Projet de Design Patterns

EasyCollab

Aboubacar HASSANE CHEKOU KORE, Théo JOFFROY, Louis LESNIAK

31 Décembre 2023

1 Rapport

1.1 Fonctionnalités réalisées

- **Création d'un compte**

  Formulaire accessible depuis le bouton "Créer un compte" sur la page d'accueil

- **Connexion à un compte existant**

  Formulaire accessible depuis le bouton "Se connecter" sur la page d'accueil

- **Création d'un document vide**

  Depuis le bouton "Créer un document" sur la page d'accueil, l'utilisateur doit préciser le nom du document.

- **Ouverture d'un document sauvegardé**

  Les documents créés par l'utilisateur apparaissent sur la page d'accueil lorsqu'il est connecté. Pour en ouvrir un, il suffit de cliquer sur son nom.

- **Modification d'un document**

  Une fois un document ouvert, il suffit de changer le contenu des cellules pour modifier le document. Il est aussi possible de changer le nom d'un document en cliquant sur le bouton "Renommer" qui lui est associé depuis la page d'accueil.

- **Sauvegarde d'un document**

  La sauvegarde des modifications est réalisée automatiquement à chaque saisie de caractère dans une cellule. Le changement de nom d'un document est sauvegardé lors de la validation.

- **Suppression d'un document**

  Possible en cliquant sur le bouton "Supprimer" associé au document depuis la page d'accueil.

- **Persistance des données**

  Toutes les données (utilisateurs et documents) sont sauvegardées dans un fichier côté serveur.

- **Partage d'un document sauvegardé avec d'autres utilisateurs**

  Il est possible de partager un document que l'on a soi-même créé à un autre utilisateur en saisissant son nom dans le champ associé à chaque document depuis la page d'accueil. On ne peut partager le document ni à soi-même ni à un utilisateur non inscrit. Les utilisateurs à qui l'on a partagé un document peuvent l'ouvrir (depuis la page d'accueil une fois qu'ils sont connectés), le modifier et ces modifications sauvegardées. Seul l'utilisateur qui a créé le document peut le supprimer.

- **Affichage des utilisateurs travaillant simultanément sur le même document**

  Tous les noms des utilisateurs ayant ouvert un même document simultanément sont visibles au-dessus du tableur.

- **Mise à jour en temps réel de l'affichage**

  Les modifications d'un utilisateur sont visibles instantanément pour tous les autres utilisateurs ayant ouvert le document.

1.2 Conception et difficultés rencontrées

Pour la partie inscription/connexion, nous nous sommes très largement inspiré du modèle vu en cours, ce qui a accéléré le développement. La persistance des données est réalisée grâce à un fichier au format *JSON*, l'interface avec ce format étant très simple d'utilisation en *Javascript*. Le temps réel est pris en charge grâce à la technique du *long polling*.

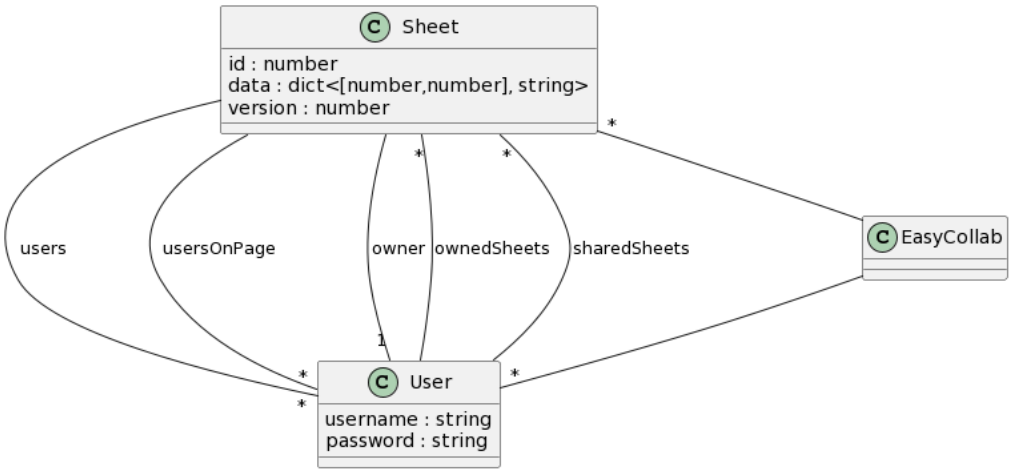

Figure 1: Diagramme de classes de l'application

.

La version présentée lors du point de suivi nécessitait de transmettre l'intégralité du document dans la requête effectuant la sauvegarde (seules les cellules vides n'étaient pas transmises). Il était difficilement envisgeable de conserver un processus aussi lourd dans le cas

où un document était volumineux, nous sommes donc parvenus à faire en sorte que seule la cellule modifiée soit transmise dans la requête (un listener est attaché à l'event *keyup* de chaque cellule et il déclenche l'envoi d'une requête paramétrée avec un objet contenant les coordonnées de la cellule ainsi que son contenu).

1.3 Déroulé du projet

Concernant la répartition du travail, Théo s'est occupé de la réalisation des fonctionnalités de base (Inscription, connexion, création d'un document, modification, sauvegarde et suppression), Louis s'est chargé du partage des documents entre les utilisateurs et de l'aspect temps réel et Aboubacar a travaillé sur le front-end de l'application.

2 Documentation

2.1 Guide d'installation

Spécifiez dans le fichier *.env* (ou *.env.local*) les champs `SECRET` et `PORT`. Le port par défaut est le port 3000, vous pouvez générer 64 bytes aléatoirement pour spécifier le champ `SECRET` à l'aide de la commande javascript :

```
require('crypto').randomBytes(64).toString('hex')
```

Ensuite, entrez dans votre terminal :

```
npm install && npm start
```

2.2 Connexion au serveur

Ouvrir la page : `http://localhost:PORT`
(où `PORT` est le port défini dans le fichier *.env* (ou *.env.local*), si vous n'avez pas défini de port, il est de 3000 par défaut.)

2.3 Description des routes

Il y a cinq méthodes de requêtes sur *EasyCollab* : `GET`, `POST`, `PUT`, `PATCH` et `DELETE`. Nous allons vous détailler l'intégralité des routes pour chacune de ces méthodes.

GET

- `/`

  Affichage de la page d'accueil du site.

  - Si l'utilisateur n'est pas connecté, il y verra une description du site et deux boutons : "Se connecter" et "Créer un compte".
  - S'il est connecté, il y verra la page de gestion de ses documents et les documents qu'il lui ont été partagés.

- `/account/signin`

  Formulaire de connexion au site (utilisateur et mot de passe).

- `/account/signup`

  Formulaire d'inscription au site (utilisateur, mot de passe et confirmation du mot de passe).

- `/account/signoff`

  Permet à l'utilisateur de se déconnecter.

- `/sheet/new`

  Formulaire de création d'un nouveau document (nom du document).

- `/sheet/:sheetId`

  Consultation d'un document avec l'identifiant `sheetId`.

- `/sheet/update/:sheetId`

  Non accessible par l'utilisateur, elle existe pour une raison technique (*long polling*). Cette route permet de recevoir les données du document partagé d'identifiant `sheetId` lorsque celui-ci a été modifié par un autre utilisateur.

- `/sheet/usersOnPage/:sheetId`

  Non accessible par l'utilisateur, elle existe une pour raison technique. Cette route permet à l'utilisateur de recevoir la liste des clients présents sur le document d'identifiant `sheetId`.

POST

- `/account/signin`

  Permet au serveur de traiter la requête de connexion lorsqu'on valide le formulaire (contient l'identifiant et le mot de passe hashé).

- `/account/signup`

  Permet au serveur de traiter la requête d'inscription lorsqu'on valide le formulaire (contient l'identifiant et le mot de passe hashé).

- `/sheet/new`

  Permet au serveur de traiter la requête de création de document lorsqu'on valide le formulaire (contient le nom du document à créer).

- `/sheet/subscribe/:sheetId`

  Permet à l'utilisateur d'envoyer au serveur qu'il est entrain de consulter le document d'identifiant `sheetId`.

- `/sheet/unsubscribe/:sheetId`

  Permet à l'utilisateur d'envoyer au serveur qu'il a fini de consulter le document d'identifiant `sheetId`.

PUT

- `/sheet/rename/:sheetId`

  Permet à l'utilisateur d'envoyer au serveur le caractère qu'il vient de taper dans le document d'identifiant `sheetId` (contient le caractère en question).

PATCH

- `/sheet/:sheetId`

  Permet à l'utilisateur d'envoyer au serveur le nouveau nom du document d'identifiant `sheetId` (contient le nouveau nom du document).

- `/sheet/users/:sheetId`

  Permet à l'utilisateur d'ajouter un utilisateur pouvant consulter et modifier son document d'identifiant `sheetId` (contient le nom d'utilisateur de la personne à ajouter).

DELETE

- `/sheet/:sheetId`

  Permet à l'utilisateur de retirer un utilisateur pouvant consulter et modifier son document d'identifiant `sheetId` (contient le nom d'utilisateur de la personne à retirer).