BAB III
ANALISIS DAN DESAIN SISTEM

Pada bab ini akan dibahas mengenai Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*) yang meliputi analisa sistem dan desain sistem.

III.1. Analisis Masalah

Adapun analisa masalah pada Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*) yaitu :

1. Banyaknya pihak-pihak yang melakukan ancaman modifikasi mengakibatkan kesalahan dalam penerimaan informasi sehingga informasi yang diterima tidak sesuai dengan keinginan penerima maupun pengirimnya.
2. Terjadinya interuksi yang dapat mengganggu ketersediaan data yaitu data yang ada dapat dihapus sehingga pihak yang membutuhkan informasi tersebut tidak dapat menemukan datanya.
3. Seringnya terjadi ancaman intersepsi yaitu merupakan ancaman terhadap kerahasiaan data.

III.2. Algoritma AES

AES (*Advanced Encryption Standard*) adalah *cipher* blok yang akan menggantikan DES. Pada bulan Januari 1997 inisiatif AES pada bulan September 1997 publik diundang untuk mengajukan proposal *block cipher* yang cocok sebagai kandidat untuk AES. Pada tahun 1999 NIST mengumumkan lima kandidat finalis yaitu MARS, RC6, Rijndael, Serpent, dan Twofish. Algoritma AES dipilih pada bulan Oktober 2001 dan standarnya diperkenalkan pada bulan November 2002. AES mendukung ukuran kunci 128 bit, 192 bit, dan 256 bit, berbeda dengan kunci 56-bit yang ditawarkan DES. Algoritma AES dihasilkan dari proses bertahun-tahun yang dipimpin NIST dengan bimbingan dan *review* dari komunitas internasional pakar kriptografi.

III.2.1 Enkripsi

Enkripsi adalah proses mengubah suatu pesan asli (*plaintext*) menjadi suatu pesan dalam bahasa sandi (*ciphertext*).

C = E (M) .....(1)

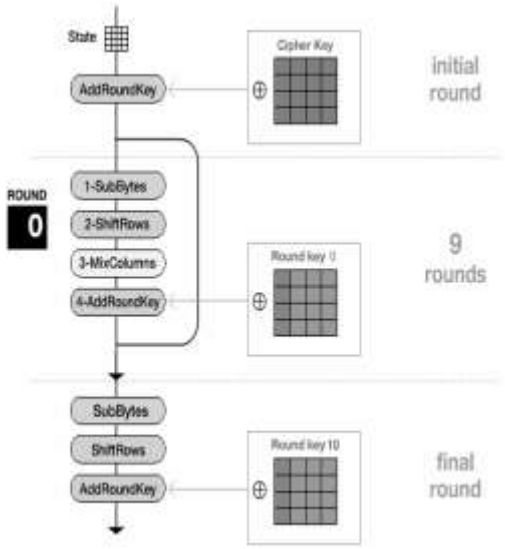
Keterangan :

M = Pesan asli (*Plaintext*)

E = Proses enkripsi dengan *Key Private*

C = *Chipertext* (*Plaintext* yang terenkripsi AES)

Dibawah ini merupakan gambar diagram proses enkripsi seperti yang ditunjukkan pada Gambar III.1.



Gambar III.1 Diagram Proses Enkripsi

Garis besar Algoritma AES *Rijndael* yang beroperasi pada blok 128-bit dengan kunci 128-bit adalah sebagai berikut (di luar proses pembangkitan *round key*):

1. *AddRoundKey*: melakukan *XOR* antara *state* awal (*plainteks*) dengan *cipher key*. Tahap ini disebut juga *initial round***.**
2. *Round :* Putaran sebanyak *Nr* – 1 kali. Proses yang dilakukan pada setiap putaran adalah:
   a) *SubBytes*: substitusi *byte* dengan menggunakan table substitusi (*S-box*).
   b) *ShiftRows*: pergeseran baris-baris *array state* secara *wrapping*.

c) *MixColumns*: mengacak data di masing- masing kolom *array state.*

d) *AddRoundKey*: melakukan XOR antara *state*

sekarang *round key*.

3. *Final round***:** proses untuk putaran terakhir:

a) *SubBytes*

b) *ShiftRows*

c) *AddRoundKey*

Langkah kerja dari enkripsi adalah sebagai berikut,

1. Transformasi *SubBytes()*

Transformasi *SubBytes()* memetakan setiap byte dari *array state* dengan menggunakan tabel.

hex		y															
		0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
x	0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
	1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
	2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
	3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
	4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
	5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
	6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
	7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
	8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
	9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
	a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
	b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
	c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
	d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
	e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
	f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

S-BOX

Gambar III.2. Tabel S-BOX

2. Transformasi *ShiftRows()*

Melakukan pergeseran secara *wrapping* (siklik) pada 3 baris terakhir dari array state. Jumlah pergeseran bergantung pada nilai baris (r). Baris *r* = 1 digeser sejauh 1 *byte*, baris r = 2 digeser sejauh 2 *byte*, dan baris *r* = 3 digeser sejauh 3 *byte*. Baris *r* = 0 tidak digeser.

Contoh ditunjukkan pada Gambar III.3 berikut. Geser baris ke-1:

Hasil pergeseran baris ke-1 dan geser baris ke-2:

Hasil pergeseran baris ke-2 dan geser baris ke-3:

Hasil pergeseran baris ke-3:

Gambar III.3. Contoh Transformasi *ShiftRows()*.

3. Transformasi *MixColumns()*

Transformasi *MixColumns()* mengalikan setiap kolom dari *array state* dengan polinom a(x) mod (x^4 + 1). Setiap kolom diperlakukan sebagai polinom 4- suku pada GF(2^8). a(x) yang ditetapkan adalah : a(x) = {03}x^3 + {01}x^2 + {01}x + {02}.

Transformasi ini dinyatakan sebagai perkalian matriks :

$$s'(x) = a(x) \otimes s(x)$$

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{3,c})$$

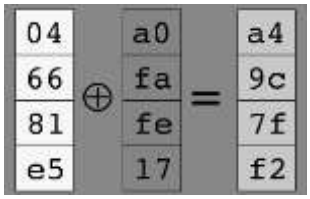
$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{3,c})$$

4. Transformasi *AddRoundKey()*

Transformasi ini melakukan operasi XOR terhadap sebuah *round key* dengan *array state*, dan hasilnya disimpan di *array state*.

Contoh:

XOR-kan kolom pertama *state* dengan kolom pertama *round key* :



Hasil *AddRoundKey()* terhadap seluruh kolom:

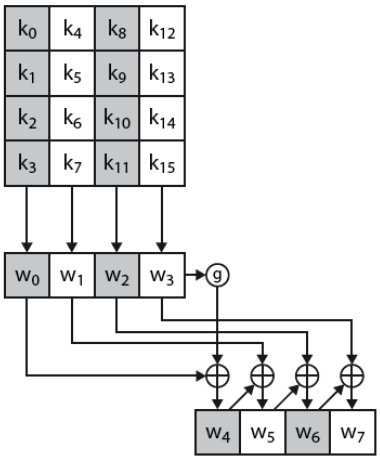
a4	68	6b	02
9c	9f	5b	6a
7f	35	ea	50
f2	2b	43	49

Gambar III.4. Contoh Transformasi *AddRoundKey()*

5. Ekspansi Kunci (*Key Expansion*)

Ekspansi kunci pada AES 128-bit (16-byte) menggunakan 4-*words* (16 byte) sebagai input dan menghasilkan perluasan kunci menjadi 44 *words* (176 bytes).

Gambar III.5 menunjukkan proses dari *key expansion().*



Gambar III.5. Proses *Key Expansion()*

III.2.2 Dekripsi

Dekripsi adalah proses mengubah pesan dalam suatu bahasa sandi menjadi pesan asli kembali.

M = D (C) .....(2)

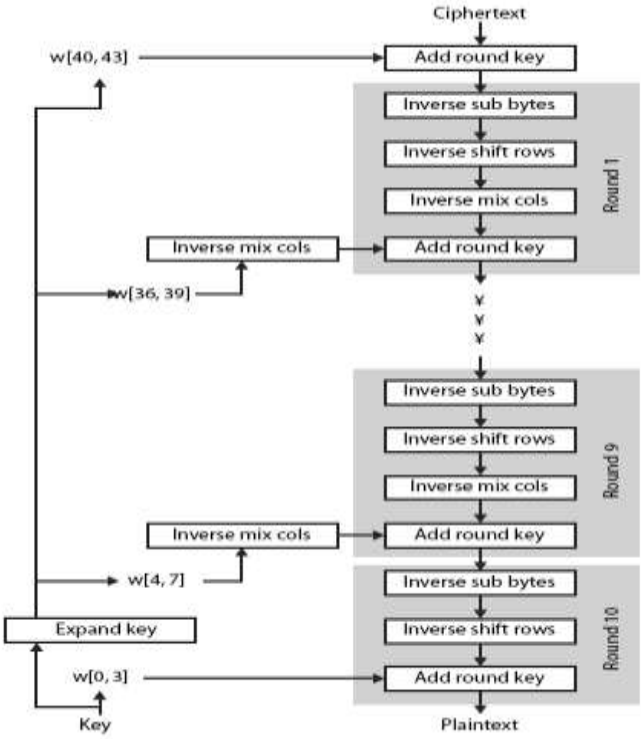
Ket :

C = *Chipertext* (Hasil *Plaintext* terenkripsi)

D = Proses dekripsi menggunakan *key private*

M = Pesan asli setelah di dekripsi

Gambar III.6 menunjukkan proses dari sebuah dekripsi pesan.



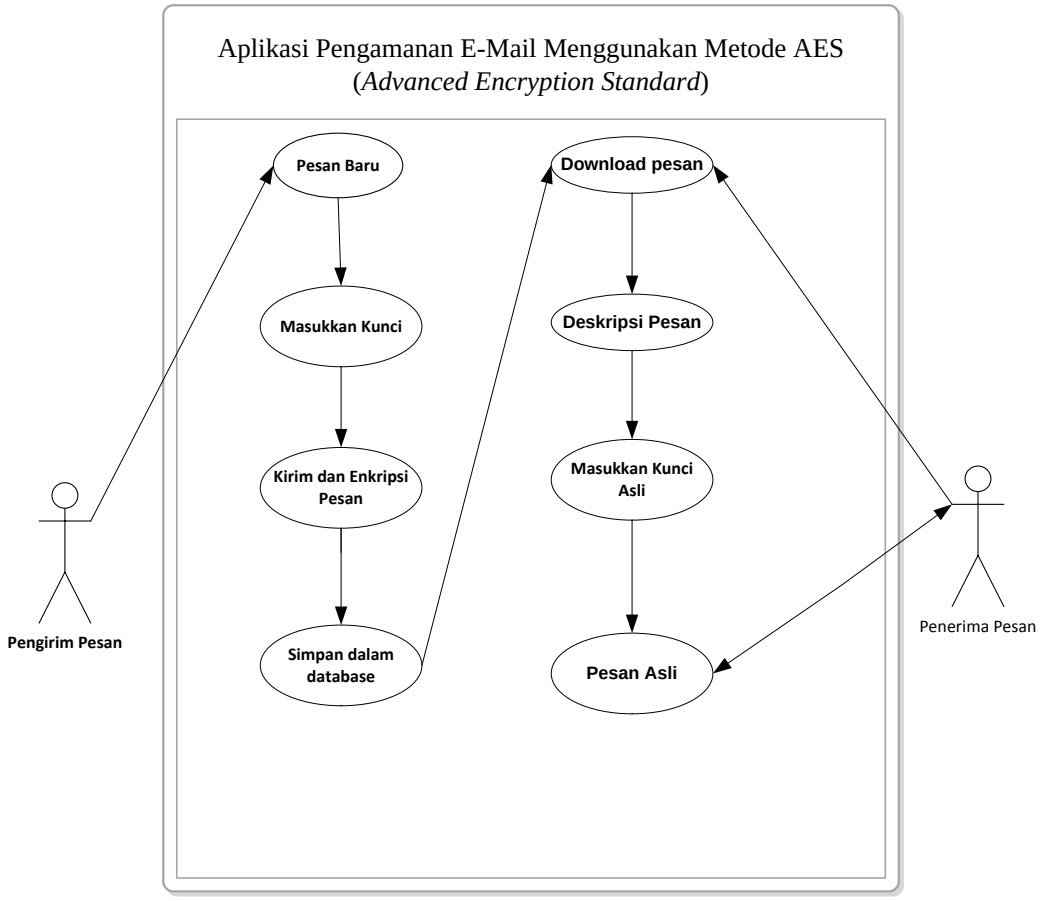
Gambar III.6. Diagram Alur Proses Dekripsi

III.3. Desain Sistem Baru

Desain Sistem Baru menggunakan bahasa pemodelan UML yang terdiri dari *Usecase Diagram*, *Class Diagram*, *Activity Diagram* dan *Sequence Diagram*.

III.3.1. *Use case* Diagram

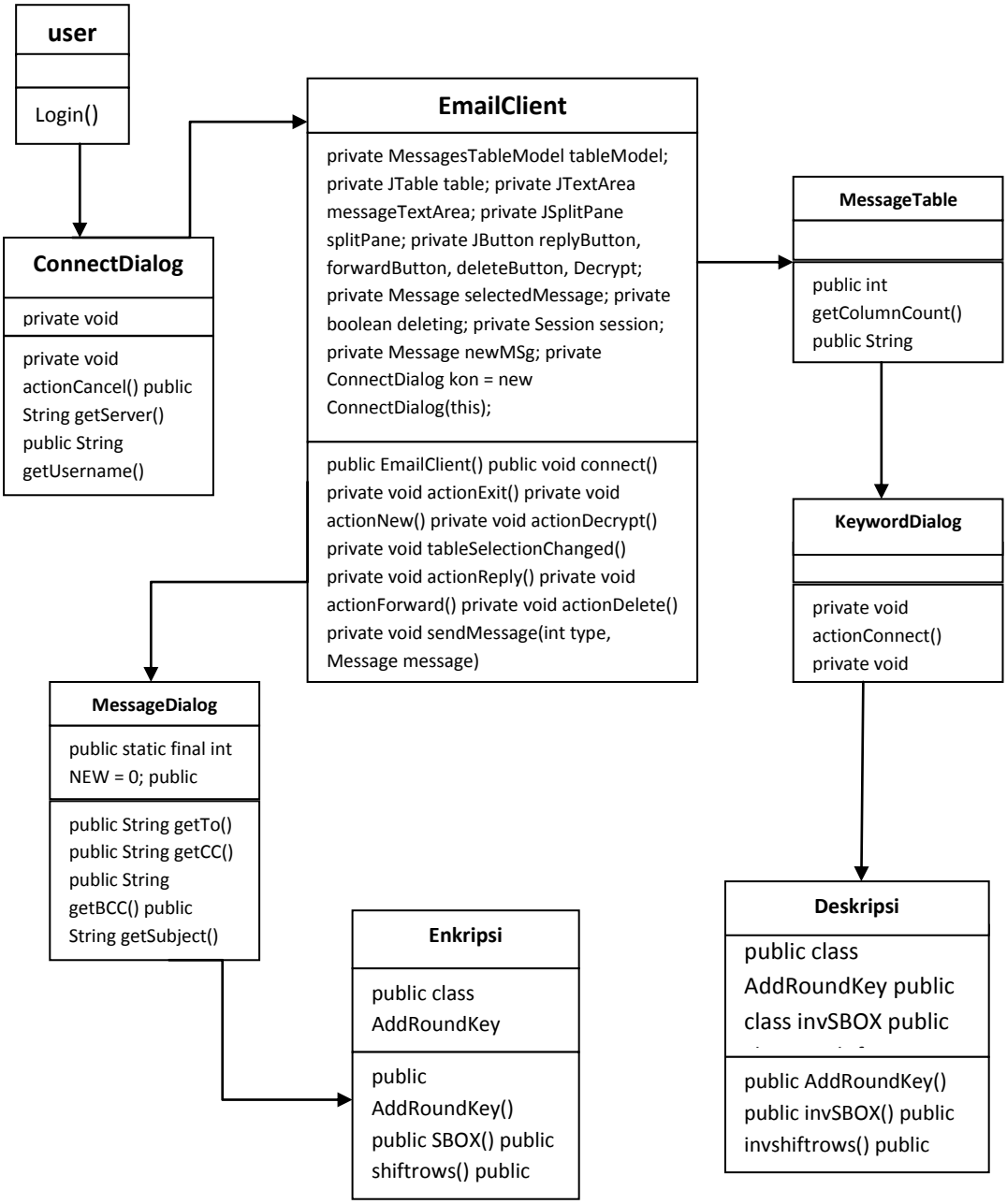
Secara garis besar, proses sistem yang akan dirancang digambarkan dengan *usecase diagram* yang terdapat pada Gambar III.7 :



Gambar III.7. *Use Case* Diagram Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*)

III.3.2.*Class* Diagram

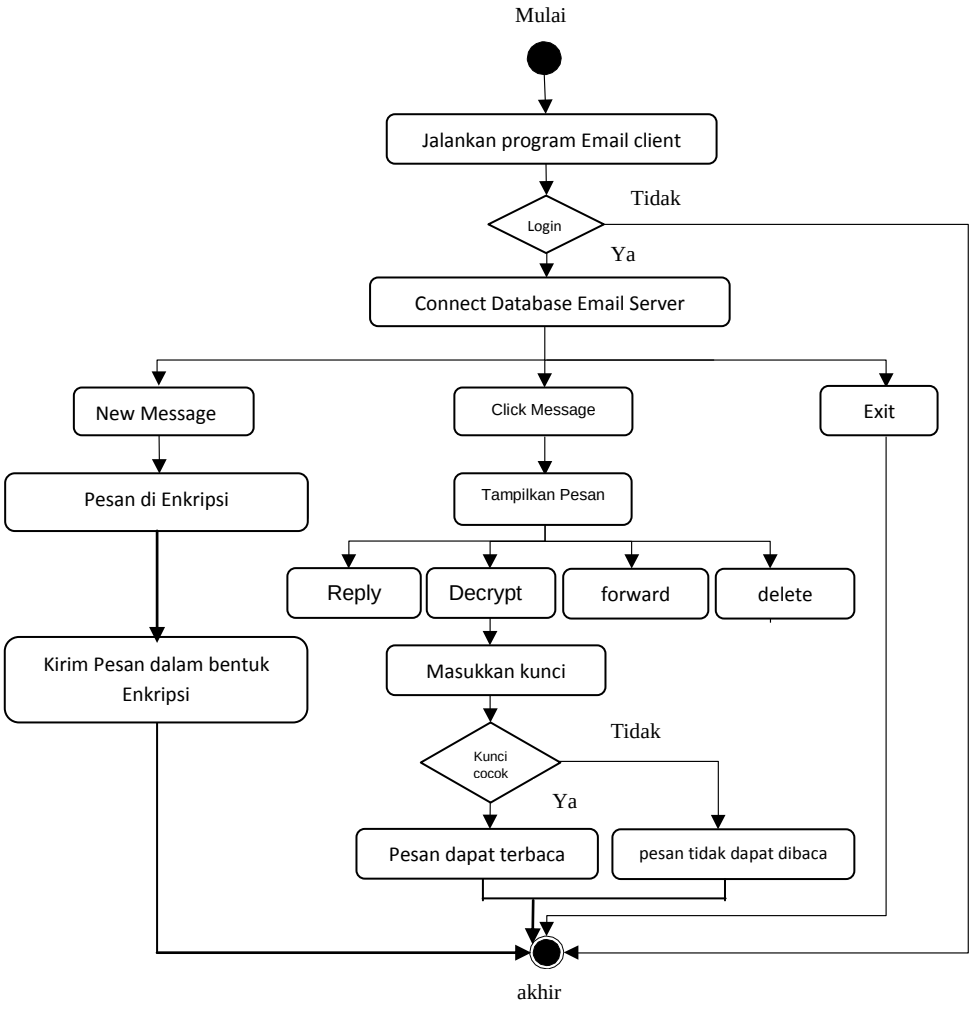
Rancangan kelas-kelas yang akan digunakan pada sistem yang akan dirancang dapat dilihat pada gambar III.8 :



Gambar III.8. *Class Diagram* Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*)

III.3.3.*Activity* Diagram

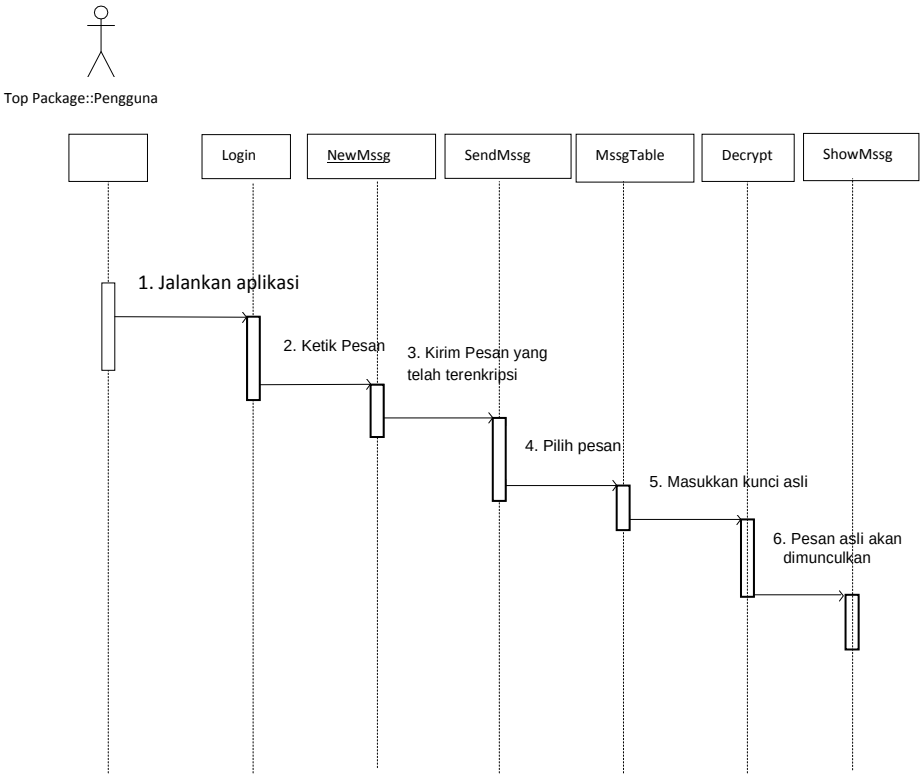
Diagram aktivitas menggambarkan suatu urutan proses yang terjadi pada sistem dari dimulainya aktivitas hingga aktivitas berhenti. Diagram aktivitas hampir mirip dengan diagram *flowchart*. Diagram aktivitas merupakan salah satu cara untuk memodelkan *event-event* yang terjadi dalam suatu *use-case*. Berikut *activity* diagram yang ditunjukkan pada gambar III.9:



Gambar III.9. *Activity Diagram* Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*)

III.3.4. *Sequence* Diagram

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, display, dan sebagainya) berupa message yang digambarkan terhadap waktu. *Sequence* diagram terdiri atas dimensi vertikal (waktu) dan dimensi horizontal (objek-objek yang terkait). Serangkaian kegiatan saat terjadi *event* pada aplikasi ini dapat dilihat pada gambar III.10:

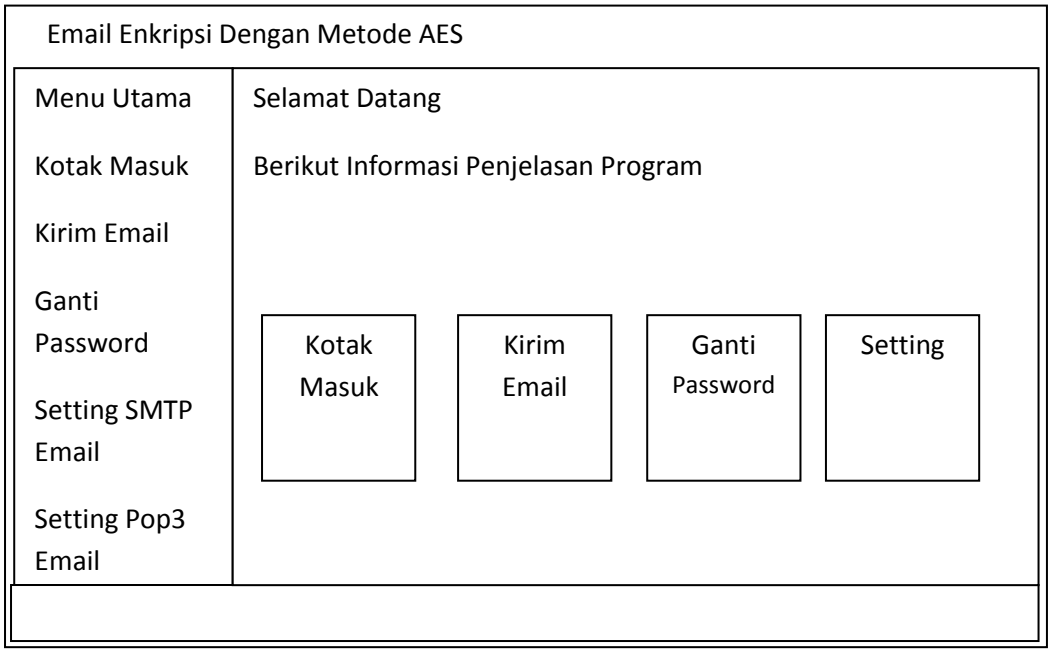


Gambar III.10. *Sequence* Diagram Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*)

III.4. Desain User Interface

1. Rancangan *Form* Menu Utama

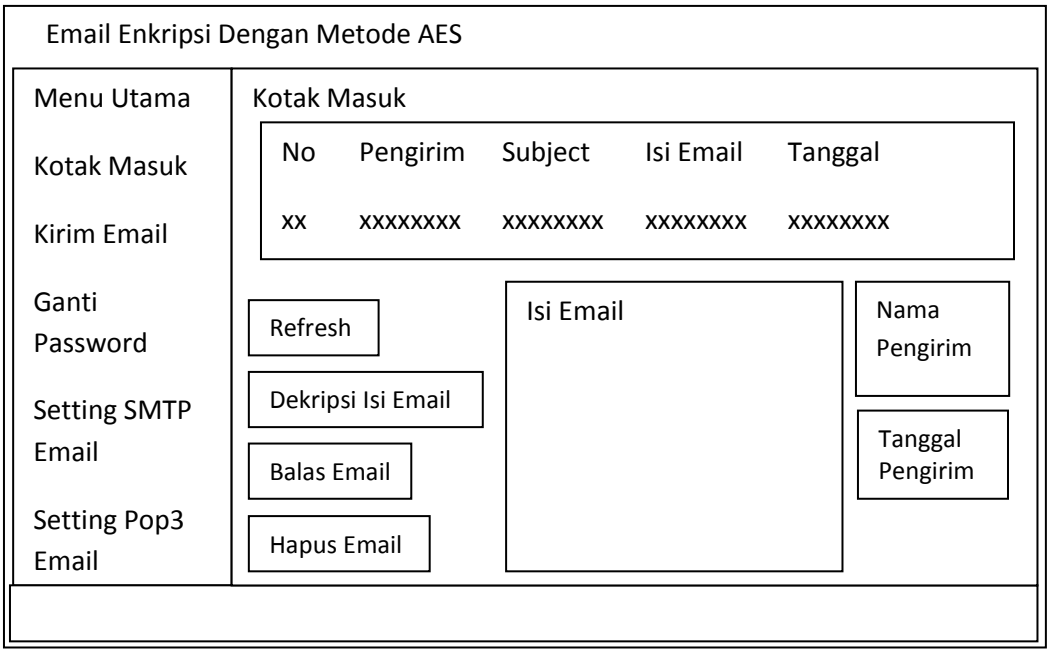
Form ini berfungsi untuk menampilkan Menu Utama Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*), rancangan dapat dilihat pada gambar berikut :



Gambar III.11. Desain Tampilan Menu Utama

2. Rancangan *Form* Kotak Masuk

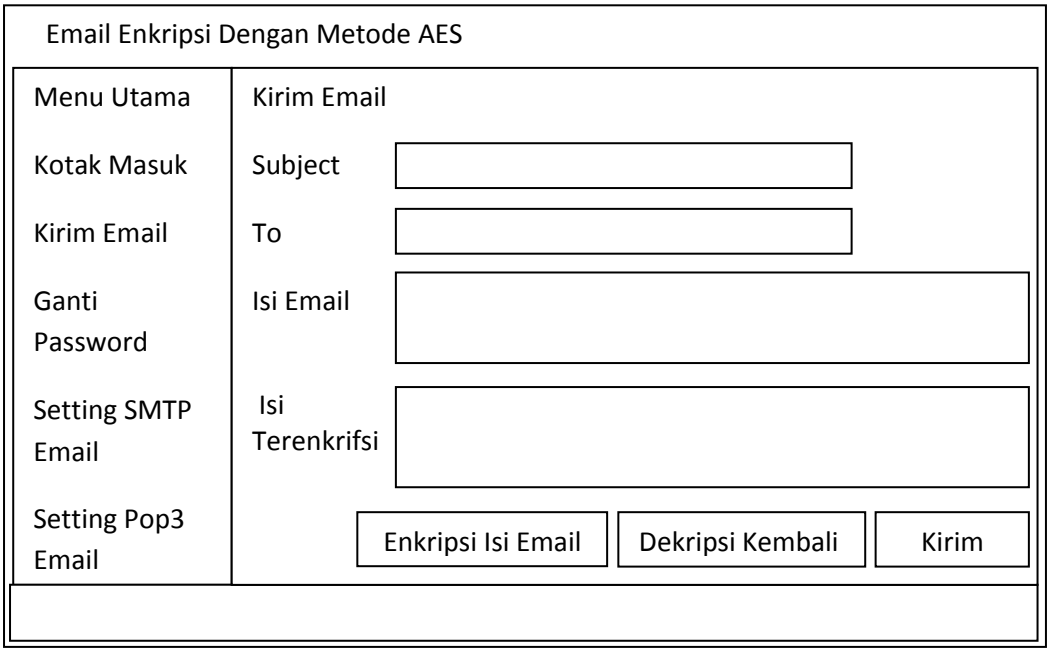
Form Kotak Masuk berfungsi untuk menampilkan *form* Kotak Masuk Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*), rancangan dapat dilihat pada gambar berikut :



Gambar III.12. Desain Tampilan *Form* Kotak Masuk

3. Rancangan *Form* Kirim Email

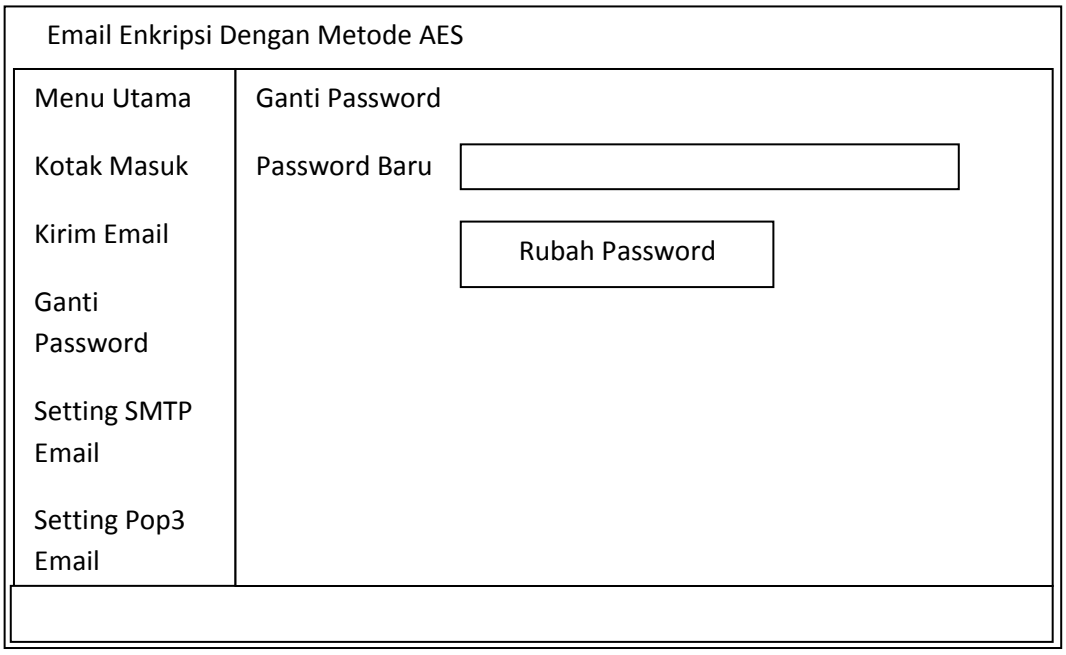
Form Kirim Email berfungsi untuk menampilkan *form* Kirim Email Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*), rancangan dapat dilihat pada gambar berikut :



Gambar III.13. Desain Tampilan *Form* Kirim Email

4. Rancangan *Form* Ganti Password

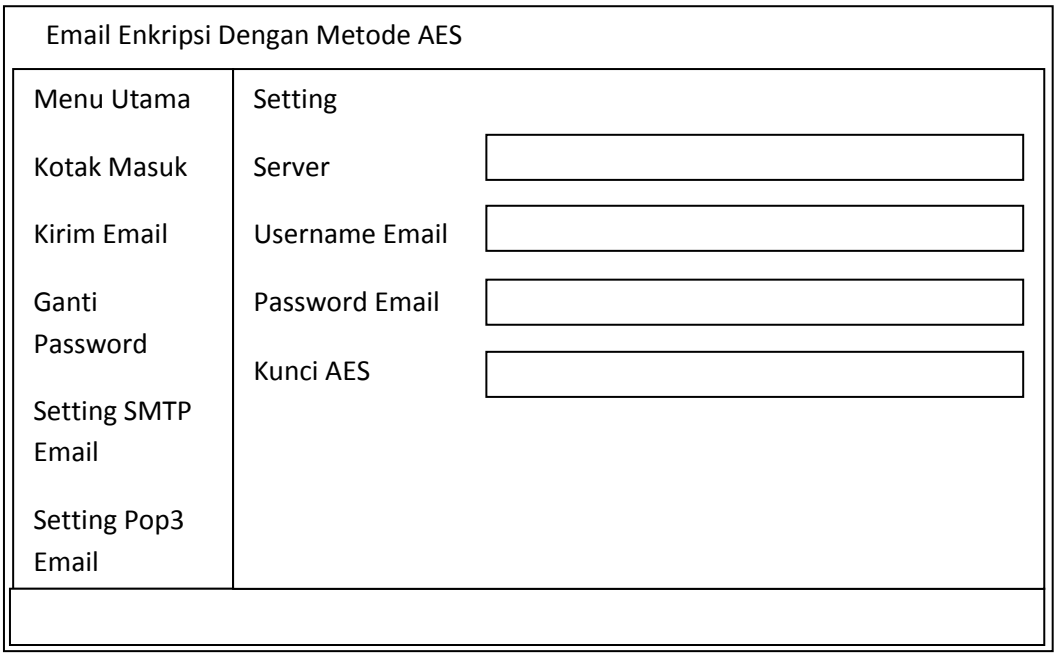
Form ini menjelaskan informasi ganti Password Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*), rancangan dapat dilihat pada gambar berikut :



Gambar III.14. Desain Tampilan Ganti Password

5. Rancangan *Form* Setting

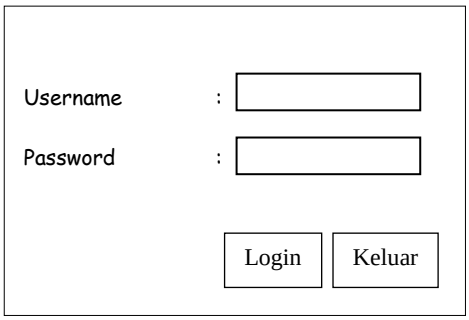
Form ini menjelaskan Setting Aplikasi Pengamanan E-Mail Menggunakan Metode AES (*Advanced Encryption Standard*), rancangan dapat dilihat pada gambar berikut :



Gambar III.15. Desain Tampilan Setting.

6. Rancangan *Form* Login

Pada *Form* ini menggambarkan proses *login* pada sistem. Tampilan *Login* ditunjukan pada gambar III.16 berikut ini :



Gambar III.16.*Login* Pakar

III.4.1. Kamus Data

Dibawah ini adalah kamus data atau referensi data yang ada pada basis data sistem yang akan dibangun :

1. user ={(nama_**email** + password)}
2. email ={(no +nama_email +subject + isi_email+tanggal)}

Normalisasi *database* biasanya jarang dilakukan dalam *database* skala kecil dan dianggap tidak diperlukan pada penggunaan personal. Namun seiring dengan berkembangnya informasi yang dikandung dalam sebuah *database*, proses normalisasi akan sangat membantu dalam menghemat ruang yang digunakan oleh setiap tabel di dalamnya, sekaligus mempercepat proses permintaan data. Pada tahap ini semua data direkam tanpa*format*tertentu dan data bisa jadi mengalami duplikasi.

1. Bentuk Normal Pertama (1NF/ *First Normal Form*)
   a. Tabel Normal Pertama

Nama_email	password	No	Subject	Isi_email	Tanggal

2. Bentuk Normal Kedua (2NF/ *Second Normal Form*)
   a. Tabel email

Nama_email	no	Subject	Isi_email	Tanggal

b. Tabel user

Nama_email	Password

3. Bentuk Normal Ketiga (3NF/ *Third Normal Form*)

a. Tabel email

Nama_email*	no	Subject	Isi_email	tanggal

b. Tabel user

Nama_email*	Password

III.4.2. Desain Tabel/ File

Pada sistem pakar ini, digunakan *database SQL Server* dengan namadbeko menggunakan 2 tabel, yaitu tabel user dan tabel email . Adapun struktur data dari tabel-tabel tersebut adalah sebagai berikut :

III.4.2.1. Struktur Tabel email

Tabel email digunakan untuk menyimpan *record* data email masuk. Tabel email ditunjukkan pada tabel III.1 berikut ini ;

Tabel III.1. Struktur Tabel Email

Field	*Type*	*Size*	Keterangan
No	*Varchar*	10	*Primary key*
Nama_email	*Varchar*	10	*foreign key*
Subject	*Varchar*	250	-
Isi_email	*Varchar*	250	-
Tanggal	*Date*	-	-

III.4.2.2. Struktur Tabel user

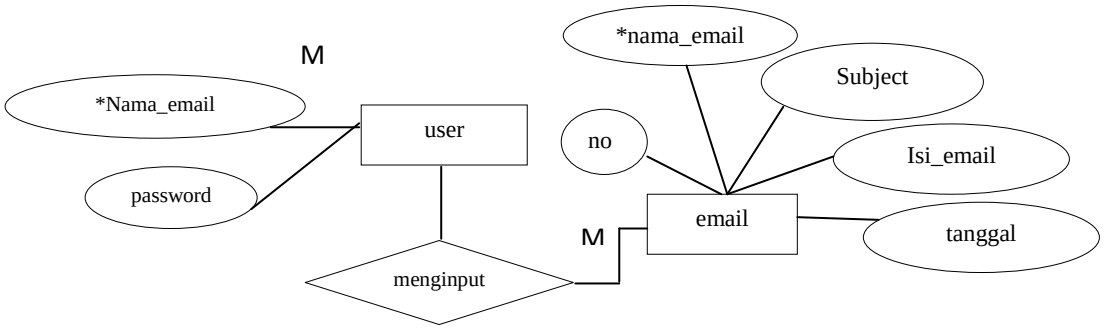
Tabel admin digunakan untuk menyimpan *record* data user dengan properti atribut name_email dan password. Tabel admin ditunjukkan pada tabel III.2 berikut ini ;

Tabel III.2. Struktur Tabel Admin

Field	*Type*	*Size*	Keterangan
Nama_email	*Varchar*	250	*Primary key*
Password	*Varchar*	250	-

III.4.3. ERD (*Entity Relationship Diagram*)

Adapun ERD (*Entity Relationship Diagram*) dari aplikasi yang akan di bangun ditunjukkan pada gambar III.17 berikut ini:



Gambar III.17. ERD (*Entity Relationship Diagram*)