UNIVERSITE DE YAOUNDE I
****
ECOLE NATIONALE SUPERIEURE POLYTECHNIQUE DE YAOUNDE
****
DEPARTEMENT DE GENIE INFORMATIQUE

UNIVERSITY OF YAOUNDE I
****
NATIONAL ADVANCED SCHOOL OF ENGINEERING OF YAOUNDE
****
DEPARTMENT OF COMPUTER ENGINEERING

MISE EN OEUVRE D'UN SYSTÈME DE MESSAGERIE INSTANTANÉ MOBILE POUR UNE PLATEFORME DE MARKETPLACE

Mémoire de fin d'études

Présenté et soutenu par

SONDI MANGA JEFRIED HUGO

En vue de l'obtention du :

Diplôme d'Ingénieur de Conception en Génie Informatique

Sous la Supervision de :

ANNE MARIE CHANA, CHARGÉ DE COURS, UY1,

THOMAS DJOTIO, MAÎTRE DE CONFERENCE, UY1

Devant le jury composé de :

Président : **ETIENNE TAKOU, MAÎTRE DE CONFÉRENCES, UYI**

Rapporteur : **ANNE MARIE CHANA, CHARGÉ DE COURS, UY1**

Examinateur : **GEORGES EDOUARD KOUAMOU, MAÎTRE DE CONFÉRENCES, UYI**

Invité : **THOMAS DJOTIO, MAÎTRE DE CONFÉRENCES, UYI**

Année Académique 2020-2021
Mémoire soutenu le 04 octobre 2021

MISE EN OEUVRE D'UN SYSTÈME DE MESSAGERIE INSTANTANÉ MOBILE POUR UNE PLATEFORME DE MARKETPLACE

Mémoire de fin d'études

Présenté et soutenu par

SONDI MANGA JEFRIED HUGO

En vue de l'obtention du :

Diplôme d'Ingénieur de Conception en Génie Informatique

Année académique 2020-2021
Mémoire soutenu le 04 octobre 2021

Dédicace

A ma famille

REMERCIEMENTS

Ce travail n'aurait pas été possible sans les conseils et l'aide d'un certain nombre de personnes, à qui vont mes remerciements les plus sincères, à savoir :

— **Pr Etienne TAKOU**, pour l'honneur qu'il me fait en présidant ce jury ;

— **Pr Georges Edouard KOUAMOU**, pour avoir accepté d'examiner ce travail ;

— **Dr Anne Marie CHANA**, mon encadreur académique, pour sa disponibilité, ses remarques, ses conseils et l'attention qu'elle a accordé à la redaction de ce mémoire ;

— **Pr Thomas DJOTIO NDIE**, mon encadreur professionnel, pour sa disponibilité, pour ses conseils et le suivi de ce travail ;

— **L'ingénieur Jean Yves ETOUGUE**, pour sa disponibilité, ses conseils et les bons moments partagés ;

— **L'ingénieur Kollo Ntengue**, pour tous ses conseils et son soutien ;

— Le corps enseignant de l'Ecole Nationale Supérieure Polytechnique de Yaoundé, en particulier celui du département du Genie Informatique, pour tous les cours dispensés et leur dévouement à la formation des futurs ingénieurs ;

— Toute ma famille, qui m'a accordé soutien et attention et particulièrement mes parents **M. MANGA Thomas** et **Mme. MANGA Eveline**, pour leurs prières et sacrifices sans lesquels rien n'aurait été possible ;

— Mes amis et camarades **Nick TEDONZE**, **Simon MENGONG**, **Luc BLONDEAU**, **Manuela KAMNO**, **Amael BOGNE**, **Pierre WOLOUNWO**, pour leur soutien et leur disponibilité ;

— Tous mes camarades de promotion et en particulier du Genie Informatique 2021 ;

— Tous mes amis et mes proches pour leur attention et leur soutien ;

— Tous ceux qui de près ou de loin m'ont soutenu.

RÉSUMÉ

La communication est un aspect important dans la prospérité de toute activité mettant en relation plusieurs acteurs. Pour les places de marché en ligne il est aussi important d'avoir une bonne communication entre les fournisseurs et les consommateurs de la plateforme. Yowyob Inc. Ltd est un nouvel acteur du e-commerce au Cameroun qui met à la disposition de ses clients un ensemble d'applications qui s'organisent autour d'un but à savoir fournir la meilleure expérience utilisateur en ce qui concerne le commerce de produits et services en ligne. De façon générale, on observe que 53% des clients ont un penchant pour les entreprises joignables via des applications de messagerie instantanée, dans le but de réaliser un achat de produit ou de services et que 63% des clients se disent plus enclins à revenir sur une boutique en ligne qui propose des services de messagerie instantanée. A cet effet la messagerie instantanée est depuis quelques années l'outil de communication le plus adapté et adopté par les utilisateurs. L'entreprise Yowyob .inc LTD souhaite mettre sur pied une application mobile de messagerie instantanée pour assurer la communication entre les différents acteurs de sa plateforme. Ce travail consiste en la mise en oeuvre d'une application mobile de messagerie instantanée et la problématique qui se dégage est de savoir comment mettre sur pied un système d'échange temps réel et sécurisé de messages et de fichiers aux formats variés entre plusieurs utilisateurs. Pour répondre à notre problème nous avons d'une part établi une architecture physique et logicielle adaptée au système de communication à mettre en oeuvre et qui basée sur un ensemble de protocoles de communication et d'autre part nous avons décrit la solution proposée à travers des diagrammes couvrant les phases d'analyses et de conception. Le modèle en V a été utilisé comme modèle de développement et le résultat obtenu est une application mobile multiplateforme de messagerie instantanée basée sur les protocoles Websocket, HTTP, STOMP et Signal.

Mots-clés : communication, marketplace, messagerie instantanée, application mobile.

Abstract

Communication is an important aspect in the success of any business that brings together several players. For online marketplaces it is also important to have a good communication between the suppliers and the consumers of the platform. The online marketPlace system Yowyob is a new e-commerce player in Cameroon that provides its customers with a set of applications that are organized around the goal of providing the best user experience in trading products and services online. In general, we observe that 53% of customers are inclined to use instant messaging applications to purchase products or services and that 63% of customers are more likely to return to an online store that offers instant messaging services. In this respect, instant messaging has been the most adapted and adopted communication tool by users for several years. The company Yowyob .inc LTD wishes to set up a mobile application of instant messaging to ensure the communication between the various actors of its platform. This work consists in the implementation of a mobile communication system and the problem that emerges is how to set up a system of real time and secure exchange of messages and files in various formats between several users. To answer our problem we have established a physical and software architecture adapted to the communication system to be implemented and which is based on a set of communication protocols and we have described the proposed solution through diagrams covering the analysis and design phases. The result is a cross-platform mobile instant messaging application based on Websocket, HTTP, STOMP and Signal protocols.

Keywords : communication, marketPlace, instant messaging, mobile application.

TABLE DES MATIÈRES

ABRÉVIATIONS

API Application Programming Interface. 1, 48

CQL Cassandra Query Language. 1

DCU Diagramme de Cas d'Utilisation. xi, 1, 33

ENSPY Ecole Nationale Supérieure Polytechnique Yaounde/National Advanced School of Engineering of Yaounde. 1

HTTP HyperText Transfert Protocol. 1, 9

IDE Integrated Development Environment. 1, 61

MVC Modèle Vue Contrôleur. 1

OSI Open Systems Interconnection. 1, 9

SMS Short Message Service. 1, 32

STOMP Streaming Text Oriented Messaging Protocol. 1, 9

UML Unified Modeling Language. 1

URL Uniform Ressource Locator. 1

UY1 University of Yaounde I. 1

GLOSSAIRE

API (Application Programming Interface), c'est un ensemble de définitions et de protocoles qui facilitent la création et l'intégration de logiciels d'applications. 1

Framework est un ensemble d'outils et de composants logiciels organisés conformément à un plan d'architecture et des patterns, l'ensemble formant ou promouvant un « squelette » de programme, un canevas. 1

Logiciel libre est un logiciel dont l'utilisation, l'étude, la modification et la duplication par autrui en vue de sa diffusion sont permises, techniquement et juridiquement. 1

Logiciel propriétaire est un logiciel qui ne permet pas légalement ou techniquement, ou par quelque autre moyen que ce soit, d'exercer simultanément les quatre libertés logicielles que sont l'exécution du logiciel pour tout type d'utilisation, l'étude de son code source (et donc l'accès à ce code source), la distribution de copies, ainsi que la modification et donc l'amélioration du code source. 1

MarketPlace est un site internet sur lequel des vendeurs indépendants, professionnels ou particuliers, ont la possibilité de vendre leurs produits ou services en ligne moyennant, pour les cas les plus connus, une commission prélevée par le site sur chaque vente. 1

OTP (One Time Pass) c'est un mot de passe à usage unique c'est à dire un mot de passe qui n'est valable que pour une session ou une transaction. 1

Protocole de communication est un ensemble de règles et de codes de langage qui définissent comment se déroule la communication entre un émetteur et un récepteur.. 1

Protocole sans état est un protocole de communication qui n'enregistre pas l'état d'une session de communication entre deux requêtes consécutives. 1

SMS (Short Message Service) est un système de téléphonie mobile permettant d'envoyer un message de 160 caractères maximum. 1

Système temps réel est un système capable de contrôler ou piloter un procédé physique à une vitesse adaptée à l'évolution du procédé contrôlé. 1

Liste des tableaux

TABLE DES FIGURES

INTRODUCTION GÉNÉRALE

CONTEXTE

Les MarketPlaces sont des plateformes en ligne qui mettent à la disposition des clients, des services ou produits provenants de plusieurs fournisseurs. Yowyob Inc. LTD est une entreprise qui propose un écosystème d'applications destinées à la mise sur pied d'une place de marché en ligne dans laquelle fournisseurs et consommateurs échangent des services. L'écosystème Yowyob se constitue des applications **Yowyob Shopping** pour la gestion des achats côté client, **Yowyob shop** pour la gestion des boutiques de fournisseur, **une application d'authentification** pour s'assurer de l'identité des utilisateurs, **une application de livraison** qui gère toutes les modalités entrant dans le processus de livraison et enfin **une application de Tracking** qui permet à chaque client de voir l'évolution de sa commande. Par cet ensemble d'applications, le système de marketplace en ligne veut reproduire au mieux les étapes qui entrent dans le processus de vente d'un produit du fournisseur au client, tel que cela se fait dans la réalité. Pour se faire, le système doit prendre en compte l'aspect négociation qui est une des phases du processus d'achat d'un produit par un client. Consciente de l'importance de l'aspect de communication dans un écosystème de marketplace en ligne, l'entreprise Yowyob Inc. LTD à travers la conception d'une application mobile de messagerie instantannée prévoit d'assurer les fonctions de conversation et de négociation entre le client et le fournisseur. Il s'agit principalement d'offrir aux acteurs de la MarketPlace en ligne (fournisseurs et consommateurs de services) un système de messagerie instantanée mobile pouvant leur permettre de communiquer en toute confiance. Ledit système doit permettre des conversations avec messages sous formats texte, image et vidéo. Le système doit permettre la constitution des groupes d'utilisateurs pour des conversations de groupe. Il doit permettre aux utilisateurs de publier des services à travers des statuts.

PROBLEMATIQUE

La problématique principale qui se dégage de la mise en oeuvre de l'application mobile de messagerie instantannée est la suivante :

Comment mettre en place un modèle d'échange temp réel et sécurisé de messages aux formats variés premièrement entre plusieurs utilisateurs en minimisant la consommation de données et la perte de messages lors de la transmission ?

De notre problématique découle quatre questions essentielles qui sont :

— Comment échanger un message entre deux utilisateurs connectés ?

— Comment échanger un message d'un utilisateur connecté vers un utilisateur non connecté ?

— Comment se passe l'échange de fichiers entre utilisateurs ?

— Comment garantir la confidentialité d'un message lors de sa transmission ?

OBJECTIF

L'objectif principal de ce travail est d'assurer la communication entre les acteurs d'une MarketPlace à travers la mise en oeuvre d'une application mobile de messagerie instantannée. Plus spécifiquement, il est question de :

1. Produire une architecture physique et logicielle de l'application de messagerie instantanée ;
2. Garantir la persistance des données en locale ;
3. Garantir la persistance des messages pour les utilisateurs non connectés ;
4. Garantir la sécurité des messages et des comptes utilisateurs ;
5. Développer le Front-end de l'application mobile avec des interfaces attrayantes et intuitives ;
6. Développer le Back-end du serveur de messagerie ;
7. Relier le Front-end et le Back-end de l'application ;
8. Mettre sur pied une documentation détaillée du projet.

PLAN DU MÉMOIRE

La suite de ce mémoire est structurée comme suit :

— **Chapitre 1 : Généralités et état de l'art** qui a pour objectif de présenter les différentes notions utiles pour la conception de notre solution de messagerie instantannée et les travaux déjà entrepris dans le sens de la résolution de notre problème.

— **Chapitre 2 : Méthodologie** qui présente la démarche que nous avons adoptée pour atteindre les objectifs fixés et résourdre le problème posé.

— **Chapitre 3 : Implémentation et résultats**, présente comment le travail a été réalisé, les outils utilisés ainsi que les résultats obtenus.

— **Conclusion générale et perspectives** qui présente le bilan du travail , dégage les limites et quelques perspectives futures.

CHAPITRE 1

GÉNÉRALITÉS ET ÉTAT DE L'ART

Dans ce chapitre, nous présentons premièrement les concepts généraux liés aux appilications mobiles et à la messagerie instantannée. Ensuite nous présentons quelques approches existantes pour la mise en oeuvre des applications mobile de messagerie instantanée ainsi que quelques solutions de messagerie instantanée existantes. Enfin nous présentons notre positionnement suite à l'étude de ces derniers.

1.1 Application mobile

1.1.1 Qu'est-ce qu'une application mobile ?

Une application est un programme informatique écrit en vue d'une utilisation précise(calcul, gestion, jeu, etc.). Une application mobile est utilisée au moyen d'un smartphone ou d'une tablette. La mobilité et la portabilité du smartphone comme de la tablette composent les caractéristiques essentielles des applications mobiles [4].

1.1.2 Les types d'application mobile

Nous avons différents types d'applications mobiles qui se distinguent soit par le mode de développement soit par le type de plateformes sur lesquelles elles peuvent être déployées. Nous en recenssons 04 types.

1.1.2.1 Application mobile native

Une application mobile est dite native si le logiciel de l'application est développé en fonction du système d'exploitation mobile sur lequel l'application devra être exécutée. Principalement nous avons trois systèmes d'exploitation sur lesquels on peut faire du développement natif : Android, iOS et Windows Phone. Ces applications sont développées à l'aide de langage de programmations précis : Java pour Android, Objective C ou Swift pour iOS et XAML pour Windows Phone. En raison de ce développement spécialisé selon le système d'exploitation, le logiciel peut interagir rapidement avec les outils spécifiques du smartphone : contacts, appareil photo, localisation. Elles ont aussi l'avantage d'être rapide et fiable. Cependant, mettre sur pied une application mobile native demande un budget conséquent pour rendre l'application disponible sur tous les systèmes d'exploitation mobile, étant donné qu'on a une phase de développement pour chaque système d'exploitation. Pour être utilisée, l'application mobile native doit être téléchargée par l'utilisateur à partir des plateformes d'applications dédiées au système d'exploitation : Google Play pour Android, App Store pour iOS et Microsoft Store pour Windows Phone [5].

1.1.2.2 Application mobile hybride

Il s'agit d'applications qui se présentent comme des applications natives à la différence qu'elles sont approvisionnées par un site web. Elles sont encapsulées dans un conteneur natif appelé WebView exploitant les langages de développement web comme HTML, CSS et Javascript. Les technologies permettant de développer ce genre d'application sont : Ionic Framework, Telerik Plateform, Trigger.IO, Cordova. Les applications hybrides ont l'avantage d'être rapide et facile à développer, elles nécessitent peu d'entretien et la mise à jour de l'application est fait automatiquement sans intervention de l'utilisateur. L'inconvénient pour ce genre d'applications mobiles est que la vitesse dépend entièrement du navigateur sur lequel est lancé l'application et aussi elles sont forcément moins rapides que les applications natives. L'accès aux fonctionnalités matérielles du téléphone se fait à travers des plugins [5].

1.1.2.3 Application mobile multiplateforme

Les applications mobiles multiplateformes fonctionnent sur plusieurs systèmes d'exploitation mobile. Elles sont développé grâce à un langage de programmation intermédiaire et par la suite, les applications natives à chaque système d'exploitation sont générées automatiquement à partir de ce code intermédiaire. Le développement de l'application repose donc sur un seul code pour toutes les plateformes. Cependant il peut arriver que l'on spécifie une partie du code pour avoir des résultats spécifiques pour chaque plateformes. Les applications multiplateformes peuvent être développées avec les technologies telles que Flutter, React Native ou Xamarin. Ce type d'application mobile a l'avantage d'être moins coûteux en budget et en temps de développement, étant donné qu'à partir du même code, les applications mobiles sont générées pour chaque

plateformes. Le rendu et la performance de telles applications peut être très proche de celui des applications mobiles natives. L'inconvénient est la difficulté pour assurer la maintenance qui est spécifique à chaque plateforme [5].

1.1.2.4 Progressive Web app

Une progressive web app (PWA, application web progressive en français) est une application web qui consiste en des pages ou des sites web, et qui peuvent apparaître à l'utilisateur de la même manière que les applications natives ou les applications mobiles. Il s'agit de faire évoluer une application web pour la faire fonctionner en tant que application de bureau ou mobile. Elles sont plus rapide que les sites web qu'on ouvre sur mobile, sont disponibles hors ligne et fonctionnent comme des applications natives mais ne sont pas installées à partir des plateformes de téléchargement telles que Google Play Store et AppStore, ce qui diminue considérablement le nombre d'utilisateurs disposés à utiliser ce genre d'applications mobile. Deplus elles consomment beaucoup de batterie, n'ont pas accès à toutes les fonctionnalités du téléphone et offrent des fonctionnalités non compatibles avec le système d'exploitation iOS [5].

1.1.3 Etude comparative des types d'application mobile

Nous mettons en évidence une étude comparative entre les types d'applications mobiles à travers le tableau 1.1 :

TABLE 1.1: Comparaison des différents types d'application mobile

	Application native	Application hybride	Application multiplateforme	Application mobile PWA
Développement et code source	Développement d'un code source pour chaque système d'exploitation	Developpement d'un unique code source	Developpement d'un unique code source et génération de code pour chaque système d'exploitation mobile	Developpement d'un unique code source et génération du code pour la version mobile
Langage/ technologies	Objective-C, Swift, iOS SDK, Java, ADT, .NET	HTML, CSS, Javascript	React native, Flutter, Xamarin	HTML, CSS, Javascript
Distribution	Google Play Store, AppStore, Windows App Store	Google Play Store, AppStore, Windows App Store	Google Play Store, AppStore, Windows App Store	Web
temps de developpement	Elevé	Moyen	Moyen	Faible
Coût de développement	Elevé	Moyen	Moyen	Faible
performance	très bonne	Moyenne	Bonne	très bonne
Mise à jour	Manuelle	Automatique	Manuelle	Automatique
Utilisation hors ligne	Oui	Non	Oui	Oui
Fonctionnalités du smartphone	Integrables rapidement	Integrables rapidement	Integrables rapidement	Plus difficilement integrables

1.2 Notions liées à la messagerie instantanée

1.2.1 Définition de la messagerie instantanée

La messagerie instantanée est un mode de communication dans lequel il y a échange instantané de messages textuels et de fichiers entre deux personnes ou plus grâce à des outils électroniques tels que les ordinateurs ou les téléphones portables. Elle est très souvent confondue avec la notion de courrier électronique. La différence fondamentale entre les deux notions est que le courrier électronique fonctionne comme un service postal qui est indépendant de l'application utilisé pour accéder aux messages (autrement dit on peut accéder à son courrier via plusieurs applications) alors que la messagerie instantanée permet une communication en temp réel entre deux utilisateurs connectés simultanément et est fortement liée au logiciel utilisé (il n'est pas possible de recevoir des messages d'une application de messagerie instantanée via une

autre). Chaque fois qu'un utilisateur envoit un message si le destinataire est connecté, celui ci reçoit immédiatement le message.

1.2.2 historique de la messagerie instantanée

L'histoire de la messagerie instantanée remonte à plusieurs années. La messagerie instantanée **un à un** existe sous UNIX depuis bien longtemps, sous forme de texte, grâce à la commande "talk", puis sous MS Windows, il y a eu son équivalent avec WinPopUp. Les deux systèmes implémentaient une messagerie instantanée utilisateur/machine.

En 1982, les DNA(Dernieres Nouvelles d'Alsace) lancent dans le cadre du service minitel expérimental Gretel le premier service de messagerie instantanée grand public.

En 1988 le protocole standard ouvert Internet Relay Chat (IRC) fournit des fonctionnalités simples de **discussion à plusieurs**. Le protocole Zephyr, créé au MIT la même année, est un ensemble très simple de services de base pour la messagerie instantanée. Cependant ces deux manières de converser sur le réseau ne sont toutefois pas encore ce qu'on appelle aujourd'hui la messagerie instantanée, du fait qu'il n'y a pas ou peu d'authentification ni de gestion de présence [6]. De plus la messagerie à cette époque adopte de nombreux traits propres à l'oralité. Par exemple avant la messagerie moderne chaque caractère écrit était instantanément envoyé au destinataire et sa suppression aussi, cela ressemblait beaucoup à la téléphonie.

La messagerie instantanée moderne grand public voit le jour en 1996 avec l'application de messagerie instantanée ICQ avec pour innovation principale la gestion d'une liste de contacts personnels. Depuis ICQ de nombreux protocoles de communication ont vu le jour dont la plupart sont propriétaires et fermés développés par des entreprises telle que Yahoo ! Messenger ou Facebook qui offrent des applications gratuitement mais à condition d'accepter des annonces et des publicités [6].

En 2002, le nombre de messages instantanés explose et la messagerie instantanée concurrence désormais le courrier électronique avec un nombre d'utilisateurs estimé à 360 millions.

En 2004, le premier protocole de communication non propriétaire voit le jour : Jabber/XMPP et sera par la suite utilisé par deux anciens ingénieurs de Yahoo pour créer WhatsApp.

En 2013, Telegram est crée et explose en 2018 avec plus de 200 millions d'utilisateurs. Depuis la messagerie instantanée a connu une très large expansion.

1.2.3 Fonctionnement de la messagerie instantanée

La messagerie instantanée se constitue d'un logiciel client installé sur un terminal (ordinateur, téléphone, tablette etc) qui représente le point de terminaison dans la circulation des messages à travers le réseau, et un serveur de messagerie dont le rôle est de relayer les messages vers les destinataires finaux connectés. Les échanges de messages entre les terminaux et les serveurs sont régit par un ensemble de protocoles de communication qui donnent des spécifications sur les règles de communication à respecter et en l'occurence les spécifications sur l'échange

des messages. Cet ensemble de protocole forme une pile de protocole. A la base de cette pile protocolaire se trouve les protocoles TCP et IP qui sont respectivement un protocole de transport fiable en mode connecté et un protocole de communication conçu pour être utilisé sur internet. Ce sont des protocoles Standard pour tout échange de paquet en mode connecté à travers le réseau internet. Ils régissent les échanges de paquets au niveau des couches réseau et transport du modèle de couche OSI [1]. Mais Au niveau de la couche application, nous devons également avoir des protocoles qui réglementent l'échange de messages. C'est au niveau du choix de cette pile protocolaire de la couche application que les logiciels de messagerie instantannée diffèrent généralement les uns des autres du point de vu de la logique de communication. Pour communiquer en toute sécurité par messages à travers un réseau nous devons assurer 03 fonctions principales à savoir :

1. Le logiciel client doit pouvoir établir une connexion avec le serveur distant pour échanger : Pour celà nous devons utiliser un protocole de communication permettant d'établir la connexion au niveau applicatif entre nos deux entités (le logiciel client et le logiciel serveur). Les deux protocoles les plus utilisés pour cette fonction sont : HTTP et Websocket.
2. Le logiciel client et le logiciel serveur doivent pouvoir s'entendre sur la structure des messages à échanger : pour celà nous devons utiliser un protocole de communication orienté message : Les deux protocoles de communication non propriétaires parmis les plus répandus sont : Jabber/XMPP et STOMP.
3. Les messages échangés par les deux entités doivent être cryptés pour garantir la confidentialité et l'intégrité. Pour cela un protocole de chiffrement de message doit être utilisé. Plusieurs protocole de chiffrement de message sont disponibles : le protocole Signal, le protocole SSL/TLS.

1.2.4 Les protocoles de connexion au serveur

1.2.4.1 Le protocole HTTP

Le protocole HTTP (HyperText Transfert Protocol) est utilisé depuis 1990 sur internet. Il reste le protocole le plus utilisé à présent. La version 0.9 était uniquement destinée à transférer des données sur internet(en particulier des pages Web écrites en HTML). Désormais la version 1.0 permet de transférer des messages avec des en-têtes décrivant le contenu du message en utilisant un codage de type MIME. [2] Il est défini au sein de la RFC 6838. Il est donc possible d'envoyer des fichiers via la version 1.0 du protocole HTTP.

1. Le modèle OSI est une norme de communication en réseau proposé par L'ISO (organisation internationale de normalisation) qui définit 07 couches nécessaires à la communication entre ordinateurs dont les couches transport et réseau

2. MIME (Multipurpose Internet Mail Extensions) est standard de codage qui permet d'insérer des documents (images, sons, texte etc) dans un courrier.

1.2.4.2 Le protocole WebSocket

Le protocole WebSocket est un standard du Web désignant un protocole réseau de la couche application et une interface de programmation du World Wide Web visant à créer des canaux de communication full-duplex par dessus une connexion TCP pour les navigateurs web[7]. Il permet donc un échange bidirectionnel avec possibilité pour le client web et le serveur web de s'échanger des informations simultanément. Plus précisement le protocole Websocket fonctionne comme suit :

1. Un client web souhaite établir un canal de communication full-duplex entre lui et le serveur web, il envoit alors une requête HTTP au serveur Web pour demander une connexion full-duplex sur l'un de ses ports. Le serveur envoi une réponse HTTP dans laquelle il spécifie les paramètres de connexion et la connexion est établie.
2. Lorsque la connexion est établie, le client peut envoyer des informations au serveur et le serveur peut également notifier/envoyer des informations au client si un évènement se produit.
3. la connexion est maintenue active jusqu'à ce qu'elle soit interrompue par le client ou le serveur. La figure 1.1 présente les étapes de connexion entre le client et le serveur avec le protocole Websocket.

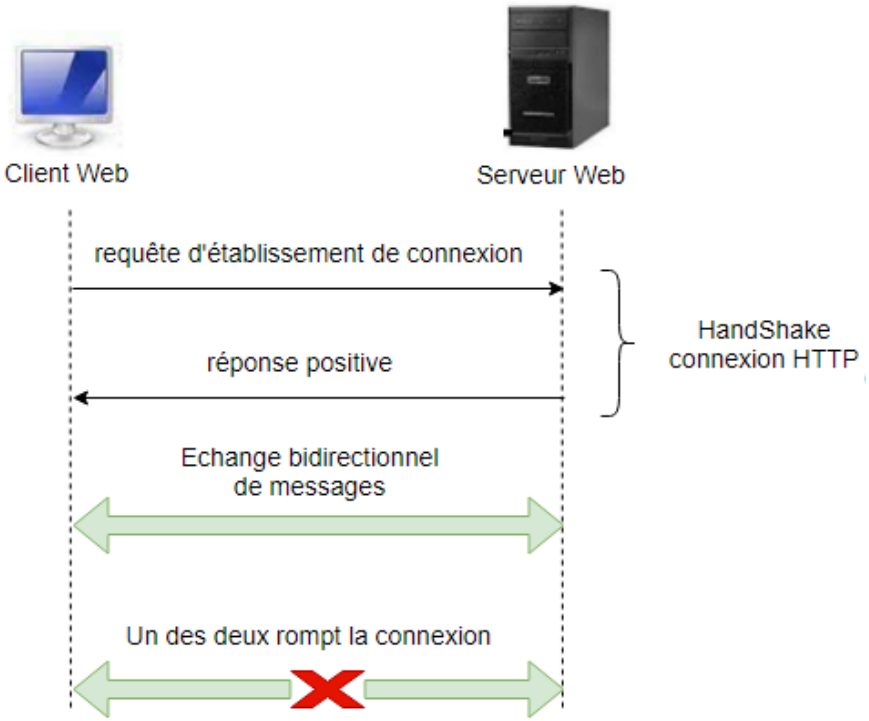

FIGURE 1.1: Etapes de connexion Websocket

1.2.4.3 Etude comparative des protocoles de connexion au serveur : HTTP VS Websocket

Tous deux sont des protocoles de communication utilisés pour établir une connexion client-serveur. Cependant ils diffèrent sur les deux points suivants :

— **Le sens de la communication** : HTTP est unidirectionnel à chaque instant. C'est le client qui envoit une requête au serveur et le serveur répond à la requête du client. Tant dis que le protocole Websocket est bidirectionnel et la communication se fait en full-duplex, le serveur tout comme le client peut envoyer des messages(qui ne sont pas forcément des messages de réponses aux requête du client).

— **La persistance de la connexion** : La connexion HTTP est initiée par le client Web à la reception de la requête HTTP par le serveur et se termine à la réception de la réponse HTTP par le client. Pour envoyer d'autres requêtes au serveur le client doit de nouveau initier une autre connexion. Il s'agit d'un protocole sans état. Par contre, la connexion Websocket est initiée juste après le Handshake HTTP entre le client et le serveur et reste active jusqu'à ce que l'une des parties rompt la connexion. Pendant toute la durée d'activité de la connexion le client et le serveur peuvent s'échanger des informations. Le protocole Websocket est dit avec état.

Différents scénarios d'utilisation pour l'un ou l'autre des deux protocoles à savoir :

— **Des applications temp réel** : il est déconseillé d'utiliser le protocole HTTP pour des applications temp réel. En effet il est possible de simuler un effet temps réel avec le protocole HTTP en implémentant un module qui lance des requête au serveur périodiquement pour vérifier s'il y a modification des données ou si de nouvelles données ont été créées. L'inconvénient avec cette pratique est l'envoi d'un grand nombre de requêtes inutiles lorsque l'état des données n'a pas changé et donc une réduction des performances au niveau de l'application cliente. Il est préférable pour un tel cas d'utiliser le protocole Websocket qui établit une connexion une fois pour toute entre le client et le serveur et à la moindre modification de données le serveur peut notifier le client du changement.

— **Applications nécessitant un import de données en une fois pour traitement** : Ici le protocole HTTP est le plus approprié il est inutile d'utiliser le protocole Websocket et de maintenir une connexion avec le serveur pour rien.

Le tableau 1.2 présente un récapitulatif de la comparaison entre les protocoles HTTP et websocket.

TABLE 1.2: Comparaison des protocoles de connexion au serveur : HTTP vs Websocket

	HTTP	Websocket
Sens de communication	Simplex	Full-Duplex
Durée de vie de la connexion	Non persistante (intervalle entre la requête et la réponse)	Persistante
Avec ou sans état	Sans état	Avec état
Domaine d'application	Application de type requête-réponse	Application temps réel

1.2.5 Les protocoles de communication orienté message

Les deux protocoles de communication orienté message et non propriétaires parmi les plus répandus sont Jabber/XMPP et STOMP. Ils permettent tous deux un échange de messages entre le serveur et le client suivant une structure bien définie. Ils fonctionnent suivant le patron publication-abonnement. C'est un patron largement utilisé dans les protocoles de messagerie, les middlewares orientés-messages (MOM) et les architectures orientées évènement. ce patron permet deux actions principales : **s'abonner à un sujet** et **publier un message**. Ces deux protocoles utilisent des variantes plus ou moins strictes de ce patron qui de manière générale contient les éléments suivants :

- **Le Courtier de message** : il contient un ensemble de sujets au niveau desquels les clients peuvent publier des messages. Chaque sujet a une liste de clients qui y sont abonnés.
- **Les clients** : ils ont la capacité de publier des messages au niveau des sujets et de s'abonner à des sujets au niveau du courtier de message.

La figure 1.2 présente le patron publication-abonnement sur lequel se base les protocoles STOMP et XMPP.

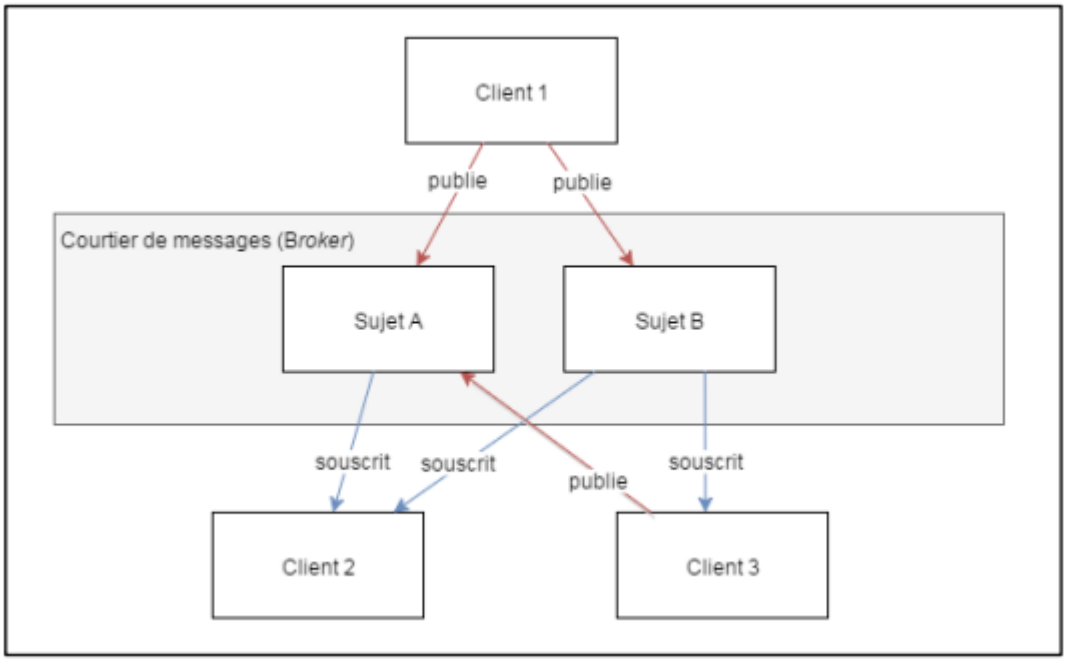

FIGURE 1.2: patron publication-abonnement [1]

1.2.5.1 Le protocole de communication Jabber/XMPP

Communément appelé XMPP pour eXtensible Messaging and Presence Protocol, c'est un protocole standard de l'IETF pour la messagerie instantanée basée sur XML. XMPP désigne le protocole de messagerie tandis que Jabber est l'application de messagerie qui utilise le protocole et qui l'a rendu open source. Ce protocole a trois caratéristiques principales : [8]

— Il est basé sur le format de données XML(eXtensible Markup Language).

— C'est un standard ouvert (libre d'utilisation)

— Il permet d'avoir une adresse de même type que l'email.

Le protocole XMPP repose sur les protocoles TCP/IP basés sur une architecture client-serveur permettant les échanges décentralisés de messages instantanés. L'ensemble des serveurs publics utilisant le protocole XMPP forme le réseau XMPP ou réseau Jabber.

L'ensemble des utilisateurs du protocole XMPP possèdent une adresse similaire à une adresse mail qui est constituée deux parties permanentes :

— Un nom d'utilisateur qui est unique sur un serveur

— un nom de serveur

1.2.5.2 Le protocole de communication STOMP

Le protocole STOMP(Streaming Text Oriented Message Protocol) est un protocole de messagerie basé sur TCP conçu pour permettre l'interaction avec un middleware orienté messages.

C'est un protocole qui permet de gérer l'abonnement et le mapping de messages entre différents utilisateurs. La communication entre le client et le serveur se fait au travers d'une trame spéciale constituée d'une série de lignes[9]. En effet une trame se compose d'une commande, d'un ensemble d'en-têtes facultatives et d'un corps facultatif. Le protocole est basé sur du texte mais permet également la transmission de messages binaires [10]. Un serveur STOMP se modélise par un ensemble de destinations vers lesquelles des messages peuvent être envoyés.

Le protocole STOMP propose plusieurs commandes qui renseignent sur la nature du message ou de l'interaction qui veut être mener par le client auprès du serveur :

— CONNECT : permet à un client de se connecter au serveur STOMP

— SEND : permet à un client d'envoyer un message au serveur STOMP

— SUBSCRIBE : permet de s'abonner à un sujet

— UNSUBSCRIBE : permet de se désabonner d'un sujet

— BEGIN : permet de débuter une transaction de message.

— COMMIT : permet de valider une transaction en cours.

— ABORT : permet d'annuler une transaction en cours

— ACK : permet d'accuser réception d'un message.

— NACK : permet d'indiquer que le message envoyé n'a pas pu être consommé

— DISCONNECT : permet au client de se déconnecter du serveur à tout moment.

Et les réponses possibles pour un serveur STOMP sont :

- **ERROR** : c'est le code envoyé par le serveur juste avant de fermer la connexion pour signifier que quelque chose ne va pas.
- **MESSAGE** : c'est une trame utilisée par le serveur pour envoyer les messages au client.
- **RECEIPT** : c'est une trame envoyée au client par le serveur pour signifier qu'il a bien traité la trame précédemment envoyé par le client.

Plus précisement, un scénario d'utilisation du protocole STOMP est le suivant : Un client A souhaite envoyer un message à un client B. Grâce à la commande SEND, le client A envoit le message à un des sujets auquel est abonné le client B. Lorsque le message est envoyé au niveau du sujet, le courtier de message envoit alors le message à B avec la commande MESSAGE.

1.2.5.3 Etude comparative des protocoles d'échange de message : XMPP VS STOMP

XMPP et STOMP sont tous deux des protocoles de communication orientés message avec chacun ses particularités. XMPP utilise le format de donnée XML pour structurer ses messages tandis que le protocole STOMP utilise un format texte en trame qui est plus léger et donc plus économique en consommation des données utilisateurs. Le protocole XMPP implémente déjà un certains nombre de fonctionnalités liés à la messagerie instantannée comme l'adressage des

utilisateurs qui est prédéfini dans un format fixé. Le protocole STOMP quant à lui fournit les fonctions de base de redirection de message à travers une convention de mappage et d'abonnement aux messages. L'adressage des clients dans le protocole STOMP n'est pas prédéfini et donc laisse un dégré de liberté quant au format de l'adresse. De plus Le protocole STOMP fonctionne très bien au dessus du protocole Websocket qui est un protocole de connexion au serveur adapté aux applications temps réel comme la messagerie instantanée.

Le tableau 1.3 présente une comparaison entre les protocoles STOMP et XMPP.

TABLE 1.3: Comparaison des protocoles de communication orienté message : STOMP et XMPP

	STOMP	XMPP
Format de message	texte en trame	XML
patron d'échange de message	patron publication-abonnement	patron publication-abonnement
protocole de communication pris en charge	TCP, Websocket	TCP, HTTP
Implementation Java/Spring	directe	indirecte (par ajout d'API comme Smark)

Si la connexion entre le client et le serveur est établie et qu'on définit un protocole pour l'échange de message, il est aussi important de garantir la sécurité des messages échangés.

1.2.6 Les protocoles de sécurité de messages

Dans cette section nous allons présenter 02 protocoles de sécurité des messages transmis à travers un réseau. Un protocole de sécurité de message est un formalisme qui définit comment les messages doivent être échangés pour garantir 03 principes de sécurité que sont [11] :

— **L'intégrité :** c'est garantir que les messages sont bien ceux qui ont été envoyé ;

— **La confidentialité :** c'est rendre l'information inintelligible à d'autres personnes que les seuls acteurs d'une transaction ;

— **L'authentification :** c'est assurer que seules les personnes autorisées aient accès aux messages.

1.2.6.1 Le protocole SSL/TLS

SSL(Secure Sockets Layer) et TLS(Transport Layer Security) sont des protocoles de sécurité de messagerie les plus courants. Ce sont tous les deux des protocoles de la couche application. Le protocole TLS vient remplacer le protocole SSL qui est devenu obsolète depuis 2015. Le protocole SSL a été développé à l'origine par NetsCape Communications Corporation pour son navigateur puis l'organisme de normalisation IETF en a poursuivi le développement en le rebaptisant TLS[12]. Le protocole est largement utilisé et ceci est dû au fait qu'il s'intègre facilement aux protocoles de la couche application comme HTTP pour donner lieu au protocole HTTPS (Hyper-Text Transfert Protocol Secure) et SMTP pour donner lieu au protocole SMTPS (Simple Mail Transfert Protocol Secure).

SSL/TLS fonctionne comme suit :

1. TLS crypte les données avant leur transmission et lance un processus d'authentification commune entre le client et le serveur appelé Handshake . Il signe également numériquement les données pour assurer l'intégrité des données, en vérifiant que les données ne sont pas falsifiées avant d'atteindre leur destinataire.
2. Le protocole TLS est implémenté uniquement par des sites web qui possèdent un certificat TLS. Le certificat TLS est un moyen d'identification unique pour un site web ou pour un groupe de sites web. Il met à la disposition du site une clé publique qui permet le chiffrement de messages envoyés au client et une clé privée pour le déchiffrement des messages reçus.

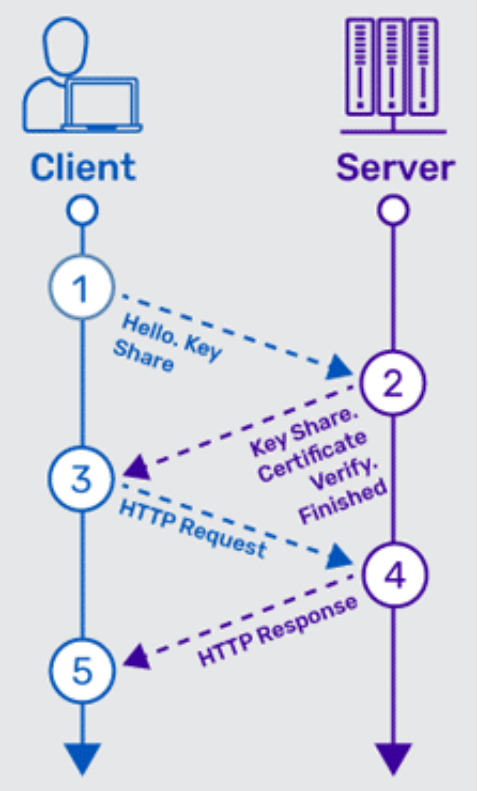

FIGURE 1.3: processus d'échange de clés entre le serveur et le client (Handshake)[2]

Les Certificats TLS sont délivrés par les autorités de certification. Il existe plusieurs type de certificats TLS :

— **Les certificats à domaine unique :** c'est un certificat qui ne s'applique qu'à un seul nom de domaine.

— **Les certificats Générique :** ils s'appliquent à un domaine unique mais contrairement aux certificats à domaine unique ils s'appliquent aussi à tous les sous domaines du domaine.

— **Les certificats Multi-domaine :** ils s'appliquent à plusieurs domaines non liés comme son nom l'indique.

Les certificats TLS sont également assortis d'un niveau de validation qui varie selon le niveau de profondeur de la vérification effectuée par les autorités de certification sur les antécédents du domaine [13] :

— **Validation de domaine :** c'est le niveau de validation le moins stricte. Comme prérequis pour l'obtention de ce type de validation, l'entreprise doit prouver qu'elle contrôle le domaine.

— **Validation de l'organisation :** l'autorité de certification rentre en contact directe avec la personne ou l'entreprise qui demande le certificat. Ces types de certificats sont les plus

fiables pour les utilisateurs

— **Validation étendue :** elle nécessite une vérification complète des antécédents de l'organisation par l'autorité de certification.

1.2.6.2 Le protocole cryptographique Signal

Le protocole signal est un protocole de chiffrement de bout en bout, c'est-à-dire que le contenu en clair d'un message donné n'est visible uniquement pour l'expéditeur et le destinataire du message. Il est développé par Open Whisper Systems en 2013 et utilisé par les applications de messagerie instantanée comme Signal, WhatsApp, Facebook Messenger (en mode conversation secrète), Google Allo (en mode incognito). Les étapes qui entre dans le processus de chiffrement et de déchiffrement des messages sont :

1. **Générer une clé d'identification et toutes les clés nécessaires au chiffrement/déchiffrement :** La première étape consiste à générer un ensemble de paires de clés du côté client et de les stocker localement. Il faut générer une paire de clés d'identité à long terme, une paire de clés signée à moyen terme et plusieurs paires de clés éphémères. Ensuite il faut regrouper toutes les clés publiques (partie publique des paires de clés) et l'identifiant d'enregistrement pour former un trousseau de clés et enfin stocker ce trousseau de clés auprès d'un centre de distribution de clés qui peut être un serveur distant[3]. Plus concrètement pour qu'un utilisateur A puisse envoyer un message à un utilisateur B, il doit récupérer au niveau du centre d'enregistrement l'identifiant d'enregistrement et le trousseau de clés publiques de l'utilisateur B.
2. **Démarrer une session de communication :** Une fois que l'utilisateur A récupère le trousseau de clés publiques de l'utilisateur B au niveau du centre de distribution, il utilise sa propre identité et ses clés privées à moyen terme pour générer un secret principal qui sera partagé par lui et B. Ce secret est ensuite utilisé pour démarrer une session de communication avec l'utilisateur B. Après la validation du secret partagé par l'utilisateur B, les deux utilisateurs peuvent alors commencer à s'envoyer des messages.
3. **Chiffrer et déchiffrer les messages :** Le processus de chiffrement des messages repose sur l'accord de clé X3DH (Extended Triple Diffie-Hellman). L'utilisateur A crypte et envoit des messages à l'utilisateur B en utilisant le secret partagé principal et les clés éphémères de B. Pour chaque message envoyé, un nouvel ensemble de clés de session(éphémère) à usage unique est généré de sorte qu'aucun des messages précédents ou futurs ne puisse être déchiffré par des tiers.

Le protocole Signal est une combinaison astucieuse de 05 éléments appelés "primitives du protocole Signal". Ces 05 protocoles sont [14] :

3. Le centre de distribution de clés dans le protocole Signal est chargé de distribuer les trousseaux de clés d'un utilisateur à d'autres utilisateurs qui souhaitent rentrer en contact par message avec ce dernier.

— **X3DH** est une méthode par laquelle deux agents peuvent se mettre d'accord sur une clé sans qu'un troisième agent puisse découvrir la clé même ayant écouté tous les échanges. Il permet d'établir la clé secrète partagée entre les deux parties qui s'authentifient mutuellement en fonction de leurs paires de clés publiques. X3DH permet également l'échange de clés lorsqu'une partie est « hors ligne ».

— **Algorithme du Double Ratchet** : Après le premier échange de clé assuré par le protocole X3DH, l'algorithme Double Ratchet gère le renouvellement continue et la maintenance des clés de session à courte durée de vie. Il combine un premier élement de renouvellement basé sur l'échange de clés Diffie-Helman (DH) et un second élement de renouvellement basé sur une fonction de dérivation de clés. L'objectif du renouvellement permanent de clés pour le chiffrement/déchiffrement des message est d'empêcher un attaquant ayant réussit à retrouver la clé de chiffrement d'un message d'utiliser cette même clé pour déchiffrer les anciens messages échangés entre deux utilisateurs et les messages futurs qu'ils échangeront.

— **Curve25519 :** c'est une courbe elliptique qui offre 128 bits de sécurité et qui intervient particulièrement dans l'algorithme de signature numérique.

— **AES (Advanced Encryption Standard) :** C'est l'algorithme de chiffrement symetrique le plus sécurisé actuellement. Il s'agit d'un chiffrement par bloc symétrique pour protéger et chiffrer les données sensibles. Il crypte et décrypte les données par blocs de 256 bits.

— **HMAC-SHA256 :** Il s'agit d'un type spécifique de code d'authentification de message impliquant une fonction de hachage cryptographique et une clé cryptographique secrète. Cet algorithme est conçu à partir d'une clé et d'une fonction de hachage (SHA-256 dans notre cas). En sortie on a une chaîne d'une longueur de 256 bits.

1.2.6.3 Etude comparative des protocoles de chiffrement de message : Signal et SSL/TLS

Le protocole Signal et le protocole SSL/TLS sont tous les deux des protocoles de chiffrement de messages à très grande complexité. Cependant, contrairement au protocole SSL/TLS, le protocole Signal est un protocole de chiffrement de bout en bout. Autrement dit le chiffrement du message demeure jusqu'au destinataire final du message qui ensuite peut le déchiffrer. L'implémentation d'une messagerie instantanée basée sur le chiffrement SSL/TLS est sous réserve d'une confiance absolu aux différents serveurs qui relaient le message car ceux-ci ont la clé de déchiffrement du message et peuvent donc voir en clair le message chiffré. Par contre Le protocole Signal établit un chiffrement qui demeure même pour les serveurs qui relaient le message à travers le réseau.

Le tableau 1.4 présente une comparaison entre les protocoles Signal et SSL/TLS.

TABLE 1.4: Comparaison des protocoles de chiffrement : Signal et SSL/TLS

	Signal	SSL/TLS
Mode de chiffrement	Chiffrement End-To-End	Chiffrement Client-Serveur
Fréquence de regénération de clés de chiffrement	A chaque envoi de message	Rarement
Mode de connexion entre utilisateurs	Asynchrone	Synchrone

1.3 Approches existantes

Il existe deux principales approches pour mettre en place une application mobile de messagerie instantanée : les générateurs d'applications mobiles et le développement. Nous allons présenter les deux approches avec leurs avantages et leurs inconvénients.

1.3.1 Les générateurs d'applications mobiles

Un générateur d'application mobile est un Content Management System (CMS) (Système de Gestion de Contenu en français) permettant de générer des applications mobiles pour smartphones et tablettes sans avoir à développer l'application. Il existe plusieurs générateurs d'application mobile qui diffèrent par la qualité du service et les tarifs. Nous pouvons citer par exemple : Appypie, GoodBarber, WordPress et AppsGeyser.

1.3.1.1 Avantages

- Les CMS offrent une panoplie de thème d'application
- Possibilité de générer une application facilement sans écrire une seule ligne de code.
- Si le CMS est OpenSource il sera régulièrement mis à jour et amélioré par une large communauté de développeurs
- La publication de l'application sur les plateformes de téléchargement est plus facile.

1.3.1.2 Inconvénients

- Difficile de trouver des experts pour ces plateformes

- Les applications mobiles déployées gratuitement présentent beaucoup de publicité.
- Il est difficile voir impossible de lier l'application à une base de données externe.
- Les fonctionnalités sont prédéfinies
- La plupart de ces plate-formes ne sont disponibles qu'en ligne et nécessite donc de disposer d'une bonne connexion.

1.3.2 Le développement de l'application

Cette approche consiste à constituer une équipe de concepteurs et développeurs pour mettre sur pied l'application. Il faudra alors choisir les différents protocoles de la couche application à utiliser pour implémenter les échanges de message.

1.3.2.1 Avantages

- On paie une seule fois pour la mise en place de l'application mobile ;
- L'on est propriétaire du code source de l'application ;
- L'on peut intégrer toutes les fonctionnalités que l'on souhaite et faire évoluer l'application ;
- L'on a accès à la base de données.

1.3.2.2 Inconvénients

- Il faut un investissement convenable au début ;
- Nécessite des compétences en conception et en programmation d'application mobile ;
- L'on doit publier l'application mobile soi-même en respectant les règles de publication ;
- Cette approche est très coûteuse et demande beaucoup de temps pour sa mise en oeuvre.

1.4 Solutions de messagerie instantanée existantes

Dans cette section nous allons citer quelques logiciels clients de messagerie existants. Nous les distinguerons en deux familles : les logiciels libres ou OpenSource et les logiciels propriétaires.

1. **Les logiciels libres**
   - **Gajim :** c'est un logiciel client de messagerie instantanée pour le réseau standard ouvert Jabber/XMPP. Gajim est disponible sur système UNIX, GNU/LINUX, GNU/hurd et Microsoft. Gajim est créé en 2004 et propose entre autres les fonctionnalités suivantes :
     - Fenêtres de discussion avec onglets
     - Support des groupes de discussion

- Gestion des émoticônes et des avatars
- transfert de fichiers
- Support des méta-contacts

— **Psi :** c'est un logiciel client de messagerie instantanée pour le réseau standard ouvert Jabber/XMPP. Psi est léger et possède une ergonomie simple et s'intègre à l'interface graphique des différentes plateformes supportées. Il est créé en 2001 et propose entre autres les fonctionnalités suivantes :

- Psi permet de fixer la priorité du client
- La découverte de l'ensemble des services proposés par les serveurs
- Les recherches d'utilisateurs
- Les discussions en groupe
- le chiffrement SSL/TLS
- le transfert de fichiers

— **Adium :** est un logiciel client de messagerie instantanée pour Mac OS. c'est un logiciel client IRC(Internet Relay Chat). Il propose entre autres les fonctionnalités suivantes :

- Adium permet la gestion du statut
- groupement de plusieurs contacts
- enregistrer les conversations
- ajout ou configuration d'un alias pour chaque contact.
- le transfert de fichiers

2. **Les logiciels propriétaires**

— **WhatsApp :** c'est un logiciel client de messagerie instantanée chiffré de bout en bout et multiplateforme. L'application WhatsApp est créée en 2009 et propose entre autres les fonctionnalités suivantes :

- conversation privé
- conversation de groupe
- ajout de statut ou story
- chiffrement signal
- le transfert de fichiers

— **Facebook Messenger :** c'est un logiciel client de messagerie instantanée multiplateforme crée par la société Facebook. L'application est créée en 2004 et propose entre autres les fonctionnalités suivantes :

- Un Mur : espace au sein de votre profil où sont rassemblées toutes vos publications et activités.
- Gestion de la liste des amis
- notifications
- publication de post, likes et commentaires
- Les groupes de discussions

— **Yahoo ! Messenger :** est un logiciel client de messagerie instantanée crée en 1998 par la société Yahoo !. l'application est créée en 1994 et propose entre autres les fonctionnalités suivantes :

- conversation à deux
- conversation de groupe
- partage de photos/vidéos
- synchronisation avec les contacts enregistrés

1.5 Positionnement

Nous avons passé en revue les différentes approches existantes pour créer des applications mobiles de messagerie instantanée et quelques solutions de messagerie instantanée existantes. Cependant, il sera question de développer une solution de messagerie instantanée propriétaire pour avoir le contrôle sur le code source et donc rendre possible l'évolution des fonctionnalités de la solution. Pour cela, nous devons prendre position, premièrement quant au type d'application mobile à developper et ensuite sur la pile de protocole qui sera utilisée au niveau de la couche application de notre solution et qui permettra un échange sécurisé de message. Nous avons présenté les différents types d'application mobiles :

- Les applications mobiles natives
- Les applications mobiles hybrides
- Les applications mobiles multiplateformes
- Les Progressive Web App

Nous avons présenté une comparaison des différents types d'application mobile au niveau du tableau 1.1. Etant donné la taille de l'équipe de développement (01) et le temps imparti, nous avons opté pour le développement d'une application mobile multiplateforme qui permet un développement unique pour plusieurs systèmes d'exploitation mobile.

Nous avons également présenté différentes possibilités de protocoles à utiliser au niveau de la couche application :

- **Les protocoles de connexion au serveur (HTTP ou Websocket)** : Nous avons présenté une comparaison entre les deux protocoles dans le tableau 1.2. Nous optons pour l'utilisation des deux protocoles, Websocket utilisé pour l'échange temps réel de message texte et le protocole HTTP pour l'export et l'import des fichiers.
- **Les protocoles d'échange de messages (STOMP ou XMPP)** : Nous avons présenté une comparaison entre les deux protocoles dans le tableau 1.3. Nous optons pour l'utilisation du protocole STOMP premièrement à cause de son format de message qui est léger et qui réduit ainsi la consommation de données utilisateur, ensuite du fait qu'il prend en charge le protocole Websocket, enfin et principalement parce les technologies imposées pour le développement de l'application (Java et Spring) implémentent directement le protocole STOMP, ce qui n'est pas le cas avec le protocole XMPP.
- **Les protocoles de chiffrement/déchiffrement de message (Signal ou SSL/TLS)** : Nous avons présenté une comparaison entre les deux protocoles dans le tableau 1.4. Nous optons pour l'utilisation du protocole de chiffrement Signal parce qu'il est de type End-to-End et donc plus sécurisé qu'un chiffrement de type client-serveur.

1.6 Bilan du chapitre

Il a été question de faire un état de l'art en ce qui concerne les applications mobiles de messagerie instantanées. Pour cela nous avons présenté dans un premier temps les différents types d'applications mobiles existantes, et par la suite les concepts liés à la messagerie instantanée à savoir son historique, son fonctionnement, les différents types de protocoles de communication utilisés pour sa mise en oeuvre, les solutions et approches existantes existantes. Nous avons pris position quant à la réalisation de ce travail à savoir la mise en oeuvre d'une application mobile multiplateforme de messagerie instantanée reposant sur une pile de protocoles de couche application constituée du protocole HTTP, WebSocket, STOMP et Signal tel que illustrée à la figure 1.4. Le chapitre suivant présente la méthodologie employée pour la réalisation de notre solution.

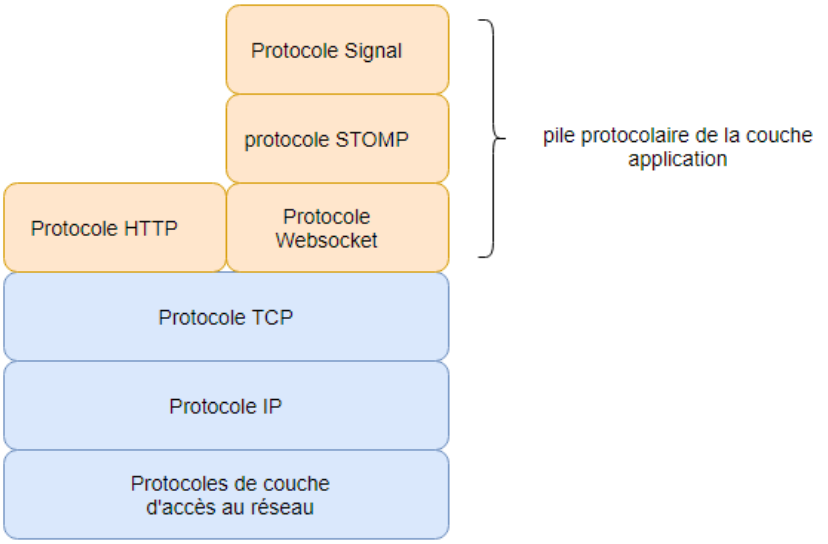

FIGURE 1.4: Pile de protocoles pour l'échange sécurisé de message

CHAPITRE 2

MÉTHODOLOGIE

Dans ce chapitre, il est question de décrire la démarche adoptée pour la résolution de notre problème. Premièrement nous présenterons la modélisation des échanges entre plusieurs entités dans une messagerie instantannée. Ensuite nous ferons l'analyse des besoins de l'application et enfin la conception de notre solution.

2.1 Modélisation des échanges dans une méssagerie instantanée

2.1.1 Les types d'échanges : échange One-To-One et échange One-To-Many

Nous partons de deux types d'échange dans la messagerie instantanée pour faire la modélisation de notre système : les échanges de type One-To-One et les échanges de type one-To-Many.

Un échange de type One-To-One implique une personne à l'émission et une personne à la réception.

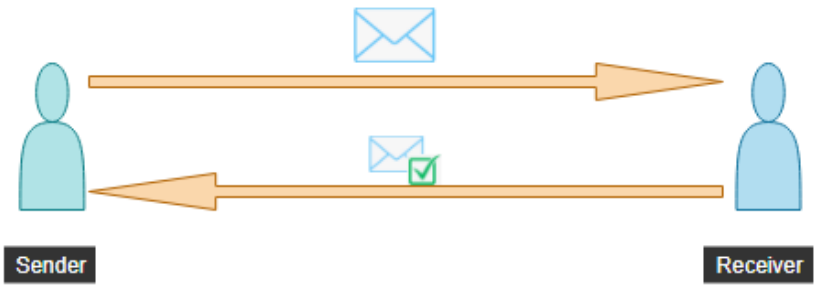

FIGURE 2.1: Echange One-To-One

Un échange de type One-To-Many implique une personne à l'émission et plusieurs personnes à la réception.

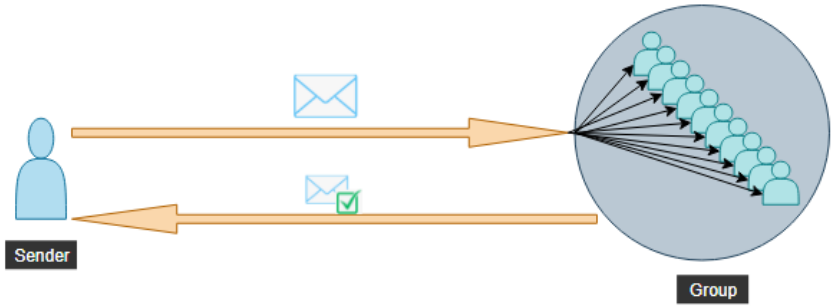

FIGURE 2.2: Echange One-To-Many

2.1.2 Le modèle des utilisateurs et des groupes d'utilisateurs de la messagerie

Dans une messagerie instantanée les utilisateurs peuvent endosser à un instant donné les rôles d'emetteur ou de recepteur de message. Un utilisateur est modélisé par un identifiant unique comme le numéro de téléphone par exemple. Des groupes de discussion peuvent se former dans une application de messagerie instantanée. Un groupe de discussion est modélisé par un identifiant unique de groupe et des caractérisques telles que :

- Un nom de groupe ;
- Une description du groupe ;
- le créateur du groupe ;
- la liste des membres du groupe.

2.1.3 Modèle de l'information échangée

Tout échange d'information entre utilisateurs de la messagerie est considéré comme un message circulant de l'émetteur vers le serveur puis du serveur vers les recepteurs. A cet effet pour une application de messagerie instantanée ayant les fonctionnalités telles que la gestion du profil utilisateur, la gestion du profil de groupe, les conversations individuelles, les converstions de groupe, les statuts et les posts nous distinguons 07 types de messages qui sont :

— Les messages de modification ou de création d'utilisateur ;

— Les messages de modification ou de création de groupe ;

— Les messages de chat individuel ;

— Les messages de chat de groupe ;

— Les statuts ;

— Les posts ;

— Les commentaires de post.

Nous classons ces 07 formes de message selon les deux grands types d'échanges. Le tableau 2.1 présente cette classification.

TABLE 2.1: Classification des formes de messages en type d'échange

Forme de message	**Type d'échange**
Message de modification ou de création d'utilisateur	Echange One-To-Many
Message de modification ou de création de groupe	Echange One-To-Many
Message de chat individuel	Echange One-To-One
Message de chat de groupe	Echange One-To-Many
Le Statut	Echange One-To-Many
Le post	Echange One-To-Many
Le Commentaire	Echange One-To-Many

- L'envoi d'un message de modification ou de création d'utilisateur est un échange de type One-To-Many car la modification du profil d'un utilisateur ou la création d'un nouveau profil utilisateur doit être signifié à tous les utilisateurs de la messagerie. le message sera envoyé de l'utilisateur dont le profil a changé ou a été créé vers tous les autres utilisateurs.
- L'envoi d'un message de modification ou de création d'un groupe est un échange de type One-To-Many car la modification d'un groupe (modification du nom du groupe ou de la description ou de la liste des membres) doit être signifié à tous les membres du groupe. Donc le message sera envoyé de l'utilisateur qui fait la modification vers tous les autres membres du groupe.
- L'envoi d'un message de chat individuel est un échange de type One-To-One car le message doit être envoyé de l'utilisateur emetteur du message à l'utilisateur récepteur qui est unique.
- L'envoi d'un message de groupe est un message de type OneToMany car le message de groupe est signifié à tous les membres du groupe.
- L'envoi d'un Statut est un message de type One-To-Many car le statut est émit par un utilisateur en direction de tous les utilisateurs de la messagerie faisant partie de ses contacts.
- L'envoi d'un post est un Echange One-To-Many car le message doit être envoyé vers tous les utilisateurs de la messagerie faisant partie de ses contacts.
- L'envoi d'un commentaire de post est signifié à tous les utilisateurs qui voient le post.

Tout message échangé respecte la structure **Entête + Corps + fichier joint**, où l'entête est obligatoire, le corps et le fichier joint sont facultatifs. Par exemple un message de modification/création d'utilisateur contiendra en entête l'identifiant de l'utilisateur et le type de message, au niveau du corps il contiendra facultativement le nom de l'utilisateur et sa description et enfin au niveau du fichier joint on aura optionnellement une URL de la photo de profil de l'utilisateur.

2.1.4 Le modèle du moteur de rédirection de messages

Le moteur de redirection de messages est l'élément intermédiaire dans l'échange des messages entre les utilisateurs de la messagerie. Il est chargé de recevoir les messages venant des émetteurs et d'envoyer aux utilisateurs destinataires. En effet selon le type de message reçu et le corps du message, le moteur de redirection de message est capable de savoir quels sont les destinataires et leur envoyer le message. Nous modélisons le moteur de redirection de messages par un ensemble de correspondances entre sources de messages et destinations de message. Généralement il est conçu en suivant le pattern publication-abonnement qui inclut les éléments suivants :

- **Un canal de messagerie en entrée** utilisé par l'expéditeur. L'expéditeur envoit des messages par le biais du canal d'entrée dans un format connu.
- **Un canal de messagerie en sortie** utilisé par le consommmateur. Les consommateurs de messages sont aussi appelés abonnés.
- **Un repartiteur de message** qui utilise un mécanisme de copie de chaque message depuis le canal d'entrée jusqu'aux canaux de sortie pour tous les abonnés ayant souscrit au message. Cette opération est généralement gérée par un intermédiaire comme un courtier de messages ou un bus d'évènements.

La figure 2.3 montre les composants logiques du modèle de distribution de message basé sur le patron publication-abonnement.

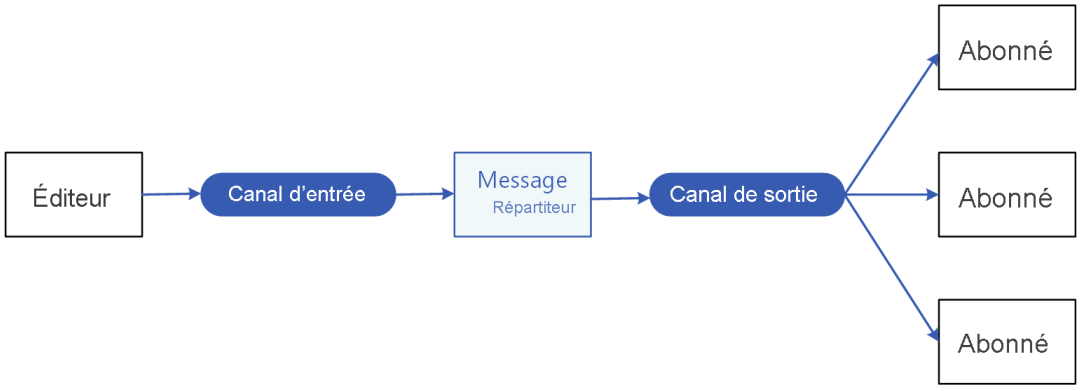

FIGURE 2.3: modèle de distribution de message "publication-abonnement"

2.2 Méthodologie de gestion du projet

La gestion de projet est l'ensemble d'activités concourant à la bonne réalisation d'un projet. Elle consiste à appliquer un ensemble de techniques et méthodes à différentes étapes du projet. Nous avons suivi le modèle de cycle en V, qui est un modèle de gestion de projet qui implique toutes les étapes du cycle de vie d'un projet : conception, réalisation, validation [15]. Cette méthodologie a pour principals avantages le fait qu'elle évite de revenir en arrière incéssamment pour redéfinir les spécifications initiales et le fait que le processus soit facile à mettre en oeuvre. L'inconvénient principal de cette méthodologie est **l'effet tunnel** : après la définition des spécifications initiales du produit à réaliser, le projet est lancé dans un « tunnel » ou les changements de besoins ne peuvent pas ou peuvent difficilement être pris en compte. La figure 2.4 représente les différentes phases du cycle en V de gestion de projet.

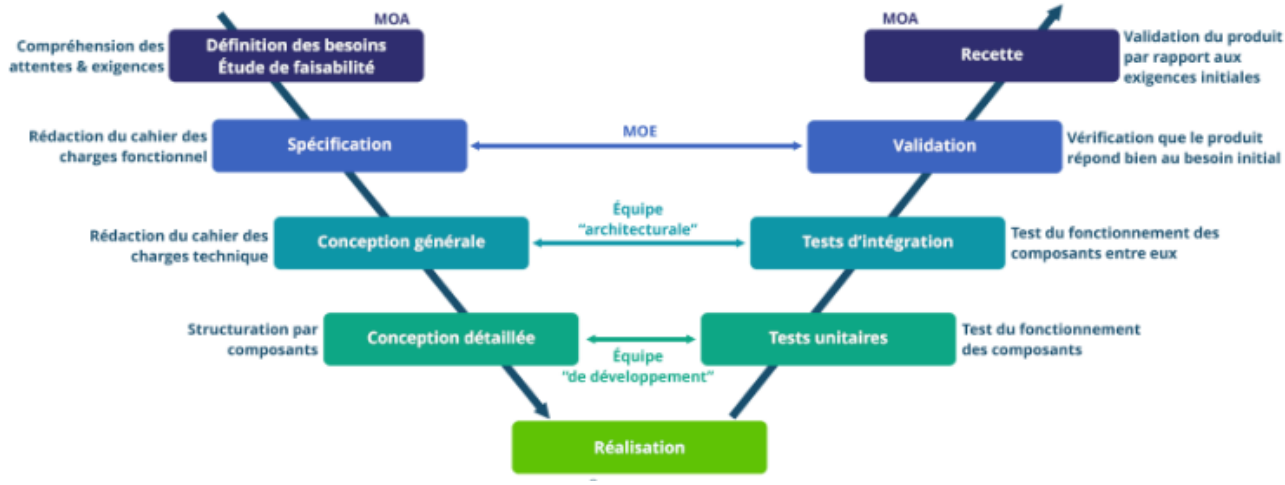

FIGURE 2.4: Cycle en V de gestion de projet

2.3 Analyse des Besoins

2.3.1 Exigences fonctionnelles

Les exigences fonctionnelles sont l'ensemble des fonctionnalités qu'un système met à la disposition des utilisateurs.

La plateforme devra proposer les fonctionnalités suivantes :

- Créer ou modifier un profil utilisateur ;
- Authentifier un utilisateur ;
- Envoyer un message à un utilisateur ;
- Recevoir un message d'un utilisateur ;
- Notifier la reception d'un message d'un utilisateur ;

- Créer ou modifier un groupe ;
- Envoyer un message dans un groupe ;
- Recevoir un message de groupe ;
- Notifier la reception d'un message de groupe ;
- Créer un statut ;
- Créer un post ;
- Ajouter un Commentaire à un post ;
- Ajouter un like à un post ;
- Consulter le profil d'un utilisateur ;
- Enregistrer un nouveau contact.

2.3.2 Exigences techniques

Les exigences techniques ou exigences non fonctionnelles sont l'ensemble des contraintes dans la réalisation des fonctionnalités d'un système. La réalisation du système sera soumise aux contraintes technique suivantes :

— La solution devra être une application mobile ;
— Le framework utilisé en Frontend devra être Flutter ;
— Le framework utilisé en Backend devra être Spring Boot ;
— Le numéro de téléphone sera utilisé comme identifiant d'un utilisateur de la messagerie ;
— Le SGBD utilisé côté serveur devra être Apache Cassandra ;
— L'ensemble des messages et échanges devront être conserver pour une période légale de 6 mois.

2.3.3 Acteurs du système

Un acteur est un utilisateur type qui a toujours le même comportement vis-à-vis d'un cas d'utilisation. Un acteur peut aussi être un système externe avec lequel le cas d'utilisation va interagir [16]. Dans le cadre de notre projet nous avons les acteurs suivant :

2.3.3.1 Acteurs primaires

Un acteur primaire est un acteur pour lequel est conçu le système. Nous avons comme acteurs primaires tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.

2.3.3.2 Acteurs externes

Dans ce projet, nous avons comme acteurs externes :

— l'application de gestion de contacts : qui fournit au système les noms des contacts attribués au numéro de téléphone des utilisateurs.

— l'application de SMS du téléphone : qui fournit le code d'authentification de numéro envoyé à l'application de messagerie.

2.3.4 Les cas d'utilisation

Un cas d'utilisation correspond à un certain nombre d'actions que le système devra exécuter en réponse à un besoin d'un acteur [16]. Suivant les exigences fonctionnelles du système nous regroupons les cas d'utilisation en modules à savoir :

— Le module de gestion des profils utilisateurs et contacts ;

— Le module de gestion des groupes ;

— Le module de gestion des messages individuels et de groupe ;

— Le module de gestion des statuts, des posts et des commentaires ;

Ces différents modules sont présentés dans la suite par des diagrammes et formulaires de cas d'utilisation.

2.3.4.1 Description Graphique : les diagrammes de cas d'utilisation

Un diagramme de cas d'utilisation est un schéma qui représente les besoins des utilisateurs d'un système par les acteurs du système et leurs différentes interactions avec les cas d'utilisation du système.

Cas d'utilisation du module de gestion des profils utilisateurs et contacts

La figure 2.5 présente les actions qu'un utilisateur peut effectuer dans le cadre de la gestion de son profil et de ses contacts.

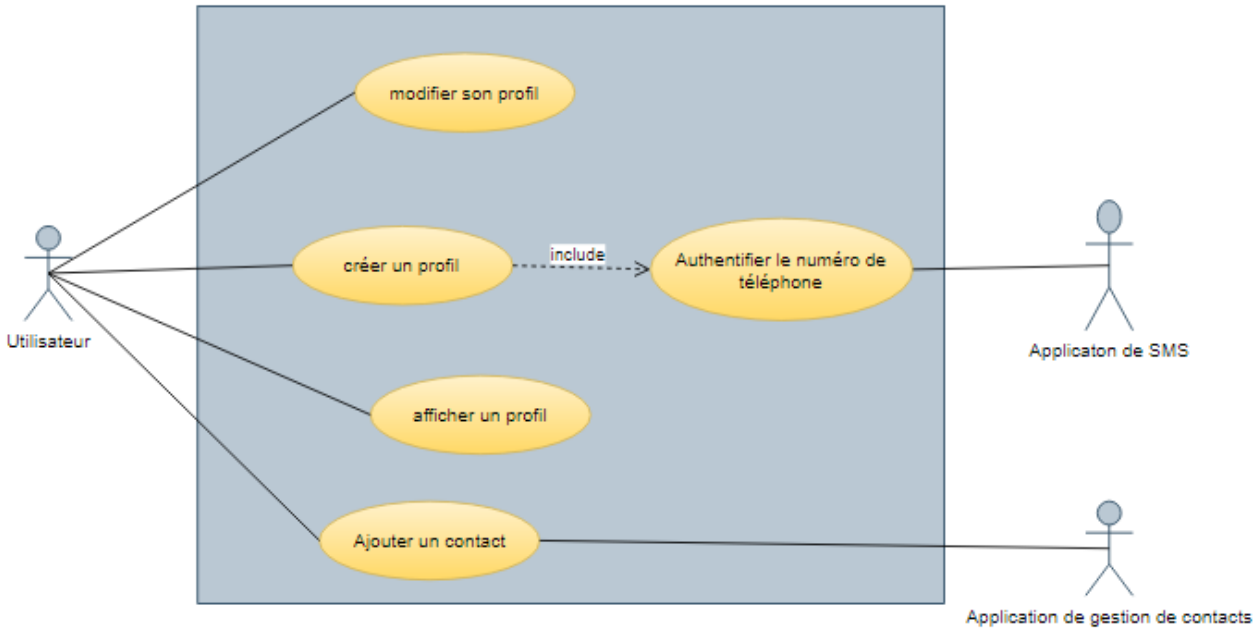

FIGURE 2.5: DCU du module de gestion des profils utilisateurs et contacts

Cas d'utilisation du module de gestion des groupes

La figure 2.6 présente les actions qu'un utilisateur peut effectuer en ce qui concerne un groupe de discussion auquel il appartient :

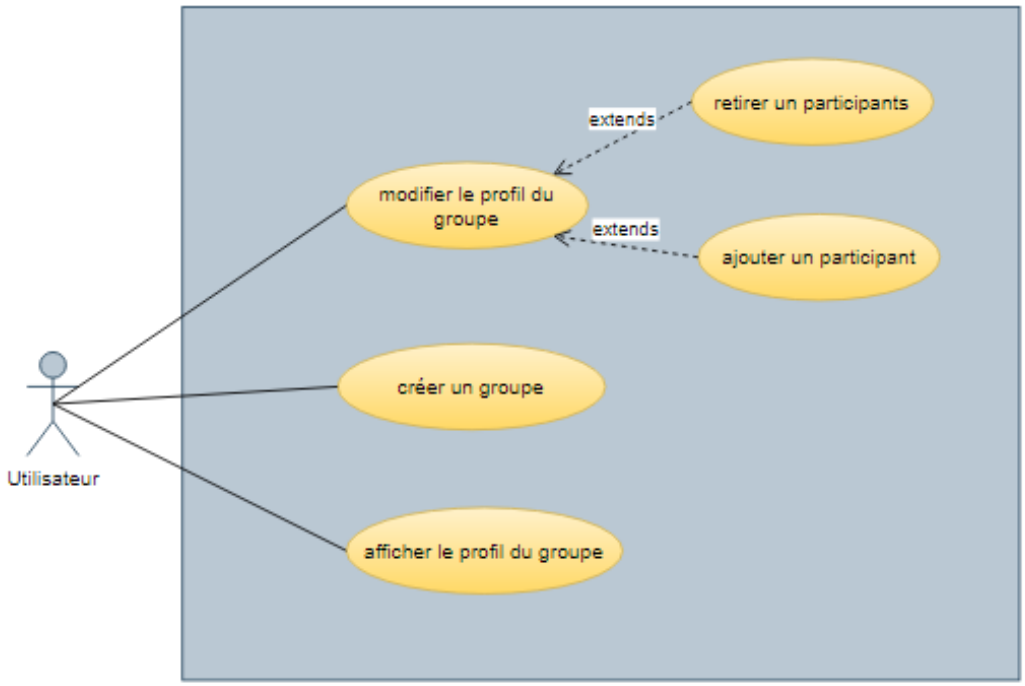

FIGURE 2.6: DCU du module de gestion des groupes

Cas d'utilisation du module de gestion des messages individuels et de groupe

La figure 2.7 présente les actions qu'un utilisateur peut effectuer en ce qui concerne l'envoi de messages individuels et des messages de groupes.

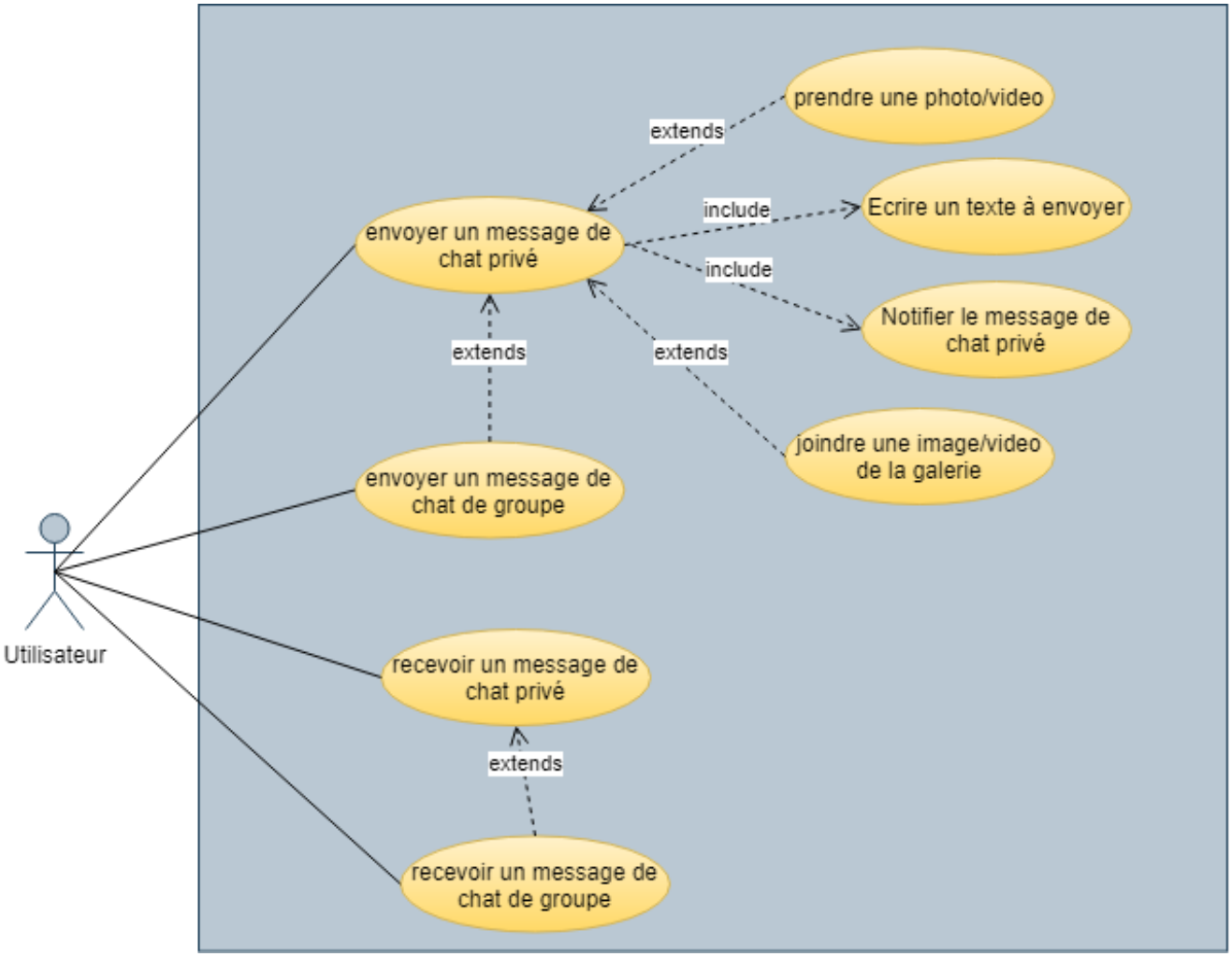

FIGURE 2.7: DCU du module de gestion des messages indiduels et de groupe

Cas d'utilisation du module de gestion des statuts, des posts et des commentaires

La figure 2.8 présente les différents cas d'utilisation liés à la gestion des statuts, des posts et des commentaires.

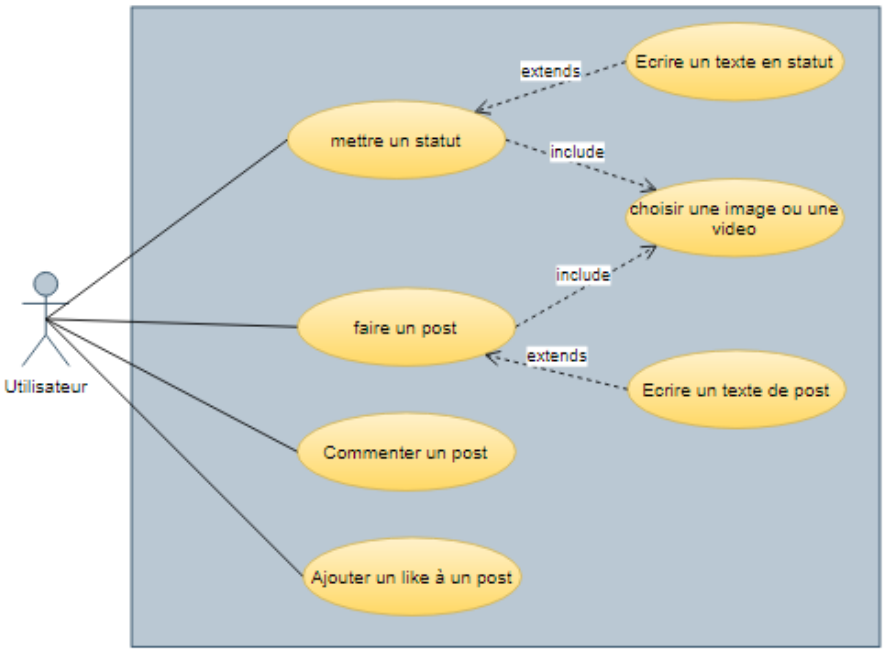

FIGURE 2.8: DCU du module de gestion des statuts, des posts et des commentaires

2.3.4.2 Description textuelle : les formulaires des cas d'utilisation

Le cas d'utilisation "Créer un profil"

- **Description :** il s'agit d'entrer l'ensemble des informations nécessaire afin de créer un profil utilisateur.
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur entre son numéro de téléphone
  2. un code à 6 chiffres est envoyé par sms au numéro mentioné
  3. il entre un code à 6 chiffres et on vérifie
  4. si le code est correct il y a création de son profil
  5. sinon la création du profil échoue
- **Préconditions :**
  - L'utilisateur doit avoir un numéro de téléphone
  - L'utilisateur doit être connecté
  - L'utilisateur doit pouvoir recevoir des sms au numéro mentioné

- **Post-conditions :** Le profil de l'utilisateur est crée
- **Fréquence :** Une seule fois

Le cas d'utilisation "modifier son profil"

- **Description :** Un utilisateur peut modifier les paramètres de son profil.
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur accède à l'interface de modification de profil
  2. Il peut modifier sa photo de profil
  3. Il peut modifier son nom
  4. Il peut modifier sa description
  5. Il enregistre les modifications.
- **Préconditions :** L'utilisateur doit avoir crée un profil utilisateur
- **Post-conditions :** Le profil de l'utilisateur est modifié
- **Fréquence :** Au besoin

Le cas d'utilisation "Ajouter un contact"

- **Description :** Un utilisateur peut entammer une conversation avec un numéro qui ne se trouve pas dans ses contacts, il doit donc enregistrer le numéro en tant que contact dans le téléphone
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur accède à l'application de gestion de contact du téléphone
  2. Il enregistre le numéro comme contact
  3. Il revient à l'interface de l'application messagerie
- **Préconditions :**
  - L'utilisateur doit avoir déjà crée un profil utilisateur
  - L'utilisateur doit avoir une application de gestion de contacts installé sur son téléphone.
- **Post-conditions :** Le numéro enregistré est désormais reconnu comme contact de l'utilisateur dans la messagerie instantanée.
- **Fréquence :** Au besoin

Le cas d'utilisation "Créer un groupe de discussion"

- **Description :** Un utilisateur peut créer un groupe de discussion pour échanger des messages de groupe avec ses contacts
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur sélectionne les membres du groupe dans la liste de ses contacts
  2. Il renseigne le nom du groupe, la description du groupe et optionnellement la photo de profil
  3. Il valide la création du groupe
- **Préconditions :**
  - L'utilisateur doit avoir crée un profil utilisateur
  - L'utilisateur doit avoir des contacts qui utilisent l'application de messagerie
- **Post-conditions :**
  - Le groupe est créé avec l'utilisateur comme créateur du groupe
  - La date de création du groupe est enregistrée
- **Fréquence :** Au besoin

Le cas d'utilisation "modifier le profil d'un groupe de discussion"

- **Description :** Un utilisateur peut modifier les caractéristique d'un groupe qu'il a créé
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. Le créateur du groupe peut modifier soit la photo de profil du groupe, soit le nom du groupe, soit la description du groupe soit la liste des membres du groupe.
  2. Il valide la modification apportée
- **Préconditions :**
  - L'utilisateur doit avoir crée un profil utilisateur
  - L'utilisateur doit être le créateur du groupe
- **Post-conditions :**
  - Les caractéristiques du groupe sont modifiées
  - Les nouvelles caractéristiques du groupe sont communiquées aux membres du groupe
- **Fréquence :** Au besoin

Le cas d'utilisation "Envoyer un message de chat privé"

- **Description :** Un utilisateur peut envoyer un message de chat privé à un autre utilisateur de la messagerie
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur entre dans l'espace de chat du destinataire
  2. Il remplit un texte à envoyer
  3. Il y joint optionnellement une image ou une vidéo qu'il prend via l'appareil photo ou qu'il sélectionne dans la galerie de médias.
- **Préconditions :**
  - L'utilisateur doit avoir crée un profil utilisateur
  - L'utilisateur doit être connecté à internet
- **Post-conditions :** Le message est envoyé au destinataire
- **Fréquence :** Au besoin

Le cas d'utilisation "Recevoir un message de chat privé"

- **Description :** Un utilisateur peut recevoir un message de chat privé envoyé par un autre utilisateur de la messagerie
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur reçoit une notification du message
  2. Il rentre dans la zone de chat privé de l'émetteur du message et voit le message
- **Préconditions :**
  - L'utilisateur doit avoir crée un profil utilisateur
  - L'utilisateur doit être connecté à internet
- **Post-conditions :** Aucune
- **Fréquence :** Au besoin

Le cas d'utilisation "Envoyer un message de chat de groupe"

- **Description :** Un utilisateur peut envoyer un message dans un chat de groupe
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.

- **Scénario nominal :**
  1. L'utilisateur entre dans l'espace de chat de groupe
  2. Il remplit un texte à envoyer
  3. Il y joint une image ou une vidéo qu'il prend via l'appareil photo ou qu'il sélectionne dans la galerie de médias.
- **Préconditions :**
  - L'utilisateur doit avoir crée un profil utilisateur
  - L'utilisateur doit être connecté à internet
- **Post-conditions :** Le message est envoyé à tous les membres du groupe
- **Fréquence :** Au besoin

Le cas d'utilisation "Recevoir un message de chat de groupe"

- **Description :** Un utilisateur peut recevoir un message envoyé par un membre d'un groupe de discussion dans lequel il est membre
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur reçoit une notification du message de groupe
  2. Il rentre dans la zone de chat de groupe et lit le message de groupe
- **Préconditions :**
  - L'utilisateur doit avoir crée un profil utilisateur
  - L'utilisateur doit être connecté à internet
- **Post-conditions :** Le message de groupe est bien reçu
- **Fréquence :** Au besoin

Le cas d'utilisation "Mettre un statut"

- **Description :** Un utilisateur peut mettre une image ou une vidéo en guise de statut que tous ses contacts de la messagerie peuvent voir.
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur sélectionne une image ou une vidéo
  2. Il ajoute un texte et valide l'envoi du statut
- **Préconditions :**

— L'utilisateur doit avoir crée un profil utilisateur

— L'utilisateur doit être connecté à internet

- **Post-conditions :** Le statut est envoyé à tous les contacts de l'utilisateur.
- **Fréquence :** Au besoin

Le cas d'utilisation "Faire un post"

- **Description :** Un utilisateur peut poster une image ou une vidéo avec un texte en plus
- **Acteurs :** tout fournisseur ou consommateur de la marketPlace ayant installé l'application de messagerie instantanée.
- **Scénario nominal :**
  1. L'utilisateur sélectionne une image ou une vidéo
  2. Il ajoute un texte et valide l'envoi du post
- **Préconditions :**
  — L'utilisateur doit avoir crée un profil utilisateur
  — L'utilisateur doit être connecté à internet
- **Post-conditions :** Le post est envoyé à tous les contacts de l'utilisateur qui peuvent commenter ou ajouter un like.
- **Fréquence :** Au besoin

2.3.5 Diagramme de classes

Le diagramme de classes constitue l'un des pivots essentiels de la modélisation avec UML, il permet de donner la représentation statique du système à développer. Cette représentation est centrée sur les concepts de classe et d'associations [16]. L'analyse des données à manipuler dans la solution et les relations entre elles nous ont permis de mettre en évidence le diagramme de classe présenté à la figure 2.9 ci-dessous :

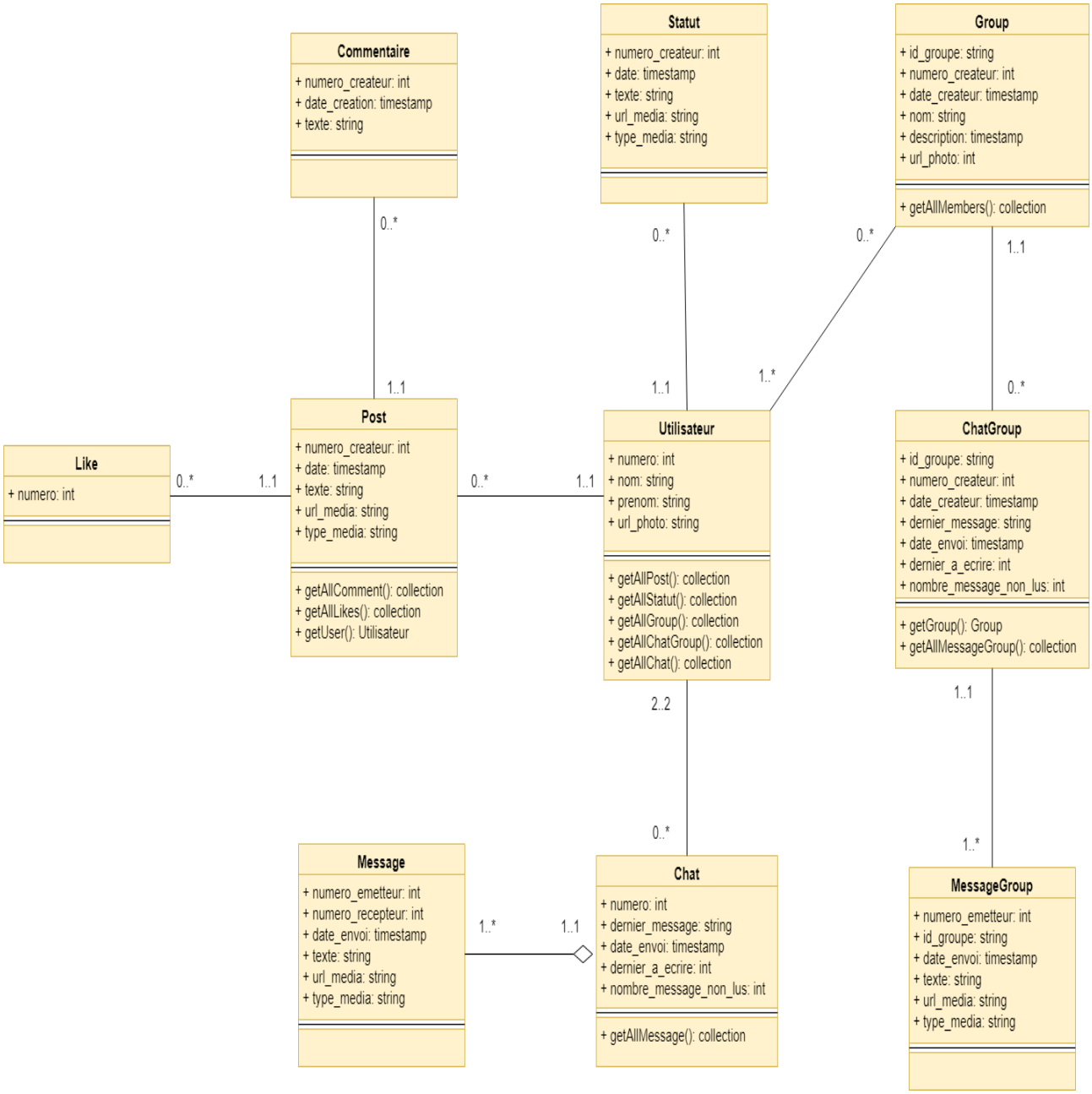

FIGURE 2.9: Diagramme de classes

2.3.6 Diagramme d'activité d'authentification de l'utilisateur

Nous présentons les étapes qui entrent dans le processus d'authentification d'un utilisateur et les différents éléments intervenant :

1. L'authentification d'un utilisateur se fait par l'authentification de son numéro de téléphone.
2. L'utilisateur entre un numéro de téléphone
3. Le serveur de SMS envoit un code d'authentification à 6 chiffres au numéro saisi.

4. si l'utilisateur est le propriétaire du numéro de téléphone alors il reçoit le sms qui contient le code à 6 chiffres.
5. Il entre le code à 6 chiffres et le numéro est authentifié.

La figure 2.10 présente ces étapes sous forme de diagramme d'activité.

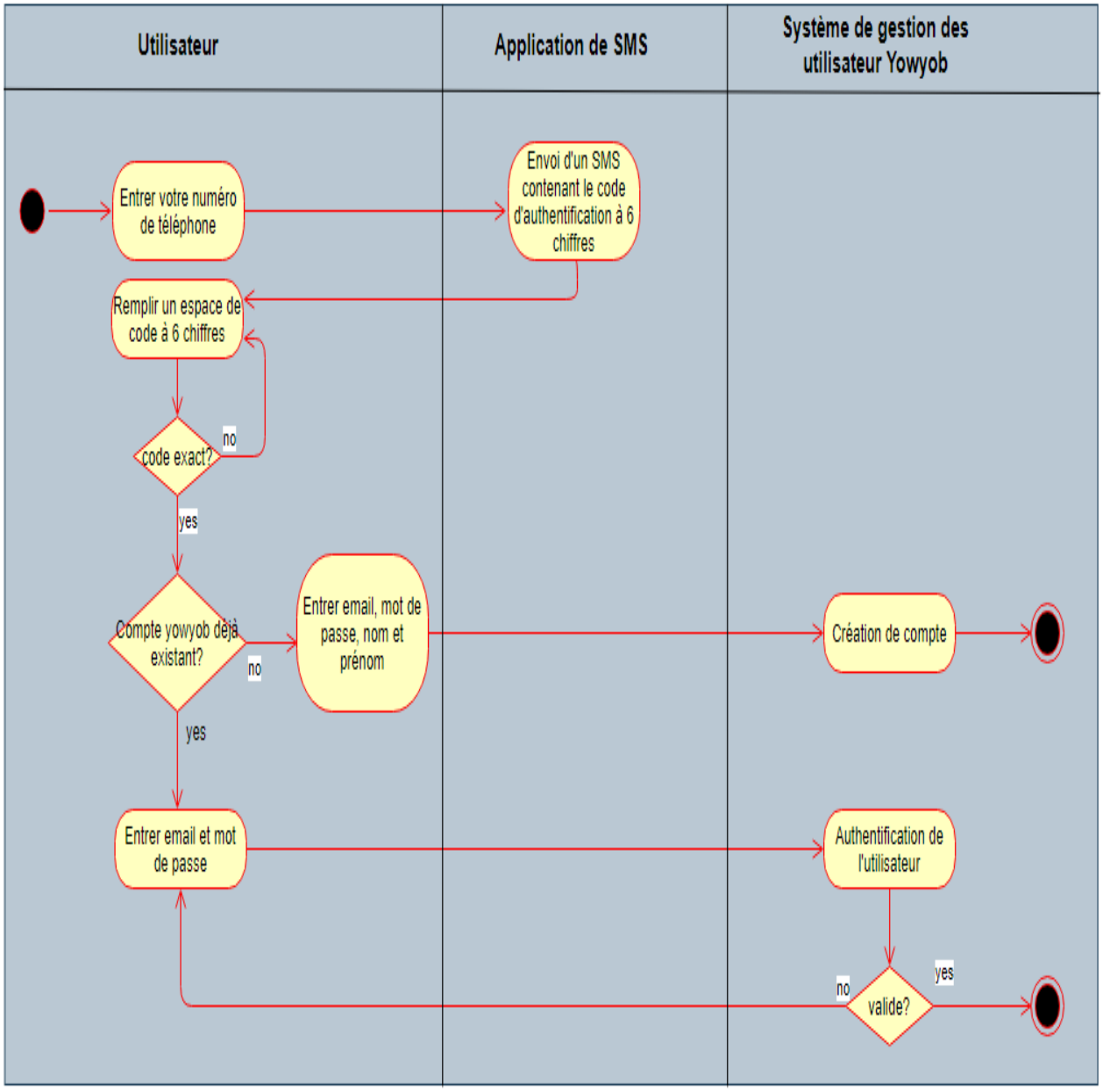

FIGURE 2.10: Diagramme d'activité d'authentification d'un utilisateur

2.3.7 Diagramme d'activité d'envoi de message

Nous présentons les étapes qui entrent dans le processus d'envoi de message et les différents éléments intervenant :

1. L'utilisateur doit être connecté pour envoyer un message.
2. L'utilisateur écrit un message et y associe un fichier image ou vidéos.
3. Le fichier joint est envoyé au serveur de fichier et le message en texte est envoyé au serveur de messagerie.
4. Si le recepteur du message est connecté le serveur de messagerie envoit directement le message au destinataire, sinon il stocke le message dans la base de données du serveur pour une récupération ultérieure.

La figure 2.11 présente ces étapes sous forme de diagramme d'activité entre l'utilisateur et le système de messagerie.

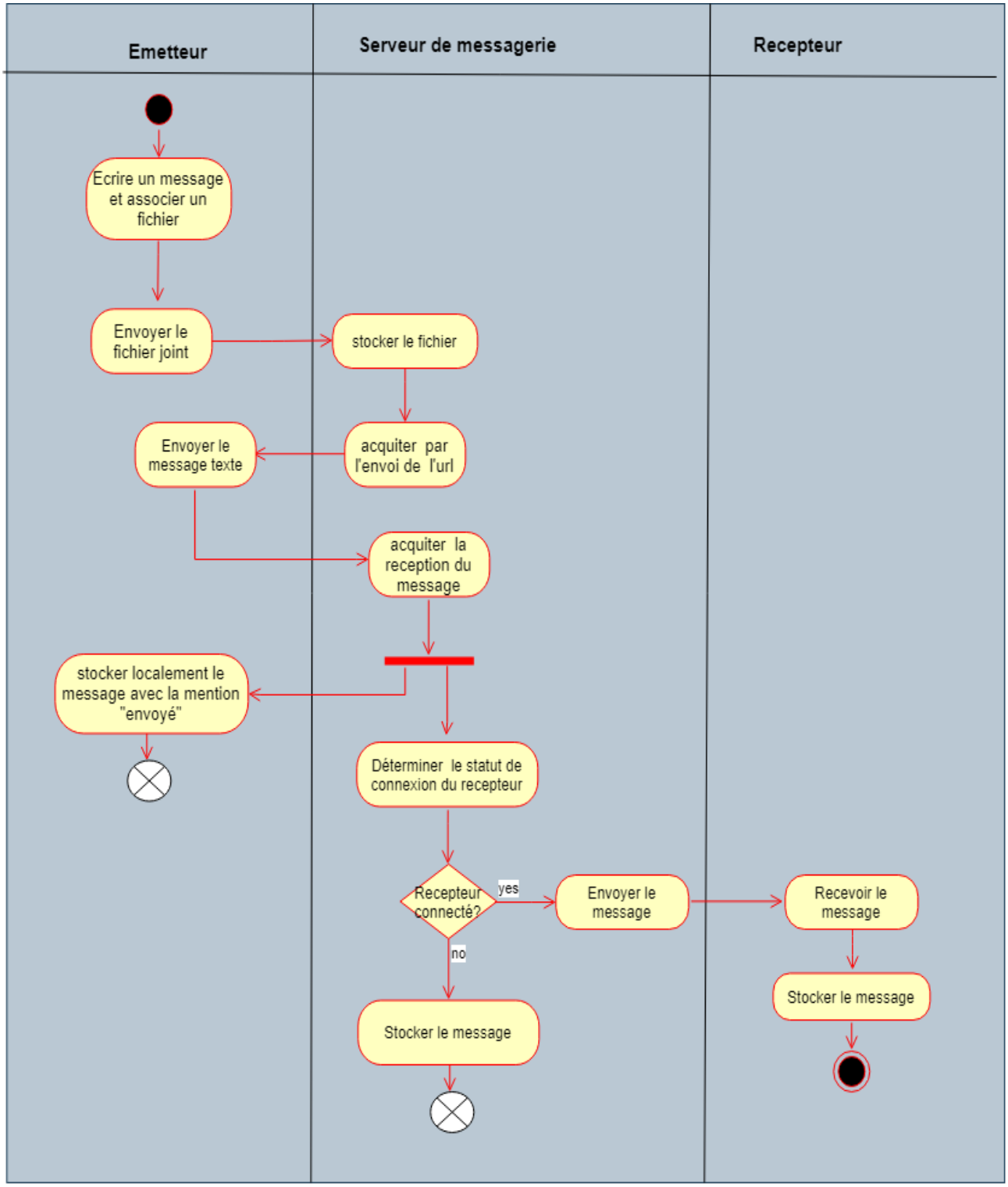

FIGURE 2.11: Diagramme d'activité d'envoi de message

2.3.8 Diagramme d'activité de réception de message

Nous présentons les étapes qui entrent dans le processus de reception de message par un utilisateur juste après qu'il se reconnecte :

1. L'utilisateur doit être connecté pour recevoir des messages.
2. Si l'utilisateur est connecté, il récupère premièrement tous les messages qu'on lui a envoyé lorsqu'il était hors connexion (Ces messages sont stockés au niveau du serveur de messagerie).
3. Il récupère ensuite les fichiers joints associés aux messages qu'il reçoit au niveau du serveur de fichier.

La figure 2.12 présente ces étapes sous forme de diagramme d'activité entre l'utilisateur et le système de messagerie.

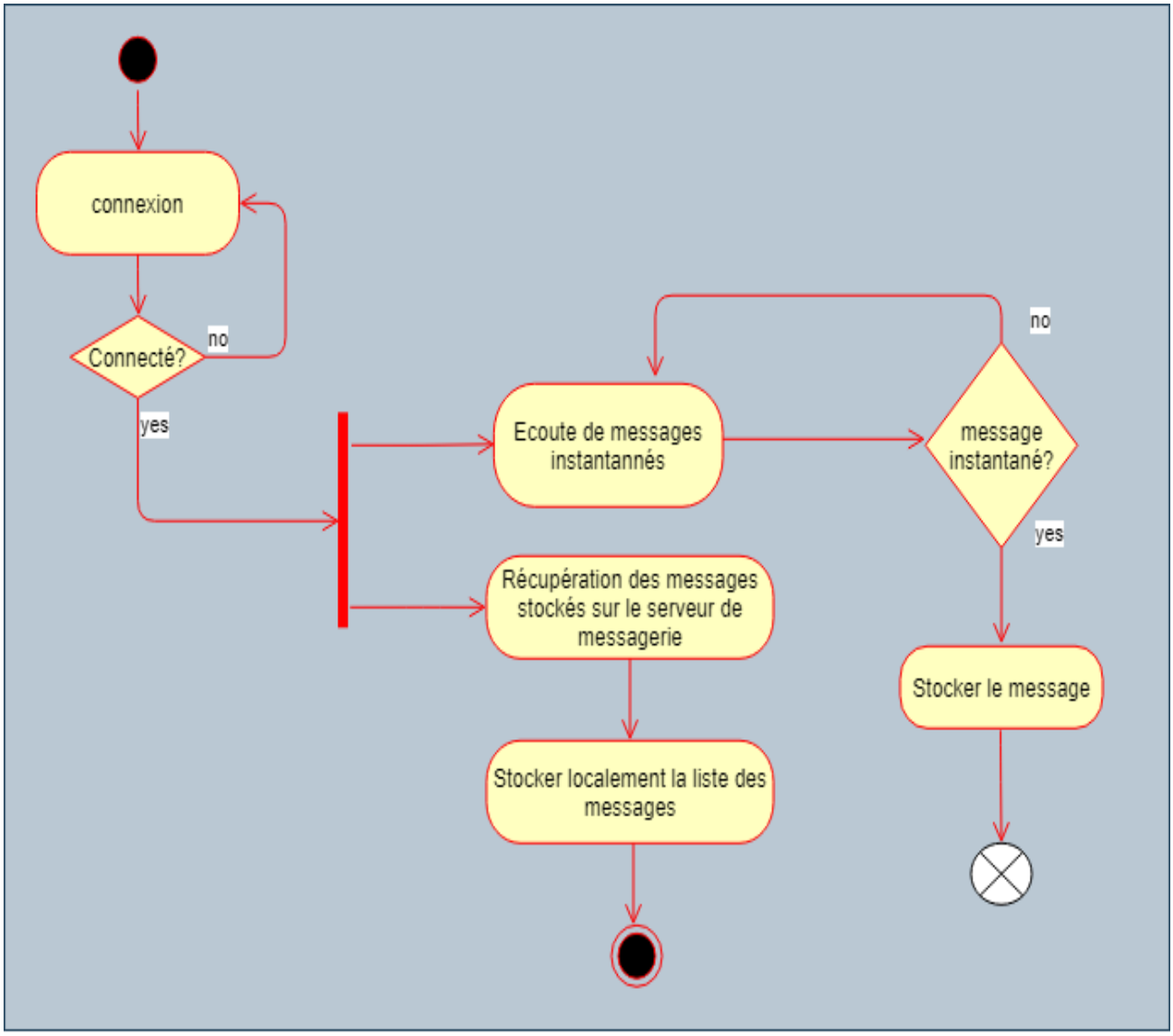

FIGURE 2.12: Diagramme d'activité de réception de message par un utilisateur après sa reconnexion

2.3.9 Diagramme d'état transition

Le diagramme d'état-transition présente le comportement interne d'un objet sous la forme d'un graphique d'états et de transitions. Nous présentons ici le diagramme d'état transition d'un message. Un message peut avoir deux états dans le système au cours du temps : "non envoyé" et "envoyé". Il passe de l'état envoyé à l'état "non envoyé" lorsque l'utilisateur se connecte. La figure 2.13 présente le diagramme d'état transition d'un message.

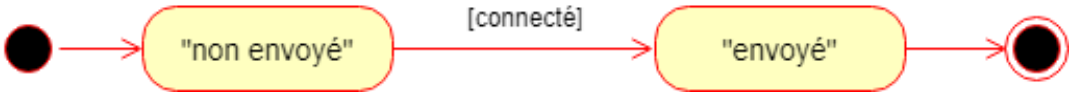

FIGURE 2.13: Diagramme d'état transition d'un message

2.4 Conception de la solution

2.4.1 Architecture de la solution

2.4.1.1 Architecture physique

Le diagramme de déploiement permet de représenter l'architecture physique supportant l'exploitation du système [16]. Nous avons adopté pour notre solution une architecture physique constituée des éléments suivants :

Un Téléphone mobile : Le téléphone mobile héberge l'application cliente de messagerie, une base de données locale, un système de gestion de fichiers, une application de gestion de contact et une application de reception/envoi de SMS.

Le serveur de la messagerie : Le serveur de la messagerie comporte un serveur STOMP responsable de la reception et la rédirection des messages, un serveur d'application qui est responsable du traitement et de la logique metier, un serveur de fichier qui permet le stockage des fichiers de la messagerie et enfin une base de données qui permet le stockage temporaire des messages.

Le serveur d'authentification Yowyob Il fournit une API à travers laquelle un utilisateur de la messagerie peut se connecter ou créer un compte utilisateur pour le système de marketplace Yowyob.

Un serveur de SMS Il fournit une API à travers laquelle on peut authentifier un numéro de téléphone par l'envoi d'un sms contenant un code d'authentification.

L'architecture physique de la solution est donné par le diagramme de déploiement présenté à la figure 2.14.

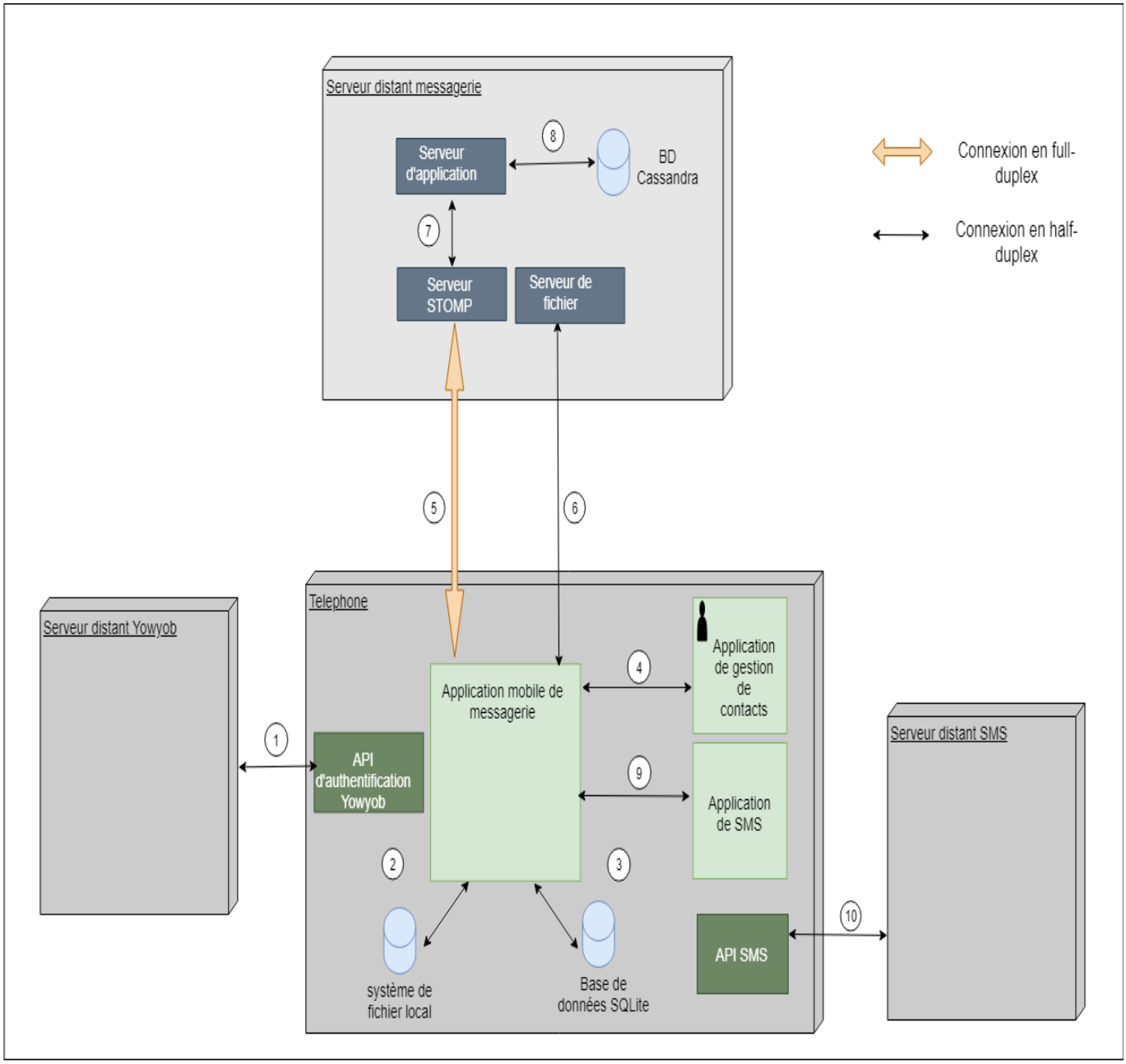

FIGURE 2.14: Diagramme de deploiement

La figure 2.15 décrit les différentes connexions de l'architecture.

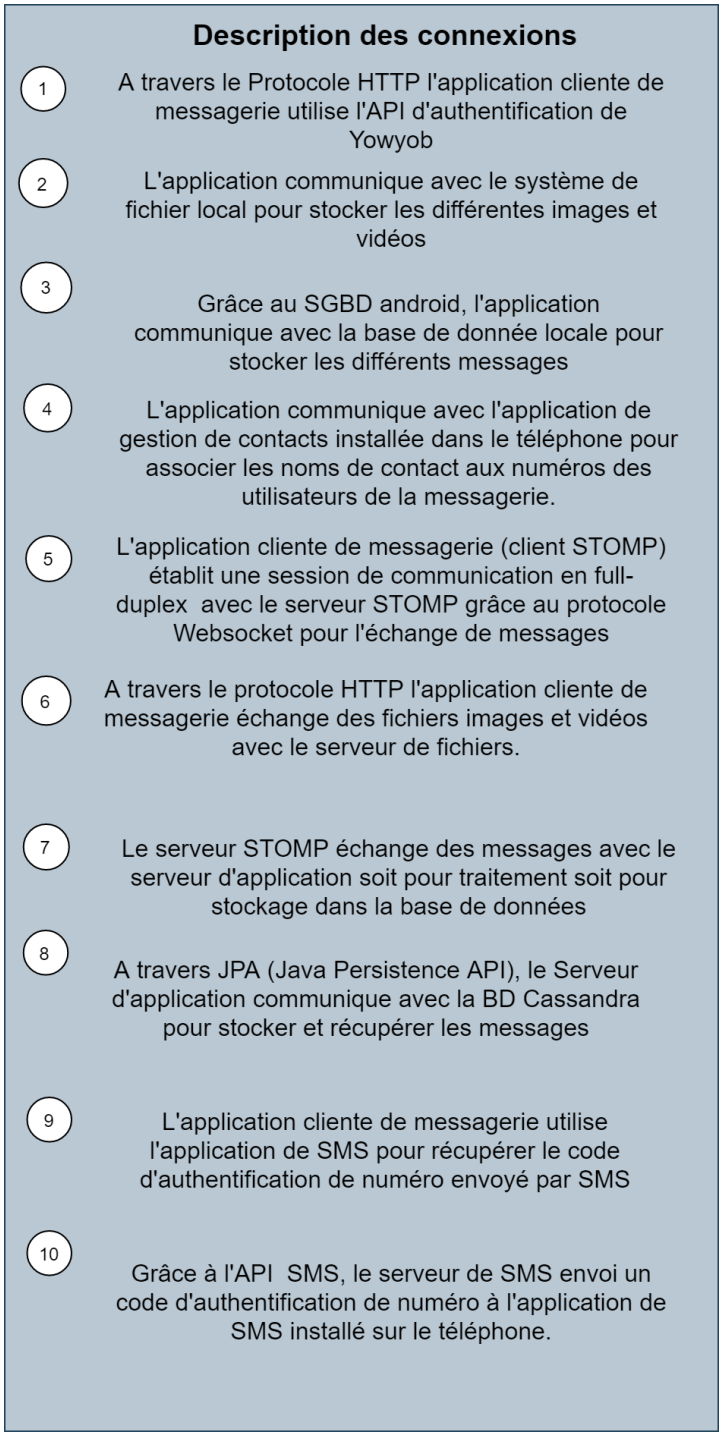

FIGURE 2.15: Description des connexions

2.4.1.2 Architecture logicielle

L'architecture d'un logiciel décrit la manière dont seront agencés les différents éléments d'une application et comment ils interagissent entre eux [**?**]. Elle définit comment le système doit être

conçu de manière logicielle pour répondre aux spécifications. Nous allons présenter l'architecture logicielle au niveau de l'application cliente et au niveau du serveur de messagerie.

Application cliente L'architecture logicielle au niveau de l'application cliente présente les couches suivantes :

- **La couche présentation :** Elle est en contact direct avec l'utilisateur. Elle est constituée de composants collaborant ensemble pour former des interfaces utilisateurs dynamiques ;
- **La couche métier :** C'est la couche qui éffectue les traitements sur l'application cliente. Toute la logique métier y est contenue ;
- **La couche d'accès aux données :** C'est la couche d'abstraction des opérations d'accès aux données.

Serveur de messagerie Il fonctionne suivant l'architecture (Modèle - Vue - Controlleur) :

- **Le modèle :** Il contient les classes d'accès aux données. Nous avons utilisé le pattern Repository pour gérer l'accès aux données. Il permet à travers un ensemble de méthodes d'éffectuer des opérations simples d'ajout, suppression, mise à jour et sélection d'éléments dans la base de données.
- **La vue :** Elle réprésente l'ensemble des résultats fournis à l'application cliente par le serveur de messagerie.
- **Le contrôleur :** Il est chargé de gérer le traitement des requêtes.

2.4.1.3 Architecture de sécurité

L'architecture de sécurité permet d'assurer la protection des données manipulées par le système qui supporte les communications locales, étendues et distantes. Elle permet de répondre éfficacement aux incidents de sécurité qui pourraient se produire. La sécurité est mise en oeuvre au niveau des deux modes d'échanges de données entre les différents noeud de l'architecture physique et aussi au niveau de l'accès à l'application cliente de messagerie instantanée.

Au niveau de l'application Cliente

La sécurité à ce niveau est assurée par l'authentification de l'utilisateur qui souhaite utiliser l'application. L'authentification est un processus visant à assurer la légitimité de la demande d'accès faite par une entité. L'authentification consiste généralement à vérifier que l'utilisateur possède une preuve de son identité ou de son statut sous une des formes suivantes :

- Ce qu'il sait (mot de passe, numéro d'identification personnel etc...)
- Ce qu'il possède (acte de naissance, carte d'identité, carte à puce, Token OTP, diplôme, passeport etc...)

— Ce qu'il est (photo, caractéristique physique, biométrie)

— Ce qu'il sait faire (geste, signature)

L'authentification peut être simple c'est à dire ne reposer que sur un seul élément de vérification ou alors forte c'est à dire reposer sur deux ou plusieurs élements de vérification. Dans le cas de la messagerie instantanée nous optons pour une authentification simple reposant sur le numéro de téléphone comme identifiant de l'utilisateur et un Token OTP envoyé par SMS au numéro de téléphone pour vérifier que le numéro appartient bien à l'utilisateur.

Au niveau des échanges de données entre l'application cliente de messagerie et le serveur de la messagerie

Pour garantir la sécurité dans les échanges de données entre l'application cliente de messagerie et le serveur de la messagerie nous avons opter pour le chiffrement des données échangées en l'occurence dans notre cas le chiffrement des messages. Nous avons deux modes de chiffrement : chiffrement de transport et chiffrement bout en bout. La figure 2.16 présente en image l'évolution de l'état du message (chiffré ou en clair) pour chacun de ces deux modes.

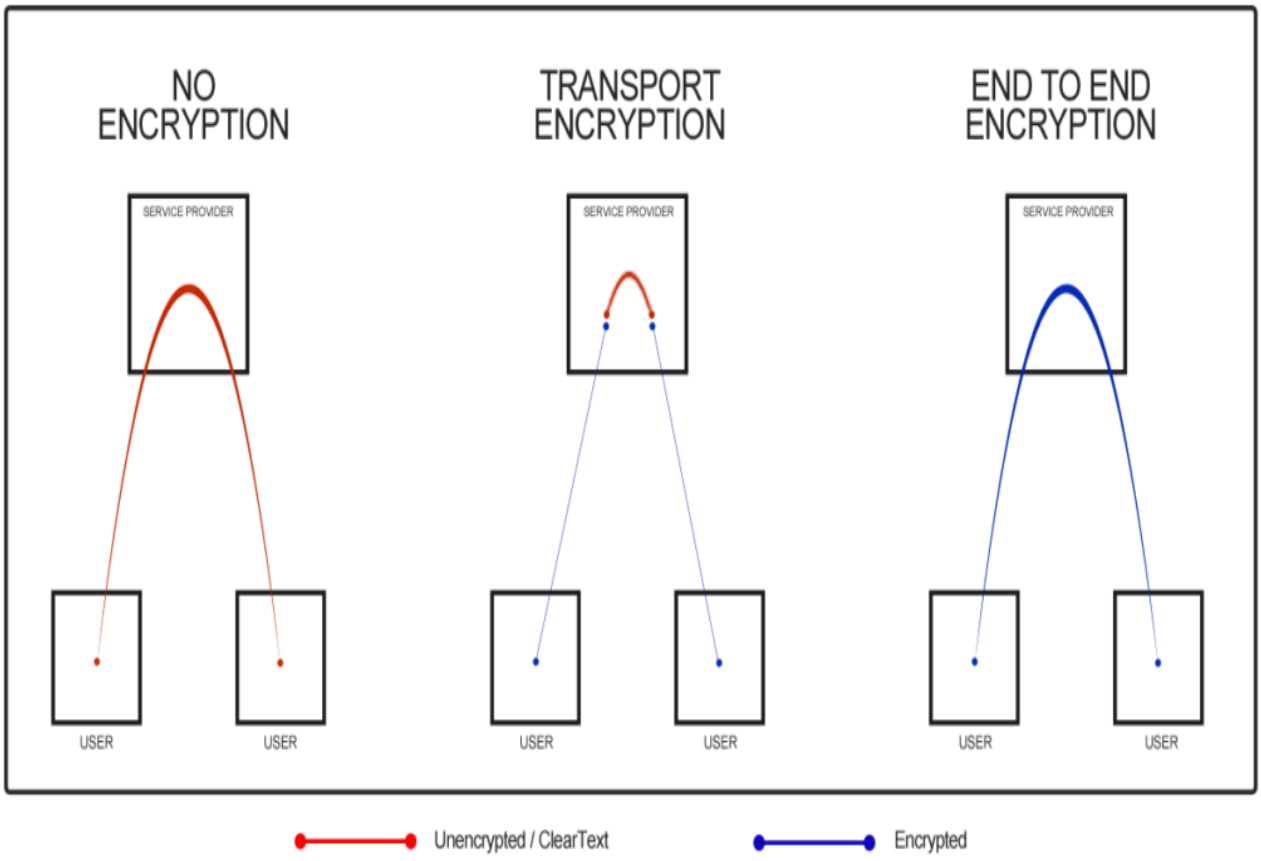

FIGURE 2.16: Les modes de chiffrement

L'échange de messages entre l'application cliente et le serveur est sécurisé et assurée par le protocole de chiffrement signal qui est un protocole de chiffrement bout en bout (End to End encryption en anglais), c'est à dire que l'échange de message entre deux interlocuteurs est chiffré de telle sorte qu'aucun intermédiaire ne puisse lire le message en clair. La figure 2.17 présente en image un scénario d'échange de message chiffré entre deux utilisateurs : Alice et Bob.

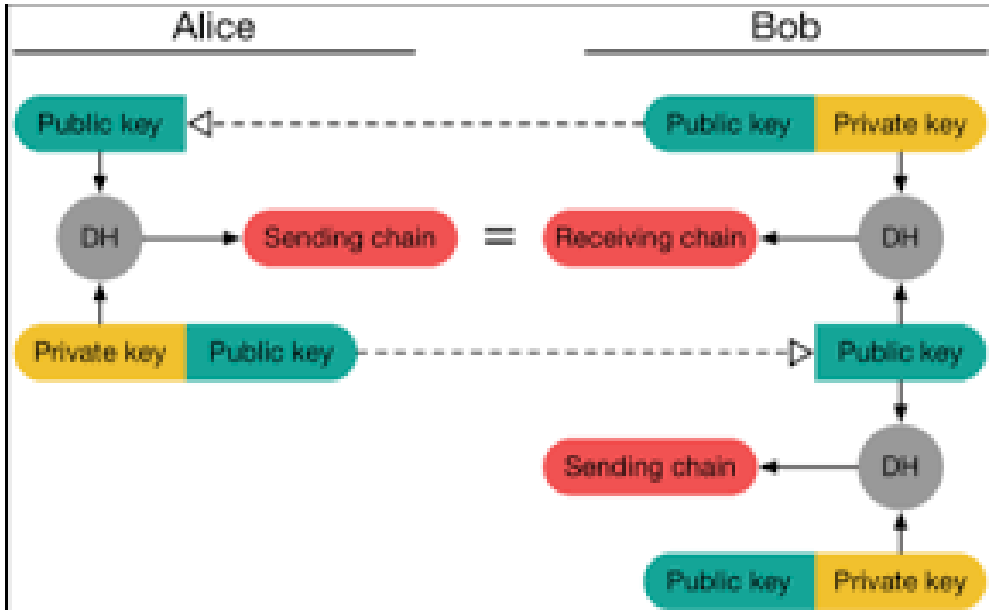

FIGURE 2.17: Chiffrement Signal [3]

Sur la figure 2.17, le processus d'échange se lit comme suit :

1. Lors du tout premier échange entre Alice et Bob, ils génèrent chacun une paire de clés (clé privée + clé publique).
2. Un échange Diffie-Helman de clés a lieu entre Alice et Bob. Alice utilise sa clé privée et la clé publique de bob pour avoir la clé de chiffrement et Bob fait pareil. Cette clé de chiffrement du tout premier échange entre Alice et Bob est appelé "clé de chiffrement racine".
3. Alice utilise la clé de chiffrement racine (DH) pour chiffrer ce premier message et Bob utilise la même clé pour déchiffrer le message.
4. Par la suite on dérive continuellement les clés de chiffrement partant de la clé racine avec une fonction de dérivation de clé pour obtenir de nouvelles clés de chiffrement à chaque échange. La fonction de dérivation utilisée est une fonction pseudo-aléatoire c'est à dire que connaissant la valeur de sortie de la fonction il n'est pas possible d'en déduire la valeur d'entrée. De ce fait, si un attaquant réussit à déterminer une clé de chiffrement de la chaîne des clés de chiffrement, il lui est impossible de pouvoir en déduire les clés précédentes. L'al-

gorithme du double Ratchet est également utilisé pour empêcher un attaquant connaissant la clé de chiffrement pour un message de pouvoir suivre la suite des échanges.

2.4.2 Diagramme de composants

Le diagramme de composant permet de représenter les composants logiciels d'un système ainsi que les liens existants entre ces composants. La figure2.18 présente les composants et leurs interactions.

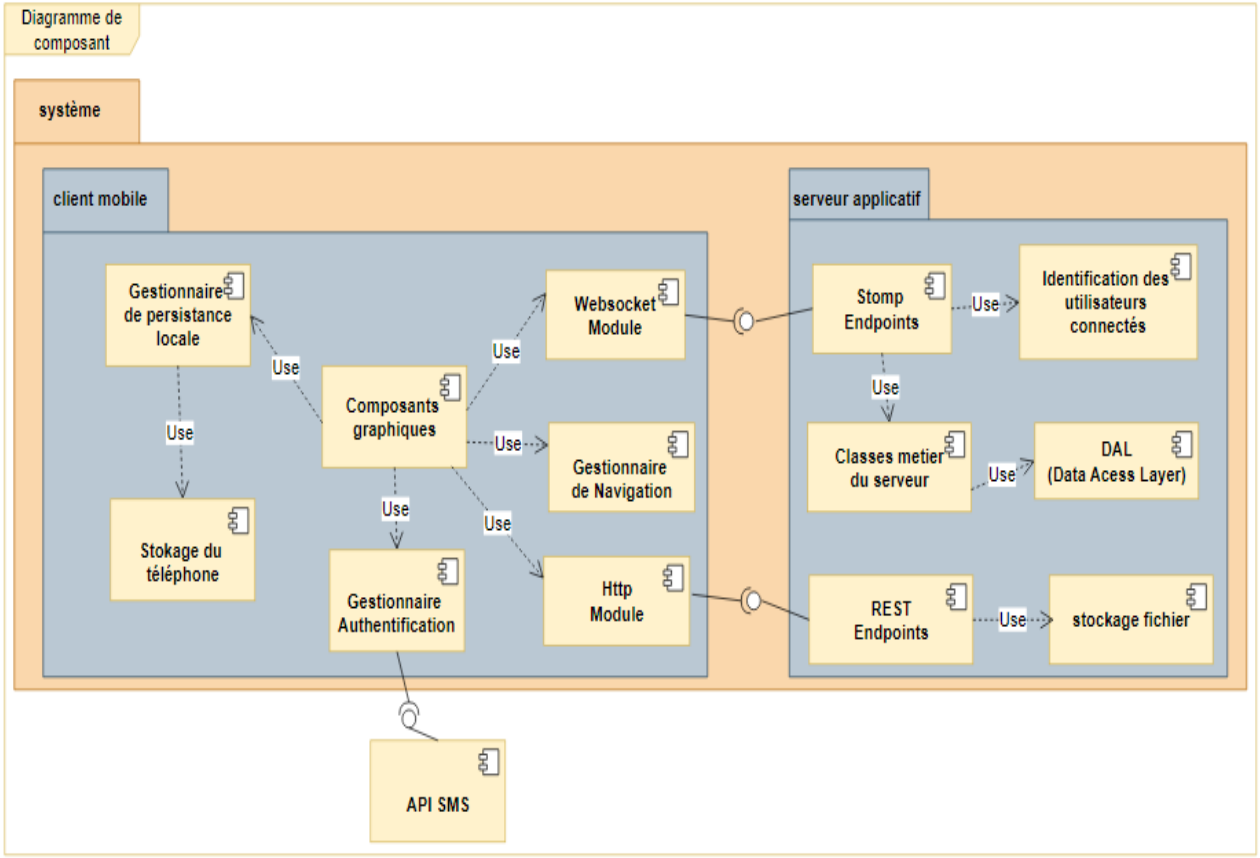

FIGURE 2.18: Diagramme de composants

2.4.3 Diagramme de séquence

L'objectif du diagramme de séquence est de représenter les interactions entre objets en indiquant la chronologie des échanges. Cette représentation peut se réaliser par cas d'utilisation en considérant les différents scénarios associés. Nous présentons ici quelques diagrammes de séquence et une description textuelle de chacun d'eux.

1. **Enregistrer un contact** : L'application mobile communique avec l'application de gestion de contact qui lui fournit un formulaire à remplir pour enregistrer un nouveau contact.

l'utilisateur remplit le formulaire et le nouveau contact est enregistré dans le téléphone. La figure 2.19, présente le diagramme de séquence de ce cas d'utilisation.

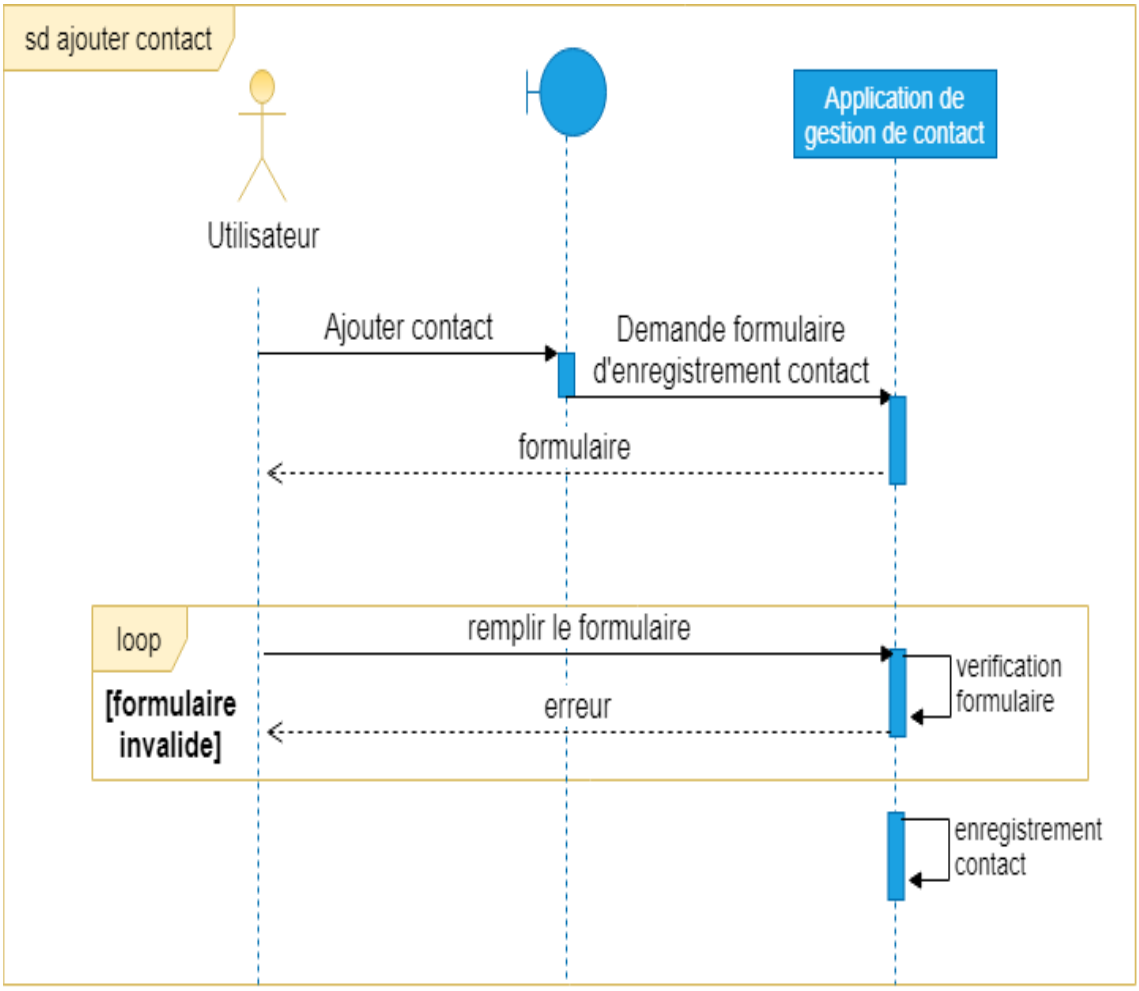

FIGURE 2.19: Diagramme de séquence "Enregistrer un contact"

2. **Envoyer un message de chat privé** : L'application mobile fourni une zone de saisie à l'utilisateur pour écrire un message à un destinataire. Par la suite l'utilisateur peut optionnellement ajouter un fichier joint. L'utilisateur valide l'envoi du message, et le message est stocké dans la base de données du téléphone. Si le fichier est joint, l'application mobile stocke le fichier sur le serveur de fichier et le message est envoyé au serveur STOMP qui envoit directement le message vers le destinataire s'il est connecté, sinon stocke le message dans la base de données du serveur pour que le destinataire puisse le récupérer lorsqu'il se reconnectera. La figure 2.20, présente le digramme de séquence de ce cas d'utilisation

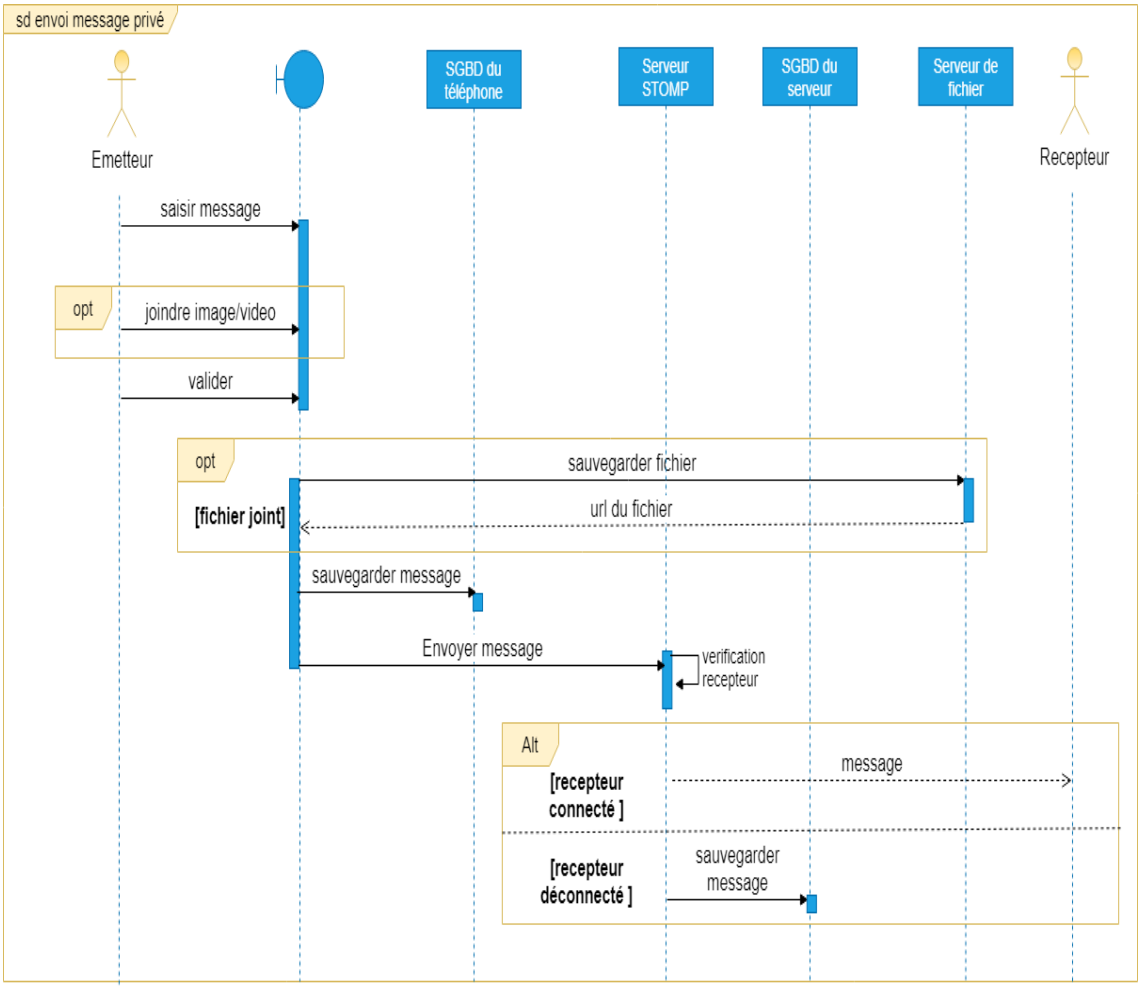

FIGURE 2.20: Diagramme de séquence "Envoyer un message de chat privé"

3. **Envoyer un message de groupe** : L'application mobile fourni une zone de saisie à l'utilisateur pour écrire un message de groupe. Par la suite l'utilisateur peut optionnellement ajouter un fichier joint. S'il ajoute un fichier, celui-ci est stocké au niveau du serveur de fichier et le message est envoyé directement à tous les membres du groupe connectés et stocké sur le serveur pour les membres du groupe déconnectés . La figure 2.21, présente le diagramme de séquence de ce cas d'utilisation

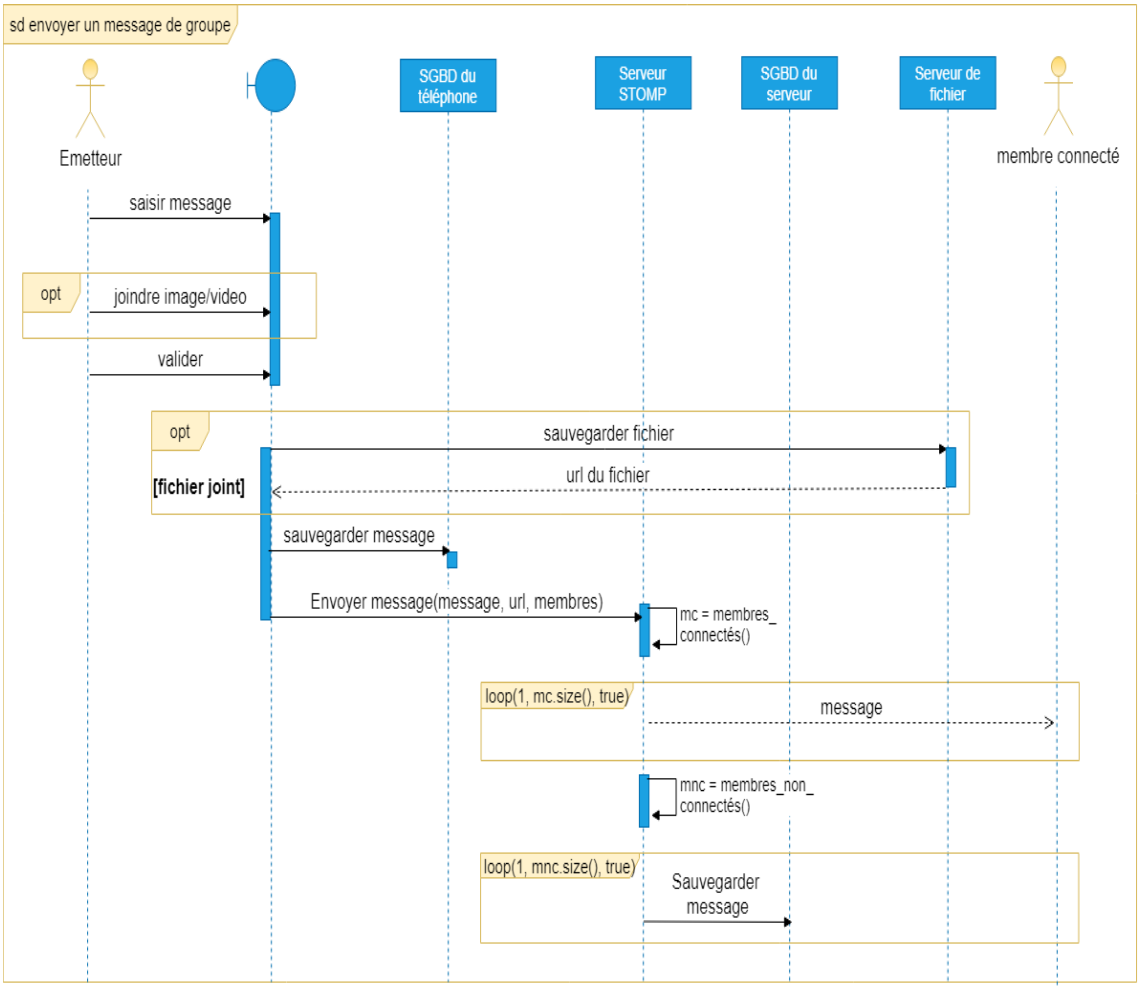

FIGURE 2.21: Diagramme de séquence "Envoyer un message de groupe"

4. **Créer un groupe** : L'utilisateur a la possibilité de créer un groupe de discussion. Il remplit un formulaire de création de groupe qui comporte le nom, la description et il mentionne une liste des membres du groupe. Il valide par la suite et la notification de création de groupe est envoyée à tous les participants connectés et la notification de création est stockée pour tous les membres du groupe qui sont déconnectés. La figure 2.22 en fait une illustration.

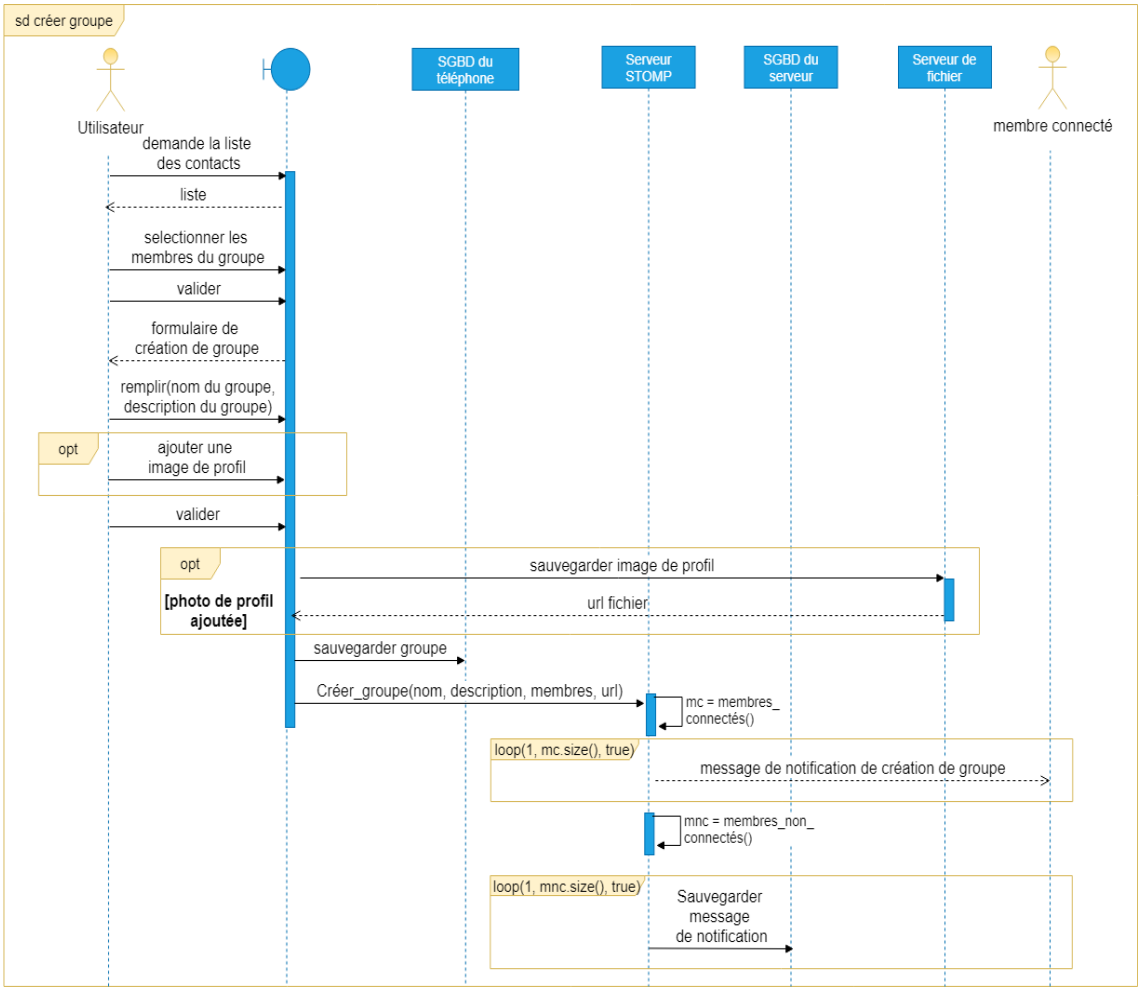

FIGURE 2.22: Diagramme de séquence "créer un groupe"

5. **publier un statut** : L'utilisateur prend une image ou une vidéo soit via la gallerie soit via l'appareil photo et y ajoute du texte. Le fichier image/vidéo est stocké sur le serveur de fichier et le statut est envoyé au serveur STOMP qui envoit directement celui-ci vers tous les contacts connectés de l'utilisateur et stocke le statut dans la base de données du serveur pour tous les contacts déconnectés de l'utilisateur. La figure 2.23 présente le diagramme de séquence de la publication d'un statut.

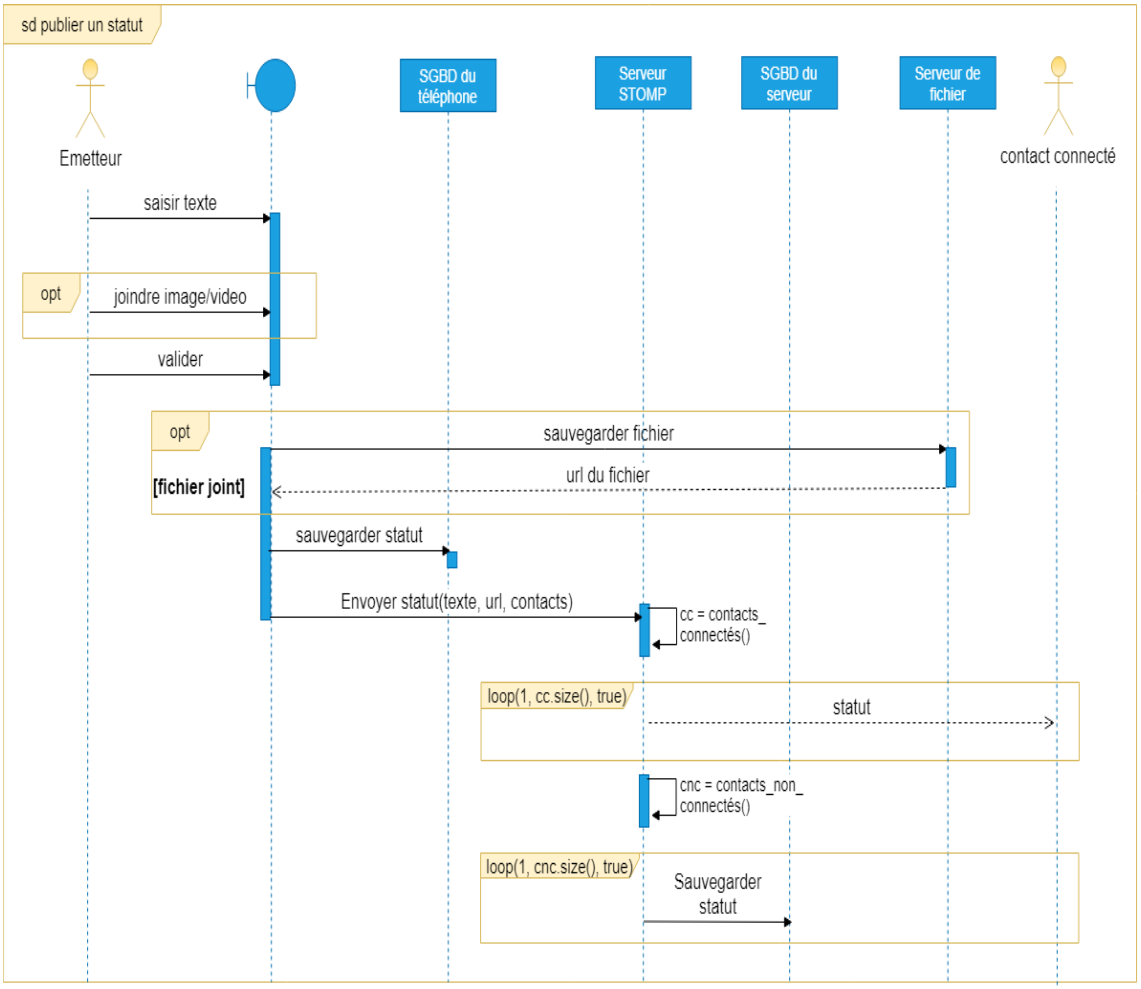

FIGURE 2.23: Diagramme de séquence "publier un statut"

2.5 Bilan du chapitre

Tout au long de ce chapitre nous avons présenté l'approche adoptée pour résoudre notre problème. Premièrement, nous avons fait une modélisation simple du problème, ensuite nous avons fait une analyse dans laquelle nous avons présenté les exigences fonctionnelles et contraintes techniques liées à la solution, les acteurs du système, les différents cas d'utilisation, les diagrammes d'activité et les diagrammes d'état transition. Enfin nous avons présenté la conception de la solution à travers une architecture physique et logicielle de la solution, une architecture de sécurité, des diagrammes de composants et des diagrammes de séquence. Le chapitre suivant concerne l'implémentation de la solution, et présente les résultats obtenus.

CHAPITRE 3

IMPLÉMENTATIONS ET RÉSULTATS

Dans ce chapitre, nous présentons les differents outils utilisés pour implémenter la solution, et les resultats obtenus.

3.1 Outils et technologies utilisés

Java : C'est un langage de programmation orienté. Les logiciels écrit dans ce langage sont compilés dans une représentation binaire intermédiaire et pouvant être exécuté sur une machine virtuelle java (JVM) en faisant abstraction du système d'exploitation. C'est le langage utilisé pour le développement du serveur de messagerie.

Spring Boot : Spring Boot est un framework qui facilite le développement d'applications fondées sur Spring en offrant des outils permettant d'obtenir une application packagée en jar , totalement autonome[17]. C'est le framework utilisé pour développer le serveur de messagerie. La figure 3.1 présente le logo du framework spring.

FIGURE 3.1: Logo du framework spring

IntelliJ IDEA : C'est un environnement de développement intégré destiné au développement de logiciels informatique reposant sur la technologie Java. Il est développé par JetBrains et

disponible en deux versions, l'une communautaire, open Source , sous licence Apache 2 et l'autre propriétaire, protégée par une licence commerciale. Nous utilisons ici la version propriétaire pour développer le serveur de messagerie. La figure 3.2 présente le logo de l'IDE IntelliJ IDEA.

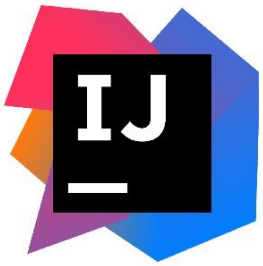

FIGURE 3.2: Logo de l'IDE IntelliJ IDEA

Android Studio : est un environnement de développement d'applications mobiles. Il est basé sur IntelliJ IDEA et utilise le moteur de production Gradle. Il s'agit de l'IDE utilisé pour développer le client de messagerie instantanée. La figure 3.3 présente le logo de l'IDE Android Studio.

FIGURE 3.3: Logo de l'IDE Android Studio

Dart : c'est un langage de programmation orienté objet et optimisé pour les applications sur plusieurs plateformes. Il est utilisé ici pour développer le client mobile de messsagerie instantanée.

Flutter : est un framework de développement de logiciel d'interface utilisateur open-source créé par Google. Il est utilisé pour développer des application Android, iOS, Linux, Mac. Il utilise Dart comme langage de programmation. Il est utilisé pour développer l'application mobile de messagerie. La figure 3.4 présente le logo du framework flutter.

FIGURE 3.4: Logo du Framework Flutter

Apache Cassandra : Il s'agit d'un système de gestion de base de donné de type NoSQL conçu pour gérer des quantités massives de données sur un grand nombre de serveurs assurant une haute disponibilité en éliminant les points de défaillance unique. Il permet une répartition robuste sur plusieurs centre de données[18]. La figure 3.5 présente le logo de la base de données Cassandra.

FIGURE 3.5: Logo de la base de données Cassandra

SQLite : C'est une bibliothèque écrite en langage C qui propose un moteur de base de données relationnelle accessible par le langage SQL. Dans notre cas il est fournit par le SDK android au niveau du mobile et permet un stockage des données au niveau du téléphone. La figure 3.6 présente de la base de données sqlite.

FIGURE 3.6: Logo de la base de données sqlite

Postman : Il s'agit d'un outil permettant de construire et tester rapidement des requêtes HTTP d'API généralement utilisé lors des tests d'intégration. Il est utilisé pour tester l'envoi et la reception de fichiers. La figure **??** présente le logo de l'outil Postman.

Draw.io : C'est un outil en ligne spécialement conçus pour la création des différents diagrammes d'analyse et de conception.

3.2 Résultats

Nous présentons dans cette section, l'application mobile sous forme de captures des différentes interfaces de l'application et les rôles associés.

3.2.1 Scénarios d'utilisation

3.2.1.1 Démarrer l'application

L'application se présente sous forme d'icône dans la liste des applications du téléphone. L'utilisateur lance l'application en cliquant sur l'icône et ensuite accepte les conditions d'utilisation en cliquant sur le bouton "AGREE AND CONTINUE". les figures 3.7 et 3.8 présentent les étapes de démarrage de l'application.

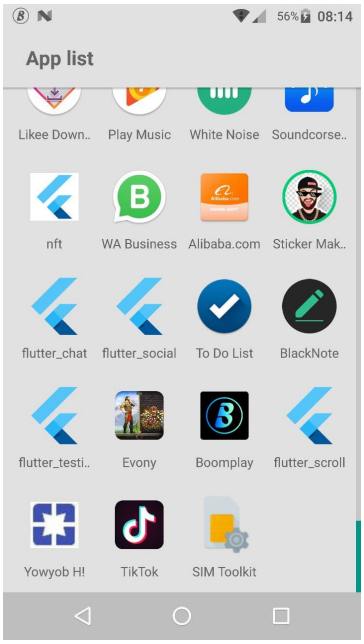

FIGURE 3.7: Icon de l'application mobile

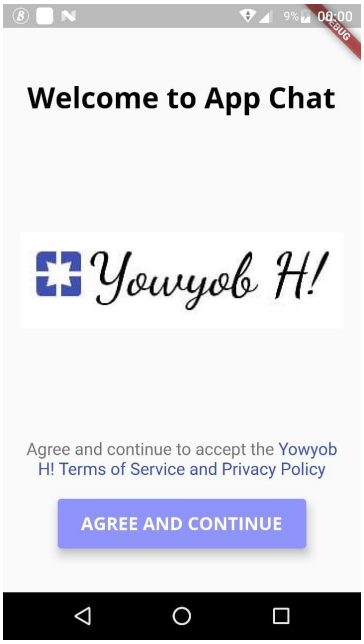

FIGURE 3.8: Accepter les conditions d'utilisation de l'application de messagerie

3.2.1.2 Authentification de l'utilisateur

L'utilisateur doit entrer un numéro de téléphone comme identifiant ainsi que le code de son pays (+237 pour le cameroun). Il valide en appuyant le bouton "NEXT" et est dirigé vers la page d'authentification OTP(One Time Pass) sur laquelle il doit renseigner le code à 6 chiffres qui lui a été envoyé par SMS au numéro de téléphone mentionné. les figures 3.9 et 3.10 présentent les étapes d'authentification du numéro de l'utilisateur.

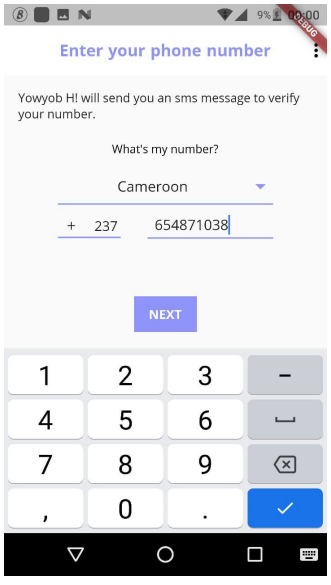

FIGURE 3.9: Entrer le numéro de téléphone

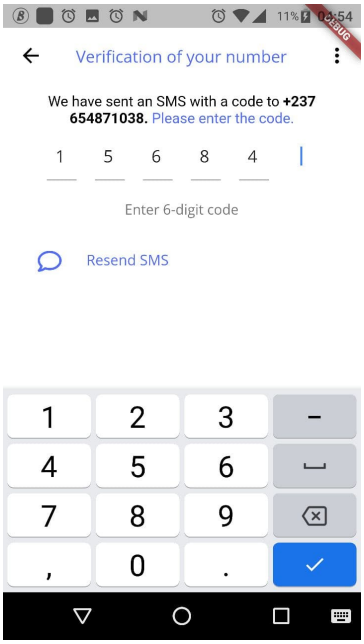

FIGURE 3.10: Vérification OTP du numéro de téléphone

L'utilisateur se connecte par la suite avec son compte de la marketPlace Yowyob ou en crée un nouveau. S'il se connecte, alors il entre les informations relative à la connexion à savoir : son adresse email et son mot de passe. S'il crée un compte alors il ajoute son nom et son prenom enplus de son adresse email et son mot de passe. Les figures 3.11 et 3.12

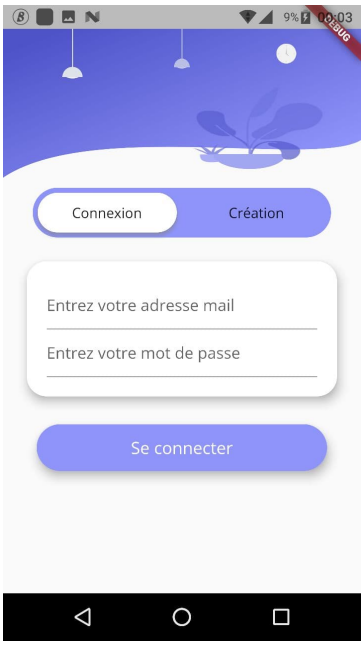

FIGURE 3.11: Connexion à un compte

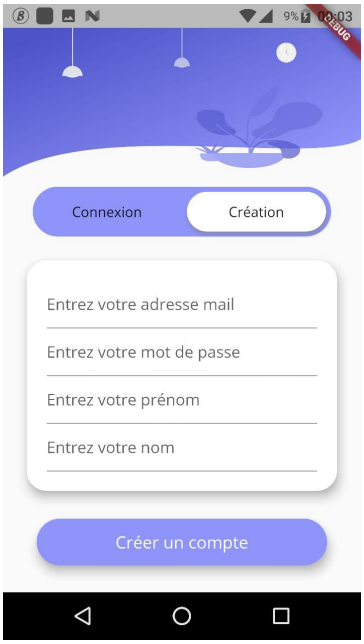

FIGURE 3.12: Création d'un compte

3.2.1.3 Les profils utilisateurs

L'utilisateur peut modifier son profil ou consulter le profil d'un autre utilisateur. L'utilisateur peut modifier son nom, sa description et sa photo de profil en sélectionnant une image dans la gallerie ou alors en prenant directement une photo avec l'appareil photo. Les figures 3.13 et 3.14 présentent respectivement l'interface modification de son profil et l'interface de visualisation du profil d'un contact.

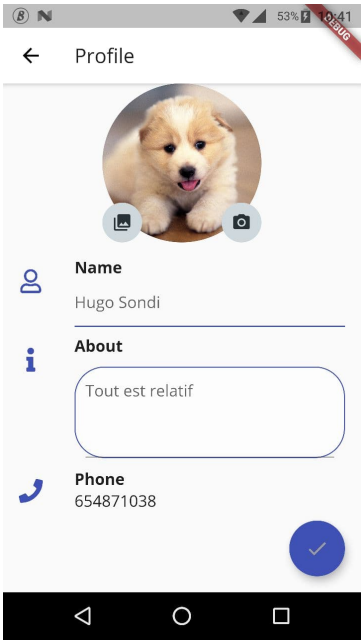

FIGURE 3.13: Consulter ou Modifier son profil

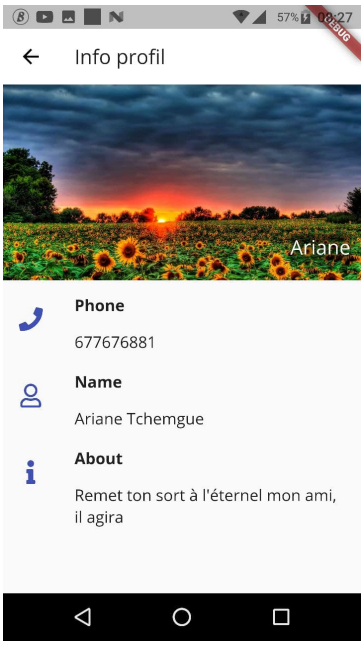

FIGURE 3.14: Consulter le profil d'un contact

3.2.1.4 Création de groupes de discussions

L'utilisateur a la possibilité de créer des groupes de discussions. Il doit sélectionner les membres du groupe parmis ses contacts et définir le nom du groupe, définir la description du groupe et optionnellement ajouter une photo de profil qu'il peut soit choisir dans la gallerie soit

prendre avec l'appareil photo directement. Les figures 3.15, 3.16 et 3.17 présentent les interfaces qui entrent dans le processus de création du groupe.

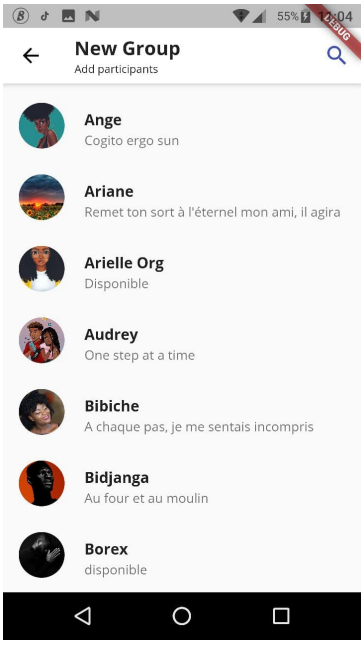

FIGURE 3.15: La liste de ses contacts

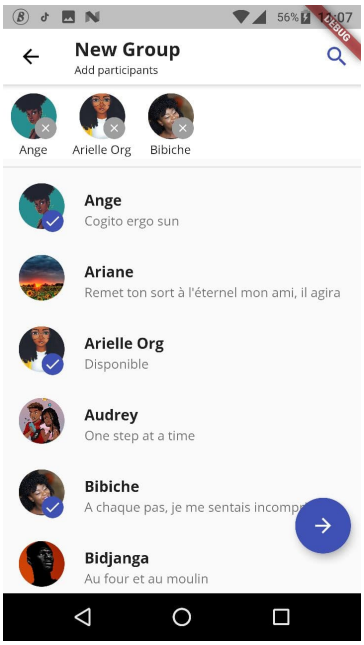

FIGURE 3.16: Sélectionner les participants du groupe parmis ses contacts

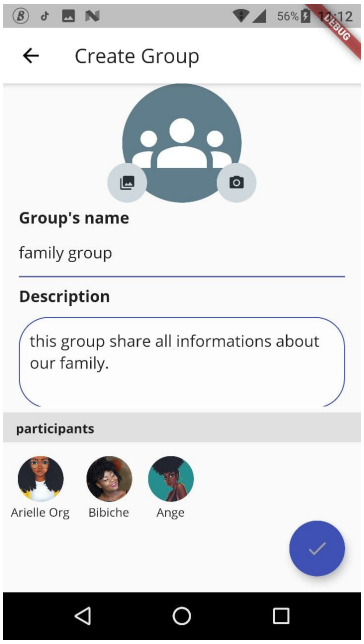

FIGURE 3.17: Ajouter le nom, la description et la photo de profil du groupe

3.2.1.5 Les discussions

L'utilisateur choisit une discussion dans la liste des discussions ou il peut sélectionner un contact et démarrer une nouvelle discussion.

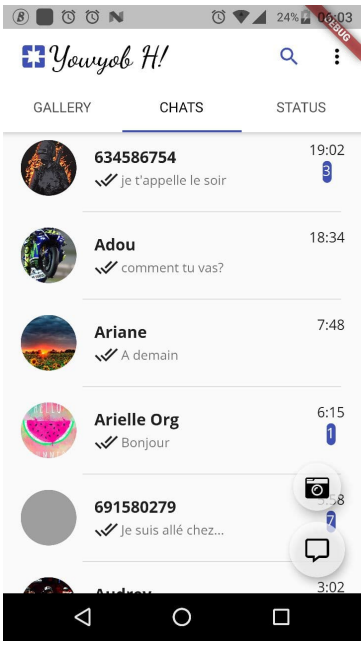

FIGURE 3.18: Choisir une discussion

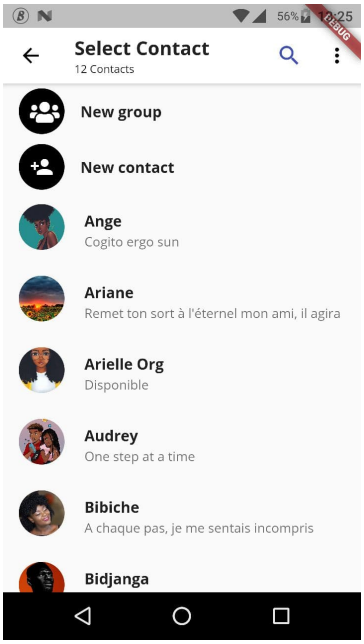

FIGURE 3.19: démarrer une nouvelle discussion avec un contact

L'utilisateur envoi des messages texte à son contact et peut joindre des fichiers images et vidéos qu'il sélectionne dans la gallery d'images et de vidéos ou qu'il prend directement avec l'appareil photo du téléphone. Il a aussi la possibilité d'ajouter des émoticônes dans son message.

FIGURE 3.20: interface d'envoi de messages (1)

FIGURE 3.21: interface d'envoi de messages (2)

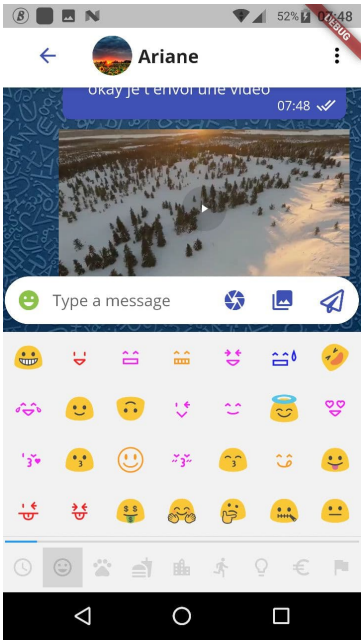

FIGURE 3.22: interface d'envoi de message (3)

Lorsque l'utilisateur reçoit un message il est notifié grâce à une "push-notification" qui se présente comme sur la figure 3.23

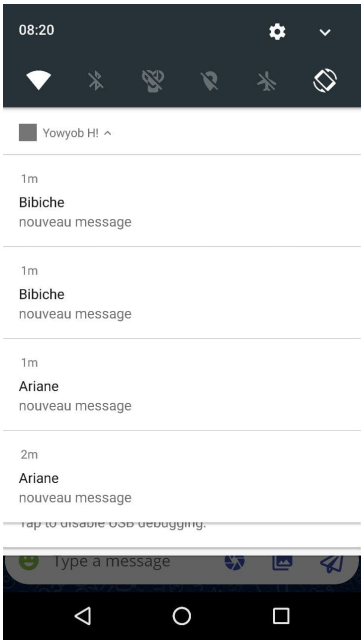

FIGURE 3.23: Notification de message

3.2.1.6 Accès à l'appareil photo et la galerie d'images/vidéos

Lorsque l'utilisateur doit envoyer un message, un post ou un statut il peut y joindre une image ou une vidéo qui peut être soit choisie dans la galerie images/vidéos soit utiliser l'appareil photo. La figure 3.24 présente l'interface de l'appareil photo, on peut prendre une photo grâce à un clique sur le cercle et on peut prendre une vidéo en appuyant longuement sur le cercle.

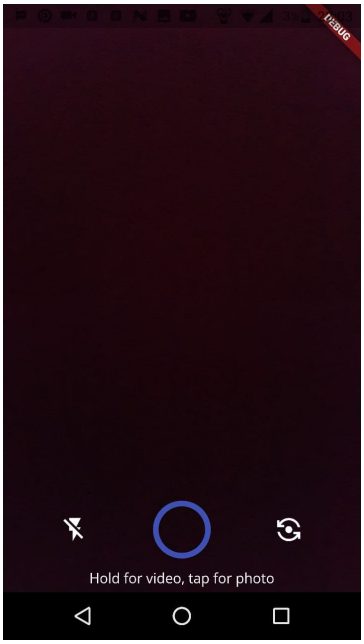

FIGURE 3.24: Utiliser l'appareil photo

La figure 3.25 présente la galerie d'images/vidéos qui permet de visualiser et de choisir les images ou les vidéos qui sont présentes dans le téléphone.

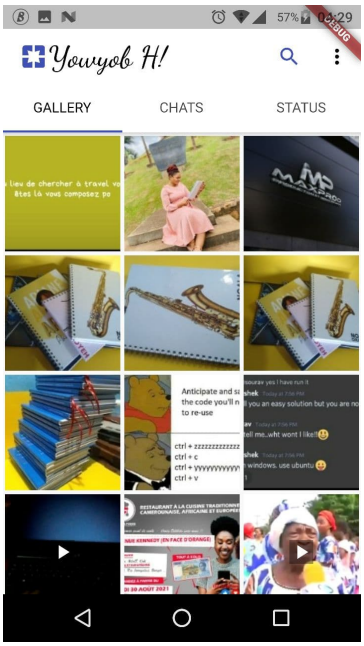

FIGURE 3.25: galerie d'images et de vidéos

L'utilisateur peut choisir une image dans la galerie et y ajouter un texte avant de l'envoyer. La figure 3.26 présente l'interface correspondante.

FIGURE 3.26: Sélectionner une image dans la galerie

L'utilisateur peut choisir une vidéo dans la galerie et y ajouter un texte avant de l'envoyer. La figure 3.27 présente l'interface correspondante.

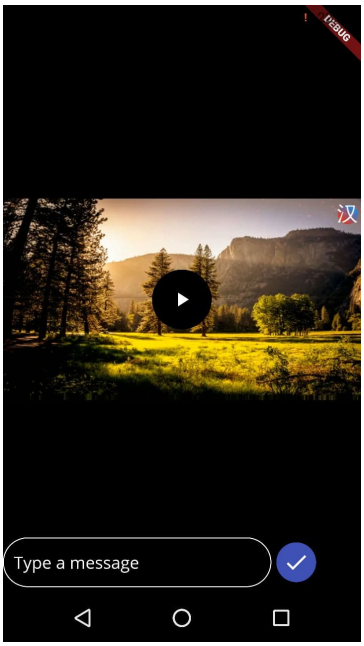

FIGURE 3.27: Sélectionner une vidéo dans la galerie

3.2.1.7 Les Statuts et les Posts d'utilisateurs

L'utilisateur peut mettre un statut ou un post qui sera visible pour l'ensemble de ses contacts qui utilisent l'application de messagerie. Le post peut recevoir des commentaires et des likes. La figure 3.28 présente l'interface d'ajout de post et de story.

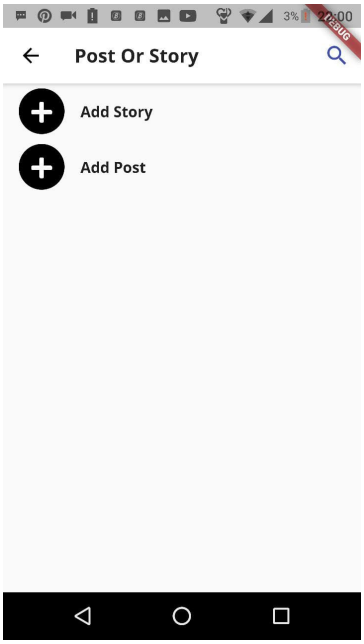

FIGURE 3.28: Ajouter un nouveau post ou une nouvelle Story

La figure 3.29 présente l'espace des statuts(stories) et des posts.

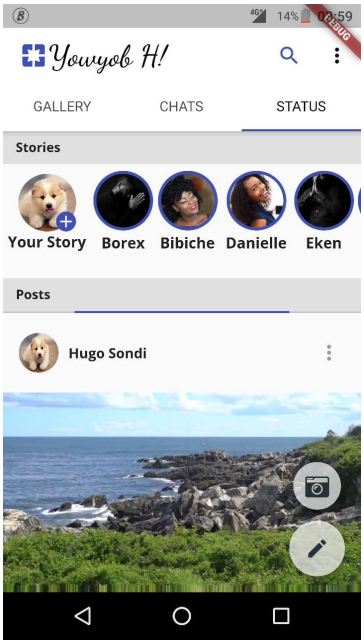

FIGURE 3.29: Interface des stories et posts des utilisateurs

Les figures 3.30 et 3.31 présente l'espace des stories (statuts de la messagerie).

FIGURE 3.30: L'interface de la story d'un contact nommé "Borex"

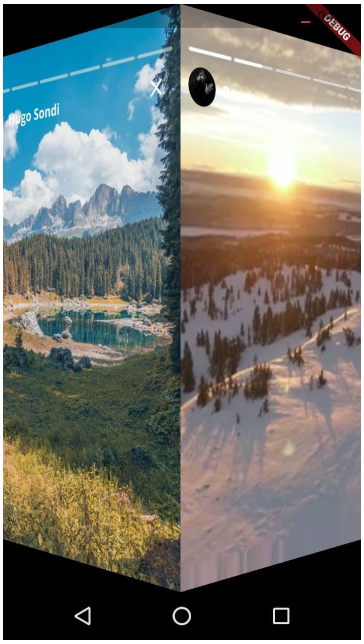

FIGURE 3.31: Transition entre les statuts des utilisateurs

Les figures 3.32 et 3.33 présentent l'espace des posts et des commentaires.

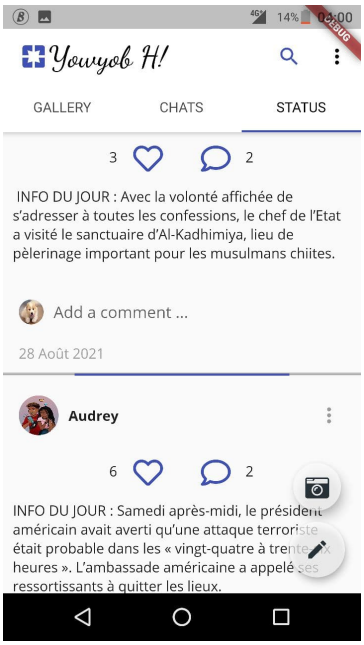

FIGURE 3.32: Les posts de ses contacts

FIGURE 3.33: Commentaires sur un post utilisateur

3.3 Bilan du chapitre

Il était question dans ce chapitre de présenter l'implémentation et les résultats obtenus. Nous avons présenté les outils et les technologies utilisés, nous avons aussi présenté les différents scénario d'utilisation de l'application au moyen d'un ensemble de captures d'écran de la solution développée.

CONCLUSION ET PERSPECTIVES

Bilan

Le processus d'achat de produits et services en ligne se fait communément sur des plateformes de marketPlaces qui mettent en relation fournisseurs et consommateurs. Cependant, la plupart du temps à l'issu des échanges éffectués sur ces plateformes, les consomateurs et les fournisseurs ne restent pas en contact ce qui pourrait pourtant faciliter la communication entre eux lors des prochains échanges. A cet effet, Yowyob .inc LTD a entrepris de mettre en place une application mobile de messagerie instantanée comme système de communication pour les acteurs de sa marketPlace en ligne. La mise en place d'un tel système dégage le problème d'échange temps réel et sécurisé de messages entre un ensemble d'utilisateurs.

Notre démarche pour résoudre ce problème, s'est appuyée sur une étude des approches existantes de mise en place des systèmes de messagerie instantanée mobile et des solutions existantes. A l'issue de celle-ci, nous avons ressorti une modélisation simple et commune à tous les systèmes de messagerie instantanée, puis nous avons effectué une analyse des besoins et une conception détaillée de la solution. Après cette phase de conception, nous sommes passé à l'implémentation de la solution grâce à un ensemble d'outils et technologies dont le framework Flutter pour le développement de l'application mobile de messagerie instantanée et le framework Spring boot pour le développement du serveur de messagerie.

Perspectives

La solution proposée n'est pas parfaite et peut prendre en compte un certain nombre de perspectives. Nous proposons quelques axes d'amélioration pour notre solution à savoir :

— Intégrer un module de compression d'image/vidéo qui sera utilisé avant toute transmission et le module de décompression y correspondant qui sera utilisé à la reception du fichier. Ce module pourra utiliser les approches intelligentes de traitement d'image et de vidéo, et permettra de réduire la consommation de données de l'utilisateur en réduisant la taille des fichiers à transporter.

— Intégrer la sauvegarde des données dans le cloud pour permettre la récupération de l'historique des discussions en cas de perte de données.

Références bibliographiques

[1] Renaud Hébert-Legault. *Rapport technique des Websockets et de la communication en temp réel*. Radboud University, 2016.

[2] Fonctionnement du protocole tls. "https://www.malekal.com/protocole-tls-fonctionnement/". consulté le 2021-09-12.

[3] Fonctionnement du protocole signal. "https://www.synetis.com/signal-messagerie-chiffree/". consulté le 2021-09-15.

[4] Elisabeth Hermann. *Les applications mobiles : analyse et proposition d'un concept de collection pour bibliothèque nationale suisse*. Dunod, Paris, Octobre 2017.

[5] choix technologiques et developpement d'application mobile. "https://ressources.mobizel.com/livre-blanc-choix-technologiques-developpement-application-mobile/". consulté le 2021-08-28.

[6] Brigite NARVOR Dominique MANIEZ. *Echanger et communiquer à distance*. C2IMES, 2006.

[7] Websockets : A conceptual deep dive. "https://ably.com/topic/websockets". consulté le 2021-08-18.

[8] Le fonctionnement du protocole xmpp. "http://wapiti.enic.fr/commun/ens/peda/options/ST/RIO/pub/exposes/exposesrio2009/CLAERHOUT-RICORD/fonctionnement.html". consulté le 2021-08-18.

[9] Stomp architecture basics. "https://www.rfwireless-world.com/Terminology/STOMP-architecture.html". consulté le 2021-08-18.

[10] Stomp protocol specification, version 1.2. "https://stomp.github.io/stomp-specification-1.2.html". consulté le 2021-08-18.

[11] Fondements de la sécurité informatique. "https://www.wooxo.fr/Conseils-Cybersecurite/Principes-securite-informatique". consulté le 2021-08-18.

[12] Transport layer security by chester rebeiro. "http://www.cse.iitm.ac.in/~chester/courses/19e_ns/slides/8_TLS.pdf". consulté le 2021-08-18.

[13] Fonctionnement du protocole ssl. "https://www.cloudflare.com/fr-fr/learning/ssl/what-is-ssl/". consulté le 2021-08-18.

[14] Dion Van Dam. *Analysing the Signal Protocol*. Radboud University, 2019.

[15] Cycle en v en gestion de projet : définition et méthode. "https://www.manager-go.com/gestion-de-projet/cycle-en-v.htm". consulté le 2021-09-15.

[16] David Gabay Joseph Gabay. *UML2 Analyse et Conception*. Dunod, Paris, 2008.

[17] Fonctionnement du framework spring boot. "https://openclassrooms.com/fr/courses/4668056-construisez-des-microservices/5122425-decouvrez-le-framework-spring-boot". consulté le 2021-08-28.

[18] Eben Hewitt. *Cassandra : The Definitive Guide*. O'Reilly, 2011.