Effectué par Zaghdane Imed Eddine

Document de spécification

L'API réalisé comporte 3 méthodes :

-book()

-library()

-libraryBooks()

Pour chaque méthode on filtre avec un switch pour traiter tous les types de requetés (GET / POST / PUT / DELETE).

Commençant par la première méthode "**book ()**" :

- **Description fonctionnelle**

Cette méthode s'occupe de traiter les méthodes GET / POST / PUT / DELETE dans le respect des principes de REST sur l'entité book.

En effet, elle est appelée pour les URL ayant les structures suivantes :1

- "/api/books"
- "/api/book/$id"

- **Détails**

GET books : Pour l'affichage de la liste des livres, on utilise le méthode GET avec l'URL ci-dessous.

- GET http://localhost:8080/api/books
- On n'a rien comme données en entrée du service.
- Mais comme données en réponse du service, une liste des livres qui s'affiche avec la format JSON.

GET book : Pour l'affichage d'un livre en utilisant son ID, on utilise le méthode GET avec l'URL ci-dessous (j'ai choisi l'id 1 comme exemple).

- GET http://localhost:8080/api/book/1
- On n'a rien comme données en entrée du service.
- Mais comme données en réponse du service, une livre qui s'affiche avec la format JSON.
- L'exception géré : Si on utilise un ID d'un livre qui n'existe pas, un message d'erreurs s'affiche.

POST book : Pour l'ajout d'un livre, on utilise le méthode POST avec l'URL ci-dessous.

- POST http://localhost:8080/api/books
- On a comme contrainte l'obligation de spécifier la bibliothèque ou va être ajouté dans la partie JSON (données en entrée du service)
- On a comme données en entrée du service un code JSON qui contient les attributs de l'entités livre avec l'ID de la bibliothèque à laquelle il va être ajouté.

Exemple :

```
{
    "library" : {
      "id" : 1
    },
    "name" : "livre1",
    "releaseDate" : "02/01/1999",
    "ISBN" : "ccc12",
    "author" : "authorName"
  }
```

- Et comme données en réponse du service, un message qui affiche "Successfully saved".

L'exception géré :

- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.
- Si on ne spécifie pas l'ID de la bibliothèque, le message "You must include the ID of the library in the JSON body" s'affiche.
- Si l'ajout ne réussit pas un message d'erreur s'affiche.

PUT book : Pour la modification d'un livre, on utilise la méthode PUT avec l'URL ci-dessous.

- PUT http://localhost:8080/api/book/1
- On a comme données en entrée du service un code JSON qui contient les attributs de l'entités livre qu'on veut changer :

Exemple :

```
{
    "library" : {
      "id" : 2
    },
```

"name" : "livre01",

"releaseDate" : "02/01/1999",

"ISBN" : "cc12",

"author" : "authorN"

}

- Et comme données en réponse du service, le livre s'affiche après modification sous format JSON.

L'exception géré :

- Si on ne spécifie l'ID du livre dans l'URL, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'existe pas, un message d'erreurs s'affiche.
- Si on veut changer l'id de la bibliothèque qui contient le livre et on utilise un ID qui n'existe pas, un message d'erreurs s'affiche.
- Si l'update ne réussit pas un message d'erreur s'affiche.

DELETE book : Pour la suppression d'un livre, on utilise le méthode DELETE avec l'URL ci-dessous.

- DELETE http://localhost:8080/api/book/1
- On n'a rien comme données en entrée du service.
- Et comme données en réponse du service, un message qui affiche "Successfully Deleted".

L'exception géré :

- Si on ne spécifie l'ID du livre dans l'URL, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'existe pas, un message d'erreurs s'affiche.

Pour la méthode "**library ()**" :

- Description fonctionnelle

Cette méthode s'occupe de traiter les méthodes GET / POST / PUT / DELETE dans le respect des principes de REST sur l'entité library.

En effet, elle est appelée pour les URL ayant les structures suivantes :

- "/api/libraries"

- "/api/library/$id"

- Détails

GET libraries : Pour l'affichage de la liste des bibliothèques, on utilise la méthode GET avec l'URL ci-dessous.

- GET http://localhost:8080/api/libraries
- On n'a rien comme données en entrée du service.
- Mais comme données en réponse du service, une liste des bibliothèques qui s'affiche avec la format JSON.

GET library : Pour l'affichage d'une bibliothèque en utilisant son ID, on utilise le méthode GET avec l'URL ci-dessous (j'ai choisi l'id 1 comme exemple).

- GET http://localhost:8080/api/library/1
- On n'a rien comme données en entrée du service.
- Mais comme données en réponse du service, une bibliothèque qui s'affiche avec la format JSON.

L'exception géré :

- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.

POST library : Pour l'ajout d'une bibliothèque, on utilise le méthode POST avec l'URL ci-dessous.

- POST http://localhost:8080/api/libraries
- On a comme données en entrée du service un code JSON qui contient les attributs de l'entité Library.

Exemple :

```
{
    "name" : "libParis",
    "address" : "Paris",
    "yearCreated" : 2015
}
```

- Et comme données en réponse du service, un message qui affiche "Library saved".

L'exception géré :

- Si l'ajout ne réussit pas un message d'erreur "Library not saved (Bad Request)" s'affiche.

PUT library : Pour la modification d'une bibliothèque, on utilise la méthode PUT avec l'URL ci-dessous.

- PUT http://localhost:8080/api/library/1
- On a comme données en entrée du service un code JSON qui contient les attributs de l'entités library qu'on veut changer.

Exemple :

```
{
    "name" : "libraryParis",
    "address" : "Paris",
    "yearCreated" : 2015
}
```

- Et comme données en réponse du service, Affichage de la bibliothèque sous format JSON après UPDATE.

```
{
  "id" : 3,
  "address" : "Paris",
  "books" : [],
  "name" : "libraryParis",
  "yearCreated" : 2015
}
```

L'exception géré :

- Si on ne spécifie l'ID de la bibliothèque dans l'URL, un message d'erreurs s'affiche.
- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.
- Si l'update ne réussit pas un message d'erreur s'affiche.

DELETE library : Pour la suppression d'une bibliothèque, on utilise le méthode DELETE avec l'URL ci-dessous.

- DELETE http://localhost:8080/api/library/1
- On n'a rien comme données en entrée du service.
- Et comme données en réponse du service, un message qui affiche "Successfully Deleted".

L'exception géré :

- Si on ne spécifie l'ID de la bibliothèque dans l'URL, un message d'erreurs s'affiche.
- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.

Pour la méthode "**libraryBooks ()**" :

- Description fonctionnelle

Cette méthode s'occupe de traiter les méthodes GET / POST / PUT / DELETE dans le respect des principes de REST pour assurer la gestion des livres d'une bibliothèque.

En effet, elle est appelée pour les URL ayant les structures suivantes :

- "/api/library/$id/books"

- "/api/library/$id/book/$idB"

- Détails

GET library's books : Pour l'affichage de la liste des livres d'une bibliothèque choisi, on utilise la méthode GET avec l'URL ci-dessous.

- GET http://localhost:8080/api/library/1/books
- On n'a rien comme données en entrée du service.
- Mais comme données en réponse du service, une liste des livres de la bibliothèque choisi qui s'affiche avec la format JSON.

L'exception géré :

- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.

GET book from library : Pour l'affichage d'un livre d'une bibliothèque choisi, on utilise la méthode GET avec l'URL ci-dessous.

- GET http://localhost:8080/api/library/1/book/1
- On n'a rien comme données en entrée du service.
- Mais comme données en réponse du service, le livre avec l'ID choisi à l'URL de la bibliothèque choisi s'affiche avec la format JSON.

L'exception géré :

- Si on utilise un ID d'un livre qui n'existe pas dans la bibliothèque choisie, un message d'erreurs s'affiche.
- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'existe pas, un message d'erreurs s'affiche.

POST book on library : Pour l'ajout d'un livre à une bibliothèque choisi, on utilise la méthode POST avec l'URL ci-dessous.

- POST http://localhost:8080/api/library/1/books
- On a comme données en entrée du service un code JSON qui contient les attributs de l'entité Book.

Exemple :

```
{
    "name" : "livreName",
    "releaseDate" : "02/01/1999",
    "ISBN" : "aaa12",
    "author" : "author"
}
```

- Et comme données en réponse du service, un message qui affiche "Library saved".

L'exception géré :

- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.
- Si l'ajout ne réussit pas un message d'erreur "Library not saved (Bad Request)" s'affiche.

PUT library's book : Pour la modification d'un livre d'une bibliothèque choisi, on utilise la méthode PUT avec l'URL ci-dessous.

- PUT http://localhost:8080/api/library/1/book/1
- On a comme données en entrée du service un code JSON qui contient les attributs de l'entités livre qu'on veut changer.

Exemple :

```
{
    "name" : "livre01",
    "releaseDate" : "02/01/1999",
    "ISBN" : "cc12",
    "author" : "authorN"
}
```

- Et comme données en réponse du service, un message qui affiche "Successfully updated".

L'exception géré :

- Si on ne spécifie l'ID de la bibliothèque dans l'URL, un message d'erreurs s'affiche.
- Si on ne spécifie l'ID du livre dans l'URL, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'existe pas, un message d'erreurs s'affiche.
- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'appartient pas à la bibliothèque choisie, un message d'erreurs s'affiche.
- Si l'update ne réussit pas un message d'erreur s'affiche.

DELETE book from library : Pour la suppression d'un livre d'une bibliothèque choisi, on utilise la méthode DELETE avec l'URL ci-dessous.

- DELETE http://localhost:8080/api/library/1/book/1
- On n'a rien comme données en entrée du service.
- Et comme données en réponse du service, un message qui affiche "Successfully Deleted".

L'exception géré :

- Si on ne spécifie l'ID de la bibliothèque dans l'URL, un message d'erreurs s'affiche.
- Si on ne spécifie l'ID du livre dans l'URL, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'existe pas, un message d'erreurs s'affiche.
- Si on utilise un ID d'une bibliothèque qui n'existe pas, un message d'erreurs s'affiche.
- Si on utilise un ID d'un livre qui n'appartient pas à la bibliothèque choisie, un message d'erreurs s'affiche.

Remarque : Le document procédure de test contient tous les tests effectués par PostMan et qui montre avec détails les résultats et les cas d'erreurs gérés.