1. Énoncé

Le but de ce projet est de réaliser une application permettant à deux utilisateurs de communiquer l’un avec l’autre par écrit et à distance. Cette application consistera en réalité en deux applications distinctes, une application client et une application serveur. Les clients seront capables de communiquer uniquement avec le serveur qui redirigera les messages aux clients correspondants.

Les informations des utilisateurs seront stockées en base de données relationnelle (mySQL). Quant aux messages des utilisateurs, ils seront conservés dans une base de données non relationnelle (Cassandra).

Afin que les utilisateurs puissent voir les messages hors connexion, ces derniers seront également enregistrés localement dans un fichier. Ce fichier sera synchronisé avec la base de données lors du lancement de l’application.

Les utilisateurs possèderont un profil contenant diverses informations sur eux-mêmes, notamment leur pseudo, leur adresse mail, leur date de naissance, leur description, leurs numéros de gsm et / ou de téléphone fixe ainsi qu’une image de profil. Ces informations seront publiques ou privées selon le choix de l’utilisateur.

Un utilisateur pourra communiquer avec un autre utilisateur lorsque ce dernier est enregistré dans sa liste de contact. Pour ce faire, il doit effectuer une recherche parmi les utilisateurs de l’application via son pseudo ou son adresse mail, le sélectionner dans les résultats et enfin lui envoyer une demande de contact.

Un utilisateur pourra voir le profil des autres utilisateurs.

1. Glossaire

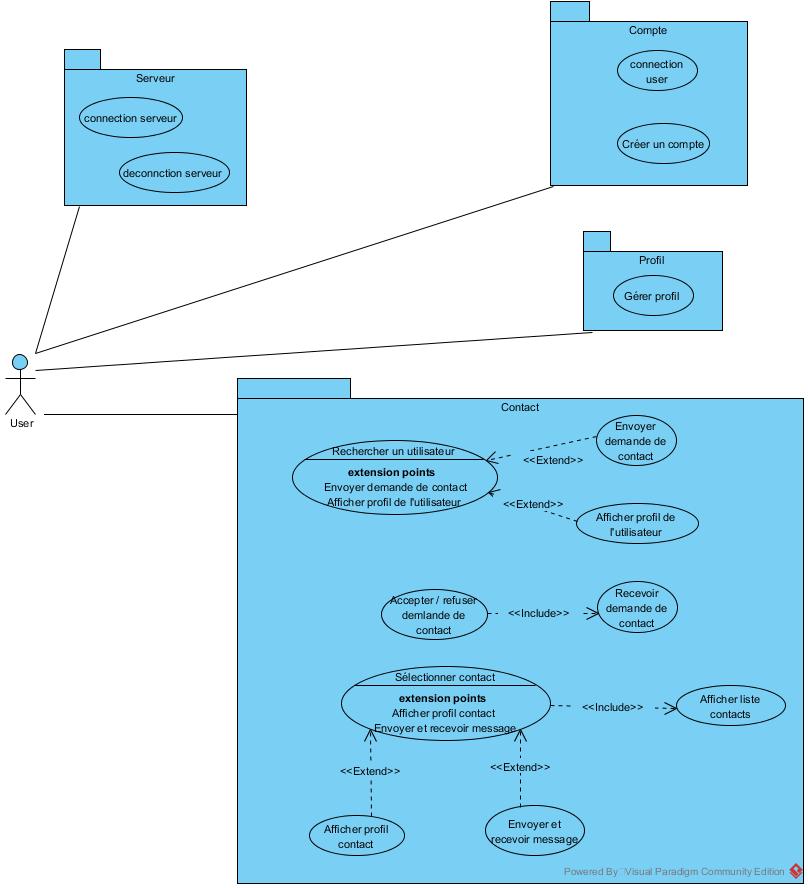
* État d’un utilisateur : indique si l’utilisateur est connecté, déconnecté, occupé, …
* Client : application utilisée par l’utilisateur

1. Détails et responsabilités des acteurs

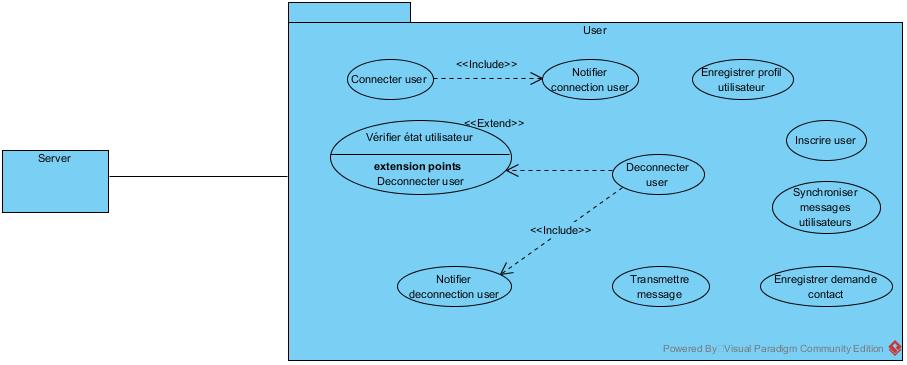
* L’utilisateur :
  + Se connecter
  + Créer un compte
  + Gérer son profil
  + Rechercher un utilisateur
  + Envoyer une demande de contact à un utilisateur
  + Recevoir et accepter / refuser une demande de contact d’un autre utilisateur
  + Sélectionner un contact pour ouvrir la fenêtre de discussion
  + Envoyer un message à un contact grâce à la fenêtre de discussion
  + Recevoir un message
* Serveur :
  + Indique aux contacts d’un utilisateur que ce dernier est connecté
  + Vérifie l’état d’un utilisateur
  + Transmet les messages d’un utilisateur à un autre
  + Connecter l’utilisateur
  + Synchroniser les messages de l’utilisateur
  + Enregistrer les demandes de contact des utilisateurs

1. Diagramme des cas d’utilisation

* Utilisateur



* Serveur



1. Description et énumération des UC

|  |  |  |  |
| --- | --- | --- | --- |
| Numéro de UC | Application | Nom du UC | Description |
| 1 | Client | Connection user | L’utilisateur rentre son identifiant / adresse mail ainsi que son mot de passe. |
| 2 | Client | Créer un compte | L’utilisateur choisira un identifiant unique, un mot de passe et renseignera ses informations personnelles (adresse mail, date de naissance, …). |
| 3 | Client | Gérer profil | L’utilisateur aura la possibilité de gérer son profil en choisissant une image de profil, renseignant une description de lui, son numéro de téléphone, son numéro de gsm, …  Il pourra aussi décider qui (lui uniquement, ses contacts, tout le monde) aura le droit de voir quelle information. |
| 4 | Client | Rechercher un utilisateur | L’utilisateur pourra rechercher un autre utilisateur en renseignant son pseudo. |
| 5 | Client | Afficher profil d’un utilisateur | Ce UC nécessite l’accomplissement de l’UC 4.  L’utilisateur peut consulter le profil d’un utilisateur. |
| 6 | Client | Envoyer demande de contact | Ce UC nécessite l’accomplissement de l’UC 4.  L’utilisateur peut envoyer une demande de contact à un autre utilisateur. |
| 7 | Client | Accepter / refuser une demande de contact | Après avoir reçu une demande de contact, l’utilisateur peut décider de refuser ou d’accepter celle-ci. |
| 8 | Client | Afficher liste de contacts | L’utilisateur peut afficher sa liste de contact et en sélectionner un. |
| 9 | Client | Afficher profil contact | Ce UC nécessite l’accomplissement de l’UC 8.  L’utilisateur peut consulter le profil d’un de ses contacts. |
| 10 | Client | Envoyer et recevoir message | Ce UC nécessite l’accomplissement de l’UC 8.  L’utilisateur peut envoyer et recevoir des messages de ses contacts. |
| 11 | Serveur | Connecter user | Lorsqu’un utilisateur voudra se connecter, le serveur vérifiera les informations de l’utilisateur et le connectera si ces dernières sont correctes.  Les contacts de l’utilisateur seront informés de sa connexion. |
|  |  |  |  |
| 12 | Serveur | Inscrire user | Lorsqu’un utilisateur voudra créer un compte, le serveur vérifiera si l’adresse mail ainsi que l’identifiant ne sont pas déjà utilisés et enregistrera les informations de l’utilisateur dans la base de données. |
| 13 | Serveur | Vérifier état user | Régulièrement, le serveur vérifiera l’état des utilisateurs connectés et adaptera le statut de ceux-ci si cela est nécessaire. |
| 14 | Serveur | Transmettre message | Lorsqu’un utilisateur enverra un message à un autre utilisateur, le serveur enregistrera le message, l’heure et la date auxquelles il a été envoyé, le destinataire, l’émetteur ainsi que son statut puis transmettra le message au destinataire. |
| 15 | Serveur | Enregistrer profil utilisateur | Lorsqu’un utilisateur modifie son profil, les modifications sont enregistrées par le serveur |
| 16 | Serveur | Synchroniser messages utilisateur | Si l’utilisateur a utilisé l’application sur un autre appareil, il sera nécessaire de synchroniser les messages stockés sur le serveur avec les messages stockés sur un fichier en local sur l’appareil de l’utilisateur |
| 17 | Serveur | Enregistrer demande contact | Lorsqu’un utilisateur envoie une demande de contact à un autre utilisateur, le serveur enregistre cette demande en base de données afin que la demande puisse être réceptionnée si le destinataire est déconnecté. |
| 18 | Client | Connection serveur | Lors du lancement de l’application client, ce dernier essaie de se connecter à l’application serveur. |
| 19 | Client | Deconnection serveur | Lorsque l’utilisateur ferme l’application, cette dernière se déconnecte d’abord du serveur. |
| 20 | Serveur | Deconnecter user | Lorsqu’un utilisateur se déconnecte, le serveur ferme tous les threads liés à cet utilisateur et notifie ses contacts de la déconnection. |

1. Répartition des UC par itération

|  |  |  |  |
| --- | --- | --- | --- |
| Itération | Application | No UC | UC |
| Itération 1 | Client | 1 | Se connecter |
| 2 | Créer un compte |
| Serveur | 11 | Connecter user |
| 12 | Inscrire user |
| 13 | Vérifier état user |
| Itération 2 | Client | 3 | Gérer profil |
| 4 | Rechercher un utilisateur |
| 5 | Afficher profil d’un utilisateur |
| Serveur | 15 | Enregistrer profil utilisateur |
| Itération 3 | Client | 6 | Envoyer demande de contact |
| 7 | Accepter / refuser une demande de contact |
| 8 | Afficher liste de contacts |
| 9 | Afficher profil contact |
| Serveur | 17 | Enregistrer demande contact |
| Itération 4 | Client | 10 | Envoyer et recevoir message |
| Serveur | 14 | Transmettre message |
| 16 | Synchroniser messages utilisateur |

1. Modèle de données

USER

adresse\_mail : varchar(300)

password : varchar(300)

pseudo : varchar (50)

contacts[0-N] : USER

date\_naissance : date

status : char(1)

description : varchar(3000)

no\_gsm : char(20)

no\_fixe : char(20)

DEMANDE\_CONTACT

id : int

date : date

message : varchar(3000)

CONFIDENTIALITE

conf\_adresseMail : char(1)

conf\_dateNaissance : char(1)

conf\_noGsm : char(1)

conf\_noFixe : char(1)

Contraintes d’intégrité

status : c = connecté

o = occupé

n = ne pas déranger

i = invisible

d = déconnecté

conf\_adresseMail : v = visible

c = contacts uniquement

p = privé

conf\_dateNaissance : voir conf\_adresseMail

conf\_noGsm : voir conf\_adresseMail

conf\_noFixe : voir conf\_adresseMail