Requisitos funcionais do projeto

RF1: O sistema deverá exibir na tela de todos os usuários conectados, incluindo o servidor, uma mensagem exibindo o nome do usuário que acabou de se conectar, assim que o usuário novo entrar.

RF2: Mensagem unicast para um usuário específico conectado no chat (utilizando o comando /nomedousuario).

RF3: O chat deve ser em “tempo real” para todos os usuários conectados, para isso utilize theads.

RF4: No servidor deverá ser exibido o IP e a porta do usuário que se conectou no chat.

RF5: Utilizar o comando “/sair” para sair do chat, assim que sair deverá ser enviado uma mensagem via broadcast informando que o usuário saio do chat.

RF6: Deverá ser feita a validação dos dados de entrada/conexão, bem como, validados possíveis erros nas funções do socket (O sistema deverá realizar o tratamento de exceções utilizando os comandos try e except).

RF7: O sistema deverá utilizar TCP (Protocolo de Controle de Transmissão) para a transmissão de dados, pois código estabelece uma conexão antes que os dados sejam trocados utilizando o método accept() no servidor e connect() no cliente, garantir a entraga com o método recv(1024) e fazer controle de fluxo com o método except.

Utilização de socket no cliente e no servidor

Servidor: O servidor cria um socket para escutar as conexões de clientes. Quando um cliente se conecta, o servidor cria uma nova thread para gerenciar essa conexão. O servidor usa o socket do cliente para receber e enviar mensagens, utilizando o tratamento de broadcast para enviar mensagens a todos os clientes conectados.

Cliente: O cliente cria um socket para se conectar ao servidor e envia seu nome. Ele utiliza o socket para enviar e receber mensagens e processa o recebimento de mensagens do servidor em uma thread separada, garantindo que a interface gráfica continue funcional. As mensagens enviadas ao servidor podem ser broadcasted (para todos) ou unicast (para um único destinatário), dependendo da lógica implementada no servidor.

A comunicação entre servidor e cliente é estabelecida através do socket TCP/IP, permitindo a troca de dados de forma confiável e ordenada. O broadcast é utilizado

para enviar mensagens a todos os clientes conectados, enquanto o unicast pode ser usado para enviar mensagens privadas entre usuários.

Utilização das threads no cliente e no servidor

No servidor, as threads são usadas para lidar com múltiplos clientes simultaneamente. Cada vez que um novo cliente se conecta, o servidor cria uma nova thread para gerenciar a comunicação com esse cliente, permitindo que o servidor continue aguardando por novas conexões ao mesmo tempo.

No cliente, as threads são usadas para manter a interface gráfica responsiva enquanto o cliente está conectado ao servidor e recebendo mensagens. Sem as threads, a interface gráfica ficaria travada ou não atualizaria quando o cliente estiver aguardando por mensagens do servidor.