



DÉDICACES

*À mon père, **NIEPA Kéi Didier**,
pour son courage, sa sagesse et ses conseils constants, qui m'ont toujours guidé avec rigueur
et bienveillance sur le chemin de la persévérance et de la réussite.*

*À ma mère, **KOUADIO Akissi Chantal, Épouse NIEPA**,
pour son amour inconditionnel, son soutien moral et ses prières silencieuses, qui ont été pour
moi une source inestimable de force et d'inspiration.*

Cette œuvre est aussi la vôtre. Merci pour tout.



REMERCIEMENTS

Je tiens à exprimer ma profonde gratitude à toutes les personnes qui ont, de près ou de loin, contribué au bon déroulement de mon stage de fin de cycle et à la réussite de ce projet :

- à Mme **Elise YRA OUATTARA**, Chef de cabinet du ministère de la Justice et des Droits de l'Homme, pour sa confiance et son implication dans le bon déroulement de mon stage;
- à M. **SEOULOU Auger**, Coordonnateur des projets digitaux au Cabinet du Garde des Sceaux, pour son soutien au quotidien, sa patience et sa bienveillance ;
- à M. **SONAN Abou Landry**, Chef de projet et maître de stage, pour son encadrement technique rigoureux, ses conseils judicieux et sa disponibilité ;
- à Dr. **OUATTARA Adama**, Directeur de l'École Supérieure d'Industrie (ESI), pour son engagement constant en faveur d'une formation de qualité ;
- à M. **KOSSONOU Yao Raky Alvarez**, mon encadreur pédagogique, pour ses retours avisés, ses orientations pertinentes et son accompagnement tout au long du projet ;
- à M. **KONÉ Siriky**, Directeur des études, pour son suivi académique rigoureux et son implication dans la réussite des étudiants ;
- à mes **parents et proches**, pour leurs encouragements constants, leur présence discrète mais essentielle et leur soutien moral tout au long de ce parcours ;
- à **tout le corps administratif et professoral de l'INP-HB**, principalement ceux de l'École Supérieure d'Industrie ;
- au **cabinet du ministère de la Justice et des Droits de l'Homme**, pour m'avoir accueilli durant mon stage de fin de cycle et offert un cadre d'apprentissage stimulant ;
- à **l'ensemble du personnel du cabinet**, pour leur accueil chaleureux, leur disponibilité et l'esprit d'équipe qui a grandement facilité mon intégration ;
- à la **promotion TS STIC 2022–2025**, pour les moments de collaboration et d'entraide.

Ce stage m'a permis de consolider les compétences acquises durant ma formation, de les confronter aux réalités professionnelles et de mieux appréhender les exigences du monde institutionnel. Il constitue l'aboutissement de mon cycle de formation de Technicien Supérieur pour la période 2022–2025.



AVANT-PROPOS

L'Institut National Polytechnique Félix Houphouët-Boigny (INP-HB) de Yamoussoukro a été créé en septembre 1996 par le décret N°96-678, résultant de la fusion de quatre (04) grandes écoles : l'École Nationale Supérieure d'Agronomie (ENSA), l'École Nationale Supérieure des Travaux Publics (ENSTP), l'Institut Agricole de Bouaké (IAB), et l'Institut National Supérieur de l'Enseignement Technique (INSET). Cette fusion visait à revitaliser ces institutions en vue de former une jeunesse capable de relever les défis du monde professionnel et de faire de Yamoussoukro une technopole. Aujourd'hui, l'INPHB regroupe neuf (11) écoles :

- École Supérieure d'Industrie (ESI) ;
- École Supérieure d'Agronomie (ESA) ;
- École Supérieure de Commerce et d'Administration des Entreprises (ESCAE) ;
- École Supérieure des Travaux Publics (ESTP) ;
- École Supérieure des Mines et de Géologie (ESMG) ;
- École de Formation Spécialisée et de Perfectionnement des Cadres (EFSPC) ;
- Classes Préparatoires aux Grandes Écoles (CPGE) ;
- École Doctorale Polytechnique de Sciences Appliquées et Procédés de Transformation (EDPSAPT) ;
- École Supérieure de Chimie Pétrole et de l'Énergie (ESCPE) ;
- École Doctorale Polytechnique en Sciences et Technologies de l'Ingénieur (EDPSTI) ;
- École Supérieure de l'Aéronautique et du Spatial (ESAS).

Dans le cadre de notre cursus de Technicien Supérieur en Informatique à l'École Supérieure d'Industrie (ESI), établissement chargé de la formation de cadres techniques et d'ingénieurs dans les domaines clés de l'industrie, des stages professionnels ont été prévus en fin de cycle. Ces stages ont visé à permettre aux étudiants de mettre en œuvre les connaissances théoriques qu'ils ont acquises durant leur formation et de les adapter aux exigences du monde professionnel. C'est dans cette dynamique que nous avons été accueillis par le cabinet du ministère de la Justice et des Droits de l'Homme, pour une période allant du 6 mars au 6 juin 2025, afin de réaliser le projet exposé dans ce mémoire.



SOMMAIRE

DÉDICACES.....	I
REMERCIEMENTS.....	II
AVANT-PROPOS.....	III
SOMMAIRE.....	IV
LISTE DES SIGLES ET ABRÉVIATIONS.....	V
LISTE DES TABLEAUX.....	VII
LISTE DES FIGURES.....	VIII
RÉSUMÉ.....	IX
ABSTRACT.....	X
INTRODUCTION.....	1
PARTIE I : CADRE ET CONTEXTE DU PROJET.....	2
CHAPITRE I : PRÉSENTATION DE LA STRUCTURE D'ACCUEIL.....	3
I. PRÉSENTATION DE LA STRUCTURE D'ACCUEIL.....	3
II. DOMAINE D'ACTIVITÉS.....	3
III. ORGANIGRAMME HIÉRARCHIQUE.....	5
CHAPITRE II : DESCRIPTION DU PROJET.....	6
I. CONTEXTE DU PROJET.....	6
II. OBJECTIFS DU PROJET.....	7
III. CAHIER DES CHARGES.....	7
IV. ÉTUDE DE L'EXISTANT.....	9
V. MÉTHODE DE GESTION DE PROJET.....	10
VI. PLANIFICATION DU TRAVAIL.....	13
PARTIE II : ÉTUDE CONCEPTUELLE.....	15
CHAPITRE III : APPROCHE MÉTHODOLOGIQUE.....	16
I. INTRODUCTION A LA MÉTHODE D'ANALYSE ET DE CONCEPTION.....	16
II. CHOIX DE LA MÉTHODE D'ANALYSE.....	19
CHAPITRE IV : CRÉATION DE L'ARCHITECTURE DU SYSTÈME.....	20
I. IDENTIFICATION DES ACTEURS ET DES CAS D'UTILISATION.....	20
II. DIAGRAMMES DE SÉQUENCE.....	26
III. DIAGRAMME DE CLASSES.....	31
PARTIE III : MISE EN ŒUVRE.....	32
CHAPITRE V : ÉTUDE TECHNIQUE.....	33
I. COMPARAISON ET CHOIX D'OUTILS.....	33
II. ARCHITECTURE LOGICIELLE.....	41
CHAPITRES VI : REALISATION.....	42
CONCLUSION.....	48
RÉFÉRENCES BIBLIOGRAPHIQUES.....	XIII
RÉFÉRENCES WEBGRAPHIQUES.....	XIV
TABLE DES MATIÈRES.....	XV



LISTE DES SIGLES ET ABRÉVIATIONS

A

API : Application Programming Interface

B

BaaS : Backend as a Service

C

CRUD : Create, Read, Update, Delete

D

DBMS : DataBase Management System

G

GED : Gestion Electronique de Documents

H

HTTP : HyperText Transfer Protocol

HTTPS : HyperText Transfer Protocol Secure

I

IDE : Integrated Development Environment

J

JSON : JavaScript Object Notation

JWT : JSON Web Token

N

Nginx : Engine X (serveur web et reverse proxy, le nom est jeu de mots sur « Engine X »)



P

PGAdmin4 : PostgreSQL Graphical Administration Tool

R

REST : Representational State Transfer

S

SQL : Structured Query Language

SSL : Secure Sockets Layers

T

TLS : Transport Layer Security

U

UI : User Interface

UX : User Experience

V

VS Code : Visual Studio Code



LISTE DES TABLEAUX

Tableau 1: Gestion des courriers pour un agent	8
Tableau 2: Gestion des courriers pour un destinataire	8
Tableau 3: Problèmes du système actuel.....	10
Tableau 4: Liste des tâches	13
Tableau 5: Comparaison Merise-UML	18
Tableau 6: Identification des acteurs	20
Tableau 7: Description textuelle de "Créer un courrier"	22
Tableau 8: Description textuelle de "Envoyer un courrier"	22
Tableau 9: Description textuelle de "Imputer un courrier"	23
Tableau 10: Description textuelle de "Rechercher un courrier"	23
Tableau 11: Description textuelle de "Consulter l'historique"	23
Tableau 12: Description textuelle de "Lire un courrier"	24
Tableau 13: Description textuelle de "Remplir le bordereau"	24
Tableau 14: Description textuelle de "Renvoyer le courrier"	25
Tableau 15: Description textuelle de "Gérer les utilisateurs"	25
Tableau 16: Comparaison de quelques environnements de développement	33
Tableau 17: Comparaison de quelques serveurs web et solutions d'hébergement.....	35
Tableau 18: Comparatif des technologies frontend/ backend	37
Tableau 19: Comparaison des SGBD courants	38
Tableau 20: Quelques outils de test d'API	39



LISTE DES FIGURES

Figure 1: Organigramme du cabinet du ministère de la Justice et des Droits de l'Homme.....	5
Figure 2: Diagramme de planification des tâches	14
Figure 3: Diagramme des cas d'utilisation.....	21
Figure 4: Diagramme des cas d'utilisation.....	21
Figure 5: Diagramme de séquence "Créer un courrier"	26
Figure 6: Diagramme de séquence "Envoyer un courrier"	26
Figure 7: Diagramme de séquence "Imputer un courrier"	27
Figure 8: Diagramme de séquence "Rechercher un courrier"	27
Figure 9: Diagramme de séquence "Consulter l'historique"	28
Figure 10: Diagramme de séquence "Lire un courrier"	28
Figure 11: Diagramme de séquence "Remplir le bordereau"	29
Figure 12: Diagramme de séquence "Renvoyer le courrier"	29
Figure 13: Diagramme de séquence "Gérer les utilisateurs et les rôles"	30
Figure 14: Diagramme des classes	31
Figure 15: Logo de Visual Studio Code.....	34
Figure 17: Logo de Render.....	36
Figure 16: Logo de BACKBLAZE	36
Figure 18: Logo de PgAdmin	39
Figure 19: Logo de Postman	40
Figure 20: Architecture logicielle	41
Figure 21: Page de connexion	42
Figure 22: Page d'accueil d'un agent.....	43
Figure 23: Formulaire de création d'un courrier	43
Figure 24: Formulaire de création d'un courrier	44
Figure 25: Formulaire d'imputation d'un courrier.....	44
Figure 26: Formulaire d'envoi d'un courrier	45
Figure 27: Historique du courrier	45
Figure 28: Page des courriers reçus	46
Figure 29: Page d'accueil d'un destinataire (directeur de cabinet)	46
Figure 30: Boîte de réception du destinataire le directeur de cabinet	47
Figure 31: Page du renvoi de courriers	47



RÉSUMÉ

Ce mémoire de fin d'études s'est inscrit dans la stratégie de transformation numérique du cabinet du ministère de la Justice et des Droits de l'Homme, visant à optimiser la gestion administrative à travers la numérisation des courriers et des bordereaux d'envoi. L'objectif principal a été de faciliter l'imputation, c'est-à-dire l'orientation rapide et structurée des courriers vers les services ou responsables concernés, tout en assurant une meilleure traçabilité.

Notre mission a donc consisté à concevoir et déployer une solution GED sécurisée, capable de répondre aux impératifs de performance, de confidentialité et d'accessibilité à distance. Pour ce faire, nous avons opté pour une architecture basée sur les protocoles sécurisés HTTPS, afin de garantir la confidentialité des échanges et la protection des données sensibles.

La mise en œuvre de cette solution a nécessité une analyse précise des besoins des utilisateurs (agents du cabinet, directeur de cabinet etc.), ainsi qu'une modélisation rigoureuse de l'architecture du système des entrées et imputations du courrier au service courrier. Des tests fonctionnels ont été réalisés pour assurer la fiabilité du système et son adéquation aux pratiques administratives en vigueur.

À l'issue du projet, nous avons pu déployer une plateforme opérationnelle permettant la gestion centralisée, l'archivage électronique, et la circulation sécurisée des courriers au sein du cabinet ministériel. Cette solution a contribué à la modernisation du traitement administratif et a renforcé la réactivité des services dans le suivi et le traitement des documents.

Néanmoins, le projet n'a pas été exempt de défis, notamment en ce qui concerne la complexité technique liée à l'intégration dans l'infrastructure existante, ainsi que la complexité des techniques utilisées. Malgré cela, notre approche structurée, notre persévérance et notre capacité à proposer des solutions adaptées nous ont permis de livrer une solution conforme aux attentes dans les délais impartis.

Mots-clés : Transformation numérique, gestion administrative, numérisation des courriers, bordereaux d'envoi, imputation, traçabilité, GED (Gestion Électronique de Documents), modélisation du système documentaire, archivage électronique, circulation sécurisée des courriers, modernisation du traitement administratif.



ABSTRACT

This final year dissertation is part of the digital transformation strategy of the Ministry of Justice and Human Rights' cabinet, aiming to optimize administrative management through the digitization of incoming mail and dispatch slips. The main objective was to facilitate imputation, that is, the rapid and structured routing of correspondence to the relevant departments or officials, while ensuring better traceability.

Our mission was therefore to design and deploy a secure Electronic Document Management (EDM) solution capable of meeting the requirements of performance, confidentiality, and remote accessibility. To achieve this, we opted for an architecture based on secure HTTPS **protocols**, in order to ensure the confidentiality of communications and the protection of sensitive data.

Implementing this solution required a precise analysis of user needs (cabinet staff, chief of staff, etc.), as well as a thorough modeling of the system architecture for mail entry and imputation within the mail service. Functional tests were carried out to ensure the system's reliability and compliance with current administrative practices.

At the end of the project, we were able to deploy an operational platform allowing for centralized management, electronic archiving, and secure circulation of mail within the ministry's cabinet. This solution contributed to the modernization of administrative processing and strengthened the responsiveness of services in the monitoring and handling of documents.

Nevertheless, the project was not without challenges, particularly regarding the technical complexity of integration into the existing infrastructure and the sophistication of the technologies used. Despite this, our structured approach, perseverance, and ability to propose appropriate solutions enabled us to deliver a system that met expectations within the given timeframe.

Keywords

Digital transformation, administrative management, digitization of mail, dispatch slips, imputation, traceability, EDMS (Electronic Document Management System), document system modeling, electronic archiving, secure mail circulation, modernization of administrative processing.



INTRODUCTION

Au sein du Cabinet du ministère de la Justice et des Droits de l'Homme, la gestion des courriers administratifs a encore largement reposé sur des procédures manuelles. Lorsqu'un courrier est reçu, il est d'abord enregistré par le service courrier dans un registre papier appelé registre de transcription, puis est accompagné d'un bordereau dont l'en-tête est rempli avant sa transmission. L'ensemble est ensuite acheminé vers le Directeur de cabinet ou certains responsables hiérarchiques habilités à effectuer l'imputation, c'est-à-dire l'orientation du courrier vers les services ou agents concernés.

Une fois les imputations réalisées, le courrier et son bordereau sont renvoyés au service courrier, qui les a enregistrés dans un second registre nommé registre de transmission. Le courrier est ensuite physiquement transmis aux destinataires, conformément aux instructions du Directeur de cabinet. Bien que structuré, ce processus a présenté de nombreuses limites : accumulation des courriers en attente d'orientation, risques de perte ou de détérioration des documents, lenteur dans la circulation, difficulté à assurer un suivi en temps réel, et absence d'un système d'archivage numérique.

Le projet présenté dans ce mémoire s'est inscrit dans une démarche de modernisation de cette chaîne de traitement. Il a visé à mettre en place une plateforme de Gestion Électronique de Documents (GED) destinée à l'archivage numérique et à la simplification du traitement des courriers, depuis leur réception au service courrier jusqu'à leur imputation. L'objectif principal a été de numériser les courriers entrants et sortants ainsi que leurs bordereaux, afin de faciliter leur traçabilité, d'optimiser l'imputation, et de permettre une recherche rapide et centralisée à travers les différents registres.

Ce mémoire retrace l'ensemble des étapes ayant conduit à la réalisation de cette solution. Il débute par une analyse des besoins et une étude de l'existant, avant de définir les exigences fonctionnelles et techniques. La phase de conception s'appuie sur la modélisation des processus à l'aide de la méthode UML, suivie par la définition de l'architecture technique du système. Enfin, on détaille le développement de la plateforme, les résultats obtenus, les perspectives d'évolution, ainsi que les recommandations en vue d'une mise en production efficace et pérenne.



PARTIE I : CADRE ET CONTEXTE DU PROJET

Cette section est dédiée à
la présentation de l'entreprise d'accueil et la description détaillée du thème.



CHAPITRE I : PRÉSENTATION DE LA STRUCTURE D'ACCUEIL

Ce chapitre a pour objectif de présenter le contexte dans lequel notre projet se déroule en mettant en évidence le rôle et les activités de ce cadre.

I. PRÉSENTATION DE LA STRUCTURE D'ACCUEIL

Le ministère de la Justice et des Droits de l'Homme, dirigé par le Garde des Sceaux, est une institution clé du système judiciaire ivoirien. Avec à sa tête le ministre Monsieur Jean Sansan KAMBILE. Il œuvre pour l'application de la justice, la promotion des droits humains et la bonne gouvernance. Il est structuré autour de plusieurs entités.

Le Cabinet Ministériel est l'organe stratégique du Ministère. Il assiste le Ministre dans l'élaboration des politiques. Il est composé notamment d'un Directeur de Cabinet, d'un Chef de Cabinet, de Conseillers Techniques et de Chargés d'Études.

L'Administration Centrale comprend plusieurs directions spécialisées :

- la DELD s'occupe de la rédaction des textes de loi ;
- la DACP veille à l'application des lois civiles et pénales ;
- la DAP gère les établissements pénitentiaires ;
- la DPJEJ est chargée de la justice des mineurs ;
- la DCECA supervise l'état civil et les archives judiciaires ;
- les Directions de la Promotion et de la Protection des Droits de l'Homme œuvrent pour le respect et la diffusion des droits fondamentaux.

Les Directions, Services Extérieurs et Services Rattachés au cabinet viennent en appui, avec des structures telles que l'Inspection Générale, la Direction des Ressources Humaines, la Direction des Finances, la Planification, les Infrastructures, la Communication et le Service Informatique.

II. DOMAINE D'ACTIVITÉS

Le ministère de la Justice et des Droits de l'Homme intervient dans plusieurs domaines clés :

- élaboration et application des textes législatifs : par le biais de la DELD, le Ministère rédige et met en œuvre des lois et règlements adaptés aux besoins de la société ;
- gestion des affaires civiles et pénales : la DACP veille à l'application des lois en matière civile, pénale et commerciale, et facilite l'accès à la justice pour tous ;
- administration pénitentiaire : la DAP gère les établissements pénitentiaires, assurant la sécurité, la réhabilitation et la réinsertion des détenus ;
- protection des mineurs : la DPJEJ s'occupe des mineurs en conflit avec la loi, en danger ou vulnérables, en leur offrant une protection judiciaire adaptée ;



- gestion de l'état civil et des archives : la DCECA centralise et conserve les archives judiciaires, et supervise l'application des lois relatives à l'état civil ;
- promotion et protection des droits de l'homme : les directions dédiées promeuvent les droits humains, luttent contre l'impunité et veillent au respect des droits des personnes vulnérables ;
- gestion des ressources humaines et financières : les services rattachés assurent la gestion efficace du personnel, des finances, des infrastructures et de la communication du Ministère.

Cette organisation permet au Ministère de remplir efficacement ses missions, en assurant une justice équitable, la protection des droits de l'homme et le renforcement de la bonne gouvernance en Côte d'Ivoire.



III. ORGANIGRAMME HIÉRARCHIQUE

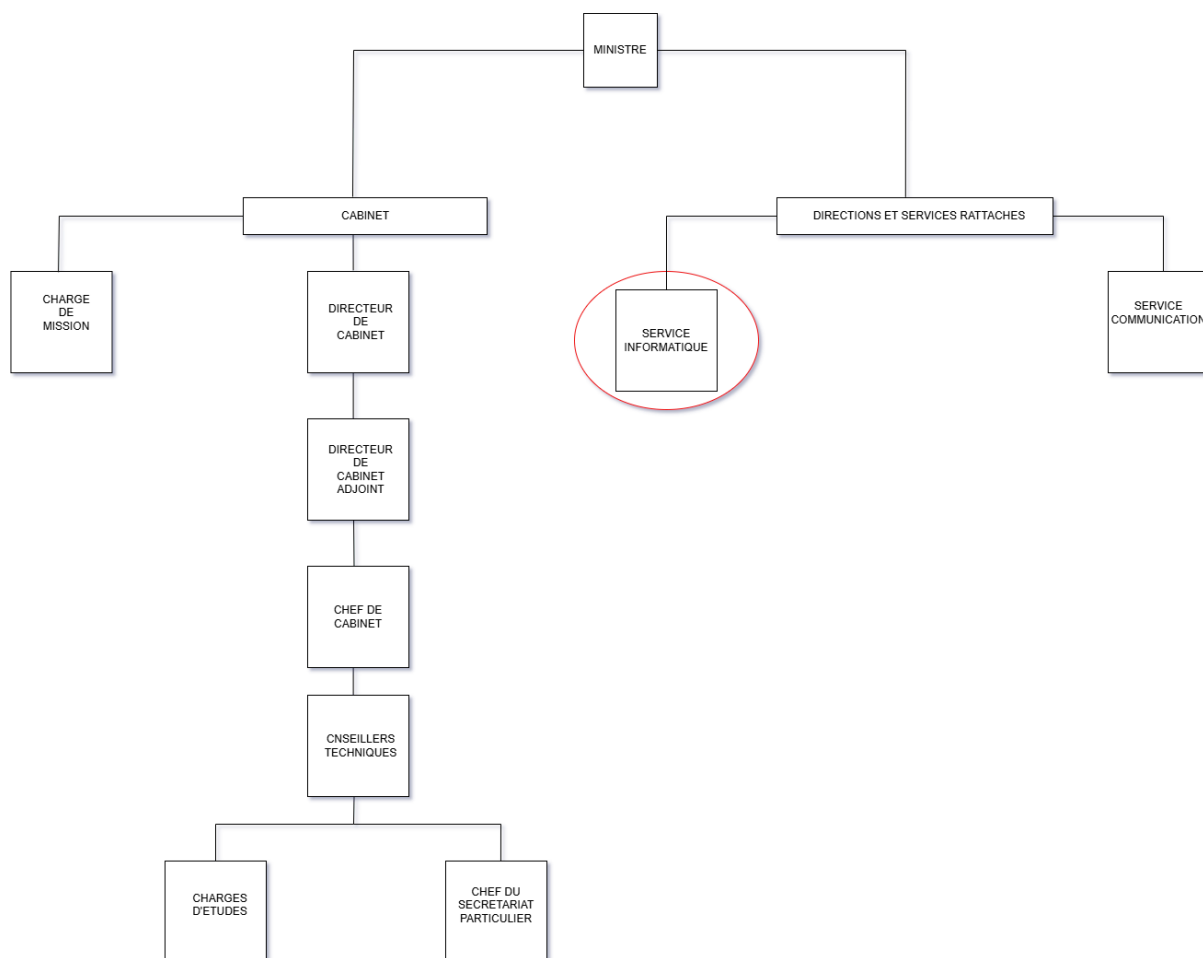


Figure 1: Organigramme du cabinet du ministère de la Justice et des Droits de l'Homme



CHAPITRE II : DESCRIPTION DU PROJET

Ce chapitre vise à présenter en détail le projet en question, en définissant clairement ses objectifs ainsi que les spécifications détaillées énoncées dans le cahier des charges.

I. CONTEXTE DU PROJET

Dans un monde en pleine transformation numérique, les administrations publiques sont appelées à moderniser leurs modes de fonctionnement afin de gagner en efficacité, en transparence et en traçabilité. Au sein du cabinet du Garde des Sceaux, ministère de la Justice et des Droits de l'Homme de Côte d'Ivoire, la gestion des courriers administratifs a encore essentiellement reposé sur des méthodes traditionnelles, principalement manuelles. Cette organisation, bien qu'établie, a présentée des limites importantes : lenteur dans l'imputation des courriers, absence de traçabilité numérique, difficulté d'accès rapide aux documents, et risque de perte ou de détérioration physique.

Actuellement, lorsqu'un courrier est reçu, il est enregistré dans un registre papier appelé registre de transcription par le service courrier, puis est accompagné d'un bordereau transmis au directeur de Cabinet pour imputation. Ce dernier a ensuite orienté le courrier vers les services concernés. Une fois l'imputation faite, le bordereau est retourné au service courrier, qui a procédé à un second enregistrement dans un autre registre appelé registre de transmission avant de distribuer le courrier conformément aux directives reçues.

Face à ce mode de gestion, il est devenu indispensable d'adopter une application de Gestion Électronique de Documents (GED). Cette solution a permis la numérisation des courriers, bordereaux, registres de transcription et de transmission, afin de faciliter le processus d'imputation du courrier, de suivi, tout en garantissant une meilleure traçabilité, accessibilité et conservation des documents.

C'est dans cette optique que le projet de « MISE EN PLACE D'UNE PLATEFORME GED (GESTION ÉLECTRONIQUE DE DOCUMENTS) POUR L'ARCHIVAGE ET LA CIRCULATION DES COURRIERS ADMINISTRATIFS : CAS DU CABINET DU MINISTÈRE DE LA JUSTICE ET DES DROITS DE L'HOMME » a vu le jour. Il vise à accompagner la transition numérique du cabinet du ministère, tout en répondant aux besoins spécifiques des utilisateurs internes, notamment le service courrier, le Directeur de Cabinet et les différents services destinataires des courriers.



II. OBJECTIFS DU PROJET

1. Objectif général

Mettre en place un système informatisé de gestion des courriers permettant aux agents et destinataires du ministère de créer, envoyer, suivre, imputer et archiver des courriers administratifs de manière sécurisée, centralisée et traçable, en respectant les règles de traitement formalisées par les bordereaux.

2. Objectifs spécifiques

Les objectifs spécifiques sont :

- permettre la création contrôlée des courriers en garantissant l'unicité de chaque courrier (vérification du numéro de référence) ;
- assurer une traçabilité complète via des registres de transcription et de transmission automatiques à chaque étape (création, envoi, imputation) ;
- générer automatiquement les bordereaux de transmission, pré-remplis à partir des informations du courrier pour faciliter le traitement par les destinataires ;
- contrôler l'imputation en empêchant qu'un courrier soit imputé plusieurs fois ou sans bordereau complet ;
- restreindre les envois à des destinataires spécifiques selon le rôle de l'agent (ex. : Directeur de Cabinet, IGSJP...) ;
- fournir une interface sécurisée de lecture seule aux destinataires, leur permettant de compléter uniquement les champs qui les concernent ;
- offrir des outils de recherche avancée et d'historique pour suivre les actions effectuées sur chaque courrier ;
- stocker et organiser toutes les données dans une base centralisée PostgreSQL, avec les pièces jointes enregistrées sur le cloud.

III. CAHIER DES CHARGES

Ci-dessous, le cahier des charges détaillant les fonctionnalités de l'application :

1. Utilisateurs concernés

Les utilisateurs concernés sont :

- agents du service courrier (création, envoi, imputation, lecture) ;
- destinataires (lecture, remplissage du bordereau, renvoi) : Directeur de Cabinet, Directeur de Cabinet Adjoint, IGSJP, Chef du Cabinet, Conseillers Techniques, Chargé d'Etudes, Chef du Secrétariat Particulier et Directeurs ;
- (optionnel) Administrateurs (gestion des comptes et sécurité).



2. Fonctionnalités principales

2.1. Gestion des courriers

Pour un agent :

Tableau 1: Gestion des courriers pour un agent

Fonction	Description
Création de courrier	Vérifie l'unicité via numéro de référence.
Envoi de courrier	Bordereau généré automatiquement. Envoi unique.
Imputation	Ne peut être faite qu'une seule fois. Nécessite un bordereau rempli.
Historique et Recherche	Recherche multicritère + journalisation des actions.
Envoi Ciblé	L'agent ne peut envoyer qu'à certains profils prédéfinis.

Pour un destinataire :

Tableau 2: Gestion des courriers pour un destinataire

Fonction	Description
Remplir le bordereau	Ils complètent uniquement les sections à leur charge.
Lire un courrier	En lecture seule, si le courrier leur est imputé.
Renvoyer à l'agent	Une fois le bordereau complété.

3. Contraintes

Un courrier ne peut être :

- créé qu'une seule fois (clé unique) ;
- envoyé qu'une seule fois ;
- imputé qu'une fois.

L'envoi nécessite un bordereau valide :

- l'imputation nécessite que le bordereau ait été rempli ;
- l'accès au courrier est réservé au destinataire imputé.



IV. ÉTUDE DE L'EXISTANT

Parfait, nous allons passer à l'étude de l'existant, une étape essentielle de ton rapport. Cette partie a servi à :

- analyser comment le système fonctionne actuellement (sans l'application) ;
- identifier les limites et problèmes du fonctionnement manuel ou partiellement informatisé ;
- justifier l'intérêt de développer une application pour améliorer la gestion des courriers.

1. Description du système actuel

Actuellement, la gestion des courriers au sein du ministère repose majoritairement sur un traitement manuel ou semi-informatisé :

- les courriers sont enregistrés manuellement dans des registres physiques (papier) ;
- le suivi se fait via des bordereaux imprimés qui accompagnent les courriers entre les services ;
- les imputations sont indiquées à la main et signées ;
- les dates de réception, d'envoi ou de retour sont inscrites à la main dans des registres ou des tableaux Excel ;
- les recherches sur les courriers passés sont longues, car elles nécessitent de feuilleter les registres. ;
- les copies ou versions numériques des courriers (PDF, scans) sont parfois stockées dans des dossiers partagés mais sans lien direct avec l'historique papier.

2. Acteurs impliqués

Agent du courrier : réceptionne, enregistre et transmet les courriers.

Destinataires (directeurs, etc.) : reçoivent les courriers, remplissent le bordereau, donnent des instructions.

Secrétaires : aident à la préparation et à l'organisation des documents à transmettre.

Archivistes : conservent les bordereaux et les courriers dans les archives physiques.

3. Documents utilisés

Les documents utilisés sont :

- un registre d'arrivée des courriers ;
- un bordereau de transmission (papier) ;
- un registre d'imputation ;
- des dossiers physiques pour chaque courrier.

4. Outils informatiques utilisés

Les outils informatiques utilisés sont :



- des ordinateurs avec Excel / Word ;
- un stockage de fichiers PDF/Word sur des clés USB ou disques partagés.

5. Problèmes rencontrés

Tableau 3: Problèmes du système actuel

Problèmes	Description
Lenteur	Le traitement est lent, surtout pour les courriers urgents.
Risques d'erreurs	Numéros de référence en double, erreurs de saisie.
Difficulté de suivi	Impossible de savoir rapidement où se trouve un courrier à un instant T.
Perte de documents	Certains bordereaux ou courriers peuvent être égarés.
Manque de traçabilité	Pas d'historique clair des actions effectuées.
Pas d'alertes ni de notifications	Aucun moyen de signaler un retard ou une urgence automatiquement.

6. L'intérêt de l'application proposée

La mise en place d'une application numérique permet de :

- automatiser les processus (création, envoi, imputation) ;
- gagner en rapidité et en efficacité dans le traitement des courriers ;
- assurer une traçabilité complète et en temps réel de chaque courrier ;
- renforcer la sécurité par une gestion des droits et rôles utilisateurs ;
- centraliser les données dans une base fiable et consultable ;
- permettre des recherches instantanées par mots-clés ou filtres ;
- éviter les doublons, erreurs d'enregistrement ou oublis ;
- réduire l'utilisation du papier et favoriser une démarche écoresponsable.

V. MÉTHODE DE GESTION DE PROJET

1. La méthode Agile

1.1. Définition

La méthode Agile est une approche de gestion de projet qui met l'accent sur l'incrémentation, l'itération et la collaboration étroite entre les membres de l'équipe et les parties prenantes. Elle vise à fournir des livrables fonctionnels régulièrement et à s'adapter rapidement aux changements de besoins ou de priorités.

1.2. Principes de la méthode Agile

Les principes fondamentaux de la méthode Agile incluent :

- la satisfaction du client par la livraison continue de logiciels de valeur ;
- l'acceptation du changement, même tard dans le développement ;
- la livraison fréquente de logiciels fonctionnels (toutes les deux semaines à deux mois) ;



- la collaboration quotidienne entre les développeurs et les parties prenantes ;
- la construction de projets autour d'individus motivés et la confiance dans leur capacité à accomplir les tâches ;
- la communication en face à face comme la forme de communication la plus efficace ;
- la mesure du progrès principalement par la quantité de logiciels fonctionnels livrés ;
- le développement durable, en maintenant un rythme constant indéfini ;
- l'excellence technique et la bonne conception améliorant l'agilité ;
- la simplicité, ou l'art de maximiser la quantité de travail non fait, est essentielle ;
- les meilleures architectures, exigences et conceptions émergent d'équipes autoorganisées ;
- les ajustements réguliers pour devenir plus efficaces.

1.3. Présentation de Scrum

Scrum est un cadre de travail Agile qui utilise des itérations courtes et fixes, appelées Sprints, généralement de deux à quatre semaines. Il se concentre sur la maximisation de la capacité de l'équipe à livrer rapidement des produits de valeur. Scrum se compose de rôles définis, d'événements, d'artéfacts et de règles, chacun ayant un objectif spécifique pour gérer le développement et la livraison de produits.

2. Structures de Scrum

2.1. Rôle dans Scrum

2.1.1. Scrum Master

Le Scrum Master est responsable de promouvoir et de soutenir Scrum. Il aide l'équipe à comprendre et à adopter les valeurs, principes et pratiques de Scrum, élimine les obstacles et facilite les événements Scrum.

2.1.2. Product Owner

Le Product Owner est responsable de maximiser la valeur du produit résultant du travail de l'équipe de développement. Il gère le Backlog Produit, en s'assurant qu'il est visible, transparent et compréhensible.

2.1.3. Equipe de développement

L'Équipe de Développement est composée de professionnels qui réalisent le travail de livraison d'un Increment potentiellement livrable à la fin de chaque Sprint. Ils sont auto-organisés et multifonctionnels.

2.2. Evènements Scrum

2.2.1. Sprint

Un Sprint est une itération de développement de 1 à 4 semaines. Chaque Sprint commence avec une planification et se termine avec une revue et une rétrospective.

2.2.2. Planification du Sprint

C'est une réunion où l'équipe de Scrum planifie le travail à accomplir pendant le Sprint. Le Product Owner définit les objectifs du Sprint et l'Équipe de Développement sélectionne les éléments du Backlog Produit à compléter.

2.2.3. Réunion quotidienne

Une courte réunion quotidienne (15 minutes maximum) où l'équipe de développement synchronise ses activités et crée un plan pour les prochaines 24 heures.



2.2.4. Revue de Sprint

À la fin de chaque Sprint, l'équipe de Scrum et les parties prenantes se réunissent pour inspecter l'incrément et adapter le Backlog Produit si nécessaire.

2.2.5. Rétrospective de Sprint

Une réunion après la Revue de Sprint où l'équipe de Scrum réfléchit sur le Sprint passé et planifie des améliorations à mettre en œuvre dans le prochain Sprint.

2.3. Artéfacts Scrum

2.3.1. Backlog Produit

Le Backlog Produit est une liste ordonnée de tout ce qui pourrait être nécessaire dans le produit. C'est la source unique de travail pour l'équipe de développement.

2.3.2. Backlog de Sprint

Le Backlog de Sprint est l'ensemble des éléments du Backlog Produit sélectionnés pour le Sprint, plus un plan pour délivrer l'incrément et atteindre l'objectif du Sprint.

2.3.3. Backlog Increment

Un Increment est la somme de tous les éléments du Backlog Produit complétés durant un Sprint et les Sprints précédents. Il doit être utilisable et répondre à la définition de "Terminé" (Definition of Done).

2.4. Mise en œuvre de Scrum

2.4.1. Initiation d'un projet Scrum

Définir les objectifs du projet, identifier les parties prenantes, former l'équipe Scrum et créer le premier Backlog Produit.

2.4.2. Planification et Estimation

Utiliser des techniques de planification Agile, telles que la planification poker, pour estimer la taille des éléments du Backlog Produit et planifier les Sprints.

2.4.3. Exécution et contrôle des Sprints

Suivre l'avancement du Sprint avec des outils comme le tableau Scrum et les graphiques de burndown. Assurer une communication constante et résoudre les obstacles.

2.4.4. Suivi et Adaptation

À la fin de chaque Sprint, utiliser les revues et rétrospectives pour évaluer les résultats et adapter le processus et le Backlog Produit en conséquence.

2.4.5. Avantage de Scrum

L'avantage de SCRUM est :

- la réactivité accrue aux changements de besoins ;
- l'amélioration de la qualité du produit grâce aux itérations fréquentes et aux retours constants ;
- la meilleure collaboration et communication au sein de l'équipe et avec les parties prenantes ;
- la livraison rapide de fonctionnalités de valeur ;
- la transparence accrue sur l'avancement et les obstacles du projet.



Scrum, en tant que cadre Agile, offre une méthode structurée mais flexible pour gérer le développement de produits complexes. En mettant l'accent sur l'adaptation continue, la collaboration étroite et la livraison itérative de valeurs, Scrum aide les équipes à répondre efficacement aux besoins changeants des clients tout en maintenant un haut niveau de qualité.

VI. PLANIFICATION DU TRAVAIL

La planification est une étape très importante dans la réalisation d'un projet car elle aide à identifier toutes les tâches, permettant ainsi de tenir compte de tous les aspects. Beaucoup négligent cette phase car elle est perçue comme une perte de temps. Cette erreur de précipitation explique en grande partie les échecs des projets actuels. La liste des tâches est présentée dans le tableau ci-dessous.

1. Liste des tâches

Tableau 4: Liste des taches

TACHES	DUREE	COMPOSITION
CADRE ET CONTEXTE DU PROJET	2 semaines	<ul style="list-style-type: none">- compréhension du thème ;- élaboration du cahier des charges.
CONNCEPTION	2 semaines	<ul style="list-style-type: none">- choix de la méthode d'analyse ;- modélisation du système ;- définition de l'architecture de l'application.
RÉALISATION	2 semaines	<ul style="list-style-type: none">- définition du design graphique ;- codage des interfaces et fonctionnalités.
RÉDACTION DU MEMOIRE	3 semaines	Structuration et rédaction du contenu
TEST ET VALIDATION	1 semaine	<ul style="list-style-type: none">- exécution de tests ;- correction des erreurs ;- Validation



2. Planification des tâches

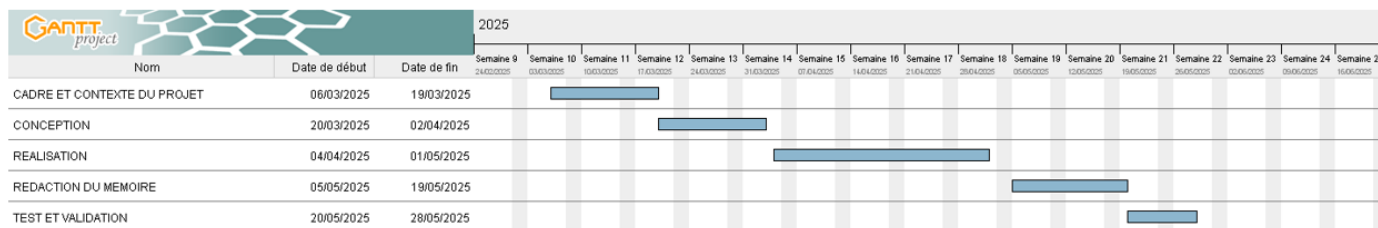


Figure 2: Diagramme de planification des tâches



PARTIE II : ÉTUDE CONCEPTUELLE

Dans cette seconde section, nous avons abordé la présentation des différentes méthodes de modélisation que nous avons étudiées, en avons sélectionné une, puis avons procédé à la modélisation conceptuelle, organisationnelle et physique pour notre projet.



CHAPITRE III : APPROCHE MÉTHODOLOGIQUE

Nous avons commencé par présenter les différentes méthodes d'analyse, puis nous avons défini la méthode d'analyse que nous avons choisi d'utiliser.

I. INTRODUCTION A LA MÉTHODE D'ANALYSE ET DE CONCEPTION

1. Méthode d'analyse

L'objectif de la méthode d'analyse réside dans la formalisation des étapes initiales de la création d'un système, visant ainsi à répondre de manière appropriée aux besoins du client. Cette démarche implique de partir d'une description informelle des besoins exprimés par le client, complétée par des recherches dans le domaine fonctionnel, ainsi que par une analyse de l'état actuel, à savoir la manière dont les processus gérés par le système sont actuellement mis en œuvre chez l'utilisateur.

La phase d'analyse consiste à énumérer les résultats attendus en termes de fonctionnalités, de performances, de robustesse, de maintenance, de sécurité, d'extensibilité, et autres. En parallèle, la phase de conception sert à décrire de manière claire, généralement à l'aide d'un langage de modélisation, le fonctionnement futur du système, facilitant ainsi sa mise en œuvre.

2. La méthode MERISE

2.1. Présentation

MERISE est une méthodologie de gestion de projets informatiques qui englobe la conception, le développement et la réalisation. Elle se fonde sur le principe de séparation des données et des traitements en différents modèles d'abstraction, ce qui garantit la pérennité du modèle. En général, les données subissent moins de modifications que les traitements. MERISE a vu le jour en 1978-1979 à la suite d'une consultation nationale organisée par le ministère de l'industrie français en 1977, dont l'objectif était de sélectionner les sociétés de conseil informatique chargées de définir la méthode de conception des systèmes d'informations.

2.2. Les aspects du cycle de développement

Le développement d'un projet selon la méthodologie MERISE repose sur trois éléments fondamentaux :

- la démarche ou cycle de vie ;
- le raisonnement ou cycle d'abstraction ;
- la maîtrise ou cycle de décision.

2.2.1. La démarche ou cycle de vie

La méthodologie MERISE définit les étapes suivantes pour mener à bien le cycle de vie d'un projet :

- le plan stratégique : il s'agit de la planification globale des projets à réaliser dans l'entreprise à moyen terme, cela inclut la définition de la qualité et de la quantité des ressources matérielles et humaines nécessaires, ainsi que les priorités à accorder aux différents projets ;



- l'étude préliminaire : cette phase a pour objectif d'évaluer la faisabilité du projet (pour décider s'il doit être poursuivi ou abandonné) et de déterminer s'il est viable ;
- l'étude approfondie : une fois la décision d'aller de l'avant prise, cette étape consiste à analyser en détail le système d'information pour mettre en évidence les règles de fonctionnement, telles que les mises à jour, les consultations, les calculs et les règles de gestion ;
- l'étude technique : cette phase permet de prendre en compte des éléments tels que le choix de la technologie, la nature de la base de données, les options de sécurité, les environnements de développement et les spécifications des équipements nécessaires ;
- la phase de maintenance : après l'installation, le logiciel est sujet à des erreurs (bugs) et aux évolutions constantes de l'environnement informatique, la maintenance vise à résoudre ces problèmes et à garantir le bon fonctionnement du système ;
- la remise en question : dans certains cas, les changements dans l'environnement technique peuvent être si significatifs que la simple maintenance ne suffit pas, il peut alors être nécessaire de revoir le cahier des charges du projet ou de décider de mettre fin définitivement à l'application.

2.2.2. Le raisonnement ou cycle d'abstraction

Le design du système d'information est un processus en plusieurs étapes visant à créer un système fonctionnel qui reflète fidèlement la réalité physique. Chaque étape doit être validée en prenant en compte les résultats de la phase précédente. De plus, il est essentiel de vérifier l'alignement entre les données et les traitements en s'assurant que toutes les données nécessaires aux traitements sont disponibles et qu'il n'y a pas de données superflues. Ce processus est connu sous le nom de "cycle d'abstraction" pour la conception des systèmes d'information et se compose de quatre niveaux distincts.

2.2.2.1. Niveau conceptuel

Il s'agit d'une étape qui concerne des décisions de gestion fondamentales visant à identifier des éléments stables en dehors des moyens, des contraintes et de l'organisation à utiliser. Cette phase répond essentiellement à la question "QUOI ?"

2.2.2.2. Niveau organisationnel

Cette étape englobe la prise de décisions relatives à l'organisation des ressources humaines et matérielles, en définissant les acteurs impliqués et les postes de travail correspondants. Elle répond aux questions "QUI ? OÙ ? QUAND ?".

2.2.2.3. Niveau logique

Cette phase implique la prise de décisions concernant les moyens et les ressources informatiques, sans entrer dans les détails techniques spécifiques. Elle se situe au niveau du modèle relationnel, du diagramme des classes et des diagrammes de séquence d'objets. Cette étape répond à la question "COMMENT ?".

2.2.2.4. Niveau physique

Cette phase représente les décisions techniques prises et leur mise en œuvre en tenant compte de leurs particularités. Elle se situe au niveau du code dans un langage de programmation spécifique, et elle vise à traduire concrètement les choix techniques en réalisations informatiques.

2.2.3. La maîtrise ou cycle de décision

Le cycle de décision englobe l'ensemble des décisions qui doivent être prises tout au long du cycle de vie d'un projet ou d'un système. Il inclut les choix stratégiques, tactiques et opérationnels nécessaires pour mener à bien le projet, résoudre les problèmes éventuels, et garantir l'atteinte des objectifs fixés.

3. PU/UML

3.1. Processus Unifié (PU)

Le Processus Unifié (PU) est une méthode de développement de logiciels qui adopte une approche itérative et orientée vers l'architecture. Elle repose sur l'utilisation de cas d'utilisation pour atténuer les risques associés au projet. L'objectif principal du PU est de maîtriser la complexité des projets informatiques en minimisant les risques. Cette méthode est composée d'un ensemble de principes génériques qui peuvent être adaptés en fonction des caractéristiques spécifiques de chaque projet.

Le PU est organisé en deux axes principaux : l'axe vertical, qui regroupe les activités en fonction de leur nature et qui décrit l'aspect statique du processus en termes de composants, de processus, d'activités, de séquences et de parties prenantes ; et l'axe horizontal, qui représente le temps et détaille le déroulement du cycle de vie du processus, en mettant en évidence les cycles, les phases et les itérations.

3.2. Unified Modeling Language (UML)

UML, acronyme de "Unified Modeling Language" (Langage de Modélisation Unifié en français), est un langage de modélisation graphique basé sur des pictogrammes. Il offre une méthode normalisée pour représenter la conception d'un système, notamment dans le contexte du développement logiciel et de la conception orientée objet. UML est largement reconnu comme un standard et est géré par l'Object Management Group (OMG).

3.3. Comparaison des méthodes d'analyse

Tableau 5: Comparaison MERISE-UML

Aspect	MERISE	UML
Utilisation principale	Conception de bases de données et systèmes d'information.	Conception de logiciels, systèmes d'information et modélisation de processus métier.
Méthode d'analyse	Utilise des schémas conceptuels, logiques et physiques pour représenter la structure des données.	Utilise des diagrammes de classe pour représenter la structure des données.
Diagramme principaux	Diagramme entité-relation, diagramme de Flux, diagramme de niveau.	Diagramme de flux, diagramme de niveau, diagramme de classe, diagramme de séquence, diagramme d'activité, etc.
Approche	Structurée, met l'accent sur la modélisation des données et des processus.	Orientée objet, met l'accent sur la modélisation des objets et de leurs interactions.



Formalisme	Moins formel, utilise des notations spécifiques à MERISE	Plus formel, utilise des notations normalisées dans l'industrie
Notations	Propres notations spécifiques à MERISE telles que les rectangles, les losanges, etc.	Utilise des notations normalisées comme les flèches, les stéréotypes, les classes.
Domaines d'application	Principalement orienté vers les systèmes d'information et les bases de données.	Applicable à un large éventail de domaines, y compris les logiciels, l'ingénierie des systèmes, le développement web.

II. CHOIX DE LA MÉTHODE D'ANALYSE

Dans le cadre de la réalisation de ce projet, nous avons jugé essentiel d'opter pour une méthode d'analyse adaptée afin de modéliser correctement les différents aspects du système à concevoir. Après avoir comparé en profondeur les méthodes MERISE et UML, notre choix s'est porté sur la méthode UML (Unified Modeling Language). En effet, UML s'est révélée particulièrement adaptée au développement de logiciels complexes comme une plateforme de Gestion Électronique de Documents (GED). Ce type de projet a impliqué la gestion de nombreux objets métier (courriers, utilisateurs, services, types de documents, etc.) et a nécessité la représentation des interactions entre ces objets. L'approche orientée objet de la méthode UML a permis de répondre efficacement à ce besoin. De plus, UML a proposé une grande variété de diagrammes tels que le diagramme de cas d'utilisation, le diagramme de classes, le diagramme d'activité ou encore le diagramme de séquence, qui ont permis de modéliser les fonctionnalités, les processus métier, ainsi que les interactions entre les différents composants du système. Cette richesse a outillé efficacement les étapes de conception de notre application.

En comparaison, la méthode MERISE, bien qu'ayant été efficace dans les projets de conception de bases de données et de systèmes d'information traditionnels, a été moins orientée vers la modélisation logicielle moderne. Elle s'est basée davantage sur une approche structurée, souvent moins flexible pour des systèmes interactifs nécessitant une modélisation comportementale précise.

Enfin, UML a été une méthode normalisée et largement utilisée dans l'industrie du développement logiciel, ce qui a facilité la collaboration avec des équipes de développement, d'intégration ou de maintenance futures.

CHAPITRE IV : CRÉATION DE L'ARCHITECTURE DU SYSTÈME

Dans ce chapitre, nous avons mis en œuvre la conception de notre système en appliquant la méthodologie UML.

I. IDENTIFICATION DES ACTEURS ET DES CAS D'UTILISATION

1. Les acteurs

Les acteurs sont des éléments essentiels dans la représentation des interactions avec le système. Ils peuvent prendre diverses formes, telles que des individus, des machines externes ou des interactions spécifiques avec le système. On distingue généralement deux catégories : les acteurs principaux, qui interagissent directement avec le système pour son utilisation, et les acteurs secondaires, qui assurent des tâches administratives et de maintenance pour garantir le bon fonctionnement du système pour les acteurs principaux. Les acteurs principaux et secondaires identifiés sont :

Tableau 6: Identification des acteurs

Acteurs	Description
Agent	Crée les courriers, les envoie, gère les bordereaux, consulte l'historique.
Destinataires : Directeur de Cabinet, Directeur de Cabinet Adjoint, IGSJP, Chef du Cabinet, Conseillers Techniques, Chargé d'Etudes, Chef du Secrétariat Particulier et Directeurs.	Reçoit les courriers, remplit les bordereaux, ajoute des instructions, renvoie les courriers.
Administrateur	Gère les utilisateurs, les droits d'accès et les logs du système.

2. Les cas d'utilisation

Pour l'Agent :

- créer un courrier ;
- vérifier unicité du courrier ;
- enregistrer dans registre de transcription ;
- générer le bordereau ;
- envoyer un courrier ;
- imputer un courrier ;
- consulter l'historique ;
- rechercher un courrier.

Pour le Destinataire :

- recevoir un courrier ;
- consulter le courrier imputé ;



- remplir le bordereau ;
- ajouter des instructions / imputations ;
- envoyer au service courrier.

Pour l'Administrateur :

- créer un compte utilisateur ;
- gérer les droits d'accès ;
- superviser les actions du système.

3. Diagramme des cas d'utilisation

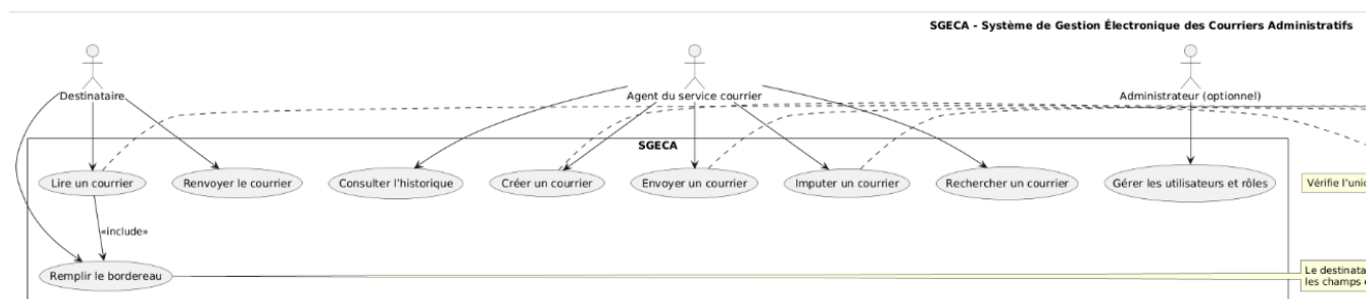


Figure 4: Diagramme des cas d'utilisation

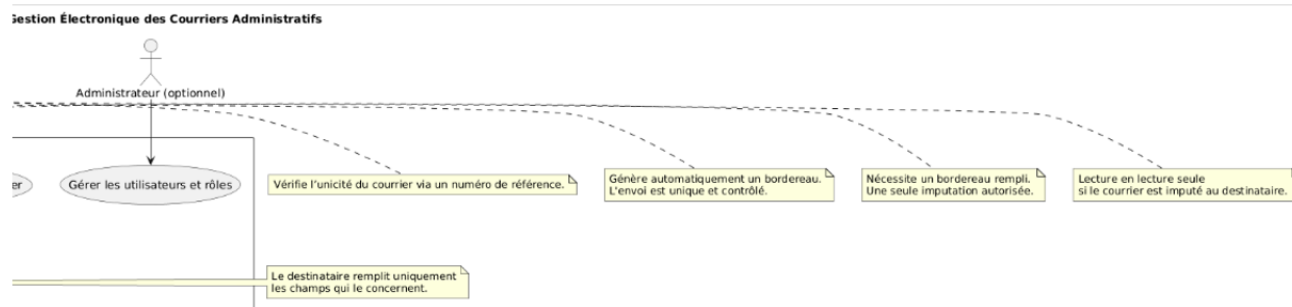


Figure 3: Diagramme des cas d'utilisation

4. Description textuelle des cas d'utilisation

4.1. Créer un courrier

Tableau 7: Description textuelle de "Créer un courrier"

Nom du Cas d'Utilisation	Créer un courrier
Acteur Principal	Agent du service courrier
But	Ajouter un nouveau courrier dans le système.
Préconditions	<ul style="list-style-type: none"> - l'agent est connecté et autorisé ; - le courrier n'existe pas encore.
Description	<ul style="list-style-type: none"> - l'agent saisit les informations nécessaires à la création d'un courrier (référence, objet, type, expéditeur, etc.) ; - le système vérifie que le numéro de référence est unique afin d'éviter les doublons.
Postconditions	Le courrier est enregistré dans la base de données avec un statut initial.

4.2. Envoyer un courrier

Tableau 8: Description textuelle de "Envoyer un courrier"

Nom du Cas d'Utilisation	Envoyer un courrier
Acteur Principal	Agent du service courrier
But	Transmettre un courrier à un ou plusieurs destinataires.
Préconditions	<ul style="list-style-type: none"> - le courrier existe ; - l'agent a sélectionné les destinataires.
Description	<p>Une fois le courrier créé, l'agent peut le transmettre.</p> <p>Lors de l'envoi, le système génère automatiquement un bordereau contenant les métadonnées de transmission.</p> <p>L'envoi est contrôlé : un même courrier ne peut être envoyé plusieurs fois.</p>
Postconditions	<ul style="list-style-type: none"> - le bordereau est généré ; - le courrier est marqué comme "envoyé".



4.3. Imputer un courrier

Tableau 9: Description textuelle de "Imputer un courrier"

Nom du Cas d'Utilisation	Imputer un courrier
Acteur Principal	Agent du service courrier
But	Affecter un courrier à un destinataire spécifique.
Préconditions	<ul style="list-style-type: none"> - le bordereau est complet ; - le courrier n'a pas encore été imputé.
Description	<p>L'imputation consiste à assigner officiellement un courrier à une personne responsable de son traitement.</p> <p>Le système n'autorise l'imputation que si le bordereau est rempli, et un courrier ne peut être imputé qu'une seule fois.</p>
Postconditions	<ul style="list-style-type: none"> - le courrier est marqué comme "imputé" ; - le destinataire devient responsable de son traitement.

4.4. Rechercher un courrier

Tableau 10: Description textuelle de "Rechercher un courrier"

Nom du Cas d'Utilisation	Rechercher un courrier
Acteur Principal	Agent du service courrier
But	Trouver un courrier existant.
Préconditions	L'agent est connecté.
Description	L'agent utilise des critères de recherche (mot-clé, date, numéro de référence, statut...) pour retrouver un courrier.
Postconditions	Les résultats correspondant aux critères sont affichés.

4.5. Consulter l'historique

Tableau 11: Description textuelle de "Consulter l'historique"

Nom du Cas d'Utilisation	Consulter l'historique
Acteur Principal	Agent du service courrier
But	Visualiser les actions réalisées sur un courrier.
Préconditions	Le courrier existe.



Description	L'agent peut consulter l'historique d'un courrier (création, envoi, imputation, lecture...).
	Cela permet un suivi des opérations effectuées.
Postconditions	L'historique des événements est affiché.

4.6. Lire un courrier

Tableau 12: Description textuelle de "Lire un courrier"

Nom du Cas d'Utilisation	Lire un courrier
Acteur Principal	Destinataire
But	Accéder au contenu d'un courrier qui lui a été imputé.
Préconditions	Le courrier est imputé au destinataire.
Description	Le destinataire peut consulter le courrier en lecture seule. Le système s'assure que ce courrier lui est bien imputé.
Postconditions	Le contenu est affiché sans modification possible.

4.7. Remplir le bordereau

Tableau 13: Description textuelle de "Remplir le bordereau"

Nom du Cas d'Utilisation	Remplir le bordereau
Acteur Principal	Destinataire
But	Compléter les champs du bordereau transmis avec le courrier.
Préconditions	Le courrier lui a été envoyé.
Description	Le destinataire remplit uniquement les champs spécifiques qui le concernent dans le bordereau (ex : avis, date de lecture...).
Postconditions	Le bordereau est partiellement ou totalement complété.



4.8. Renvoyer le courrier

Tableau 14: Description textuelle de "Renvoyer le courrier"

Nom du Cas d'Utilisation	Renvoyer le courrier
Acteur Principal	Destinataire
But	Retourner un courrier au service ou le transmettre à un autre destinataire.
Préconditions	Le courrier est en sa possession (imputé).
Description	Après lecture, le destinataire peut décider de renvoyer le courrier pour traitement ou annotation supplémentaire.
Postconditions	Le statut du courrier est mis à jour (ex : renvoyé, en traitement...).

4.9. Gérer les utilisateurs et rôles

Tableau 15: Description textuelle de "Gérer les utilisateurs"

Nom du Cas d'Utilisation	Renvoyer le courrier
Acteur Principal	Administrateur
But	Créer, modifier ou supprimer des utilisateurs, et attribuer des rôles.
Préconditions	L'administrateur est connecté avec les droits nécessaires.
Description	L'administrateur a accès à une interface pour gérer tous les utilisateurs (agents, destinataires...) et leur attribuer des rôles selon leurs fonctions.
Postconditions	La base des utilisateurs est mise à jour.

II. DIAGRAMMES DE SÉQUENCE

1. Créer un courrier

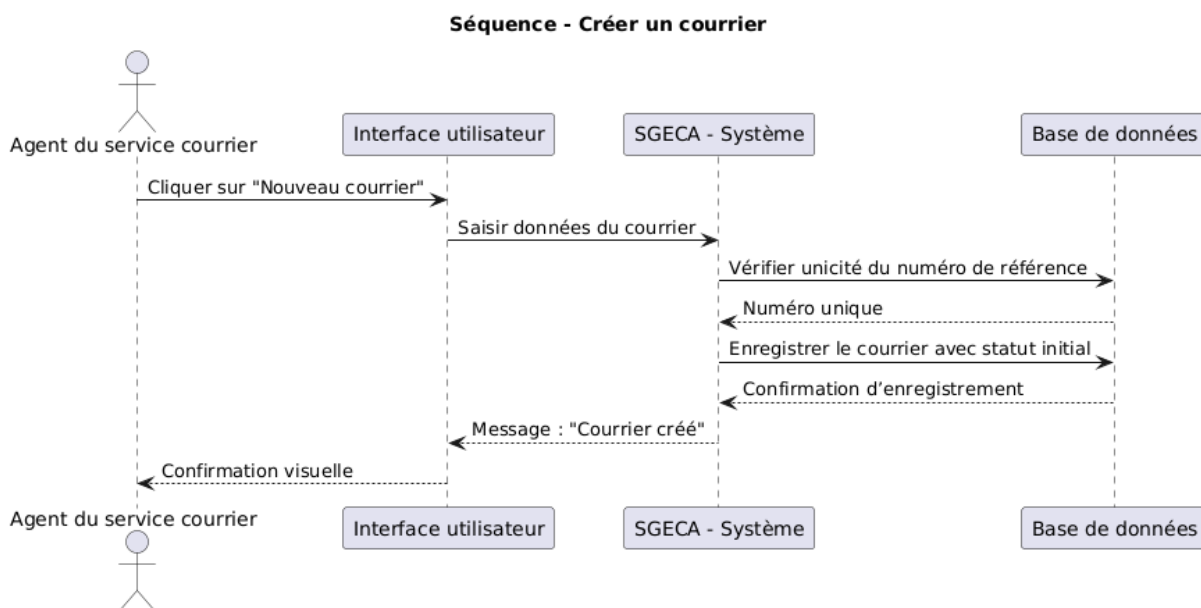


Figure 5: Diagramme de séquence "Créer un courrier"

2. Envoyer un courrier

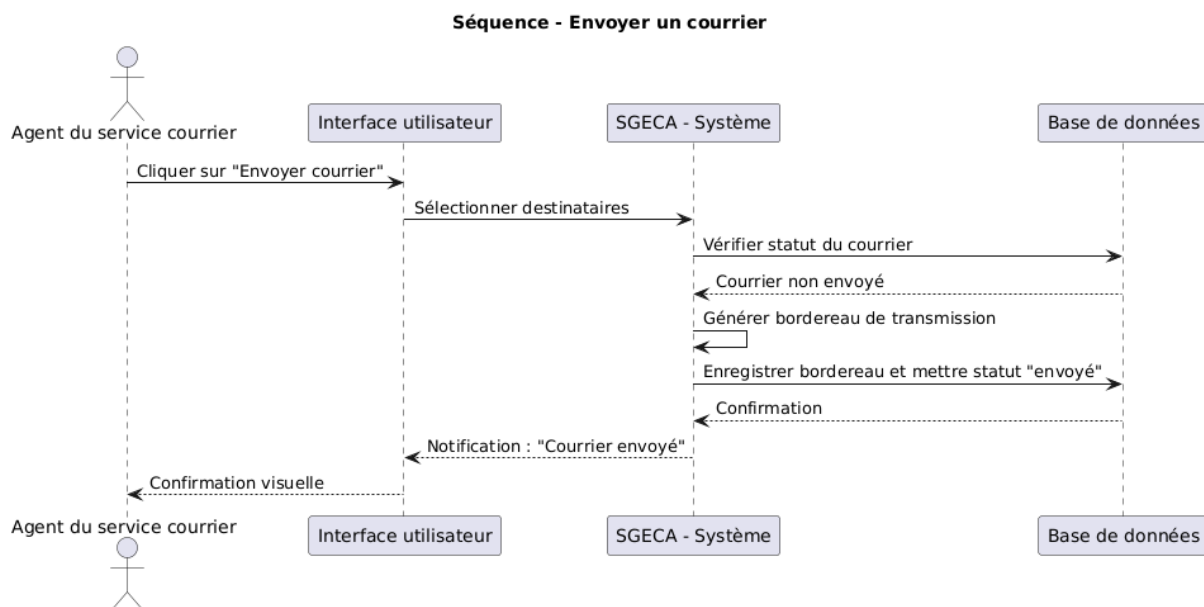


Figure 6: Diagramme de séquence "Envoyer un courrier"

3. Imputer un courrier

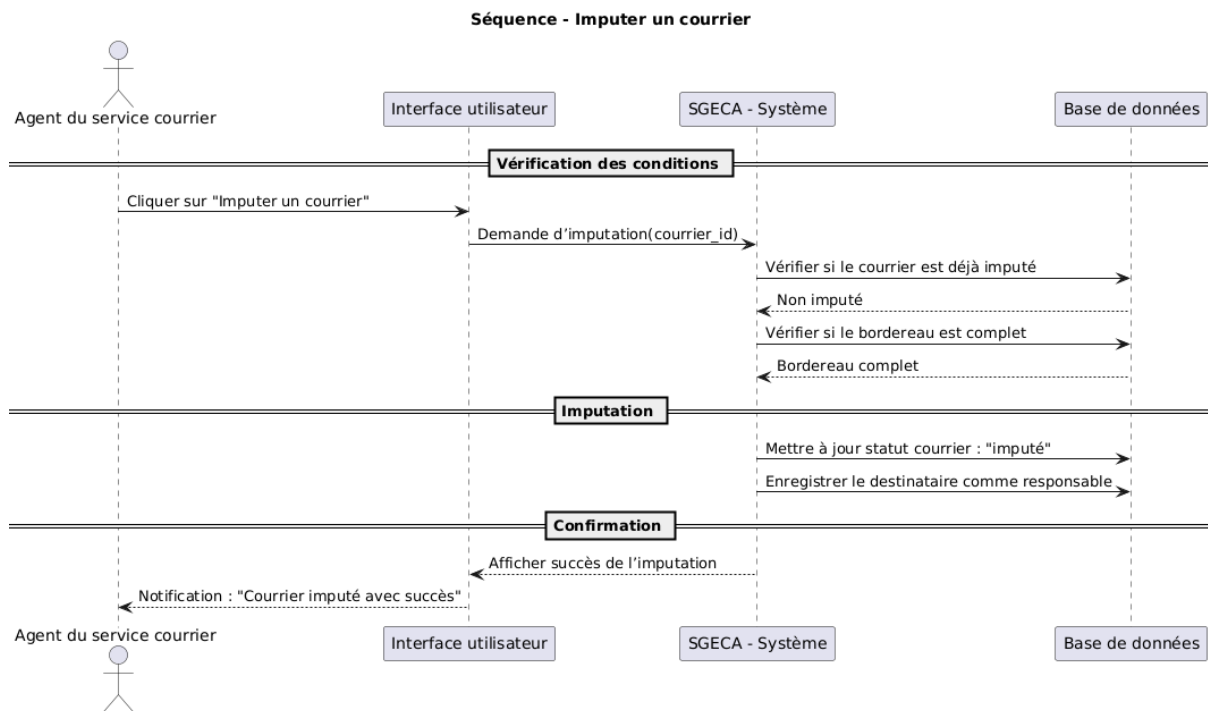


Figure 7: Diagramme de séquence "Imputer un courrier"

4. Rechercher un courrier

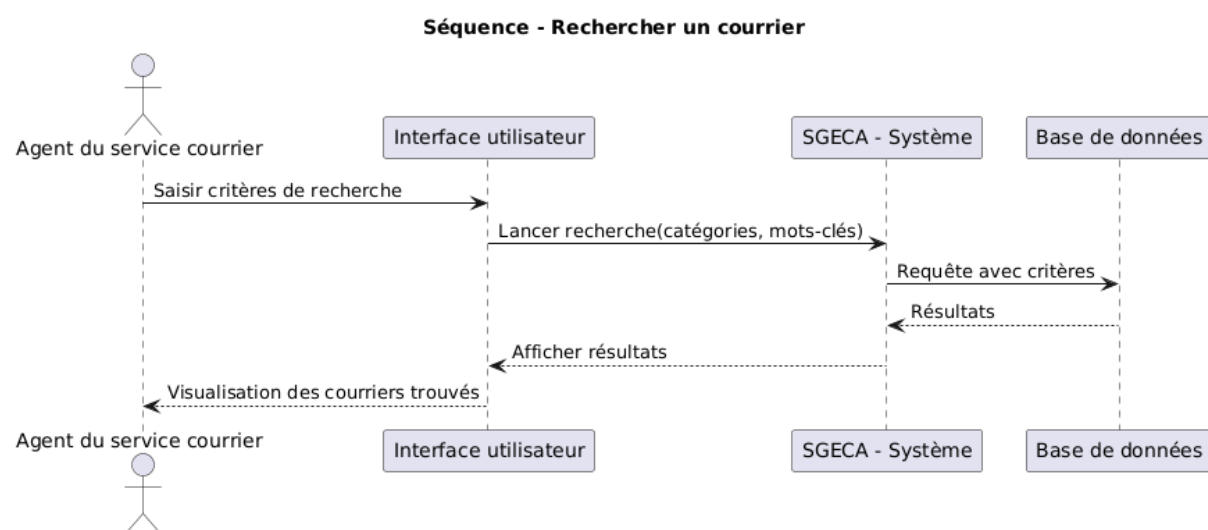


Figure 8: Diagramme de séquence "Rechercher un courrier"

5. Consulter l'historique

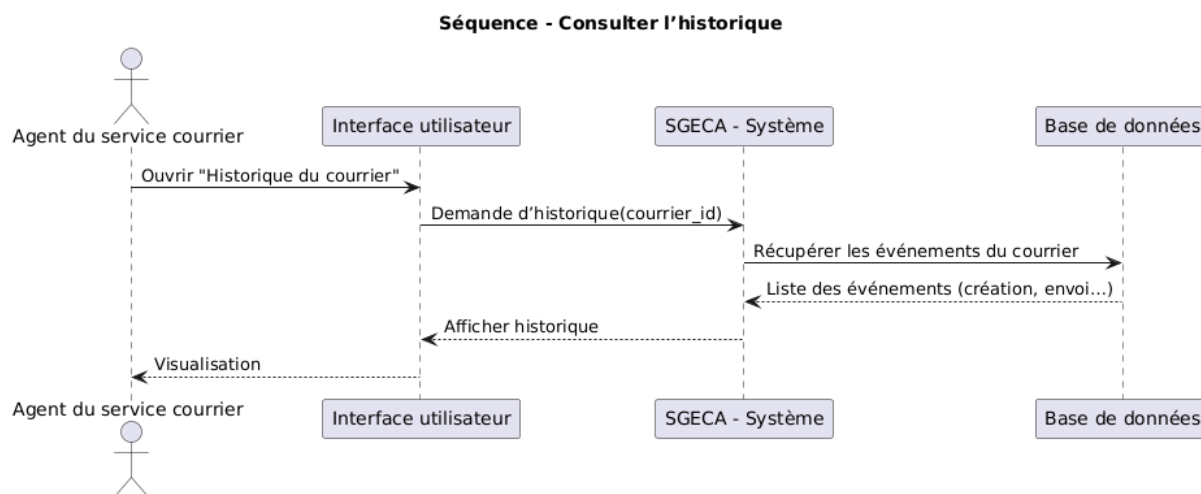


Figure 9: Diagramme de séquence "Consulter l'historique"

6. Lire un courrier

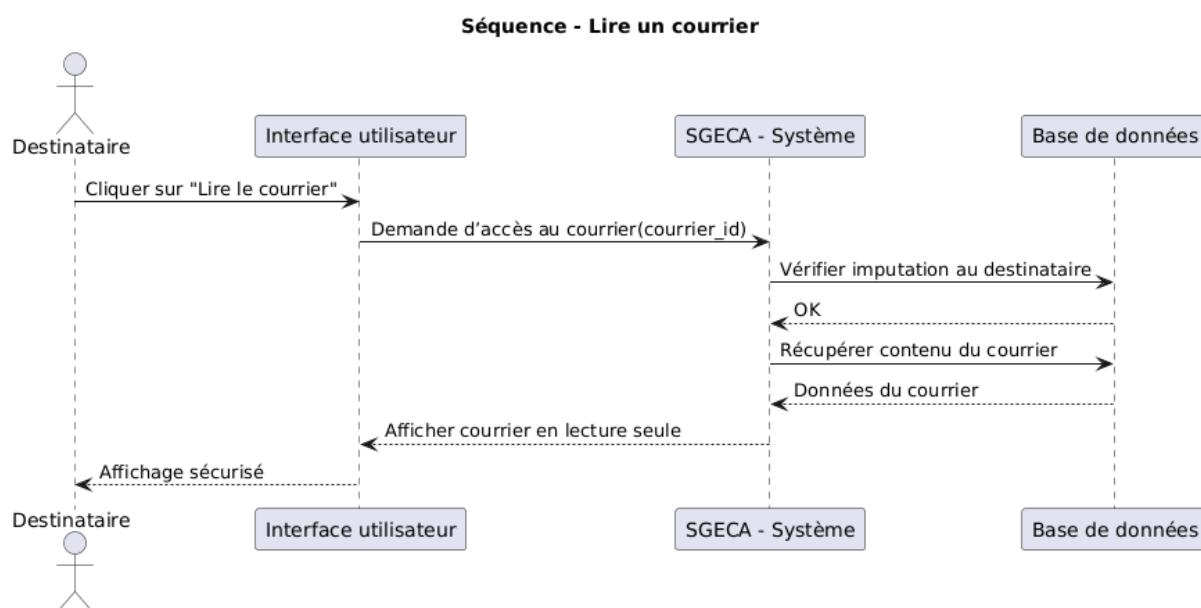


Figure 10: Diagramme de séquence "Lire un courrier"

7. Remplir le bordereau

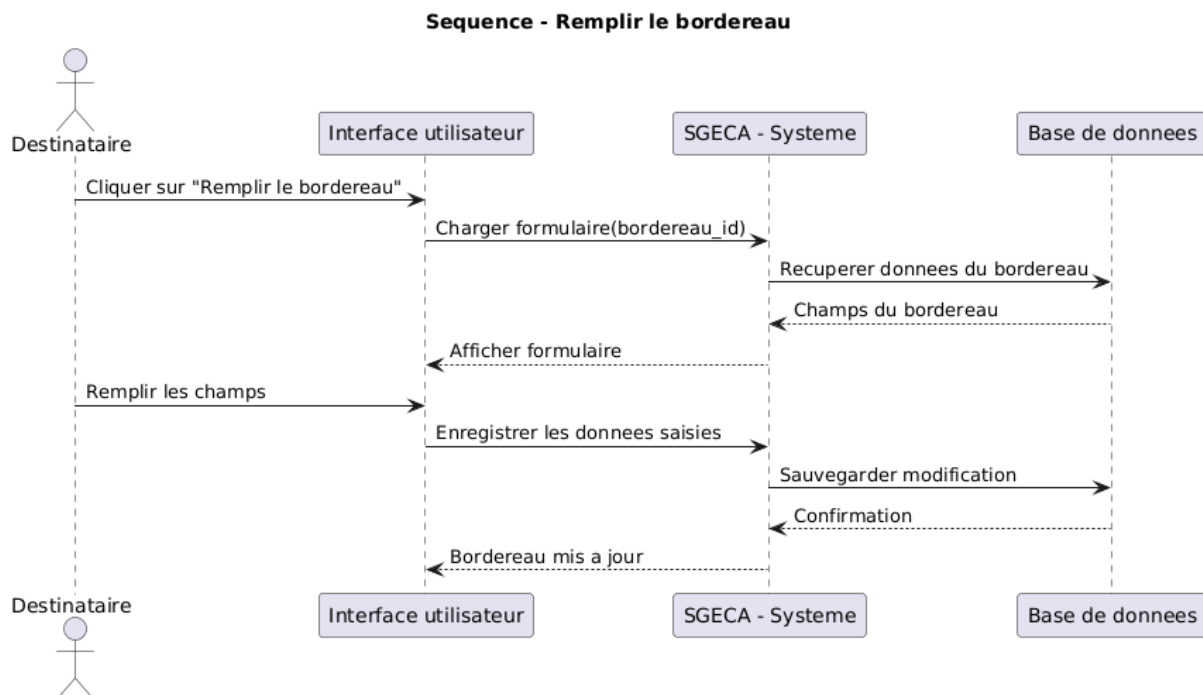


Figure 11: Diagramme de séquence "Remplir le bordereau"

8. Renvoyer le courrier

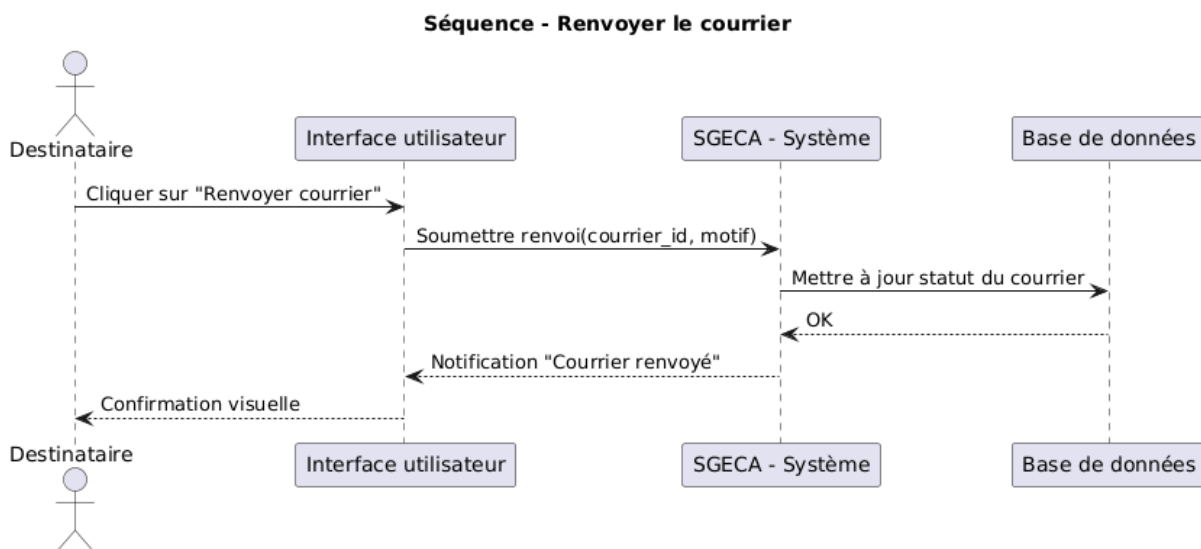


Figure 12: Diagramme de séquence "Renvoyer le courrier"



9. Gérer les utilisateurs et rôles

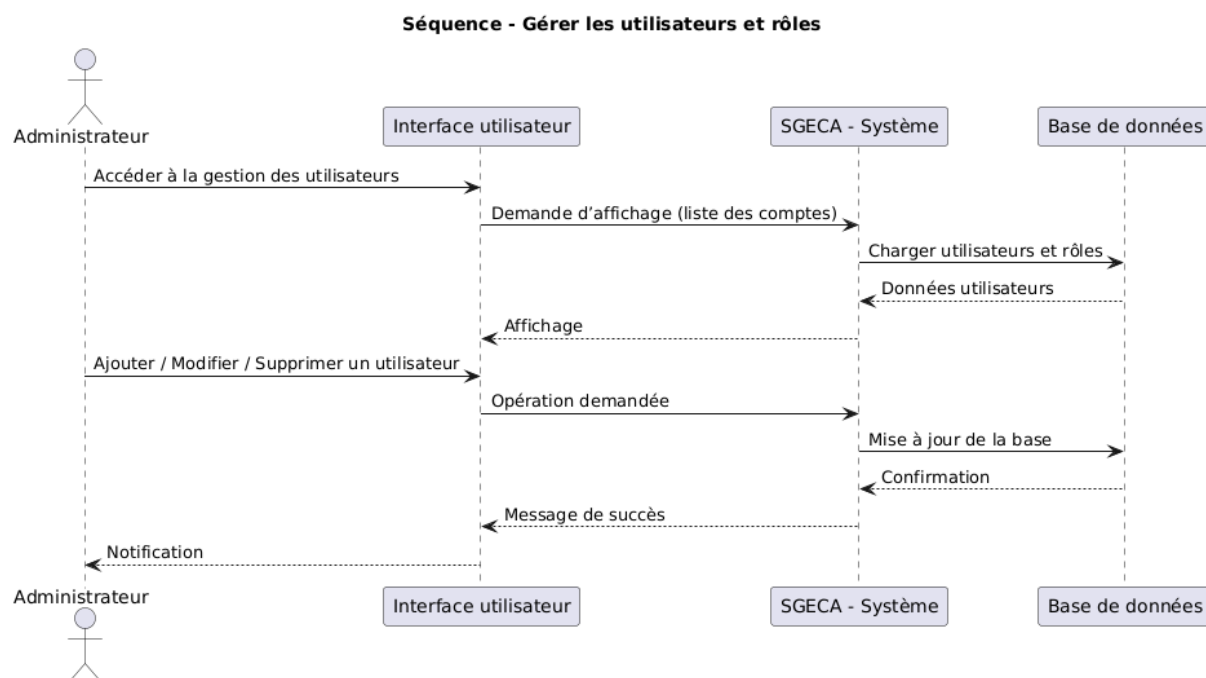


Figure 13: Diagramme de séquence "Gérer les utilisateurs et les rôles"

III. DIAGRAMME DE CLASSES

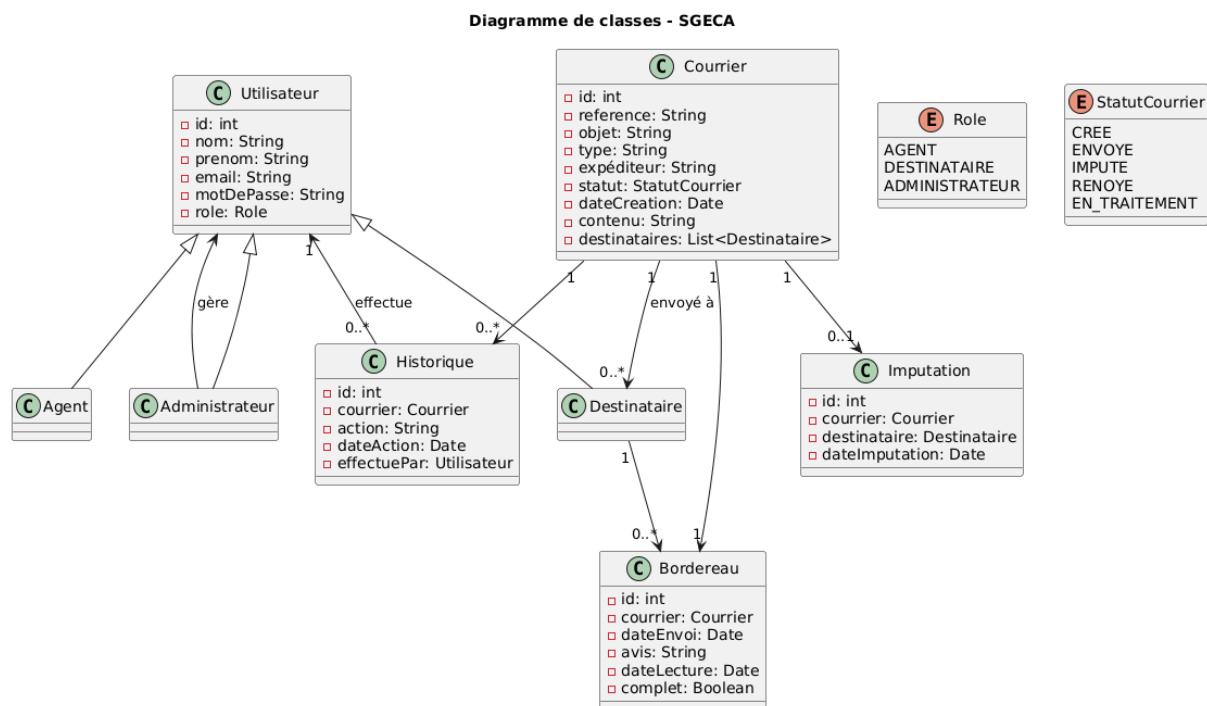


Figure 14: Diagramme des classes



PARTIE III : MISE EN ŒUVRE

Dans cette section, nous avons présenté les éléments et les ressources qui ont été utilisés pour accomplir cette tâche, ainsi que pour réaliser notre système avec succès.

CHAPITRE V : ÉTUDE TECHNIQUE

Dans cette section, nous avons détaillé les composants et les ressources que nous avons mobilisés pour accomplir cette tâche et pour concrétiser notre système.

I. COMPARAISON ET CHOIX D'OUTILS

Pour faire du développement web, nous utilisons divers outils, dont un environnement de développement, un serveur local, un framework web, et une base de données.

1. L'environnement de développement

L'environnement de développement désigne l'ensemble des outils, éditeurs, plateformes et configurations utilisés par un développeur pour écrire, tester et maintenir le code source d'une application. Il inclut généralement :

- un éditeur de code ou IDE (Environnement de Développement Intégré) ;
- des outils de gestion de versions (comme Git) ;
- des extensions ou plugins utiles (auto-complétion, débogage, linters) ;
- parfois une interface de ligne de commande, un terminal intégré ou une connexion à des conteneurs/Docker.

Le bon choix d'environnement améliore la productivité, la lisibilité du code, et facilite le débogage et la collaboration.

Comparaison de quelques environnements de développement :

Tableau 16: Comparaison de quelques environnements de développement

Environnement	Type	Points forts	Limites ou inconvénients
VS Code	Éditeur léger	Rapide, personnalisable, large écosystème d'extensions, Git intégré	Nécessite une configuration initiale
IntelliJ IDEA	IDE complet	Outils puissants pour Java, prise en charge avancée des frameworks	Plus lourd, consommation de ressources
PyCharm	IDE spécialisé Python	Debugger puissant, support Django/Flask intégré	Version gratuite limitée
Sublime Text	Éditeur minimaliste	Ultra-rapide, interface épurée	Moins d'intégrations natives que VS Code
Eclipse	IDE Java	Stable, bon pour gros projets Java	Interface datée, peu intuitif
WebStorm	IDE JS professionnel	Outils intégrés pour React, Node, test, linting...	Payant



Pour notre projet, nous avons choisi Visual Studio Code (VS Code) comme environnement de développement principal. Ce choix repose sur les critères suivants :

- léger, rapide et multiplateforme, VS Code s'installe facilement et fonctionne bien même sur des configurations modestes ;
- il dispose d'un écosystème très riche d'extensions, ce qui permet d'ajouter des fonctionnalités comme : le support du linting JavaScript/Python, les outils de formatage automatique (Prettier, ESLint), l'intégration Git avec vue des commits, branches et push/pull, des extensions pour PostgreSQL, REST Client, et Docker ;
- il offre aussi un terminal intégré, une navigation facile dans les fichiers du projet, ainsi qu'un débogueur puissant ;
- enfin, VS Code est open-source et gratuit, ce qui le rend accessible à toute l'équipe sans contrainte de licence.

Cette combinaison de souplesse, extensibilité et performance en fait un excellent choix pour le développement web moderne, que ce soit en frontend, backend ou en gestion de base de données.



Figure 15: Logo de Visual Studio Code

2. Serveur web et hébergement

Un serveur web est un logiciel ou un service qui permet de stocker, traiter et distribuer du contenu web aux utilisateurs, généralement via le protocole HTTP. Il joue un rôle central dans l'architecture d'une application web, puisqu'il est responsable de :

- la réception des requêtes des clients (navigateurs, applications) ;
- le traitement de ces requêtes côté backend (API, base de données) ;
- et le renvoi des réponses au client (pages web, JSON, fichiers...).

Dans le cadre du développement web, plusieurs types de serveurs peuvent être utilisés selon les besoins :

- serveurs de développement : installés localement pour tests pendant la phase de développement ;
- serveurs de test/staging : environnements intermédiaires pour validation avant production ;
- serveurs de production : serveurs publics accessibles aux utilisateurs finaux ;



- pour rendre ces serveurs accessibles via Internet, on utilise des solutions d'hébergement, locales ou cloud.

Comparaison de quelques serveurs web et solutions d'hébergement.

Tableau 17: Comparaison de quelques serveurs web et solutions d'hébergement

Solution / Plateforme	Type	Points forts	Limites ou inconvénients
XAMPP	Serveur local	Environnement complet (Apache, PHP, MySQL), idéal pour tests locaux	Configuration manuelle, limité au développement
WAMPServer	Serveur local (Windows)	Similaire à XAMPP, interface graphique intuitive	Moins flexible, réservé à Windows
Caddy	Serveur web moderne	HTTPS automatique intégré, léger, performant	Moins connu, nécessite une courbe d'apprentissage
Render	Plateforme d'hébergement cloud	Déploiement facile, HTTPS, déploiement continu, support backend/frontend	Dépendance à Internet, moins contrôlable localement
Backblaze B2	Stockage cloud (fichiers)	Idéal pour stocker des fichiers volumineux (PDF, images, backups)	Ne gère pas les requêtes dynamiques
Supabase	Backend-as-a-Service	Authentification, base de données, API auto	Moins personnalisable, dépendance forte au cloud

Pour notre projet, nous avons fait le choix de combiner deux services complémentaires :

Render comme serveur web cloud principal :

- il nous permet de déployer une API backend Node.js (Express) ou toute autre application facilement, avec une gestion automatique du serveur, des déploiements continus (CI/CD) et des certificats HTTPS sécurisés ;
- il prend en charge l'ensemble de l'infrastructure cloud (machine virtuelle, réseau, monitoring) ce qui nous libère des tâches complexes liées à la maintenance serveur ;
- le choix répond à notre besoin de mise en ligne rapide, scalable et professionnelle de notre application web, accessible 24/7.



Figure 16: Logo de Render

Backblaze B2 comme solution de stockage de fichiers

- il n'est pas un serveur web en soi, mais une solution idéale de stockage pour les ressources volumineuses de notre projet : fichiers PDF, images, backups, etc ;
- les fichiers y sont conservés de manière sécurisée, scalable et accessibles via API, ce qui permet de les intégrer dynamiquement dans notre application hébergée sur Render.

Ce choix technologique répond à un équilibre optimal entre flexibilité, performance et simplicité de gestion.



Figure 17: Logo de BACKBLAZE

3. Le front-end et le back-end

Dans le développement d'une application web, on distingue généralement deux grandes parties:

Frontend (interface utilisateur)

Le frontend correspond à la partie visible de l'application, avec laquelle l'utilisateur interagit directement. Il s'agit principalement :

- de l'interface graphique (pages, boutons, formulaires, menus...) ;
- de la logique côté client (validation de formulaire, affichage dynamique...) ;
- de la communication avec l'API backend via HTTP (AJAX, fetch, etc.).

Il s'appuie sur des langages comme HTML, CSS et JavaScript, ainsi que des frameworks modernes (React, Vue, Angular...).

Backend (logique métier et traitement serveur)

Le backend est la partie invisible côté serveur, chargée de :

- gérer la logique métier (authentification, traitements des données) ;
- communiquer avec les bases de données ;
- exposer des API REST ou GraphQL utilisées par le frontend.

Il est développé à l'aide de langages serveur comme Node.js, Python, PHP, Java, etc.

Comparatif des technologies frontend / backend :

Tableau 18: Comparatif des technologies frontend/ backend

Technologie	Rôle	Points forts	Limites ou inconvénients
HTML/CSS/JS	Frontend	Standard du web, simple à intégrer, universel	Nécessite frameworks pour projets complexes
React.js	Frontend	Composants réutilisables, SPA performant, communauté très active	Courbe d'apprentissage pour les débutants
Vue.js	Frontend	Simple à prendre en main, efficace pour petits projets	Moins utilisé que React dans les grandes entreprises
Node.js + Express	Backend	Léger, rapide, full JavaScript (frontend/backend), idéal pour API REST	Nécessite gestion manuelle de la structure
Django (Python)	Backend	Rapide à développer, très structuré, inclut ORM et admin	Plus rigide, lourd pour petits projets
Laravel (PHP)	Backend	Framework complet et moderne, gestion simple de la base de données	Langage PHP moins en vogue

Frontend (HTML, CSS et JavaScript) :

- nous avons opté pour une interface simple et légère, adaptée à notre besoin fonctionnel ;
- le choix de HTML/CSS/JS natif permet de garantir compatibilité universelle et de garder le contrôle total sur la structure de l'interface ;
- le choix est également cohérent avec une phase de prototypage ou MVP, où l'objectif est d'avoir un frontend fonctionnel sans complexité inutile.



Backend (Node.js avec Express) :

- le backend est développé en JavaScript avec Express.js, un framework minimaliste qui nous permet de créer une API REST rapidement ;
- l'un des avantages de cette solution est la cohérence des langages entre le frontend et le backend (tout en JavaScript), ce qui facilite le développement ;
- Express est particulièrement adapté aux projets d'API connectées à une base de données et déployées sur le cloud, comme c'est notre cas avec Render.

4. Le système de gestion de la base de données (SGBD)

Dans le développement d'applications web, un Système de Gestion de Base de Données (SGBD) est un composant essentiel qui permet de stocker, organiser, interroger et manipuler les données utilisées par l'application. Le SGBD est généralement connecté au backend de l'application, et communique via un langage de requête, le plus courant étant le SQL (Structured Query Language).

Il existe plusieurs types de SGBD, chacun ayant ses avantages et cas d'usage spécifiques :

- les SGBD relationnels (comme MySQL, PostgreSQL, SQLite) utilisent des tables avec des relations définies entre les données ;
- les SGBD NoSQL (comme MongoDB, Firebase) sont plus flexibles pour des structures de données non tabulaires et évolutives.

Le choix du SGBD dépend de critères comme la structure des données, la performance, la scalabilité, la sécurité et la compatibilité avec le reste du projet.

Comparaison des SGBD couramment utilisés :

Tableau 19: Comparaison des SGBD courants

SGBD	Type	Points forts	Limites
MySQL	Relationnel	Rapide, bien documenté, large communauté	Moins rigoureux sur l'intégrité des données
PostgreSQL	Relationnel	Très robuste, respecte les standards SQL, support avancé	Configuration parfois plus complexe
SQLite	Relationnel	Léger, sans serveur, idéal pour les petits projets	Pas adapté aux projets à grande échelle
MongoDB	NoSQL	Flexible, bon pour les données non structurées	Pas de support SQL natif, cohérence à gérer soi-même
Firebase	NoSQL cloud	Temps réel, facile à intégrer avec apps mobiles	Moins de contrôle sur les requêtes complexes

Pour notre projet, nous avons opté pour PostgreSQL comme système de gestion de base de données, accompagné de PgAdmin4 comme outil de gestion visuelle. Ce choix repose sur plusieurs raisons :

- PostgreSQL est reconnu pour sa robustesse, sa conformité stricte aux standards SQL, et sa capacité à gérer des données complexes et volumineuses.
- il prend en charge des fonctionnalités avancées telles que les transactions ACID, les triggers, les fonctions stockées et les types personnalisés, ce qui est un atout dans une architecture backend solide ;
- l'outil PgAdmin4 offre une interface web conviviale pour visualiser les tables, exécuter des requêtes SQL, gérer les utilisateurs et administrer la base facilement sans ligne de commande.

Ce choix garantit la fiabilité, la sécurité et la scalabilité des données de notre application tout au long de son cycle de vie.

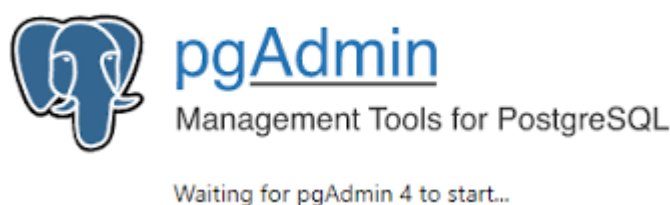


Figure 18: Logo de PgAdmin

5. Outils de test d'API

Dans le développement d'applications web modernes, les API (Interfaces de Programmation d'Applications) jouent un rôle fondamental. Elles permettent à différentes parties d'une application (frontend, backend, services tiers) de communiquer entre elles via des requêtes HTTP, en échangeant des données souvent au format JSON. Tester les API est une étape cruciale pour s'assurer du bon fonctionnement de la logique métier, de la communication avec la base de données, et de la sécurité des endpoints exposés.

Pour cela, plusieurs outils ont été conçus afin de simuler ces requêtes, observer les réponses et détecter d'éventuelles erreurs côté serveur ou client.

Tableau 20: Quelques outils de test d'API

Outil	Utilisation principale	Points forts	Limites
Postman	Test et documentation d'API REST	Interface intuitive, support des collections, gestion des tokens	Moins adapté aux tests automatisés de production

Insomnia	Alternative plus légère	Interface fluide, import facile de fichiers Postman	Moins d'extensions et d'intégration
curl	Requêtes via ligne de commande (terminal)	Léger, scriptable, intégré à tous les systèmes Unix	Nécessite une bonne maîtrise en ligne de commande

Dans notre projet, nous avons choisi Postman comme outil principal de test des API. Ce choix s'explique par sa facilité d'utilisation, sa puissance dans la gestion des collections de requêtes, et la possibilité d'y ajouter des scénarios d'authentification, des headers personnalisés, et de tester les routes dynamiques de manière simple et visuelle. Postman nous a permis de valider le bon fonctionnement de nos routes backend hébergées sur Render, et d'interagir efficacement avec notre base de données en simulant différents cas d'usage.



Figure 19: Logo de Postman

6. Récapitulatif des choix

Pour réaliser le projet, nous avons finalement choisi :

- Visual Studio Code, pour sa légèreté, ses extensions pratiques (linting, Git, etc.) et sa communauté très active ;
- Render pour l'hébergement cloud de l'application (backend, frontend, BDD) avec HTTPS et déploiement continu ;
- Backblaze B2 pour le stockage sécurisé des fichiers volumineux accessibles via API ;
- frontend : HTML, CSS et JavaScript pour une interface simple, efficace, légère ;
- backend : Node.js avec Express pour sa rapidité, flexibilité et cohérence full JavaScript ;
- PostgreSQL avec pgAdmin 4 pour ses capacités avancées, sa fiabilité, son support du SQL standard et sa bonne intégration avec Node.js ;
- Postman utilisé pour tester les routes de l'API, envoyer des requêtes personnalisées, visualiser les réponses, et organiser les collections de tests.

II. ARCHITECTURE LOGICIELLE

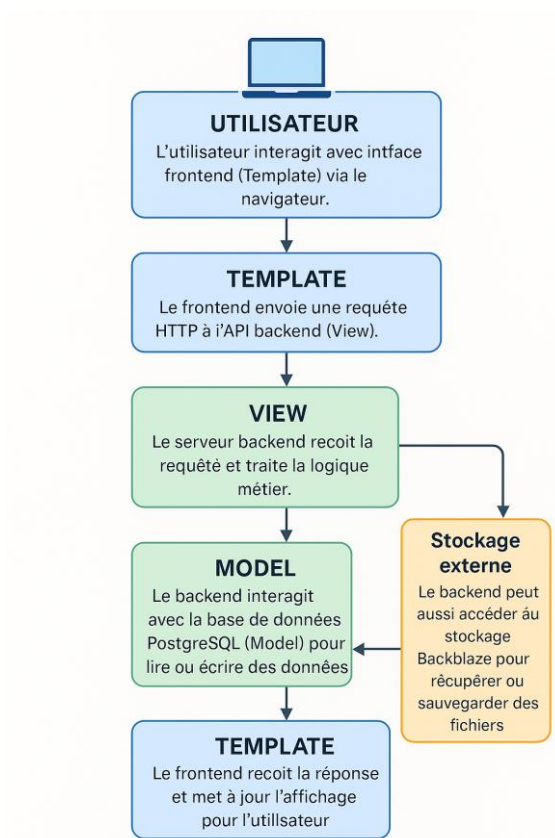


Figure 20: Architecture logicielle

C'est une architecture de type Modèle-Vue-Templates (MVT). Pour mieux comprendre cette architecture, voici une explication en une ligne pour chaque étape du déroulement de l'application selon l'architecture MVT :

- l'utilisateur interagit avec l'interface frontend (Template) via le navigateur ;
- le frontend envoie une requête HTTP à l'API backend (View) ;
- le serveur backend reçoit la requête et traite la logique métier ;
- le backend interagit avec la base de données PostgreSQL (Model) pour lire ou écrire des données ;
- le backend peut aussi accéder au stockage Backblaze pour récupérer ou sauvegarder des fichiers ;
- le backend prépare la réponse (données JSON, pages HTML, etc.) et la renvoie au frontend ;
- le frontend reçoit la réponse et met à jour l'affichage pour l'utilisateur.

CHAPITRES VI : REALISATION

Dans cette partie, nous présentons les différentes interfaces réalisées.

1. L'interface de connexion

Elle est identique pour tous les utilisateurs et l'administrateur.

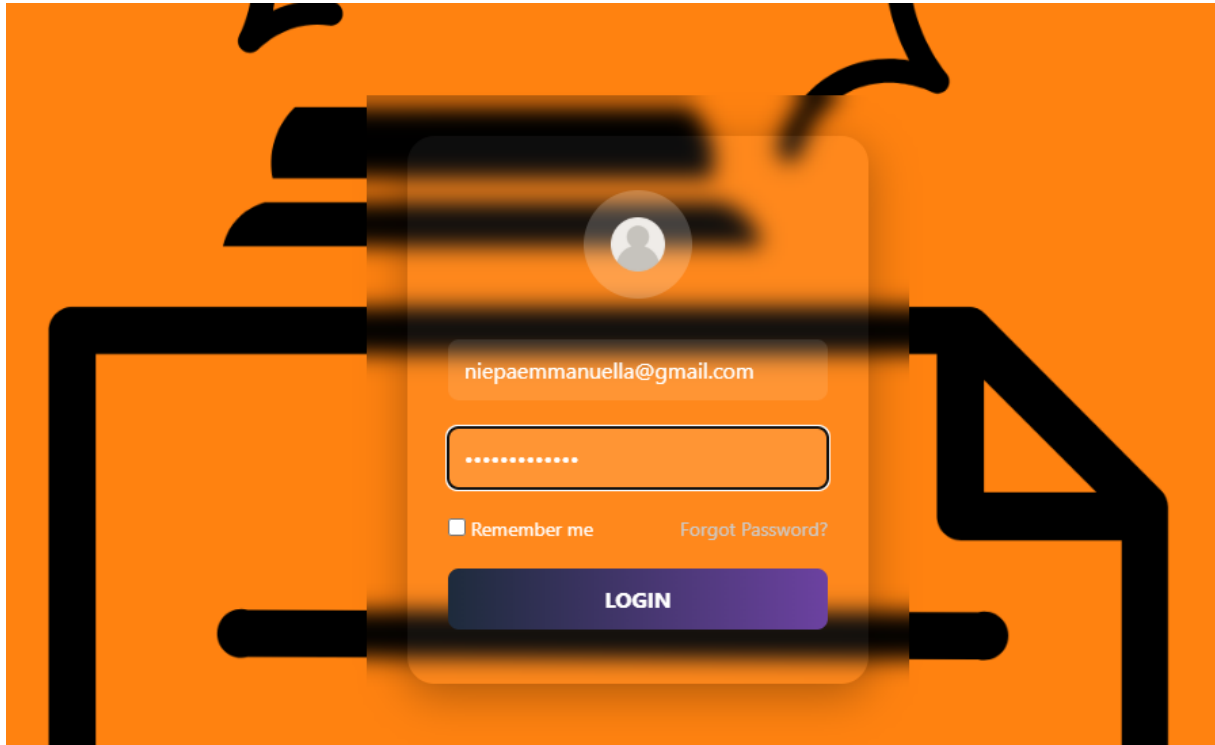


Figure 21: Page de connexion

2. Agent courrier

2.1. Page d'accueil

Cette page présente le dashboard pour un agent du service courrier.



The screenshot shows the JURIMAIL dashboard. On the left is a dark sidebar with the JURIMAIL logo and a menu with options: 'Créer un courrier', 'Imputer un courrier', 'Envoyer un courrier', 'Historique', and 'Mes Courriers'. The main area has a search bar at the top with the text 'Rechercher un courrier...'. Below the search bar, it says 'Bienvenue dans JURIMAIL ! Votre application de gestion de courriers administratifs.' and 'Sélectionnez une action dans le menu à gauche.' There is a 'Déconnexion' button in the top right corner.

Figure 22: Page d'accueil d'un agent

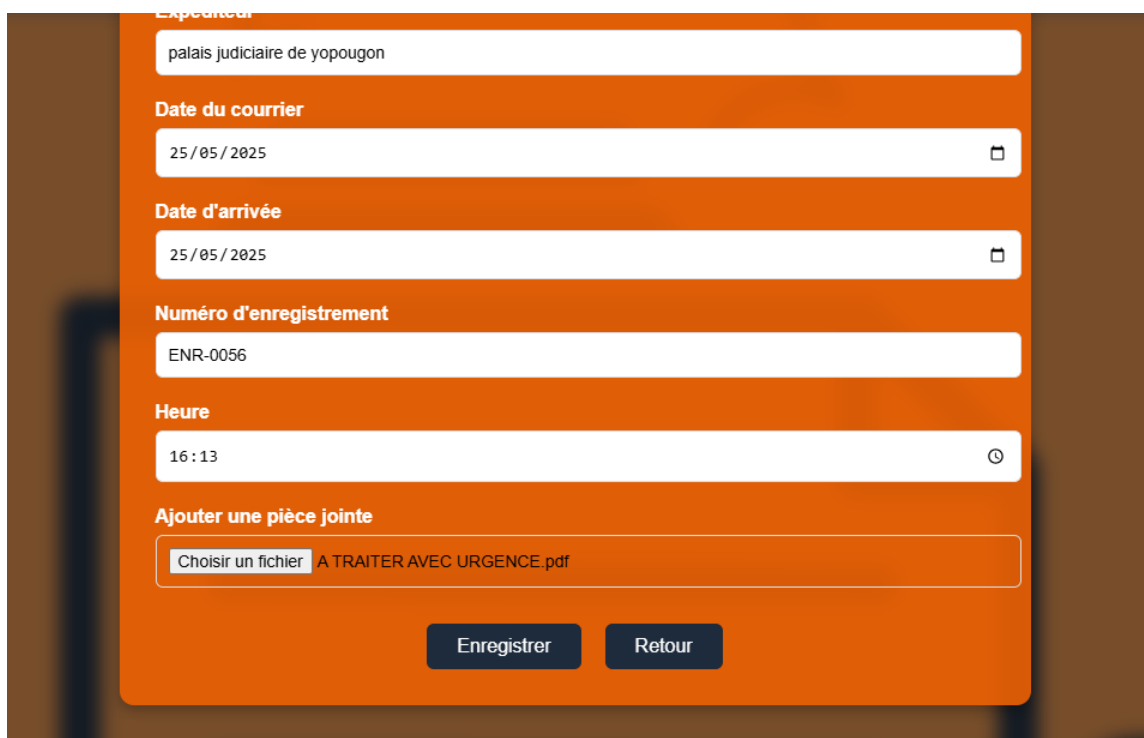
2.2. Page de création d'un courrier

Cette page présente le formulaire de création d'un courrier.



The screenshot shows the 'Créer un courrier' form. It has a title 'Créer un courrier' at the top. Below the title are several input fields: 'Référence' with the value 'REF-2025-0012', 'Objet' with the value 'Demande d'audience', 'Expéditeur' with the value 'palais judiciaire de yopougon', 'Date du courrier' with the value '25/05/2025' and a calendar icon, 'Date d'arrivée' with the value '25/05/2025' and a calendar icon, and 'Numéro d'enregistrement' with the value 'FNR-0056'.

Figure 23: Formulaire de création d'un courrier



The form is titled 'Expéditeur' and is set against an orange background. It contains several input fields: 'palais judiciaire de yopougon' in the 'Expéditeur' field, '25/05/2025' in the 'Date du courrier' and 'Date d'arrivée' fields, 'ENR-0056' in the 'Numéro d'enregistrement' field, and '16:13' in the 'Heure' field. Below these is a section 'Ajouter une pièce jointe' with a file selection button 'Choisir un fichier' and a file name 'A TRAITER AVEC URGENCE.pdf'. At the bottom are two buttons: 'Enregistrer' and 'Retour'.

Figure 24: Formulaire de création d'un courrier

2.3. Page d'imputation d'un courrier

Cette page présente le formulaire d'imputation d'un courrier.



The form is titled 'Imputer un courrier' and is set against an orange background. It contains a text area for 'Description / Observations' with the text 'C'est pour une demande d'audience'. Below this is a dropdown menu for 'Destinataire' with the value 'palais judiciaire de yopougon'. At the bottom are three buttons: 'Ajouter Courrier', 'Imputer', and 'Retour'.

Figure 25: Formulaire d'imputation d'un courrier

2.4. Page d'envoi d'un courrier

Cette page présente le formulaire d'envoi de courrier.

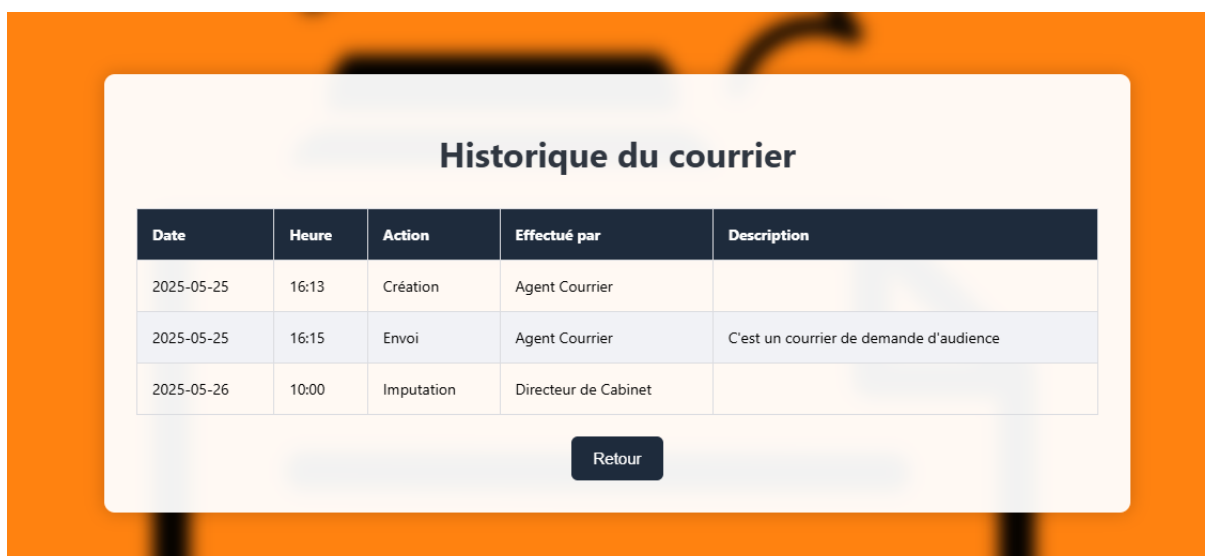


The screenshot shows a web form titled "Envoyer un courrier". It contains four input fields: "Numéro de référence" with the value "REF-2025-0012", "Objet" with the value "Demande d'audience", "Destinataire(s)" with the value "palais judiciaire de yopougon", and "Description" with the value "C'est un courrier de demande d'audience". At the bottom, there are four buttons: "Ajouter courrier", "Ajouter bordereau", "Envoyer", and "Retour".

Figure 26: Formulaire d'envoi d'un courrier

2.5. Page de consultation de l'historique

Cette page présente l'historique du courrier.



The screenshot shows a web page titled "Historique du courrier" containing a table with the following data:

Date	Heure	Action	Effectué par	Description
2025-05-25	16:13	Création	Agent Courrier	
2025-05-25	16:15	Envoi	Agent Courrier	C'est un courrier de demande d'audience
2025-05-26	10:00	Imputation	Directeur de Cabinet	

Below the table, there is a "Retour" button.

Figure 27: Historique du courrier

2.6. Page des courriers reçus

Cette page présente un dashboard des courriers reçus.



The screenshot shows a web interface titled 'Mes Courriers Reçus'. It features a table with the following data:

Référence	Objet	Expéditeur	Statut	Date	Action
REF-2025-0012	Demande d'audience	Directeur de cabinet	Renvoyé	2025-05-26	Voir

Below the table is a 'Retour' button.

Figure 28: Page des courriers reçus

3. Destinataire

3.1. Page d'accueil

Cette page présente le dashboard pour un destinataire plus précisément le directeur de cabinet.



Figure 29: Page d'accueil d'un destinataire (directeur de cabinet)

3.2. Page de la boîte de réception

Cette page présente la boîte de réception du destinataire le directeur de cabinet.

Boîte de réception						Retour
Référence	Objet	Expéditeur	Statut	Date	Action	
REF-2025-0012	Demande d'audience	Agent courrier	traité	2025-05-25	Voir	

Figure 30: Boîte de réception du destinataire le directeur de cabinet

3.3. Page du renvoi de courriers

Cette page présente les courriers renvoyés par le directeur de cabinet.

Renvoyer un courrier						Retour
Référence	Objet	Expéditeur	Date	Actions		
REF-2025-0012	Demande d'audience	palais judiciaire de yopougon	2025-05-26	Voir	Remplir Bordereau	Renvoyer

Figure 31: Page du renvoi de courriers



CONCLUSION

Ce mémoire de fin d'étude s'inscrit dans la mise en place d'une plateforme GED (Gestion Électronique de Documents) au sein du Cabinet du ministère de la Justice et des Droits de l'Homme s'est inscrit pleinement dans une dynamique de transformation numérique et d'amélioration des processus de travail. Face aux limites des procédures manuelles, cette solution a apporté une réponse concrète aux besoins de traçabilité, de rapidité, de sécurité et d'accessibilité dans la gestion quotidienne des courriers.

Grâce à une architecture logicielle structurée en trois couches, combinée à des technologies web sécurisées, la solution développée a permis une gestion centralisée et fluide des documents, de leur création à leur archivage, en passant par l'envoi, l'imputation et la consultation. Elle a ainsi contribué à rationaliser les échanges internes et à renforcer la réactivité des services dans le traitement des courriers.

Ce projet a été l'occasion d'appliquer de manière concrète les connaissances acquises en analyse des besoins, modélisation UML, conception logicielle, sécurité informatique et gestion de projet. Il a également mis en lumière l'importance de l'écoute des utilisateurs finaux, dont les retours ont permis d'adapter le système aux réalités du terrain.

En dépit des contraintes techniques et organisationnelles rencontrées, les objectifs ont été atteints, et la solution déployée a répondu aux attentes exprimées. Elle a constitué un socle évolutif sur lequel des fonctionnalités supplémentaires ont pu être envisagées à l'avenir, telles que l'automatisation des relances, l'intégration avec d'autres systèmes ministériels ou encore l'exploitation de données statistiques.

En définitive, ce projet a représenté une avancée significative vers une administration plus moderne, plus efficace et mieux outillée pour relever les défis de la gestion documentaire à l'ère du numérique



RÉFÉRENCES BIBLIOGRAPHIQUES

MÉMOIRES :

[1] ZAMBLE LOU CERISE : << **CONCEPTION ET RÉALISATION D'UNE APPLICATION POUR LA GESTION DES CONTENTIEUX LIÉS A LA SÉCURISATION FONCIÈRE RURALE** >> ; Mémoire de fin d'étude du cycle Technicien Supérieur en Informatique ESI ; INPHB ; 58 pages ; 2023-2024 ;

[2] KOFFI JUDE : << **CONCEPTION ET DEVELOPPEMENT D'UNE API DE PAIEMENT POUR L'AMELIORATION DES OPERATIONS FINANCIERES ET DE L'EXPERIENCE CLIENT** >> ; Mémoire de fin d'étude du cycle Technicien Supérieur en Informatique ESI ; INPHB ; 89 pages ; 2023-2024 ;

OUVRAGE :

[1] M. Adjé Louis ASSALÉ. UML2.2_diapos, 2022, 124 pages.



RÉFÉRENCES WEBOGRAPHIQUES

- (1) <https://justice.gouv.ci/> consulté le 08/06/2025 à 8h ;
- (2) <https://nodejs.org/docs/latest/api/webstreams.html> consulté le 10/03/2025 à 10h ;
- (3) <https://dashboard.render.com/> consulté le 17/03/2025 à 14h ;
- (4) <https://www.backblaze.com/sign-up-thank-you> consulté le 10/04/2025 à 9h ;
- (5) https://www.plantuml.com/plantuml/png/RP9BRi9038RtSmehgtR11Rge880giaHTLsxYWA1CJ6fxAlNsDT0W7aXPH3FV_YZpDLGhCGX6Dj0qD1teTGIp61BSHnOm4y_itGaX2K4R2qBL2jIuuI6B5G1AbmUXD4zkUKHaq6rIKRIetiMuu4LZOeSDU-RQx9vMoyNdjpygb4y-axlrs2DIj3varCYqbE1EeJB4Z1xzG1uWUN0lab0Y_T-DAeApZdz7tMGx55sBkcpn38St9xEz0Ttps52NRMgNR2bjen-BLCbwCh5LtDcTXiGnnlex3aLN_Gzj_kh6eyNdaTvRdjqlbkqJa8xLXCwOLDDHLOclHlKOFcX7KhdVCmTUnCqpkd1b9nwbdYwlTjGJ_CWiWB2LouImfnIT_Zd_W00 consulté le 12/04/2025 à 14h ;
- (6) <https://www.youtube.com/watch?v=fBNz5xF-Kx4> consulté le 20/04/2025 à 10h ;



TABLE DES MATIÈRES

DÉDICACES.....	I
REMERCIEMENTS.....	II
AVANT-PROPOS.....	III
SOMMAIRE.....	IV
LISTE DES SIGLES ET ABRÉVIATIONS.....	V
LISTE DES TABLEAUX.....	VII
LISTE DES FIGURES.....	VIII
RÉSUMÉ.....	IX
ABSTRACT.....	X
INTRODUCTION.....	1
PARTIE I : CADRE ET CONTEXTE DU PROJET.....	2
CHAPITRE I : PRÉSENTATION DE LA STRUCTURE D'ACCUEIL.....	3
I. PRÉSENTATION DE LA STRUCTURE D'ACCUEIL.....	3
II. DOMAINE D'ACTIVITÉS.....	3
III. ORGANIGRAMME HIÉRARCHIQUE.....	5
CHAPITRE II : DESCRIPTION DU PROJET.....	6
I. CONTEXTE DU PROJET.....	6
II. OBJECTIFS DU PROJET.....	7
1. Objectif général.....	7
2. Objectifs spécifiques.....	7
III. CAHIER DES CHARGES.....	7
1. Utilisateurs concernés.....	7
2. Fonctionnalités principales.....	8
2.1. Gestion des courriers.....	8
3. Contraintes.....	8
IV. ÉTUDE DE L'EXISTANT.....	9
1. Description du système actuel.....	9
2. Acteurs impliqués.....	9
3. Documents utilisés.....	9
4. Outils informatiques utilisés.....	9
5. Problèmes rencontrés.....	10
6. L'intérêt de l'application proposée.....	10
V. MÉTHODE DE GESTION DE PROJET.....	10
1. La méthode Agile.....	10
1.1. Définition.....	10
1.2. Principes de la méthode Agile.....	10
1.3. Présentation de Scrum.....	11



2.	Structures de Scrum	11
2.1.	Rôle dans Scrum.....	11
2.1.1.	Scrum Master	11
2.1.2.	Product Owner	11
2.1.3.	Equipe de développement	11
2.2.	Evènements Scrum	11
2.2.1.	Sprint	11
2.2.2.	Planification du Sprint	11
2.2.3.	Réunion quotidienne	11
2.2.4.	Revue de Sprint	12
2.2.5.	Rétrospective de Sprint.....	12
2.3.	Artéfacts Scrum.....	12
2.3.1.	Backlog Produit.....	12
2.3.2.	Backlog de Sprint	12
2.3.3.	Backlog Increment	12
2.4.	Mise en œuvre de Scrum	12
2.4.1.	Initiation d'un projet Scrum	12
2.4.2.	Planification et Estimation	12
2.4.3.	Exécution et contrôle des Sprints	12
2.4.4.	Suivi et Adaptation.....	12
2.4.5.	Avantage de Scrum	12
VI.	PLANIFICATION DU TRAVAIL	13
1.	Liste des tâches	13
2.	Planification des tâches	14
PARTIE II : ÉTUDE CONCEPTUELLE		15
CHAPITRE III : APPROCHE MÉTHODOLOGIQUE.....		16
I.	INTRODUCTION A LA MÉTHODE D'ANALYSE ET DE CONCEPTION.....	16
1.	Méthode d'analyse.....	16
2.	La méthode MERISE	16
2.1.	Présentation	16
2.2.	Les aspects du cycle de développement	16
2.2.1.	La démarche ou cycle de vie	16
2.2.2.	Le raisonnement ou cycle d'abstraction	17
2.2.3.	La maîtrise ou cycle de décision	18
3.	PU/UML	18
3.1.	Processus Unifié (PU)	18



3.2.	Unified Modeling Language (UML).....	18
3.3.	Comparaison des méthodes d'analyse.....	18
II.	CHOIX DE LA MÉTHODE D'ANALYSE	19
CHAPITRE IV : CRÉATION DE L'ARCHITECTURE DU SYSTÈME		20
I.	IDENTIFICATION DES ACTEURS ET DES CAS D'UTILISATION	20
1.	Les acteurs.....	20
2.	Les cas d'utilisation	20
3.	Diagramme des cas d'utilisation	21
4.	Description textuelle des cas d'utilisation	22
4.1.	Créer un courrier	22
4.2.	Envoyer un courrier	22
4.3.	Imputer un courrier	23
4.4.	Rechercher un courrier	23
4.5.	Consulter l'historique.....	23
4.6.	Lire un courrier	24
4.7.	Remplir le bordereau.....	24
4.8.	Renvoyer le courrier.....	25
4.9.	Gérer les utilisateurs et rôles	25
II.	DIAGRAMMES DE SÉQUENCE	26
1.	Créer un courrier	26
2.	Envoyer un courrier	26
3.	Imputer un courrier	27
4.	Rechercher un courrier	27
5.	Consulter l'historique.....	28
6.	Lire un courrier	28
7.	Remplir le bordereau.....	29
8.	Renvoyer le courrier	29
9.	Gérer les utilisateurs et rôles	30
III.	DIAGRAMME DE CLASSES	31
PARTIE III : MISE EN ŒUVRE		32
CHAPITRE V : ÉTUDE TECHNIQUE		33
I.	COMPARAISON ET CHOIX D'OUTILS	33
1.	L'environnement de développement.....	33
2.	Serveur web et hébergement	34
3.	Le front-end et le back-end	36
4.	Le système de gestion de la base de données (SGBD)	38



5. Outils de test d'API.....	39
6. Récapitulatif des choix	40
II. ARCHITECTURE LOGICIELLE	41
CHAPITRES VI : REALISATION.....	42
1. L'interface de connexion	42
2. Agent courrier	43
2.1. Page d'accueil	43
2.2. Page de création d'un courrier	43
2.3. Page d'imputation d'un courrier	44
2.4. Page d'envoi d'un courrier	45
2.5. Page de consultation de l'historique	45
2.6. Page des courriers reçus.....	46
3. Destinataire.....	46
3.1. Page d'accueil	46
3.2. Page de la boîte de réception	47
3.3. Page du renvoi de courriers	47
CONCLUSION.....	48
RÉFÉRENCES BIBLIOGRAPHIQUES	XIII
RÉFÉRENCES WEBGRAPHIQUES.....	XIV
TABLE DES MATIÈRES	XV