EX: 12 b) chat client server

Date: 23/10/24

Aim:

To implement chat client server using TCP/ UDP sockets

Procedure:

chat_server.py:

→ Create a UDP socket & bind it to 127.0.0.1 and port 12345

→ Wait for a message from a client using recv2-py.

→ Decode & print the received message.

→ Take input from the user, encode it, & sent it back to the client using echo.

→ If the reply is 'end', break the loop & close the socket.

hat client server

nt chat client
el UDP sockets

ver.py:

socket & bind
and port 12345
message from
2.py.
the received

the user, encode
t to the client

'end', break the
ocket.

receiver.py:

→ create a UDP socket.
→ prompt the user for input & send the message.
→ wait for a response.
→ decode & print the message.
→ If the user input is 'bend', break the loop & close the socket.

code:

chat_server.py:

```
import socket
def receiver():
    port = 12345
    host = '127.0.0.1'
    with socket.socket(socket.AF_INET, socket.SOCK_DGRAM) as s:
        s.bind((host, port))
        while (True):
            d, add = s.recvfrom(1024)
            print("Client", (d.decode()))
            a = input("Enter reply")
            s.sendto(a.encode(), add)
            if (a == "end"):
                break
                exit
recv()
```

recv1.py:

```
import socket
import time
def recv2(a):
    host = '127.0.0.1'
    port = 12345
    with
socket.socket(socket.AF_INET,
    socket.SOCK_DGRAM) as s:
    s.sendto(a.encode(), (host, port))
    d, addr = s.recvfrom(1024)
    print("Received", d.decode())
while (True):
    a = input("Enter message")
    if (a == "end")
        recv1(a)
        break
    else:
        recv2(a)
```

Output:

```
Enter Message : hello
Received 'Hi'
Enter Message : I'm from REC
Received 'Me also'
```

Result:

Implementation for chat client server using TCP / UDP socket is successfully implemented