**Université Saad DAHLAB - Blida 1**

**Faculté des sciences**

**Département d’Informatique**

Master 1 : Sécurité des systèmes d'information

*Sujet :*

Réalisation d’une application de chat avec les sockets

*Présenté par :* *Encadré par :*

**HABABOU Mounir  Madame Mancer.Y**

Année universitaire: 2019/2020

**Le Sommaire:**

**- La conception:**

1. Le diagramme de classe………………………………………………………………..03
2. Le diagramme des cas d’utilisations…………………………………………………...03
3. La base des donées…………………………………………………………………….04

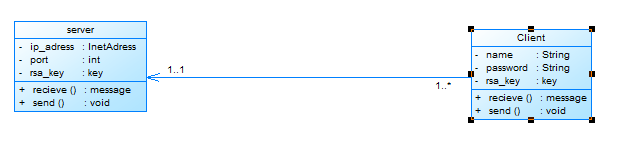
**- L’mplementation:**

1. Les interfaces…………………………………………………………………………..04
2. Le code…………………………………………………………………………………08
3. La sécurité de l’application…………………………………………………………….10

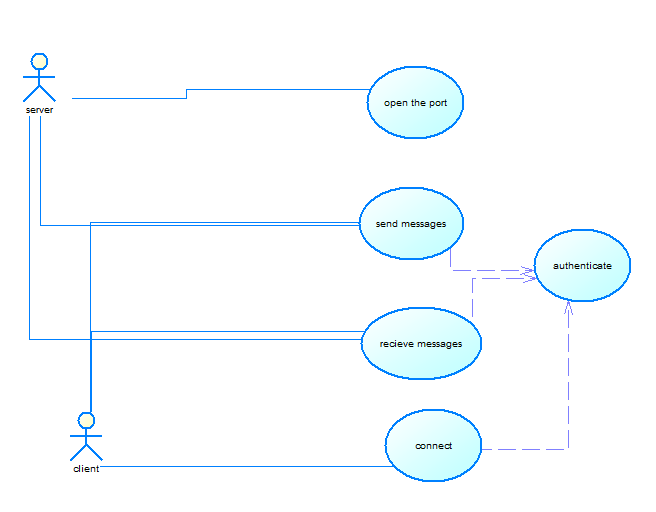
**- Les moyens utilisés**

**La conception:**

**Diagramme de classe:**

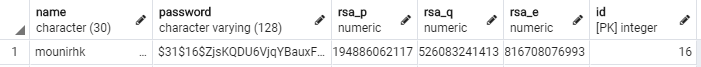


**Diagramme des cas d’utilisation:**



**La base des donées:**

On une base des données avec une seulle table, la table client :



L’implementation:

1. Les interfaces:

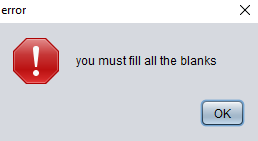
On a créé 3 interface :

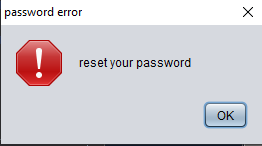
1. L’interface “create accounte” pour créer un cmpte cient elle se compose de 3 champs :

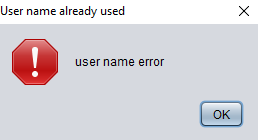
Nom d’utilisateur, mot de passe, confirmatin du mot de passe.



Si le mot de passe n’été pas bien confirmé ou bien le nom d’utilisateur est déja utilisé ou l’utilisateur n’a pa rempli tou les champs le système affichera un de ses messages d’érreur (le message d’erreur affiché depends au type d’érreur commit) :



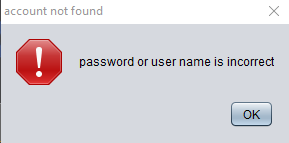


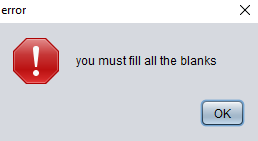


1. L’interface “log in” :

C’est la premiere interface qui s’affiche lorsqu’un client lance l’application elle permet au utilisateur à s’authentifier pour pouvoir accéder au chat :



On cas d’érreur le système affiche un de ces message :

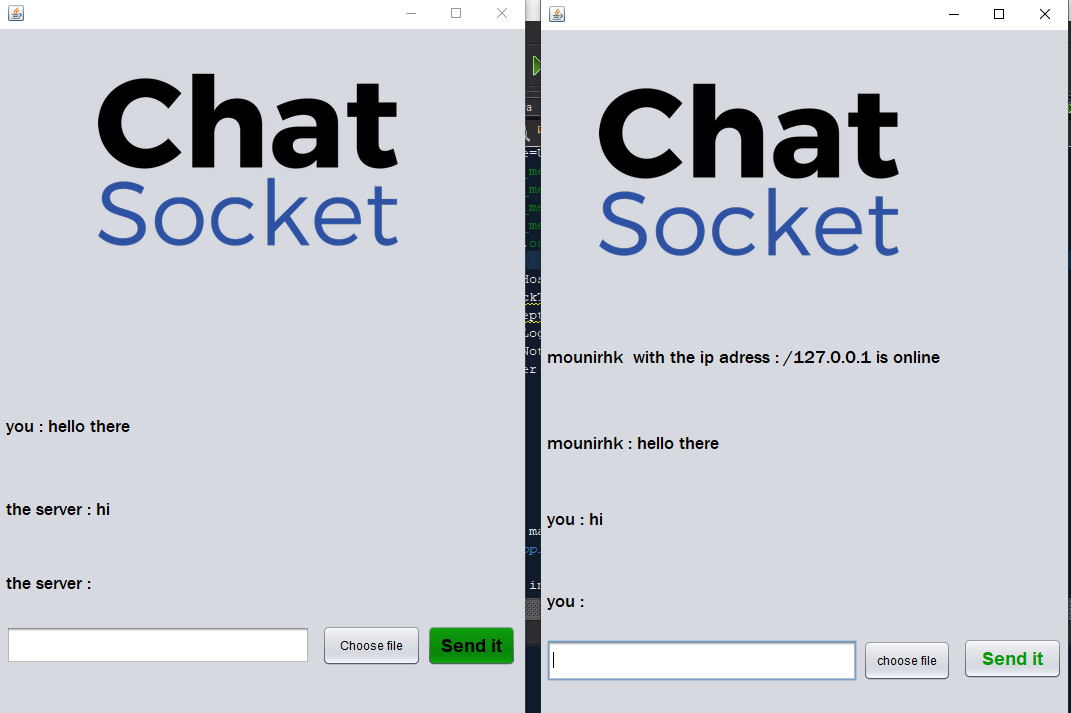


1. Les interface de chat (client ou bien serveur) permet au utilisateur à ecrire des textes et les envoyer

ou bien selectionner des fichiers ou bien des images et les envoyer :



Lorsqu’un client rejoint le chat un message de son nom et son adresse ip sera afficher, voice un d’un serveur et un clent quit sont en train de discuter dans l’adresse local “localhost” :



1. **Le code:**

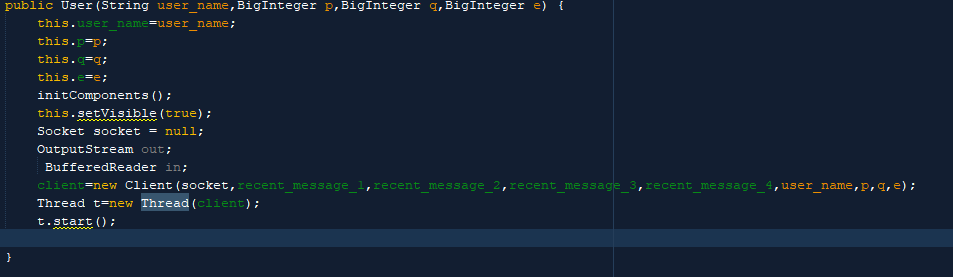
Pour le client ona deux classes:

La classe “User” chargée sur l’interface.

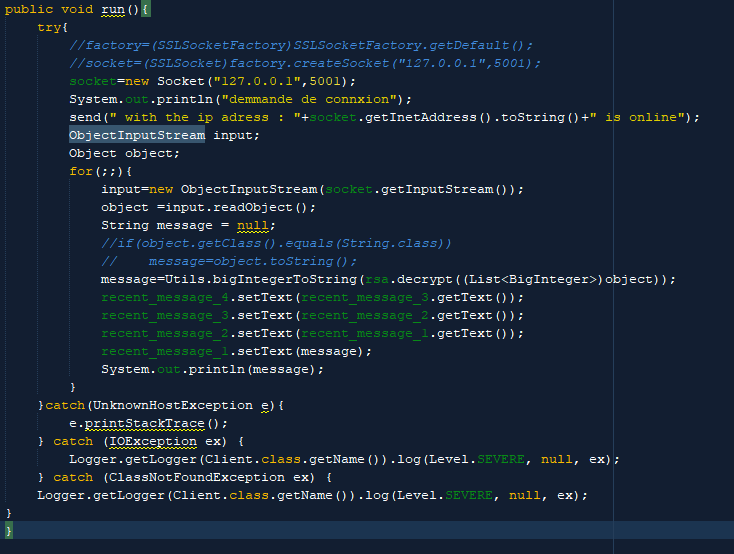
La classe “client” chargée sur la connection, l’émetre, l’envoi des messages elle implement l’interface “Runnable” est ce lance automatiquement (thread) avec le lancement de l’interface.

Et on a fait la meme chose pour le serveur deux classe, une de ces classes est chargée sur l’interface (la classe “The\_server”) et l’autre sur la connection (la classe “Serveur”), voici qulque aperçus:

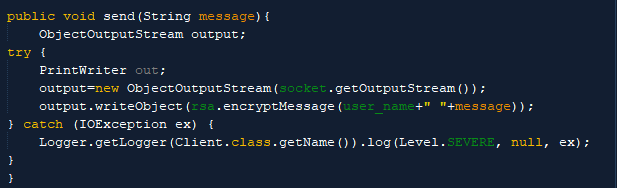
- L’appelle de la classe “Client” sur la classe “User” :



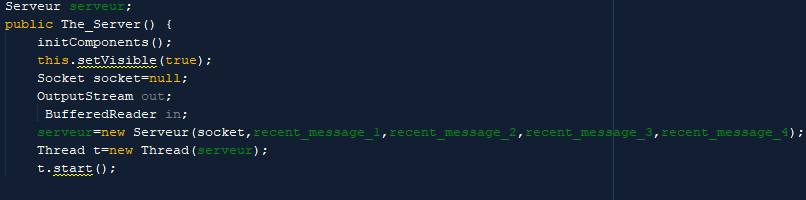
- L’implémentatoin de la méthode run de la classe “Client”:



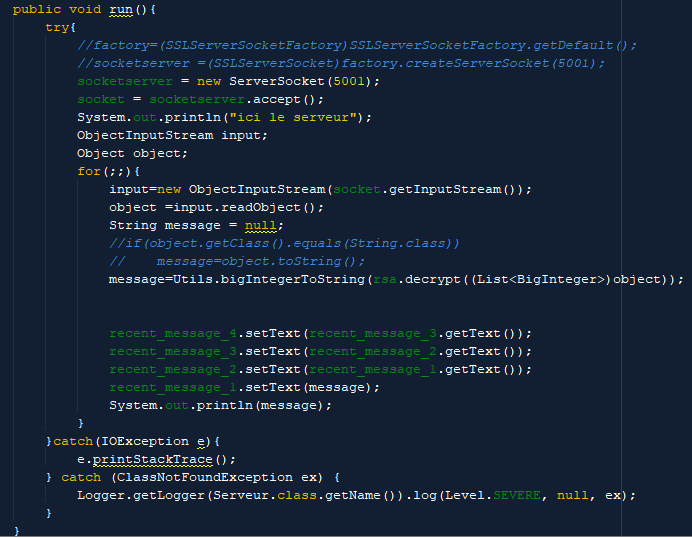
- L’implémenttion de la méthode “send” sur la classe “Client” :(permet d’envoyer les messages)



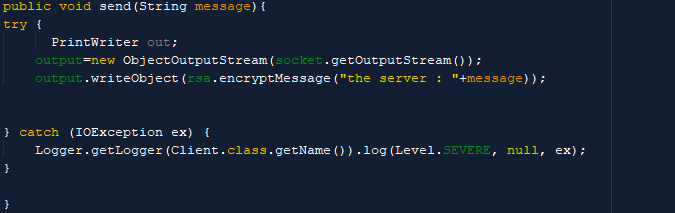
- L’appelle de la classe “Serveur” sur la classe “The\_server” :



- L’implémentatoin de la méthode run de la classe “Serveur”:



- L’implémenttion de la méthode “send” sur la classe “Servrur” :(permet d’envoyer les messages)



1. **La seécurité de l’application:**

Dans la sécurité on a trois principes fondamentaux (CAI) :

1. **La cofidentialité** :

Dans ce domaine on assurer la sécurité de mot de passe donc les mots de passe des utilisateur sont enregistrer dans la base des données d’une maniére non claire en utilisant le hashage avec l’algorithme ‘PBDFK2” pourquoi det algorithme ?

PBDFK2:

(**Password-Based Key Derivation Function 2Password-Based Key Derivation Function 2** )

Cette algorithme est permet les meilleurs algorithme de hashage, il est mieux que le md5 et le sha-256 et les autres aniciennes méthodes, PBKDF2 applique une fonction pseudo-aléatoire, comme un code d'authentification de message basé sur le hachage (HMAC), au mot de passe ou à la phrase secrète d'entrée avec une valeur de salt et répète le processus plusieurs fois pour produire une clé dérivée, qui peut ensuite être utilisée comme clé cryptographique dans les opérations ultérieures. Le travail de calcul supplémentaire rend le craquage de mot de passe beaucoup plus difficile et est connu sous le nom “Key stretching”.

On a utiliser la classe PBDFK2Hasher pour efféctuer le hashage.

1. **Integrity** (l’intégrité) :

On a utilisé le chfrement RSA pour garder le contenue des message avec chaque création d’un compte une clé se produisera et sera enregistrer dans la base de données.

On a utilisé le package “crypt” pour efféctuer le chiffrement.

1. **Availabity** (la disponibilité) :

On a essayé d’utiliser les SSLSocket au lieu d’utliser les Socket et les SSLSocketServer au lieu de SocketServer car ils sont plus sécurisé et assurent plus de sécurité sur le réseau mais on n’a pas pu les faire.

**Les moyen utilisés:**

- Language de programmation java.

- Les requetes sql.

- L’editeur netbeans.

- Les SGBD postegresql.

- Le chifrement rsa.

- Algorithme de hashage PBDFK2.

- les socket et les socketServer.

- javax.swing pour les interfaces graphique.

Voici le lien pour voir l’application:

https://github.com/mounirProgrammer/chat\_sockets\_by\_Mounir\_HABABOU