République Algérienne Démocratique et Populaire
Ministère de l'Enseignement Supérieur et de la Recherche Scientifique

Université A. Mira de Béjaia
Faculté des Sciences Exactes
Département d'Informatique

Mémoire de MASTER EN INFORMATIQUE

Génie logiciel & Administration sécurité réseau

Thème

Développement agile d'une application mobile à base de cloud. Cas d'étude : Gestion des patients

Présenté par :
Ikhlef KENOUCHE & Maathouk BENZIANE

Évalué par le jury composé de :

Encadrante : Dr AIT-ABDELOUHAB Karima U. A/Mira Béjaia. **M.C.B**
Co-Encadrante : Dr AZOUI Aicha U. A/Mira Béjaia. **M.A.A**
Présidente : Dr SOUADIH Rebiha U. A/Mira Béjaia. **M.C.B**
Examinatrice : Mme LAHLAH Souaad U. A/Mira Béjaia. **M.C.B**

Béjaia, Session normale septembre 2021.

⁕ *Remerciements* ⁕

Nos sentiments de reconnaissance vont en premier lieu à l'endroit de nos encadrantes, Mme. D. K.AIT ABDELOUHAB et Mme A.AZOUI pour leur grande disponibilité, leur esprit de rigueur et de méthode, leurs conseils et leurs remarques pertinentes. Nous avons particulièrement apprécié leur soutient sans relâche ainsi que leurs critiques constructives qu'elles nous ont fournies à tout moment du déroulement de ce travail.

Nous tenons également à remercier tous les membres de jury pour avoir accepter d'évaluer notre travail. Nous adressons nos sincères remerciements à tous ceux qui ont contribué par de diverses manières à l'aboutissement de ce travail. De tout cœur nous exprimons nos profondes gratitudes aux membres de nos familles, nos parents, nos frères, sœurs et petits neveux, pour leur soutien tout au long de notre parcours. En dernier lieu, nous pensons à tous nos amis qui nous ont soutenu d'une manière constante, et aux personnes chères à nos yeux qui veillent sur nous tout là haut.

KENOUCHE ikhlef & BENZIANE Maathouk

⁕ *Dédicaces* ⁕

Je dédie ce travail :
À mes très chers parents, quoi que je fasse ou que je dise, je ne pourrais pas te remercier comme il le doit, ils ont jamais cessé, de formuler des prières à mon égard, de me soutenir et de m'épauler pour que je puisse atteindre mes objectifs.
À mes chers frères et mon unique sœur Kahina qui n'ont pas cessé de me conseiller, encourager et soutenir tout au long de mes études. Que Dieu les protège et leurs offre la chance et le bonheur.
À mon cher binôme M.Benziane , que je vais jamais oublier nos souvenirs et tous les moments qu'on a passer ensemble, son soutien moral, sa patience et sa compréhension tout au long de ce projet, que dieu te bénisse et te protège.

M. ***KENOUCHE Ikhlef***

⁂ *Dédicaces* ⁂

Je dédie ce travail :
À mes très chers parents qui ont toujours été là pour moi et qui m'ont données un magnifique modèle de labeur et de persévérance.
À ma sœur **Nadjet** qui m'a toujours soutenu dans n'importe quelle situation.
À mon ami **Omar** avec qui j'ai partagé des agréable moments, Que dieu réunisse nos chemins pour un long commun serein.
À mon binome **KENOUCHE Ikhlef** pour avoir était un personne de travail exceptionnelle.
Je vous dois ce que je suis aujourd'hui grâce à votre amour, à votre patience et vos innombrables sacrifices.
Que ce modeste travail, soit pour vous une petit compensation et reconnaissance envers ce que vous avez fait d'incroyable pour nous. Qu'allah, le tout puissant, vous préserve et vous procure santé et longue vie afin que je puisse à mon tour vous combler.

M. ***BENZIANE Maathouk***

Table des matières

Table des matières i

Liste des figures v

Liste des tableaux vii

Liste des abréviations viii

Introduction générale 1

1 Cloud Computing & Méthodes Agiles 3

1.1 Introduction . . . 3

1.2 Cloud Computing . . . 3

1.2.1 Définition . . . 3

1.2.2 Caractéristiques du cloud computing . . . 4

1.2.3 Types de services du cloud . . . 5

1.2.3.1 IaaS (Infrastructure as a service) . . . 6

1.2.3.2 PaaS (Platform as a service) . . . 6

1.2.3.3 SaaS (Software as a service) . . . 7

1.2.4 Modèles de déploiement du cloud computing . . . 7

1.2.4.1 Cloud privé . . . 7

1.2.4.2 Cloud public . . . 8

1.2.4.3 Cloud hybride . . . 8

1.2.4.4 Cloud communautaire . . . 9

1.2.5 Les challenges dans le cloud computing . . . 9

1.2.5.1 Qualité de service (QoS) . . . 9

1.2.5.2 Gestion et maitrise des couts . . . 10

1.2.5.3 Interoperabilité . . . 10

1.2.5.4 Confidentialité et sécurité . . . 10

1.3 Méthodes Agiles . . . 11

1.3.1 Définition et principes . . . 11

1.3.2 Différence entre les approches traditionnelles et agiles . . . 12

1.3.3 Supports et outils agiles . . . 13
1.3.3.1 Outils de communication et collaboration . . . 13
1.3.3.2 Outils pour la gestion du projet . . . 13
1.3.3.3 Outils de Développement et stockage . . . 13
1.3.4 Principales méthodes . . . 14
1.3.4.1 RAD : Développent Rapide d'application . . . 14
1.3.4.2 Développement de logiciels adaptatifs (ASD : Adaptive Software Development) . . . 16
1.3.4.3 eXtreme Programming . . . 16
1.3.4.4 Processus Unifié (PU) ou Unified Porcessing (UP) . . . 19
1.3.4.5 Scrum . . . 20
1.3.5 Combiner Agile avec le Cloud computing . . . 22
1.3.5.1 Synthèse . . . 22
1.3.5.2 Avantages . . . 23
1.4 Conclusion . . . 24

2 Etat de l'art & Proposition **25**
2.1 Introduction . . . 25
2.2 Quelques travaux existants . . . 25
2.2.1 Travail de Gaurav et al [27] . . . 25
2.2.1.1 Description . . . 25
2.2.1.2 Discussion . . . 26
2.2.2 Travail de Chung Yung et Yu-Tang Lin [39] . . . 27
2.2.2.1 Description . . . 27
2.2.2.2 Discussion . . . 28
2.2.3 Travail de YOUNAS et al [38] . . . 29
2.2.3.1 Description . . . 29
2.2.3.2 Discussion . . . 30
2.2.4 Travail de E.Toews et al [31] . . . 31
2.2.4.1 Description . . . 31
2.2.4.2 Discussion . . . 32
2.2.5 Étude comparative des travaux existants . . . 32
2.3 Description globale du Framework Agile-Cloud proposé . . . 34
2.3.1 Présentation des acteurs du Framework . . . 36
2.3.2 Description détaillée des différentes phases du Framework . . . 37
2.3.2.1 Description de la Pré-phase . . . 37
2.3.2.2 Description de la Phase de développement . . . 38
2.3.2.3 Description de la post-phase . . . 39
2.3.2.4 Revue d'itération (Sprint Review) . . . 40

2.3.2.5 Rétrospective d'itération (Sprint Retroscpective) 40
2.4 Conclusion 41

3 Validation du framework proposé par une étude de cas : Gestion de patients 42
3.1 Introduction 42
3.2 Outils de développement 42
3.2.1 Draw.io 43
3.2.2 JIRA 43
3.2.3 Android studio 44
3.2.4 GitHub 44
3.3 Langages de programmation 45
3.3.1 Java 45
3.3.2 PHP 45
3.3.3 XML : Extensible Markup Language 45
3.4 Implémentation de la base de données 46
3.4.1 MYSQL 46
3.5 Bibliothèques utilisées 46
3.5.1 Volley 46
3.6 Mise en application du Framework proposé 46
3.6.1 Phase 1 : Pré-phase (Phase de planification) 47
3.6.1.1 Étape 1 : Collection des user stories 47
3.6.1.2 Étape 2 : Planification des itérations (Planning sprint) . . . 47
3.6.2 Phase 2 : Phase de développement 51
3.6.2.1 Étape 1 : Conception 51
3.6.2.1.1 Sprint 1 : Authentification 51
3.6.2.1.2 Gestion des comptes des utilisateurs de l'application 53
3.6.2.1.3 Sprint 3 : Gestion des rendez-vous 57
3.6.2.1.4 Sprint 4 : Consulter planning rendez-vous/Gestion des dossiers médicaux/ Gestion des listes patients/ Consulter dossier médical. 60
3.6.2.1.5 Sprint 5 : Consulter la disponibilité des rendez vous et mon dossier médicale 72
3.6.2.2 Étape 2 : Implémentation 82
3.6.2.3 Étape 3 : Test unitaire 83
3.6.2.4 Etape 4 : Validation 84
3.6.2.4.1 Sprint 1 84
3.6.2.4.2 Sprint 2 86
3.6.2.4.3 Sprint 3 87

3.6.2.4.4 Sprint 4 88
3.6.2.4.5 Sprint 5 91
3.6.3 Phase 3 : Post- phase 92
3.6.3.1 Étape 1 :Test d'intégration 92
3.6.3.2 Étape 2 :Test d'acceptation utilisateur 93
3.7 Conclusion 94

Conclusion générale **95**

Bibliographie **97**

Table des figures

1.1 Les services du Cloud Computing [33]. 4
1.2 Les 3 couches du Cloud Computing [16]. 5
1.3 Les modèles de services du Cloud Computing [26]. 6
1.4 Les modèles de déploiement du Cloud Computing [5]. 9
1.5 Modèle Développent Rapide d'application [14]. 15
1.6 Adaptive Software Development cycle de vie [6]. 16
1.7 Processus XP au niveau du projet [32]. 19
1.8 Un processus itératif et incrémental. Chaque itération a pour finalité une version exécutable [35]. 20
1.9 Schéma de déroulement d'un Sprint [7]. 21
1.10 Cycle Scrum basé sur 5 Sprints – Itérations [7]. 22
1.11 Des artefacts pour montrer le développement agile dans le cloud computing [37]. 23

2.1 Agile SCRUM process Over Cloud.[27]. 26
2.2 The overall architecture of TOAST.[39]. 28
2.3 Configuration expérimentale pour la mise en œuvre du Framework.[38]. . . . 30
2.4 Agile-Cloud Framework. 35
2.5 Description d'une user story (modèle d'une user story). 37

3.1 Configuration expérimentale pour la mise en œuvre du Framework. 43
3.2 L'architecture d'une application mobile. 44
3.3 La durée d'essai gratuite pour JIRA 50
3.4 Cloud product backlog. 50
3.5 Diagramme de cas d'utilisation Authentification. 51
3.6 Diagramme de séquence "Authentification" 53
3.7 Diagramme de cas d'utilisation «Gestion des comptes des personnels médicaux». 53
3.8 Diagramme de séquence de «Gestion des comptes personnels médicaux». . . 56
3.9 Diagramme de séquence de cas d'utilisation «Gestion des rendez-vous patients». 57
3.10 Diagramme de séquence du cas d'utilisation «Gestion des rendez-vous». . . 59
3.11 Diagramme de cas d'utilisation «Consulter planning des RDV». 60

3.12 Diagramme de séquence «Consulter planning des RDV». 61
3.13 Diagramme de cas utilisation «Gestion des dossiers médicaux». 61
3.14 Diagramme de séquence du cas d'utilisation «Gestion des dossiers médicaux». 66
3.15 Diagramme cas utilisation «Gestion de la liste des patients» 67
3.16 Diagramme de séquence du cas d'utilisation «Gestion de la liste des patients». 69
3.17 Diagramme cas utilisation «Consulter dossier médical». 70
3.18 Diagramme de séquence du cas d'utilisation «Consulter dossier médical». . . 72
3.19 Diagramme de cas d'utilisation «Consulter disponibilité des RDV». 73
3.20 Diagramme de séquence du cas d'utilisation «Consulter disponibilité des RDV». 74
3.21 Diagramme séquence de cas utilisation «Consulter mon dossier médical». . . 75
3.22 Diagramme de séquence du cas utilisation «Consulter mon dossier médical». 76
3.23 Diagramme de cas d'utilisation global de notre application gestion des patient. 77
3.24 Diagramme de classe globale de notre application mobile. 78
3.25 Équivalence entre objet et relationnel 80
3.26 Exemple codage sur "Edit_Patient". 82
3.27 La partie code partagé sur GitHub. 83
3.28 Exemple d'utilisation de Appium. 84
3.29 Interface d'accueil. 85
3.30 Interface de connexion. 86
3.31 Espace administration. 87
3.32 Espace sécuritaire. 88
3.33 Espace médecin. 89
3.34 Interface ajoute fiche et ordonnance. 90
3.35 Espace infirmier. 91
3.36 Notification pour le médecin de service 91
3.37 Espace patient. 92
3.38 Exemple test d'intégration effectué pour "Authentification". 93

Liste des tableaux

1.1 Liste des services en fonction des couches du cloud [12]. 7
1.2 Différences entre les méthodes agiles et traditionnelles [18]. 12

2.1 Tableau comparatif des travaux. 33
2.2 Tableau collecte l'ensemble d'outils cloud agile. 40

3.1 Backlog du produit. 49
3.2 Description textuelle du cas utilisation « Authentification ». 52
3.3 Description textuelle du cas utilisation «Gestion des comptes personnels médicaux». 55
3.4 Description textuelle du cas d'utilisation «Gestion des rendez-vous». 58
3.5 Description textuelle du cas utilisation «Consulter planning RDV». 60
3.6 Description textuelle du cas utilisation «Gestion des dossiers médicaux». . . 64
3.7 Description textuelle du cas utilisation «Gestion de la liste des patients». . . 68
3.8 Description textuelle du cas utilisation «Consulter dossier médical». 71
3.9 Description textuelle du cas utilisation «Consulter disponibilité des RDV».. . 73
3.10 Description textuelle du cas utilisation «Consulter mon dossier médical». . . 75
3.11 Dictionnaire de données. 80
3.12 Formulaire de test d'acceptation. 94

Liste des abréviations

ASD Adaptive software développement

API Application Programming Interface

ASD Adaptive Software Deve-lopment

AWS Amazon Web Services

ADCC Développement Agile et du Cloud Computing

BI Business Intelligence

CESWP Cloud-Enabled Space Weather Platform

CPU Central processing unit

CSCI Computer Software Configuration Item

CSP CloudService Provider

DSDM Dynamic Systems Development Method

FDD Fea-ture Driven Development

JSP Java Server Pages

HTTP Hypertext Transfer Protocol

HMS Hopital Médical Système

IaaS Infrastructure as a service

IHM Interface homme-machine

IDE Integrated development environment

GAE Google App Engine

MOAT Model Operation of agile tasks

MYSQL My Structured Query Language

MyRen Malaysia Research and education network

NoSQL Not only Structured Query Language

OS Operating System

PaaS Platform as a Service

PU	Processus Unifié
RAD	Développent Rapide d'application
SaaS	SaaS Software as a Service
SDK	Sotfware Development Kit
SGML	Standard Generalized Markup Language
TI	Technologies de l'information
Taas	Test as a Service
TOAST	Tool for Agile Software Technology
UML	Unified Modeling Language
UP	Unified Process
VM	virtual machines
WEB	World Wide Web
XML	Extensible Markup Language
XP	EXtreme Programming

Introduction générale

Les méthodes de développement logiciel constituent un des facteurs importants dans le succès des projets. En occurrence, l'utilisation de processus agile et l'implication des utilisateurs (pratique de l'agilité) sont deux facteurs majeurs dans le succès des projets TI (Informations Technologies). la méthode Scrum est devenue de nos jours de plus en plus adoptée dans les équipes de développement. Cette méthode "agile" permet la réalisation de projets complexes en favorisant l'interaction avec les membres de l'équipe et les managers, la collaboration du client et la réactivité face aux changements.

Le cloud computing «L'informatique en nuage» est la fourniture de services informatiques (notamment des serveurs, du stockage, des bases de données, la gestion réseau, des logiciels, des outils d'analyse, l'intelligence artificielle) via Internet dans le but d'offrir une innovation plus rapide, des ressources flexibles et des économies d'échelle.

Les données et les applications sont localisées dans le cloud, ce qui facilite la mise à l'échelle rapide des applications. Ces dernières sont pour la plupart développées suivant les méthodes de développement logiciel actuelles (méthodes agiles : XP (eXtreme Programming), Scrum, UP (Unified Process), DSDM (Dynamic Systems Development Method), FDD (Feature Driven Development), ASD (Adaptive software développement) et celles traditionnelles ou hybrides. Ces applications sont déployées dans le cloud.

À cet effet, il est important d'étudier le processus de développement des applications dans le cloud afin d'adapter les méthodes agiles usuelles pour assurer le succès des projets de développement et accroître la satisfaction des clients et utilisateurs (un aspect essentiel dans le principe de l'agilité).

Également, l'utilisation des services de cloud computing est considérée comme une solution pour certains problèmes comme les pertes des données, de temps dans le déploiement des livrables à chaque itération.

Notre cas d'étude sur la gestion des dossier médicaux en combinant la méthode agile Scrum avec le cloud ,apporte des avantages et des outils utiles à la prise de décision, un accès omniprésent aux données médicales des patients et développe la possibilité et l'esprit de travailler en équipe.

Les professionnels de santé doivent pouvoir communiquer entre eux, de manière sécurisée, au sujet d'un patient et ses soins. Le dossier médical personnel informatisé joue à présent un rôle important dans la coordination des traitements en favorisant l'échange et le partage des documents médicaux, il contribue de façon plus générale à l'amélioration de la qualité des soins en facilitant l'accès, sans perte du temps, aux données nécessaires à la prise de décision professionnel.

Le cloud computing, permet d'offrir à des coûts assez bas aux utilisateurs (entreprises, clients, notre cas : Gestion des patients pour la clinique privé) des ressources informatiques mutualisées et accessibles de n'importe où n'importe quand. Il permet donc aux développeurs logiciel et web d'offrir plus facilement et à moindre coût aux clients et utilisateurs des applications dédiées à tout instant à travers les navigateurs web.

C'est à cet égard que ce mémoire est consacré à la combinaison du cloud avec la méthode agile pour réaliser une application mobile afin d'assurer la gestion des patients dans une clinique privée. La méthode Scrum qui est l'une des plus utilisées actuellement est adaptée aux contextes du développement de cette application dans le cloud. Le but principal de cette recherche est de vérifier l'applicabilité du développement agile de notre application mobile dans le cloud. Ce qui conduit aux objectifs suivants :

(1) Démontrer les avantages d'utilisation des méthodes agiles dans le cloud, aussi (2) analyser et déterminer les couches (SaaS, PaaS et IaaS) concernées par le développement, et concevoir une méthode Scrum adaptée aux contextes de développement de notre application mobile.

Afin d'aboutir à la réalisation de notre application notre travail est reparti en trois chapitres qu'on peut décrire comme suit :

Le premier chapitre présente une vue d'ensemble sur les différentes notions fondamentales du Cloud Computing et de développement logiciel par les méthodes agiles.

Le deuxième chapitre décrit dans sa première partie quelques travaux existants dans le domaine qui combine le cloud computing et les méthodes agile. La deuxième partie présente une description détaillée du framework proposé.

Le troisième chapitre présente le contexte applicatif de notre travail, à savoir la gestion des dossiers médicaux des patients. Ce chapitre a pour objectif de valider notre cadre méthodologique et de montrer sa faisabilité.

Chapitre 1

Cloud Computing & Méthodes Agiles

1.1 Introduction

Actuellement, une nouvelle "tendance" a fait son apparition dans le monde de l'IT (information Technologies), il s'agit du Cloud Computing. Cette technologie, s'appuie sur le WEB 2.0, offre des occasions aux sociétés de réduire les coûts d'exploitation des logiciels par leurs utilisations directement en ligne.

Ce chapitre montre les différentes notions fondamentales du Cloud Computing et de développement logiciel par les méthodes agiles. Dans la première partie de ce chapitre, nous allons présenter l'informatique en nuage, ses caractéristiques et types de services, les challenges et les modèles de déploiement. La deuxième partie du chapitre présente les différentes méthodes qualifiées d'agiles, ses outils et ses principales méthodes, usuelles dans le domaine du développement logiciel. Terminant ce chapitre par une synthèse, en combinant l'agile avec le Cloud Computing.

1.2 Cloud Computing

1.2.1 Définition

Le Cloud Computing ou « l'informatique dans les nuages », on retrouve comme synonyme l'informatique dématérialisée.

Le Cloud Computing est un modèle permettant un accès pratique et sur demande à un pool partagé de ressources informatiques configurables (réseaux, serveurs, stockage, applications et services) qui peuvent être provisionnés rapidement et libérés avec un effort de gestion minimale ou interaction de fournisseur de service[23].

Toute fois l'idée principale à retenir est que le Cloud n'est pas un ensemble de technologies, mais un modèle de fourniture, de gestion et de consommation de services et de ressources informatiques.

La figure ci-dessous (1.1) présente les services du Cloud Computing.

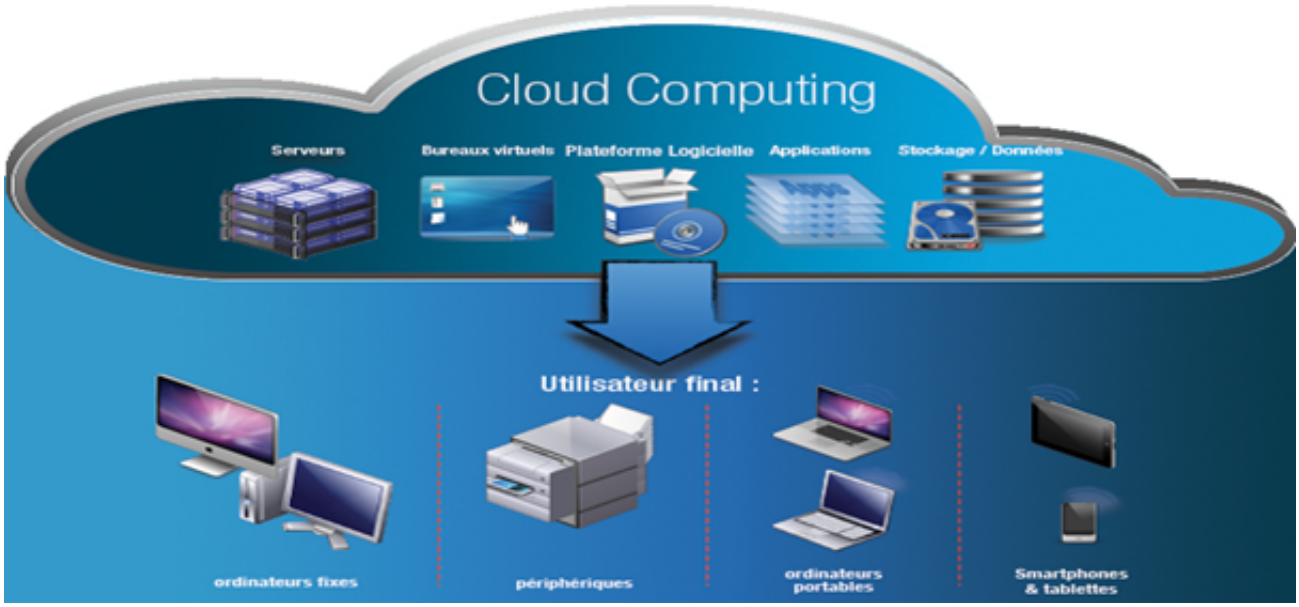


FIGURE 1.1 – Les services du Cloud Computing [33].

1.2.2 Caractéristiques du cloud computing

Le Cloud Computing est un modèle essentiellement caractérisé par :

- ***Service à la demande automatisé :***
  Le client gère et consomme les ressources informatiques sans l'intervention humaine du fournisseur de services.
- ***Élasticité rapide :***
  Les capacités des applications pour passer à l'échelle de manière dynamique et rapide, peuvent être augmenter ou diminuer en fonction des besoins . En ce qui concerne le consommateur, les ressources utilisées semblent être illimitées et peuvent être louées à n'importe quel moment et en quantité illimitée.
- ***L'accès via le réseau :***
  Les ressources sont disponibles sur le réseau et accessibles par des mécanismes standards, qui favorisent l'accès au service par des clients lourds ou légers via des plates-formes hétérogènes, tels que les téléphones mobiles, tablettes, ordinateurs portables, poste de travail.
- ***Un service sur mesure :***
  Un contrôle automatique et une optimisation des ressources allouées par rapport à une moyenne estimée de consommation du service. L'utilisation des ressources peut être gérée, contrôlée et communiquée, fournissant ainsi de la transparence au client

et au fournisseur. Ce mécanisme s'opère à un certain niveau d'abstraction approprié pour le type de service utilisé (par exemple, stockage, traitement, bande passante).

- ***La mise en commun des ressources :***
  Les ressources de calcul sont mises à disposition des clients sur un modèle multi-locataires, avec différentes ressources physiques et virtuelles (ressources de stockage, le traitement, la mémoire, et la bande passante du réseau). Le client n'a généralement aucun contrôle ou connaissance sur l'emplacement exact des ressources fournies.

1.2.3 Types de services du cloud

L'architecture des services du Cloud est subdivisée en trois couches essentielles ;

- *Application en tant que service (**SaaS**, software as a service).*
- *Plate-forme en tant que service (**PaaS**, Platform as a service).*
- *Infrastructure en tant que service (**IaaS**, Infrastructure as a service) .*

Ces trois catégories de service doivent être déployés sur des infrastructures qui possèdent les cinq caractéristiques essentielles citées plus haut pour être considérées comme du Cloud Computing [20]. La Figure (1.2) ci-dessous représente les différentes couches du cloud computing de la couche la moins visible pour les utilisateurs finaux à la plus visible.

FIGURE 1.2 – Les 3 couches du Cloud Computing [16].

L'infrastructure as a Service (IaaS) est plutôt gérée par les architectes réseaux, la couche PaaS est destinée aux développeurs d'applications et finalement le logicielle comme un service (SaaS) est le « produit final » pour les utilisateurs [20].

La Figure (1.3) représente les différents modèles de services du cloud computing ;

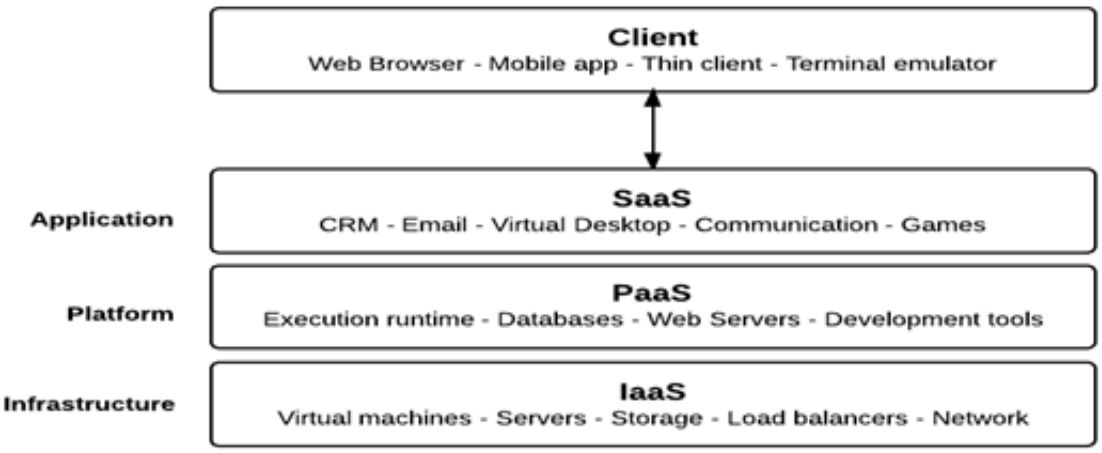


FIGURE 1.3 – Les modèles de services du Cloud Computing [26].

Afin de mieux comprendre les trois termes ci-dessous nous allons parler de chacune de ces catégories indépendamment, on a :

1.2.3.1 IaaS (Infrastructure as a service)

C'est la couche de base de l'informatique en nuage. Concerne la mise à disposition de ressources informatiques virtualisées et déportés (puissance CPU, mémoire, stockage, réseau), sont fournies à la demande du client pour le déploiement et l'exécution de ses applications, ainsi que sur certains composants réseau (pare-feu, par exemple), et le contrôle sur le système d'exploitation (OS), le stockage [19] [20].
Exemples de fournisseurs de service IaaS : Amazon EC2, Windows Azure, Google Compute Engine et Rackspace.
Avantages : Administration, Personnalisation, flexibilité d'utilisation, Capacité de stockage infini.
Inconvénients : Sécurité, Besoin d'un administrateur système, demande pour les acteurs du cloud des investissements très élevé.

1.2.3.2 PaaS (Platform as a service)

Elle est au-dessus de la couche de service IaaS. C'est la plateforme d'exécution, de déploiement et de développement des applications, elle fournit aux développeurs des outils tels que les langages de programmation, les API, les bibliothèques nécessaires pour la création et le déploiement des applications dans l'informatique en nuage.
L'utilisateur ne gère pas ou ne contrôle pas l'infrastructure Cloud sous-jacente (réseaux, serveurs, stockage) mais il contrôle l'application déployée et sa configuration [19] [20].
Exemples de fournisseurs de service PaaS : tels que Salesforce.com, Google App Engine,

CloudBees, Heroku et CloudFoundry.
Avantages : Pas d'infrastructure nécessaire, Pas d'installation, Environnement hétérogène.
Inconvénients : Limitations des langages, Pas de personnalisation dans la configuration des machines virtuelles.

1.2.3.3 SaaS (Software as a service)

C'est la dernière couche de service proposé par l'informatique en nuage. Des applications sont mises à la disposition des consommateurs. Les applications peuvent être manipulées à l'aide soit d'une interface client légère, comme un navigateur Web, ou d'une interface de programme. Et le consommateur n'a pas à se soucier d'effectuer des mises à jour, d'ajouter des patches de sécurité et d'assurer la disponibilité du service [19] [20].
Exemples de fournisseurs de service SaaS tels que ; Gmail, Google Docs, DrpoBox.
Avantages : administration, plus de licence, Migration, accessible via un abonnement.
Inconvénients : logiciel limité, sécurité, dépendance des prestataires .
Ce tableau résume la liste des services en fonction des couches du cloud ;

Cloud services Catégories	Le type IaaS	Le type PaaS	Le type SaaS
Compute as a Service	X		
Communication as a Service		X	X
Data Storage as a Service	X	X	X
Infrastructure as a Service	X		
Network as a Service	X	X	X
Platform as a Service		X	
Software as a Service			X

Tableau 1.1 – Liste des services en fonction des couches du cloud [12].

Pour simplifier ces différentes définitions, on peut retenir qu'avec le SaaS on utilise une application, avec le PaaS on fournit un Framework d'application et finalement l'IaaS permet d'héberger le tout.

1.2.4 Modèles de déploiement du cloud computing

Nous distinguons quatre modèles de déploiement du Cloud Computing :

1.2.4.1 Cloud privé

La notion de « cloud privé » peut-être divisée en deux : le cloud privé interne et le cloud privé externe [12].

Cloud privée interne

Déployé uniquement au sein d'une entreprise, elle doit gérer toute seule son infrastructure (interne à l'entreprise), ou Cloud entièrement dédié à cette même entreprise, accessible via des réseaux sécurisés, opéré par les équipes internes.

Cloud privée externe

Dédiés à une seule entreprise mais dont la gestion est externalisée à un prestataire, c'est à dire : vise à fournir, de manière externalisée, les services et garanties équivalents à ceux offerts par un Cloud privé interne, tout en bénéficiant des avantages des services de gestion par un tiers. Il peut être accessible par Internet ou par un réseau privé [12] [13]. Le cloud privé est un modèle :

- Cher pour le client.
- Dédier et sécurisé.
- Moins flexible au cloud public.

1.2.4.2 Cloud public

Il est externe à l'entreprise, est mis à la disposition du grand public ou à de grands groupes industriels, accessible via Internet ou un réseau privé, peut être gratuit ou fonctionner selon le paiement à la consommation, ce qui fait y aura pas de gaspillage de ressources. Géré par un opérateur externe propriétaire des infrastructures appelé le fournisseur nuage, avec des ressources totalement partagées entre tous ses clients [12] [24].

Le cloud public est un modèle qui :

- Demande de lourds investissements pour le fournisseur de services.
- Offre un maximum de flexibilité.
- N'est pas sécurisé.

1.2.4.3 Cloud hybride

C'est la combinaison de deux ou plusieurs nuages, qu'ils soient privés ou publics, partageant des données et des applications, un Cloud dédié pour les applications (exporter des applications dans un cloud public) , et l'autre pour les données (ces applications utiliseront également des données stockées sur un cloud privé) [24].

Le cloud hybride est un modèle :

- Permet d'allier les avantages des deux modèles de déploiement.
- Permet la gestion de deux Cloud qui peut s'avérer plus contraignant.

La Figure ci-dessous représente les différents types du cloud computing :

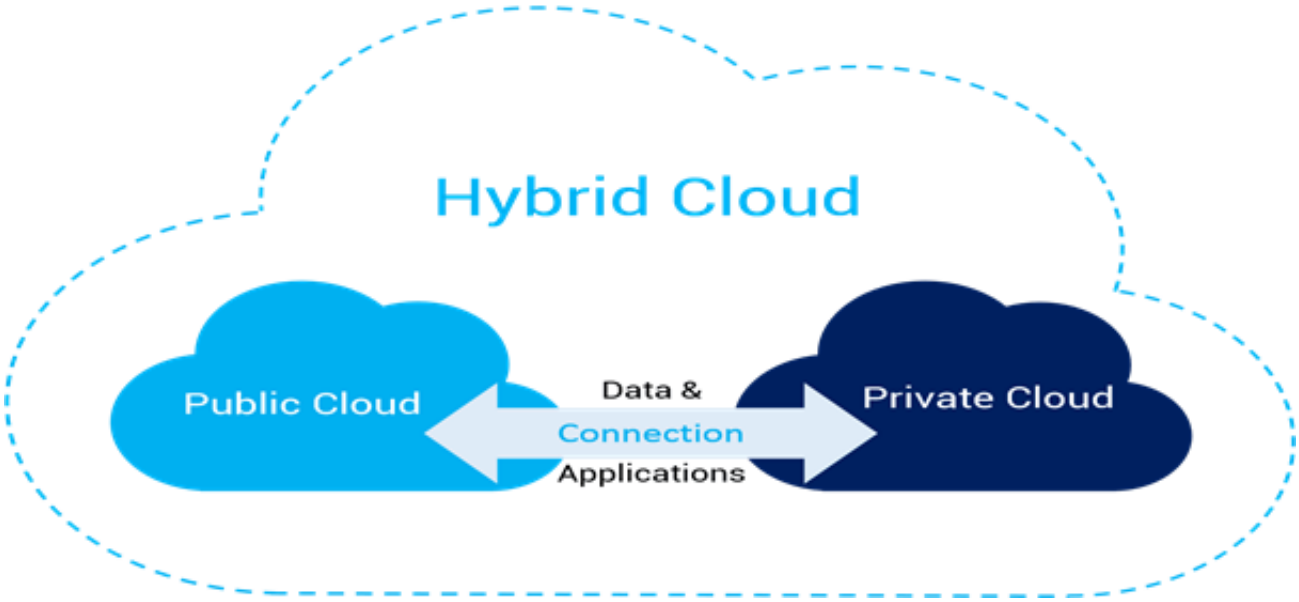


FIGURE 1.4 – Les modèles de déploiement du Cloud Computing [5].

1.2.4.4 Cloud communautaire

L'infrastructure est partagée par plusieurs organisations qui ont des intérêts communs (par exemple les exigences de sécurité, de conformité, la mission). Il peut être géré par les organisations elles-mêmes ou par un tiers.

1.2.5 Les challenges dans le cloud computing

Le cloud computing est utilisé pour permettre un accès global à des pools de ressources mutuels tels que des services, des applications, des données, des serveurs et des réseaux informatiques. Cela se fait sur un serveur tiers situé dans un centre de données ou dans un cloud privé. Cela rend les dispositifs d'accès aux données plus fiables et plus efficaces, avec un effort d'administration minime.

Malgré tout le développement et le potentiel des services de cloud computing, les entreprises doivent relever de nombreux défis liés aux services de cloud computing, afin de réaliser les avantages. Ici, nous avons compilé une liste des défis du cloud computing qui doivent être pris en charge, pour tirer parti de la capacité maximale du cloud. Commençons :

1.2.5.1 Qualité de service (QoS)

Sur le flux internet, il n'est pas possible de garantir un niveau donné de qualité de service au niveau applicatif, même occasionner des pertes de productivité. Pour éviter ces difficultés et garantir un accès constant aux applications hébergées dans le cloud, la solution

est d'établir une interconnexion privée, entre le réseau de l'entreprise et celui du « Cloud Service Provider » (CSP). Concrètement, il s'agit de bâtir une liaison entre le réseau Azure, Amazon ou Cegid par exemple et celui du client final de même nature que celle qui relie déjà deux de ses sites distants.

1.2.5.2 Gestion et maitrise des couts

Le cloud computing vous permet d'accéder aux logiciels d'application via une connexion Internet rapide et vous permet d'économiser sur les investissements coûteux en matériel informatique, en logiciels, en gestion et en maintenance. Le réglage de la plate-forme en fonction des besoins de l'entreprise peut être coûteux, le coût du transfert de données vers un cloud public, en particulier pour une petite entreprise ou un projet est très couteux.

Pour la maîtrise des coûts du cloud , par exemple, en optimisant les coûts en effectuant de meilleures analyses et rapports financiers , en automatisant les politiques de gouvernance ou en maintenant la pratique des rapports de gestion sur la bonne voie, afin que ces problèmes dans le cloud computing puissent être réduits.

1.2.5.3 Interoperabilité

Depuis l'émergence du cloud, les entreprises utilisatrices se posent la question de l'interopérabilité entre les différents services cloud. Cette préoccupation n'est pas anodine. Pour nombre d'entreprises, il s'agit d'éviter les phénomènes de verrouillage (vendor « lock-in »), et rendre fluide la migration de ressources d'un cloud vers un autre. Pour d'autres, il s'agit de simplifier la constitution d'architectures hybrides visant à mixer ressources internes et ressources cloud.

Pour certains, il s'agit d'assurer la portabilité des données d'un cloud à un autre ou du data center de l'entreprise vers le cloud et vice-versa. Enfin d'autres encore rêvent de la possibilité de coordonner de façon transparente l'orchestration de ressources distribuées sur de multiples cloud, et donc sur des infrastructures utilisant souvent des APIs très différentes pour permettre leur pilotage.

1.2.5.4 Confidentialité et sécurité

Les problèmes de sécurité dans le cloud computing constituent la principale préoccupation de l'investissement dans les services cloud en 2018, comme 77 % des répondants l'ont déclaré dans l'enquête référée. Pendant longtemps, le manque de ressources / d'expertise a été le défi numéro un du cloud. En 2018, cependant, la sécurité a progressé légèrement, chaque jour ou l'autre, rencontré des authentifications cassées, des informations d'identification compromises, de piratage de compte, de violations de données..etc.

Suite aux efforts des fournisseurs de cloud, pour améliorer les capacités de sécurité. Pour garantir la sécurité d'une organisation on doit s'assurer que, le fournisseur SaaS a mis en place des mécanismes sécurisés de gestion de l'identité des utilisateurs, d'authentification et de contrôle d'accès. Vérifiez également les lois sur la confidentialité et la sécurité des bases de données auxquelles ils sont soumis, Comme il est important de mettre en œuvre un outil cloud BI sécurisé capable de tirer parti des mesures de sécurité appropriées.

1.3 Méthodes Agiles

1.3.1 Définition et principes

Les projets de développement logiciel sont souvent caractérisés par des besoins non totalement définis et exiger des modifications du projet à tout moment. Cette complexité a conduit progressivement à l'adoption des méthodes agiles dans le domaine Génie logiciel, Les méthodes agiles visent fondamentalement la grande satisfaction des parties prenantes du projet (les clients, les utilisateurs, les membres d'équipe, la direction, etc.).
Suivant le «manifeste agile», on retient la définition suivante : « Une méthode agile est une approche itérative et incrémentale pour le développement de logiciel, réalisée de manière très collaborative par des équipes responsabilisées, appliquant un cérémonial minimal, qui produisent, dans un délai contraint, un logiciel de grande qualité répondant aux besoins changeants des utilisateurs »
Selon le manifeste agile, une méthode agile possède quatre valeurs et 12 principes [7] :

Valeurs

Les méthodes agiles prônent 4 valeurs fondamentales :

1. Individus et interactions plutôt que processus et outils.
2. Fonctionnalités opérationnelles plutôt que documentation exhaustive.
3. Collaboration avec le client plutôt que contractualisation des relations.
4. Acceptation du changement plutôt que le suivi d'un plan.

Principes

1. Satisfaire le client en livrant rapidement et régulièrement des logiciels utiles, qui offrent une véritable valeur ajoutée.
2. Accueillir favorablement les demandes de changement, même tard dans le projet. Les processus Agiles exploitent le changement pour donner un avantage compétitif au client.
3. Livrer le plus souvent possible des versions opérationnelles de l'application.
4. Assurer une coopération permanente entre le client et l'équipe projet.

5. Construire des projets autour d'individus motivés, Fournissez-leur l'environnement et le soutien, en leur faisant confiance pour atteindre les objectifs fixés.
6. Privilégier la conversation en face à face.
7. Mesurer l'avancement du projet en termes de fonctionnalités de l'application
8. Faire avancer le projet à un rythme soutenable et constant.
9. Porter une attention continue à l'excellence technique et à la conception.
10. La simplicité, minimiser la quantité de travail inutile, est essentielle.
11. Laisser l'équipe s'auto organise.
12. À intervalles réguliers, réfléchir aux moyens de devenir plus efficace.

1.3.2 Différence entre les approches traditionnelles et agiles

Les méthodes traditionnelles et les méthodes agiles ont plusieurs point fort et limitation, Le tableau1.2 résume les principales différences entre ces deux approches

Critères	**Méthodes traditionnelles**	**Méthodes agiles**
Approche	prédictive	adaptative
Mesure du succès	conformité au plan	valeur client
Domaine	prévisible	imprévisible/exploratoire
Retour sur investissement	à la fin du projet	au début du projet
Style de management Culture	autocratique commande, contrôle	décentralisé leadership, collaboration
Environnement	stable, faible taux de changement	turbulent, taux élevé de changement
Relation client	interaction au besoin, insistance sur les exigences du contrat	un représentant du client est dévoué au projet, insistance sur la priorisation des incréments
perspective de changement	durabilité de changement	adaptabilité au changement
Focus	processus	humain
Communication	explicite	tacite
Documentation	lourde	faible
Cycles	en nombre limité	nombreux
Planification en amont	complète	minimale
Taille du projet	grande taille	petite taille
Taille de l'équipe	grande taille	petite taille/créative
Développement	conception extensive, longs incréments	conception simple, courts incréments

Tableau 1.2 – Différences entre les méthodes agiles et traditionnelles [18].

1.3.3 Supports et outils agiles

1.3.3.1 Outils de communication et collaboration

Dropbox
Prend en charge plusieurs utilisateurs, fournit un espace de stockage par utilisateur, administration centralisée, surveillance des activités et capacité pour suivre et récupérer les versions précédentes des fichiers.
Google drive
Un service de stockage et de partage de fichiers dans le cloud lancé par la société Google. Il sert à synchroniser, partager et modifier les données entre plusieurs ordinateurs et/ou utilisateurs.
Google meet
Un service de visioconférence développé par Google, avec La possibilité d'enregistrer une conférence, partage d'écran.

1.3.3.2 Outils pour la gestion du projet

Microsoft Azure
C'est un service de cloud computing créé par Microsoft pour créer, tester, déployer et gérer des applications et des services , Il offre (Saas, Paas, Iaas), et prend en charge de nombreux langages de programmation, outils et frameworks.
Jira
Un service de gestion de projet destiné au développement logiciel, Avant de commencer un projet, l'utilisateur devra choisir la méthode de gestion de projet souhaitée entre Scrum et Kanban, Jira offre une vision globale de l'état d'avancement d'un projet. Et permet de créer et de répartir toutes sortes de tickets (que cela soit une tâche à réaliser ou un bug à corriger), la progression des taches est facile a identifié.
Amazone web service
AWS est un service de stockage dans le cloud permet aux entreprises, aux gouvernements et aux particuliers de stocker leurs données et propose des API (Artificial Interface de programmation) [8].

1.3.3.3 Outils de Développement et stockage

Google App Engine
Google App Engine est une plate-forme de cloud computing pour développer des applications Web. Propose des outils pour coder des applications sans se soucier de gestion du serveur ou configurations matérielles car ceci est géré par le moteur, et tester l'application localement. Une fois l'application est terminée, GAE fournit un

outil de gestion qui donne le contrôle de l' application à l'aide d'un simple tableau de bord basé sur le Web. Et offre toutes les ressources informatiques dont l'application a besoin, le développeur doit payer pour plus des ressources après certain nombre d'utilisation [22].

Eclipse IDE

Un environnement de développement intégré libre, permettant potentiellement de créer des projets de développement mettant en œuvre n'importe quel langage de programmation.

GitHub

Un site web d'hébergement et de gestion, un service de cloud facilite la collaboration entre plusieurs intervenants sur un projet, à travers GitHub le développeur peut accéder aux fichiers d'un projet que quelqu'un d'autre a créé ou bien créer votre projet et permettre à d'autres utilisateurs d'y accéder.

MongoDB

MongoDB est une base de données NoSQL multiplateforme orientée document qui offre des performances élevées, une haute disponibilité et une évolutivité facile. MongoDB travaille sur le concept de collection et de document.

1.3.4 Principales méthodes

1.3.4.1 RAD : Développent Rapide d'application

RAD est un modèle de processus de développement logiciel qui a un cycle de développement extrêmement court. C'est une version rapide du modèle séquentiel linéaire dans lequel le développement rapide est réalisé en utilisant des composants construction. Si les exigences sont bien comprises et la portée du projet est limitée, le processus RAD permet à une équipe de développement de créer un système fonctionnel dans un délai de 60 à 90 jours. Le modèle RAD comprend les phases suivantes :

- **Modélisation d'affaires (Business Modeling) :**

Il s'agit de la première étape d'un modèle RAD. L'étape utilise généralement des données qui ont été recueillies à partir de différentes sources liées à l'entreprise. Au cours de l'analyse, toutes les informations pertinentes sur l'entreprise sont prises [14]. Après avoir collecté les informations, elles sont combinées pour former une description utile de la façon dont elles peuvent être utilisées pour le succès de l'entreprise.

- **La modélisation des données (Data Modeling) :** Les informations recueillies auprès de Business Modeling est affinée en un ensemble d'objets de données (entités) nécessaires pour soutenir le business. Les attributs (caractère de chaque entité) sont identifiés et la relation entre ces objets de données (entités) est Défini [14].

Au cours de cette étape, les informations qui avaient été recueillies à partir de l'étape précédente sont analysées [14]. L'analyse permettra ensuite de classer les données dans différents groupes qui seront d'une grande importance pour l'entreprise. Les informations de chaque groupe sont ensuite examinées attentivement avant de recevoir leur description exacte.

• **Modélisation de processus (Process Modeling) :** Dans cette étape toutes les données collectées lors de l'étape de modélisation des données sont converties en informations utiles. L'étape permet également d'effectuer des optimisations et des modifications sur les informations recueillies. La phase de modélisation des processus permet également aux entreprises de créer des descriptions pour ajouter, supprimer ou même modifier les objets de données [14].

• **Générateur d'applications (Application Generation) :** C'est la phase de codage de toutes les données qui ont été collectées, des outils automatisés sont utilisés pour faciliter la construction du logiciel [14], Le système qui sera utilisé dans la création du prototype est également construit au cours de cette étape. Les modèles de données créés ici seront ensuite convertis en prototypes lors de la dernière étape.

•**Tester et retourner (Testing and Turn over) :** A ce stade du modèle RAD une grande partie des composants programmé ont déjà été testés. Cela réduit la durée globale des tests, et les chances de rencontrer des problèmes avec le prototype sont peu probable. Mais les nouveaux composants doivent être testés [14].

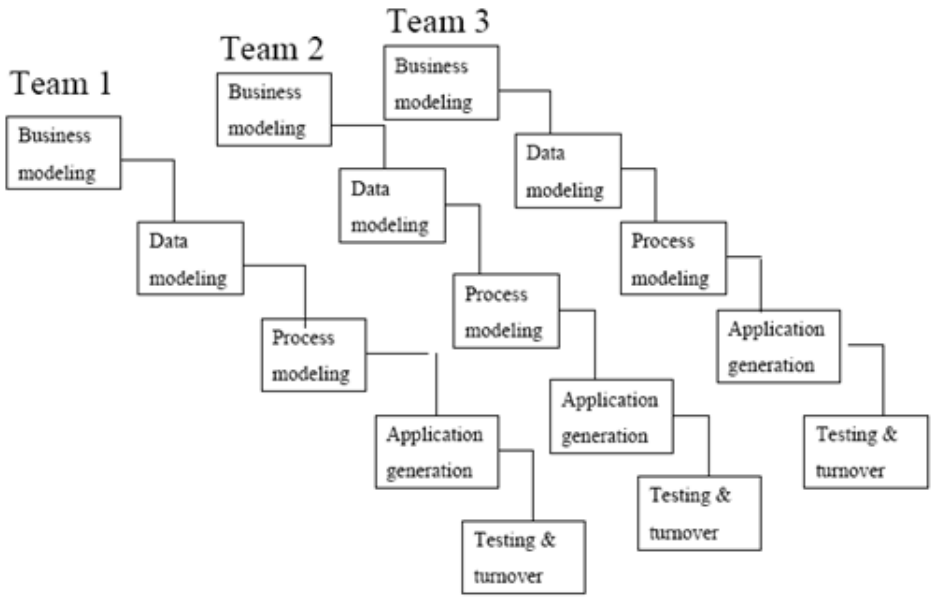


FIGURE 1.5 – Modèle Développent Rapide d'application [14].

1.3.4.2 Développement de logiciels adaptatifs (ASD : Adaptive Software Development)

Le développement de logiciels adaptatifs a été introduit par James A. Highsmith en 2000. Le processus commence par la phase de lancement du projet où les objectifs du cycle de développement et les calendriers sont fixés. Plusieurs composants sont en cours de développement simultané en phase de collaboration. Les composants sont affinés en permanence, c'est pourquoi la planification du cycle de développement fait partie d'un processus itératif. À l'étape finale, il est important de saisir les leçons apprises sur la qualité du résultat du point de vue du client, la qualité du résultent d'un point de vue technique, des pratiques utilisées et fonctionnement de l'équipe de livraison et état d'avancement du projet [6].

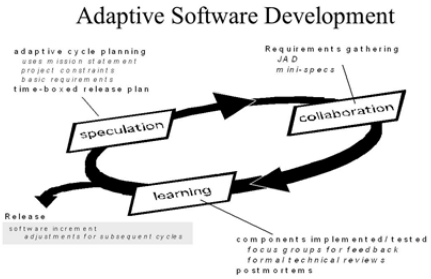


FIGURE 1.6 – Adaptive Software Development cycle de vie [6].

1.3.4.3 eXtreme Programming

Cette méthode a été créée pour répondre aux changements émergeants de l'apprentissage des clients en cours de projet, Conçus pour les petites équipes composées de 2 à 12 personnes, qui travaille dans un espace commun, se concentre sur la participation des clients et la satisfaction simplicité et l'attention continue des développeurs ainsi que les clients, elle vise principalement à produire une qualité supérieure de logiciel.

1. **Les valeurs de XP**
   • La communication : est essentiel pour éviter les erreurs, la conversation face à face est privilégiée.
   • Le courage : Il est nécessaire à tous les intervenants.
   • Le retour d'information (feedback) : permettant une adéquation totale entre l'application finale et la demande du client.
   • La simplicité : éviter tout évolutions futures n'a pas de sens, La meilleure manière de rendre une application extensible est de la garder simple.
2. **Les pratiques**

   II Pratique de planification
   • Séance de planification (the planning game) : Cela implique le développement

stratégie de publication des plans du projet et des réunions entre les développeurs et les clients. Les stratégies pour améliorer la communication entre les parties prenantes sont également axé sur cette pratique de XP. Le système libère des plans et dates des réunions et des revues de projets. Les discussions générales pour connaître l'avancement des travaux sont détenu par XP. Les discussions entre les parties prenantes garantissent que les requêtes sont résolues et que le développeur ainsi que le client ont une bonne compréhension du système et le déroulement du projet [36].

III **Pratiques de conception**

• Conception simple : simplifier tous les éléments pour rendre le code simple à comprendre et facile à modifier.

• Utilisation de métaphores : Il décrit à quoi ressemble le programme. C'est un document décrivant le fonctionnement du système et exprime la vision évolutive du projet qui définirait la portée et le but du système. Les métaphores sont directement dérivées du principe et des normes du projet architecture et exigences Permet de le rendre compréhensible par les non-informaticiens.

• Réfection du code : Amélioration continue de la qualité du code sans modifier le comportement. Il assure la suppression des doublons code et rend le code facile à comprendre.

IV **Pratiques de realization**

• Un représentant du client sur place : pour assurer une meilleure réactivité et comprendre tous les requiers de développement [36].

• Intégration continue : Afin d'avoir toujours une version à jour, Tout le produit est assemblé à chaque fois qu'une tâche est terminée.

• Livraisons fréquentes : Les livraisons doivent être les plus fréquentes possibles, afin d'obtenir un feed-back rapide, tout en offrant des fonctionnalités complètes [36]

• Rythme soutenable : Il faut éviter les heures supplémentaires. Dans tel cas, il faut redéfinir le planning. La cohérence du temps d'investissement est importante pour le calcul Vitesse d'équipe [36].

• Standard de code : Le code est une propriété collective. Les standards facilitent la communication et la rapidité de compréhension. Pour assurer la propriété collective afin que n'importe quel programmeur puisse changer le code dans le système à tout moment, la pratique de la propriété collective est mise en œuvre. XP impose l'utilisation de normes de codage. Les programmeurs doivent suivre une norme de codage commune afin que tout le code semble avoir été écrit par une seule personne [36].

• Programmation en binôme : C'est le concept d'avoir deux les programmeurs travaillent ensemble pour un code de fonction particulier. Un programmeur

écrit le code tandis que l'autre est l'observateur qui révise le code écrit par le programmeur. Les développeurs changent fréquemment de partenaires, ce qui permet d'améliorer la connaissance collective de l'application et d'améliorer la communication au sein de l'équipe [36].

• Appropriation collective du code : C'est une pratique sous XP qui S'assure que tous les membres de l'équipe doivent être familiarisés avec le code de projet. Cette pratique permet à tout programmeur de changer le code dans le système à tout moment [36].

V **Pratiques de tests**

• Tests de recette : À partir de critères définis par le client, on créer des procédures de test, souvent automatisées, qui permettent de vérifier fréquemment le bon fonctionnement de l'application.

• Tests unitaires : Un test unitaire est développé avant d'implémenter une fonctionnalité. Ces tests découlent des recettes de plus haut niveau.

3. **Cycle de vie de XP**

Le cycle de vie itératif suivi dans XP passe par l'analyse, la planification, la conception, la mise en œuvre et le phase de livraison. Toutes ces phases sont exécutées suivant les pratiques de XP. Au départ, des histoires sont créées qui donnent estimation, plan de version, durée de l'itération, etc. Dans la phase d'analyse les stories sont analysées par le développeur pour voir si l'implémentation peut être réalisée ou non. Le résultat de la phase d'analyse est le rapport de faisabilité. Pour former les stories, les séances de brainstorming sont encouragées parmi les parties prenantes. Graphiques de valeur qui aident à répondre à des questions telles que « pourquoi », « comment » sont également suivies. Le graphique de valeur est un outil utilisé pour trouver la valeur et les fonctions exigé. La phase d'analyse est suivie de la phase de planification dont la stratégie est planifiée. Cela se fait selon le la pratique du jeu de planification de XP. Les stories ayant échoué seront analysées dans cette phase et de nouvelles stories seront créées pour surmonter les stories ratées. Après la phase d'analyse, conception et la phase d'implémentation est exécutée, ce qui garantit des tests approfondis. Pair programming est suivie pendant le codage. La dernière étape du cycle de vie est la phase de livraison. Cette la phase comprend les activités telles que l'installation du logiciel, formation du client pour s'habituer aux améliorations ou modifications.

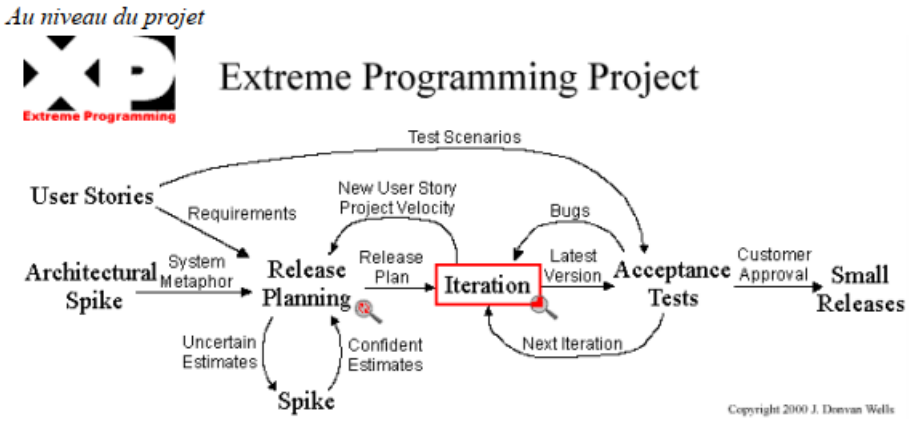


FIGURE 1.7 – Processus XP au niveau du projet [32].

1.3.4.4 Processus Unifié (PU) ou Unified Porcessing (UP)

Le UP ou Unified Process est un processus de conception/développement de logiciel. Son but est d'assurer la production de logiciels de qualité qui rencontrent les besoins de ses utilisateurs finaux dans un horaire et un budget prédictible. Un processus unifié est un processus construit sur UML (Unified Modeling Language).

✓ ***Les caractéristiques de ce processus :***

◇ **Itératif et incrémental :** les itérations se succèdent dans un ordre logique permettant de donner lieu à un incrément et donc d'établir un développement plus optimisé et efficace.

◇ **Piloté par les risques :** Les causes majeures d'échec d'un projet logiciel doivent être écartées en priorité.

◇ **Centré sur architecture :** l'architecture peut être considérée comme l'ensemble de vues du système qui doit être construit. Elle décrit des choix stratégiques qui déterminent en grande partie les qualités du logiciel.

◇ **Conduit par les cas d'utilisation :** Le but principal d'un système informatique est de satisfaire les besoins des utilisateurs. L'utilisateur représente une personne ou une chose dialoguant avec le système en cours de développement. Les cas d'utilisation font apparaître les besoins fonctionnels et leur ensemble constitue le modèle des cas d'utilisation qui décrit les fonctionnalités complètes du système.

✓ ***Les activités de développement du processus unifié :***

◇ **Expression des besoins :** elle permet de définir les différents besoins principaux et fournir une liste de leurs fonctions, recenser les besoins fonctionnels qui conduisent à l'élaboration des modèles de cas d'utilisation, appréhender les besoins non fonctionnels et livrer une liste des exigences [35].

◇ **Analyse et Conception :** ont pour but de comprendre le cahier des charges et d 'écrire les spécifications internes qui décrivent comment implémenter le système et

définir une architecture robuste du système facile à comprendre, construire et faire évoluer [35].

- ◇ **Implémentation :** il est le résultat de la conception pour implémenter le système de composants, c'est-à-dire de code source, de script, de binaire et d'autres éléments de ce type [35].
- ◇ **Test :** la vérification que toutes les exigences ont été implémentées correctement, l'identification et la vérification que toutes les défectuosités découvertes sont corrigées avant le déploiement du logiciel [35].

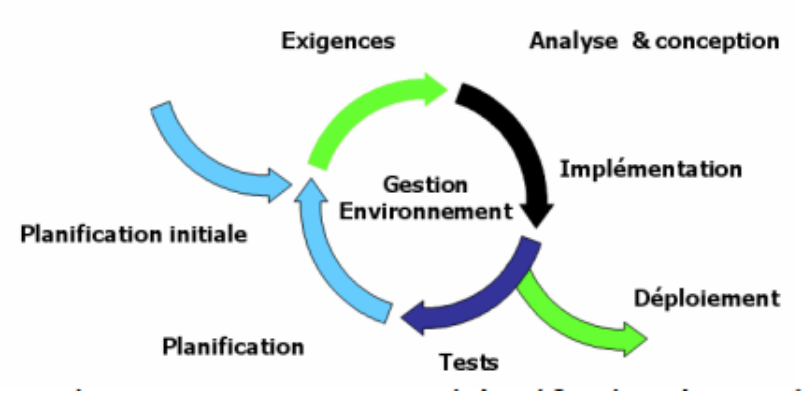


FIGURE 1.8 – Un processus itératif et incrémental. Chaque itération a pour finalité une version exécutable [35].

1.3.4.5 Scrum

La méthode Scrum est la méthode agile la plus populaire. Elle est perçu comme un jeu d'équipe. Elle est définie par son fondateur Ken Schwaber comme un cadre (framework) qui présente les éléments qui feront partie du processus appliqué pour la réalisation d'un produit. Scrum impose des itérations de courtes durées appelées **Sprint** pour le développement du produit. Le schéma ci-dessous montre le déroulement d'un Sprint [7].

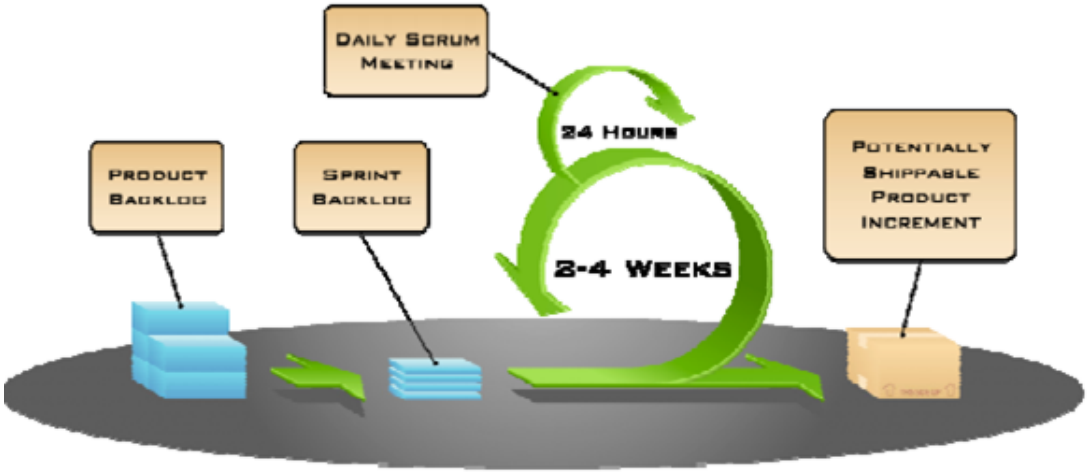


FIGURE 1.9 – Schéma de déroulement d'un Sprint [7].

✓ ***Les caractéristiques :***

Les fonctionnalités à réaliser sont définies et regroupées par ordre de priorité dans le **Backlog** de produit par le **Product Owner**, À chaque itération ou Sprint de durée courte et fixe et définie à l'avance, des fonctionnalités sont développées pour créer une version du produit appelée **Release**. Le contenu de chaque itération est défini par le **Product Owner**.

Au cours des Sprints des rencontres quotidiennes appelées mêlées sont organisées par le **Scrum Master** pour l'application des principes de Scrum et suivre l'avancement par rapport aux engagements afin d'assurer le succès du **Sprint**.

À la fin de chaque Sprint, un produit partiel est livré. Son évaluation et les rétroactions permettent d'ajuster le **Backlog** pour le prochain **Sprint**. À cette étape, l'équipe de projet identifie les anomalies afin d'améliorer le processus lors des prochains Sprints. Cette étape est très importante pour le succès du projet. Elle implique l'intervention du **Product Owner**, du client ou des utilisateurs finaux.

✓ ***Les phases de développement :***

En ingénierie logicielle, quatre phases sont habituellement utilisées pour le développement des produits :

Spécification fonctionnelle (définition des exigences fonctionnelles). **A**rchitecture (conception). **C**odage (et test unitaire) et **T**est (d'intégration et de recette).

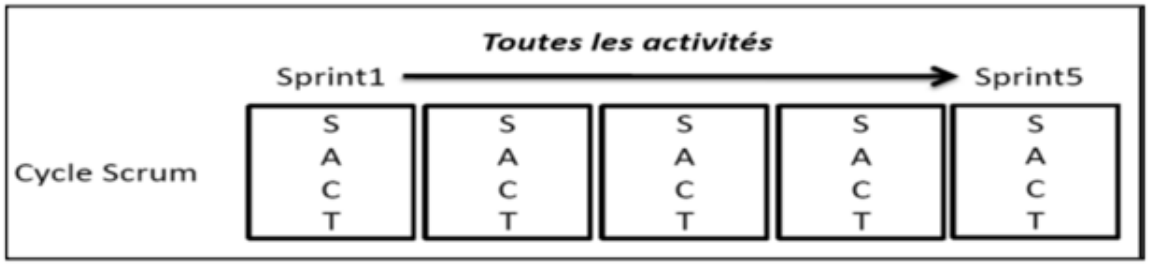


FIGURE 1.10 – Cycle Scrum basé sur 5 Sprints – Itérations [7].

✓ ***Les acteurs :*** Un acteur représente l'abstraction d'un rôle joué par des entités externes (utilisateur, dispositif matériel ou autre système) qui interagissent directement avec le système étudié.
(a) Responsable du produit (Le Product Owner) :
Définit les spécifications fonctionnelles et communique la vision globale du produit à l'équipe Il établit la priorité des fonctionnalités à développer ou à corriger et valide les fonctionnalités développées.
(b) Le Scrum Master :
Ce dernier agit en tant que facilitateur entre le responsable produit et l'équipe. Son rôle principal est d'éliminer tous les obstacles qui peuvent empêcher l'équipe d'atteindre les objectifs fixés pour chaque sprint de travail. Il s'assure que les principes et les valeurs Scrum sont respectés.
(c) L'équipe de développement :
Dans la méthode SCRUM, l'équipe est responsable de la réalisation opérationnelle des tâches. L'équipe est d'ailleurs généralement composée de 6 à 10 personnes mais pouvant aller jusqu'à 200 personnes. C'est toute l'équipe qui est responsable du résultat final de chaque sprint.

1.3.5 Combiner Agile avec le Cloud computing

1.3.5.1 Synthèse

Les méthodes agiles dépendent de la communication interactive entre les développeurs et les clients. Sur site, Il est facile d'établir les communications et interaction, cependant, dans un environnement distribué, il est difficile. Cloud computing aide en fournissant différents moyens de communication entre l'utilisateur et l'équipe logicielle tels que le partage de fichiers, partage d'idées et forums de discussion, wikis, rapports en temps réel et partage de code. Outils de gestion de projet, outils de gestion code et de test sont fournis sous forme de Software as a Service (SaaS). Pour le développement et le déploiement de projets différents IDE et plates-formes sont fournis via Platform as a service (PaaS) dans le cloud computing. Plusieurs études affirment que le cloud computing aide dans les méthodes agiles (rajouter les

refs des etudes). En outre, e-mails, Skype et vidéo conférence, la téléphonie cloud d'Amazon Web Service (AWS) sont également utilisées pour la communication. La figure (1.11) represente un schéma qui montre le développement agile dans l'environnement cloud computing.

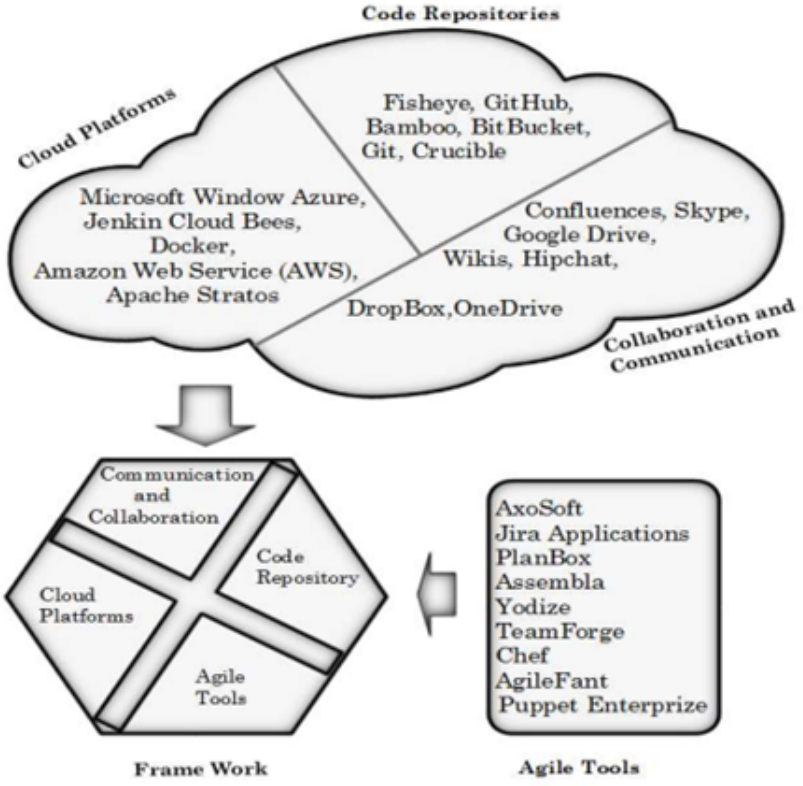


FIGURE 1.11 – Des artefacts pour montrer le développement agile dans le cloud computing [37].

1.3.5.2 Avantages

Le cloud fournit l'environnement pour un développement agile. La combinaison de l'agilité et du cloud permet de créer des logiciels plus rapidement et avec une meilleure qualité. Il y a certains avantages lorsque nous combinons agile et cloud :

Infrastructure : sans cloud, il y a beaucoup des dépenses capitales qui incluent la licence logicielle, le matériel, les outils pour la surveillance ou, les produits de sécurité, etc. Avec l'aide du cloud le besoin d'infrastructure a été supprimé et il n'y a pas besoin d'acheter des logiciels ou du matériel. Les ressources sont allouées en fonction de la demande et des besoins de l'application.

Communication fréquente : en tant qu'organisation géographiquement dispersés il y a un manque de communication entre les membres de l'équipe qui mènent à l'échec d'une application, combinant le cloud avec certains outils de gestion, nous pouvons surmonter ce problème. Fourniture de serveurs : dans un environnement cloud l'équipe de développement

obtient elle-même le serveur requis.

Tests automatisés : Dans le cloud les développeurs peuvent tester le logiciel sur différentes plates-formes utilisant des images virtuelles. Ce qu'augmentent la vitesse de développement logiciel.

Prioriser les tâches dans le cloud : prioriser en permanence les tâches nécessite un suivi approprié et si l'équipe est distribuée donc l'organisation doit installer n'importe quel Outil de gestion de projet Agile dans la plate-forme cloud afin que chaque membre puisse accéder aux données.

1.4 Conclusion

Le premier chapitre vise à donner une description globale de la technologie de Cloud Computing, et les méthodes de développement agiles. Le cloud offre l'environnement et les outils de développement et déploiement des applications. Les modèles de services SaaS, IaaS et PaaS sont les plus utilisés dans le développement dans le cloud. L'utilisation de ces modèles de services pour le développement engendre donc l'adaptation des méthodes agiles usuelles afin de répondre aux besoins des clients et utilisateurs. Le prochain chapitre aborde l'analyse et discussion d'applicabilité des méthodes de développement agiles dans le cloud.

Chapitre 2

Etat de l'art & Proposition

2.1 Introduction

Ce chapitre est consacré à présenter quelques travaux existants dans le domaine qui combine le cloud computing et les méthodes agiles afin de pouvoir fournir un meilleur processus de développement logiciel en décrivant les avantages et les inconvénients de chaque travail. Après nous allons faire une étude comparative entre ces travaux, par la suite nous allons intégrer les concepts et les principes provenant d'une part du développement agile et de l'autre part du cloud computing dans le but de proposer un cadre méthodologique (Framework) de développement agile à base du Cloud.

2.2 Quelques travaux existants

Dans cette partie nous allons traiter quelques travaux existants :

2.2.1 Travail de Gaurav et al [27]

2.2.1.1 Description

Les auteurs citer dessus [27], proposent un processus méthodologique afin de trouver et renforcer la méthodologie agile en donnant à l'utilisateur l'accès à un environnement largement répondu qui est le cloud. Ils proposent un processus SCRUM modifié pour fournir un environnement de test rapide, efficace, à la demande et accessible via le cloud à tout moment. Ce processus, a défini l'importance d'utilisation d'un test as a service (Taas) pour fournir le service de test au client ou à l'organisation sur le cloud à n'importe quel moment. Le processus de développement est divisé en 3 phases (2.1) :

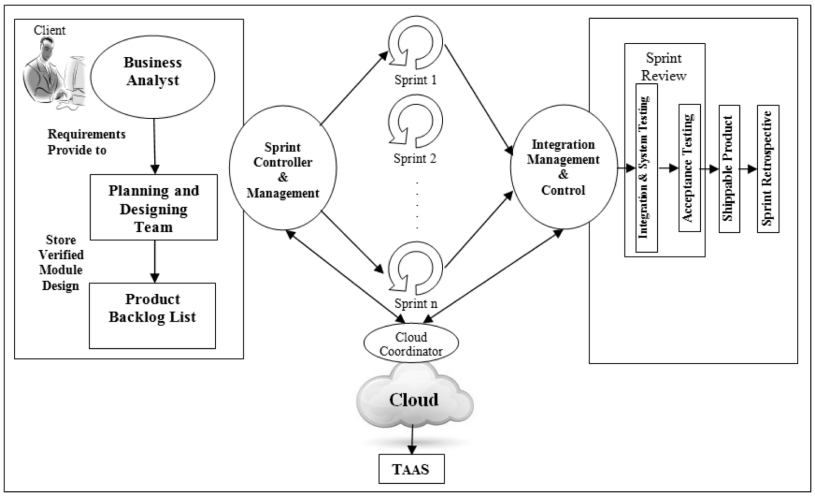


FIGURE 2.1 – Agile SCRUM process Over Cloud.[27].

- ***Pré-itération Phase (planification, conception) :*** l'utilisateur donne ses exigences sous forme de user story à l'organisation, ces besoins sont stockés après dans un espace de stockage en utilisant le stockage as a service (SaaS), la faisabilité de user story est vérifiée, les exigences et la conception du logiciel sont faites et le backlog de produit est créé.
- ***Phase de Développement :*** une fois le product backlog est fait (terminé, réalisé), l'ensemble du projet est divisé en petits sprints. Ensuite, la phase de développement commence, au cours de cette phase les développeurs développent de code. Une machine virtuelle est créée pour tester le code en utilisant TaaS, puis effectuer des modifications sur le code si nécessaire. L'intégration continue du code se fait par le biais d'un développement automatisé.
- ***Post-itération Phase :*** une fois tous les sprints terminés, tous les modules sont intégrés et testés à l'aide de Test-as-a-Service, le produit final est envoyé au client pour des tests d'acceptation.

2.2.1.2 Discussion

Suite à l'étude de ce travail, l'équipe a planifié un processus d'implémentation SCRUM sur le cloud, afin de satisfaire au maximum les besoins des utilisateurs ,en intégrant des tests à l'aide des test en tant que service. les membres de l'équipe étaient bien qualifiés pour réaliser cet combinaison du cloud avec la méthode agile SCRUM, mais leur travail n'a pas été comparé avec une autre étude de même objectif .

✓ ***Avantages :***

◇ L'utilisation Test-as-a-Service (TaaS) permet d'automatiser le processus de test (cout

et temps de test) et à aider les entreprises à être plus rapides dans la livraison des applications à leurs clients.

◇ Collaboration avec les clients.

◇ L'utilisation d'un cloud privé permet d'assuré la qualité de processus de sécurité des applications déployées en empêchant tout accès non autorisé.

✓ ***Inconvénients :***

◇ La non précision des identifications des équipes et leurs rôles.

◇ La non spécification des fonctionnalités du backlog.

Le temps et le cout sont parmi les facteurs qui motivent toute entreprise qui accède aux outils et aux services via le cloud, qui fournit une plateforme pour le développement et faire des tests automatisés afin de réduire le cout et le temps et d'améliorer la qualité du produit délivré à l'utilisateur final (client). A l'aide de la mise en œuvre de TaaS dans ce processus SCRUM l'environnement fourni la possibilité de faire des tests rapides, efficaces et à la demande qui sont accessibles via le cloud à tout moment.

2.2.2 Travail de Chung Yung et Yu-Tang Lin [39]

2.2.2.1 Description

Le travail présenté par Chung Yung et Yu-Tang Lin [39], décrit la construction d'un outil de gestion de projet appelé TOAST qui est un outil de gestion de projet de développement agile d'un logiciel dans des environnements de cloud. Ce dernier est conçu et implémenté en se basant sur le modèle MOAT avec une architecture client/serveur, dont le client TOAST est déployé en tant qu'application Android et le serveur TOAST est conçu pour fonctionner dans un environnement cloud. MOAT est un modèle à deux couches de gestion de projets utilisant une méthodologie agile. Dans MOAT, les principales opérations de développement logiciel agile sont classées en deux couches ; à savoir, la couche projet , MOAT modélisé les 12 opérations principales suivantes dans le développement de logiciels agiles (p1 : Trier les exigences et diviser en sprints, p2 : Répartir en tâches , p3 : développer des taches , p4 : intégrer des taches p5 : Test de vitesse, p6 : signaler des bugs de sprint, p7 : commencer le prochain sprint, p8 :intégrer des sprints, p9 : test de projet, p10 :signaler les bogues du projet, p11 : corriger les bugs du projet, p12 : projet complet) et la couche tâche MOAT modélise les 03 opérations suivantes dans le Développent de logiciels agiles (Q1 : Ramasser la tache, Q2 : Retirer la tache, Q3 : Terminer la tache) . Concernant TOAST, il implémente toutes les opérations MOAT à l'aide d'une architecture client/serveur, dont le client TOAST est déployé en tant qu'application Android et le serveur TOAST est conçu pour fournir des services dans un environnement cloud. TOAST est divisé en 3 sous-systèmes (2.2) dans :

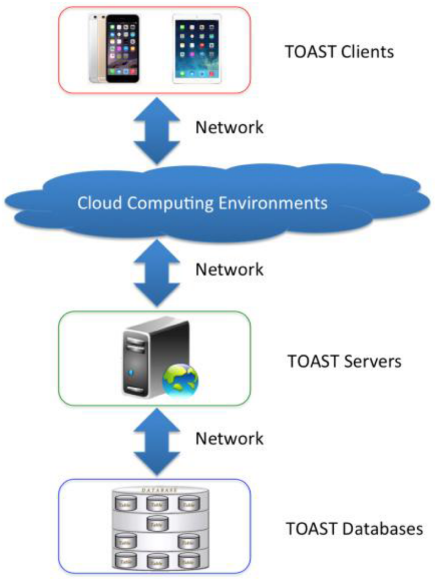


FIGURE 2.2 – The overall architecture of TOAST.[39].

- ***Le sous-système client TOAST :*** a deux types d'interfaces implémentées. La première est une version de page Web consultée avec des navigateurs Web, le second est une application Android autonome packagée.
- ***Le sous-système du serveur TOAST :*** comprend un ensemble de programmes codés en Java et Java Server Pages (JSP).
- ***Le sous-système de base de données TOAST :*** se compose de quelques tables dans une base de données SQL qui sont chargées de manipuler les données utilisateur, les données de projet et les données d'exploitation.

Pour la validation, ils ont appliqué MOAT à un projet pratique CSCI, qui est un système de vente coopératif d'assurance automobile en ligne, le système logiciel CSCI est développé en utilisant une méthodologie agile avec deux cycles de sprint A et B.

2.2.2.2 Discussion

Cette étude de Yung et Lin, n'a pas était comparé aux autres études sur la combinaison du cloud avec la méthode agile SCRUM, TOAST est un outil de gestion de projet logiciel de méthodologie de développement agile, implémente toutes les opérations de développement agile dans MOAT et est spécialement conçu pour gérer des projets logiciels agiles dans des environnements de cloud computing. Le système TOAST est implémenté en utilisant une architecture client/serveur, aussi à partir de la description précédente de cet article on déduit :

✓ ***Avantages :***

◇ Collaboration avec les clients.

- ◇ La transparence : c'est à dire les aspects importants du processus MOAT sont visibles à Tous.
- ◇ La faisabilité de l'application TOAST est assuré car à la fin ils ont fait un bilan sur les résultats produits et ils l'ont appliqué pour réaliser une application de vente coopératif d'assurance automobile en ligne.

✓ ***Inconvénients :***

- ◇ La Non répartition des rôles.
- ◇ Aucun communiqué sur l'environnement du cloud utilisé (Les Types de service, Les modèles déploiement du cloud)
- ◇ Non spécification de processus de sécurité pour la gestion de l'identité des utilisateurs, d'authentification et de contrôle d'accès pour l'outil du cloud.

Un outil de gestion de projet de développement agile d'un logiciel dans des environnements de cloud, appelé TOAST, implémente toutes les opérations de développement agile dans MOAT est conçu. Le système TOAST est développé en utilisant une architecture client/serveur. Le travail démontre que la gestion des progrès dans un projet de développement d'un logiciel en suivant un processus agile est simple et facile. Leur orientation future est de coopérer avec l'industrie en appliquant l'outil TOAST pour gérer des projets de développement de logiciels agiles au niveau de l'industrie dans des environnements de cloud computing.

2.2.3 Travail de YOUNAS et al [38]

2.2.3.1 Description

Dans ce travail [38], Les auteurs proposent un framework générique avec la conjonction du Développement Agile et du Cloud Computing (ADCC)

L'environnement ADCC est évalué en développant une application pour un system de gestion d'hôpital avec la méthode SCRUM. Quatre équipes participent dans les 4 différents scenarios dans un environnement local et distribué avec un développement agile simple ou ADCC. Chaque équipe se compose de 4 personnes et sont tous capable de faire le design, codage et le test. L'application est complétée en 6 versions. L'évaluation est mesurée en terme de nombre de jours pour compléter les trois phases de développement, i) l'élicitation des exigences, ii) la planification, la conception et le codage et iii) les tests et le déploiement, dans défirent scénarios et environnements. Dans l'environnement ADCC toutes les activités sont effectuées en parallèle via cloud virtuel machines. Les artéfacts (outils) utilisés pour évaluer l'environnement ADCC sont décrit dans (2.3) :

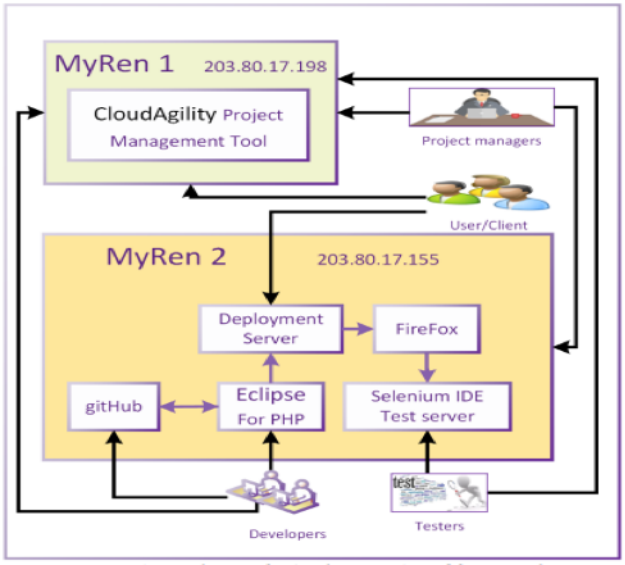


FIGURE 2.3 – Configuration expérimentale pour la mise en œuvre du Framework.[38].

CloudAgility est configuré dans MyRen 1 (Malaysia Research and education network) Virtuel machine. Selenium IDE pour les tests est configuré dans Firefox. Le test server et l'environnement de développement sont configuré dans des machines virtuelles séparés, eclipse IDE pour le développement est configuré avec GitHub pour la gestion de code dans un environnement distribué, Skype et Google Drive sont utilisé pour la communication et le partage de code et le design.

2.2.3.2 Discussion

Selon les résultats de cas d'étude

✓ ***Avantages :***

Selon les résultats de cas d'étude

- ◇ L'utilisation du Framework ADCC produit un environnement structuré pour un développement agile en fournissant une infrastructure de développement, des serveurs de test à la demande et un déploiement continue.
- ◇ Les membres de chaque équipe sont bien qualifiés à mener cette étude.
- ◇ Le Framework ADCC réduit le temps de communication dans le développement agile grâce à l'utilisation du cloud.
- ◇ Le Framework ADCC décrit les outils pour supporter les activités de développement agile dans le contexte du cloud computing et leur compatibilité.

✓ ***Inconvénients :***

- ◇ Aucun control de communication entre les membres d'équipe de la même université dans le cas d'un environnement distribué.

◇ Il y a une chance que le niveau de compétence diffère entre les équipes.
◇ Cette étude n'a pas été comparé avec d'autres études.

Un framework générique pour un développement agile dans un environnement de cloud computing (ADCC) est proposé et une étude de cas est menée pour évaluer le framework avec le développement d'une application dédiée à la gestion d'un hôpital (HMS). Le framework ADCC fournit un environnement interconnecté pour tous les membres de l'équipe de développement et tous les systèmes se comportent comme un seul système. Cependant, comme un aspect futur, les auteurs visent à appliquer le framework sur d'autres méthodes de génie logiciel en sélectionnant d'autres outils et environnements.

2.2.4 Travail de E.Toews et al [31]

2.2.4.1 Description

Le but de project Cloud-Enabled Space Weather Platform (CESWP) est d'apporter la puissance et la flexibilité du cloud computing aux physiciens de la météorologie spatiale et faciliter la collaboration avec d'autres scientifiques, développer des modèles de météo spatiale, exécuter des simulations, produire des visualisations et permettre la provenance. Pour fournir le cloud computing et le stockage, infrastructure as a service. Le projet a construit un cloud distribué à l'échelle internationale basé sur Eucalyptus. Pour fournir une interface graphique utilisateur avec laquelle les physiciens peuvent interagir une application web été développé, le langage de programmation Groovy et Grails Web framework, la méthodologie agile Scrum et l'environnement de développement intégré Intellij IDEA qui prend en charge à la fois Groovy et Grails ont été sélectionné pour construire le logiciel. Les scientifiques peuvent se connecter à leurs machines virtuelles via NoMachine qui est un logiciel d'accès à distance avec prise en charge d'une interface graphique utilisateur.

Le CESWP lui-même est composé du CESWP cloud, il fournir IaaS que les physiciens utiliseront pour mener leur science. CESWP toolkit qui est inclus sur chaque image VM dans le cloud et composé de scripts, both bash et Python, Ces scripts remplissent des fonctions telles que l'exécution de simulations en parallèle, regroupement d'images de VM. Et l'application CESWP.

Les principaux éléments constitutifs de CESWP cloud ont identifié : Eucalyptus, NoMachine et Grails. Ces blocs de construction étaient évalués en fonction de leur adéquation au projet Cloud Enabled Space Weather Platform. Et ont déterminé que L'eucalyptus ne convient pas au CESWP mais que NoMachine et Les Grails conviennent au CESWP et sont probablement adaptés au projets cloud en général

2.2.4.2 Discussion

✓ ***Avantages :***

◇ CESWP cloud baisse les barrières à la spécification et à l'acquisition de matériel, la maintenance de ce matériel.

◇ CESWP Il permet de sécuriser, un accès global à leurs machines virtuelles à partir de divers appareils, Cet accès facilite le partage des données et la collaboration entre chercheurs au niveau international.

✓ ***Inconvénients :***

◇ L'eucalyptus ne convient pas au CESWP.

◇ Le cloud CESWP limite la sélection de systèmes d'exploitation vers Ubuntu et CentOS sur la machine virtuelle.

2.2.5 Étude comparative des travaux existants

Tous les articles cités ont comme objectif principale, traité les manières de combiné le cloud computing et les méthodes agiles afin de fournir un meilleur processus de développement logiciel qui facilite la collaboration et le partage des ressources entre les membres de l'équipe de développement. Un résumé d'une étude comparative des des travaux traités est présenté dans le tableau ci-dessous (2.1).

Critères	**Méthode agile**	Service Cloud	Outils de développement	Partage de données	Réponse aux changements	Méthode d'évaluation
Travail de YOUNAS et al [38]	SCRUM	IaaS	Eclips, Skype, Google Drive, selenuim IDE, CloudAgility	Oui	Oui	Développement d'un système de gestion d'un hopital
Travail de Gaurav et al [27]	SCRUM	Taas Iaas	Machine virtuelle SCM, mingle cockpit	Non	Non	
Travail de C.Yung and Y. Lin [39]	MOAT	Paas	TOAST		Oui	système de vente coopératif d'assurance automobile (CSCI)
Travail de E.Toews et al [31]	SCRUM	IaaS	Intellij IDEA Eucalyptus, NoMachine, Grails	Oui	Oui	Cloud-Enabled Space Weather Platform (CESWP)

Tableau 2.1 – Tableau comparatif des travaux.

2.3 Description globale du Framework Agile-Cloud proposé

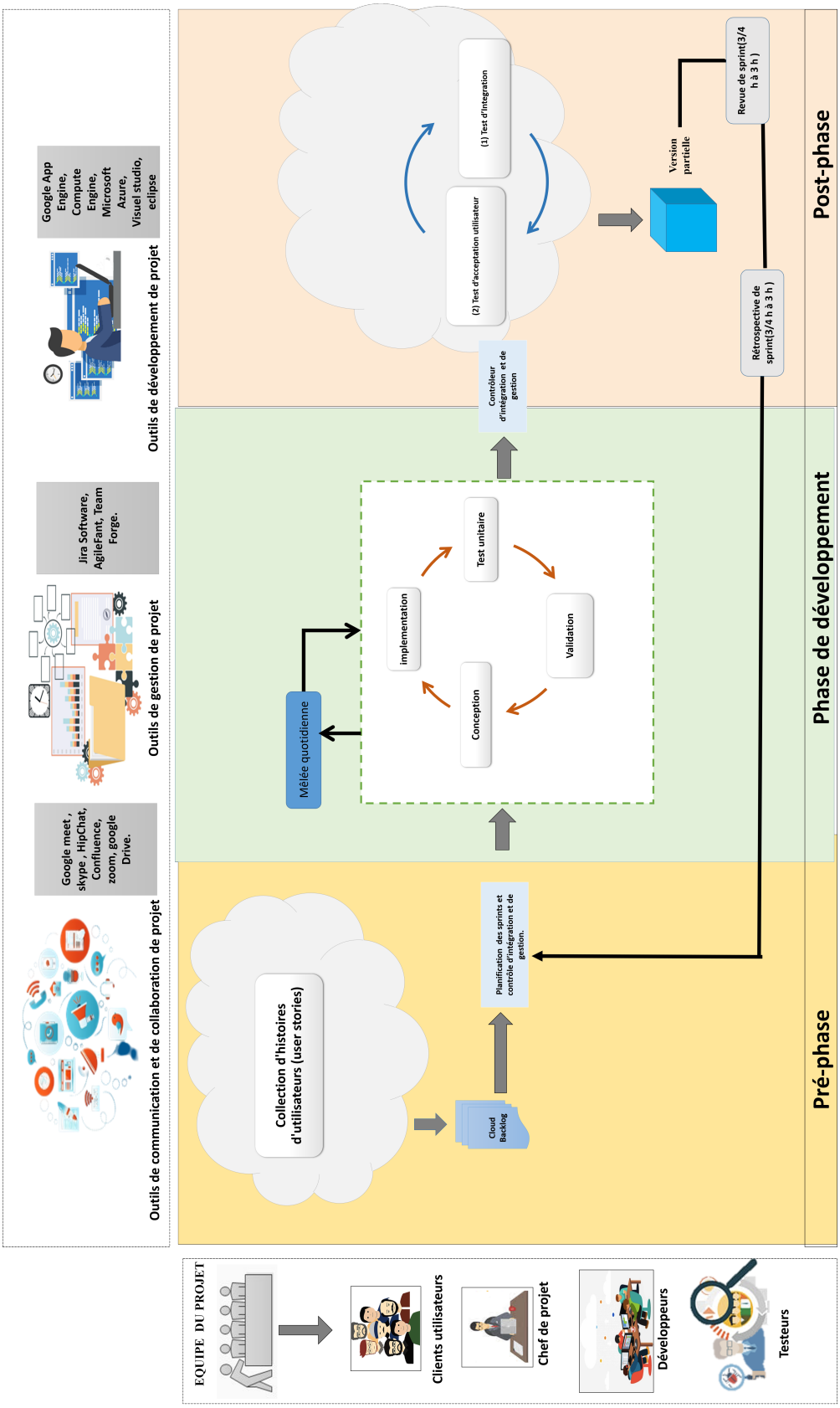


FIGURE 2.4 – Agile-Cloud Framework.

2.3.1 Présentation des acteurs du Framework

Le succès de la mise en œuvre d'un Framework repose sur l'implication d'une équipe projet en interne, rassemblant des acteurs de différentes spécialités. Les types d'acteurs principalement concernés sont successivement décrits.

(a) Responsable du produit (Le Product Owner) :
Le responsable du produit (Le Product Owner) [30] est un rôle défini par Scrum. Il s'agit du représentant des utilisateurs finaux. Le responsable du produit a pour rôle de définir les spécifications fonctionnelles et communique la vision globale du produit à l'équipe. De plus, il établit la priorité des fonctionnalités à développer ou à corriger et valide les fonctionnalités développées.

(b) Le Scrum Master : Le Scrum Master agit en tant que facilitateur entre le responsable produit et l'équipe. Son rôle principal est d'éliminer tous les obstacles qui peuvent empêcher l'équipe d'atteindre les objectifs fixés pour chaque sprint de travail. Il s'assure que les principes et les valeurs Scrum sont respectés.

(c) Contrôleur de planification et de gestion de sprint : Il réalise certaines activités :

- ◇ Planification des besoins du product backlog.
- ◇ Vérification de la faisabilité des besoins et exigences collectées.
- ◇ Classifier les données du sprint backlog et priorisation des tâches à réaliser.
- ◇ Attribuez les machines virtuelles selon les besoins.
- ◇ Vérifiez la progression quotidienne de chaque Sprint.

(d) Contrôleur d'intégration et de gestion : Il réalise les taches suivantes :

- ◇ Intégrer les modules nécessaires aux modules dépendants en effectuant des tests d'intégration.
- ◇ Livrer les résultats des Sprints à l'utilisateur pour les tests d'acceptation.

(e) L'équipe de développement :

- **Les concepteurs :** Les concepteurs ont pour objectif de traduire les besoins de l'utilisateur en spécifications fonctionnelles prêtes pour l'implémentation. Pour réaliser cette traduction, ils utilisent des modèles (modèle de cas d'utilisation, diagramme de séquence, etc.) et des formalismes de représentation.
- **Les développeurs :** C'est l'équipe responsable de la réalisation opérationnelle des tâches. Ils ont pour objectif de traduire les modèles et les formalismes conçus par les concepteurs pour réaliser une application mobile. Ils concrétisent et implémentent les noyaux fonctionnels des differentes parties et modules de l'applicaion. Dans la méthode SCRUM, l'équipe est généralement composée de 6 à 10 personnes mais pouvant aller jusqu'à 200 personnes. C'est toute l'équipe qui est responsable du résultat final de chaque sprint.

(f) Testeur : En agile, les testeurs font partie de l'équipe et interagissent directement

avec les développeurs et sont présents à chaque phase du cycle de vie du développement du produit.

Dans ce qui suit, nous présentons une description détaillée des trois phases du framework proposé ainsi que les étapes impliquées dans chaque phase.

2.3.2 Description détaillée des différentes phases du Framework

2.3.2.1 Description de la Pré-phase

(a) Etape 1 : Collecte et création des histoires d'utilisateurs (Besoins des utilisateurs)

Cette étape consiste à identifier et collecter les besoins d'utilisateur en créant des user stories [11](en français : les histoires d'utilisateur) qui seront traduits en termes de besoins relativement à des services. En fait, les histoires d'utilisateurs sont une pratique agile, provenant particulièrement de la méthode eXtreme Programming (XP), permettant de capturer les besoins de l'utilisateur. Ils sont utilisés comme un moyen considéré comme très efficace pour exprimer et comprendre les besoins des utilisateurs. Une histoire de l'utilisateur est une description centrée sur les objectifs à réaliser.

Après la collecte des histoires d'utilisateurs, le Product Backlog est géré par la plateforme Jira, qui permet de planifier, gérer et suivre l'état d'avance des projets de développement agile. Le Product Backlog est stocké dans un espace de stockage utilisant le stockage en tant que service, du cloud privé de l'organisation. On appelle (Cloud Product Backlog).

Il existe de nombreux modèles différents pour créer une histoire d'utilisateur, mais les propriétaires de produits doivent adopter un modèle qui inclut les dimensions « qui », « quoi » et « pourquoi », Ces trois éléments clés permettent d'exprimer clairement l'histoire d'utilisateur. D'autres informations doivent être mentionnées : la portée, précondition, et critère d'acceptation. La Figure (2.5) représente un modèle d'une user story (histoire d'utilisateur).

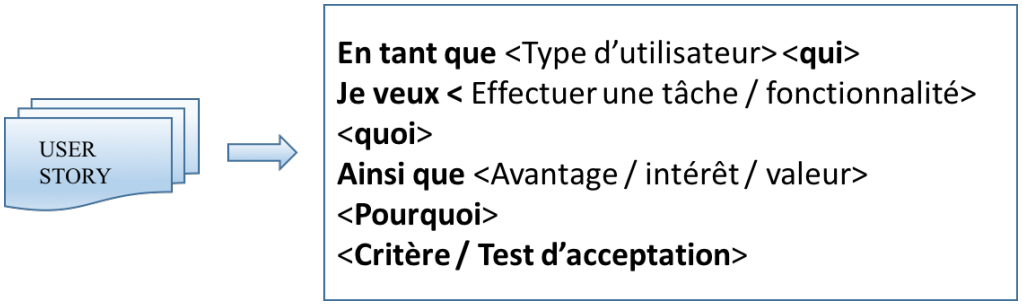


FIGURE 2.5 – Description d'une user story (modèle d'une user story).

(b) Etape 2 : Planification des Sprints et Contrôle d'intégration et de gestion

En utilisant la pratique de la méthode agile Scrum [34] , la planification des itérations (sprints) est réalisée par le propriétaire du produit et l'équipe de développement dans un but de partager la même vision du projet. En fait, l'équipe de développement sélectionne et ordonne les besoins des utilisateurs en fonction de leurs priorités représentés par des éléments dans le product backlog. Ensuite, l'équipe de développement estime le temps nécessaire pour leur réalisation et détermine quels sont les besoins qui feront partie de l'itération (sprint) actuelle. Par la suite, l'équipe de développement décompose les éléments choisis en tâches formant le sprint backlog.

2.3.2.2 Description de la Phase de développement

Durant les itérations de développements, l'équipe de développement se focalise sur les besoins définis dans le sprint backlog. En effet, cette équipe se réunit chaque jour lors de la réunion quotidienne (Daily meeting) afin d'évoquer le travail réalisé la veille, celui de la journée et relever les obstacles rencontrés. Toutefois, l'équipe de développement peut être dans un environnement géographiquement distribué. Donc, pour réaliser cette réunion, un ensemble d'outils de collaboration et communication peut être utilisés tels que Skype, Confluences, Discussion forum, wikis, real-time reports, Google Meet, etc. Durant ces réunions, chaque membre de l'équipe actualise les tâches qu'il lui reste à faire de manière à actualiser le graphique d'avancement de l'itération. Cette réunion est en principe limitée à 15 minutes (comme préconisé dans la méthode Scrum) ; d'autres acteurs du projet peuvent être présents mais seuls les membres de l'équipe de développement ont principalement la parole. L'itération englobe les quatre activités de conception, implémentation, test unitaire et validation qui sont présentées ci-dessous.

(a) Etape 1 : Conception

Après avoir identifié et sélectionné les besoins qui rentrent dans l'itération, il est nécessaire de spécifier comment ces derniers vont être construits. En effet, cette étape consiste à transformer les exigences en caractéristiques spécifiques.

(b) Etape 2 : Implémentation

Dans cette étape, les développeurs codent les parties fonctionnelles et IHM des besoins identifiés et conçus auparavant.

(c) Etape 3 : Tests Unitaires

Cette étape appelée aussi test de composant, il s'agit de vérifier les unités du code s'il ne contient pas de bugs. Ces tests doivent être automatisés rapidement pour permettre de valider la non régression du fonctionnement du composant lors des multiples livraisons des différentes versions du projet. Malgré des tests consécutifs sur une ma-

chine, l'équipe de développement peut exécuter les tests unitaires en parallèle via des machines cloud.

(d) Etape 4 : Validation

Une fois les tests unitaires sont effectués, l'équipe de développement doit vérifier si les résultats de la partie du codage sont conformes à chaque sprint, c'est-à-dire, si ces derniers réalisent toutes les fonctionnalités pour lesquels elles ont été conçues.
Une fois la conformité fonctionnel est assuré, un bilan ou un récapitulatif est réalisé à la fin pour qu'ils soient stockées dans le contrôle d'intégration et de gestion.

2.3.2.3 Description de la post-phase

Dans cette phase un contrôle d'intégration et de gestion est effectuer, il consiste à stocker les résultats du Sprint et à intégrer les modules nécessaires aux modules dépendants, à effectuer des tests d'intégration de ces modules particuliers et les tester. Le contrôleur d'intégration fournit les résultats des Sprints à l'utilisateur pour effectuer des tests d'acceptation.

(a) Etape 1 : Test d'intégration

Cette étape est effectuée après que les développeurs ont validé leurs développements ou leurs corrections, puis fusionnant les modifications, l'intégration pour garantir que toutes les pièces développées indépendamment fonctionnent ensemble. L'intégration continue est une combinaison de tests unitaires et d'intégration. Il existe deux types de tests d'intégration :

- ◇ ***Tests d'intégration composants :*** permettent de vérifier si plusieurs unités de code fonctionnent bien ensemble, dans un environnement de test assez proche du test unitaire, c'est-à-dire de manière isolée, sans lien avec des composants extérieurs et ne permettant pas le démarrage d'une vraie application.
- ◇ ***Tests d'intégration système :*** permettent de vérifier le fonctionnement de plusieurs unités de code au sein d'une configuration d'application, avec éventuellement des liens avec des composants extérieurs comme une base de données, des fichiers, ou des API en réseau.

(b) Etape 2 : Test d'acceptation utilisateur

Les développeurs développent des logiciels selon les spécifications fournies et selon leurs propres compréhensions qui peuvent ne pas être conforme aux besoins d'utilisateurs. Le test d'acceptation utilisateur permet de vérifier si le logiciel fonctionne selon les exigences utilisateurs ou non.
L'objectif des tests d'acceptation par l'utilisateur est d'assister le logiciel par rapport aux exigences demandées. Ce test est généralement la dernière étape avant que le produit ne soit publié dans l'environnement réel pour recevoir des données directes ou avant que la livraison du produit soit acceptée.

2.3.2.4 Revue d'itération (Sprint Review)

À l'issue de chaque itération (Sprint), une revue d'itération est réalisée par l'équipe de développement avec le propriétaire du produit et l'ensemble des utilisateurs finaux afin de leur présenter les services réalisés au cours de l'itération sous forme d'une démonstration. Cette réunion permet de déterminer l'acceptation ou le rejet (partiel ou dans le pire des cas total) de l'ensemble de services produits. Des feedbacks (des retours ou des critiques) seront notés en conséquence. Généralement, une réunion de ce type dure 2 heures au maximum.

2.3.2.5 Rétrospective d'itération (Sprint Retroscpective)

Après la revue d'itération, une rétrospective d'itération est menée par l'équipe de développement ainsi que le propriétaire du produit afin d'identifier les adaptations susceptibles d'augmenter sa productivité. Les services qui fonctionnent, ceux qui ne fonctionnent pas ainsi que les améliorations à apporter, sont identifiés dans cette réunion. La durée maximale de cette réunion est d'environ une heure et demi, en se référant à la méthode Scrum.

Le tableau ci dessous (2.2) résume les différents outils de cloud qu'on peut utiliser dans un processus de développement agile :

	Nom d'outil	**Étape de Framework correspondante**
Communication et collaboration	Confluence, discussion Forums, wikis, real-time reports, google meet, skype, google drive.	Toutes les étapes du Framework.
Gestion de projet agile	AgileFant, CloudForge, Jira, CloudAgility.	Planification
Stockage et partage de code source	BitBucket, GitHub, CodeSpace, Google-Code, SourceForge, Unfuddle.	Implémentation.
Plateformes de déploiement de logiciel	Puppet, Chef, Jcloud, Whirr.	Test d'acceptation utilisateur.
Tests	Test des infrastructures réseau : TestInfra. Test unitaires : Junit, Appium. Environnements de test : Jenkins, Xunit, Junit, Genymotion Cloud Saas.	Test unitaire. Test d'intégration.
Autres outils cloud	Google App Engine, Microsoft azure, Amazon web service.	Implémentation.

Tableau 2.2 – Tableau collecte l'ensemble d'outils cloud agile.

2.4 Conclusion

Dans ce chapitre, nous avons d'abord dressé un état de l'art sur quelques travaux existants dans le domaine qui combine le cloud computing et les méthodes agiles.

Après, nous avons effectué une étude comparative entre ces travaux. Par la suite, nous avons présenté l'ensemble des phases et des étapes de notre cadre méthodologique (framework) que nous avons proposé.

Notre framework s'appuie essentiellement sur un ensemble de méthodes et modèles issus du domaine de l'ingénierie logicielle plus précisément sur les méthodes agiles d'une part, et de la technologie du cloud computing d'autre part. Il se compose de trois grandes phases (1) pre-phase, (2) Phase de développement et (3) post phase avec leurs étapes.

Dans le chapitre suivant, nous nous focalisons sur l'application du framework proposé à un cas d'étude concret pour la gestion des patients. L'objectif de ce cas d'application est d'évaluer et de valider le framework proposé dans le cadre de ce travail.

Chapitre 3

Validation du framework proposé par une étude de cas : Gestion de patients

3.1 Introduction

Ce chapitre est consacré à la validation et l'implémentation de la solution proposée dans le chapitre précédent afin de réaliser notre application mobile sur la gestion des patients.

D'abord, nous décrirons les différents outils de développements utilisés pour la réalisation de notre application mobile. Ces derniers se basent essentiellement sur des technologies gratuites et open source afin d'éviter les différents problèmes liés à la licence, contrat et coût. Ensuite, nous présentons la mise en application du framework proposé à travers les phases déjà énoncées (cf. chapitre 2) sur un cas d'étude réel relatif à la gestion des patients. Enfin, nous terminons ce chapitre par une conclusion.

3.2 Outils de développement

Afin de réaliser et d'implémenter notre solution nous avons eu recours à un ensemble d'outils et de langages de développement.

Les détails des outils sélectionné, et leurs configurations, sont décrits dans la figure (3.1), Jira plateforme est utilisé pour la gestion de projet agile, Draw.io pour la modélisation, Android studio est utilisé comme environnement de développement permettant de créer des applications android, ce dernier est configuré avec Git et GitHub pour le partage de code. Pour la communication Google meet est utilisé et pour le partage de conception et livraison des version d'application google Drive est utilisé.

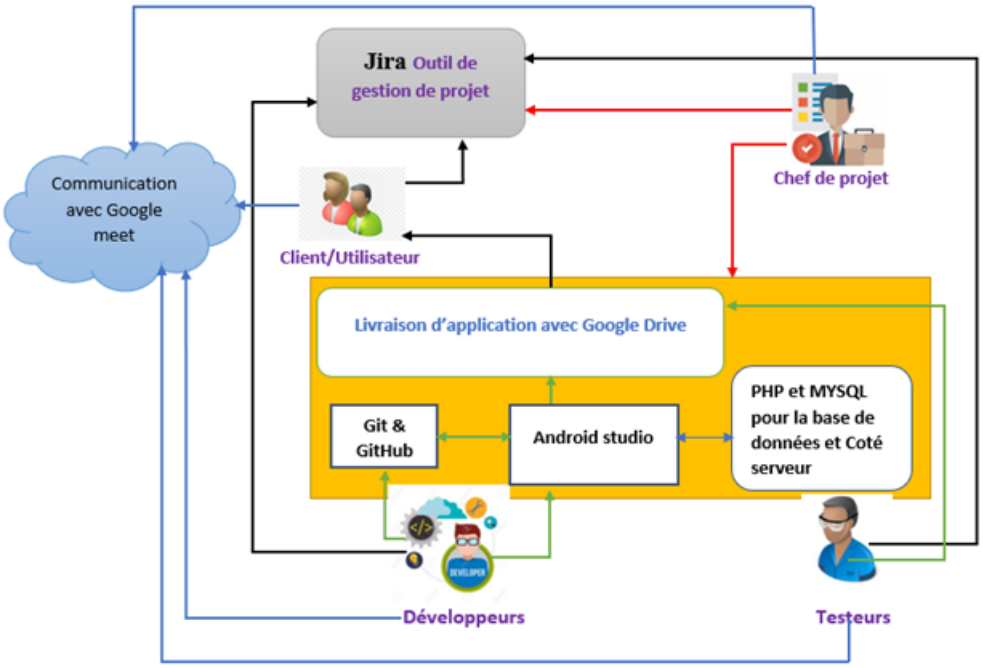


FIGURE 3.1 – Configuration expérimentale pour la mise en œuvre du Framework.

3.2.1 Draw.io

[25] Est une application de création de diagrammes et d'organigrammes sous licence Apache disponible sous Windows , MacOs, Linux, sous forme d'application web intégrée à des services cloud tels que NextCloud ou Google Drive. Une fois les diagrammes et les organigrammes sont créés, en peut les enregistrer au format XML sur un compte Dropbox ou Google Drive puis les exporter sous différents formats (PNG, JPG, GIF, SVG, html, intégration en pages web).

3.2.2 JIRA

[4] est un système de gestion de projets principalement dédié à la gestion de projet de développement d'applications en mode agile, développer par une entreprise australienne appelé atlassian. Atlassian propose Jira gratuitement pour un essai de 14 jours. A partir du 15 ème jours d'utilisation, en hébergement cloud, il coûte entre 7$ à 10$ en un mois par utilisateur jusqu'à 10 utilisateurs comme capacité maximale .

Jira est un logiciel qui convient parfaitement aux chefs de projet qui adoptent une méthode agile. Cet outil permet de planifier des tâches avec une grande flexibilité (en mode Scrum), d'obtenir des estimations précises d'état d'avancement ou encore de prioriser les user stories en fonction des besoins. Jira Software est disponible en deux versions : en mode cloud et en version sur serveur (ou auto-hébergement). Baptisée Jira Cloud, sa version cloud

s'accompagne par ailleurs d'une application mobile pour permettre aux utilisateurs de faire avancer leurs tâches, quel que soit l'endroit où ils se trouvent. Avec Jira, il est possible de :
-Personnaliser des tableaux Scrum.
-Consulter des rapports temps réel prêts à l'emploi.
-Bénéficier d'une digital workplace pour le travail d'équipe.
-Créer des templates personnalisées en fonction des besoins.

3.2.3 Android studio

Est un environnement de développement permet de développer des applications mobiles Android. Il est gratuit et compatibles avec certains systèmes d'exploitation tels que Apple, Windows et Linux. il a été conçu pour fournir un environnement de développement et une alternative à Eclipse qui est l'IDE le plus utilisé.

Android Studio offre aux développeurs une boîte à outils bien fournie pour créer des applications Android ou d'autres projets. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il peut être téléchargé sous les systèmes d'exploitation Windows, macOS, Chrome OS et Linux.

Principalement il permet de d'éditer les fichiers Java et les fichiers de configuration XML d'une application Android [1].

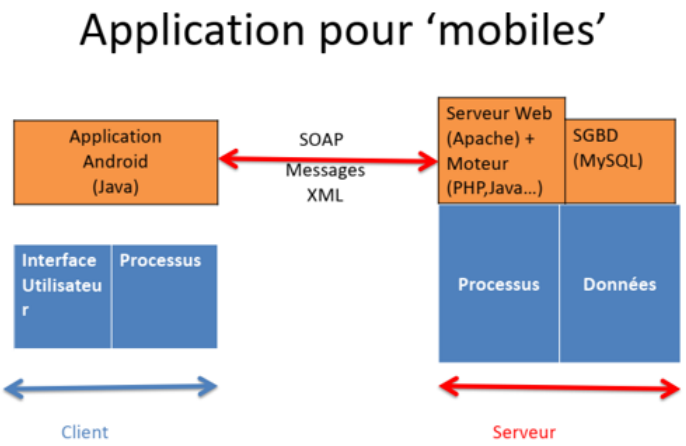


FIGURE 3.2 – L'architecture d'une application mobile.

3.2.4 GitHub

[3] GitHub est un mot valise qui associe le terme Git à Hub qui fait référence à la nature collaborative du service. Son logo est une mascotte représentant un chat avec des tentacules de poulpe. GitHub est un site web conçu pour fédérer et partager le code source d'un projet de

développement d'application mis à œuvre par plusieurs programmeurs. GitHub propose une offre d'entrée de gamme gratuite baptisée Free. Il autorise un nombre de repository (publics et privés) et d'utilisateurs illimités. Mais évidemment, il impose des limitations, ne permet pas de dépasser 2000 minutes (soit environ 33 heures) d'exécution de GitHub Actions par mois, et surtout il plafonne le stockage de packages à 500 Mb.

3.3 Langages de programmation

3.3.1 Java

C'est un langage de programmation orienté objet, développée initialement par Sun Microsystems puis acquise par Oracle à la suite du rachat de l'entreprise. le code Java est compilé pour une machine dite virtuelle : le code machine résultant est nommé "Byte Code". Il permet de créer des logiciels compatibles avec de nombreux systèmes d'exploitation (Windows, Linux, Macintosh, Solaris), de développer des programmes pour téléphones portables et assistants personnels. depuis 2007, Java est passé en "Open Source" ce qui signifie que le code source de l'environnement d'exécution Java est disponible sur Internet et que vous pouvez contribuer (de différentes manières) au développement futur de Java. Enfin le langage Java sert à créer des logiciels dans des environnements très divers : applications sur client lourd (JFC) ; applications Web, côté serveur (servlets, JSP, Struts, JSF) [2].

3.3.2 PHP

Signifie d'abord Personal Home Pages puis HypertextPreProcessor. Est un langage de programmation libre, de script HTML exécuté du côté du serveur.Sa syntaxe est largement inspirée du langage C, de Java et de Perl, avec des améliorations spécifiques. Le but de ce langage est d'écrire rapidement des pages HTML dynamiques, via généralement un serveur HTTP. C'est un langage interprété qui au même titre que Perl, Python ou TCL est capable de lancer des scripts interactifs ou non. On peut même utiliser PHP pour créer des interfaces graphiques (extension GTK). Le minimum nécessaire et vital pour apprendre PHP est donc l'interpréteur PHP lui-même sur un environnement supporté (Unix, Windows, Mac, ...).Utilisable en local (Serveur local : wamp, easyphp,etc..) [28].

3.3.3 XML : Extensible Markup Language

Le langage XML dérive de SGML (Standard Generalized Markup Language) et de HTML (HyperText Markup Language). Comme ces derniers, il s'agit d'un langage orienté texte et formé de balises qui permettent d'organiser les données de manière structurée. Développé et standardisé par le World Wide Web Consortium à la fin des années 1990, il répondait

à l'objectif de définition d'un langage simple, facile à mettre en application. Utilisé pour le stockage de documents que pour la transmission de données entre applications. Sa simplicité, sa flexibilité et ses possibilités d'extension ont permis de l'adapter à de multiples domaines allant des données géographiques au dessin vectoriel en passant par les échanges commerciaux. De nombreuses technologies se sont développées autour de XML et enrichissent ainsi son environnement [10].

3.4 Implémentation de la base de données

3.4.1 MYSQL

[17] Est un système de gestion de bases de données relationnelles (SGBDR), Open source, ce qui signifie que son code source est librement disponible, et utilise le langage SQL "Structured Query Language" : le langage standard pour les traitements de bases de données. Un serveur de bases de données stocke les données dans des tables séparées plutôt que de tout rassembler dans une seule table. Cela améliore la rapidité et la souplesse de l'ensemble. Les tables sont reliées par des relations dénies, qui rendent possible la combinaison de données entre plusieurs tables durant une requête .

3.5 Bibliothèques utilisées

3.5.1 Volley

[9] c'est une librairie HTTP qui permet facilement de faire des appels réseaux rapides sur Android. Disponible à partir d'Android 2.2 (API Level 8- Froyo). Elle utilise les méthodes GET / POST / PUT / DELETE du protocole HTTP et est particulièrement efficace pour des téléchargements courts et rapides tels que le téléchargement d'images ou de données utilisateurs. En effet, l'un des nombreux avantages de Volley est que tous les résultats des requêtes sont automatiquement sauvegardés en cache (peut-être désactivé) afin de pouvoir être réutilisé, si besoin, plus tard.

3.6 Mise en application du Framework proposé

Dans cette section, nous présentons la mise en application du Framework proposé à travers les 3 phases déjà énoncées (chapitre 2) sur un cas d'étude réel relatif à la gestion des patients.

3.6.1 Phase 1 : Pré-phase (Phase de planification)

Cette phase est la plus importante dans le cycle de développement Scrum puisqu'elle influence directement la réussite des sprints et en particulier le premier. Nous menons cette phase à travers deux étapes : (1) Collection des use stories (2) Planification des itérations. Ces étapes sont successivement décrites.

3.6.1.1 Étape 1 : Collection des user stories

Pour pouvoir concevoir cette application de gestion des patients, le Product Owner (Mme.Ait Abdelouhab) à lister et préciser les idées d'une manière claire, et défini sur papier afin de créer les user stories (histoires des utilisateurs) correspond aux besoins de notre application. Dans cette étape, nous donnons les besoins fonctionnels d'une manière globale :
- Authentification dès l'utilisateur de l'application.
- Gérer les comptes utilisateurs.
- Gérer / Consulter des rendez vous.
- Gérer / Consulter les dossiers médicaux.

3.6.1.2 Étape 2 : Planification des itérations (Planning sprint)

Dans cette étape, en tant que équipe Scrum, nous étions tous réuni (Product Owner : Mme Ait Abdelouhab ; Scrum master : Mme. Azoui ; Equipe de développement : Mr. kenouche et Mr. Benziane) afin de définir un ensemble d'actions (Sprint 0) à réaliser avant d'entamer les différents sprints. En effet, dans le sprint 0, nous avons :
- Identification des acteurs du systèmes de gestions des patients.
- Écriture et priorisation du Product Backlog.
- Estimation et planification des Sprints et éventuellement des Releases. Par ailleurs, les différentes actions du sprint 0 sont réalisées et enregistrées dans un logiciel qui est JIRA. Ce dernier peut être hébergé dans le Cloud (JIRA est un outil de gestion du travail pour toutes sortes de cas d'usage, de la gestion des exigences et des cas de test au développement logiciel Agile).

Dans ce qui suit, nous présentons les différentes actions du sprint 0 cités ci-dessus.

(a) Identification des acteurs et leurs rôles : Un acteur est une entité externe qui interagit directement avec le système (utilisateur, dispositif matériel ou autre système). Un acteur peut consulter et/ou modifier directement l'état du système, en émettant et/ou en recevant des messages éventuellement porteurs de données. La notation graphique d'un acteur dans le langage de modélisation unifié (UML), se fait par un simple dessin d'un personnage

(stick man) accompagné au-dessous de par le nom de l'acteur ce qui permet d'avoir un repère précis. Les principaux acteurs de notre application mobile sont décrits comme suit :

(1) L'administrateur de l'application mobile : 1. a pour rôle d'administrer l'environnement du système de l'application. Il possède un compte de type administrateur, responsable de la gestion des comptes des utilisateurs, et la maintenance de l'application.

(2) Le patient : a pour rôle de consulter son dossier médical et de suivre en temps réel ses résultats, prescriptions, bilans, Etc.

(3) L'équipe médicale : C'est l'ensemble des médecins (généralistes, spécialistes), des infirmiers, des secrétaires ainsi que les assistants médicaux :

(a) Le médecin : c'est l'utilisateur principal du système. Cette unité extérieure interagie avec le système via une interface graphique. Sa mission est de faire introduire dans l'application toutes les informations sur toutes les opérations effectuées durant la consultation, notamment le saisi des rapports et les prescriptions.

(b) Infirmier : sont les personnes qui soigne des malades et s'en occupe des patients, sous la direction des médecins.

(c) Secrétaire médicale : elle prend en charge la gestion des rendez-vous vérifie la file d'attente des patients avec les données introduites dans l'application de gestion.

(b) Écriture et priorisation du Product Backlog :

Le tableau (3.1) présente les différentes fonctionnalités de notre product backlog :

ID	Thème	User Story	Priorité
1	Authentification	En tant Que : Utilisateur de l'application (administrateur, personnel médicale, patient, secrétaire) je voudrai m'authentifier pour accéder à l'application.	1
2	Gestion des comptes des utilisateurs de l'application	En tant que administrateur je voudrai gérer les comptes des médecins, infirmiers, sécuritaire	2
3	Gestion des rendez vous	En tant que secrétaire je voudrais afficher liste des rendez-vous , ajouter rendez-vous de les patients.	3
4	Consulter le planning des rendez-vous	En tant que médecin je voudrais consulter le planning des rendez-vous, comme je peux supprimer un rendez-vous.	4

Continuation of Table ??			
ID	Thème	User Story	Priorité
5	Gestion des dossier médicaux	En tant que médecin, je voudrais (Consulter, ajouter, Modifier) d'ordonnance, fiche, résultat d'analyse.	4
6	Gestion de la liste des patients	En tant que secrétaire, équipe médecin, je voudrais (Ajouter, Supprimer, Modifier) un patient, aussi consulter ses propres informations.	4
7	Consultation des dossiers médicaux	En tant que (Infirmier), je voudrais consulter l'ordonnance, résultats d'analyse, et la Fiche	4
8	Consulter la disponibilité des rendez vous	En tant que patient, je voudrais Consulter la disponibilité des rendez-vous de cet clinique privé, et prendre un rendez vous	5
9	Consulter mon dossier médical	En tant que patient je voudrais consulter mes résultats , mon ordonnance,résultats, et la fiche médicale.	5

Tableau 3.1 – Backlog du produit.

(c) Estimation et planification des Sprints et éventuellement des Releases avec Jira (Cloud backlog product) :

Dans cette section nous avons utilisé le fournisseur du cloud SAAS qui est Jira afin de réaliser notre sprint backlog, disponible gratuitement pour un essai de 14 jours. La figure (3.3) nous montre ça :

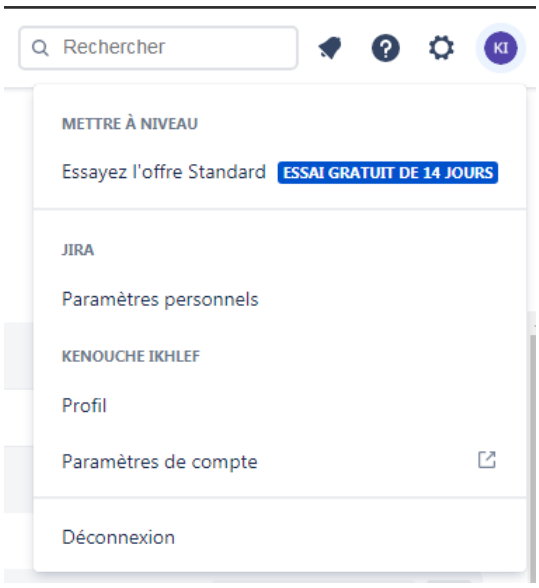


FIGURE 3.3 – La durée d'essai gratuite pour JIRA .

La figure (3.4) montre la mise en pratique du product backlog sous le logiciel Jira. Pour notre travail, on a obtenu 5 sprints. Le backlog sur Jira contient des tickets qui sont tous estimé en remplissant les champs story points et tous affecté à chaque membre de l'équipe de développement (exemple : ticket Authentification vaut 5 story points, notre projet vaut 81 story point), ce qui permet d'organiser les sprints et de les prioriser. Chaque sprint se compose des tickets et possède : un nom, une duré, une date de début et de fin, et l'objectif du sprint.

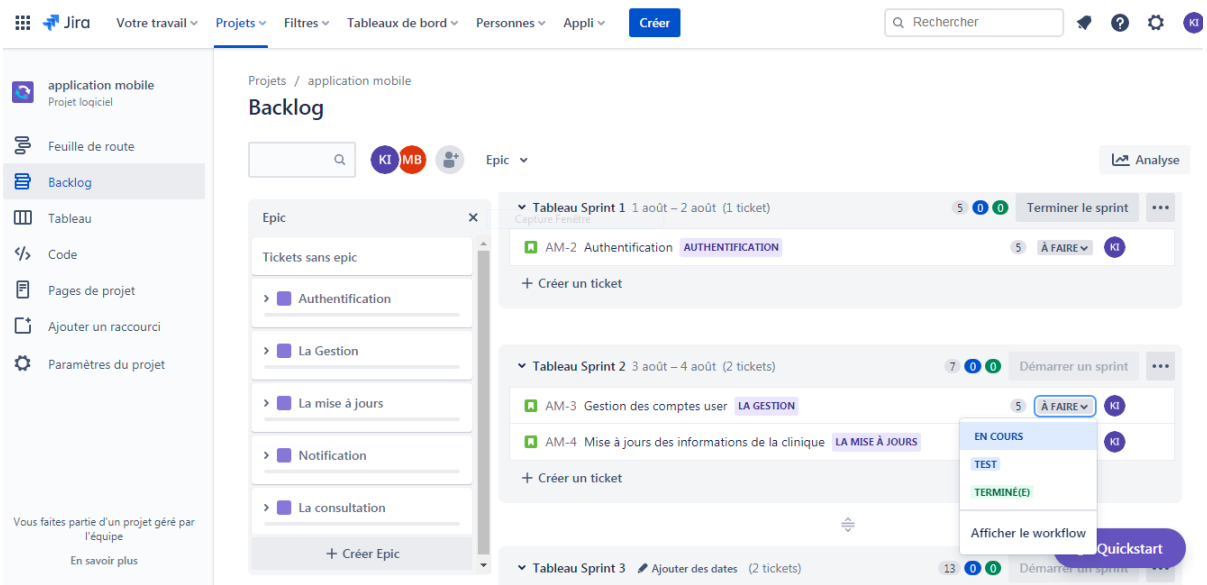


FIGURE 3.4 – Cloud product backlog.

3.6.2 Phase 2 : Phase de développement

Lors de cette phase, l'équipe de développement réaliser le sprint backlog définit lors de l'étape précédente. Pour ce faire, quatre étapes sont réalisées itérativement à savoir la conception, l'implémentation, le test unitaire, et la validation. La durée fixe de cette itération (sprint) est d'une semaine à 4 semaines. Ces étapes sont décrites ci-après.

3.6.2.1 Étape 1 : Conception

Durant cette étape, nous avons réalisé la modélisation de sprint backlog en utilisant le langage de modélisation UML (Unified Modeling Languge) [15],[29]. En fait, dans cette étape, nous avons élaboré les différents diagrammes UML les plus importants pour chaque sprint, à savoir :
a. Diagramme de cas d'utilisation ;
b. Diagramme de séquence ;
c. Diagramme de classe global de notre application.
Dans ce qui suit, nous allons modéliser les 5 sprints définit auparavant.

3.6.2.1.1 Sprint 1 : Authentification

Pour accéder à notre application mobile en tant que patient, médecin, infirmier, secrétaire, administrateur, je dois m'authentifier en insérant nom d'utilisateur et le mot de passe de mon compte. La figure 3.5 illustre le diagramme de cas d'utilisation correspond au sprint(1) :

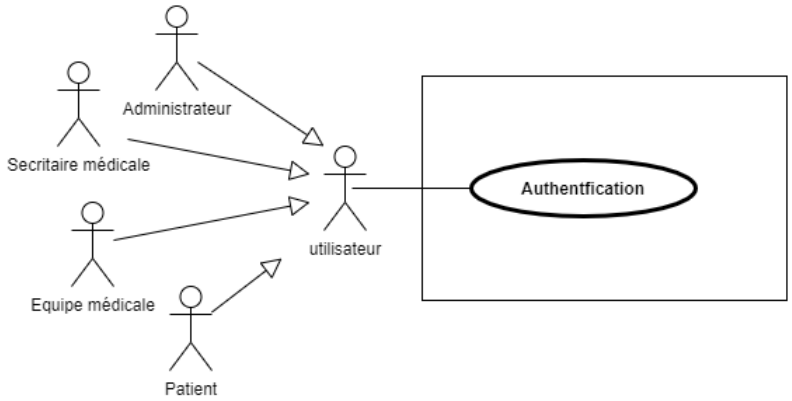


FIGURE 3.5 – Diagramme de cas d'utilisation Authentification.

Le tableau (3.2) montre la description textuelle du cas d'utilisation correspond au cas d'utilisation « Authentification » :

Nom	Authentification		
Résumé	Vérifier de l'identité de l'utilisateur		
Acteur	Utilisateurs		
Liens	Include	/	
		Extends	/
Description	L'authentification permet de vérifier si l'utilisateur est autorisé d'accéder à des fonctionnalités réservées à un type d'utilisateur donné.		
Pré-conditions	L'utilisateur à son propre login et mot de passe		
Scénario nominal	1. L'utilisateur sélectionne le type d'utilisateur. 2. Le system affiche formulaire de connexion. 3. L'utilisateur saisit son email et mot de passe et puis valide. 4. Le system vérifié l'existence du compte utilisateur et validité de son mot de passe. 5. Le système affiche l'interface correspond à l'utilisateur.		
Post-conditions	L'utilisateur est authentifié et accédé au system		

Tableau 3.2 – Description textuelle du cas utilisation « Authentification ».

La figure (3.6) montre le diagramme de séquence correspondant au cas d'utilisation « Authentification » du sprint 1 :

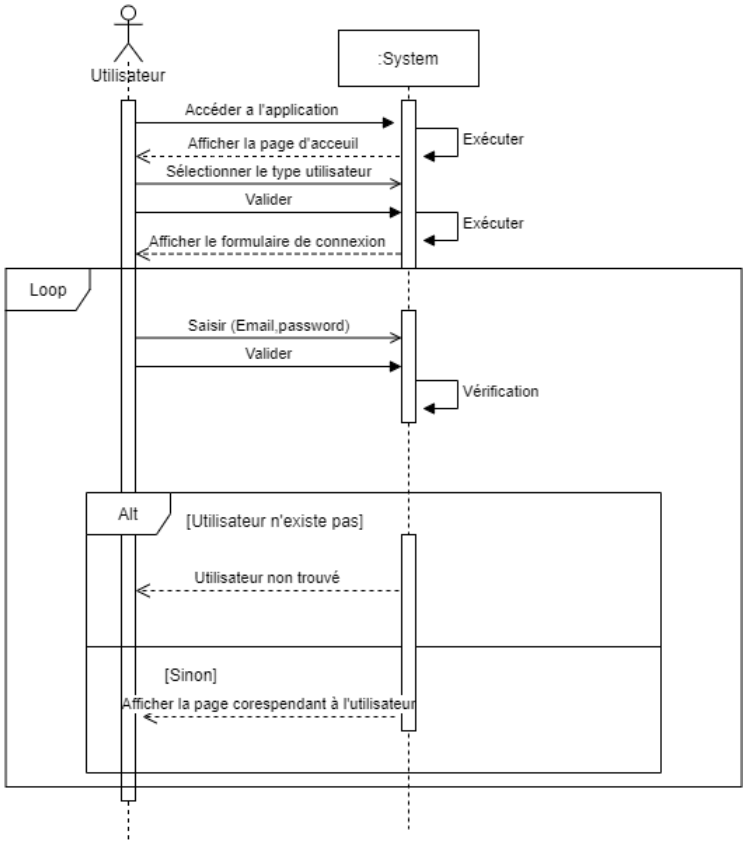


FIGURE 3.6 – Diagramme de séquence "Authentification" .

3.6.2.1.2 Gestion des comptes des utilisateurs de l'application

L'administrateur gère les comptes user en ajoutant , supprimant, modifiant les comptes des patients , des infirmiers , des médecins, des infirmiers, et mets à jours les informations correspondant à l'adresse de la clinique et son numéro de téléphone. La figure (3.7) montre le diagramme de cas d'utilisation correspond au «Gestion des comptes des personnels médicaux» :

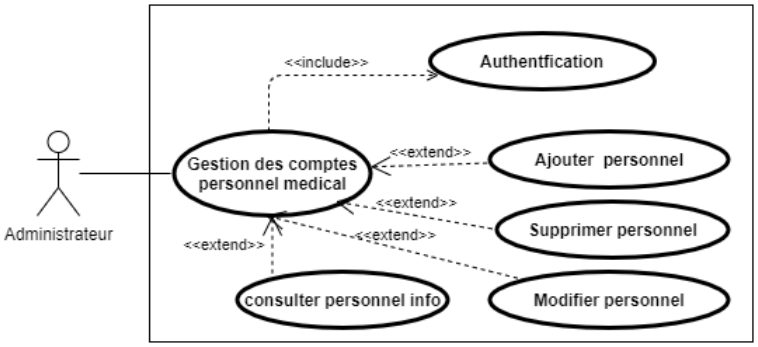


FIGURE 3.7 – Diagramme de cas d'utilisation «Gestion des comptes des personnels médicaux».

Le tableau (3.3) présente la description textuelle de cas d'utilisation « Gestion des comptes des personnels médicaux » :

<table>
<tr><td colspan="4">Début de table</td></tr>
<tr><td>Nom</td><td colspan="3">Gestion des comptes des personnels médicaux</td></tr>
<tr><td>Résumé</td><td colspan="3">L'administrateur gère les comptes des infirmiers, médecins, secrétaire.</td></tr>
<tr><td>Acteur</td><td colspan="3">Administrateur</td></tr>
<tr><td rowspan="2">Liens</td><td>Include</td><td colspan="2">Authentification</td></tr>
<tr><td></td><td>Extends</td><td>Ajouter personnel, consulter personnel info, modifier Personnel info, Supprimer personnel.</td></tr>
<tr><td>Description</td><td colspan="3">La gestion des comptes des personnels médicaux permet d'ajouter, de supprimer et de modifier un utilisateur et lui donné des privilèges correspondants.</td></tr>
<tr><td>Pré-conditions</td><td colspan="3">L'utilisateur est authentifié au system correctement en tant qu'administrateur Qui possède les privilèges pour la gestion des comptes personnel médical.</td></tr>
<tr><td>Scénario nominal</td><td colspan="3">1- L'administrateur clique sur le bouton Gérer médecin / infirmier / secrétaire.
2- Le système affiche l'interface correspondante (liste personnel).
3- L'administrateur recherche le patient par nom.
[Si patient n'existe pas] :
Ajouter personnel :
1- L'administrateur demande l'ajoute de personnel.
2- Le système affiche le formulaire d'ajoute.
3- L'administrateur remplit le formulaire d'ajout et valide.
4- Le système vérifie les données introduites et affiche l'erreur en cas d'erreur, sinon, le système affiche un message 'Ajout avec succès'.
[Si personnel existe] :
- L'administrateur Sélection le personnel.
Modifier personnel :
1- L'administrateur demande la modification.
2- Le système affiche personnel info.
3- L'administrateur modifie les données de personnel et valide.
4- Le système affiche le message de modification avec succès.
Consulter personnel info :
1- L'administrateur demande de consulter information personnel..
2- Le système affiche les données correspondantes.
Supprimer personnel :
1- L'administrateur demande la suppression d'un personnel.
2- Le système demande de la confirmation de suppression.
3- L'administrateur valide la suppression.
4- Le système affiche le message de suppression avec succès.</td></tr>
<tr><td>Post-conditions</td><td colspan="3">L'ajoute/modification/suppression d'un personnel avec succès.</td></tr>
</table>

Tableau 3.3 – Description textuelle du cas utilisation «Gestion des comptes personnels médicaux».

fin de table

La figure (3.8) montre le diagramme de séquence du cas utilisation «Gestion des comptes personnels médicaux» :

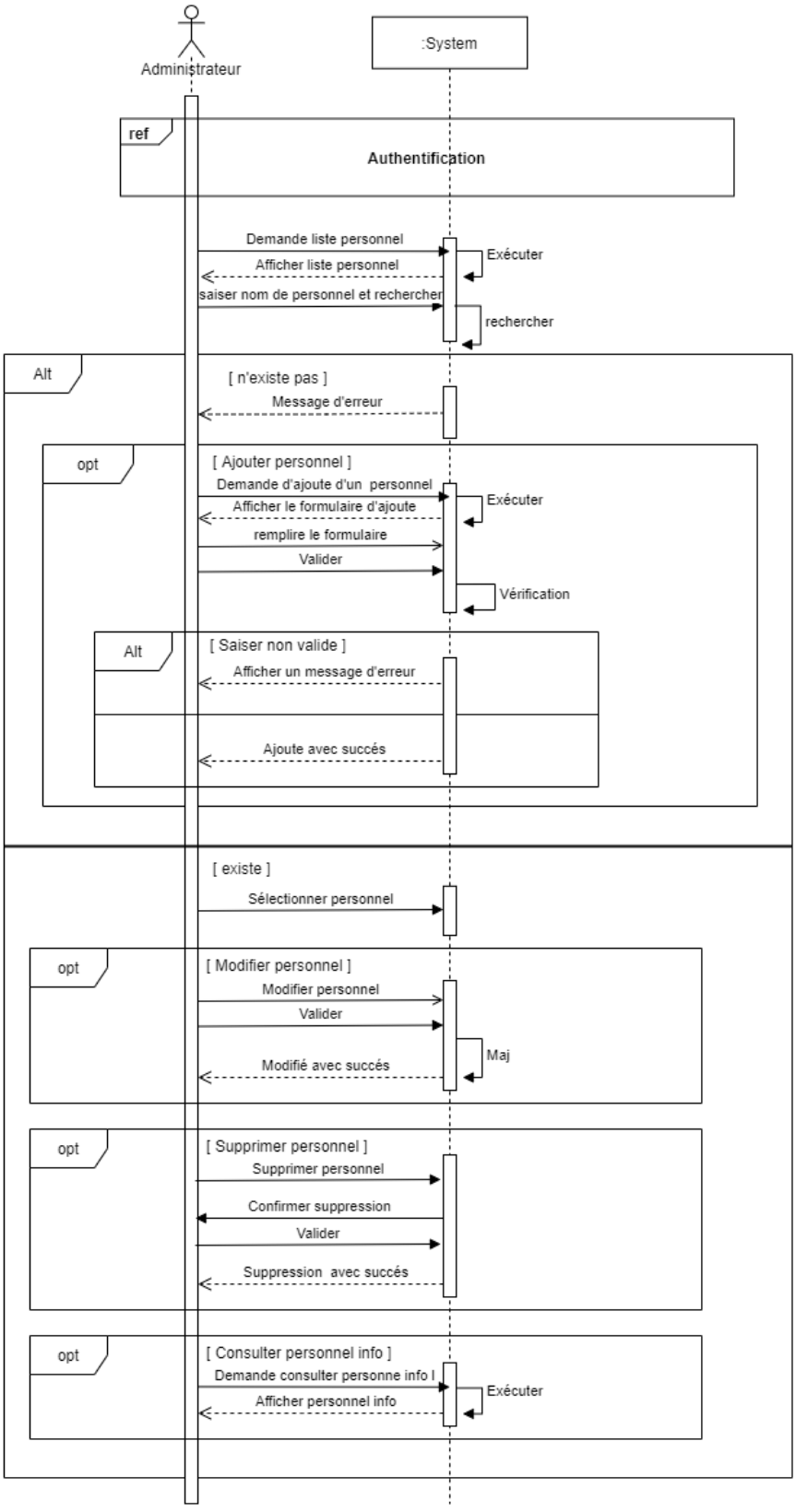


FIGURE 3.8 – Diagramme de séquence de «Gestion des comptes personnels médicaux».

3.6.2.1.3 Sprint 3 : Gestion des rendez-vous

Dans ce sprint le secrétaire affiche la liste des rendez-vous, supprime ou ajoute un rendez-vous. La figure (3.9) présente le diagramme de cas utilisation «Gestion des rendez-vous patients» :

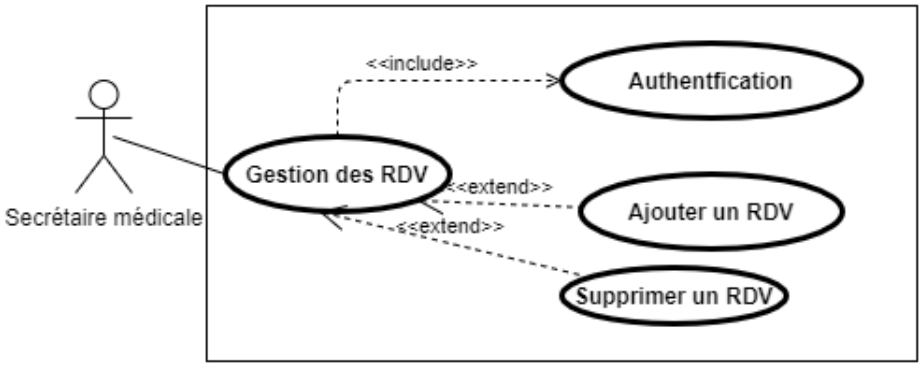


FIGURE 3.9 – Diagramme de séquence de cas d'utilisation «Gestion des rendez-vous patients».

Le tableau (3.4) décrit la description textuelle de cas d'utilisation «Gestion des rendez-vous patients» :

Début de table			
Nom	Gestion des rendez-vous		
Résumé	Le Secrétaire gère les rendez-vous des patients.		
Acteur	Secrétaire.		
Liens	Include	Authentification	
		Extends	Ajouter rendez-vous, supprimer Rendez-vous.
Description	Le secrétaire médical gère les différents rendez-vous en ajoutant , supprimant ou consultant les prises de RDV des patients		
Pré-conditions	Le Secrétaire médical s'est correctement authentifié et le patient est déjà inscrit.		

Scénario nominal	1- Le secrétaire médical demande la gestion de planning RDV. 2- Le système affiche la liste des RDV. **Ajouter RDV :** 1- Le secrétaire médical demande l'ajout des RDV. 2- Le système affiche liste des patients. 3- Le secrétaire sélectionne le patient et demande d'ajouter son RDV. 4- Le système affiche le formulaire d'ajout. 5- Le secrétaire remplit le formulaire et le valide. 6- Le système vérifie les données introduit, si un champ manquant alors le système affiche un message d'erreur, et la secrétaire réintroduit les données. 7- Le système effectue la vérification de la disponibilité de l'heure RDV. -Si l'heure du RDV disponible est disponible alors le système affiche message RDV et l'ajoute dans la liste des RDV. **Supprimer RDV :** 1- Le secrétaire médical sélectionne le RDV, clique sur supprimer RDV. 2- Le système demande la confirmation de suppression. 3- Le secrétaire médical confirme la suppression. 4- Le système affiche un message de suppression avec succès.
Post-conditions	Ajouter / supprimer RDV avec succès.

Tableau 3.4 – Description textuelle du cas d'utilisation «Gestion des rendez-vous».

fin de table

La figure (3.10) montre le diagramme de séquence du cas d'utilisation «Gestion des rendez-vous» :

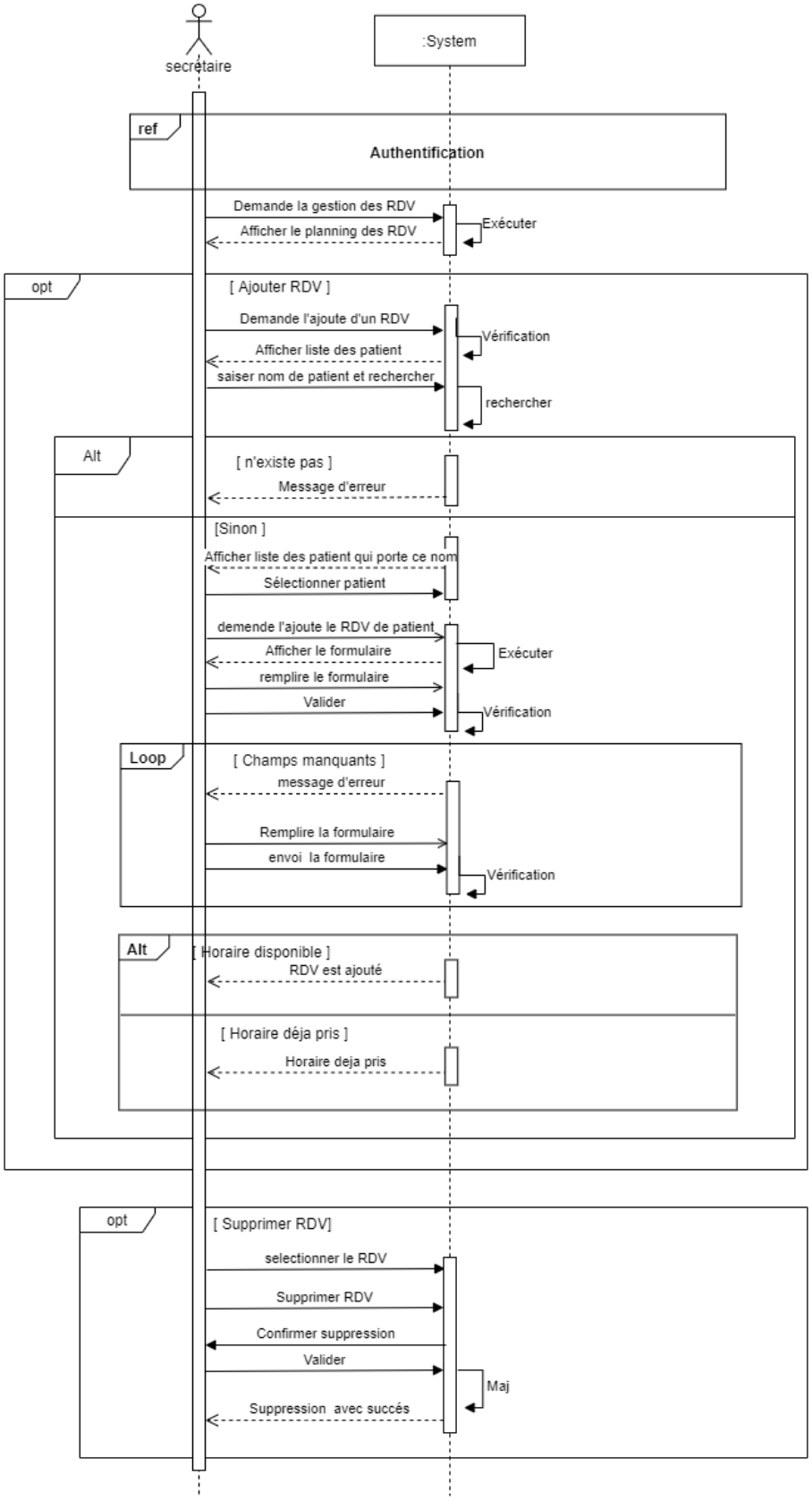


FIGURE 3.10 – Diagramme de séquence du cas d'utilisation «Gestion des rendez-vous».

3.6.2.1.4 Sprint 4 : Consulter planning rendez-vous/Gestion des dossiers médicaux/ Gestion des listes patients/ Consulter dossier médical.

(a) Consulter planning rendez-vous

La figure (3.11) montre le diagramme de cas d'utilisation «Consulter planning des RDV» :

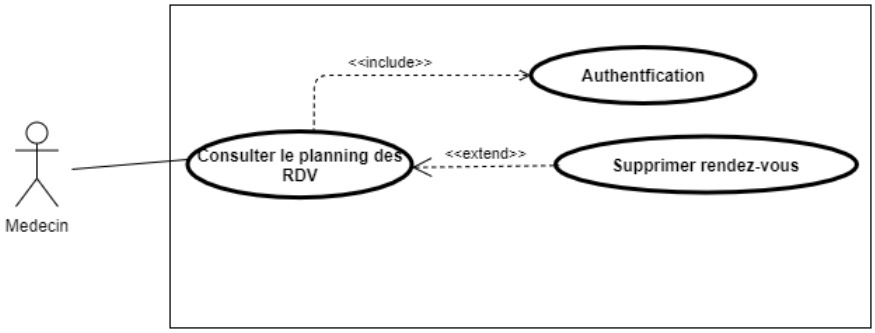


FIGURE 3.11 – Diagramme de cas d'utilisation «Consulter planning des RDV».

Le tableau (3.5) décrit la description textuelles de cas d'utilisation «Consulter planning des RDV» :

Début de table			
Nom	Consulter planning des RDV.		
Résumé	Le médecin peut consulter la liste des rendez-vous, et supprimer un rendez-vous.		
Acteur	Médecin.		
Liens	Include	Authentification	
		Extends	Supprimer rendez-vous.
Description	La consultation de planning des RDV permet au médecin de connaître ses prochain RDV, et de pouvoir les annulés.		
Pré-conditions	Le médecin est authentifié.		
Scénario nominal	1- Le médecin demande de consulter le planning. 2- Le système affiche la liste des RDV. 3- Le médecin sélectionne un RDV et clique sur supprimer RDV. 4- Le système affiche message «rendez-vous supprimé avec succès».		
Post-conditions	Consulter / supprimer RDV avec succès.		

Tableau 3.5 – Description textuelle du cas utilisation «Consulter planning RDV».

fin de table

La figure (3.12) montre le diagramme de séquence «Consulter planning des RDV» :

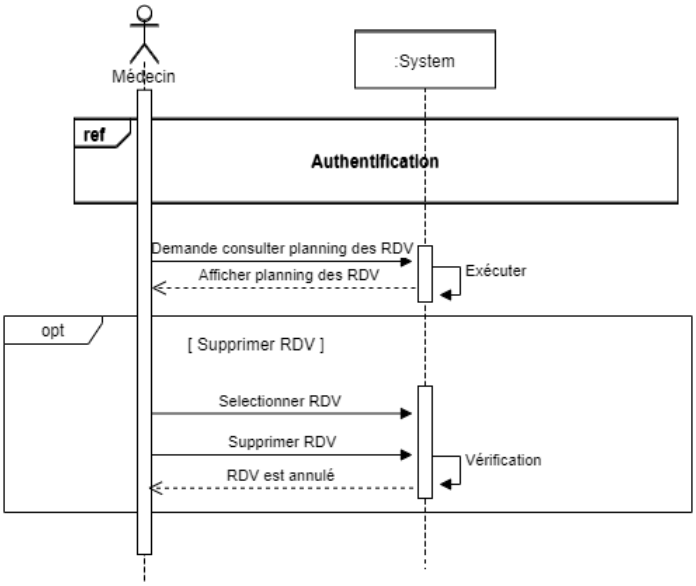


FIGURE 3.12 – Diagramme de séquence «Consulter planning des RDV».

(b) Gestion des dossiers médicaux

La figure 3.13 suivante décrit le diagramme de cas utilisation «Gestion des dossiers médicaux».

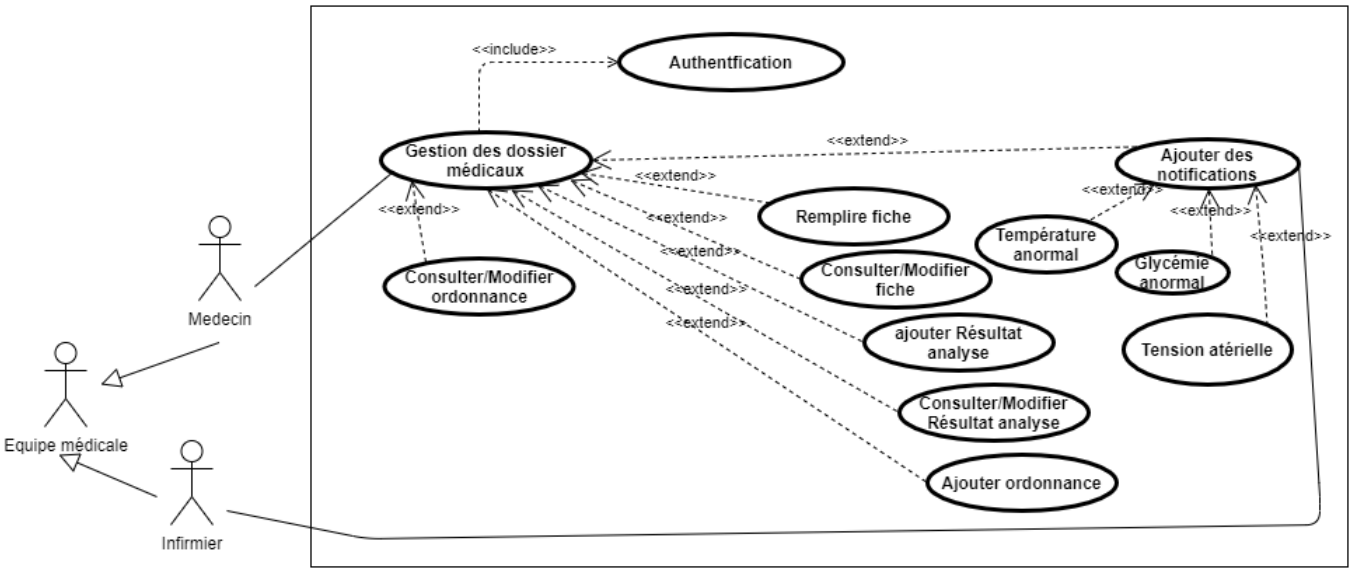


FIGURE 3.13 – Diagramme de cas utilisation «Gestion des dossiers médicaux».

Le tableau (3.6) suivant décrit la description textuelle de cas utilisation «Gestion des dossiers médicaux» :

Début de table			
Nom	Gestion dossier médical.		
Résumé	Le médecin gère les dossiers médicaux des patients.		
Acteur	Médecin - Infirmier		
Liens	Include	Authentification	
		Extends	- Remplir fiche. - Consulter / Modifier fiche. - Ajouter résultats analyse. - Consulter / modifier résultats analyse. - Ajouter ordonnance. - Consulter / Modifier ordonnance. - Ajouter notification (Température, Glycémie, Tension anormale).
Description	La gestion des dossiers médicaux permet au médecin de suivre l'état des patients et faire des ajouts, des modifications sur son dossier médical.		
Pré-conditions	Le médecin et l'infirmier sont authentifiés.		

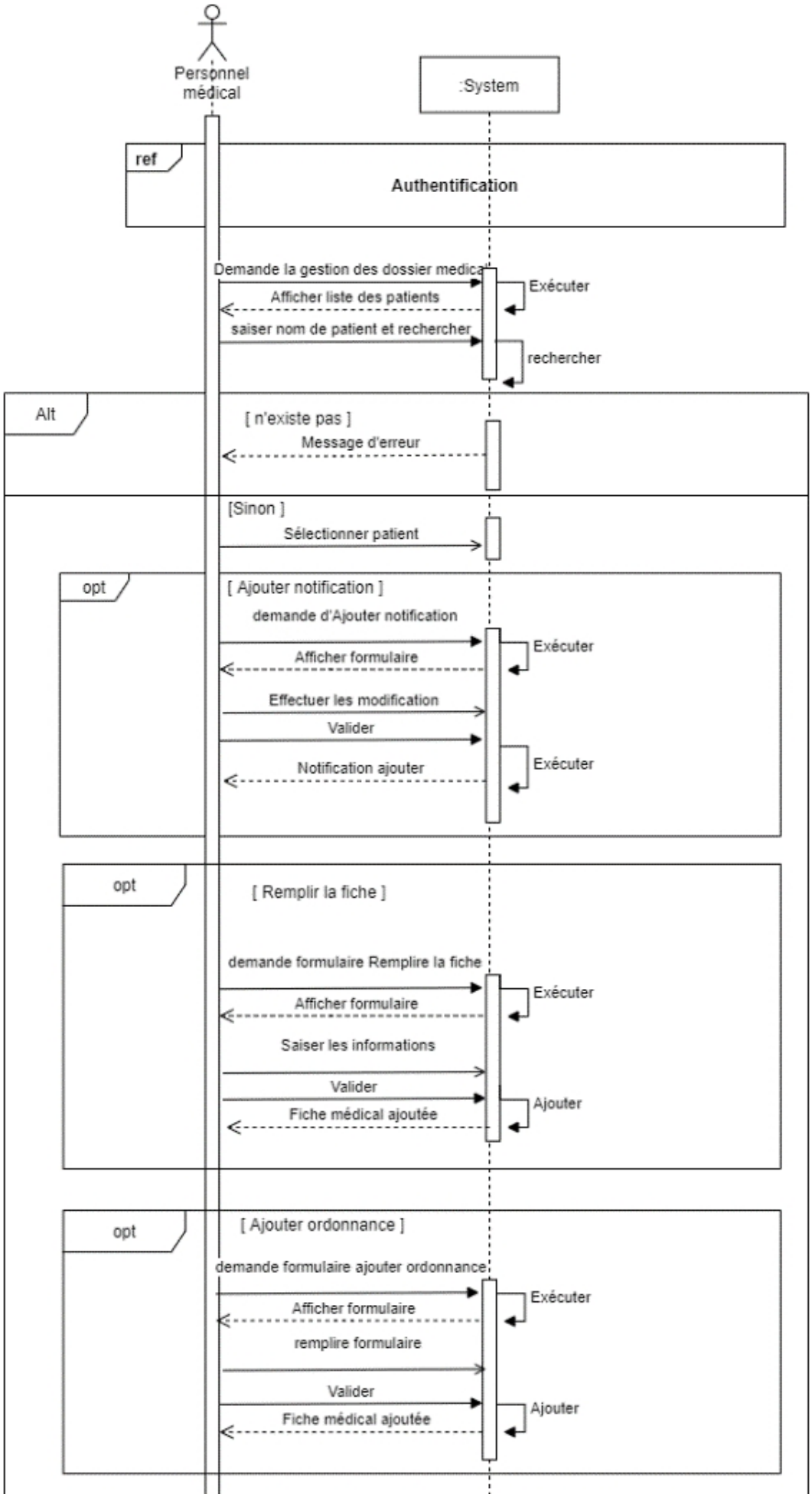
Scénario nominal	1- Le médecin demande la gestion des dossiers médicaux. 2- Le système affiche liste des patients. 3- Le médecin recherche patient par nom. [Si patient n'existe pas] - Le système affiche un message d'erreur. [Si existe] - Le médecin sélectionne le patient. **Ajouter notification :** 1- Le médecin / infirmier demande d'ajouter notification. 2- Le système affiche le formulaire d'ajout. 3- Le médecin / infirmier remplit le formulaire. 4- Le médecin / infirmier valide la notification. 5- Le système affiche notification ajoutée. **Remplir fiche médicale :** 1- Le Médecin demande l'ajout de fiche médicale. 2- Le système affiche le formulaire fiche médicale. 3- Le médecin remplit le formulaire puis le valide. 4- Le système affiche un message «fiche ajoutée ». **Ajouter ordonnance :** 1- Le Médecin demande l'ajout «Ajouter ordonnance». 2- Le système affiche le formulaire ordonnance. 3- Le médecin remplit le formulaire et le valide. 4- Le system affiche message ordonnance ajoutée. **Ajouter résultats analyse :** 1- Le Médecin demande l'ajout de résultats analyse. 2- Le système affiche le formulaire résultats analyse. 3- Le médecin remplit le formulaire et valide. 4- Le système affiche message résultats analyse ajoutée. **Consulter / Modifier fiche médicale :** 1- Le Médecin demande de Consulter / Modifier la fiche médicale. 2- Le système affiche le formulaire Consulter / Modifier fiche médicale. 3- Le médecin Effectue les modifications de fiche médicale et valide. 4- Le système affiche message fiche médicale modifié. **Consulter / Modifier résultats analyse :** 1- Le Médecin demande Consulter / Modifier résultats analyse. 2- Le système affiche le formulaire Consulter / Modifier résultats analyse. 3- Le médecin effectue les modifications de résultats analyse et valide. 4- Le système affiche message résultats analyse modifié. **Consulter / Modifier ordonnance :** 1- Le Médecin demande de Consulter / Modifier ordonnance. 2- Le système affiche le formulaire Consulter / Modifier ordonnance. 3- Le médecin effectue les modifications de résultats analyse et valide. 4- Le système affiche message ordonnance modifié.

Post-conditions	Le médecin accède au dossier du patient souhaité puis introduit des modifications avec succès. Le médecin et l'infirmier ajoutent des notifications avec succès.

Tableau 3.6 – Description textuelle du cas utilisation «Gestion des dossiers médicaux».

fin de table

La figure (3.14) montre le diagramme de séquence correspondant au cas d'utilisation «Gestion des dossiers médicaux» :


Personnel médical
:System
ref
Authentification
Demande la gestion des dossier medica
Exécuter
Afficher liste des patients
saiser nom de patient et rechercher
rechercher
Alt
[n'existe pas]
Message d'erreur
[Sinon]
Sélectionner patient
opt
[Ajouter notification]
demande d'Ajouter notification
Exécuter
Afficher formulaire
Effectuer les modification
Valider
Notification ajouter
Exécuter
opt
[Remplir la fiche]
demande formulaire Remplire la fiche
Exécuter
Afficher formulaire
Saiser les informations
Valider
Ajouter
Fiche médical ajoutée
opt
[Ajouter ordonnance]
demande formulaire ajouter ordonnance
Exécuter
Afficher formulaire
remplire formulaire
Valider
Ajouter
Fiche médical ajoutée

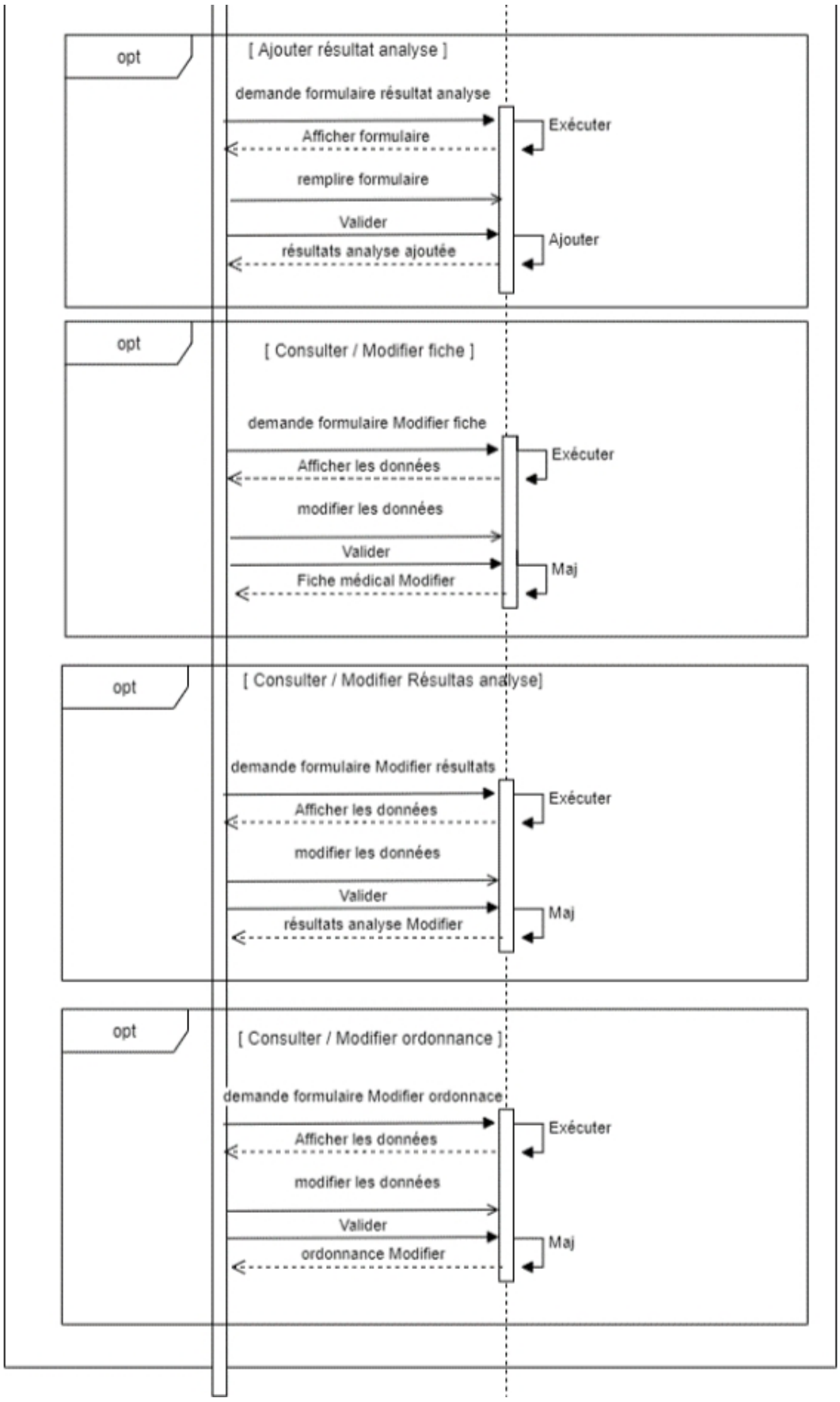


FIGURE 3.14 – Diagramme de séquence du cas d'utilisation «Gestion des dossiers médicaux».

(c) Gestion liste des patient

La figure (3.15) suivante décrit le diagramme de cas utilisation «Gestion de la liste des patients» :

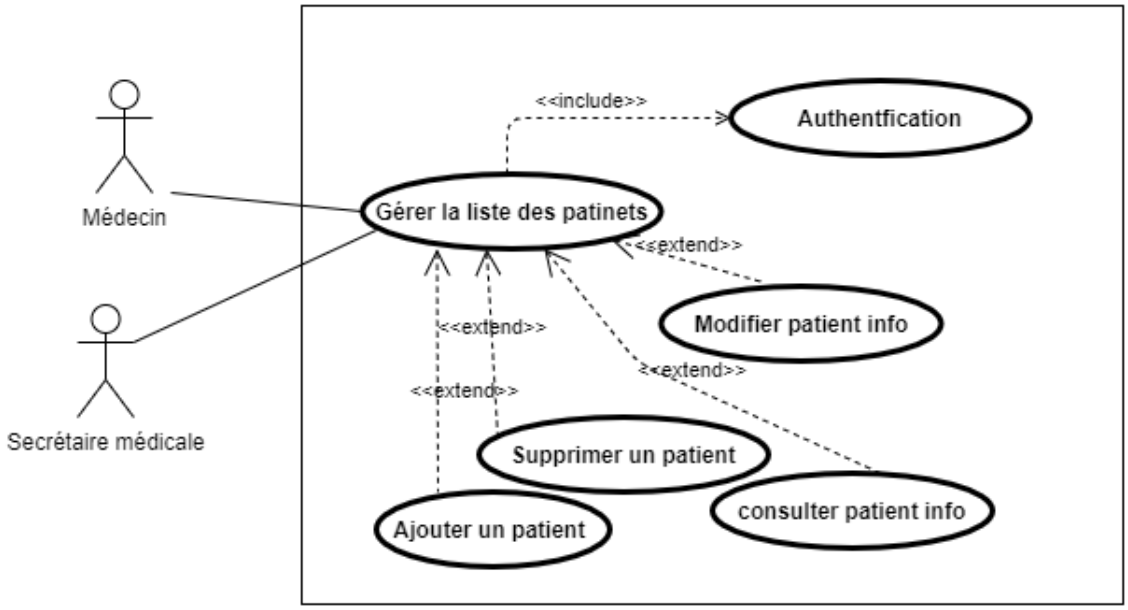


FIGURE 3.15 – Diagramme cas utilisation «Gestion de la liste des patients» .

Le tableau (3.7) décrit la description textuelle de cas d'utilisation «Gestion de la liste des patients» :

Début de table			
Nom	Gérer la liste des patients.		
Résumé	Le médecin et infirmier gèrent la liste et données des patient.		
Acteur	Médecin, secrétaire.		
Liens	Include	Authentification	
		Extends	Ajouter patient, consulter patient info, modifier patient info, Supprimer patient.
Description	La gestion des comptes patients permet aux Médecins et secrétaires d'ajouter, de supprimer et de modifier un patient et lui donner des privilèges correspondant.		
Pré-conditions	Le médecin et infirmier sont authentifiés et possèdent les privilèges pour la gestion des comptes des patients.		

Scénario nominal	1- L'utilisateur clique sur le bouton «Gérer patient». 2- Le système affiche l'interface correspondante (liste patient). 3- L'utilisateur recherche le patient par nom. [Si patient n'existe pas] **Ajouter personnel :** 1- L'utilisateur demande l'ajout de patient. 2- Le système affiche le formulaire d'ajout. 3- L'utilisateur remplit le formulaire d'ajout et valider. 4- Le système vérifie les données introduites, et affiche l'erreur en cas d'erreur, sinon il afficher un message «ajouter avec succès». [Si personnel existe] : - L'utilisateur sélectionne le patient. **Modifier patient :** 1- L'utilisateur demande la modification. 2- Le système affiche personnel info. 3- L'utilisateur modifie les données de personnel et valide. 4- Le système affiche un message «modifier avec succès». **Consulter patient info :** 1- L'administrateur demande «consulter patient info». 2- Le système affiche les données correspondantes. **Supprimer patient :** 1- L'utilisateur demande la suppression. 2- Le système demande de confirmer la suppression. 3- L'utilisateur valide la suppression. 4- Le système affiche un message «supprimer avec succès».
Post-conditions	Consultation /L'ajoute/modification/suppression d'un patient avec succès.

Tableau 3.7 – Description textuelle du cas utilisation «Gestion de la liste des patients».

fin de table

La figure (3.16) suivante décrit le diagramme de séquence correspondant au cas d'utilisation «Gestion de la liste des patients» :

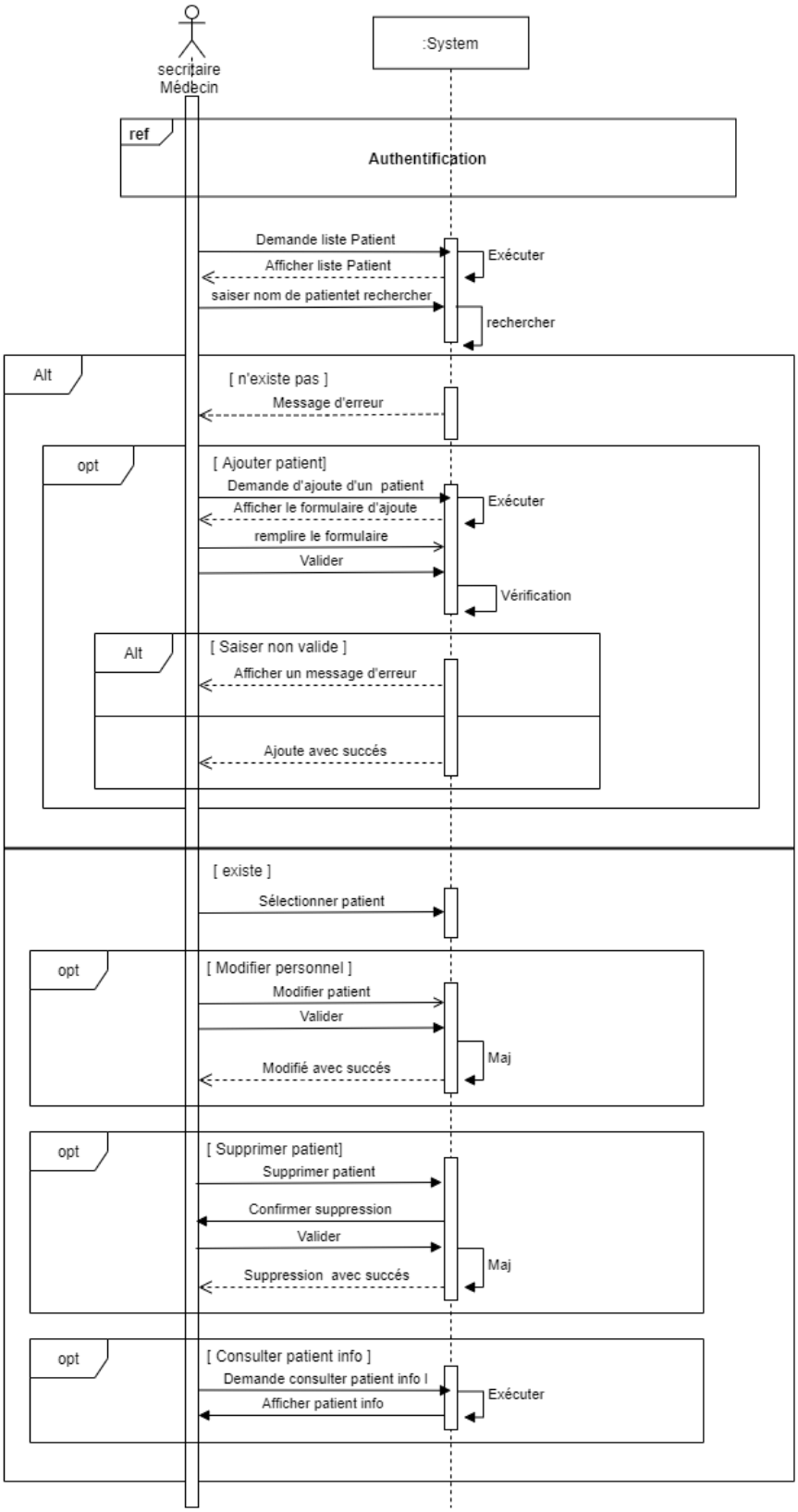


FIGURE 3.16 – Diagramme de séquence du cas d'utilisation «Gestion de la liste des patients».

(d) Consulter dossier médical

la figure (3.17) suivante décrit le diagramme de cas utilisation «Consulter dossier médical» :

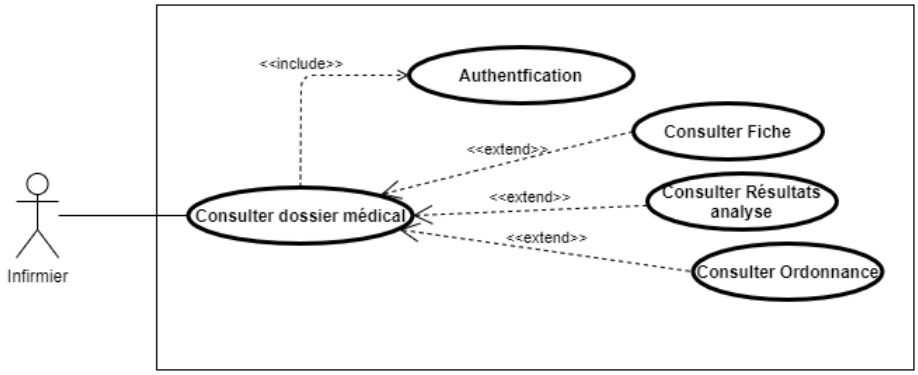


FIGURE 3.17 – Diagramme cas utilisation «Consulter dossier médical».

Le tableau (3.8) décrit la description textuelle de cas utilisation «Consulter dossier médical» par l'infirmier :

Début de table			
Nom	Consulter dossier médical.		
Résumé	L'infirmier gère les comptes des infirmiers, médecins, secrétaire		
Acteur	Infirmier		
Liens	Include	Authentification	
		Extends	- Consulter fiche médicale. - Consulter Résultats analyse. - Consulter ordonnance.
Description	Consulter dossier médical, permet à l'infirmier de suivre l'état des patients d'une manière plus efficace.		
Pré-conditions	L'infirmier est authentifié au système correctement. Qui possède les privilèges pour Consulter les dossiers médicaux.		

Scénario nominal	1- L'infirmier accède à l'espace infirmier. 2- Le système affiche l'interface correspondante (liste personnel). 4- L'infirmier recherche le patient par nom. [Si patient n'existe pas] : -Le système affiche erreur. [Si personnel existe] : - L'infirmier Sélection le personnel. **Consulter fiche médicale :** 1- L'infirmier demande Consulter fiche médicale. 2- Le système affiche la fiche médicale correspondant. **Consulter ordonnance :** 1- L'infirmier demande Consulter l'ordonnance. 2- Le système affiche l'ordonnance médicale correspondant. **Consulter résultat analyse :** 1- L'infirmier demande Consulter l'ordonnance. 2- Le système affiche les résultats d'analyse correspondants.
Post-conditions	Consulter dossier médicale par l'infirmier avec succès.

Tableau 3.8 – Description textuelle du cas utilisation «Consulter dossier médical».

fin de table

la figure (3.18) suivante décrit le diagramme de séquence de cas d'utilisation Consulter dossier médical :

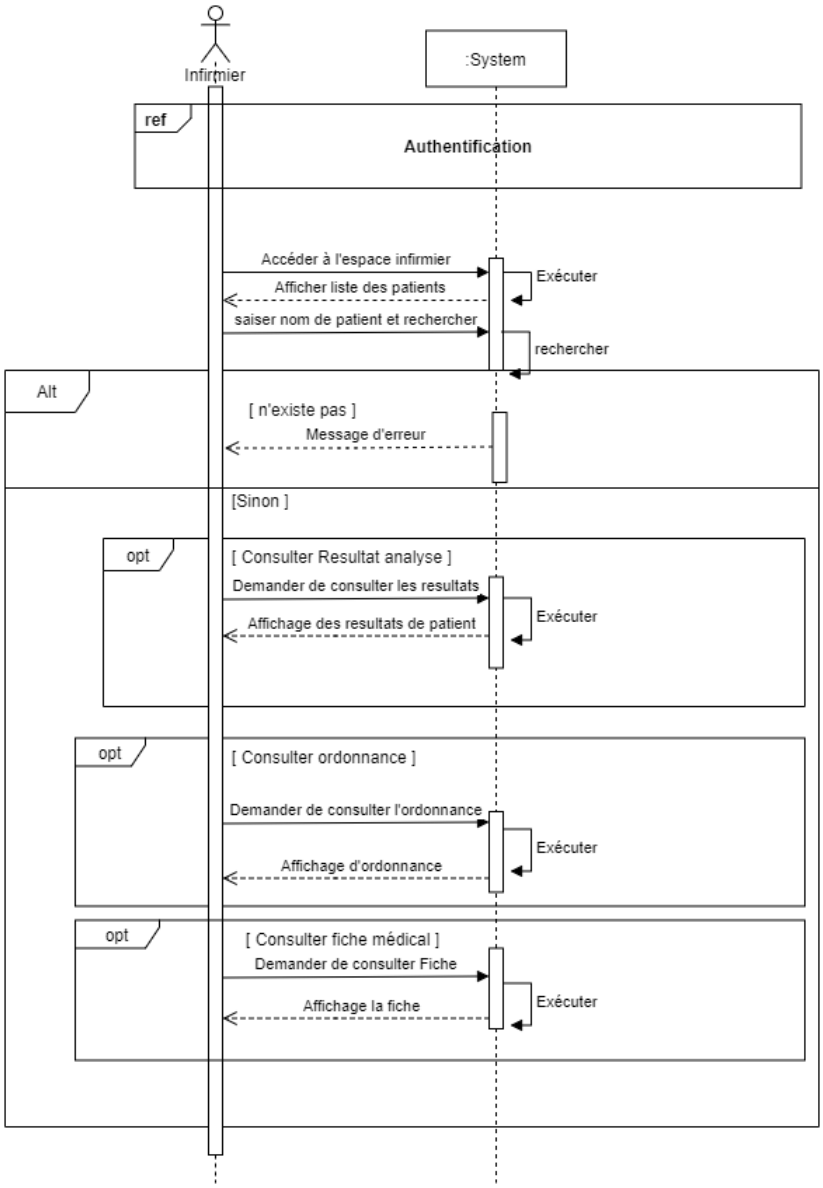


FIGURE 3.18 – Diagramme de séquence du cas d'utilisation «Consulter dossier médical».

3.6.2.1.5 Sprint 5 : Consulter la disponibilité des rendez vous et mon dossier médicale

(a) Consulter la disponibilité des rendez-vous

Pour chaque patient qui veut faire une consultation chez un médecin peut prendre un rendez-vous.

La figure (3.19) suivante montre le diagramme de cas d'utilisation «Consultation de la disponibilité des rendez-vous» :

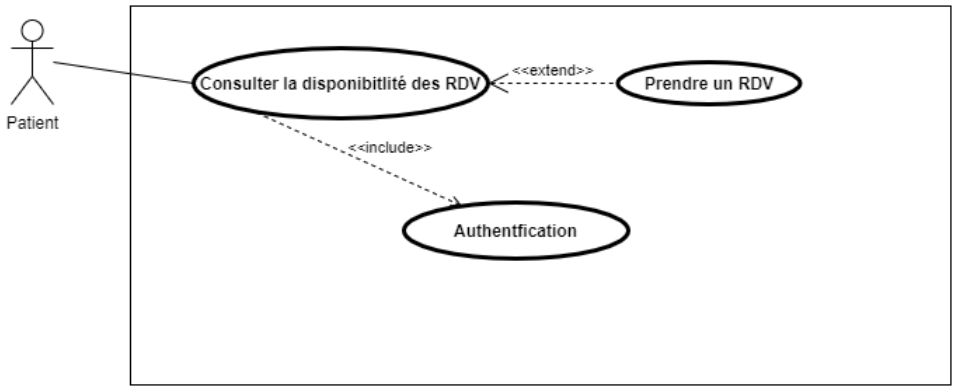


FIGURE 3.19 – Diagramme de cas d'utilisation «Consulter disponibilité des RDV».

Le tableau (3.9) décrit la description textuelle de cas utilisation «Consulter disponibilité des RDV». :

Début de table			
Nom	Consulter disponibilité des RDV.		
Résumé	Le patient capable de prendre un rendez-vous.		
Acteur	Patient		
Liens	Include	Authentification	
		Extends	Prendre rendez-vous.
Description	Consulter disponibilité d'un rendez-vous permet aux patients de prendre un rendez-vous médical et choisir le service médical correspondant.		
Pré-conditions	Patient inscrit est correctement authentifié.		
Scénario nominal	1- Le patient demande «prendre rendez-vous».. 2- Le système affiche le formulaire correspondant. 3- Le patient remplit le formulaire et valide. 4- Le système vérifie les données introduites, s'il y a des champs manquants alors le système affiche un message d'erreur, et le patient réintroduit les données et puis les valident. 5- Le système effectue la vérification de la disponibilité de l'heure RDV. -Si l'heure RDV disponible alors le système affiche message RDV est ajouté.		
Post-conditions	Prendre le rendez-vous avec succès par le patient.		

Tableau 3.9 – Description textuelle du cas utilisation «Consulter disponibilité des RDV»..

fin de table

la figure (3.20) suivante décrit le diagramme de séquence de cas d'utilisation Consulter disponibilité des RDV :

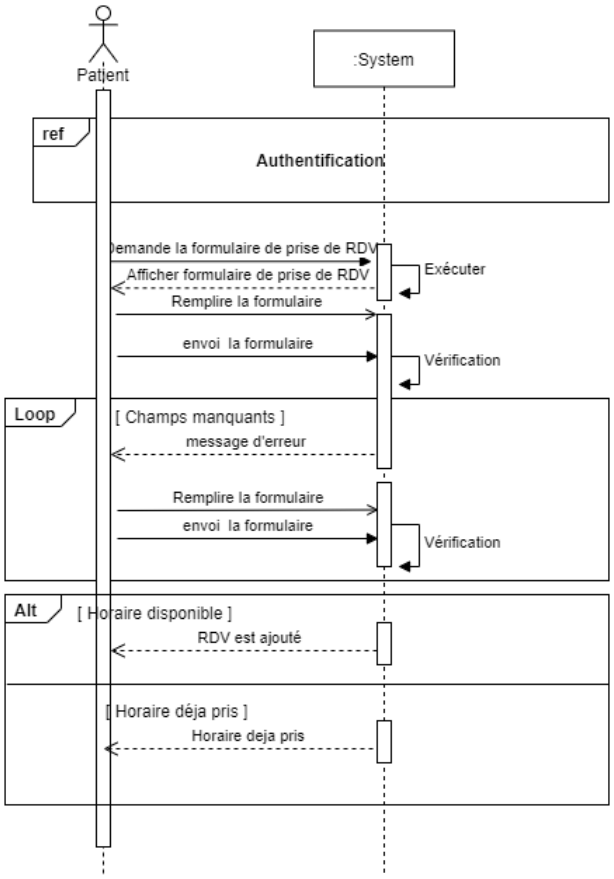


FIGURE 3.20 – Diagramme de séquence du cas d'utilisation «Consulter disponibilité des RDV».

(b) Consulter mon dossier médical

Pour chaque patient qui a fait une consultation chez un médecin, il peut consulter son propre dossier médical.

La figure (3.21) suivante montre le diagramme de cas d'utilisation «Consulter mon dossier médical» :

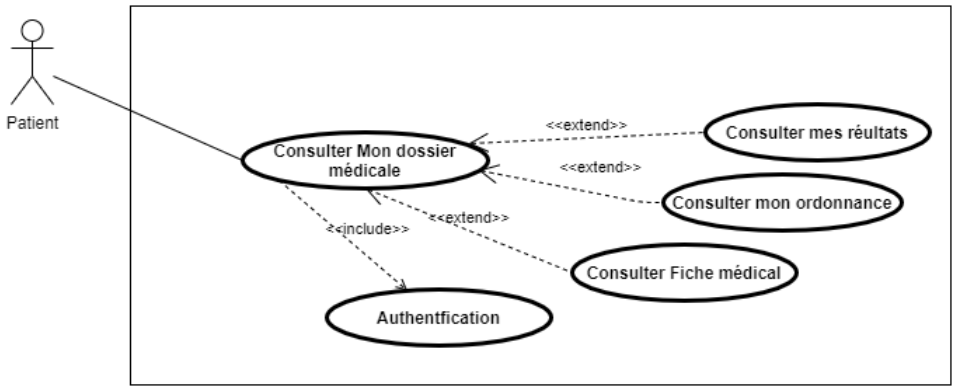


FIGURE 3.21 – Diagramme séquence de cas utilisation «Consulter mon dossier médical».

Le tableau (3.10) décrit la description textuelle de cas d'utilisation «Consulter mon dossier médical». :

Début de table			
Nom	Consulter mon dossier médical.		
Résumé	Le patient consulte son dossier médical.		
Acteur	Patient		
Liens	Include	Authentification	
		Extends	- Consulter fiche médicale. - Consulter Résultats analyse. - Consulter ordonnance.
Description	Consulter mon dossier médical, permet au patient de récupérer son ordonnance, fiche, résultats analyse rapidement sans attendre la version papier		
Pré-conditions	- Le patient est inscrit est authentifié au système correctement. - Le dossier médical de patient existe.		
Scénario nominal	1- Le patient accède à l'espace patient. 2- Le système affiche l'interface correspondante. **Consulter fiche médicale :** 1- Le patient demande de consulter sa fiche médicale. 2- Le système affiche la fiche médicale correspondant. **Consulter ordonnance :** 1- Le patient demande Consulter l'ordonnance. 2- Le système affiche l'ordonnance médicale correspondant. **Consulter résultat analyse :** 1- Le patient demande de consulter ses résultats analyses. 2- Le système affiche les résultats d'analyses correspondants.		
Post-conditions	Consulter le dossier médical par patient avec succès.		

Tableau 3.10 – Description textuelle du cas utilisation «Consulter mon dossier médical».

fin de table

La figure (3.22) suivante décrit le diagramme de séquence du cas utilisation «Consulter mon dossier médical» :

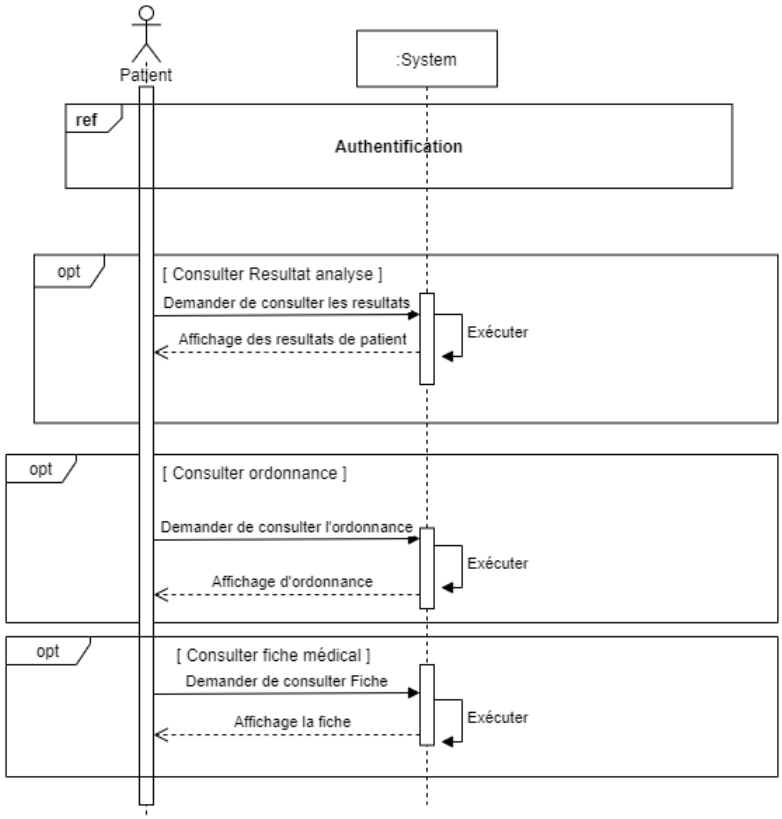


FIGURE 3.22 – Diagramme de séquence du cas utilisation «Consulter mon dossier médical».

La figure (3.23) suivante résume le diagramme de cas d'utilisation globale de notre application mobile de gestion des dossiers médicaux des patients :

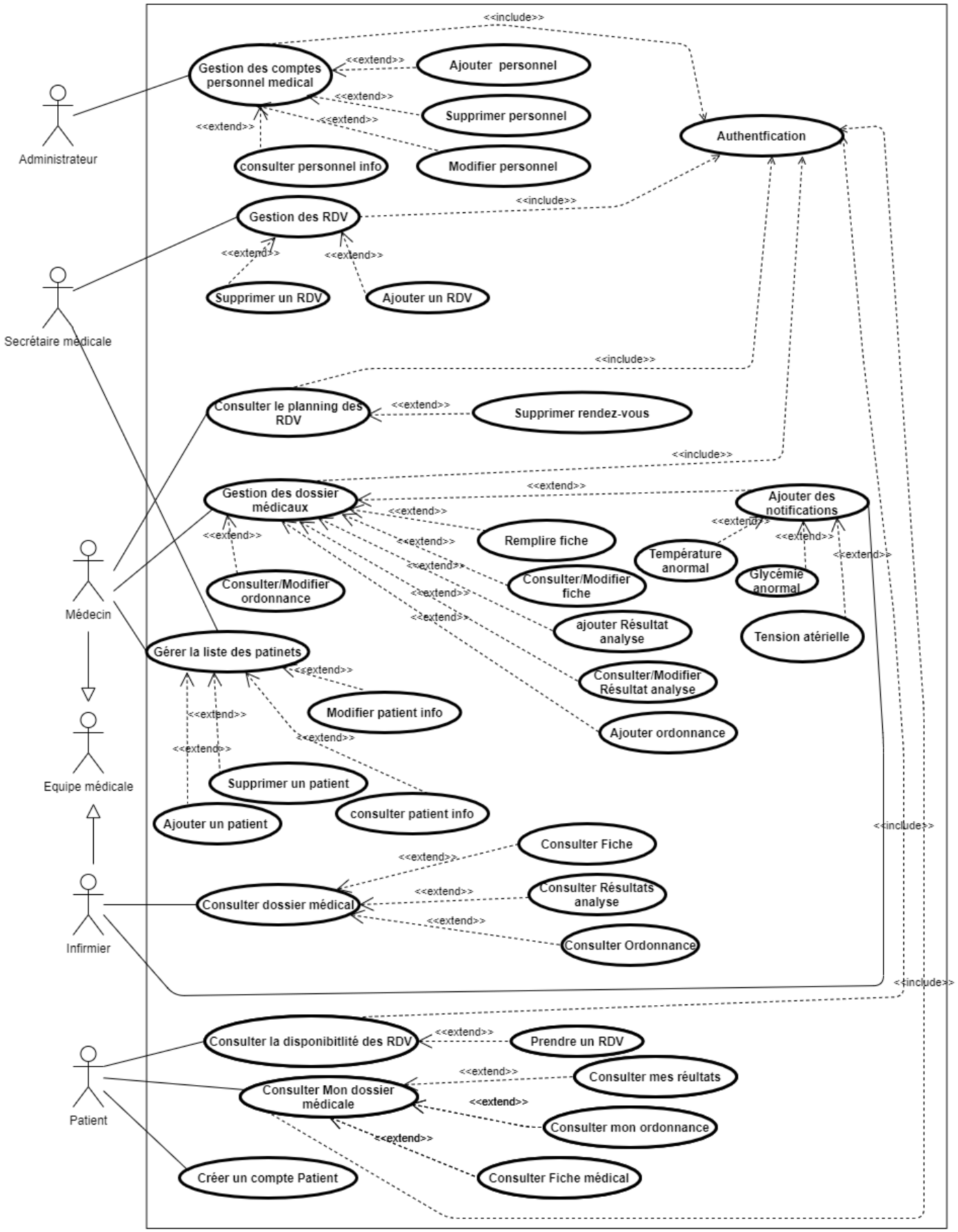


FIGURE 3.23 – Diagramme de cas d'utilisation global de notre application gestion des patient.

✓ ***Diagramme de classe :***

La figure (3.24) suivante présente le diagramme de classe correspondant à notre application gestion des patients dans une clinique privée :

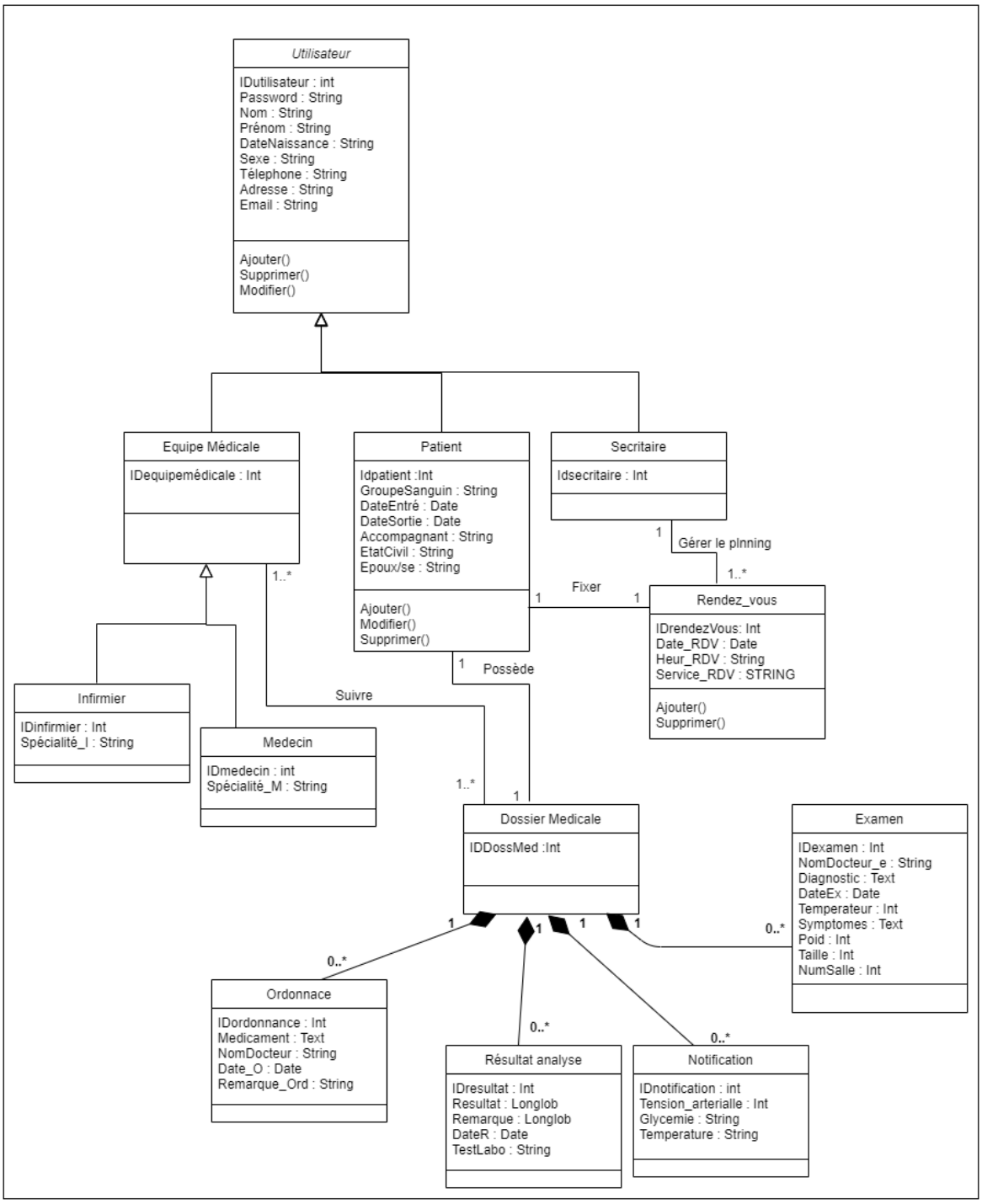


FIGURE 3.24 – Diagramme de classe globale de notre application mobile.

✓ ***Dictionnaire de données :***

Le dictionnaire de données du diagramme de classes illustré ci-dessus est donné par le tableau (3.11) suivant :

L'attribut	Désignation	Type
Utilisateur		
IDUtilisateur	Identifiant d'un utilisateur.	INT
Nom	Nom de l'utilisateur.	VARCHAR
Prenom	Prénom de l'utilisateur.	VARCHAR
DateNaissance	Date de naissance de l'utilisateur.	DATE
Téléphone	Numéro de téléphone de l'utilisateur.	INT
Adresse	Adresse de l'utilisateur.	VARCHAR
Sexe	Sexe de l'utilisateur.	VARCHAR
Email	Email de l'utilisateur.	VARCHAR
Password	Le mot de passe de l'utilisateur.	VARCHAR
Patient		
idpatient	Identifiant d'un patient.	INT
Date_entree	Date de l'entrée du patient.	DATE
Date_sortie	Date de sortie du patient.	DATE
EtatCivil	Situation familiale du patient.	VARCHAR
Epoux	L'époux du patient.	VARCHAR
Accompagnant	Accompagnant du patient.	VARCHAR
Equipe_Medical		
idEquipe_Medical	Identifiant du personnel médical.	INT
Médecin		
idMedecin	Identifiant du médecin.	INT
Spésialité_M	Spécialisation de médecin.	VARCHAR
Infirmier		
idInfirmier	Identifiant de l'infirmier.	INT
Spésialité_I	Spécialisation de l'infirmier.	VARCHAR
Secrétaire		
idSecritaire	Identifiant de l'infirmier.	INT
Rendez-vous		
IDrendezVous	Identifiant de rendez-vous	INT
DateRDV	Date de rendez-vous du patient.	VARCHAR
HeureRDV	Heure rendez-vous du patient.	VARCHAR
ServiceRDV	Service médical spécialisé.	VARCHAR
Dossier_Medical		
id_DosssMed	Identifiant du dossier médical.	INT
Notification		
idNotif	Identifiant de la notification	INT
Temperature	Température du patient.	VARCHAR
Glycemie	Glycémie du patient.	VARCHAR
tension	Tension du patient.	VARCHAR
Resultat_analyse		

idResultat	Identifiant des résultats d'analyses.	INT
TestLabo	Les tests d'analyses effectués.	TEXT
Resultat	Les résultats d'analyses.	LONGLOB
Remarques	Remarques mentionnées par rapport résultats des résultats d'analyses	LONGLOB
Examen		
IDExamen	Identifiant de la fiche médicale.	INT
NomDocteur_E	Nom et prénom de docteur.	VARCHAR
DateEX	La Date du jour.	DATE
Diagnostic	Les diagnostiques du patient.	TEXT
Symptômes	Les Symptômes de patient.	VARCHAR
Température	La Température de patient.	INT
Salle	Le numéro de la salle occupée par le patient.	INT
Poids	Poids de patient.	INT
Taille	Taille de patient.	INT
Ordonnance		
IDordonnance	Identifiant de l'ordonnance.	INT
NomDocteurOrd	Nom et prénom de docteur. _	VARCHAR
Médicament	Médicament préinscrire.	TEXT
RemarqueOrd	Les diagnostiques du patient.	TEXT
DateOrd	La Date du jour d'ordonnance.	VARCHAR

Tableau 3.11 – Dictionnaire de données.

✓ ***Passage au modèle relationnel :***

Dans cette section, nous passons au modèle relationnel à partir de diagramme de classes UML de notre application, afin de pouvoir implémenter notre base de données. La table suivante représente la correspondance entre les concepts du modèle objet et relationnel :

Model d'objet	Model relationnel
Classe	Table
Attribut de type simple	Colonne
Attribut de type composé	Colonne ou clé étrangère
Instance	T_uplet
ID	Clé primaire
Association	Clé étrangère
héritage	Clé primaire identique sur plusieurs tables

FIGURE 3.25 – Équivalence entre objet et relationnel

✓ ***Les règles de passage au modèle relationnel :***

Ce passage de modèle doit respecter un certain nombre de règles qu'on doit les utilise pour faire le passage :

Règle 1 : Transformation des classes : Chaque classe devient une relation l'iden-

tifiant de la classe devient la clé primaire de la relation, les attributs de la classe devient des attributs de la relation.

Règle 2 : Association un-à-un : La clé primaire de l'une des relations doit figurer comme clé étrangère dans l'autre relation.

Règle 3 : Association un-à-plusieurs : La relation de côté plusieurs reçoit comme clé étrangère la clé primaire de la relation du coté 1.

Règle 4 : Association plusieurs-à- plusieurs : L'association devient une relation dont les attributs sont les clés primaires des relations en association, et dont la clé primaire est la concaténation de ces deux attributs, si l'association possède des attributs, ils deviennent des attributs de la relation correspondante.

Règle 5 : Cas d'héritage : Chaque sous classe devient une relation et la clé primaire de la super classe devint clé primaire de chaque sous classe.

Règle 6 : Cas de composition : La clé primaire de la classe composée devient clé étrangère de la classe composant.

Règle 7 : cas d'agrégation, Le même principe que Règle 3.

✓ ***Les tables de la base de données :***

En appliquant cette règle à notre modèle, nous obtenons les relations suivantes :

Patient(idPatient, Password, Nom, Prenom, DateNaissance, Sexe, Télephone, Adresse,Email, GroupeSanguin, DateEntré, DateSortie, Accompagnant, Etat_civile, Epeux).

Secritaire(IDsecritaire, Password, Nom, Prenom, DateNaissance, Genre, Télephone, Adresse, Email).

Medecin(IDmedecin , Password, Nom, Prenom, DateNaissance, Sexe, Télephone, Adresse,Email, Login ,Specialité_M).

Infirmier(IDinfirmier, Password, Nom, Prenom, DateNaissance, Sexe, Télephone, Adresse, Email, Specialité_I).

Rendez_Vous(idRendez_Vous, Date_RDV, Heure_RDV,#idPatient, #IDSecritaire).

Dossier_Medicale(idDossMed, #idPatient_DM).

Suivre_MedecinD(IDDossMed, IDMedecin).

Suivre_infirmierD(IDDossMed, IDinfirmier).

Examen(IDexamen, Nomdocteur, Diagnostic, DateEx, Temperateure, Symptomes, Poids, Taille, NumSalle, #idDossMed).

Ordonnance(IdOrdonnance, Medicament, NomDocteur, Date_O, Remarque_Ord,#idDossMed).

Notification(idNotification , #idDossMed, Tension_arterielle, Glycemie, Température).

Resultat_analyse(IDresultat, Resultat, Remarque, DateR, TestLabo, #idDossMed).

3.6.2.2 Étape 2 : Implémentation

Dans cette étape en tant que équipe de développeurs, nous avons fait la mise en œuvre des taches qui répond à la première étape "conception", et exécuter le plan d'action préciser durant Daily scrum.

En effet, durant cette étape, nous avons utilisé l'environnement de développement Android studio pour réaliser notre application mobile sur la gestion des patients, en utilisant le langage de programmation Java. La figure (3.26) suivante montre une partie du code de l'interface "Edit_patient" :

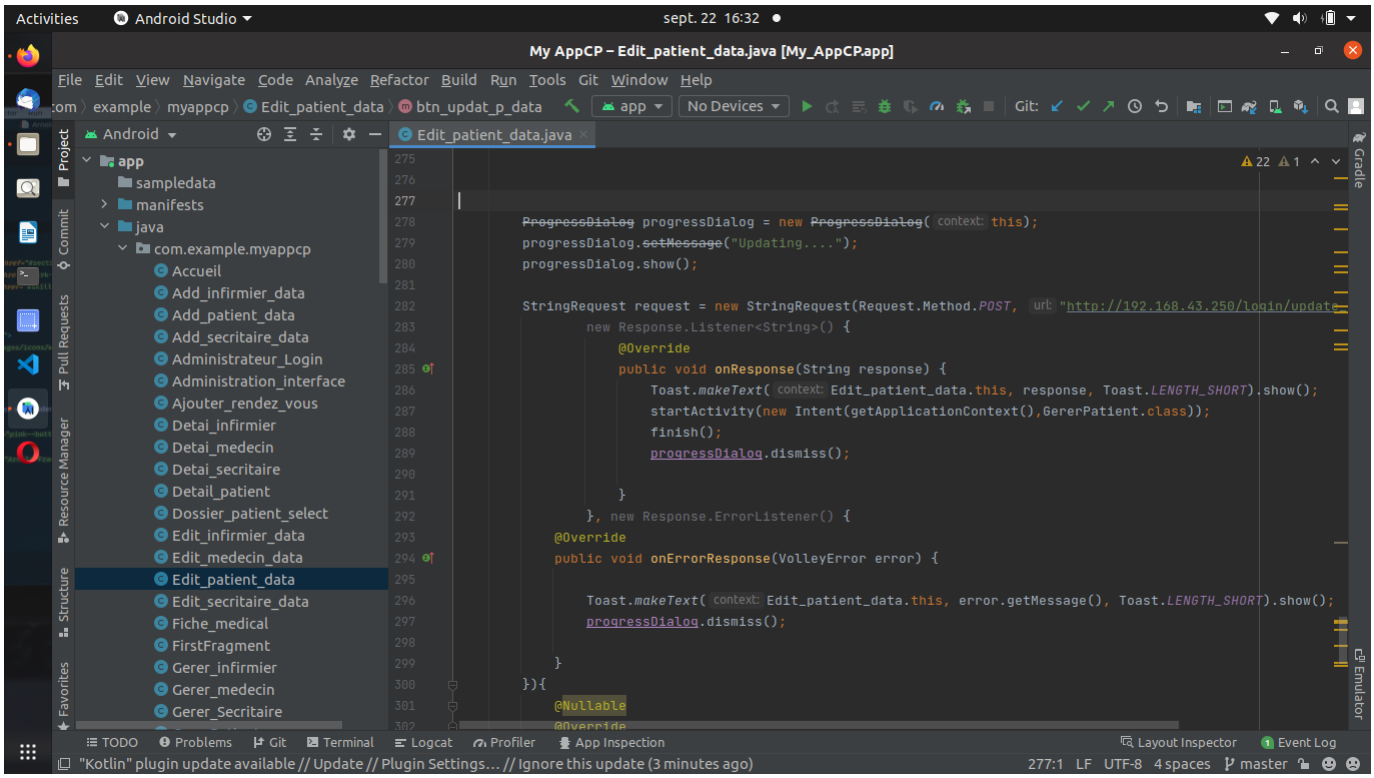


FIGURE 3.26 – Exemple codage sur "Edit_Patient".

Egalement, nous avons utilisé un site de partage de code source pour notre application gestion des patients qui est GitHub (Open Source). Ce dernier, nous permet de consulter le code et de faire des modifications et des améliorations à notre projet.

La figure (3.27) suivante montre la partie code partager sur GitHub :

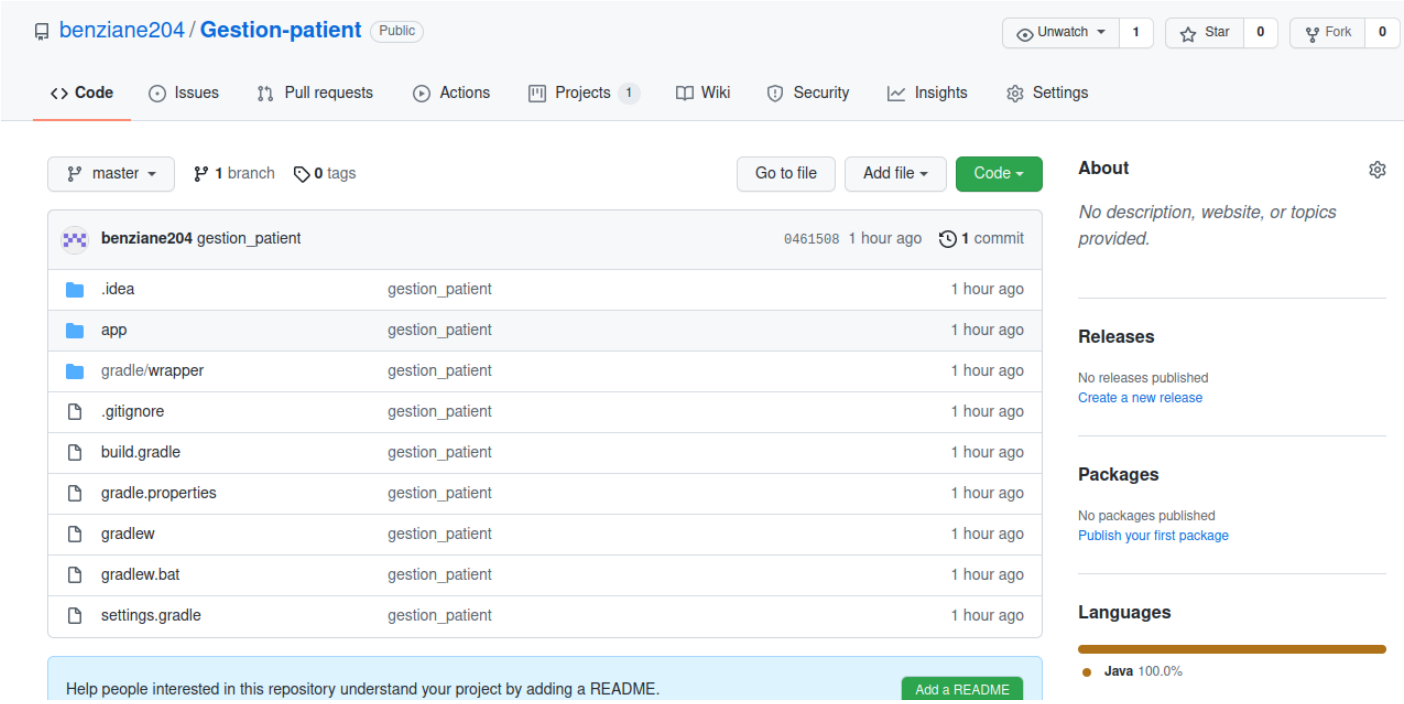


FIGURE 3.27 – La partie code partagé sur GitHub.

3.6.2.3 Étape 3 : Test unitaire

Durant cette étape, nous avons vérifié (les testeurs) le code au niveau de chaque sprint s'ils ne contiennent pas de bugs. Ensuite, nous avons utilisé ces tests automatisés pour effectuer les tests parallèles. Plusieurs logiciels existent sur internet, parmi ces logiciels on peut citer :

Genymotion Cloud SaaS : qui est un émulateur sur cloud qui fournit des machines virtuelles préconfigurée (émulateur payant prix 0.05 $ par minute).

Appium : est un Framework d'automatisation de test open source à utiliser avec des applications Web natives, hybrides et mobiles, OS et Android. Appium permet de générer le code source de test en Java, python, JS, Ruby et Robot Framework.

Intellij IDEA : Environnement de développement intégré qui va contenir le code source des tests en java.

Pour réaliser nos tests automatisés, nous avons utilisé Appium qui est gratuit (Open Source). La figure (3.28) illustre un exemple d'utilisation :

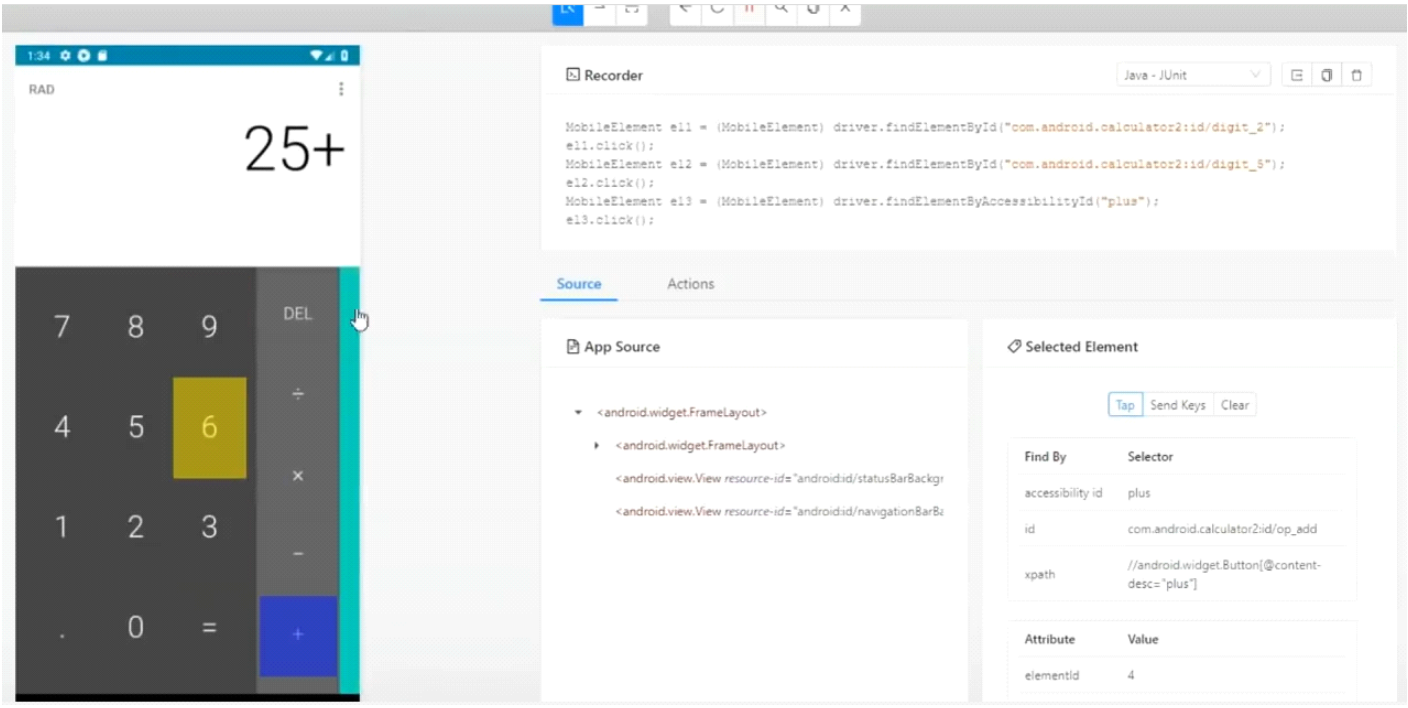


FIGURE 3.28 – Exemple d'utilisation de Appium.

3.6.2.4 Etape 4 : Validation

Dans cette étape, en tant que équipe de développeurs, nous avons vérifié les résultats de la partie codage si elles sont conformes à chaque sprint. De plus, nous avons défini pour chaque sprint sa propre interface sous forme d'un guide utilisateur.

Dans ce qui suit, nous allons montrer les différentes interfaces de chaque sprint.

3.6.2.4.1 Sprint 1

(a) Interface d'accueil :

La figure (3.29) représente la première interface après le lancement de l'application, L'utilisateur doit préciser son identité soit en tant que Patient, médecin, infirmier, administrateur, ou secrétaire, puis clique sur bouton valider :

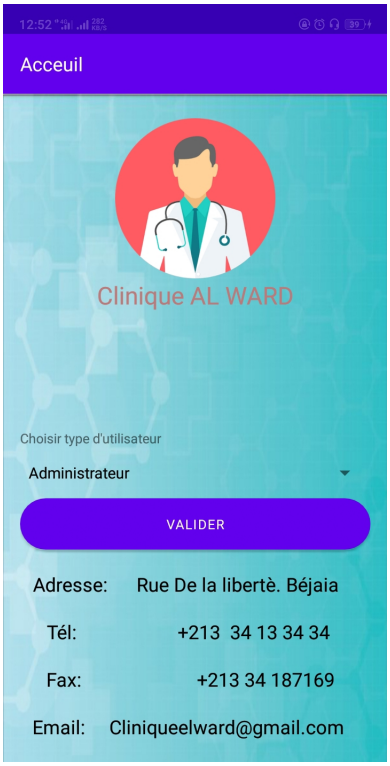


FIGURE 3.29 – Interface d'accueil.

(b) Interface de connexion :

La figure (3.30) représente l'interface de connexion :

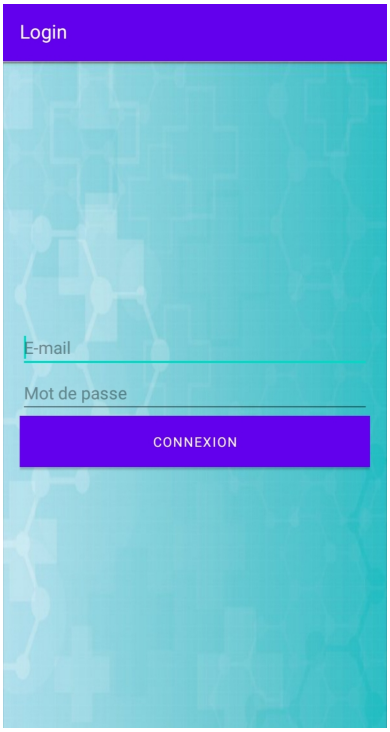
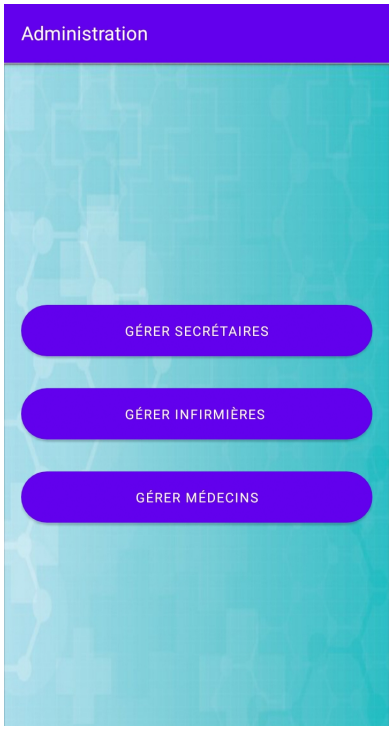


FIGURE 3.30 – Interface de connexion.

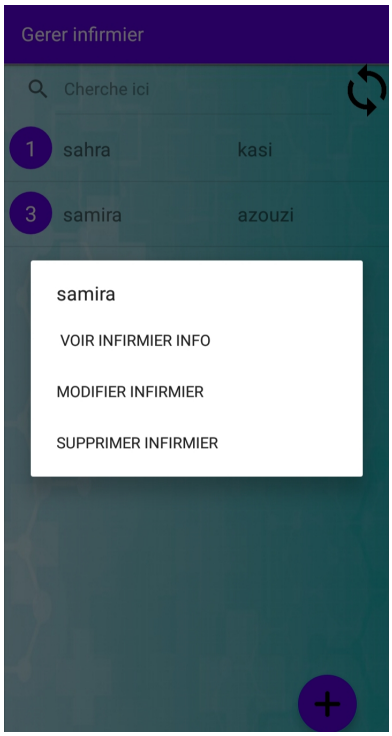
• L'utilisateur doit saisir son émail et mot de passe pour s'authentifier et accéder à l'application.

3.6.2.4.2 Sprint 2

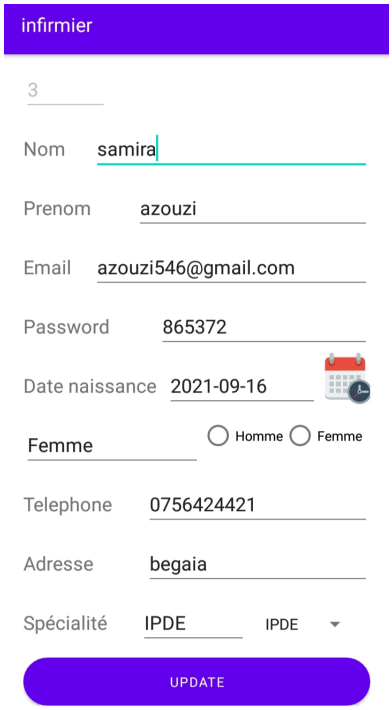
L'acteur principal de ce sprint est l'administrateur qui fait la gestion des comptes utilisateurs (secrétaire, infirmier et médecin). La figure (3.31) ci-dessous, présente l'interface de gestion des comptes utilisateurs :



(a) Interface administration.



(b) Interface gérer infirmier



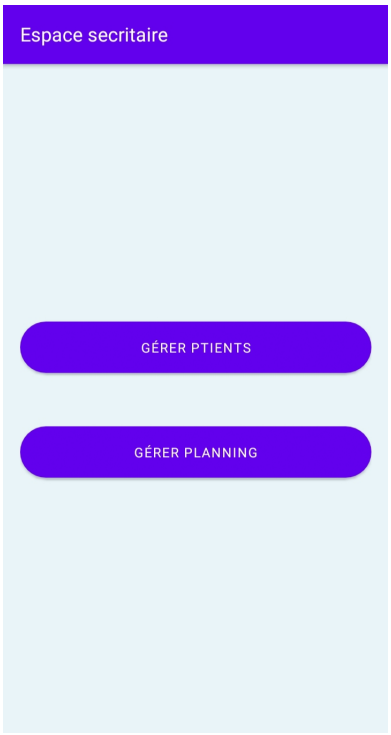
(c) Interface modifier infirmier.

FIGURE 3.31 – Espace administration.

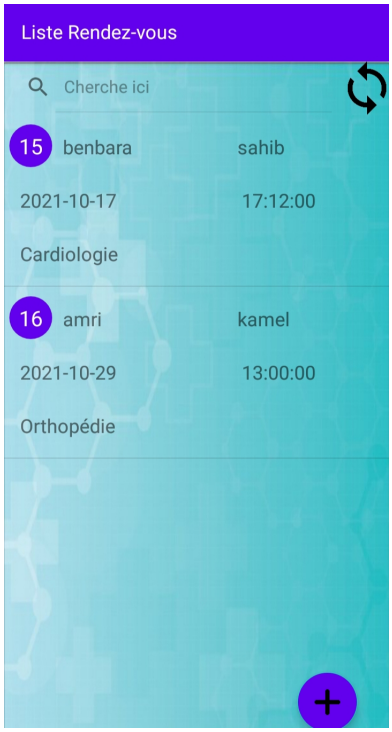
3.6.2.4.3 Sprint 3

L'acteur principal de ce sprint est le secrétaire médical qui fait la gestion des patients et la planification des rendez-vous.

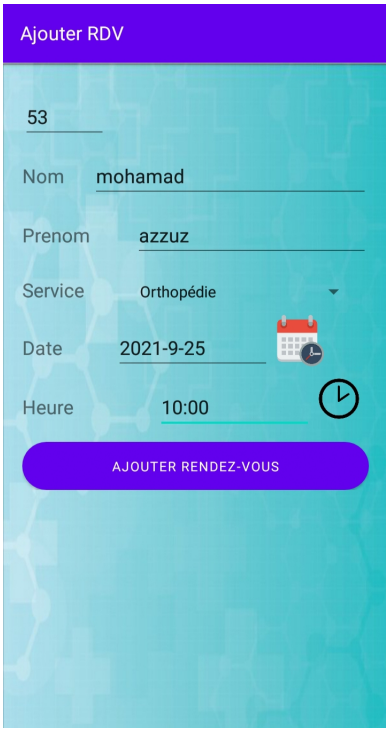
les figures (3.32) présentent les interfaces suivantes : Interface Espace secrétaire, interface liste des rendez-vous et interface ajouter rendez-vous :



(a) Interface sécuritaire.



(b) Interface liste des rendez-vous.



(c) Interface ajouter rendez-vous.

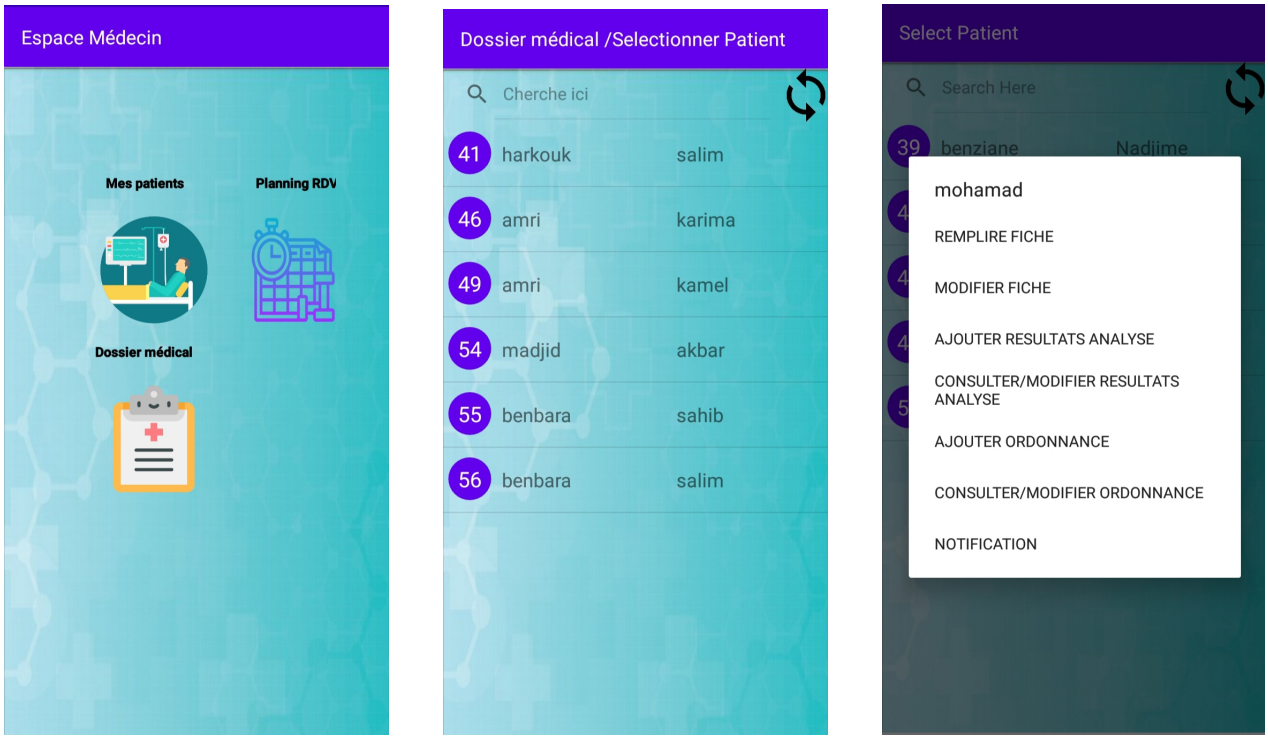
FIGURE 3.32 – Espace sécuritaire.

3.6.2.4.4 Sprint 4

(a) Interface dédié aux médecins :

• Après l'authentification du médecin de service, un menu (a) apparaît (Figure 3.33).

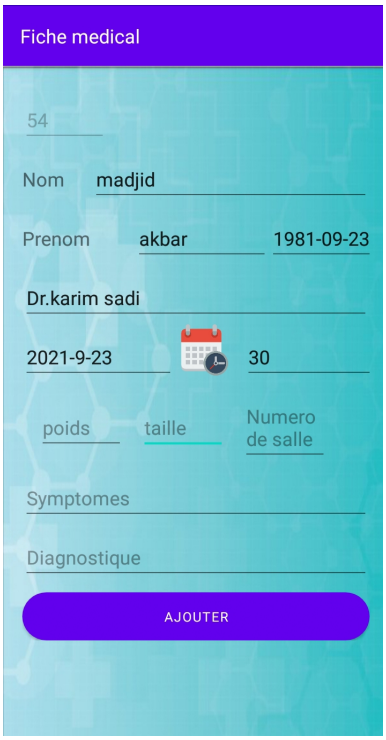
• En clinquants sur "Dossier médical", une liste des patients s'affiche, le médecin recherche et clique sur le patient, puis une liste (c) d'option s'affiche (Figure 3.33) :



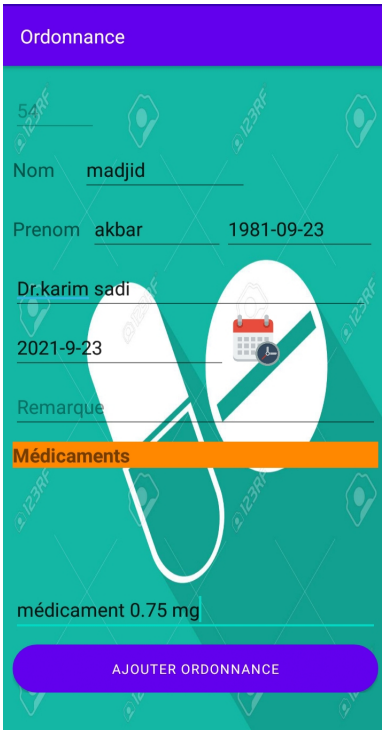
(a) Interface espace médecin. (b) Interface liste des patients. (c) Interface dossier médical.

FIGURE 3.33 – Espace médecin.

• Le médecin clique sur "remplir fiche" pour ajouter une fiche, ou clique sur "ajouter ordonnance" s'il souhaite prescrire des médicaments, l'interface de "ordonnance" se charge (Figure 3.34) :



(a) Interface ajouter fiche



(b) Interface ajouter ordonnance.

FIGURE 3.34 – Interface ajoute fiche et ordonnance.

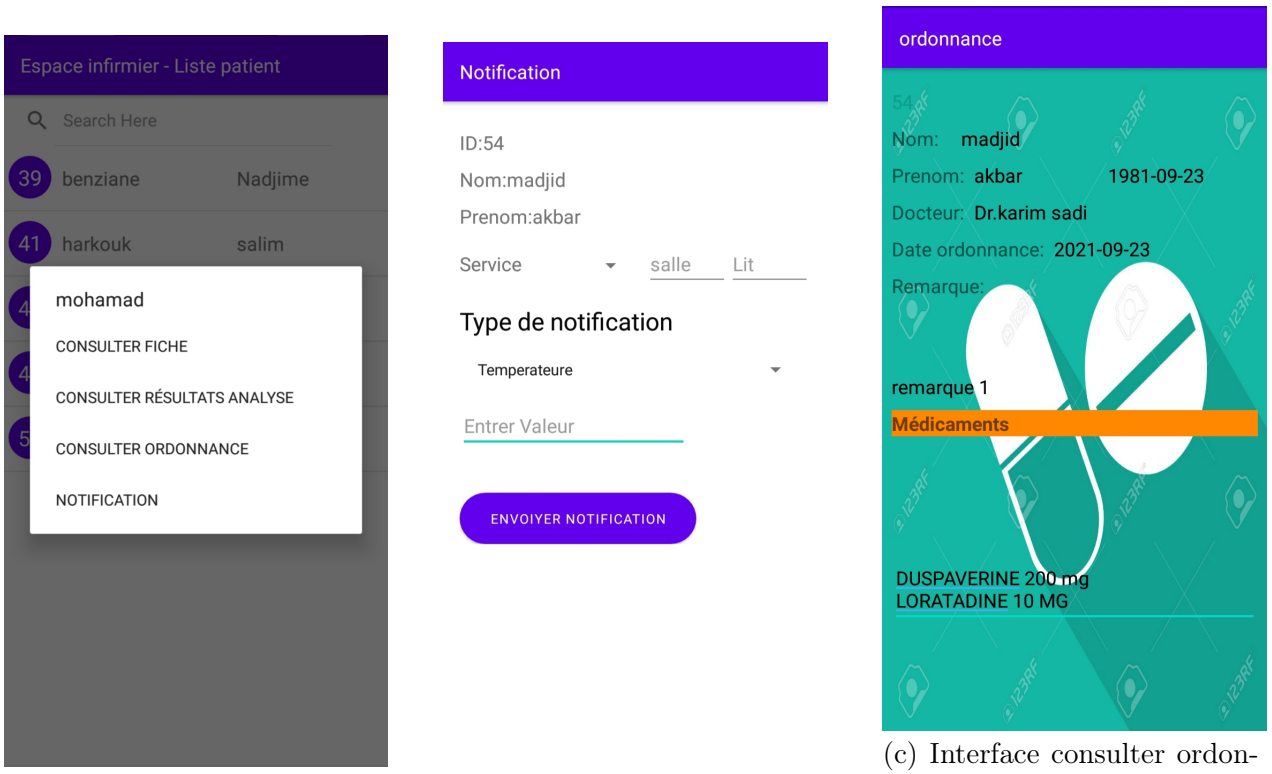
(b) Interface dédié aux infirmiers :

Les infirmiers de la clinique eux aussi peuvent contribuer au suivi des patients, ils ont un espace dans notre application :

• Ils peuvent consulter les dossiers médicaux des patients sans avoir le droit à la modification (Figure 3.35).

• Après la sélection de patient une liste d'option s'affiche, En clinquants sur "Consulter fiche" une liste des fiches médicaux correspondant au patient sélectionné s'affiche, puis l'infirmier choisissait l'une de ces fiches pour la consulter.

• En clinquants sur "Notification", l'infirmier peut alerter le médecin du service en choisissant le type de la valeur à ajouter (Température, Glycémie, tension) (Figure 3.35).



(a) Interface espace infirmier. (b) Interface notification. (c) Interface consulter ordonnance.

FIGURE 3.35 – Espace infirmier.

• Le médecin du service va recevoir une notification avec le numéro de la salle et du lit du patient, la notification est de type (Température, Glycémie, tension), Figure (3.36).

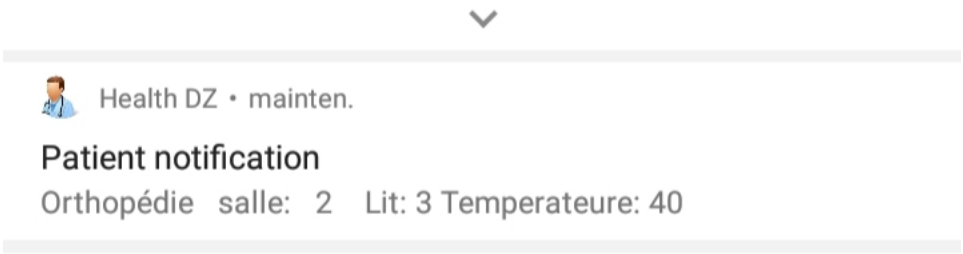
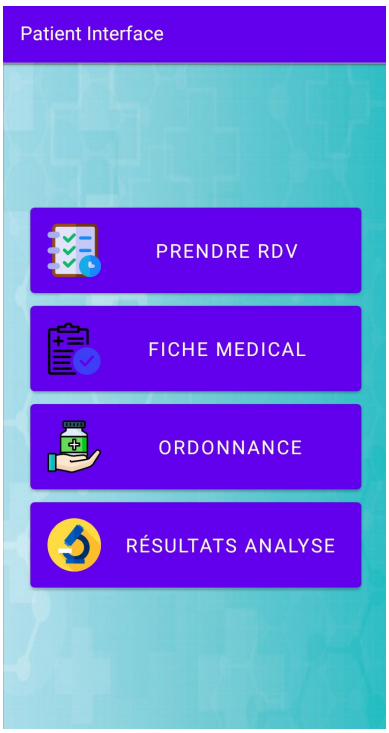


FIGURE 3.36 – Notification pour le médecin de service

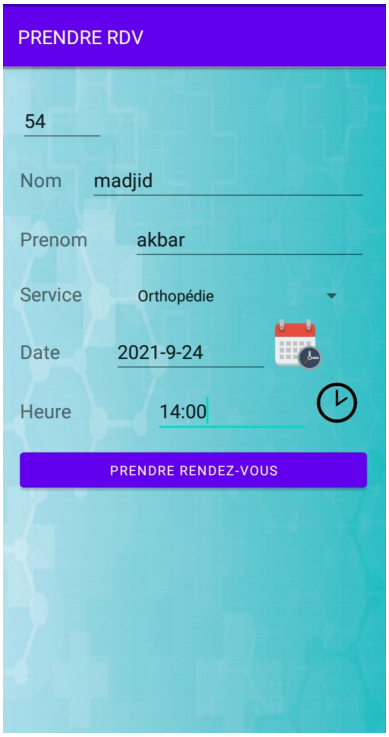
3.6.2.4.5 Sprint 5

L'acteur principal de ce sprint est le patient qui consulte son dossier médical et pourra prendre un rendez-vous (Figure 3.37).

• En clinquant sur "prendre rendez-vous" une interface prendre rendez-vous s'affiche là où le patient choisi le service médical, la date et l'heure de rendez-vous.



(a) Interface espace patient.



(b) Interface prendre rendez-vous.

FIGURE 3.37 – Espace patient.

• Le patient est capable de consulter tous ses fiches, ordonnances et résultats d'analyses en cliquant sur "fiche médical", "ordonnance" ou "résultat analyse".

3.6.3 Phase 3 : Post- phase

C'est la dernière phase de notre framework, après que les développeurs ont validé leurs correctifs, puis fusionnant les modifications.

Cette phase se constitue de deux étapes importantes à savoir : l'étape des tests d'intégration, et l'étape des tests d'acceptation utilisateurs.

3.6.3.1 Étape 1 :Test d'intégration

Durant cette étape, nous avons validé l'interaction entre les différentes parties de notre code et prennent généralement un certain temps. Cette étape est similaire à l'étape de test unitaire, où nous avons utilisé les mêmes outils de test. Elle permet de vérifier le bon assemblage de toutes les parties du code.

La figure (3.38) montre un exemple de test d'intégration pour la fonction d'authentification :

```
public class TestMyApp_Integration
{

    public void TestReconnu()
    {

        string email = "Email@test.com";
        EmailController emailController = new
            EmailController(new EmailService());
        var result = emailController.VerifyEmail(email);
        Assert.IsInstanceOfType(result,
            typeof(RedirectToRouteResult));
        Assert.AreEqual("ProceedRegistration",
            ((RedirectToRouteResult)result).RouteValues["action"]);
    }


    public void TestNonReconnu()
    {
        string email = "NonreconnuEmail@test.com";
        EmailController emailController = new
            EmailController(new EmailService());
        var result = emailController.VerifyEmail(email);
        Assert.IsInstanceOfType(result,
            typeof(ContentResult));
        Assert.AreEqual("Adresse e-Mail dj  utilise !",
            ((ContentResult)result).Content);
    }
}
```

FIGURE 3.38 – Exemple test d'intégration effectué pour "Authentification".

3.6.3.2 Étape 2 :Test d'acceptation utilisateur

C'est l'étape finale de processus de développent de notre application mobile, les testeurs ont pour objectif de tester l'application avec des utilisateurs.

A cette étape, nous considérons l'équipe médicale (médecins, infirmiers) de la clinique privée comme étant une équipe experte (symbolisé par **X**) et les patients comme étant une équipe non expérimenté (symbolisé par **P**).

• Le nombre de testeurs de l'équipe experte est : 03 testeurs (02 médecin, 1 infirmier).

• le nombre de testeurs de l'équipe non experte est : 02 testeurs.

Ces deux équipes ont tester l'application pour savoir s'ils sont capables d'effectuer, en situation réelle, les différentes tâches pour la qu'elle a été conçue tout en se basant sur un questionnaire qui l'aura été donnés (selon un niveau de 1 à 6). le tableau (3.12) suivant représente la moyenne des résultats des tests effectuées :

Début de table

Question	**Pas du tout d'accord**			**Tout à fait d'accord**		
	1	2	3	4	5	6
1.Je pense que je vais utiliser cette application fréquemment.					**X / P**	
2.Je trouve cette application inutilement complexe.	**X / P**					
3.Je pense que cette application est facile à utiliser.						**X / P**

4.Je pense que j'aurai besoin de l'aide d'un technicien pour être capable d'utiliser cette application.	**X / P**					
5.J'ai trouvé que les différentes fonctions de cette application ont été bien intégrés.				**X**	**P**	
6.Je pense qu'il y a trop d'incohérence dans cette application.	**X / P**					
7.J'imagine que la plupart des gens seront capable d'apprendre à utiliser cette application très rapidement.						**X / P**
8.J'ai trouvé cette application très lourd à utiliser.		**X / P**				
9.Je me sentais très en confiance en utilisant cette application .						**X / P**
10.J'ai besoin d'apprendre beaucoup de choses avant de pouvoir utiliser cette application.						**X / P**

Tableau 3.12 – Formulaire de test d'acceptation.

fin de table

3.7 Conclusion

Dans ce dernier chapitre sur la validation de notre Framework proposé en chapitre 2, nous avons présenté l'environnement de développement et les différents outils que nous avons utilisé, ainsi que les bibliothèques et les librairies qui nous ont aidées à la réalisation de notre application mobile. Nous avons par la suite décrit le processus de réalisation de notre application mobile en se basant sur la combinaison de la méthode agile Scrum avec le cloud computing on respectant la méthodologie d'application de la méthode scrum (pré-phase, phase de développement, post-phase) , et les services de cloud associés à chaque phases, vers la fin nous avons terminé par une conclusion

Conclusion générale

Les méthodes agiles sont devenues les plus utilisées dans le développement logiciel, car elles favorisent la livraison continue de logiciels fonctionnels répondant aux besoins des clients. Malgré ses avantages, elles connaissent beaucoup de challenges et de limitations. Pour y remédier, plusieurs entreprises de développement choisissent de migrer leurs activités dans le cloud qui connaît ces dernières années une croissance rapide. Cependant, pour bénéficier des avantages de l'utilisation des ressources du cloud, il est nécessaire d'évaluer l'applicabilité de ces méthodes agiles dans le cloud.

Nous avons proposé un cadre méthodologique basé sur la méthode Scrum et adaptée au développement agile d'applications dans un environnement cloud. Le cadre proposé définit les différentes phases de la méthode Scrum en montrant l'implication des services du cloud dans le processus de développement.

Nous avons également commencer par la phase de planification qui influence sur la réussite des sprints de la phase développement, ensuite, entamer la phase clé est celle de la « phase développement » qui décrit les activités nécessaires pour le choix du développement d'applications dans le cloud, finir par la post phase ou les équipes expérimentés et les équipes métiers valide l'application réalisé selon les demandes originales de client.

il faut noter que pour ce travail, nous nous sommes limités au développement d'applications dans le cloud avec des fournisseurs de cloud open source. Ces fournisseurs ont été abordées en détail dans le dernier chapitre. Il faut noter également que la méthode Scrum est l'une des méthodes les plus utilisées actuellement dans le cloud pour le développement et l'exploitation des applications. Il est donc important que d'autres travaux de recherche se penchent sur le développement agile dans l'ensemble des 3 couches du cloud afin d'établir une grille de choix des services du cloud nécessaires pour le développement d'applications agile en fonction des besoins et de leur complexité.

Nous suggérons donc que d'autres recherches soient faites sur la standardisation des environnements, outils, services et processus de développement d'applications plus performant dans les trois couches PaaS, SaaS, IaaS, du cloud computing.

Pour finir, il est important aussi d'expérimenter cette méthode Scrum en cas réel de projet de développement afin d'identifier ses potentielles limites et l'améliorer.

Bibliographie

[1] https://developer.android.com. consulté le : 28//08/2021.

[2] cours sur le langage de programmation java. https://koor.fr/Java/Tutorial/Index.wp. consulté le : 28/08/2021.

[3] Github (gratuit) : la plateforme de développement microsoft au crible. https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443812-github-gratuit-la-plateforme-de-developpement-microsoft-au-crible/. Consulté le 01/08/2021.

[4] Jira (gratuit) : de la gestion de projet à la digital workplace. https://www.journaldunet.fr/web-tech/guide-de-l-entreprise-digitale/1443890-jira-de-la-gestion-de-projet-agile-digital-workplace/. Consulté le 01/08/2021.

[5] https://romotech.ca/prive-et-hybride/, 2020. consulté le : 25/05/2021.

[6] Mouhib Alnoukari. Asd-bi : A knowledge discovery process modeling based on adaptive software development agile methodology. In *Business Intelligence and Agile Methodologies for Knowledge-Based Organizations : Cross-Disciplinary Applications*, pages 183–207. IGI Global, 2012.

[7] Claude Aubry. *Scrum-3e éd. : Le guide pratique de la méthode agile la plus populaire*. Dunod, 2013.

[8] Avinash Bandaru. Amazon web services.

[9] Olivier Cadet. Webwag mobile blog, introduction à volley et gson. https://blog.webwag.com/2017/02/14/introduction-a-volley-gson/. consulté le 28/08/2021.

[10] Olivier Carton. L'essentiel de xml : Cours xml m2 pro à l'université paris diderot. https://www.irif.fr/~carton/Enseignement/XML/Cours/support.pdf. consulté le : 28//08/2021.

[11] Mike Cohn. User stories applied : For agile software development. *Addison-Wesley Professional*, 2004.

[12] International Standards Organization/International Electrotechnical Commission et al. Information technology–cloud computing–overview and vocabulary. *International Standard*, 17788, 2014.

[13] International Standards Organization/International Electrotechnical Commission et al. Information technology–cloud computing–reference architecture. *International Standard*, 17789, 2014.

[14] Goyal Ruchi Dhanotia, Shruti. Rapid application developmen.

[15] Olivier Glasseyr and Jean-Loup Chappelet. Comparaison de trois techniques de modélisation de processus : Adonis, ossad et uml. Technical report, IDHEAP, 2002.

[16] Nicolas GREVET. Le cloud computing : Evolution ou révolution. http://www.nicolasgrevet.com/files/mr09_ngrevet_cloudcom.pdf. consulté le : 25/05/2021.

[17] Chantal Gribaumont. Cours :administrez vos bases de données avec mysql. https://zestedesavoir.com/tutoriels/pdf/730/administrez-vos-bases-de-donnees-avec-mysql.pdf.

[18] Lamia Ben Hiba and Mohammed Abdou Janati Idrissi. Tendances des méthodes de gestion des projets informatiques. *Revu*, page 7, 2012.

[19] David Hilley. Cloud computing : A taxonomy of platform and infrastructure-level offerings. Technical report, Georgia Institute of Technology, 2009.

[20] Anne Sophie Boisard Jean Francois Pépin, Sophie Bouteiller et al. Fondamentaux du cloud computing : le point de vue des grandes entreprises. 2013.

[21] Dusanka Boskovic Kalem, Dzenana Donko. Agile methods for cloud computing. In *2013 36th International Convention on Information and Communication Technology, Electronics and Microelectronics (MIPRO)*, pages 1079–1083. IEEE, 2013.

[22] Roloff Eduardo Maillard Nicolas Magalhães, Guilherme. Developing on google app engine.

[23] Peter Mell, Tim Grance, et al. The nist definition of cloud computing. 2011.

[24] Peter Mell, Tim Grance, et al. Sp 800-145. the nist definition of cloud computing. Technical report, Gaithersburg, MD, United States, 2011.

[25] Fidel Navamuel. L'actualité des outils numériques pour l'éducation. https://outilstice.com/2018/04/draw-io-outil-gratuit-pour-dessiner-des-diagrammes-en-ligne/, 2018. consulté le 28/08/2021.

[26] Pavan Raheja. https://www.ques10.com/p/30825/enlist-and-explain-various-service-model-and-deplo/. consulté le : 25/05/2021.

[27] Gaurav Raj, Komal Yadav, and Arunima Jaiswal. Emphasis on testing assimilation using cloud computing for improvised agile scrum framework. *In 2015 International Conference on Futuristic Trends on Computational Analysis and Knowledge Management (ABLAZE)*, pages 219–225, February,2015.

[28] Sébastien ROHAUT. Cours php.

[29] Pascal Roques. Uml 2 par la pratique. editions eyrolles. 8, 2018.

[30] Ken Schwaber. Agile project management with scrum. *Microsoft press*, 2004.

[31] Everett Toews, Barton Satchwill, Robert Rankin, John Shillington, and Todd King. An internationally distributed cloud for science : The cloud-enabled space weather platform. *In Proceedings of the 2nd International Workshop on Software Engineering for Cloud Computing*, pages 1–7, 2011.

[32] Richard Tremblay. Implantation d'une méthode agile de développement logiciel en entreprise : une culture accueillant le changement. 2007.

[33] Yannick Kuhn Allan Lefort Vincent Kherbache, Mohamed Moussalih. Cloud computing. 2010.

[34] Kevin Vlaanderen, Slinger Jansen, Sjaak Brinkkemper, and Erik Jaspers. the agile requirements refinery : Applying scrum principles to software product management. *Addison-Wesley Professional*, 53(1) :58–70, 2011.

[35] Yves Wautelet, Laurent Louvigny, and Manuel Kolp. Le unified process comme méthodologie de gestion de projet informatique. eléments d'application en milieu sidérurgique. Technical report, Working Paper IAG 109/04, Université Catholique de Louvain, 2004.

[36] Khusbhu Sahendrasingh Yadav and Maleeha Arif Yasvi. Review on extreme programming-xp. In *International Conference on Robotics, Smart Technology and Electronics*, 2019.

[37] Muhammad Younas, Dayang NA Jawawi, Israr Ghani, Muhammad Irfan Khan, and Imran Ghani. A survey of agile development methods and tools in cloud environment.

[38] Muhammad Younas, Dayang Norhayati Abang Jawawi, Ahmad Kamil Mahmood, Mohammad Nazir Ahmad, Muhammad Umer Sarwar, and Mohd Yazid Idris. Agile software development using cloud computing : A case study. *IEEE Access*, 8 :4475–4484, 2019.

[39] Chung Yung and Yutang Lin. Implementing toast, a tool for agile software project management in cloud computing environments. *JSW*, 10(11) :1310–1318, 2015.

Résumé

Depuis ces dernières années, le secteur de l'ingénierie logicielle a connu de grands changements en raison de l'adoption de méthodes agiles. Elles favorisent le développement itératif et incrémental des applications pour la satisfaction des clients et utilisateurs. Malgré ces avancées, ces méthodes font face à plusieurs challenges parmi lesquels, le manque d'environnements adéquats capables de faciliter les activités de développement et d'assurer la mise en marché rapide des applications. Pour faire face à ce challenge, le cloud computing constitue une solution car, il permet de disposer d'un ensemble de ressources, de services et d'outils rapidement et facilement configurables à moindre coût.

Il est donc question dans ce travail d'identifier comment développer de façon agile, les applications dans le cloud. À cet effet, nous avons proposé un cadre méthodologique qui s'appuie sur la méthode agile SCRUM adaptée aux contextes de développement dans un environnement cloud. Le cadre proposé suit un cycle de développement agile permettant ainsi l'identification des besoins, la planification des itérations ainsi qu'une livraison plus rapide et efficace de notre application.

En vue de démontrer la validité du notre travail, nous avons appliqué le cadre méthodologique proposé sur une étude de cas dans le domaine de la gestion des patients dans une clinique privée.

Mots clés : Cloud Computing, méthodes agiles, SCRUM, gestion des patients.

Abstract

In recent years, the software engineering sector has experienced great changes due to the adoption of agile methods. They promote development iterative and incremental applications for the satisfaction of customers and users. Despite these advances, these methods face several challenges, among which, the lack of suitable environments capable of facilitating development activities and ensuring the rapid marketing of applications. To face this challenge, cloud computing is a solution because it provides a set of resources, services and tools that can be quickly and easily configured at a lower cost.

This work therefore involves identifying how to develop in an agile way, cloud applications. To this end, we have proposed a methodological framework based on the agile SCRUM method adapted to development contexts in a cloud environment. The proposed framework follows an agile development cycle allowing the identification of needs, the planning of iterations as well as a faster and more efficient delivery of our application.

In order to demonstrate the validity of our work, we applied the proposed methodological framework to a case study in the field of patient management in a private clinic.

Keywords : Cloud Computing, agile methods, SCRUM, patients management.