BAB 3
ANALISIS DAN PERANCANGAN SISTEM

3.1. Analisis Sistem

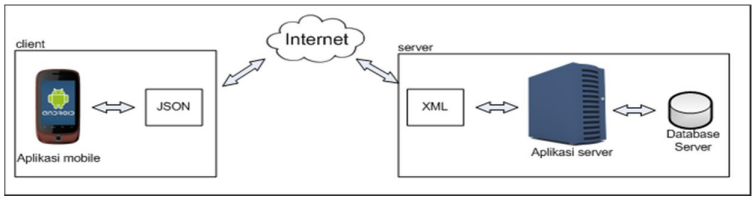
Analisis sistem dapat didefinisikan sebagai penguraian dari suatu sistem informasi yang utuh kedalam bagian-bagian komponennya dengan maksud untuk mengidentifikasi dan mengevaluasi permasalahan-permasalahan, kesempatan-kesempatan, hambatan-hambatan yang terjadi dan kebutuhan-kebutuhan yang diharapkan sehingga dapat diusulkan perbaikan-perbaikannya.

3.1.1. Analisis Masalah

PT PLN (Persero) adalah BUMN yang mengurusi aspek kelistrikan yang ada di Indonesia. Berdasarkan hasil pengamatan yang dilakukan di PT. PLN (Persero) Rayon Bandung Timur yaitu kurang koordinasi dalam penugasan kinerja keluhan dan gangguan yang ada, dan terjadinya permasalahan tumbukan laporan keluhan dan gangguan pelanggan. Selain itu, belum adanya pemecahan masalah dengan solusi yang optimal pada persoalan penjadwalan dimana solusi yang ingin dicapai adalah berdasarkan waktu dan jarak tempuh yang minimal sehingga rute perjalanan ke lokasi perbaikan sehinggal penjadwalan keluhan dan gangguan pelanggan menjadi lebih optimal.

3.1.2. Analisis Arsitektur Sistem

Pada penelitian ini, sistem menerapkan arsitektur sistem yang terdiri dari client dan server.Arsitektur sistem yang digunakan dalam merancang sistem ditunjukan pada gambar 3.1.



Gambar 3.1. Arsitektur Sistem

Aplikasi *client* merupakan antarmuka antara pengguna dengan sistem. Pada bagian *client* terdapat dua bagian yaitu JSON dan aplikasi *mobile.* JSON merupakan bagian dari sistem yang terdapat pada sisi *client*, dimana berfungsi untuk mengumpulkan data yang terdapat di database server dengan cara mengalokasikan data yang didapat dari database server ke dalam masing-masing variabel. Aplikasi *mobile* merupakan sistem yang berfungsi untuk menampilkan semua informasi yang terdapat di *database server*dengan cara menampilkan informasi yang sudah dialokasikan ke dalam variabel-variabel yang terdapat di dalam JSON.

Server merupakan lapisan yang berfungsi sebagai pengelola data yang terdapat pada sistem serta sebagai penghubung antara aplikasi *client* dengan *database server*. Pada perancangan sistem, lapisan ini dibagi menjadi tiga bagian yaitu aplikasi XML, aplikasi *server* dan *database server*. XML merupakan pemformatan data dengan mengalokasikan data yang di-*parsing* ke dalam variabel sesuai dengan variabel yang dibutuhkan, pemformatan data ini berguna dalam pengambilan data oleh JSON. Aplikasi *server* bertanggung jawab dalam mengelola data serta sebagai penghubung aplikasi *client* dan *database server*. *Database server* merupakan aplikasi yang berfungsi menyimpan data-data yang dibutuhkan sistem.

3.1.3. Analisis Sistem Yang Sedang Berjalan

Analisis sistem yang berjalan bertujuan untuk memberikan gambaran tentang proses yang selama ini dilakukan. Proses penjadwalan yang dilakukan selama ini yaitu masih dilakukan secara manual dengan menggunakan *Microsoft Excel* dan *dashboard*. Adapun prosedur yang dilakukan dalam membentuk penjadwalan YANTEK (pelayanan teknik) untuk melayani keluhan pelanggan di Rayon Bandung Timur akan digambarkan dengan diagram alur kerja (Flow Gangguan Teknis). Untuk lebih jelasnya gambaran tentang prosedur penjadwalan dapat dilihat pada Flow Gangguan Teknis pada Gambar 3.2.

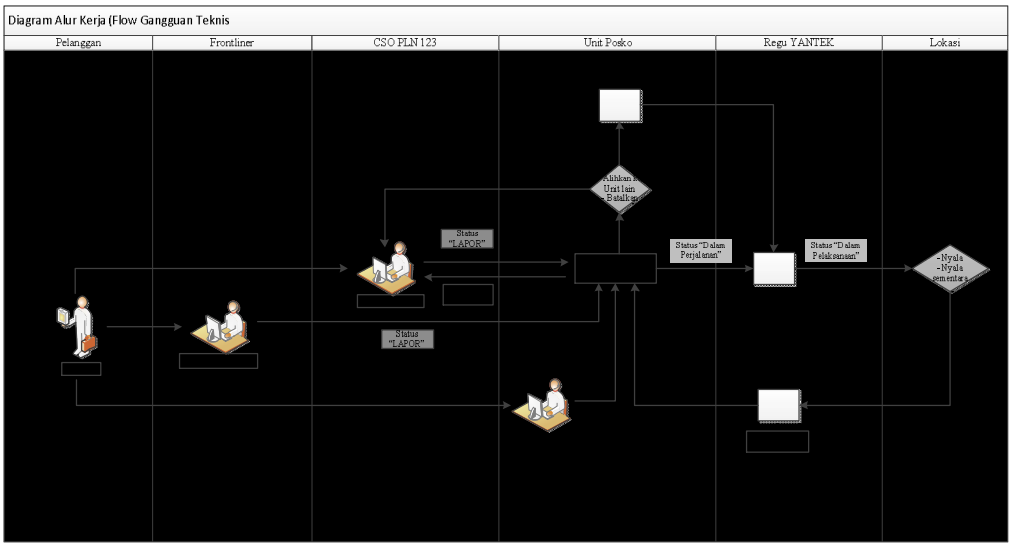
3.1.3.1. Diagram Alur Kerja Yang Sedang Berjalan

Diagram alur kerja merupakan bagan yang menggambarkan dan menjelaskan urutan prosedur-prosedur, arus pekerjaan secara keseluruhan dari system dan menggambarkan aliran data atau dokumen dari satu entitas ke entitas lain.

Diagram alur kerja penjadwalan yang sedang berjalan di PT PLN (Persero) Rayon Bandung Timur saat ini adalah sebagai berikut:

1. Pelanggan memberikan keluhan dan gangguan melalui Fronliner, CSO PLN 123 atau langsung ke unit posko YANTEK
2. Data keluhan pelanggan di inputkan ditiap bagian masing-masing tergantung dari laporan pelanggan yang masuk melalui Frontliner, CSO PLN 123 atau kebagian unit posko YANTEK
3. Frontliner memberikan status “LAPOR” kepada pihak unit posko
4. CSO PLN 123 memberikan status “LAPOR” kepada pihak unit posko
5. Data laporan yang sudah masuk akan di update status dan di input dibagian unit posko YANTEK
6. Jika selesai, maka memberikan status “Selesai” kepada CSO PLN 123
7. Unit posko dapat mengalihkan ke unit lain atau membatalkan proses laporan dari pelanggan
8. Jika di alihkan maka unit posko memberikan status “Dalam perjalanan” ke regu YANTEK
9. Jika dibatalkan, unit posko lapor kembali ke CSO PLN 123
10. Regu YANTEK status “Dalam Pelaksanaan” ke lokasi
11. Di lokasi status sudah nyala atau nyala sementara
12. Regu YANTEK mencatat data lapangan
13. Selanjutnya regu YANTEK melapor dan kembali ke unit posko

Proses kegiatan pada sistem perbaikan yang saat ini sedang dilaksanakan dapat dilihat pada Flow Gangguan Teknis pada Gambar 3.2



Gambar 3.2 *Flow* Gangguan teknis Perbaikan Yang Sedang Berjalan

Setelah melakukan analisis terhadap prosedur yang sedang berjalan, dapat dilihat bahwa masih adanya penjadwalan ke lokasi-lokasi gangguan pelanggan unit posko masih memberikan instruksi atau penugasan ke regu YANTEK secara manual dengan bantuan radio *transmitter* sehingga belum optimalnya penjadwalan penugasan regu YANTEK menangani gangguan yang ada secara cepat dan efisien.

Solusi yang diajukan yaitu penjadwalan dilakukan dengan menggunakan penerapan algoritma genetika dalam persoalan TSP (*Travelling Salesman Problem*), agar penjadwalan dapat terjadwal dengan baik dan optimal.

3.1.4. Analisis Kebutuhan Data

Sistem yang akan dibangun merupakan sistem untuk membantu YANTEK dalam mendapatkan informasi penjadwalan. Untuk proses pengolahan data dapat dilakukan melalui aplikasi *backend* yaitu pada *web server*.

Http Connection digunakan sebagai penghubung antara aplikasi android ke *database* pada *web server*, sehingga *client* pada aplikasi mobile android dapat melakukan proses pemanggilan *method* atau fungsi untuk mendapatkan data berupa informasi penjadwalan yang dikelola melalui aplikasi *backend*.

3.1.4.1. Data Masukan

Data masukan dari aplikasi penjadwalan keluhan dan gangguan jaringan listrik di PT PLN (Persero) Rayon Bandung Timur yang digunakan terdiri dari data gardu, data lokasi, data jenis gardu, data jenis gangguan. Rincian data di atas adalah sebagai berikut:

1. **Data Gardu**

   Data gardu yang ada di PT PLN (Persero) Rayon Bandung Timur dapat dilihat pada **Lampiran 3.1 Data Gardu.**

2. **Data Lokasi**

   Data lokasi yang dimiliki oleh PT PLN (Persero) Rayon Bandung Timur adalah lokasi dimana gardu tersebut berada yang memiliki titik kordinat tiap lokasi gardu yang akan dilakukan perbaikan oleh tiap unit perbaikan YANTEK. Untuk lebih detailnya bisa dilihat di **Lampiran 3.2 Data Lokasi.**

3. **Data Jenis Gangguan**

   Berikut beberapa jenis gangguan tiap bulannya yang sering di tangani oleh pihak unit YANTEK di PT PLN (Persero) Rayon Bandung Timur, diantaranya sebagai berikut:

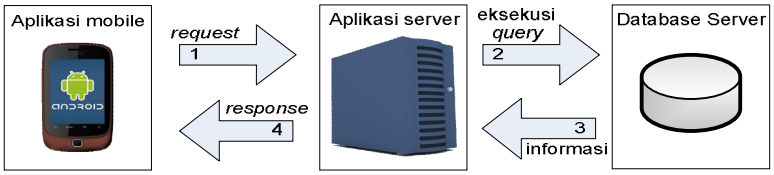
Tabel 3.1 Data Jenis Gangguan

NO	JENIS GANGGUAN	ESTIMASI PERBAIKAN (MENIT)
1	PENYULANG	20
1	TRAFO	30
2	SUTR PUTUS/KEBAKAR	50
3	SR PUTUS/RUSAK	60
4	JUMPER SUTR PUTUS/LOSKONTAK	35
5	JTR MELOROT	45
6	LOSKONTAK TIANG	40
7	LOSKONTAK TA	25
8	LOSKONTAK OK	35
9	LOSKONTAK METER	45
10	LOSKONTAK PEMBATAS	15
11	TERMINAL OK TERBAKAR	15
12	KABEL NAIK	25
13	PEMBATAS RUSAK (MCB)	90
14	KWH METER RUSAK	15

NO	JENIS GANGGUAN	ESTIMASI PERBAIKAN (MENIT)
15	PELEBUR PUTUS (NHF)	120
16	HANTARAN DALAM RUSAK	75
17	INSTALASI	45
18	FUSE BASE	10
19	TOKEN ERROR	20
20	LAIN - LAIN	-

3.1.5. Analisis Komunikasi Data

Analisis komunikasi data menjelaskan cara pertukaran data antara *client* dan *server*. Dalam penelitian ini pertukaran data menggunakan teknik *HTTP request*, dimana aplikasi akan mengirim sebuah *request* pada suatu aplikasi *server* melalui jaringan *internet* lalu akan menerima *response* dari aplikasi *server* tersebut. Mekanisme komunikasi data dalam perancangan Aplikasi penjadwalan pelayanan keluhan pelanggan dapat dilihat pada gambar 3.3.



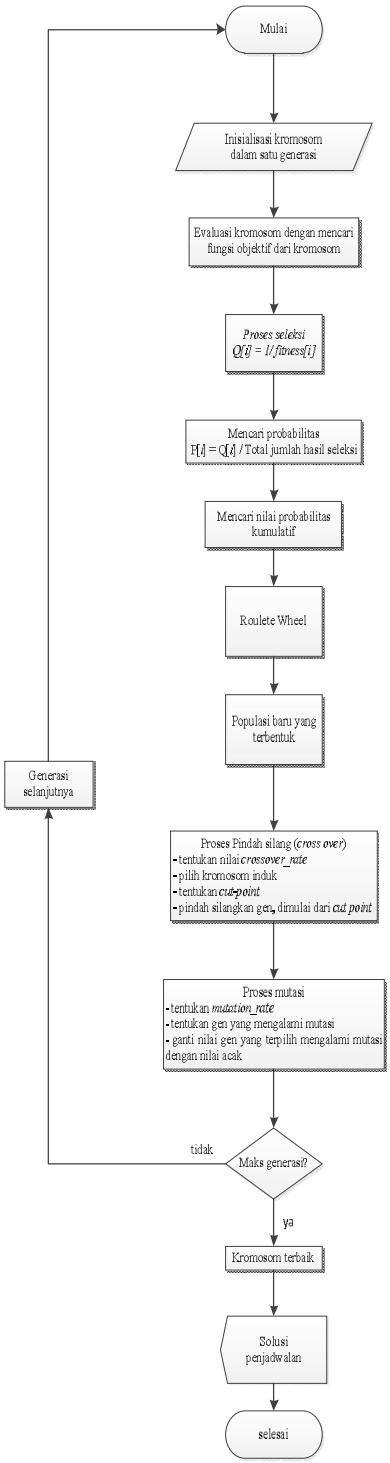
Gambar 3.3. Mekanisme Komunikasi Data Aplikasi

Proses mekanisme komunikasi data dalam perancangan aplikasi dimulai dari aplikasi *client* yang akan mengirim *request* ke aplikasi *server*, kemudian aplikasi *server* pun menerima *request* tersebut, lalu melakukan eksekusi *query* terhadap *database server* dan menerima informasi yang diminta dari *database server* sebagai hasil eksekusi *query* sebelumnya. Setelah itu aplikasi server akan menampilkan informasi dengan format yang sudah ditentukan sebagai *response* yang akan diterima aplikasi *client*. Sehingga aplikasi *client* dapat menampilkan informasi yang didapat dari *database server*.

3.1.6. Analisis Algoritma

3.1.6.1. Analisis Metode Algoritma Genetika dalam penyelesaian kasus *Travelling Salesman Problem* (TSP)

Algoritma Genetika adalah algoritma yang memanfaatkan proses seleksi alamiah yang dikenal dengan proses evolusi. Dalam proses evolusi, individu secara terus-menerus mengalami perubahan gen untuk menyesuaikan dengan lingkungan hidupnya. Dalam penerapan metode dari penelitian ini yang dibutuhkan untuk membuat penjadwalan perbaikan jaringan listrik di Rayon Badung Timur adalah dengan cara pendekatan penjadwalan menggunakan TSP (*Travelling Salesman Problem*) untuk mengoptimasikan penjadwalan pelayanan teknik kelokasi perbaikan yang berada di ruang lingkup PLN Rayon Bandung Timur agar semua keluhan dari pelanggan dapat ditangani dengan baik dan optimal dalam penjadwalannya. Pada gambar 3.4 merupakan diagram alir proses penyelesaian kasus TSP menggunakan algoritma genetika:



Gambar 3.4 *FlowChart* umum Algoritma Genetika dalam TSP

Contoh Kasus:

Ketika mendapatkan laporan keluhan dan gangguan pelanggan, pegawai pelayanan teknik menemukan 6 gardu yang harus dilakukan perbaikan karena mengkhawatirkan akan mengganggu kehandalan pasokan listrik. 6 gardu tersebut diwakili oleh gardu A, B, C, D, E dan F yang merupakan lokasi yang akan dilakukan perbaikan oleh pegawai pelayanan teknik. Gardu tersebut terdapat beberapa pekerjaan perbaikan yang harus dilakukan estimasi pengerjaan berdasarkan kerusakan yang biasa ditangani oleh regu YANTEK (pelayanan teknik) di PT. PLN Rayon Bandung Timur, adapun rincian perbaikan di setiap gardunya berdasarkan tabel 3.1 data jenis gangguan dapat dilihat pada tabel 3.2 contoh kasus dibawah ini adalah sebagai berikut:

Tabel 3.2 Contoh kasus

No	Lokasi Gardu	Jenis Gangguan	Waktu Perbaikan (menit)	Total Waktu (menit)
1	Gardu A	- Pelebur Putus (NHF) - Penyulang - Terminal OK terbakar	- 120 - 20 - 15	155
2	Gardu B	- Trafo - Fuse Base - Token Eror	- 30 - 10 - 20	60
3	Gardu C	- Instalasi - Kabel Naik - Trafo - Fuse Base	- 45 - 25 - 30 - 10	110
4	Gardu D	- Loskontak Pembatas - Fuse Base	- 15 - 10	30
5	Gardu E	- SUTR Putus/Terbakar - Kabel Naik	- 50 - 25	75
6	Gardu F	- JTR Melorot - Instalasi	- 45 - 45	90
				520 menit

Untuk melakukan perbaikan tersebut, maka pegawai YANTEK (pelayanan teknik) menjadwalkan gardu-gardu tersebut untuk segera dilakukan perbaikan jaringan listrik agar tidak mengalami kerusakan yang dapat mengganggu pasokan listrik.

Diketahui jarak YANTEK (pelayan teknik) ke tiap gardu yang ada di PT. PLN Rayon Bandung Timur bisa dilihat dari tabel 3.3 dan jarak antara lokasi

gardu yang diwakili oleh gardu A, B, C, D, E dan F dapat dilihat pada tabel 3.4 Jarak Tempuh Antar Gardu, dimana data yang ada sesuai dengan data hasil di lapangan sebagai pengujian di analisis algoritma genetika adalah sebagai berikut:

Tabel 3.3 Jarak YANTEK Ke Tiap Gardu (Dalam Satuan KM)

GARDU	A	B	C	D	E	F
YANTEK	2.5	1.5	4.1	8.6	5.2	3

Tabel 3.4 Jarak Tempuh Antar Gardu (Dalam Satuan KM)

Gardu	A	B	C	D	E	F
A	-	3.6	6.4	8.8	7.5	5.3
B	3.6	-	2.5	9.7	3.4	1.4
C	6.4	2.5	-	11.8	3.3	2.1
D	8.8	9.7	11.8	-	13.4	10.9
E	7.5	3.4	3.3	13.4	-	3.1
F	5.3	1.4	2.1	10.9	3.1	-

Dari tabel 3.3 dapat dilihat jarak dari posisi unit YANTEK ke lokasi gardu yang akan di perbaiki, jadi dapat disimpulkan bahwa jarak terdekat adalah gardu B, maka gardu B akan dijadikan inisialisasi gardu awal pengerjaan perbaikan jaringan listrik karena jaraknya terdekat dari posisi unit YANTEK dibandingkan dengan gardu lain. Maka penyelesaian dari contoh kasus diatas, yaitu optimasi penjadwalan penugasan YANTEK di PT PLN (persero) Rayon Bandung Timur adalah dengan menggunakan metode algoritma genetika pada penyelesaian TSP (*Travelling Salesman Problem*) dengan 2 unit YANTEK, yaitu unit 51 dan unit 52. Penjelasan mengenai langkah-langkah penyelesaian kasus di atas menggunakan metode algoritma genetika adalah sebagai berikut:

1. Pembentukan kromosom

Dalam kasus ini, pengkodean kromosom yang digunakan adalah pengkodean permutasi dimana pada pengkodean ini setiap kromosom merupakan barisan angka yang merepresentasikan angka pada urutan. Pengkodean permutasi hanya berguna pada masalah pengurutan. Istilah kromosom dalam algoritma genetika pada kasus ini dapat diartikan sebagai gardu atau lokasi perbaikan. Gardu A, B, C, D E,dan F merupakan kromosom dari gangguan dan perbaikan jaringan listrik yang akan dilakukan.

2. Inisialisasi Populasi dalam 1 generasi

Istilah populasi dalam algoritma genetik dapat diartikan sebagai solusi dimana solusi tersebut yang akan menyelesaikan masalah TSP ini. Proses inisialisasi dilakukan dengan cara menentukan jumlah populasinya dan memberikan nilai awal gen-gen dengan nilai acak sesuai batasan yang telah ditentukan. Selanjutnya menentukan dengan menggunakan 6 buah populasi dalam 1 generasi yang dimana kromosom didalamnya dipilih secara acak dan dikelompokkan menjadi 2 bagian sesuai dengan penugasan yang dilakukan pada 2 tim YANTEK.

Kromosom [1] = [B D A] [C F E]
Kromosom [2] = [D B C] [A E F]
Kromosom [3] = [E D B] [C F A]
Kromosom [4] = [C A B] [F E D]
Kromosom [5] = [A D E] [F B C]
Kromosom [6] = [A B C] [D E F]

3. Evaluasi kromosom

Permasalahan yang ingin diselesaikan adalah bagaimana mencari rute terpendek dan waktu tersingkat dalam melaksanakan penugasan keluhan dan gangguan jaringan listrik, maka hitung nilai Fungsi Objektif dari setiap kromosom yang telah dibangkitkan dari langkah 1 diatas dengan menghitung bobot dari setiap lintasan yang mempunyai waktu. Proses perhitungan nilai Fungsi Objektif dari kromosom tersebut adalah sebagai berikut:

Rumus mencari Fungsi Objektif :

Fungsi Objektif (ABCDEF) = JT(AB) + JT(BC) + JT(CD) + JT(DE) + JT(EF)

Keterangan:

JT = Jarak Tempuh

Perhitungan Fungsi Objektif:

I. Fungsi Objektif Kromosom [1] [B D A] [C F E]:

Fungsi Objektif [B D A] [C F E]

= (JT(BD) + JT(DA)) + (JT(CF) + JT(FE))

(9.7 + 8.8) + (2.1 + 3.1) = 23.7

II. Fungsi Objektif Kromosom [2] [D B C] [A E F]:

Fungsi Objektif [D B C] [A E F]

= (JT(DB) + JT(BC)) + (JT(AE) + JT(EF))

(9.7 + 2.5) + (5.3 + 3.1) = 20.6

III. Fungsi Objektif Kromosom [3] [E D B] [C F A]:

Fungsi Objektif [E D B] [C F A]

= (JT(ED) + JT(DB)) + (JT(CF) + JT(FA))

(13.4 + 9.7) + (2.1 + 5.3) = 30.5

IV. Fungsi Objektif Kromosom [4] [C A B] [F E D]:

Fungsi Objektif [C A B] [F E D]:

= (JT(CA) + JT(AB)) + (JT(FE) + JT(ED))

(6.4 + 3.6) + (3.1 + 13.4) = 26.5

V. Fungsi Objektif Kromosom [5] [A D E] [F B C]:

Fungsi Objektif [A D E] [F B C]

= (JT(AD) + JT(DE)) + (JT(FB) + JT(BC))

(8.8 + 13.4) + (1.4 + 2.5) = 26.1

VI. Fungsi Objektif Kromosom [6] [A B C] [D E F]:

Fungsi Objektif [A B C] [D E F]

= (JT(AB) + JT(BC)) + (JT(DE) + JT(FE))

(3.6 + 2.5) + (13.4 + 3.1) = 22.6

Pada kasus ini yang diinginkan adalah mencari rute terpendek dan waktu optimal, maka kromosom yang lebih kecil akan mempunyai probabilitas untuk terpilih kembali lebih besar. Untuk itu dapat digunakan rumus seleksi:

Fitness [i] = 1/ nilai Fungsi Objektif[i]

Fitness [1] = 1/ 23.7 = 0.0422

Fitness [2] = 1/ 20.6 = 0.0485

Fitness [3] = 1/ 30.5 = 0.0328

Fitness [4] = 1/ 26.5 = 0.0377

Fitness [5] = 1/ 26.1 = 0.0383

Fitness [6] = 1/ 22.6 = 0.0442

Total *Fitness* = 0.0422 + 0.0485 + 0.0328 + 0.0377 + 0.0383 + 0.0442 = **0.2437**

Setelah didapat nilai *fitness* selanjutnya adalah memilih hasil maksimum dengan nilai *fitness* tertinggi dengan rumus :

$$max_i\ Fitness[1] = \frac{1}{fungsi\ Objektif[i]}$$

= max {0.0422, 0.0485, 0.0328, 0.0377, 0.0383, 0.0442} = **0.0485**

Jadi, didapatkan hasil optimal dari nilai *fitness* dan nilai objektif terendah berasal dari **kromosom 2 = [D B C] [A E F]** pada literasi pertama.

4. Seleksi Kromosom

Setelah mendapatkan nilai *fitness* dan total *fitness*, selanjutnya adalah mencari nilai probabilitas dari masing - masing kromosom. Dengan rumus mencari probabilitas adalah :

P[i] = Fitness [i] / Total Fitness

P[1] = 0.0422 / 0.2437 = 0.173

P[2] = 0.0485 / 0.2437 = 0.199

P[3] = 0.0328 / 0.2437 = 0.135

P[4] = 0.0377 / 0.2437 = 0.155

P[5] = 0.0383 / 0.2437 = 0.157

P[6] = 0.0442 / 0.2437 = 0.181

5. Probabilitas Kumulatif

Dari probabilitas diatas dapat dilihat yang mempunyai *fitness* paling kecil maka kromosom tersebut mempunyai probabilitas untuk terpilih pada generasi selanjutnya lebih besar dari kromosom lainnya. Untuk proses seleksi digunakan *roulete wheel*, untuk itu harus mencari terlebih dahulu nilai probabilitas kumulatifnya:

C[1] = 0.173

C[2] = 0.173 + 0.199 = 0.372

C[3] = 0.372 + 0.135 = 0.507

C[4] = 0.507 + 0.155 = 0.662

C[5] = 0.662 + 0.157 = 0.819

C[6] = 0.819 + 0.181 = 1

6. Roulete Wheel

Setelah didapat probabilitas kumulatifnya, maka selanjutnya adalah proses seleksi menggunakan *roulete wheel*. Tahapan prosesnya adalah dengan membangkitkan bilangan acak R (0..1) terlebih dahulu. Misal nilai R acak yang diperoleh adalah sebagai berikut:

R[1] = 0.314

R[2] = 0.111

R[3] = 0.342

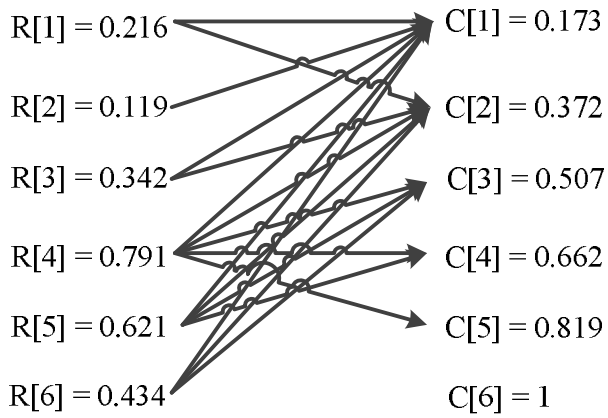
R[4] = 0.743

R[5] = 0.521

R[6] = 0.411

Setelah didapatkan bilangan acak R, tahapan selanjutnya adalah mencari kromosom induk dengan cara membandingkan antara bilangan acak R yang telah dibangkitkan dengan probabilitas kumulatif yang telah didapat dengan syarat R[i] < C[i]. Bandingkan nilai R[1] terhadap nilai C[1], Jika R[1] < C[1] maka pilih kromosom ke-1 tersebut sebagai induk pertama, apabila R[1] > C[1] maka bandingkan kembali dengan C[2], C[3], C[4], C[5], C[6] sampai terpenuhi syarat *roulete wheel*. Apabila sudah terpenuhi, maka ambil indeks i dari C sebagai

indeks kromosom yang terpilih untuk menggantikan kromosom sebelum proses seleksi *roulete wheel*. Ulangi proses ini untuk semua R[i].



Hasil dari proses *roulete wheel*:

Kromosom [1] = [2] [D B C] [A E F]

Kromosom [2] = [1] [B D A] [C F E]

Kromosom [3] = [2] [D B C] [A E F]

Kromosom [4] = [5] [A D E] [F B C]

Kromosom [5] = [4] [C A B] [F E D]

Kromosom [6] = [3] [E D B] [C F A]

7. Pindah Silang (*Cross Over*)

Setelah proses seleksi maka proses selanjutnya adalah proses *crossover*. Metode yang digunakan salah satunya adalah *one-cut point*, yaitu memilih secara acak satu posisi dalam kromosom induk kemudian saling menukar gen. Kromosom yang dijadikan induk dipilih secara acak dan jumlah kromosom dipengaruhi oleh *crossover probability* (pc). Misal tentukan *crossover probability* adalah sebesar 10%, maka dalam satu generasi ada 10% kromosom dari satu generasi mengalami proses *crossover*. Prosesnya adalah sebagai berikut:

Langkah pertama adalah dengan membangkitkan bilangan acak R (0..1) sebanyak jumlah populasi:

R[1] = 0.092

R[2] = 0.021

R[3] = 0.854

R[4] = 0.589

R[5] = 0.441

R[6] = 0.071

Kromosom ke-i yang dipilih sebagai induk jika R[i] < pc. Berdasarkan contoh diatas R[1] < 0.1, R[2] < 0.1 dan R[6] < 0.1 maka yang akan dijadikan induk adalah kromosom[1] sebagai induk pertama, kromosom[2] sebagai induk kedua, dan kromosom[6] sebagai induk ketiga dari proses pemilihan induk di atas. Setelah pemilihan induk, proses selanjutnya adalah menentukan posisi *crossover*. Hal tersebut dilakukan dengan membangkitkan bilangan acak dari 1 sampai dengan panjang kromosom-1. Dalam kasus ini, bilangan acaknya adalah antara 1-5. Bilangan acak untuk 3 kromosom induk yang akan di *crossover*:

A[1] = 1

A[2] = 3

A[3] = 2

Misal diperoleh bilangan acaknya adalah 2, maka 2 gen awal pada kromosom induk pertama diambil dan dipertahankan kemudian ditukar dengan gen pada kromosom induk kedua yang belum ada pada induk pertama dengan tetap memperhatikan urutannya.

Proses *crossover*:

Kromosom[1] = Kromosom[1] x Kromosom[2]

= [D B C] [A E F] x [**B** D **A**] [**C F E**]

= [**D** B A] [C F E]

Kromosom[2] = Kromosom[2] x Kromosom[6]

= [B D A] [C F E] x [**E** D B] [**C F** A]

= [**B D A**] [E C F]

Kromosom[6] = Kromosom[6] x Kromosom[1]

= [[E D B] [C F A]x [D **B C**] [**A** E **F**]

= [**E D** B] [C A F]

Populasi setelah di *crossover*:

Kromosom [1] = [D B A] [C F E]

Kromosom [2] = [B D A] [E C F]

Kromosom [3] = [D B C] [A E F]

Kromosom [4] = [A D E] [F B C]

Kromosom [5] = [C A B] [F E D]

Kromosom [6] = [E D B] [C A F]

8. Mutasi

Dalam kasus ini skema mutasi yang digunakan adalah *swapping mutation*. Jumlah kromosom yang mengalami mutasi dalam satu populasi ditentukan oleh parameter probabilitas mutasi (pm). Proses mutasi dilakukan dengan cara menukar gen yang dipilih secara acak dengan gen sesudahnya. Jika gen tersebut berada di akhir kromosom, maka ditukar dengan gen yang pertama.

Pertama hitung terlebih dahulu panjang total gen yang ada pada satu populasi:
Panjang total gen = jumlah gen dalam 1 kromosom* jumlah kromosom

= 6*6 = 36

Untuk memilih posisi gen yang mengalami mutasi dilakukan dengan membangkitkan bilangan acak antara 1 - panjang total gen yaitu 1-36. Misal ditentukan pm = 10%. Maka jumlah gen yang akan dimutasi adalah

0.1 * 36 = 3.6 = 4 (dibulatkan ke atas)

4 buah posisi gen yang akan dimutasi, setelah diacak adalah posisi 5, 8, 17, dan 29. Proses Mutasi:

Kromosom [1] = [D B A] [C **F** E]

Kromosom [2] = [B **D** A] [E C F]

Kromosom [3] = [D B C] [A **E** F]

Kromosom [4] = [A D E] [F B C]

Kromosom [5] = [C A B] [F **E** D]

Kromosom [6] = [E D B] [C A F]

Hasil mutasi:

Kromosom [1] = [D B A] [C **E F**]

Kromosom [2] = [B **A D**] [E C F]

Kromosom [3] = [D B C] [A **F E**]

Kromosom [4] = [A D E] [F B C]

Kromosom [5] = [C A B] [F **D E**]

Kromosom [6] = [E D B] [C A F]

Proses algoritma genetika untuk 1 generasi telah selesai. Maka nilai fungsi objektif tiap kromosom setelah 1 generasi adalah:

I. Fungsi Objektif Kromosom [1] = [D B A] [C **E F**]
=(JT(DB) + JT(BA)) + (JT(CE) + JT(EF))
= (9.7 + 3.6) + (3.3 + 3.1) = 19.7

II. Fungsi Objektif Kromosom [2] = [B **A D**] [E C F]
=(JT(BA) + JT(AD)) + (JT(EC) + JT(CF))
= (3.6 + 8.8) + (3.3 + 2.1) = 17.8

III. Fungsi Objektif Kromosom [3] = [D B C] [A **F E**]
=(JT(DB) + JT(BC)) + (JT(AF) + JT(FE))
= (9.7 + 2.5) + (5.3 + 3.1) = 20.6

IV. Fungsi Objektif Kromosom [4] = [A D E] [F B C]
=(JT(AD) + JT(DE)) + (JT(FB) + JT(BC))
= (8.8 + 13.4) + (1.4 + 2.5) = 26.1

V. Fungsi Objektif Kromosom [5] = [C A B] [F **D E**]
=(JT(CA) + JT(AB)) + (JT(FD) + JT(DE))
= (6.4 + 3.6) + (10.9 + 13.4) = 34.3

VI. Fungsi Objektif Kromosom [6] = [E D B] [C A F]
=(JT(ED) + JT(DB)) + (JT(CA) + JT(AF))
= (13.4 + 9.7) + (6.4 + 5.3) = 34.8

Berdasarkan Nilai dari fungsi objektif diatas adalah

Fitness [i] = 1/ nilai Fungsi Objektif[i]

Fitness [1] = 1/ 19.7 = 0.0508

Fitness [2] = 1/ 17.8 = 0.0562

Fitness [3] = 1/ 20.6 = 0.0485

Fitness [4] = 1/ 26.1 = 0.0383

Fitness [5] = 1/ 34.3 = 0.0292

Fitness [6] = 1/ 34.8 = 0.0287

Setelah didapat nilai *fitness* selanjutnya adalah memilih hasil maksimum dengan nilai *fitness* tertinggi dengan rumus :

$$\max Fitness\ [i] = \frac{1}{fungsi\ Objektif[i]}$$

= max {0.0508, 0.0562, 0.0485, 0.0383, 0.0292, 0.0287}

= 0.0562

Hasil dari tiap generasi akan dibandingkan dengan generasi sebelumnya, apabila kromosom yang mempunyai nilai objektif dan nilai *fitness*nya bernilai baik maka digenerasi selanjutnya kromosom tersebut akan digunakan. Literasi diulang kembali sebanyak maksimum generasi, maka akan dihasilkan literasi waktu tersingkat yang optimal untuk penjadwalan keluhan dan gangguan pelanggan. Jadi, hasil generasi maksimum yang optimal dari perhitungan di atas adalah: **Kromosom [2] = [B A D] [E C F]** dengan nilai fungsi objektif kromosom tersebut adalah 17.8.

Maka dapat di simpulkan total jarak tempuh adalah = ∑ Jarak Tempuh Terpendek = 17.8 KM.

Jarak Tempuh Terpendek Tiap Unit YANTEK berdasarkan tabel 3.4 adalah :

JT Unit 51: JT**[B A D]** = JT(BA) + JT(AD) = (3.6 + 8.8) =12.4 KM.

JT Unit 52 : JT**[E C F]** = JT(EC) + JT(CF) = (3.3 + 2.1) = 5.4 KM.

9. Perhitungan Perkiraan Waktu Tempuh Minimal

Untuk mendapatkan perkiraan waktu tempuh diperlukan asumsi kecepatan rata-rata dalam kota dengan menggunakan kendaraan transportasi yang dipakai oleh tim untuk sampai ke setiap lokasi gardu, yaitu sebesar 14.3 KM / Jam, maka jika ∑ Jarak Tempuh = 17.8 KM perkiraan waktu tempuhnya adalah sebagai berikut :

$WT_{minimal} = JT_{terpendek}$ / KR

$WT_{minimal}$ Total= 17.8 / 14.3 = 1.245 Jam ≈ 74.7 Menit

$WT_{minimal}$ Unit 51 = JT Unit 51/KR

= 12.4 / 14.3 = 0.867 Jam ≈ 52.02 Menit

$WT_{minimal}$ Unit 52 = JT Unit 52/KR

= 5.4 / 14.3 = 0.377 Jam ≈ 22.62 Menit

Keterangan :

WT = Waktu Tempuh

JT = Jarak Tempuh

KR = Kecepatan Rata-Rata

10. Perhitungan Perkiraan Waktu Operasional Minimal

Untuk mendapatkan perkiraan waktu operasional adalah dengan menjumlahkan waktu tempuh dan waktu estimasi perbaikan, yaitu :

$WO_{minimal} = WT_{minimal}$ + Total WEP

Keterangan :

WO = Waktu Operasional

WT = Waktu Tempuh

WEP = Waktu Estimasi Perbaikan

Untuk mengetahui estimasi perbaikan layanan keluhan dan gangguan pelanggan di PT. PLN Rayon Bandung Timur dapat dilihat pada tabel 3.2 contoh kasus.

Total WEP : [B A D E C F] = (60 + 155 + 30 + 75 + 110 + 90) = 520 menit

Total WEP Unit 51: [B A D] = (60 + 155 + 30) = 245 menit

Total WEP Unit 52: [E C F] = (75 +110 + 90) = 275 menit

$WO_{minimal}$ Total = $WT_{minimal}$ + Total WEP = 74.7 Menit + 520 Menit

= 594.7 Menit ≈ 9.9 Jam

$WO_{minimal}$ Unit 51 = $WT_{minimal}$ Unit 51 + Total WEP Unit 51

= 52.02 Menit + 245 Menit = 297.02 Menit

$WO_{minimal}$ Unit 51 = $WT_{minimal}$ Unit 52 + Total WEP Unit 52

= 22.62 Menit + 275 Menit = 297.62 Menit

11. Perhitungan Perkiraan Optimasi Biaya

Untuk mendapatkan perkiraan optimasi biaya diperlukan asumsi penggunaan bahan bakar 1 liter untuk 11,2 KM (jenis kendaraan Toyota avanza). Sehingga didapat total penggunaan bahan bakar untuk jarak tempuh adalah sebagai berikut :

$TPB = JT_{terpendek} / PB_{1\ liter}$

TPB = 17.8 KM / 11,2 KM = 1.589 liter

Unit 51 : TPB = 12.4 KM / 11,2 KM = 1.107 liter

Unit 52 : TPB = 5.4 KM / 11,2 KM = 0.482 liter

Keterangan :

TPB = Total Penggunaan Bahan Bakar

$JT_{terpendek}$ = Jarak Tempuh

$PB_{1\ liter}$ = Penggunaan Bahan Bakar 1 Liter

Kemudian dihitung perkiraan biaya, yaitu dengan mengalikan total penggunaan bahan bakar dengan harga pasar bahan bakar jenis premium yaitu Rp. 4.500,- / 1 liter.

$PBY_{minmal} = TPB \times HB_{1\ liter}$

$PBY_{minimal}$ = 1.589 liter x Rp. 6.500,- = Rp. 10.328,-

Unit 51 : $PBY_{minimal}$ = 1.107 liter x Rp. 6.500,- = Rp. 7.195,-

Unit 52 : $PBY_{minimal}$ = 0.482 liter x Rp. 6.500,- = Rp. 3.333,-

Keterangan :

PBY = Perhitungan Biaya Minimal

TPB = Total Penggunaan Bahan Bakar

$HB_{1\ liter}$ = Harga Bahan Bakar

Kasus yang terjadi dilapangan pihak YANTEK Rayon Bandung Timur belum dapat mengalokasikan penggunaan bahan bakar sesuai kebutuhan, hanya memperkirakan penggunaan bahan bakar yang digunakan. Misal dijatah perhari adalah 20 Liter BBM untuk tiap Unit maka dari kondisi diatas penggunaan Bahan bakar tidak optimal dalam jangka panjang akan terjadi pemborosan, dengan penerapan algoritma genetika untuk mendapatkan pemecahan masalah dengan solusi yang optimal pada persoalan penjadwalan dimana solusi yang ingin dicapai

adalah berdasarkan waktu dan jarak tempuh rute perjalanan kelokasi perbaikan yang minimal sehingga penjadwalan keluhan dan gangguan pelanggan menjadi lebih optimal.

3.1.7. Analisis Kebutuhan Non Fungsional

Analisa kebutuhan non fungsional menggambarkan keadaan sistem yang ada di PT PLN (Persero) Rayon Bandung Timur, Analisis non fungsional adalah sebuah tahap di mana seorang pembangun perangkat lunak menganalisis sumber daya yang akan menggunakan perangkat lunak yang di bangun. Sehingga dapat di tentukan kompatibilitas aplikasi yang di bangun terhadap sumber yang ada. Diantaranya keadaan perangkat keras, perangkat lunak, serta *user* sebagai bahan analisis kekurangan dan kebutuhan yang harus dipenuhi dalam perancangan sistem yang akan diterapkan.

3.1.7.1. Analisis Kebutuhan Perangkat Keras

Untuk menjalankan suatu aplikasi maka diperlukan perangkat keras yang dapat mendukung proses kerja dari sistem itu sendiri. Spesifikasi perangkat keras yang digunakan dalam pembuatan aplikasi ini adalah :

Processor : Processor 3.0 GHz
RAM : 2 GB Dedicated
VGA : 512 MB Dedicated
HDD : 120 GB
Monitor : LCD 17”

3.1.7.2. Analisis Kebutuhan Perangkat Lunak

Secara keseluruhan sistem operasi yang digunakan pada komputer adalah Windows Xp dan perangkat kerja yang sering digunakan adalah *Microsoft Office Word* dan *Excel* serta menjalankan aplikasi pengaduan dan keluhan terpadu (APKT) yang sudah tersistem di PT. PLN (Persero) Rayon Bandung Timur.

Sistem yang sudah ada tidak akan terpengaruh oleh perangkat lunak yang akan diaplikasikan. Aplikasi yang akan dibangun menggunakan *Macromedia*

Dreamweaver CS3 sebagai *editor,* PHP dan MYSQL sebagai *database*nya serta XAMPP sebagai *server* local.

Alasan digunakannya *Macromedia Dreamweaver CS3* sebagai *editor* PHP sebagai *tools* perancangan program *website* adalah dalam pengembangan sistem untuk kedepannya lebih mudah dan menghasilkan program *website* yang mudah dimengerti dan digunakan oleh *user* yang familiar dengan sistem operasi dan program-program *Windows*. Kemudian alasan digunakannya MYSQL sebagai *database* yaitu mudah digunakan dan memiliki dukungan *user* yang luas dan memiliki tingkat keamanan yang baik. Dalam melakukan pengujian *website* sebelum di hosting ke internet maka terlebih dahulu menggunakan XAMPP sebagai server local, hal ini memudahkan dalam pengembangan website dengan cepat dan mudah, karna XAMPP merupakan tool yang menyediakan paket perangkat lunak ke dalam satu buah paket. Dalam paketnya sudah terdapat Apache (*web server*), MySQL (*database*), PHP (*server side scripting*), Perl, FTP *Server*, phpMyAdmin dan berbagai pustaka bantu lainnya.

Website ini dibuat guna untuk mendukung dan *monitoring* aktifitas laporan keluhan dan gangguan pelanggan yang ada di PT PLN (Persero) Rayon Bandung Timur. Untuk itu dibutuhkan sebuah *website* yang akan dikelola oleh operator YANTEK sebagai admin dan akan terintegrasi langsung ke *Platform* Android melalu HTTP *Request* sehingga saling berkorelasi antar admin dan regu YANTEK.

Dari analisis perangkat lunak yang dilakukan, sistem operasi yang dijalankan di PT PLN (Persero) Rayon Bandung Timur dapat mendukung perangkat lunak yang akan diaplikasikan.

3.1.7.3. Analisis *User*

Analisis *user* yang ada meliputi admin dan regu/unit yantek (pelayanan teknik) adalah sebagai berikut:

1. **Admin**

   Admin disini bertindak sebagai admin *website* yang berperan penting dalam input dan mengelolah data keluhan dan gangguan yang diterima

dari pelanggan PT PLN (Persero) Rayon Bandung Timur serta harus memiliki keterampilan dalam komputerisasi.

2. **Unit / regu YANTEK (pelayanan teknik) unit 51 dan unit 52.**
   Kepada tiap unit dari pelayanan teknik baik unit 51 atau unit 52 diharapkan menguasai dan dapat mengoperasikan sistem yang ada di aplikasi *mobile*, hal ini sangat berpengaruh ketika sistem sedang berjalan dan tiap unit mendapatkan penugasan kerja yang sudah dijadwalkan pada aplikasi mobile yang sudah terintegrasi dengan *website* tersebut.

3.1.8. Analisis Kebutuhan Fungsional

Analisis kebutuhan fungsional menggambarkan proses kegiatan yang akan diterapkan dalam sebuah sistem dan menjelaskan kebutuhan yang diperlukan sistem agar sistem dapat berjalan dengan baik.

Analisis yang dilakukan dimodelkan dengan menggunakan UML (*Unified Modeling Language*). Tahap-tahap pemodelan dalam analisis tersebut antara lain identifikasi aktor, *use case diagram, skenario, activity diagram, sequence diagram, class diagram*.

3.1.8.1. Identifikasi aktor

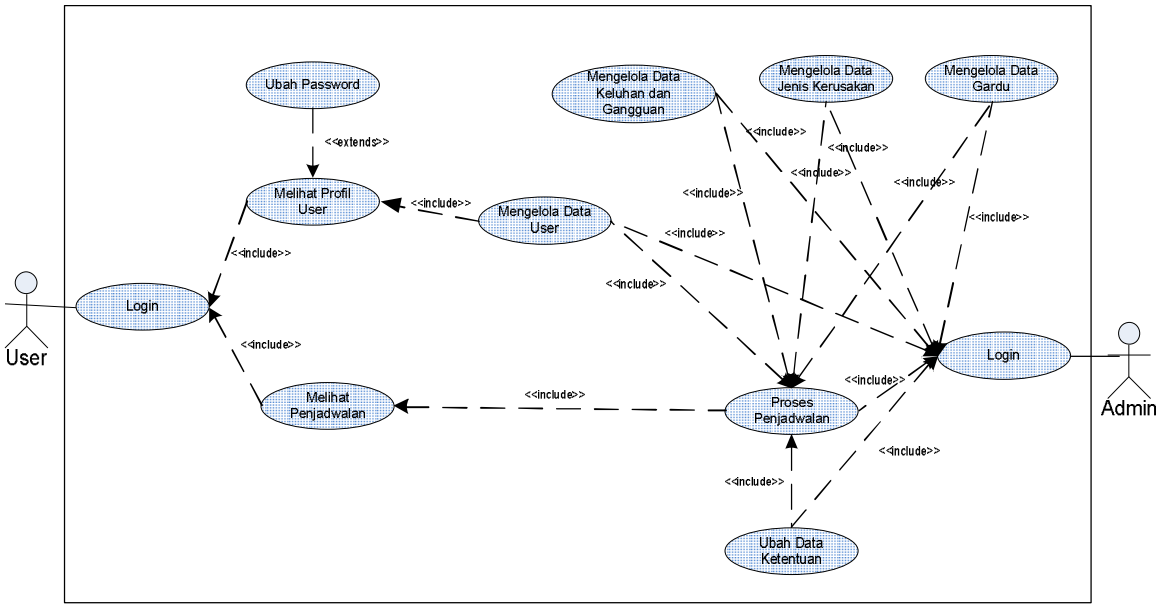
Identifikasi aktor dimaksudkan untuk mengetahui siapa saja aktor yang terlibat di dalam sistem aplikasi ini. Deskripsi dari tiap aktor yang terlibat dapat dilihat pada tabel 3.5.

Tabel 3.5 Identifikasi Aktor

No	Aktor	Deskripsi
1	*User*	*User* ini merupakan pengguna aplikasi pada aplikasi mobile yaitu Unit / regu YANTEK (pelayanan teknik) unit 51 dan unit 52 Dimana dalam menggunakan aplikasi *user* harus memiliki koneksi internet yang berguna dalam *parsing* data ke *database server* untuk melihat optimasi penjadwalan di Rayon Bandung Timur.
2	Admin	Pengelola data penjandwalan, dimana admin memiliki hak akses penuh untuk dapat menambah, mengubah dan menghapus data yang terdapat di *database server*.

3.1.8.2. *Use Case Diagram*

Use case Diagram merupakan konstruksi untuk mendeskripsikan hubungan-hubungan yang terjadi antar aktor dengan aktivitas yang terdapat pada sistem. Sasaran pemodelan *use case* diantaranya adalah mendefinisikan kebutuhan fungsional dan operasional sistem dengan mendefinisikan skenario penggunaan sistem yang akan dibangun. Dari hasil analisis aplikasi yang ada maka *use case diagram* pada aplikasi ini dapat dilihat pada gambar 3.5



Gambar 3.5 *Use Case Diagram* Aplikasi Penjadwalan YANTEK

3.1.8.2.1. **Definisi *Use Case***

Use case diagram merupakan konstruksi untuk mendeskripsikan hubungan-hubungan yang terjadi antar aktor dengan aktivitas yang terdapat pada sistem. Sasaran pemodelan *use case* diantaranya adalah mendefinisikan kebutuhan fungsional dan operasional sistem dengan mendefinisikan skenario penggunaan yang disepakati antara pemakai dan pengembang. Definisi dari *use case* aplikasi dapat dilihat pada tabel 3.6

Tabel 3.6 Definisi *Use Case* Admin

No	*Use Case*	Deskripsi
1	Profil *User*	Proses untuk melihat profil *user*, dimana terdapat konten info user dan ubah *password*.
2	Penjadwalan	Proses melihat penjadwalan pada aplikasi.
3	Ubah *Password*	Proses mengubah *password login* aplikasi.
4	Mengelola data user	Proses untuk melakukan tambah, edit, hapus dan blokir data user.
5	Mengelola data gardu	Proses untuk melakukan tambah, edit, dan hapus data gardu.
6	Mengelola data jenis kerusakan	Proses untuk melakukan tambah, edit, dan hapus data jenis kerusakan.
7	Mengelola data gangguan dan kerusakan	Proses untuk melakukan tambah, edit, dan hapus data gangguan dan kerusakan.
8	Proses Penjadwalan	Proses untuk melakukan perhitungan algoritma genetika dan hasil penjadwalan.
9	data ketentuan	Proses untuk melakukan modifikasi ketentuan/batasan.
10	Login	Prose untuk login masuk ke halaman aplikasi.

3.1.8.3. Skenario *Use Case*

Skenario *Use Case* mendeskripsikan urutan langkah-langkah dalam proses bisnis, baik yang dilakukan aktor terhadap sistem maupun yang dilakukan oleh sistem terhadap aktor.

3.1.8.3.1. *Use case* sekenario Profil *User*

Interaksi antara aktor *user* dengan *use case* profil *user* dijelaskan dalam tabel 3.7

Tabel 3.7 *Use Case* Skenario Profil *User*

Identifikasi		
Nomor	1	
Nama	Profil *User*	
Tujuan	*User* dapat melihat Profil *User*.	
Deskripsi	Menu ini berisi *form* yang menampilkan data-data *user* serta informasi yang terkait dengan profil *user*	
Aktor	*User*	
Skenario Utama		
Kondisi Awal	*User* berada di halaman *login*	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login*.
2. *User* mengisi *form login*		
3. *User* menekan tombol *login*.		4. Menampilkan halaman utama aplikasi.
5. *User* memilih menu Profil *user*.		6. Menampilkan halaman Profil *User*.
Kondisi Akhir	*User* telah melihat halaman profil *user*.	

3.1.8.3.2. *Use case* sekenario Penjadwalan

Interaksi antara aktor *user* dengan *use case* penjadwalan dijelaskan dalam tabel 3.8

Tabel 3.8 *Use Case* Skenario Penjadwalan

Identifikasi		
Nomor	2	
Nama	Penjadwalan	
Tujuan	*User* dapat melihat data penjadwalan.	
Deskripsi	Menu ini berisi *form* yang menampilkan data-data penjadwalan serta informasi penugasan kepada regu YANTEK.	
Aktor	*User*	
Skenario Utama		
Kondisi Awal	*User* berada di halaman *login*	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. *User* mengisi *form login*		
3. User menekan tombol *login*		4. Menampilkan halaman utama aplikasi.
5. User memilih menu Penjadwalan.		6. Aplikasi menampilkan halaman penjadwalan
Kondisi Akhir	*User* telah melihat data penjadwalan.	

3.1.8.3.3. *Use case* sekenario Ubah *Password*

Interaksi antara aktor *user* dengan *use case* ubah *password* dijelaskan dalam tabel 3.9

Tabel 3.9 *Use Case* Skenario Ubah *Password*

Identifikasi		
Nomor	3	
Nama	Ubah *Password*	
Tujuan	*User* dapat mengubah *password* ketika *login* ke aplikasi.	
Deskripsi	Menu ini mengisi *form* untuk mengubah *password login.*	
Aktor	*User*	
Skenario Utama		
Kondisi Awal	*User* berada di halaman *login*	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. *User* mengisi *form login.*		
3. User menekan tombol *login.*		4. Menampilkan halaman utama aplikasi.
5. User memilih menu profil *user*		6. Menampilkan halaman profil *user.*
7. User mengisi *field* ubah *password*		
8. menekan tombol simpan.		9. Menampilkan pesan *password* berhasil diubah.
		10. *Password* behasil diubah
Kondisi Akhir	*User* telah merubah *password login.*	

3.1.8.3.4. *Use case* skenario Tambah Data Pengguna

Interaksi antara aktor admin dengan *use case* tambah data pengguna dijelaskan dalam tabel 3.10

Tabel 3.10 *Use Case* Skenario Tambah Data Pengguna

Identifikasi		
Nomor	4	
Nama	Tambah data pengguna.	
Tujuan	Admin dapat menambah data pengguna.	
Deskripsi	Pada halaman ini, admin mengisi form untuk menambah data pengguna, seperti nama *user, password,* ulang *password,* dan level pengguna.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data *user*.		6. Menampilkan halaman data-data pengguna yang tersimpan.
7. Admin memilih tambah pengguna.		8. Menampilkan halaman tambah data pengguna.
9. Admin mengisi *form* tambah data pengguna.		
10. menekan tombol simpan.		11. Memeriksa apakah data sudah sesuai, jika : - Ya, menyimpan data pengguna ke *database* server. - Tidak, kembali ke halaman tambah data pengguna.
Kondisi Akhir	Tambah data pengguna berhasil dan data tersimpan ke *database server*.	

3.1.8.3.5. *Use case* skenario *Edit* Data Pengguna

Interaksi antara aktor admin dengan *use case edit* data pengguna dijelaskan dalam tabel 3.11

Tabel 3.11 *Use Case* Skenario *Edit* Data Pengguna

Identifikasi		
Nomor	5	
Nama	*Edit* data pengguna.	
Tujuan	Admin dapat meng-*edit* data pengguna yang sudah tersimpan sebelumnya di *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu data pengguna, dan mengisi *form* peng-*edit*-an data pengguna.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*

Aksi Aktor	Reaksi Sistem
5. Admin memilih menu data *user*.	6. Menampilkan daftar data-data pengguna yang tersimpan.
7. Admin memilih data pengguna	
8. menekan tombol *edit*.	9. Menampilkan *form edit* data pengguna.
10. Mengisi form *edit* data pengguna.	
11. menekan tombol simpan.	12. Memeriksa apakah data sudah sesuai, jika : - Ya, meng-*upadate* data pengguna ke *database* server. - Tidak, kembali ke form *edit* data pengguna.
Kondisi Akhir	Peng-*edit*-an data pengguna berhasil dan data pengguna yang telah tersimpan sebelumnya telah berubah.

3.1.8.3.6. *Use case* skenario Hapus Data Pengguna

Interaksi antara aktor admin dengan *use case* hapus data pengguna dijelaskan dalam tabel 3.12

Tabel 3.12 *Use Case* Skenario Hapus Data Pengguna

Identifikasi	
Nomor	6
Nama	Hapus data pengguna.
Tujuan	Admin dapat menghapus data pengguna yang sudah tidak diperlukan lagi dari *database server*.
Deskripsi	Pada halaman ini, admin akan memilih salah satu dari data pengguna yang tersimpan, dan memilih hapus data tersebut dari *database server*.
Aktor	Admin.
Skenario Utama	
Kondisi Awal	Admin berada di halaman login admin.

Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman *login*.
2. Admin mengisi *form login*.	
3. Admin menekan tombol *login*.	4. Menampilkan halaman utama *website*.
5. Admin memilih menu data *user*.	6. Menampilkan daftar data pengguna
7. Admin memilih data pengguna.	
8. Admin menekan tombol hapus.	9. Menampilkan jendela konfirmasi hapus data pengguna.
10. Menjawab jendela konfirmasi hapus data pengguna.	11. Memeriksa jawaban konfirmasi, jika : - Ya, menghapus data pengguna. - Tidak, kembali ke halaman daftar data pengguna.
Kondisi Akhir	Data pengguna yang dipilih telah terhapus dari *database server*.

3.1.8.3.7. *Use case* skenario Blokir Data Pengguna

Interaksi antara aktor admin dengan *use case* blokir data pengguna dijelaskan dalam tabel 3.13

Tabel 3.13 *Use Case* Skenario Blokir Data Pengguna

Identifikasi		
Nomor	7	
Nama	Blokir data pengguna.	
Tujuan	Admin dapat memblokir data pengguna yang aktif dari *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu dari data pengguna yang tersimpan, dan memblokir data tersebut dari *database server*.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data *user*.		6. Menampilkan daftar data pengguna.
7. Admin memilih data pengguna		
8. Admin menekan tombol blokir.		9. Menampilkan jendela konfirmasi blokir data pengguna
10. Menjawab jendela konfirmasi blokir data pengguna		11. Memeriksa jawaban konfirmasi, jika : - Ya, blokir data pengguna. - Tidak, kembali ke halaman daftar data pengguna.
Kondisi Akhir	Data pengguna yang dipilih telah diblokir dari *database server*.	

3.1.8.3.8. *Use case* skenario Tambah Data Gardu

Interaksi antara aktor admin dengan *use case* tambah data gardu dijelaskan dalam tabel 3.14

Tabel 3.14 *Use Case* Skenario Tambah Data Gardu

Identifikasi		
Nomor	8	
Nama	Tambah data gardu.	
Tujuan	Admin dapat menambah data gardu.	
Deskripsi	Pada halaman ini, admin mengisi *form* untuk menambah data gardu, seperti nama gardu, lokasi, *longitude*, dan *latitude*.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data gardu.		6. Menampilkan halaman data gardu.
7. Admin memilih tambah gardu		8. Menampilkan halaman tambah data gardu.
9. Admin mengisi *form* tambah data gardu.		

Aksi Aktor	Reaksi Sistem
10. Admin menekan tombol simpan.	11. Memeriksa apakah data yang diisikan telah sesuai, jika : - Ya, menyimpan data gardu ke *database* server. - Tidak, kembali ke halaman tambah data gardu.
Kondisi Akhir	Tambah data gardu berhasil dan data tersimpan ke *database server*.

3.1.8.3.9. *Use case* skenario *Edit* Data Gardu

Interaksi antara aktor admin dengan *use case edit* data gardu dijelaskan dalam tabel 3.15

Tabel 3.15 *Use Case* Skenario *Edit* Data Gardu

Identifikasi		
Nomor	9	
Nama	*Edit* data gardu.	
Tujuan	Admin dapat meng-*edit* data gardu yang sudah tersimpan sebelumnya di *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu data gardu, dan mengisi form peng-*edit*-an data gardu.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login*.
2. Admin mengisi *form login*.		
3. Admin menekan tombol *login*.		4. Menampilkan halaman utama *website*.
5. Admin memilih menu data gardu.		6. Menampilkan halaman data gardu
7. Admin memilih tambah gardu		8. Menampilkan halaman tambah data gardu.
9. Mengisi form data gardu.		
10. Admin menekan tombol simpan		11. Memeriksa apakah data sudah sesuai, jika : - Ya, simpan data gardu ke *database* server. - Tidak, kembali ke halaman data gardu.
Kondisi Akhir	Peng-*edit*-an data gardu berhasil dan data gardu yang telah tersimpan sebelumnya telah berubah.	

3.1.8.3.10. *Use case* sekenario Hapus Data Gardu

Interaksi antara aktor admin dengan *use case* hapus data gardu dijelaskan dalam tabel 3.16

Tabel 3.16 *Use Case* Skenario Hapus Data Gardu

Identifikasi		
Nomor	10	
Nama	Hapus data gardu.	
Tujuan	Admin dapat menghapus data gardu yang sudah tidak diperlukan lagi dari *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu dari data gardu yang tersimpan, dan memilih hapus data tersebut dari *database server*.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman login admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data gardu.		6. Menampilkan daftar data gardu.
7. Admin memilih data gardu.		
8. dan menekan tombol hapus.		9. Menampilkan jendela konfirmasi hapus data gardu.
10. Menjawab jendela konfirmasi hapus data gardu.		11. Menerima jawaban konfirmasi hapus data gardu, jika : - Ya, menghapus data gardu. - Tidak, kembali ke halaman menampilkan daftar data gardu.
Kondisi Akhir	Data gardu yang dipilih telah terhapus dari *database server*.	

3.1.8.3.11. *Use case* skenario Tambah Data Jenis Kerusakan

Interaksi antara aktor admin dengan *use case* tambah data jenis kerusakan dijelaskan dalam tabel 3.17

Tabel 3.17 *Use Case* Skenario Tambah Data Jenis Kerusakan

Identifikasi		
Nomor	11	
Nama	Tambah data jenis kerusakan.	
Tujuan	Admin dapat menambah data jenis kerusakan.	
Deskripsi	Pada halaman ini, admin mengisi *form* untuk menambah data jenis kerusakan, seperti nama jenis kerusakan, estimasi perbaikan (menit), dan keterangan.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data jenis kerusakan.		6. Menampilkan halaman data jenis kerusakan.
7. Admin memilih tambah jenis kerusakan		8. Menampilkan halaman tambah data jenis kerusakan.
9. Admin mengisi *form* tambah data jenis kerusakan.		

Aksi Aktor	Reaksi Sistem
10. Admin menekan tombol simpan.	11. Memeriksa apakah data yang diisikan telah sesuai, jika : - Ya, menyimpan data data kerusakan ke *database* server. - Tidak, kembali menampilkan ke halaman data jenis kerusakan.
Kondisi Akhir	Tambah data jenis kerusakan berhasil dan data tersimpan ke *database server*.

3.1.8.3.12. *Use case* skenario *Edit* Data Jenis Kerusakan

Interaksi antara aktor admin dengan *use case edit* data jenis kerusakan dijelaskan dalam tabel 3.18

Tabel 3.18 *Use Case* Skenario *Edit* Data Jenis Kerusakan

Identifikasi		
Nomor	12	
Nama	*Edit* data jenis kerusakan.	
Tujuan	Admin dapat meng-*edit* data jenis kerusakan yang sudah tersimpan sebelumnya di *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu data jenis kerusakan, dan mengisi form peng-*edit*-an data jenis kerusakan.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website*.
5. Admin memilih menu data jenis kerusakan.		6. Menampilkan daftar data jenis kerusakan yang tersimpan.
7. Admin memilih data jenis kerusakan.		
8. Admin menekan tombol *edit*.		9. Menampilkan form *edit* data jenis kerusakan.
10. Mengisi *form edit* data jenis kerusakan		
11. Admin menekan tombol simpan.		12. Memeriksa apakah data jenis kerusakan yang diisikan sesuai, jika : - Ya, meng-*update* data jenis kerusakan ke *database* server. - Tidak, kembali ke halaman menampilkan *form edit* data jenis kerusakan.
Kondisi Akhir	Peng-*edit*-an data jenis kerusakan berhasil dan data jenis kerusakan yang telah tersimpan sebelumnya telah berubah.	

3.1.8.3.13. *Use case* skenario Hapus Data Jenis Kerusakan

Interaksi antara aktor admin dengan *use case* hapus data jenis kerusakan dijelaskan dalam tabel 3.19

Tabel 3.19 *Use Case* Skenario Hapus Data Jenis Kerusakan

Identifikasi		
Nomor	13	
Nama	Hapus data jenis kerusakan.	
Tujuan	Admin dapat menghapus data jenis kerusakan yang sudah tidak diperlukan lagi dari *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu dari data jenis kerusakan yang tersimpan, dan memilih hapus data tersebut dari *database server*.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data jenis kerusakan.		6. Menampilkan daftar data jenis kerusakan.
7. Admin memilih data jenis kerusakan.		
8. Admin menekan tombol hapus.		9. Menampilkan jendela konfirmasi hapus data jenis kerusakan.
10. Menjawab jendela konfirmasi hapus data jenis kerusakan.		11. Memeriksa jawab konfirmasi hapus data jenis kerusakan, jika : - Ya, menghapus data jenis kerusakan. - Tidak, kembali menampilkan daftar data jenis kerusakan.
Kondisi Akhir	Data jenis kerusakan yang dipilih telah terhapus dari *database server*.	

3.1.8.3.14. *Use case* skenario Tambah Data Keluhan dan Gangguan

Interaksi antara aktor admin dengan *use case* tambah data keluhan dan gangguan dijelaskan dalam tabel 3.20

Tabel 3.20 *Use Case* Skenario Tambah Data Keluhan dan Gangguan

Identifikasi		
Nomor	14	
Nama	Tambah data keluhan dan gangguan.	
Tujuan	Admin dapat menambah data keluhan dan gangguan.	
Deskripsi	Pada halaman ini, admin mengisi *form* untuk menambah data keluhan dan gangguan.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data keluhan dan gangguan.		6. Menampilkan halaman data keluhan dan gangguan.

Aksi Aktor	Reaksi Sistem
7. Admin memilih tambah keluhan dan gangguan.	8. Menampilkan halaman tambah data keluhan dan gangguan.
9. Admin mengisi *form* tambah data keluhan dan gangguan.	
10. Admin menekan tombol simpan.	11. Memeriksa apakah data telah sesuai, jika : - Ya, menyimpan data keluhan dan gangguan ke *database* server. - Tidak, kembali menampilkan halaman tambah data keluhan dan gangguan.
Kondisi Akhir	Tambah data keluhan dan gangguan berhasil dan data tersimpan ke *database server*.

3.1.8.3.15. *Use case* skenario *Edit* Data Keluhan dan Gangguan

Interaksi antara aktor admin dengan *use case edit* data keluhan dan gangguan dijelaskan dalam tabel 3.21

Tabel 3.21 *Use Case* Skenario *Edit* Data Keluhan dan Gangguan

Identifikasi		
Nomor	15	
Nama	*Edit* data keluhan dan gangguan.	
Tujuan	Admin dapat meng-*edit* data keluhan dan gangguan yang sudah tersimpan sebelumnya di *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu data keluhan dan gangguan, dan mengisi *form* peng-*edit*-an data keluhan dan gangguan.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login*.
2. Admin mengisi *form login*.		
3. Admin menekan tombol *login*.		4. Menampilkan halaman utama *website*.
5. Admin memilih menu data keluhan dan gangguan.		6. Menampilkan daftar data keluhan dan gangguan yang tersimpan.
7. Admin memilih data keluhan dan gangguan.		
8. Admin menekan tombol *edit*.		9. Menampilkan *form edit* data keluhan dan gangguan.
10. Mengisi *form edit* data keluhan dan gangguan.		
11. Admin menekan tombol simpan.		12. Memeriksa apakah data keluhan dan gangguan yang diisikan telah sesuai, jika : - Ya, meng-*update* keluhan dan gangguan ke *database* server. - Tidak, kembali ke *form edit* data keluhan dan gangguan.
Kondisi Akhir	Peng-*edit*-an data keluhan dan gangguan berhasil dan data keluhan dan gangguan yang telah tersimpan sebelumnya telah berubah.	

3.1.8.3.16. *Use case* skenario Hapus Data Keluhan dan Gangguan

Interaksi antara aktor admin dengan *use case* hapus data keluhan dan gangguan dijelaskan dalam tabel 3.22

Tabel 3.22 *Use Case* Skenario Hapus Keluhan dan Gangguan

Identifikasi		
Nomor	16	
Nama	Hapus data keluhan dan gangguan.	
Tujuan	Admin dapat menghapus data keluhan dan gangguan yang sudah tidak diperlukan lagi dari *database server*.	
Deskripsi	Pada halaman ini, admin akan memilih salah satu dari data keluhan dan gangguan yang tersimpan, dan memilih hapus data tersebut dari *database server*.	
Aktor	Admin.	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login* admin.	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu data keluhan dan gangguan.		6. Menampilkan daftar data keluhan dan gangguan yang tersimpan..
7. Admin memilih data keluhan dan gangguan.		
8. Admin menekan tombol hapus.		9. Menampilkan jendela konfirmasi hapus data keluhan dan gangguan.
10. Menjawab jendela konfirmasi hapus data keluhan dan gangguan.		11. Mememrisa jawab komfirmasi hapus data keluhan dan gangguan, jika : - Ya, menghapus data keluhan dan gangguan. - Tidak, kembali menampilkan daftar data keluhan dan gangguan.
Kondisi Akhir	Data keluhan dan gangguan yang dipilih telah terhapus dari *database server*.	

3.1.8.3.17. *Use case* skenario Data Penjadwalan

Interaksi antara aktor *user* dengan *use case* penjadwalan dijelaskan dalam tabel 3.23

Tabel 3.23 *Use Case* Skenario Data Penjadwalan

Identifikasi	
Nomor	17
Nama	Data Penjadwalan
Tujuan	*User* dapat melihat data penjadwalan.
Deskripsi	Menu ini berisi *form* yang menampilkan output perhitungan algoritma genetika dari data penjadwalan serta informasi penugasan kepada regu YANTEK.
Aktor	Admin
Skenario Utama	
Kondisi Awal	Admin berada di halaman *login*

Aksi Aktor	Reaksi Sistem
	1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*	
3. Admin menekan tombol *Login.*	4. Menampilkan halaman utama aplikasi *website.*
5. Admin memilih menu data penjadwalan.	6. Menampilkan halaman data penjadwalan
Kondisi Akhir	Admin telah melihat data penjadwalan.

3.1.8.3.18. *Use case* skenario Data Ketentuan

Interaksi antara aktor *user* dengan *use case* data ketentuan dijelaskan dalam tabel 3.24

Tabel 3.24 *Use Case* Skenario Data Ketentuan

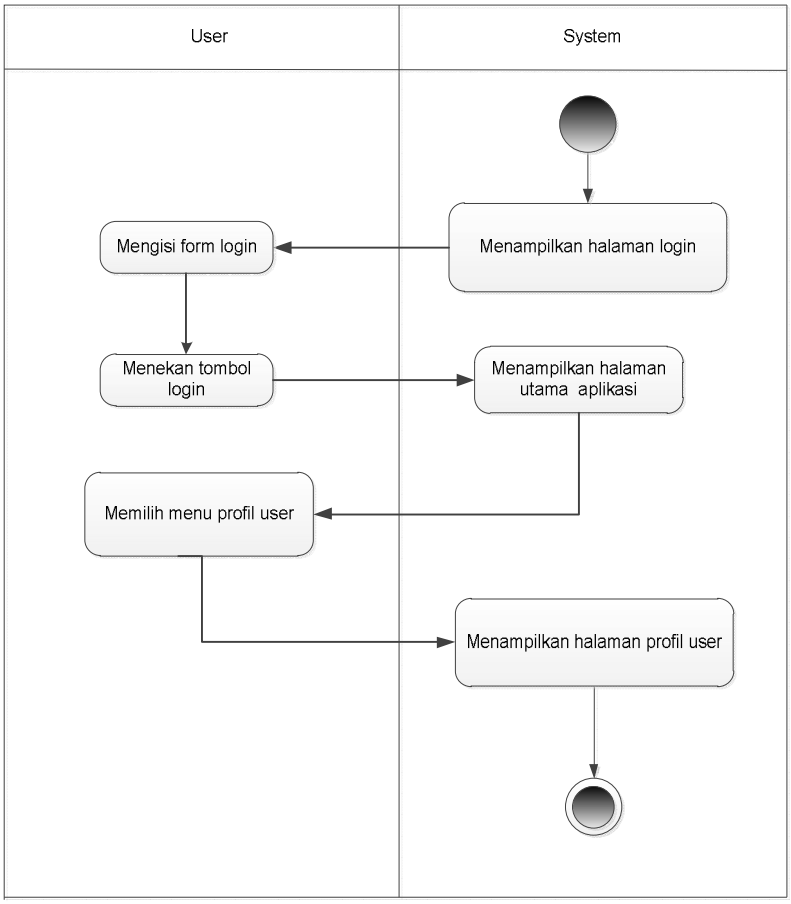
Identifikasi		
Nomor	18	
Nama	Data Ketentuan	
Tujuan	Admin dapat modifikasi data ketentuan.	
Deskripsi	Menu ini berisi *form* modifikasi data ketentuan untuk perhitungan algoritma genetika seperti jumlah jam kerja per hari, jatah liter BBM per hari, rata-rata asumsi kecepatan kendaraan(km/jam), dan jumlah jam istirahat (toleransi) per hari .	
Aktor	Admin	
Skenario Utama		
Kondisi Awal	Admin berada di halaman *login*	
Aksi Aktor		**Reaksi Sistem**
		1. Menampilkan halaman *login.*
2. Admin mengisi *form login.*		
3. Admin menekan tombol *login.*		4. Menampilkan halaman utama *website.*
5. Admin memilih menu ketentuan.		6. Aplikasi menampilkan *form* data ketentuan.
7. Admin mengisi *field* modifikasi ketentuan		
8. Admin memilih tombol simpan		9. Memeriksa apakah data ketentuan yang diisikan sesuai, jika : - Ya, meng-*update* data ketentuan ke *database* server. - Tidak, kembali menampilkan *form* data ketentuan.
Kondisi Akhir	Admin telah modifikasi data ketentuan.	

3.1.8.4. *Activity Diagram*

Activity diagram merupakan *diagram* yang memodelkan aliran kerja atau *workflow* dari urutan aktifitas dalam suatu proses yang mengacu pada *use case diagram* yang ada. Berikut ini penjelasan dari tiap *activity diagram*:

3.1.8.4.1. ***Activity Diagram* Profil *User***

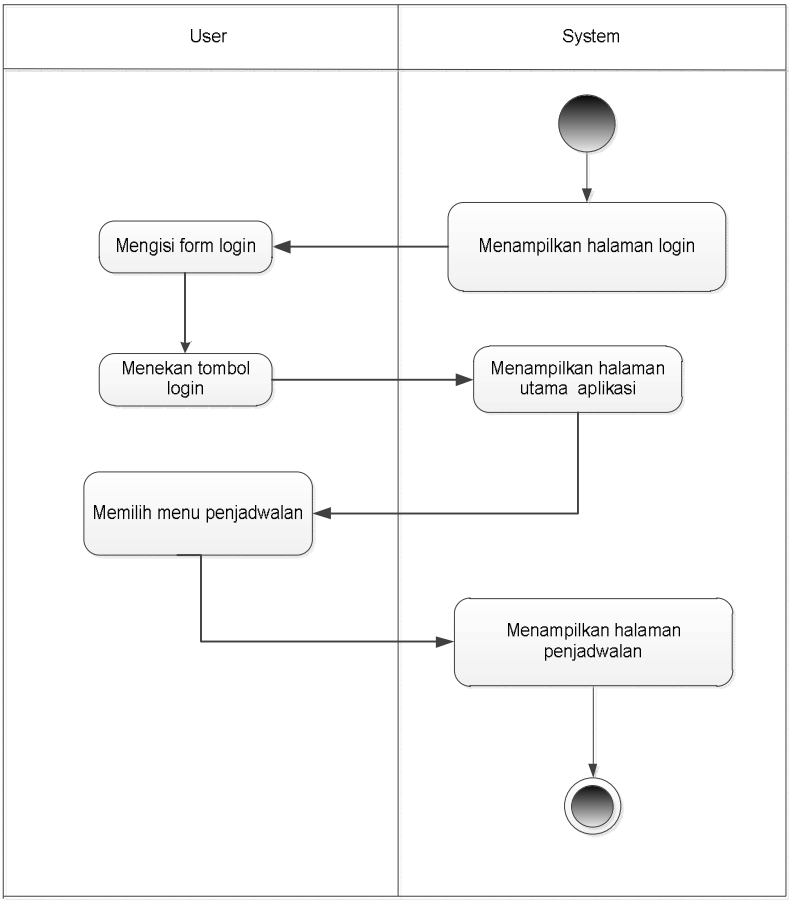
Activity diagram profil *user* menjelaskan aliran kerja aktor pengguna pada saat memilih menu profil *user* pada halaman menu utama. *Activity diagram* profil *user* dapat dilihat pada gambar 3.6.



Gambar 3.6 *Activity Diagram* Profil *User*

3.1.8.4.2. ***Activity Diagram* Penjadwalan**

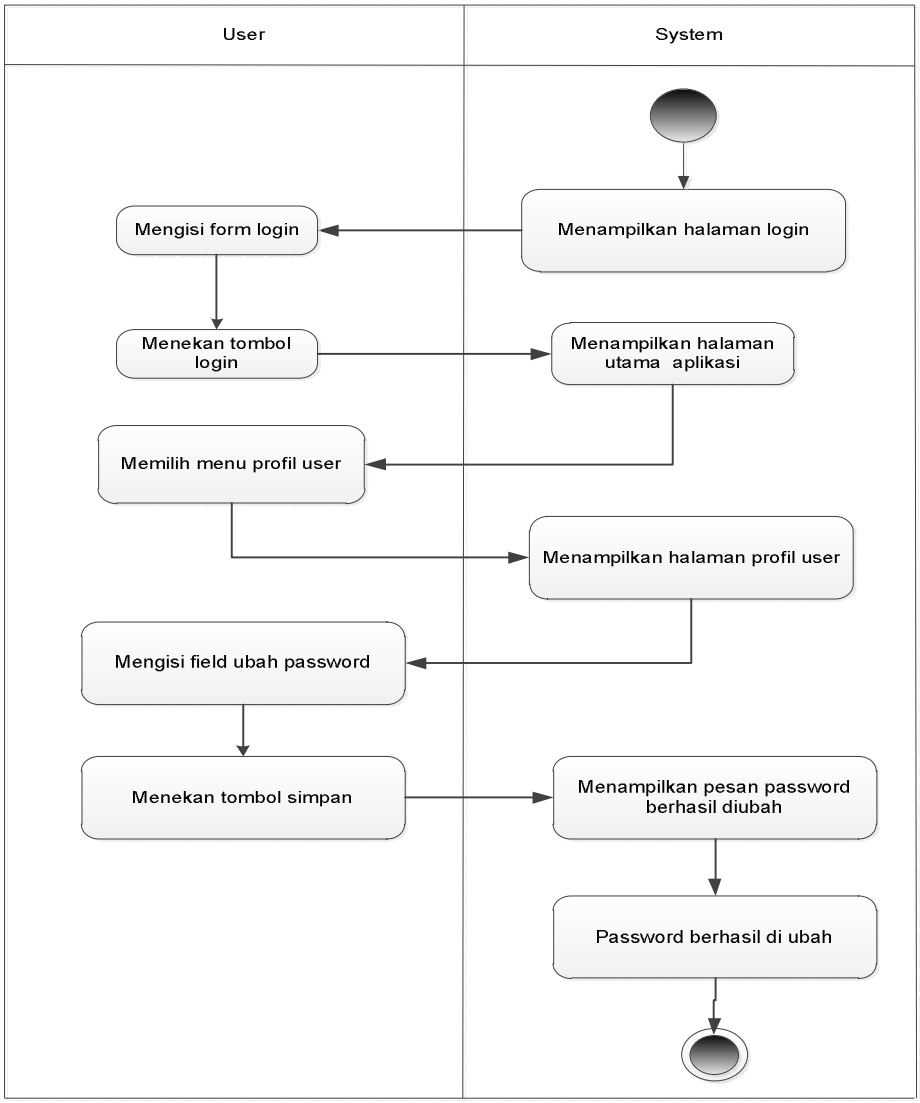
Activity diagram penjadwalan menjelaskan aliran kerja aktor pengguna pada saat memilih menu penjadwalan pada halaman menu utama. *Activity diagram* penjadwalan dapat dilihat pada gambar 3.7.



Gambar 3.7 *Activity Diagram* Penjadwalan

3.1.8.4.3. ***Activity Diagram* Ubah *Password***

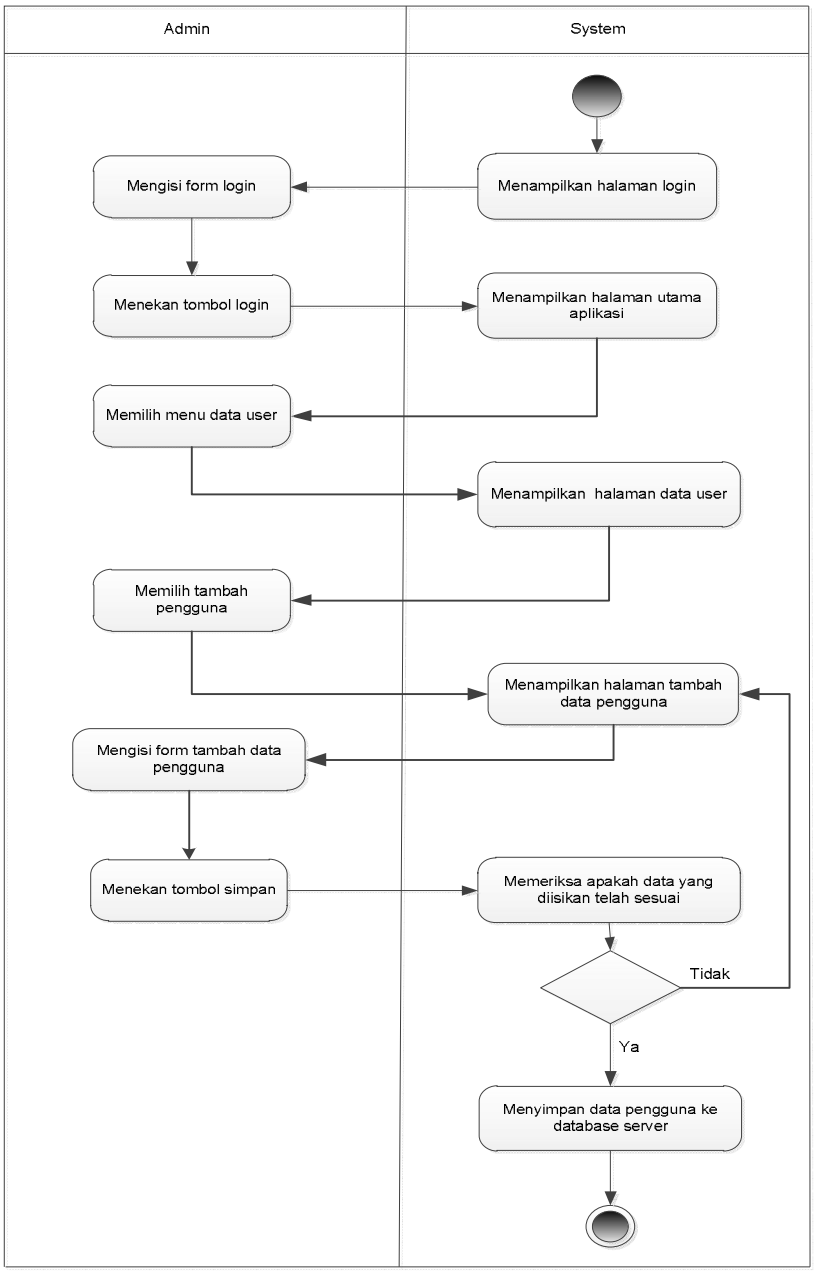
Activity diagram ubah *password* menjelaskan aliran kerja aktor pengguna pada saat memilih menu ubah *password* pada halaman menu utama. *Activity diagram* ubah *password* dapat dilihat pada gambar 3.8.



Gambar 3.8 *Activity Diagram* Ubah *Password*

3.1.8.4.4. ***Activity Diagram* Tambah Data Pengguna**

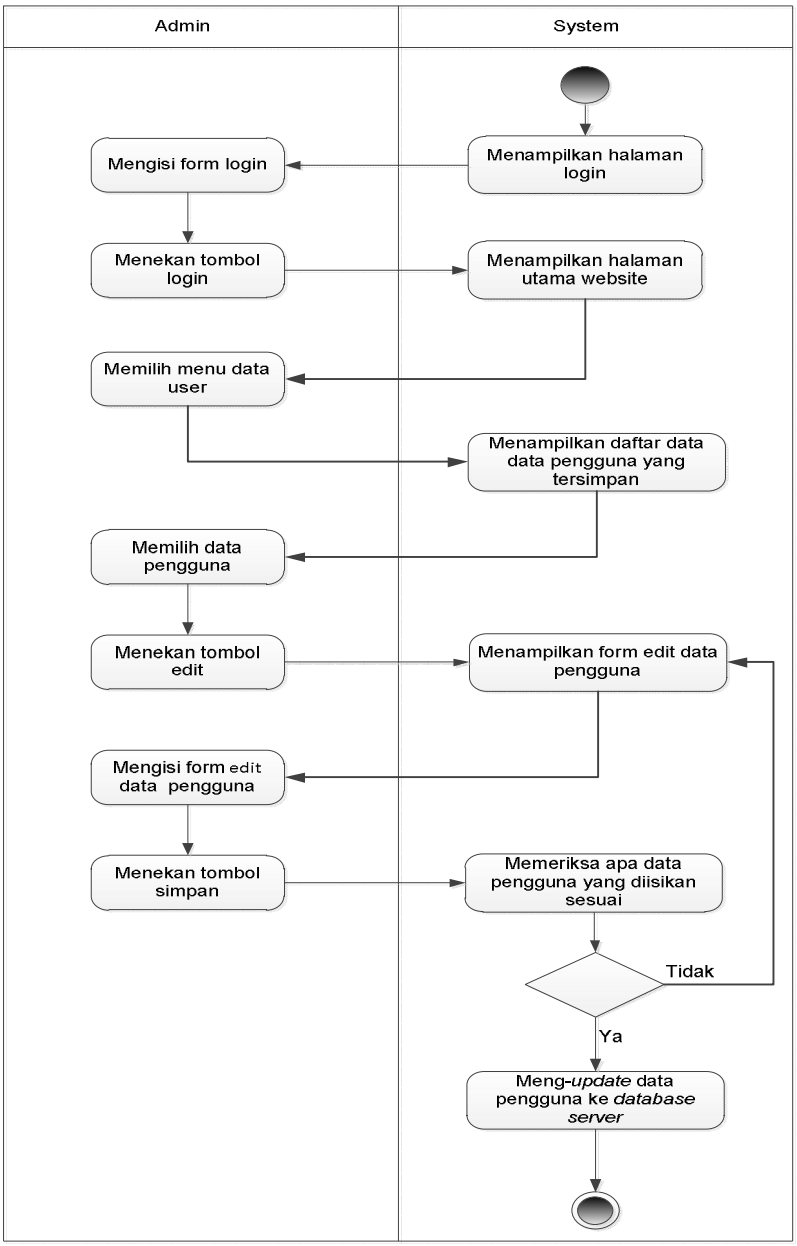
Activity diagram tambah data pengguna menjelaskan aliran kerja aktor admin pada saat melakukan penambahan data pengguna ke *database server*. *Activity diagram* tambah data pengguna dapat dilihat pada gambar 3.9.



Gambar 3.9 *Activity Diagram* Tambah Data Pengguna

3.1.8.4.5. *Activity Diagram Edit* Data Pengguna

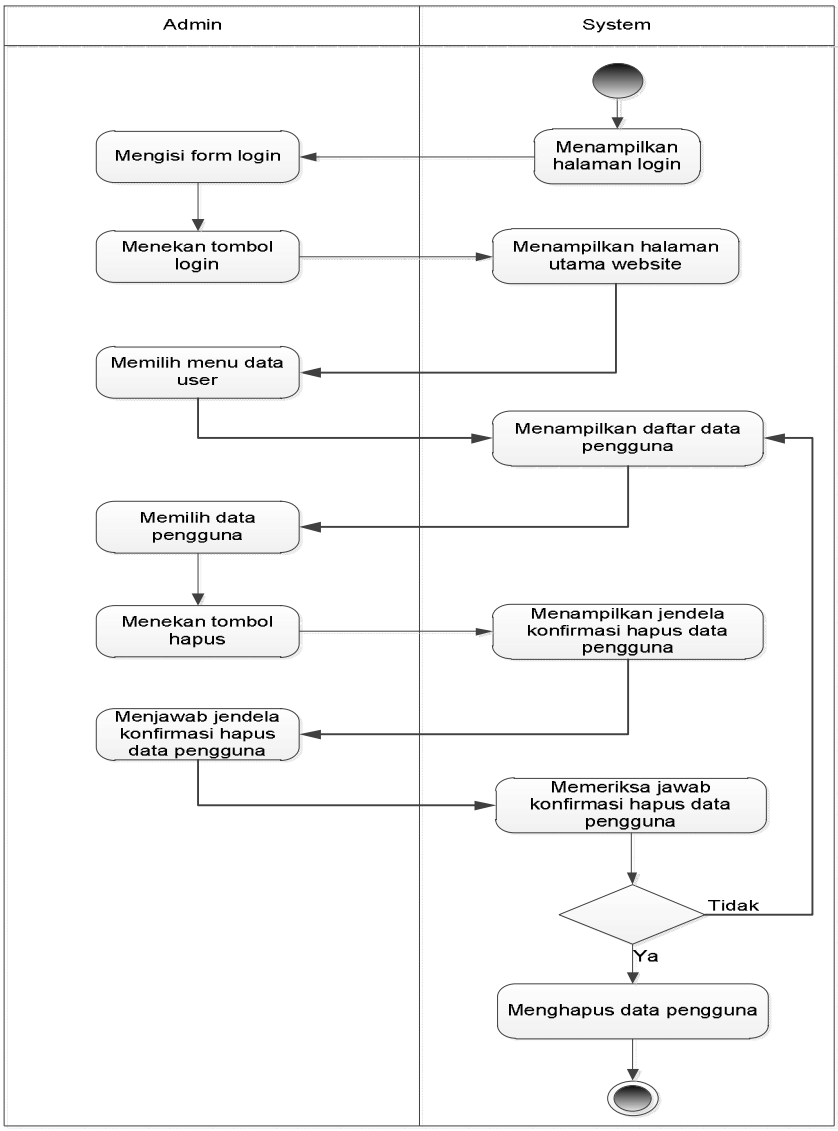
Activity diagram edit data pengguna menjelaskan aliran kerja aktor admin pada saat melakukan peng-*edit*-an data pengguna yang telah tersimpan di *database server*. *Activity diagram edit* data pengguna dapat dilihat pada gambar 3.10.



Gambar 3.10 *Activity Diagram Edit* Data Pengguna

3.1.8.4.6. ***Activity Diagram* Hapus Data Pengguna**

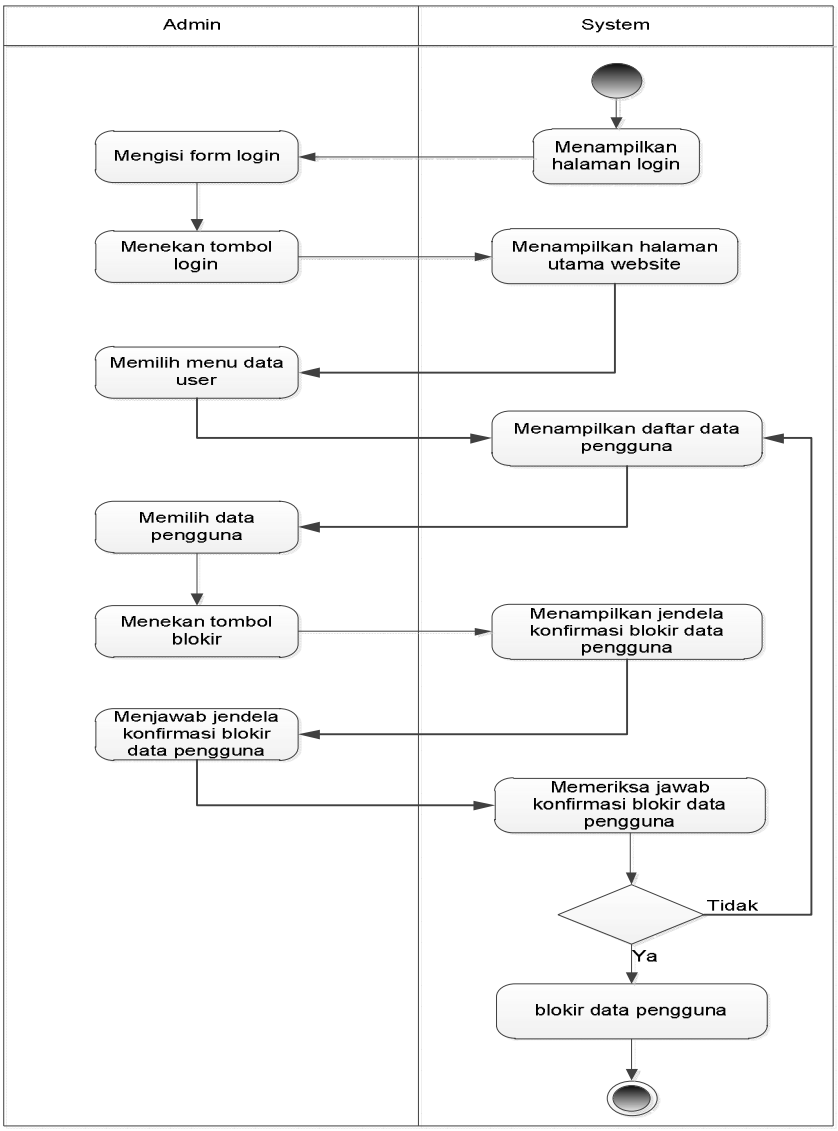
Activity diagram hapus data pengguna menjelaskan aliran kerja aktor admin pada saat melakukan penghapusan data pengguna dari *database server*. *Activity diagram* hapus data pengguna dapat dilihat pada gambar 3.11.



Gambar 3.11 *Activity Diagram* Hapus Data Pengguna

3.1.8.4.7. *Activity Diagram* Blokir Data Pengguna

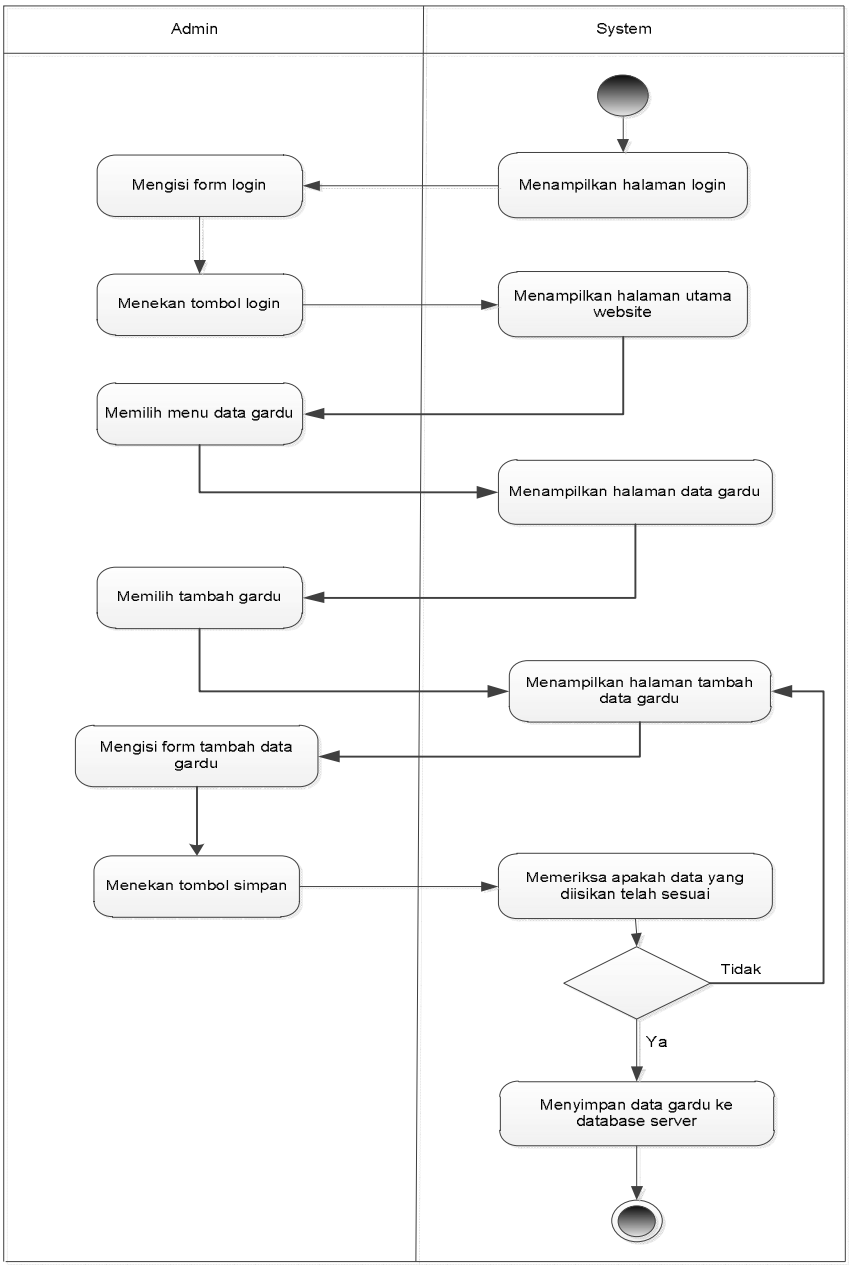
Activity diagram blokir data pengguna menjelaskan aliran kerja aktor admin pada saat melakukan pemblokiran data pengguna dari *database server*. *Activity diagram* blokir data pengguna dapat dilihat pada gambar 3.12.



Gambar 3.12 *Activity Diagram* Blokir Data Pengguna

3.1.8.4.8. ***Activity Diagram* Tambah Data Gardu**

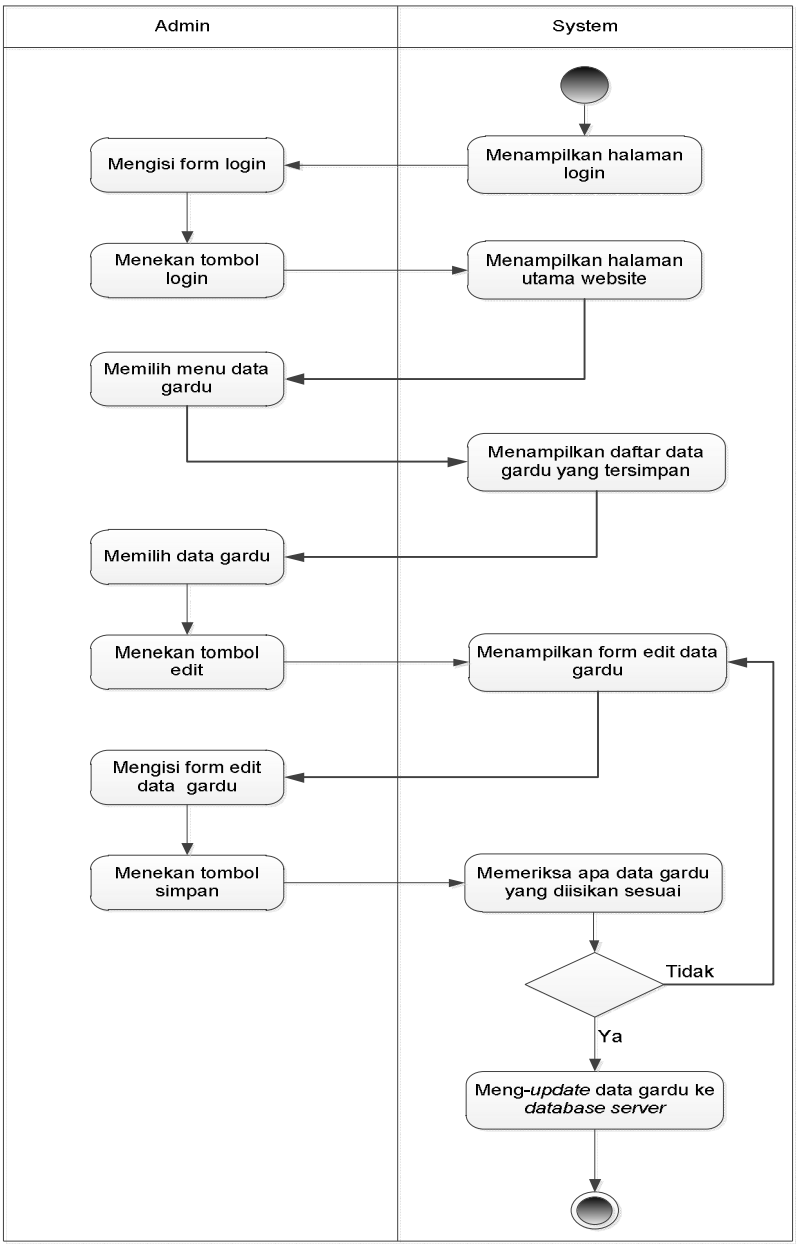
Activity diagram tambah data gardu menjelaskan aliran kerja aktor admin pada saat melakukan penambahan data gardu ke *database server*. *Activity diagram* tambah data gardu dapat dilihat pada gambar 3.13.



Gambar 3.13 *Activity Diagram* Tambah Data Gardu

3.1.8.4.9. *Activity Diagram Edit* Data Gardu

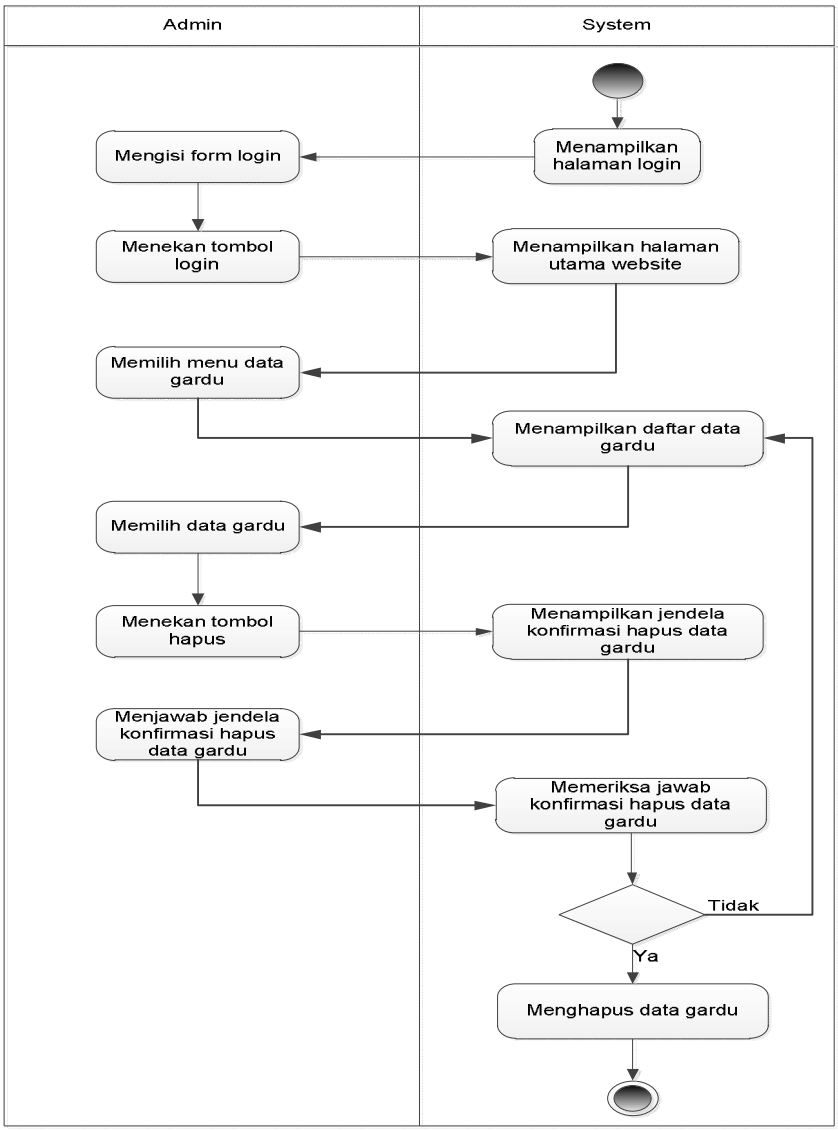
Activity diagram edit data gardu menjelaskan aliran kerja aktor admin pada saat melakukan peng-*edit*-an data gardu yang telah tersimpan di *database server*. *Activity diagram edit* data gardu dapat dilihat pada gambar 3.14.



Gambar 3.14 *Activity Diagram Edit* Data Gardu

3.1.8.4.10. ***Activity Diagram* Hapus Data Gardu**

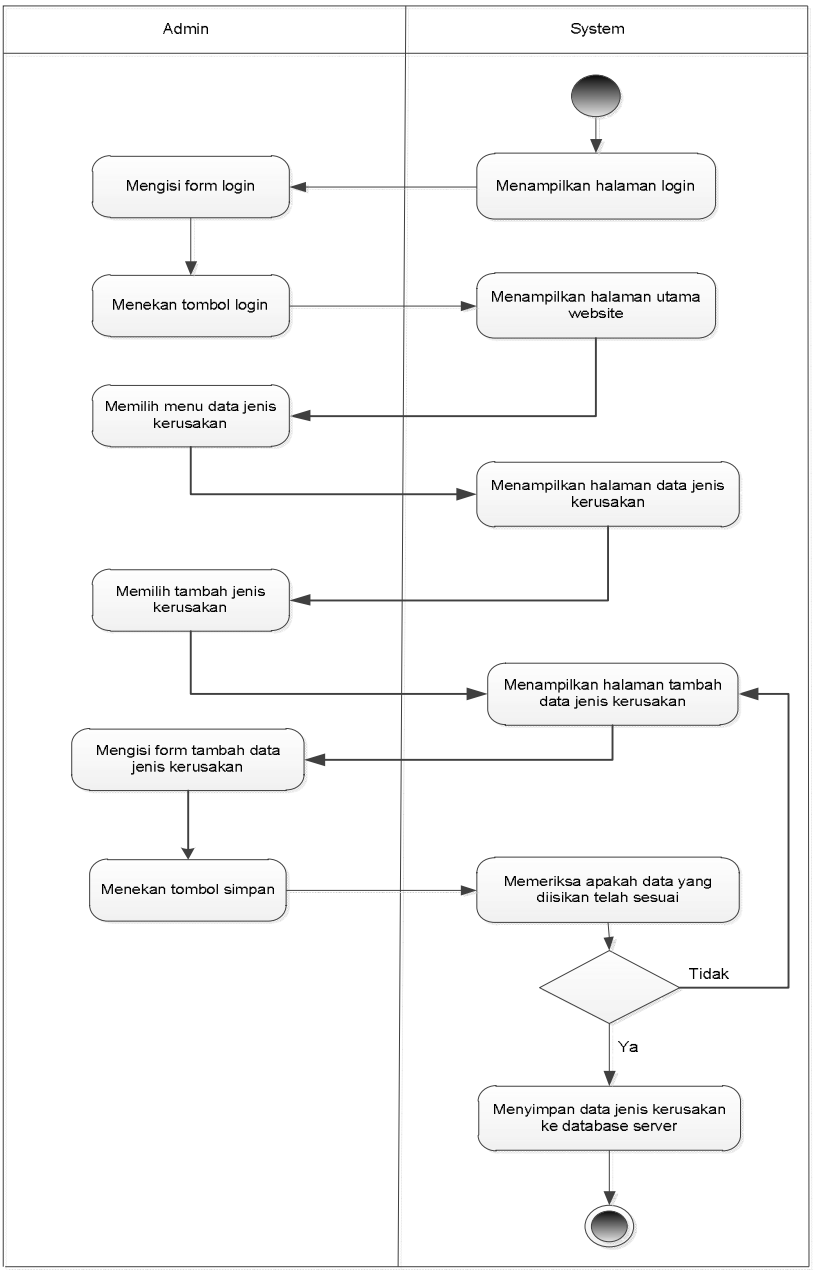
Activity diagram hapus data gardu menjelaskan aliran kerja aktor admin pada saat melakukan penghapusan data gardu dari *database server*. *Activity diagram* hapus data gardu dapat dilihat pada gambar 3.15.



Gambar 3.15 *Activity Diagram* Hapus Data Gardu

3.1.8.4.11. ***Activity Diagram* Tambah Data Jenis Kerusakan**

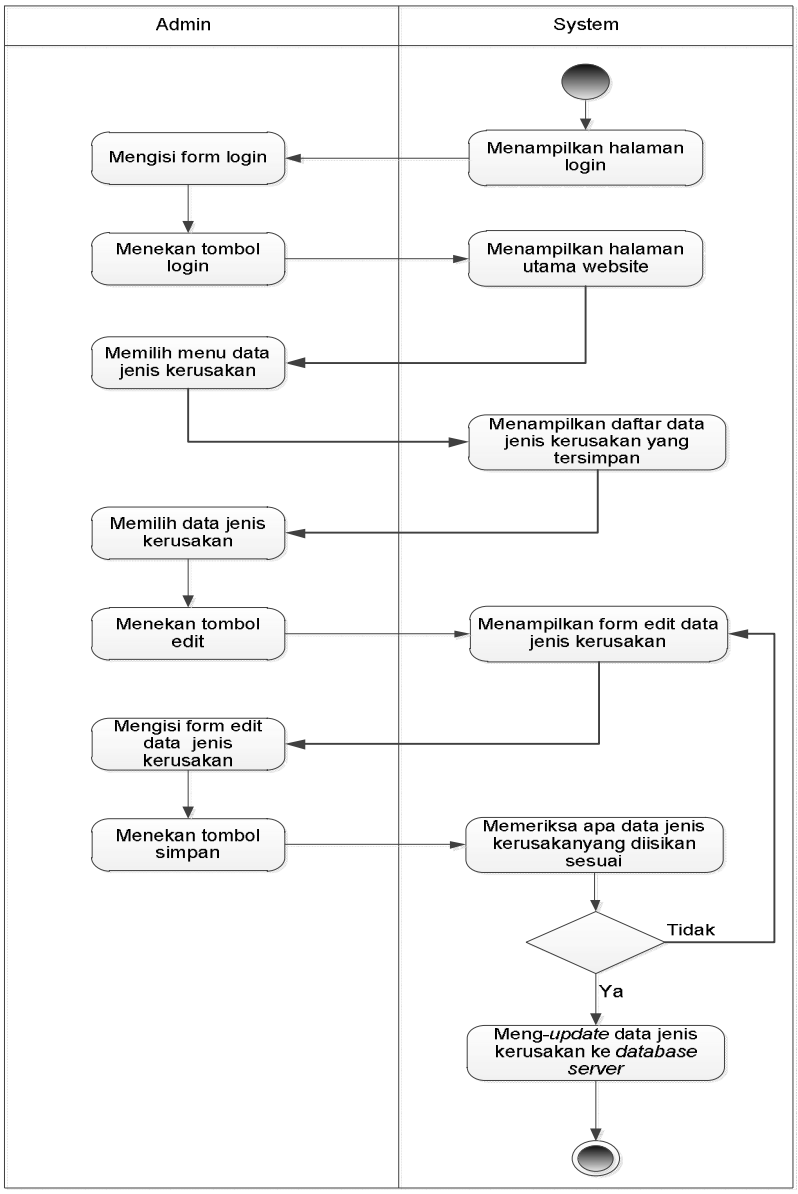
Activity diagram tambah data jenis kerusakan menjelaskan aliran kerja aktor admin pada saat melakukan penambahan data jenis kerusakan ke *database server*. *Activity diagram* tambah data jenis kerusakan dapat dilihat pada gambar 3.16.



Gambar 3.16 *Activity Diagram* Tambah Data Jenis Kerusakan

3.1.8.4.12. ***Activity Diagram Edit* Data Jenis Kerusakan**

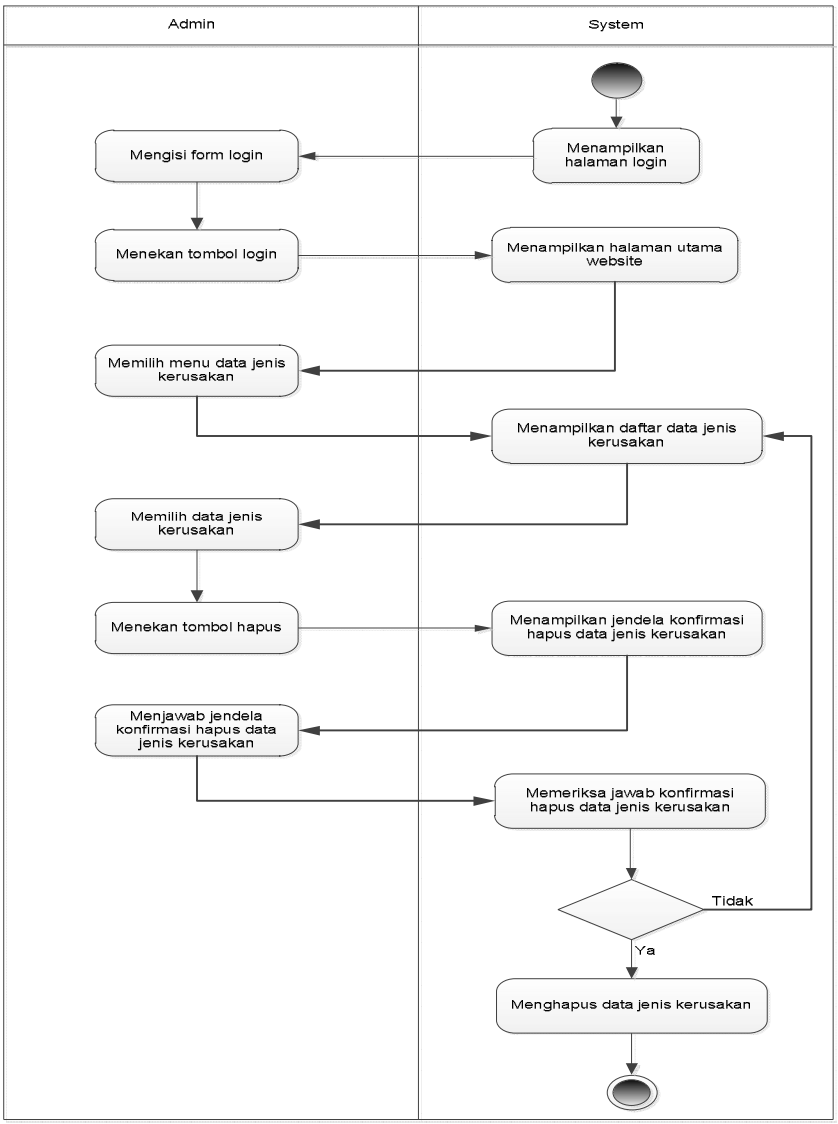
Activity diagram edit data jenis kerusakan menjelaskan aliran kerja aktor admin pada saat melakukan peng-*edit*-an data jenis kerusakan yang telah tersimpan di *database server*. *Activity diagram edit* data jenis kerusakan dapat dilihat pada gambar 3.17.



Gambar 3.17 *Activity Diagram Edit* Data Jenis Kerusakan

3.1.8.4.13. ***Activity Diagram*** **Hapus Data Jenis Kerusakan**

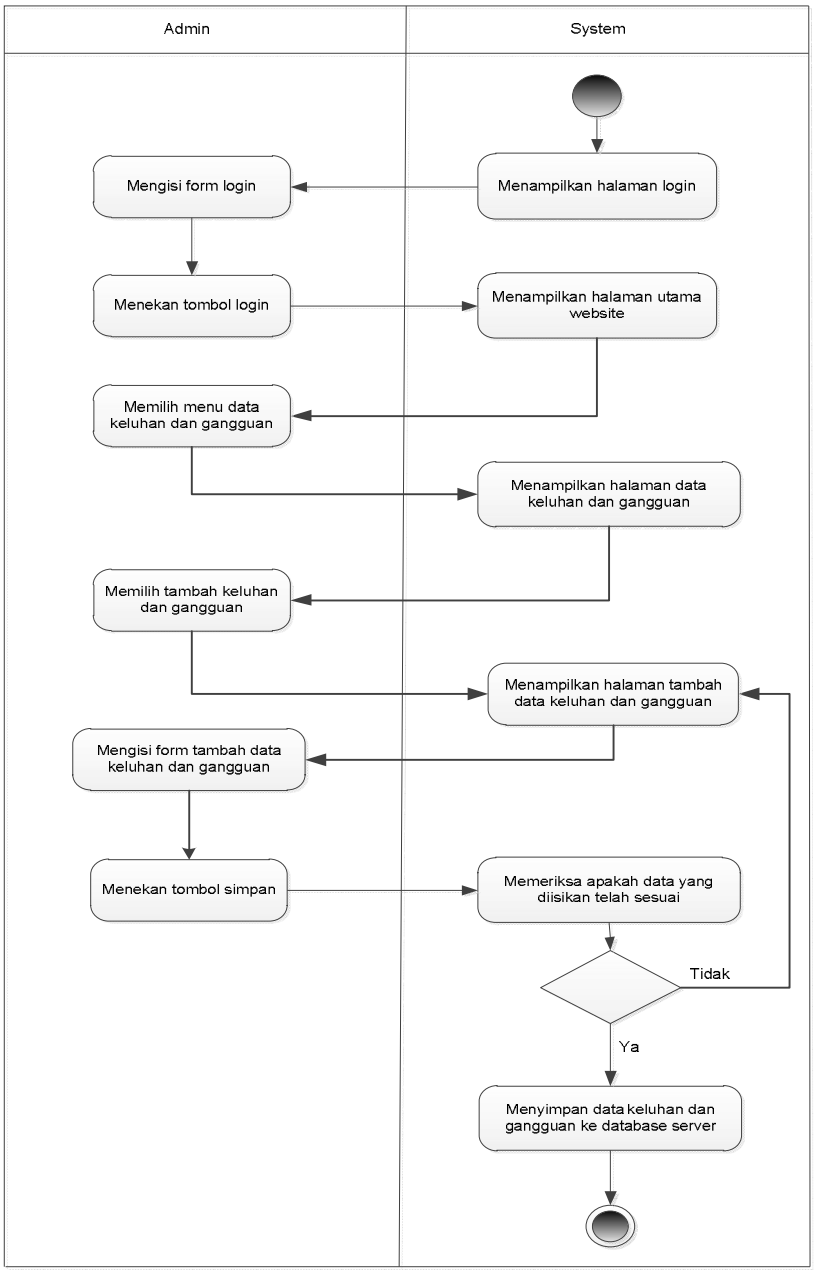
Activity diagram hapus data jenis kerusakan menjelaskan aliran kerja aktor admin pada saat melakukan penghapusan data jenis kerusakan dari *database server*. *Activity diagram* hapus data jenis kerusakan dapat dilihat pada gambar 3.18.



Gambar 3.18 ***Activity Diagram*** **Hapus Data Jenis Kerusakan**

3.1.8.4.14. ***Activity Diagram*** **Tambah Data Keluhan dan Gangguan**

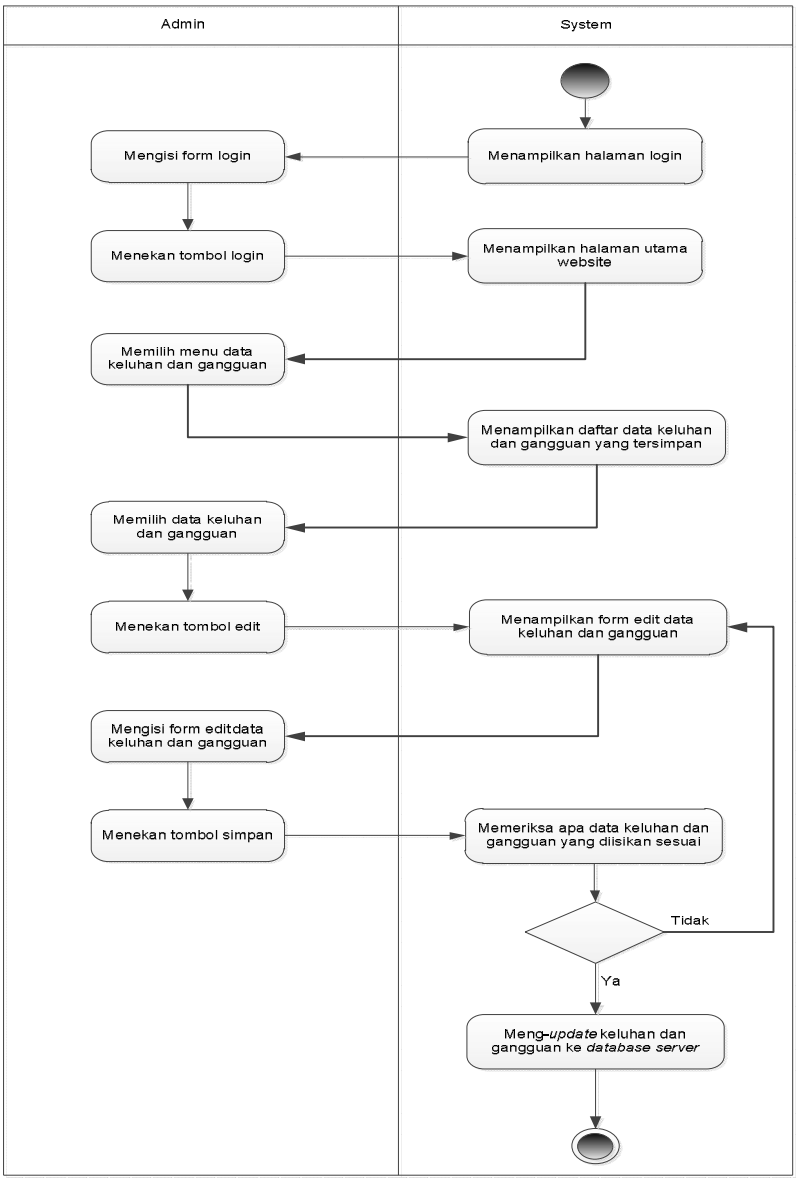
Activity diagram tambah data keluhan dan gangguan menjelaskan aliran kerja aktor admin pada saat melakukan penambahan data keluhan dan gangguan ke *database server*. *Activity diagram* tambah data keluhan dan gangguan dapat dilihat pada gambar 3.19.



Gambar 3.19 ***Activity Diagram*** **Tambah Data Keluhan dan Gangguan**

3.1.8.4.15. ***Activity Diagram Edit* Data Keluhan dan Gangguan**

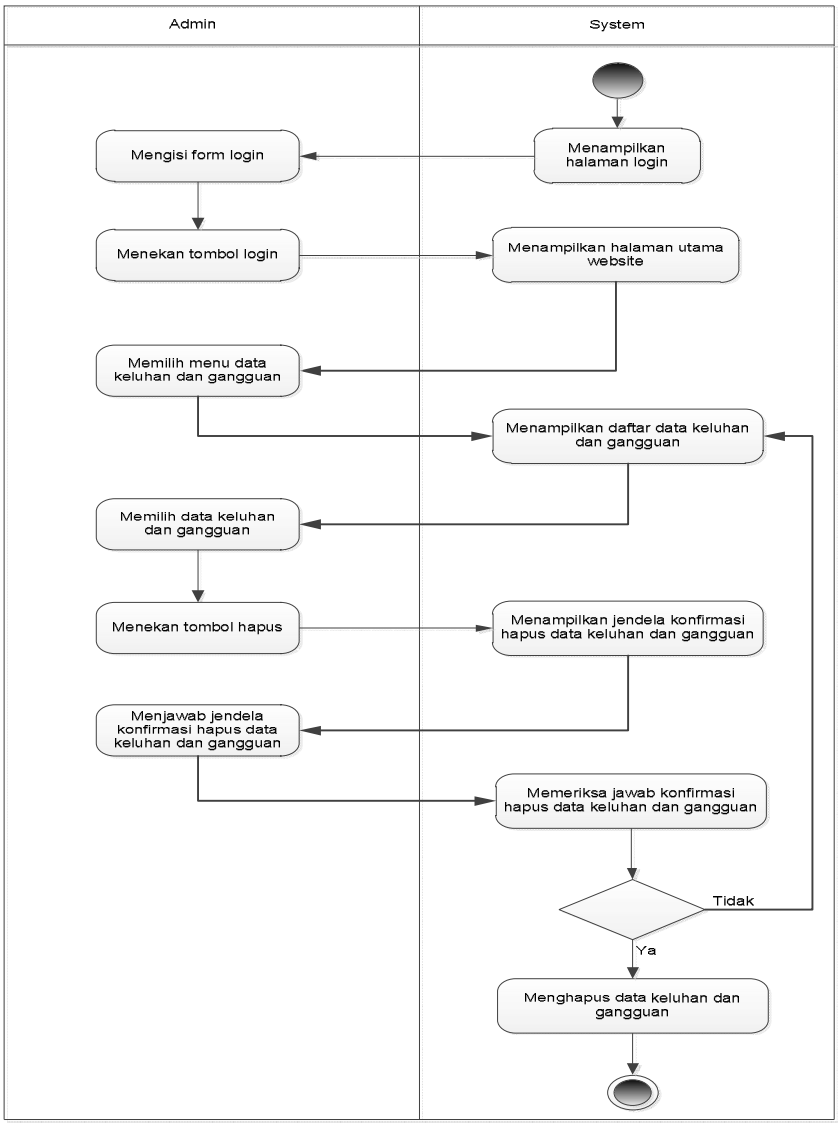
Activity diagram edit data keluhan dan gangguan menjelaskan aliran kerja aktor admin pada saat melakukan peng-*edit*-an data keluhan dan gangguan yang telah tersimpan di *database server*. *Activity diagram edit* data keluhan dan gangguan dapat dilihat pada gambar 3.20.



Gambar 3.20 *Activity Diagram Edit* Data Keluhan dan Gangguan

3.1.8.4.16. ***Activity Diagram* Hapus Data Keluhan dan Gangguan**

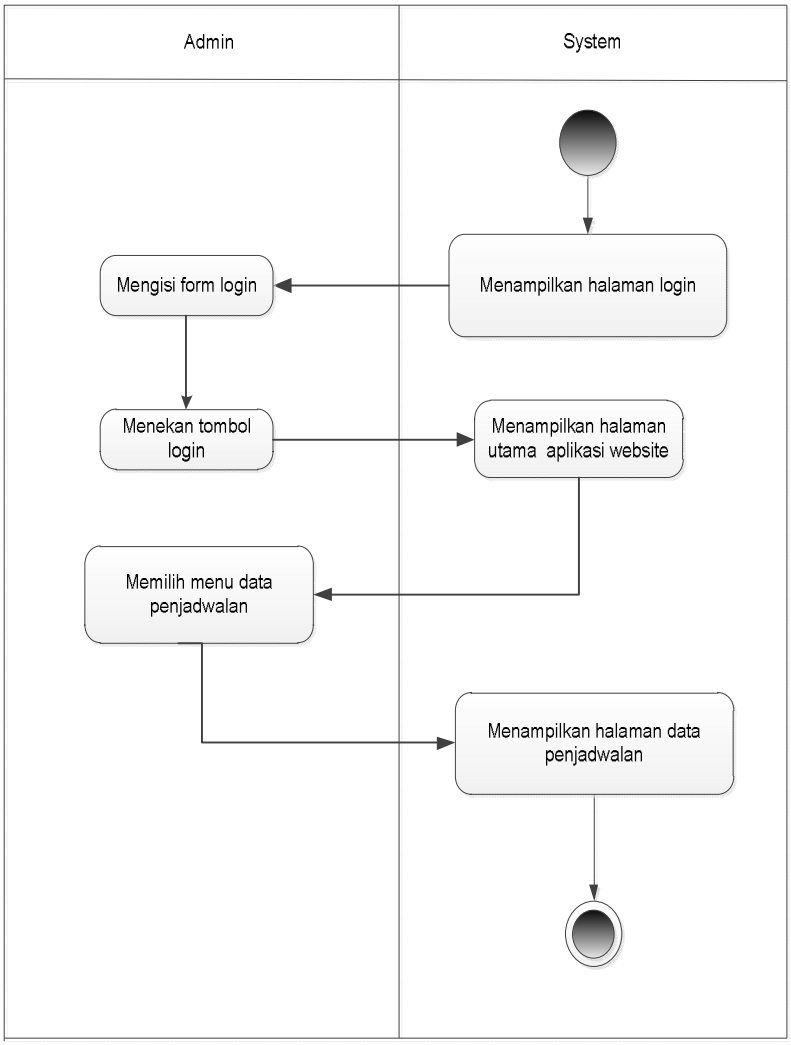
Activity diagram hapus data keluhan dan gangguan menjelaskan aliran kerja aktor admin pada saat melakukan penghapusan data keluhan dan gangguan dari *database server*. *Activity diagram* hapus data keluhan dan gangguan dapat dilihat pada gambar 3.21.



Gambar 3.21 *Activity Diagram* Hapus Data Keluhan dan Gangguan

3.1.8.4.17. ***Activity Diagram* Data Penjadwalan**

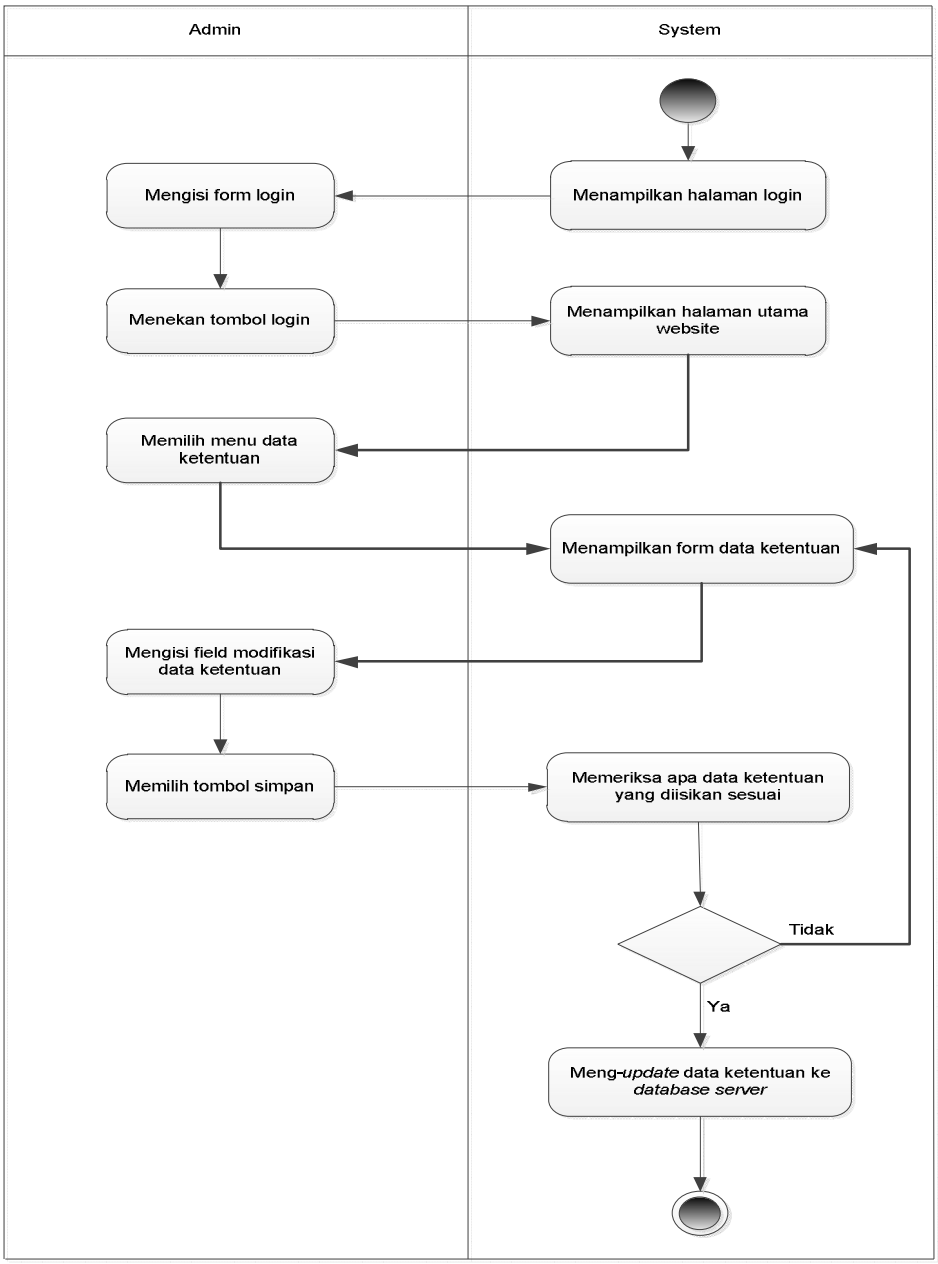
Activity diagram data penjadwalan menjelaskan aliran kerja aktor admin pada saat memilih menu penjadwalan pada halaman menu utama website. *Activity diagram* data penjadwalan dapat dilihat pada gambar 3.22.



Gambar 3.22 *Activity Diagram* Data Penjadwalan

3.1.8.4.18. ***Activity Diagram* Data Ketentuan**

Activity diagram data ketentuan menjelaskan aliran kerja aktor admin pada saat melakukan modifikasi data ketentuan yang telah tersimpan di *database server*. *Activity diagram* data ketentuan dapat dilihat pada gambar 3.23.



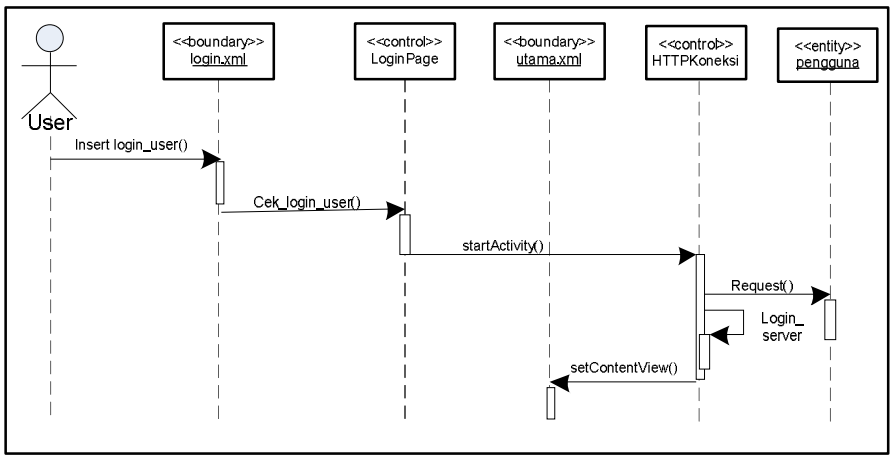
Gambar 3.23 *Activity Diagram* Data Ketentuan

3.1.8.5. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar masing-masing objek pada setiap *use case* dalam urutan waktu. Interaksi ini berupa pengiriman serangkaian data antar objek-objek yang saling berinteraksi.

3.1.8.5.1. *Sequence Diagram Login User*

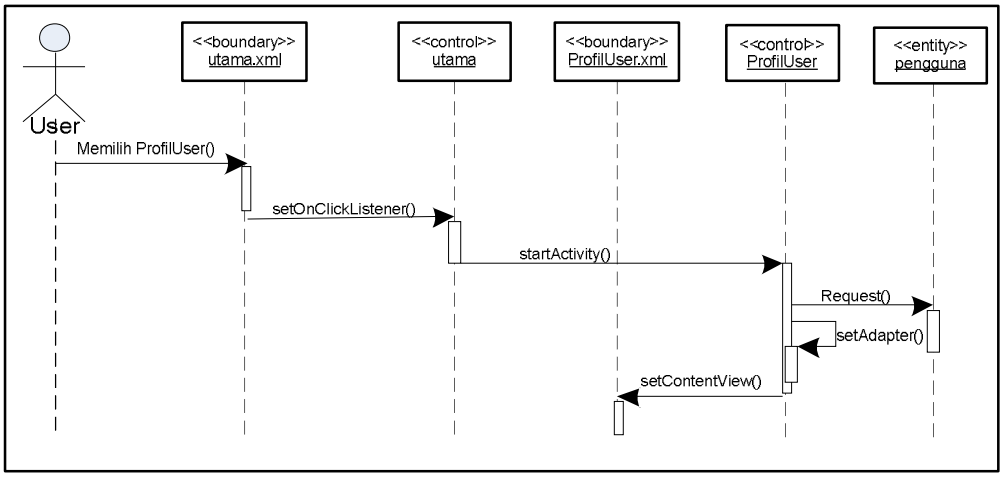
Sequence diagram login user menunjukkan rangkaian pesan yang dikirim antara objek *user* dengan objek lainnya dalam proses login. *Sequence diagram login user* dapat dilihat pada gambar 3.24.



Gambar 3.24 *Sequence Diagram* Login *User*

3.1.8.5.2. *Sequence Diagram* Profil *User*

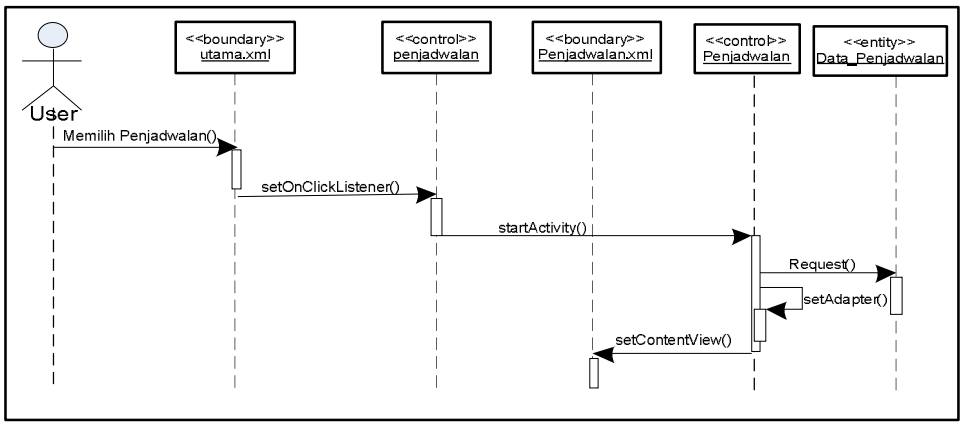
Sequence diagram profil *user* menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses melihat profil *user*. *Sequence diagram* profil *user* dapat dilihat pada gambar 3.25.



Gambar 3.25 *Sequence Diagram* Profil *User*

3.1.8.5.3. ***Sequence Diagram* Penjadwalan**

Sequence diagram penjadwalan menunjukkan rangkaian pesan yang dikirim antara objek pengguna dengan objek lainnya dalam proses melihat penjadwalan. *Sequence diagram* penjadwalan dapat dilihat pada gambar 3.26.

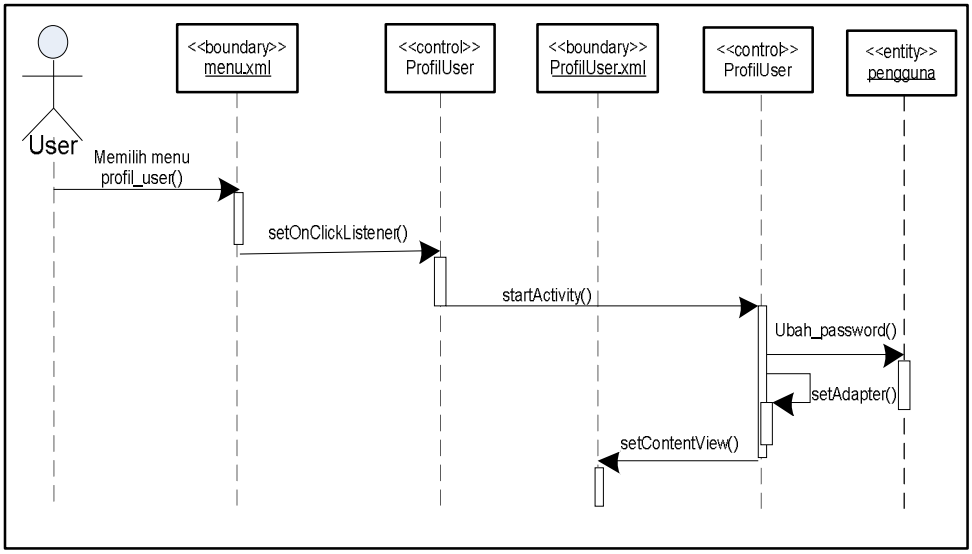


Gambar 3.26 *Sequence Diagram* Penjadwalan

3.1.8.5.4. ***Sequence Diagram* Ubah *Password***

Sequence diagram ubah *password* menunjukkan rangkaian pesan yang dikirim antara objek *user* dengan objek lainnya dalam proses ubah

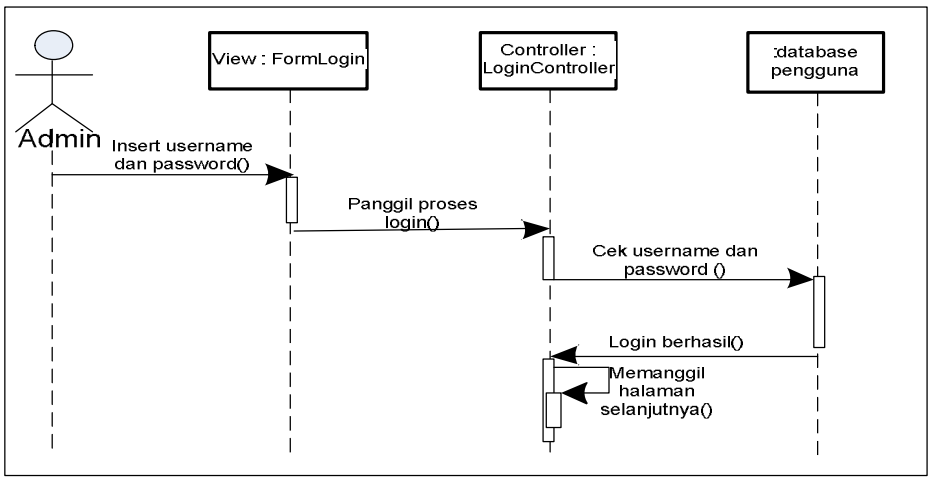
password. *Sequence diagram* ubah *password* dapat dilihat pada gambar 3.27.



Gambar 3.27 *Sequence Diagram* Ubah *Password*

3.1.8.5.5. ***Sequence Diagram Login* Admin**

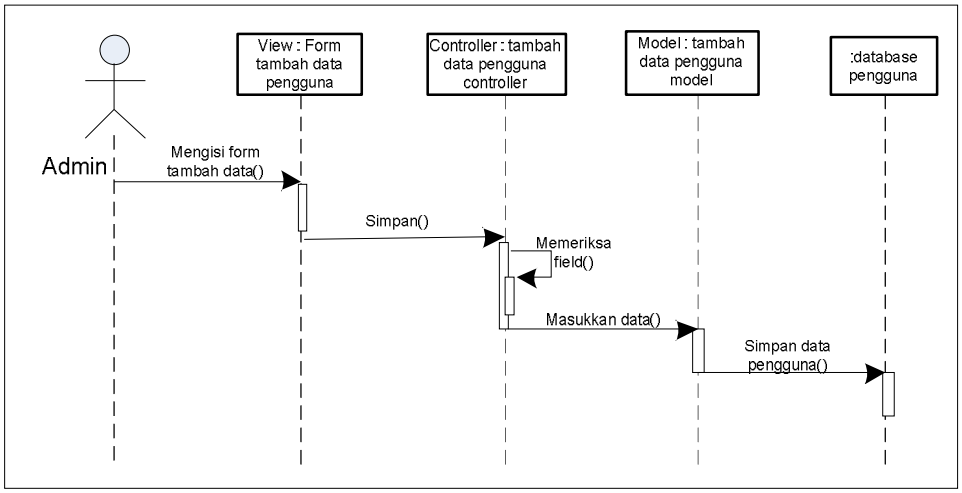
Sequence diagram Login Admin menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses *login*. *Sequence diagram login* admin dapat dilihat pada gambar 3.28.



Gambar 3.28 *Sequence Diagram* Login Admin

3.1.8.5.6. ***Sequence Diagram* Tambah Data Pengguna**

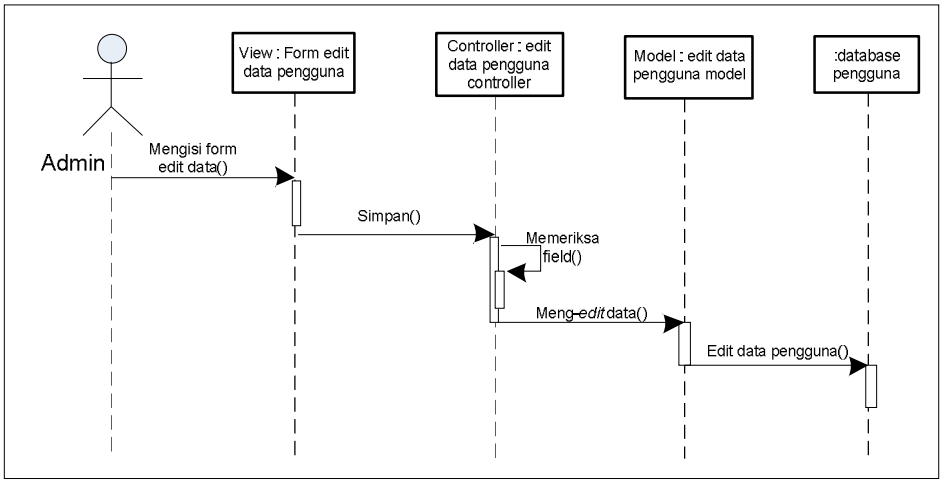
Sequence diagram tambah data Pengguna menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menambah data pengguna. *Sequence diagram* tambah data pengguna dapat dilihat pada gambar 3.29.



Gambar 3.29 *Sequence Diagram* Tambah Data Pengguna

3.1.8.5.7. ***Sequence Diagram Edit* Data Pengguna**

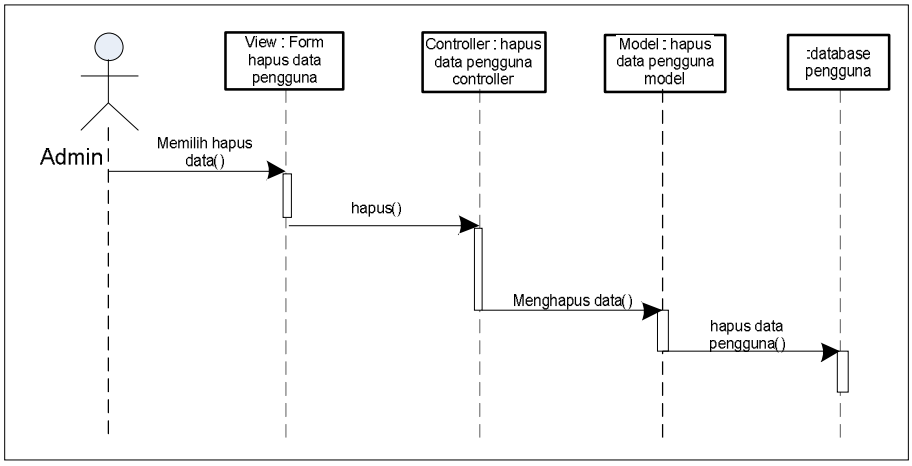
Sequence diagram edit data pengguna menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses meng-*edit* data pengguna. *Sequence diagram edit* data pengguna dapat dilihat pada gambar 3.30.



Gambar 3.30 *Sequence Diagram Edit* Data Pengguna

3.1.8.5.8. ***Sequence Diagram* Hapus Data Pengguna**

Sequence diagram hapus data pengguna menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menghapus data pengguna. *Sequence diagram* hapus data pengguna dapat dilihat pada gambar 3.31.

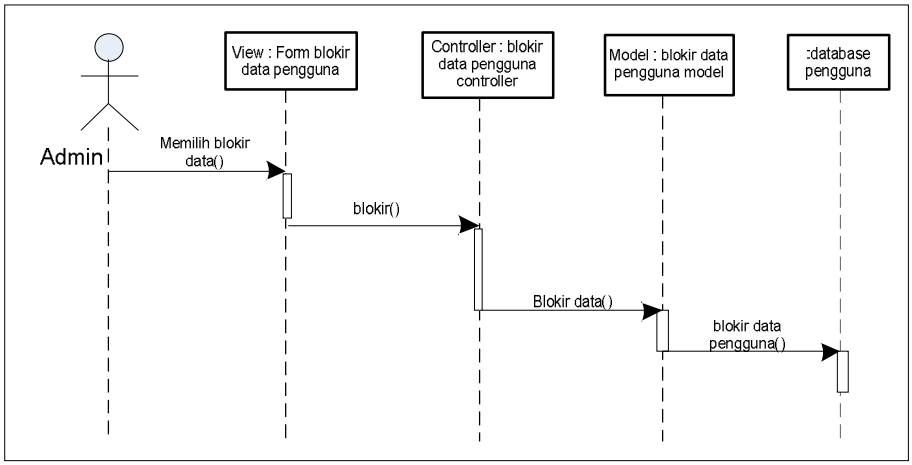


Gambar 3.31 *Sequence Diagram* Hapus Data Pengguna

3.1.8.5.9. ***Sequence Diagram* Blokir Data Pengguna**

Sequence diagram blokir data pengguna menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses

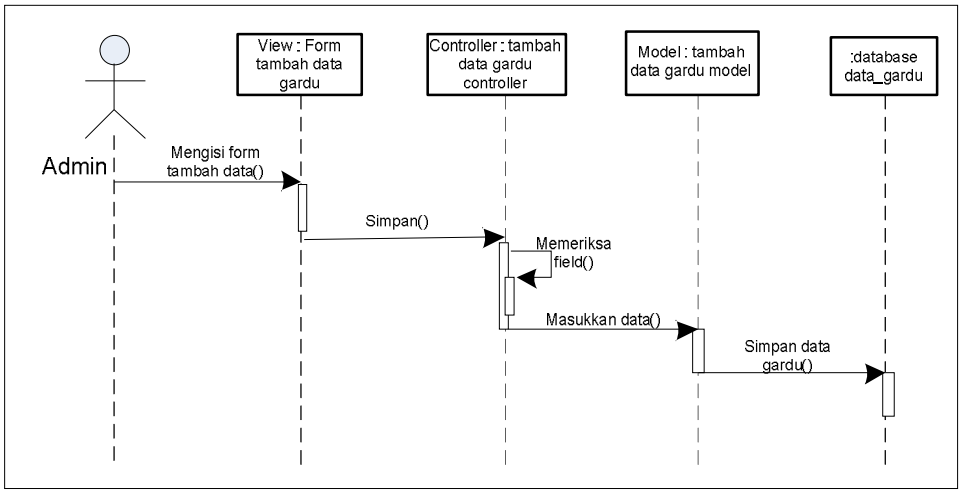
blokir data pengguna. *Sequence diagram* blokir data pengguna dapat dilihat pada gambar 3.32.



Gambar 3.32 *Sequence Diagram* Blokir Data Pengguna

3.1.8.5.10. ***Sequence Diagram* Tambah Data Gardu**

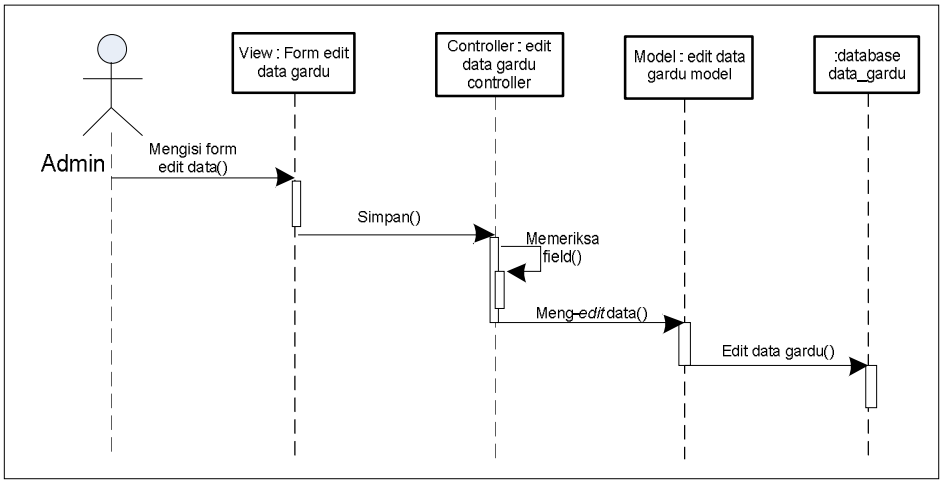
Sequence diagram tambah data gardu menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menambah data gardu. *Sequence diagram* tambah data gardu dapat dilihat pada gambar 3.33.



Gambar 3.33 *Sequence Diagram* Tambah Data Gardu

3.1.8.5.11. ***Sequence Diagram Edit* Data Gardu**

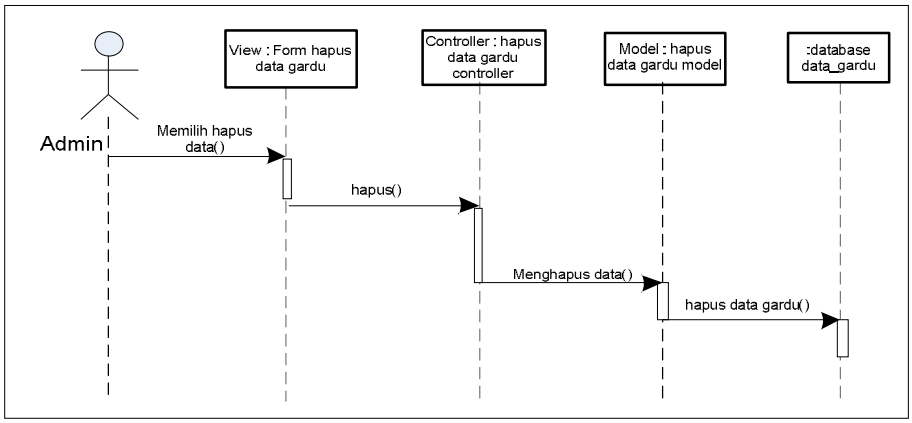
Sequence diagram edit data gardu menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses meng-*edit* data gardu. *Sequence diagram edit* data gardu dapat dilihat pada gambar 3.34.



Gambar 3.34 *Sequence Diagram Edit* Data Gardu

3.1.8.5.12. ***Sequence Diagram* Hapus Data Gardu**

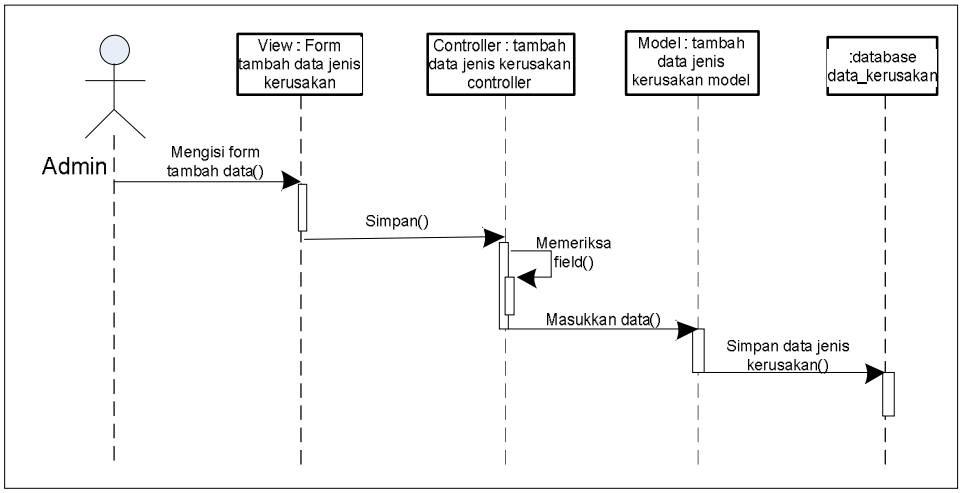
Sequence diagram hapus data gardu menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menghapus data gardu. *Sequence diagram* hapus data gardu dapat dilihat pada gambar 3.35.



Gambar 3.35 *Sequence Diagram* Hapus Data Gardu

3.1.8.5.13. ***Sequence Diagram* Tambah Data Jenis Kerusakan**

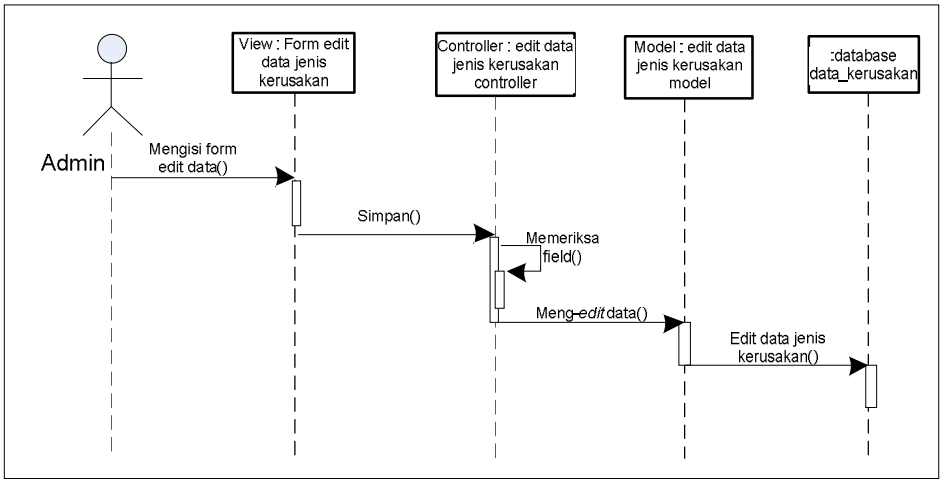
Sequence diagram tambah data jenis kerusakan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menambah data jenis kerusakan. *Sequence diagram* tambah data jenis kerusakan dapat dilihat pada gambar 3.36.



Gambar 3.36 *Sequence Diagram* Tambah Data Jenis Kerusakan

3.1.8.5.14. ***Sequence Diagram Edit* Data Jenis Kerusakan**

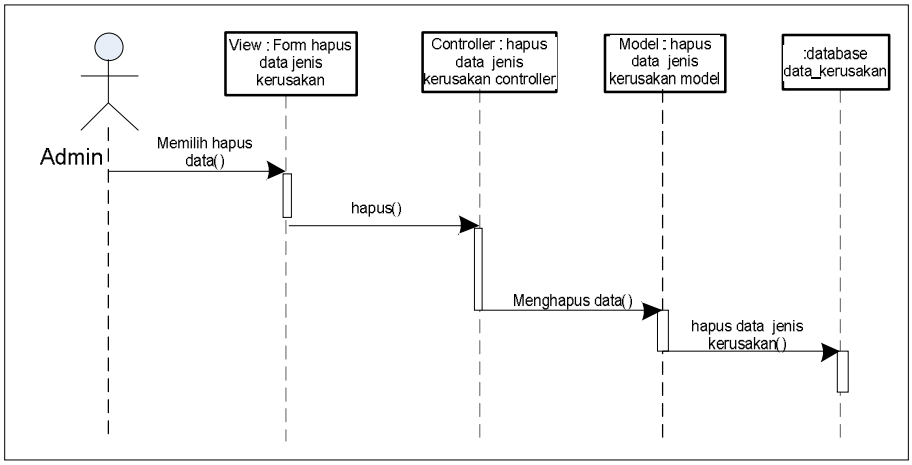
Sequence diagram edit data jenis kerusakan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses meng-*edit* data jenis kerusakan. *Sequence diagram edit* data jenis kerusakan dapat dilihat pada gambar 3.37.



Gambar 3.37 *Sequence Diagram Edit* Data Jenis Kerusakan

3.1.8.5.15. ***Sequence Diagram* Hapus Data Jenis Kerusakan**

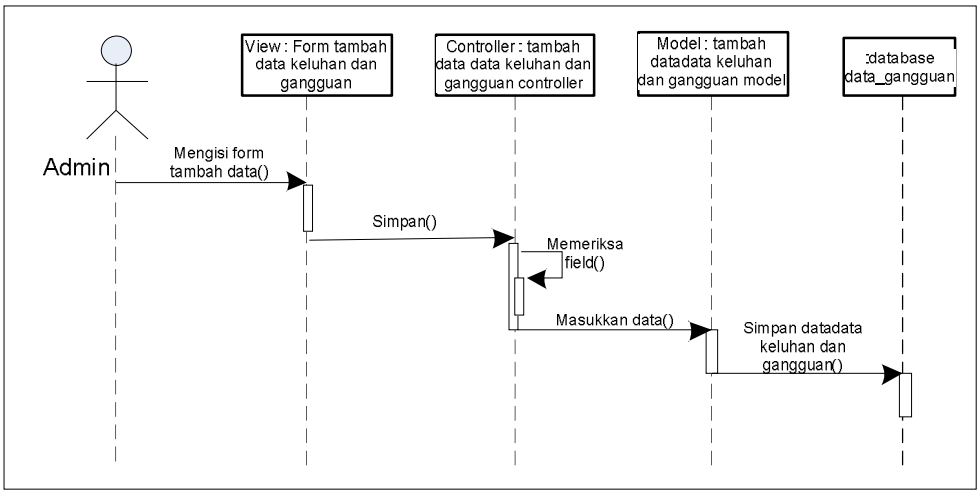
Sequence diagram hapus data jenis kerusakan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menghapus data jenis kerusakan. *Sequence diagram* hapus data jenis kerusakan dapat dilihat pada gambar 3.38.



Gambar 3.38 *Sequence Diagram* Hapus Data Jenis Kerusakan

3.1.8.5.16. ***Sequence Diagram* Tambah Data Keluhan dan Gangguan**

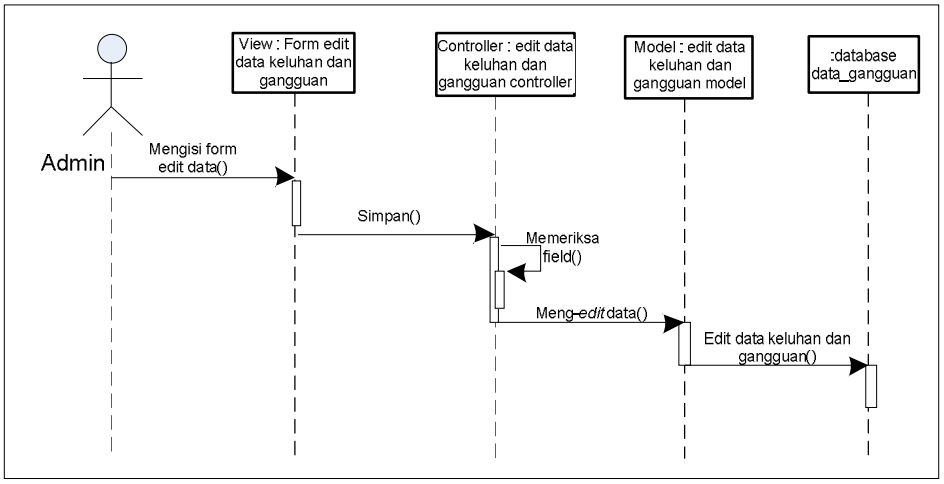
Sequence diagram tambah data keluhan dan gangguan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menambah data keluhan dan gangguan. *Sequence diagram* tambah data keluhan dan gangguan dapat dilihat pada gambar 3.39.



Gambar 3.39 *Sequence Diagram* Tambah Data Keluhan dan Gangguan

3.1.8.5.17. ***Sequence Diagram Edit* Data Keluhan dan Gangguan**

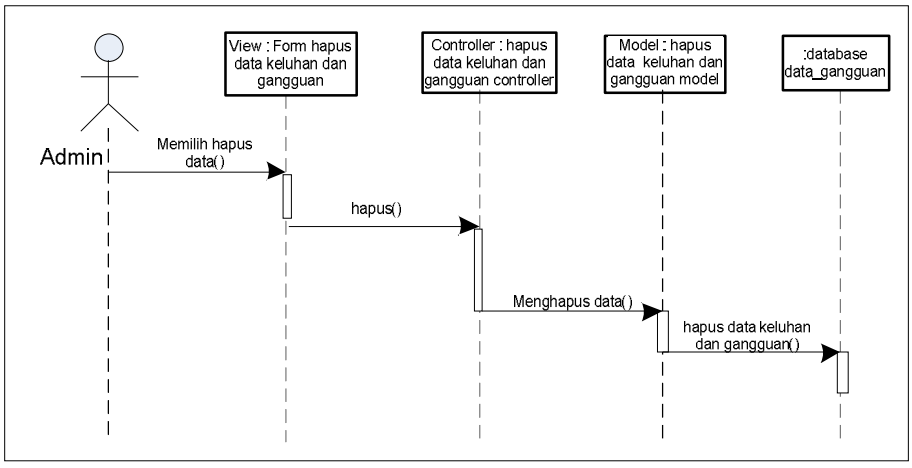
Sequence diagram edit data keluhan dan gangguan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses meng-*edit* data keluhan dan gangguan. *Sequence diagram edit* data keluhan dan gangguan dapat dilihat pada gambar 3.40.



Gambar 3.40 *Sequence Diagram Edit* Data Keluhan dan Gangguan

3.1.8.5.18. ***Sequence Diagram* Hapus Data Keluhan dan Gangguan**

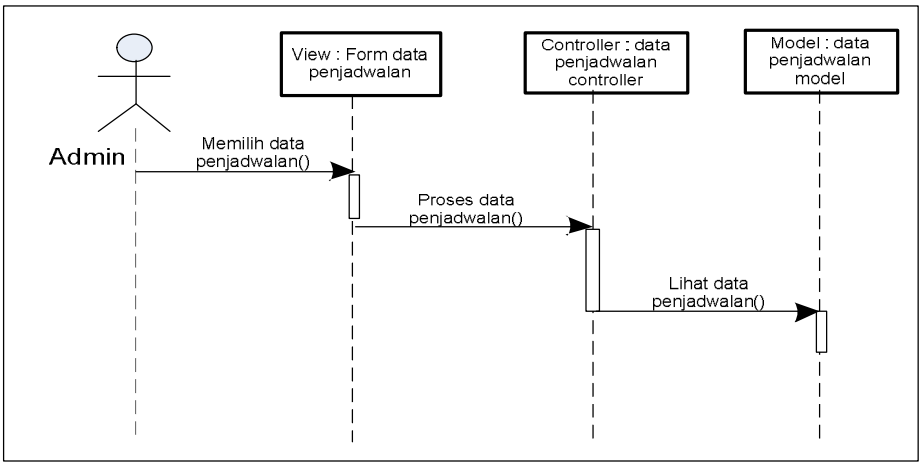
Sequence diagram hapus data keluhan dan gangguan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses menghapus data keluhan dan gangguan. *Sequence diagram* hapus data keluhan dan gangguan dapat dilihat pada gambar 3.41.



Gambar 3.41 *Sequence Diagram* Hapus Data Keluhan dan Gangguan

3.1.8.5.19. ***Sequence Diagram* Data Penjadwalan**

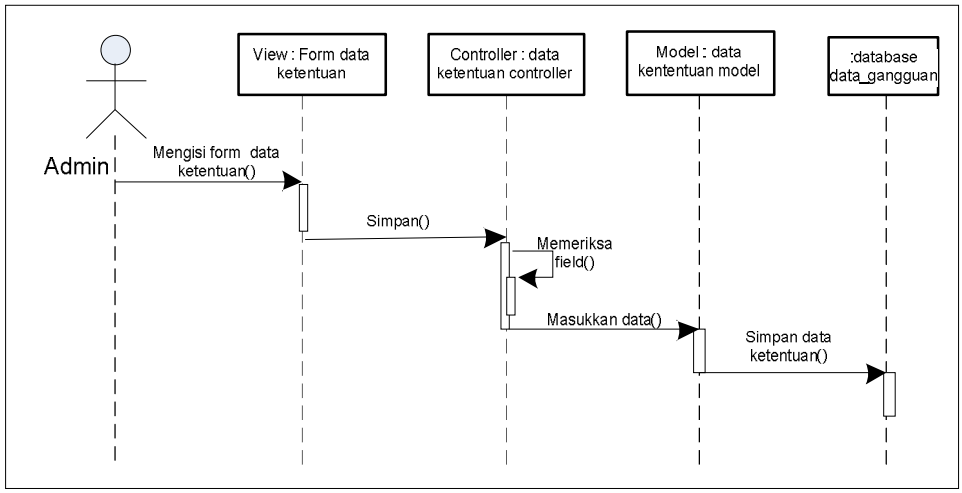
Sequence diagram data penjadwalan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses melihat data penjadwalan. *Sequence diagram* data penjadwalan dapat dilihat pada gambar 3.42.



Gambar 3.42 *Sequence Diagram* Data Penjadwalan-

3.1.8.5.20. ***Sequence Diagram* Data Ketentuan**

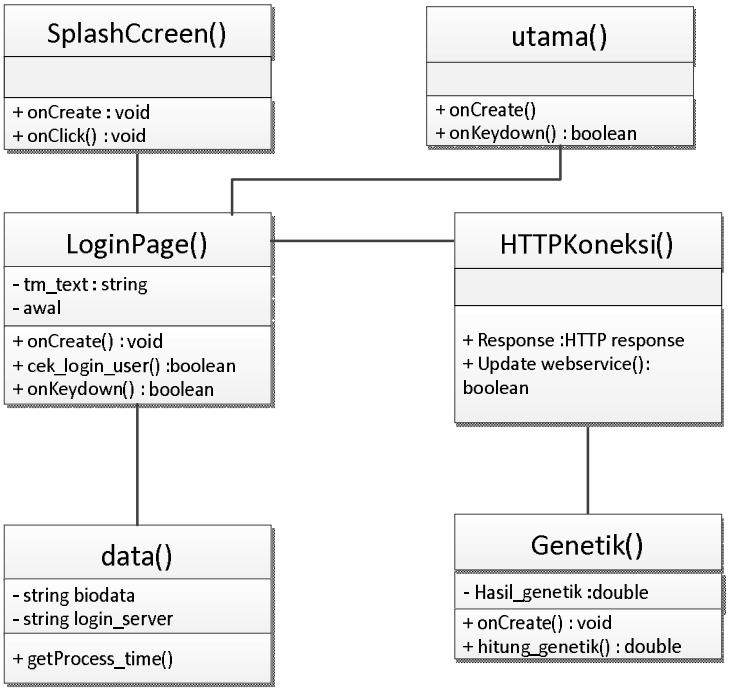
Sequence diagram data ketentuan menunjukkan rangkaian pesan yang dikirim antara objek admin dengan objek lainnya dalam proses modifikasi data ketentuan. *Sequence diagram* data ketentuan dapat dilihat pada gambar 3.43.



Gambar 3.43 *Sequence Diagram* Tambah Data Keluhan dan Gangguan

3.1.8.6. *Class Diagram*

Class Diagram menggambarkan struktur dan hubungan antar objek-objek yang ada pada sistem. Struktur itu meliputi atribut-atribut dan metode-metode yang ada pada masing-masing *class.* Hubungannya dari masing-masing *class* yang ada digambarkan seperti pada gambar 3.44.



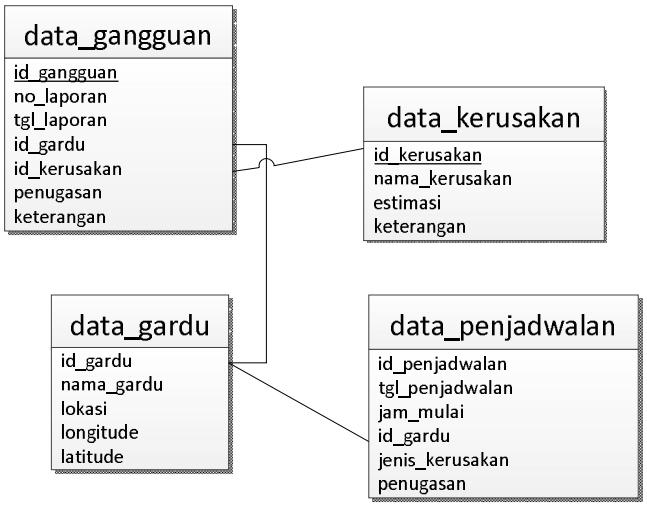
gambar 3.44 *class* diagram aplikasi

3.2. Perancangan Arsitektur

Perancangan arsitektur merupakan perancangan yang dibuat sebelum program aplikasi dibuat. Dengan perancangan arsitektur akan mempermudah dalam proses pembangunan perangkat lunak. Perancangan perangkat lunak ini mengikuti konsep yang telah dilakukan pada tahap analisis sistem. Perancangan arsitektur digambarkan sebagai proses multi-langkah dimana representasi struktur data, perancangan struktur menu, perancangan antar muka, perancangan Pesan, dan perancangan prosedural.

3.2.1. Skema Relasi

Skema relasi menggambarkan suatu hubungan antar tabel yang sudah ada dalam keadaan normal. Perancangan tabel relasi dalam membangun sebuah perangkat lunak aplikasi ini dapat dilihat pada gambar 3.45.



Gambar 3.45 Skema Relasi

3.2.2. Perancangan Data

Perancangan data merupakan tahapan untuk memetakan model konseptual ke model basis data yang akan digunakan. Perancangan data dari aplikasi ini merupakan perancangan struktur tabel dari tabel kategori. Struktur tabel kategori menggambarkan detail tabel yang berisi *field*, tipe data, panjang data, dan keterangan lainnya. Berikut ini gambaran struktur tabel perancangan data yang terlihat pada Tabel 3.25 sampai dengan tabel 3.31

Tabel 3.25 Perancangan Data Gangguan

Nama Field	Tipe Data	Keterangan
id_gangguan	BIGINT (20)	Primary key
no_laporan	VARCHAR(50)	
tgl_laporan	VARCHAR(50)	
nama_gardu	VARCHAR(100)	
jenis_kerusakan	VARCHAR(100)	

Nama Field	Tipe Data	Keterangan
penugasan	VARCHAR(100)	
keterangan	VARCHAR(255)	

Tabel 3.26 Perancangan Data Kerusakan

Nama Field	Tipe Data	Keterangan
id_kerusakan	BIGINT (20)	Primary key
nama_kerusakan	VARCHAR(100)	
estimasi	INT(11)	
keterangan	VARCHAR(255)	

Tabel 3.27 Perancangan Data Menu

Nama Field	Tipe Data	Keterangan
id_menu	TINYINT(3)	Primary key
parent_id	TINYINT(3)	
title	VARCHAR(100)	
url	VARCHAR(100)	
menu_order	TINYINT(3)	

Tabel 3.28 Perancangan Data Gardu

Nama Field	Tipe Data	Keterangan
id_gardu	BIGINT(20)	Primary key
nama_gardu	VARCHAR(50)	
lokasi	VARCHAR(255)	
longitude	DOUBLE	
latitude	DOUBLE	

Tabel 3.29 Perancangan Data Penjadwalan

Nama Field	Tipe Data	Keterangan
id_penjadwalan	BIGINT(20)	Primary key
tgl_penjadwalan	VARCHAR(50)	
jam_mulai	VARCHAR(50)	
nama_gardu	VARCHAR(100)	
jenis_kerusakan	VARCHAR(100)	

Nama Field	Tipe Data	Keterangan
penugasan	VARCHAR(100)	

Tabel 3.30 Perancangan Data Pengguna

Nama Field	Tipe Data	Keterangan
id_pengguna	BIGINT(20)	Primary key
username	VARCHAR(50)	
password	VARCHAR(50)	
nama	VARCHAR(255)	
level	VARCHAR(30)	
nama_profil	BIGINT(20)	
aktif	TINYINT(3)	

Tabel 3.31 Perancangan Data Ketentuan

Nama Field	Tipe Data	Keterangan
Id_ketentuan	BIGINT(20)	Primary key
jam_kerja	DECIMAL	
jth_liter	INT(11)	
rata_asumsi	DOUBLE	
jlh_istirahat	DECIMAL	

3.2.3. Perancangan Struktur Menu

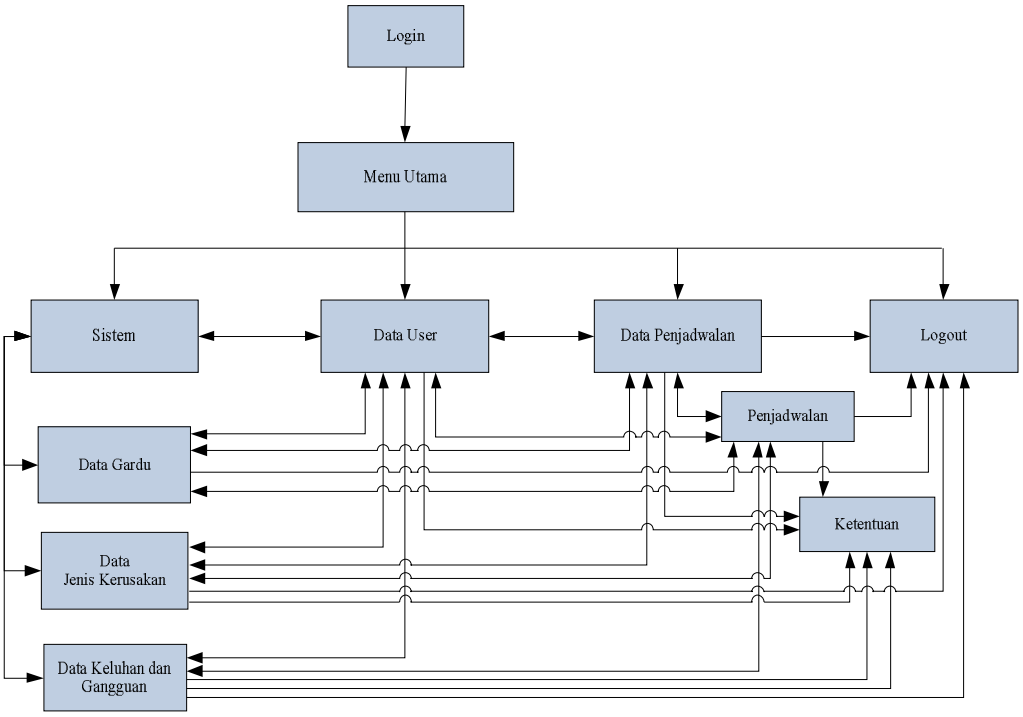
Dalam perancangan sebuah aplikasi dibutuhkan struktur menu yang berisikan menu dan submenu yang berfungsi untuk memudahkan admin dalam menggunakan aplikasi tersebut. Berikut ini digambarkan mengenai stuktur menu dalam aplikasi ini terbagi dua, yaitu :

1. **Perancangan Struktur Menu Aplikasi Yantek PLN**
2. **Perancangan Struktur Menu Pada *Platform* Android**

3.2.3.1. Perancangan Struktur Menu Aplikasi Yantek PLN

Berikut adalah perancangan struktur menu aplikasi yang akan dibangun berbasis web dengan *Macromedia Dreamweaver CS3* sebagai *editor* PHP dan MYSQL sebagai *database*nya, yang mana menu-menu tersebut memiliki fungsi

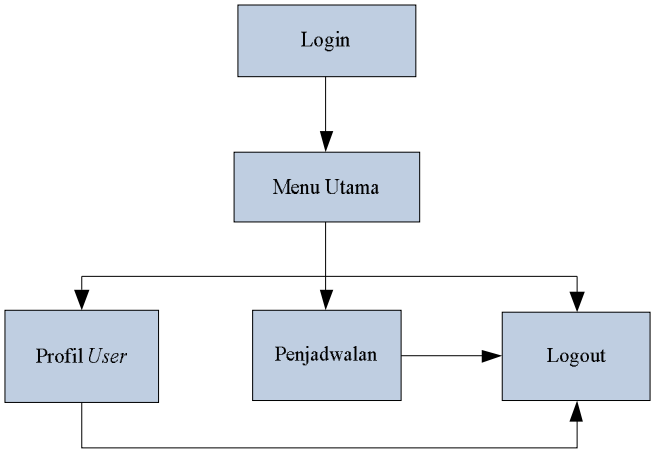
dan peranan masing-masing dalam mengelola dan *monitoring* kinerja yang ada di PT. PLN (Persero) Rayon Bandung Timur.



Gambar 3.46 Struktur Menu Aplikasi Yantek PLN

3.2.3.2. Perancangan Struktur Menu Aplikasi Android

Berikut adalah perancangan struktur menu aplikasi yang dibangun di *platform* Android sebagai menu pilihan yang bisa di gunakan oleh regu YANTEK dalam menjalankan tugas dan tanggung jawab menangani keluhan dan gangguan pelanggan di PT. PLN (persero) Rayon Bandung Timur.



Gambar 3.47 Struktur Menu Aplikasi Android

3.2.4. Perancangan Antar Muka

Dalam membuat aplikasi dibutuhkan tampilan yang sangat menarik, dimana dalam perancangan antar muka ini akan dituangkan beberapa imajinasi dalam merancang tampilan program dan tampilan pesan yang nantinya akan di pakai untuk membuat aplikasi sesungguhnya. Perancangan antar muka ini terdiri dari perancangan tampilan program, dan perancangan tampilan pesan. Dalam perancangan ini terdapat dua perancangan antar muka yaitu:

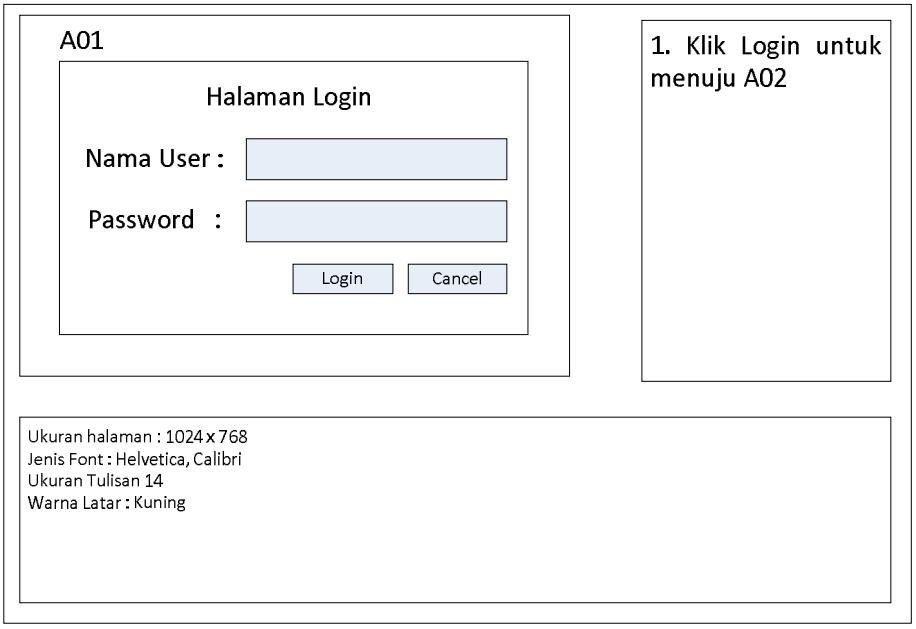
1. **Perancangan Antar Muka Aplikasi Yantek PLN**
2. **Perancangan Antar Muka Aplikasi Android**

3.2.4.1. Perancangan Antar Muka Yantek PLN

Berikut adalah perancangan antar muka Aplikasi Yantek PLN yang akan dibangun sesuai dengan kebutuhan, perancangan ini memiliki fungsi dan peranan masing-masing dalam mengelola dan *monitoring* kinerja yang ada di PT. PLN (Persero) Rayon Bandung Timur.

1. **Perancangan Halaman Login**

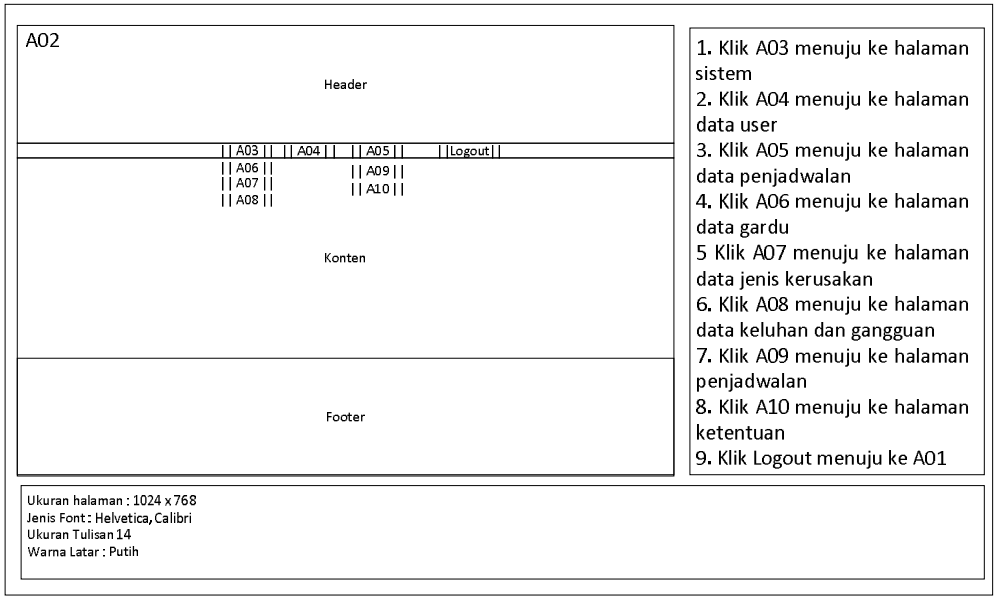
   Gambaran umum dari perancangan antarmuka halaman login pada aplikasi yang diakses dapat dilihat pada gambar 3.48.



Gambar 3.48 Tampilan Halaman Login

2. Perancangan Halaman Utama

Gambaran umum dari perancangan antarmuka halaman utama pada aplikasi yang diakses setelah proses *login* dapat dilihat pada gambar 3.49.



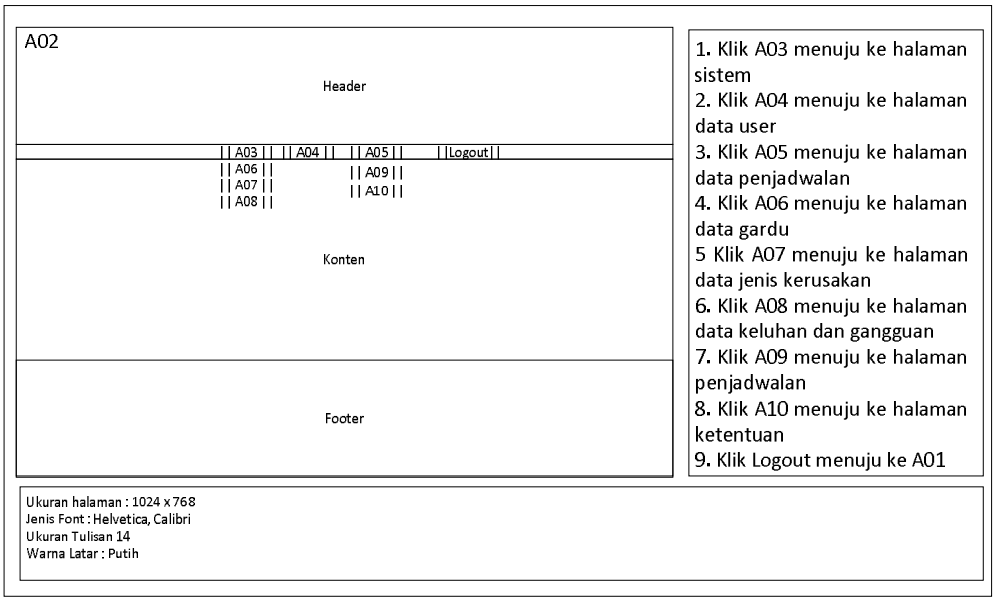
Gambar 3.49 Tampilan Halaman Utama

Tabel 3.32 Keterangan Perancangan Halaman Utama

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan

3. Perancangan Halaman Sistem

Gambaran umum dari perancangan antarmuka halaman sistem pada aplikasi yang diakses dapat dilihat pada gambar 3.50.



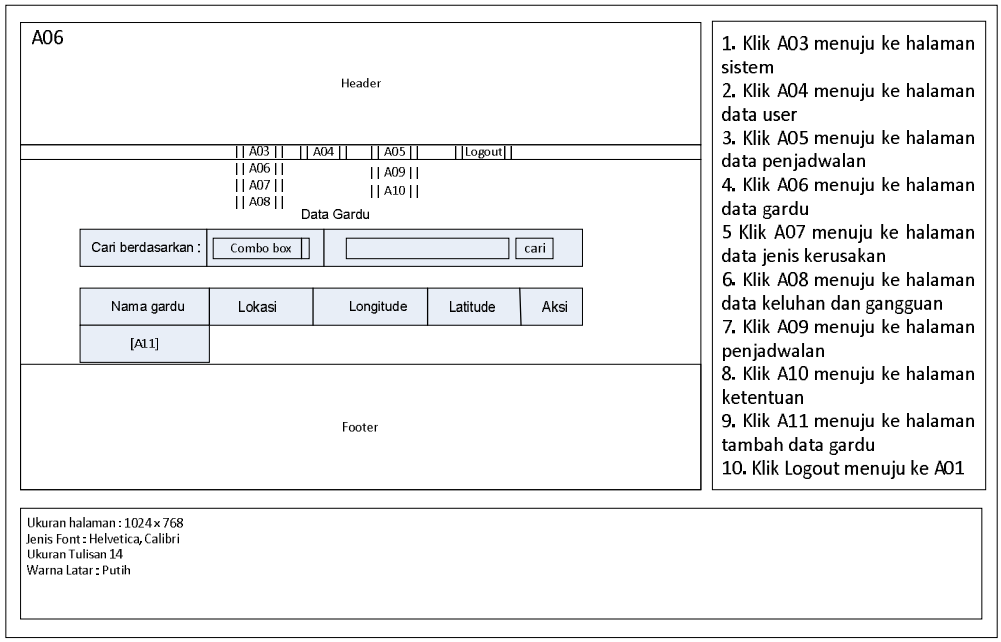
Gambar 3.50 Tampilan Halaman Sistem

Tabel 3.33 Keterangan Perancangan Halaman Sistem

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan

4. Perancangan Halaman Data Gardu

Gambaran umum dari perancangan antarmuka halaman data gardu pada aplikasi yang diakses dapat dilihat pada gambar 3.51.



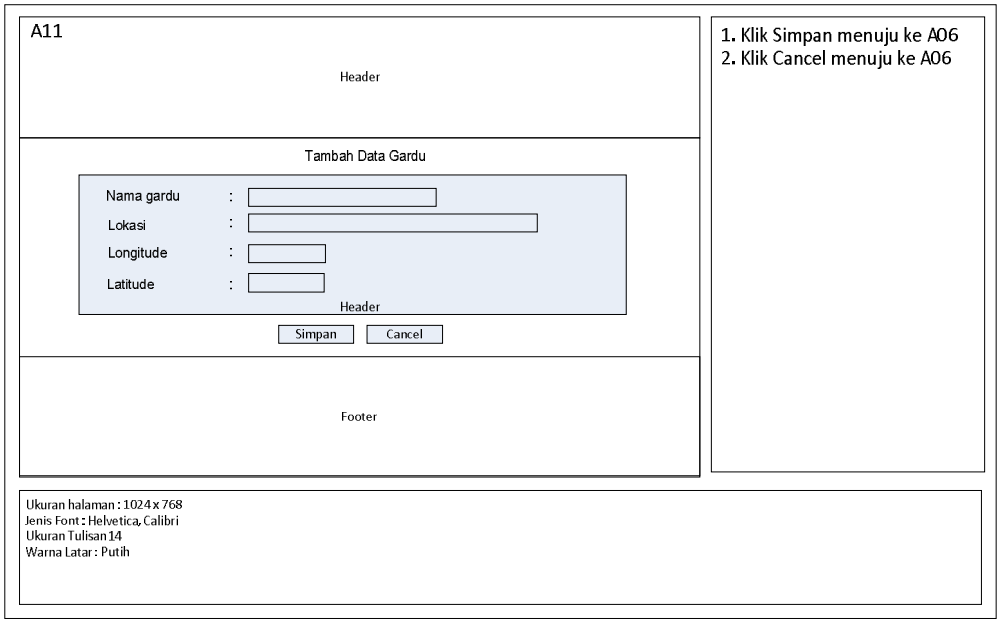
Gambar 3.51 Tampilan Halaman Data Gardu

Tabel 3.34 Keterangan Perancangan Halaman Data Gardu

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan
A11	Halaman Tambah Data Gardu

5. Perancangan Halaman Tambah Data Gardu

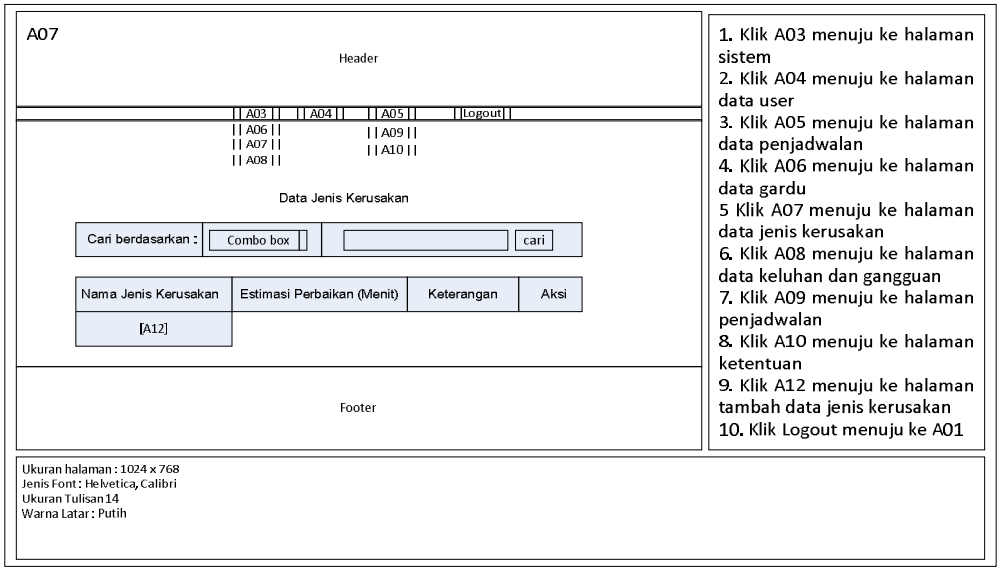
Gambaran umum dari perancangan antarmuka halaman tambah data gardu pada aplikasi yang diakses dapat dilihat pada gambar 3.52.



Gambar 3.52 Tampilan Halaman Tambah Data Gardu

6. Perancangan Halaman Data Jenis Kerusakan

Gambaran umum dari perancangan antarmuka halaman data jenis kerusakan pada aplikasi yang diakses dapat dilihat pada gambar 3.53.



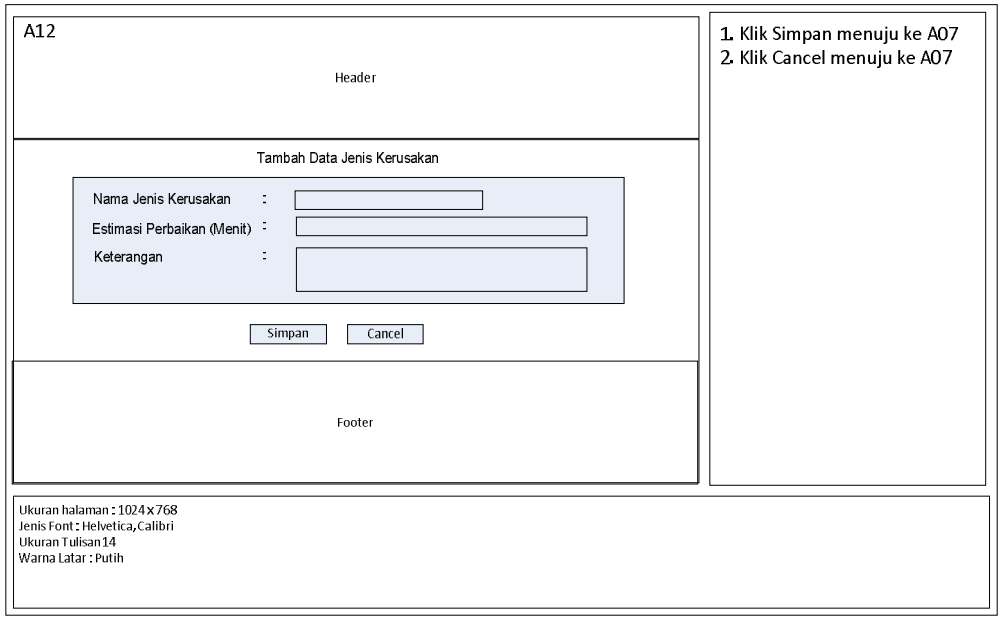
Gambar 3.53 Tampilan Halaman Data Jenis Kerusakan

Tabel 3.35 Keterangan Perancangan Halaman Data Jenis Kerusakan

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan
A12	Halaman Tambah Data Jenis Kerusakan

7. Perancangan Halaman Tambah Data Jenis Kerusakan

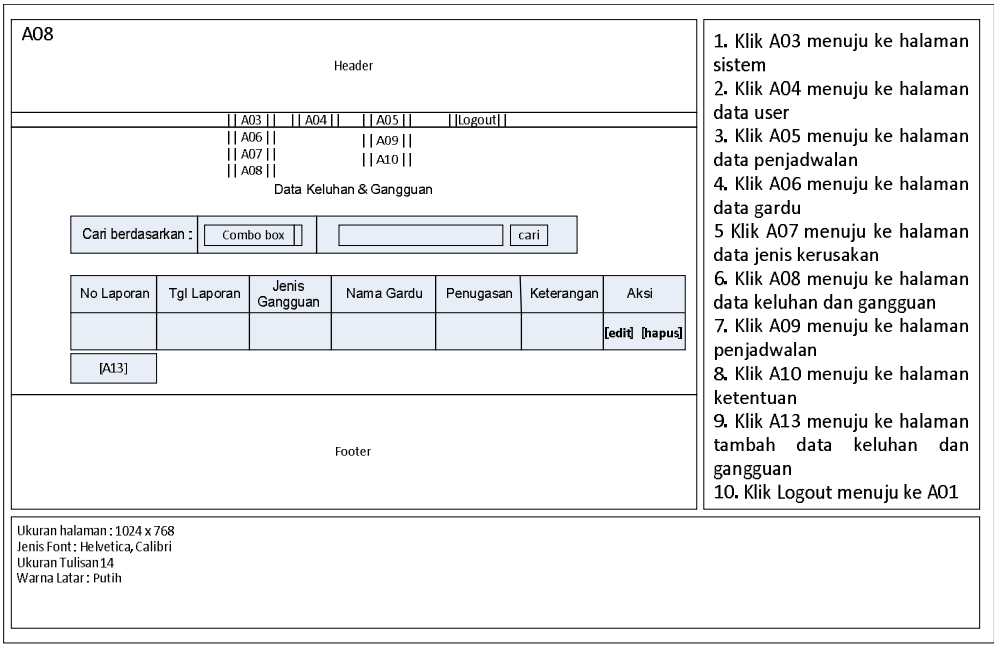
Gambaran umum dari perancangan antarmuka halaman tambah data jenis kerusakan pada aplikasi yang diakses dapat dilihat pada gambar 3.54.



Gambar 3.54 Tampilan Halaman Tambah Data Jenis Kerusakan

8. Perancangan Halaman Data Keluhan & Gangguan

Gambaran umum dari perancangan antarmuka halaman data keluhan & gangguan pada aplikasi yang diakses dapat dilihat pada gambar 3.55.



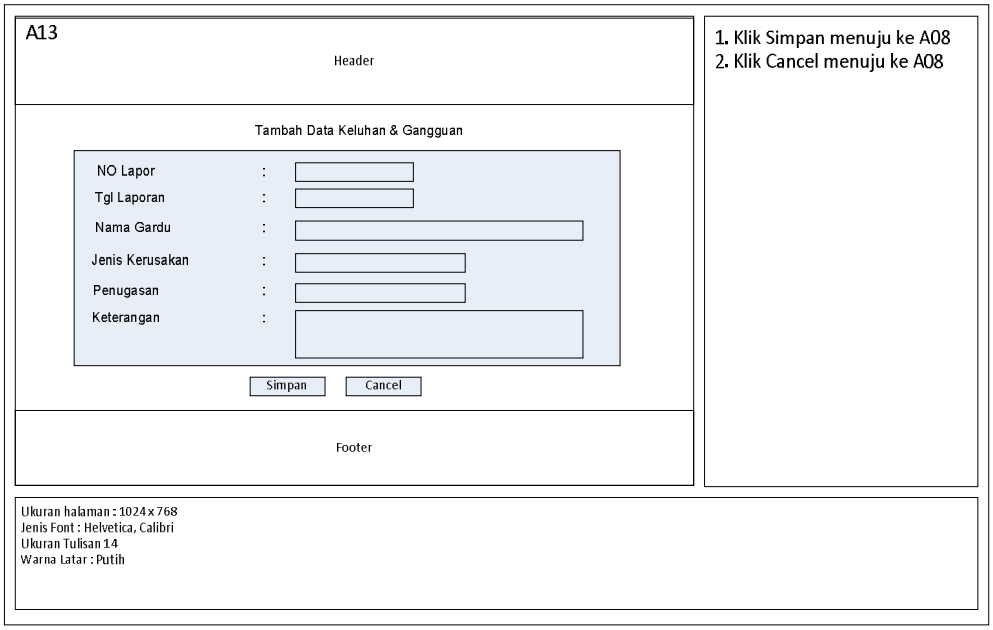
Gambar 3.55 Tampilan Halaman Data Keluhan & Gangguan

Tabel 3.36 Keterangan Perancangan Halaman Data Keluhan & Gangguan

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan
A13	Halaman Tambah Data Keluhan & Gangguan

9. Perancangan Halaman Tambah Data Keluhan & Gangguan

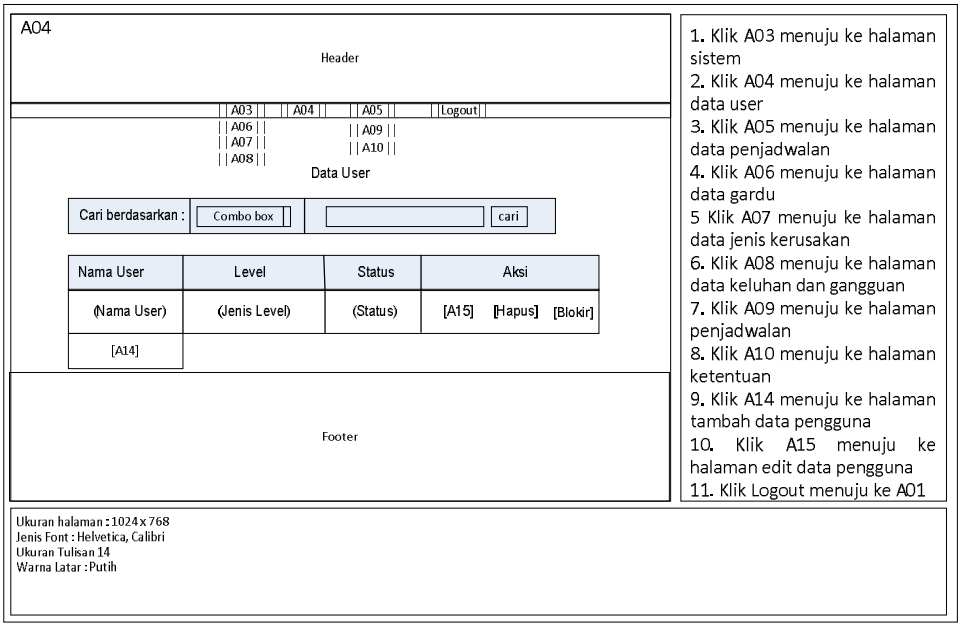
Gambaran umum dari perancangan antarmuka halaman tambah data keluhan dan gangguan pada aplikasi yang diakses dapat dilihat pada gambar 3.56.



Gambar 3.56 Tampilan Halaman Tambah Data Keluhan & Gangguan

10. Perancangan Halaman Data *User*

Gambaran umum dari perancangan antarmuka halaman data *User* pada aplikasi yang diakses dapat dilihat pada gambar 3.57.



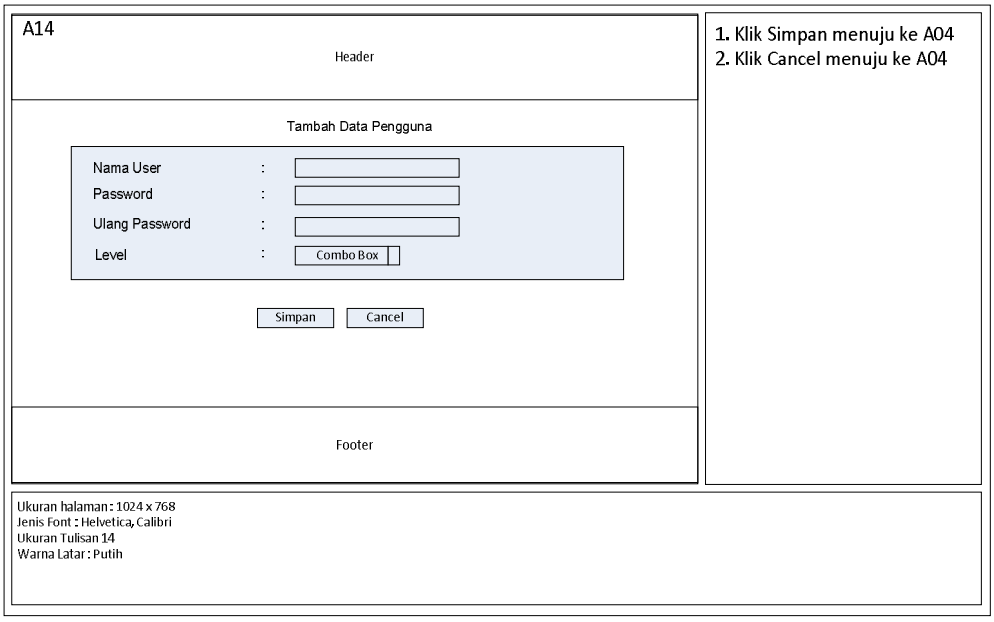
Gambar 3.57 Tampilan Halaman Data *User*

Tabel 3.37 Keterangan Perancangan Halaman Data *User*

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan
A14	Halaman Tambah Data Pengguna
A15	Halaman Edit Data Pengguna

11. Perancangan Halaman Tambah Data Pengguna

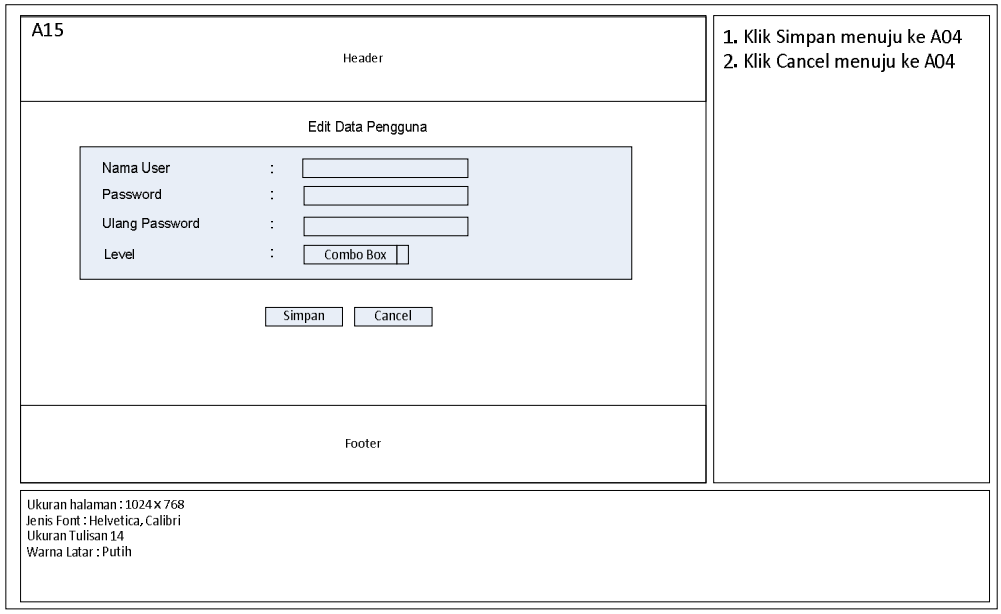
Gambaran umum dari perancangan antarmuka halaman tambah data pengguna pada aplikasi yang diakses dapat dilihat pada gambar 3.58.



Gambar 3.58 Tampilan Halaman Tambah Data Pengguna

12. Perancangan Halaman Edit Data Pengguna

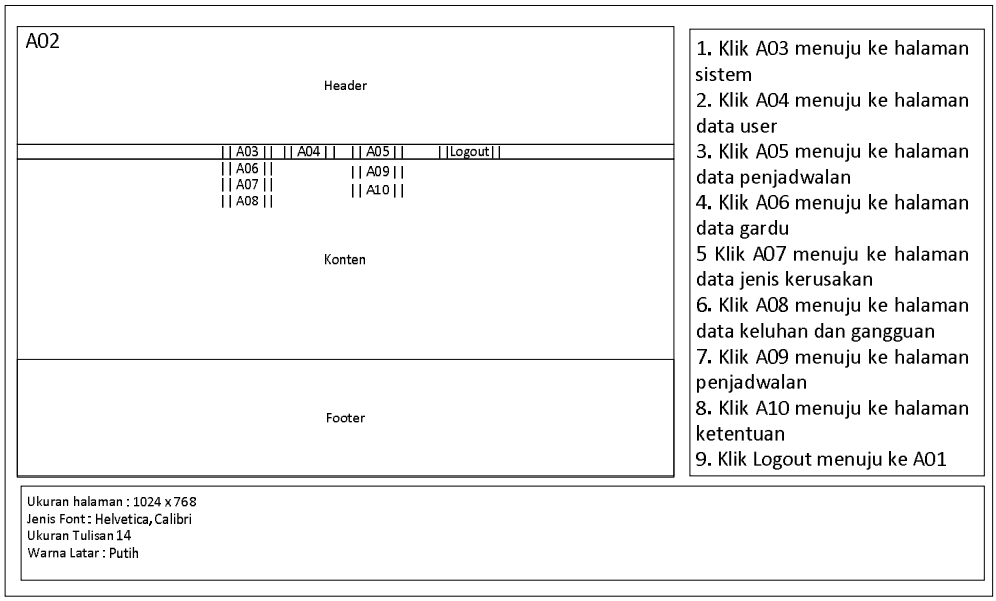
Gambaran umum dari perancangan antarmuka halaman edit data pengguan pada aplikasi yang diakses dapat dilihat pada gambar 3.59.



Gambar 3.59 Tampilan Halaman Tambah Data Pengguna

13. Perancangan Halaman Data Penjadwalan

Gambaran umum dari perancangan antarmuka halaman data penjadwalan pada aplikasi yang diakses dapat dilihat pada gambar 3.60.



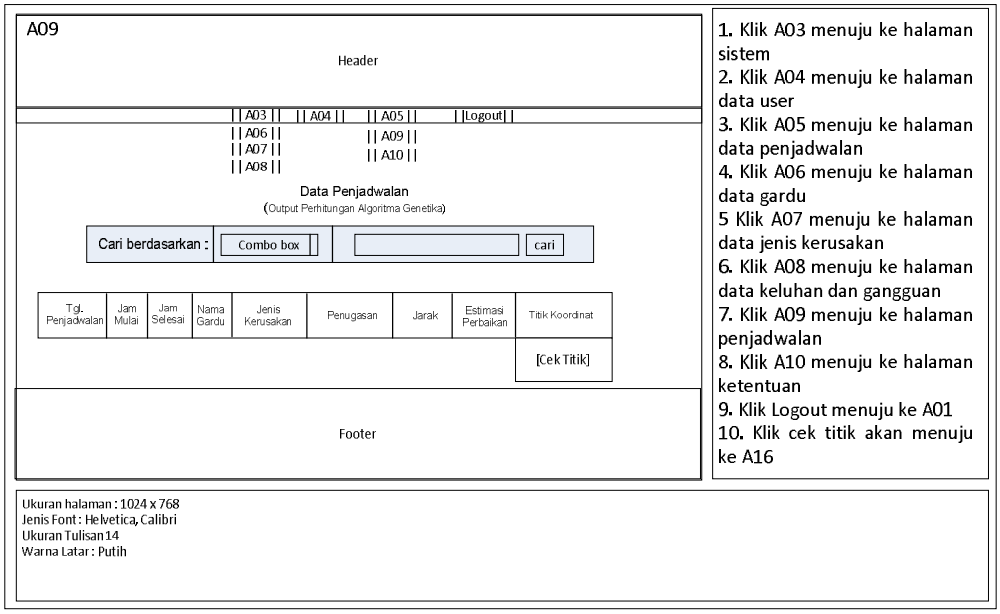
Gambar 3.60 Tampilan Halaman Data Penjadwalan

Tabel 3.38 Keterangan Perancangan Halaman Data Penjadwalan

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan

14. Perancangan Halaman Penjadwalan

Gambaran umum dari perancangan antarmuka halaman penjadwalan pada aplikasi yang diakses dapat dilihat pada gambar 3.61.



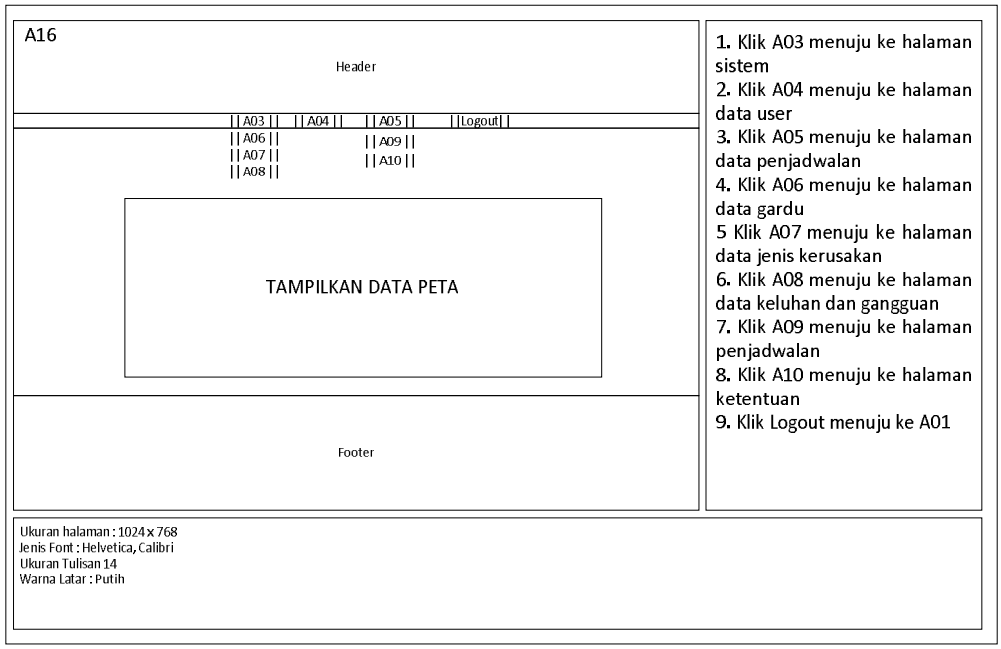
Gambar 3.61 Tampilan Halaman Data Penjadwalan

Tabel 3.39 Keterangan Perancangan Halaman Penjadwalan

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan

15. Perancangan Halaman Peta

Gambaran umum dari perancangan antarmuka halaman Peta pada aplikasi yang diakses dapat dilihat pada gambar 3.62.



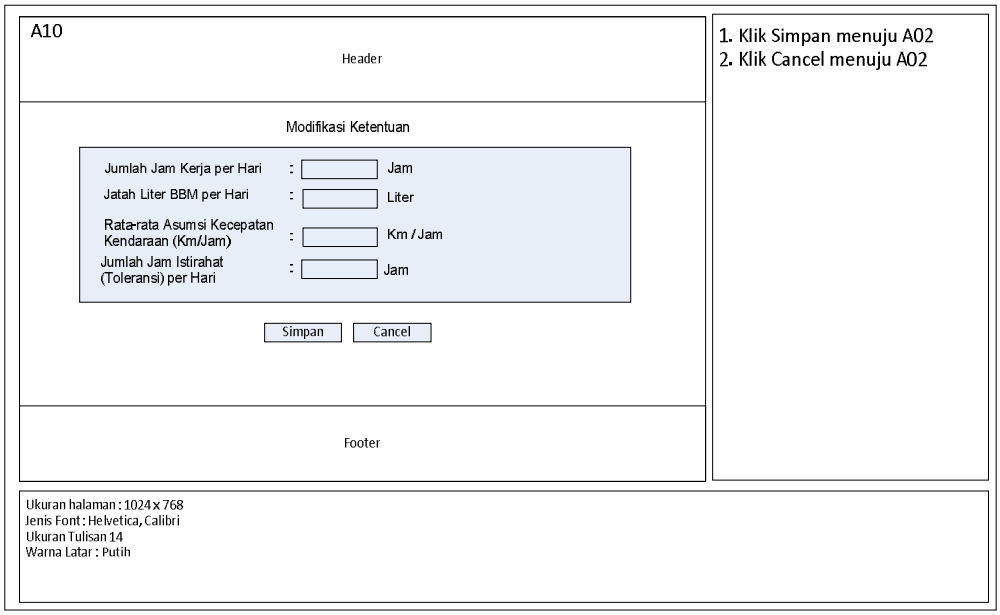
Gambar 3.62 Tampilan Halaman Data Peta

Tabel 3.40 Keterangan Perancangan Halaman Peta

Kode	Keterangan
A03	Halaman sistem
A04	Halaman data *user*
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan & gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan

16. Perancangan Halaman Ketentuan

Gambaran umum dari perancangan antarmuka halaman ketentuan pada aplikasi yang diakses dapat dilihat pada gambar 3.63.



Gambar 3.63 Tampilan Halaman Ketentuan

3.2.4.2. Perancangan Antar Muka Aplikasi Android

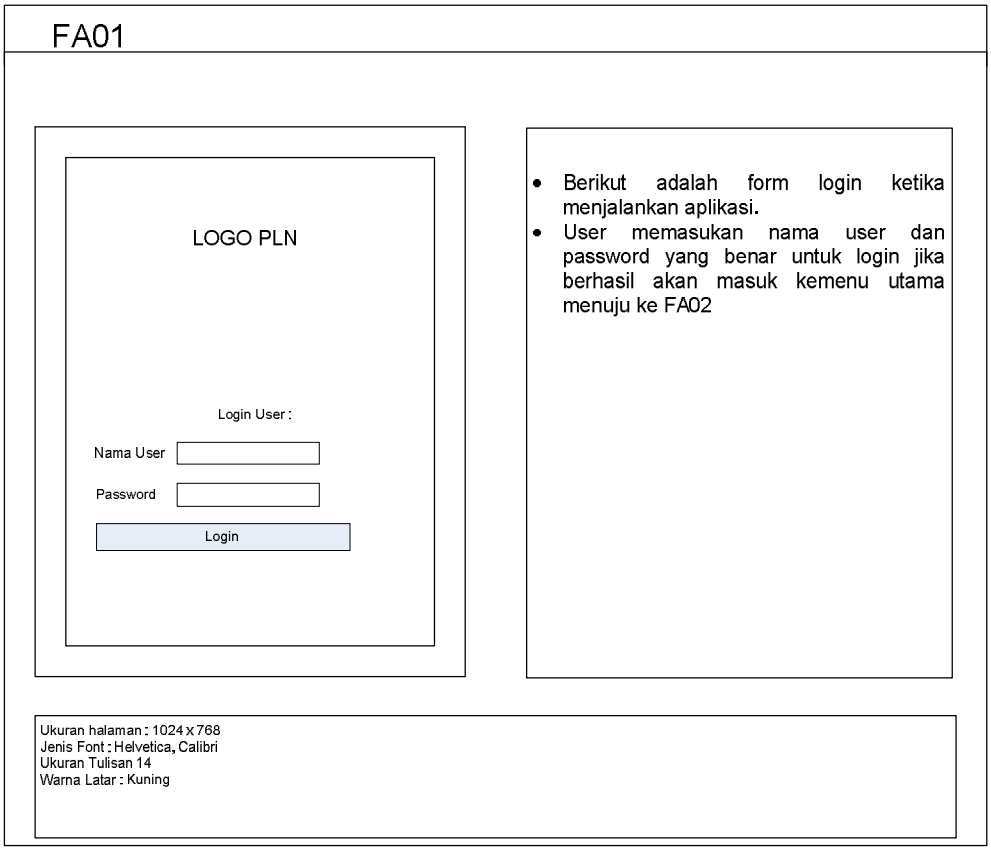
Berikut adalah perancangan antar muka aplikasi Android yang akan dibangun sesuai dengan kebutuhan, perancangan ini memiliki fungsi dan peranan masing-masing dalam mendukung kinerja yang ada di PT. PLN (Persero) Rayon Bandung Timur.

Perancangan Tampilan

Perancangan *form* mendeskripsikan rencana tampilan dari setiap *form* yang akan digunakan pada tampilan aplikasi *mobile* Android Yantek PLN. Adapun gambaran dari perancangan antarmuka aplikasi *mobile* Yantek PLN adalah sebagai berikut:

1. **Perancangan Menu *Login***

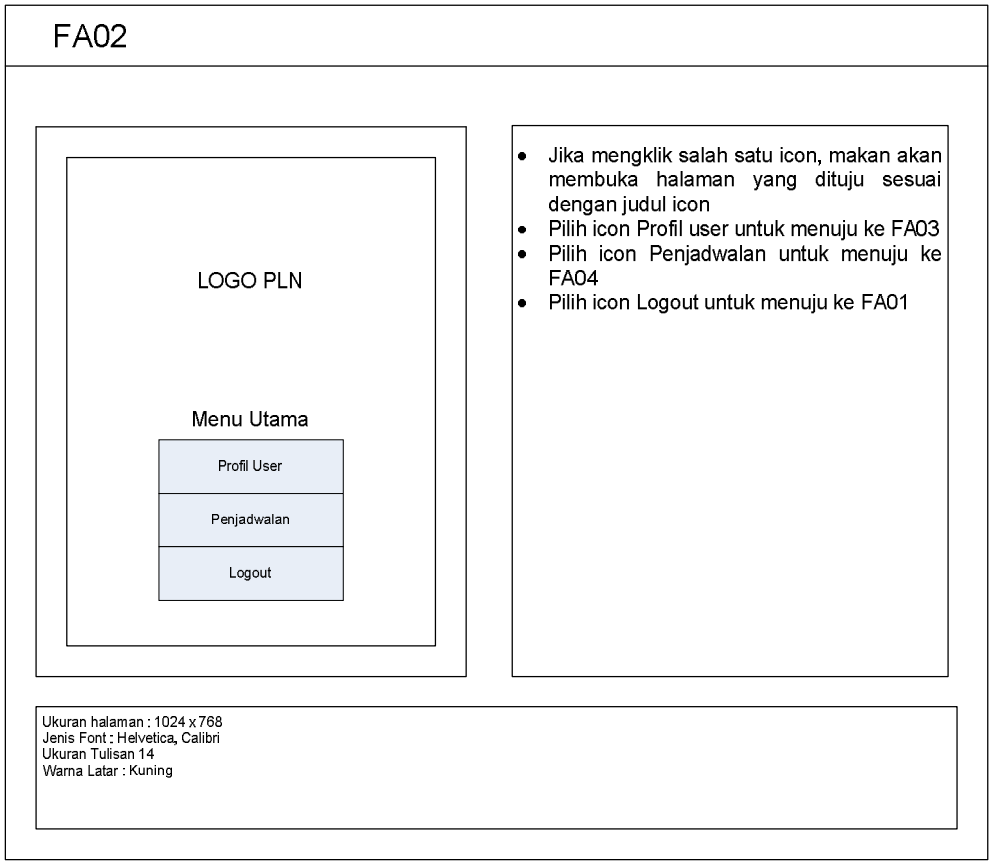
Gambaran umum dari perancangan antarmuka menu *login* pada aplikasi *mobile* Android Yantek PLN dapat dilihat pada gambar 3.64.



Gambar 3.64 Tampilan Menu *Login*

2. Perancangan Menu Utama

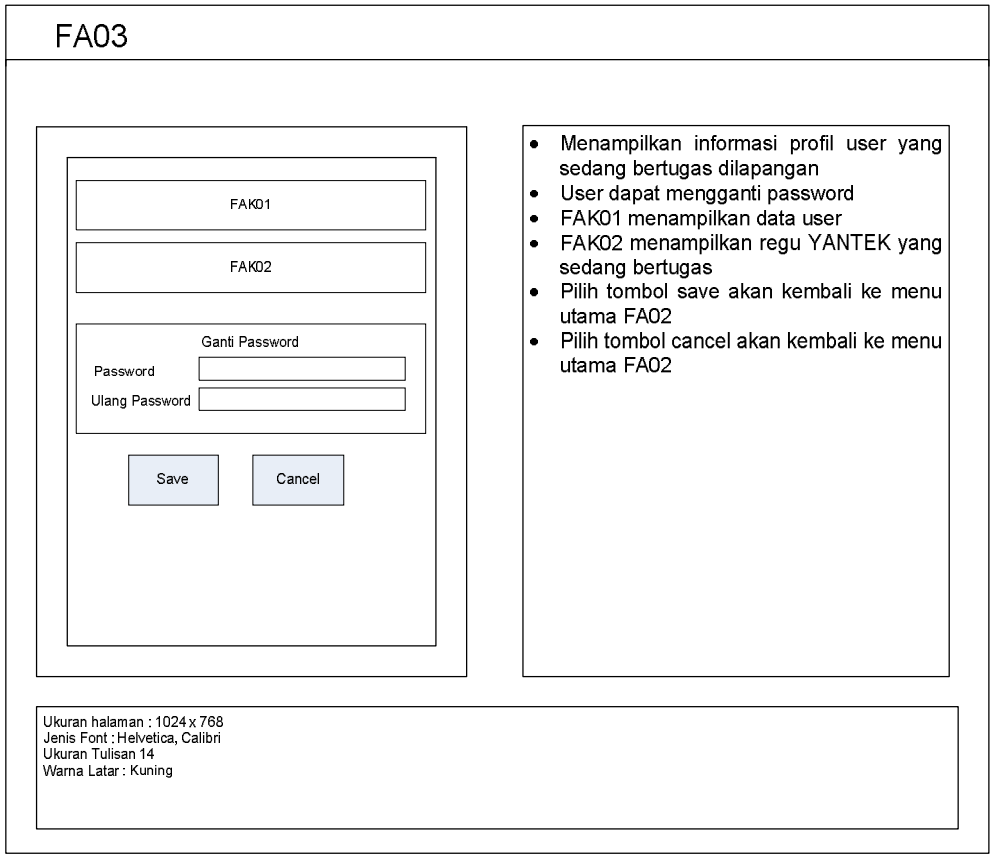
Gambaran umum dari perancangan antarmuka menu utama pada aplikasi *mobile* Android Yantek PLN dapat dilihat pada gambar 3.65.



Gambar 3.65 Tampilan Menu Utama

3. Perancangan Menu Profil User

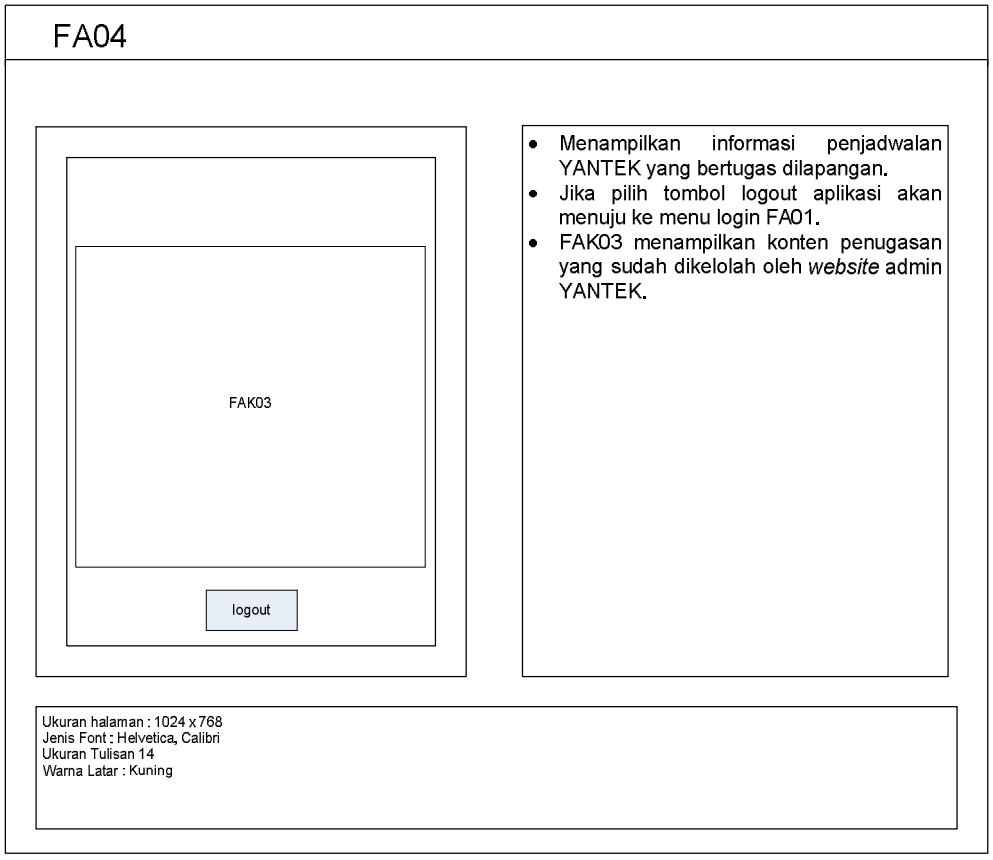
Gambaran umum dari perancangan antarmuka menu profil *user* pada aplikasi *mobile* Android Yantek PLN dapat dilihat pada gambar 3.66.



Gambar 3.66 Tampilan Menu Profil *User*

4. Perancangan Menu Penjadwalan

Gambaran umum dari perancangan antarmuka menu penjadwalan pada aplikasi *mobile* Android Yantek PLN dapat dilihat pada gambar 3.67.



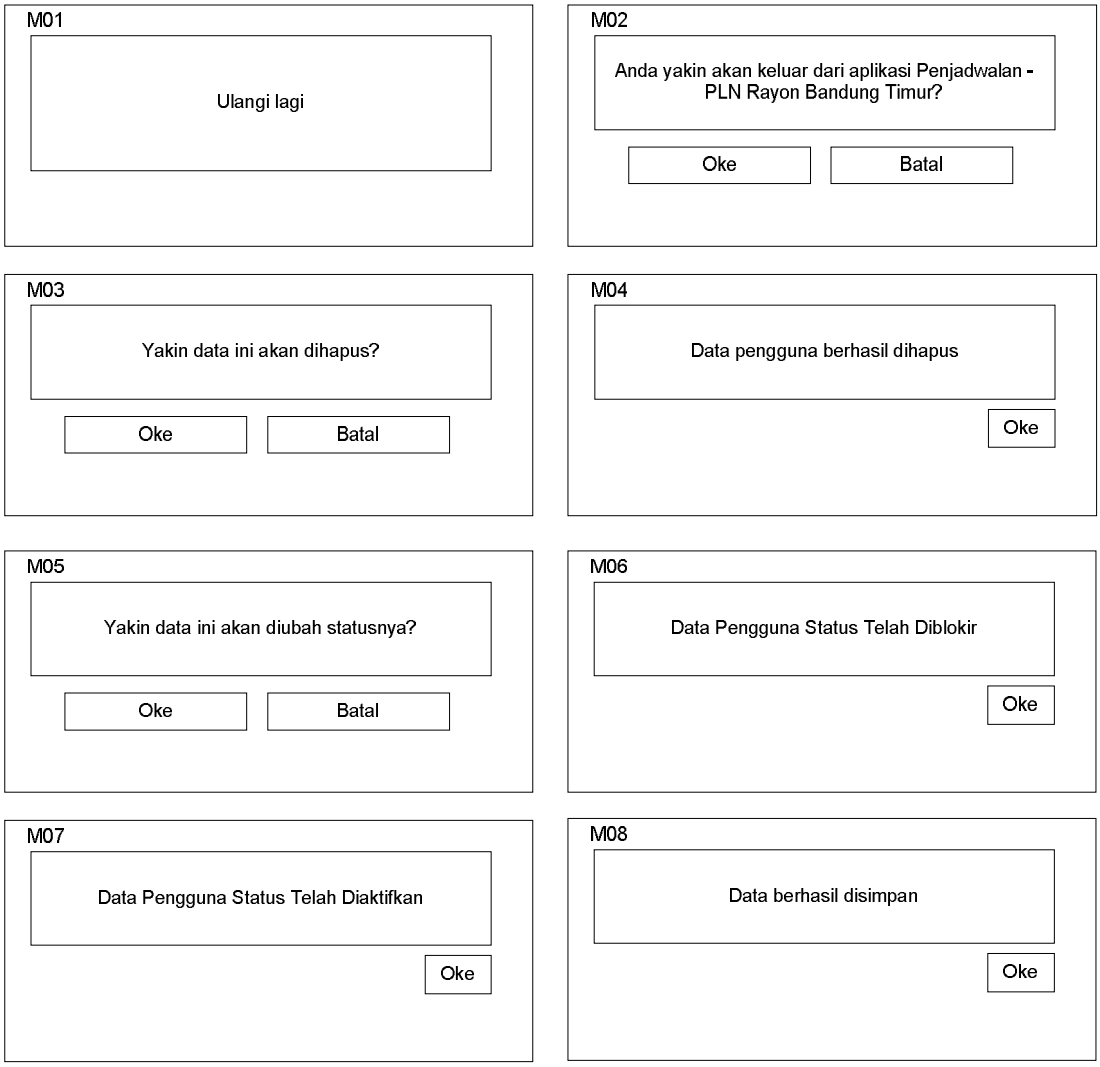
Gambar 3.67 Tampilan Menu Penjadwalan

3.2.5. Perancangan Pesan

Perancangan pesan disini adalah pesan yang akan tampil pada tiap aplikasi ketika aplikasi tersebut dijalankan sesuai pilihan yang ada, diantaranya adalah :

1. Perancangan Pesan pada Aplikasi Yantek PLN

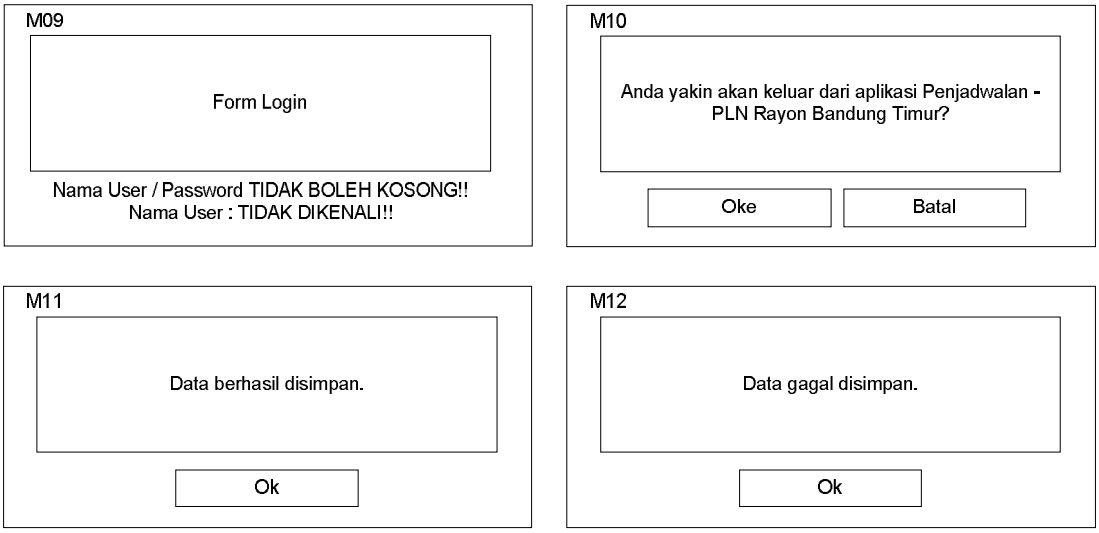
Perancangan pesan mendeskripsikan rencana tampilan dari setiap pesan yang digunakan pada Aplikasi Yantek PLN yang diakses. Berikut ini gambaran dari perancangan pesan pada Aplikasi Yantek PLN ini dapat dilihat pada gambar 3.68.



Gambar 3.68 Perancangan Pesan Aplikasi Yantek PLN

2. Perancangan Pesan pada Aplikasi Android

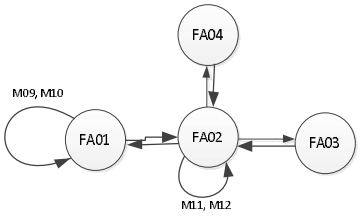
Perancangan pesan mendeskripsikan rencana tampilan dari setiap pesan yang digunakan pada aplikasi Android. Berikut ini gambaran dari perancangan pesan pada aplikasi ini dapat dilihat pada gambar 3.69.



Gambar 3.69 Perancangan Pesan Aplikasi Android

3.2.6. Jaringan Semantik

Jaringan semantik merupakan gambaran pengetahuan grafis yang mana menunjukkan hubungan antar objek (lingkaran) dan grafis yang menggambarkan formasi antar objek tersebut. Adapun jaringan semantik untuk *mobile* pada aplikasi ini dapat dilihat pada gambar 3.70 dan untuk Website pada aplikasi ini dapat dilihat pada gambar 3.71



Gambar 3.70 Jaringan Semantik Mobile aplikasi

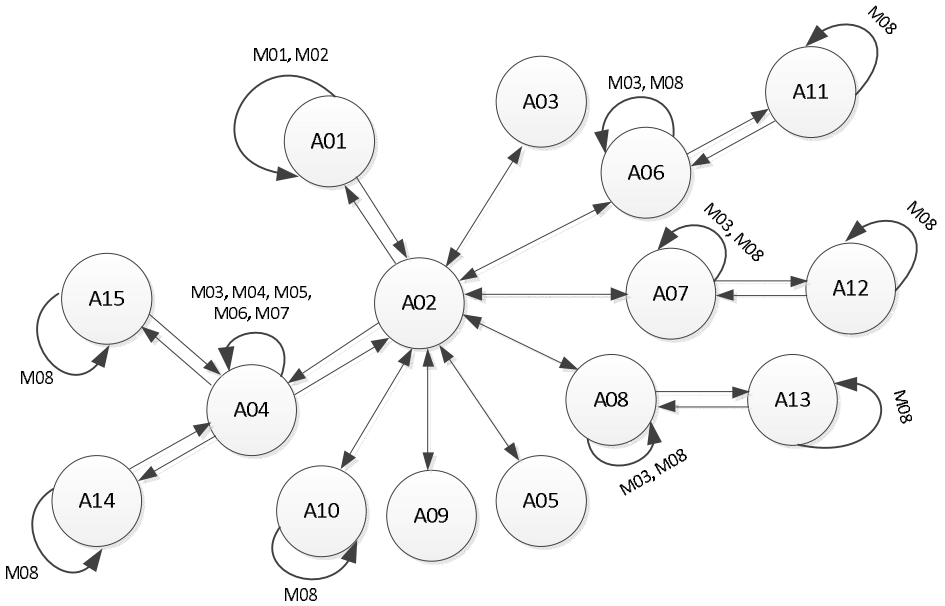
Keterangan :

FA01 Halaman Login

FA02 Halaman menu utama

FA03 Halaman profil user

FA04 Halaman Penjadwalan



Gambar 3.71 Jaringan Semantik untuk Aplikasi Yantek PLN

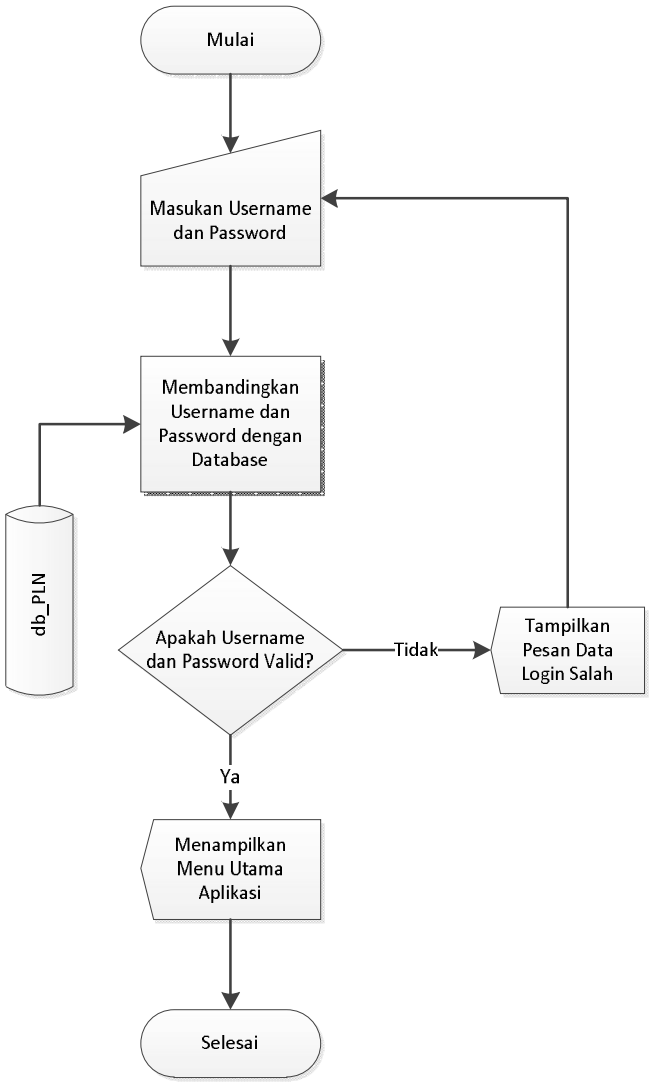
Keterangan :

A01	Halaman login
A02	Halaman utama *website*
A03	Halaman sistem
A04	Halaman data user
A05	Halaman data penjadwalan
A06	Halaman data gardu
A07	Halaman data jenis kerusakan
A08	Halaman data keluhan dan gangguan
A09	Halaman penjadwalan
A10	Halaman ketentuan
A11	Halaman tambah data gardu
A12	Halaman tambah data jenis kerusakan
A13	Halaman tambah data keluhan dan gangguan
A14	Halaman tambah data pengguna
A15	Halaman edit data pengguna

3.2.7. Perancangan *Method*

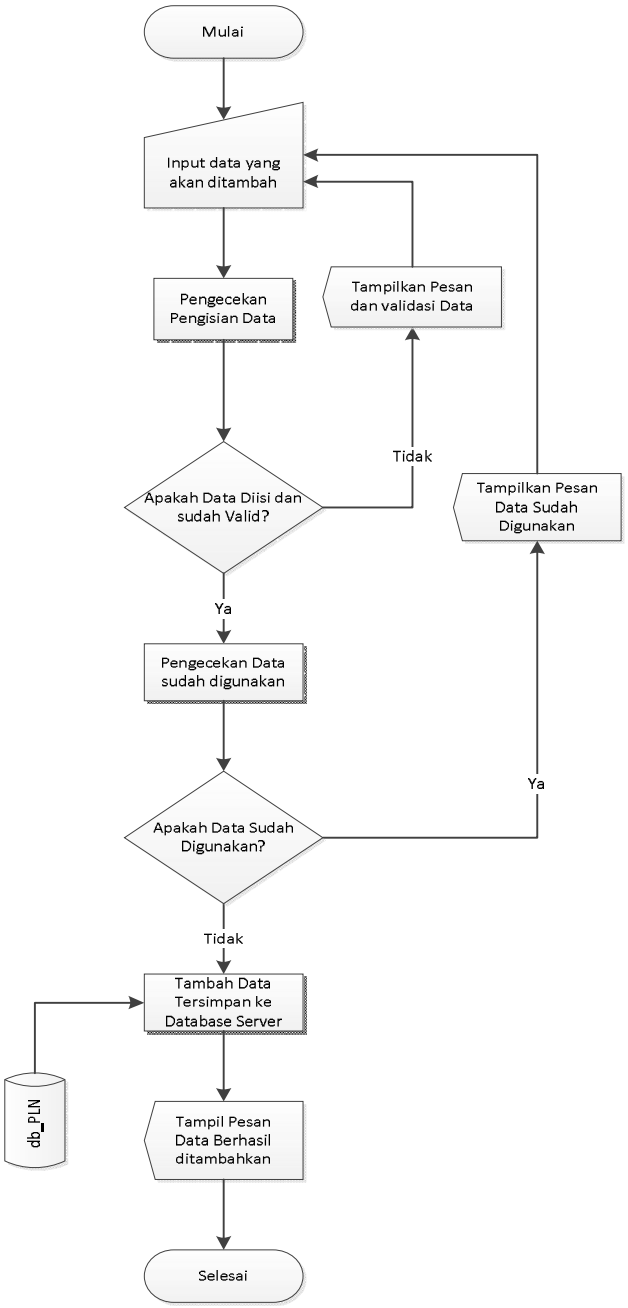
Sebagai langkah terakhir dalam proses perancangan yaitu merancang *method* yang akan diimplementasikan ke dalam sistem. *Method* ini akan digunakan sebagai algoritma dasar dalam mengkodekan prosedur yang ada. Adapun perancangan *method* untuk membangun aplikasi penjadwalan yang akan dibangun adalah sebagai berikut :

1. *Method login*, merupakan prosedur yang terjadi ketika admin mulai mengakses aplikasi. Berikut ini gambaran dari perancangan *Method login* pada aplikasi ini dapat dilihat pada gambar 3.72.



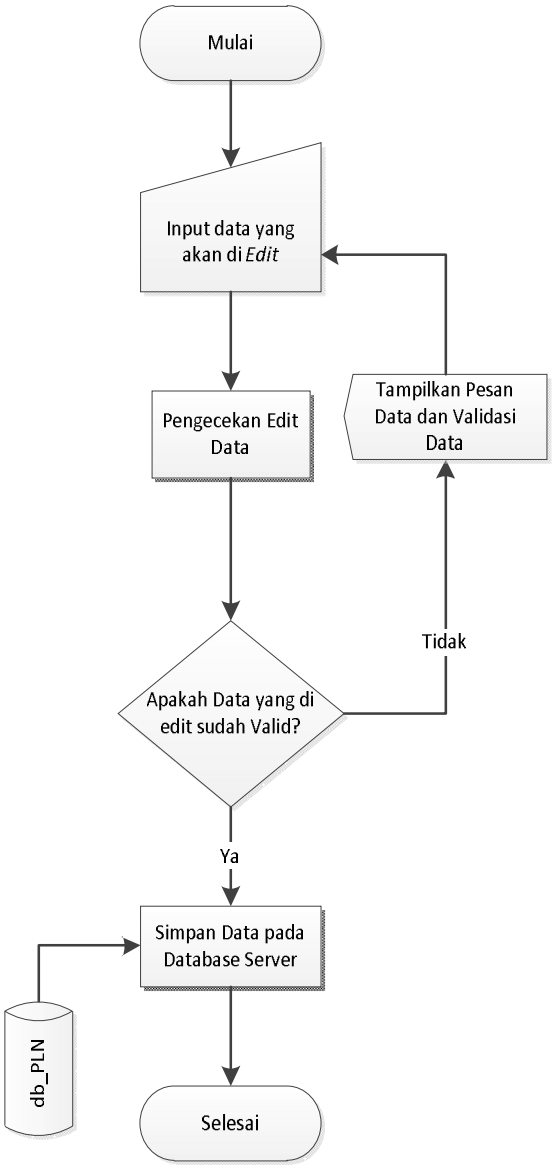
Gambar 3.72 Perancangan *Method Login*

2. *Method* tambah data, merupakan prosedur yang terjadi ketika admin menambahkan data baru. Berikut ini gambaran dari perancangan *Method* tambah data pada aplikasi ini dapat dilihat pada gambar 3.73.



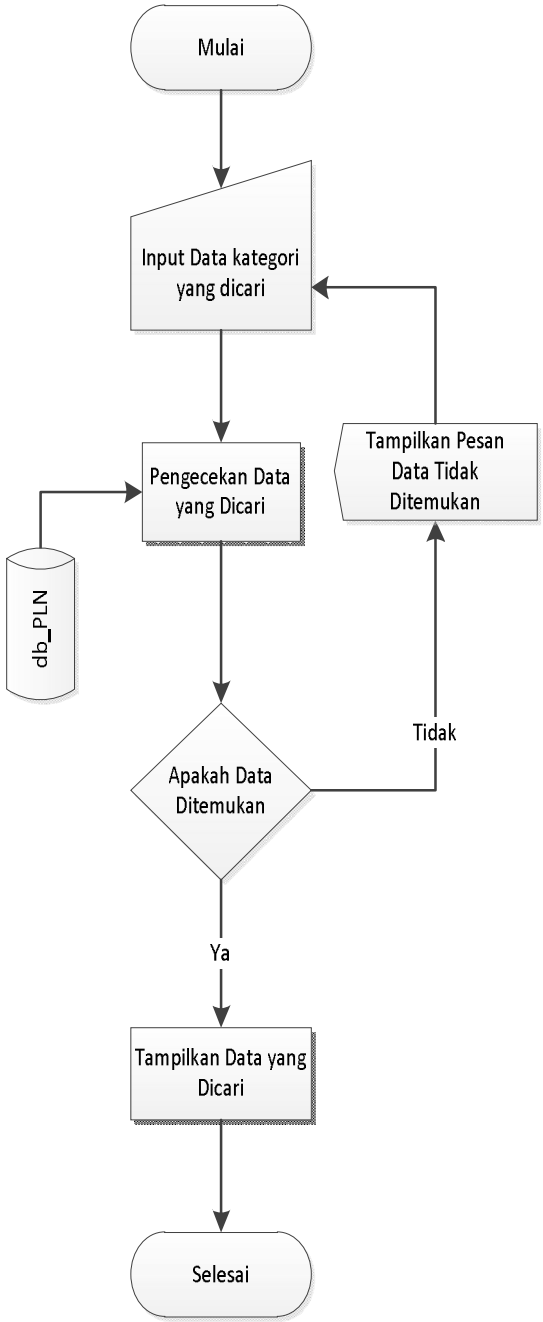
Gambar 3.73 Perancangan *Method* Tambah Data

3. *Method edit* data, merupakan prosedur yang terjadi ketika admin merubah data. Berikut ini gambaran dari perancangan *Method edit* data pada aplikasi ini dapat dilihat pada gambar 3.74.



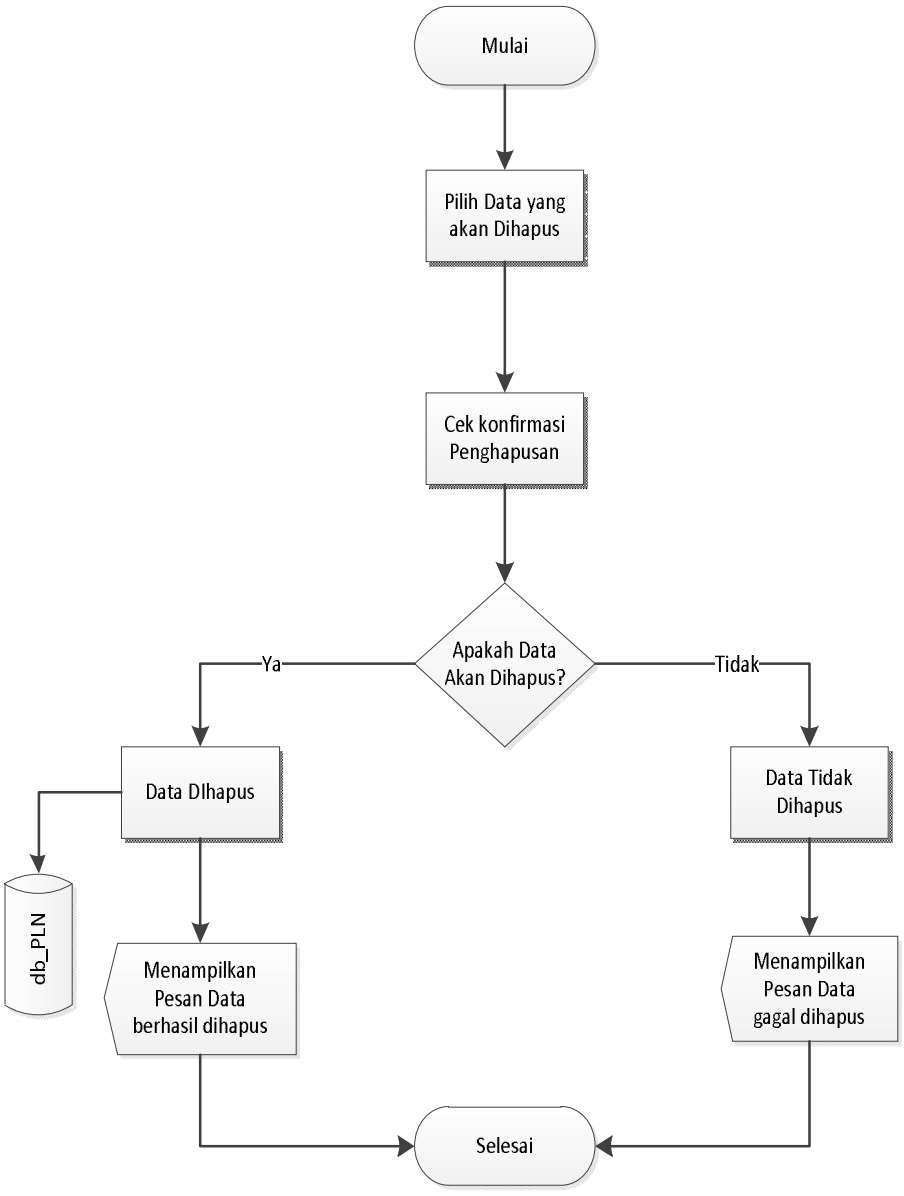
Gambar 3.74 Perancangan *Method Edit* Data

4. *Method* cari data, merupakan prosedur yang terjadi ketika admin mencari data berdasarkan kategori data. Berikut ini gambaran dari perancangan *method* cari data pada aplikasi ini dapat dilihat pada gambar 3.75.



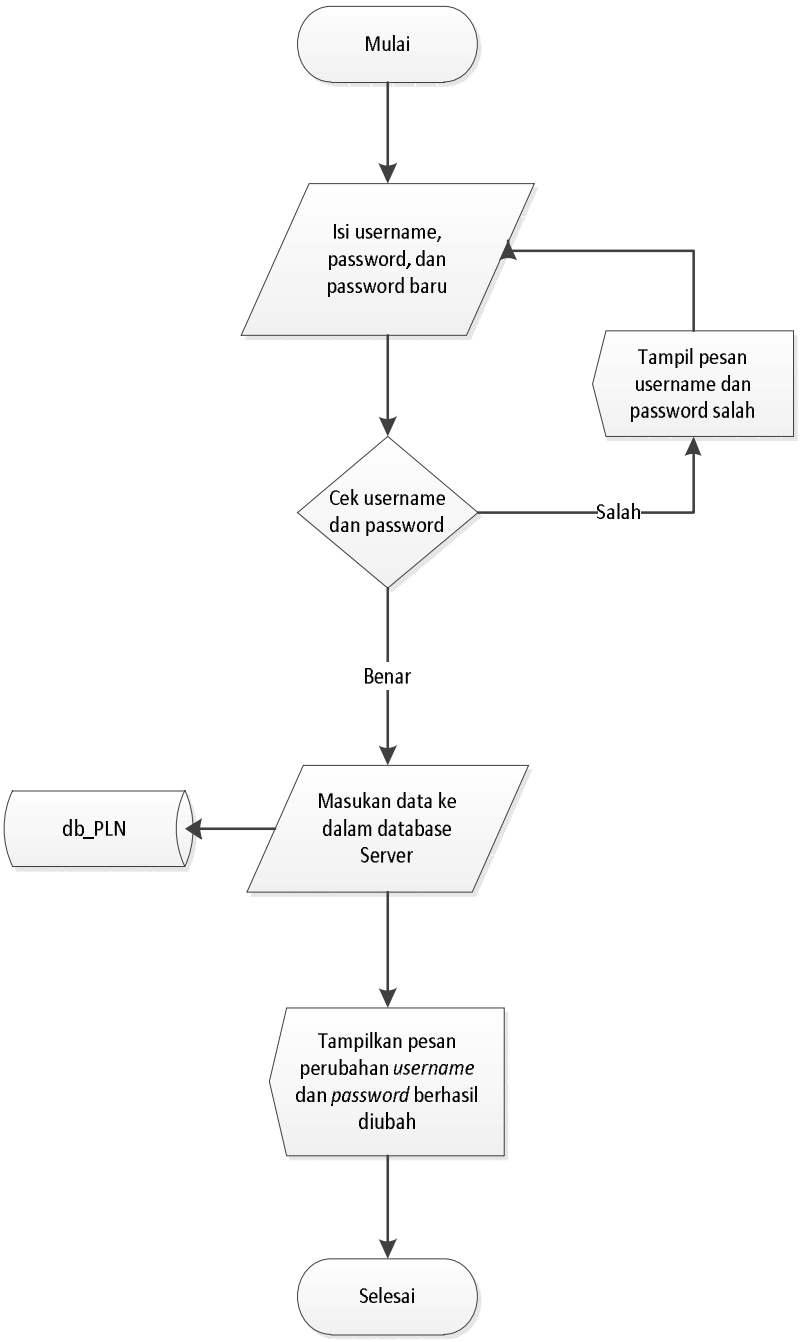
Gambar 3.75 Perancangan *Method* Cari Data

5. *Method* hapus data, merupakan prosedur yang terjadi ketika admin menghapus data. Berikut ini gambaran dari perancangan *method* hapus data pada aplikasi ini dapat dilihat pada gambar 3.76.



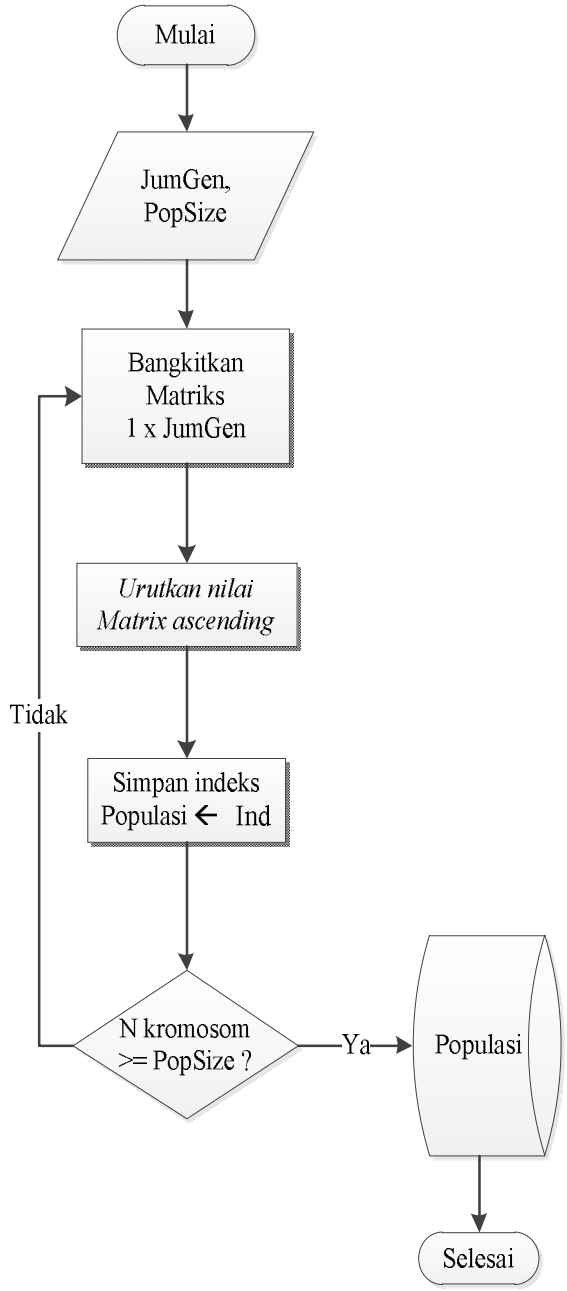
Gambar 3.76 Perancangan *Method* Hapus Data

6. *Method* ganti *password*, merupakan prosedur yang terjadi ketika admin merubah *password* lama. Berikut ini gambaran dari perancangan *Method* ganti *password* pada aplikasi ini dapat dilihat pada gambar 3.77.



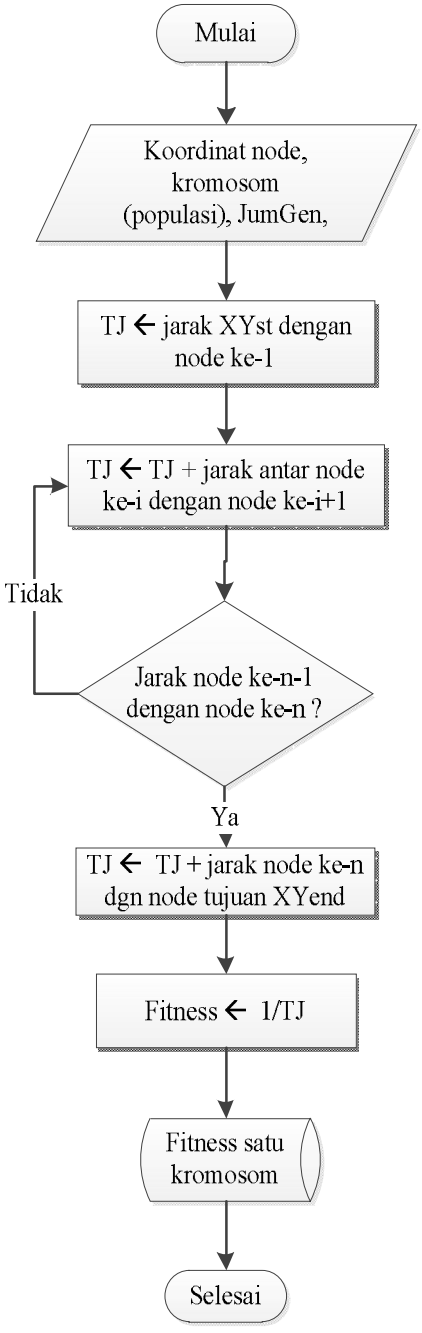
Gambar 3.77 Perancangan *Method* Ganti *password*

7. *Method* inisialisasi populasi, merupakan prosedur yang terjadi ketika admin melakukan inisialisasi populasi dengan menggunakan metode algoritma genetika. Berikut ini gambaran dari perancangan *Method* inisialisasi populasi pada aplikasi ini dapat dilihat pada gambar 3.78.



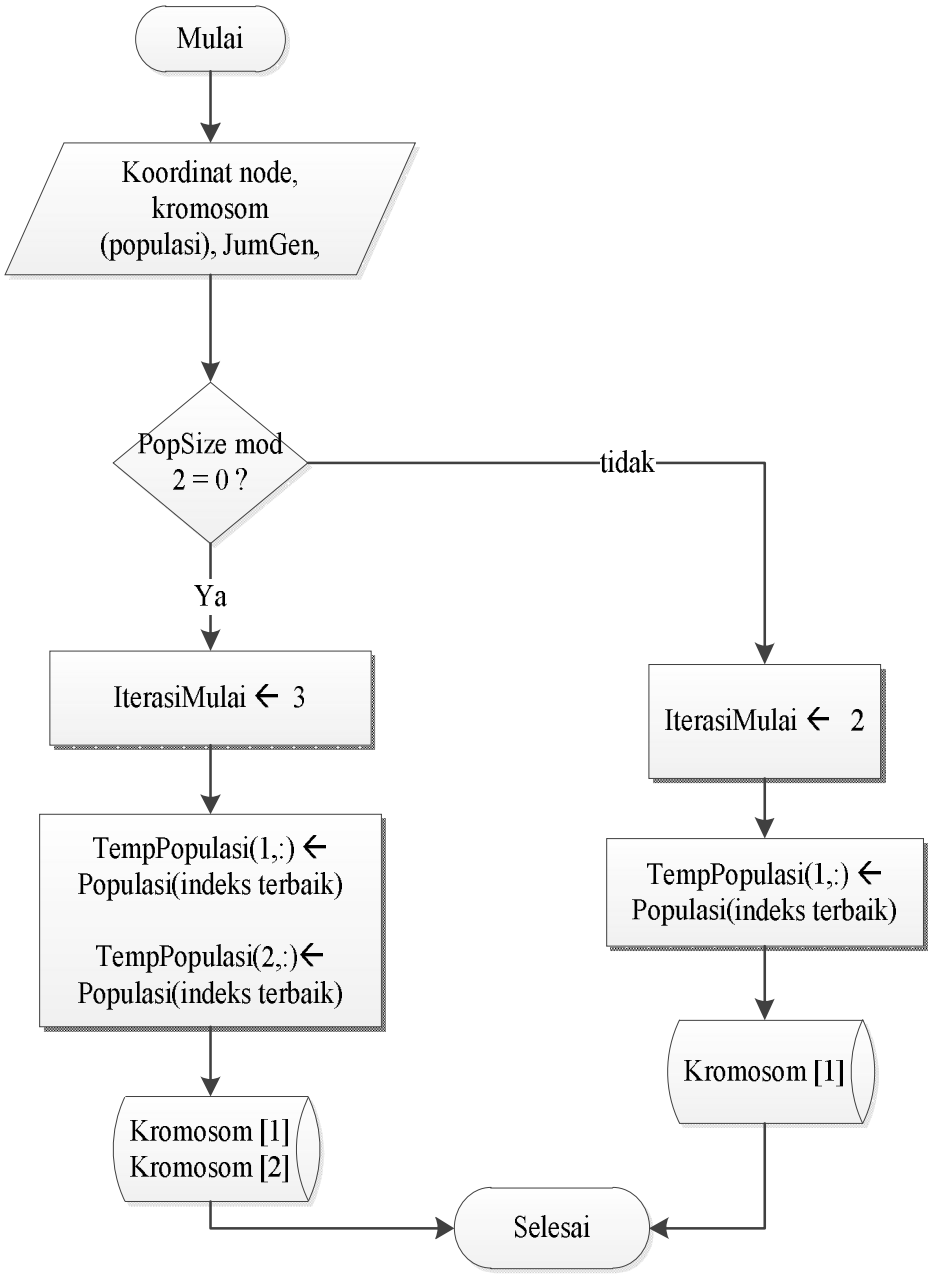
Gambar 3.78 Perancangan *Method* inisialisasi populasi

8. *Method* evaluasi individu, merupakan prosedur yang terjadi ketika melakukan evaluasi individu dengan menggunakan metode algoritma genetika. Berikut ini gambaran dari perancangan *method* evaluasi individu pada aplikasi ini dapat dilihat pada gambar 3.79.



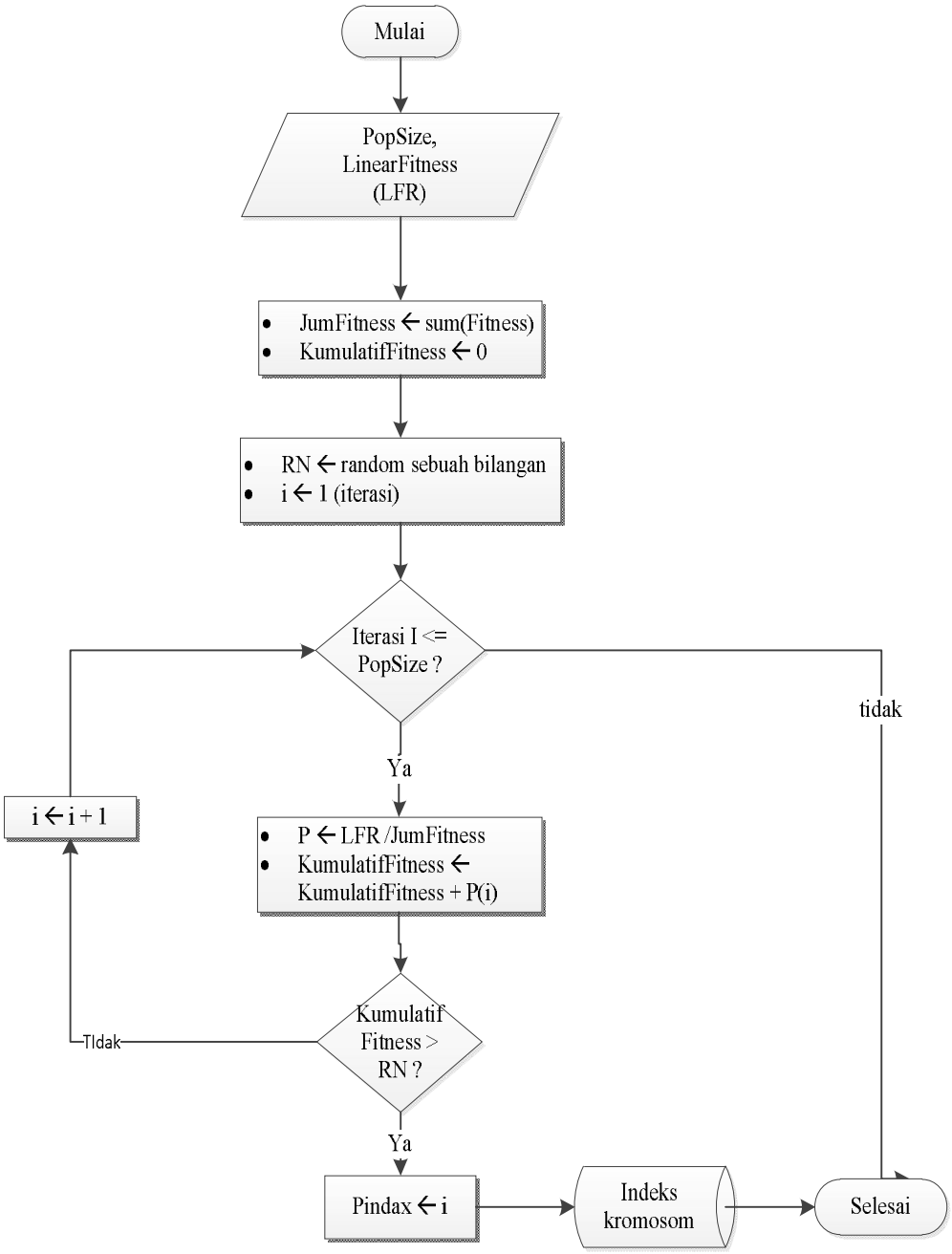
Gambar 3.79 Perancangan *Method* evaluasi individu

9. *Method* elitisme, merupakan prosedur yang terjadi ketika melakukan elitisme dengan menggunakan metode algoritma genetika. Berikut ini gambaran dari perancangan *method* elitisme pada aplikasi ini dapat dilihat pada gambar 3.80.



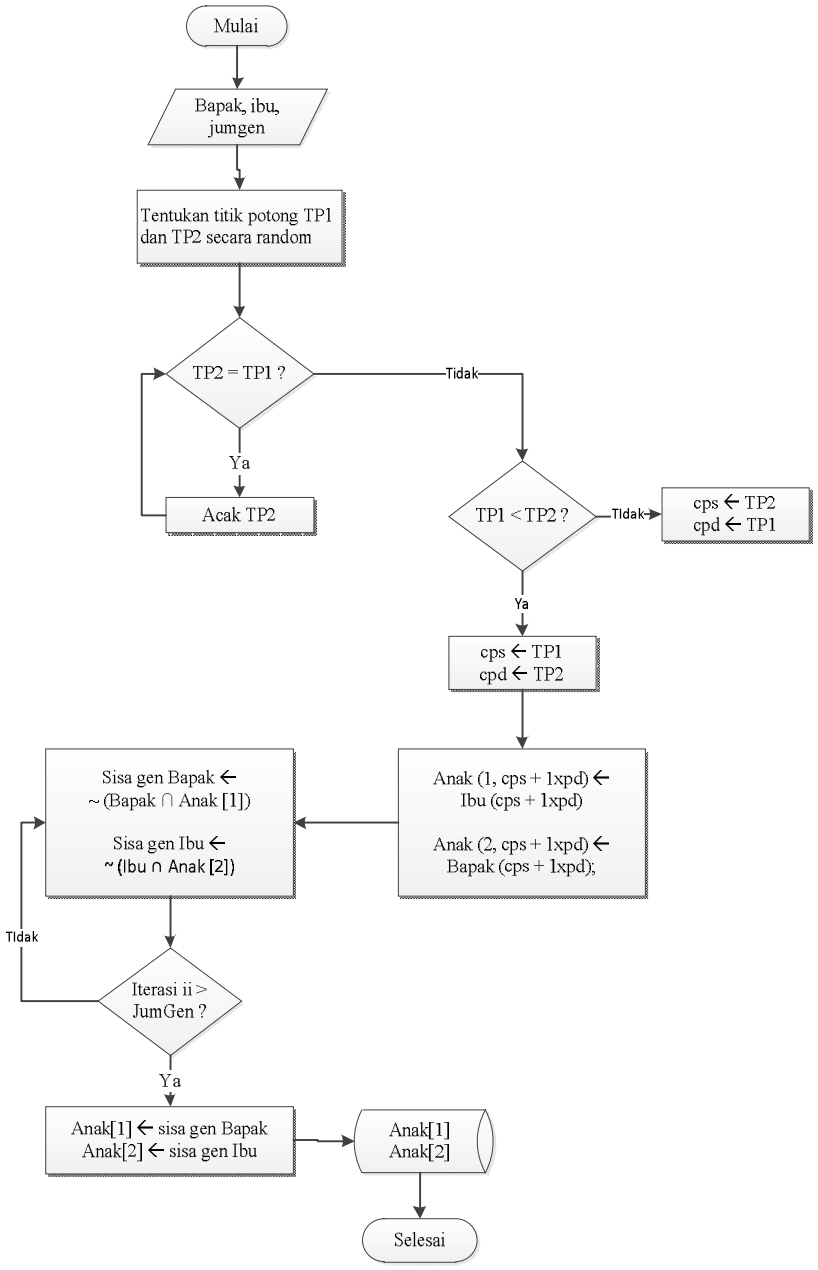
Gambar 3.80 Prancangan *Method* elitism

10. *Method* seleksi, merupakan prosedur yang terjadi ketika melakukan seleksi dengan menggunakan metode algoritma genetika. Berikut ini gambaran dari perancangan *method* seleksi pada aplikasi ini dapat dilihat pada gambar 3.81.



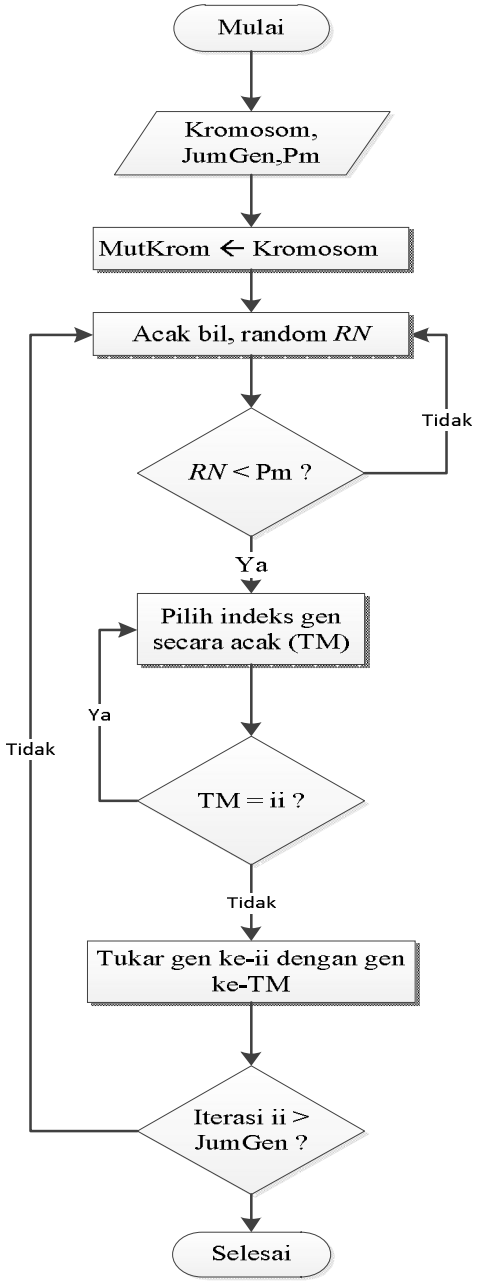
Gambar 3.81 Perancangan *Method* seleksi

11. *Method* pindah silang, merupakan prosedur yang terjadi ketika melakukan pindah silang dengan menggunakan metode algoritma genetika. Berikut ini gambaran dari perancangan *method* pindah silang pada aplikasi ini dapat dilihat pada gambar 3.82.



Gambar 3.82 Perancangan *Method* pindah silang

12. *Method* mutasi, merupakan prosedur yang terjadi ketika melakukan mutasi dengan menggunakan metode algoritma genetika. Berikut ini gambaran dari perancangan *method* mutasi pada aplikasi ini dapat dilihat pada gambar 3.83



Gambar 3.83 Perancangan *Method* mutasi