Id du groupe évaluant :

Description du requirement	Exemples et précisions	Oui	Partiel	Non	Commentaires
Un utilisateur doit spécifier l'adresse IP et le port lors du lancement du client (1.1)	Le programme gère correctement l'oubli d'un paramètre. Le programme peut fonctionner avec une adresse ip valide, et pas uniquement "127.0.0.1".				
Un utilisateur doit spécifier le port lors du lancement du serveur (1.2)	Le programme gère correctement l'oubli du port. Il est nécessaire de tester sur plusieurs ports différents.				
Le serveur doit être capable d'accepter les connexions et de répondre à plusieurs clients en même temps (1.3)	Il faudra tester avec au moins 4 clients.				
Le client doit pouvoir prendre une chaîne de caractère tapée au clavier, l'envoyer au serveur et recevoir de ce dernier la même chaîne de caractère (1.4)	Des chaînes de caractères très longues (>100) ou avec des caractères spéciaux (\) sont correctement gérées. Ce cas est valable si l'utilisateur n'utilise pas de commande spéciale et n'est pas dans un salon.				
Le client doit pouvoir gérer les chaînes de caractère tapées au clavier et les messages provenant du serveur en même temps (1.5)	L'utilisateur peut à tout moment commencer à taper un message, ou recevoir un message provenant du serveur.				
Le serveur, en recevant une chaîne de caractère depuis un client, doit répéter cette chaîne uniquement à ce même client (1.6)	Si plusieurs clients sont connectés, seul celui ayant envoyé le message le reçoit (en mode echo). Ce cas est valable si l'utilisateur n'utilise pas de commande spéciale et n'est pas dans un salon.				
Un utilisateur doit pouvoir couper la connection ('/quit') (1.7)	La commande '/quit' permet de quitter correctement le client s'il n'est pas dans un salon, et est gérée par le serveur.				

Une fois la connexion établie avec le serveur, le client doit s'identifier par son pseudo ('/nick') (2.1)	Si l'utilisateur tape un autre message que '/nick', il n'est pas pris en compte.				
Le cas particulier où un utilisateur se connectent avec un pseudo trop long ou un pseudo avec des espaces et autres caractères spéciaux doivent être gérés (2.1)	Des pseudos tels que 'T0t0 le r0xx0r du 33' ne font pas crasher le serveur ou le client.				
Un utilisateur doit recevoir une erreur si le pseudo qu'il désire est déjà attribué (2.2)	Il faut essayer de mettre deux fois le même pseudo.				
Le serveur doit gérer plusieurs utilisateurs et plusieurs connexions (2.3)	Le serveur gère le cas où un client A se connecte, puis un client B se connecte, puis A se déconnecte et reconnecte.				
Le serveur doit tenir compte du changement de pseudo d'un utilisateur (2.4)	Il faut tester avec des nouveaux pseudos, puis avec des pseudos existants.				
Un utilisateur doit pouvoir obtenir du serveur la liste des utilisateurs connectés ('/who') (2.5)	Il faut bien penser à vérifier que la liste correspond bien aux utilisateurs qui sont connectés.				

Un utilisateur doit pouvoir obtenir du serveur des informations sur un utilisateur en particulier ('/whois') (2.6)	Le format des dates, adresses IP et ports est correct.				
Un utilisateur doit pouvoir envoyer un message à tous les autres utilisateurs ('/msgall') (2.7)	Il faut vérifier que le message est bien reçu par tout le monde.				
Un message envoyé en broadcast, multicast ou anycast ne doit pas être retransmis à l'expéditeur (2.8)	Il s'agit d'un message avec /msg, /msgall, ou d'un message dans un salon.				
Un utilisateur doit pouvoir envoyer un message privé à un autre utilisateur ('/msg') (2.9)	Il faut vérifier que le message est bien reçu par l'utilisateur.				
Le serveur doit traiter les requetes '/msg' lorsque l'utilisateur spécifié n'existe pas (2.10)	Le client et le serveur ne doivent pas crasher. Dans l'idéal, un message d'erreur est retourné au client.				
Le serveur doit renvoyer le message à l'utilisateur si aucune commande n'est tapée avant le message (2.11)	Ce cas est valable si l'utilisateur n'est pas dans un salon.				

Un utilisateur doit pouvoir créer un salon ('/create') (3.1)					
Les cas particulier où un utilisateur déclare un salon avec des espaces ou des caractères spéciaux doivent être gérés (3.1)	Des noms tels que 'T4t4 le super sal0n du 33' ne font pas crasher le serveur ou le client.				
Un utilisateur doit pouvoir rejoindre automatiquement le salon qu'il vient de créer (3.2)					
Un utilisateur doit pouvoir demander la liste des salons ('/channel_list') (3.3)	Il faut vérifier que cette liste est cohérente.				
Le serveur doit retourner un message d'erreur à l'utilisateur qui demande la création d'un salon déjà existant (3.4)					
Un utilisateur doit pouvoir rejoindre et quitter un salon ('/join' et '/quit') (3.5)	Pour le "/join", il faut vérifier que rien ne crash si le salon n'existe pas. Le "/quit" permet de faire quitter un salon si l'utilisateur est dans un salon, ou de faire se déconnecter le client s'il n'est pas dans un salon.				
Un utilisateur inscrit à un salon doit pouvoir changer de salon, ce qui lui fait quitter le salon en cours (3.6)	Il faut également vérifier ce qu'il se passe si le salon demandé n'existe pas.				
Le serveur doit détruire le salon lorsque son dernier occupant le quitte (3.7)	Si le dernier utilisateur d'un salon le quitte et tente de le rejoindre, cela doit échouer.				
Un message envoyé dans un salon ne doit pas être transmis à d'autres utilisateurs que ceux présents dans le salon (3.8)	La création de trois salons avec deux utilisateurs dans chacun ne fait pas crasher le serveur ou le client. Les messages doivent être reçus correctement par les utilisateurs concernés.				

Un utilisateur (l'émetteur) doit pouvoir envoyer un fichier à un autre utilisateur (le récepteur) ('/send') (5.1)					
Lors du transfert d'un fichier, le récepteur doit donner son approbation (5.2)					

Si le récepteur refuse l'échange de fichier, l'émetteur doit en être informé (5.4)					
Lors du transfert d'un fichier, le récepteur et l'émetteur doivent avoir confirmation que l'envoi s'est déroulé correctement (5.5)					

Tout fonctionne en IPv6 et IPV4.	Le client gère correctement l’ adresse IPv6 '::1' pour se connecter au serveur.				

Jalon bonus	Décrivez les fonctionnalités dans la case Commentaires puis évaluez le bonus (Oui = +2 / Partiel = +1 / Non = +0).				