AES-128 bit Computer Security Project

Overview:

This project implements AES 128-bit(Advanced Encryption Standard) encryption and decryption functionality to securely handle sensitive data. It is designed with a modern architecture featuring a **React-based frontend** and a **Python Flask backend** to provide a seamless and efficient user experience.

* **Frontend (React):** The user interface allows users to easily upload files or input text for encryption and decryption, displaying results in real time.
* **Backend (Flask):** The backend handles the cryptographic operations, leveraging the AES algorithm to perform secure encryption and decryption, ensuring data confidentiality.

This project serves as a robust solution for applications requiring secure data exchange and protection.

Backend:

**Encryption**: The plaintext and a key are the inputs, and the ciphertext is the output, with provided steps of what is done in each round.

**Decryption**: The ciphertext and the key are the inputs, and the plaintext is the output, with provided steps of what is done in each round.

**AES logic**: implements cryptographic operations such as add round key, sub bytes, shift rows, and mix columns

**Steps provided**: each round operations are provided in order to encrypt/decrypt

**Validate input**: plain text and key must be exactly 128 bits

Frontend:  
The frontend contains one page which is the homepage that have 2 options encrypt and decrypt. On clicking “encrypt” two textboxes are displayed which are plaintext and key, and on clicking decrypt the two textboxes are the ciphertext and the key and finally a button to encrypt or decrypt. The encryption/decryption processes is displayed real-time.

Summary:

AES-128 bit encryption and decryption: The AES 128-bit is implemented for ensure the confidentiality of data.

User-friendly GUI: a GUI using react which has a reactive background and easy to interact with.

Real-time visualization: each round of the encryption/decryption process is displayed to the user to inform him the operations done each round.

Application Screenshots:

Simple system architecture: