République **A**lgérienne **D**émocratique et **P**opulaire
Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU **G**ENIE E ET D'INFORMATIQUE
ELECTRIQU
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes
Master Académique

Domaine : **Mathématiques et Informatique**
Filière : **Informatique**
Spécialité : **Conduite de projet informatique**

Thème

Conception et réalisation d'une application web jee et mobile pour la gestion de la bibliothèque.

Mémoire soutenu publiquement le 03/07/2017 devant le jury composé de :

Président : Mr Habet.
Examinateur : Mlle Yesli Y.
Examinateur : Mlle Bougchiche L.
Encadreur : Mr. Chebouba L.

Présenté par :

BADJI SABIHA

ZAMOUM NASSIM

2017/2018

Remerciements

Nous tenons à témoigner notre reconnaissance à DIEU tout puissant, qui nous a aidé et bénit par sa volonté durant toute cette période.

Notre profonde gratitude et sincères remerciements vont à notre promoteur ***Mr l.cheboubа****pour sa présence continuelle, son encouragement, son humour et sa patience tout au long de ce travail.*

Nous adressons nos remerciements aux membres du jury, devant qui, nous avons l'honneur d'exposer notre travail, et qui ont pris la peine de lire ce mémoire pour juger son contenu.

Nous réservons ici une place particulière pour remercier vivement tous ceux qui, d'une manière ou d'une autre, nous ont aidés et encouragés à la réalisation de ce modeste travail.

Dédicaces

Je dédie ce travail :

*A mes très chères parents (Ma mère Saliha et Mon Papa Mouloud) qui se sacrifient pour moi, que dieu les protège.

*A Ma Sœur Katia

* A mon frère Anis.

* A tous mes amis(es).

* A tous ceux qui m'aiment et que j'aime.

NASSIM

Dédicaces

Je dédie ce modeste travail à :

À mes chers parents, à qui je dois tout.

A mes frères.

À mon frère Brahim et sa femme Sabrina ainsi leur fille Lina.

Ma sœur (Ghania) et son mari (Hamid).

À tous mes amis.

A Mon binôme nassim.

À tous les étudiants de l'Informatique.

À toute la famille universitaire.

SABIHA

SOMMAIRE :

Chapitre 3: réalisation et implémentation.

Les figures :

Chapitre I : présentation de l'architecture JavaEE.

Chapitre II : Analyse et conception.

Chapitre III : réalisation et implémentation.

Introduction générale :

Les systèmes d'information sont devenus indispensables au fonctionnement des entreprises car ils ont été utilisés comme un élément permet d'améliorer leur productivité, ce qui donne à l'informatique une place importante au sein de tous les demains.

La technologie de traitement de l'information offre une grande variété d'outils permettant d'améliorer les services.

Apres l'analyse de la méthode de travail de la bibliothèque et en particulier le service de prêt et de réservation des livres nous avons relevé un ensemble de points que l'on pourrait améliorer. Ces derniers font l'objet de notre problématique qui peut être formuler comme suit : quelles sont les propositions envisageables pour faciliter la maintenance de l'application de gestion des livres ? Ajoutons a cela la réservation des livres ainsi que la gestion des étudiants par un administrateur.

Notre travail consiste a répondre a cette problématique a travers les trois chapitres qui constituent.

Organisation du mémoire :

Dans le premier chapitre de ce mémoire, nous aborderons les points de la technologie JavaEE qui repose sur le principe de l'oriente objet qui facilite la maintenance et l'extension de l'application.

Dans le deuxième chapitre, nous le consacrerons a l'analyse et la conception de notre application.

Dans le quatrième chapitre donnera une vue globale des solutions techniques qui vont nous permettre de réaliser notre application.

Et pour terminer, une conclusion synthétise notre travail.

Chapitre1

Introduction :

Dans ce chapitre nous allons aborder les bases essentielles a la compréhension de notre application .le standard JEE fait partie des technologies dites révolutionnaires de notre époques précisément en développement applications de grande envergure. Elle présente plusieurs solutions et elle est en perpétuelle évolution.

Nous n'allons pas aborder le standard JEE avec l'ensemble de ses composants, car il est parfois lourd d'utiliser un composant conçu pour de grande architecture alors que notre application est restreinte. C'est pour cela que nous avons choisi d'aborder une architecture JEE avec l'intégration des framework.

L'ensemble de la communauté open source s'est occupée de lancer sur le marché des framework servant à simplifier l'utilisation de telle ou telle technologie. Les framework sont plus simple d'utilisation et plus performants dans certain cas.

Architecture client/serveur :

Définition :

On peut définir l'architecture client/serveur comme étant une méthodologie informatique de communication entre plusieurs machines (clients) et une machine serveur. Ce dernier est installé quelque part dans le monde il s'occupe de l'installation des services, le partage des périphériques, de mémoriser des données sur son disque dur, et les distribuées suite a demande du client.

Quelque notion de base :

- **Client :** Est un processus qui est connecter à un serveur, le client envoie une requête et reçoit une réponse immédiatement.
- **Serveur :** Est un ordinateur qui accueille des requetés les traitent et renvoie des réponses sous forme de page web aux clients il est toujours allumée et fonctionne 24h/24h.
- **Requête :** Est un message envoyé du client a un serveur comportant la tache à exécuter.
- **Réponse :** Est un message émis par un serveur a un client comportent les éléments du résultats des taches exécuter.

Principe de l'architecture client/serveur :

Le client envoie une requête vers le serveur à l'aide de son adresse IP et le port qui désigne un service particulier du serveur. Quand le serveur reçoit la Demande, une réponse est envoyé directement à l'aide de la machine client et son port.

Avantages de l'architecture client /serveur :

Le modèle client/serveur est particulièrement recommandé pour des réseaux nécessitent un grand niveau de fiabilité ses principe atouts sont :

- **Des ressources centralisées** :

Étant donné que le serveur et au centre du réseau peut gérer des ressources communes à tous les utilisateurs comme par exemple une base de données centralises afin d'éviter des problèmes de redondance et de coordination.

- **Une meilleure sécurité :**

Car le nombre de point d'entrée permanentent l'accès au donne est moins important.

- **Une administration au niveau serveur :**

Les clients ayant peu d'importance dans ce modèle ils ont besoin d'être administrés.

- **Un réseau évolutif :**

Grâce à cette il est possible de supprimer ou rajouter des clients sans perturber le fonctionnement du réseau et sans modification majeur.

Inconvénient de l'architecture client/serveur :

L'architecture client/serveur a tout même quelque lacune :

- **Un maillant faible :**

Le serveur est le seul maillon faible du réseau client-serveur étant donné que tout le réseau est architecturé au tour de lui heureusement le serveur a une grande tolérance aux pannes (notamment grâce à son système RAID)

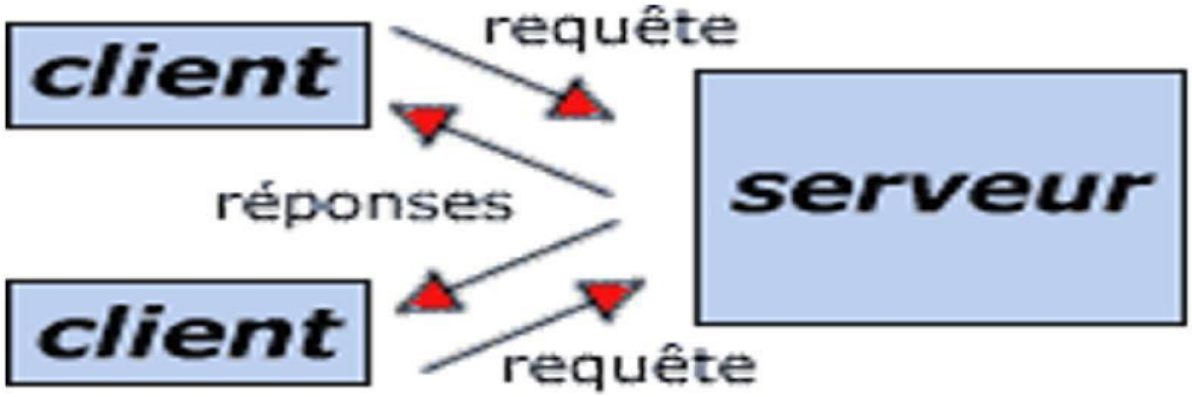

Figure I.1 : architecture client /serveur.

JavaEE.

présentation:

Dans le monde actuel de l'informatique et des technologies de l'information et de la communication les applications sont de plus en plus complexe et doivent être réalise en un minimum de temps et en cout et portable à travers les différent system utilisées. D'où Sun de Microsystème a décidé de mettre en œuvre une plateforme javaEE dédié à la construction des sites web basés sur java.

Architecture de javaEE : [1]

- ✓ Un modèle (Model) contient les données à afficher.
- ✓ Une vue (View) contient la présentation de l'interface graphique.
- ✓ Un contrôleur (Controller) contient la logique concernant les actions effectuées par l'utilisateur.

Figure I.2 : Architecture javaEE.

Les couches logicielles de javaEE : [1]

La plateforme javaEE utilise un modèle d'application distribuée multi-tiers pour des applications d'entreprise.la logique d'application est divisée en composants selon la fonction. Les différents composants qui forment l'application javaEE sont installées sur des machines différentes en fonction du Niveau de l'environnement javaEE multi tiers au quel le composant d'application appartient.

- **La couche client :**

Ses composants s'exécutent sur la machine du client ex logiciel installé en local ou navigateur web.

- **La couche web** :

Ses composants s'exécutent sur le serveur javaEE ex servlet et JSP.

- **La couche métier :**

Ses composants s'exécutent sur le serveur javaEE ex EJB.

- **La couche EIS (entreprise informatique system) :**

Ses composants s'exécutent sur la base de données c'est pour le stockage des informations.

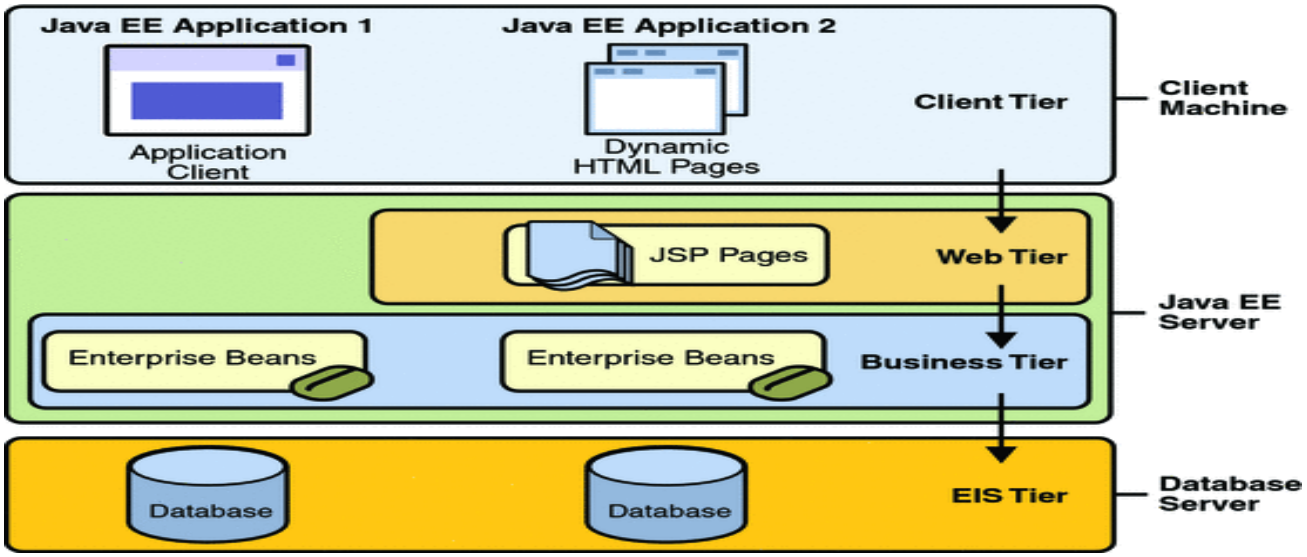

Figure I.3 : les couches logicielles application javaEE.

Les avantages d'utiliser Java EE :

Utilisation de Java EE pour développer exécuter une application en présentant plusieurs avantages :

Une architecture d'application basée sur des composants qui permet un découpage de l'application est donc une séparation des romans de développement.

La possibilité de s'interfacer avec le système d'information existant grâce a de nombreuse API JDBC JNDI JMS JCA.

La possibilité de choisir les outils de développement et les serveurs d'application utilisée qu'ils soient commerciaux ou libre.

Web service.

Définition:

Un service Web est une application logicielle identifiée par un URI dont les interfaces et les liaisons sont définies, décrites et découvertes et supporte une interaction directe avec les autres applications logicielles en utilisant des messages XML ou JSON via un protocole internet.

Les Web services sont la nouvelle vague des applications web. Ce sont des applications modulaire, auto-contenues et auto-descriptives qui peuvent être publiées, localisée et invoquée depuis le web. Les web services effectuent des actions allant de simple requête à des processus métiers complexes. Une fois qu'un web service est déployé, d'autres applications (y compris des web services) peuvent les découvrir et l'invoquer.

le service web REST: [1] [2]

Autre que ce qui précède, il y a une autre approche, permet de faire des services web plus simplement la solution existe et s'appelle REST ou représentation state web transfer .RESET est un style d'architecture de services web qui utilise les standard web déjà très utilise, plus spécifiquement HTTP.

Utilise des conventions inspires de http comme :

- ✓ L'appel d'un service se fait par un url et passe ses paramètres exactement comme une application web conventionnelle.
- ✓ HTTP Get pour un appel de procédure idempotente ne change pas l'état du système disant et peut répéter sans changement a la réponse).

- ✓ HTTP POST pour un appel de procédure nécessitant des paramétré plus complexe.

- ✓ HTTP PUT pour la création d'éléments.
- ✓ HTTP DELETE pour destruction.
- ✓ Les url définissent les ressources (/utilisateur /)
- ✓ La sérialisation d'objet ou de message peut se faire en XML, mais aussi en texte seulement en JSON (JavaScript Object Notation).

La sécurité se fait de la même manière qu'une application web conventionnelle (cors filter, spring security).

Architecture des services web : [1][2]

Les services Web reprennent la plupart des idées et des principes du Web et les appliquent à des interactions entre machines. Comme pour le World Wide Web

les services Web communiquent via un ensemble de technologies fondamentales qui partagent une architecture commune. Ils ont été conçus pour être réalisés sur de nombreux systèmes développés et déployés de façon indépendante. Les technologies utilisées par les services Web sont REST et SOAP.

- **REST :**

REST (Représentationnel State Transfer) est une architecture de services Web. Élaborée en l'an 2000 par Roy Fiedling, l'un des créateurs du protocole HTTP, du serveur Apache HTTP et d'autres travaux fondamentaux,

REST est un style d'architecture qui repose sur le protocole HTTP : On accède à une ressource (par son URI unique) pour procéder à diverses opérations (GET lecture / POST
Écriture / PUT modification / DELETE suppression), opérations supportées nativement par HTTP.

- **SOAP :**

SOAP (Simple Object Access Protocol) est un protocole standard de communication. C'est l'épine dorsale du système d'interopérabilité. SOAP est un protocole décrit en XML et standardisé par le W3C. Il se présente comme une enveloppe pouvant être signée et pouvant contenir des données ou des pièces jointes. Il circule sur le protocole HTTP et permet d'effectuer des appels de méthodes à distance.

les différences entre SOAP et REST : [3]

REST et SOAP sont tous les deux des architectures utilisées pour fournir des services web. Contrairement à ce que l'acronyme SOAP laisse entendre (Simple Object Access Protocol), REST est souvent utilisé lorsque la simplicité de mise en œuvre est recherchée.

REST est lisible (pas d'enveloppe XML superflue) et facile à tester (un navigateur suffit) tout en étant facile de mise en œuvre (un script PHP classique peut souvent être considéré comme RESTful).

SOAP reste toutefois intégré dans de nombreux outils de développements (possibilité d'export de classes en web services, possibilité de génération automatique de clients à partir des WSDL) et permet des contrôles forts sur les types de données attendus.

Dans le cadre de notre étude, nous allons utiliser Rest. Nous présentons son fonctionnement dans ce qui suit.

- **Fonctionnement de web-service rest :**

Figure I.4. : Web service rest.

Creer un service web rest avec spring boot:

- **Exigences** :

✓ Bonne compréhension du langage de programmation Java.

✓ Connaissance de base de Maven.

- **Outils nécessaires :**

✓ Kit de développement Java (JDK) 1.7+

✓ Netbeans, Eclipse ou Spring Tool.

Pour commencer: Ajouter le framework spring boot dans netbeans a travers maven (injecteur de dépendance) et accéder a https://spring.io/ pour crée notre projet tout on spécifiant le GROUP-ID et ARTIFACTE-ID.
Ajoutez l'annotation "@SpringBootApplication" à la classe main pour en faire une application Spring Boot.

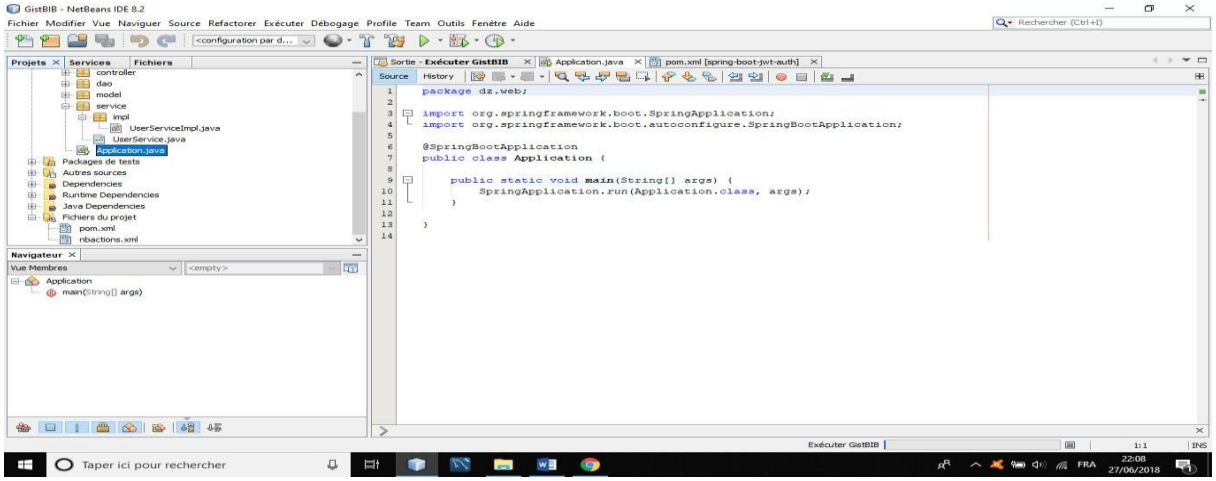

Figure I.5. : Netbeans.

Ensuite on crée notre contrôleur qui sera notre web service et cette classe gère toutes les demandes HTTP entrantes de l'utilisateur et renvoie une réponse appropriée.

L'annotation @RestController indique a spring que cette classe est un web service.

@RequestMapping ("/") annotation signifie les différentes opérations (GET, POST, PUT etc) et la racine c'est-à-dire le modèle concerné.

Conclusion :

Dans ce chapitre nous avons abordé les notions de bases qui nous ont permis de comprendre le fonctionnement du modèle client/serveur, l'architecture J2E et enfin l'architecture des services web, le chapitre suivant est consacré à la présentation de l'analyse et la conception.

Chapitre2

Introduction :

Afin d'aboutir à une meilleure organisation. Nous devons tout naturellement avoir recours un processus de conception orientée objet qui est une démarche de développement et souvent utilisé conjointement au langage UML qui va nous permettre de comprendre et de décrire les besoins de spécifier et documenter le système.

Acteur :

Un acteur est un utilisateur type qui a toujours un comportement vis-à-vis d'un cas d'utilisation.

Les acteurs de notre champ d'étude sont liés à une bibliothèque universitaire (notre système) et sont les suivants:

- ✓ ADMINISTRATEUR.
- ✓ ETUDIANT.

Diagramme de contexte :

Le diagramme de contexte est un diagramme conceptuel de flux qui permet d'avoir une vision globale des interactions entre le système et les liens avec l'environnement extérieure, sachant que notre objectif étant de concevoir et réaliser une application web pour l'administrateur et une application mobile pour l'étudiant.

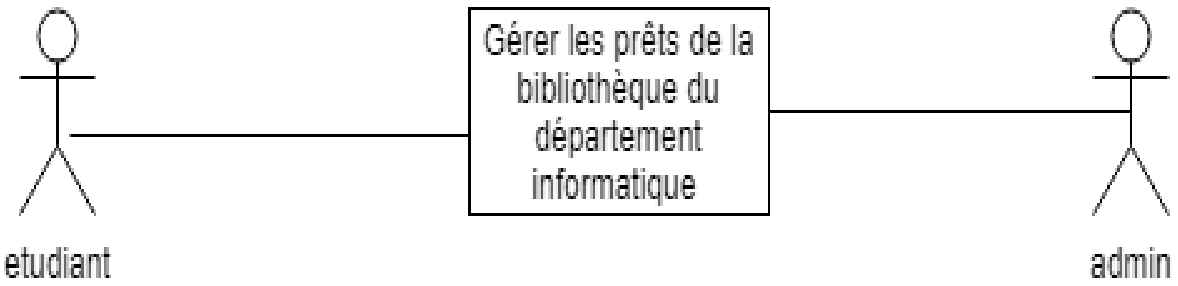

Figure II.1 : Diagramme de contexte.

Les fonctionnalités :

- **Gestion des prêts :** tout lecteur (un étudiant déjà inscrit à la bibliothèque) désirant emprunter un livre ne doit pas être exclu et peut réserver un livre 24h à l'avance et peu le garder au maximum pendant un délai de 15 jours.
- **Gestion réservations :** elle s'effectuer 24h avant le prêt à partir d'un Smartphone.
- **Gestion restitutions :** les livres empruntés doivent être restitue au plus tard 15 jours après la date du prêt sinon la procédure d'exclusion sera déclenchée.
- **Gestion exclusions :** tout retard dans les restitutions est sanctionné par une exclusion des lecteurs pour une durée déterminé par l'administrateur.
- **Gestion des étudiants.**
- **Gestion des livres.**

Spécification des scénarios :

➢ Définition d'un scénario :

Un scénario est un déroulement et description des scènes qui composeront un événement, et qui représente un cas d'utilisation qui peut avoir plusieurs instances.

Les taches de chaque acteur et les scénarios menant à leur réalisation sont définit dans ce tableau suivant:

Acteurs	Taches	Scénarios	
Etudiant		T1 : s'authentifier	S0 accede a l'application S1 écrire username S2 écrire password
		T2 : chercher un livre	S3 choisir le module S4 écrire le titre du livre S5 effectuer la recherche
		T3 : réserver un livre	S6 faire la réservation si le livre est disponible après avoir authentifier
		T4: déconnexion	S7 déconnexion
Admin		T5 : S'authentifier.	S8 accede a l'application S9 écrire username S10 écrire password
		T6 : gérer les étudiants.	S11 ajouter un étudiant S12 modifier un étudiant S13 supprimer un étudiant S14 rechercher un étudiant
		T7 : gérer les livres.	S15 ajouter un livre S16 modifier un livre S17 supprimer un livre S18 rechercher un livre
		T8: gérer les réservations.	S19 confirmer une réservation S20 annuler une réservation
		T9 : gérer les prêts.	S21 ajouter un prêt automatiquement « a partir d'une réservation » S22 ajouter un prêt manuel **NB** la présence de l'étudiant

		guichet est obligatoire
	T10 gérer les restitutions	S23 ajouter à la liste des restitutions si l'étudiant na pas au retard par rapport a la date prévue de la restitution S24 ajouter a la liste des exclut si l'étudiant a fait un retard par rapport a la date prévue de la restitution **NB** la présence de l'étudiant au guichet est obligatoire
	T11 déconnexion	S25 cliquer sur « déconnexion »

Tableau II.1 : spécification des scénarisons.

Diagramme de cas d'utilisation :[5]

Le diagramme de cas d'utilisation est le premier diagramme du modèle UML utilise pour la modélisation des besoins des utilisateurs.

Les cas d'utilisations décrivent le comportement du système étudié du point de vue de l'utilisateur et décrivent la possibilité d'interaction fonctionnel entre le système et les acteurs il permet de définir les relations les limites et les relations entre le système et son environnement il est destiné à structurer les besoins des utilisateurs et les objectifs par rapport au system

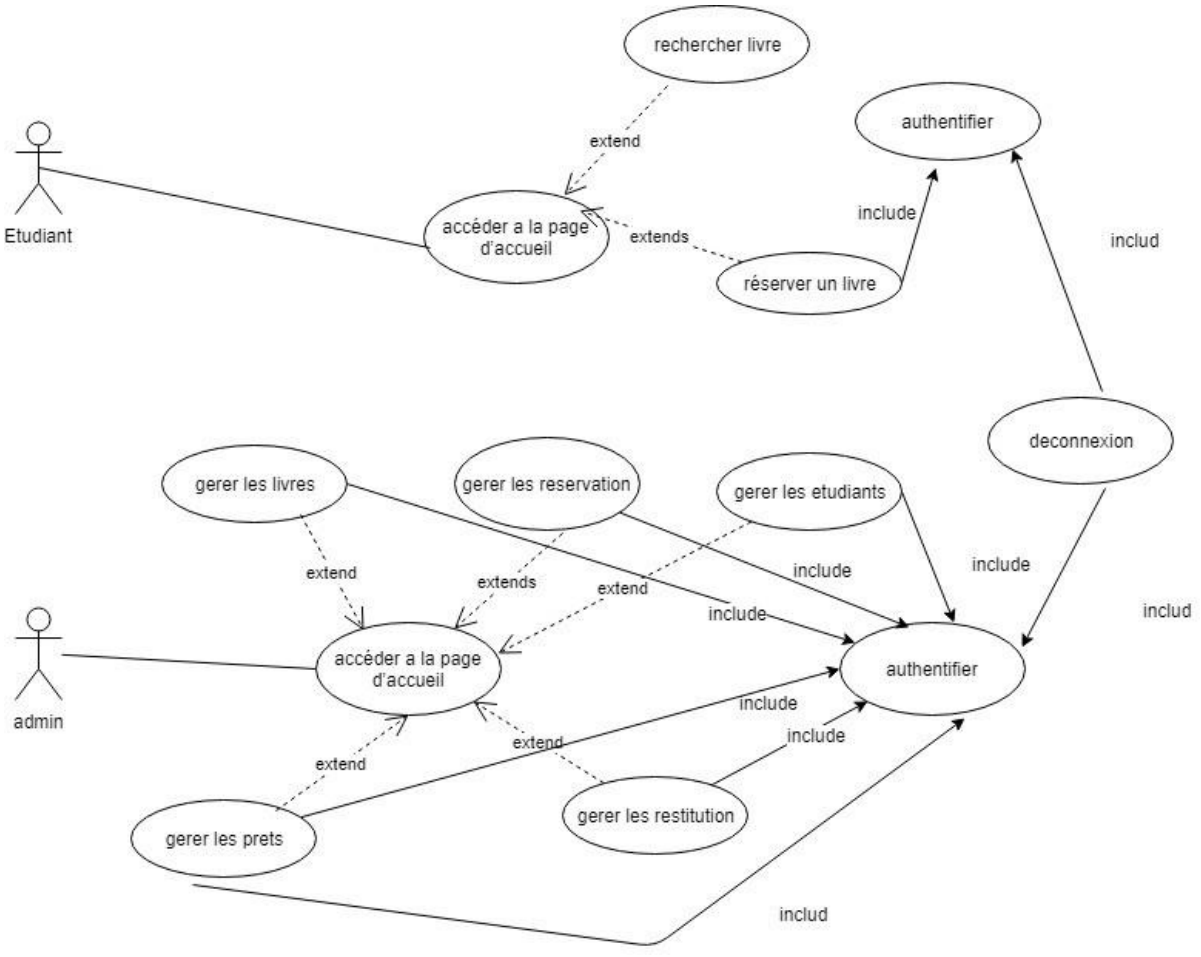

Figure II.2 : cas d'utilisation général.

diagramme de séquence :[5]

Un diagramme de séquence est un diagramme d'interaction qui expose en détail la façon dont les opérations sont effectuées : quels messages sont envoyés et quand ils le sont. Les diagrammes de séquence sont organisés en fonction du temps. Le temps s'écoule au fur et à mesure que vous parcourez la page. Les objets impliqués dans l'opération sont répertoriés de gauche à droite en fonction du moment où ils prennent part dans la séquence de messages.

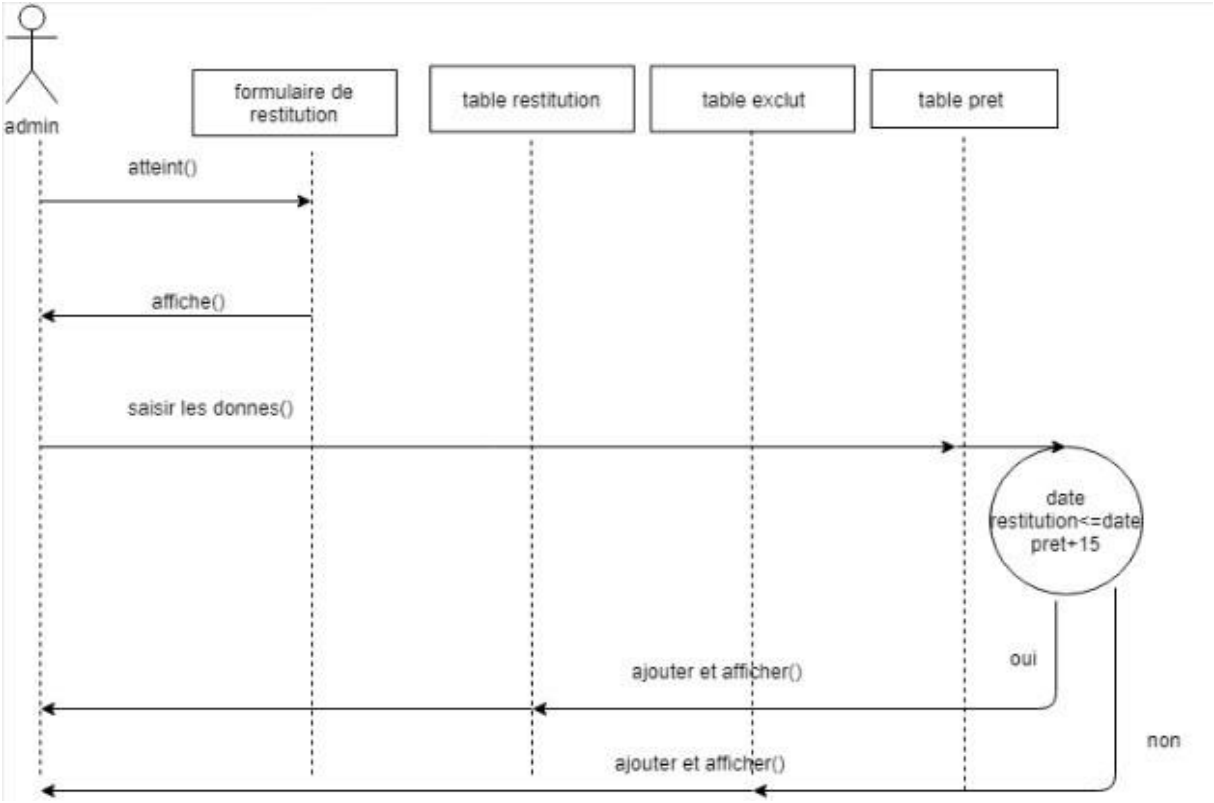

Figure II.3 : Diagramme de séquence dans le cas d'utilisation de la « restitution »

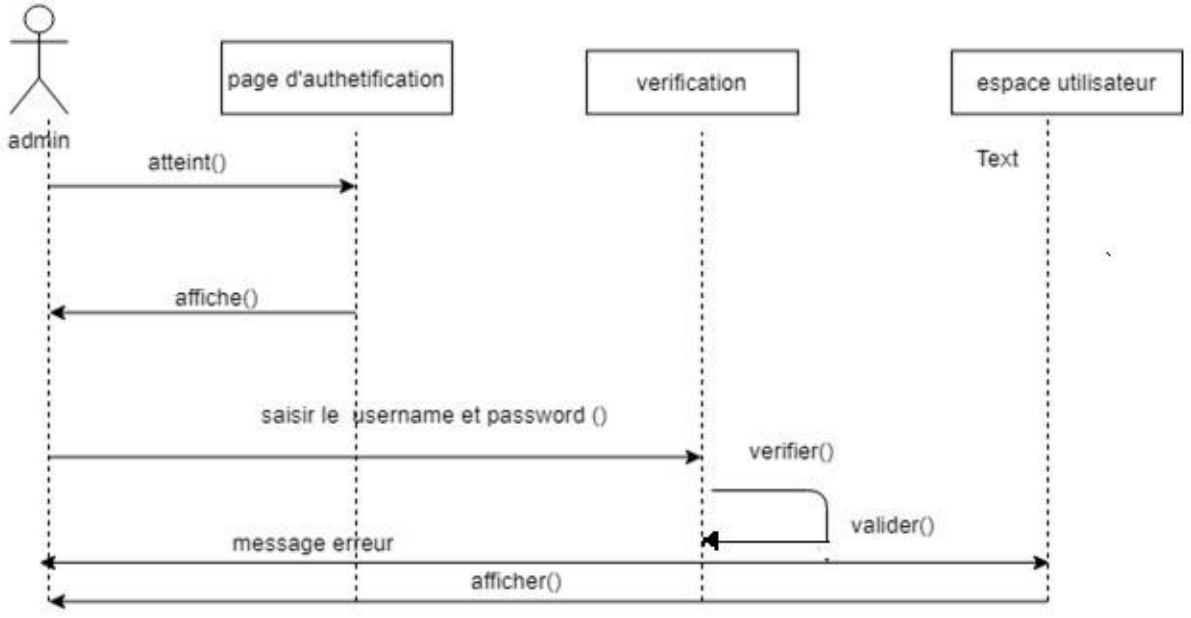

Figure II.4 : Diagramme de séquence dans le cas d'utilisation de la « authentification »

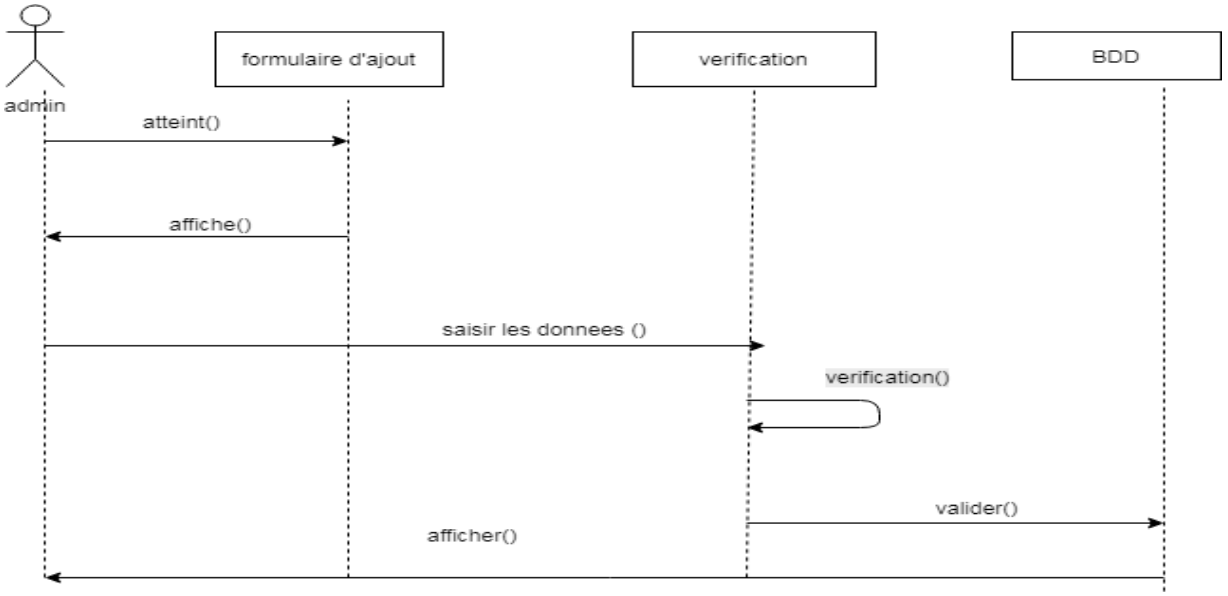

Figure II.5 : Diagramme de séquence dans le cas d'utilisation de la « ajouter»

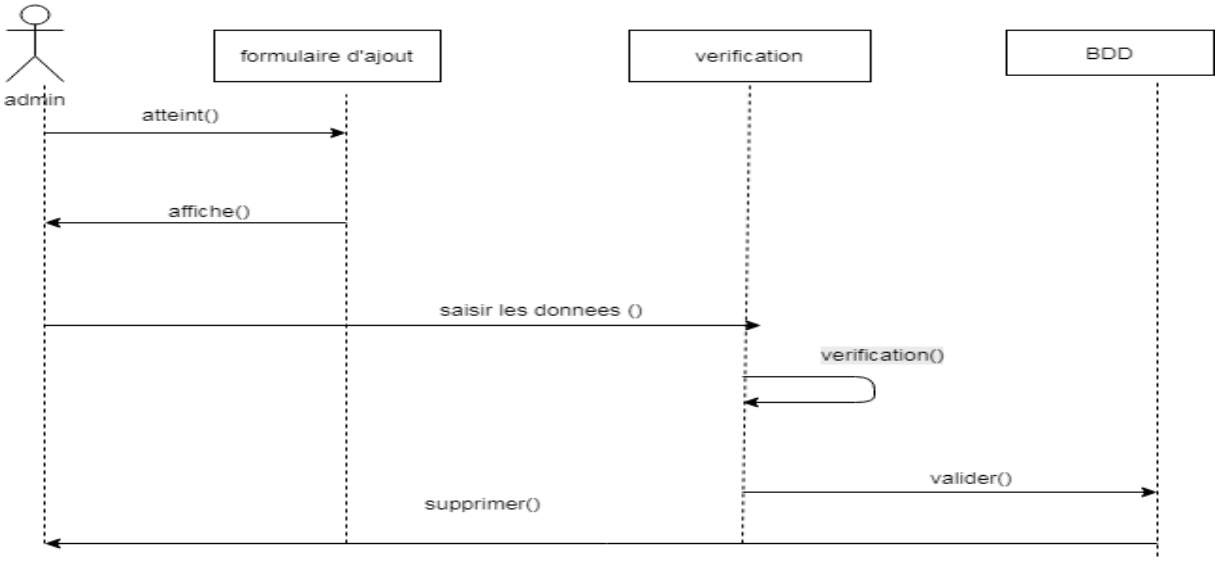

Figure II.6 : Diagramme de séquence dans le cas d'utilisation de la « suppression »

Diagrammes de Classe :[4]

Le diagramme de classes montre les blocs de construction de tout système orienté-objet. Les diagrammes de classes représentent une vue statique du modèle. Ou une partie du modèle, décrivant ce que les attributs et les comportements, qu'il a plutôt que de détailler les méthodes pour atteindre les opérations. Les diagrammes de classes sont les plus utiles pour illustrer les relations entre les classes et les interfaces. Généralisations, agrégations et les associations sont tous précieux reflétant l'héritage, la composition ou l'utilisation, et les connexions respectivement.

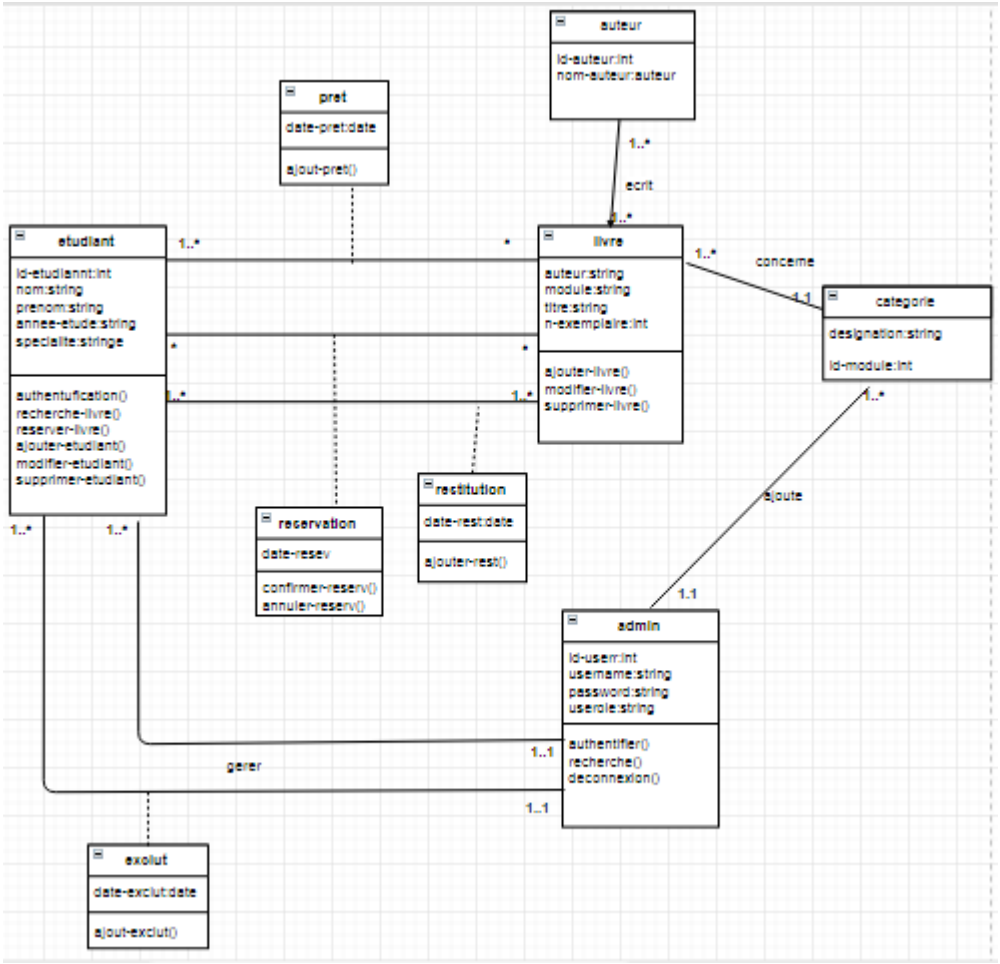

Figure II.7 :diagramme de classe.

Conclusion :

Dans ce chapitre, nous avons introduit les fonctionnalités de notre application. Pour les atteindre, nous avons proposé une solution fondée sur une analyse et une conception modélisées à l'aide du langage de modélisation unifié UML. Pour cela, nous avons défini les acteurs de notre application, les tâches qu'ils assurent et les scénarios associés à chaque tâche. Nous avons élaboré les diagrammes de cas d'utilisation de chaque acteur, élaboré les diagrammes de séquence et enfin le diagramme de classes de quelque cas d'utilisation. Dans le chapitre suivant, qui touche à la réalisation, nous essayerons de construire notre application.

Chapitre3

Introduction :

Après avoir présenté dans le chapitre précédent les différentes étapes d'analyse et de conception nous allons présenter dans ce chapitre l'environnement de développement et les outils qui n'ont servi et nous finirons par la présentation de quelques interfaces du logiciel.

Outils de développement :

- Netbeans :

NetBeans est un projet open source fondée par Sun Microsystems. L'ide NetBeans et un environnement de développement permettant d'écrire compiler déboguer et déployer des programmes il est écrit en Java ne peux supporter n'importe quel langage de programmation il a également un grand nombre de modules pour étendre l'ide NetBeans.

L'ide NetBeans est un produit gratuit.

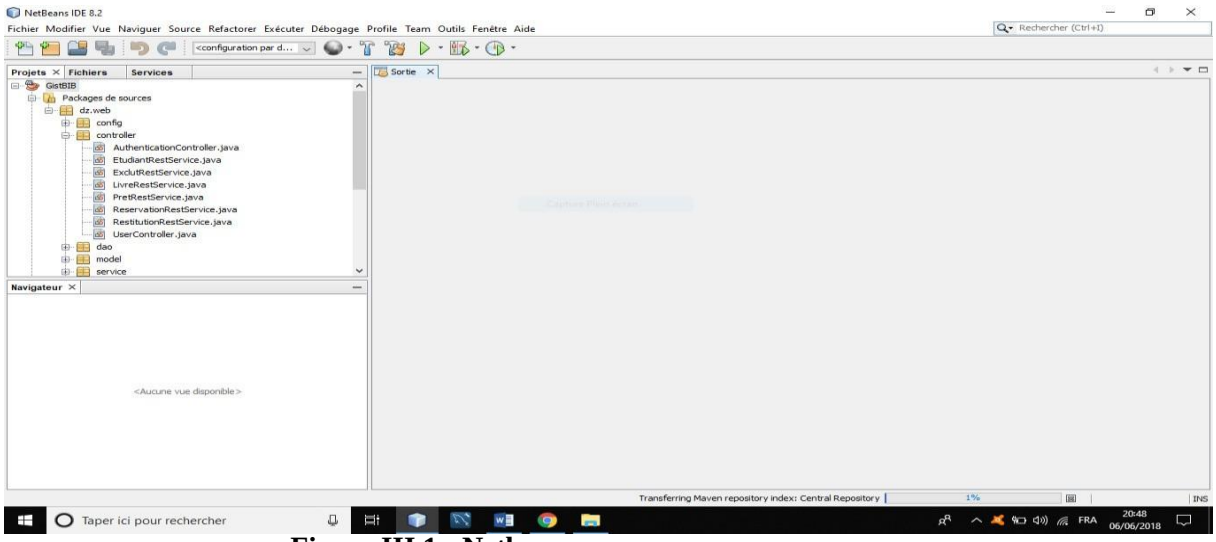

Figure III.1 : Netbeans.

- Le serveur web apache tomcat :

Un serveur web est un logiciel capable d'interpréter les requêtes http arrivant au port associé au protocole http et de fournir la réponse avec le même protocole.

Le serveur web choisie est parmi les meilleurs et de loin le plus répandu : c'est le fruit du mouvement pour le développement du logiciel libre.

Le choix du serveur est basé :

*sur sa **disponibilité** sur toutes les plateformes (Unix, linux, Windows).

* sur un niveau élevé de pour de l'exigence matérielle modeste.

*sur sa **conception modeste** (qualité principale par rapport aux autres serveurs web).

*sur sa **gratuite.**

***sur sa **robustesse.**

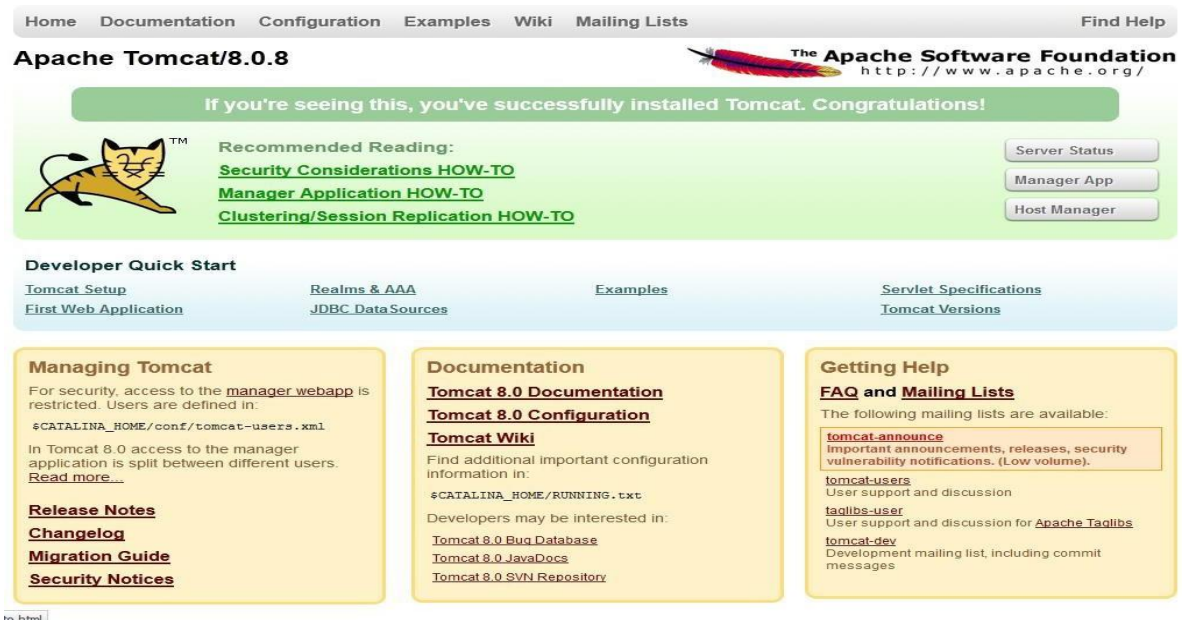

Figure III.2 : Apache tomcat/8.0.8.

- Webstorme : est un IDE puissant pour le développement JavaScript moderne. WebStorm fournit un support complet pour JavaScript, HTML, CSS ainsi que pour les frameworks tels que React, Angular, pour les applications côté client.

Avantages pour l'utilisation de WebStorm :

- réduit les erreurs :

WebStorm analyse notre code tout en tapant et, si une erreur survient, il vous en informera.

- Il nous permet d'écrire du code plus rapidement :

WebStorm a la capacité d'analyser tous vos fichiers de projet HTML et JavaScript. De plus, il nous offre la possibilité de renommer n'importe quelle fonction JavaScript. En parcourant notre code, il corrige toutes les références.

- Il nous permet de crée des applications hybrides :

Il nous permet d'ajouter des frameworks qui nous permettent de le faire comme ionic et cordova

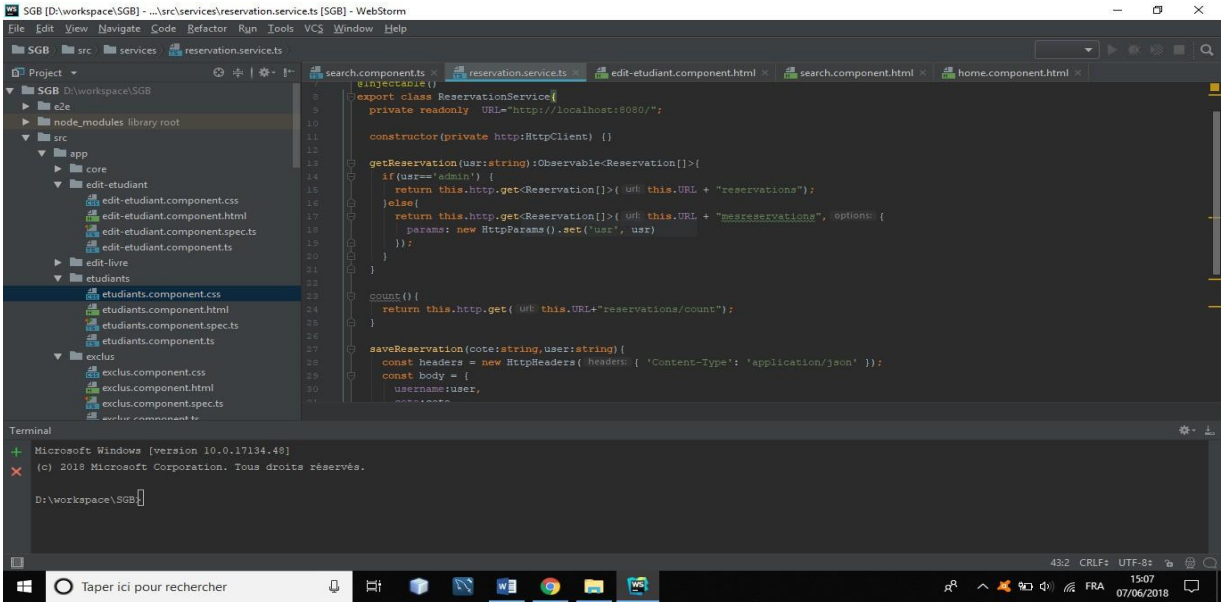

Figure III.3 : WebStorm.

Définition d'une base de données :

Une base de données peut-être défini comme étant un ensemble intégré de données modélisant un univers de données.

III.4.1 MySQL Workbench?

MySQL Workbench est un outil graphique pour travailler avec MySQL.

MySQL Workbench fournit une interface facile à utiliser pour effectuer les nombreuses tâches nécessaires lorsqu'on travaille avec des bases de données. Il intègre le développement SQL, l'administration, la conception de base de données, la création et la maintenance dans un environnement de développement intégré visuel.

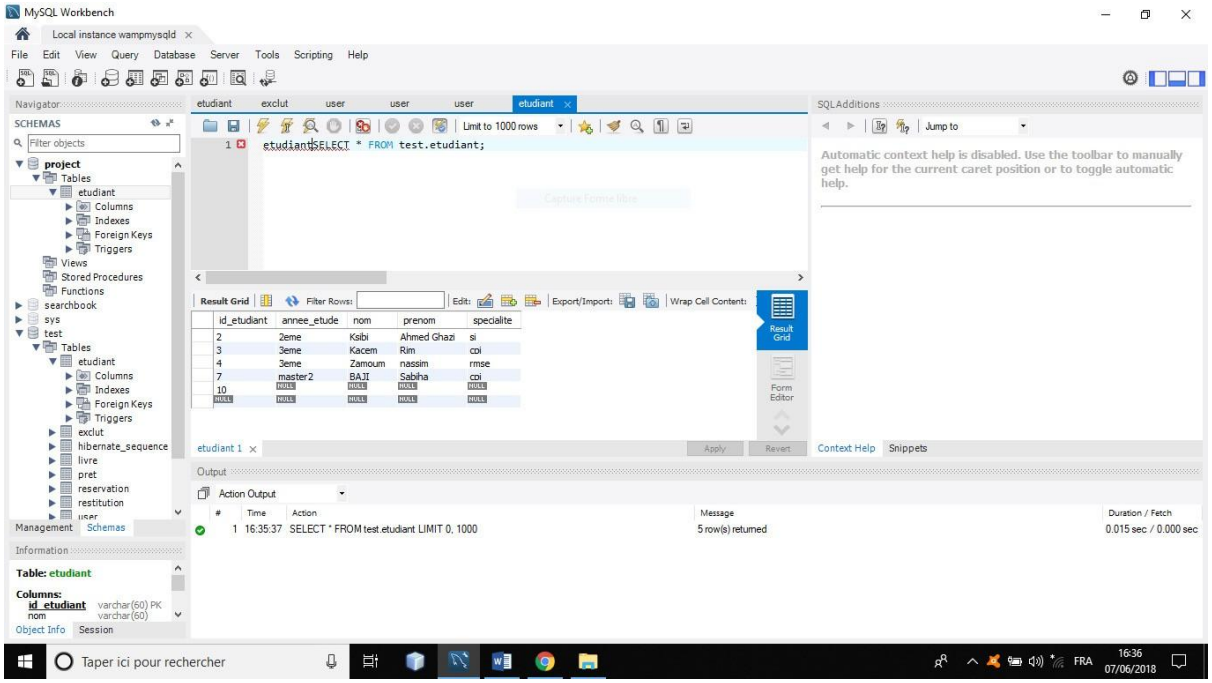

Figure III.4 : MySQL Workbench.

Qu'est-ce qu'un framework ?

Un framework est, comme son nom l'indique en anglais, un "cadre de travail". L'objectif d'un framework est généralement de simplifier le travail des développeurs informatiques (les codeurs si vous préférez), en leur offrant une architecture "prête à l'emploi" et qui leur permette de ne pas repartir de zéro à chaque nouveau projet.

Les framework utilisées:

- ✓ Spring boot.
- ✓ Angular.
- ✓ Ionic.
- ✓ Cordova.

spring Boot ?

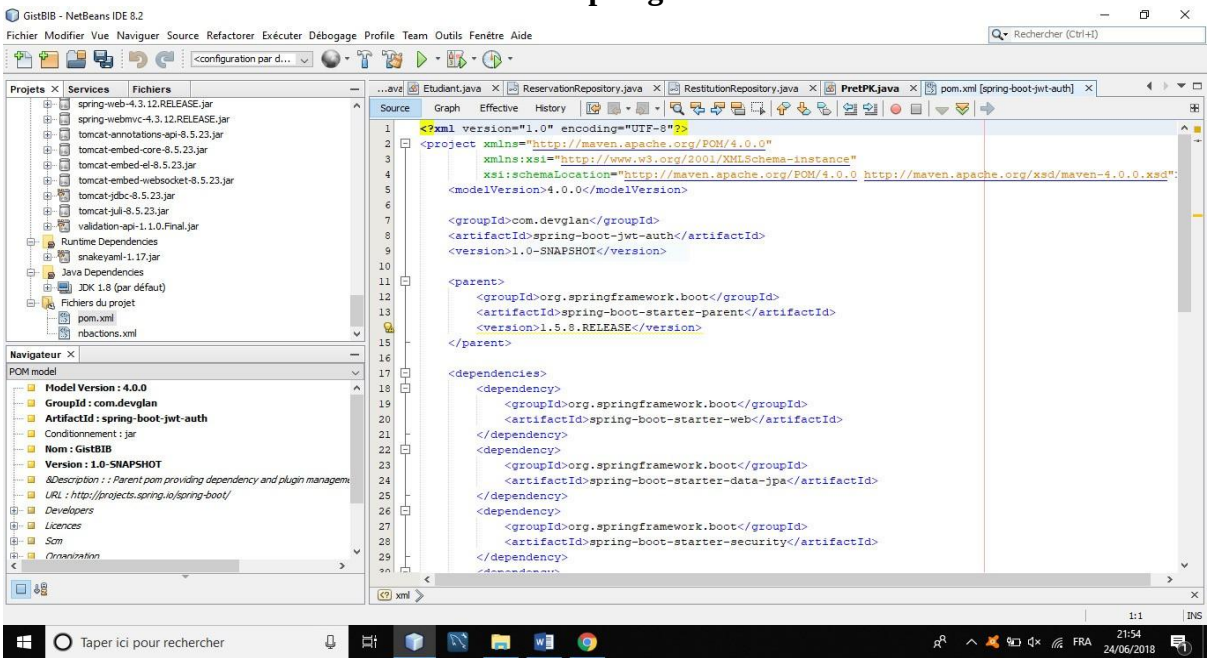

Figure III. 5: Spring Boot.

Spring Boot est un Framework pour faciliter le démarrage et le développement de nouvelles applications Spring et il est destiné spécialement aux applications web.

- Pourquoi Spring Boot ?

✓ Pour faciliter le développement d'applications web avec Java.

✓ Réduire le temps de développement.

✓ Pour augmenter la productivité.

✓ Il évite d'écrire beaucoup de code.

✓ Il fournit des serveurs HTTP comme Tomcat, Jetty, etc. pour développer et tester nos applications web très facilement.

✓ En terminologie simple, Spring Boot signifie

Figure III. 6: Architecture de springBoot.

La notion la plus importante dans spring boot et l'injection des dépendances

Injection des dépendances :

Le concept fondamental du Spring est l'injection des dépendances (dependency injection). C'est une réponse à 3 facteurs du mauvais codage, définis en 1994 par Robert C. Martin :

Et dans notre cas ont a utilisé « MAVEN »

La gestion des dépendances est l'une des fonctionnalités de Maven qui est la mieux connue par les développeurs. Il n'y a pas beaucoup de difficulté à gérer les dépendances pour un seul projet, mais quand lorsqu'on à gérer des projets multi-modules et des applications composées de dizaines ou de centaines de modules, Maven peut nous aider à maintenir un niveau élevé degré de contrôle et de stabilité

1. **Rigidité :** il est difficile de changer quoi que ce soit car chaque changement affecte trop le fonctionnement du système.
2. **Fragilité :** un changement provoque le mauvais fonctionnement des parties qui devraient fonctionner correctement.
3. **Immobilité** : il est difficile de réutiliser le code dans une autre application.

Ces 3 problèmes sont appelés RFI (Rigidity, Fragility, Immobility). L'injection des dépendances est une solution à cette problématique**.**

- **Nodejs :**

Node.js est une plateforme de développement JavaScript. Ce n'est pas un serveur, ce n'est pas un framework, c'est juste le langage Javascript avec des bibliothèques permettant de réaliser des actions comme écrire sur la sortie standard, ouvrir/fermer des connections réseau ou encore créer un fichier.

- **Le gestionnaire de packages npm**

Initialement gestionnaire de packages de Node.js, npm est aujourd'hui le gestionnaire de packages du monde

JavaScript. On y trouve aussi bien des modules pour le backend que pour le frontend.

Npm compte aujourd'hui plus de 500 000 packages et le nombre ne cesse d'augmenter, bien plus rapidement que pour les autres langages.

angular ?

Angular est un framework JavaScript qui étend le HTML pour le rendre dynamique, et permet de développer ses propres balises et attributs HTML. C'est un framework qui se veut extensible et qui pousse vers un développement structuré, en couches, mais bien d'apporter un aspect applicatif au front-end.

- Le Concept de base
- ✓ Data Binding : il s'agit d'un moyen de lier la partie vue à la partie logique. En d'autres termes, grâce à cela, les éléments de votre code HTML seront liés à votre contrôleur JavaScript.Grace a des commandes ng.
- ✓ Routage : angular peut prendre en charge le routage, ce qui signifie passer d'une vue à l'autre. C'est la clé fondamentale des applications d'une seule page; dans lequel on peut

passer à différentes fonctionnalités de notre application Web, mais rester sur la même page.

II.5.3 Ionic ?

Ionic est un framework, c'est-à-dire un ensemble d'outils spécifiques, qui permet de concevoir des applications (hybrides) mobiles de très haute qualité, de manière très rapide**.**

- Application hybride :

Une application hybride est une application pour mobiles qui combine des éléments de web (HTML5, css, js) cette application hybride a des éléments d'une application native permettant d'utiliser les fonctionnalités natives des Smartphones et d'être distribuée en tant qu'application sur les plateformes d'applications (Ios, Android, etc.).Le principe de l'application hybride permet de réduire les coûts et délais de développement nécessaires pour proposer plusieurs applications natives pour les différents systèmes d'exploitation mobiles.

III.5.4 pache Cordova ?

Il s'agit d'un Framework – un ensemble d'outils de développement – permet de saisir les différentes fonctionnalités qu'un smart phone offre exemple camera géolocalisation et ceci sur chaque plateforme ou il a été déployé l'ionic

- Configurer Ionic pour Android :

Grâce à npm, il est très facilement de mettre en place un environnement de développement Ionic en quelques commandes.

Installer Ionic et Cordova

```
$ npm install -g ionic cordova
```

Créer un projet Ionic

```
$ ionic start nom-application blank
```

Configurer votre plate-forme

```
$ ionic platform add android
```

Construire et émuler l'app

```
$ ionic build android
$ ionic emulate android
```

Quelque interface de l'application :

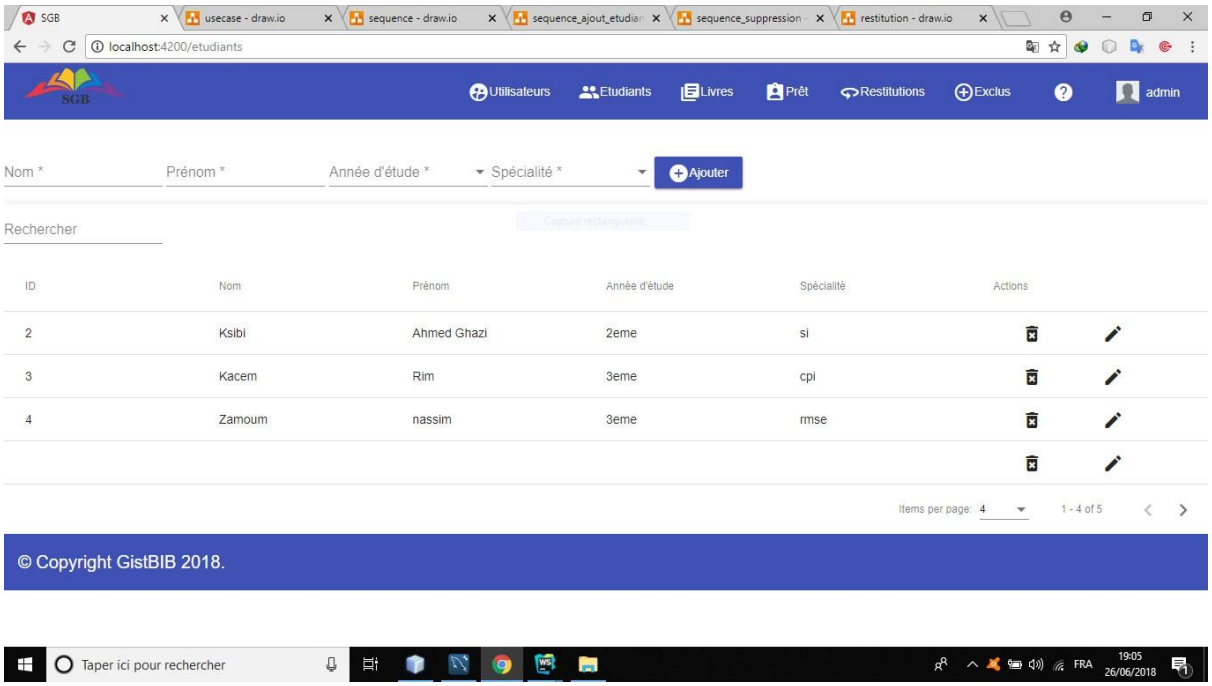

Figure III.7 : l'interface Ajouter étudiant.

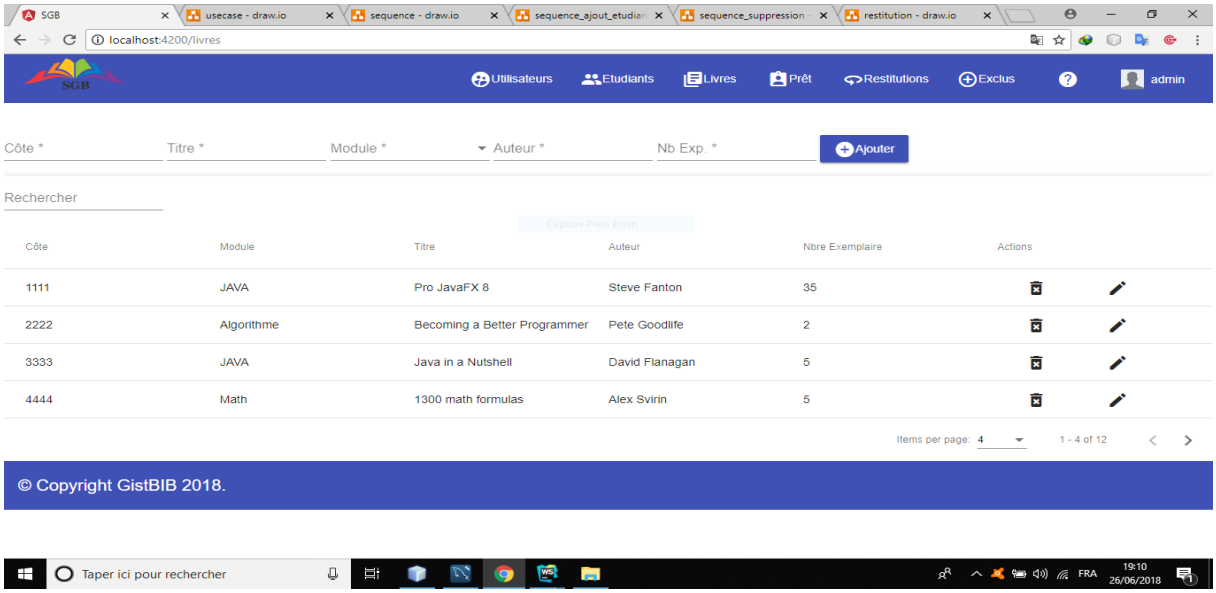

Figure III.8 : l'interface ajouter livre.

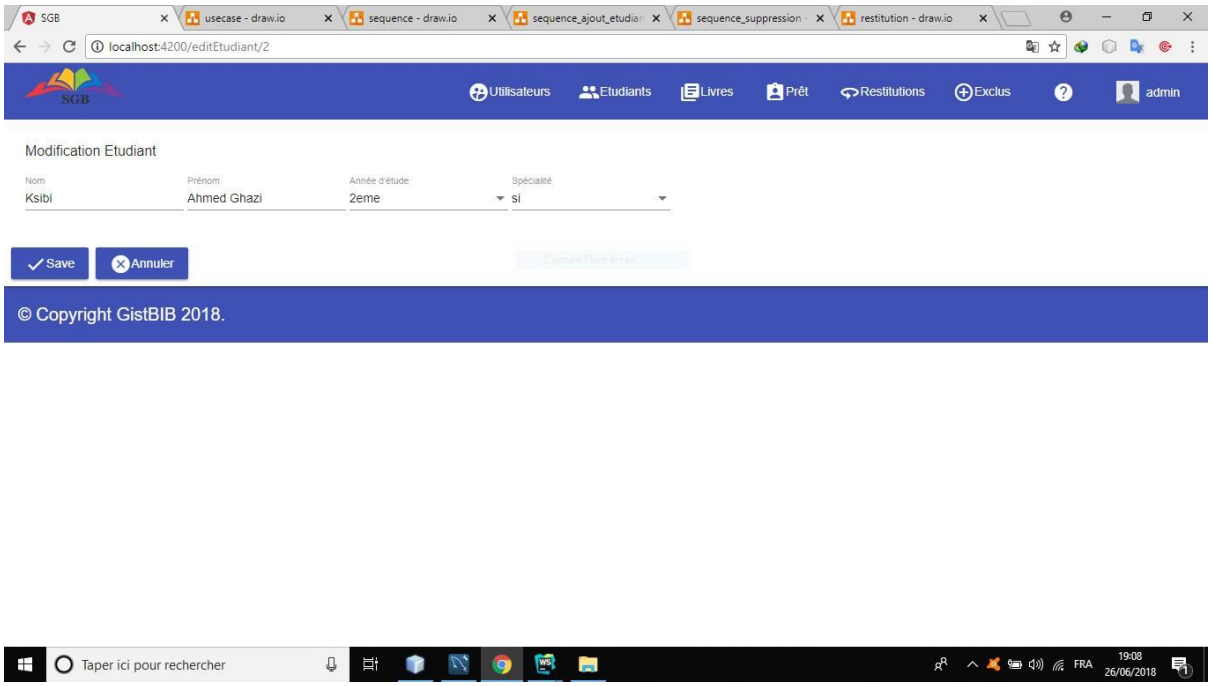

Figure III.9 : l'interface Modifier étudiant.

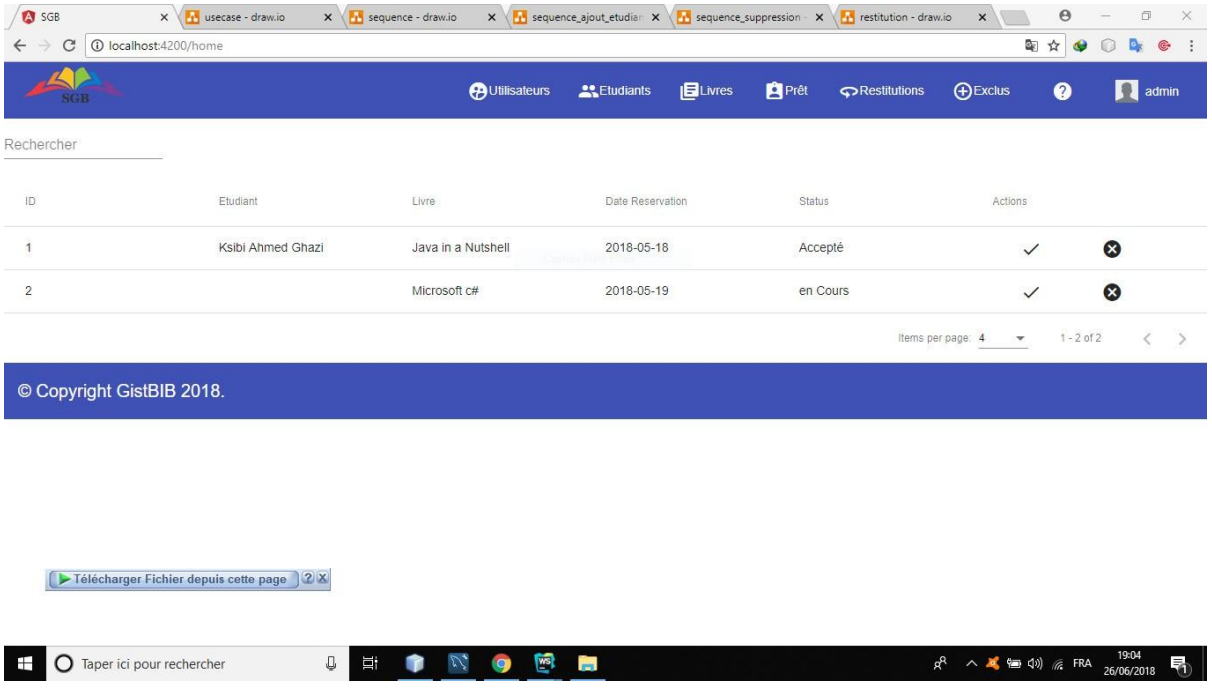

Figure III.10 : l'interface Page réservation.

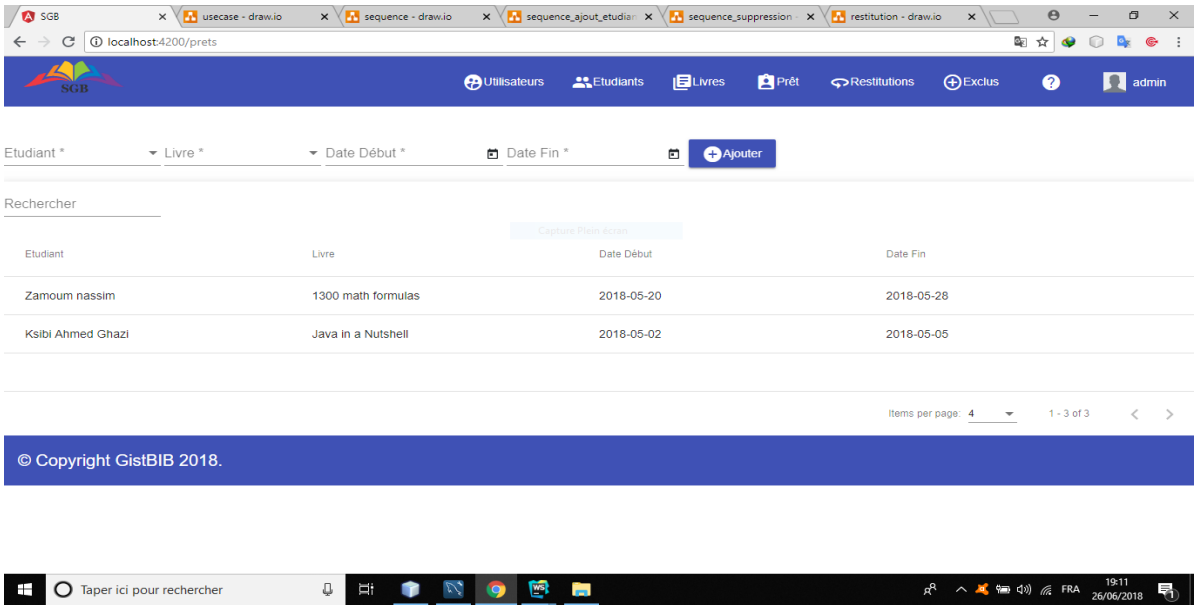

Figure III.11 : l'interface des Prêt.

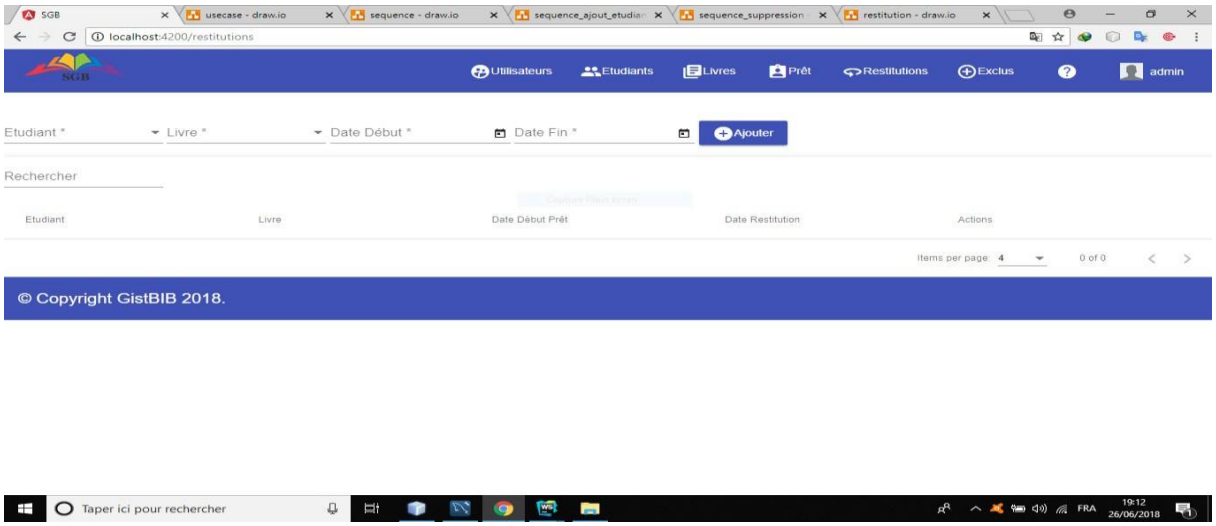

Figure III.12 : l'interface de Restitution.

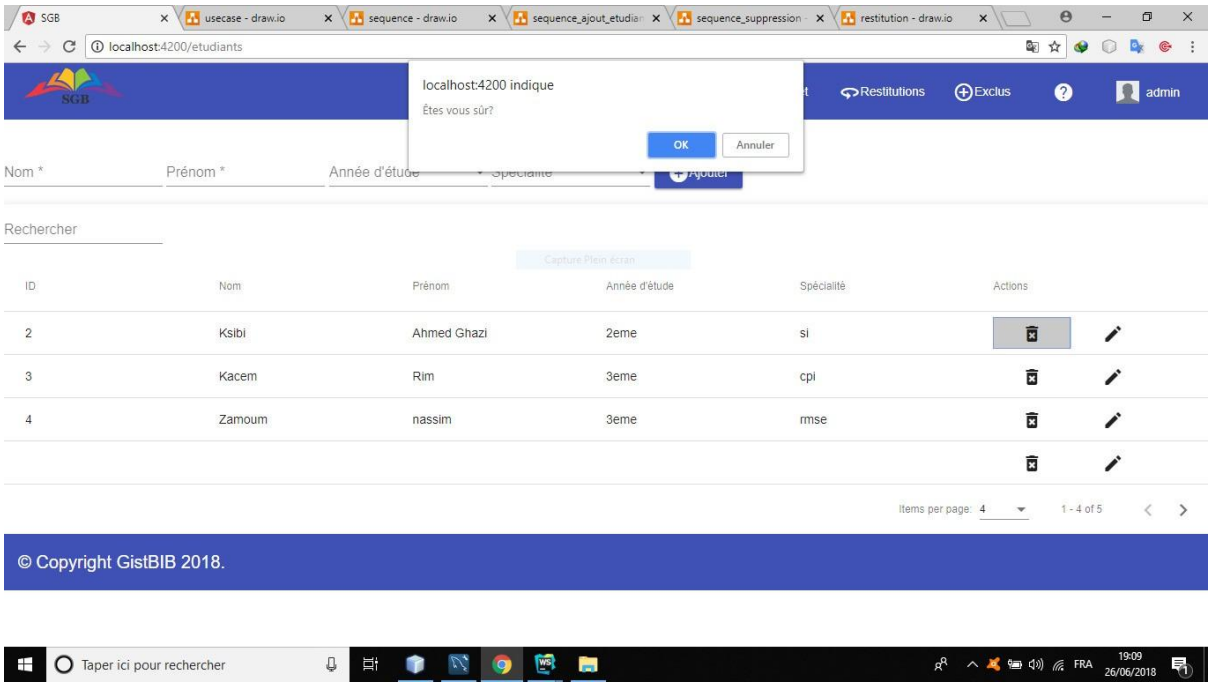

Figure III.13 : l'interface Supprimer étudiant.

Les tables :

- ✓ Table etudiant :

Champs	Type	Cle
Id_etudiant	Int	primaire
Nom	String	
Prenom	String	
Annee_etude	String	
Specialite	String	

- ✓ Table livre :

Champs	Type	cle
Cote_livre	Int	primaire
Auteur	String	
Module	String	
Titre	String	
Nb_exemplaire	Int	

- ✓ Table exclu :

Champs	Type	Observation
Id_etudiant	Int	Primaire
Date restitution	String	
Date fin	String	

- ✓ Table reservation :

Champs	Type	Clé
Id_reservation	Int	Primaire
Id_etudiant	Int	Etrangere
Cote_kivre	String	Etrangere
Date reservation	Date	
statut	String	

- ✓ Table restitution :

Champs	Type	Clé
Id_restitution	Int	Primaire
Id_etudiant	Int	Etrangère
Cote_kivre	String	Etrangère
Date_debut_pret	Date	
Date_restitution	Date	

- ✓ Table prêt :

Champs	Type	clé
Id_pret	Int	primaire
Id_etudiant	Int	étrangère
Cote_kivre	String	étrangère
Date_reservation	Date	
Date_fin_pret	Date	

- ✓ Table admin :

Champs	Type	cle
Id_admin	Int	primaire
Username	String	
Password	String	

Conclusion :

Dans ce chapitre, nous avons présenté la conception de l'application en s'intéressant à la modélisation basée sur la méthode UML.

Nous avons entamé le présent de chapitre par la spécification des besoins et la présentation de divers cas d'utilisations, puis la conception des diagrammes de séquences, ceci pour la phase d'analyse. En ce qui concerne la phase de conception nous avons élaboré les diagrammes de classes.

A la fin de chapitre, nous avons défini les différentes tables de notre base de données avec leurs relations ainsi que le schéma conceptuel de données.

Conclusion générale :

L’objectif de notre travail consistait à concevoir et réaliser une application web et mobile pour la gestion de la bibliothèque.

La réalisation de ce travail nous a donné l’occasion d’acquérir de nouvelles connaissances et d’en approfondir d’autres sur le développement des applications Web et application mobile et web service on citera le langage de scripts javasEE, le langage de requête SQL etc. Ça nous a permis aussi de nous familiariser avec un certain nombre d’outils informatique de développement à Titre d’exemple nous citons Apache tomcat, MySQL Workbench, Webstorm et un accès distant à partir d’un lap top et de mettre en nos connaissances préalables sur la gestion de la bibliothèque.

En perspectives, nous pensons à étendre notre applications a la gestion des autres lecteurs enseignants et fonctionnaires, étendre l’application aux bibliothèques de l’université et améliorer notre application.

Bibliographie :

Ouvrages consultes lors de ce travail :

[HUK 03]Huber Kadima.

[Bry04] Sean Brydon&BethhStreans, Designing Web services with the J2EE 1.4 Platform.

[VM03]ValérieMonfort.

[JF02]Jeremy Fierstonnc.

[Juric05]M.juric.

[Fost03]H.FOSTER,S.UCHITEL,J.KREMER.model-based verification of web service composition, oct 2003.

[KUZ07]Kuzmanovic, 2007.

[Dor01][SM02]Dor et al, 2001, Simon et Kinng 2002.

[VL06]Vincent Lamarielle. XML schema et XML infoset pour les services web, CEPADUES, 2006.

[B&A095]Boasson& al 95.

[JD94]Jean Michel DOUDOUX.

[A&J&C90]Andew,Jum et conallen 1990.

Sites consultes :

http://www.google.fr/search?hl=fr&site=&source=hp&q=s%C3%A9curit%C3%A9+des+web+services&btnnk=recherche+google.

http://www.softeam.fr/technologie_web_services.php.2007.

http://www.soapuser.com/fr/basics3.hhtml.

http://www.Isdis.cs.uga.odu/proj/meteor/mwscf/stanndards.html.

http://www.bpms.nfo/lexique_detailasp?ref=145.

http://www.guideinformatique.com/fichhe-informatique_distribue_et _linteroperabilites-345.html.