🔐 Secure Chat App with End-to-End Encryption

By Golla venkatesham

🧭 Introduction

In an age where digital communication is omnipresent, ensuring privacy and data security is more critical than ever. This project, developed by Golla, presents a secure, real-time chat application that implements end-to-end encryption (E2EE) using a hybrid cryptographic model. The goal is to ensure that only the intended recipients can decrypt and read the messages, safeguarding user private.

🧠 Abstract

The Secure Chat App is a real-time messaging platform that uses RSA and AES encryption to protect user communication. Each user generates a unique RSA key pair, and messages are encrypted using a randomly generated AES key. This AES key is then encrypted with the recipient’s RSA public key, ensuring secure key exchange. Flask-SocketIO enables real-time communication, while optional encrypted chat logs provide secure message storage. This project demonstrates the practical application of cryptographic principles in a user-friendly web environment.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

🛠️ Tools Used

| **Tool/Library** | **Purpose** |
| --- | --- |
| Python | Core programming language |
| Flask + Flask-SocketIO | Web framework and real-time messaging |
| cryptography | RSA and AES encryption/decryption |
| HTML/CSS/JS | Frontend interface |
| SQLite (optional) | Encrypted chat log storage |

🧩 Steps Involved in Building the Project

1. **RSA Key Generation**
   * Each user generates a 2048-bit RSA key pair upon registration.
   * Public keys are shared with the server; private keys remain securely on the client side.
2. **AES Key Generation & Encryption**
   * A new AES key is generated for each message.
   * The message is encrypted using AES (CFB mode).
   * The AES key is encrypted using the recipient’s RSA public key.
3. **Real-Time Messaging with Flask-SocketIO**
   * Socket events (connect, send\_message, receive\_message) handle encrypted message transmission.
   * Messages are routed only to the intended recipient.
4. **Client-Side Decryption**
   * The recipient decrypts the AES key using their RSA private key.
   * The message is decrypted using the AES key and displayed in the chat interface.
5. **Encrypted Chat Logs (Optional)**
   * Encrypted messages and metadata are stored in a secure JSON or SQLite format.
   * No plaintext messages are stored on the server.

# ✅ Conclusion

This project, developed by Golla venkatesham, successfully demonstrates the implementation of end-to-end encryption in a real-time chat application. By combining RSA for secure key exchange and AES for efficient message encryption, the app ensures confidentiality and integrity in digital communication. The modular and extensible design allows for future enhancements such as group messaging, file sharing, and mobile integration. This project not only reinforces cryptographic concepts but also showcases their real-world application in cybersecurity.