

Projet Technologies des services du Web

Application pour la gestion des
utilisateurs des bibliothèques d'une
ville

Issam FADLAOUI





Plan :




Ce document sert de compte rendu du projet des technologies des services du Web, il contient les explications des éléments ci-dessous avec des captures d'écran du code sous Eclipse et de l'application en local:

- L'ensemble des classes et pages web utilisées.
- Web services utilisés.
- Persistance.
- Front.
- Exemples et tests sur l'application.






Classes et pages utilisées:



Bibliothèque:

- >  Bibliotheque.java : entité bibliothèque
- >  BibliothequeInput.java :Bibliothèque Input (avec nomBiblio)
- >  BibliothequeRepository.java :Services CRUD
- >  BibliothequeResource.java :Ressource Bibliothèque





-  rechbiblio.html : page html pour la recherche d'une bibliothèque
-  servicesBib.html :page html pour accéder aux services possibles sur les bibliothèques
-  biblios.html :page html pour ajouter une nouvelle bibliothèque




Livre:

- >  Auteur.java :Auteur de Livre
- >  Livre.java :entité Livre
- >  LivreInput.java :Livre Input(Pojo avec idLivre)
- >  LivreRepository.java :Services Crud
- >  LivreResource.java :Ressource Livre

-  livres.html :page d'ajout d'un livre
-  servicesLiv.htm :service de gestion des livres
-  rechlivre.html :page de recherche d'un livre

Personne:

- >  Personne.java :entité personne
- >  PersonneInput.java :pojo avec idPersonne
- >  PersonneRepository.java :services CRUD
- >  PersonneResource.java :Ressource Personne

-  emprunt.html :emprunt d'un livre
-  rechPers.html :page de recherche d'une personne
-  servicesUtil.html :service de gestion des utilisateurs des bibliothèques

Autres fichiers utilisés:

 accueil.html : page html d'accueil de l'application

 pom.xml : gestion des dépendances maven

 control.js : fichier JavaScript pour le FRONT

 CSS1.css : feuille de style CSS

 application.properties : propriétés application (numéro de port, url ...)

APPLICATION:

>  DemoApplication.java : Application Spring Boot

>  JerseyConfiguration.java : Classe d'appel aux services de JERSEY

Web Services:

consommés depuis control.js

Mode de communication utilisé: JSON

BIBLIOTHEQUES:

- Ajout d'une bibliothèque avec son adresse à la base de données (POST)
- Consulter la liste des bibliothèques contenues dans la base de données (GET)
- Rechercher une bibliothèque selon son nom ou son adresse. (Filtrage de la liste des Bibliothèques dans JS)
- Supprimer une bibliothèque de la base de données (DELETE)

LIVRE:

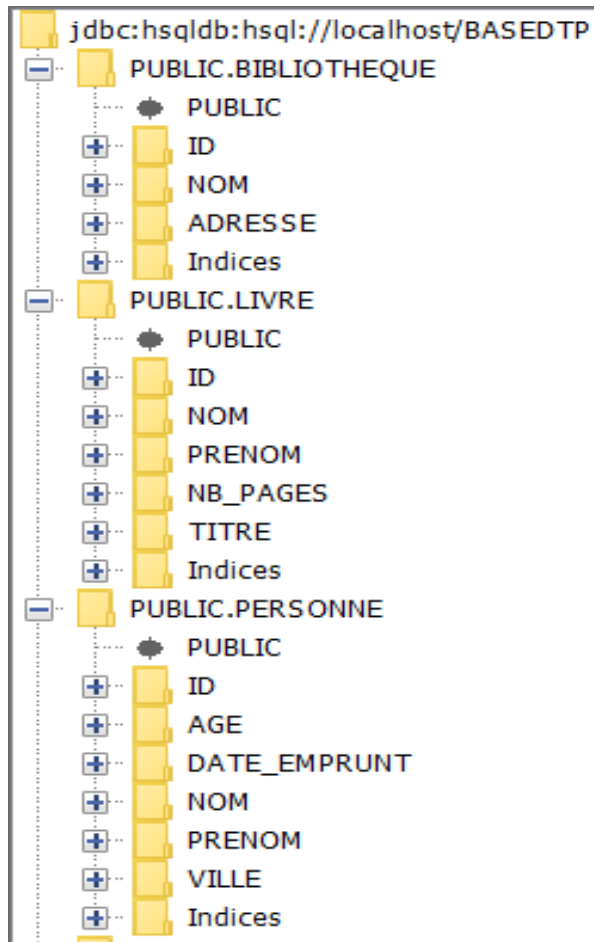
- Ajout d'un livre dans la base de données .(POST)
- Consulter la liste des livres contenus dans la base de données.(GET)
- Rechercher un livre selon son titre, le nom ou le prénom de son auteur.(Filtrage de la liste des Livres dans JS)
- Supprimer un livre de la base de données.(DELETE)

PERSONNE:

- Ajout d'un utilisateur à la base de données avec le livre qu'il a pris et la bibliothèque dans laquelle il l'a pris.(POST)
- Consulter la liste des utilisateurs des bibliothèques de la ville.(GET)
- Rechercher un utilisateur selon son nom, prénom, ses livres ou bibliothèques .(Filtrage de la liste des Livres ou celles des bibliothèques d'un certain utilisateur dans JS)
- Supprimer une personne de la base de données.(DELETE)

Persistence:

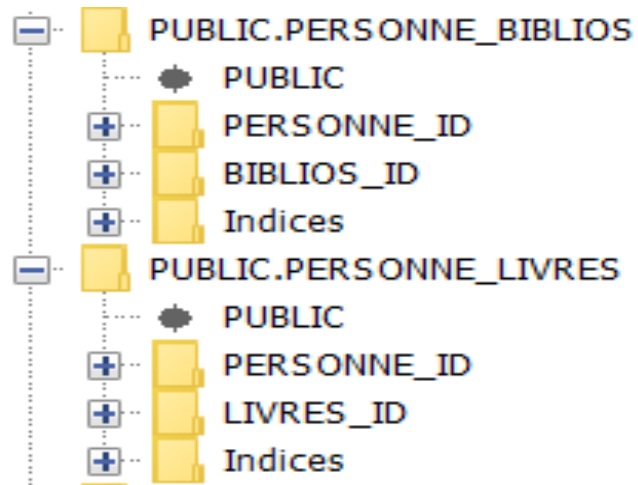
Schéma de la base de données (hsqldb) :



RELATIONS:

Personne: (ManyToMany) Livre ; une personne peut posséder plusieurs livres

Personne: (ManyToMany) Bibliothèque ; une personne peut prendre des livres de plusieurs bibliothèques.



Front:

control.js

```
let $listeLivres = $("#listLivres");
let $listePersonnes = $("#listPersonnes");
let $selectLivres = $("#selectLivres");
let $listeBiblios = $("#listBiblios");
let $InputLivre = $("#filt-input");
let $RechbtnLiv = $("#RechbtnLivre");
let $selectBiblio = $("#selectBib");
let listLiv = [];
let $RechBib = $("#bib-input1");
let $Rechbtn = $("#RechbtnBiblio1");
let listBib = [];
let listPers = [];
let $RechbtnPers = $("#RechbtnPers1");
let $InputPers = $("#pers-input1");
```

```
/*UL des Livres contenus en BD*/
/*UL des Personnes contenus en BD*/
/*UL des Livres à sélectionner dans l'ajout de personne*/
/*UL des Bibliothèques contenus BD */
/*input de recherche de livre*/
/*bouton recherche livre*/
/*UL des Bibliothèques à sélectionner dans l'ajout de personne*/
/*Liste intermédiaire pour le filtrage des livres*/
/* input de recherche Bibliothèque*/
/* bouton recherché Bibliothèque*/
/*Liste intermédiaire pour le filtrage des Bibliothèques*/
/*Liste intermédiaire pour le filtrage des Personnes*/
/* bouton recherche personne*/
/*input de recherche de personne*/
```

Méthodes utilisées:

GET des données de la BD :

```
$.get("http://localhost:8080/api/biblio",function(resp))
```

```
$.get("http://localhost:8080/api/personnes",function(resp))
```

```
$.get("http://localhost:8080/api/livres",function(resp))
```

Ajout des données:

\$('#addbtnPersonne').click: Au clique sur le bouton d'ajout d'une personne, ajax envoie les données en input au serveur avec une requête POST pour les enregistrer en base de données

\$('#addbtnLivre').click: Au clique sur le bouton d'ajout d'un livre, ajax envoie les données en input au serveur avec une requete POST pour les enregistrer en base de données

\$('#addbtnBiblio').click: Au clique sur le bouton d'ajout d'une bibliothèque, ajax envoie les données en input au serveur avec une requête POST pour les enregistrer en base de données

Utilisant les fonctions suivantes:

appendToListLivre(livre): Ajoute un élément li dans la liste de livres

appendToListBiblio(biblio): Ajoute un élément li dans la liste des bibliothèques

appendToSelectsBib(biblio): Ajoute un élément li dans la liste des bibliothèques à sélectionner

appendToSelects(livre): Ajoute un élément li dans le select des livres

appendToListPersonne(personne): Ajout e un élément li dans la liste de personne

Recherche des données:

\$Rechbtn.click: Au clique sur le bouton de recherche d'une bibliothèque, on remplace l'affichage de la liste des bibliothèques par la liste des bibliothèques contenant l'input de recherche dans leurs noms.

\$RechbtnLiv.click: Au clique sur le bouton de recherche d'un Livre, on remplace l'affichage de la liste des livres par la liste des livres contenant l'input de recherche dans leurs titres, noms ou prénoms d'auteurs.

\$RechbtnPers.click: Au clique sur le bouton de recherche d'une Personne, on remplace l'affichage de la liste des personnes par la liste des personnes contenant l'input de recherche dans leurs noms, prénoms, bibliothèque.

Utilisant les fonctions suivantes:

filtresLiv(searchFilter):Fonction qui filtre la liste des livres

filtresBib(searchFilter):Fonction qui filtre la liste des bibliothèques

filtresPers(searchFilter):Fonction qui filtre la liste des personnes

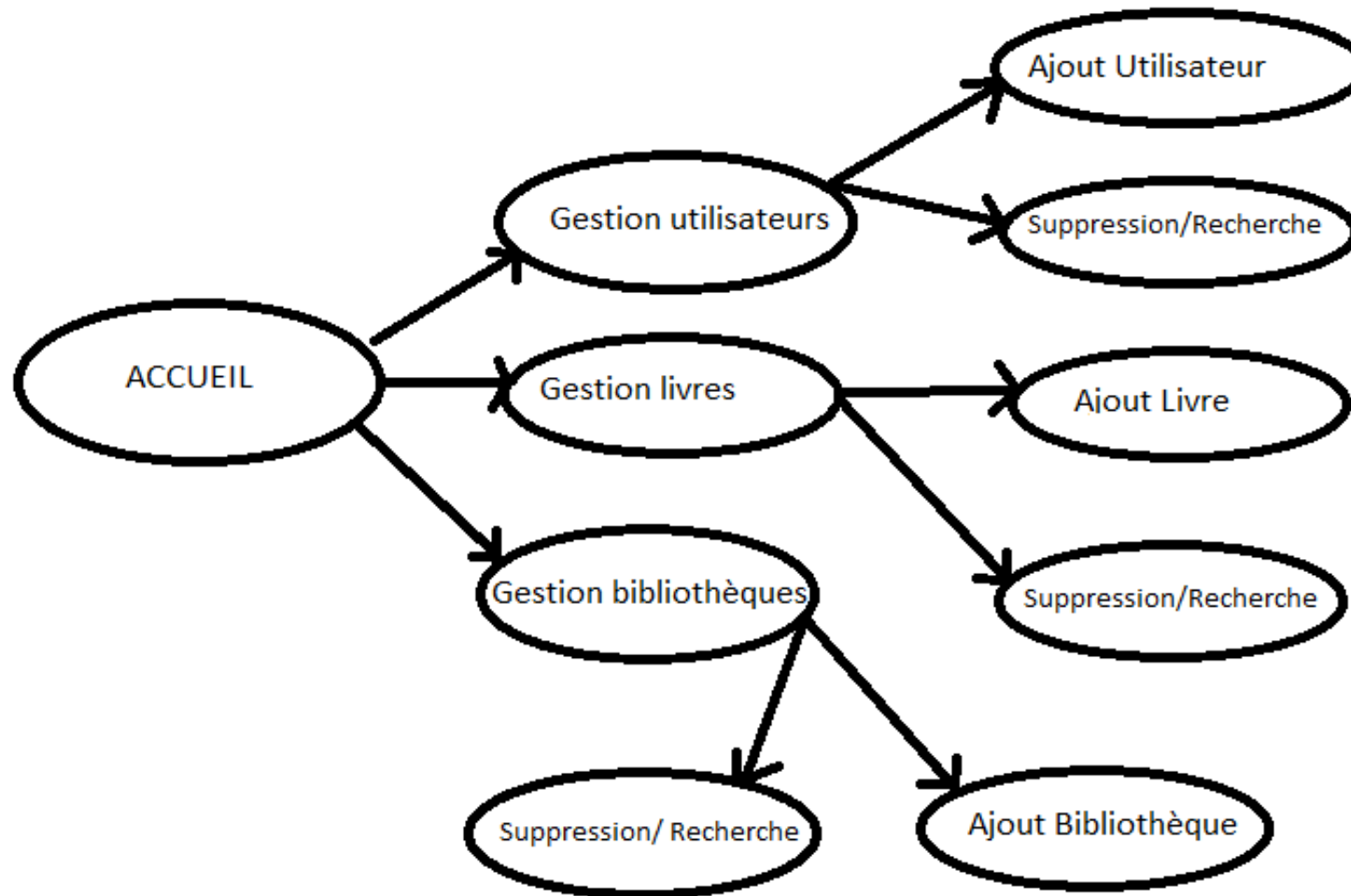
Suppression des données:

\$listePersonnes.on("click", "li button"):Au clique sur la croix devant l'affichage d'une personne, on supprime celle-ci de la base de données

\$listeLivres.on("click", "li button"):Au clique sur la croix devant l'affichage d'un livre, on supprime celui-ci de la base de données

\$listeBiblios.on("click", "li button"):Au clique sur la croix devant l'affichage d'une bibliothèque, on supprime celle-ci de la base de données

Fonctionnement de l'Application:



Démonstration:

accueil.html

Bienvenue dans le site des Bibliothèques de votre ville !



Gestion des utilisateurs des bibliothèques de la ville:

Si vous souhaitez consulter, ajouter, ou supprimer une personne: [Cliquez-ici](#).

Gestion des livres disponibles dans les bibliothèques de la ville:

Si vous souhaitez consulter, ajouter, ou supprimer un livre: [Cliquez-ici](#).

Gestion des bibliothèques de la ville:

Si vous souhaitez consulter, ajouter, ou supprimer une bibliothèque: [Cliquez-ici](#).

BIBLIOTHEQUES:

Gestion des bibliothèques de la ville:



Pour ajouter de nouvelles bibliothèques: [Cliquez-ici.](#)

Pour rechercher ou supprimer une bibliothèque: [Cliquez-ici.](#)

Ajout des bibliothèques :

Ajoutez une nouvelle bibliothèque :

Nom :

Adresse :

Bibliothèques de la ville:

Recherchez votre bibliotheque selon son nom:

Bibliothèque
Universitaire des
Tertiales-Caserne
Ronzier,
Boulevard Henri
Harpignies, 59300
Valenciennes

LIVRES:

Gestion des livres disponibles dans les bibliothèques de la ville:



Pour ajouter un nouveau livre: [Cliquez-ici.](#)

Pour rechercher ou supprimer un livre: [Cliquez-ici.](#)

LIVRES :

Ajout d'un nouveau livre :

Titre :

Nb pages :

Nom auteur :

Prenom auteur :

Add

Livres disponibles dans les bibliothèques de la ville:

Recherchez votre livre selon son titre, le nom ou le prénom de son auteur:

Auteur2

Recherche

Livre2 -
NomAuteur2
PrenomAuteur2

X

Utilisateurs:

servicesUtil.html

Gestion des utilisateurs des bibliothèques de la ville:



Pour ajouter une nouvelle personne: [Cliquez-ici.](#)

Pour rechercher ou supprimer une personne: [Cliquez-ici.](#)

Personnes ayant emprunté récemment des livres de la bibliothèque :

Ajout d'une nouvelle personne :

Prenom :


Nom :

Age :

Livre :

Ville :

Bibliothèque :

Date d'emprunt: 

Add

La liste des livres disponibles apparaissent dans la sélection du Livre. Ainsi que les bibliothèques enregistrées dans le réseau apparaissent dans la sélection de la bibliothèque.

control.js:180

```
{id: 10, prenom: "PrenomUtilisateur3", nom: "NomUtilisateur3", age: 17, date_
emprunt: "2021-03-29", ...}
  age: 17
  biblios: Array(1)
    0: {id: 3, nom: "Médiathèque Simone Veil", adresse: "4 Rue Ferrand, 59300...
        length: 1
        __proto__: Array(0)
      date_emprunt: "2021-03-29"
      id: 10
    }
  livres: Array(1)
    0: {id: 7, nbPages: 300, titre: "De la démocratie en Amérique", auteur: {...
        length: 1
        __proto__: Array(0)
      nom: "NomUtilisateur3"
      prenom: "PrenomUtilisateur3"
      ville: "Valenciennes"
    }
    __proto__: Object
```



jdbc:hsqldb:hsq://localhost/PROJTW						
+	PUBLIC.BIBLIOTHEQUE	SELECT * FROM "PUBLIC"."PERSONNE"				
+	PUBLIC.LIVRE					
+	PUBLIC.PERSONNE					
+	PUBLIC.PERSONNE_BIBLIOS					
+	PUBLIC.PERSONNE_LIVRES					
+	Properties					
ID	AGE	DATE_EMPRUNT	NOM	PRENOM	VILLE	
8	20	2021-04-01	FADLAOUI	ISSAM	Valenciennes	
9	15	2021-03-31	NomUtilisateur2	PrenomUtilisateur2	Aulnoy-Lez-Valenciennes	
10	17	2021-03-29	NomUtilisateur3	PrenomUtilisateur3	Valenciennes	

JOINTURE:

jdbc:hsqldb:hsq://localhost/PROJTW

- PUBLIC.BIBLIOTHEQUE
- PUBLIC.LIVRE
- PUBLIC.PERSONNE
- PUBLIC.PERSONNE_BIBLIOS
- PUBLIC.PERSONNE_LIVRES
- Properties

SELECT * FROM "PUBLIC"."PERSONNE_LIVRES"

PERSONNE_ID	LIVRES_ID
8	6
9	5
10	7

jdbc:hsqldb:hsq://localhost/PROJTW

- PUBLIC.BIBLIOTHEQUE
- PUBLIC.LIVRE
- PUBLIC.PERSONNE
- PUBLIC.PERSONNE_BIBLIOS
- PUBLIC.PERSONNE_LIVRES
- Properties

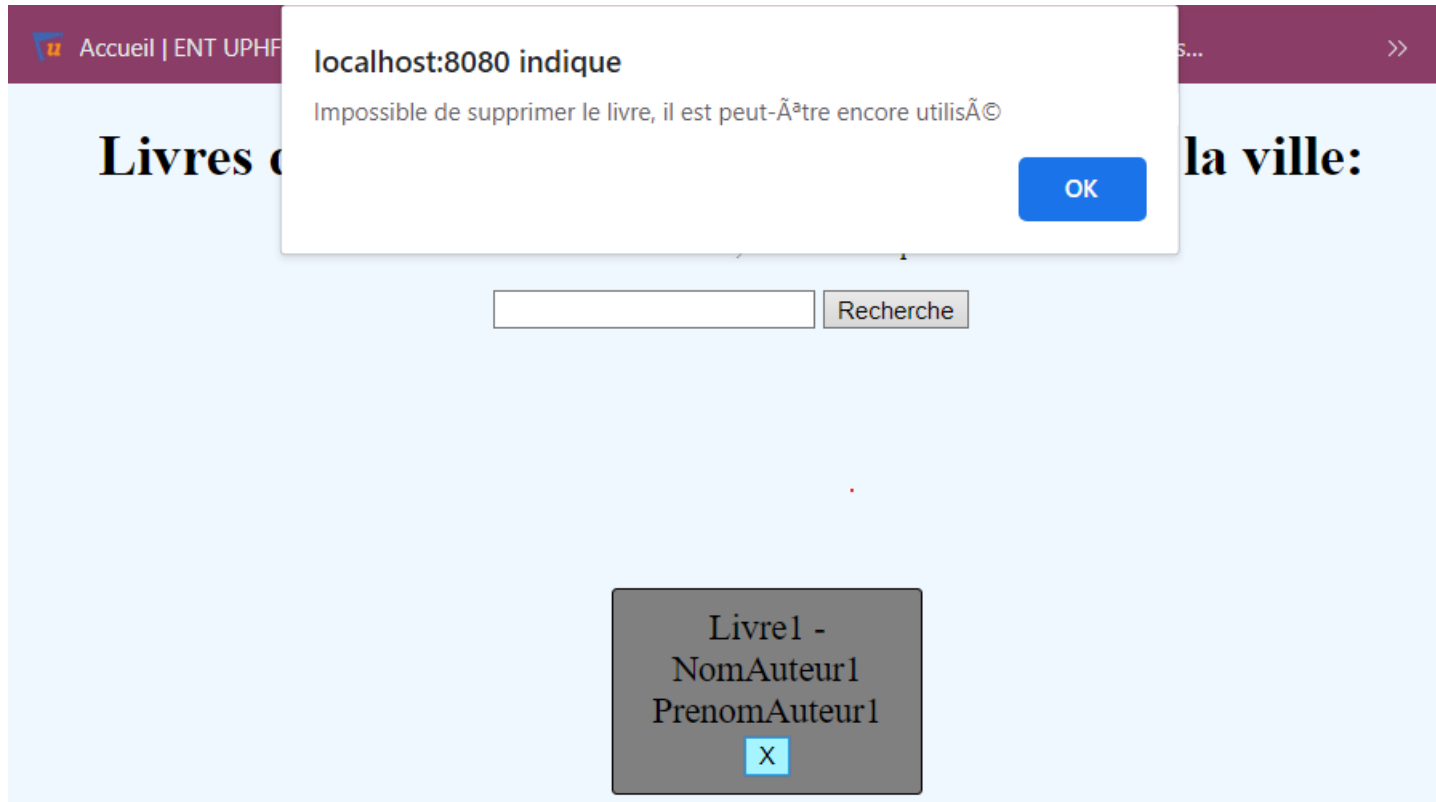
SELECT * FROM "PUBLIC"."PERSONNE_BIBLIOS"

PERSONNE_ID	BIBLIOS_ID
8	2
9	1
10	3

Exceptions:



On ne peut pas supprimer une bibliothèque alors qu'elle a encore des utilisateurs.



On ne peut pas supprimer un livre alors qu'il est déjà pris par un utilisateur.