```
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
```

TERMINAL PORTS python

```
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 python

```
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 python

```
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

```python
nickname = input("Enter your nickname: ")

client = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

try:
    client.connect(('127.0.0.1', 5555))
    print("[Connected to Main Server]")
except ConnectionRefusedError:
    print("[!] Main server down, attempting to connect to backup...")
    try:
        client.connect(('127.0.0.1', 5556))
        print("[Connected to Backup Server]")
    except ConnectionRefusedError:
        print("[!] Both servers are down. Exiting.")
        sys.exit()

stop_thread = False

def receive():
    global stop_thread
    while not stop_thread:
        try:
            message = client.recv(1024).decode('utf-8')
            if message == 'NICK':
```

```python
def write():
    print("You can now start chatting. Type /exit to leave.")
    while not stop_thread:
        try:
            msg = input(f"{nickname}> ")
            if stop_thread:
                break

            if msg.lower() == "/exit":
                client.close()
                break

            # Send the message only if it's not empty
            if msg:
                client.send(msg.encode('utf-8'))

        except (EOFError, KeyboardInterrupt):
            client.close()
            break
        except:
            break
```

Serverfile:

server.py > ...

```python
import socket
import threading
from datetime import datetime

HOST = '127.0.0.1'
PORT = 5555

server = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
server.bind((HOST, PORT))
server.listen()

clients = {}  # client socket -> nickname
last_dm_partner = {} # client socket -> last DM sender socket

print(f"[SERVER RUNNING] on {HOST}:{PORT}")
```

```python
def broadcast(message, _client=None):
    """Send message to all clients except _client if specified"""
    for client in list(clients.keys()):
        if client != _client:
            try:
                client.send(message)
            except:
                client.close()
                if client in clients:
                    del clients[client]
                if client in last_dm_partner:
                    del last_dm_partner[client]

def send_user_list(client):
    """Send numbered list of current users to a client"""
    if len(clients) > 1:
        message = "[INFO] Current users:\n"
        client_list = list(clients.keys())
        for idx, c in enumerate(client_list):
            name = clients[c]
            message += f"{idx + 1}) {name}\n"
        client.send(message.encode('utf-8'))
```

```python
            # ## MODIFIED ## Add timestamp to all messages
            timestamp = datetime.now().strftime('%H:%M')

            if msg_decoded.lower() == "/users":
                send_user_list(client)

            elif msg_decoded.startswith("/msg"):
                try:
                    parts = msg_decoded.split(" ", 2)
                    num = int(parts[1]) - 1
                    text = parts[2]
                    client_list = list(clients.keys())

                    if 0 <= num < len(client_list):
                        target_client = client_list[num]
                        sender = clients[client]
```

```python
def server_console():
    while True:
        msg = input('')
        if msg.lower() == "/disconnect":
            print("[SERVER] Shutting down.")
            broadcast("[SERVER] Server is shutting down.\n".encode('utf-8'))
            for client in list(clients.keys()): client.close()
            server.close()
            break
        else:
            broadcast(f"[SERVER] {msg}".encode('utf-8'))

threading.Thread(target=server_console, daemon=True).start()
receive()
print("[SERVER] Closed.")
```