BAB III
ANALISIS DAN PERANCANGAN

3.1 Analisis Sistem

Analisis sistem yang sedang berjalan merupakan penguraian dari suatu informasi yang utuh ke dalam bagian-bagian komponennya yang dimaksudkan untuk mengidentifikasi dan mengevaluasi permasalahan-permasalahan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikannya.

Untuk memahami alur dari informasi dalam sistem, diperlukan pendokumentasian dalam merancang suatu aplikasi system pendukung keputusan rute optimal dengan metode algoritma genetik sehingga akan mempermudah tahap pengembangan sistem.

3.1.1 Analisis Masalah

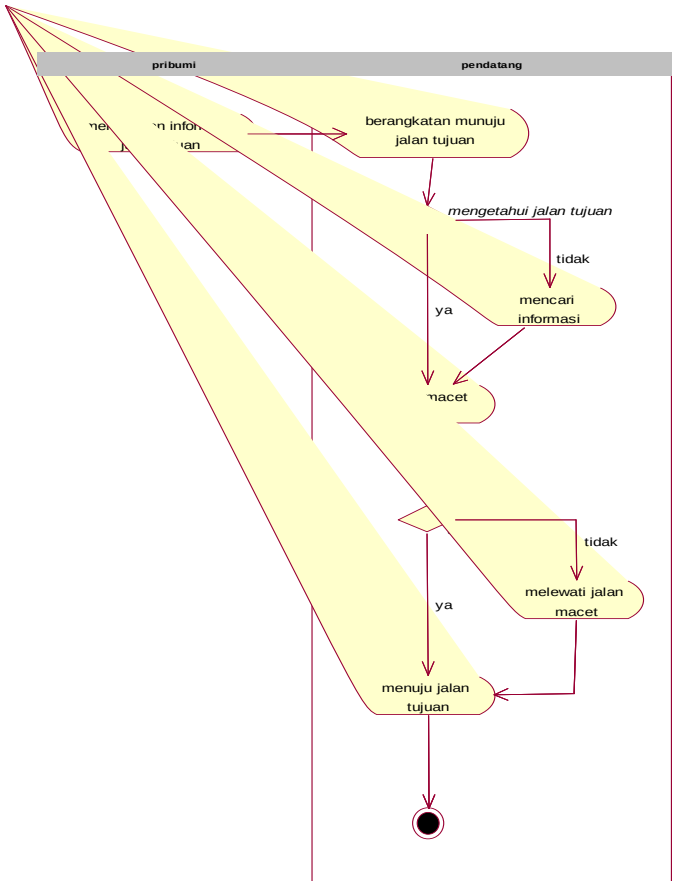
Berdasarkan hasil pengamatan dilapangan, dan wawancara di Dinas Perhubungan, jalan-jalan dikota besar sangat menbingungkan dan Bandung adalah salah satu kota besar di INDONESIA, walau pemetaan kota Bandung yang cukup lumayan baik, tetapi pendatang masih dihadapkan kebingungan dengan jalan-jalan kota Bandung, dengan itu pendatang menjadi kesulitan untuk mencari jalan yang dituju, dan pendatangpun kebingungan untuk mencari rute menuju jalan tujuan

3.1.2 Analisis Fungsional

Sebelum membangun sebuah program aplikasi, tahap pertama yang harus dilakukan mempelajari dan menganalisa sistem yang sedang berjalan. Analisis pemilihan rute menuju jalan tujuan saat ini akan digambarkan dalam *activity diagram*. Pemahaman terhadap sistem dilakukan dengan mempelajari bagaimana sistem tersebut berjalan.

3.1.2.1 Penentuan Rute Menuju Jalan Tujuan

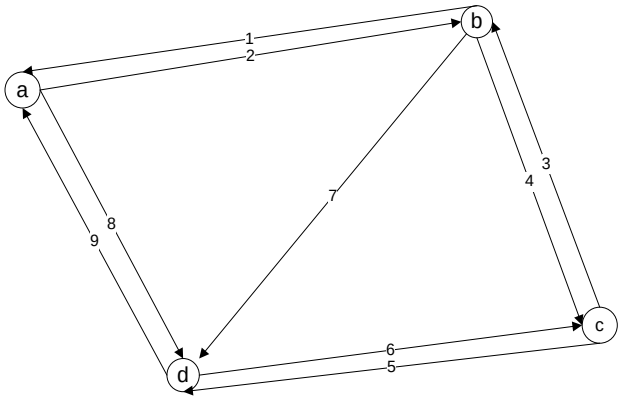
Pribumi memberikan informasi kepada pendatang munuju jalan tujuan, pendatang menuju lokasi tujuan bila terdapat jalan menuju macet pendatang melakukan jalan anternatif atau pendatang melewati jalan yang macet



Gambar 3.1 Diagram Aktifitas Penentuan Rute Tujuan

3.1.2.2 Penyelesaian Masalah Dengan Menggunakan Metode Algoritma Genetik

Berikut adalah contoh kasus pencarian rute optimal yang akan dipecahkan dengan menggunakan algoritma genetika. Dalam kasus ini, akan dicari rute yang harus ditempuh dari jalan 1 ke jalan 4.



Gambar 3.2 Gambar Contoh Kasus Graf

Dalam kasus ini, akan dicari rute yang harus ditempuh dari jalan 1 ke jalan 4 untuk menemukan rute yang tercepat dari 5 jalan, maka pada pencarian rute di asumsikan setiap jalur jalan hanya dapat dilalui sebanyak 1 kali dan jalur jalan yang digunakan dapat berupa jalan 1 arah atau jalan 2 arah.dan dimisalkan data jalannya seperti pada tabel 3.1

Tabel 3.1 Tabel Contoh Data Jalan

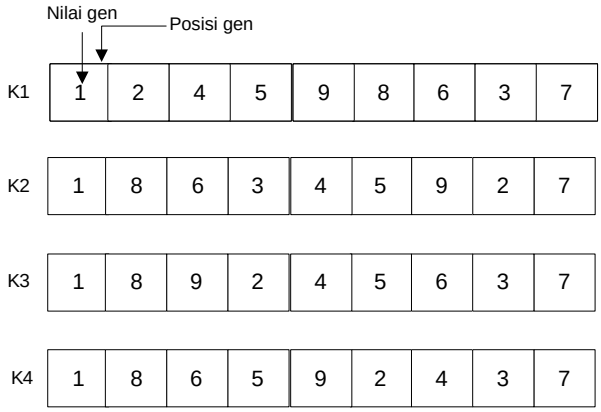
id_jalan	nama_jalan	Dari	Ke	Jarak (m)	Kecepatan (km/jam)
1	Jln.Ahmad Yani	b	a	1050	12,32
2	Jln.Ahmad Yani	a	b	1050	12,32
3	Jln.ABC	c	b	420	9,32
4	Jln.ABC	b	c	420	9,32
5	Jln.Aceh	c	d	150	16,21
6	Jln.Aceh	d	c	150	16,21
7	Jln.Asia Afrika	d	b	400	10,34
8	Jln.Banda	a	d	300	14,34
9	Jln.Banda	d	a	300	14,34

Penyelesaian dengan algoritma genetika:

1. Inisialisasi Populasi

   Pada saat inisialisasi populasi, algoritma genetika membangkitkan secara acak sejumlah individu sebagai suatu populasi.

   Misalkan jumlah kromosom dalam setiap populasi (ukuran populasi) adalah 4 dan pembangkitan populasi awal menghasilkan 4 kromosom. Pada kasus ini representasi kromosom yang dipakai adalah representasi permutasi dimana nilai gen menunjukan id jalan dan posisi gen menunjukan urutan kunjungan.



Gambar 3.3 Kromosom Hasil Inisialisasi Populasi

2. Evaluasi individu

Setiap individu dievaluasi dengan cara menghitung nilai fitnessnya dengan menggunakan fungsi:

$$k(i)=\sum_{i=1}^{n-1} Si/Vi \qquad \ldots(3.1)$$

keterangan

Jalan awal: 1

Jalan tujuan : n

S : jarak

V: kecepatan

Dalam kasus ini, akan dicari nilai fitness dari jalan 1 ke jalan 4.

K adalah kromosom yang dibangkitkan

K1 → 0,085 + 0,085 = 0,170 jam

K2 → 0,085 + 0.020 + 0.009 + 0,045 = 0,159 jam

K3 → 0,085 + 0,020 + 0,020 + 0,085 = 0,210 jam

K4 → 0,085 + 0,020 + 0,009 + 0,009 + 0,020 + 0,085 = 0,228 jam

3. Reverse Nilai Fitness

pada kasus ini yang dicari adalah nilai minimal (minimasi), maka reverse fitness ini digunakan untuk mengubah individu-individu yang memberikan nilai semakin kecil untuk fungsi yang dioptimasi menjadi individu - individu yang memberikan nilai fitness semakin besar.

$$F(vk) = \frac{fmax - f(vk) + \alpha}{fmax - fmin + \alpha} \quad \text{...(3.2)}$$

F(vk) adalah fungsi fitness, *f(vk)* adalah original nilai fitness, *vk* kromosom ke k dari generasi saat ini, *f max* dan *f min* adalah nilai maximum dan nilai minimum asli dari fungsi fitness generasi saat ini, α = 0.001 α adalah suatu bilangan yang ditambahkan untuk menghindari nilai fitness 0 .

K adalah kromosom yang di bangkitkan

$$K1 \rightarrow \frac{0{,}228 - 0{,}170 + 0{,}001}{0{,}228 - 0{,}159 + 0{,}001} = 0{,}84$$

$$K2 \rightarrow \frac{0{,}228 - 0{,}159 + 0{,}001}{0{,}228 - 0{,}159 + 0{,}001} = 1{,}00$$

$$K3 \rightarrow \frac{0{,}228 - 0{,}210 + 0{,}001}{0{,}228 - 0{,}159 + 0{,}001} = 0{,}27$$

$$K4 \rightarrow \frac{0{,}228 - 0{,}228 + 0{,}001}{0{,}228 - 0{,}159 + 0{,}001} = 0{,}01$$

4. Roulette Whell

Memilih individu-individu yang akan dijadikan induk pada proses *crossover* ataupun yang akan bertahan hidup pada generasi selanjutnya. Adapun tahapannya adalah sebagai berikut:

a. Hitung total *fitness* (F) seluruh kromosom dalam populasi

$$Tf = f(vk)\ 1 + f(vk)2 .. + f(vk)n \qquad \ldots(3.3)$$

Ket :

Tf : total fitness

F(vk): reverse fitness

$$Tf = 0{,}84 + 1{,}00 + 0{,}27 + 0{,}01 = 2{,}12$$

b. Hitung probabilitas seleksi (pk) setiap kromosom

Dari persamaan (3.2) dan (3.3) didapatkan persamaan

$$Pk(i) = \sum_{i=1}^{n} (f(vk)i/tf) \qquad \ldots(3.4)$$

Ket:

Pk: seleksi probabilitas

F(vk):reverse fitness

Tf:total fitnes

Pk1=0,84/2,12 pk2=1,00/2,12 pk3=0,27/2,12 pk4=0.01/2,12

Pk1= 0.396 Pk2 0.472 Pk3=0.127 Pk4=0.005

c. Hitung probabilitas kumulatif (qk) setiap kromosom

$$qk(j)=\sum_{i=1;j=1}^{n} qkj+pki \qquad \ldots(3.5)$$

qk : probabilitas kumulatif

pk : seleksi probabilitas

qk1=0,396 qk2=0,396 + 0,472 qk3=0,868 + 0,127 qk4=0,995 + 0,005

d. Buat bilangan *random* acak untuk memilih kromosom induk sebanyak jumlah kromosom.

Bilangan random ke-1 → 0,90 : kromosom 3 terpilih karena qk > 0,90

Bilangan random ke-2 → 0,58 : kromosom 2 terpilih karena qk > 0,58

Bilangan random ke-3 → 0,47 : kromosom 2 terpilih karena qk > 0,47

Bilangan random ke-4 → 0,97 : kromosom 4 terpilih karena qk > 0,97

K3	1	8	9	2	4	5	6	3	7
K2	1	8	6	3	4	5	9	2	7
K2	1	8	6	3	4	5	9	2	7
K4	1	8	6	5	9	2	4	3	7

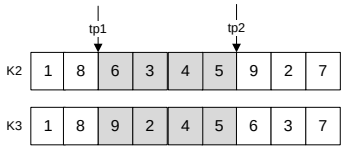
Gambar 3.4 Gambar kromosom hasil seleksi (dalam matting pool)

5. *Crossover*

metode *crossover* yang digunakan adalah *partially mapped crossover*. *Crossover* atau rekombinasi dilakukan apabila probabilitas *crossover* (pc) > nilai *random* yang dibangkitkan. Misalkan dibangkitkan suatu nilai *random* dan nilai *random* tersebut lebih kecil atau sama dengan nilai pc

maka, lakukan *crossover*. Setelah kondisi pc > *random* terpenuhi lakukan pemilihan sepasangan orangtua secara acak, dalam hal ini kromosom K2 dan K3 terpilih untuk dilakukan rekombinasi.

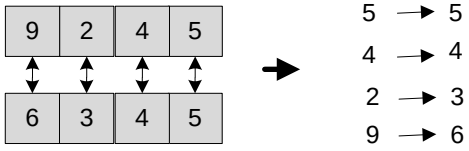
a. Pilih segmen kromosom dari kedua orangtua secara acak dengan cara membangkitkan dua titik, titik potong satu TP1 dan titik potong dua TP2.



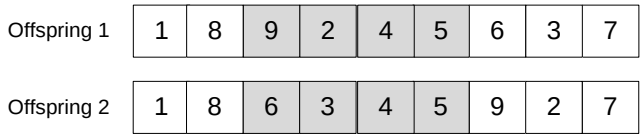
b. Tukar segmen kedua orangtua untuk membuat *proto-child*



c. Lakukan *mapping relationship.*



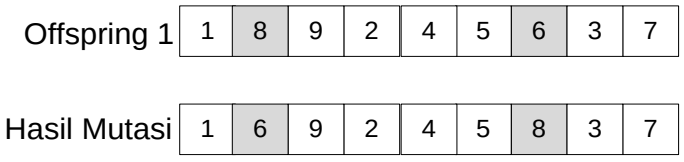
d. Buat offspring yang valid dengan *mapping relationship.*



Gambar 3.5 Ilustrasi rekombinasi PMX

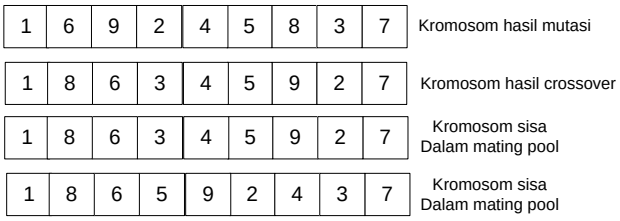
6. Mutasi

   Metode mutasi yang digunakan adalah *swap mutation.* Mutasi dilakukan apabila probabilitas mutasi (pm) > nilai *random* yang dibangkitkan. Misalkan dibangkitkan suatu nilai *random* dan nilai *random* tersebut lebih kecil atau sama dengan nilai pm maka, lakukan mutasi. Misalkan offspring satu terpilih untuk dilakukan mutasi.



Gambar 3.6 Ilustrasi Swap Mutation

7. Populasi baru menggantikan populasi lama dalam proses iterasi (kromosom hasil *crossover*, kromosom hasil mutasi, dan kromosom sisa yang terdapat didalam *mattingpool*).

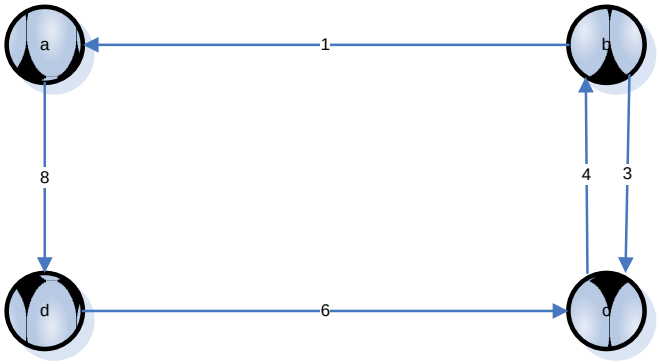


Gambar 3.7 Kromosom Baru Dalam Populasi

8. Proses dihentikan apabila beberapa generasi tertentu tidak ada peningkatan nilai fitness.

9. Hasil dari analisi di atas adalah

1	8	6	3	4	5	9	2	7



Gambar 3.8 Hasil Pencarian Rute Optimal

Dengan hasil fitnessnya adalah

K2 → 0,085 + 0.020 + 0.009 + 0,045 = 0,159 jam

Rute jalan yang di lalui -> Jln. Achmad yani-> Jln. Banda-> Jln. Aceh-> Jln. Abc->

Dikarenakan tidak ada peningkatan nilai fitnessnya.

3.1.3 Analisis Kebutuhan Sistem Non Fungsional

Analisis kebutuhan dilakukan untuk mengetahui spesifikasi kebutuhan untuk sistem. Spesifikasi kebutuhan melibatkan analisis *user*, analisis perangkat keras, dan analisis perangkat lunak.

3.1.3.1 Analisis Pengguna (*User*)

Pengguna dari system ini adalah pendatang , sedangkan administratornya adalah staff dari dinas kebudayaan dan pariwisata kota Bandung

Tabel 3.2 Karakteristik Pengguna

Pengguna	Tanggung Jawab	Hak Akses	Tingkat Pendidikan	Tingkat Keterampilan	Pengalaman
Administrator	Mengecek kevalidan data dan sistem	Menambah, menghapus dan mengedit data	Min. SMA	Bisa mengikuti petunjuk yang ada pada sistem	Min. menguasai bahasa pemrograman dan menguasai tentang database sql server. Pelatihan administrator
Pengguna	- Melakukan tugas yang diberikan	Melakukan kegiatan yang berhubungan dengan isi tampilan yang ada pada program aplikasi	Min. SMA	Bisa mengikuti petunjuk yang ada pada sistem	Min. mampu mengopersikan komputer

Dari penjelasan diatas, dapat diambil kesimpulan bahwa karakteristik *user* yang telah ada saat ini sudah memenuhi kriteria untuk dapat menggunakan program aplikasi yang akan dibangun karena sebagian besar user sudah dapat menggunakan aplikasi yang berbasis komputer.

3.1.3.2 Analisis Kebutuhan Perangkat Keras (*Hardware*)

Dari hasil pengamatan, spesifikasi perangkat keras yang digunakan untuk membantu pengguna dalam menggunakan aplikasi ini adalah sebagai berikut:

Tabel 3.3 Spesifikasi Perangkat Keras

Uraian	**Spesifikasi**
Processor	1243 MHz
Memory	512MB
Hardisk(free space)	5 GB
VGA	64 MB
Monitor	15”

Kebutuhan minimum perangkat keras yang diperlukan untuk mengimplementasikan program aplikasi yang akan dibangun adalah perangkat keras komputer. Semakin tinggi spesifikasi komputer yang digunakan untuk menjalankan aplikasi, akan semakin baik.

3.1.3.3 Analisis Kebutuhan Perangkat Lunak (*Software*)

Perangkat lunak yang digunakan untuk membangun aplikasi ini adalah dengan menggunakan aplikasi microsoft visual studio 2008 yang menggunakan bahasa pemrograman C# yang berfungsi untuk membuat aplikasi dan SQLserver yang berfungsi untuk membangun *database*.

3.2 Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional dilakukan untuk memberikan gambaran aliran data yang ada pada program aplikasi yang akan dibangun.

3.2.1 Identifikasi Aktor

Aktor yang terlibat dalam sistem ini terdiri dari 2 aktor yaitu administrator dan pengguna umum.

3.2.2 *Use case, Scenario,* dan *Activity Diagram*

Usecase menangkap perilaku yang dibutuhkan dan dikehendaki dari suatu sistem (subsistem, kelas, atau antarmuka) yang akan dikembangkan tanpa menspesifikasikan bagaimana perilaku itu akan diimplementasikan. Dalam kasus ini, pada dasarnya *use case* merupakan interaksi khusus antara para aktor (administrator dan pengguna) dan sistem (system pedukung keputusan) untuk menangkap sasaran serta kebutuhan para aktor (*User needs and expectations*).

Usecase diagram untuk aplikasi ini dibagi ke dalam dua kelompok, yakni *usecase* untuk administrator dan *usecase* untuk pengguna. *Usecase* untuk administrator menjelaskan pengolahan data apa saja yang bisa dilakukan oleh administrator sedangkan *usecase* untuk pengguna menjelaskan kegiatan apa saja yang bisa dilakukan oleh pengguna.

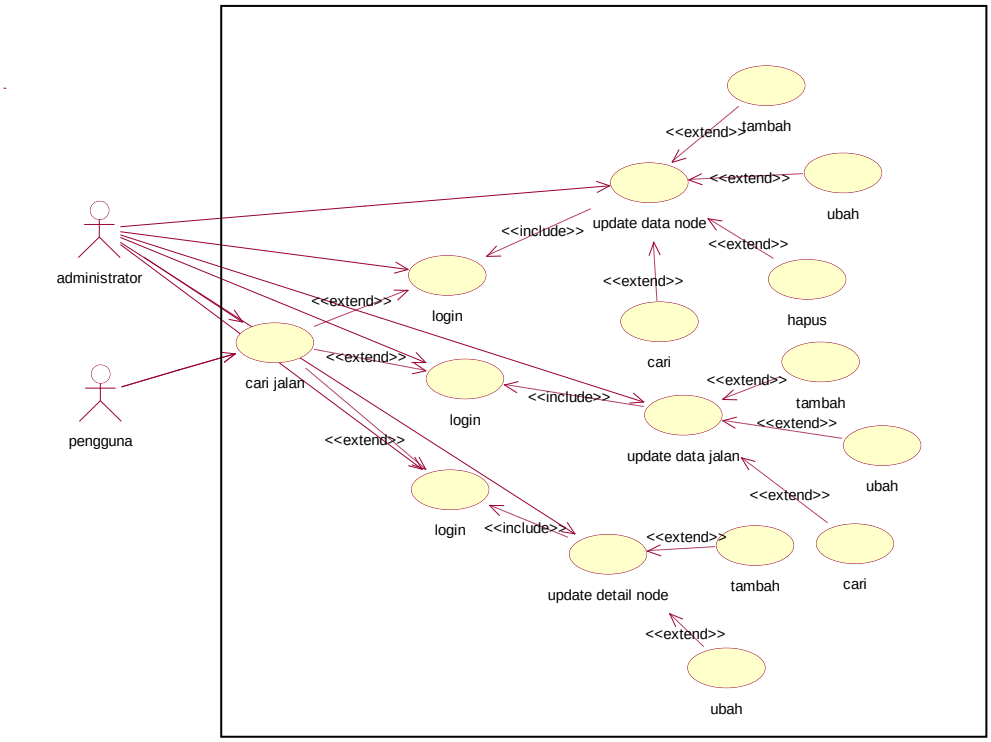
Usecase bekerja dengan skenario. Skenario untuk mendeskripsikan urutan langkah-langkah (proses bisnis-proses bisnis) yang menerangkan apa yang dilakukan aktor terhadap sistem maupun sebaliknya, sistem terhadap aktor.

Acitivity Diagram digunakan untuk memodelkan aspek dinamis dari sistem. *Activity diagram* berfungsi memodelkan alur kerja (*work flow*) sebuah proses

bisnis dan urutan aktifitas pada suatu proses. *Activity diagram* dibuat untuk menggambarkan aktifitas aktor.

1. Diagram *Use Case*

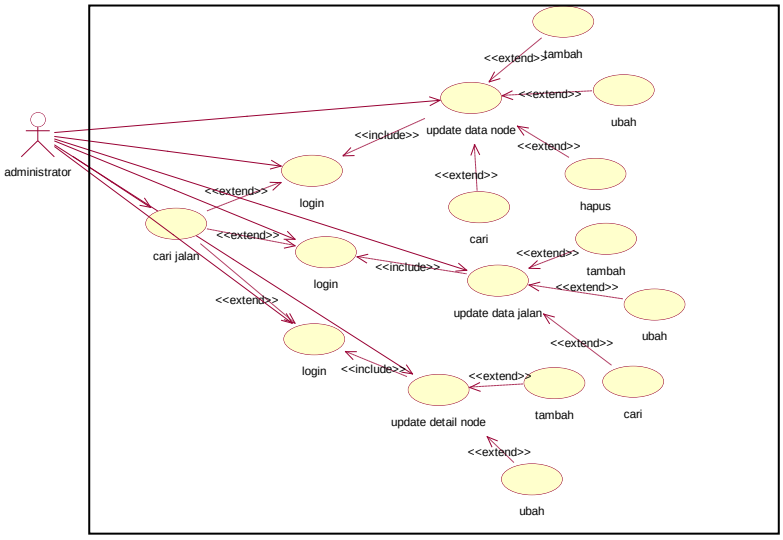
Diagram *use case* untuk semua aktor dapat dilihat pada gambar 3.9.



Gambar 3.9 Diagram *Usecase* semua actor

2. Diagram *Use Case* Administrator

Diagram *use case* untuk administrator dapat dilihat pada gambar berikut.

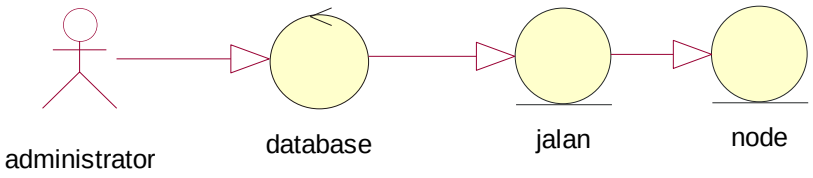


Gambar 3.10 Diagram *Usecase* untuk Administrator

a. *Usecase, Skenario dan Activity Diagram* Cari Jalan



Gambar 3.11 *Usecase* Cari Jalan



Gambar 3.12 Kelas Diagram untuk *Usecase Realitation* Cari Jalan

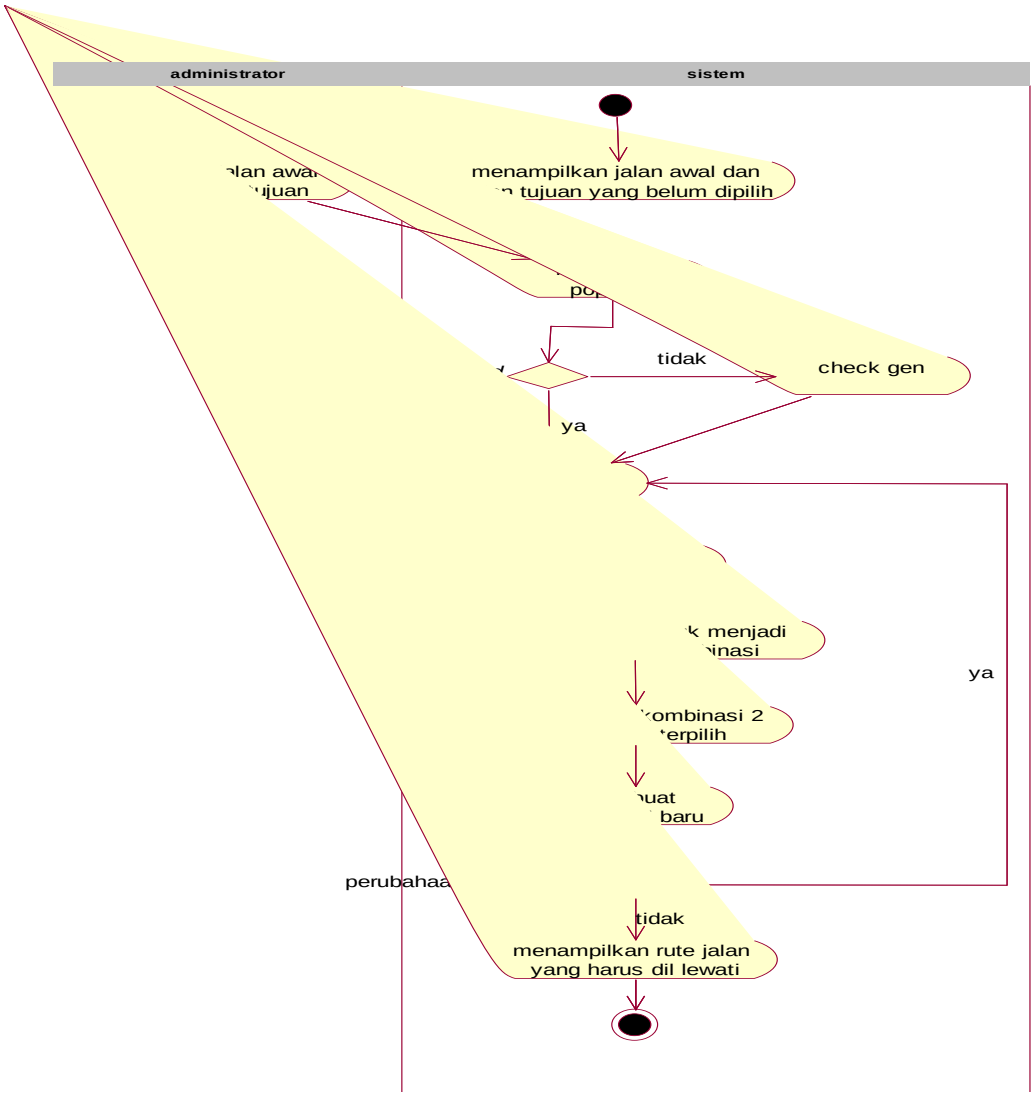
Skenario :

Tabel 3.4 Skenario *Usecase* Cari Jalan

Nama	Cari Jalan
Aktor	Administrator
Tujuan	Mencari rute jalan
Keadaan Awal	Form awal jalan dan tujuan jalan belum terpilih
Usecase yang berkaitan	
Deskripsi : System mencari rute jalan dengan memasukan jalan awal dan jalan tujuan dan system akan menampilkan rute jalan yang harus di lewati	
Langkah-Langkah	
Aksi Aktor	**Aksi Sistem**
1.Mengisi jalan awal dan jalan tujuan	2.mengbangkitkan secara acak sejumlah indivindu sebagai suatu populasi
	3.mengecek apakah suatu jalanterhubung dengan jalan lainya
	4.menperbaiki kromosom bila suatu jalan tidak terhubung dengan jalan selanjutnya
	5.menghitung fitness setiap kromosom
	6.mencari fitness maksimum dari suatu populasi
	7.mencari fitness minimum dari suatu populasi
	8.mengubah nilai fitness minimum dari suatu populasi
	9.melakukan seleksi untuk meilih individu-individu yang akan di jadikan pada proses rekombinasi
	10.melakukan rekombinasi terhadap dua kromosom induk terpilih dengan tujuan menciptakan kromosom anak baru

Aksi Aktor	Aksi Sistem
	11.melakukan mutasi terhadap kromosom terpilih
	12.menbuat generasi baru
	13.menampilkan rute yang harus di lewati

Diagram aktifitas untuk proses Cari Jalan digambarkan pada gambar dibawah ini.

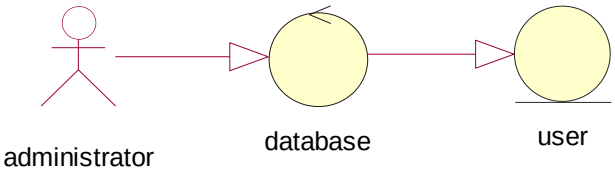


Gambar 3.13 Diagram Aktifitas Cari Jalan

b.*Usecase, Skenario dan Activity Diagram* Login



Gambar 3.14 *Usecase* Login



Gambar 3.15 Kelas Diagram untuk *Usecase Realitation* Login

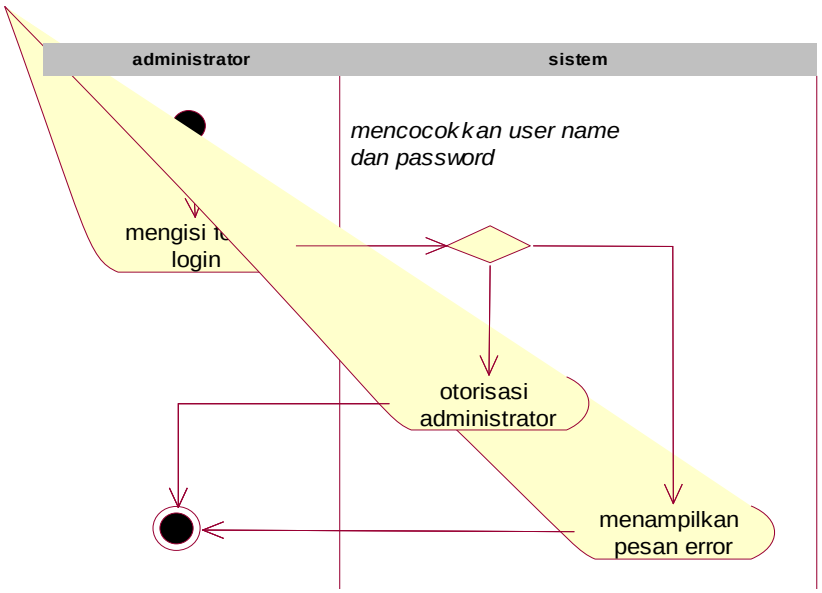
Skenario :

Tabel 3.5 Skenario *Usecase* Login

Nama	Login
Aktor	Administrator
Tujuan	Masuk ke dalam akses Administrator
Keadaan Awal	Form login ditampilkan
Usecase yang berkaitan	
Deskripsi : Proses login administrator merupakan proses untuk memasuki halaman administrator dan menggunakan otoritas administrator yang telah terdaftar.	
Langkah-Langkah	
Aksi Aktor	**Aksi Sistem**
1.Mengisi Form Login	2.Mencocokan Data Login dengan Data Account

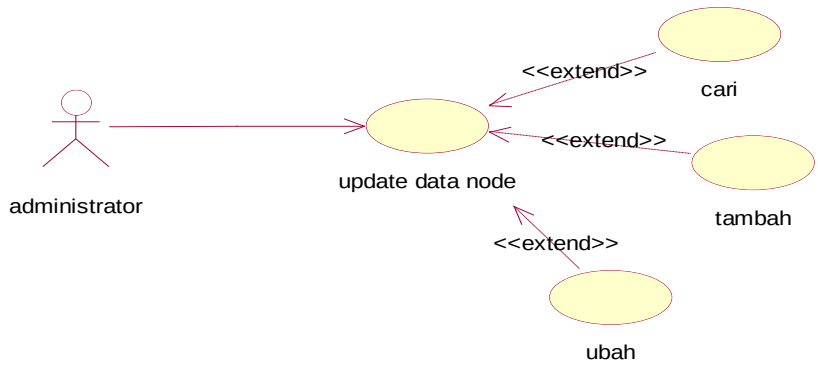
Aksi Aktor	Aksi Sistem
	3.Bila tidak cocok Sistem akan menampilkan Pesan Error Login
	4.bila cocok akan Memberikan Otoritas sebagai Administrator pada User yang melakukan login

Diagram aktifitas untuk proses login digambarkan pada gambar dibawah ini.

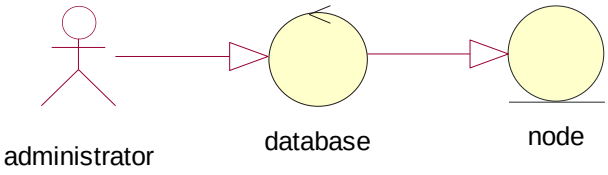


Gambar 3.16 Diagram Aktifitas Login

***c. Usecase, Skenario dan Activity Diagram* Update Data Node**



Gambar 3.17*Usecase* Update Data node



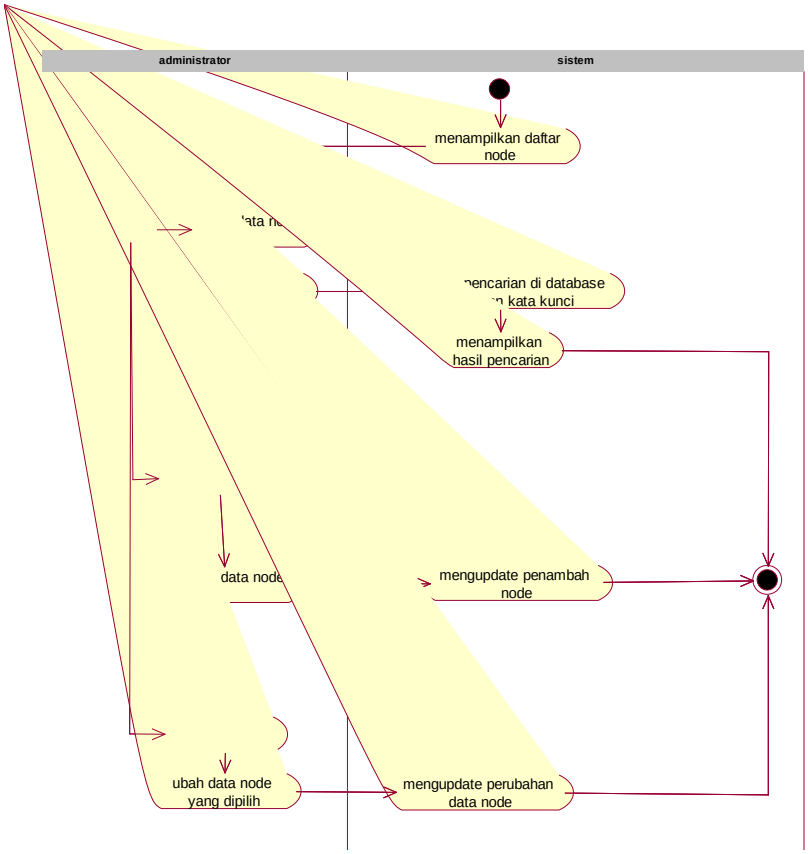
Gambar 3.18Kelas Diagram untuk *Usecase Realitation* update Data node

Skenario :

Tabel 3. 6 Skenario *Usecase* update data node

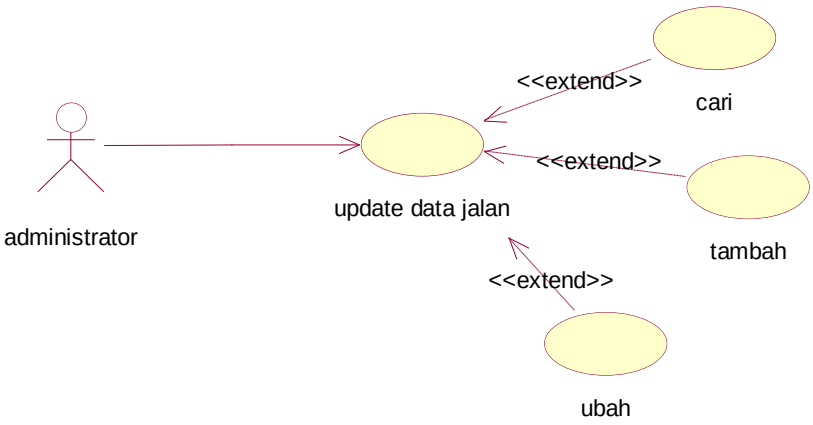
Nama	Update Data node
Aktor	Administrator
Tujuan	Daftar data node teratur dengan baik
Keadaan Awal	Daftar data node
Usecase yang berkaitan	Cari, tambah, ubah
Deskripsi : Sistem melakukan pengaturan pada kumpulan atau daftar node berdasarkan masukan dari admin yang dapat melakukan proses tambah, ubah, hapus dan cari jenis catatan	
Langkah-Langkah	
Aksi Aktor	**Aksi Sistem**
1.Melihat daftar node	2.Menampilkan sesuai pilihan aktor
3.Memilih tindakan yang akan dilakukan (cari, tambah,ubah, hapus)	4.Menampilkan layar pemberitahuan

Diagram aktifitas untuk proses data node digambarkan pada gambar dibawah ini.

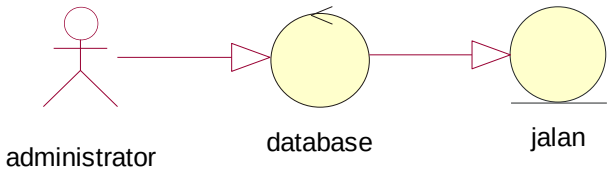


Gambar 3.19Diagram Aktifitas Update Data Node

d. *Usecase, Skenario dan Activity Diagram* Update Data jalan



Gambar 3.20 *Usecase* Paket Update Data jalan



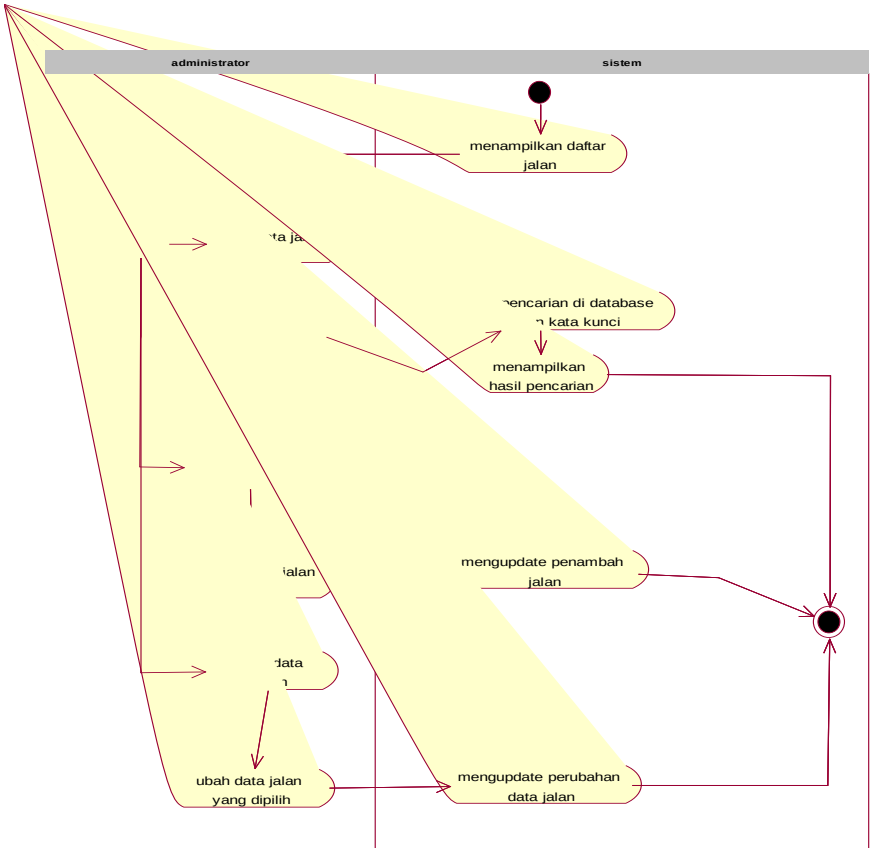
Gambar 3.21 Kelas Diagram untuk *Usecase Realitation* update Data jalan

Skenario :

Tabel 3.7 Skenario *Usecase* update data jalan

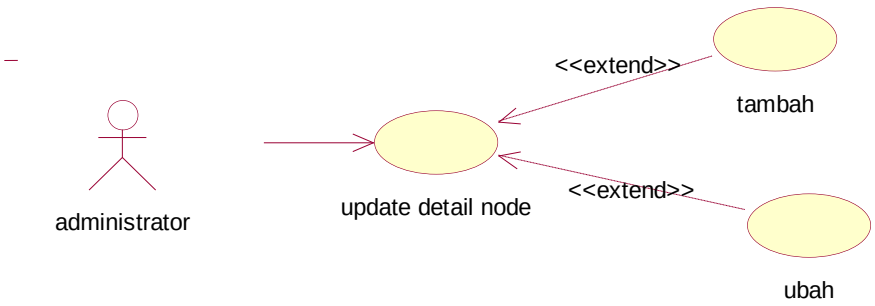
Nama	Update Data jalan
Aktor	Administrator
Tujuan	Daftar data jalan teratur dengan baik
Keadaan Awal	Daftar data jalan
Usecase yang berkaitan	Cari, tambah, ubah
Deskripsi : Sistem melakukan pengaturan pada kumpulan atau daftar jalan berdasarkan masukan dari admin yang dapat melakukan proses tambah, ubah, hapus dan cari jenis catatan	
Langkah-Langkah	
Aksi Aktor	**Aksi Sistem**
1.Melihat daftar jalan	2.Menampilkan sesuai pilihan aktor
3.Memilih tindakan yang akan dilakukan (cari, tambah,ubah)	4.Menampilkan layar pemberitahuan

Diagram aktifitas untuk proses data jalan digambarkan pada gambar dibawah ini.

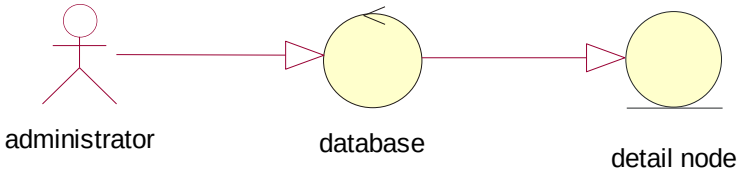


Gambar 3.22 Diagram Aktifitas Update Data Jalan

***e.Usecase, Skenario dan Activity Diagram* Update detail node**



Gambar 3.23 *Usecase* Paket Update detail node



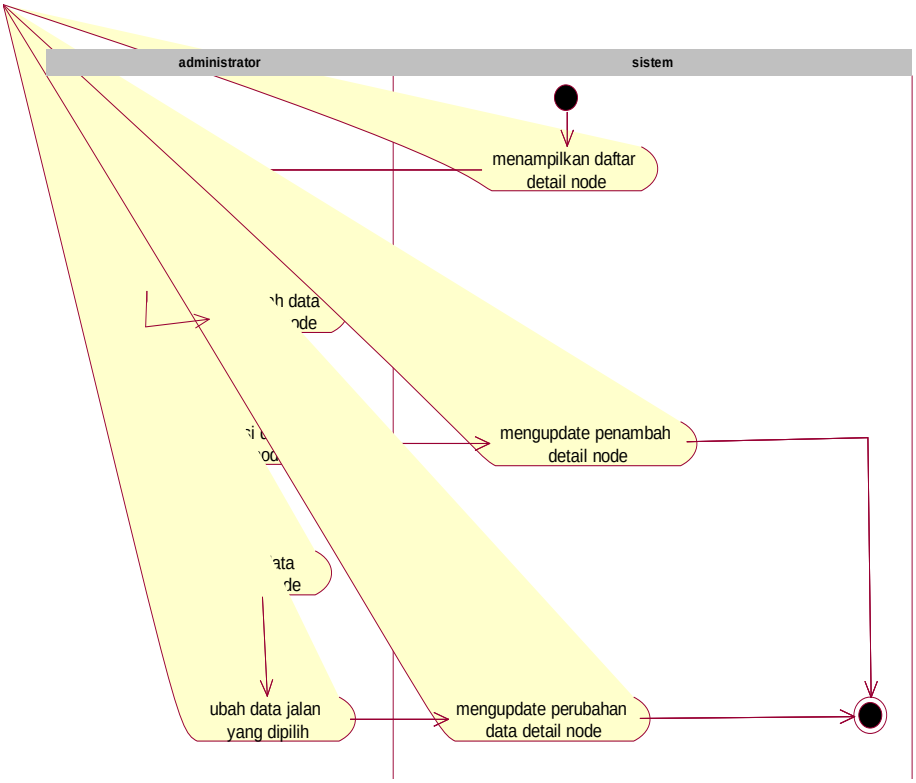
Gambar 3.24 Kelas Diagram untuk *Usecase Realitation* update detail node

Skenario :

Tabel 3.8 Skenario *Usecase* update detail node

Nama	Update detail node
Aktor	Administrator
Tujuan	Daftar data jalan teratur dengan baik
Keadaan Awal	Daftar detail node
Usecase yang berkaitan	tambah, ubah
Deskripsi : Sistem melakukan pengaturan pada kumpulan atau daftar jalan berdasarkan masukan dari admin yang dapat melakukan proses tambah, ubah, hapus dan cari jenis catatan	
Langkah-Langkah	
Aksi Aktor	**Aksi Sistem**
1.Melihat daftar detail node	2.Menampilkan sesuai pilihan aktor
3.Memilih tindakan yang akan dilakukan (tambah,ubah)	4.Menampilkan layar pemberitahuan

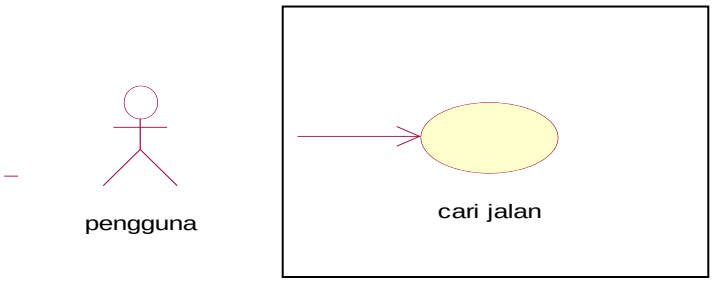
Diagram aktifitas untuk proses data jalan digambarkan pada gambar dibawah ini.



Gambar 3.25 Diagram Aktifitas Update Data detail node

2. Diagram *Use Case* Pengguna

Diagram *use case* untuk pengguna dapat dilihat pada gambar berikut.

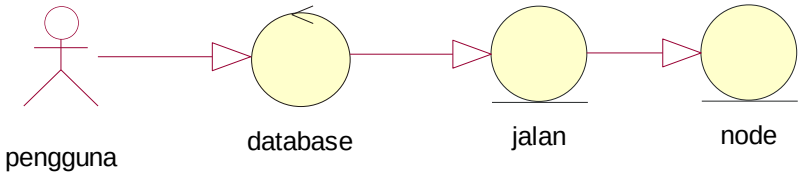


Gambar 3.26 Diagram *Usecase* untuk Pengguna

f.Usecase, Skenario dan Activity Diagram Paket Cari Jalan



Gambar 3.27 *Usecase* Paket Cari Jalan



Gambar 3.28 Kelas Diagram untuk *Usecase Realitation* cari jalan

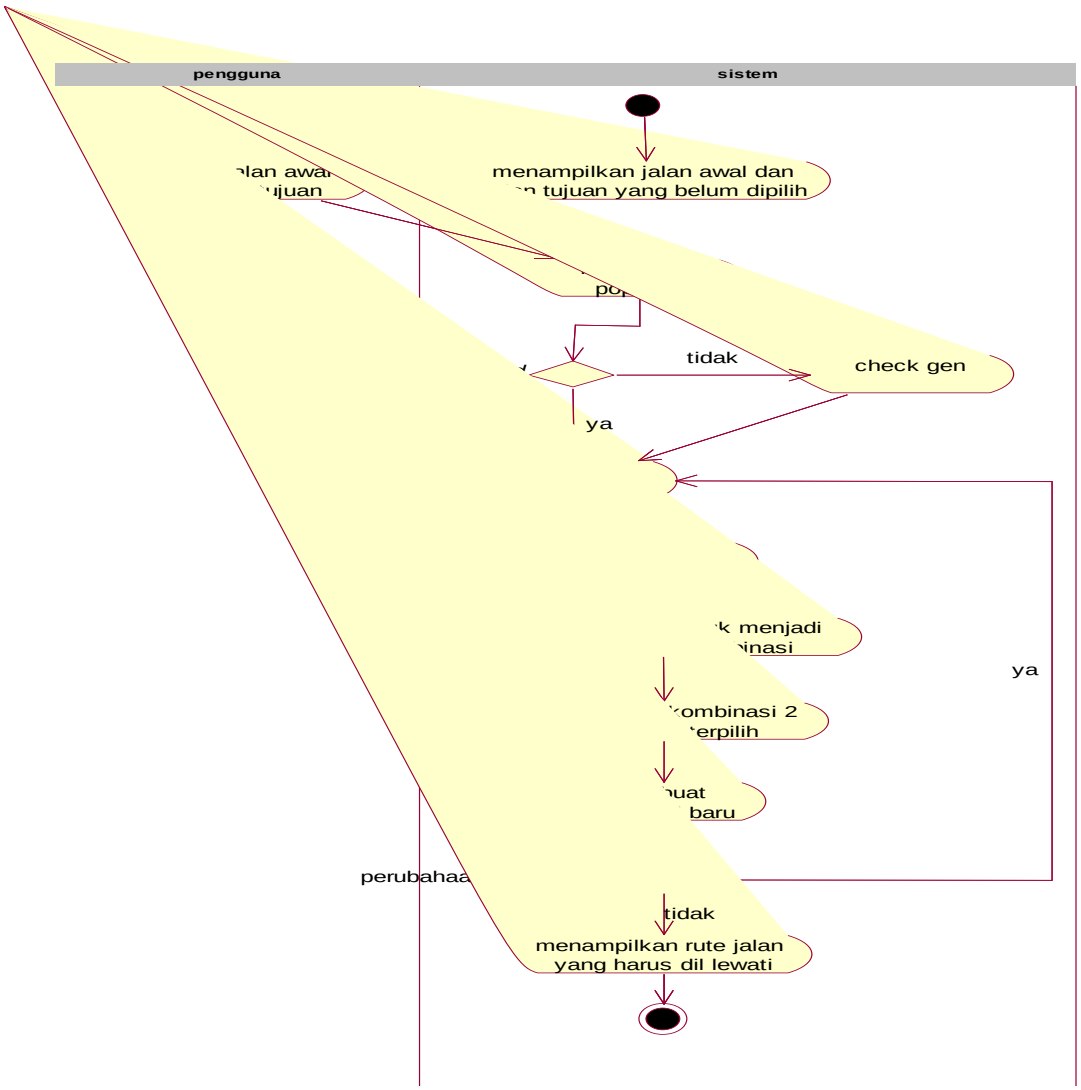
Skenario :

Tabel 3.9 Skenario *Usecase* Cari Jalan

Nama	Cari Jalan
Aktor	Pengguna
Tujuan	Mencari rute jalan
Keadaan Awal	Form awal jalan dan tujuan jalan belum terpilih
Usecase yang berkaitan	
Deskripsi : System mencari rute jalan dengan memasukan jalan awal dan jalan tujuan dan system akan menampilkan rute jalan yang harus di lewati	
Langkah-Langkah	
Aksi Aktor	**Aksi Sistem**
1.Mengisi jalan awal dan jalan tujuan	2.mengbangkitkan secara acak sejumlah individu sebagai suatu populasi

Aksi Aktor	Aksi Sistem
	3.mengecek apakah suatu jalan terhubung dengan jalan lainya
	4.menperbaiki kromosom bila suatu jalan tidak terhubung dengan jalan selanjutnya
	5.menghitung fitness setiap kromosom
	6.mencari fitness maksimum dari suatu populasi
	7.mencari fitness minimum dari suatu populasi
	8.mengubah nilai fitness minimum dari suatu populasi
	9.melakukan seleksi untuk meilih individu-individu yang akan di jadikan pada proses rekombinasi
	10.melakukan rekombinasi terhadap dua kromosom induk terpilih dengan tujuan menciptakan kromosom anak baru
	11.melakukan mutasi terhadap kromosom terpilih
	12.menbuat generasi baru
	13.menampilkan rute yang harus di lewati

Diagram aktifitas untuk proses Cari Jalan digambarkan pada gambar dibawah ini.



Gambar 3.30 Diagram Aktifitas Cari Jalan

3.2.3 Sequence Diagram

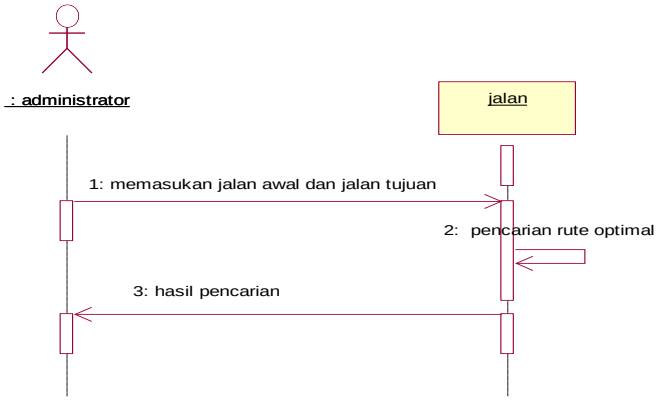
Sequence Diagram digunakan untuk menggambarkan interaksi antara sejumlah object dalam urutan waktu. Kegunaannya untuk menunjukkan rangkaian pesan yang dikirim antara *object* juga interaksi antar object yang terjadi pada titik tertentu dalam eksekusi sistem [SOF[10]].

Sama seperti *usecase diagram*, *sequence diagram* aplikasi system pendukung keputusan ini juga dibagi menjadi dua kelompok, yaitu untuk Administrator dan Pengguna.

3.2.3.1 Sequence Diagram Administrator

a. Pencarian jalan

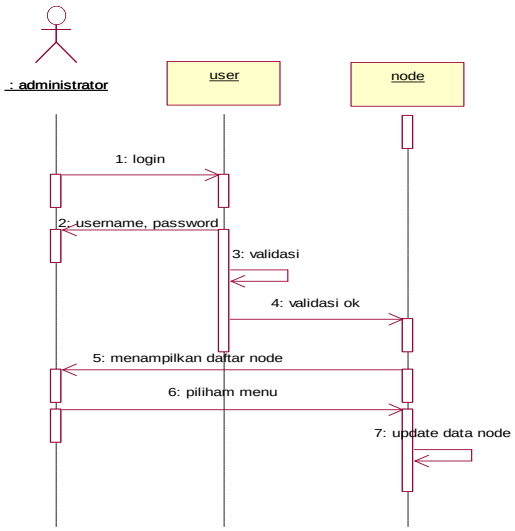
Berikut diagram urutan dari proses pencarian jalan dari administrator. Administrator masukan jalan awal dan jalan tujuan, sistem melakukan pencarian rute, setelah itu menampilkan hasil pencarian rute.



Gambar 3.31 Diagram Urutan Cari Jalan

b. Update data node

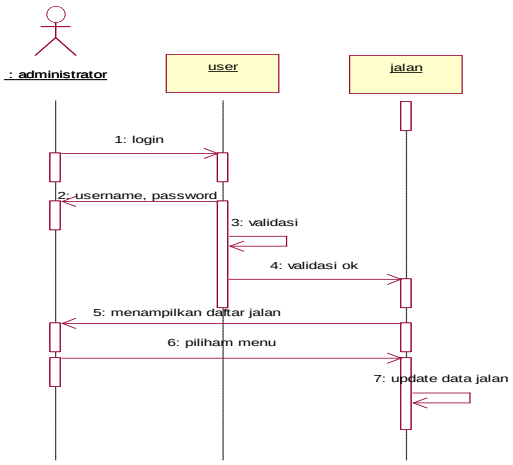
Berikut diagram urutan dari proses update data node dari administrator. Sebelum masuk ke halaman update node, user harus login terlebih dahulu, jika valid maka bisa mengakses data *node* dan dapat mengupdatenya.



Gambar 3.32 Diagram Urutan Update Data Node

c. Update data jalan

Berikut diagram urutan dari proses update data jalan dari administrator. Sebelum masuk ke halaman update jalan, user harus login terlebih dahulu, jika valid maka bisa mengakses data *jalan* dan dapat mengupdate *jalan*.

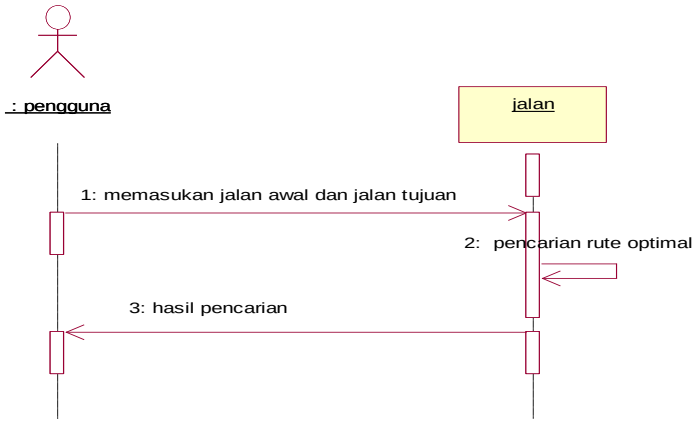


Gambar 3.33 Diagram Urutan Update Data Jalan

3.2.3.2 Sequence Diagram Pengguna

d.Pencarian rute

Berikut diagram urutan dari proses pencarian jalan dari pengguna. Pengguna masukan jalan awal dan jalan tujuan, sistem melakukan pencarian rute, setelah itu menampilkan hasil pencarian rute**.**



Gambar 3.34 Diagram Urutan Cari Jalan

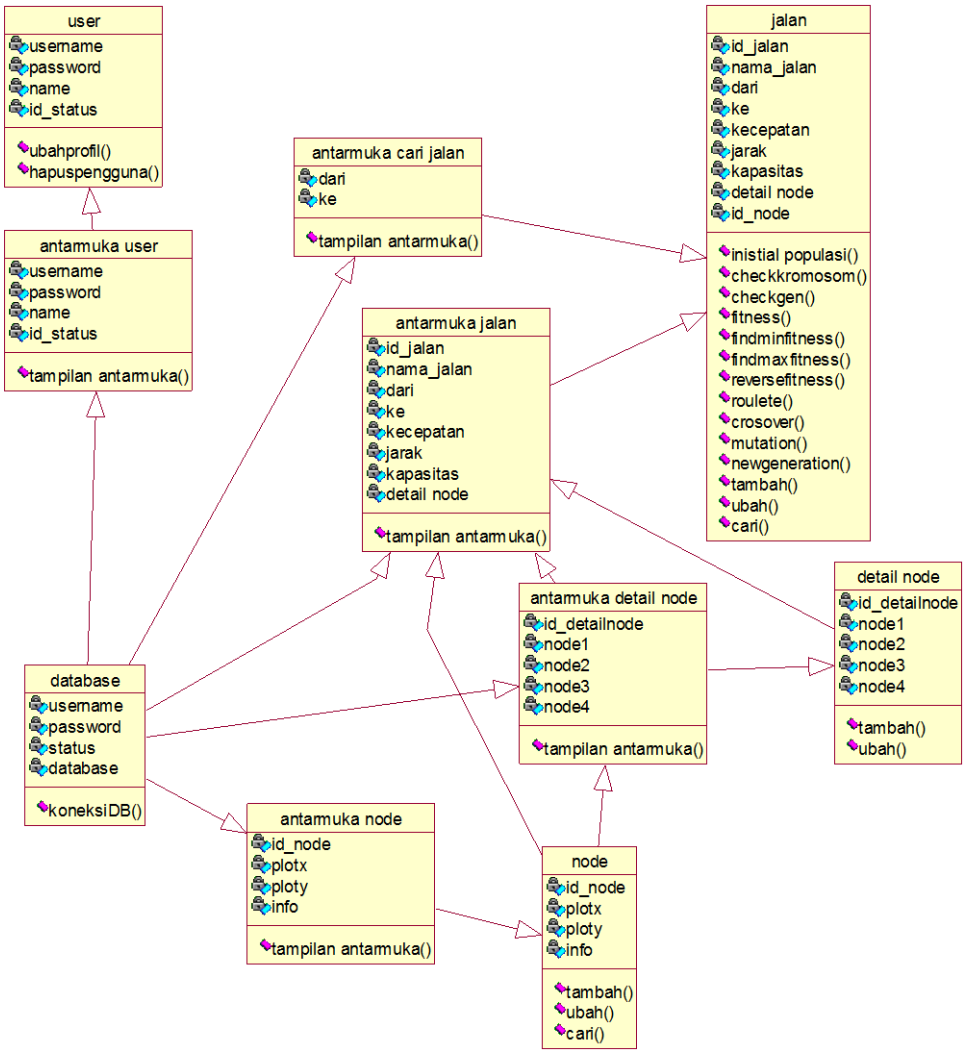
3.2.4 *Class Diagram*

Class Diagram berfungsi untuk menggambarkan relasi atau hubungan antar *Class*. Diagram kelas sendiri terbagi menjadi tiga, yaitu diagram kelas untuk *interface*, control, dan data.

Tabel 3.10 Jenis Kelas Diagram

Nama Kelas	**Jenis Kelas**
Database	*Control*
Antarmuka Jalan	*interface*
Antarmuka Cari Jalan	*interface*
Jalan	*Entity*
Antarmuka Node	*Interface*
Node	*Entity*
Antarmuka User	*Interface*
User	*Entity*
Antarmuka detail node	*Interface*
Detail node	*Entity*

Berikut ini diagram kelas dari aplikasi ini disertai tabel keterangan jenis dari masing masing kelas.



Gambar 3.35 *Class Diagram* Aplikasi

3.3 Perancangan Sistem

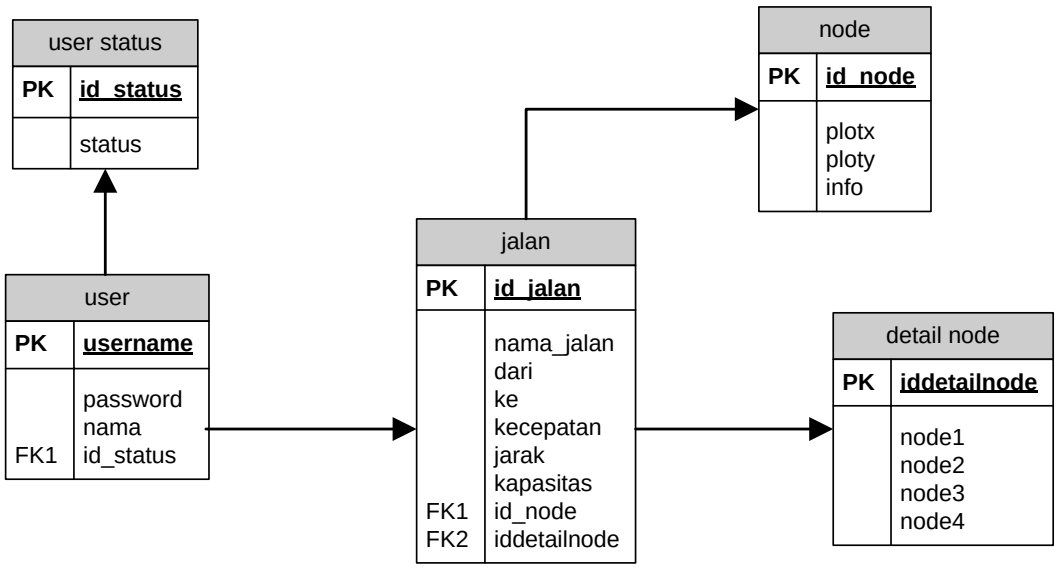
Tahap perancangan merupakan kelanjutan dari proses analisis dimana dilakukan perubahan-perubahan terhadap sistem yang sedang berjalan. Hal ini dilakukan untuk mengatasi kekurangan yang ada, memudahkan pekerjaan yang dilakukan oleh orang yang terlibat dan menghemat waktu pekerjaan.

Dalam perancangan sistem aplikasi ini meliputi perancangan data, perancangan arsitektur, perancangan antarmuka dan perancangan prosedural. Semua perancangan yang telah disebutkan diatas harus saling berhubungan agar menciptakan suatu sistem yang baik. Jika perancangan diatas berhubungan dengan baik maka aplikasi yang dibuat akan berjalan dengan baik.

3.3.1 Perancangan Data

3.3.1.1 Tabel Relasi

Relasi antar tabel merupakan gabungan antar *file* yang mempunyai kunci utama yang sama, sehingga *file-file* tersebut menjadi satu kesatuan yang dihubungkan oleh *field* kunci (*Primary Key*). Pada proses ini elemen-elemen data dikelompokkan menjadi satu *file database* beserta *entitas* dan hubungannya.



Gambar 3.36Tabel Relasi

3.3.1.2 Struktur Tabel

Didalam pembuatan program dibutuhkan suatu *spesifikasi file* yang dimaksudkan agar dapat melakukan kegiatan-kegiatan dalam pengaturan pencarian data yang dapat memudahkan sistem omputer. Untuk itu sistem pengolahan data ini membutuhkan *spesifikasi file* untuk mempermudah dalam pembuatan program. Tabel-tabel yang dibuat adalah sebagai berikut:

1. **Tabel User Status**

Tabel 3.10 Tabel User Status

Atribut	Tipe data	Panjang	Keterangan
Id_status	Int		*
status	Varchar	30	

2. **Tabel User**

Tabel 3.11 Tabel User

Atribut	Tipe data	Panjang	Keterangan
username	Varchar	50	*
password	Varchar	10	
nama	Varchar	30	
Id_status	Int		**

3. **Tabel Node**

Tabel 3.12 Tabel Node

Atribut	Tipe data	Panjang	Keterangan
id_node	Int	4	*
plotx	Float	8	
ploty	Float	8	
info	Varchar	10	

4. **Tabel Detail Node**

Tabel 3.13 Tabel detail node

Atribut	Tipe data	Panjang	Keterangan
iddetailnode	Int	4	*
Idnode1	Varchar	8	
Idnode2	Varchar	8	
Idnode3	Varchar	10	
Idnode4	Varchar		

5. Tabel Jalan

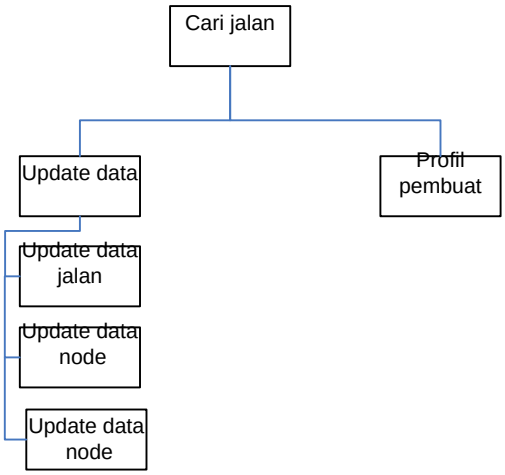
Tabel 3.14 Tabel Jalan

Atribut	Tipe data	Panjang	Keterangan
Id_jalan	Int	4	*
Nama_jalan	Varchar	30	
dari	Int	4	**
ke	Int	4	**
kecepatan	Float	8	
jarak	Float	10	
kapasitas	int		
Id_detail node	varchar		**
Id_node	int	4	**

3.3.2 Perancangan Sruktur Menu

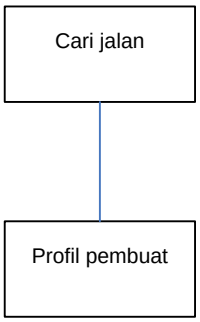
Berikut adalah perancangan struktur menu program dari aplikasi system pendukung keputusan rute optimal kota bandung:

a. Perangcangan struk menu administrator



Gambar 3.37 Struktur Menu Administrator

b. Perancangan struktur menu penguna

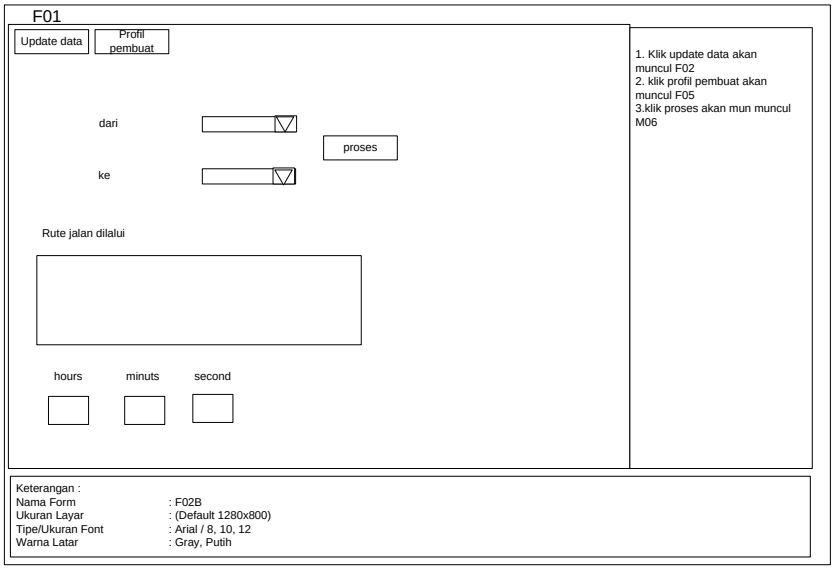


Gambar 3.38Struktur Menu Pengguna

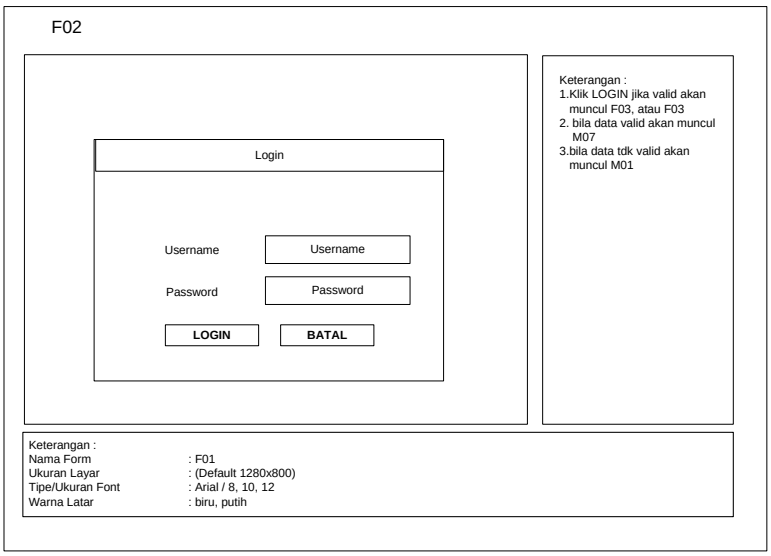
3.3.3 Perancangan Antar Muka

3.3.3.1 Perancangan Tampilan

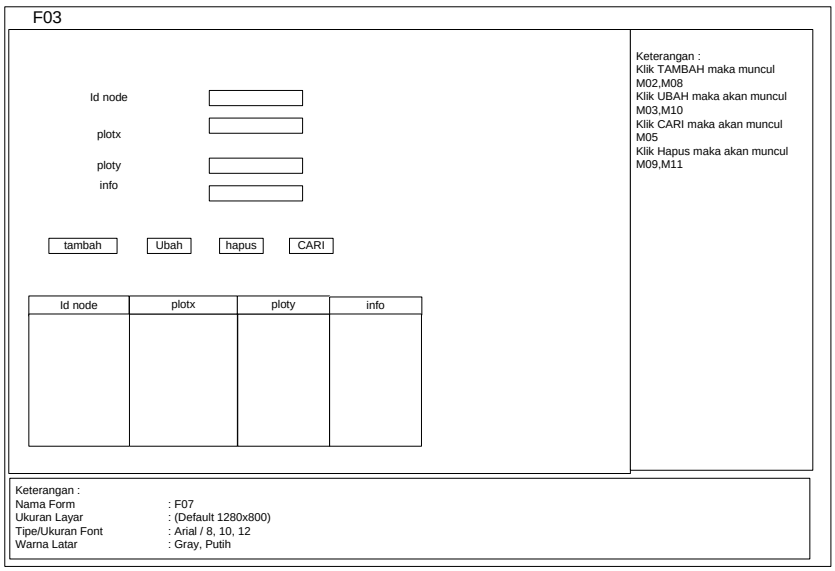
Perancangan tampilan aplikasi bertujuan untuk memberikan gambaran tentang aplikasi yang akan dibangun. Sehingga akan mempermudah dalam mengimplementasikan aplikasi. Dan juga akan memudahkan pembuatan aplikasi yang *user friendly*. Perancangan ini akan diimplementasikan menjadi sebuah program yang utuh. Rancangan aplikasi yang akan dibuat sebagai berikut :



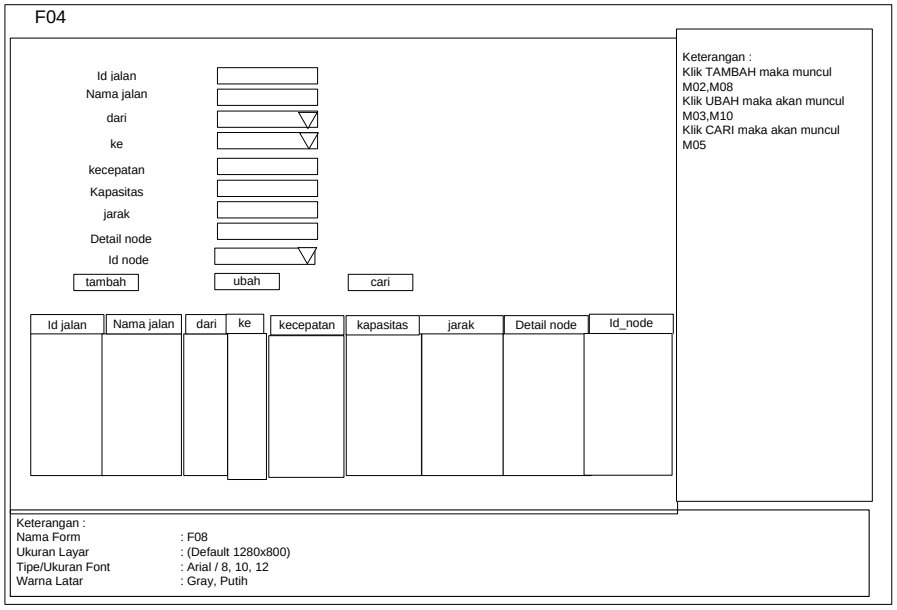
Gambar 3.39Tampilan Halaman Cari Jalan



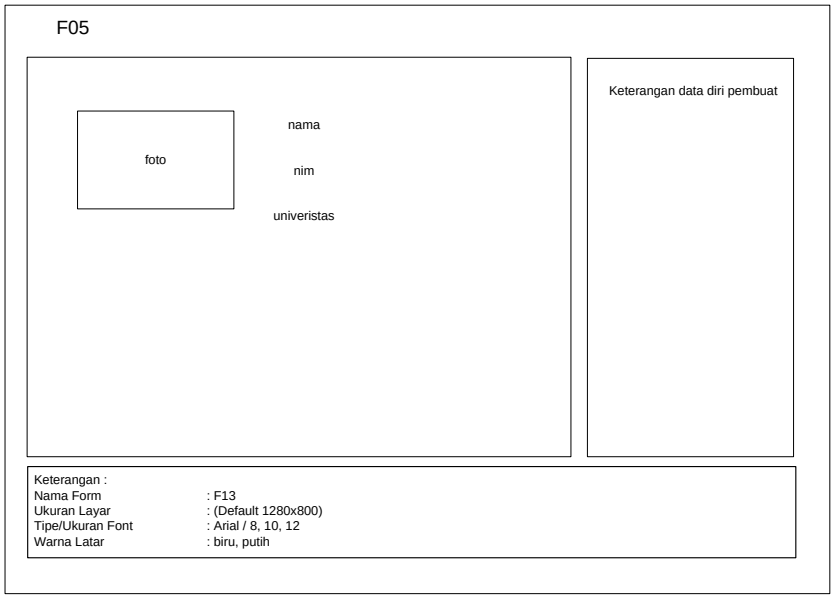
Gambar 3.40Tampilan Halaman Login



Gambar 3.41Tampilan Halaman update data node



Gambar 3.42Tampilan Halaman update data jalan

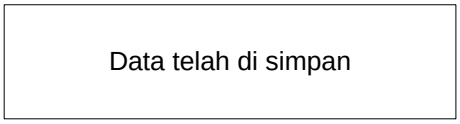


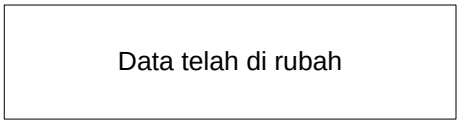
Gambar 3.43 Tampilan Halaman profil pembuat

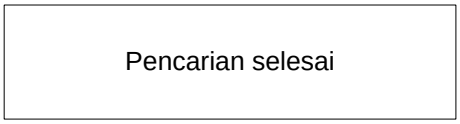
3.3.3.2 Perancangan Pesan

Dibawah ini adalah perancangan pesan yang ada pada aplikasi Sistem pedukung keputusan rute optimal kota bandung

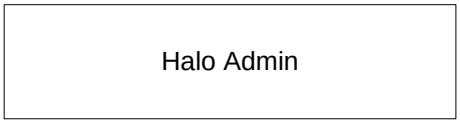
<table>
<tr>
<td>M01

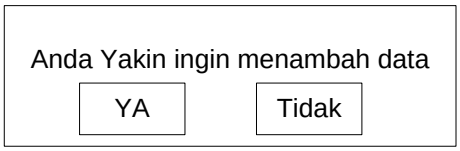

Gambar 3.44 Tampilan Pesan M01</td>
<td>M02


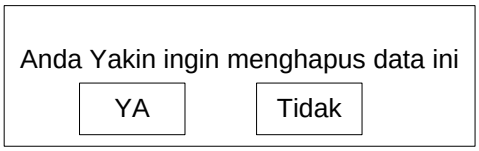
Gambar 3.45 Tampilan Pesan M02</td>
</tr>
<tr>
<td>M03


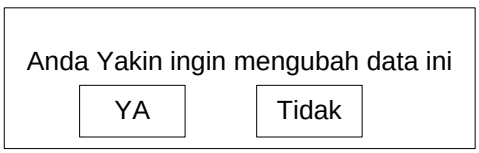
Gambar 3.46 Tampilan Pesan M03</td>
<td>M05


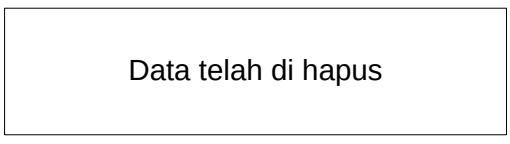
Gambar 3.47 Tampilan Pesan M05</td>
</tr>
<tr>
<td>M06


Gambar 3.48 Tampilan Pesan M06</td>
<td>M07


Gambar 3.49Tampilan Pesan M07</td>
</tr>
<tr>
<td>M08


Gambar 3.50 tampilan pesan M08</td>
<td>M09


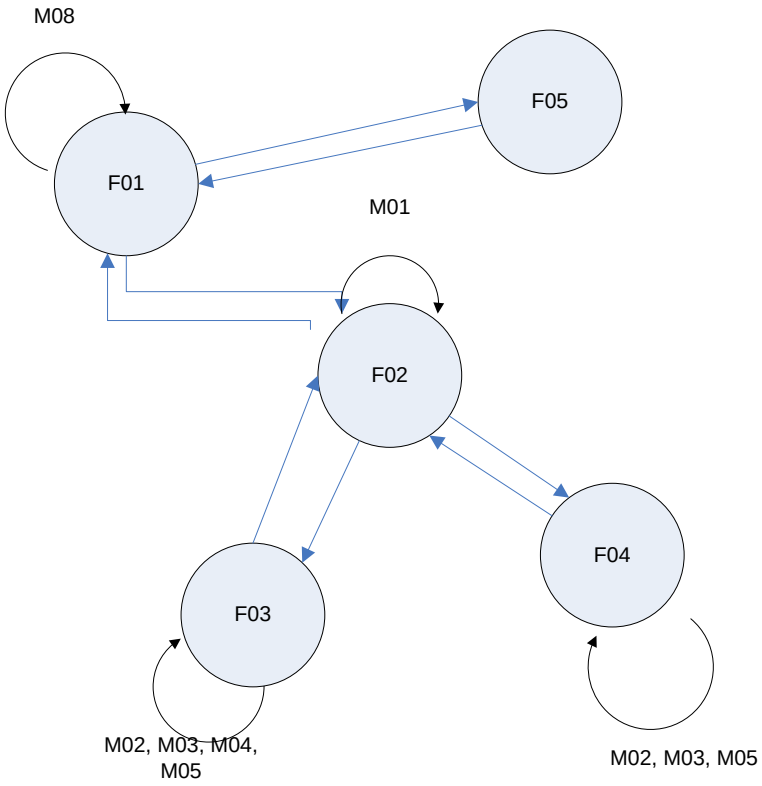
Gambar 3.51 tampilan pesan M09</td>
</tr>
<tr>
<td>M10


Gambar 3.52 tampilan pesan M10</td>
<td>M11


Gambar 3.53 tampilan pesan m11</td>
</tr>
</table>

3.3.3.3Jaringan semantic

3.3.3.3.1 jaringan semantik antar muka menu administrator

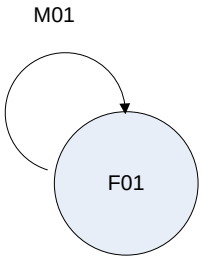


Gambar 3.54Diagram Semantik Administrator

Untuk penjelasan dari jaringan semantik yang ada dibawah ini akan diuraikan sebagai berikut :

a. F01: halaman cari jalan login
b. F02: halaman login
c. F03: halaman update data node
d. F04: halaman update data jalan
e. F05: halaman data pembuat

3.3.3.3.2jaringan semantik antar muka menu pengguna



Gambar 3.55 Diagram Semantik Pengguna

Untuk penjelasan dari jaringan semantik yang ada dibawah ini akan diuraikan sebagai berikut :

a. F01: Halaman Cari Jalan