International Journal of Recent Development in Engineering and Technology
Website: www.ijrdet.com (ISSN 2347-6435(Online) Volume 14, Issue 12, December 2025)

"Steganography Tool"

Parv Rawat[1], Varun Rajput[2], Anurag Porwal[3], Aman Yadav[4], Prof. Sameeksha Rahangdale[5]

[1,2,3,4]*Students,* [5]*Assistant Professor, Department of Electronics and Communication Engineering, Lakshmi Narain College of Technology Excellence, Bhopal (M.P.)*

Abstract-- **In today's digital age, where data breaches, surveillance, and cyber threats are increasingly prevalent, the protection of sensitive information has become more critical than ever. While encryption techniques are widely used to secure data, they can also attract attention due to the visible presence of encrypted content. In contrast, steganography provides a subtle yet powerful approach to secure communication by concealing the existence of the message itself within a non- suspicious medium—commonly an image.**

This paper introduces sabrowser-based Image Steganography Tool, developed using HTML, CSS, and Java Script, that allows users to hide and retrieve textual messages within image files. The application is built as a fully client-side tool, ensuring that no data is ever transmitted to a server. This not only enhances user privacy but also simplifies deployment, as it can run directly in any modern web browser without requiring any external software installations or backend support.

The Image Steganography Tool bridges the gap between theoretical security concepts and real-world applications by offering a secure, private, and user-friendly platform for concealing information in images. It's entirely client-side nature ensures data privacy, making it a valuable tool for individuals and institutions concerned with digital information security.

Keywords: - **Image Steganography Tool, HTML, CSS, and JavaScript.**

I. INTRODUCTION

Introduction to the Steganography-

Steganography is the practice of hiding information within other non-secret media to avoid detection. This paper implements an image-based steganography tool where secret messages can be embedded within images using Java Script. It offers a simple yet functional frontend for users to upload an image, enter a message, encode it, and download the modified image.

Paper Formation

This paper falls under the category of Web-based Security Tools with a focus on Information Hiding and Data Security.

Motivation

In the age of digital communication, safeguarding sensitive data is crucial. Steganography offers an alternative to encryption by hiding the existence of the data itself. This manuscript was inspired by the need for a light weight, client-side tool for hiding messages in images, useful in secure communications and digital water marking.

Goals and Objectives

Develop a web-based application for image steganography. Enable message encoding and decoding without external tools or servers. Ensure ease of use with an intuitive UI. Support client-side execution to maintain message confidentiality**.**

Scope and Applications

Secure Messaging: Private communication that is visually undetectable.

Digital Watermarking: Embed identifiers in digital images.

Educational Use: Demonstrate how steganography works in real-time.

II. PROBLEM FORMULATION

Traditional communication systems are vulnerable to interception. Even encrypted messages can raise suspicion. There is a need for a tool that enables secret, undetectable communication by embedding text within images in a secure and user-friendly manner.

III. LITERATURE SURVEY

1. Wayner, P. (2009). Disappearing Cryptography: Information Hiding: Steganography & Watermarking. Morgan Kaufmann**.**

- A foundational book on information hiding techniques.
- Explains both steganography and digital watermarking, with emphasis on cryptographic principles.
- Useful for understanding the theoretical background and evolution of steganography methods**.**

2. Johnson, N. F., & Jajodia, S. (1998). Exploring Steganography: Seeing the Unseen. IEEE Computer, 31(2), 26–34.

- A classic IEEE paper introducing steganography concepts.
- Discusses how hidden data can be detected and analyzed.
- Important for understanding steganalysis (the detection of hidden information) and the vulnerabilities of basic methods like LSB**.**

3. Katzenbeisser, S., & Petitcolas, F. A. P. (2000). Information Hiding Techniques for Steganography and Digital Watermarking. Artech House.

- A comprehensive text covering various hiding techniques.
- Explains both practical algorithms and security aspects.
- Helps compare traditional methods with modern approaches, giving context for your project's design choices.

4. Mozilla Developer Network (MDN) – https://developer.mozilla.org

- Official documentation for web technologies.
- Essential for implementing your browser-based steganography tool.
- Provides reliable references for JavaScript, HTML5, and CSS.

5. Documentation for JavaScript functions, HTML5 Canvas API, and FileReader API

- Technical references for the actual coding part of your project.
- Canvas API → lets you manipulate image pixels directly in the browser.
- File Reader API → Allows reading image files locally without server upload, ensuring client-side privacy.
- These are the backbone of your research implementation.

6. W3Schools Online Web Tutorials – https://www.w3schools.com

- Beginner-friendly tutorials for web technologies.
- Useful for quick syntax checks and examples when coding.
- Complements MDN by offering simplified explanations and sample code snippets.

7. HTML, CSS, and JavaScript Guides and Examples

- General references for building the user interface of your tool.
- Ensures your project has a clean, responsive design.
- Important for making the tool accessible and user-friendly.

8. Stack Overflow – https://stackoverflow.com

- Community-driven Q&A platform.
- Helps solve specific coding challenges during implementation.
- Valuable for debugging errors and learning from other developers' solutions.

9. Solutions to specific coding challenges and community discussions

- Refers to practical problem-solving threads on Stack Overflow and similar forums.
- Provides real-world fixes for issues like pixel manipulation, encryption integration, or browser compatibility.
- Enhances the robustness of your project.

10. GitHub repositories and gists related to steganography implementations

- Open-source code examples and projects.
- Useful for inspiration and benchmarking your own implementation.
- Helps you learn best practices and avoid common pitfalls in steganography coding**.**

11. Gonzalez, R. C., & Woods, R. E. (2008). Digital Image Processing (3rd ed.). Pearson Education.

- A standard textbook in image processing.
- Explains fundamentals like pixel representation, colour models, and transformations.
- Provides the mathematical foundation for techniques like LSB steganography.
- Essential for understanding how image modifications affect hidden data.

IV. Working Principle

This part describes the architecture and design methodologies used in the development of the Steganography Tool. It includes system structure, data flow, and visual representations such as diagrams to aid understanding of the overall design.

Block Diagram/System Architecture

The system architecture is divided into three main components:

```
+---------------------------------------------- +

|User Interface|
|(HTML/CSS Frontend) |

+---------+ ------------------------------------ +

| v
+---------------------------------------------- +

|Control Logic                                  |
|(JavaScript-based) |

+---------+ ------------------------------------ +

|

+---------+ ------------------------------------ +

|Image Processor|

| (Canvas API -Pixel|

|level manipulation) |

+----------------------------------------------- +
```

Explanation:

- *User Interface:* Allows the user to upload images, enter secret messages, encode/decode, and download results.
- *Control Logic:* Handles encoding/decoding logic using LSB steganography.
- *Image Processor:* Interacts with the HTML5 <canvas> to manipulate image pixel data.

V. Methodology

Development Methodology

The **Agile Development Model** was loosely followed during this project. Agile allowed iterative development, testing, and enhancement of features based on observations and test cases.

Phases Followed:

1. *Requirement Gathering:*
   - Identified the need for a light weight, client-side steganography tool.
   - Determined that only image-based steganography and text messages would be in scope.
2. *Design Phase:*
   - Designed a simple and intuitive UI with HTML and CSS.
   - Created system architecture and planned the logical flow of the application.
   - Finalized the use of Java Script for client-side logic.
3. *Implementation Phase:*
   - Built frontend structure with HTML and styled it using CSS.
   - Implemented the encoding and decoding logic using LSB steganography with JavaScript.
   - Used the <canvas> API for pixel manipulation and rendering.
4. *Testing Phase:*
   - Conducted testing with different image sizes and message lengths.
   - Verified accuracy of encoding/decoding and ensured the image quality remained visually unaffected.
   - Tested across multiple web browsers (Chrome, Firefox, Edge).
5. *Deployment:*
   - As a browser-based tool, deployment simply involves sharing the HTML, CSS and JS files.
   - No server or hosting is required. Users can run it offline.

VI. Implementation, Testing, And Maintenance

a. Description of Frontend Code (Client Side)

The frontend is built using HTML, CSS and JavaScript. The main structure is defined in index.html, which includes:

i. **File Upload** input (<input type="file">) for selecting an image.
ii. **Text Area** to input the secret message.
iii. **Buttons** for encoding, decoding, and downloading the final image.
iv. **Canvas** element used to manipulate image data on the client side.

Styling is managed through styles.css:

v. The UI is visually appealing with dark-mode aesthetics and responsive design.
vi. Button shave hover effects for better UX.
vii. Input are as are styled for clarity and us ability.

b. Description of Admin Code (Administration Side)

This is a client -only application; there is no separate admin panel or backend infrastructure. All encoding, decoding, and processing is done locally in the browser, ensuring complete user privacy.

c. Description of Backend Code (Database)

There is **no data base** used in this project. The image and message data extemporarily held in memory using the **Canvas API** during the encoding and decoding process. No data is stored or transmitted, maintaining local confidentiality.

There is one more thing to be noticed from this circuits Schematics. The input pins of the module work inversely. As we can see the relay will be activated when the input pin will be LOW because in that way the current will be able to flow from the VCC to the input pin which is low or ground, and the LED will light up and active the relay. When the input pin will be HIGH there will be no current flow, so the LED will not light up and the relay will not be activated.

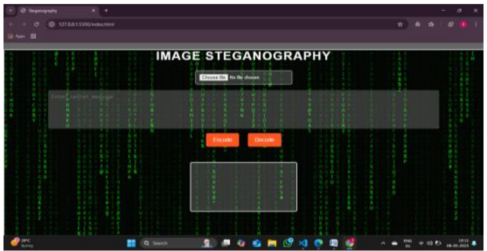
VII. Results

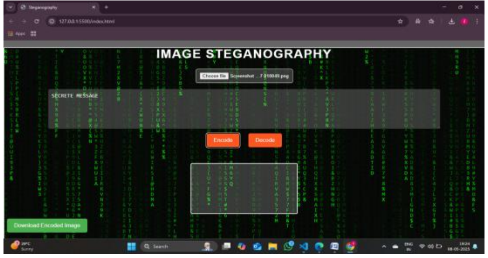
User Interface Representation

The user interface of the Steganography Tool is designed for simplicity and ease of use. It includes:

i. A header titled" Image Steganography"
ii. File input to upload the image
iii. Text area to enter the secret message
iv. Buttons: Encode, Decode and Download Image
v. A canvas area for previewing the image

This clean and intuitive interface allow seven on-technical users to perform steganography operations easily.





1. *Initial Screen:* User is prompted to upload an image and enter a message.
2. *After Upload:* Selected image appears on the canvas.
3. *After Encoding:* Message is hidden inside the image and the down load button becomes visible.
4. *After Decoding:* An alert box displays the decoded message.

VIII. Conclusion

The Image Steganography web application successfully demonstrates the ability to hide and retrieve text messages within image files using basic LSB (Least Significant Bit) encoding. The project fulfills the primary objective of providing a user-friendly, client-side tool that performs steganographic encoding without the need for server-side processing, making it lightweight and secure.

The system:

- Accepts user-uploaded images.
- Encodes text messages into the image's pixel data.
- Allows users to decode and view the hidden message.
- Supports down loading the modified image with embedded content.

This application show cases a practical use of steganography principles for secure communication and can be used for educational, experimental, and low-scale secure messaging purposes.

REFERENCES

[1] Wayner, P. (2009). Disappearing Cryptography: Information Hiding: Steganography & Watermarking. Morgan Kaufmann.

[2] Johnson, N.F., & Jajodia, S. (1998). Exploring Steganography: Seeing the Unseen. IEEE Computer, 31(2), 26–34.

[3] Katzenbeisser, S., & Petitcolas, F.A.P. (2000). Information Hiding Techniques for Steganography and Digital Watermarking. Artech House.

[4] MozillaDeveloper Network(MDN).https://developer.mozilla.org

[5] Documentation for Java Script functions ,HTML5 Canvas API ,and File Reader API.

[6] W3 Schools Online Web Tutorials.https://www.w3schools.com HTML, CSS and Java Script guides and examples.

1. W3 Schools Online Web Tutorials .https://www.w3schools.com
2. HTML, CSS and Java Script guides and examples.
3. StackOverflow. https://stackoverflow.com