Dossier technique application Uptalk

Tables des matières

1. Présentation du projet :
   1. - Description de l’objectif
   2. - Description de l’application
   3. - Architecture de l’application
2. L’application :
   1. Architecture :
      1. Page de registration et de connexion
      2. Page de création de profil
      3. Page d’accueil
      4. Page de like
      5. Page de profil
      6. Messagerie
   2. Base de données
3. Améliorations futures

Presentation du projet

# Description de L’objectif

Une image contenant texte

Description générée automatiquement

Presentation du projet

# Description de l’application

Lien du GitHub : https://github.com/timothyWu1/Uptalk

L’objectif du projet est de réaliser un site de rencontre basé sur l’exemple de tinder. La différence réside lors de la sélection des profils qui ne contient aucune photo.

L’idée est d’avoir une alternative a Tinder plus orienté sur le caractère des personnes, chose qu’on ne trouve pas ou peu sur les applications de rencontre.

L’utilisateur peut créer un compte. Il indique son nom, son prénom, son courriel, son mot de passe, sa date de naissance, son sexe et sa photo.

Ensuite il indique ses préférences (localisation, Age, préférence sexuelle).

Par la suite l’utilisateur doit remplir son profil, il rentre une biographie, choisit ses centres d’intérêt depuis une liste définie et choisit de répondre à 3 questions parmi une liste de questions définies.

Viens par la suite la sélection de profil :

Un profil est affiché, l’utilisateur peut voir sa biographie, ses centres d’intérêt et les questions auquel l’utilisateur a répondu.

Comme sur Tinder il peut choisir de liker ou non un profil.

Les profils seront affichés en fonction des préférences de l’utilisateur, mais aussi de ses centres d’intérêt et des profils qu’il a précédemment sélectionnés.

Si l’utilisateur like un profil, ce profil auras plus de chance d’apparaitre sur l’algorithme de l’autre utilisateur.

Si deux utilisateurs se sont mutuellement likés, alors il y a un match.

Ils peuvent alors communiquer via une page de messagerie.

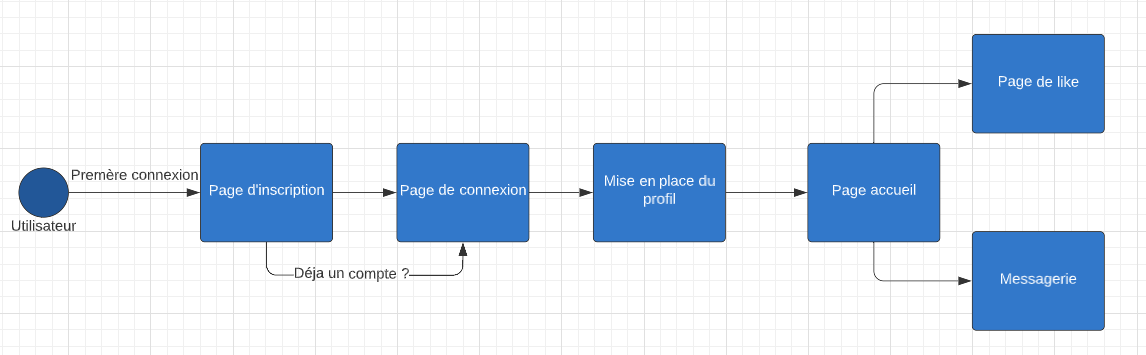
Au bout d’un certain nombre de messages échangés l’application propose aux utilisateurs de d’échanger leurs photos.

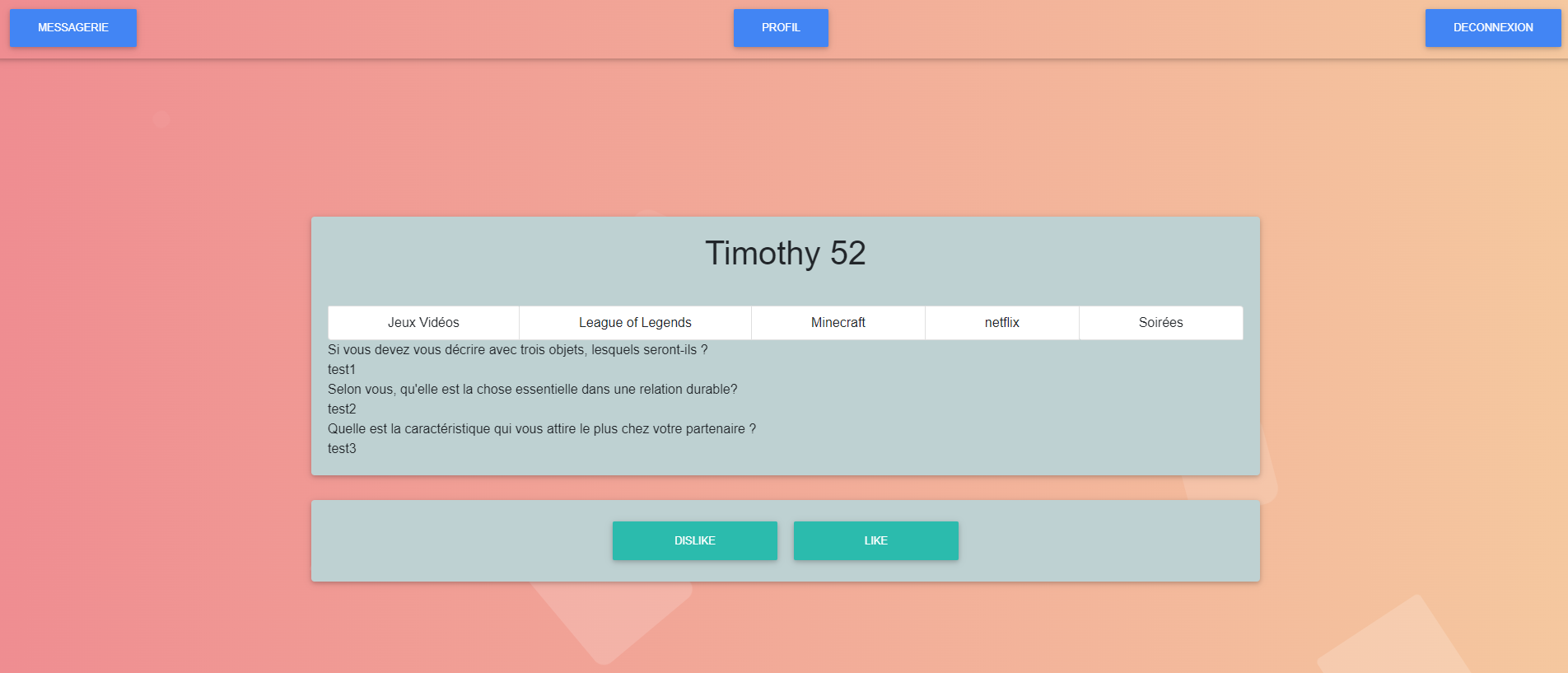
Ensuite les utilisateurs sont libres de continuer à communiquer ou non.

Presentation du projet

# Architecture de L’application

Structure de l’application en page :



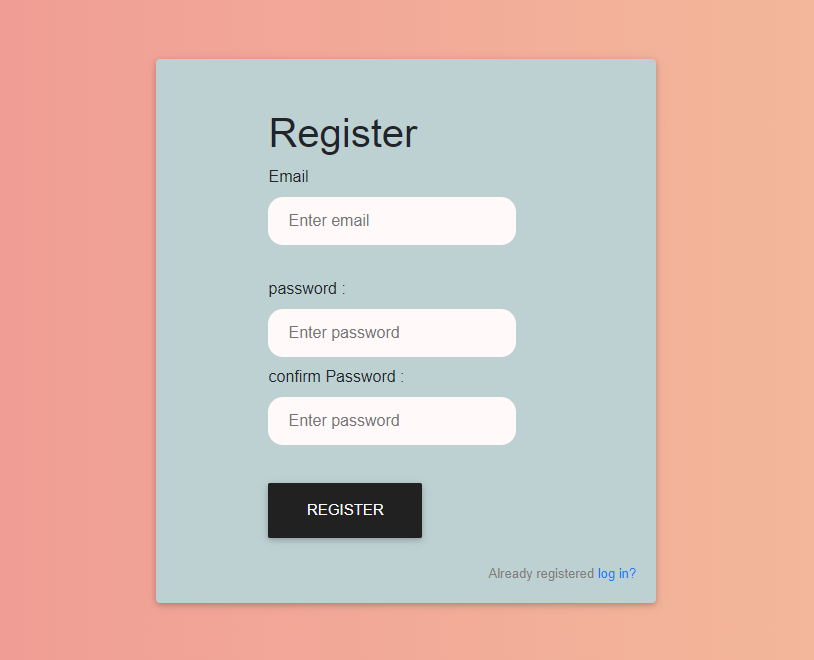


Presentation de l’application

# architecture : page de INSCRIPTION et connexion

La page de création de compte demande uniquement de renseigner un mail ainsi que d’un mot de passe qui seront stockée dans la base de données comme moyen de connexion. La connexion par mail et mot de passe sera pour l’instant le seul moyen de se connecter. La page de création comportera aussi un lien vers la page de connexion pour les utilisateurs ayant déjà un compte.

La page de connexion se voit demander le mail et le mot de passe de l’utilisateur pour s’authentifier et entrer sur le compte de l’utilisateur. Elle comporte aussi un lien en pied de page vers la page d’inscription pour les utilisateurs sans compte. Quand un utilisateur souhaite se connecter, il envoie ses informations de connexion sur l’API qui va vérifier les informations. Si les informations sont valides, l’api va renvoyer un token d’authentification a l’application. Ce token permettra a la base de données de valider toutes les requêtes de l’utilisateur afin de permettre de protéger les informations sur la base de données.



Une image contenant texte, capture d’écran, carte de visite

Description générée automatiquement

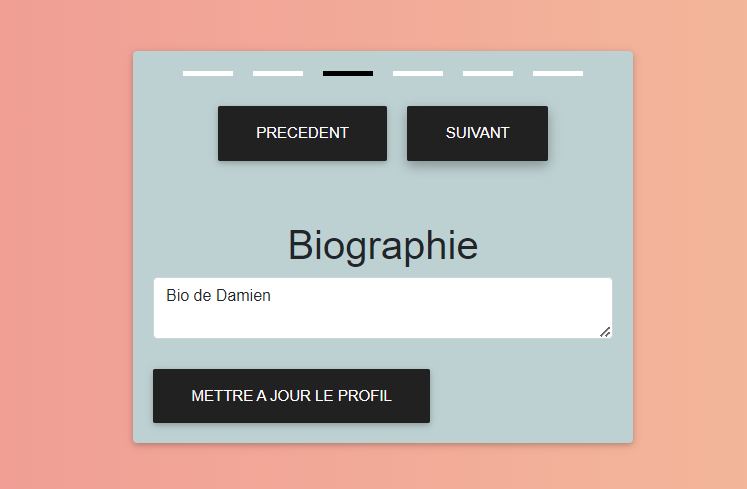
Presentation de l’application

# architecture : page de création de profil

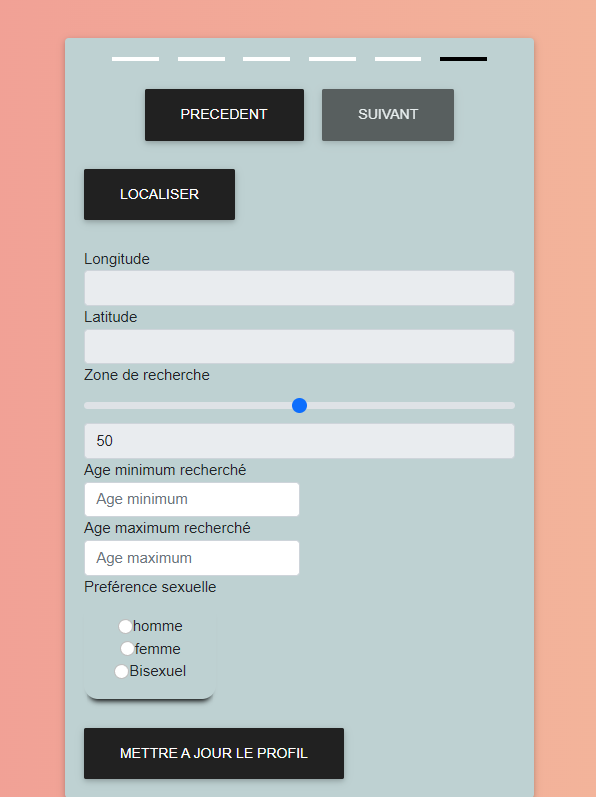
Les pages de création de profil permettent à un nouvel utilisateur de créer son profil qu’il souhaite que les autres utilisateurs puissent voir. Pour cela, le nom et le prénom de l’utilisateur seront demandées dans un premier temps, puis le sexe, la date d’anniversaire seront requis. Et enfin le profil recherché et une biographie.

L’utilisateur aura accès par la suite à la seconde partie de création de profil. L’utilisateur devra choisir un certain nombre de centre d’intérêt puis répondre à un certain nombre de questions préconfigurées qui se verront être afficher dans son profil vu par les autres utilisateurs.



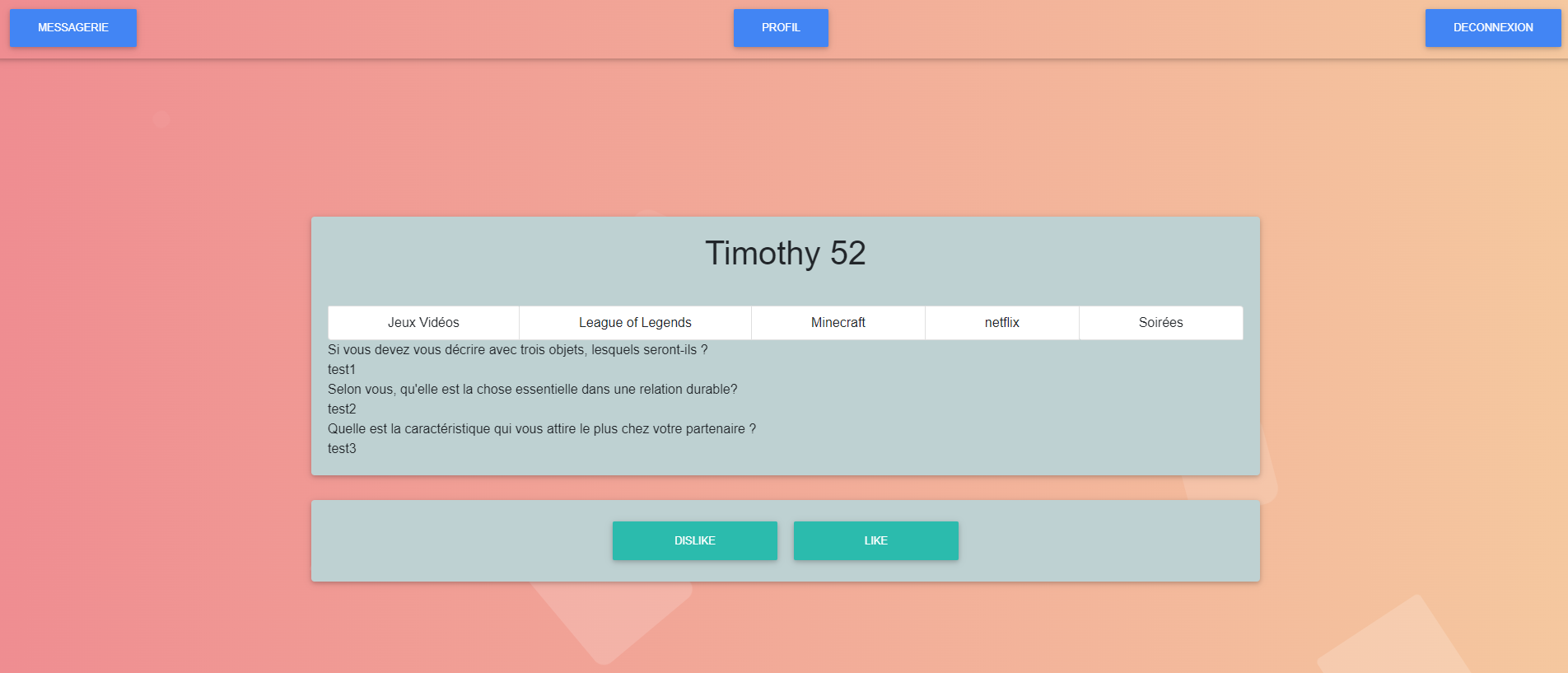


Presentation de l’application

# architecture : page de like

Cette page contient un des aspects les plus importants de notre application, celle de la sélection des autres utilisateurs. L’utilisateur pourra voir le nom et prénom de l’autre utilisateur et lire sa biographie ainsi que les réponses aux questions et ses centres d’intérêt. L’utilisateur pourra ensuite faire le choix ou non de liker le profil de la personne si le profil lui plait.

Si deux utilisateurs se likes l’un l’autre, il y a un match. Les utilisateurs sont donc ajoutés dans leurs listes de contacts respective.

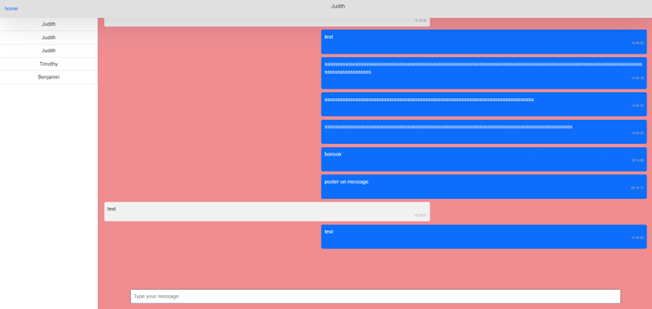


Presentation de l’application

# architecture : page de MESSAGERIE

Sur la page de messagerie, les utilisateurs peuvent voir leurs contacts et discuter avec eux.

Nous avons implémenté un système de sockets afin que les utilisateurs puissent communiquer en temps réel sans actualiser la page. Dès que le serveur reçoit une requête d’ajout de message, il ajoute celui-ci a la base de données, puis il envoie une requête au destinataire du message s’il est connecté en écoute sur le serveur.



Presentation de l’application

# base de donnée

Le type de base de données utilisée sera MySQL qui sera reliée à l’application en passant par une API Node JS qui permettra de lire et écrire les données d’un utilisateur via des routes. Cette base de données se verra dynamique et enregistrera toutes les données de chaque utilisateur.

Si un utilisateur veut lire des données sur la base de données (BDD), il envoie une requête de type GET a l’API. Si l’utilisateur veut un objet spécifique, il envoie une requête contenant l’ID de l’objet dans les paramètres de la requête.

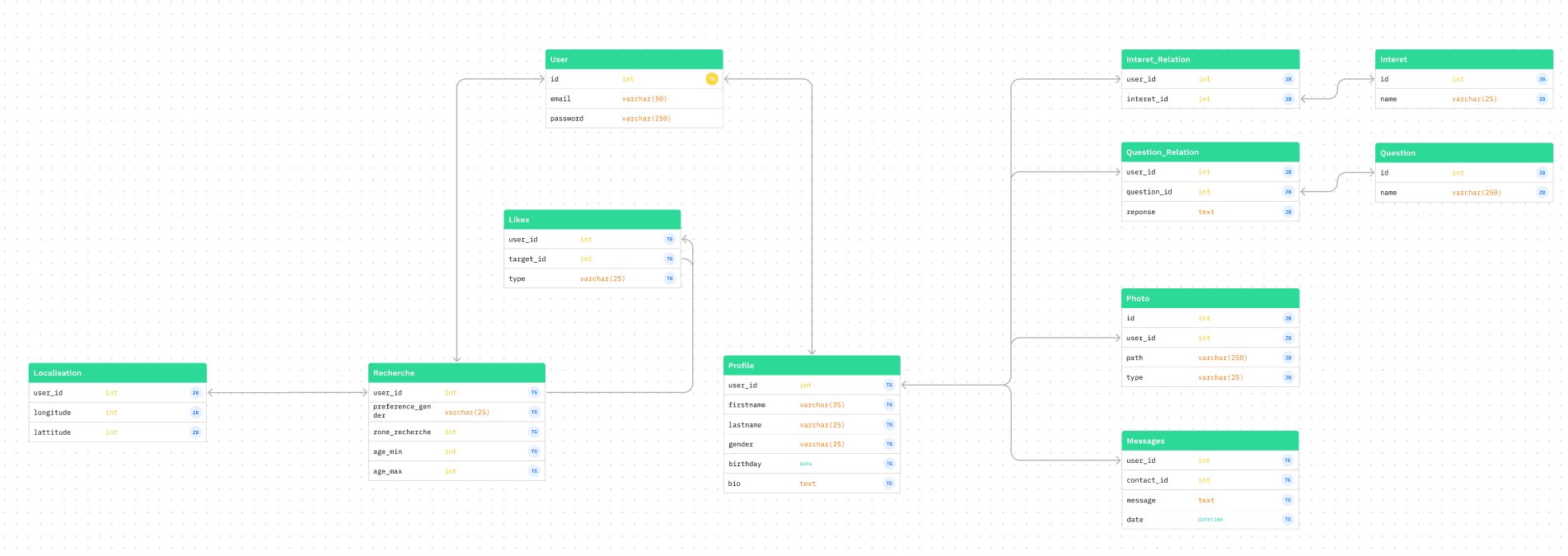
Une image contenant table

Description générée automatiquement

Pour envoyer une requête au niveau de la base de données l’utilisateur doit obligatoirement être connecté car il doit posséder la clé d’authentification permettant au serveur de valider la requête.

Une fois que la requête est récupérée par l’API, elle va chercher les données dans la BDD et l’API va les renvoyer. Si ces données n’existent pas, l’API renvoie un message d’erreur.

Architecture de la base de données :



Amelioration futures

Nous prévoyons de déployer l’application d’ici quelques temps et d’améliorer l’esthétisme de celle-ci.

Nous avons aussi pensée à sortir une version mobile puisque React Native est un langage qui permet cela.

L’idée de pouvoir personnaliser son profil est aussi à mettre en forme. Pouvoir choisir sa couleur de page, un mode nuit et plusieurs moyens de connexion

Nous aimerions aussi ajouter un système de carte pour pouvoir voir la zone de recherche en temps réel dans la page de modification du profil

Ajouter un business modèle au site (exemple : abonnement payant, vente de données…)