

# **MANUEL UTILISATEUR CHAT SYSTEM CLAVARDAGE SYSTEM**

UN PROJET DE L'UF PROGRAMMATION ORIENTÉE OBJET

Réalisé par

- CALVILLO ABREGO, José Eduardo
- TRAN Trong Hieu
- TRAN Le Minh

Groupe 4IR à l'INSA de TOULOUSE

---

# **SOMMAIRE**

<b>I. Architecture du ChatSystem</b>	<b>2</b>
1. La conception du programme	2
a. ChatSystem, qu'est-ce que c'est ?	2
b. Quels sont les caractéristiques de ChatSystem ?	2
c. Qu'est-ce qu'il y a comme composante de ChatSystem ?	2
2. Le design de l'interface d'utilisateur	3
<b>II. Aspects techniques implémentés</b>	<b>5</b>
1. Gestion de connexion/déconnexion des utilisateurs	5
2. Envoi/réception des messages	5
3. Gestion de la liste d'utilisateurs en ligne	5
4. Gestion des pseudonymes	5
5. Gestion des notifications	6
6. Traitement des messages textuels	7
7. Stockage des messages	8
8. Convivialité du système et de l'interface	8
<b>III. Fonctionnalités en développement</b>	<b>9</b>
1. Développement d'un serveur central	9
2. Envoi/réception des messages audios	9
<b>IV. Procédures d'utilisation</b>	<b>10</b>
1. Phase de connexion	10
2. Manipulation d'une session de clavardage	10
3. Manipulation d'une fenêtre de clavardage	12
4. Phase de déconnexion	13
<b>V. Exigence du ChatSystem</b>	<b>14</b>
<b>VI. Diagrammes</b>	<b>14</b>
A. Use Case	14
ChatSystem	14
B. Sequence	15
Afficher Liste En Ligne	15
Chat	15
Rename	16
C. Class	16
Model	16
UI	17
Data	17
Connection	17
<b>VII. Conclusion</b>	<b>18</b>

# I. Architecture du ChatSystem

## 1. La conception du programme

a. *ChatSystem*, qu'est-ce que c'est ?

*ChatSystem*, c'est un programme qui, jusqu'à la version **1.03**, assure la communication entre des membres d'une entreprise utilisant un même réseau local.

b. *Quels sont les caractéristiques de ChatSystem ?*

*ChatSystem* est conçu de sorte que tous les utilisateurs de tout niveau informatique puissent y accéder.

Pour la version **1.02**, *ChatSystem* se pose dans le contexte d'une entreprise où chaque membre a sa poste de travail individuelle connectant à un même réseau local. Ainsi, la fonctionnement se base sur le principe pair-à-pair (P2P), et toute sorte de stockage se fait localement.

Puis, pour la version **1.03**, *ChatSystem* change d'une architecture pair-à-pair vers un système centralisé avec un serveur central qui gère tous les changements du réseaux locale. On considère que cette version n'est pas ce qui est attendu et donc à partir de maintenant on parlera en faisant référence à la version 1.01.

c. *Qu'est-ce qu'il y a comme composante de ChatSystem ?*

Pour faire fonctionner les caractéristiques montrées en partie (b), *ChatSystem* est conçu avec les composantes suivantes :

Modèle d'utilisateur : permet de représenter chaque utilisateur de *ChatSystem* selon le principe P2P, où chaque poste de travail peut être à la fois client et serveur, sans passer par un serveur central.

Service de connexion : garantit l'interconnexion des utilisateurs, leur accès au système et notamment le transport des messages entre des utilisateurs.

Modèle de donnée : assure la persistance des messages.

Interface d'utilisateur : permet l'interaction entre l'utilisateur et le système.

Utilité : permet d'améliorer de différents aspects des messages.

## 2. Le design de l'interface d'utilisateur

Pour assurer une interface la plus simple possible pour les utilisateurs de tout niveau informatique, le paquetage Swing (*en Java : javax.swing*) est employé, grâce à ses nombreux composants qui possèdent des fonctions étendues et des mécanismes de gérer des événements tout en proposant une apparence graphique qui emploie le style des systèmes d'exploitation familières, tel que Windows.

*ChatSystem* possède au total 4 interfaces graphiques utilisateurs, qui sont présentés ici et dont leurs fonctionnalités seront expliquées dans la suite du manuel.

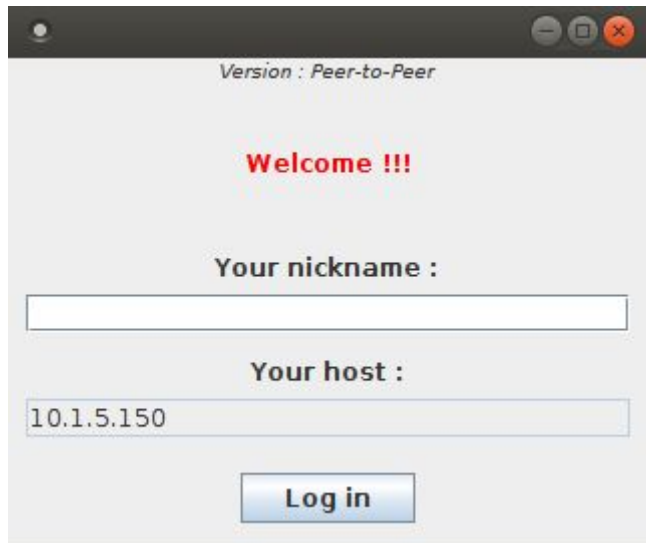


Fig1. Fenêtre de login

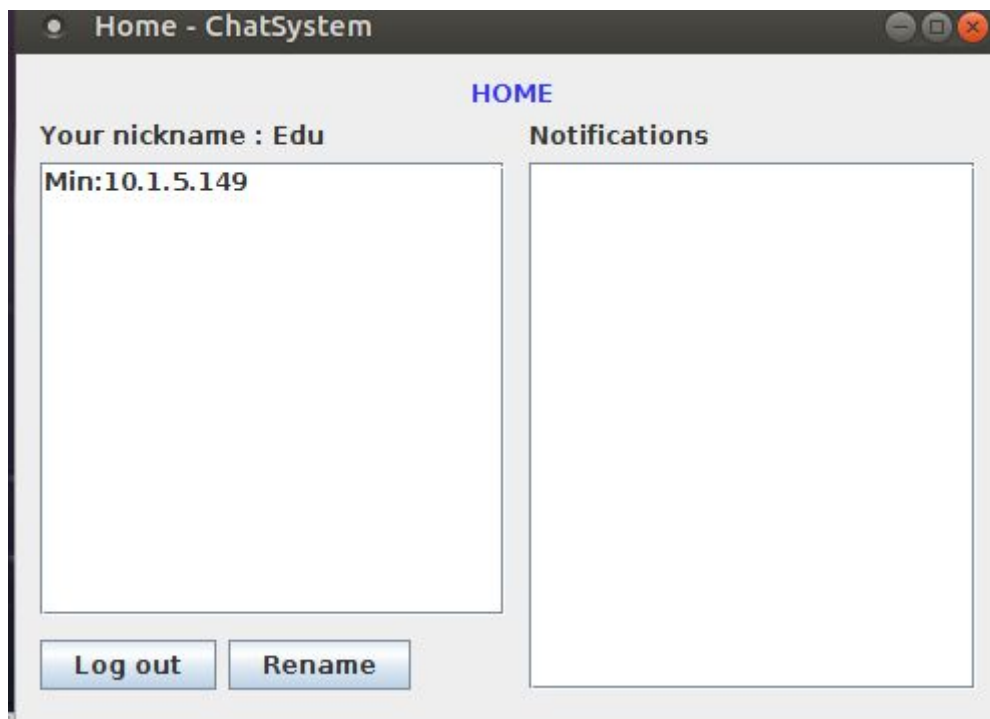


Fig2. Fenêtre Home

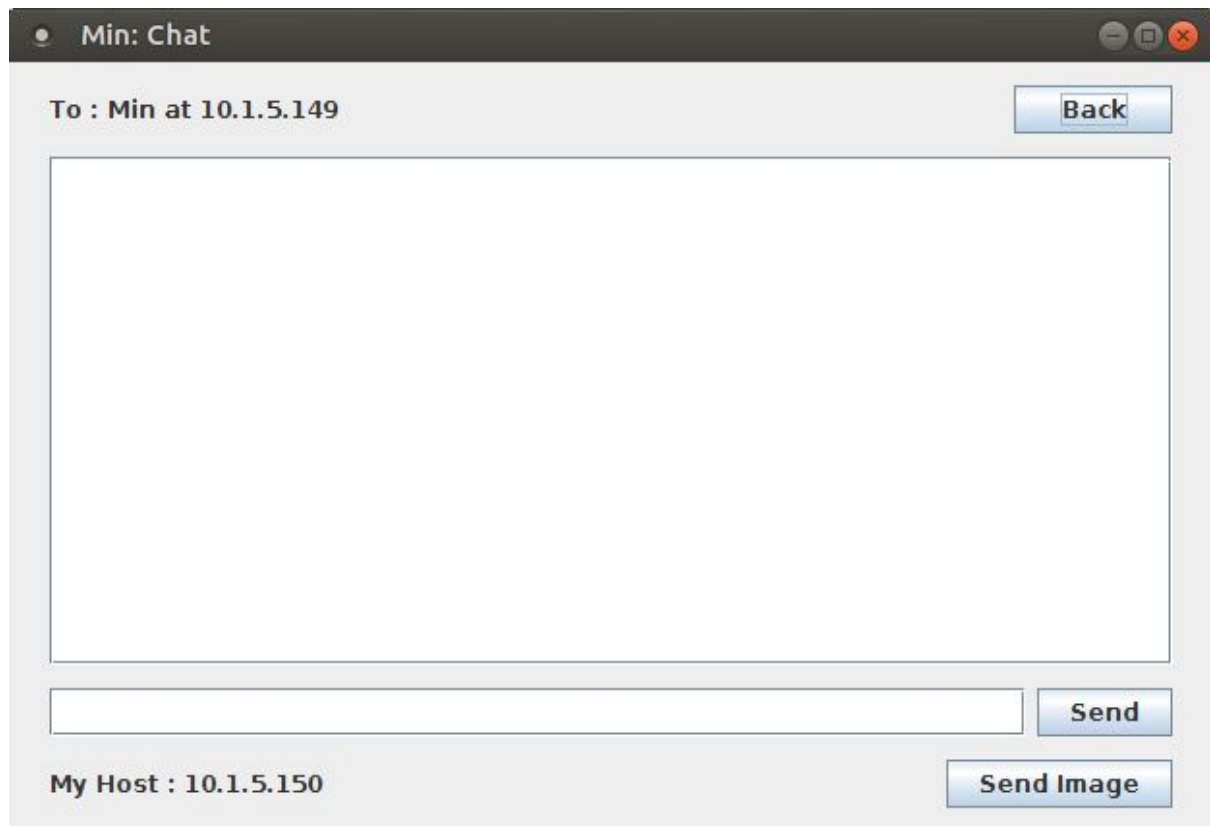


Fig3. Fenêtre de Chat

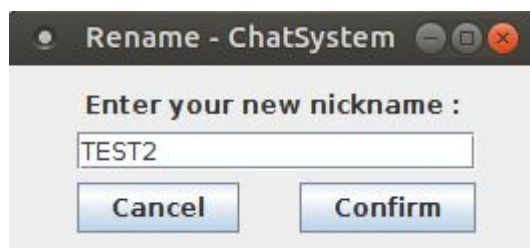


Fig4. Fenêtre de changement de pseudonyme - Rename

## II. Aspects techniques implémentés

### 1. Gestion de connexion/déconnexion des utilisateurs

*ChatSystem* vise une communication au sein d'un réseau local selon le modèle P2P, ainsi la connexion et déconnexion des utilisateurs sont gérées via des messages Broadcast UDP: à chaque fois qu'un utilisateur fait un login, il va, automatiquement, envoyer un message UDP à tous les utilisateurs en ligne à ce moment-là pour mettre à jour leur table des utilisateurs en ligne. Idem pour la déconnexion.

### 2. Envoi/réception des messages

*ChatSystem* utilise le service TCP dans le transport des messages, à l'envoi comme à la réception.

### 3. Gestion de la liste d'utilisateurs en ligne

La liste d'utilisateurs en ligne, se trouvant dans la fenêtre *Home*, est gérée de façon automatique: en fonction du "signal" reçu - sous forme des messages broadcast UDP, que cela soit un signal de connexion, de déconnexion ou de changement de pseudonyme, la liste d'utilisateurs en ligne va voir ajouter, enlever ou mettre à jour automatiquement l'utilisateur qui a envoyé ce signal.

En bref, la liste d'utilisateurs en ligne assure d'afficher correctement les utilisateurs en ligne, non déconnecté, ainsi que leurs pseudonymes actuellement utilisés.

### 4. Gestion des pseudonymes

A chaque login, un utilisateur a le droit de choisir un pseudonyme à sa volonté, sous le réserve que le pseudonyme choisi soit unique. En effet, le mécanisme de signal broadcast UDP permet à l'utilisateur faisant le login de pouvoir vérifier, sur l'ensemble des utilisateurs en ligne dans le réseau local interne, si son pseudonyme choisi est unique. Tant que l'unicité du pseudonyme n'est pas satisfait, l'utilisateur ne peut entrer dans le système, à savoir la fenêtre *Home*, et reste dans la fenêtre *Login* avec un message de warning.

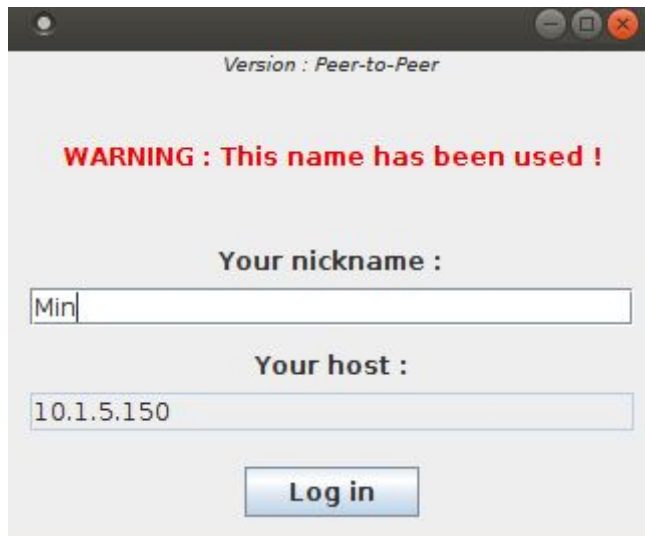


Fig5. Message de warning dans la fenêtre *Login*

Une fois accédé au système, l'utilisateur a également le droit de changer son pseudonyme, à condition que ce nouveau pseudonyme soit unique et différent à l'ancien pseudonyme. En cas de non satisfaction des contraintes, les messages de warning seront affichés, sur la fenêtre *Rename*.

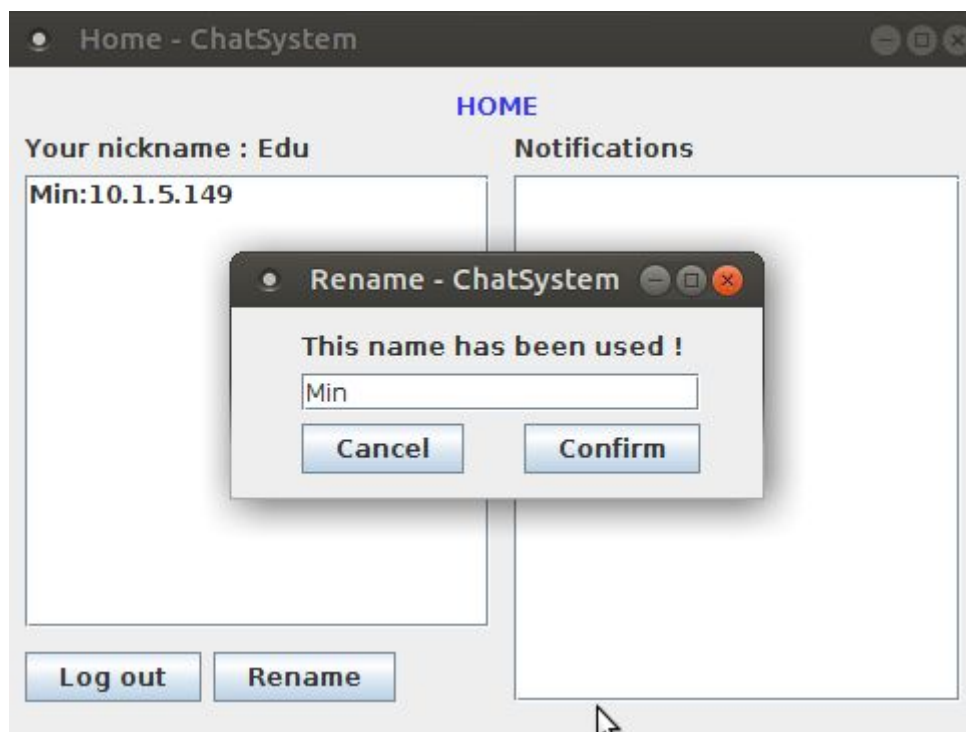


Fig6. Message de warning dans la fenêtre *Rename*

## 5. Gestion des notifications

Le champ de notifications se trouvant dans le fenêtre *Home* sert à informer, pour l'instant - la version **1.01**, à l'utilisateur du changement de pseudonyme des autres utilisateurs dans le réseau. A chaque réception d'un signal de changement de pseudonyme, le champ de notifications va faire afficher une notification correspondante.

De plus, quand l'utilisateur reçoit des messages qui ne sont pas encore lus, il y aura une marque [!] à côté du pseudonyme de l'expéditeur de ces messages non lus.

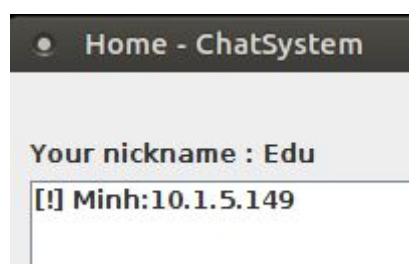


Fig7. Notification de nouveau message.

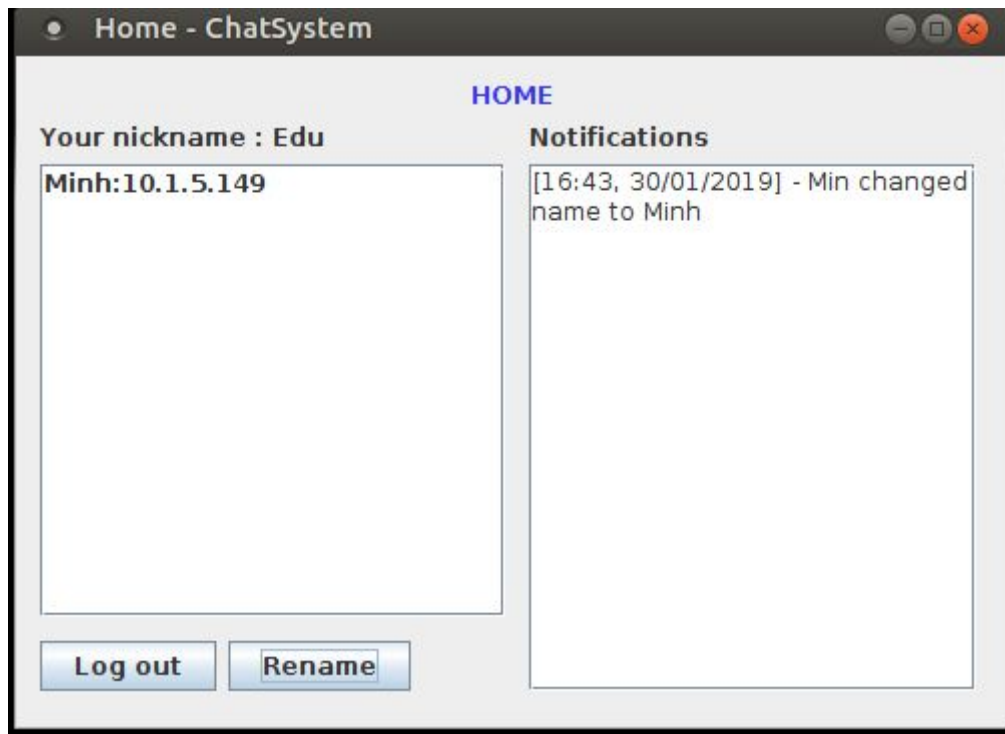


Fig8. Exemple de notification de changement de pseudonyme

## 6. Traitement des messages textuels

*ChatSystem* version **1.01** supporte actuellement des messages textuels et visuels. A la réception d'un message, le système va distinguer entre un message textuel ou un image. Pendant une conversation, tous les messages textuels seront affichés sur le champ de messages de la fenêtre de *Chat*, tandis que, lors de la réception d'une image, elle sera affichée directement sur l'écran de l'utilisateur, plus une notifications sur le champ des messages de *Chat*.



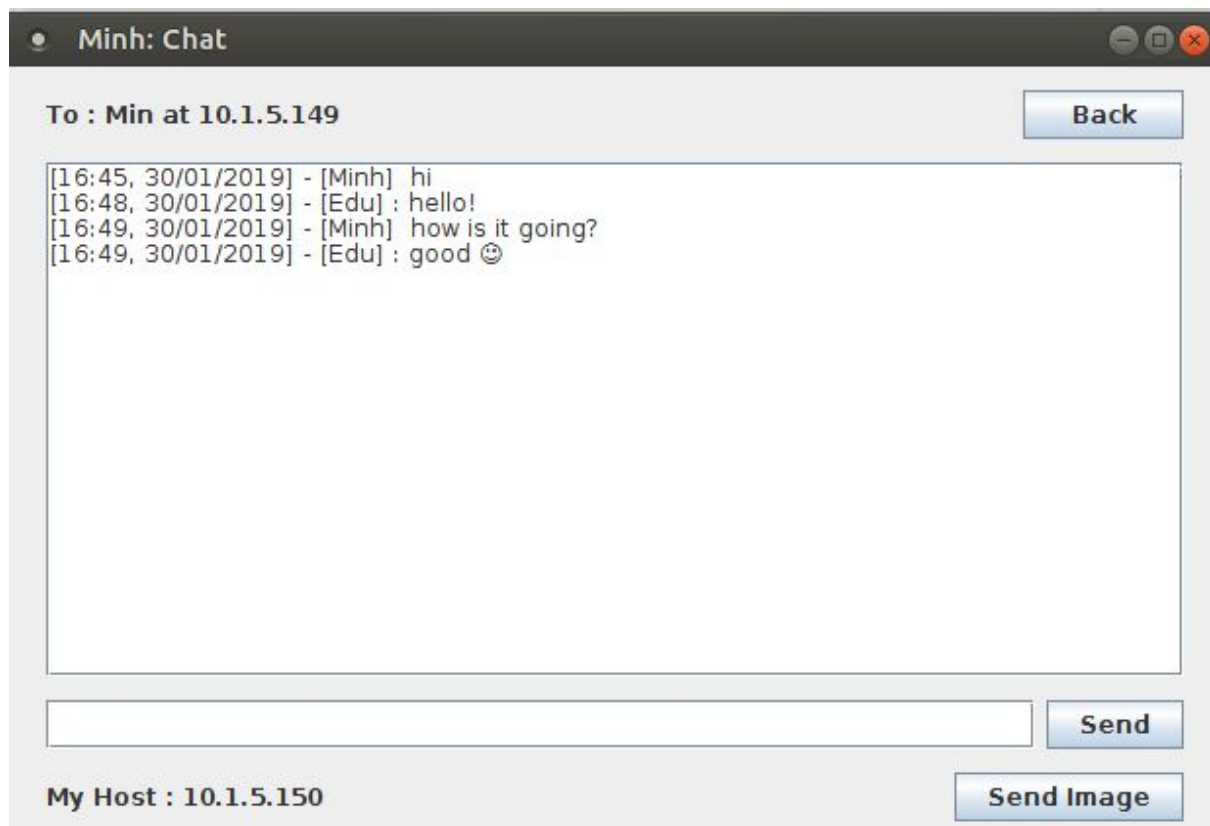


Fig9. Exemple d'une conversation


## 7. Stockage des messages

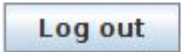
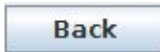



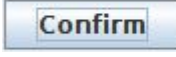
Pour la version **1.01**, *ChatSystem*, supportant la communication des utilisateurs d'une entreprise où chacun possède une poste de travail individuelle, le stockage de messages fonctionne donc localement et en fonction de l'adresse host de chaque utilisateurs.

Pour la persistance des messages, *ChatSystem* est capable de sauvegarder les conversations, afficher au plus 1000 messages récemment reçus et envoyé à chaque démarrage d'une session de clavardage.

Pour la technologie, *ChatSystem* utilise SQL comme base de donnée pour stocker les messages.

## 8. Convivialité du système et de l'interface

L'interface d'utilisateur, conçu avec Swing, est très simple pour les utilisateurs de tout niveau informatique. Ces interfaces sont dotées, d'une part, des composants utiles tels que des boîtes de textes, utilisées pour la liste des utilisateurs en ligne, affichage des notifications et affichages des messages textuels pendant une conversation, qui sont claires et lisibles; des boutons poussoirs, dont l'étiquette est rédigée en anglais simple pour faciliter la compréhension de tout utilisateur, qui prennent en charge de différents événements, ainsi que compléter la bouton fermer  des fenêtres des systèmes d'exploitation Windows,

Linux ou Unix - à savoir  de *Home*,  de *Chat* et  de *Rename*. Les fenêtres de *ChatSystem* possèdent également des boutons *Réduire*, *Minimiser/Maximiser* et *Fermer* . Pendant l'utilisation du système, l'utilisateur peut aussi employer la touche *Entrée* du clavier  pour faciliter la manipulation de certaines étapes : par exemple l'envoi des messages dans *Chat*  et la confirmation de changement de pseudonyme dans *Rename* .

De plus, tous les messages textuels sont horodatés. Par ailleurs, cherchant à créer une ambiance amicale et amusant pour les utilisateurs, tous les émoticons seront traités à l'affichage des messages textuels et convertis en symbole unicode correspondant - à découvrir tous ces émoticons !



Fig10. Exemple d'horodatage de message et traitement d'émoticon

### III. Fonctionnalités en développement

#### 1. Développement d'un serveur central

On souhaite construire un serveur central afin de stocker les comptes des utilisateurs ainsi que de bien gérer ses statuts (par exemple, "en ligne", "hors ligne", "ne pas déranger",...) .Ce serveur sera implémenté en tant qu'un servlet permet les clients de connecter au système via les non-HTTP sockets et HTTP.

=> La technologie conseillée est d'utiliser Java HTTP Servlet.

De toute façon, on a tenté d'implémenter une serveur centrale selon le principe en dessus (ChatSystem Version 1.03). Cette version fonctionne avec un serveur centrale qui écoute les changements du réseau; ces changements peuvent être du type connection/déconnection ou une modification du pseudonyme. Chaque fois que quelqu'un se connecte, il s'abonne au serveur et puis le serveur lui enverra les informations des autres abonnés. Pourtant, on pense qu'il n'est pas la même chose qu'un servlet, et on a besoin de développer plus cette partie.

#### 2. Envoi/réception des messages audios

Pour cette partie, c'est le même principe comme on a envoyé des images. En fait, on va encoder les messages audios sous formes d'un string des bits et puis on va les envoyer par

le protocole TCP; une fois reçu, le message sera décodé à une extension de type audio et téléchargé dans l'ordinateur du récepteur.

## IV. Procédures d'utilisation

### 1. Phase de connexion

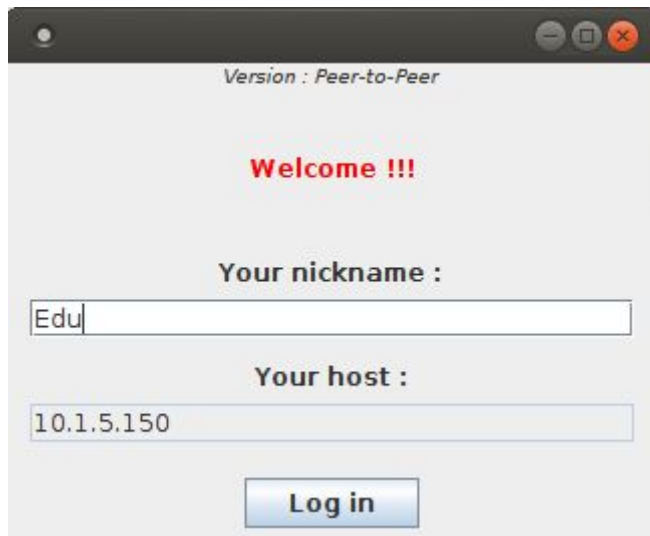





Fig11. Fenêtre Login

 **!! Attention !!** : Pour la version **1.01**, *ChatSystem* ne supporte que des sessions de clavardages **décentralisés** et **non** un **serveur central**. Par conséquent, veuillez **ne pas** remplir le champ **PASSWORD** avant le login pour garantir le bon fonctionnement du système. La fonctionnalité *Serveur Central* va être employé dans les prochaines mises à jour.

Etape 1 : Remplissez votre **pseudonyme** dans le champ **Your nickname**.

Etape 2 : Appuyez sur  pour faire le login.

**Attention !** Si le pseudonyme entré a été déjà choisi, un message de warning  **WARNING : This name has been used !** va être affiché et vous forcer d'entrer un autre pseudonyme, tant qu'il n'est pas unique.

### 2. Manipulation d'une session de clavardage

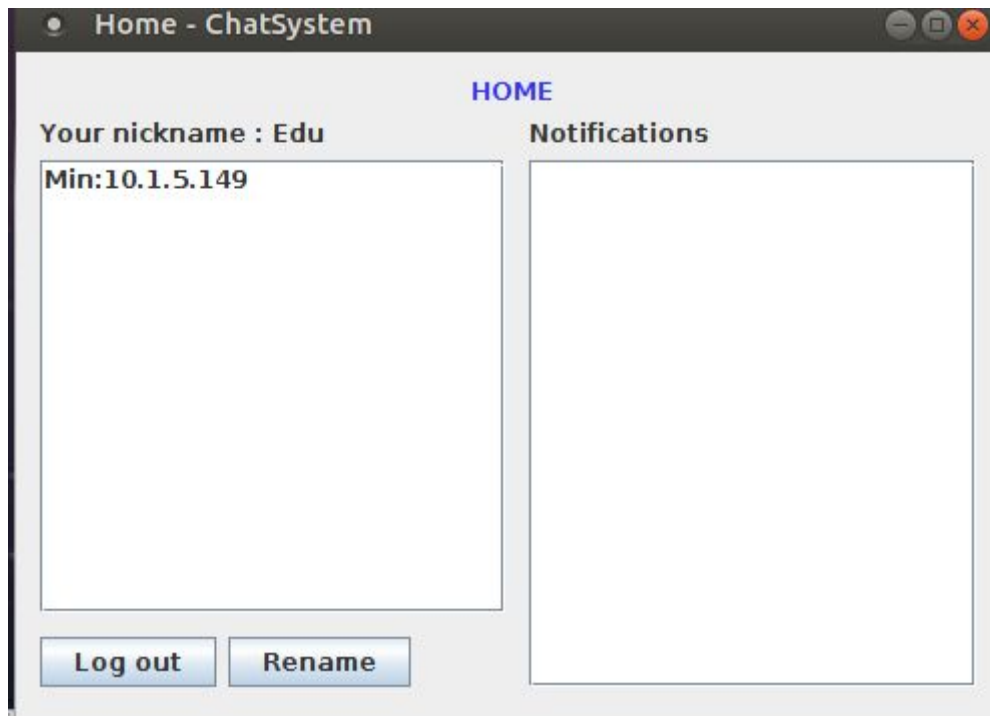


Fig12. Fenêtre *Home*

Après avoir réussi à faire le login, vous serez mené à la fenêtre *Home*, où se trouve la liste des utilisateurs en ligne.

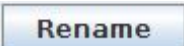


Pour démarrer une session de clavardage, veuillez choisir un utilisateur disponible dans la liste des utilisateurs en ligne qui se trouve sous votre pseudonyme (*Your nickname*)



Tout changement de pseudonyme sera informé sur le champ *Notification*



En tout moment, vous pouvez changer votre pseudonyme en cliquant sur . La fenêtre suivante va apparaître :

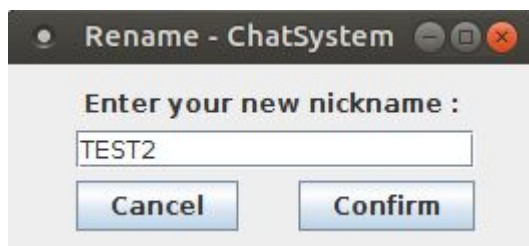






Fig13. Fenêtre *Rename*

Etape 1 : Entrez votre nouveau pseudonyme

Etape 2 : Cliquez sur  ou appuyez sur  de votre clavier pour confirmer le changement de pseudonyme. Cliquez sur  ou  pour annuler le changement.

**Attention !** Si votre nouveau pseudonyme existe déjà, un message de warning **This name has been used !** va être affiché et vous forcer à choisir un autre pseudonyme, tant qu'il n'est pas encore unique.

### 3. Manipulation d'une fenêtre de clavardage

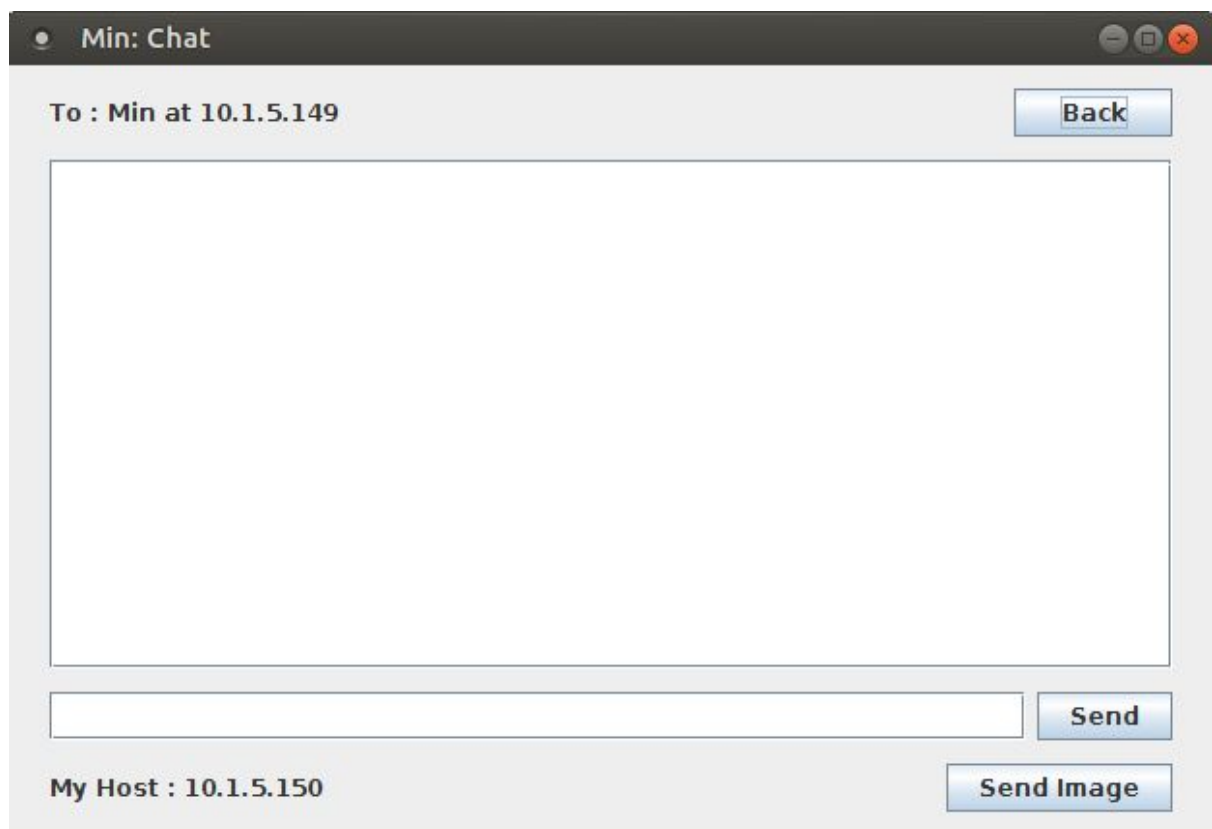






Fig14. Fenêtre de Chat

Etape 1 : rédiger un message dans le champ d'entrer le message

Etape 2 : cliquez sur  ou appuyez sur  de votre clavier pour envoyer ce message.

Pour envoyer une image, cliquez sur  puis choisir une image à envoyer.

A tout moment, pour fermer la fenêtre de Chat, cliquer sur  ou 

#### 4. Phase de déconnexion

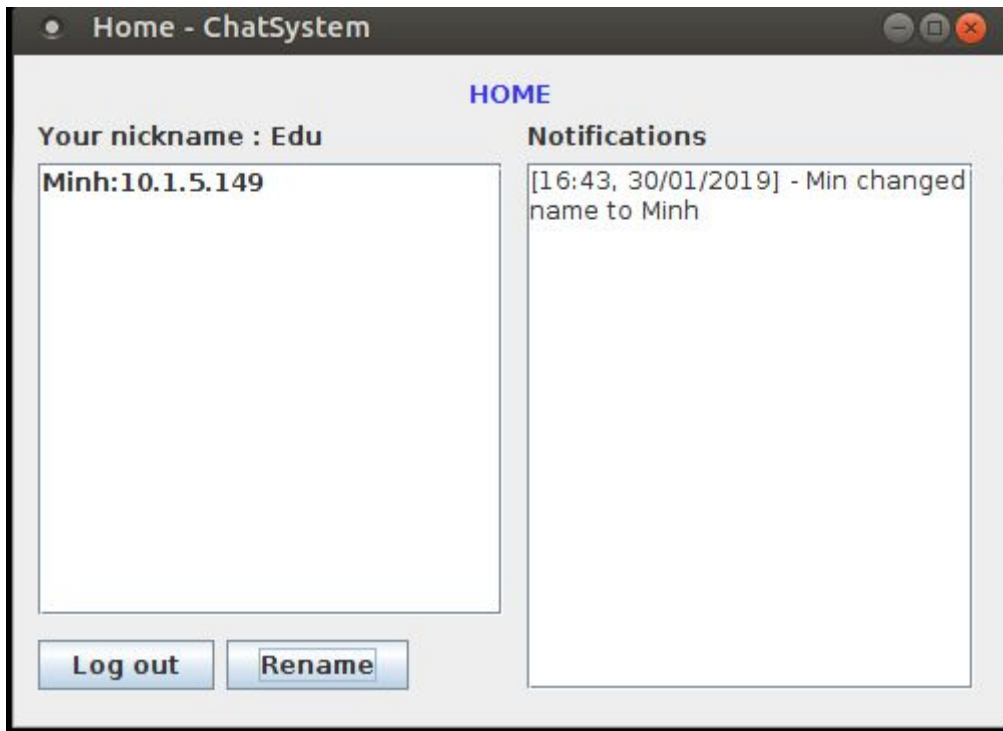
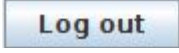

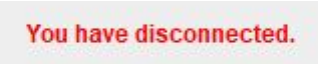


Fig15. Fenêtre *Home* avec une notification

Dans la fenêtre *Home*, pour vous déconnecter, veuillez cliquer sur  ou . Toutes les fenêtres de *Chat* actives seront fermés et vous serez ramené à la fenêtre *Login*, avec une notification .

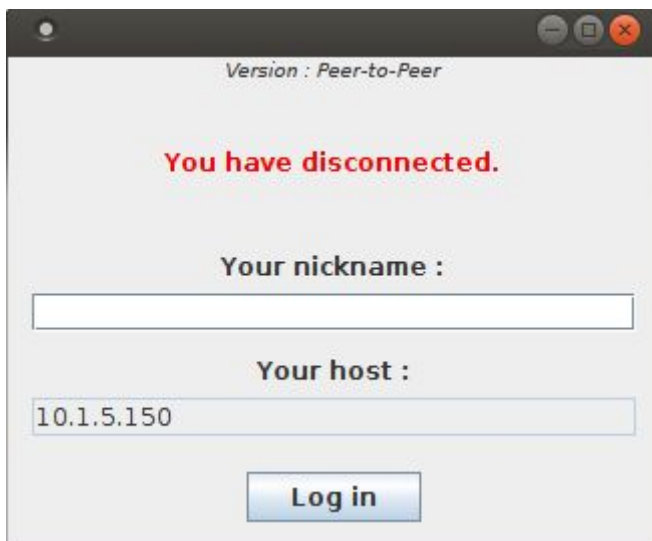


Fig16. Fenêtre de *Login* quand on log out

## V. Exigence du ChatSystem

*ChatSystem* est testé sous des systèmes d'exploitation suivants : Windows 10 et Unix. Les tests dans ces deux cas montrent un bon fonctionnement.

*ChatSystem* requiert une connexion à Internet pour pouvoir fonctionner.

### **!!Notes pour l'installation!!**

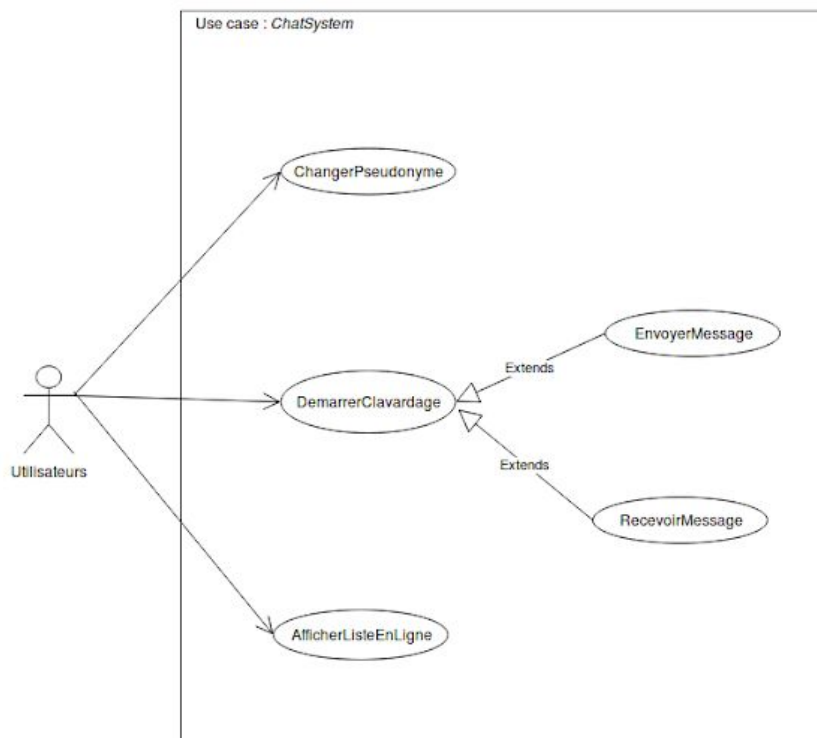
En tant qu'un programme Java, *ChatSystem* exige Java Runtime Environment (JRE) version 6 au minimum.

*ChatSystem* utilise les bibliothèques supplémentaires : Apache Commons Lang 3 et SQLite, qui sont déjà inclus dans le système.

## VI. Diagrammes

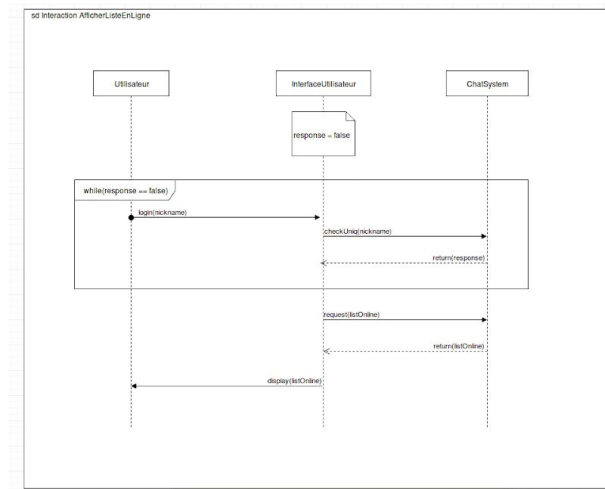
### A. Use Case

#### 1. ChatSystem

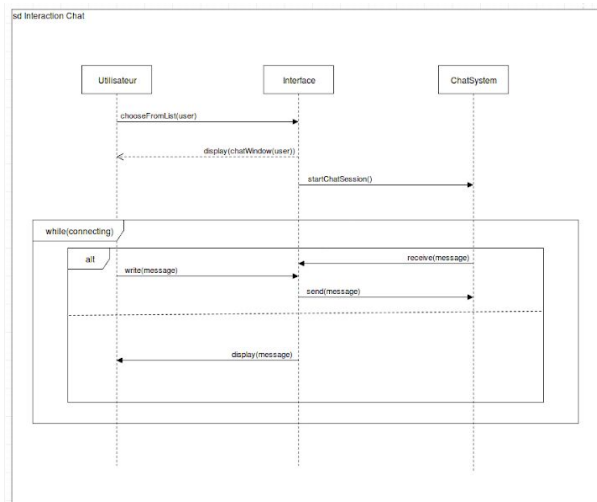


## B. Sequence

### 1. Afficher Liste En Ligne

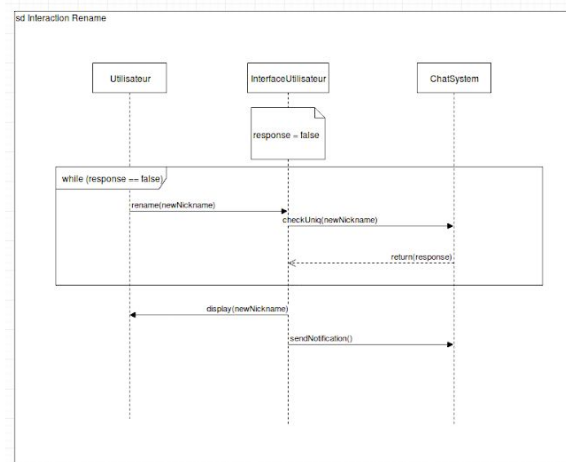


### 2. Chat



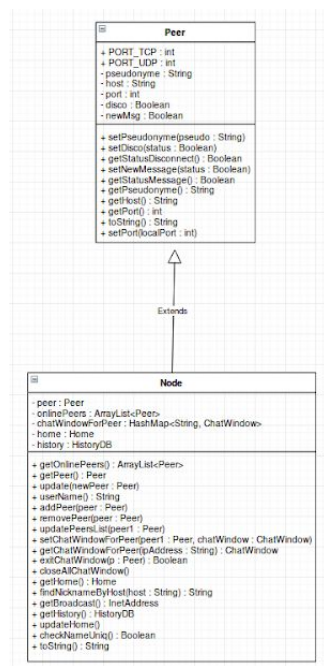


### 3. Rename

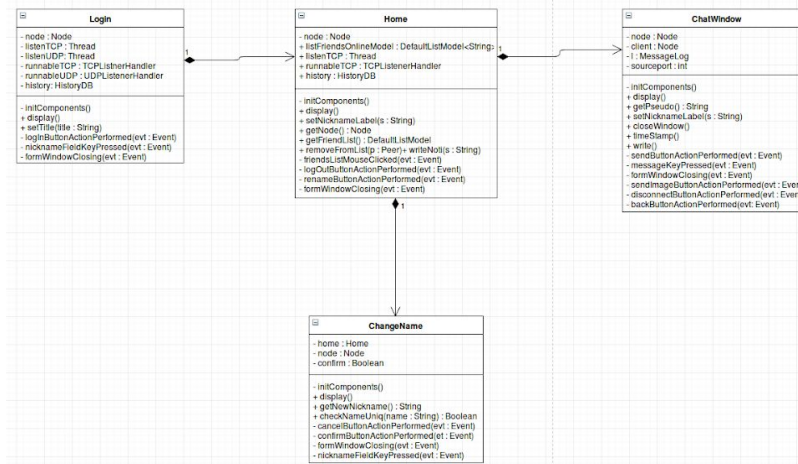


## C. Class

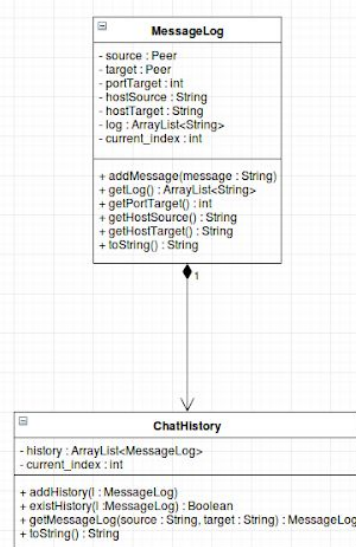
### 1. Model



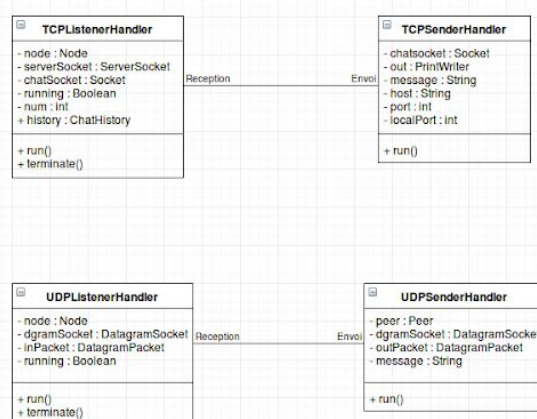
## 2. UI



## 3. Data



## 4. Connection



## VII. Conclusion

Tab 1. Tableau d'histoire des mises à jour

Version	Date de mise à jour	Description
<b>1.00</b>	24/12/2018	<ul style="list-style-type: none"> <li>• Lancement du programme</li> </ul>
<b>1.01</b>	17/01/2018	<ul style="list-style-type: none"> <li>• Amélioration de la gestion de notification : <ul style="list-style-type: none"> <li>- Les notifications de changement de nom sont horodatées et affichées dans l'ordre plus récent.</li> <li>- Ajout des notifications de messages non lus.</li> </ul> </li> <li>• Amélioration d'affichage de la liste d'utilisateurs en ligne.</li> </ul>

Tab 2. Tableau récapitulatif des fonctionnalités

Version	Modèle d'utilisateur	Service de connexion	Modèle de donnée	Interface d'utilisateur	Utilités
<b>1.00</b> - <b>1.01</b>	P2P (Client - Serveur confondu)	<ul style="list-style-type: none"> <li>- UDP pour l'interconnexion</li> <li>- TCP pour le transport de message</li> </ul>	<ul style="list-style-type: none"> <li>- Base de données SQL</li> <li>- Stockage local</li> </ul>	JAVA Swing	<ul style="list-style-type: none"> <li>- Message horodaté</li> <li>- Emoticon unicode</li> </ul>
<b>1.01</b> - <b>1.02</b>	Serveur central	<ul style="list-style-type: none"> <li>- UDP pour l'envoi des notifications (online / offline / rename) vers le serveur</li> <li>- TCP pour le transport de message</li> </ul>	<ul style="list-style-type: none"> <li>- Base de données SQL</li> <li>- Stockage local</li> </ul>	JAVA Swing	<ul style="list-style-type: none"> <li>- Message horodaté</li> <li>- Emoticon unicode</li> <li>- Envoi de notifications seulement à ceux qui sont abonnés (en lieu d'utiliser broadcasts)</li> </ul>

-Fin-