**PROJECT IMPLEMENTASI STEGANOGRAFI**

****

**Disusun Oleh:**

Nama : I Dewa Gde Putra Anga Biara

NIM : 2008561105

**Dosen Pengampu :**

I Komang Ari Mogi, S.Kom., M.Kom.

**PROGRAM STUDI INFORMATIKA**

**FAKULTAS MATEMATIKA DAN ILMU PENGETAHUAN ALAM**

**UNIVERSITAS UDAYANA**

**2023**

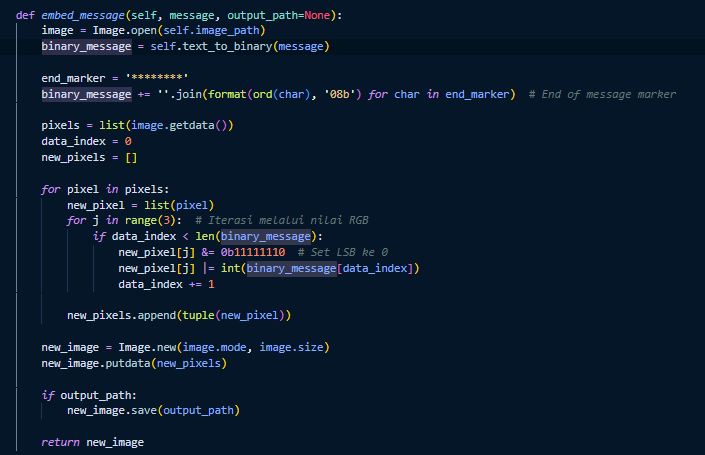
**Penjelasan Program**

Dalam proyek ini, program yang dikembangkan menggunakan metode Least Significant Bit (LSB). LSB didefinisikan sebagai ide untuk menyembunyikan bit terakhir dari setiap bit atau sampel. Perubahan dilakukan pada bit paling kanan karena bit tersebut memiliki nilai terendah dan tidak secara signifikan memengaruhi nilai RGB gambar secara keseluruhan. Implementasi program ini dibuat menggunakan bahasa pemrograman Python dengan menggunakan library Tkinter untuk membuat antarmuka grafis (GUI) dan library Pillow untuk memproses gambar.

Dalam struktur program, implementasi LSB terletak di dalam kelas Program, yang sudah memiliki dua fungsi, yaitu embed\_message dan extract\_message. Fungsi embed\_message digunakan untuk menyisipkan pesan ke dalam gambar yang dipilih, sementara fungsi extract\_message bertujuan untuk mengekstrak dan menampilkan pesan yang telah disisipkan ke dalam gambar.

**Fungsi Penyisipan Pesan Kedalam Gambar**

Didalam implementasi yang dibuat, fungsi yang digunakan untuk menyisipkan pesan kedalam gambar adalah fungsi *embed\_message* yang terdapat dalam *class* Program. Potongan *code* dapat dilihat pada gambar dibawah ini.



*Gambar 1. Fungsi Menyisipkan Pesan*

Pada fungsi *embed\_message* dilakukan dengan beberapa langkah yaitu:

**Fungsi Menampilkan Pesan Yang Disispkan**

1. Pilih gambar yang akan digunakan untuk menyisipkan pesan. Pada langkah ini, gunakan modul PIL dengan memanfaatkan fungsi Image.open(self.image\_path).
2. Konversikan pesan yang akan disisipkan menjadi bentuk biner dengan menggunakan fungsi text\_to\_binary yang telah disertakan dalam program. Setelah konversi, representasi biner dari setiap karakter akan disimpan dalam variabel binary\_message.
3. Tentukan akhir dari pesan yang akan disisipkan dan simpan representasi biner akhir tersebut dalam variabel binary\_message.
4. Representasikan nilai RGB dari setiap piksel menggunakan perulangan for pada setiap pixel.
5. Ubah setiap nilai RGB piksel dengan mengganti bit paling rendahnya (LSB) menggunakan bit pesan yang akan disisipkan. Proses ini dilakukan melalui operasi bitwise (AND, OR) pada new\_pixel[j].
6. Gunakan indeks data\_index untuk melacak posisi saat ini dalam binary\_message. Setiap kali satu bit pesan berhasil disisipkan, tingkatkan nilai data\_index untuk beralih ke bit berikutnya.
7. Piksel yang telah dimodifikasi kemudian ditambahkan ke dalam daftar new\_pixels.
8. Langkah terakhir adalah memilih lokasi untuk menyimpan gambar baru yang telah disisipkan pesan ke dalam folder lokal.

Didalam implementasi yang dibuat, fungsi yang digunakan untuk menyisipkan pesan kedalam gambar adalah fungsi *extract\_message* yang terdapat dalam *class* Program. Potongan *code* dapat dilihat pada gambar dibawah ini.



*Gambar 2. Fungsi Menampilkan Pesan Yang Disisipkan*

Pada fungsi *extract\_message* dilakukan dengan beberapa langkah yaitu:

1. Buka gambar yang mengandung pesan tersembunyi menggunakan modul PIL dengan memanfaatkan fungsi Image.open(self.image\_path).
2. Ambil nilai piksel dari gambar dan simpan dalam bentuk daftar menggunakan pixels = list(image.getdata()).
3. Ekstraksi Bit Pesan: Dalam loop pertama (for pixel in pixels), iterasikan setiap nilai RGB dari setiap piksel. Kemudian, ambil bit paling rendah (Least Significant Bit, LSB) menggunakan operasi bitwise value & 1. Bit tersebut ditambahkan ke dalam variabel binary\_message sebagai representasi biner dari pesan tersembunyi.
4. Penanda Akhir Pesan: Tentukan penanda akhir pesan ('\*\*\*\*\*\*\*\*'). Buat representasi biner dari penanda ini dengan menggunakan metode format(ord(char), '08b') dan simpan dalam variabel end\_marker\_binary.
5. Cari akhir pesan menggunakan binary\_message.find(end\_marker\_binary). Lakukan pencarian dalam binary\_message untuk menemukan indeks dari penanda akhir pesan. Jika ditemukan, simpan indeks tersebut dalam end\_marker\_index.
6. Pengecekan Akhir Pesan: Jika end\_marker\_index tidak sama dengan -1, artinya penanda akhir pesan ditemukan dalam binary\_message. Selanjutnya, lakukan langkah-langkah berikut:
   1. Hapus Penanda Akhir Pesan: Bagian pesan setelah penanda akhir dihapus dari binary\_message menggunakan slicing (binary\_message[:end\_marker\_index]).
   2. Konversi Biner ke Teks: Ubah representasi biner dari pesan yang tersisa menjadi teks dengan menggunakan metode binary\_to\_text. Simpan hasilnya dalam variabel text\_message.
   3. Simpan Gambar Hasil (Opsional): Jika output\_path tidak None, gambar yang mengandung pesan tersembunyi dapat disimpan sebagai gambar baru. Setiap nilai biner dikembalikan ke nilai piksel, dan gambar baru disimpan menggunakan new\_image.save(output\_path).
   4. Kembalikan Pesan: Fungsi mengembalikan teks pesan yang berhasil diekstrak dari gambar.
7. Ada kondisi di mana gambar tidak memiliki pesan, dan dalam situasi ini, fungsi akan menampilkan pesan "Tidak ada pesan di gambar ini". Pengecekan dilakukan dengan memeriksa penanda akhir pesan; jika tidak ditemukan, gambar dianggap tidak mengandung pesan.