

Rapport de stage de fin d'année

présenté à

**Université Nord Américaine Privée :
Institut International de Technologie**

en vue de l'obtention du

**Diplôme National d'Ingénieur en :
Génie logiciel et Informatique Décisionnelle**

par

**Rawand Lazez , Khayreddine Derbali , Oumayma
Bouchaala**

**Plateform en ligne
de
Gestion des Réclamations**

Soutenu le 10/06/2024



REMERCIEMENT

Avant de présenter ce rapport du projet de fin d'étude, il apparaît opportunité de le commencer par des remerciements à ceux qui nous ont beaucoup appris lors de ce stage et qui ont contribué à sa réalisation.

Nous tenons à adresser nos remerciements les plus distingués aux membres de jury qui ont eu la faveur d'avoir accepté d'évaluer notre travail tout en espérant que la clarté du rapport soit à la hauteur de leurs attentes.

Nous remercions aussi Monsieur Slim Kallel, notre encadrant universitaire pour son encadrement, ses précieux conseils, support moral durant cette période, sa compétence incontestable, sa disponibilité sans faille et surtout son souci du travail sérieux et bien fait.

Enfin ,Nous tenons également à remercier Notre institut l'IIT pour cette opportunité qu'il nous a offert à travers ce stage, et à la confiance qu'il nous a attribué pour réaliser ce projet.



TABLE DES MATIÈRES

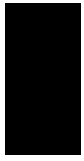
LISTE DES FIGURES	v
INTRODUCTION GÉNÉRALE	1
1 Chapitre 1 : Cadre générale	2
1.1 INTRODUCTION	3
1.2 Présentation du projet	3
1.2.1 Contexte	3
1.2.2 Problématique	3
1.2.3 Etude de l'existant	4
1.2.4 Analyse de l'existant	4
1.2.5 Solution proposée	6
1.3 Concept de base	7
1.3.1 Comparaison des méthodologies	7
1.3.2 Méthode Scrum	7
1.3.3 Les rituels de la méthode Scrum	7
1.3.4 Choix de la méthodologie	10
1.4 CONCLUSION	11
2 Chapitre 2 : Analyse et Spécification des besoins	12
2.1 INTRODUCTION	13
2.2 Spécification des besoins	13
2.2.1 Identification des acteurs	13
2.2.2 Besoins fonctionnels	13
2.2.3 Récit utilisateur	14
2.2.4 Besoins non fonctionnels	16
2.2.5 Besoins techniques	16
2.3 Environnement de travail	18

2.3.1	Architecture technique	18
2.3.2	Etude technique	20
2.4	CONCLUSION	23
3	Sprint 1 : Authentification	24
3.1	INTRODUCTION	25
3.2	Objectifs attendus	25
3.3	Sprint Backlog	25
3.4	Langage de Modelisation	26
3.4.1	Unified Modeling Language UML	26
3.4.2	Diagramme de cas d'utilisation	27
3.4.3	Diagramme de classe	28
3.4.4	Description textuelle des cas d'utilisation	29
3.5	Réalisation	31
3.6	CONCLUSION	33
4	Sprint 2 : Gestion des réclamations	34
4.1	INTRODUCTION	35
4.2	Objectifs attendus	35
4.3	Sprint Backlog	35
4.4	Langage de Modelisation	36
4.4.1	Diagramme de cas d'utilisation sprint 2	36
4.4.2	Diagramme de classe sprint 2	37
4.4.3	Description textuelle des cas d'utilisation	39
4.5	Réalisation	40
4.6	CONCLUSION	46
	CONCLUSION GÉNÉRALE	47
	BIBLIOGRAPHIE	47

LISTE DES FIGURES

1.1	Page d'accueil	5
1.2	les étapes de signalement d'un problème	6
1.3	Cycle de vie d'un projet Scrum	10
2.1	Architecture logicielle de l'application	17
2.2	Architecture technique de l'application	18
2.3	Logo Angular	20
2.4	Logo Type Script	21
2.5	Logo Bootstrap	21
2.6	Logo Spring Boot	21
2.7	Logo VisualParadigm	22
2.8	Logo MySql	22
2.9	Logo MySql Workbench	23
2.10	Logo Postman	23
3.1	Logo UML	26
3.2	Diagramme des cas sprint 1 Authentification	27
3.3	Diagramme des classe sprint 1 Authentification	28
3.4	Inscription	31
3.5	Connexion	32
3.6	Modifier Mot de passe	32
3.7	Voir les informations du profil	33
4.1	Interface sprint 2	37
4.2	Interface sprint 2	38
4.3	Interface création de réclamation	40
4.4	Interface Tableau de bord réclamation	40
4.5	Suivre l'état de réclamation	41
4.6	Interface détails de réclamation	41
4.7	Interface Ajouter Département	42

4.8	Interface liste des départements	42
4.9	Interface Ajouter Spécialité	43
4.10	Interface liste des spécialités	43
4.11	Interface visualiser Tableau de bord des réclamations	44
4.12	Interface détails de réclamation	44
4.13	Interface modifier réclamation	45
4.14	Interface supprimer réclamation	45



INTRODUCTION GÉNÉRALE

Notre plateforme de signalement en ligne, un espace dédié à la gestion efficace des réclamations et des problèmes. Notre service vise à fournir une solution accessible et transparente pour signaler tout type de préoccupation, qu'il s'agisse de problèmes environnementaux, de défauts de produits, de comportements nuisibles ou d'autres incidents nécessitant une action. Notre objectif est de faciliter le processus de signalement, de centraliser les données et de collaborer avec les parties concernées pour résoudre les problèmes de manière rapide et efficace.

Grâce à notre plateforme conviviale et sécurisée, les utilisateurs peuvent signaler des incidents, suivre le statut de leurs réclamations et contribuer à l'amélioration de la qualité de vie et de l'environnement.

Chapitre 1 : Cadre générale

Sommaire

1.1	INTRODUCTION	3
1.2	Présentation du projet	3
1.2.1	Contexte	3
1.2.2	Problématique	3
1.2.3	Etude de l'existant	4
1.2.4	Analyse de l'existant	4
1.2.5	Solution proposée	6
1.3	Concept de base	7
1.3.1	Comparaison des méthodologies	7
1.3.2	Méthode Scrum	7
1.3.3	Les rituels de la méthode Scrum	7
1.3.4	Choix de la méthodologie	10
1.4	CONCLUSION	11

1.1 INTRODUCTION

Le présent chapitre a comme objectif de situer le projet dans son cadre général. D'abord, nous présentons le contexte du projet ainsi que sa problématique et les objectifs à atteindre. Par la suite, nous détaillons la solution à développer. Enfin, nous introduisons la méthodologie adoptée durant la réalisation de notre projet.

1.2 Présentation du projet

1.2.1 Contexte

Ce projet est un travail en groupe de 3 dans le cadre du projet de fin d'année (PFA) ayant comme sujet "Plateforme en ligne de gestion de Réclamation." L'objectif est de développer une solution numérique innovante permettant de gérer de manière efficace et centralisée les réclamations des utilisateurs. Le système vise à automatiser le processus de réception, de traitement et de suivi des réclamations afin d'améliorer la satisfaction des utilisateurs et d'optimiser les opérations internes. La plateforme permettra aux utilisateurs de soumettre leurs réclamations via une interface conviviale et d'en suivre l'état en temps réel pour une prise de décision éclairée.

1.2.2 Problématique

Face à l'augmentation constante du volume de réclamations dans divers secteurs, les entreprises et institutions se trouvent confrontées à des défis majeurs pour gérer efficacement ces retours clients. L'absence de systèmes centralisés et automatisés pour le traitement des réclamations entraîne des délais de réponse prolongés, une gestion inefficace des ressources et une diminution de la satisfaction client. De plus, le suivi manuel des réclamations rend difficile l'identification rapide des problèmes récurrents et des tendances. Pour répondre à ces enjeux, il est crucial de développer une plateforme en ligne de gestion des réclamations, intégrant des outils de suivi

en temps réel, d'automatisation des processus, afin d'améliorer la réactivité et la qualité des réponses apportées aux utilisateurs.

1.2.3 Etude de l'existant

L'étude de l'existant permet de dégager les atouts et les points de faiblesse de chacune des solutions existantes et de situer le projet dans son contexte. Nous avons réalisé une analyse comparative des plateformes de gestion de réclamations actuellement disponibles sur le marché. Cette étude se concentre sur des systèmes tels que BeSignal, Zendesk, Freshdesk , en examinant leurs fonctionnalités principales, leur facilité d'utilisation, leurs capacités d'intégration et leur efficacité en termes de traitement des réclamations. Nous avons identifié les limitations de ces solutions, telles que des interfaces utilisateur complexes, des coûts élevés et une personnalisation limitée. Cette analyse nous a permis de mieux comprendre les besoins spécifiques des utilisateurs et de définir les fonctionnalités clés à intégrer dans notre propre plateforme pour offrir une solution plus adaptée et plus performante.

1.2.4 Analyse de l'existant

Interface 1 : Page d'accueil : La page d'accueil présentée dans la Figure 1 renferme les principales fonctionnalités que les plateformes de gestion de réclamations offrent à leurs utilisateurs. Cette interface offre les modules suivants :

- **Bienvenue et Introduction :** Le message d'accueil informe les étudiants qu'ils peuvent signaler des problèmes concernant la sécurité, le bien-être et l'intégrité de leur communauté académique. Il souligne l'importance de la confidentialité et de la sécurité des informations rapportées.
- **Bouton Signaler un problème :** Permet aux utilisateurs de commencer le processus de signalement en cliquant simplement sur le bouton dédié.
- **À propos :** Une section qui détaille la mission de la plateforme et son engagement à traiter chaque signalement de manière sérieuse, professionnelle, respectueuse et

confidentielle. L'utilisateur peut naviguer facilement vers les autres sections grâce aux liens de navigation situés en haut de la page.

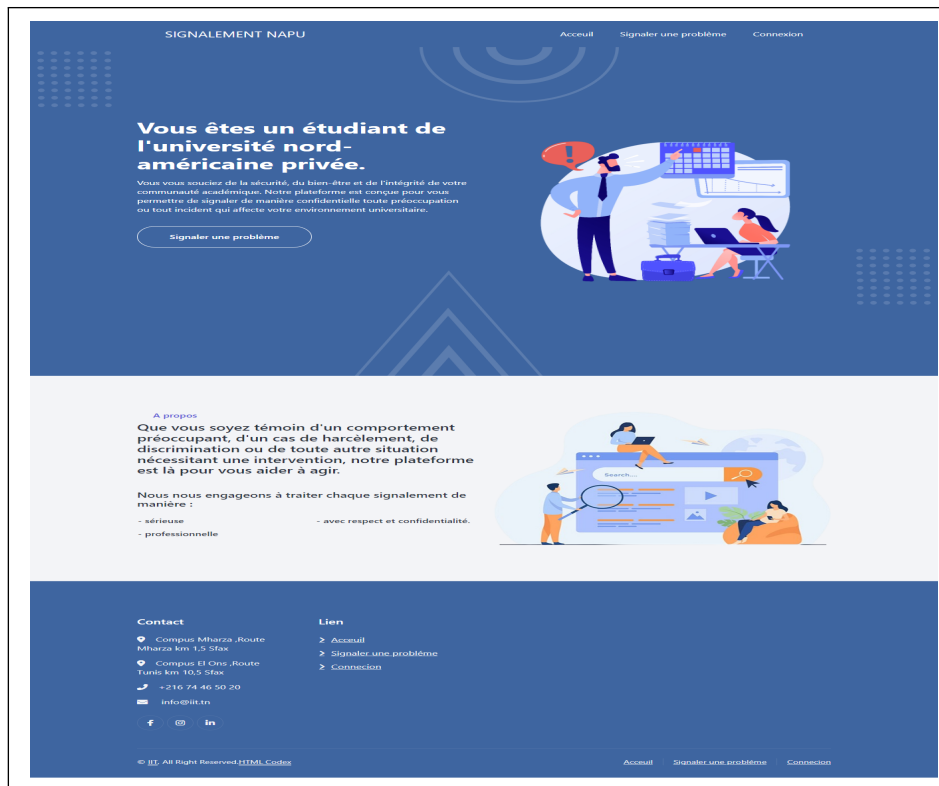


FIGURE 1.1 – Page d'accueil

Interface 2 : Comment faire un signalement ? : La page "Comment faire un signalement ?", présentée dans la figure ci dessous, guide l'utilisateur à travers le processus de signalement en trois étapes simples :

- **Accéder à votre compte** : Cette première étape invite l'utilisateur à se connecter à son compte pour pouvoir effectuer un signalement. Si l'utilisateur n'a pas encore de compte, il sera dirigé vers une interface d'inscription.
- **Remplir le formulaire de signalement ou de suggestion** : Une fois connecté, l'utilisateur est dirigé vers un formulaire où il peut décrire le problème ou faire une suggestion. Cette étape est cruciale pour recueillir toutes les informations nécessaires à l'évaluation du signalement.

- Faire le suivi de votre signalement ou de suggestion : Après avoir soumis le formulaire, l'utilisateur peut suivre l'état de son signalement ou de sa suggestion pour être informé des actions prises et des résolutions proposées. Ces étapes sont clairement illustrées avec des icônes et des descriptions pour guider l'utilisateur de manière intuitive.



FIGURE 1.2 – les étapes de signalement d'un problème

1.2.5 Solution proposée

Face à tous ces problèmes, notre objectif dans ce travail est de proposer une solution efficace. Cette solution est une plateforme de réclamation en ligne de type client-serveur, composée de différents modules :

- Gestion des réclamations : Consulter, ajouter, rechercher, modifier et supprimer une réclamation. Les utilisateurs peuvent soumettre leurs réclamations via un formulaire convivial, suivre l'état de leur demande en temps réel, et voir les détails et les mises à jour importantes.
- Visualiser le tableau de bord : Consulter les réclamations , suivre l'état de demande en temps réel, et voir les détails et les mises à jour importantes.

1.3 Concept de base

Afin de garantir la réussite de notre projet, il faut dès le début fixer un plan de travail détaillé, définir une stratégie bien testée et une méthodologie claire pour pouvoir bien guider et cadrer le projet avant le lancement de ce dernier.

1.3.1 Comparaison des méthodologies

Pour bien illustrer les caractéristiques des différentes méthodologies les bien connues et utilisées pour l'élaboration d'un projet informatique, nous avons choisi de dresser un tableau comparatif qui comporte une petite description de deux méthodologies classiques et Scrum ainsi que les avantages et les inconvénients de chacune comme le montre le tableau (1).

1.3.2 Méthode Scrum

Le Scrum [1] est une méthode agile [2] repose un certain nombre d'itérations et d'incréments. Les itérations consistent à diviser le projet afin de le développer par étapes successives. Une incrémentation consiste à livrer de manière fréquente le client. En plus d'être holistique, la méthode Scrum est agile, c'est-à-dire qu'elle intègre le client dans la réalisation du projet.

1.3.3 Les rituels de la méthode Scrum

La méthode Scrum suppose une équipe composée de trois principaux acteurs :

- Le Product Owner : Il représente le propriétaire de produit, c'est lui qui porte la vision du projet, il représente les utilisateurs, les commanditaires du projet. Il est donc chargé de maximiser la valeur du produit et travail de l'équipe de développement.
- Le Scrum master : C'est le chef de l'équipe Scrum, il assure que la méthodologie Scrum est correctement appliquée, il facilite les interactions entre les membres et s'assurer que l'équipe est opérationnelle et productive.
- L'équipe de développement : L'équipe est constituée de développeurs, Elle est chargée de transformer les besoins définis par le Product Owner en fonctionnalités utilisables.

Méthodologie	Description	Avantages	Inconvénients
Méthode classique	<ul style="list-style-type: none"> • Processus de développement basé sur l'agilité • Modèle itératif 	<ul style="list-style-type: none"> • Avoir une idée claire sur différents étapes du projet. • Avoir une flexibilité totale du projet : le client peut avoir une flexibilité totale au niveau de la définition des priorités selon leur besoins. • Avoir une meilleure qualité : une évolution périodique 	<ul style="list-style-type: none"> • Faible documentation • Maintenance relativement difficile
Scrum	<ul style="list-style-type: none"> • Un modèle basé sur l'agilité et rythmé par des itérations de quelques semaines appelés les sprints. 	<ul style="list-style-type: none"> • Livraison fréquente d'une application fonctionnelle.. • Simplicité des processus. • Augmentation de productivité. 	<ul style="list-style-type: none"> • Peu de documentation écrite • Exige une certaine responsabilité des développeurs

Elle est pluridisciplinaire et possède toutes les compétences nécessaires pour réaliser le projet.

La méthode Scrum est tâche du Product Backlog est appelée « User Story »[4]. Un ensemble d'User Stories seront sélectionnés et participeront à une itération de développement, c'est le « Sprint » qui est d'une durée d'une à quatre semaines.la méthodologie agile la plus utilisé. Elle regroupe l'ensemble de ses tâches disponibles à développer en un « Product Backlog » (le Backlog du produit) qui est sujet à évoluer avec le projet et peut être mis à jour pendant la mise en œuvre du projet en fonction des besoins du client .

Scrum	<ul style="list-style-type: none"> • Pour chaque sprint une réunion de planification est mise en œuvre pour fixer les exigences les plus prioritaires pour le client et à la fin de chaque sprint un composant du logiciel souhaité sera développé, terminé, testé et livré au client. 	<ul style="list-style-type: none"> • Le client apprécie la progression de l'élaboration du logiciel. 	<ul style="list-style-type: none"> • Violation de responsabilité
-------	---	---	---

TABLE 1.1 – Comparaison des méthodologies agile

Notre projet a été développé avec des sprints de 5 semaines. Les notions suivantes vont de pair avec les rituels du Scrum, à savoir :

- Sprint planning : La planification du sprint consiste à choisir les User Story à réaliser durant le sprint courant.
- DSM : Daily Scrum Meeting (Mêlée quotidienne) : C'est une réunion qui regroupe le Scrum Master et l'équipe de développement. Elle est réalisée tous les jours durant le sprint. Chaque personne présente le travail réalisé depuis la dernière réunion et les parties qu'il faut préparer jusqu'à la prochaine réunion.
- SpringReview (Revue de Sprint) : C'est une réunion qui est réalisée à la fin du sprint et regroupe tous les acteurs du modèle Scrum. L'objectif est de faire une présentation des fonctionnalités réalisées.
- Spring Rétro-perspective : Réalisée à la fin de chaque Sprint, elle regroupe le Scrum master, l'équipe de développement et le Product Owner pour relever ce qui a bien marché et les difficultés rencontrées lors du sprint précédent et dégager les bonnes pratiques pour le sprint suivant.
- Product BacklogGrooming : Il s'agit du nettoyage du Product Backlog afin de supprimer les User Story usés ou ajouter de nouveaux User Story. Cette réunion regroupe le ProductOwner, Scrum Master ainsi que l'équipe de développement.

Le responsable dans l'IIT est le product Owner et Mr Slim kallel est le scrum master et nous sommes l'équipe de développement.

La figure 3 nous montre le cycle de vie d'un projet Scrum[3]. Qui résume en fait tout ce qui vient d'être détaillé.

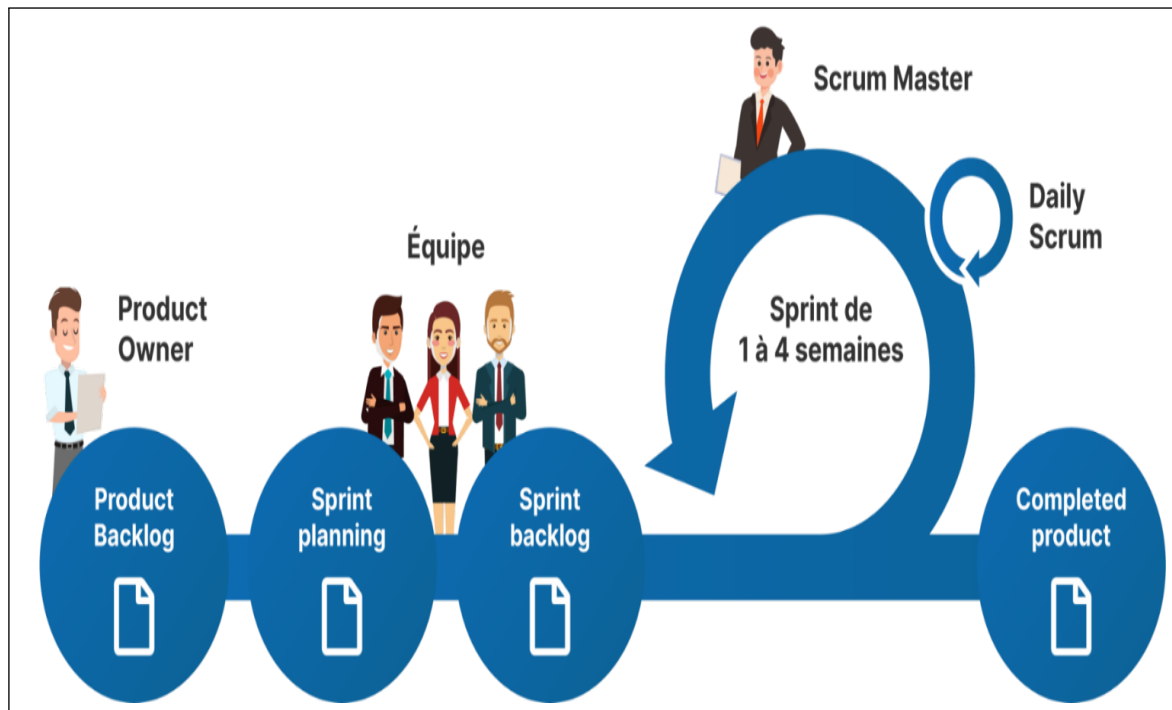


FIGURE 1.3 – Cycle de vie d'un projet Scrum

1.3.4 Choix de la méthodologie

Suite à l'étude de divers critères des différentes méthodologies, Nous avons choisi d'adopter la méthodologie Scrum pour l'élaboration des différentes phases de notre projet. Notre choix est justifié par plusieurs raisons :

- La simplicité et la souplesse de ce modèle,
- La flexibilité aux changements dans chaque sprint,
- La livraison périodique de produits bien développés et testés pour de courtes itérations,
- L'organisation du travail en définissant des règles clairement définies dans chaque réunion au début de chaque sprint,
- Le client est considéré comme un élément clé tout au long du déroulement des phases d'élaboration du projet.

1.4 CONCLUSION

Dans ce chapitre nous avons eu l'occasion de présenter le projet dans son cadre. Ce chapitre nous a permis de problématiser le sujet afin de montrer son importance. Avant de présenter la solution proposée, nous sommes partis d'une étude critique des quelques systèmes existants sur le marché afin de tenir compte de leurs insuffisances pour customiser notre réalisation. Enfin, ce chapitre nous a prodigué les connaissances nécessaires pour nous préparer à ce qui suivra dans le reste de ce projet. Le chapitre suivant attaque la partie spécification des besoins.

Chapitre 2 : Analyse et Spécification des besoins

Sommaire

2.1	INTRODUCTION	13
2.2	Spécification des besoins	13
2.2.1	Identification des acteurs	13
2.2.2	Besoins fonctionnels	13
2.2.3	Récit utilisateur	14
2.2.4	Besoins non fonctionnels	16
2.2.5	Besoins techniques	16
2.3	Environnement de travail	18
2.3.1	Architecture technique	18
2.3.2	Etude technique	20
2.4	CONCLUSION	23

2.1 INTRODUCTION

Dans le présent chapitre, nous commençons par présenter la première phase du cycle de développement d'un logiciel qui consiste à l'analyse et l'extraction des besoins fonctionnels, non fonctionnels et techniques de notre application.

2.2 Spécification des besoins

Dans cette partie, nous allons préciser les besoins fonctionnels et non fonctionnels de notre application pour s'assurer le bon fonctionnement du système afin de satisfaire les exigences et les attentes de nos clients.

2.2.1 Identification des acteurs

Dans cette étape, nous dégagons les principaux acteurs intervenant dans l'exploitation de notre système. En effet, un acteur est toute entité qui joue un rôle, actif ou passif, qui interagit directement avec le système étudié. Les acteurs principaux du plateforme de gestion de réclamation sont :

- Administrateur : C'est l'acteur principale de l'application (un membre de l'équipe Iit), l'utilisateur c'est l'étudiant, il a tous les droits d'accès aux données des utilisateurs et gérer les réclamations, afficher la liste des specialités et des départements.

2.2.2 Besoins fonctionnels

Les besoins fonctionnels sont les besoins auxquels devrait répondre notre système en dehors de toute contrainte. Les besoins fonctionnels de notre système se résument, principalement, dans :

- Gérer les réclamations et voir leur suivie et les détails
- Visualiser le tableau de bord des réclamations

- Ajouter département
- Ajouter spécialité
- Voir les détails du profil
- Modifier le mot de passe.

2.2.3 Récit utilisateur

Dans ce sous-paragraphe, nous allons détailler les besoins fonctionnels de nos acteurs sous forme des récits utilisateurs. Un récit utilisateur (User story) est une description fonctionnelle d'un besoin ou d'une attente de l'utilisateur . C'est une méthode pour préciser les besoins fonctionnels des clients. L'objectif d'un récit utilisateur est la détermination des fonctionnalités à développer par type d'utilisateur, il permet d'approcher le maximum à la vision du client. La rédaction d'une user story doit comporter 3 dimensions :

- Qui : l'utilisateur finale ou intermédiaire du produit.
- Quoi : la fonctionnalité demandée par l'utilisateur.
- Pourquoi : la valeur ajoutée par la fonctionnalité.

Utilisateur	Description des Tâches
Administrateur	<ul style="list-style-type: none">• En tant qu'Administrateur, je veux avoir un login et un mot de passe afin de me connecter à l'application.• En tant qu'Administrateur, je veux gérer les réclamations(ajouter,modifier, supprimer).• En tant qu'Administrateur, je veux assurer le suivi des actions nécessaires pour chaque réclamation.• En tant qu'Administrateur, je veux voir les détails de réclamations.• En tant qu'Administrateur, je veux ajouter une spécialité.• En tant qu'Administrateur, je veux ajouter un département.• En tant qu'Administrateur, je veux ajouter une spécialité.• En tant qu'Administrateur, je veux consulter le tableau de bord pour voir le nombre et l'état de mes réclamations.• En tant qu'Administrateur, je voir la liste des spécialités.• En tant qu'Administrateur, je voir la liste des départements.• En tant qu'Administrateur, je voir la liste des réclamations.
Étudiant	<ul style="list-style-type: none">• En tant qu'Étudiant, je veux avoir un login et un mot de passe afin de me connecter à l'application.• En tant qu'Étudiant, je veux signaler un problème via la plateforme.• En tant qu'Étudiant, je veux modifier mon mot de passe lors de la création du compte pour sécuriser mon accès.• En tant qu'Étudiant, je veux consulter le tableau de bord personnel pour voir l'état de mes réclamations.• En tant qu'Étudiant, je veux voir les informations de mon profil.• En tant qu'Étudiant, je veux suivre mes réclamations.• En tant qu'Étudiant, je veux suivre les détails de mes réclamations.

2.2.4 Besoins non fonctionnels

Un besoin non fonctionnel est un besoin spécifiant les propriétés du système. Les besoins non fonctionnels sont indispensables et permettent l'amélioration de la qualité logicielle de notre système. Ils agissent comme des contraintes sur les solutions, mais leur prise en considération nous fera éviter plusieurs incohérences. Dans notre système les besoins non fonctionnels sont entre autres :

- **Sécurité** : C'est l'une des fonctionnalités les plus importantes de l'application. Notre application doit être fortement sécurisée, Ce besoin sera principalement garanti par :
 - L'obligation de l'authentification pour accéder à notre plateforme.
 - La manipulation des mots de passe est faite avec le principe de cryptage /décryptage.
 - La génération automatique d'un token de communication suite à l'étape de l'authentification de ce Token sera envoyé au serveur avec chaque appel à une ressource pour qu'il puisse identifier le demandeur.
- **Ergonomie et bonne Interface** : L'application doit être adaptée à l'utilisateur avec une utilisation simple et facile dans la navigation entre les pages, la mise en textes et les couleurs utilisés.
- **Maintenabilité** : Notre application doit être facile à maintenir. En cas de bugs l'origine doit être détectée et le bug doit être fixé très rapidement.
- **Fiabilité** : L'application doit fonctionner de manière cohérente, être sans erreur et doit être satisfaisante.

2.2.5 Besoins techniques

Les besoins techniques sont ceux liés à l'environnement de travail. En effet, Pour décrire d'une manière symbolique et schématique les différents éléments du système, on a recours à une architecture 3 tiers qui garantit la stabilité et l'efficacité de notre plateforme.

L'architecture de notre plateforme est partagée entre :

- Client léger (Navigateur Web) : Couche de présentation

Un client léger sous forme de navigateur web qui va permettre de formuler des requêtes aux serveurs web.

- Serveur Web (Spring Boot) : Couche de logique métier

Le serveur web est basé sur Spring Boot, un framework Java pour le développement d'applications web.

Son rôle principal est de gérer la communication entre le client (navigateur) et le serveur de base de données.

- Serveur de Base de Données : Couche de stockage

Le serveur de base de données stocke les données nécessaires à l'application.

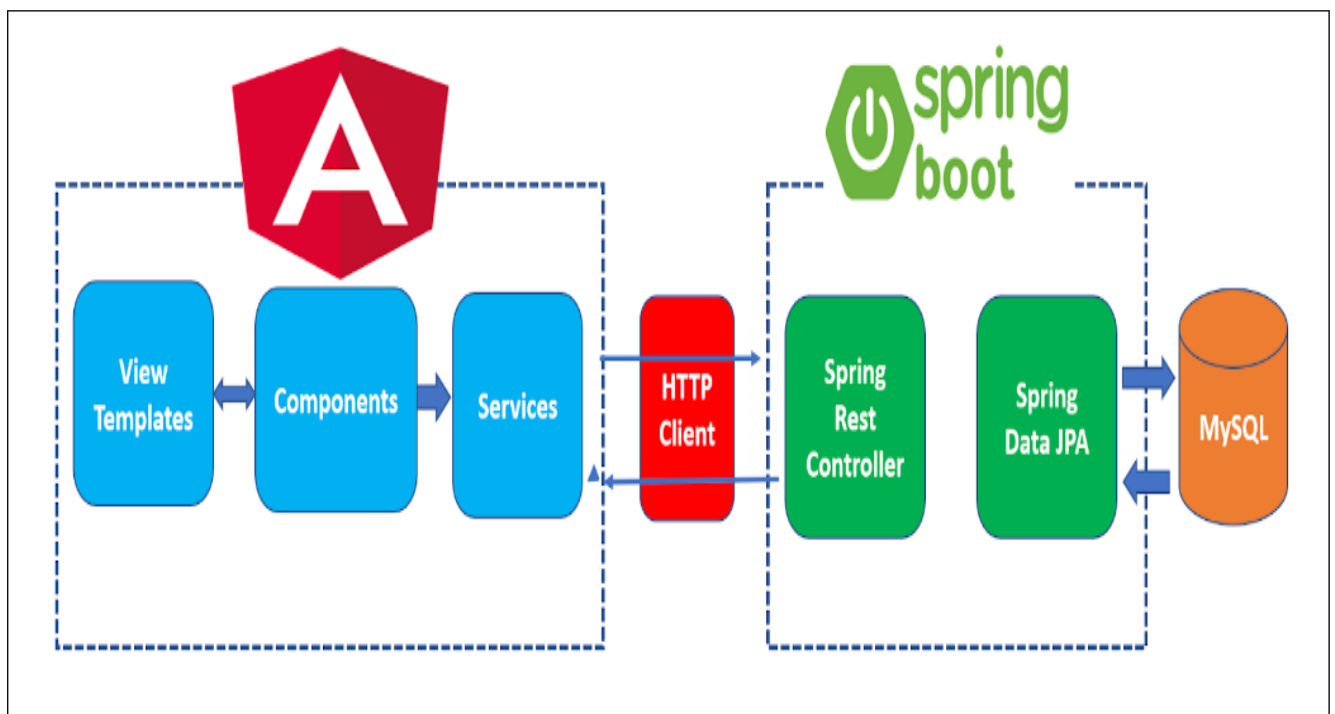


FIGURE 2.1 – Architecture logicielle de l'application

2.3 Environnement de travail

Dans cette section nous allons présenter l'architecture technique adoptée pour notre application ainsi que les outils et les technologies utilisées dans notre travail.

2.3.1 Architecture technique

Dans cette section nous allons présenter l'architecture technique adoptée pour notre application ainsi que les outils et les technologies utilisées dans notre travail.

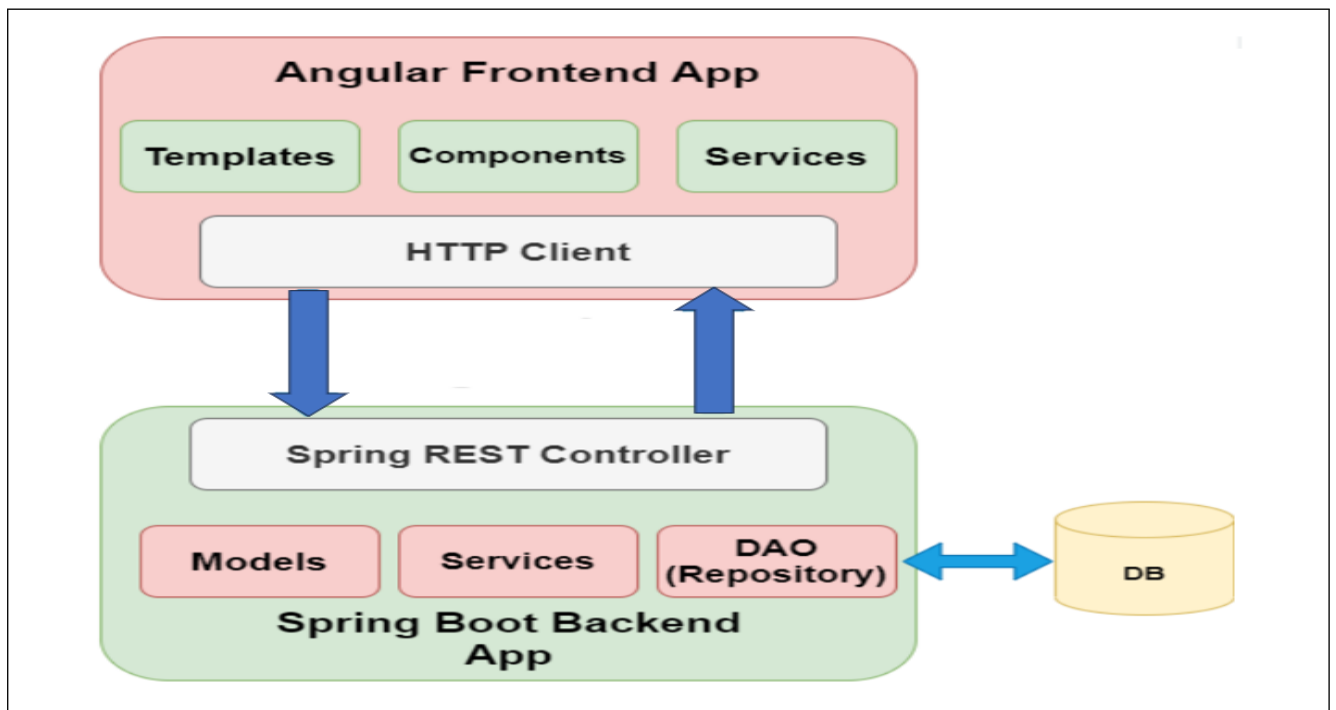


FIGURE 2.2 – Architecture technique de l'application

- Frontend (Angular) : L'interface utilisateur (UI) de notre application est développée avec Angular. Le frontend présente les interfaces visuelles avec lesquelles les utilisateurs interagissent. Il comprend les éléments suivants :
 - Templates : Les modèles de vue qui définissent la structure et le layout de l'interface utilisateur.
 - Components : Les composants Angular qui encapsulent la logique et le style d'une partie

spécifique de l'interface utilisateur(les formulaires, listes, etc).

- Services : Les services Angular qui gèrent la logique métier et les communications HTTP(les appels API, la gestion des états).
- HTTP Client : Utilisé pour effectuer des requêtes HTTP vers le backend Spring Boot.
- Backend (Spring Boot) : Le backend gère la logique métier et les fonctionnalités de l'application.

Il est invisible pour les utilisateurs et garantit le dynamisme de l'application.

Les éléments clés du backend sont :

- Models : Les objets métiers qui représentent les données de l'application (les réclamations, les utilisateurs).
- Services : La couche de services qui implemente la logique métier. (la validation, traitement des données).
- DAO (Repository) : La couche d'accès aux données qui interagit avec la base de données (DB).
- Spring REST Controller : Les contrôleurs REST qui reçoivent les requêtes HTTP du frontend, appellent les services appropriés et renvoient les réponses.
- Base de Données (MySQL) :

La base de données stocke les données de l'application. Elle permet aux utilisateurs de gérer les données :

- Consultation des informations.
- Saisie de nouvelles données.
- Modification des données existantes.
- Suppression de données.

Elle assure également la sécurité et les droits d'accès des utilisateurs.

L'interaction entre l'Angular Frontend App et le Spring Boot Backend App se fait via des appels HTTP, où le client HTTP d'Angular envoie des requêtes aux contrôleurs REST de Spring Boot, qui ensuite traitent les requêtes, accèdent à la base de données si nécessaire, et renvoient les réponses au frontend.

Cette architecture permet une séparation claire des responsabilités, facilitant ainsi la maintenance et l'extension de l'application. Les composants front-end gèrent l'affichage et l'interaction utilisateur, tandis que le backend se charge de la logique métier et de la gestion des données.

En résumé, notre architecture technique offre une séparation claire entre le frontend (interface utilisateur) et le backend (logique métier et stockage des données). Cela garantit une application performante, sécurisée et évolutive.

2.3.2 Etude technique

L'étude technique est la phase d'adaptation de la conception à l'architecture technique retenue, tout en décrivant et documentant le fonctionnement de chaque unité du logiciel. Dans cette section, nous décrivons brièvement les outils logiciels et les technologies mises en œuvre pour la réalisation de notre application :

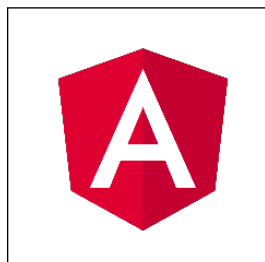


FIGURE 2.3 – Logo Angular

Angular 17 : C'est un framework Front-End orienté composants qui permet de créer les « single page application ». En effet la page web est divisée sous forme de plusieurs composants, à un changement sur un composant Angular ne recharge que ce composant et non toute la page. Il est basé sur la séparation des données et de la vue en utilisant les « binding » pour la communication des données vers le HTML .

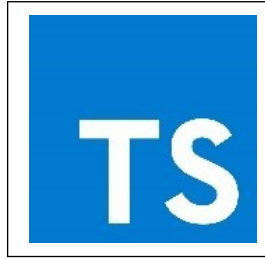


FIGURE 2.4 – Logo Type Script

Type Script : C'est un langage développé par Microsoft dans le but de simplifier le JavaScript et rendre les applications plus interactives. Il est utilisé par la framework Angular .



FIGURE 2.5 – Logo Bootstrap

Bootstrap : c'est un framework développé par l'équipe du réseau social Twitter. Proposé en open source, 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 .



FIGURE 2.6 – Logo Spring Boot

Spring Boot : c'est un **framework open-source** basé sur **Spring**, qui facilite la création d'applications autonomes, prêtes pour la production et pouvant être exécutées directement. Il est souvent utilisé pour développer des **microservices**, des **applications web** et des **API REST**. Spring Boot simplifie la configuration et le déploiement des applications en fournissant des

configurations par défaut et en réduisant le besoin de configuration manuelle.



FIGURE 2.7 – Logo VisualParadigm

Visual Paradigm : c'est un outil de modélisation de logiciels open source qui prend en charge le cadre UML (Unified Modeling Language), il permet la création de diagrammes basé sur le Web .



FIGURE 2.8 – Logo MySql

MySQL : c'est un système de gestion de bases de données relationnelles open source, populaire pour les applications web. Il est reconnu pour sa performance, sa fiabilité et sa facilité d'utilisation. MySQL supporte des requêtes SQL complexes et la gestion des transactions pour divers projets, des petits sites aux grandes entreprises.



FIGURE 2.9 – Logo MySql Workbench

MySQL Workbench : c'est un outil visuel de conception et de gestion de bases de données pour MySQL. Il permet de modéliser des données, de concevoir des schémas et d'administrer des bases de données via une interface graphique intuitive. MySQL Workbench facilite la création et l'édition de modèles de bases de données, l'exécution de requêtes SQL, et la gestion des serveurs et des utilisateurs.



FIGURE 2.10 – Logo Postman

Postman : c'est un outil de collaboration pour le développement d'API qui permet de tester, documenter et partager des APIs facilement. Il offre une interface intuitive pour créer et exécuter des requêtes HTTP, et analyser les réponses. Postman est essentiel pour les développeurs pour automatiser les tests et assurer la qualité des APIs.

2.4 CONCLUSION

Cette plateforme vise à améliorer la gestion des réclamations au sein de notre université, en offrant une interface conviviale et sécurisée pour les étudiants et les administrateurs. Grâce à ces fonctionnalités, nous nous engageons à traiter chaque réclamation de manière sérieuse, professionnelle et confidentielle.

Sprint 1 : Authentication

Sommaire

3.1	INTRODUCTION	25
3.2	Objectifs attendus	25
3.3	Sprint Backlog	25
3.4	Langage de Modelisation	26
3.4.1	Unified Modeling Language UML	26
3.4.2	Diagramme de cas d'utilisation	27
3.4.3	Diagramme de classe	28
3.4.4	Description textuelle des cas d'utilisation	29
3.5	Réalisation	31
3.6	CONCLUSION	33

3.1 INTRODUCTION

Dans le cadre du développement de notre plateforme en ligne de gestion de réclamations, une fonctionnalité essentielle est la gestion des réclamations. La gestion des réclamations permet aux utilisateurs de soumettre, suivre et résoudre leurs plaintes de manière efficace et structurée. Cette fonctionnalité est cruciale pour assurer une réponse rapide et appropriée aux problèmes signalés par les utilisateurs, tout en permettant aux administrateurs de suivre et de gérer les réclamations de manière systématique.

3.2 Objectifs attendus

Le but du premier sprint est de réaliser le module d'authentification. Ce module est crucial pour garantir la sécurité et la confidentialité des utilisateurs de la plateforme de gestion de réclamation. Une fois ce module implémenté, il permettra de vérifier l'identité des utilisateurs avant de leur accorder l'accès aux fonctionnalités de la plateforme, assurant ainsi que seules les personnes autorisées puissent accéder aux données sensibles et aux services offerts.

3.3 Sprint Backlog

Le tableau ci-dessous représente le Sprint Backlog du premier sprint, détaillant les tâches nécessaires pour accomplir le module d'authentification. Chaque tâche est essentielle pour atteindre l'objectif principal de ce sprint et pour assurer une mise en œuvre réussie et sécurisée du système d'authentification.

Backlog	Tâches	Temps alloués en jours	Priorité
- Authentification	S'inscrire	3	1
	Création compte utilisateur pour l'administrateur	5	1

TABLE 3.1 – Backlog sprint 1

- Authentification	modifier mot de passe	5	1
	voir les informations du compte	5	1

TABLE 3.2 – Backlog sprint 1

3.4 Langage de Modelisation

Dans la partie conception, nous avons opté pour le langage UML comme formalisme de modélisation, nous allons définir et justifier notre choix dans ce paragraphe. La modélisation est une projection de l'image du monde réel sous forme de graphe et de schéma permettant de faciliter la compréhension des activités d'un système. Pour la représentation des diagrammes de notre projet, nous avons choisi d'utiliser le langage de modélisation UML qui permet une représentation claire des activités, des données et des traitements.

3.4.1 Unified Modeling Language UML

UML [5] est un langage orienté objet qui permet la modélisation standardisée de l'architecture du système. Ce langage définit des fondements normalisés basés sur la notation graphique qui couvre le cycle de développement d'un logiciel de la spécification des besoins jusqu'à l'implémentation. Il offre également de différents diagrammes permettant de représenter le système en son ensemble et de couvrir les différents niveaux d'abstraction ce qui aide à la bonne compréhension du fonctionnement du système. La figure 21 représente le logo de UML.

**FIGURE 3.1 – Logo UML**

3.4.2 Diagramme de cas d'utilisation

Les diagrammes de cas d'utilisation [6] permettent une meilleure représentation des interactions entre les acteurs du système et le système lui-même. Il regroupe l'ensemble des fonctionnalités que doit fournir le système. Il est donc basé sur les acteurs et les cas d'utilisation. Pour cela nous allons avant tout, identifier les acteurs de notre système et les cas d'utilisation sur les quels ils interviennent

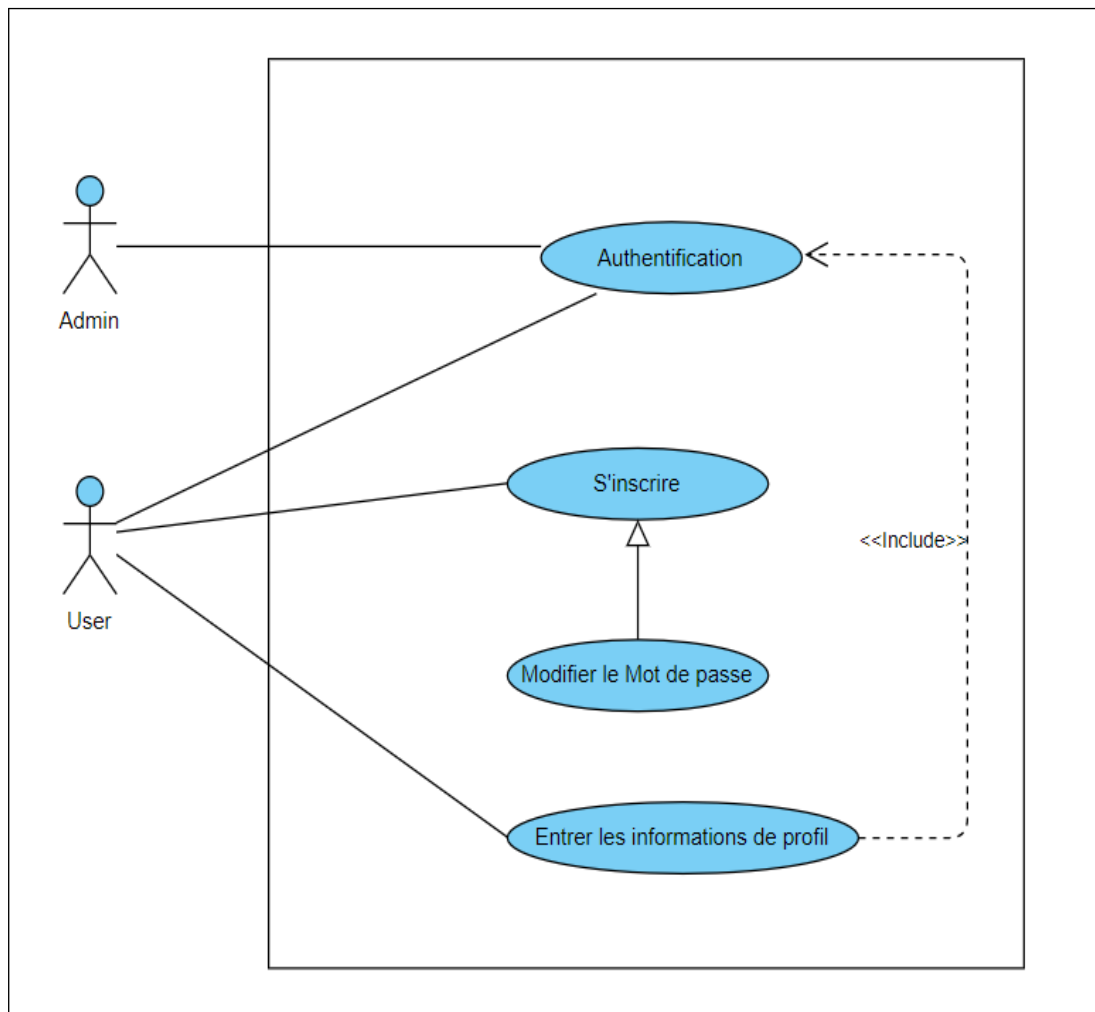


FIGURE 3.2 – Diagramme des cas sprint 1 Authentification

3.4.3 Diagramme de classe

Le diagramme de classes [7] est un digramme d'un aspect statique dans le standard UML permettant de modéliser les entités du système ainsi que les relations établies entre elles.

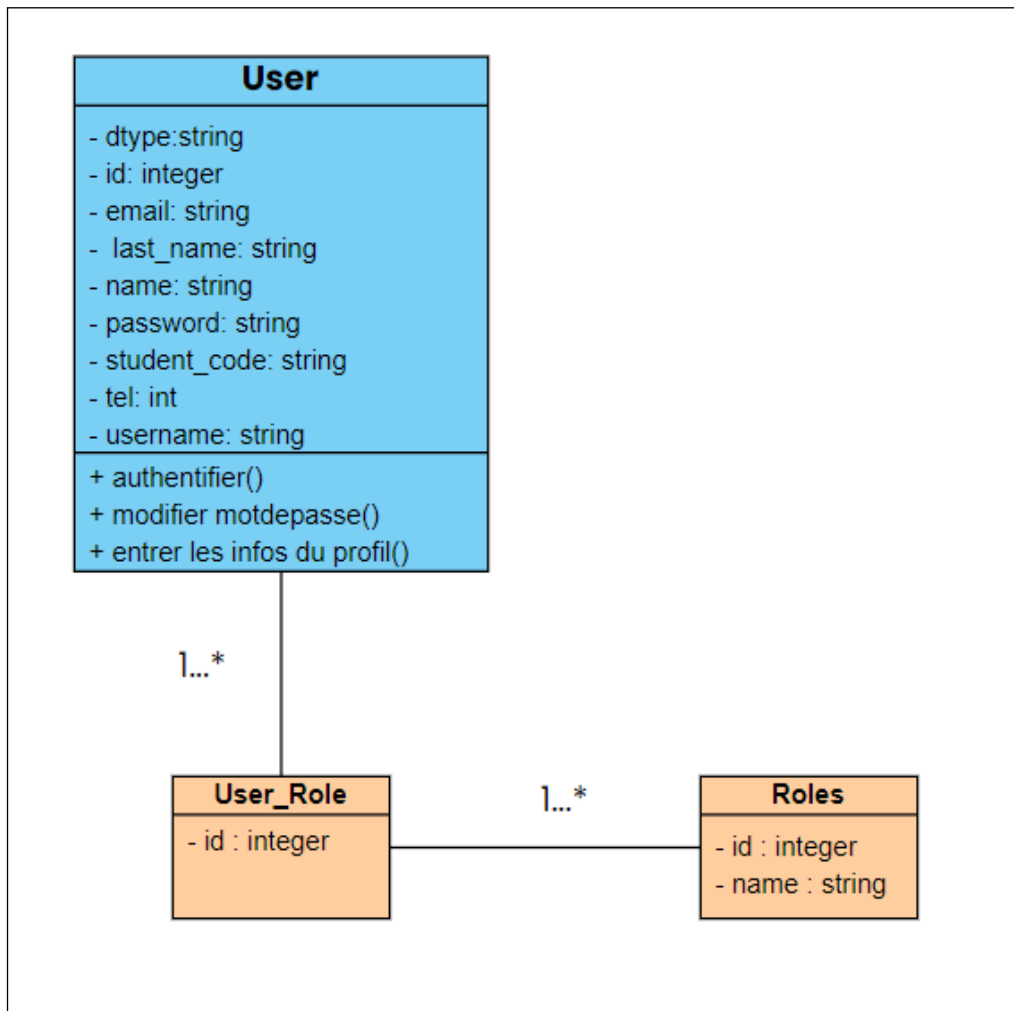


FIGURE 3.3 – Diagramme des classe sprint 1 Authentification

3.4.4 Description textuelle des cas d'utilisation

S'authentifier
Description des scénarios : Pré condition : L'administrateur possède le droit d'accès Post condition : La session de l'administrateur s'ouvre.
Scénario nominal : S'authentifier Le scénario se poursuit selon les étapes suivantes : <ol style="list-style-type: none">L'administrateur doit accéder à l'interface et l'authentification, saisir son email et son mot de passe.Le système vérifie la validité de l'email et du mot de passe.Si l'une des informations saisies est invalide, L'exception 1 est déclenché.S'il n'y a pas d'exception le système re-directe l'administrateur vers la page d'accueil.
Exception 1 : Le système affiche le message : « Email ou Mot de passe incorrecte ».
Identification : Titre : Création comptes client Acteurs : Administrateur Objectif : Il permet de créer des comptes client.
Description des scénarios : Pré condition : Administrateur est authentifié. Post condition : Compte client créé.

TABLE 3.3 – Description textuelle du cas d'utilisation « Authentification Admin »

S'authentifier
<p>Description des scénarios :</p> <p>Pré condition : L'utilisateur possède le droit d'accès</p> <p>Post condition : La session de l'utilisateur s'ouvre.</p>
<p>Scénario nominal : S'authentifier</p> <p>Le scénario se poursuit selon les étapes suivantes :</p> <ol style="list-style-type: none"> L'utilisateur doit accéder à l'interface et l'authentification, saisir son email et son mot de passe. Le système vérifie la validité de l'email et du mot de passe. Si l'une des informations saisies est invalide, L'exception 1 est déclenché. S'il n'y a pas d'exception le système re-directe l'utilisateur vers la page d'accueil.
<p>Exception 1 :</p> <p>Le système affiche le message : « Email ou Mot de passe incorrecte ».</p>
<p>Identification :</p> <p>Titre : Création comptes client</p> <p>Acteurs : Utilisateur</p> <p>Objectif : Il permet de créer des comptes client.</p>
<p>Description des scénarios :</p> <p>Pré condition : Utilisateur est authentifié.</p> <p>Post condition : Compte client créé.</p>

TABLE 3.4 – Description textuelle du cas d'utilisation « Authentification Utilisateur »

S'inscrire
Description des scénarios : Pré condition : Permet au utilisateur d'être membre. Post condition : Créer un compte dans le site web.
Scénario nominal : S'inscrire Le scénario se poursuit selon les étapes suivantes : <ol style="list-style-type: none"> L'utilisateur choisit de s'inscrire. Le système remplit le formulaire correspondant. L'utilisateur remplit le formulaire. Le système vérifie les données saisies. Le système affiche l'espace du membre..
Exception 1 : Si un champ lui manque le saisit ou présente une erreur de saisie, le système affiche un message d'erreur.

TABLE 3.5 – Description textuelle du cas d'utilisation « Inscription »

3.5 Réalisation

Interface Inscription :

The screenshot shows a web interface for account creation. The header is dark blue with the text 'SIGNALEMENT NAPU' and navigation links 'Accueil', 'Signaler un problème', and 'Connexion'. The main content area has a light blue background. On the left, a white box titled 'Créer un compte' contains several input fields, each with an '@' icon: 'CIN', 'Numéro d'inscription', 'E-mail', 'Prénom', 'Nom', 'Tel', and 'Delegate?' (with a checkbox). A blue 'S'inscrire' button is at the bottom of the form. To the right of the form is an illustration of a person sitting at a desk with a computer, a clock, and a login form overlay with fields for 'Email', 'Password', and 'Login'.

FIGURE 3.4 – Inscription

Interface connexion :



FIGURE 3.5 – Connexion

Interface modifier Mot de passe :

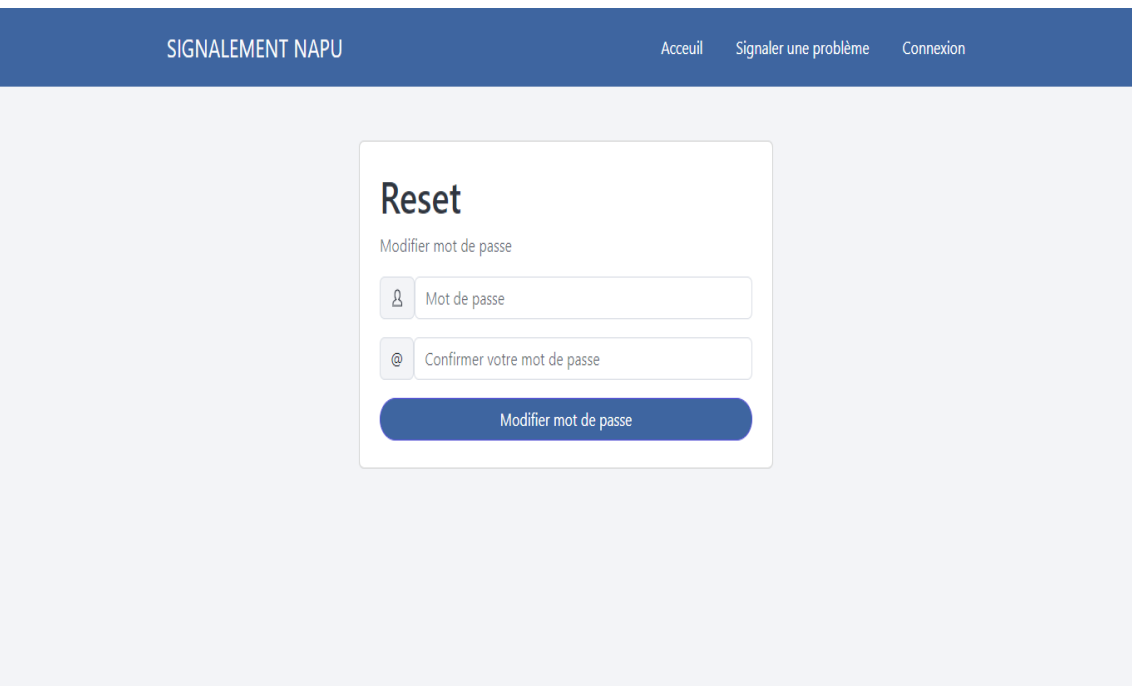
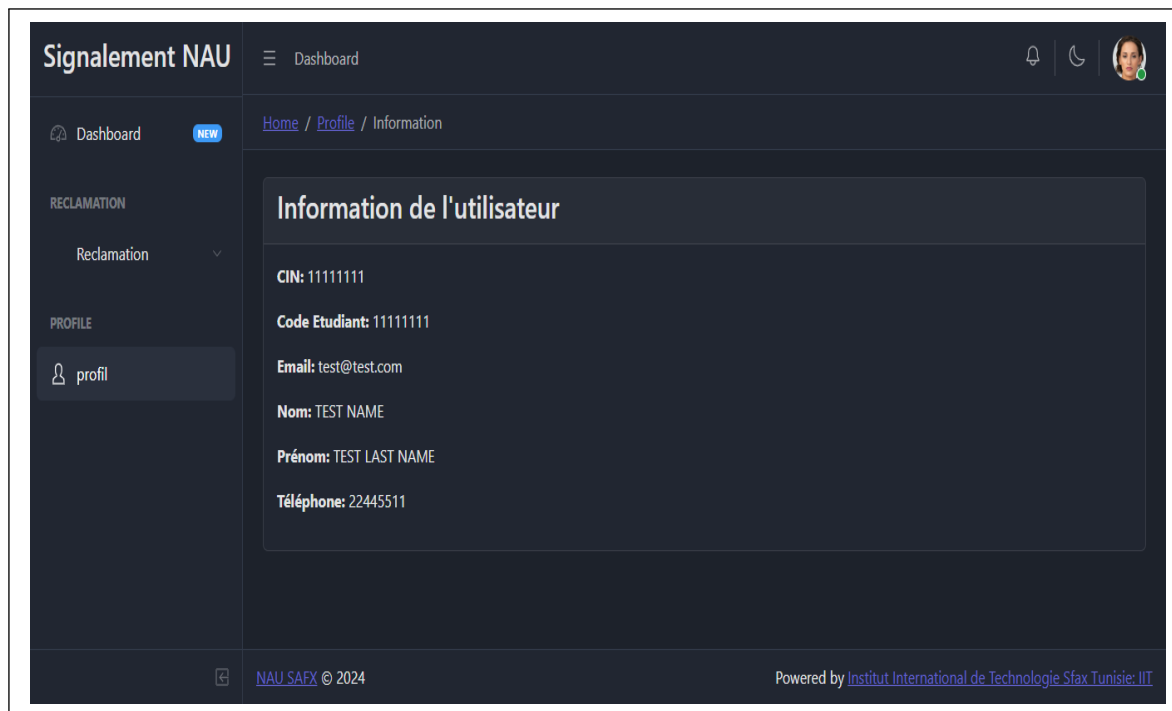


FIGURE 3.6 – Modifier Mot de passe

Interface voir les informations de profil :**FIGURE 3.7 – Voir les informations du profil**

3.6 CONCLUSION

Tout au long de ce chapitre, nous avons présenté les tâches traitées pour le premier sprint «Authentification». Puis, nous avons illustré les différents diagrammes UML. Nous avons aussi décrit les interfaces et le fonctionnement des différentes tâches traitées du module d'authentification, crucial pour garantir la sécurité de notre plateforme de gestion de réclamations en ligne.

Sprint 2 : Gestion des réclamations

Sommaire

4.1	INTRODUCTION	35
4.2	Objectifs attendus	35
4.3	Sprint Backlog	35
4.4	Langage de Modelisation	36
4.4.1	Diagramme de cas d'utilisation sprint 2	36
4.4.2	Diagramme de classe sprint 2	37
4.4.3	Description textuelle des cas d'utilisation	39
4.5	Réalisation	40
4.6	CONCLUSION	46

4.1 INTRODUCTION

Ce chapitre porte sur la présentation du deuxième sprint. Dans le cadre du développement de notre plateforme en ligne de gestion de réclamation, une fonctionnalité essentielle est l'authentification des utilisateurs. L'authentification permet de vérifier l'identité des utilisateurs avant de leur accorder l'accès aux fonctionnalités de la plateforme. Ce mécanisme de sécurité est crucial pour protéger les données sensibles et garantir que seules les personnes autorisées puissent accéder aux informations et aux services offerts par la plateforme.

4.2 Objectifs attendus

Le but du deuxième sprint est de réaliser le module de gestion des réclamations en ligne. Ce sprint se concentre sur la création, la mise à jour, le suivi et la résolution des réclamations soumises par les utilisateurs. L'objectif principal est d'assurer que les utilisateurs peuvent soumettre leurs plaintes de manière intuitive et que les administrateurs peuvent gérer ces réclamations efficacement.

4.3 Sprint Backlog

Le tableau ci-dessous représente le Sprint Backlog du deuxième sprint, détaillant les tâches nécessaires pour accomplir le module de gestion des réclamations. Chaque tâche est essentielle pour atteindre l'objectif principal de ce sprint, qui est de fournir une solution intuitive et efficace pour la soumission, le suivi et la résolution des réclamations des utilisateurs.

Backlog	Tâches	Temps alloués en jours	Priorité
- Création de réclamation	Sélectionner une spécialité , un type , le titre et la description	2	1
	Envoyer la réclamation	1	1
- Suivre l'état de réclamation	Afficher la liste des réclamations	3	1
	Filtrer les réclamations par date, par specialite , par type	2	2
- Visualiser le tableau de Bord	Afficher le nombre des réclamations par département ,par type , par specialite	3	1
- Envoyer un message sous forme de commentaire	Ecrire un message sous forme de commentaire dans la page de réclamation	2	1

TABLE 4.1 – Backlog sprint 2

4.4 Langage de Modelisation

4.4.1 Diagramme de cas d'utilisation sprint 2

Le diagramme de cas d'utilisation suivant décrit les principales fonctionnalités de la plateforme en ligne de gestion des réclamations pour une université. Ce diagramme identifie les interactions entre les utilisateurs (étudiants et administrateurs) et le système. Les acteurs principaux sont Admin et User.

Les administrateurs ont des cas d'utilisation tels que gérer les réclamations, visualiser et filtrer le tableau de bord, ajouter des départements et des spécialités, et envoyer des messages sous forme de commentaires. Les utilisateurs peuvent s'inscrire, s'authentifier, créer des réclamations, suivre leurs réclamations, entrer des informations de profil et envoyer des messages sous forme de commentaires.

Les cas d'utilisation incluent plusieurs inclusions, indiquant que certaines fonctionnalités dépendent d'autres actions. Par exemple, l'authentification est incluse pour toutes les actions

nécessitant un accès utilisateur. Le suivi et la gestion des réclamations incluent l'envoi de messages pour permettre la communication au sein du système

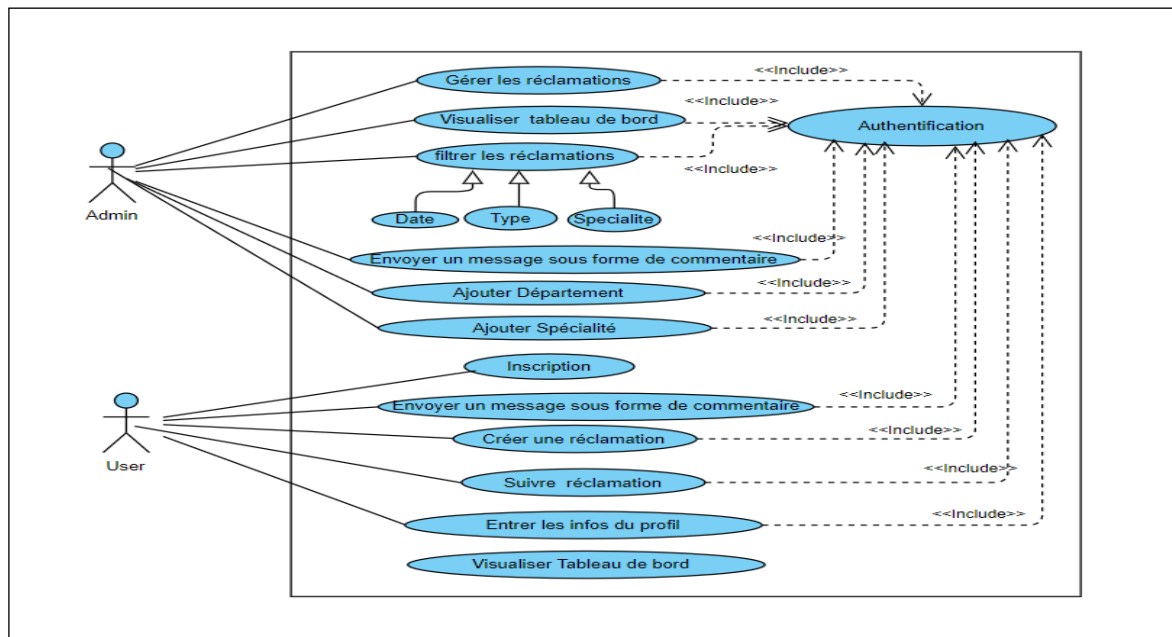


FIGURE 4.1 – Interface sprint 2

4.4.2 Diagramme de classe sprint 2

Le diagramme de classe ci-dessous décrit la structure de la plateforme en ligne de gestion des réclamations pour une université. La plateforme permet aux administrateurs de gérer les réclamations, de visualiser des tableaux de bord et d'ajouter des départements et des spécialités. Les utilisateurs peuvent s'inscrire, s'authentifier, créer des réclamations, suivre leurs réclamations, et visualiser le tableaux de bord personnalisés.

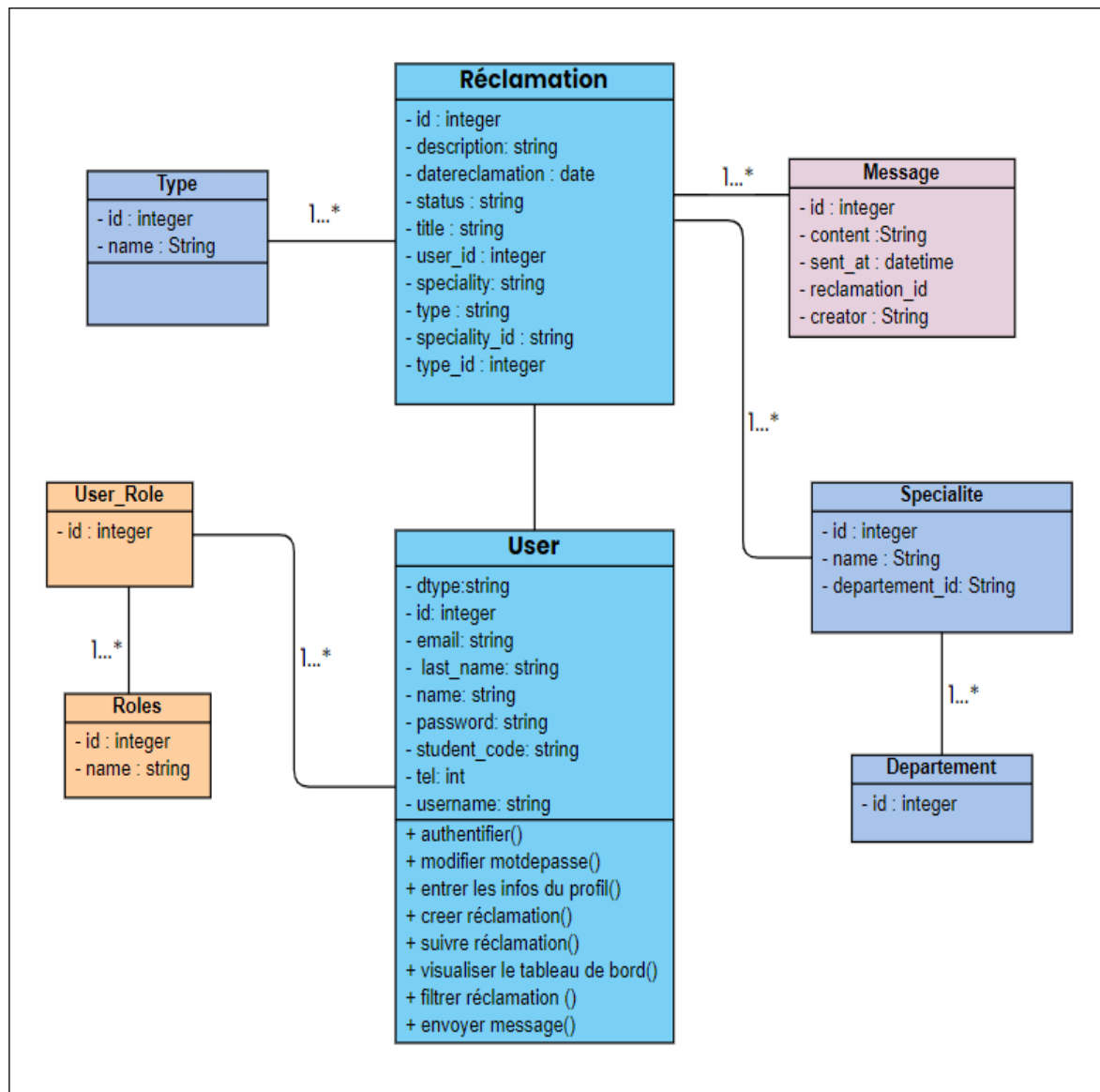


FIGURE 4.2 – Interface sprint 2

4.4.3 Description textuelle des cas d'utilisation

Gestion de réclamations
Description des scénarios : Pré condition : L'utilisateur est authentifié et possède les droits nécessaires. Post condition : La réclamation est créée et suivie avec succès.
Scénario nominal : Créer une réclamation Le scénario se poursuit selon les étapes suivantes : a. L'utilisateur accède à l'interface de gestion de réclamation. b. L'utilisateur choisit "Créer Réclamation". c. L'utilisateur sélectionne une spécialité et un type d'enregistrement. d. L'utilisateur saisit le titre et la description de la réclamation. e. L'utilisateur clique sur "Envoyer". f. Le système enregistre la réclamation et confirme la création.
Exception 1 : Le système affiche un message d'erreur si les champs requis ne sont pas remplis correctement..
Identification : Titre : Création et Suivi des Réclamations Acteurs : Utilisateur Objectif : Permettre à l'utilisateur de créer et suivre des réclamations.
Description des scénarios : Pré condition : Utilisateur est authentifié. Post condition : La réclamation est enregistrée et peut être suivie.

TABLE 4.2 – Description textuelle du cas d'utilisation « Gestion des Réclamations »

4.5 Réalisation

Pour l'utilisateur :

- Interface création de réclamation

The screenshot shows the 'Créer Reclamation - Suggestion' form in the Signalement NAU application. The interface is dark-themed. On the left is a sidebar with navigation links: Dashboard (NEW), RECLAMATION (Reclamation, Cree Reclamation, Suivre Reclamation), and PROFILE (profil). The main content area has a breadcrumb 'Home / Reclamation / crée Fiche' and a title 'Créer Reclamation - Suggestion'. The form includes a 'Spécialité' dropdown (placeholder: 'Sélectionnez une spécialité'), a 'Type d'enregistrement' dropdown (placeholder: 'Sélectionnez un type'), a 'Titre' text input, and a 'Description' text area. An 'Envoyer' button is at the bottom. The footer shows 'NAU SAFEX © 2024' and 'Powered by Institut International de Technologie Sfax Tunisie: IIT'.

FIGURE 4.3 – Interface création de réclamation

- Interface Tableau de bord pour suivre l'état de réclamation

The screenshot shows the dashboard in the Signalement NAU application. The sidebar is the same as in Figure 4.3. The main content area has a breadcrumb 'Home / Dashboard', a welcome message 'Bienvenue, TEST NAME, TEST LAST NAME', and a section 'Mes Réclamations :'. Below this is a table with the following data:

Titre	Statuts	Date de creation	Action
proposition d'une session de formation	En cours	2024-06-07	
Condition d'examan	En cours	2024-06-07	

The footer shows 'NAU SAFEX © 2024' and 'Powered by Institut International de Technologie Sfax Tunisie: IIT'.

FIGURE 4.4 – Interface Tableau de bord réclamation

- Interface suivre l'état de réclamation

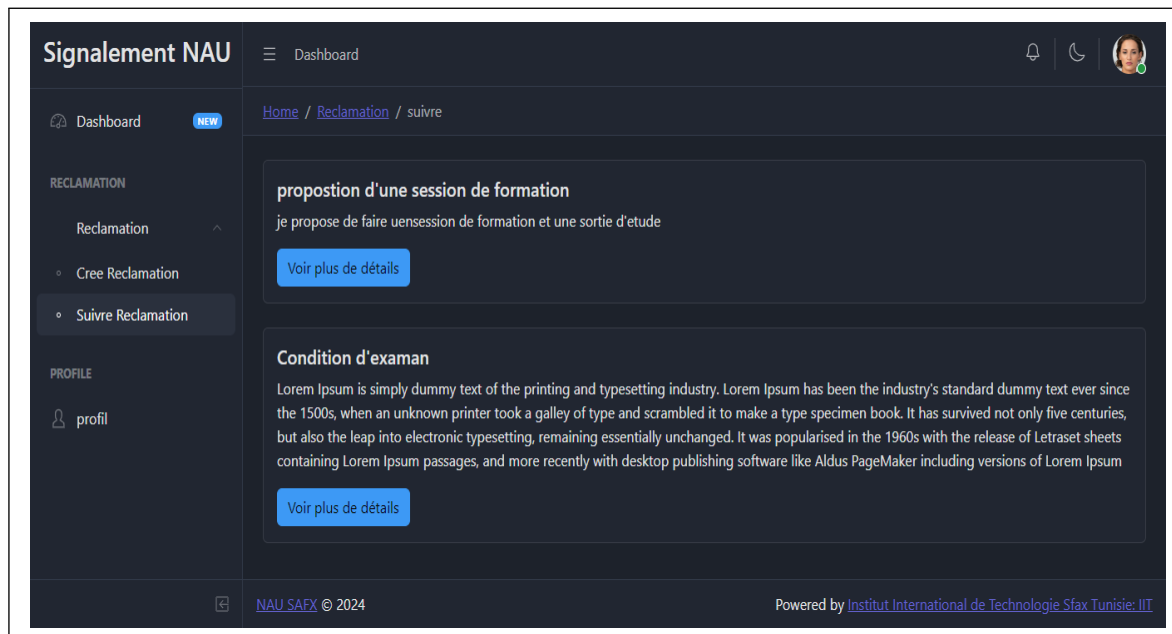


FIGURE 4.5 – Suivre l'état de réclamation

Pour l'admin :

- Interface détails de réclamations

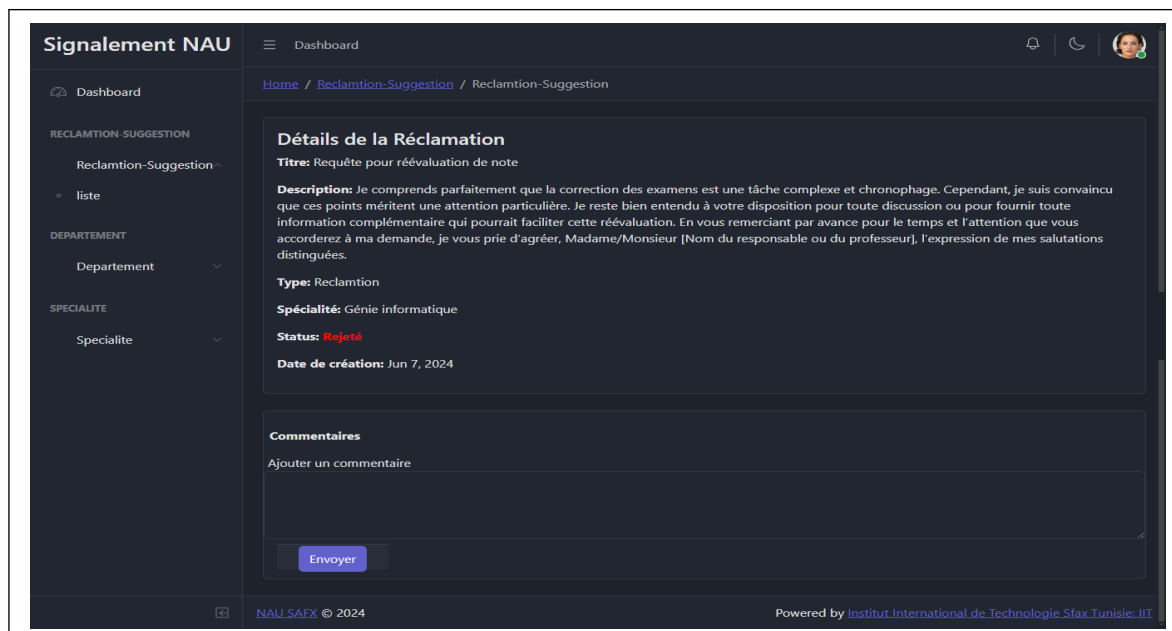


FIGURE 4.6 – Interface détails de réclamation

- Interface ajouter département

The screenshot shows the 'Ajouter Département' (Add Department) interface. The left sidebar contains the following menu items: Dashboard, RECLAMTION-SUGGESTION (with a sub-item 'Reclamtion-Suggestion'), DEPARTEMENT (with sub-items 'Departement', 'Liste', and 'Ajouter Departemant'), and SPECIALITE (with a sub-item 'Specialite'). The 'Ajouter Departemant' item is selected. The main content area is titled 'Create Department' and features a 'Department Name' input field and a 'Create' button. The breadcrumb trail at the top reads 'Home / Departemnt / ajouter'. The footer includes 'NAU SAFEX © 2024' and 'Powered by Institut International de Technologie Sfax Tunisie: IIT'.

FIGURE 4.7 – Interface Ajouter Département

- Interface listes des départements

The screenshot shows the 'Liste des départements' (List of Departments) interface. The left sidebar is identical to the previous screenshot, with 'Liste' selected under the 'DEPARTEMENT' section. The main content area displays a list of departments: 'Informatique', 'Mecanique', 'civil', 'Procédés', and 'Industriel'. The breadcrumb trail at the top reads 'Home / Departemnt / Departemnt'. The footer includes 'NAU SAFEX © 2024' and 'Powered by Institut International de Technologie Sfax Tunisie: IIT'.

FIGURE 4.8 – Interface liste des départements

- Interface ajouter spécialité

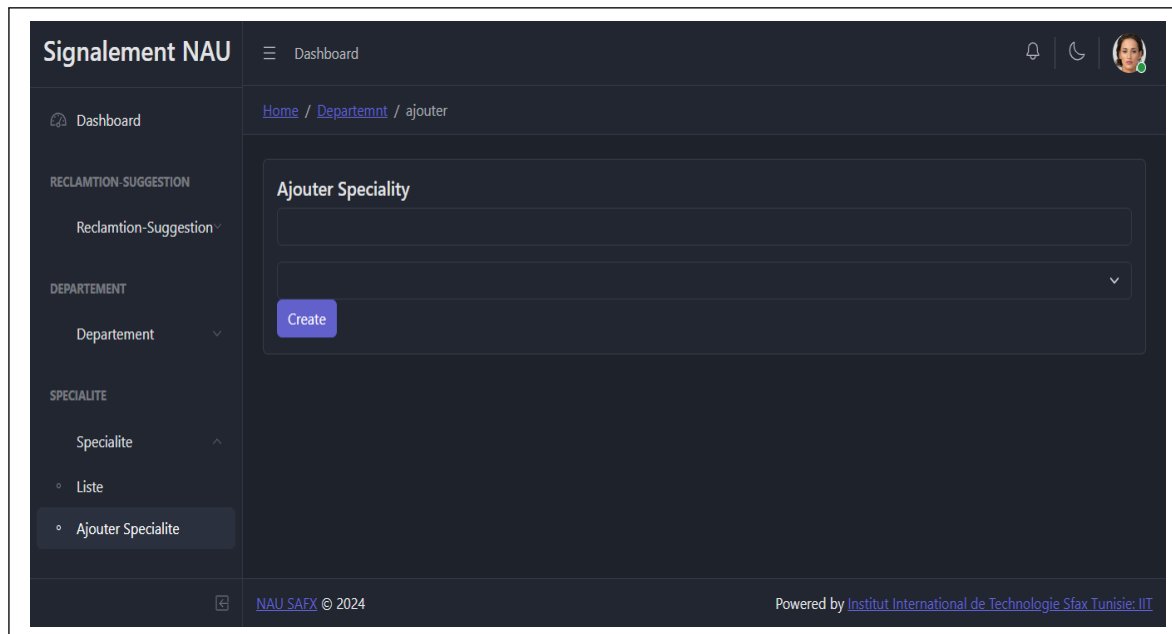


FIGURE 4.9 – Interface Ajouter Spécialité

- Interface listes des spécialités

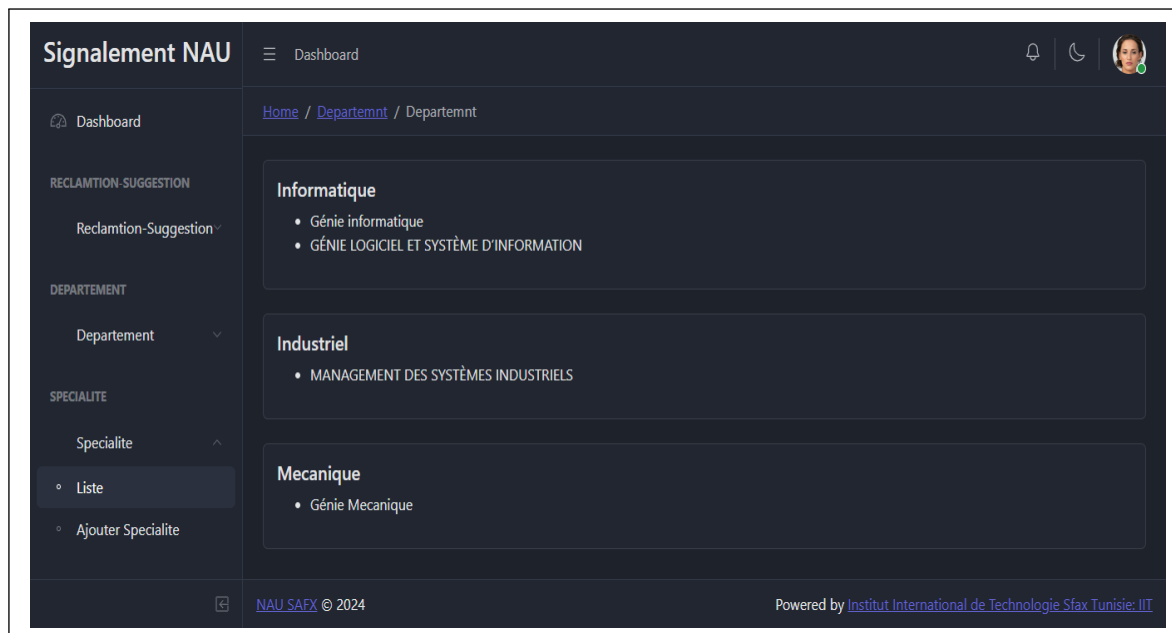


FIGURE 4.10 – Interface liste des spécialités

- Interface visualiser Tableau de bord des réclamations

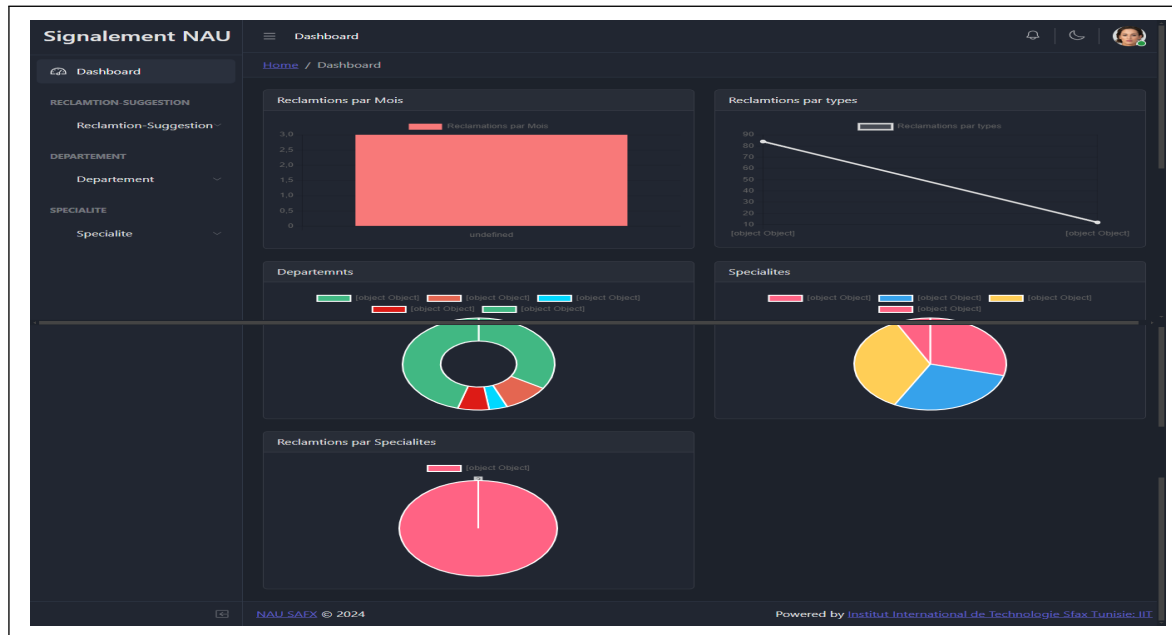


FIGURE 4.11 – Interface visualiser Tableau de bord des réclamations

- Interface détail de réclamations

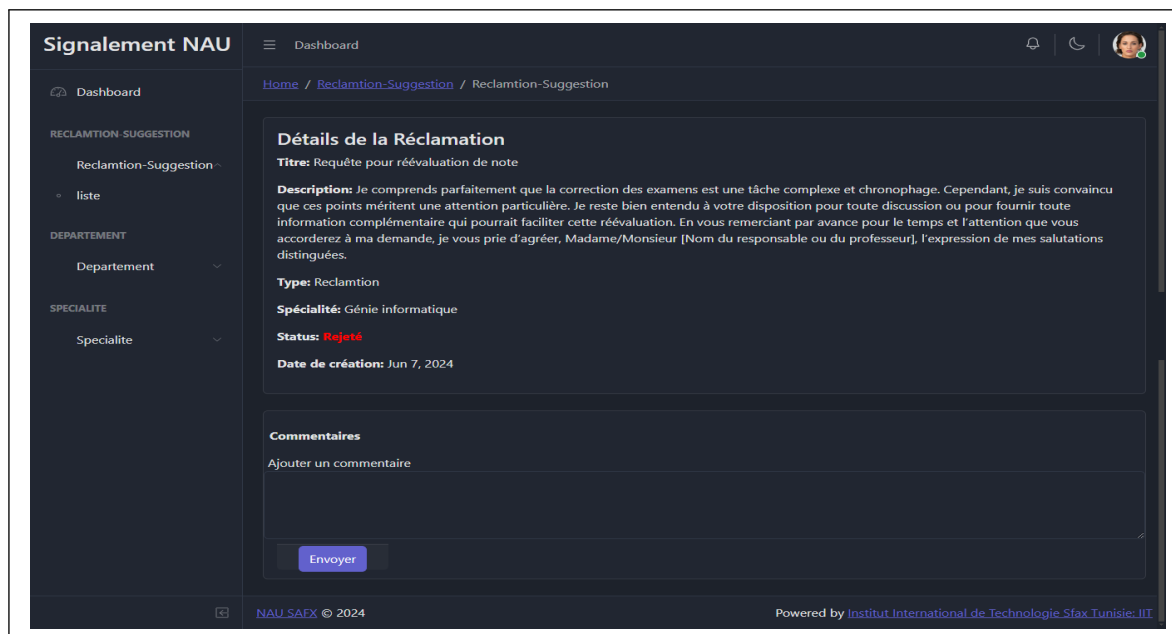


FIGURE 4.12 – Interface détails de réclamation

- Interface détail de réclamation

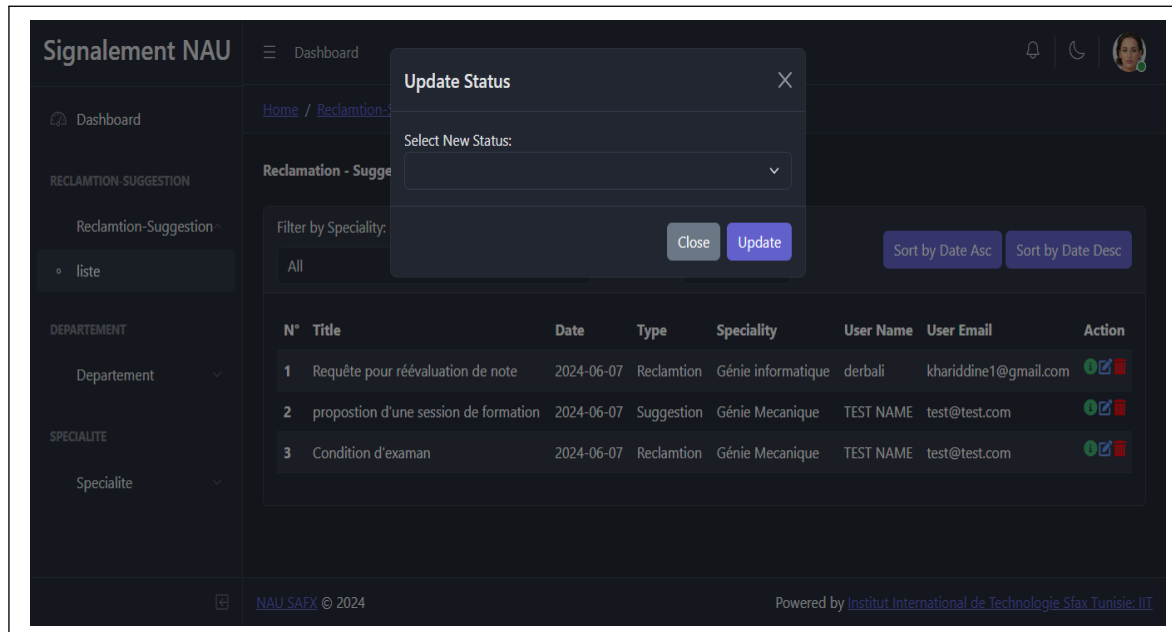


FIGURE 4.13 – Interface modifier réclamation

- Interface supprimer réclamation

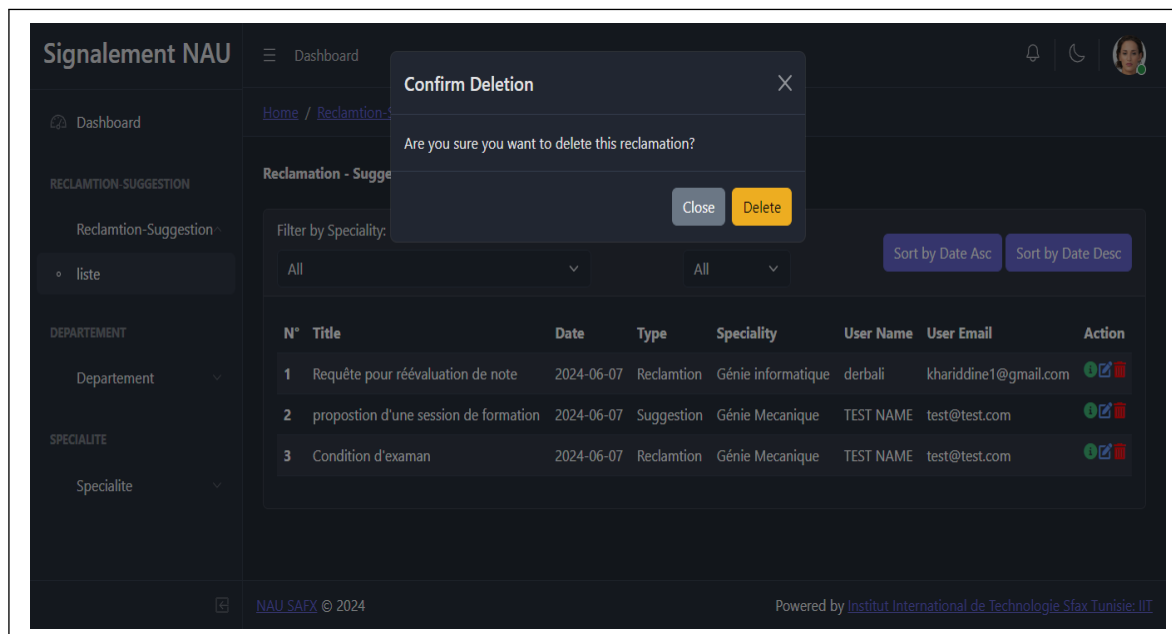


FIGURE 4.14 – Interface supprimer réclamation

4.6 CONCLUSION

Le Sprint 2 de la gestion des réclamations a permis d'implémenter plusieurs fonctionnalités clés pour améliorer l'efficacité et l'utilité du système de réclamation. Les tâches essentielles telles que la création de réclamations, y compris la sélection de spécialités et de types, ainsi que l'ajout de titres et de descriptions, ont été achevées avec succès. De plus, la fonctionnalité d'envoi de réclamations a été intégrée, assurant que les utilisateurs peuvent soumettre leurs plaintes sans problème.

En outre, le suivi des réclamations a été amélioré avec la possibilité d'afficher la liste des réclamations et de filtrer celles-ci par statut, offrant ainsi une meilleure visibilité et gestion des réclamations soumises.



CONCLUSION GÉNÉRALE

Ce projet s'inscrit dans le cadre d'un stage de projet de fin d'année effectué au sein de l'IIT. Par ailleurs, la réalisation de ce projet nous a été très bénéfique et très enrichissante sur le plan technique et théorique.

En conclusion, la plateforme en ligne de gestion de réclamations constitue une solution moderne et efficace pour gérer les réclamations. Elle améliore non seulement l'expérience utilisateur, mais elle permet également une gestion plus transparente et réactive des réclamations. Nous sommes confiants que cette plateforme continuera à évoluer et à s'adapter aux besoins futurs, garantissant ainsi une gestion des réclamations toujours plus efficace et satisfaisante pour toutes les parties prenantes.



BIBLIOGRAPHIE

- [1] **Scrum**. <https://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-FR.pdf>,
- [2] **Agile**. <https://agiliste.fr/introduction-methodes-agiles>
- [3] **Cycle de vie de projet scrum**. <https://www.cfi.ch/scrup-une-methode-de-developpement-agile/>,
- [4] **User Story**. User Story, <https://blog.thiga.co/glossaire/definition-user-story/>,
- [5] **UML**.UML, <https://www.uml.org/>,
- [6] **Diagramme de cas d'utilisation**. UML, <http://www.uml-sysml.org/diagrammes-uml-et-sysml/diagramme-uml/use-case-diagramme>,
- [7] **Diagramme de class UML**.Diagramme de class UML, <https://www.uml-sysml.org/diagrammes-uml-et-sysml/diagramme-uml/diagramme-de-classe>,