BAB III
ANALISIS DAN PERANCANGAN SISTEM

3.1 Analisis Sistem

Dalam proses pengamanan file berupa gambar awam biasanya menggunakan beberapa metode algoritma tertentu yang dapat memberikan pengamanan terhadap data maupun informasi dalam file tersebut agar tidak bisa dibaca maupun tidak dapat dimengerti oleh pihak luar. Dalam hal seperti ini sistem yang sedang berjalan dalam melakukan pengamanan data berupa gambar saat ini dilakukan dengan menggunakan metode RSA. Namun dalam penelitian sebelumnya didapatkan hasil bahwa bisa saja sewaktu-waktu file maupun data tersebut dapat dimanipulasi oleh pihak yang tidak berwenang. Maka dari itu masih terdapat sebuah celah untuk orang yang tidak berkepentingan dapat mengambil maupun memiliki data file berupa gambar tersebut. Maka diperlukan sebuah teknik maupun metode tambahan untuk melakukan pengamanan yang ganda agar sistem yang berjalan saat ini semakin kuat dalam melakukan pengamanan data. Dalam hal ini metode tambahan tersebut dapat melakukan pengamanan dokumen yang ditandatanganinya dalam aspek *integrity, authentication,* dan *non-repudiation*. Pengamanan data tambahan tersebut menggunakan metode *Digital signature*.

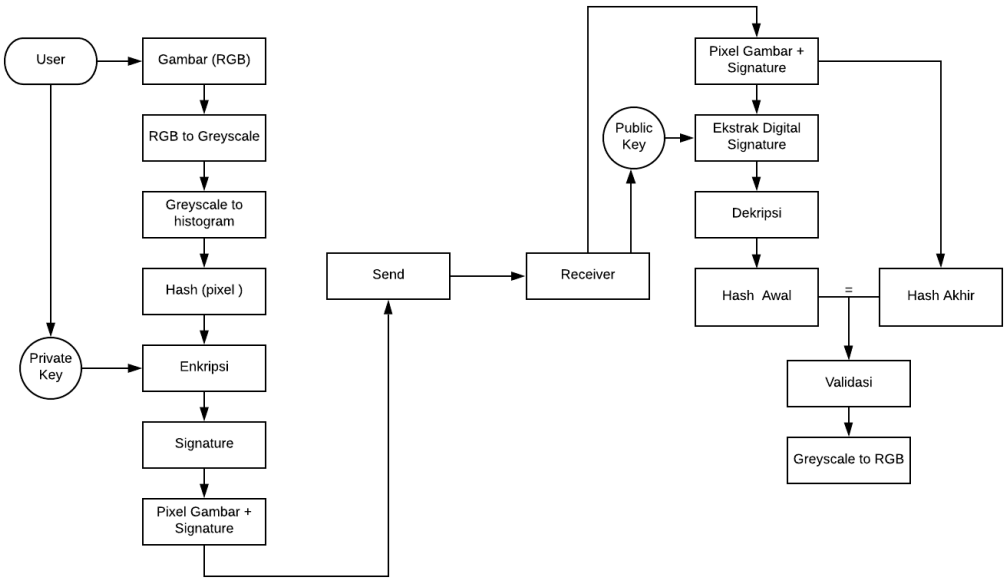
3.2 Hasil Analisis

Hasil analisis dari penambahan teknik maupun metode pengamanan data berupa gambar yang dibangun yaitu *digital signature* dengan kriptografi RSA terdiri dari tiga proses, yaitu proses pembangkitaan kunci, proses *sign digital signature*, dan proses *verify digital signature*. Pada dasarnya, ketiga proses tersebut memiliki langkah yang sama persis dengan tiga proses algoritma RSA, namun memiliki beberapa perbedaan. Perbedaannya yaitu pada pembuat pembangkitan kunci merupakan si pengirim pesan, berbeda dengan kriptografi RSA, dengan penerima pesan sebagai pembangkitan kunci. Selain itu terdapat beberapa perbedaan dari proses *sign* dan *verify* dengan proses enkripsi dan dekripsi. Jika dalam proses enkripsi menggunakan kunci *public* dan proses dekripsi menggunakan

kunci pribadi, sedangkan pada saat proses *sign digital signature* menggunakan kunci pribadi, dan saat proses *verify* menggunakan kunci publik .

3.3 Dekripsi Sistem

Sistem yang akan dibangun merupakan sistem dimana algoritma RSA akan di tambahkan dengan metode lain sehingga didapatkan sistem pengamanan data berupa gambar dengan menggunakan pengamanan ganda. Metode tambahan tersebut ialah metode *Digital signature.* Gambaran umum sistem yang akan dibangun seperti pada gambar 3.1



Gambar 3. 1 Gambaran Sistem Kriptografi pengamanan file berupa gambar menggunakan *Digital signature* & metode RSA

Pada *User* :

1. *User* harus memiliki dua pasang kunci,yaitu kunci publik dan kunci *private* khususnya kunci *private* untuk memproses *signature*.
2. File berupa gambar dengan 3 ruang warna yaitu RGB akan dirubah menjadi *Greyscale* dengan rumus *greyscale* = 0.299R + 0.587G + 0.114B

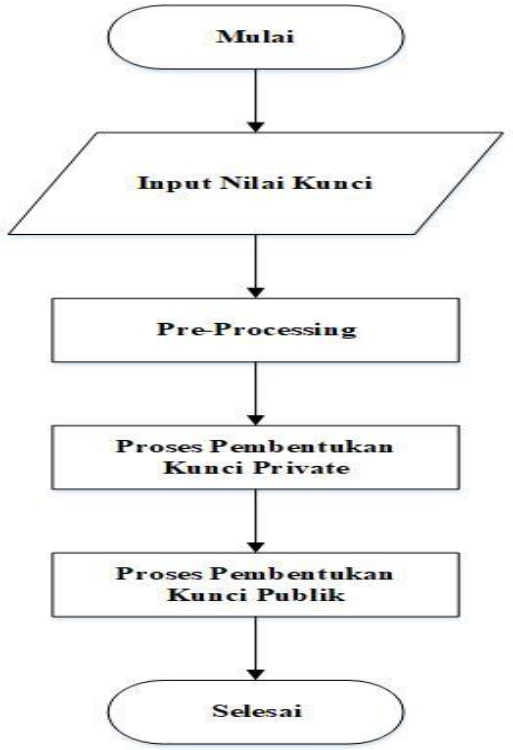
3. File berupa RGB dan *Greyscale* akan di *hash* yang dimana akan menghasilkan *Massage Digest(Value).*
4. Hasil *message digest* tersebut akan dienkrip menggunakan *private key*. Hasil enkrip inilah yang dinamakan dengan *digital signature*.
5. Kemudian *digital signature* dilekatkan dengan *pixel* gambar (dengan cara menyambung),lalu keduanya dikirimkan melalui saluran komunikasi. Dalam hal ini kita katakan bahwa pesan sudah ditandatangani oleh pengirim.

Pada *Receiver* :

1. *Signature* di otentifikasi untuk dibuktikan keaslian
2. *Signature* di dekrip menggunakan kunci publik pengirim, menghasilkan *message digest* semula.
3. Gambar di *hash*ing kembali untuk menghasilkan *message digest*.
4. Selanjutnya *message digest* akan dibandingkan Jika MD‟=MD, berarti otentifikasi valid dan *signature* yang diterima berasal dari pengirim yang benar.
5. Jika hasil valid maka gambar *greyscale* akan dirubah dalam bentuk RGB.

3.3.1 Pembentukan Kunci

Dalam pembentukan kunci ini, memiliki 3 tahap, tahap pertama adalah *pre – processing*, tahap kedua adalah tahap pemrosesan kunci *private*, tahap ketiga adalah pemrosesan kunci publik. Pembentukan kunci pada aplikasi ini mengikuti alur sesuai yang di tunjukan pada gambar 3.2



Gambar 3. 2. Alur Pembentukan Kunci

3.3.1.1 *Pre-Processing*

Pembentukan kunci yang akan dibuat adalah kunci *private* (untuk *user*) dan kunci publik (untuk *reciver*). Pada *pre-processing* ini akan menjelaskan tentang 5 parameter yang digunakan pada saat pembuatan kunci *private* maupun kunci publik ,parameter tersebut adalah *p,q,n,q(n)* dan *e.* Parameter yang pertama adalah *p*, yaitu suatu nilai angka yang harus prima dimana nilai *p* ini sudah ditentukan,alasan sudah ditentukan karena nilai *p* ini tidak ada perhitungan khusus. Parameter yang kedua adalah *q*, yaitu suatu nilai angka yang harus prima . Parameter yang ketiga adalah *n*,yaitu suatu nilai dimana syarat $n = p \times q$. Parameter yang keempat adalah

q(n),yaitu suatu nilai dimana syarat *q(n) = (p-1)(q-1)* . Parameter yang keempat adalah *v*,yaitu suatu nilai dimana syarat *1 < v < q(n).* Parameter yang kedua adalah *e*, yaitu suatu nilai angka yang harus bilangan bulat. Alasan nilai *p,q,n,q(n)* dan *e* ini harus dihitung terlebih dahulu karena proses perhitungan kunci *private* dan kunci publik akan memakai nilai – nilai tersebut. Prosedur perhitungannya adalah sebagai berikut :

1. Penentuan nilai *p*

   Proses nilai *p* tidak ada perhitungan khusus namun memiliki syarat bahwa nilai *p* harus berupa bilangan prima. Contoh Inisialisasi nilai *p* adalah 13. Mengapa 13,karena 13 adalah bilangan prima atau bias juga di inisialisasi dengan nilai bilangan prima lainnya.

2. Penentuan nilai *q*

   Proses nilai *q* tidak memiliki perhitungan khusus namun memiliki syarat yang harus terpenuhi. Syarat pertama adalah *q* harus bilangan prima . Contoh Inisialisasi nilai *q* adalah 23. Mengapa 23,karena 23 adalah bilangan prima atau bias juga di inisialisasi dengan nilai bilangan prima lainnya.

3. Penentuan nilai *n*

   Proses nilai *n* tidak ada perhitungan khusus tapi memiliki syarat yang harus terpenuhi. Yaitu jika memakai rumus :

   $$n = p \times q \qquad (3.1)$$

   Contoh perhitungan :

   Diinisialisasikan

   $p = 13$

   $q = 23$

   $= p \times q$

   $= 13 \times 23$

   $= 299$

   Berdasarkan contoh perhitungan di atas nilai *n* yang di dapat adalah 299. Karena nilai tersebut telah memenuhi syarat berdasarkan rumus (3.1).

4. Penentuan nilai *q(n)*

   Proses nilai *q(n)* tidak ada perhitungan khusus tapi memiliki syarat yang harus terpenuhi. Yaitu jika memakai rumus :

   $$q(n) = (p-1)(q-1) \quad (3.2)$$

   Contoh perhitungan :

   Diinisialisasikan

   $p = 13$

   $q = 23$

   $= (p-1)(q-1)$

   $= (13-1)(23-1)$

   $= 12 \times 22$

   $= 264$

   Berdasarkan contoh perhitungan di atas nilai *q(n)* yang di dapat adalah 264. Karena nilai tersebut telah memenuhi syarat berdasarkan rumus (3.2).

5. Penentuan nilai *e.*

   Proses niali *e* tidak memiliki perhitungan khusus namun memiliki syarat yang harus terpenuhi. Syarat pertama adalah *e* harus bilangan bulat .

3.3.1.2 Pembangkitan Kunci Publik

Proses pembentukan kunci publik ini adalah mencari nilai *n* dan *e* dengan menggunakan cara sebagai berikut:

Pilih sebuah bilangan bulat sebagai kunci publik, katakanlah *e*, yang relatif prima terhadap *q(n).* Misalkan kita pilih *e* = 5 . Dapat kita lihat bahwa 5 relatif prima terhadap 264. Kemudian masukkan nilai *n* yang terdapat di proses pre-processing diatas untuk dipasangkan dengan nilai *e*. maka terbentuklah Kunci Publik yakni pasangan (*e*,*n*) = (5,299).

3.3.1.3 Pembangkitan Kunci *Private*

Proses pembentukan kunci publik ini adalah mencari nilai *d* dengan menggunakan cara sebagai berikut:

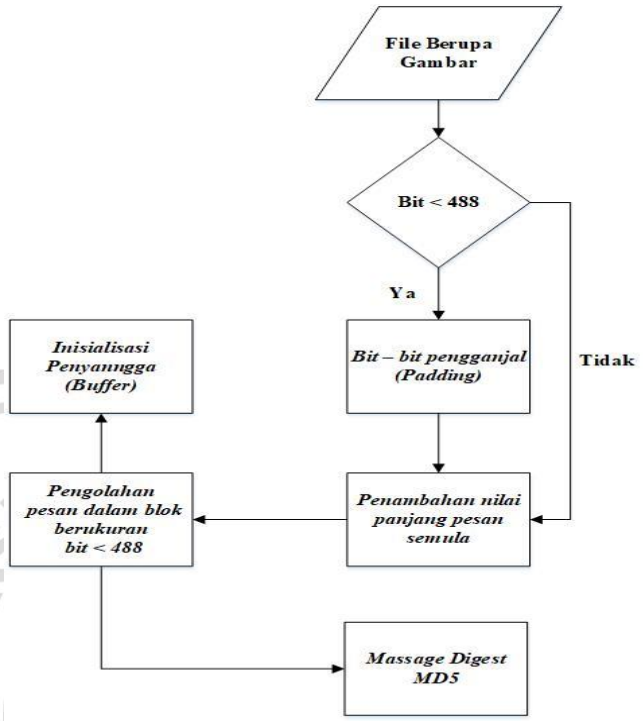
Bangkitkan kunci *private d* dengan menggunakan persamaan $e.d \equiv 1(mod\ q(n))$. Perhatikan bahwa $e.d \equiv 1(mod\ q(n))$ ekivalen dengan $e.d \equiv 1 + kq(n)$ dimana $k \in N$. jadi kunci *private* d dapat juga dihitung dengan cara sederhana sebagai $d = \frac{1+kq(n)}{e}$. Dalam hal ini kita peroleh kunci *private*nya d =53, sebab $e.d$ =5.53 = 265 ≡ 1(mod264). Jadi kunci *private d* adalah 53.

3.3.2 Pembentukan *Massage Diggest*

Message digest adalah suatu besaran (*value*) yang berasal dari suatu data atau pesan yang memiliki sifat yang unik yang menandai bahwa pesan tersebut mempunyai suatu besaran tertentu.

Perhitungan *message digest* menggunakan *hash* MD5 pada file data berupa gambar yang diberikan sebagai input. MD5 merupakan salah satu jenis fungsi *hash* searah yang dimana hasilnya tidak dapat dikembalikan seperti sedia kala. MD5 akan memberikan output berupa 128-*bit message digest* dimana output tersebut akan digunakan untuk proses autentikasi. File berupa gambar yang identik mampu menghasilkan *message digest* yang identik juga dan seandainya citra mengalami sebuah perubahan pada 1 bit saja maka akan menghasilkan *message digest* yang berbeda (Shah, 2015). Pada prosesnya, MD5 akan melakukan beberapa langkah untuk menghasilkan 128-*bit message digest*. Pembentukan *Hash/ Massage*

Diggest pada aplikasi ini mengikuti alur sesuai yang di tunjukan pada gambar 3.3.



Gambar 3. 3 Alur Pembentukan *Massage Digest*

Pada gambar 3.3 ditunjukan alur proses pembentukan *message digest* dengan menggunakan MD5. Citra akan diubah menjadi barisan biner kemudian hasil tersebut akan di-padding jika panjangnya kurang dari 448-bit. Lalu, hasil padding tadi akan di tambahkan 64-bit panjang pesan sehingga total bit nya ialah 512-bit. Setelah itu, proses 512-bit tadi sebanyak 16 kali dengan menggunakan *initial buffer* serta menggunakan 4 operasi putaran. Berikut adalah *initial buffer* yang umum digunakan (Rachmawati, Tarigan dan Ginting, 2018):

A = 0 1 2 3 4 5 6 7

B = 8 9 A B C D E F

C = F E D C B A 9 8

D = 7 6 5 4 3 2 1 0

Untuk melakukan operasi putarannya, MD5 akan menggunakan persamaan berikut (Seta, Ridho dan Theresiawati, 2017):

Putaran ke-1 : $F(X, Y, Z) = (X \wedge Y) \vee (\sim X \wedge Z)$

Putaran ke-2 : $G(X, Y, Z) = (X \wedge Z) \vee (Y \wedge \sim Z)$

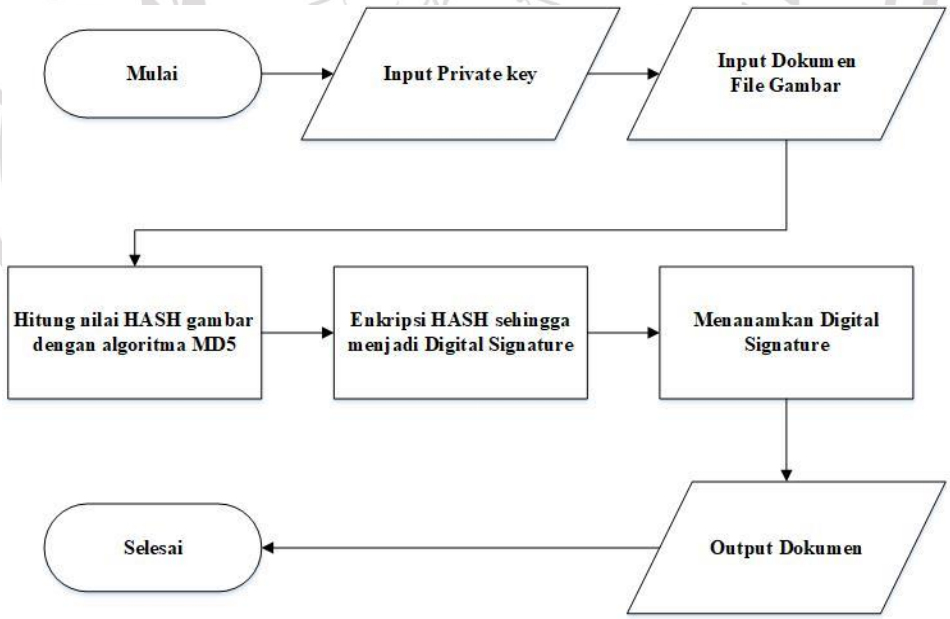
Putaran ke-3 : $H(X, Y, Z) = X \oplus Y \oplus Z$

Putaran ke-4 : $I(X, Y, Z) = Y \oplus (X \vee \sim Z)$

Message digest yang dihasilkan akan berupa A, B, C, dan D dimana untuk membentuk sebuah *message digest* akhir maka hasil tersebut akan diurutkan berdasarkan abjad yaitu A hingga D.

3.3.3 Pembentukan *Digital signature*

Tanda tangan digital (*digital signature*) merupakan protokol yang dijalankan untuk dapat memverifikasi keaslian dan keutuhan dari pesan yang akan dikirimkan dalam suatu saluran komunikasi. Pembentukan *digital signature* pada aplikasi ini mengikuti alur sesuai yang ditunjukan pada gambar 3.4.



Gambar 3.4 Alur Proses Pemberian *Digital signature*

Proses pembuatan *digital signature* dijelaskan menggunakan contoh berikut. Jika terdapat suatu gambar berukuran 4 x 2 piksel, dan tiap piksel terdiri dari 3 warna (RGB).

Tabel 3.1 Contoh Susunan Nilai Warna pada file gambar

R	**G**	**B**	**R**	**G**	**B**	**R**	**G**	**B**	**R**	**G**	**B**
200	200	100	45	55	10	90	120	90	255	0	0
80	70	0	255	255	255	90	90	10	178	190	240

Pada Tabel 3.1 Piksel yang pertama memiliki komponen warna R = 200, G=200, B=100, piksel yang kedua memiliki komponen warna R = 45, G = 55, B = 10. Tiap komponen warna ini kemudian akan dihitung nilai ringkasan (*hash*) dengan menggunakan algoritma MD5. MD5 akan menghasilkan output *hash* dengan panjang 16 *byte*, dengan panjang input bebas, sehingga input dengan panjang 1 *byte* akan tetap menghasilkan 16 *byte hash*, input dengan panjang 100 *byte* akan menghasilkan 16 *byte hash* juga.

Tabel 3.2 Contoh Hasil Proses Tahap Pertama: Pembuatan Ringkasan

Nilai Warna		***Hash***
200		
200		
100		
45		
55	`Diproses dengan Algoritma MD5`	120
10		60
90		70
120		44
90		110
255		45
0		121
0		190

80		180
70		245
0		235
255		89
255		100
255		112
90		78
90		90
10		
178		
190		
240		

Pada tahap pertama menghasilkan nilai ringkasan (*hash*). Pada penelitian ini digunakan algoritma MD5 untuk membuat ringkasan, sehingga berdasarkan spesifikasi MD5, *hash* yang dihasilkan memiliki panjang 16 *byte*. Pada contoh di Tabel 3.2. Dihasilkan nilai *hash*: 120, 60, 70, 44, 110, 45, 121, 190, 180, 245, 235, 89, 100, 112, 78, 90

Tahap kedua yaitu mengenkripsi nilai *hash* ini dengan algoritma RSA. Algoritma RSA bersifat asimetrik. Sehingga kunci yang digunakan untuk enkripsi berbeda dengan kunci dekripsi pada penggunaan di *digital signature*, kunci unntuk enkripsi dimiliki dan dirahasilakan oleh pemeilik citra digital. Karena sifatnya ini, maka kunci enkripsi disebut sebagai kunci *private*.

Tabel 3.3 Contoh Hasil Proses Tahap Pertama: Pembuatan Ringkasan/*Hash*

Nilai Warna		***Hash***	***Hash* Terenkripsi**
200			
200			
100			

45			
55	Diproses dengan Algoritma MD5	120	244
10		60	245
90		70	105
120		44	198
90		110	123
255		45	162
0		121	234
0		190	90
80		180	67
70		245	45
0		235	68
255		89	11
255		100	84
255		112	25
90		78	47
90		90	135
10			
178			
190			
240			

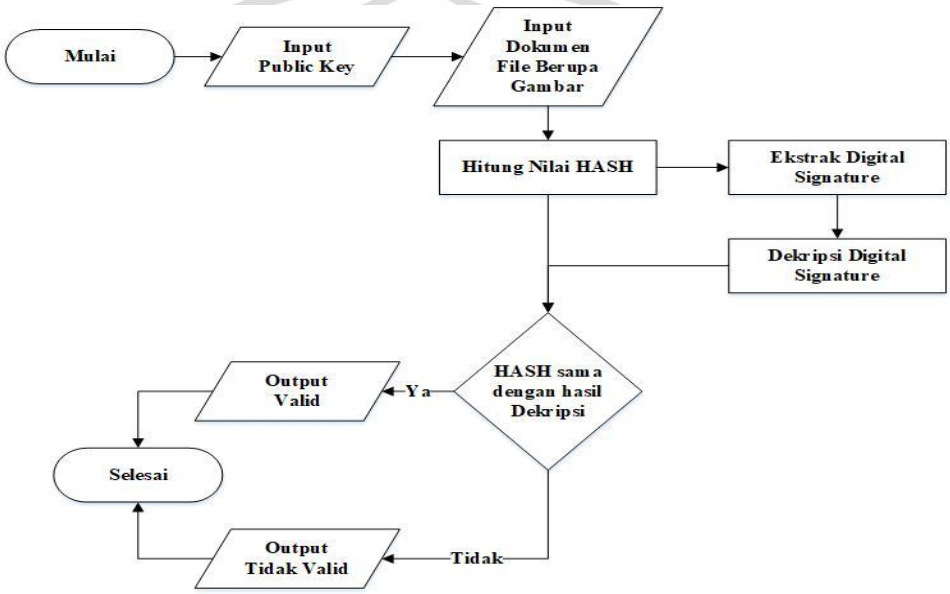
Pada Tahap kedua , dihasilkan *hash* terenkripsi : 244, 245, 105,198, 123, 162, 234, 90, 67, 45, 68, 11, 84, 25, 47, 135. *Hash* terenkripsi kemudian disisipkan ke dalam file gambar.

Kunci dekripsi hanya bisa digunakan saat proses dekripsi, sehingga tidak dapat digunakan untuk menyandikan sebuah nilai ringkasan. Kunci dekripsi bisa diberikan oleh pengirim kepada para pihak-pihak yang bermasksud untuk menggunakan informasi dari citra digital. Kunci dekripsi ini berupa sebuah proses

validasi keutuhan citra digital, dengan cara membandingkan hasil dekripsi dengan nilai ringkasan yang baru.

3.3.4 Proses Verifikasi *Digital signature*

Proses verifikasi adalah proses memastikan keaslian si pengirim. Artinya apakah yang mengirimkan itu asli dari orang yang bersangkutan. Berikut adalah alur proses verifikasi yang ditunjukan pada gambar 3.5.



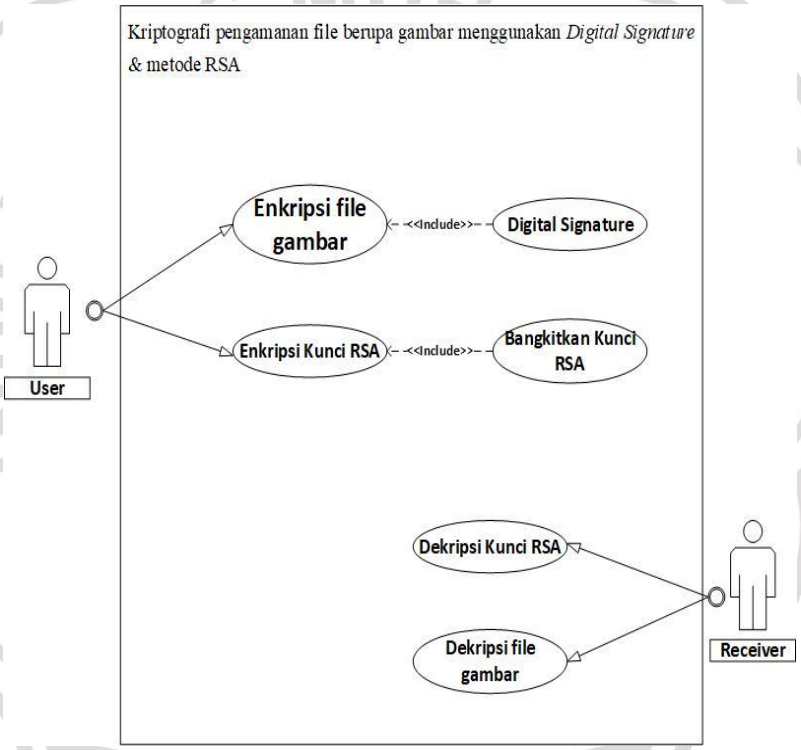
Gambar 3.5 Alur Proses Verifikasi *Digital signature*

Sama halnya dengan proses pemberian *digital signature***,**langkah dalam proses verifikasi pun sama dengan proses dekripsi pada algoritma RSA. Namun pada proses verifikasi,menggunakan kunci publik dari pengirim, berbeda dengan dekripsi algoritma RSA yang menggunakan pribadi dari penerima itu sendiri. Dalam proses verifikasi *hash* yang telah disisipkan, diekstrak kemudian didekripsi. Hasil pada dekripsi tersebut dibandingkan dengan *hash* file gambar sekarang. Jika nilai *hash* ini berbeda, maka dapat dipastikan bahwa file gambar tersebut telah mengalami sebuah perubahan.

3.4 Perancangan Sistem

3.4.1 Use Case Diagram

Use case diagram merupakan interaksi yang saling berkaitan antara sistem dengan aktor,*use case* digunakan untuk membentuk perilaku (*behavior*) sistem yang dibuat. *Use case diagram* pada gambar 3.6 ini akan menggambarkan interaksi sistem Kriptografi pengamanan file berupa gambar menggunakan *Digital signature* & metode RSA dengan penerima.



Gambar 3.6 *Use Case Diagram*

Pada gambar 3.6 terdapat 2 aktor, yaitu *User* dan *Receiver*. *User* dapat melakukan operasi enkripsi file gambar dan kuncinya. *Receiver* dapat melakukan operasi dekripsi file gambar dan *cipherkey*. Dokumentasi naratif *use -case* enkripsi file gambar, enkripsi kunci RSA, dekripsi kunci RSA, dekripsi file gambar,*digital signature* dan bangkitkan kunci RSA dapat dilihat pada Tabel 3.4., Tabel 3.5., Tabel 3.6.,Tabel 3.7., Tabel 3.8. dan Tabel 3.9.

Tabel 3.4 Dokumentasi naratif *use-case* enkripsi file gambar

Nama	Enkripsi Citra	
Aktor	Pengirim	
Deskripsi	Mendeskripsikan proses enkripsi file gambar	
Pre Kondisi	*User* menginput file gambar dan membangkitkan kunci RSA	
Post Kondisi	*User* dapat melihat file gambar hasil enkripsi	
	Kegiatan *User*	Respon Sistem
Skenario Normal	1. *User* menginput file gambar 2. *User* membangkitkan kunci RSA 3. *User* menekan tombol enkripsi	1. Sistem menampilkan file gambar 2. Sistem menampilkan kunci RSA 3. Sistem menampilkan file gambar hasil enkripsi
Skenario Alternatif	1. *User* menginput file gambar 2. *User* menekan tombol enkripsi	1. Sistem menampilkan file gambar 2. Sistem menampilkan pesan *error*

Tabel 3.5. Dokumentasi naratif *use-case* enkripsi kunci RSA

Nama	Enkripsi Kunci RSA
Aktor	Pengirim
Deskripsi	Mendeskripsikan proses enkripsi kunci RSA
Pre Kondisi	*User* membangkitkan kunci RSA

Post Kondisi	*User* dapat melihat kunci hasil enkripsi	
	Kegiatan *User*	Respon Sistem
Skenario Normal	1. *User* membangkitkan kunci RSA 2. *User* menekan tombol enkripsi	1. Sistem menampilkan kunci RSA 2. Sistem menampilkan kunci hasil enkripsi
Skenario Alternatif	1. *User* membangkitkan kunci RSA 2. *User* menekan tombol enkripsi	1. Sistem menampilkan kunci RSA 2. Sistem menampilkan pesan *error*

Tabel **3.6**. Dokumentasi naratif *use-case* dekripsi kunci RSA

Nama	Dekripsi Kunci RSA	
Aktor	Penerima	
Deskripsi	Mendeskripsikan proses dekripsi kunci RSA	
Pre Kondisi	*User* menginput *cipherkey* dan menginput kunci RSA	
Post Kondisi	*User* dapat melihat kunci RSA	
	Kegiatan *User*	Respon Sistem
Skenario Normal	1. *User* menginput *cipherkey* 2. *User* menginput kunci RSA 3. *User* menekan tombol dekripsi	1. Sistem menampilkan *cipherkey* 2. Sistem menampilkan kunci RSA 3. Sistem menampilkan kunci RSA

Skenario Alternatif	1. *User* menginput *cipherkey* 2. *User* menekan tombol dekripsi	1. Sistem menampilkan *cipherkey* *2.* Sistem menampilkan pesan *error*

Tabel 3.7 Dokumentasi naratif *use-case* dekripsi file gambar

Nama	Dekripsi File Gambar	
Aktor	Penerima	
Deskripsi	Mendeskripsikan proses dekripsi file gambar	
Pre Kondisi	*User* menginput file gambar dan menginput kunci RSA	
Post Kondisi	*User* dapat melihat file gambar hasil dekripsi	
	Kegiatan *User*	Respon Sistem
Skenario Normal	1. *User* menginput file gambar 2. *User* menginput kunci RSA 3. *User* menekan tombol dekripsi	1. Sistem menampilkan file gambar 2. Sistem menampilkan kunci RSA 3. Sistem menampilkan File gambar hasil dekripsi
Skenario Alternatif	1. *User* menginput file gambar 2. *User* menekan tombol dekripsi	1. Sistem menampilkan file gambar 2. Sistem menampilkan pesan *error*

Tabel 3.8. Dokumentasi naratif *use-case* bangkitkan kunci RSA

Nama	Bangkitkan Kunci RSA	
Aktor	Pengirim	
Deskripsi	Mendeskripsikan proses pembangkitan kunci RSA	
Pre Kondisi	*User* berada pada halaman enkripsi	
Post Kondisi	*User* dapat melihat kunci RSA	
	Kegiatan *User*	Respon Sistem
Skenario Normal	1. *User* menekan tombol bangkitkan kunci	1. Sistem menampilkan kunci RSA
Skenario Alternatif	1. *User* menekan tombol bangkitkan kunci	1. Sistem menampilkan pesan *error*

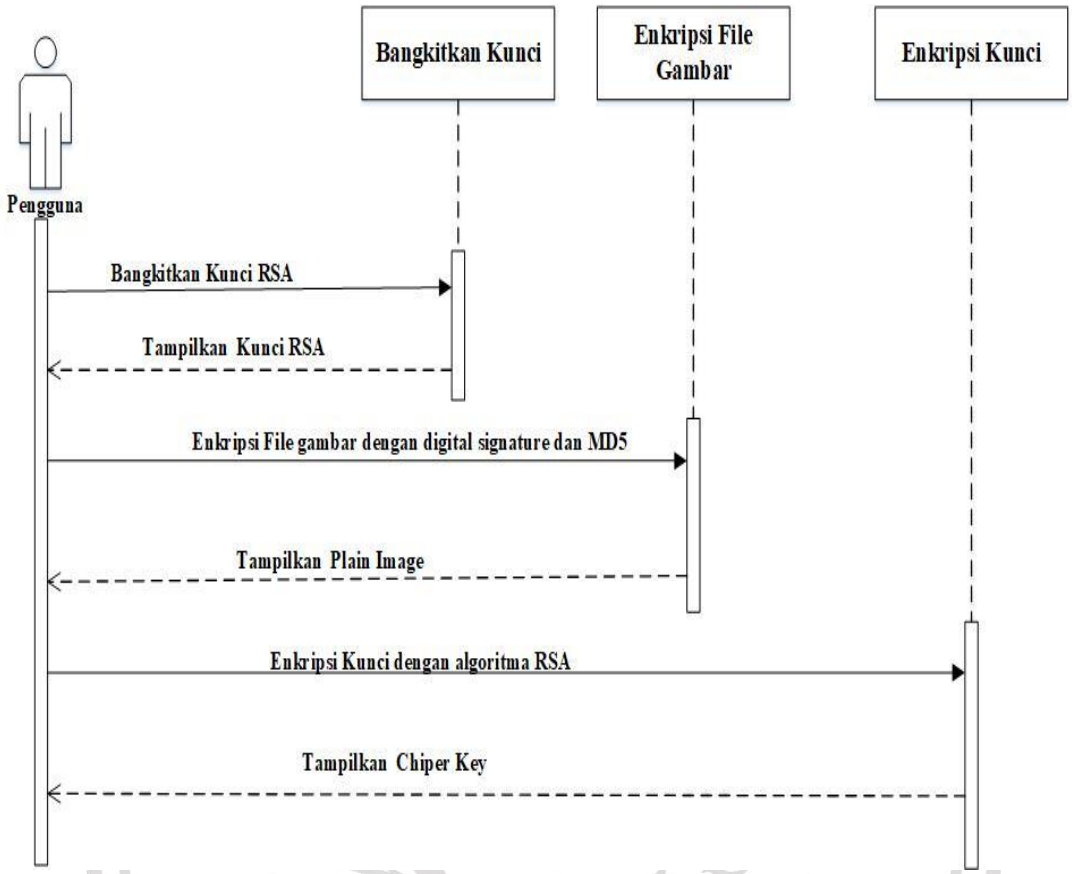
Tabel 3.9. Dokumentasi naratif *use-case digital signature*

Nama	Pemberian *Digital signature*	
Aktor	Pengirim	
Deskripsi	Mendeskripsikan proses pemberian *digital signature*	
Pre Kondisi	*User* berada pada halaman enkripsi	
Post Kondisi	*User* dapat melihat *digital signature*	
	Kegiatan *User*	Respon Sistem
Skenario Normal	1. *User* menekan tombol enkripsi	1. Sistem menampilkan *Digital signature*
Skenario Alternatif	1. *User* menekan tombol enkripsi	1. Sistem menampilkan pesan *error*

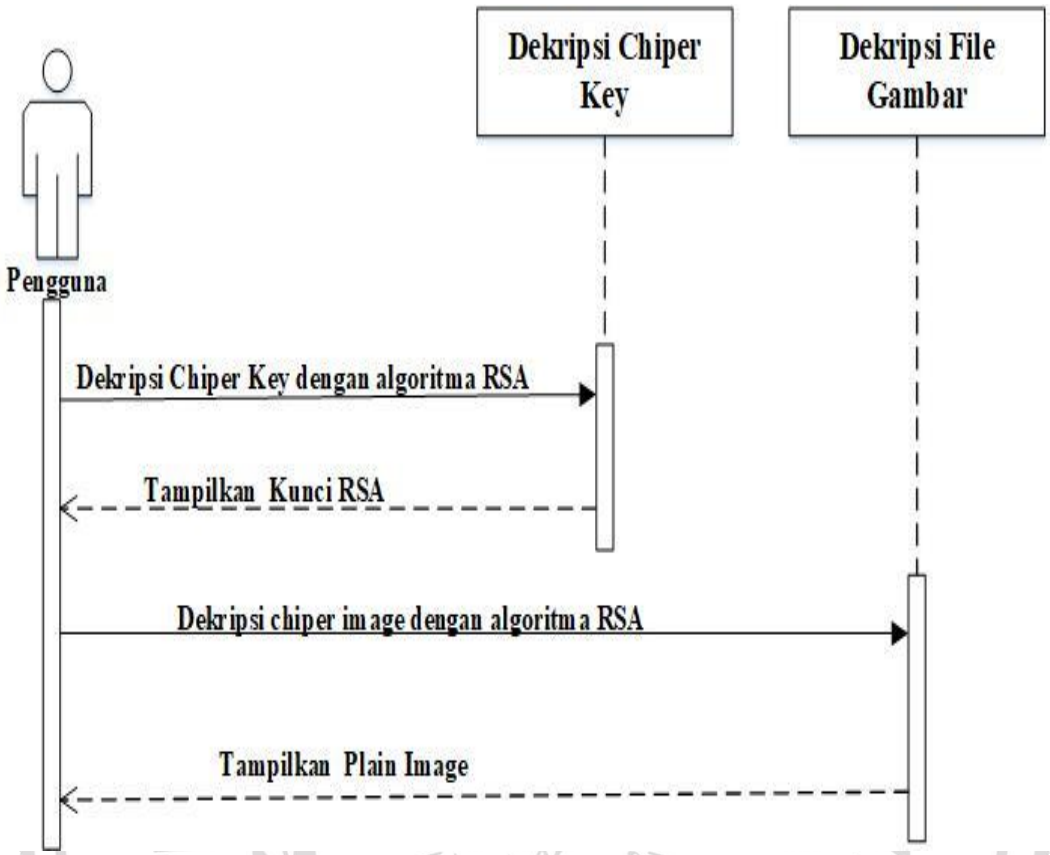
3.4.2 Sequnce diagram

Sequence diagram adalah diagram yang menjelaskan interaksi pesan dalam serangkaian waktu yang terjadi dalam suatu sistem. *Sequence diagram* menunjukkan peran dan pesan yang dikirim maupun diterima oleh instansi/objek yang menjalankan peran tersebut (Booch et al, 2005). *Sequence*

diagram dari sistem yang akan dibangun dapat dilihat pada gambar 3.7 dan gambar 3.8



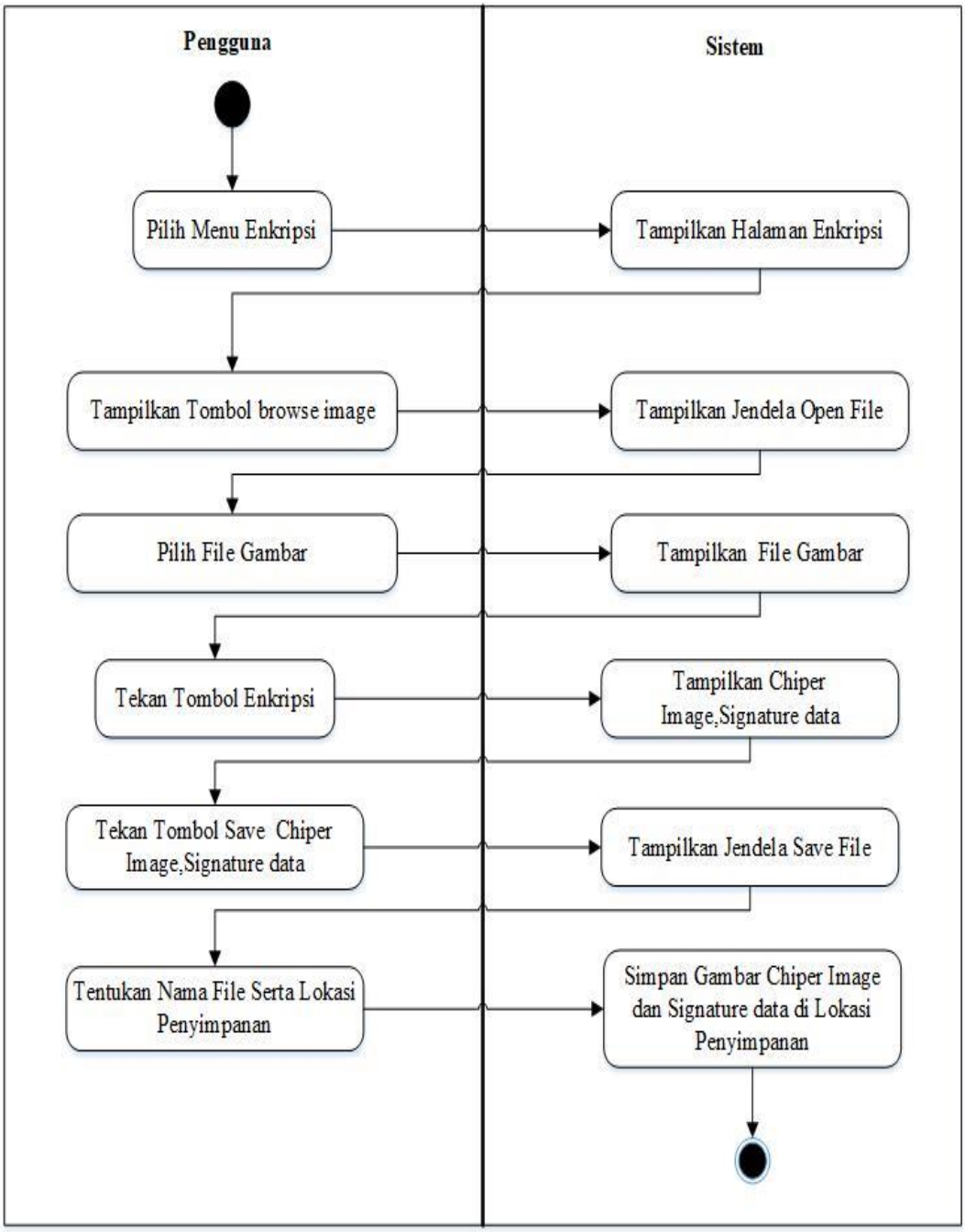
Gambar 3.7 *Sequence diagram* proses enkripsi



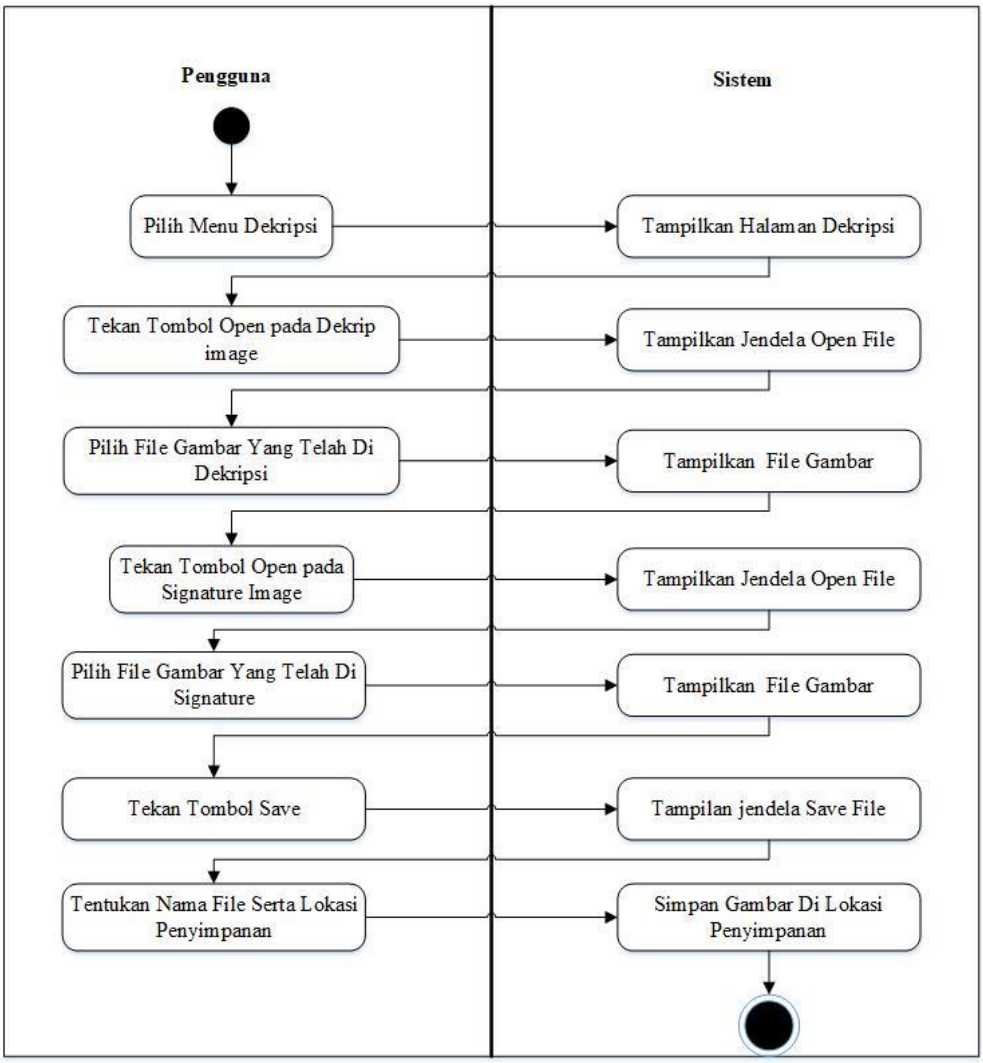
Gambar 3.8 *Sequence diagram* proses dekripsi

3.4.3 Activity Diagram

Activity diagram adalah diagram yang menunjukkan aliran kontrol dari suatu aktivitas ke aktivitas lainnya dalam suatu sistem. Activity diagram pada dasarnya dalah sebuah *flowchart* (Booch et al, 2005). *Activity diagram* dari sistem yang akan dibangun dapat dilihat pada gambar 3.9 dan gambar 3.10.



Gambar 3.9. *Activity diagram* proses enkripsi



Gambar 3.10. *Activity diagram* proses dekripsi

3.5 Kebutuhan Pembuatan Sistem

Dalam melakukan analisis dan perancangan sebuah sistem diperlukan perangkat keras (*hardware*) dan perangkat lunak (*software*) sebagai berikut :

3.5.1 Spesifikasi Perangkat Keras (Hardware)

Sistem Perangkat keras (*Hardware*) adalah komponen-komponen pendukung kinerja dari sistem komputer. Adapun Spesifikasi perangkat keras (*hardware*) yang dipakai dalam membuat Sistem pencatatan surat masuk dan keluar di Badan Narkotika Nasional Kabupaten Gresik adalah :

1. Prosesor Intel Core i3-4030U 1,9 GHz
2. Memory (RAM) 6 GB
3. Hardisk 500 GB
4. Mouse
5. Keyboard
6. Monitor

3.5.2 Spesifikasi Perangkat Lunak (Software)

Perangkat lunak (*Software*) adalah suatu sistem yang terkomputerisasi berupa program-program yang berfungsi menjalankan perangkat keras yang berupa sistem operasi, bahasa pemrograman dan aplikasi. Adapun perangkat lunak (*Software*) yang diperlukan dalam pembuatan Kriptografi pengamanan datafile berupa gambar menggunakan metode RSA dan *Digital Signature* adalah sebagai berikut :

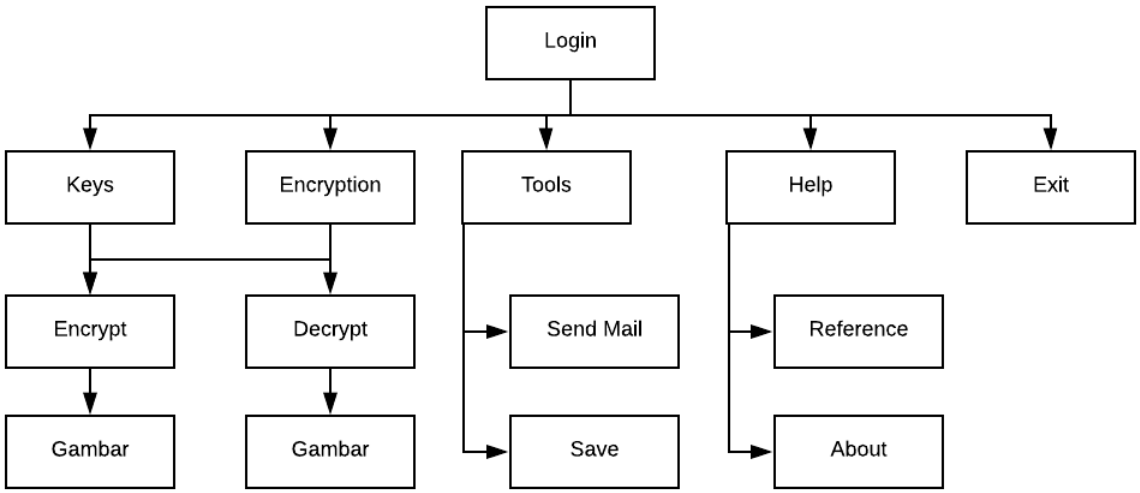
1. Sistem Operasi Windows 10
2. Microsoft Visio

   Microsoft Visio merupakan salah satu aplikasi yang digunakan untuk membuat diagram, diagram alir (*flowchart*), brainstorm, dan skema jaringan. Aplikasi ini menggunakan grafik vektor untuk membuat diagram-diagramnya.
3. Microsoft Visual Studio

   Microsoft visual studio adalah *software* yang digunakan untuk memprogram aplikasi dengan bahasa pemrograman yang meliputi C, python, basic. Untuk bahasa pemrograman penulis menggunakan bahasa C.

3.6 Struktur model sistem

Pada Struktur model sistem ini terdapat beberapa menu dalam aplikasi yang akan dibuat. Menu tersebut dapat dilihat pada Gambar 3.11.

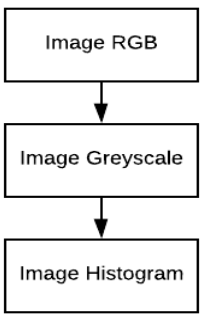


Gambar 3.11 Struktur model sistem

Dari struktur model diatas terdapat lima menu pada aplikasi ini. Yaitu menu keys, encryption, tools, help dan exit. Diantara menu menu tersebut terdapat sub-sub menu yang malakukan proses untuk mendukung menu tersebut. berikut penjelasan algoritma yang diimplementasikan kedalam form masing-msing menu dan sub-sub menu.

3.7 Pre Processing

Pada bab ini terdapat proses pengubahan gambar sebelum dijadikan *encrypsi*, perubahan gambar tersebut digunakan untuk mempermudah proses *decrypt* dan menambah tingkat akurasi dari proses *decrypt*. Pre processing dapat dilihat pada Gambar berikut.

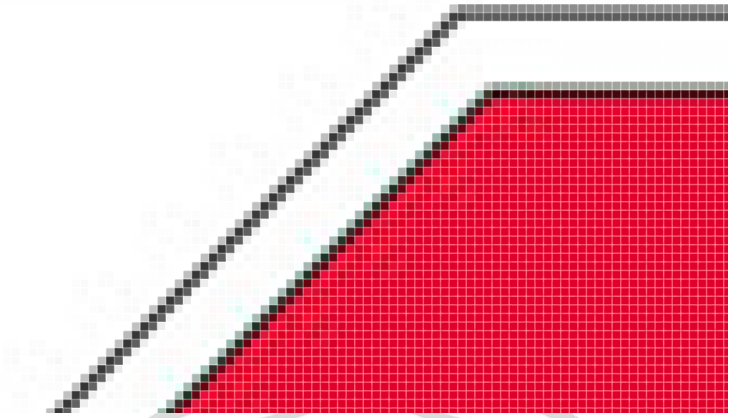


Gambar 3.12 Pre-processing

Pada Gambar 3.7 terdapat proses untuk perubahan gambar Proses yang pertama adalah proses memasukkan gambar dengan tiga ruang warna yaitu RGB atau biasa dikenal dengan red, green dan blue. Jika proses ini langsung digunakan untuk encrypsi maka tingkat keakurasian akan berkurang karena tidak ada intensitas cahaya dari gambar. Contoh gambar RGB dapat dilihat pada Gambar 3.13. Untuk merubah ke histogram maka dibutuhkan proses greyscale dimana proses greyscale adalah pencarian 1 ruang warna sehingga dapat lebih mudah untuk diproses. Setelah mengubah ke ruang warna greyscale maka akan dirubah pada histogram dimana histogram adalah proses untuk menentukan intensitas cahaya pada Gambar. proses dapat dilihat pada Gambar 3.13.

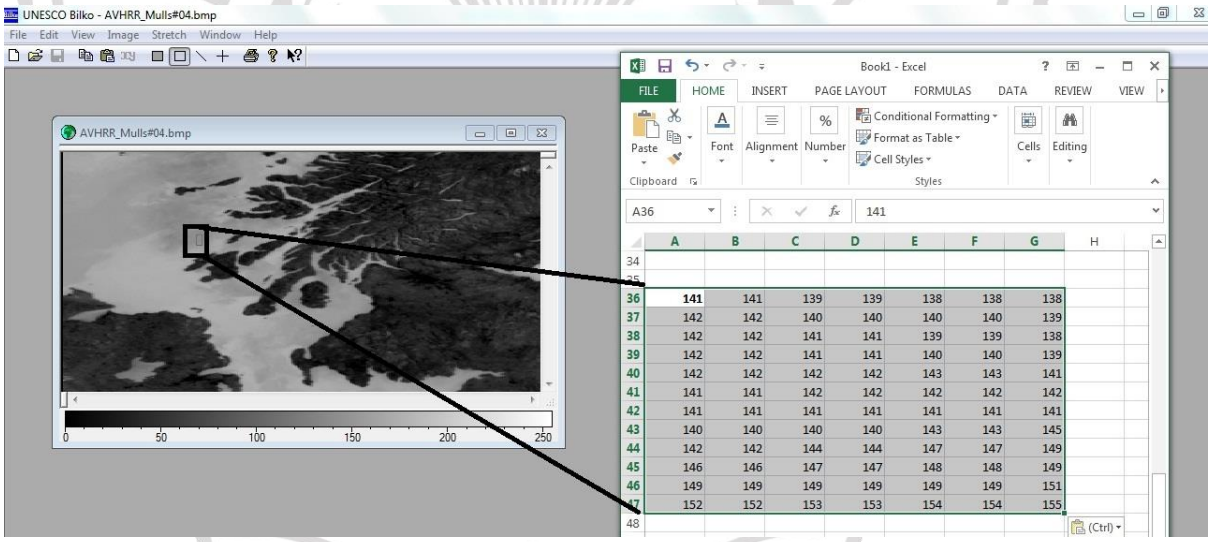
Gambar 3.13 Gambar RGB (304 x 304)

Pada Gambar tersebut memiliki dimensi 304 x 304 yang artinya memiliki pixel 304 baris dan 304 kolom. Untuk membuktikan pixel itu ada maka gambar diatas dapat diperbesar sehingga mengeluarkan baris dan kolom pada gambar. Contoh pixel dapat dilihat pada Gambar dibawah ini.



Gambar 3.14 Perbesar Gambar

Pada Gambar 3.14 terdapat proses perbesar gambar sehingga mendapatkan gambar yang mempunyai baris dan kolom dimana pada baris dan kolom tersebut mempunyai nilai berkisar antara 0 – 255. Contoh nilai pada pixel dapat dilihat pada Gambar 3.15.



Gambar 3.15 pixel pada gambar

Pixel tersebut akan diproses satu persatu untuk mendapatkan pre processing pada image processing.

Proses pertama adalah mengambil nilai R, G dan B dari suatu citra bertipe RGB. Pada tipe .bmp citra direpresentasikan dalam 24 bit, sehingga diperlukan

proses untuk mengambil masingmasing 3 kelompok 8 bit dari 24 bit tadi. Sebagai contoh suatu pixel memiliki nilai RGB 24 bit sebagai berikut :

111100001111000011111111, untuk mendapatkan masing-masing nilai R, G dan B dilakukan operasi-operasi sebagai berikut. Untuk mendapatkan nilai R dilakukan operasi modulo dengan bilangan 256 sebagai berikut :

Nilai R = 111100001111000011111111 mod 10000000 = 11111111

Sedangkan untuk nilai G, dapat dicari dengan cara sebagai berikut:

Nilai G = (111100001111000011111111 and 1111111100000000)/100000000 = 11110000

Untuk Nilai B, dapat dicari dengan menggunakan rumus

Nilai B= (111100001111000011111111 *and* 1111111100000000000000000)/10000000000000000 = 11110000

sehingga dari nilai pixel 111100001111000011111112 atau 15790335 diperoleh nilai R = 11111111 = 255 G = 11110000 = 240 B = 11110000= 240

Sehingga diperoleh triplet RGB= (255,240,240).

Setelah nilai triplet RGB kita peroleh, maka kita bisa mendapatkan nilai grayscale dari pixel tersebut.

Ide dasarnya sebenarnya adalah membuat band tunggal dari 3 band RGB tadi dengan rumus tertentu.

Pada penelitian ini digunakan rumus :

red = (red * 5) \ 10 green = (green * 8) \ 10 blue = (blue * 3) \ 10

gray = ((red + green + blue) * 10) \ 16

Dengan mengaplikasikan prosedur tadi pada semua pixel akan kita dapatkan citra dengan format grayscale. Contoh citra greyscale dari hasil perhitungan tersebut terdapat pada Gambar 3.16.

Gambar 3.16 Gambar Greyscale

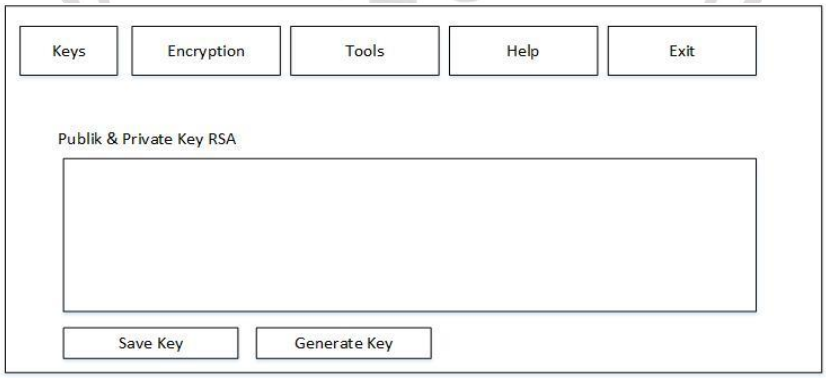
Gambar greyscale tersebut akan diproses untuk mencari histogram dimana rumus untuk mencari histogram adalah mengubah gambar menjadi grafik pixel.

3.8 Perancangan Antarmuka Sistem

Pada rancangan desain antarmuka sistem terdapat rancangan tampilan aplikasi yang akan dibuat. Mulai dari interaksi tampilan, inputan, proses dan output.

3.8.1 Rancangan Antarmuka Form Pembentukan Kunci

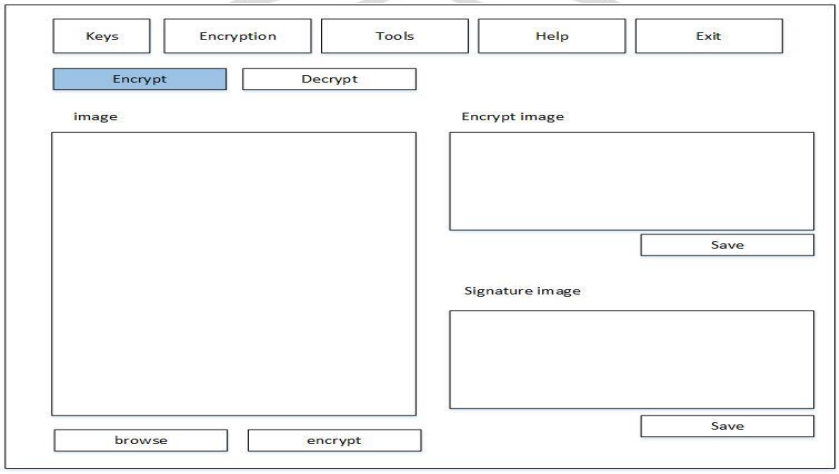
Form ini dirancang untuk membuat pasangan kunci publik dan kunci privat sebelum pengguna melakukan proses enkripsi/dekripsi dan *signature* data. Pada form ini, terdapat fungsi untuk membuat pasangan kunci RSA dimana pengguna tidak perlu menginputkan kunci secara manual tetapi secara otomatis dibuat langsung dari sistem. Berikut gambar tampilan *form key* :



Gambar 3.17 Form Pembentukan Kunci

3.8.2 Rancangan Antarmuka Form Enkripsi

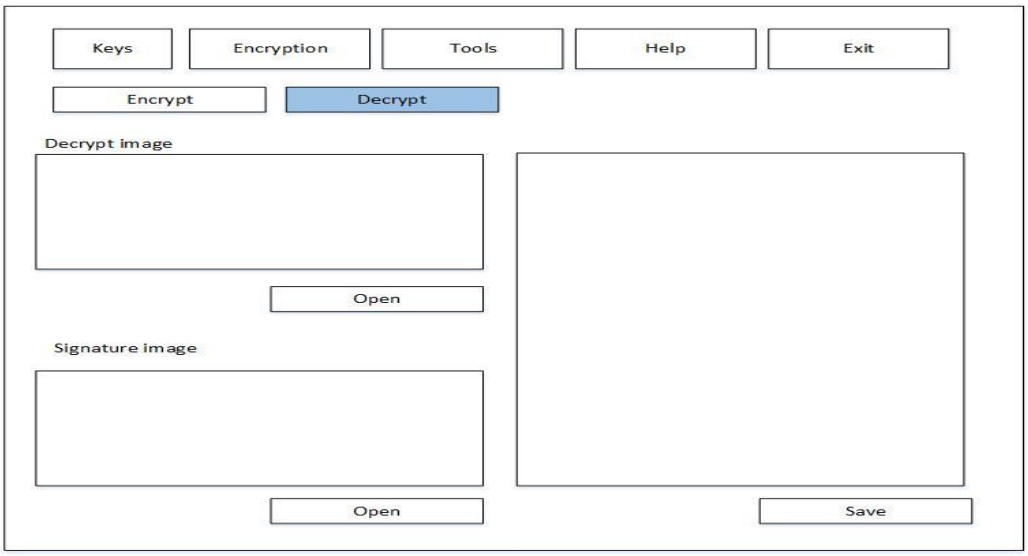
Form ini berfungsi mengenkripsi dan memberi *signature* file gambar yaitu menggunakan algoritma RSA dan menerapkan algoritma *hash*ing MD5. Output yang dihasilkan dari form ini adalah ciphertext dan *signature* data yang secara terpisah. Form ini didesain dengan semenarik mungkin dan memperhatikan pada aspek *user friendly*



Gambar 3.18 Form Enkripsi

3.8.3 Rancangan Antarmuka Form Dekripsi

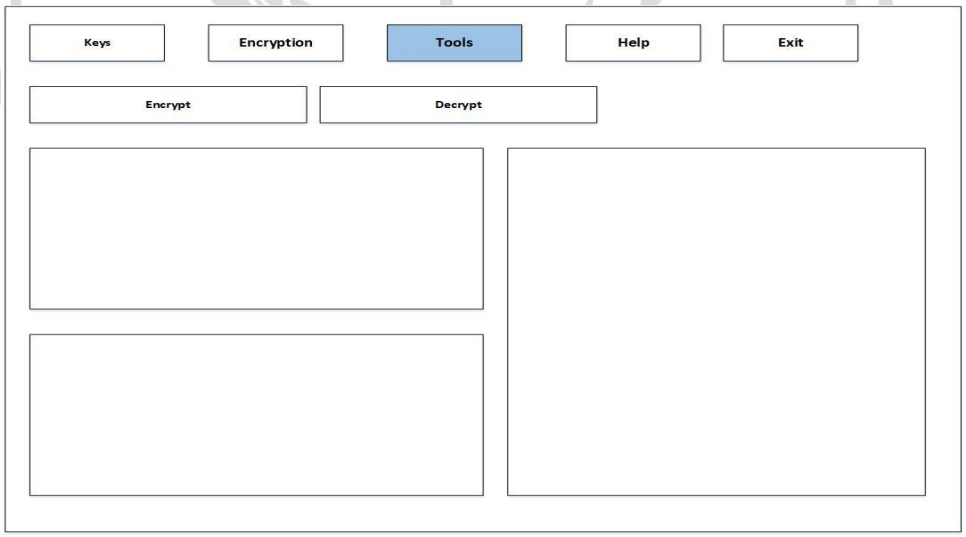
Form Dekripsi digunakan untuk mengembalikan file gambar yang telah dienkripsi ke bentuk gambar aslinya, maka dilakukan proses dekripsi. Namun tidak sekedar melakukan dekripsi tetapi juga melakukan verifikasi pada *signature* gambar yang telah dibuat, untuk mengetahui apakah file gambar mengalami perubahan atau tidak.



Gambar 3.19 Form 4 Dekripsi

3.8.4 Rancangan Antarmuka Form Tools

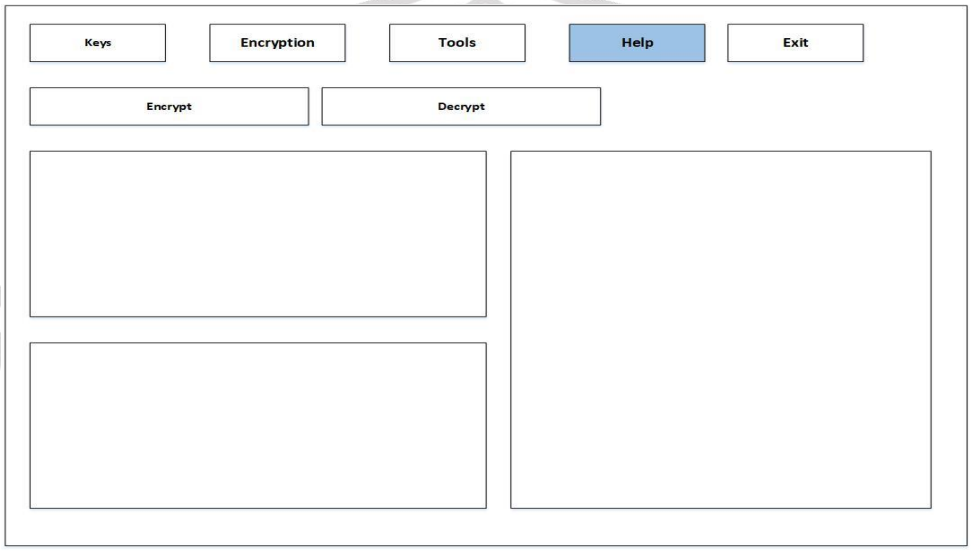
Rancangan antarmuka form tools adalah isi dari menu tools dimana terdapat send Email, send Whatshapp, send telegram serta save. Dimana masing-masing menu adalah menu untuk mengirim hasil *encrypsi* atau menyimpan hasil *encrypsi*. Berikut rancangan antar muka menu *tools* :



Gambar 3.20 Menu *tools*

3.8.5 Rancangan Antarmuka Form *Help*

Rancangan antar muka menu *help* digunakan untuk memberikan informasi tentang kendala pada aplikasi. Pada menu tersebut terdapat sub menu *About* yang digunakan untuk memberikan informasi tentang aplikasi dan menu *send message to Developer* adalah untuk memberikan kritik atau pesan untuk pemrogram aplikasi. Berikut rancangan antar muka menu *tools* :



Gambar 3.21 Menu *Help*

3.9. Skenario Pengujian

Untuk mengukur evaluasi kinerja sistem pengamanan file, yaitu menggunakan otentikasi dan integritas .

3.9.1 Pengujian RSA

Dalam pengujian metode RSA di butuhkan pengujian otentikasi yaitu dengan menguji apakah dengan pasangan kunci yang berbeda proses verifikasi dapat dilakukan. Keutuhan (*integrity*) suatu dokumen diuji dengan cara melakukan perubahan (manipulasi) pada dokumen dimana dibutuhkan 2 buah kunci *public* dan *private*

yang berbeda untuk perbandingan dalam menguji sistem yang akan di gunakan nanti..

3.9.2 Pengujian *Digital signature*

Dalam pengujian metode *Digital signature* di butuhkan pengujian integritas dimana bertujuan untuk mengetahui apakah aplikasi tersebut bisa mendeteksi sebuah perubahan pada dokumen yang telah diberi *digital signature*. *Digital signature* dapat disimpulkan berhasil dalam menjaga sebuah keotentikan dokumen jika perubahan yang dilakukan pada dokumen dapat terdeteksi (hasil verifikasi "tidak valid"). Dimana dalam pengujian ini di butuhkan 1 file gambar dengan jenis perubahan yaitu *crop,resize image*,manipulasi file gambar untuk mengetahui tingkat keamanan dari sistem pengamanan file berupa gambar apabila gambar mengalami perubahan.