

République Algérienne Démocratique et Populaire  
Ministre de l'Enseignement Supérieur et de la Recherche Scientifique  
Université Abderrahmane Mira-Bejaia

---

Faculté des Sciences Exactes  
Département Informatique

---



## PROJET FIN DE CYCLE

---

### T H È M E

*Conception et réalisation d'une application desktop  
pour la gestion de la bibliothèque universitaire*

---

PRÉSENTÉ PAR :

BOUREDJOUA *Amine* C2    MABED *Zoubir* C3

KACI *Aziz* C2    MADAOUY *Celine* C3

MEZHOUD *Rafik* C4

Encadrant :            D<sup>r</sup> H. GADOUCHE

Année universitaire : 2020/2021

---

# TABLE DES MATIÈRES

---

1	CAHIER DES CHARGES .....	7
1.1	PROBLÉMATIQUE : .....	7
1.2	OBJECTIF : .....	8
1.3	BESOINS FONCTIONNELS ET NON FONCTIONNELS : .....	8
1.3.1	Besoins fonctionnels : .....	8
1.3.2	Besoins non fonctionnels : .....	8
2	ANALYSE ET CONCEPTION DU SYSTÈME .....	10
2.1	MÉTHODOLOGIE ET APPROCHE ADOPTÉE .....	10
2.2	UML (UNIFIED MODELING LANGUAGE) .....	11
2.3	DIAGRAMME DE CAS D'UTILISATION .....	11
2.3.1	Diagramme de cas d'utilisation "Etudiant" .....	12
2.3.2	Diagramme de cas d'utilisation "Administrateur" .....	12
2.3.3	Diagramme de cas d'utilisation "Assistant" .....	13
2.3.4	Diagramme de cas d'utilisation général .....	14
2.4	DESCRIPTIONS TEXTUELLES DES CAS D'UTILISATION .....	15
2.4.1	Authentification .....	15
2.4.2	Demande de prolongation .....	16
2.4.3	Consulter la liste des ouvrages .....	17
2.4.4	Gérer des assistants .....	18
2.4.5	Gérer des ouvrages .....	19

2.4.6	Gérer des prêts .....	20
2.5	DIAGRAMME DE SÉQUENCE.....	21
2.5.1	Diagramme de séquence "Authentification" .....	21
2.5.2	Diagramme de séquence "Réservation d'un ouvrage" .....	22
2.5.3	Diagramme de séquence "Mise à jour" .....	23
2.6	DIAGRAMME DE CLASSE.....	24
2.7	MODÈLE RELATIONNEL.....	25
3	IMPLÉMENTATION .....	27
3.1	ENVIRONNEMENT DE DÉVELOPPEMENT :.....	27
3.1.1	IDE NetBeans.....	27
3.1.2	Wamp.....	27
3.1.3	Visual Paradigm for UML .....	28
3.1.4	Adobe Photoshop.....	28
3.1.5	Adobe Illustrator .....	28
3.2	PRÉSENTATION DE L'APPLICATION.....	29
3.2.1	Splash Screen.....	29
3.2.2	Authentification .....	30
3.2.3	Interface Administrateur.....	30
3.2.4	Interface Assistant.....	31
3.2.5	Interface Etudiant.....	31
4	CONCLUSION .....	32

---

# TABLE DES FIGURES

---

1	Diagramme de cas d'utilisation "Etudiant" . . . . .	12
2	Diagramme de cas d'utilisation "Administrateur" . . . . .	12
3	Diagramme de cas d'utilisation "Assistant" . . . . .	13
4	Diagramme de cas d'utilisation général . . . . .	14
5	Diagramme de séquence "Authentification" . . . . .	21
6	Diagramme de séquence "Réservation d'un ouvrage" . . . . .	22
7	Diagramme de séquence "Mise à jour" . . . . .	23
8	Diagramme de classe . . . . .	24
9	Splash Screen de l'application . . . . .	29
10	Interface authentification . . . . .	30
11	Interface administrateur . . . . .	30
12	Interface assistant . . . . .	31
13	Interface étudiant . . . . .	31

---

# REMERCIEMENTS

---

*C'est avec un grand plaisir qu'on réserve ces quelques lignes en signe de gratitude et de profonde reconnaissance à tous ceux qui, de près ou de loin, ont contribué à la réalisation et l'aboutissement de ce travail.*

*Nous souhaiterons tout d'abord remercier Dieu qui nous a maintenu en santé et nous a doté d'intelligence.*

*Nous tenons aussi à adressé nos remerciements à notre encadrant madame H.GADOUCHE pour ses conseils et encouragements, et monsieur Z.FARAH pour sa disponibilité.*

*Nous nous acquittons, enfin, volontiers d'un devoir de gratitude et de remerciements à tous nos enseignants pour la qualité de l'enseignement qu'ils ont bien voulu nous prodiguer durant nos études afin de nous fournir une formation efficiente.*

---

# RÉSUMÉ

---

La bibliothèque universitaire est un ensemble de sources d'informations, rendues accessibles à une communauté définie pour référence ou emprunt. Ainsi, le processus de gestion manuelle d'une bibliothèque est très complexe et inefficace. En ce qui concerne ce point de vue, le système informatisé de gestion des activités de la bibliothèque offre un moyen complet de gérer le travail physique, et de réduire la complexité du système manuel.

Ce projet vise à concevoir et à mettre en œuvre un système entièrement réactif de gestion de bibliothèque informatisé. Le système de gestion des bibliothèques a été conçu et implémenté en utilisant le JAVA (NetBeans comme éditeur) et My SQL database. Le système a été développé en utilisant le modèle de conception UML.

Une évaluation approfondie du projet détermine que le projet a atteint bon nombre de ses objectifs prédéfinis des fonctionnalités simples comme la gestion des documents et des utilisateurs à la navigation et la réservation en ligne avec d'autres fonctionnalités essentielles de la bibliothèque.

---

# CAHIER DES CHARGES

---

## 1.1 Problématique :

Plusieurs parmi nous ont déjà emprunté un livre dans une bibliothèque, du moins essayé de le faire. Cependant, cette procédure n'a pas toujours été agréable. Effectivement, au long du processus d'emprunt, l'étudiant rencontre certains obstacles :

- Le travail est mal organisé par rapport à la mauvaise gestion.
- L'assistant n'a pas la possibilité de rappeler le délai, de ce fait, la bibliothèque subit une perte d'ouvrages.
- Absence de vérification de la disponibilité des ouvrages.
- Manque d'informations sur les ouvrages.
- Limitation des heures de travail, ce qui engendre généralement une perte de temps.

Toutes ces contraintes nous ont poussé à établir un plan de conception d'une application de gestion d'une bibliothèque en ligne, qui permet de résoudre ces problèmes.

## 1.2 Objectif :

L'objectif de cette application est de résoudre les problèmes rencontrés dans un système de gestion d'une bibliothèque manuel, c'est-à-dire offrir une bonne gestion à la bibliothèque en informatisant la procédure de réservation des ouvrages et même la façon de consultation, tout en facilitant le travail pour l'assistant de la bibliothèque et l'utilisation pour l'étudiant. On optera alors pour trois espaces d'authentification (Étudiant, assistant et administrateur).

## 1.3 Besoins fonctionnels et non fonctionnels :

### 1.3.1 Besoins fonctionnels :

Il s'agit des fonctionnalités de l'application, dans notre cas :

- **Un espace d'authentification** : C'est l'interface principale qui permet l'accès aux fonctionnalités de l'application (L'utilisateur doit s'authentifier pour avoir cet accès).
- **Un espace administrateur** : C'est l'interface qui englobe les fonctionnalités d'un administrateur (Mise à jour des ouvrages et étudiants).
- **Un espace assistant** : C'est l'interface qui permet à un assistant de gérer l'ajout, l'emprunt et la réservation des ouvrages.
- **Un espace étudiant** : C'est l'interface de consultation pour l'étudiant où il pourra consulter et réserver des ouvrages.

### 1.3.2 Besoins non fonctionnels :

Dans cette partie, on présente les besoins qui ont une relation avec la performance du système en temps de réponse et stockage mémoire, la sécurité, la facilité d'utilisation et l'ergonomie de l'interface graphique. Pour notre application on s'est basé sur les points suivants :



- **Graphisme :**

- Les couleurs principales seront : le blanc et le bleu qui correspondent aux couleurs de notre plate-forme E-learning ainsi qu'au logo de notre université, toutefois on pourra intégrer du gris à notre interface.
- On a privilégié une police simple, facilement lisible et prise en charge par la plus part des appareils, dans notre cas on a choisi (Calibri).

- **Ergonomie :**

Le but est d'offrir une interface simple et facile à utiliser avec un espace dédié à chaque utilisateur (Étudiant, assitant, administrateur).

- **Fiabilité :**

- La possibilité de mettre à jour des informations.
- La sécurité de l'accès à l'application via l'authentification.

- **Performance :**

Satisfaire l'utilisateur en répondant à ces besoins et cela dans un minimum de temps de réponse.

Dans ce chapitre, nous avons pu collecter des informations et présenter l'organisation de la bibliothèque universitaire et sa gestion. Afin de nous donner un plan général tout en essayant de prendre en compte la direction de l'analyse et la conception des données pour la rédaction du chapitre suivant qui définit la conception de notre système.

---

# ANALYSE ET CONCEPTION DU SYSTÈME

---

La conception est une étape préliminaire et primordiale qui doit précéder l'étape de développement de toute application informatique. Pour décrire la conception de l'application, on commencera avec le diagramme de cas d'utilisation et séquence. Par la suite on passera au diagramme de classes, puis au modèle relationnel.

## 2.1 Méthodologie et approche adoptée

Avant de programmer l'application et se lancer dans l'écriture du code, il faut tout d'abord organiser les idées, les documenter, puis organiser la réalisation en définissant les modules et les étapes de la réalisation. Cette démarche antérieure à l'écriture que l'on appelle modélisation.

La modélisation consiste à créer une représentation virtuelle d'une réalité de telle façon à faire ressortir les points auxquels on s'intéresse. Dans le cadre de notre projet on a utilisé la méthodologie UML pour la modélisation des différents diagrammes.

## 2.2 UML (Unified Modeling Language)

UML est un langage de modélisation unifié permet de modéliser une application logicielle d'une façon standard dans le cadre de conception orienté objet.

UML permet de couvrir le cycle de vie d'un logiciel depuis la spécification des besoins jusqu'au codage en offrant plusieurs moyens de description et de modélisation des acteurs et d'utilisation système, du comportement des objets, du flot de contrôle internes aux opérations, des composants d'implémentation et leurs relations, de la structure matérielle et de la distribution des objets et des composants indépendamment des techniques d'implémentation et peut être mis à jour selon les besoins.

## 2.3 Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation (DCU) sont des diagrammes UML utilisés pour donner une vision globale du comportement fonctionnel d'un système logiciel. Ils sont utiles pour des présentations auprès de la direction ou des acteurs d'un projet, mais pour le développement, les cas d'utilisations sont plus appropriés. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur (humain ou machine) et un système. Il est une unité significative de travail. Dans un diagramme de cas d'utilisation, les utilisateurs sont appelés acteurs, ils interagissent avec les cas d'utilisation.

Pour notre application nous avons trois acteurs qui sont l'administrateur, assistant et étudiant.

### 2.3.1 Diagramme de cas d'utilisation "Etudiant"

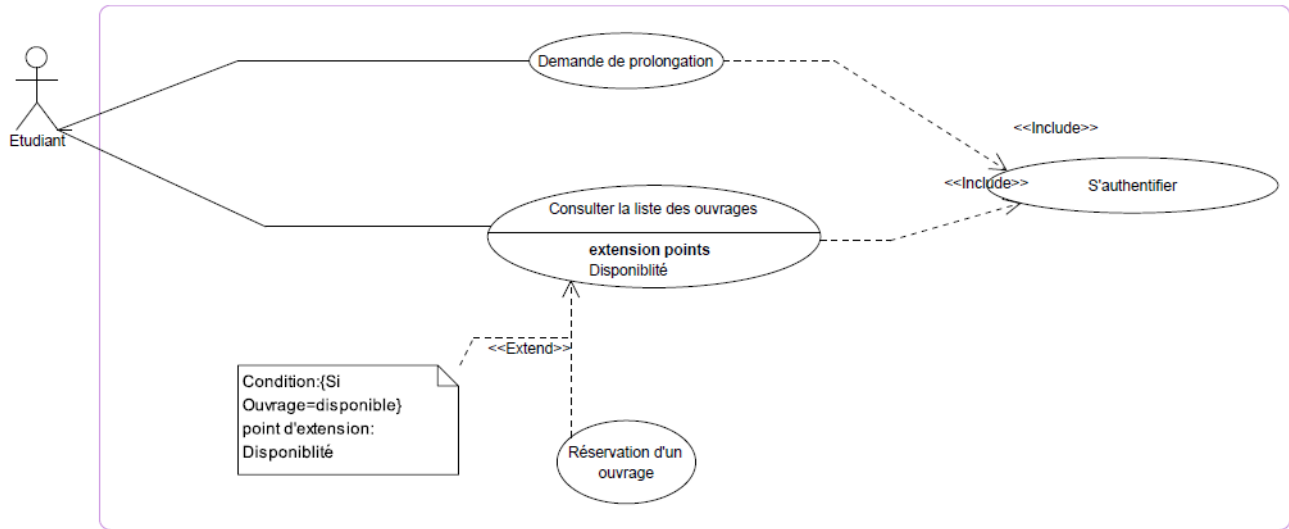


FIGURE 1 – Diagramme de cas d'utilisation "Etudiant"

### 2.3.2 Diagramme de cas d'utilisation "Administrateur"



FIGURE 2 – Diagramme de cas d'utilisation "Administrateur"

### 2.3.3 Diagramme de cas d'utilisation "Assistant"

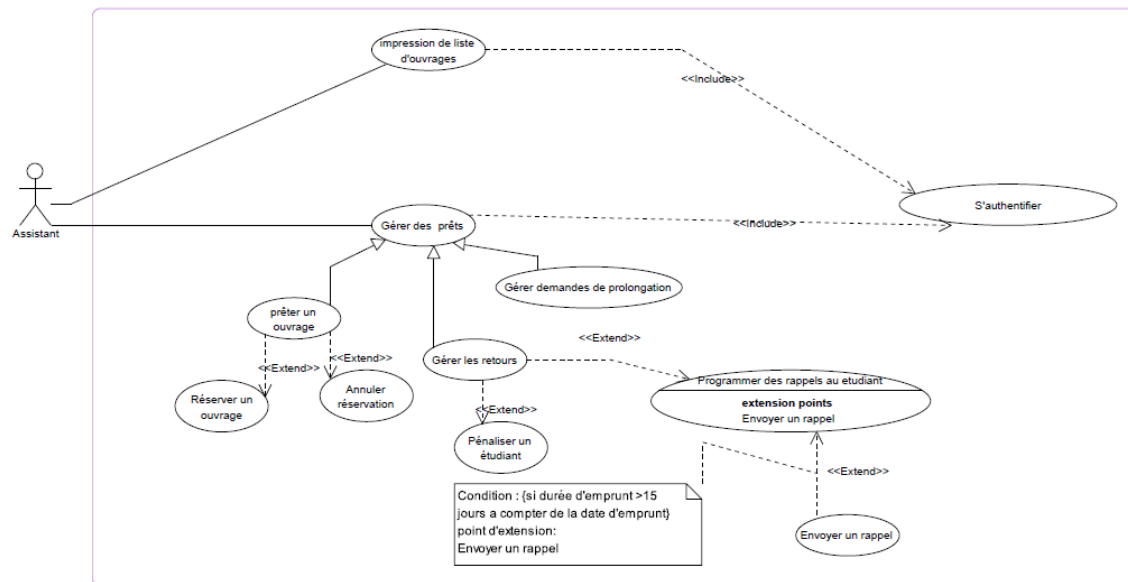


FIGURE 3 – Diagramme de cas d'utilisation "Assistant"

### 2.3.4 Diagramme de cas d'utilisation général

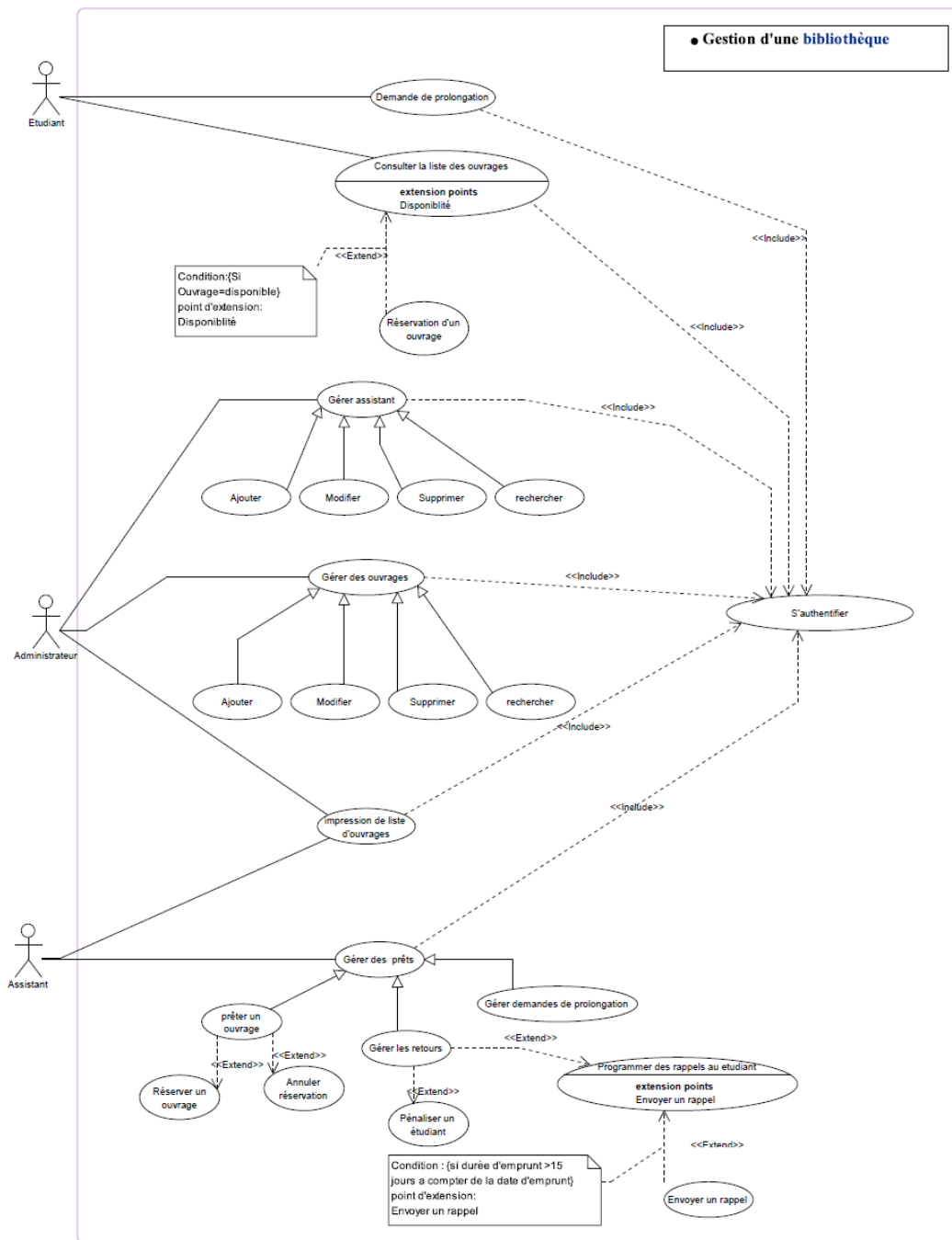


FIGURE 4 – Diagramme de cas d'utilisation général

## 2.4 Descriptions textuelles des cas d'utilisation

### 2.4.1 Authentification

- **Identification :**

- **Nom du cas :** Authentification.
- **But :** C'est l'action qui permet aux utilisateurs de se connecter afin qu'ils puissent avoir accès à leurs fonctionnalités.
- **Acteurs principales :**
  - Administrateur.
  - Assistant.
  - Étudiant.
- **Date :** 25/05/2021
- **Version :** 1.0

- **Séquencement :**

Le cas d'utilisation commence lorsque l'un des acteurs veut effectuer n'importe quelle opération.

- **Pre-condition :**

L'utilisateur doit avoir un compte.

- **Enchaînement nominal :**

1. Le système affiche le formulaire d'identification.
2. L'utilisateur saisie son Email et son mot de passe ainsi le type d'utilisateur.
3. Le système vérifie les informations saisies par l'utilisateur.
4. Le système donne accès aux fonctionnalités de l'utilisateur.

- **Enchaînement D'exception :**

- L'utilisateur n'a pas saisie les bons identifiants.
- Le système renvoie un message d'erreur et signale à l'utilisateur de recommencer.

- **Post-condition :**

L'utilisateur aura accès à son compte et ses fonctionnalités.

### 2.4.2 Demande de prolongation

- **Identification :**

- **Nom du cas :** Demande de prolongation.
- **But :** C'est l'action qui permet à l'étudiant de demander un prolongement de délai.
- **Acteurs principales :**
  - Étudiant.
- **Date :** 26/05/2021
- **Version :** 1.0

- **Séquencement :**

Le cas d'utilisation commence lorsque l'étudiant souhaite recevoir un prolongement de délai.

- **Pre-condition :**

L'étudiant doit avoir un compte.

- **Enchaînement nominal :**

1. L'étudiant saisit son Email et mot de passe.
2. Le système affiche l'interface adéquate.
3. L'étudiant clique sur le bouton "Demander prolongation".
4. L'étudiant remplit le formulaire et envoie la demande.

- **Enchaînement D'exception :**

- L'étudiant n'a pas saisi les bons identifiants.
- Le système renvoie un message d'erreur et signale à l'étudiant de recommencer.

- **Post-condition :**

L'étudiant envoie sa demande.



### 2.4.3 Consulter la liste des ouvrages

- **Identification :**

- **Nom du cas :** Consulter la liste des ouvrages.
- **But :** C'est l'action qui permet à l'étudiant de vérifier la disponibilité et consulter les ouvrages.
- **Acteurs principales :**
  - Étudiant.
- **Date :** 25/05/2021
- **Version :** 1.0

- **Séquencement :**

Le cas d'utilisation commence lorsque l'étudiant veut emprunter un ouvrage.

- **Pre-condition :**

L'étudiant doit avoir un compte.

- **Enchaînement nominal :**

1. L'étudiant saisit son Email et mot de passe.
2. Le système affiche l'interface adéquate.
3. L'étudiant choisit un mode de recherche.
4. Le système affiche les ouvrages disponible selon la recherche effectuée.
5. L'étudiant choisit un ouvrage.
6. L'étudiant a la possibilité de consulter les informations de cet ouvrage ou le réserver.
7. Dans le cas de réservation, l'étudiant insère ses coordonnées.
8. Le système valide le réservation.

- **Enchaînement D'exception :**

- L'étudiant n'a pas saisi les bons identifiants.
- Le système renvoie un message d'erreur et signale à l'étudiant de recommencer.

- **Post-condition :**

L'étudiant effectue la réservation.

#### 2.4.4 Gérer des assistants

- **Identification :**

- **Nom du cas :** Gérer des assistants.
- **But :** C'est l'action qui permet à l'administrateur d'ajouter, modifier ou supprimer un assistant.
- **Acteurs principales :**
  - Administrateur.
- **Date :** 26/05/2021
- **Version :** 1.0

- **Séquencement :**

Le cas d'utilisation commence lorsque l'administrateur veut gérer la liste des assistants (Ajout, modification, suppression).

- **Pre-condition :**

L'administrateur doit avoir un compte.

- **Enchaînement nominal :**

1. L'administrateur saisit son Email et mot de passe.
2. Le système affiche l'interface adéquate.
3. L'administrateur choisit l'opération à effectuer (Ajout, modification, suppression).

- **Enchaînement D'exception :**

- L'administrateur n'a pas saisi les bons identifiants.
- Le système renvoie un message d'erreur et signale à l'étudiant de recommencer.

- **Post-condition :**

L'administrateur met à jour les données.

#### 2.4.5 Gérer des ouvrages

- **Identification :**

- **Nom du cas :** Gérer des ouvrages.
- **But :** C'est l'action qui permet à l'administrateur d'ajouter, modifier ou supprimer un ouvrage.
- **Acteurs principales :**
  - Administrateur.
- **Date :** 26/05/2021
- **Version :** 1.0

- **Séquencement :**

Le cas d'utilisation commence lorsque l'administrateur veut gérer la liste des ouvrages (Ajout, modification, suppression).

- **Pre-condition :**

L'administrateur doit avoir un compte.

- **Enchaînement nominal :**

1. L'administrateur saisit son Email et mot de passe.
2. Le système affiche l'interface adéquate.
3. L'administrateur choisit l'opération à effectuer (Ajout, modification, suppression).

- **Enchaînement D'exception :**

- L'administrateur n'a pas saisi les bons identifiants.
- Le système renvoie un message d'erreur et signale à l'étudiant de recommencer.

- **Post-condition :**

L'administrateur met à jour les données.

#### 2.4.6 Gérer des prêts

- **Identification :**

- **Nom du cas :** Gérer des prêts.
- **But :** C'est l'action qui permet à l'assistant de gérer les emprunts.
- **Acteurs principales :**
  - Assistant.
- **Date :** 26/05/2021
- **Version :** 1.0

- **Séquencement :**

Le cas d'utilisation commence lorsque l'assistant enregistre un emprunt ou valide une valide une prolongation.

- **Pre-condition :**

L'assistant doit avoir un compte.

- **Enchaînement nominal :**

1. L'assistant saisit son Email et mot de passe.
2. Le système affiche l'interface adéquate.
3. l'assistant choisit l'opération à effectuer.

- **Enchaînement D'exception :**

- L'assistant n'a pas saisi les bons identifiants.
- Le système renvoie un message d'erreur et signale à l'étudiant de recommencer.

- **Post-condition :**

L'assistant effectue ses fonctionnalités.

## 2.5 Diagramme de séquence

Les diagrammes de séquences sont la représentation graphique des interactions entre les acteurs et le système selon un ordre chronologique, il permet de montrer les interactions d'objets dans le cadre d'un scénario d'un Diagramme des cas d'utilisation. Dans un souci de simplification, on représente l'acteur principal à gauche du diagramme, et les acteurs secondaires éventuels à droite du système. Le but étant de décrire comment se déroulent les actions entre les acteurs ou objets.

### 2.5.1 Diagramme de séquence "Authentification"

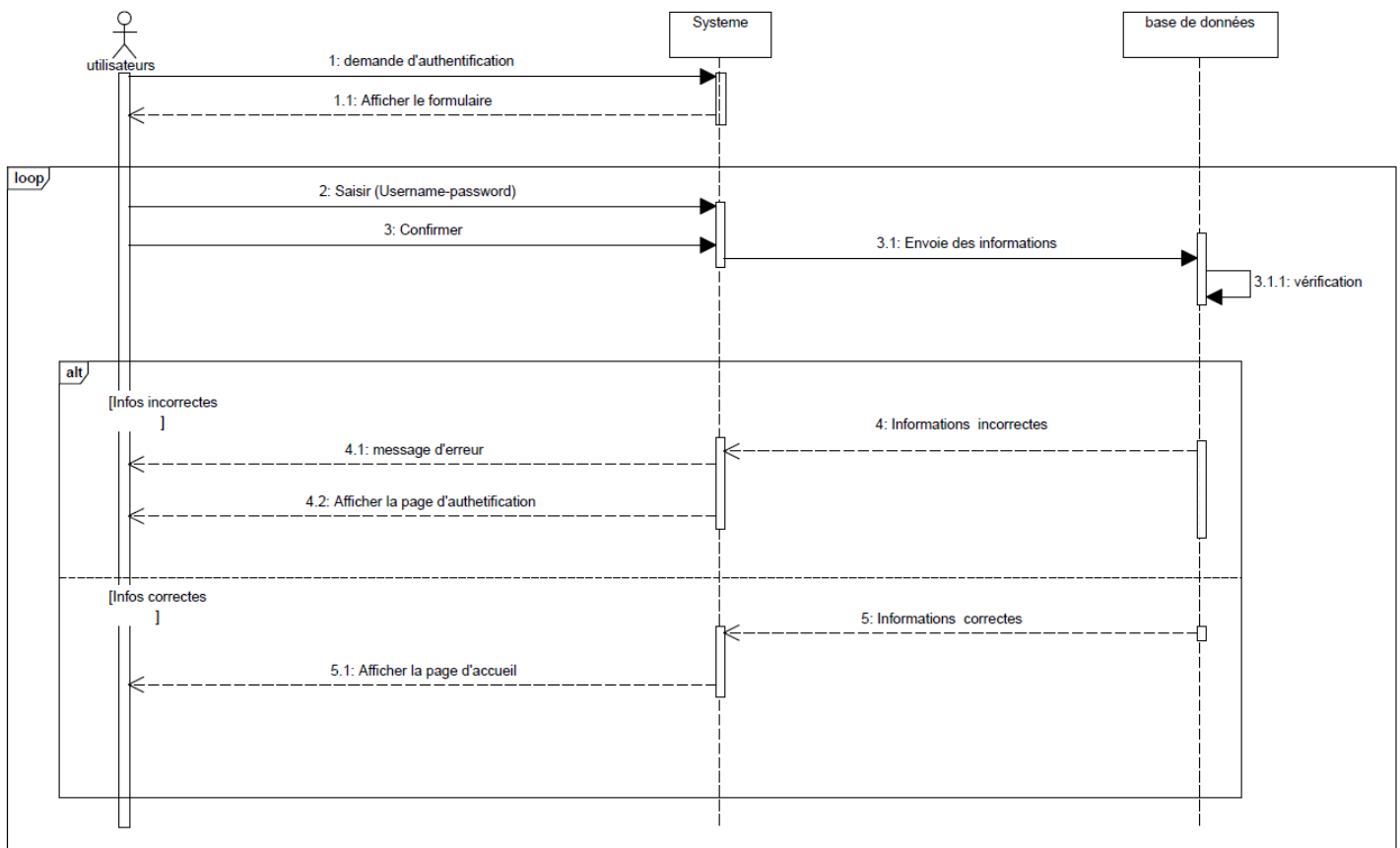


FIGURE 5 – Diagramme de séquence "Authentification"

## 2.5.2 Diagramme de séquence "Réservation d'un ouvrage"

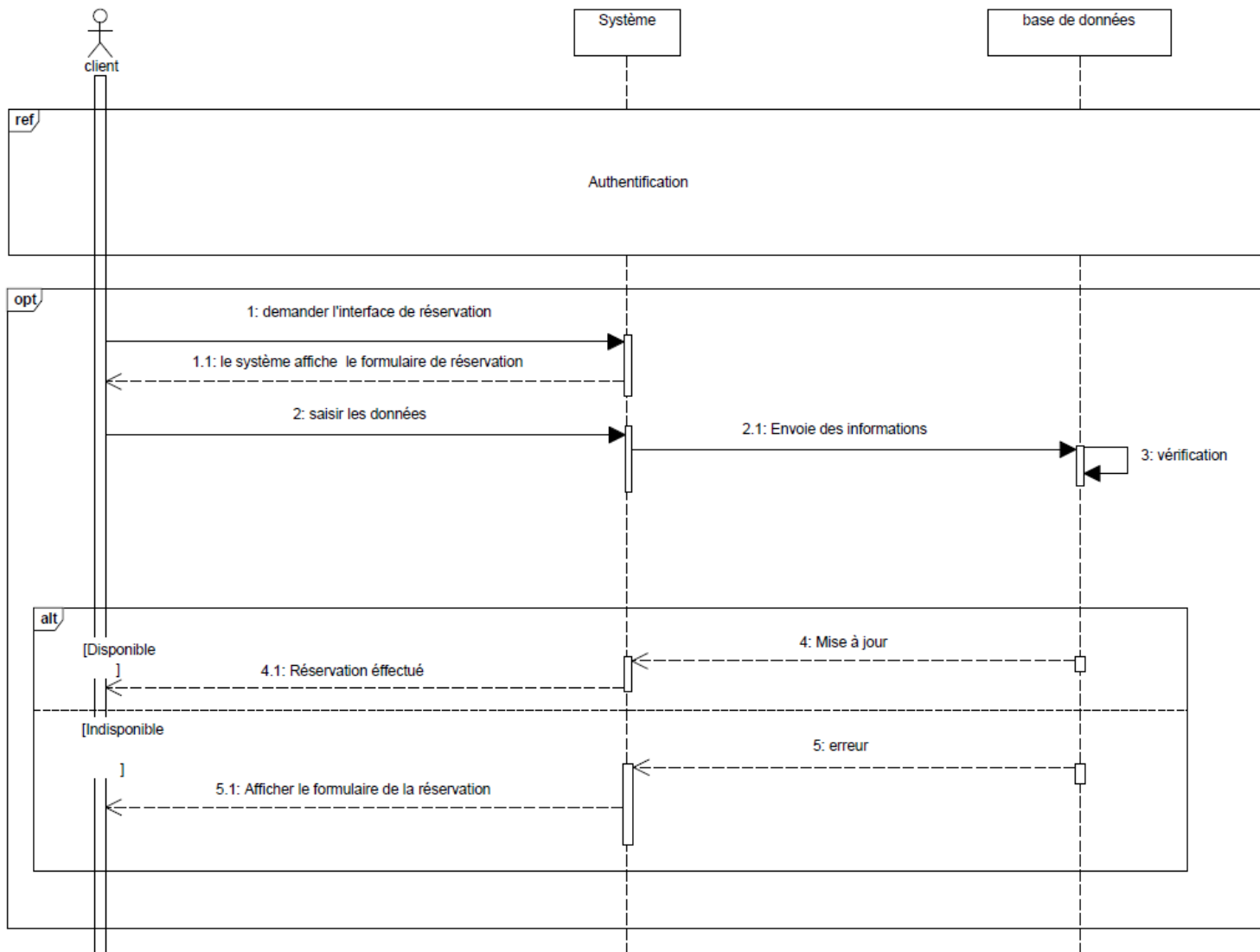


FIGURE 6 – Diagramme de séquence "Réservation d'un ouvrage"

## 2.5.3 Diagramme de séquence "Mise à jour"

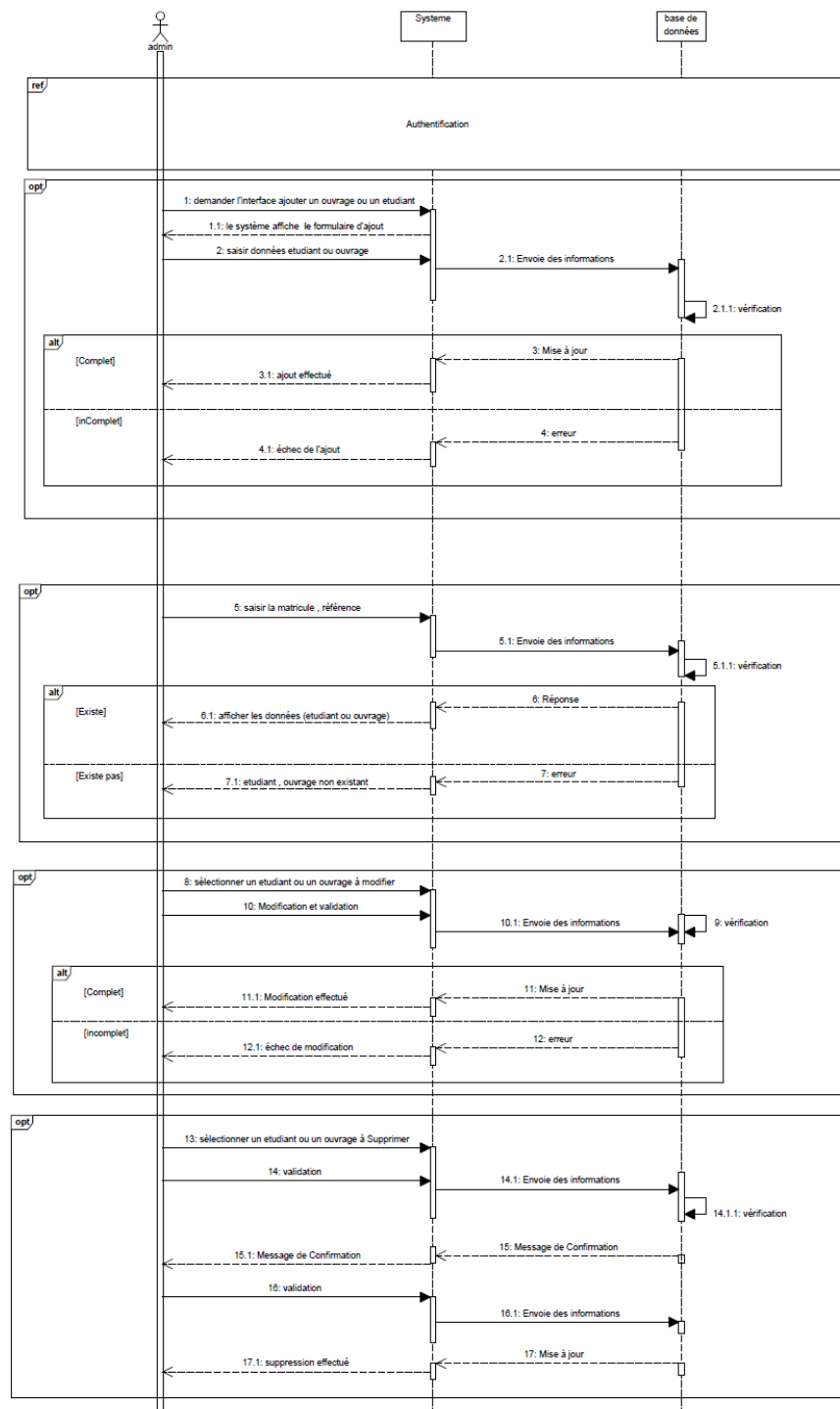


FIGURE 7 – Diagramme de séquence "Mise à jour"

## 2.6 Diagramme de classe

Le diagramme de classe est un schéma utilisé en génie logiciel pour présenter les classes et les interfaces des systèmes, ainsi que les différentes relations entre celles-ci. L'intérêt du diagramme de classe est de modéliser les entités du système d'information. Le diagramme de classe permet de représenter l'ensemble des informations finalisées qui sont gérées par le domaine. Ces informations sont structurées, c'est à-dire qu'elles ont regroupées dans des classes. Le diagramme met en évidence d'éventuelles relations entre ces classes.

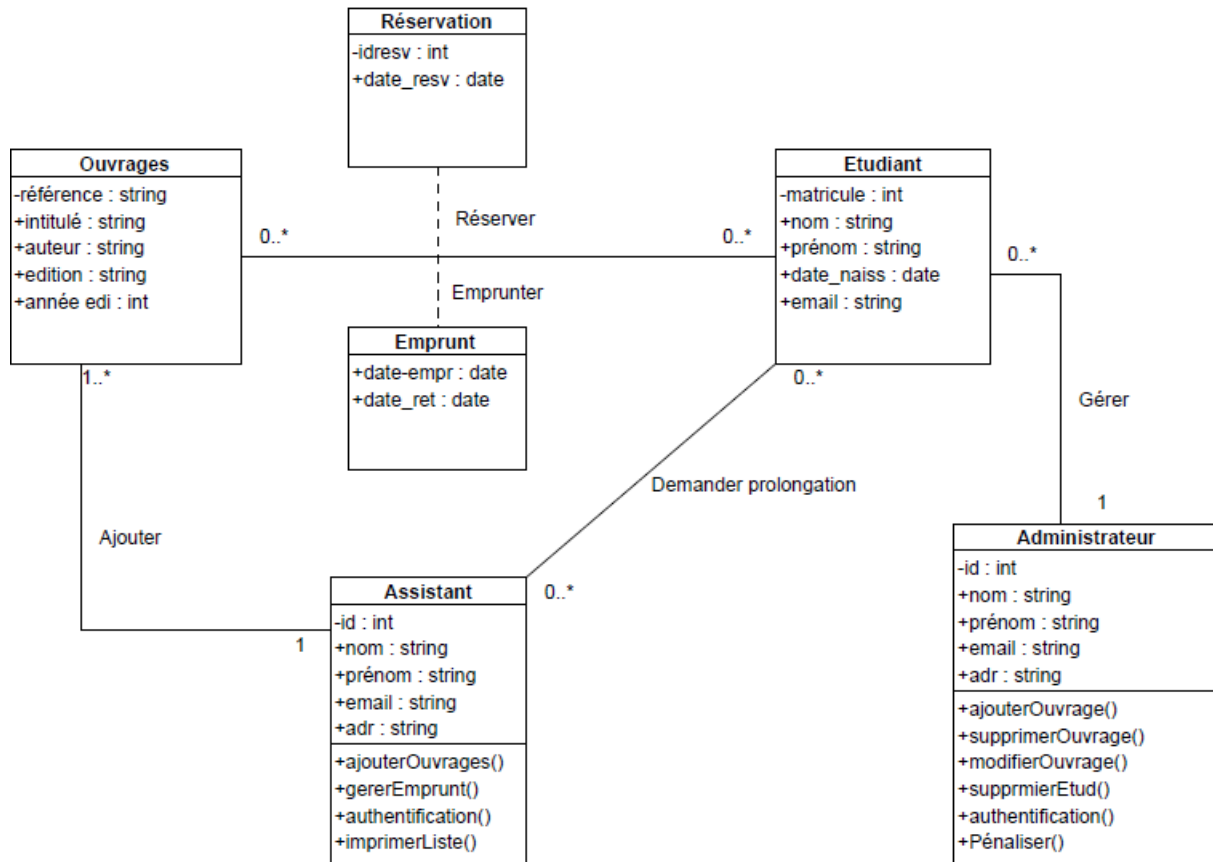


FIGURE 8 – Diagramme de classe



## 2.7 Modèle relationnel

A partir du diagramme de classes de notre application, on a pu obtenir le modèle relationnel tout en appliquant les règles de passage suivantes :

### Règle 1 :

- Obtention des tables avec leurs champs.
- Chaque classe du diagramme de classes devient une table (relation).
- Chaque attribut de la classe se transforme en un champ de table.
- Il faut choisir un champ qui peut jouer le rôle de la clé primaire et ce dernier sera souligné.

### Règle 2 :

- Présence de la cardinalité « \* » sur un côté de l'association.
- Chaque classe se transforme en une relation.
- Chaque attribut de classe se transforme en un champ de relation.
- L'identifiant de la classe qui est associée à la cardinalité « 1 » devient une clé étrangère dans l'autre classe.

### Règle 3 :

- Présence de la cardinalité « \* » sur les deux côtés.
- Chaque classe se transforme en une relation.
- Chaque attribut de classe se transforme en un champ de relation.
- L'association se transforme en une relation qui aura comme champs l'identifiant de chacune des deux classes (plus d'éventuels autres attributs).

### Règle 4 :

- Présence d'une généralisation (*méthode 1 : push-up*).
- Créer une relation avec tous les attributs des classes.
- Ajouter un attribut pour distinguer les types des objets.

### Règle 5 :

- Présence d'une généralisation (*méthode 2 : push-down*).
- Créer une relation pour chaque sous type.
- Chaque relation se compose des attributs génériques et des attributs spécifiques.

### Règle 6 :

- Présence d'une généralisation (*méthode 3 : distinction*).
- Créer une relation par classe et les relier par des associations : la clé primaire de la classe mère devient une clé étrangère dans les classes filles et en même temps sera considérée comme une clé primaire pour les deux classes filles.

Le modèle relationnel correspondant à notre base de données est le suivant :

**assistant**( id, nom, prenom, email, adr) ;

**administrateur**( id, nom, prenom, email, adr) ;

**ouvrages**( reference, intitule, auteur, edition, annee\_edi) ;

**etudiant**( matricule, nom, prenom, date\_naiss, email, #idadmin) ;

**reservation**( #matric, #ref, idresv, date\_resv) ;

**prolongation**( #matric, #idassistant, reference, nom, prenom, motif, date\_prolo) ;

**emprunt**( #matric, #ref, date\_empr, date\_ret) ;

Tout au long de ce chapitre, nous avons détaillé la conception de notre application à travers le diagramme de cas d'utilisation, diagrammes de séquence de quelques cas ainsi le diagramme de classe associés afin que la phase de réalisation et la mise en place de l'application soit plus souple et plus aisée. Le chapitre suivant mettra en évidence, le fruit de ce passage et les différents résultats du développement de l'application demandée.

---

# IMPLÉMENTATION

---

Pour pouvoir mener à bien un projet informatique, il est nécessaire de choisir des technologies permettant de simplifier sa réalisation. Pour cela, après avoir complété l'étude conceptuelle dans le chapitre précédent, nous allons aborder la partie implémentation dans ce qui suit. Nous commençons par présenter l'environnement matériel et logiciel, et ensuite, l'état de réalisation.

## 3.1 Environnement de développement :

### 3.1.1 IDE NetBeans

NetBeans est un environnement de développement intégré (EDI), placé en open source par Sun. Il constitue par ailleurs une plateforme qui permet le développement d'applications spécifiques (bibliothèque Swing (Java)). L'IDE NetBeans s'appuie sur cette plate forme.

### 3.1.2 Wamp

Une plateforme de développement Web, permettant de faire fonctionner localement des scripts PHP. WampServer n'est pas un logiciel, mais un environnement comprenant trois serveurs (Apache, MySQL et MariaDB), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.

### **3.1.3 Visual Paradigm for UML**

Logiciel de modélisation UML, cédé comme open source par son éditeur, à la fin de son exploitation commerciale, sous une licence modifiée de GNU GPL. StarUML gère la plupart des diagrammes spécifiés dans la norme UML 2.0.

### **3.1.4 Adobe Photoshop**

Logiciel de retouche, de traitement et de dessin assisté par ordinateur, Édité par Adobe, il est principalement utilisé pour le traitement des photographies numériques, mais sert également à la création ex nihilo d'images.

Il travaille essentiellement sur images matricielles car les images sont constituées d'une grille de points appelés pixels. L'intérêt de ces images est de reproduire des gradations subtiles de couleurs.

### **3.1.5 Adobe Illustrator**

Logiciel de création graphique vectorielle. Tout comme Photoshop, Illustrator fait partie de la gamme Adobe, peut être utilisé indépendamment ou en complément de Photoshop, et offre des outils de dessin vectoriel puissants. Les images vectorielles sont constituées de courbes générées par des formules mathématiques.

## 3.2 Présentation de l'application

### 3.2.1 Splash Screen

La toute première fenêtre affichée par un logiciel. Cette fenêtre incite l'utilisateur à patienter pendant le chargement et l'installation d'un logiciel tout en lui apportant diverses informations comme le nom du logiciel, le logo du logiciel, la version et l'état du chargement du logiciel.




FIGURE 9 – Splash Screen de l'application

### 3.2.2 Authentification

La fenêtre d'authentification permet à l'utilisateur d'accéder à ses fonctionnalités.

# Espace Authentification



*Veillez vous authentifier*

Identifiant :	<input type="text"/>
Mot de passe :	<input type="password"/>
Type :	<div>Administrateur</div>

[Se Connecter](#) [Annuler](#)

FIGURE 10 – Interface authentication

### 3.2.3 Interface Administrateur

Après avoir authentifié en tant qu'administrateur, cette fenêtre s'affiche contenant l'ensemble de fonctionnalités de l'administrateur.

[Coursages](#) | [Reservation d'Empunt](#) | [Ajouter](#)

# Gestion des ouvrages

---

## Formulaire

Référence :

Intitule :

Auteur :

Année :  2012

Edition :

Filière :  Architecture

Description :

Quantité :

Mise à jour

☒ Impression

### Liste des Ouvrages

Réf.	intitulé	Auteur	Edition	Année	Photo	Fath	Société	Crescent	quantité
CWTHG	TIGET	ZITZTIZI	VINNYAT	2021	com.m.	C.Lisa	Chisme	UNEAL	12
CWTHG	CHIMB	Laposte	2019	com.m.	C.Lisa	Inform.	Loisim	6	+
CWTHG	Elle Et	DUNKO	2015	com.m.	C.Lisa	Inform.	Loisim	6	-
MECP	Pendat	Talut	Loos	2019	com.m.	C.Lisa	Melico	Loisim	8
ACUFA	Aubin	William	DUNKO	2017	com.m.	C.Lisa	Aubin	Loisim	8
BFLA2	Agout	Corne	DUNKO	2016	com.m.	C.Lisa	Inform.	Loisim	55
BFLA2	Lois im	Orest	Laposte	2019	com.m.	C.Lisa	Inform.	Loisim	134
BFLA2	Agout	Richar	INTER	2009	com.m.	C.Lisa	Inform.	Loisim	9

FIGURE 11 – Interface administrateur

### 3.2.4 Interface Assistant

Après avoir authentifié en tant qu'assistant, cette fenêtre s'affiche contenant l'ensemble de fonctionnalités de l'assistant.

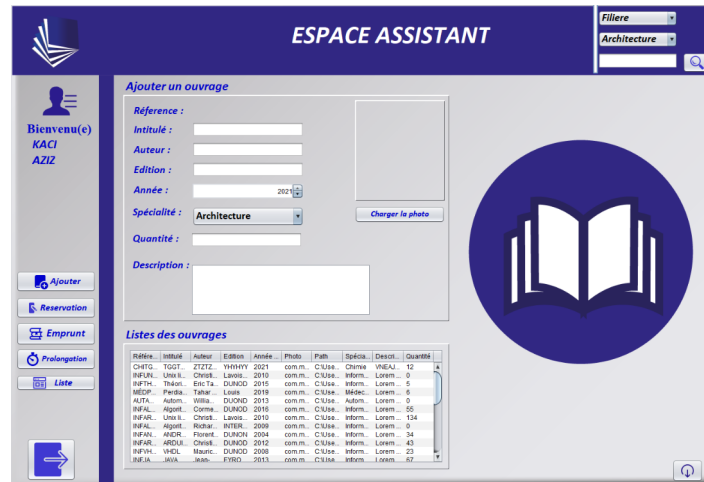


FIGURE 12 – Interface assistant

### 3.2.5 Interface Etudiant

Après avoir authentifié en tant qu'étudiant, cette fenêtre s'affiche contenant l'ensemble de fonctionnalités de l'étudiant.



FIGURE 13 – Interface étudiant

**N.B :** Une vidéo plus détaillé des fonctionnalités de l'application sera accompagné avec le rapport.

---

## CONCLUSION

---

Dans ce rapport, nous avons suivi une démarche bien déterminée pour résumer ce que nous avons réalisé durant la période de préparation du projet de fin d'étude en vue de l'obtention du diplôme de Licence en Informatique. En effet, nous avons commencé par un chapitre représentatif dont on a essayé de donner une idée sur le cadre de notre projet, puis nous avons entamé une étude conceptuelle, étant une phase de base dans toute application informatique. Ensuite nous avons abordé la partie implémentation ou nous avons présenté les différents outils et langages de développement utilisés, ainsi qu'une présentation de quelques interfaces de notre application.

Pendant la réalisation de notre travail nous avons acquis beaucoup de connaissances concernant le langage de modélisation UML et les différents langages utilisés.

Nous avons souhaité d'avoir plus de temps pour mieux traiter le sujet proposé, mais nous espérons que notre travail sera évolué et amélioré par autres promotion et qu'il sera un aide pour eux.



---

# RÉFÉRENCES

---

- [1] C.Delannoy (2020), Programmer en Java : Couvre Java 10 à Java 14. EYROLLES, 966 pages.
- [2] J.Gabay, D.Gabay(2008), UML 2 Analyse et Conception - Mise en Oeuvre. Dunod, 256 pages.
- [3] [https ://www.tutorialspoint.com/java/java\\_documentation.htm](https://www.tutorialspoint.com/java/java_documentation.htm)