ECE PARIS
ÉCOLE D'INGÉNIEURS

Objectifs et Projet

2020/2021

PROJET : ECchat

Ce projet a pour sujet la reproduction d'un tchat à la manière de Discord/Whatsapp/Keybase (de manière très simplifiée)... Le but est de fournir une application Web permettant d'accéder à des channels, d'écrire des messages, ...
L'utilisateur pourra :

- S'authentifier auprès de l'application soit en se connectant sur une compte existant avec un provider externe et OAuth (groupes 1 et 3) soit avec un compte local et un formulaire d'inscription (groupes 2 et 4).
- Naviguer au travers de ses channels et des messages du channel sélectionné.
- Partager l'accès au channel avec d'autres utilisateurs.
- Accéder aux channels auxquels il a été ajouté.
- Envoyer un nouveau message.
- Modifier et supprimer **son** message.
- Automatiser l'association de leur email avec leur gravatar, sélectionner un avatar proposé par l'application et uploader son propre gravatar.
- Système de préférence de compte.
- Bénéficier d'une application sécurisée dans laquelle l'accès aux ressources est vérifié (authentification et autorisation).

David Worms / Kévin Frydman

Rendu

Votre projet est à faire par groupe de deux, au sein d'un même groupe de TD.

Votre projet devra être déposé sur campus avec le nom suivant : **Gr0X_Nom1_Prenom1_Nom2_Prenom2.rar**

Exemple :
Gr01_Frydman_Kevin_Worms_David.rar

Vous ne devez pas inclure dans votre archive les dossiers node_modules
Votre dossier aura la structure suivante :
Gr01_Frydman_Kevin_Worms_David
- back-end (l'API avec le serveur Express et la base de données LevelDB)
  - package.json
  - ...
- front-end (l'application client en React)
  - package.json
  - ...

Tout non respect de ces consignes se verra pénalisé d'un -6

Deadline

Dimanche 20 décembre 23h55

Evaluation

Attention à bien respecter les spécifications pour les fonctionnalités demandées.
Vous devez respecter les conventions de nommage de variable comme vu en cours cf le fichier sur campus.

Tout plagiat sera sanctionné d'un 0 et d'un avertissement

Interdiction d'utiliser :
- d'autres frameworks ou librairie autre que React (Vuejs, Angular, …),
- tout autres bases de données que LevelDB qui n'est pas du type Column Family ou Sorted Key Value Set
- des templates React "tout fait".

Il y a des points pour la qualité du code dans le barème !!

Vous serez évalué sur les fonctionnalités. Cela doit marcher !
Vous devez développer des fonctionnalités coté serveur et coté client
À vous de réfléchir sur les éventuelles spécifications (blindage, gestion des erreurs, etc)

Barème (à titre indicatif)

Gestion de projet

Respect des conventions de nommage	2
Structure des projets simple, compréhensible et stable, organisation des dossiers, services, composants	4
Qualité globale du code (indentation, clarté, …)	4
Apparence globale de l'application web	4

Développement

Formulaire de Sign in, création du compte (moyen)	4
Si vous proposez aux utilisateurs une alternative à OAuth par la création d'un compte local, créer le formulaire permettant de s'inscrire.	
Ecran de bienvenue, de login, d'accueil sur l'espace privé (simple)	2
Faite en sorte que les écrans de bienvenue lorsque l'utilisateur arrive sur l'application et une fois qu'il se connecte soient accueillants et fournissent des informations pertinentes sur le service.	
Création de nouveau channels (dur)	6
Insertion d'une action (eg bouton) permettant la création d'un nouveau channel, création du formulaire avec ses propriétés (nom, membre, …) et enregistrement du channel.	
Associer la création de channels avec l'identité de l'utilisateur en cours (moyen)	4
Toute requête sécurisée envoyée au serveur doit contenir l'access token de l'utilisateur dans l'en-tête HTTP. Une fois le token validé dans le middleware d'authentification, l'identité de l'utilisateur est connue (son email) et l'ID de l'utilisateur doit être associé aux channels créés. Si l'utilisateur n'existe pas encore dans la base de données, il doit être créé.	
Restreindre l'accès aux channels et aux messages (moyen)	4
Un utilisateur ne doit pouvoir visionner que les channels qu'il a créés et ceux pour lequel l'accès lui a été ouvert. Les APIs doivent être adaptés pour ne retourner que les channels	

en question. L'accès à d'autres channels doit être validé et retourner le code HTTP approprié en cas d'intrusion.	
Ajout d'utilisateurs à un channel (dur)	6
Possibilité d'ajouter voir supprimer des utilisateurs associés à un channel, soit au moment de la création du channel soit plus tard.	
Modification des messages (simple)	2
Une fois le message envoyé et partagé, l'auteur du message doit pouvoir modifier son contenu.	
Suppression des messages (simple)	2
Une fois le message envoyé et partagé, l'auteur du message doit pouvoir le supprimer.	
Préférence de compte (moyen)	4
Création d'une page permettant à l'utilisateur de modifier des propriétés personnelles (email, nom, langue, ...). Ce n'est pas grave si les propriétés ne sont pas actives, utilisez par contre des éléments de formulaire différents (input, dropdown, switch, …)	
Affichage du Gravatar (facile)	2
Associer l'email de l'utilisateur avec son compte Gravatar (https://en.gravatar.com/). Si aucun compte gravatar n'est trouvé, proposer un avatar générique ou aléatoire. Il existe des librairies React qui font juste ça.	
Sélection d'un avatar (moyen)	4
Proposer à l'utilisateur de sélectionner parmi plusieurs avatars, par exemple à l'intérieur d'un écran de préférences utilisateurs.	
Avatar personnel (dur)	6
Proposer à l'utilisateur la possibilité de télécharger son propre Avatar sous forme de png, svg...	

Bonus

- Système de temps réel avec https://socket.io/
- Gestion d'administrateur/super administrateur qui peut par exemple, supprimer tous les messages, modifier les channels, supprimer des channels, bref soyez créatif !
- Système permettant de gérer des smileys
- Remplacer LevelDB par une alternative distribuées (HBase, Cassandra)
- D'autres idées ? :)

Les bonus ne sont comptabilisés que **si et seulement si** les fonctionnalités de base sont mises en place et **fonctionnelles**