

## Report : WhatsApp-Style Chat with Clock Synchronization

### 1. Objective

To develop a simple chat application using Python sockets, threading, and Tkinter, with integrated clock synchronization using Cristian's algorithm.

### 2. System Overview

- Server: Multi-threaded socket server that broadcasts messages and handles clock sync requests.
- Client: Tkinter-based GUI for chat and time display, maintaining both local and synchronized clocks.
- Communication: Implemented using TCP sockets for reliable, real-time message delivery.

### 3. Clock Synchronization (Cristian's Algorithm)

Clients send a sync request to the server and record local timestamps( $t_{\text{send}}$  and  $t_{\text{recv}}$ ). The offset is calculated as:  $\text{offset} = t_{\text{server}} + (\text{RTT}/2) - t_{\text{recv}}$ . This ensures client clocks remain aligned with the server's clock.

### 4. Results

- Multiple clients successfully exchange messages in real time.
- Clock synchronization verified through periodic '[sync] clock adjusted' messages.
- GUI displays both local and synced time accurately.
- System stable with and without simulated drift.

### 5. Conclusion

All assignment objectives were achieved: threaded server-client communication, Tkinter GUI interface, message broadcasting, and Cristian's clock synchronization successfully implemented.