BAB III

ANALISA DAN PERANCANGAN SISTEM

Dalam pembuatan aplikasi ini menerapkan konsep SDLC (*Systems Development Life Cycle* (Siklus Hidup Pengembangan Sistem) yang berfungsi untuk menggambarkan tahapan-tahapan utama dan langkah-langkah dari setiap tahapan. Langkah-langkah yang akan dilakukan dalam pembuatan Pemilihan Alat Transportasi Umum Kota Surabaya menggunakan metode *Spanning Tree* pada *Smartphone Android* yaitu sebagai berikut:

3.1 Analisis Permasalahan

3.1.1 Identifikasi masalah

Kota Surabaya mengalami perkembangan yang cukup besar dalam beberapa tahun ini, terutama perkembangan dalam hal perdagangan dan pertumbuhan penduduk. Surabaya yang berkembang menjadi kota dagang dan tujuan bisnis menjadi salah satu kota tujuan bagi banyak orang untuk mencari pekerjaan, baik penduduk kota Surabaya atau bahkan pendatang dari berbagai daerah, hal ini akan berbanding lurus dengan meningkatnya tingkat kemacetan di kota Surabaya. Salah satu solusi yang untuk bisa mengurangi kemacetan adalah dengan memanfaatkan angkutan umum. Namun sebagai pendatang tentu saja kebanyakkan dari mereka belum mengetahui informasi rute angkutan umum, bahkan tidak menutup kemungkinan bahwa warga asli atau yang sudah lama menetap di Surabaya juga kurang informasi mengenai rute angkutan umum dalam kota terutama bemo dan bis kota yang merupakan salah satu faktor mengapa

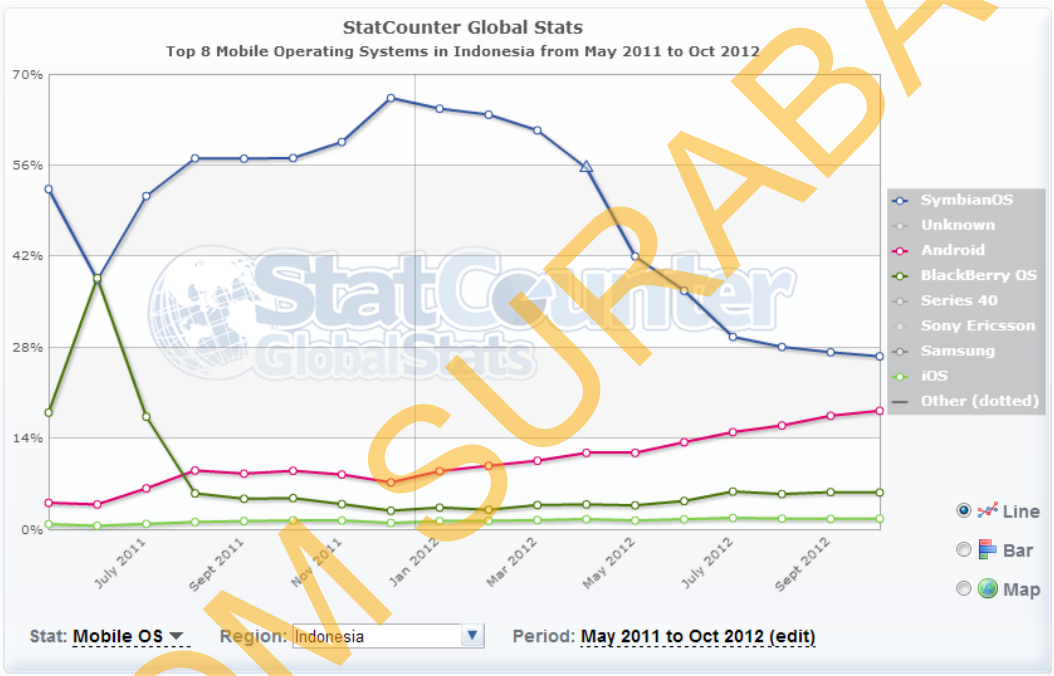
mereka enggan menggunakan angkutan umum dan lebih memilih menggunakan kendaraan pribadi. Oleh karena itu diperlukan suatu media yang bisa memberikan informasi mengenai angkutan umum, terutama angkutan umum jenis bemo dan bis kota. Dengan cukupnya informasi mengenai rute angkutan umum tersebut, calon penumpang bisa dengan tenang menggunakan angkutan umum tanpa perlu khawatir salah jurusan atau salah naik angkot.

3.1.2 Identifikasi Kebutuhan

Berdasarkan identifikasi permasalahan, dapat disimpulkan bahwa diperlukan suatu sistem yang mampu memberikan informasi angkutan umum dan juga solusi alternatif untuk pemilihan alat transportasi umum Surabaya yang bersifat *mobile* dan mudah diakses, dalam hal ini aplikasi yang dibuat menggunakan perangkat *mobile smartphone android*. Seperti yang kita ketahui, saat ini *Smartphone Android* sedang mengalami peningkatan yang sangat pesat. Dan ada beberapa faktor penting yang menyebabkan peningkatan pengguna *Android* yaitu harga *handset* yang cukup terjangkau sehingga *handset Android* bisa dengan mudah dimiliki oleh berbagai kalangan, tampilan antar muka yang cukup menarik dan tidak membosankan, dan yang terakhir adalah dikarenakan sistem operasi *Android* yang digunakan oleh banyak produsen telepon genggam seperti Sony, Samsung, Motorola dan bahkan merk lokal juga menggunakan sistem operasi *Android*.

Seperti yang dimuat di Harian Seputar Indonesia pada tanggal 27 Juni 2012, Gideon Edi Purnomo selaku Head of VAS, Aplications and Device Management Group Telkomsel menyebutkan bahwa pertumbuhan pengguna Android mencapai 15 kali lipat dibandingkan tahun 2011 atau sekitar 2,5juta

pengguna saat ini dan diperkirakan akan terus mengalami peningkatan lebih besar pada tahun depan. dimana solusi alternatif tersebut didapat dari pertimbangan jarak dan biaya. Hal ini juga dapat dilihat dari hasil survey *StatCounter Global State* yang menunjukkan jumlah peningkatan pengguna *Smartphone Android* pada periode Mei 2011 hingga Oktober 2012 yang cukup signifikan seperti yang terlihat pada gambar berikut.



Gambar 3.1 Grafik Peningkatan Pengguna *Smartphone* Berdasarkan *Mobile Operating System*

Karena beberapa pertimbangan tersebut diatas, maka penulis memilih untuk membuat aplikasi Pemilihan Alat Transportasi Umum Kota Surabaya pada *Smartphone Android.* Informasi yang diterima pengguna dari aplikasi ini nantinya berupa informasi rute atau trayek beserta estimasi biaya angkutan umum bemo dan bis yang bisa memandu pengguna untuk dapat sampai ke tempat yang akan

dituju, aplikasi ini juga menampilkan informasi jadwal keberangkatan kereta komuter, alamat dan *call center* armada taksi yang beroperasi di kota Surabaya.

3.2 Rancangan Sistem

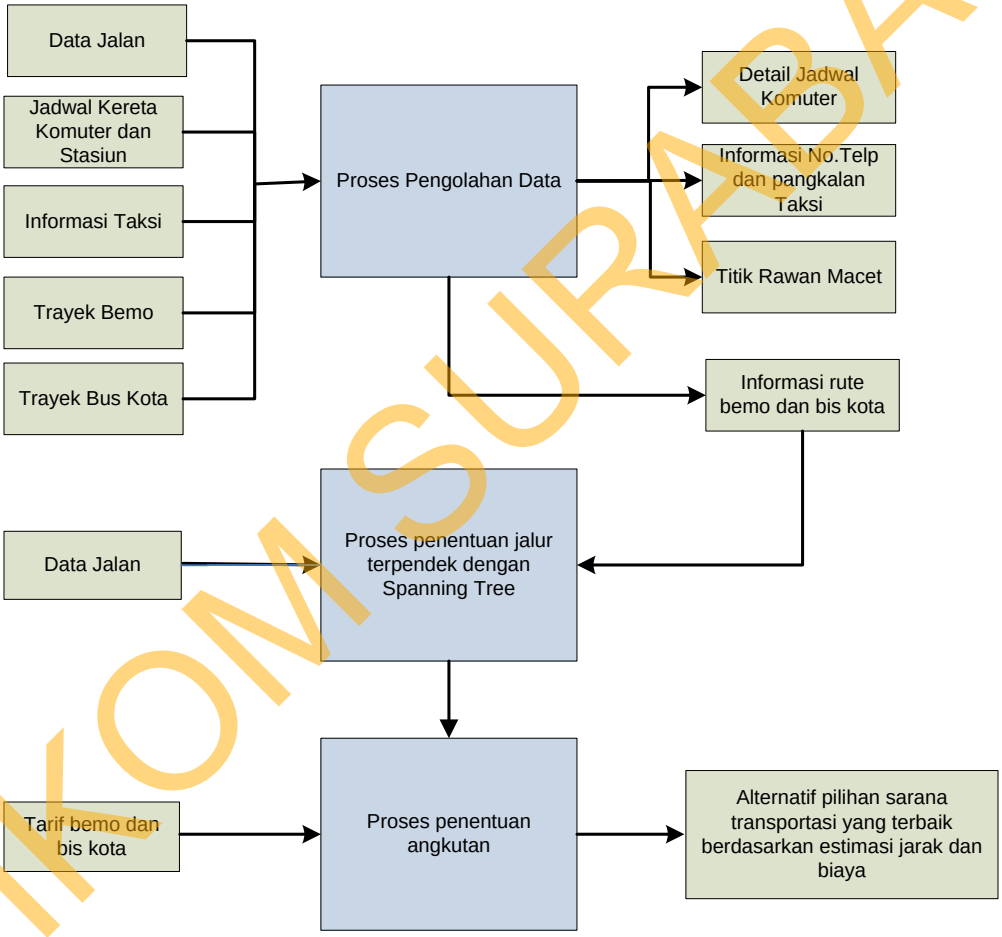
Gambaran umum arsitektur aplikasi Pemilihan Alat Transportasi Umum Kota Surabaya Menggunakan Metode *Spanning Tree* pada *Smartphone Android* dapat dilihat pada gambar 3.2 berikut.

Gambar 3.2. Gambaran Umum Sistem Pemilihan Alat Transportasi Umum Surabaya

Terdapat dua buah aplikasi yang akan dibangun seperti yang terlihat pada gambar 3.2, yaitu *mobile application* dan *web application.* Pengguna meminta informasi rute angkutan dengan memasukkan nama jalan berangkat dan nama jalan tujuan. Informasi nama jalan berangkat dan nama jalan tujuan tersebut kemudian diproses untuk mencari rute angkutan yang terpendek. Setelah proses pencarian rute terpendek selesai, sistem melakukan pencocokan rute tersebut dengan database trayek angkutan kota bemo dan bis kota untuk menginisialisasi trayek-trayek apa saja yang sesuai. Proses ini akan berhenti setelah sistem mengirim informasi yang telah diproses.

Pada tugas akhir ini sistem yang dibuat nantinya dapat digunakan oleh semua orang untuk memilih alat transportasi umum sesuai dengan tujuan.

Informasi yang diberikan berupa solusi alternatif pemilihan alat trasnportasi umum berdasarkan pertimbangan jarak dan biaya. Sistem akan memberikan beberapa alternatif alat trasnportasi yang bisa dipilih. Selain itu juga informasi yang didapat dari aplikasi ini diantaranya jadwal keberangkatan kereta api komuter dan informasi nomor operator taksi, serta beberapa titik jalan rawan kemacetan di kota Surabaya.



Gambar 3.3. Blok Diagram Sistem Pemilihan Alat Transportasi Umum Surabaya

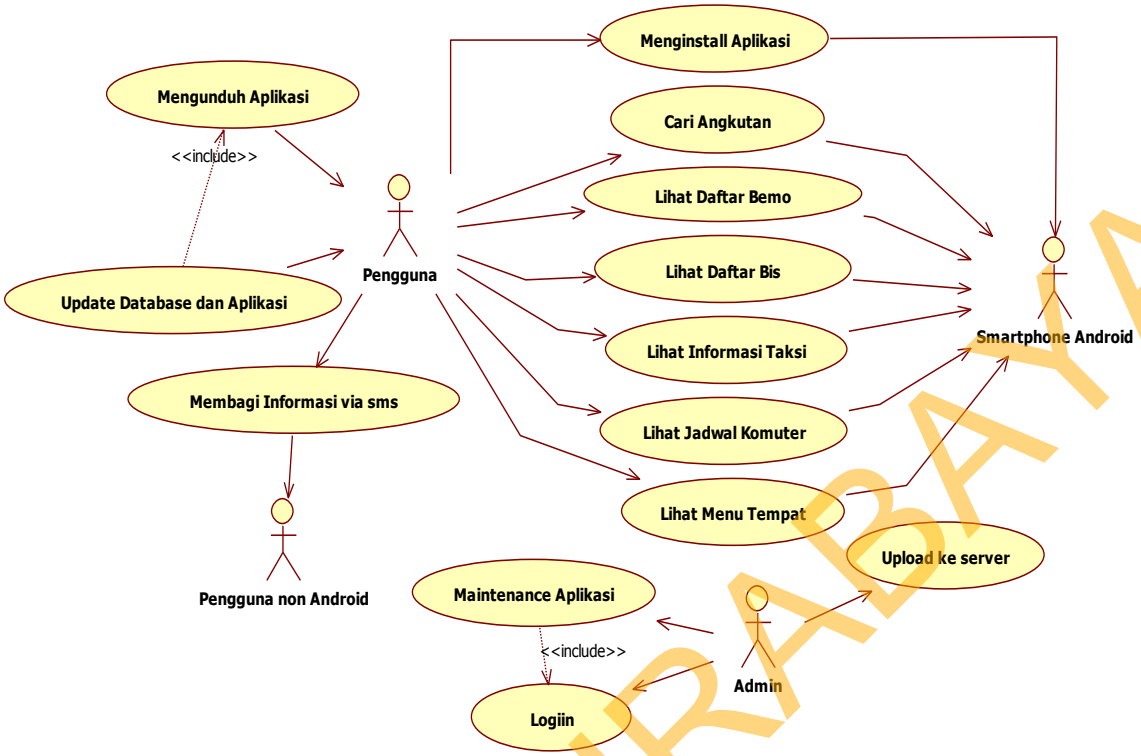
Konsep dari aplikasi ini nantinya bermula dari data-data bemo dan bis kota beserta rute jalan dan tarifnya, jadwal komuter, dan data taksi, dan *user* yang mencari informasi angkutan umum yang sesuai yang dapat mengantarkan atau

memandu *user* sampai ke tempat yang dituju. Kemudian *user* akan memberikan inputan tempat asal *user* berada dan tempat yang akan dituju, sehingga nanti informasi yang tampil adalah angkot/bemo apa saja yang bisa digunakan agar sampai tempat yang dituju. Faktor yang menjadi pertimbangan dalam menentukan rute adalah jarak yang ditempuh merupakan yang paling dekat. Dan jumlah solusi alternatif yang ditampilkan dibatasi maksimal 5 solusi terbaik, sehingga *user* tidak dibingungkan dengan terlalu banyaknya pilihan. Dan untuk informasi lain yang bisa didapatkan pengguna adalah jadwal keberangkatan kereta api komuter dan informasi alamat beserta nomor telepon taksi. Dengan cara ini penyampaian informasi mengenai transportasi kota Surabaya akan lebih efektif.

Pada saat aplikasi dijalankan, *user* memberikan inputan lokasi berangkat saat ini dan tempat yang akan dituju. Aplikasi ini nantinya akan menyediakan beberapa menu diantaranya adalah untuk mencari jenis angkutan umum yang sesuai dengan kebutuhan untuk sampai ketempat yang akan dituju, output nantinya akan berupa alternatif angkutan umum (bemo atau bis kota) dan juga estimasi biaya yang diperlukan. Dan juga menu untuk menampilkan jadwal keberangkatan kereta komuter. Kemudian menu *call taksi*, yaitu menu yang menampilkan berupa daftar *call center* armada taksi yang beroperasi di Surabaya.

3.3 *Use Case Diagram*

Use case diagram digunakan untuk menspesifikan apa yang dapat dilakukan oleh sistem atau untuk menspesifikan kebutuhan fungsional utama dari sistem. Berikut akan dijelaskan *use case diagram* untuk sistem.



Gambar 3.4. *Use Case Diagram* Sistem Pemilihan Alat Transportasi

Berikut adalah penjelasan singkat *use case* yang dimiliki oleh aplikasi.

Tabel 3.1 Penjelasan singkat *use case diagram*

Nama Use Case	Deskripsi
Menggunduh Aplikasi	Proses yang digunakan untuk melakukan unduh aplikasi dari *web site*
Menginstall Aplikasi	Proses yang menangani install aplikasi pada perangkat *mobile*
Cari Angkutan	Proses yang menangani pencarian angkutan yang menghasilkan *output* solusi alternatif angkutan
Lihat Daftar Bemo	Proses yang menangani atau menampilkan daftar bemo, rute dan tarifnya
Lihat Daftar Bis	Proses yang menangani atau menampilkan daftar bis, rute dan tarifnya
Lihat Informasi Taksi	Proses yang menangani atau menampilkan detail taksi mulai dari alamat, no.telp dan gambar taksi
Lihat Jadwal Komuter	Proses yang menangani dan menampilkan jadwal keberangkatan dan kedatangan komuter
Lihat Menu Tempat	Menampilkan lokasi beberapa tempat yang tersimpan di database

Tabel 3.1 Penjelasan singkat *use case diagram* (lanjutan)

Nama Use Case	Deskripsi
Update Database dan Aplikasi	Proses yang menangani pembaruan *database* atau perubahan.
Maintenance Aplikasi	Proses yang memelihara atau menjaga agar aplikasi tetap berjalan dengan baik.
Upload ke Server	Proses yang menangani penambahan data ke *web server*
Membagi Informasi	Proses yang dimana informasi mengenai angkutan disebarkan atau dibagikan (*share*) menggunakan sms ke perangkat handphone lain.

Pertama-tama pengguna menggunduh aplikasi, kemudian menginstall aplikasi pada perangkat *Smartphone Android.* Setelah aplikasi di install, pengguna dapat langsung menggunakannya tetapi pada saat pertama kali menggunakan aplikasi akan terlebih dahulu mengambil data dari *web server*. Untuk dapat memperoleh informasi solusi alternatif angkutan mana yang bisa digunakan agar sampai ke tempat atau jalan yang diinginkan, pengguna harus memberikan *inputan* lokasi atau nama jalan berangkat dan tujuan yang diinginkan pengguna untuk kemudian *inputan* tersebut diproses. Pada aplikasi ini juga terdapat menu jadwal komuter dan informasi taksi, informasi yang ditampilkan pada kedua menu ini berupa alamat dan nomor telepon taksi dan juga jadwal komuter. Dan untuk update bisa dilakukan secara otomatis selama *handset* berada dalam keadaan *online*.

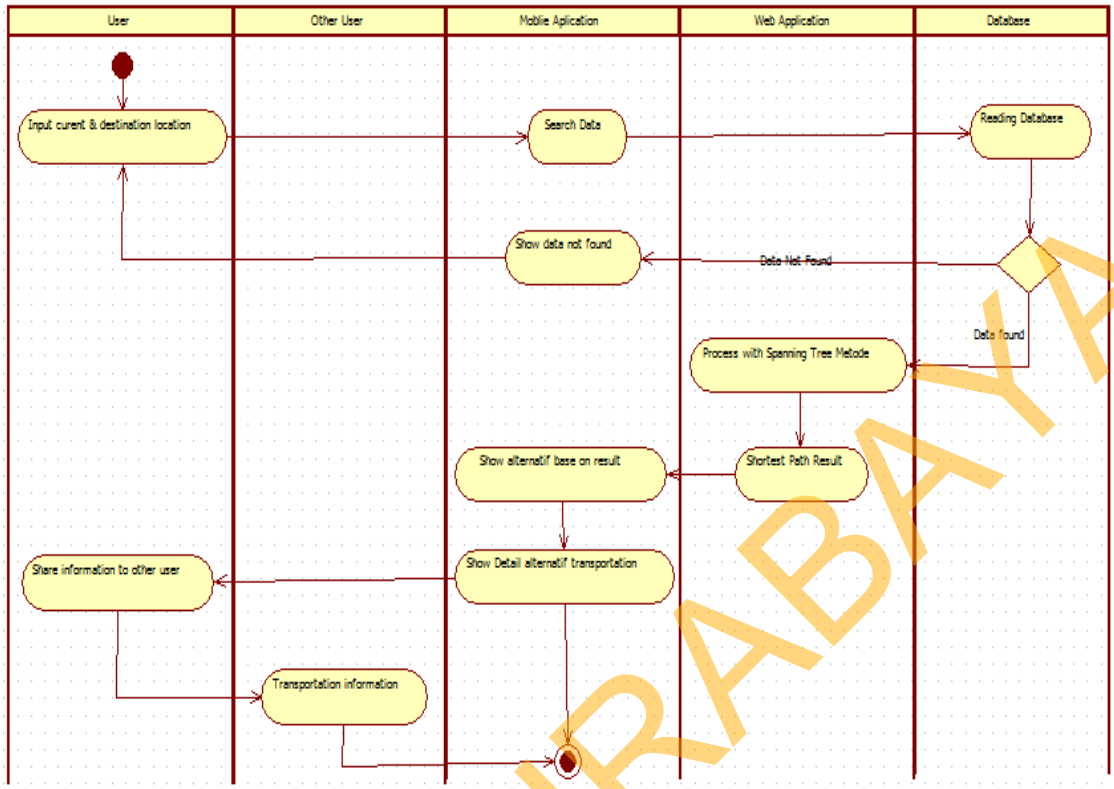
3.4 *Activity Diagram*

Activity Diagram digunakan untuk memodelkan aliran kerja proses dalam bentuk simbol untuk menspesifikasikan bagaimana sistem akan mencapai tujuan. *Activity Diagram* adalah salah satu bentuk diagram UML yang paling

mudah dimengerti dikarenakan diagram ini memiliki simbol yang menyerupai simbol *flowchart*, yang sangat berguna untuk menerangkan langkah-langkah proses ke pihak lain.

A. *Activity Diagram* untuk Proses Cari Angkutan

Proses dimulai ketika *user m*embuka menu cari angkutan pada *mobile aplication, k*emudian akan tampil dafar nama jalan atau rute angkutan yang tersimpan pada database. Untuk melakukan pencarian atau mendapatkan informasi rute angkutan, pertama u*ser h*arus memasukkan lokasi nama jalan *user* berada dan lokasi jalan tujuan yang diinginkan. Untuk nama jalan atau lokasi yang di inputkan hanya bisa yang terdapat pada database, artinya tidak semua daerah di Surabaya yang dapat dilakukan pencarian pada aplikasi. Kemudian kedua inputan tersebut akan diproses oleh sistem yang ada di *web application* menggunakan metode *Spanning Tree*, kemudian hasil dari proses *Spanning Tree* tersebut digunakan untuk mencari angkutan umum (bemo dan bis) apa yang bisa digunakan pengguna. Hasil yang ditampilkan nantinya berupa pilihan solusi alternatif angkutan umum dan juga pada detail solusi alternatif dapat dilihat berapa estimasi biaya dan jarak yang diperlukan untuk dapat sampai ke tujuan. Gambaran dari proses tersebut dapat dilihat pada gambar 3.5.

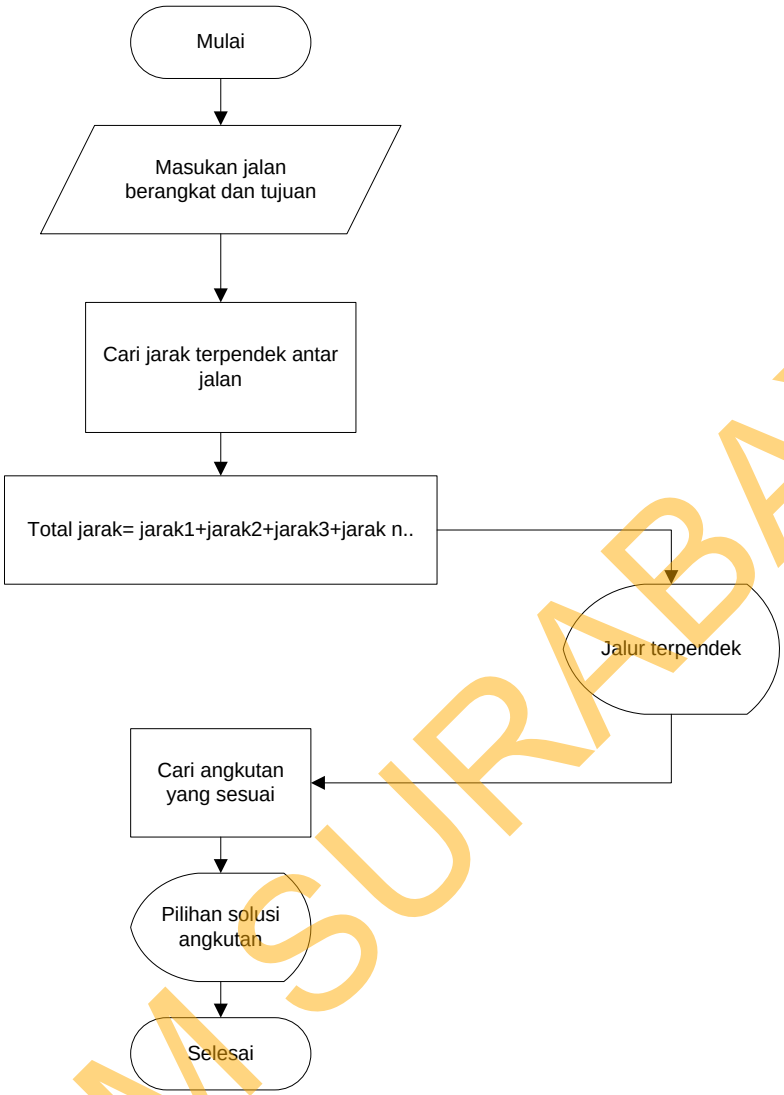


Gambar 3.5 *Activity diagram* untuk proses "Cari Angkutan dan Detail Informasi"

Dan pada gambar 3.6 berikut ini adalah *flowchart* penerapan Metode *Spanning Tree* untuk mencari jalur terpendek yang kemudian menghasilkan *output* solusi alternatif angkutan umum kota Surabaya.

Berikut ini adalah penjelasan dari gambar 3.6 :

1. Pertama adalah menentukan inputan yang dibutuhkan untuk mendapatkan jalur terpendek. Input dibutuhkan berupa nama jalan berangkat dan tujuan.
2. Langkah selanjutnya sistem akan memeriksa apakah inputan nama jalan ditemukan atau tidak pada database.



Gambar 3.6 *Flowchart* Metode *Spanning Tree* pada Aplikasi Pemilihan Angkutan Umum Surabaya

3. Langkah selanjutnya apabila inputan sudah benar, sistem akan melakukan proses pencarian jalur terpendek antar titik atau node jalan yang saling terhubung, proses pencarian ini akan terus dilakukan sampai jalan yang dituju. Kemudian semua jarak terpendek antar node jalan yang diperoleh tadi dijumlahkan. Proses penghitungan antara lain sebagai berikut :

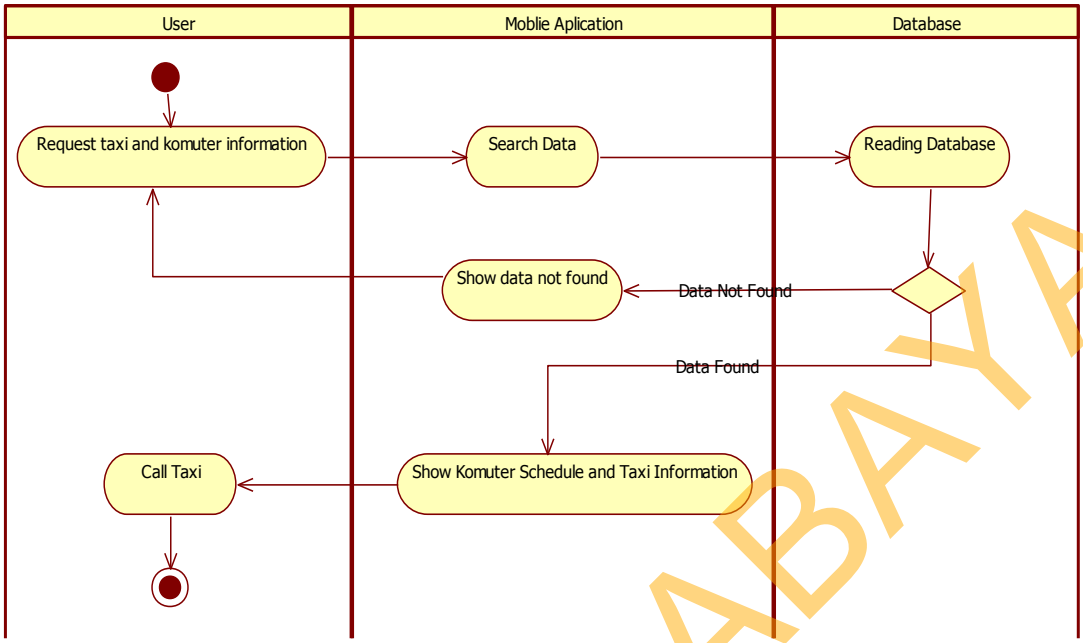
A - B = 8 **A - C = 6** A – D = 7 Jarak1 = 6	**C – E = 4** C – F = 7 C – G = 9 Jarak2 = 4	E – H = 5 E – I = 7 **E – J = 2** Jarak3 = 2

Total jarak= jarak1+jarak2+jarak3+jarak n..

4. Selanjutnya hasil yang didapat dari perhitungan tersebut berupa jalan mana saja yang harus dilewati, kemudian sistem akan menjadikan jarak terpendek dan rute tersebut sebagai acuan untuk mencari angkutan umum yang sesuai. Sehingga bisa didapatkan beberapa solusi alternatif angkutan berdasarkan hasil dari perhitungan tersebut.

B. *Activity Diagram* untuk Proses Lihat Informasi Taksi dan Komuter

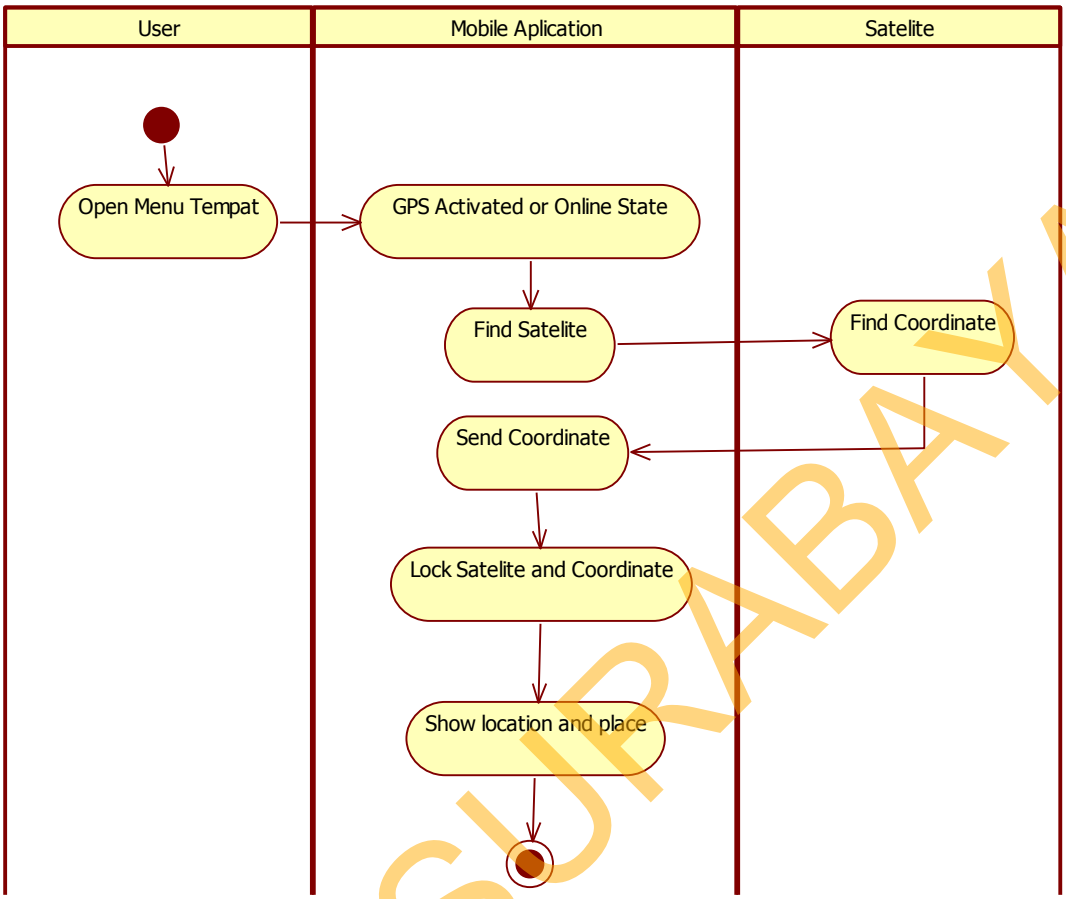
Proses dimulai ketika *user* masuk ke form menu taksi dan komuter, kemudian aplikasi akan menampilkan berbagai nama armada taksi beserta alamat dan nomor telepon. Dan untuk menu jadwal komuter aplikasi akan menampilkan nama komuter dan jadwal komuter dengan jam keberangkatan tertentu. Dan untuk menu taksi, *user* bisa melakukan panggilan ke nomor telepon armada taksi tersebut tanpa harus menyimpan nomor taksi tersebut di daftar kontak *handphone.* Gambaran dari proses tersebut dapat dilihat pada gambar 3.7.



Gambar 3.7 *Activity diagram* untuk proses "Informasi Taksi dan Komuter"

C. *Activity Diagram* untuk Menu Tempat

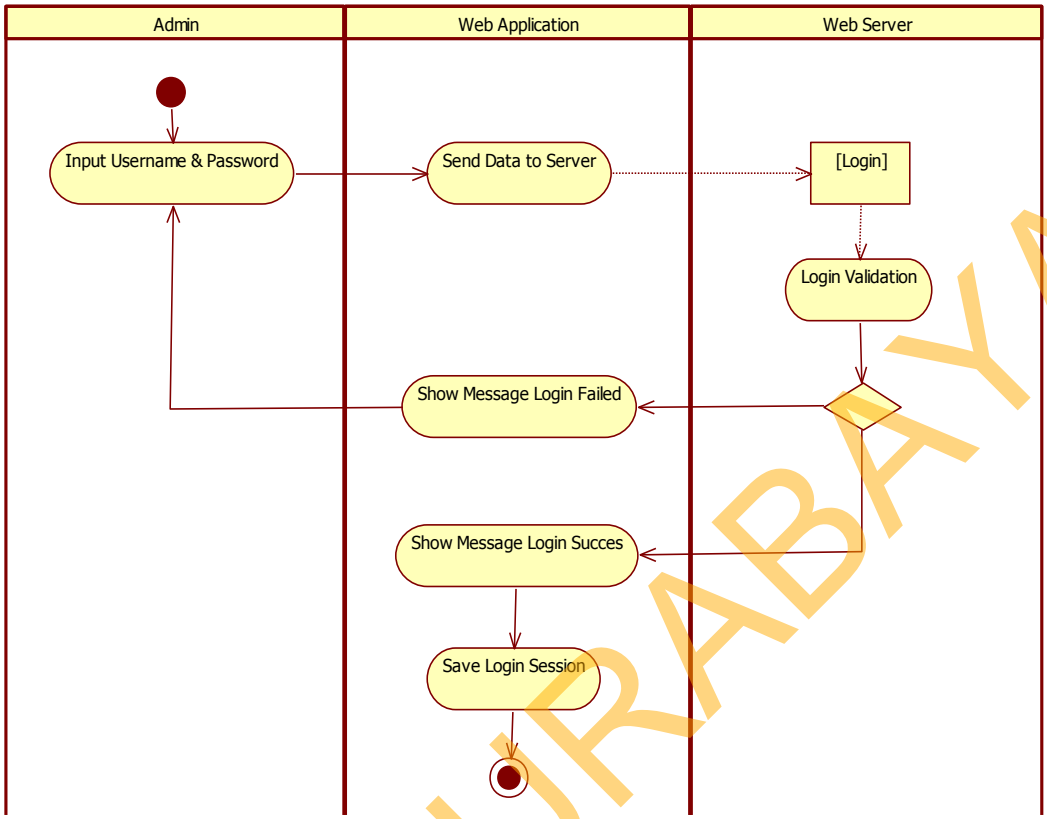
Proses dimulai ketika *user* masuk ke form menu tempat, kemudian aplikasi akan menangkap lokasi dari *user*. Proses ini dapat dilakukan secara otomatis oleh aplikasi selama fitur GPS atau dalam keadaan *online*. Dengan fitur GPS yang ada pada perangkat *mobile Android* inilah lokasi dapat *user* dapat diketahui. Selanjutnya aplikasi akan menampilkan letak atau lokasi *user* pada peta yang ada pada aplikasi beserta beberapa tempat umum yang sudah didefinisikan sebelumnya. Hal ini untuk membantu *user* mencari tempat-tempat umum seperti pusat perbelanjaan, instansi pemerintah, kampus, restaurant cepat saji dll. Gambaran dari proses tersebut dapat dilihat pada gambar 3.8.



Gambar 3.8 *Activity diagram* untuk proses "Lihat Lokasi *User*"

D. *Activity Diagram* untuk Proses Login Website

Proses dimulai ketika *user* menginputkan *username* beserta *password*. Inputan *user* akan dikirimkan oleh *web application* kepada *web server* untuk di validasi. Jika data login tidak valid maka *web application* akan menampilkan pesan pemberitahuan bahwa login gagal, tetapi jika data login valid maka *web application* akan menampilkan pesan pemberitahuan bahwa login berhasil dan *web application* akan menyimpan *login session*. Gambaran dari proses tersebut dapat dilihat pada gambar 3.9.

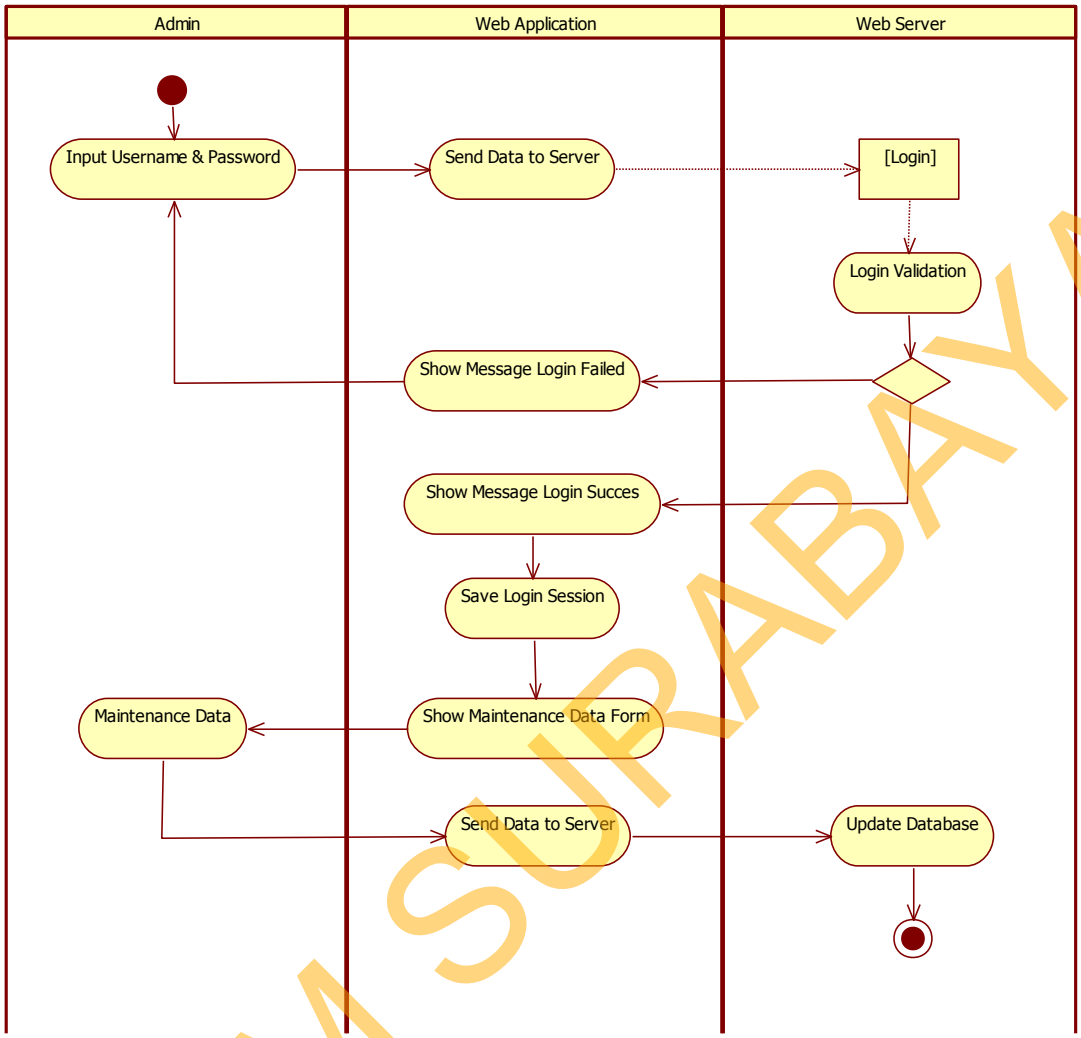


Gambar 3.9 *Activity diagram* untuk proses "Login Website"

.

E. *Activity Diagram* untuk Proses Update Data

Proses dimulai ketika *user* menginputkan *username* beserta *password.* Inputan *user* akan dikirimkan oleh *web application* kepada *web server* untuk di validasi. Jika data login tidak valid maka *web application* akan menampilkan pesan pemberitahuan bahwa login gagal, tetapi jika data login valid maka *web application* akan menampilkan pesan pemberitahuan bahwa login berhasil dan *web application* akan menyimpan *login session*. Setelah *user* melakukan proses login, maka *web application* akan menampilkan form *maintenance* atau *update data*. Pada form *maintenance* terdapat pilihan untuk menambah, menghapus dan edit. Gambaran dari proses tersebut dapat dilihat pada gambar 3.10.



Gambar 3.10 *Activity diagram* untuk proses "Update Data"

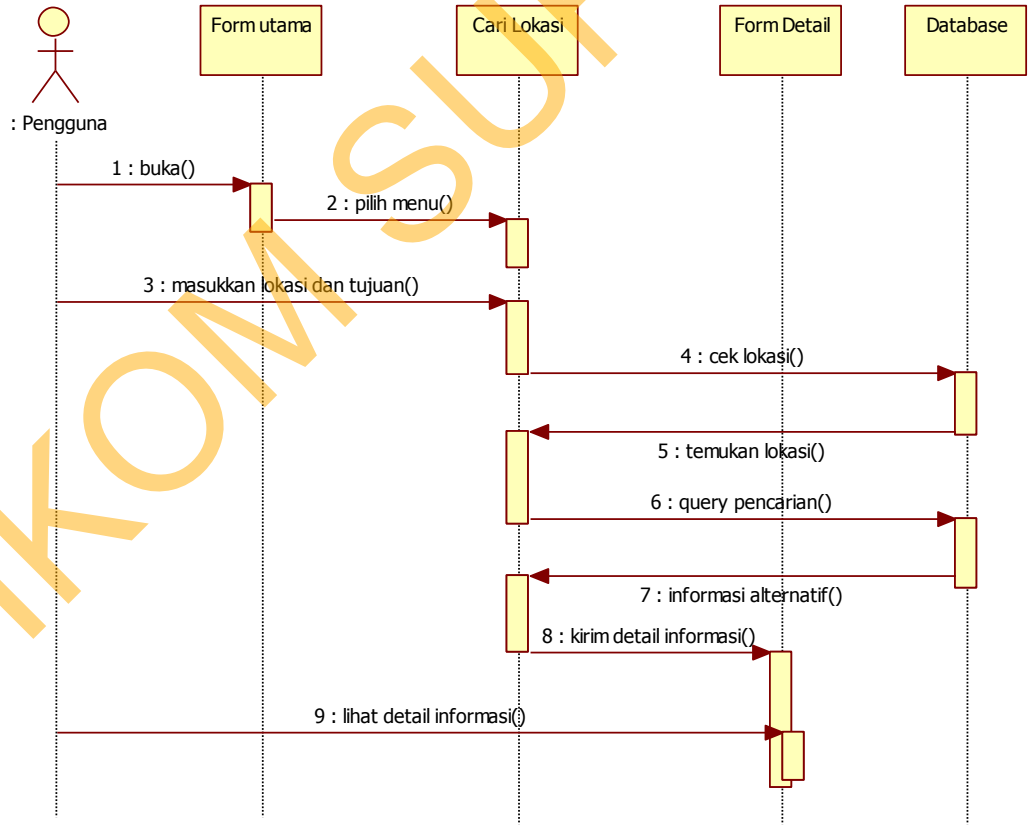
3.5 *Sequence Diagram*

Sequece Diagram digunakan untuk menggambarkan interaksi antar objek berdasarkan urutan waktu yang digambarkan dari atas ke bawah.

A. *Sequence Diagram* untuk Proses Mencari Angkutan

Proses dimulai ketika *user* membuka aplikasi dan pada tampilan awal akan keluar tampilan menu utama, kemudian *user* memilih menu cari angkutan. Untuk dapat melihat informasi alternatif angkutan yang diinginkan, *user* terlebih dahulu

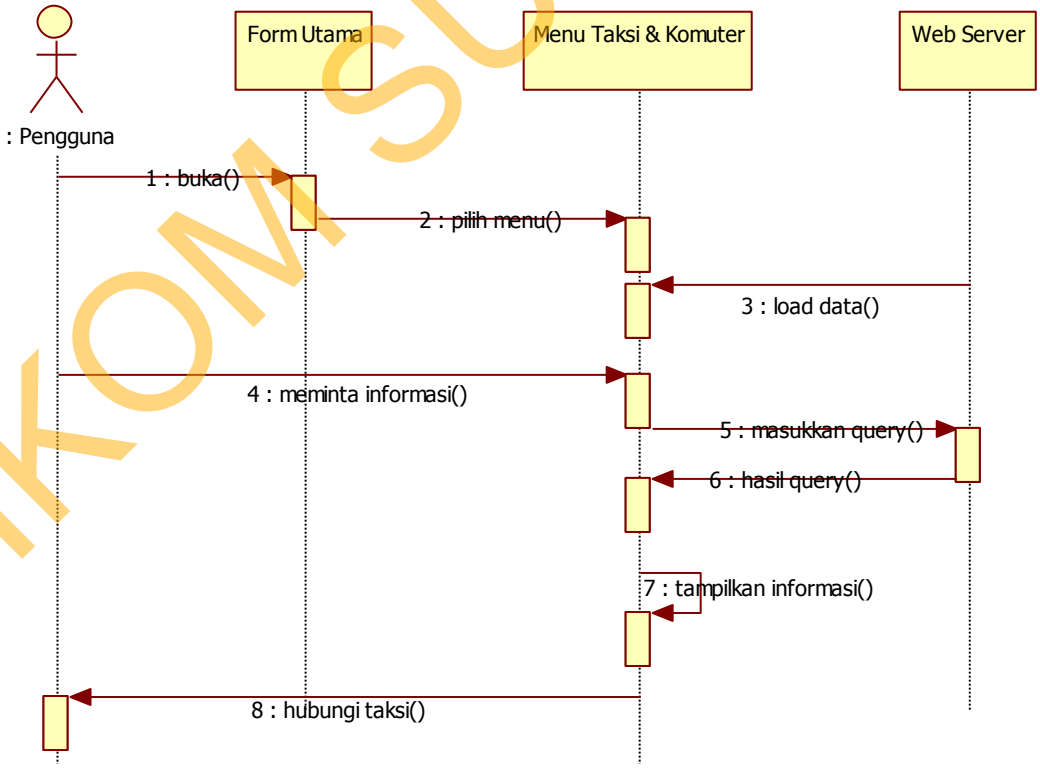
harus memasukkan nama jalan dimana *user* berada dan nama jalan yang hendak dituju pada *textbox* yang disediakan. Untuk lokasi yang bisa diproses terbatas hanya pada lokasi yang tersimpan pada database, sehingga pada saat *user* memasukkan nama jalan dimana *user* berada dan lokasi jalan yang dituju tujuan, sistem akan terlebih dahulu mengecek apakah nama jalan tersebut ada pada database. Kemudian inputan tersebut akan diproses sehingga akan muncul beberapa pilihan informasi alternatif angkutan umum beserta estimasi biaya yang dibutuhkan. Dari informasi tersebut *user* dapat memilih salah satu alternatif kemudian akan muncul detail informasi tersebut. Gambaran dari proses tersebut dapat dilihat pada gambar 3.11.



Gambar 3.11 *Sequence diagram* untuk proses "Mencari Angkutan"

B. *Sequence Diagram* untuk Proses Lihat Info Taksi dan Jadwal Komuter

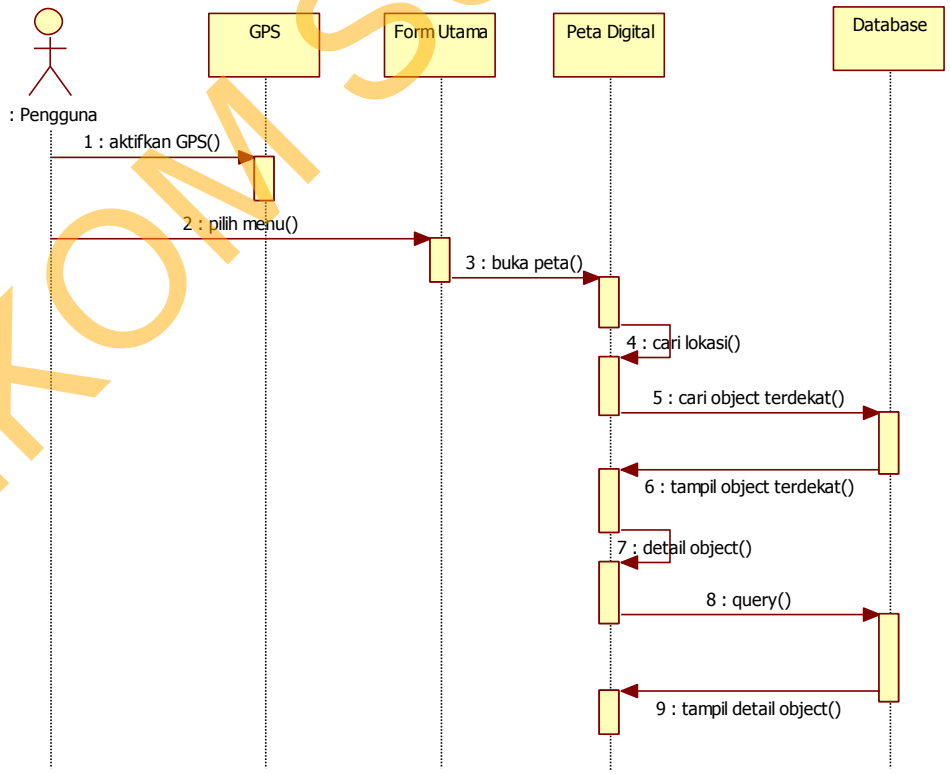
Proses dimulai ketika *user* masuk ke form menu taksi dan komuter, kemudian aplikasi akan menampilkan berbagai nama armada taksi beserta alamat dan nomor telepon. Dan untuk menu jadwal komuter aplikasi akan menampilkan nama komuter dan jadwal komuter. Untuk menu taksi, *user* bisa melakukan panggilan ke nomor telepon armada taksi tersebut tanpa harus menyimpan nomor taksi tersebut di daftar kontak *handphone*. Pada saat pertama kali menjalankan aplikasi akan membutuhkan waktu sedikit lama karena aplikasi mengambil data dari *web server* dan untuk selanjutanya data akan tersimpan sehingga untuk menu taksi dan komuter bisa digunakan dalam keadaan *offline*. Gambaran dari proses tersebut dapat dilihat pada gambar 3.12.



Gambar 3.12 *Sequence diagram* untuk proses "Info Taksi dan Jadwal Komuter"

C. *Sequence Diagram* untuk Proses Melihat Lokasi Pengguna pada Menu Tempat

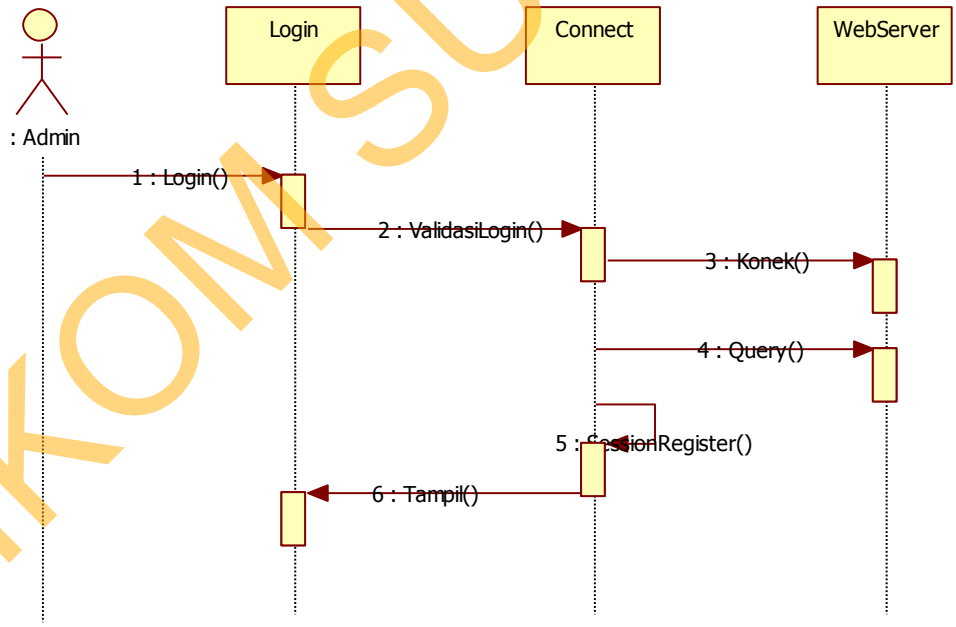
Proses dimulai ketika aplikasi menampilkan menu utama dan *user* memilih menu peta Surabaya. Sebelumnya *user* sudah harus mengaktifkan fitur GPS yang ada pada perangkat *mobile*, atau bisa juga menggunakan jaringan internet perangkat *mobile*. GPS dan jaringan internet disini berfungsi agar perangkat *mobile* dapat menangkap lokasi *user*, sehingga pada peta digital yang ada pada aplikasi dapat dilihat letak atau lokasi *user* berada. Kemudian akan muncul pula obyek-obyek yang berada pada radius tertentu dari lokasi *user* berada. Obyek-obyek yang dimaksudkan disini berupa rute angkutan umum yang sudah disimpan dalam database. Gambaran dari proses tersebut dapat dilihat pada gambar 3.13.



Gambar 3.13 *Sequence diagram* untuk proses "Mencari Lokasi Pengguna"

D. *Sequence Diagram* untuk Proses Login Website

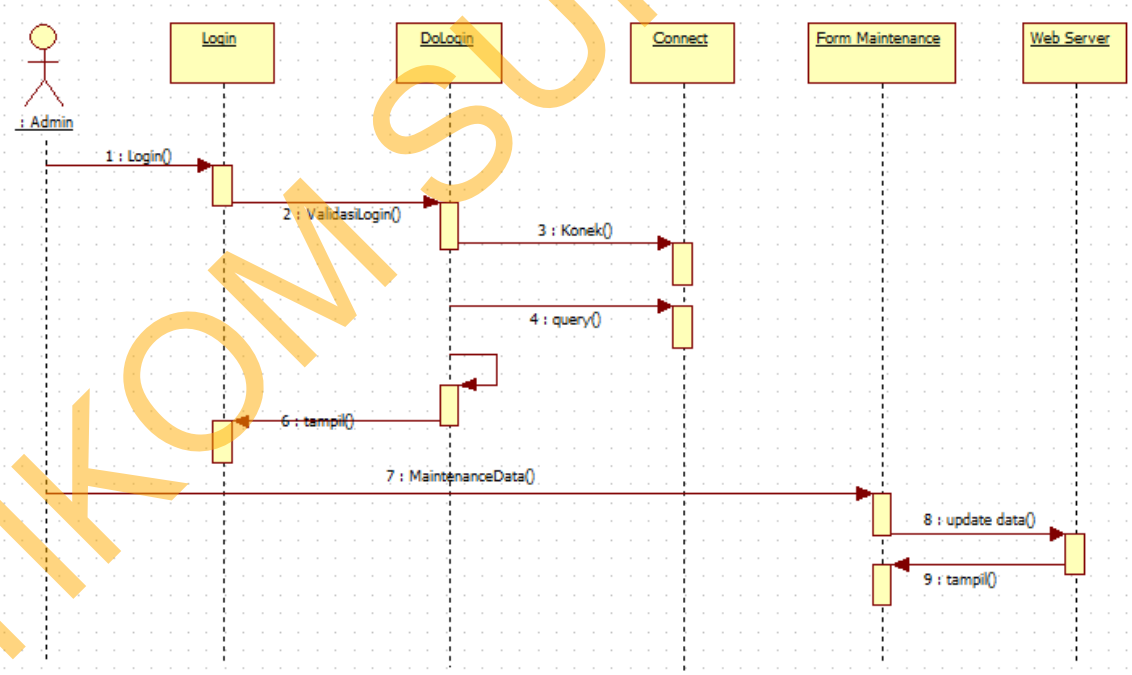
Proses dimulai ketika aplikasi menampilkan form login. Dari form login, *user* diharuskan untuk menginputkan *username* beserta *password*. Setelah *user* menekan tombol login pada form login, maka kelas login.php akan memanggil prosedur *loginValidation()*. Kelas DoLogin.php akan memvalidasi inputan *user* dan melakukan *query sql* pada tabel *user*. Jika data *user* tidak ditemukan, maka form login akan menampilkan pesan pemberitahuan bahwa login gagal. Tetapi jika data login ditemukan, maka form login akan menampilkan pesan pemberitahuan bahwa login berhasil dan *user* dapat mengakses *web application Transportasi Surabaya*. Gambaran dari proses tersebut dapat dilihat pada gambar 3.14.



Gambar 3.14 *Sequence diagram* untuk proses "Login Website"

E. *Sequence Diagram* untuk Proses Update Data

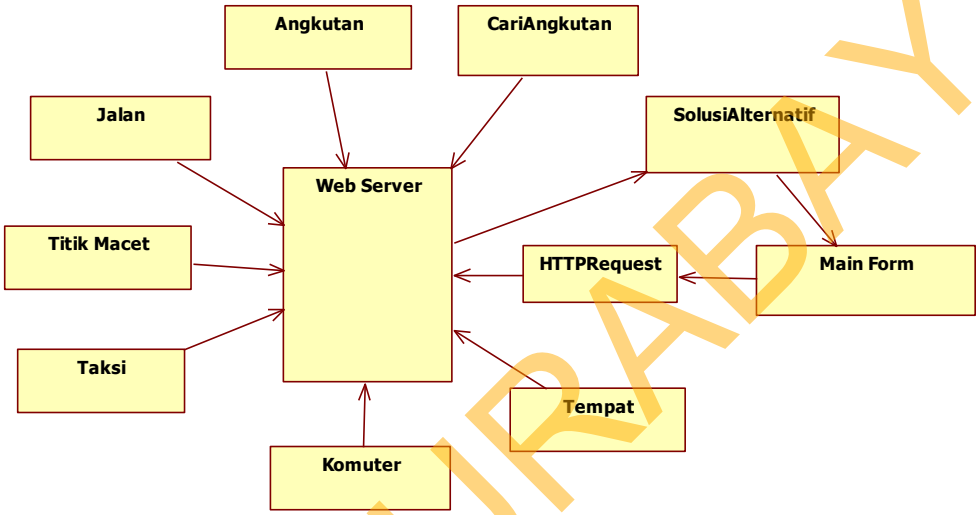
Proses dimulai ketika aplikasi menampilkan form login. Dari form login, *user* diharuskan untuk menginputkan *username* beserta *password*. Setelah *user* menekan tombol login pada form login, maka kelas login.php akan memanggil prosedur *loginValidation()* pada kelas DoLogin.php. Kelas DoLogin.php akan memvalidasi inputan *user* dan melakukan *query* pada tabel *user*. Jika data *user* tidak ditemukan, maka form login akan menampilkan pesan pemberitahuan bahwa login gagal. Tetapi jika data login ditemukan, maka form login akan menampilkan pesan pemberitahuan bahwa login berhasil dan *user* dapat mengakses form admin yang berarti memiliki hak akses penuh. Gambaran dari proses tersebut dapat dilihat pada gambar 3.15.



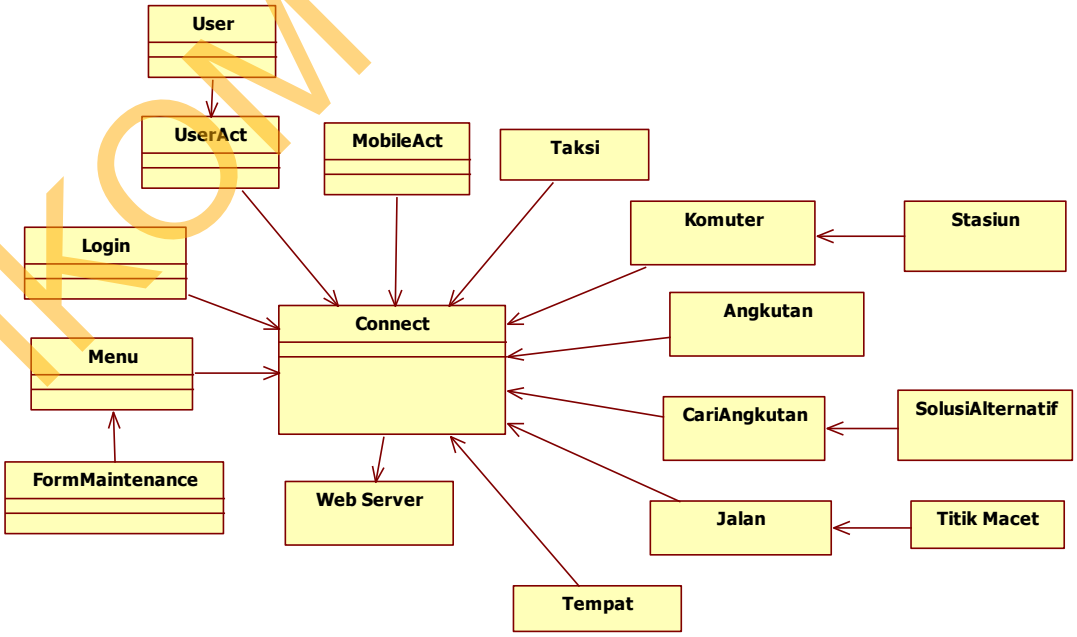
Gambar 3.15 *Sequence diagram* untuk proses "Update Data"

3.6 *Class Diagram*

Class Diagram digunakan untuk menampilkan kelas-kelas atau paket-paket didalam sistem dan relasi antar kelas tersebut (menunjukkan interaksi antar kelas di dalam aplikasi). Seperti pada gambar 3.16 dan gambar 3.17.



Gambar 3.16 *Class Diagram* Pada *Mobile Application*

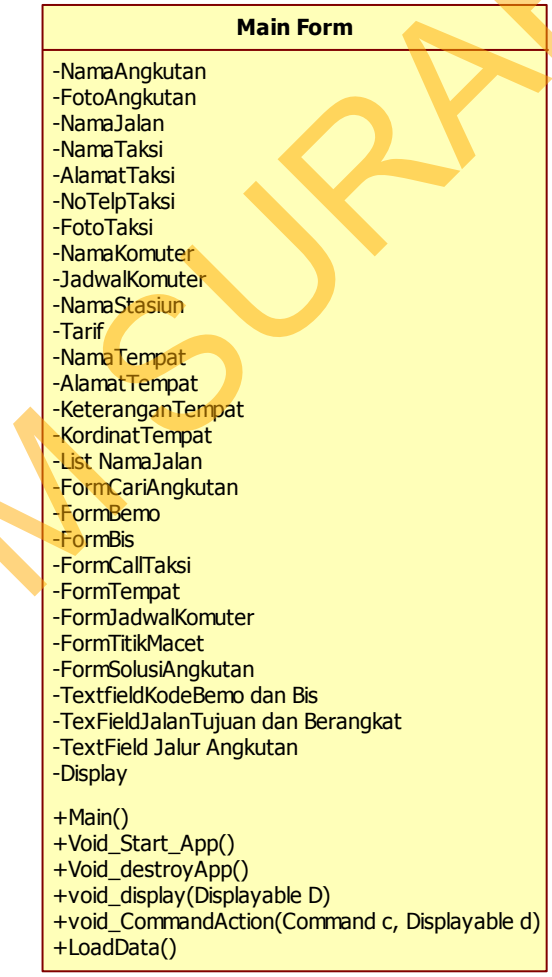


Gambar 3.17 *Class Diagram* Pada *Web Application*

3.7 *Class Diagram* Pada *Mobile Application*

A. *Class Main*

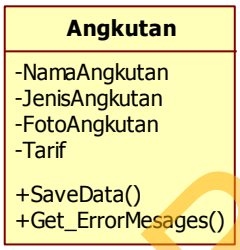
Kelas *Main* digunakan sebagai *form* utama pada *mobile application.* Kelas ini mengkoordinasikan beberapa operasi seperti inisialisasi data awal saat aplikasi dijalankan, penentuan tampilan awal aplikasi, dan lainnya.. Dengan kata lain kelas ini digunakan sebagai penghubung dengan kelas-kelas yang lain. Untuk lebih jelasnya dapat dilihat pada gambar 3.18.



Gambar 3.18 *Class* Main Pada *Mobile Application*

B. *Class* Angkutan

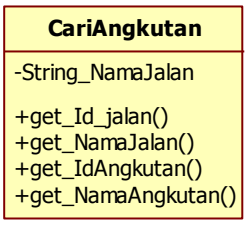
Kelas Angkutan digunakan untuk menyimpan dan menampilkan data angkutan yang melekat pada suatu menu. Di dalam kelas angkutan ini juga mempunyai atribut NamaAngkutan, JenisAngkutan, FotoAngkutan dan Tarif. *Class* ini juga mempunyai operasi saveData dan getErrorMessage. *Class diagram* Angkutan pada sistem ini dapat dilihat pada Gambar 3.19.



Gambar 3.19 *Class* Angkutan Pada *Mobile Application*

C. *Class* Cari Angkutan

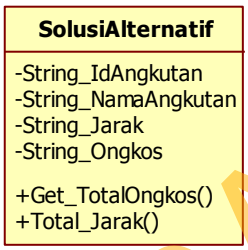
Kelas Cari Angkutan digunakan untuk menangani pencarian angkutan umum yang sesuai. Kelas ini memiliki prosedur *getNamaJalan()* untuk mengambil data jalan. Untuk lebih jelasnya dapat dilihat pada gambar 3.20.



Gambar 3.20 *Class* Cari Angkutan Pada *Mobile Application*

D. *Class* Solusi Alternatif

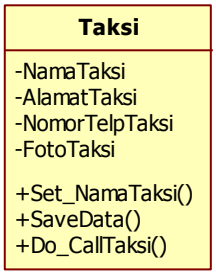
Kelas Solusi Alternatif digunakan untuk menampilkan pilihan alternatif angkutan umum. Prosedur *getTotalOngkos(int* ongkos*)* digunakan untuk menghitung total ongkos atau biaya. Untuk lebih jelasnya dapat dilihat pada gambar 3.21.



Gambar 3.21 *Class* Solusi Alternatif Pada *Mobile Application*

E. *Class* Taksi

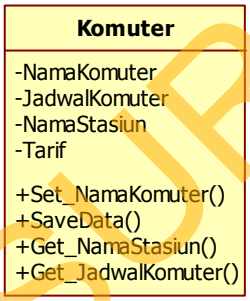
Kelas Taksi digunakan untuk menampilkan data alamat dan nomor telepon taksi. Di dalam kelas ini juga terdapat prosedur *setter* dari tiap-tiap atribut penting. Prosedur *setter* digunakan untuk mengubah nilai dari atribut yang bersesuaian. Pada *class* ini memiliki atribut NamaTaksi, AlamatTaksi, NoTelpTaksi dan FotoTaksi. Untuk lebih jelasnya dapat dilihat pada gambar 3.22.



Gambar 3.22 *Class* Taksi Pada *Mobile Application*

F. *Class* Komuter

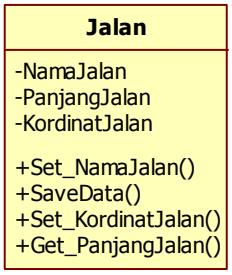
Kelas Komuter digunakan untuk menampilkan data komuter. Di dalam kelas ini juga terdapat prosedur *setter* dari tiap-tiap atribut penting. Prosedur *setter* digunakan untuk mengubah nilai dari atribut yang bersesuaian. Pada *class* ini memiliki atribut NamaKomuter, JadwalKomuter, NamaStasiun dan Tarif. Kelas ini juga memiliki prosedur *getNamaStasiun()* dan *get_JadwalKomuter* untuk mengambil data stasiun dan jadwal komuter. Untuk lebih jelasnya dapat dilihat pada gambar 3.23.



Gambar 3.23 *Class* Komuter Pada *Mobile Application*

G. *Class* Jalan

Kelas jalan digunakan untuk menampilkan data jalan. Di dalam kelas ini juga terdapat prosedur *setter* dari tiap-tiap atribut penting. Prosedur *setter* digunakan untuk mengubah nilai dari atribut yang bersesuaian. Pada *class* ini memiliki atribut NamaJalan, PanjangJalan, dan KordinatJalan. Untuk lebih jelasnya dapat dilihat pada gambar 3.24.



Gambar 3.24 *Class* Jalan Pada *Mobile Application*

H. *Class* Titik Macet

Kelas titik macet digunakan untuk menampilkan data jalan yang sering mengalami kemacetan. Di dalam kelas ini juga terdapat prosedur *setter* dari tiap-tiap atribut penting. Prosedur *getter* digunakan untuk mengambil nilai dari atribut. Pada *class* ini memiliki atribut NamaJalan, KordinatJalan, dan Keterangan. Untuk lebih jelasnya dapat dilihat pada gambar 3.25.

Gambar 3.25 *Class* Titik Macet Pada *Mobile Application*

I. *Class* Tempat

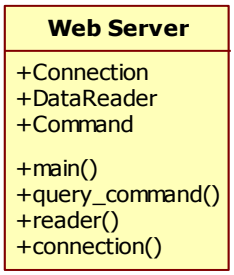
Kelas jalan digunakan untuk menampilkan data jalan. Di dalam kelas ini juga terdapat prosedur *setter* dari tiap-tiap atribut penting. Prosedur *setter*

digunakan untuk mengubah nilai dari atribut yang bersesuaian. Pada *class* ini memiliki atribut NamaJalan, PanjangJalan, dan KordinatJalan. Untuk lebih jelasnya dapat dilihat pada gambar 3.26.

Gambar 3.26 *Class* Tempat Pada *Mobile Application*

J. *Class Web Server*

Kelas *Web Server* digunakan oleh *mobile application* untuk melakukan koneksi dengan *web server*. Kelas ini mendefinisikan semua atribut dan operasi yang berguna untuk melakukan koneksi. Untuk lebih jelasnya dapat dilihat pada gambar 3.27.

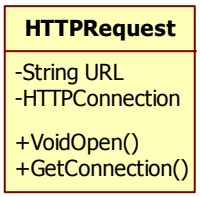


Gambar 3.27 *Class Webserver* Pada *Mobile Application*

K. *Class* HTTPRequest

Kelas HTTPRequest digunakan oleh *mobile application* untuk melakukan koneksi dengan *web server*. Kelas ini mendefinisikan semua atribut dan operasi

yang berguna untuk melakukan koneksi. Untuk lebih jelasnya dapat dilihat pada gambar 3.28.



Gambar 3.28 *Class* HTTPRequest Pada *Mobile Application*

3.8 *Component Diagram*

Component Diagram atau diagram komponen adalah diagram UML yang menampilkan komponen dalam sistem dan hubungan antara mereka. Hanya ada satu tipe relasi di dalam diagram ini yaitu relasi dependensi yang berarti suatu komponen memiliki ketergantungan dengan komponen yang lain atau satu komponen harus dikompilasi sebelum komponen lain yang bergantung padanya dikompilasi.

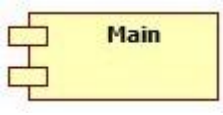
Pada Tugas Akhir ini, komponen-komponen di dalam *mobile application* dan *web application* dibagi menjadi dua komponen, yaitu komponenGUI yang menangani antar muka dengan pengguna dan komponenControl yang berisi semua kelas kontrol.

Gambar 3.29 Hubungan antar komponen

3.8.1 *Package Spesification* Pada *Mobile Application*

A. *Package Spesification* KomponenGUI

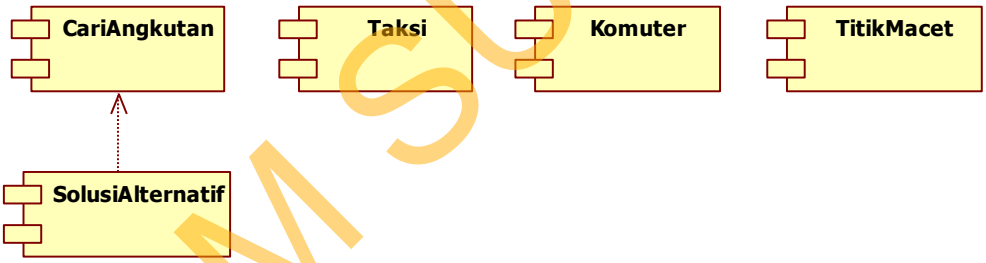
Package Spesification yang terdapat dalam KomponenGUI *mobile application* ini dapat dilihat pada gambar 3.30.



Gambar 3.30 *Package Spesification* KomponenGUI *Mobile Application*

B. *Package Spesification* KomponenKontrol

Package Spesification yang terdapat dalam KomponenKontrol *mobile application* ini dapat dilihat pada gambar 3.31.

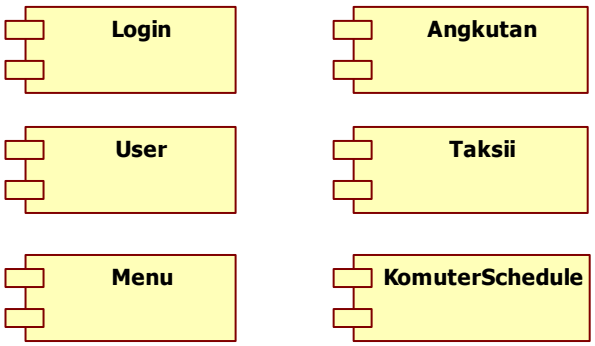


Gambar 3.31 *Package Spesification* KomponenKontrol *Mobile Application*

3.8.2 *Package Spesification* Pada *Web Application*

A. *Package Spesification* KomponenGUI

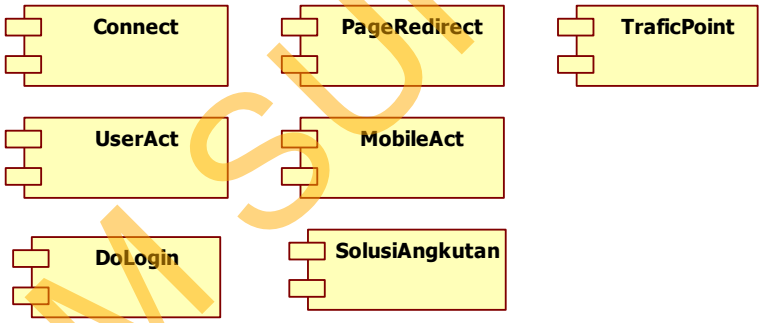
Package Spesification yang terdapat dalam KomponenGUI *web application* ini dapat dilihat pada gambar 3.32.



Gambar 3.32 *Package Spesification* KomponenGUI *Web Application*

B. Package Spesification KomponenKontrol

Package Spesification yang terdapat dalam KomponenKontrol *web application* ini dapat dilihat pada gambar 3.33.



Gambar 3.33 *Package Spesification* KomponenKontrol *Web Application*

3.9 *Deployment Diagram*

Deployment Diagram menunjukkan pandangan secara fisik dari suatu sistem dan menunjukkan bagaimana sistem diimplementasikan di perangkat nyata. *Deployment Diagram* menampilkan semua *node* dalam suatu jaringan dan hubungan di antara mereka. *Node* adalah perangkat keras yang dapat menjadi *host* dari suatu aplikasi. Deployment *Diagram* pada Tugas Akhir ini digambarkan pada gambar 3.34.

Gambar 3.34 *Deployment diagram*

3.10 Struktur Tabel

Tabel-tabel yang digunakan dalam aplikasi pemesanan makanan berbasis *mobile application* adalah sebagai berikut:

1. Nama Tabel : User

   Fungsi : Menyimpan data *users*/pengguna yang dapat mengakses *web application*.

Tabel 3.2 Struktur Tabel *User*

Nama Kolom	Tipe	Ukuran	Keterangan
ID_user	Integer	11	PK
Username	Varchar	50	
Password	Varchar	50	
Level	Varchar	25	
Blokir	Char	1	
Secret	Varchar	50	

2. Nama Tabel : Angkutan Umum

   Fungsi : Menyimpan data-data Bemo dan Bis Kota

Tabel 3.3 Struktur Tabel Bis Kota

Nama Kolom	Tipe	Ukuran	Keterangan
ID_angkutan	Varchar	5	PK
Nama_angkutan	Varchar	6	
Jenis_angkutan	Varchar	50	
Gambar	Varchar	100	

3. Nama Tabel : Taksi

   Fungsi : Menyimpan data-data taksi.

Tabel 3.4 Struktur Tabel Taksi

Nama Kolom	Tipe	Ukuran	Keterangan
ID	Varchar	5	PK
Nama	Varchar	25	
Alamat	Varchar	50	
Telephone	Varchar	-	
Gambar	Varchar	100	

4. Nama Tabel : Komuter

   Fungsi : Menyimpan data-data taksi.

Tabel 3.5 Struktur Tabel Komuter

Nama Kolom	Tipe	Ukuran	Keterangan
ID_komuter	Varchar	5	PK
Nama	Varchar	35	

5. Nama Tabel : Jalur

   Fungsi : Menyimpan data jalur yang nantinya digunakan untuk menemukan jalur atau rute angkutan.

Tabel 3.6 Struktur Tabel Jalur

Nama Kolom	Tipe	Ukuran	Keterangan
ID_jalur	Int	11	PK
ID_angkutan	Int	11	FK
ID_jalan	Int	11	FK
No_urut	Int	11	
ID_jaringan_jalan	Int	11	FK

6. Nama Tabel : Jadwal Kommuter

   Fungsi : Menyimpan data jadwal keberangkatan komuter.

Tabel 3.7 Struktur Tabel Jadwal Komuter

Nama Kolom	Tipe	Ukuran	Keterangan
ID_jadwal	Int	11	PK
Stasiun_berangkat	Int	11	
Stasiun_tujuan	Int	11	
ID_komuter	Int	11	FK
Jam_berangkat	Time		
Jam_tiba	Time		

7. Nama Tabel : Tempat

   Fungsi : Menyimpan data tempat-tempat umum.

Tabel 3.8 Struktur Tabel Tempat

Nama Kolom	Tipe	Ukuran	Keterangan
ID_Tempat	Integer	11	PK
Nama Tempat	Varchar	50	
Alamat	Varchar	100	
Keterangan	Varchar	200	
ID Kategori	Integer	11	FK
Koordinat Tempat	Varchar	30	

8. Nama Tabel : Stasiun

   Fungsi : Menyimpan data stasiun keberangkatan komuter.

Tabel 3.9 Struktur Tabel Stasiun

Nama Kolom	Tipe	Ukuran	Keterangan
ID	Integer	11	PK
Nama_Stasiun	Varchar	50	
Kota_Stasiun	Varchar	50	

9. Nama Tabel : Jalan

   Fungsi : Menyimpan data jalan yang digunakan untuk rute angkot.

Tabel 3.10 Struktur Tabel Jalan

Nama Kolom	Tipe	Ukuran	Keterangan
ID_Jalan	Integer	11	PK
Nama_Jalan	Varchar	50	
Kordinat	Varchar	35	
Panjang_Jalan	Double		
ID_Daerah	Integer		FK

10. Nama Tabel : Titik Macet

    Fungsi : Menyimpan data daerah titik rawan macet

Tabel 3.11 Struktur Tabel Titik Macet

Nama Kolom	Tipe	Ukuran	Keterangan
ID_titik_macet	Integer	11	PK
ID_jalan	Integer	50	FK
Keterangan	Text		

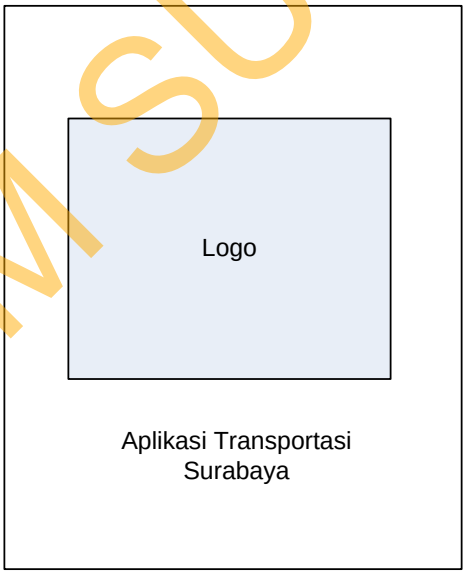
3.11 Desain *Interface* Aplikasi

Pembuatan desain *input/output* diperlukan untuk membantu pengguna berinteraksi dengan sistem. Desain *input/output* yang dibuat meliputi desain untuk *mobile application*.

3.11.1 Desain *Mobile Application*

A. Halaman Pembuka

Halaman pembuka merupakan halaman yang akan ditampilkan pertama kali ketika aplikasi dijalankan. Halaman pembuka akan menampilkan simbol Kota Surabaya. Halaman ini hanya muncul beberapa detik saja hingga kemudian akan hilang dan menampilkan halaman selanjutnya yang berisi menu-menu dari aplikasi.

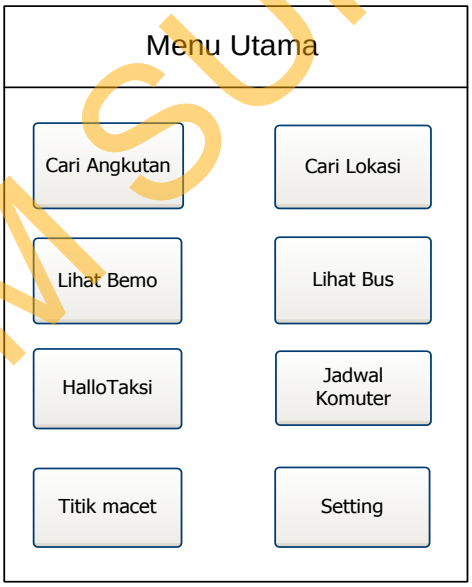


Gambar 3.35 Desain Halaman Pembuka *Mobile Application*

B. Halaman Utama

Halaman utama terdiri dari 8 pilihan, yaitu cari angkutan yang digunakan untuk melakukan pencarian angkutan kota (bemo dan bis kota) mana yang bisa

digunakan agar *user* bisa sampai pada tempat tujuanya beserta jumlah ongkos yang harus dikeluarkan. Cari lokasi yang digunakan untuk melihat dimana lokasi *user* dan tempat-tempat lainnya di Kota Surabaya dalam bentuk peta. Bis kota yang digunakan untuk melihat trayek atau rute yang dilewati bis kota. Bemo yang digunakan untuk melihat trayek atau rute yang dilewati bemo. Hallo taksi yang berisi daftar nama taksi dan nomor telepon yang bisa digunakan untuk memesan seluruh armada taksi yang beroperasi di Surabaya. Jadwal kommuter yang digunakan untuk melihat jadwal keberangkatan dan kedatangan kommuter. Menu titik kemacetan yang didalamnya berisi nama-nama tempat atau titik yang sering mengalami kemacetan. Dan yang terakhir adalah menu keluar untuk keluar dari aplikasi.

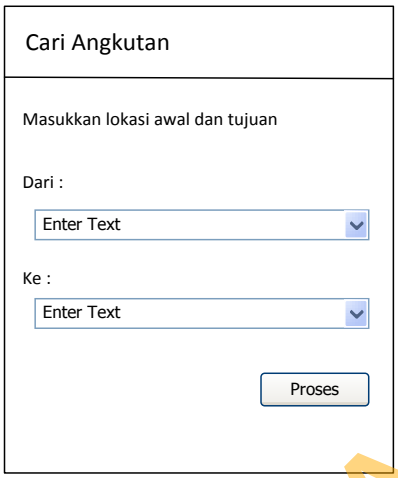


Gambar 3.36 Desain Halaman Utama *Mobile Application*

C. Halaman Cari Angkutan

Halaman cari angkutan terdapat 2 *textbox* , yang pertama adalah *textbox* "dari" yaitu untuk memasukkan lokasi *user* akan berangkat dan yang kedua

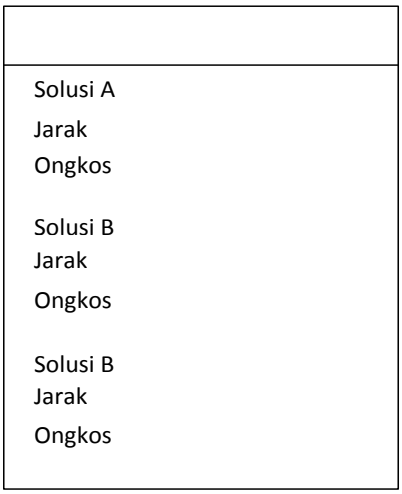
adalah *textbox* “tujuan” yaitu lokasi tujuan *user*. Dan tombol “proses” untuk melakukan pencarian setelah kedua *texbox* sudah diisi.



Gambar 3.37 Desain Halaman Cari Angkutan *(mobile)*

D. Form Solusi Detail Solusi Angkutan

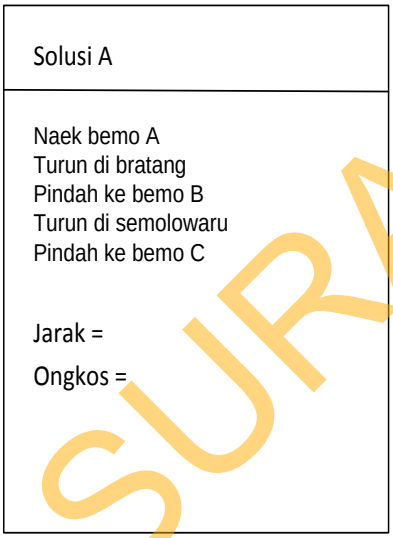
Form hasil merupakan form yang digunakan untuk menampilkan hasil dari pencarian angkutan yang sesuai dengan inputan *user*. Di dalam form solusi dapat dilihat angkutan mana saja yang bisa digunakan beserta jumlah ongkos yang diperlukan dan jarak yang ditempuh apabila memilih solusi tersebut.



Gambar 3.38 Desain Halaman Solusi Angkutan *(mobile)*

E. Form Detail Solusi Angkutan

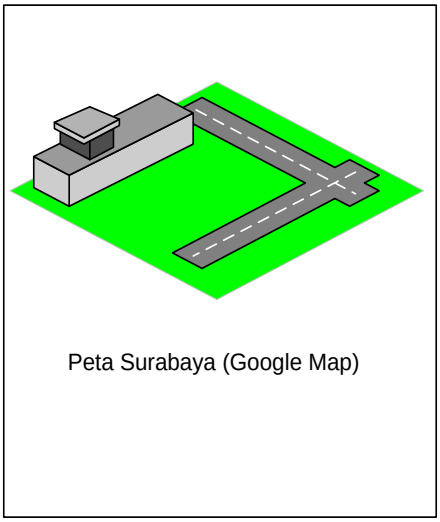
Form detail solusi merupakan form yang digunakan untuk menampilkan detail dari tiap solusi angkutan yang didapat. Form detail solusi akan tampil apabila *user* menekan salah satu solusi yang ada pada form sebelumnya. Pada form ini berupa panduan dimana *user* harus naik,turun dan berganti angkutan. Dan juga menampilkan jarak dan total ongkos yang diperlukan.



Gambar 3.39 Desain Halaman Detail Solusi Angkutan *(mobile)*

F. Form Menu Tempat

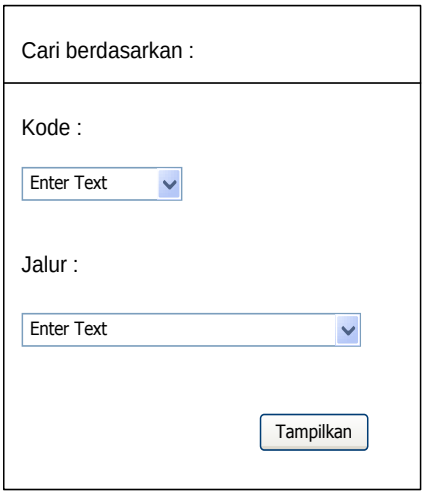
Form menu tempat menampilkan letak *user* dan tempat-tempat tertentu di Kota Surabaya dalam bentuk peta digital. Pada form cari lokasi ini *user* dapat melihat dimana lokasi dia berada dan beberapa tempat yang sudah ditandai (*mark*) pada sistem di kota Surabaya. *User* juga bisa mengetahui jarak antara dia berada dengan tempat yang sudah ditandai dalam peta tersebut dengan cara mengklik tempat tersebut. *User* juga bisa melakukan pembesaran (*zoom in*) atau pengecilan (*zoom out*) tampilan peta.



Gambar 3.40 Desain Halaman Peta *(mobile)*

G. Form Cari Bis Kota

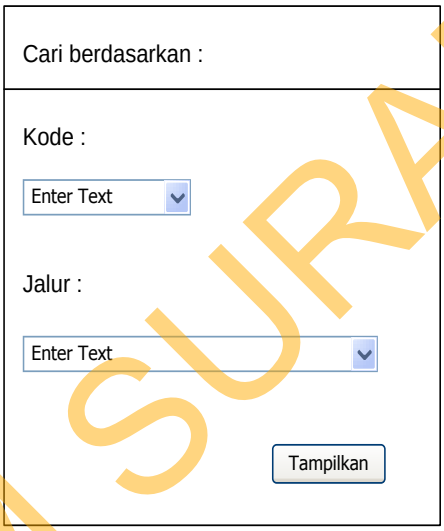
Form cari bis kota digunakan untuk mencari bis kota yang melewati rute tertentu sesuai yang diinginkan *user*. Pencarian bisa dilakukan dengan 2 cara, yaitu berdasarkan kode bis kota atau suatu lokasi tertentu yang menjadi trayek bis kota. Dan hanya bisa dipilih salah satu saja, berdasarkan kode atau rute bis kota. Setelah itu tekan tombol proses, dan akan tampil hasil pencarian berdasarkan pilihan *user* (kode atau rute).



Gambar 3.41 Desain Halaman Cari Bus *(mobile)*

H. Form Cari Bemo

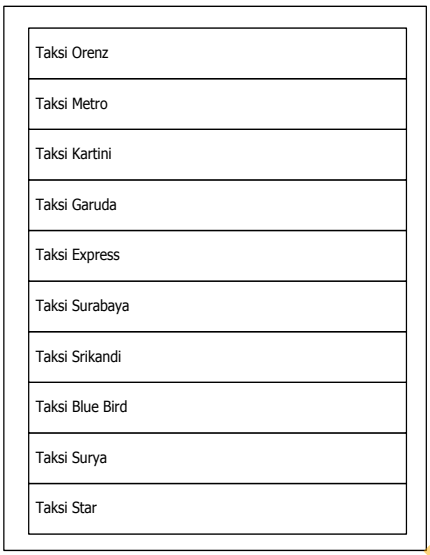
Form cari bemo digunakan untuk mencari bemo yang melewati rute tertentu sesuai yang diinginkan *user*. Pencarian bisa dilakukan dengan 2 cara, yaitu berdasarkan kode bemo atau suatu lokasi tertentu yang menjadi rute bemo. Dan hanya bisa dipilih salah satu saja, berdasarkan kode atau rute bemo. Setelah itu tekan tombol proses, dan akan tampil hasil pencarian berdasarkan pilihan *user* (kode atau rute).



Gambar 3.42 Desain Halaman Cari Bemo *(mobile)*

I. Form Hallo Taksi

Form hallo taksi berisi nama semua armada taksi yang beroperasi di Surabaya, menu ini digunakan untuk melakukan panggilan ke operator atau *call center* armada taksi tersebut. Sehingga *user* tidak perlu menyimpan nomor telephone armada taksi tersebut ke buku telepon *handphone* karena sudah tersimpan dalam aplikasi.

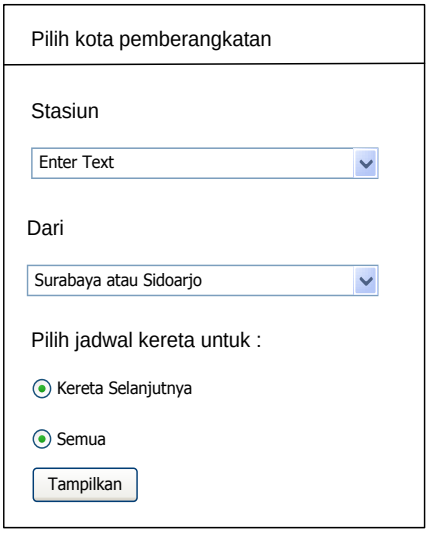


Gambar 3.43 Desain Halaman Hallo Taksi *(mobile)*

Gambar 3.44 Desain Halaman Telepon Taksi *(mobile)*

J. Form Jadwal Kommuter

Form jadwal kommuter digunakan untuk melihat jadwal keberangkatan dan kedatangan kommuter pada stasiun-stasiun tertentu. *User* dapat melihat jadwal kommuter berangkat atau datang pada stasiun yang diinginkan dengan cara memilih pada nama stasiun pada *combobox* yang ada. *User* juga bisa memfilter jadwal dalam satu hari maupun pada jam-jam tertentu.

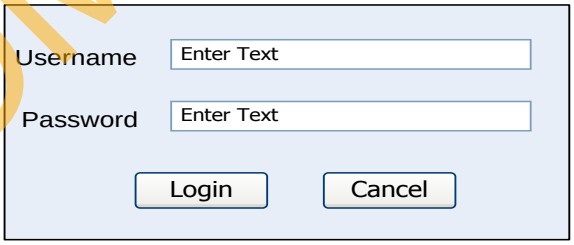


Gambar 3.45 Desain Halaman Jadwal Komuter *(mobile)*

3.11.2 Desain *Web Application*

A. Form Login

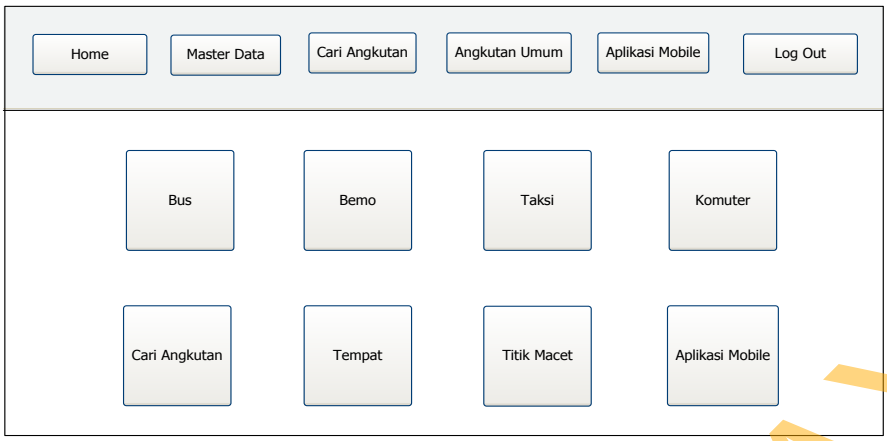
Form login merupakan halaman utama yang akan ditampilkan ketika *users*/pengguna membuka halaman *web transportasi Surabaya*. Form login digunakan untuk memverifikasi pengguna yang melakukan login, karena hanya admin yang memiliki akses penuh.



Gambar 3.46 Desain Halaman Form Login (*web*)

B. Halaman Utama

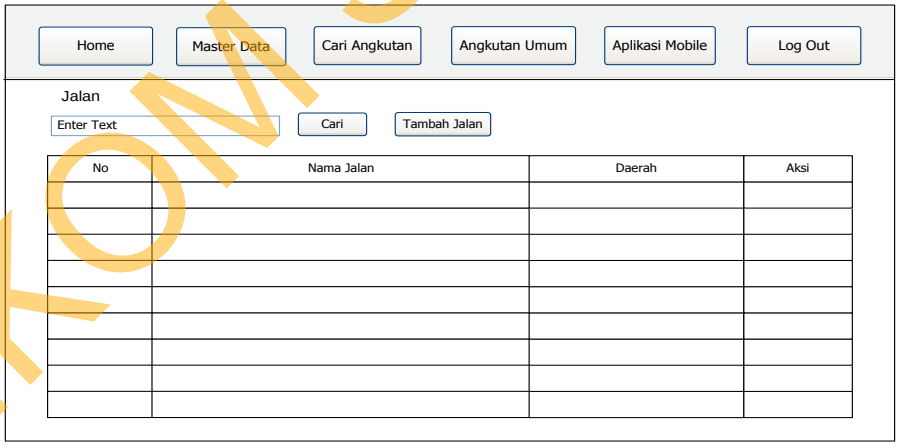
Pada halaman utama terdapat waktu sistem yang ditampilkan pada bagian kiri atas dari halaman utama dan pilihan menu yang berupa ikon.



Gambar 3.47 Desain Halaman Utama (*web*)

C. Form Jalan

Form jalan digunakan untuk memelihara master jalan, seperti menyimpan, mengubah dan menghapus data jalan yang merupakan jalur yang dilewati angkutan umum. Data yang ditambahkan antara lain, nama jalan, koordinat jalan, panjang jalan dan daerah. Pada form ini terdapat tombol tambah jalan baru, tombol ubah dan tombol hapus.

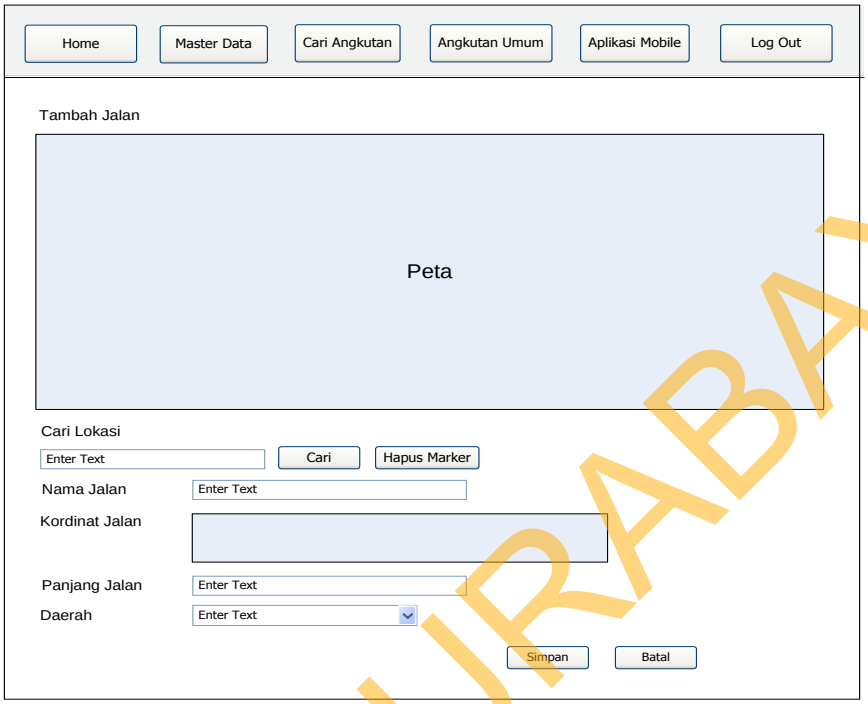


Gambar 3.48 Desain Halaman Form Jalan (*web*)

D. Form Tambah Jalan

Form tambah jalan digunakan untuk menambah data jalan baru. Data yang ditambahkan antara lain nama jalan, koordinat jalan, panjang jalan, dan daerah.

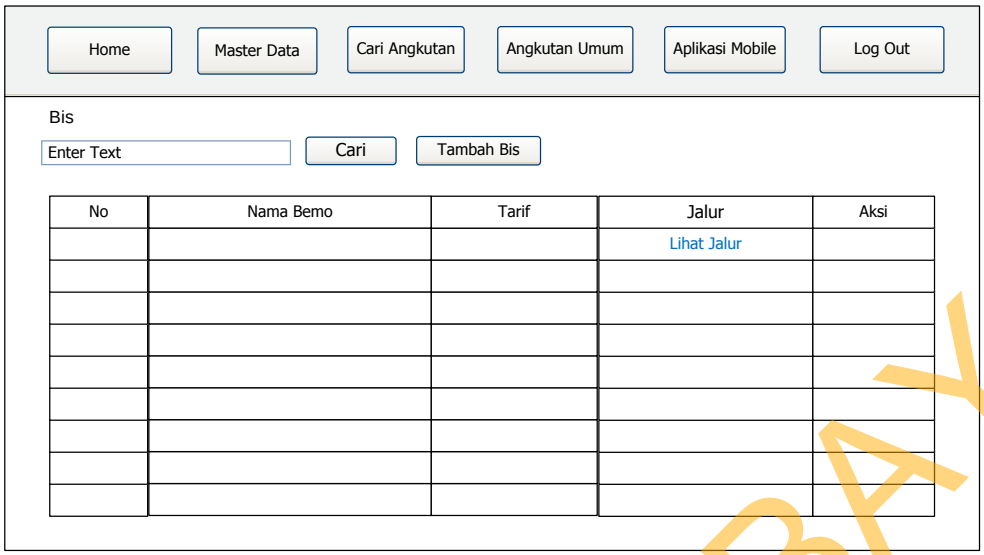
Koordinat jalan didapat secara otomatis dari peta yang tersedia pada halaman web.



Gambar 3.49 Desain Halaman Form Tambah Jalan (*web*)

E. Form Bis

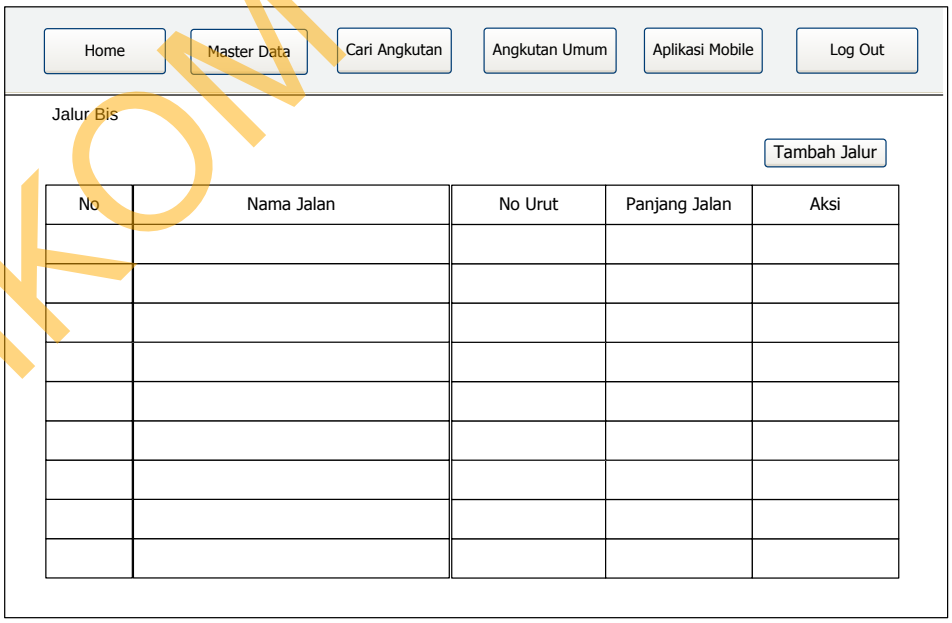
Form bis digunakan untuk memelihara master bis, seperti menyimpan, mengubah dan menghapus data bis. Data yang ditambahkan antara lain, nama bis dan jalur yang merupakan rute bis. Pada form ini terdapat tombol tambah bus baru, tombol ubah dan tombol hapus.



Gambar 3.50 Desain Form Bis (*web*)

F. Form Lihat Jalur Bis

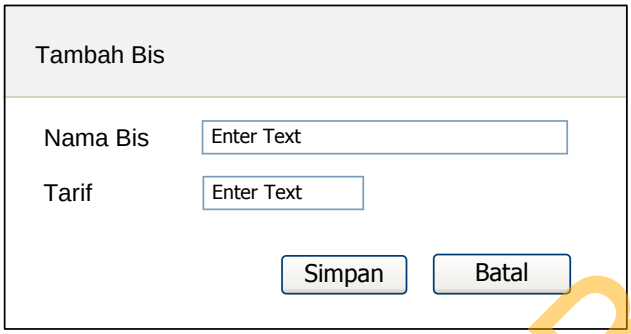
Form lihat jalur bis digunakan untuk melihat rute bis, dan bisa juga untuk menambah, menghapus dan menambah rute baru. Pada form ini terdapat tombol tambah rute baru dan hapus rute.



Gambar 3.51 Desain Form Lihat Jalur Bis (*web*)

G. Form Tambah Bis

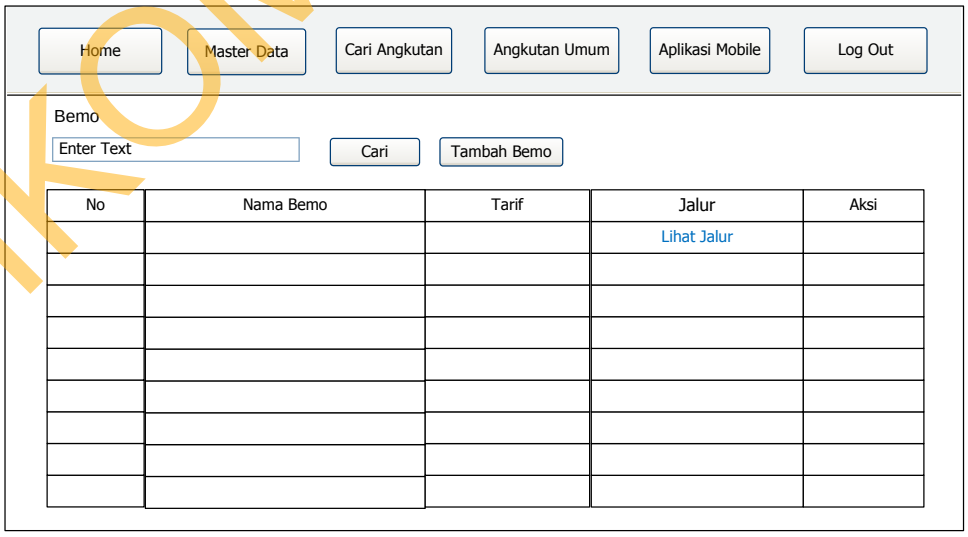
Form tambah bis digunakan untuk menambah data bis baru. Data yang ditambahkan antara lain nama bis dan tarif bis.



Gambar 3.52 Desain Form Tambah Bis (*web*)

H. Form Bemo

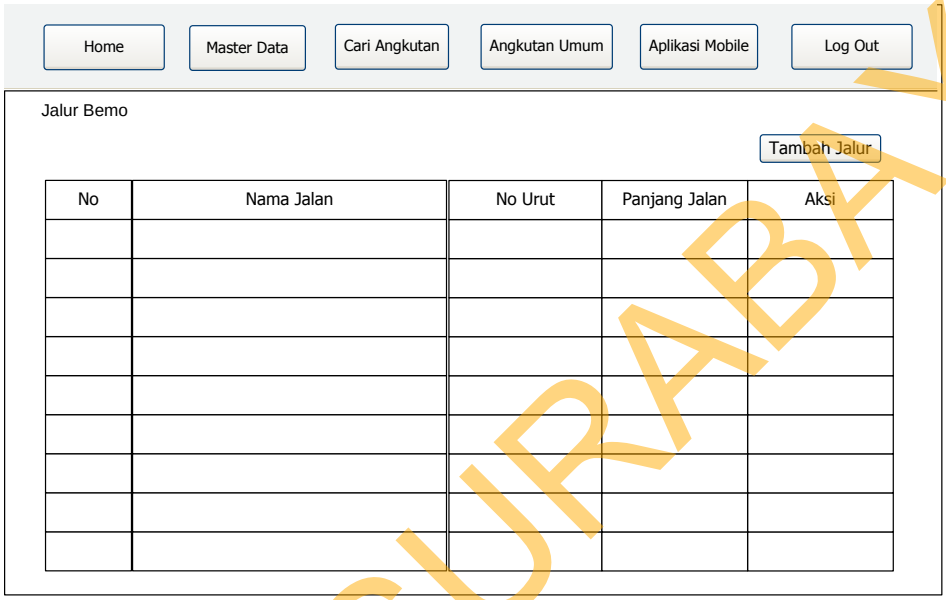
Form bemo digunakan untuk memelihara master bemo, seperti menyimpan, mengubah dan menghapus data bemo. Data yang ditambahkan antara lain, nama bemo dan jalur yang merupakan rute bemo. Pada form ini terdapat tombol tambah bemo baru, tombol ubah dan tombol hapus.



Gambar 3.53 Desain Form Bemo (*web*)

I. Form Lihat Jalur Bemo

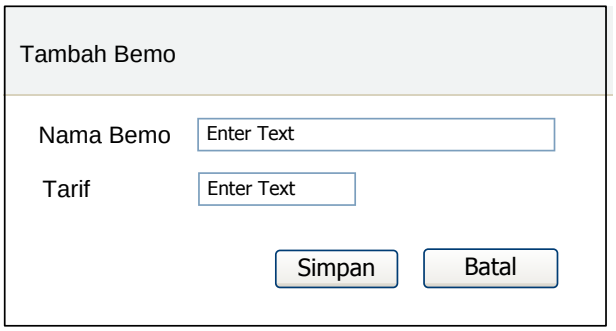
Form lihat jalur bemo digunakan untuk melihat rute bemo, dan bisa juga untuk menambah, menghapus dan menambah rute baru. Pada form ini terdapat tombol tambah rute baru dan hapus rute.



Gambar 3.54 Desain Form Lihat Jalur Bemo (*web*)

J. Form Tambah Bemo

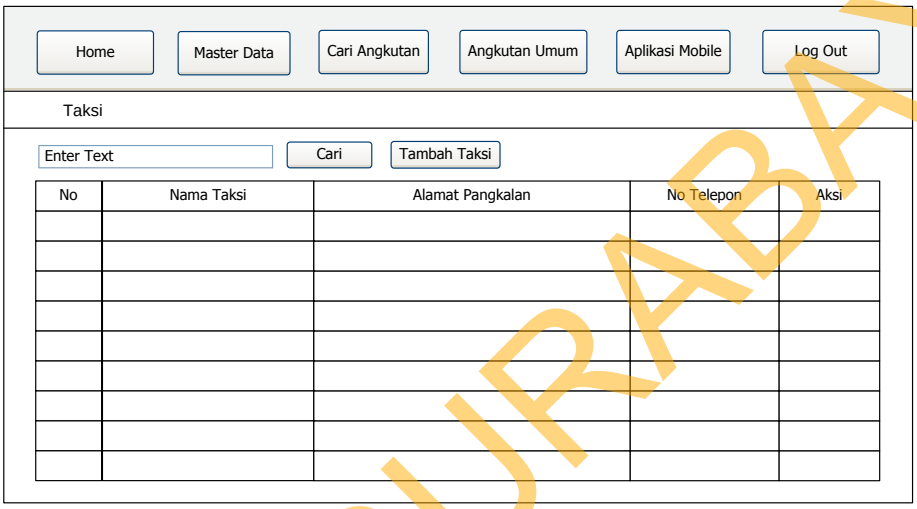
Form tambah bemo digunakan untuk menambah data bemo baru. Data yang ditambahkan antara lain nama bemo dan tarif bemo.



Gambar 3.55 Desain Form Bemo (*web*)

K. Form Taksi

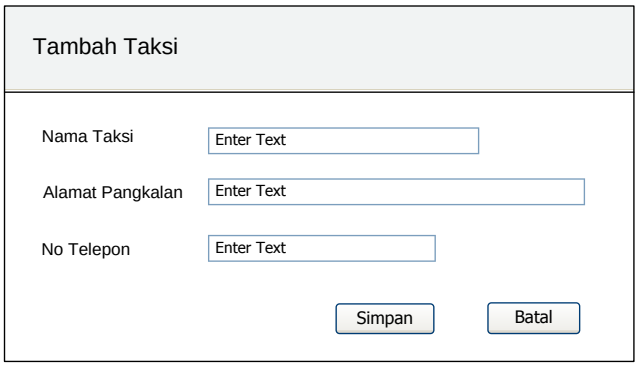
Form taksi digunakan untuk memelihara master taksi, seperti menyimpan, mengubah dan menghapus data taksi. Data yang ditambahkan antara lain, nama taksi, alamat pangkalan dan nomor telepon taksi. Pada form ini terdapat tombol tambah taksi baru, tombol ubah dan tombol hapus.



Gambar 3.56 Desain Form Taksi (*web*)

L. Form Tambah Taksi

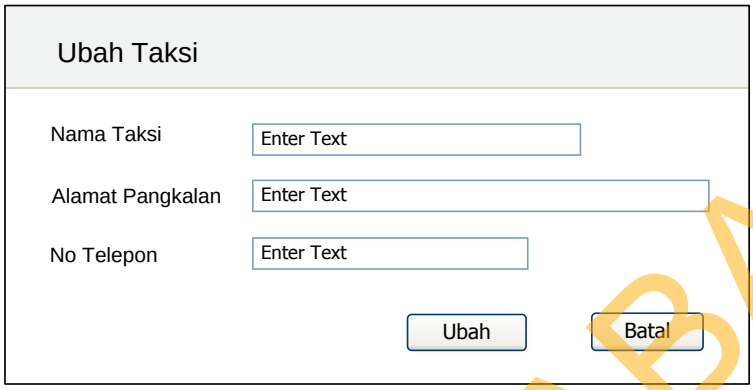
Form tambah taksi digunakan untuk menambah data taksi baru. Data yang ditambahkan antara lain nama taksi, alamat pangkalan dan no telepon *call center* armada taksi.



Gambar 3.57 Desain Form Tambah Taksi (*web*)

M. Form Edit Taksi

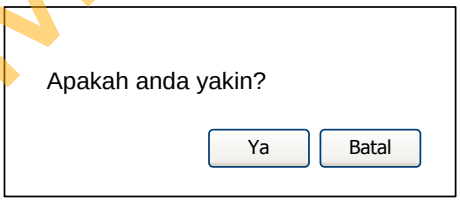
Form edit taksi digunakan untuk mengubah data taksi yang ada. Data yang dapat diubah adalah nama, alamat pangkalan dan nomor telepon taksi.



Gambar 3.58 Desain Form Edit Taksi (*web*)

F. Form Hapus Taksi

Form hapus taksi digunakan untuk menghapus data taksi. Setelah data terhapus, maka web akan me – *refresh* otomatis dan data yang telah terhapus tidak akan tampil lagi pada form taksi.

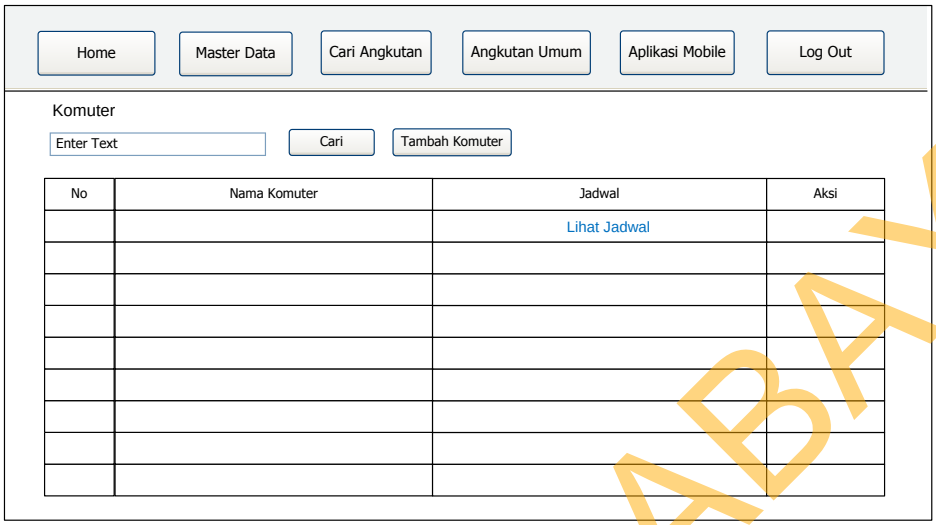


Gambar 3.59 Desain Form Hapus Taksi (*web*)

N. Form Kommuter

Form kommuter digunakan untuk memelihara master kommuter, seperti menyimpan, mengubah dan menghapus data kommuter. Data yang ditambahkan antara lain, nama kommuter dan jadwal keberangkatan kommuter. Pada form ini

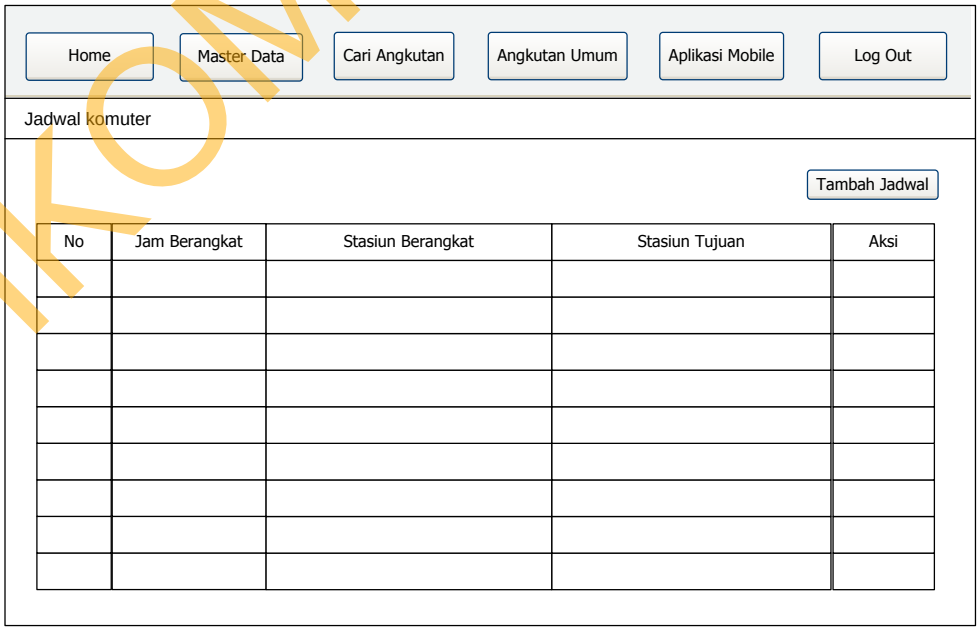
terdapat tombol tambah kommuter baru, tombol ubah, tombol hapus dan untuk melihat jadwal keberangkatan.



Gambar 3.60 Desain Form Komuter (*web*)

O. Form Jadwal Komuter

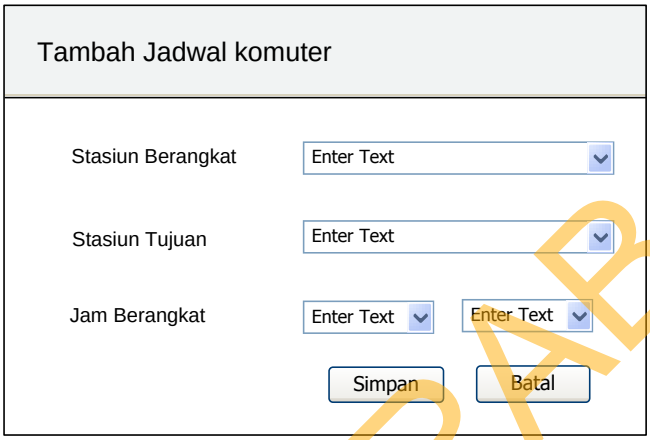
Form lihat jadwal komuter digunakan untuk melihat jadwal keberangkatan komuter dari satu stasiun ke stasiun lain. Pada form ini juga terdapat tombol tambah, mengubah dan menghapus jadwal.



Gambar 3.61 Desain Form Jadwal Komuter (*web*)

P. Form Tambah Jadwal Komuter

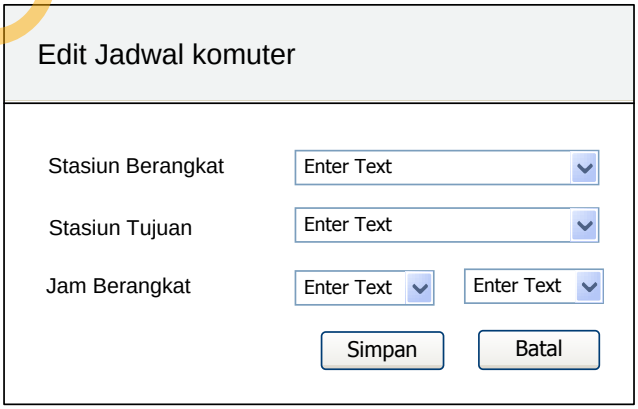
Form tambah jadwal komuter digunakan untuk menambah data jadwal baru. Data yang ditambahkan antara lain stasiun berangkat, stasiun tujuan dan jam keberangkatan komuter.



Gambar 3.62 Desain Form Tambah Jadwal Komuter (*web*)

Q. Form Edit Jadwal Komuter

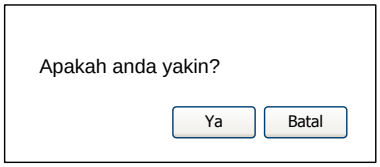
Form edit jadwal komuter digunakan untuk mengubah jadwal komuter yang ada. Data yang dapat diubah adalah stasiun berangkat, stasiun tujuan dan jam berangkat.



Gambar 3.63 Desain Form Edit Jadwal Komuter (*web*)

R. Form Hapus Komuter

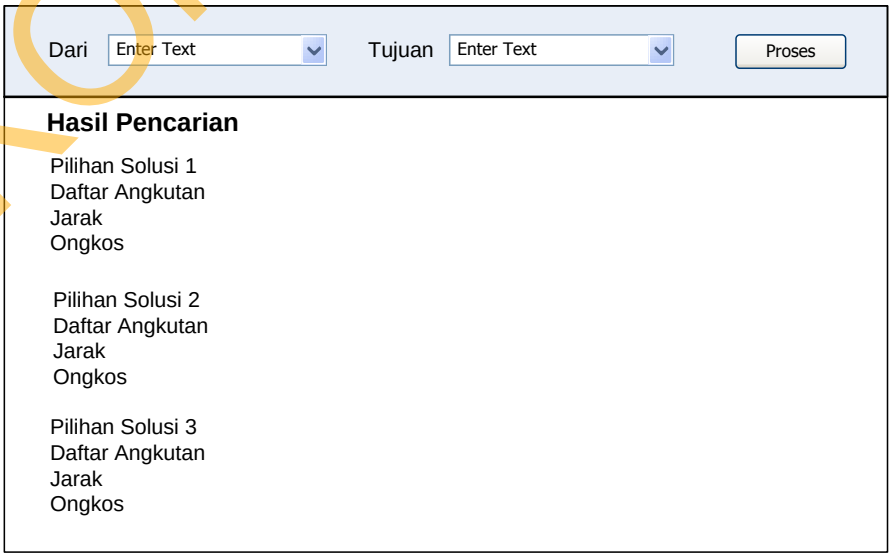
Form hapus komuter digunakan untuk menghapus data komuter. Setelah data terhapus, maka web akan me – *refresh* otomatis dan data yang telah terhapus tidak akan tampil lagi pada form komuter.



Gambar 3.64 Desain Form Hapus Komuter (*web*)

S. Form Cari Angkutan

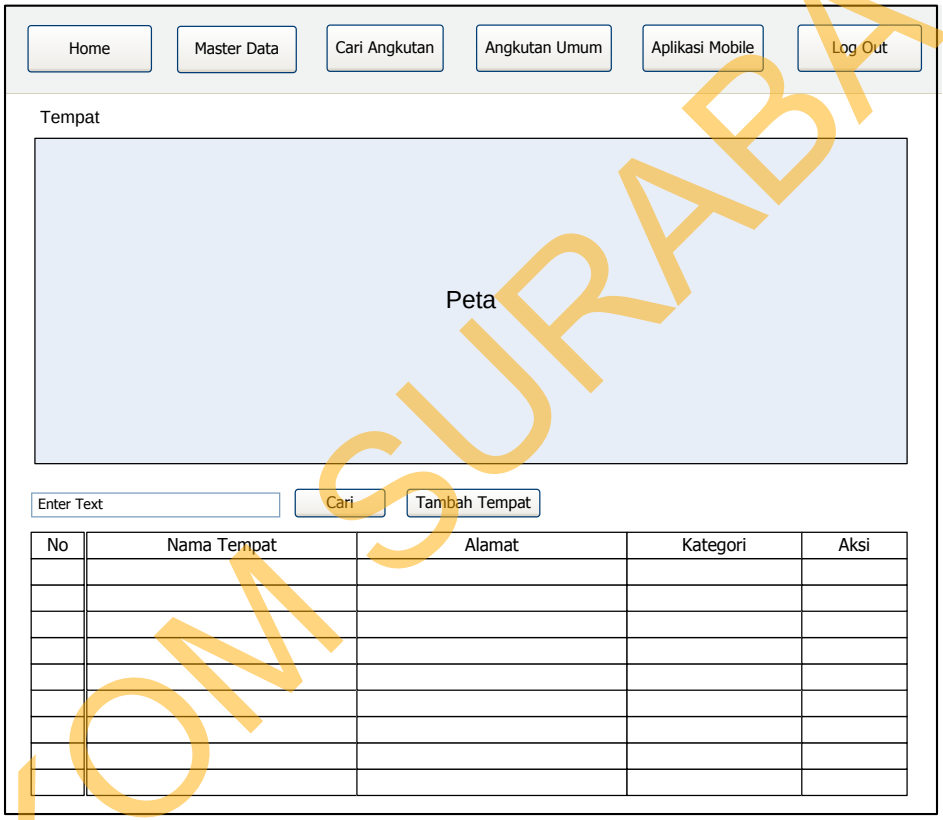
Form cari angkutan terdapat 2 *combobox* , yang pertama adalah *combobox* "dari" yaitu untuk memasukkan lokasi *user* akan berangkat dan yang kedua adalah *combobox* "tujuan" yaitu lokasi tujuan *user*. Dan tombol "proses" untuk melakukan pencarian setelah kedua lokasi sudah diisi. Setelah melakukan proses, hasil pencarian angkutan akan ditampilkan pada bagian bawah. Dan hasil pencarian tersebut terdiri dari beberapa pilihan solusi.



Gambar 3.65 Desain Form Cari Angkutan (*web*)

T. Form Tempat

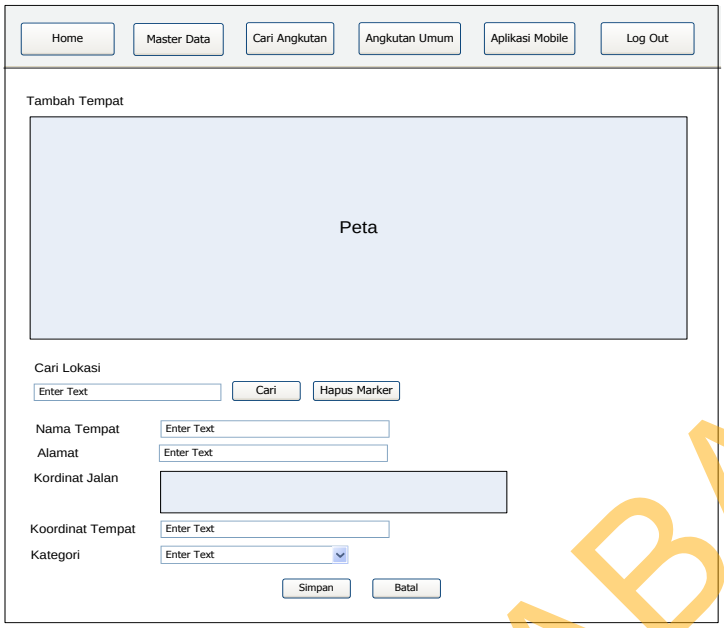
Form tempat digunakan untuk memelihara master tempat, seperti menyimpan, mengubah dan menghapus data tempat. Data yang ditambahkan antara lain, nama tempat, alamat, keterangan, kordinat tempat dan kategori tempat. Pada form ini terdapat tombol tambah tempat baru, tombol ubah dan tombol hapus.



Gambar 3.66 Desain Form Tempat (*web*)

U. Form Tambah Tempat

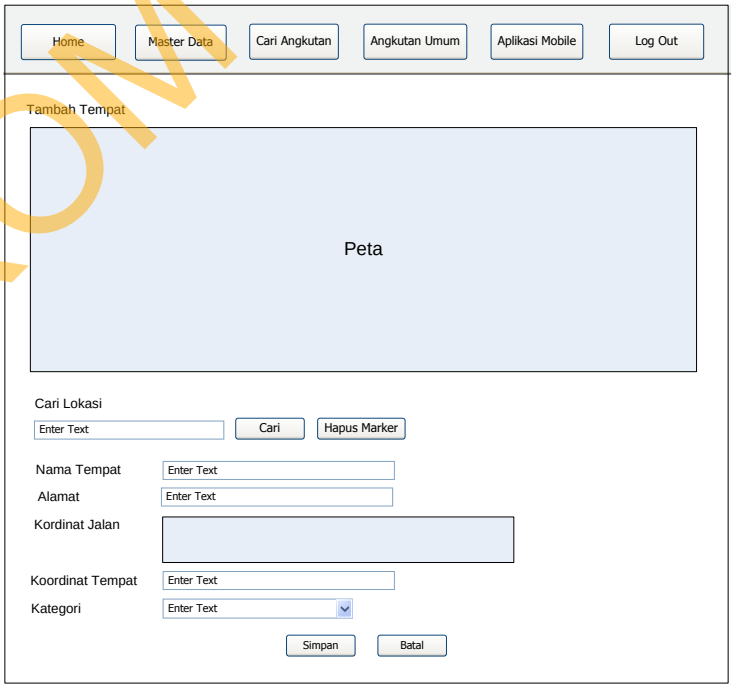
Form tambah tempat digunakan untuk menambah data tempat baru. Data yang ditambahkan antara lain id tempat, nama tempat, alamat, keterangan, koordinat dan kategori tempat. Koordinat tempat didapat secara otomatis dari peta yang tersedia pada halaman web.



Gambar 3.67 Desain Form Tambah Tempat (*web*)

V. Form Edit Tempat

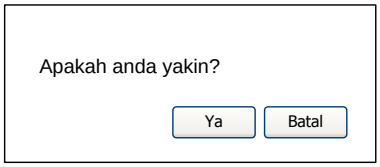
Form edit tempat digunakan untuk mengubah data tempat yang ada. Data yang dapat diubah adalah nama tempat, alamat, keterangan, koordinat dan kategori tempat.



Gambar 3.68 Desain Form Edit Tempat (*web*)

W. Form Hapus Tempat

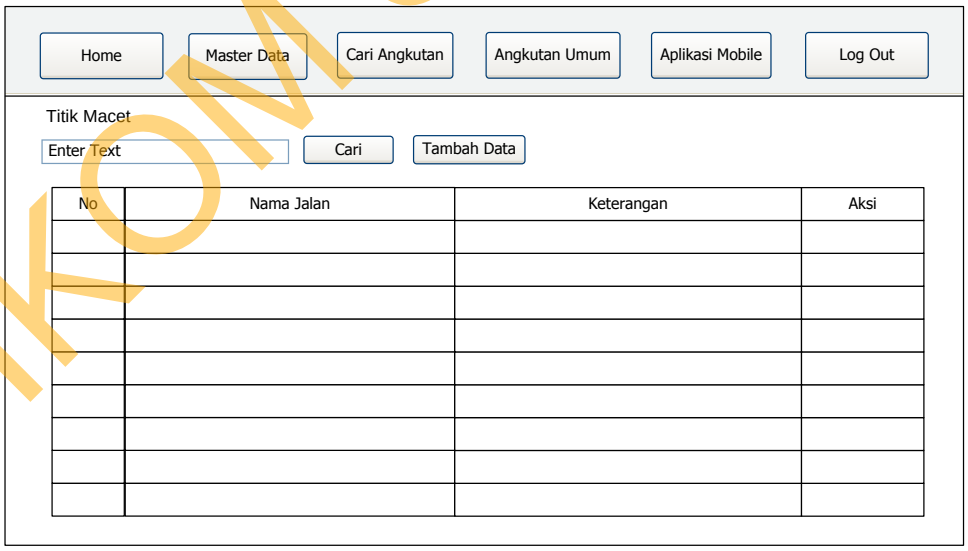
Form hapus tempat digunakan untuk menghapus data tempat. Setelah data terhapus, maka web akan me – *refresh* otomatis dan data yang telah terhapus tidak akan tampil lagi pada form tempat.



Gambar 3.69 Desain Form Hapus Tempat (*web*)

X. Form Titik Macet

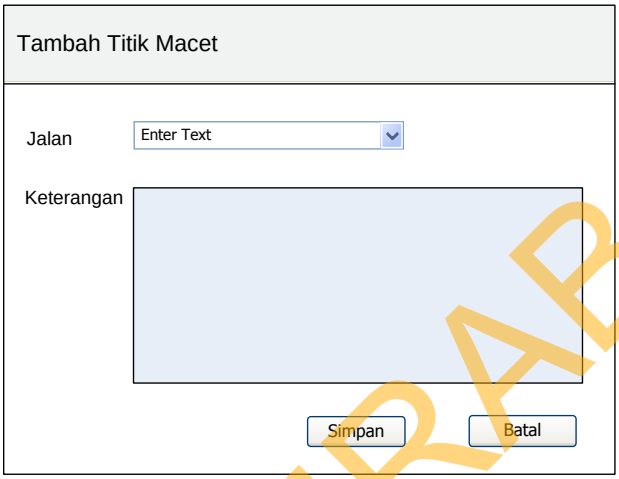
Form titik macet digunakan untuk memelihara master titik macet, seperti menyimpan, mengubah dan menghapus data titik macet. Data yang ditambahkan antara lain, nama tempat dan keterangan. Pada form ini terdapat tombol tambah tempat baru, tombol ubah dan tombol hapus.



Gambar 3.70 Desain Form Titik Macet (*web*)

Y. Form Tambah Titik Macet

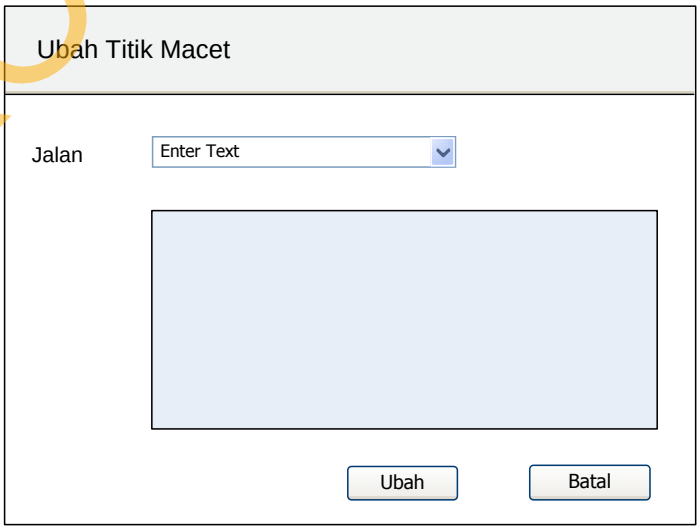
Form tambah titik macet digunakan untuk menambah data titik kemacetan baru. Data yang ditambahkan adalah jalan yang sering mengalami kemacetan dan juga keterangannya.



Gambar 3.71 Desain Form Tambah Titik Macet (*web*)

Z. Form Ubah Titik Macet

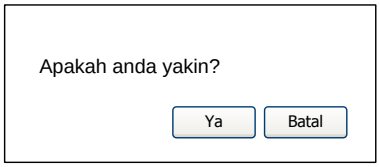
Form edit titik macet digunakan untuk mengubah data titik macet yang ada. Data yang dapat diubah adalah nama jalan dan keterangan.



Gambar 3.72 Desain Form Ubah Titik Macet (*web*)

AA. Form Hapus Titik Macet

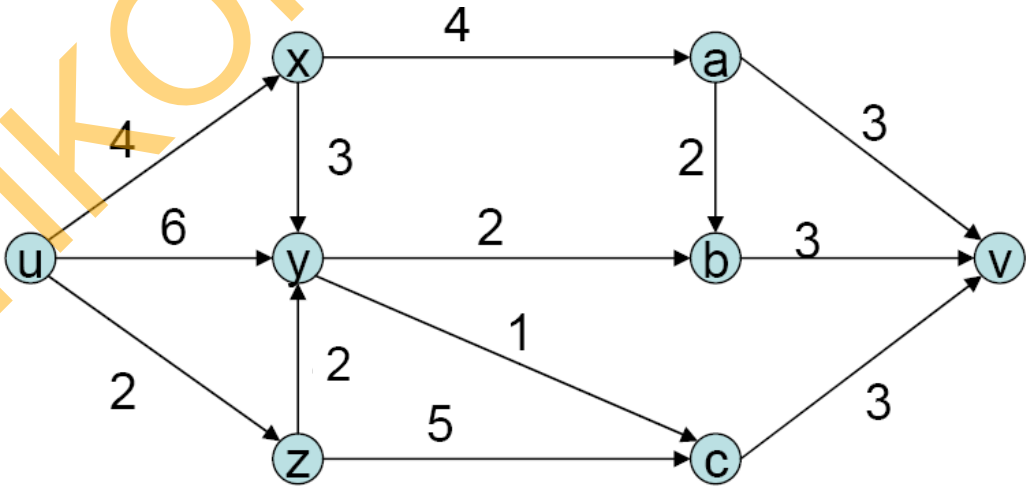
Form hapus titik macet digunakan untuk menghapus data titik macet. Setelah data terhapus, maka web akan me – *refresh* otomatis dan data yang telah terhapus tidak akan tampil lagi pada form titik macet.



Gambar 3.73 Desain Form Hapus Titik Macet (*web*)

3.12. Rancangan Perhitungan Jalur Terpendek

Dalam pencarian jalur terpendek pada aplikasi ini diperlukan suatu perhitungan dalam menentukan rute yang harus dilewati yang merupakan jalur terpendek dari suatu jaringan (graf) jalan yang sudah terbentuk. Penyelesaian masalah pencarian *Spanning Tree* pada aplikasi ini menggunakan perhitungan algoritma jalur terpendek. Proses perhitungan jalur terpendek menggunakan *Spanning Tree* adalah sebagai berikut :



Gambar 3.74 Jaringan Jalan (Graf) *Spanning Tree*

Gambar 3.74 di atas merupakan jaringan jalan *Spanning Tree* yang terdiri dari beberapa node. Dan untuk menentukan jalur terpendek yang harus dilewati dari node U ke V harus dilakukan suatu perhitungan. Dan berikut adalah langkah-langkah perhitungan jalur terpendek.

1. Buat tabel jarak

Tabel 3.12 Perhitungan Jalur Terpendek

u	x	y	z	a	b	c	v
ux = 4	xy = 3	yb = 2	zy = 2	ab = 2	bv = 3	cv = 3	
uy = 6	xa = 4	yc = 1	zc = 5	av = 3			
uz = 2							

2. Mulai dari kolom U, beri nilai 0. Pada kolom ini pilih jarak atau busur dengan nilai terkecil, yaitu uz=2. Lingkari uz. Semua busur yang berakhir di z dihapus (pada kasus ini tidak ada). Beri nilai 2 untuk kolom z.

Tabel 3.13. Perhitungan Jalur Terpendek (part.2)

u (0)	x	y	z (2)	a	b	c	v
ux = 4	xy = 3	yb = 2	zy = 2	ab = 2	bv = 3	cv = 3	
uy = 6	xa = 4	yc = 1	zc = 5	av = 3			
uz = 2							

3. Dari kolom yang sudah diberi nilai, cari busur lain yang belum dilingkari nilainya paling kecil jika dijumlahkan dengan nilai kolom. Dalam hal ini ada 2 pilihan yaitu ux=4 dan zy=2 (lingkari keduanya). Beri nilai kolom x= 0+4=4 dan y= 2+2=4. Hapus semua busur yang menuju x dan y.

Tabel 3.14. Perhitungan Jalur Terpendek (part.3)

u (0)	x (4)	y (4)	z (2)	a	b	c	v
ux = 4	~~xy = 3~~	yb = 2	zy = 2	ab = 2	bv = 3	cv = 3	
~~uy = 6~~	xa = 4	yc = 1	zc = 5	av = 3			
uz = 2							

4. Ulangi langkah ke 3 sampai semua kolom memiliki nilai. Berikut adalah hasil akhir tabel setelah mengulangi langkah ke 3 beberapa kali.

Tabel 3.15. Perhitungan Jalur Terpendek (part.4)

u (0)	x (4)	y (4)	z (2)	a (8)	b (6)	c(5)	v(8)
ux = 4	~~xy = 3~~	yb = 2	zy = 2	~~ab = 2~~	~~bv = 3~~	cv = 3	
~~uy = 6~~	xa = 4	yc = 1	~~zc = 5~~	~~av = 3~~			
uz = 2							

Berdasarkan hasil perhitungan dari langkah-langkah diatas, lakukan penelusuran terbalik mulai dari simpul akhir (v), sehingga diperoleh :

$$v \leftarrow c \leftarrow y \leftarrow z \leftarrow u$$

Hasil diatas adalah rute terpendek dari u ke v dengan nilai = 8.