Jurnal Teknik Informatika (JUTIF)
Vol. 4, No. 2, April 2023, hlm. 333-344

DOI: https://doi.org/10.52436/1.jutif.2023.4.2.340
p-ISSN: 2723-3863
e-ISSN: 2723-3871

IMPLEMENTATION OF STEGANOGRAPHY ON DIGITAL IMAGE WITH MODIFIED VIGENERE CIPHER ALGORITHM AND LEAST SIGNIFICANT BIT (LSB) METHOD

Gilang Miftakhul Fahmi[*1], Khairunnisak Nur Isnaini[2], Didit Suhartono[3]

[1,2,3]Informatika, Fakultas Ilmu Komputer, Universitas Amikom Purwokerto, Indonesia
Email: [1]gilangfahmi20@gmail.com, [2]nisak@amikompurwokerto.ac.id , [3]didit@amikompurwokerto.ac.id

(Naskah masuk: 24 Mei 2022, Revisi: 7 Juli 2022, Diterbitkan: 23 Maret 2023)

Abstract

Digital image can be utilized as a media to hide messages. There should be a special treatment in securing the messages so that the information accessibility can be controlled. Hiding messages to an image can be conducted using steganography. It can be combined with cryptography algorithms in order to make it harder to be accessed by unauthorized parties. The modification of cryptography algorithms can improve the confidentiality of the message content. This research is aimed to know the implementation of steganography in desktop applications using Least Significant Bit (LSB) that is combined with the modification of Vigenere Cipher algorithms. The developed software namely waterfall supports the creation of applications. The result of the research shows that the encrypted picture will change into gray or display in grayscale so that the hidden message will be more difficult to be accessed by the attacker. The encrypted picture will also have a bigger size file than the original version and the extended file. The LSB method will work precisely according to the formula of the modified cryptography algorithm. The further research is suggested to focus on the variation of media for content message and the picture format in their encryption.

Keywords: *Digital Image, Cryptographic Algorithm, Least Significant Bit (LSB), Steganography.*

IMPLEMENTASI STEGANOGRAFI PADA CITRA DIGITAL DENGAN MODIFIKASI ALGORITMA VIGENERE CIPHER DAN METODE LEAST SIGNIFICANT BIT (LSB)

Abstrak

Citra digital dapat dijadikan sebagai media penyembunyian pesan. Tentunya dalam mengamankan pesan perlu cara khusus agar isi informasi tidak dapat ditembus. Penyembunyian pesan pada gambar dapat menggunakan teknik steganografi agar pesan yang disembunyikan semakin sulit ditembus maka perlu dikombinasikan dengan algoritma kriptografi. Adanya modifikasi algoritma kriptografi tentu dapat meningkatkan kerahasiaan dari isi pesan yang disembunyikan pada gambar. Penelitian ini bertujuan untuk mengimplementasikan steganografi dalam bentuk aplikasi desktop menggunakan metode *Least Significant Bit* (LSB) yang dikombinasikan dengan modifikasi algoritma *Vigenere Cipher*. Penelitian ini dibantu dengan metode pengembangan perangkat lunak metode *waterfall* dalam pembuatan aplikasinya. Hasil penelitian didapatkan bahwa hasil gambar yang sudah terenkripsi akan berubah menjadi warna abu-abu atau bercitra *grayscale* sehingga pesan yang disembunyikan akan semakin sulit ditemukan oleh *attacker*. Perbedaan yang mencolok pada gambar hasil enkripsi memiliki ukuran *file* yang lebih besar dibandingkan dengan gambar asli dan ekstensi *file* yg dihasilkan. Metode LSB bekerja tepat sesuai dengan perumusan pada modifikasi algoritma kriptografi. Saran penelitian selanjutnya adalah terdapat variasi media isi pesan dan format gambar hasil enkripsi.

Kata kunci: Citra Digital, Algoritma Kriptografi, *Least Significant Bit* (LSB), Steganografi**.**

1. PENDAHULUAN

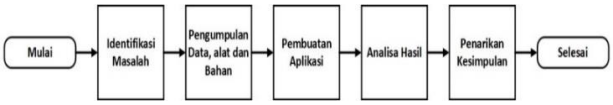
Steganografi merupakan kegiatan atau proses yang bertujuan untuk menyembunyikan data dan informasi yang penting kedalam media lain yang dapat berupa teks, video, audio dan gambar [1]. Steganografi tidak lepas dari kriptografi, perbedaan yang menonjol adalah kriptografi membuat data tidak dapat dipecahkan dan tidak bisa di baca tetapi teks sandi dapat dilihat manusia, berbeda dengan steganografi bertujuan untuk menyembunyikan informasi rahasia sehingga tidak dilihat oleh kasap mata manusia dalam bentuk media, salah satu media steganografi yaitu pada gambar atau citra digital [2]. Steganografi gambar merupakan proses

menyembunyikan informasi rahasia seperti pesan atau teks, video dan audio di dalam citra digital atau *cover image*. Citra digital merupakan bidang 2 dimensi yang memuat informasi penting dan rahasia dan digunakan sebagai barang bukti yang memiliki resiko kehilangan informasi atau kerusakan pada barang bukti kejahatan komputer [3][4][5]. Keamanan pada suatu informasi menjadi sangat penting untuk dilindungi salah satunya dapat menggunakan steganografi. Saat ini banyak masalah muncul yang berhubungan steganografi dengan citra digital atau gambar. Yaitu pada sisi positif steganografi dapat membantu untuk mengamankan data rahasia [6]. Disatu sisi tentunya ada sisi negatif yaitu contoh kasus kejahatan komputer yang terindentifikasi menggunakan steganografi di Eropa adalah pada kelompok *Al-qaeda* menyembunyikan file dokumen kedalam bentuk video yang berisikan rencana penyerangan eropa dan gagasan tentang perebutan kapal pesiar. Pada kasus lain terdapat pada perdagangan narkoba illegal dengan memanfaatkan steganografi gambar untuk berkomunikasi dengan kelompoknya [7][8].

Steganografi telah banyak diteliti pada penelitian sebelumnya, namun dalam beberapa pendekatan, informasi yang disembunyikan menjadi menurunkan kualitas visual pada gambar dengan merubah format gambar [9]. Dalam penelitian sebelumnya, membuat aplikasi untuk menyembunyikan pesan dapat menggunakan metode LSB karena dalam pengimplementasiannya tidak terlalu kompleks dan dapat mengamankan pesan yang tersembunyi didalam gambar [10]. Pada penelitian yang dilakukan oleh Amit Kumar [11] dalam mengamankan pesan dapat dilakukan dengan mengkombinasikan kriptografi dengan steganografi sehingga data yang terdapat dalam media terjamin pada keamanan data. Penelitian selanjutnya membuat aplikasi steganografi mengkombinasikan algoritma substitusi dan *vigenere*, sedangkan untuk menyisipkan pesan menggunakan metode LSB dengan jumlah 4 bit, hasil dari penelitian ini berhasil menyisipkan pesan rahasia [12]. Sedangkan pada penelitian lain, membuat aplikasi steganografi menggunakan algoritma kriptografi berhasil mengamankan dokumen rahasia, namun semakin banyak pesan yang di sisipkan maka proses enkripsi dan dekripsi semakin lama [13]. Disisi lain, penelitian lainnya [14] *waterfall* juga diterapkan sebagai metode pengembangan perangkat lunak dalam pembuatan *webiste*. *Waterfall* dipilih karena dinilai praktis, terawasi dan terstruktur. UML pada penelitian [15] digunakan untuk menggambarkan arsitektur dalam pemrograman berorientasi objek sebagai bahasa standar dalam pembuatan aplikasi yang berfungsi sebagai mendefinisikan kebutuhan, analisis dan desain. Hal ini juga diterapkan pada aplikasi *dekstop* yang akan dibuat pada penelitian ini. UML dipilih karena dianggap tepat sesuai dengan kebutuhan fungsional dan kebutuhan sistem yang ada.

2. METODE PENELITIAN

Penelitian ini membuat aplikasi steganografi dengan algoritma *Vigenere Cipher* yang memodifikasi huruf alfabet diubah menjadi Huruf Hijaiyah. Algoritma *Vigenere Cipher* merupakan salah satu algoritma kriptografi yang sering digunakan untuk mengamankan data dan informasi menggunakan substitusi *pliAlfabetic* dengan cara mensubtitusikan antara pesan (*plaintext)* dan huruf kunci (*key*) yang berhadapan letaknya [16]. Alur penelitian ini dapat dilihat pada gambar berikut.



Gambar 1. Alur Penelitian

Alur penelitian gambar 1 dapat diuraikan sebagai berikut:

2.1 Identifikasi Masalah

Identifikasi masalah pada penelitian ini dilakukan dengan menganalisa kekurangan keamanan informasi pada gambar supaya pesan yang disembunyikan tetap terjaga dari pihak yang berwenang dan mempelajari masalah masalah yang berhubungan dengan penelitian sejenis terutama untuk meningkatkan keamanan pada penyembunyian pesan pada gambar dengan metode *Least Significant Bit* (LSB) dan algoritma *Vigenere Cipher*.

2.2 Pengumpulan Data

Mengumpulkan data menggunakan teknik kualitatif untuk mendapatkan informasi dengan studi dokumen yang berasal dari dari jurnal, buku dan sumber lainnya yang membahas tentang steganografi dengan algoritma *Vigenere Cipher* dan metode *Least Significant Bit* [17] dan menentukan kebutuhan fungsional yang merupakan proses awal pembuatan aplikasi dan non fungsional yang menentukan perangkat lunak dan perangkat keras [18].

2.3 Pembuatan Aplikasi

Aplikasi steganografi ini dibuat menggunakan bahasa pemrograman Java. Java merupakan bahasa pemrograman berorientasi objek yang memiliki banyak *libraries* yang cocok digunakan seperti membuat animasi halaman *web*, membuat aplikasi interaktif yang dapat juga digunakan di *handphone* dan *desktop*. Menurut [19] memilih bahasa permoraman Java dianggap salah satu bahasa pemrograman yang sudah berorientasi objek, platform independen, terdistribusi secara cepat yang dapat berjalan di sistem operasi apapun [20]. Dalam pembuatan aplikasi dibutuhkan metode pengembang perangkat lunak salah satunya menggunakan metode *waterfall* [16][17].

2.3.1 Analisis

Proses ini menentukan hal-hal yang dibutuhkan dalam pembuatan aplikasi steganografi meliputi bahasa pemrograman yang digunakan, platform, sistem operasi dan menentukan kebutuhan fungsional dan non fungsional.

2.3.2 Desain

Tahap selanjutnya merupakan tahap untuk menggambarkan bentuk visualisasi sistem yang akan dibuat menggunakan UML (*Unified Modelling Language*) dan membuat tampilan aplikasi yang akan dibuat UI (*User Interface*).

2.3.3 Implementasi

Tahap implementasi yaitu untuk menerapkan rancangan sistem dan antarmuka yang sudah dibuat setelah tahap desain. Tahap ini melakukan proses implementasi dalam bentuk program aplikasi. Implemetasi sistem yang paling utama adalah proses enkripsi dan dekripsi.

Proses enkripsi yaitu proses untuk menyembunyikan pesan dengan algoritma *Vigenere Cipher* yang telah dimodifikasi dengan huruf Hijaiyah dan untuk penyisipan pesan menggunakan metode *Least* Significant *Bit*. Langkah awal proses enkripsi yaitu dengan memasukkan pesan dan kunci kemudian memilih gambar sebagai media penyisipan pesan tersembunyi. Sedangkan pada proses dekripsi merupakan proses untuk menampilkan pesan dari gambar yang sudah dienkripsi.

Proses dekripsi dilakukan dengan cara memasukkan gambar yang telah disisipkan kemudian masukan kunci yang sama pada saat proses enkripsi.

2.3.4 Pengujian

Pengujian yang dilakukan yaitu menyatukan komponen pada aplikasi steganografi dan proses pengujian terhadap aplikasi. Pada tahap ini diuji coba proses utama yaitu proses enkripsi dan proses dekripsi agar sesuai dengan alur kerja sistem yang diinginkan.

2.3.5 Perawatan

Tahap terakhir pada *waterfall* yaitu tahap perawatan. Tahap perawatan yang akan dilakukan perbaikan terhadap kesalahan program yang terjadi pada saat proses uji coba sehingga sistem dapat berjalan baik ketika digunakan.

2.4 Analisa Hasil

Pada tahap ini peneliti akan mengalisa perubahan yang terjadi pada objek yaitu gambar asli dan gambar yang telah disisipi pesan yang tersaji pada hasil metadata. Metadata merupakan sekumpulan data yang mengandung informasi pada data tersebut seperti nama, jenis *file*, ekstensi *file*, tanggal pembuatan *file* dan lain lain [23].

2.5 Penarikan Kesimpulan

Hasil yang akan diperoleh akan disimpulkan secara spesifik dan informatif.

3. HASIL DAN PEMBAHASAN

3.1 Pembuatan Perangkat Lunak

Penelitian ini menggunakan metode *waterfall* sebagai metode pengembangan perangkat lunak. Keunggulan dari metode *waterfall* adalah praktis pada rekayasa pada *software* sehingga dapat menjaga kualitas pada *software* karena pengembangannya terstruktur dan terawasi [24]. Pembuatan perangkat lunak dapat diuraikan sebagai berikut:

3.1.1 Analisis

Dalam tahap analisis dilakukan analisa pada sistem yaitu dengan menenuntukan kebutuhan fungsional dan non fungsional. Kebutuhan fungsional terdiri dari analisa masukan, analisa proses dan antarmuka. Analisa masukan meliputi input gambar pada *encode* dan *decode*, *input* pesan yang akan disembunyikan pada *encode*, dan *input key* atau kata kunci pada *encode* dan *decode*. Pada analisa masukan, sebelum masuk proses enkripsi yang pertama mengetahui tabel konversi huruf alfabet ke dalam Huruf Hijaiyah yang terdapat pada tabel 1.

Tabel 1. Konversi huruf A-Z

Angka	Huruf Alfabet	Huruf Hijaiyah
1	A	ا
2	B	ب
3	C	ث
4	D	د
5	E	ع
6	F	ف
7	G	غ
8	H	ه
9	I	ح
10	J	ج
11	K	ك
12	L	ل
13	M	م
14	N	ن
15	O	ط
16	P	ظ
17	Q	ق
18	R	ر
19	S	س
20	T	ت
21	U	ص
22	V	ض
23	W	و
24	X	ش
25	Y	ي
26	Z	ز

Setelah mengetahui tabel 1 konversi huruf alfabet huruf A - Z ke dalam Hijaiyah, selanjutnya mengetahui Tabel Algoritma *Vigenere Cipher* yang dimodifikasi menggunakan Huruf Hijaiyah berikut untuk dapat dilihat pada gambar 2 dan gambar 3.

	A	B	C	D	E	F	G	H	I	J	K	L
A	ا	ب	ث	د	ع	ف	غ	ه	ح	چ	ك	ل
B	ب	ث	د	ع	ف	غ	ه	ح	چ	ك	ل	م
C	ث	د	ع	ف	غ	ه	ح	چ	ك	ل	م	ن
D	د	ع	ف	غ	ه	ح	چ	ك	ل	م	ن	ط
E	ع	ف	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ
F	ف	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق
G	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر
H	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر	س
I	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر	س	ت
J	چ	ك	ل	م	ن	ط	ظ	ق	ر	س	ت	ص
K	ك	ل	م	ن	ط	ظ	ق	ر	س	ت	ص	ض
L	ل	م	ن	ط	ظ	ق	ر	س	ت	ص	ض	و
M	م	ن	ط	ظ	ق	ر	س	ت	ص	ض	و	ش
N	ن	ط	ظ	ق	ر	س	ت	ص	ض	و	ش	ي
O	ط	ظ	ق	ر	س	ت	ص	ض	و	ش	ي	ز
P	ظ	ق	ر	س	ت	ص	ض	و	ش	ي	ز	ا
Q	ق	ر	س	ت	ص	ض	و	ش	ي	ز	ا	ب
R	ر	س	ت	ص	ض	و	ش	ي	ز	ا	ب	ث
S	س	ت	ص	ض	و	ش	ي	ز	ا	ب	ث	د
T	ت	ص	ض	و	ش	ي	ز	ا	ب	ث	د	ع
U	ص	ض	و	ش	ي	ز	ا	ب	ث	د	ع	ف
V	ض	و	ش	ي	ز	ا	ب	ث	د	ع	ف	غ
W	و	ش	ي	ز	ا	ب	ث	د	ع	ف	غ	ه
X	ش	ي	ز	ا	ب	ث	د	ع	ف	غ	ه	ح
Y	ي	ز	ا	ب	ث	د	ع	ف	غ	ه	ح	چ
Z	ز	ا	ب	ث	د	ع	ف	غ	ه	ح	چ	ك

Gambar 2. Tabel *Vigenere Cipher* Huruf Hijaiyah A – L

	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	م	ن	ط	ظ	ق	ر	س	ت	ص	ض	و	ش	ي	ز
B	ن	ط	ظ	ق	ر	س	ت	ص	ض	و	ش	ي	ز	ا
C	ط	ظ	ق	ر	س	ت	ص	ض	و	ش	ي	ز	ا	ب
D	ظ	ق	ر	س	ت	ص	ض	و	ش	ي	ز	ا	ب	ث
E	ق	ر	س	ت	ص	ض	و	ش	ي	ز	ا	ب	ث	د
F	ر	س	ت	ص	ض	و	ش	ي	ز	ا	ب	ث	د	ع
G	س	ت	ص	ض	و	ش	ي	ز	ا	ب	ث	د	ع	ف
H	ت	ص	ض	و	ش	ي	ز	ا	ب	ث	د	ع	ف	غ
I	ص	ض	و	ش	ي	ز	ا	ب	ث	د	ع	ف	غ	ه
J	ض	و	ش	ي	ز	ا	ب	ث	د	ع	ف	غ	ه	ح
K	و	ش	ي	ز	ا	ب	ث	د	ع	ف	غ	ه	ح	چ
L	ش	ي	ز	ا	ب	ث	د	ع	ف	غ	ه	ح	چ	ك
M	ي	ز	ا	ب	ث	د	ع	ف	غ	ه	ح	چ	ك	ل
N	ز	ا	ب	ث	د	ع	ف	غ	ه	ح	چ	ك	ل	م
O	ا	ب	ث	د	ع	ف	غ	ه	ح	چ	ك	ل	م	ن
P	ب	ث	د	ع	ف	غ	ه	ح	چ	ك	ل	م	ن	ط
Q	ث	د	ع	ف	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ
R	د	ع	ف	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق
S	ع	ف	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر
T	ف	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر	س
U	غ	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر	س	ت
V	ه	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر	س	ت	ص
W	ح	چ	ك	ل	م	ن	ط	ظ	ق	ر	س	ت	ص	ض
X	چ	ك	ل	م	ن	ط	ظ	ق	ر	س	ت	ص	ض	و
Y	ك	ل	م	ن	ط	ظ	ق	ر	س	ت	ص	ض	و	ش
Z	ل	م	ن	ط	ظ	ق	ر	س	ت	ص	ض	و	ش	ي

Gambar 3. Tabel *Vigenere Cipher* Huruf Hijaiyah M-Z

Gambar 2 merupakan tabel algoritma *Vigenere Cipher* A - L dan gambar 3 merupakan tabel algoritma *Vigenere Cipher* L - Z yang telah dimodifikasi dengan huruf Hijaiyah yang akan diterapkan kedalam bentuk perintah bahasa pemrograman *(coding)* pembuatan aplikasi.

Sedangkan analisa kebutuhan proses meliputi proses simpan gambar dan simpan pesan hasil dekripsi. Kebutuhan antarmuka pada Aplikasi steganografi juga memerlukan antarmuka atau tampilan antara lain tampilan *encode*, *decode* dan *about.*

Selanjutnya menentukan kebutuhan non fungsional yang terbagi menjadi 2 yaitu perangkat keras dan perangkat lunak. Pada perangkat keras yang digunakan dalam penelitian ini menggunakan laptop Lenovo L440 dengan spesifikasi prosessor I3-4100M CPU @2.50Hz, RAM 8 GB, penyimpanan SSD 120 GB dan *system type* 64 bit. Sedangkan pada perangkat lunak yang digunakan pada pembuatan aplikasi steganografi meliputi windows 10, *text editor* (IDE) Neatbeans versi 8.2 dan proses desain menggunakan Visio 2019.

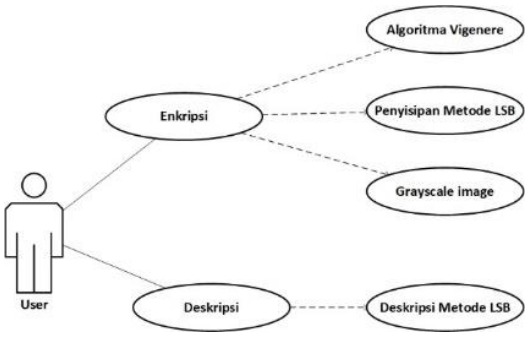
Dalam pembuatan aplikasi steganografi ini menggunakan bahasa pemrograman Java yang berbasis desktop. Pada aplikasi menerapkan algoritma kriptografi *Vigenere Cipher* yang telah dimodifikasi kedalam Huruf Hijaiyah dan untuk penyisipan data menggunakan metode *Least Significant Bit*. Pada tahap ini akan terjadi 2 proses utama yang pertama yaitu proses enkripsi dan dekripsi. Proses enkripsi yaitu proses menyembunyikan pesan dengan memasukkan gambar, teks pesan dan kata kunci yang digunakan untuk menghasilkan *stego image*. Tahap selanjutnya yaitu proses dekripsi. Dekripsi merupakan proses yang bertujuan untuk memisahkan antara gambar dan pesan sehingga pesan asli dapat dibaca oleh penerima.

3.1.2 Desain

Dalam pembuatan aplikasi steganografi ini terdapat perancangan alur sistem menggunakan desain *use case* dan *activity diagram*. Selain menggunakan UML dalam tahap ini membuat rancangan antarmuka atau yang disebut *user interface*.

1. UML (*Unified Modeling Language*)

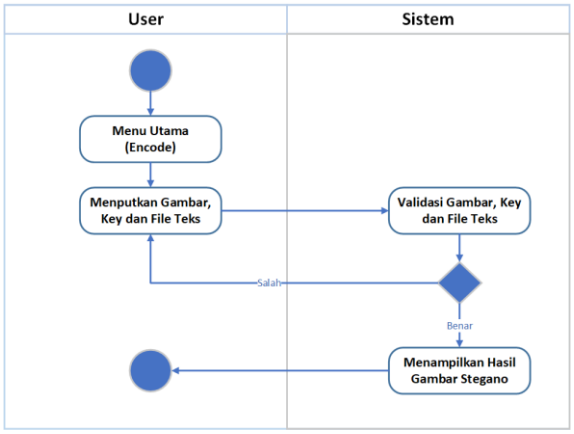
a. *Use Case*



Gambar 4. *Use Case*

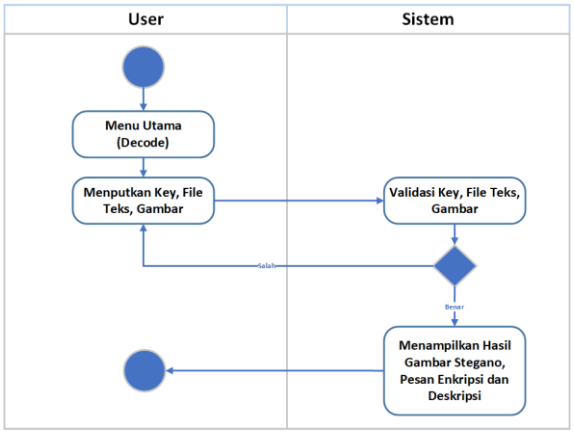
Gambar 4 merupakan gambar *use case* yang terdapat satu *actor* yaitu selaku *user* atau pengguna aplikasi steganografi yang dapat melakukan proses enkripsi dan dekripsi.

b. Activity Diagram



Gambar 5. *Activity Diagram* Enkripsi

Gambar 5 merupakan *diagram activity* proses enkripsi. Proses ini memasukkan kata kunci dan *file text* berformat .txt yang akan disisipkan kedalam gambar.

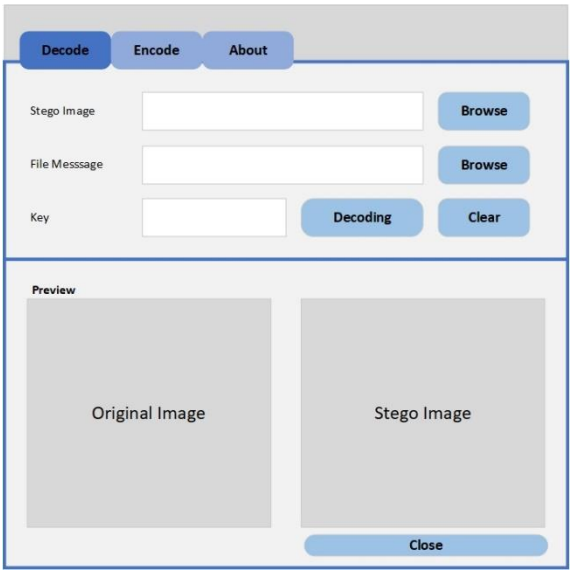


Gambar 6. *Diagram Activicty* Dekripsi

Selanjutnya gambar 6 merupakan *diagram activity* proses dekripsi. Proses awal dekripsi adalah dengan memasukkan gambar yang sudah mengandung pesan dan kata kunci yang digunakan untuk menampilkan pesan rahasia. Proses dekripsi bertujuan untuk memisahkan *stego image* dengan pesan rahasia yang sudah di enkripsi sehingga pesan dapat ditampilkan.

2. *User Interface*

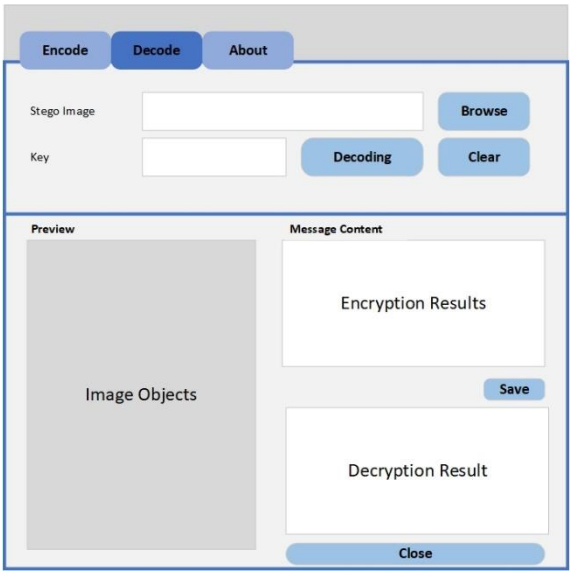
a. Tampilan *Encode*



Gambar 7. *User Interface Encode*

Gambar 7 merupakan rancangan *user interface encode* berfungsi untuk melakukan proses enkripsi sekaligus tampilan awal pada saat aplikasi berhasil dibuka. Tampilan ini terdapat akan terdapat 3 inputan yaitu gambar, pesan dan kata kunci. Tampilan ini akan menampilkan gambar asli dan gambar yang telah disisipi pesan.

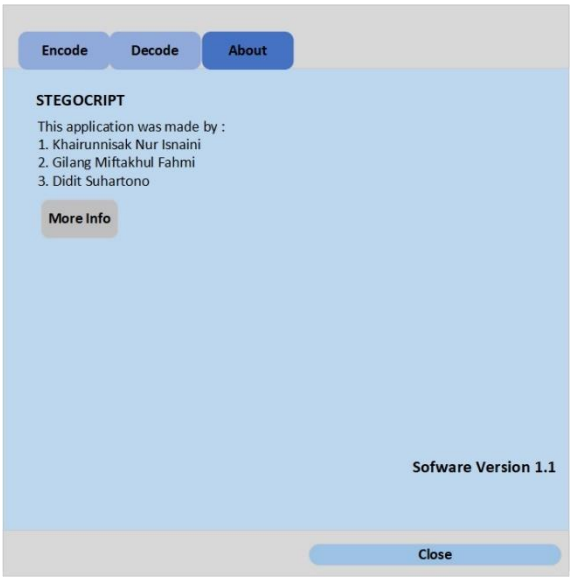
b. Tampilan *Decode*



Gambar 8. *User Interface Decode*

Gambar 8 merupakan rancangan *user interface* yang berfungsi sebagai proses dekripsi. Tampilan ini menampilkan *stego image* yang akan didekripsi. Setelah proses dekripsi tampilan ini akan menampilkan pesan hasil enkripsi dan pesan hasil dekripsi atau pesan asli.

c. Tampilan *About*



Gambar 9. *User Interface About*

Gambar 9 merupakan *user interface about* yang befungsi untuk menjelaskan aplikasi steganografi kepada pengguna seperti versi, pembuat aplikasi dan deskripsi singkat aplikasi.

3.1.3 Implementasi

Tahap implementasi alur aplikasi terdiri dari 2 proses yaitu proses enkripsi dan proses dekripsi. proses ini adalah proses *coding* atau membuat program aplikasi menggunakan *text editor* Neatbeans versi 8.2. Aplikasi ini dibuat untuk aplikasi desktop pada sistem operasi windows. Sebelum masuk proses *coding* berikut proses enkripsi. hal-hal yang harus diperhatikan adalah [25] mengetahui tabel konversi alfabet dan tabel *Vigenere Cipher*. Tahap selanjutnya, yang terlihat pada gambar 10.

```
char mapPesanCr[] = {' ','\n','A', 'B', 'C', 'D', 'E',
                     'F', 'G', 'H', 'I', 'J', 'K', 'L',
                     'M', 'N', 'O', 'P', 'Q', 'R', 'S',
                     'T', 'U', 'V', 'W', 'X', 'Y', 'Z'};
char mapHasilCr[] = {' ','\n','ا', 'ب', 'ت', 'د', 'ع',
                     'ف', 'غ', 'ه', 'ح', 'ج', 'ك', 'ل',
                     'م', 'ن', 'ط', 'ظ', 'ق', 'ر', 'س',
                     'ت', 'ص', 'ض', 'و', 'ش', 'ي', 'ز'};
```

Gambar 10. *Sorce code inisialisasi algoritma Vigenere Cipher*

Gambar 10 *Source code* inisialisasi pada konversi algoritma *Vigenere Cipher* yang dimodifikasi dengan Huruf Hijaiyah dengan bahasa pemrograman Java yang dapat dilihat pada gambar 10. *Source code* ini merupakan penerapan dari algoritma *Vigenere Cipher* yang telah dimodifikasi huruf alfabet diubah menjadi Huruf Hijaiyah.

```
private static final char[] alphnum = {'A', 'B', 'C', 'D', 'E', 'F', 'G', 'H',
                                       'I', 'J', 'K', 'L', 'M', 'N', 'O', 'P',
                                       'Q', 'R', 'S', 'T', 'U', 'V', 'W', 'X',
                                       'Y', 'Z', '0', '1', '2', '3', '4', '5',
                                       '6', '7', '8', '9', '@', '?', '#', '-',
                                       '+', '!', '$', '%', '&', '*', '(', ')',
                                       '_', '+', '=', '.', ',', ';', ':', '"',
                                       '\'', '}', '{', '\\', '/', '^', '<', '>',
                                       '[', ']', '^', '|', '`', '~'};
```

Gambar 11. *Source code inisialisasi* hasil pesan

Source code inisialisasi huruf abjad alfabet bahasa pemrograman Java yang dapat dilihat pada gambar 11. Proses inisialisasi ini digunakan untuk menampilkan hasil pesan dengan keluaran huruf alfabet.

a. Menentukan pesan atau *plain text* dan kata kunci yang akan proses bersama algoritma yang akan menghasilkan *ciphertext* gambar 12.

Plaintext	:	MAHASISWA
Key	:	mahas
Chippertext	:	ص ي ك ع ص ج ج ط ن

Gambar 12. hasil *ciphertext*

b. Hasil *ciphertext* akan diubah kembali menjadi huruf alfabet menggunakan bantuan tabel ASCII yang dapat dilihat pada tabel 2.

Tabel 2. ASCII 8 bit

Decimal	Binary	Value
65	01000001	A
66	01000010	B
67	01000011	C
68	01000100	D
69	01000101	E
70	01000110	F
71	01000111	G
72	01001000	H
73	01001001	I
74	01001010	J
75	01001011	K
76	01001101	L
77	01001101	M
78	01001110	N
79	01001111	O
80	01010000	P
81	01010001	Q
82	01010010	R
83	01010011	S
84	01010100	T
85	01010101	U
86	01010110	V
87	01010111	W
88	01011000	X
89	01011001	Y
90	01011010	Z

Pada tabel 2 merupakan tabel ASCII 8 bit konversi huruf A - Z yang digunakan untuk menerapkan proses penyisipan pesan menggunakan metode LSB. Selanjutnya yaitu membaca *chipertext* dan konversi dalam binary.

Chippertext	:	ص	ي	ك	ع	ص	ج	ج	ط	ن
Huruf Alfabet LSB	:	U	Y	K	E	U	J	J	O	N
		01010101-01011001-01001011								
Binary	:	1000101-01010101-01001010								
		1001010-01001111-01001110								

Gambar 13 Pembacaan *Ciphertext* dan Konversi kedalam *Bit*

Gambar 13 merupakan pembacaan hasil *ciphertext*. Kunci pada proses enkripsi adalah **miftah** yang berjumlah 6 digit. Berdasarkan tabel ASCII, urutan ke 6 pada tabel akan di konversikan kedalam huruf F dengan *binary* 01000110.

Binary LSB	:	01001110-01001111-01001011		
		1001010 - 01010100-01000102		
		01001011-01011000-01010101		
Huruf Alfabet (LSB)	:	N	O	J
		J	T	E
		K	H	U

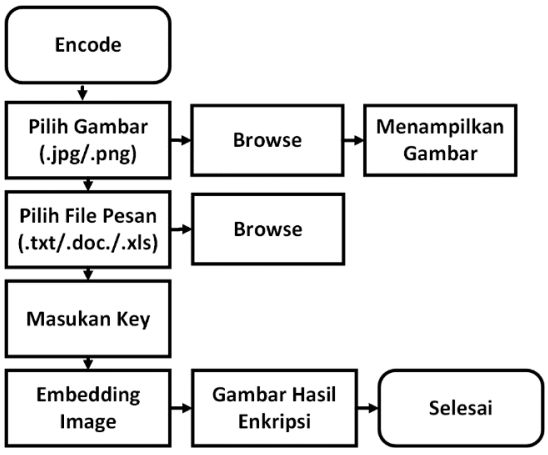
Gambar 14. Hasil metode LSB

```
private byte[] bit_conversion(int i) {

    byte byte3 = (byte) ((i & 0xFF000000) >>> 24); //0
    byte byte2 = (byte) ((i & 0x00FF0000) >>> 16); //0
    byte byte1 = (byte) ((i & 0x0000FF00) >>> 8); //0
    byte byte0 = (byte) ((i & 0x000000FF));
    return (new byte[]{byte3, byte2, byte1, byte0});
}
```

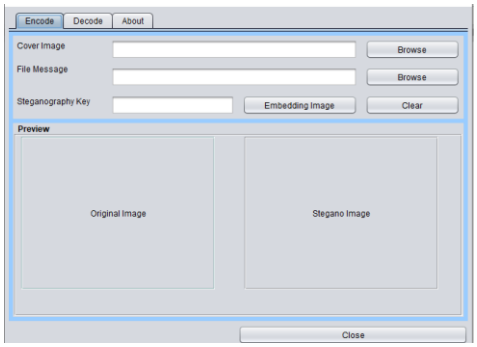
Gambar 15. *Source code* LSB

Gambar 14 menunjukkan gambar hasil *ciphertext* yang dikonversi kedalam *binary* LSB. Proses metode LSB mengganti *bit bit* terakhir gambar. Sedangkan gambar 15 merupakan *source code* penerapan metode LSB. Pengoperasian aplikasi terdapat langkah langkah proses enkripsi sebagai berikut:



Gambar 16. Penggunan Aplikasi Proses *Encode*

Gambar 16 tahap awal enkripsi yaitu membuat *plain text* yang akan dimasukan kedalam *file* berformat .txt pada tampilan *encode*. Selanjutnya memilih gambar yang akan disisipi pesan. Langkah terakhir memasukkan kata kunci dan klik *embedding image* untuk melakukan proses enkripsi. Berikut tampilan *encode* pada proses enkripsi:



Gambar 17. Tampilan *Encode*

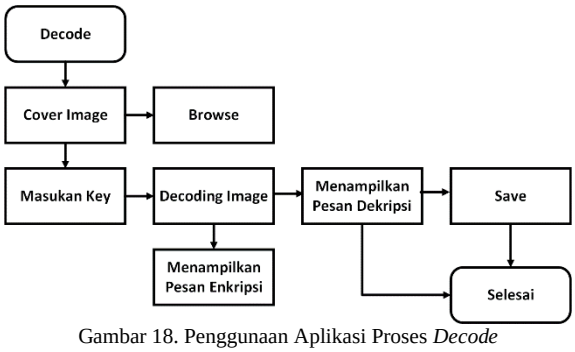
Gambar 17 merupakan tampilan *encode* yang berfungsi sebagai tampilan untuk enkripsi pada aplikasi steganografi. Fungsi-fungsi pada antarmuka gambar 17 antara lain:

a. Antarmuka untuk memasukkan gambar, *key*, dan teks pesan.
b. Antarmuka untuk melakukan proses enkripsi yang menghasilkan *stego jmage*
c. Antarmuka untuk menampilkan gambar asli dan gambar hasil enkripsi.

Sedangkan pada proses dekripsi, hal hal yang harus di perhatikan meliputi:

a. Mengetahui hasil *binary text* pada metode LSB dari *ciphertext*
b. Mengetahui hasil binary text dari kunci
c. Mengubah Kembali hasil metode LSB kedalam *binnary text* pada kunci
d. Mengubah hasil *binary text* kedalam huruf alfabet dengan mengkonversikan huruf Arab/Hijaiyah.
e. Terakhir mengkonversikan isi pesan dan kunci menjadi pesan asli.

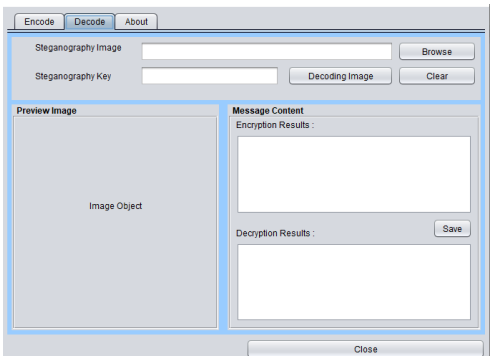
Selanjutnya penggunaan aplikasi proses dekripsi dapat dilihat pada gambar berikut:



Gambar 18. Penggunaan Aplikasi Proses *Decode*

Proses dekripsi menggunakan langkah-langkah gambar 18, langkah awal yaitu dengan memasukkan gambar yang sudah disisipi pesan. Selanjutnya memasukkan kata kunci *(key)* yang sesuai pada saat proses enkripsi. Tekan tombol *Decoding Image*

untuk melakukan proses dekripsi pada gambar. Hasil pesan enkripsi dan pesan dekripsi akan ditampilkan dan menghasilkan gambar abu-abu atau *grayscale* untuk memaksimalkan isi pesan. Kemudian pesan dapat disimpan sebagai teks berformat .txt. Proses tersebut dapat dilihat pada gambar 19.



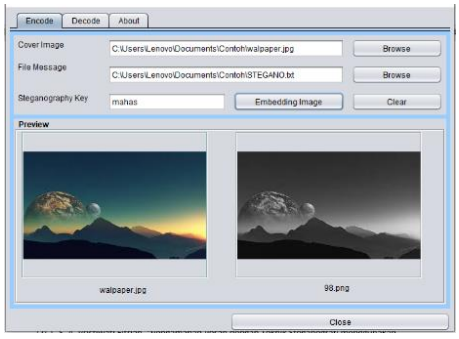
Gambar 19. Tampilan *Decode*

Gambar 19 merupakan bentuk tampilan *decode* yang digunakan sebagai proses dekripsi yang bertujuan untuk menampilkan pesan yang terkandung dari *stego image*. Fungsi-fungsi pada antarmuka gambar 18 antara lain:

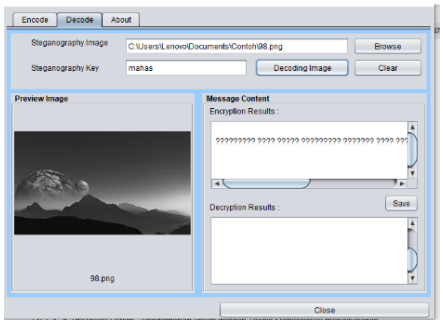
a. Antarmuka untuk memasukkan gambar *stego image* dan *key*
b. Antarmuka untuk melakukan proses dekripsi untuk menampilkan *stego image*, pesan enkripsi dan hasil pesan dekripsi

3.1.4 Pengujian

Tahap ini merupakan tahap pengujian aplikasi proses enkripsi dan proses dekripsi. Aplikasi pada tahap ini sudah berbentuk .exe (integrasi), sehingga aplikasi dapat digunakan di komputer atau laptop yang diinginkan. Berikut tampilan aplikasi:



Gambar 20. *Testing Encode*



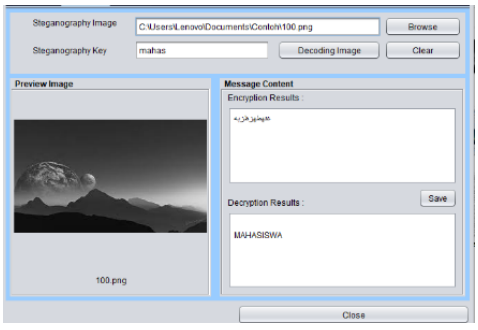
Gambar 21. *Testing Decode*

Gambar 20 merupakan pengujian tampilan *encode* dan gambar 21 merupakan hasil pengujian tampilan *decode*. Saat proses enkripsi berjalan dengan baik yang meliputi memasukkan gambar, teks, *key* dan hasil *preview* (gambar asli dan gambar hasil enkripsi). Namun pada saat *decode* saat aplikasi dijalankan terjadi adanya *bug* saat proses dekripsi pada *encryption result* tidak menampilkan hasil enkripsi yang berbentuk Huruf Hijaiyah pada tampilan *decode*.

3.1.5 Perawatan

Proses perawatan sistem aplikasi berguna untuk menjaga sistem berjalan dengan baik tanpa ada kesalahan sehingga aplikasi dapat terus digunakan oleh pengguna.

Sebelumnya, ditemukan *bug* saat proses dekripsi yaitu sistem tidak dapat menampilkan pesan enkripsi, hal ini terjadi karena *Virtual Machine* pada aplikasi tidak mendukung pada Huruf Hijaiyah. Solusi dari hal tersebut yaitu dengan mengkonversi pesan hasil enkripsi menjadi *encoding* UTF-8 serta mengecek langkah-langkah secara urut pada program sehingga pesan enkripsi dapat ditampilkan yang dapat dilihat pada gambar 22.



Gambar 22. *maintenance* proses dekripsi

3.2 Hasil Analisis Aplikasi

Hasil analisis aplikasi yaitu dengan menganalisis meta data dari gambar asli dan gambar yang sudah disisipi pesan. Aplikasi steganografi yang baik memiliki kriteria antara lain *Fidelity*, *Recovery* dan *Capacity*. Hasil dari proses enkripsi dan dekripsi steganografi dapat dilihat pada gambar berikut:

Gambar 23. Gambar Asli

Gambar 24. Gambar Hasil Enkripsi

Gambar 23 merupakan gambar asli dan gambar 24 merupakan gambar hasil proses dekripsi yang berwarna abu-abu yang merupakan hasil *encode*. Tujuan gambar *encode* berwarna abu-abu pada steganografi digunakan untuk menyamarkan pesan yang tersembunyi pada gambar [19][20]. Hasil pada metadata gambar dapat dilihat metadata pada tabel 3.

Tabel 3. Hasil Metadata

Daftar metadata	Contoh Gambar	
	Gambar	
	Asli	Hasil Enkripsi
Nama Gambar	Asli.jpg	Hasil Enkripsi.png
Ukuran dalam *byte*	709,76 kB	5,93 MB
Ukuran dalam *pixel*	309 x 163	309 x 163
Number of unique color	87047	286
Current directory index	3 / 3	2 / 3
Loaded in (*milisc*)	0	0
Bit depth	24	24
Format	JPEG	PNG

Hasil metadata tabel 3 gambar 23 yang merupakan gambar asli berukuran 709,76 kB dalam *byte*, *pixel* gambar berukuran 309 x 163, *Number of unique color* 87047, *current directory index* 3/3, *Loaded in (milisc)* 0, *bit depth* 24 dan berformat JPEG.

Hasil metadata tabel 3 gambar 24 yang merupakan gambar hasil enkripsi berukuran 5,93 MB dalam *byte*, *pixel* gambar berukuran 309 x 163, *Number of unique color* 286, *currunt directory index* 2/3, *Loaded in (milisc)* 0, *bit depth* 24 dan berformat PNG.

Perbedaan yang paling menonjol adalah pada gambar hasil enkripsi yang telah disisipi pesan memiliki *size* yang yang lebih besar dibandingkan dengan gambar asli meningkat lebih dari 5 MB. Perbedaan selanjutnya pada metadata *number of unique* pada gambar hasil sangat menurun drastis karena warna telah berubah menjadi abu-abu yaitu dari 87047 menjadi 286 dan perbedaan yang terakhir terlihat pada format gambar hasil berubah menjadi PNG dari yang sebelumnya pada gambar asli berformat JPEG.

4. KESIMPULAN

Berdasarkan implementasi dan hasil analisis aplikasi steganografi ini berhasil dibuat sesuai dengan kebutuhan dan tujuan yaitu penyembunyian pesan pada gambar dalam rangka mengamankan informasi. Kedua, modifikasi algoritma *Vigenere Cipher* dengan mengubah huruf alfabet ke dalam Huruf Hijaiyah turut membantu meningkatkan keamaman pesan yang disembunyikan yang dilihat dari hasil *encode*. Selain itu, *hasil stego image* yang berwarna abu-abu dapat menyamarkan pesan agar tidak mudah diketahui isi pesan. Isi pesan yang disamarkan dan dapat ditampilkan kembali sehingga memenuhi kriteria *recovery* yaitu pesan yang sudah dienkripsi dapat dibaca kembali.

DAFTAR PUSTAKA

[1] N. Subramanian, O. Elharrouss, S. Al-Maadeed, and A. Bouridane, "Image Steganography: A Review of the Recent Advances," *IEEE Access*, vol. 9, pp. 23409–23423, 2021, doi: 10.1109/ACCESS.2021.3053998.

[2] M. B. Akbar and E. V Haryanto, "Aplikasi Steganografi dengan Menggunakan Metode F5," *... Sist. Inf. dan Teknol. ...*, no. April 2013, pp. 165–178, 2015, [Online]. Available: https://ejurnal.dipanegara.ac.id/index.php/jusiti/article/view/58.

[3] K. Khairunnisak, H. Ashari, and A. P. Kuncoro, "Analisis Forensik Untuk Mendeteksi Keaslian Citra Digital Menggunakan Metode Nist," *J. Resist. (Rekayasa Sist. Komputer)*, vol. 3, no. 2, pp. 72–81, 2020, doi: 10.31598/jurnalresistor.v3i2.634.

[4] K. Lysander, A. M. Simarmata, D. Lusandy, and I. Iswandi, "Pengamanan Citra Dengan Kombinasi Modified Serpent IWT Dengan Modified Logistic Chaotic Map," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 8, no. 3, pp. 1090–1104, 2021, doi: 10.35957/jatisi.v8i3.1019.

[5] D. Alamsyah, "Pengenalan Mobil pada Citra Digital Menggunakan HOG-SVM," *Jatisi*, vol. 1, no. 2, pp. 162–168, 2017.

[6] Hermansa, R. Umar, and A. Yudhana, "Pangamanan Pesan Menggunakan Kriptografi Caesar Cipher dan Steganografi EOF pada Citra," *J. Sains Komput. Mat.*, vol. Vol 4, pp. 1–13, 2020.

[7] A. P. Saputra and N. Widiyasono, "Analisis Digital Forensik pada File Steganography (Studi kasus : Peredaran Narkoba)," *J. Tek. Inform. dan Sist. Inf.*, vol. 3, no. 1, pp. 179–190, 2017, doi: 10.28932/jutisi.v3i1.594.

[8] S. Gallagher, "Steganography: how al-Qaeda hid secret documents in a porn video," *arstechnica.com*, 2012. https://arstechnica.com/information-technology/2012/05/steganography-how-al-qaeda-hid-secret-documents-in-a-porn-video/ (accessed May 20, 2022).

[9] M. Saravanan and A. Priya, "An Algorithm for Security Enhancement in Image Transmission Using Steganography," *J. Inst. Electron. Comput.*, vol. 1, no. 1, pp. 1–8, 2019, doi: 10.33969/jiec.2019.11001.

[10] A. Hafiz, "Steganografi Berbasis Citra Digital Untuk Menyembunyikan Data Menggunakan Metode Least Significant Bit (Lsb)," *J. Cendikia*, vol. 17, no. 1 April, pp. 194–198, 2019, [Online]. Available: https://jurnal.dcc.ac.id/index.php/JC/article/view/201.

[11] A. K. Agrahari *et al.*, "Comprehensive Survey on Image Stegnography Using LSB With AES," vol. 13, no. 8, pp. 5841–5844, 2018.

[12] P. Hernawandra, S. Supriyadi, and U. T. Lenggana, "Aplikasi Steganografi Menggunakan LSB 4 Bit Sisipan dengan Kombinasi Algoritme Substitusi dan Vigenere Berbasis Android," *J. Teknol. dan Sist. Komput.*, vol. 6, no. 2, pp. 44–50, 2018, doi: 10.14710/jtsiskom.6.2.2018.44-50.

[13] B. E. Widodo and A. S. Purnomo, "Implementasi Advanced Encryption Standard Pada Enkripsi Dan Dekripsi Dokumen Rahasia Ditintelkam Polda Diy," *J. Tek. Inform.*, vol. 1, no. 2, pp. 69–77, 2020, doi: 10.20884/1.jutif.2020.1.2.21.

[14] T. Rijanandi *et al.*, "Web-Based Application with SDLC Waterfall Method on Population Administration and Registration Information System (Case Study: Karangklesem Village, Purwokerto)," *J. Tek. Inform.*, vol. 3, no. 1, pp. 99–104, 2022, [Online]. Available: https://doi.org/10.20884/1.jutif.2022.3.1.145.

[15] D. Djesmedi, F. Firdalius, and M. R. Aditia, "MULTIMEDIA-BASED INTERACTIVE LEARNING MEDIA FOR DEAF AND MENTALLY DISABLED ELEMENTARY STUDENTS ON SLBN 1 LENGAYANG MEDIA PEMBELAJARAN INTERAKTIF BERBASIS MULTIMEDIA UNTUK SISWA SDLB TUNARUNGU DAN TUNAGRAHITA PADA SLBN 1 LENGAYANG," vol. 3, no. 2, pp. 437–445, 2022.

[16] E. N. T. Guruh M Arindra Pratama, "MODIFIKASI ALGORITMA VIGENERE CIPHER MENGGUNAKAN METODE CATALAN NUMBER DAN DOUBLE COLUMNAR TRANSPOSITION," *J. Teknol. Yogyakarta*, vol. 4, no. 5, pp. 31–40, 2015.

[17] P. Fitrian and T. S. Alasi, "PENGAMANAN PESAN DENGAN TEKNIK STEGANOGRAFI MENGGUNAKAN METODE LEAST SIGNIFICANT BIT PADA CITRA DIGITAL," vol. I, no. 2, pp. 35–38, 2019.

[18] S. Herwindyo, H. A. Wibawa, and U. Diponegoro, "Steganografi pesan teks ke dalam citra dengan metode lsb pada ruang warna ycocg terdekomposisi iwt," vol. 9, pp. 18–23, 2017.

[19] Y. Nyura, "Pembuatan Aplikasi Pembelajaran Bahasa Inggris Pada Handphone dengan J2ME," *J. Inform. Mulawarman*, vol. 5, no. 3, pp. 18–27, 2010.

[20] D. I. D. Merangin *et al.*, "PERANCANGAN DAN PENERAPAN SISTEM INVENTORY BARANG PADA TOKO BIG STORE PADANG DENGAN MENGGUNAKAN BAHASA PEMROGRAMAN JAVA DAN MYSQL," *J. Inf. Technol. Comput. Sci.*, vol. 2, no. 2, p. 2016, 2018, [Online]. Available: https://doi.org/10.1016/j.gecco.2019.e00539%0Ahttps://doi.org/10.1016/j.foreco.2018.06.029%0Ahttp://www.cpsg.org/sites/cbsg.org/files/documents/Sunda Pangolin National Conservation Strategy and Action Plan %28LoRes%29.pdf%0Ahttps://doi.org/10.1016/j.forec.

[21] N. D. Dzikrillah, "Aplikasi Pelayanan Jasa Laundry Di Zazi Laundry Berbasis Web Dan Nexmo Sms Api," *JATISI (Jurnal Tek. Inform. dan Sist. Informasi)*, vol. 8, no. 4, pp. 1854–1867, 2021, doi: 10.35957/jatisi.v8i4.1195.

[22] M. Tabrani, "Penerapan Metode Waterfall Pada Sistem Informasi Inventori Pt. Pangan Sehat Sejahtera," *J. Inkofar*, vol. 1, no. 2, pp. 30–40, 2018, doi: 10.46846/jurnalinkofar.v1i2.12.

[23] L. Woolcott, *Understanding Metadata: What is Metadata, and What is it For?* , , vol. 55, no. 7–8. 2017.

[24] J. Junaedy and A. Munir, "Rancang Bangun Sistem Pengelolaan Data Kuliah Kerja Lapang Plus Memanfaatkan Framework Codeigniter Dengan Menggunakan Metode Waterfall," *Ilk. J. Ilm.*, vol. 9, no. 2, pp. 203–210, 2017, doi: 10.33096/ilkom.v9i2.141.203-210.

[25] Khairunnisak, Alvi, and Dony, "Steganografi Gambar dengan Modifikasi Vigenere Chiper Huruf Arab (Hijaiyah) dan Teknik Least Significant Bit (LSB)," pp. 1–8, 2018.

[26] A. Susanto, U. Kahuripan, and K. Indonesia, "Peningkatan Deteksi Steganografi Algoritma Least Significant Bit pada Citra Peningkatan Deteksi Steganografi Algoritma Least Significant Bit pada Citra Grayscale," no. July, pp. 0–8, 2019.

[27] L. M. Jannah, I. Santoso, and Y. Christyono, "Kinerja Steganografi Metode End of File Pada Data Citra Digital," *Transient*, vol. 7, no. 1, p. 34, 2018, doi: 10.14710/transient.7.1.34-39.