**Set Up Environment**

pip install cryptography

**Server Implementation**

import socket

import threading

import sqlite3

from cryptography.fernet import Fernet

# Setup the database

def setup\_db():

conn = sqlite3.connect('chat\_users.db')

cursor = conn.cursor()

cursor.execute('CREATE TABLE IF NOT EXISTS users (username TEXT PRIMARY KEY, password TEXT)')

conn.commit()

conn.close()

setup\_db()

# Encrypt and Decrypt functions

def generate\_key():

return Fernet.generate\_key()

def encrypt\_message(key, message):

fernet = Fernet(key)

return fernet.encrypt(message.encode())

def decrypt\_message(key, encrypted\_message):

fernet = Fernet(key)

return fernet.decrypt(encrypted\_message).decode()

# Handle client connections

def handle\_client(client\_socket):

key = generate\_key()

while True:

try:

message = client\_socket.recv(1024)

if not message:

break

decrypted\_message = decrypt\_message(key, message)

print(f"Received: {decrypted\_message}")

response = encrypt\_message(key, "Message received")

client\_socket.send(response)

except Exception as e:

print(f"Error: {e}")

break

client\_socket.close()

# Start server

def start\_server():

server = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

server.bind(('0.0.0.0', 12345))

server.listen(5)

print("Server listening on port 12345")

while True:

client\_socket, addr = server.accept()

print(f"Accepted connection from {addr}")

client\_handler = threading.Thread(target=handle\_client, args=(client\_socket,))

client\_handler.start()

if \_\_name\_\_ == "\_\_main\_\_":

start\_server()

**Client Implementation**

import socket

from cryptography.fernet import Fernet

def connect\_to\_server():

client = socket.socket(socket.AF\_INET, socket.SOCK\_STREAM)

client.connect(('127.0.0.1', 12345))

return client

def send\_message(client, key, message):

encrypted\_message = encrypt\_message(key, message)

client.send(encrypted\_message)

def encrypt\_message(key, message):

fernet = Fernet(key)

return fernet.encrypt(message.encode())

def main():

key = input("Enter your session key: ") # Use the same key as the server

client = connect\_to\_server()

while True:

message = input("Enter message (or 'exit' to quit): ")

if message.lower() == 'exit':

break

send\_message(client, key.encode(), message)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**User Authentication**

pip install bcrypt