UserController Le UserController gère les opérations relatives aux utilisateurs dans l'application. Voici un résumé des principales méthodes : createUser: Crée un nouvel utilisateur.

Réponse: 201 OK (utilisateur créé), 400 Mauvaise requête (erreurs de validation), 500 Erreur serveur.

deleteUser : Supprime un utilisateur par son identifiant.

Réponse: 200 OK (utilisateur supprimé), 400 Mauvaise requête (id manquant), 500 Erreur serveur.

getUsers : Récupère tous les utilisateurs.

Réponse : 200 OK (utilisateurs), 500 Erreur serveur.

updateMyData: Met à jour les informations de l'utilisateur actuel.

Réponse: 200 OK (données utilisateur mises à jour), 404 Utilisateur non trouvé, 500 Erreur serveur.

updateUserById: Met à jour les informations d'un utilisateur par son id.

Réponse: 200 OK (données utilisateur mises à jour), 404 Utilisateur non trouvé, 500 Erreur serveur.

getUserById: Récupère un utilisateur par son identifiant.

Réponse: 200 OK (utilisateur), 404 Utilisateur non trouvé, 500 Erreur serveur.

getMyDataById : Récupère les données de l'utilisateur actuel.

Réponse: 200 OK (données utilisateur), 404 Utilisateur non trouvé, 500 Erreur serveur.

login : Permet à un utilisateur de se connecter.

Réponse : 200 OK (réponse d'authentification), 400 Mauvaise requête (erreurs de validation), 500 Erreur serveur.

Points importants : Validation : Chaque méthode qui reçoit des données (comme lors de la création ou de la mise à jour) effectue des validations des données entrantes.

Gestion des erreurs : Les erreurs sont gérées de manière centralisée, avec des réponses de type 400 ou 500 selon le cas.

Interaction avec UsersService : Les données des utilisateurs sont manipulées via le service UsersService, qui gère la logique métier et l'accès aux données (création, mise à jour, suppression, récupération).

Cela permet de maintenir une architecture propre et modulaire, en séparant les responsabilités de gestion des utilisateurs entre le contrôleur et le service.

`UserController`

Le `UserController` gère les opérations relatives aux utilisateurs dans l'application. Voici un résumé des principales méthodes :

1. **createUser**: Crée un nouvel utilisateur.
    - **Réponse**: 201 OK (utilisateur créé), 400 Mauvaise requête (erreurs de validation), 500 Erreur serveur.
2. **deleteUser**: Supprime un utilisateur par son `id`.
    - **Réponse**: 200 OK (utilisateur supprimé), 400 Mauvaise requête (id manquant), 500 Erreur serveur.
3. **getUsers**: Récupère tous les utilisateurs.
    - **Réponse**: 200 OK (utilisateurs), 500 Erreur serveur.
4. **updateMyData**: Met à jour les informations de l'utilisateur actuel.
    - **Réponse**: 200 OK (données utilisateur mises à jour), 404 Utilisateur non trouvé, 500 Erreur serveur.
5. **updateUserById**: Met à jour les informations d'un utilisateur par son `id`.
    - **Réponse**: 200 OK (données utilisateur mises à jour), 404 Utilisateur non trouvé, 500 Erreur serveur.

6. **getUserById**: Récupère un utilisateur par son `id`.
    - **Réponse**: 200 OK (utilisateur), 404 Utilisateur non trouvé, 500 Erreur serveur.
7. **getMyDataById**: Récupère les données de l'utilisateur actuel.
    - **Réponse**: 200 OK (données utilisateur), 404 Utilisateur non trouvé, 500 Erreur serveur.
8. **login**: Permet à un utilisateur de se connecter.
    - **Réponse**: 200 OK (réponse d'authentification), 400 Mauvaise requête (erreurs de validation), 500 Erreur serveur.

Points importants :

- **Validation**: Chaque méthode qui reçoit des données (comme lors de la création ou de la mise à jour) effectue des validations des données entrantes.
- **Gestion des erreurs**: Les erreurs sont gérées de manière centralisée, avec des réponses de type 400 ou 500 selon le cas.
- **Interaction avec `UsersService`**: Les données des utilisateurs sont manipulées via le service `UsersService`, qui gère la logique métier et l'accès aux données (création, mise à jour, suppression, récupération).

Cela permet de maintenir une architecture propre et modulaire, en séparant les responsabilités de gestion des utilisateurs entre le contrôleur et le service.