Application web pour la gestion d'entrepôt de documents

IV.1. Introduction

Afin d'appliquer l'approche dans [1] après son adaptation, nous avons recouru à la création d'un système de gestion d'entrepôt de documents multimédia. Nous cherchons essentiellement à travers ce système a déployer le modèle de représentation des documents multimédia, la démarche de classification, et surtout montrer que l'implantation du modèle permet d'exploiter les documents de façon simple et flexible.

Nous avons donc créé une application Web intitulée MERN-WMS pour gérer les entrepôts de documents multimédias. MERN est considérée de nos jours comme une moderne et populaire approche Web, elle signifie MongoDB, Express, React et NodeJs. WMS signifie Warehouse Management System.

Ce chapitre est considéré comme le dernier de notre mémoire, dans lequel nous démontrons les différents outils et technologies utilisées lors du développement de notre application, nous décrivons notamment les résultats des expérimentations réalisées avec cette application.

IV.2. Architecture de l'application MERN-WMS

L'application MERN-WMS permet d'optimiser l'intégration et l'exploitation des documents hétérogènes dans le cadre d'un entrepôt de documents multimédia, l'architecture cette application est présentée dans la figure suivante :

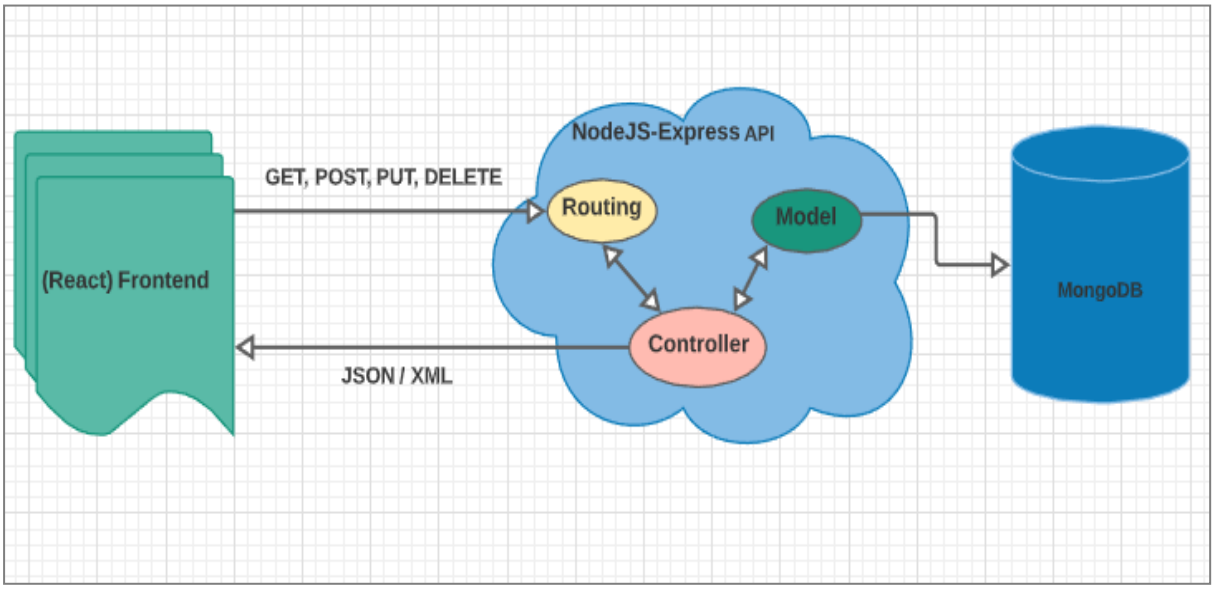

Figure IV.1. Architecture de l'application MERN-WMS

IV.3. Analyse et conception

IV.3.1. Présentation UML

UML est un langage de modélisation graphique conçu pour fournir une méthode normalisée pour visualiser la conception d'un système. C'est un support de communication performant qui permet grâce à sa représentation graphique, de concevoir des solutions, de faciliter la comparaison et de les évoluer. Il est couramment utilisé en développement logiciel et en conception orientée objet.

IV.3.2. Diagramme de cas d'utilisation

Il est utilisé pour donner une vision globale du comportement fonctionnel d'un système logiciel. Un cas d'utilisation représente une unité discrète d'interaction entre un utilisateur 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.

Dans l'application MERN-WMS, nous avons deux acteurs :

- **Admin** : il peut accéder à n'importe quelle partie du système, il peut ajouter, supprimer ou mettre à jour des utilisateurs ou des documents, il peut également modifier les autorisations des utilisateurs et mettre à niveau un utilisateur vers un administrateur.
- **User** : il ne peut pas accéder au système sans l'autorisation de l'administrateur et il ne peut que prévisualiser et télécharger des documents.

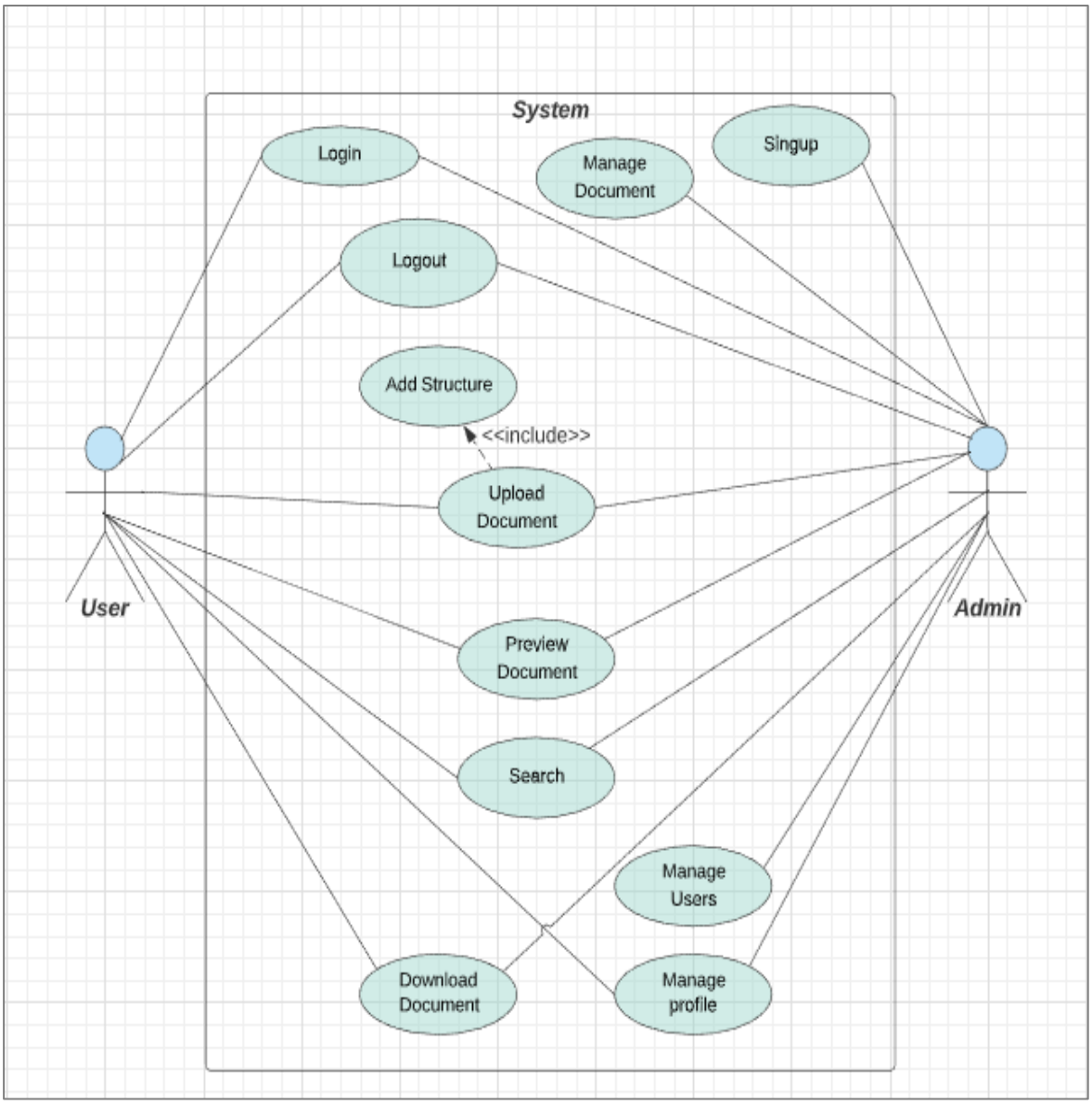

Figure IV.2. Diagramme de cas d'utilisation de l'application MERN-WMS

IV.3.3. Diagramme de séquence

Un diagramme de séquence est un diagramme d'interaction dont le but est de décrire comment les objets collaborent au cours du temps et quelles responsabilités ils assument. Il décrit un scénario d'un cas d'utilisation.

C'est un diagramme qui représente la structure dynamique d'un système car il utilise une représentation temporelle. Les objets, intervenant dans l'interaction, sont matérialisés par une « ligne de vie », et les messages échangés au cours du temps sont mentionnés sous une forme textuelle.

Dans le diagramme de séquence suivant, on a représenté seulement les deux principales opérations du système : 'Login' et 'Upload'.

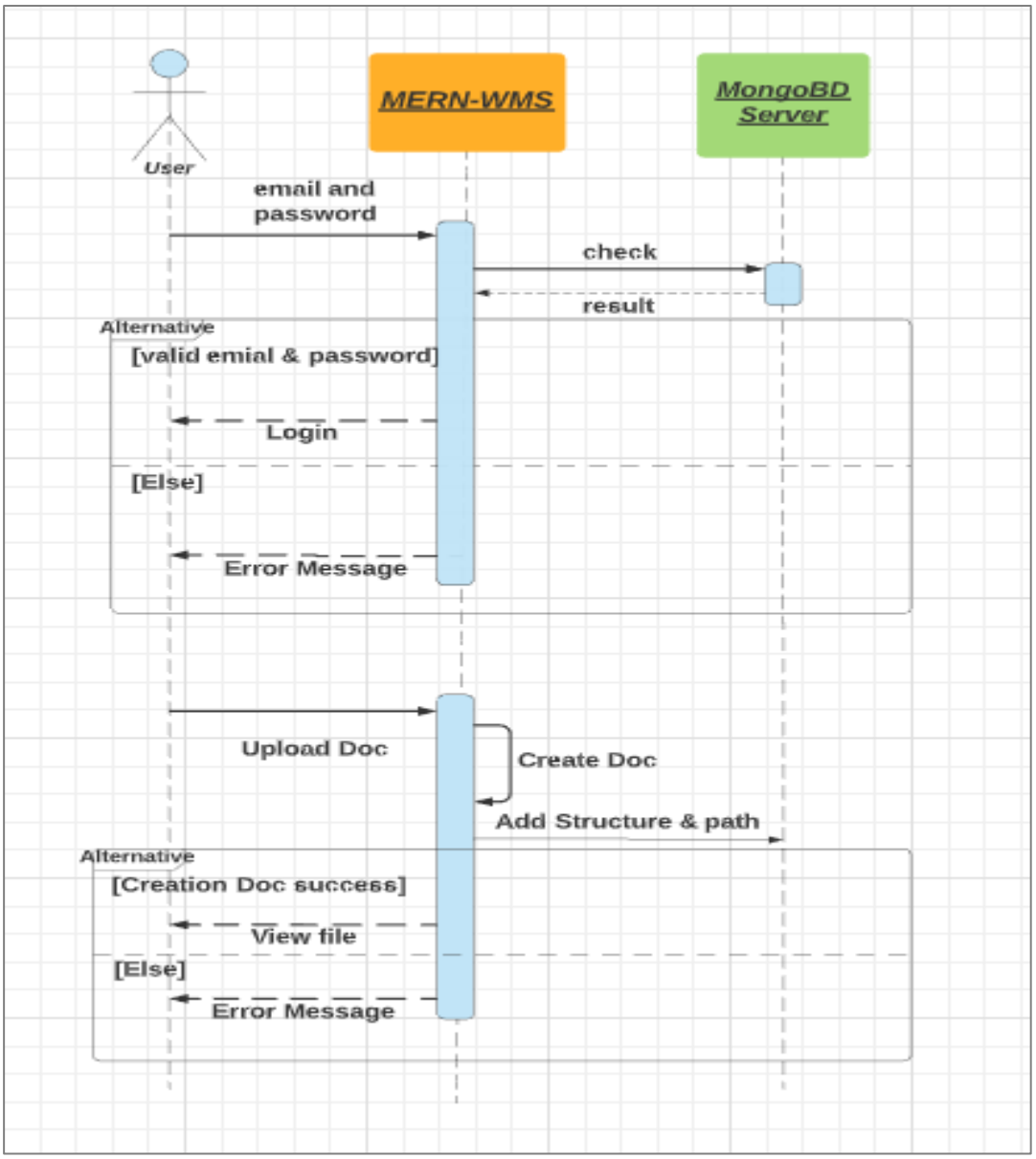

Figure IV.3. Diagramme de séquence des cas 'Login' et 'Upload'

IV.3.4. Diagramme de classe

Ce digramme permet de donner la représentation statique du système à développer. Cette représentation est centrée sur les concepts de classe et d'association. Chaque classe se décrit par les attributs et les opérations.

Un digramme de classe se définit comme un ensemble de classes contenant des attributs et des opérations, reliées les unes aux autres par des relations et ceci ayant des conditions de participation (cardinalités).

Une classe décrit un groupe d'objets ayant les mêmes propriétés (attributs), un même comportement (opérations) et une sémantique commune. Un attribut est une propriété élémentaire d'une classe qui prend une valeur. Une opération est une fonction applicable aux objets d'une classe. Elle est également appelée méthode.

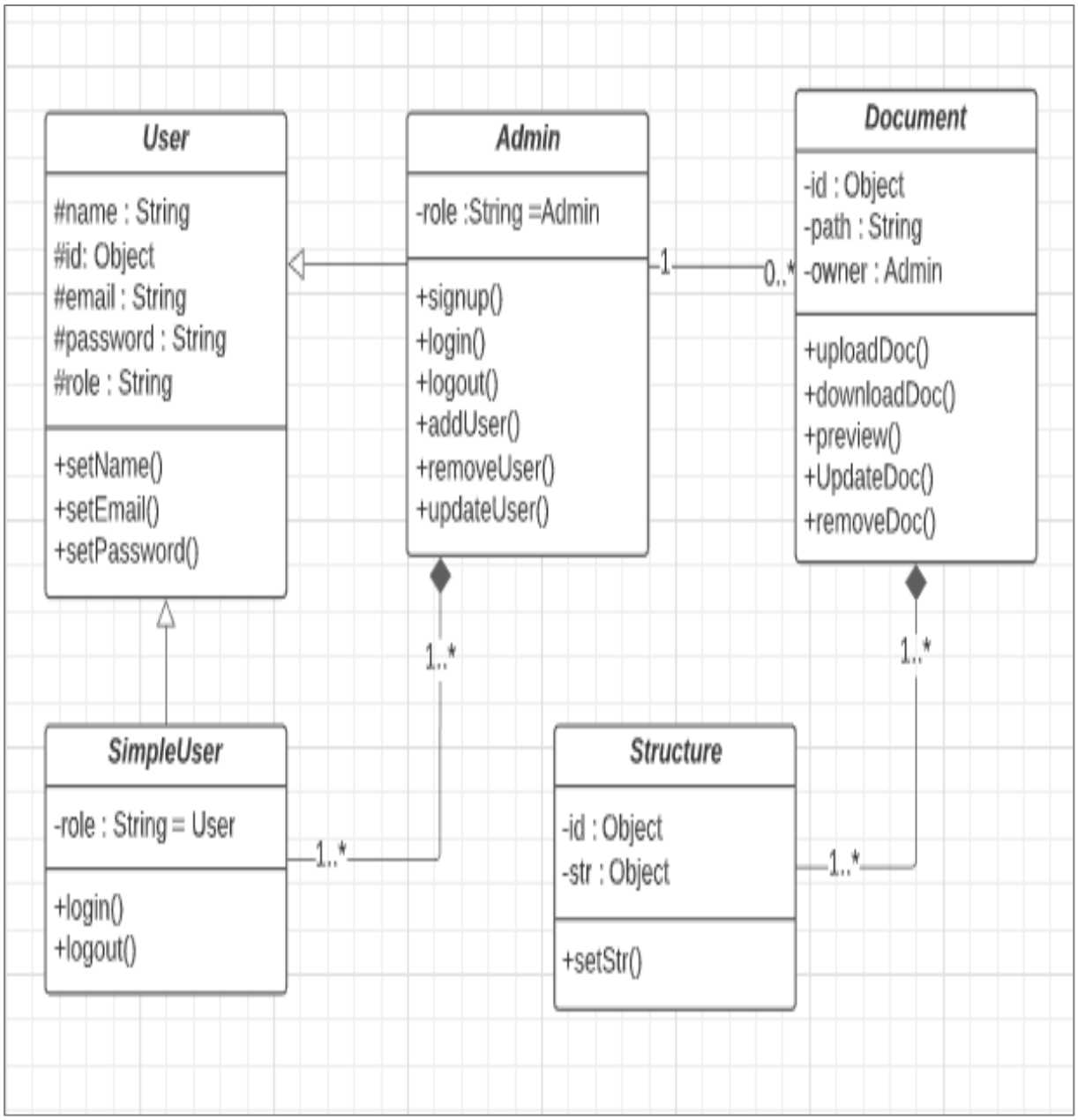

Figure IV.4. Diagramme de classe de l'application MERN-WMS

IV.4. Implémentation

IV.4.1 Environnement et outils de développement

Avant de commencer la présentation de notre application, nous allons tout d'abord, citer les outils utilisés lors du développement.

- **HTML**

Langage de marquage hypertexte (HyperText Markup Language). Il s'agit du langage de base du Web (World Wide Web).HTML n'est pas un langage de programmation proprement dit, mais plutôt un code de marquage. Il permet de décrire la page Web élément par élément en se servant de balises de description.

- **CSS**

CSS est un langage de style qui définit la présentation des documents HTML. Par exemple, CSS couvre les polices, les couleurs, les marges, les lignes, la hauteur, la largeur, les images d'arrière-plan, les positionnements évolués et bien d'autres choses.

- **JavaScript**

Javascript est un langage de script orienté objet principalement utilisé dans les pages HTML. A l'opposé des langages serveurs (qui s'exécutent sur le server), Javascript est exécuté sur l'ordinateur de l'internaute par le navigateur lui-même. Ainsi, ce langage permet une interaction avec l'utilisateur en fonction de ses actions.

- **Visual Studio Code**

Visual Studio Code est un éditeur léger mais puissant, disponible pour Windows, macOS et Linux. Il a un support intégré pour JavaScript, TypeScript et Node.js, et dispose d'un riche écosystème d'extensions pour d'autres langages tels que C ++, C #, Java, Python, PHP, Go, …, et des environnements tels que .NET et Unity.

- **Git**

Un logiciel de versioning, ou logiciel de gestion de version est un logiciel qui permet de conserver un historique des modifications effectuées sur un projet afin de pouvoir rapidement identifier les changements effectués et de revenir à une ancienne version en cas de problème.

- **GitHub**

GitHub est un service en ligne qui permet d'héberger des dépôts ou repo Git. C'est le plus grand hébergeur de dépôts Git du monde.

- **MongoDB Compass**

MongoDB Compass propose une interface graphique pour MongoDB afin de simplifier la gestion de la base de données NoSQL.

IV.4.2 Les technologies et Les Framework utilisées

- **Sass**

Sass est un langage de stylesheet compilé en CSS. Il nous permet d'utiliser des variables, des fonctions, etc., Sass aide à garder les stylesheets bien organisées et facilite le partage de la conception au sein et entre les projets.

- **NodeJS**

Node.js est un environnement d'exécution JavaScript qui exécute du code JavaScript en dehors d'un navigateur Web. Node.js permet aux développeurs d'utiliser JavaScript côté serveur pour produire des pages Web dynamiques.

- **Express**

Express.js est un Framework d'application Web pour Node.js. Il fournit diverses fonctionnalités qui rendent le développement d'applications Web rapide et facile, ce qui prend autrement plus de temps en utilisant uniquement NodeJS.

- **ReactJS**

Est une bibliothèque JavaScript pour créer des interfaces utilisateurs, il est facile de créer des interfaces utilisateurs interactives. Définissez des vues simples pour chaque état de votre

application, et lorsque vos données changeront, React mettra à jour, de façon optimale, juste les composants qui en auront besoin.

- **MongoDB**

MongoDB est une base de données orientée document multiplateforme et open source, une sorte de base de données NoSQL. En tant que base de données NoSQL, MongoDB évite la structure basée sur les tables de la base de données relationnelle pour adapter les documents de type JSON qui a des schémas dynamiques qu'il appelle BSON.

IV.5. Présentation de l'application

Dans cette section, nous présentons une expérimentation de notre application par l'utilisateur, comprenant la façon dont l'utilisateur peut accéder à l'application, et les opérations qu'il peut faire, …, etc.

IV.5.1. Page d'accueil

Elle représente la page principale et l'interface de notre application, elle expose aux utilisateurs les différentes opérations fournies par l'application. La figure IV.5 montre la page d'accueil dans le cas d'utilisation de l'application avec des ordinateurs, tandis que la figure IV.6 montre la page d'accueil dans cas d'utilisation avec des appareils mobiles.

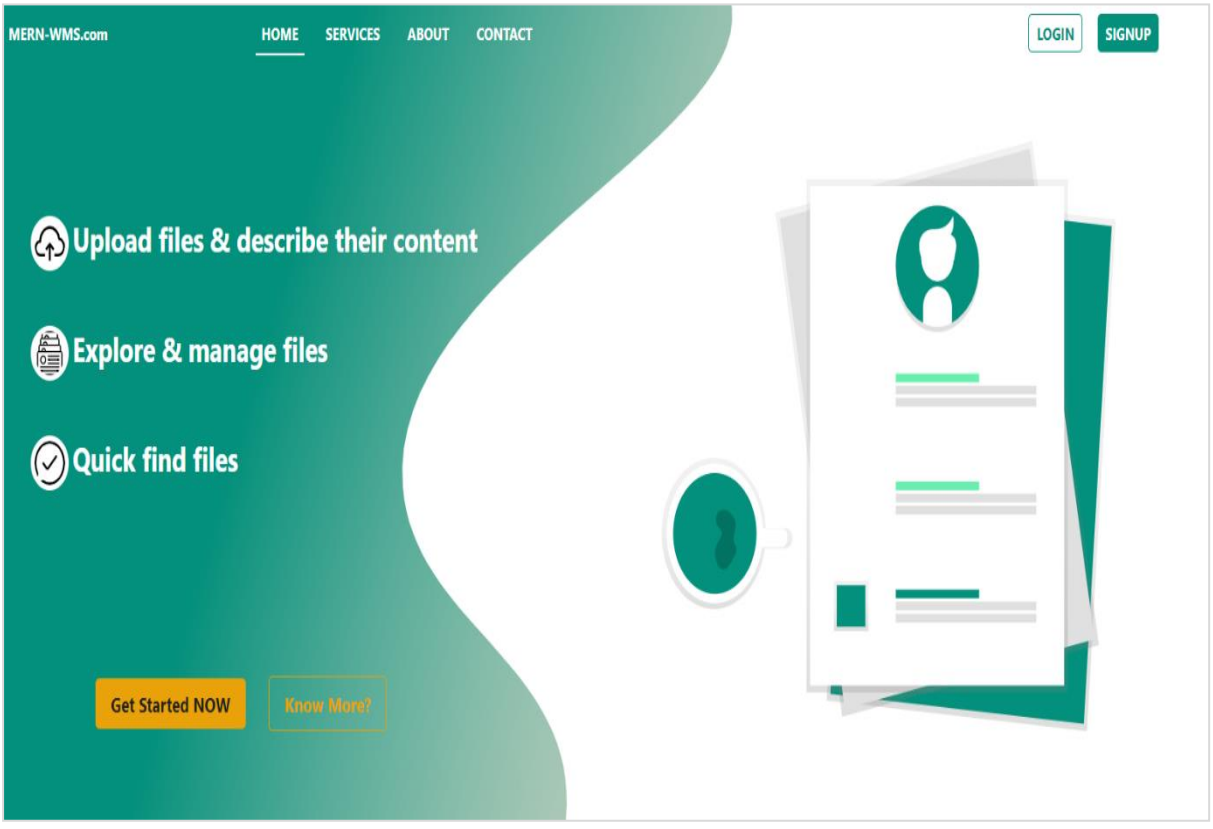

Figure IV.5. Page d'accueil dans les ordinateurs

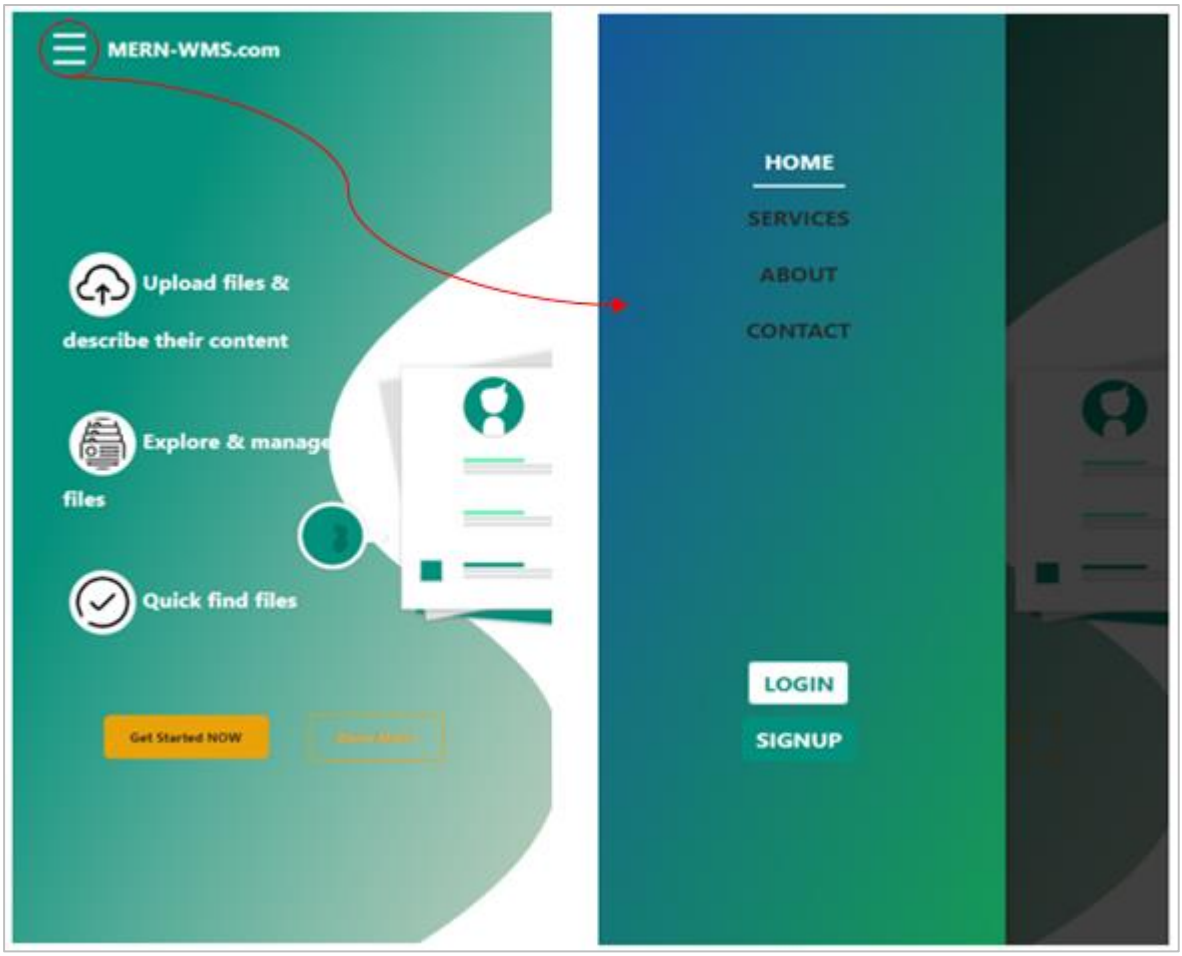

Figure IV.6. Page d'accueil dans les appareils mobiles

IV.5.2. Authentification d'utilisateur

Pour commencer à utiliser le système, l'utilisateur doit créer un compte avec son nom, son email et son mot de passe. L'utilisateur ne peut pas saisir de champs non valides, car l'application contient un système de validation.

La figure IV.7 montre la page d'inscription, tandis que la figure IV.8 affiche des erreurs que l'utilisateur pourrait obtenir lors du 'Login' (e-mail existant, entrées invalides, ..., etc.).

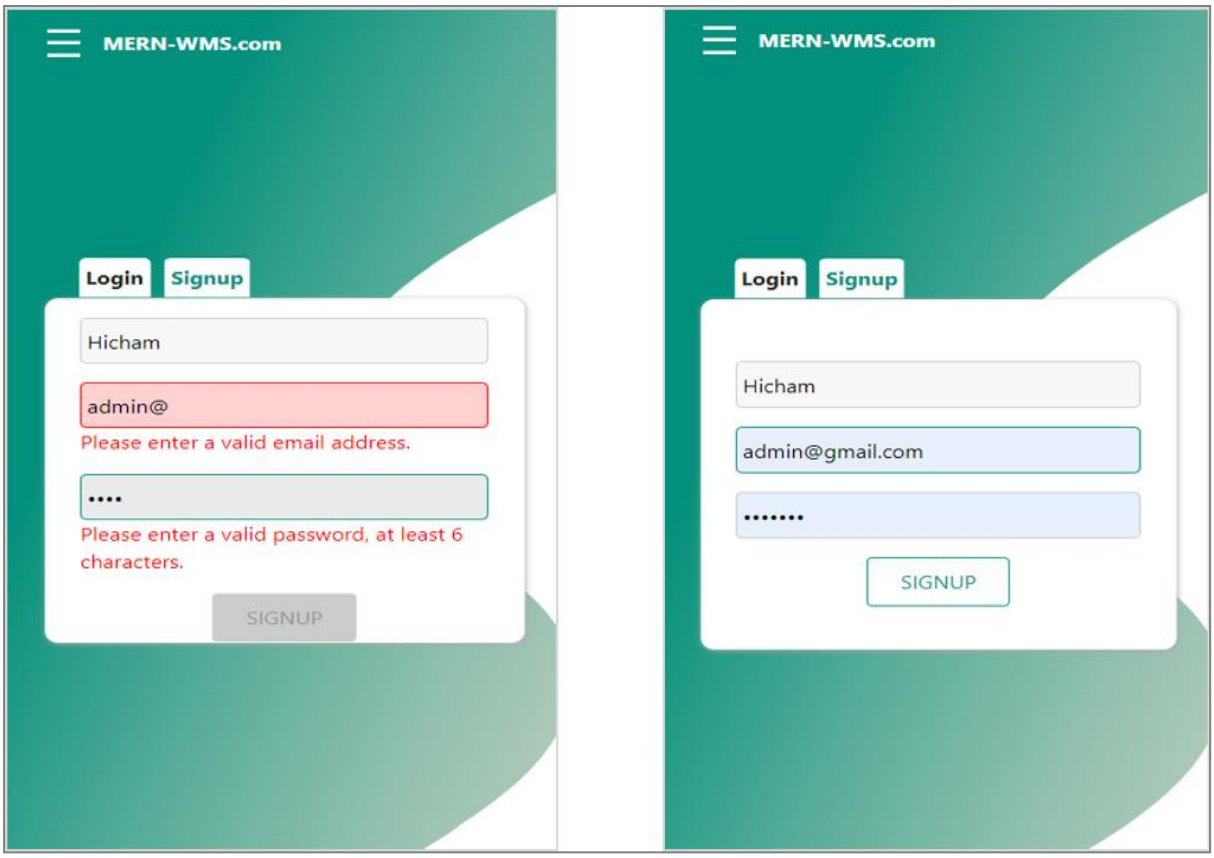

Figure IV.7. Page d'inscription

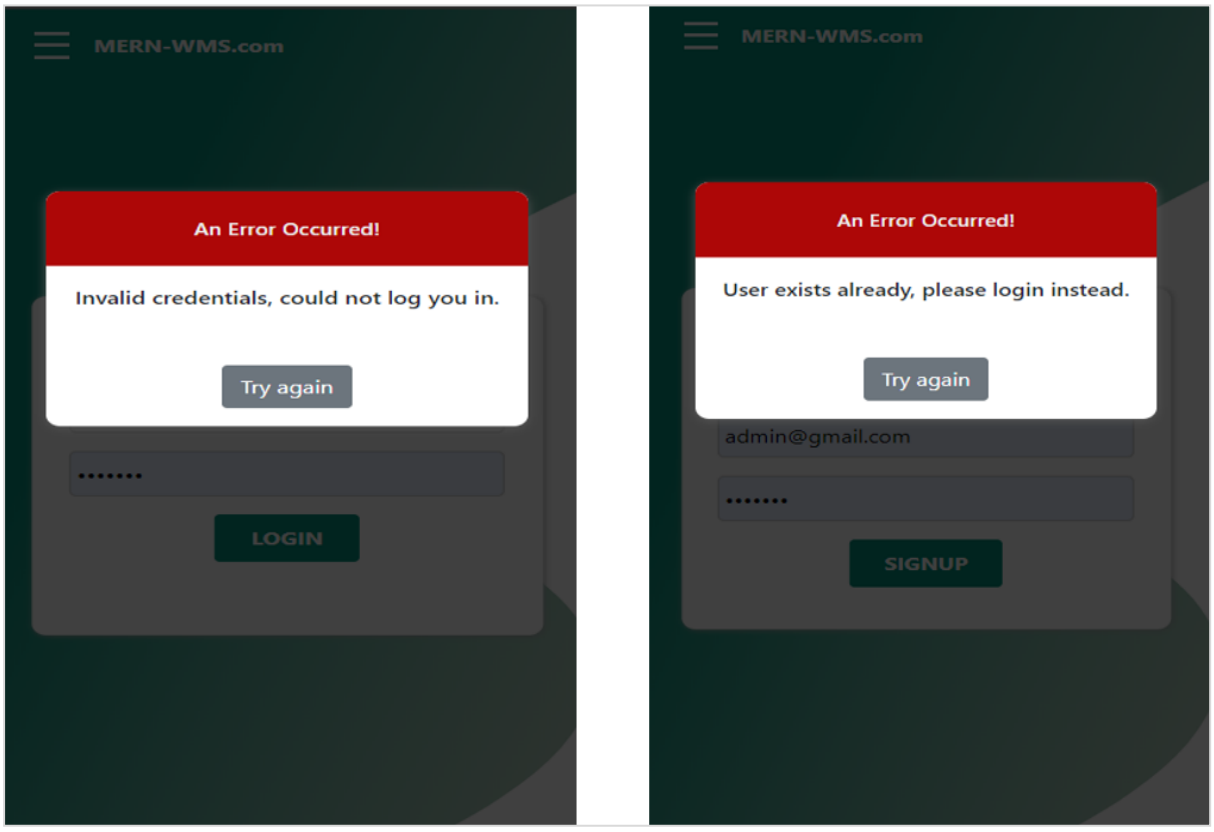

Figure IV.8. Quelques erreurs gérées

IV.5.3. Ajout de document et leurs structures

Lorsque l'utilisateur crée un compte avec succès, il sera dirigé vers la page suivante (figure IV.9), où il peut ajouter des documents et les gérer.

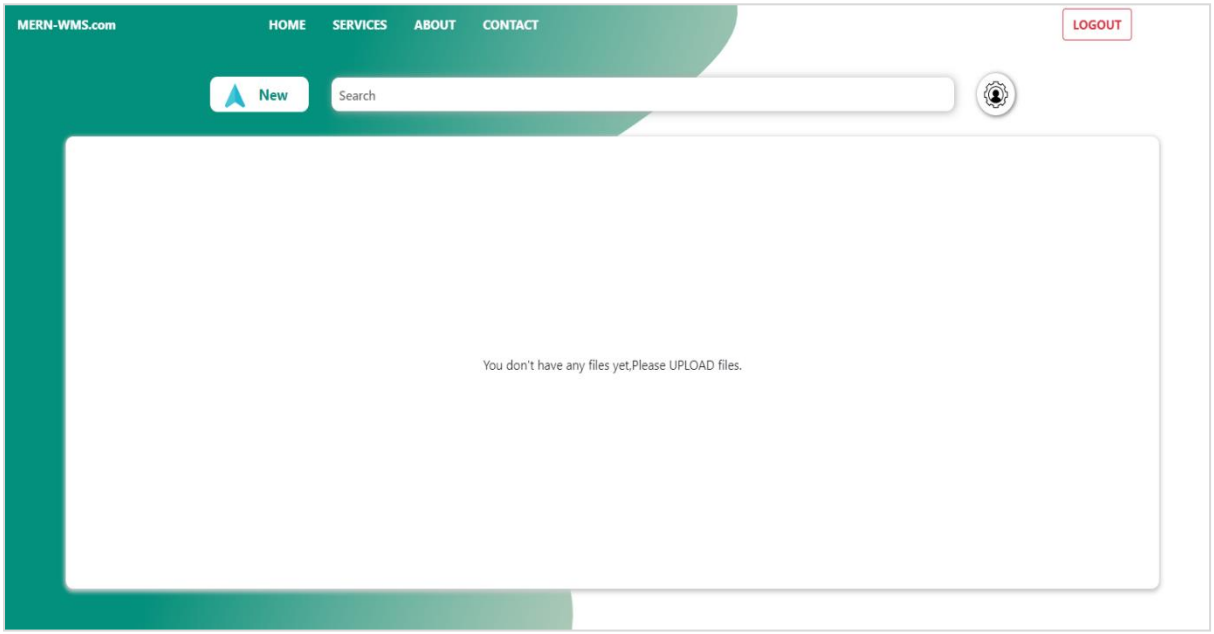

Figure IV.9. Page des utilisateurs authentifiés

Pour ajouter un nouveau document l'utilisateur clique sur le bouton **« New »**, puis importe le document et sa structure en cliquant sur « **PickFile** » et « **ImportStructure** ».

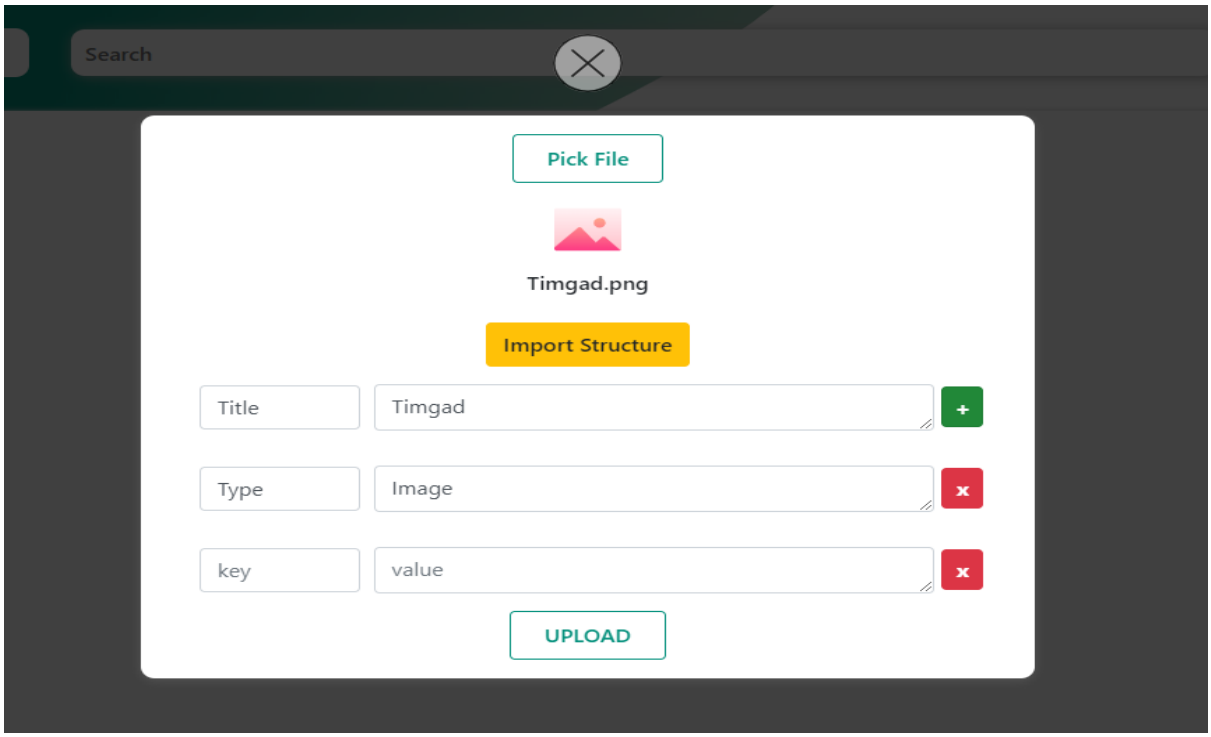

Figure IV.10. Fenêtre d'ajout de documents avec leurs structures

Si l'utilisateur ne connaît pas XML et JSON, il peut simplement décrire le contenu du document dans les champs comme dans la figure IV.11.

L'utilisateur peut aussi personnaliser les champs selon le besoin du document, il peut les ajouter ou les supprimer en cliquant sur les boutons « **+** » et « **x** ».

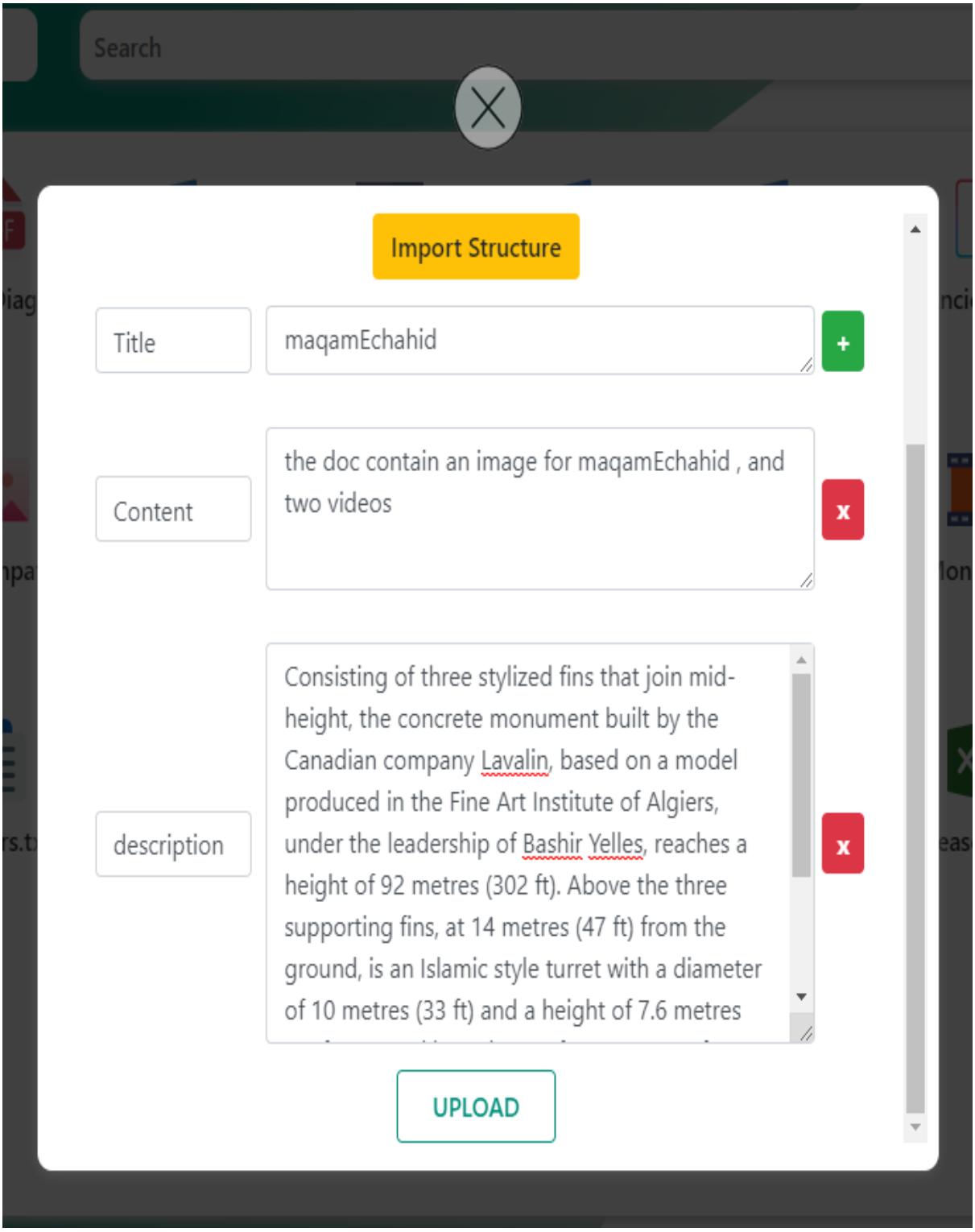

Figure IV.11. Décrire le document par champs

Après que l'utilisateur importe le document et remplie les champs, le bouton « **UPLOAD** » sera actif, avec un clic sur ce bouton l'opération sera terminée avec succès, et le document sera affiché dans la section des fichiers.

La figure suivante (figure IV.12) affiche le compte utilisateur après l'ajout de plusieurs documents.

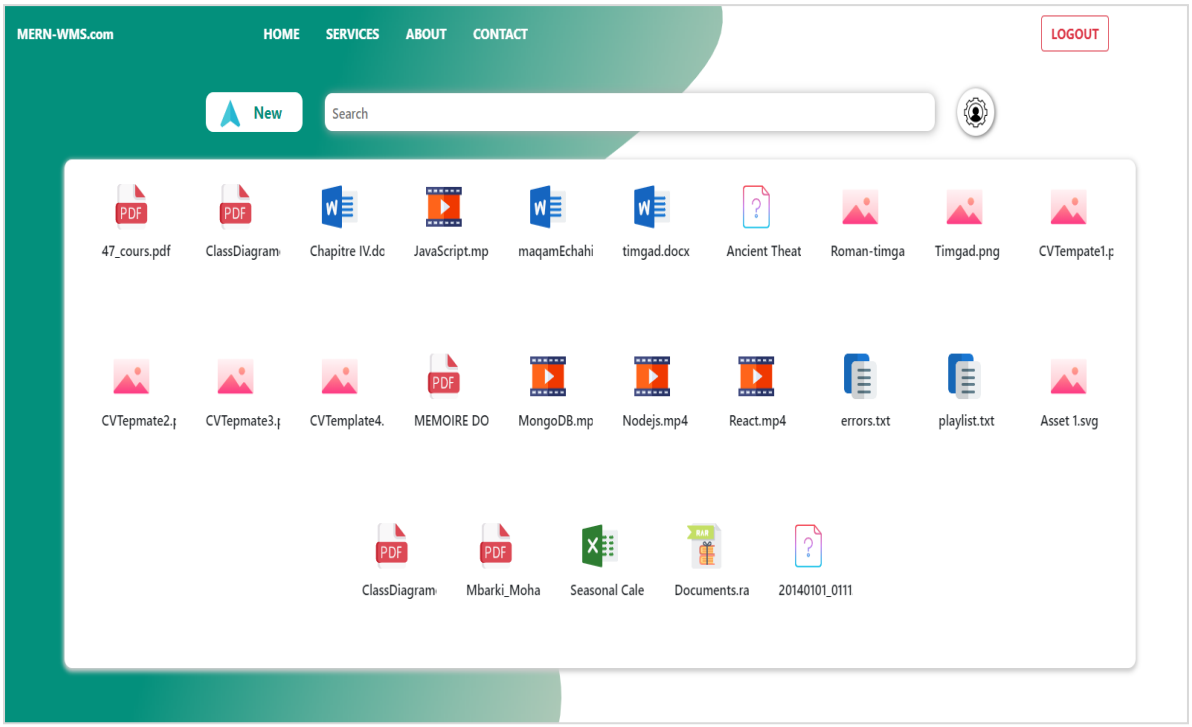

Figure IV.12. Compte d'utilisateur après l'ajout de documents

IV.5.4. Gestion des documents

La gestion des documents comprend plusieurs opérations soit pour les administrateurs, soit pour les utilisateurs simples, chaque document à une icône d'options qui affiche les opérations disponibles (figure IV.13).

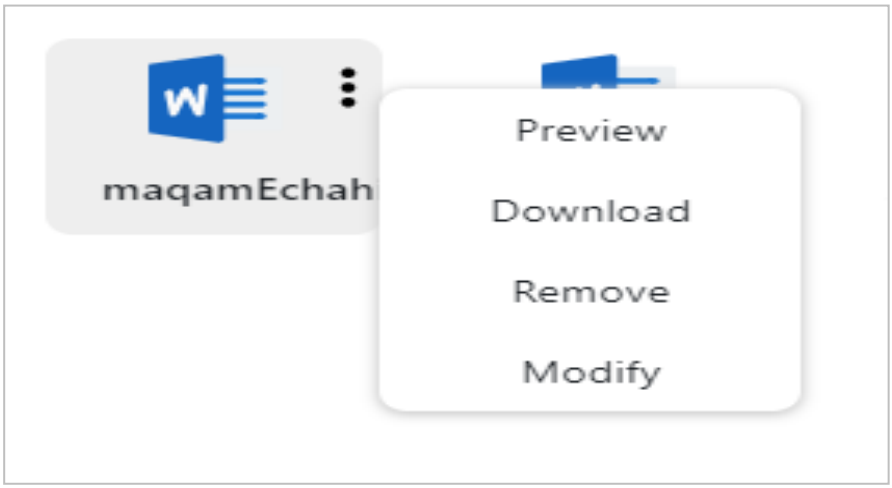

Figure IV.13. Liste d'options de document

Comme nous pouvons le voir dans la liste des options, nous avons quatre options pour chaque document qui nous permettent de prévisualiser, télécharger, modifier et supprimer Les documents.

Les options « **Preview** » et « **Download** » sont disponibles pour tous les utilisateurs (administrateurs ou utilisateurs simples).

Les options « **Remove** » et « **Modify** » sont juste pour les administrateurs.

IV.5.5. Filtrage des documents

Pour trouver certains documents par titre, contenu, métadonnées ou type, etc. L'utilisateur peut taper n'importe quoi dans le champ de recherche, puis le système filtrera les documents et n'affichera que les documents qui correspondent à ce que l'utilisateur a saisi. Les figures suivantes montrent quelques exemples de filtrage :

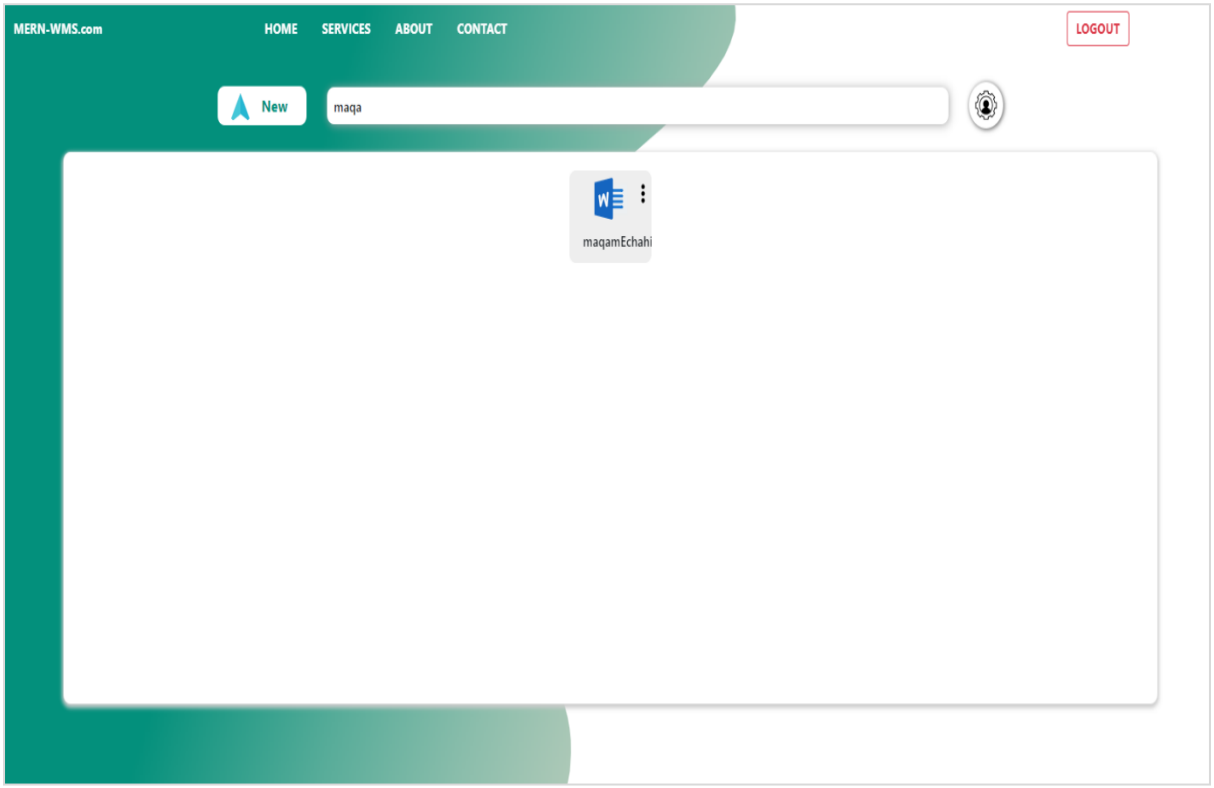

Figure IV.14. Filtrage des documents par titre = "maqa"

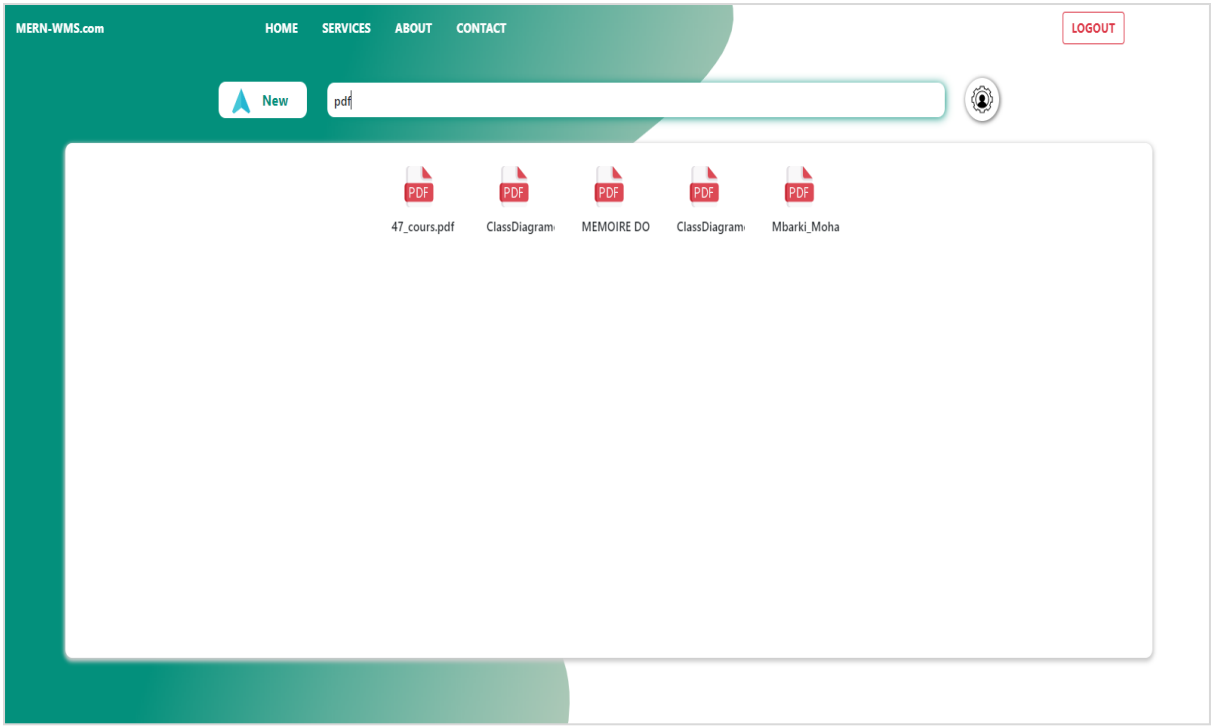

Figure IV.15. Filtrage des documents par type= "pdf"

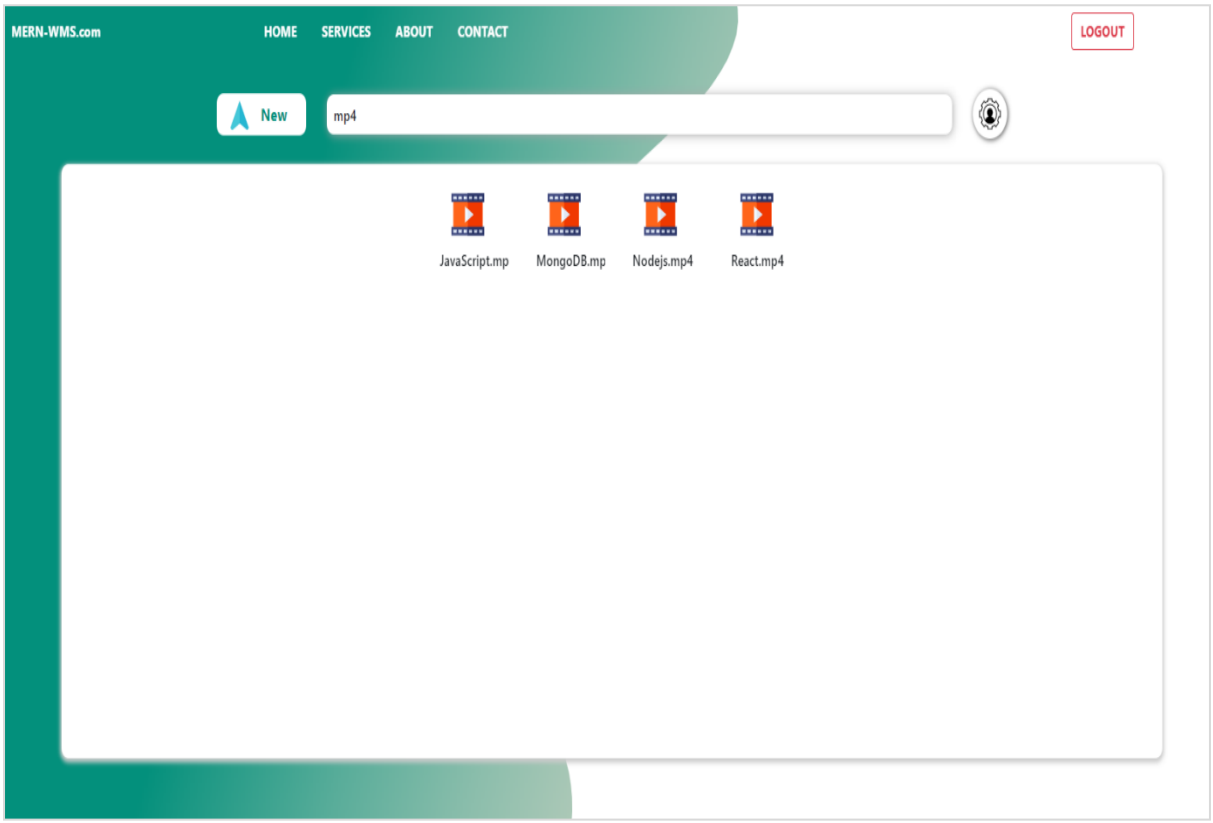

Figure IV.16. Filtrage des documents par type = "mp4"

IV.5.6. Gestion des comptes

Tous les utilisateurs, qu'ils soient administrateurs ou utilisateurs simples peuvent gérer leurs comptes, comme la mise à jour des informations du compte, et la suppression du compte (figure IV.17). Cependant, les administrateurs ont plus d'options que les utilisateurs simples (les administrateurs ont des options système complètes), ils peuvent ajouter leurs propres utilisateurs et donner à chaque utilisateur un rôle, soit "admin" ou "user" (figure IV.18), ils peuvent aussi gérer la liste des utilisateurs (figure IV.19).

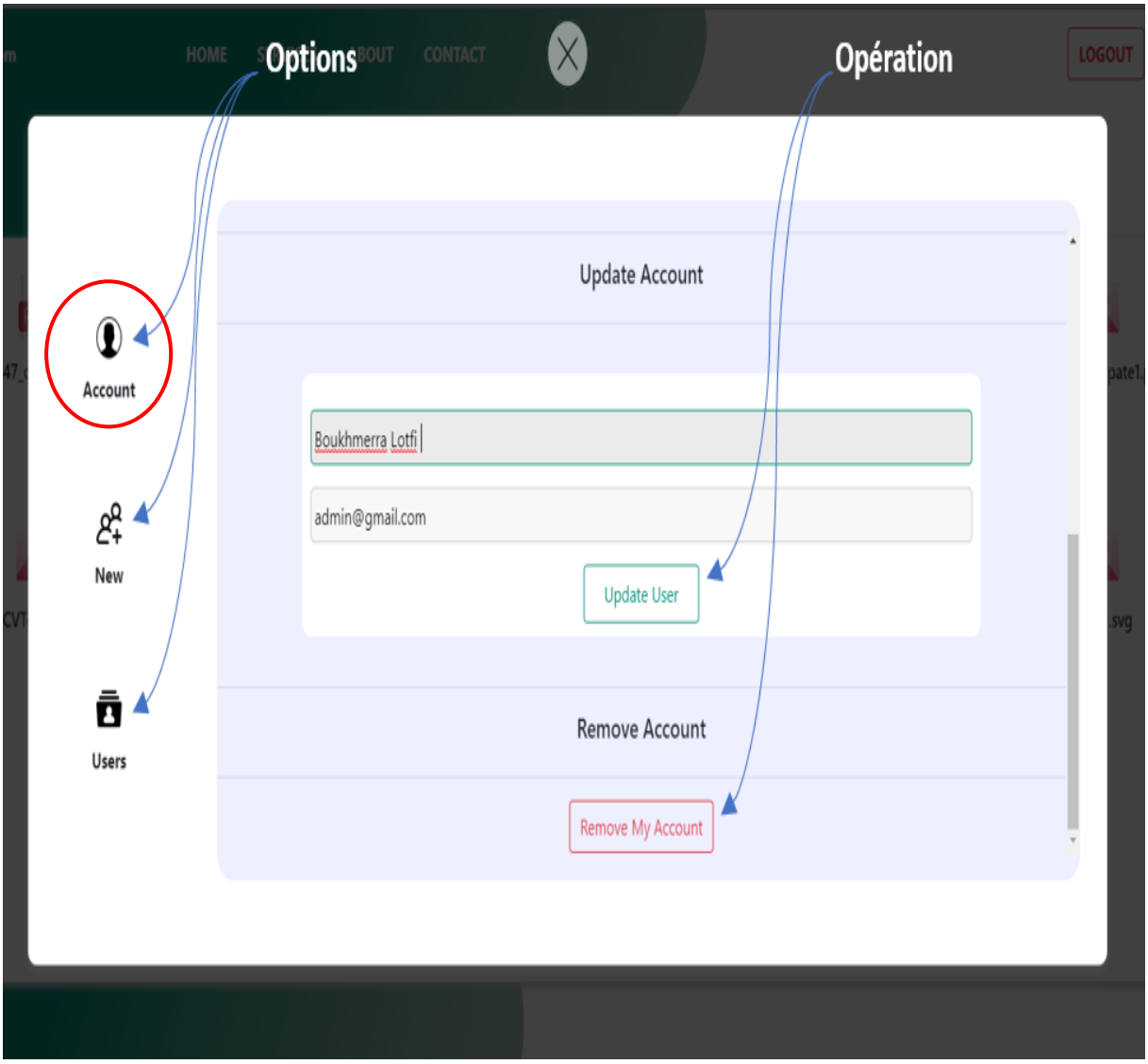

Figure IV.17. Fenêtre du tableau de bord

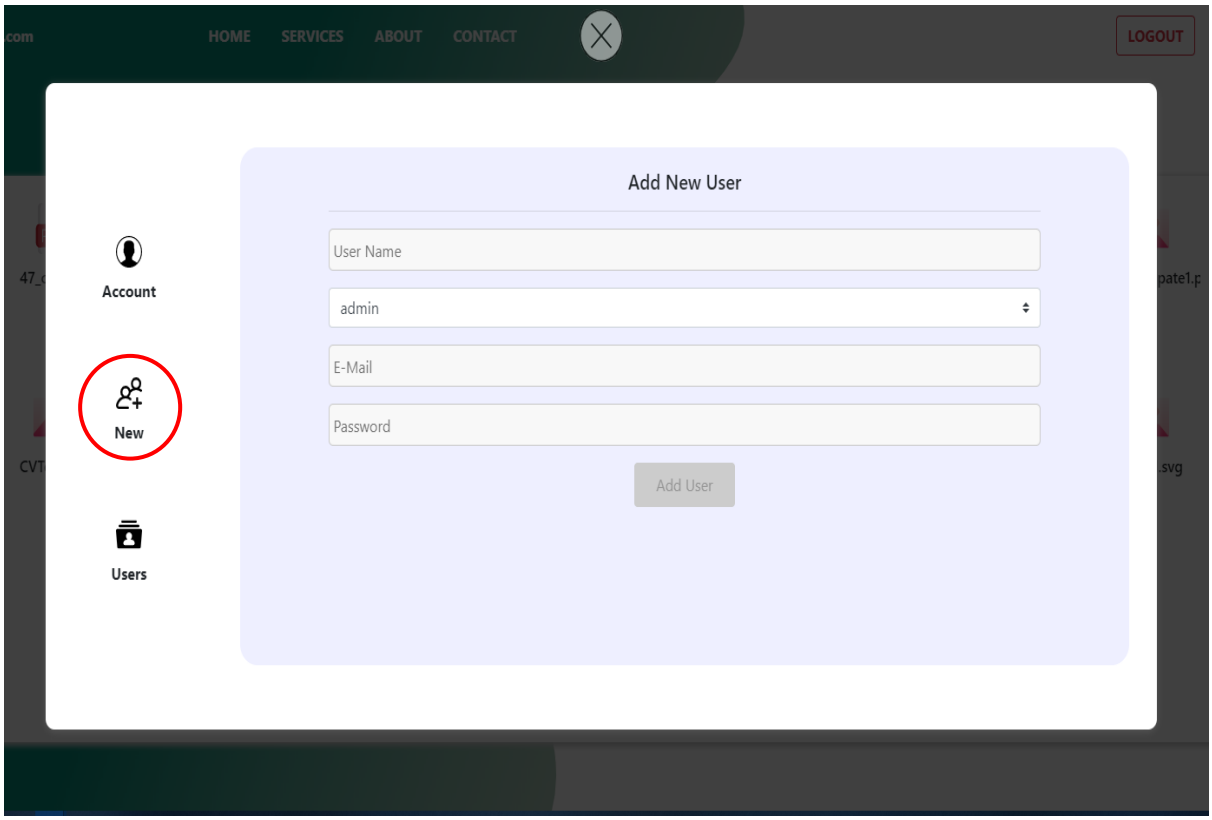

Figure IV.18. Fenêtre pour ajouter un nouvel utilisateur

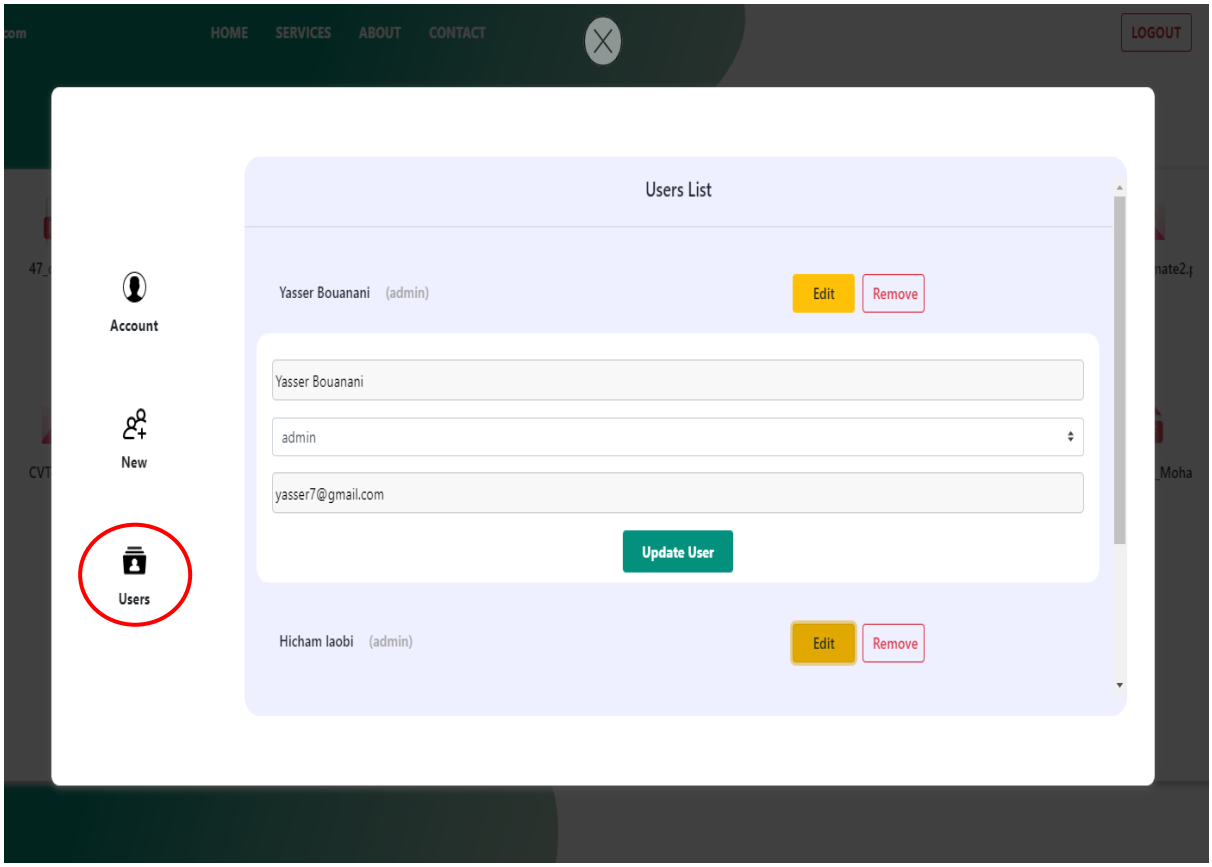

Figure IV.19. Fenêtre de gestion de la liste des utilisateurs

IV.6. Conclusion

Au cours de ce chapitre, nous avons présenté une implémentation d'un système de gestion d'entrepôt de documents dans une application web réactive et moderne. Nous avons commencé par une conception générale à travers une modélisation UML. Ensuite nous avons recensé les différentes technologies logicielles utilisées pour le développement. Enfin, nous avons exposé certaines captures d'écrans qui illustrent les différentes facettes de notre application.

Malheureusement, nous avons dû limiter les cas d'expérimentations à quelques types de documents, et nous n'avons pas pu effectuer toutes les expérimentations souhaitées, et ce pour faute de temps. Néanmoins, les résultats obtenus dans le cadre de la gestion des entrepôts de documents multimédia restent très encourageants.

Conclusion générale

Ce mémoire avait pour ambition de réalisé une adaptation et une application d'une approche de classification et de gestion de documents multimédias hétérogènes afin d'offrir un outil qui permet de gérer des grandes masses de documents stocker dans un entrepôt. L'approche choisie est celle proposée dans [1], elle se base sur trois axes : la modélisation, la classification et l'exploitation de documents multimédia fragmentés en granules. La combinaison de ces axes permet de stocker et de manipuler, sous plusieurs facettes et selon plusieurs points de vue, un grand nombre de documents multimédia hétérogènes.

Il nous a fallu dans un premier temps porté notre attention sur la modélisation des documents multimédia. L'adaptation du modèle de représentation des documents multimédias proposé dans [1] permet d'éliminer les faiblesses rencontrée lors de l'emploi de ce modèle, il s'appuie essentiellement sur la notion de fragmentation, ce qui facilite la gestion des contenus et des structures de documents complexes, ceci sans perdre la vision globale de ces documents. Il intègre également plusieurs niveaux d'organisation : une couche spécifique qui représente un document particulier, et une couche générique qui représente une classe de documents, une description structurelle qui correspond à la modélisation de structures logiques, et une description de métadonnées qui décrive le contenu des éléments logiques. Ces deux descriptions (structurelle et métadonnées) forment la description sémantique du document. Le modèle permet également à l'auteur du document de définir si les autres utilisateurs peuvent mettre à jour la description de son document ou pas, cela offre plus de flexibilité lors de la phase d'exploitation et évite la violation des droits d'auteur.

Ensuite, nous avons utilisée lors de l'intégration de nouveaux documents la même démarche de classification que dans [1] (sans adaptation). Cette démarche de classification par comparaison de structures permet d'ajouter et d'organiser ces nouveaux documents dans une base de stockage tout en respectant les caractéristiques de l'approche de modélisation. Elle repose sur l'extraction de la structure spécifique du document à intégrer, cette structure traduit l'organisation spécifique d'un document indépendamment de son contenu, ceci permet d'une part de fragmenter le document en granules monomédia. D'autre part, cette démarche permet de regrouper dans la même classe, c'est à dire sous une même structure générique, des documents ayant des structures spécifiques similaires à cette structure générique. Cette classification peut nécessiter, lorsque cela est possible, c'est à dire lorsque la structure spécifique est suffisamment proche d'une des structures génériques connues, l'adaptation d'une des structures génériques ou de la structure spécifique.

Dans un troisième temps, nous avons mis en place une démarche de recherche utilisée dans la phase d'exploitation. La démarche proposé facilite la recherche des fragments documentaire et donne plus de flexibilité, de rapidité et surtout de précision lors de cette recherche, et cela en offrant un mode de recherche précise qui permet d'effectué deux type de recherche dans l'entrepôt de documents : une recherche dans les descriptions structurelle, et une recherche dans les descriptions des métadonnées.

Enfin, pour appliquer cette approche nous avons développé une application web qui permet de gérer les entrepôts de documents multimédia. Cette application permet de stocker les

documents de façon dynamique de manière à faciliter leur gestion et leur exploitation. Ainsi il est possible de retrouver et restituer des granules documentaires (éléments, attributs, composants ou métadonnées) ou des documents entiers (selon le niveau de précision souhaité par l'utilisateur) qui vérifient des critères de recherches bien spécifiques.

Cependant, les résultats obtenu dans ce travail de mémoire sont très encourageants et nous permettent d'envisager un approfondissement des approches présentées, elles nous aident aussi à mettre en évidence un certain nombre de perspectives. En particulier, l'évolution du modèle générique en étendant la gestion de la multi-structuralitée des documents afin d'être capable de gérer tous types de structures, que ça soit une structure logique, sémantique, temporelle, physique, etc. Cette évolution nécessite d'envisager une représentation sous forme de graphes.

Bibliographie

[1] M. Mbarki, "Gestion De L'heterogeneite Documentaire : Le Cas D'un Entrepot De Documents Multimedia", *Thèse de doctorat en* informatique, L'universite De Toulouse III – Paul Sabatier, 2008.

[2] A. Jdidi, "Modélisation Générique De Documents Multimédia Par Des Métadonnées : Mécanismes D'annotation Et D'interrogation", *Thèse de doctorat en* informatique, L'universite De Toulouse III – Paul Sabatier, 2005.

[3] M-R. Koivunen, R. Swick, E. prud'hommeaux, "Annotea Shared Bookmarks, Annotea Shared Bookmerks" In proc of the KCAP 2003 workshop on knowledge markup & semantic annotation, Sanibel, Florida, USA, octobre, 2003.

[4] M. Kunze, D. Rosner, "An XMLbased Approach for the Presentation and Exploitation of Extracted Information", The 1 st International Workshop on Web Documents Analysis (WDA 2001), Seattle, Washington, USA, Septembre, 2001.

[5] J.B. Dowdalla, I. Pavlidisa, G. Bebisb, "Face Detection in the NearIR Spectrum", Image and Vision Computing, vol 21 N°7, 2003, pp. 565-578.

[6] C. Barras, X. Zhu, S. Meignier, J-L. Gauvain, "Improving Speaker Diarization", DARPA RT04, Palisades, New York, USA, 2004.

[7] F. Pelisson, D. Hall, O. Riff, J-L. Crowley, "Brand Identification Using Gaussian Derivative Histograms", Machine Vision and Applications, vol. 16, no 1, 2004, pp. 41- 46.

[8] I. Amous, A. Jedidi, F. Sèdes, "A contribution to multimedia document modeling and organizing", 8Th International conference on Object Oriented Information Systems, OOIS'02, Montpelier, France, 2002, Springer LNCS n° 2425, pp. 434-444.

[9] E. Loisant, H. Ishikawa, J. Martinez, "Designing a Model Independent Multimedia Database", Days of Science and Technology, Tokyo, Japan, 2002.

[10] E. Bertino, G. Guerrini, I. Merlo, M. Mesiti, "An Approach to Classify Semi-Structured Objects", European Conference on Object-Oriented Programming, Lisboa, Portugal, LNCS 1628, 1999, pp. 416-440.