BAB III

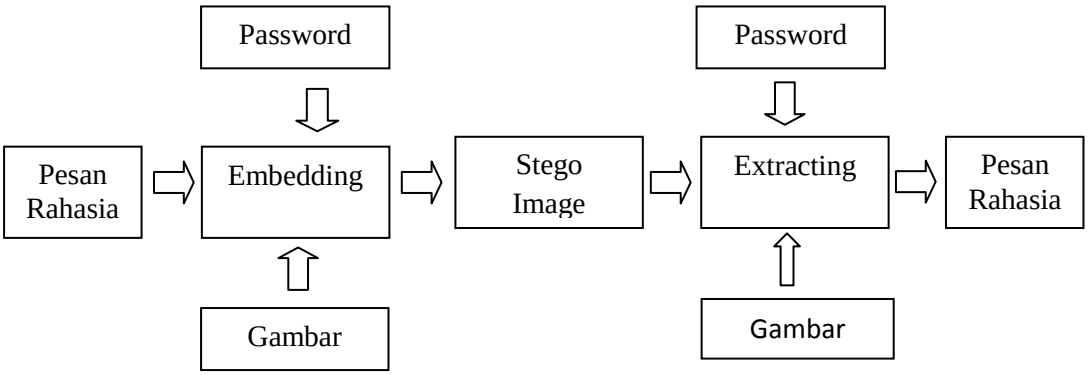
ANALISIS MASALAH DAN RANCANGAN PROGRAM

III.1. Analisa Masalah

Perancangan umum di dalam bab ini akan membahas mengenai perancangan dan pembuatan skripsi yang berjudul Perancangan Aplikasi Steganografi penyisipan data kedalam gambar dengan Metode LSB (*Least Significant Bit*) dan Vigenere Cipher. Proses penyisipan data/pesan pada steganografi membutuhkan dua buah masukan, yaitu data/pesan yang ingin disembunyikan, dan media penyisipan yaitu gambar. Gambar yang dimaksudkan disini adalah sebagai media penampung dari pesan rahasia yang akan kita sisipkan/sembunyikan. Gambar yang biasa juga dinamakan file image digunakan dalam pembuatan aplikasi ini hanya berbentuk format *bitmap* (.bmp). File image yang berfungsi sebagai wadah penampung teks yang akan di rahasiakan di dalam image tersebut. Steganografi merupakan suatu teknik berkomunikasi dimana informasi disembunyikan pada media pembawa seperti citra, suara atau video tanpa memberikan perubahan yang berarti pada media tersebut.

Perangkat lunak yang dirancang ini menggunakan teknik image steganografi dengan metode LSB dan Vigenere Cipher yang membutuhkan image yang berformat *bitmap* (.bmp) digunakan sebagai wadah penampung pesan yang akan dirahasiakan dan pesan yang akan dirahasiakan dalam image yang berupa teks. Proses modifikasi perubahan yang terjadi antara media penampung dengan hasil modifikasi media penampung tersebut secara kasat mata tidak ada perubahan

yang mencolok. Sehingga untuk menghasilkan sebuah objek digital yang telah dimodifikasi yang didalamnya terdapat suatu pesan rahasia yang disebut dengan istilah stego image. Adapun perancangan umum dari sistem aplikasi steganografi yang akan dibuat seperti yang ditampilkan pada gambar III.1.



Gambar III.1. Perancangan Umum Sistem

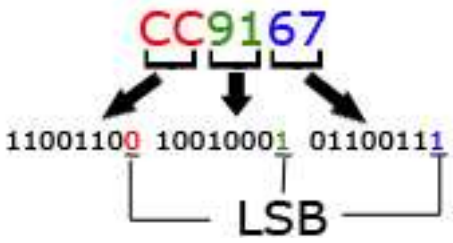
III.2. Strategi Pemecahan Masalah

Didalam hal penyelesaian masalah penyisipan pesan ke dalam gambar yang akan diterapkan dalam aplikasi ini, agar proses berjalan dengan baik dan hasil yang diinginkan sesuai dengan yang diharapkan. Maka untuk menyelesaikan masalah yang terjadi dalam proses pembuatan aplikasi ini, penulis menerapkan beberapa strategi yaitu dengan membuat bagan alir program yang menyangkut semua alir program mulai dari awal program sampai akhir program. Selanjutnya, penulis menganalisa tentang bagaimana cara untuk membuat suatu aplikasi ini lebih efektif dan efisien sehingga aplikasi ini dapat digunakan. Penulis mengumpulkan teori-teori yang berhubungan dengan masalah perancangan aplikasi ini, dan dari buku-buku, artikel yang didapat dari internet. Dan bahasa

pemograman yang penulis pilih dalam implementasi rancangan program adalah bahasa pemograman Visual Basic 2010.

III.3. Analisa Metode Steganografi

Teknik dalam penyisipan sebuah data dalam gambar ini menggunakan metode LSB (Least Significant Bit) dan Vigenere Cipher. Pada metode LSB data akan disisipkan pada bit-bit rendah (4 bit bawah) tetapi pada bit-bit ganjil dari nilai RGB sebuah pixel (satuan data terkecil yang membentuk gambar). Model warna RGB merupakan gambar yang tiap pikselnya direpresentasikan oleh kombinasi intensitas tiga buah komponen warna, yaitu *Red* (R), *Green* (G), dan *Blue* (B). Masing-masing komponen tersebut terdiri atas 8 bit. Sebuah warna dalam piksel gambar RGB dapat direpresentasikan dalam tiga *byte* atau 24-bit. Kombinasi ketiga intensitas warna tersebut dapat menghasilkan 16 juta (224 = 16.777.216) variasi warna (Curran & Bailey 2003). Pada Gambar III.2 dibawah terlihat bit-bit LSB pada 1 piksel warna, penanaman informasi dapat dilakukan pada bit-bit tersebut.



Gambar III.2 Metode LSB (Least Significant Bit)

Adapun tabel konversi biner 8 bit dapat dilihat pada tabel III.1.

Tabel III.1. Tabel kode ASCII 8 bit

Letter	ASCII Kode	Binary	Letter	ASCII Kode	Binary
a	097	01100001	A	065	01000001
b	098	01100010	B	066	01000010
c	099	01100011	C	067	01000011
d	100	01100100	D	068	01000100
e	101	01100101	E	069	01000101
f	102	01100110	F	070	01000110
g	103	01100111	G	071	01000111
h	104	01101000	H	072	01001000
i	105	01101001	I	073	01001001
j	106	01101010	J	074	01001010
k	107	01101011	K	075	01001011
l	108	01101100	L	076	01001100
m	109	01101101	M	077	01001101
n	110	01101110	N	078	01001110
o	111	01101111	O	079	01001111
p	112	01110000	P	080	01010000
q	113	01110001	Q	081	01010001
r	114	01110010	R	082	01010010
s	115	01110011	S	083	01010011
t	116	01110100	T	084	01010100
u	117	01110101	U	085	01010101
v	118	01110110	V	086	01010110
w	119	01110111	W	087	01010111
x	120	01111000	X	088	01011000
y	121	01111001	Y	089	01011001
z	122	01111010	Z	090	01011010

Pada metode LSB menerapkan penempatan bit-bit yang akan disisipkan ke dalam file gambar dengan susunan yang terstruktur dan terletak di angka akhir setiap bit pada file gambar (citra cover). Metode LSB ini rentan untuk dimanipulasi dan diserang karena susunan bit yang mudah dibaca. Dengan penggunaan Vigenere Cipher, bit-bit yang disisipkan akan diacak dan menyulitkan penyerang untuk mendeteksi letak bit file sisipan. Adapun proses pada metode Vigenere Cipher, file gambar dan pesan diubah menjadi deretan bit, selanjutnya digunakan password yang berfungsi sebagai seed key untuk

membangkitkan PRNG (*Pseudo Random Number Generator*). *Pseudo Random Number Generator* adalah algoritma yang membangkitkan deretan bilangan yang tidak benar-benar acak. Bilangan acak dihasilkan dengan rumus-rumus matematika dan dapat berulang kembali secara periodik. Bilangan acak banyak digunakan di dalam kriptografi, misalnya untuk pembangkitan elemen elemen kunci, pembangkitan *initialization vector* (IV), pembangkitan parameter kunci di dalam sistem kriptografi kunci publik, dan sebagainya. Beberapa algoritma PRNG yang sering digunakan di antaranya *Linear Congruential Generator* (LCG), *Lagged Fibonacci Generator*, *Linear Feedback Shift Register* (LFSR), Blum Blum Shub, Fortuna, dan Mersenne Twister (Munir 2006).

Linear Congruential Generator (LCG) adalah salah satu pembangkit bilangan acak tertua dan sangat terkenal. LCG didefinisikan dalam bentuk:

$$Xn = (aXn - 1 + b) \bmod m$$

yang dalam hal ini,

Xn = bilangan acak ke-n dari deretnya

$Xn - 1$ = bilangan acak sebelumnya

a = faktor pengali

b = *increment*

m = modulus

(a, b, dan m semuanya konstanta)

Kunci pembangkit adalah *X0* yang disebut umpan (*seed*). LCG mempunyai periode tidak lebih besar dari m. LCG mempunyai periode penuh $(m - 1)$ jika memenuhi syarat berikut:

1. *b* relatif prima terhadap *m*.
2. *a* – 1 dapat dibagi dengan semua faktorprima dari *m*.
3. *a* – 1 adalah kelipatan 4 jika *m* adalah kelipatan 4.
4. *m* > maks(*a*, *b*, *X0*).
5. *a* > 0, *b* > 0.

Keunggulan dari LCG adalah cepat dan hanya membutuhkan sedikit operasi bit. Akan tetapi, LCG tidak dapat digunakan untuk kriptografi karena bilangan acaknya dapat diprediksi urutan kemunculannya (Munir 2006).

Untuk melakukan penyisipan pesan ke dalam file citra, terlebih dahulu dilakukan penghitungan nilai *pixel* (Px) citra dengan rumus:

Nilai R = Px Mod 256

Nilai G = (Px \ 256) Mod 256

Nilai B = (Px \ 256 \ 256) Mod 256

Sebagai contoh nilai piksel (1,1) citra adalah 111100001111000011111111.

Nilai komponen *Red* (R) dilakukan perhitungan modulo dengan bilangan 256 dengan nilai ASCII 10000000 sebagai berikut:

Nilai komponen R dihitung dengan persamaaan (3.2) = nilai piksel Blok-1 *mod* 10000000 R = 111100001111000011111111 *mod* 10000000 R = 11111111 Nilai komponen *Green* (G) dihitung dengan persamaan (3.3): G = (111100001111000011111111\10000000) *mod* 100000000 G = 11110000 Nilai komponen Blue (B) dihitung dengan persamaan (3.4): B = (111100001111000011111111\10000000\10000000) mode 10000000 = B =

11110000 Sehingga diperoleh nilai RGB piksel citra frame-1 pada Blok-1 adalah: 11110000 11110000 11111111: R = 11111111= 254, G = 11110000 = 240 dan B = 11110000 = 240 Sehingga diperoleh nilai piksel (1,1) dengan komponen RGB = (254,240,240).

proses cara kerja perhitungan Vigenere Cipher

Rumus enkripsi vigenere cipher :

Pi = (Ci – Ki) mod 256

Atau

Ci = (Pi + Ki) – 256 kalau hasil penjumlahan ***Pi*** dan ***Ki*** lebih dari 256

Rumus dekripsi vigenere cipher :

Pi = (Ci – Ki) mod 256

Atau

Pi = (Ci – Ki) + 256 kalau hasil pengurangan ***Ci*** dengan ***Ki*** minus

Dimana:

Ci = nilai desimal karakter ciphertext ke-i

Pi = nilai desimal karakter plaintext ke-i

Ki = nilai desimal karakter kunci ke-i

Nilai desimal karakter:256 kode ascii

Sebagai contoh, jika plaintext adalah **UNIVERSITAS POTENSI UTAMA** dan kunci adalah **DARWAN** maka proses enkripsi yang terjadi adalah sebagai berikut:

Plaintext:	UNIVERSITAS POTENSI UTAMA
Key:	DARWAN
Ciphertext:	™□›-† —Š¦˜”n” • ¦œ□¡□a§«,›…

Pada contoh diatas kata kunci **DARWAN** diulang sedemikian rupa hingga panjang kunci sama dengan panjang plainteksnya. Jika dihitung dengan rumus enkripsi vigenere plainteks huruf pertama **U** (yang memiliki nilai **Pi=85**) akan dilakukan pergeseran dengan huruf **D** (yang memiliki **Ki=69**) maka prosesnya sebagai berikut:

Ci = (Pi + Ki) mod 256

= (85 + 69) mod 256

= 154 mod 256

= -102

Ci=2 maka huruf ciphertext dengan nilai **-102** adalah ™ . Begitu seterusnya dilakukan pergeseran sesuai dengan kunci pada setiap huruf hingga semua plainteks telah terenkripsi menjadi ciphertext. Setelah semua huruf terenkripsi maka proses dekripsinya dapat dihitung sebagai berikut:

Pi = (Ci – Ki) + 256

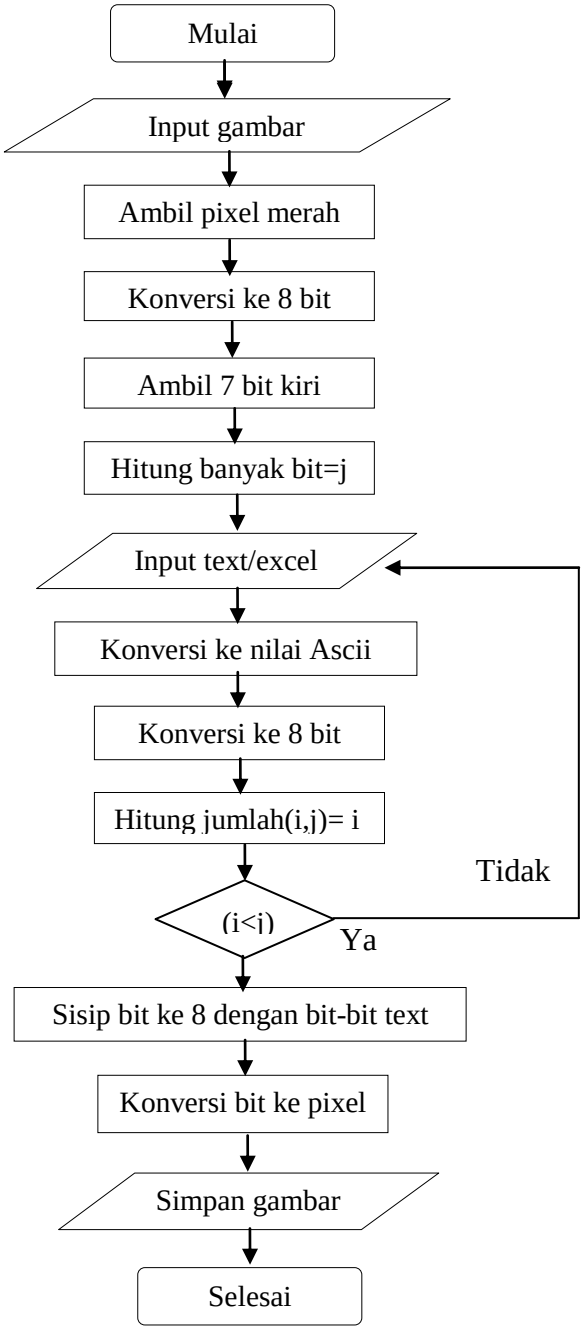
= (-102 – 69) + 256

= -171 + 256

= 85

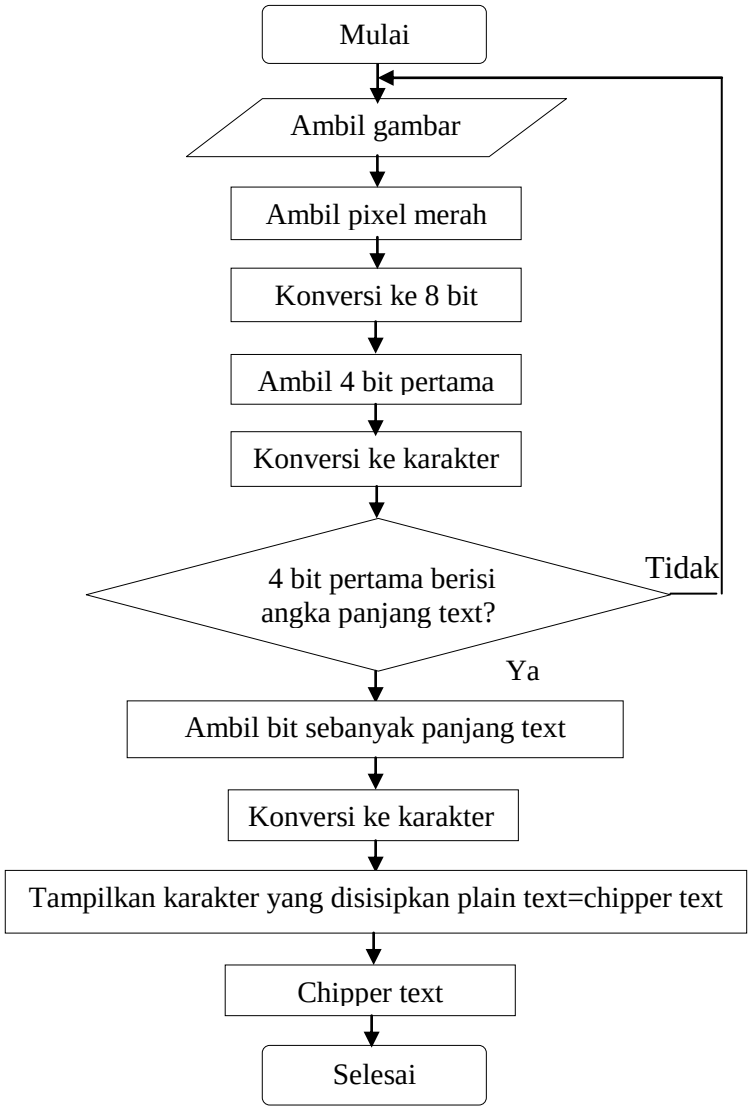
Pi=18 maka huruf plainteks dengan nilai **18** adalah **S**. Begitu seterusnya dilakukan pergeseran sesuai dengan kunci pada setiap huruf hingga semua ciphertext telah

terdekripsi menjadi plainteks. Pada metode LSB (Least Significant Bit) terdapat proses yaitu proses enkripsi penyisipan dan dekripsi. Berikut ini adalah diagram alir dari proses yang ditampilkan pada gambar III.3 dan gambar III.4



Gambar III.3. Diagram alir proses enkripsi penyisipan LSB

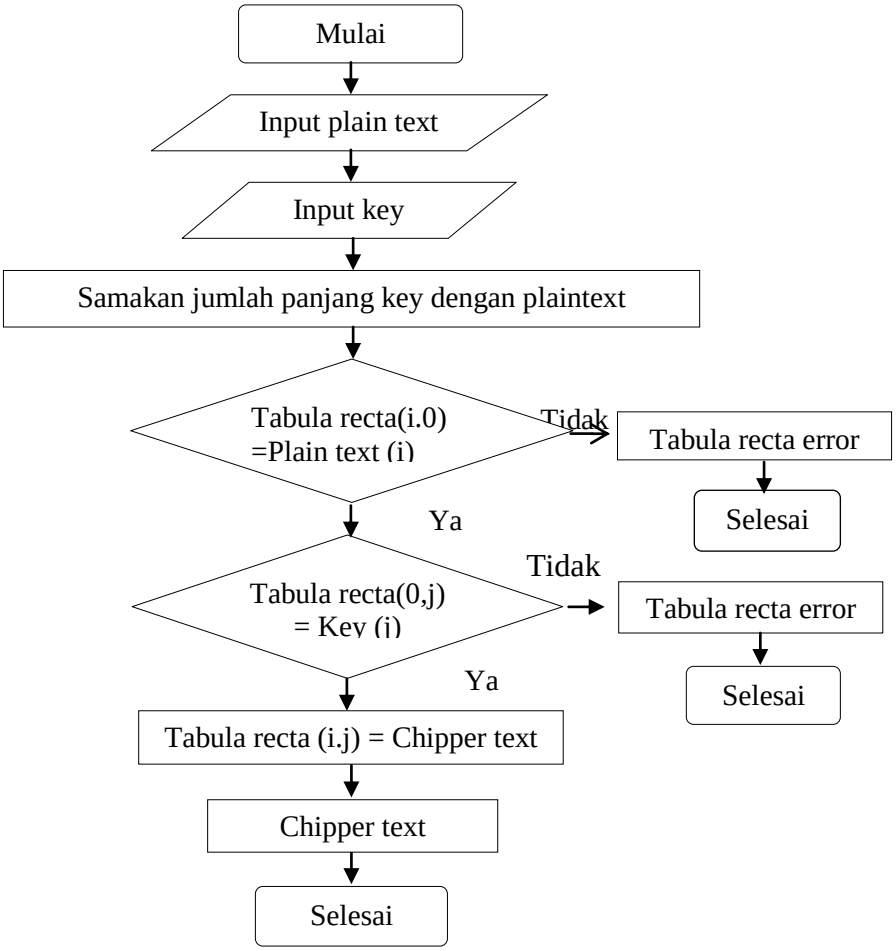
Dari gambar III.3 di atas dapat dijelaskan proses enkripsi penyisipan file pesan dimulai dengan penginputan pesan dan pemilihan gambar sebagai citra cover, kemudian mengubah file citra cover dan mengambil pixel merah(red) konversikan ke 8 bit lalu dari 8 bit di ambil 7 bit dari sebelah kiri disisakan bit ke 8 kemudian hitung jumlah banyak bit, kemudian input text konversi text ke nilai ascii file pesan menjadi deretan bit,konversi ke 8 bit, hitung jumlah banyak bit text, kemudian jika jumlah bit gambar lebih kecil dari bit text, maka sisipkan bit ke 8 dari gambar dengan bit-bit text, dari file pesan pada bit-bit LSB dari setiap pixel yang terpilih, menyisipkan kembali bit-bit yang telah disisipi ke dalam citra cover (gambar), mengubah kembali deretan bit menjadi bentuk pixel, menyimpan citra yang telah berisi pesan ke dalam file (*stegoimage*), kemudian menampilkan *stegoimage*.



Gambar III.4. Diagram alir proses dekripsi LSB

Dari gambar III.4 dapat dijelaskan proses alir proses deskripsi LSB dari file pesan dimulai dari pemilihan gambar yang sudah di enkripsi, kemudian ambil pixel merah(red), kemudian pixel merah tersebut di konversi menjadi 8 bit dari ke 8 bit tersebut di ambil 4 bit pertama kemudian di konversi ke karakter ascii yang 4 bit pertama berisi angka panjang text, kemudian ambil bit sebanyak panjang text, lalu bit yang sudah di ambil di konversi kembali ke karakter, dan kemudian karakter tersebut menampilkan karakter yang disisipkan pesan(plaintext) yang berubah chippertext dan sudah terenkripsi dari vigenere cipher.

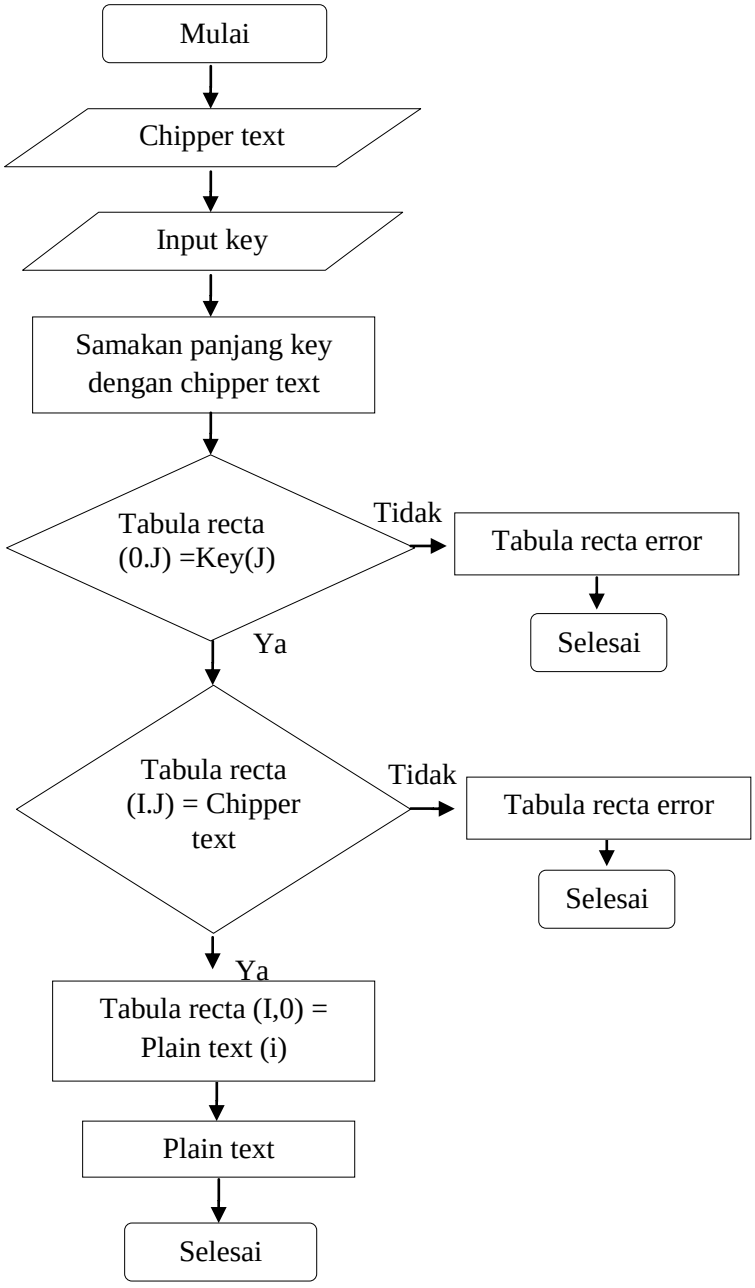
Pada metode vigenere cipher terdapat proses yaitu proses enkripsi pesan(plaintext) dan deskripsi pesan(plaintext). Berikut ini adalah diagram alir dari proses yang ditampilkan pada gambar III.5 dan gambar III.6



Gambar III.5. Diagram alir proses enkripsi Vigenere Cipher

Pada gambar III.5 dapat dijelaskan proses enkripsi vigenere cipher dari file pesan dimulai dari pengimputan pesan/excel kemudian masukan kunci/*key* pada vigenere cipher samakan jumlah panjang *key* dengan plaintext yang sudah di input, kemudian tabal tabula recta menyamakan panjang plaintext dengan ke *key*

lalu hitung jumlah *key* dan plaintext sama dengan ciphertext,sehingga menampilkan chipertext.



Gambar III.6. Diagram alir proses dekripsi Vigenere Cipher

Pada gambar III.6 dapat dijelaskan proses dekripsi vigenere cipher dari file pesan dimulai dari plaintext/pesan yang yang sudah terenkripsi kemudian masukan kunci/*key* pada vigenere cipher samakan jumlah panjang *key* dengan chipertext yang sudah di input, kemudian tabal tabula recta menyamakan panjang chipertxet dengan ke *key* lalu hitung jumlah *key* dan chipertext sama dengan plaintext ,sehingga menampilkan plaintext/pesan.

III.4. Analisis Kebutuhan non-Fungsional

Analisis kebutuhan nonfungsional adalah sebuah langkah dimana seorang pembangun aplikasi menganalisis sumber daya yang dibutuhkan untuk menggunakan aplikasi yang akan dibangun. Analisis kebutuhan nonfungsional yang dilakukan dibagi dalam tiga tahap, yaitu analisis pengguna (*user*), analisis kebutuhan perangkat keras, dan analisis perangkat lunak.

III.4.1. Analisis Pengguna

Pengguna yang akan menggunakan aplikasi steganografi ini dibagi menjadi dua bagian yaitu bagian pengirim dan penerima dengan kebutuhan spesifikasi seperti pada tabel III.2.

Tabel III.2. Kebutuhan Pengguna Aplikasi

Tipe Pengguna	Hak Akses	Tingkat Keterampilan	Jenis pelatihan
Pengirim	Dapat melakukan operasi penyisipan pesan kedalam citra (gambar).	Dapat mengoperasikan komputer dan memahami konsep steganografi.	Tidak diperlukan pelatihan khusus.
Penerima	Dapat melakukan operasi ekstrak pesan dari citra (gambar).	Dapat mengoperasikan komputer dan memahami konsep steganografi.	Tidak diperlukan pelatihan khusus.

III.4.2. Analisis Kebutuhan Perangkat Keras

Perangkat keras merupakan salah satu kebutuhan yang sangat penting bagi pembuatan steganografi. Perangkat keras akan mempengaruhi kinerja dari pembuatan steganografi, semakin tinggi spesifikasi dari perangkat keras yang digunakan maka akan semakin cepat pula pembuatan steganografinya. Perangkat keras yang digunakan pada pembangunan aplikasi steganografi ini yaitu seperti pada tabel III.3.

Tabel III.3. Spesifikasi perangkat keras untuk membangun aplikasi

No	Nama Perangkat	Spesifikasi
1	Prosessor	Intel Core 2.26 GHz
2	Monitor	Monitor 14 inch (1366x768)
3	Memori	RAM 2 GB DDR2
4	Harddisk	320 GB SATA 7200rpm

Sedangkan kebutuhan perangkat keras untuk menjalankan aplikasi yang dibangun, yang harus dipenuhi yaitu seperti pada tabel III.4.

Tabel III.4. Spesifikasi perangkat keras untuk menjalankan aplikasi

No	Nama Perangkat	Spesifikasi
1	Prosessor	Intel Core 2.26 GHz
2	Monitor	Monitor 14 inch (1366x768)
3	Memori	RAM 1 GB DDR2
4	Harddisk	250 GB SATA 7200rpm

III.4.3. Analisis Perangkat Lunak

Perangkat lunak yang digunakan untuk membangun aplikasi steganografi ini yaitu seperti pada tabel III.5.

Tabel III.5. Kebutuhan perangkat lunak untuk membangun aplikasi

No	Nama Perangkat	Spesifikasi
1	Sistem Operasi	Windows 7 Professional
2	VB. Net	Visual Studio 2010
3	JDK (*Java Development Kit*)	JDK 1.7.0
4	JRE (*Java Runtime Environment*)	JRE 7

Sedangkan kebutuhan Perangkat keras untuk menggunakan aplikasi steganografi ini yaitu seperti pada tabel III.6.

Tabel III.6. Kebutuhan perangkat lunak untuk menjalankan aplikasi

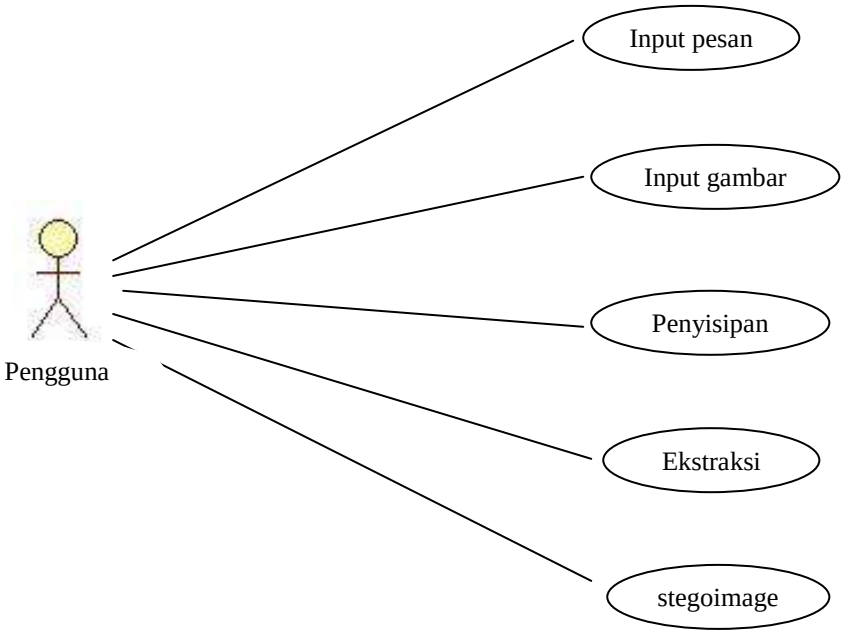
No	Nama Perangkat	Spesifikasi
1	Sistem Operasi	Windows XP
2	JDK (*Java Development Kit*)	JDK 1.7.0
3	JRE (*Java Runtime Environment*)	JRE 7

III.5. Analisis Kebutuhan Fungsional

Analisi kebutuhan fungsional adalah segala bentuk data yang dibutuhkan oleh sistem agar sistem dapat berjalan sesuai dengan prosedur yang dibangun. Aplikasi yang dibangun akan dimodelkan menggunakan *Unified Modeling Language* (UML), dan *tools* yang akan digunakan yaitu *use case* diagram, *activity* diagram, dan *sequence* diagram.

III.5.1. *Use Case Diagram*

Use case diagram digunakan untuk mengetahui apa saja yang dapat dilakukan oleh pengguna/user terhadap fungsionalitas yang terdapat pada aplikasi yang dibangun. *Use case* diagram pada aplikasi steganografi terlihat pada gambar III.7.



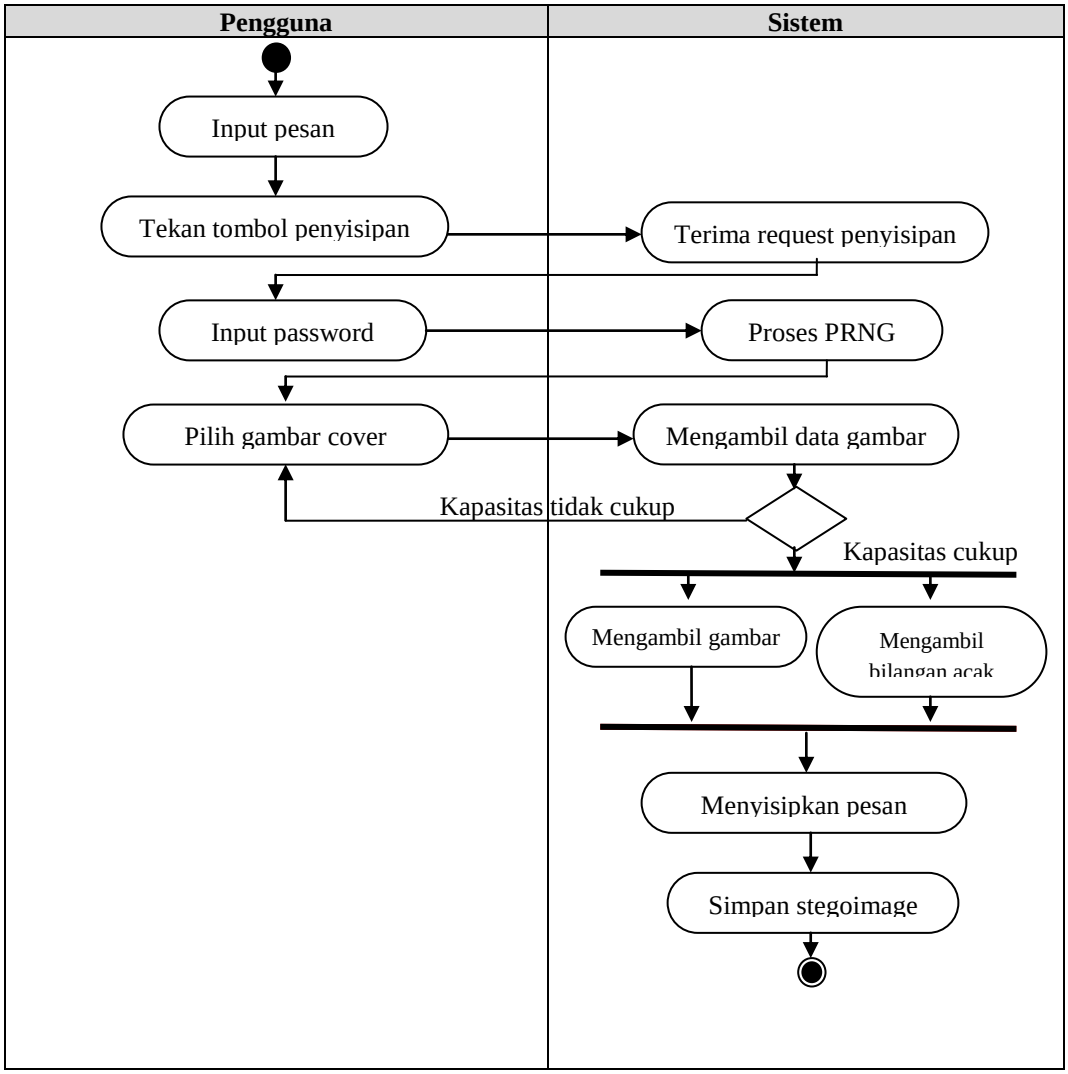
Gambar III.7. *Use case* diagram aplikasi

III.5.2. *Activity Diagram*

Activity diagram menggambarkan berbagai alir aktivitas dalam sistem yang sedang dirancang, bagaimana masing-masing alir berawal, *decision* yang mungkin terjadi, dan bagaimana mereka berakhir. *Activity* diagram yang terdapat pada aplikasi yang dibangun yaitu sebagai berikut:

1. *Activity Diagram* penyisipan pesan

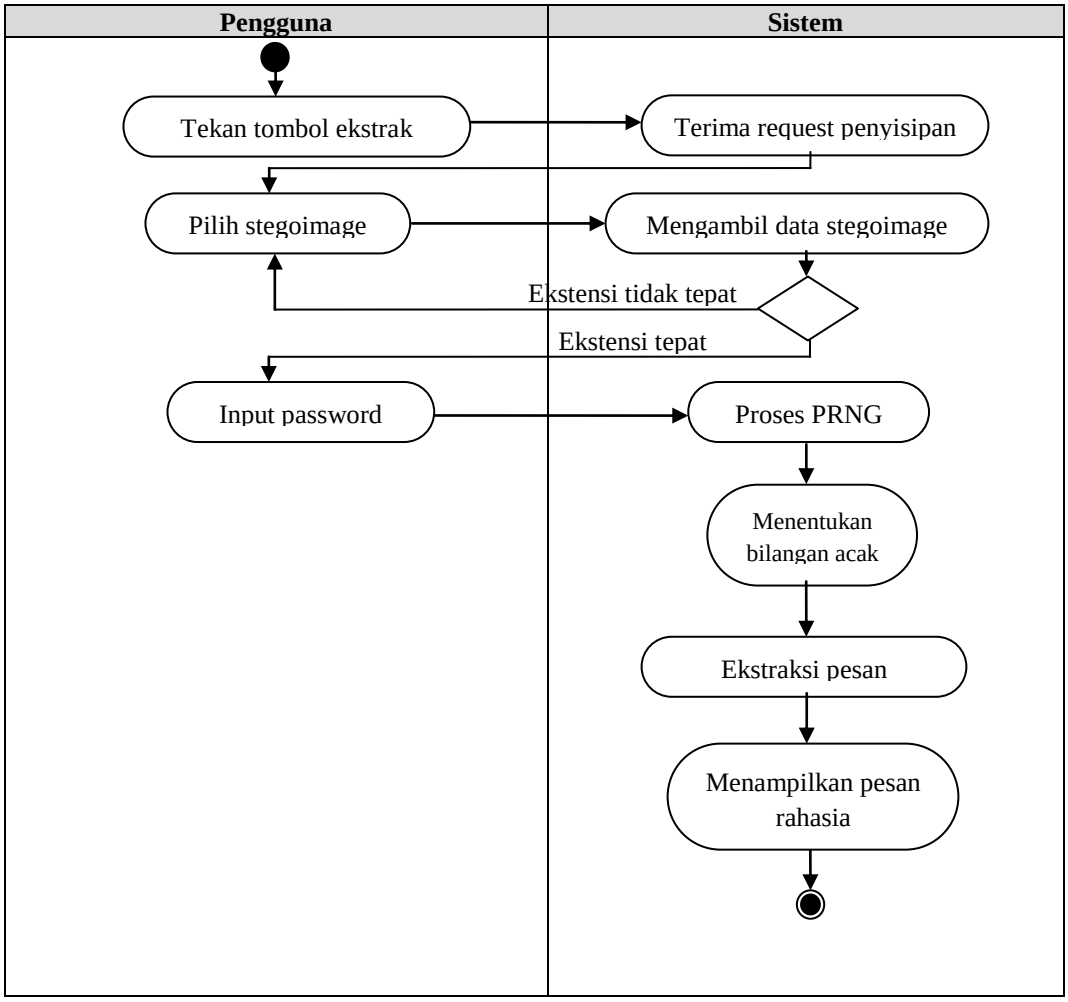
Activity diagram penyisipan menggambarkan alir aktivitas penyisipan yang dilakukan antara pengguna dangan sistem seperti terlihat pada gambar III.8.



Gambar III.8. *Activity Diagram* penyisipan pesan

2. *Activity Diagram* ekstraksi pesan

Activity diagram ekstrak menggambarkan alir aktivitas ekstrak yang dilakukan antara penerima dangan sistem seperti terlihat pada gambar III.9.



Gambar III.9. *Activity Diagram* ekstraksi pesan

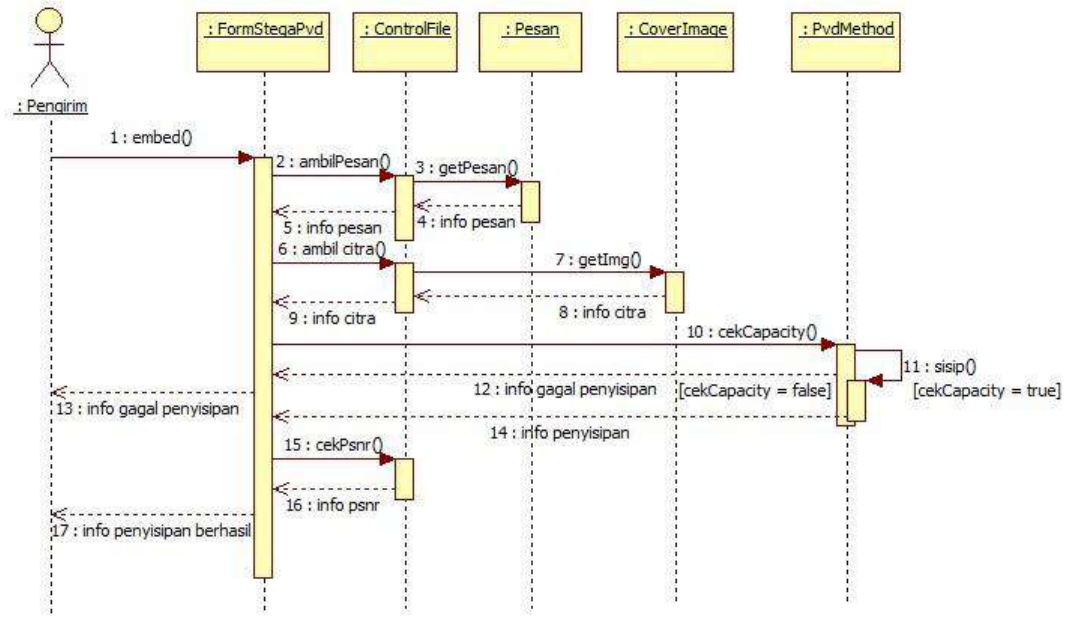
III.5.3. *Sequence Diagram*

Sequence diagram menggambarkan interaksi antar objek di dalam dan di sekitar sistem (termasuk pengguna, *display*, dan sebagainya) yang digambarkan terhadap waktu. *Sequence* diagram yang terdapat pada aplikasi steganografi yaitu sebagai berikut:

1. ***Sequence* Diagram Penyisipan**

Sequence diagram penyisipan merupakan diagram yang mengambarkan interaksi yang terjadi didalam sistem antara pengguna dengan sistem dalam

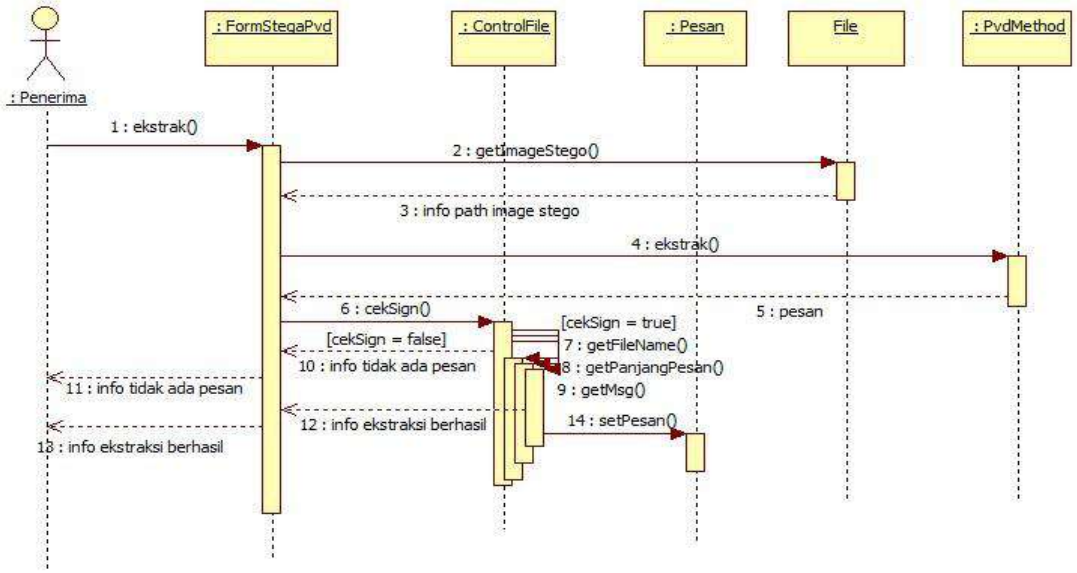
penyisipan pesan kedalam citra (gambar). *Sequence* diagram penyisipan aplikasi steganografi terlihat seperti pada gambar III.10.



Gambar III.10. *Sequence Diagram* penyisipan

2. *Sequence* Diagram Ekstrak

Sequence diagram ekstrak merupakan diagram yang mengambarkan interaksi yang terjadi didalam sistem antara penerima dengan sistem dalam ekstraksi pesan dari citra. *Sequence* diagram ekstrak aplikasi steganografi terlihat seperti pada gambar III.11.



Gambar III.11. *Sequence Diagram* ekstrak

III.6. Perancangan

Perancangan merupakan proses yang dilakukan oleh perancang sistem untuk mengerjakan spesifikasi sistem, membuat keputusan tentang bagaiman komponen sistem diaktualisasikan. Perancangan steganografi penyisipan pesan ini menggunakan visual studio 2010.

Perancangan sistem secara umum untuk membangun aplikasi data steganografi dengan metode LSB dan Vigenere Cipher pada file image ini terdiri atas beberapa tahap, meliputi perancangan :

1. Proses

Perancangan proses yang dimaksud adalah bagaimana sistem akan bekerja, proses-proses yang digunakan, mulai dari user melakukan input kemudian diposes oleh aplikasi sehingga dapat mengeluarkan output berupa *stegoimage*.

2. Perancangan Antarmuka

Perancangan antarmuka mengandung penjelasan tentang desain dan implementasi sistem yang digunakan dalam sistem yang dibuat.

a. Perancangan Antarmuka Home

b. Perancangan Sistem Aplikasi

Perancangan aplikasi ini terdiri dari dua proses yaitu :

1. Tahap *Embedded* Data

Tahap embedded data merupakan tahap penyisipan file atau suatu data teks ke dalam suatu gambar yang format file nya (.bmp) bertujuan untuk menyembunyikan data agar tidak terlihat oleh orang yang tidak berjak melihatnya.

2. Proses *Retriview* Data

Proses atau tahap yang dilakukan untuk membaca pesan rahasia yang telah disisipkan ke dalam file image yang disebut stego image dan akan menampilkan pesan rahasia yang ada di dalam stego image tersebut.

III.7. Rancangan Layar

Rancangan layar pada aplikasi steganografi ini dibagi menjadi 2 bagian, yaitu:

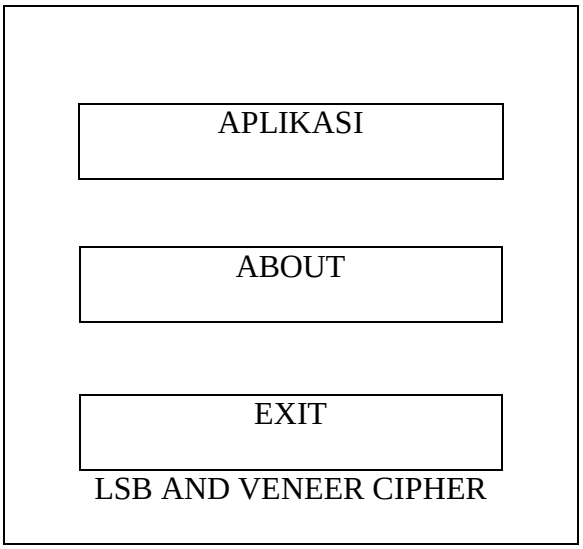
1. Rancangan antarmuka.
2. Rancangan kotak pesan.

III.7.1. Rancangan Antarmuka

Perancangan antarmuka adalah tahapan pembuatan rancangan antarmuka untuk digunakan pada pembangunan aplikasi steganografi, yang dibagi menjadi dua bagian, yaitu rancangan antarmuka *home* dan rancangan antarmuka aplikasi.

1. **Rancangan Antarmuka *Home***

Perancangan antarmuka *home* merupakan tahapan rancangan antarmuka dari awal aplikasi ketika dijalankan. Perancangan antarmuka *home* dari aplikasi steganografi ini terlihat pada gambar III.10.

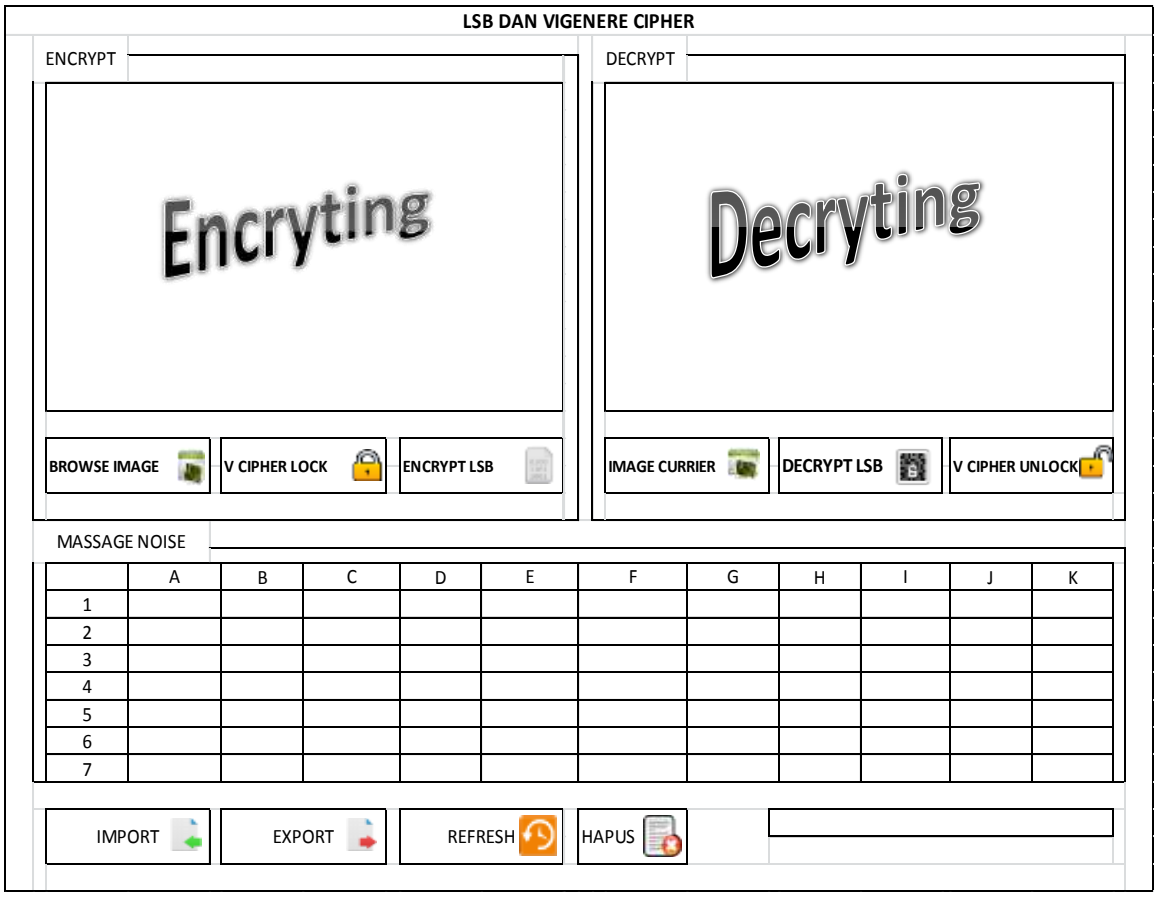


Gambar III.12. Rancangan antarmuka home

Tampilan gambar III.12. merupakan tampilan pembuka dari aplikasi, dimana dalam tampilan halaman pembuka terdapat form pengenalan aplikasi dan kemudian dapat melanjutkan untuk menjalankan aplikasi.

2. Rancangan Antarmuka Aplikasi

Perancangan antarmuka aplikasi merupakan tahapan rancangan antarmuka dari aplikasi ketika dijalankan. Perancangan antarmuka aplikasi steganografi ini terlihat pada gambar III.13.



Gambar III.13. Rancangan antarmuka aplikasi

Tampilan gambar III.13 merupakan tampilan halaman utama dari aplikasi yang berfungsi untuk melakukan proses penyisipan dan ekstrak pesan, dimana dalam form aplikasi di atas tampak beberapa bagian yang memiliki fungsi sebagai berikut :

1. Tombol browse image berfungsi untuk mengambil gambar yang masukkan ke dalam cover gambar *Encryptyng*.

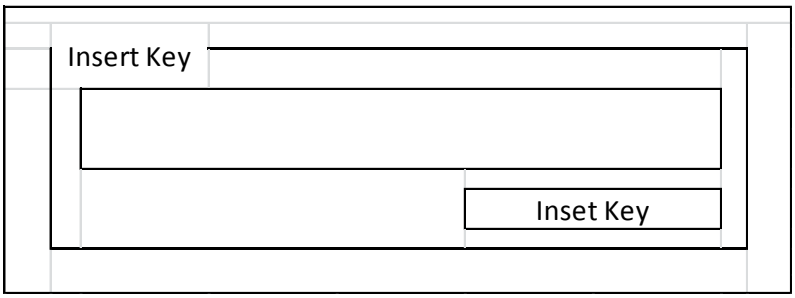
2. Tombol impor berfungsi mengambil pesan yang akan disisipkan ke dalam gambar pada kolom noise.
3. Tombol *vigenere cipher lock* berfungsi untuk memasukkan password jika pesan akan disisipkan ke gambar.
4. Tombol encrypt lsb berfungsi untuk meyimpan gambar yang data diingkan sudah disisipkan dan sudah di beri password.
5. Tombol *image currier* berfungsi untuk mengambil dan menampilkan gambar yang sudah disimpan pada cover gambar *Decryting*.
6. Tombol decrypt lsb berfungsi untuk menampilkan data yang sudah disisipkan ke dalam gambar pada kolom massage noise.
7. Tombol vigenere cipher unlock berfungsi untuk memasukan kata password yang sudah disimpan dan akan memulihkan data yang sebenarnya pada kolom *massage noise*.
8. Tombol berfungsi untuk menyimpan data dan manampilkan data dalam bentuk excel.
9. Tombol refresh berfungsi untuk mengembalikan tampilan program semula.
10. Tombol hapus berfungsi untuk menghapus bagian-bagian yang salah.

III.7.2. Rancangan Kotak Pesan

Kotak pesan merupakan tampilan dari suatu perangkat lunak yang berfungsi untuk menyampaikan notifikasi dan informasi kepada pengguna agar perangkat lunak lebih interaktif. Perancangan kotak pesan pada aplikasi steganografi terdiri dari:

1. **Rancangan kotak pesan input password penyisipan**

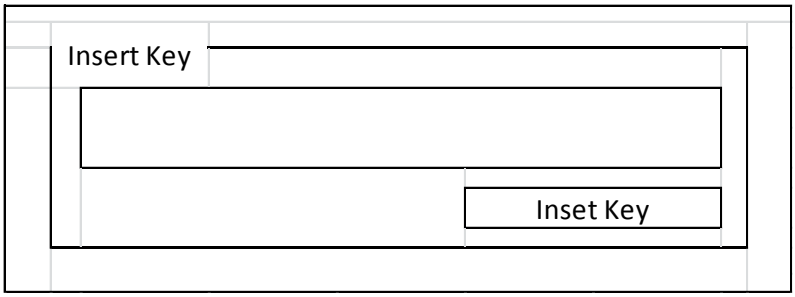
Rancangan kotak pesan input password penyisipan ini menampilkan pesan untuk memasukkan password pengaman pesan rahasia. Adapun tampilan rancangan dapat dilihat pada gambar III.14.



Gambar III.14. Rancangan kotak pesan input password penyisipan

2. **Rancangan kotak pesan input password ekstrak**

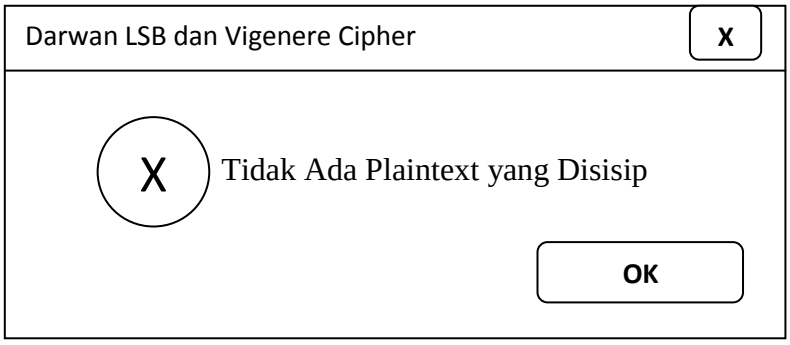
Rancangan kotak pesan input password pengekstrakan ini menampilkan pesan untuk memasukkan password menampilkan pesan rahasia yang ada pada *stegoimage*. Adapun tampilan rancangan dapat dilihat pada gambar III.15.



Gambar III.15. Rancangan kotak pesan input password ekstrak

3. Rancangan kotak pesan error tidak ada data yang disisipkan

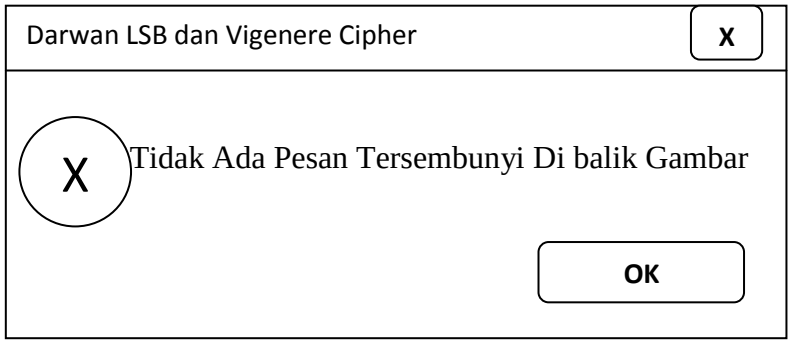
Rancangan kotak pesan error tidak ada data yang disisipkan ini menampilkan pesan untuk menampilkan pesan error untuk steganografi dikarenakan data tidak ada disisipakn di file gambar *bitmap* (.bmp). Adapun tampilan rancangan dapat dilihat pada gambar III.16.



Gambar III.16. Rancangan kotak pesan input error ekstension

4. Rancangan kotak pesan error tidak ada pesan

Rancangan kotak pesan error ukuran gambar ini menampilkan pesan untuk menampilkan pesan error tidak data yang tersembunyi atau disisipkan pada suatu gambar untuk proses steganografi. Adapun tampilan rancangan dapat dilihat pada gambar III.17.

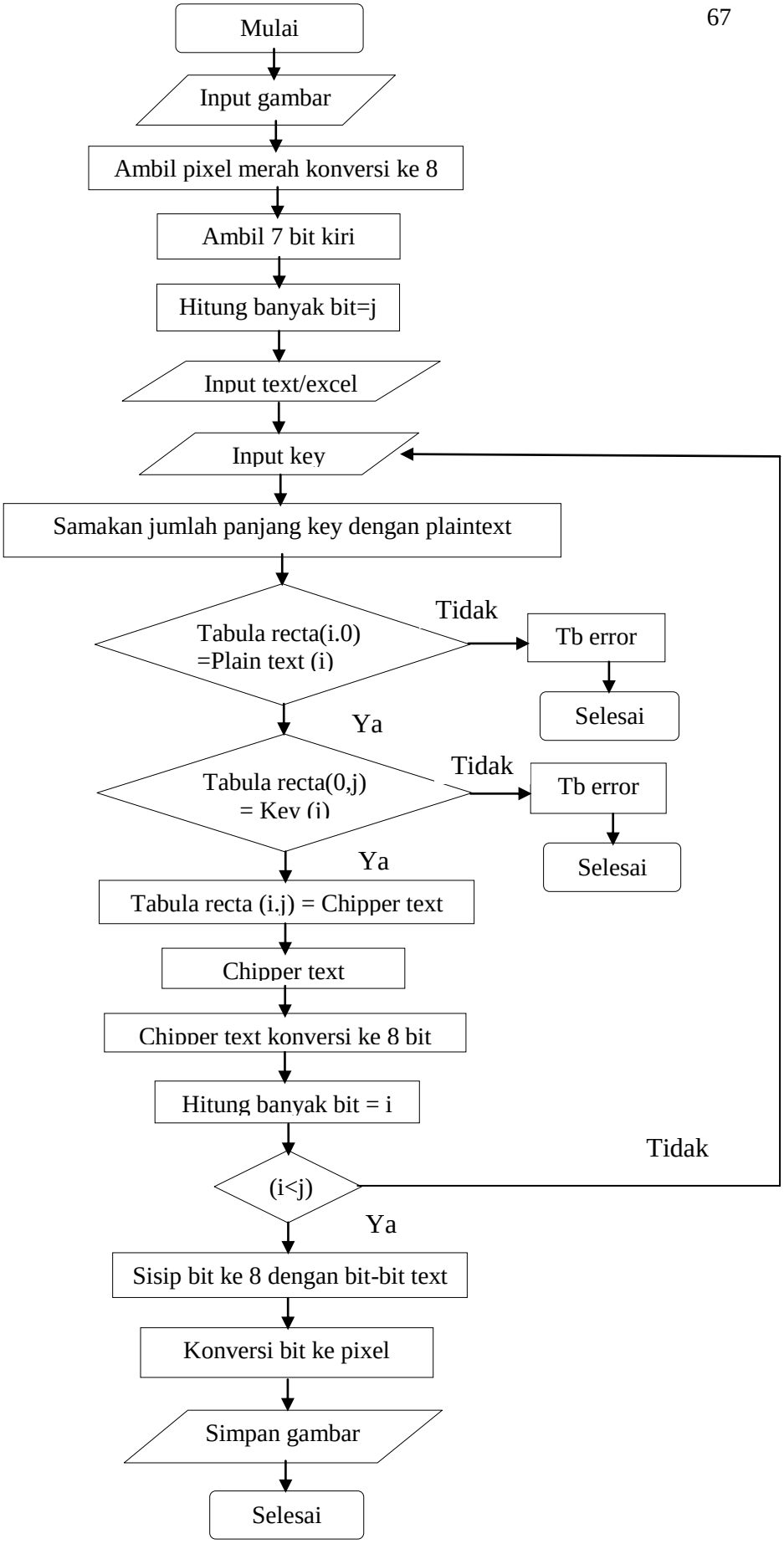


Gambar III.17. Rancangan kotak pesan error ukuran gambar

III.8. Flowchart Algoritma

1. Flowchart Algoritma Penyisipan Pesan Pada Aplikasi

Flowchart pada program ini dimulai dengan membuka aplikasi program. Jika pengguna menjalankan program tersebut maka program berjalan sesuai instruksinya. Flowchart penyisipan merupakan gambaran umum dari langkah-langkah proses penyisipan di aplikasi steganografi yang dibangun. Proses penyisipan dimulai dengan menginput pesan dan melakukan penyisipan terhadap pesan yang telah teracak ke dalam cover gambar. Flowchart penyisipan pesan pada aplikasi diperlihatkan pada gambar III.18.



Gambar III.18. Flowchart algoritma entraksi penyisipan pesan

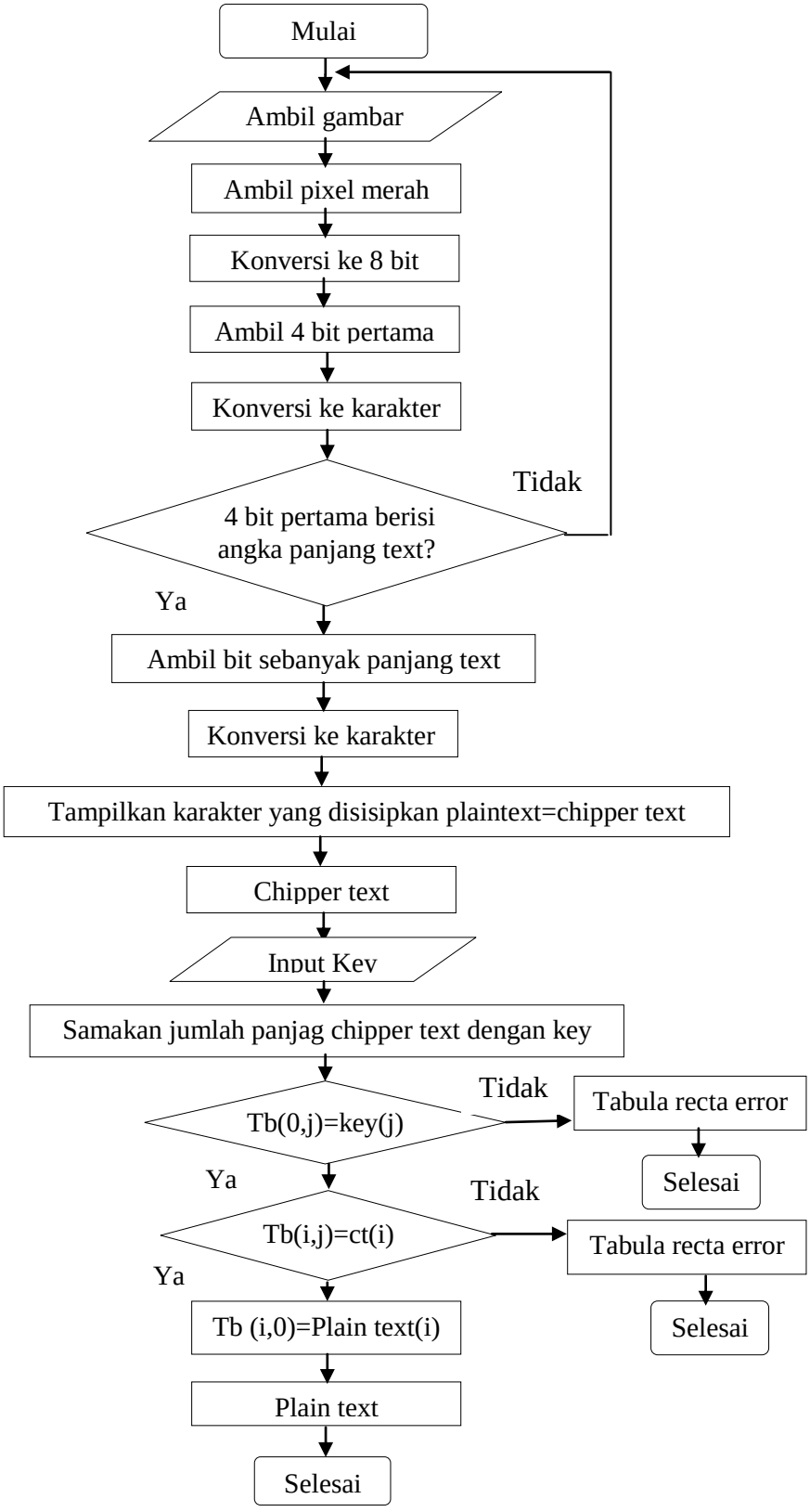
Adapun keterangan gambar III.18 di atas sebagai berikut :

1. Mulai menyatakan awal dari rangkaian proses
2. Ambil gambar pada polder yang ditentukan
3. Kemudian ambil pixel merah pada gambar.
4. Konversi ke 8 bit pixel mereh tersebut.
5. Kemudian ambil bit ke 7 dari kiri dari bit ke 8.
6. Kemudian hitung banyak jumlah bit dari gambar.
7. Input plaintext/pesan
8. Masukan key pada proses vigenere chipper,
9. Kemudian samakan jumlah key dan plaintext
10. Kondisi menyatakan proses plaintext tabula recta berhasil atau kembali menginputkan pesan sehingga proses error.
11. Kondisi menyatakan proses plaintext tabula recta berhasil atau kembali menginputkan pesan sehingga proses error.
12. Kondisi menyatakan proses plaintext tabula recta berhasil atau kembali menginputkan pesan sehingga proses error.
13. Kondisi menyatakan proses *key* tabula recta berhasil atau kembali menginputkan pesan sehingga proses error.
14. Kemudian proses plaintext dan key sama berada pada titik yang sama sehingga terbentuk chipertext.
15. Kemudian chiphertext di konversi ke 8 bit.
16. Kemudian chipertext yang sudah di konversi menjadi bit, hitung jumlah banyak bit.

17. Kemudian jika jumlah bit gambar lebih besar dari pada bit text maka proses berhsil, jika bit gambar lebih kecil dari pada bit text maka proses error.
18. Kemudian sisipkan bit ke 8 dengan bit-bit text.
19. Konversi bit-bit yang sudah disisipkan menjadi sebuah pixel.
20. Selanjutnya adalah aplikasi melakukan proses menyisipkan pesan ke dalam gambar.
21. Hasil akhir dari proses adalah *stegoimage* yaitu file *bitmap* (.bmp) yang telah disisipi pesan yang tersimpan pada directory yang telah user tentukan
22. Selesai

2. Flowchart algoritma Ekstraksi Pesan Pada Aplikasi

Flowchart ekstraksi merupakan gambaran umum dari langkah-langkah proses ekstraksi di aplikasi steganografi yang dibangun. Proses ekstraksi pesan dimulai dari pemilihan file *stegoimage*, kemudian menginputkan password untuk membangkitkan bilangan acak semu dan menampilkan pesan pada akhir prose. Flowchart ekstraksi pada aplikasi steganografi digambarkan dalam diperlihatkan seperti pada gambar III.19.



\

Gambar III.19. Flowchart algoritma dekripsi pesan

Adapun keterangan gambar III.19 di atas sebagai berikut :

1. Mulai menyatakan awal dari rangkaian proses.
2. Mengambil/inputkan file gambar (*stegoimage*) yang disisipi pesan.
3. Ambil pixel merah(*red*) pada gambar.
4. Konversi pixel merah ke 8 bit.
5. Kemudian ambil 4 bit pertama dari 8 bit.
6. Kemudian 4 bit tersebut di konversi ke dalam karakter.
7. Lalu kondisi proses 4 bit pertama berisi angka panjang text, jika sesuai proses lanjut.
8. Kemudian proses selanjutnya ambil bit sebanyak panjang text.
9. Kemudian bit yang sudah di ambil di konversi ke dalam karakter, sehingga berada pada posisi yang sama, mengahasilkan chipertext.
10. Kemudian vigenere chipper input *key* (password yang di pakai pasa saat enkripsi pesan)
11. Kemudian samakan panjang chipertext dengan *key.*
12. Lalu proses tabula recta mencari posisi letak panjang *key.*
13. Kemudian tabula recta mencari letak posisi panjang chipertext.jika panjang chipertext tidak sama dengan *key* maka proses error.
14. Kondisi ini menyatakan tabula recta menyatukan letak titik posisi chippertext dan *key*, sehingga posisi plaintext ditemukan,
15. Plaintext/Pesan di tampilkan.
16. Selesai