Module Inscription

# Introduction

Afin qu'un utilisateur puisse créer un compte directement depuis l'application, il est essentiel d'avoir un module inscription intégré à l'interface. Dans cette section du rapport, nous décrirons brièvement les choix d'implémentations pour ce module.

# Acteur du module

## ControllerUser

Ce contrôleur a pour but de faire l'intermédiaire entre viewInscription et le reste du programme : modelChator et serveur. Pour interagir avec le serveur, il se doit de respecter le protocole de sécurité décrit plus tard dans ce document.

## ViewInscription

La vue permet de faire une première vérification des champs lors de l'inscription. Cette vérification est détaillée plus tard dans cette section. La vue est également chargée d'informé le ControllerUser lorsque l'utilisateur veut soumettre son formulaire d'inscription.

La vue contient également le GUI décrit ci-dessous.

# Détails sur l'interface graphique

L'interface est très simpliste et basique. Il permet à l'utilisateur de comprendre très rapidement la logique derrière l'application.

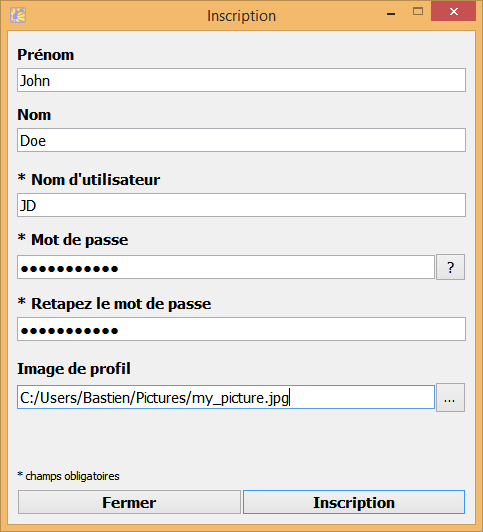


Figure Exemple d'inscription

# Validation des champs

Lors de la saisie d'une inscription, l'application vérifie les champs soumis par l'utilisateur pour des raisons de sécurité et de cohérence avec le serveur.

## Champs requis

Nous demandons à l'utilisateur le strict minimum d'information lors de l'inscription : un nom d'utilisateur et un mot de passe.

## Mot de passe

Dans l’optique d’une application sécurisée, nous imposons certaines contraintes concernant le mot de passe lors de l’inscription. Tout d'abord le mot de passe ne doit pas être identique au nom d'utilisateur. Ensuite, il doit contenir au moins 8 caractères et au moins :

* Une minuscule
* Une majuscule
* Un chiffre
* Un caractère spécial

Pour ce faire, nous utilisons une expression régulière à l’aide de l’objet QRegExp. Voici le pattern :

^(?=.\*[a-z])(?=.\*[A-Z])(?=.\*[0-9])(?=.\*[^a-z^A-Z^0-9]).{8,}$

"^" et "$" indiquent le début et la fin du String respectivement.

Ensuite nous avons 4 blocs, ils requirent l'utilisation d'au moins une minuscule, d'une majuscule, d'un chiffre et d'un caractère spécial respectivement. Le caractère spécial est simplement un caractère qui ne correspond à aucun des trois précédents blocs.

Pour finir, on indique avec {8,} le nombre minimum de caractères.

Cette même validation de mot de passe est bien sûr utilisée lors du changement de mot de passe d'un utilisateur.

## Nom d'utilisateur

L'unique contrainte concernant le nom d'utilisateur est le type de caractère utilisé : il peut uniquement contenir des lettres, des chiffres ainsi que les caractères – et \_ ainsi on évite d'avoir des noms d'utilisateur incompréhensibles.

Le nom d'utilisateur "Anonyme" (non sensible à la casse) est réservé pour des raisons d'implémentation. En effet ce nom est utilisé dans la fenêtre de chat, lorsqu'un utilisateur est inconnu. Ceci peut se produire qu'on utilisateur quitte une salle après avoir écrit des messages. Il devient "anonyme" pour les utilisateurs n'ayant aucune salle en commun avec lui

## Image de profil

L'utilisateur a la possibilité de définir son image de profile. Cette image doit respecter un format standard d'image (jpg, jpeg, png, gif), pour ce faire on utilise une variable de type QImage qui se charge elle-même de faire les vérifications nécessaires. Cette image sera immédiatement redimensionnée pour des raisons de stockage sur le serveur ainsi que pour optimiser l'affichage dans l'application.

## Avertissement à l'utilisateur

Si une de ces précédentes conditions n'est pas respectée concernant les champs, un de ces messages sera affiché à l'utilisateur dans un label dédié en bas de fenêtre :













# Cheminement pour la création de compte

Afin de permettre l'utilisation du chiffrement et un certain niveau de sécurité au sein de l'application. Il est nécessaire d'effectuer des tâches bien précises lors de la création d'un compte. Ces tâches sont principalement effectuées dans controllerUser mais elles sont également partagées avec les différents acteurs de l'application pour permettre une communication avec le serveur.

Le module cryptor permet d'effectuer toute action relative à la sécurité. Il sera donc vivement sollicité lors de l'inscription.

## Récupération des informations depuis le formulaire

Une fois que l'utilisateur a saisi les informations relatives à son nouveau compte (nom, prénom, nom d'utilisateur, mot de passe, image de profile). Le contrôleur se charger de récupérer les valeurs saisies après avoir été sollicité par la vue grâce au processus de signal.

## Génération du sel pour le mot de passe et hachage de ce dernier

Etant donné qu'un mot de passe ne doive pas transiter sur le réseau en clair pour des raisons évidentes, il est nécessaire d'hacher ce dernier. De plus, le serveur lui-même ne doit pas connaitre ce dernier. Il se contentera de le stocker sous forme de hash.

Grâce à la fonction "generateSalt" du Cryptor, on génère un sel. Ensuite ce sel est utilisé en complément du mot de passe afin d'obtenir un hash.

## Génération du sel pour la paire de clé et chiffrement de la clé privée

Ensuite, on va générer une paire de clé, et ceci toujours en local, puis du sel. Ce sel contribuera au chiffrement AES de la clé privée. Par définition, la clé publique n'a aucun besoin d'être "cachée" aux autres.

De cette manière, on pourra stocker une paire de clé (dont la clé privée est chiffrée) directement sur le serveur. Lors d'une connexion, l'utilisateur pourra recevoir cette paire de clé et déchiffrer la partie privée.

Pour l'envoi d'un message chiffré à l'utilisateur dans le cadre d'une salle privée, on utilisera sa clé publique pour le chiffrement. Ainsi il pourra déchiffrer le message avec sa clé privée.

## Envoi au serveur

Une fois toutes les opérations précédemment décrites effectuées, on peut maintenant transmettre toutes les informations au serveur.

## Pour conclure

Toutes ses manipulations permettent de rendre invisible à l'utilisateur les différentes couches de sécurité. A partir d'un seul mot de passe et nom d'utilisateur, il est capable d'accéder à une application sécurisée. Ceci marque l'importance d'un mot de passe avec une certaine complexité étant donné que toute la sécurité se base sur ce dernier.

Dans le cas où le serveur serait compromis, cela ne mettrait en aucun cas les informations de l'utilisateur en péril. Car comme expliqué précédemment aucune information critique (clé privée et mot de passe) n'est stockée en clair.