Bisegna David

Rapport Programmation Client-side

Identifiant GitHub : davidbisegna

Tâches effectuées

- Création du tableau pour afficher les données, dans des colonnes que l'on peut trier par ordre croissant/décroissant, avec une pagination.
- Ajout de la fonctionnalités de thème, où l'utilisateur peut choisir une couleur, et choisir d'activer le mode sombre. Avec une fenêtre pop-up qui apparaît si jamais l'utilisateur utilise le mode sombre sur son ordinateur pour lui proposer de l'activer
- Ajout de la page de connexion, avec gestions des redirections si jamais l'utilisateur essaie d'accéder à des pages auxquelles il n'a pas accès.

Stratégie Git

Nous avons tout d'abord créé des tâches en fonction des besoins du projet, puis nous les avons partagées entre les membres du groupe. Ensuite, chacun créer une branche qui correspond à sa tâche/fonctionnalité en cours de développement, et une fois que le développement est terminé, la personne qui a créé la branche crée une Pull Requests. Une fois que la Pull Request est approuvée (bien souvent après quelques changements demandés par les autres membres du groupe), on merge cette Pull Request sur la branche develop.

Solutions choisies

Pour le tableau de données, j'ai choisi le composant DataGrid de MaterialUI, car il embarque déjà toutes les fonctionnalitées, de recherche, de tri, de pagination, ce qui correspondait exactement à mon besoin, car sans la pagination, lorsque qu'il y a trop de données le site a de gros problème de performance.

Pour les thèmes, j'ai compris que l'on pouvait dissocier le mode sombre de la couleur du thème, j'ai donc fait en sorte de pouvoir facilement ajouter des thèmes basique, avec une méthode générique qui prend en paramètre un booléen qui indique si le mode sombre est activé, ainsi qu'un objet Color de MaterialUI, et qui renvoi le thème créer, on peut donc facilement ajouter plusieurs thème. De plus, j'ai choisi de mettre le bouton de la gestion de thème dans la barre de navigation afin qu'on puisse tout le temps accéder à ce bouton. Celui-ci ouvre un Popover au clic, afin de gagner en espace, sans devoir ouvrir une Modal qui prend toute la page.

Bisegna David

Difficultés rencontrées

J'ai rencontré des difficultés lors de l'ajout de la gestion de l'authentification. On a utilisé un composant RestfulProvider, ce qui a été très utile, car j'ai pu capturer les erreurs directement dans le composant, ce qui me permet de rediriger l'utilisateur vers la page de connexion lorsqu'un appel au back-end renvoi un code HTTP 401 Unauthorized.

Temps de développement

- Création du tableau : ~3h
- Thèmes : ~5h
- Authentification : ~10h
- Bugfix : ~1h

Code

Authentification et redirection

Je trouve que le système de redirection lorsque l'on reçoit une erreur 401 est à revoir. Comme je n'avais pas accès aux méthodes du Router, je ne pouvais pas rediriger l'utilisateur vers la page de connexion. La bonne solution pour moi aurait été de créer un Context, qui contiendrait un état qui indique si l'utilisateur est connecté ou pas. Mais par manque de temps, j'ai fait un code moins élégant pour pouvoir arriver à mon but, j'ai créé une variable à côté du RestfulProvider, cette variable étant vide mais destinée à recevoir une fonction. Ensuite, j'ai créé une fonction qui prend en paramètre une fonction, et qui l'assigne à cette variable. Ensuite, je passe cette fonction en prop au composant App, puis dans le composant, j'ai accès au Router, j'ai donc créer une fonction qui redirige l'utilisateur vers la page voulue, puis j'ai appelé la fonction passer en prop, en donnant en paramètre la fonction de redirection. C'est fonctionnel, mais le comportement n'est vraiment pas clair au 1er coup d'œil, il faudrait donc vraiment remplacer cela par un Context.

Thèmes

Le système est assez bien fait à mon avis, même si améliorable en mettant un Context qui permet d'y accéder depuis n'importe quel composant, c'est faisable sans devoir changer trop de code donc je considère que la fonctionnalité de thème reste élégante même sans ça. Ce qui est intéressant, c'est que l'on peut ajouter des thèmes facilement, et on pourrait même étendre le code pour proposer à l'utilisateur d'utiliser une couleur qu'il sélectionne lui-même. De plus, l'application ne spamme pas l'utilisateur avec la pop-up qui propose de passer au mode sombre. Si l'utilisateur a déjà refusé au moins une fois de passer au mode sombre, on ne lui propose plus, et bien sûr on ne lui propose pas s'il est déjà en mode sombre.