Documents techniques du site de la médiathèque de La Chapelle Curreaux

Spécifications techniques :

Serveur :

- WAMP (local)
- Heroku (En ligne)
- JawsDB (BDD en ligne)
- Version PHP (7.4.9)
- Extension PHP : PDO
- MySQL (5.7.31)
- Apache (2.4.46)

Front-end :

- HTML5
- CSS3
- JAVASCRIPT ES6
- AJAX
- BOOTSTRAP 4

Back-end :

- PHP 7
- PDO
- MYSQL

Diagramme cas d'utilisations :

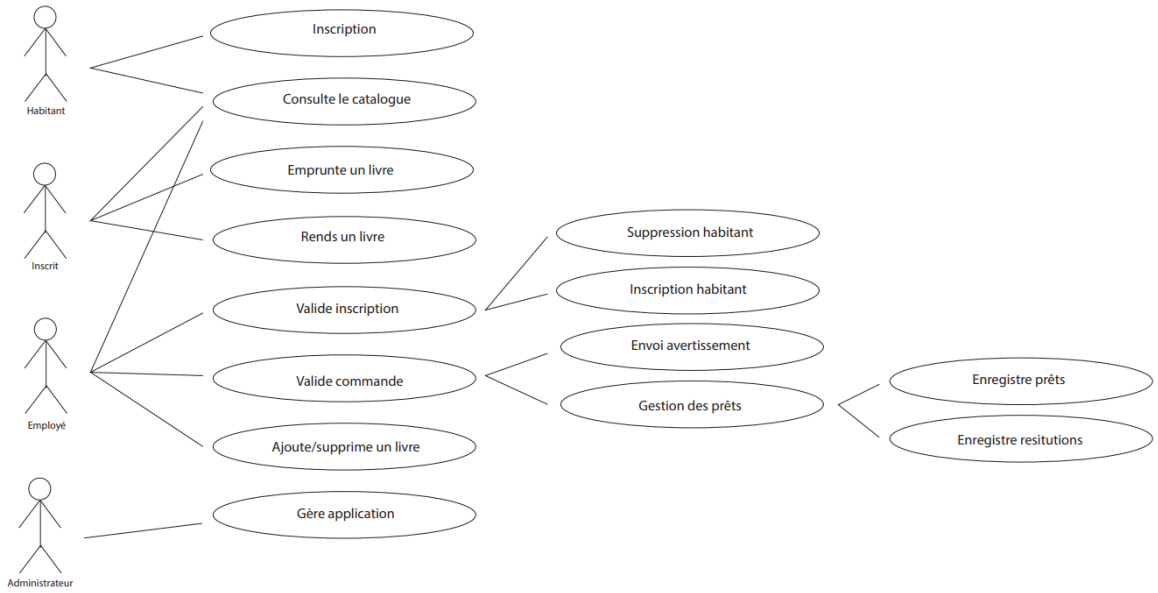
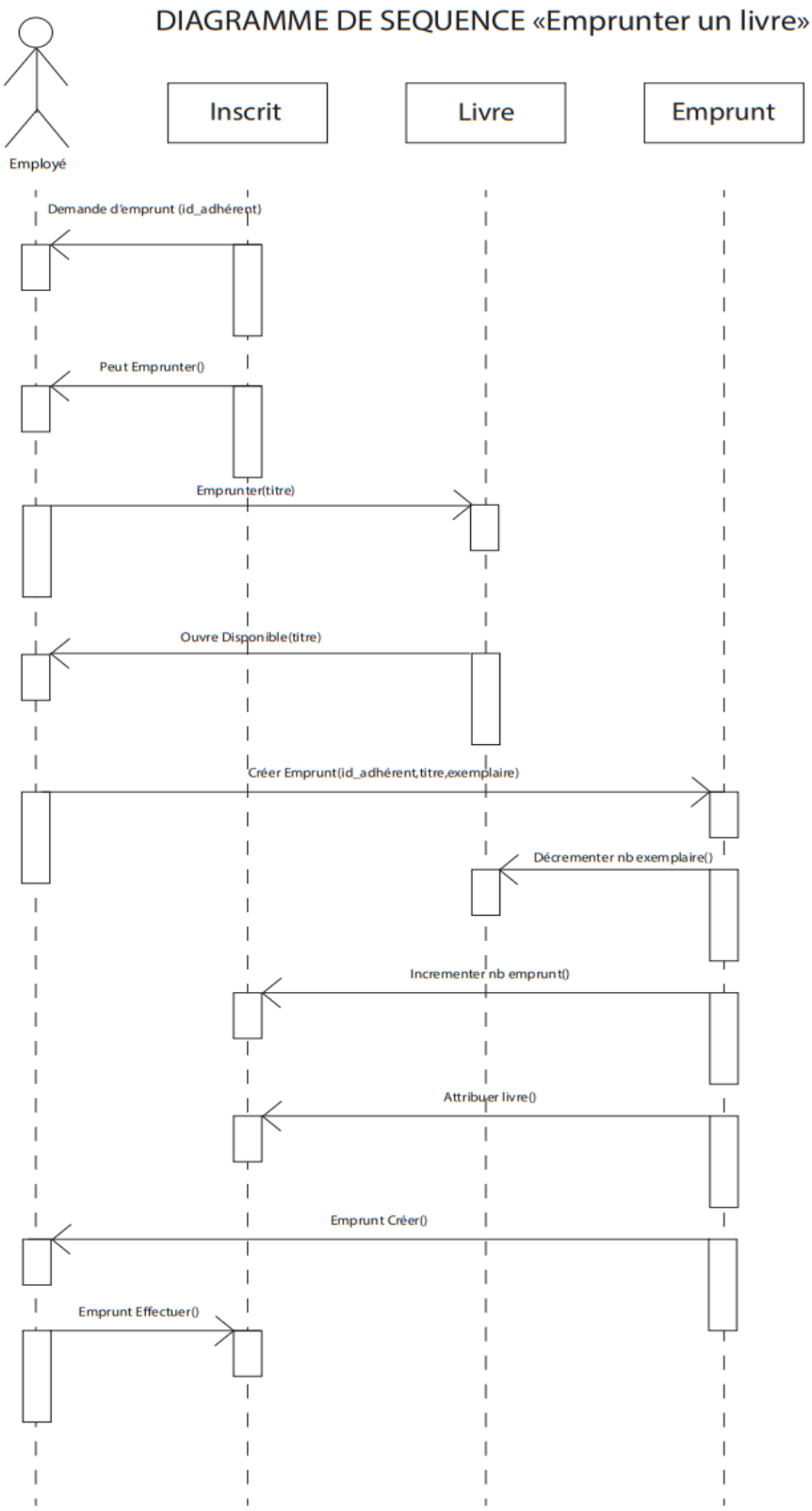


Diagramme de séquence :

« Emprunter un livre »



« Rendre un livre »

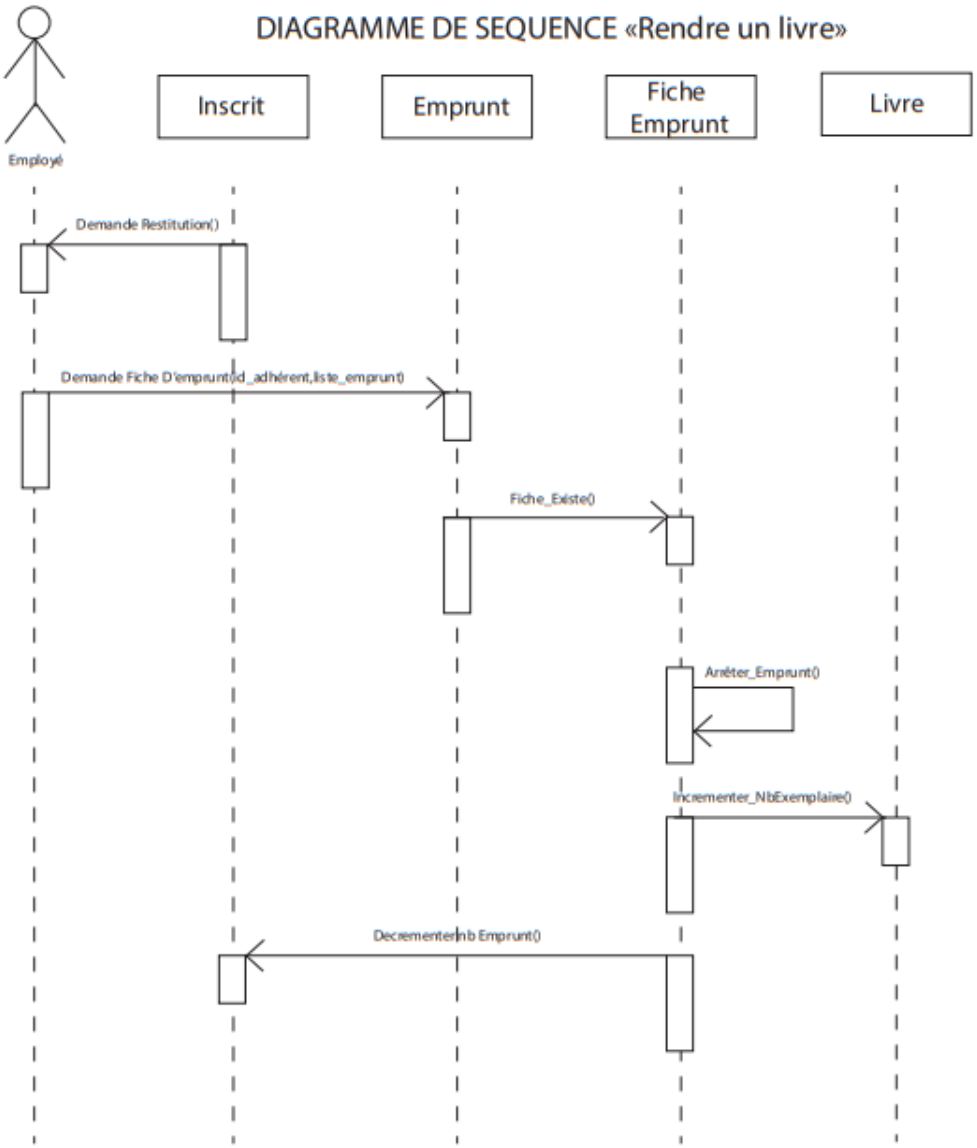
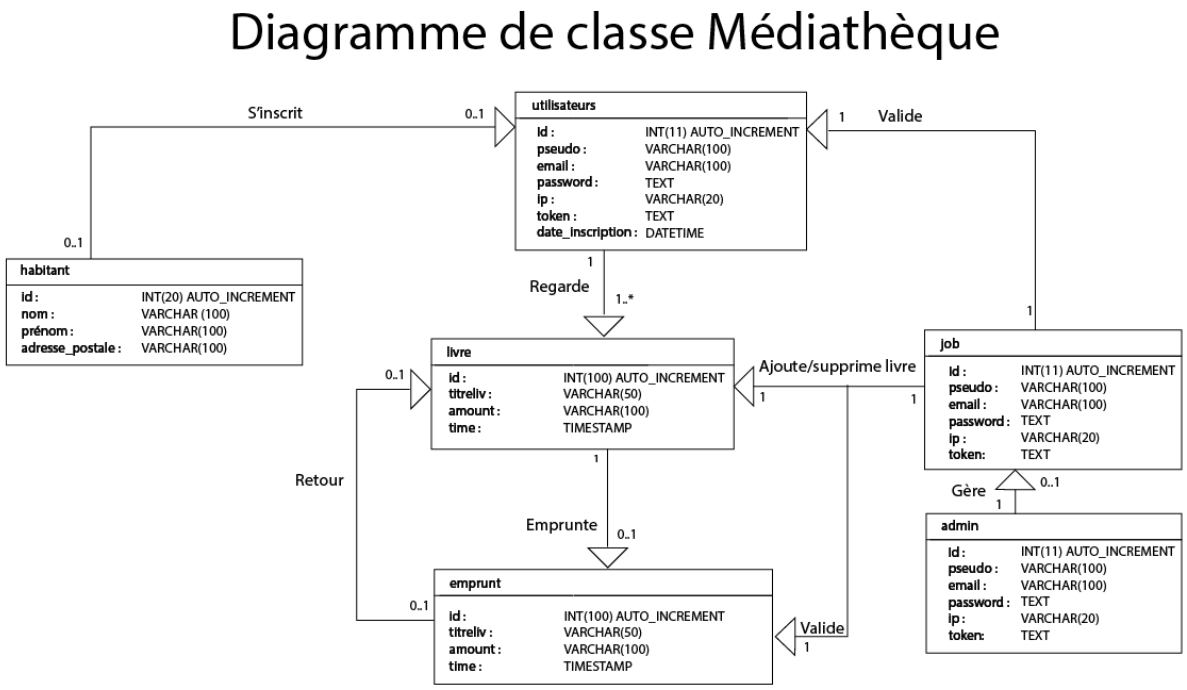


Diagramme de classe :



Mesures de sécurité :

J'ai pris les mesures de sécurité suivantes afin d'assurer le maximum de protection :

- **Utiliser des requêtes préparées :**

Cela me permet de protéger l'application des injections SQL et en même temps de gagner en performance dans le cas d'une requête exécutée plusieurs fois par la même session.

- **Crypté les mots de passe avec BCRYPT :**

Le hachage de mot de passe est l'une des pratiques de sécurité les plus basiques qui doit être effectuée. Sans cela, chaque mot de passe stocké peut être volé si le support de stockage (typiquement une base de données) est compromis.

BCRYPT utilise l'algorithme CRYPT_BLOWFISH pour créer la clé de hachage. Ceci va créer une clé de hachage standard crypt() utilisant l'identifiant "$2y$". Le résultat sera toujours une chaîne de 60 caractères, ou false si une erreur survient.

- **Utiliser La fonction htmlspecialchars :**

Elle identifie toute sortie que vous ne voulez pas comme une sortie HTML et la convertit en entités HTML, par exemple: '&' devient '&' et '"' (double guillemet) devient '' '';

- Supprimer la base de données d’exemple :

Elle contient les identifiants des comptes d’accès et les droits associés.

- J’ai veillé à appliquer les bonnes relations et contraintes entre mes tables :

Afin permettre à un langage de requête de haut niveau comme SQL d'éviter une navigation complexe dans la base de données.