BAB III
METODOLOGI PENELITIAN

3.1 Identifikasi Masalah

Penelitian ini menggabungkan teknik Kriptografi ElGamal dan Steganografi LSB diacak dengan PRNG BBS. Implementasi ini dilakukan dengan mengenkripsi pesan dalam bentuk teks menggunakan algoritma ElGamal hingga diperoleh teks hasil enkripsi (*cipherteks*), yang kemudian disisipkan ke dalam *cover-image* berupa gambar untuk menghasilkan gambar yang sudah disisipkan pesan (*stego-image*). Selanjutnya, proses ekstraksi dilakukan untuk mendapatkan kembali cipherteks dari *stego-image*, yang kemudian didekripsi menggunakan algoritma ElGamal untuk menghasilkan plainteks berupa teks.

Program yang dibangun terdiri dari lima fungsi utama: pembangkitan kunci ElGamal, enkripsi menggunakan algoritma ElGamal, dekripsi menggunakan algoritma ElGamal, *embedding* menggunakan *Least Significant Bit*, dan ekstraksi menggunakan *Least Significant Bit*. Proses pembangkitan kunci ElGamal dilakukan oleh penerima pesan dengan hasil berupa kunci publik (*y, g, p*) dan kunci privat (*x, p*). Selanjutnya, penerima pesan mengirimkan kunci publik (*y, g, p*) kepada pengirim. Pada proses enkripsi menggunakan algoritma ElGamal, diperlukan *input* berupa bilangan acak *k* dan kunci publik (*y, g, p*). Proses ini menghasilkan output berupa cipherteks.

Untuk proses steganografi, diperlukan input berupa *file* gambar, cipherteks hasil enkripsi kriptografi ElGamal, dan kunci BBS (*p, q,* dan *seed*). Proses *embedding* ini menghasilkan output berupa *stego-image* yang telah disisipkan pesan. Pada proses ekstraksi, diperlukan input berupa *stego-image* yang telah disisipkan pesan dan kunci BBS. Proses ini menghasilkan *output* berupa cipherteks yang telah disisipkan pada *stego-image*. Kemudian, pada proses dekripsi kriptografi ElGamal, diperlukan input berupa kunci privat (*x, p*) ElGamal dan cipherteks yang telah disisipkan pada *stego-image*. Proses ini menghasilkan *output* berupa plainteks.

Dengan penggabungan teknik Kriptografi ElGamal dan Steganografi LSB diacak dengan PRNG BBS, pesan rahasia dapat disembunyikan dengan aman dalam *cover*

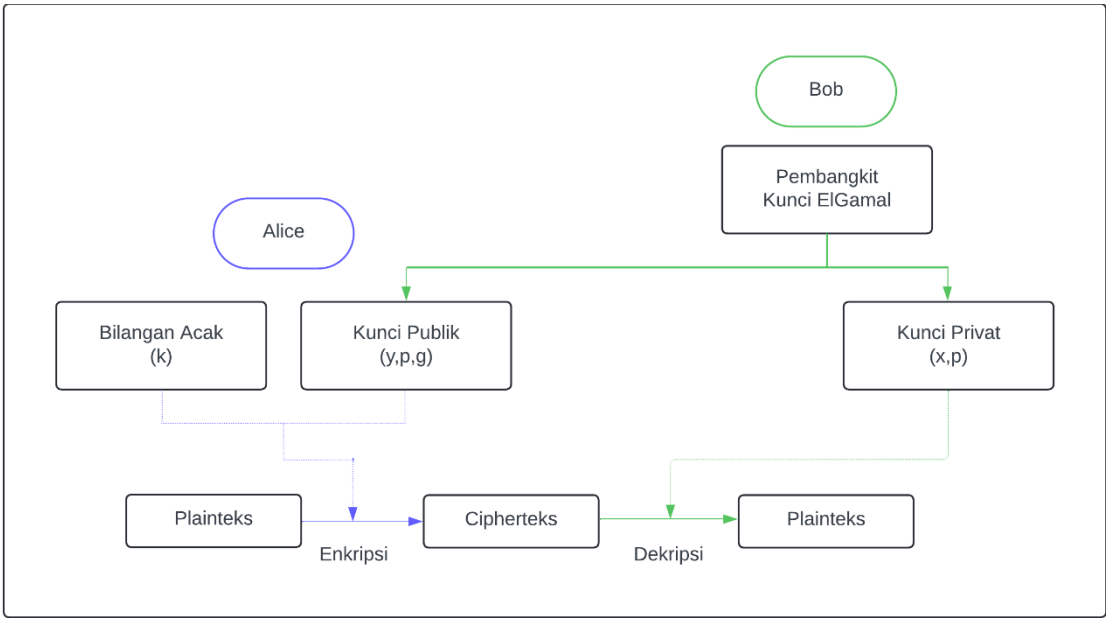
image dan dikirim sebagai *stego-image*. Proses ekstraksi dan dekripsi memungkinkan penerima untuk mendapatkan kembali pesan asli dengan menggunakan kunci yang sesuai.

3.2 Model Dasar

Model dasar yang digunakan dalam penelitian ini adalah algoritma kriptografi ElGamal dan steganografi LSB BBS.

3.2.1 Algoritma kriptografi ElGamal

Skema Algoritma kriptografi ElGamal berdasarkan yang dipaparkan dalam BAB II bagian 2.2.9 akan ditunjukkan dalam Gambar 3.1



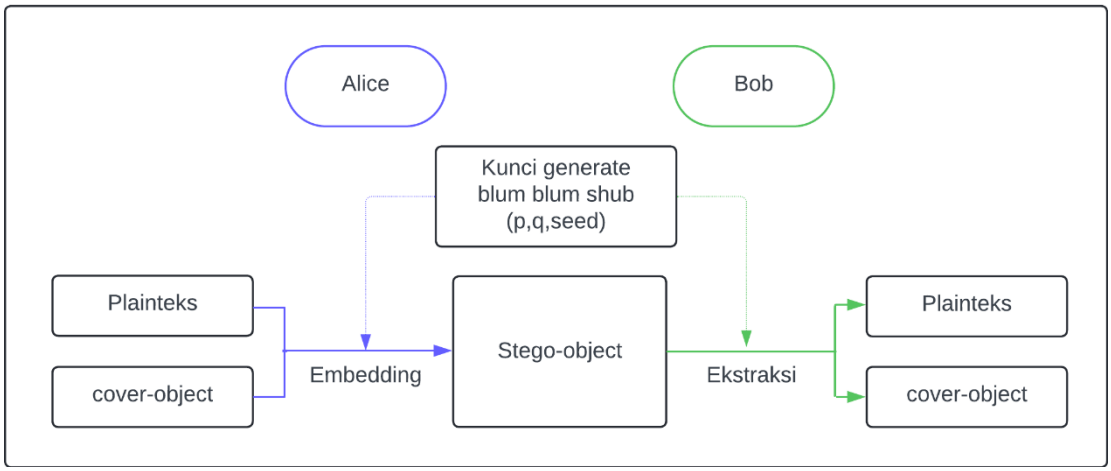
Gambar 3.1 Skema Algoritma Kriptografi ElGamal

Pada skema dalam Gambar 3.1 di atas memperlihatkan bahwa sebelum melakukan proses enkripsi dan dekripsi, harus dibangkitkan dulu suatu pasangan kunci publik dan kunci privat. Bob membangkitkan kunci ElGamal sehingga diperoleh pasangan kunci publik (*y*,*p*,*g*) dan kunci privat (*x*,*p*). Langkah selanjutnya, Bob mengirimkan kunci publik ke Alice. Saat melakukan enkripsi, selain menggunakan

kunci publik yang diberikan Bob, Alice juga menggunakan suatu bilangan acak *k* untuk mengenkripsi plainteks yang hendak dikirimkan.Cipherteks yang diperoleh dari proses enkripsi plainteks menggunakan kunci publik ElGamal dan bilangan acak *k* dikirimkan ke Bob, lalu Bob dapat mendekripsi cipherteks tersebut dengan kunci privat yang dimiliki.

3.2.2 Steganografi LSB BBS

Skema Steganografi LSB BBS berdasarkan yang dipaparkan dalam BAB II bagian 2.4 akan ditunjukkan dalam Gambar 3.2



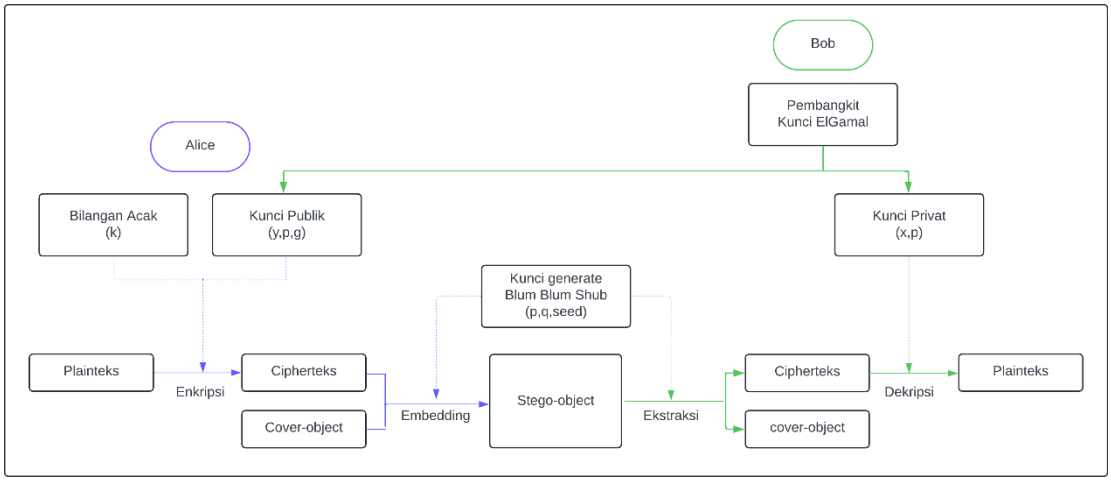
Gambar 3.2 Skema Algoritma LSB BBS

Pada Steganografi LSB BBS, Alice menentukan kunci BBS dan plainteks yang akan disembunyikan dalam *cover-object*. Langkah selanjutnya, Alice melakukan proses *embedding* menggunakan kunci BBS sehingga menghasilkan *stego object*. Alice mengirimkan *stego-object* dan kunci BBS kepada Bob. Lalu bob menggunakan kunci BBS tersebut untuk proses ekstraksi *stego-object* sehingga diperoleh plainteks.

3.3 Pengembangan Model Dasar

Pengembangan model pada penelitian ini dengan menggabungkan algoritma pada model dasar yaitu algoritma kriptografi ElGamal dan steganografi LSB BBS. Cara kerja penggabungan kriptografi ElGamal dan steganografi LSB BBS dapat ditunjukkan

pada skema pengembangan model seperti pada Gambar 3.3



Gambar 3.3 Skema Pengembangan Model Dasar

Dalam implementasi pengembangan modelnya, Bob membangkitkan kunci dengan algoritma ElGamal, hingga diperolah kunci publik dan kunci privat. Kemudian, kunci publik tersebut dikirimkan kepada Alice. Alice mengenkripsi pesan yang akan dikirim menggunakan kunci publik dan bilangan acak *k* hingga di peroleh cipherteks. kemudian Alice menentukan kunci BBS dan *cover-object* untuk digunakan pada proses *embedding*. Saat melakukan proses *embedding* Alice menggunakan cipherteks dan kunci BBS sehingga menghasilkan *stego-object*. Alice mengirim kunci BBS ke Bob. Lalu bob menggunakan kunci BBS tersebut untuk proses ekstraksi sehingga diperoleh cipherteks. Kemudian Bob menggunakan kunci privat untuk proses dekripsi sehingga diperoleh plainteks.

3.4 Konstruksi Program Aplikasi

Pembuatan program aplikasi dari hasil penggabungan algoritma kriptografi ElGamal dan steganografi LSB BBS ini akan dikonstruksi menggunakan *Graphical User Interface* (GUI) dalam bahasa pemrograman *Phyton*. Program ini akan memiliki beberapa pilihan tab seperti pembangkitan kunci ElGamal, enkripsi menggunakan algoritma ElGamal, dekripsi menggunakan algoritma ElGamal, *embedding*

menggunakan *Least Significant Bit*, dan ekstraksi menggunakan *Least Significant Bit*.

3.4.1 *Input* dan *Output*

Input dari pembangkitan kunci ElGamal adalah batas maksimal karakter. *Output* dari pembangkitan kunci adalah kunci publik (*y, p, g*) dan kunci privat (*x,p*). Input dari enkripsi pesan menggunakan algoritma ElGamal adalah plainteks, kunci publik (*y, g, p*), dan bilangan acak *k*. *Output* dari enkripsi pesan adalah cipherteks. *Input* dari proses embedding menggunakan steganografi LSB diacak dengan PRNG BBS adalah cipherteks, *cover image*, dan kunci BBS (*p*, *q*, dan *seed*). *Output* dari proses *embedding* adalah *stego-image*. *Input* dari proses ekstraksi adalah *stego-image* dan kunci BBS. *Output* dari proses ekstraksi adalah cipherteks. *Input* dari dekripsi pesan menggunakan algoritma ElGamal adalah cipherteks dan kunci privat (*x,p*). *Output* dari dekripsi pesan adalah plainteks.

3.4.2 Algoritma Deskriptif

Berikut adalah algoritma deskriptif program aplikasi yang akan dikonstruksi:

1. Bob melakukan proses pembangkitan kunci ElGamal dengan menggunakan algoritma pembangkitan kunci ElGamal sehingga diperoleh pasangan kunci publik (*y,p,g*) dan kunci privat (*x,p*).
2. Bob mengirimkan kunci publik (*y,p,g*) kepada Alice.
3. Alice menyiapkan teks yang akan diamankan dan gambar yang akan digunakan sebagai *cover object*.
4. Alice melakukan proses enkripsi pesan dengan menggunakan kunci publik ElGamal yang telah dibangkitkan hingga diperoleh cipherteks.
5. Alice melakukan proses pembangkitan kunci *Blum Blum Shub* (BBS) dengan menggunakan bilangan prima besar *p* dan *q* serta *seed*.
6. Alice melakukan proses *embedding* cipherteks ke dalam gambar yang telah

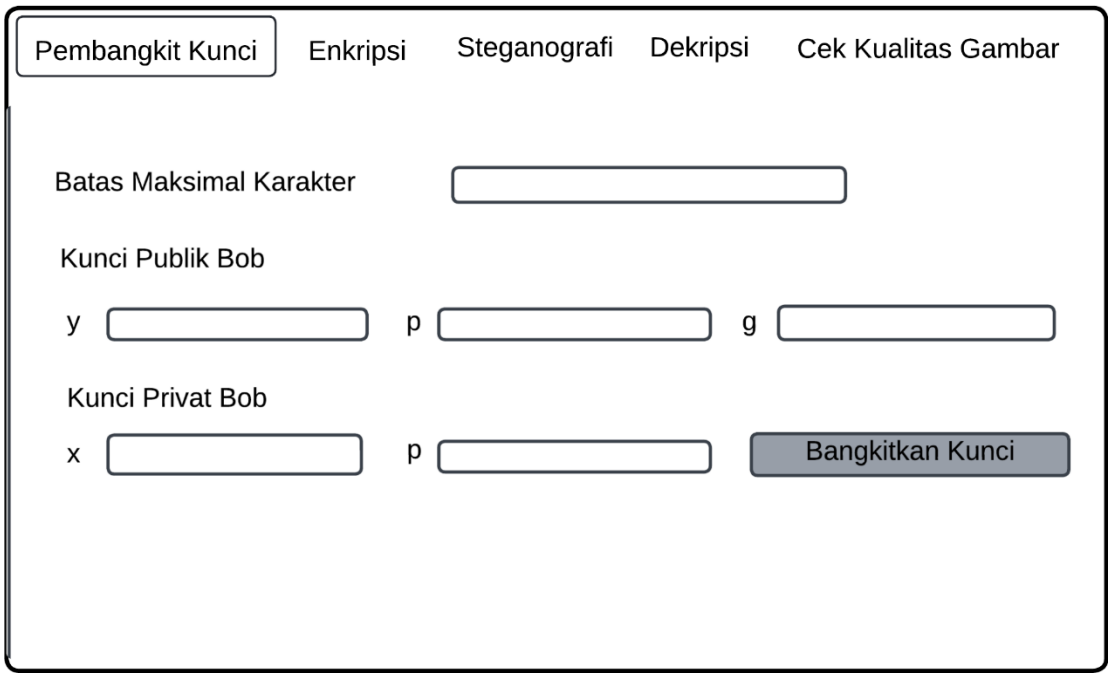
disiapkan dengan menggunakan kunci BBS hingga diperoleh hasil akhir yaitu *stego-image*.

7. Alice mengirimkan kunci BBS dan *stego-image* kepada Alice
8. Bob yang telah mendapatkan *stego-image* melakukan proses ekstraksi gambar menjadi cipherteks dengan menggunakan kunci BBS yang diterima dari Alice.
9. Bob melakukan proses dekripsi cipherteks dengan menggunakan kunci privat ElGamal yang telah dibangkitkan hingga diperoleh plainteks.

3.4.3 Rancangan Tampilan Program Aplikasi

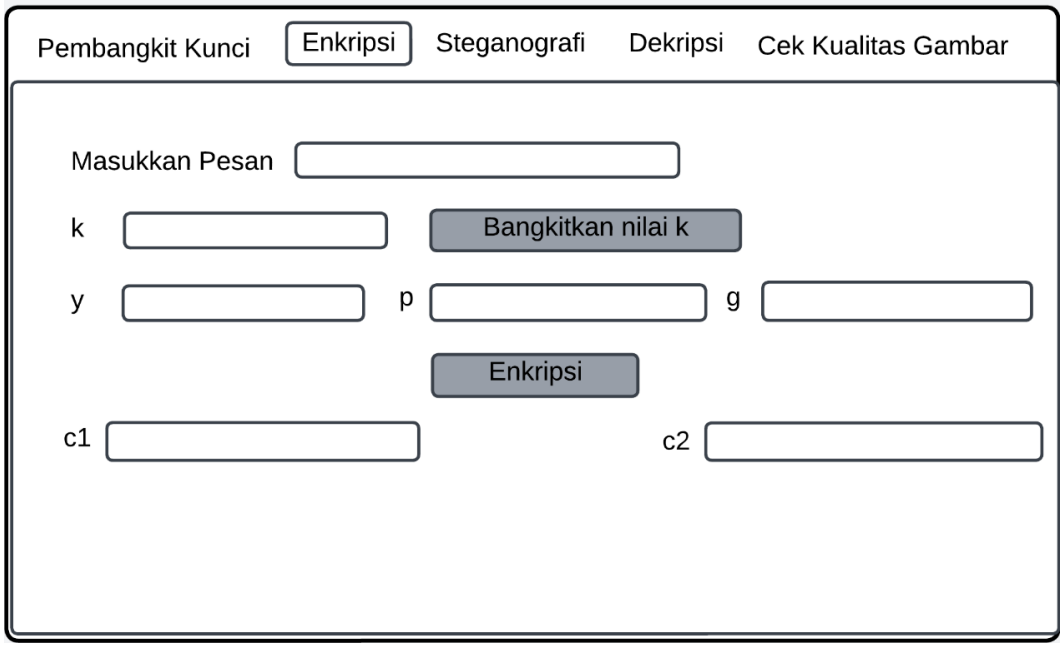
Rancangan tampilan program aplikasinya akan memiliki 4 tab, yaitu pembangkitan kunci, enkripsi, dekripsi, dan Steganografi. Rancangan tampilan program aplikasi yang akan dibuat ada pada Gambar 3.4, Gambar 3.5, Gambar 3.6, Gambar 3.7, dan Gambar 3.8.

a. Pembangkit Kunci



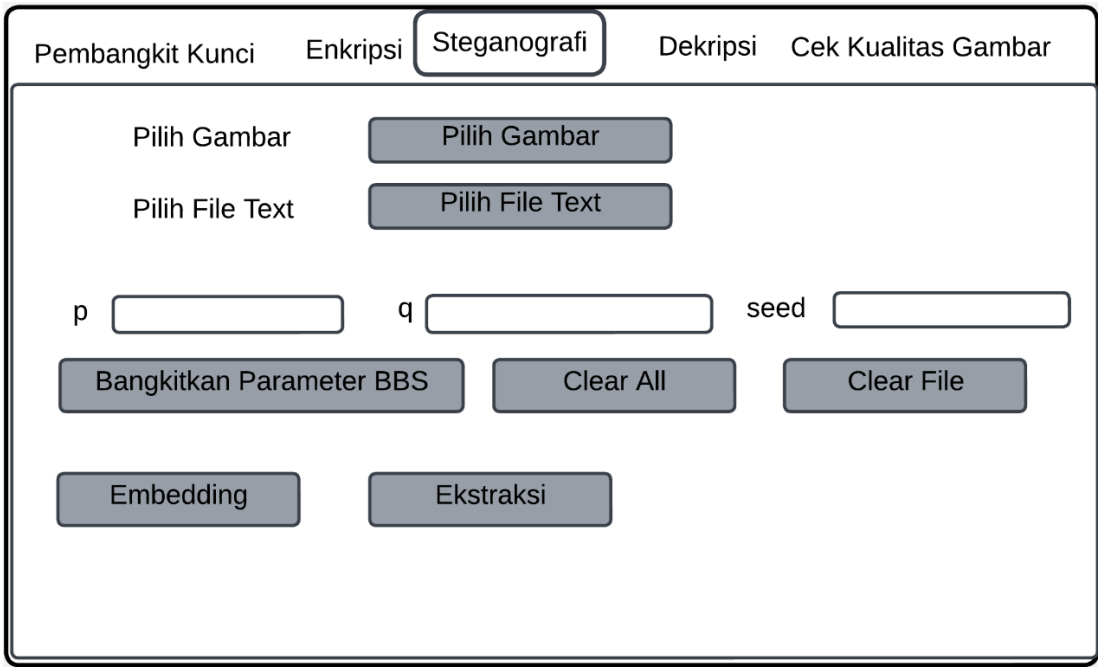
Gambar 3.4 Rancangan Tampilan Pembangkit Kunci

b. Enkripsi



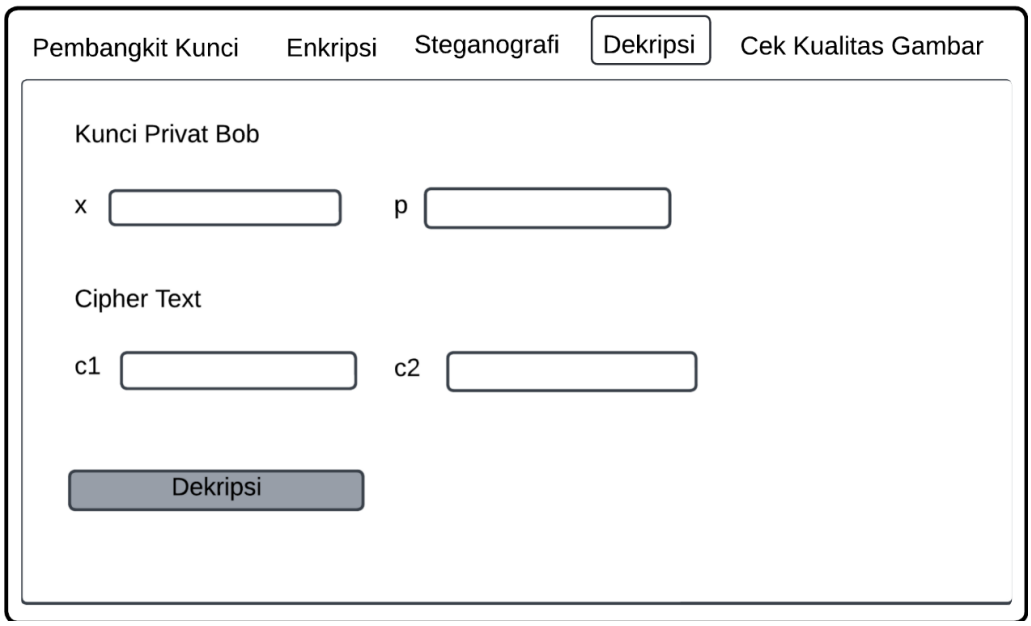
Gambar 3.5 Rancangan Tampilan Enkripsi ElGamal

c. Steganografi



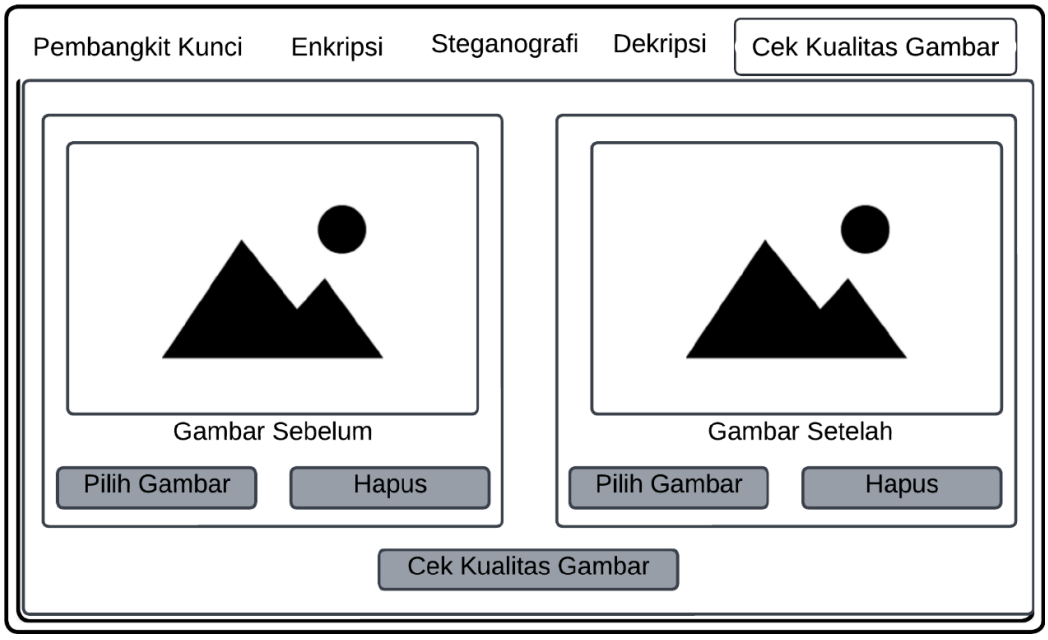
Gambar 3.6 Rancangan Tampilan Steganorafi

d. Dekripsi



Gambar 3.7 Rancangan Tampilan Dekripsi ElGamal

e. Cek Kualitas Gambar



Gambar 3.8 Rancangan Tampilan Cek Kualitas Gambar

3.4.4 *Library Python*

Dalam penggunaannya, *library python* yang akan digunakan untuk menunjang pembuatan aplikasi adalah sebagai berikut:

1. *Tkinter*

   Tkinter adalah *library* yang digunakan untuk membuat antarmuka grafis pada aplikasi *Python*. Dengan *Tkinter*, para pengembang dapat dengan mudah membuat jendela, tombol, kotak teks, dan elemen-elemen lainnya untuk berinteraksi dengan pengguna.

2. *Filedialog*

   Filedialog adalah *library* yang menyediakan dialog untuk memilih *file* dari sistem *file*. *Filedialog* digunakan untuk memberikan kemampuan bagi pengguna untuk memilih gambar dari sistem mereka yang akan digunakan dalam proses steganografi.

3. *Messagebox*

   Messagebox adalah *library* yang digunakan untuk menampilkan kotak dialog pesan kepada pengguna. *Messagebox* digunakan untuk memberikan informasi atau pesan konfirmasi kepada pengguna terkait hasil proses steganografi atau jika terjadi kesalahan dalam proses tersebut. Dengan *Messagebox*, aplikasi dapat memberikan umpan balik yang lebih informatif kepada pengguna tentang status dan hasil dari operasi steganografi.

4. PIL (*Python Imaging Library*)

   PIL adalah *library* yang menyediakan berbagai fungsi untuk manipulasi gambar dalam *Python*. PIL digunakan untuk membuka gambar yang dipilih oleh pengguna dan untuk menyimpan gambar yang telah diubah setelah proses steganografi. PIL memungkinkan aplikasi untuk melakukan operasi seperti

membuka, menyimpan, mengubah ukuran, dan mengubah format gambar.

5. *OS*

   OS adalah *library* yang menyediakan berbagai fungsi untuk berinteraksi dengan sistem operasi. Dalam kode tersebut, *OS* digunakan untuk mengatur atau memanipulasi *path file* gambar yang digunakan dalam proses steganografi. Dengan menggunakan *OS*, aplikasi dapat dengan mudah berinteraksi dengan sistem *file*.

6. *Sympy*

   Sympy adalah *library* yang menyediakan alat-alat untuk matematika simbolik. Ini digunakan dalam kode untuk menghasilkan bilangan prima secara acak dan melakukan operasi matematika terkait kriptografi.

7. *Random*

   Random adalah *library* yang menyediakan fungsi-fungsi untuk menghasilkan angka acak. Dalam penelitian ini, *random* digunakan untuk menghasilkan angka acak untuk *seed* pada algoritma *Blum Blum Shub*.

8. *Pyperclip*

   Pyperclip adalah *library* yang menyediakan fungsi untuk mengakses *clipboard* komputer. Dalam penelitian ini, *pyperclip* digunakan untuk menyalin pesan terdekripsi ke clipboard agar pengguna dapat dengan mudah mengaksesnya.

9. *Crypto*

   Crypto adalah *library* yang menyediakan alat-alat untuk berbagai operasi kriptografi, termasuk pembangkitan bilangan prima, enkripsi, dan dekripsi.

10. *Binascii*

    Binascii adalah *library* yang digunakan untuk mengkonversi antara representasi *biner* dan ASCII.

11. *Hexlify* dan *Unhexlify*

    Hexlify adalah fungsi yang digunakan untuk mengkonversi data biner menjadi representasi heksadesimal. Ini berguna untuk menampilkan atau menyimpan data biner dalam format yang lebih mudah dibaca dan ditransmisikan. *Unhexlify*, sebaliknya, mengkonversi data heksadesimal kembali menjadi representasi *biner*. Fungsi ini penting dalam proses enkripsi dan dekripsi di mana data sering kali perlu dikonversi antara berbagai format.

12. *Pandas*

    Pandas adalah *library* yang menyediakan alat-alat untuk analisis data.

3.5 Proses Validasi

Proses validasi dilakukan untuk memastikan program aplikasi yang dirancang berjalan dengan tepat. Proses validasi dilakukan dengan cara melakukan percobaan pada program aplikasi dengan menggunakan beberapa contoh. Program aplikasi akan tervalidasi jika pada seluruh percobaan, cipherteks yang disisipkan pada *stego-image* dapat dikembalikan menjadi plainteks. Setelah proses enkripsi dan penyisipan pesan menghasilkan *stego-image*, validasi dilakukan untuk memeriksa kualitas gambar hasil steganografi. Validasi ini menggunakan PSNR sebagai metrik evaluasi. Nilai PSNR mengukur perbedaan antara *cover-image* dan *stego-image*. PSNR yang tinggi menunjukkan kualitas gambar yang baik setelah penyisipan pesan. Validasi ini dilakukan menggunakan contoh yang diambil dari *output* program aplikasi yang telah dikonstruk untuk memastikan kualitas *stego-image* yang dihasilkan.

3.6 Pengambilan Kesimpulan

Pada tahap ini akan ditarik kesimpulan dari pengembangan model yang dilakukan melalui hasil dari proses validasi, yaitu terkait penggabungan Kriptografi ElGamal dan Steganografi LSB diacak dengan PRNG BBS beserta implementasinya ke dalam program aplikasi dengan menggunakan GUI *Python*