Plateforme de discussions et d'apprentissage de langues « SocialTaal »

BINV3140 – Spring : Projet (2ème session)

Contexte

Vous désirez créer une startup proposant une plate-forme permettant aux utilisateurs de discuter entre eux dans différentes langues pour apprendre en se faisant des amis.

Son fonctionnement sera le suivant :

- Les utilisateurs créent leur compte en enregistrant leurs informations de base : leur pseudo, leur genre, leur date de naissance, leur pays d'origine et leur langue maternelle. Ces informations ne peuvent pas être modifiées par la suite.
- Les utilisateurs remplissent dans leur profil une biographie décrivant leurs centres d'intérêts. Ils peuvent également indiquer s'ils souhaitent pouvoir être contactés par d'autres utilisateurs pour initier une discussion dans leur langue maternelle. Ces informations pourront être modifiées par la suite.
- Les utilisateurs peuvent effectuer une recherche pour trouver des profils correspondant à leurs critères : langue maternelle, âge minimal/maximal, genre, et/ou pays d'origine. Les profils renvoyés doivent avoir indiqué être ouverts à de nouveaux contacts, et ne pas être déjà en relation avec l'utilisateur.
- Lorsque l'utilisateur a trouvé un profil qui l'intéresse, il peut lui envoyer une demande de contact. Le contact entre deux utilisateurs peut avoir les statuts suivants : "en attente", "actif" et "fermé".
- Lorsqu'une demande est envoyée, elle est initialement dans le statut "en attente". L'utilisateur qui a reçu la demande de contact peut l'accepter, ce qui lui donne le statut "actif", ou la refuser, ce qui lui donne le statut "fermé". Quand un contact est actif, l'un ou l'autre utilisateur peut décider d'y mettre un terme, ce qui lui donne le statut "fermé".
- Un utilisateur peut également consulter la liste de ses contact "actifs", ainsi que la liste de ses contacts "en attente".
- Deux utilisateurs qui ont un contact actif peuvent s'envoyer des messages. La langue de discussion entre deux utilisateurs correspond à la langue maternelle de l'utilisateur qui a reçu la demande de contact.
- Un utilisateur peut récupérer la liste des messages envoyés et reçus avec un autre utilisateur, triés par leur date d'envoi.
- Les utilisateurs ont la possibilité de supprimer leur compte. Cette suppression n'efface pas les données, le compte est seulement marqué comme "désactivé". De plus, l'utilisateur n'est alors plus ouvert à de nouveaux contacts et tous ses contacts actuels passent au statut "fermé".

Vous utiliserez le framework Spring pour développer le backend de votre application en utilisant les concepts et techniques vus au cours. A cet effet, vous utiliserez une architecture micro-services et documenterez l'API de tous les services et toutes les routes créées avec OpenAPI 3.0. Il n'est pas nécessaire de développer un frontend pour votre application.

Documentation

Votre première tâche est de concevoir une découpe en micro-services. Cette découpe devra respecter les principes vus au cours. L'API de chaque service devra être spécifiée au format OpenAPI 3.0.

Implémentation

Les services devront être implémentés avec Spring. Chaque service devra être accompagné d'un fichier de test http comprenant suffisamment de requêtes pour tester efficacement toutes les routes du service.

Organisation

Ce projet est à réaliser seul. Nous utiliserons des logiciels de détection du plagiat pour repérer toute triche. Vous pouvez réutiliser des parties de code développées par vous ou les membres de votre groupe durant le projet en première session. Indiquez-le visiblement en commentaire lorsque c'est le cas.

Le projet est à remettre le **vendredi 16 août à 20h** au plus tard sur moodle. Compressez la documentation et l'implémentation des différents services en une archive ZIP. Incluez également un document PDF donnant une courte présentation des responsabilités de chacun de vos micro-services et comprenant un schéma des relations entre ces services.

Evaluation

Vous serez évalués sur la documentation et de l'implémentation des services, chaque partie comptant pour 50% de la note. Les critères sont les suivants :

Documentation et architecture :

- La documentation respecte les besoins exprimés par le présent document.
- Les services sont spécifiés correctement suivant la spécification OpenAPI.
- L'architecture respecte les principes de conception en micro-services vue au cours et notamment le principe de responsabilité unique.

Implémentation :

- L'implémentation des services respecte leur spécification.
- Le code est de qualité :
    - Documenté
    - Utilise Spring correctement
    - Respecte les conventions Java
    - Les tests sont présents, pertinents et corrects pour chaque service