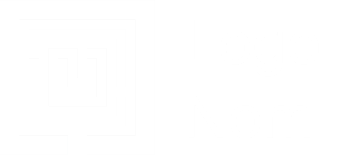


|  |
| --- |
| TITRE DU RAPPORT  2018 |
|  |
| 6 juillet  NOM DE LA SOCIÉTÉ  Créé par: Votre nom |



|  |
| --- |
| Dédicaces : A mes très chers parents, Aucun hommage ne pourrait étre a la hauteur de l’amour et de l’affectation dont ils ne cessent de me combler  Qu’ils trouvent dans ce travail un témoignage de mon profond amour et éternelle reconnaissance. Que dieu leur procure bonne santé et longue vie.  A mes chéres fréres Malek et Mahdi pour leur soutien durant toutes ces années. Une spéciale dédicace à ces personnes qui comptent déjà énormement pour moi, et pour qui je porte beaucoup de tendresse et d’amour. A tout mes amis, sans vos aide, vos conseils et vos encouragements ce travail n’aurait vu le jour. A tous mes amis, je dédie ce travail |
|  |
|  |

Remerciements :

J'exprime mes sincères reconnaissances à l'égard de tous ceux qui ont contribué à ma formation. J'adresse mes profonds remerciements à Madame Saoussen BEN JABRA, mon encadrante au sein de l'EPI, pour son encadrement, son soutien, sa disponibilité et ses conseils qui m'ont guidé tout au long de mon stage. Je tiens à remercier tout particulièrement et à témoigner toute ma reconnaissance à Mr Anwar MANSOURI pour l'expérience enrichissante et pleine d'intérêt qu'il m'a fait vivre durant la période du stage et pour tous les conseils et les informations qu'il m'a prodigué. Je tiens à remercier Madame Syrin Mrad mon tuteur de stage qui a dirigé mon travail, pour son engagement permanent, son soutien constant et la confiance totale qu'il m'a accordée.

Résumé :

Le présent rapport synthétise le travail effectué dans le cadre du projet de fin d’études pour l’obtention du diplôme national de licence en génie logiciel au sein de l’entreprise Com and Dev. Le travail consiste à développer une application de gestion documentaire dédiée aux professionnels de la santé et de la rééducation.

Introduction Générale :

L'entreprise opérant dans le secteur de développement logiciel devra concentrer sur la fiabilité et la robustesse de ses solutions tout en répondant aux besoins évolutifs de ses clients. Il est essentiel aussi de bien fonder ses choix technologiques et logiciels en étudiant les avantages et les limites de chacun en particulier le choix architectural de ses applications. En effet, l'architecture désigne la structure générale inhérente à un système informatique. À cet effet, l'équipe devra suivre les changements apportés qui ont pour but de développer des applications maintenables, extensibles et performantes. À cet égard, devant un développement exponentiel et continu de ses processus métier, Com and Dev fait face à des problèmes du fonctionnement tels que : la redondance des modules ainsi que les besoins des clients qui ne cessent pas d'augmenter. C'est dans ce cadre, que s'inscrit notre projet de fin d'études à l'École pluridisciplinaire internationale, réalisé à Com and Dev. Notre tâche consiste à concevoir et développer une nouvelle "Plateforme web de gestion documentaire dédiée aux professionnels de la santé et de la rééducation"

Par le biais de ce rapport, nous allons détailler les différentes étapes par lesquelles nous sommes passées pour réaliser ce projet. Le travail s’articule autour de quatre chapitres :

- Dans le premier chapitre, nous décrivons l'analyse de l'existant, la critique de l'existant, la solution proposée, les objectifs du projet et l'architecture de l'application.

- Dans le deuxième chapitre, nous étudions les besoins relatifs à notre applications qui identifie les acteurs, Ensuite, nous spécifions et raffinons les cas d'utilisations, présentons leurs diagrammes et nous établissons une liste des exigences fonctionnelles liées à notre application.

- Dans le troisième chapitre, nous présentons les diagrammes de séquences de chaque cas d'utilisation ainsi que le diagramme de classes de conception de l'application et enfin le diagramme de déploiement.

- Dans le dernier chapitre, nous détaillons notre outils utilisés pour développer l'application, ainsi que quelques captures d'écran pour mon projet .

Enfin, nous concluons notre travail par une conclusion générale en donnant des perspectives d'avenir pour notre application.

Chapitre1

Etude préalable :

* 1. Introduction

Dans ce chapitre nous commencerons par décrivez l'analyse et la critique de l'existant, qui nous a permis de cerner nos objectifs . Enfin, nous proposerons les différentes solutions aux problèmes soulevés.

* 1. Présentation de l’organisme d’accueil
  2. Etude préalable

L'étude préalable constitue une étape primordiale à la réalisation d'une application. En e et, elle permet d'analyser, d'évaluer et de critiquer le fonctionnement habituel, tout en élaborant la liste des solutions possibles.

* + 1. Analyse de l'existant

Plusieurs plateformes d'événements gèrent son travail à travers un site web qui n'offre pas suffisamment les fonctionnalités primordiales pour faciliter la communication entre chercheur et expert dans plusieurs domaines.

1.2.2 Critique de l'existant

Lors de l'étude que nous avons faite, nous avons constaté les fonctionnalités suivantes : Pour déposer un nouveau papier le chercheur doit se déplacer sur place. Pour consulter son résultat le chercheur doit parcourir tous les résultats. Pour noter les papiers il n' y a pas un système intelligent pour faciliter l'évaluation de papier.

* + 1. Solution proposée

A travers les problèmes élucidés, nous avons déduit que la solution est de développer une plateforme d'évènements qui offrent les fonctionnalités suivantes :

Faciliter au chercheur de soumettre son papier.

Permettre au chercheur de suivre son évaluation.

Il y a de la fiabilité grâce à la présence de l’expert.

1.2.4 Objectifs du projet

La plateforme permet aux chercheurs scientifiques de publier des papiers scientifiques décrivant leurs travaux de recherche et recevoir le résultat approuvé par les experts facilement à travers un système de notification .

* 1. Méthode et langage de modélisation

utilisés

pour la conception de notre système nous avons utilise UML (Unified Modeling Langage) pour bien modéliser notre système.

* + 1. Langage de modélisation choisi

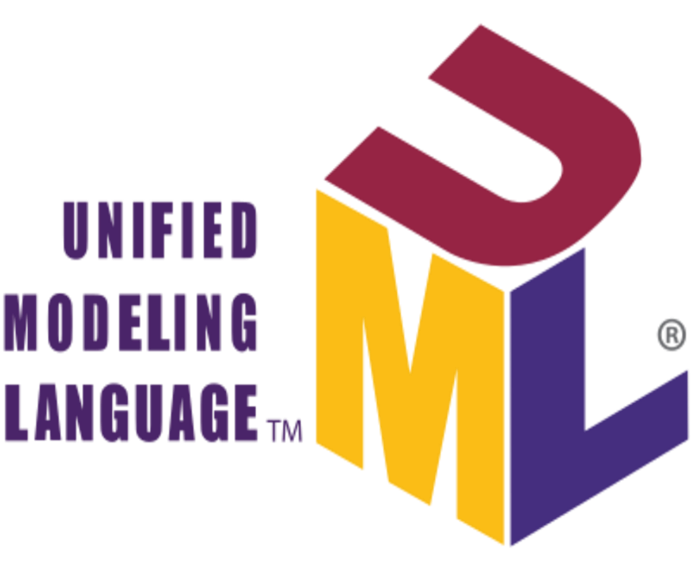


Figure 1.1 \_ Logo d’UML

Le langage UML (Unified Modeling Language, ou langage de modélisation unifié) a été pensé pour être un langage de modélisation visuelle commun, et riche sémantiquement et syntaxiquement. Il est destiné à l'architecture, la conception et la mise en œuvre de systèmes logiciels complexes par leur structure aussi bien que leur comportement. L'UML a des applications qui vont au-delà du développement logiciel, notamment pour les flux de processus dans l'industrie.

Il ressemble aux plans utilisés dans d'autres domaines et se compose de différents types de diagrammes. Dans l'ensemble, les diagrammes UML décrivent la limite, la structure et le comportement du système et des objets qui s'y trouvent.

L'UML n'est pas un langage de programmation, mais il existe des outils qui peuvent être utilisés pour générer du code en plusieurs langages à partir de diagrammes UML. L'UML a une relation directe avec l'analyse et la conception orientées objet.

1.4.2 Méthode de modélisation choisie

Il existe plusieurs méthodes de développement logiciel construites sur UML comme la méthode :UP, RUP, TTUP, UP agile, XP, 2TUP . . .. Parmi ses méthodes notre choix est basé sur la méthode UP (Unified Process),C'est un processus de développement logiciel itératif et incrémental , centré sur l'architecture, Piloté par des cas d'utilisation et orienté vers la diminution des risques .C'est un patron de processus pouvant être adaptée à une large classe de systèmes logiciels, à différents domaines d'application, à différents types d'entreprises, à différents niveaux de compétences et à différentes tailles de l'entreprise.

• Itératif et incrémental : Il se déroule en quatre phases : incubation, élaboration, construction et transition. Chaque phase répète un nombre de fois une série d'itérations. Chaque itération est composée de cinq activités à savoir la capture des besoins, l'analyse, la conception, l'implémentation et l'activité de test. Après chaque itération on a un nouvel incrément.

• Centré sur l'architecture : tout système complexe doit être décomposé en partie modulaire d'en faciliter la maintenance et l'évolution .Cette architecture (fonctionnelle, logique, matérielle, etc.) doit être modélisée en UML.

• Piloté par les cas d'utilisation : le projet est mené en tenant compte des besoins et des exigences des utilisateurs qui sont modélisés en cas d'utilisation décrits avec précision.

* 1. Architecture à envisager

Notre application est architecturée se base sur le principe client-serveur, nous adopterons donc l'architecture 3-tiers.

L'architecture 3-tiers est composée de trois éléments, ou plus précisément dans ce cadre là de trois couches.

1. La couche présentation associée au client qui de fait est dit "léger" dans la mesure où il n'assume aucune fonction de traitement à la différence du modèle Client / Serveur ou 2-tiers.

2. La couche fonctionnelle liée au serveur, comprend le serveur d'applications ou middleware ou encore serveur intermédiaire, qui dans de nombreux cas est un serveur Web muni d'extensions applicatives.

3. La couche de données liée au serveur de base de données (SGBD).

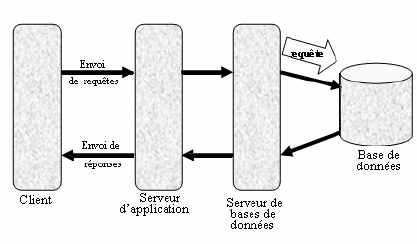


FIGURE 1.2 – Architecture 3\_tiers

Le principal avantage d'une architecture à trois niveaux est que, chaque niveau fonctionnant sur sa propre infrastructure, il peut être développé simultanément par une équipe de développement distincte. Il peut également être mis à jour ou dimensionné selon les besoins sans impact sur les autres niveaux.

1.6 Planification du projet

La répartition du temps est un parmi les plus efficaces des outils de développement des aptitudes personnelles en gestion et en performance du travail. C'est un ensemble de techniques claires et d'outils pratiques d'aménagement personnelle et de planification qui permet de diminuer le temps perdu au minimum et d'améliorer la performance personnelle très rapidement c'est pour cela , nous avons utilisé le diagramme de Gantt pour la planification de notre projet.

1.6.1 Diagramme de Gantt

Le diagramme de Gantt, couramment utilisé en gestion de projet, est l'un des outils les plus efficaces pour représenter visuellement l'état d'avancement des différentes activités (tâches) qui constituent un projet. La colonne de gauche du diagramme énumère toutes les tâches à effectuer, tandis que la ligne d'en-tête représente les unités de temps les plus adaptées au projet (jours, semaines, mois etc.). Chaque tâche est matérialisée par une barre horizontale, dont la position et la longueur représentent la date de début, la durée et la date de fin. Ce diagramme permet donc de visualiser d'un seul coup d'œil :

* Les différentes tâches à envisager
* La date de début et la date de fin de chaque tâche
* La durée escomptée de chaque tâche
* Le chevauchement éventuel des tâches, et la durée de ce chevauchement
* La date de début et la date de fin du projet dans son ensemble

1.6.2 Présentation du diagramme de Gantt

Le diagramme suivant va représenter les taches principales à réaliser dans notre projet, nous avons commencé notre projet par la recherche, ensuite la spécification et l'analyse des besoins viennent les phases de conception et la réalisation de l'application. sLa rédaction du rapport prend cours tout au long du projet.

FIGURE 1.3 – Diagramme de Gantt

* 1. Conclusion

Dans ce chapitre nous avons présenté l'étude et la critique de l'existant, la méthode et la langage de modélisation ainsi que l'architecture d'application . Le chapitre suivant se chargera pour l'expression des besoins .

Chapitre2

Expression des besoins

2.1 Introduction

C'est la première phase du cycle de développement d'un logiciel. Dans ce chapitre, nous identifions d'abord les différents acteurs impliqués dans notre application, Ensuite nous présentons les besoins fonctionnels de notre application en s'appuyant sur les diagrammes de cas d'utilisation de la notation UML. Ensuite, nous spécifions et raffinons les cas d'utilisation. Enfin, nous présentons les besoins non fonctionnels .

2.2 Identification des acteurs

Un acteur est un élément externe, un homme, une machine ou un autre système, qui interagit avec le système pour avoir un résultat. Dans le cas de notre projet nous allons identifier les acteurs du système informatique en prenant compte de ceux qui toucheront de façon directe ce système. Les acteurs du métier qui sont impliqués dans notre système informatique sont :

**-Chercheur : Cet utilisateur a le droit de créer un compte, de soumettre un article, puis de recevoir son évaluation après traitement.**

-L'éditeur : Cet acteur est le responsable de gérer les experts, d'affecter les papiers aux experts et de transmettre l'avis final aux chercheurs.

-L'expert : Cet utilisateur est responsable d’évaluer les articles qui lui sont assignés et de transmettre ses rapports à l’éditeur.

2.3 Prototypes des interfaces

Pour élaborer rapidement et concrètement l'agencement de l'information et des fonctionnalités d'une future interface, le prototype est un modèle qui possède toutes les qualités techniques et toutes les caractéristiques de fonctionnement d'un nouveau projet, mais il s'agit aussi parfois d'un exemplaire incomplet (non définitif).

Nous avons présenter ci-dessous quelques prototypes d'interfaces liés à notre application :

2.3.1 Prototypes d’interfaces pour utilisateur

Ce prototype d'interface permet à l'utilisateur de s'inscrire :

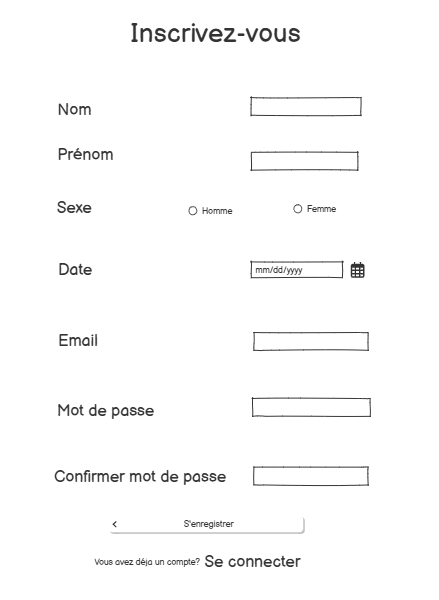


Figure 2.1 prototype d'interface << Inscrivez-vous>>

Une fois l'utilisateur a fini la création de son compte, ce prototype d'interface lui permet de se connecter soit le compte :



Figure 2.2 prototype d'interface <<Connectez-Vous>>

Ce prototype représente l'interface d’utilisateur dans notre application :

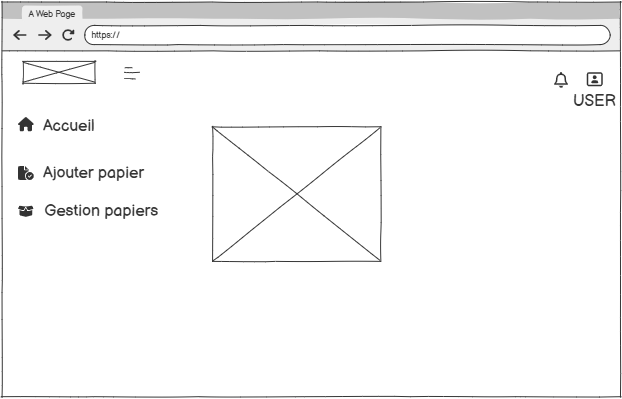


Figure 2.3 prototype d'interface << utilisateur >>

2.3.2 Prototypes d'interfaces pour entreprise

Ce prototype représente l'interface d’éditeur dans notre application :

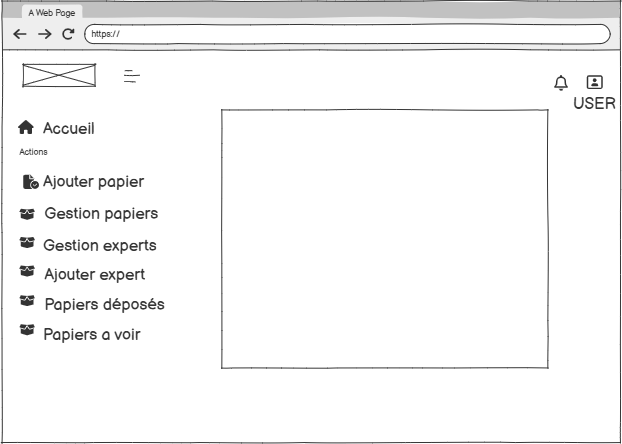


Figure 2.3 prototype d'interface <<éditeur>>

2.4 Identification des besoins fonctionnels

Les exigences fonctionnelles se réfèrent aux fonctions primaires que l'application doit remplir une fois développées. L'application doit fournir un ensemble spécifique de services qui sont :

Le chercheur doit pouvoir les fonctions suivantes :

-Authentification : le chercheur de notre application doit saisir son E-mail et son mot

de passe pour que le système s'assure de son existence avant de lui donner le droit d'accès.

Gestion des papiers : Notre application doit donner la possibilité d'ajouter, consulter

modifier et supprimer un papier.

-Soumettre papier : Notre application doit donner la possibilité au chercheur de soumettre un papier à l'éditeur.

- Revoir l'avis final : Notre application doit donner la possibilité au chercheur de revoir

l'avis final sur un papier .

L'éditeur doit pouvoir les fonctions suivantes :

-Authentification : l'éditeur de notre application doit saisir son E-mail et son mot de

passe pour que le système s'assure de son existence avant de lui donner le droit d'accès.

-Gestion des experts : Notre application doit donner la possibilité d'ajouter, consulter, modifier , supprimer un expert.

-Affecter papiers aux experts : Notre application doit donner la possibilité à l'éditeur

d'affecter un papier à l'expert .

- Transmettre l'avis final : Notre application doit donner la possibilité à l'expert de transmettre l'avis final au chercheur .

L'expert doit pouvoir les fonctions suivantes :

- Authentification : l'éditeur de notre application doit saisir son E-mail et son mot de

passe pour que le système s'assure de son existence avant de lui donner le droit d'accès.

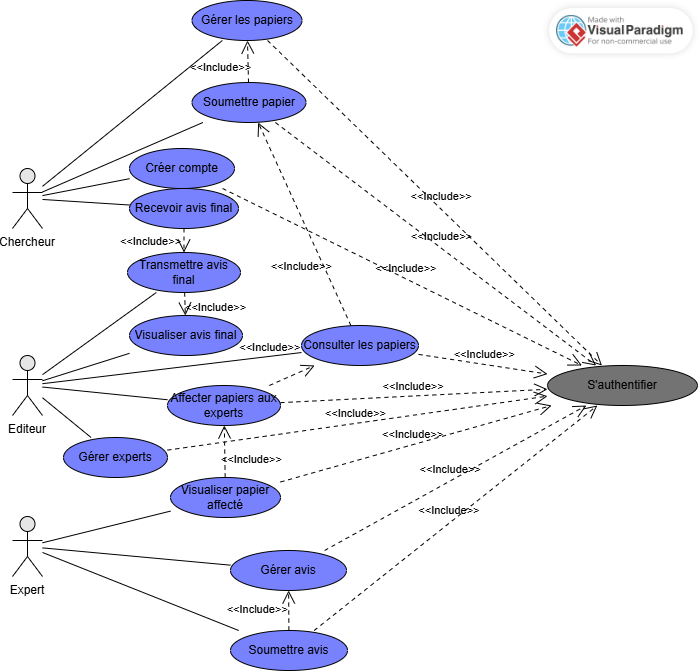
- Gestion des avis : Notre application doit donner la possibilité d'ajouter , consulter,

modifier et supprimer un avis .

2.5 Diagramme de cas d'utilisation

En langage UML, les diagrammes de cas d'utilisation modélisent le comportement d'un système et permettent de capturer les exigences du système. Ils décrivent les fonctions générales et la portée d'un système. Ces diagrammes identifient également les interactions entre le système et ses acteurs.

2.5.1 Diagramme de cas d'utilisation global

Le diagramme de cas d'utilisation global est le diagramme général du système qu'on modélise et qui d'écrit les interactions, les fonctionnalités et les relations globales entre les différents acteurs. 

2.4 Diagramme de cas d'utilisation global

2.6 Raffinement des Cas d'Utilisation Généraux par les Cas d'Utilisation Précis

2.6.1 Cas d'utilisation <<Gérer papiers>>

Le chercheur a la possibilité d’ajouter, de modifier ou de supprimer des papiers. Pour cela, la tâche est modélisée par le cas d’utilisation Gérer papiers, qui est en relation d’extension avec les cas Ajouter un papier, Modifier un papier et Supprimer un papier. Cependant, une fois qu’un papier est soumis par le chercheur, il ne peut plus être ni modifié ni supprimé.

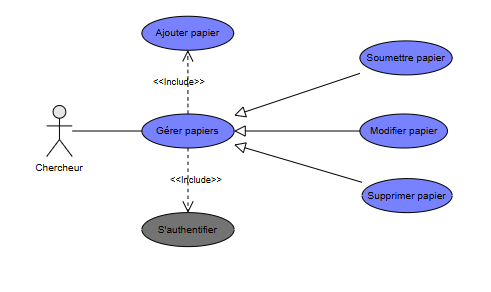


Figure 2.5 Cas d'utilisation <<Gérer papiers>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Ajouter un papier |
| Résumé : | Permet à un chercheur d’ajouter un nouveau papier scientifique dans le système. |
| Acteur principal : | Chercheur |
| Précondition : | Le chercheur doit être authentifié. |
| Postcondition : | Le papier est enregistré avec succès dans le système. |
| Scénario principal : | 1. Le chercheur accède à l’onglet ****"Ajouter papier"****.  2. Le système affiche le formulaire d’ajout.  3. Le chercheur remplit les informations du papier.  4. Il joint le fichier via l’option de téléversement.  5. Il clique sur ****"Enregistrer"****.  6. Le système enregistre le papier et confirme l’ajout. |
| Exceptions : | Si des données invalides sont saisies, le système affiche un message d’erreur. |
| Interface Homme-Machine : | Champs textuel : pour saisir les informations de papier.  ****Téléversement de fichier**** : pour joindre le papier.  ****Bouton "Enregistrer"**** : pour enregistrer le papier. |

Table…\_Cas d'utilisation << Ajouter papier>>

|  |  |
| --- | --- |
| ****Élément**** | Description |
| Cas d'utilisation : | |  | | --- | |  |  |  | | --- | | Modifier un papier | |
| Résumé : | Permet à un chercheur de modifier les informations d’un papier existant. |
| Acteur principal : | Chercheur |
| Précondition : | Le chercheur doit être authentifié |
| Postcondition : | Les informations du papier sont mises à jour avec succès dans le système. |
| Scénario principal : | 1. Le chercheur clique sur l’onglet ****"Gestion papiers"****.  2. Il choisit un papier et clique sur ****"Modifier"****.  3. Le système affiche le formulaire de modification.  4. Le chercheur modifie les informations souhaitées.  5. Il clique sur ****"Modifier"****.  6. Le système enregistre les nouvelles données du papier. |
| Exception : | Un message d'erreur est affiché si l'éditeur saisit des données erronées. |
| Interface Homme-Machine : | - ****Champs textuels**** : pour modifier les informations du papier. - ****Bouton "Modifier"**** : pour valider les modifications. |

Table…\_Cas d'utilisation << Modifier papier>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Supprimer un papier |
| Résumé : | Permet à un chercheur de supprimer un papier existant dans le système. |
| Acteur principal : | Chercheur |
| Précondition : | Le chercheur doit être authentifié. |
| Postcondition : | Le papier est supprimé avec succès du système. |
| Scénario principal : | 1. Le chercheur accède à l’onglet ****"Gestion papiers"****.  2. Il choisit un papier et clique sur ****"Supprimer"****.  3. Le système traite la demande et supprime le papier sélectionné. |
| Exception : | Si le chercheur annule l’action, la suppression est interrompue. |
| Interface Homme-Machine : | **Bouton "Supprimer"** : pour confirmer la suppression.  **Message de confirmation** : indiquant que la suppression effectuée avec succès. |

Table…\_Cas d'utilisation << Supprimer papier>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Soumettre un papier |
| Résumé : | Permet à un chercheur de soumettre un papier pour évaluation. |
| Acteur principal : | Chercheur |
| Précondition : | Le chercheur doit être authentifié et le papier doit être déjà ajouté. |
| Postcondition : | Le papier est soumis avec succès et enregistré dans le système |
| Scénario principal : | 1. Le chercheur accède à l’onglet ****"Gestion papier"****.  2. Il choisit un papier et clique sur ****"Soumettre"****.  3. Le système traite la demande et marque le papier comme "soumis". |
| Exception : | Si des données manquantes ou incorrectes sont détectées, un message d’erreur s’affiche. |
| Interface Homme-Machine : | **Bouton "Soumettre"** : pour soumettre le papier.   **Message de confirmation** : indiquant que la soumission a été effectuée avec succès. |

Table…\_Cas d'utilisation << Soumettre papier>>

2.6.2 Cas d'utilisation <<Gérer expert>>

L'éditeur a la possibilité d'ajouter, de supprimer et de modifier des experts pour cela cette tâche modéliser par le cas d'utilisation <<Gérer expert>> est en relation d'extension avec ajouter, modifier, supprimer un expert.

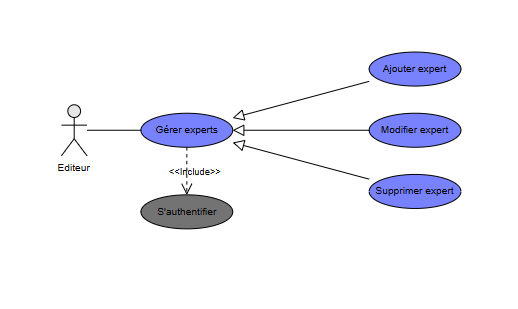


Figure 2.6 Cas d'utilisation <<Gérer expert >>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Ajouter un expert |
| Résumé : | Permet à un éditeur d’ajouter un nouvel expert dans le système. |
| Acteur principal : | Éditeur |
| Précondition : | L’éditeur doit être authentifié. |
| Postcondition : | L’expert est ajouté et enregistré dans le système. |
| Scénario principal : | |  | | --- | |  |  |  | | --- | |  |   1. L’éditeur accède à l’onglet ****"Ajouter expert"****.  2. Le système affiche le formulaire d’ajout.  3. L’éditeur saisit les informations de l’expert.  4. Il clique sur le bouton ****"Appliquer"****.  5. Le système enregistre les données et confirme l’ajout. |
| Exception : | Un message d'erreur est affiché si l'éditeur saisit des données erronées. |
| Interface Homme-Machine : | ****Champs textuels**** : pour saisir les données de l’expert.  ****Bouton "Appliquer"**** : pour valider l’ajout. |

Table …… Cas d'utilisation << Ajouter expert>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Modifier un expert |
| Résumé : | Permet à un éditeur de modifier les informations d’un expert existant. |
| Acteur principal : | Éditeur |
| Précondition : | L’éditeur doit être authentifié. |
| Postcondition : | Les informations de l’expert sont mises à jour avec succès dans le système. |
| Scénario principal : | 1. L’éditeur accède à l’onglet ****"Gestion experts"****.  2. Il choisit un expert et clique sur ****"Modifier"****.  3. Le système affiche le formulaire de modification.  4. L’éditeur modifie les informations de l’expert.  5. Il clique sur ****"Appliquer"****.  6. Le système enregistre les modifications et confirme la mise à jour. |
| Exception : | Un message d'erreur est affiché si l'éditeur saisit des données erronées. |
| Interface Homme-Machine : | ****Champs textuels**** : pour modifier les informations de l’expert .  ****Bouton "Appliquer"**** : pour valider les modifications. |

Table……..Cas d'utilisation <<Modifier expert>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Supprimer un expert |
| Résumé : | Permet à un éditeur de supprimer un expert du système. |
| Acteur principal : | Éditeur |
| Précondition : | L’éditeur doit être authentifié. |
| Postcondition : | L’expert est supprimé du système. |
| Scénario principal : | 1. L’éditeur accède à l’onglet ****"Gestion experts"****.  2. Il choisit un expert à supprimer et clique sur ****"Supprimer"****.  3. Le système traite la demande et supprime l’expert sélectionné. |
| Exception : | L'acteur annule la suppression. |
| Interface Homme-Machine : | **Bouton "Supprimer"** : pour confirmer la suppression. |

Table…… Cas d'utilisation << Supprimer expert >>

2.6.3 Cas d'utilisation <<Gérer avis>>

L’expert a la possibilité d’ajouter son avis. Cette fonctionnalité est modélisée par le cas d’utilisation *Gérer avis*, qui est en relation d’extension avec les cas *Ajouter un avis*, Cependant, une fois que l’expert a envoyer son avis, il ne peut plus le modifier ni le supprimer.

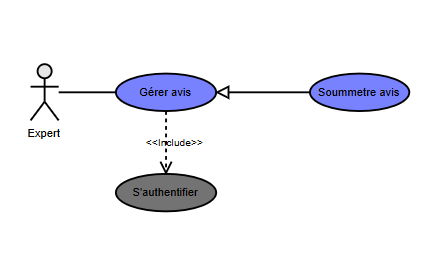


Figure 2.4 Cas d'utilisation <<Gérer avis>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Soumettre un avis |
| Résumé : | Permet à un expert de soumettre un avis concernant un papier. |
| Acteur principal : | Expert |
| Précondition : | L’expert doit être authentifié. |
| Postcondition : | L’avis est enregistré avec succès dans le système. |
| Scénario principal : | 1. L’expert accède à l’onglet ****"papiers à voir"****.  2. Il clique sur ****"Ajouter un avis"****.  3. Le système affiche le formulaire d’ajout d’avis.  4. L’expert saisit les informations requises.  5. Il clique sur ****"Envoyer"****.  6. L’avis est marqué comme "soumis" et n’est plus modifiable. |
| Exception : | Si des données sont erronées ou incomplètes, un message d’erreur s’affiche. |
| Interface Homme-Machine : | **Champs textuels** : pour saisir le contenu de l’avis   **Bouton "Envoyer"** : pour enregistrer l’avis. |

Table …… Cas d'utilisation <<Soumettre avis>>

2.6.3 Cas d'utilisation <<Gérer avis final>>

L’Editeur a la possibilité de modifier l’avis d’expert. Cette fonctionnalité est modélisée par le cas d’utilisation *Gérer avis final*, qui est en relation d’extension avec les cas *modifier un avis*, Cependant, une fois que l’Editeur a envoyer son avis, il ne peut plus le modifier ni le supprimer.

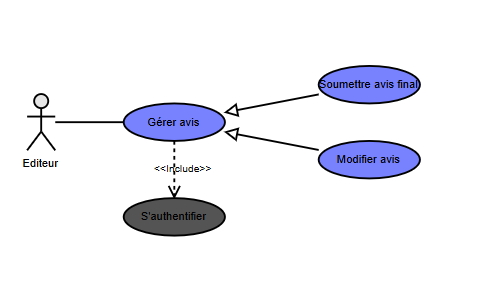


Figure 2.4 Cas d'utilisation <<Gérer avis final>>

|  |  |
| --- | --- |
| ****Élément**** | Description |
| Cas d'utilisation : | Modifier un avis |
| Résumé : | Permettre de modifier un avis. |
| Acteur principal : | Éditeur |
| Précondition : | L’éditeur doit être authentifié et l’avis doit être en état "non soumis". |
| Postcondition : | L’avis est mis à jour avec les nouvelles informations saisies. |
| Scénario principal : | 1. L’éditeur accède à l’onglet ****"Papiers à voir"****.  2. Il sélectionne un avis non soumis.  3. Il clique sur ****"Modifier"****.  4. Le système affiche le formulaire de modification.  5. L’éditeur modifie les données nécessaires.  6. Il clique sur ****"Modifier"****.  7. Le système met à jour l’avis. |
| Exception : | -Un message d'erreur est affiché si l'expert saisit des données erronées. |
| Interface Homme-Machine : | - ****Champs textuels**** : pour modifier le contenu de l’avis.  - ****Bouton "Modifier"**** : pour valider la modification. |

Table …… Cas d'utilisation <<Modifier avis final>>

|  |  |
| --- | --- |
| Élément | Description |
| Cas d'utilisation : | Soumettre avis final |
| Résumé : | Permet à l’éditeur de soumettre un avis d’expert. |
| Acteur principal : | Éditeur |
| Précondition : | L’éditeur doit être authentifié |
| Postcondition : | L’avis est marqué comme ****"soumis"**** |
| Scénario principal : | 1. L’éditeur accède à l’onglet ****"Papiers à voir"****.  2. Il sélectionne un avis non encore soumis.  3. Il clique sur ****"Envoyer"****.  4. Le système affiche une confirmation.  5. Le système marque l’avis comme soumis. |
| Exception : | Si les données de l’avis sont incomplètes ou invalides, un message d’erreur s’affiche. |
| Interface Homme-Machine : | ****Bouton "Envoyer"**** : pour lancer la soumission.   ****Message de succès**** après soumission. |

Table …… Cas d'utilisation <<Soumettre avis final>>

Conclusion

L'analyse et la spécification des besoins nous ont permis de déterminer les fonctionnalités principales de notre application à travers la présentation des besoins fonctionnels et non fonctionnels et en les détaillant avec les diagrammes des cas d'utilisation. Le chapitre suivant sera réservé à la conception de ces cas d'utilisation.

Chapitre3 :

Conception

Après avoir déterminé les besoins relatifs à notre application, nous passons à la conception de ses besoins, Ce chapitre se concentre sur la conception d'une structure appropriée pour l'application. Cette étape est cruciale car une bonne conception permet d'entreprendre et de préparer le terrain pour la phase de mise en œuvre nous présentons les diagrammes de séquence pour chaque cas d'utilisation.

0.0 Définition d'un diagramme de séquence

Le diagramme de séquence 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 systéme.

0.0 Diagrammes des séquences

0.0.0 Diagramme de séquence du cas <S'authentifier>

Le diagramme ci-dessous illustre le processus de connexion d’un utilisateur à l’application. Le scénario se déroule comme suit :

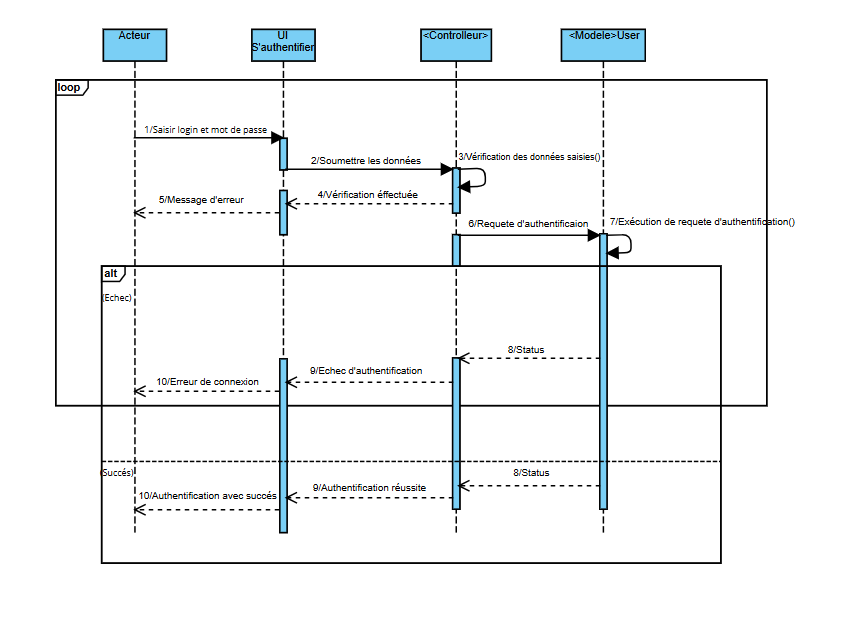
L’utilisateur initie la procédure d’authentification en cliquant sur le bouton « Se connecter » depuis l’interface d’accueil. En réponse, le système affiche l’interface d’authentification, où l’utilisateur saisit son identifiant (login) et son mot de passe.

Une fois les champs renseignés, une requête de vérification est envoyée au contrôleur pour valider les informations. À ce niveau, deux alternatives sont possibles, modélisées par une fragmentation ALT  dans le diagramme :

Si les identifiants sont valides, le contrôleur effectue une requête GET vers la base de données pour récupérer les informations de l’utilisateur. Une fois l’authentification réussie, le système redirige l’utilisateur vers l’interface d’accueil.

En revanche, si les identifiants sont invalides, un message d’erreur est retourné et l’authentification est refusée.

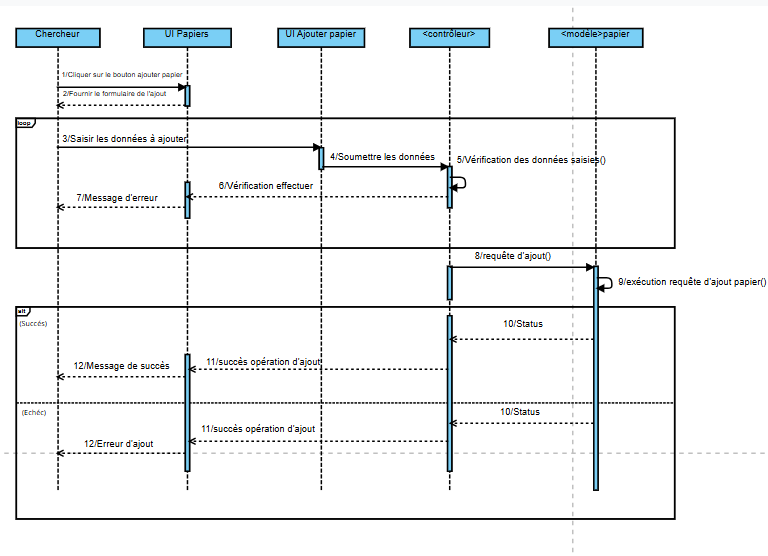
Ce diagramme de séquence permet de visualiser clairement les échanges entre l’utilisateur, le contrôleur, et la base de données dans le cadre du processus de connexion.



Figure….Diagramme de séquence du cas <S’authentifier>

0.0.0 Diagramme de séquence du cas <ajouter papier>

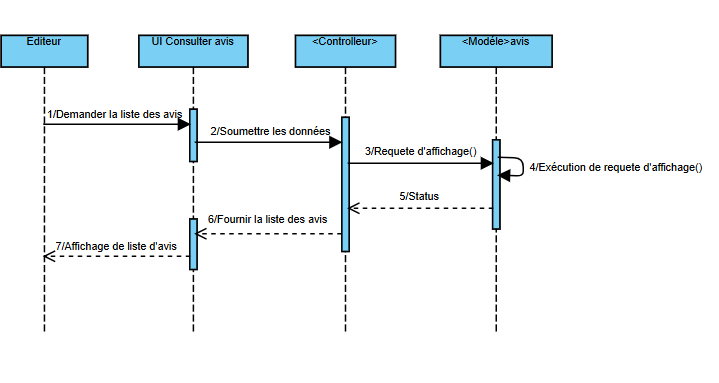
Le chercheur est autorisé d'ajouter un papier . Il clique sur le bouton ajouter papier . L'application offre l'interface de saisir des données au chercheur . Après la saisie des champs, une requête de vérification est renvoyée au contrôleur afin de vérifier si ces derniers sont valides. Deux cas ici se présentent, ajout validé ou décliné. Cela est modélisé dans le diagramme de séquence ci-dessus avec l'étiquette 'ALT'. si les champs sont valides, une requête 'Ajout Papier' d'ajout au niveau du contrôleur renvoie une requête ADD à la table `Papier' afin d'ajouter le nouveau papier. Le chercheur est par la suite redirigé vers l'interface de liste de papiers de notre application.



Figure……. Diagramme de séquence du cas <ajouter papier>

0.0.0 Diagramme de séquence du cas <ajouter avis>

Ci-dessous le diagramme de séquence qui simule l'action de consulter les avis.  
L’éditeur demande la liste des avis existants. Une requête de récupération est envoyée au contrôleur, qui interroge la table avis pour récupérer la liste. Ensuite, l'application fournit la liste à l’éditeur.



Figure….. Diagramme de séquence du cas <consulter avis>

Définition d'un diagramme de classes :

Les diagrammes de classes sont l'un des types de diagrammes UML les plus utiles, car ils décrivent clairement la structure d’un système particulier en modélisant ses classes, ses attributs, ses opérations et les relations entre ses objets. Avec notre logiciel de diagrammes UML, créer des diagrammes n’a jamais été aussi facile. Ce guide vous montrera comment comprendre, planifier et créer vos propres diagrammes de classes.

Diagramme de classes :

Dans notre système, plusieurs informations sont communes à toute personne physique. Afin d'assurer la flexibilité et la réutilisabilité du modèle, une classe abstraite nommée ****Utilisateur**** a été conçue pour regrouper ces données partagées. Trois types spécifiques d'utilisateurs en héritent : **Chercheur**, **Éditeur** et **Expert**.

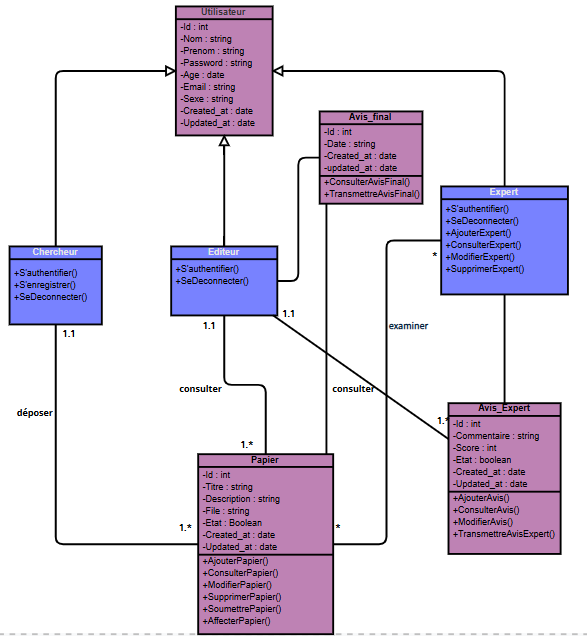
Le ****Chercheur**** et l’****Éditeur**** partagent plusieurs attributs, tandis que l’****Expert**** possède en plus un champ spécifique représentant son ****domaine de spécialisation****.

Une fois inscrit, le ****Chercheu**r** a la possibilité de ****soumettre des papiers****. Un papier est obligatoirement déposé par un seul chercheur, ce qui est modélisé par une relation de type ****1 à plusieurs**** (1..\*) entre la classe Chercheur et la classe Papier.

Les ****Experts**** peuvent ****consulter**** un ou plusieurs papiers, tout comme un papier peut être consulté par plusieurs experts ou par aucun. Cette interaction est représentée par une relation ****many-to-many**** entre les classes Expert et Papier. Cette relation est implémentée à travers une table associative nommée ****Avis****, qui référence les identifiants des deux entités et intègre des attributs ainsi que diverses fonctionnalités.

De leur côté, les ****Éditeurs**** peuvent consulter plusieurs papiers, mais chaque ****papier**** ne peut être associé qu’à un seul éditeur (ou à aucun). Cette relation est modélisée par une association appelée ****Avis** **final****, liant la classe Éditeur à la classe Papier. Cette classe associative contient les identifiants des deux entités, la ****date de soumission de** **l’avis** **final****, ainsi que d’autres paramètres et comportements métier spécifiques.

Le diagramme de classes ci-dessous illustre l'ensemble de cette modélisation orientée objet.



3.6 Conclusion

Dans ce chapitre, nous avons présenté les ****diagrammes de** **séquence**** correspondant à chaque cas d'utilisation, ainsi que le ****diagramme de classes****. Ces éléments ont permis de modéliser les interactions entre les différents acteurs du système et de structurer les entités métier de manière claire et cohérente.

Le chapitre suivant marquera l’****étape finale**** de notre projet, consacrée à la ****phase de** **réalisation****, où nous mettrons en œuvre les éléments spécifiés précédemment.