INF3405–Réseaux informatiques

Hiver 2024

TP No. 1 Système de clavardage interactif

Groupe 05

2125282 – Irem Yildiz

2089901 – Thasaarah Vasanthakumar

2211017 – Andreea Maria Ghioltan

Soumis à :

Bilal Itani

Mehdi Kadi

16 février 2024

Introduction

De nos jours, nous avons recours à plusieurs plateformes pour discuter avec les autres. Cependant, par le biais de leur usage, nous accordons beaucoup de pouvoir aux géants des réseaux sociaux. Frustrés par cette réalité, nous décidons de créer notre propre application client-serveur afin de communiquer aisément avec notre entourage, sans être brimés par ces grosses compagnies.

Dans le cadre de ce laboratoire, nous avons développé une interface console de clavardage en nous familiarisant avec les sockets afin d'établir une connexion privée entre un serveur et des clients sur un réseau, ainsi que les threads permettant de gérer plusieurs connexions clientes de manière simultanée. Le système est constitué de deux applications principales : un serveur central gérant les connexions et un client permettant aux utilisateurs de s'engager dans des conversations. Le serveur est conçu pour être exécuté sur une machine dédiée, capable de gérer plusieurs connexions client simultanément, tandis que le client est développé pour fonctionner sur n'importe quelle machine avec accès au réseau, permettant à l'utilisateur de se connecter au serveur et d'échanger des messages en temps réel.

Le serveur démarre en demandant à l'utilisateur d'entrer une adresse IP (127.0.0.1 qui est utilisé dans ce laboratoire) valide et un port d'écoute (entre 5000 et 5050). Une fois lancé, il est capable de gérer les connexions entrantes, d'authentifier les utilisateurs, de stocker et de transmettre l'historique des messages. Les données utilisateur sont sauvegardées dans une base de données locale, permettant une persistance des informations entre les sessions. Le client, offre une interface en ligne de commande pour la saisie de l'adresse IP du serveur et du port, ainsi que les identifiants de l'utilisateur. Après vérification de la validité des données saisies, le client se connecte au serveur, reçoit l'historique des derniers messages et peut participer à la conversation en temps réel.

Ce travail pratique permet également de développer deux qualités de l'ingénieur définies par le BCAPG : la conception et la communication.

Présentation

Lorsque le serveur est lancé, il est demandé de saisir son adresse IP et son port. Une fois validé, le serveur initialise un socket *Listener* spécifique configuré de manière que le port puisse être réutilisé suite à sa fermeture. Une boucle infinie est utilisée pour garder le serveur à l'écoute en tout temps jusqu'à la connexion d'un client qui génère une nouvelle classe *ClientHandler*.

La librairie GSON a été intégrée à l'application pour convertir les objets Java en format JSON afin de stocker la base de données des correspondances nom d'usager et mot de passe localement.

Pour régler le problème de cet enjeu d'espionnage dans les chats, plusieurs classes ont été implémenté pour former en ensemble l'architecture de base du système de clavardage interactif, chacune jouant un rôle spécifique pour assurer le fonctionnement fluide et sécurisé de la communication entre les utilisateurs.

Server:

La classe Server agit comme le cœur du système de clavardage. En tant que point d'entrée principal de l'application serveur, elle est conçue pour rester opérationnelle en continu, assurant une écoute constante pour les tentatives de connexion des clients. Sa fonction principale est de gérer le démarrage du serveur, d'initialiser les ressources nécessaires et de créer une instance de *ClientHandler* pour chaque client qui se connecte. Le serveur maintient également une liste des flux de sortie (*DataOutputStreams*) correspondant à chaque client connecté.

ClientHandler:

ClientHandler est la classe responsable de la gestion individuelle de chaque connexion client. Pour chaque client connecté au serveur, une instance distincte de *ClientHandler* est créée. Cette classe s'occupe de toutes les interactions avec le client : elle traite les requêtes d'informations, établit la connexion, authentifie l'utilisateur, et gère l'envoi et la réception des messages. Le *ClientHandler* joue un rôle crucial dans la gestion des communications, assurant que les messages envoyés par un client sont correctement retransmis à tous les autres clients via le serveur. Il agit comme un intermédiaire entre le client et le serveur.

Client:

La classe *Client* représente le programme utilisé par les utilisateurs pour interagir avec le serveur de clavardage. Fonctionnant de manière indépendante du serveur, cette application client permet aux utilisateurs de se connecter au serveur, de s'authentifier, et d'envoyer/recevoir des messages au sein du système de clavardage. L'interface utilisateur du client est conçue pour être intuitive, permettant une interaction facile avec le serveur pour une expérience utilisateur optimale. La classe gère également la connexion réseau, l'envoi des requêtes au serveur, et la réception des messages du serveur, assurant une communication transparente au sein du système de clavardage.

User:

La classe *User* est un modèle de données représentant un utilisateur du système de clavardage. Elle contient les informations essentielles sur l'utilisateur, telles que son nom d'utilisateur, son mot de passe, et éventuellement d'autres métadonnées associées à son profil. Cette classe est utilisée principalement pour la gestion de l'authentification et la sauvegarde des informations utilisateur dans la base de données.

UserDataBase:

UserDataBase est responsable de la gestion de la base de données locale des utilisateurs (*UserDB*.json). Elle contient une liste de tous les utilisateurs inscrits dans le système (indépendamment de leur état de connexion) et fournit des fonctionnalités pour manipuler cette base de données : ajout de nouveaux utilisateurs, chargement des informations utilisateur au démarrage du serveur, récupération des informations d'un utilisateur spécifique, et validation des identifiants utilisateur lors de la tentative de connexion. Cette classe joue un rôle crucial dans la persistance et la gestion sécurisée des données utilisateur, assurant l'intégrité et la confidentialité des informations dans le système de clavardage.

UserDB:

UserDB.json est le fichier de base de données local qui stocke les informations de tous les comptes utilisateurs. Ce fichier sert de stockage persistant pour les données des utilisateurs, permettant au système de maintenir un état cohérent des comptes utilisateurs, indépendamment de leur statut en ligne ou hors ligne.

Message:

La classe *Message* est conçue pour encapsuler les informations relatives aux messages échangés dans le système. Elle contient le nom d'utilisateur de l'expéditeur, son adresse IP et son port, la date et l'heure d'envoi du message, ainsi que le contenu du message lui-même. Grâce à la méthode *toString*(), cette classe offre une représentation formatée des messages, facilitant leur affichage et leur gestion au sein du système.

MessageDataBase:

MessageDataBase.java joue un rôle similaire à *UserDataBase*, mais pour la gestion des messages. Elle maintient une liste des messages échangés dans le système et gère le fichier de base de données local des messages (*messageDB*.json). Cette classe permet d'ajouter de nouveaux messages à la base de données, de charger les messages existants, de sauvegarder les messages dans le fichier de base de données, et de fournir l'affichage des 15 derniers messages échangés.

MessageDB.json:

MessageDB.json est le fichier qui sert de base de données locale, stockant tous les messages échangés par les clients, accompagnés de leurs informations détaillées. Cette persistance des données assure que l'historique des conversations est maintenu, même après un redémarrage du serveur, permettant ainsi une continuité dans l'expérience de clavardage des utilisateurs.

Validation:

La classe validation vérifie la validité de l'adresse IP. La méthode consiste à vérifier que l'adresse IP entrée soit non-nulle, séparée en 4 parties correspondant à des entiers de 0 à 255 par des points sans toutefois que l'adresse IP se termine par un point.

Difficultés rencontrées

Les problèmes rencontrés étaient reliés à l'installation de la librairie GSON. En effet, le problème principal résidait dans la difficulté à connecter correctement la base de données à notre application, ce qui nous empêchait de lire ou d'écrire dans les fichiers JSON. La solution à ce problème est venue de l'utilisation de Maven, un outil de gestion et d'automatisation de projet pour Java. Maven a permis de gérer les dépendances de manière plus efficace, en automatisant le téléchargement, l'installation et la mise à jour des librairies nécessaires, dont GSON. Grâce à Maven, il était possible d'inclure facilement le fichier .jar de la librairie GSON dans notre projet, résolvant ainsi les problèmes de connexion à la base de données.

De plus, synchroniser les échanges de messages entre clients et serveur pour prévenir les plantages était difficile. La solution a consisté à adopter une communication asynchrone via des threads séparés, garantissant une interaction fluide et stable.

Critiques et améliorations

Il serait intéressant d'améliorer l'application en intégrant plusieurs instances représentant chacun une salle de clavardage indépendante. Cette modification permettrait de diviser le volume de données reçu entre chaque instance, ce qui rend l'application globalement plus équilibrée et améliore les performances.

Conclusion

Ce projet de système de clavardage interactif nous a permis d'appliquer concrètement nos connaissances en programmation réseau et en développement logiciel. Il a souligné l'importance de la conception soignée et de la programmation rigoureuse dans le développement de solutions technologiques répondant à des besoins réels de communication sécurisée. Cette expérience a été une opportunité enrichissante de développement professionnel et personnel, nous ouvrant à de nouvelles perspectives dans le domaine des réseaux informatiques.