RÉPUBLIQUE ALGÉRIENNE DÉMOCRATIQUE POPULAIRE
MINISTÈRE DE L'ENSEIGNEMENT SUPÉRIEUR ET DE LA RECHERCHE SCIENTIFIQUE
Université Yahia Farès Médéa
Faculté des Sciences
Département de mathématique et informatique



Mémoire de fin d'étude

Présenté pour l'obtention du Diplôme de Master en Informatique.
Option : Ingénierie des Systèmes et Technologies du Web.

Conception et réalisation d'une application web pour la gestion des malades Covid-19

Proposé par :
M. Abdoun Mohamed Nabil
Présenté par :
Mr Zouaouine Zineddine et Mr Berriahi Ismail

2020/2021

Remerciement

Il y a quelques années, après avoir choisi de terminer nos études dans cette discipline, nous ne pouvons pas deviner que notre projet de fin d'étude serait l'aboutissement d'un long chemin, une réalisation personnelle et une expérience humaine. C'est très agréable de comprendre, aujourd'hui, qu'une PFE est une réalisation collective, car sur ce chemin plusieurs personnes nous ont aidés et nous ont suivis. Sans ces personnes, nous ne pourrions pas le faire, et c'est à eux que nous voulons exprimer ici nos sincères remerciements.

Tout d'abord, grâce au Dieu tout puissant qui a nous avons donné la force et la volonté que nous avons achevé ce modeste travail.

Puis, nous tenons à exprimer toute notre gratitude à notre encadreur Monsieur *Abdoun Nabil* de nous avoir proposées ce sujet de fin d'étude, et pour nous avoir suivis, guidés, conseillés et aidés durant toute la période du travail. Nous remercions sincèrement les professeurs qui nous ont enseignée durant notre parcours universitaire à l'université « YAHYA FARES » Médéa.

Un grand merci à nos familles pour leur indéfectible soutien, leur patience et leur complaisance. Ce travail vous est dédicacé. Nous avons encore un merci particulier à tous nos amis pour leur prévenance et leurs encouragements.

Toutes les lettres ne sauraient trouver les mots qu'il faut, tous ces mots ne sauraient exprimer la gratitude, l'amour, le respect, et la reconnaissance à nos très chers parents qui nous ont offert sans condition leur soutien morale et financier et à qui je dois ce travail.

A notre encadreur monsieur *Abdoun Nabil* pour son soutien, sa compréhension, sa patience ces encouragements et ces aides. Et à toutes ces personnes spéciales qui ont, durant toutes ces années, été nos compagnons de tous les jours et une deuxième famille pour nous, ceux-là qui n'ont cessé de partager avec moi leur sympathie, leur chaleur, leur estime et leur indulgence.

Sommaire

1 Généralités sur le web **2**
- 1.1 L'architecture client/serveur . . . 2
- 1.2 Notion de base . . . 3
  - 1.2.1 Serveur . . . 3
  - 1.2.2 Client . . . 3
  - 1.2.3 Une requête . . . 3
  - 1.2.4 Réponse . . . 3
- 1.3 Les différentes architectures client-serveur . . . 3
  - 1.3.1 Architecture 1-tiers . . . 3
  - 1.3.2 Architecture 2-tiers . . . 3
  - 1.3.3 Architecture 3-tiers . . . 4
  - 1.3.4 Architecture n-tiers . . . 4
- 1.4 Les caractéristiques du client/serveur . . . 4
- 1.5 Présentation du web . . . 4
  - 1.5.1 L'internet . . . 4
  - 1.5.2 Le web . . . 5
- 1.6 Standards du web . . . 5
- 1.7 Type des pages web . . . 5
  - 1.7.1 Les pages web dynamiques . . . 5
  - 1.7.2 Les pages web statiques . . . 6
- 1.8 Site web . . . 6
  - 1.8.1 Site web dynamique . . . 6
  - 1.8.2 Site statique . . . 7
- 1.9 Les services web . . . 7
  - 1.9.1 Historique . . . 8
  - 1.9.2 Objectifs . . . 8
  - 1.9.3 Les standards . . . 9
- 1.10 Hébergement . . . 11
  - 1.10.1 Les hébergements sur un serveur dédié . . . 11
  - 1.10.2 Les hébergements VPS (Serveur Privé Virtuel) . . . 12
  - 1.10.3 Les hébergements cloud . . . 12
- 1.11 UML . . . 12

2 Analyse et conception **15**
- 2.1 Quelques définitions . . . 16
  - 2.1.1 Définition du langage UML . . . 16
- 2.2 Pourquoi utiliser UML ? . . . 16

2.3 Etude de l'existant . . . 16
2.3.1 Présentation de l'organisme d'accueil . . . 16
2.3.2 Description du projet . . . 17
2.3.3 Problématique . . . 17
2.3.4 Objectifs . . . 17
2.4 La spécification du système et analyse des besoins . . . 18
2.4.1 Besoins du chef de service . . . 18
2.4.2 Besoins du médecin généraliste . . . 18
2.4.3 Besoin du pneumologue et l'infectiologue . . . 18
2.4.4 Besoin du médecin spécialiste Covid-19 . . . 18
2.5 Les Règles de gestion . . . 18
2.6 Identification des Acteurs . . . 19
2.6.1 Chef service covid-19 . . . 19
2.6.2 Médecin généraliste . . . 19
2.6.3 Médecin pneumologue et l'infectiologue . . . 19
2.6.4 Médecin spécialiste covid-19 . . . 19
2.6.5 Chef service covid-19 . . . 19
2.8 Diagramme de cas d'utilisation . . . 21
2.8.1 Diagramme de cas d'utilisation "chef de service (administrateur) global " . . . 21
2.8.2 Diagramme de cas d'utilisation 'chef de service (gestion des dossiers médicaux)' 22
2.8.3 Diagramme de cas d'utilisation 'chef de service (gestion des médecins)' . . . 23
2.8.4 Diagramme de cas d'utilisation 'chef de service (gestion des infirmiers)' . . . 23
2.8.5 Diagramme de cas d'utilisation 'chef de service (gestion des comptes)' . . . 24
2.8.6 Diagramme de cas d'utilisation 'Médecin généraliste (Global)' . . . 24
2.8.7 Diagramme de cas d'utilisation 'médecin généraliste (mise à jour de la liste des malades)' . . . 25
2.8.8 Diagramme de cas d'utilisation 'médecin généraliste (gestion des dossiers médicaux)' . . . 25
2.8.9 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (global)' . . . 26
2.8.10 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (gestion d'hospitalisatio)' . . . 27
2.8.11 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (gestion des dossiers médicaux)' . . . 27
2.8.12 Diagramme de cas d'utilisation 'Médecin spécialiste' . . . 28
2.8.13 Diagramme de cas d'utilisation 'médecin spécialiste (consultation des dossiers médicaux)' . . . 28
2.8.14 Diagramme de cas d'utilisation 'Médecin spécialiste (gestion de sortie)' . . . 29
2.8.15 Diagramme de cas d'utilisation 'infirmier' . . . 29
2.9 Description de la vue dynamique (Diagramme de séquence) . . . 30
2.9.1 Diagramme de séquence "Authentification" . . . 30
2.9.2 Diagramme de séquence " Gestion du dossier médical par un chef de service" . . . 31
2.9.3 Diagramme de séquence "Gestion du dossier médical par un médecin généraliste" 33
2.9.4 Diagramme de séquence "Gestion du dossier médical par un infirmier" . . . 34
2.9.5 Diagramme de séquence "Gestion du dossier médical par un spécialiste Covid -19" 35
2.10 Description de la vue statique (Diagramme de classes) . . . 36

3 Réalisation **37**
3.1 Environnement Logiciel 37
3.1.1 Visual Studio Code 37
3.1.2 Wamp server (anciennement WAMP5) 37
3.1.3 Langage HTML 38
3.1.4 Langage PHP 38
3.1.5 Langage JavaScript 38
3.1.6 Langage SQL 38
3.1.7 Langage CSS 38
3.2 Présentation de l'application 38
3.2.1 Conception de l'interface 38
3.2.2 Architecture générale de la plateforme 39
3.3 Sécurisation l'application 42

Table des figures

1.1 Architecture client/serveur. 2
1.2 Structure d'URL 5
1.3 Client-Serveur (site web dynamique). 7
1.4 Client-serveur (site web statique). 8
1.5 Structure d'un message SOAP. 9
1.6 Service d'hébergement. 11
1.7 Types d'hébergement. 12
1.8 Type de diagramme UML. 13
1.9 Example d'une classe. 13
1.10 Example d'une classe. 14

2.1 Diagramme de classe. 20
2.2 Diagramme de cas d'utilisation du 'chef de service (administrateur) global'. 21
2.3 Diagramme de cas d'utilisation 'chef de service (gestion des dossiers médicaux)'. 22
2.4 Diagramme de cas d'utilisation 'chef de service (gestion des médecins)'. 23
2.5 Diagramme de cas d'utilisation 'chef de service (gestion des infirmiers)'. 23
2.6 Diagramme de cas d'utilisation 'chef de service (Gestion des comptes)'. 24
2.7 Diagramme de cas d'utilisation 'Médecin généraliste (Global)'. 25
2.8 Diagramme de cas d'utilisation 'médecin généraliste (mise à jour de la liste des malades)'. 25
2.9 Diagramme de cas d'utilisation 'Médecin généraliste (gestion des dossiers médicaux)'. . 26
2.10 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (global)'. 26
2.11 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (gestion d'hospitalisation)'. 27
2.12 Diagramme de cas d'utilisation 'Pneumologue infectiologue (gestion des dossiers médicaux)'. 27
2.13 Diagramme de cas d'utilisation 'Médecin spécialiste'. 28
2.14 Diagramme de cas d'utilisation 'médecin spécialiste (consultation des dossiers médicaux)'. 29
2.15 Diagramme de cas d'utilisation 'médecin spécialiste (gestion de sortie)'. 29
2.16 Diagramme de cas d'utilisation 'Infirmier'. 30
2.17 Diagramme de séquence 'Authentification'. 31
2.18 Diagramme de séquence 'Gestion du dossier médical par chef de service'. 32
2.19 Diagramme de séquence 'Gestion du dossier médical par médecin généraliste'. 33
2.20 Diagramme de séquence 'Gestion du dossier médical par l'infirmier'. 34
2.21 Diagramme de séquence 'Gestion du dossier médical par spécialiste Covid-19'. 35

3.1 Module d'authentification. 39
3.2 Module du médecin généraliste. 40
3.3 Module du médecin pneumologue. 40
3.4 Module du médecin infectiologue. 41

3.5 Module du médecin spécialiste Covid-19. 41
3.6 Module du médecin spécialiste Covid-19 (suivi du malade). 42

Liste des abréviations

UML	Unified Modeling Language
HTML	HyperText Markup Language
HTTP	HyperText Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
UDDI	Universal Description Discovery and Integration
SGBD	Système de Gestion de Base de Données
CSS	Cascading Style Sheets
PHP	Hypertext Preprocessor
WSDL	Web Service Description Language
WWW	World Wide Web
SQL	Structured Query Language
SOAP	Simple Object Access Protocol
DCOM	Distributed Componet Object Model
EJB	Entreprise Java Bean

INTRODUCTION GENERALE

L'informatique est une nouvelle science qui s'est constituée durant le vingtième siècle, c'est la science du traitement automatique de l'information.

L'informatique est devenue un instrument indispensable de toute bonne administration. Les établissements de santé publique tels que l'hôpital Mohamed Boudiaf de Médéa a besoin de logiciels qui leur permettraient d'améliorer la gestion et la qualité des services en faveur des patients.

Vu la quantité importante d'information manipulées continuellement au niveau du service Covid-19, et sa gestion qui est encore manuel, d'où elle est confrontée à un bon nombre de difficultés telles que :

— Lenteur dans l'élaboration des dossiers des malades entrants et sortants.

— La difficulté de manipulations des informations.

— Un mauvais suivi des dossiers médicaux des malades.

Il s'agit pour nous de mettre en place un système d'information permettant d'éditer et d'assurer le suivi médical des patients. Nous avons devisé notre travail en trois chapitres qui sont :

— Chapitre 1 : nous avons présenté des généralités sur le web.

— Chapitre 2 : nous avons décrit l'analyse du besoin et la conception.

— Chapitre 3 : nous avons cité les différents programmes utilisés dans la réalisation de notre application, ainsi que des capture d'image de cette dernière.

Chapitre 1

Généralités sur le web

Introduction

L'informatique est une nouvelle science qui s'est constituée durant le vingtième siècle.
C'est la science du traitement automatique de l'information, elle est devenue un instrument indispensable de toute bonne administration.

Les établissements de santé publique, les hôpitaux ont besoin de logiciels qui leur permettraient d'améliorer la gestion et la qualité des services en faveur des patients.

Ce chapitre aura pour objectif de présenter quelques notions sur l'architecture Client/serveur, et les concepts des nouvelles technologies d'information et de communication et UML.

1.1 L'architecture client/serveur

L'architecture client/serveur désigne un mode de communication entre plusieurs ordinateurs d'un réseau qui distingue un ou plusieurs postes clients du serveur : chaque logiciel client peut envoyer des requêtes à un serveur.

A chaque fois qu'un navigateur (client) a besoin d'un fichier hébergé sur un serveur web, le navigateur va demander (envoyer une requête) ce fichier via le logiciel http. [**1**]

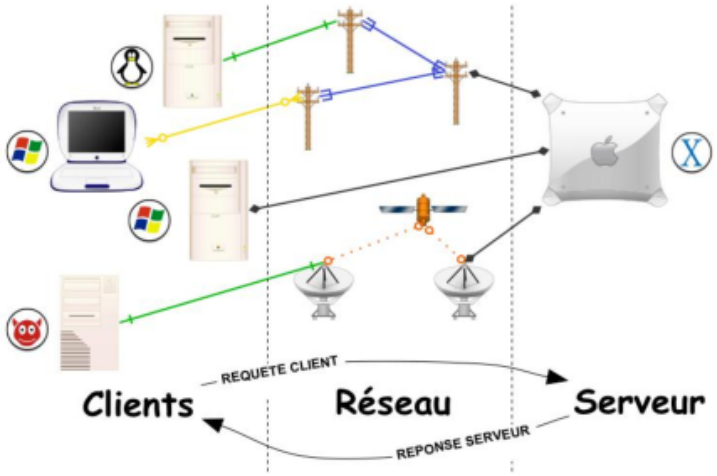


Figure 1.1 – Architecture client/serveur.

1.2 Notion de base

1.2.1 Serveur

1. Il est passif (ou maître).
2. Il est à l'écoute, prêt à répondre aux requêtes envoyées par des clients.
3. Dès qu'une requête lui parvient, il la traite et envoie une réponse.

1.2.2 Client

1. Il est actif (ou esclave).
2. Il envoie des requêtes au serveur.
3. Il attend et reçoit les réponses du serveur.[**2**]

1.2.3 Une requête

Message transmis par un client à un serveur décrivant l'opération à exécuter pour le compte du client.

1.2.4 Réponse

Message transmis par un serveur à un client suite à l'exécution une opération, contenant le résultat de l'opération. Le client et le serveur doivent bien sûr utiliser le même protocole de communication.

Dans une architecture client-serveur on trouve d'un côté le client et de l'autre le serveur.

Cette d'architecture peut se faire sur tout type d'architecture matérielles interconnectées.

Concrètement, Le client demande un service au serveur, le serveur reçoit cette requête effectue un traitement, et renvoie la ressource demandée par le client.[**3**]

1.3 Les différentes architectures client-serveur

1.3.1 Architecture 1-tiers

Présentation de L'architecture 1-tiers : Dans une application un tiers les couche s applicatives sont liées et s'exécutent sur le même ordinateur.

1.3.2 Architecture 2-tiers

L'architecture à deux niveaux caractérise les systèmes clients/serveurs pour lesquels le client demande une ressource et le serveur lui fournit directement, en utilisant ses propres ressources. Cela signifie que le serveur ne fait pas appel à une autre application afin de fournir une partie du service.

L'échange de messages transite à travers un réseau reliant les deux machines, il met en oeuvre des mécanismes relativement complexes qui sont, en général, pris en charge par un middleware.

L'architecture 2-tiers présente de nombreux avantages qui lui permettent de présenter un bilan globalement positif : elle permet l'appropriation des applications par l'utilisateur, l'utilisation d'une interface utilisateur riche et elle introduit la notion d'interopérabilité.

1.3.3 Architecture 3-tiers

Cette architecture 3-tiers également appelée client-serveur de deuxième génération ou client-serveur distribué applique les principes suivants : Les données sont toujours gérées de façon centralisée, la présentation est toujours prise par le post client, la logique applicative est prise en charge par un serveur intermédiaire.

L'architecture 3-tiers sépare l'application en 3 niveaux de services distincts, conforme aux principes précédents :

- **Premier niveau** : l'affichage et les traitements locaux sont pris en charge par le post client.
- **Deuxième niveau** : les traitements applicatifs globaux sont pris en charge par le service applicatif.
- **Troisième niveau** : les services de base de données sont pris en charge par un SGBD.

1.3.4 Architecture n-tiers

L'architecture multi-niveaux a été pensée pour pallier les limitations des architectures trois tiers et concevoir des applications puissantes et simples à maintenir.

Elle qualifie la distribution d'application entre de multiples services et non la multiplication des niveaux de services.

1.4 Les caractéristiques du client/serveur

- **Service :** le client est le fournisseur de service, le client est le consommateur de service.
- **Protocole :** c'est toujours que les clients qui déclenchent la demande de service.
- Le serveur attend passivement les requêtes des clients.
- **Partage des ressources :** Un serveur traite plusieurs clients au même temps et contrôle leurs accès aux ressources.
- **Localisation :** le logiciel client/serveur masque aux clients la localisation de serveur.
- **Hétérogénéit :** le logiciel client/serveur est indépendant des plates-formes matérielles et logicielles.
- **Redimensionnement :** il est possible d'ajouter et de retirer des stations clients.
  - Il est possible de faire évoluer des serveurs.
- **Intégrité :** les données du serveur sont gérées sur le serveur de façon centralisée.
- **Souplesse et adaptation :** on peut modifier le module serveur sans toucher au module client.

1.5 Présentation du web

Le Web est rapidement devenu le service le plus utilisé sur Internet. Il a conçu le HyperText Markup Language (HTML) à partir d'un autre format utilisé pour les documents appelé le SGML.

Le WWW fonctionne en utilisant le concept d'hypertexte. À l'intérieur d'une page, il y a des mots clés ou des images qui ont des liens qui, lorsque vous cliquez dessus, vous amènent à une autre page Web.

1.5.1 L'internet

L'Internet est un système de communication qui permet aux ordinateurs autour du monde de communiquer et de s'échanger de l'information entre eux.

1.5.2 Le web

Dans les années 90's un nouveau service d'internet est apparu, Word Wide Web désigne en anglais une toile d'araignée mondiale, désignée aussi par l'acronyme WWW, C'est une technologie informatique permettant de consulter des pages regroupées sur des sites. Il est toujours possible de consulter le tout premier site web.

1.6 Standards du web

Le web rassemble trois standards : URL, http et HTML.

URL

Uniform Resource Locator est une simple ligne de texte qui permet de retrouver une ressource (texte, image, musique, vidéo, programme...) sur internet. Il s'agit d'une chaîne de caractères ASCII imprimables qui se décompose en cinq parties :

— Le nom du protocole
— Identifiant et mot de passe
— Le nom du serveur
— Le numéro de port
— Le chemin d'accès à la ressource

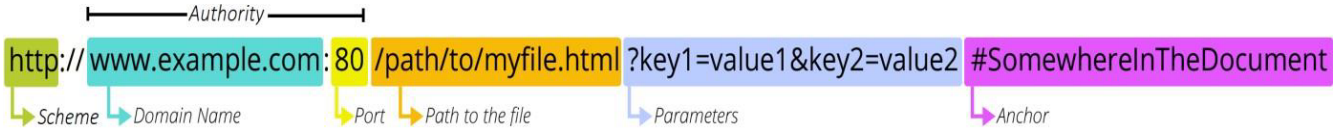


Figure 1.2 – Structure d'URL

Protocole HTTP

L'HTTP (HyperText Transfer Protocol) est le protocole de transport utilisé par les navigateurs Web et les serveurs Web a but de communiquer entre eux. [**1**]

Protocole HTML

C'est un langage de description de documents, il utilise des marqueurs explicités. [**1**]

1.7 Type des pages web

1.7.1 Les pages web dynamiques

Une page Web dynamique (interactive) est un document élaboré qui utilise des programmes externes pour réaliser des fonctions spécifiques.

Ces pages interactives permettent aux usagers de soumettre des formulaires, d'interroger des bases de données, de formater des résultats, de structurer l'affichage et d'avoir des accès à des parties du site protégées exemple saisie d'un mot de passe pour une session client.

Les pages dynamiques sont mises en oeuvre grâce à un langage de programmation.

Grâce à lui, on pourra disposer d'instructions conditionnelles, de boucles et de fonctions de traitement complexes.

1.7.2 Les pages web statiques

Une page Web statique est un document qui peut être lu de haut en bas sans quitter le document, elle présente ainsi la même information à tous les usagers.

Les pages statiques font appel au langage HTML qui est un langage de description de données. [**4**]

1.8 Site web

1.8.1 Site web dynamique

Un site dynamique (appelé aussi application Web) constitué essentiellement de pages dynamiques, nous offres des services qui dépassent la simple visualisation d'informations statiques, l'application est souvent reliée à une base de données. Les Fonctionnalités offertes sont plus importantes : recherche, saisie de formulaire…et la mise jour des informations.

Fonctionnement d'un site dynamique : L'administrateur du site et le visiteur utilisent le même outil : le navigateur Web.

Mais les pages avec lesquelles travaille l'administrateur ne sont pas les mêmes que celle que le visiteur voit.

De son côté, le visiteur visualise es pages qui font appel au contenu de la base de données.

C'est le serveur Web qui s'occupe de récupérer le contenu des formulaires que remplit l'administrateur, et de renvoyer cette information dans les pages vues par le visiteur.

Avantages des sites dynamiques

- Moins de publication de pages Web sur le site donc moins de risque d'incohérence.
- Plus grande automatisation de la mise à jour.
- Possibilité de séparer contenu et forme.
- Plus d'interactivité et personnalisation.

Inconvénients des sites dynamiques

- Nécessité des connaissances préalables en programmation.
- Nécessité de connaissances préalables en base de données.
- Traitement des requêtes plus long.



Figure 1.3 – Client-Serveur (site web dynamique).

1.8.2 Site statique

Définition

Un site statique est constitué essentiellement de page Web statique dont tous les documents sont statiques et proviennent directement et sans traitement, de fichiers chargés à partir d'un site distant.

Le Fonctionnement d'un site statique : dans un site statique l'administrateur du site compose les pages Web avec un éditeur HTML puis ces pages sont stockées sur le serveur Web.

Celui-ci renvoie ces pages à la demande au visiteur (client), par la suite ces pages ne pourront être modifiées que via un éditeur HTML, par l'administrateur.

Le contenu de ces pages est fixe comme un fichier Word, et n'est pas modifié par le serveur.

Le site est donc dit « statique » car son contenu ne change que par intervention humaine et non pas par des fonctions automatiques opérés par le serveur.

Avantages des sites statiques

— Peu de ressources.

— Simplicité traitement serveur.

Inconvénients des sites statiques

— Manque de séparation contenu/forme.

— Obligation de créer et de publier sur le serveur avant consultation.

— Obligation de réécrire la page en cas de modification du contenu ou de présentation.

— Obligation de réécrire toutes les pages du site en cas d'une nouvelle présentation. Personnalisation de site impossible.

1.9 Les services web

Les services web sont la nouvelle vague des applications modulaires, auto contenues et auto descriptives qui peuvent être publies, localisées, invoquées depuis le Web.

Les Web services effectuent des actions allant de simples requêtes a des processus métiers complexes.

Une fois qu'un Web service est déployé, d'autres applications (y compris des Web services) peuvent le découvrir et l'invoquer.



Figure 1.4 – Client-serveur (site web statique).

1.9.1 Historique

Les services Web prennent leur origine dans l'informatique distribuée et dans l'avènement du Web. Le Web en passant par trois grandes générations, dont la première se basait sur de sites Web fournissant des pages Web statiques (html) sans structure et impossible à fusionner et la seconde ou un site Web fournissait des applications réelles (pages Web dynamique, ASP, JSP, PHP…) a enfin abouti à une troisième génération ou un site Web est un composant fournissant des services en XML (Web services) qui sont devenus structurés et possible à fusionner.

Le but de l'informatique distribuée quand a elle est de permettre à une application sur une machine d'accéder à une fonction d'une autre application sur une machine distante et ce de la même manière que l'appel d'une foncions locale, indépendamment des plates formes et des langages utilisés.

Le second point qu'a favorisé l'essor des services Web, c'est le développement d'Internet par la démocratisation du haut débit, la structuration des données via XML et la recherche d'interopérabilité.

Les entreprises publiaient déjà de l'information via des sites Web, utilisaient du commerce électronique, elle l'utilise maintenant pour leurs applications métiers (ressources humaines, ventes, finances etc..).

Une grande course entre les principaux acteurs du marché a progressivement, par soucis d'interopérabilité, adopté les principaux standards des services Web que sont SOAP, WSDL et UDDI.

Et enfin, le dernier point qu'a contribué à la naissance des Web services est une nouvelle façon de penser qu'a commencé depuis 1990 avec le développement orienté objet qu'a atteint très vite ses limites, suivi en 1998 du développement composant (ou super-objet comportant des services mais sans polymorphisme ni héritage) comme les EJB dans le modèle JEE, DCOM dans l'architecture DNA et COBRA. [**5**]

1.9.2 Objectifs

La technologie Web service a été mise au point en vue d'atteindre un certain nombre d'objectif qui sont :

- Accès rapide, intégré et généralisé à l'information pertinente
- Système ouvert réduisant le cout
- Développement rapide d'application
- Utilisation de composants distribués.
- Réduction des couts de développement.
- Administration simplifiée du système.
- Depuis le Web avec des outils d'assistance.

1.9.3 Les standards

XML

Est en quelle sorte un langage HTML amélioré permettant de définir de nouvelles balises.

XML est un sous ensemble de SGML, Utilisé dans le milieu de la gestion électronique documentaire (GED).XML reprend la majeure partie des fonctionnalités de SGML.

En réalité les balises XML décrivent le contenu plutôt que la présentation (contrairement à HTML).

Ainsi, XML permet de séparer le contenu de la présentation. Ce qui permet par exemple d'afficher sans pour autant nécessiter de créer autant de versions du document que l'on nécessite de représentations La force de XML réside dans sa capacité à pouvoir décrire n'importe quel domaine de données grâce à son extensibilité. [**5**] [**6**] [**7**]

Le SOAP

Le SOAP est un protocole de communication défini à l'origine par Microsoft, puis standardisé par le W3C, utilisant la notion XML permettant de définir.

Les mécanismes d'échanges d'information entre des clients et des fournisseurs de services Web. Le mécanisme de transport des messages SOAP peut être le protocole http, SMTP, FTP, etc.[**7**]

— **Les messages SOAP** Un message SOAP n'est en effet pas qu'un fichier XML classique, avec seulement un espace de nom prenant en charge les diverses extensions. Le document décrivant un message de ce type est en réalité une déclaration très structurée, appelée « enveloppe », et contenant le nécessaire à l'exécution de l'opération qu'elle contient. Principalement, une enveloppe SOAP peut être décomposée en deux composants : l'entête (header) et le corps (body) du message

En résumé, la structure d'un message SOAP peut être illustrée par la figure suivante :

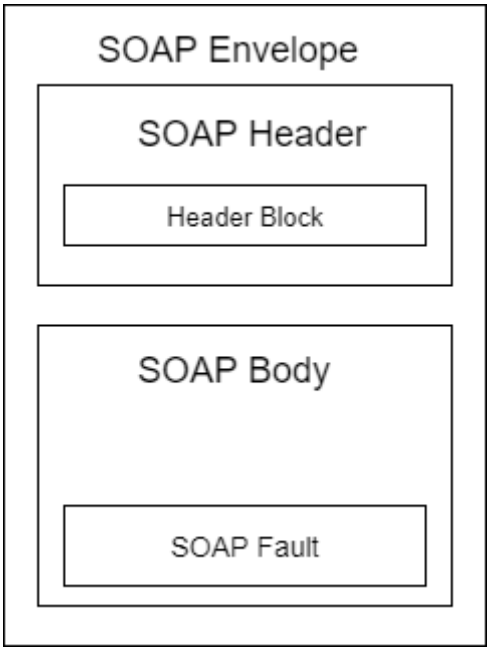


Figure 1.5 – Structure d'un message SOAP.

— **L'enveloppe de message** C'est l'élément supérieur du document, in englobe en-tête et le corps. Il est obligatoire « sans enveloppe, le message ne peut pas être transporté ».

L'enveloppe est la racine du document XML contenant le message SOAP, marquée par la balise. La spécification impose que tous les attributs de cette balise et de celles imbriquées soient explicitement associés à un Namespace, de manière a supprimé toute ambigüité.

— **L'en-tête de message** Placé au sein de l'enveloppe avant même le corps, l'entête peut être utilisé pour compléter les informations nécessaires à une requête.

Les en-têtes SOAP sont optionnels et sont typiquement utilisées pour transmettre des données d'authentification ou de gestion de session.

L'entête reconnaît plusieurs attributs spécifiques :

- L'attribut « actor » : indique le destinataire de l'information indiqué par l'entête.
- L'attribut « mustunderstand » : prend une valeur booléenne et indique si le traitement de l'élément est obligatoire ou non.
- L'attribut « encodingstyle » : dont la valeur est un URI spécifie les règles d'encodage s'appliquent à l'élément.

— **Le corps de message** Cette section contient les données transportées par le message SOAP qui comme pour les éléments précédents, doit voir tous ces sous-éléments correctement qualifiés par des espaces de nom. Un message d'erreurs utilise le sous-élément Fault, qui lui-même dispose de quatre sous éléments possibles :

- Fault code, identifiant l'erreur par un code.
- Faultstring, une explication lisible de l'erreur.
- Faultactor, désigne l'origine de l'erreur.
- Détail, donne des détails spécifiques.

UDDI

C'est un annuaire de services basé sur XML et plus particulièrement destiné aux Web services, il peut être vu comme les pages blanches des services Web.

C'est un annuaire permettant à des fournisseurs de présenter leurs services à des clients. L'annuaire UDDI est consultable sous plusieurs facettes :

— **Les pages blanches :** elles comportent des informations sur les entreprises et contiennent des coordonnées et des descriptions de l'entreprise consultable par l'utilisateur.

— **Les pages jaunes :** comportent la description de service Web déployés par les entreprises.

— **Les pages vertes :** fournissent des informations techniques (souvent en format WSDL) détaillées sur les services fournis.

WSDL

WSDL est un langage reposant sur XML dont on se sert pour décrire les services-Web.

Il est indispensable à UDDI pour permettre aux clients de trouver les méthodes leur permettant d'invoquer les services Web.

WSDL décrit une interface publique d'accès à un service Web, notamment dans le cadre d'architecture de type SOA.

Eléments d'une définition WSDL

Au-delà de la notion de service et de définition de types de données classiques avec les schémas, WSDL introduit les notions de port, liaison, message pour décrire les services.

— L'élément <définition> est la racine du document WSDL.

— **Elément <TYPE> :** contient les définitions de types utilisant un système de typage.

— **Elément <message> :** contient les noms et types d'un ensemble de champs à transmettre (paramètre d'une invocation, valeur de retour).

— **Elément <port type> :** décrit un ensemble d'Operations, chaque opération à zéro ou un message en entrée, zéro ou plusieurs messages de sortie ou de fautes.

— **Elément <binding> :** spécifie une liaison d'un à un protocole concret (SOAP, http, MIME). Un peut avoir plusieurs liaisons !

— **Elément <port> :** spécifie un point d'entrée (end point) comme la combinaison d'un et d'une adresse réseau.

— **Elément <service> :** une collection de points d'entrée (end point) relatifs.

1.10 Hébergement

Un hébergeur est une entreprise de services informatiques hébergeant (mettant en ligne) sur ses serveurs Web les ressources constituant les sites Web de ses clients. On distingue :

— Les hébergeurs gratuits : Les fournisseurs d'accès à Internet proposent généralement dans leur offre une formule d'hébergement de site Web avec un espace de stockage assez limité, en effet, le service d'hébergement est souvent rendu en contrepartie de publicités sur le site.

— Les hébergeurs professionnels : Ce type d'hébergement garanti un service de qualité et de sécurité.

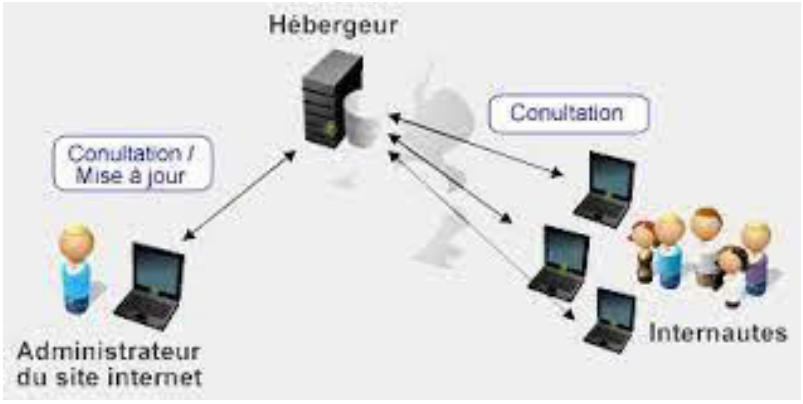


Figure 1.6 – Service d'hébergement.

Voici les différentes méthodes d'hébergement :

1.10.1 Les hébergements sur un serveur dédié

Ici, votre site est sur un serveur que vous avez loué ou acheté.

Si vous prévoyez un très gros site avec plus de 10 0000 visiteurs/mois, cette solution est idéale, mais s'avère très onéreuse.

Si vous avez plusieurs sites vous pouvez ainsi les mettre sur un même serveur.

Ce type d'hébergement est souvent privilégié par les grandes entreprises avec une équipe dédiée à l'hébergement du site.

1.10.2 Les hébergements VPS (Serveur Privé Virtuel)

Le serveur est partagé, mais cela est limité à 20 sites.

Tous les sites du même serveur ont la même capacité de mémoire et d'espace disque. C'est une solution égalitaire, contrairement au serveur mutualisé.

Vous êtes donc limité dans les capacités de développement de votre site.

Cette solution est privilégiée pour les sites tests ou les petits sites qui n'évolueront pas ou très peu.

1.10.3 Les hébergements cloud

Le serveur qui héberge les données peut accéder à des ressources extérieures au serveur utilisé.

Dans l'exemple où l'un d'eux tombe en panne, d'autres serveurs dédiés interviennent pour que tout fonctionne, contrairement aux autres solutions qui n'utilisent qu'un seul serveur.

Ce type d'hébergement est cependant plus friable au niveau de la sécurité. [**8**]

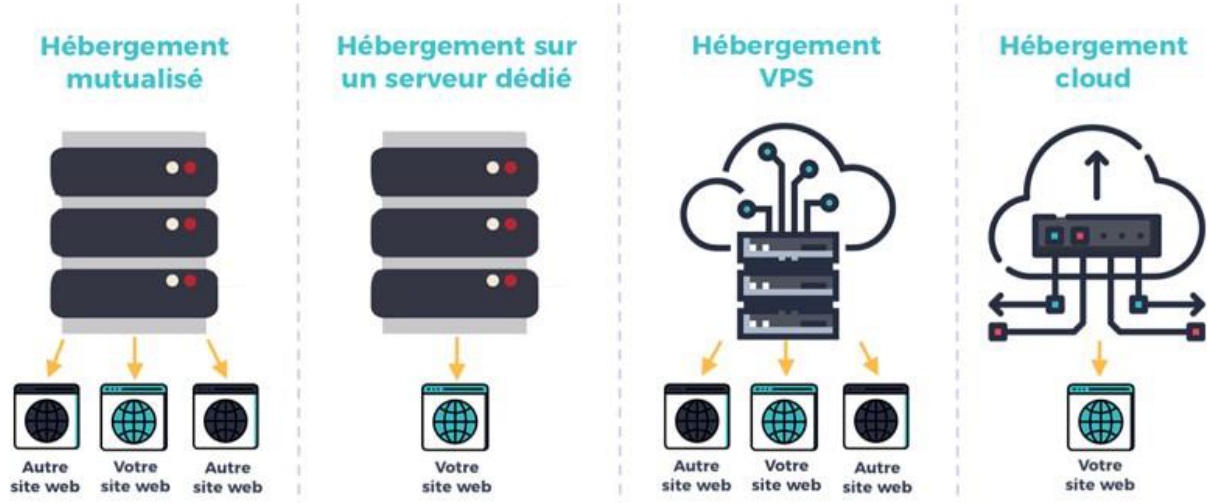


Figure 1.7 – Types d'hébergement.

1.11 UML

UML se définit comme un langage de modélisation graphique et textuel formel et normalisé.

Il est destiné à comprendre et décrire des besoins, spécifier et documenter les systèmes, et sert aussi à esquisser des architectures logicielles, concevoir des solutions et communiquer des points de vue.

UML unifie à la fois les notations et les concepts orientés objet. Il ne s'agit pas d'une simple notation graphique, car les concepts transmis par un diagramme ont une sémantique précise et sont porteurs de sens au même titre que les mots d'un langage.

C'est un support de communication performant : Il encadre l'analyse, facilite la compréhension de représentation abstraite complexe. Son caractère polyvalent et sa souplesse en font un langage universel.

Les bases d'UML et Les éléments se sont les abstractions essentielles aux modèles. [**9**]

— Les relations Expriment les liens existants entre les différents éléments

— L'ensemble des diagrammes réunis permet d'obtenir une vue globale du système, chaque diagramme va permettre de modéliser ou spécifier une vue (spécifié) du système à concevoir.

Les diagrammes fréquents sont :

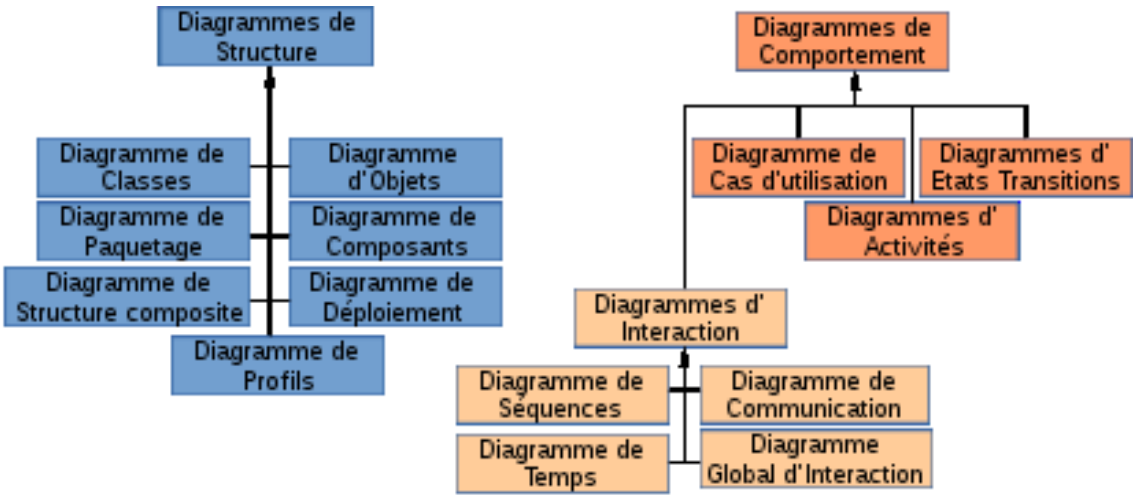


Figure 1.8 – Type de diagramme UML.

— **Diagramme de classe**
  — Représentation de la structure interne du logiciel
  — Utilisé surtout en conception mais peut être utilisé en analyse

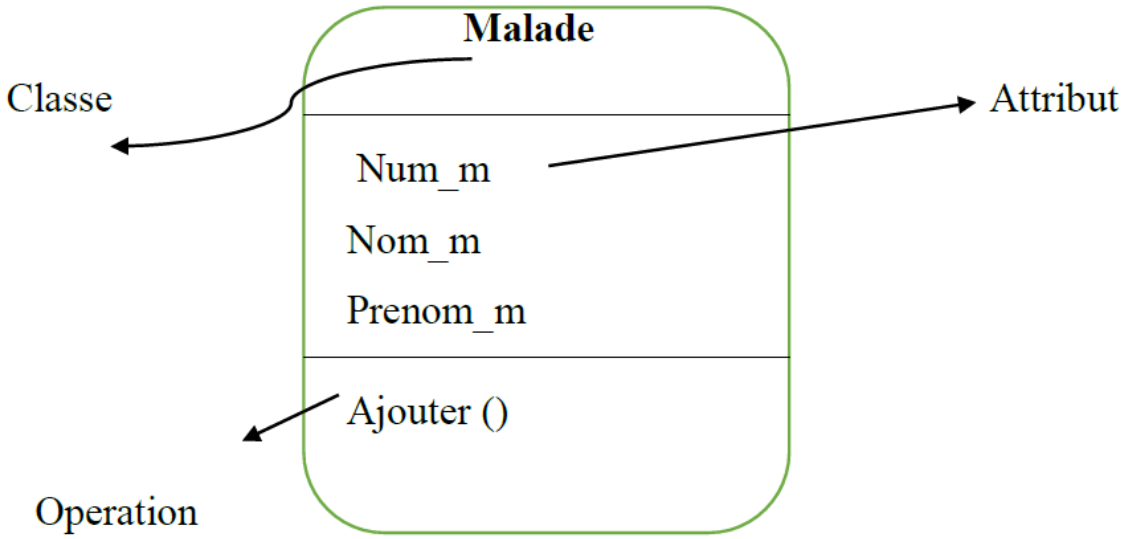


Figure 1.9 – Example d'une classe.

— **Diagramme d'objets**
  — Représentation de l'état du logiciel (objets + relations)
  — Diagramme évoluant avec l'exécution du logiciel - création et suppression d'objets - modification de l'état des objets (valeurs des attributs) - modification des relations entre objets.[**10**]

— **Diagramme de cas d'utilisation** Ce diagramme est utilisé dans l'activité de spécification des besoins. Il montre les interactions fonctionnelles entre les acteurs et le système à l'étude. [**10**]

— **Diagramme d'état** Représente le cycle de vie commun aux objets d'une même classe. Ce diagramme complète la connaissance des classes en analyse et en conception en montrant les différents états et transitions possibles des objets d'une classe à l'exécution. [**10**]

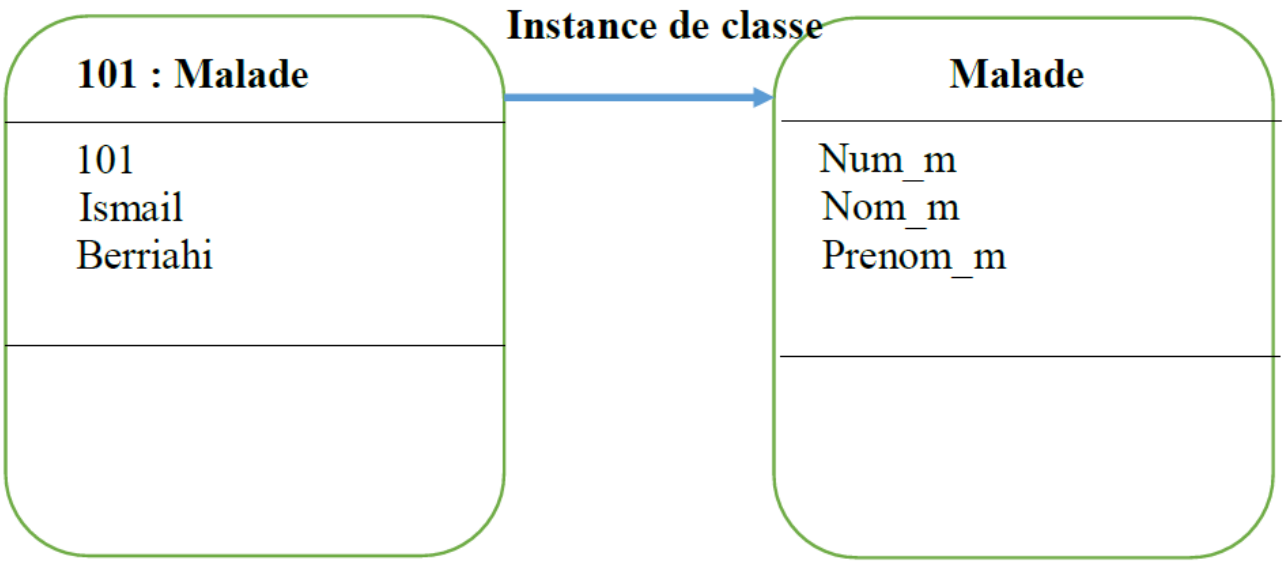


Figure 1.10 – Example d'une classe.

— **Diagramme d'activités** Représente les règles d'enchaînement des actions et au sein d'une activité. Il peut également être utilisé comme alternative au diagramme d'états pour décrire la navigation dans un site web. [**10**]

Conclusion

Après avoir abordé dans ce chapitre globalement les concepts des nouvelles technologies de l'information et de la communication telles que l'Internet, le Web et le Web service.

Nous étalerons dans le chapitre suivant toutes les étapes de l'analyse et conception de notre application.

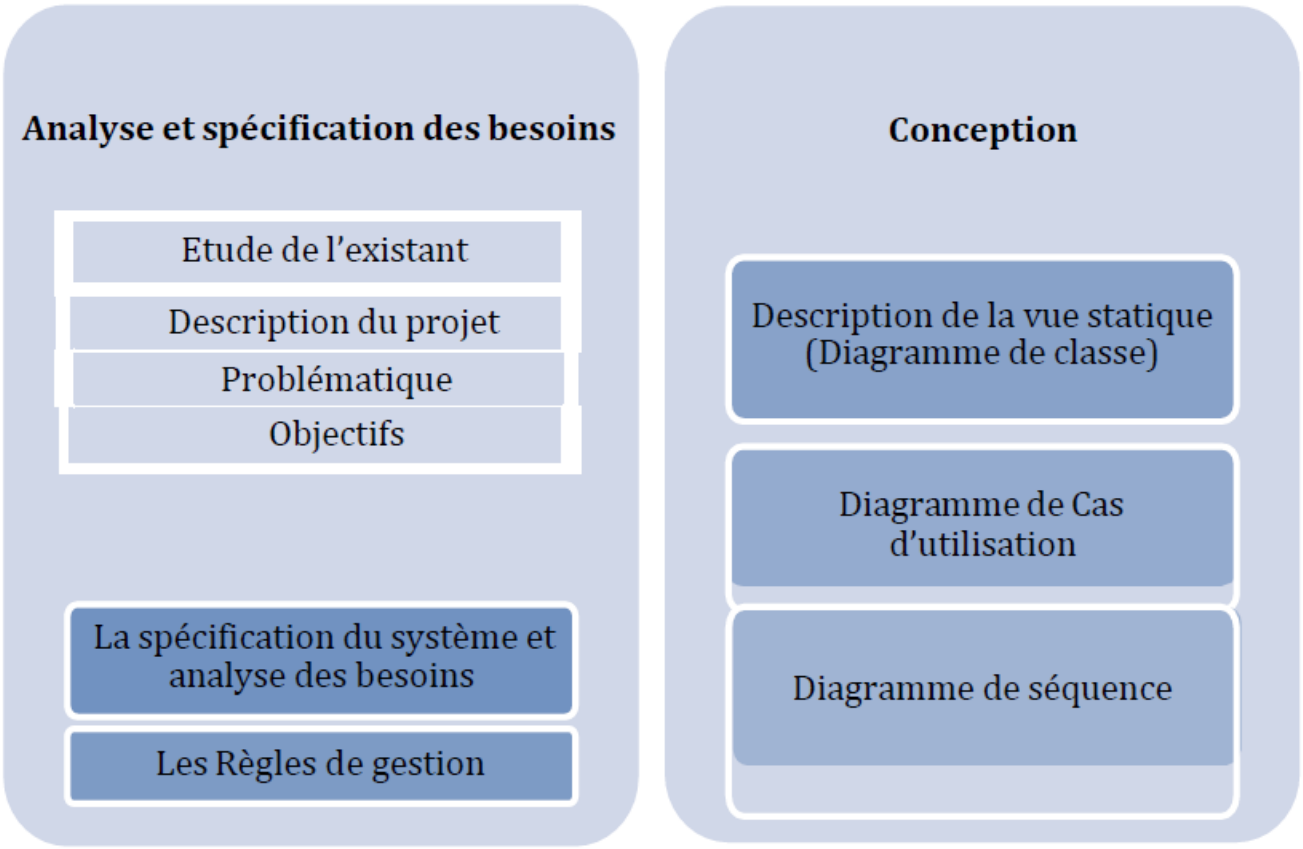
Chapitre 2

Analyse et conception

Introduction

Dans ce chapitre, nous décrirons dans la première partie le contexte du projet, nous définissons la problématique et nous exposerons les objectifs de notre travail ; puis nous passerons aux choix de la méthode de conception et à l'identification des besoins.

Dans la deuxième partie, nous proposons une conception de l'application, ainsi que la présentation des différents diagrammes : diagramme de classe, diagramme de cas d'utilisation et en fin diagramme de séquence.



Partie I : Analyse et spécification des besoins

2.1 Quelques définitions

2.1.1 Définition du langage UML

UML est un standard conçu pour l'écriture de plans d'élaboration de logiciels.

Il peut être utilisé pour visualiser, spécifier, construire et documenter les artefacts d'un système.

Les diagrammes UML permettent de communiquer l'architecture d'un système aux concepteurs, développeurs et à toutes personnes externes à l'aspect technique d'un système.

Cette communication prend une forme visuelle et narrative dans :

Le document des exigences fonctionnelles

Qui regroupe les diagrammes de cas d'utilisation avec leurs descriptions, les diagrammes d'objet, les diagrammes d'interactions et les diagrammes d'activités.

Le document d'architecture technique

Qui regroupe les diagrammes de classes, les diagrammes de package, les diagrammes de composants, les diagrammes déploiement et les diagrammes d'états.

2.2 Pourquoi utiliser UML ?

L'un des tout premiers avantages de l'UML est de faire rencontrer et communiquer les utilisateurs et les informaticiens.

— UML est riche (il couvre toutes les phases d'un cycle de développement)

— UML est ouvert (il est indépendant du domaine d'application et des langages d'implémentations)

— Sa notation graphique permet d'exprimer visuellement une solution objet

— L'aspect formel de sa notation limite les ambigüités et les incompréhensions

— Son aspect visuel facilite la comparaison et l'évaluation des solutions

— Il facilite la compréhension des représentations abstraites, complexes

— Son caractère polyvalent et sa souplesse en font un langage universel

C'est principalement pour ces raisons que l'OMG groupe de gestion de l'objet a adopté les spécifications UML comme standard commun à la modélisation.

2.3 Etude de l'existant

2.3.1 Présentation de l'organisme d'accueil

Cette application vise à apporter le meilleur suivi médical possible aux malades Covid-19 par les différents acteurs au sein des hôpitaux.

2.3.2 Description du projet

Tout d'abord, le malade arrive avec une lettre d'orientation (privée ou public), il va au service de consultation des malades Covid-19, ensuite il est consulté par le médecin généraliste puis il appelle pneumologue ou l'infectiologue pour la décision d'hospitalisation, après qu'il soit hospitalisé il doit ramener une fiche navette, et il doit être transféré au service maladie infectieuse, après il sera isolé et déclaré pour les statistiques.

Ensuite le médecin lui fait les soins nécessaires (la perfusion, sonde nasogastrique, sonde vésical), on continue avec un diagnostic par PCR, test antigénique, sérologie, et le scanner, après la confirmation du diagnostic on applique le protocole de traitement (ATP, anticoagulant et les antibiotique), pendant tout ce temps le médecin spécialiste le surveille 24/24h, lorsqu'il voit son état de santé s'aggrave, il le transfert au service de réanimation Covid-19.

A la fin, soit le malade sort de l'hôpital guéri, la sortie doit être signée par le bureau de sortie, soit il décède et est transféré à la morgue.

Le but essentiel de la gestion des malades est le suivi et l'historique, et l'intérêt de l'informatisation est :

— D'informatiser les dossiers médicaux des malades.

— Consulter les dossiers médicaux en ligne.

2.3.3 Problématique

Les différents services d'hôpital désirent informatiser et synchroniser les dossiers médicaux des malades covid-19 d'une manière performante, facile et efficace afin de faciliter la tâche aux différents acteurs des hôpitaux et garantir le maintien de l'activité pour détecter les problèmes existants.

Nous avons pu observer les insuffisances suivantes :

— Volume important des informations traitées manuellement (ordonnances, scanner, …), ce qui provoque parfois des erreurs dans l'établissement des documents.

— Recherche difficile sur les archives qui engendre une perte de temps.

— Insécurité des informations.

— Possibilité d'erreur dans le remplissage des différents dossiers médicaux.

— Possibilité d'erreur dans le calcul des statistiques.

— Nombre important des archives qui engendre une difficulté de stockage.

— Détérioration des archives à force de leur utilisation trop fréquente.

2.3.4 Objectifs

Notre objectif est de proposé une conception d'une application Web pour la gestion des malades Covid-19 aux seins de l'hôpital, qui doit répondre mieux aux besoins et corriger les anomalies souvent fréquentées dans la gestion manuelle et permet d'assurer :

— La rapidité dans l'établissement de traiter les dossiers médicaux.

— La facilité de la recherche et l'accès aux informations.

— Le stockage des informations sur des supports informatiques ce qui assurera leur sécurité.

— Le gain de temps dans le calcul des statistiques.

— L'automatisation des tâches qui se traitent manuellement.

2.4 La spécification du système et analyse des besoins

Nous présentons ici les besoins du système à concevoir, Le but de cette phase est de définir ce qu'on doit faire.

Ce système doit permettre d'effectuer les tâches fondamentales suivantes :

2.4.1 Besoins du chef de service

— Gestion des dossiers médicaux.

— Gestion des comptes (ajout, modification, suppression).

— Gestion des malades.

— Gestion des médecins et infirmiers.

2.4.2 Besoins du médecin généraliste

— Gestion des malades (ajout, retrait).

— Gestion des dossiers médicaux.

— Gestion des résultats de consultations médicales.

2.4.3 Besoin du pneumologue et l'infectiologue

— Consulter les dossiers médicaux des malades.

— Gestion d'hospitalisation (hospitaliser le malade, autoriser la sortie du malade).

— Gestion des scanners et bilans.

2.4.4 Besoin du médecin spécialiste Covid-19

— Consulter les dossiers médicaux des malades.

— Consulter la liste des consultations médicales.

— Gestion des soins appliqués au malades.

— Gestion de sortie du malade (autoriser la sortie du malade, prolonge la durée d'hospitalisation).

2.5 Les Règles de gestion

— Un chef de service, un généraliste, un spécialiste Covid-19, un infirmier, un infectiologue et un pneumologue ont un ou plusieurs comptes.

— Un chef de service et un spécialiste Covid-19 peuvent gérer un ou plusieurs dossiers médicaux.

— Un infectiologue et un pneumologue peuvent consulter un ou plusieurs dossiers médicaux.

— Un généraliste peut créer un ou plusieurs dossiers médicaux.

— Un infirmier peut appliquer un ou plusieurs soins.

— Un dossier médical peut être géré un ou plusieurs par un spécialiste Covid-19.

— Un dossier médical peut être géré par un seul chef de service.

— Un dossier médical peut être consulté par un ou plusieurs infectiologues et pneumologues.
— Un infectiologue demande un ou plusieurs bilans.
— Un bilan est demandé par un seul infectiologue.
— Un pneumologue demande un ou plusieurs scanners.
— Un scanner est demandé par un seul pneumologue.
— Un malade a un seul dossier médical.
— Un dossier médical appartient à un seul malade.
— Un malade habite dans une seule commune.
— Une commune appartient à une seule wilaya.

2.6 Identification des Acteurs

2.6.1 Chef service covid-19

Son rôle détient tous les droits du système :

— Gestion des autres acteurs de l'application.
— Gestion des comptes des acteurs de l'application.
— Gestion des dossiers médicaux.

2.6.2 Médecin généraliste

Il peut accéder au système à travers l'application pour gérer les dossiers médicaux, et mettre à jour la liste des malades et la gestion des résultats des consultations médicales.

2.6.3 Médecin pneumologue et l'infectiologue

Le pneumologue et l'infectiologue peuvent appliquer les mêmes actions que le médecin généraliste, ils peuvent appliquer en plus la mise à jour de l'hospitalisation et la demande de scanners et bilans.

2.6.4 Médecin spécialiste covid-19

Le médecin spécialiste peut consulter les dossiers médicaux et la liste de consultations médicales du malade, gestion du soin appliqué au malade et la gestion de sortie du malade.

2.6.5 Chef service covid-19

L'infirmier a pour rôle de consulter les dossiers médicaux pour appliquer les soins nécessaires au malades.

Partie II : Les diagrammes

2.8 Diagramme de cas d'utilisation

2.8.1 Diagramme de cas d'utilisation "chef de service (administrateur) global "

Le chef de service a comme rôle principal de gérer toutes les tâches de la plateforme ; gestion des médecins, infirmier et malade et la gestion des dossiers médicaux.

La figure ci-dessous représente le diagramme de cas d'utilisation globale du chef de service :

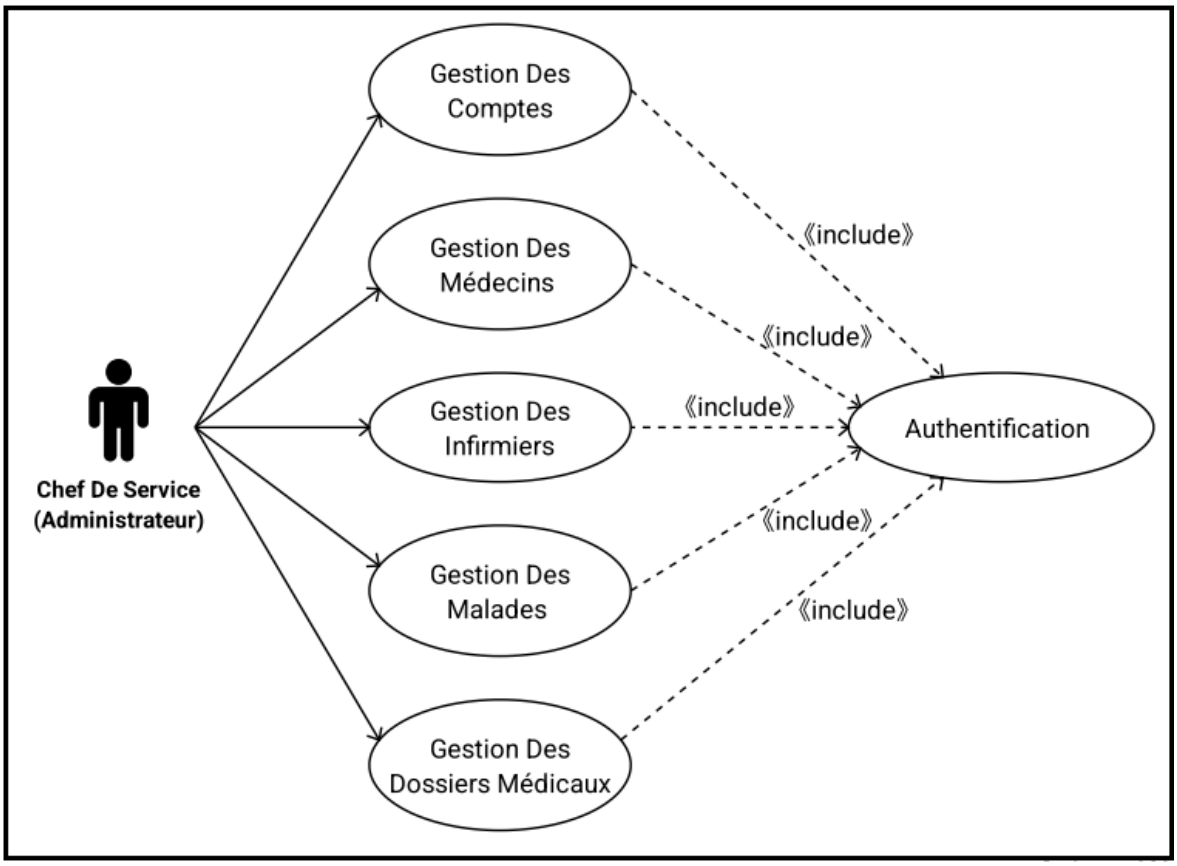


Figure 2.2 – Diagramme de cas d'utilisation du 'chef de service (administrateur) global'.

2.8.2 Diagramme de cas d'utilisation 'chef de service (gestion des dossiers médicaux)'

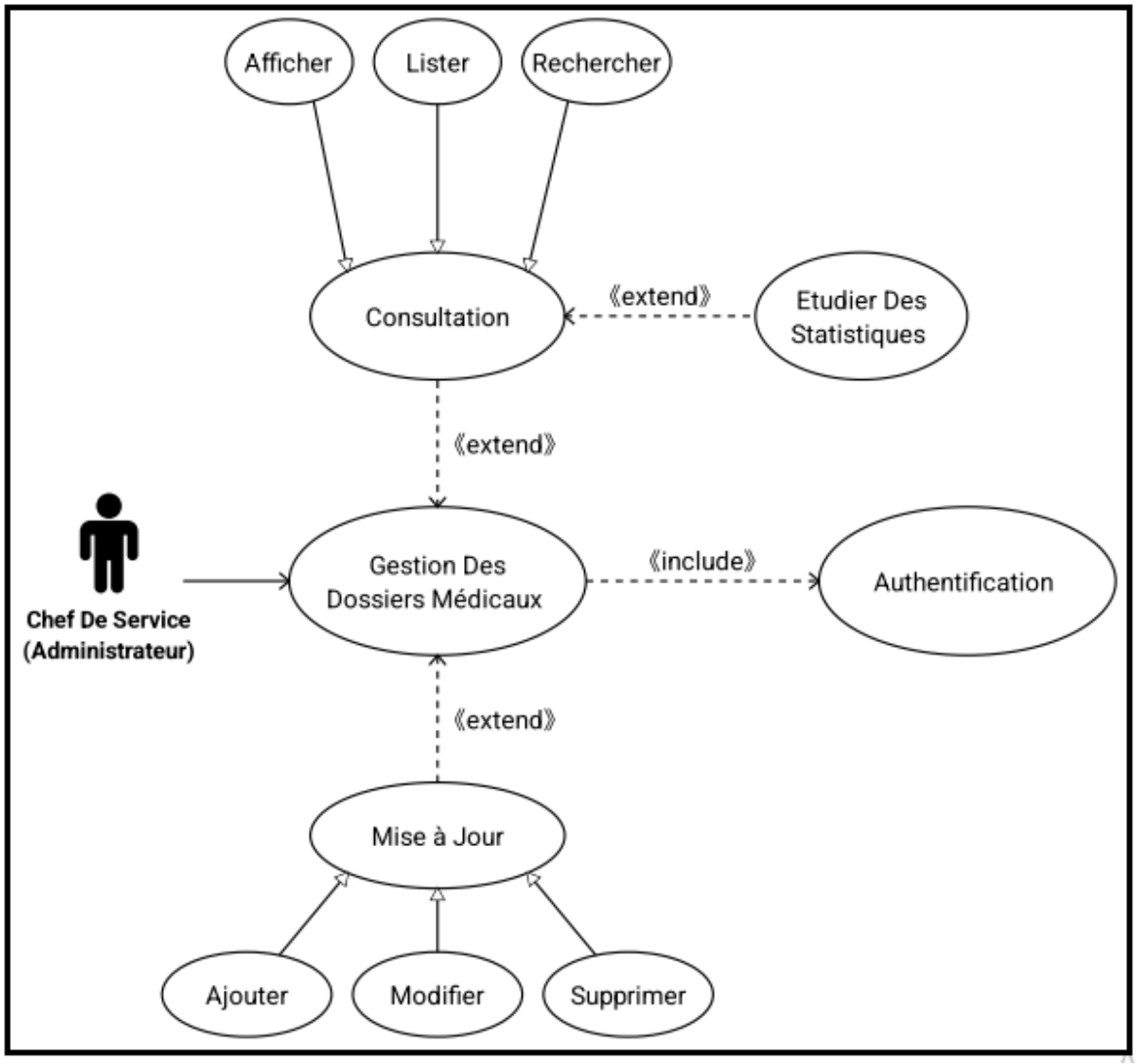


Figure 2.3 – Diagramme de cas d'utilisation 'chef de service (gestion des dossiers médicaux)'.

2.8.3 Diagramme de cas d'utilisation 'chef de service (gestion des médecins)'

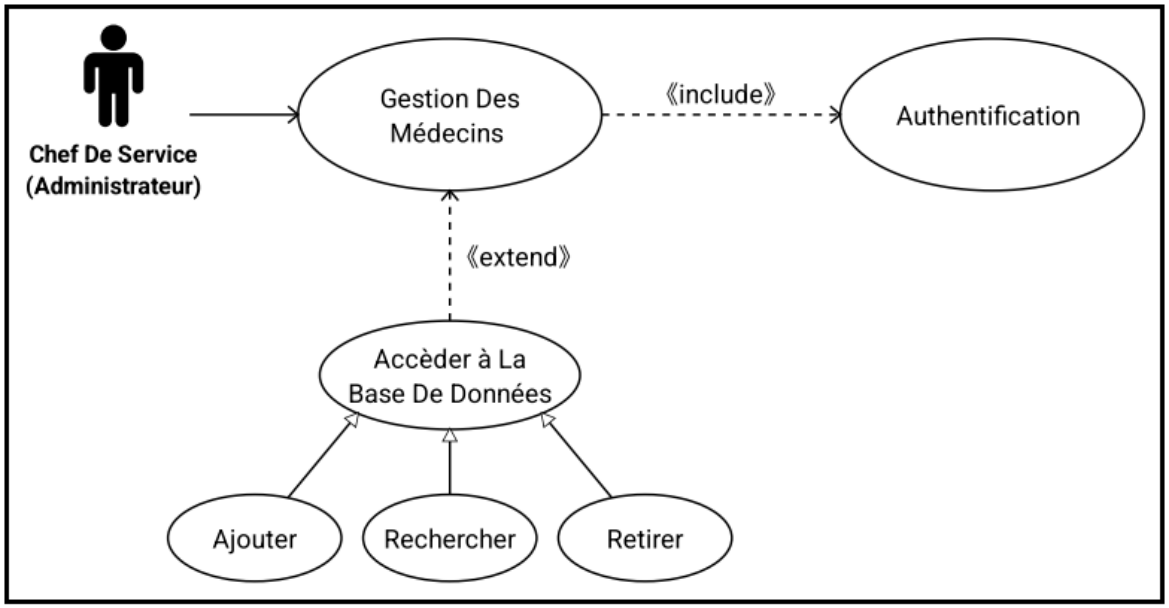


Figure 2.4 – Diagramme de cas d'utilisation 'chef de service (gestion des médecins)'.

2.8.4 Diagramme de cas d'utilisation 'chef de service (gestion des infirmiers)'

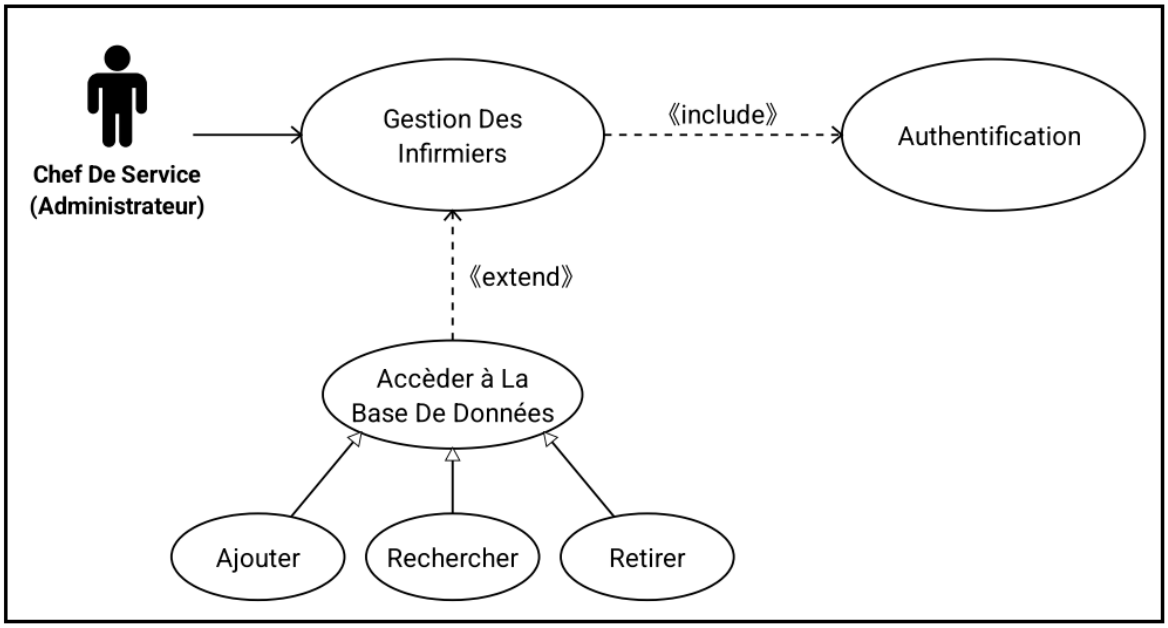


Figure 2.5 – Diagramme de cas d'utilisation 'chef de service (gestion des infirmiers)'.

2.8.5 Diagramme de cas d'utilisation 'chef de service (gestion des comptes)'

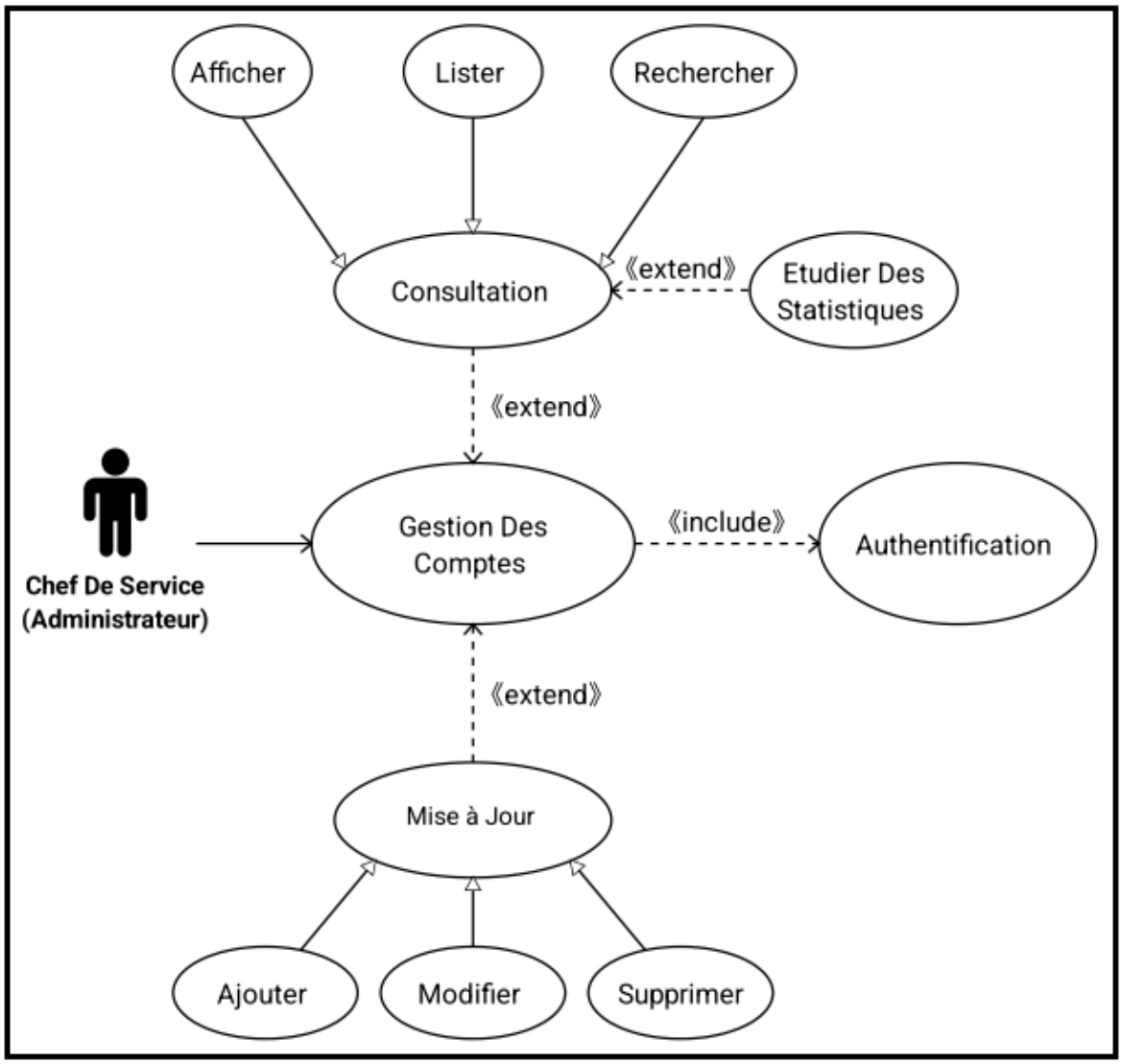


Figure 2.6 – Diagramme de cas d'utilisation 'chef de service (Gestion des comptes)'.

2.8.6 Diagramme de cas d'utilisation 'Médecin généraliste (Global)'

Le médecin généraliste fait la gestion des dossiers et les consultations médicales du malade, la mise à jour de la liste des malades et la consultation du nombre du malade.

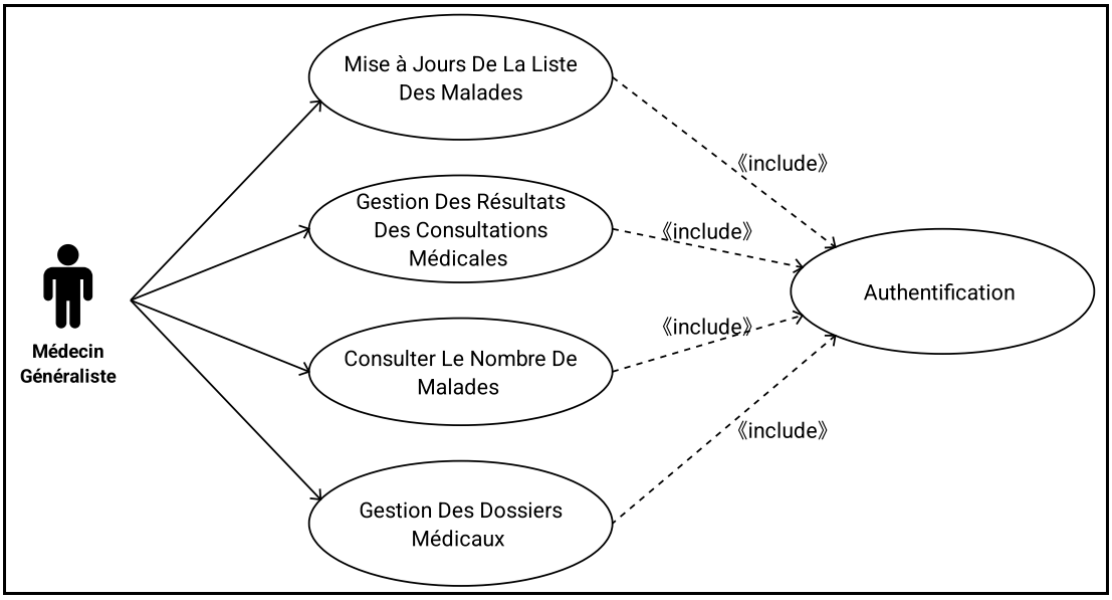


Figure 2.7 – Diagramme de cas d'utilisation 'Médecin généraliste (Global)'.

2.8.7 Diagramme de cas d'utilisation 'médecin généraliste (mise à jour de la liste des malades)'

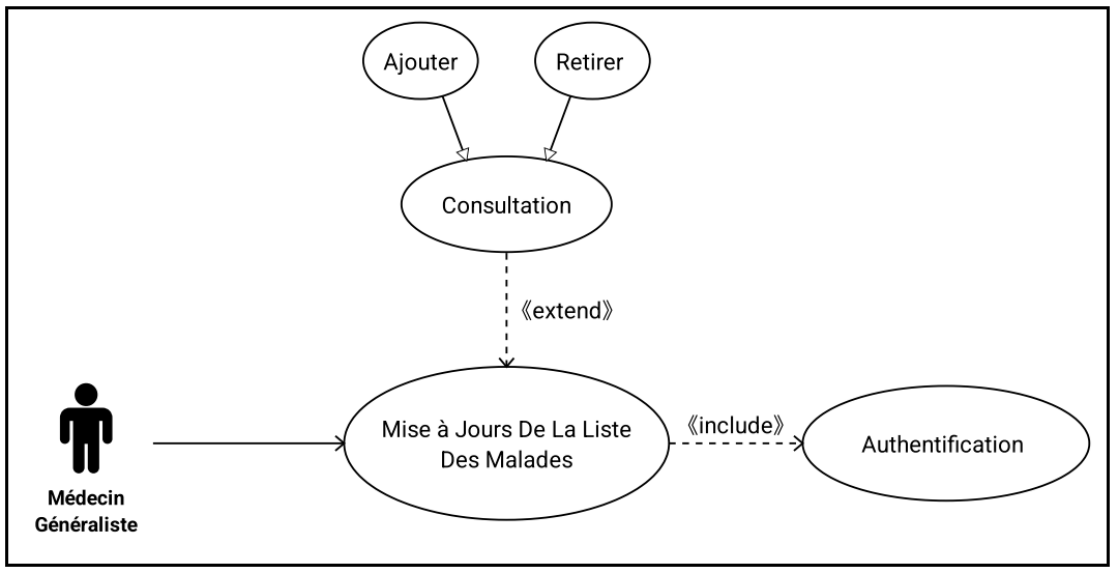


Figure 2.8 – Diagramme de cas d'utilisation 'médecin généraliste (mise à jour de la liste des malades)'.

2.8.8 Diagramme de cas d'utilisation 'médecin généraliste (gestion des dossiers médicaux)'

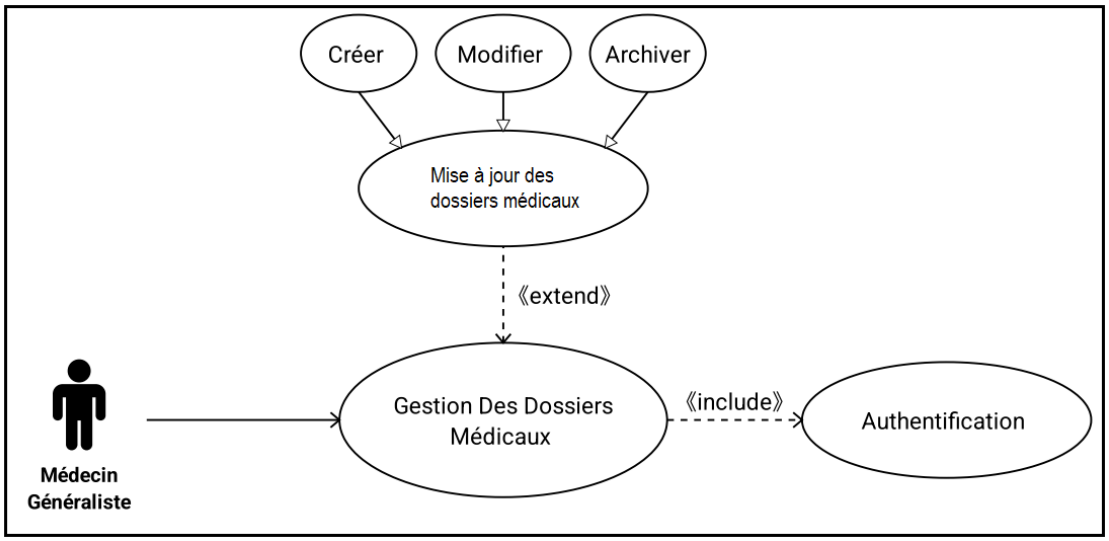


Figure 2.9 – Diagramme de cas d'utilisation 'Médecin généraliste (gestion des dossiers médicaux)'.

2.8.9 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (global)'

Le pneumologue et l'infectiologue ont pour rôle de consulter les dossiers médicaux, gestion d'hospitalisation, et la gestion des scanners et bilans, comme décrit ci-dessous :

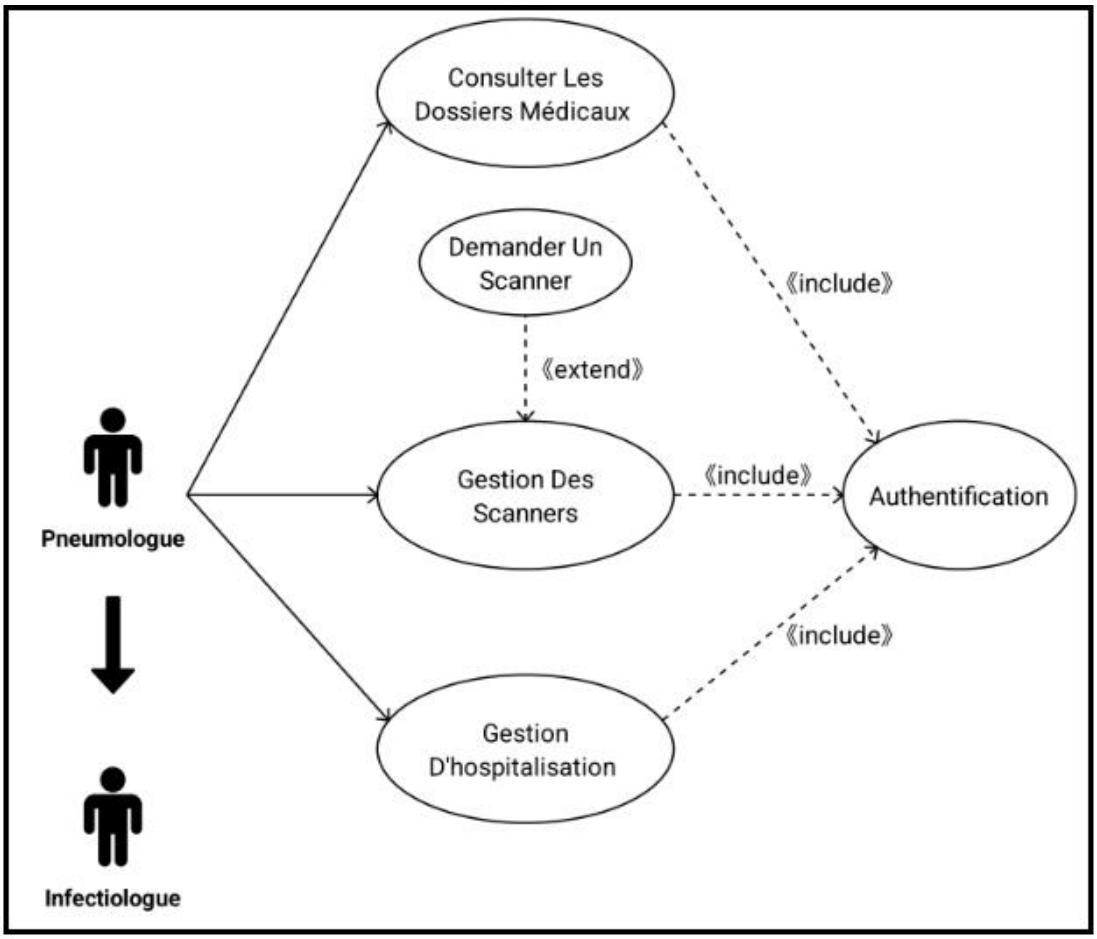


Figure 2.10 – Diagramme de cas d'utilisation 'Pneumologue/infectiologue (global)'.

2.8.10 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (gestion d'hospitalisatio)'

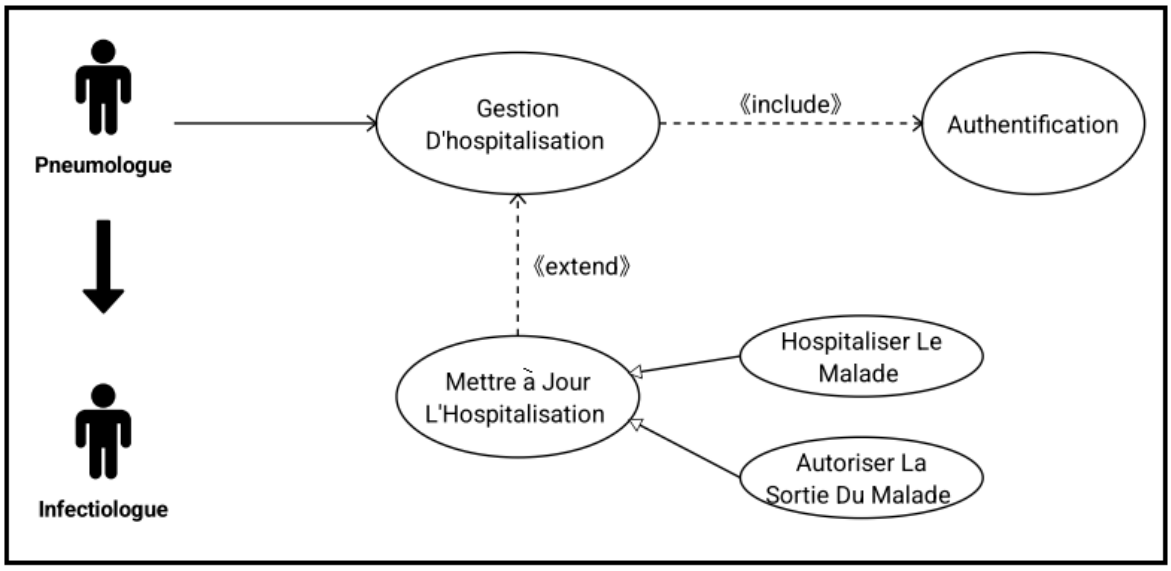


Figure 2.11 – Diagramme de cas d'utilisation 'Pneumologue/infectiologue (gestion d'hospitalisation)'.

2.8.11 Diagramme de cas d'utilisation 'Pneumologue/infectiologue (gestion des dossiers médicaux)'

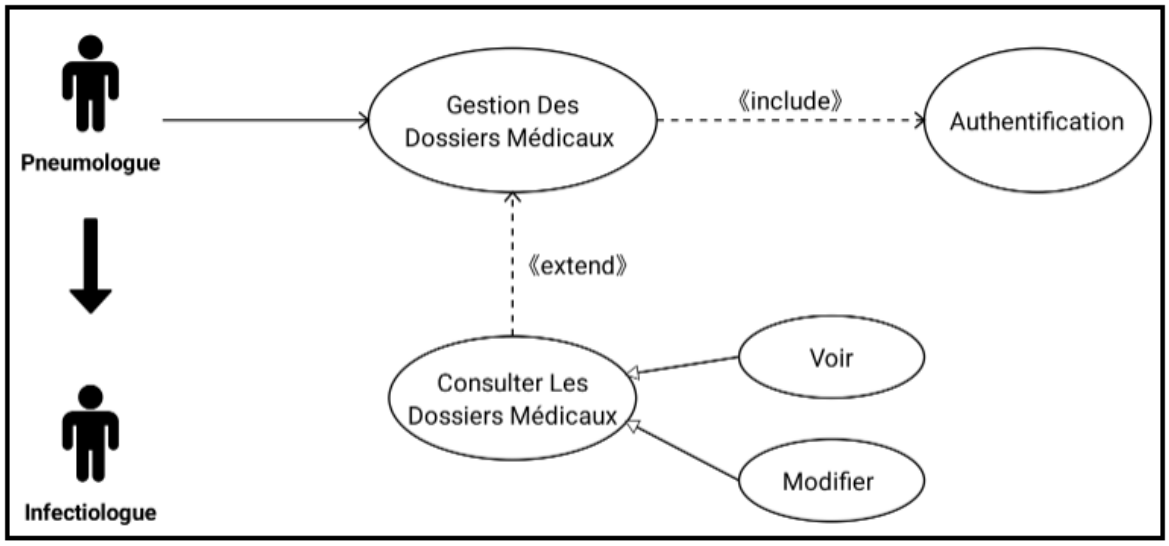


Figure 2.12 – Diagramme de cas d'utilisation 'Pneumologue infectiologue (gestion des dossiers médicaux)'.

2.8.12 Diagramme de cas d'utilisation 'Médecin spécialiste'

Le médecin spécialiste a pour rôle de consulter les dossiers médicaux et la liste des consultations médicales du malade, la gestion des soins et la sortie du malade.

La figure ci-dessous représente le diagramme de cas d'utilisation globale du médecin spécialiste Covid-19 :

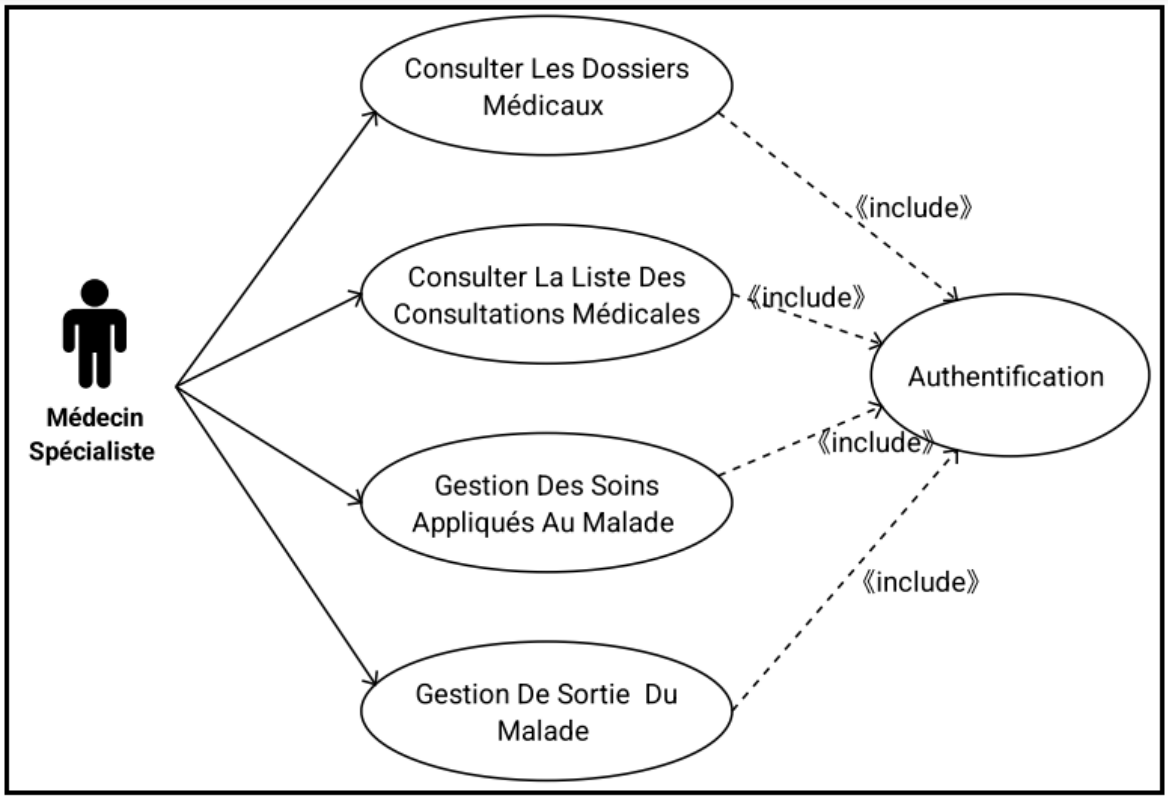


Figure 2.13 – Diagramme de cas d'utilisation 'Médecin spécialiste'.

2.8.13 Diagramme de cas d'utilisation 'médecin spécialiste (consultation des dossiers médicaux)'

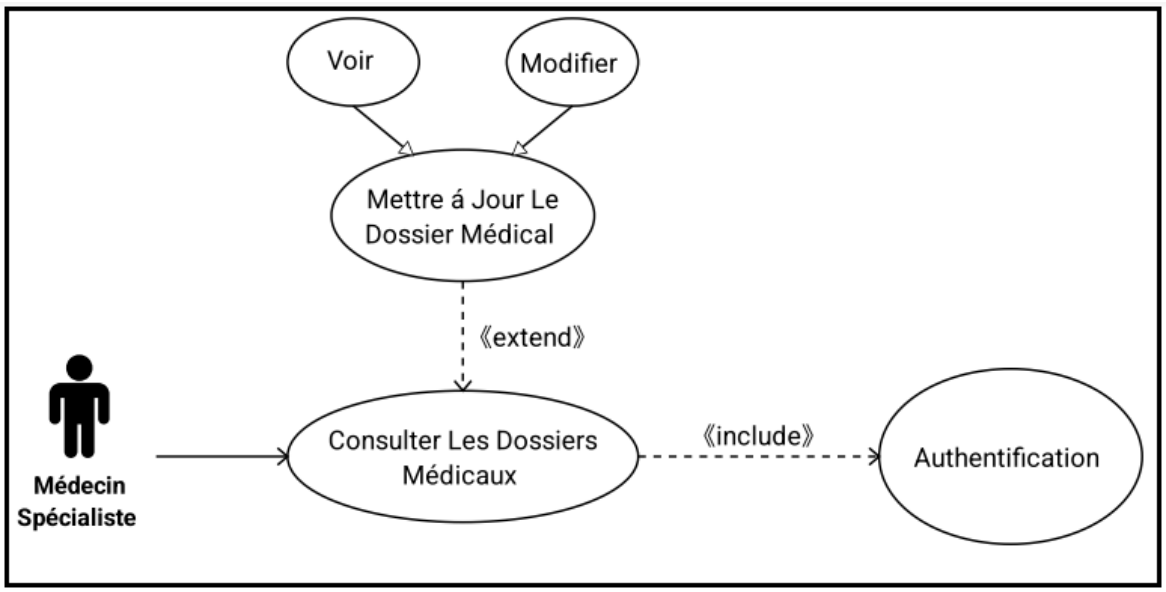


Figure 2.14 – Diagramme de cas d'utilisation 'médecin spécialiste (consultation des dossiers médicaux)'.

2.8.14 Diagramme de cas d'utilisation 'Médecin spécialiste (gestion de sortie)'

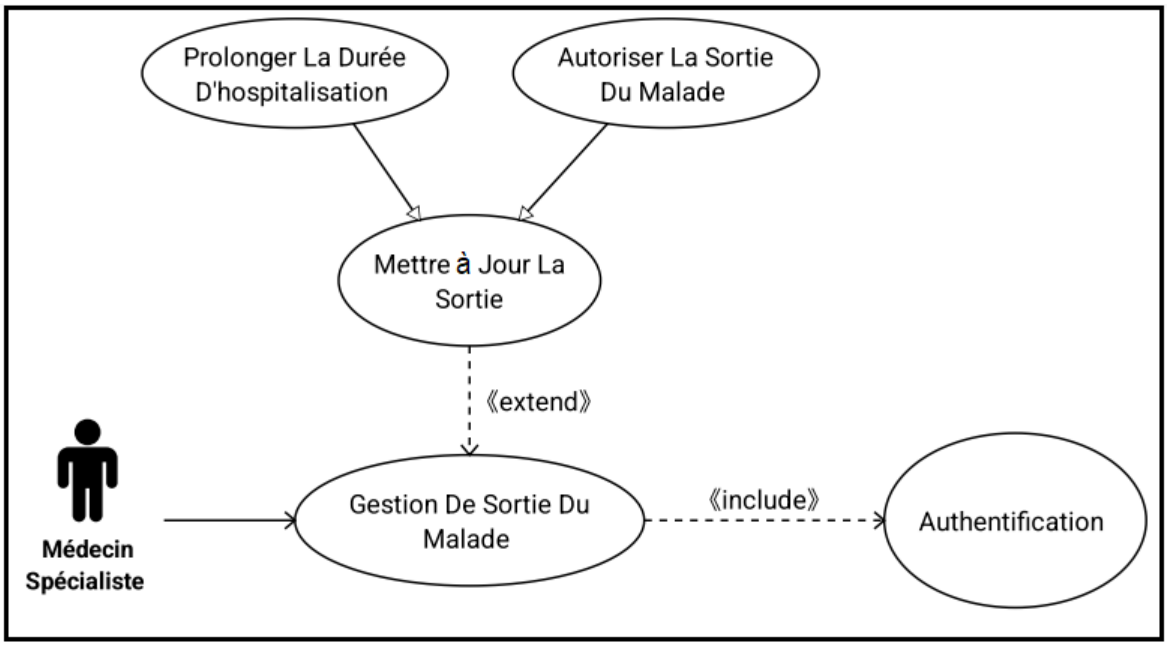


Figure 2.15 – Diagramme de cas d'utilisation 'médecin spécialiste (gestion de sortie)'.

2.8.15 Diagramme de cas d'utilisation 'infirmier'

L'infirmier a pour rôle de consulter les dossiers médicaux pour appliquer les soins nécessaires au malades, comme décrit ci-dessous :

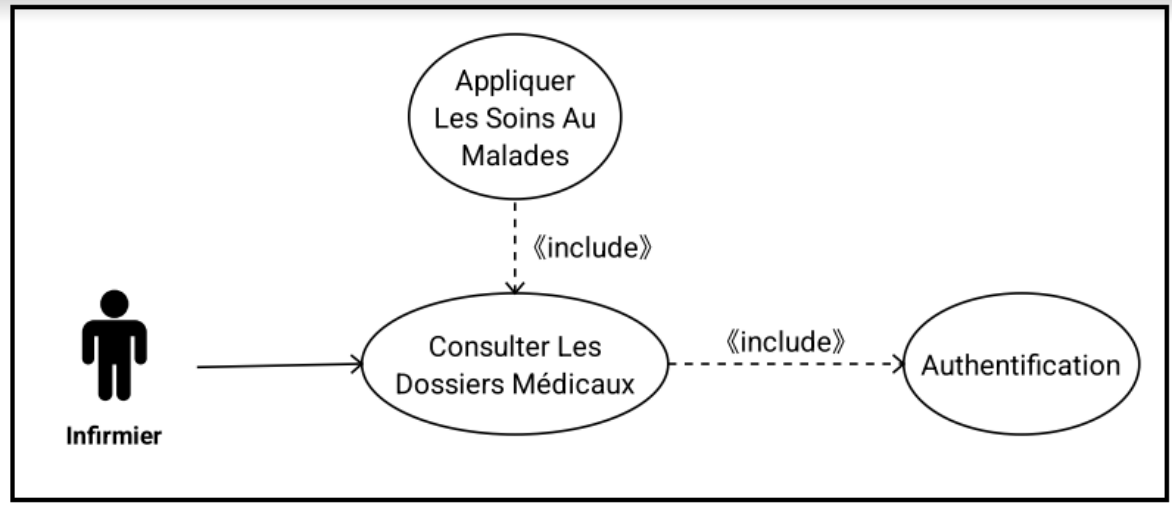


Figure 2.16 – Diagramme de cas d'utilisation 'Infirmier'.

2.9 Description de la vue dynamique (Diagramme de séquence)

La description de la vue dynamique de notre application est réalisée à travers les différents diagrammes de séquence :

2.9.1 Diagramme de séquence "Authentification"

Dans ce diagramme l'utilisateur a un nom d'utilisateur et un mot de passe qui va l'entrée dans les systèmes qui va vérifier ce dernier, afin de laisser l'utilisateur accéder à son interface ou l'empêcher dans le cas d'erreur.

Le diagramme ci-dessous présente l'enchaînement de la phase "Authentification" :

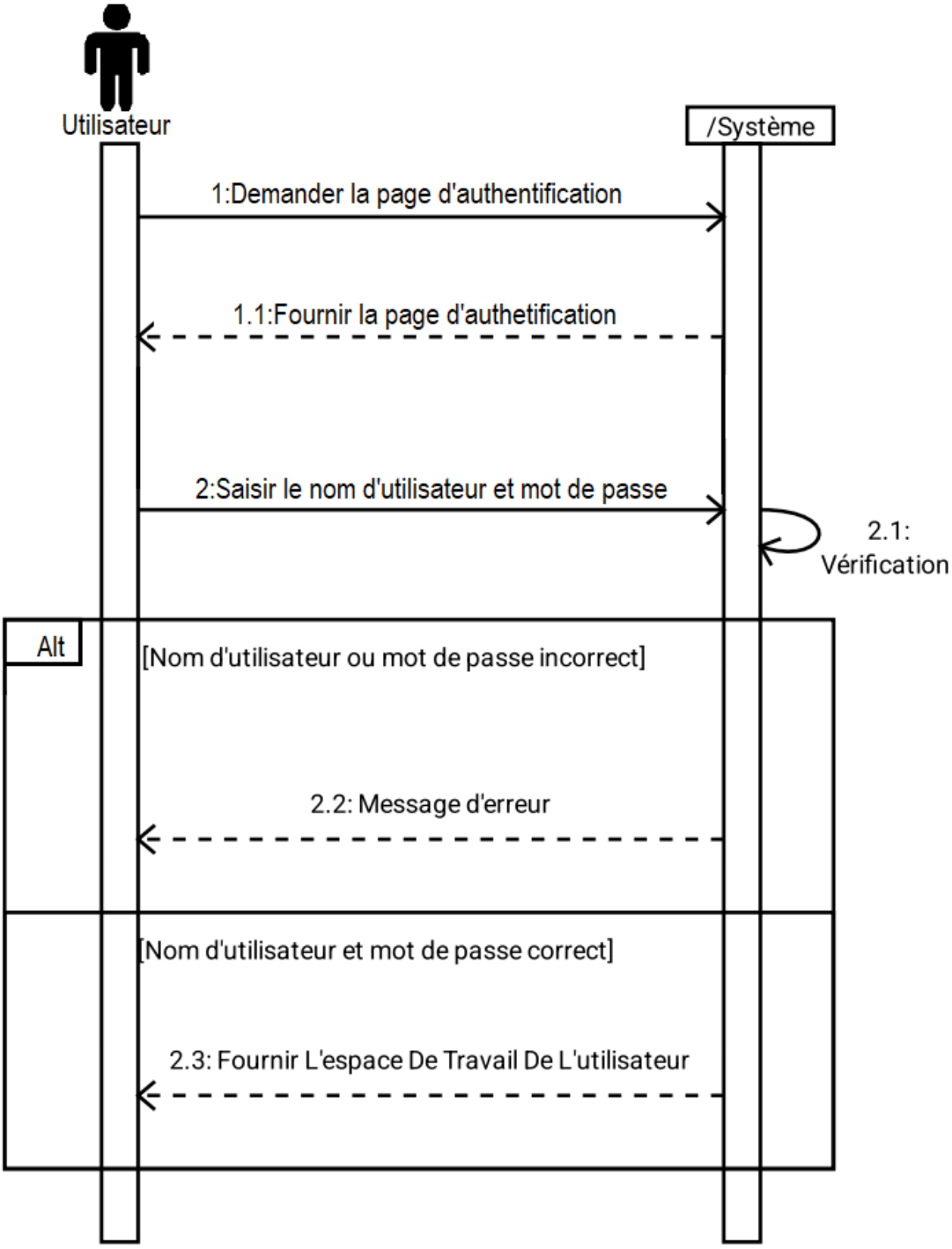


Figure 2.17 – Diagramme de séquence ‘Authentification‘.

2.9.2 Diagramme de séquence ” Gestion du dossier médical par un chef de service”

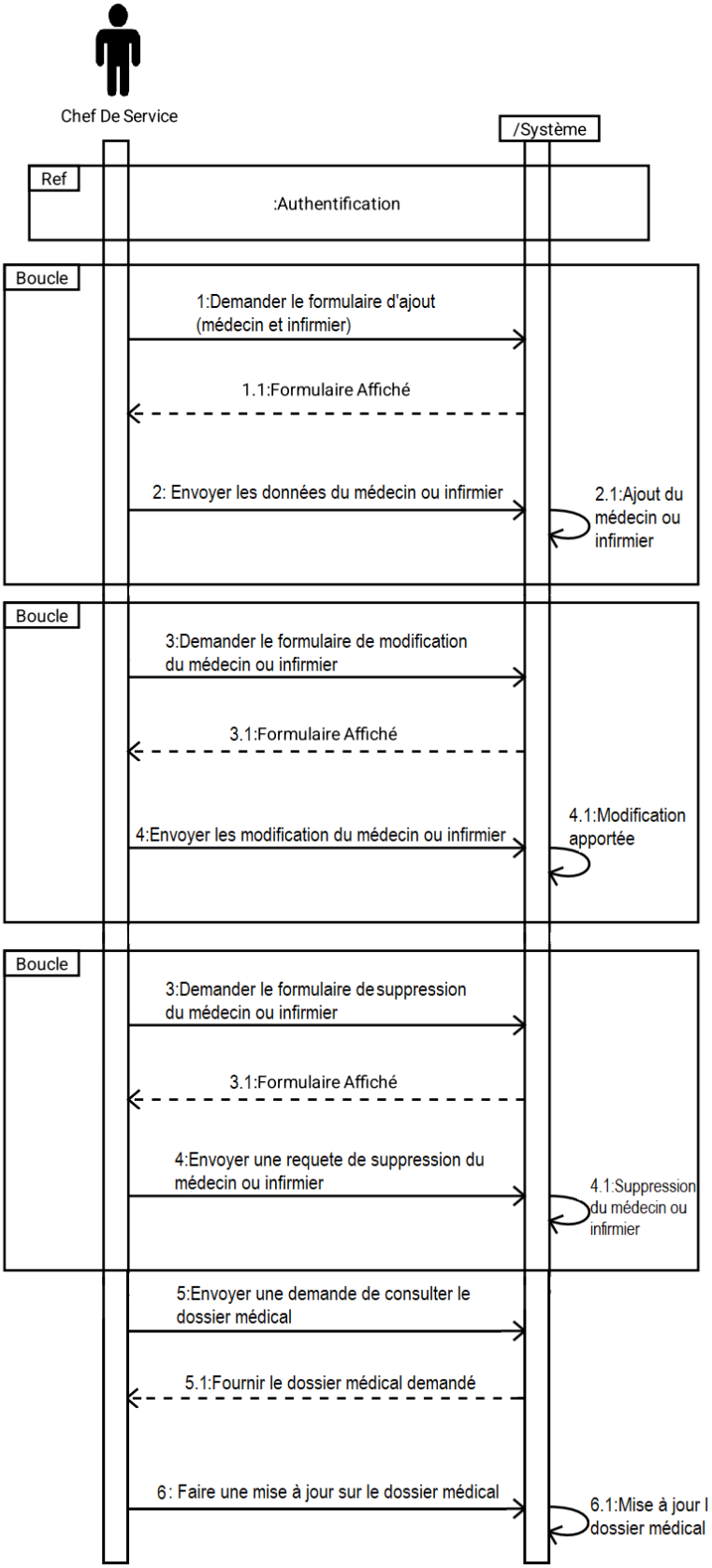


Figure 2.18 – Diagramme de séquence ‘Gestion du dossier médical par chef de service‘.

2.9.3 Diagramme de séquence ”Gestion du dossier médical par un médecin généraliste”

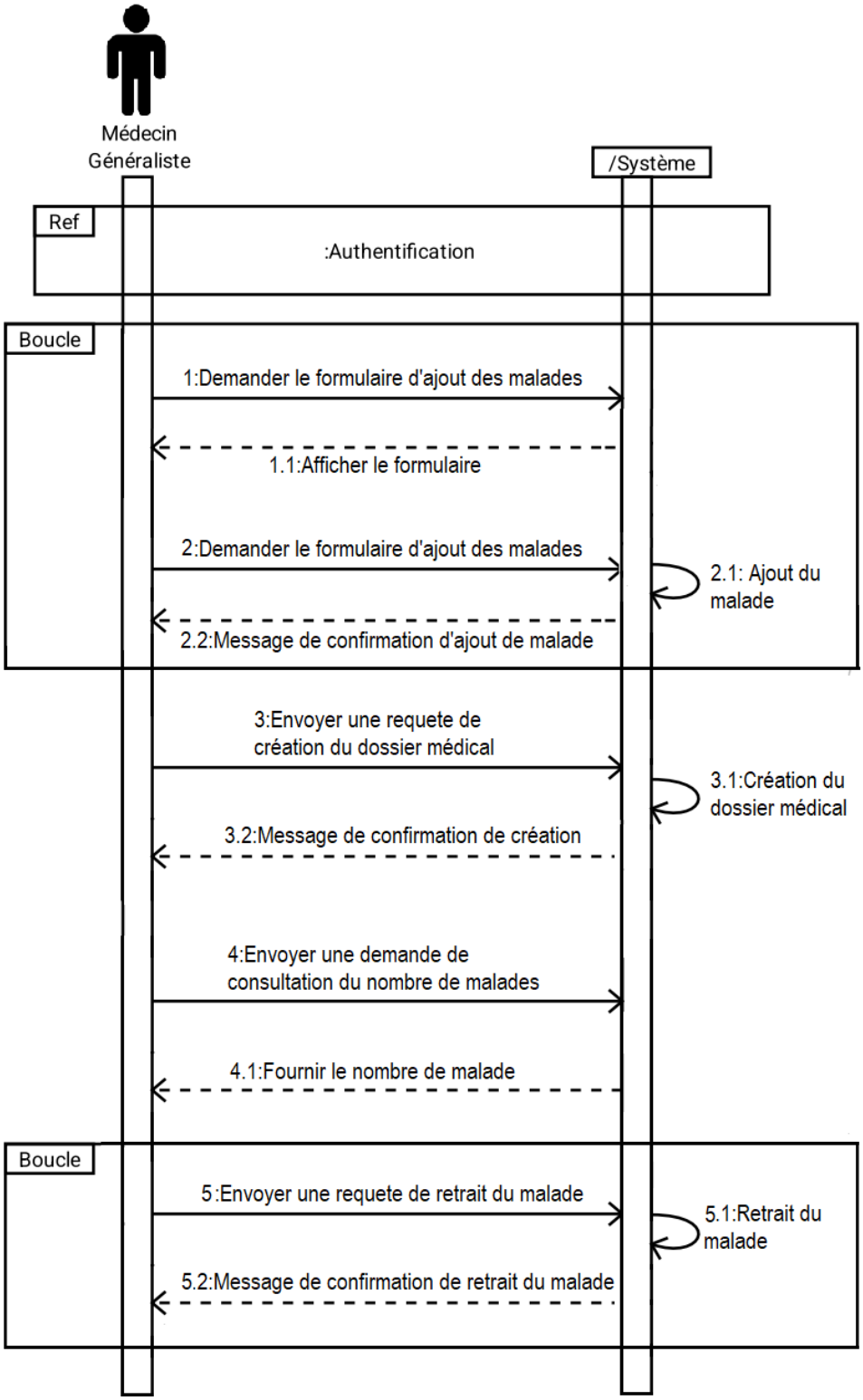


Figure 2.19 – Diagramme de séquence ‘Gestion du dossier médical par médecin généraliste‘.

2.9.4 Diagramme de séquence “Gestion du dossier médical par un infirmier“

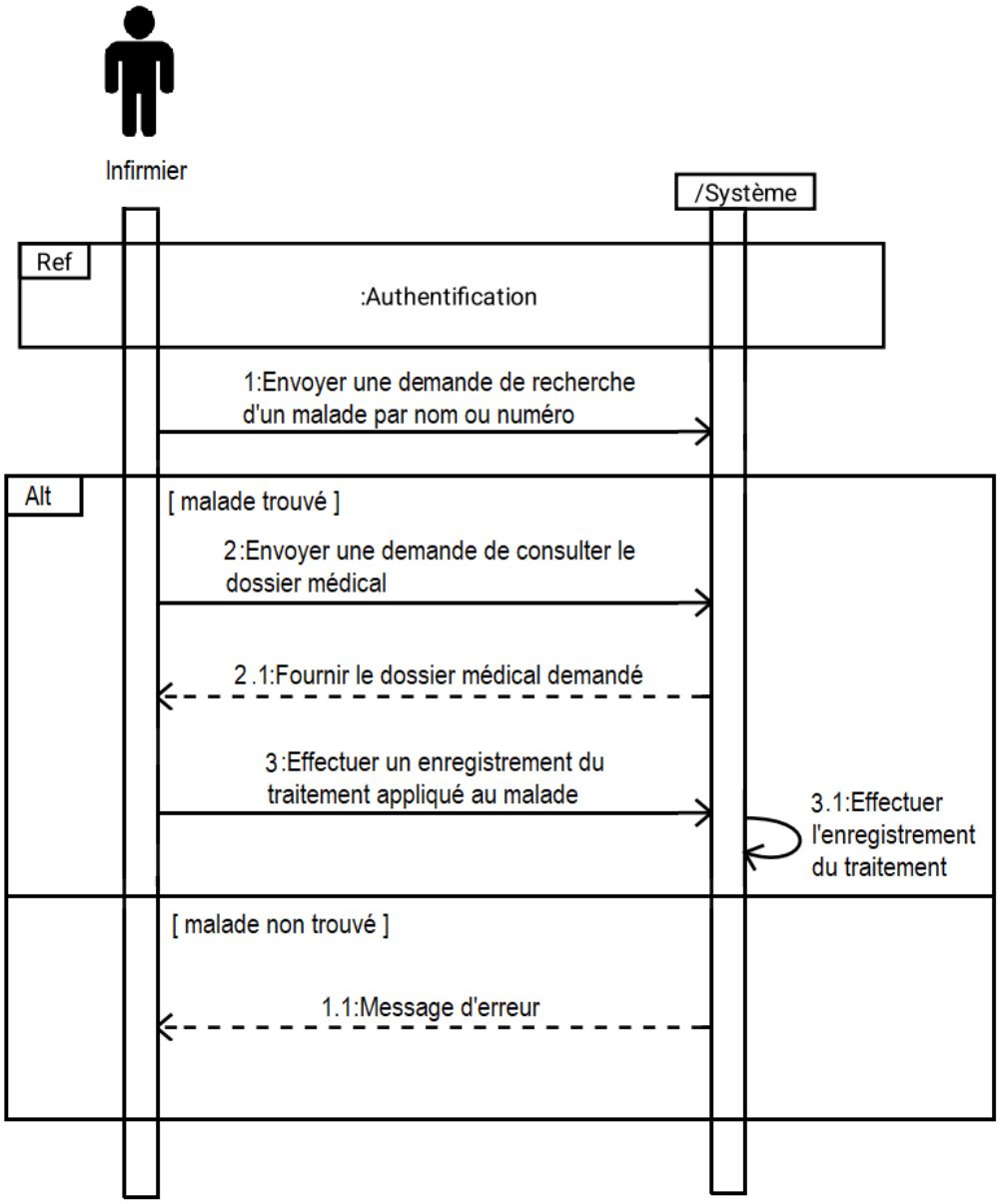


Figure 2.20 – Diagramme de séquence ‘Gestion du dossier médical par l’infirmier‘.

2.9.5 Diagramme de séquence “Gestion du dossier médical par un spécialiste Covid -19“

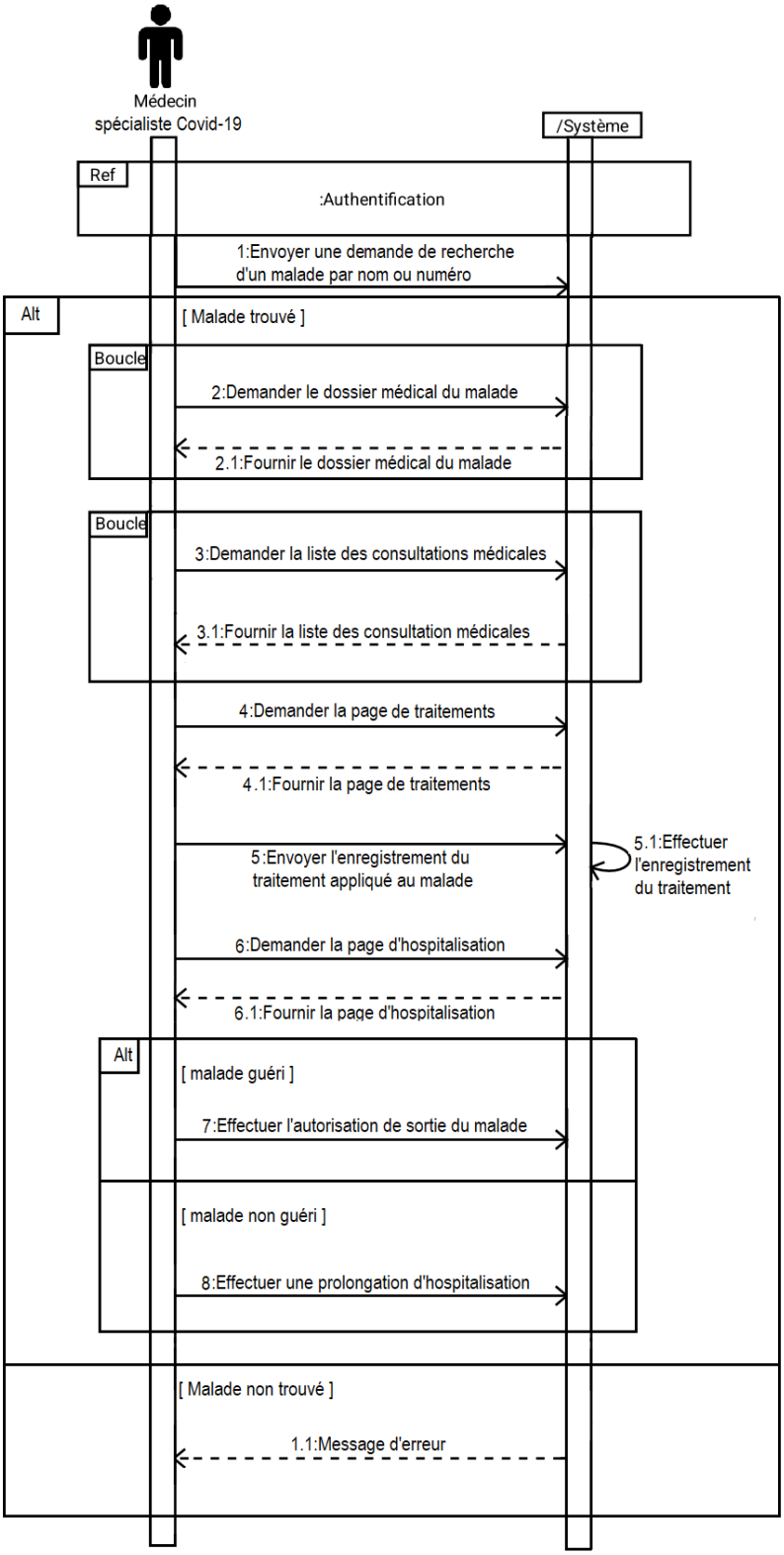


Figure 2.21 – Diagramme de séquence ‘Gestion du dossier médical par spécialiste Covid-19‘.

2.10 Description de la vue statique (Diagramme de classes)

Le diagramme de classes est considéré comme le plus important de la modélisation orienté objet, il s'agit d'une vue statique du fait qu'on ne tient pas compte du facteur temporel dans le comportement du système.

Le diagramme de classe permet de modéliser les classes du système et leurs relations indépendamment d'un langage de programmation particulier.

Dans notre cas, on a un diagramme de classe contient 14 classes.

Le diagramme de classe retenu au début de la conception est le suivant :

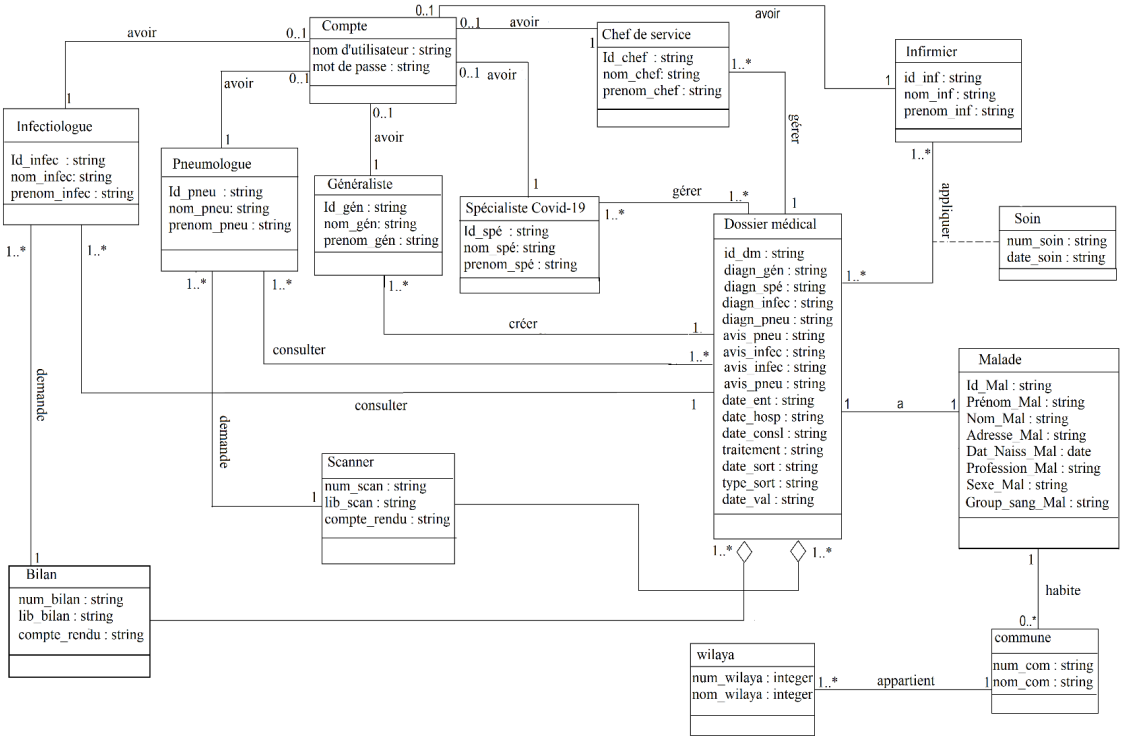


Figure 2.1 – Diagramme de classe.

Modèle relationnel

Note : nous avons utilisé le caractère "#" qui désigne les clés étrangères, et le soulignement qui désigne les clés primaires.

- **Dossier médical**(id_dm, diagn_gén, diagn_gén, diagn_gén, diagn_gén, diagn_gén, diagn_gén, diagn_gén, diagn_gén, date_ent, date_hosp, date_consl, traitement, date_sort, type_sort, date_val, #id_chef, #id_gén, #id_spé, #id_infec, #id_pneu, #id_inf, #id_mal, #num_scan, #num_bilan)
- **Compte** (nom_d'utilisateur, mot_de_passe)
- **Chef de service** (id_chef, nom_chef, prenom_chef)
- **Infectiologue** (id_infec, nom_infec, prenom_infec)
- **Pneumologue** (id_pneu, nom_pneu, prenom_pneu)

— **Généraliste** (id_gén, nom_gén, prenom_gén)
— **Spécialiste Covid-19** (id_spé, nom_spé, prenom_spé)
— **Infirmier** (id_inf, nom_inf, prenom_inf)
— **Bilan** (num_bilan, lib_bilan, compte_rendu, #id_infec)
— **Scanner** (num_scan, lib_scan, compte_rendu, #id_pneu)
— **Malade** (id_mal, prenom_mal, nom_mal, adresse_mal, dat_naiss_mal, profession_mal, sexe_mal, group_sang_mal)
— **Soin** (num_soin, date_soin, #id_inf)
— **Commune** (num_com, nom_com, #num_wilaya)
— **Wilaya** (num_wilaya, nom_wilaya)

Conclusion

L'activité de la phase analyse et conception a facilité la compréhension de notre système, qui ébauche vers l'activité d'implémentation ; et mettre le point sur les éléments importants du système à réaliser et de ce fait diminuer les risques et maintenir le projet à un niveau raisonnable afin d'entamer l'étape de développement sur de bonnes bases.

Chapitre 3

Réalisation

Introduction

Dans ce chapitre, nous présentons la partie réalisation qui est la phase finale de notre projet. Nous décrirons l'environnement ainsi que les outils qui nous ont servis pour la réalisation de l'application et nous terminerons enfin par la présentation de l'application.

Nous nous sommes appuyés sur différents langages pour l'implémentation, ceux qu'on va définir juste après.

3.1 Environnement Logiciel

3.1.1 Visual Studio Code

Visual Studio Code est un éditeur de code extensible développé par Microsoft pour Windows, Linux et MacOs, c'est un éditeur de code source qui peut être utilisé avec une variété de langages de programmation, notamment Java, JavaScript, Go, Node.js et C++.

Il est basé sur le framework Électron, qui est utilisé pour développer des applications Web Node Js.

Dans notre projet nous avons utilisé Visual Studio Code version 1.55.1 comme un éditeur de texte pour éditer du code PHP et html et sql, javascriptet MySQL. [1]

3.1.2 Wamp server (anciennement WAMP5)

Est une plateforme de développement Web de type WAMP, permettant de faire fonctionner localement (sans avoir à se connecter à un serveur externe) des scripts PHP.

WampServer n'est pas en soi un logiciel, mais un environnement comprenant deux serveurs (Apache et MySQL), un interpréteur de script (PHP), ainsi que phpMyAdmin pour l'administration Web des bases MySQL.

Il dispose d'une interface d'administration permettant de gérer et d'administrer ses serveurs au travers d'un tray icon (icône près de l'horloge de Windows).

La grande nouveauté de WampServer 2 réside dans la possibilité d'y installer et d'utiliser n'importe quelle version de PHP, Apache ou MySQL en un clic.

Ainsi, chaque développeur peut reproduire fidèlement son serveur de production sur sa machine locale. Nous avons utilisé Wamp Server dans notre projet version 3.2.3 pour exécuter les pages php dans ce serveur local. [1]

3.1.3 Langage HTML

Le HTML5, pour HyperText Markup Language 5, est une version du célèbre format HTML utilisé pour concevoir les sites Internet.

Celui-ci se résume à un langage de balisage qui sert à l'écriture de l'hypertexte indispensable à la mise en forme d'une page Web. [1]

Dans notre projet nous avons utilisé le langage html 5 comme langage de base pour une page web.

3.1.4 Langage PHP

Langage de programmation libre, principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP , mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale.

PHP est un langage impératif objet. PHP a permis de créer un grand nombre de sites web célèbres, comme Facebook et Wikipédia.

Dans notre projet, nous avons utilisé php version 7.3.21 pour rendre nos pages dynamiques, notamment aussi pour la connexion à la base de donné avec SQL.

3.1.5 Langage JavaScript

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web.

Avec les technologies HTML et CSS, JavaScript est parfois considéré comme l'une des technologies coeur du World Wide Web.

C'est un langage orienté objet à prototype : les bases du langage et ses principales interfaces sont fournies par des objets.

Il est aussi employé pour les serveurs avec l'utilisation (par exemple) de Node.js ou de Deno. [1]

3.1.6 Langage SQL

SQL est un langage informatique normalisé servant à exploiter des bases de données relationnelles.

La partie langage de manipulation des données de SQL permet de rechercher, d'ajouter, de modifier ou de supprimer des données dans les bases de données relationnelles. [1]

Dans notre projet nous avons utilisé le langage SQL pour la connexion avec la base de données.

3.1.7 Langage CSS

Le CSS pour Cascading Style Sheets, est un langage informatique utilisé sur Internet pour la mise en forme de fichiers et de pages HTML. On le traduit en français par feuilles de style en cascade. [1]

Dans notre projet nous avons utilisé css3 comme outil de design pour le front end des pages web.

3.2 Présentation de l'application

3.2.1 Conception de l'interface

Pour être utilisable, une interface devrait répondre à certains critères qui sont :

— Montrer clairement les fonctionnalités du système.
— Etre facile d'accès avec une navigation simple.
— Faciliter la tâche de l'utilisateur.
— Fournir une information que l'utilisateur final puisse la comprendre et l'interpréter.
— Permettre une saisie facile des données.

3.2.2 Architecture générale de la plateforme

Module d'authentification (Connexion)

Dans cet espace on a un formulaire d'authentification (connexion) pour tous les acteurs de cette plateforme, chacun son nom d'utilisateur et son mot de passe.



Figure 3.1 – Module d'authentification.

Module du médecin généraliste

Après que le médecin généraliste s'authentifie avec son nom d'utilisateur et son mot de passe, il accède à sa page principale dont il va ajouter les nouveaux malades consultés, en fournissant tous les informations du malade.



Figure 3.2 – Module du médecin généraliste.

Module du médecin pneumologue et infectiologue

Après que le médecin pneumologue et l'infectiologue s'authentifient avec leurs nom d'utilisateur et mot de passe, ils accèdent à leurs pages principales dont ils peuvent voir les statistiques des malades, et peuvent prendre la décision d'hospitaliser ou pas un malade.



Figure 3.3 – Module du médecin pneumologue.



Figure 3.4 – Module du médecin infectiologue.

Module du médecin spécialiste Covid-19

Après que le médecin spécialiste s'authentifie avec son nom d'utilisateur et son mot de passe, il accède à sa page principale dont il va voir tous les détails des malades (leur état de santé, statistique, …), il a le droit de transférer le malade au service de réanimation, et aussi le faire sortir s'il est guéri.



Figure 3.5 – Module du médecin spécialiste Covid-19.

Module du spécialiste Covid-19 pour le suivi du malade

Dans cette page le médecin spécialiste Covid-19 peut consulter le suivi du malade tous les jours, et il a le rôle de mettre à jour le dossier médical.

Comme présenté dans la figure ci-dessous :

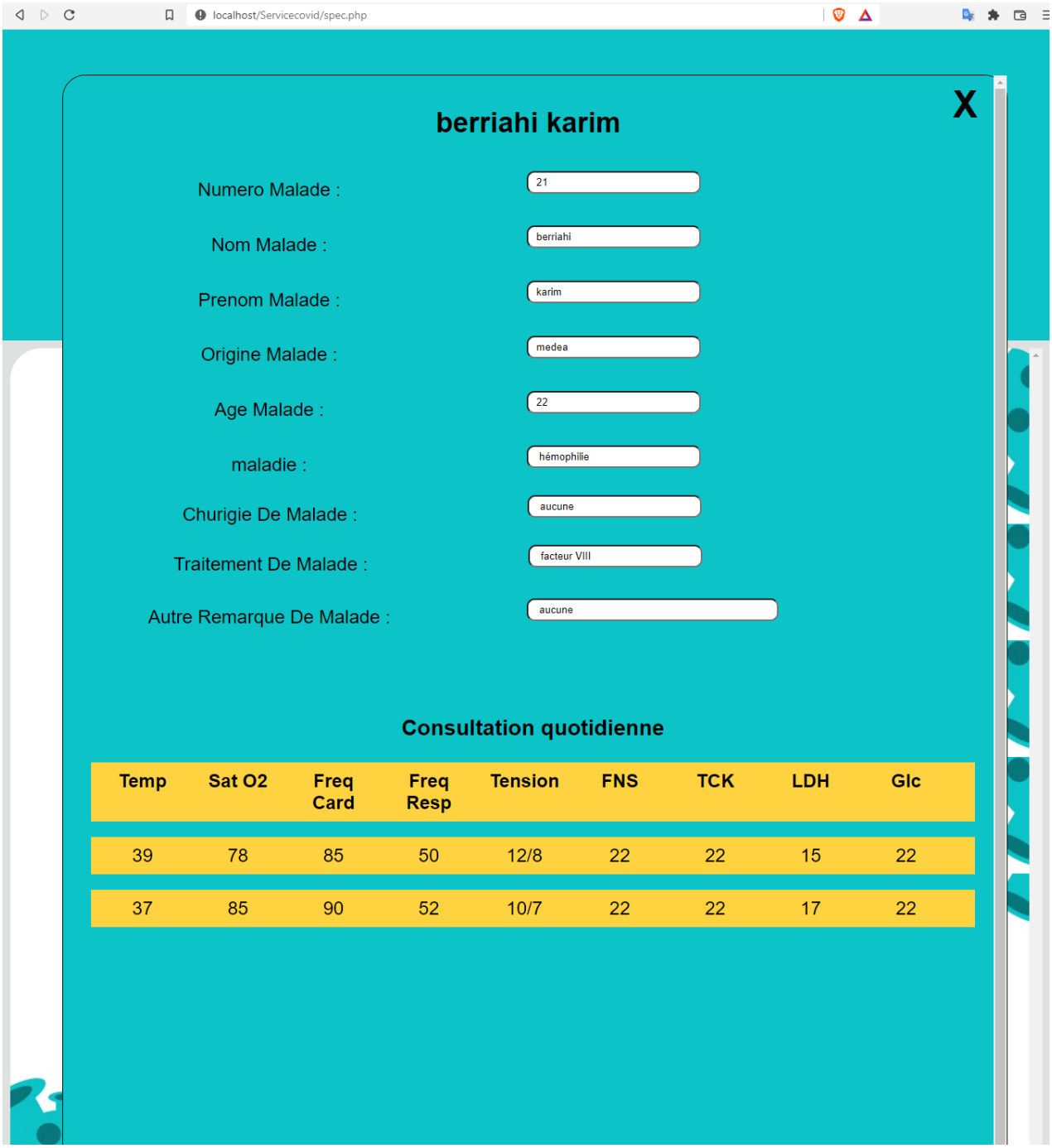


Figure 3.6 – Module du médecin spécialiste Covid-19 (suivi du malade).

3.3 Sécurisation l'application

La sécurité Web, la cyber-sécurité ou la sécurité des applications Web est le moyen de protéger les services en ligne et les sites Web contre diverses menaces exploitant les vulnérabilités associées aux codes d'une application. Certaines des cibles courantes de ces attaques sont les solutions de gestion de bases de données telles que phpMyAdmin.

La sécurité Web vise à empêcher de telles attaques en refusant l'accès, l'utilisation, la destruction / perturbation ou la modification non autorisés.

Pour assurer la sécurité de notre PWGF nous avons utilisé les méthodes suivantes :

— **Utiliser les sessions :** Nous utilisons les sessions pour obtenir la plus grande sécurité possible.

— **Méthode Post :** Cette méthode est plus sécurisée en PHP que la méthode GET, notamment pour les informations personnelles, privées et sensibles.

— **L'authentification :** Utilisation d'un nom utilisateur et un mot de passe pour accéder aux espaces des utilisateurs.

Conclusion

Tout au long du chapitre, nous avons essayé de donner une vision globale sur l'implémentation de notre plate-forme et cela par la présentation de quelques interfaces principales dont le rôle répond à un besoin parmi ceux qui ont été décrit à la phase analyse et conception.

Conclusion générale

Notre projet consistait à concevoir et réaliser une plateforme web pour la gestion des malades Covid-19 au sein d'hôpital.

Afin d'aboutir à notre objectif, nous avons effectué une recherche sur les principales notions des technologies web et les taches de chaque médecin (généraliste, pneumologue, infectiologue, spécialiste) dans son service dédié, la spécification des besoins au terme de laquelle nous avons proposé la plateforme en donnant les détails de sa conception et sa réalisation en optant principalement pour UML comme méthode de modélisation et PHP comme langage de programmation orienté objet ainsi que MySQL comme SGBD.

La plateforme web est robuste et efficace qui présente les points forts suivants :

— Assurer une gestion administrative en ligne sécurisant toutes les informations notamment celles relatives aux malades et aux médecins.

— Apporter des fonctionnalités : authentification, gestion et consultation de : dossiers médicaux, médecins, malades, résultats de test, etc.

— Nous estimons avoir atteint l'objectif initial tracé malgré les difficultés rencontrées que nous avons pu surmonter notamment la contrainte temps.

Nous espérons combler les lacunes qui existent et améliorer davantage la qualité de notre travail.

Enfin nous espérons que ce modeste travail sera est une référence pour d'autres projets de fin d'études.

Bibliographie

[**1**] *Wikipidia*, 2022. https://fr.wikipedia.org/wiki/.

[**2**] *Techno-science*, 2022. https://www.techno-science.net/definition/3743.html.

[**3**] Cay S Horstmann and Gary Cornell. *Au cœur de Java 2*. CampusPress, 2000.

[**4**] Patrice Silverio Roger Engel. *Applications Cobol sur le Web : avec une étude de cas détaillée*. Edition Eyrolles,, 2000.

[**5**] BOUTRID SAMIA BECHAR SABRINA. *Développement d'une plateforme d'interchange*. 2004-2005.

[**6**] Zineb Bakraouy, Amine Baina, and Mostafa Bellafkih. Availability of web services based on autonomous classification and negotiation of slas. In *2018 6th International Conference on Multimedia Computing and Systems (ICMCS)*, pages 1–6. IEEE, 2018.

[**7**] Djilali Idoughi and Christophe Kolski. Vers un développement orienté services des applications interactives dans le domaine de la logistique, étude de cas. In *Workshop International IEEE Logistique et Transport*, 2009.

[**8**] *«simplebo»*, 2022. https://blog.simplebo.fr/comment-fonctionne-hebergementsite-internet.

[**9**] Pascal Roques and Franck Vallée. Uml 2 en action. *De l'analyse des besoins à la conception J2EE, 3ème édition Eyrolles*, 2004.

[**10**] *«lri»*, 2022. https://www.lri.fr/~longuet/Enseignements/16-17/Et3-UML/Et3-3DiagClasses.pdf.