL>
      - <JADWAL diffgr:id="JADWAL3" msdata:rowOrder="2">
```


Gambar 15. Tampilan dari XML *file* wsGetAllJadwal

Aplikasi Pada Desktop

Setiap *user* aplikasi *desktop* nantinya diberikan *user name* dan *password* untuk bisa

mengakses data yang ada pada aplikasi penjualan tiket bus DAMRI. Antara *user* admin dan agen memiliki *user name* dan *password* berbeda karena mereka mempunyai hak akses yang berbeda sesuai tugasnya masing-masing. Adapun tampilannya seperti gambar 16.

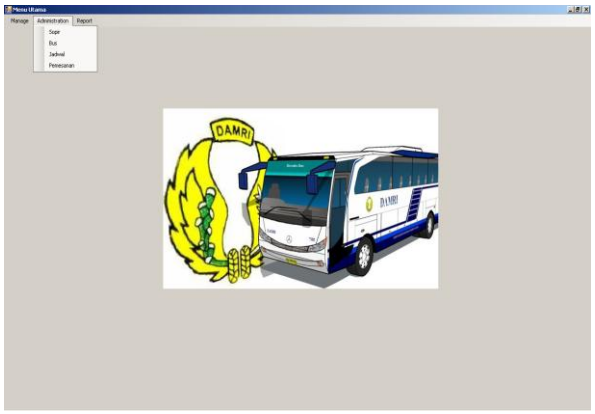


Gambar 16. Tampilan *form Login*

Menu Utama

Ada perbedaan antara menu utama aplikasi untuk admin dan masing-masing agen. Untuk halaman admin desain menu adalah statis tidak berubah–ubah, mencakup semua menu yang ada. Sedangkan menu untuk agen didesain secara dinamis, dimana tampilan akan berubah sesuai dengan hak akses pengguna aplikasi.

Pada gambar 17 menampilkan menu utama untuk admin dimana semua menu dapat diakses oleh admin, karena admin mempunyai hak akses penuh pada sistem.



Gambar 17. Tampilan Menu Utama aplikasi *desktop* pada admin

Berbeda dengan tampilan aplikasi admin, tampilan aplikasi pada agen pada gambar 18 hanya menampilkan sebagian menu, diantaranya adalah menu jadwal dan menu pemesanan sedangkan menu yang lainnya terkunci.

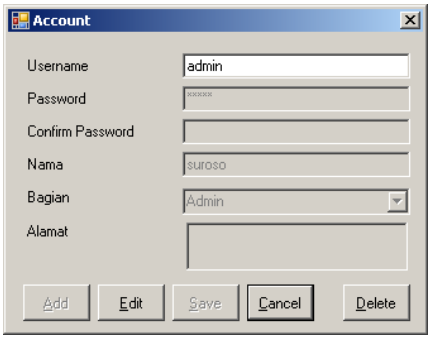
Gambar 18. Tampilan Menu Utama pada agen

Entri Master Data

Pada bagian *entri* data Perusahaan DAMRI, terdapat empat data *master* yang di*entri* oleh *administrator*. Yaitu *entri* data *account*, *entri* data supir, *entri* data bus, dan *entri* jadwal keberangkatan.

1. *Entri Account*

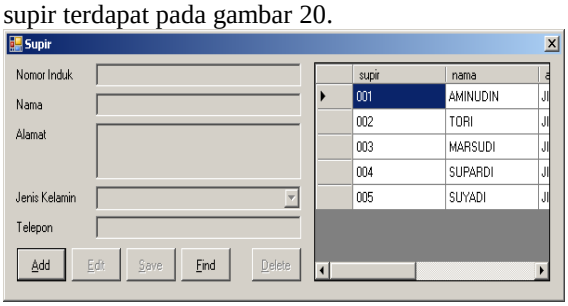
Pada menu *account* admin meng-*entri acoount* admin dan agen yang telah ditunjuk untuk melayani penjualan tiket. Data yang di-*entri* adalah *Username* admin atau agen, *Password*, Nama, Bagian, Alamat. Adapun data *account* terdapat pada gambar 19.



Gambar 19. *Form Account*

2. *Entri* Supir

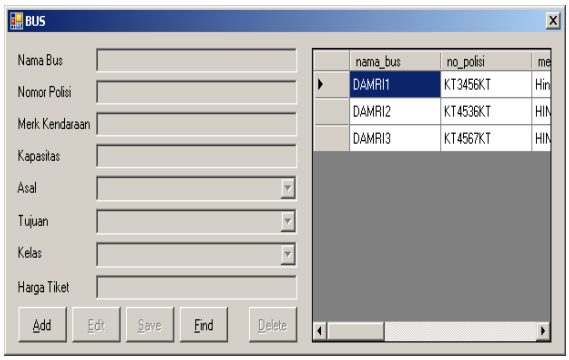
Menu supir berfungsi untuk meng-*entri* data supir yang akan mengendarai bus yang telah ditunjuk oleh pihak perusahaan DAMRI. Data yang di-*entri* diantaranya adalah Nomor Induk, Nama, Alamat, Jenis Kelamin, dan Telepon. Adapun data supir terdapat pada gambar 20.



Gambar 20. *Form* supir

3. *Entri* Bus

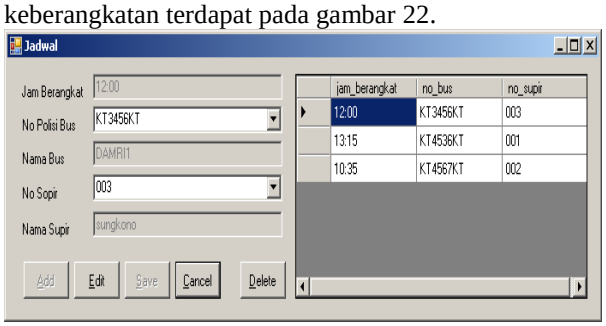
Menu bus berfungsi untuk meng-*entri* data bus yang dimiliki perusahaan DAMRI sesuai dengan trayek tujuan masing-masing. Data bus yang dientri adalah Nama bus, Nomor polisi, Merk kendaraan, Kapasitas, Asal, Tujuan, Kelas, dan Harga tiket. Ada pun data bus terdapat pada gambar 21.



Gambar 21. *Form* Bus

4. *Entri* Jadwal

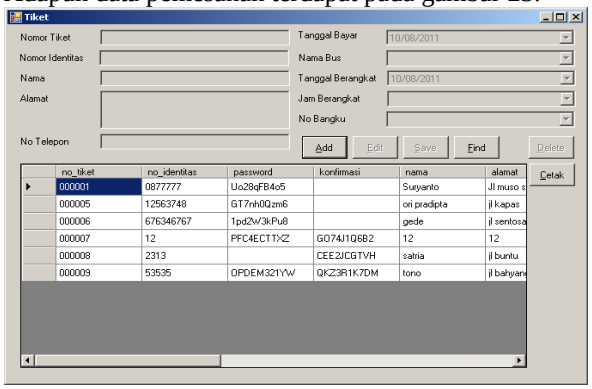
Menu jadwal berfungsi untuk meng-*entri* jadwal keberangkatan bus yang nantinya dapat di akses oleh agen dan pengunjung *website*. Data tersebut adalah Jam keberangkatan, No polisi bus, Nama bus, No supir, Nama supir. Adapun jadwal keberangkatan terdapat pada gambar 22.



Gambar 22. *Form* Jadwal

Entri Pemesanan

Menu pemesanan berfungsi untuk mencatat data penumpang, serta mencetak tiket. *Entri* pemesanan dapat dilakukan oleh admin dan agen. Adapun data pemesanan terdapat pada gambar 23.



Gambar 23. *Form* Pemesanan

Report

Hasil akhir dari aplikasi ini adalah berupa *report*/laporan yang akan diberikan kepada perusahaan DAMRI. Tampilan menu *report* dibuat dengan memuat beberapa kriteria sesuai yang diinginkan, dan mengisikan kata kunci, seperti pada gambar 24. Tampilan laporan dalam *crystal report viewer*, dan dapat langsung dicetak.

Gambar 24. Tampilan *Report*

Aplikasi Pada Web

Web PERUM DAMRI berfungsi sebagai pelayanan kepada calon penumpang yang ingin mengetahui seluk beluk tentang PERUM DAMRI, jadwal keberangkatan bus, *reservasi* pemesanan tiket dan mencetak tiket. Menu yang terdapat pada *website* meliputi *Home*, Visi dan Misi, Maksud dan Tujuan, Jadwal, Pemesanan, Agen dan *Form login* untuk mencetak tiket dan konfirmasi pembayaran.

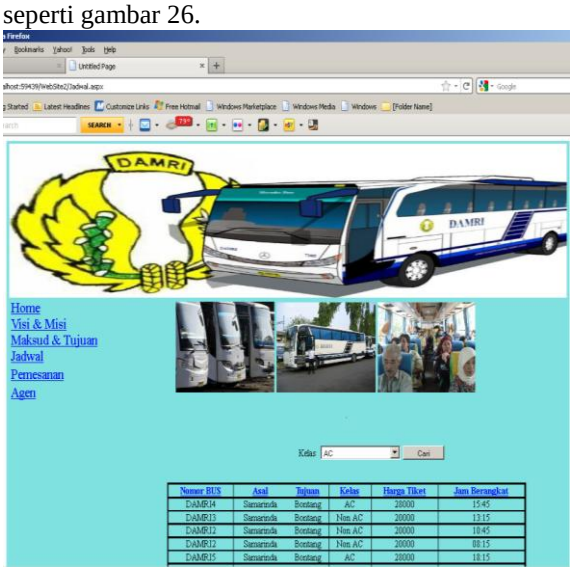
Halaman Home

Halaman *home* merupakan tampilan awal dari *website* PERUM DAMRI dimana berisikan tentang sejarah singkat awal berdirinya PERUM DAMRI. Pada menu *home* pemesan dapat melakukan cetak tiket dan konfirmasi pembayaran di *form login* yang telah disediakan seperti gambar 25.

Gambar 25. Tampilan halaman *Home*

Halaman Jadwal

Halaman jadwal berfungsi untuk memanggil dan menampilkan tabel jadwal keberangkatan bus, yang telah di-*input* oleh pihak admin di aplikasi *desktop*. Adapun tampilannya seperti gambar 26.

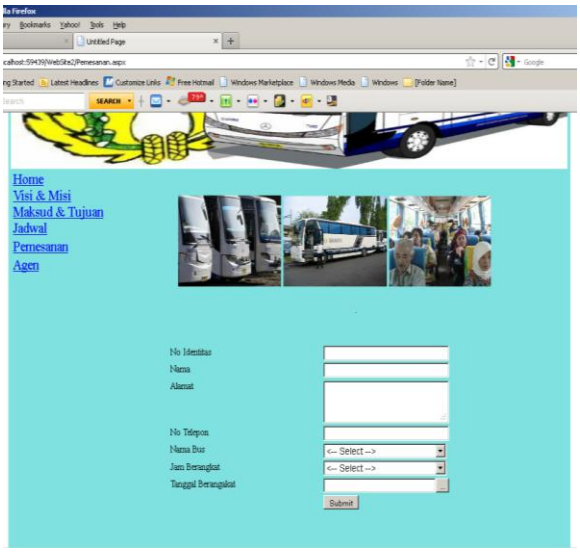


Gambar 26. Tampilan halaman Jadwal

Halaman Pemesanan

Halaman pemesanan berfungsi untuk menampilkan *form* pemesanan tiket seperti gambar 27. Dimana pemesan (pembeli tiket/*user*) mengisikan data diri di *form* pemesanan untuk *reservasi* tiket dan akan mendapatkan nomor tiket untuk konfirmasi pemesanan, setelah melakukan pembayaran melalui rekening yang telah ditentukan pemesan akan mendapatkan kode konfirmasi.

Setelah itu *user* dapat melakukan konfirmasi pemesanan pada halaman *home* atau langsung konfirmasi pada halaman pemesanan dengan menginputkan nomor tiket dan kode konfirmasi, selanjutnya *user* akan mendapatkan *password* untuk cetak tiket. *User* dapat mencetak tiket pada halaman *home* dengan meginputkan nomor tiket dan *password* yang telah diberikan pada *form* cetak tiket.



Gambar 27. Tampilan halaman Pemesanan

Halaman Agen

Halaman agen berfungsi untuk menampilkan informasi kontak agen dan admin yang diinputkan oleh admin pada aplikasi *desktop*, informasi ini berfungsi untuk membantu *user* mengetahui agen yang bisa melayani penjualan tiket bus DAMRI. Adapun informasi data agen terdapat pada gambar 28.

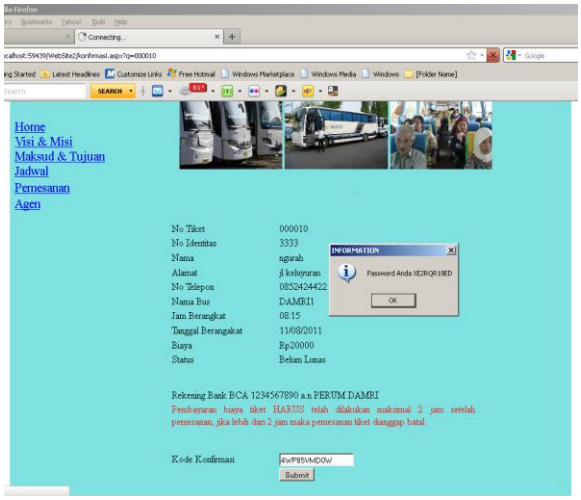
Gambar 28. Tampilan halaman Agen

Proses Mencetak Tiket

Pada layanan *web* penjualan tiket bus DAMRI ini disediakan sebuah *form* untuk mencetak tiket, *user* dapat mencetak tiket setelah mengisi data pada *menu* pemesanan, dan telah melakukan pembayaran. Dimana setelah melakukan pembayaran *user* akan mendapatkan sebuah kode konfirmasi untuk konfirmasi pembayaran.

Konfirmasi pembayaran dapat dilakukan langsung di *website* seperti pada gambar 29, konfirmasi pembayaran harus dilakukan dua jam setelah melakukan pemesanan, jika melewati batas

waktu yang diberikan maka pemesanan tiket secara otomatis akan dibatalkan.



Gambar 29. Tampilan halaman konfirmasi

Jika konfirmasi berhasil, *user* akan mendapat *password* untuk men-cetak tiket. Selanjutnya pembeli tiket dapat mencetak tiket dengan menginputkan nomor tiket dan *password* di *form* cetak tiket di halaman *home* seperti gambar 30.

Gambar 30. Tampilan untuk cetak tiket

KESIMPULAN

Berdasarkan hasil pengujian sistem yang dilakukan, dapat diambil kesimpulan beberapa hal antara lain :

1. Telah dibangun aplikasi *web service* sebagai penyedia layanan penjualan tiket bus DAMRI dan sebagai interfacenya dibangun juga aplikasi *desktop* dan *website*.
2. Berdasarkan pengujian sistem, proses penjualan tiket dan pengelolaan datanya menjadi lebih cepat dan efisien. Melalui *web service*, perusahaan DAMRI dan Agen tidak perlu membuat jaringan komputer pribadi, sehingga proses pertukaran data lebih sederhana. Yang dikelola secara bersama melalui sistem *middleware* pada *web service*.

DAFTAR PUSTAKA

[1] Angga, I, P. 2010. ***Perkembangan Teknologi Web***. http://www.balionlucky.co.cc/2010/05/http-dan-www.html.(12 Oktober 2010, 16:54)

[2] Angga, I, P. 2002. ***HTTP dan WWW*** . http://www.balionlucky.co.cc/2010/05/http-dan-www .html. (13 Oktober 2010, 19:54)

[3] Anggita. 2007. ***Web Sevice***. http://inherent.brawijaya.ac.id/portal/?hlm=artikel_detail&id=28.(23 September 2010, 19:00)

[4] Ilham. 2010. ***PHP Web Service.*** http://blog unsri.ac.id/userfiles/PHP%20WEB%2 SERVICE. doc. (23 September 2010, 19:44).

[5] Salam, U, A. 2010. ***Implementasi Web Service.*** http://dazti.blogspot.com /2010/01/contoh-implementasi-web-services.html. (23 September 2010, 19:00)

[6] Sholiq. 2006. ***Pemodelan Sistem Informasi Berorientasi Objek dengan UML***. Yogyakarta: Graha Ilmu.

[7] Tim Comlabs. 2006. ***Web Service dengan PHP SOAP***. http://www.comlabs.itb.ac.id/?p=112. (23 September 2010, 19:34)

[8] Tim Virashero. 2006. ***Web Service***. http://virashero.awardspace.com/?pilih=lihat&id= 38. (24 Oktober 2010, 21.45)