

Programmation orientée objet

I)Introduction

Nous devions réaliser un projet de chat system sur l'ensemble du semestre en binôme. Dans une première partie nous avons dû penser toute la partie conception orientée objet de notre projet. Dans un deuxième temps, nous avons programmé notre chat System que nous avons appelé BlueScarf. Un cahier des charges nous a été transmis et nous avons pu demander aux professeurs de l'affiner en fonction des questions qui nous venaient lors de la première phase de COO. Nous avons notamment la contrainte de gérer le chat system de manière décentralisée et avec une taille limitée de l'application à 50 Mo.

II)Choix techniques

Nous avons réalisé la partie COO sur ordinateur à l'aide de l'extension Eclipse Papyrus qui nous a été conseillée par M. VAN WANBEKE. Ça nous a permis de prendre en main cet outil et d'avoir une meilleure flexibilité car nous pouvions apporter des modifications tout au long du projet sans devoir redessiner l'ensemble de nos diagrammes. Nous avons réfléchi à comment satisfaire au mieux au cahier des charges notamment à l'aspect décentralisé de l'application et à la limite en taille. Nous sommes donc arrivé à deux conclusions: l'envoi de fichier étant requis nous ne pourrions jamais rester dans la limite des 50 Mo imposés en local et qu'il nous paraissait être une mauvaise direction que d'utiliser un système de broadcast. Pour cela nous avons demandé des précisions quant à la partie décentralisée et de savoir s'il serait acceptable d'avoir une communication entre les utilisateurs de manière décentralisée mais de créer une base de données pour les conversations et les fichiers échangées. Cette requête nous a été accordée et nous avons dû put mettre en place nos choix.

1)Différentes requêtes

Pour faire les requêtes nécessaires au programme de fonctionner nous avons créé des requêtes avec un en-tête spécial. Pour la phase de connexion nous utilisons l'en-tête "-c:". Nous l'appelons la requête 'connexion'. Pour contacter l'ensemble des personnes qu'un utilisateur a dans son fichier contact.txt nous utilisons l'en-tête "-u:". Nous appelons la requête 'update'. Pour l'envoi des messages lors des sessions de clavardages nous utilisons

l'en-tête "-m:" et "-s:" nous l'appelons la requête 'init conversation' et 'speak'. Nous avons choisi de toujours avoir la même longueur d'en-tête afin de nous faciliter la vie.

2) Deux fichiers pour gérer les personnes connectées au réseau

Nous avons choisi d'utiliser deux fichiers différents pour la gestion des personnes en ligne. Ces deux fichiers se situent dans le dossier configFiles et se nomment "contact.txt" et "onlineUsers.txt". Nous avons choisi de les mettre en forme à la manière d'un Csv. Dans le fichier contact.txt nous avons à la fois les noms des personnes connectées ainsi que leur adresse IP. Ce fichier est mis en forme de la manière suivante: "nom:adresseIP". Dans le fichier onlineUsers.txt nous avons seulement les noms des personnes qui sont en ligne. Ce fichier est mis en forme de la manière suivante: "nom". Lors d'une requête update, tous les utilisateurs vont mettre à jours leur fichier contact.txt puis se servent de contact.txt pour réécrire le fichier onlineUsers.txt. Nous avons choisi d'avoir deux fichiers car nous avons un démon sur la modification du fichier onlineUsers.txt ce qui ne nous permet pas de le modifier aussi aisément que contact .txt.

3)Phase de connexion

Pour allier décentralisation des conversations et non broadcast nous avons décidé que la dernière personne qui se connecte à notre système sera enregistré par la base de données. Ainsi, tout nouvel utilisateur se connectant au système va interroger la base de données, celle-ci va nous renvoyer l'adresse IP du dernier utilisateur s'étant connecté.

Le nouvel utilisateur va alors contacter le dernier utilisateur ,à l'aide d'une requête connexion. Ce dernier va l'ajouter dans son fichier contact.txt puis envoyer à l'ensemble des personnes qu'il a dans le fichier contact.txt les informations sur le nouvel utilisateur, à l'aide d'une requête update. Il va également répondre au nouvel utilisateur pour lui indiquer qui sont les autres personnes sur le réseaux.

Nous avons choisi ce système qui permet de ne pas inonder le réseau en broadcast et d'établir des connexion en TCP afin d'avoir une fiabilité dans l'acheminement des messages. Notre système permet à un utilisateur en dehors de l'entreprise de se connecter à l'application si tout le monde possède une adresse IP publique, ce qui est le cas à l'insa. Nous avons donc un système qui possède les avantages du servelt qui était demandé dans une seconde moitié du cahier des charges.

4)Double système de base de données

Nous ne souhaitons pas envoyer des requêtes à la base de données centrales lors de chaque message pour conserver un aspect décentralisé. Dans cette optique, lors d'une session de clavardage, nous avons des fichiers en local qui représentent les conversations avec chaque personne, ces fichiers sont nommés ainsi: "nomutilisateur.txt". Ce fichier est mis en forme de la manière suivante: "send: message|recv:message". send est l'en-tête qui

indique que c'est un message que nous avons envoyé et recv est l'en-tête qui indique les messages que nous avons reçus. À chaque message que nous envoyons, nous écrivons dans le fichier local le message et le message est aussi transmis à l'autre utilisateur avec une requête message. Dans la base de données centralisée, nous avons choisi de créer une table avec cinq colonnes: "Utilisateur1", "Utilisateur2", "Conversation" et deux flags qui permettent de connaître l'état de la conversation. Lorsqu'on sélectionne un utilisateur avec qui clavier, on envoie une requête à la base de données centrales qui nous envoie l'historique s'il existe des conversations avec cet utilisateur.

4) interface de connexion

Lorsqu'on démarre l'application, une interface de connexion s'affiche. Celle-ci nous demande notre identifiant et notre mot de passe. Dans le cas où l'un de ces deux derniers est faux alors une fenêtre s'ouvre pour nous indiquer que nous avons fait une erreur. Si nos identifiants sont bons alors la fenêtre principale de l'application s'ouvre et nous pouvons commencer à clavier.

5) Phase de clavier

Nous avons fait une liste sur le côté gauche avec toutes les personnes connectées et nous. Pour commencer une conversation, il suffit de cliquer sur le nom de la personne. Nous avons l'historique de la conversation qui s'affiche. Lors d'une première connexion avec un correspondant, nous avons un message de bienvenue "Welcome on BlueScarf, you can start a new chat with this user. Enjoy!". Pour se repérer dans les conversations, avant chaque message reçu, on écrit le nom du correspondant. Avant chaque message envoyé, on écrit "You :". Pour faire une redondance dans les informations, les messages reçus sont affichés sur un fond Gold et les messages envoyés sont affichés sur un fond Bleu clair. Pour envoyer un message, il faut écrire dans l'espace blanc en dessous de la conversation puis cliquer sur le bouton "Send". Lorsque nous recevons un nouveau message, une notification s'affiche sur l'écran en nous indiquant la personne qui vient de nous envoyer un message, sauf dans le cas où nous étions déjà en conversation avec cet utilisateur et dans ce cas seulement le nouveau message s'affiche.

6) Phase de déconnexion

Nous avons deux étapes lors de la phase de déconnexion. Une première partie entre l'utilisateur et la base de données centrale et une seconde avec une requête update. Dans la partie avec la base de données centrale, nous avons trois scénarios. Le premier est dans le cas où l'adresse IP enregistré dans la base de données est celle d'un autre utilisateur alors dans ce cas on ne fait rien. Dans le cas où c'est notre adresse IP, on choisit une autre adresse IP dans le fichier contact .txt. Si jamais il n'y a plus d'utilisateur dans le fichier contact.txt alors on met 'null' dans la base de données. La requête de déconnexion est envoyée à tous les contacts se trouvant dans le fichier contact.txt. Nous parcourons l'ensemble des

fichiers présents dans le dossier configFiles/conv/. Ces fichiers représentent l'ensemble des conversations qui ont été faites durant la phase de connexion de l'utilisateur. Nous récupérons la totalité de chaque fichier puis on l'envoie à la base de données pour enregistrer l'historique.

7)Interface graphique

Nous avons réalisé l'interface graphique avec java swing. Nous avons à gauche les utilisateurs connectés, au centre la conversation en cours.

A chaque fois que nous recevons une requête de type connexion, nous mettons à jour le panel qui contient tous les utilisateurs en ligne. Même chose, lorsque nous recevons un nouveau message, nous sommes soit notifié soit le nouveau message s'affiche à l'écran.

Lors de la réception d'un nouveau message, le panel contenant les messages ajuste sa taille et descend pour afficher les messages.

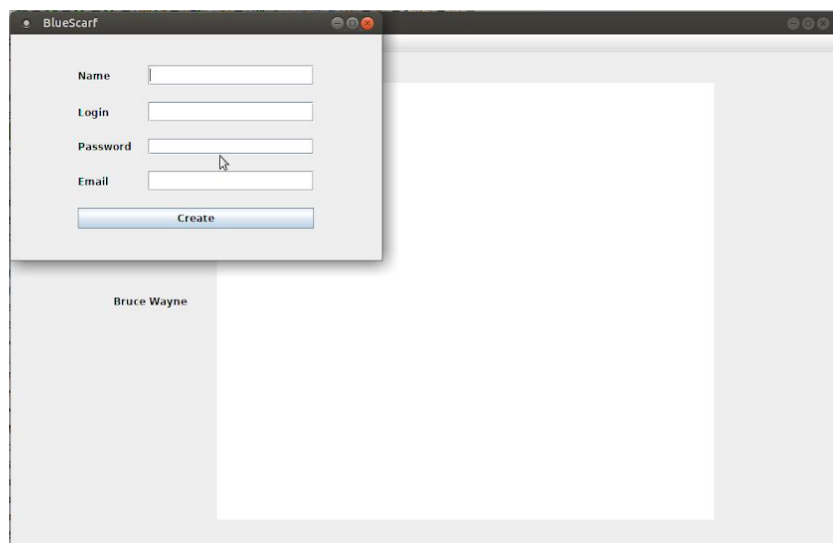
Lorsqu'un nouvel utilisateur se connecte, son nom apparaît sur le côté et la liste des noms se centrent verticalement automatiquement.

8)Partie Administrateur

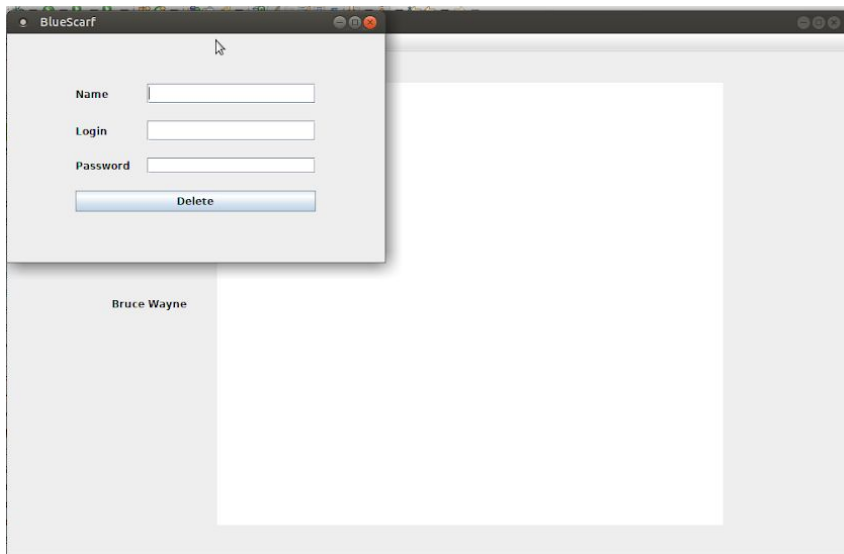
Nous avons un compte administrateur qui a des privilèges particuliers. Il peut ajouter de nouveaux utilisateurs à la base de données. Les champs à remplir sont le nom, le pseudo, le mot de passe et l'adresse e-mail.

L'administrateur peut aussi ajouter d'autres administrateurs avec les mêmes champs à remplir.

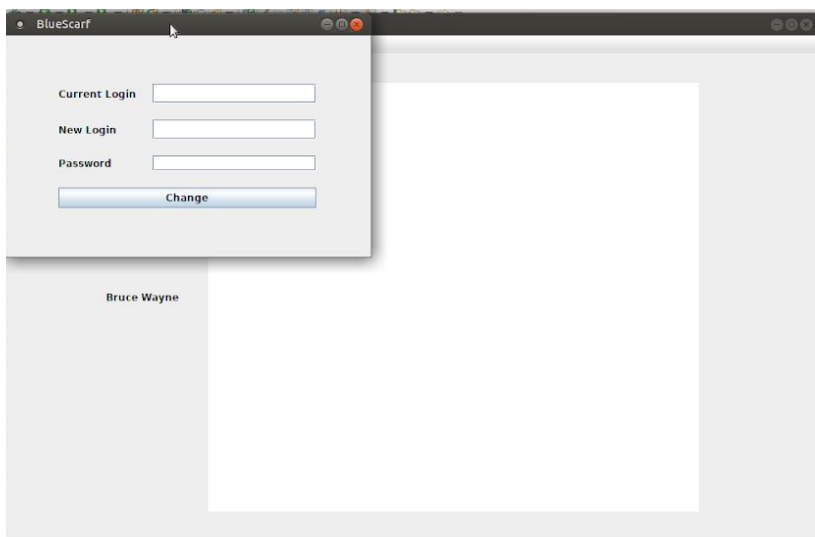
Notre nom d'administrateur par défaut est Bruce Wayne et son login est root. Son mot de passe est lamgroot.



Capture d'écran de l'interface administrateur avec la fenêtre d'ajout d'un user.



Capture d'écran de l'interface administrateur avec la fenêtre de suppression d'un user.



Capture d'écran de l'interface principale avec la fenêtre de changement de login. Cette une option disponible pour tous les users.

III) Guide d'utilisation

Nous avons quelques interdictions afin de ne pas perturber le bon fonctionnement de BlueScarf.

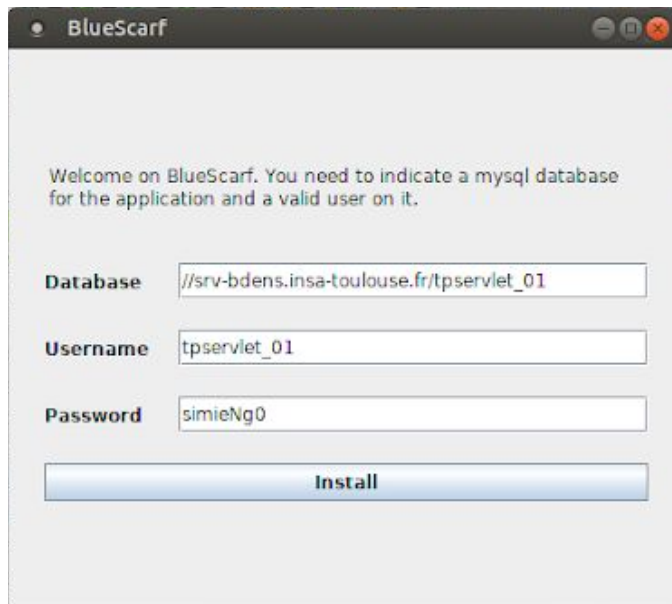
Notre système supporte les prénoms composés mais il faut avoir un nom de famille tout attaché. Il est possible d'envoyer des messages vides.

Notre base de données est sensible aux attaques de type SQL car nous ne vérifions pas les input que nous lui envoyons.

Ci-dessous nous avons plusieurs captures d'écrans qui permettent de comprendre l'utilisation de BlueScarf.

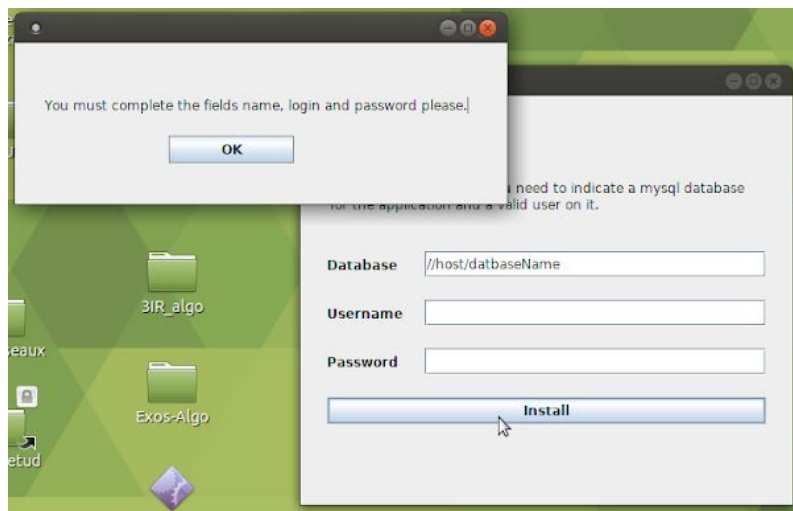
Tout d'abord, pour l'installation récupérez les fichiers présents sur la branches master du github du projet. Merci de bien lire le README présent dans les fichiers.

Une fois l'exécutable d'installation lancé veuillez rentrez les paramètres propres à votre base de données. Ici, nous avons rentré les paramètres propres à la base de données de l'insa sur laquelle nous avons travaillée.

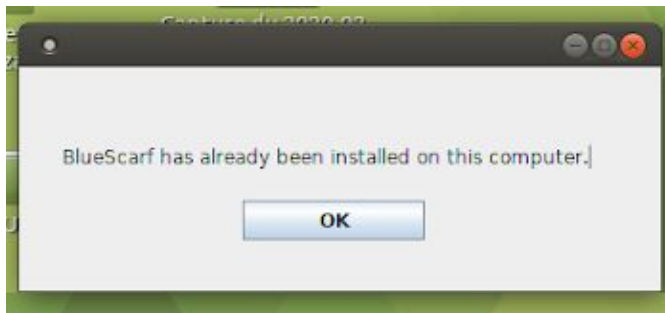


The screenshot shows a window titled "BlueScarf" with a light gray background. At the top, it says "Welcome on BlueScarf. You need to indicate a mysql database for the application and a valid user on it." Below this, there are three input fields: "Database" with the value "://srv-bdens.insa-toulouse.fr/tpervlet_01", "Username" with the value "tpervlet_01", and "Password" with the value "simieNg0". At the bottom, there is a blue "Install" button.

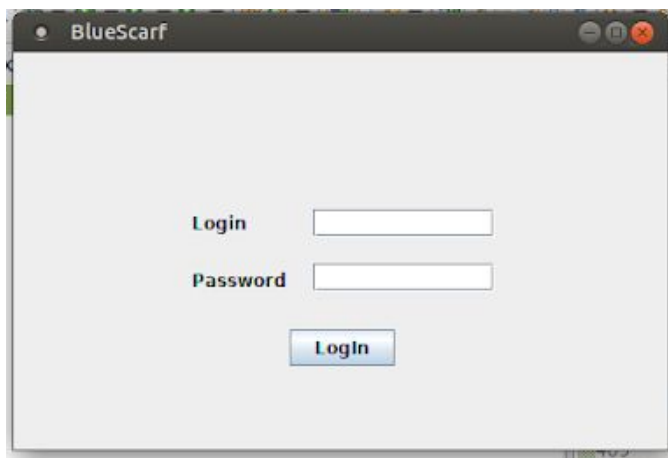
Il y a plusieurs messages d'erreurs possibles à cette étape. Le premier concerne une erreur lors de la saisie des identifiants.



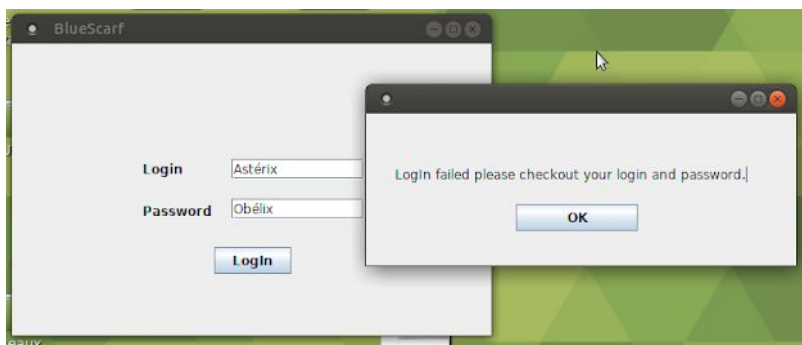
La seconde est utilisée si on essaie de réinstaller BlueScarf sur une machine sur laquelle il est déjà installé.



Une fois l'installation complétée, on peut lancer l'exécutable de BlueScarf et la capture d'écran suivante s'affiche. Comme la base de données vient d'être créée, il faut utiliser le compte administrateur par défaut qui est login root et mot de passe lamgroot.



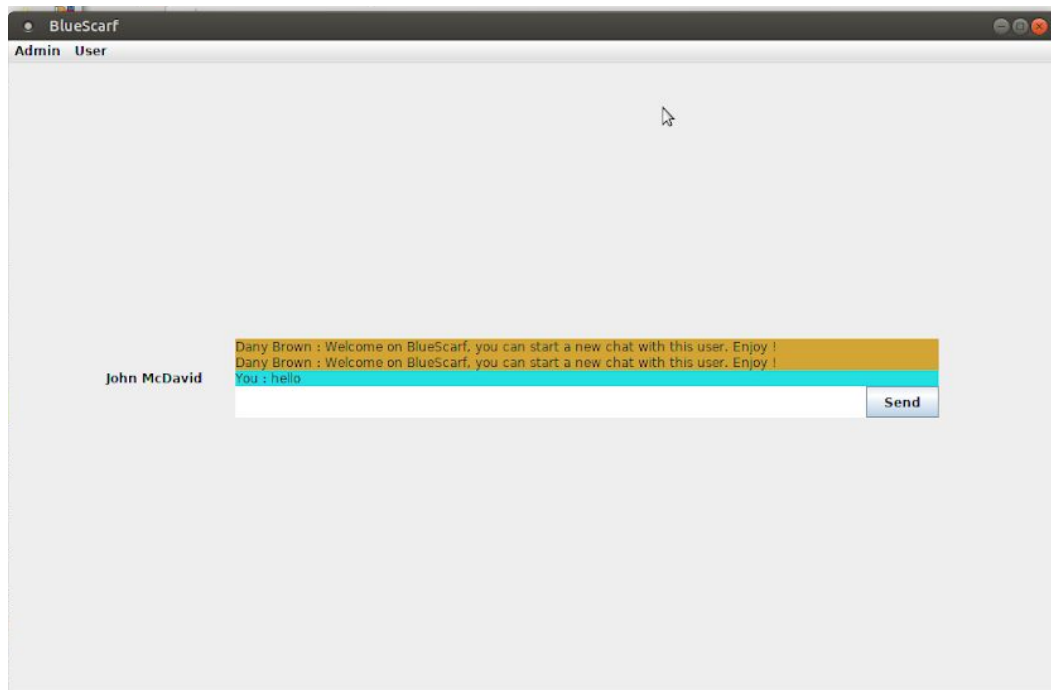
Si l'on fait une erreur lors de la saisie un message d'erreur s'affiche



A partir de là, il faut créer les utilisateurs que l'on souhaite. Pour cela vous devez vous rendre dans le menu Admin en haut à gauche et cliquer sur 'Add User'. Renseignez les champs et l'utilisateur est automatiquement créé.

Désormais, nous avons donc tous les utilisateurs de créer et nous souhaitons commencer une session de clavardage. Pour cela, il suffit de cliquer sur un des noms

d'utilisateur connectés affichés sur le côté droit. On se retrouve avec la fenêtre suivante



On peut distinguer les différents messages envoyés et reçus ainsi qu'une zone de rédaction des messages avec le bouton pour les envoyer.

Lorsqu'on reçoit un message d'un utilisateur autre que celui avec lequel on clavarde, une notification apparaît avec le nom de l'envoyeur.

