Overall Idea

The application allows multiple clients to communicate securely over a network. It enables users to join a chat room, send messages, and leave the chat, all while ensuring that messages are encrypted for confidentiality.

Features

1. **Client-Server Architecture**: The application consists of a server that manages client connections and facilitates communication between clients.
2. **User Authentication**: Each client provides a name upon connecting, which is displayed in the chat.
3. **Secure Communication**: Messages are encrypted before being sent, ensuring that they cannot be easily intercepted or read by unauthorized parties.
4. **Threading**: The server handles multiple clients simultaneously using threading, allowing for real-time communication.
5. **Graceful Disconnection**: Clients can leave the chat gracefully, with notifications sent to other participants.

Encryption and Decryption Algorithms

1. **RSA (Rivest-Shamir-Adleman)**:
   - **Purpose**: Used for securely exchanging a shared secret key between the server and clients.
   - **Key Generation**: Each client and the server generate their own RSA key pairs (public and private keys).
   - **Public Key Encryption**: The server sends its public key to the client, which uses it to encrypt the shared secret key.
2. **AES (Advanced Encryption Standard)**:
   - **Purpose**: Used for encrypting the actual chat messages exchanged between clients.
   - **Key Size**: The code uses AES-256, which requires a 256-bit key.
   - **Mode of Operation**: Cipher Block Chaining (CBC) mode is used, which provides confidentiality by chaining together blocks of plaintext.
3. **Caesar Cipher**:

  - **Purpose**: A simple substitution cipher is used to encrypt messages before broadcasting them to clients.
  - **Shift Value**: A fixed shift value (3) is used for encrypting and decrypting messages.

Summary

This chat application demonstrates fundamental principles of secure communication. By combining RSA for key exchange and AES for message encryption, it ensures that messages remain confidential. The use of threading allows multiple users to communicate in real time, making it a practical example of both networking and cryptography in software development.