**1. Project Title**

**Crypto WebApp – Secure Encryption and Decryption using AES & RSA**

**2. Introduction**

The **Crypto WebApp** is a Flask-based web application that allows users to encrypt and decrypt messages using two widely used cryptographic algorithms: **AES (Advanced Encryption Standard)** and **RSA (Rivest–Shamir–Adleman)**.

The system provides a simple and attractive web interface where the user can:

* Enter a plaintext message.
* Choose an encryption method (**AES** or **RSA**).
* View the encrypted ciphertext.
* Decrypt ciphertext back to plaintext.

This project demonstrates the use of **symmetric encryption (AES)** and **asymmetric encryption (RSA)**, along with a clean web interface for user interaction.

**3. Objectives**

* Provide users with a web-based platform to securely encrypt and decrypt messages.
* Demonstrate practical implementation of **AES (symmetric)** and **RSA (asymmetric)** encryption techniques.
* Offer a user-friendly interface to understand the difference between encryption methods.
* Build a secure and educational tool for learning cryptography.

**4. Technologies Used**

* **Backend Framework:** Flask (Python)
* **Frontend:** HTML, CSS (via Jinja2 templates)
* **Cryptography Library:** PyCryptodome
* **Programming Language:** Python
* **Other Tools:** Base64 encoding/decoding

**5. System Design**

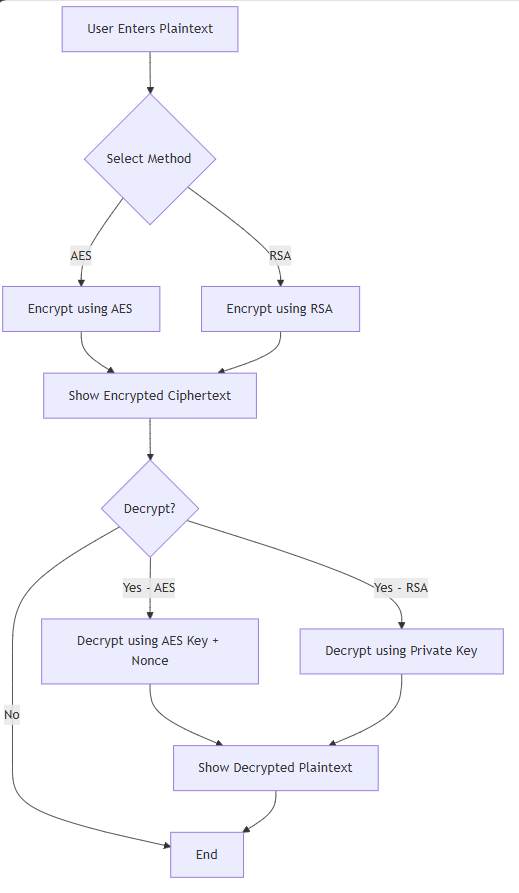
**5.1 AES (Advanced Encryption Standard)**

* **Key Size:** 16 bytes (128-bit)
* **Mode Used:** EAX (provides both confidentiality and authentication)
* **Process:**
  1. Generate random AES key.
  2. Encrypt plaintext with AES cipher (EAX mode).
  3. Package **nonce + tag + key + ciphertext** into one string.
  4. Encode using Base64 for transmission.
  5. During decryption, extract key, nonce, and ciphertext → decrypt & verify.

**5.2 RSA (Rivest–Shamir–Adleman)**

* **Key Size:** 2048 bits
* **Mode Used:** PKCS1\_OAEP (Optimal Asymmetric Encryption Padding)
* **Process:**
  1. Generate RSA public and private key pair.
  2. Use the **public key** for encryption.
  3. Use the **private key** for decryption.
  4. Encrypted data is Base64 encoded for display.

**6. Workflow Diagram**



**7. Features**

✔ Encrypt messages using **AES or RSA**.  
✔ Decrypt encrypted text to original plaintext.  
✔ Secure handling of keys & ciphertexts.  
✔ Easy-to-use **web interface** with form inputs.  
✔ Demonstrates real-world cryptographic concepts.