Rendu Projet Final

Site Web Organiz-Asso

Stefano Defina 21110014, Melissa Setbel 21210442

Répartition du Travail

Dès le début du projet, nous avons choisi de simuler une organisation professionnelle en répartissant les rôles entre un développeur frontend (Stefano Defina) et un développeur backend (Melissa Setbel). L'objectif était de clarifier les responsabilités tout en favorisant une collaboration fluide entre les deux pôles.

- **Stefano Defina** a pris en charge le **développement frontend**. Il a conçu et implémenté les différentes pages du site avec React, HTML et CSS (WelcomePage, UserPage, Forum, etc.). Il s'est également occupé de l'intégration de l'API via Axios, assurant la communication client-serveur. Selon l'historique GitHub, Stefano a commencé à travailler activement dès avril et a continué de contribuer de manière soutenue jusqu'à la finalisation du projet, y compris sur des tâches backend comme la connexion serveur-client et des ajustements dans les contrôleurs.

- **Melissa Setbel** était responsable du **développement backend**, incluant l'écriture des routes, des contrôleurs, et l'interaction avec la base de données MongoDB. Ses contributions principales sont concentrées autour du 1er au 22 mai. Elle a mis en place des fonctionnalités fondamentales comme la gestion des utilisateurs et des messages. Cependant, certaines implémentations ont nécessité des ajustements importants par Stefano en fin de projet pour assurer l'intégration complète avec le frontend.

Problèmes Rencontrés

- **Manque de coordination et déséquilibre dans la charge de travail** : La répartition initiale des tâches n'a pas toujours été suivie de manière équilibrée, comme le montrent les commits GitHub. Stefano a pris en charge plusieurs éléments du backend initialement attribués à Melissa, en y apportant des corrections et en restructurant certaines parties pour assurer la cohérence globale. Cela a généré des ajustements supplémentaires et des allers-retours techniques, ce qui a pu ralentir notre progression. Nous en tirons une leçon importante sur la nécessité d'une meilleure coordination et communication au sein du binôme.

- **Petite confusion sur l'utilisation des méthodes HTTP** : Certaines requêtes ont été initialement formulées en GET tout en contenant un corps de données, ce qui n'est pas pris en charge par ce type de requête selon les standards HTTP. Une fois identifié, ce point a été ajusté en remplaçant ces appels par des requêtes POST, permettant ainsi une transmission correcte des données. Cette situation, fréquente dans les phases d'apprentissage, nous a permis de mieux comprendre les bonnes pratiques liées aux échanges client-serveur.

- **Difficultés d'intégration des composants partagés (CSS)** : Le style global a été difficile à maintenir, notamment pour les composants React partagés comme les listes de messages. Une absence de normalisation des fichiers CSS a mené à des incohérences visuelles.

- **Préparation de l'environnement de développement** : en raison de la diversité des technologies utilisées, la configuration du projet s'est révélée complexe.

Améliorations Potentielles

- Intégration d'une image de profil personnalisable
- Mise en place de forums avec thématiques variées
- Système d'amis, demandes et création de réseaux sociaux
- Ajout de pièces jointes (images, vidéos, documents) aux messages
- Fonctionnalité de redirection vers un message cité

Choix de Modélisation

- **Séparation entre `pages/` et `components/`** :
  Nous avons distingué les fichiers en deux grandes catégories :

  - Le dossier `pages/` regroupe les vues principales du site, telles que `WelcomePage.jsx`, `UserPage.jsx` ou `WaitingPage.jsx`. Ces fichiers représentent les pages complètes affichées selon les routes de navigation. Chaque page contient la logique d'agencement des composants, les appels au backend et la gestion d'état locale si nécessaire.

  - Le dossier `components/` contient des éléments réutilisables comme `MessageCard.jsx`, `MessageFilter.jsx`, `UserProfile.jsx`, etc. Ces composants sont conçus pour être intégrés dans différentes pages tout en conservant une responsabilité unique (affichage d'un message, d'un filtre, d'un profil, etc.). Cela favorise la réutilisabilité, la lisibilité du code, et la cohérence visuelle entre les différentes parties de l'application.

  Ce choix d'architecture correspond aux bonnes pratiques de développement avec React, où l'on sépare les **composants fonctionnels** (centrés sur l'affichage ou la logique réutilisable) des **pages de structure** (représentant des ensembles cohérents de contenu et de logique). Il nous a permis de maintenir un projet plus lisible, d'éviter la redondance de code, et de faciliter les tests ainsi que les ajustements visuels.

- **Page d'accueil (WelcomePage)** : conçue pour être élégante et moderne, et servir d'introduction cohérente à l'interface utilisateur.
- **Séparation claire entre les pages d'accueil et d'utilisation** : deux structures distinctes facilitant l'organisation des composants et la lisibilité du site.
- **Page d'attente post-inscription** : confirmation de la création du compte utilisateur en base de données, tout en informant sur l'attente de validation.
- **Composants dynamiques selon le rôle de l'utilisateur** : cette modularité rend l'interface plus ergonomique et facilite les tests.