Group 6

Asher amir and Rewaj panta

Server-Client Communication

• The server registers itself on 0.0.0.0:12345 (all the network interfaces).

• The clients establish a connection to 127.0.0.1:12345 (the localhost).

• The communication happens through JSON objects classified into four types: chat, pm, file, and sync_request.

• The server sends messages to all the clients and forwards the private messages only to the recipient clients.

____

2. Threading and Concurrency

• On the server side, there is a separate thread for each client connecting to the server.

• The client side comprises two threads - one for GUI and the other for receiving messages.

• This guarantees a real-time chat without the need of blocking the other clients.

____

3. Cristian's Clock Synchronization

• The clients simulate the clock drift (a very small offset from the real time).

• The clients ask the server for the time every five seconds.

• offset calculation:

• offset = server_time + RTT/2 - client_time

- The clients correct their time according to the server time which is already synced.

____

4. Tkinter GUI

- The chat window shows the messages with the time stamps of the client and the time stamps of the synced server.
- The input box can be used for sending normal messages, private messages (/pm), and files.
- The Local Time, Synced Server Time, and Offset are shown on the labels.

____

5. Conclusion

- The project illustrates the creation of a multi-client TCP chat application in Python sockets.
- The proper implementation of threading technology for the concurrent processing of clients.
- The classical methods of clock synchronization developed by Cristian are applied.
- A Tkinter GUI is developed for the interactive communication.
- The optional file transfer feature is included and accompanied with checks of file sizes.