

Dédicaces

Je dédie ce travail à :

À mon cher père Adel, dont le soutien indéfectible et la présence ont toujours été une source d'inspiration et de motivation. Ce travail est le fruit de ses encouragements constants.

À ma chère mère Leila, un modèle de courage, d'amour et de persévérance. Son soutien inconditionnel et ses prières m'ont porté tout au long de ce parcours.

À ma partenaire Nehrine, pour sa bienveillance, son soutien inébranlable et les instants de joie partagés. Sa présence a été un pilier fondamental dans l'accomplissement de ce projet.

À tous ceux qui, de près ou de loin, m'ont soutenu et encouragé. Je dédie ce modeste travail en témoignage de ma profonde gratitude et de mon estime.

Remerciements

Je tiens à adresser mes plus sincères remerciements à toutes les personnes qui ont contribué, directement ou indirectement, à la réalisation de ce projet. Ces mots de gratitude marquent le début de mon rapport.

Tout d'abord, je souhaite exprimer ma profonde reconnaissance aux membres du jury pour avoir accepté d'évaluer mon projet de stage de fin d'études.

Je tiens à adresser mes vifs remerciements à mon encadrant au sein de SIGA, M. Iheb YOUSFI. Son accompagnement, ses conseils avisés et son soutien constant ont été précieux tout au long de ce projet. Sa disponibilité et son expertise m'ont grandement motivé et guidé.

Mes remerciements vont également à mon encadrante académique, Mme Sahar BEN AZIZA. Ses remarques pertinentes, sa disponibilité et ses orientations ont enrichi mon travail et m'ont permis de progresser.

Je remercie chaleureusement l'ensemble du corps enseignant de l'École supérieure privée d'ingénierie et de technologie (ESPRIT) pour les bases solides qu'ils m'ont transmises, essentielles à ma première expérience professionnelle.

Enfin, je tiens à exprimer ma gratitude à l'équipe technique et administrative de SIGA pour leur accueil, leur professionnalisme et l'environnement de travail favorable qu'ils ont su créer, contribuant ainsi au succès de ce projet.

Table des matières

Liste des figures	vii
Liste des tableaux	xi
Terminologie	xiii
Liste des abréviations	xvi
Introduction générale	1
1 Contexte général	3
1.1 Organisme d'accueil	3
1.1.1 SIGA	3
1.1.2 Secteur d'activité	4
1.2 Présentation du Projet	4
1.2.1 Présentation du Portail de Gestion des Approbations	4
1.2.2 Spécificités du Portail de Gestion des Approbations	5
1.2.3 Problématique	5
1.2.4 Situation actuelle et Critiques	5
1.3 Solution proposée	6
1.4 Méthodologie de gestion de projet	8
1.4.1 Méthodologies agiles	8
1.4.2 Méthode Scrum	8
1.5 Langage de modélisation : UML	11
1.6 Outils adoptés pour la gestion de projet	11
2 Sprint 0 : Analyse et Spécification des besoins	13
2.1 Introduction	13
2.2 Capture des besoins	13
2.2.1 Identification des acteurs	13
2.2.2 Identification des besoins fonctionnels	14
2.2.3 Identification des cas d'utilisation	15
2.2.4 Identification des besoins non fonctionnels	16
2.3 Diagramme de cas d'utilisation globale	17
2.4 Diagramme de classe global	18

2.5	Backlog du Produit	18
2.6	Planification de projet	21
2.6.1	Diagramme de Gantt	21
2.7	Environnement de travail	21
2.7.1	Environnement matériel	22
2.7.2	Environnement logiciel	22
2.8	Choix technologiques	24
2.8.1	Frontend	24
2.8.2	Backend	24
2.8.3	Workflow	24
2.8.4	Base de données	25
2.8.5	CI/CD	25
2.8.6	Conteneurisation	25
2.8.7	Orchestration	26
3	Sprint 1 : Accès et administration de base	27
3.1	Introduction	27
3.2	Backlog de Sprint 1	27
3.3	Raffinement des cas d'utilisation	28
3.3.1	Identification des acteurs du premier sprint	28
3.3.2	Raffinement du cas d'utilisation «S'authentifier»	28
3.3.3	Raffinement du cas d'utilisation «Se déconnecter»	29
3.3.4	Raffinement du cas d'utilisation «Gérer les utilisateurs»	29
3.3.5	Raffinement du cas d'utilisation «Mise en place et configuration de Camunda»	32
3.4	Conception	33
3.4.1	Diagramme de classe du sprint 1	33
3.4.2	Conception du cas d'utilisation «S'authentifier»	34
3.4.3	Conception du cas d'utilisation «Se déconnecter»	35
3.4.4	Conception du cas d'utilisation «Gérer les utilisateurs»	37
3.5	Réalisation	39
3.5.1	S'authentifier	39
3.5.2	Se déconnecter	39
3.5.3	Gérer les utilisateurs	40
3.6	Conclusion	44
4	Sprint 2 : Gestion des demandes	45
4.1	Introduction	45
4.2	Backlog du Sprint 2	45
4.3	Raffinement de cas d'utilisation	46
4.3.1	Identification des acteurs du deuxième sprint	46
4.3.2	Raffinement du Cas d'Utilisation «Gérer son Profil»	46
4.3.3	Raffinement du Cas d'Utilisation «Réinitialiser son Mot de Passe»	47
4.3.4	Raffinement du Cas d'Utilisation «Soumettre une Demande»	48
4.3.5	Raffinement du Cas d'Utilisation «Consulter ses Demandes»	49

4.3.6	Raffinement du Cas d'Utilisation «Consulter ses Congés»	50
4.3.7	Raffinement du Cas d'Utilisation «Traiter une Demande»	51
4.4	Conception	51
4.4.1	Diagramme de classe du sprint 2	51
4.4.2	Conception du Cas d'Utilisation «Gérer son Profil»	52
4.4.3	Conception du Cas d'utilisation «Réinitialiser son mot de passe»	54
4.4.4	Conception du Cas d'Utilisation «Soumettre une Demande»	56
4.4.5	Conception du Cas d'Utilisation «Consulter ses demandes»	59
4.4.6	Diagramme de Séquence	59
4.4.7	Conception du Cas d'Utilisation «Consulter ses congés»	60
4.4.8	Conception du Cas d'Utilisation «Traiter une demande»	61
4.5	Réalisation	63
4.5.1	Gérer son profil	63
4.5.2	Réinitialiser son mot de passe	63
4.5.3	Soumettre une demande	66
4.5.4	Consulter ses demandes	67
4.5.5	Consulter ses congés	68
4.5.6	Traiter une demande	68
4.6	Conclusion	69
5	Sprint 3 : Suivi et Supervision	70
5.1	Introduction	70
5.2	Backlog du Sprint 3	70
5.3	Raffinement de cas d'utilisation	71
5.3.1	Identification des acteurs du troisième sprint	71
5.3.2	Raffinement du cas d'utilisation «Recevoir une notification»	71
5.3.3	Raffinement du cas d'utilisation «Consulter les processus métiers»	72
5.3.4	Raffinement du cas d'utilisation «Consulter les demandes d'approbation» .	73
5.3.5	Raffinement du cas d'utilisation «Consulter les membres de l'équipe» . .	75
5.3.6	Raffinement du cas d'utilisation «Consulter ses crédits»	75
5.4	Conception	76
5.4.1	Diagramme de classe du sprint 3	77
5.4.2	Conception du cas d'utilisation «Recevoir une notification»	78
5.4.3	Conception du cas d'utilisation «Consulter les processus métiers»	79
5.4.4	Conception du cas d'utilisation «Consulter les demandes d'approbation» .	80
5.4.5	Conception du cas d'utilisation «Consulter les membres de l'équipe» . .	81
5.4.6	Conception du cas d'utilisation «Consulter ses crédits»	82
5.5	Réalisation	83
5.5.1	Recevoir une notification	83
5.5.2	Consulter les processus métiers	83
5.5.3	Consulter les demandes d'approbation	84
5.5.4	Consulter les membres de l'équipe	86
5.5.5	Consulter ses crédits	86
5.6	Conclusion	87

6 Sprint 4 : Analyse et Améliorations	88
6.1 Introduction	88
6.2 Backlog du Sprint 4	88
6.3 Raffinement de cas d'utilisation	89
6.3.1 Identification des acteurs du quatrième sprint	89
6.3.2 Raffinement du cas d'utilisation «Consulter le calendrier d'équipe»	89
6.3.3 Raffinement du cas d'utilisation «Consulter les tâches accomplies»	90
6.3.4 Raffinement du cas d'utilisation «Consulter les rapports»	91
6.3.5 Raffinement du cas d'utilisation «Implémentation d'un chatbot»	92
6.4 Conception	94
6.4.1 Diagramme de classe du sprint 4	94
6.4.2 Conception du Cas d'Utilisation «Consulter le calendrier d'équipe»	94
6.4.3 Conception du Cas d'Utilisation «Consulter les tâches accomplies»	96
6.4.4 Conception du Cas d'Utilisation «Consulter les rapports»	96
6.4.5 Conception du Cas d'Utilisation «Implémentation d'un chatbot»	97
6.5 Réalisation	99
6.5.1 Consulter le calendrier d'équipe	99
6.5.2 Consulter les tâches accomplies	100
6.5.3 Consulter les rapports	100
6.5.4 Implémentation d'un chatbot	101
6.6 Conclusion	103
7 Sprint 5 : Pipeline DevOps et Gestion GitOps	104
7.1 Introduction	104
7.2 Backlog du Sprint 5	104
7.3 Raffinement des cas d'utilisation	105
7.3.1 Identification des acteurs du Sprint 5	105
7.3.2 Raffinement du cas d'utilisation «Installation Automatisée»	106
7.3.3 Raffinement du cas d'utilisation «Workflows CI»	106
7.3.4 Raffinement du cas d'utilisation «Runner SonarQube»	107
7.3.5 Raffinement du cas d'utilisation «Gestion des Secrets»	108
7.3.6 Raffinement du cas d'utilisation «Exposition des Services»	109
7.3.7 Raffinement du cas d'utilisation «Surveillance et Rollback»	109
7.4 Conception	110
7.4.1 Diagramme de déploiement	110
7.4.2 Architecture du système	111
7.5 Réalisation	112
7.5.1 Installation Automatisée	112
7.5.2 Workflows CI	119
7.5.3 Runner SonarQube	123
7.5.4 Gestion des Secrets	124
7.5.5 Exposition des Services	125
7.5.6 Surveillance et Rollback	128
7.6 Conclusion	133

Conclusion générale	134
Nétographie	136
Bibliographie	140

Liste des figures

1.1 Logo de SIGA	3
1.2 Le cycle de vie en Scrum	9
1.3 Interface de ClickUp	11
1.4 Interface de ClickUp	12
2.1 Les acteurs	14
2.2 Diagramme de cas d'utilisation globale	17
2.3 Diagramme de classe globale	18
2.4 Diagramme de Gantt	21
2.5 Logo de Git	22
2.6 Logo de Github	22
2.7 Logo de WampServer	23
2.8 Logo de phpMyAdmin	23
2.9 Logo de VSCode	23
2.10 Logo de IntelliJ	24
2.11 Logo d'Angular	24
2.12 Logo de Spring Boot	24
2.13 Logo de Camunda	25
2.14 Logo de MySQL	25
2.15 Logo d'ArgoCD	25
2.16 Logo de Docker	26
2.17 Logo de Kubernetes	26
3.1 Diagramme du cas d'utilisation «S'authentifier»	28
3.2 Diagramme du cas d'utilisation «Se déconnecter»	29
3.3 Diagramme du cas d'utilisation «Gérer les utilisateurs»	29
3.4 Diagramme de classe du sprint 1	33
3.5 Diagramme de classe du cas d'utilisation «S'authentifier»	34
3.6 Diagramme de séquence du cas d'utilisation «S'authentifier»	35
3.7 Diagramme de classe du cas d'utilisation «Se déconnecter»	36
3.8 Diagramme de séquence du cas d'utilisation «Se déconnecter»	36
3.9 Diagramme de classe du cas d'utilisation «Gérer les utilisateurs»	37
3.10 Diagramme de séquence du cas d'utilisation «Gérer les utilisateurs»	38
3.11 Interface du cas d'utilisation «S'authentifier»	39
3.12 Interface du cas d'utilisation «Se déconnecter»	40

3.13	Interface du cas d'utilisation «Ajouter un utilisateur»	40
3.14	Interface du cas d'utilisation «Consulter les utilisateurs»	41
3.15	Interface du cas d'utilisation «Consulter les détails d'un utilisateur»	41
3.16	Interface du cas d'utilisation «Modifier un utilisateur»	42
3.17	Interface du cas d'utilisation «Supprimer un utilisateur»	42
3.18	Interface d'intégration de Camunda	43
3.19	Interface de configuration de Camunda	43
3.20	Interface cockpit de Camunda	44
4.1	Diagramme du cas d'utilisation «Gérer le profil»	46
4.2	Diagramme du cas d'utilisation «Réinitialiser son Mot de Passe»	47
4.3	Diagramme du cas d'utilisation «Soumettre une Demande»	48
4.4	Diagramme du cas d'utilisation «Consulter ses Demandes»	49
4.5	Diagramme du cas d'utilisation «Consulter ses Congés»	50
4.6	Diagramme du cas d'utilisation «Traiter une Demande»	51
4.7	Diagramme de classe du Sprint 2	52
4.8	Diagramme de classe du cas d'utilisation «Gérer son Profil»	52
4.9	Diagramme de séquence du cas d'utilisation «Gérer son Profil»	53
4.10	Diagramme de classe du cas d'utilisation «Réinitialiser son Mot de Passe»	54
4.11	Diagramme de séquence du cas d'utilisation «Réinitialiser son Mot de Passe»	55
4.12	Diagramme de classe du cas d'utilisation «Soumettre une demande»	56
4.13	Diagramme de séquence du cas d'utilisation «Soumettre une demande»	57
4.14	Diagramme BPMN demande de congé	58
4.15	Diagramme BPMN demande d'autorisation	58
4.16	Diagramme de classe du cas d'utilisation «Consulter ses demandes»	59
4.17	Diagramme de séquence du cas d'utilisation «Consulter ses demandes»	59
4.18	Diagramme de classe du cas d'utilisation «Consulter ses congés»	60
4.19	Diagramme de séquence du cas d'utilisation «Consulter ses congés»	60
4.20	Diagramme de classe du cas d'utilisation «Traiter une demande»	61
4.21	Diagramme de séquence du cas d'utilisation «Traiter une demande»	62
4.22	Interface du cas d'utilisation «Gérer son profil»	63
4.23	Page "Mot de passe oublié ?"	64
4.24	Confirmation de l'envoi du lien de réinitialisation	64
4.25	E-mail de réinitialisation de mot de passe	65
4.26	Page de création d'un nouveau mot de passe	65
4.27	Page de création d'une nouvelle demande de congé	66
4.28	Page de création d'un nouvelle demande d'autorisation	67
4.29	Page de consultation des demandes personnelles	67
4.30	Page de consultation des congés acceptés	68
4.31	Page de traitement des demandes	69
5.1	Diagramme du cas d'utilisation «Recevoir une notification»	71
5.2	Diagramme du cas d'utilisation «Consulter les processus métiers»	72
5.3	Diagramme du cas d'utilisation «Consulter les demandes d'approbation»	73
5.4	Diagramme du cas d'utilisation «Consulter les membres de l'équipe»	75

5.5	Diagramme du cas d'utilisation «Consulter ses crédits»	76
5.6	Diagramme de classe du Sprint 3	77
5.7	Diagramme de classe du cas d'utilisation «Recevoir une notification»	78
5.8	Diagramme de séquence du cas d'utilisation «Recevoir une notification»	78
5.9	Diagramme de classe du cas d'utilisation «Consulter les processus métiers»	79
5.10	Diagramme de séquence du cas d'utilisation «Consulter les processus métiers»	79
5.11	Diagramme de classe du cas d'utilisation «Consulter les demandes d'approbation»	80
5.12	Diagramme de séquence du cas d'utilisation «Consulter les demandes d'approbation»	80
5.13	Diagramme de classe du cas d'utilisation «Consulter les membres de l'équipe»	81
5.14	Diagramme de séquence du cas d'utilisation «Consulter les membres de l'équipe»	81
5.15	Diagramme de classe du cas d'utilisation «Consulter ses crédits»	82
5.16	Diagramme de séquence du cas d'utilisation «Consulter ses crédits»	82
5.17	Interface du cas d'utilisation «Recevoir une notification»	83
5.18	Interface du cas d'utilisation «Consulter les processus métiers»	84
5.19	Interface du cas d'utilisation «Consulter les demandes d'approbation»	85
5.20	Interface du cas d'utilisation «Consulter les demandes d'approbation»	85
5.21	Interface du cas d'utilisation «Consulter les membres de l'équipe»	86
5.22	Interface du cas d'utilisation «Consulter ses crédits»	87
6.1	Diagramme du cas d'utilisation «Consulter le calendrier d'équipe»	89
6.2	Diagramme du cas d'utilisation «Consulter les tâches accomplies»	90
6.3	Diagramme du cas d'utilisation «Consulter les rapports»	91
6.4	Diagramme de classe du Sprint 4	94
6.5	Diagramme de classe du cas d'utilisation «Consulter le calendrier d'équipe»	95
6.6	Diagramme de séquence du cas d'utilisation «Consulter le calendrier d'équipe»	95
6.7	Diagramme de classe du cas d'utilisation «Consulter les tâches accomplies»	96
6.8	Diagramme de séquence du cas d'utilisation «Consulter les tâches accomplies»	96
6.9	Diagramme de classe du cas d'utilisation «Consulter les rapports»	97
6.10	Diagramme de séquence du cas d'utilisation «Consulter les rapports»	97
6.11	Diagramme de classe du cas d'utilisation «Implémentation d'un chatbot»	98
6.12	Diagramme de séquence du cas d'utilisation «Implémentation d'un chatbot»	98
6.13	Interface du cas d'utilisation «Consulter le calendrier d'équipe»	99
6.14	Interface du cas d'utilisation «Consulter le calendrier d'équipe (voir details)»	99
6.15	Interface du cas d'utilisation «Consulter les tâches accomplies»	100
6.16	Interface du cas d'utilisation «Consulter les rapports»	100
6.17	Interface du cas d'utilisation «Consulter les rapports»	101
6.18	Interface du cas d'utilisation «Implémentation d'un chatbot»	102
6.19	Interface du cas d'utilisation «Implémentation d'un chatbot»	102
6.20	Interface du cas d'utilisation «Implémentation d'un chatbot»	103
7.1	Diagramme de déploiement du système	111
7.2	Architecture du système	112
7.3	Répertoire du Projet Ansible	113
7.4	Fichier ansible.cfg	113
7.5	Fichier main.yaml	114

7.6 Fichier inventory	114
7.7 Fichier install ansible dependencies.yaml	115
7.8 Fichier setup microk8s.yaml	116
7.9 Fichier setup_argocd.yaml	117
7.10 Fichier setup_argocd.yaml	118
7.11 Fichier deploy_argocd_application.yaml	119
7.12 Fichier Workflow Frontend	120
7.13 Fichier Workflow Frontend	121
7.14 Fichier Workflow Backend	122
7.15 Fichier Workflow Backend	123
7.16 Capture du Runner GitHub Actions	124
7.17 Capture du fichier secrets.yaml	125
7.18 Fichier ingress-frontend.yaml	126
7.19 Fichier ingress-backend.yaml	127
7.20 Fichier ingress-sonarqube.yaml	127
7.21 Interface ArgoCD - Vue générale de l'application "probation-suite"	128
7.22 Interface ArgoCD - Vue des services et secrets	129
7.23 Interface ArgoCD - Vue détaillée des déploiements et pods	130
7.24 Interface ArgoCD - Tableau de bord de déploiement	131
7.25 Architecture du dépôt des déploiements	132

Liste des tableaux

2.1	tab:Les rôles des acteurs	14
2.2	Backlog du Produit - Partie 1	19
2.3	Backlog du Produit - Partie 2	20
2.4	Spécifications des machines utilisées	22
3.1	Backlog du Sprint 1	27
3.2	Description textuelle du cas d'utilisation «S'authentifier»	28
3.3	Cas d'utilisation : Se déconnecter	29
3.4	Description textuelle du cas d'utilisation « Ajouter un utilisateur »	30
3.5	Description textuelle du cas d'utilisation « Consulter les utilisateurs »	30
3.6	Description textuelle du cas d'utilisation « Consulter les détails d'un utilisateur » .	31
3.7	Description textuelle du cas d'utilisation « Modifier un utilisateur »	31
3.8	Description textuelle du cas d'utilisation « Supprimer un utilisateur »	32
3.9	Description textuelle du cas d'utilisation «Mise en place et configuration de Camunda»	32
4.1	Backlog du Sprint 2	45
4.2	Description textuelle du Cas d'utilisation «Consulter le Profil»	46
4.3	Description textuelle du Cas d'utilisation «Modifier le Profil»	47
4.4	Description textuelle du Cas d'utilisation «Réinitialiser son Mot de Passe»	48
4.5	Description textuelle du Cas d'utilisation «Soumettre une Demande»	49
4.6	Description textuelle du Cas d'utilisation «Consulter ses Demandes»	50
4.7	Description textuelle du Cas d'utilisation «Consulter ses Congés»	50
4.8	Description textuelle du Cas d'utilisation «Traiter une Demande»	51
5.1	Backlog du Sprint 3 : Suivi et Supervision	70
5.2	Description textuelle du Cas d'utilisation «Recevoir une notification par mail» . . .	71
5.3	Description textuelle du Cas d'utilisation «Recevoir une notification push»	72
5.4	Description textuelle du Cas d'utilisation «Consulter les processus métiers»	73
5.5	Description textuelle du Cas d'utilisation «Consulter les demandes d'approbation»	74
5.6	Description textuelle du Cas d'utilisation «Consulter les détails d'une demande d'approbation»	74
5.7	Description textuelle du Cas d'utilisation «Consulter les membres de l'équipe» . .	75
5.8	Description textuelle du Cas d'utilisation «Consulter ses crédits»	76
6.1	Backlog du Sprint 4 : Fonctionnalités avancées	88

6.2	Description textuelle du Cas d'utilisation «Consulter le calendrier d'équipe»	90
6.3	Description textuelle du Cas d'utilisation «Consulter les tâches accomplies»	91
6.4	Description textuelle du Cas d'utilisation «Consulter les rapports»	92
6.5	Description textuelle du Cas d'utilisation «Implémentation d'un chatbot»	93
7.1	Backlog du Sprint 5 : Pipeline DevOps et Gestion GitOps	105
7.2	Description textuelle du Cas d'utilisation «Installation Automatisée»	106
7.3	Description textuelle du Cas d'utilisation «Workflows CI»	107
7.4	Description textuelle du Cas d'utilisation «Runner SonarQube»	108
7.5	Description textuelle du Cas d'utilisation «Gestion des Secrets»	108
7.6	Description textuelle du Cas d'utilisation «Exposition des Services»	109
7.7	Description textuelle du Cas d'utilisation «Surveillance et Rollback»	110

Terminologie

Automatisation : c'est lorsque un processus est complètement automatisé et ne requiert aucune intervention d'un utilisateur.

Environnement de développement : n'est pas seulement un IDE (Integrated Development Environment) comme Eclipse ou Visual Studio mais les bibliothèques, les fichiers de configuration et les serveurs en font partie aussi.

Intégration : est la combinaison des parties de code source séparées dont le but est de déterminer comment elles fonctionnent comme un tout.

Build : présente l'ensemble d'activités dont l'objectif est la construction de l'application. Il comprend plusieurs tâches comme :

- La mise à jour du code source depuis le gestionnaire de version
- La compilation du code source de l'application
- La génération des artefacts

Une tâche (Job) : est un ensemble d'instructions que le runner doit exécuter. [1] L'ensemble des jobs est défini dans un fichier YAML. Nous pouvons voir en temps réel le résultat d'une tâche afin que les développeurs puissent savoir si le job a été exécuté avec succès ou bien il a échoué. Un job peut être automatiquement lancé lorsqu'un commit est poussé ou bien manuellement exécuté. En effet, le job manuel peut être utile, par exemple, pour automatiser un déploiement, mais ne le déployer que lorsque quelqu'un l'approuve manuellement. En plus, il existe des moyens qui limitent le nombre de personnes pouvant exécuter un job.

Artefact (Artifact) : est généré par un job en cas d'exécution avec succès. Les utilisateurs peuvent télécharger cet artefact pour le tester ou bien l'utiliser directement [1].

Pipeline : est un groupe de jobs exécutés par étapes. En effet, tous les jobs d'une même étape sont exécutés en parallèle s'il y a suffisamment de Runners simultanés disponibles [3]. Le pipeline passe à l'étape suivante lorsque les jobs en cours terminent leur exécution sans erreurs. En cas d'échec d'au moins un job, l'exécution de pipeline s'arrête.

Runner : est une machine virtuelle, un container Docker ou un cluster de containers. Son rôle principal est de personnaliser l'environnement d'exécution du job [3]. En particulier, GitLab Runner est un projet Open Source utilisé dans la fonctionnalité d'intégration continue GitLab. Ce dernier communique avec le Runner à travers une API pour exécuter les scripts écrits dans la configuration GitLab-CI.

GOROCO : permet de récupérer la valeur exacte de la version [4]. Il se décompose en:

1. Generation: la génération du logiciel
2. Revision: les services pack (le cas d'un nombre impair implique que la version a encore des Bugs à éliminer pour la révision suivante)
3. Correction: les mises à jour

Liste des abréviations

BRMC: Cloud privé d'Orange France

CI: Continuous Integration

CD: Continuous Deployment

DSIF: Direction des Services d'Information France

DTSI: Direction Technique et du Système d' Information

IHM: Interface Homme-Machine

IT: Information Technology

ODE: Orange Deployment Engine

PLA: Produit Logiciel Applicatif

PLI: Produit Logiciel Infrastructure

SCM: Software Configuration Management

SI: Systèmes d'Information

SQAAS: SonarQube As A Service

URL: Uniform Resource Locator

YML: Yet Another Markup Language

Introduction générale

À une époque où les technologies de l'information redessinent les contours de notre quotidien, leur influence sur les organisations est devenue incontournable. Depuis janvier 2023, le nombre d'internautes a franchi la barre des 5,181 milliards à l'échelle mondiale, illustrant une adoption massive du numérique. Cette révolution technologique impose aux entreprises de repenser leurs processus pour gagner en agilité et en efficacité. Parmi ces processus, la gestion des approbations – qu'il s'agisse de demandes de congés, d'achats ou de budgets – se distingue par sa complexité. Souvent fragmentée, impliquant plusieurs acteurs et sujette aux erreurs, elle représente un défi que les outils traditionnels peinent à relever.

C'est dans ce contexte que SIGA, une entreprise spécialisée dans les solutions informatiques innovantes, s'impose comme un acteur clé pour accompagner les organisations vers une transformation numérique réussie. Reconnue pour son expertise, SIGA développe des systèmes qui simplifient les opérations tout en répondant aux exigences d'un monde en constante évolution. Mon projet de fin d'études s'inscrit pleinement dans cette mission : la création d'un « Portail de Gestion des Approbations avec Workflows Dynamiques et Déploiement Automatisé ». Ce portail vise à centraliser et fluidifier la gestion des demandes, en permettant aux utilisateurs de les soumettre via une interface intuitive, de suivre leur progression en temps réel, et de bénéficier d'une validation automatisée grâce à des workflows dynamiques.

Pour concrétiser cette vision, j'ai mobilisé un éventail de technologies modernes : Angular pour une interface utilisateur conviviale, Spring Boot pour un backend robuste, et Camunda pour orchestrer des processus flexibles et évolutifs. En adoptant une approche DevOps, j'ai intégré un pipeline CI/CD avec Jenkins pour automatiser les tests et le déploiement, tout en utilisant Docker pour la conteneurisation et Kubernetes pour garantir la scalabilité. Ces choix techniques reflètent un double objectif : offrir une expérience utilisateur optimale tout en assurant une solution fiable et maintenable pour les équipes techniques.

Ce rapport retrace les étapes de cette réalisation au sein de SIGA, structuré en sept chapitres qui suivent la progression du projet, depuis son contexte jusqu'à son déploiement final.

Le premier chapitre explore le contexte général du projet. Nous présenterons SIGA, ses ambitions, et les spécificités du portail, avant d'analyser la problématique des processus d'approbation et de poser les bases du projet.

Le deuxième chapitre, correspondant au Sprint 0, se concentre sur l'analyse et la spécification des besoins. Nous définirons les exigences fonctionnelles et non fonctionnelles, ainsi que les bases de la modélisation des workflows.

Le troisième chapitre, Sprint 1, aborde l'accès et l'administration de base du portail. Nous détaillerons la mise en place des fonctionnalités initiales, telles que l'authentification et la gestion des utilisateurs, en utilisant Angular et Spring Boot.

Le quatrième chapitre, Sprint 2, est dédié à la gestion des demandes. Nous présenterons la soumission, le suivi et la validation des demandes, en intégrant les workflows dynamiques avec Camunda.

Le cinquième chapitre, Sprint 3, se focalise sur le suivi et la supervision. Nous explorerons les fonctionnalités permettant aux utilisateurs de suivre les processus, consulter les demandes d'approbation et gérer les crédits de congés.

Le sixième chapitre, Sprint 4, met l'accent sur l'analyse et les améliorations. Nous décrirons les outils d'analyse avancés, comme la consultation du calendrier d'équipe, des tâches accomplies et des rapports, ainsi que l'intégration d'un chatbot pour assister les administrateurs.

Le septième chapitre conclut le projet en abordant la partie DevOps et le déploiement. Nous détaillerons l'utilisation de 'ArgoCD' pour l'intégration continue, Docker pour la conteneurisation, et Kubernetes pour l'orchestration, garantissant une scalabilité et une fiabilité optimales, accompagnées de la documentation associée.

En conclusion de cette introduction, ces sept chapitres forment un récit cohérent qui reflète les différentes étapes de notre travail. Ensemble, ils illustrent comment ce portail, né d'une problématique concrète, a évolué pour devenir une solution complète, alignée sur les besoins des utilisateurs et les standards technologiques actuels. Ce projet au sein de SIGA a été une opportunité unique de conjuguer créativité, rigueur technique et collaboration, dans un effort pour rendre les processus d'approbation plus simples et plus intelligents.

Chapitre 1

Contexte général

Introduction

Dans ce premier chapitre, nous allons poser les bases de notre travail en explorant son contexte général. Nous débuterons par une présentation de l'organisme d'accueil, la société SIGA, en mettant en lumière son domaine d'expertise et son rôle. Ensuite, nous introduirons le projet en détail, avant d'examiner l'existant afin d'identifier ses limites et les opportunités d'amélioration que notre solution vise à concrétiser. Enfin, nous exposerons la méthodologie suivie pour concevoir ce projet ainsi que les grandes lignes de la solution proposée.

1.1 Organisme d'accueil

1.1.1 SIGA

Créée en 1996, SIGA (Système Informatique et Gestion Automatisée) s'est imposée comme un acteur majeur dans le domaine du développement de logiciels et des solutions informatiques en Tunisie. Basée à Tunis, l'entreprise regroupe une équipe de plus de vingt ingénieurs et six consultants seniors, cumulant entre 5 et 25 ans d'expérience dans la conception et l'intégration de systèmes d'information. Depuis sa fondation, SIGA se distingue par son expertise multidisciplinaire, offrant des services qui couvrent l'ensemble du cycle de vie d'un produit informatique : de l'analyse des besoins à la mise en œuvre, en passant par le développement, la formation et la maintenance.



Figure 1.1: Logo de SIGA

1.1.2 Secteur d'activité

SIGA déploie son expertise dans divers secteurs, proposant des solutions technologiques adaptées aux besoins spécifiques de ses clients. Ses principaux domaines d'intervention incluent :

- **Banque et assurance** : développement de systèmes pour optimiser la gestion des processus financiers et administratifs.
- **Industrie** : conception de logiciels pour améliorer la gestion de la production et des ressources.
- **Transports** : mise en place de solutions facilitant la logistique et la coordination opérationnelle.
- **Télécommunications** : création de systèmes pour rationaliser les interactions et la gestion des données.
- **Autres secteurs** : accompagnement à la digitalisation des opérations pour des entreprises variées, avec une présence étendue en Tunisie et dans plusieurs pays africains (Mali, Côte d'Ivoire, Tchad, Mauritanie).

1.2 Présentation du Projet

Le projet que j'ai réalisé s'inscrit dans le cadre de mon projet de fin d'études en dernière année à l'École Supérieure Privée d'Ingénierie et de Technologie (ESPRIT). J'ai effectué mon stage au sein de SIGA, une entreprise reconnue pour son expertise dans le développement de logiciels et de solutions informatiques innovantes en Tunisie.

Dans cette section, nous présentons notre projet en détaillant, dans un premier temps, un aperçu du « Portail de Gestion des Approbations avec Workflows Dynamiques et Déploiement Automatisé » et ses spécificités. Dans un second temps, nous aborderons la problématique, l'étude de l'existant, notre contribution ainsi que la méthodologie adoptée pour mener à bien cette mission.

1.2.1 Présentation du Portail de Gestion des Approbations

Le projet « Portail de Gestion des Approbations avec Workflows Dynamiques et Déploiement Automatisé » est une solution conçue pour centraliser et optimiser la gestion des processus d'approbation au sein des organisations. Son objectif principal est de simplifier la soumission, le suivi et la validation des demandes – telles que les congés ou les autorisations – en offrant une interface intuitive et des mécanismes automatisés. Ce portail se distingue par sa capacité à adapter dynamiquement les workflows en fonction des besoins spécifiques de chaque demande, garantissant ainsi une flexibilité et une personnalisation optimales.

Le portail prend en charge divers types de demandes et génère des outputs variés, notamment des notifications par email ou push notification, des statuts en temps réel, et des rapports analytiques sous forme de tableaux de bord. À terme, il vise à gérer un volume significatif de requêtes quotidiennes tout en assurant une traçabilité complète des actions effectuées. Développé avec Angular pour le frontend, Spring Boot pour le backend, et Camunda pour l'orchestration des workflows, le système repose sur une architecture moderne déployée via Docker et orchestrée par Kubernetes. Il est hébergé sur plusieurs environnements distincts : développement (DEV), tests (TEST) et production (PROD), avec une infrastructure évolutive adaptée à chaque contexte.

Chez SIGA, ce projet joue un rôle clé dans la digitalisation des processus internes, et toute interruption pourrait affecter la fluidité des opérations, rendant sa disponibilité une priorité essentielle.

1.2.2 Spécificités du Portail de Gestion des Approbations

Le portail est un outil de gestion des processus qui permet aux utilisateurs de soumettre et de suivre des demandes via différents canaux et fonctionnalités, notamment :

- **Workflows dynamiques:** des processus personnalisables définis avec Camunda.
- **Suivi en temps réel:** un statut actualisé et un historique des actions.
- **Reporting:** des tableaux de bord pour analyser les processus.
- **Notifications:** des alertes automatiques par email ou temps réel.

1.2.3 Problématique

Dans de nombreuses organisations, les processus d'approbation sont encore réalisés de manière manuelle. Cela engendre plusieurs problématiques :

- **Lenteur des traitements:** Les délais sont souvent rallongés à cause des allers-retours entre les différents acteurs.
- **Manque de traçabilité :** Il est difficile de connaître l'état d'avancement d'une demande ou de retracer les actions passées.
- **Risque d'erreurs humaines :** En raison du manque d'automatisation, certaines étapes peuvent être oubliées ou mal exécutées.

1.2.4 Situation actuelle et Critiques

Actuellement, pour obtenir une approbation (congé, autorisation...), le processus suit généralement les étapes suivantes :

1. Remplissage manuel d'un formulaire papier ou d'un document bureautique (Word, Excel).
2. Transmission physique ou par email au supérieur hiérarchique.
3. Approbation manuelle, parfois verbale, sans traçabilité.
4. Notification au demandeur uniquement si ce dernier relance.

Inconvénients:

- **Non-centralisation** : Les demandes sont éparpillées (emails, papiers, discussions).
- **Perte de temps** : Les relances sont fréquentes.
- **Absence de supervision globale** : Aucune visibilité d'ensemble pour l'administration.
- **Aucune possibilité d'analyse** : Pas de données statistiques sur les demandes, les délais, les refus, etc.

1.3 Solution proposée

Afin de répondre aux limites identifiées dans le processus actuel, il est proposé de mettre en place un **Portail Web de Gestion des Approbations** moderne, centralisé et automatisé. Ce portail repose sur une architecture modulaire et des technologies éprouvées pour assurer une solution performante, évolutive et facilement maintenable.

Automatisation des workflows avec Camunda BPM

Le cœur de la solution repose sur l'intégration de **Camunda BPM**, un moteur de workflow open-source basé sur le standard BPMN (Business Process Model and Notation). Chaque type de demande (congé, achat, autorisation, etc.) est modélisé sous forme de processus graphique. Ces workflows définissent les différentes étapes à suivre, les rôles impliqués et les règles de validation.

L'utilisation de Camunda permet :

- La **définition claire et visuelle** des processus métiers.
- L'**exécution automatique** des étapes selon les règles définies.
- La **flexibilité** de modifier les processus sans redéployer l'application.
- L'**assignation dynamique** des tâches aux utilisateurs concernés.

Digitalisation de la soumission des demandes

L'interface utilisateur, développée avec **Angular**, permet aux utilisateurs de soumettre leurs demandes via des formulaires en ligne ergonomiques et adaptés au type de demande. Chaque formulaire est connecté à une instance de processus Camunda qui orchestre automatiquement les étapes suivantes. Cette digitalisation permet de :

- Réduire les erreurs de saisie grâce à des contrôles intégrés.
- Gagner du temps en supprimant les supports papier.
- Standardiser la soumission et le traitement des demandes.

Suivi en temps réel et notifications automatiques

Chaque utilisateur peut suivre en temps réel le statut de ses demandes (approuvée, rejetée ou en cours de traitement). Le portail intègre un système de **notifications en temps réel** directement dans l'interface de l'application, basé sur des technologies de type WebSocket:

- Alerte à l'approbateur lorsqu'une tâche lui est assignée.
- Notification au demandeur une fois la décision prise.
- Rappels pour les tâches non traitées dans un délai défini.

Ce mécanisme renforce la réactivité des acteurs et limite les retards.

Infrastructure moderne, scalable et résiliente

L'ensemble de l'application est conteneurisé à l'aide de **Docker** afin de garantir une portabilité maximale et un déploiement simplifié. Le déploiement se fait sur un cluster **Kubernetes** pour bénéficier :

- D'une haute disponibilité des services.
- D'un équilibrage de charge automatique.
- D'une scalabilité horizontale en fonction de la charge.

Pipeline DevOps pour l'intégration et le déploiement continu

Un pipeline CI/CD est mis en place avec **ArgoCD** pour automatiser :

- La compilation et les tests des composants backend et frontend.
- La construction des images Docker.
- Le déploiement automatique sur l'environnement de production.

Cela permet de garantir la rapidité, la fiabilité et la répétabilité du processus de livraison.

Analyse des performances et tableaux de bord

Des tableaux de bord sont mis à disposition des administrateurs pour visualiser :

- Le nombre de demandes traitées par type ou par service.
- Les délais moyens de traitement.
- Les goulets d'étranglement dans les workflows.

Ces indicateurs permettent d'optimiser les processus métier et de prendre des décisions basées sur des données concrètes.

1.4 Méthodologie de gestion de projet

Afin d'assurer le bon déroulement de notre projet, tout en respectant les contraintes de qualité, de délais et les attentes fonctionnelles, il est essentiel d'adopter une méthodologie adaptée qui structure et optimise les différentes phases de travail.

1.4.1 Méthodologies agiles

Les méthodologies agiles représentent une nouvelle génération d'approches de gestion de projet, axées sur des cycles itératifs, adaptatifs et incrémentaux. Elles permettent une réévaluation régulière des besoins, facilitent l'évolution des exigences et garantissent une livraison progressive et maîtrisée des fonctionnalités. [2].

Ces méthodes reposent sur des principes forts, tels que :

- L'acceptation du changement tout au long du cycle de développement.
- La collaboration constante avec le client, considéré comme partie prenante active.
- L'accent mis sur la valeur fonctionnelle livrée à chaque itération.
- La priorité donnée aux interactions humaines plutôt qu'aux processus rigides.

Les méthodes agiles favorisent ainsi la réactivité, l'amélioration continue et l'adaptabilité face à un environnement en perpétuelle évolution.

1.4.2 Méthode Scrum

Parmi les différentes approches agiles, la méthodologie Scrum s'est imposée comme une référence incontournable. Elle fournit un cadre structuré mais souple permettant d'organiser efficacement le travail d'équipe, tout en maximisant la productivité et la qualité du produit livré.

Nous avons opté pour Scrum, car cette méthodologie répond parfaitement aux besoins des projets complexes en nous offrant une flexibilité optimale et une forte capacité d'adaptation face aux imprévus.

Le contexte méthodologique de Scrum s'articule, tel que le montre la figure 1.2, autour de la définition des fonctions, d'un tempo bien rythmé, d'éléments concrets et des réunions particulières.

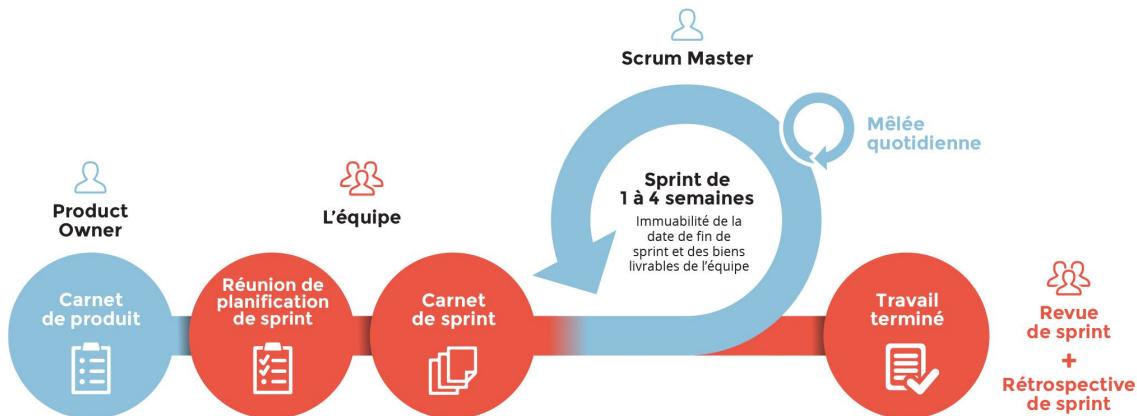


Figure 1.2: Le cycle de vie en Scrum

Les Rôles :

Scrum définit trois rôles clés, indispensables à la réussite du projet :

- **Le Product Owner** : Il est fréquent qu'un spécialiste de domaine assume le rôle de représentant officiel du client au sein d'un projet Scrum. Son rôle principal consiste à définir les besoins fonctionnels, établit les priorités dans le Product Backlog et veille à la satisfaction du client.
- **Le Scrum Master** : Garant de la méthode Scrum, il facilite le travail de l'équipe en supprimant les obstacles et en s'assurant que les pratiques agiles sont correctement appliquées.
- **L'Équipe** : Les membres responsable de la conception, du développement, des tests et de la livraison des fonctionnalités à la fin de chaque sprint. Ils sont auto-organisés et multidisciplinaires.

Le Rythme Itératif :

- **Sprint** : Le travail est organisé en cycles courts appelés sprints, d'une durée généralement comprise entre 2 et 4 semaines. À chaque sprint, une version incrémentale du produit est développée, testée et potentiellement livrable.

Les Artefacts :

- **Product Backlog** : Liste hiérarchisée de toutes les fonctionnalités à développer, rédigées sous forme de User Stories. Ce backlog est régulièrement mis à jour en fonction des retours du client.
- **Sprint Backlog** : Sous-ensemble du Product Backlog sélectionné pour être réalisé lors d'un sprint donné. Il contient les User Stories détaillées et les tâches associées.

les Evénements :

- **Sprint Planning** : Réunion de planification marquant le début de chaque sprint. L'équipe sélectionne les éléments à développer et définit les objectifs à atteindre.
- **Daily Scrum (Mêlée quotidienne)** : Courte réunion quotidienne, souvent appelée "stand-up" en raison de sa brièveté (15 minutes maximum), visant à synchroniser les efforts de l'équipe. Chaque membre partage trois points : ce qu'il a accompli depuis la dernière mêlée, ce qu'il prévoit de faire avant la prochaine, et les éventuels blocages rencontrés. Facilitée par le Scrum Master, cette rencontre n'est pas destinée à résoudre les problèmes sur-le-champ, mais à les identifier pour un traitement ultérieur. Elle favorise la transparence, renforce la collaboration et permet d'ajuster rapidement le plan du sprint si nécessaire.
- **Definition of Done (DoD)** : Ensemble de critères définissant qu'une tâche est considérée comme terminée (tests réussis, documentation rédigée, code relu, etc.).
- **Definition of Ready (DoR)** : Ce concept repose sur une compréhension commune des préparatifs requis pour une tâche, intégrant une liste de contrôle pour établir les User Stories. Pour être considérées comme prêtes à être mises en œuvre, les User Stories doivent répondre à plusieurs exigences, notamment la description exhaustive des scénarios d'utilisation, la disponibilité des ressources essentielles, des critères d'acceptation clairement définis et vérifiables, une taille adaptée, une présentation à l'équipe par le propriétaire du produit, des méthodes de test fonctionnelles, une estimation fiable de leur taille relative basée sur l'expérience utilisateur, ainsi que la spécification des appareils ciblés.
- **Revue de Sprint** : Réunion organisée à la fin de chaque sprint pour évaluer le travail accompli. L'équipe présente l'incrément terminé au Product Owner et aux parties prenantes, qui valident sa conformité aux attentes. Si nécessaire, des ajustements sont apportés au "Product Backlog" pour refléter l'évolution des besoins ou des priorités. D'une durée maximale de 4 heures pour un sprint d'un mois, cette étape vise à garantir que le projet progresse efficacement, tout en maintenant un alignement constant avec les objectifs stratégiques et les exigences changeantes.
- **Rétrospective de Sprint** : Rencontre tenue immédiatement après la revue, dédiée à l'amélioration continue. L'équipe, guidée par le Scrum Master, analyse le sprint écoulé en identifiant ce qui a bien fonctionné, les difficultés rencontrées, et les opportunités d'optimisation. Des actions concrètes sont définies pour le sprint suivant, comme ajuster les processus ou résoudre des blocages récurrents. Limitée à 3 heures pour un sprint d'un mois, cette réunion renforce l'auto-organisation de l'équipe et assure une progression constante dans la qualité et l'efficacité du travail.

1.5 Langage de modélisation : UML

Le langage UML (Unified Modeling Language) est un langage standardisé de modélisation utilisé dans la conception orientée objet des systèmes logiciels. Il permet de représenter, visualiser, spécifier, construire et documenter de manière formelle les différents aspects d'un système, tant structurels que comportementaux.

UML repose sur un ensemble de diagrammes graphiques permettant de décrire les éléments clés d'un système : les classes, les objets, les interactions, les activités, les cas d'utilisation, etc. Ces représentations facilitent la communication entre les différents acteurs du projet (développeurs, analystes, chefs de projet) et garantissent une compréhension commune du fonctionnement global du système.

Dans le cadre de notre projet, nous avons choisi d'utiliser UML pour modéliser les différentes composantes de notre application. Ce choix s'explique par la richesse expressive d'UML, sa capacité à structurer efficacement un système complexe, et sa large adoption dans l'industrie comme outil de conception et de documentation. La figure 1.3 présente le logo du language de modélisation UML.



Figure 1.3: Interface de ClickUp

1.6 Outils adoptés pour la gestion de projet

Pour assurer le succès de notre projet et garantir une livraison réussie de notre produit, nous avons mis en place un ensemble d'outils de gestion de projet stratégiques, parmi lesquels nous pouvons citer :

- **Git** : Un système de contrôle de version permettant de suivre les modifications apportées aux fichiers de code, facilitant ainsi une collaboration fluide et efficace entre plusieurs contributeurs.
- **GitLab** : Une plateforme de gestion de dépôts basée sur Git, qui simplifie le suivi et l'organisation de nos ressources tout en optimisant leur gestion.
- **Microsoft Teams** : Un outil collaboratif de communication qui renforce les interactions au sein de l'équipe, favorisant une coordination transparente et une communication améliorée.

- **ClickUp** : Une solution open source de gestion de projet, essentielle pour planifier, suivre et gérer les tâches, contribuant ainsi à une administration efficace et structurée de l'ensemble du projet. La figure 1.4 présente l'interface de ClickUp.

Name	Assignee	Due date	Priority	Comments	Date de tache
ajout de recherche et sort	SA				Feb 11
Reimplémentation du process congé	SA				Feb 12
Ajout interfaces user liste congé et demand...	SA				Feb 17
Ajout de la traduction	SA				Feb 18
ajout interface user congé	SA				Feb 20
ajout liste demande user	SA				Feb 20
ajout autorisation	SA				Feb 24
implementation autorisation front	SA				Feb 26
Ajout calendrier d'équipe	SA				Mar 3
Integration camunda modeler dans NG	SA				Mar 6
Ajout de notification avec webSocket	SA				Mar 7
Correction RBAC for Modeler and rh Confirm	SA				Mar 10
Correction des retours de qqe api backend	SA				Mar 11
Preparation présentation pour restitution e...	SA				Mar 12
Faire des recherches sur l'architecture de d...	SA				Mar 13
Début redaction du rapport	SA				Mar 14
Optimisation du code	SA				Mar 17
redaction rapport	SA				Mar 18
recherches sur argoCD	SA				Mar 21
Conception et rapport	SA				Mar 24
Ajout de mise à jour nom et synchronisation	SA				Mar 25

Figure 1.4: Interface de ClickUp

Conclusion

Dans ce premier chapitre, nous avons introduit l'organisme d'accueil tout en offrant un aperçu général de la problématique et des objectifs du projet.

Nous avons également décrit la méthodologie retenue ainsi que le formalisme encadrant le processus mis en œuvre pour le développement de notre application.

Le chapitre suivant se concentrera sur l'analyse de la solution existante, en mettant en lumière ses limites et en proposant une nouvelle solution à concevoir.

Chapitre 2

Sprint 0 : Analyse et Spécification des besoins

2.1 Introduction

Ce chapitre est consacré à l'analyse des besoins pour le développement du Portail de Gestion des Approbations. Nous y identifions les exigences fonctionnelles et non fonctionnelles du système, puis nous illustrons la modélisation à l'aide de diagrammes UML tels que les cas d'utilisation et le diagramme de classes.

Nous présentons ensuite le Backlog produit défini selon la méthodologie Scrum, avant de conclure par une présentation des principaux outils utilisés dans cette phase.

2.2 Capture des besoins

Dans cette section nous présentons les acteurs du système, les besoins fonctionnels et non fonctionnels présents dans le projet.

2.2.1 Identification des acteurs

Notre système contient quatre acteurs ,illustrés dans la figure 2.1, qui sont :

- **Administrateur** : Gère l'ensemble du système, incluant les utilisateurs et les configurations globales (e.g., permissions, intégrations).
- **Manager** : Soumet, consulte et traite les demandes (validation/rejet) des employés qu'il supervise.
- **RH (Ressources Humaines)** : Gère les congés, soumet, consulte et traite les demandes, tout en assurant le suivi des politiques RH.
- **Utilisateur** : Employé qui soumet des demandes (congés, absences) et consulte ses propres demandes et soldes de congés.

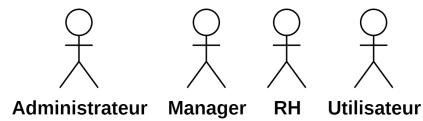


Figure 2.1: Les acteurs

2.2.2 Identification des besoins fonctionnels

Les besoins fonctionnels sont les interactions entre l'acteur(Une personne,un matériel ou un logiciel) et le système de manière à exploiter les fonctionnalités de ce dernier. Notre système offre aux acteurs la possibilité de faire certaines tâches.

Le tableau 2.1 illustre le rôle de chaque acteur dans ce système.

Tableau 2.1: tab:Les rôles des acteurs

Administrateur	<ul style="list-style-type: none"> - S'authentifier. - Se déconnecter. - Gérer les utilisateurs. - Consulter les processus métiers. - Consulter les demandes d'approbations.
Utilisateur	<ul style="list-style-type: none"> - S'authentifier. - Se déconnecter. - Consulter ses demandes. - Consulter ses congés. - Soumettre une demande. - Recevoir une notifications. - Consulter les rapports. - Gérer son profil. - Consulter le calendrier d'équipe. - Consulter les membres d'équipe. - Consulter ses crédits.
Manager / RH	<ul style="list-style-type: none"> - S'authentifier. - Se déconnecter. - Consulter ses demandes. - Consulter ses congés. - Soumettre une demande. - Recevoir une notifications. - Consulter les rapports. - Gérer son profil. - Consulter le calendrier d'équipe. - Consulter les membres d'équipe. - Consulter ses crédits. - Consulter ses taches accomplies. - Traiter une Demande.

2.2.3 Identification des cas d'utilisation

- **S'authentifier:** L'authentification est le cas d'utilisation qui permet à tous les utilisateurs (Administrateur, Utilisateur, Manager, RH) d'accéder à la plateforme. Il contient deux champs à remplir, qui sont le nom d'utilisateur et le mot de passe, et un bouton de validation pour accéder aux fonctionnalités de la plateforme en cas de succès de l'authentification.
- **Se déconnecter:** Les utilisateurs (Administrateur, Utilisateur, Manager, RH) peuvent se déconnecter du système en cliquant sur le bouton "Déconnexion". Le système vérifie l'existence et la conformité des coordonnées de l'utilisateur avant de le quitter.
- **Gérer les utilisateurs:** À travers ce cas d'utilisation, l'Administrateur est le seul à avoir le droit de gérer les utilisateurs de la plateforme, en incluant l'ajout, la consultation, la modification et la suppression des utilisateurs
- **Consulter les processus métiers:** L'Administrateur a la possibilité de consulter les différents processus métiers définis dans le système pour mieux comprendre les opérations internes et leur suivi.
- **Consulter les demandes d'approbation:** L'Administrateur peut accéder aux demandes d'approbation soumises pour les consulter.
- **Consulter ses demandes:** Les Utilisateurs, les Managers et les RH peuvent consulter l'historique de leurs demandes dans la plateforme.
- **Consulter ses congés:** Les utilisateurs, les Managers et les RH peuvent consulter la liste de leurs congés, passés et à venir, ainsi que leur solde restant.
- **Soumettre une demande:** Les Utilisateurs, les Managers et les RH peuvent soumettre une nouvelle demande via la plateforme. Cette demande peut concerner des congés, des absences ou d'autres requêtes spécifiques.
- **Recevoir une notification:** Ce cas permet aux Utilisateurs, Managers et aux RH de recevoir des notifications concernant les actions importantes, comme la validation d'une demande, ou des rappels de tâches.
- **Consulter les rapports:** Les Utilisateurs, les Managers et les RH peuvent consulter les rapports concernant leurs activités, comme les demandes soumises, les congés, les tâches accomplies, etc.
- **Gérer son profil:** Ce cas permet à chaque acteur (Utilisateur, Manager, RH) de gérer son profil personnel en modifiant ses informations comme son mot de passe, son adresse e-mail, etc.
- **Consulter le calendrier d'équipe:** Ce cas permet à l'Utilisateur, au Manager et au RH de consulter le calendrier de leur équipe, afin de voir les absences, les congés et les événements à venir.

- **Consulter les membres de l'équipe:** Les Utilisateurs, les Managers et les RH peuvent consulter la liste des membres de leur équipe, leurs informations et les tâches qui leur sont attribuées.
- **Consulter ses crédits:** Ce cas permet aux Utilisateurs, aux Managers et aux RH de consulter le solde de leurs crédits de congé ou autres types de crédits sur la plateforme.
- **Consulter ses tâches accomplies:** Les Managers et les RH peuvent consulter les tâches accomplies par ses subordonnés, pour le suivi des activités et des performances.
- **Traiter une demande:** Le Manager et l'RH sont en charge de traiter les demandes soumises, en les validant ou en les rejetant, selon les critères définis dans la plateforme.

2.2.4 Identification des besoins non fonctionnels

Les besoins non fonctionnels décrivent les contraintes et les exigences de qualité que doit respecter le système. Ils garantissent la performance, la fiabilité et la sécurité de l'application. Voici les principaux besoins non fonctionnels identifiés pour notre projet :

- **Performance :** Le système doit offrir une bonne réactivité, avec un temps de réponse inférieur à 2 secondes pour l'affichage des principales fonctionnalités. Il doit également supporter un nombre élevé de connexions simultanées sans dégradation notable des performances.
- **Sécurité :** Le système doit garantir la confidentialité, l'intégrité et la disponibilité des données. L'authentification des utilisateurs doit être sécurisée, et les accès doivent être strictement contrôlés selon le rôle de chaque utilisateur.
- **Fiabilité :** Le système doit être capable de fonctionner de manière continue sans erreurs critiques, avec un taux de disponibilité supérieur à 99.5%.
- **Scalabilité :** Le système doit pouvoir évoluer facilement pour prendre en charge un plus grand nombre d'utilisateurs ou de données sans refonte majeure de l'architecture.
- **Traçabilité :** Le système doit enregistrer les actions importantes effectuées par les utilisateurs (audit logs) pour permettre un suivi en cas d'incident ou d'investigation.

2.3 Diagramme de cas d'utilisation globale

Le diagramme de cas d'utilisation globale illustré dans la figure 2.2 consiste à décrire les fonctions fondamentales et identifie , également la relation et les interactions entre le système et ses acteurs.

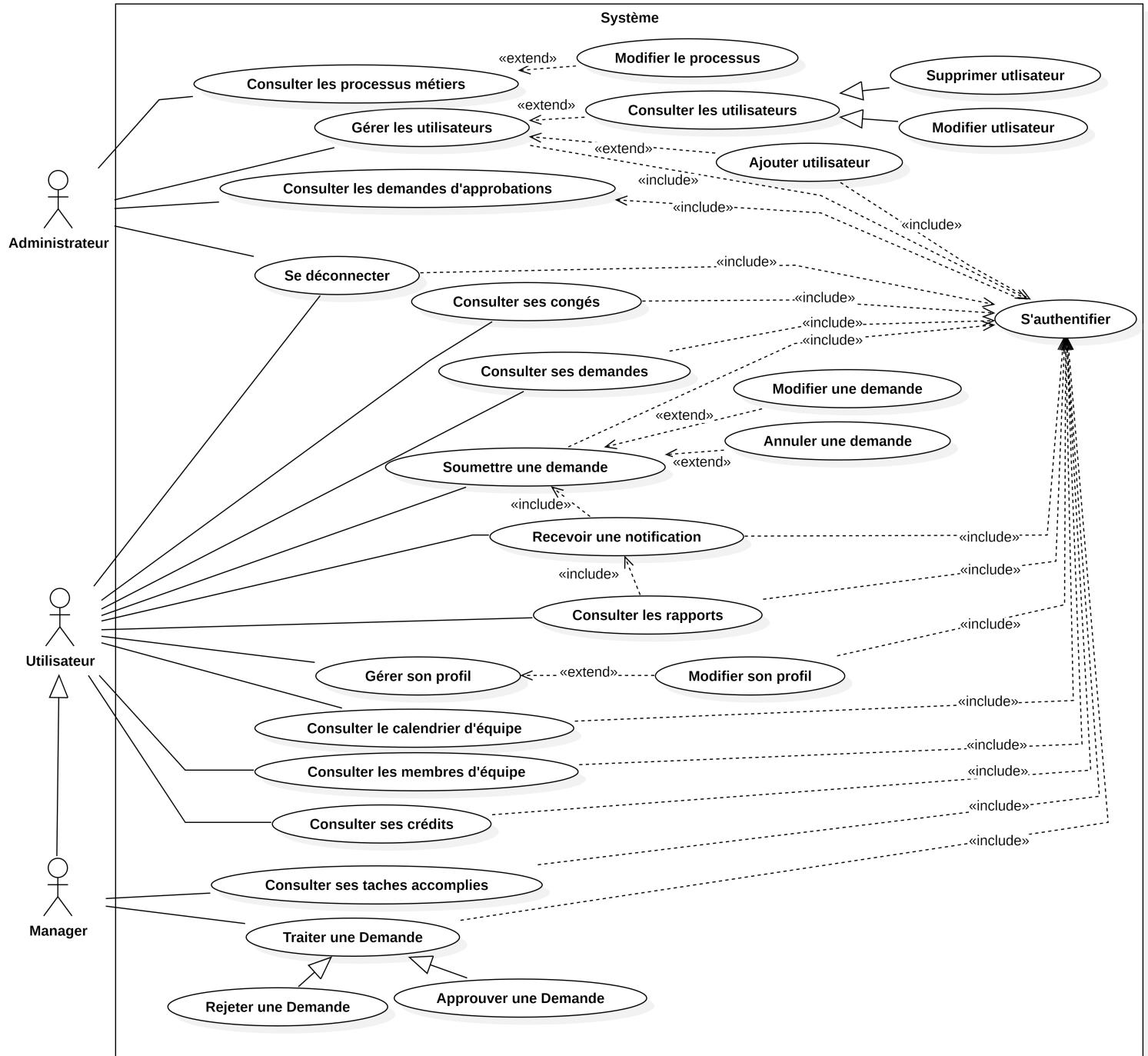


Figure 2.2: Diagramme de cas d'utilisation globale

2.4 Diagramme de classe global

Le diagramme de classe globale illustré dans la figure 2.3 montre la constitution du système à l'aide de la modélisation de ses classes, ses attributs, ses opérations et l'identification des relations entre ses objets.

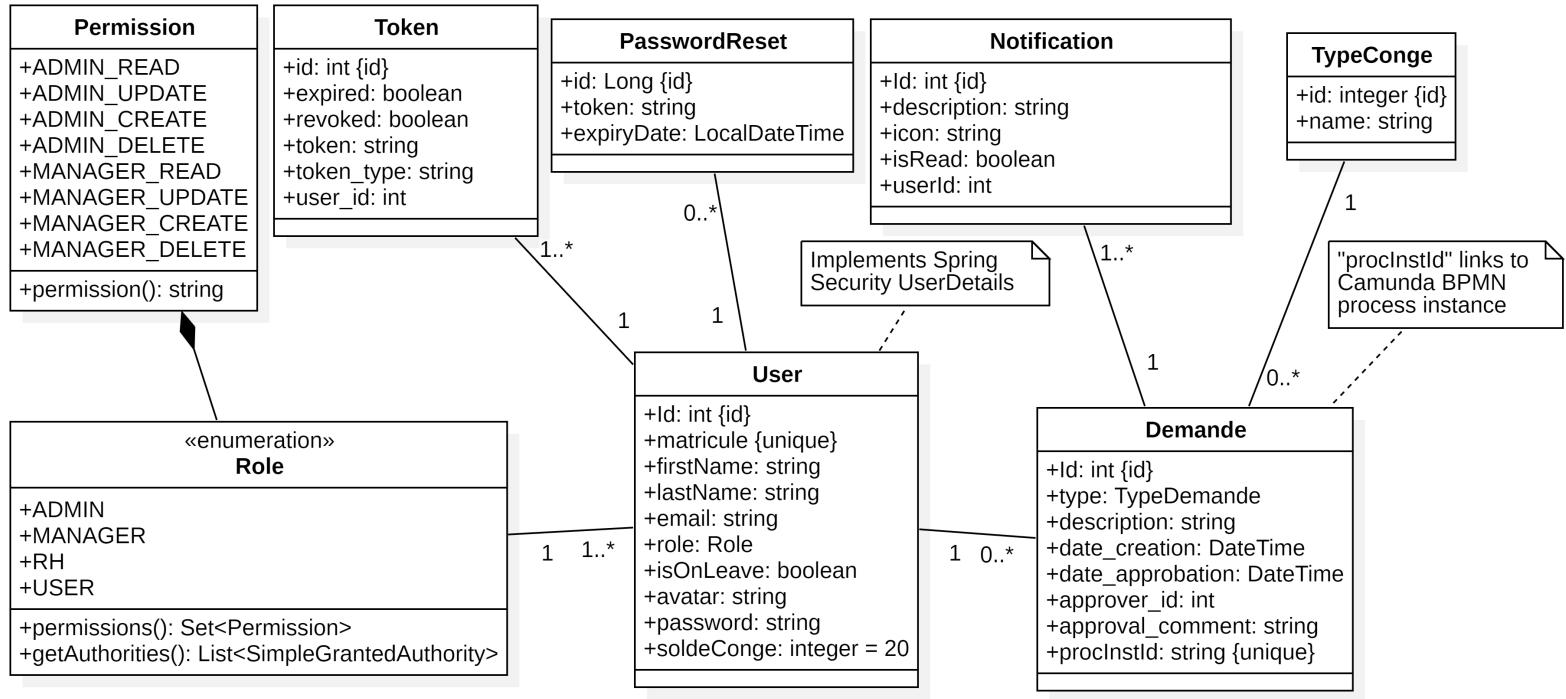


Figure 2.3: Diagramme de classe globale

2.5 Backlog du Produit

Le backlog du produit est un ensemble de tâches priorisées qui caractérise un produit. C'est l'un des éléments indispensables de la méthodologie Scrum. C'est l'outil de travail du Product Owner. Ce qui nous mène à le modéliser dans le tableau 2.2 et 2.3 []:

Sprint	User Story	Description	Priorité
Sprint 1 : Accès et Administration de Base	<ul style="list-style-type: none"> • S'authentifier • Se déconnecter • Gérer les utilisateurs • Mise en place et configuration de Camunda 	<ul style="list-style-type: none"> • Permet à tous les utilisateurs d'accéder à la plateforme via nom d'utilisateur et mot de passe. • Permet aux utilisateurs de quitter le système après vérification des coordonnées. • L'Administrateur peut ajouter, consulter, modifier et supprimer des utilisateurs. • Permet de configurer et d'intégrer le moteur Camunda dans l'application Spring Boot pour orchestrer les processus métiers. 	1 1 1 1
Sprint 2 : Gestion des Demandes	<ul style="list-style-type: none"> • Gérer son profil • Réinitialiser son mot de passe • Soumettre une demande • Consulter ses demandes • Consulter ses congés • Traiter une demande 	<ul style="list-style-type: none"> • Chaque acteur modifie ses informations personnelles (mot de passe, e-mail, etc.). • En cas d'oubli, l'utilisateur peut réinitialiser son mot de passe via un lien sécurisé. • Les Utilisateurs, Managers et RH soumettent des demandes (congés, absences, etc.). • Les Utilisateurs, Managers et RH consultent l'historique de leurs demandes. • Les Utilisateurs, Managers et RH consultent leurs congés et leur solde. • Les Managers et RH valident ou rejettent les demandes soumises. 	2 2 2 2 2 2
Sprint 3 : Suivi et Supervision	<ul style="list-style-type: none"> • Recevoir une notification • Consulter les processus métiers • Consulter les demandes d'approbation • Consulter les membres de l'équipe • Consulter ses crédits 	<ul style="list-style-type: none"> • Les Utilisateurs, Managers et RH reçoivent des alertes sur des actions importantes. • L'Administrateur consulte les processus métiers définis dans le système. • L'Administrateur accède aux demandes soumises pour consultation. • Les Utilisateurs, Managers et RH accèdent aux infos des membres de leur équipe. • Les Utilisateurs, Managers et RH vérifient leur solde de congés ou crédits. 	2 3 3 3 2

Tableau 2.2: Backlog du Produit - Partie 1

Sprint	User Story	Description	Priorité
Sprint 4 : Analyse et Améliorations	<ul style="list-style-type: none"> • Consulter le calendrier d'équipe • Consulter les tâches accomplies • Consulter les rapports • Exporter les rapports • Implémentation d'un chatbot 	<ul style="list-style-type: none"> • Les Utilisateurs, Managers et RH consultent les absences et événements d'équipe. • Les Managers et RH suivent les tâches réalisées par leurs subordonnés. • Les Utilisateurs, Managers et RH consultent des rapports sur leurs activités. • Permet d'exporter les rapports (PDF, Excel) pour une utilisation hors ligne. • Permet de développer et d'intégrer un chatbot dans l'application pour assister les utilisateurs dans leurs interactions (consultation, soumission de demandes, etc.). 	3 3 4 4 4
Sprint 5 : Pipeline DevOps et Gestion GitOps	<ul style="list-style-type: none"> • Installation Automatisée • Workflows CI • Runner SonarQube • Gestion des Secrets • Exposition des Services • Surveillance et Roll-back 	<ul style="list-style-type: none"> • Créer un playbook Ansible pour installer MicroK8s, ArgoCD et leurs dépendances, incluant les add-ons Ingress et DNS, pour un déploiement rapide et fiable et configurer ArgoCD pour synchroniser automatiquement avec le dépôt k8s-manifests et gérer les déploiements. • Implémenter des workflows GitHub Actions pour construire et pousser les images Docker vers DockerHub, avec mise à jour des tags dans k8s-manifests. • Configurer un runner auto-hébergé pour exécuter SonarScanner et analyser la qualité du code dans l'instance SonarQube locale. • Encoder les variables sensibles (identifiants DB, tokens API) en Base64 et les gérer via des secrets Kubernetes. • Déployer des règles Ingress pour exposer les services frontend, backend et SonarQube. • Vérifier la surveillance automatique des déploiements et le rollback natif d'ArgoCD en cas d'échec d'un nouveau pod. 	4 5 5 5 6 6

Tableau 2.3: Backlog du Produit - Partie 2

2.6 Planification de projet

Cette section a pour objectif de présenter la planification de notre projet. Pour ce faire, nous utiliserons un diagramme de Gantt qui permettra de visualiser les différentes étapes du projet sous forme de tâches à réaliser, facilitant ainsi le suivi et l'organisation du travail.

2.6.1 Diagramme de Gantt

Le diagramme de Gantt présenté par la figure 2.4 nous permet de visualiser les différentes tâches ainsi que leurs durées.

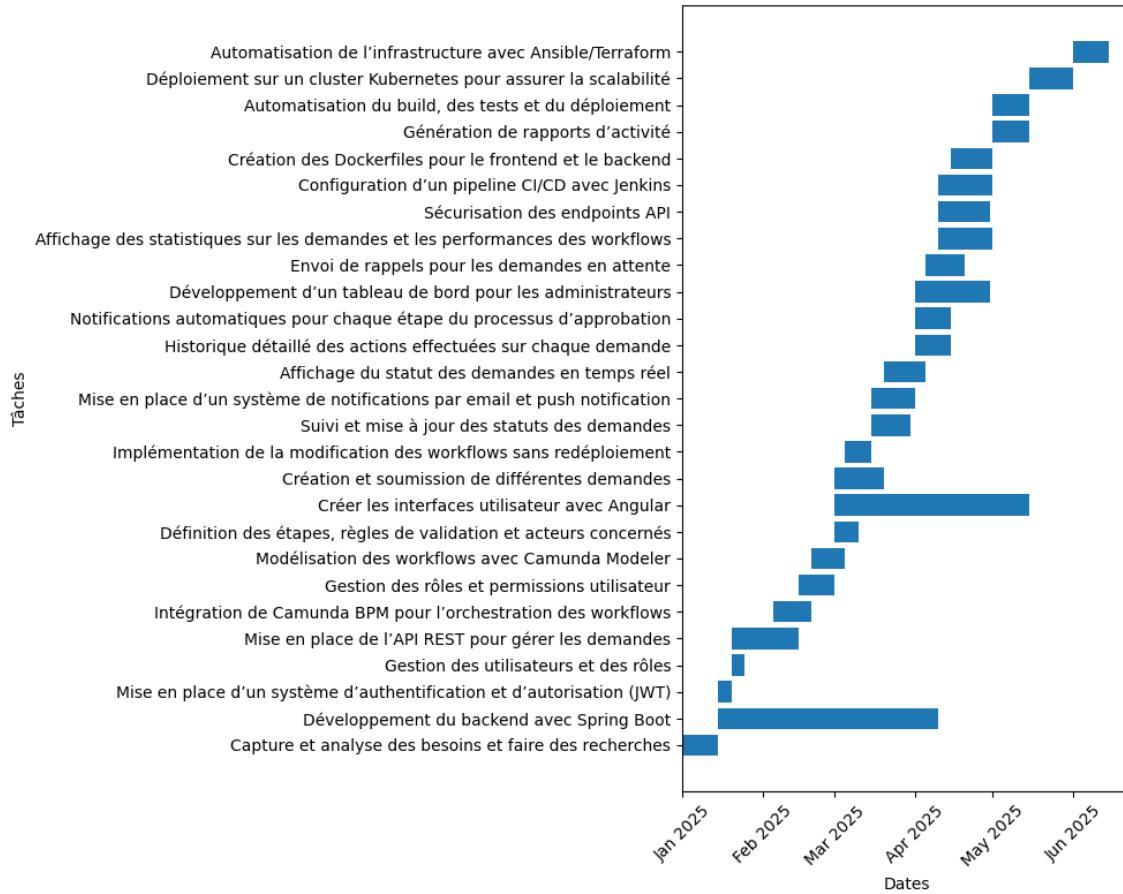


Figure 2.4: Diagramme de Gantt

2.7 Environnement de travail

Dans cette section, nous examinerons les différentes outils technologiques adoptées dans le cadre de ce projet.

2.7.1 Environnement matériel

Durant ce projet,nous avons utilisé deux machines pour bien mener notre projet.Les caractéristiques techniques de ces machines sont présentés dans le tableau 2.4.

Tableau 2.4: Spécifications des machines utilisées

	Machine 1	Machine 2
Marque	Gigabyte AERO 15	Machine virtuelle
Système d'exploitation	Windows 11 64 bit	Ubuntu 22.04 64 bit
Processeur	I7-11800H 2.3 GHz	I5-11400F 2.6 GHz
RAM	16 GB 3200MHz	16 GB 3200MHz
Disque Dur	1 TB	500 GB

2.7.2 Environnement logiciel

Dans cette section, nous présentons l'environnement logiciel utilisé dans ce projet :

- **Git**:Dont le logo est présenté dans la figure 2.5, cet outil constitue un système de gestion de versions distribué. Il permet de tracer l'évolution des fichiers et des répertoires d'un projet, d'accéder à des versions précédentes et de consulter en détail l'historique des modifications réalisées.



Figure 2.5: Logo de Git

- **Github**:Dont le logo est exposé dans la figure 2.6, est une plateforme collaborative basée sur Git, utilisée pour héberger du code, suivre les changements et faciliter le travail d'équipe sur des projets de développement logiciel.



Figure 2.6: Logo de Github

- **WampServer:** Dont le logo est illustré dans la figure 2.7, WampServer est un environnement de développement web local pour Windows. Il regroupe Apache, MySQL et PHP, permettant aux développeurs de tester et d'exécuter des applications web en local avant leur mise en production.



Figure 2.7: Logo de WampServer

- **phpMyAdmin:** Dont le logo est représenté dans la figure 2.8, phpMyAdmin est une interface web permettant d'administrer facilement des bases de données MySQL ou MariaDB. Il offre aux utilisateurs un accès simplifié pour gérer les tables, exécuter des requêtes SQL, importer ou exporter des données, le tout sans avoir à utiliser la ligne de commande.



Figure 2.8: Logo de phpMyAdmin

- **Visual Studio Code (VSCode):** Représenté dans la figure 2.9, VSCode est un éditeur de code source léger et extensible, populaire pour le développement Angular. Il offre une riche palette d'extensions, telles que des outils de linting, de débogage, et des intégrations Git, ce qui le rend idéal pour travailler sur des applications JavaScript et TypeScript, comme celles développées avec Angular.



Figure 2.9: Logo de VSCode

- **IntelliJ IDEA:** Représenté dans la figure 2.10, IntelliJ IDEA est un environnement de développement intégré (IDE) robuste, principalement utilisé pour le développement d'applications Java, telles que celles créées avec Spring. Il offre des fonctionnalités avancées comme la gestion des dépendances, le débogage intégré, et la prise en charge complète de Spring, ce qui facilite le développement et le déploiement d'applications Spring Boot.



Figure 2.10: Logo de IntelliJ

2.8 Choix technologiques

2.8.1 Frontend

- **Angular:** Représenté dans la figure 2.11, Angular est un framework de développement web permettant de créer des applications web dynamiques et interactives.



Figure 2.11: Logo d'Angular

2.8.2 Backend

- **Spring Boot:** Représenté dans la figure 2.12, Spring Boot est un framework Java qui facilite le développement d'applications Spring en simplifiant la configuration et le déploiement, tout en offrant des outils prêts à l'emploi comme un serveur embarqué et une sécurité intégrée.



Figure 2.12: Logo de Spring Boot

2.8.3 Workflow

- **Camunda:** Dont le logo est représenté dans la figure 2.13, Camunda est une plateforme open-source de gestion des processus métier (BPMN), de gestion des décisions (DMN) et de gestion des cas (CMMN). Camunda est particulièrement utilisé pour l'orchestration des workflows complexes et s'intègre facilement dans des architectures Java et microservices.



Figure 2.13: Logo de Camunda

2.8.4 Base de données

- **MySQL:** Représenté dans la figure 2.14, MySQL est un système de gestion de bases de données relationnelles open-source. Il permet de stocker, gérer et interroger des données de manière structurée à l'aide de SQL.



Figure 2.14: Logo de MySQL

2.8.5 CI/CD

- **ArgoCD:** Représenté dans la figure 2.15, ArgoCD est un outil open-source d'intégration et de déploiement continu (CI/CD) spécialement conçu pour Kubernetes. Il permet de déployer et gérer des applications dans des environnements Kubernetes en suivant la méthodologie GitOps.



Figure 2.15: Logo d'ArgoCD

2.8.6 Conteneurisation

- **Docker:** Représenté dans la figure 2.16, Docker est une plateforme open-source permettant de créer, déployer et exécuter des applications dans des conteneurs. Ces conteneurs encapsulent l'application et ses dépendances dans un environnement léger et portable.

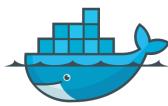


Figure 2.16: Logo de Docker

2.8.7 Orchestration

- **Kubernetes:** Représenté dans la figure 2.17, Kubernetes est un système open-source d'orchestration de conteneurs qui automatise le déploiement, la gestion, la mise à l'échelle et l'administration des applications conteneurisées. Il permet de gérer efficacement les clusters de conteneurs Docker, offrant des fonctionnalités telles que l'autoscaling, la gestion des ressources et le monitoring des applications.



Figure 2.17: Logo de Kubernetes

Conclusion

Dans ce deuxième chapitre, nous avons détaillé les besoins fonctionnels et non fonctionnels du projet, ainsi que les différents acteurs et leurs interactions avec le système à travers des diagrammes de cas d'utilisation et de classes globaux. Nous avons également présenté le backlog du produit, structuré en sprints, et défini les priorités pour chaque fonctionnalité. Nous avons également présenté les choix technologiques retenus pour le développement du projet, couvrant l'environnement matériel, l'environnement logiciel. Le chapitre suivant sera consacré à la présentation du Sprint 1, qui couvrira les premières étapes du développement, notamment l'accès et l'administration de base du système.

Chapitre 3

Sprint 1 : Accès et administration de base

3.1 Introduction

Après avoir tracer les grandes lignes de notre projet, concentrons-nous maintenant sur le premier sprint. Dans ce qui suit, nous expliquerons chaque fonctionnalité de ce sprint en detaillant ses différents besoins, ainsi que sa conception .

3.2 Backlog de Sprint 1

Le tableau 3.1 représente le backlog du premier sprint. Ce tableau détaille les cas d'utilisation, leurs priorités, estimations et tâches associées.

Tableau 3.1: Backlog du Sprint 1

Cas d'utilisation	Priorité	Tâche
En tant qu'utilisateur ou admin, je peux m'authentifier	1	Authentifier l'utilisateur a l'aide du token JWT
En tant qu'utilisateur ou admin, je peux me déconnecter	1	Déconnecter l'utilisateur du session
En tant qu'admin, je peux gérer les utilisateurs	1	Ajouter, consulter, modifier, supprimer
En tant que développeur, je peux configurer et intégrer Camunda	1	Configurer dépendances Camunda

3.3 Raffinement des cas d'utilisation

Cette section raffine les cas d'utilisation du Sprint 1 en identifiant les acteurs et en détaillant chaque cas d'utilisation.

3.3.1 Identification des acteurs du premier sprint

Les acteurs de ce sprint sont :

Administrateur : Acteur principal, il peut s'authentifier, se déconnecter, et gérer les utilisateurs (ajouter, consulter, modifier, supprimer).

Utilisateur : Tout acteur ayant un compte, pouvant s'authentifier et se déconnecter.

Développeur : Acteur technique qui configure le moteur Camunda dans l'application pour activer les workflows.

3.3.2 Raffinement du cas d'utilisation «S'authentifier»

La figure 3.1 illustre le diagramme de cas d'utilisation « S'authentifier ».

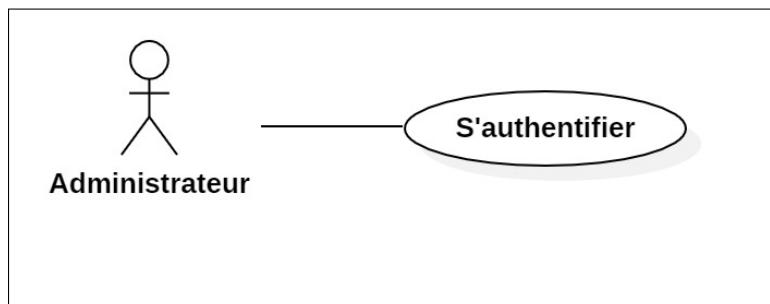


Figure 3.1: Diagramme du cas d'utilisation «S'authentifier»

Le tableau 3.2 illustre la description détaillée du cas d'utilisation ‘s’authentifier’.

Tableau 3.2: Description textuelle du cas d'utilisation «S'authentifier»

Cas d'utilisation	S'authentifier
Acteur	Utilisateur
Pré-conditions	Système en marche, utilisateur inscrit
Post-conditions	Utilisateur authentifié, redirigé selon son rôle
Scénario de base	<ol style="list-style-type: none">1. Affichage de l'interface de connexion2. Saisie de l'email et du mot de passe3. Clic sur « Se connecter »4. Vérification des identifiants5. Redirection vers l'accueil
Exceptions	<ul style="list-style-type: none">- Erreur si identifiants incorrects- Redirection vers l'authentification si le token JWT est expiré

3.3.3 Raffinement du cas d'utilisation «Se déconnecter»

La figure 3.2 illustre le diagramme de cas d'utilisation « Se déconnecter ».

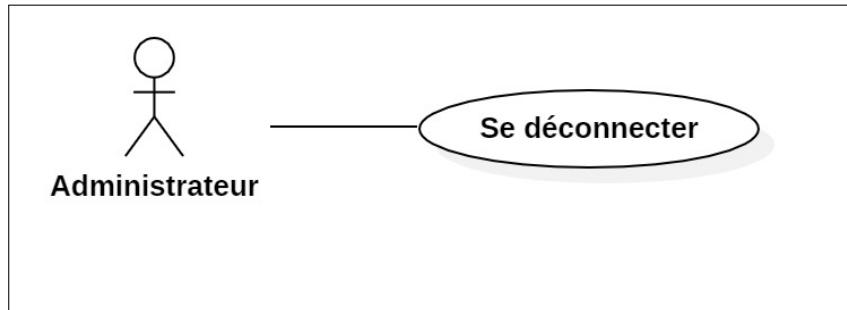


Figure 3.2: Diagramme du cas d'utilisation «Se déconnecter»

Le tableau 3.3 illustre la description détaillée du cas d'utilisation 'Se déconnecter'.

Tableau 3.3: Cas d'utilisation : Se déconnecter

Cas d'utilisation	Se déconnecter
Acteur	Utilisateur
Pré-conditions	Système en marche, utilisateur connecté
Post-conditions	Utilisateur déconnecté
Scénario de base	1. L'utilisateur clique sur « Se déconnecter » 2. Le système redirige vers l'interface d'authentification
Exceptions	- Échec de déconnection (erreur API)

3.3.4 Raffinement du cas d'utilisation «Gérer les utilisateurs»

La figure 3.3 illustre le diagramme de cas d'utilisation « Gérer les utilisateurs ».

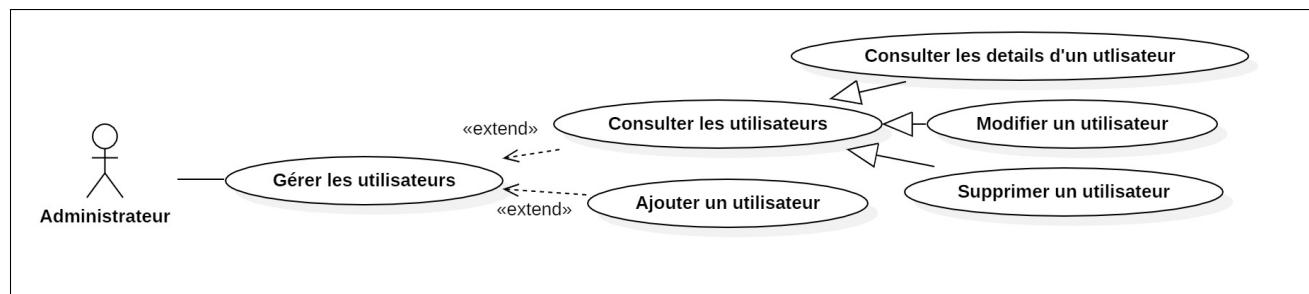


Figure 3.3: Diagramme du cas d'utilisation «Gérer les utilisateurs»

Le tableau 3.4 illustre la description détaillée du cas d'utilisation « Ajouter un utilisateur ».

Tableau 3.4: Description textuelle du cas d'utilisation « Ajouter un utilisateur »

Cas d'utilisation	Ajouter un utilisateur
Acteur	Admin
Pré-conditions	Le système est en marche. L'admin est authentifié.
Post-conditions	L'utilisateur est ajouté.
Scénario de base	<ol style="list-style-type: none"> 1. Le système affiche l'interface d'ajout. 2. L'admin saisit les données du nouvel utilisateur (nom, email, rôle, etc.). 3. L'admin clique sur le bouton « Ajouter ». 4. Le système enregistre les informations. 5. Le système affiche un message de succès.
Exceptions	Échec d'ajout (entrées invalides, problème avec l'API POST, ou erreur de base de données).

Le tableau 3.5 illustre la description détaillée du cas d'utilisation « Consulter les utilisateurs ».

Tableau 3.5: Description textuelle du cas d'utilisation « Consulter les utilisateurs »

Cas d'utilisation	Consulter les utilisateurs
Acteur	Admin
Pré-conditions	Le système est en marche. L'admin est authentifié.
Post-conditions	La liste des utilisateurs est consultée.
Scénario de base	<ol style="list-style-type: none"> 1. Le système affiche la liste des utilisateurs. 2. L'admin consulte la liste.
Exceptions	Échec de consultation (problème avec l'API GET, ou erreur de base de données).
Extensions	Modifier un utilisateur. Supprimer un utilisateur.

Le tableau 3.6 illustre la description détaillée du cas d'utilisation « Consulter les détails d'un utilisateur ».

Tableau 3.6: Description textuelle du cas d'utilisation « Consulter les détails d'un utilisateur »

Cas d'utilisation	Consulter les détails d'un utilisateur
Acteur	Admin
Pré-conditions	Le système est en marche. L'admin est authentifié.
Post-conditions	Les informations détaillées de l'utilisateur sont affichées.
Scénario de base	<ol style="list-style-type: none"> L'admin clique sur le bouton pour afficher les détails d'un utilisateur depuis la liste. Le système affiche les informations détaillées de l'utilisateur (nom, email, rôle, date de création, etc.).
Exceptions	Échec de consultation (problème avec l'API GET, ou erreur de base de données).

Le tableau 3.7 illustre la description détaillée du cas d'utilisation « Modifier un utilisateur ».

Tableau 3.7: Description textuelle du cas d'utilisation « Modifier un utilisateur »

Cas d'utilisation	Modifier un utilisateur
Acteur	Admin
Pré-conditions	Le système est en marche. L'admin est authentifié.
Post-conditions	L'utilisateur est modifié.
Scénario de base	<ol style="list-style-type: none"> Le système affiche l'interface de modification. L'admin saisit les nouvelles données. L'admin clique sur le bouton « Modifier ». Le système enregistre les modifications. Le système affiche un message de succès.
Exceptions	Échec de modification (problème avec l'API PUT, ou erreur de base de données).

Le tableau 3.8 illustre la description détaillée du cas d'utilisation « Supprimer un utilisateur ».

Tableau 3.8: Description textuelle du cas d'utilisation « Supprimer un utilisateur »

Cas d'utilisation	Supprimer un utilisateur
Acteur	Admin
Pré-conditions	Le système est en marche. L'admin est authentifié.
Post-conditions	L'utilisateur est supprimé.
Scénario de base	<ol style="list-style-type: none"> 1. L'admin clique sur le bouton « Supprimer » d'un utilisateur. 2. Le système supprime les données correspondantes. 3. Le système affiche un message de succès.
Exceptions	Échec de suppression (problème avec l'API DELETE, ou erreur de base de données).

3.3.5 Raffinement du cas d'utilisation «Mise en place et configuration de Camunda»

Le tableau 3.9 illustre la description détaillée du cas d'utilisation « Mise en place et configuration de Camunda ».

Tableau 3.9: Description textuelle du cas d'utilisation «Mise en place et configuration de Camunda»

Cas d'utilisation	Mise en place et configuration de Camunda
Acteur	Développeur
Pré-conditions	Le projet backend (Spring Boot) est accessible. L'environnement de développement est configuré (Java, Maven, IDE).
Post-conditions	Le moteur Camunda est intégré, configuré et fonctionnel dans l'application Spring Boot.
Scénario de base	<ol style="list-style-type: none"> 1. Le développeur identifie la version de Camunda compatible avec la version de Spring Boot utilisée dans le projet. 2. Le développeur ajoute les dépendances Camunda nécessaires dans le fichier <code>pom.xml</code>. 3. Le développeur configure les propriétés Camunda dans le fichier <code>application.properties</code>. 4. Le développeur initialise et configure le moteur de workflow dans la classe de configuration Spring. 5. Le développeur crée un processus BPMN simple pour tester l'intégration. 6. Le système valide le bon fonctionnement du moteur.
Exceptions	Erreurs d'intégration : dépendances manquantes, mauvaise configuration, problèmes de base de données ou de compatibilité.

3.4 Conception

La conception est une phase importante pour la bonne réalisation d'un projet. Dans cette partie nous allons exposer la conception des cas d'utilisations de ce sprint qui se traduit par un diagramme de classe globale de ce sprint suivi par les diagrammes de classes et les diagrammes de séquences de chaque cas d'utilisation.

3.4.1 Diagramme de classe du sprint 1

La figure 3.4 illustre le diagramme de classe du Sprint 1. Il est formé de :

- **Classe "Permission"** : C'est une énumération qui définit les permissions des différents rôles (ADMIN, MANAGER, etc.).
- **Classe "Role"** : C'est une énumération qui permet d'identifier les rôles des utilisateurs de la plateforme.
- **Classe "Token"** : C'est la classe qui gère les jetons d'authentification des utilisateurs.
- **Classe "User"** : C'est la classe qui représente tous les utilisateurs ayant accès à la plate-forme.
- **Interface "Implements Spring Security UserDetailsService"** : Indique que la classe User implémente cette interface pour gérer l'authentification et l'autorisation.

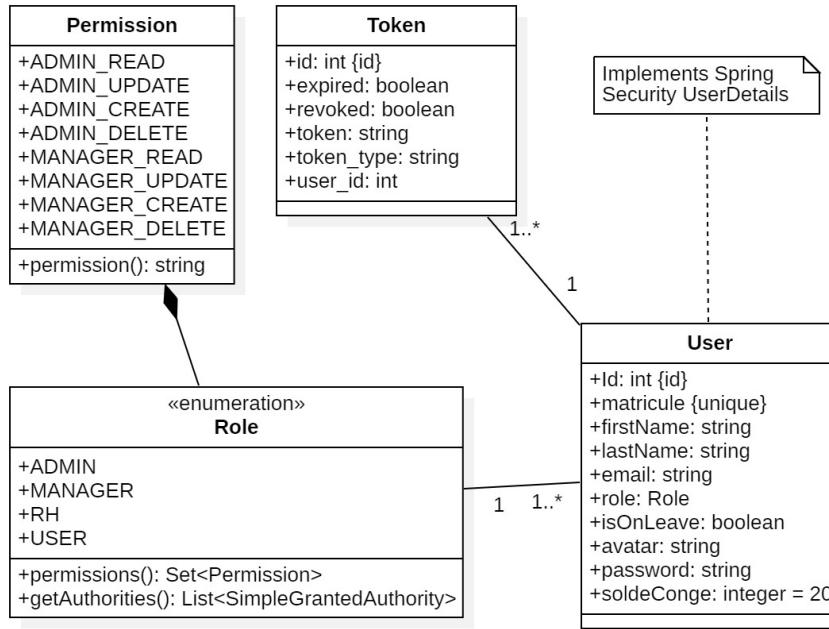


Figure 3.4: Diagramme de classe du sprint 1

3.4.2 Conception du cas d'utilisation «S'authentifier»

Diagramme de classe

La figure 3.5 illustre le diagramme de classes du cas d'utilisation s'authentifier.

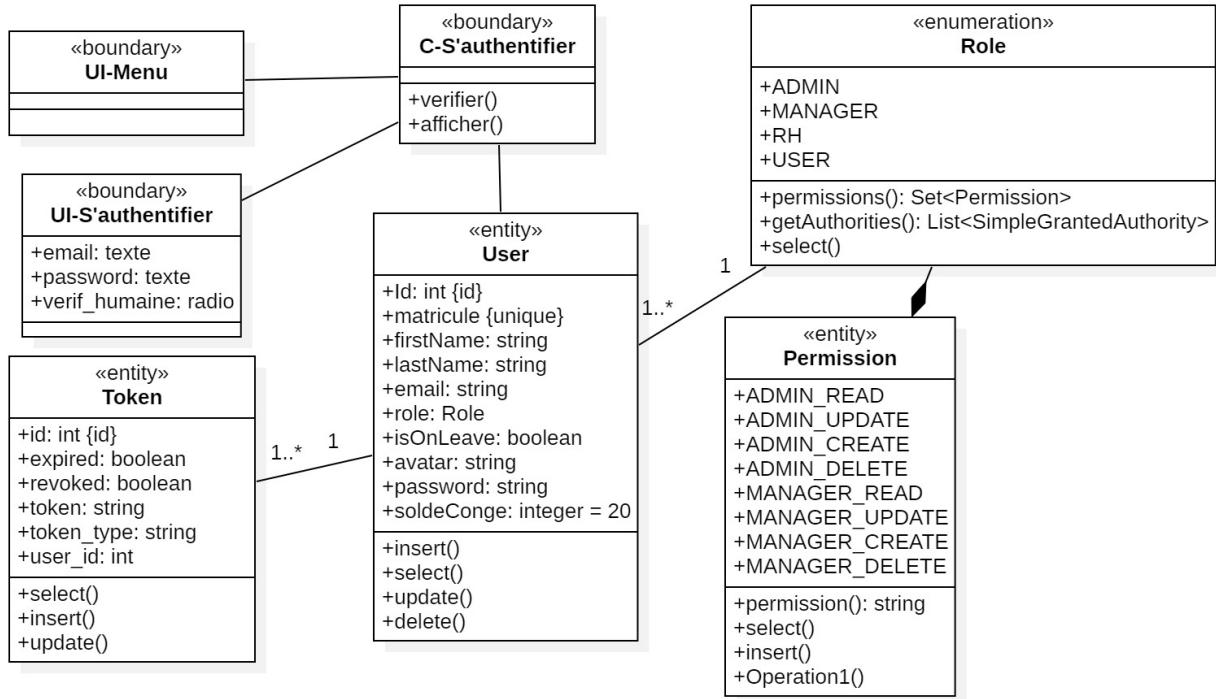


Figure 3.5: Diagramme de classe du cas d'utilisation «S'authentifier»

Diagramme de séquence

Au lancement de l'application, une interface d'authentification apparaîtra. Ensuite, les utilisateurs peuvent entrer leurs adresses mail et mots de passe, et vérifier qu'ils ne sont pas un robot (recaptcha), qui seront envoyés via l'interface d'authentification au contrôleur d'authentification, qui à son tour vérifie les informations dans l'entité *Utilisateur*.

Si les coordonnées sont correctes, ils seront amenés à un espace approprié selon leurs rôles. Sinon, un message d'erreur d'authentification sera affiché.

La figure 3.6 modélise le diagramme de séquence du cas d'utilisation « S'authentifier ».

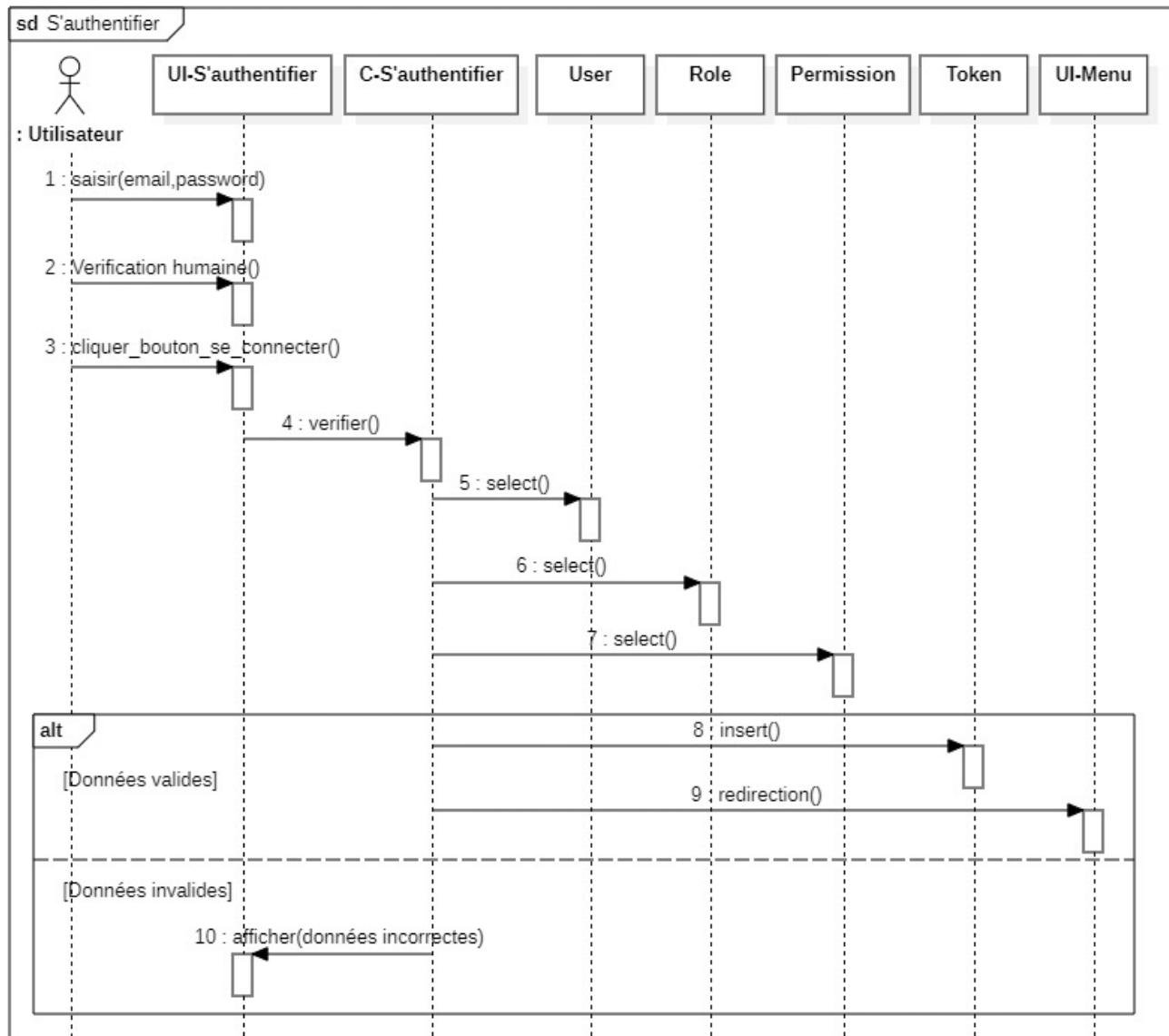


Figure 3.6: Diagramme de séquence du cas d'utilisation «S'authentifier»

3.4.3 Conception du cas d'utilisation «Se déconnecter»

Diagramme de classe

La figure 3.7 illustre le diagramme de classe du cas d'utilisation se déconnecter.

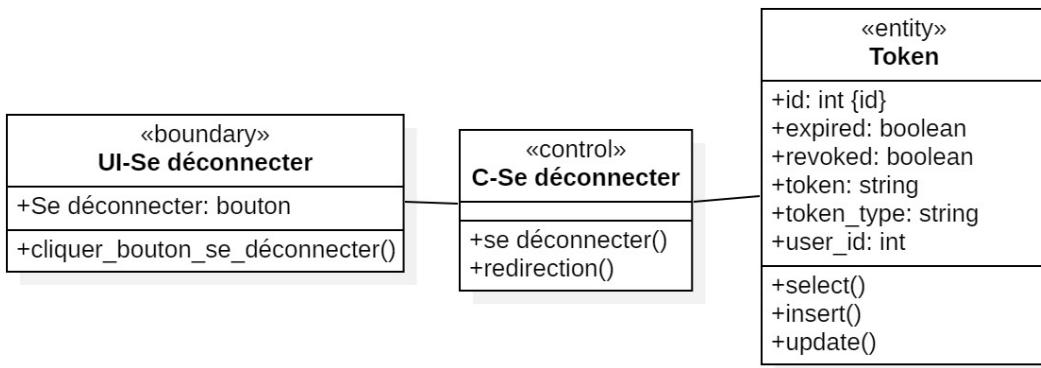


Figure 3.7: Diagramme de classe du cas d'utilisation «Se déconnecter»

Diagramme de séquence

Si l'utilisateur souhaite se déconnecter, il clique sur le bouton de déconnexion dans le menu. Le système supprime les données obtenues avec l'authentification de l'utilisateur et retire son token. Ce dernier sera finalement redirigé vers l'interface d'authentification.

La figure 3.8 illustre le diagramme de séquence du cas d'utilisation se déconnecter.

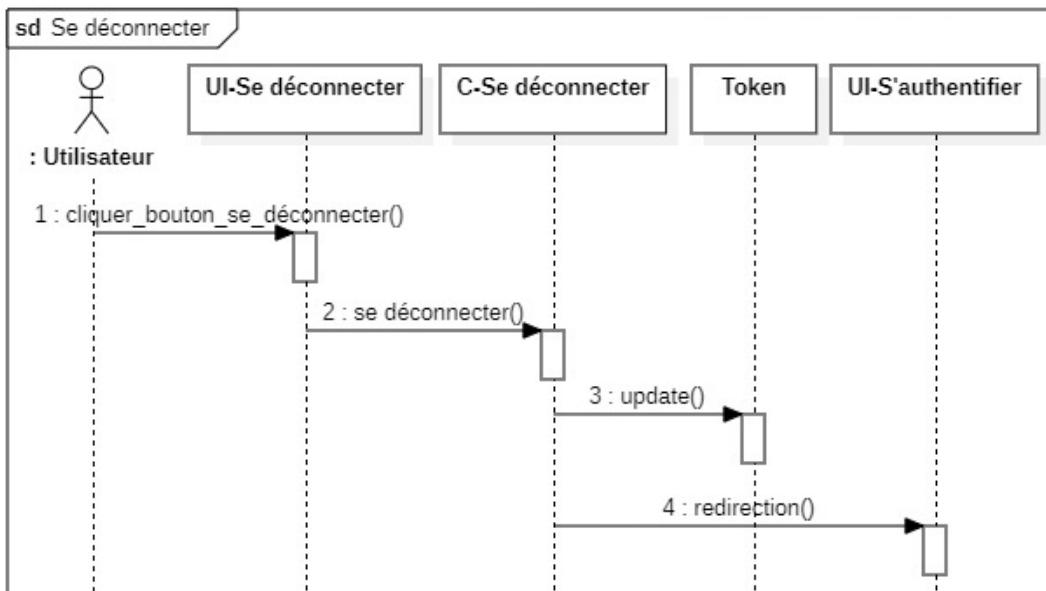


Figure 3.8: Diagramme de séquence du cas d'utilisation «Se déconnecter»

3.4.4 Conception du cas d'utilisation «Gérer les utilisateurs»

Diagramme de classe

La figure 3.9 illustre le diagramme de classe du cas d'utilisation gérer les utilisateurs.

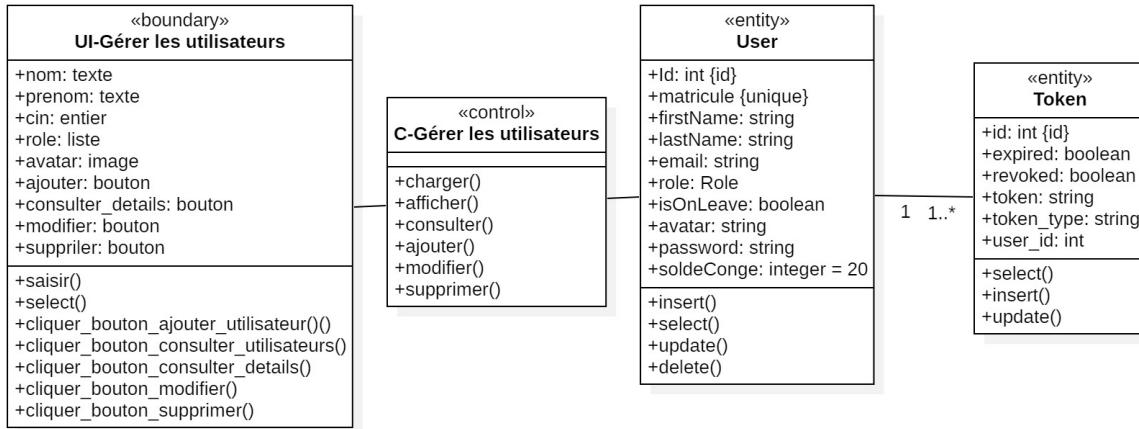


Figure 3.9: Diagramme de classe du cas d'utilisation «Gérer les utilisateurs»

Diagramme de séquence

Seul l'administrateur dispose des droits nécessaires pour gérer les utilisateurs depuis son interface dédiée. Cette gestion inclut un ensemble d'opérations essentielles telles que l'ajout de nouveaux utilisateurs, la consultation de la liste des utilisateurs existants, l'affichage des détails d'un utilisateur spécifique, la modification de ses informations, ainsi que la suppression d'un compte utilisateur. Toutes ces actions sont déclenchées depuis l'interface d'administration, puis transmises au contrôleur via des requêtes HTTP. Ce dernier assure la logique métier en interagissant avec les entités du système, notamment l'entité "User" pour manipuler les données des utilisateurs, et l'entité "Token" pour révoquer les tokens en cas de suppression d'un utilisateur.

La figure 3.10 illustre le diagramme de séquence du cas d'utilisation « Gérer les utilisateurs ».

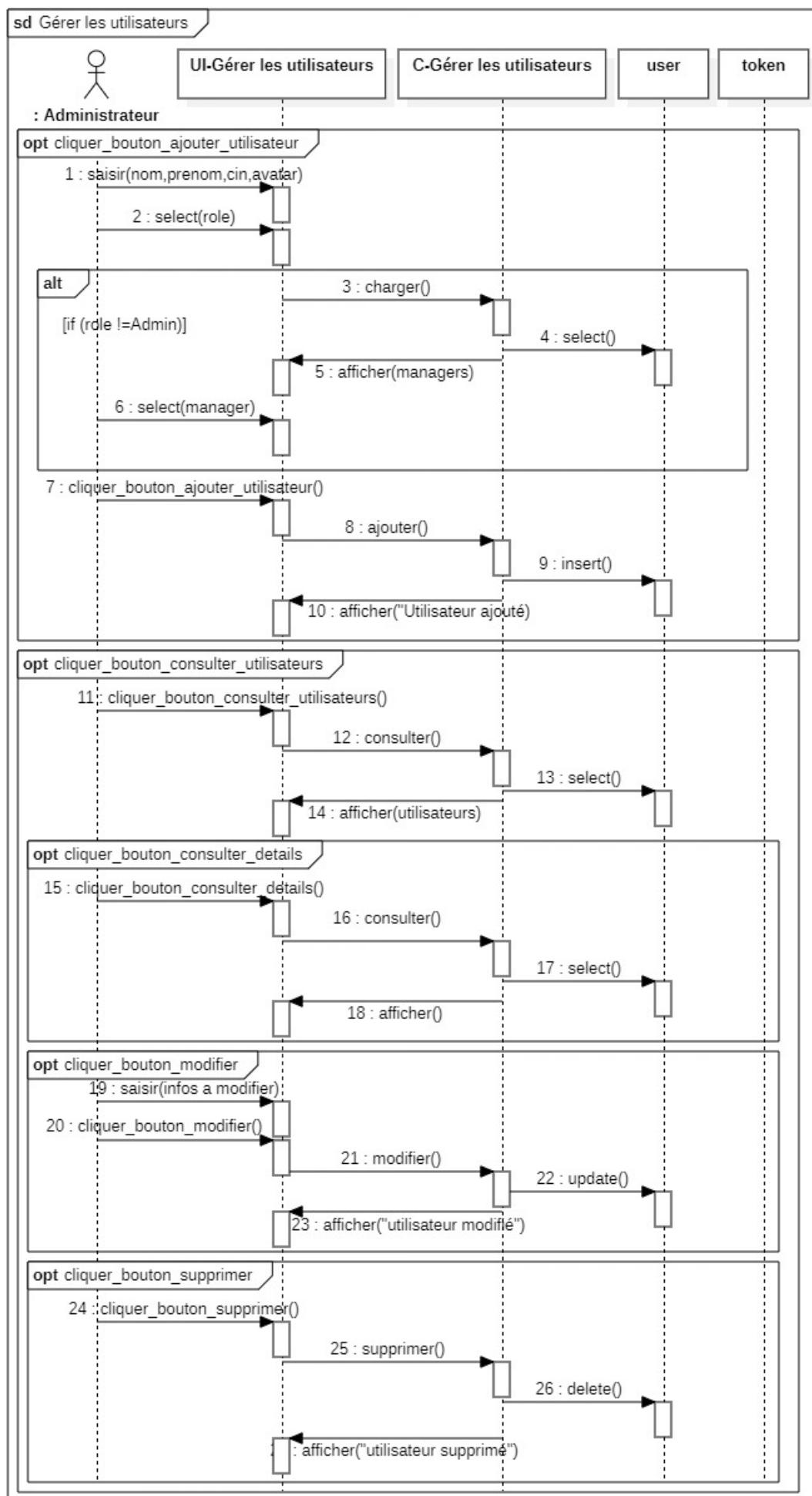


Figure 3.10: Diagramme de séquence du cas d'utilisation «Gérer les utilisateurs»³⁸

3.5 Réalisation

Dans cette partie, nous présentons les modules de notre premier sprint en utilisant des captures d'écran.

3.5.1 S'authentifier

L'utilisateur n'est pas autorisé à accéder à l'application, sauf s'il a un compte. Donc, afin d'explorer les fonctionnalités de l'application, l'utilisateur est invité à s'authentifier avec un email et un mot de passe comme le montre la figure 3.11 ci-dessous. L'utilisateur ne peut pas créer un compte puisque les comptes sont attribués par un administrateur.

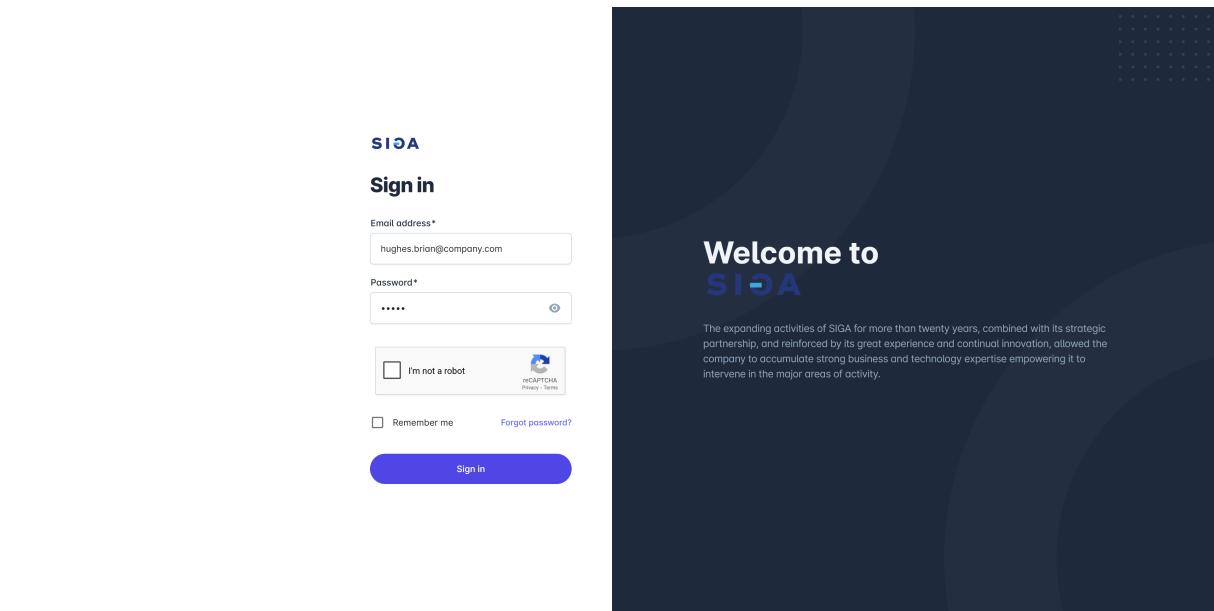


Figure 3.11: Interface du cas d'utilisation «S'authentifier»

3.5.2 Se déconnecter

L'utilisateur a la possibilité de se déconnecter de l'application lorsqu'il le souhaite, comme le montre la figure 3.12. Cette action met fin à sa session en cours et le redirige vers la page d'authentification.

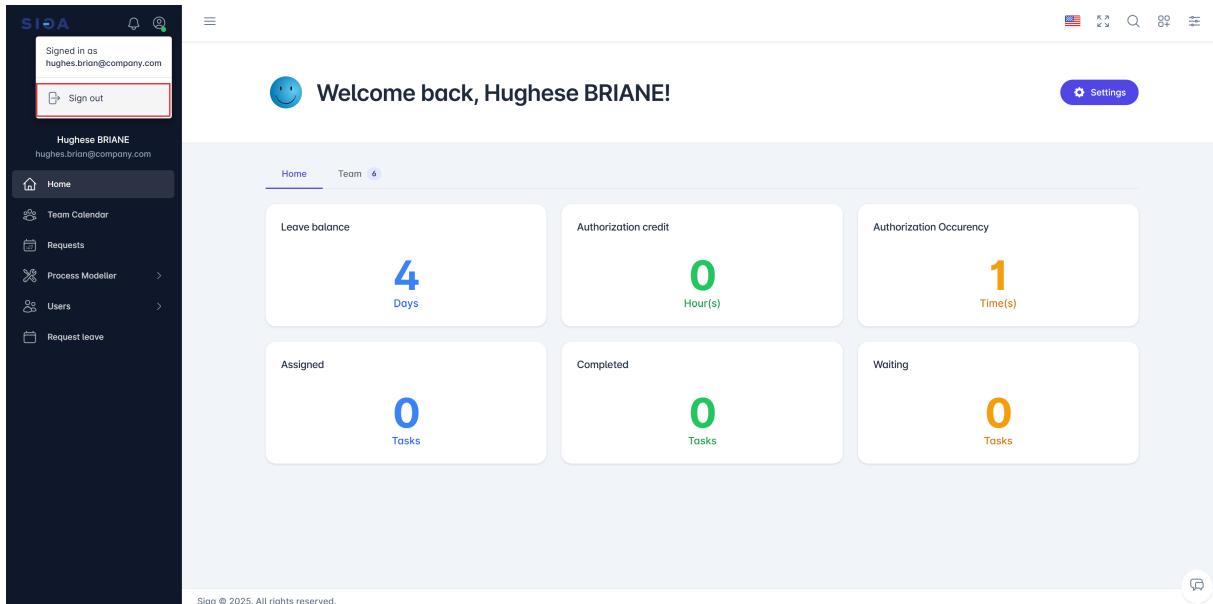


Figure 3.12: Interface du cas d'utilisation «Se déconnecter»

3.5.3 Gérer les utilisateurs

La gestion des utilisateurs permet à l'administrateur de consulter, ajouter, modifier, supprimer et consulter les détails des utilisateurs.

Ajouter un utilisateur

L'administrateur peut ajouter un nouvel utilisateur à l'application via l'interface dédiée accessible depuis le menu latéral, comme illustré dans la figure 3.13. Il suffit de renseigner les informations demandées telles que le prénom, le nom, l'adresse e-mail, le mot de passe, le rôle ainsi qu'un avatar, puis de cliquer sur Create account pour finaliser l'ajout.

Figure 3.13: Interface du cas d'utilisation «Ajouter un utilisateur»

Consulter les utilisateurs

L'administrateur peut consulter la liste des utilisateurs existants via l'onglet Users > List dans le menu latéral, comme le montre la figure 3.14. Cette interface permet d'avoir un aperçu global des comptes enregistrés, incluant leurs informations de profil et leur rôle dans le système.

Matricule	Firstname	Lastname	Email	Role	Leave balance	Manager Matricule	Details
2501EMP001	mourad	BenFlen	self.abidi@esprit.tn	USER	20	2501EMP002	
2501EMP002	iheb	Youfli	iheb@esprit.tn	MANAGER	23		
2501EMP003	Seifeddine	Abidi	seifeddine.abidi@esprit.tn	RH	12	2501EMP002	
2501EMP004	Ahmed	jaballah	imed.jaballah@esprit.tn	ADMIN	5	2501EMP002	
2501EMP005	Hughese	BRIANE	hughes.briane@company.com	ADMIN	4	2501EMP002	
2501EMP006	string	string	string	MANAGER	16	2501EMP002	
2502EMP001	imed	jaballah	self.briane@company.com	MANAGER	21	2501EMP002	

Figure 3.14: Interface du cas d'utilisation «Consulter les utilisateurs»

Consulter les détails d'un utilisateur

L'administrateur peut consulter les détails d'un compte spécifique en sélectionnant un utilisateur dans la liste, comme illustré dans la figure 3.15. Cette action affiche les informations complètes du profil sélectionné.

Matricule	Firstname	Lastname	Email	Role	Leave balance	Manager Matricule	Details
2501EMP001	mourad	BenFlen	self.abidi@esprit.tn	USER	20	2501EMP002	

Figure 3.15: Interface du cas d'utilisation «Consulter les détails d'un utilisateur»

Modifier un utilisateur

L'administrateur peut modifier les informations d'un compte existant en accédant à son profil depuis la liste des utilisateurs, comme le montre la figure 3.16. Une fois les champs mis à jour (nom, e-mail, rôle, etc.), il peut enregistrer les modifications pour qu'elles soient prises en compte immédiatement.

The screenshot shows the SIDA application's 'Users list' page. On the left, there is a sidebar with navigation links: Home, Team Calendar, Requests, Process Modeler, Users (selected), List, Add, and Request leave. The main area displays a table with columns: Matricule, Firstname, Lastname, Email, Role, Leave balance, Manager Matricule, and Details. One row is selected, showing details for a user named 'moured' with Matricule '2501EMP001'. The 'Details' section contains input fields for Firstname ('moured'), Lastname ('BenFlen'), Matricule ('2501EMP001'), Role ('USER'), Solde Conge ('20'), Manager Matricule ('2501EMP002'), and Email ('seif.abidi@esprit.tn'). At the bottom right of the table, there is a blue 'Update' button, which is highlighted with a red box.

Figure 3.16: Interface du cas d'utilisation «Modifier un utilisateur»

Supprimer un utilisateur

L'administrateur peut supprimer un compte utilisateur en accédant à la liste des utilisateurs et en sélectionnant l'option de suppression associée, comme illustré dans la figure 3.17. Une confirmation est demandée avant la suppression définitive afin d'éviter toute action involontaire.

The screenshot shows the SIDA application's 'Users list' page, similar to Figure 3.16. The sidebar and table structure are identical. The 'Details' section for the selected user 'moured' shows the same information as before. At the bottom left of the table, there is a red 'Delete' button, which is highlighted with a red box.

Figure 3.17: Interface du cas d'utilisation «Supprimer un utilisateur»

Intégrer Camunda

Afin d'intégrer le moteur de workflow Camunda au projet, l'utilisateur doit ajouter les dépendances nécessaires dans le fichier pom.xml, comme illustré dans la figure 3.18. Ces dépendances permettent d'activer le moteur BPM, l'interface web de gestion, ainsi que l'accès aux API REST de Camunda.

```
<dependency>
    <groupId>org.camunda.bpm.springboot</groupId>
    <artifactId>camunda-bpm-spring-boot-starter</artifactId>
    <version>7.20.0</version>
</dependency>
<dependency>
    <groupId>org.camunda.bpm.springboot</groupId>
    <artifactId>camunda-bpm-spring-boot-starter-webapp</artifactId>
    <version>7.20.0</version>
</dependency>
<dependency>
    <groupId>org.camunda.bpm.springboot</groupId>
    <artifactId>camunda-bpm-spring-boot-starter-rest</artifactId>
    <version>7.20.0</version>
</dependency>
```

Figure 3.18: Interface d'intégration de Camunda

Configurer Camunda

Après avoir ajouté les dépendances nécessaires, l'utilisateur doit configurer Camunda dans le fichier application.properties, comme montré dans la figure 3.19. Cette configuration permet de personnaliser des paramètres essentiels tels que l'authentification, l'accès à l'interface web ou encore le comportement du moteur de workflow au démarrage.

```
camunda.bpm.admin-user.id= demo
camunda.bpm.admin-user.password= demo
camunda.bpm.authorization.enabled=true
camunda.bpm.history-level=full
camunda.bpm.generate-unique-process-instance-id=true
camunda.bpm.job-execution.enabled=true
camunda.bpm.database.schema-update=true
camunda.bpm.process-engine-name=default
camunda.bpm.enabled=true
camunda.bpm.webapp.enabled=true
camunda.bpm.webapp.cors.enabled=true
camunda.bpm.webapp.cors.allowed-origins=http://localhost:4200
camunda.bpm.webapp.ui-authorization=false
```

Figure 3.19: Interface de configuration de Camunda

Interface de Camunda

Une fois l'application démarrée, l'utilisateur ayant les droits appropriés peut accéder à l'interface Cockpit de Camunda, comme illustré dans la figure 3.20. Cette interface web permet de surveiller les processus en cours d'exécution, d'examiner les incidents, et d'analyser les performances du moteur BPMN en temps réel.

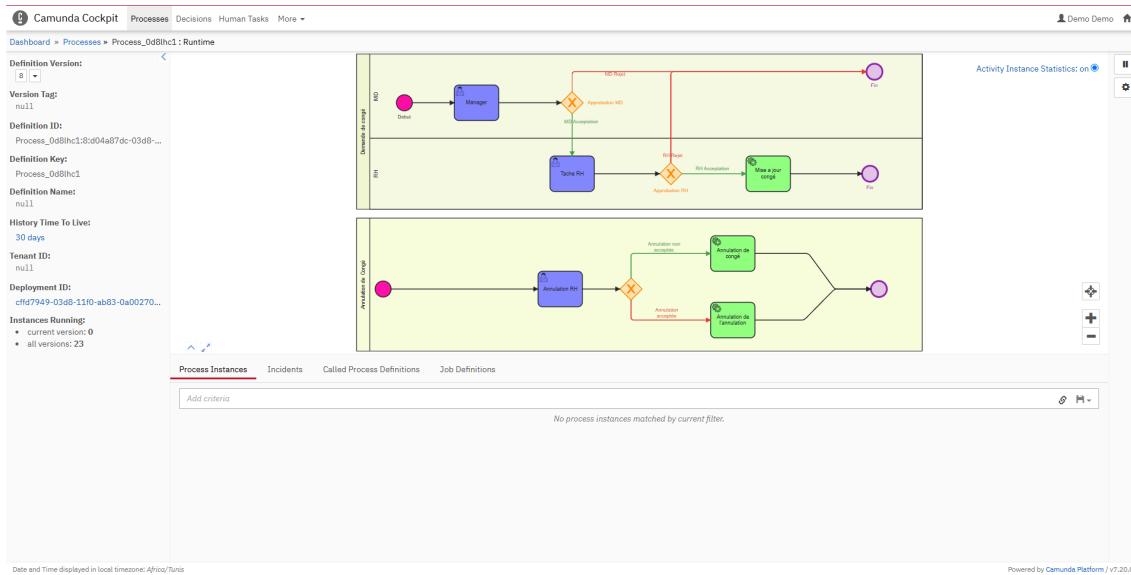


Figure 3.20: Interface cockpit de Camunda

3.6 Conclusion

Le premier sprint du projet a permis de poser des bases solides pour la gestion des accès et l'administration de la plateforme. En implémentant les fonctionnalités clés d'authentification, de déconnexion, de gestion des utilisateurs et d'intégration de Camunda, nous avons établi une infrastructure robuste et sécurisée, essentielle pour les sprints futurs. Les cas d'utilisation ont été soigneusement raffinés, conçus et réalisés, avec des interfaces utilisateur intuitives et des mécanismes techniques fiables, tels que l'utilisation des tokens JWT pour l'authentification, renforcés par les adresses IP des clients et les informations du client-agent (navigateur) pour ajouter une couche supplémentaire de sécurité. L'intégration du moteur de workflow Camunda a également permis de structurer et gérer les processus de manière fluide et évolutive.

Cette étape a non seulement répondu aux besoins fonctionnels prioritaires, mais a également renforcé la qualité, la sécurité et la fiabilité du système grâce à des pratiques de développement rigoureuses. Les fondations établies dans ce sprint garantiront une évolutivité et une maintenabilité optimales pour les prochaines itérations du projet.

Chapitre 4

Sprint 2 : Gestion des demandes

4.1 Introduction

Après avoir établi les fondations d'accès et d'administration dans le Sprint 1, le Sprint 2 se concentre sur la gestion des demandes au sein de la plateforme. Ce chapitre détaille les fonctionnalités prévues pour permettre aux utilisateurs, managers et RH de gérer efficacement les demandes et congés, tout en offrant des outils pour la consultation et le traitement de ces demandes. Nous aborderons les besoins, la conception et les étapes de réalisation de ce sprint.

4.2 Backlog du Sprint 2

Tableau 4.1: Backlog du Sprint 2

Cas d'utilisation	Priorité	Tâche
Gérer son profil	2	Modifier ses informations (nom, prénom, e-mail, etc.)
Réinitialiser son mot de passe	2	Utilisateur réinitialise via un lien sécurisé
Soumettre une demande	2	Utilisateurs, Managers, RH soumettent des demandes
Consulter ses demandes	2	Utilisateurs, Managers, RH consultent l'historique
Consulter ses congés	2	Utilisateurs, Managers, RH consultent congés, solde
Traiter une demande	2	Managers, RH valident ou rejettent les demandes

4.3 Rafinement de cas d'utilisation

Cette partie consiste à analyser et spécifier les besoins de ce deuxième sprint à travers l'identification des acteurs et le raffinement des cas d'utilisations.

4.3.1 Identification des acteurs du deuxième sprint

Les acteurs de ce sprint sont :

Utilisateur : Soumet des demandes (congés, absences) et consulte ses propres demandes et congés.

Manager : Soumet, consulte et traite les demandes (validation/rejet).

RH (Ressources Humaines) : Soumet, consulte et traite les demandes, gère les congés.

4.3.2 Raffinement du Cas d'Utilisation «Gérer son Profil»

La figure 4.1 illustre le diagramme de cas d'utilisation « Gérer le profil ».

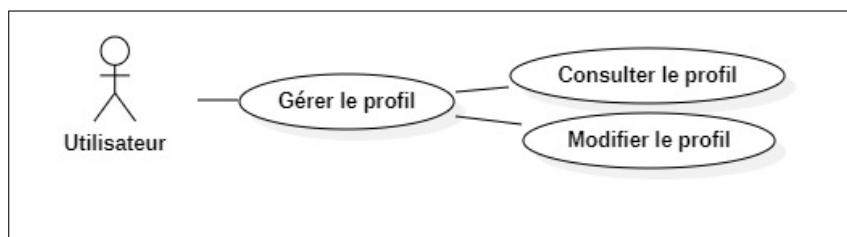


Figure 4.1: Diagramme du cas d'utilisation «Gérer le profil»

Le tableau 4.2 illustre la description textuelle du cas d'utilisation « Consulter le Profil ».

Tableau 4.2: Description textuelle du Cas d'utilisation «Consulter le Profil»

Cas d'utilisation	Consulter le Profil
Acteur	Utilisateur, Manager, RH
Pré-conditions	Système en marche. Utilisateur authentifié.
Post-conditions	Profil consulté.
Scénario de Base	1. Le système affiche l'interface du profil de l'utilisateur. 2. L'utilisateur consulte la liste des informations de son profil.
Exceptions	Échec de consultation (problème dans l'API GET, problème dans la base de données).

Le tableau 4.3 illustre la description textuelle du cas d'utilisation « Modifier le Profil ».

Tableau 4.3: Description textuelle du Cas d'utilisation «Modifier le Profil»

Cas d'utilisation	Modifier le Profil
Acteur	Utilisateur
Pré-conditions	Système en marche. Utilisateur authentifié.
Post-conditions	Profil modifié.
Scénario de Base	<ol style="list-style-type: none">1. Le système affiche l'interface du profil de l'utilisateur.2. L'utilisateur modifie les informations choisies.3. L'utilisateur clique sur le bouton « Modifier ».4. Le système modifie ces informations.5. Le système affiche un message de succès de modification.
Exceptions	Échec de modification (problème dans l'API PUT, problème dans la base de données).

4.3.3 Raffinement du Cas d'Utilisation «Réinitialiser son Mot de Passe»

La figure 4.2 illustre le diagramme de cas d'utilisation « Réinitialiser son Mot de Passe ».

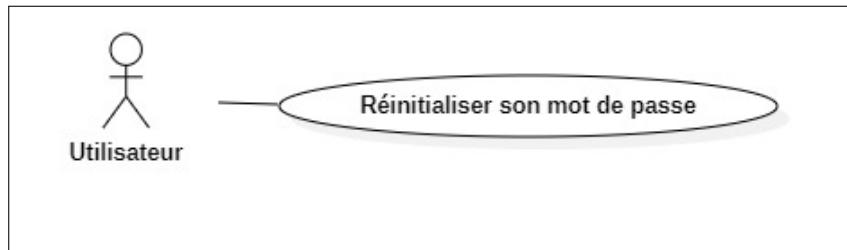


Figure 4.2: Diagramme du cas d'utilisation «Réinitialiser son Mot de Passe»

Le tableau 4.4 illustre la description textuelle du cas d'utilisation « Réinitialiser son Mot de Passe ».

Tableau 4.4: Description textuelle du Cas d'utilisation «Réinitialiser son Mot de Passe»

Cas d'utilisation	Réinitialiser son Mot de Passe
Acteur	Utilisateur
Pré-conditions	Système en marche. Utilisateur inscrit avec un e-mail valide.
Post-conditions	Mot de passe réinitialisé.
Scénario de Base	<ol style="list-style-type: none"> 1. L'utilisateur clique sur « Mot de passe oublié » sur la page de connexion. 2. Il entre son adresse e-mail et soumet la demande. 3. Le système envoie un lien sécurisé à l'e-mail de l'utilisateur. 4. L'utilisateur clique sur le lien et entre un nouveau mot de passe. 5. Le système enregistre le nouveau mot de passe et affiche un message de succès.
Exceptions	Échec de réinitialisation (e-mail invalide, lien expiré, problème API).

4.3.4 Raffinement du Cas d'Utilisation «Soumettre une Demande»

La figure 4.3 illustre le diagramme de cas d'utilisation « Soumettre une Demande ».

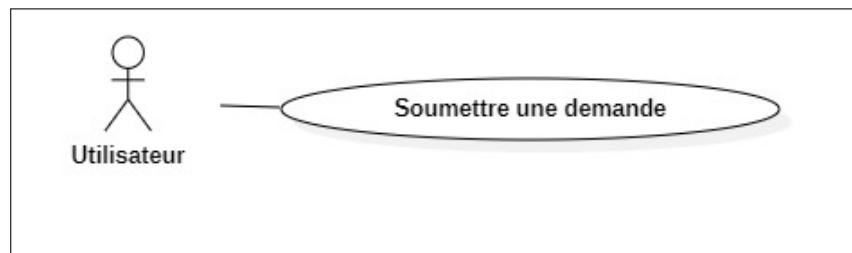


Figure 4.3: Diagramme du cas d'utilisation «Soumettre une Demande»

Le tableau 4.5 illustre la description textuelle du cas d'utilisation « Soumettre une Demande ».

Tableau 4.5: Description textuelle du Cas d'utilisation «Soumettre une Demande»

Cas d'utilisation	Soumettre une Demande
Acteur	Utilisateur, Manager, RH
Pré-conditions	Système en marche. Acteur authentifié.
Post-conditions	Demande enregistrée et parties concernées notifiées.
Scénario de Base	<ol style="list-style-type: none"> 1. L'acteur accède au formulaire de soumission. 2. L'acteur sélectionne le type de demande : <ol style="list-style-type: none"> a. <i>Demande de Congé</i> : Formulaire avec les champs (dates de début et fin, type de congé, motif). b. <i>Demande d'Autorisation</i> : Formulaire avec les champs (date, durée). 3. Il remplit les détails selon le formulaire correspondant. 4. Il soumet la demande. 5. Le système enregistre la demande et notifie les parties concernées.
Exceptions	Échec de soumission (données invalides, erreur API).

4.3.5 Raffinement du Cas d'Utilisation «Consulter ses Demandes»

La figure 4.4 illustre le diagramme de cas d'utilisation « Consulter ses Demandes ».

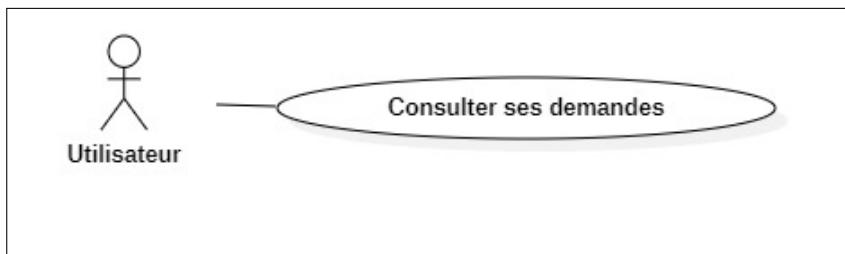


Figure 4.4: Diagramme du cas d'utilisation «Consulter ses Demandes»

Le tableau 4.6 illustre la description textuelle du cas d'utilisation « Consulter ses Demandes ».

Tableau 4.6: Description textuelle du Cas d'utilisation «Consulter ses Demandes»

Cas d'utilisation	Consulter ses Demandes
Acteur	Utilisateur, Manager, RH
Pré-conditions	Système en marche. Acteur authentifié.
Post-conditions	Historique des demandes affiché.
Scénario de Base	1. L'acteur accède à la section « Mes Demandes ». 2. Le système affiche la liste des demandes avec leur statut.
Exceptions	Échec de consultation (erreur API).

4.3.6 Raffinement du Cas d'Utilisation «Consulter ses Congés»

La figure 4.5 illustre le diagramme de cas d'utilisation « Consulter ses Congés ».

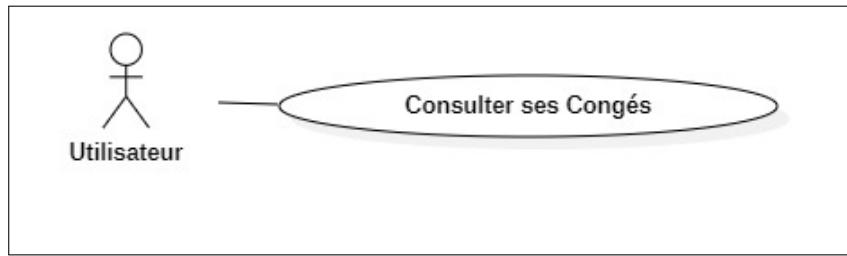


Figure 4.5: Diagramme du cas d'utilisation «Consulter ses Congés»

Le tableau 4.7 illustre la description textuelle du cas d'utilisation « Consulter ses Congés ».

Tableau 4.7: Description textuelle du Cas d'utilisation «Consulter ses Congés»

Cas d'utilisation	Consulter ses Congés
Acteur	Utilisateur, Manager, RH
Pré-conditions	Système en marche. Acteur authentifié.
Post-conditions	Solde et historique des congés affichés.
Scénario de Base	1. L'acteur accède à la section « Mes Congés ». 2. Le système affiche le solde et l'historique des congés.
Exceptions	Échec de consultation (erreur API).

4.3.7 Raffinement du Cas d'Utilisation «Traiter une Demande»

La figure 4.6 illustre le diagramme de cas d'utilisation « Traiter une Demande ».

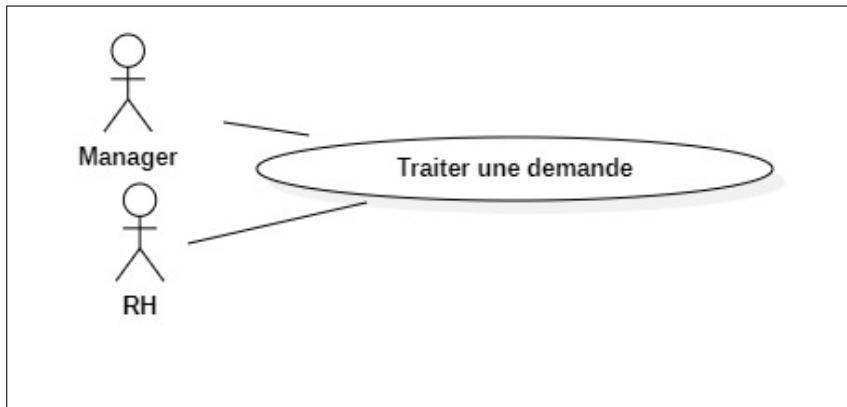


Figure 4.6: Diagramme du cas d'utilisation «Traiter une Demande»

Le tableau 4.8 illustre la description textuelle du cas d'utilisation « Traiter une Demande ».

Tableau 4.8: Description textuelle du Cas d'utilisation «Traiter une Demande»

Cas d'utilisation	Traiter une Demande
Acteur	Manager, RH
Pré-conditions	Système en marche. Acteur authentifié.
Post-conditions	Statut de la demande mis à jour et émetteur notifié.
Scénario de Base	1. L'acteur accède à la liste des demandes en attente. 2. Il sélectionne une demande. 3. Il choisit de valider ou rejeter. 4. Le système met à jour le statut et notifie l'émetteur.
Exceptions	Échec de traitement (erreur API).

4.4 Conception

La conception du Sprint 2 vise à modéliser les interactions entre les demandes, les congés et les notifications, en s'appuyant sur une architecture orientée objet et une intégration avec Camunda pour la gestion des processus.

4.4.1 Diagramme de classe du sprint 2

La figure 4.7 illustre le diagramme de classe du Sprint 2.

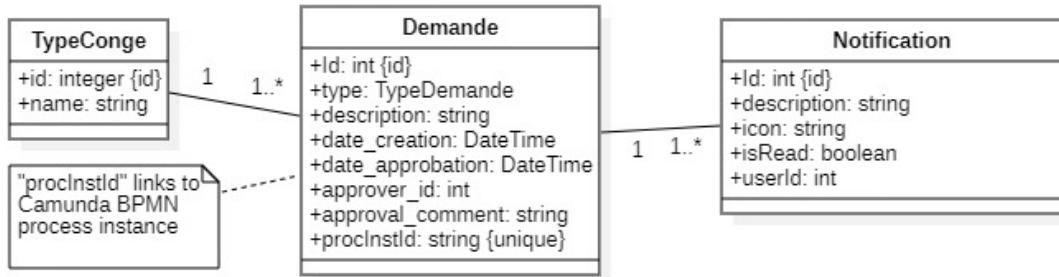


Figure 4.7: Diagramme de classe du Sprint 2

Le diagramme de classe est formé de :

- **Classe "TypeConge"** : Représente les différents types de congés disponibles sur la plate-forme.
- **Classe "Demande"** : Gère les demandes soumises par les utilisateurs, comme les congés ou autorisations.
- **Classe "Notification"** : Gère les notifications envoyées aux utilisateurs pour les informer sur leurs demandes.
- **Note sur procInstId** : L'attribut procInstId est un identifiant unique qui lie une instance de la classe (e.g., TypeConge et Demande) à une instance de processus BPMN dans Camunda. Cela permet de suivre et de gérer le workflow associé (e.g., approbation de demande) via le moteur Camunda.

4.4.2 Conception du Cas d'Utilisation «Gérer son Profil»

La figure 4.8 illustre le diagramme de classe du cas d'utilisation « Gérer son Profil ».

Diagramme de Classe

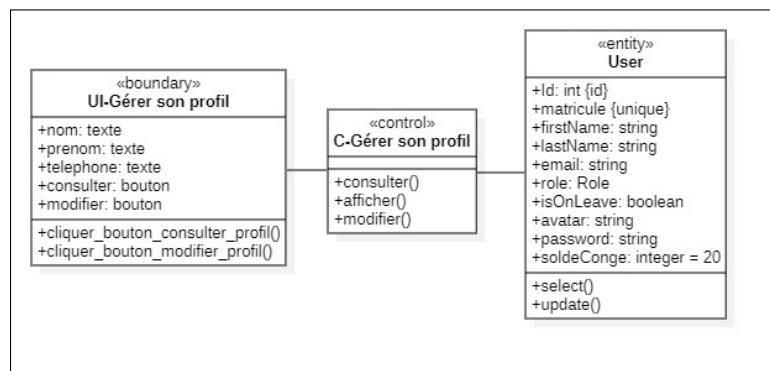


Figure 4.8: Diagramme de classe du cas d'utilisation «Gérer son Profil»

Diagramme de Séquence

Tous les acteurs (Utilisateur, Manager, RH) ont la possibilité de gérer leurs profils à partir de leurs interfaces respectives. Lorsqu'un acteur consulte ou modifie son profil, l'interface communique avec le contrôleur de profil, qui interagit avec l'entité « User » pour récupérer ou mettre à jour les informations dans la base de données.

Ce scénario est présenté dans la figure 4.9, intitulé « diagramme de séquence du cas d'utilisation Gérer son Profil ».

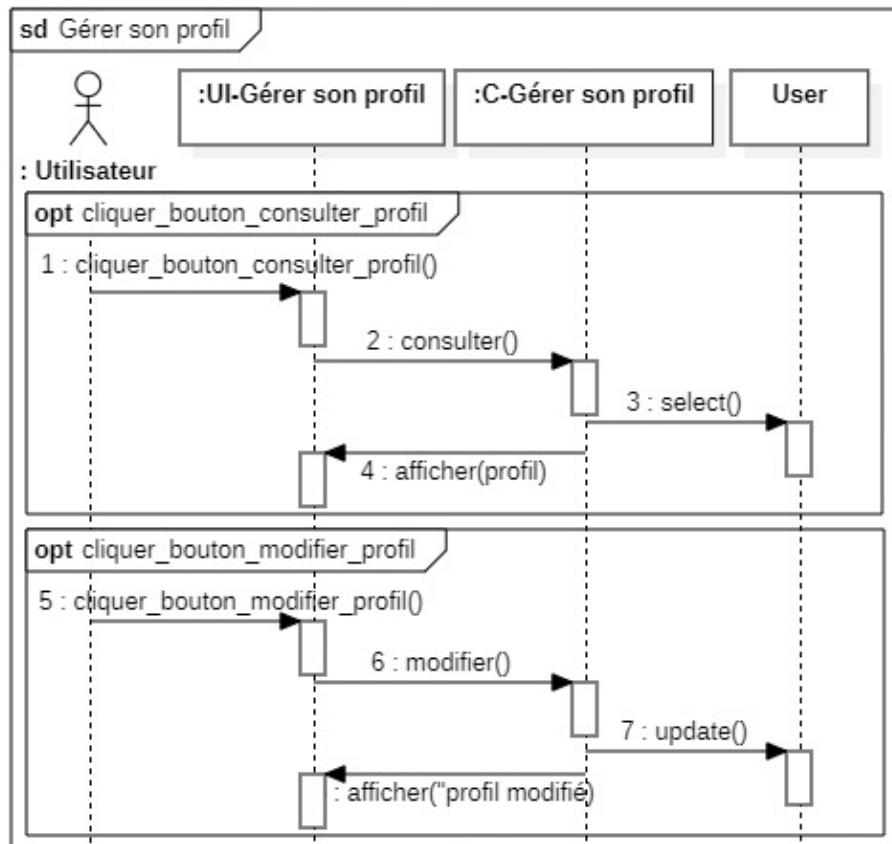


Figure 4.9: Diagramme de séquence du cas d'utilisation «Gérer son Profil»

4.4.3 Conception du Cas d'utilisation «Réinitialiser son mot de passe»

Diagramme de Classe

La figure 4.10 illustre le diagramme de classe du cas d'utilisation « Réinitialiser son mot de passe ».

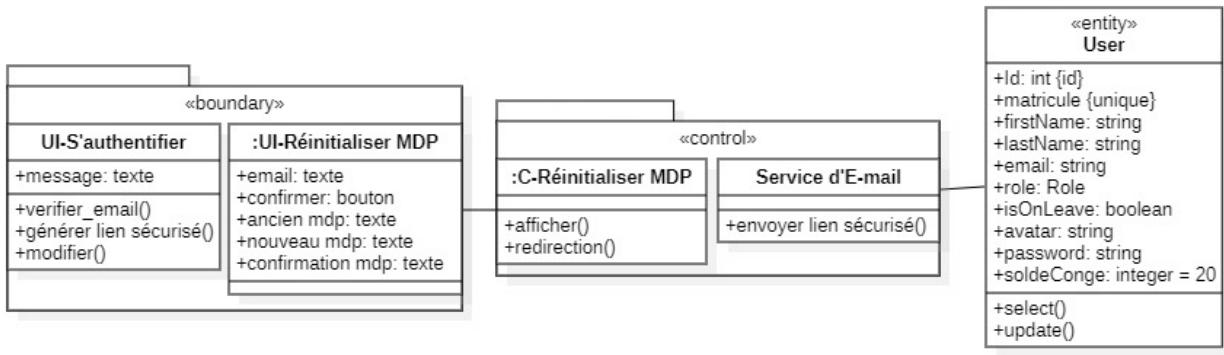


Figure 4.10: Diagramme de classe du cas d'utilisation «Réinitialiser son Mot de Passe»

Diagramme de Séquence

L'utilisateur qui a oublié son mot de passe initie le processus via l'interface de connexion. L'interface communique avec le contrôleur, qui vérifie l'e-mail de l'utilisateur dans l'entité « User ». Si l'e-mail n'existe pas, le système renvoie un message : « Si votre e-mail existe dans la base, vous recevrez un e-mail pour réinitialiser votre mot de passe », afin de ne pas divulguer l'existence de l'e-mail pour des raisons de sécurité. Si l'e-mail existe, le contrôleur génère un lien sécurisé et envoie une notification par e-mail via un service. Un attaquant potentiel recevant le lien sans connaître le propriétaire de l'e-mail ne peut pas modifier le mot de passe, car l'utilisateur doit resaisir son adresse e-mail pour confirmer son identité. Une fois le lien utilisé, l'utilisateur soumet un nouveau mot de passe, qui est mis à jour dans la base de données via le contrôleur. Après cette mise à jour, le lien devient expiré pour des raisons de sécurité. Ce scénario est présenté dans la figure 4.11, intitulé diagramme de séquence du cas d'utilisation « Réinitialiser son Mot de Passe ».

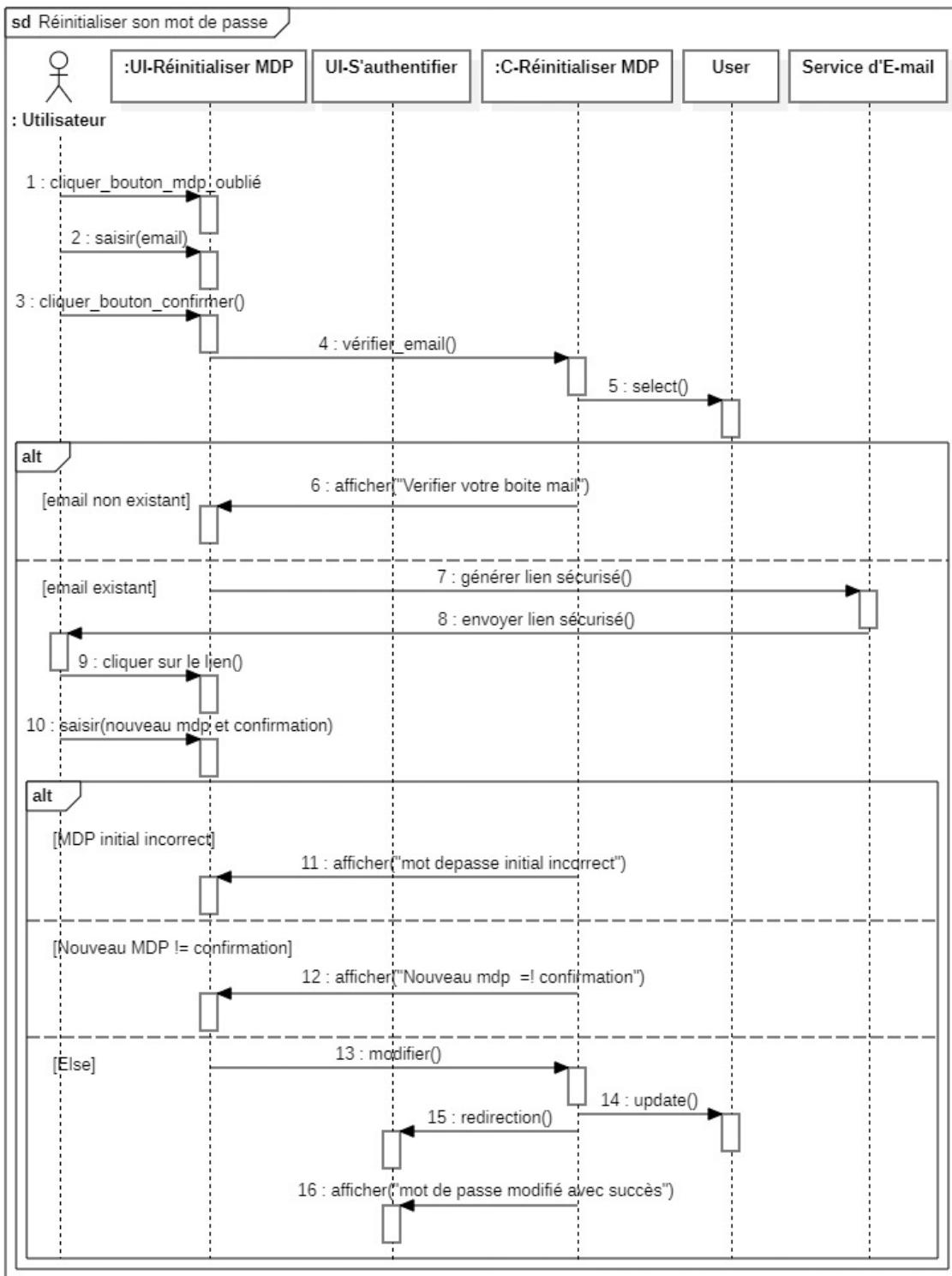


Figure 4.11: Diagramme de séquence du cas d'utilisation «Réinitialiser son Mot de Passe»

4.4.4 Conception du Cas d'Utilisation «Soumettre une Demande»

Diagramme de Classe

La figure 4.12 illustre le diagramme de classes du cas d'utilisation « Soumettre une demande ».

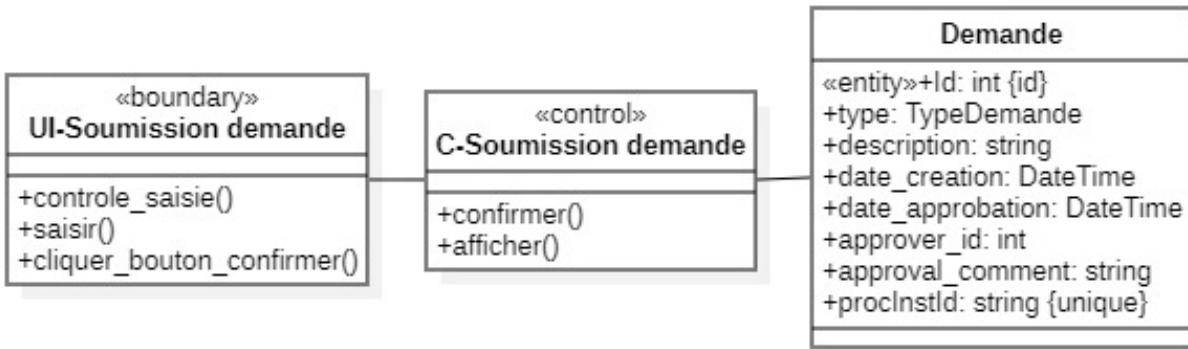


Figure 4.12: Diagramme de classe du cas d'utilisation «Soumettre une demande»

Diagramme de Séquence

Ce scénario décrit le processus de soumission d'une demande de congé par un utilisateur via l'interface de l'application. L'utilisateur commence par remplir un formulaire de demande (étape 1). La saisie est ensuite contrôlée (étape 2) pour vérifier la validité des données. En cas d'erreur, un message est affiché (étape 3) et l'utilisateur peut corriger sa saisie, dans une boucle de validation.

Une fois les données correctement saisies (étape 4), l'utilisateur clique sur le bouton de confirmation (étape 5). Cette action est capturée par le contrôleur de soumission (**C-Soumission demande**), qui déclenche la méthode `confirmer()` (étape 6). Le contrôleur interagit ensuite avec le module de gestion des demandes (**Demande**) pour insérer la nouvelle demande dans la base de données via `insert()` (étape 7).

Parallèlement, un processus BPMN est initié dans Camunda BPM afin d'orchestrer la suite du workflow de validation (par exemple, affectation de la demande à un manager pour approbation). Enfin, un message de confirmation est retourné à l'interface pour informer l'utilisateur que sa demande a été créée avec succès (étape 8).

La figure 4.13, intitulé diagramme de séquence du cas d'utilisation « Soumettre une Demande ».

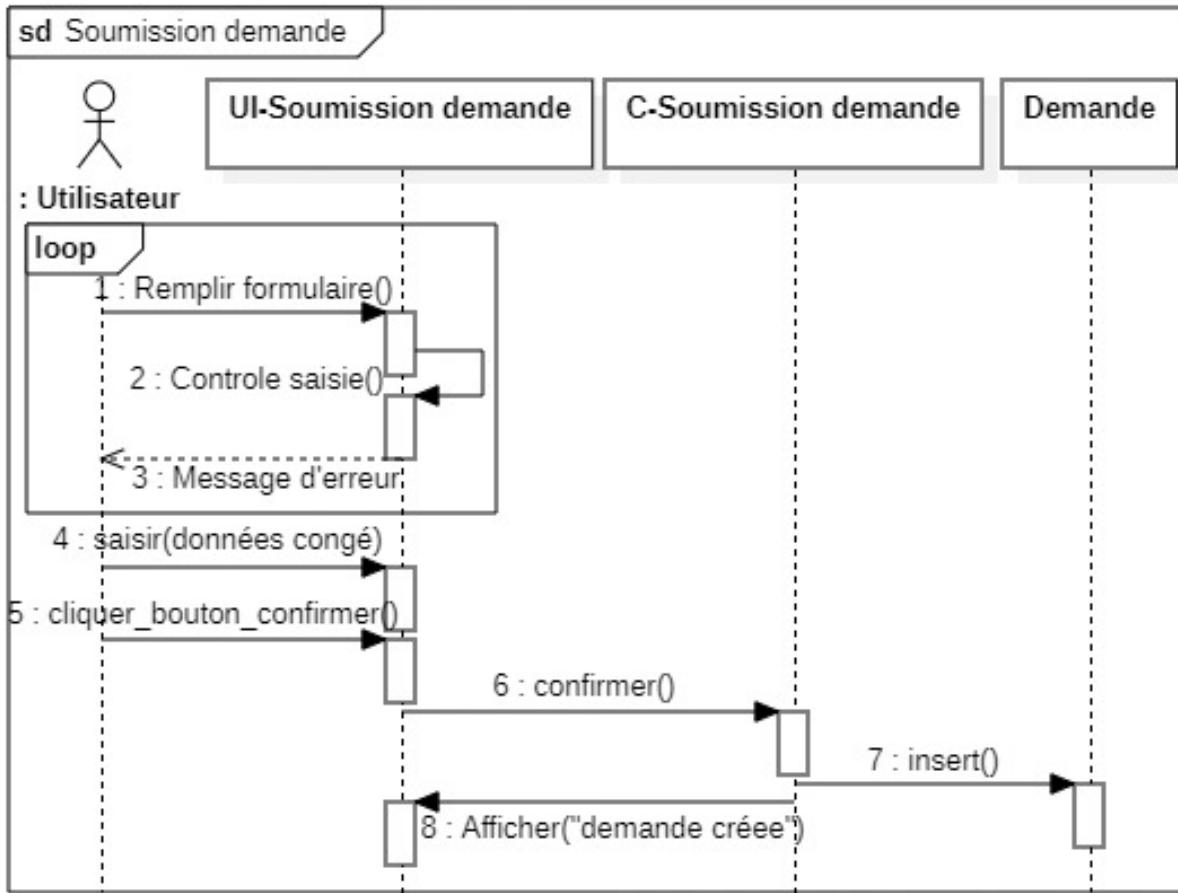


Figure 4.13: Diagramme de séquence du cas d'utilisation «Soumettre une demande»

Diagramme Business Process Model and Notation

Le diagramme BPMN ci-dessous illustre le processus métier de gestion des demandes et annulations de congé dans la plateforme. Il est structuré en deux grands processus : Demande de congé et Annulation de congé, chacun réparti entre plusieurs rôles (ou swimlanes), notamment le Manager, le service RH, et l'initiateur de l'annulation.

Demande de congé : Le processus débute par la soumission d'une demande par l'utilisateur. Celle-ci est d'abord examinée par le Manager, qui peut soit approuver, soit rejeter la demande.

En cas de rejet, le processus s'arrête immédiatement.

En cas d'approbation, la demande est transmise au RH pour une seconde validation. Si le RH rejette la demande, le processus s'arrête également.

En cas de double approbation (Manager + RH), une tâche de type service est exécutée pour mettre à jour les données du congé dans le système, marquant la fin du processus.

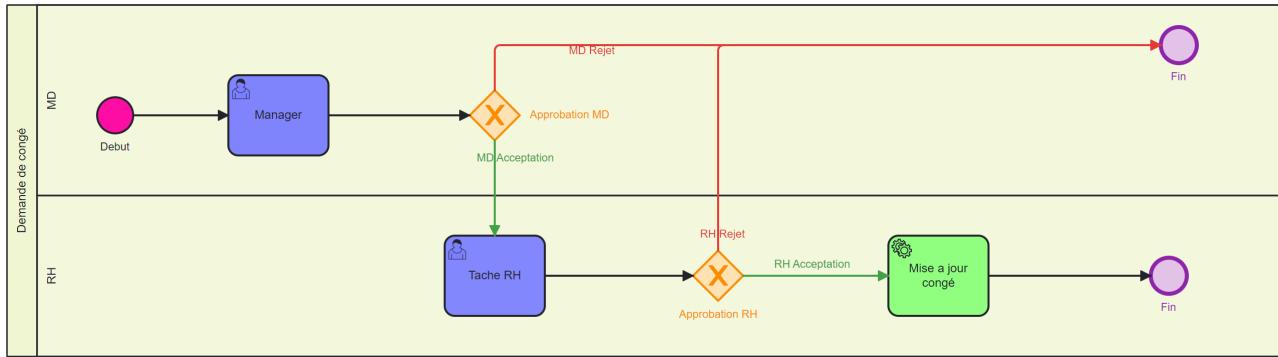


Figure 4.14: Diagramme BPMN demande de congé

Demande d'autorisation : Le processus débute par un événement de départ (**Début**). Ensuite, une tâche utilisateur intitulée **Demande autorisation** est initiée, dans laquelle un utilisateur soumet une requête d'autorisation.

Un **gateway exclusif** (losange avec croix) permet de rediriger le processus selon le résultat de la demande :

Si la demande est refusée (chemin rouge), le processus se termine immédiatement par un événement de fin.

Si la demande est approuvée (chemin vert), une tâche de service **Update autorisation** est exécutée. Celle-ci permet de mettre à jour les informations d'autorisation dans le système. Le processus se termine ensuite par un événement de fin classique.

Ce processus est typiquement orchestré par un moteur BPM tel que **Camunda**, permettant d'associer automatiquement des règles métiers et des traitements conditionnels à chaque étape. L'utilisation du gateway permet de gérer de façon claire et contrôlée les différentes issues de la demande.

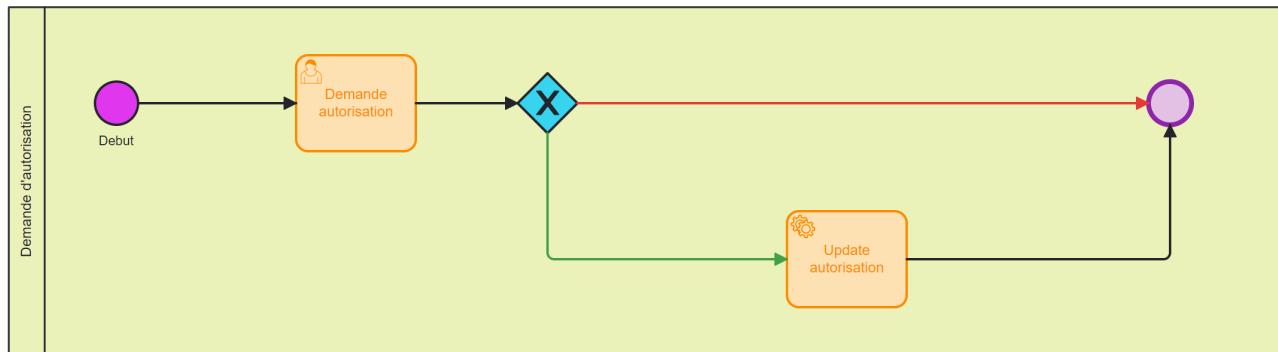


Figure 4.15: Diagramme BPMN demande d'autorisation

4.4.5 Conception du Cas d'Utilisation «Consulter ses demandes»

Diagramme de Classe

La figure 4.16 illustre le diagramme de classes du cas d'utilisation « Consulter ses demandes ».



Figure 4.16: Diagramme de classe du cas d'utilisation «Consulter ses demandes»

4.4.6 Diagramme de Séquence

Les utilisateurs, les managers et les RH ont la possibilité de consulter leurs demandes à partir de leurs interfaces respectives. Les actions qu'ils réalisent pour consulter leurs demandes sont exécutées suite à la communication entre l'interface utilisateur et le contrôleur de consultation, qui interagit avec l'entité "Demande" pour récupérer les informations correspondantes depuis la base de données. Ce scénario est présenté dans la figure 4.17.

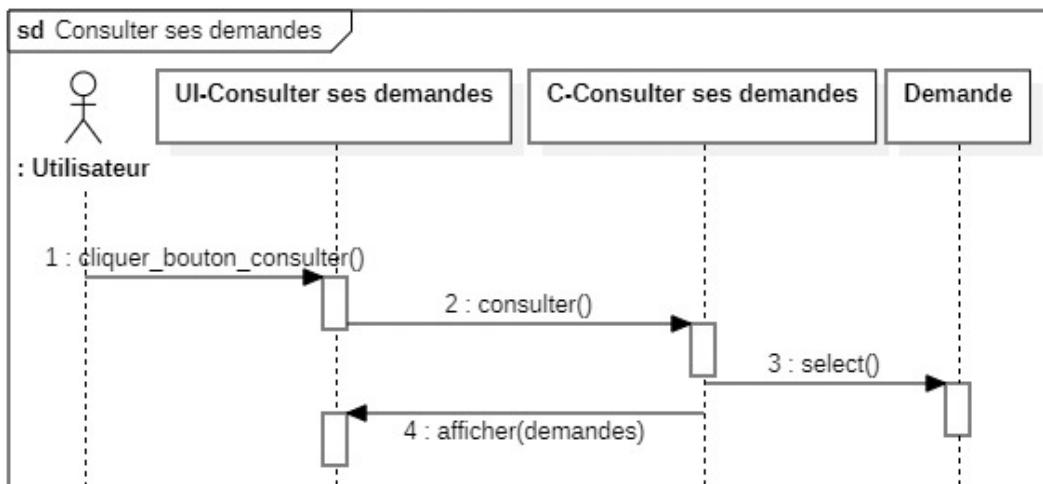


Figure 4.17: Diagramme de séquence du cas d'utilisation «Consulter ses demandes»

4.4.7 Conception du Cas d'Utilisation «Consulter ses congés»

Diagramme de Classe

La figure 4.18 illustre le diagramme de classes du cas d'utilisation « Consulter ses congés ».



Figure 4.18: Diagramme de classe du cas d'utilisation «Consulter ses congés»

Diagramme de Séquence

Les utilisateurs, les managers et les RH ont la possibilité de consulter leurs congés à partir de leurs interfaces respectives. Les actions qu'ils réalisent pour consulter leurs congés sont exécutées suite à la communication entre l'interface utilisateur et le contrôleur de consultation, qui interagit avec l'entité "Demande" pour récupérer les informations correspondantes depuis la base de données(Les demandes approuvées). Ce scénario est présenté dans la figure 4.19.

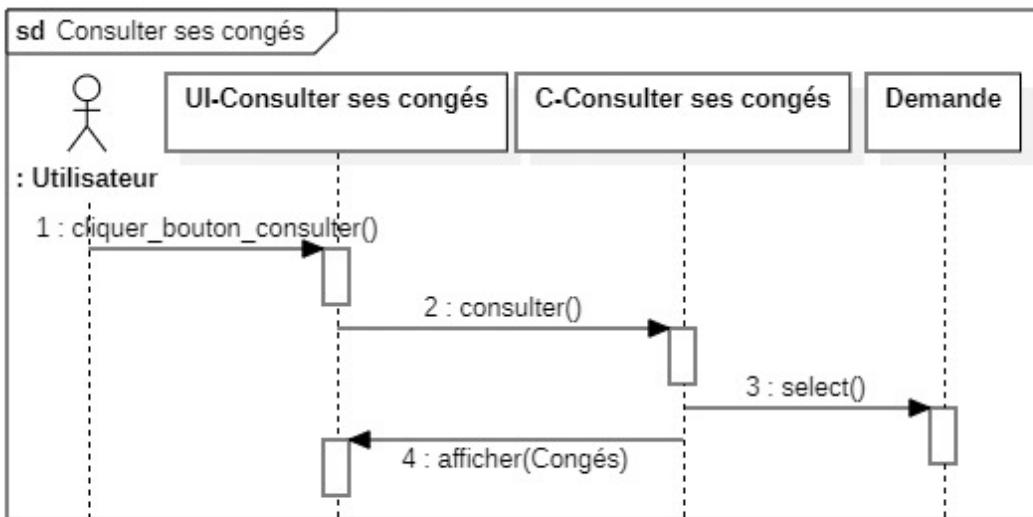


Figure 4.19: Diagramme de séquence du cas d'utilisation «Consulter ses congés»

4.4.8 Conception du Cas d'Utilisation «Traiter une demande»

Diagramme de Classe

La figure 4.20 illustre le diagramme de classes du cas d'utilisation « Traiter une demande ».

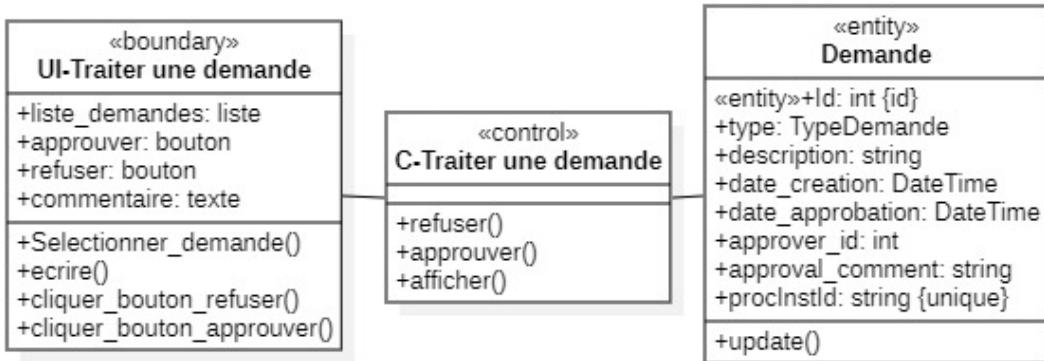


Figure 4.20: Diagramme de classe du cas d'utilisation «Traiter une demande»

Diagramme de Séquence

Les managers et les responsables RH ont la possibilité de traiter les demandes qui leur sont adressées. Lors du traitement d'une demande, le manager doit d'abord approuver ou rejeter la demande. Si le manager rejette, la demande est automatiquement refusée. Si le manager approuve, la demande est transmise au responsable RH qui, à son tour, doit également approuver ou rejeter. Si le responsable RH rejette, la demande est refusée ; sinon, elle est approuvée. Ce scénario est illustré dans la figure 4.21.

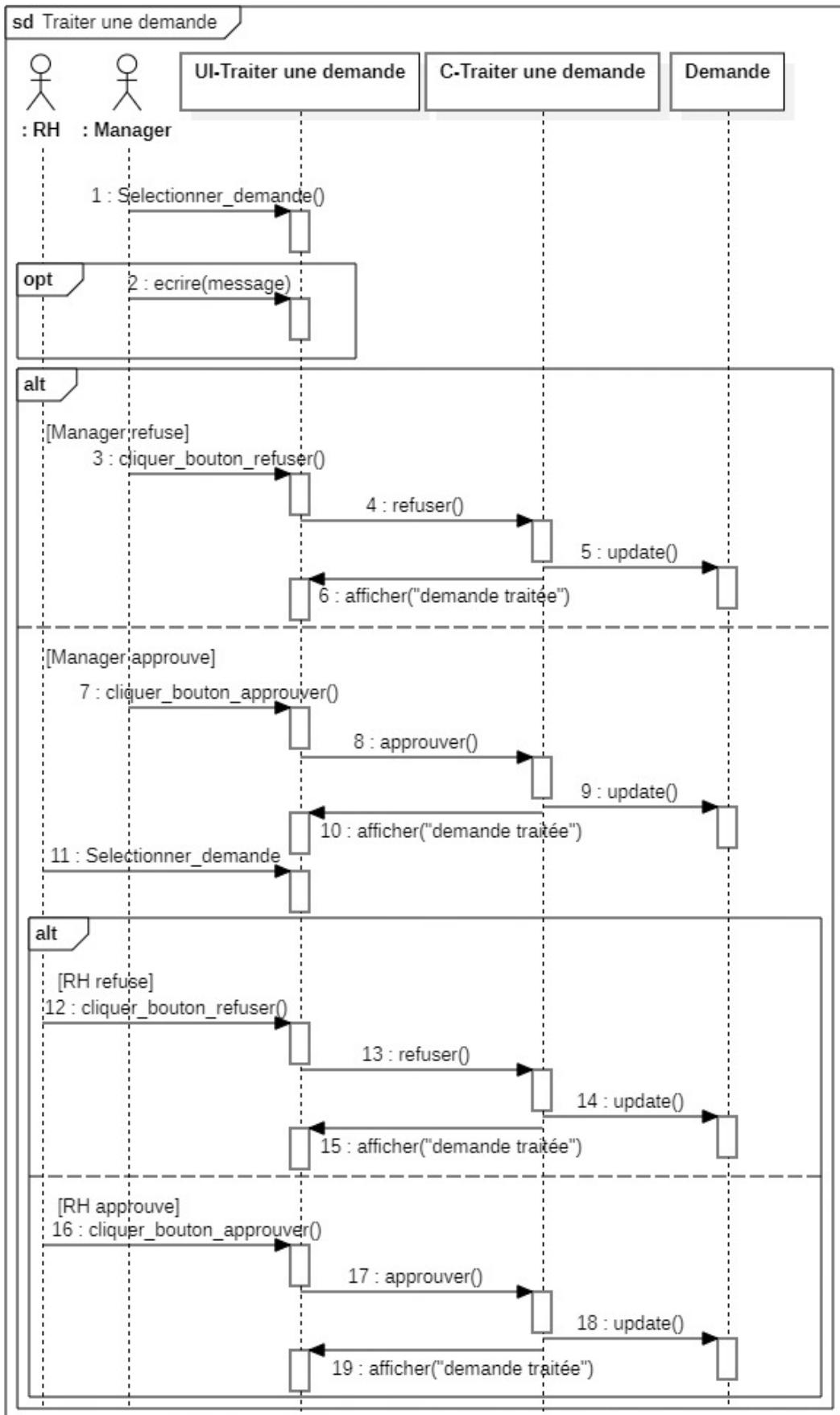


Figure 4.21: Diagramme de séquence du cas d'utilisation «Traiter une demande»

4.5 Réalisation

Dans cette partie, nous présentons les modules de notre premier sprint en utilisant des captures d'écran.

4.5.1 Gérer son profil

L'interface de gestion du profil présentée dans la figure 4.22 permet aux utilisateurs de consulter et de modifier leurs informations personnelles. Ils ont la possibilité de mettre à jour leur photo de profil, ainsi que de corriger leur nom et prénom en cas d'erreur. Cette fonctionnalité garantit que les données utilisateur restent exactes et à jour, offrant ainsi une expérience personnalisée et fiable.

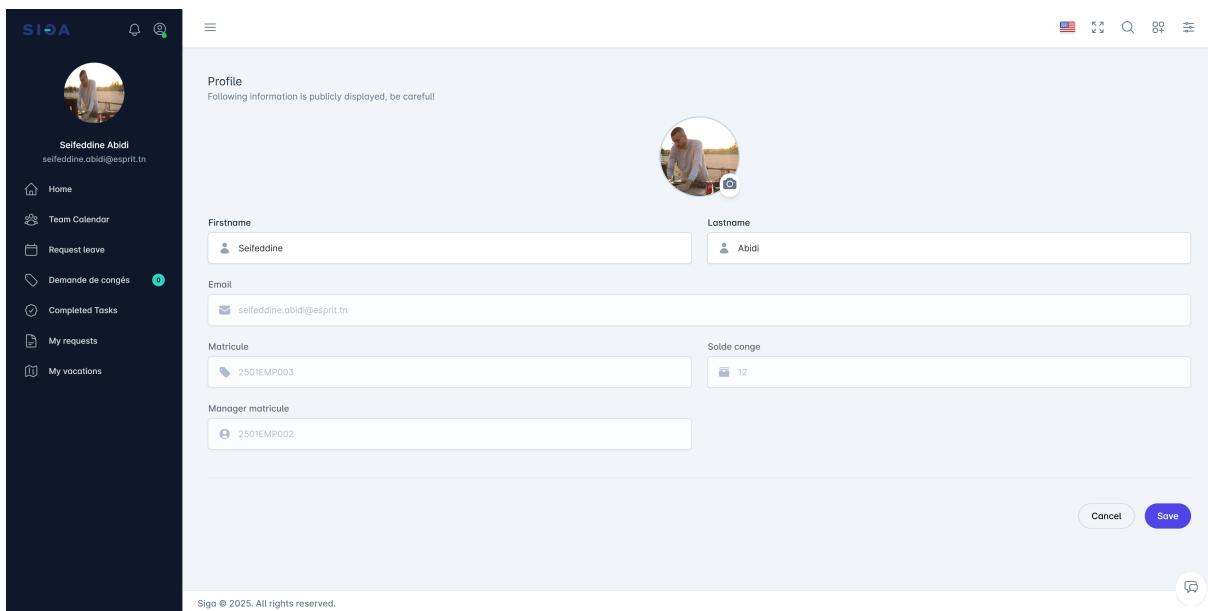


Figure 4.22: Interface du cas d'utilisation «Gérer son profil»

4.5.2 Réinitialiser son mot de passe

Pour réinitialiser le mot de passe, on doit suivre ces étapes :

- Accédez à la page "Mot de passe oublié ?"** : Depuis la page de connexion, cliquez sur "Forgot password?".
- Entrez votre adresse e-mail** : Saisissez l'adresse e-mail associée à votre compte dans le champ prévu ("Email address").
- Envoyez la demande** : Cliquez sur le bouton "Send reset link". Un message confirmera l'envoi d'un e-mail si votre adresse est enregistrée dans le système.
- Consultez votre boîte de réception** : Vous recevrez un e-mail intitulé "Password Reset Request" avec un bouton "Reset Password". Cliquez sur ce bouton.

5. **Créez un nouveau mot de passe** : Sur la page de réinitialisation, entrez votre e-mail, un nouveau mot de passe, et confirmez ce mot de passe dans les champs correspondants.
6. **Validez** : Cliquez sur "Reset your password" pour enregistrer votre nouveau mot de passe.

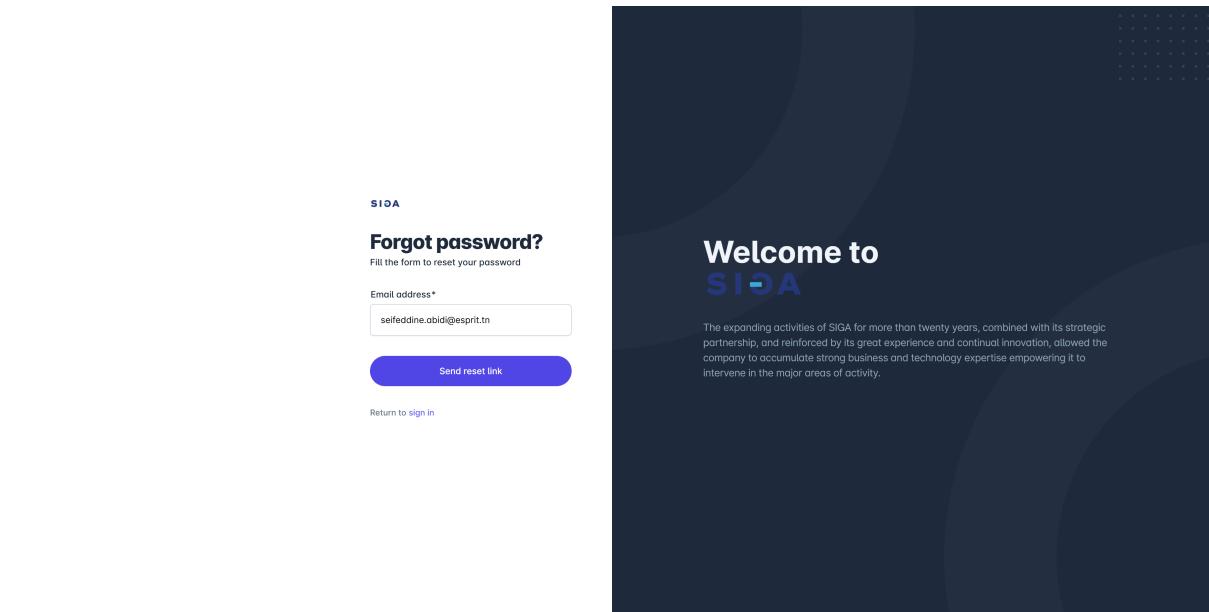


Figure 4.23: Page "Mot de passe oublié ?"

Forgot password?
Fill the form to reset your password

Password reset sent! You'll receive an email if you are registered on our system.

Email address*

Send reset link

Return to [sign in](#)

Figure 4.24: Confirmation de l'envoi du lien de réinitialisation

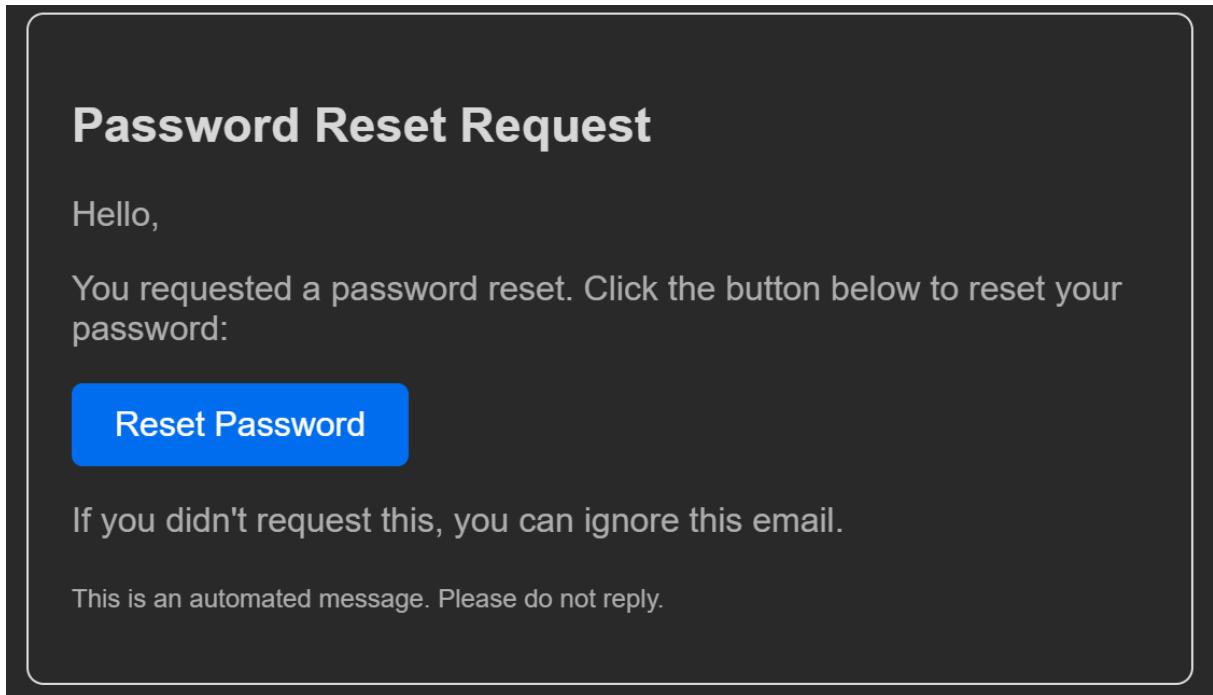


Figure 4.25: E-mail de réinitialisation de mot de passe

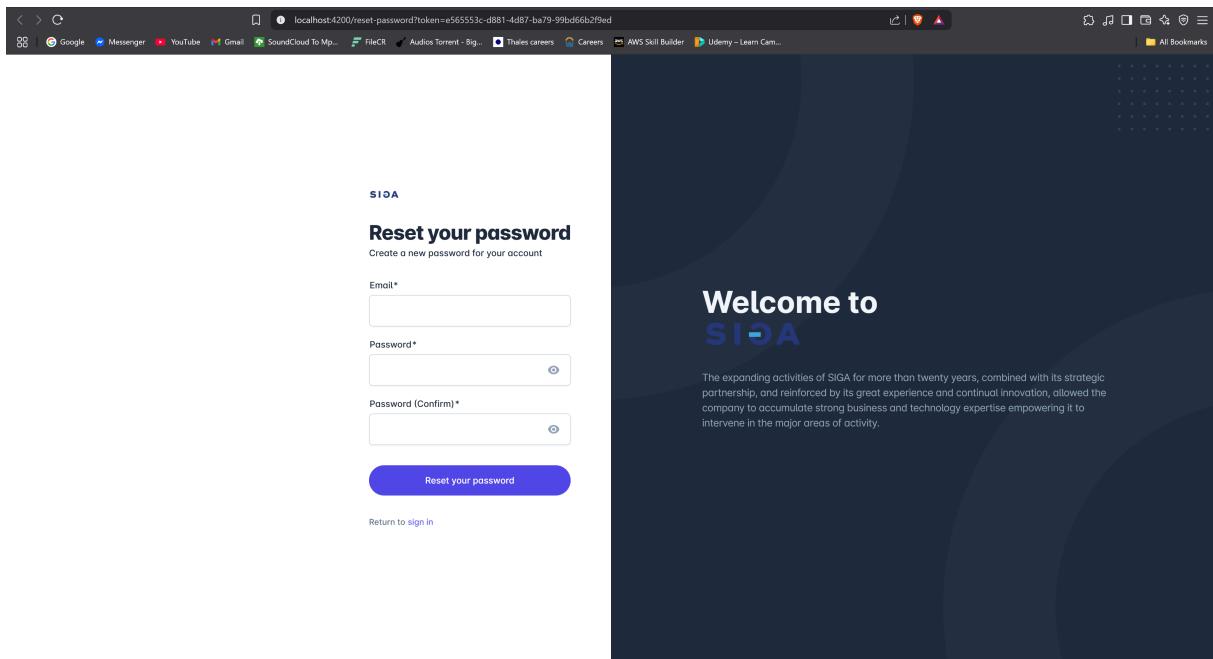


Figure 4.26: Page de création d'un nouveau mot de passe

4.5.3 Soumettre une demande

- 1. Sélectionner le type de la demande :** Commencez par choisir le type de demande que vous souhaitez soumettre via un menu déroulant ou une liste (par exemple, "Leave request" pour une demande de congé comme illustré dans la figure 4.27 et 4.28).
- 2. Remplir les informations nécessaires :** Indiquez les détails requis, tels que les dates de début et de fin, ainsi que les horaires si pertinents.
- 3. Ajouter des options supplémentaires :** Si applicable, précisez des préférences ou conditions spécifiques (par exemple, un départ ou un retour en milieu de journée).
- 4. Soumettre la demande :** Une fois toutes les informations saisies, validez votre demande en cliquant sur un bouton de soumission. Vous pouvez également annuler si nécessaire.

The screenshot shows a web-based application interface for creating a new request. At the top, there is a navigation bar with icons for search, refresh, and other functions. Below the navigation is a breadcrumb trail: 'Request > Add'. The main title is 'Add Request'. A dropdown menu is open, showing 'Leave request' as the selected option. The form itself has a light gray background and contains the following fields:

- A section labeled 'Request' with the sub-instruction 'Add here your request details and submit it.'
- 'Start Date*' field with a date input box and a calendar icon.
- 'End Date*' field with a date input box and a calendar icon.
- Two checkboxes:
 - Departure After Midday
 - Return After Midday
- At the bottom right of the form area are 'Cancel' and 'Create request' buttons, along with a small message icon.

Figure 4.27: Page de création d'une nouvelle demande de congé

Request > Add

Add Request

Authorization request

Authorization
Add here your request details and submit it.

Date *

Start Time *

End Time *

Cancel **Submit** 

Figure 4.28: Page de création d'un nouvelle demande d'autorisation

4.5.4 Consulter ses demandes

Requests > List

Requests list

Motricule	Email	Manager	Request Date	Start Date	End Date	Number of days	GoAfterMidday	BackAfterMidday	Status
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-22 10:19:19	2025-04-30T00:00:00	2025-04-05T00:00:00	26 days	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 16:16:40	2025-05-05T12:00:00	2025-05-05T13:00:00	1 day	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 13:29:28	2025-04-07T12:00:00	2025-04-07T13:00:00	1 day	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 13:18:22	2025-05-01T12:00:00	2025-05-01T14:00:00	1 day	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 11:15:53	2025-04-30T11:00:00	2025-04-30T12:00:00	1 day	NO	NO	REJECTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 10:01:33	2025-04-29T12:00:00	2025-04-29T13:00:00	1 day	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 09:59:37	2025-04-25T12:00:00	2025-04-25T13:00:00	1 day	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 09:46:08	2025-04-28T12:00:00	2025-04-28T13:00:00	1 day	NO	NO	ACCEPTED
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 09:32:24	2025-04-27T12:00:00	2025-04-27T13:00:00	1 day	NO	NO	ACCEPTED



Figure 4.29: Page de consultation des demandes personnelles

L'interface pour consulter ses demandes sur une plateforme comme celle présentée est généralement conçue pour être intuitive et organisée. Une fois dans cette section, l'utilisateur voit une liste ou un tableau répertoriant ses demandes soumises. Chaque entrée affiche des informations clés telles que la date de soumission, les dates demandées (début et fin), et l'état de la demande (en attente, approuvée ou rejetée). Des options de tri peuvent être présentes, permettant de classer les demandes par statut ou par date.

4.5.5 Consulter ses congés

L'interface pour consulter ses congés, c'est-à-dire les demandes de congés qui ont été acceptées, est conçue pour fournir une vue organisée et spécifique des congés approuvés. Une fois dans cette section, l'utilisateur voit une liste ou un tableau répertoriant uniquement ses demandes de congés ayant été acceptées. Chaque entrée affiche des informations clés telles que les dates de début et de fin du congé, la date d'approbation, et éventuellement la durée.

Matricule	Email	Manager	Request Date	Start Date	End Date	Number of days	GoAfterMidday	BackAfterMidday
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-22 10:19:19	2025-04-30T00:00:00	2025-04-05T00:00:00	26 days	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 16:16:40	2025-05-05T12:00:00	2025-05-05T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 13:29:28	2025-04-07T12:00:00	2025-04-07T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 13:18:22	2025-05-01T12:00:00	2025-05-01T14:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 10:01:33	2025-04-29T12:00:00	2025-04-29T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 09:59:37	2025-04-25T12:00:00	2025-04-25T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 09:46:08	2025-04-28T12:00:00	2025-04-28T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 09:32:24	2025-04-27T12:00:00	2025-04-27T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-04-10 08:11:48	2025-04-09T00:00:00	2025-04-14T00:00:00	6 days	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-03-07 10:36:03	2025-03-07T16:00:00	2025-03-07T17:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-02-28 15:37:20	2025-03-31T12:00:00	2025-03-31T13:00:00	1 day	NO	NO
2501EMP005	hughes.brian@company.com	Iheb Yousfi	2025-02-28 15:03:45	2025-02-21T12:00:00	2025-02-21T13:00:00	1 day	NO	NO

Figure 4.30: Page de consultation des congés acceptés

4.5.6 Traiter une demande

L'interface pour traiter une demande est conçue pour permettre l'approbation ou le refus d'une demande. Le manager ou l'RH peut choisir entre deux options : accepter ou refuser la demande. Une fois la décision prise, il peut cliquer sur un bouton de validation, ce qui envoie une notification à l'utilisateur qui a soumis la demande. Les notifications peuvent être envoyées par e-mail ou par un message de notification sur la plateforme.

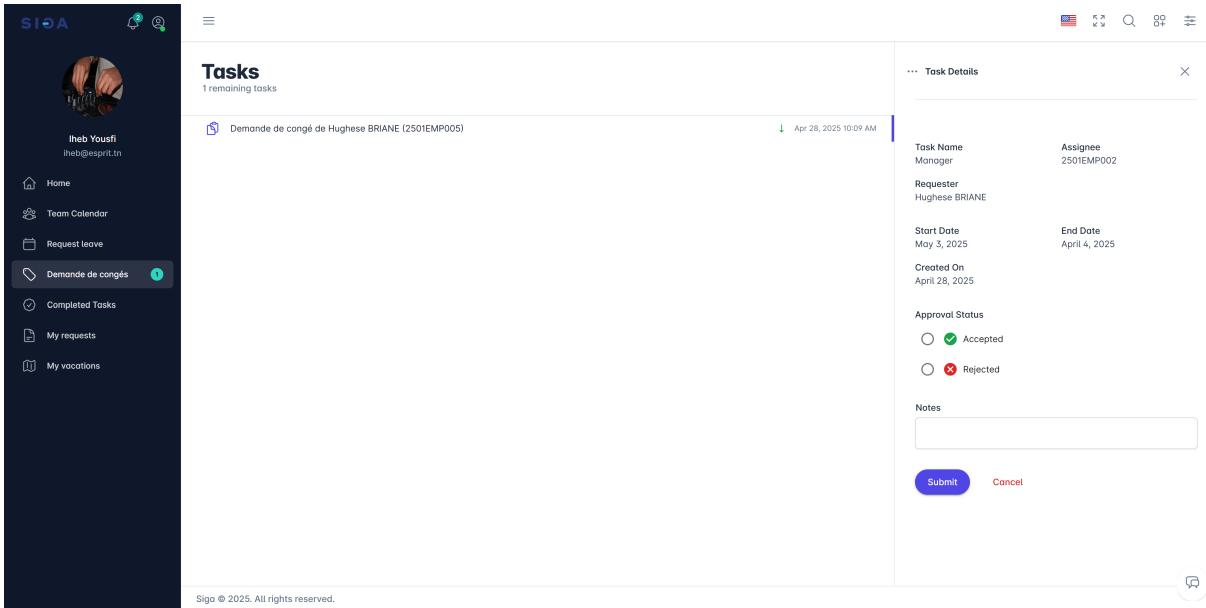


Figure 4.31: Page de traitement des demandes

4.6 Conclusion

Le Sprint 2 a permis de développer des fonctionnalités essentielles pour la gestion des demandes et congés, répondant aux besoins des utilisateurs, managers et RH.

Le backlog a défini les cas d'utilisation prioritaires, raffinés via des diagrammes UML (cas d'utilisation, classes, séquences) et BPMN pour modéliser les interactions et workflows.

La conception, basée sur une architecture orientée objet et intégrant Camunda, a soutenu une réalisation réussie, avec des interfaces intuitives pour gérer les profils, réinitialiser les mots de passe, soumettre, consulter et traiter les demandes.

Ce sprint a établi une gestion efficace des demandes, ouvrant la voie à de futures améliorations comme des notifications avancées ou des tableaux de bord analytiques.

Chapitre 5

Sprint 3 : Suivi et Supervision

5.1 Introduction

Fort des avancées réalisées dans les Sprints 1 et 2, qui ont mis en place les bases de l'accès, de l'administration et de la gestion des demandes, le Sprint 3 se focalise sur le suivi et la supervision au sein de la plateforme. Ce chapitre présente les fonctionnalités conçues pour permettre aux utilisateurs, managers, RH et administrateurs de superviser efficacement les processus, de consulter les demandes d'approbation, de suivre leurs équipes et de consulter les crédits de congés. Nous détaillerons les besoins identifiés, la conception de ces fonctionnalités et les étapes de leur mise en œuvre dans ce sprint.

5.2 Backlog du Sprint 3

Le tableau 5.1 représente le backlog du troisième sprint. Ce tableau détaille les cas d'utilisation, leurs priorités, estimations et tâches associées.

Tableau 5.1: Backlog du Sprint 3 : Suivi et Supervision

Cas d'utilisation	Priorité	Tâche
Recevoir une notification	2	Utilisateurs, Managers et RH reçoivent des alertes sur des actions importantes.
Consulter les processus métiers	3	Administrateur consulte les processus métiers définis dans le système.
Consulter les demandes d'approbation	3	Administrateur accède aux demandes soumises pour consultation.
Consulter les membres de l'équipe	3	Utilisateurs, Managers et RH accèdent aux informations des membres de leur équipe.
Consulter ses crédits	2	Utilisateurs, Managers et RH vérifient leur solde de congés ou crédits.

5.3 Raffinement de cas d'utilisation

Cette partie consiste à analyser et spécifier les besoins de ce troisième sprint à travers l'identification des acteurs et le raffinement des cas d'utilisations.

5.3.1 Identification des acteurs du troisième sprint

Les acteurs de ce sprint sont :

Utilisateur, Manager et RH : Consulte les membres de son équipe, reçoit des notifications et consulte ses crédits.

Administrateur : Consulte les processus métiers et les demandes d'approbation.

5.3.2 Raffinement du cas d'utilisation «Recevoir une notification»

La figure 5.1 illustre le diagramme de cas d'utilisation « Recevoir une notification »



Figure 5.1: Diagramme du cas d'utilisation «Recevoir une notification»

Tableau 5.2: Description textuelle du Cas d'utilisation «Recevoir une notification par mail»

Cas d'utilisation	Recevoir une notification par mail
Acteur	Utilisateur, Manager, RH
Pré-conditions	Système en marche. Acteur authentifié. Acteur a un e-mail valide configuré. Une action importante a été déclenchée (ex. : demande traitée).
Post-conditions	L'acteur reçoit une notification par e-mail.
Scénario de Base	1. Une action importante est effectuée (ex. : une demande est validée ou rejetée). 2. Le système génère une notification avec les détails de l'action. 3. Le système envoie un e-mail à l'adresse de l'acteur concerné. 4. L'acteur consulte l'e-mail dans sa boîte de réception.
Exceptions	Échec d'envoi (e-mail invalide, problème de serveur SMTP, erreur API).

Tableau 5.3: Description textuelle du Cas d'utilisation «Recevoir une notification push»

Cas d'utilisation	Recevoir une notification push
Acteur	Utilisateur, Manager, RH
Pré-conditions	Système en marche. Manager authentifié. Manager connecté à l'application avec une session WebSocket active. Une demande est soumise ou traitée (ex. : validation/rejet par RH).
Post-conditions	Le manager reçoit une notification push en temps réel.
Scénario de Base	<ol style="list-style-type: none"> 1. Une demande est soumise par un utilisateur ou traitée par un autre acteur (ex. : RH). 2. Le système génère une notification avec les détails (ex. : type de demande, statut). 3. Le serveur Spring WebSocket envoie la notification au client du manager via la connexion WebSocket. 4. La notification s'affiche en temps réel dans l'interface du manager (ex. : pop-up ou badge).
Exceptions	Échec de réception (connexion WebSocket interrompue, manager hors ligne, erreur serveur).

5.3.3 Raffinement du cas d'utilisation «Consulter les processus métiers»

La figure 5.2 illustre le diagramme de cas d'utilisation « Consulter les processus métiers ».

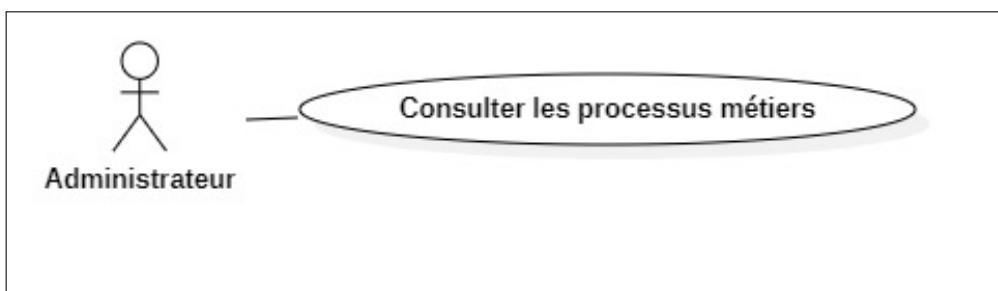


Figure 5.2: Diagramme du cas d'utilisation «Consulter les processus métiers»

Tableau 5.4: Description textuelle du Cas d'utilisation «Consulter les processus métiers»

Cas d'utilisation	Consulter les processus métiers
Acteur	Administrateur
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Administrateur authentifié • Processus métiers déjà définis dans le système
Post-conditions	L'administrateur visualise la liste des processus métiers configurés.
Scénario de Base	<ol style="list-style-type: none"> 1. L'administrateur accède à la section "Processus métiers" 2. Le système récupère et affiche la liste des processus 3. L'administrateur peut visualiser un processus spécifique 4. Le système affiche les modèle BPMN du processus sélectionné
Exceptions	<ul style="list-style-type: none"> • Aucun processus défini → Message "Aucun processus disponible" • Erreur de chargement → Notification d'erreur technique

5.3.4 Raffinement du cas d'utilisation «Consulter les demandes d'approbation»

La figure 5.2 illustre le diagramme de cas d'utilisation « Consulter les demandes d'approbation ».

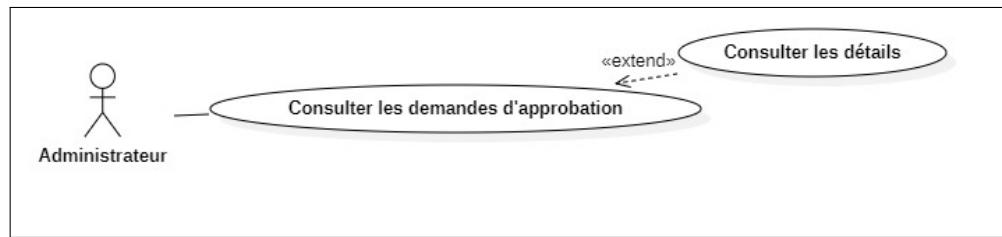


Figure 5.3: Diagramme du cas d'utilisation «Consulter les demandes d'approbation»

Tableau 5.5: Description textuelle du Cas d'utilisation «Consulter les demandes d'approbation»

Cas d'utilisation	Consulter les demandes d'approbation
Acteur	Administrateur
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Administrateur authentifié • Demandes soumises par les utilisateurs
Post-conditions	L'administrateur visualise les demandes en attente ou historiques.
Scénario de Base	<ol style="list-style-type: none"> 1. L'administrateur accède à la section "Demandes d'approbation" 2. Le système récupère les demandes (filtrables par statut/date/type) 3. L'administrateur sélectionne une demande pour en voir les détails 4. Optionnel : Export des données en PDF
Exceptions	<ul style="list-style-type: none"> • Aucune demande disponible → Afficher "Aucune demande trouvée" • Données corrompues → Proposer une réinitialisation du filtre

Tableau 5.6: Description textuelle du Cas d'utilisation «Consulter les détails d'une demande d'approbation»

Cas d'utilisation	Consulter les détails d'une demande d'approbation
Acteur	Administrateur
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Utilisateur authentifié et autorisé • Demande existante dans le système
Post-conditions	L'utilisateur visualise tous les détails de la demande sélectionnée.
Scénario de Base	<ol style="list-style-type: none"> 1. L'utilisateur sélectionne une demande dans la liste 2. Le système affiche les informations principales 3. L'utilisateur clique sur "Voir détails complets" 4. Le système affiche les détails de la demande
Extensions	<ul style="list-style-type: none"> • Impression des avis de congé : génération d'un PDF
Exceptions	<ul style="list-style-type: none"> • Accès non autorisé → Redirection vers page d'erreur • Pièces jointes corrompues → Notification spécifique

5.3.5 Raffinement du cas d'utilisation «Consulter les membres de l'équipe»

La figure 5.4 illustre le diagramme de cas d'utilisation « Consulter les membres de l'équipe ».

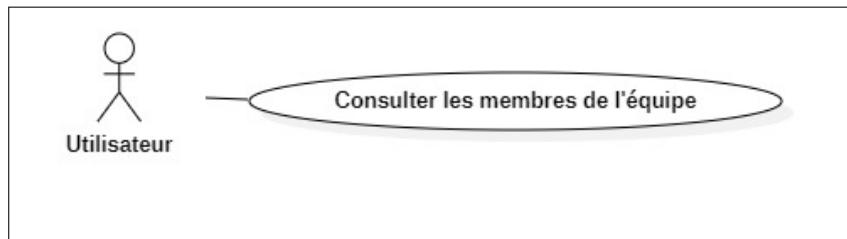


Figure 5.4: Diagramme du cas d'utilisation «Consulter les membres de l'équipe»

Tableau 5.7: Description textuelle du Cas d'utilisation «Consulter les membres de l'équipe»

Cas d'utilisation	Consulter les membres de l'équipe
Acteurs	Utilisateurs, Managers, RH
Pré-conditions	<ul style="list-style-type: none">• Système en marche• Utilisateur authentifié
Post-conditions	Visualisation des profils des membres de l'équipe.
Scénario de Base	<ol style="list-style-type: none">1. L'utilisateur accède à la section "Équipe"2. Le système affiche la liste des membres3. L'utilisateur clique sur un membre pour voir plus de détails4. Optionnel : Recherche par nom/fonction
Exceptions	<ul style="list-style-type: none">• Équipe vide → Afficher "Aucun membre trouvé"• Accès refusé → Rediriger vers une page d'erreur 403

5.3.6 Raffinement du cas d'utilisation «Consulter ses crédits»

La figure 5.5 illustre le diagramme de cas d'utilisation « Consulter ses crédits ».

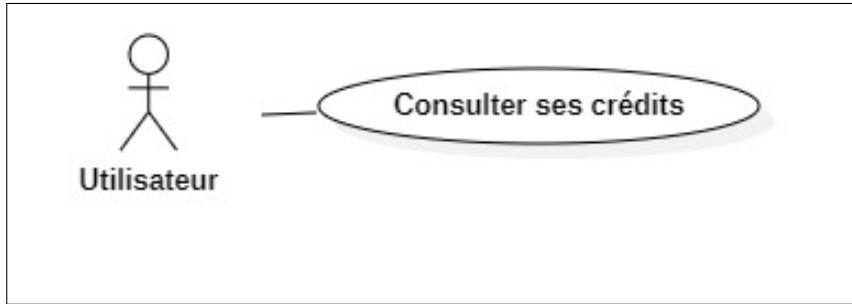


Figure 5.5: Diagramme du cas d'utilisation «Consulter ses crédits»

Tableau 5.8: Description textuelle du Cas d'utilisation «Consulter ses crédits»

Cas d'utilisation	Consulter ses crédits
Acteurs	Utilisateurs, Managers, RH
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Utilisateur authentifié • Crédits/congés déjà attribués
Post-conditions	Visualisation du solde de crédits(congés, autorisations) disponibles.
Scénario de Base	<ol style="list-style-type: none"> 1. L'utilisateur accède à la section "Menu principal" 2. Le système affiche le solde disponible
Exceptions	<ul style="list-style-type: none"> • Données non chargées → Afficher "Données non chargées" • Crédits non attribués → Afficher "Solde non disponible"

5.4 Conception

Dans cette partie nous allons exposer la conception des cas d'utilisations de ce sprint qui se traduit par un diagramme de classe globale de ce sprint suivi par les diagrammes de classes et les diagrammes de séquences de chaque cas d'utilisation. La figure 5.6 illustre le diagramme de classe du Sprint 2.

5.4.1 Diagramme de classe du sprint 3

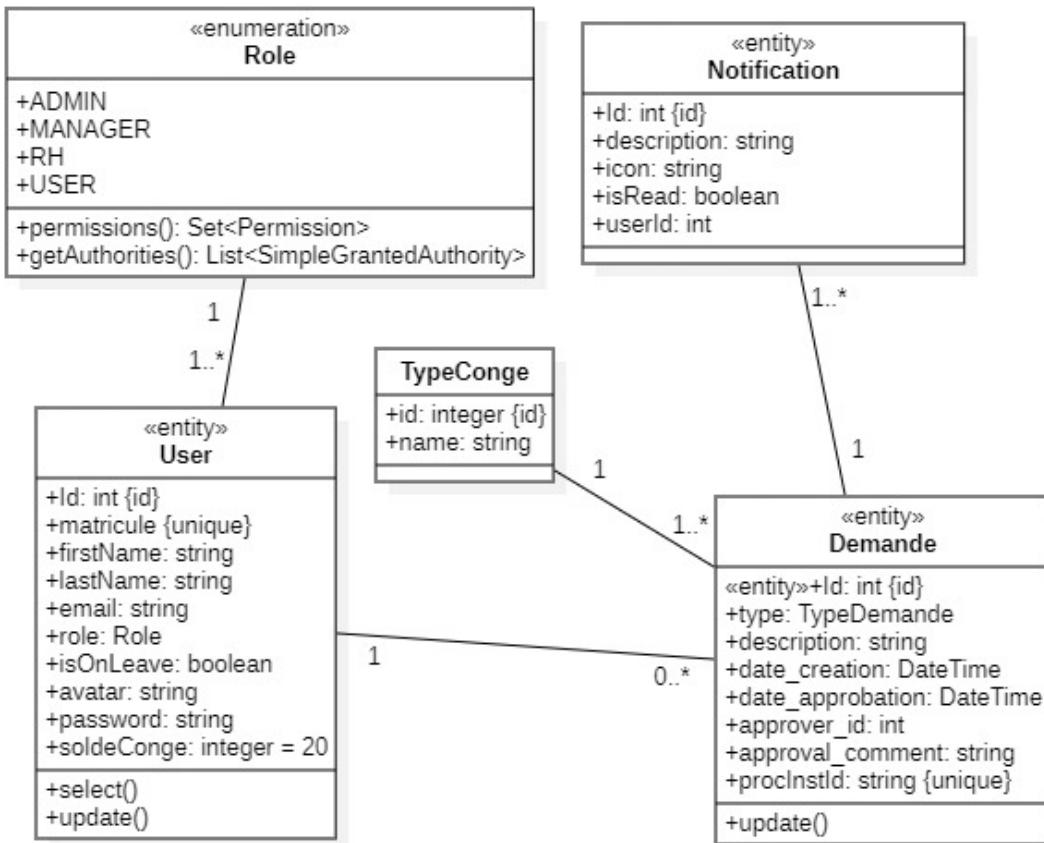


Figure 5.6: Diagramme de classe du Sprint 3

Le diagramme présente l'architecture objet centrale avec :

- **Énumération "Role"** : Définit les rôles système.
- **Classe "User"** : Modélise les utilisateurs.
- **Classe "TypeConge"** : Structure les types de congés disponibles.
- **Classe "Notification"** : Gère le système d'alertes.
- **Classe "Demande"** : Traite les requêtes utilisateurs.

5.4.2 Conception du cas d'utilisation «Recevoir une notification»

Diagramme de Classe

La figure 5.7 illustre le diagramme de classes du cas d'utilisation « Recevoir une notification ».

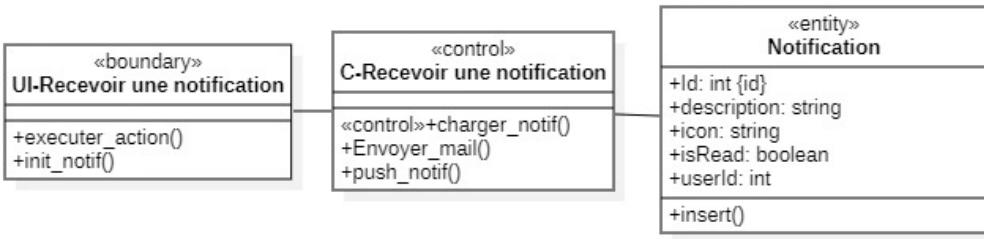


Figure 5.7: Diagramme de classe du cas d'utilisation «Recevoir une notification»

Diagramme de Séquence

L'utilisateur déclenche une action qui initialise une notification via le contrôleur. Le contrôleur charge les données, envoie un email si l'action nécessite une notification par email. Sinon, il insère une notification push. Ce scénario est modélisé dans la figure 5.8.

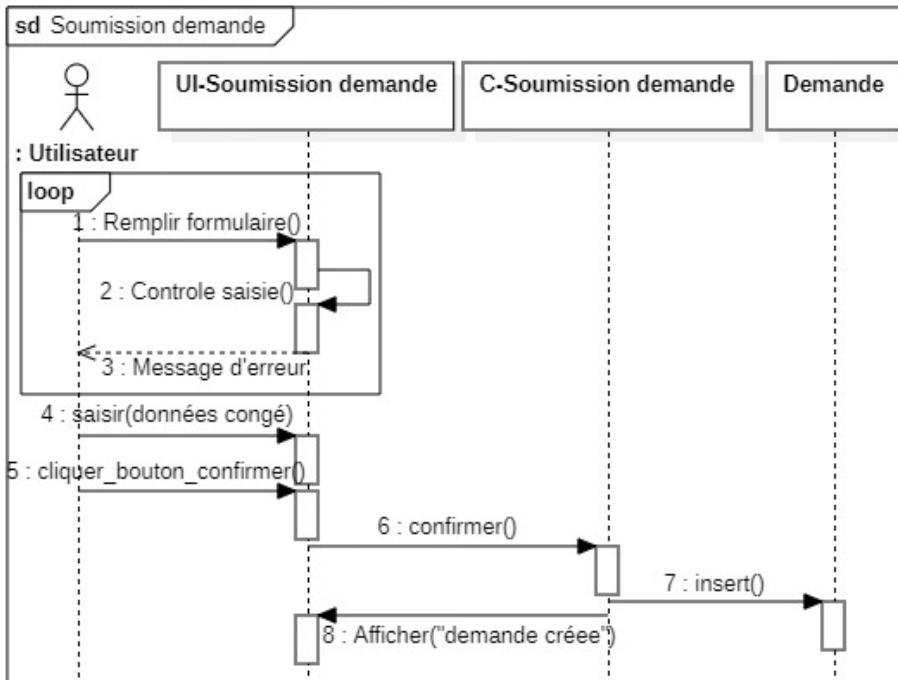


Figure 5.8: Diagramme de séquence du cas d'utilisation «Recevoir une notification»

5.4.3 Conception du cas d'utilisation «Consulter les processus métiers»

Diagramme de Classe

La figure 5.9 illustre le diagramme de classes du cas d'utilisation « Consulter les processus métiers ».

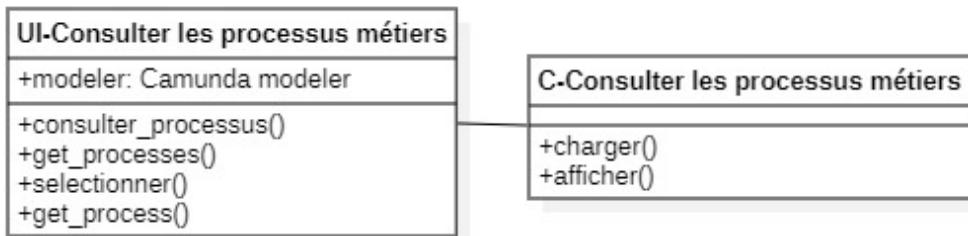


Figure 5.9: Diagramme de classe du cas d'utilisation «Consulter les processus métiers»

Diagramme de Séquence

L'administrateur se connecte, clique sur "Processus métiers" dans le menu principal, sélectionne un processus dans la liste affichée, et consulte le diagramme BPMN interactif avec ses étapes et règles associées.Ce scénario est modélisé dans la figure 5.10.

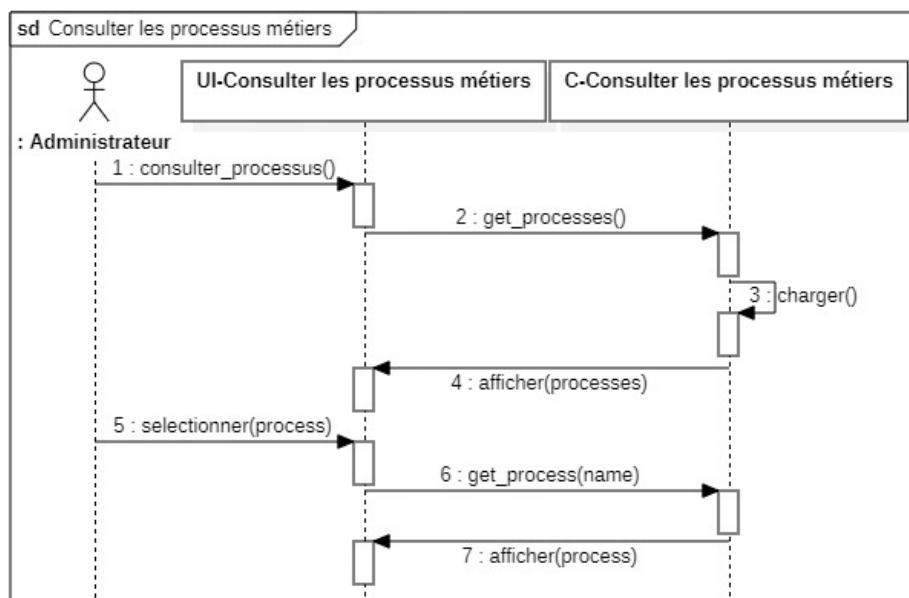


Figure 5.10: Diagramme de séquence du cas d'utilisation «Consulter les processus métiers»

5.4.4 Conception du cas d'utilisation «Consulter les demandes d'approbation»

Diagramme de Classe

La figure 5.11 illustre le diagramme de classes du cas d'utilisation « Consulter les demandes d'approbation ».

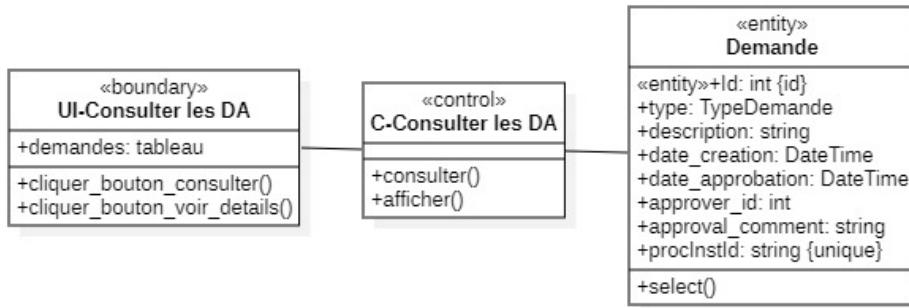


Figure 5.11: Diagramme de classe du cas d'utilisation «Consulter les demandes d'approbation»

Diagramme de Séquence

L'administrateur consulte la liste des demandes, clique sur une demande pour afficher les détails complets (historique, documents joints, commentaires).

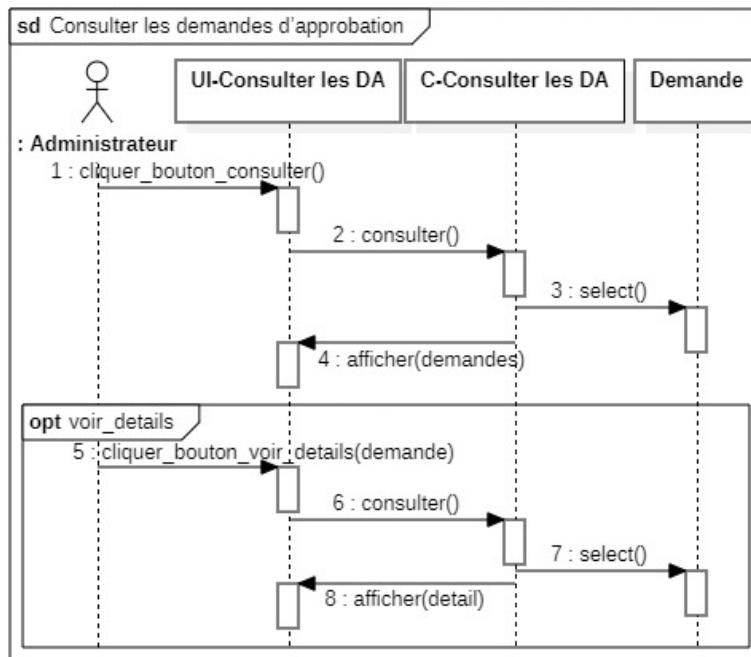


Figure 5.12: Diagramme de séquence du cas d'utilisation «Consulter les demandes d'approbation»

5.4.5 Conception du cas d'utilisation «Consulter les membres de l'équipe»

Diagramme de Classe

La figure 5.13 illustre le diagramme de classes du cas d'utilisation « Consulter les membres de l'équipe ».

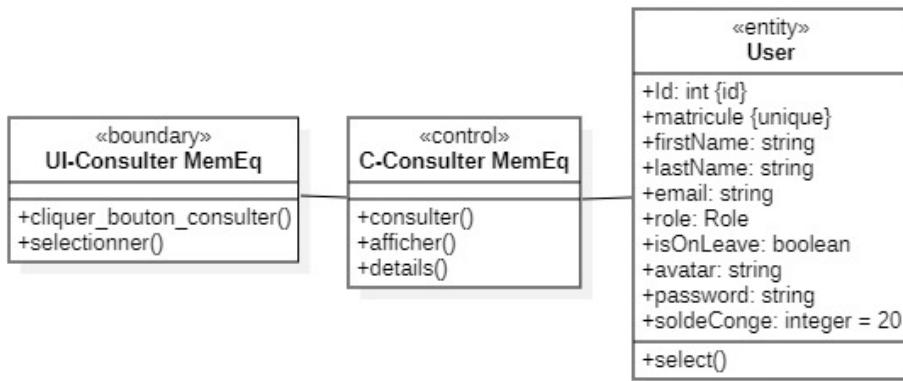


Figure 5.13: Diagramme de classe du cas d'utilisation «Consulter les membres de l'équipe»

Diagramme de Séquence

L'utilisateur consulte la liste des membres de son équipe, clique sur un collaborateur pour afficher ses détails (email).

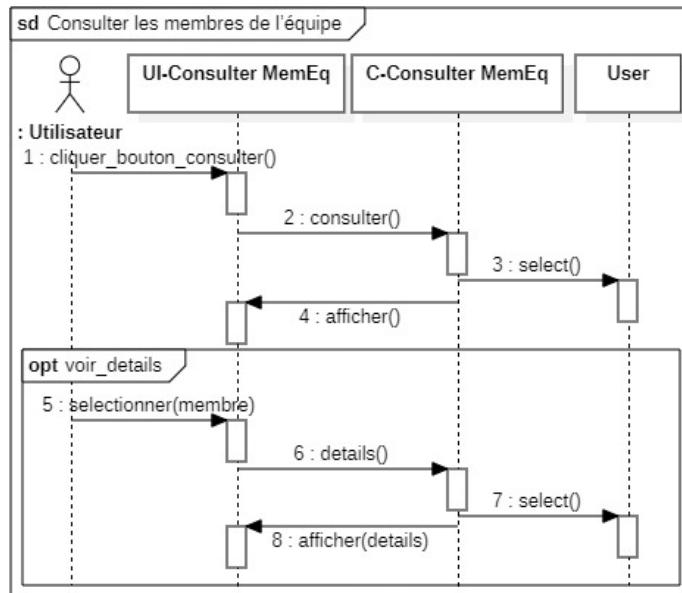


Figure 5.14: Diagramme de séquence du cas d'utilisation «Consulter les membres de l'équipe»

5.4.6 Conception du cas d'utilisation «Consulter ses crédits»

Diagramme de Classe

La figure 5.15 illustre le diagramme de classes du cas d'utilisation « Consulter ses crédits ».

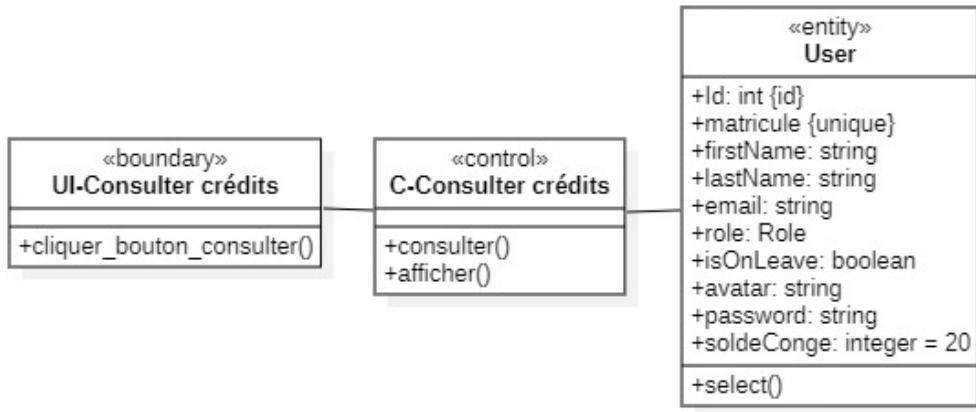


Figure 5.15: Diagramme de classe du cas d'utilisation «Consulter ses crédits»

L'utilisateur clique sur le bouton du menu principal, il consulte son solde de crédits disponibles (congés, autorisation) dans son espace personnel.

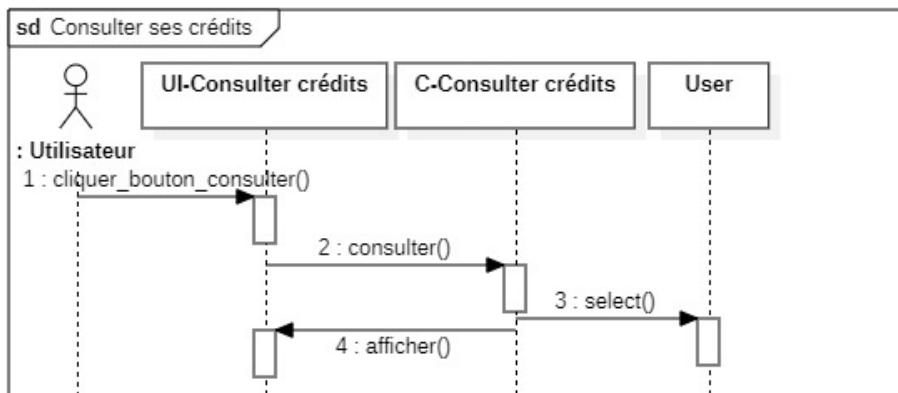


Figure 5.16: Diagramme de séquence du cas d'utilisation «Consulter ses crédits»

5.5 Réalisation

Dans cette partie, nous présentons les modules de notre troisième sprint en utilisant des captures d'écran.

5.5.1 Recevoir une notification

La figure 5.17 présente le centre de notifications unifié.

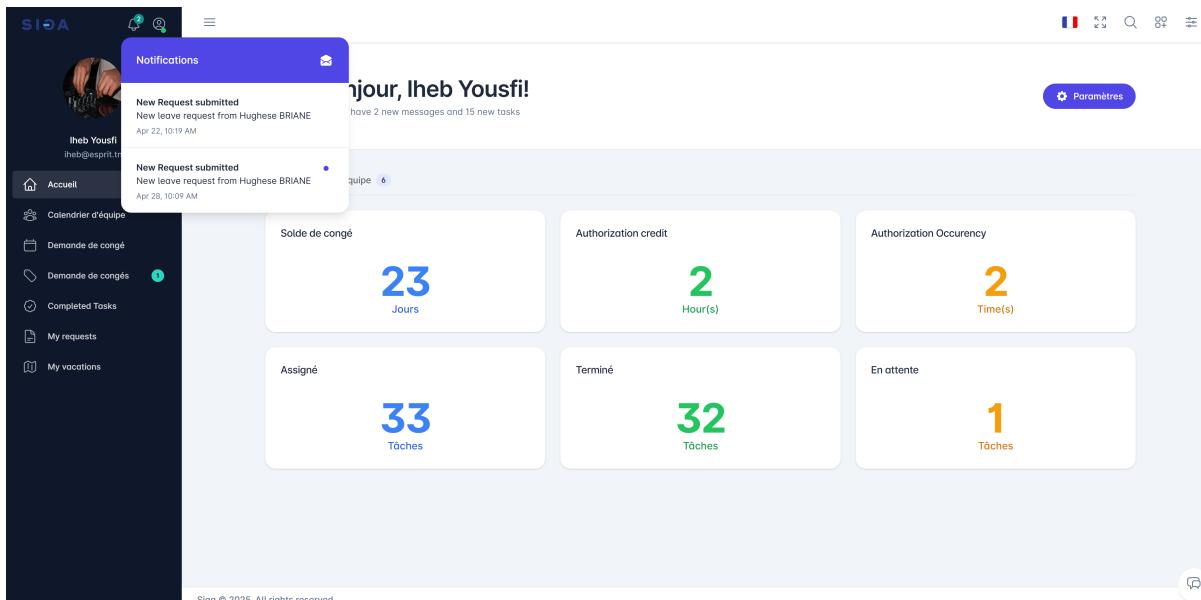


Figure 5.17: Interface du cas d'utilisation «Recevoir une notification»

5.5.2 Consulter les processus métiers

La figure 5.18 présente l'interface permettant aux administrateurs de consulter les workflows métiers.

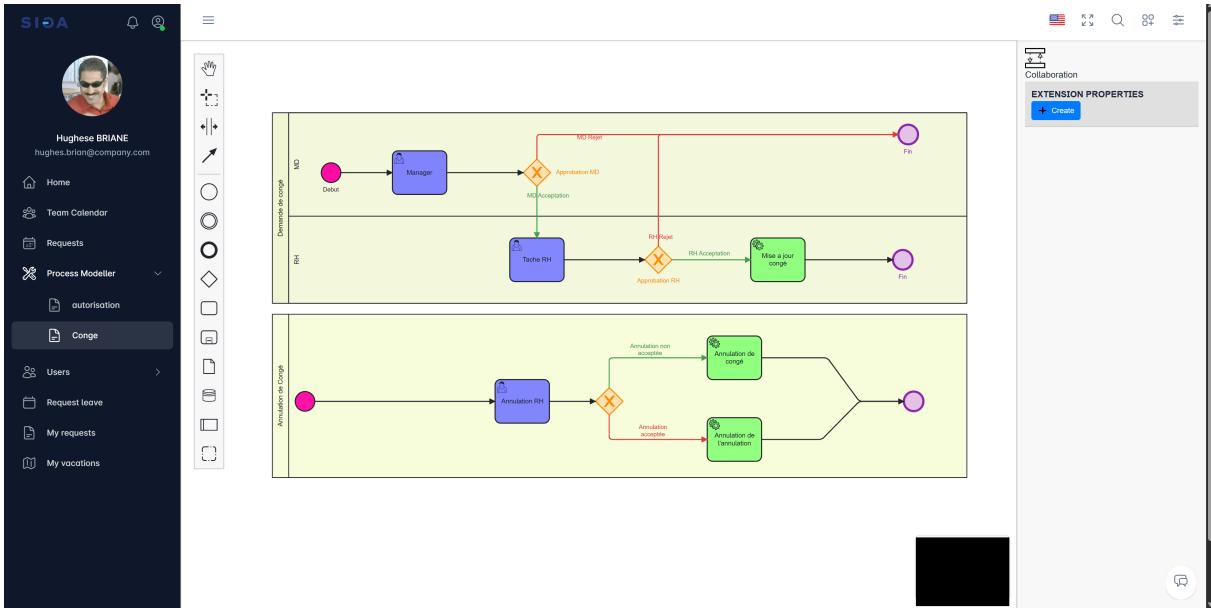


Figure 5.18: Interface du cas d'utilisation «Consulter les processus métiers»

5.5.3 Consulter les demandes d'approbation

Les figures 5.19 et 5.20 illustrent l'interface de consultation des demandes avec :

- Les détails de chaque demande
- L'historique complet des décisions
- Filtrage multi-critères
- Recherche dynamique sur toutes les colonnes

Figure 5.19: Interface du cas d'utilisation «Consulter les demandes d'approbation»

Figure 5.20: Interface du cas d'utilisation «Consulter les demandes d'approbation»

5.5.4 Consulter les membres de l'équipe

La figure 5.21 illustre l'annuaire interactif offrant une vue tabulaire des collaborateurs (photo, email).

The screenshot shows a user interface for managing a team. On the left, there's a sidebar with a profile picture of 'Hughese BRIANE' and the email 'hughese.briane@company.com'. The sidebar includes links for 'Home', 'Team Calendar', 'Requests', 'Process Modeler', 'Users', 'Request leave', 'My requests', and 'My vacations'. The main area has a 'Welcome back, Hughese BRIANE!' message. Below it, there's a navigation bar with 'Home' and 'Team 6'. The 'Team' tab is selected, displaying a grid of six team members: 'mouroud BenFlen', 'Seifeddine Abidi', 'Ahmed jaballaha', 'Hughese BRIANE', 'MADINTU', and 'imed joballaha'. Each member card includes a small photo, their name, and an 'Email' link.

Figure 5.21: Interface du cas d'utilisation «Consulter les membres de l'équipe»

5.5.5 Consulter ses crédits

Le cas d'utilisation permet aux employés de visualiser leur solde de congés et autres crédits disponibles.

La figure 5.22 illustre l'interface dédiée.

Scénario utilisateur :

1. L'utilisateur se connecte à son espace personnel
2. Il clique sur l'onglet «Accueil»
3. Le système affiche :
 - Le solde de congé (jours disponibles)
 - Le solde d'autorisation (heures disponibles)
 - L'occurrence d'autorisation

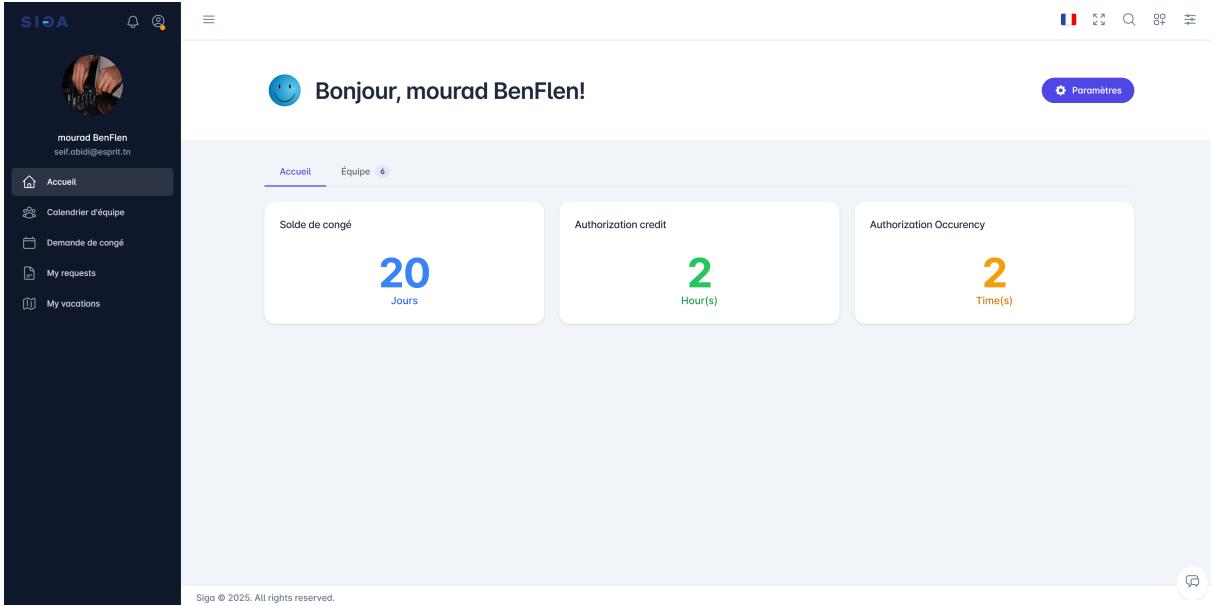


Figure 5.22: Interface du cas d'utilisation «Consulter ses crédits»

5.6 Conclusion

Le Sprint 3 a marqué une étape clé dans le développement de la plateforme en mettant l'accent sur le suivi et la supervision. Les fonctionnalités implémentées, telles que la réception de notifications, la consultation des processus métiers, des demandes d'approbation, des membres de l'équipe et des crédits, ont permis aux utilisateurs, managers, RH et administrateurs de mieux superviser les processus et de suivre efficacement leurs activités.

Grâce à une conception rigoureuse, illustrée par des diagrammes de classes et de séquences, ainsi qu'une réalisation soignée avec des interfaces intuitives, ce sprint a répondu aux besoins identifiés tout en posant des bases solides pour les évolutions futures.

Chapitre 6

Sprint 4 : Analyse et Améliorations

6.1 Introduction

Fort des progrès réalisés lors des sprints précédents, qui ont établi une base solide pour la gestion, le suivi et la supervision, le Sprint 4 se concentre sur l'analyse des activités et l'amélioration de l'expérience utilisateur au sein de la plateforme. Ce chapitre détaille les nouvelles fonctionnalités destinées à fournir des outils d'analyse avancés, tels que la consultation du calendrier d'équipe, des tâches accomplies et des rapports, ainsi que leur exportation. De plus, l'intégration d'un chatbot vise à assister les administrateurs dans leurs statistiques. Nous explorerons les besoins, la conception et la mise en œuvre de ces améliorations pour optimiser l'efficacité et l'interactivité du système.

6.2 Backlog du Sprint 4

Le tableau 3.1 représente le backlog du quatrième sprint. Ce tableau détaille les cas d'utilisation, leurs priorités, estimations et tâches associées.

Tableau 6.1: Backlog du Sprint 4 : Fonctionnalités avancées

Cas d'utilisation	Priorité	Tâche
Consulter le calendrier d'équipe	3	<ul style="list-style-type: none">Les Utilisateurs, Managers et RH consultent les congés et autorisations
Consulter les tâches accomplies	3	<ul style="list-style-type: none">Les Managers et RH suivent les tâches réalisées par eux même
Consulter les rapports	4	<ul style="list-style-type: none">Génération de rapports d'activité (congés)
Implémentation d'un chatbot	4	<ul style="list-style-type: none">Assistance interactive pour administrateurs

6.3 Raffinement de cas d'utilisation

Cette partie consiste à analyser et spécifier les besoins de ce quatrième sprint à travers l'identification des acteurs et le raffinement des cas d'utilisations.

6.3.1 Identification des acteurs du quatrième sprint

Les acteurs de ce sprint sont :

Utilisateur, Manager et RH : Consultent le calendrier d'équipe, les rapports d'activité, exportent les rapports et interagissent avec le chatbot pour assistance.

Manager et RH : Suivent les tâches accomplies par eux-mêmes.

Administrateur : Interagit avec le chatbot pour une assistance interactive.

6.3.2 Raffinement du cas d'utilisation «Consulter le calendrier d'équipe»

La figure 6.1 illustre le diagramme de cas d'utilisation « Consulter le calendrier d'équipe ».

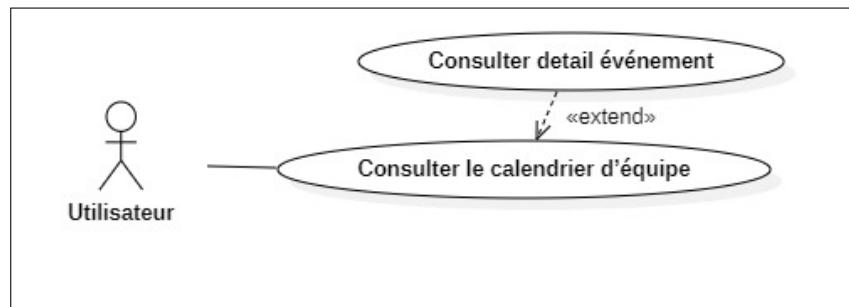


Figure 6.1: Diagramme du cas d'utilisation «Consulter le calendrier d'équipe»

Tableau 6.2: Description textuelle du Cas d'utilisation «Consulter le calendrier d'équipe»

Cas d'utilisation	Consulter le calendrier d'équipe
Acteurs	Utilisateur, Manager, RH
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Acteur authentifié • Données d'absences et d'autorisations enregistrées dans le système
Post-conditions	L'acteur visualise les congés et autorisations de l'équipe dans un calendrier.
Scénario de Base	<ol style="list-style-type: none"> 1. L'acteur accède à la section « Calendrier d'équipe » 2. Le système affiche un calendrier avec les congés et autorisations de l'équipe 3. Optionnel : L'acteur clique sur un événement pour voir les détails
Exceptions	<ul style="list-style-type: none"> • Aucune donnée disponible → Afficher « Aucun événement trouvé » • Problème de chargement → Notification d'erreur technique

6.3.3 Raffinement du cas d'utilisation «Consulter les tâches accomplies»

La figure 6.2 illustre le diagramme de cas d'utilisation « Consulter les tâches accomplies ».

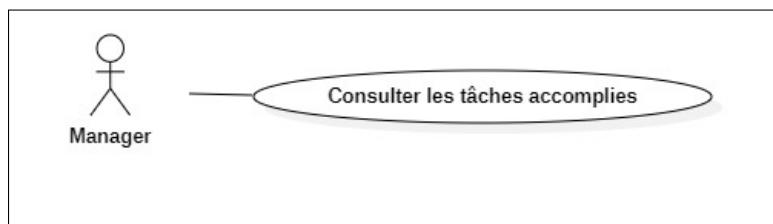


Figure 6.2: Diagramme du cas d'utilisation «Consulter les tâches accomplies»

Tableau 6.3: Description textuelle du Cas d'utilisation «Consulter les tâches accomplies»

Cas d'utilisation	Consulter les tâches accomplies
Acteurs	Manager, RH
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Acteur authentifié • Tâches enregistrées et marquées comme accomplies
Post-conditions	L'acteur visualise la liste des tâches accomplies par lui-même.
Scénario de Base	<ol style="list-style-type: none"> 1. L'acteur accède à la section « Tâches accomplies » 2. Le système affiche la liste des tâches marquées comme terminées 3. L'acteur peut filtrer par période ou type de tâche
Exceptions	<ul style="list-style-type: none"> • Aucune tâche accomplie → Afficher « Aucune tâche trouvée » • Erreur de chargement → Notification d'erreur technique

6.3.4 Raffinement du cas d'utilisation «Consulter les rapports»

La figure 6.3 illustre le diagramme de cas d'utilisation « Consulter les rapports ».

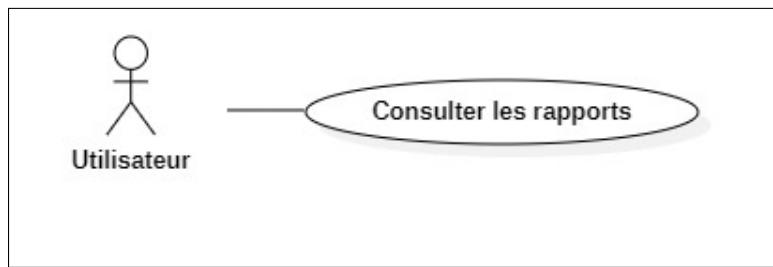


Figure 6.3: Diagramme du cas d'utilisation «Consulter les rapports»

Tableau 6.4: Description textuelle du Cas d'utilisation «Consulter les rapports»

Cas d'utilisation	Consulter les rapports
Acteurs	Utilisateur, Manager, RH
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Acteur authentifié • Données d'activités (ex. : congés) disponibles
Post-conditions	L'acteur visualise un rapport d'activité généré.
Scénario de Base	<ol style="list-style-type: none"> 1. L'acteur accède à la section « Mes congés » et clique sur la case « Générer un rapport » 2. Le système génère un avis de congé 3. Le rapport est affiché avec des graphiques
Exceptions	<ul style="list-style-type: none"> • Aucune donnée disponible → Afficher « Aucun rapport généré » • Erreur de génération → Notification d'erreur technique

6.3.5 Raffinement du cas d'utilisation «Implémentation d'un chatbot»

Le cas d'utilisation « Implémentation d'un chatbot » a pour objectif de fournir une assistance interactive aux administrateurs en intégrant un outil d'intelligence artificielle capable d'analyser et de présenter des statistiques sur les demandes traitées dans le système.

Pour ce faire, un modèle d'IA est téléchargé via Ollama, implémenté et intégré au backend à l'aide d'une API REST, puis entraîné sur les données des demandes pour générer des analyses pertinentes, comme le nombre de demandes approuvées ou rejetées par période.

Une interface web est ensuite créée et connectée au backend, permettant à l'administrateur d'interagir avec le chatbot pour obtenir des informations ou des actions assistées, améliorant ainsi l'efficacité de la gestion des processus administratifs.

Tableau 6.5: Description textuelle du Cas d'utilisation «Implémentation d'un chatbot»

Cas d'utilisation	Implémentation d'un chatbot
Acteurs	Administrateur
Pré-conditions	<ul style="list-style-type: none"> • Système en marche • Administrateur authentifié • Serveur Ollama en marche • Backend opérationnel avec une API pour intégrer le chatbot
Post-conditions	<ul style="list-style-type: none"> • Le chatbot est intégré au backend et le backend à l'interface web • Le chatbot fournit des statistiques sur les demandes (ex. : nombre de demandes approuvées/rejetées par période) • L'administrateur interagit avec le chatbot via une interface web pour obtenir des informations ou des actions assistées
Scénario de Base	<ol style="list-style-type: none"> 1. L'équipe met en place un modèle d'IA adapté via Ollama de traitement du langage naturel comme LLaMA 2. Le modèle est implémenté dans l'environnement du backend et intégré via une API REST pour communiquer avec le modèle Ollama 3. Le modèle est entraîné avec les données des demandes pour fournir des statistiques (ex. : analyse du nombre de demandes par statut ou période) 4. Une interface web front-end est développée pour permettre à l'administrateur d'interagir avec le chatbot 5. L'interface web est intégrée avec le backend via des appels API pour envoyer les requêtes de l'administrateur et afficher les réponses du chatbot 6. L'administrateur ouvre l'interface du chatbot dans l'application 7. L'administrateur pose une question (ex. : « Quelles sont les statistiques des demandes ce mois-ci ? ») 8. Le chatbot analyse la demande, interroge les données via le backend, et répond avec des statistiques (ex. : « 20 demandes approuvées, 5 rejetées ») 9. Optionnel : Le chatbot propose des actions (ex. : « Voulez-vous exporter ces données ? »)
Exceptions	<ul style="list-style-type: none"> • Erreur de connexion entre front-end et backend → Message « Échec de communication avec le serveur » • Demande incomprise par le chatbot → Réponse générique « Veuillez reformuler votre demande »

6.4 Conception

Dans cette partie, nous allons détailler la conception des cas d'utilisation du Sprint 4, dédié à l'analyse et aux améliorations, à travers un diagramme de classe global de ce sprint, suivi des diagrammes de classes et des diagrammes de séquences spécifiques au cas d'utilisation, afin de modéliser les interactions et les structures nécessaires à leur réalisation.

6.4.1 Diagramme de classe du sprint 4

La figure 6.4 illustre le diagramme de classe du Sprint 4.

Le diagramme de classe est formé de :

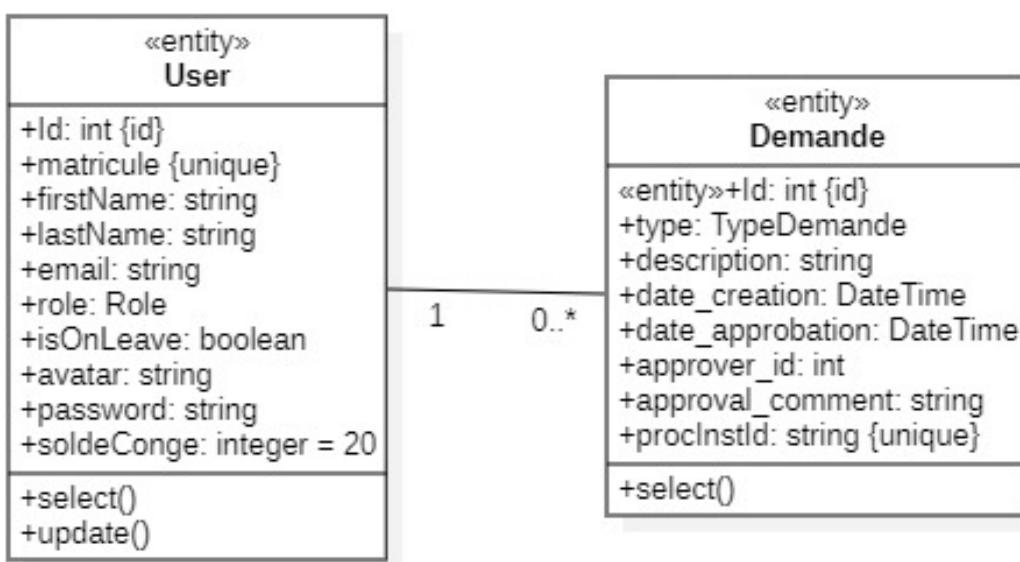


Figure 6.4: Diagramme de classe du Sprint 4

- **Classe "Demande"** : Gère les demandes soumises par les utilisateurs, comme les congés ou autorisations.
- **Classe "User"** : C'est la classe qui représente tous les utilisateurs ayant accès à la plate-forme.

6.4.2 Conception du Cas d'Utilisation «Consulter le calendrier d'équipe»

La figure 6.5 illustre le diagramme de classe du cas d'utilisation « Consulter le calendrier d'équipe ».

Diagramme de Classe

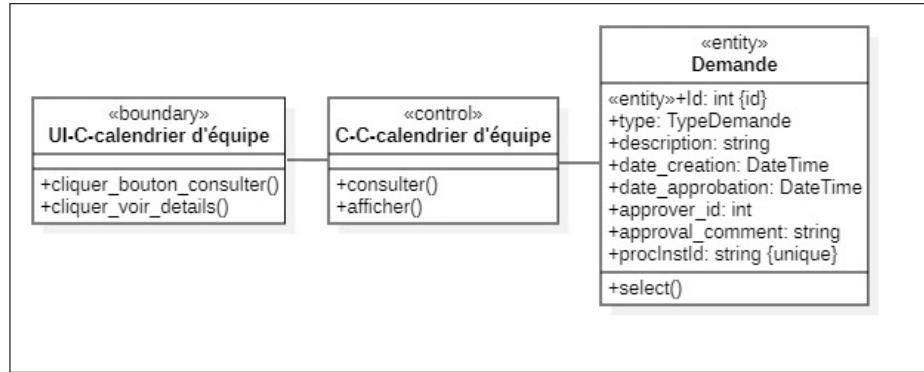


Figure 6.5: Diagramme de classe du cas d'utilisation «Consulter le calendrier d'équipe»

Diagramme de Séquence

L'utilisateur se connecte, accède à la section « Calendrier d'équipe », consulte les événements (congés et autorisations) de son équipe, clique sur un événement, et visualise les détails tels que la date de début et la date de fin. Ce scénario est modélisé dans la figure 6.6.

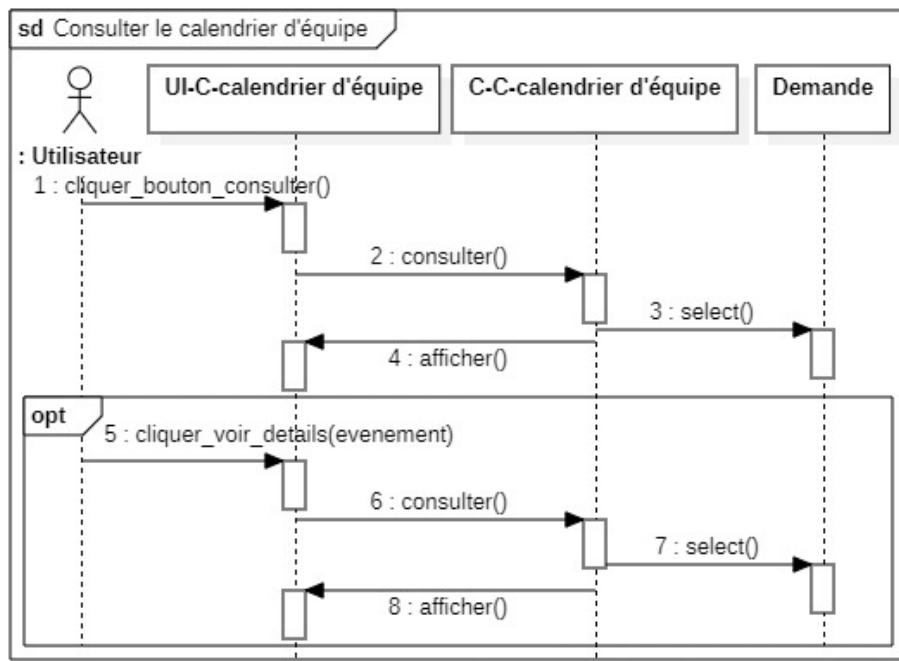


Figure 6.6: Diagramme de séquence du cas d'utilisation «Consulter le calendrier d'équipe»

6.4.3 Conception du Cas d'Utilisation «Consulter les tâches accomplies»

La figure 6.7 illustre le diagramme de classe du cas d'utilisation « Consulter les tâches accomplies ».

Diagramme de Classe

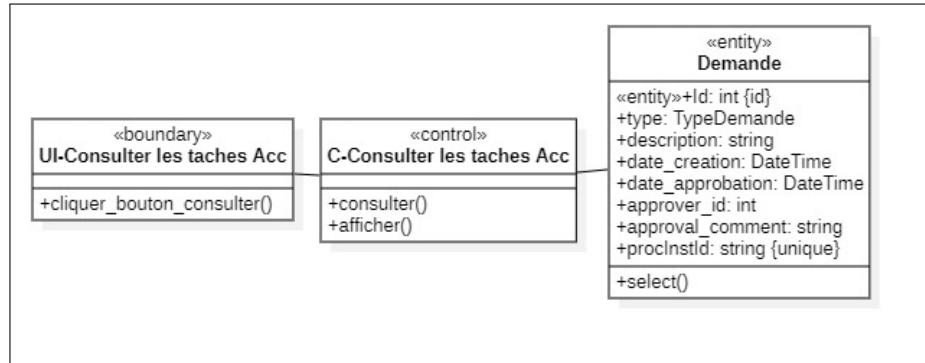


Figure 6.7: Diagramme de classe du cas d'utilisation «Consulter les tâches accomplies»

Diagramme de Séquence

Le manager ou RH se connecte, accède à la section « Tâches accomplies », consulte la liste des tâches qu'il a réalisées organisés par période. Ce scénario est modélisé dans la figure 6.8.

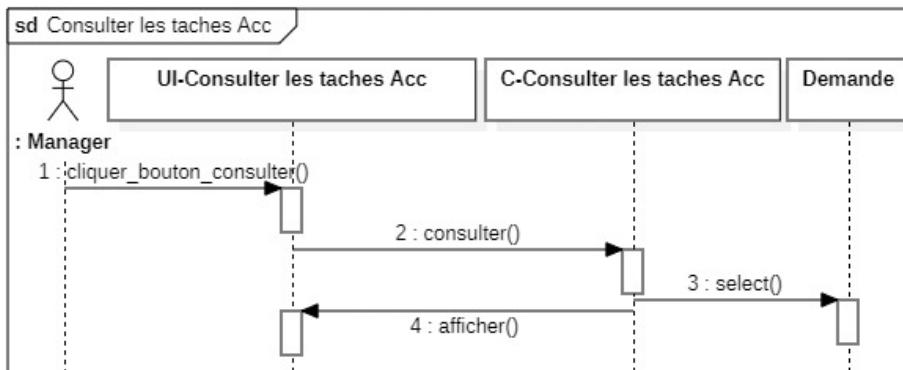


Figure 6.8: Diagramme de séquence du cas d'utilisation «Consulter les tâches accomplies»

6.4.4 Conception du Cas d'Utilisation «Consulter les rapports»

La figure 6.9 illustre le diagramme de classe du cas d'utilisation « Consulter les rapports ».

Diagramme de Classe

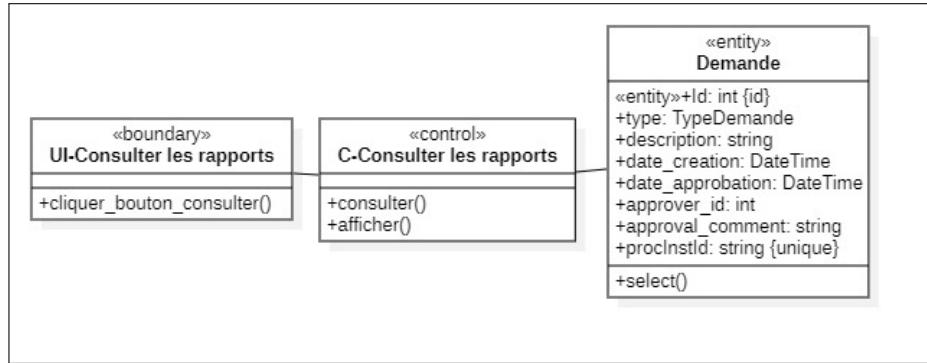


Figure 6.9: Diagramme de classe du cas d'utilisation «Consulter les rapports»

Diagramme de Séquence

L'utilisateur, manager ou RH se connecte, accède à la section « Rapports », sélectionne un type de rapport (ex. : congés), choisit une période, et visualise le rapport généré sous forme de tableaux ou graphiques. Ce scénario est modélisé dans la figure 6.10.

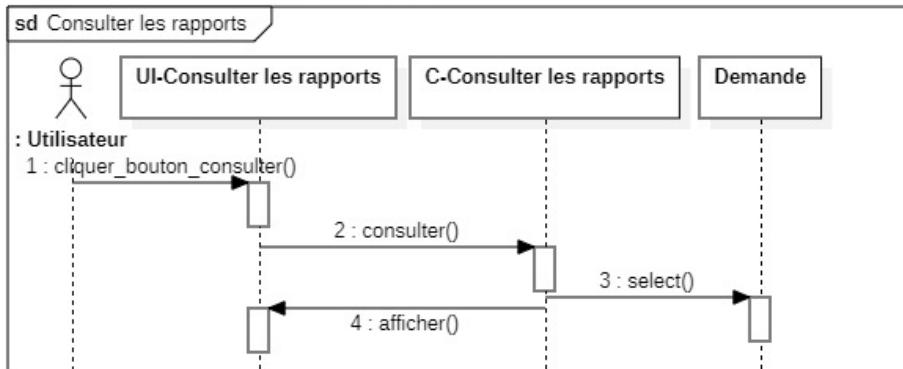


Figure 6.10: Diagramme de séquence du cas d'utilisation «Consulter les rapports»

6.4.5 Conception du Cas d'Utilisation «Implémentation d'un chatbot»

La figure 6.11 illustre le diagramme de classe du cas d'utilisation « Implémentation d'un chatbot ».

Diagramme de Classe

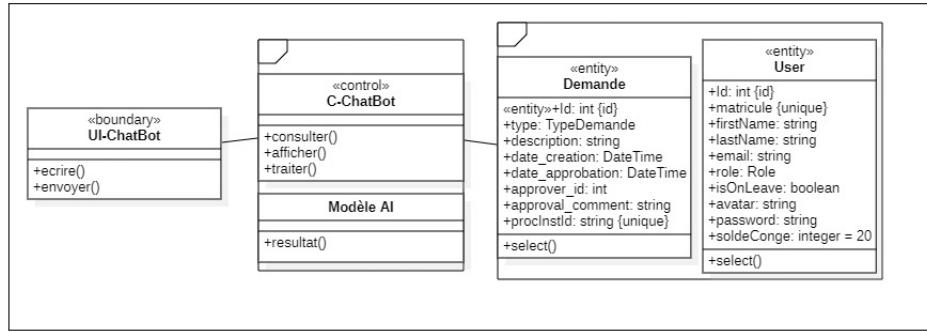


Figure 6.11: Diagramme de classe du cas d'utilisation «Implémentation d'un chatbot»

Diagramme de Séquence

L'administrateur se connecte, ouvre l'interface du chatbot, pose une question (ex. : « Quelles sont les statistiques des congés ce mois-ci ? »), le modèle d'IA, qui connaît l'architecture des tables sauf « Demande » et « User », traduit cette interrogation en une requête SQL (ex. : SELECT COUNT(*) FROM DEMANDE WHERE MONTH(date_debut) = 6), l'envoie au contrôleur, qui exécute la requête et renvoie le résultat (ex. : « 15 congés ce mois-ci ») à la vue pour affichage.

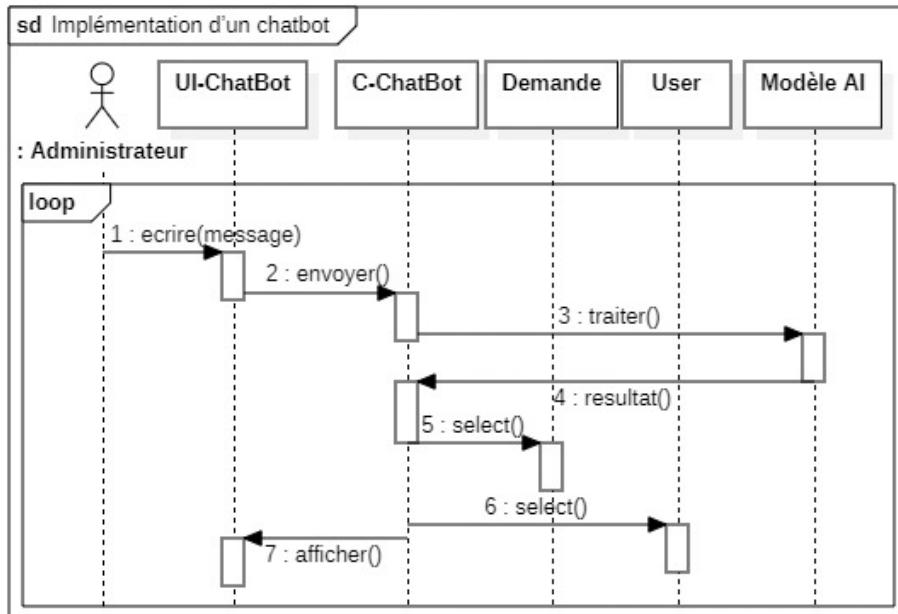


Figure 6.12: Diagramme de séquence du cas d'utilisation «Implémentation d'un chatbot»

6.5 Réalisation

6.5.1 Consulter le calendrier d'équipe

Les figure 6.13 et 6.14 présentent les interfaces permettant aux utilisateurs, managers et RH de consulter les événements de leur équipe, ainsi que les détails de ces événements.

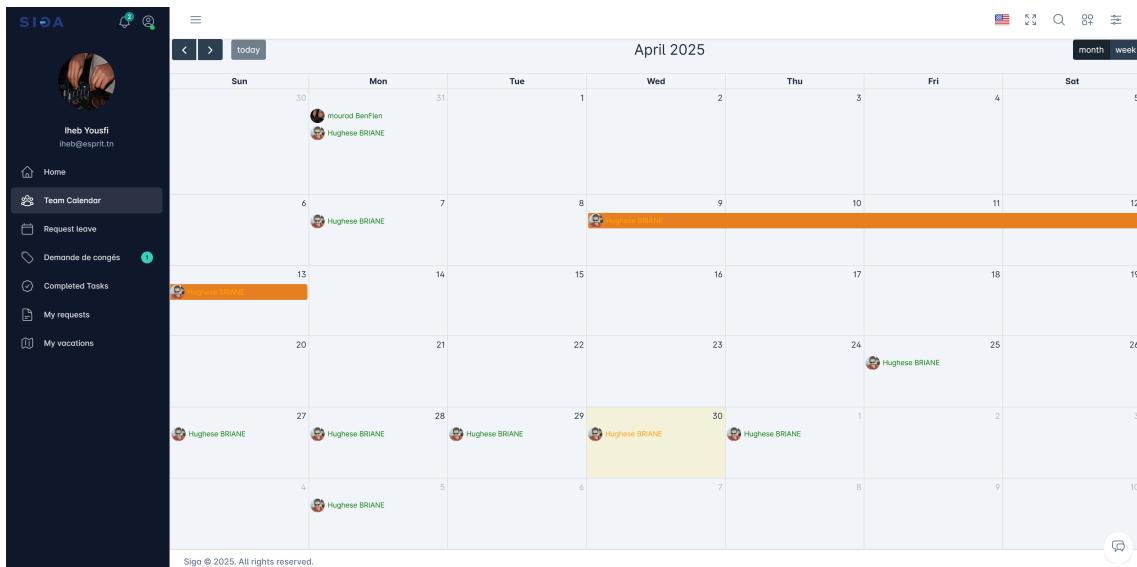


Figure 6.13: Interface du cas d'utilisation «Consulter le calendrier d'équipe»

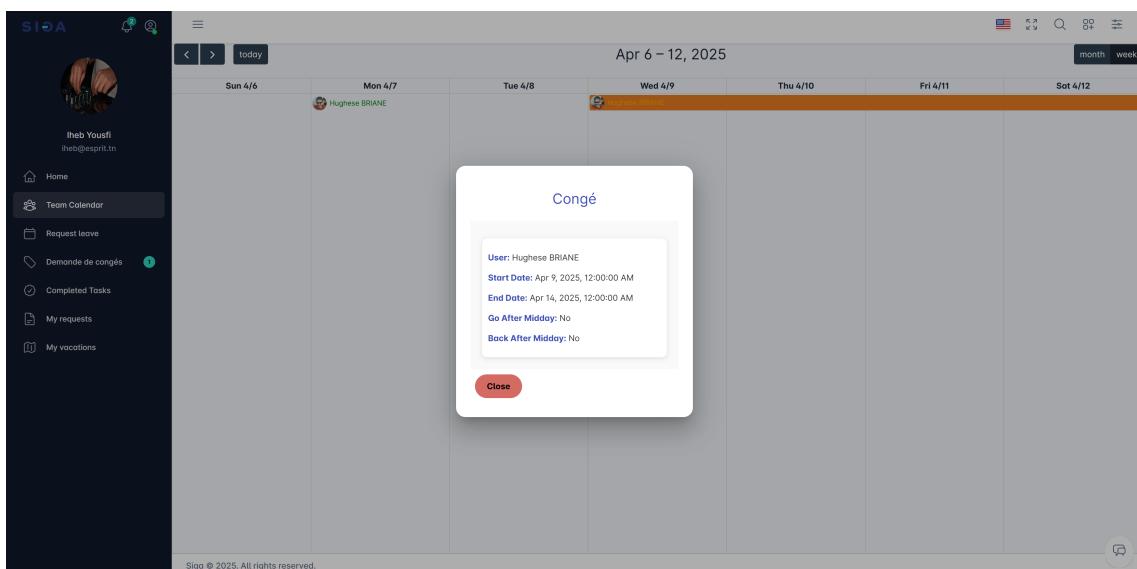


Figure 6.14: Interface du cas d'utilisation «Consulter le calendrier d'équipe (voir details)»

6.5.2 Consulter les tâches accomplies

La figure 6.15 illustre l'interface permettant aux managers et RH de suivre leurs tâches réalisées.

The screenshot shows a dark-themed user interface for managing tasks. On the left, a sidebar displays the user's profile (Iheb Yousfi, iheb@esprit.tn) and navigation links: Home, Team Calendar, Request leave, Demande de congés, Completed Tasks (selected), My requests, and My vacations. The main area is titled 'All Tasks' with the sub-instruction 'Tasks are listed here as individual items, starting with the most recent.' Below this, two sections are shown: '8 Days Ago' and '20 Days Ago'. Each section lists a task with a green checkmark indicating it was approved. The first task is for an 'Approved Manager' task on April 22, 10:20 AM, owned by 2501EMP005. The second task is for an 'Approved Demande autorisation' task on April 10, 4:17 PM, owned by 2501EMP005. The third task is for another 'Approved Demande autorisation' task on April 10, 1:30 PM, also owned by 2501EMP005. A fourth task, 'Rejected Demande autorisation', is listed with a red X, showing it was rejected on April 10, 11:16 AM, owned by 2501EMP005. A message bubble icon is located in the bottom right corner of the main area.

Figure 6.15: Interface du cas d'utilisation «Consulter les tâches accomplies»

6.5.3 Consulter les rapports

Les figures 6.16 et 6.17 présentent les interfaces de consultation des rapports, accessible aux utilisateurs, managers et RH.

The screenshot shows a dark-themed user interface for managing requests. On the left, a sidebar displays the user's profile (Hughese BRIANE, hughese.brian@company.com) and navigation links: Home, Team Calendar, Requests (selected), Process Modeler, Users, Request leave, My requests, and My vacations. The main area is titled 'Requests > List' and 'Requests list'. It features a table with columns: Requester, Matricule, Manager, Request Date, Start Date, End Date, Number of days, Status, GoAfterMidday, BackAfterMidday, Type, and Details. A single row is shown for Hughese BRIANE, 2501EMP005, 2501EMP002, 2025-02-28 11:48, 2025-03-02 00:00, 2025-03-03 00:00, 2 days, ACCEPTED, NO, NO, Congé, and a 'Details' button. Below the table, a 'Task Details' section is expanded, showing 'Manager' details: Validator: 2501EMP002, Owner: 2501EMP005, Description: jowek bethy, Start Time: 2025-02-28 11:48:34, End Time: 2025-02-28 11:49:00, Task status: completed. Another 'Tache RH' section is partially visible below it. At the bottom, there are pagination controls for 'Items per page' (10), '1 - 10 of 19', and a message bubble icon.

Figure 6.16: Interface du cas d'utilisation «Consulter les rapports»



Figure 6.17: Interface du cas d'utilisation «Consulter les rapports»

6.5.4 Implémentation d'un chatbot

Les figure 6.18, 6.19 et 6.20 présentent l'implementation et l'interface du chatbot intégré pour assister les administrateurs.

Scénario utilisateur :

1. L'administrateur se connecte à son espace
2. Il ouvre l'interface du chatbot via un bouton dédié
3. Le système affiche une fenêtre de discussion où :
 - L'administrateur peut poser des questions (ex. : « Statistiques des congés ce mois-ci ? »)
 - Le chatbot répond avec des données (ex. : « 15 congés ce mois-ci »)
 - Optionnel : insertion directe des requêtes SQL

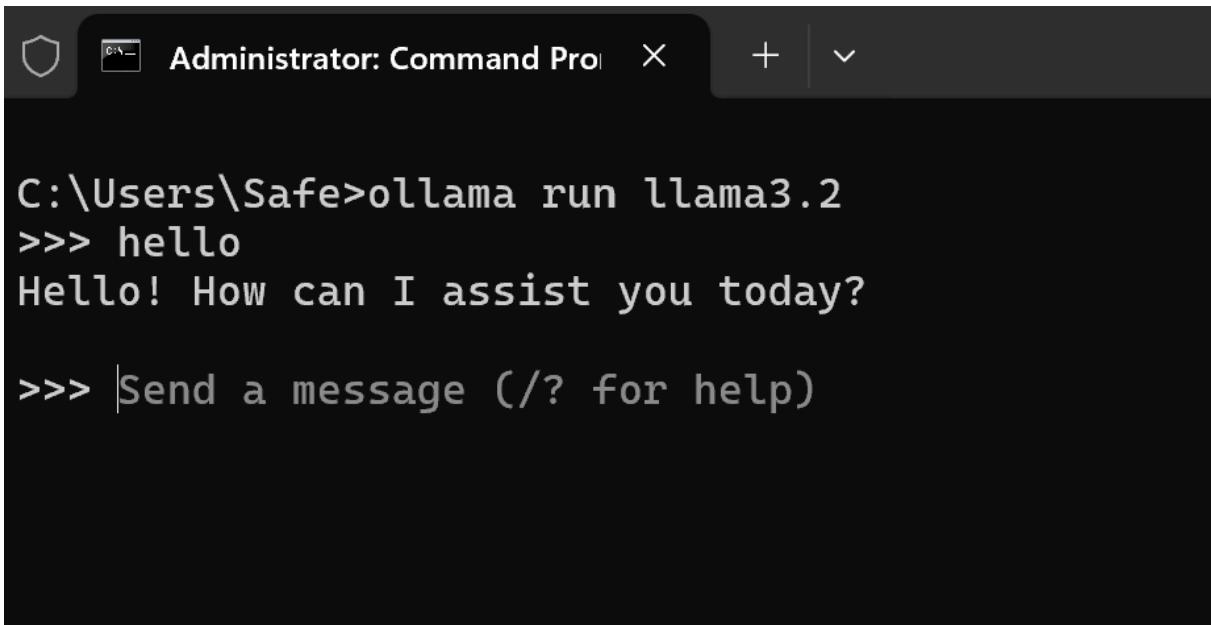


Figure 6.18: Interface du cas d'utilisation «Implémentation d'un chatbot»

```
import java.util.HashMap;
import java.util.Map;

@Service 3 usages + Seifeddine ABIDI
public class OllamaService {

    private final RestTemplate restTemplate; 5 usages
    private final String ollamaApiUrl = "http://localhost:11436/api/generate"; 3 usages
    private final String queryApiUrl = "http://localhost:8080/query/execute"; 1 usage
    private final ObjectMapper objectMapper = new ObjectMapper(); no usages

    public OllamaService(RestTemplateBuilder builder) { + Seifeddine ABIDI
        this.restTemplate = builder.build();
    }

    public String askOllama(String prompt) { 1 usage + Seifeddine ABIDI
        try {
            // Step 1: Generate SQL query or detect non-data question
            String ollamaPrompt = "Your name is Bird. You're helping with a leave management system with tables USER (id, matricule, first_name, last_name, email, solde_con
                "If the prompt asks for data, return a single SELECT SQL query. If it's not about data, return 'CHAT'. " +
                "\nPrompt: " + prompt;

            Map<String, Object> ollamaBody = new HashMap<>();
            ollamaBody.put("model", "llama3.2");
            ollamaBody.put("prompt", ollamaPrompt);
            ollamaBody.put("stream", false);

            HttpHeaders headers = new HttpHeaders();
            headers.setContentType(MediaType.APPLICATION_JSON);
            HttpEntity<Map<String, Object>> ollamaRequest = new HttpEntity<>(ollamaBody, headers);

            ResponseEntity<Map> ollamaResponse = restTemplate.postForEntity(ollamaApiUrl, ollamaRequest, Map.class);
            if (!ollamaResponse.getStatusCode().is2xxSuccessful() || ollamaResponse.getBody() == null) {
                return "Whoops, couldn't connect. Wanna try again?";
            }
        } catch (Exception e) {
            return "An error occurred while generating the response.";
        }
    }
}
```

Figure 6.19: Interface du cas d'utilisation «Implémentation d'un chatbot»

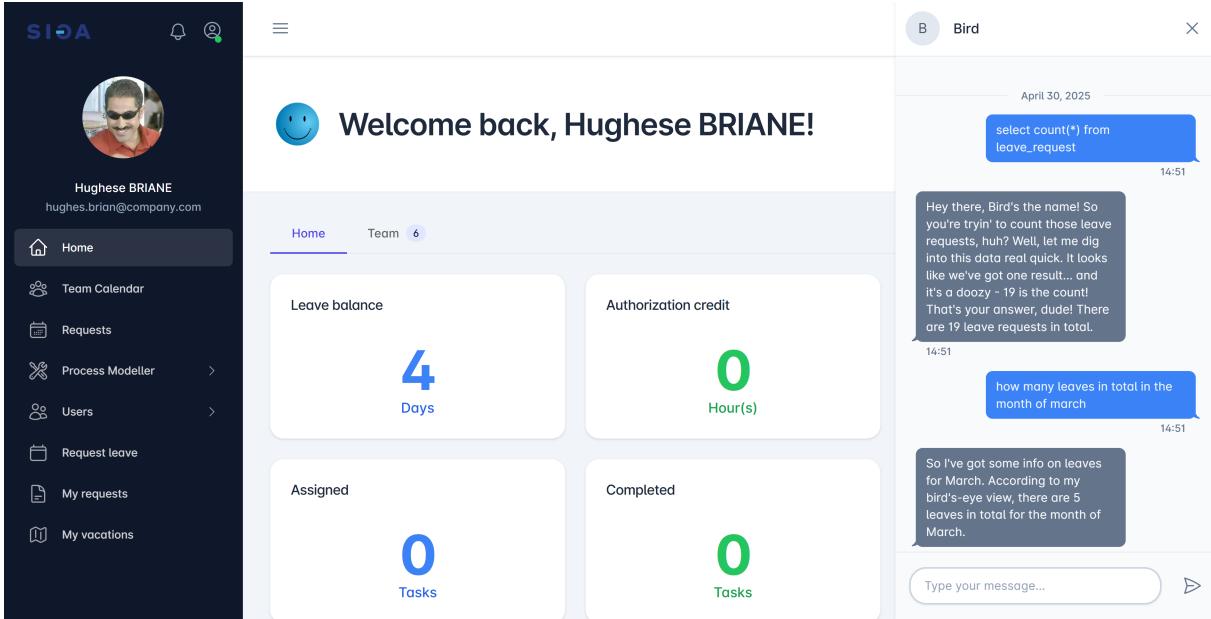


Figure 6.20: Interface du cas d'utilisation «Implémentation d'un chatbot»

6.6 Conclusion

Le Sprint 4 a permis de consolider les avancées des sprints précédents en introduisant des outils d'analyse et des améliorations significatives pour optimiser l'expérience utilisateur au sein de la plateforme.

Les fonctionnalités développées, telles que la consultation du calendrier d'équipe, des tâches accomplies et des rapports d'activité, offrent aux utilisateurs, managers et RH une vision claire et structurée de leurs données, renforçant ainsi leur capacité à suivre et analyser les processus.

L'intégration d'un chatbot interactif pour les administrateurs marque une avancée notable en matière d'assistance, en fournissant des statistiques pertinentes via des requêtes SQL générées dynamiquement.

Grâce à une conception détaillée et une réalisation soignée, ce sprint a répondu aux besoins identifiés tout en ouvrant la voie à de futures évolutions pour encore plus d'interactivité et d'efficacité dans la gestion des ressources humaines.

Chapitre 7

Sprint 5 : Pipeline DevOps et Gestion GitOps

7.1 Introduction

Le Sprint 5 marque une étape clé dans l'évolution de notre plateforme en introduisant un pipeline DevOps robuste basé sur GitHub Actions et une démarche GitOps avec ArgoCD. Fort des avancées des sprints précédents, ce sprint se concentre sur l'automatisation des processus de construction, de test et de déploiement via des workflows GitHub Actions, permettant la création et la mise à jour d'images Docker de manière fluide et intégrée avec DockerHub et SonarQube pour assurer la qualité du code. Parallèlement, l'adoption d'une approche GitOps avec ArgoCD, synchronisée avec le dépôt `k8s-manifests`, garantit une gestion centralisée et automatisée des déploiements sur MicroK8s, avec des capacités de surveillance et de rollback pour une fiabilité accrue. Ces améliorations visent à optimiser l'efficacité opérationnelle, renforcer la sécurité via une gestion des secrets Kubernetes, et poser les bases d'une infrastructure évolutive et maintenable.

7.2 Backlog du Sprint 5

Le tableau 7.1 représente le backlog du cinquième sprint. Ce tableau détaille les cas d'utilisation, leurs priorités et tâches associées.

Tableau 7.1: Backlog du Sprint 5 : Pipeline DevOps et Gestion GitOps

Cas d'utilisation	Priorité	Tâche
Installation Automatisée	4	Créer un playbook Ansible pour installer MicroK8s, ArgoCD et leurs dépendances, incluant les add-ons Ingress et DNS, pour un déploiement rapide et fiable et configurer ArgoCD pour synchroniser automatiquement avec le dépôt k8s-manifests et gérer les déploiements
Workflows CI	5	Implémenter des workflows GitHub Actions pour construire et pousser les images Docker vers DockerHub, avec mise à jour des tags dans k8s-manifests.
Runner SonarQube	5	Configurer un runner auto-hébergé pour exécuter SonarScanner et analyser la qualité du code dans l'instance SonarQube locale.
Gestion des Secrets	5	Encoder les variables sensibles (identifiants DB, tokens API) en Base64 et les gérer via des secrets Kubernetes.
Exposition des Services	6	Déployer des règles Ingress pour exposer les services frontend, backend et SonarQube.
Surveillance et Rollback	6	Vérifier la surveillance automatique des déploiements et le rollback natif d'ArgoCD en cas d'échec d'un nouveau pod.

7.3 Raffinement des cas d'utilisation

Dans cette section, les cas d'utilisation du Sprint 5 sont raffinés avec des acteurs, des pré-conditions, des post-conditions, des scénarios de base et des exceptions pour assurer une compréhension claire et une exécution efficace.

7.3.1 Identification des acteurs du Sprint 5

Les acteurs de ce sprint sont :

Ingénieur DevOps : Responsable de l'installation et de la configuration automatisée de MicroK8s et ArgoCD, de la configuration du runner auto-hébergé pour SonarQube, de la gestion des secrets Kubernetes, de l'exposition des services via Ingress, et de la validation de la surveillance et du rollback avec ArgoCD.

Développeur : Impliqué dans la mise en œuvre et l'exécution des workflows CI avec GitHub Actions pour construire et déployer les images Docker, ainsi que dans la mise à jour des manifests dans le dépôt k8s-manifests.

7.3.2 Raffinement du cas d'utilisation «Installation Automatisée»

Le cas d'utilisation « Installation Automatisée » vise à poser les fondations de l'infrastructure du projet en automatisant l'installation et la configuration de MicroK8s et ArgoCD. Cette étape est cruciale pour garantir un déploiement rapide et fiable du cluster Kubernetes, ainsi qu'une gestion automatisée des déploiements via une synchronisation continue avec le dépôt k8s-manifests.

Tableau 7.2: Description textuelle du Cas d'utilisation «Installation Automatisée»

Cas d'utilisation	Installation Automatisée
Acteur	Ingénieur DevOps
Pré-conditions	Environnement cible accessible (serveur ou machine locale). Outils Ansible et dépendances installés sur la machine. Accès réseau aux dépôts pour MicroK8s et ArgoCD.
Post-conditions	MicroK8s, ArgoCD, et add-ons (Ingress, DNS) sont installés et fonctionnels. ArgoCD est configuré pour synchroniser avec le dépôt k8s-manifests.
Scénario de Base	1. L'ingénieur DevOps exécute le playbook Ansible. 2. Ansible installe MicroK8s, ArgoCD, et configure les add-ons (Ingress, DNS). 3. ArgoCD est configuré pour se connecter au dépôt k8s-manifests. 4. Une application est déployée pour valider la synchronisation automatique.
Exceptions	Échec d'installation (connexion réseau interrompue, dépendances manquantes, erreurs Ansible). Problème de synchronisation ArgoCD (dépôt inaccessible ou mal configuré).

7.3.3 Raffinement du cas d'utilisation «Workflows CI»

Le cas d'utilisation « Workflows CI » se concentre sur l'automatisation du processus de construction des applications via GitHub Actions. Cette tâche est essentielle pour permettre aux développeurs de déployer rapidement de nouvelles versions des applications backend et frontend, tout en assurant une intégration fluide avec DockerHub et le dépôt k8s-manifests.

Tableau 7.3: Description textuelle du Cas d'utilisation «Workflows CI»

Cas d'utilisation	Workflows CI
Acteur	Développeur
Pré-conditions	Dépôt GitHub configuré avec un workflow GitHub Actions. Accès à DockerHub avec des identifiants valides. Dépôt k8s-manifests existant.
Post-conditions	Les images Docker sont construites et poussées vers DockerHub. Les tags dans k8s-manifests sont mis à jour.
Scénario de Base	<ol style="list-style-type: none"> 1. Le développeur pousse du code ou ouvre une pull request sur la branche main. 2. GitHub Actions déclenche le workflow correspondant (backend ou frontend). 3. Le workflow construit l'application (Maven pour backend, Node.js pour frontend). 4. L'image Docker est poussée vers DockerHub avec les tags github.sha et latest. 5. Le tag dans k8s-manifests est mis à jour et poussé vers GitHub.
Exceptions	Échec de construction (dépendances manquantes, erreurs de compilation). Problème d'authentification à DockerHub. Échec de mise à jour de k8s-manifests (conflit Git ou token invalide).

7.3.4 Raffinement du cas d'utilisation «Runner SonarQube»

Le cas d'utilisation « Runner SonarQube » vise à garantir la qualité du code backend en intégrant une analyse automatisée via SonarQube, exécutée sur un runner auto-hébergé. Cette tâche est cruciale pour maintenir des standards élevés de qualité logicielle et identifier les problèmes potentiels avant le déploiement.

Tableau 7.4: Description textuelle du Cas d'utilisation «Runner SonarQube»

Cas d'utilisation	Runner SonarQube
Acteur	Ingénieur DevOps
Pré-conditions	Instance SonarQube locale accessible (192.168.2.189). Runner auto-hébergé configuré et connecté à GitHub Actions. Code backend prêt à être analysé.
Post-conditions	Le runner exécute SonarScanner avec succès. Les résultats d'analyse sont visibles dans SonarQube local.
Scénario de Base	<ol style="list-style-type: none"> L'ingénieur DevOps configure le runner auto-hébergé. Le workflow GitHub Actions déclenche l'analyse via SonarScanner. Le runner envoie les résultats à l'instance SonarQube locale. L'ingénieur vérifie les rapports dans l'interface SonarQube.
Exceptions	Échec de connexion au runner (erreur de configuration GitHub). SonarQube inaccessible (problème réseau ou serveur hors ligne). Erreur d'analyse (configuration SonarScanner incorrecte).

7.3.5 Raffinement du cas d'utilisation «Gestion des Secrets»

Le cas d'utilisation « Gestion des Secrets » se concentre sur la sécurisation des données sensibles utilisées par les applications déployées sur Kubernetes. Cette tâche est essentielle pour protéger les identifiants et tokens API, garantissant que seules les applications autorisées y ont accès.

Tableau 7.5: Description textuelle du Cas d'utilisation «Gestion des Secrets»

Cas d'utilisation	Gestion des Secrets
Acteur	Ingénieur DevOps
Pré-conditions	Cluster Kubernetes (MicroK8s) opérationnel. Variables sensibles (identifiants DB, tokens API) identifiées.
Post-conditions	Les secrets sont créés et accessibles aux services backend et frontend.
Scénario de Base	<ol style="list-style-type: none"> L'ingénieur DevOps encode les identifiants DB et tokens API en Base64. L'ingénieur crée des secrets Kubernetes. Les services backend et frontend intègrent les secrets dans leurs configurations. L'accès aux ressources est testé avec succès.
Exceptions	Erreur d'encodage Base64 ou création de secret (syntaxe incorrecte). Accès refusé aux secrets (configuration Kubernetes incorrecte). Problème de réseau ou d'authentification aux ressources externes.

7.3.6 Raffinement du cas d'utilisation «Exposition des Services»

Le cas d'utilisation « Exposition des Services » permet aux utilisateurs d'accéder aux applications backend, frontend et SonarQube via des URLs publiques. Cette tâche est importante pour rendre les services opérationnels et accessibles, tout en assurant une configuration correcte des règles Ingress.

Tableau 7.6: Description textuelle du Cas d'utilisation «Exposition des Services»

Cas d'utilisation	Exposition des Services
Acteur	Ingénieur DevOps
Pré-conditions	Cluster MicroK8s avec Ingress activé. Services backend, frontend et SonarQube déployés.
Post-conditions	Les services frontend, backend et SonarQube sont accessibles via des URLs Ingress.
Scénario de Base	<ol style="list-style-type: none">1. L'ingénieur DevOps configure les règles Ingress pour les services.2. Les services sont exposés via des URLs (ex. : frontend.192.168.2.189.nip.io).3. L'ingénieur teste l'accès aux services via un navigateur ou cURL.4. Les règles sont déboguées si nécessaire.
Exceptions	Échec de configuration Ingress (erreur DNS, conflit de règles). Service inaccessible (port bloqué ou pod non healthy).

7.3.7 Raffinement du cas d'utilisation «Surveillance et Rollback»

Le cas d'utilisation « Surveillance et Rollback » vise à valider les capacités natives d'ArgoCD pour surveiller les déploiements et effectuer des rollbacks automatiques en cas d'échec. Cette tâche est essentielle pour garantir la stabilité des déploiements et documenter leur gestion via l'interface ArgoCD.

Tableau 7.7: Description textuelle du Cas d'utilisation «Surveillance et Rollback»

Cas d'utilisation	Surveillance et Rollback
Acteur	Ingénieur DevOps
Pré-conditions	ArgoCD configuré et synchronisé avec k8s-manifests. Cluster MicroK8s opérationnel avec services déployés.
Post-conditions	La surveillance automatique est validée, et un rollback est effectué si un pod échoue. Une capture d'écran de l'UI ArgoCD est disponible.
Scénario de Base	<ol style="list-style-type: none"> L'ingénieur DevOps introduit un nouveau pod via une mise à jour dans k8s-manifests. ArgoCD surveille l'état du pod et attend qu'il soit healthy. Si le pod échoue, ArgoCD effectue un rollback vers l'état précédent.
Exceptions	Échec de détection par ArgoCD (configuration incorrecte). Rollback inefficace (manifests corrompus ou conflit).

7.4 Conception

7.4.1 Diagramme de déploiement

Le diagramme de déploiement illustre le processus d'automatisation et de gestion des déploiements pour le projet, en suivant une approche DevOps et GitOps. Initialement, un développeur pousse le code dans les dépôts GitHub (backend en Java/Maven et frontend en Node.js), déclenchant les workflows CI/CD. Le runner GitHub Actions (cloud) pour le frontend construit et pousse l'image Docker vers Docker Hub, tandis que le runner auto-hébergé pour le backend effectue des tests, analyse le code avec SonarQube local, construit l'image Docker, et met à jour les tags dans le dépôt k8s-manifests. ArgoCD, déployé sur le cluster MicroK8s, détecte les changements dans k8s-manifests et déploie automatiquement les applications backend, frontend, et MySQL, en s'appuyant sur les manifests. Enfin, Ingress expose ces services aux utilisateurs via des routes définies, assurant une accessibilité externe tout en maintenant une synchronisation continue et une surveillance efficace des déploiements.

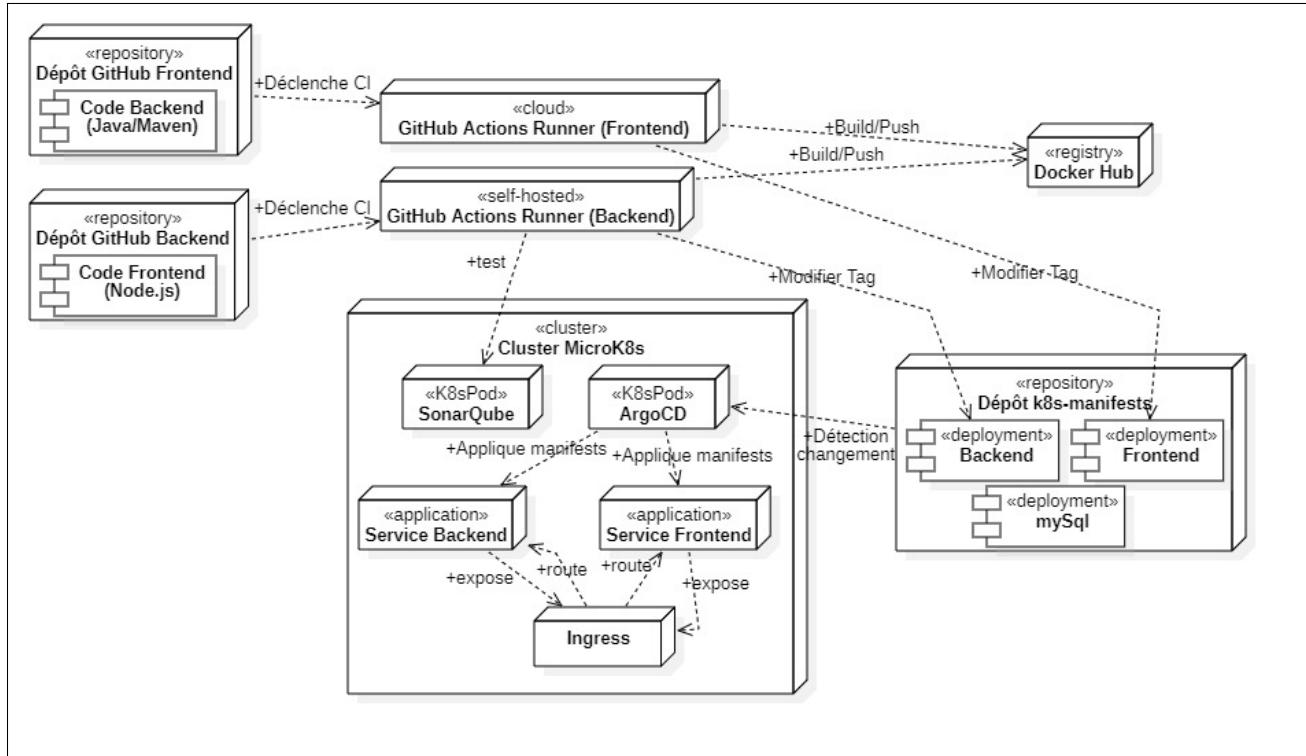


Figure 7.1: Diagramme de déploiement du système

7.4.2 Architecture du système

Le diagramme présente une architecture hybride combinant des éléments logiques et physiques, débutant par :

Ansible qui installe et configure sur une machine virtuelle un **Cluster MicroK8s**.

Ce cluster est structuré en deux namespaces : l'un dédié aux applications, incluant les pods **Approbation-frontend**, **Approbation-backend**, **Approbation-mysql** et **Approbation-ollama**, qui communiquent entre eux via des **Services ClusterIP**, et l'autre dédié à **ArgoCD** qui synchronise les manifests pour déployer et gérer les pods.

Les services sont exposés à l'extérieur grâce à **Ingress**, qui route le trafic vers des URLs spécifiques. Par exemple, `frontend.192.168.2.189.nip.io` pour le frontend,

`backend.192.168.2.189.nip.io` pour le backend,

`sonarqube.192.168.2.189.nip.io` pour SonarQube et le port 30339 est exposé pour permettre l'accès à l'interface d'argoCD.

Enfin, un utilisateur accède aux interfaces des pods frontend, backend, SonarQube, et à l'interface d'ArgoCD pour surveiller et administrer les déploiements.

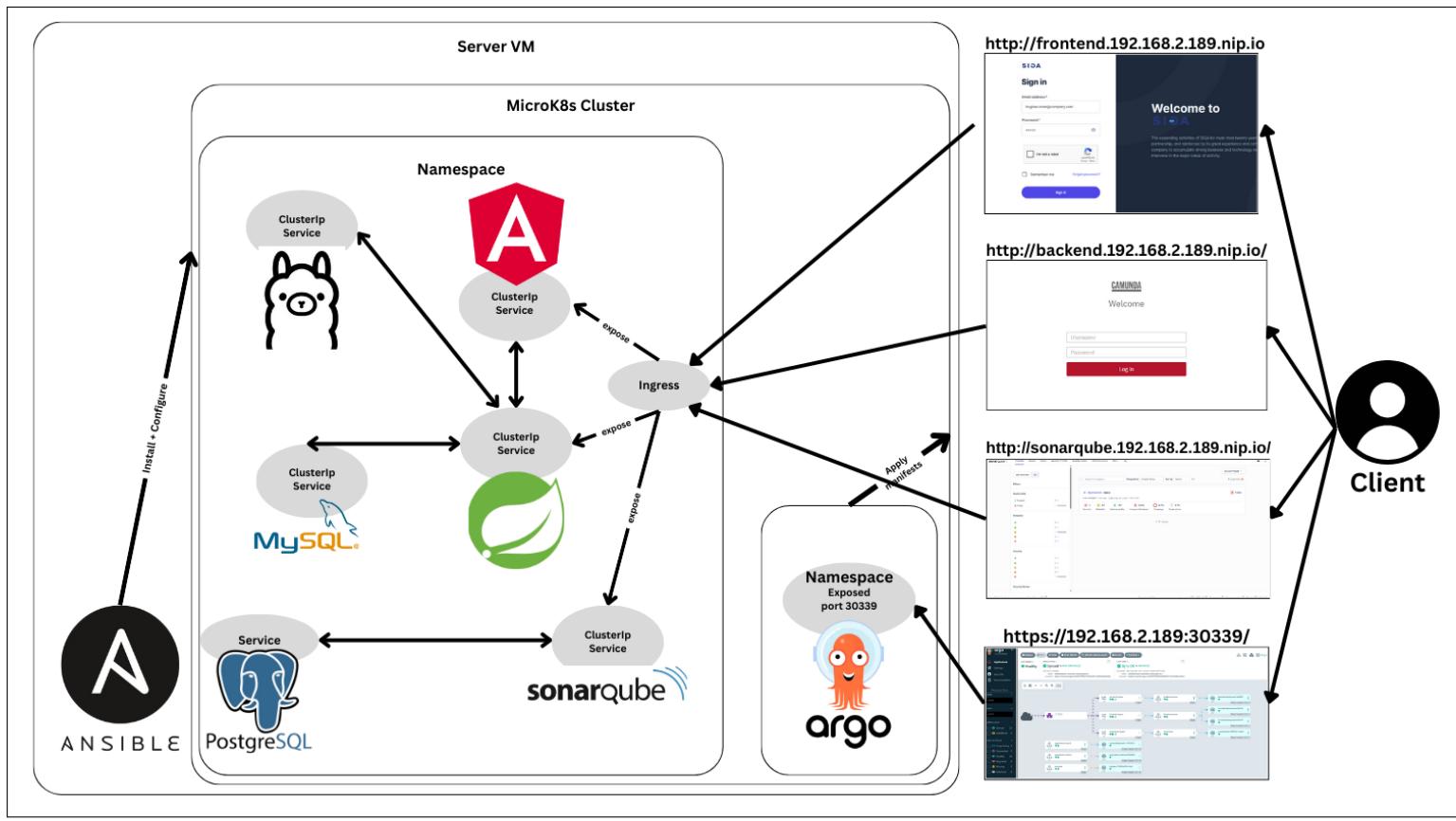


Figure 7.2: Architecture du système

7.5 Réalisation

7.5.1 Installation Automatisée

Arborescence des fichiers d'Ansible

Cette image montre le contenu du répertoire du projet Ansible, incluant les fichiers `ansible.cfg`, `argocd-app-of-apps.yaml`, `argocd-ns.json`, `deploy_argocd_application.yaml`, `install_ansible_dependencies.yaml`, `inventory`, `main.yaml`, `README.md`, `setup_argocd` et `setup_microk8s.yaml`. Elle illustre l'organisation des fichiers nécessaires pour automatiser l'installation et la gestion du cluster MicroK8s et d'ArgoCD.

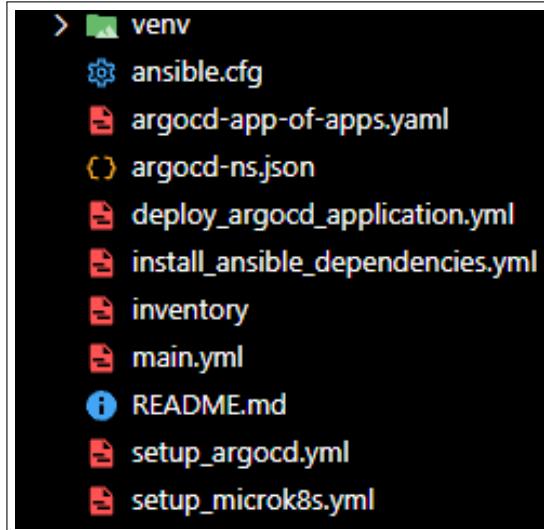


Figure 7.3: Répertoire du Projet Ansible

Fichier ansible.cfg

Cette image présente le fichier de configuration `ansible.cfg`, qui spécifie les paramètres par défaut pour Ansible. On y voit les options `inventory = inventory`, `host_key_checking = False`, `interpreter_python = auto_silent`, et une section `[privilege_escalation]` avec `become = True`, `become_method = sudo`, et `become_ask_pass = False` pour gérer les priviléges d'administration.

```

[defaults]
inventory = inventory
host_key_checking = False
interpreter_python = auto_silent
ansible_python_interpreter = ./venv/bin/python

[privilege_escalation]
become = True
become_method = sudo
become_user = root
become_ask_pass = False

```

Figure 7.4: Fichier `ansible.cfg`

Fichier main.yaml

Cette image illustre le fichier `main.yaml`, le playbook principal qui orchestre l'ensemble du processus. On y voit les imports des playbooks `install_ansible_dependencies.yaml`, `setup_microk8s.yaml`, `setup_argocd.yaml`, et `deploy_argocd_application.yaml`, suivis d'une tâche finale affichant des instructions et un résumé des étapes automatisées (installation, configuration, vérification).

```

main.yml
1  ---
2  - import_playbook: install_ansible_dependencies.yml
3
4  - import_playbook: setup_microk8s.yml
5
6  - import_playbook: setup_argocd.yml
7
8  - import_playbook: deploy_argocd_application.yml
9
10 - name: Final Instructions & Summary
11   hosts: localhost
12   connection: local
13   gather_facts: no
14   tasks:
15     - ansible.builtin.debug:
16       msg:
17         - "-----"
18         - "Automated Setup Process Triggered!"
19         - "-----"
20         - "1. Ansible Dependencies: Playbook attempted to install/verify Ansible and required Python libraries."
21         - "2. MicroK8s: Playbook attempted to install and configure MicroK8s. Addons (dns, ingress, hostpath-storage, registry) should be enabled."
22         - "   IMPORTANT: If this is the first time MicroK8s is configured for your user, you might need to run 'source ~/.bashrc' or log out and back in for 'kubectl' (via microk8s)"
23         - "3. ArgoCD: Playbook attempted to install ArgoCD in the 'argocd' namespace."
24         - "   ArgoCD initial admin password was displayed during the 'Setup ArgoCD' phase. Please note it down."
25         - "   ArgoCD UI access details (likely via NodePort on MicroK8s, e.g., https://YOUR\_VM\_IP: NODE\_PORT) were also displayed."
26         - "4. ArgoCD Application 'approbation-suite': Playbook attempted to create this application in ArgoCD. It should point to your k8s-manifests GitHub repository."
27         - "-----"
28         - "Next Steps & Verification (after playbook finishes):"
29         - "-----"
30         - "   Refresh Shell: If prompted, run 'source ~/.bashrc' or re-login."
31         - "   Verify MicroK8s: Run 'microk8s status --wait-ready' and 'microk8s kubectl get nodes'."
32         - "   Access ArgoCD UI: Use the URL and the admin password obtained from the playbook output. Login as 'admin'."
33         - "   Check 'approbation-suite' Application in ArgoCD: It should appear in the UI. It might take a few moments to sync for the first time. Ensure it becomes Healthy and Synced."
34         - "   Check Deployed Pods: Once ArgoCD syncs, run 'microk8s kubectl get pods -n default' (or the namespace specified in your k8s-manifests/ArgoCD application). You should see one pod in the 'Running' state."
35         - "   Access Frontend Application: Once pods are running and Ingress is set up by ArgoCD (based on your manifests), try accessing your frontend via the MicroK8s node IP: 'http://YOUR\_VM\_IP'"
36         - "-----"
37         - "If any step failed, review the Ansible output for errors."

```

Figure 7.5: Fichier main.yaml

Fichier inventory

Cette image est une capture d'écran du fichier `inventory`, définissant les hôtes cibles pour Ansible. On y voit la section `[all]` avec un hôte `localhost`, une connexion locale, et un interpréteur Python spécifié à `/opt/ansible_venv/bin/python3`, configuré pour exécuter les playbooks localement sur la VM.

```

inventory
1  ---
2  all:
3    hosts:
4      localhost:
5        ansible_connection: local
6        ansible_python_interpreter: /opt/ansible_venv/bin/python3

```

Figure 7.6: Fichier inventory

Fichier install_ansible_dependencies.yaml

Cette image est une capture d'écran du fichier `install_ansible_dependencies.yaml`, un playbook Ansible qui installe les dépendances nécessaires sur la VM Ubuntu. On y voit les tâches pour mettre à jour le cache APT, installer des packages essentiels (comme `software-properties-common`, `python3-pip`, `python3-venv`), ajouter le PPA Ansible, et configurer un environnement virtuel Python.

```
install_ansible_dependencies.yml
```

```
1 ---  
2 - name: Install Ansible and dependencies on Ubuntu VM  
3   hosts: localhost  
4   connection: local  
5   become: yes  
6   gather_facts: yes  
7  
8   pre_tasks:  
9     - name: Update apt cache  
10    apt:  
11      update_cache: yes  
12      changed_when: false  
13  
14   tasks:  
15     - name: Install essential packages  
16       apt:  
17         name:  
18           - software-properties-common  
19           - python3-pip  
20           - python3-venv  
21           - git  
22         state: present  
23  
24     - name: Add Ansible PPA  
25       ansible.builtin.apt_repository:  
26         repo: ppa:ansible/ansible  
27         state: present  
28  
29     - name: Install Ansible  
30       apt:  
31         name: ansible  
32         state: present  
33         update_cache: yes  
34  
35     - name: Create Python virtual environment  
36       ansible.builtin.command:  
37         cmd: python3 -m venv /opt/ansible_venv  
38         creates: /opt/ansible_venv  
39         register: venv_create  
40  
41     - name: Install Python libraries in virtual environment  
42       ansible.builtin.pip:  
43         name:  
44           - kubernetes  
45           - docker  
46           - openshift  
47           - pyyaml  
48           - jsonpatch  
49         virtualenv: /opt/ansible_venv  
50  
51     - name: Verify Ansible installation  
52       command: ansible --version  
53       register: ansible_version  
54       changed_when: false  
55  
56     - name: Display Ansible version  
57       debug:  
58         var: ansible_version.stdout_lines
```

Figure 7.7: Fichier install ansible dependencies.yaml

Fichier setup_microk8s.yaml

Ce playbook Ansible permet d'automatiser l'installation et la configuration de MicroK8s sur une machine Ubuntu. Il installe MicroK8s via Snap, attend que le service soit prêt, puis active plusieurs modules essentiels tels que dns, ingress, hostpath-storage et registry avec une taille personnalisée. Il est conçu pour être idempotent, c'est-à-dire qu'il peut être relancé sans provoquer d'erreurs si les composants sont déjà installés ou activés.

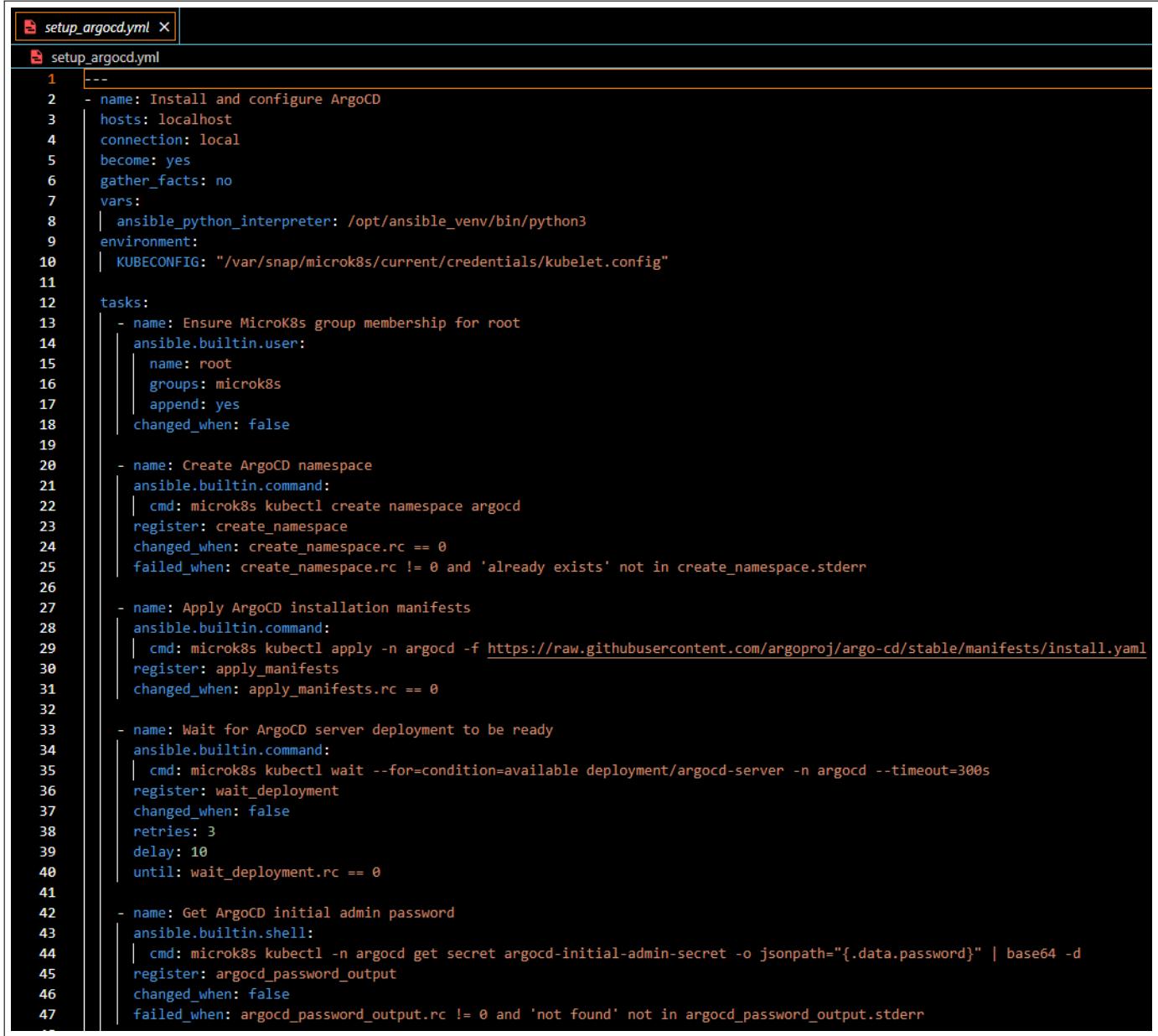


```
setup_microk8s.yaml
setup_microk8s.yaml
2 - name: Setup MicroK8s on Ubuntu VM
3   become: yes
4   gather_facts: yes
5   vars:
6     ansible_python_interpreter: /opt/ansible_venv/bin/python3 # Ensure virtualenv is used
7
8   tasks:
9     - name: Install MicroK8s using snap
10    community.general.snap:
11      name: microk8s
12      classic: yes
13      channel: 1.28/stable
14
15    - name: Wait for MicroK8s to be ready
16      ansible.builtin.command:
17        cmd: microk8s status --wait-ready
18        register: microk8s_status
19        retries: 30
20        delay: 10
21        until: microk8s_status.rc == 0
22        changed_when: false
23
24    - name: Enable MicroK8s DNS addon
25      ansible.builtin.command:
26        cmd: microk8s enable dns
27        register: enable_dns
28        changed_when: enable_dns.rc == 0
29        failed_when: false # Ignore if already enabled
30
31    - name: Enable MicroK8s Ingress addon
32      ansible.builtin.command:
33        cmd: microk8s enable ingress
34        register: enable_ingress
35        changed_when: enable_ingress.rc == 0
36        failed_when: false # Ignore if already enabled
37
38    - name: Enable MicroK8s Hostpath Storage addon
39      ansible.builtin.command:
40        cmd: microk8s enable hostpath-storage
41        register: enable_storage
42        changed_when: enable_storage.rc == 0
43        failed_when: false # Ignore if already enabled
44
45    - name: Enable MicroK8s Registry addon with correct size
46      ansible.builtin.command:
47        cmd: microk8s enable registry --size 20Gi
48        register: enable_registry
49        changed_when: enable_registry.rc == 0
50        failed_when: enable_registry.rc != 0 and 'already enabled' not in enable_registry.stderr
```

Figure 7.8: Fichier setup microk8s.yaml

Fichier setup_argocd.yaml

les images 7.9 et 7.10 présentent le fichier `setup_argocd.yaml`, un playbook Ansible pour installer et configurer ArgoCD. On y voit les tâches créant un namespace ArgoCD, appliquant les manifests d'installation depuis un dépôt GitHub, attendant que le déploiement soit prêt, et récupérant le mot de passe initial de l'administrateur via une commande `kubectl`.



```
setup_argocd.yaml
setup_argocd.yaml
1  ---
2  - name: Install and configure ArgoCD
3    hosts: localhost
4    connection: local
5    become: yes
6    gather_facts: no
7    vars:
8      ansible_python_interpreter: /opt/ansible_venv/bin/python3
9      environment:
10     KUBECONFIG: "/var/snap/microk8s/current/credentials/kubelet.config"
11
12  tasks:
13    - name: Ensure MicroK8s group membership for root
14      ansible.builtin.user:
15        name: root
16        groups: microk8s
17        append: yes
18        changed_when: false
19
20    - name: Create ArgoCD namespace
21      ansible.builtin.command:
22        cmd: microk8s kubectl create namespace argocd
23        register: create_namespace
24        changed_when: create_namespace.rc == 0
25        failed_when: create_namespace.rc != 0 and 'already exists' not in create_namespace.stderr
26
27    - name: Apply ArgoCD installation manifests
28      ansible.builtin.command:
29        cmd: microk8s kubectl apply -n argocd -f https://raw.githubusercontent.com/argoproj/argo-cd/stable/manifests/install.yaml
30        register: apply_manifests
31        changed_when: apply_manifests.rc == 0
32
33    - name: Wait for ArgoCD server deployment to be ready
34      ansible.builtin.command:
35        cmd: microk8s kubectl wait --for=condition=available deployment/argocd-server -n argocd --timeout=300s
36        register: wait_deployment
37        changed_when: false
38        retries: 3
39        delay: 10
40        until: wait_deployment.rc == 0
41
42    - name: Get ArgoCD initial admin password
43      ansible.builtin.shell:
44        cmd: microk8s kubectl -n argocd get secret argocd-initial-admin-secret -o jsonpath=".data.password" | base64 -d
45        register: argocd_password_output
46        changed_when: false
47        failed_when: argocd_password_output.rc != 0 and 'not found' not in argocd_password_output.stderr
```

Figure 7.9: Fichier `setup_argocd.yaml`

```

- name: Display ArgoCD initial admin password
  ansible.builtin.debug:
    msg: "ArgoCD initial admin password: {{ argocd_password_output.stdout }}"

- name: Patch ArgoCD server service to NodePort
  ansible.builtin.command:
    cmd: |
      | microk8s kubectl patch svc argocd-server -n argocd -p '{"spec": {"type": "NodePort"}}'
    register: patch_service
  changed_when: patch_service.rc == 0
  failed_when: patch_service.rc != 0 and 'not found' not in patch_service.stderr
  ignore_errors: true

- name: Get ArgoCD server NodePort
  ansible.builtin.shell:
    cmd: |
      | NODE_IP=192.168.2.189
      | NODE_PORT=$(microk8s kubectl get svc argocd-server -n argocd -o jsonpath='{.spec.ports[?(@.name=="https")].nodePort}')
      | echo "$NODE_IP:$NODE_PORT"
    register: argocd_server_ip
  changed_when: false

- name: Display ArgoCD server access information
  ansible.builtin.debug:
    msg: "ArgoCD server can be accessed at: https://{{ argocd_server_ip.stdout }} (username: admin, password from previous step)"

```

Figure 7.10: Fichier setup_argocd.yaml

Fichier deploy_argocd_application.yaml

Cette image montre le fichier deploy_argocd_application.yaml, un playbook Ansible qui déploie une application ArgoCD. On y voit les tâches vérifiant l'existence du fichier manifeste argocd-app-of-apps.yaml, déployant la ressource via kubectl, et vérifiant l'état de l'application avec un délai de 10 secondes, assurant une synchronisation réussie.

```

1  ---
2  - name: Deploy ArgoCD Application for Approbation Suite
3  hosts: localhost
4  connection: local
5  become: yes
6  gather_facts: no
7
8  vars:
9    | username: "{{ ansible_user_id }}"
10   | argocd_app_manifest_local_path: "argocd-app-of-apps.yaml"
11   | ansible_python_interpreter: "{{ playbook_dir }}/venv/bin/python"
12   | kubeconfig_path: "/var/snap/microk8s/current/credentials/client.config"
13
14 environment:
15   | KUBECONFIG: "{{ kubeconfig_path }}"
16
17 tasks:
18   - name: Check if ArgoCD application manifest file exists
19     stat:
20       | path: "{{ argocd_app_manifest_local_path }}"
21       | register: app_manifest_file_status
22       | delegate_to: localhost
23
24   - name: Fail if ArgoCD application manifest is missing
25     fail:
26       | msg: "ArgoCD application manifest file {{ argocd_app_manifest_local_path }} not found.
27       | Ensure it is in the same directory as this playbook {{ playbook_dir }}."
28       | when: not app_manifest_file_status.stat.exists
29
30   - name: Deploy the ArgoCD Application resource
31     kubernetes.core.k8s:
32       state: present
33       src: "{{ argocd_app_manifest_local_path }}"
34       namespace: argocd
35       register: app_deploy_status
36       until: app_deploy_status.is_succeeded
37       retries: 3
38       delay: 10
39
40   - name: Verify ArgoCD application creation
41     command: "microk8s kubectl get application approbation-suite -n argocd --kubeconfig {{ kubeconfig_path }}"
42     changed_when: false
43     register: argo_app_status
44
45   - name: Display ArgoCD application status
46     debug:
47       | var: argo_app_status.stdout_lines

```

Figure 7.11: Fichier deploy_argocd_application.yaml

7.5.2 Workflows CI

Workflow Frontend

Les images 7.12 et 7.13 sont des captures d'écran d'un fichier de configuration GitHub Actions nommé `.ci.yml`, définissant un pipeline pour le frontend, intitulé Frontend Pipeline. Le workflow est déclenché sur des événements push et pull_request sur la branche main. Il comprend un job `build-deploy` exécuté sur un environnement `ubuntu-latest`, avec plusieurs étapes : checkout du code, configuration de Node.js (version 18), installation des dépen-

dances avec npm install, construction de l’application en mode production avec npm run build --configuration=production, configuration de l’environnement Docker avec docker/setup-buildx-action et docker/login-action utilisant des identifiants secrets (DOCKERHUB_USERNAME et DOCKERHUB_TOKEN), et enfin la construction et la poussée de l’image Docker avec docker buildx build et docker push vers dockerhub approbation-frontend:latest, en intégrant le hash GitHub (github.sha). Cette image illustre le processus automatisé de construction et de déploiement du frontend.

```

1  name: Frontend Pipeline
2  on:
3    | push:
4    |   | branches: [main]
5    | pull_request:
6    |   | branches: [main]
7  jobs:
8    | build-deploy:
9    |   | runs-on: ubuntu-latest
10   |   | steps:
11   |   |     - name: Checkout code
12   |   |     | uses: actions/checkout@v4
13
14   |   |     - name: Set up Node.js
15   |   |     | uses: actions/setup-node@v4
16   |   |     | with:
17   |   |       | node-version: '18'
18
19   |   |     - name: Install dependencies
20   |   |     | run: npm install
21
22   |   |     - name: Build production app
23   |   |     | run: npm run build -- --configuration=production
24
25   |   |     - name: Set up Docker Buildx
26   |   |     | uses: docker/setup-buildx-action@v3
27
28   |   |     - name: Login to Docker Hub
29   |   |     | uses: docker/login-action@v3
30   |   |     | with:
31   |   |       | username: ${{ secrets.DOCKERHUB_USERNAME }}
32   |   |       | password: ${{ secrets.DOCKERHUB_TOKEN }}
33
34   |   |     - name: Build and push Docker image
35   |   |     | run: |
36   |   |       | docker buildx build \
37   |   |         | --platform linux/amd64 \
38   |   |         | -t ${{ secrets.DOCKERHUB_USERNAME }}/approbation-frontend:${{ github.sha }} \
39   |   |         | -t ${{ secrets.DOCKERHUB_USERNAME }}/approbation-frontend:latest \
40   |   |         | --push .

```

Figure 7.12: Fichier Workflow Frontend

```

42      - name: Checkout k8s-manifests repository
43        uses: actions/checkout@v4
44        with:
45          repository: SeifeddineABIDI/k8s-manifests
46          path: manifests
47          token: ${{ secrets.GH_PAT }}
48
49      - name: Update image tag
50        run:
51          - sed -i "s|image:.*|image: ${{ secrets.DOCKERHUB_USERNAME }}/approbation-frontend:${{ github.sha }}|" manifests/frontend/deployment.yaml
52          - cd manifests
53          - git config user.name "seifeddineAbidi"
54          - git config user.email "seifeddine.abidi@esprit.tn"
55          - git add .
56          - if git diff --staged --quiet; then
57            - echo "No changes to commit"
58          else
59            - git commit -m "Update frontend image to ${{ github.sha }}"
60            - git push
61        fi

```

Figure 7.13: Fichier Workflow Frontend

Workflow Backend

Les images 7.14 et 7.15 sont des captures d'écran d'un fichier de configuration GitHub Actions nommé `.ci.yml`, définissant un pipeline pour le backend, intitulé `Backend Pipeline`. Le workflow est déclenché sur des événements `push` et `pull_request` sur la branche `main`, exécuté sur un runner auto-hébergé (`self-hosted`). Les étapes incluent : checkout du code avec une profondeur complète (`fetch-depth: 0`), configuration de JDK 17 avec `actions/setup-java`, mise en cache des dépendances Maven pour accélérer les builds, construction avec Maven en sautant les tests (`mvn clean package -DskipTests`), vérification de la connectivité à SonarQube (via Ingress sur `192.168.2.189:30000`), et exécution des tests avec JaCoCo pour générer un rapport de couverture. Cette image met en évidence le processus CI/CD backend, intégrant des analyses statiques et des tests unitaires sur un environnement local.

```
ci.yml •
.github > workflows > ci.yml
1   name: Backend Pipeline
2   on:
3     push:
4       branches: [main]
5     pull_request:
6       branches: [main]
7
8   jobs:
9     build-deploy:
10    runs-on: self-hosted
11    steps:
12      - name: Checkout code
13        uses: actions/checkout@v4
14        with:
15          fetch-depth: 0
16
17      - name: Set up JDK 17
18        uses: actions/setup-java@v4
19        with:
20          java-version: '17'
21          distribution: 'temurin'
22
23      - name: Cache Maven dependencies
24        uses: actions/cache@v4
25        with:
26          path: ~/.m2/repository
27          key: ${{ runner.os }}-maven-${{ hashFiles('**/pom.xml') }}
28          restore-keys:
29            - ${{ runner.os }}-maven-
30
31      - name: Build with Maven
32        run: mvn clean package -DskipTests
33
34      - name: Test SonarQube connectivity
35        run:
36          - curl -v -f http://sonarqube.192.168.2.189.nip.io || echo "Failed to reach SonarQube via Ingress"
37          - curl -v -f http://192.168.2.189:30000 || echo "Failed to reach SonarQube via NodePort"
38
39      - name: Run tests with JaCoCo
40        run: mvn test jacoco:report
```

Figure 7.14: Fichier Workflow Backend

```

ci.yml
github > workflows > ci.yml
8   jobs:
9     build-deploy:
11       steps:
12         - name: Run SonarQube analysis
13           run: |
14             mvn sonar:sonar -X \
15               -Dsonar.projectKey=${{ secrets.SONAR_PROJECT_KEY }} \
16               -Dsonar.host.url=http://sonargube.192.168.2.189.nip.io \
17               -Dsonar.login=${{ secrets.SONAR_TOKEN }} \
18               -Dsonar.coverage.jacoco.xmlReportPaths=target/site/jacoco/jacoco.xml && break
19             echo "Attempt $i failed, retrying in 10 seconds..."
20             sleep 10
21         done
22
23         - name: Set up Docker Buildx
24           uses: docker/setup-buildx-action@v3
25
26         - name: Login to Docker Hub
27           uses: docker/login-action@v3
28           with:
29             username: ${{ secrets.DOCKERHUB_USERNAME }}
30             password: ${{ secrets.DOCKERHUB_TOKEN }}
31
32         - name: Build and push Docker image
33           run: |
34             docker buildx build \
35               --platform linux/amd64 \
36               -t ${secrets.DOCKERHUB_USERNAME}:/approbation-backend:${{ github.sha }} \
37               -t ${secrets.DOCKERHUB_USERNAME}:/approbation-backend:latest \
38               --push .
39
40         - name: Checkout k8s-manifests repository
41           uses: actions/checkout@v4
42           with:
43             repository: SeifeddineABIDI/k8s-manifests
44             path: manifests
45             token: ${{ secrets.GH_PAT }}
46
47         - name: Update image tag
48           run: |
49             sed -i "s|image:.*|image: ${secrets.DOCKERHUB_USERNAME}:/approbation-backend:${{ github.sha }}|" manifests/backend/deployment.yaml
50             cd manifests
51             git config user.name "seifeddineAbidi"
52             git config user.email "seifeddine.abidi@esprit.tn"
53             git add .
54             if git diff --staged --quiet; then
55               echo "No changes to commit"
56             else
57               git commit -m "Update backend image to ${{ github.sha }}"
58               git push
59             fi

```

Figure 7.15: Fichier Workflow Backend

7.5.3 Runner SonarQube

L'image 7.16 est une capture d'écran d'un terminal affichant les logs d'un runner GitHub Actions configuré pour exécuter des tâches CI/CD, spécifiquement pour un service backend nommé SeifeddineABIDI-Approbation-backend-siga. Elle montre le démarrage et le fonctionnement du runner sur un environnement Ubuntu VM, avec les détails suivants : le runner est chargé avec le statut enabled et preset: enabled, utilisant un fichier de configuration système situé dans:

/etc/systemd/system/actions.runner.SeifeddineABIDI-Approbation-backend-siga.service. Les logs indiquent un démarrage réussi à 09:22:35 le 21 mai 2025, avec un PID de 1571 (processus runsvc.sh) et une mémoire utilisée de 171,8 Mo (peak : 173,2 Mo). Le CPU consomme 8,437 secondes, et le runner écoute sur le port 1801. On observe également des tentatives de connexion à GitHub, avec un délai HTTP après 00:00:10 le 21 mai à 09:22:40, suivies d'une reconnexion réussie à 09:51:55, démontrant la résilience du runner face aux interruptions réseau. Cette image illustre la stabilité et la gestion des erreurs du runner auto-hébergé dans un environnement DevOps.

```

siga@siga:/actions-runner$ sudo systemctl status actions.runner.*
● actions.runner.SeifeddineABIDI-Approbation-backend.siga.service - GitHub Actions Runner (SeifeddineABIDI-Approbation-backend.siga)
  Loaded: loaded (/etc/systemd/system/actions.runner.SeifeddineABIDI-Approbation-backend.siga.service; enabled; preset: enabled)
  Active: active (running) since Wed 2025-05-21 09:22:35 CET; 5h 26min ago
    Main PID: 1571 (runsvc.sh)
      Tasks: 21 (limit: 12461)
     Memory: 171.8M (peak: 173.2M)
        CPU: 8.437s
      CGroup: /system.slice/actions.runner.SeifeddineABIDI-Approbation-backend.siga.service
              └─1571 /bin/bash /home/siga/actions-runner/runsvc.sh
                  ├─1630 ./externals/node20/bin/node ./bin/RunnerService.js
                  ├─1801 /home/siga/actions-runner/bin/Runner.Listener run --startuptype service
                  └─1801 /home/siga/actions-runner/bin/Runner.Listener run --startuptype service

May 21 09:22:35 siga systemd[1]: Started actions.runner.SeifeddineABIDI-Approbation-backend.siga.service - GitHub Actions Runner (SeifeddineABIDI-Approbation-backend.siga).
May 21 09:22:36 siga runsvc.sh[1571]: .path=/usr/local/sbin:/usr/local/bin:/usr/sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/local/games:/snap/bin:/snap/bin
May 21 09:22:36 siga runsvc.sh[1630]: Starting Runner listener with startup type: service
May 21 09:22:36 siga runsvc.sh[1630]: Started listener process, pid: 1801
May 21 09:22:36 siga runsvc.sh[1630]: Started running service
May 21 09:22:39 siga runsvc.sh[1630]: ✓ Connected to GitHub
May 21 09:22:40 siga runsvc.sh[1630]: Current runner version: '2.324.0'
May 21 09:22:40 siga runsvc.sh[1630]: 2025-05-21 08:22:40Z: Listening for Jobs
May 21 09:50:38 siga runsvc.sh[1630]: 2025-05-21 08:50:38Z: Runner connect error: The HTTP request timed out after 00:01:00.. Retrying until reconnected.
May 21 09:51:55 siga runsvc.sh[1630]: 2025-05-21 08:51:55Z: Runner reconnected.
siga@siga:/actions-runner$ ./run.sh
✓ Connected to GitHub

```

Figure 7.16: Capture du Runner GitHub Actions

7.5.4 Gestion des Secrets

L'image 7.17 est une capture d'écran du fichier secrets.yaml, qui définit plusieurs secrets Kubernetes pour sécuriser les identifiants et mots de passe utilisés dans l'environnement DevOps. Le fichier est structuré en trois ressources de type Secret avec l'API version v1. Le premier secret, nommé mysql-credentials, dans le namespace default, contient des données opaques pour l'utilisateur (username) et le mot de passe (password) encodés en base64 (valeurs masquées comme cm9vdA== et c29tZXBhc3N3b3Jk). Le deuxième secret, nommé backend-secrets, dans le namespace default, inclut un secret JWT (jwt-secret) et un mot de passe par e-mail (mail-password) encodés en base64, ainsi qu'un secret de recaptcha (recaptcha-secret) pour la validation. Le troisième secret, nommé postgres-secret, dans le namespace sonarqube, contient un mot de passe PostgreSQL (postgres-password) encodé en base64 (YWRtaW4=). Cette image illustre la gestion sécurisée des credentials dans Kubernetes, essentielle pour l'authentification et la protection des données dans les pods (MySQL, backend, et SonarQube).

```

secrets.yaml
1  apiVersion: v1
2  kind: Secret
3  metadata:
4    name: mysql-credentials
5    namespace: default
6  type: Opaque
7  data:
8    username: cm9vdA==
9    password: cm9vdA==
10 ---
11 apiVersion: v1
12 kind: Secret
13 metadata:
14   name: backend-secrets
15   namespace: default
16 type: Opaque
17 data:
18   jwt-secret: N2YwNTRmNmQwNjdlMzgzYTJmNTRiZDFhMjQwODg3OWFjOWU3YjIyMTk5NWI5MzIzOTE4MzQ3ZWU2ZjB1MzQxYQ==
19   mail-username: c2F5Zi5hYmlkaTFAZ21haWwuY29t
20   mail-password: eWRwZSBnbmhoIG52bmwgYmJ6aQ==
21   recaptcha-secret: NkxkRHVoY3JBQUFBQUtGSHQ1WEx0QXVDdWJjcFd6NFBJOFhfcDJucg==
22 ---
23 apiVersion: v1
24 kind: Secret
25 metadata:
26   name: postgres-secret
27   namespace: sonarqube
28 type: Opaque
29 data:
30   postgres-password: YWRtaW4=
31 ---

```

Figure 7.17: Capture du fichier secrets.yaml

7.5.5 Exposition des Services

Service Frontend

L'image 7.18 est une capture d'écran du fichier `ingress-frontend.yaml`, qui définit les règles Ingress pour exposer le service frontend dans un cluster MicroK8s. Le fichier utilise l'API `networking.k8s.io/v1` et spécifie un Ingress nommé `frontend-ingress` dans le namespace `default`. Les règles dirigent le trafic HTTP vers le domaine `frontend.192.168.2.189.nip.io`, acheminant les requêtes vers le service `frontend-service` sur le port 80. Une annotation `nginx.ingress.kubernetes.io/rewrite-target: /` est incluse pour réécrire les chemins d'URL, et l'Ingress est configuré pour utiliser un contrôleur Ingress NGINX, assurant un accès externe sécurisé à l'application frontend.

```

frontend > 📄 ingress.yaml
 1  apiVersion: networking.k8s.io/v1
 2  kind: Ingress
 3  metadata:
 4    name: frontend-ingress
 5    namespace: default
 6    annotations:
 7      | nginx.ingress.kubernetes.io/rewrite-target: /
 8  spec:
 9    ingressClassName: public
10    rules:
11      - host: frontend.192.168.2.189.nip.io
12        http:
13          paths:
14            - path: /
15              pathType: Prefix
16              backend:
17                service:
18                  name: frontend-service
19                  port:
20                    number: 80

```

Figure 7.18: Fichier ingress-frontend.yaml

Service Backend

L'image 7.19 montre le fichier `ingress-backend.yaml`, qui configure l'exposition du service backend dans le cluster MicroK8s. Le fichier définit un Ingress nommé `backend-ingress` dans le namespace `default`, utilisant l'API `networking.k8s.io/v1`. Les règles associent le domaine `backend.192.168.2.189.nip.io` au service `backend-service` sur le port `8080`, avec une annotation `nginx.ingress.kubernetes.io/backend-protocol: HTTP` pour spécifier le protocole. Une configuration supplémentaire pour les limites de taille des requêtes (`nginx.ingress.kubernetes.io/proxy-body-size: "50m"`) est incluse, permettant la gestion de payloads plus volumineux, comme les appels API du backend.

```

backend > ingress.yaml
 1 apiVersion: networking.k8s.io/v1
 2 kind: Ingress
 3 metadata:
 4   name: backend-ingress
 5   namespace: default
 6   annotations:
 7     nginx.ingress.kubernetes.io/rewrite-target: /
 8     nginx.ingress.kubernetes.io/enable-cors: "true"
 9     nginx.ingress.kubernetes.io/cors-allow-origin: "http://frontend.192.168.2.189.nip.io"
10    nginx.ingress.kubernetes.io/cors-allow-methods: "GET, POST, PUT, DELETE, OPTIONS"
11    nginx.ingress.kubernetes.io/cors-allow-headers: "DNT,User-Agent,X-Requested-With,If-Modified-Since,Cache-Control,Content-Type,Range,Authorization"
12    nginx.ingress.kubernetes.io/cors-allow-credentials: "true"
13    nginx.ingress.kubernetes.io/cors-max-age: "3600"
14 spec:
15   ingressClassName: public
16   rules:
17     - host: backend.192.168.2.189.nip.io
18       http:
19         paths:
20           - path: /
21             pathType: Prefix
22             backend:
23               service:
24                 name: backend-service
25                 port:
26                   number: 8080

```

Figure 7.19: Fichier ingress-backend.yaml

Service SonarQube

L'image 7.20 présente le fichier `ingress-sonarqube.yaml`, qui expose le service SonarQube pour l'analyse de code dans le cluster MicroK8s. L'Ingress, nommé `sonarqube-ingress`, est défini dans le namespace `sonarqube` avec l'API `networking.k8s.io/v1`. Les règles acheminent le trafic du domaine `sonarqube.192.168.2.189.nip.io` vers le service `sonarqube-service` sur le port `9000`. Une annotation `nginx.ingress.kubernetes.io/proxy-read-timeout: "300"` est ajoutée pour augmenter le délai d'attente, adapté aux analyses longues de SonarQube, et une annotation `nginx.ingress.kubernetes.io/ssl-redirect: "false"` désactive la redirection HTTPS pour simplifier l'accès local.

```

apiVersion: networking.k8s.io/v1
kind: Ingress
metadata:
  name: sonarqube-ingress
  namespace: sonarqube
  annotations:
    nginx.ingress.kubernetes.io/rewrite-target: /
spec:
  rules:
    - host: sonarqube.192.168.2.189.nip.io
      http:
        paths:
          - path: /
            pathType: Prefix
            backend:
              service:
                name: sonarqube
                port:
                  number: 9000

```

Figure 7.20: Fichier ingress-sonarqube.yaml

7.5.6 Surveillance et Rollback

Vue générale de l'application

Cette image montre une capture d'écran de l'interface utilisateur d'ArgoCD, affichant les détails de l'application nommée approbation-suite. L'état de l'application est indiqué comme Healthy et Synced to HEAD (083e95d), avec une dernière synchronisation réussie le 19 mai 2025 à 06:00 GMT+0100, il y

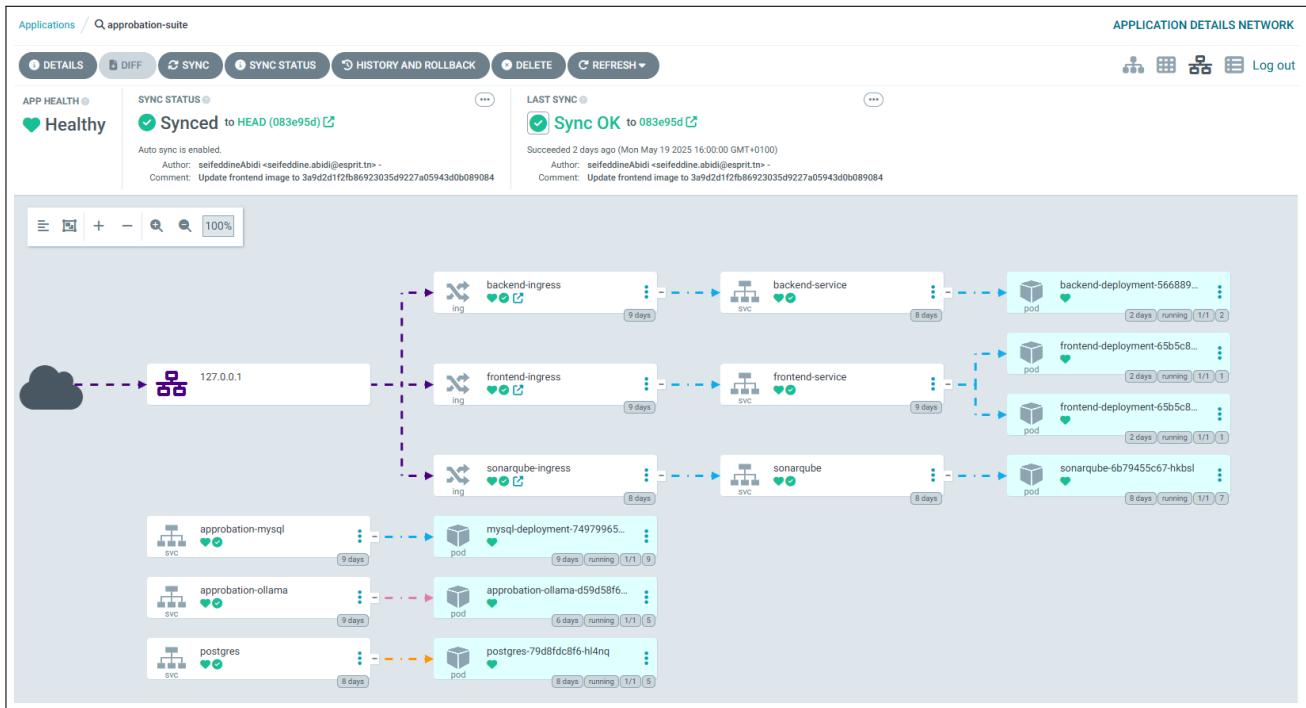


Figure 7.21: Interface ArgoCD - Vue générale de l'application "approbation-suite"

Vue des services et secrets

Cette image illustre une section de l'interface ArgoCD listant les services (SVC) et secrets synchronisés pour l'application approbation-suite. Elle inclut les services approbation-mysql, approbation-ollama, backend-service, frontend-service, postgres, et sonarqube, tous marqués comme synchronisés avec succès depuis 6 à 9 jours. Les secrets associés, tels que backend-secrets, mysql-credentials, et postgres-secret, sont également listés, indiquant une gestion sécurisée des identifiants. Les volumes persistants (PVC) comme mysql-pvc, ollama-models, postgres-pvc, et sonarqube-pvc sont synchronisés depuis 6 à 9 jours, assurant la persistance des données pour les bases de données et les modèles, ce qui souligne la robustesse de l'infrastructure Kubernetes gérée par ArgoCD.

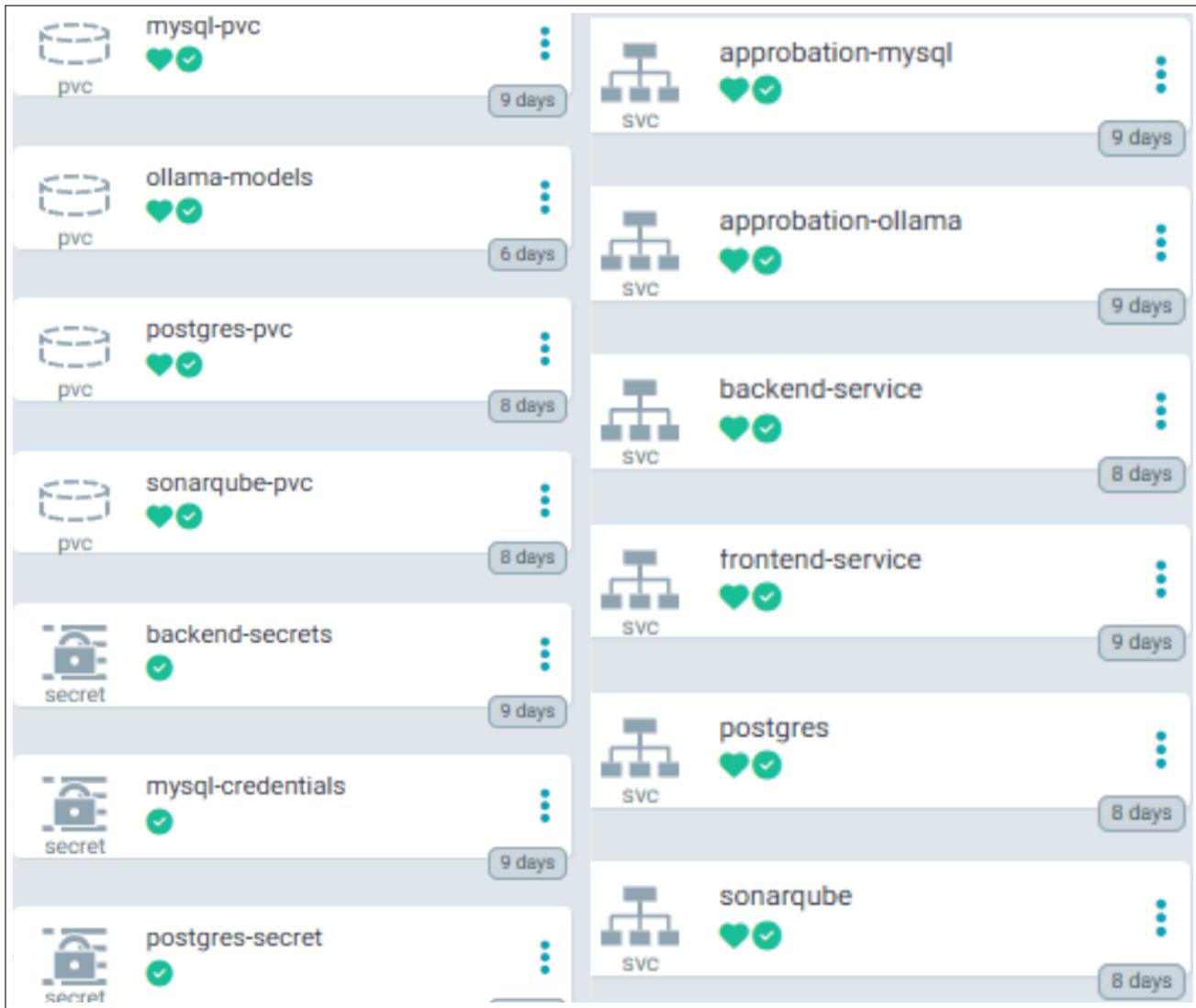


Figure 7.22: Interface ArgoCD - Vue des services et secrets

Vue détaillée des déploiements et pods

Cette image montre une capture d'écran de l'interface utilisateur d'ArgoCD, affichant les détails de l'application nommée approbation-suite. L'état de l'application est indiqué comme Healthy et Synced to HEAD (083e95d), avec une dernière synchronisation réussie le 19 mai 2025 à 06:00 GMT+0100, il y a 2 jours.

L'auteur de la synchronisation automatique est seifeddineAbidi-seifeddine.abidi@esprit.tn, avec un commentaire indiquant une mise à jour de l'image frontend vers un hash GitHub spécifique (3a9d21f2b6923035d9227a5943d0b9084).

La vue réseau affiche les connexions entre l'adresse IP 127.0.1 et divers Ingress (backend-ingress, frontend-ingress, sonarqube-ingress), reliant les services (backend-service, frontend-service, sonarqube) à leurs déploiements correspondants (backend-deployment, frontend-deployment, sonarqube) et pods.

Chaque ressource est marquée comme synchronisée avec succès, avec des durées de fonctionnement allant de 2 à 9 jours, reflétant une infrastructure stable et bien orchestrée.

Figure 7.23: Interface ArgoCD - Vue détaillée des déploiements et pods

Tableau de bord de déploiement

Le tableau de bord de déploiement Kubernetes présenté dans l'image offre une vue d'ensemble claire de l'état des composants de l'application.

Il inclut plusieurs déploiements tels que approbation-ollama, backend-deployment, frontend-deployment, mysql-deployment, postgres et sonarqube, chacun avec des pods en exécution et des durées d'activité variées.

Les ressources d'ingress, comme backend-ingress, frontend-ingress et sonarqube-ingress, sont également affichées, garantissant une gestion efficace des accès.

Tous les éléments sont marqués comme sains, comme en témoignent les coches vertes, reflétant une infrastructure stable et opérationnelle.

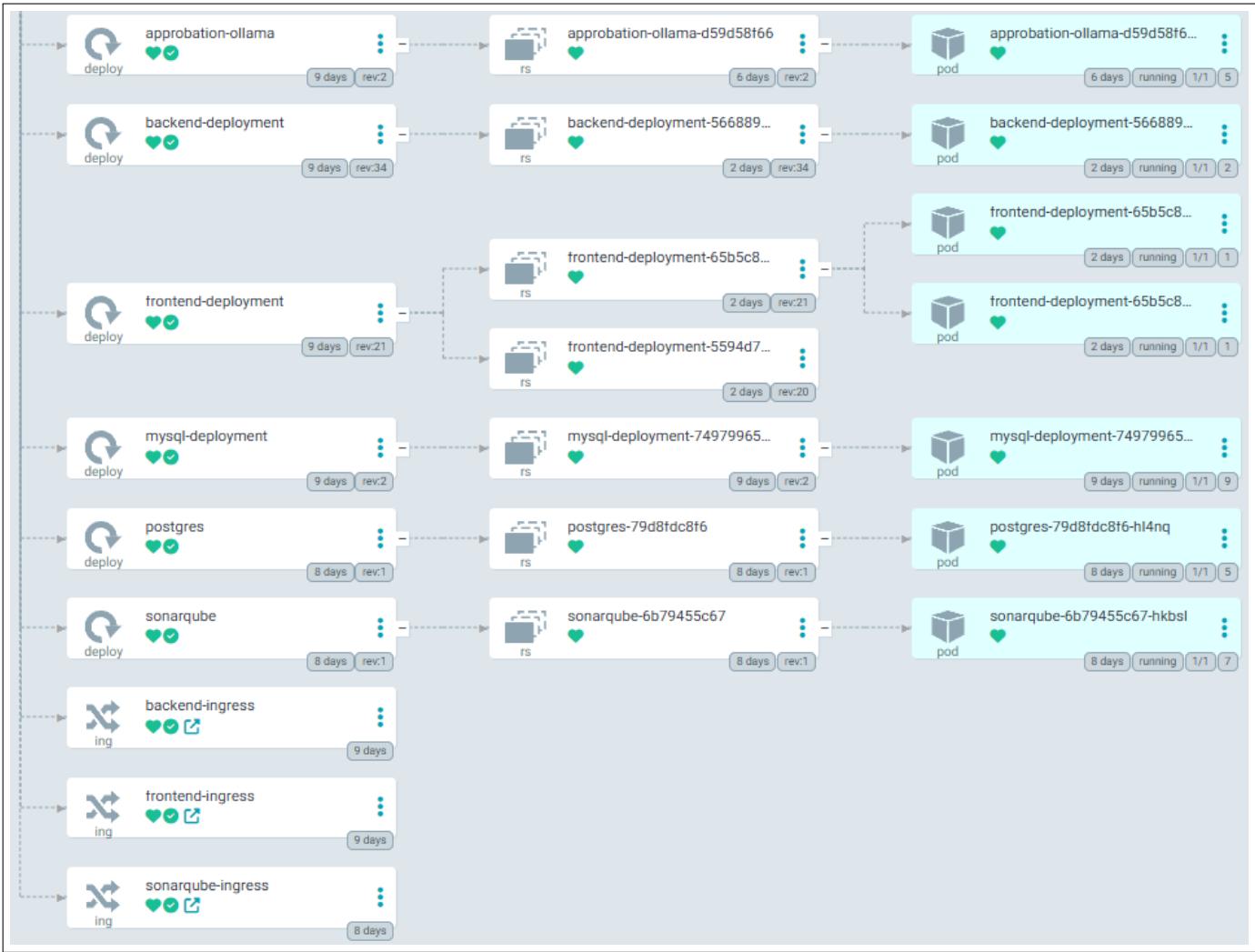


Figure 7.24: Interface ArgoCD - Tableau de bord de déploiement

Architecture du dépôt des déploiements

L'architecture du dépôt GitHub illustré dans l'image, nommé **K8S-MANIFESTS**, contient les fichiers de configuration Kubernetes pour une application multi-composants. L'architecture est organisée en plusieurs dossiers, chacun correspondant à un service spécifique : backend, frontend, mysql, ollama, postgres, sonarqube, ainsi que des fichiers globaux kustomization.yaml et secrets.yaml. Le dossier backend inclut les fichiers deployment.yaml, ingress.yaml, kustomization.yaml, et service.yaml, définissant le déploiement, l'accès réseau, et les services du backend. De manière similaire, le dossier frontend suit la même structure pour l'interface utilisateur. La base de données mysql est configurée via deployment.yaml, kustomization.yaml, pvc.yaml (volume persistant), et service.yaml, utilisée pour stocker les données des applications frontend et backend. Pour sonarqube, qui utilise postgres comme base de données, les fichiers incluent deployment.yaml, ingress.yaml, kustomization.yaml, pvc.yaml, et service.yaml, assurant l'analyse de code avec un stockage persistant via PostgreSQL. Enfin, le dossier ollama gère un composant supplémentaire avec une structure similaire, tandis que les fichiers globaux kustomization.yaml et secrets.yaml centralisent la configuration et les secrets de l'application.

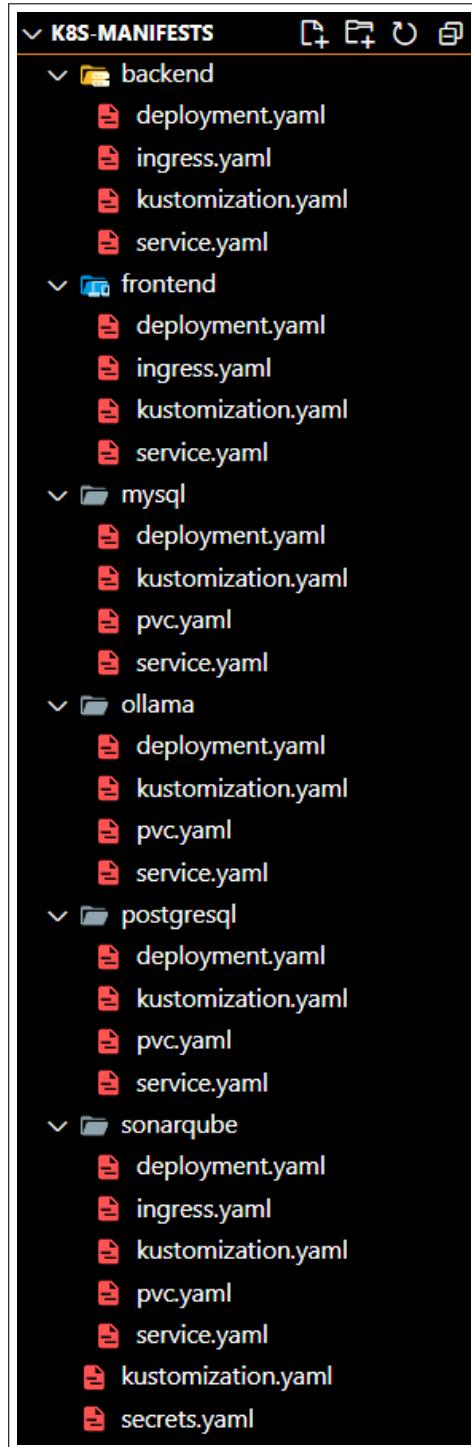


Figure 7.25: Architecture du dépôt des déploiements

7.6 Conclusion

Ce chapitre a détaillé le déroulement du Sprint 5, axé sur la mise en place d'un pipeline DevOps et d'une gestion GitOps pour notre plateforme. Grâce à l'intégration de GitHub Actions, nous avons automatisé la construction, les tests et le déploiement des images Docker, en assurant une qualité de code élevée via SonarQube. L'adoption d'ArgoCD et du dépôt `k8s-manifests` a permis une synchronisation continue et une gestion centralisée des déploiements sur MicroK8s, renforçant la fiabilité grâce à des capacités de surveillance et de rollback. La sécurisation des données sensibles via des secrets Kubernetes et l'exposition des services avec Ingress ont également été des étapes clés pour garantir un accès sécurisé et opérationnel. Ces avancées ont posé des bases solides pour une infrastructure évolutive, maintenable et robuste, tout en optimisant l'efficacité opérationnelle de l'équipe. Ce sprint marque une transition réussie vers des pratiques DevOps et GitOps modernes, prêtes à supporter les évolutions futures du projet.

Conclusion générale

Ce projet de fin d'études, réalisé au sein de SIGA, a permis de concevoir et de déployer un « Portail de Gestion des Approbations avec Workflows Dynamiques », répondant à un besoin critique des organisations modernes : simplifier et centraliser la gestion des processus d'approbation. À travers une démarche structurée en sept sprints, nous avons transformé une problématique complexe en une solution intuitive, performante et évolutive, tout en adoptant des technologies et des pratiques alignées sur les standards actuels.

Les premiers chapitres ont posé les fondations du projet, en analysant le contexte et les besoins, puis en implémentant les fonctionnalités de base comme l'authentification et la gestion des utilisateurs avec Angular et Spring Boot. Les sprints suivants ont enrichi le portail avec des fonctionnalités avancées, telles que la soumission et le suivi des demandes via des workflows dynamiques orchestrés par Camunda, ainsi que des outils d'analyse pour superviser les activités et améliorer l'expérience utilisateur, notamment grâce à l'intégration d'un chatbot. Le dernier sprint, axé sur les pratiques DevOps et GitOps, a permis d'automatiser le déploiement avec GitHub Actions, ArgoCD, Docker et Kubernetes, garantissant une scalabilité et une fiabilité optimales tout en sécurisant les données sensibles via des secrets Kubernetes.

Ce projet a été une opportunité d'apprentissage exceptionnelle, mêlant défis techniques et collaboration étroite avec l'équipe de SIGA. Il a nécessité une maîtrise approfondie des technologies modernes et une adaptation constante aux imprévus, comme la configuration des runners auto-hébergés ou la gestion des synchronisations avec ArgoCD. Ces obstacles, bien que complexes, ont renforcé notre capacité à résoudre des problèmes de manière méthodique et à produire une solution robuste, prête à répondre aux besoins réels des utilisateurs. Au-delà des aspects techniques, cette expérience a également permis de développer des compétences en gestion de projet et en travail d'équipe, essentielles dans un environnement professionnel dynamique.

En perspectives, ce portail pourrait évoluer vers une intégration plus poussée avec des outils d'intelligence artificielle pour anticiper les besoins des utilisateurs, ou encore vers une adoption de solutions serverless pour optimiser les coûts et la performance. Une extension à des environnements multi-clusters Kubernetes pourrait également être envisagée pour répondre aux besoins de grandes entreprises.

En conclusion, ce projet a atteint ses objectifs initiaux en offrant une solution complète et moderne pour la gestion des approbations, tout en ouvrant la voie à de futures améliorations. Il illustre la puissance des approches DevOps et GitOps dans le développement de systèmes complexes, et

témoigne de l'impact transformateur des technologies de l'information lorsqu'elles sont mises au service des besoins organisationnels. Cette expérience au sein de SIGA restera une étape marquante, tant pour les compétences techniques acquises que pour la satisfaction d'avoir contribué à une solution qui simplifie le quotidien des utilisateurs.s

Nétographie

Sites Web

- ref1 Sofrecom Tunisie. Consulté le 15/05/2023.
<https://www.sofrecom.com/fr>
- ref2 La méthodologie DevOps. Consulté le 15/05/2023.
<https://openclassrooms.com/fr/courses/6093671-decouvrez-la-methodologie-devops/6183233-decouvrez-la-methode-devops>
- ref3 Guide to continuous integration. Consulté le 18/05/2023.
<https://about.gitlab.com/2018/01/22/a-beginners-guide-to-continuous-integration/>
- ref4 Gitlab CI/CD. Consulté le 18/05/2023.
<https://www.supinfo.com/articles/single/6822-integration-continue-gitlab-gitlab-ci-cd>
- ref5 La méthode GOROCO. Consulté le 16/05/2023.
<http://tumador.blogspot.com/2009/08/goroco-method.html>
- ref6 DevOps pour les nuls. Consulté le 17/05/2023.
<https://pdfslide.net/technology/devops-pour-les-nuls.html>
- ref7 Internet et réseaux sociaux. Consulté le 15/05/2023.
<https://www.blogdumoderateur.com/chiffres-cles-internet-reseaux-sociaux-monde-avril-2023/>
- ref8 Processus DevOps. Consulté le 20/05/2023.
<https://keltio.fr/devops/les-differentes-etapes-du-devops/>
- ref9 DEVOPS ADOPTION AND IMPLEMENTATION IN SOFTWARE DEVELOPMENT PRACTICE. Consulté le 25/05/2023.
- ref10 Méthodes agiles. Consulté le 30/05/2023.
<https://www.qualitystreet.fr/2010/01/13/manifeste-agile/>
- ref11 Le Guide Scrum. Consulté le 27/05/2023.
- ref12 Git. Consulté le 27/05/2023.
<https://robertgreiner.com/getting-started-with-git-and-tortoisegit-on-windows/>

ref13 GitLab et Gitlab CI. Consulté le 03/06/2023.
<https://blog.eleven-labs.com/fr/introduction-gitlab-ci/>

ref14 Docker. Consulté le 03/06/2023.
<https://docs.docker.com/get-started/>

ref15 Ansible. Consulté le 03/06/2023.
<https://medium.com/formcept/configuration-management-andcontinuousdeployment>

ref16 DevOps Store. Consulté le 03/06/2023.
<https://devops-store.rd.francetelecom.fr/dos>

ref17 VS Code. Consulté le 03/06/2023.
<https://github.com/Microsoft/vscode>

ref18 Red et Eclipse. Consulté le 03/06/2023.
<https://marketplace.eclipse.org/content/red-robot-editor>

ref19 Robot Framework. Consulté le 06/06/2023.
<https://deusyss.developpez.com/tutoriels/Python/Robotframework/>

ref20 Tests d'acceptation. Consulté le 06/06/2023.
https://gayerie.dev/docs/testing/test_acceptation/test_acceptation.html

ref21 Qualité de code avec Sonar. Consulté le 07/06/2023.
https://linsolas.developpez.com/articles/java/qualite/sonar/?page=page_1

ref22 Test et analyse de code. Consulté le 07/06/2023.
<https://plazza.orange.com/groups/techlead-devops/blog/2019/08/16/sonarqube-comment-tester-et-analyser>

ref23 Tests automatisés. Consulté le 09/06/2023.
http://mattischneider.fr/software/watai/WATAI-web_testing_whitepaper.pdf

ref24 Formation Robot Framework. Consulté le 09/06/2023.
<https://plazza.orange.com/docs/DOC-673969>

ref25 Robot Framework. Consulté le 11/06/2023.
<https://www.linkedin.com/pulse/robot-framework-test-automation-smart-way-agha-anas/>

ref26 Keywords Robot Framework. Consulté le 11/06/2023.
https://devops-store.rd.francetelecom.fr/dos?page_id=1150

ref27 Architecture test. Consulté le 11/06/2023.
<https://medium.com/arcadia-software-development/automate-test-api-with-robot-framework-3035af1c9e2>

ref28 Composants de test. Consulté le 12/06/2023.
<https://www.linkedin.com/pulse/robot-framework-test-automation-smart-way-agha-anas/>

ref29 Architecture Sonarqube. Consulté le 12/06/2023.
<https://plazza.orange.com/docs/DOC-1255071>

- ref30 ODE et New Delivery. Consulté le 13/06/2023.
<https://plazza.orange.com/docs/DOC-2089482>
- ref31 Sonar Quality Gate et Profile. Consulté le 14/06/2023.
<https://docs.sonarqube.org/latest/instance-administration/quality-profiles/>
- ref32 Intégration de template ODE. Consulté le 14/06/2023.
<https://orange-deployment-engine.pages.gitlab.tech.orange/ode-documentation/1-principes-fonctionnement/>
- ref33 Ode Deployment. Consulté le 14/06/2023.
<https://plazza.orange.com/people/rahendatriprakosa.rosmedi@orange.com/blog/2021/04/20/>
- ref34 Structure de rôle Ansible. Consulté le 15/06/2023.
<https://devops-store.rd.francetelecom.fr/dos?p=3280>
- ref35 JaCoCo-Maven Plugin. Consulté le 15/06/2023.
<https://www.lambdatest.com/blog/reporting-code-coverage-using-maven-and-jacoco-plugin/>
- ref36 Cycle de vie Maven. Consulté le 15/06/2023.
<https://www.baeldung.com/maven-goals-phases#:~:text=A>
- ref37 Maven enforcer plugin. Consulté le 15/06/2023.
<https://fullstackcode.dev/2023/01/09/complete-guide-to-maven-enforcer-plugin/>
- ref38 Maven deploy plugin. Consulté le 15/06/2023.
<https://maven.apache.org/plugins/maven-deploy-plugin/>
- ref39 Maven CI friendly plugin. Consulté le 16/06/2023.
<https://developpaper.com/ci-friendly-versions-revision-under-maven-multi-module-architecture/>
- ref40 Maven SCM plugin. Consulté le 18/06/2023.
<https://subscription.packtpub.com/book/application-development/9781785286124/8/ch08lv11sec84/using-the-maven-scm-plugin>
- ref41 Architecture Protractor. Consulté le 20/06/2023.
<https://pfs-ltaas.pages.gitlab.si.francetelecom.fr/ltaas-wiki/protractor/>
- ref42 Tests unitaires Angular. Consulté le 20/06/2023.
<https://medium.com/swlh/angular-unit-testing-jasmine-karma-step-by-step-e3376d110ab4>
- ref43 GitLab docs. Consulté le 28/06/2023.
<https://docs.gitlab.com/ee/ci/>

Bibliography

- [1] GitLab. A Beginner’s Guide to Continuous Integration, 2018. URL <https://about.gitlab.com/2018/01/22/a-beginners-guide-to-continuous-integration/>. Consulté le 18/05/2023.
- [2] Quality Street. Manifeste Agile, 2010. URL <https://www.qualitystreet.fr/2010/01/13/manifeste-agile/>. Consulté le 30/05/2023.
- [3] Supinfo. Intégration continue avec Gitlab et Gitlab CI/CD, 2023. URL <https://www.supinfo.com/articles/single/6822-integration-continue-gitlab-gitlab-ci-cd>. Consulté le 18/05/2023.
- [4] Tumador. La méthode GOROCO, 2009. URL <http://tumador.blogspot.com/2009/08/goroco-method.html>. Consulté le 16/05/2023.

Bibliographie

- [1] Abdelaziz ABDELLATIF. *Ingénierie des méthodes et des processus - Méthodes agiles*, 2000.