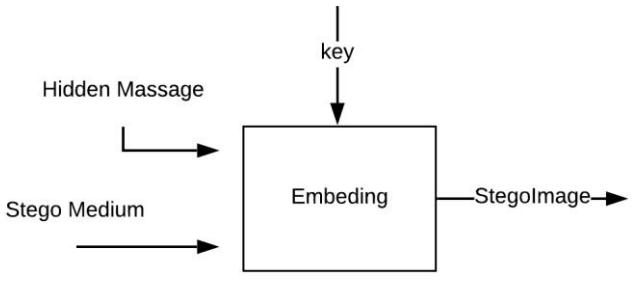
BAB III

ANALISI DAN PERANCANGAN SISTEM

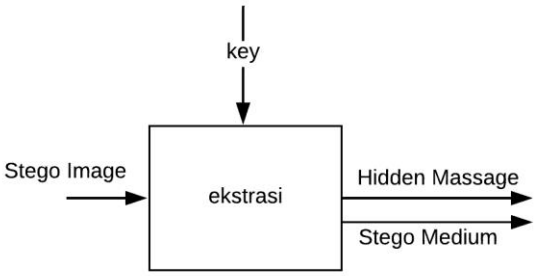
3.1. Analisis Sistem

Permasalahan yang ditimbulkan dalam penyembunyian informasi penting yang aman adalah menyembunyikan pesan rahasia agar tidak menimbulkan kecurigaan untuk orang yang melihatnya. Pada tahapan pengambilan data penampung gambar (*cover image*) dilakukan secara langsung melalui internet. Setelah itu citra penampung gambar melalui proses *embedding massage,* setelah berhasil melalui tahap tersebut maka akan menghasilkan *stego image* dimana hasil citra tersebut adalah sebuah citra yang sudah di sisipi pesan tersembuyi.

Pada umumnya, terdapat dua tahapan dalam steganografi, yaitu proses penanaman pesan (*embedding*) untuk menyimpan pesan yang tersembunyi, dan proses ekstraksi (*extraction*) untuk mengambil kembali pesan yang telah disimpan tersebut. Proses penyisipan (*embedding*) digunakan untuk menyembunyikan pesan rahasia di dalam media penyimpanan. (*stegomedium*) diilustrasikan pada Gambar 3.1. Bagian awal dari proses ini melibatkan penggunaan sebuah kunci tertentu (*key*) untuk menyembunyikan pesan tersebut, sehingga dihasilkan stegoimage yang berisi pesan tersembunyi. Gambar 3.2 menunjukkan proses ekstraksi dari stegoimage dengan memasukkan key yang sama, sehingga pesan rahasia tersembunyi kembali ditemukan. Berikut tampilan gambar 3.1 dan gambar 3.2 :



Gambar 3.1 *Embedding* Citra



Gambar 3.2 Retrieving Citra

Keterangan : ⟶ *= Input/Output*

Proses *embedding* pesan pada gambar memiliki fungsi untuk menyembunyikan pesan rahasia sehingga tidak diketahui oleh orang awam. Dalam pembuatan perangkat lunak, Dalam penelitian ini, digunakan bahasa pemrograman MATLAB (R2017a) sebagai alat bantu untuk menyelesaikan masalah yang ada.

3.2. Hasil Analisis

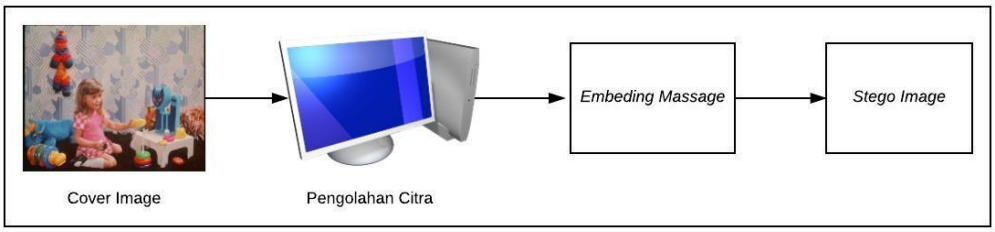
Sistem penyisipan pesan pada gambar dapat memberikan hasil analisis yang berguna dalam pengiriman pesan rahasia secara aman. tanpa menimbulkan kecurigaan kepada orang yang tidak berkepentingan menerima pesan tersebut. Selanjutnya dilakukan proses penyisipan pesan pada gambar menggunakan metode LSB (*Least Significant Bit*) dimana pesan tersebut di ubah menjadi nilai biner dan disisipkan pada bit terendah gambar penampung (*Cover Image*). Berdasarkan penjelasan di atas, judul untuk skripsi ini adalah **"Rancang Bangun Aplikasi Enkripsi Informasi Pada Citra Digital Dengan Algoritma Steganografi Menggunakan Metode Least Significant Bite".**

3.3. Deskripsi Sistem

Tujuan dari deskripsi sistem adalah untuk memberikan sebuah gambaran menyeluruh terkait aplikasi yang sudah atau akan dibuat dan perangkat keras yang diperlukan. Fungsinya adalah untuk mendukung pembuatan aplikasi dengan cara mengidentifikasi kebutuhan aplikasi yang sebelumnya.

3.3.1 Gambaran Umum Sistem

Untuk menciptakan suatu sistem, tahapan perancangan sistem sangat penting dilakukan. Tujuan dari perancangan sistem adalah untuk memberikan penjelasan secara menyeluruh tentang bagaimana proses dimulai dan bagaimana mengatasi masalah yang mungkin muncul. Berikut ini adalah ikhtisar dari rancangan sistem tersebut :

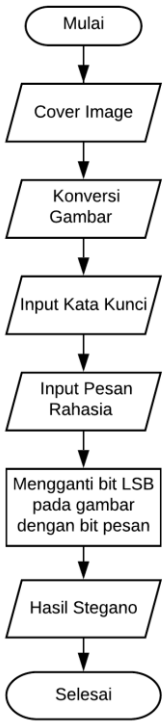


Gambar 3.3 Perancangan Umum Alur Sistem

Gambar nomor 3.3 yang terlihat di atas menunjukkan sebuah sistem yang akan diciptakan dengan menggunakan sebuah citra cover image yang mempunyai ekstensi BMP dan mempunyai resolusi sebesar 720 x 576 piksel sebagai sumber bahan untuk menyisipkan sebuah pesan tersembunyi. Pada kesempatan ini, sistem ini akan menggunakan MATLAB adalah bahasa pemrograman yang digunakan sebagai alat untuk memproses data digital dan berjalan pada sistem operasi Windows 10 Pro 64-bit. Selanjutnya, dapat melanjtkan proses penyisipan pesan tersembunyi pada *cover image* untuk menghasilkan sebuah stego image yang telah menyimpan pesan rahasia dalam citra tersebut.

3.4. Perancangan Sistem

Flowchart bertujuan untuk memberikan gambaran tentang program yang akan dibuat dalam penelitian ini, dengan fokus pada bagaimana pengolahan citra diterapkan untuk memproses pesan teks dan memberikan kemampuan untuk menyembunyikan pesan rahasia. Dapat dilihat pada gambar 3.4.

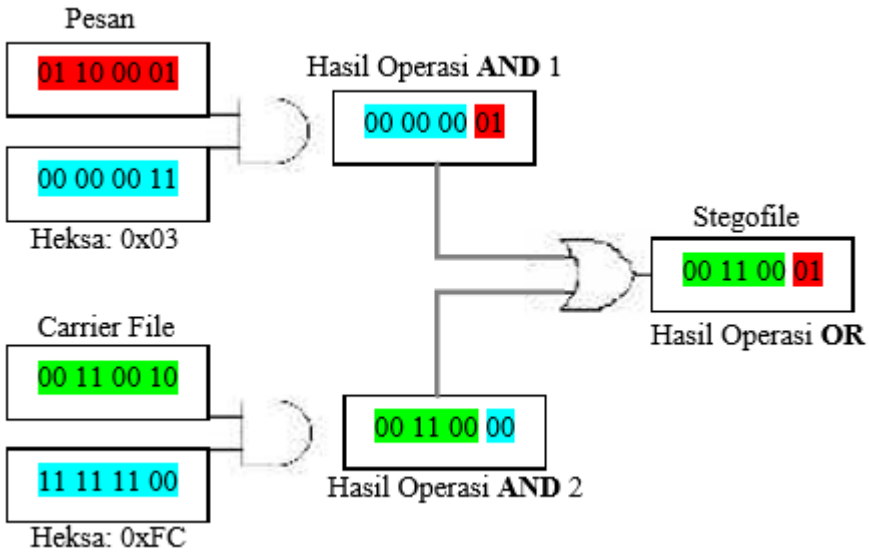


Gambar 3.4 *Flowchart* Proses penyisipan pesan tersembunyi pada gambar

Berdasarkan gambar 3.4 berikut penjelasan proses atau langkah-langkah steganografi dalam menyembunyikan pesan tersembunyi. menggunakan metode LSB :

1. Melakukan input data digital yang berperan sebagai wadah untuk menyimpan pesan rahasia..
2. Kemudian konversi gambar ke bentuk citra biner
3. Inputkan kata kunci
4. Proses penyisipan pesan tersembunyi dilakukan.

5. Pada proses penyisipan, bit terakhir pada citra cover akan diganti dengan bit dari pesan tersembunyi yang ingin disisipkan.
6. Jika berhasil, maka akan dihasilkan file stagnografi *(stego image).*



Gambar 3.5. Diagram Blok Metode LSB untuk Proses *Embedding*

Berikut contoh proses steganografi, misalkan bilangan biner pada kotak dibawah ini berisi *text* pesan yang akan disembunyikan. Misal isi pesan tersebut adalah "KUDA"

01001011	01010101	01000100	01000001

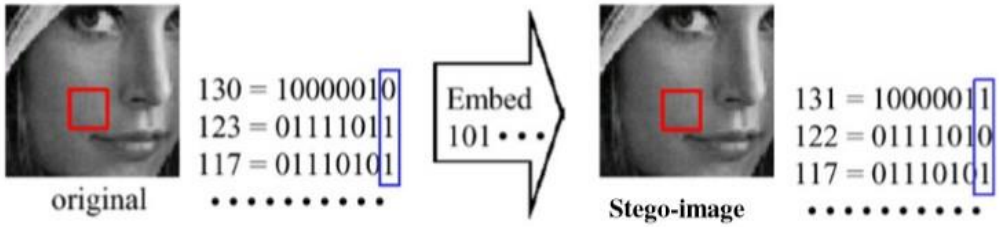
Biner – biner di atas merupakan angka biner dari kata "KUDA", disini penulis mengambil salah satu huruf di atas yaitu huruf " K " memiliki nilai biner (**01001011**), Sehingga piksel-piksel pada citra cover akan mengalami perubahan seperti :

1100001**0** 1101001**1** 1100001**0** 1010011**0** 0001110**1** 0111100**0** 0111011**1**
0100000**1** 10111011 01001110 11001100 00111010 01100000 11101101
10100111 00001100 00011100 11111010 11110000 11100001 10110011

10000010 01111011 01110101
(130) (123) (117)

Bit Bit-bit *embedded message :* **101**

Embed : 1000001**1** 0111101**0** 0111010**1**



Contohnya, jika piksel (130, 123, 117) pada citra cover memiliki warna ungu, maka pada stego-image piksel (131, 122, 117) tetap memiliki warna ungu namun dengan perubahan yang sangat kecil. Perubahan warna yang sangat kecil, misalnya pergeseran warna sebesar 1 dari total 256 warna, tidak dapat terdeteksi oleh mata manusia.

Jika pesan memiliki panjang 10 bit, maka akan digunakan 2 byte untuk menyimpan pesan tersebut. Berikut adalah contoh susunan byte yang lebih Panjang :

00110011 10100010 11100010 10101011 00100110
10010110 11001001 11111001 10001000 10100011

Pesan : 1110010111

Hasil dari penyisipan pesan pada bit LSB :

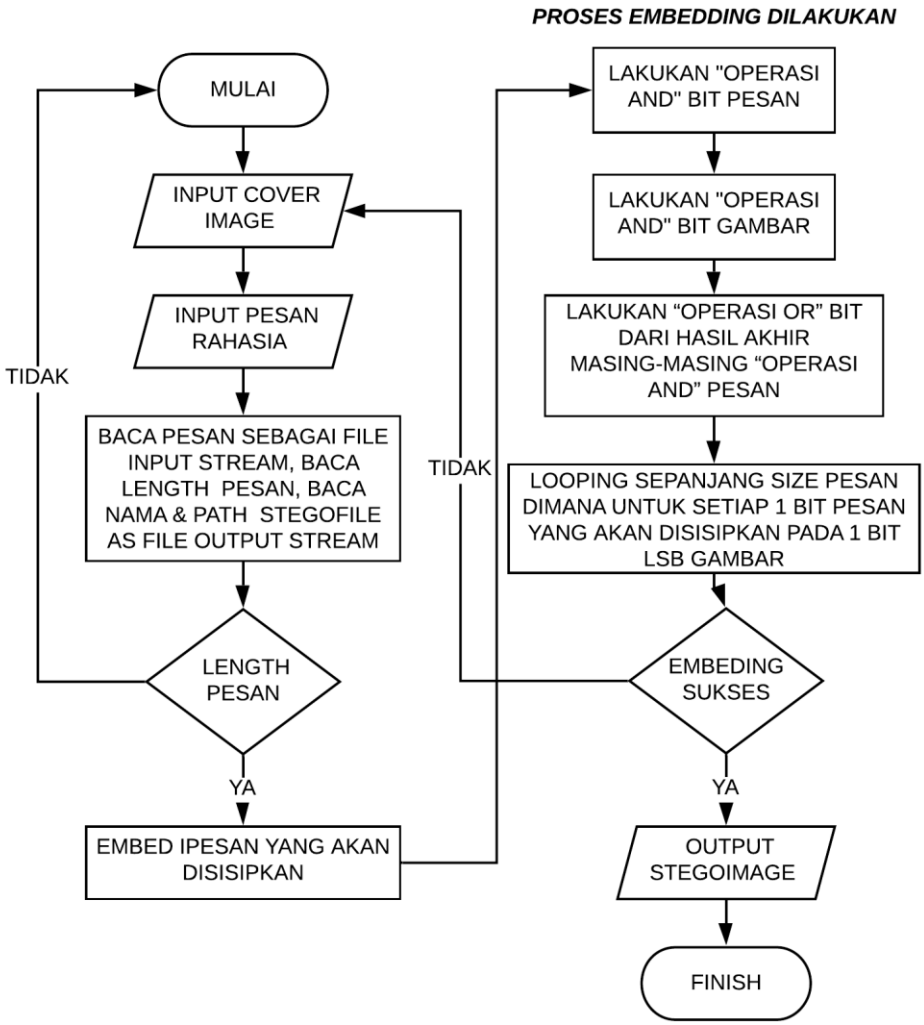
00110011 10100011 11100011 10101010 00100110
10010111 11001000 11111001 10001001 10100011

Besar pesan yang dapat disembunyikan tergantung pada ukuran cover-object. Dalam contoh citra grayscale dengan resolusi 256 x 256 piksel, tiap piksel memiliki 1 byte, sehingga jumlah total byte adalah 256 x 256 = 65536. Dengan menyisipkan 1 bit pesan pada LSB setiap byte, maka ukuran maksimal pesan adalah 65536 bit, atau 8192 byte, atau 8 KB. Sementara pada citra berwarna 24-bit berukuran 256 x 256 piksel, setiap piksel memiliki 3 byte, sehingga jumlah total byte adalah 65536 x 3 = 196608. Setiap byte bisa menyembunyikan 1 bit pesan, sehingga ukuran maksimal pesan adalah 196608 bit, atau 24576 byte, atau 24 KB.

3.4.1. Penerapan Metode Penyembunyian Pesan (*Embedding*)

Dalam proses *embedding* pesan rahasia, metode LSB digunakan dengan memasukkan file yang diperlukan dan membaca bit terakhir dalam gambar. Nilai biner pesan tersebut akan disisipkan pada bit terakhir. Proses ini dapat terlihat pada gambar 3.6 sebagai bagian dari pembangunan aplikasi steganografi.

Gambar 3.6. Alur proses *embedding* pesan



Berdasarkan flowchart gambar 3.6 proses *embedding* pesan dijelaskan bahwa :

1. Proses penyembunyian pesan yang rahasia dimulai dengan memasukkan data yang diperlukan ke dalam sistem, termasuk

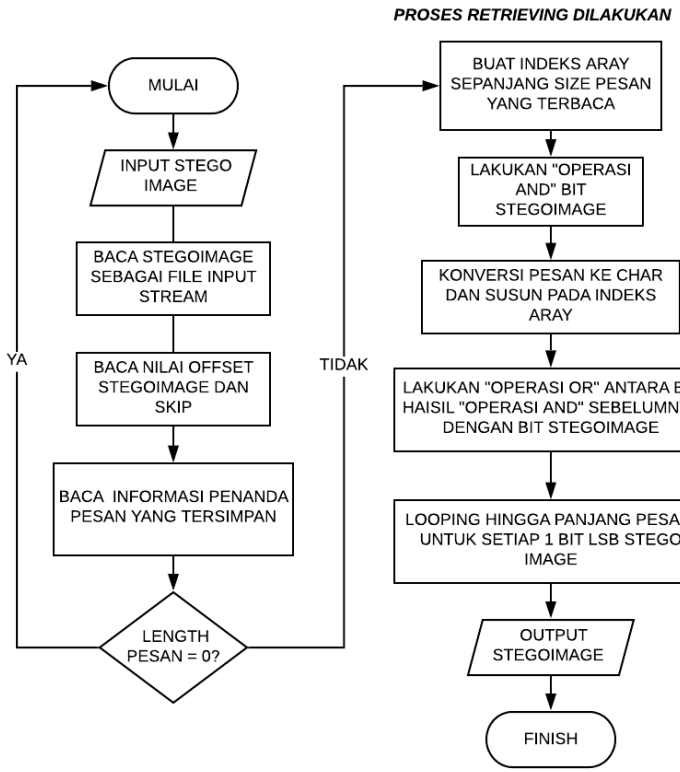
pesan yang ingin disembunyikan dan tempat di mana output harus disimpan.

2. Untuk menyisipkan pesan rahasia, aplikasi akan menjalankan beberapa tahapan, dimulai dengan membaca input data berupa pesan rahasia dan lokasi output. Setelah itu, program akan membaca aliran data dari gambar sampul dan pesan yang akan disisipkan sebagai aliran masukan file, Setelah membaca aliran data dari nama dan path stegofile sebagai aliran keluaran file, dilakukan verifikasi untuk menjamin bahwa ukuran gambar sampul memiliki ukuran yang sama atau lebih besar daripada pesan rahasia yang disematkan. Jika ukuran gambar cover memenuhi persyaratan, maka proses akan dilanjutkan dengan membaca nilai bit gambar cover. Namun, jika ukuran gambar cover lebih kecil, Jika tidak memenuhi kriteria validasi, maka aplikasi akan kembali ke tahap awal dan meminta input gambar sampul yang lebih besar.
3. Langkah berikutnya dalam proses penyisipan terjadi pada bit setelah OFFSET adalah sebagai berikut :
    a) Langkah pertama dalam proses ini melibatkan penggunaan operasi *AND* pada bit-bit pesan dengan nilai heksadesimal 0x03 yang telah dikonversi ke dalam bentuk biner ialah (00000011).
    b) Lakukan operasi *AND* antara bit-bit gambar cover dan nilai heksadesimal 0xFC (bentuk biner : 11111100).
    c) Langkah berikutnya adalah menjalankan operasi OR pada setiap keluaran hasil dari operasi *AND* antara bit-bit pesan dan gambar sampul. Tujuannya adalah untuk mengganti bit pesan dengan 1 bit LSB dari gambar sampul.

4. Ditemukan hasil berupa stegoimage (file baru yang mengandung pesan tersembunyi). .
5. Selesai.

3.4.2. Proses Ekstrasi Pesan (*Retrieving*)

Tahap pengambilan melibatkan upaya untuk mengekstrak kembali pesan rahasia yang disembunyikan pada gambar stego sebelumnya. Detail langkah-langkah dari proses ini dijelaskan secara visual pada Gambar 3.7.



Gambar 3.7. *Flowchart* Proses *Retrieving*

Langkah-langkah proses yang ditunjukkan dalam Gambar 3.7 diterangkan sebagai berikut :

1. Proses *Retrieving* terjadi jika data yang dibutuhkan, yaitu stegoimage, telah dimasukkan ke dalam sistem.
2. Aliran data dari gambar stego yang dimasukkan akan dibaca oleh sistem.
3. Melakukan proses skip atas nilai OFFSET dari stegoimage yang dibaca.

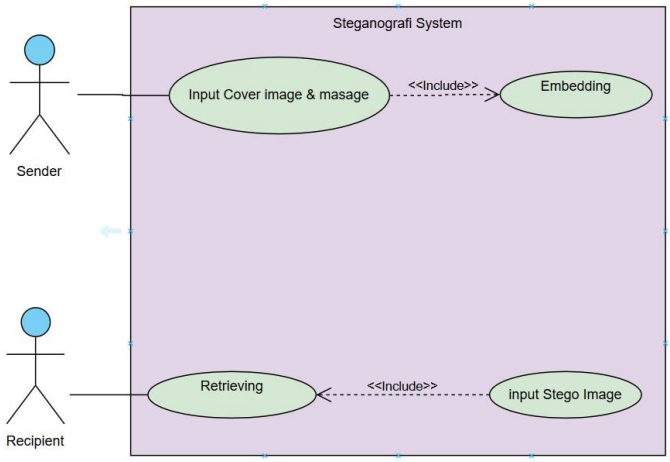
4. Menguraikan pesan tersembunyi pada gambar stego .
5. Membaca urutan bit-bit gambar stego untuk menentukan besar informasi pesan yang terselip. Apabila ukuran pesan memiliki nilai 0, hal tersebut menandakan jika input data tidak memuat sebuah pesan rahasia (bukan stego file). Apabila ukuran pesan disembunyikan ditemukan dalam gambar stego :
    a) Membuat indeks array sesuai dengan besar informasi pesan yang telah didapatkan.
    b) Gunakan operasi logika AND untuk memeriksa bit pada gambar stego yang dapat direpresentasikan dalam bentuk biner 00000011 ketika nilai heksadesimalnya adalah 0x03.
    c) Masukkan nilai ke dalam *indeks array* yang sudah ada dan sambungkan bit-bitnya sampai terbentuk 8 bit (1 *byte*). Kemudian, ubah menjadi tipe karakter.
    d) Gunakan operasi OR untuk memproses normalisasi bit pada gambar stego dengan hasil operasi AND sebelumnya.
6. Akhir dari proses tersebut menghasilkan pesan rahasia.
7. Selesai.

3.5. Perancangan model sistem

Bahasa pemrograman yang dipakai untuk mengembangkan sebuah aplikasi steganografi ini ialah MATLAB dan terdapat beberapa diagram, seperti diagram use case dan diagram urutan, yang akan dipergunakan untuk memperjelas rancangan sistem.

3.5.1. Perancangan Use Case Diagram

Gambaran sistem objek yang dihasilkan dari pendekatan metode USDP direpresentasikan dalam diagram use case yang berfungsi sebagai panduan untuk merancang model sistem. Dalam konteks aplikasi steganografi, gambar 3.8 adalah visualisasi dari diagram use case yang menunjukkan interaksi antara aktor dan sistem.



Gambar 3.8. *Use Case Diagram* Rancang Bangun Aplikasi Steganografi LSB

Pada gambar 3.8, terdapat rancangan aplikasi steganografi yang terdiri dari 4 use case dan 2 aktor. Urutan *use case* dimulai dari input gambar sampul dan pesan, langkah selanjutnya adalah melakukan embedding untuk menyisipkan pesan dan retrieving untuk mengambil kembali pesan. Sedangkan dua use case terakhir meliputi input gambar stego dan ekstraksi pesan (*Retrieving*). Dalam sistem ini terdapat dua aktor terlibat, termasuk dalam cakupan dari pengirim dan penerima pesan.

Berikut beberapa penjelasan mengenai *use case* secara detil pada tabel 3.1 sampai dengan tabel 3.4.

Tabel 3.1. Spesifikasi *Use Case Input Embedding Requirement Data*

Aktor utama	Pengirim (*Sender*)
Kondisi awal	Pengirim menginput *Stego Image*
Kondisi Akhir	Sistem validasi membutuhkan ukuran Stego Image minimal empat kali lebih besar daripada ukuran pesan tersebut.
Sekenario Normal	1. Dijalankan ketika aktor memutuskan untuk melakukan proses *embedding* pesan rahasia. 2. Dalam sistem tersebut, akan terdapat jenis pesan yang dapat disisipkan, yaitu berupa teks. 3. Persyaratan yang sama berlaku untuk jenis pesan, yaitu pesan yang akan di sisipkan (*message*), media akan memuat pesan tersebut (*cover image*), dan lokasi penyimpanan hasil output (*stego image*). 4. Sistem melakukan validasi ukuran antara pesan yang akan disisipkan dan media penampungnya. Syaratnya adalah ukuran media penampung harus minimal empat kali lebih besar dari ukuran pesan. 5. Apabila media penampung memenuhi syarat yang ditentukan, maka sistem dapat melanjutkan dengan proses *embedding*.
Skenario Alternatif	1. Apabila ukuran pesan lebih besar dari media penampung, pesan akan ditampilkan untuk mengindikasikan bahwa media penyimpanan yang lebih besar diperlukan untuk menampung pesan tersebut..

Tabel 3.2. Spesifikasi use case untuk memasukkan dan mengambil data persyaratan.

Aktor utama	Penerima (*Recipient*)
Kondisi awal	Penerima menginput *Stego Image* dari pengirim
Kondisi Akhir	Data yang dimasukkan adalah data *Stego Image*
Skenario Normal	1. Proses pengambilan dimulai saat aktor menunjuk opsi *retrieving*. 2. Sistem akan menunjukkan pesan yang dapat diambil Kembali. 3. Persyaratan dasar untuk pesan adalah identik. 4. Terdapat data berisi atau mencakup pesan (*stegofile*). 5. Sistem akan memvalidasi data yang dimasukkan. 6. Sistem akan melanjutkan proses *retrieving* apabila *stegofile* yang dimasukkan dianggap valid oleh sistem.
Skenario Alternatif	1. Apabila data yang dimasukkan bukan merupakan stegoimage, sebuah pesan *error* akan ditampilkan pada kotak pesan.

Tabel 3.3. Spesifikasi *Use Case Embedding*

Aktor utama	Pengirim (*Sender*)
Kondisi awal	Verifikasi ukuran data yang dimasukkan telah dilakukan.
Kondisi akhir	File hasil akhir adalah stegoimage yang membawa pesan yang telah disisipkan.
Skenario Normal	1. Proses embedding dimulai saat aktor memulai aksi pada sistem.

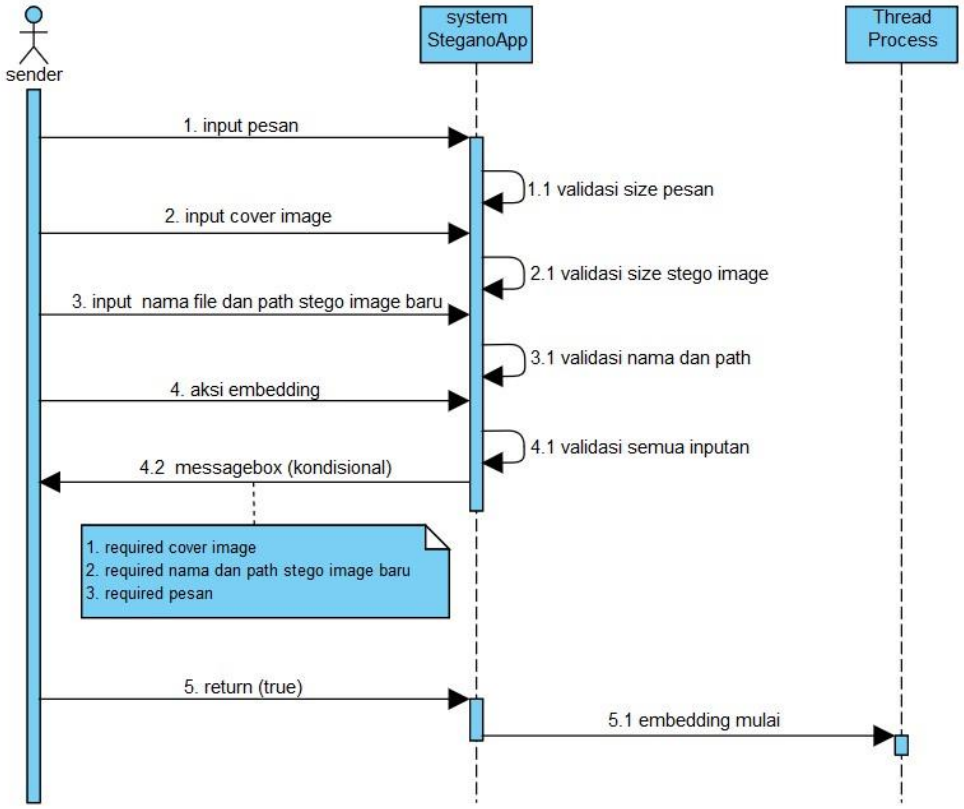
	2. Data input akan diubah menjadi aliran data oleh sistem. 3. Sistem membaca nilai OFFSET lalu mengabaikannya. 4. Sistem akan menempatkan bit-bit pesan pada media penampung dengan menggantikan satu bit LSB (Least Significant Bit) pada media penampung secara berurutan. Proses ini melibatkan beberapa operasi, termasuk operasi AND antara bit pesan dan nilai heksa 0x7F, kemudian operasi AND lagi dengan nilai heksa 0x03, diikuti dengan operasi AND pada bit media penampung dengan nilai heksa 0xFC. Setelah itu, hasil dari operasi AND tersebut akan digabungkan dengan operasi OR pada bit pesan. Proses ini akan diulang sampai seluruh bit pesan berhasil disisipkan ke dalam media penampung. 5. Apa bila berhasil, hasilnya berupa stegoimage.
Skenario Alternatif	1. Jika proses penggantian bit tidak berhasil, akan muncul peringatan pesan yang menyatakan bahwa proses embedding pesan tidak berhasil dilakukan.

Tabel 3.4. Spesifikasi *Use Case Retrieving*

Aktor awal	Penerima (*Recipient*)
Kondisi awal	Data dimasukkan sudah diverifikasi sesuai dengan syarat yang dibutuhkan.
Kondisi akhir	Pesan telah diterima oleh penerima.
Skenario Normal	1. Proses dimulai saat aktor melakukan tindakan pengambilan data dari sistem. 2. Sistem akan mengubah data input menjadi bentuk aliran data. 3. Lalu mengecek angka OFFSET dan tidak memperhatikannya. 4. Berikutnya, sistem mengetahui informasi tentang besar pesan yang tersembunyi. 5. Membuat array dengan ukuran yang sesuai dengan data panjang pesan yang didapatkan. 6. Melakukan operasi AND pada *byte stegoimage* bernilai heksadesimal 0x03, menggabungkan bit hasil operasi menjadi satu *byte* (8 bit) dengan jumlah genap, dan mengonversinya menjadi karakter. Selanjutnya, karakter tersebut akan diindekskan ke dalam indeks *array* yang telah ditentukan sebelumnya. Proses ini dilakukan secara berulang hingga seluruh karakter pesan tersembunyi berhasil diambil. 7. Hasil akhir yaitu, berhasil diperoleh pesan rahasia dari stegoimage yang telah diambil.
Skenario Alternatif	-

3.5.2. Perancangan *Sequence Diagram*

Dalam konteks sistem, *Sequence Diagram* urutan merupakan suatu ilustrasi yang menjelaskan bagaimana pesan-pesan dikirim dan diterima pada urutan waktu tertentu. Melalui diagram urutan, kita dapat melihat bagaimana objek-objek dalam sistem saling berinteraksi dan memainkan perannya masing-masing (Booch et al, 2005). Gambar 3.9 menunjukkan diagram urutan untuk sistem yang sedang dalam proses pengembangan.

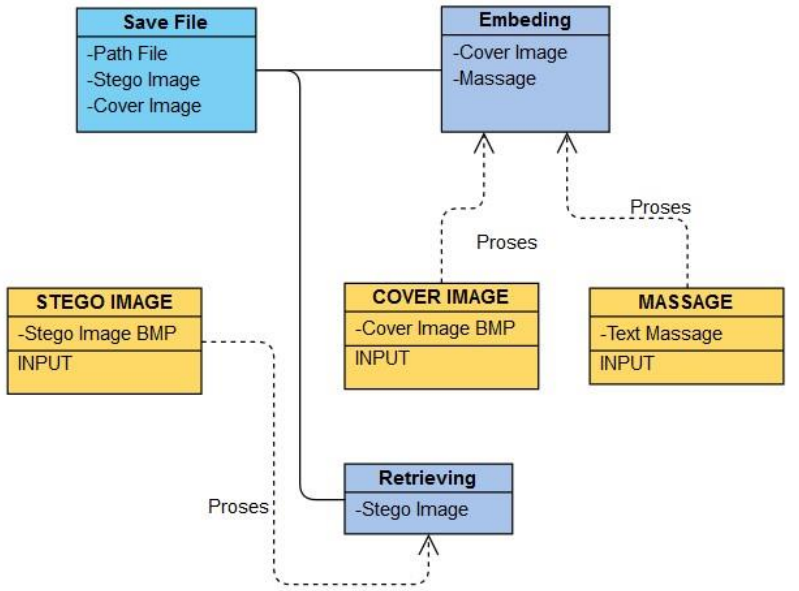


Gambar 3.9 *Sequence Diagram*

3.6. Class Diagram

Class diagram ialah gambaran visual yang dimanfaatkan untuk mengilustrasikan suatu sistem dengan menjelaskan relasi antara kelas satu dengan kelas yang lain. pada sistem tersebut. Pada pembangunan aplikasi steganografi, terdapat class diagram yang digunakan dan dapat ditemukan pada gambar 3.10.

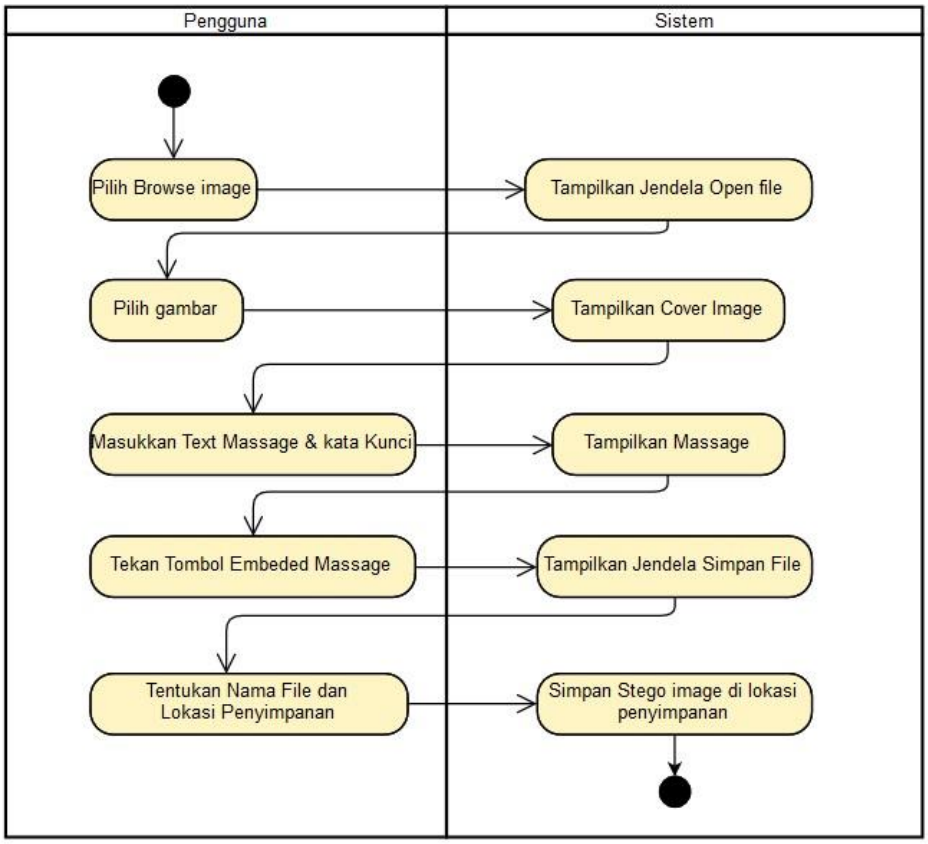
.



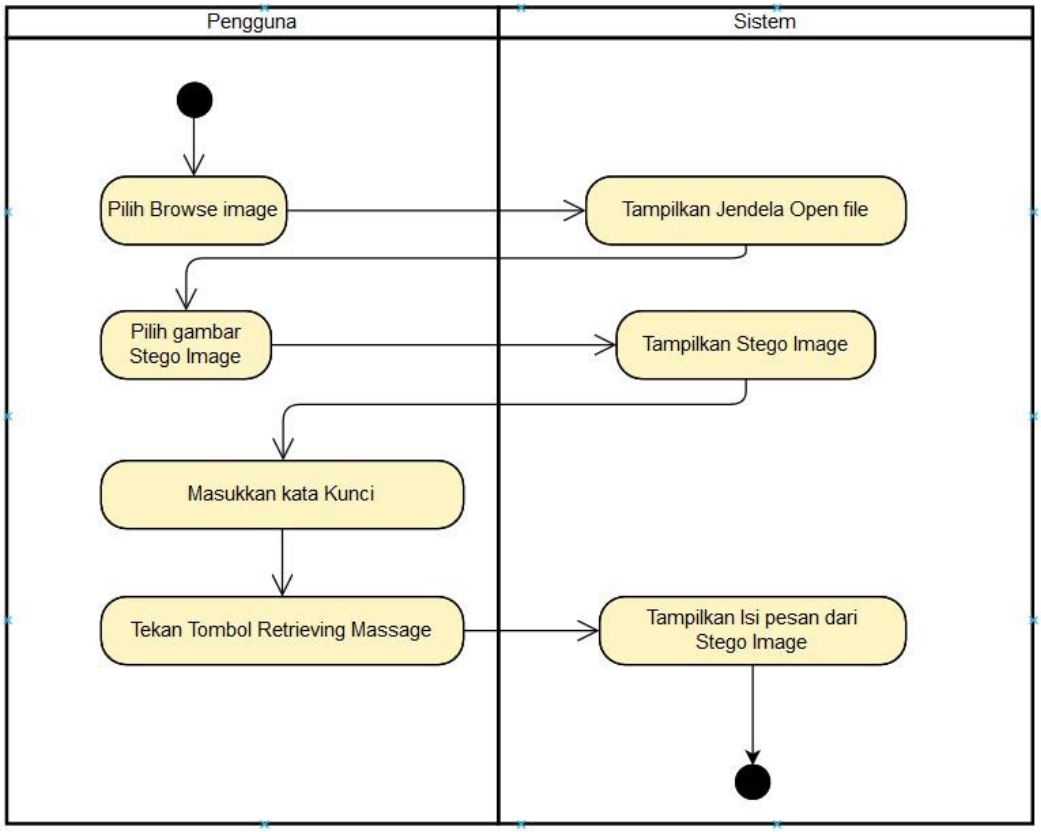
Gambar 3.10. *Class* Diagram

3.7. Activity Diagram

Diagram aktivitas adalah suatu bentuk diagram yang menunjukkan bagaimana kontrol aliran aktivitas dalam sistem bergerak dari satu aktivitas ke aktivitas lainnya. Dalam dasarnya, diagram aktivitas mirip dengan flowchart (Booch et al, 2005). Gambar 3.11 dan 3.12 menunjukkan diagram aktivitas dari sistem yang sedang dibangun.



Gambar 3.11. Activity diagram proses *Embedding Massage*



Gambar 3.12. Activity Diagram proses *Retrieving Massage*

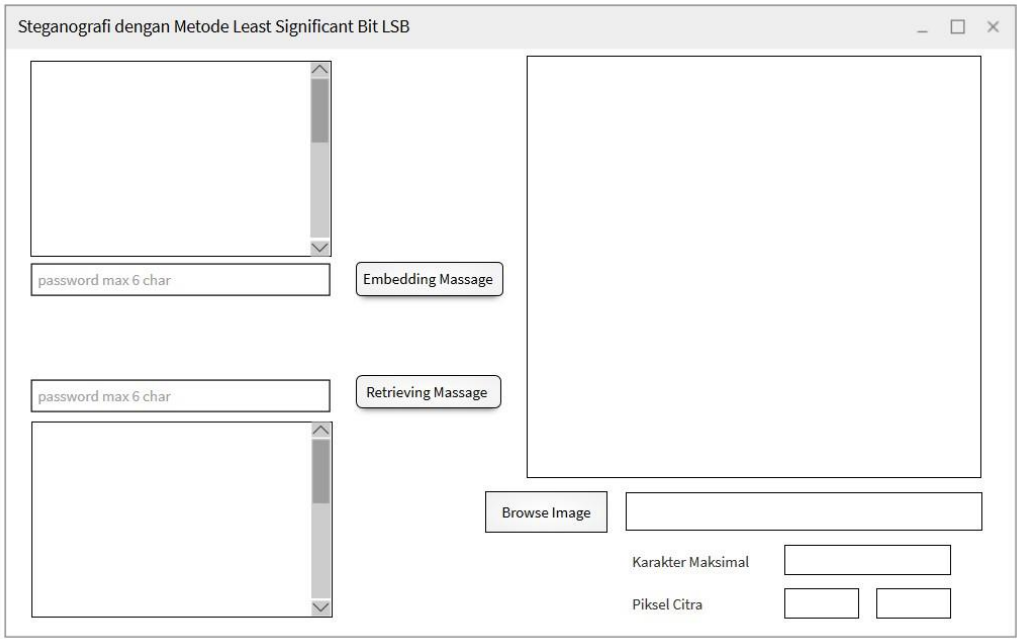
3.8. Design antarmuka

Merencanakan suatu sistem melibatkan proses perancangan antarmuka agar dapat menampilkan gambar sampul yang akan diproses oleh sistem saat sedang dibuat. Dapat ditemukan desain antarmuka tersebut pada gambar 3.13 di bawah ini :

Pengujian proses bertujuan untuk menyisipkan pesan ke dalam objek gambar dan beberapa fungsi yang terlibat dalam proses pengujian tersebut, Berikut contoh fungsi :

1. Browse Image : Untuk memilih gambar (*cover image*) mana yang akan disisipi pesan
2. *Embedding* Massage : Untuk proses penyisipan pesan pada wadah penampung (*cover image)*

3. Retrieving Massage *:* Untuk menampilkan pesan yang telah di sisipkan sebuah pesan pada wadah penampung gambar (*StegoImage*)



Gambar 3.13. Desain Antar muka Rancang Bangun aplikasi Steganografi dengan LSB

Alur Progam : Setelah tombol browse image dijalankan maka akan menampilkan cover image dan path letak file *cover image*. Kemudian pada form kotak di isi dengan pesan yang akan disisipkan. Kemudian mengisi password dengan maksimal 6 char dan tekan tombol *embedding massage* untuk melanjutkan proses penyisipan pesan pada wadah penampung (cover image), lalu simpan hasil *stegoimage* sesuai keinginan.