

## Partie 2 : mise en œuvre du développement dirigé par les tests

Nous allons réaliser des tests pour une application pour laquelle nous disposons seulement d'éléments d'architecture (incomplets) et de scénarios énoncés par les utilisateurs.

L'application est un système d'information pour une petite bibliothèque municipale.

Les abonnés (connus par leur nom, leur prénom et leur numéro d'abonné) peuvent réaliser différentes opérations sur le système. Le système d'information comprend un catalogue de livres décrivant le fonds documentaire, dont certains sont présents en plusieurs exemplaires. Ce catalogue est donné sous la forme d'un fichier (Fonds.csv). A noter que les ouvrages sont identifiés par leur numéro ISBN auquel il est nécessaire d'adjoindre un nombre, assigné automatiquement lors de l'achat d'ouvrages (1 puis 2, puis 3, ...), qui permet de distinguer les exemplaires du même ouvrage.

Le système enregistre les différentes opérations réalisées par les abonnés (emprunts, retours, réservations d'ouvrages) et maintient ainsi notamment à jour l'ensemble des livres disponibles à chaque instant (gestion du stock), leur « localisation » (identité de l'emprunteur, date attendue de retour), ainsi que des files d'attente pour les ouvrages les plus demandés (réservations). La date attendue de retour d'un livre est un mois exactement après l'emprunt (exemple : un livre emprunté le 16 Janvier 2026 de devra être retourné avant le 17 février 2026)



### Scénarios à tester (à compléter si vous le souhaitez):

S1- Marie Dupont cherche à se connecter au système avec un numéro d'abonné mais elle n'est pas reconnue par le système (mauvais nom, prénom ou numéro d'abonné), le système doit retourner une exception.

S2- Jeanne Dupont cherche à se connecter au système et l'opération se passe avec succès. Elle réalise ensuite une recherche pour connaître l'ensemble des titres de la catégorie Polar. Elle obtient la liste de tous les Polars.

S3- Un abonné identifié réalise une recherche sur la catégorie Voyage. Or il n'y a aucun livre de cette catégorie dans le fonds documentaire. La recherche lui renvoie une liste vide.

S4- Un abonné identifié réserve un ouvrage existant dans le fonds mais indisponible. Son numéro d'abonné et la date de la réservation sont ajoutés à la liste des abonnés ayant réservé cet ouvrage.

S5- Un abonné identifié réserve un ouvrage existant dans le fonds et disponible. Le système lui propose de l'emprunter.

S6- Un abonné identifié réserve un ouvrage n'existant pas dans le fonds. Le système doit retourner une exception.

S7- Un abonné s'identifie, il se voit retourner la liste de ses emprunts en retard.

S8- Un abonné a emprunté un livre le 30 janvier. Il s'identifie à nouveau le 1 mars. Le livre doit figurer dans la liste des emprunts en retard.

S9- Un abonné identifié emprunte un livre. Le stock est mis à jour pour refléter l'emprunt. Le numéro d'abonné et celui de l'ouvrage emprunté sont mémorisés.

S10- Un abonné ayant emprunté un ouvrage, le retourne dans les temps. Le stock est mis à jour.

S11- Un abonné ayant emprunté un ouvrage, le retourne en retard. Le stock est mis à jour. L'abonné se voit notifier le retard.

S12- Un abonné ayant réservé un ouvrage vient l'emprunter. Il est le premier sur la liste des personnes l'ayant réservé. L'emprunt aboutit.

S12- Un abonné ayant réservé un ouvrage vient l'emprunter. Il n'est pas le premier sur la liste des personnes l'ayant réservé. L'emprunt n'aboutit pas et l'abonné est averti de sa position sur la liste d'attente.

### **Instructions :**

Implémenter les tests d'intégration / acceptation portant sur la classe Bibliothèque correspondant au système à construire. La classe de test s'appellera TestBibliothèque. Pour chaque Test, utiliser Mockito pour pallier au défaut d'implémentation correspondantes de l'abonné et de la bibliothèque.

Merci d'indiquer en commentaire dans votre fichier codant les tests le numéro du scénario auquel il correspond. S'il s'agit d'un scénario additionnel que vous souhaitez tester, merci de le décrire en français ou en anglais dans les commentaires du test.

Déposer le projet Eclipse zippé dans le dépôt disponible sous Campus.

Documentation de Mockito : sur le site vogella et sur le site de Mockito lui-même.