Amélioration du système d'information de la bibliothèque

Sommaire

1 – Introduction

Dans le prolongement du travail précédemment effectué pour la bibliothèque municipale, j'ai été missionné afin de traiter 3 tickets :

- la gestion des réservations d'ouvrages,
- l'optimisation d'une fonctionnalité en partie déjà présente,
- ainsi que l'élaboration d'une stratégie de tests unitaires et fonctionnels sur l'API REST.

NB : chaque ticket fait l'objet d'un versionning nommé par le numéro de ticket correspondant.

A l'issue de ce travail, un merge des 3 tickets est effectué sur le Main/Master.

Ticket 1

2.1 – Traitement des tickets

- **Ticket 1 : ajout d'un système de réservation d'ouvrages**
- **Côté site Web :**
  - Au cours de la recherche d'un ouvrage non disponible, afficher :
    - la date de retour la plus proche
    - le nombre de réservataires dans la liste d'attente
    - le bouton de réservation (si autorisé cf. slide n+1)
  - Liste personnelle des réservations en cours, afficher :
    - position dans la liste de réservation
    - bouton d'annulation de réservation

2.1 – Traitement des tickets

- **Ticket 1 : ajout d'un système de réservation d'ouvrages**
- **Côté API :**
  - Ouvrages tous réservables
  - Création d'une liste de réservations d'ouvrage
  - L'emprunt et la réservation s'excluent mutuellement
  - Nombre de personnes qui réservent < 1/2 du nombre d'exemplaires empruntés
  - Au retour d'un exemplaire, le réservataire en tête de liste ne dispose que de 48h, après notification, avant sa sortie de liste

2.1 – Traitement des tickets

- **Ajouts au code 1/3 : *revue de code***
  - Un nouveau model (+ service + controller) « PreLoan.java » pour les réservations
  - Il implémente principalement le getter :
    - getPreLoanExpiryDate() (date expiration à 48h)
    - https://github.com/OMorlotti/P10/blob/main/WebAPI/src/main/java/xyz/morlotti/virtualbookcase/webapi/models/PreLoan.java

2.1 – Traitement des tickets

- Ajouts au code 2/3 : ***revue de code***
  - Service « PreLoanServiceImpl »
  - API pour ajouter, modifier, supprimer des réservations
  - Implémentation de toutes les contraintes sur la réservation dans la méthode « addPreLoan »

    (voir détails dans le code ci-dessous)
  - https://github.com/OMorlotti/P10/blob/main/WebAPI/src/main/java/xyz/morlotti/virtualbookcase/webapi/services/impl/PreLoanServiceImpl.java#L74

2.1 – Traitement des tickets

- Ajouts au code 3/3 : ***revue de code***
  - Ajout de la réservation dans l'interface (vue) « search.pebble »
  - https://github.com/OMorlotti/P10/blob/main/UserWebSite/src/main/resources/templates/search.pebble#L96
  - Contraintes sur la réservabilité implémentées également dans la vue pour un comportement solide
  - Utilisation de JavaScript/Ajax pour le bouton d'ajout
  - Ajout du visionnage des réservations dans l'interface (vue) « user.pebble »
  - https://github.com/OMorlotti/P10/blob/main/UserWebSite/src/main/resources/templates/user.pebble#L115
  - Utilisation de JavaScript/Ajax pour le bouton supprimer

Ticket 2

2.2 – Traitement des tickets

- **Un usager ne peut prolonger (30j additionnels) un prêt après la date butoir (30j initiaux)**
- **La feature était déjà en grande partie implémentée dans la première mouture (P7) du logiciel**
- **La fonction getState(), dans le model « Loan.java », qui indique si la prolongation est autorisée, est restée inchangée**
- **Code initial (59bf3a80a7) :**
  - https://github.com/OMorlotti/P10/blob/59bf3a80a727375f984d871efc6830dd69ef979e/WebAPI/src/main/java/xyz/morlotti/virtualbookcase/webapi/models/Loan.java
  - https://github.com/OMorlotti/P10/blob/59bf3a80a727375f984d871efc6830dd69ef979e/WebAPI/src/main/java/xyz/morlotti/virtualbookcase/webapi/services/impl/LoanServiceImpl.java
- **Code final (depuis la branche Main en tenant compte d'une optimisation sur l'API réalisée dans ticket3 pour la rendre plus utilisable (bean APILoan.java)):**
  - https://github.com/OMorlotti/P10/blob/main/WebAPI/src/main/java/xyz/morlotti/virtualbookcase/webapi/models/Loan.java#L51
  - https://github.com/OMorlotti/P10/blob/main/WebAPI/src/main/java/xyz/morlotti/virtualbookcase/webapi/services/impl/LoanServiceImpl.java#L93

Ticket 3

• 2.3 – Traitement des tickets

- **Mise au point d'une stratégie de tests sur le code API :**
  - Tests unitaires via JUnit avec code coverage et dashboard Sonar (ici, les TU ne nécessitent pas une DB importante) :
    - Mocks inutiles => utilisation d'une DB minimale dédiée aux tests
    - *Démonstration*
  - Tests fonctionnels automatisés avec Postman :
    - *Même DB que pour les TU*
    - *Démonstration*
  - *NB : Mise en place d'un endpoint (admin only) RESET dans l'API REST pour resetter tous emprunts et réservations d'un utilisateur*