BAB III

ANALISA DAN DESAIN SISTEM

BAB III

ANALISA DAN DESAIN SISTEM

III.1 Analisis Masalah

Adapun permasalahan yang dihadapi oleh peneliti adalah tidak ada sebuah aplikasi yang memiliki sistem keamanan data yang dapat mengamankan data teks dan gambar pada PT. Pos Indonesia. Algoritma *3DES* (*Triple Data Encryption Standard*) adalah salah satu algoritma yang dapat digunakan untuk melakukan enkripsi data sehingga data asli hanya dapat dibaca oleh seseorang yang memiliki kunci enkripsi tersebut dengan cara menyandikan data.

III.1.1. Strategi Pemecahan Masalah

Strategi dalam melakukan pemecahan masalah yang sedang dianalisa oleh peneliti mengenai rancang bangun aplikasi pengamanan data teks dan gambar pada PT. Pos Indonesia dengan metode *Triple DES* adalah sebagai berikut:

1. Membangun sebuah aplikasi pengamanan data teks dan gambar pada PT. Pos Indonesia.
2. Menerapkan aplikasi pengamanan data dengan tingkat keamanan yang tinggi.
3. Merancang aplikasi dengan menggunakan metode *Triple DES* yang memiliki sistem keamanan data yang aman pada penyimpanan data file.

III.2. Penerapan Metode *Triple-DES*

Algoritma *3DES* (*Triple Data Encryption Standard*) merupakan salah satu algoritma simetris pada kriptografi yang digunakan untuk mengamankan data dengan cara menyandikan data. Proses yang dilakukan dalam penyandian datanya, yaitu proses enkripsi dan proses *Dekripsi*. Algoritma *3DES* adalah suatu algoritma pengembangan dari algoritma *DES* (*Data Encryption Standard*). Perbedaan *DES* dengan *3DES* terletak pada panjangnya kunci yang digunakan. Pada *DES* menggunakan satu kunci yang panjangnya 56-bit, sedangkan pada *3DES* menggunakan 3 kunci yang panjangnya 168- bit (masing-masing panjangnya 56-bit). Pada *3DES*, 3 kunci yang digunakan bisa bersifat saling bebas (K1, K2, K3) atau hanya dua buah kunci yang saling bebas dan satu kunci lainnya sama dengan kunci pertama (K1,K2 dan K3 = K1). Karena tingkat kerahasiaan algoritma *3DES* terletak pada panjangnya kunci yang digunakan, maka penggunaan algoritma *3DES* dianggap lebih aman dibandingkan dengan algoritma *DES*. Untuk memudahkan penggunaan algoritma *3DES*, maka dibuat suatu program algoritma *3DES* dengan alat bantu *software* komputer, yaitu Visual Studio yang dapat mengenkripsi dan men*Dekripsi* file.

III.2.1 Proses Enkripsi Algoritma *Triple-DES*

Proses enkripsi *3DES* dapat dihitung sebagai berikut ini:

Gambar III.1 Contoh Plainteks Penerima Paket

Asumsikan Kunci:

Kunci = ZULHAM SETIAWAN

Kunci *3DES* kedalam urutan bilangan HEX dan BINER pada ASCII:

Tabel III.1. Konversi Kunci ke Biner

K1			
Char	**Biner**	**Char**	**Biner**
Z	1011010	T	1010100
U	1010101	I	1001001
L	1001100	A	1000001
H	1001000	W	1010111
A	1000001	A	1000001
M	1001101	N	1001110
S	1010011		
E	1000101		

223 = 11011111

56 = 00111000

111 = 01101111

2 = 00000010 BLOK 2

94 = 01011110

215 = 11010111

226 = 11100010 BLOK 1
13 = 00001011
113 = 01110001
206 = 11001110
180 = 10110100
149 = 10010101
172 = 10101100 BLOK 2
244 = 11110100
178 = 10110010
140 = 10001100

155 = 10011011
63 = 00111111
89 = 01011001
43 = 00101011 BLOK 1
112 = 01110000
20 = 00010100
230 = 11100110
89 = 01011001

Blok chipher asli dipermutasi dengan matrik permutasi awal (*initial permutation* atau *IP*). Bisa ditulis x0 = IP (x) = L0 R0, dimana L0 terdiri dari 32 bit pertama dari x0 dan 32 bit terakhir dari R0. IP(x) : 11011111 00111000 01101111 11100010 00001011 01110001 11001110 10110100 Pecah bit pada IP(x) menjadi 2 bagian yaitu: L0 : 11011111 00111000 01101111 11100010 R0: 00001011 01110001 11001110 10110100 Lakukan pergeseran kiri (*Left Shift*) pada L0 dan R0, sebanyak 1 atau 2 kali berdasarkan kali putaran yang ada pada tabel putaran sebagai berikut:

Tabel III.2. Left Shift

putaran, i	*Jumlah Pergeseran Bit*
1	1
2	1
3	2
4	2
5	2

6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2
15	2
16	1

Untuk putaran ke 1 sampai dengan 16 dilakukan pegeseran 1 bit ke kiri Berikut hasil outputnya:

L0 :11011111 00111000 01101111 11100010

R0 :00001011 01110001 11001110 10110100

Digeser 1 bit ke kiri menjadi

L1 : 011011111 00111000 01101111 1110001
R1 : 000001011 01110001 11001110 1011010

Digeser 2 bit ke kiri menjadi

L2 :01011011111 00111000 01101111 11100

R2 : 10000001011 01110001 11001110 10110

Digeser 2 bit ke kiri menjadi

L3 :0001011011111 00111000 01101111 111

R3 : 1010000001011 01110001 11001110 101

Digeser 2 bit ke kiri menjadi

L4 : 110001011011111 00111000 01101111 1

R4 : 011010000001011 01110001 11001110

Digeser 2 bit ke kiri menjadi

L5 :11110001011011111 00111000 0110111

R5 :10011010000001011 01110001 110011

Digeser 2 bit ke kiri menjadi

L6 : 1111110001011011111 00111000 01101

R6 : 1110011010000001011 01110001 1100

Digeser 2 bit ke kiri menjadi

L7 : 0111111100011011011111 00111000 011

R7 : 0011100110100000001011 01110001 11

Digeser 2 bit ke kiri menjadi

L8 : 1101111111000110110111111 00111000 0

R8 : 1100111001101000001011 01110001

Digeser 1 bit ke kiri menjadi

L9 : 0110111111100011011011111 00111000

R9 : 1110011100110100000001011 0111000

Digeser 2 bit ke kiri menjadi

L10 : 0001101111111000110110111111 001110

R10: 0011100111001101000001011 01110

Digeser 2 bit ke kiri menjadi

R11 : 10000110 11111110 00101101 11110011

L11 : 10001110 01110011 01000000 10110110

Digeser 2 bit ke kiri menjadi

L12 : 11100001 10111111 10001011 01111100

R12 : 11100011 10011100 11010000 00101101

Digeser 2 bit ke kiri menjadi

L13 : 00111000 01101111 11100010 11011111

R13 : 10111000 11100111 00110100 00001010

Digeser 2 bit ke kiri menjadi

L14 : 11001110 00011011 11111000 10110111

R14 : 01101110 00111001 11001101 00000010

Digeser 2 bit ke kiri menjadi

L15 : 11110011 10000110 11111110 00101101

R15 : 01011011 10001110 01110011 01000001

Digeser 1 bit ke kiri menjadi

L16 : 11111001 11000011 01111111 00010110

R16 : 00101101 11000111 00111001 10100001

Hasil putaran terakhir digabungkan kembali menjadi LiRi dan di XOR kan dengan kunci yang telah di tentukan: Berikut hasil prosesnya: Proses iterasi 1 adalah melakukan XOR antara kunci L16R16dan K1

Iterasi - 1 E(R(1) = 11111001 11000011 0111111 10001011 00010110 11100011 10011100 11010000

K1 = 01011001 01010101 01001100 01001001 01000001 01001110 01010100 01001001

A1 = 10100000 10010110 00110011 11000011 01010111 10101101 11001000 10011001

Hasil gabungan L16-R16

L16R16 = 10100000 10010110 00110011 11000011 01010111 10101101 11001000 10011001

Hasil putaran terakhir digabungkan kembali menjadi LiRi dan di XOR kan dengan kunci yang telah di tentukan : Berikut hasil prosesnya: Proses iterasi 1 adalah melakukan XOR antara kunci L16R16dan K1 Berdasarkan hasil dari S-box diatas dikonversikan ke biner dengan jumlah 8 bit sebagai berikut:

2=00000010 4=00000100 13=00001101 1=00000001 11=00001011 12=00001100 4=00000100 15=00001111

Tabel III.3. Hasil Pengambilan Nilai S-Box

S-BOX	HASIL	
	***DES*IMAL**	**BINER**
1	2	00000010
2	4	00000100
3	13	00001101
4	1	00000001
5	11	00001011
6	12	00001100
7	4	00000100
8	15	00001111

Kemudian biner hasil nilai S-box digabungkan menjadi L0R0 berikut pembentukkannya:

L0R0 = 00000010 00000100 00001101 00000001 00001011 00001100 00000100 00001111

Setelah 64 bit nilai biner digabungkan maka nilai biner di bagi menjadi dua bagian yaitu 32 bit L0 dan 32 bit R0, berikut pembentukkannya : Kemudian hasil nilai S-box dibagi menjadi dua bagian yaitu 32 bit L0 dan 32 bit R0, berikut pembentukkannya :

L0= 00000010 00000100 00001101 00000001

R0= 00001011 00001100 00000100 00001111

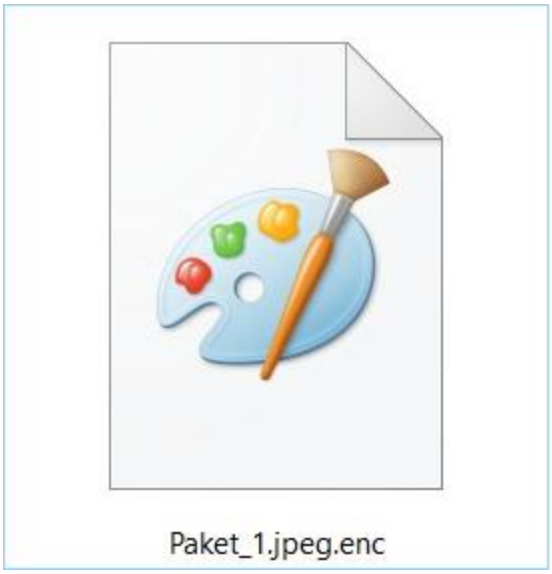
Langkah terakhir adalah menggabungkan R16 dengan L16Sehingga Input:

R16L16 = 00000111 00000101 00001110 00001001 00000001 00000000 00001000 00000011

Menghasilkan Output: *Cipher* (dalam biner) sebagai berikut:

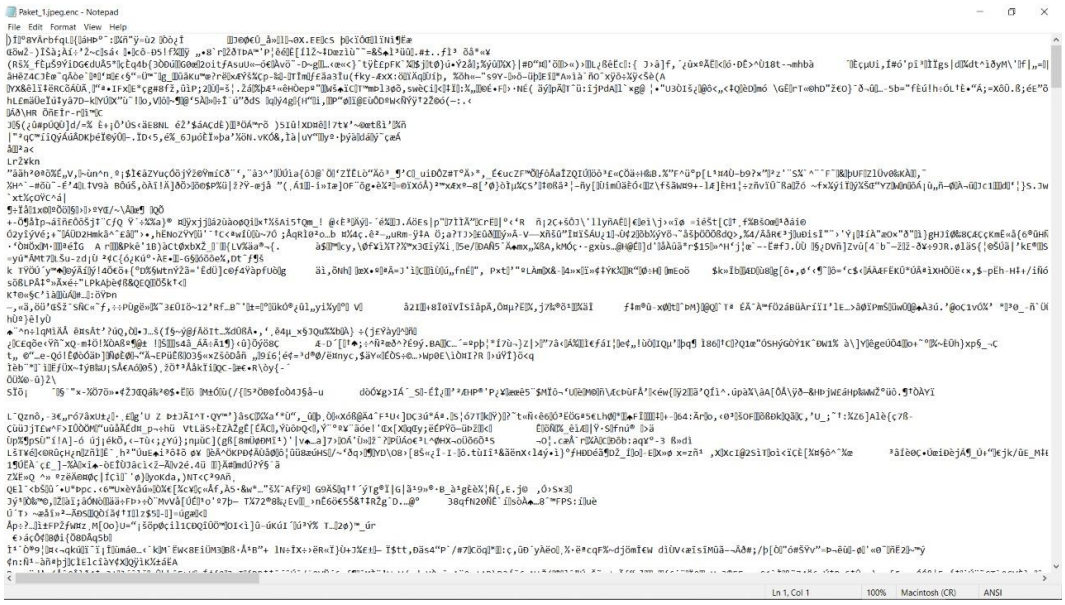
10011000=**152** 00011110=**30** 10000001=**129** 01110010=**114** 10001101=**141** 00000010=**3** 11010111=**215** 00010110=**22**

Akhirnya, keempat *sub-block* tersebut digabungkan kembali lagi sehingga menghasilkan sandi yang dirahasiakan atau disebut dengan *chiper*. Berikut adalah contoh gambar atau plaintext setelah dilakukannya proses enkripsi:



Gambar III.2 Contoh *Chiperteks* Penerima Paket

Jika *chiperteks* dibuka dengan mengunakan notepad maka file akan hancur dan tidak sama sekali dikenal isi dari file tersebut seperti gambar dibawah ini:



Gambar III.3 Contoh *Chiperteks* dibuka dengan Notepad

III.2.2 Proses *Dekripsi* Algoritma *Triple-DES*

Selanjutnya akan dilakukan proses *Dekripsi* berdasarkan *3DES* pada nilai *RGB* lakukan perulangan dari biner *chiper* ke *Initial Permutation* (*IP*) berikut:

152 = 10011000

30 = 00011110

129 = 10000001

114 = 01110010 BLOK 1

141 = 10001101

215 = 11010111

22 = 00010110

120 = 01111000

111 = 01101111

95 = 01011111 BLOK 2

1 = 00000000 BLOK 2

33 = 00100001

90 = 01011010

75 = 01001011

16 = 00010000

30 = 00011110

15 = 00001110 BLOK 3

30 = 00011110

8 = 00000111

110 = 01101110

101 = 01100101 20 = 00010100

76 = 01001100

97 = 01100001

Blok pertama *cipher audio* dipermutasi dengan matrik permutasi awal (*initial permutation* atau IP). Bisa ditulis *x0 = IP (x) = L0 R0*, dimana *L*0 terdiri dari 32bit pertama dari x0 dan 32bit terakhir dari *R*0.

IP(x) : 01000011 00001010 00101110 00110010 00000010 00011011 01010100 01111000

Pecah bit pada IP(x) menjadi 2 bagian yaitu:

L_0 : 10011000 00011110 10000001 01110010

R_0: 10001101 00000010 11010111 00010110

Lakukan pergeseran kanan (*right Shift*) pada L0dan R0, sebanyak 1 atau 2 kali berdasarkan kali putaran yang ada pada tabel putaran sebagai berikut:

Tabel III.4. *Left Shift*

Putaran, *i*	**Jumlah Pergeseran Bit**
1	1
2	1
3	2
4	2
5	2
6	2
7	2
8	2
9	1
10	2
11	2
12	2
13	2
14	2

15	2
16	1

Untuk putaran ke 1 sampai dengan 16 dilakukan pegeseran 1 bit ke kanan Berikut hasil outputnya:

L0 : 10011000 00011110 10000001 01110010

R0: 10001101 00000010 11010111 00010110

Digeser 1 bit ke kanan menjadi

L1 :0011000 00011110 10000001 011100101

R1 :0001101 00000010 11010111 000101101

Digeser 2 bit ke kanan menjadi

L2 :11000000 11110100 00001011 10010100

R2 :01101 00000010 11010111 00010110100

Digeser 2 bit ke kanan menjadi

L3 :000000 11110100 00001011 1001010011

R3 :101 00000010 11010111 0001011010001

Digeser 2 bit ke kanan menjadi

L4 :00001111 01000000 10111001 01001100

R4 :10000001 01101011 10001011 01000110

Digeser 2 bit ke kanan menjadi

L5 :001111 01000000 10111001 0100110000

R5 :000001 01101011 10001011 0100011010

Digeser 2 bit ke kanan menjadi

L6 :1111 01000000 10111001 010011000000

R6 :0001 01101011 10001011 010001101000

Digeser 2 bit ke kanan menjadi

L7 :11 01000000 10111001 01001100000011

R7 :01 01101011 10001011 01000110100000

Digeser 2 bit ke kanan menjadi

L8 :01000000 10111001 0100110000001111

R8 :01101011 10001011 0100011010000001

Digeser 1 bit ke kanan menjadi

L9 :1000000 10111001 01001100000011110

R9 :1101011 10001011 01000110100000010

Digeser 2 bit ke kanan menjadi

L10:00000 10111001 0100110000001111010

R10:01011 10001011 0100011010000001011

Digeser 2 bit ke kanan menjadi

L11:00010111001

01001100000111101000

R11:011 10001011 010001101000000101101

Digeser 2 bit ke kanan menjadi

L12:010111001 0100110000011110100000

R12:110001011 0100011010000010110101

Digeser 2 bit ke kanan menjadi

L13: 0111001 0100110000001111010000001

R13:0001011 0100011010000001011010111

Digeser 2 bit ke kanan menjadi

L14: 11001 01001100000011111010000000101

R14:01011 01000110100000010110101110 0

Digeser 2 bit ke kanan menjadi

L15 :00101001100000011110100000010111

R15 :01101000110100000010110101110001

Digeser 1 bit ke kanan menjadi

L16 : 01010011000000111101000000101110

R16 :11010001101000000101101011100010

Hasil putaran terakhir digabungkan kembali menjadi LiRi dan di XOR

kan dengan kunci yang telah di tentukan :

Berikut hasil outputnya:

Iterasi - 1

= 01010011 00000011 11010000 00101110

11010001 10100000 01011010 11100010

K1 = 10001100 00001010 10011101

01110010 10001101 00011000

11001010 00010101

A1 = 11011111 00001001 01001101

01011100 01011100 10111000

10010000 11110111

Lakukan hal yang sama sampai proses iterasi ke-16 Iterasi – 16

E(R(16) = 11010101 01011111 11010011 00111011 11001100 01010110

10011110 01011101

Setiap *Vektor* Ai disubstitusikan kedelapan buah *S-Box*(*Substitution Box*), Tetapi pada proses ini hanya di butuhkan 6 bit untuk setiap *sbox*, maka 2 bit terakhir akan diabaikan.

Tabel 4. Pendefenisian *S-Boxes* dari Algoritma *3DES Column*

S1:

row	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S2:

row	**0**	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	15	4	8	14	6	11	3	4	9	7	2	13	12	0	5	10

1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

Berdasarkan S-Box di atas di peroleh hasil sebagai berikut: 7=00000111 5=00000101 14=00001110 9=000010011=00000001 0=00000000 8=00001000 3=00000011

Tabel III.5. Hasil Pengambilan Nilai*S-box*

S-BOX	HASIL	
	DECIMAL	**BINER**
1	7	00000111
2	5	00000101
3	14	00001110
4	9	00001001
5	1	00000001
6	0	00000000
7	8	00001000
8	3	00000011

Kemudian biner hasil nilai *S-box* digabungkan menjadi L0R0 berikut pembentukkannya:

L0R0 : 00000111 00000101 00001110 00001001

00000001 00000000 00001000 00000011

Setelah 64 bit nilai biner digabungkan maka nilai biner di bagi menjadi dua bagian yaitu 32 bit L0 dan 32 bit R0, berikut pembentukkannya :

L0= 00000111 00000101 00001110 00001001

R0= 00000001 00000000 00001000 00000011

Langkah selanjutnya adalah menentukan matriks permutasi untuk menentukan jumlah pergeseran bit yang akan diproses, berikut tabel permutasinya :

Langkah selanjutnya melakukan pergeseran kiri ke kanan sesuai tabel sebelum nya dengan memecah bit menjadi 2 bagian:

L0=00000111 00000101 00001110 00001001

R0=00000001 00000000 00001000 00000011

L1=10000011 10000010 10000111 00000100

R1=10000000 10000000 00000100 00000001

L2=01000001 11000001 01000011 10000010

R2=11000000 01000000 00000010 00000000

L3=10010000 01110000 01010000 11100000

R3=00110000 00010000 00000000 10000000

L4=00100100 00011100 00010100 00111000

R4=00001100 00000100 00000000 00100000

L5=00001001 00000111 00000101 00001110

R5=00000011 00000001 00000000 00001000

L6=10000010 01000001 11000001 01000011

R6=00000001 10000000 10000000 00000101

L7=11100000 10010000 01110000 01010000

R7=10000000 00110000 00010000 00000000

L8=00111000 00100100 00011100 00010100

R8=00100000 00001100 00000100 00000000

L9=01110000 01001000 00111000 00101000

R9=01000000 00110000 00010000 00000000

L10=00011100 00010010 00001110 00001010

R10=00010000 00000110 00000010 00000000

L11=10000111 00000100 10000011 10000010

R11=00000100 00000001 10000000 10000000

L12=10100001 11000001 00100000 11100000

R12=00000000 00000000 11000000 01000000

L13=10101000 01110000 01001000 00111000

R13=00000000 01000000 00011000 00001000

L14=00101010 00011100 00010010 00001110

R14=00000000 00010000 00000110 00000010

L15=00000111 00000101 00001110 00001001

R15=00000001 00000000 00001000 00000011

Langkah terakhir adalah menggabungkan R_{16} dengan L_{16}Sehingga Input :

L16 =00000111 00000101 00001110 00001001

R16 =00000001 00000000 00001000 00000011

Menghasilkan Output:

Cipher (dalam biner)

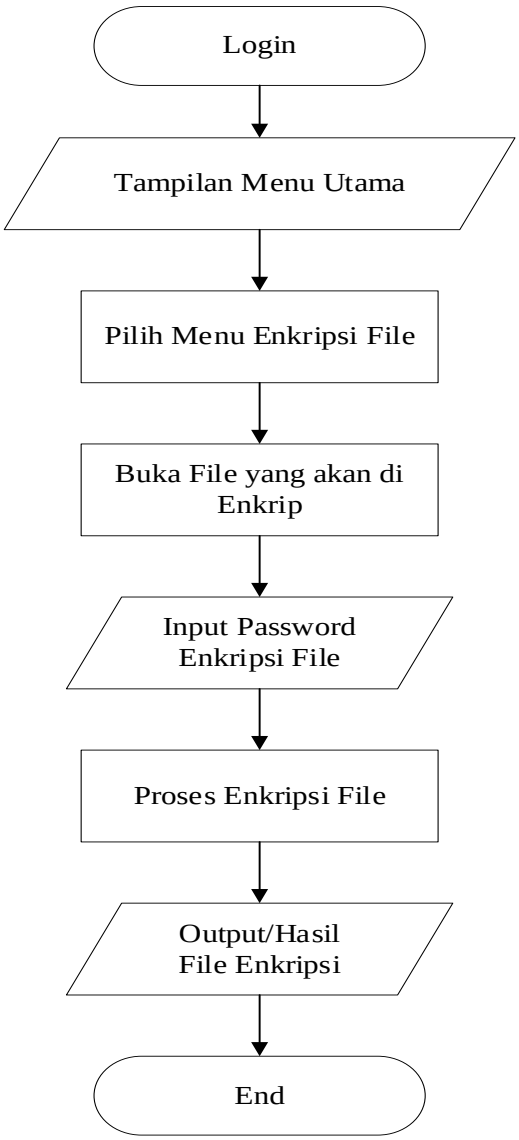
11011111= **223** 00111000= **56** 01101111= **111** 11100010= **226**

00001110= **13** 01110001 =**113** 11001110 =**206** 10110100=**180**

Akhirnya, keempat *sub-block* tersebut digabungkan kembali sehingga menghasilkan nilai RGB yang sudah disandi rahasiakan atau disebut dengan *descipher*. Akhirnya ketiga *sub-block* tersebut digabungkan kembali sehingga menghasilkan gambar asli seperti semula.

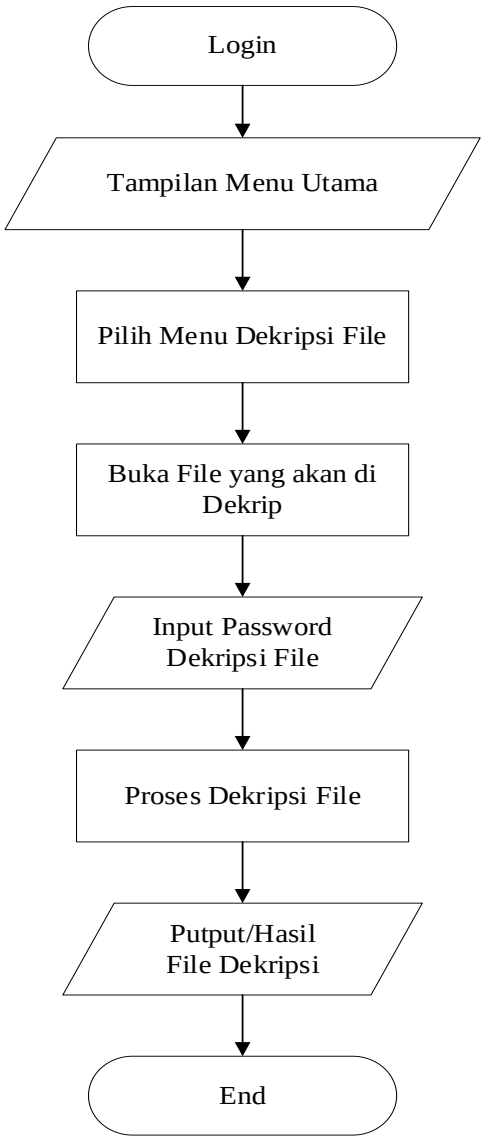
III.2.3 Flowchat Metode Triple-DES

Flowchart adalah adalah suatu bagan dengan simbol-simbol tertentu yang menggambarkan urutan proses secara mendetail dan hubungan antara suatu proses (instruksi) dengan proses lainnya dalam suatu program. Enkripsi adalah proses mengamankan suatu informasi dengan membuat informasi tersebut tidak dapat dibaca tanpa bantuan pengetahuan khusus. atau bisa didefinisikan juga Enkripsi merupakan proses untuk mengubah plainteks menjadi *chiperteks*. Plainteks sendiri adalah data atau pesan asli yang ingin dikirim, sedangkan *chiperteks* adalah data hasil enkripsi. Enkripsi dapat diartikan sebagai kode atau *chiper*.



Gambar III.4. *Flowchart* Enkripsi Algoritma *Triple-DES*

Proses *Dekripsi* yaitu proses konversi data yang sudah dienkripsi (*ciphertext*) kembali menjadi data aslinya (*Original Plaintext*) sehingga dapat dibaca/ dimengerti kembali.



Gambar III.5. *Flowchart Dekripsi* Algoritma *Triple-DES*

III.3 Desain Sistem

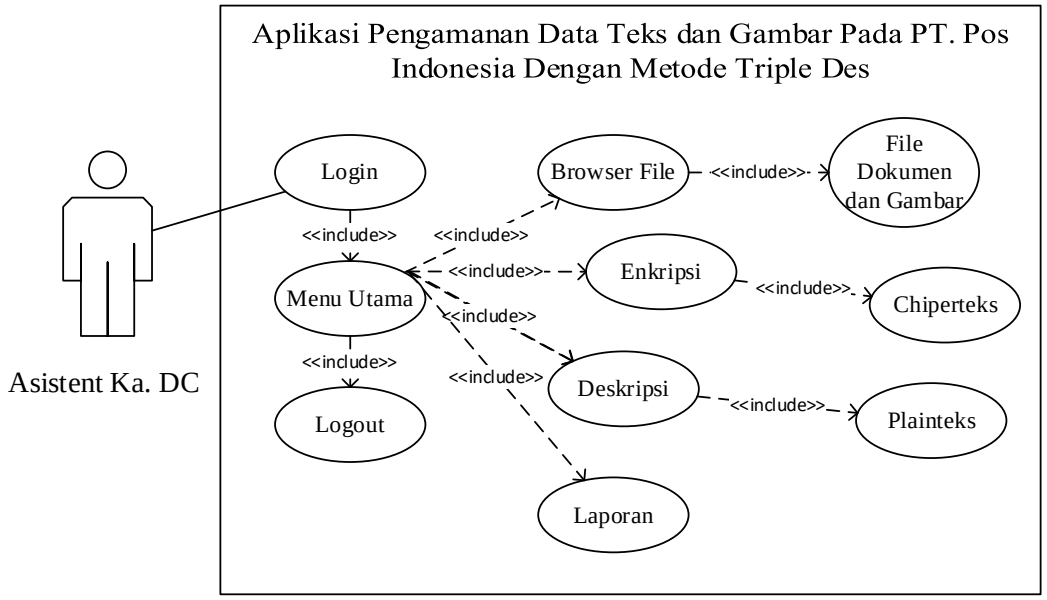
Desain sistem pada penelitian ini dibagi menjadi dua desain, yaitu desain sistem secara global untuk penggambaran model sistem secara garis besar dan desain sistem secara detail untuk membantu dalam pembuatan sistem.

III.3.1 Desain Sistem Secara Global

Desain sistem secara global menggunakan bahasa pemodelan *UML* yang terdiri dari *Usecase Diagram*, *Acitvity Diagram* dan *Sequence Diagram*.

1. *Usecase* Diagram

Secara garis besar, proses sistem yang akan dirancang digambarkan dengan *usecase* diagram yang terdapat pada Gambar III.3 sebagai berikut:



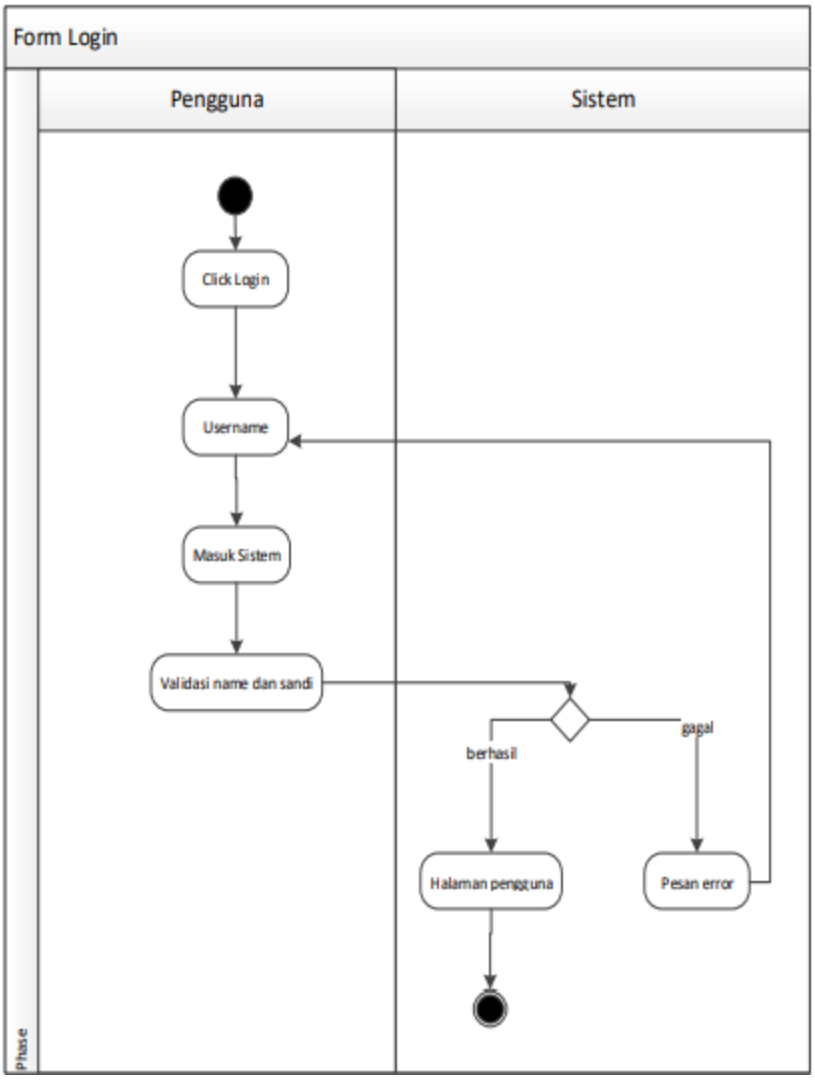
Gambar III.6. *Use Case* Diagram Rancang Bangun Aplikasi Pengamanan data Teks dan Gambar Pada PT.Pos Indonesia dengan Metode *Triple DES*

2. *Acitvity* Diagram

Proses yang telah digambarkan pada *use case diagram* akan dijabarkan dengan *Activity diagram* sebagai berikut:

a. Activity Diagram Login

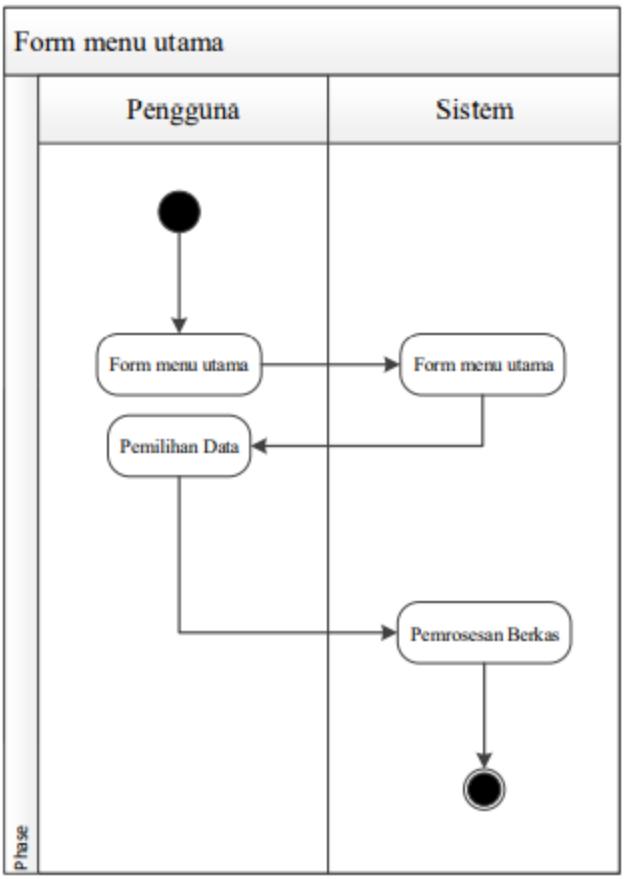
Aktivitas *login* yang dilakukan oleh user digambarkan dengan langkah-langkah sebagai berikut:



Gambar III.7. *Activity* Diagram Login

b. *Activity Diagram* Menu utama

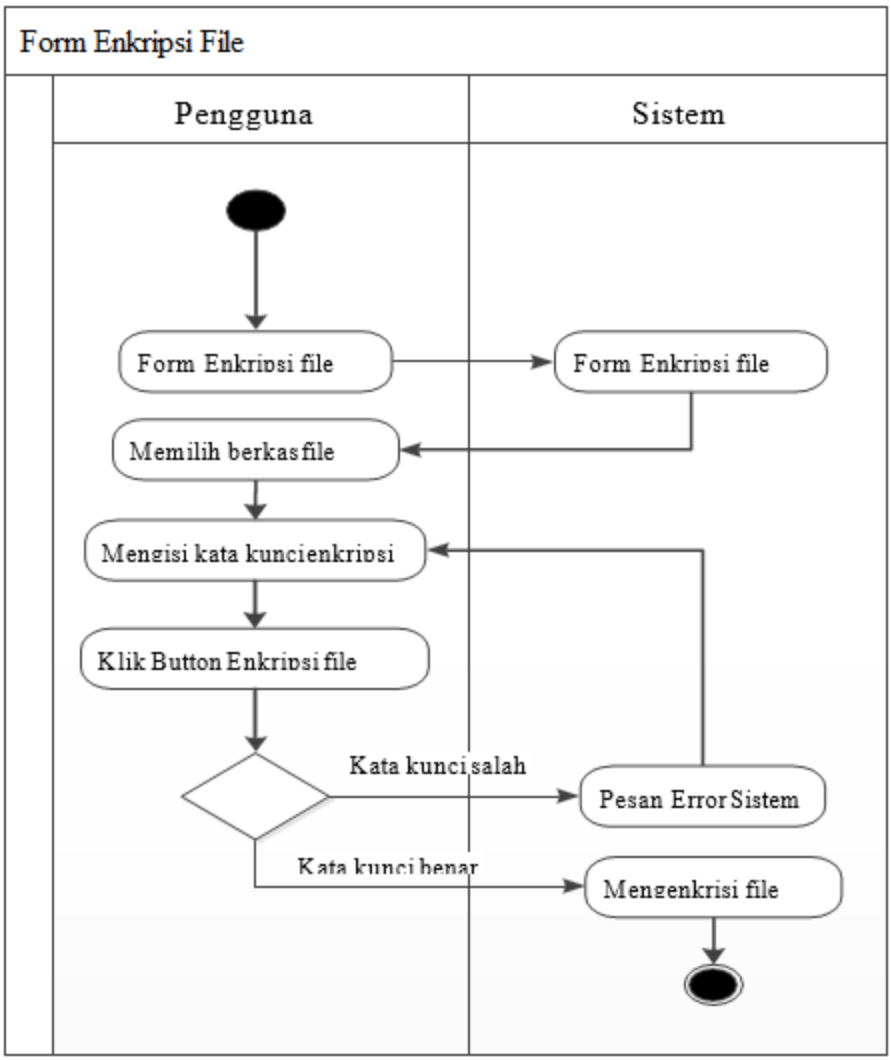
Aktivitas yang dilakukan oleh user pada form menu utama diterangkan dengan langkah-langkah pada gambar III.8 sebagai berikut:



Gambar III.8. *Activity Diagram* Form Menu utama

c. *Activity Diagram* Form Enkripsi File

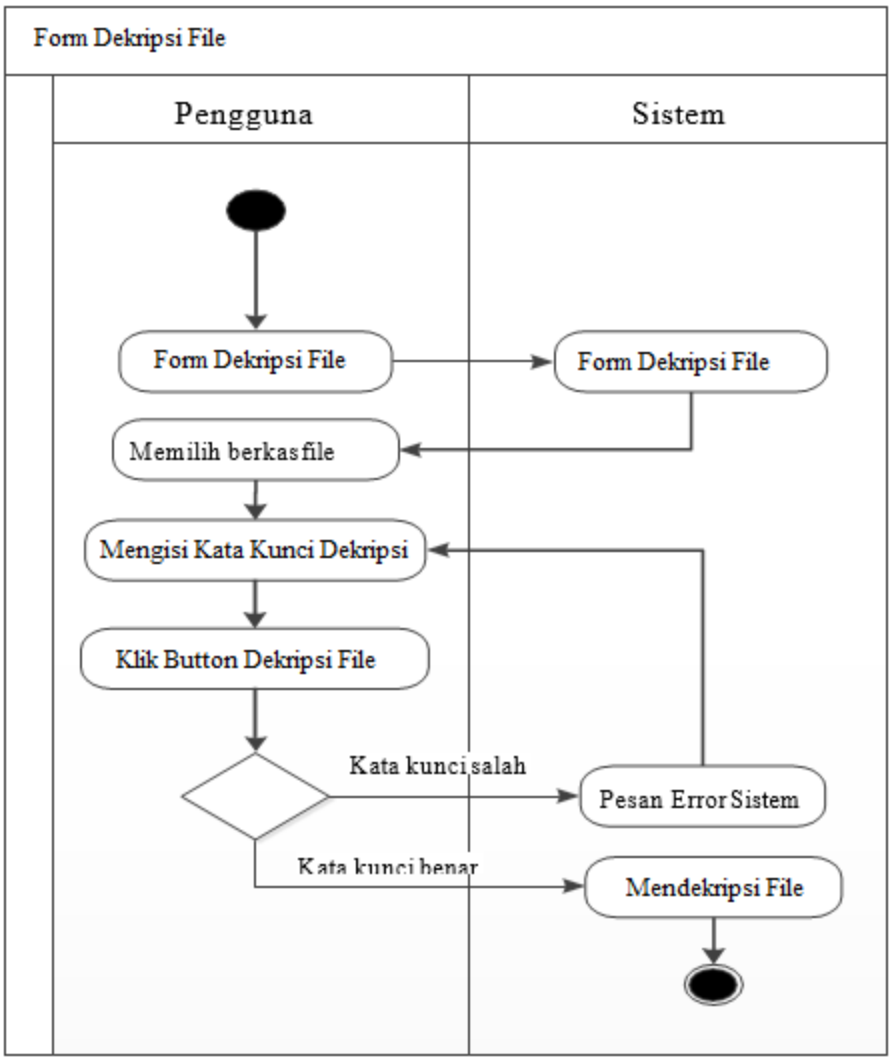
Aktivitas yang dilakukan oleh user pada form Enkripsi File diterangkan dengan langkah-langkah pada Gambar III.9 sebagai berikut:



Gambar III.9. *Activity Diagram* Form Enkripsi File

d. *Activity Diagram* Form *Dekripsi* File

Aktivitas yang dilakukan oleh user pada form *Dekripsi* File diterangkan dengan langkah-langkah pada Gambar III.10 sebagai berikut:

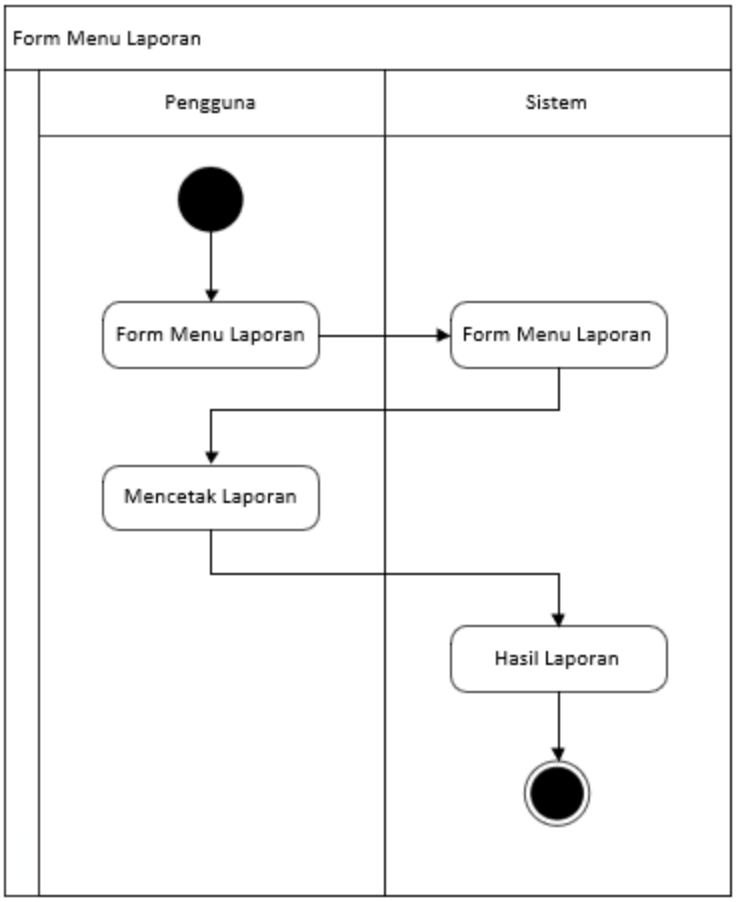


Gambar III.10. *Activity Diagram* Form *Dekripsi* File

e. *Activity Diagram* Form Laporan

Aktivitas yang dilakukan oleh user pada pada form laporan diterangkan

dengan langkah-langkah pada Gambar III.11 sebagai berikut:



Gambar III.11. *Activity Diagram* Form *Dekripsi* File

3. Sequence Diagram

Sequence Diagram adalah suatu diagram yang memperlihatkan atau menampilkan interaksi-interaksi antar objek di dalam sistem yang disusun pada

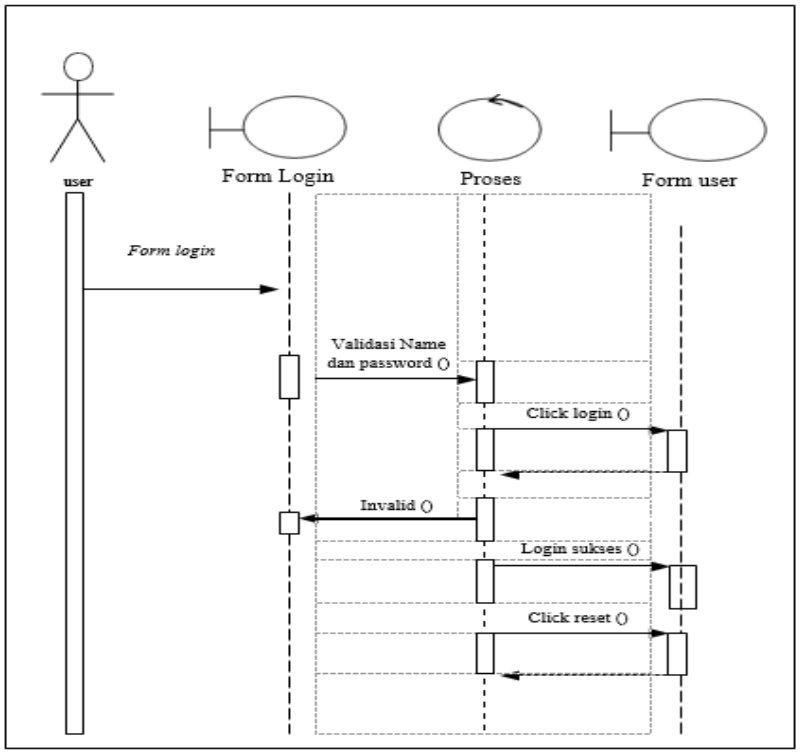
sebuah urutan atau rangkaian waktu. Interaksi antar objek tersebut termasuk user, *display*, dan sebagainya berupa pesan/*message*.

Sequence Diagram digunakan untuk menggambarkan skenario atau rangkaian langkah–langkah yang dilakukan sebagai sebuah respon dari suatu kejadian/event untuk menghasilkan output tertentu. *Sequence* Diagram diawali dari apa yang me-trigger aktivitas tersebut, proses dan perubahan apa saja yang terjadi secara internal dan output apa yang dihasilkan.

Diagram ini secara khusus berasosiasi dengan *use case* diagram. *Sequence* diagram juga memperlihatkan tahap demi tahap apa yang seharusnya terjadi untuk menghasilkan sesuatu di dalam use case. *Sequence* Diagram juga dapat merubah atribut atau method pada class yang telah dibentuk oleh class diagram, bahkan menciptakan sebuah class baru. *Sequence* Diagram memodelkan aliran logika dalam sebuah sistem dalam cara yang visual.

a. Sequence Diagram Login

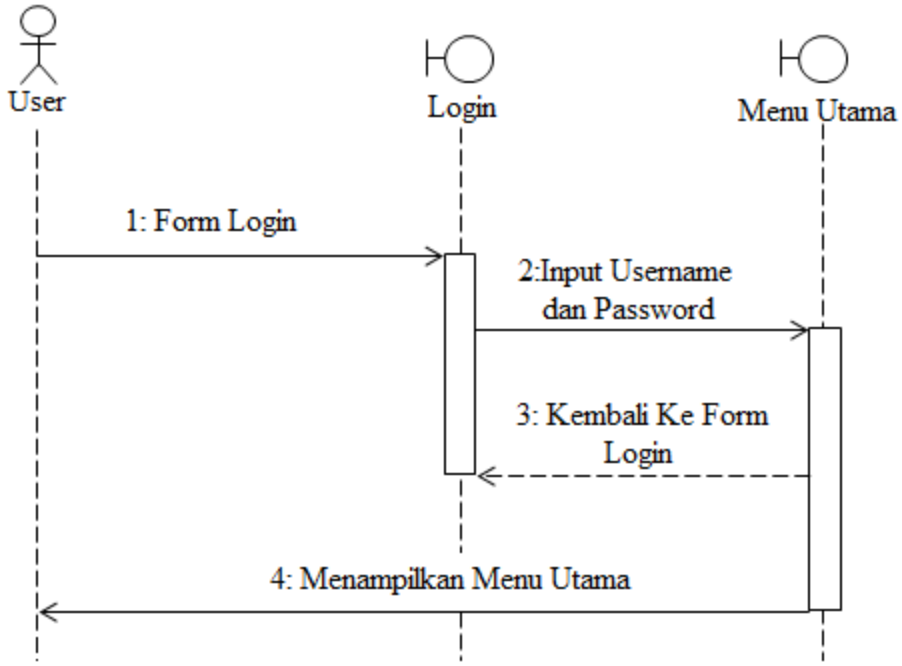
Serangkaian kegiatan *login* yang dilakukan oleh user dapat diterangkan dengan langkah-langkah sebagai berikut:



Gambar III.12. *Sequence Diagram Login*

b. *Sequence Diagram* Form Menu utama

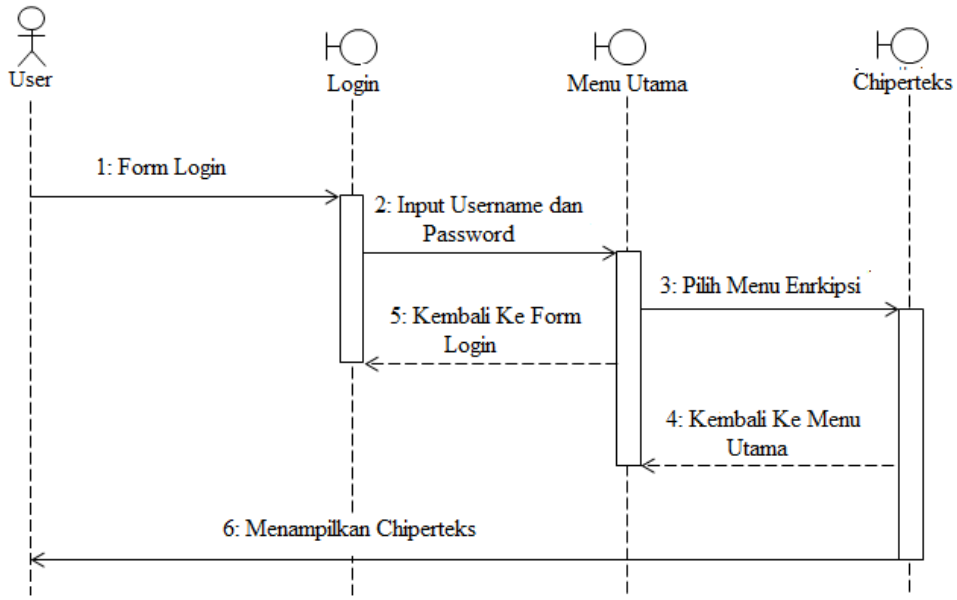
Serangkaian kinerja sistem yang dilakukan oleh User pada form menu utama dapat diterangkan dengan langkah-langkah pada gambar III.12 sebagai berikut:



Gambar III.12. *Sequence Diagram* Form Menu utama

c. *Sequence Diagram* Form Enkripsi File

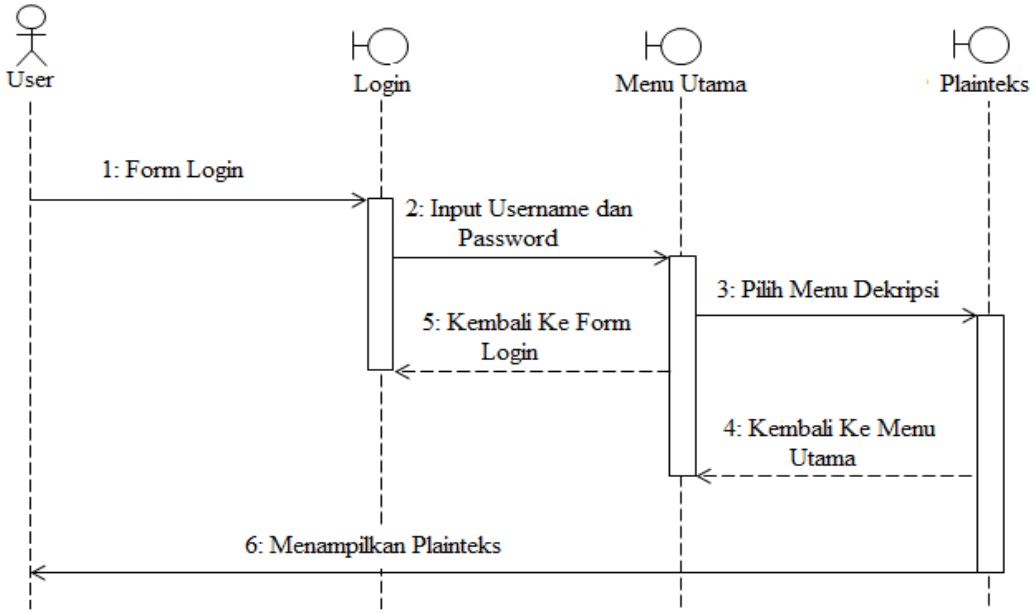
Serangkaian kinerja sistem yang dilakukan oleh User pada form Enkripsi File dapat diterangkan dengan langkah-langkah pada gambar III.13 sebagai berikut:



Gambar III.13. *Sequence Diagram* Form Enkripsi File

d. *Sequence Diagram* Form *Dekripsi* File

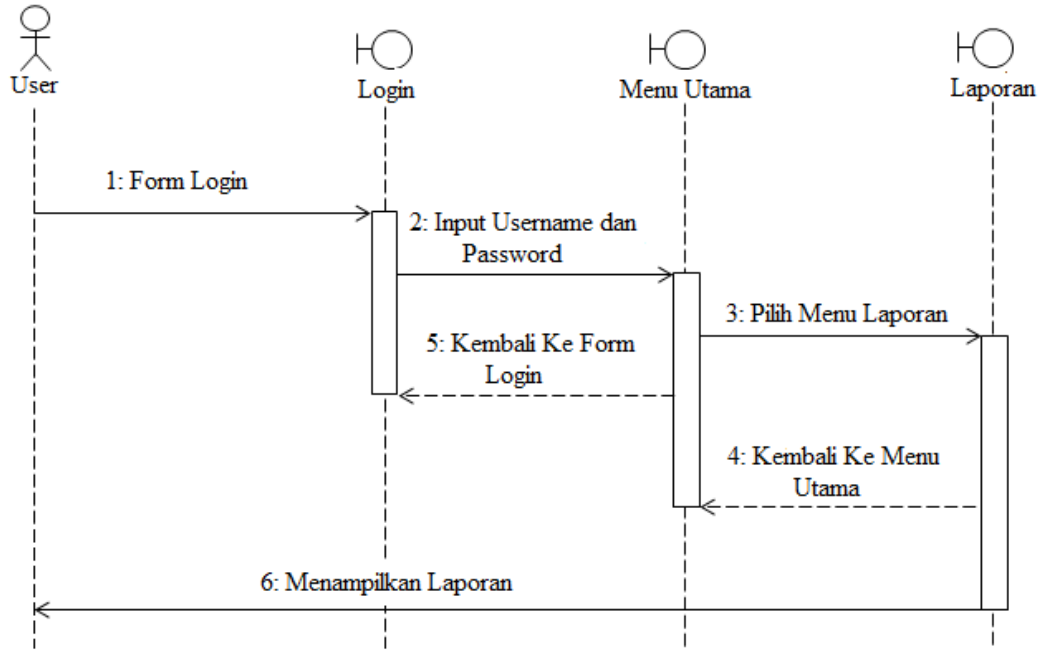
Serangkaian kinerja sistem yang dilakukan oleh User pada form *Dekripsi* File dapat diterangkan dengan langkah-langkah pada gambar III.14 sebagai berikut:



Gambar III.14. *Sequence Diagram* Form *Dekripsi* File

e. *Sequence Diagram* Form Laporan

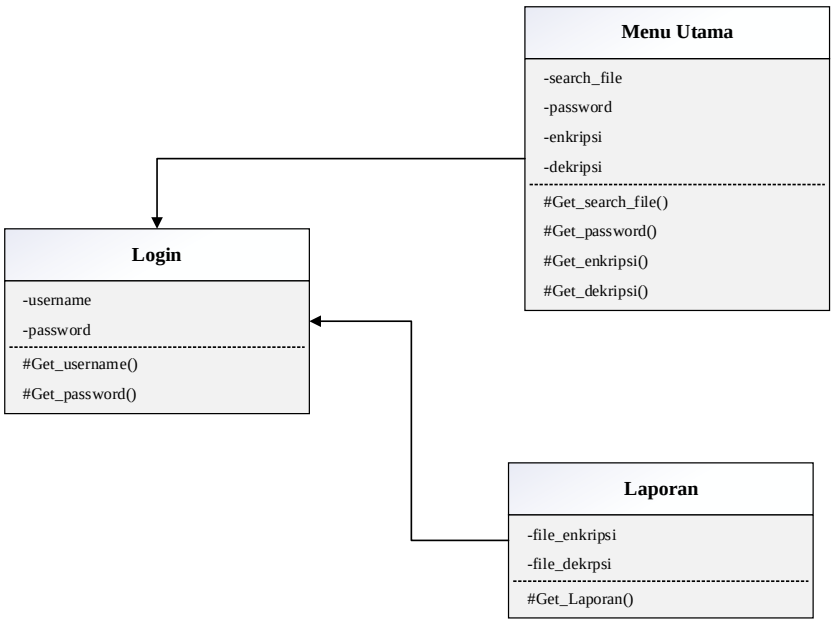
Serangkaian kinerja sistem yang dilakukan oleh user pada form Laporan dapat diterangkan dengan langkah-langkah pada gambar III.15 sebagai berikut:



Gambar III.15. *Sequence Diagram* Form Laporan

4. *Class Diagram*

Rancangan *class diagram* yang akan digunakan pada sistem dapat dilihat pada Gambar III.16 sebagai berikut:



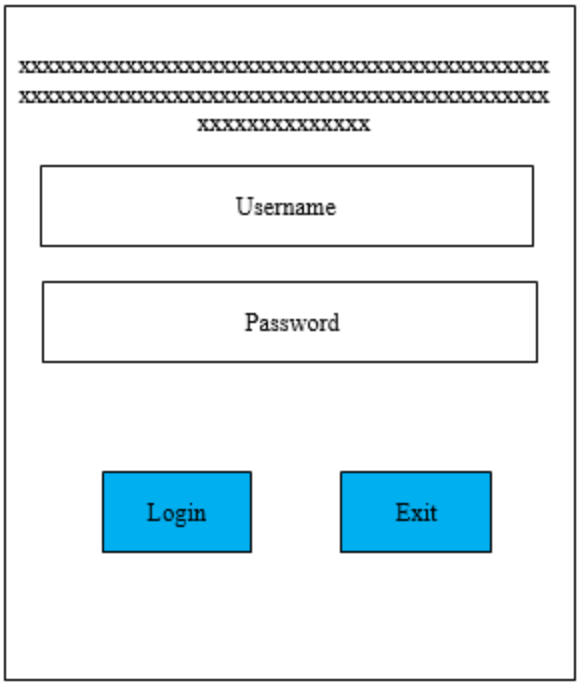
Gambar III.16. *Class Diagram*

III.1.2. Desain Sistem Secara Detail

Tahap perancangan berikutnya yaitu Desain sistem secara detail yang meliputi Desain sistem.

1. Desain Form *Login*

Desain tampilan *login* yang dilakukan oleh user dapat diterangkan dengan langkah-langkah *state* berikut:



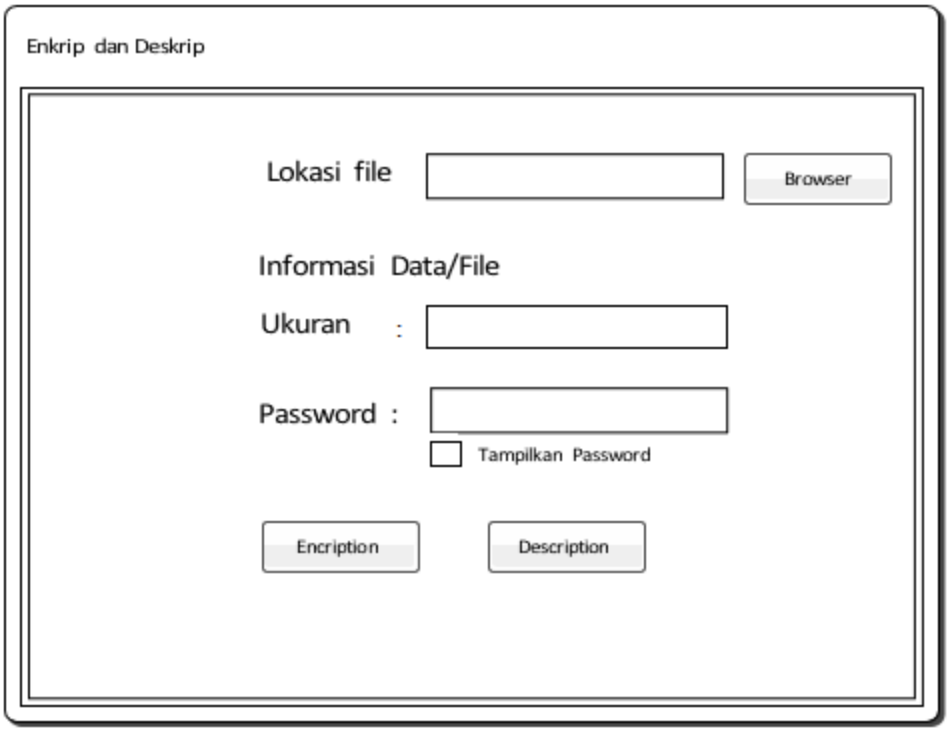
Gambar III.17. Desain Form *Login*

Penjelasan :

Pada form login, sistem akan menampilkan menu login seperti username (teks), password (teks), user wajib mengisi data tersebut apabila ingin masuk ke sistem setelah mengisi data kemudian klik login (button).

2. Desain Menu utama

Tampilan sistem yang dilakukan oleh User pada form menu utama dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada gambar III.18 berikut:



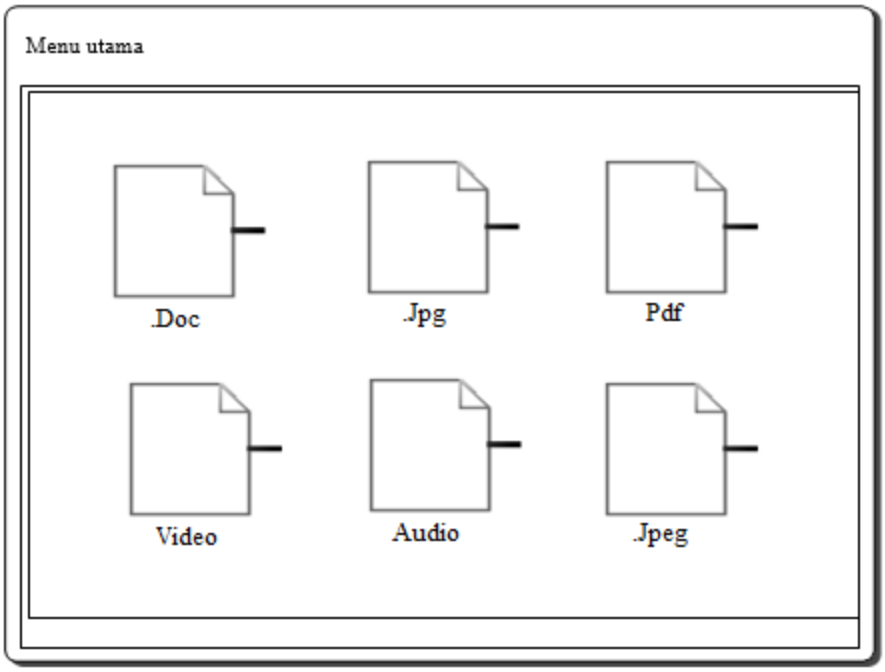
Gambar III.18. Desain Form Menu utama

Penjelasan:

Pada form menu utama, sistem akan menampilkan menu utama yang memiliki beberapa pilihan button atau menu untuk menjalankan aplikasi.

3. Desain Form Pilih file/browser

Tampilan sistem yang dilakukan oleh User pada form Pilih file/browser dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada gambar III.19 berikut:



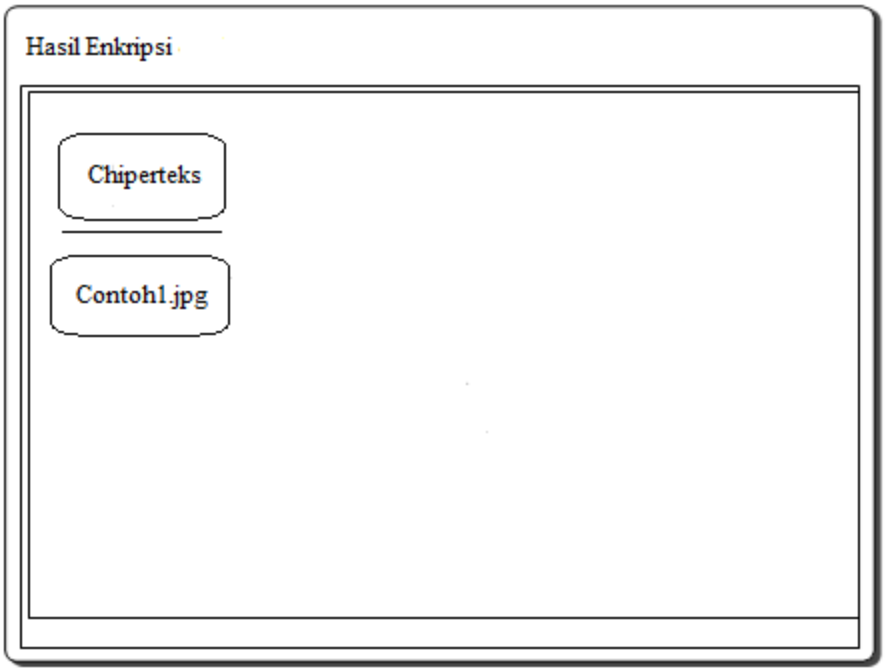
Gambar III.19. Desain Form Pilih File/Browser

Penjelasan:

Pada form menu pilih file/browser, sistem akan menampilkan menu browser yang memiliki beberapa pilihan data yang akan di enkripsi seperti Dokumen, Foto, dan lain-lain.

4. Desain Form Hasil Enkripsi File

Tampilan sistem yang dilakukan oleh User pada form Enkripsi File dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada gambar III.20 berikut:

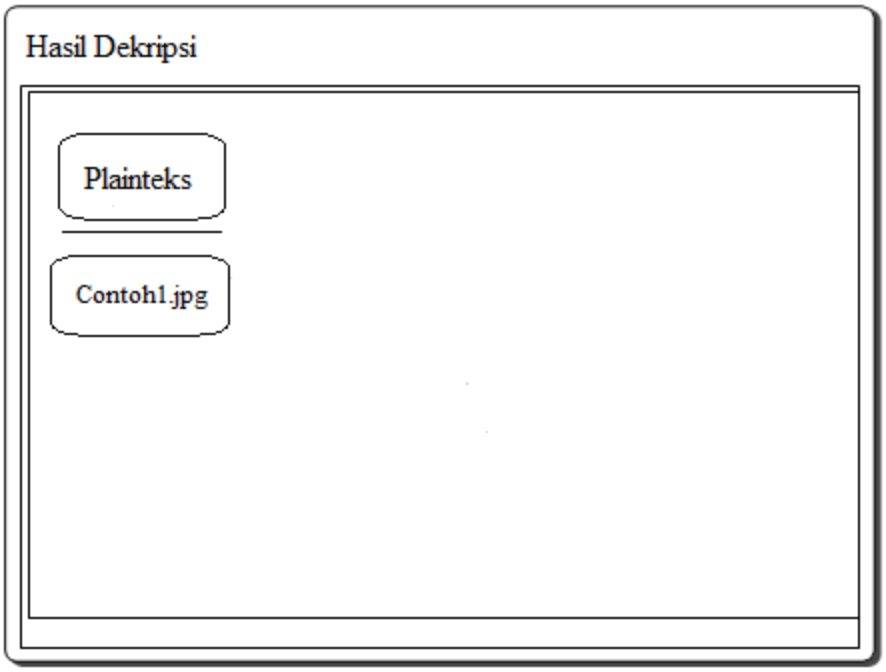


Gambar III.20. Desain Form Hasil Enkripsi

Pada form hasil Enkripsi File, sistem akan menampilkan hasil enkripsi yang otomatis akan tesimpan kedalam folder pada komputer. Dalam hal ini, hasil dari file yang telah dienkripsi akan otomatis tersimpan pada folder dimana awal sebelum file di enkripsi.

5. Desain Form Hasil *Dekripsi* File

Tampilan sistem yang dilakukan oleh user pada form *Dekripsi* File dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada gambar III.21 berikut:

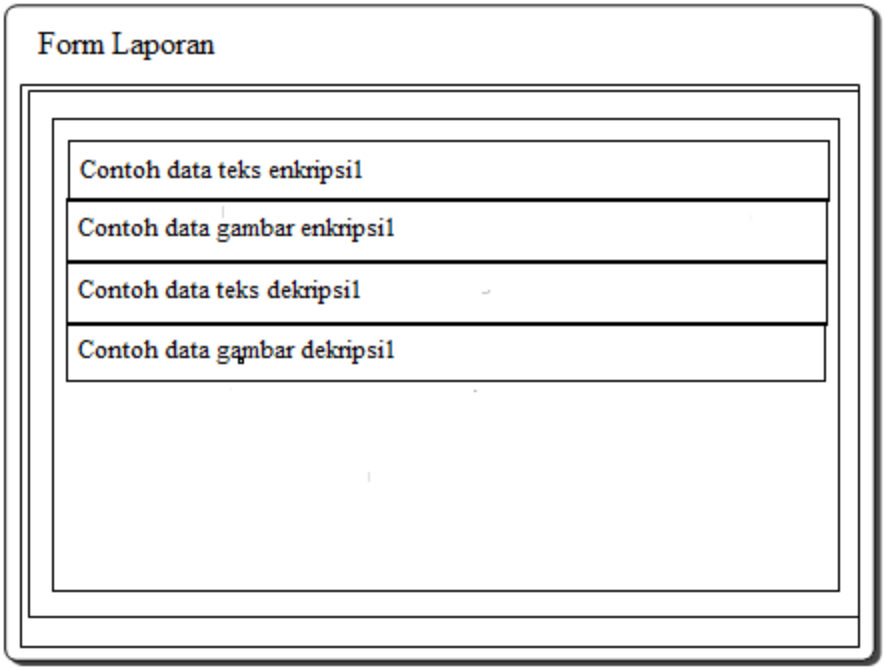


Gambar III.21 Desain Form Hasil *Dekripsi*

Pada form hasil *Dekripsi* File, sistem akan menampilkan hasil *Dekripsi* yang otomatis akan tesimpan kedalam folder pada komputer. Sama halnya dengan proses enripsi, hasil dari file yang telah *Dekripsi* akan otomatis tersimpan pada folder dimana awal sebelum file di *Dekripsi.*

6. Desain Form Laporan

Tampilan sistem yang dilakukan oleh user pada form Laporan dapat diterangkan dengan langkah-langkah *state* berikut, yang ditunjukkan pada gambar III.22 berikut:



Gambar III.22 Desain Form Laporan