Steganography: Secure Data Hiding in Images Using LSB Encoding for Message Encryption and Decryption

A Course Completion Report in part

fulfilment of the degree

Bachelor of Technology

In

Computer Science & Artificial Intelligence

By

Roll. No : 2203A51813 **Name**: NADIPELLY KARTHIKEYA(TEAM LEAD)

Roll. No : 2203A51544 **Name**: BASWARAJU HARINI HRUDAYA

Roll. No : 2203A51452 **Name**: THALLAPELLI SADHVIK

Roll. No : 2203A51543 **Name**: BASABATHINI RAVI TEJA

Roll. No : 2203A51412 **Name:** ETOORI RUPA SREE

UNDER THE GUIDENCE OF

DR. M. RANJEETH KUMAR

Submitted to

SCHOOLOFCOMPUTERSCIENCE &ARTIFICIAL INTELLIGENCE SR UNIVERSITY, ANANTHASAGAR, WARANGAL

March, 2025.

SCHOOL OF COMPUTERSCIENCE & ARTIFICIAL INTELLIGENCE

CERTIFICATE

This is to certify that the **Course Project** Report entitled **"Steganography: Secure Data Hiding in Images Using LSB Encoding for Message Encryption and Decryption"** using **PYTHON**. The work carried out by the students **N. Karthikeya, B. Harini Hrudaya, B. Ravi Teja, T.Sadhvik, E.RupaSree** bearing RollNo(s) 2203A51813 ,2203A51544,2203A51543,2203A51452,2203A51412 during the academic year 2024-25 in partial fulfillment of the award of the degree of ***Bachelor of Technology*** in **Computer Science & Engineering** by the SR University , Anantasagar

Course Faculty **Head of the Department**

TABLE OF CONTENTS

TOPIC **PAGE NO(s)**

Caesar Cipher Encryption and Decryption: A Web-Based Implementation

1. Abstract

This project introduces a comprehensive and interactive web-based application that demonstrates the functioning of the Caesar Cipher encryption and decryption algorithm through the use of core web technologies—namely HTML, CSS, and JavaScript. The Caesar Cipher is a classical substitution cipher technique in which each letter of the plaintext is systematically shifted by a fixed number of positions along the alphabet, either forward or backward, to produce the corresponding ciphertext. Although simple in nature, this cipher holds significant historical value and provides foundational insights into the principles of cryptography.

The developed application provides users with an intuitive interface that facilitates the input of either plaintext (for encryption) or ciphertext (for decryption). Users are given full control over key parameters such as the shift value, modulo length (alphabet size), letter casing (maintain, lowercase, or uppercase), and the handling of non-alphabetic characters (such as punctuation, symbols, and numbers). These customizable features enable a wide range of experimental scenarios, making the tool highly versatile for academic demonstrations or self-paced cryptography practice.

Importantly, the application is designed to operate seamlessly on all modern web browsers without requiring any external frameworks or libraries, ensuring ease of access and deployment. Its primary aim is to enhance the learning experience for students and enthusiasts by visualizing how basic cryptographic transformations work, ultimately building a strong conceptual foundation for exploring more advanced encryption techniques.

2.Introduction

The Caesar Cipher stands as one of the most foundational and historically significant encryption techniques in the field of classical cryptography. Named after the Roman general and statesman Julius Caesar, this cipher was famously employed to safeguard sensitive military communications. Its method is strikingly straightforward: each

letter in the plaintext is shifted a fixed number of positions along the alphabet, wrapping around from 'Z' to 'A' as needed. For example, with a shift of 3, the letter 'A' becomes 'D,' 'B' becomes 'E,' and so on.

Despite its vulnerability to brute-force attacks due to the limited number of possible shifts, the Caesar cipher remains an essential educational tool for introducing the core principles of encryption, such as substitution, modular arithmetic, and keybased transformations. It exemplifies how coded communication can be both created and broken using systematic logic.

This project seeks to bring the Caesar cipher to life through a web-based platform that utilizes HTML, CSS, and JavaScript. By providing an interactive environment where users can input messages, adjust cipher parameters, and instantly observe encrypted or decrypted results, the application transforms abstract cryptographic theory into a tangible learning experience. Its simplicity makes it ideal for beginners, while its customizable features allow for deeper exploration into the dynamics of classical encryption.

3.Problem Statement

In today's digital era, encryption plays a vital role in securing data transmission and protecting user privacy. While modern encryption algorithms are highly complex and secure, understanding their foundational principles is often challenging for beginners. Classical encryption methods such as the Caesar Cipher offer a simple yet effective way to introduce core cryptographic concepts such as substitution, keybased encryption, and modular arithmetic.

However, despite its educational value, there is a lack of interactive, user-friendly tools that allow learners to experiment with and visualize how these classical algorithms work in real-time. Most available tools are either too technical for beginners or lack customization features such as adjustable shift values, case sensitivity handling, or support for custom alphabets.

This project aims to address that gap by creating a web-based Caesar Cipher tool using HTML, CSS, and JavaScript. The tool allows users to perform encryption and decryption operations interactively, customize cipher parameters, and instantly view the results. It serves both as a practical learning aid and a demonstration of how simple cryptographic systems can be built using web technologies without relying on external frameworks.

4.Elements used In Project :

1. HTML Elements

These are used to create the structure and user interface of the application.

- `<form>` – Wraps the entire set of input controls and submits user data.
- `<input type="radio">` – Allows the user to choose between "Encode" and "Decode" modes.
- `<input type="text">` – Used to input the shift value, modulo, and custom alphabet.
- `<select>` – Provides dropdowns for selecting letter case handling and foreign character options.
- `<textarea>` – Used for multi-line text input (plaintext/ciphertext) and output display.
- `<label>` – Describes each input field for better usability and accessibility.
- `<h1>`, `<h2>` – Headings used for titles and section labels.
- `<div>` – Container elements used to group and style sections of the UI.
- `<script>` – Includes the JavaScript file for functional logic.

2. CSS Elements (Classes and Styles)

CSS is used for styling and layout control.

- **Flexbox Layout (**display: flex**)** – Aligns boxes and sections side-by-side responsively.
- **Box Shadows, Borders, Padding** – Visually separates content areas and adds aesthetic value.
- **Custom Fonts (**@import from Google Fonts**)** – Enhances typography using 'Roboto'.
- **Responsive Styling** – Ensures compatibility with various screen sizes and resolutions.
  - **Hover and Focus Effects** – Improve interactivity and user feedback for form controls and buttons.

3. JavaScript Functions and Components

JavaScript is responsible for all interactivity, logic, and dynamic output.

- **Event Listeners** – Detect clicks and form submissions.

> caesarCipher() **function** – Core function that applies Caesar cipher logic.

> removeForeignChars() **function** – Removes symbols and special characters when required.

- **String Manipulation** – Converts text to lowercase/uppercase and performs character shifting.
- **Modular Arithmetic** – Ensures that character indexing wraps around the alphabet correctly.
- **DOM Manipulation (**document.getElementById**,** textContent**)** – Dynamically updates input/output fields in real time.

5.Existing System

The existing systems or tools for Caesar cipher encryption are mostly limited to basic command-line programs, static desktop applications, or outdated online utilities. These typically have the following limitations:

- **Lack of Interactivity:** Most tools do not offer real-time encryption/decryption feedback based on user input changes.
- **Limited Customization:** Many implementations do not support configurable options like custom alphabets, case sensitivity, or non-alphabetic character handling.
- **Non-Responsive Design:** Existing tools are often not mobile-friendly or visually optimized, making them difficult to use across different devices.

- **No Educational Focus:** Tools often assume prior knowledge and do not guide the user through the encryption process, limiting their usefulness in educational contexts.
- **Dependency on Installation:** Some applications require downloads or thirdparty frameworks, reducing accessibility and convenience.

6.Proposed System

The proposed system is a modern, lightweight, browser-based Caesar Cipher encryption and decryption tool designed to be intuitive, customizable, and educational. Its key features include:

- **Web-Based Access:** The system runs entirely in a browser using HTML, CSS, and JavaScript—no installation required.
- **Interactive User Interface:** Users can input text, select encoding/decoding mode, and instantly view results without refreshing the page.
- **Customizable Settings:** Allows users to set shift values, choose letter casing, define custom alphabets, and decide how to handle symbols or foreign characters.
- **Responsive and Aesthetic Design:** Fully responsive interface designed with modern CSS styling, making it usable on desktops, tablets, and smartphones.
- **Educational Utility:** Tailored for students and learners, offering a clear visualization of how classical substitution encryption works.
- **Framework-Free Implementation:** Developed using only vanilla JavaScript, HTML5, and CSS3—ensuring fast loading and maximum compatibility.

7.Comparison Table

Feature	Caesar Cipher	Modern Encryption (AES)
Type	Substitution Cipher	Block Cipher
Key Complexity	Simple integer shift	128/192/256-bit key
Security Level	Low	High
Use Case	Educational, Simple	Secure Data Transfer
Implementation Effort	Easy	Complex
Real-Time Performance	Fast	Moderate
Flexibility	Low	High

8.Software Requirements :

- **Web browser (Chrome, Firefox, Edge)**

The application is built to run within any modern web browser that supports the latest web standards. Google Chrome, Mozilla Firefox, and Microsoft Edge are all compatible choices as they offer robust support for HTML5, CSS3, and JavaScript. These browsers also ensure high performance and responsive rendering of the user interface.

- **HTML5**

Hyper Text Markup Language version 5 is used to structure the content of the application. HTML5 provides the semantic elements required to design accessible, wellorganized interfaces. It supports features like form controls, input fields, and document structure without the need for external plugins.

- **CSS3**

Cascading Style Sheets version 3 is used for styling the application. CSS3 introduces powerful styling capabilities such as flexible box layout, transitions, custom fonts, and media queries. These features help ensure the user interface is visually appealing, userfriendly, and adaptable to different screen sizes.

- **JavaScript (ES6+)**

JavaScript is the programming language responsible for the core logic of the Caesar cipher. The application uses ECMAScript 6 (ES6) or later, which includes modern features like arrow functions, let/const declarations, and template literals that simplify code and enhance maintainability. It enables dynamic behavior such as form validation, text processing, and real-time output rendering.

- **No additional libraries or frameworks**

The project is intentionally built without reliance on external libraries (e.g., jQuery, React) or frameworks (e.g., Angular, Vue.js). This decision keeps the project lightweight, improves loading time, and makes it easier for learners to understand the core implementation of Caesar cipher logic using pure, vanilla JavaScript.

9.Hardware Requirements

➢ Processor: 1 GHz or faster

A basic processor with a clock speed of 1 GHz or higher is sufficient to run the Caesar Cipher web application. Since the application performs lightweight operations such as character substitutions and DOM manipulations, it does not demand high CPU power. Any modern laptop, desktop, or even mobile device meets this requirement.

- **RAM: Minimum 2 GB**

A minimum of 2 GB of RAM is recommended to ensure smooth operation of the web browser while running the application. This allows the system to handle browser tasks efficiently, especially when multiple tabs or applications are open concurrently.

- **Storage: 100 MB available space**

Although the application itself is lightweight and runs directly in the browser without installation, around 100 MB of free storage is suggested. This accounts for browser cache, temporary storage, and any saved files related to the project or offline use.

- **Display: 1024x768 or higher resolution**

A screen resolution of 1024x768 or higher is recommended to ensure that all interface elements are displayed correctly and comfortably. The application layout is designed to be responsive, but using a screen with at least this resolution improves usability and readability across devices.

10.Methodology

User Interface Design

- Designed using HTML5 to structure forms, buttons, labels, and text areas.
- Allows users to input plaintext or ciphertext and choose between encoding or decoding modes.
- Radio buttons and dropdown menus provide easy selection of shift values, casing preferences, and character filtering options.
- Clearly labeled interface enhances usability and accessibility.

Styling and Layout

- CSS3 is used to style and visually organize the interface components.
- Flexbox layout ensures that the design is responsive and adjusts across devices.
- Colors, fonts, and spacing are customized to improve readability and user engagement.
- Uses modern web font integration (Google Fonts) for improved aesthetics.

Logic Development with JavaScript

- JavaScript (ES6) implements the core Caesar cipher logic using modular arithmetic.
- Functions handle both encryption and decryption processes dynamically.
- DOM manipulation allows for real-time user interaction and result updates without reloading the page.
- Event listeners manage input submission and mode switching based on user actions.

Advanced Feature Handling

- Supports three letter casing options: maintain case, convert to lowercase, and convert to uppercase.
- Offers the ability to either remove or preserve non-alphabetic (foreign) characters.
- Allows users to define custom alphabets beyond standard a-z characters.
- Includes basic input validation to ensure numeric values for shift and modulo inputs.

Testing and Verification

- Tested with a variety of inputs including:

  All lowercase and all uppercase text

  Mixed alphanumeric characters

  Sentences with punctuation and symbols

- Multiple shift values and alphabet lengths were used to ensure correctness.

- Edge cases like empty input, invalid shift, and large wrap-around values are handled safely.
- Outputs were manually compared against expected results for accuracy.

Deployment and Execution

- Runs directly in any modern web browser (no installations required).

- Compatible with Chrome, Firefox, Edge, and similar browsers.
- Can be used locally by opening `index.html`, or deployed on any static web server.
- No external dependencies or libraries required, making it lightweight and portable.

11.Implementation

1. HTML (index.html) – Structure and Interface Layout

1.1 The HTML file defines the basic structure and content of the web interface.

1.2 Key Elements in the HTML file:

1.2.1 Radio Buttons: Allow the user to select between "Encode" and "Decode" modes.

1.2.2 Input Fields: Used to collect parameters like the shift key, modulo value, and the alphabet.

1.2.3 Dropdown Menus: Enable users to choose letter casing (maintain, lowercase, uppercase) and whether to ignore or remove foreign characters.

1.2.4 Textareas: One for entering plaintext or ciphertext and another for displaying the result.

1.2.5 Labels and Headings: Clearly describe each section, improving user experience and accessibility.

2. CSS (style.css) – Styling and Layout

2.1 The CSS file ensures that the interface is visually appealing and responsive.

2.2 Styling Techniques Used:

2.2.1 Flexbox Layout: Used to arrange sections in rows and columns, ensuring the design adapts to various screen sizes.

2.2.2 Color Scheme and Typography: Utilizes consistent colors and Google Fonts to create a clean, modern appearance.

2.2.3 Padding, Margins, and Borders: Applied to space out sections and elements for readability and alignment.

2.2.4 Interactive Feedback: Button hover effects and checked radio styles enhance user interactivity.

3. JavaScript (caesar.js) – Core Functionality and Logic

3.1 The JavaScript file implements the logic for Caesar cipher operations and user interaction.

3.2 Functional Modules:

3.2.1 Event Listeners: Detect when users switch between encoding and decoding or submit the form.

3.2.2 Caesar Cipher Function: Implements the encryption/decryption using modular arithmetic with custom alphabet handling.

3.2.3 Letter Case Management: Modifies the final output based on the user's selected case option.

3.2.4 Foreign Character Handling: Uses regular expressions to remove or ignore symbols, numbers, or punctuation.

3.2.5 Dynamic Output Display: Updates the output field in real-time based on user input and settings without page reload.

3.3 JavaScript is written using ES6+ syntax, making the code cleaner, modular, and easier to maintain.

12.Result

The Caesar Cipher web tool performs encryption and decryption with high accuracy, adhering strictly to the Caesar cipher algorithm defined by the user's input parameters. Upon entering text and selecting the desired operation mode—either encode or decode—the output is generated and displayed instantly without requiring a page reload. This real-time interaction significantly enhances the user experience, enabling immediate feedback and exploration.

One of the key strengths of this tool is its high level of customization. Users can adjust various parameters including the shift value, the length of the alphabet (modulo), character casing preferences (maintain, convert to lowercase, or uppercase), and how non-alphabetic characters (such as symbols and punctuation) should be treated. These features make it possible to experiment with different cipher configurations and observe how small changes influence the output.

For instance, when the user inputs the text "HELLO" with a shift value of 3 in encode mode, the resulting ciphertext displayed is "KHOOR." Conversely, when the ciphertext "KHOOR" is decoded using the same shift value, the original message "HELLO" is accurately restored. These examples demonstrate the tool's effectiveness in both encoding and decoding tasks.

Additionally, the application's handling of character case is precise. It respects the user's selection to either maintain the original casing or convert the entire output to a uniform case. The tool also offers the option to filter out or retain non-alphabetic characters, which adds an extra layer of flexibility for various use cases such as educational demonstrations, fun cryptographic exercises, or simple encoding tasks.

Overall, the results produced by this application are reliable, immediate, and fully configurable, offering a robust platform for understanding and experimenting with the Caesar cipher technique.

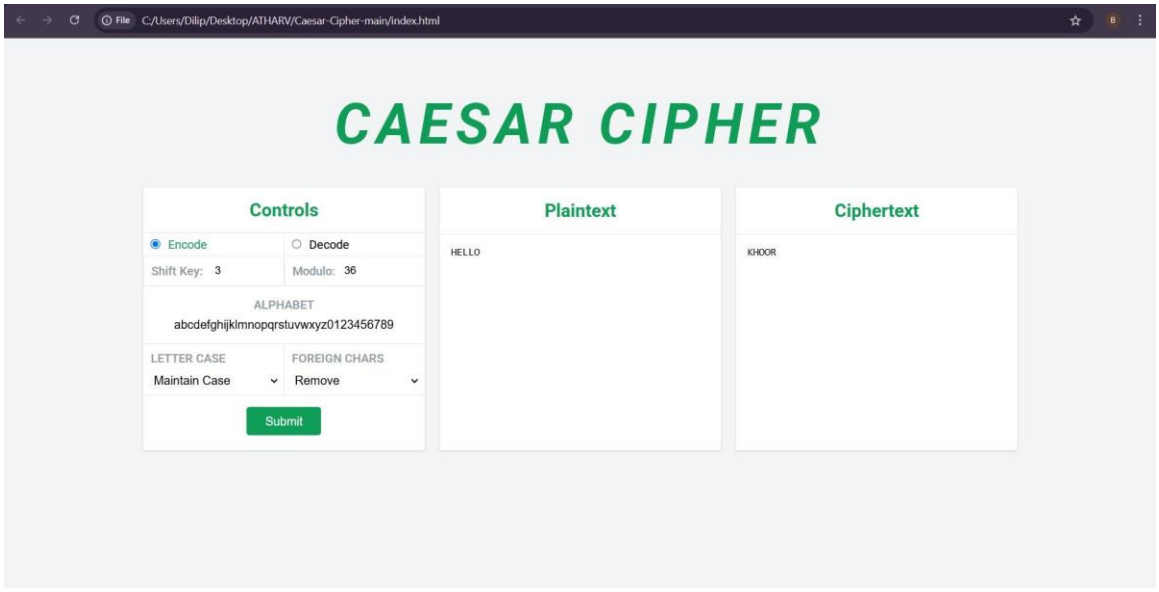
Examples:

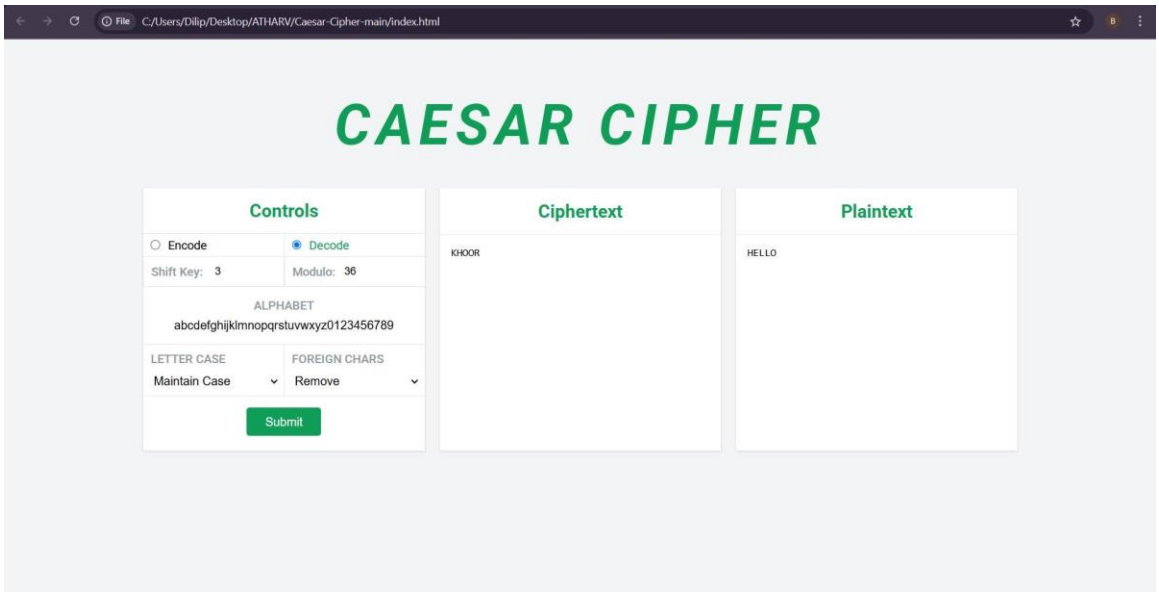
- Input: HELLO | Shift: 3 | Mode: Encode → Output: KHOOR

- Input: KHOOR | Shift: 3 | Mode: Decode → Output: HELLO

The tool supports both uppercase and lowercase handling, and can ignore or remove symbols depending on user configuration.

12.Screen output:





13.Conclusion

The Caesar Cipher project successfully demonstrates the foundational concepts of classical encryption through a simple yet engaging web-based application. By utilizing

core web technologies such as HTML for structure, CSS for styling, and JavaScript for interactive logic, the tool creates a cohesive and interactive platform that is accessible to a wide range of users—from beginners in computer science to students exploring cryptographic methods.

While the Caesar cipher is no longer considered secure for protecting sensitive information due to its simplicity and vulnerability to brute-force attacks, it remains one of the most effective teaching tools for introducing key principles in cryptography. These include substitution techniques, modular arithmetic, and the importance of encryption keys.

The implementation of this cipher within a browser-based interface offers users a hands-on experience with real-time encryption and decryption. Features such as configurable shift values, customizable alphabets, letter casing options, and foreign character handling make the tool not only functional but also flexible for educational experimentation.

Moreover, the modular and lightweight architecture of the project lays a strong foundation for future enhancements. It can be extended to support more sophisticated encryption algorithms like Vigenère Cipher, Transposition Cipher, or even introductory implementations of modern encryption methods. In doing so, this project not only educates users on classical encryption techniques but also prepares them for deeper exploration into the field of cybersecurity and data protection.