

République Tunisienne
Ministère de l'Enseignement Supérieur et de la
Recherche Scientifique

Université Technologique
Ecole Supérieure des Sciences Appliquées et de
la Technologie privée de Gabès

Département de Génie Logiciel



Cycle de Formation d'Ingénieurs dans la
discipline Génie Logiciel

Projet de Fin d'Etudes

N° d'ordre:

MEMOIRE

Présenté à

L'Ecole Supérieure des Sciences Appliquées et de la
Technologie Privée de Gabès

Département de Génie Logiciel

En vue de l'obtention

Du Diplôme National d'Ingénieur en *Génie Logiciel*

Présenté par

Mondher GUEMRI

**Conception et réalisation d'une application inter service
multiplateforme modulaire**

Soutenu le

devant le jury d'examen :

M.

Président

M.

Rapporteur

M.

Encadreur

Dédicaces

*Je dédie ce travail à mes très chers parents ; à ma mère qui ne cesse
Jamais de m'encourager, à mon père qui était toujours à mes côtés à
Tout moment, Que dieu tout puissant les garde pour moi,
À mon frère et ma sœur pour leur amour et leur soutien inconditionnel,
À toute ma famille,
Et À tous mes amis où qu'ils soient.*

Remerciement

Je tiens à exprimer mes remerciements à tous ceux qui ont rendu ce travail possible. Leurs aides précieuses, leurs conseils fructueux et leurs encouragements, tout au long de l'élaboration de ce projet de fin d'études, m'a permis de le réaliser dans la meilleure considération.

Je veux rendre un hommage particulier à mon encadreur :

Mr Gharbi Sami

Pour son soutien et ses conseils précieux, pour la richesse et la qualité de leur enseignement et qui déploient de grands efforts pour assurer à leurs étudiants une formation actualisée.

À mon enseignante :

Madame Madhbouh khouloud

Qui n'a pas cessé de nous prodiguer ses conseils et qui n'a épargner aucun effort pour contribuer à la réussite de notre travail.

Aux membres du jury qui ont bien voulu m'honorer de leur présence d'évaluer notre travail.

Un grand merci à toutes les personnes qui m'ont soutenue de près ou de loin au cours de la réalisation de ce modeste travail.

Merci pour tous

Table des matières

INTRODUCTION GENERALE	1
CHAPITRE I. ETUDE PREALABLE	3
INTRODUCTION	4
I.1. CADRE DE PROJET	4
<i>I.1.1 La gestion de tache</i>	4
<i>I.1.2 La gestion documentaire</i>	5
I.2. CONTEXTE DE L'APPLICATION	5
I.3. ETUDE DE L'EXISTANT	5
I.4. CRITIQUE DE L'EXISTANT.....	5
I.5. PROBLEMATIQUE.....	6
I.6. SOLUTION PROPOSEE.....	7
CONCLUSION	8
CHAPITRE II. CONCEPTION	9
INTRODUCTION	10
II.1. CHOIX DE LA METHODOLOGIE DE CONCEPTION	10
<i>II.1.1 Présentation d'UML</i>	10
II.2. DIAGRAMME DES CAS D'UTILISATION.....	11
<i>II.2.1 Identification des acteurs</i>	11
<i>II.2.2 Identification des fonctionnalités</i>	12
II.3. SPECIFICATIONS FONCTIONNELLES DETAILLEES	15
<i>II.3.1 Cas d'utilisation globale</i>	15
<i>II.3.2 Module Gestion des Managers</i>	17
<i>II.3.3 Module Gestion des Utilisateurs</i>	18
<i>II.3.4 Module Gestion des Taches</i>	19
<i>II.3.5 Module Gestion des Évènements</i>	20
<i>II.3.6 Module Gestion des Documents</i>	22
<i>II.3.7 Module Gestion de profil</i>	23
<i>II.3.8 Module Gestion d'archives</i>	24
II.4. DIAGRAMME DE CLASSES	25
II.5. DIAGRAMMES DE SEQUENCES	26
<i>II.5.1 Diagramme de séquence « Authentification et Autorisation »</i>	27
<i>II.5.2 Diagramme de séquence « Consulter la liste des taches »</i>	30
II.6. SPECIFICATION NON FONCTIONNELLE.....	32
CONCLUSION	32
CHAPITRE III. REALISATION	33
INTRODUCTION	34
III.1. ENVIRONNEMENT DE TRAVAIL.....	34
<i>III.1.1 Environnement logiciel</i>	34
<i>III.1.2 Outils de développement et SGBD</i>	36
III.2. ARCHITECTURE	40
<i>III.2.1 Architecture technique</i>	41
<i>III.2.2 Architecture applicative</i>	41

III.3.	SOA - ARCHITECTURE ORIENTEE SERVICE	42
III.3.1	<i>Le notion service Web</i>	43
III.3.2	<i>REST</i>	43
III.3.3	<i>Représentation des ressources : JSON</i>	43
III.4.	API REST.....	44
III.5.	PRESENTATION DES INTERFACES	48
III.5.1	<i>Interface d'authentification</i>	48
III.5.2	<i>Interface d'inscription</i>	49
III.5.3	<i>Interface d'accueil</i>	50
III.5.4	<i>Interface d'ajout d'une tache</i>	51
III.5.5	<i>Interface de modification d'une tache</i>	52
III.5.6	<i>Interface Commentaire</i>	53
III.5.7	<i>Interface Listes des taches</i>	54
III.5.8	<i>Interface Ajout d'un Evènement</i>	55
III.5.9	<i>Interface de modification d'un Evènement</i>	56
III.5.10	<i>Interface d'ajout d'un manager</i>	57
III.5.11	<i>Interface d'Activation de compte Utilisateur</i>	58
III.5.12	<i>Interface d'ajout d'un document :</i>	59
III.5.13	<i>Interface Profil</i>	60
III.5.14	<i>Interface d'Archive</i>	61
	CONCLUSION	62
	CONCLUSION GENERALE	63
	BIBLIOGRAPHIE ET WEBOGRAPHIE	64
	GLOSSAIRE	65
	ANNEXES	66

Acronymes

HTTP	Hyper Text Transfer Protocol
UML	Unified Modeling Language
IoC	Inversion of Control
MVC	Model-View-Controller
JSON	JavaScript Object Notation
REST	Representational State Transfer
SOA	Service-oriented architecture
HQL	Hibernate Query Language
JEE	Java Enterprise Edition
API	Application programming interface
JWT	JSON Web Token

Table des figures

FIGURE 1. PROBLEMATIQUE-OUTILS CLASSIQUES DE GESTION DE TACHE	6
FIGURE 2. PROBLEMATIQUE-OUTILS CLASSIQUES DE GESTION DE DOCUMENT	7
FIGURE 3. DIAGRAMME DE CAS D'UTILISATION GLOBAL	16
FIGURE 4. DIAGRAMME DE CAS D'UTILISATION GESTION DES MANAGERS	17
FIGURE 5. CAS D'UTILISATION GESTION DES UTILISATEURS	18
FIGURE 6. CAS D'UTILISATION GESTION DES TACHES	19
FIGURE 7. CAS D'UTILISATION « GESTION DES EVENEMENTS ».....	20
FIGURE 8. CAS D'UTILISATION « GESTION DES DOCUMENTS »	22
FIGURE 9. CAS D'UTILISATION « GESTION DE PROFIL ».....	23
FIGURE 10. CAS D'UTILISATION « GESTION D'ARCHIVE »	24
FIGURE 11. REPRESENTATION DU DIAGRAMME DE CLASSE	26
FIGURE 12. DIGRAMME DE SEQUENCE AUTHENTIFICATION ET AUTORISATION BASE SUR JWT.....	27
FIGURE 13. DIGRAMME DE SEQUENCE « AUTHENTIFICATION ET AUTORISATION ».....	29
FIGURE 14. DIAGRAMME DE SEQUENCE « CONSULTER LA LISTE DES TACHES ».....	31
FIGURE 15. ARCHITECTURE TECHNIQUE DE L'APPLICATION	41
FIGURE 16. ARCHITECTURE APPLICATIVE DE L'APPLICATION.....	42
FIGURE 17. ARCHITECTURE SOA	43
FIGURE 18. PAGE D'AUTHENTIFICATION	49
FIGURE 19. PAGE D'INSCRIPTION	50
FIGURE 20. PAGE D'ACCUEIL.....	51
FIGURE 21. PAGE D'AJOUT D'UNE NOUVELLE TACHE	52
FIGURE 22. PAGE MODIFIER UNE TACHE.....	53
FIGURE 23. PAGE COMMENTAIRE	54
FIGURE 24. PAGE LISTES DES TACHES	55
FIGURE 25. PAGE D'AJOUT D'UN EVENEMENT	56
FIGURE 26. PAGE MODIFIER UN EVENEMENT	57
FIGURE 27. PAGE D'AJOUT D'UN MANAGER	58
FIGURE 28. PAGE D'ACTIVATION DE COMPTE UTILISATEUR	59
FIGURE 29. PAGE D'AJOUT D'UN DOCUMENT	60
FIGURE 30. PAGE PROFIL	61
FIGURE 31. PAGE D'ARCHIVE	62

Liste des tableaux

TABLEAU 1 . <i>LES PRINCIPAUX CAS D'UTILISATION ET LES ACTEURS CORRESPONDANTS</i>	17
TABLEAU 2. <i>DESCRIPTION DE CAS D'UTILISATION GESTION DES MANAGERS.....</i>	18
TABLEAU 3. <i>DESCRIPTION DE CAS D'UTILISATION « GESTION DES UTILISATEURS »</i>	19
TABLEAU 4. <i>DESCRIPTION DE CAS D'UTILISATION « GESTION DES TACHES ».....</i>	20
TABLEAU 5. <i>DESCRIPTION DE CAS D'UTILISATION « GESTION DES EVENEMENTS »</i>	21
TABLEAU 6. <i>DESCRIPTION DE CAS D'UTILISATION « GESTION DES DOCUMENTS »</i>	23
TABLEAU 7. <i>DESCRIPTION DE CAS D'UTILISATION « GESTION DE PROFIL »</i>	24
TABLEAU 8. <i>DESCRIPTION DE CAS D'UTILISATION « GESTION D'ARCHIVE »</i>	25
TABLEAU 9. <i>DESCRIPTION DES API REST DE L'APPLICATION.....</i>	48

Introduction générale

Dans le cadre de la formation d'ingénieurs à l'Ecole Supérieur des Sciences Appliquées et de technologie privée de Gabes, nous sommes appelés à consolider notre formation théorique par des connaissances et des acquis pratiques à travers des projets dont le plus évaluatif en terme de contenu de la formation est celui de fin d'études, c'est dans ce cadre que s'articule notre application que nous sommes appelés à développer (concevoir et réaliser).

La compétitivité des entreprises repose essentiellement sur leur capacité à utiliser efficacement les ressources dont elles disposent afin de satisfaire au mieux la demande de leurs clients. Selon le contexte, le terme « ressource » peut renvoyer à des notions très différentes telles que les matières premières, l'outil de production, les canaux de distribution, ou le personnel. Nous nous intéressons ici à la gestion efficace du personnel et de document qui est primordiale pour de nombreuses entreprises.

Les objectifs d'une bonne organisation administrative sont de faire vite, bien et pas cher. Donc d'être productif en maintenant un excellent niveau de qualité dans les tâches exécutées. En effets, ces tâches ne produisent pas de valeur ajoutée, mais contribuent à ce que l'ensemble soit bien huilé.

La gestion de tâches, ça peut être aussi simple qu'une liste de tâches notées sur une feuille, que vous cochez au fur et à mesure (c'est ce que font un grand nombre de gens). Bien sûr, des applications permettent aujourd'hui de faire la même chose sur votre ordinateur ou votre téléphone, avec plus ou moins d'options superflues. Mais, Quand vous devez gérer plusieurs tâches et plusieurs membres de votre équipe, les choses peuvent rapidement devenir ingérables avec une application de gestion de tâches basique.

Aussi comme la gestion de tâches, un bon système de gestion documentaire, correctement communiqué à l'ensemble des employés est donc un outil précieux, voire essentiel au bon fonctionnement d'une entreprise. Ce n'est plus à l'utilisateur d'aller à la recherche du document, mais c'est le document qui vient directement vers l'utilisateur.

C'est dans cette optique que se situe notre projet de fin d'études. Il s'agit de développer une application inter service multiplateforme modulaire.

Le présent rapport trace les phases du déroulement du projet. Il sera présenté en trois chapitres :

Le premier chapitre intitulé Etude de Préalable. Dans ce chapitre, nous décrivons le contexte de l'application et les solutions proposés.

Le chapitre suivant intitulé conception est dédié à la présentation de méthodologies de conception et la modélisation conceptuelle de notre solution.

Le dernier chapitre intitulé réalisation présente les étapes de la réalisation du projet.

Le rapport s'achève par une Conclusion Générale dans laquelle nous exposerons les perspectives du projet.

Chapitre I. Etude Préalable

Introduction

L'étude préalable constitue une étape préliminaire pour la réalisation d'une application.

En effet, elle permet d'analyser, d'évaluer et de critiquer le fonctionnement habituel, tout en élaborant la liste des solutions possibles.

Ce chapitre sera réservé pour présenter l'étude préalable de notre projet. Nous commençons par le cadre de projet. Ensuite problématique et étude de l'existant nous ont permis de cerner nos objectifs afin de développer un système de qualité dans le futur. Enfin, nous proposons les différentes solutions aux problèmes soulevés.

I.1. Cadre de projet

Recourir à un logiciel de gestion dans une entreprise, c'est faire un choix stratégique pour son entreprise et être assuré d'une optimisation de son organisation et communication :

- C'est s'organiser à l'aide d'un agenda collaboratif simplifié
- C'est gagner en productivité grâce à un environnement de gestion de tâche.
- C'est être performant et faire comme les entrepreneurs réussis qui ont compris qu'il faut savoir déléguer mais aussi digitaliser, avec le partage des documents ou fichiers en ligne.

I.1.1 *La gestion de tâche*

La gestion de tâches permet à une entreprise de mieux organiser son activité et de savoir à tout moment qui fait quoi dans l'entreprise et quel est le taux d'occupation des salariés.

I.1.2 *La gestion documentaire*

La gestion de documents d'entreprise représente l'organisation et le contrôle de tous les documents et des données dont une entreprise a besoin pour exercer son activité.

I.2. Contexte de l'application

La gestion de l'information, enjeu essentiel à l'heure du tout numérique, apparaît comme un atout de plus en plus recherché. De fait, dans le monde des entreprises, une gestion performante de données semble essentielle et passe, entre autre, par une bonne gestion documentaire.

Une bonne méthode de classement permet donc au personnel de l'entreprise de retrouver un document beaucoup plus rapidement et plus facilement.

Aussi La gestion de tâches ne se limite pas à établir une liste de tâches, mais implique également de mener chaque tâche de sa création à son achèvement, de définir des échéances à respecter. Un logiciel de gestion des tâches permet aux équipes de suivre leurs tâches et de gagner en productivité et en efficacité.

I.3. Etude de l'existant

Les fonctionnalités classiques d'un logiciel de gestion de tâche sont :

- Créer et affecter des tâches à des collaborateurs
- Définir une échéance
- Attachement de document
- Permettre aux individus de gérer leurs tâches personnelles
- Modifier les statuts d'une tâche (à faire, en cours, terminé)

I.4. Critique de l'existant

Il y a peu d'inconvénients à un gestionnaire de tâches : ils sont presque indispensables dans tous les domaines.

Pour autant on peut noter quelques points qui peuvent être décevants

- Il peut devenir complexe et illisible.
- La suppression des tâches est parfois trop facile : une mauvaise manipulation peut entraîner une suppression sans s'en rendre compte.
- Aucun suivi des documents ajouté.

I.5. Problématique

Une mauvaise organisation dans la gestion des tâches peut mener à des résultats chaotiques qui sont catastrophiques pour la productivité des entreprises et une mauvaise communication ou l'absence de communication au sein d'une équipe qui gèrent des tâches peut causer l'échec d'un projet.

Mal utilisées, les messageries sont chronophages et sources de stress.

Envoyer, recevoir, transférer un message, joindre un fichier... Autant de tâches qui nuisent l'efficacité de salariés.

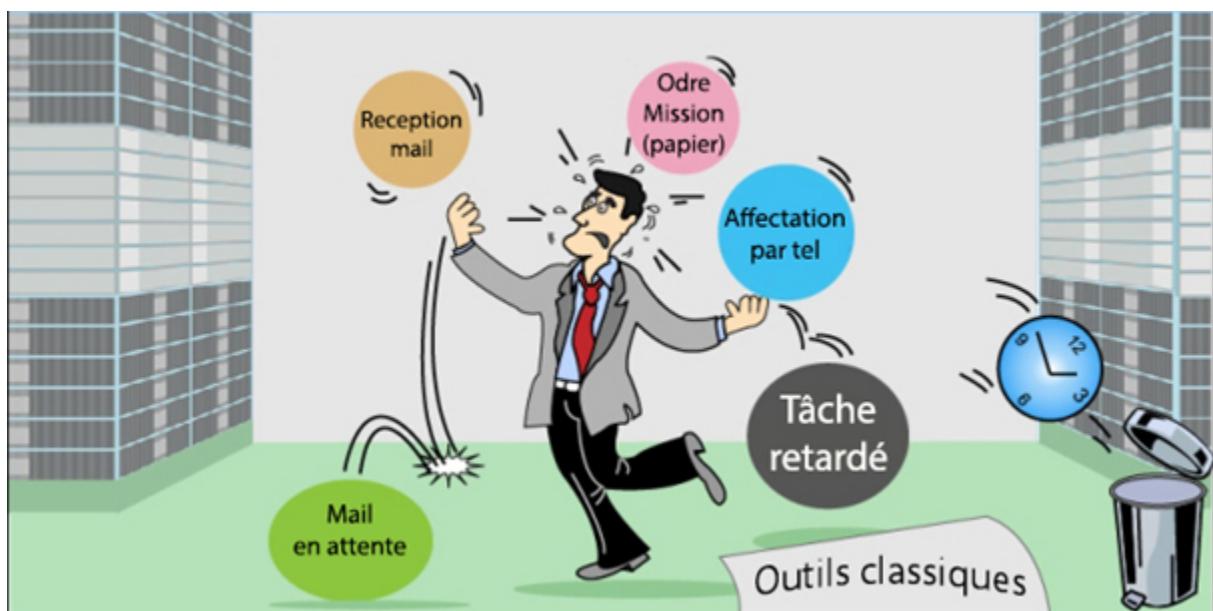


Figure 1. Problématique-Outils classiques de gestion de tache

Comme la gestion de tâche, Le stockage des données physiques reste encore problématique dans la plupart des sociétés et les expose à d'importants risques de perte ou de fuite d'informations de valeur.

La plupart des entreprises gardent leurs archives papier dans des caissons ou dans les tiroirs de leurs bureaux. Cette manière de classer pose des problèmes lors de la recherche d'informations qui peut durer plusieurs minutes, voire plusieurs heures, ce qui rend inefficace les entreprises.

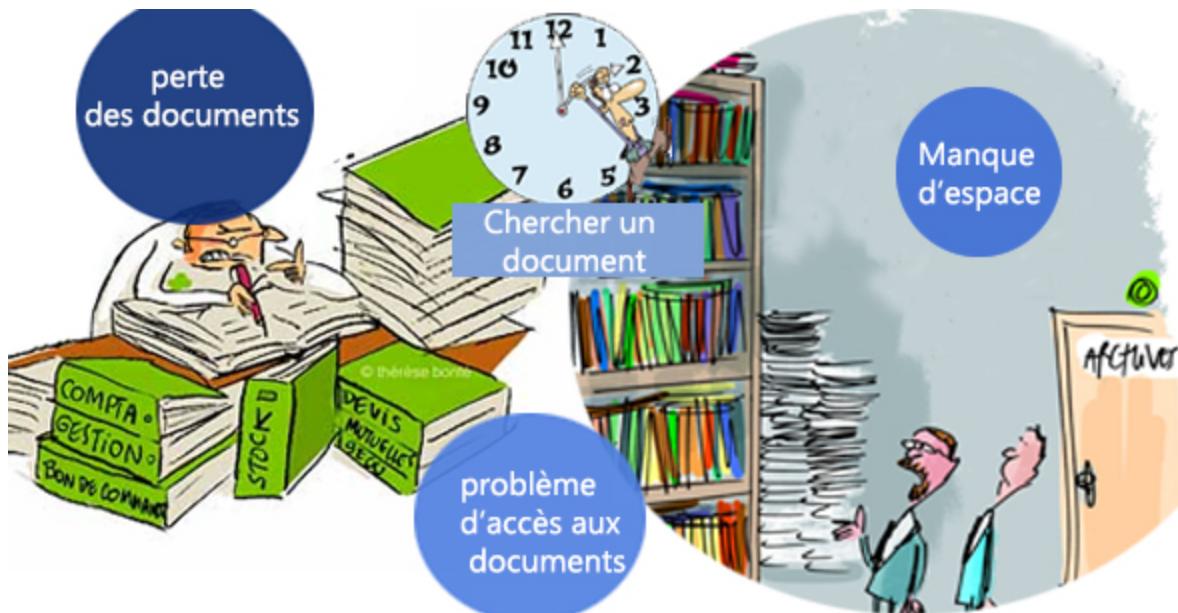


Figure 2. Problématique-Outils classiques de gestion de document

I.6. Solution proposée

Notre solution consiste à mettre en œuvre une application inter service qui facilite au entreprise la gestion de tâche et La gestion électronique de documents qui permet de remplacer des "documents papier" par leur représentation sous forme de "documents électroniques", en réponse à la fois aux problèmes d'archivage et de recherche des documents complets.

Nous chercherons donc à travers cette application à atteindre les objectifs suivants

- Dématerrialisation, Partage et synchronisation des documents.
- Assigner aisément les différentes tâches avec une date butoir, la description du travail, des commentaires, des pièces jointes...
- Plus de problème pour trouver la dernière version d'un document, l'avancée d'une tâche, le responsable d'une mission... Tout sera

centralisé et simple d'utilisation, avec une interface claire et ergonomique.

- Assuré la sécurité de l'information et l'accès au document.
- Un espace sociaux collaboratif qui permet de crée des évènements.
- Moins d'emails, optimisation du temps, moins de réunions, plus de temps pour la production et les idées.
- Doit être disponible et consultable de partout (tablette, mobile ou ordinateur).

Conclusion

Dans ce chapitre, nous avions pu insérer notre projet dans son contexte en présentant la solution adoptée pour résoudre les problèmes et qui répond à nos besoins. Dans le chapitre suivant, nous allons présenter la conception qui a été mise en œuvre tout au long de la réalisation de ce projet.

Chapitre II. Conception

Introduction

Lors de ce chapitre, nous allons identifier les diagrammes de classes, de cas d'utilisation et de séquence réalisés pour mettre en œuvre l'architecture de notre application. La motivation fondamentale de la modélisation est de fournir une démarche antérieure afin de réduire la complexité du système étudié lors de la conception et d'organiser la réalisation du projet en définissant les modules et les étapes de la réalisation. Plusieurs démarches de modélisation sont utilisées.

Nous adoptons dans notre travail une approche objet basée sur un outil de modélisation UML.

II.1. Choix de la méthodologie de conception

Dans le cadre de notre projet, nous avons opté pour le langage UML comme une approche de conception. Ci-dessous, nous présentons ce langage puis nous justifions notre choix.

II.1.1 *Présentation d'UML*

UML (Unified Modeling Language) est un langage formel et normalisé en termes de modélisation objet. Son indépendance par rapport aux langages de programmation, aux domaines de l'application et aux processus, son caractère polyvalent et sa souplesse ont fait de lui un langage universel. En plus UML est essentiellement un support de communication, qui facilite la représentation et la compréhension de solution objet. Sa notation graphique permet d'exprimer visuellement une solution objet, ce qui facilite la comparaison et l'évaluation des solutions. L'aspect de sa notation, limite l'ambigüité et les incompréhensions.

UML fournit un moyen astucieux permettant de représenter diverses projections d'une même représentation grâce aux vues.

Une vue est constituée d'un ou plusieurs diagrammes. On distingue deux types de vues :

- **La vue statique**, permettant de représenter le système physiquement

- *Diagrammes de classes* : représentent des collections d'éléments de modélisation statiques (classes, paquetages...), qui montrent la structure d'un modèle.
- *Diagrammes de cas d'utilisation* : identifient les utilisateurs du système (acteurs) et leurs interactions avec le système.

➤ **La vue dynamique**, montrant le fonctionnement du système

- *Diagrammes de séquence* : permettent de représenter des collaborations entre objets selon un point de vue temporel, on y met l'accent sur la chronologie (envois de messages).

II.2. Diagramme des cas d'utilisation

Les cas d'utilisation décrivent un ensemble d'actions réalisées par le système, en réponse à une action d'un acteur.

II.2.1 *Identification des acteurs*

L'identification des acteurs sert à délimiter le contour du système. Par définition : un acteur représente l'abstraction d'un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système) qui interagissent directement avec le système(Bourque,2014).

- **Administrateur** : cet acteur possède tous les droits d'accès qui lui permettent d'administrer le système.sa fonction principale est la gestion des manager.
- **Manager** : cet acteur est la gestionnaire de l'application .sa fonction principale est la gestion des taches et des documents.
- **Utilisateur** : cet acteur est en interaction avec le manager par la tâche assignée.

II.2.2 *Identification des fonctionnalités*

Nous décrivons pour chaque acteur les cas d'utilisation. On distingue les cas d'utilisation suivants

➤ *Utilisateur*

- S'inscrire (saisie de coordonnée)
- S'authentifier (si le compte est validé par l'administrateur ou le manager)
- Gérer le profil (mettre à jour ses informations personnel).
- Consulter la liste des tâches (chercher une tâche par nom)
- Consulter la liste des évènements (chercher un évènement par nom)
- Modifier l'état d'une tâche (changer l'état d'une tâche de 'non-commencer' à 'en cours' où 'terminer')
- Ajouter un document (uploader un document sur le plateforme).
- Ajouter un évènement (création d'un nouvel évènement)
- Modifier un évènement (mettre à jour les informations sur un évènement créer).
- Supprimer un évènement (supprimer un évènement qu'il la créer).
- Télécharger un document (accès pour téléchargement d'un document partager ou privé)
- Consulter la liste de contact (chercher et afficher les information d'un contact)

➤ *Administrateur :*

- S'authentifier (par de coordonnée enregistrer par défaut dans la base de donnée)
- Consulter la liste des utilisateurs (chercher un utilisateur par nom).
- Consulter la liste des managers (chercher un manager par nom).
- Consulter la liste des tâches (chercher une tâche par nom).

- Consulter la liste des évènements (chercher un évènement par nom).
- Consulter la liste des documents (chercher un document par nom).
- Consulter la liste des contacts (chercher et afficher les information d'un contact)
- Gérer le profil (mettre à jour ses informations personnel).
- Ajouter une Tache (assigné une tache à un utilisateur)
- Ajouter un manager (saisir la coordonnée de compte manager).
- Ajouter un évènement (création d'un nouvel évènement).
- Ajouter un document (uploader un document sur le plateforme)
- Archiver un document (placer un document dans l'archive)
- Archiver un évènement (placer un évènement dans l'archive)
- Archiver une tâche (placer une tâche dans l'archive)
- Modifier un document (mettre à jour les informations d'un document).
- Activer par mail un compte utilisateur (envoie d'un Email d'activation).
- Modifier le compte manager (mettre à jour l'adresse Email et username de compte manager).
- Modifier le compte utilisateur (mettre à jour l'adresse Email et username de compte utilisateur).
- Activer par email un compte Utilisateur (envoie d'un email d'activation de compte utilisateur).
- Modifier un évènement (mettre à jour les informations sur un évènement créer).
- Modifier une tache (mettre à jour les informations sur une tache créer).
- Valider un évènement (changer l'état d'un évènement de 'En Attente' à 'reporter', 'annuler' ou 'confirmer')

- Télécharger un document (accès pour téléchargement d'un document partager ou privé)
- Restaurer un document (Récupérer un document placer dans l'archive)
- Supprimer un document (suppression d'un document archive)
- Restaurer une tâche (Récupérer une tâche placer dans l'archive)
- Supprimer une tâche (suppression d'une tâche archivé).
- Supprimer un évènement (suppression d'un évènement archivé ou créer).

➤ **Manager :**

- S'authentifier (par de coordonnée fournit par l'administrateur).
- Consulter la liste des utilisateurs (chercher un utilisateur par nom).
- Consulter la liste des tâches (chercher une tâche par nom).
- Consulter la liste des évènements (chercher un utilisateur par nom).
- Consulter la liste des documents (chercher un document par nom).
- Consulter la liste des contacts (chercher un contact par nom).
- Ajouter un document (uploader un document sur le plateforme).
- Ajouter évènement (création d'un nouvel évènement)
- Gérer le profil (mettre à jour ses informations personnel).
- Ajouter une tâche (assigné une tâche a un utilisateur).
- Archiver un document (placer un document dans l'archive)
- Archiver un évènement (placer un évènement dans l'archive)
- Archiver une tâche (placer un évènement dans l'archive)
- Modifier un document (mettre à jour les informations sur un document créer).
- Modifier un évènement (mettre à jour les informations sur un évènement créer).

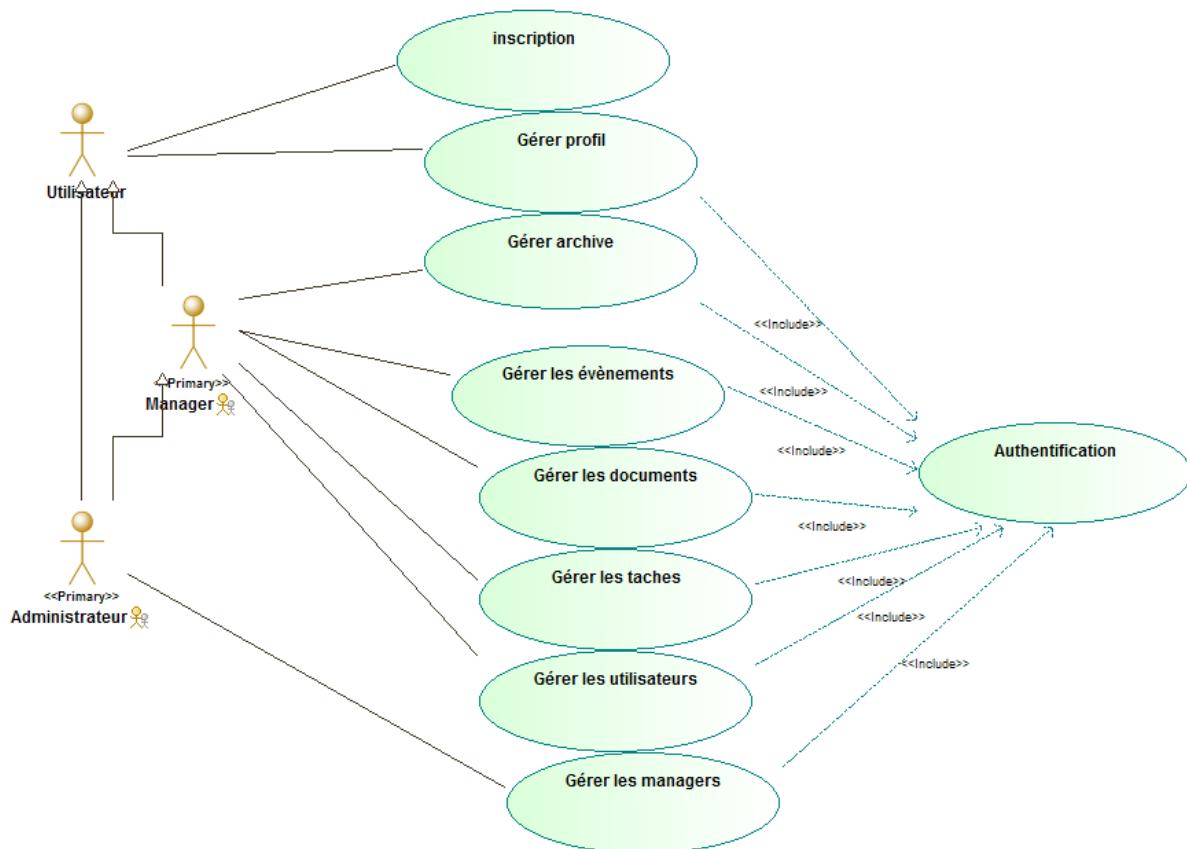
- Modifier une tache (mettre à jour les informations sur une tâche créer).
- Valider un évènement (changer l'état d'un évènement de ‘En Attente’ à ‘reporter’, ‘annuler’ ou ‘confirmer’)
- Télécharger un document (accès pour téléchargement d'un document partager ou privé)
- Supprimer un évènement (suppression d'un évènement archivé ou créer)
- Activer par mail un compte utilisateur (envoie d'un Email d'activation).
- Consulter la liste de contact (chercher et afficher les information d'un contact)
- Restaurer un document (récupérer un document placer dans l'archive)
- Supprimer un document (suppression d'un document archive).

II.3. Spécifications fonctionnelles détaillées

La spécification des besoins sert à associer chaque acteur réactif du système à l'ensemble d'actions avec lesquelles il intervient. Nous utilisons les diagrammes des cas d'utilisations pour modéliser les besoins fonctionnels.

II.3.1 *Cas d'utilisation globale*

Le diagramme des cas d'utilisation dans la figure donne une vue globale sur le système

**Figure 3.** Diagramme de cas d'utilisation global

Le tableau 1 présente les différents cas d'utilisation avec les messages reçus et émis

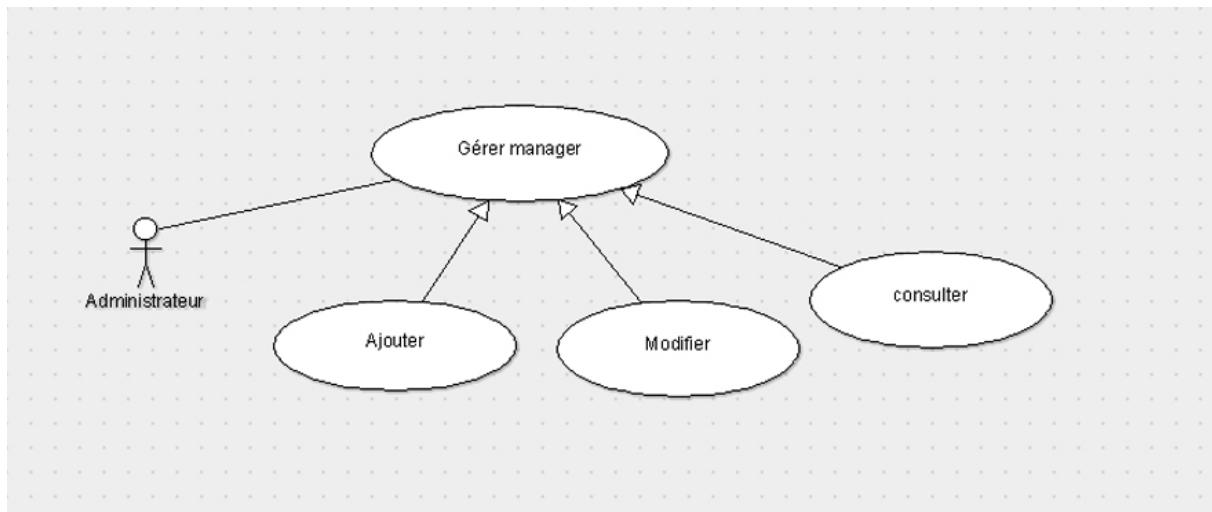
Cas d'utilisation	Acteur	Message reçus / émis
Gérer les utilisateurs	Manager, Administrateur	Reçus : Listes des utilisateurs. Emis : Activer par mail les Comptes des Utilisateurs, chercher et modifier
Gérer le profil	Utilisateur, Administrateur, Manager	Reçus : information sur le profil Emis : modifier, ajouter
Gérer les Taches	Manager, Administrateur,	Reçus : Listes des Taches Emis : ajouter, Archiver, modifier, chercher
Gérer les Evènements	Manager, Administrateur	Reçus : Listes des Evènements

		Emis : ajouter , modifier, Archiver ,chercher
<i>Gérer les manager</i>	Administrateur	Reçus : Listes des Managers Emis : ajouter , modifier, chercher
<i>Inscription</i>	Utilisateur	Reçus : Formulaire d'inscription Emis : nouvel utilisateur inscrit.
<i>Gérer les documents</i>	Manager, Administrateur	Reçus : Listes des Documents Emis : ajouter , archiver , chercher , modifier
<i>Gérer l'archive</i>	Manager, Administrateur	Reçus : Listes des Documents, Taches, Évènements Emis : supprimer, restaurer

Tableau 1 . Les principaux cas d'utilisation et les acteurs correspondants

II.3.2 **Module Gestion des Managers**

Le diagramme des cas d'utilisation suivant donne une vue globale sur le module « Gestion de manager »

**Figure 4. Diagramme de cas d'utilisation Gestion des managers**

Le tableau 2 représente les différents cas d'utilisation « Gestion des managers » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
Ajouter	Administrateur	Reçus : Formulaire d'ajout Emis : nouvel manager créer
Modifier	Administrateur	Reçus : informations de manager. Emis : manager modifié
consulter	Administrateur	Reçus : Listes des managers Emis : chercher par nom

Tableau 2. Description de cas d'utilisation Gestion des managers

II.3.3 Module Gestion des Utilisateurs

Le diagramme des cas d'utilisation suivant donne une vue globale sur le module « Gestion des utilisateurs »

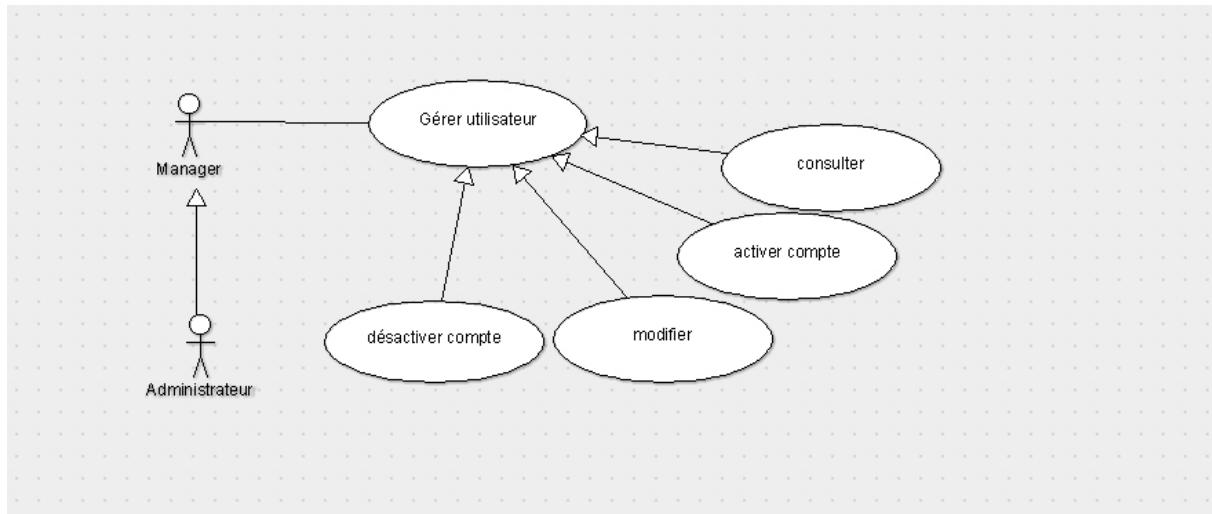


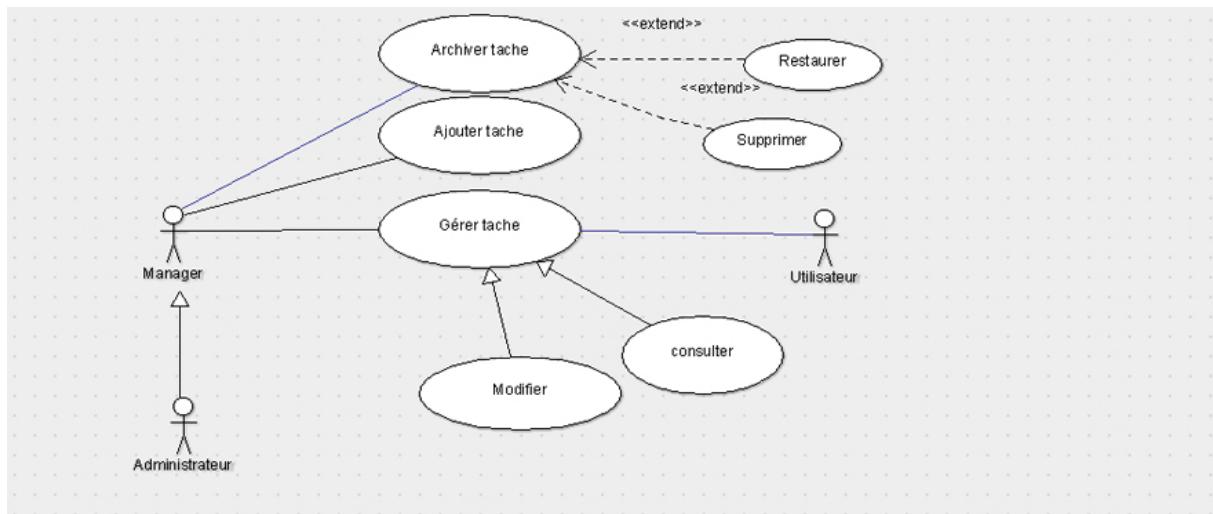
Figure 5. Cas d'utilisation Gestion des utilisateurs

Le tableau 3 représente les différents cas d'utilisation « Gestion des utilisateurs » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
Consulter	Manager, Administrateur	Reçus : Listes des utilisateurs Emis : consulter , chercher
Modifier	Manager, Administrateur	Reçus : informations de l'utilisateur. Emis : utilisateur modifié
désactiver compte	Manager, Administrateur	Reçus : informations de l'utilisateur. Emis : utilisateur désactivé
Activer compte	Manager, Administrateur	Reçus : Listes des utilisateurs Emis : envoi d'un email d'activation du compte.

Tableau 3. Description de cas d'utilisation « Gestion des utilisateurs »

II.3.4 Module Gestion des Taches

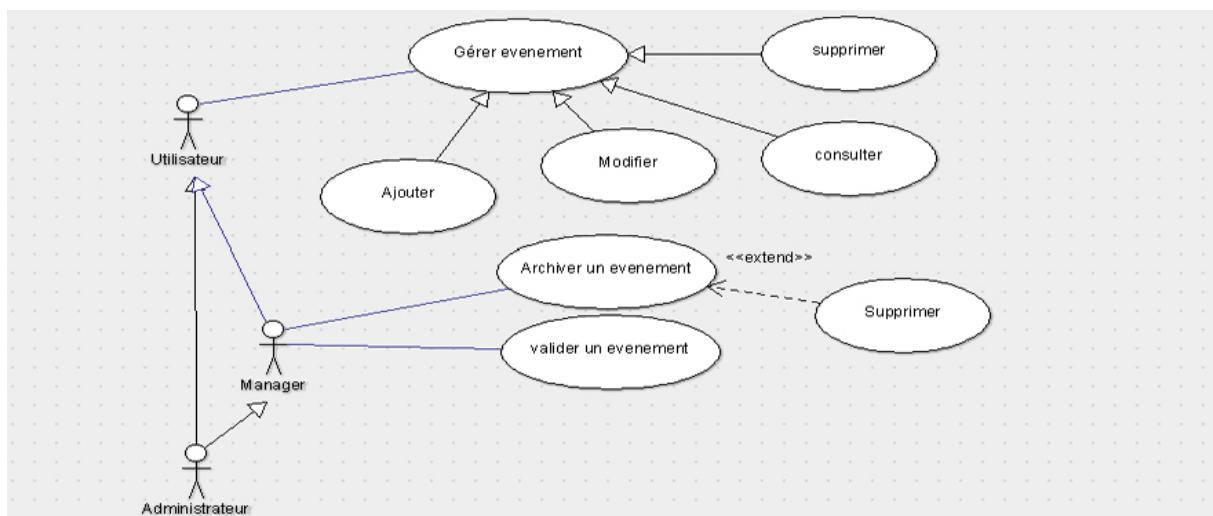
**Figure 6.** Cas d'utilisation Gestion des taches

Le tableau 4 représente les différents cas d'utilisation « Gestion des taches » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
Consulter	Manager, Utilisateur, Administrateur	Reçus : Listes des taches Emis : consulter, chercher
Modifier	Manager, Utilisateur, Administrateur	Reçus : informations sur une tache Emis : tache modifiée
Archiver une tache	Manager, Administrateur	Reçus : message de confirmation Emis : tache archiver
Ajouter une tache	Manager, Administrateur	Reçus : Formulaire d'ajout Emis : nouvel tache créer
supprimer	Manager, Administrateur,	Reçus : message de confirmation Emis :tache supprimer.
restaurer	Manager, Administrateur,	Reçus : Listes des taches Emis :tache restauré.

Tableau 4. Description de cas d'utilisation « Gestion des taches »

II.3.5 Module Gestion des Évènements

**Figure 7.** Cas d'utilisation « Gestion des évènements »

Le tableau 5 représente les différents cas d'utilisation « Gestion des taches » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
<i>Consulter</i>	Manager, Utilisateur, Administrateur	Reçus : Listes des évènements Emis : consulter, chercher
<i>Modifier</i>	Manager, Utilisateur, Administrateur	Reçus : informations sur un évènement Emis : évènements modifié
<i>Supprimer</i>	Manager, Administrateur, Utilisateur	Reçus : message de confirmation Emis : évènement supprimer
<i>Ajouter</i>	Manager, Utilisateur, Administrateur	Reçus : Formulaire d'ajout Emis : nouvel évènement créer
<i>Valider un évènements</i>	Manager, Administrateur	Reçus : informations sur un évènement Emis : évènements validé
<i>Archiver un évènement</i>	Manager, Administrateur	Reçus : message de confirmation Emis : évènement archivé

Tableau 5. Description de cas d'utilisation « Gestion des évènements »

II.3.6 *Module Gestion des Documents*

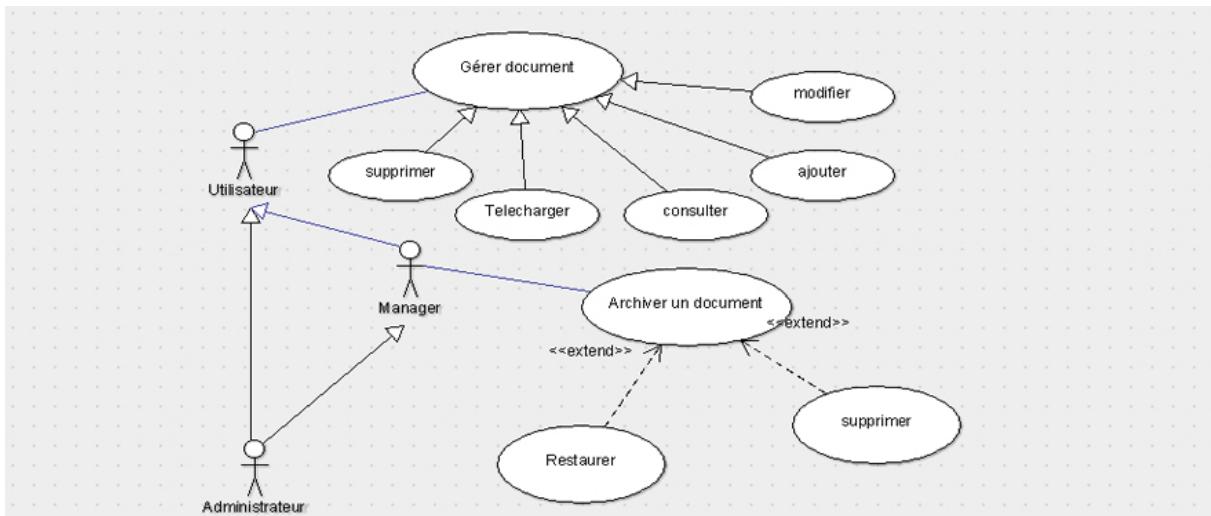
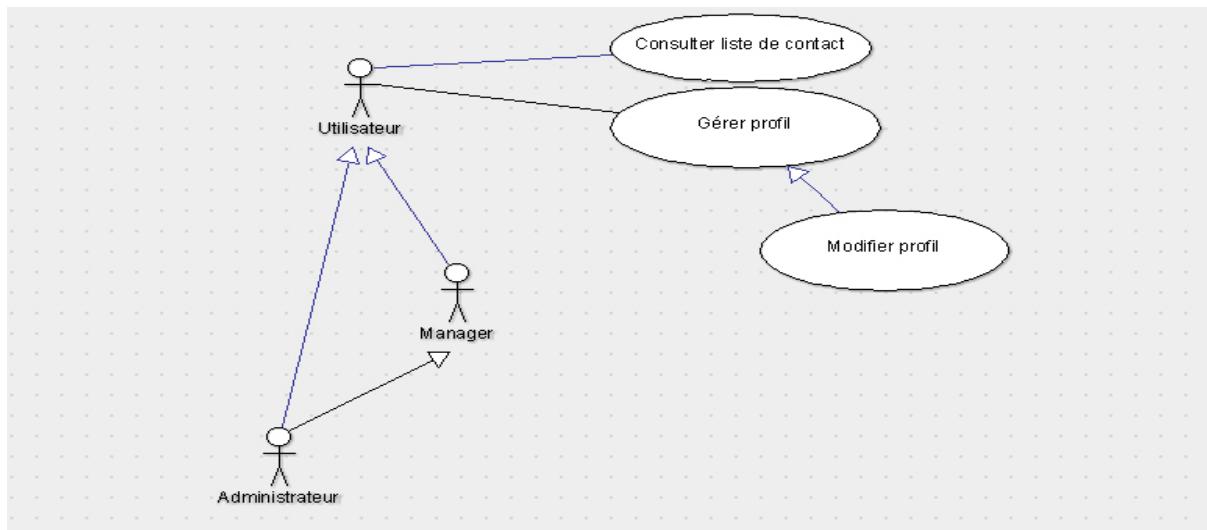


Figure 8. Cas d'utilisation « Gestion des documents »

Le tableau 6 représente les différents cas d'utilisation « Gestion des documents » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
<i>Consulter</i>	Manager, Utilisateur, Administrateur	Reçus : Listes des documents Emis : consulter, chercher
<i>Modifier</i>	Manager, Utilisateur, Administrateur	Reçus : informations sur un document Emis : document modifié
<i>Archiver un document</i>	Manager, Administrateur	Reçus : message de confirmation Emis : document archivé
<i>Ajouter</i>	Manager, Utilisateur, Administrateur	Reçus : Formulaire d'ajout Emis : nouveaux documents ajouter
<i>Télécharger</i>	Manager, Administrateur, Utilisateur	Reçus : informations sur un document Emis : document téléchargé

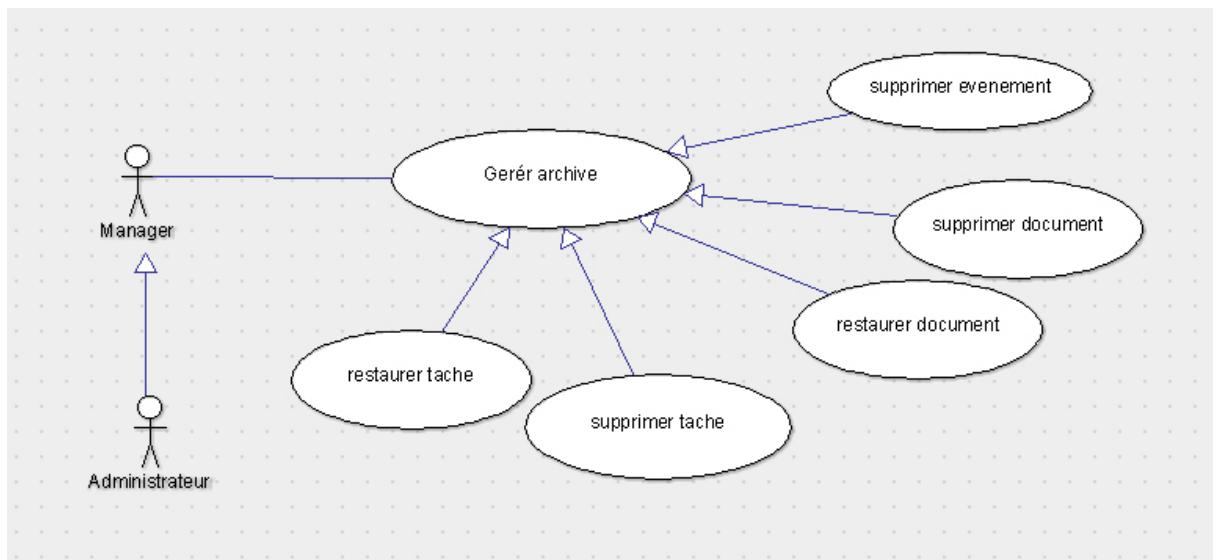
<i>Supprimer</i>	Manager, Administrateur	Reçus : message de confirmation Emis : document supprimé
<i>restaurer</i>	Manager, Administrateur,	Reçus : Listes des documents Emis : document restauré

Tableau 6. Description de cas d'utilisation « Gestion des documents »**II.3.7 Module Gestion de profil****Figure 9.** Cas d'utilisation « Gestion de profil »

Le tableau 7 représente les différents cas d'utilisation « Gestion de profil » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
<i>Modifier profil</i>	Utilisateur, Manager, Administrateur	Reçus : informations sur le profil Emis : profil modifié

<i>Consulter listes des contact</i>	Utilisateur, Manager, Administrateur	Reçus : listes des contacts Emis : consulter , chercher
-------------------------------------	--	--

Tableau 7. Description de cas d'utilisation « Gestion de profil »**II.3.8 *Module Gestion d'archives*****Figure 10.** Cas d'utilisation « Gestion d'archive »

Le tableau 8 représente les différents cas d'utilisation « Gestion d'archive » avec les messages reçus et émis

Cas d'utilisation	Acteur	Message reçus / émis
<i>Restaurer document</i>	Manager, Administrateur	Reçus : listes des documents archivés Emis : document restaurer
<i>Supprimer document</i>	Manager, Administrateur,	Reçus : message de confirmation Emis : document supprimé

<i>Restaurer tache</i>	Manager, Administrateur	Reçus : listes des taches archivées Emis : tache restaurer
<i>Supprimer tache</i>	Manager, Administrateur,	Reçus : message de confirmation Emis : tache supprimé
<i>Supprimer évènement</i>	Manager, Administrateur,	Reçus : message de confirmation Emis : évènement supprimer

Tableau 8. Description de cas d'utilisation « Gestion d'archive »

II.4. Diagramme de classes

La figure ci-dessous présente le diagramme de classes qui Contient toutes les informations telles que les classes, les méthodes, les associations et les

propriétés.

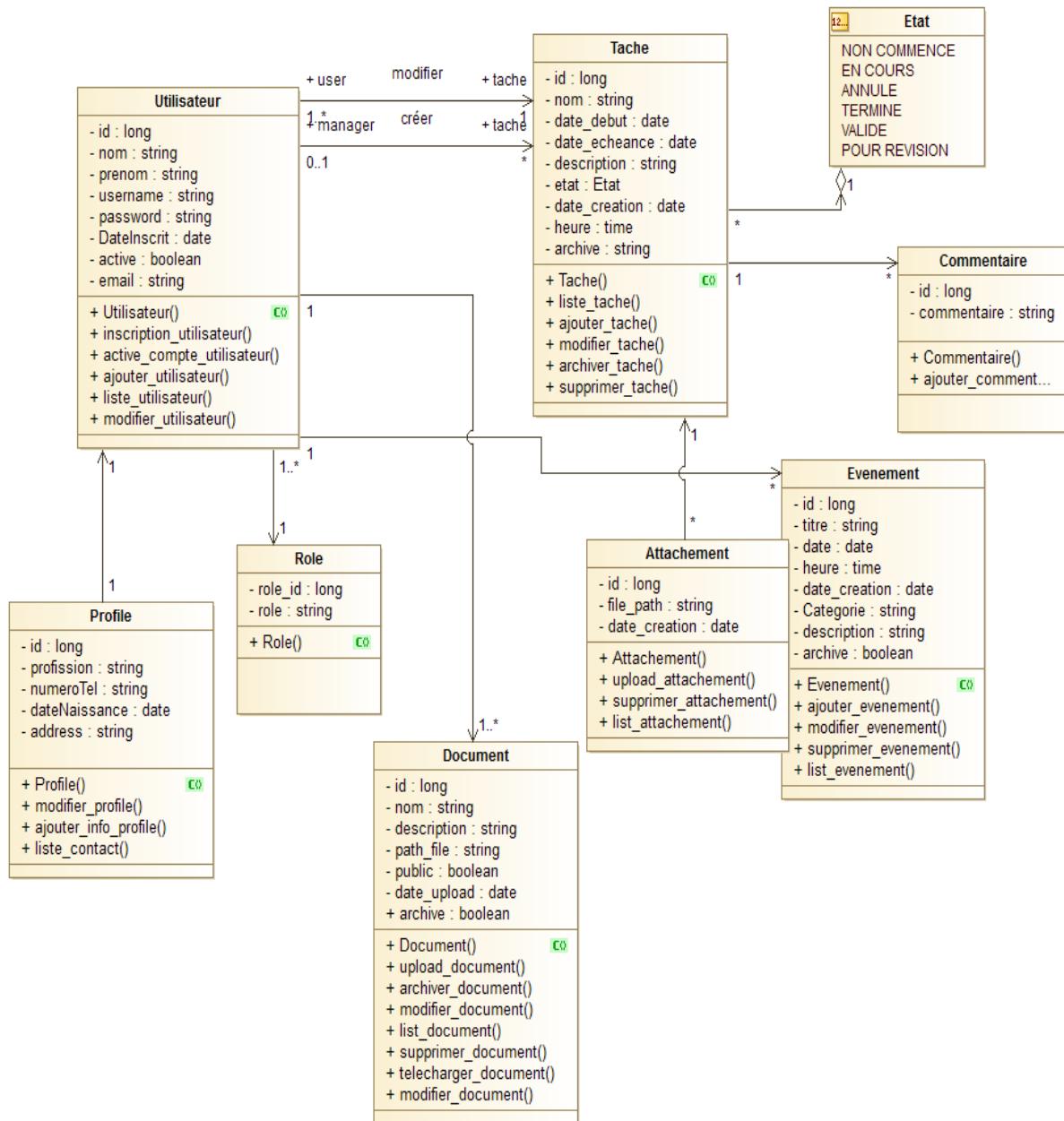


Figure 11. Représentation du diagramme de classe

II.5. Diagrammes de séquences

Les diagrammes de séquences représentent les interactions entre les objets en indiquant la chronologie des séquences. Les diagrammes de séquences ajoutent une dimension temporelle par rapport aux autre diagramme.

II.5.1 Diagramme de séquence « Authentification et Autorisation »

JWT (JSON Web Token) est une authentification stateless basée sur l'échange d'un token entre l'utilisateur et le serveur. Le token contient les informations suffisantes :

- Pour identifier l'utilisateur (le token est bien celui qui a été remis lors de l'authentification)
- Pour autoriser l'utilisateur (le token contient les droits du client)

Le diagramme de séquence ci-dessous présente une vue globale sur le séquencement des Interactions entre utilisateur, Authentication Controller et la base de donnée.

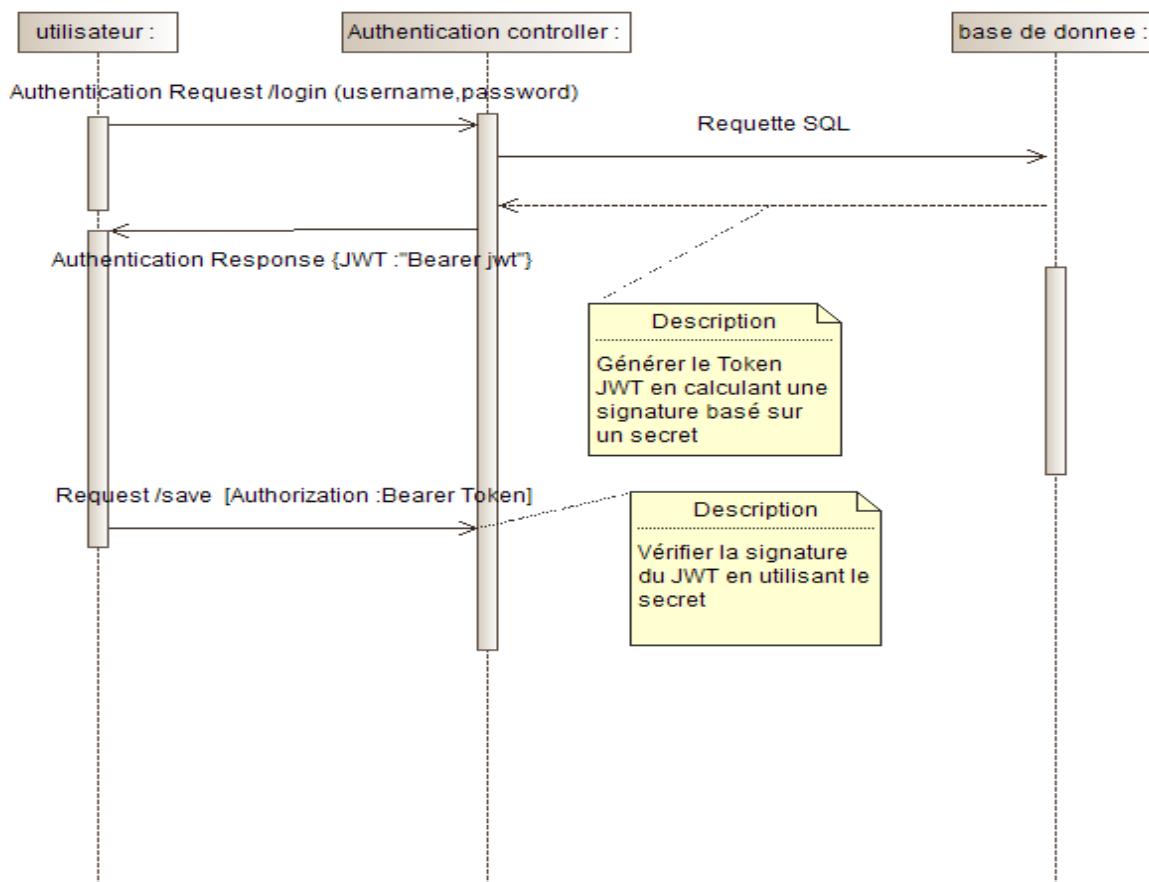


Figure 12. Diagramme de séquence Authentification et Autorisation basé sur JWT

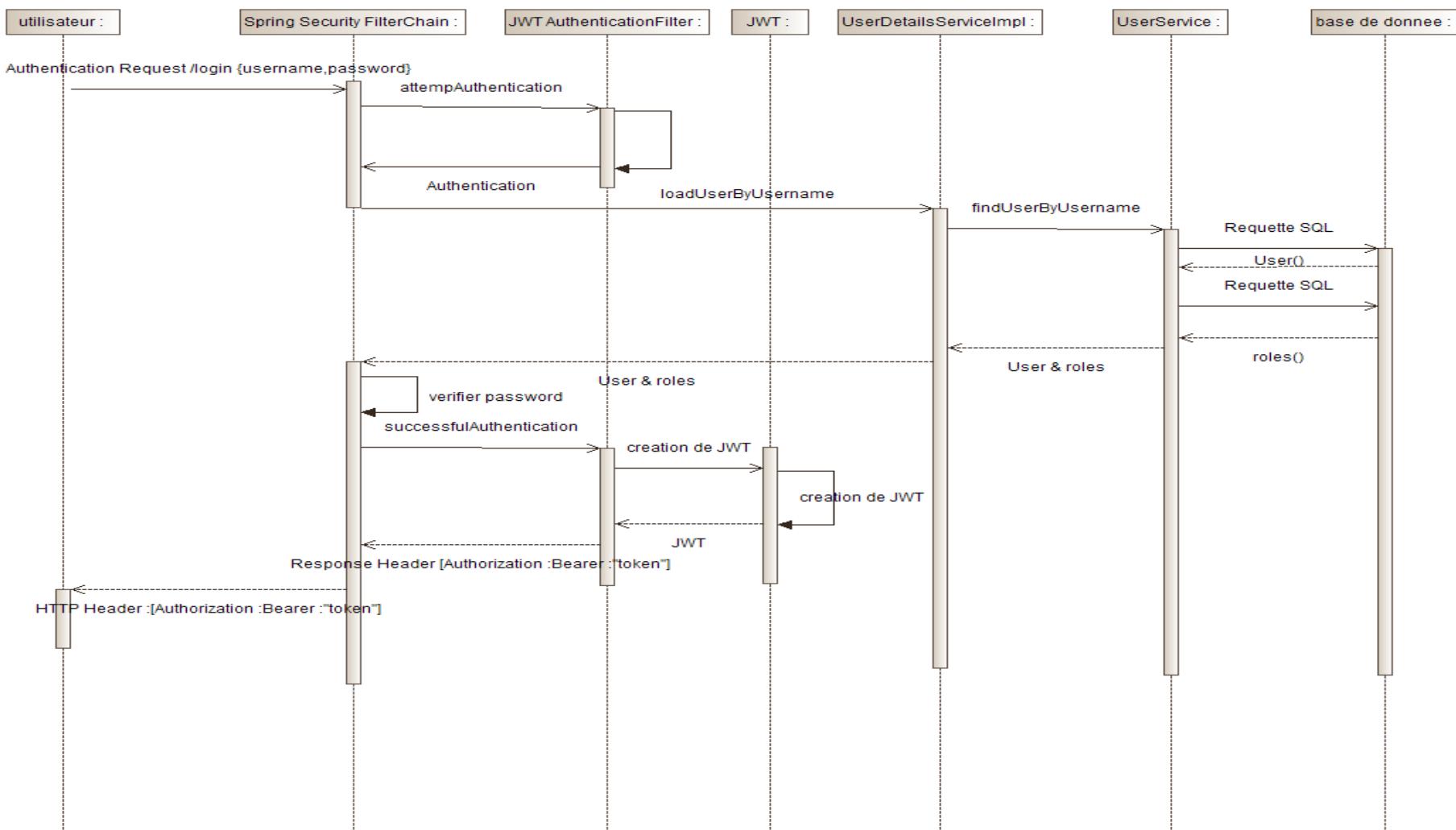
Le token est envoyé avec chaque requête que le client fera auprès de l'application, qui autorisera, ou non, le client à accéder à ses services, suivant la

validité du token. Ce type d'authentification, ne stocke pas les sessions utilisateurs dans le contexte de l'application.

Pour utiliser notre token, il faut tout d'abord le créer. Pour cela, il est nécessaire de s'authentifier avec son username et son mot de passe auprès de l'application afin que celle-ci nous renvoie le token. Une fois le token obtenu, on peut faire appel à nos URL sécurisées en envoyant le token avec notre requête. La méthode la plus courante pour envoyer le token est de l'envoyer à travers l'en-tête HTTP Authorization en tant que Bearer token :

[Authorization : Bearer 'token']

Le diagramme de séquence « Authentification et Autorisation » présente le séquencement détaillé des interactions entre l'utilisateur, Spring Security (FilterChain, JWTAuthenticationFilter), l'api JWT et la base de donnée.

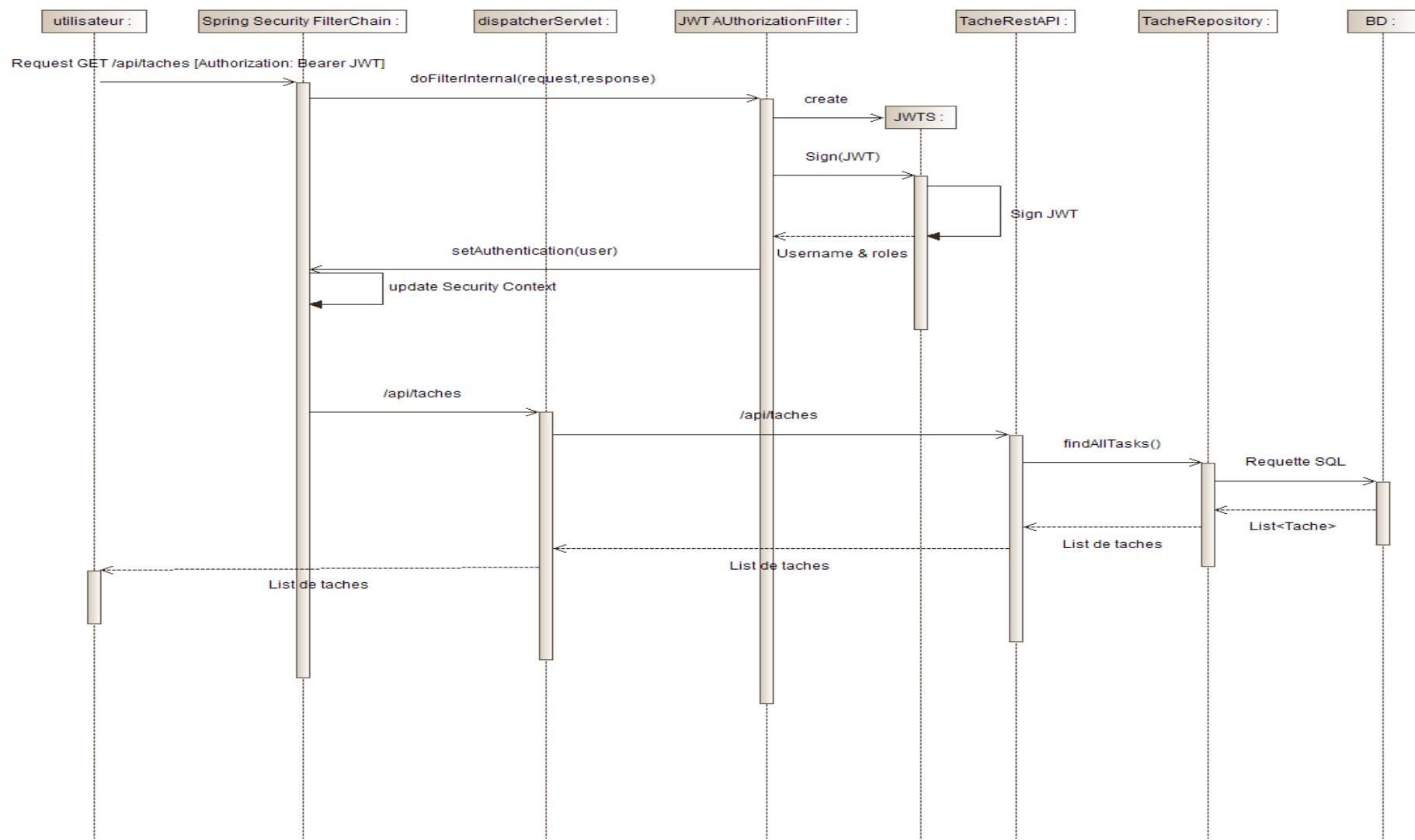
**Figure 13.** Diagramme de séquence « Authentification et Autorisation »

II.5.2 *Diagramme de séquence « Consulter la liste des taches »*

Pour traiter le token, on utilise un filtre qui va l'extraire du header, le valider puis ajouter au contexte de Spring une authentification correspondant à l'utilisateur pour lequel le token a été émis : notre utilisateur est authentifié pour le reste de sa requête.

Le filtre est sans doute la partie essentielle de notre chaîne d'authentification. Il va intercepter toutes les requêtes et vérifier la présence d'un JWT, et va ensuite valider le token et récupérer l'objet "authentication" pour l'ajouter au contexte de spring-security. Notre client sera donc authentifié pour la suite de l'exécution de sa requête. À la fin du processus, on supprime l'authentification du contexte.

Le diagramme de séquence « Consulter la liste des taches » présente le séquencement détaillé des interactions entre l'utilisateur, Spring Security (FilterChain, JWTAuthenticationFilter), l'api JWT et la base de donnée.

**Figure 14.**Diagramme de séquence «Consulter la liste des taches »

II.6. Spécification non fonctionnelle

Ce sont des exigences qui ne concernent pas spécifiquement le comportement du système mais plutôt identifient des contraintes internes et externes du système.

Les principaux besoins non fonctionnels de notre application se résument dans les points suivants :

- *Sécurité* : l'application doit respecter la confidentialité des données.
- *Performance* : l'application devra être performante c'est-à-dire que le système doit réagir dans un délai précis quel que soit l'action de l'utilisateur.
- *Extensibilité* : l'application doit être ferme à la modification ouverte à l'extension c'est-à-dire qu'il pourra y avoir une possibilité d'ajouter de nouvelles fonctionnalité.

Conclusion

Ce chapitre a été consacré à la conception de la solution et le modèle d'analyse à partir de l'approche statique à travers le diagramme de cas d'utilisation et de classes et l'approche dynamique à travers les diagrammes de séquence.

Dans le chapitre suivant nous détaillons les étapes de la réalisation de l'application.

Chapitre III. Réalisation

Introduction

Après avoir élaboré la conception de notre application, nous abordons dans ce chapitre le dernier volet de ce rapport, qui a pour objectif d'exposer la phase de réalisation.

La phase de réalisation est considérée comme étant la concrétisation finale de toute la méthode de conception.

Nous menons tout d'abord une étude technique où nous décrivons les ressources logicielles utilisées dans le développement de notre projet. Nous présentons en premier lieu notre choix de l'environnement de travail, où nous spécifions l'environnement logiciel qu'on a utilisé pour réaliser notre application puis nous détaillons l'architecture, aussi nous présentons quelques interfaces réalisées pour illustrer le fonctionnement de quelques activités du système.

III.1. Environnement de travail

L'implémentation consiste la phase d'achèvement et d'aboutissement du projet. Pour accomplir cette tâche, il faut utiliser des outils adéquats à la réalisation du projet. Ces outils représentent l'environnement de travail.

III.1.1 *Environnement logiciel*

Concernant l'environnement logiciel de notre projet, nous avons utilisé parmi les outils et logiciels ce qui suit



Est un outil d'administration de base de données possédant un éditeur SQL et un constructeur de requête. Il a été développé et optimisé pour être utilisé avec le SGBD relationnel MySQL disponible commercialement ou gratuitement.

Le logiciel est devenu un projet libre en 2006 sous le nom de Heidi SQL. Heidi SQL est capable de se connecter à des bases MySQL, SQL Server ainsi que PostgreSQL.



Est un outil de modélisation UML disponible sur les plates-formes Windows, Linux et Mac. Il intègre également la modélisation BPMN, et le support de la modélisation des exigences, du dictionnaire, des règles métier et des objectifs.



Est un client REST proposé par Google. Il est disponible sous la forme d'une extension Chrome ou bien d'une application stand-alone. Parmi les nombreuses solutions pour interroger ou tester web services et API, Postman propose de nombreuses fonctionnalités, une prise en main rapide et une interface graphique agréable.



C'est un projet de la Fondation Eclipse visant à développer tout un environnement de développement libre, extensible, universel et polyvalent.

Son objectif est de produire et fournir divers outils gravitant autour de la réalisation de logiciel, englobant les activités de codage logiciel proprement dites (avec notamment un environnement de développement intégré) mais aussi de modélisation, de conception, de test, de reporting, etc. Son environnement de développement notamment vise à la généricité pour lui permettre de supporter n'importe quel langage de programmation.



Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et OS X.

Présenté lors de la conférence des développeurs Build d'avril 2015 comme un éditeur de code cross-Platform, open source et gratuit, supportant une dizaine de langages comme Type Script.

III.1.2 *Outils de développement et SGBD*

➤ *PostgreSQL*

PostgreSQL est un système de gestion de base de données relationnelle et objet (SGBDRO). C'est un outil libre disponible selon les termes d'une licence de type BSD.

Ce système est concurrent d'autres système de gestion de base de données, qu'ils soient libres (comme MySQL), ou propriétaires (Comme Oracle et Microsoft SQL Server).

➤ *Java*

Le langage java, développé par Sun, est un langage orienté objet, en effet il possède un mécanisme qui permet de décrire les caractéristiques d'un objet de façon unique et de pouvoir lui faire subir des opérations.

Java Entreprise Edition JEE, qui peut être considéré comme une extension de java, est un ensemble de spécifications destinées aux applications d'entreprises. Ce langage permet la création d'applications performantes et robustes.

JEE s'appuie sur le modèle MVC (Model Vue Contrôleur).

➤ *JavaScript*

JavaScript est un langage de programmation qui permet d'implémenter des mécanismes complexes sur une page web. À chaque fois qu'une page web fait plus que simplement afficher du contenu statique — afficher du contenu mis à jour à des temps déterminés, des cartes interactives, etc... — JavaScript a de bonnes chances d'être impliqué. C'est la troisième couche des technologies standards du web, les deux premières (HTML et CSS).



➤ *Le Framework Angular*

Angular est un Framework JavaScript côté client qui permet de réaliser des applications de type "Single Page Application". Il est basé sur le concept de l'architecture MVC (Model View Controller) qui permet de séparer les données, les vues et les différentes actions que l'on peut effectuer.

Le code source d'Angular est écrit en Type Script. Le Type Script est une couche supérieure au JavaScript développée par Microsoft qui se compile en JavaScript simple. Étant un langage typé, il permet de créer des classes, des variables, des signatures de fonction et l'utilisation de modules. Il est important de noter que l'utilisation du Type Script est facultative, on peut tout à fait utiliser du JavaScript dans un fichier Type Script.



➤ *Le Framework Spring*

Est un Framework très riche permettant de structurer, d'améliorer et de simplifier l'écriture d'application Java EE.

Spring est un Framework libre, un conteneur dit « léger », c'est à dire une infrastructure similaire à un serveur d'application Java EE.

Spring s'appuie principalement sur l'intégration de trois concepts clés :

- L'inversion de contrôle est assurée de deux façons différentes : la recherche de dépendances et l'injection de dépendances, cette injection peut être effectuée de trois manières possibles :
 - ❖ L'injection de dépendance via le constructeur.
 - ❖ L'injection de dépendance via les modificateurs (setters).
 - ❖ L'injection de dépendance via une interface.
- Une couche d'abstraction : La couche d'abstraction permet d'intégrer d'autres Framework et bibliothèques avec une plus grande facilité. Cela se fait par l'apport ou non de couches d'abstraction spécifiques à des Framework particuliers. Il est ainsi possible d'intégrer un module d'envoi de mails plus facilement.
- La programmation orientée aspect

✓ *Spring Boot*

Est un micro-Framework créé par l'équipe de chez Pivotal, conçu pour simplifier le démarrage et le développement de nouvelles applications Spring. Le Framework propose une approche dogmatique de la configuration, qui permet d'éviter aux développeurs de redéfinir la même configuration à plusieurs endroits du code. Dans ce sens, Boot se veut d'être un acteur majeur dans le secteur croissant du développement d'applications rapide.

✓ *Spring Data*

C'est un projet supplémentaire de Spring créé il y a quelques années pour répondre aux besoins d'écrire plus simplement l'accès aux données et d'avoir une couche d'abstraction commune à de multiples sources de données.

Spring Data s'interface avec plusieurs sources de données parmi lesquelles JPA, Neo4j, Mongo DB, REST et quelques autres.

✓ *Spring Security*

Est un module incontournable d'une application développée en Spring. Il apporte tout le nécessaire pour sécuriser une application et il a l'avantage d'être vraiment personnalisable. La notion de sécurité informatique n'est pas une mince

affaire et sa mise en place est parfois longue et demande d'être constamment adaptée au niveau réseau, serveurs, application... Spring Security n'intervient que sur le domaine applicatif.

✓ *Spring MVC*

Le Framework Spring offre une implémentation innovante du patron MVC par le biais d'un Framework nommé Spring MVC, qui profite des avantages de l'injection de dépendances et qui, depuis la version 2.5, offre une intéressante flexibilité grâce aux annotations Java 5. Ce module permet dès lors de s'abstraire de l'API Servlet de Java EE, les informations souhaitées étant automatiquement mises à disposition en tant que paramètres des méthodes des contrôleurs.

De plus, à partir de la version 3.0, Spring MVC intègre un support permettant de gérer la technologie REST, les URL possédant la structure décrite par cette dernière étant exploitable en natif.



➤ *Le Framework Hibernate*

Est un Framework de persistance utilisé pour gérer la persistance des objets java dans une base de données, il peut être utilisé dans un développement web ou bien un développement client lourd.

Hibernate peut utiliser SQL comme il peut utiliser son propre langage de requête HQL (Hibernate Query Language).



➤ *Le Framework Bootstrap*

Est un Framework développé par l'équipe du réseau social Twitter. Proposé en open source (sous licence MIT), ce Framework utilisant les langages HTML, CSS et JavaScript fournit aux développeurs des outils pour créer un site facilement. Ce Framework est pensé pour développer des sites avec un design responsive, qui s'adapte à tout type d'écran, et en priorité pour les smartphones. Il fournit des outils avec des styles déjà en place pour des typographies, des boutons, des interfaces de navigation et bien d'autres encore. On appelle ce type de Framework un "Front-End Framework".



➤ *Git et GitHub*

GitHub Est une plateforme open source de gestion de versions et de collaboration destinée aux développeurs de logiciels. La solution GitHub a été lancée en 2008. Elle repose sur Git, qui permet de stocker le code source d'un projet et de suivre l'historique complet de toutes les modifications apportées à ce code.

III.2. Architecture

L'architecture est l'ensemble des aspects techniques et applicatifs qui sont importants pour un logiciel. Les choix architecturaux influent sur la réussite ou l'échec d'un projet. Nous exposons d'abord l'architecture technique cible de la solution ainsi que l'architectures applicative.

III.2.1 Architecture technique

L'architecture technique est l'environnement technique permettant l'exécution des composants informatiques et les échanges de données. Pour notre application, nous proposons l'architecture cible suivante

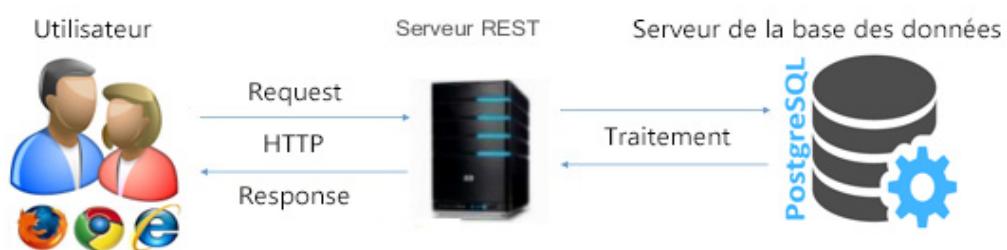


Figure 15. Architecture technique de l'application

L'utilisateur se connecte à l'application à travers un navigateur web. Le navigateur permet d'envoyer des requêtes au serveur REST et d'en interpréter la réponse. Le navigateur et le serveur REST communiquent en utilisant le protocole http. Le serveur REST traite les requêtes HTTP, interprète et exécute le code de l'application, puis génère une réponse qu'il renvoie au navigateur de l'utilisateur. Le système de gestion de bases de données PostgreSQL permet d'interroger les données et de les mettre à jour.

III.2.2 Architecture applicative

Nous pouvons définir l'architecture applicative comme une organisation des données et des traitements qui mettent en œuvre les fonctions métiers. La figure 16 présente l'architecture applicative de la solution avec les Framework utilisés.

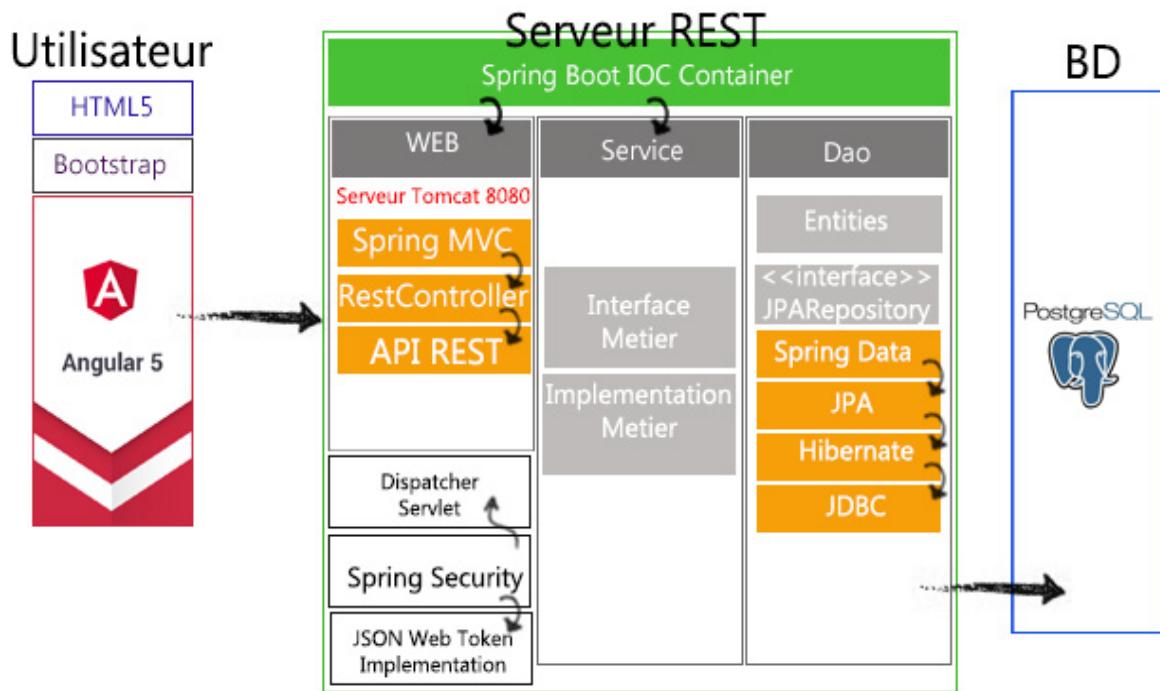


Figure 16. Architecture applicative de l'application

III.3. SOA - Architecture Orientée Service

L'architecture orientée services (SOA, Service-Oriented Architecture) est une approche permettant de créer une architecture qui s'appuie sur l'utilisation de services. Ces services (les services Web RESTful, par exemple) remplissent de petites fonctions, telles que la production de données, la validation d'un client ou la mise à disposition d'analyses simples.

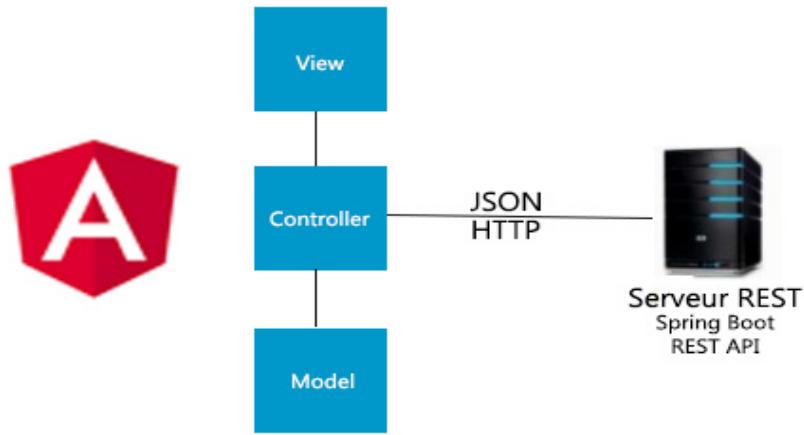


Figure 17. Architecture SOA

III.3.1 **Le notion service Web**

La technologie des services Web est un moyen rapide de distribution de l'information entre clients et leurs différentes plates-formes. Les services Web sont basés sur le modèle SOA.

La spécificité des Web Services est l'utilisation de HTTP comme support des messages entre clients et serveur.

III.3.2 **REST**

Est l'acronyme de Representational State Transfer, 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.

III.3.3 **Représentation des ressources : JSON**

JSON est l'acronyme de JavaScript Object Notation. C'est un format texte qui permet de représenter des données et de les échanger facilement.

Que ce soit au niveau du serveur ou au niveau des clients, une API RESTful manipule des ressources par une représentation de celles-ci.

Le principe en pratique est de passer de la représentation utilisée en interne, que ce soit sur le client ou le serveur, à la représentation utilisée dans le message HTTP, requête ou réponse.

Deux types de structures sont disponibles pour décrire le modèle objet de JavaScript :

- *Objet* : une collection de pair nom/valeur
- *Tableau* : une liste ordonnée de valeurs.

Les valeurs peuvent être des types suivants : booléen, chaîne de caractères, nombre, ou valeur nulle. (boolean, string, number, null).

III.4. API REST

Ci-dessous un tableau qui illustre les API s'appuyant sur les principes REST créer dans notre application

URI	Méthode	Sémantique
<i>API REST :Utilisateur</i>		
http://localhost:8080/api/users	GET	Récupérer la liste des utilisateurs
http://localhost:8080/api/users/ id	GET	Récupérer la représentation de L'utilisateur identifié par id
http://localhost:8080/api/regis ter	POST	Inscription d'un nouvel utilisateur
http://localhost:8080/api/users/ statistique	GET	Récupérer le nombre de compte d'utilisateur désactiver
http://localhost:8080/api/users/ JWT/username	GET	Récupérer la représentation de L'utilisateur identifié par son username JWT

http://localhost:8080/api/users/ totale	GET	Récupérer le nombre totale de compte d'utilisateur, manager et administrateur .
http://localhost:8080/api/users/ roleUser	GET	Récupérer la liste des utilisateurs avec le Rôle user
http://localhost:8080/api/users/ roleManager	GET	Récupère la liste des utilisateurs avec le Rôle manager
http://localhost:8080/api/users/ manager	POST	Création d'un nouvel compte manager
<i>API REST :Tache</i>		
http://localhost:8080/api/tasks	GET	Récupérer la liste des taches
http://localhost:8080/api/tasks/ id	GET	Récupère la représentation de tache identifié par id
http://localhost:8080/api/tasks/ id	PUT	Modifier une tache
http://localhost:8080/api/tasks/ encours	GET	Récupère la liste des taches en cours
http://localhost:8080/api/tasks/ comment	POST	Ajouter un commentaire
http://localhost:8080/api/tasks/ totale	GET	Récupérer le nombre totale de tâche .
http://localhost:8080/api/tasks/ total/archive	GET	Récupérer le nombre totale de tache archive.
http://localhost:8080/api/tasks/ auth/username	GET	Récupérer les taches d'un utilisateur authentifier par son username JWT
http://localhost:8080/api/tasks/ comment/id	GET	Récupérer la liste des commentaire d'une tache par son identifiant id
http://localhost:8080/api/tasks/ total/username	GET	Récupérer le nombre totale des taches d'un utilisateur par son username JWT

http://localhost:8080/api/tasks/archives	GET	Récupérer le nombre totale des taches archive.
http://localhost:8080/api/tasks/archive/id	PUT	Archiver une tâche par son identifiant id
<i>API REST :Evènement</i>		
http://localhost:8080/api/events	GET	Récupérer la liste des évènements
http://localhost:8080/api/events	POST	Création d'un nouvel évènement
http://localhost:8080/api/events/id	GET	Récupérer la représentation d'un évènement identifié par id
http://localhost:8080/api/events/id	PUT	Modifier un évènement
http://localhost:8080/api/events/id	DELETE	Supprimer un évènement
http://localhost:8080/api/events/totale	GET	Récupérer le nombre totale des évènement
http://localhost:8080/api/events/archive/totale	GET	Récupérer le nombre totale des évènement archive
http://localhost:8080/api/events/auth/username	GET	Récupérer la listes des évènements d'un utilisateur authentifier par son username JWT
http://localhost:8080/api/events/archive/id	PUT	Archiver un évènement par son identifiant id
http://localhost:8080/api/events/totals/archives	GET	Récupérer le nombre totale des évènement archive.
http://localhost:8080/api/events/archive/username	GET	Récupérer le nombre totale des évènement archive d'un utilisateur par son username JWT
<i>API REST : Document</i>		

http://localhost:8080/api/documents/upload	POST	Uploader un document sur le serveur
http://localhost:8080/api/documents	POST	Ajouter un document
http://localhost:8080/api/documents/public	GET	Récupérer la listes des documents public
http://localhost:8080/api/documents/prive	GET	Récupérer la listes des documents prive
http://localhost:8080/api/documents	GET	Récupère la liste des document
http://localhost:8080/api/documents/totale	GET	Récupère le nombre totale des documents
http://localhost:8080/api/documents/totale/archive	GET	Récupère le nombre totale des documents archive
http://localhost:8080/api/documents/totale/archives	GET	Récupère la listes des documents archive
http://localhost:8080/api/documents/filename	GET	Récupère un document par son nom
http://localhost:8080/api/documents/archive/id	PUT	Archiver un document par son identifiant id
<i>API REST : Attachement</i>		
http://localhost:8080/api/attachments/upload	POST	Upload d'un attachement sur le serveur
http://localhost:8080/api/attachments	POST	Ajouter un attachement
http://localhost:8080/api/attachments/tasks/id	GET	Récupère la listes attachements par son identifiant id
http://localhost:8080/api/attachments	DELETE	Supprimer un attachement
<i>API REST : Profil</i>		
http://localhost:8080/api/profil/s/compte/id	PUT	Modifier le compte d'un utilisateur par son identifiant id

http://localhost:8080/api/profil s/profil/id	PUT	Modifier le profil d'un utilisateur par son identifiant id
http://localhost:8080/api/profil s/password/id	PUT	Modifier le mot de passe d'un utilisateur par son identifiant id
http://localhost:8080/api/profil s/JWT/username	PUT	Récupère le profil d'un utilisateur par son username JWT

Tableau 9. *Description des API REST de l'application*

III.5. Présentation des interfaces

Durant cette sous activité, nous présentons quelques interfaces de notre application réalisée.

III.5.1 *Interface d'authentification*

La figure 18 : présente l'interface d'Authentification qui est la première page qui s'affiche à l'utilisateur. Cette interface est décomposée de deux champs (username et mot de passe) qui permet d'accéder à l'application.

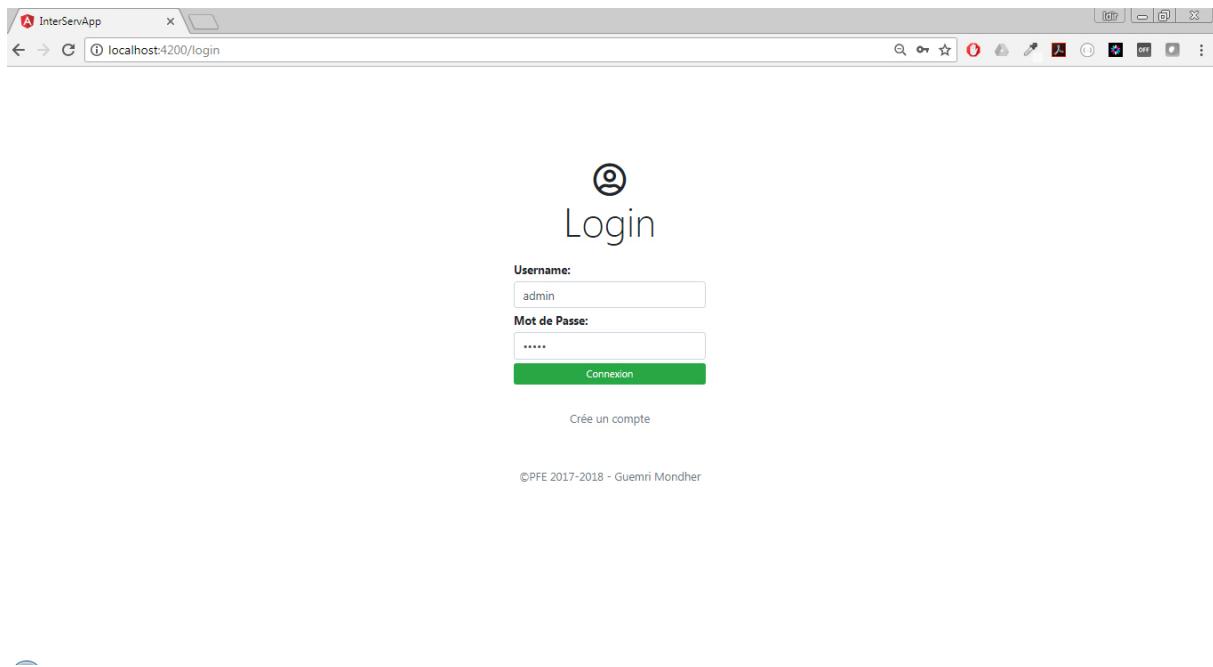


Figure 18. Page d'authentification

III.5.2 Interface d'inscription

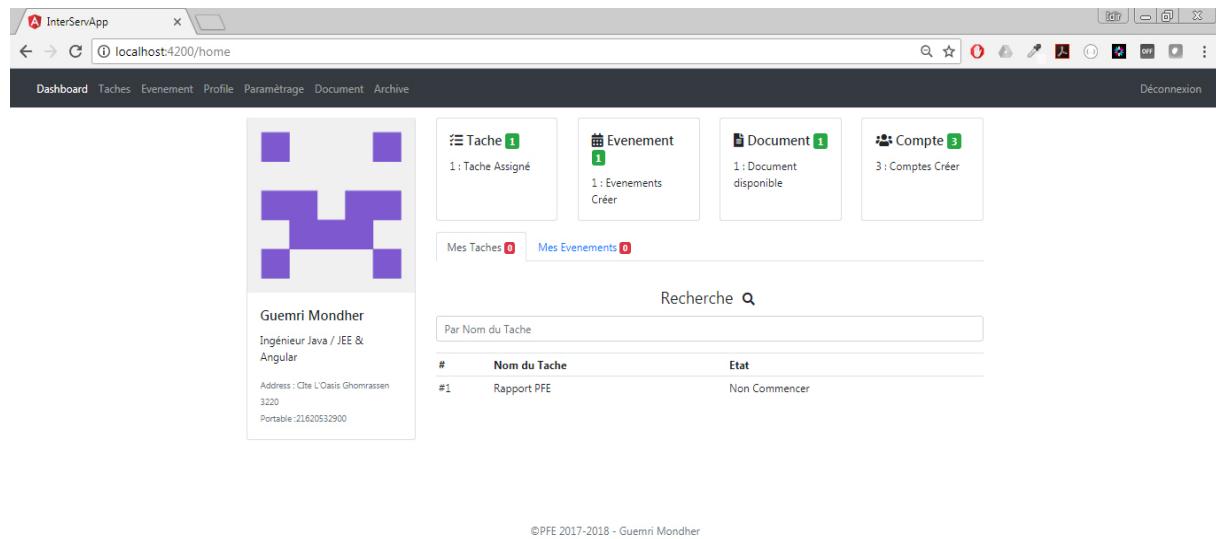
La figure 19 : présente l'interface d'inscription d'un utilisateur. Chaque nouvel utilisateur est appelé à s'inscrire en remplissant le formulaire, après la validation de son inscription il recevra un email qui lui indique l'activation du compte.

The screenshot shows a web browser window titled "InterServApp" with the URL "localhost:4200/register". The page is titled "Créer un Compte". It contains several input fields with placeholder text: "Nom : Guemri", "Prenom : Mondher", "Username : idir", "Email : mondher25@gmail.com", and "Mot de Passe :". Below the inputs is a green "Créer" button. At the bottom left, there is a link "J'ai déjà un compte". The browser toolbar includes icons for search, refresh, and other common functions.

Figure 19. Page d'inscription

III.5.3 Interface d'accueil

La figure 20 : présente l'interface d'accueil. Cette interface est partagée par tous les acteurs de notre application (Utilisateur, Manager, Administrateur), est la première page qui s'affiche après la page d'authentification.

**Figure 20.** Page d'accueil

III.5.4 Interface d'ajout d'une tache

La figure 21 : présente l'interface d'ajout d'une nouvelle tâche. Cette interface permet au manager et à l'administrateur d'assigner une tâche à un utilisateur.

L'état d'une tâche créé est par défaut « Non Commencer ».

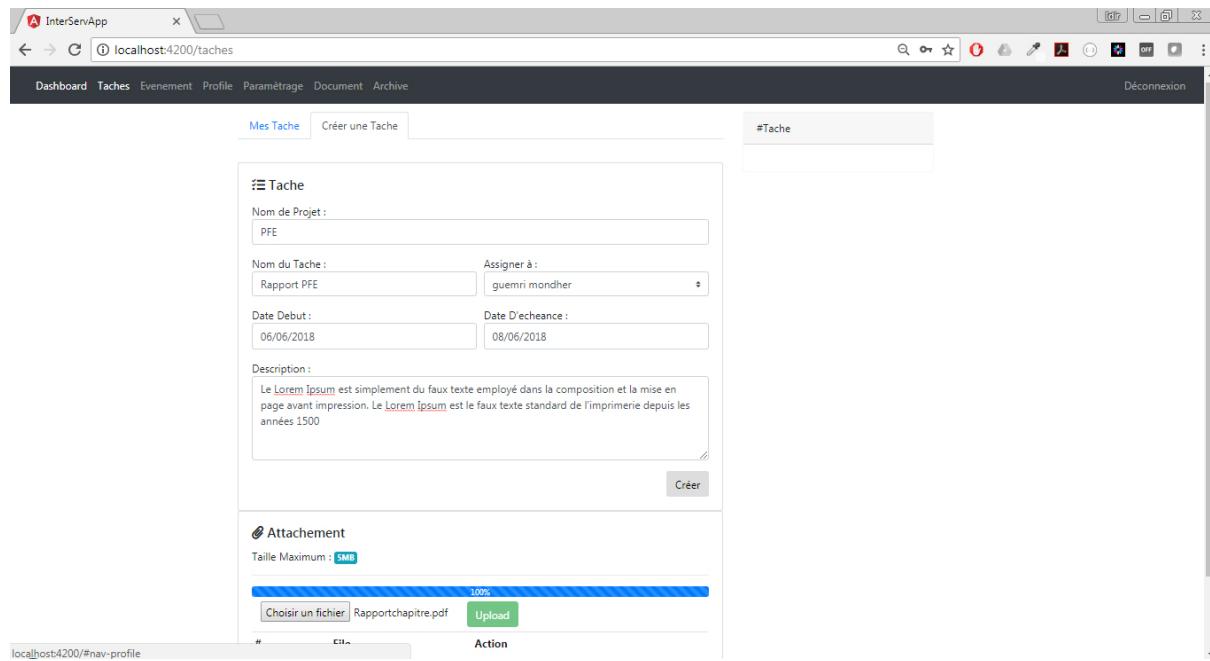


Figure 21. Page d'ajout d'une nouvelle tache

III.5.5 Interface de modification d'une tache

La figure 22 : présente l'interface de modification d'une tâche. Cette interface permet au manager et à l'administrateur de modifier les informations d'une tache et de changer l'état d'une tache (valider, annuler, pour révision).

Un utilisateur n'a le droit de modifier que l'état de tache (en cours, terminer)

The screenshot shows a web-based application window titled "InterServApp". The URL in the address bar is "localhost:4200/detailTache/41". The top navigation bar includes links for Dashboard, Taches, Evenement, Profile, Paramétrage, Document, and Archive, along with a "Déconnexion" button. The main content area is divided into two sections: "Edit Tache" and "Attachment".
In the "Edit Tache" section, there are fields for "Nom de Projet" (PFE), "Nom du Tache" (Rapport PFE), "Etat" (Validé), "Date Début" (06/06/2018), "Date D'échéance" (08/06/2018), and a "Description" text area containing placeholder text about Lorem Ipsum.
In the "Attachment" section, it shows a maximum file size of 5MB and a table with one entry:

#	File	Action
#1	Rapportchapitre.pdf	[Delete]

Figure 22. Page modifier une tache

III.5.6 Interface Commentaire

La figure 23 : présente l'interface d'ajout et d'affichage d'un commentaire à une tache déjà créée.

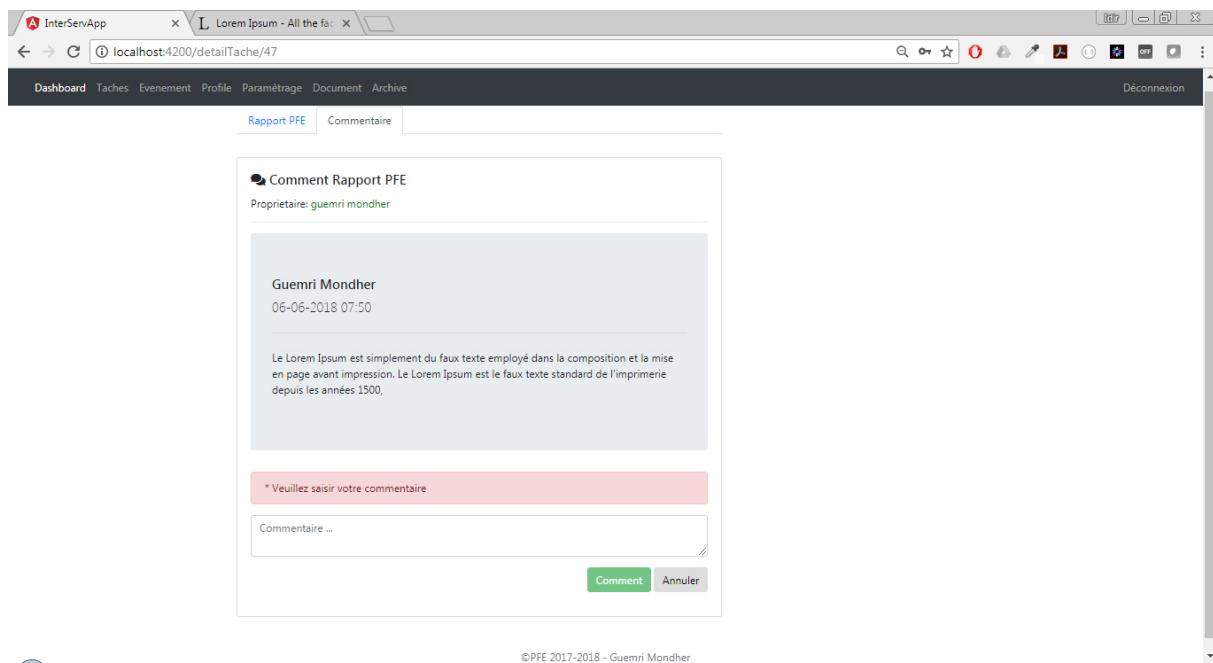


Figure23. *Page commentaire*

III.5.7 Interface Listes des taches

La figure 24 : présente l'interface de la liste de taches assigné a des utilisateurs et contient une barre de recherche et une journalisation des tous les tâche par date de création.

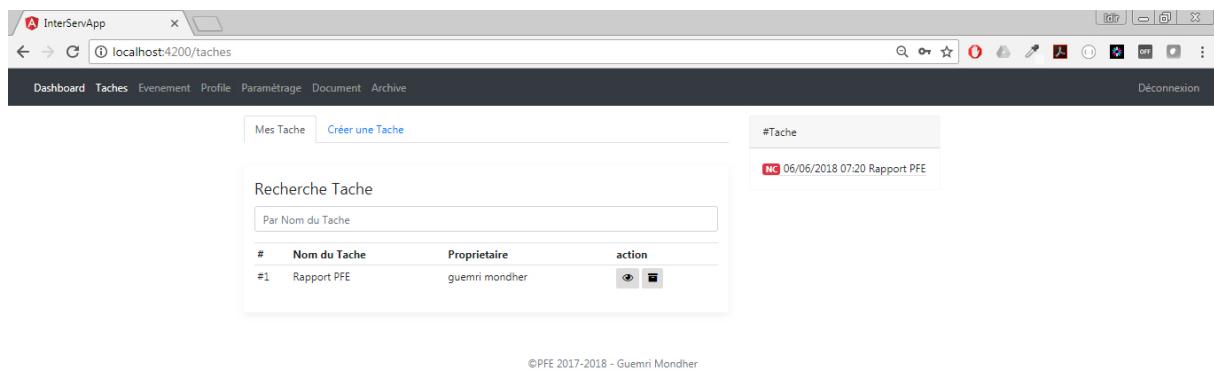


Figure 24. Page listes des taches

III.5.8 Interface Ajout d'un Evènement

La figure 25 : présente l'interface d'ajout d'un nouvel évènement. Tous nouvel évènement est créé avec un état « En Attente ».

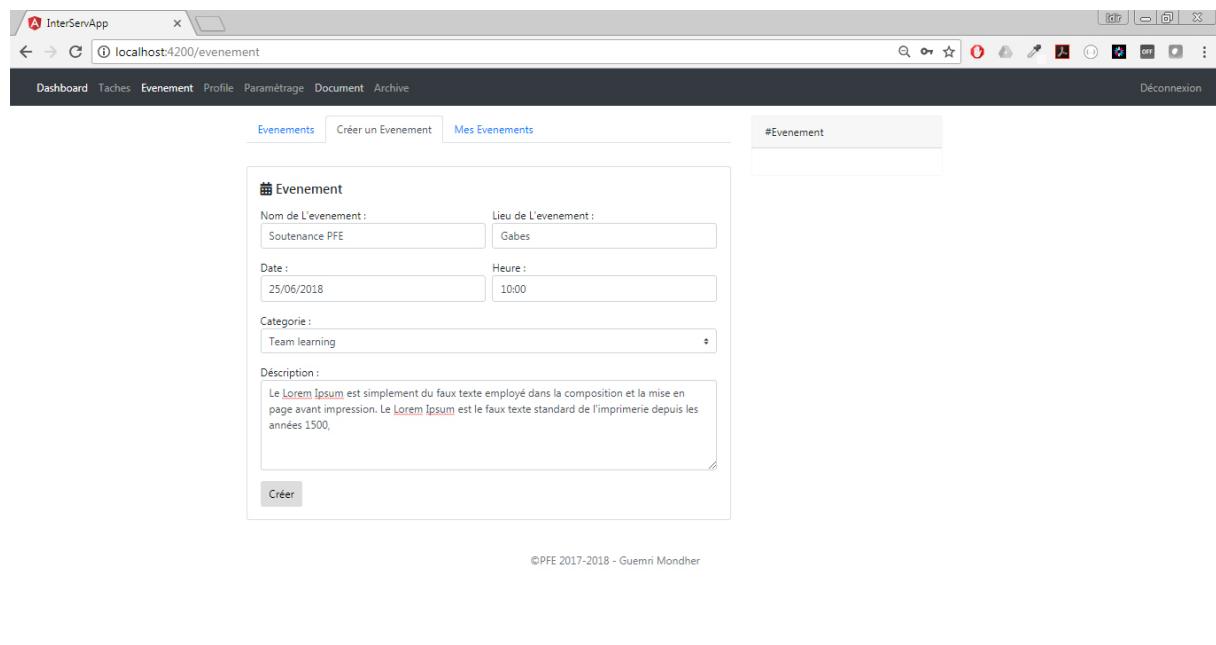


Figure 25. Page d'ajout d'un évènement

III.5.9 Interface de modification d'un Évènement

La figure 26 : présente l'interface de modification d'un évènement.

Cette interface est partagée par tous les acteurs mais seul le manager et l'administrateur qui peuvent valider un évènement en changeant l'état de l'évènement (reporter, confirmer, annuler).

The screenshot shows a web application window titled "InterServApp" with the URL "localhost:4200/detailEvent/45". The top navigation bar includes links for Dashboard, Tâches, Événement, Profile, Paramétrage, Document, and Archive, along with a "Déconnexion" button. The main content area is titled "Edit Evenement" and displays the following form fields:

- Nom de l'événement: Soutenance PFE
- Lieu de l'événement: Gabes
- Date: 25/06/2018
- Heure: 10:00
- Catégorie: Team learning
- Etat: Confirmé

A description text area contains the following placeholder text: "Le Lorem Ipsum est simplement du faux texte employé dans la composition et la mise en page avant impression. Le Lorem Ipsum est le faux texte standard de l'imprimerie depuis". At the bottom of the form are two buttons: "Modifier" (in green) and "Retour".

Figure 26. Page modifier un évènement

III.5.10 **Interface d'ajout d'un manager**

La figure 27 : présente l'interface d'ajout d'un manager. Cette interface est dédiée seulement à l'administrateur de l'application.

Le mot de passe fournit au manager est générer automatiquement.

The screenshot shows a web-based application interface titled 'InterServApp'. The URL in the address bar is 'localhost:4200/parametrazione'. The top navigation bar includes links for Dashboard, Taches, Evenement, Profile, Paramétrage (selected), Document, and Archive, along with a Déconnexion link. The main content area has tabs for Utilisateur, Manager (selected), and Nouveaux Compte. The 'Compte Manager' section contains fields for Nom (benchaaben), Prenom (Mohamed), Username (mohamed), Email (benchaaben.mohamed@gmail.com), and Mot de passe (FvS8DE65). A 'Créer' button is at the bottom. To the right, a sidebar titled '#Paramétrage' shows 'Compte Utilisateur' with a note: '1 : compte(s) désactivé'. The footer of the page includes the text '©PFE 2017-2018 - Guemri Mondher'.

Figure 27. Page d'ajout d'un manager

III.5.11 **Interface d'Activation de compte Utilisateur**

La figure 28 : présente l'interface de modification de compte utilisateur.

Pour qu'un utilisateur puisse accéder à l'application son compte doit être activer par un envoi d'un email d'activation à l'adresse mail saisie lors de l'inscription.

Cette tâche est effectuée par un manager ou un administrateur.

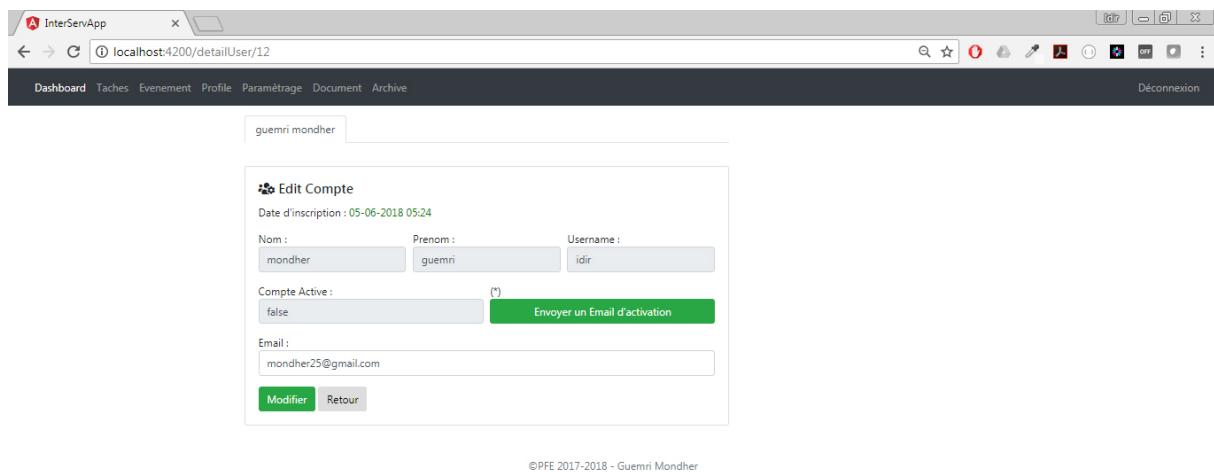


Figure 28. Page d'activation de compte utilisateur

III.5.12 Interface d'ajout d'un document :

La figure 29 : présente l'interface d'ajout d'un document. Il est possible d'ajouter un document privé ou bien public

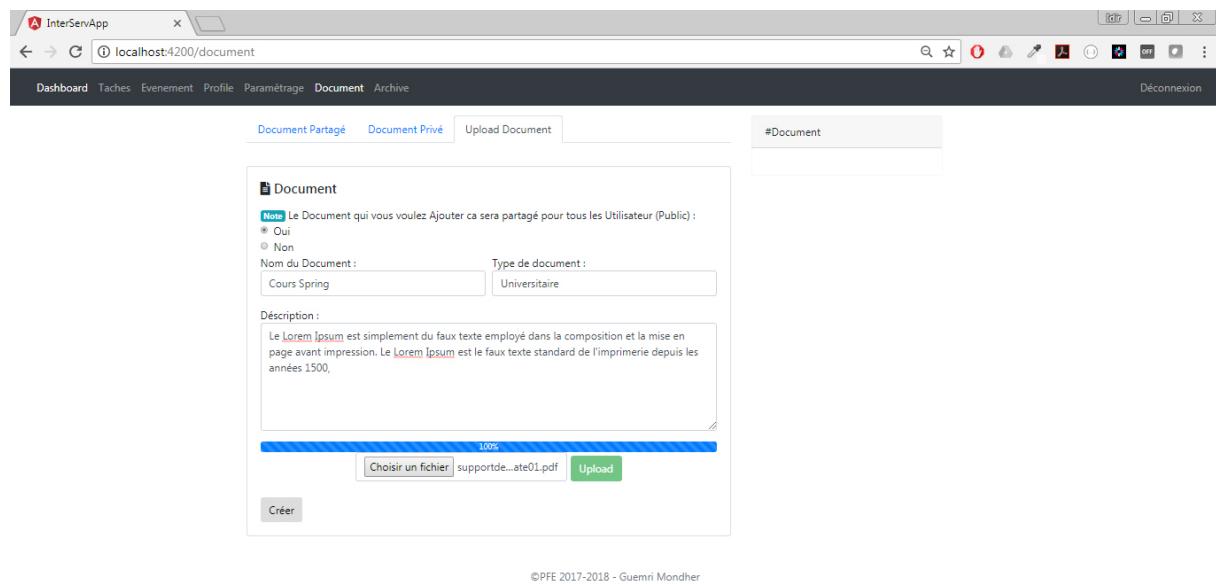


Figure 29. Page d'ajout d'un document

III.5.13 **Interface Profil**

La figure 30 : présente l'interface profil qui contient les informations sur les compte de l'utilisateur connecté ainsi la listes des tous les contacts.

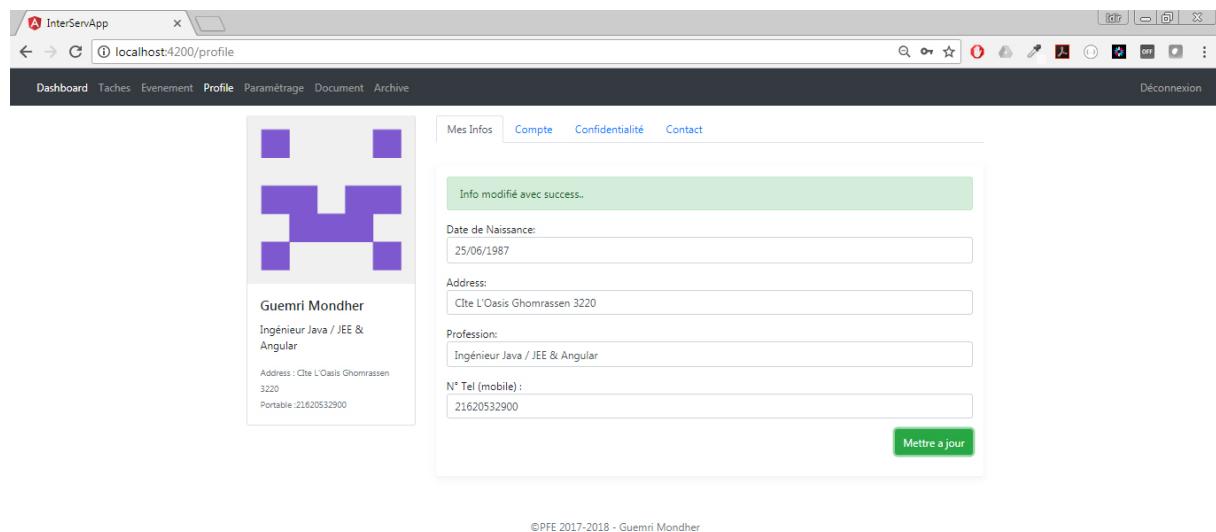


Figure 30. *Page profil*

III.5.14 *Interface d'Archive*

La figure 31 : présente l'interface d'archive des évènements, des taches et des documents. Cette interface est réservée au manager et à l'administrateur.

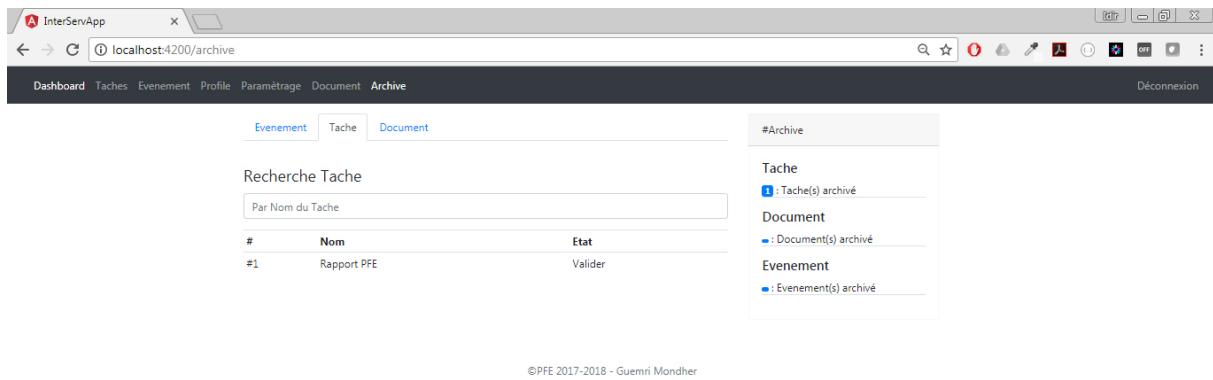


Figure 31. Page d'archive

Conclusion

A travers ce dernier chapitre, nous avons présenté, tout d'abord, l'architecture de l'environnement matériel et logiciel de notre projet, ainsi que le choix du langage de développement. Ensuite, nous avons illustré quelques scenarios de ce travail à travers des captures d'écran témoignant des différentes interfaces que contient notre application.

Conclusion Générale

Dans le cadre de notre projet de fin d'études, nous avons conçu et développé une application inter service multiplateforme modulaire. Le présent rapport couvre l'étude, la conception et l'implémentation de la solution.

Ce projet m'a permis de mettre en pratique mon esprit d'étude, d'analyse et de critique. De mettre en application certaines de nos connaissances et notre savoir acquis lors de la période de la formation à l'ESSAT.

Sur un plan plus technique, nous avons découvert le monde du développement web dans un langage que nous connaissions déjà. Néanmoins, la spécificité de JEE fait que nous avons dû apprendre toutes les bases de cette nouvelle architecture ce qui nous a pris plus de temps que nous ne l'espions. Finalement, une fois maîtrisé, nous nous sommes rendu compte de la puissance de l'outil et avons compris pourquoi, pour des sites de grosse audience nécessitant de fréquentes mises à jour et proposant des services complexes, le mode est aujourd'hui au JEE plutôt qu'à un langage comme PHP.

Nous avons pu apprendre les bases des Framework les plus couramment utilisés tels que Spring, Angular et hibernate.

Finalement, notre travail ne s'arrête pas à ce niveau, en effet il est possible d'ajouter des nouvelles fonctionnalités à notre application, citons comme exemple la gestion des notifications et le chat.

Bibliographie et Webographie

- [1] Angular, <https://angular.io/> Consulté le 06/06/2018
- [2] Hibernate, <http://www.hibernate.org/> consulté le 06/06/2018
- [3] Spring, <https://www.spring.io/> consulté le 06/06/2018
- [4] D. Alur, J. Crupi, D. Malks, Core J2EE™ Patterns: Best Practices and Design Strategies, Second Edition
- [5] M. Fowler, InversionOfControl, <http://martinfowler.com/bliki/InversionOfControl.html/> consulté le 25/05/2015
- [6] PostgreSQL, <http://www.postgresql.org/> consulté le 01/06/2018
- [7] Heidisql, <https://www.heidisql.com/> consulté le 30/05/2018
- [8] Spring Security, <https://www.linkedin.com/pulse/json-web-token-jwt-spring-security-real-world-example-boris-trivic/> consulté le 15/05/2018
- [9] Planon Gestion de travail, <https://planonsoftware.com/> consulté le 10/05/2018
- [10] Divers Tutorial, <http://javasampleapproach.com> consulté le 10/05/2018

Glossaire

Base de données relationnelle : Dans laquelle les données sont stockées dans des tables, selon le modèle introduit par Codd. La base de données consiste en deux ou plusieurs tables reliées par des relations.

Cette base de données est interrogée par des opérations d'algèbre relationnelle.

Framework : Un Framework est un kit de composants logiciels structurels, qui a créé les fondations ainsi que les grandes lignes de tout ou d'une partie d'un logiciel (architecture).

Classe : C'est un des concepts de base de la programmation orientée objet. On appelle classe un ensemble d'objets partageant certaines propriétés (les méthodes et les attributs).

Persistiance de données : Au sens général, il s'agit simplement du terme utilisé pour décrire le fait de stocker des données d'une application de manière... persistante ! Autrement dit, de les sauvegarder afin qu'elles ne disparaissent pas lorsque le programme se termine.

Interface de programmation applicative : est un ensemble normalisé de classes, de méthodes ou de fonctions qui sert de façade par laquelle un logiciel offre des services à d'autres logiciels

ANNEXES

```
29         <groupId>org.springframework.boot</groupId>
30         <artifactId>spring-boot-starter-data-jpa</artifactId>
31     </dependency>
32     <dependency>
33         <groupId>org.springframework.boot</groupId>
34         <artifactId>spring-boot-starter-security</artifactId>
35     </dependency>
36     <dependency>
37         <groupId>org.springframework.boot</groupId>
38         <artifactId>spring-boot-starter-web</artifactId>
39     </dependency>
40
41     <dependency>
42         <groupId>org.springframework.boot</groupId>
43         <artifactId>spring-boot-devtools</artifactId>
44         <scope>runtime</scope>
45     </dependency>
46     <dependency>
47         <groupId>org.postgresql</groupId>
48         <artifactId>postgresql</artifactId>
49         <scope>runtime</scope>
50     </dependency>
51     <dependency>
52         <groupId>org.projectlombok</groupId>
53         <artifactId>lombok</artifactId>
54         <optional>true</optional>
55     </dependency>
56     <dependency>
57         <groupId>org.springframework.boot</groupId>
58         <artifactId>spring-boot-starter-test</artifactId>
59         <scope>test</scope>
60     </dependency>
61     <!-- https://mvnrepository.com/artifact/io.jsonwebtoken/jjwt -->
62     <dependency>
63         <groupId>io.jsonwebtoken</groupId>
64         <artifactId>jjwt</artifactId>
```

Les Dépendance maven chargé par spring boot

```
6 import javax.persistence.Entity;
7 import javax.persistence.GeneratedValue;
8 import javax.persistence.Id;
9 import javax.persistence.OneToOne;
10 import javax.persistence.Temporal;
11 import javax.persistence.TemporalType;
12
13 import lombok.Data;
14 import lombok.NoArgsConstructor;
15
16 @Entity
17 @Data
18 @NoArgsConstructor
19 public class Tache implements Serializable {
20
21     private static final long serialVersionUID = 1L;
22
23     @Id @GeneratedValue
24     private Long id;
25     private String nomProjet;
26     private String nomTache;
27     private String description;
28     @Temporal(TemporalType.DATE)
29     private Date dateDebut;
30     @Temporal(TemporalType.DATE)
31     private Date dateEcheance;
32     private String etatTache;
33     @OneToOne
34     private Utilisateur utilisateur;
35     private Date dateAffectation;
36     private boolean archive;
```

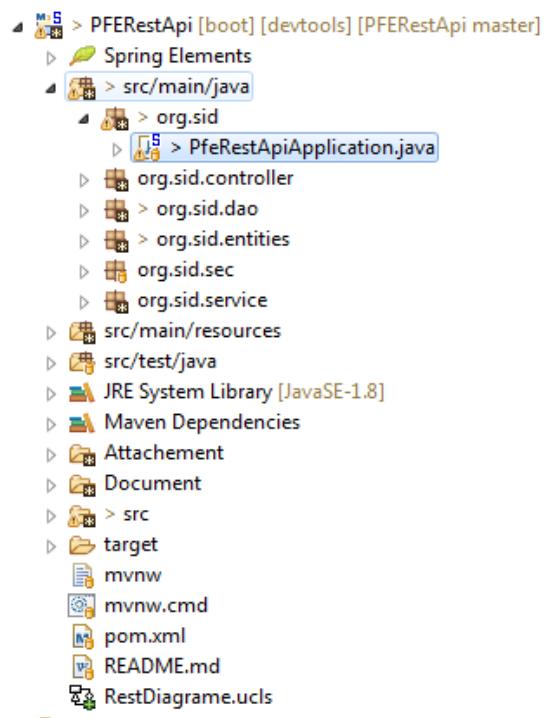
La classe d'entités Tache

```

1 package org.sid.dao;
2
3 import java.util.List;
4
5 public interface EvenementRepository extends JpaRepository<Evenement, Long> {
6
7     @Query("SELECT e FROM Evenement e Where id=:id")
8     Evenement findEventById(@Param("id") Long id);
9
10    @Query("SELECT COUNT(e) FROM Evenement e WHERE e.archive=false ")
11    int totalEvent();
12
13    @Query("SELECT COUNT(e) FROM Evenement e WHERE e.archive=true ")
14    int totalEventArchived();
15
16    @Query("SELECT COUNT(e) FROM Evenement e WHERE e.utilisateur.id=:id")
17    int findTotalByUtilisateurId(@Param("id") Long id);
18
19    List<Evenement> findByUtilisateurIdAndArchiveFalse(Long id);
20
21    List<Evenement> findByArchiveTrue();
22
23    List<Evenement> findByArchiveFalse();
24
25 }

```

L'interface EvenementRepository qui hérite du JpaRepository



Structure de projet spring boot

```

1 package org.sid.service;
2
3④ import java.util.List;⑤
4
5 public interface AttachementService {
6     void store(MultipartFile file);
7
8     Resource loadFile(String filename);
9
10    Attachement saveAttachement(Attachement attachement);
11
12    List<Attachement> findAttachementByTacheId(Tache tache);
13
14    List<Attachement> findAllAttachement();
15
16    void deleteAttachementById(Long id);
17
18 }
19
20
21
22
23
24

```

L'interface Attachement du package service

```

48
49④     }
50④     @Override
51④     public Attachement saveAttachement(Attachement attachement) {
52
53         return attachementRepository.saveAndFlush(attachement);
54     }
55④     @Override
56④     public List<Attachement> findAttachementByTacheId(Tache tache) {
57
58         return attachementRepository.findByTacheId(tache.getId());
59     }
60④     @Override
61④     public List<Attachement> findAllAttachement() {
62
63         return attachementRepository.findAll();
64     }
65④     @Override
66④     public void deleteAttachementById(Long id) {
67         attachementRepository.deleteById(id);
68     }

```

L'implémentation de l'interface attachement

```

1 package org.sid.sec;
2
3④ import org.springframework.beans.factory.annotation.Autowired;[]
4
5 @EnableWebSecurity
6 public class SecurityConfig extends WebSecurityConfigurerAdapter {
7
8     @Autowired
9     private UserDetailsService userDetailsService;
10
11     @Bean
12     public BCryptPasswordEncoder bCryptPasswordEncoder() {
13         return new BCryptPasswordEncoder();
14     }
15
16
17     @Override
18     protected void configure(AuthenticationManagerBuilder auth) throws Exception {
19         auth.userDetailsService(userDetailsService).passwordEncoder(bCryptPasswordEncoder());
20     }
21
22
23     @Override
24     protected void configure(HttpSecurity http) throws Exception {
25         http.csrf().disable();
26
27         http.sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS);
28         http.authorizeRequests().antMatchers("/login/**", "/api/register/**").permitAll();
29
30         http.authorizeRequests().anyRequest().authenticated();
31         http.addFilter(new JWTAuthenticationFilter(authenticationManager()));
32         http.addFilterBefore(new JWTAuthorizationFilter(), UsernamePasswordAuthenticationFilter.class);
33     }
34 }
35
36
37
38
39
40
41
42
43
44
45
46

```

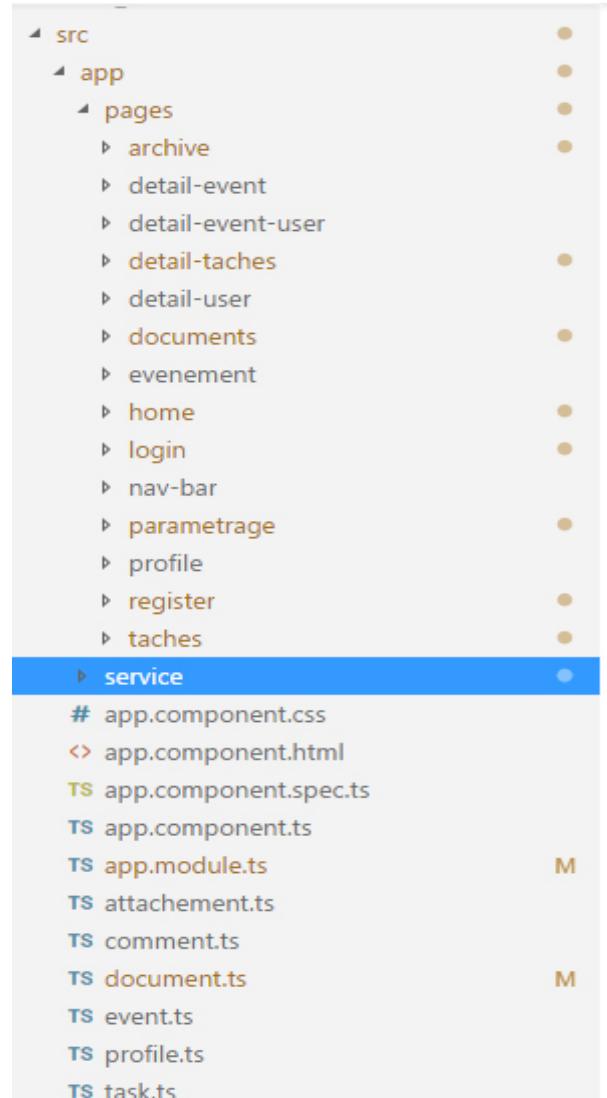
Code source de la classe SecurityConfig de spring Security et l'implémentation de JWT

```

59 @NgModule({
60   declarations: [
61     AppComponent,
62     LoginComponent,
63     HomeComponent,
64     RegisterComponent,
65     NavBarComponent,
66     ParametrageComponent,
67     TachesComponent,
68     ProfileComponent,
69     EvenementComponent,
70     DocumentsComponent,
71     DetailTachesComponent,
72     DetailEventComponent,
73     DetailUserComponent,
74     ParametrageManagerFilter,
75     ParametrageUserFilter,
76     DocumentPriveFilter,
77     TacheFilter,
78     EventFilter,
79     DocumentFilter,
80     ArchiveComponent,
81     DetailEventUserComponent
82   ],
83   imports: [
84     BrowserModule,
85     FormsModule,
86     HttpModule,
87     HttpClientModule,
88     NgxPaginationModule,
89     ChartsModule,
90     RouterModule.forRoot(appRoutes)
91   ],
92   providers: [ArchiveService, UserServiceService, EventServiceService, AuthenticationServiceService, TaskServiceService
93     DataChatServiceService, FileUploadServiceService, AttachementServiceService , ProfileServiceService, AuthGuardService
94     ]
95 }

```

Les services et les composants du module Angular



Structure de projet angular

```
TS user.ts  ×
1  export class User {
2    id: number;
3    nom: string;
4    prenom: string;
5    username: string;
6    password: string;
7    email: string;
8  }
9
```

La classe d'entité user en typeScript

```

1 import { Injectable } from '@angular/core';
2 import { HttpClient, HttpHeaders } from '@angular/common/http';
3 import { JwtHelper } from 'angular2-jwt';
4 import { Router } from '@angular/router';
5 import { User } from '../user';
6 @Injectable()
7 export class AuthenticationService {
8
9     private host = 'http://localhost:8080';
10    private URL = 'http://localhost:8080/api';
11    private jwtToken = null;
12    public username;
13    private roles: Array<any> ;
14    public loggedInUser: User;
15    public isAuthenticated = false ;
16    private subject ;
17    public logedUser;
18    i: number;
19
20    constructor(private http: HttpClient, private router: Router) { }
21
22    login(user) {
23        return this.http.post(this.host + '/login' , user, { observe: 'response'});
24    }
25
26    saveToken(jwt: string) {
27        this.jwtToken = jwt;
28        localStorage.setItem('token', jwt);
29        const jwtHelper = new JwtHelper();
30        this.roles = jwtHelper.decodeToken(this.jwtToken).roles;
31        this.username = jwtHelper.decodeToken(this.jwtToken).sub;
32        console.log(this.subject);
33    }
34    loadToken() {
35        this.jwtToken = localStorage.getItem('token');
36        this.isAuthenticated = true;
37    }
38 }

```

Code source du service d'authentification en typeScript

```

        </div>
    </div>
    <!-- liste de utilisateur --&gt;
&lt;div class="tab-pane fade" id="nav-profile" role="tabpanel" aria-labelledby="nav-profile-tab" *ngIf="auth.isAuthenticated"&gt;
    &lt;div class="table-responsive"&gt;
        &lt;fieldset&gt;
            &lt;legend&gt;Recherche Manager&lt;/legend&gt;
        &lt;div class="form-group"&gt;
            &lt;input [(ngModel)]="motCle" type="text" placeholder=" Par Nom de Manager" class="form-control"&gt;
        &lt;/div&gt;
        &lt;/fieldset&gt;
        &lt;table class="table table-sm"&gt;
            &lt;tr&gt;
                &lt;th&gt;#&lt;/th&gt;
                &lt;th&gt;Nom Prenom&lt;/th&gt;
                &lt;th&gt;Username &lt;/th&gt;
                &lt;th&gt;action&lt;/th&gt;
            &lt;/tr&gt;
            &lt;tr *ngFor="let manager of managers | parametrageManagerFilter:motCle ; let i = index"&gt;
                &lt;td&gt;#{{i+1}}&lt;/td&gt;
                &lt;td&gt;{{manager.nom}} {{manager.prenom }}&lt;/td&gt;
                &lt;td&gt;{{ manager.username }}&lt;/td&gt;

                &lt;td&gt;
                    &lt;button type="button" class="btn btn-sm btn-default" (click)="detailUser(manager)"&gt;&lt;i class="fa fa-eye"&gt;&lt;/i&gt;&lt;/button&gt;
                &lt;/td&gt;
            &lt;/tr&gt;
        &lt;/table&gt;
    &lt;/div&gt;
</pre>

```

Code source d'une page en HTML et Angular5 d'affichage et recherche dans la listes des managers

Conception et réalisation d'une application inter service multiplateforme modulaire

Mondher GUEMRI

الخلاصة: يتمثل العمل المقدم في مشروع ختم الدروس في تصميم وتطوير برنامج للتحكم في الإدارة المشتركة للخدمات داخل المؤسسات مهما كان حجمها او عدد الموظفين وقد قمنا فيها بالتركيز على عملية تنظيم المهام المختلفة داخل المؤسسات ورقمنة المعاملات الورقية.
ويعتمد هذا البرنامج على تكنولوجيا JEE وتقنية WEB SERVICE

Résumé : Le travail réalisé dans ce projet de fin d'études consiste à concevoir et implémenter une application de gestion des services au sein des Institutions, indépendamment de leur taille ou de leur nombre d'employés. Nous nous sommes concentrés sur le processus d'organisation des différentes tâches et la numérisation des transactions sur papier.
L'application développée est fondée sur la technologie JEE et le WEB SERVICE.

Abstract: The work achieved in this graduating project, consists in designing and implement a service management application across enterprises, regardless of size or number of employees.
We focused on the process of organizing different tasks and Paper dematerialization.
The application developed is based on JEE technology and WEB SERVICE.

المفاتيح : سبرينغ فريموورك / انغيلار / واب سرفيس / الإداره المشتركة للخدمات

Mots clés: Framework spring/ Angular/service web/ gestion inter service

Key-words: spring Framework / Angular / web service / inter-service management