Tests d'Intégration – MySurvey

1. Introduction

Les tests d'intégration visent à vérifier l'interaction entre les différentes couches de l'application (modèles, repositories, services et configurations de mapping) dans un contexte réel. Ils valident notamment :

- Le mapping entre les DTOs et les entités (via ModelMapper).
- La persistance et la mise à jour des entités dans la base de données.
- Le respect des relations entre entités (OneToMany, ManyToOne).

Ces tests s'exécutent dans un contexte Spring Boot (via l'annotation `@SpringBootTest`) et utilisent une transaction pour assurer l'isolement des tests.

2. Environnement de Test

- **Framework de test :** Spring Boot Test avec JUnit 5.
- **Configuration :** Les tests sont annotés avec `@Transactional` pour garantir le rollback après chaque exécution.
- **Application :** La classe principale `MySurveyApplication` est utilisée pour initialiser le contexte.
- **Base de données :** Une base de données est utilisée pour simuler les opérations CRUD.

3. Détail des Tests d'Intégration

3.1. Tests de Mapping avec ModelMapper

- **Objectif :** Vérifier que la conversion entre `CommentaireDto` et `Commentaire` se fait correctement dans les deux sens.
- **Exemples :**
    - **DTO → Entité :** Un `CommentaireDto` avec un identifiant de participant est mappé sur une instance de `Commentaire`. Le test vérifie que l'objet résultant est créé et que le champ relatif au participant est correctement initialisé (ou reste vide si la configuration du mapping ne prévoit pas de conversion automatique).
    - **Entité → DTO** : Un `Commentaire` associé à un `Participant` est mappé sur un `CommentaireDto`, et le test vérifie que l'ID du participant est correctement transféré.

3.2. Tests d'Intégration sur la Gestion des Commentaires

- **Objectif :** Valider la création, la mise à jour et la suppression d'un commentaire en base de données.
- **Scénarios :**
    - **Création :** Sauvegarder un commentaire associé à un sondage et un participant, puis vérifier que l'ID est généré et que les associations (sondage et participant) sont correctement établies.
    - **Mise à jour :** Modifier le texte d'un commentaire sauvegardé et vérifier que la modification est persistée.
    - **Suppression :** Supprimer un commentaire et vérifier qu'il n'existe plus en base de données.

3.3. Tests d'Intégration sur DateSondage

- **Objectif :** Vérifier la gestion des dates associées à un sondage.
- **Scénarios :**
    - **Création :** Enregistrer une date de sondage et vérifier que l'ID est généré, que la date est correctement stockée et que le sondage associé est bien lié.
    - **Mise à jour :** Modifier la date d'un enregistrement et vérifier que la nouvelle valeur est persistée.
    - **Suppression :** Supprimer une DateSondage et vérifier son absence dans la base.

3.4. Tests d'Intégration sur DateSondee

- **Objectif :** Tester l'insertion, la mise à jour et la suppression d'une réponse à une date proposée (DateSondee).
- **Scénarios :**
    - **Création et Récupération** : Sauvegarder une DateSondee avec un choix (ex. : DISPONIBLE) et vérifier que les informations (ID, choix, associations avec DateSondage et Participant) sont correctes.
    - **Mise à jour :** Modifier le choix (par exemple, passer de DISPONIBLE à INDISPONIBLE) et vérifier que la modification est effective.
    - **Suppression :** Supprimer une DateSondee et s'assurer de son absence en base de données.

3.5. Tests d'Intégration sur Participant

- **Objectif :** Valider les opérations CRUD sur l'entité Participant.
- **Scénarios :**
    - **Insertion :** Créer et sauvegarder un participant, puis récupérer et vérifier ses attributs.
    - **Mise à jour :** Modifier les informations d'un participant (ex. nom, prénom) et vérifier la persistance de ces modifications.
    - **Suppression :** Supprimer un participant et vérifier qu'il n'est plus présent en base de données.

3.6. Tests d'Intégration sur Sondage et ses Relations

- **Objectif :** Tester la persistance d'un sondage et vérifier ses relations avec d'autres entités (Participant, Commentaire, DateSondage).
- **Scénarios :**
    - **Création et Récupération :** Créer un sondage avec un créateur, vérifier que les informations sont correctement sauvegardées, et récupérer le sondage pour vérifier ses attributs.
    - **Mise à jour :** Modifier les attributs du sondage (nom, description, statut de clôture) et vérifier que les modifications sont prises en compte.
    - **Suppression :** Supprimer un sondage et s'assurer qu'il n'est plus accessible en base.
    - **Relations :** Ajouter des commentaires et des dates de sondage à un sondage, enregistrer ces relations et vérifier lors de la récupération que les listes associées contiennent les éléments attendus.

4. Conclusion

Les tests d'intégration réalisés pour MySurvey confirment que :

- Les mappings entre DTOs et entités sont correctement configurés (via ModelMapper).
- Les opérations CRUD sur les entités (Commentaire, DateSondage, DateSondee, Participant, Sondage) fonctionnent comme prévu.
- Les relations entre entités (par exemple, les associations d'un sondage avec ses commentaires et dates) sont correctement établies et persistées.
- L'ensemble des composants interagit de manière cohérente dans un contexte Spring Boot avec une base de données réelle ou simulée.

Ces tests d'intégrité assurent que l'application respecte les règles métier et la structure de données attendue, garantissant ainsi la fiabilité du système dans un environnement de production.