Rayane Rachid Kennaf et Lemar Andar

Projet synthèse

Groupe 01

Projet synthèse :

LOCs

Travail présenté à

Jean-Christophe Demers

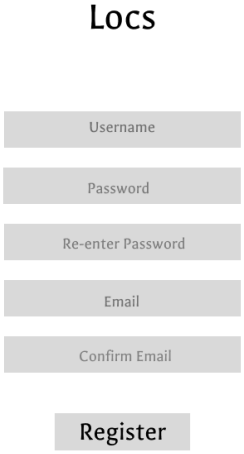
Cégep du Vieux-Montréal

02 février 2023

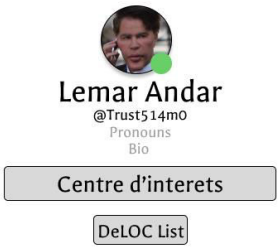
Maquette du UI

Login

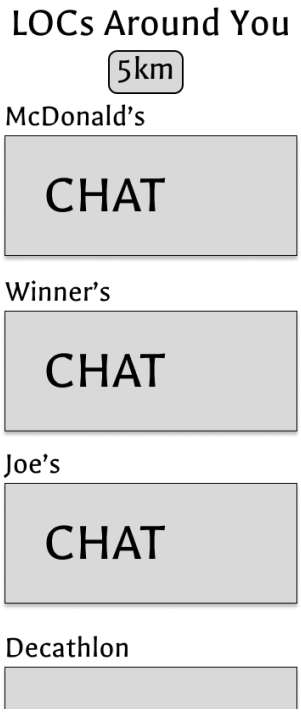
Register



Voir profil

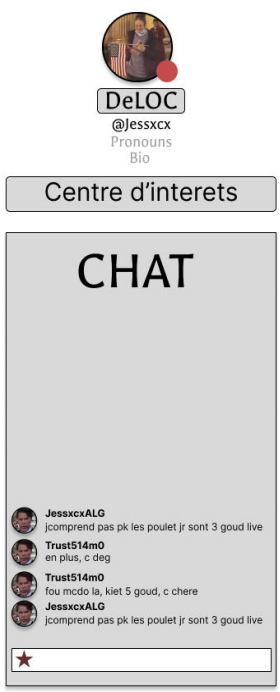


Chat autour



Deloc

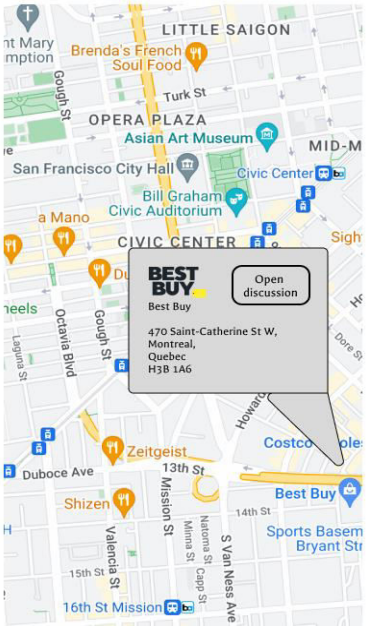
Chat privée



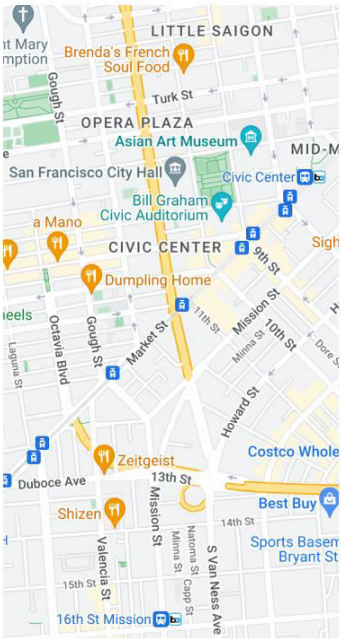
Map des lieux

Chat public

Location



Location



McDonald's

Diagramme des classes

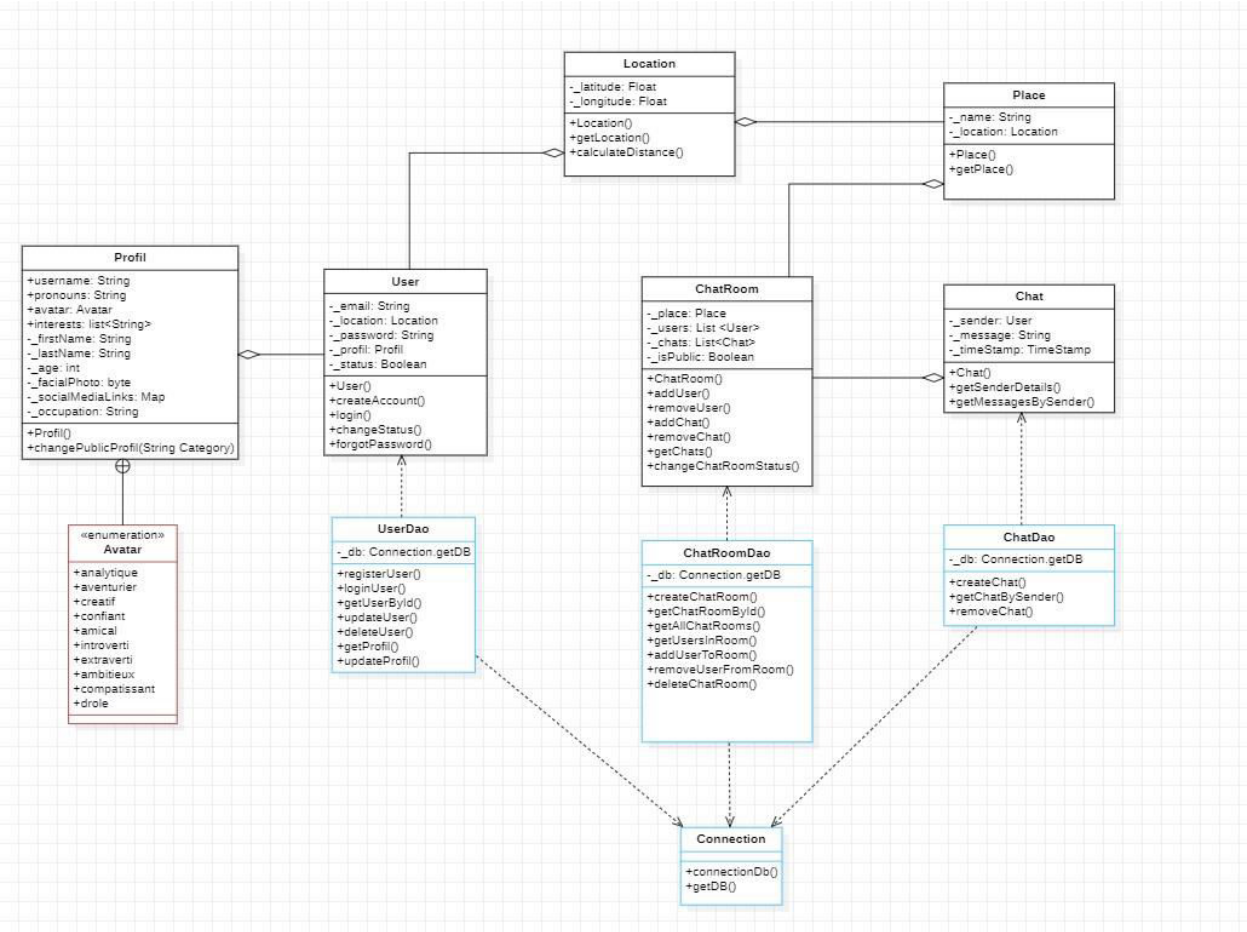
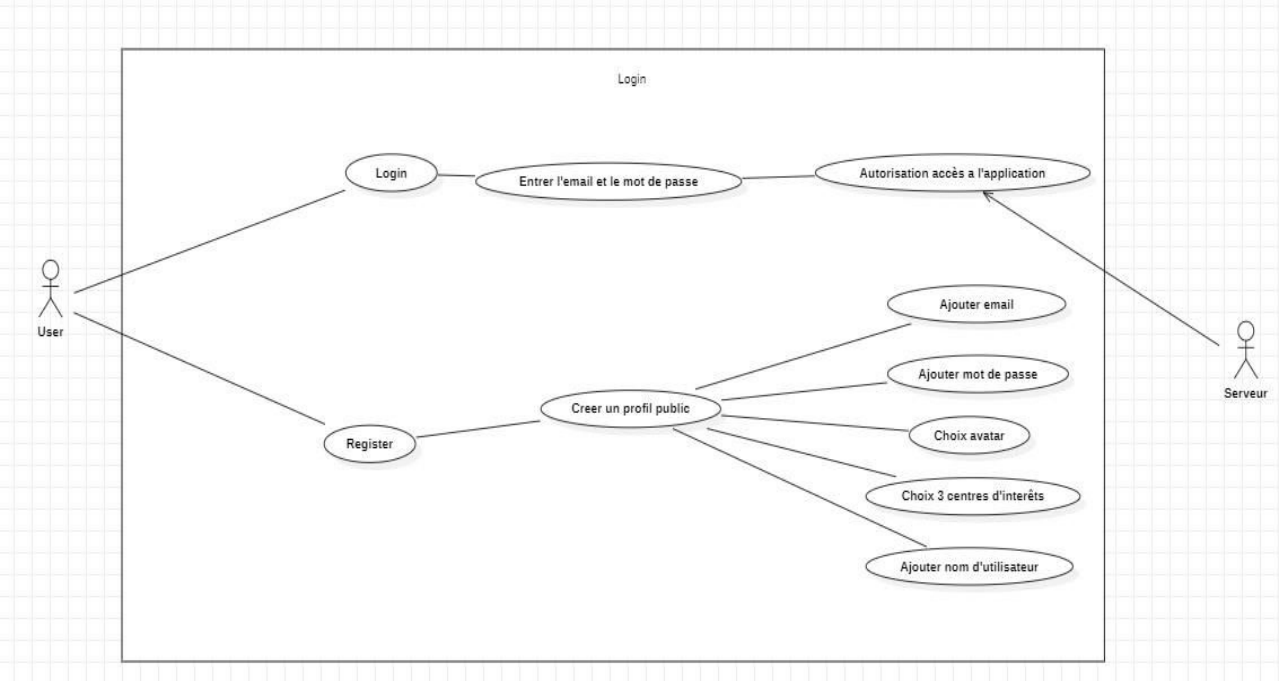
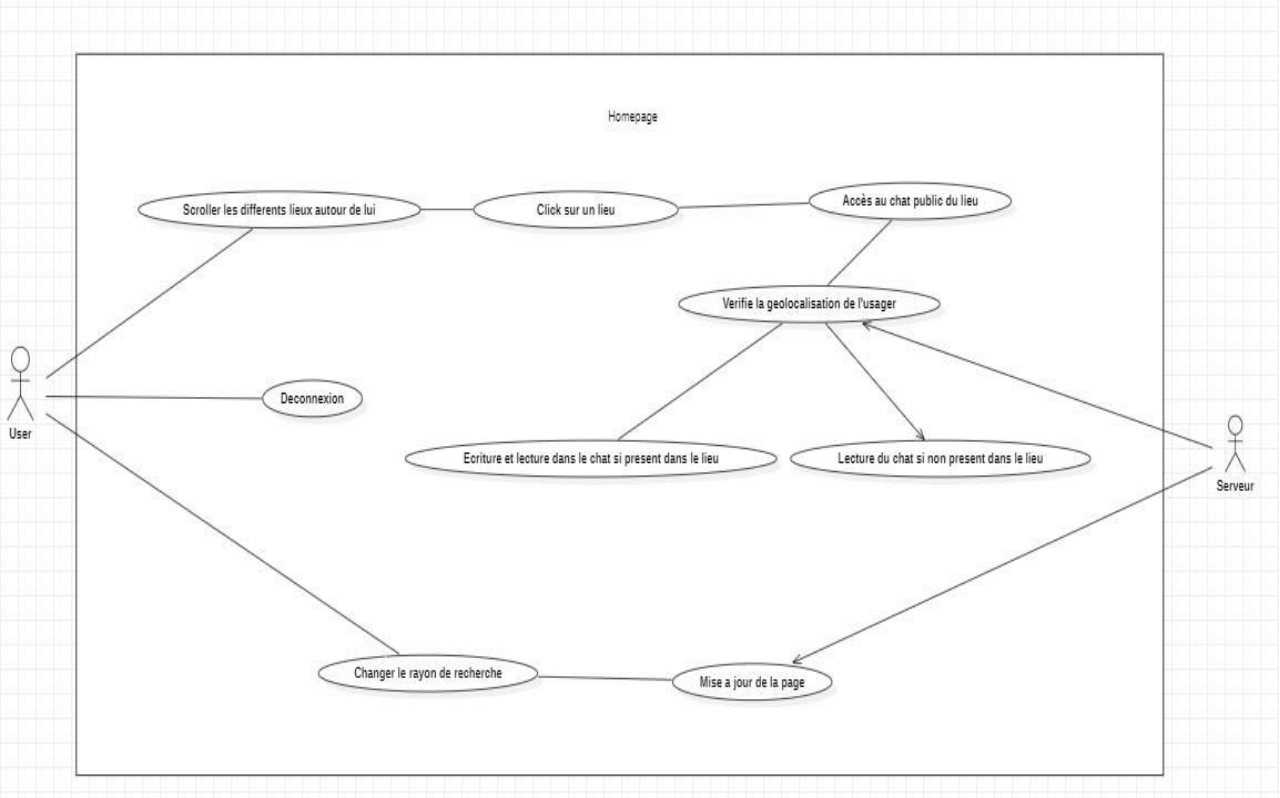


Diagramme des cas d'usages

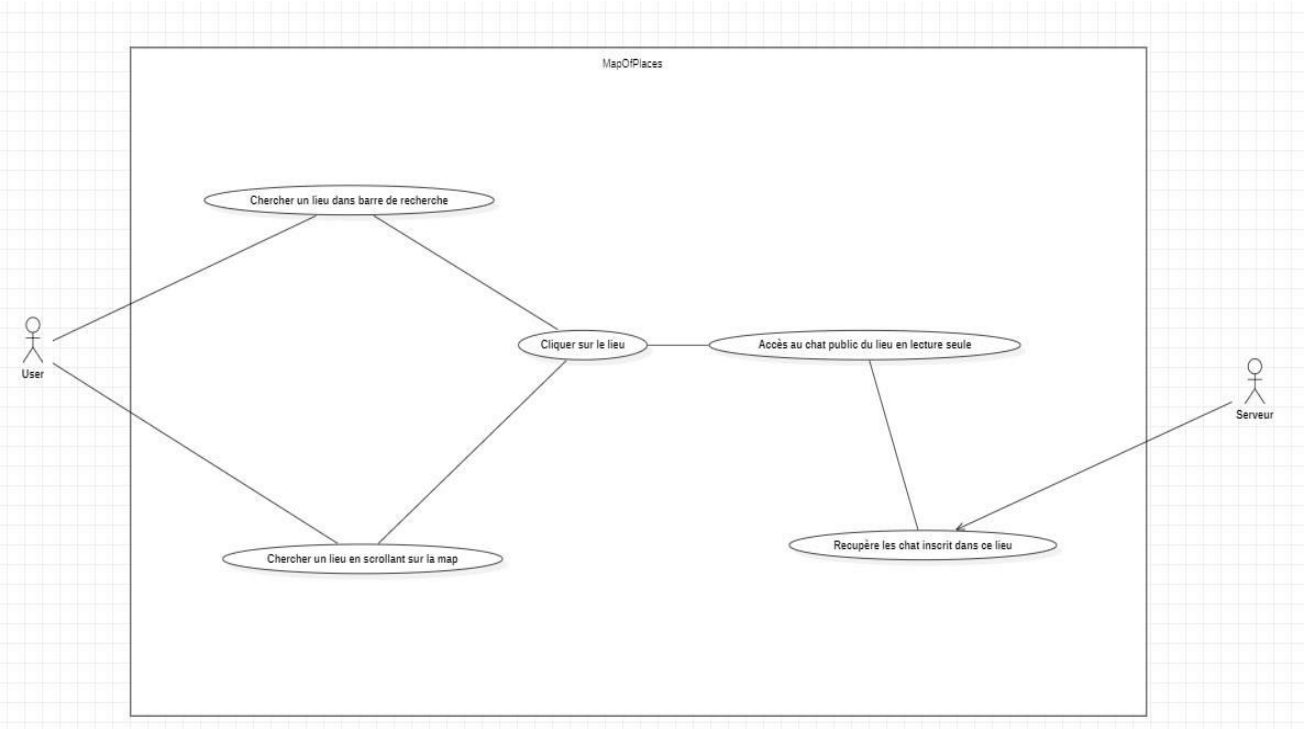
Login/Register



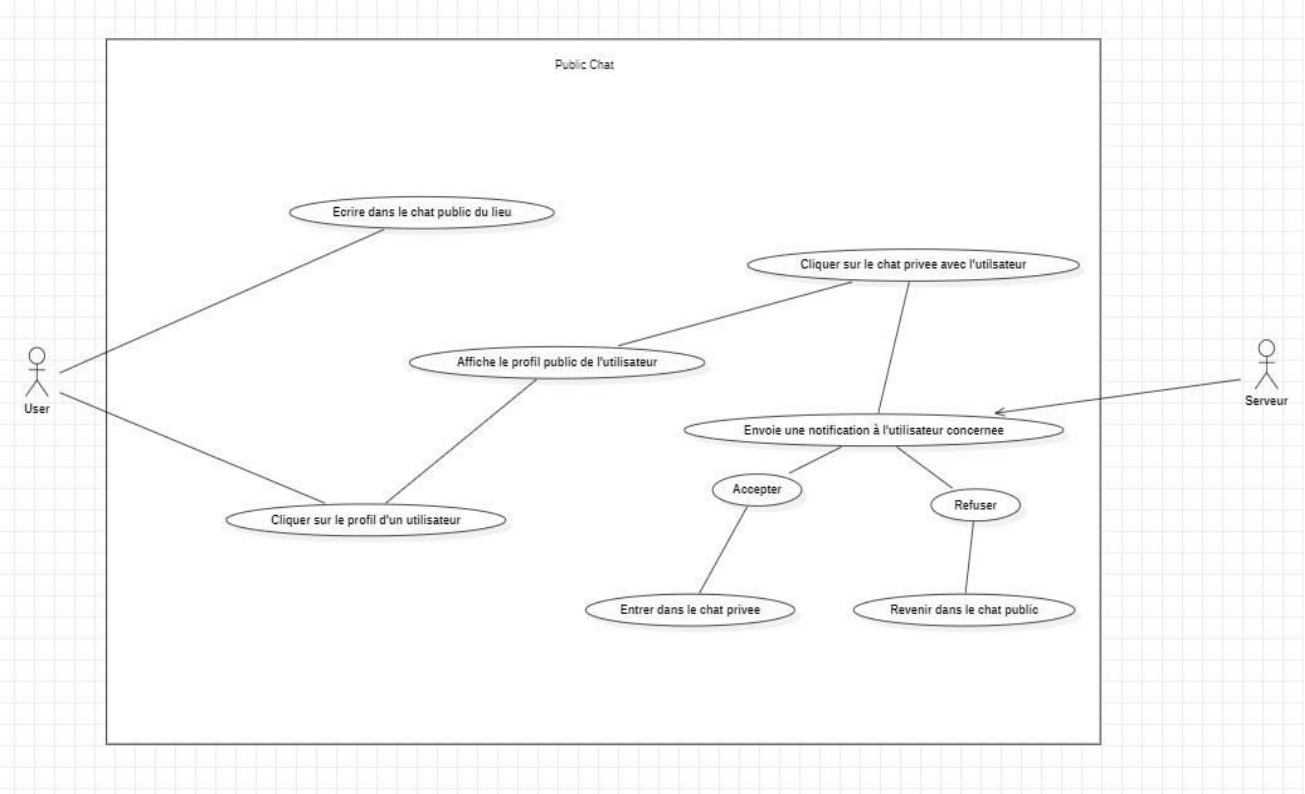
Homepage



Map des lieux



Chat Public



Chat Privée

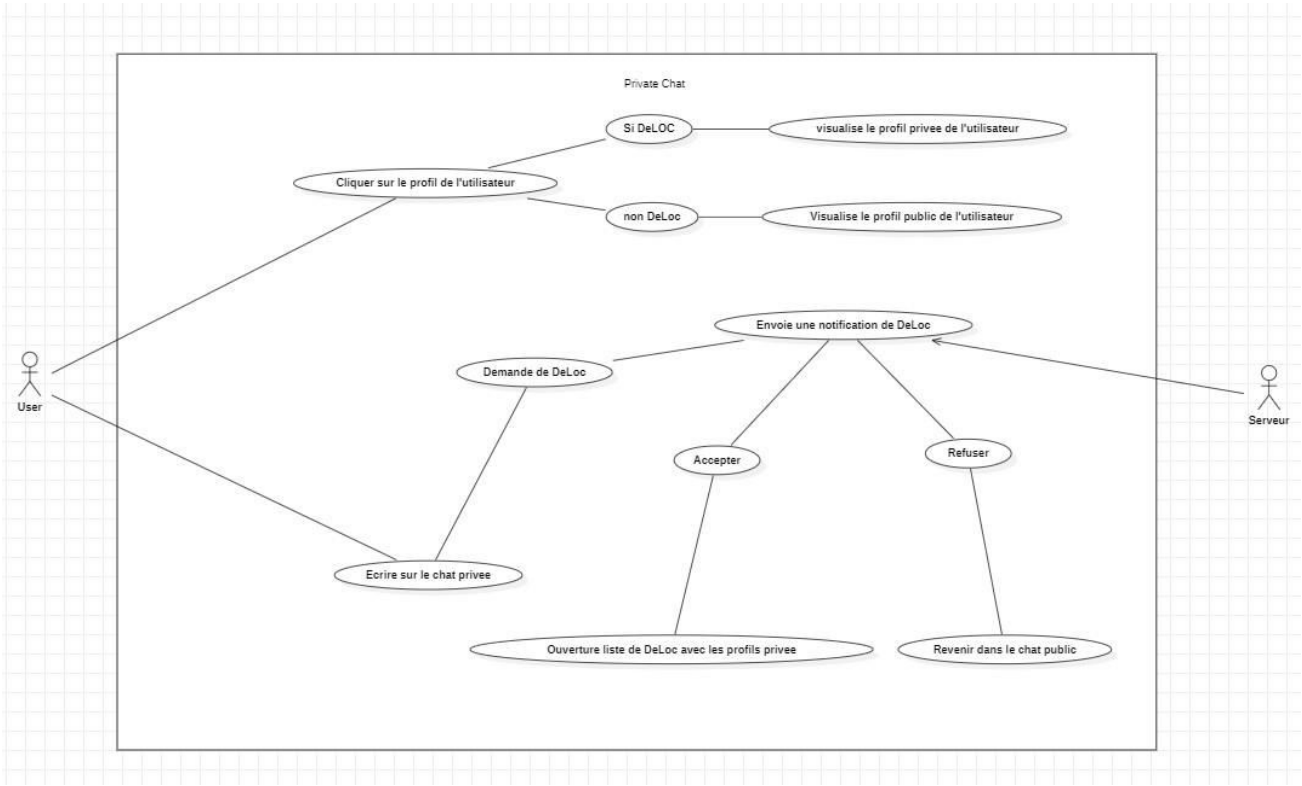
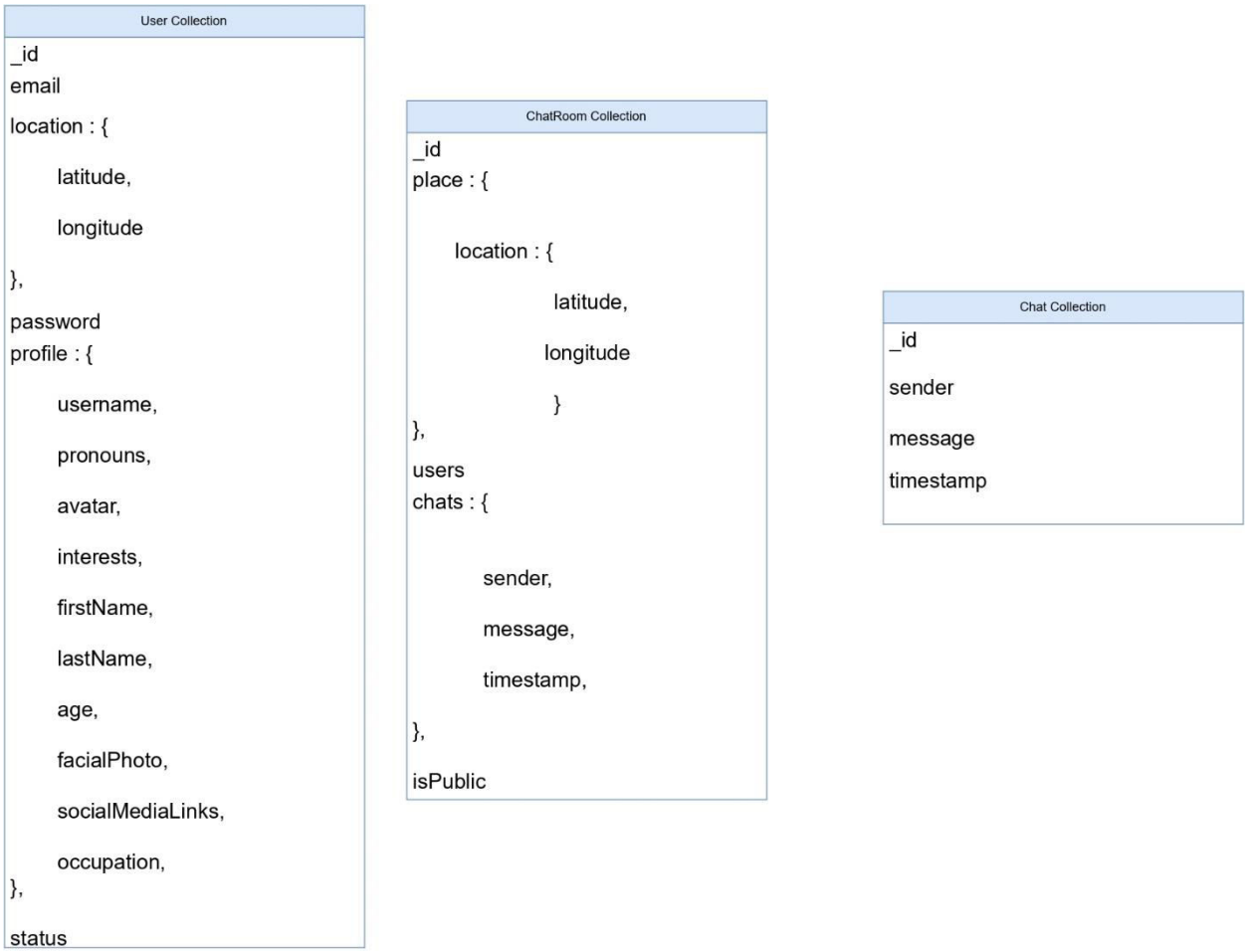

Private Chat
Si DeLOC
visualise le profil privee de l'utilisateur
Cliquer sur le profil de l'utilisateur
non DeLoc
Visualise le profil public de l'utilisateur
Envoie une notification de DeLoc
Demande de DeLoc
User
Serveur
Accepter
Refuser
Ecrire sur le chat privee
Ouverture liste de DeLoc avec les profils privee
Revenir dans le chat public

Schéma de la structure de données externe



Notes :

Pour la base de données, on utilise MongoDB, qui est une base de données type document, donc pas de clé étrangère. On utilise l'embending (incorporation). Par exemple : dans la Collection ChatRoom, il y a une liste de chats qui elles contient les messages, le temp ou ils ont été envoyés et par qui.

Éléments de conception

Pour ce qui est des structures de données, nous avons choisi d'utiliser un tableau associatif, des types primitifs, des types composés et des types abstraits. Un tableau associatif, également appelé map, est un type de structure de données qui permet aux développeurs de stocker des données sous forme de paires clé-valeur. Cela rend facile la récupération de données en fonction de leur clé, ce qui peut être une fonctionnalité utile dans notre application où les utilisateurs peuvent vouloir récupérer des informations sur d'autres utilisateurs.

Les types primitifs sont des types de données simples intégrés aux langages de programmation, tels que les ints, les chars et les float. Ces types de données peuvent être utiles pour représenter des informations de base dans notre application, telles que l'âge, le nom ou l'emplacement de l'utilisateur.

Les types composés, tels que les array et les strings, sont des structures de données qui peuvent être utilisées pour stocker plusieurs valeurs dans une seule variable. Dans notre application, les array pourraient être utilisés pour stocker les intérêts des utilisateurs tandis que les strings pourraient être utilisées pour représenter du texte à l'écran ou de récupérer les inputs des usagers. (Chat, courriel, mot de passe).

Enfin, les types abstraits, tels que les listes, permettent la gestion de collections de données. Les listes peuvent être utilisées pour stocker des structures de données complexes de manière plus organisée et efficace que d'autres structures de données. Dans notre application de médias sociaux, nous avons utilisé des listes pour gérer des listes de chats ainsi que des listes d'usagée dans un chat room

En ce qui a trait aux patrons de conceptions, nous avons utilisé le patron de conception façade pour simplifier les parties complexes de notre application, ce qui la rend plus facile à comprendre et à naviguer pour les utilisateurs. Dans React Native, l'API StyleSheet est un exemple de patron de façade. Avec cette API, qui ressemble à du CSS, permet de définir des styles pour notre application, qui ainsi peuvent être appliquer à l'interface graphique de notre application.

Nous avons également utilisé le patron de conception listener, qui est largement utilisé dans React Native pour gérer les interactions des utilisateurs. Les listeners permettent aux développeurs de répondre aux entrées des utilisateurs de manière flexible et dynamique, rendant les applications plus interactives et réactives. Dans notre application de médias sociaux, nous avons utilisé des listeners pour répondre aux actions des utilisateurs telles que les pressions de bouton et les soumissions de formulaire tel que l'enregistrement de l'usager, de son profil public et privée.

Le patron de conception state est un autre patron de conception couramment utilisé dans React Native. Le patron de conception state permet aux à un objet de modifier son comportement lorsque son état interne change. Dans notre application de médias sociaux,

nous avons utilisé le patron de conception state pour gérer l'état de la session utilisateur, tel que la connexion ou la déconnexion.

Le dernier patron de conception que nous avons choisi est le patron de conception Builder. Ce dernier vous permettez de construire des objets d'une classe étape par étape. Dans le cas de notre application, l'inscription pourrait bénéficier d'un Builder. Notre user a une instance d'un profil public ainsi qu'une autre instance d'un profil privée. Le profil public est instancié directement dans l'inscription, mais pas le profil privé, qui lui va être instancier plus tard. C'est pour ça qu'un Builder peut nous aider dans ce cas de figure.

Pour les expressions régulières, les deux qui serait les plus évidentes et que nous avons choisi sont les expressions régulières pour le courriel et le mot de passe. Pour le courriel, nous le validons sous la forme suivante : Le nom d'utilisateur peut contenir des lettres majuscules et minuscules, des chiffres et les caractères spéciaux «. », « _ », « % », « + » et « - ».

- Il doit y avoir une @ entre le nom d'utilisateur et le domaine.
- Le nom de domaine peut contenir des lettres majuscules et minuscules, des chiffres et les caractères spéciaux «. » et « - ».

En ce qui pour le mot de passe, nous le validons sous la forme suivante :

- Le mot de passe doit contenir au moins une lettre majuscule, un chiffre et un symbole spéciale
- La longueur minimale du mot de passe est de 8 caractères.

En plus de cette validation des mots de passe, nous avons également utilisé l'algorithme de hachage SHA-3 pour stocker de manière sécurisée les mots de passe dans notre base de données. SHA-3 est une fonction de hachage cryptographique qui prend en entrée une chaîne de caractères et renvoie une valeur de hachage, qui est une représentation numérique unique de la chaîne de caractères. Cette fonction est considérée comme très sécurisée et est largement utilisée pour stocker les mots de passe de manière sécurisée. Nous pensons que l'algorithme le plus approprier par rapport à notre application dans la mesure où la sécurité est un enjeu très important dans des applications tel que les nôtres. Si on n'utilisait pas d'algorithme de hachage et qu'on sauvegarderait les mots de passes des utilisateurs directement dans la base de données alors cela serait très risquée. Si la base de données est compromise, alors ceux qui l'attaquerait auraient un accès facile aux mots de passes des utilisateurs.

Aussi, pour la formule mathématique, nous avons décidé d'utiliser la formule de Haversine. La formule de Haversine est une formule mathématique utilisée pour calculer la distance entre deux points sur une sphère, telle que la surface terrestre. Elle est couramment utilisée pour calculer des distances entre deux points sur une carte. La formule de Haversine est utile pour notre application car elle permet de calculer avec précision la distance entre deux utilisateurs qui se trouvent à des endroits différents, même si ces endroits sont éloignés. En utilisant cette formule, nous pouvons nous assurer que les utilisateurs qui sont les plus proches les uns des autres sont jumelés. Dans notre logique pour l'application, nous aurions ainsi pour chaque endroit une localisation avec un point qui utilise comme unité de mesure

latitude/longitude. Avec cette unité de mesure, on peut récupérer la position de l'utilisateur et voir, à l'aide de la formule de Haversine, si l'utilisateur est proche d'un certain lieu.

Finalement, en ce qui est de comment nous comptons lier le cours de Veille technologique au cours de Projet synthèse, nous avons choisi en grande partie l'exploration de la technologie de React Native. Lorsqu'on a eu l'idée de cette application, nous voulions que cette dernière soit principalement sur iOS, mais cela n'aurait pas pu être possible. Tout d'abord, la grande majorité des utilisateurs sont sur Android. De plus, vu que c'est un projet ou il fallait utiliser un langage qu'on avait déjà appris, alors cela aurait été impossible d'avoir choisi Swift (Objective-C) pour notre application. La solution parfaite était de prendre React Native, car elle se base sur du javascript, un langage qu'on avait déjà exploré lors de notre cursus. Ce Framework règle la très grande majorité de nos problèmes. Elle est multiplateforme, n'a qu'une seule codebase et elle utilise javascript/typescript. En ce qu'elle apportera, elle permettra d'utiliser des apis qui nous serons fondamentale à notre projet telle que l'api pour les cartes, l'api permettant de glisser entre les pages et pleins d'autres apis qui elles sont complètement open-sources.

En conclusion, pour le développement de notre application de médias sociaux, nous avons choisi d'utiliser diverses structures de données telles que des tableaux associatifs, des types primitifs, des types composés et des types abstraits pour stocker et organiser les données. Nous avons également utilisé plusieurs patrons de conception tels que la façade, le listener, le state et le Builder pour faciliter la gestion de notre application, la rendre plus interactive et plus réactive. Nous avons également utilisé des expressions régulières pour valider les entrées de l'utilisateur et stocker de manière sécurisée les mots de passe en utilisant l'algorithme de hachage SHA-3 dans notre base de données. Finalement, nous avons choisi la formule de Haversine pour calculer la distance entre deux points.