Encrypted Chat App - Code Sections

Server Code

```
# SERVER CODE (server.ipynb equivalent)
import socket
import threading
from Crypto.Cipher import AES
import base64

KEY = b'16byteslongkey!!'

def encrypt(message):
    cipher = AES.new(KEY, AES.MODE_EAX)
    ciphertext, tag = cipher.encrypt_and_digest(message.encode())
    return base64.b64encode(cipher.nonce + tag + ciphertext).decode()

def decrypt(data):
    data = base64.b64decode(data)
    nonce = data[:16]
    tag = data[16:32]
    ciphertext = data[32:]
    cipher = AES.new(KEY, AES.MODE_EAX, nonce=nonce)
    return cipher.decrypt_and_verify(ciphertext, tag).decode()

clients = []

def handle_client(conn, addr):
    print(f"[+] Connected: {addr}")
    while True:
        try:
            encrypted_msg = conn.recv(4096).decode()
            if not encrypted_msg:
                break
            message = decrypt(encrypted_msg)
            print(f"{addr}: {message}")

            for c in clients:
                if c != conn:
                    c.send(encrypt(message).encode())
        except:
            break
    conn.close()
    clients.remove(conn)

def start_server():
    server = socket.socket()
    server.bind(("0.0.0.0", 8080))
    server.listen()
    print("Server started on port 8080")
    while True:
        conn, addr = server.accept()
        clients.append(conn)
        threading.Thread(target=handle_client, args=(conn, addr)).start()

start_server()
```

Client Code

```
# CLIENT CODE (client.ipynb equivalent)
import socket
import threading
from Crypto.Cipher import AES
```

```
import base64

KEY = b'16byteslongkey!!'

def encrypt(message):
    cipher = AES.new(KEY, AES.MODE_EAX)
    ciphertext, tag = cipher.encrypt_and_digest(message.encode())
    return base64.b64encode(cipher.nonce + tag + ciphertext).decode()

def decrypt(data):
    data = base64.b64decode(data)
    nonce = data[:16]
    tag = data[16:32]
    ciphertext = data[32:]
    cipher = AES.new(KEY, AES.MODE_EAX, nonce=nonce)
    return cipher.decrypt_and_verify(ciphertext, tag).decode()

def receive_messages(sock):
    while True:
        try:
            encrypted_msg = sock.recv(4096).decode()
            if encrypted_msg:
                msg = decrypt(encrypted_msg)
                print("\n[Chat] " + msg)
        except:
            break

sock = socket.socket()
sock.connect(("127.0.0.1", 8080))

print("Connected to chat server!")
threading.Thread(target=receive_messages, args=(sock,), daemon=True).start()

def send_message(text):
    encrypted = encrypt(text)
    sock.send(encrypted.encode())
    print(f"You: {text}")
```