

**Name:** Muhammad Ahsan

**Enrollment no:** 03134221-018

**Course Name:** Information Security

**Instructor Name:** Dr. Nadeem Sarwar

**Assignment Name:** Web-Based Text Encryption Tool

**Due Date:** October 20, 2025

**1. Introduction**

In the modern digital world, data privacy and secure communication have become essential. Encryption plays a vital role in protecting sensitive information from unauthorized access. This project, Web-Based Text Encryption Tool, is designed to demonstrate various encryption techniques implemented through a single-page web application.

The tool allows users to enter plain text, choose an encryption algorithm (such as Caesar Cipher, AES, DES, Base64, or SHA-256), and perform encryption or decryption operations. It provides a simple, user-friendly interface for understanding the fundamentals of cryptography and how different encryption methods work.

**2. Objectives**

The main objectives of this project are as follows:

1. To develop a single-page web application that performs encryption and decryption.
2. To implement multiple encryption algorithms (Caesar Cipher, AES, DES, Base64, and SHA-256).
3. To ensure the interface is interactive, user-friendly, and visually appealing.
4. To validate user input to avoid empty fields or invalid selections.
5. To provide optional advanced features such as decryption, hashing, and copy-to-clipboard functionality.

**3. Project Description**

The Web-Based Text Encryption Tool enables users to encrypt or decrypt messages using different algorithms directly from their web browser.  
It uses HTML, CSS, and JavaScript, along with CryptoJS, a widely used JavaScript cryptographic library.

The system workflow is straightforward:

1. The user enters text in the input box.
2. The user selects an encryption algorithm.
3. A password or key (if required) is entered.
4. The user selects “Encrypt” or “Decrypt” mode.
5. The resulting ciphertext or plaintext is displayed.

**4. Core Features**

**Frontend (HTML, CSS, JavaScript)**

* Text area for plain text input.
* Drop down menu for selecting encryption algorithms.
* Password/key field for algorithms like AES, DES, and Caesar.
* Buttons for processing, copying, and resetting data.
* Output area to display encrypted or decrypted text.
* A visually appealing glass orphic design with gradient backgrounds and neon accents.

**Backend Logic (JavaScript)**

* Implemented encryption and decryption algorithms using CryptoJS.
* Caesar Cipher implemented it manually using ASCII shifting logic.
* Validation to prevent empty inputs or missing selections.
* Handling of encryption/decryption modes based on selected algorithms.

**5. Encryption Techniques Used**

**1. Caesar Cipher**

* A substitution cipher that shifts each letter by a fixed number.
* Example:
  + Plaintext: **HELLO WORLD**
  + Key: 3
  + Ciphertext: **KHOOR ZRUOG**

**Decryption:** Reverse shifting by the same number.

**2. Base64 Encoding**

* Converts text into ASCII-based encoding, often used for data transmission.
* Example:
  + Plaintext: **Hello Base64**
  + Ciphertext: **SGVsbG8gQmFzZTY0**

**Decryption:** Decoding the Base64 string restores the original text.

**3. AES (Advanced Encryption Standard)**

* A symmetric encryption algorithm used globally for secure data.
* Implemented using CryptoJS.AES with a user-provided password.
* Example:
  + Plaintext: This is a secret message
  + Password: myPassword123
  + Ciphertext: U2FsdGVkX1+N/1ls1Q2k4Vh4wldoV+oKJea7zQ3IjcU=

**Decryption:** Uses the same password to recover the original message.

**4. DES (Data Encryption Standard)**

* Another symmetric-key encryption algorithm used before AES.
* Implemented using CryptoJS.DES for demonstration purposes.
* Example:
  + Plaintext: DES example text
  + Password: abc123
  + Ciphertext: U2FsdGVkX1/AznkOZ7YI8AaRveQ3Cq6Uq3XALfS4IsQ=

**5. SHA-256 (Secure Hash Algorithm)**

* A **one-way hashing algorithm** used for password storage and integrity verification.
* Cannot be decrypted.
* Example:
  + Input: mypassword
  + Hash Output:  
    34819d7beeabb9260a5c854bc85b3e44b15bff32a4d5c0d6a5b8f06c3b5b9f1b

**6. System Design**

**User Interface Flow**

1. User opens the web application.
2. Enter plain text in the text area.
3. Selects the desired encryption algorithm.
4. Enter password or key if needed.
5. Chooses “Encrypt” or “Decrypt”.
6. Clicks “Run” to process.
7. Output is displayed and can be copied easily.

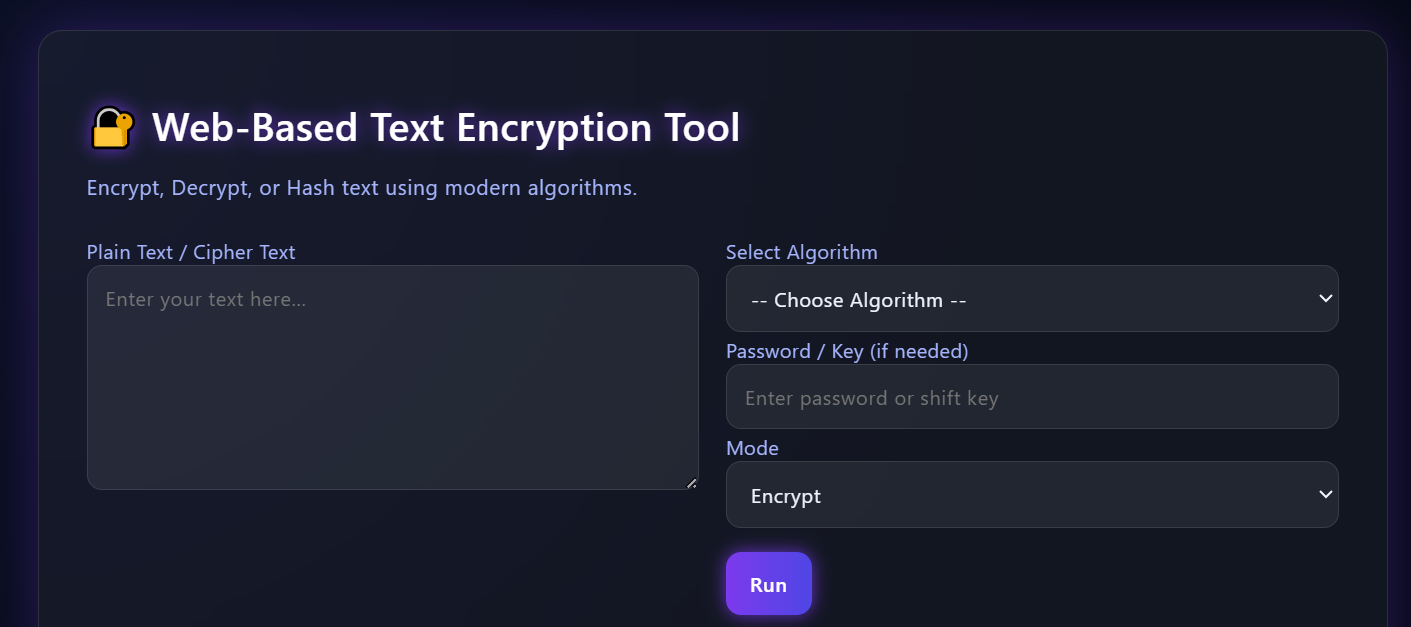
**Validation Features**

* Empty input prevention.
* Alerts for missing algorithm selection.
* Error handling for incorrect decryption keys.

**7. Technology Stack**

| **Layer** | **Technology Used** |
| --- | --- |
| Frontend | HTML5, CSS3, JavaScript |
| Encryption Library | CryptoJS |
| Backend Logic | JavaScript (Browser-based) |
| Styling | CSS (Neon glassmorphism theme) |
| Hosting | Any static web server / GitHub Pages |

**8. Screenshots (Suggested for Report)**



**9. Project Demo Video Link:**

<https://drive.google.com/file/d/1g5_qANmzaqcNMimhxOfBsmAtMzcPQ_iE/view?usp=sharing>

**10. Conclusion**

The Web-Based Text Encryption Tool successfully demonstrates the principles of data encryption and decryption through multiple algorithms. The project enhances understanding of both classical and modern cryptography and their implementation in web technologies. It serves as an educational tool and a lightweight browser-based encryption utility that requires no server ensuring privacy and simplicity.

**11. GitHub Repository Link:**

<https://github.com/mahsan6842/Web-Based-Text-Encryption-Tool>

**12. References**

* CryptoJS Library Documentation
* MDN Web Docs – atob() and btoa() Functions
* Wikipedia – Caesar Cipher, AES, DES, SHA-256