Analyse TFE - WorkMatch - OZKARA Ozkan

Use cases :

- Page d'entrée avec bouton de signup et de login
- Page de login avec mail et mdp demandé
- Page de signup avec un email, un username et un mot de passe demandé

Onboarding :

- Page suivante : demande de "chercheur d'emploi" ou "entreprise" ?
- Si entreprise, nouvelle page avec numéro de vérification unique fournie car toutes les entreprises peuvent avoir que 1 seul compte (sauf exception, amélioration)
- Si chercheur d'emploi (individu), nouvelle page avec âge demandé
- Si chercheur d'emploi (individu), nouvelle page avec photos demandé

Main page :

- Ensuite redirection vers page principale avec les offres d'emploi (info et bouton oui non et swipe à droite ou swipe à gauche), également un bouton profil en haut et chats. Si entreprise, bouton ajouter une offre d'emploi et bouton voir mes offres d'emploi en haut

Profile page :

- Page avec profile contient photo et deux boutons , editer profile et settings

Edit page :

- Page editer profile : ajouter/supp photos et changer bio

Settings page :

- Page settings : pour filtrer distance offre d'emploi ou individu, ou autoriser ou pas notification push , possibilité de déco, supprimer profile ou faire une pause, nous contacter etc…

Chat page :

- Page chatting : contient les matchs et les discussions en cours , possibilté de mettre favoris etc

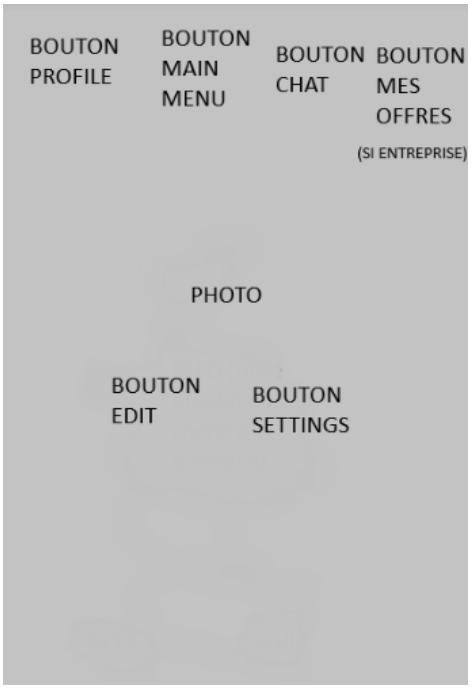
Mes offres d'emplois page :

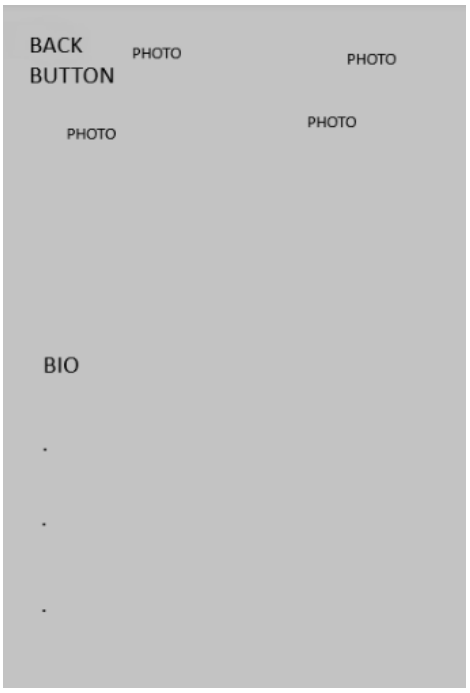
- Liste d'offres d'emploi avec possibilité d'agrandir pour voir info et les users qui ont matchés

Maquettes :


WORK MATCH
LOGO
SIGN UP
LOGIN


BOUTON PROFILE
BOUTON MAIN MENU
BOUTON CHAT
BOUTON MES OFFRES
(SI ENTREPRISE)
PHOTO
INFO (offre emploi ou du chercheur d'emploi)
NON
OUI


BOUTON PROFILE
BOUTON MAIN MENU
BOUTON CHAT
BOUTON MES OFFRES
(SI ENTREPRISE)
PHOTO
BOUTON EDIT
BOUTON SETTINGS


BACK BUTTON
PHOTO
PHOTO
PHOTO
PHOTO
BIO
.
.
.

BACK BUTTON

SETTING 1

SETTING 2

SETTING 3

LOGOUT

DELETE ACCOUNT

FAIRE UNE PAUSE

BACK BUTTON

MATCHES LOGO 1 2 3 4

CHAT LIST

1

2

3

BACK BUTTON

LISTE OFFRE D'EMPLOIS

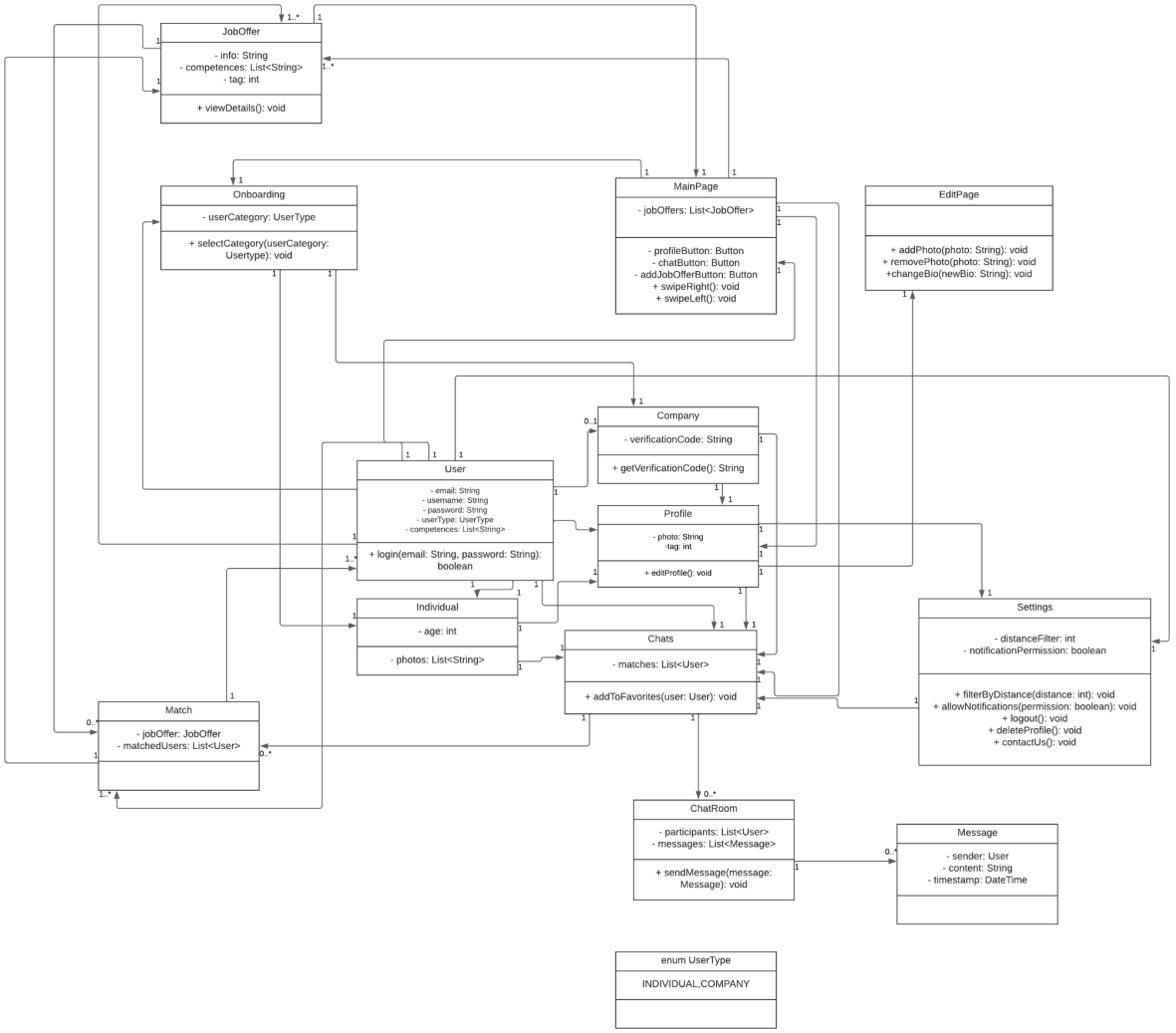
1

2

3

4

Diagramme des classes :



Techno utilisées :

JAVA backend, react native front et pour le cross platform , expo : pour construire, déployer et gérer app react, mongodb pour la BDD, elastic search pour recherche données users, redis comme système de cache pour la perf, firebase pour l'auth = non car trop simple, pour auth, notif et chat plutot -> jwt ou utiliser syst de token ou 0Auth2, utiliser websocket pour le chat, notifications etc…

Conditions supplémentaires:

- Utilisation des données réelles (que je peux prendre dans un api comme celle d'indeed.com), seulement pour offre d'emploi (1000) , aux moins 200 users créé pour le jour de la défense (seed db), pouvoir expliquer que ca peut fonctionnner avec un , api gratuite, voir cmt intégrer offre d'emploi , voir le format et comment intégrer dans mon app

- ensemble de données, avoir des diff offre d'emploi sur domaine diff, utiliser émulateur sur pc pour projeter sur écran
- Système matching de compétences pour optimiser les demandes entre employeurs/employés, ajouter un tag pour chaque profil ,
- l'user coche les compétences qu'il a et créer un algorithme pour matcher les bonnes compétences avec les bons offres d'emplois , il faut pas que nptq personnes ait nptq offres
- user et offre d'emploi devrait avoir une liste de label/compétence sur base desquels on fera des matchs
- utiliser emulateur lors de presentation sur pc

Par ailleurs, voici quelques consignes générales (à prendre en compte si elles sont pertinentes pour ton travail) :

- Les applications Web et les backends doivent être déployés auprès de (au choix) :
  - Un hébergeur gratuit, par exemple :
    - Heroku, en hébergement spécifique ou via Docker : un accès gratuit vous est fourni via GitHub Student Developer Program
    - un conteneur ou une machine virtuelle sur Azure for Students (attention à ne pas dépasser le crédit gratuit)
    - ou une machine virtuelle chez Oracle Cloud Always Free
  - Un hébergeur payant (exemples : OVH, Hostinger)
  - Une infrastructure personnelle (exemple : via un nano-ordinateur avec un serveur web accessible depuis l'extérieur de ton réseau privé)
- Si, dans ton application, il est pertinent d'intégrer des fonctionnalités de base telles que login, logout, signup, chat ou relations d'amitiés, tu dois bien entendu les implémenter. Cependant, sache qu'elles ne seront pas prises en compte pour évaluer la richesse fonctionnelle de ton travail.
- Ton application ne doit pas se résumer à un ensemble d'opérations CRUD réalisées dans des listes ou des tableaux sans liens entre eux. Tu dois prévoir et implémenter des règles, des comportements et des traitements métier complexes. Soigne l'ergonomie de l'application en veillant à offrir à tes utilisateurs une interface riche, intuitive et fluide.