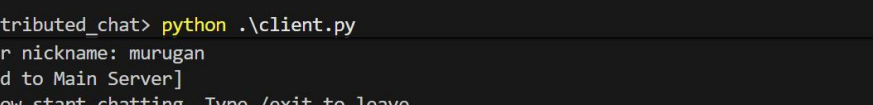


```
PS D:\distributed_chat> python .\server.py
PS D:\distributed_chat> python .\server.py
PS D:\distributed_chat> python .\server.py
PS D:\distributed_chat> python .\server.py
[SERVER RUNNING] on 127.0.0.1:5555
[NEW CONNECTION] murugan connected from ('127.0.0.1', 63082)
[NEW CONNECTION] ganesh connected from ('127.0.0.1', 63085)
```

```
PS D:\distributed_chat> python .\backup_server.py
PS D:\distributed_chat> python .\backup_server.py
PS D:\distributed_chat> python .\backup_server.py
[BACKUP SERVER RUNNING] on 127.0.0.1:5556
```



```
PS D:\distributed_chat> python .\client.py
Enter your nickname: murugan
[Connected to Main Server]
You can now start chatting. Type /exit to leave.
Connected! Commands: /users, /msg <num> <msg>, /r <msg>
murugan>
[INFO] You are the only one here.

(12:32) ganesh joined the chat!

murugan> 
```

```
TERMINAL PORTS
PS D:\distributed_chat> python .\client.py
Enter your nickname: ganesh
[Connected to Main Server]
You can now start chatting. Type /exit to leave.
Connected! Commands: /users, /msg <num> <msg>, /r <msg>
ganesh>
[INFO] Current users:
1) murugan
2) ganesh

ganesh> 
```

```
PS D:\distributed_chat> python .\client.py
Connected! Commands: /users, /msg <num> <msg>, /r <msg>
ganesh>
[INFO] Current users:
1) murugan
2) ganesh

(12:38) shathi joined the chat!

(12:39) [DM from shathi]: hi how are you
ganesh> /r hello good to see you
ganesh> 
```

TERMINAL PORTS

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.12.3 @ Go Live

```
PS D:\distributed_chat> python .\client.py
[Connected to Main Server]
You can now start chatting. Type /exit to leave.
Connected! Commands: /users, /msg <num> <msg>, /r <msg>

[INFO] Current users:
1) murugan
2) ganesh
3) shathi

shathi> /msg 2 hi how are you
[INFO] Message sent to ganesh.

shathi> shathi> 
```

Ln 1, Col 1 Spaces: 4 UTF-8 CRLF {} Python 3.12.3 @ Go Live

Clientcode

```
5
6  nickname = input("Enter your nickname: ")
7
8  client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9
10 try:
11     client.connect(('127.0.0.1', 5555))
12     print("[Connected to Main Server]")
13 except ConnectionRefusedError:
14     print("[!] Main server down, attempting to connect to backup...")
15     try:
16         client.connect(('127.0.0.1', 5556))
17         print("[Connected to Backup Server]")
18     except ConnectionRefusedError:
19         print("[!] Both servers are down. Exiting.")
20         sys.exit()
21
22 stop_thread = False
23
24 def receive():
25     global stop_thread
26     while not stop_thread:
27         try:
28             message = client.recv(1024).decode('utf-8')
29             if message == 'NICK':
30
31
32
33
34 def write():
35     print("You can now start chatting. Type /exit to leave.")
36     while not stop_thread:
37         try:
38             msg = input(f"{nickname}> ")
39             if stop_thread:
40                 break
41
42             if msg.lower() == "/exit":
43                 client.close()
44                 break
45
46             # Send the message only if it's not empty
47             if msg:
48                 client.send(msg.encode('utf-8'))
49
50         except (EOFError, KeyboardInterrupt):
51             client.close()
52             break
53     except:
54         break
```

Serverfile:

```
server.py > ...
1  import socket
2  import threading
3  from datetime import datetime
4
5  HOST = '127.0.0.1'
6  PORT = 5555
7
8  server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
9  server.bind((HOST, PORT))
10 server.listen()
11
12 clients = {} # client socket -> nickname
13 last_dm_partner = {} # client socket -> last DM sender socket
14
15 print(f"[SERVER RUNNING] on {HOST}:{PORT}")
16
17 def broadcast(message, _client=None):
18     """Send message to all clients except _client if specified"""
19     for client in list(clients.keys()):
20         if client != _client:
21             try:
22                 client.send(message)
23             except:
24                 client.close()
25                 if client in clients:
26                     del clients[client]
27                 if client in last_dm_partner:
28                     del last_dm_partner[client]
29
30 def send_user_list(client):
31     """Send numbered list of current users to a client"""
32     if len(clients) > 1:
33         message = "[INFO] Current users:\n"
34         client_list = list(clients.keys())
35         for idx, c in enumerate(client_list):
36             name = clients[c]
37             message += f"{idx + 1}) {name}\n"
38         client.send(message.encode('utf-8'))
39
```

```

54     # ## MODIFIED ## Add timestamp to all messages
55     timestamp = datetime.now().strftime('%H:%M')
56
57     if msg_decoded.lower() == "/users":
58         send_user_list(client)
59
60     elif msg_decoded.startswith("/msg"):
61         try:
62             parts = msg_decoded.split(" ", 2)
63             num = int(parts[1]) - 1
64             text = parts[2]
65             client_list = list(clients.keys())
66
67             if 0 <= num < len(client_list):
68                 target_client = client_list[num]
69                 sender = clients[client]
70

```

```

138 def server_console():
139     while True:
140         msg = input('')
141         if msg.lower() == "/disconnect":
142             print("[SERVER] Shutting down.")
143             broadcast("[SERVER] Server is shutting down.\n".encode('utf-8'))
144             for client in list(clients.keys()): client.close()
145             server.close()
146             break
147         else:
148             broadcast(f"[SERVER] {msg}".encode('utf-8'))
149
150 threading.Thread(target=server_console, daemon=True).start()
151 receive()
152 print("[SERVER] Closed.")

```