Steganography in Digital Marketing Using the Least Significant Bit Method

Bayu Charisma Putra[1], Muhammad Daffa Darmawan[2], Agung Santoso[3], Khairil Anam[4], Dhofirul Yahya[5], M.Farkhan[6]
Informatics Engineering, Faculty of Engineering, Maarif Hasyim Latif University, Sidoarjo

Advances in information and communication technology have significantly increased the accessibility of cyberspace, enabling data exchange to occur anytime and anywhere. However, this convenience also increases the risk of unauthorized access and manipulation of confidential information. Therefore, effective data protection mechanisms are required to maintain information security in digital systems. One technique that can be utilized to enhance data protection is steganography, which hides secret messages within digital media so that their existence is not easily detected. This study aims to implement a steganography technique in a digital catalog system to secure product description data by embedding the information within image files. The system was developed as a web-based application using the PHP programming language, while the message hiding process employed the Least Significant Bit (LSB) method on image media in JPG format. The research involved designing and implementing an embedding and extraction mechanism that allows product data to be hidden and retrieved from digital images. The testing results indicate that the system successfully embeds and extracts text messages without causing visible differences between the original image and the stego image, thus fulfilling the fidelity and recovery criteria. However, the embedded data cannot be maintained when the image undergoes manipulation such as cropping, rotation, or contrast modification, indicating that the robustness criterion is not achieved. Overall, the LSB-based steganography approach is effective for concealing product information in digital catalog images, although further development is required to improve resistance to image manipulation.

Keywords: Digital Catalog, Data Security, Least Significant Bit, Digital Media, Steganography.

Corresponding Author:
Bayu Charisma Putra
Informatics Engineering, Faculty of Engineering, Maarif Hasyim Latif University, Sidoarjo
bayu_charisma@dosen.umaha.ac.id

1. Introduction

The rapid development of information and communication technology has significantly transformed the way digital information is created, stored, and exchanged. Digital media, particularly images, have become one of the most widely used forms of information representation in various fields such as communication, education, marketing, and data documentation. Images function not only as visual representations but also as tools for conveying information, illustrating concepts, and supporting communication in digital environments. As a result, the role of image processing technology has become increasingly important in managing and manipulating digital images to improve their quality and usability [1].

Image processing refers to a form of signal processing in which the input is an image and the output is a modified or enhanced version of that image [2].. Through various computational techniques, image processing enables images to be analyzed, manipulated, or transformed so that the information contained in them can be interpreted more effectively by humans or computer systems. The application of image processing technology has expanded into many areas, including digital communication, multimedia systems, and information security.

Along with the rapid growth of the internet as a global communication network, data exchange between users can now occur quickly and without geographical limitations. Although this development provides significant benefits in terms of accessibility and efficiency, it also increases the risk of unauthorized access, data leakage, and manipulation of confidential information. Therefore, ensuring the security of digital data has become an important concern in modern information systems [3].

One technique that can be used to protect digital information is steganography. Steganography is a method of concealing secret messages within other digital media such as images, audio, video, or text so that the presence of the hidden information is not easily recognized [4]. This technique works by embedding message bits into the pixel components of a digital image, creating a carrier medium known as a cover image. However, certain steganography methods may experience limitations when the carrier media undergo modification or format changes, which can potentially cause the hidden message to be lost.

Among various steganographic techniques, the Least Significant Bit (LSB) method is one of the most widely used approaches for hiding information in digital images. This method embeds secret data by modifying the least significant bits of pixel values, allowing messages to be inserted while maintaining minimal visual changes to the image. As a result, the presence of hidden information is generally imperceptible to the human visual system [5].

Based on these conditions, this study focuses on the development of a steganography application for digital images using the Least Significant Bit method. The research specifically examines the implementation of this technique for securing product information in digital catalog systems by embedding textual data within image files. The objective of this research is to design and implement an image-based steganography system using the LSB method that can support data security and serve as an alternative mechanism for storing product information in digital catalogs.

2. Literature Review

Steganography is the art of secret communication by hiding messages within objects that appear harmless. The existence of a steganographic message is kept secret [6]. This Greek term comes from *Steganos*, meaning covered, and *Graphia*, meaning writing. Steganography is a type of hidden communication that literally means “covered writing.” The message is open and always visible, yet its existence as a secret message is undetected. Another popular description of steganography is *Hidden in Plain Sight*. In contrast, cryptography produces messages that appear random and unreadable, and the existence of the message is usually known [7].

One method of steganography is LSB (Least Significant Bit). LSB is a technique that works by taking the last bits of color values in an image and replacing them with data bits [8]. There are many ways to replace color bits in an image, including adding or subtracting color values or performing AND and OR operations between color bits and data bits. The main objective of LSB is to manipulate the value of a pixel so that data can be hidden within it while minimizing the changes so that they cannot be detected by the human eye [9].

3. Methods

The system design is a description to the author of the system that will be created. Designing the application to be created begins with creating block diagrams, flowcharts and interfaces for the application being created.

Block Diagrams

The block diagram to be designed can be seen in Figure 1.

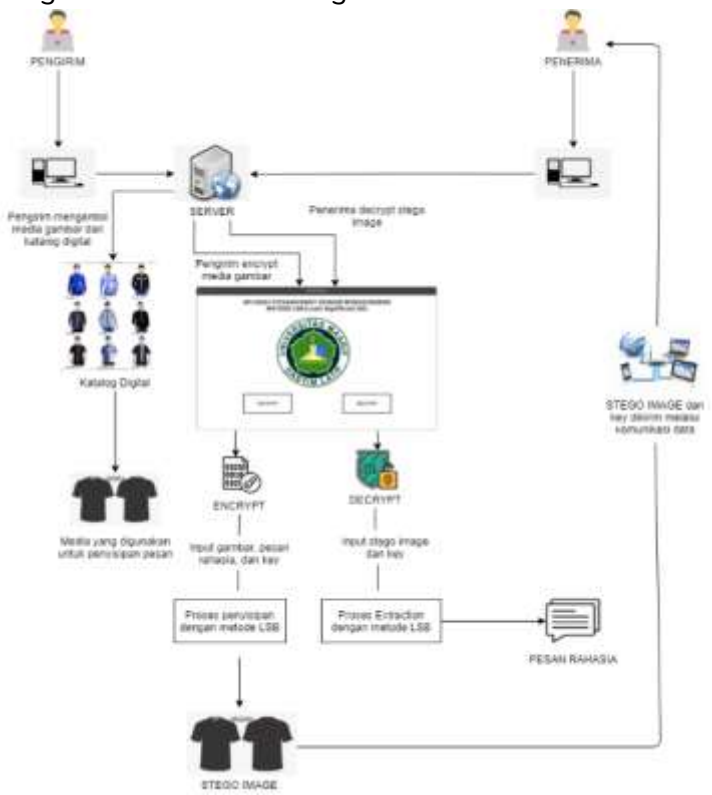

Figure 1. Block diagram

Based on Figure 1 above, the sender logs into the digital catalog website and selects an image to embed a message, then selects the encrypt menu. The sender enters the image and message to be embedded, and the system performs the embedding process using the Least Significant Bit method. The result of the embedding process is a stego image file, or an image with an embedded message. To view the embedded message, the recipient logs into the website and selects the decrypt menu. The recipient inputs the stego image file, and the system performs the extraction process by generating the bits containing the message using the Least Significant Bit method. The result of the decryption process is text embedded into the image.

The following is a flowchart for a steganography application using the Least Significant Bit method, which involves the message embedding and extraction process:

a. Message Embedding Process

   The embedding process involves inserting the message text into the container. The flowchart for the message embedding process can be seen in Figure 2.

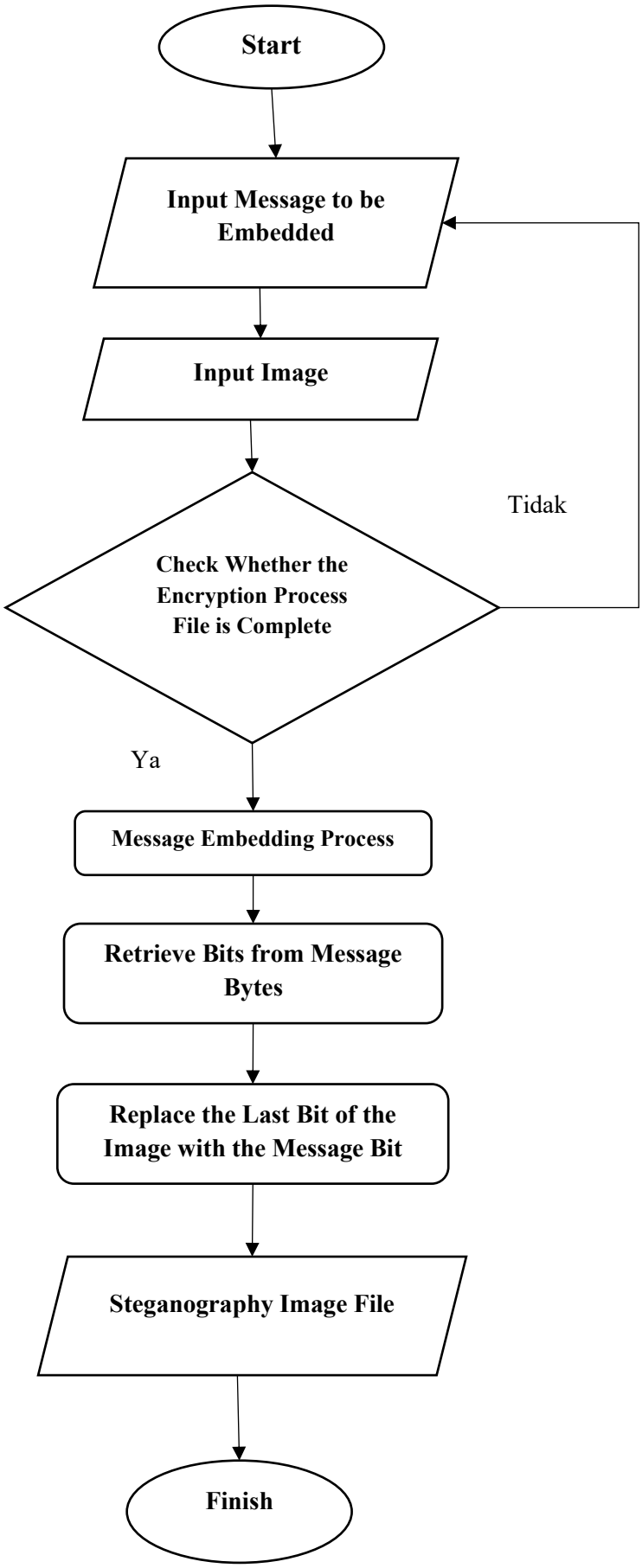

Figure 2. Embedding Process Flowchart

The flowchart in Figure 2 shows the embedding process by entering the message to be embedded into the image, then inserting the image to be used as the message container. After entering the message and selecting the image, the embedding process begins, where the system reads the RGB image file and converts the decimal numbers to binary. For example, the media container that has been changed into binary code is (**00100111) (11101001) (11001000) (00100111) (11001000) (11101001) (11001000) (00100111) (11101001)** and the message to be inserted has previously been changed into a binary number, namely 01001000, then the result of steganography using the LSB method produces **(0010011**0**) (1110100**1**) (1100100**0**) (0010011**0**) (1100100**1**) (1110100**0**) (1100100**0**) (0010011**0**) (11101001).** Replacing the last bit of the image with the message bit will produce a stego image file, or an image with an embedded message.

b. Message Extraction Process

The message extraction process is the process of retrieving the message that has been embedded into the storage medium. A flowchart of the message extraction process can be seen in Figure 3.

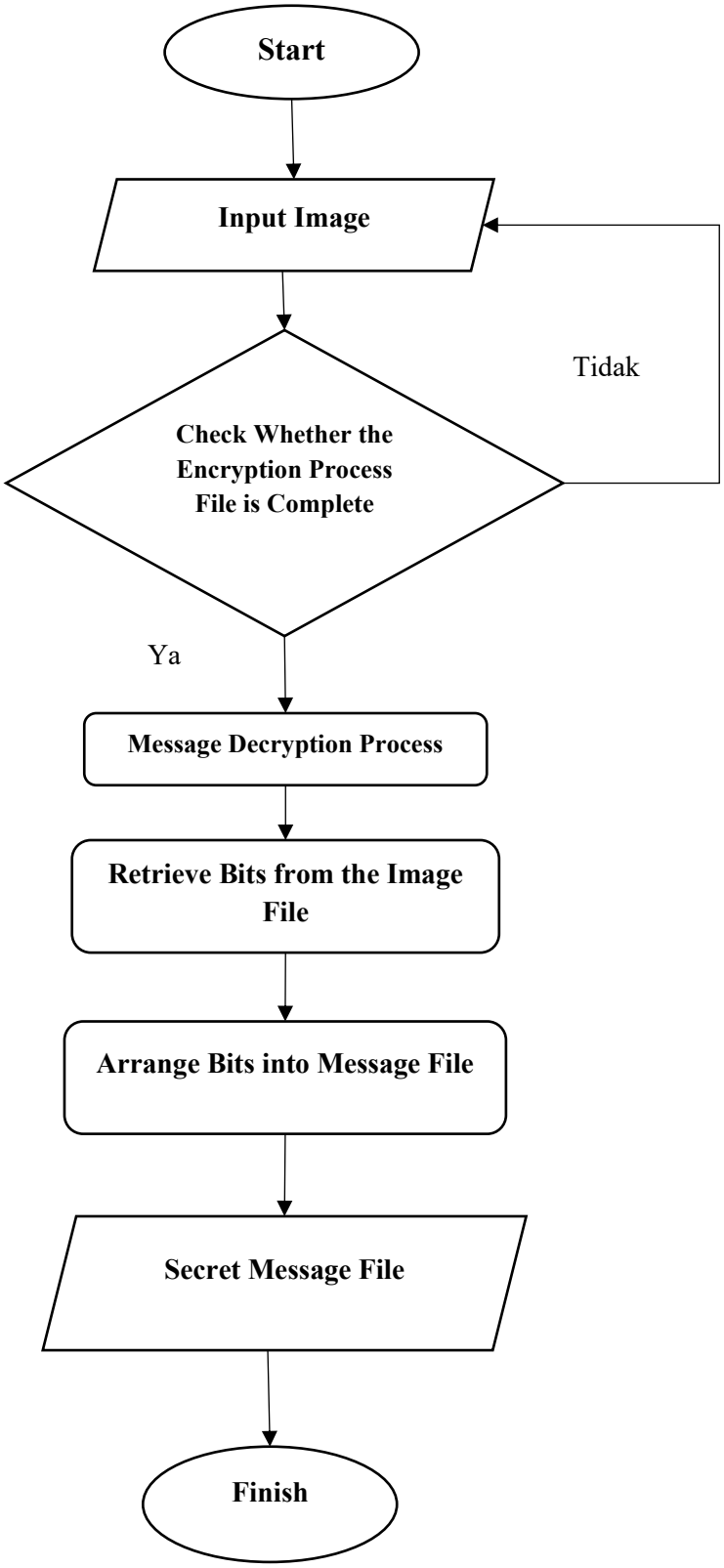

Figure 3. Extraction Process Flowchart

The flowchart in Figure 3 shows that the extraction process is carried out by inputting a stego image file or image media with an embedded message. The initial process is to read the RGB image file and then convert the RGB image into binary. The system then generates the bits contained in the stego image file. Once the

bits containing the message are obtained, the extraction process begins by counting the number of message bytes in the image. The system then reassembles the resulting bits into a message. For example, the binary data obtained from a stego image file is (00100110) (11101001) (11001000) (00100110) (11001001) (11101000) (11001000) (00100110) (11101001), so the rearranged bits become 01001000.

Use Case Diagram

A use case diagram is a model for the behavior of the system to be created. Use case diagrams are used to identify the functions within the system. A use case diagram for the application to be created can be seen in Figure 4.

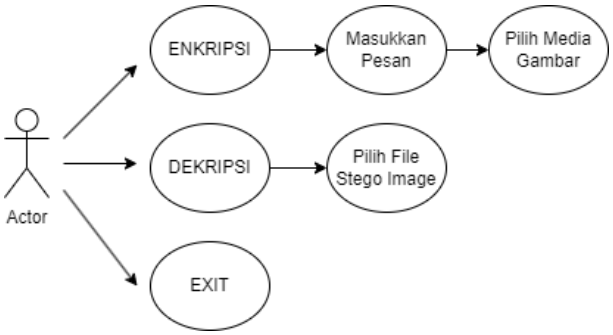

Figure 4. Use Case Diagram

In Figure 4, the user can choose between encryption and decryption. If they choose encryption, they then enter the message to be added and the media to be used. If they choose decryption, they then insert a stego image file or media file with the message added to it.

Activity Diagram

An activity diagram explains the flow of activities in a program being designed, including the initial process, the decisions that occur, and how the system ends.

1. Encryption Activity Diagram
   The encryption activity diagram explains the message insertion process, starting with entering the password, entering the added message, and selecting the media to be used. The encryption activity diagram can be seen in Figure 5.
   According to Figure 5, the user will perform the encryption process, enter the message to be added, and insert the image media to be used. Then, the user presses the Encrypt Now button to begin the insertion process. The result is an image file with the message added.

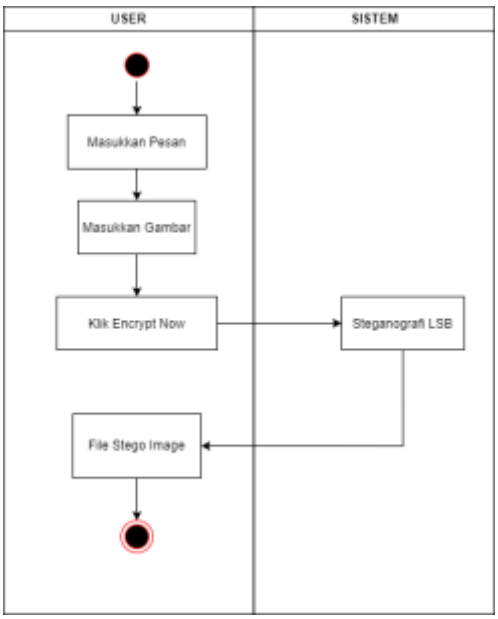

Figure 5. Encryption activity diagram

2. Decryption activity diagram

   The decryption activity diagram explains the process of decrypting an image with a message added. This process begins with entering the password added during the encryption process and inserting the media to which the message has been added. The decryption activity diagram can be seen in Figure 6.

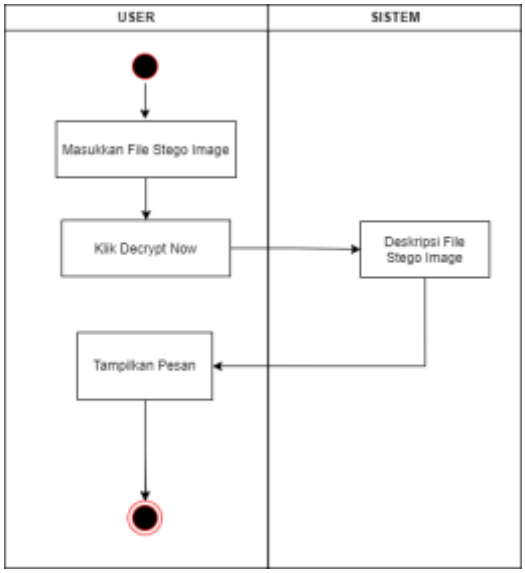

Figure 6. Decryption activity diagram

According to Figure 6, the user will initiate the decryption process. The user will insert the image with the message attached, then press the Decrypt Now button to begin the decryption process for the stego image file. The results of the decryption process will be displayed on the application's main page.

4. Result and Discussion

Digital Catalog Display

When the website is opened, the first thing you see is the digital catalog display. The top section displays the website's name, followed by a selection of products available in the digital catalog. Users can select products from the catalog. The digital catalog display can be seen in Figure 7.

ISSN 2828-9463

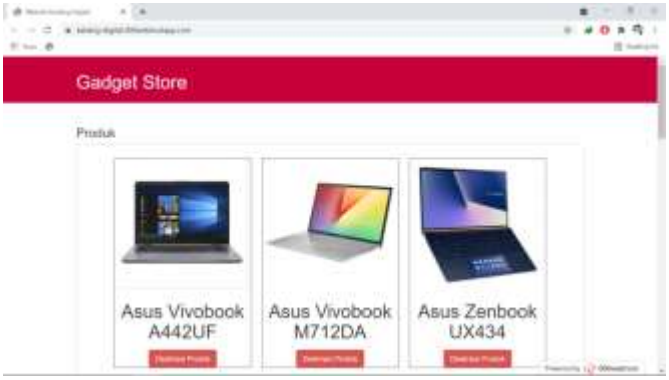

Figure 7. Digital catalog display

Embed Menu Display

The embed menu display is used for encryption and embedding of product data into images. Before starting the process of inserting product data into an image, users must fill out the provided form [10]. The required input includes the text of the product data to be added and the image used as the container. The embed menu display can be seen in Figure 8.

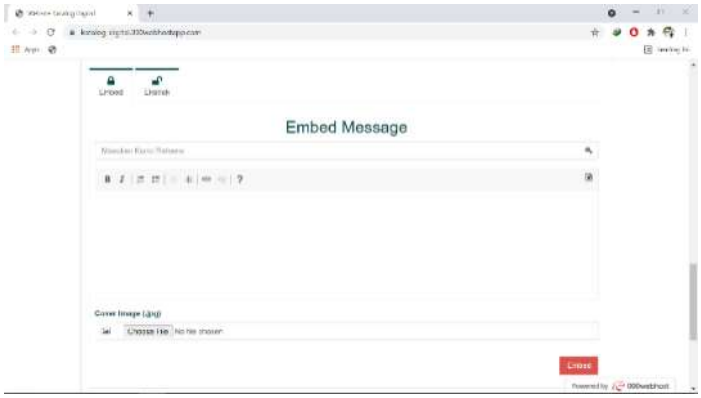

Figure 8. Embed menu display

Extract menu display

The extract menu display is used to re-extract product data that has been embedded into an image, allowing the product data contained in the image to be viewed and read again [11]. The extract menu display can be seen in Figure 9.

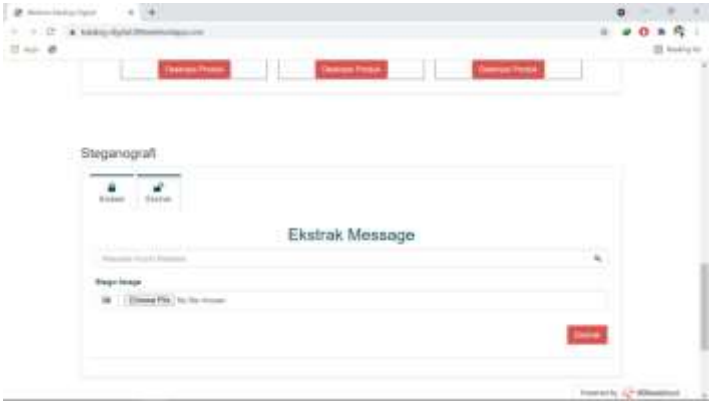

Figure 9. Extract menu display

Product Data Insertion

Product data insertion occurs during the encryption process, where the product data in the digital catalog is inserted into the image media. The results of the product data insertion can be seen in Table 1.

Table 1. Product Data Insertion

Images used	Added product data	Character length
	- Asus Vivobook A442UF - Intel Core i5-8250U Processor (1.60 Ghz, Up to 3.40 Ghz, 6M Cache) - RAM 8GB DDR4 - HDD 1 TB - Nvidia Geforce GT130MX - Layar 14 inch - OS Windows 10	164
	- Acer Aspire 3 - Intel Celeron N4120 Processor - RAM 4GB DDR4 - SSD 256GB - Intel UHD Graphics 600 - Layar 14 inch - OS Windows 10	117
	- Acer Aspire 5 - Intel Core i3-10210U Processor - RAM 4GB - HDD 1TB - Nvidia Geforce MX250 - Layar 14 inch - OS Windows 10	109
	- Lenovo Ideapad C340 - Intel Core i3-8145U Processor - RAM 8GB DDR4 - SSD 512GB - Nvidia GeForce MX250 with 2GB GDDR5 VRAM - Layar 14 inch - OS Windows 10 Home	146
	- Lenovo Legion 5 Pro - AMD Ryzen 7 5800H - RAM 16GB - SSD 512GB - Nvidia GeForce RTX3060 with 6GB GDDR6 VRAM - Layar 15,6 inch - OS Windows 10 Home	134
	- Lenovo Thinkpad X395 - AMD Ryzen 7 Pro Mobile 3700U - RAM 16GB DDR4 - SSD 512GB - AMD Radeon Vega Gfx - Layar 13,3 inch - OS Windows 10	123

Images used	Added product data	Character length
	- Acer Nitro 5 - Intel Core i7-8750H Processor - RAM 8GB DDR4 - SSD 256GB - HDD 1 TB - Nvidia GeForce GTX 1050Ti with 4GB of GDDR5 - Layar 15,6 inch - OS Windows 10 Home	153
	- Asus Vivobook M712DA - AMD Ryzen 7 3700U - RAM 16GB DDR4 - HDD 1 TB - SSD 512GB - AMD Radeon RX 540X - Layar 17 inch - OS Windows 10 Pro	122
	- Asus Zenbook UX434 - Intel Core i5-10210U Processor - RAM 8GB - SSD 1 TB - Nvidia GeForce MX250 with 2GB GDDR5 VRAM - Layar 14 inch - OS Windows 10	135
	- Axioo Mybook 14 - Intel Celeron N3350 dual-core processor (1.1GHz Up to 2,4GHz) - RAM 3GB - HDD 500GB - Intel HD Graphics 500 - Layar 14 inch - OS Windows 10	145

Steganography Criteria Testing

This steganography criteria testing aims to determine whether the system meets the steganography criteria established by steganography theory. The tests performed on this system include fidelity, robustness, and recovery [12].

Fidelity

The fidelity criterion means that the addition of a message to the carrier media does not significantly change the message [13]. The results of the embedding process remain visible, and when viewed by the human eye, the image with the message inserted appears identical to the image used as the carrier media. The test results can be seen in Table 2.

Table 2. Fidelity Criteria Testing

Image Used	Image with Embedded Message	Respondents
 Format: JPGDimensions: 500×500 pixels	 Format: JPGDimensions: 500×500 pixels	10 people did not see any difference
 Format: JPGDimensions: 500×500 pixels	 Format: JPGDimensions: 500×500 pixels	10 people did not see any difference
 Format: JPGDimensions: 195×195 pixels	 Format: JPGDimensions: 195×195 pixels	10 people did not see any difference
 Format: JPGDimensions: 220×220 pixels	 Format: JPGDimensions: 220×220 pixels	10 people did not see any difference
 Format: JPGDimensions: 800×800 pixels	 Format: JPGDimensions: 800×800 pixels	10 people did not see any difference
 Format: JPGDimensions: 800×515 pixels	 Format: JPGDimensions: 800×515 pixels	10 people did not see any difference
 Format: JPGDimensions: 500×500 pixels	 Format: JPGDimensions: 500×500 pixels	10 people did not see any difference
 Format: JPGDimensions: 1000×1000 pixels	 Format: JPGDimensions: 1000×1000 pixels	10 people did not see any difference

Image Used	Image with Embedded Message	Respondents
 Format: JPGDimensions: 360×360 pixels	 Format: JPGDimensions: 360×360 pixels	10 people did not see any difference
 Format: JPGDimensions: 280×280 pixels	 Format: JPGDimensions: 280×280 pixels	10 people did not see any difference

Based on the test results in Table 2, there is no visible difference between the original image and the image with the message added. Therefore, the test results indicate that the fidelity criteria have been met.

Robustness

The robustness criterion refers to the ability of the added message to withstand manipulation operations such as changing the contrast, sharpening the image, adding noise, enlarging the image, and cropping the image [14]. The results of the robustness criterion test can be seen in Table 3.

Table 3. Robustness Criteria Test

Manipulated Image Result	Type of Manipulation	Result
	Contrast Adjustment	Failed
	Image Sharpening	Failed
	Image Rotation	Failed
	Noise Addition	Failed
	Image Pixel Enlargement	Failed
	Image Cropping	Failed

Based on the test results in Table 3, images that had been added with messages and had undergone several image manipulation experiments could not be extracted. Therefore, this system does not meet the robustness criteria [15].

Recovery

The recovery criterion refers to the ability to extract the message added to the storage medium, from the characters of each word to the length of the added message. The results of the recovery criterion test are shown in Table 4.

Table 4. Recovery Criteria Test

Image Used	Message After Extraction	Result
	**Asus Vivobook A442UF**- Intel Core i5-8250U Processor (1.60 GHz, Up to 3.40 GHz, 6M Cache)- 8GB DDR4 RAM- 1TB HDD- Nvidia GeForce GT130MX- 14-inch Display- Windows 10 OS	Successful
	**Acer Aspire 3**- Intel Celeron N4120 Processor- 4GB DDR4 RAM- 256GB SSD- Intel UHD Graphics 600- 14-inch Display- Windows 10 OS	Successful
	**Asus Zenbook UX434**- Intel Core i5-10210U Processor- 8GB RAM- 1TB SSD- Nvidia GeForce MX250 with 2GB GDDR5 VRAM- 14-inch Display- Windows 10 OS	Successful
	**Lenovo Ideapad C340**- Intel Core i3-8145U Processor- 8GB DDR4 RAM- 512GB SSD- Nvidia GeForce MX250 with 2GB GDDR5 VRAM- 14-inch Display- Windows 10 Home OS	Successful
	**Lenovo Legion 5 Pro**- AMD Ryzen 7 5800H- 16GB RAM- 512GB SSD- Nvidia GeForce RTX 3060 with 6GB GDDR6 VRAM- 15.6-inch Display- Windows 10 Home OS	Successful
	**Lenovo ThinkPad X395**- AMD Ryzen 7 Pro Mobile 3700U- 16GB DDR4 RAM- 512GB SSD- AMD Radeon Vega Graphics- 13.3-inch Display- Windows 10 OS	Successful
	**Acer Nitro 5**- Intel Core i7-8750H Processor- 8GB DDR4 RAM- 256GB SSD- 1TB HDD- Nvidia GeForce GTX 1050Ti with 4GB GDDR5- 15.6-inch Display- Windows 10 Home OS	Successful
	**Asus Vivobook M712DA**- AMD Ryzen 7 3700U- 16GB DDR4 RAM- 1TB HDD- 512GB SSD- AMD Radeon RX 540X- 17-inch Display- Windows 10 Pro OS	Successful
	**Acer Aspire 5**- Intel Core i3-10210U Processor- 4GB RAM- 1TB HDD- Nvidia GeForce MX250- 14-inch Display- Windows 10 OS	Successful
	**Axioo MyBook 14**- Intel Celeron N3350 Dual-Core Processor (1.1 GHz up to 2.4 GHz)- 3GB RAM- 500GB HDD- Intel HD Graphics 500- 14-inch Display- Windows 10 OS	Successful

Based on the test results in Table 4 different images, each with a message added, could be extracted. The results of the extraction process were also the same as before the insertion process. Therefore, it can be concluded that the recovery criteria have been met [16].

Maximum Character Capacity in Images

To determine the maximum number of characters that can be added to an image, tests were conducted using several images of different sizes [17]. After several tests, the maximum number of characters that can be added was determined based on the image size. The test results can be seen in Table 5.

Table 5. Maximum Character Capacity in Images

Image Size (pixels)	Maximum Characters That Can Be Added
195x195	4.753
200x200	5.000
220x220	6.050
280x280	9.800
360x360	16.200
500x500	31.250
800x515	51.500
800x800	80.000
900x900	101.250
1000x1000	125.000

In this study, the image used is an RGB image which has 3 bytes in each pixel so that the image size is multiplied by 3. From the test results in Table 5, the calculation of the maximum character size is obtained using the following formula:

$$Max\ Character = \frac{\text{Image length x image width}}{8} \quad (1)$$

The image size is divided by 8 because 1 character consists of 8 bits hidden in every 8 image pixels.

To strengthen the validity of the results, the findings of this study were compared with several previous studies related to steganography and the Least Significant Bit (LSB) method [18]. The comparison shows that the use of LSB in image-based steganography can effectively embed secret messages into digital images without causing significant visual changes [19]. This finding is consistent with previous studies which explain that LSB modifies only the least significant bits of pixel values, allowing hidden data to be inserted while maintaining the visual quality of the image so that the difference between the original image and the stego image cannot be easily detected by the human eye [20], [21].

However, differences were observed in terms of robustness. In this research, the hidden messages could not be successfully extracted after the image underwent several manipulation processes such as contrast adjustment, sharpening, rotation, noise addition, enlargement, and cropping. Similar results have also been reported in previous studies which state that the basic LSB method has limitations in resisting image manipulation because the embedded data is directly stored in the pixel bits of the carrier image [22], [23]. As a result, even small modifications to the image may alter or remove the embedded message bits.

The phenomena observed from the analysis results indicate that the LSB method performs well in maintaining fidelity and recovery criteria but performs poorly in the robustness criterion. This pattern is consistent with the theoretical characteristics of LSB steganography, where the main objective is to minimize visual distortion in the carrier image rather than to provide strong resistance to image processing operations. Therefore, although the hidden message remains visually undetectable, the embedded data becomes vulnerable when the image undergoes modification or transformation[24] [25].

Based on the findings and interpretations obtained, it can be concluded that the implementation of LSB-based steganography in this study is effective for embedding and extracting textual information within digital images under normal conditions [26] [27]. The system successfully preserves image quality and ensures that the hidden message can be recovered accurately [28]. However, the limitation in robustness indicates that further development is required, such as combining LSB with encryption or other steganographic techniques to improve resistance against image manipulation [29]. These preliminary

conclusions provide an overview of the strengths and limitations of the proposed system and can serve as a basis for further validation and development in future research [30].

5. Conclusion

The steganographic performance using the least significant bit method is quite good and meets the fidelity and recovery criteria. However, it is not resistant to image manipulation operations and therefore does not meet the robustness criteria. The larger the size of the media container, the more characters can be added. The system design created by the author can be further developed for better performance, such as use with PNG or BMP image formats and further development on mobile-based systems such as Android and iOS.

6. References

[1] Ade Dwi Harisna. (2009). *Image Processing*. https://ndoware.com/image-processing.html.
[2] Akhsanu Ridlo, I. (2017). Panduan pembuatan flowchart. *Fakultas Kesehatan Masyarakat*.
[3] Abbas, W. (2013). Analisa Kepuasan Mahasiswa Terhadap Website Universitas Negeri Yogyakarta (Uny). *Manajemen*, 1–6.
[4] Aboalsamh, H., Mathkour, H., Dokheekh, S., Mursi, M., & Ghazyassassa. (2008). An improved steganalysis approach for breaking the F5 algorithm. *WSEAS Transactions on Computers*, *7*(9), 1447–1456.
[5] Aditya, D., Panchadria, P. A., & Setiyanto, R. (2017). *Application Development Method Steganography Least Significant Bit (Lsb) With Combination Cryptographic Algorithm Rc4 and Base64 Based on Php Pengembangan Aplikasi Steganografi Metode Least Significant Bit (Lsb) Dengan Kombinasi Algoritma Kriptografi* . *April*, 350–359.
[6] Apriyansyah, Unik, M., & Mukhtar, H. (2020). *Implementasi Sistem Keamanan Pesan Text Dengan Teknik Steganografi Menggunakan Metode Least Significant Bit (LSB)*. *1*(1), 8–12.
[7] Atoum, M. S., Suleiman, M., Rababaa, A., Ibrahim, S., & Ahmed, A. (2011). A Steganography Method Based on Hiding secrete data in MPEG / Audio Layer III. *Journal of Computer Science*, *11*(5), 184–188.
[8] Bandyopadhyay, S. K., Bhattacharyya, D., Ganguly, D., Mukherjee, S., & Das, P. (n.d.). *A Tutorial Review on Steganography*.
[9] Baritha Begum, M., & Venkataramani, Y. (2012). LSB based audio steganography based on text compression. *Procedia Engineering*, *30*(2011), 703–710. https://doi.org/10.1016/j.proeng.2012.01.917
[10] Cox, I., Miller, M., Bloom, J., & Fridrich, J. (2008). *Digital Watermarking and Steganography 2nd Ed*. Morgan Kauffman., MA.
[11] Dewi, L. P., Indahyanti, U., & S, Y. H. (2017). Pemodelan Proses Bisnis Menggunakan Activity Diagram Uml Dan Bpmn (Studi Kasus Frs Online). *Informatika*, 1–9.
[12] Dutta, P., Bhattacharyya, D., & Kim, T. (2009). Data Hiding in Audio Signal : A Review. *International Journal of Database Theory and Application*, *2*(2), 1–8. http://www.sersc.org/journals/IJDTA/vol2_no2/1.pdf
[13] Edy Winarno ST, M.Eng, Ali Zaki, SmitDev Community. (2014). Pemrograman Web Berbasis HTML5, PHP, dan Javascript. PT. Elex Media Komputindo, Jakarta.
[14] Hariri, M., Karimi, R., & Nosrati, M. (2011). An introduction to steganography methods. *World Applied Programming*, *13*, 191–195.
[15] Hendini, A. (2016). Pemodelan Uml Sistem Informasi Monitoring Penjualan Dan Stok Barang. *Jurnal Khatulistiwa Informatika*, *2*(9), 107–116. https://doi.org/10.1017/CBO9781107415324.004.

[16] Hidayat Rahmat. (2010). Cara Praktis Membangun Website Gratis. PT. Elex Media Komputindo, Jakarta.
[17] Kaur, N., & Behal, S. (2014). A Survey on various types of Steganography and Analysis of Hiding Techniques. *International Journal of Engineering Trends and Technology*, *11*(8), 388–392. https://doi.org/10.14445/22315381/ijett-v11p276
[18] Kipper, G. (n.d.). (2004). *Investigator's Guide to Steganography, Florida: CRC* Press LLC.
[19] Munir, R. (2015). Bahan Kuliah Kriptografi IF4020. diakses dari https://informatika/stei.itb.ac.id/~rinaldi.munir/Kriptografi/2020-2021/kripto2021.htm
[20] Reddy Karunakar, D. & Dr. T. Satya Savithri. (2013). Area Efficient Rapid Signal Acquistionscheme For High Doppler DSSS Signals. International Journal of Computer Networks & Communications (IJCNC) Vol.5, No.2, March 2013.
[21] Saefullah, A., Himawan, & Agani, N. (2012). Aplikasi Steganografi Untuk Menyembunyikan Teks Dalam Media Image Dengan Menggunakan Metode LSB. *Seminar Nasional Teknologi Informasi & Komunikasi Terapan 2012 (Semantik 2012)*, *2012*(Semantik), 151–157.
[22] Sidik, A., Hakim, Z., & Permana, E. A. (2014). Analisis Dan Implementasi Teknik Steganografi Sebagai Fasilitas Pengamanan Proses Pengiriman File Secara Online. *Jurnal Sisfotek Global*, *4*(1), 1–4.
[23] Singh, H., Singh, P.K., & Saroha, K. (2009). A Survey on Text Based Steganography. *Proceedings, 3*(3), 332-335. http://bvicam.ac.in/news/INDIACom/2009/Proceedings/pdfs/papers/119.pdf
[24] Sunyoto Andi. (2007). Pemrograman Database dengan Visual Basic dan Microsoft SQL 2000. Yogyakarta: Andi Offset.
[25] Syawal, M. F., Fikriansyah, D. C., & Agani, N. (2016). Implementasi Teknik Steganografi Menggunakan Algoritma Vigenere Cipher Dan Metode LSB. *Jurnal TICOM*, *4*(3), 91–99.
[26] Wiryawan, I Gede., Sariyasa., & I Gede Aris Gunadi. (2019). Steganografi Berdasarkan Metode Least Significant Bit (LSB). *Jurnal Ilmu Komputer Indonesia (JIKI)*, *1*, 34–40.
[27] B. C. Putra and M. I. Khoir, "DETEKSI HELM PADA PENGGUNA SEPEDA MOTOR MENGGUNAKAN METODE YOLO SECARA REALTIME," SPIRIT: Journal Of Computing and Cybernetic System, vol. 17, no. 2, pp. 148-157, 2025.
[28] B. C. Putra and Y. N. Afifah, "Gaussian Mixture Model untuk Penghitungan Tingkat Kebersihan Sungai Berbasis Pengolahan Citra," Teknika: Engineering and Sains Journal, vol. 2, no. 1, pp. 53–58, 2018.
[29] B. C. Putra, B. Setiyono, D. R. Sulistyaningrum, Soetrisno, and I. Mukhlash, "Moving Vehicle Classification Using Pixel Quantity Based on Gaussian Mixture Models," Proc. 2018 3rd Int. Conf. Comput. Commun. Syst. (ICCCS), pp. 27–30, 2018, doi: 10.1109/CCOMS.2018.8463218.
[30] B. C. Putra and R. A. J. Firdaus, "Storm Detection Application on Satellite Image Using the Hough Circle Method Based on Digital Image Processing," J. Inf. Comput. Technol. Educ. (JICTE), vol. 4, no. 2, pp. 1–8, 10.21070/jicte.v4i2.1018.