VISVESVARAYA TECHNOLOGICAL UNIVERSITY

Jnana Sangama ,Belagavi–590018

Report On

“AES Encryption-Decryption Algorithm”

Bachelor Of Engineering in Artificial Intelligence &Machine Learning

Submitted by

FARHAN AHMED SHARIEFF
(1AM23AI018)
NEHA RATANSINGH PATEL
(1AM23AI034)

Under the Support and Guidance of

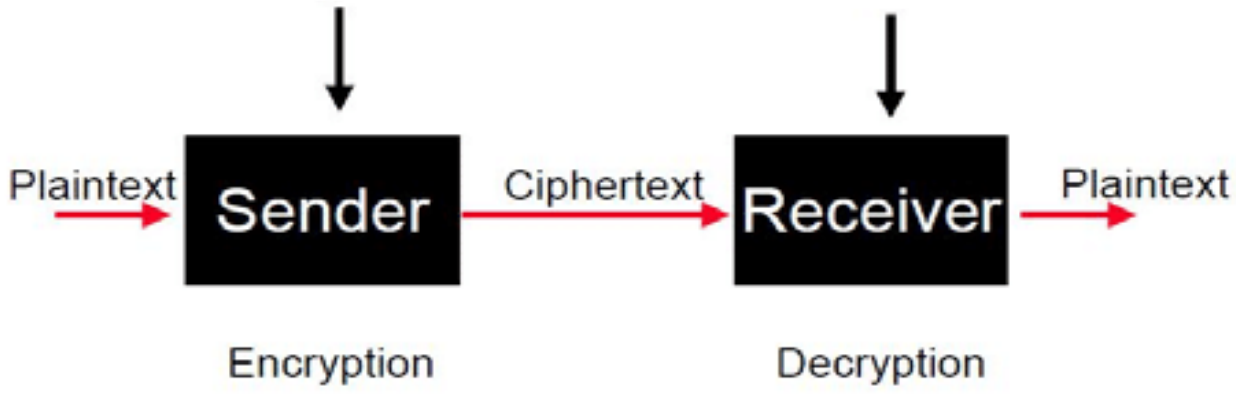
Prof. K B Bini

AMC ENGINEERING COLLEGE
DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND MACHINE LEARNING
18th K.M.Bannerghatta Main Road, Bengaluru-560083

2025-26

INTRODUCTION

In today's digital world, the security and confidentiality of information have become vital due to the rapid growth of online communication and data sharing. The AES Encryption Web Application is a browser-based tool developed to demonstrate how encryption ensures secure message transmission between users. It implements the **Advanced Encryption Standard (AES)** — a symmetric key encryption algorithm widely recognized for its efficiency and strong security. This project provides an interactive interface divided into two sections: the **Sender Side**, where plain text messages are encrypted into unreadable ciphertext, and the **Receiver Side**, where the same encrypted data can be decrypted back into its original form using the shared key. The application is built using **HTML**, **CSS**, and **JavaScript**, with the help of the **CryptoJS** library to perform AES operations directly within the browser environment. This implementation not only illustrates the practical use of cryptographic techniques but also highlights the importance of using secure key management to maintain data integrity and privacy. Overall, the project serves as a simple yet impactful demonstration of how modern encryption algorithms protect sensitive information in real-world communication systems.



DESCRIPTION ABOUT PROJECT

The AES Encryption Web Application is designed to demonstrate how digital messages can be securely exchanged using the **Advanced Encryption Standard (AES)** algorithm. The project focuses on ensuring confidentiality — meaning that only authorized users can read the information being sent. It uses a **symmetric key encryption** method, where the same secret key is used for both encryption and decryption.

The application consists of two main interfaces:

- **Sender Side:** Allows the user to input a plain text message and convert it into encrypted ciphertext using the AES algorithm. This encrypted data is unreadable to anyone who does not possess the secret key.
- **Receiver Side:** Enables the recipient to paste the encrypted text and decrypt it back to its original readable form using the same key.

The entire application is implemented using **HTML** for structure, **CSS** for styling, and **JavaScript** for functionality. The **CryptoJS** library is integrated to perform AES encryption and decryption directly within the browser, ensuring that no external server is required for processing the data.

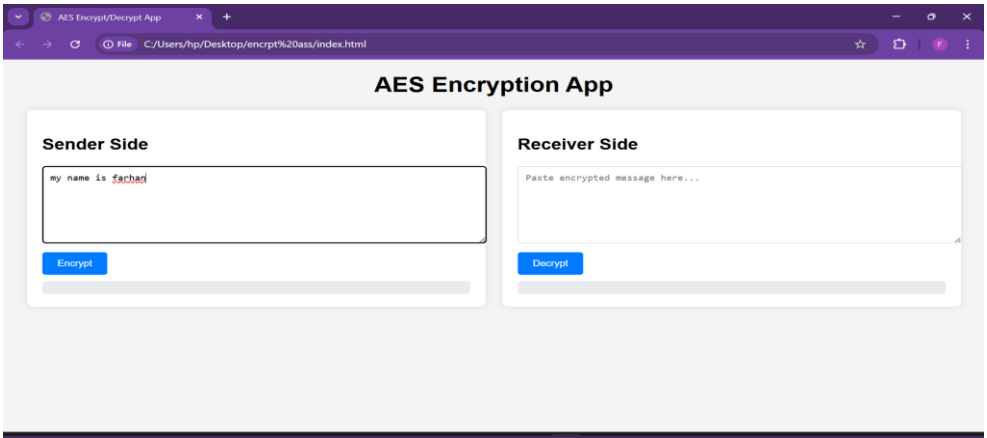
Through this project, users gain a clear understanding of how encryption works, how sensitive data can be protected from unauthorized access, and how modern web technologies can be used to build simple yet powerful cybersecurity tools. It serves as an educational demonstration of cryptographic principles and their application in secure communication system

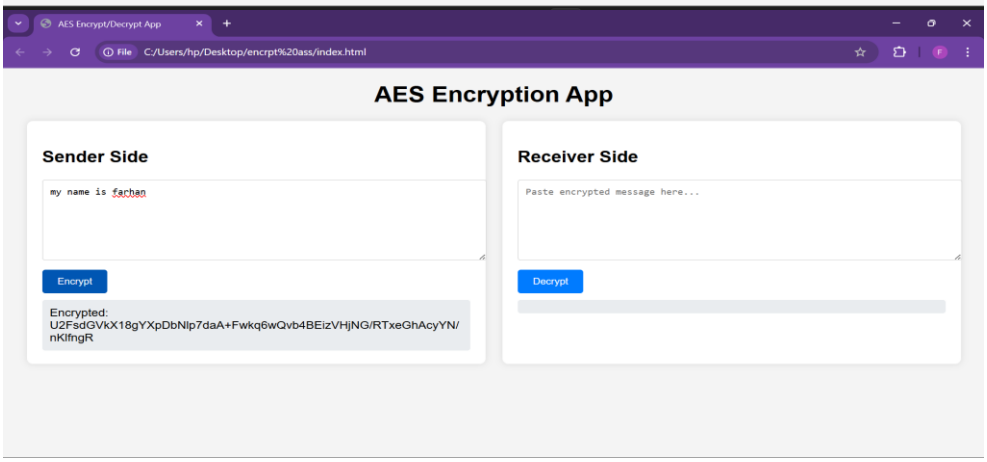
SOURCE CODE
(aes_app.js)

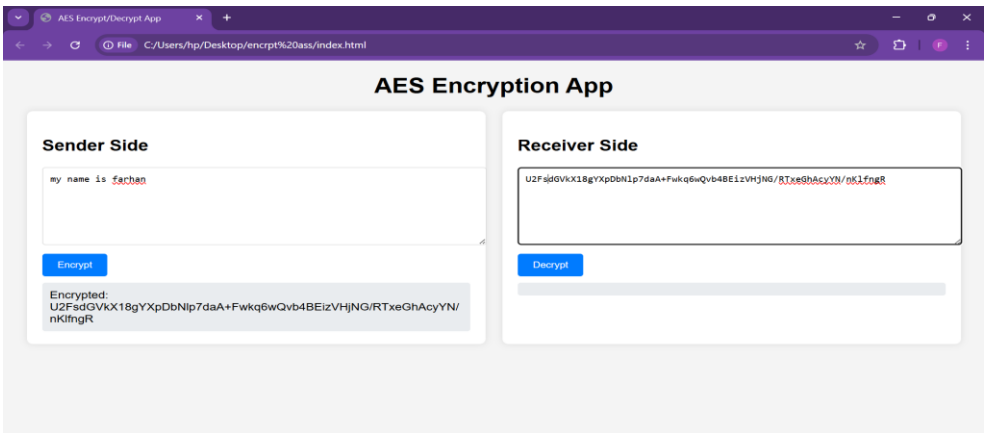
```
// AES Encryption App (Equivalent to the Python Tkinter version)
// Requires CryptoJS (CDN or import)

const key = CryptoJS.enc.Utf8.parse("mySecretKey12345"); // 16 bytes key
const iv = CryptoJS.enc.Utf8.parse("mySecretIV123456");  // 16 bytes IV

// Encryption function
function encryptMessage(message) {
    const encrypted = CryptoJS.AES.encrypt(
        CryptoJS.enc.Utf8.parse(message),
        key,
        {
            iv: iv,
            mode: CryptoJS.mode.CBC,
            padding: CryptoJS.pad.Pkcs7
        }
    );
    return encrypted.toString(); // Base64 encoded
}

// Decryption function
function decryptMessage(encryptedMessage) {
    try {
        const decrypted = CryptoJS.AES.decrypt(
            encryptedMessage,
            key,
            {
                iv: iv,
                mode: CryptoJS.mode.CBC,
                padding: CryptoJS.pad.Pkcs7
            }
        );
        return decrypted.toString(CryptoJS.enc.Utf8);
    } catch (error) {
        throw new Error("Invalid encrypted message");
    }
}

// Equivalent of AESApp class in JS
class AESApp {
    constructor() {
        this.senderInput = document.getElementById("senderInput");
        this.encryptedOutput = document.getElementById("encryptedOutput");
        this.receiverInput = document.getElementById("receiverInput");
        this.decryptedOutput = document.getElementById("decryptedOutput");
```

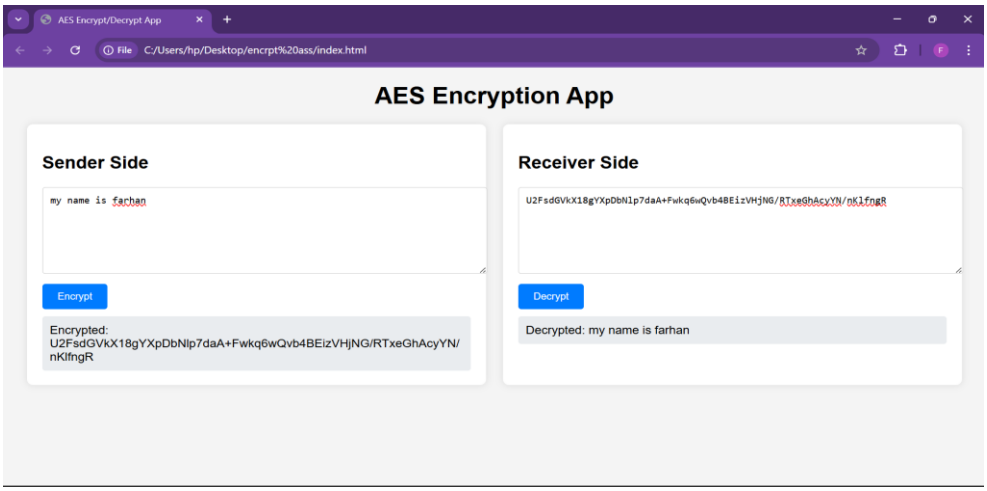
```
        document.getElementById("encryptBtn").addEventListener("click", () =>
this.encrypt());
        document.getElementById("decryptBtn").addEventListener("click", () =>
this.decrypt());

        // Simulate Python's "<Return>" key binding
        this.senderInput.addEventListener("keydown", (event) => {
            if (event.key === "Enter") {
                event.preventDefault();
                this.encrypt();
                const message = this.senderInput.value.trim();
                if (message) {
                    this.receiverInput.value = message;
                }
            }
        });
    }

    encrypt() {
        const message = this.senderInput.value.trim();
        if (!message) {
            alert("Please enter a message to encrypt.");
            return;
        }
        const encrypted = encryptMessage(message);
        this.encryptedOutput.innerText = `Encrypted: ${encrypted}`;
    }

    decrypt() {
        const encryptedMessage = this.receiverInput.value.trim();
        if (!encryptedMessage) {
            alert("Please enter an encrypted message to decrypt.");
            return;
        }
        try {
            const decrypted = decryptMessage(encryptedMessage);
            this.decryptedOutput.innerText = `Decrypted: ${decrypted}`;
        } catch (error) {
            alert("Invalid encrypted message.");
        }
    }
}

// Initialize when page loads
window.onload = () => {
    new AESApp();
};
```

SCREENSHOTS









CONCLUSION

The AES Encryption Application effectively demonstrates the concept and practical working of the Advanced Encryption Standard (AES) algorithm using a simple and interactive interface. Through this project, users can understand how plaintext messages are converted into encrypted ciphertext and later restored to their original form using the same secret key and initialization vector (IV). The application ensures data confidentiality by applying AES in CBC (Cipher Block Chaining) mode, making it resistant to pattern-based attacks.

This project also emphasizes the real-world importance of cryptography in securing digital communication — such as in messaging apps, banking systems, and secure data transfers. The use of a graphical interface (built using Tkinter or web UI) makes the process user-friendly and easy to follow, bridging the gap between theoretical encryption concepts and their real-time implementation.

Overall, the AES Encryption App successfully showcases the core principles of symmetric key cryptography, promotes awareness of cybersecurity practices, and provides a strong foundation for future development of more advanced encryption-based communication systems.