INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Rapport de Projet Java

ChatSystem

Réalisé par :

ABONNEAU Benjamin

DUAN Yu

GUO Junjiang

Groupe B - 4IR - Promo 57

Sommaire

I. Introduction

Dans le cadre de notre formation en 4ème année Informatique et Réseaux, il nous est demandé de réaliser un projet d'un système de clavardage (Chat System). Pendant plusieurs séances, nous avons fini ce projet en respectant toutes les notions dans le cahier des charges.

Lors du processus de création de toute application, et même de tout projet de manière générale, nous avons utilisé les connaissances dans les cours de PDLA, UML, Réseaux, Programmation en Java, Base de Données, etc. En plus, avec l'application JIRA, nous pouvons mieux gérer notre projet pendant les séances de TD de gestion de projet.

Le rapport suivant rassemble ainsi une présentation de notre projet, la conception et les diagrammes réalisées, l'architecture du système et les choix technologiques, les procédures d'évaluation et des tests, la procédure d'installation et de déploiement et finalement un petit manuel d'utilisation de notre application.

II. Conception et diagrammes réalisées

- Diagramme d'utilisation

Dans le cahier des charges, en analysant les demandes des clients, nous avons une première compréhension de ce projet. Nous avons ensuite réalisé un diagramme d'utilisation basé sur cette compréhension.

- Diagrammes de classes

Ensuite, on a réfléchi aux classes qu'on va utiliser dans la programmation Java. A l'aide de notre professeur, nous avons terminé la version initiale du diagramme de classe. Cependant, au cours du développement du code, nous avons découvert que certaines classes peuvent être fusionnées et que d'autres devaient être ajoutées. Ainsi, après avoir écrit le code, nous avons terminé la version finale du diagramme de classe.

- Diagrammes de séquences

Les différents diagrammes de séquence qu'on a décidé d'effectuer permettent de montrer les échanges au cours du temps des différents composants du système. Il est important de notifier tous les utilisateurs actifs du réseau lors de leur connexion, déconnexion et lorsqu'il change leur pseudo.

CONNEXION : l'utilisateur se connecte à l'application et demande donc une connexion au réseau.

RENOMMER : Chaque utilisateur possède un pseudonyme unique et il peut le changer à tout moment.

RÉDUIRE : l'utilisateur peut cliquer sur un bouton afin de réduire l'application.

DÉCONNEXION : l'utilisateur se déconnecte de l'application et demande donc une déconnexion du réseau.

ÉCHANGE : l'utilisateur envoie ou reçoit un message depuis un autre utilisateur.

III. Architecture du système et choix technologiques

- Base de données

Afin de gérer les échanges de messages et en particulier la gestion de l'historique, nous avons décidé d'utiliser une base de données locale sur la machine de l'INSA. Chaque client aura sa propre base de données sur sa machine, comprenant une table ***CONVERSATION*** qui répertorie l'historique des messages.

Dans cette table, nous avons quatre colonnes :
DATE : text qui représente l'horodatage des messages.
MESSAGE : text qui représente les contenus de messages.
SENDERPSEUDO : varchar(30) qui représente le pseudonyme de l'émetteur du message.
RECEIVERPSEUDO : varchar(30) qui représente le pseudonyme du récepteur du message.

Nous avons utilisé SQLite pour réaliser cette base de données et créé un fichier nommé "java.db" pour enregistrer les informations là-dessus.

- Swing

Pour créer nos interfaces graphiques, nous avons choisi d'utiliser SWING qui permet à l'application de présenter la même interface sur tout système d'exploitation au choix de l'utilisateur. Avec les TDs de Java, nous avons essayé d'écrire quatre interfaces simples de notre cru, qui sont décrites dans les sections suivantes.

- Maven

En utilisant notre connaissance du cours PDLA, nous avons créé directement un projet maven pour faciliter notre application. Il est utilisé pour résoudre le principal problème de l'importation de jar pour les dépendances des classes java et la compilation des projets java. Nous utilisons l'attribut de

dépendance dans le fichier pom.xml pour gérer les paquets jar dépendants, qui contiennent des fichiers de classe et certains fichiers de ressources nécessaires.

Grâce au plugin d'assemblage Apache Maven, ce format nous permet d'obtenir un seul fichier exécutable contenant toutes les bibliothèques et dépendances du projet. Par exemple, afin de connecter SQLite à notre projet Java, nous avions besoin d'un pilote approprié. Nous avons cherché sur le web et l'avons ajouté directement au fichier pom.xml, et cela a fonctionné.

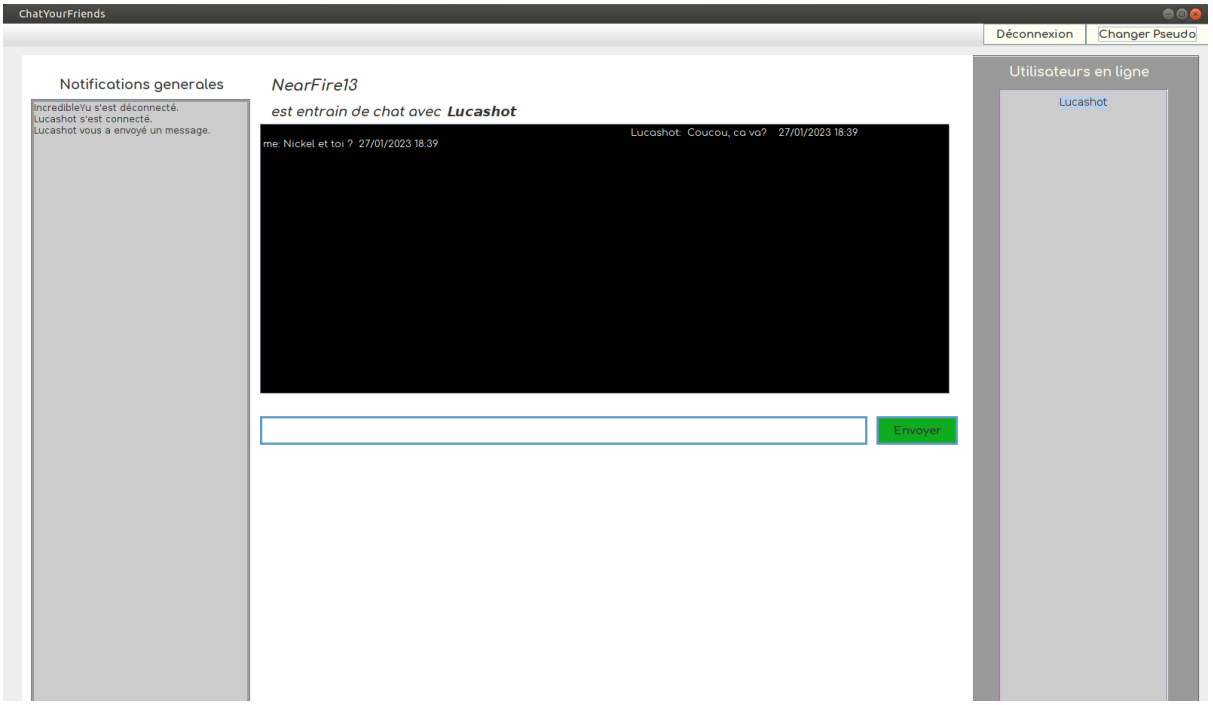
IV. Procédures d'évaluation et de tests

Nous avons testé les classes *Database*, *Message*, *Controller*, *UserManager* au fur et à mesure de leur conception et en direct en affichant les informations sur la console. Nous avons également des tests unitaires pour la classe UserManager afin de s'assurer que toutes les fonctions fonctionnaient correctement. Pour s'assurer que les fonctions de la classe Database et que la base de données fonctionnait correctement, nous avons créé deux utilisateurs (user1, user2 sur le screen ci-dessous) et des messages qui leur sont associés. Ainsi, après affichage, nous pouvons voir qu'ils sont bien dans la base de données et que tout fonctionne correctement.

```
Le nom de driver c'est SQLite JDBC
Création de database avec succès.
Sender: _toto_10.1.5.20_1234
Receiver:  _tata_10.1.5.20_1235
Time:  23/01/2023 10:56
Data:  Hello user2

Sender: _tata_10.1.5.20_1235
Receiver:  _toto_10.1.5.20_1234
Time:  23/01/2023 10:56
Data:  Hello user1
```

Une fois qu'on fut sûr que les classes fonctionnaient correctement, nous sommes passés aux tests sur les interfaces graphiques.



Nous pouvons bien sélectionner un utilisateur dans la section "*Utilisateurs en ligne*" à droite. Nous avons testé avec succès la réception et la transmission de messages par le protocole TCP.

En cliquant sur le bouton "Changer de pseudo" en haut à droite, nous testons avec succès la réception de l'information de tous les autres utilisateurs de cette opération par diffusion UDP. Sur les autres ordinateurs, nous pouvons voir que le pseudonyme de l'utilisateur a changé dans la liste des utilisateurs sur la droite. En plus, nous avons même réussi à charger l'historique des messages même après un changement de pseudo en effectuant une mise à jour dans la base de données.

La section "*Notification générale*" à gauche informe toutes les opérations ci-dessus (l'utilisateur s'est connecté, l'utilisateur a changé de pseudonyme, l'utilisateur envoie un message, etc.).

V. Procédure d'installation et de déploiement

- Installation

Afin de pouvoir utiliser notre ChatSystem, il faut tout d'abord récupérer le logiciel. Pour ça, il vous faut cloner le répertoire suivant:

git clone https://github.com/IncredibleYu/ChatSystemProject.git

Avant de lancer l'application, il faut aussi taper la commande :

mvn clean package

Cette commande permet de faire une combinaison de sept commandes, à savoir, clean, resources, compile, testResources, testCompile, test et jar; les sept principales étapes pour s'assurer d'un bon fonctionnement de l'application par la suite.

- Utilisation

Pour lancer notre programme, vous pouvez choisir un de trois commandes suivant :

mvn exec:java -Dexec.mainClass="fr.insa.gei.ChatSystemProject.Main"
ou
java -cp target/classes fr.insa.gei.ChatSystemProject.Main
ou
java -jar target/ChatYourFriends-jar-with-dependencies.jar

La troisième solution est la meilleure façon afin d'être sûr que l'application se lancera correctement.

- Distribution

Même commande qu'avant :

mvn clean package

Pour distribuer le fichier situé dans votre propre machine aux autres utilisateurs :

target/ChatYourFriends-jar-with-dependencies.jar

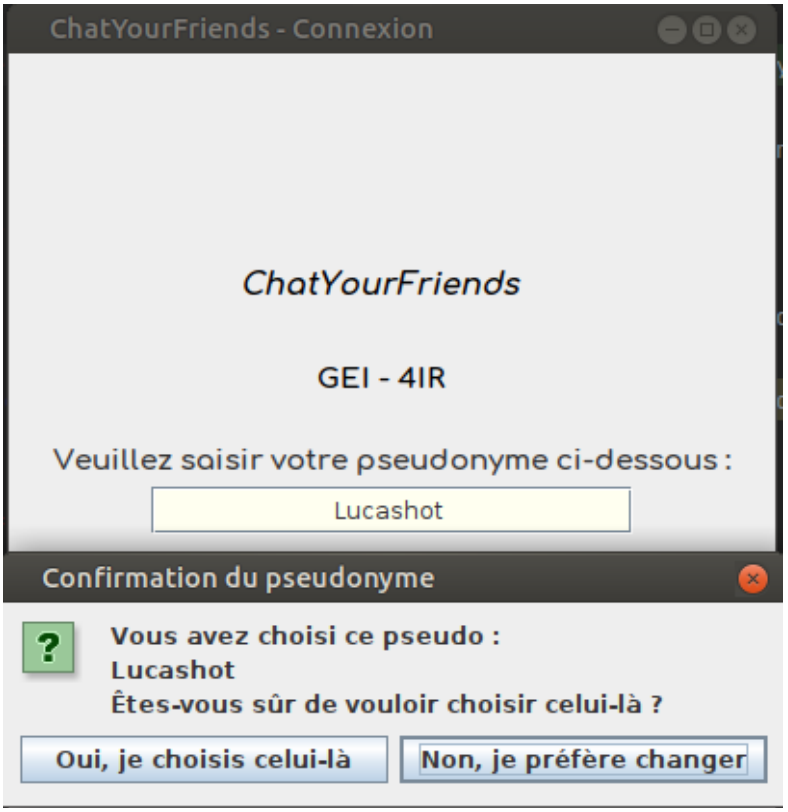
Les utilisateurs peuvent alors lancer l'application à l'aide de la commande suivante :

java -jar ChatYourFriends-jar-with-dependencies.jar

VI. Manuel d'utilisation simplifié

- Interface de login

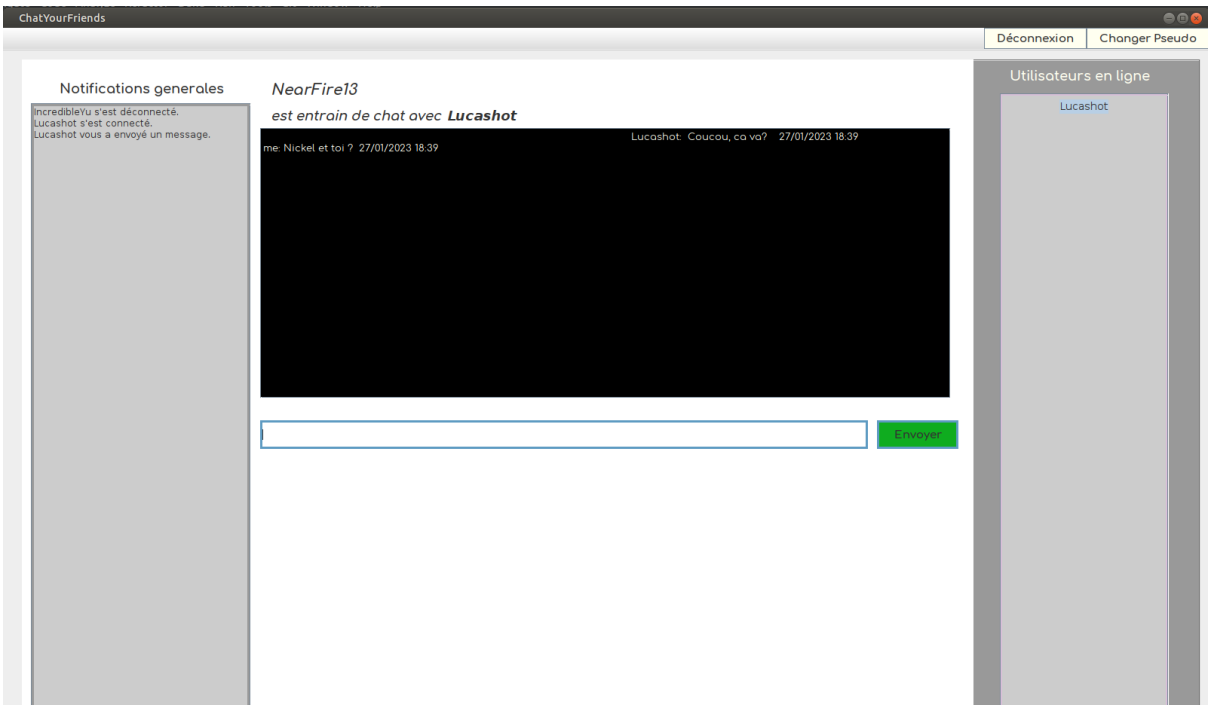
Pour la première interface "login", il suffit de taper le pseudo que l'utilisateur veut et cliquer sur "confirmer". L'interface principale s'affiche alors directement. Le nombre de caractères requis pour les pseudonyme est compris entre 6 et 12, sinon un message d'erreur apparaîtra. Après la tentative de connexion, il y aura toujours une nouvelle fenêtre qui s'affichera afin que l'utilisateur confirme que le pseudonyme choisi est correct.



Représentation de l'interface de connexion

- Interface principale

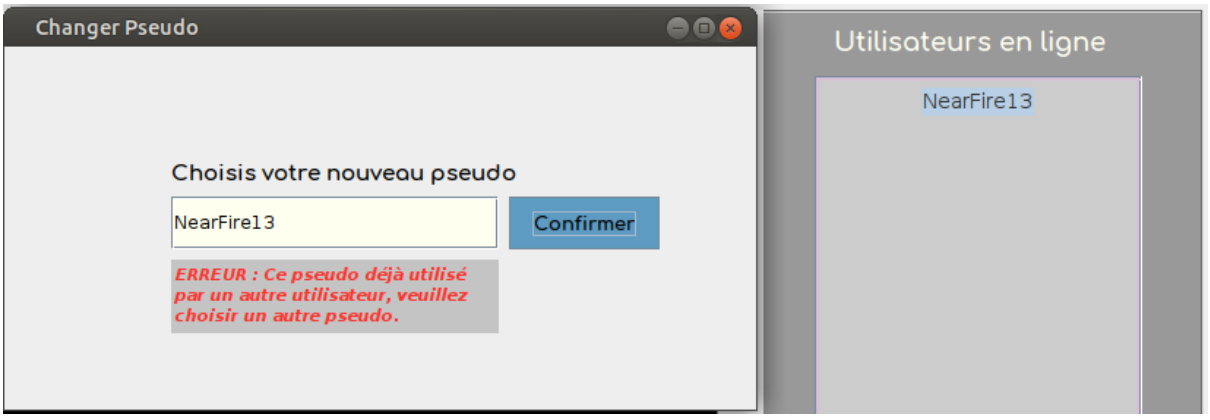
Cette grande interface permet à l'utilisateur de voir les notifications(à gauche), les personnes connectés en ligne(à droite) et l'historique entre l'utilisateur et la personne choisie. En haut à droite, il y a deux boutons Déconnexion et Changer Pseudo pour les interfaces suivantes.



Représentation de l'interface principale

- **Interface de changerPseudo**

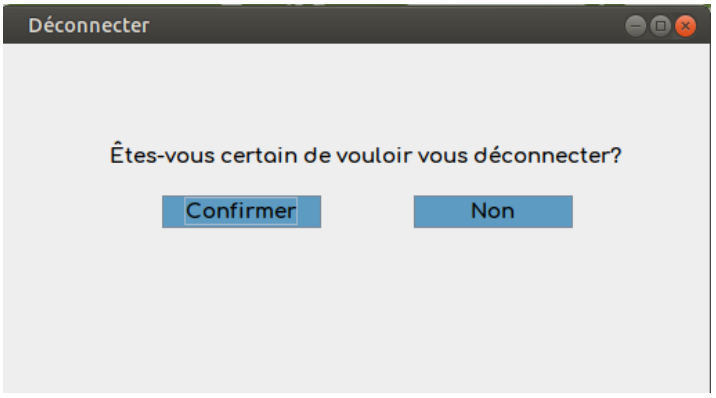
Cette fenêtre est affichée lorsque l'utilisateur clique sur le bouton "Changer Pseudo" de l'interface principale. Comme les interfaces précédentes, si le pseudo d'utilisateur est trop long/court ou s'il y a déjà le même pseudo, l'utilisateur sera invité à saisir à nouveau.



Représentation de l'interface pour changer de pseudonyme

- **Interface de déconnexion**

Cette dernière sert à quitter notre système, lorsque l'utilisateur choisit de déconnecter, nous lui demandons de faire un choix final pour confirmer.



Représentation de l'interface pour confirmer sa déconnexion

VII. Conclusion

Avec le projet de réalisation, nous utilisons les connaissances du POO, du COO et des Réseaux passés pour faire fonctionner le système comme une communication entre les deux utilisateurs. Nous utilisons également les protocoles TCP et UDP pour envoyer et recevoir des messages, pour changer de pseudo et pour charger l'historique.

Pour améliorer le projet, nous pouvons affiner l'interface si le temps le permet, bien que cette fois nous ayons fait la plupart des fonctions demandées.

Ce projet nous a permis de mettre en pratique nos capacités de travail en équipe, de gestion du temps et d'améliorer nos compétences en programmation. Cela nous aidera également dans nos études et notre futur travail.

VIII. Annexes

Le lien vers les diagrammes:
https://app.diagrams.net/#G1wqAlUXd0Xgmdm4DQGgZDIolwXDsuYUm5

INSA Toulouse
135, avenue de Rangueil
31077 Toulouse Cedex 4 - France
www.insa-toulouse.fr

Université Fédérale Toulouse Midi-Pyrénées
Liberté • Égalité • Fraternité
RÉPUBLIQUE FRANÇAISE
MINISTÈRE
DE L'ÉDUCATION NATIONALE,
DE L'ENSEIGNEMENT SUPÉRIEUR
ET DE LA RECHERCHE