Rapport Technologie Web

ZENG Fanxiang 28600693

WANG Zhe 2116827

1. Répartition du travail

Dans le cas de notre binôme le travail était très bien répartis. Fanxiang travaillait plus dans le côté serveur, avec l'implémentation des API et effectué les requêtes BD nécessaires pour une bonne interaction client-serveur. Zhe de son côté crée les pages et les components besoin du côté client avec les codes html et css écrite entièrement à la main. Du coup, Fanxiang faisait principalement le parti serveur, Zhe le client et les partis où on avait besoin de faire la liaison client-serveur on était toujours à deux dessus, pour faire les fonctions nécessire. Nous deux a trouvé que c'était un plaisir de travailler avec l'autre en tant que binome, on arrivait toujours à trouvé un temps pour travailler ensemble, et dans le cas ou l'un de nous deux rencontre un problème on se met à deux dessus pour résoudre. Cela fait que tous les deux connaissent bien le projet dans sa totalité.

2. Problème qu nous avons rencontré

- Comme on est entière novice pour react, au début quand on commençait le serveur, ça nous a posé beaucoup de problèmes que le cookie ne s'échange pas bien entre le client et le serveur. Mais cela a été résolu facilement avec l'ajout de `withCredentials: true` à la création d'axios
- La gestion d'asyncronicité nous a posée beaucoup de problème, souvent la page s'affiche avant que `axios` a récupéré les données de puis le serveur, nous obtenons des pages vides dans ce cas car certain de nos `state` sont `undefined`. Nous avons régler ce problème par des méthodes pas très élégante, en faisant des raffrachissements des pages si les `state` que nous avons besoin est indéfinie.
- Le comportement de `Link to` du react-router nous a posé de problème lorsque on voulez passer de la page profil d'un autre utilisateur sur notre profil. En étant deja sur le path `/profil` ce dernier n'a pas fait d'action lorsqu'on voulez aller sur notre page profil de path `/profil` également. Pour résoudre ce problème nous avons utilisé une méthode pas très élégante, c'est de redéfinir une page `MyProfil` afin de distinguer le page profil accédé par le `NavBar`, et les pages profil accédés via au messages ou via au recherche des utilisateurs.
- Des problèmes du fait qu'on utilise des `class` pour faire le projet, de nombreux `hook` pratique en fonctionnel c'était pas utilisable pour nous, et certaine fonctionnalité que nous voulons utilisé pour les class était enlevé pour la nouvelle version de React comme `withRouter`. Nous avons découvert une fois qu'on a bien avancé dans le projet que de procéder en fonctionnel était un meilleur choix pour faire du react.
- Des petits bugs lorsque des raffraichissements des pages, des fois la page s'affiche avant que `axios` renvoie les résultats depuis la base de donnée.

3. Ce qui reste à faire

Dans notre cas, les 5 services de base ,authentification, messages, amis, recherche et commentaires sont faites, mais ça prennait vraiment beaucoup de temps à tous les faires qui a pris beaucoup de jour sur les

vacances et réduit fortement notre temps pour réviser l'examen. Ensuite les fonctionnalité qu'on aimerais implémenter, sont des messages avec des likes et sur `Homepage` dans la zone gauche de mettre les messages qui ont le plus de nombre de like.

4. Fonctionnalités ajoutées

- possibilité de choisir votre propre photo profile, en l'uploadant sur le serveur
- possibilité d'envoyer des messages avec des images

5. Choix de modélisation

Pour le choix de modélisation de la BD, nous avons décidé de faire seulement un `api` pour user, et `apimessages` pour messages. On a pas fait de `apifriend`, la liste des `following et followers` sont directement dans user. Nous avons fait ce choix car c'était la méthode la plus simple, sinon dans `apifriend` on devait gérer cela en passant par les `login` ou `_id` des utilisateurs qui va complexifier les choses.