PROJET SYNTHÈSE - GROUPE 1017

PRÉPARÉ POUR

Vincent Baylly

Cégep de Trois-Rivières

PRÉPARÉ PAR

Mathieu Thériautl

09 AVRIL 2021

VINCENT BAYLLY
CÉGEP DE TROIS-RIVIÈRES

3175 BOULEVARD LAVIOLETTE
TROIS-RIVIÈRES, G8Z 1E9

Ce projet est l'épreuve synthèse et l'accomplissement de notre AEC en Développement Front-End. Il représente et culmine toutes nos connaissances apprises à ce jour durant notre programme. Durant ce projet, j'ai pu expérimenter avec certaines techniques et concepts de programmation comme la memoization, les closures ainsi que la gestion d'états globaux. Le projet doit être remis pour le 13 avril 2021. Durant ce temps imparti, il consiste à mettre en place, en utilisant un framework de notre choix (React, Vue ou Angular) une application d'application et d'affichage de stages. Vous pourrez voir dans les prochaines sections, les différentes librairies, techniques et structures utilisées.

SOMMAIRE

L’application consiste à pouvoir afficher ou appliquer sur des stages. Un utilisateur peut se créer un compte et définir s’il agit pour une entreprise ou à son compte en tant que futur stagiaire. Ce choix est obligatoire durant le processus d’inscription. L’utilisateur peut ensuite contacter ou consulter les offres de stages (futurs stagiaires) ou le profil d’étudiants (entreprises). L’étudiant peut postuler sur les offres de stage. Il ne peut postuler qu’une seule fois. Lorsque l’étudiant a postulé sur une offre de stage, l’entreprise reçoit un message par le billet de la messagerie intégrée de l’application. L’entreprise peut, à ce moment, voir qui a envoyé le message, consulter le profil et contacter l’étudiant à même la messagerie de l’application, si elle est intéressée. L’entreprise peut ajouter, modifier et supprimer des offres de stages. Il n’y a pas de minimum. L’étudiant, pour sa part, peut modifier son profil.

1. Technologies utilisées

- React comme framework avec Create React App pour construire la structure initiale de l'application
- Typescript
- React Router V6 pour la navigation côté client
- UUID pour la génération de clés uniques
- SASS avec le compilateur Dart pour le pré-processeur de CSS
- Recoil pour la gestion des états globaux
- React Query pour la gestion de la cache, des requêtes et de la pagination des requêtes
- Bootstrap pour la génération de composant React rapide et l'utilisation de grilles
- Normalize.css pour remettre à zéro le CSS sur tous les navigateurs web
- Luxon pour la gestion de la date
- Lodash pour certaines fonctions performantes en Javascript permettant de rendre le développement plus rapide
- Framer Motion pour l'animation de composants
- Formik pour la gestion des états des formulaires
- Classnames pour la gestion facile des classes HTML optionelles
- Axios pour la gestion et l'accomplissement des requêtes API
- NodeJs pour le serveur
- Express pour le routage de l'API
- MongoDB utilisée en tant que base de données
- Mongoose utilisé comme enveloppe à la librairie MongoDB
- Bcrypt pour "hasher" et "salter" les mots de passes afin de les encrypter avant de les enregistrer dans la base de données
- Jest et React Testing Library pour les tests unitaires

2. Obstacles

- La création et la gestion des formulaires était plus complexe que prévu. En créant un formulaire qui gère lui-même la création des lignes et des champs, j'ai simplifié la création d'un formulaire mais complexifié la gestion des données déjà présentes. Il serait possible, si j'avais plus de temps, de créer un petit utilitaire qui pourrait gérer ces données plus facilement.
- La gestion des multiples requêtes à l'API interdépendantes. Certains composants requièrent certaines données. Ces données dépendent d'une autre donnée. React Query a grandement facilité cette gestion mais la complexité des composants s'en est trouvée amplifiée.

- Le temps relativement limité vu l'ampleur du projet. En effet, je n'ai pas pu mettre en place la totalité des fonctions que j'aurais voulu ajouter. Il me resterait encore le panneau d'administration, la gestion des mots de passe (changement de mots de passe suite à un oubli etc...), le "lazy-loading" des différents composants ainsi qu'une plus grande quantité de SASS afin de rendre l'application unique.

3. Obstacles techniques

- Mis à part des troubles avec l'environnement de tests, aucun obstacle concernant des possibles troubles techniques ont été rencontrés

7. Logiciels

- Visual Studio Code
- Chrome & Mozilla Firefox