



UNIVERSITE D'ANTANANARIVO
Domaine Sciences et Technologies
Mention Mathématiques et Informatique

Mémoire en vue de l'obtention du diplôme de Master 2 en
Mathématiques Informatique et Statistique Appliquées

**Echange de données sécurisé pour
un Système d'assurance en ligne**

Présenté par :

Nasoavina Christinah RAZAFIMANANTSOA

Devant le jury composé de :

Président :	M. Arthur <i>RANDRIANARIVONY</i>	<i>Université d'Antananarivo</i>
Examineur :	M. Hanjarivo <i>LALAHARISON</i>	<i>Université d'Antananarivo</i>
Encadreur :	M. Joelson <i>SOLOFONIAINA</i>	<i>Université d'Antananarivo</i>
Co-Encadreur :	M. Andry <i>RASOANAIVO</i>	<i>Université d'Antananarivo</i>

Ambohitsaina, 30 Juillet 2019

REMERCIEMENTS

Je voudrais exprimer mes gratitudeux aux personnes qui ont contribué à la réalisation de ce mémoire, pour l'expérience et les partages enrichissantes qu'ils m'ont fait vivre.

J'adresse d'abord mes remerciements à Dieu qui n'a cessé de faire don de sa grâce, sa grâce envers moi n'a pas été vaine ;

Je remercie spécialement ma famille pour m'avoir supportée, soutenue et encouragée, et pour m'avoir toujours poussée à avancer dans la vie.

Je remercie Monsieur Andry RASOANAIVO et Monsieur Tsialiva RAJAABELINA fondateurs de ce projet pour m'avoir confié un tel projet me permettant d'acquérir beaucoup d'expériences et d'approfondir mes connaissances. Je les remercie aussi pour avoir apporté conseil dans l'achèvement de mon stage.

Je remercie également Monsieur Joelson SOLOFONIAINA mon encadreur pédagogique, Maître de Conférences au domaine Sciences et Technologies de l'Université d'Antananarivo pour ses conseils concernant la rédaction de ce mémoire et d'avoir témoigné de sa disponibilité et de sa patience.

Je suis vivement reconnaissante envers Monsieur Arthur RANDRIANARIVONY, Professeur titulaire au domaine Sciences et Technologies de l'Université d'Antananarivo, étant le président des Jury et Monsieur Hanjarivo LALAHARISON , Maître de Conférences au domaine Sciences et Technologies de l'Université d'Antananarivo, en tant qu' examinateur qui ont accepté de juger mon travail.

Mes remerciements et reconnaissances s'adressent à tout le corps enseignant au sein de la MISA dirigé par Monsieur Andry RASOANAIVO , Directeur de la MISA et tous les professeurs qui nous ont transmis les connaissances, et conseils durant ces années universitaires.

Enfin, j'adresse mes plus sincères remerciements à ma promotion de la MISA, aux promotions aînées et cadettes, à mes amis et à tous ceux qui ont contribué, de près ou de loin à la réalisation de ce travail.

Merci du fond du coeur.

Table des matières

Liste des abréviations	vii
1 Contexte	2
1.1 Présentation de la start-up IOKAII	2
1.2 Présentation du projet	2
1.3 Problèmes soulevés	2
1.4 Objectif du projet	3
1.4.1 Objectif principal	3
1.4.2 Objectif spécifique	4
1.5 Fonctionnalités attendues	5
1.5.1 Client de l'application	5
1.5.1.1 Phase 1 : Inscription	5
1.5.1.2 Phase 2 : Information sur l'assurance demandée	5
1.5.1.3 Phase 3 : Comparaison de prix et Choix de l'assureur	5
1.5.1.4 Phase 4 : Confirmation et Paiement	5
1.5.2 Renouvellement : Client de la plateforme	5
1.5.2.1 Renouveler une assurance	5
1.5.2.2 Obtenir un autre type d'assurance	6
2 Mise en Oeuvre	7
2.1 METHODE EBIOS	8
2.1.1 Étude du contexte :	9
2.1.1.1 Périmètre de l'étude	9
2.1.1.2 Les métriques prises en compte	10
2.1.2 Étude des événements redoutés :	12
2.1.3 Étude des scénarios de menaces :	13
2.1.4 Étude des risques :	13
2.1.5 Étude des mesures de sécurité	13

3	Conception de la base de données	14
3.1	Choix de l'outil	15
3.1.1	SGBD NoSQL	15
3.1.2	MONGODB	16
3.2	MODELISATION	16
3.2.1	Collection User	17
3.2.2	Document InfoUser	17
3.2.3	Document Type d'Assurance	17
3.2.4	Sous-document Fournisseur d'assurance	17
4	Réalisation de l'Interface	22
4.1	Outil et Technologie	22
4.1.1	HTML ou HyperText Markup Language	22
4.1.2	CSS ou Cascading Style Sheets	22
4.1.3	JavaScript	23
4.1.4	PHP ou Hypertext Preprocessor	23
4.1.5	CodeIgniter 3	23
4.1.6	PhoneGap	23
4.2	Réalisation	24
5	Echange de données	26
5.1	Récupération des données	27
5.1.1	Service Web	27
5.1.2	XML	27
5.1.3	SOAP	27
5.1.4	RPC	28
5.1.5	HTTP	28
5.1.6	HTTP POST	29
5.2	Acheminement des données	29
5.2.1	Gestion et Contrôle des données	30
5.2.1.1	Pour un utilisateur simple	30
5.2.1.2	Pour les systèmes d'informations	30
5.3	Sécurité des transmissions	31
5.3.1	Algorithme de hachage	31
5.3.2	Algorithme de chiffrement	31
5.3.2.1	L'algorithme de cryptographie symétrique	32
5.3.2.2	Algorithme de cryptographie asymétrique	32

5.3.2.2.1	RSA	32
5.3.2.2.2	Echange de clés Diffie-Hellman	32
5.3.2.3	Les algorithmes hybrides	33
5.3.3	XML Signature	33
5.3.4	Algorithme de canocalisation	34
5.3.5	XML Encryption	34
5.3.6	WS Security	34
5.3.7	Authentication	34
5.3.7.0.1	X509	35
Résumé		VIII
Abstract		VIII

Table des figures

1.1	Présentation du projet	3
2.1	Etape de EBIOS	8
3.1	Logo de MongoDB	16
3.2	Démarche d'un nouveau client	18
3.3	Démarche d'un nouveau client de la plateforme	19
3.4	Un type d'architecture de la base de données d'un utilisateur	20
3.5	Un autre type d'architecture de la base de données d'un autre utilisateur	21
4.1	Interface du site internet	24
4.2	Interface de l'application mobile	25
4.3	Interface de l'application mobile	25
5.1	But de l'échange	26
5.2	Message SOAP inclus dans une requête ou une réponse HTTP	29
5.3	Chiffrement RSA	32
5.4	Processus de l'échange	35
5.5	Base de données MongoDB de la plateforme	I
5.6	Enveloppe SOAP	I
5.7	Corps SOAP	II
5.8	Requête SOAP	II
5.9	Extrait du code de l'interface utilisateur	III
5.10	Document échangé	IV

Liste des tableaux

2.1	Tableau de l'échelle de disponibilité	10
2.2	Tableau de l'échelle d'intégrité	10
2.3	Tableau de l'échelle de Confidentialité	10
2.4	Tableau de l'échelle d'Authenticité	11
2.5	Tableau des événements redoutés	12

Liste des abréviations

EBIOS Expression de Besoins et Identification des Objectifs de Sécurité

ISO International Organization for Standardization

SI Système d'Information

SSI Sécurité du Système d'Information

SGBD Système de Gestion de Base de données

SQL Structured Query Language

NoSQL Not Only Structured Query Language

XML EXtensible Markup Language

SGML Standard Generalized Markup Language

W3C World Wide Web Consortium

LDA Analyse Discriminante linéaire

HTTP Hypertext Transfer Protocol

SOAP Simple Object Access Protocol

RPC Remote Procedure Call

SHA Secure Hash Algorithm

RSA Rivest Shamir Adleman

WSS Web Service Security

INTRODUCTION

Dans de nombreux domaines, les clients sont encouragés à opter ou à migrer vers la solution numérique afin de pouvoir tirer profit du progrès technologique. Sous un autre angle, les entreprises doivent donc mettre en place une excellente gestion et une sécurité pour de grandes quantités de données.

De nos jours, plusieurs services sont offerts par les fournisseurs d'assurance, et il nous est devenu primordial et nécessaire d'être assuré par une entreprise. Dans certains domaines, avoir une assurance est même une obligation exigée par la loi. Pourtant, d'une part, les démarches administratives, le temps de déplacement pour les clients posent fréquemment un empêchement et ne permettent pas à tout le monde de bénéficier largement d'une assurance. D'autre part, permettre un accès rapide et facile apportera aux fournisseurs d'assurance beaucoup plus de marchés potentiels à travers une facilité de prestation de service : Un système qui facilitera l'obtention et la délivrance d'assurance.

C'est dans cette perspective que les fondateurs de Iokaii, une Startup en phase de création et de lancement ont eu l'idée de créer un logiciel conforme à un système d'assurance en ligne. Un fournisseur d'assurance peut déjà avoir une solution en ligne mais notre but est de permettre une large vente et facilité d'utilisation au client en utilisant un échange optimisé entre tous les acteurs du monde de l'assurance. Le but de cette recherche est de répondre à la question : Comment permettre à un client de gérer ses assurances s'il décide de s'octroyer diverses assurances dans différents fournisseurs.

Afin de traiter le sujet et de répondre à la question émise, un plan de recherche a été établi. L'intérêt de ce mémoire portera de ce fait sur les travaux réalisés. Il consiste d'abord à la conception et la présentation de l'application dans sa généralité avec le choix des technologies utilisées. Avec les progrès constants de l'informatique, des outils et des langages de programmations permettent de concevoir le logiciel. Le choix des outils et des langages utilisés à la réalisation du logiciel dépend des spécificités du système. Ensuite, derrière ce système, il nous est réservé d'étudier la gestion de la quantité de données à échanger entre les différents acteurs ; une étude qui mérite beaucoup de réflexion au niveau conceptuel. La dernière partie de ce mémoire présentera et développera les travaux réalisés qui concernent la sécurité des échanges dans tous les termes.

Chapitre 1

Contexte

1.1 Présentation de la start-up IOKAII

Iokaii est une startup en cours de lancement, fondée par Mr Andry RASOANAIVO et Mr Tsialiva RAJAABELINA. Le but de cette startup est de pouvoir vendre facilement les produits d'assurance en offrant au client un service rapide, sécurisé, accessible 24h sur 24 et 7 jours sur 7. Le projet est en cours de négociation avec les opérateurs d'assurance et sera mis en activité dès qu'il sera finalisé.

1.2 Présentation du projet

Le projet consiste à concevoir une plateforme comprenant un site internet et une application mobile mais aussi un système d'échange de données sécurisé afin de l'utiliser pour un système d'assurance en ligne. Le système doit être capable de fonctionner avec d'autres systèmes d'information dans le but d'une bonne interopérabilité et distribution de données. Il doit être non-restreint au niveau technologie et facile à adapter pour une évolution future. Le projet se divise en plusieurs modules notamment la réalisation et le développement de l'interface utilisateur, la conception de la base de données, enfin la conception et la réalisation des échanges comprenant la transformation des données, la sécurisation des données, la sécurisation du flux.

1.3 Problèmes soulevés

Concevoir et ensuite permettre un échange de données entre différents systèmes d'information ne sont pas toujours évidents. La première étape est d'abord de rechercher

et de faire beaucoup d'analyses de méthodes pour avoir un système crédible, sécuritaire et digne de confiance. L'étape suivante consiste à réaliser ces études tout en respectant les normes correspondantes.

Un utilisateur peut choisir plusieurs assurances avec des fournisseurs différents. Beaucoup d'informations seront alors circulées et mises à jour progressivement. Le système demandera beaucoup d'informations au fournisseur concerné et ce dernier est invité à soumettre les données demandées. Cela soulève donc deux problèmes, les données sur les utilisateurs peuvent se désordonner et ne respecteront pas les droits de propriétaires. Ensuite, les fournisseurs d'assurance ont leur propre architecture de système d'information. Quelle méthode permettra la sécurisation des requêtes au niveau des différents systèmes d'information des fournisseurs sachant que l'enjeu de l'application est de fournir l'assurance en tout temps avec le minimum de temps possible et un meilleur garanti.

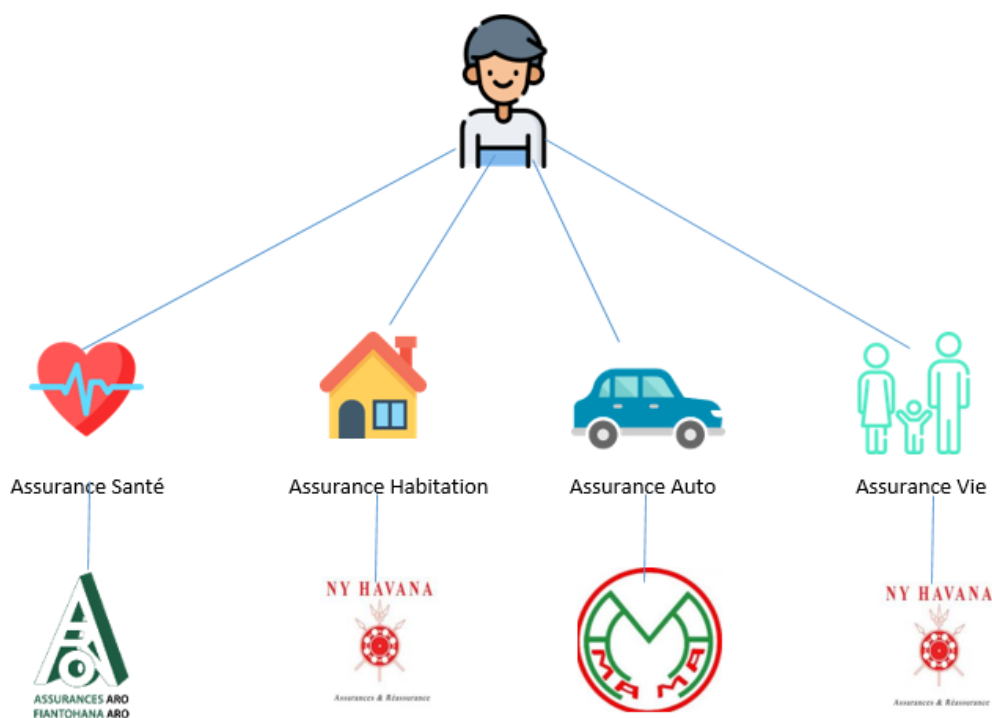


FIGURE 1.1 – Présentation du projet

1.4 Objectif du projet

1.4.1 Objectif principal

Faire un travail qui respecte les normes et méthodes de travail reconnues dans chaque étape : Conception, Choix des outils et Réalisation.

1.4.2 Objectif spécifique

Le but de ce travail est de

- Créer un système qui ne dépend pas d'un système d'information particulier
- Avoir un système équivalent à une base de données d'escale rassemblant en un toutes les informations sur un utilisateur
- Créer un système qui peut interagir avec n'importe architecture pour permettre la sécurité des informations dans toutes ses composantes.

En gros, à partir de la plateforme, un utilisateur peut gérer toutes ses assurances.

1.5 Fonctionnalités attendues

1.5.1 Client de l'application

Pour chaque type de client, le système offre différentes démarches pour fournir l'assurance. Ci-dessous, nous présentons les étapes à suivre pour une première inscription.

1.5.1.1 Phase 1 : Inscription

Si le client est nouveau à la plateforme et n'est pas encore inscrit à aucune assurance, il s'inscrit à la plateforme en remplissant les informations lui concernant.

1.5.1.2 Phase 2 : Information sur l'assurance demandée

Le client entre ensuite les informations nécessaires à l'assurance qu'il demande. Pour chaque type d'assurance, les informations demandées sont différentes.

1.5.1.3 Phase 3 : Comparaison de prix et Choix de l'assureur

Selon les informations et besoins demandées par le client, la plateforme affiche la comparaison de prix et l'utilisateur pourrait choisir le fournisseur qui lui convient.

1.5.1.4 Phase 4 : Confirmation et Paiement

L'utilisateur confirme son inscription à la plateforme par le paiement de son service via mobile Banking. Un résumé de ses besoins sera affiché et après confirmation du paiement, il devient un client de la plateforme, un identifiant unique lui est créé permettant de l'identifier dans tout le système.

1.5.2 Renouvellement : Client de la plateforme

1.5.2.1 Renouveler une assurance

Pour renouveler son assurance, l'utilisateur entre le numéro de police de l'assurance qui n'a plus cours. L'application affiche les informations concernant ce numéro de police. Si le client confirme son renouvellement, la plateforme affiche les tableaux de comparaison. Deux choix sont possibles à l'utilisateur :

a. Rester au même fournisseur

Si l'utilisateur décide de rester à son assureur habituel, les informations du client dans le système sont mises à jour et un nouveau numéro de police lui est délivré.

b.Choisir un autre fournisseur

L'utilisateur devient client de l'assureur choisi. Les données le concernant seront transmises à l'assureur.

1.5.2.2 Obtenir un autre type d'assurance

Au fur et à mesure du temps, le client peut demander une assurance d'un nouveau matériel à son nom. Il est tout de suite redirigé dans la phase2 de l'application et continue les procédures habituelles. A la phase4 terminée, l'information sur l'identifiant de ce client sera mise à jour, autant que celle de l'assureur choisi.

Chapitre 2

Mise en Oeuvre

Il y a risque de sécurité de l'information dès lors qu'on a conjointement :

- une source de menace
- une menace
- une vulnérabilité
- un impact

On peut ainsi comprendre qu'il n'y a plus de risque si l'un de ces facteurs manque. Or, il est extrêmement difficile, voire dangereux, d'affirmer avec certitude qu'un des facteurs est absent. Par ailleurs, chacun des facteurs peut contribuer à de nombreux risques différents, qui peuvent eux-mêmes s'enchaîner et se combiner en scénarios plus complexes, mais tout autant réalistes. Une bonne sécurité assure :

- La confidentialité : seul l'expéditeur et le receveur ont accès au contenu du message. La confidentialité est compromise si une entité non autorisée peut consulter le contenu.
 - L'authenticité : qui concerne l'identification des échanges. Elle assure que l'originaire du document est bien identifié.
 - L'intégrité : aucun changement du contenu du document n'a été effectué tout au long de la transmission.
 - La non-répudiation : L'expéditeur ne pourra nier qu'il a envoyé son message.
 - La contrôle d'accès : détermine les droits d'accès des utilisateurs du document.
 - La disponibilité : La ressource doit être disponible pour les utilisateurs autorisés en tout temps.
-

Afin de répondre à l'objectif principal du projet, nous avons choisi une méthode de travail qu'on a appliquée dans l'étude.

2.1 METHODE EBIOS

La méthode EBIOS (Expression des Besoins et Identification des Objectifs de Sécurité) est un outil complet de gestion des risques liées à la sécurité des systèmes d'information (SSI) conforme aux normes ISO 27001, 27005 et 31000 [6]. Elle a été créée en 1995 par l'ANSSI, la méthode EBIOS permet d'apprécier et de traiter les risques relatifs à la sécurité des systèmes d'information (SSI). Elle permet aussi de communiquer à leur sujet au sein de l'organisme et vis-à-vis de ses partenaires, constituant ainsi un outil complet de gestion des risques SSI.

La méthode comprend cinq étapes : Étude du contexte, Étude des événements redoutés, Étude des scénarios de menaces, Étude des risques, Étude des mesures de sécurité.

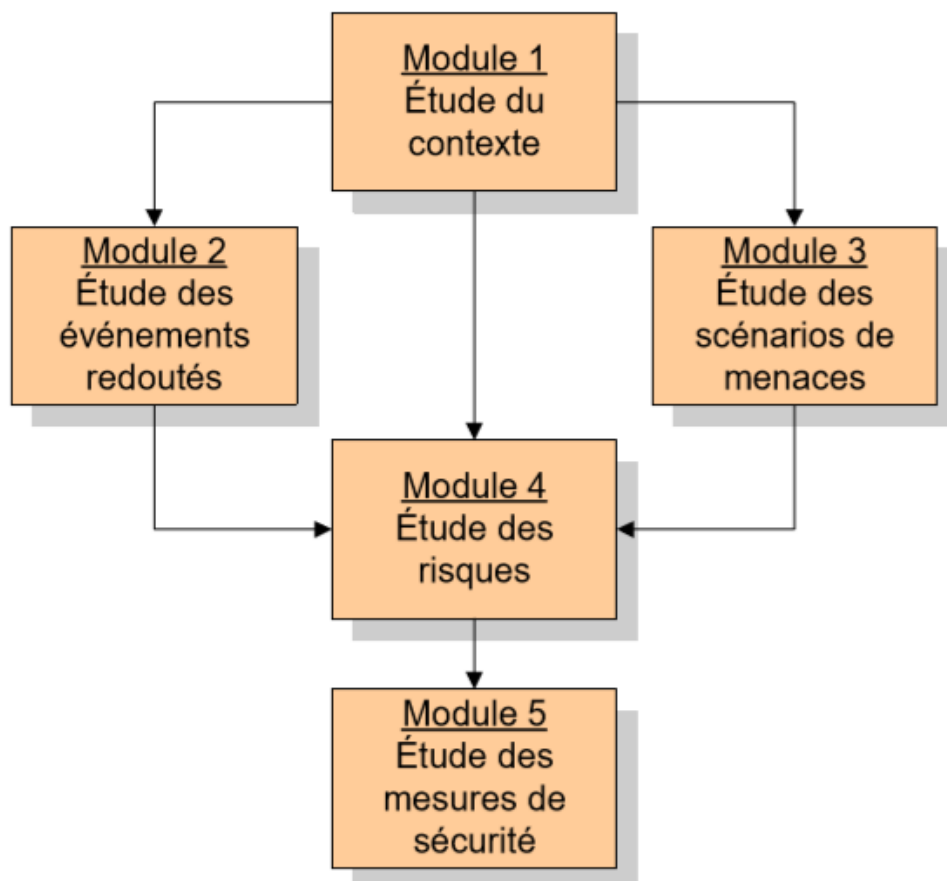


FIGURE 2.1 – Etape de EBIOS

2.1.1 Étude du contexte :

Cette étape consiste à établir le contexte, le périmètre de l'étude, les paramètres à prendre en compte.

2.1.1.1 Périmètre de l'étude

Dans le cadre du sujet d'étude, on a retenu les processus suivants :

- Gestion de l'interface
- Gestion des comptes clients
- Gestion des assurances délivrées
- Gestion des fournisseurs
- Gestion de la sécurité du système

2.1.1.2 Les métriques prises en compte

Pour les critères de sécurité, on a pris en compte les métriques suivantes : Echelle de Disponibilité, Echelle d'intégrité, Echelle de Confidentialité, Echelle d'Authenticité

Niveau de l'échelle	Description
Plus de 12h	La plateforme doit être disponible dans plus de 12 heures
Entre 4h et 12h	La plateforme doit être disponible dans les 12 heures
Entre 1h et 4h	La plateforme doit être disponible dans les 4 heures
Entre 15mn et 1h	La plateforme doit être disponible dans les 1 heures

TABLE 2.1 – Tableau de l'échelle de disponibilité

Niveau d'échelle	Description
Déectable	Les données sur la plateforme peuvent ne pas être intègres si l'altération est identifiée
Maîtrisé	Les données circulées sur la plateforme peuvent ne pas être intègres, si l'altération est identifiée et l'intégrité de la plateforme retrouvée
Intègre	Les données circulées sur la plateforme doivent être rigoureusement intègres

TABLE 2.2 – Tableau de l'échelle d'intégrité

Niveau d'échelle	Description
Public	Les données sont accessibles à tout le monde
Limité	Les données ne sont accessibles qu'aux entités autorisées
Privée	Les données ne sont accessibles qu'aux propriétaires

TABLE 2.3 – Tableau de l'échelle de Confidentialité

Niveau d'échelle	Description
Faible	Le concerné n'est pas identifié
Moyen	L'identité du concerné est altéré
Authentique	Le concerné est bien identifié

TABLE 2.4 – Tableau de l'échelle d'Authenticité

2.1.2 Étude des événements redoutés :

Cette étape permet d'identifier et d'estimer les besoins de sécurité du système ainsi que les impacts en cas de non-respect de ces besoins. Cette étape identifie aussi les sources de menaces susceptibles d'en être l'origine.

Événement redouté	Sources de menaces	Impacts
Indisponibilité de l'interface	<ul style="list-style-type: none"> • Panne électrique, • Bug • Attaque 	<ul style="list-style-type: none"> • Perte d'un client • Impossibilité d'accéder à la plateforme
Altération de l'interface	Attaque	<ul style="list-style-type: none"> • Perte d'un client • Impossibilité d'accéder à la plateforme
Indisponibilité des données	Bug	<ul style="list-style-type: none"> • Perte de notoriété • Impossibilité d'utiliser à la plateforme
Altération des données	<ul style="list-style-type: none"> • Attaque externe • Concurrent 	<ul style="list-style-type: none"> • Perte de Client • Perte de notoriété • Perte de crédibilité
Compromission des données	Attaque externe, Concurrent	<ul style="list-style-type: none"> • Perte de Client • Action en justice à l'encontre du startup

TABLE 2.5 – Tableau des événements redoutés

2.1.3 Étude des scénarios de menaces :

Il consiste à identifier et à estimer les scénarios qui peuvent engendrer les événements redoutés, et ainsi composer des risques.

2.1.4 Étude des risques :

Cette étape met en évidence les risques pesant sur l'organisme en confrontant les événements redoutés aux scénarios de menaces. Il décrit également comment estimer et évaluer ces risques, et enfin comment identifier les objectifs de sécurité qu'il faudra atteindre pour les traiter.

2.1.5 Étude des mesures de sécurité

Il explique comment spécifier les mesures de sécurité à mettre en œuvre, comment planifier la mise en œuvre de ces mesures

Les deux dernières étapes seront développées dans le prochain chapitre.

Chapitre 3

Conception de la base de données

Cette partie consiste à travailler sur la base de données d'escale pour le système. Il s'agit de la base de données qui récupèrera et qui va gérer toutes les informations entrantes et sortantes, utilisées par le système. La partie conception est une étape importante dans la réalisation du projet et dans le développement de l'application parce qu'elle facilitera les modifications et les évolutions ultérieures du projet en fonction des retours d'informations.

Nous devons alors prendre en compte une base de données qui peut s'évoluer au cours du temps pour s'adapter aux besoins du moment et besoins du futur. Est-ce qu'on saura répondre aux besoins, aux contraintes. Une organisation de données qui répond mieux aux contraintes de disponibilité de l'information face à l'augmentation exponentielle des données ainsi qu'à leur distribution à travers le système. La structure de données d'un utilisateur n'est pas fixée. En effet, si on considère un utilisateur choisissant en sa première utilisation de l'application une assurance automobile à un fournisseur assurance¹, l'application ne lui a demandé que les informations concernant son identité et sa voiture. Notons *informationI* les informations sur l'identité de l'utilisateur, *informationsA* les informations sur la voiture. Mais à un certain moment du futur, il peut vouloir une assurance santé avec un autre fournisseur assurance², de nouvelles informations sont donc nécessaires sur l'utilisateur en question, notons *informationsS*. Le système gèrera le fait que l'*informationI* concernant l'utilisateur sera accessible aux fournisseurs assurance¹ et assurance², mais l'*informationA* et *informationS* seront octroyées respectueusement à assurance¹ et à assurance².

3.1 Choix de l'outil

3.1.1 SGBD NoSQL

NoSQL signifie Not Only SQL.

La première fonction d'une base de données est de permettre de stocker et de retrouver l'information. À la naissance de l'informatique, plusieurs modèles de stockage d'information ont été explorés mais c'est finalement le modèle relationnel qui l'emporte dans les années 1970. Les bases de données NoSQL remettent en cause l'hégémonie des SGBDR telle qu'elle s'est constituée dans les années 1980. Notre objectif est de concevoir une base de données performante, optimisée et adaptée à nos besoins futurs. Ce logiciel est destiné aux traitements et aux stockages des informations concernant les clients, les assureurs, les assurances délivrées. Il est donc indispensable de consacrer le maximum d'analyse pour la base de données. Un grand choix s'est donc posé entre créer une base de données relationnelle ou une base de données non relationnelles. Au regard de la maturité et de la bonne réputation dont jouissent les bases de données classiques, on peut se poser la question de savoir ce qu'offrent les bases de données non relationnelles pour justifier le choix. Les SGBD relationnels ne sont pas adaptés aux environnements distribués ayant des volumes gigantesques de données et intégrant de nombreuses fonctionnalités importantes, adaptées aux besoins des assureurs, en particulier la gestion de l'intégrité des données et l'implémentation des échanges, indispensables pour l'application. Avec l'explosion de quantité de données à gérer, il faut alors effectuer beaucoup de jointures pour reconsolider l'ensemble de l'information, le SGBD relationnel n'est plus une bonne solution pour des accès globaux. En plus, les systèmes de jointures et les requêtes complexes nous causeront un grand problème de performance sur le volume de données que nous traiterons. Comment gérer une énorme base de données et comment l'interroger efficacement ? Le NoSQL s'est naturellement imposé dans ce contexte en proposant une nouvelle façon de gérer les données. Les plus grandes qualités d'un SGBD NoSQL sont d'être décentralisé, évolutif et tolérant aux pannes. Les bases de données NoSQL permettent de personnaliser la solution de gestion des données pour chaque cas d'utilisation. La plupart des produits NoSQL sont plus légers et induisent moins de gestion qu'une solution relationnelle. Un SGBD NoSQL demande moins de code, ce qui lui donne potentiellement un avantage en termes de performances par rapport à un SGBD plus complexe.

Les bases NoSQL présentent l'avantage d'être plus agiles, plus durables, et d'offrir une maintenance plus intelligente, ce qui offre lorsqu'elles sont maîtrisées une intéressante robustesse [11].

Il existe différentes familles de bases NOSQL :

- Clé/Valeur :

Les données sont représentées par un couple clé-valeur, la valeur pouvant être une simple chaîne de caractères ou un objet. Ce modèle offre une forte évolutivité grâce à l'absence de structure ou de typage.

- Orientée Colonne :

Ce modèle se rapproche de la table de la base de données relationnelle, il dispose de la capacité de créer des colonnes dynamiquement.

- Orientée documents : Cette base se repose sur le paradigme clé-valeur, elle remplace la valeur par un document de type JSON ou XML. Une seule clé permet ainsi de récupérer l'ensemble des informations de manière hiérarchique.

Chacune de ces familles répond à des besoins très spécifiques mais ce qui a correspondu le plus à notre problème c'est la base orientée document. Le but de ce stockage est de manipuler des documents contenant des informations avec une structure complexe (types, listes, imbrications). L'avantage de cette solution est d'avoir une approche structurée de chaque valeur, formant ainsi un document.

3.1.2 MONGODB

MongoDB est un système de gestion de base de données de type NoSQL orienté documents. MongoDB est développé depuis 2007 par MongoDB [9]. Les données prennent la forme de documents enregistrés eux-mêmes dans des collections, une collection contenant un nombre quelconque de documents. Les collections sont comparables aux tables, et les documents aux enregistrements des bases de données relationnelles. Les champs d'un enregistrement sont libres et peuvent être différents d'un enregistrement à un autre au sein d'une même collection. Le seul champ obligatoire et commun est le champ de clé principale Id [2].



FIGURE 3.1 – Logo de MongoDB

3.2 MODELISATION

Pour stocker les données dans la base de données, il a fallu créer des collections pour accueillir ces données. Les collections sont comparables aux tables, et les documents aux enregistrements des bases de données relationnelles. Une collection est un système conçu pour le stockage d'un ensemble de données organisées sous forme de XML. Les données sont donc

représentées par des champs et des valeurs associées. L'avantage est de pouvoir récupérer, via une seule clé, un ensemble d'informations structurées de manière hiérarchique.

3.2.1 Collection User

Les informations sur un utilisateur sont stockées dans la collection User. Cette dernière contient les informations personnelles sur l'utilisateur et les informations sur l'assurance qu'il a choisie.

3.2.2 Document InfoUser

Les informations personnelles de l'utilisateur sont stockées dans le document InfoUser. Ce sont les informations qui vont être rarement changées ou mises à jour.

3.2.3 Document Type d'Assurance

Les informations sur l'assurance que l'utilisateur a choisie sont stockées dans un document différent pour chaque type d'assurance qu'il choisira. Un nouveau document est créé à chaque fois que l'utilisateur renouvelle son assurance ou choisit de s'octroyer une nouvelle assurance.

3.2.4 Sous-document Fournisseur d'assurance

Ce sous-document contient les informations particulières exigées par le fournisseur sur l'assurance que le client a choisie.

On reconnaît NoSQL par sa capacité de scalabilité horizontale. La structure de la base n'est pas alors fixée mais varie selon le choix d'assurance de l'utilisateur. La base sera homogène à toute information ultérieure. Pour gérer la mise à jour des informations, seules les informations concernant l'objet à assurer seront redondantes.

Lorsqu'on a un nouveau client sur la plateforme soit c'est pour sa première inscription pour une assurance soit c'est pour opter pour la solution en ligne. Dans le premier cas, le client fournit les informations pour être stockées dans le document InfoUser, puis en fonction de l'assurance qu'il a choisie, les autres informations seront demandées. Les données seront ensuite envoyées vers le système d'information de l'assurance choisie à travers notre plateforme.

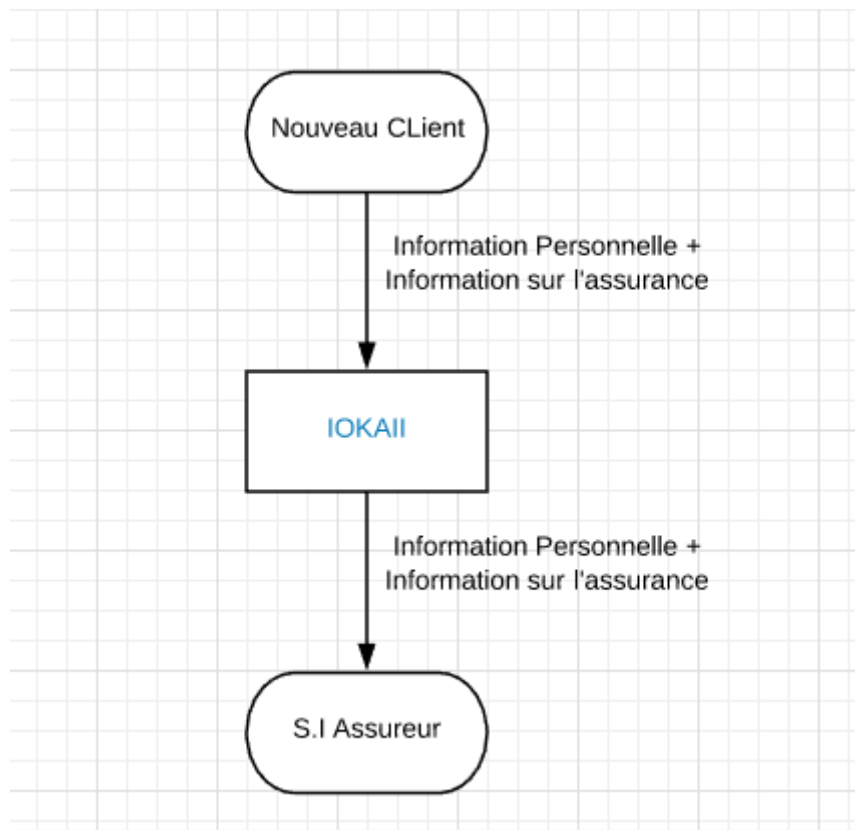


FIGURE 3.2 – Démarche d'un nouveau client

Dans le second cas, si le client possède déjà une assurance mais s'inscrit pour la première fois sur la plateforme, selon le numéro de police de son assurance la plateforme effectue une recherche du fournisseur correspondant, ensuite elle envoie une requête demandant les informations nécessaires au système d'information de fournisseur. Ce dernier répondra en envoyant les informations. Donc le client ne va plus remplir à la main les informations qui sont complétées.

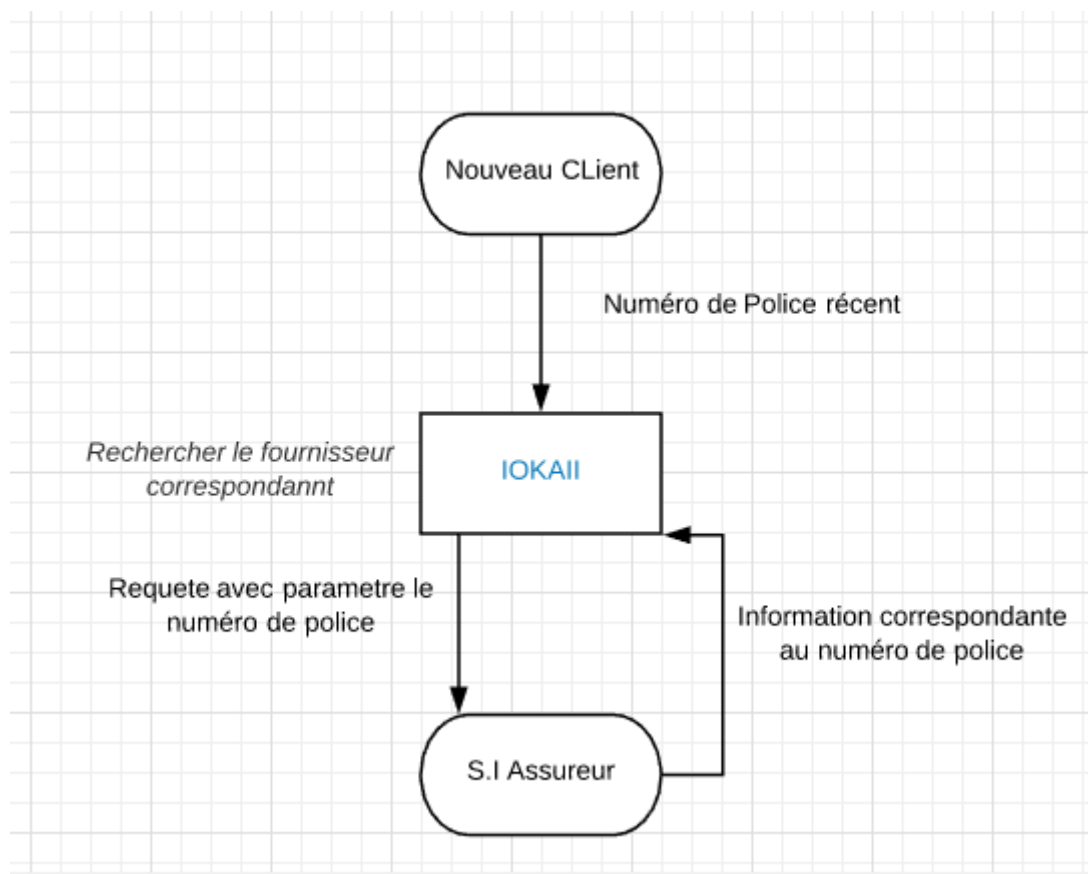


FIGURE 3.3 – Démarche d'un nouveau client de la plateforme

Donc, on a obtenu la conception suivante :

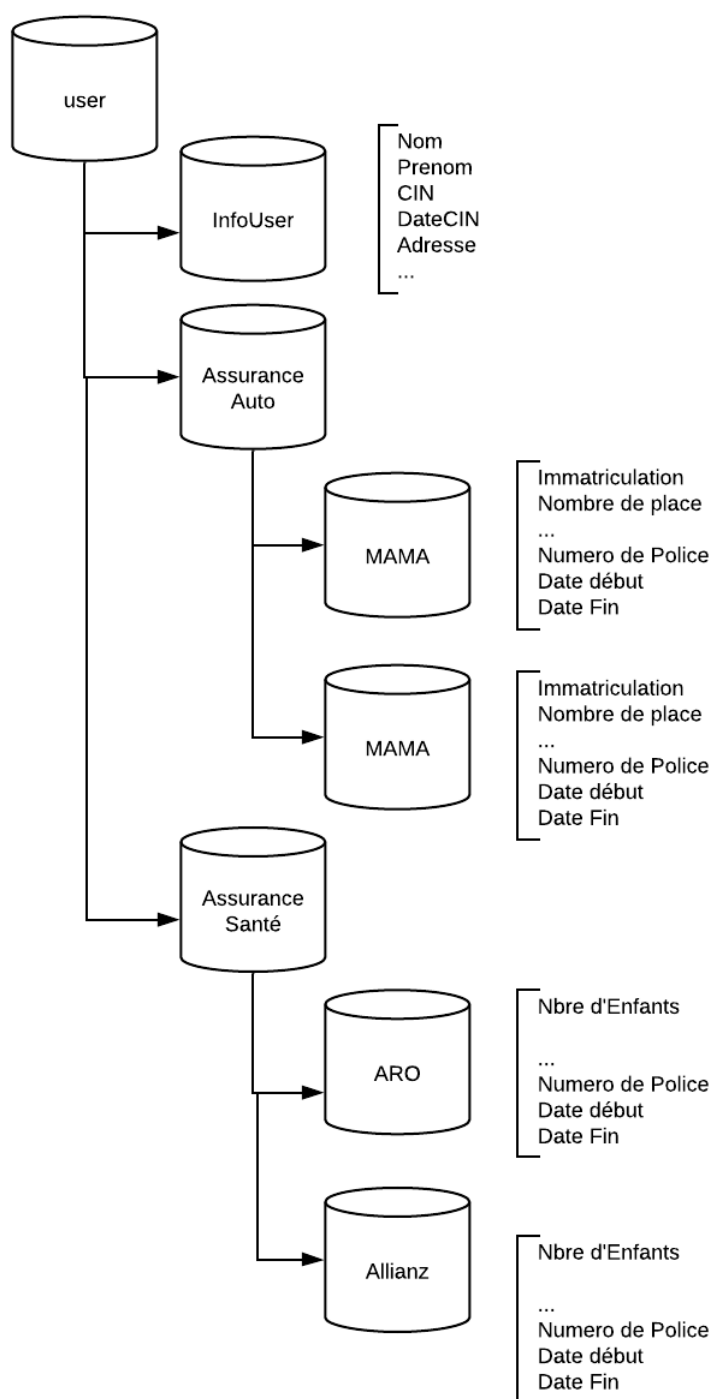


FIGURE 3.4 – Un type d'architecture de la base de données d'un utilisateur

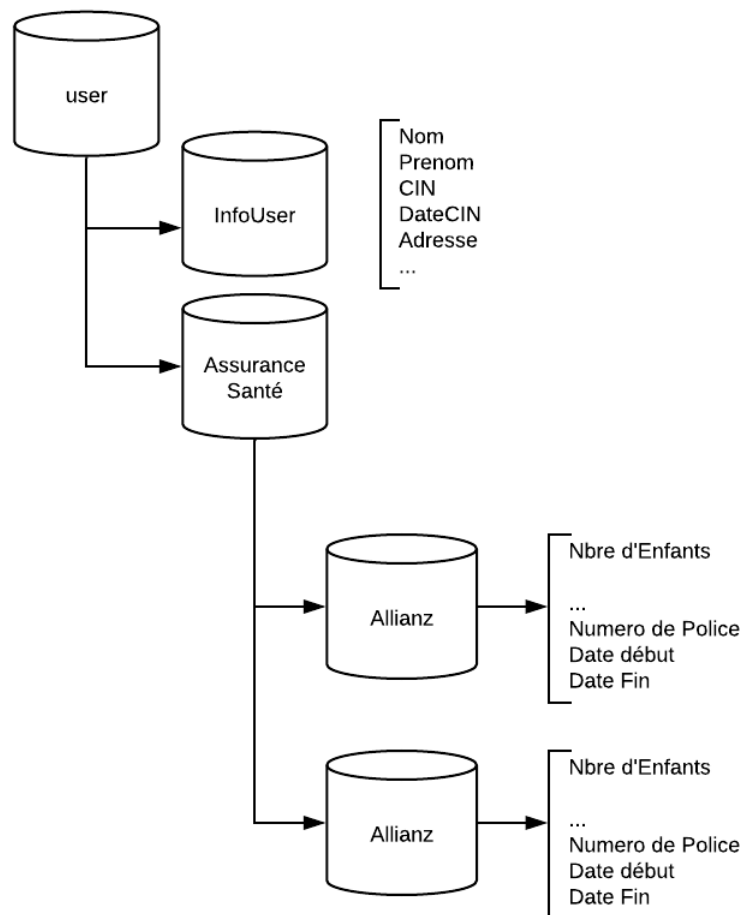


FIGURE 3.5 – Un autre type d'architecture de la base de données d'un autre utilisateur

Un extrait de la base de données peut être consulté dans l'annexe .

Grâce à cette conception, il nous est facile d'établir les diverses requêtes suivantes sachant que c'est à notre plateforme de gérer les réponses :

- Fournir l'identifiant et le mot de passe du client pour la connexion à l'espace client
- Fournir la liste de toutes les assurances d'un utilisateur
- Fournir la liste des clients d'un assureur
- Fournir la liste des clients pour un type d'assurance. (Exemple : Liste des clients d'une assurance santé)
- Fournir le bilan des assurances depuis l'activation du compte

Chapitre 4

Réalisation de l'Interface

4.1 Outil et Technologie

Pour l'élaboration de ce projet, nombreux outils ont été utilisés. Ces outils et ces technologies ont leur domaine d'usage, leurs performances et leurs faiblesses. Le choix de ces outils et de ces technologies dépend de la spécificité de la plateforme elle-même. Pour ce projet, on a opté sur les technologies WEB à cause des infrastructures déjà mises en place. Le WEB ou World Wide Web est un service d'Internet par lequel les pages sur le réseau Internet peuvent être visitées. La majorité des technologies WEB est gratuite et libre. Les outils et technologies qu'on va présenter ci-dessous sont les technologies de base qu'on a utilisées.

4.1.1 HTML ou HyperText Markup Language

HTML est un langage dit de « marquage » conçu pour représenter les pages d'un site web, dont le rôle est la formalisation de l'écriture d'un document avec des balises de formatage. Les balises permettent d'indiquer la façon de présenter la page. HTML est un Système de gestion de formations Outils et technologies actuellement à sa cinquième version appelée HTML5. Les pages créées avec HTML sont des pages statiques, c'est-à-dire que les pages ont besoin d'une aide ultérieure pour pouvoir changer leurs contenus. Ce langage est exécuté du côté client.

4.1.2 CSS ou Cascading Style Sheets

CSS est un langage qui décrit la présentation ou le style d'une page web. Il met à zéro tous les styles par défaut d'une balise HTML. le logiciel est actuellement à sa troisième version le CSS3.

4.1.3 JavaScript

JavaScript est un langage de programmation interprété avec des fonctionnalités orientées objet. Syntaxiquement, le langage JavaScript de base ressemble à C, C ++ et Java, mais la similitude se termine par cette ressemblance syntaxique. Javascript est un langage peu typé, ce qui signifie que les variables n'ont pas besoin d'avoir un type spécifié. JavaScript est principalement employé dans les pages web interactives mais aussi pour les serveurs. JavaScript possède plusieurs bibliothèques de fonctions, la plus connue est JQuery. JQuery lui-même est doté de plusieurs plugins pour l'affichage des données dans un tableau flexible et interactif.

4.1.4 PHP ou Hypertext Preprocessor

Ce langage est un langage informatique exécuté côté serveur, réputé dans le développement web, mais permet de faire autre chose. Le PHP permet de créer des pages web dynamiques. Les fonctionnalités de la plateforme ont été développées dans la version 5 (PHP5) de PHP [3].

4.1.5 CodeIgniter 3

En programmation informatique, un Framework est un ensemble de composants logiciels qui permettent de créer le squelette d'un logiciel ou d'une application [5]. Ici, le Framework CodeIgniter3, gratuit et libre, est un Framework PHP. Par rapport aux autres Framework PHP, CodeIgniter3 est légère et très modulable. Son avantage est que ce Framework utilise la structure MVC qui est un patron d'architecture logiciel destiné à répondre aux besoins des applications interactives en séparant les problématiques liées aux différents composants au sein de leur architecture respective. Ce modèle permet de gérer facilement la qualité du logiciel, mais il n'est pas intéressant de l'utiliser pour un petit projet à cause de sa configuration.

4.1.6 PhoneGap

PhoneGap est un framework open source pour créer des applications mobiles multi-plateforme avec les technologies traditionnelles du web : HTML, CSS, JavaScript. Il permet d'avoir une application hybride [13].

4.2 Réalisation

L'interface client qui réalise la Phase 1 définie dans la partie 1.5.1 jusqu'à la phase 4 est faite d'une "one page" permettant un échange fluide et sécurisé. Un site avec une seule page est avant tout ergonomique et très épuré, la simplification du design aide à comprendre facilement et rapidement le ou les messages à transmettre à l'utilisateur. Le one-page s'accorde très bien pour les supports mobiles (tablette et smartphones). Cette solution est idéale, pour rendre la page vivante. Elle est développée avec le framework CodeIgniter3 et nous avons conçu un système qui nous permet d'avoir à la fois un site internet et une application mobile (Android et IOS) après compilation sur PhoneGap.

L'image ci-dessous présente l'interface du site internet.



FIGURE 4.1 – Interface du site internet

L'image ci-dessous présente l'interface de la plateforme en application mobile.



iokaii

Information sur le véhicule

ETAPE 3

Création

Début du contrat

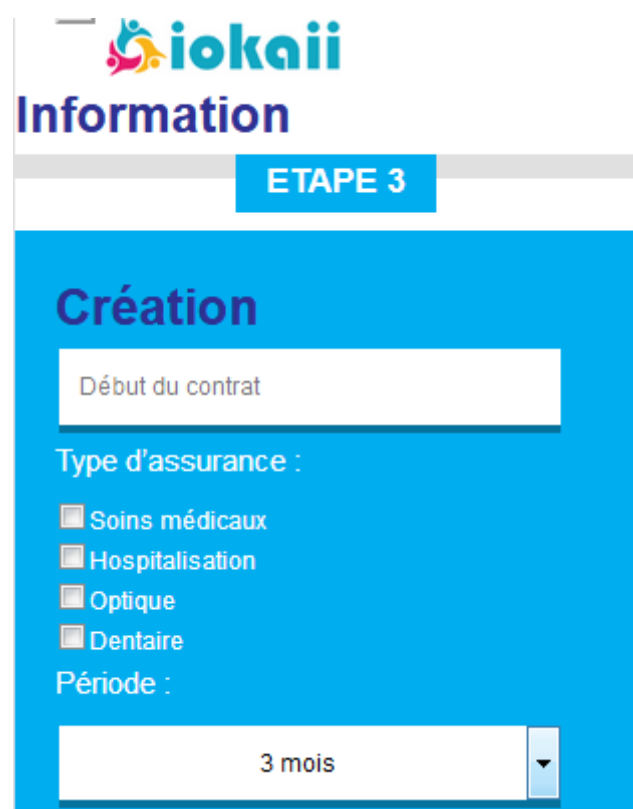
Immatriculation

Carte Grise

N° de Série

Année deCirculation

FIGURE 4.2 – Interface de l'application mobile



iokaii

Information

ETAPE 3

Création

Début du contrat

Type d'assurance :

☐ Soins médicaux

☐ Hospitalisation

☐ Optique

☐ Dentaire

Période :

3 mois

FIGURE 4.3 – Interface de l'application mobile

Chapitre 5

Echange de données

Maintenant qu'on a terminé la partie base de données c'est à dire, là où les données vont être stockées, nous allons maintenant passer à comment les données vont être échangées. Cette partie se divise en trois étapes, à savoir, la transmission, l'acheminement et en dernier la sécurisation.

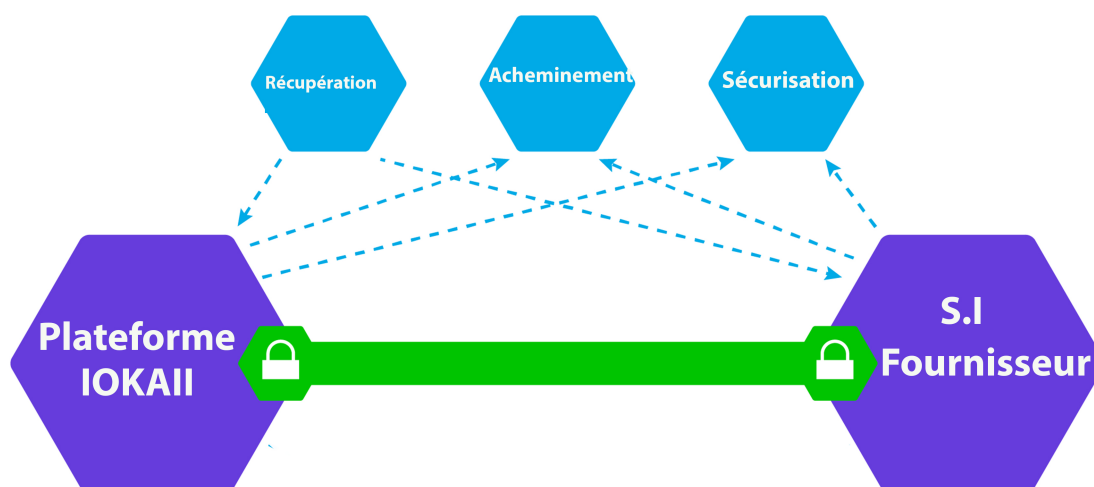


FIGURE 5.1 – But de l'échange

5.1 Récupération des données

5.1.1 Service Web

Selon W3C, Un service Web est un composant logiciel identifié par une URI, dont les interfaces publiques sont définies et appelées en XML. Sa définition peut être découverte par d'autres systèmes logiciels. Les services Web peuvent interagir entre eux d'une manière prescrite par leurs définitions, en utilisant des messages XML portés par les protocoles Internet. [12] Les services Web sont normalisés car ils utilisent les standards XML et HTTP pour transférer des données et ils sont compatibles avec de nombreux autres environnements de développement. Ils ont été conçus pour être réalisés sur de nombreux systèmes développés et déployés de façon indépendante [7].

5.1.2 XML

XML ou eXtensible Markup Language est un langage informatique de balisage générique. Un langage de balisage est un langage qui s'écrit grâce à des balises. Ces balises permettent de structurer de manière hiérarchisée et organisée les données d'un document. En d'autre terme, le langage XML est un langage qui permet de décrire des données à l'aide de balises et de règles que l'on peut personnaliser. XML a été largement adopté en informatique dès sa normalisation en 1998 après le langage SGML ou Standard Generalized Markup Language [8]. L'objectif initial de XML est de faciliter l'échange automatisé de contenus complexes (arbres, texte enrichi, etc.) entre systèmes d'informations hétérogènes (interopérabilité) , et aussi de décrire les données de manière aussi bien compréhensible par les hommes qui écrivent les documents XML que par les machines qui les exploitent.

5.1.3 SOAP

SOAP (Simple Object Access Protocol) est un standard développé par le W3C (World Wide Web Consortium). Ce protocole repose entièrement sur le langage de description XML. C'est un protocole pour échanger des informations structurées dans un environnement décentralisé et distribué. Il a pour principal objectif d'assurer la communication entre machines. [8] Un message SOAP est composé de deux parties obligatoires :

- l'enveloppe SOAP qui contient la spécification des espaces de nommages (namespace) et du codage de données. C'est l'élément supérieur du document.
- l'entête qui est placé au sein de l'enveloppe avant même le corps, l'entête peut-être utilisé pour compléter les informations nécessaires à une requête.

- le corps SOAP qui contient les méthodes et les paramètres qui seront exécutés par le destinataire final ; il contient l'information destinée au receveur c'est à dire l'information à échanger. Cet élément peut avoir plusieurs fils et, comme pour l'en-tête, il faut seulement que chaque fils soit un élément XML bien formé, valide, qualifié avec un espace de nommage. [4] En réponse, il contiendra soit un appel méthode, soit une réponse à sens unique, ou finalement un message d'erreur détaillée.

Notre but est de développer un système accessible à d'autres applications par le biais de messages SOAP transportés sur le simple protocole HTTP. SOAP est placé au dessus des couches de transport. Cette particularité rend SOAP extrêmement flexible et permet l'utilisation en n'importe quelle situation ou structure de réseau. [10]

5.1.4 RPC

RPC ou Remote Procedure Call construit un message SOAP contenant le nom de la méthode qu'on veut utiliser avec une liste de paramètres. Quand le message arrive au serveur, la requête est transformée en méthode et une réponse contenant la sortie est générée

5.1.5 HTTP

HTTP ou Hypertext Transfer Protocol, désigne dans le langage informatique un protocole de communication entre un client et un serveur pour le World Wide Web. Le but du protocole HTTP est de permettre un transfert de fichiers entre un client et un serveur.

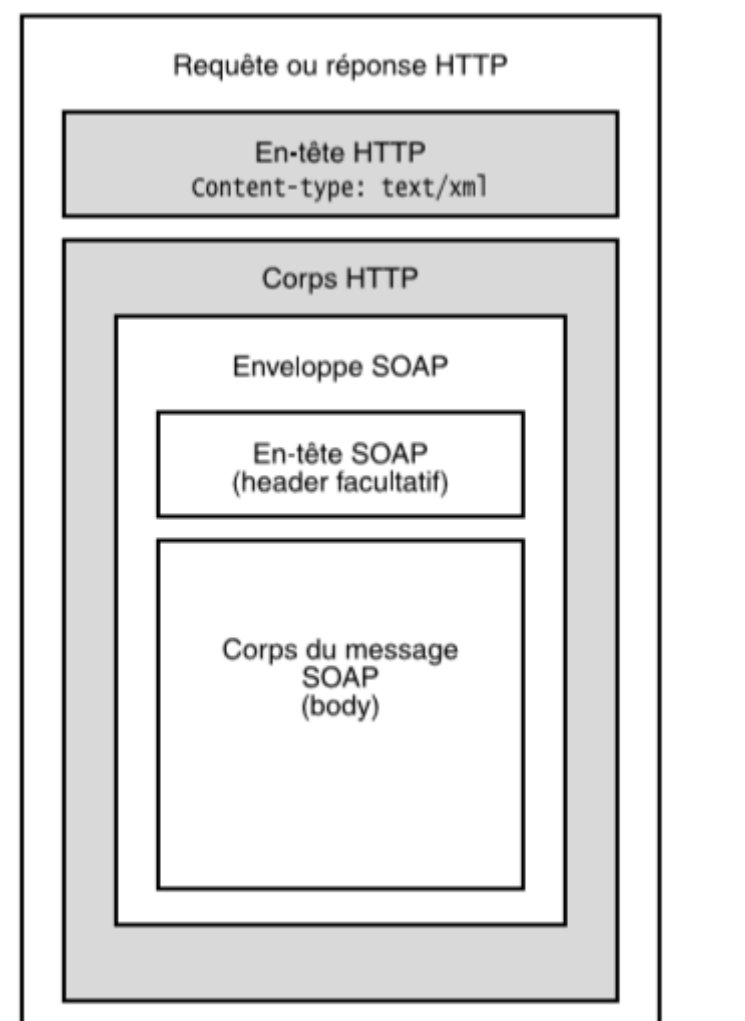


FIGURE 5.2 – Message SOAP inclus dans une requête ou une réponse HTTP

5.1.6 HTTP POST

L'utilisation de la liaison sur HTTP avec l'échange de messages SOAP de type Requête-Réponse est restreinte à la méthode HTTP POST.

SOAP et HTTP travaillent naturellement ensemble parce que HTTP est un protocole basé sur le système requête/réponse et reflète parfaitement le concept du modèle SOAP RPC

5.2 Acheminement des données

Dans cette partie, nous allons développer la façon dont les données seront acheminées dans notre base de données. Notre but est d'avoir une base de données transparente et orchestrée. Chaque nouvelle information est acheminée automatiquement avec l'architecture de l'application de façon à s'accorder avec la structure de la base et à respecter les droits de propriétés.

5.2.1 Gestion et Contrôle des données

La conception de la base de données concerne aussi la gestion des accès et propriétaires des informations, la gestion des circulations et passages d'information. Le but est de mener à bien les opérations nécessaires à l'activité en conformité avec les conventions sur la protection des données afin de respecter les consentements des propriétaires des données. Le propriétaire de données est en droit de retirer à tout moment son accord quant au traitement de ses informations. La plateforme gère les données et contrôle comment et par qui elles peuvent être utilisées et consultées. Les données doivent être techniquement faciles à utiliser et accessibles, et une infrastructure partagée en place doit permettre le transfert et la gestion des données.

5.2.1.1 Pour un utilisateur simple

Les utilisateurs disposent des moyens pratiques et appropriés de gérer leurs données personnelles. Les ensembles de données devraient être interopérables et portables entre les systèmes d'information et les secteurs. La plateforme gère les autorisations créées par chaque action. Un consentement donne à un service utilisant des données une autorisation d'accéder aux données personnelles d'un individu fourni par une source de données. En d'autres termes, une personne utilise le compte pour accorder à des services l'accès à ses données personnelles. Grâce au compte, une personne devrait également être en mesure de consulter les journaux d'accès concernant ses données personnelles - quelles services utilisant les données ont accédé à quelle information, à quel moment. Le but c'est d'avoir une plateforme hétérogène pour qu'une personne utilisant la plateforme aie une identité unique indépendante du nombre d'assureur qu'il a.

5.2.1.2 Pour les systèmes d'informations

L'idée principale de la plateforme est que chaque fournisseur de données est responsable de la gestion des droits d'accès de ses services. En d'autres termes, l'envoi de données sur la plateforme ne signifie pas que celles-ci sont automatiquement accessibles à toutes les organisations membres de la plateforme. Habituellement, les droits d'accès sont accordés au niveau du système d'information : un fournisseur de services accorde à un système d'information spécifique l'accès aux informations. Chaque information est donc libellée de son droit d'accès.

5.3 Sécurité des transmissions

Notre but dans cette partie est de :

- Vérifier l'intégrité des données signées
- Identifier et garantir l'identité du signataire
- Assurer la non-répudiation

Pour assurer la sécurité des transmissions, nous avons d'abord appliqué la signature électronique.

La signature électronique repose sur l'utilisation de deux algorithmes qui sont l'algorithme de hachage et l'algorithme de chiffrement.

5.3.1 Algorithme de hachage

Les fonctions de hachage sont des fonctions de calcul de l'empreinte (condensat - hashcode) d'une donnée de telle sorte que cette empreinte signe de manière unique cette donnée sans en révéler la donnée elle-même. Voici des exemples de fonctions de hachages les plus connus traduites en algorithmes de hachage :

- SHA :SHA-0, SHA-1 , SHA-2, SHA-3
SHA (Secure Hash Algorithm) désigne une fonction de hachage conçue par la National Security Agency des États-Unis (NSA)
- MD5 :
MD5 (Message Digest 5) est une fonction de hachage cryptographique (un algorithme) qui permet d'obtenir l'empreinte numérique (hashcode - condensat) d'un fichier. L'algorithme MD5 a été mis au point par Ronald Rivest en 1991 (il améliore l'algorithme MD4).

5.3.2 Algorithme de chiffrement

L'algorithme de chiffrement permet de protéger des données en les rendant illisibles si on ne possède pas la clé pour les déchiffrer. On distingue deux types d'algorithme de chiffrement :

5.3.2.1 L'algorithme de cryptographie symétrique

Il se fonde sur une même clé pour chiffrer et déchiffrer un message. L'un des problèmes de cette technique est que la clé, qui doit rester totalement confidentielle, doit être transmise au correspondant de façon sûre Exemples :

- Chiffre de César : une méthode de chiffrement très simple utilisée par Jules César dans ses correspondances secrètes. Le texte chiffré s'obtient en remplaçant chaque lettre du texte clair original par une lettre à distance fixe, toujours du même côté, dans l'ordre de l'alphabet
- DES Data Encryption Standard, Son emploi n'est plus recommandé aujourd'hui, du fait de sa lenteur à l'exécution et de son espace de clés trop petit permettant une attaque systématique en un temps raisonnable
- AES Advanced Encryption Standard, Il est actuellement le plus utilisé et le plus sûr.

5.3.2.2 Algorithme de cryptographie asymétrique

Il se base sur le principe de deux clés : une publique permettant le chiffrement, une privée permettant le déchiffrement. Exemples :

5.3.2.2.1 RSA Rivest Shamir Adleman créé par Ronald Rivest, Adi Shamir et Leonard Adleman en 1977. Le système RSA est basé sur les fonctions à sens unique. C'est à dire qu'il est simple d'appliquer la fonction, mais extrêmement difficile de retrouver l'antécédente de la fonction à partir de son image seulement. Le but est de pouvoir transmettre un message codé que seul le récepteur "officiel" puisse décrypter. Nous appellerons Alice la destinatrice du message, et Bernard l'émetteur

1. Alice génère deux gros nombres premiers p et q , ainsi qu'un gros nombre d premier avec le produit $w = (p - 1)(q - 1)$.
2. Alice calcule $n = pq$ et e tel que $de \equiv 1[w]$.
3. Alice diffuse n et e , garde d et oublie w .
4. Bernard crypte un message M par $M \mapsto M^e[n]$ et envoie le résultat à Alice.
5. Alice décode alors le message crypté par $C \mapsto C^d[n]$

FIGURE 5.3 – Chiffrement RSA

5.3.2.2.2 Echange de clés Diffie-Hellman C'est une méthode publiée en 1976 par laquelle deux agents, nommés par convention Alice et Bob, peuvent se mettre d'accord sur un nombre (qu'ils peuvent utiliser comme clé pour chiffrer la conversation suivante) sans

qu'un troisième agent appelé Ève puisse découvrir le nombre, même en ayant écouté tous leurs échanges.

5.3.2.3 Les algorithmes hybrides

Le principe c'est d'utiliser un algorithme symétrique et un algorithme asymétrique dans le même processus (d'où le terme hybride) en ne prenant que le meilleur de chacun. L'idée générale c'est de générer une clé pour l'algorithme symétrique. On chiffre le message avec cette clé (avantage = rapidité). Ensuite on chiffre la clé avec un algorithme asymétrique (avantage = transmission sécurisée de la clé par le biais d'un système clé publique/clé privée). Côté destinataire il suffit de déchiffrer la clé avec la clé publique pour déchiffrer ensuite le message.

5.3.3 XML Signature

XML Signature (XMLDsig, XML-DSig, XML-Sig) est un standard publié par W3C qui permet de créer une signature pour les documents XML. La signature XML est une méthode qui consiste à associer une clé avec des données référencées. XML Signature permet de signer digitalement des portions d'un document XML, mais également de n'importe quel format de données. Mais une importante caractéristique de XML Signature dans les web services est sa capacité à pouvoir signer uniquement des données sélectionnées. Traitement :

- L'émetteur crée un message, le canonise et le signe.
- Le récepteur reçoit un message, le canonise et vérifie sa signature.

Les quatre éléments principaux de XMLDsig sont :

- L'élément SignedInfo contient l'algorithme de canonicalisation qui sera appliqué à l'élément SignedInfo avant le calcul de la signature, l'algorithme de chiffrement utilisé pour le calcul de la signature ainsi que les données à signer (éléments Reference).
- L'élément Reference contient l'algorithme de hachage et l'empreinte calculée avec cet algorithme. L'URI référence l'identifiant des données à signer. L'URI et la transformation (élément Transform) indiquent comment les données à hacher ont été obtenues.
- L'élément SignatureValue contient la valeur de la signature.
- L'élément KeyInfo contient les informations sur le certificat utilisé pour signer, ce qui permet de déchiffrer la signature et de la vérifier.

5.3.4 Algorithme de canocalisation

C'est un algorithme qui permet de convertir les données en un format standard en éliminant les superflues ou pour permettre d'ordonner des éléments en fonction de leur sens [1].

5.3.5 XML Encryption

XML Encryption est une spécification du W3C. Il apporte un moyen de crypter des parties d'un document XML, mais également n'importe quelles données, puis rend les données cryptées au format XML.

XML Encryption est utilisé pour assurer que c'est le vrai destinataire qui est capable de lire et comprendre le document et XML Signature est utilisé pour assurer que le document est bien reçu sans modification [7].

5.3.6 WS Security

WS-Security ou Web Service Security (WSS) est un framework qui permet d'appliquer de la sécurité aux web services. C'est une extension flexible et plus riche appliquée à SOAP pour plus de sécurité. WSS assure l'authentification, l'autorisation, l'intégrité, la non-répudiation, la confidentialité des échanges. L'utilisation de WS-Security nous permet de protéger les données échangées par l'intermédiaire de XML Signature, XML Encryption, Timestamp et Security Token.

5.3.7 Authentification

L'authentification s'appuie sur la notion de token c'est à dire des headers SOAP capables de véhiculer l'identification (login et/ou mot de passe) jusqu'au service cible. Les principaux tokens disponibles sont :

- User Name Token : échange de couples login/mot de passe
- SAML Token : échange de données d'authentification au format XML SAML
- X509 Token : échange de données d'authentification par certificat (authentification forte)

Dans ce projet, c'est X509 Token que nous avons appliqué.

5.3.7.0.1 X509 X.509 est le standard le plus utilisé pour la création de certificat ; c'est une norme spécifiant les formats pour les certificats à clé publique, les listes de révocation de certificat, les attributs de certificat, et un algorithme de validation du chemin de certification, définie par l'Union internationale des télécommunications.

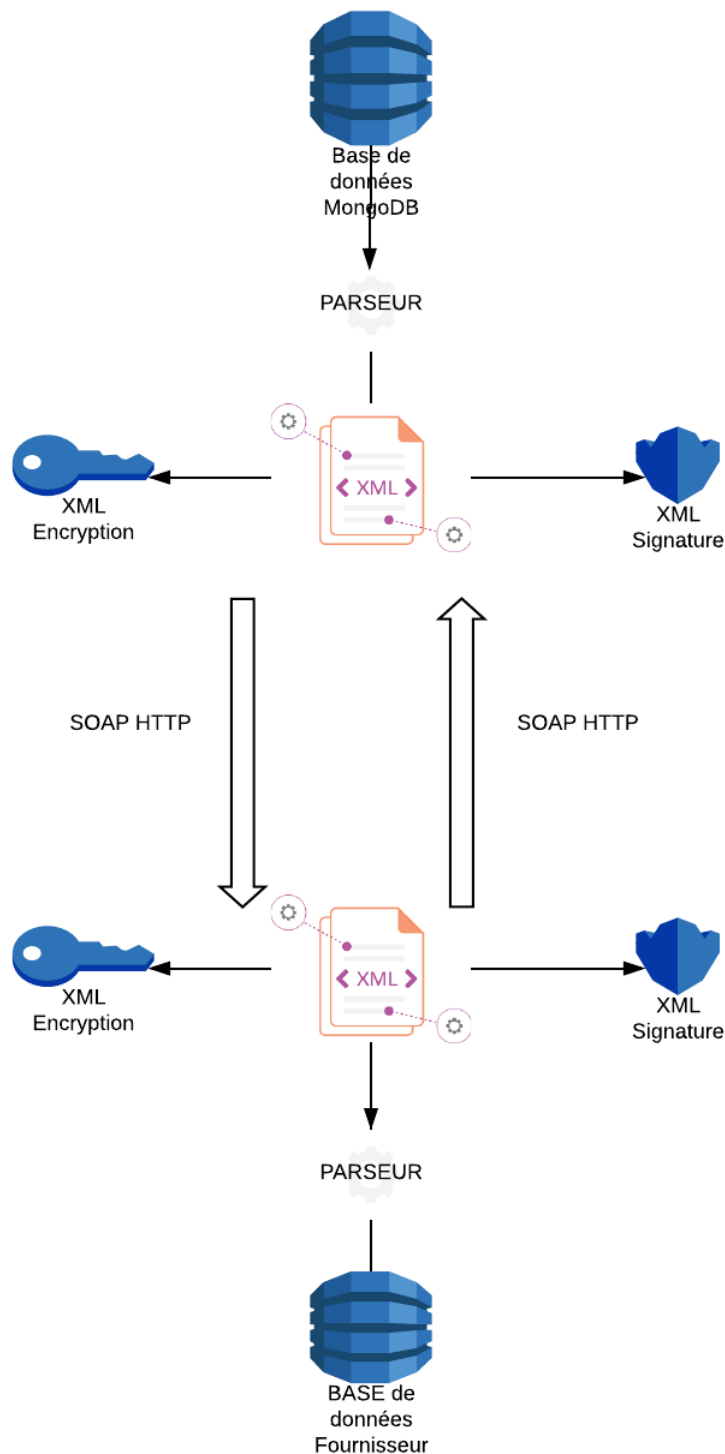


FIGURE 5.4 – Processus de l'échange

ANNEXES

Extrait de la base de données

```
{
  "infoUser": [
    {
      "nom": "RAZAFIMANANTSOA",
      "prenom": "Nasoavina Ch",
      "date_de_naissance": "1950-02-22",
      "CIN": "101889856896",
      "date_CIN": "2000-03-22",
      "adresse": "AN Ambohitsoa"
    }
  ],
  "assuranceAuto": [
    {
      "MAMA": [
        {
          "Immatriculation": "1111TBL",
          "Nombre_de_place": "5",
          "Type_energie": "Diesel",
          "Numero_de_police": "",
          "Date_debut": "2013-05-10",
          "Date_fin": "2013-11-10"
        }
      ]
    },
    {
      "MAMA": [
        {
          "Immatriculation": "1111TBL",
          "Nombre_de_place": "5",
          "Type_energie": "Diesel",
          "Numero_de_police": "",
          "Date_debut": "2013-11-10",
          "Date_fin": "2014-04-10"
        }
      ]
    }
  ],
  "assuranceMaladie": [
    {
      "ARO": [
        {
          "Nombre_d_enfant": "3",
          "Profession": "Directrice",
          "Numero_de_police": "",
          "Date_debut": "2013-11-10",
          "Date_fin": "2014-04-10"
        }
      ]
    }
  ]
}
```

FIGURE 5.5 – Base de données MongoDB de la plateforme

Squelette d'un message SOAP

```
<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  soap:encodingStyle="http://schemas.xmlsoap.org/soap/encoding">
  <soap:Header>
    <!-- en-tête -->
  </soap:Header>
  <soap:Body>
    <!-- corps -->
  </soap:Body>
</soap:Envelope>
```

FIGURE 5.6 – Enveloppe SOAP

```
<soap:Body>
  <demanderInformation>
    <identifiant xsi:type="xsd:int">1234567890</identifiant>
  </demanderInformation>
</soap:Body>
```

FIGURE 5.7 – Corps SOAP

Ci-dessous, la requête qu'on va envoyer via le protocole HTTP. Il s'agit de la méthode demanderInformation et qui a pour paramètre l'identifiant à demander.

```
POST /information HTTP/1.1
Host: 189.123.255.239 Content-Type: text/xml; charset=utf-8
Content-Length: length
SOAPAction: "www.fournisseurA.net/demandeInfo"
<?xml version="1.0" encoding="utf-8"?>

<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">

  <soap:Body>
    <demanderInformation>
      <identifiant xsi:type="xsd:int">101221896806</identifiant>
    </demanderInformation>
  </soap:Body>
</soap:Envelope>
```

FIGURE 5.8 – Requête SOAP

Ci-dessous, nous présentons un extrait du code de l'interface utilisateur.

```

        "email":email,
        "tel":tel,
        "type_assurance":type_assurance,
        "type_tarif":type_tarif,
        "periode":periode,
        "source":source,
        "puissance_fiscale":puissance_fiscale,
    }
    rpage_4.innerHTML="Email envoyé";
    /*
    xhr(data,"http://sopromer.com/assurance/mail.php",function(r){
        setTimeout(function(){
            rpage_4.innerHTML="Email bien envoyé";
        },100);
        console.log(r);
    });
    */

    xhr(data,"<?php echo base_url(); ?>/app/mail",function(r){
        setTimeout(function(){
            rpage_4.innerHTML="Information reçue";
        },100);
        console.log(r);
    });
}

function isFunction(data,object){
    if(object){
        return (typeof object[data]=="function");
    }
    return (typeof data=="function");
}

function xhr(data,to,handle,custom,customs,error){
    var rdata={};
    if(data && typeof data.getAttribute=="function"){
        Array.prototype.forEach.call(data.elements,function(elt){
            if(elt && elt.getAttribute("name")){
                rdata[elt.getAttribute("name")]=elt.value;
            }
        });
    }
    else{
        rdata=data;
    }
    tpr_sendJsForm(rdata,to,handle,custom,customs,error);
}

function tpr_sendJsForm(data,to,handle,custom,customs,error){
};

```

FIGURE 5.9 – Extrait du code de l'interface utilisateur

Nous définissons ci-dessous la structure du message SOAP avec les paramètres de sécurités.

```
<soap:Envelope ...>
  <soap:Header>
    <wsse:Security>
      <wsse:BinarySecurityToken wsu:Id="myKey" ...>
... security token ...
      </wsse:BinarySecurityToken>
      <sig:Signature>
        <sig:SignedInfo>
          <sig:Reference URI="#myMsg">...digest...</sig:Reference>
        </sig:SignedInfo>
      </sig:Signature>
... signature ...
      <sig:KeyInfo>
        <wsse:SecurityTokenReference>
          <wsse:Reference URI="#myKey"/>
        </wsse:SecurityTokenReference>
      </sig:KeyInfo>
    </sig:Signature>
  </wsse:Security>
</soap:Header>
<soap:Body wsu:Id="myMsg">
  <app:StockSymbol ...>
... application sub messages ...
  </app:StockSymbol>
</soap:Body>
</soap:Envelope>
```

Black - SOAP elements
Red - WSS elements/attributes
Green - XML Signature elements

FIGURE 5.10 – Document échangé

CONCLUSION

Concevoir un système hétérogène ayant une fonctionnalité d'interopérabilité s'avère souvent difficile et mérite beaucoup d'analyse. La migration à partir des différentes démarches administratives manuelles et migrer vers la solution numérique nous a permis d'obtenir beaucoup plus de facilités. Les recherches et les études sur le système ont abouti à une plateforme web et mobile pour les utilisateurs muni d'un système capable de s'interagir avec les différents types de système d'information pour les fournisseurs

BILAN

Ce mémoire a pour ambition de créer un système d'assurance en ligne permettant à un utilisateur de gérer toutes ses assurances peu importe le nombre d'assurances dont il s'octroie ou le nombre de fournisseurs qu'il choisit. Le but ultime du projet est de faciliter la délivrance et la gestion des assurances. Le but de cette recherche est de concevoir et puis réaliser un système qui permet un échange de données sécurisé dans toutes les termes de sécurité. Les deux parties concernées sont les fournisseurs d'assurance et les utilisateurs. Ce rapport mentionne les étapes traversées pour arriver au résultat attendu. Il développe le contexte dans lequel le projet veut se fonder puis, identifie les différentes exigences du futur système. Au moyen de la méthode et analyse EBIOS, nous faisons l'effort de travailler dans les normes et standards internationaux.

Le travail est divisé en trois parties : la première partie consiste à la réalisation de l'interface du système comprenant le choix des outils adaptés avec son développement, la seconde partie concerne la conception de la base de données d'escale sur qui reposera le système ; cette partie concerne le choix du type de système de base de données et la conception de la base elle-même en prenant en compte le grand paramètre exigé pour permettre une évolution future ; et la troisième partie du document décrit le système de sécurité adopté par la plateforme ainsi que sa mise en œuvre.

Cela nous a permis de comprendre la difficulté de gérer la création d'un système afin de travailler dans la norme, la difficulté dans le choix des technologies. Ainsi, à l'issue des travaux réalisés, nombreuses optimisations sont encore envisageables : optimisations au niveau des technologies utilisées, optimisation au niveau des étapes des échanges.

Bibliographie

- [1] *Algorithme de canocalisation.*
 - [2] *MongoDB, Manual.*
 - [3] *MongoDB, PHP.*
 - [4] *The SOAP Protocol.*
 - [5] Sébastien Adam. *Réalisez votre blog avec le framework CodeIgniter 3.* 2017.
 - [6] ANSSI/ACE/BAC. *EBIOS – Méthode de gestion des risques.* 2010.
 - [7] Patrick CHAMBET. *Sécuriser ses services Web.* 2009.
 - [8] Jean-Marie Chauvet. *Communiquer : SOAP.* 2002.
 - [9] Kristina Chodorow. *MongoDB, The Definitive Guide.* 2013.
 - [10] Sabrina De Capitani di Vimercati Pierangela Samarati Claudio A. Ardagna, Ernesto Damiani. *XML Security.*
 - [11] Ashwani Kumar. *Enabling Extreme Scalability with NoSQL.* 2018.
 - [12] Martin Naedele. *Standards for XML and Web Services Security.*
 - [13] Yogesh Patel Rohit Gathol. *Beginning PhoneGap.*
-

TITRE : Echange de données sécurisé pour un système d'assurance en ligne
AUTEUR : RAZAFIMANANATSOA Nasoavina Christinah
ENCADREUR : SOLOFONIAINA Joelson
CO-ENCADREUR : RASOANAIVO Andry

Résumé

Ce projet vise à permettre une facilité d'obtention et de gestion d'assurances. Dans ce travail, une partie permet que à partir de la plateforme, un client peut gérer librement les assurances qu'il possède à travers une application mobile ou un site internet avec les framework Codeigniter et PhoneGap. Une autre partie du travail consiste à réaliser un échange de données sécurisé entre les différents acteurs dans le but d'avoir une bonne interopérabilité par l'intermédiaire du service SOAP. On se doit de travailler dans les normes donc on a choisi la méthode EBIOS pour la sécurité des différents systèmes d'information concernés.

Mots clés :Echange, Sécurité, Système d'information, Base de données, Norme.

Abstract

This project aims to enable ease of obtaining and managing assurance. In this work, a part allows that from the platform, a customer can freely manage all his assurances through a mobile application or a website with the Codeigniter and PhoneGap frameworks. Another part of the work is to realize a Secure data exchange between the various actors in order to have a good interoperability via the SOAP service. We must work in the standards so we chose the EBIOS method for the security of the different information systems concerned.

Keywords : Exchange, Security, Information System, Database, Standard.
