Rapport chat-book

Projet COO - M1MIAGE-FA 2016-2017

Laurent THIEBAULT & Ludovic LANDSCHOOT

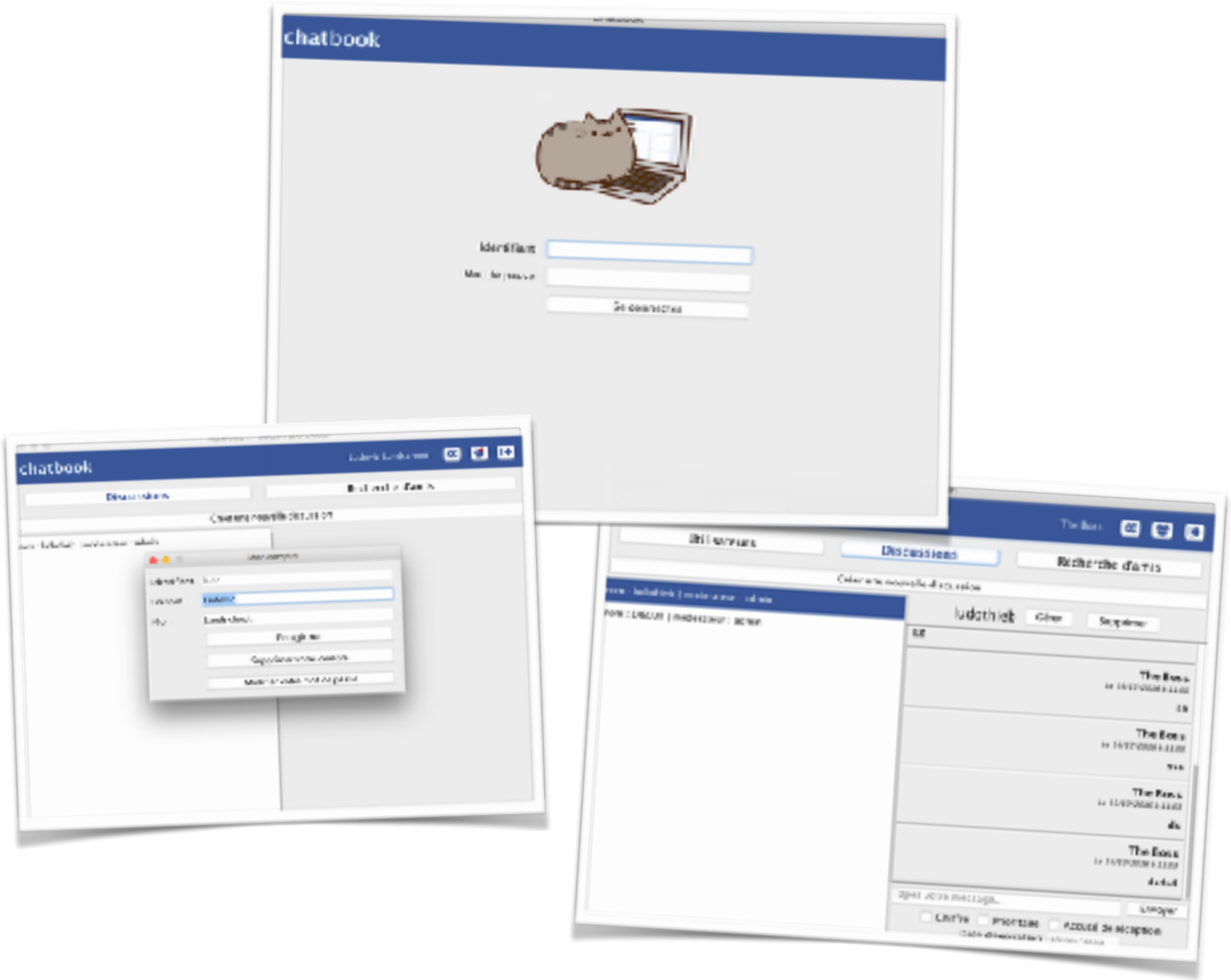


Table des matières

Présentation générale

Le principe général consiste à réaliser une plate-forme de discussion / forum / réseau social.

Pour utiliser l'application, les utilisateurs devront s'identifier avec leur compte (et un mot de passe). Ils pourront ensuite s'ajouter en ami, et s'envoyer des messages (entre amis uniquement).

Puisque l'application ne permet pas d'avoir plusieurs utilisateurs simultanément, le système de messagerie fonctionne en différé.

Chaque utilisateur peut également créer un groupe de discussion, il en devient automatiquement le modérateur, et il peut ensuite ajouter ou supprimer des utilisateurs.

Il existe un compte unique nommé Administrateur qui dispose de droits étendus : créer, modifier ou supprimer des comptes utilisateurs, modifier les groupe de discussion, etc.

Architecture du projet

Notre projet utilise une architecture classique avec un modèle en couches :

- **gui** : *IHM - Intéraction de l'utilisateur en Swing et appels de services*
- **service** : *Traitements métiers de l'application - Communication controller / mapper*
- **persistence** : *Accès à la base de données - Transformation en objet Java*
- **domain** : *Objets métiers*
- **resources** : *Ensemble des ressources utiles à l'application :*
  - Configuration de la base de données
  - Images
  - Schéma, jeu de données et requêtes SQL.

Détail des cas d'utilisation

Voici la liste des cas d'utilisation pris en charge par notre application :

- **CU 1** : Création et suppression de compte utilisateur
  - *Dans la partie GUI via l'AccountsPanel on peut ouvrir une CreateAccountFrame. Celle-ci va appeler le UserService qui utilisera le UserMapper pour créer l'utilisateur en base. On peut également supprimer un utilisateur via ce cheminement.*

- **CU 2** : Modification de compte utilisateur
  - *Dans le cas d'une modification de son propre compte : la partie GUI on ouvre une UpdateAccountFrame qui va appeler le UserService qui utilisera le UserMapper pour mettre à jour les informations du compte.*
  - *Dans le cas d'une modification de compte via l'admin : Un Unit Of Work est mis en place pour sauvegarder les modifications sur un ensemble d'utilisateurs et sera sauvegarder en base qu'au déclenchement du bouton Enregistrer. Pour information, les utilisateurs sont chargés grâce à un système de Virtual Proxy, ils ne chargent pas directement la liste de leurs amis.*

- ***CU 3*** *:* Connexion et déconnexion
  - *Pour la connexion, la LoginFrame est ouverte au lancement de l'application, on appelle le UserService en se servant du SecurityService pour crypter le mot de passe saisi. Enfin, une fois cela effectué, le UserMapper est appelé et retourne l'utilisateur connecté. Les cas d'erreur de saisie sont gérés.*
  - *Pour la déconnexion, lors du clic sur le bouton, on ferme la fenêtre principale, l'utilisateur connecté dans le UserService est remis à null et on rollback l'ensemble des objets modifiés dans l'Unit Of Work.*

- **CU 4** : Affichage de la liste des discussions disponibles
  - *Le DiscussionsPanel affiche la liste des discussions disponibles par l'intermédiaire du DiscussionsService qui appellera le DiscussionsMapper et nous retournera l'ensemble des discussions disponibles pour l'utilisateur connecté.*

- **CU 5** : Envoi et réception de messages
  - *L'envoi et la réception de messages se fait par l'intermédiaire du DiscussionsPanel, il possède un panneau de droite avec une zone d'affichage des messages disponibles dans la discussion sélectionné dans la liste de gauche (à droite les messages de l'utilisateur connecté, à gauche les autres).*
  - *On peut envoyer 4 types de messages, cela est géré par des flags en base de données. Les prioritaires sont affichés en rouge, les accusés de réception sont affichés avec la date de vue en dessous du message concerné. Les chiffrés sont chiffrés en base de données et la date d'expiration est gérée lors de la récupération en base de données par le MessageMapper.*

- **CU 6** : Affichage des notifications
  - *L'affichage d'une notification se fait via le bouton situé en haut à droite dans l'application. Il aura une pastille rouge si une demande d'amitié en attente de confirmation est disponible. Lors du clic on ouvre une WaitingFriendshipFrame qui permet d'accepter ou refuser une demande via le FriendshipService qui va appeler le FriendshipMapper pour la persistance.*

- **CU 7** : Création d'un groupe
  - *La création d'un groupe correspond également à la création d'une discussion entre deux personnes. Il suffit de cliquer sur créer une discussion dans le DiscussionsPanel. Celui-ci ouvrira une fenêtre qui va appeler les DiscussionsService et DiscussionsMapper pour créer le groupe.*

- **CU 8** : Modération du groupe
  - *La modification d'un groupe se fait via le bouton Gérer dans le panneau de droite du DiscussionsPanel. On peut ajouter et enlever des personnes à la discussion. Un administrateur peut ajouter n'importe quel utilisateur de l'appli alors qu'un utilisateur verra uniquement ses amis. Lorsque l'on double-clic sur une personne de la liste droite, on peut la rendre modératrice de ce groupe.*

- **CU 9** : Recherche
  - *La recherche se fait via le SearchPanel, il suffit d'entrer un prénom et/ou un nom. L'appui sur le bouton Rechercher utilisera le SearchService qui va appeler le UserMapper afin de retourner une liste d'utilisateurs en fonction des valeurs des champs. Il retourne les prénoms et noms commençant par les chaînes de caractères saisies en ne prenant pas en compte la casse.*

- **CU 10** : Suppression d'un ami, départ d'un groupe
  - *La suppression d'un ami se fait via le SearchPanel. En cliquant sur le bouton Voir mes amis on obtient la liste de nos amis et on peut supprimer une amitié. Cela utilisera le FriendshipService qui va appeler le FriendshipMapper afin de supprimer le lien d'amitié entre l'utilisateur connecté et l'utilisateur sélectionné.*
  - *Le départ d'un groupe se fait via le panneau de droite du DiscussionsPanel. En cliquant sur gérer on peut basculer un utilisateur dans la liste de gauche et par conséquent quitter le groupe.*

Base de données

Pour le développement du projet, nous avons utilisé une base MySQL.

Ne pas oublier de modifier le fichier **config.yml** avec les bonnes informations si vous souhaitez vous connecter à votre base de données.

Vous retrouverez le schéma de la base de données (tables, colonnes, clés, clés externes, …) dans le fichier **schema.sql**.

Le jeu de données permettant de créer des utilisateurs, des amitiés et un groupe avec des messages se trouve dans le fichier **init.sql**.

Enfin, l'ensemble des requêtes SQL utilisées sont disponibles dans le fichier **requests.yml**

Pour information, ces fichiers sont disponibles dans le dossier **resources**.

Industrialisation

Afin d'obtenir une expérience de développement optimale, nous avons mis en place quelques tâches Maven :

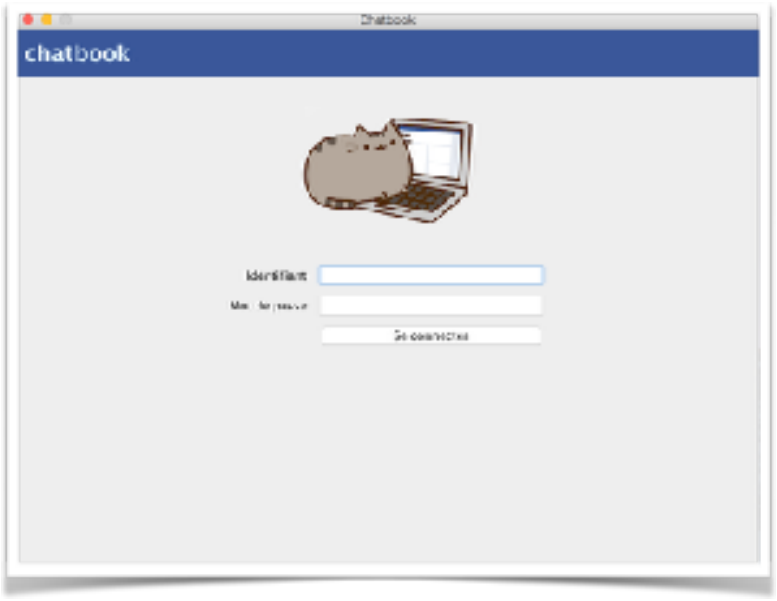
- **mvn clean install** *: Installation des dépendances du projet*
- **mvn initialize** *: Purge et initialisation de la base de données*
- **mvn exec:java** *: Execution du projet*
- **mvn clean compile assembly:single** *: Génération d'un fichier JAR exécutable*

UML - Diagramme de classes

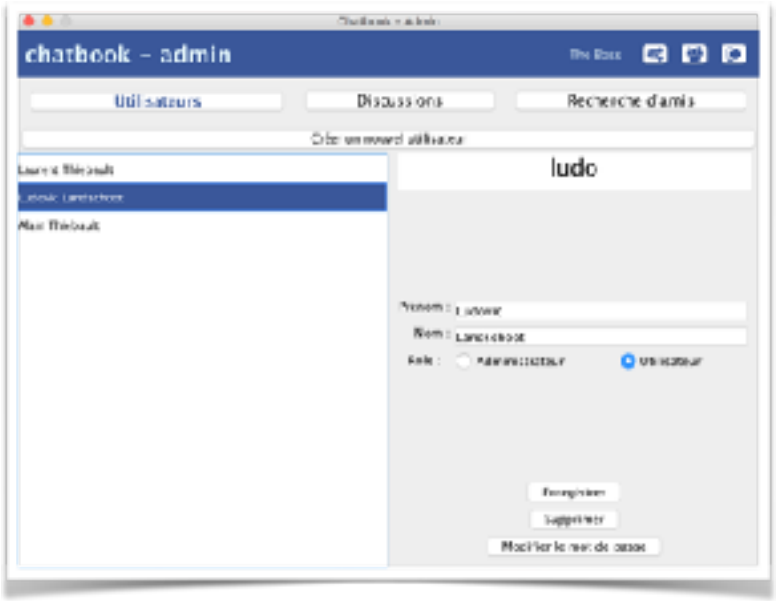
Couche domaine

TODO

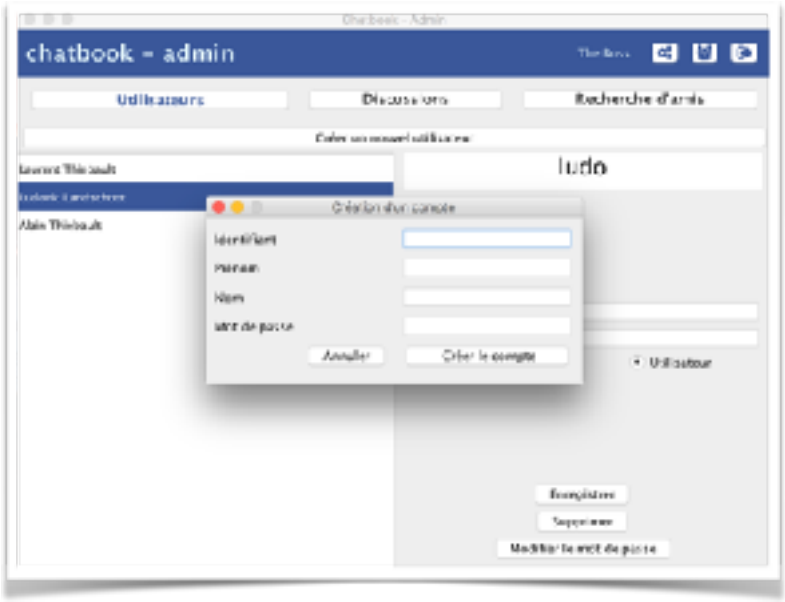
Interface utilisateur



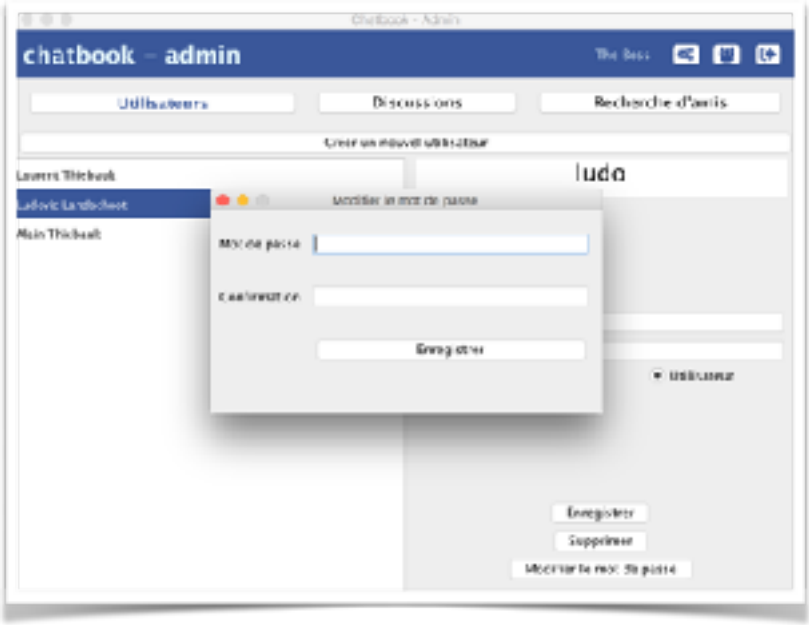
Connexion à l'application



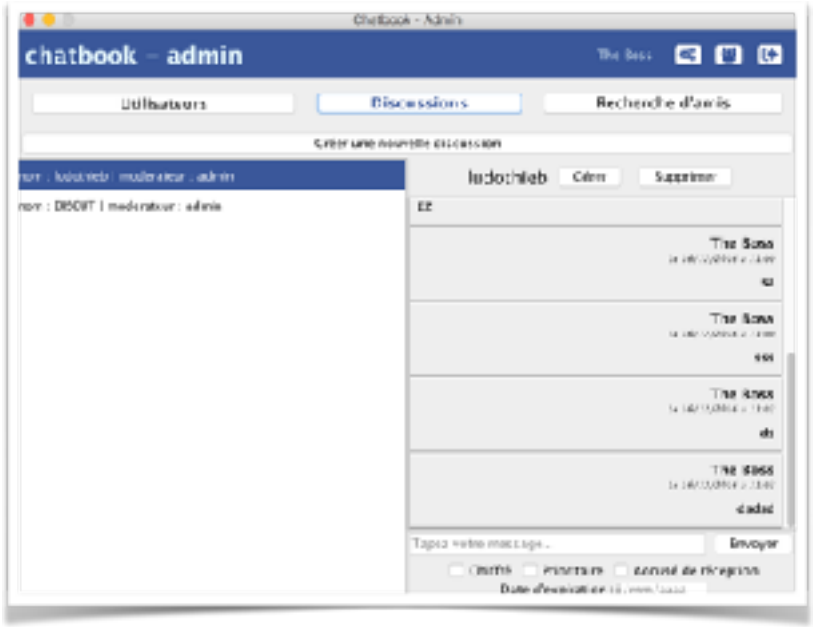
Onglet utilisateurs (admin) : Informations, modification d'un compte



Onglet utilisateurs (admin) : Création d'un compte

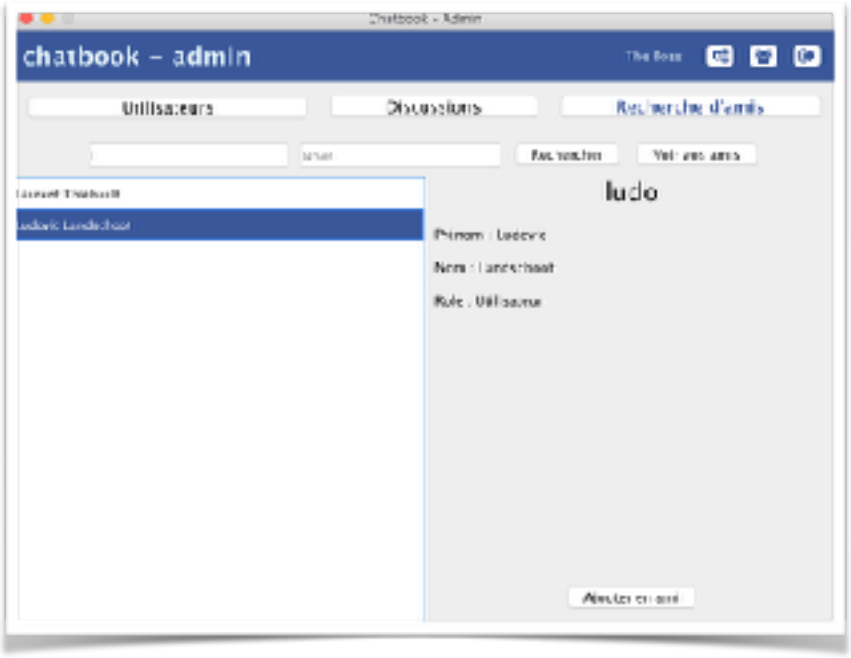


Onglet utilisateurs : Modification du mot de passe

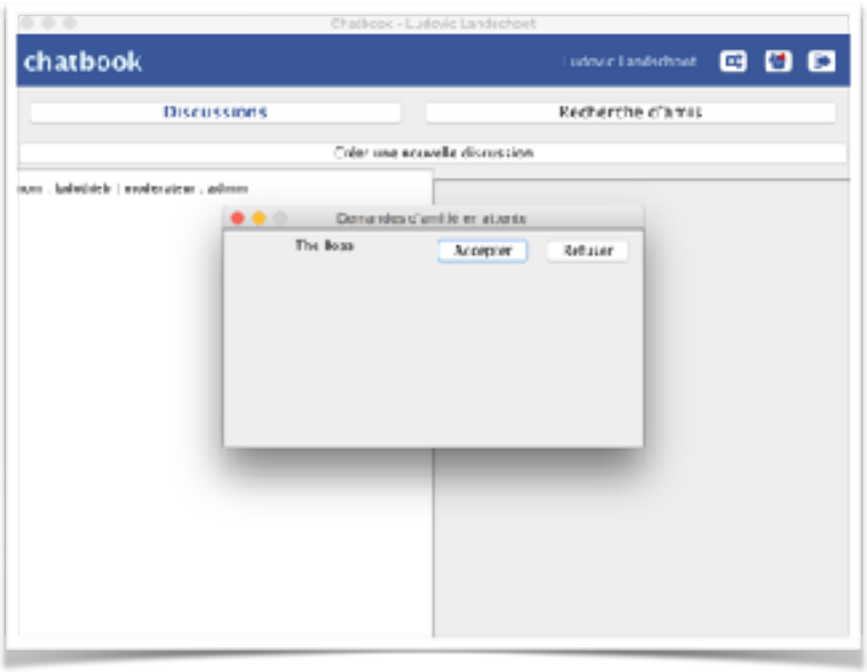


Onglet discussions : Consultation des discussions / messages

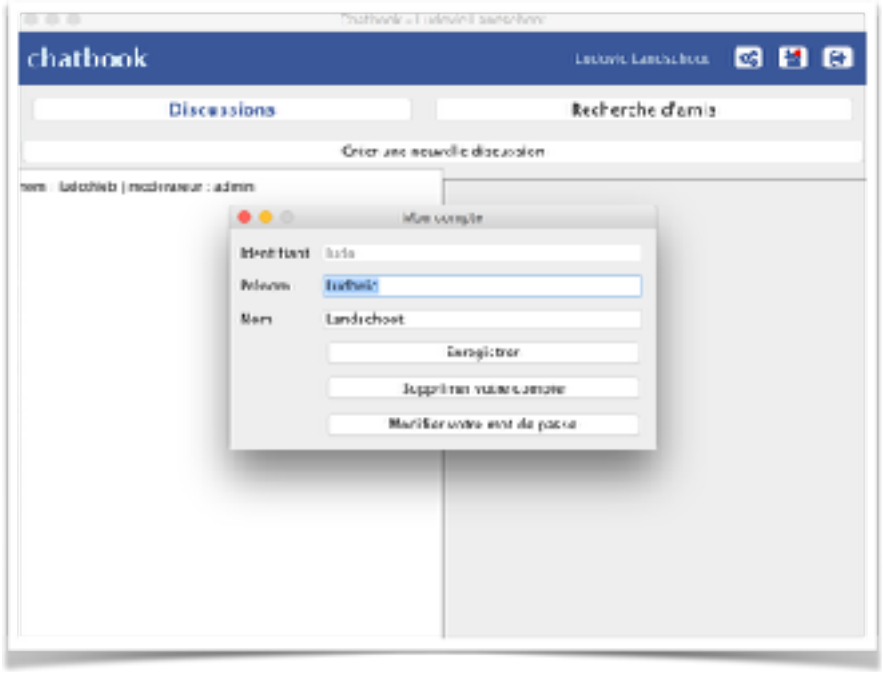
Onglet discussions : Gestion d'une discussion



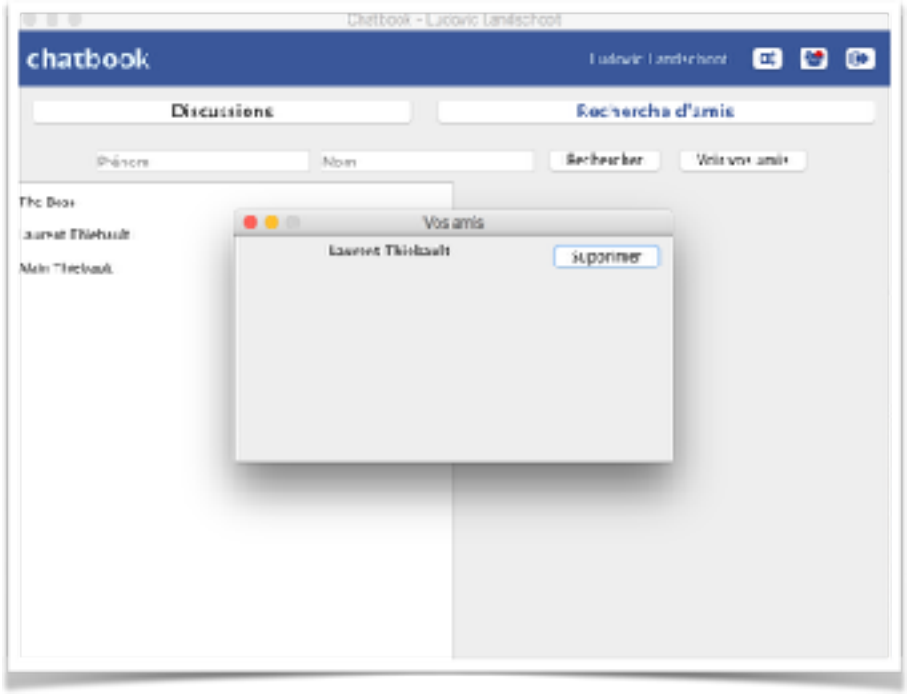
Onglet recherche d'amis : Recherche d'utilisateur et ajout en ami



Notification et demandes d'amitié en attente



Modification d'un compte



Onglet recherche d'amis : Voir vos amis