**Conception et Développement d’une Application Web de Gestion d’Assurance**

Réalisé par :

#### Encadrant académique :

#### Encadrant professionnel :

Société d’accueil :

# TABLE DES MATIÈRES

[INTRODUCTION GÉNÉRALE 1](#_bookmark0)

[CHAPITRE 1 : CADRE GENERALE DU PROJET 2](#_bookmark1)

1. [Introduction 3](#_bookmark2)
2. [Présentation de la société 3](#_bookmark3)
3. [Cadre de projet 3](#_bookmark4)
   1. [Présentation de projet 3](#_bookmark5)
4. [Etude et critique de l’existence 4](#_bookmark6)
   1. [Les Problématiques 4](#_bookmark8)
   2. [Solution proposée 5](#_bookmark9)
5. [Cadre méthodologique 5](#_bookmark10)
   1. [Méthodologie de travail 5](#_bookmark11)
      1. [Etude comparative: 5](#_bookmark12)
      2. [Modèle en V 6](#_bookmark14)
   2. [Design pattern Model-View-Controller 7](#_bookmark15)
      1. [Présentation 7](#_bookmark16)
   3. [Un modèle a trois couches 7](#_bookmark17)
   4. [Fonctionnement de modèle MVC 7](#_bookmark19)
6. [Conclusion 8](#_bookmark21)

[CHAPITRE 2 : MODELISATION DES BESOINS 9](#_bookmark22)

1. [Introduction 10](#_bookmark23)
2. [Spécification des besoins 10](#_bookmark24)
   1. [Les besoins fonctionnels 10](#_bookmark25)
   2. [Les besoins non fonctionnels 10](#_bookmark26)
3. [Langage de modélisation 11](#_bookmark27)
4. [Identification des acteurs 11](#_bookmark29)
5. [Les diagrammes de cas d’utilisations 12](#_bookmark31)
   1. [Diagramme de cas d’utilisation global 13](#_bookmark32)
      1. [Description textuelle de cas d’utilisations « S’authentifier» 13](#_bookmark34)
   2. [Diagramme raffiné de cas d’utilisation de l’administrateur 14](#_bookmark36)
      1. [Diagramme du cas d’utilisation « Gérer les clients» 15](#_bookmark38)
      2. [Description textuelle de cas d’utilisation « Ajouter un client » 15](#_bookmark40)
      3. [Description textuelle de cas d’utilisation « Ajouter un véhicule » 16](#_bookmark42)
      4. [Diagramme du cas d’utilisation « Gérer les véhicules » 17](#_bookmark44)
      5. [Description textuelle de cas d’utilisation « Ajouter une assurance » 17](#_bookmark46)
      6. [Diagramme du cas d’utilisation « Consulter les messages » 18](#_bookmark48)
      7. [Description textuelle de cas d’utilisation « Consulter les messages » 18](#_bookmark50)
   3. [Diagramme raffiné de cas d’utilisation de Visiteur 19](#_bookmark52)
      1. [Description textuelle de cas d’utilisation «Envoyer un message » 19](#_bookmark54)
6. [Conclusion 19](#_bookmark56)

[CHAPITRE 3 : CONCEPTION 20](#_bookmark57)

1. [Introduction 21](#_bookmark58)
2. [Conception de l’application 21](#_bookmark59)
   1. [Vue statique 21](#_bookmark60)
      1. [Diagramme des classes 21](#_bookmark61)
      2. [Présentation des classes 22](#_bookmark63)
   2. [Vue dynamique 23](#_bookmark64)
      1. [Diagramme de séquence 23](#_bookmark65)
      2. [Diagramme de séquence « s’authentifier » 24](#_bookmark66)
      3. [Diagramme de séquence « Ajouter un client » 25](#_bookmark68)
      4. [Diagramme de séquence « Ajouter un véhicule » 26](#_bookmark70)
      5. [Diagramme de séquence « Modifier un client » 27](#_bookmark72)
3. [Conclusion 27](#_bookmark74)

[CHAPITRE 4 : IMPLEMENTATION ET REALISTAION 28](#_bookmark75)

1. [Introduction 29](#_bookmark76)
2. [Environnement de travail 29](#_bookmark77)
   1. [Environnement matériel 29](#_bookmark78)
   2. [Environnement de développement 29](#_bookmark80)
   3. [Architecture de notre application 32](#_bookmark81)
3. [Principales interfaces graphiques 32](#_bookmark83)
4. [Conclusion 42](#_bookmark105)

[CONCLUSION GENERALE 43](#_bookmark106)

[BIBLIOGRAPHIE 44](#_bookmark107)

**LISTE DES FIGURES**

[Figure 1:L'architecture MVC 7](#_bookmark18)

[Figure 2:Echange d'information entre les éléments 8](#_bookmark20)

[Figure 3:Composition UML 11](#_bookmark28)

[Figure 4:diagramme cas d’utilisation global 13](#_bookmark33)

[Figure 5:Diagramme de cas d’utilisation administrateur 14](#_bookmark37)

[Figure 6:Diagramme de cas d’utilisation « Gérer les clients » 15](#_bookmark39)

[Figure 7:Diagramme de cas d’utilisation « Gérer les véhicules » 17](#_bookmark45)

[Figure 8:Diagramme de cas d’utilisation « Consulter les messages » 18](#_bookmark49)

[Figure 9:Diagramme de cas d’utilisation « visiteur » 19](#_bookmark53)

[Figure 10:Diagramme de classes global 22](#_bookmark62)

[Figure 11:Diagramme de séquence « s’authentifier » 24](#_bookmark67)

[Figure 12:Diagramme de séquence « Ajouter un client » 25](#_bookmark69)

[Figure 13:Diagramme de séquence « Ajouter un véhicule » 26](#_bookmark71)

[Figure 14:Diagramme de séquence « Modifier un client » 27](#_bookmark73)

[Figure 15:L’architecture de notre application 32](#_bookmark82)

[Figure 16:Page d’accueil visiteur 33](#_bookmark84)

[Figure 17:les services offrent par notre agence 33](#_bookmark85)

[Figure 18:les services offrent par notre agence 33](#_bookmark86)

[Figure 19: les services offrent par notre agence 34](#_bookmark87)

[Figure 20:Barre d’informations de contact au-dessus de page d’accueil 34](#_bookmark88)

[Figure 21:Envoyer un message 34](#_bookmark89)

[Figure 22:Espace d’authentification administrateur 35](#_bookmark90)

[Figure 23:Espace de vérification 35](#_bookmark91)

[Figure 24:Email 36](#_bookmark92)

[Figure 25:Barre latérale 36](#_bookmark93)

[Figure 26:Page d’accueil administrateur 37](#_bookmark94)

[Figure 27:Liste des clients 37](#_bookmark95)

[Figure 28:Ajouter un client 38](#_bookmark96)

[Figure 29:Modifier un client 38](#_bookmark97)

[Figure 30:Voir client 39](#_bookmark98)

[Figure 31:Ajouter un véhicule 39](#_bookmark99)

[Figure 32:Ajouter une assurance 40](#_bookmark100)

[Figure 33:Liste des clients avec des véhicules assuré et non assuré 40](#_bookmark101)

[Figure 34:Gestion d’assurance 41](#_bookmark102)

[Figure 35:Contrat en PDF 41](#_bookmark103)

[Figure 36:Les messages envoyés par les visiteurs. 42](#_bookmark104)

[Tableau 1:Points forts et faibles 4](#_bookmark7)

[Tableau 2:Etude comparative 6](#_bookmark13)

[Tableau 3:les acteurs et leurs rôles 12](#_bookmark30)

[Tableau 4:Description textuelle de cas d’utilisation « S’authentifier» 14](#_bookmark35)

[Tableau 5:Description textuelle de cas d’utilisation « Ajouter un client» 16](#_bookmark41)

[Tableau 6:Description textuelle de cas d’utilisation « Ajouter un véhicule» 16](#_bookmark43)

[Tableau 7:Description textuelle de cas d’utilisation « Ajouter une assurance» 17](#_bookmark47)

[Tableau 8:Description textuelle de cas d’utilisation « Consulter les messages » 18](#_bookmark51)

[Tableau 9:Description textuelle de cas d’utilisation « Envoyer un message» 19](#_bookmark55)

[Tableau 10:Caractéristique de l'ordinateur 29](#_bookmark79)

**MVC =** Modèle-vue-contrôleur **UML =** Unified Modeling Language **PDF =** Portable Document Format **SQL =** Structured Query Language

**SGBD =** Système de Gestion de Base de Données

**PHP =** Hypertext Preprocessor **HTML =** HyperText Markup Language **CSS =** Cascading Style Sheet

**VSC =** Visual Studio Code

# INTRODUCTION GÉNÉRALE

L’Homme cherche inévitablement la facilité, la rapidité, la flexibilité, l’innovation et une certaine amélioration dans la vie quotidienne. Dans nos jours, Le développement informatique permet de répondre à la demande des entreprises afin de leur créer des programmes personnalisés qui peuvent leur simplifier la vie, voire développer leur activité, assurer ses biens exemples (véhicules) et bien organiser les contrats des clients….

C’est dans ce contexte que notre projet de fin d’étude consiste à développer une application web dynamique et sécurisée dont le but de faire des contrats d’assurances aux clients pour assurer leurs véhicules.

A partir de la conception, nous nous lançons dans le développement de notre application, tout en respectant les directives de la conception.

Ce présent rapport est composé de quatre chapitres.

Dans le premier chapitre nous avons commencé par la présentation de notre stage qui s’est déroulé au sein de l’assurance « Lloyd ».Puis nous avons identifié les critiques de l’existence,

D’autre part, nous avons focalisé sur la méthodologie appliquée pour la bonne gestion de notre projet. Aussi nous avons présenté l’architecture générale.

Dans le deuxième chapitre, nous avons identifié et spécifié les besoins de notre système, les langages de modélisation ainsi les acteurs de l’application.

Dans le troisième chapitre nous allons présenter l’étude conceptuelle du projet. Le dernier chapitre portera l’objet de la réalisation et interprétation des résultats.

Nous clôturons notre rapport avec une conclusion générale et quelques perspectives.

# CHAPITRE 1 : CADRE GENERALE DU PROJET

## Introduction :

L’objectif de ce chapitre est de mettre notre projet dans son cadre général. Tout d’abord nous commençons par la présentation de société d’accueil. En premier lieu nous présentons le sujet en s’appuyant sur la problématique traitée. En deuxième lieu nous parlons de notre projet en citons un exemple d’application web de gestion d’assurance en précisons les avantages et les inconvénients.

## Présentation de la société :

Fondée en 1945, LLOYD ASSURANCES est une compagnie multi branches proposant une gamme complète de produits destinés à une clientèle de particuliers, de professionnels et d’entreprises. Après une opération de recapitalisation menée en 2014 et 2015 par le Groupe BAYAHI, LLOYD ASSURANCES s’est engagée dans une stratégie de modernisation de ses standards de travail pour répondre plus efficacement aux exigences du métier ainsi qu’aux besoins évolutifs de ses assurés **[1]**.

## Cadre de projet :

### Présentation de projet :

Le but de notre projet « Développement d’une application web de gestion d’assurance ».

Ce projet est réalisé pour faciliter le travail de l’assurance.

En premier lieu, la page d’accueil affiche le nombre total de client et véhicule et nombre d’assurance total, les messages et l’historique de notre application.

En deuxième lieu il permet de stocker les données personnelles de chaque client (nom, prénom, numéro carte identité nationale, ville…) de même les données techniques de ses véhicules (immatricule, marque, énergie…) et les garanties qu’il veut faire pour assurer son véhicule (incendie, accident, bris de glace…).

Chaque assurance a une date début et date fin ça dépend du fractionnement que le client a choisi (annuel, trimestriel et semestriel) et puis il peut la télécharger en PDF aussi les visiteurs du site peuvent contacter l’assurance par messages.

Donc, l’objectif de notre projet est d’organiser et bien gérer les informations des clients et leurs véhicules.

## Etude et critique de l’existence :

Dans cette section, nous allons faire part d’une plateforme ayant les mêmes objectifs que la nôtre. Dans ce cadre nous effectuons des recherches sur quelques applications similaires à mon sujet qui existent dans le monde. Nous avons présenté dans le tableau suivant les avantages et les inconvénients de cette plateforme.

PASSInsurance : est une plateforme qui permet à l’administrateur de faire des contrats d’assurance pour assurer les véhicules des clients.



La figure 3 représente la page d’accueil« PASSInsurance ».

Le tableau 1.1 représente les avantages et les inconvénients « PASSInsurance ».

|  |  |
| --- | --- |
| **Avantage** | **Inconvénient** |
| Un site web responsif  Trier les clients par catégories. | Il n’y a pas d’espace de communication entre l’administrateur et le client.  Le site n’affiche pas les services de l’assurance |

**Tableau 1:Points forts et faibles**

### Les Problématiques

Les primordiaux problèmes que nous avons révélés lors de notre séjour au service et qui ont vraiment attiré notre attention sont :

* Il n’y a pas d’espace de communication entre l’administrateur et le client.
* L’application n’affiche pas les statistiques.
* N’affiche pas les services de l’assurance

### Solution proposée

L’étude de l’existant nous a permis de dégager plusieurs anomalies que nous avons détaillées dans la section précédente. Afin de remédier à ces lacunes et dans le but de faciliter certaines tâches, nous formulerons une solution de la problématique. Ainsi dans cette section, nous proposons de développer une plateforme pour répondre à diverses fonctionnalités.

En effet avec ce système l’administrateur peut créer des contrats d’assurance et télécharger ces derniers en PDF et aussi de bien organiser les données des clients et leurs véhicules.

De plus, les clients ont la possibilité d’envoyer des messages

## Cadre méthodologique :

### Méthodologie de travail :

#### Etude comparative:

Le processus de développement constitue un facteur déterminant dans la réussite d’un projet, du fait qu’il cadre ses différentes phases et caractérise les principaux traits de sa conduite. Pour cela, le choix d’une méthode de développement, qui soit adéquate aux particularités et exigences d’un projet, doit être élaboré au préalable afin d’obtenir un produit de qualité qui répond aux besoins et aux attentes des utilisateurs. Parmi les processus de développement les plus répandus on trouve :

|  |  |  |  |
| --- | --- | --- | --- |
|  | **Description** | **Points forts** | **Points faibles** |
| **Cascade** | Propose de dérouler les  phases projet de manière séquentielle. **[2]** | Distingue clairement les phases projet. | Ne propose pas de modèle de document.  Non itératif. |
| **En V** | Il est représenté par un V dont la branche descendante contient toutes les étapes du projet, et  branche montante toutes | Facile à utiliser.  Le contrôle se fait progressivement à chaque étape. | Non itératif.  Ne contient pas les activités d’analyses de risques.  Une mauvaise prise en  compte de changement de |

|  |  |  |  |
| --- | --- | --- | --- |
|  | les étapes de tests du projet  **[3]** | Les tests sont effectués à  chaque étape. | la spécification des  besoins. |
| **Scrum** | Un schéma d’organisation | Scrum est centrée sur le | Projet Scrum sans Product |
|  | de développement de | produit. | owner actif et sans acteurs |
|  | produits complexes. Il est | Le charge de travail est | économiques engagés sera |
|  | défini par ses créateurs | réglable, basé sur la | probablement fiasco. |
|  | comme un « cadre de | capacité de l’équipe et la | Si le propriétaire du produit |
|  | travail permettant de | priorité des taches à | n’a pas apprécié Scrum, ou |
|  | répondre à des problèmes | réaliser. | ne comprend pas le rôle du |
|  | complexes et changeants, |  | propriétaire de produit le |
|  | tout en livrant de manière |  | succès est plus difficile. |
|  | productive et créative des |  |  |
|  | produits de la plus grande |  |  |
|  | valeur possible ». |  |  |

#### Tableau 2:Etude comparative

 Après avoir étudié la nature de notre projet, ce dernier incite à suivre le modèle en V.

#### Modèle en V :

Le modèle en V implique toutes les étapes du cycle de mise en œuvre du produit ou du logiciel. Il peut définir le flux d'un projet en neuf étapes, divisées en trois étapes : conception, production et vérification.

**Conception** : Initialement, la phase de conception peut recueillir les exigences et. Des spécifications sont nécessaires pour programmer l'ensemble du projet avec précision. La conception d'un logiciel à l'aide de la méthode V-Cycle comporte quatre étapes : Analyses exigences, spécifications. Conception globale et conception détaillée.

**Implémentation** : Il s'agit de la fabrication de l'appareil ou du codage du logiciel. Dans ce dernier cas, l'équipe choisit le langage de programmation en fonction des attentes fonctionnelles et architecturales établies lors de la phase de conception. Les étapes de production doivent ensuite suivre les directives et les normes de codage en vigueur.

**Validation** : La phase de vérification comprend divers tests qui vérifient chaque étape. La conception de le produits ces mesures d'assurance. La qualité prend forme tests unitaires, tests d'intégration, tests de vérification et acceptation fonctionnelle.

### Design pattern Model-View-Controller:

#### Présentation :

L’architecture Model\_view\_controler (MVC) permet de séparer l’application en trois parties principaux (le modèle, la vue, le contrôleur). Cette architecture est devenue populaire pour la conception d’application web et mobile car elle permet de bien organiser le code source.

### Un modèle a trois couches :

**Modèle** : représente la partie de l’application qui exécute le logique métier c’est à dire qu’elle est responsable de récupérer les données, de les convertir selon des concepts chargées de sens pour l’application comme les données de l’application, les méthodes manipulant ces données et le stockage et l’extraction de la base de données.

**Vue :** s’occupe des interactions avec l’utilisateur c’est à dire l’interface graphique.

**Contrôleur :** géré la dynamique de l’application. Elle fait le lien entre l’utilisateur et le reste de l’application

La figure suivante schématise le rôle de chacun de ces éléments.



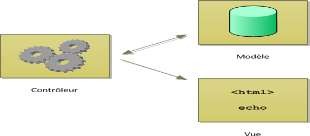
**Figure 1:L'architecture MVC**

### Fonctionnement de modèle MVC :

Le modèle MVC est comme tous les modèles (modèle-vue-présentation) ou (présentation-abstraction-contrôle), est un modèle conçu pour répondre aux besoins de

l’application en séparent les préoccupations liées au différents compostant dans leurs architectures respectives.

La figure suivante schématise comment ces éléments se communiquent entre eux.



#### Figure 2:Echange d'information entre les éléments

Le scénario du modèle MVC est comme suit:

1. L’utilisateur envoie un signal à l’application.
2. Le Contrôleur reçoit la requête, il l’analyse et effectue l’opération sur le Modèle.
3. Il fournit ce Modèle (modifié ou pas) à la vue.
4. La vue s’initialise ou s’actualise en utilisant le Contrôleur.
5. La Vue est représentée à l’utilisateur. En tout le modèle MVC organise bien son code source par faciliter l’affectation des modules à réaliser et le partage entre les différentes couches.

## Conclusion :

Durant ce chapitre nous avons mis notre projet dans le cadre général et sa méthode de développement en vue de déterminer les besoins qu'il faut satisfaire. Dans le prochain chapitre, nous allons jeter un œil sur les besoins spécifiques du projet pour avoir une idée générale de la conception et de la planification du développement de notre application***.***

# CHAPITRE 2 : MODELISATION DES BESOINS

## Introduction :

Ce chapitre présente les différents besoins de l’utilisateur et examine chaque spécification pour aboutir à une synthèse complète de l’utilité de l’application à réaliser. Dans un premier temps, nous présentons les besoins fonctionnels et non fonctionnels. En second temps, nous identifions les acteurs ainsi que les cas d’utilisation de notre application.

## Spécification des besoins :

Cette partie présente notre application les besoins sont différencies en deux catégories : les besoins fonctionnels et les besoins non fonctionnels.

### Les besoins fonctionnels :

L'application est composée de deux volets : L'espace administration et l'espace utilisateur.

Nous allons montrer les besoins fonctionnels de chacun de ces volets. Espace administrateur : joue le rôle principal de notre application.

* S’authentifier
* Consulter les statistiques
* Consulter les historiques
* Consulter les messages
* Ajouter des clients
* Ajouter des véhicules
* Faire l’assurance
* Télécharger l’assurance en PDF
* Répondre aux messages

#### Espace visiteur :

* Envoyer un message

### Les besoins non fonctionnels :

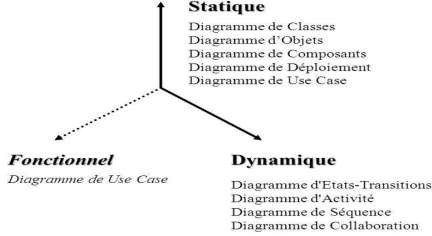
Les besoins non fonctionnels concernent les contraintes à prendre en considération pour assurer le bon fonctionnement du système.

* **L*’extensi*bilité** : dans le cadre de ce travail, l’application devra être extensible, c’est-à- dire qu’il pourra y avoir une possibilité d’ajouter ou de modifier de nouvelles fonctionnalités.
* **La sécurité *:*** la sécurisation est très nécessaire, les informations des clients et des véhicules devront être hautement sécurisés.
* **L’ergonomie** *:* le thème adopte par l’application doit être inspire des couleurs et du logo type de l’entreprise d’accueil**.**
* **La convivialité *:*** l’application doit être simple et facile à prendre en main même pour des non experts.

## Langage de modélisation :

Dans notre projet notre choix est orienté vers le langage de modélisation UML. Dans ces langages nous avons un grand choix des digrammes d’où nous avons choisir :

* Les diagrammes de cas d’utilisation
* Les diagrammes de classe
* Les diagrammes de séquence.



**Figure 3:Composition UML**

## Identification des acteurs :

Un acteur est une entité extérieure qui joue le rôle avec le système par l’échange d’information (entrée /sortie), dans notre application nous avons deux acteurs. Les acteurs de notre application web sont présentés dans le tableau 3.

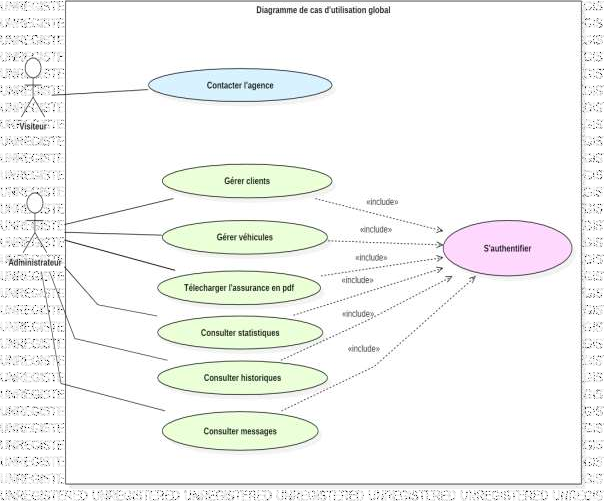
|  |  |
| --- | --- |
| **Acteurs** | **Rôle** |
| **Administrateur** | * Consulter les statistiques * Consulter les historiques * Gérer la liste des clients * Gérer la liste des véhicules * Faire l’assurance des véhicules * Télécharger les assurances en PDF * Consulter les messages envoyés |
| **Visiteur** | * Il peut envoyer des messages à   l’administrateur |

**Tableau 3:les acteurs et leurs rôles**

## Les diagrammes de cas d’utilisations :

Un diagramme de cas d'utilisation est un diagramme de modélisation UML utilisé pour fournir une vue générale du comportement fonctionnel d'une application. Ils peuvent être représentés graphiquement dans des diagrammes et des documents texte.

### Diagramme de cas d’utilisation global :



#### Figure 4:diagramme cas d’utilisation global

Cette figure représente le diagramme de cas d’utilisation global. Nous allons maintenant décrire chaque cas en détail.

#### Description textuelle de cas d’utilisations « S’authentifier» : Cas d’utilisation S’authentifier

**Acteur** Administrateur

**Précondition** L’administrateur visite la page d’accueil.

**Post-condition** L’administrateur est authentifié.

#### Scénario principal

**Scénario alternatif**

* + - 1. L’utilisateur ouvre l’application et clique sur « Se connecter».
      2. Le système affiche l’interface d’authentification.
      3. L’utilisateur saisit les données et clique sur

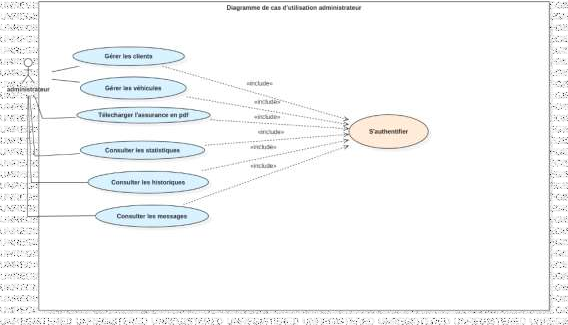
« Se connecter».

* + - 1. Le système vérifie la validité de champs.
      2. Si les données saisis sont corrects il sera dirigé vers profil de l’administrateur avec toutes les fonctionnalités

Si les données sont incorrectes le système affichera un message à l’utilisateur indiquant qu’il n’est pas autorisé à accéder à l’application.

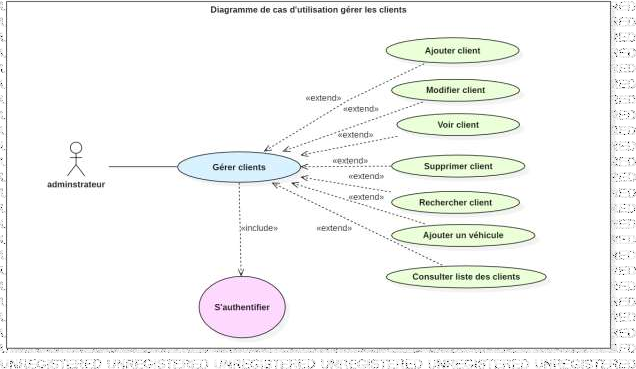
**Tableau 4:Description textuelle de cas d’utilisation « S’authentifier».**

### Diagramme raffiné de cas d’utilisation de l’administrateur :

****

**Figure 5:Diagramme de cas d’utilisation administrateur**

#### Diagramme du cas d’utilisation « Gérer les clients» :

****

**Figure 6:Diagramme de cas d’utilisation « Gérer les clients »**

* + 1. **Description textuelle de cas d’utilisation « Ajouter un client » Cas d’utilisation Ajouter un client**

**Acteur** Administrateur

**Précondition** L’administrateur est authentifié.

**Post-condition** Client est ajouté.

#### Scénario principal

* + - 1. L’administrateur s’authentifie.
      2. Le système vérifie login et mot de passe.
      3. L’administrateur clique sur « liste clients ».
      4. L’administrateur choisit l’option

« ajouter un client ».

* + - 1. Le système affiche le formulaire d’ajout un client.
      2. L’administrateur rempli les champs et valide l’ajout.

#### Scénario alternatif

Le système affiche un message s’il y a un champ vide, les données ne s’envoient pas.

#### Tableau 5:Description textuelle de cas d’utilisation « Ajouter un client».

* + 1. **Description textuelle de cas d’utilisation « Ajouter un véhicule » : Cas d’utilisation Ajouter un véhicule**

**Acteur** Administrateur

**Précondition** L’administrateur est authentifié.

**Post-condition** un véhicule est ajouté.

#### Scénario principal

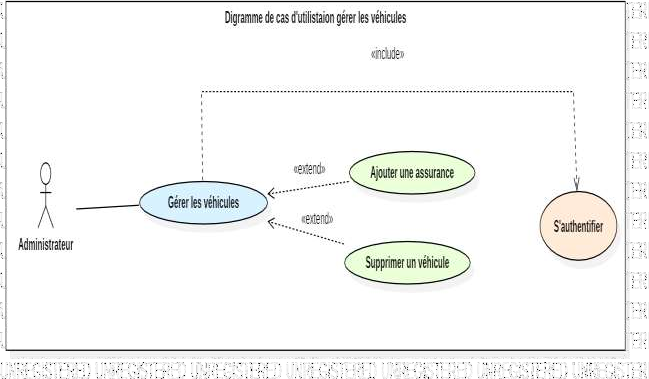
**Scénario alternatif**

* + - 1. L’administrateur s’authentifie.
      2. Le système vérifie login et mot de passe.
      3. L’administrateur clique sur « liste clients ».
      4. L’administrateur choisit l’option « ajouter un véhicule ».
      5. Le système affiche le formulaire d’ajout un véhicule.
      6. L’administrateur rempli les champs et valide l’ajout.

Le système affiche un message s’il y a un champ vide, les données ne s’envoient pas.

**Tableau 6:Description textuelle de cas d’utilisation « Ajouter un véhicule».**

#### Diagramme du cas d’utilisation « Gérer les véhicules » :

****

**Figure 7:Diagramme de cas d’utilisation « Gérer les véhicules »**

* + 1. **Description textuelle de cas d’utilisation « Ajouter une assurance » : Cas d’utilisation Ajouter une assurance**

**Acteur** L’administrateur

**Précondition** L’administrateur est authentifié

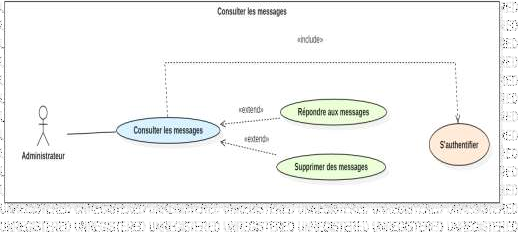
**Post-condition** Ajouter une assurance au véhicule.

#### Scénario principal

1. l’administrateur s’authentifie.
2. le système vérifie login et mot de passe.
3. l’administrateur clique sur « Ajouter des garanties » puis il choisir des garanties et le fractionnement et clique sur « ajouter ».
4. le système informe l’administrateur de succès de l’opération.

**Tableau 7:Description textuelle de cas d’utilisation « Ajouter une assurance».**

#### Diagramme du cas d’utilisation « Consulter les messages » :

****

**Figure 8:Diagramme de cas d’utilisation « Consulter les messages »**

* + 1. **Description textuelle de cas d’utilisation « Consulter les messages » : Cas d’utilisation Consulter les messages**

**Acteur** Administrateur

**Précondition** L’administrateur est authentifié.

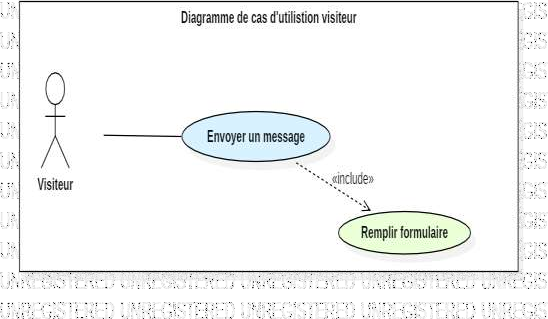
**Post-condition** Répondre aux messages.

#### Scénario principal

1. l’administrateur s’authentifie.
2. le système vérifie login et mot de passe.
3. l’administrateur clique sur « Messages » pour voir les messages envoyer et on peut les supprimer.

**Tableau 8:Description textuelle de cas d’utilisation « Consulter les messages ».**

### Diagramme raffiné de cas d’utilisation de Visiteur :

****

#### Figure 9:Diagramme de cas d’utilisation « visiteur »

* + 1. **Description textuelle de cas d’utilisation «Envoyer un message » : Cas d’utilisation Envoyer un message**

**Acteur** Visiteur

**Post-condition** Envoyer un message.

1) Le visiteur peut consulter notre application web sans s’authentifier et s’il a une question ou il veut prendre un rendez-vous ou il a une

#### Scénario principal

réclamation il clique sur « Envoyer un message » puis il remplit le formulaire.

.

**Tableau 9:Description textuelle de cas d’utilisation « Envoyer un message»**

## Conclusion :

Dans ce chapitre, nous énumérons nos besoins fonctionnels et non fonctionnels de l'application et les acteurs qui l'utilisent. Les diagrammes de cas d'utilisation nous ont permis de formaliser les besoins de manière globale, puis d'expliquer chaque cas de façon textuelle. Après cela, nous continuerons l’étude conceptuelle dans le chapitre suivant

.

# CHAPITRE 3 : CONCEPTION

## Introduction :

Après l’analyse des besoins et pour mieux comprendre la spécification de système dans ce chapitre nous allons présenter la phase conception c’est une phase très importante pour développer notre application avec la présentation de diagrammes des classes ainsi les diagrammes de séquences.

## Conception de l’application

### Vue statique :

Il s’agit de représenter des diagrammes qui définissent l'avis de Conception statique. Ils représentent des schémas de Conception à caractère stable ou immobile pour les différents processus de l'application. Pour structurer notre modèle, nous allons organiser les classes et les regrouper dans un ensemble cohérent. Pour ce faire, nous utilisons le concept générique UML, package. Afin d'organiser nos classes, nous allons utiliser trois principes fondamentaux: cohérence, couplage et cohésion.

Le premier principe consiste à regrouper les classes de point de vue sémantique. Un critère intéressant consiste à évaluer les durées de vie des instances et à rechercher l'homogénéité. Le deuxième principe cherche à minimiser les relations entre les classes. Le troisième principe est le fait de savoir déléguer les tâches de chaque classe. Nous allons s'intéresser au diagramme de classe.

#### Diagramme des classes :

Les diagrammes de classes sont l'un des types de diagrammes UML les plus utiles, car ils décrivent clairement la structure d’un système particulier en modélisant ses classes, ses attributs, ses opérations et les relations entre ses objets. Nous allons présenter la structure statique de notre application à travers le diagramme de classe qui présente un système en termes de classes et de relations.



#### Figure 10:Diagramme de classes global

#### Présentation des classes :

* + - * La classe « Administrateur » cette classe possède les informations d’un administrateur (Email, mot de passe).
      * La classe « Client» cette classe représente les clients que l’administrateur l’ajoute dans l'application, elle définit toutes les informations personnelles. Elle possède les attributs suivants (id client, nom, prénom, numéro de téléphone, sexe, nationalité, ville, code postale, type, type document, numéro document).
      * La classe « Véhicule» possède toutes les informations relatives des véhicules.
      * La classe « Assurance» cette classe définit les différentes garanties que le client veut ajouter pour assurer son véhicule.
      * La classe « Statistique » cette classe ou l’administrateur peut consulter le nombre des clients, véhicules, assurances, messages dans le système.
      * La classe « Historique » cette classe ou l’administrateur peut consulter les historiques de système.
      * La classe « Message » cette classe ou l’administrateur peut consulter les messages envoyé par un visiteur et il peut répondre ou supprimer ces messages.
      * La classe « Visiteur » cette classe ou le visiteur peut envoyer un message.

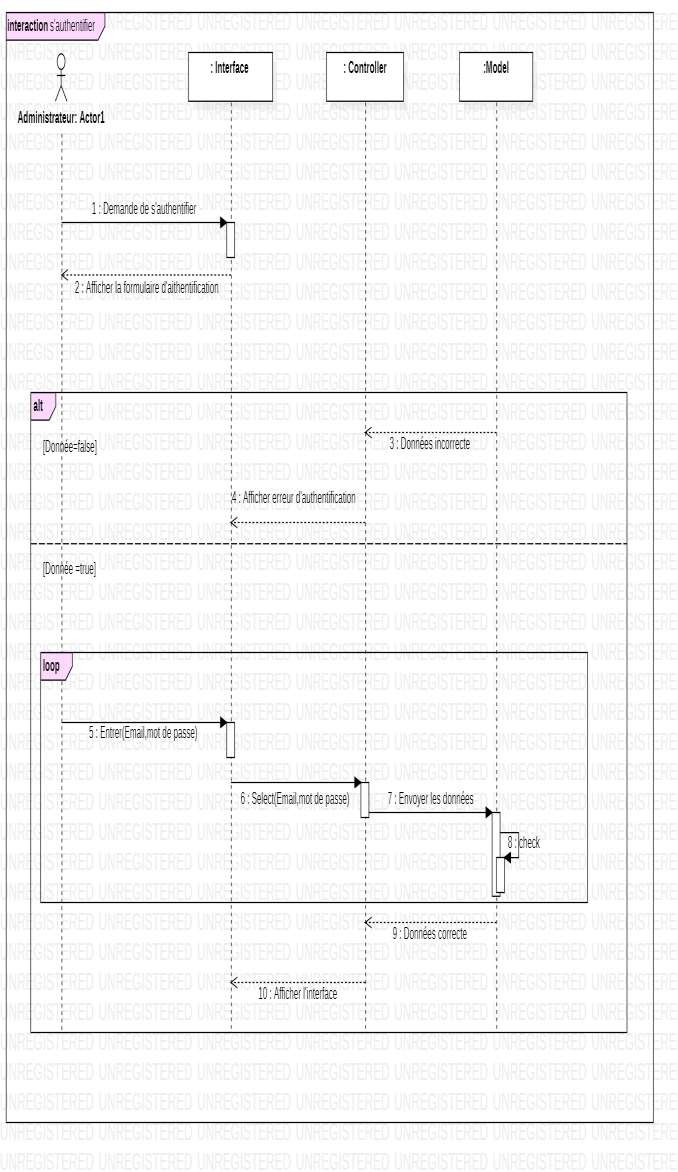
### Vue dynamique :

Les vues dynamiques affichent une vue microscopique du fonctionnement d'une application. Ils permettent de mettre en évidence des relations temporelles entre objets et une représentation de type automate du comportement de chaque objet, intervenant après la définition d'un modèle statique. A ce titre, ils révèlent le comportement du système et l'évolution des objets dans le temps. De plus, ils exposent différents événements du monde extérieur et montrent l'ordre dans lequel ces événements sont provoqués dans le système. Quand on dit "événement", c'est "quelque chose" qui se passe à un moment donné et qui n'a pas de durée précise. Nous allons maintenant nous concentrer sur les diagrammes de séquence.

#### Diagramme de séquence :

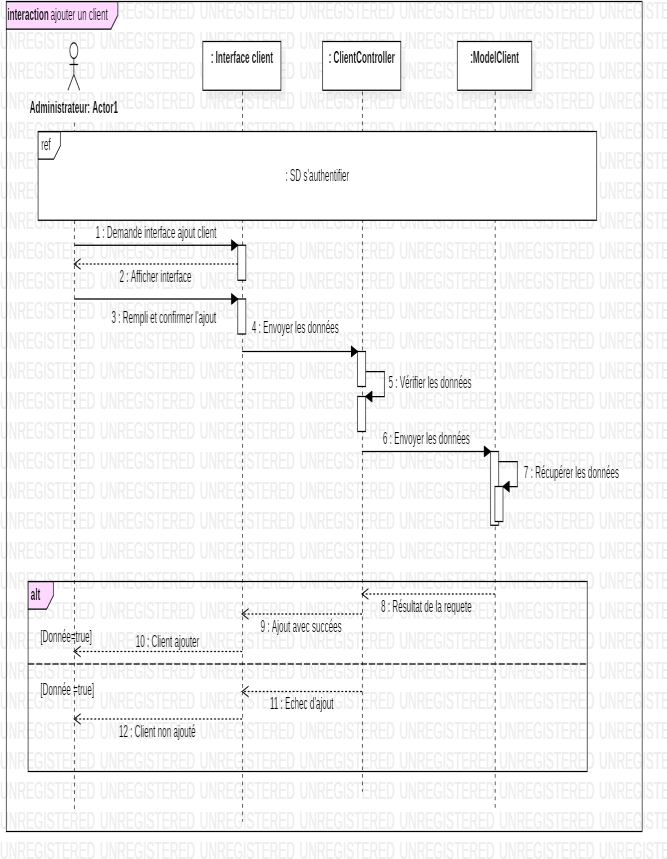
Il s'agit d'une modélisation de l'enchaînement des messages transmis entre objets. Les diagrammes de séquence représentent les interactions entre les objets (données) et les utilisateurs. Ils permettent d'utiliser des séquences spécifiques pour représenter des séquences de scènes. Dans le langage UML, ces diagrammes sont une solution de modélisation dynamique populaire car ils se concentrent principalement sur les lignes de vie, les processus et objets coexistant et les messages transmis entre eux pour exécuter des fonctions avant la fin de la ligne de vie. Un diagramme de séquence de système traite d'un système informatique et est considéré comme une boîte noire. Le comportement du système est exposé de l'extérieur sans nécessiter de connaissance de conception. Ensuite, nous pouvons voir que certains cas d'utilisation sont similaires. Pour cette raison, nous avons choisi d'afficher les diagrammes les plus importants dans notre application.

#### Diagramme de séquence « s’authentifier » :

****

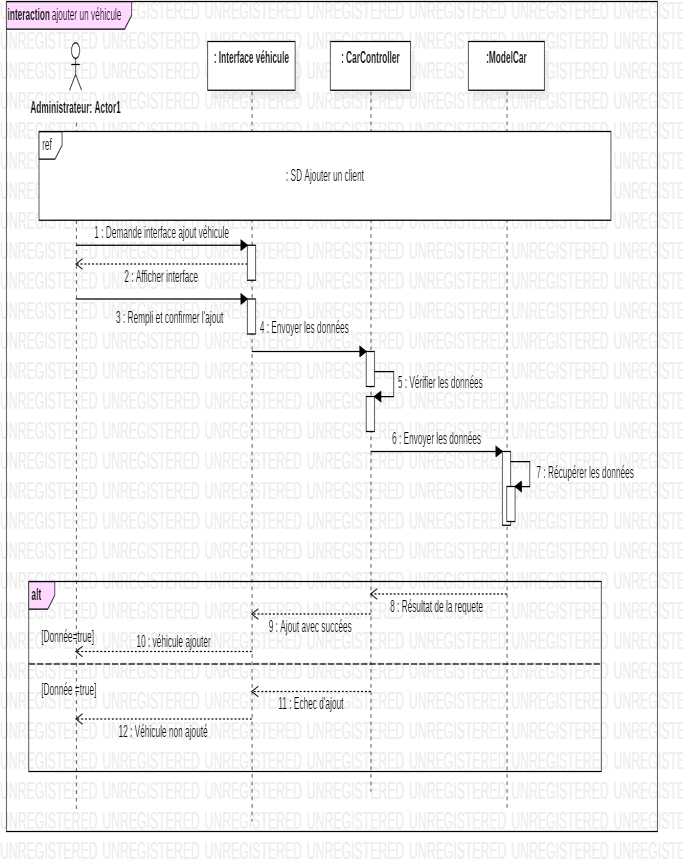
**Figure 11:Diagramme de séquence « s’authentifier »**

#### Diagramme de séquence « Ajouter un client » :

****

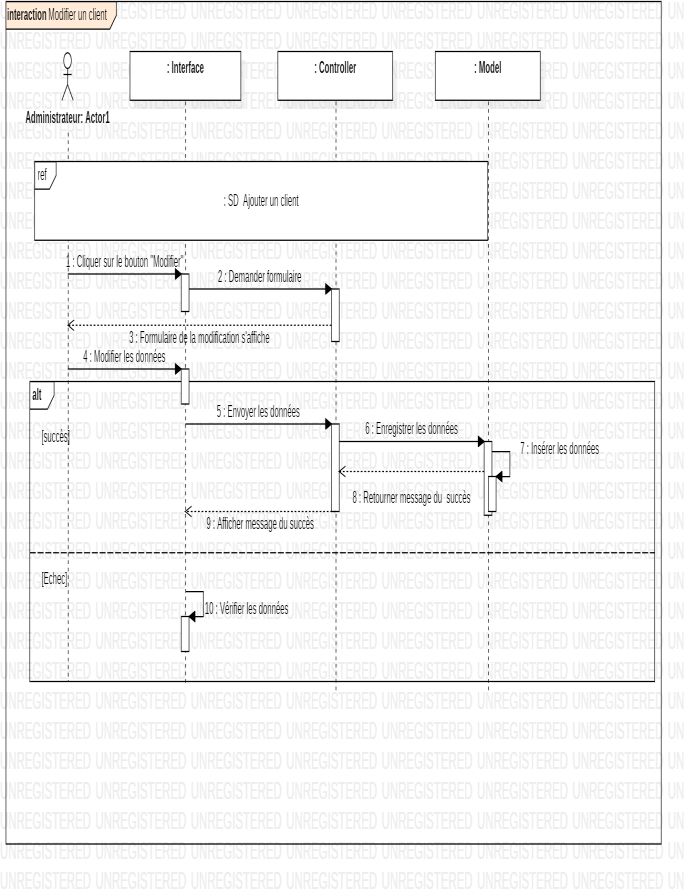
**Figure 12:Diagramme de séquence « Ajouter un client »**

#### Diagramme de séquence « Ajouter un véhicule » :

****

**Figure 13:Diagramme de séquence « Ajouter un véhicule »**

#### Diagramme de séquence « Modifier un client »

****

**Figure 14:Diagramme de séquence « Modifier un client »**

## Conclusion :

Dans ce chapitre nous avons élaboré une vue globale et représentative des différentes parties de notre application. Cette vue globale est accompagnée d'une description détaillée du diagramme de classe et des diagrammes de séquence présentant un nombre de scénarios possibles. Par la suite, nous allons aborder la dernière partie qui représente la réalisation à base de solutions déterminées dans la partie de Conception.

# CHAPITRE 4 : IMPLEMENTATION ET REALISTAION

## Introduction :

Ce chapitre a pour objectif majeur de présenter notre application. C’est la phase de la réalisation de cette application web dynamique qui utilise des technologies spécifiques et la conception des différents composants décrits au niveau du chapitre précédent.

Elle est composée de deux parties : la première consiste à présenter l'environnement matériel et logiciel du travail, tandis que la seconde partie concerne le travail réalisé en détaillant quelques captures d’écrans des fonctionnalités réalisées.

## Environnement de travail :

Dans cette section, nous décrivons donc les outils matériels et logiciels utilisés pour la réalisation de notre projet.

### Environnement matériel :

Pour mener à bien ce projet, nous avons travaillé sur un ordinateur ayant les caractéristiques suivantes.

|  |  |
| --- | --- |
| **Nom pc portable** | **Lenovo** |
| **Processeur** | **Intel Core i5** |
| **Mémoire ram** | **8Go** |
| **Disque dur** | **1To** |
| **Système d’exploitation** | **Windows 10 pro** |

***Tableau 10:Caractéristique de l'ordinateur***

### Environnement de développement :

Dans cette section, nous présentons les différentes technologies adoptées pour la réalisation de notre projet.

 **HTML5 :** (Hypertext Markup Language 5) est la cinquième évolution majeure du HTML. Cette version, finalisée le 28 octobre 2014, servait à la conception de sites web et son

rôle était de formaliser l'écriture de documents grâce à des balises de formatage **[4].**

 **CSS3** : Est le dernier développement du langage (Cascading Style-Sheets), est un langage de feuille de style, utilisé pour décrire le format des documents écrits en langage de

balisage. La dernière version apporte beaucoup de nouveautés, faisant référence à des coins arrondis, des ombres, des dégradés, des transitions ou des animations, ainsi que de nouvelles mises en page comme des mises en page multi-colonnes, des boîtes flexibles ou les mises en pages en grilles **[5].**

 **JAVASCRIPT** : Est un langage de programmation très récent, distinct du langage Html. Il nous a permis d’appliquer de l’interactivité à nos pages web. Sans aucun doute c’est

l’un des langages les plus populaires dans le monde de développement puisqu’il fonctionne sur plusieurs navigateurs web **[6].**

 **MYSQL** : Est un système de gestion de base de données (SGBD). C'est l'un des

plus populaires au monde. Il est distribué sous une double licence, la Licence Publique Générale GNU et la Licence Propriétaire, selon la manière dont il est utilisé. **[7].**

#### 1.1 Framework utilisé :

****

**Laravel** : est un framework Web innovant, complet et open source qui utilise les dernières fonctionnalités PHP et est impeccablement codé et organisé. Laravel adopte le modèle MVC, mais ne l'applique pas. Il s'agit d'une distribution entièrement orientée objet sous licence MIT [8]. C'est un outil qui développe son propre système et augmente l'efficacité en se basant sur le meilleur de plusieurs autres frameworks dans sa conception. Laravel est fondamentalement assez complet pour ne nécessiter qu'une extension pour des besoins spécifiques.

La simplicité et la richesse de la documentation de Laravel peuvent être apprises rapidement et offrent les fonctionnalités suivantes :

* + - Afficher le système de routage
    - Générateur de requêtes SQL et système de migration pour les bases de données
    - Moteur de modèle efficace
    - Système d'envoi d'e-mails
    - Système d'événements et bien d'autres choses.

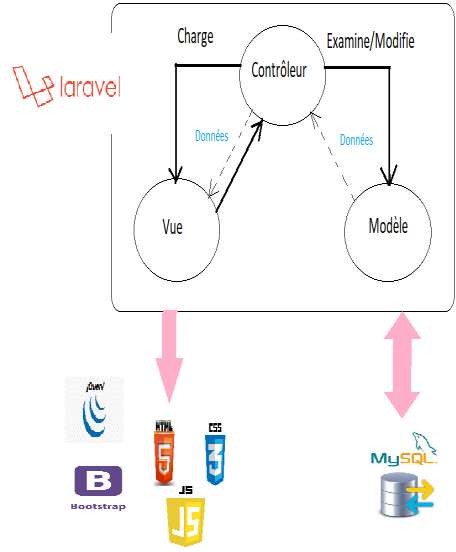
**Bootstrap** : est un framework CSS3, HTML5 et JavaScript créé par les développeurs de Twitter. Bootstrap est un ensemble d'outils qui peuvent être utilisés pour créer des conceptions pour des sites Web et des applications Web (graphiques, animations, interaction avec des pages dans le navigateur, etc.). C'est l'un des projets les plus populaires sur la plateforme de gestion de développement Git Hub. **[9].**

**JQuery** : est un Framework JavaScript sous licence libre et multiplateforme, il a été créé pour faciliter l’écriture de scripts côté client dans le code HTML des pages web.

**Visual studio code** : C'est un éditeur de code extensible développé par Microsoft pour Windows, Linux et MacOs. Visual Studio Code prend en charge presque tous les principaux langages de programmation prêts à l'emploi. Certaines d'entre elles sont incluses par défaut, telles que JavaScript, CSS et HTML, mais d'autres extensions de langage peuvent être trouvées et téléchargées gratuitement sur VS Code Marketplace **[10].**

**StarUML** : Il s'agit d'un logiciel de modélisation UML (Unified Modeling Language) open source et flexible qui prend en charge UML2. Nous l'utilisons lors de la modélisation de diagrammes. Plus important encore, le logiciel est conçu pour prendre en charge la modélisation agile.

### Architecture de notre application :

****

**Figure 15:L’architecture de notre application**

## Principales interfaces graphiques :

Cette partie résume avec une manière claire tout ce que nous avons réalisé au cours de notre travail de projet « Lloyd assurance » même que pour la présentation des scénarios applicatifs de l’application. Nous allons montrer au cours de ce chapitre, les imprimes écran des principales interfaces réalisées. Notre application web repose sur plusieurs interfaces dédiées pour l’administrateur et un formulaire de contact pour les visiteurs.

#### Réalisation et interprétation des résultats :

Les utilisateurs de l’application web principale sont que ce soit un visiteur ou un client qui peut contacter l’administrateur par le remplissage de formulaire.



**Figure 16:Page d’accueil visiteur**

****

**Figure 17:les services offrent par notre agence**

****

**Figure 18:les services offrent par notre agence**



**Figure 19: les services offrent par notre agence**

****

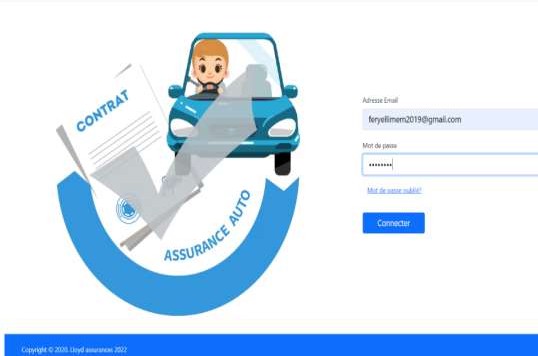
**Figure 20:Barre d’informations de contact au-dessus de page d’accueil.**

Nous avons au-dessus de la page d’accueil un bouton « Envoyer un message » ce bouton affiche un formulaire qui permet au visiteur d’envoyer un message à l’administrateur.



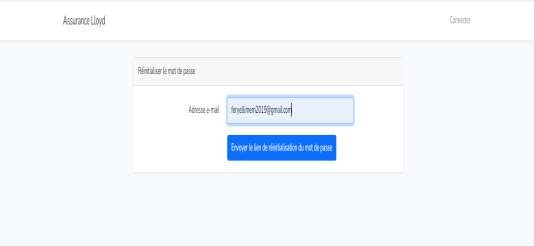
#### Figure 21:Envoyer un message

Cette figure présente l’interface d’authentification de l’administrateur qui présente la saisir de l’adresse email et le mot de passe.



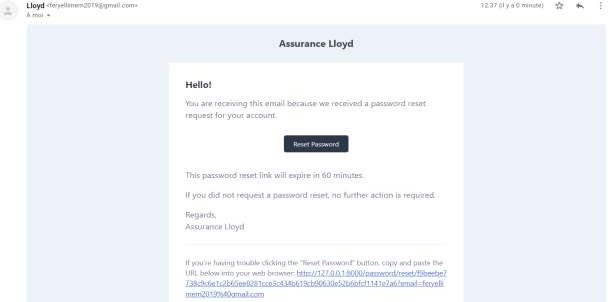
#### Figure 22:Espace d’authentification administrateur

Au cas où nous avons oublié le mot de passe, nous pouvons vérifier avec notre adresse Email.



#### Figure 23:Espace de vérification

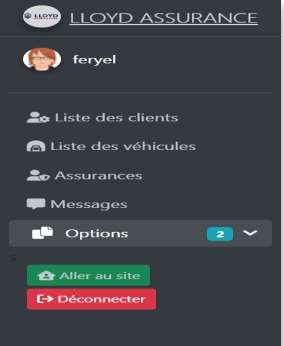
Cette figure montre l’email envoyé par l’application.



#### Figure 24:Email

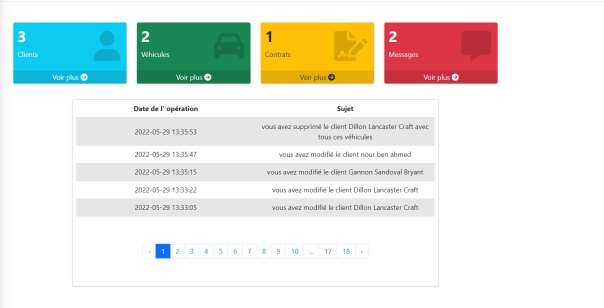
 ***Interface administrateur :***

Cette figure nous présente la barra latérale de l’application



#### Figure 25:Barre latérale

Dans cette figure, nous pouvons voir la page d’accueil d’un administrateur lors de son authentification, dans la première page affichée nous trouvons les statistiques et les historiques de notre site.



#### Figure 26:Page d’accueil administrateur

Si nous cliquons sur la liste de clients le système nous mené vers cette figure suivante ou l’administrateur peut gérer (ajouter, modifier, supprimer ou voir des clients et d’ajouter un ou plusieurs véhicules à chaque client) ainsi qu’il consulte la liste des clients disponibles. Si l’administrateur veut ajouter un client il clique sur « +Ajouter un client ».



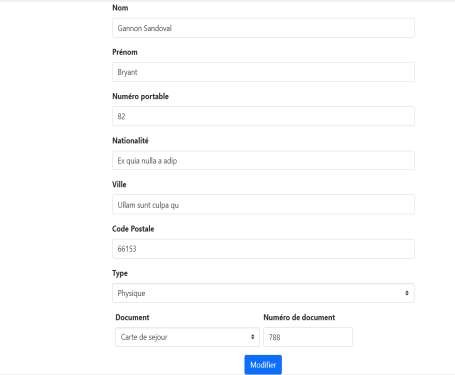
#### Figure 27:Liste des clients

Cette figure montre le formulaire d’ajout d’un client, l’administrateur remplit les champs puis il clique sur « confirmer » pour valider l’opération.



#### Figure 28:Ajouter un client

Cette figure montre le formulaire de modification d’informations d’un client, l’administrateur remplit les champs puis il clique sur « modifier » pour valider l’opération.



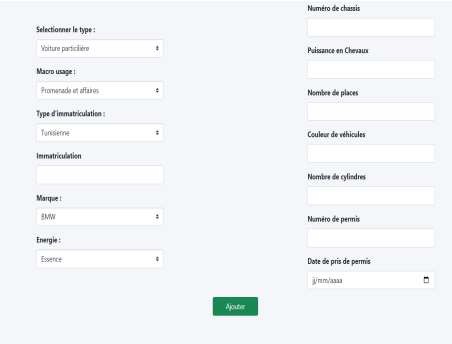
#### Figure 29:Modifier un client

Cette figure montre les informations de client.



#### Figure 30:Voir client

Cette figure montre le formulaire d’ajout d’un véhicule, l’administrateur remplit les champs puis il clique sur « ajouter » pour valider l’opération.



#### Figure 31:Ajouter un véhicule

Cette figure nous montre les garanties que nous pouvons sélectionner pour assurer le véhicule du client.



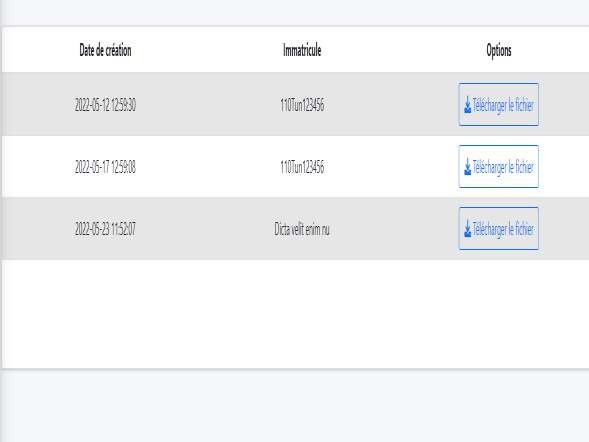
#### Figure 32:Ajouter une assurance

Cette figure montre la liste des assurances, cette liste comporte des contrats complet et des contrats pas encore complet.



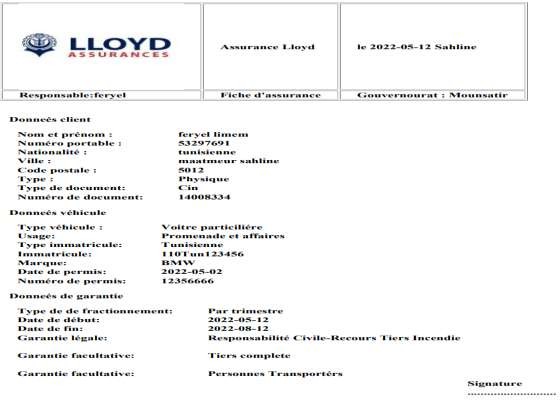
#### Figure 33:Liste des clients avec des véhicules assuré et non assuré.

Cette figure montre la liste des véhicules où nous pouvons télécharger les contrats sous forme des PDF.



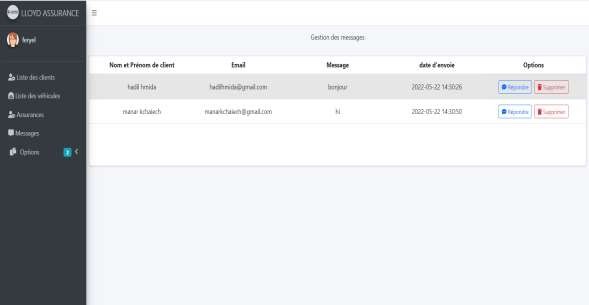
#### Figure 34:Gestion d’assurance.

Cette figure montre le contrat en PDF.



#### Figure 35:Contrat en PDF

Cette figure montre les messages envoyés et leurs expéditeurs et nous pouvons répondre à chaque message ou le supprimer.



**Figure 36:Les messages envoyés par les visiteurs.**

## Conclusion :

Dans ce chapitre, nous avons présentée l’implémentation de chaque cas d’utilisation pour à aboutir l’architecture logicielle de notre application. Nous avons illustré des quelques interfaces utilisateurs et nous avons décrire la description de l’environnement matériel et logiciel qui nous a permis d’implémenter notre application

# CONCLUSION GENERALE

Dans le cadre de notre projet, nous avons conçu et développé une application web de gestion véhicules pour une assurance.

Ce rapport s’articule autour des quatre chapitres structurés comme suit : Le premier chapitre

«introduction générale» est dédié à présenter le contexte du travail. Le deuxième chapitre

«Modélisation des besoins » sera consacré à l’identification des besoins fonctionnels et non fonctionnels attendus de notre application, les acteurs et les diagrammes des cas d’utilisations. Ensuite le troisième chapitre «Conception» détaille l’étude conceptuelle du projet. Le dernier chapitre «Implémentation et réalisation» présente les démarches techniques pour la réalisation de l’application ainsi que l’environnement et les outils utilisés

Pendant ce stage de fin d’études, nous avons acquis des nouvelles compétences dans le domaine du développement informatique. La réalisation de l’application décrite dans ce rapport nous a permis essentiellement de maîtriser une panoplie de langages de programmation. Ainsi, nous avons franchi notre premier pas avec le Framework Laravel qui a été couplé avec d’autres technologies Web comme le Framework Bootstrap, la bibliothèque Jquery. L’objectif déclaré au début de ce travail a été atteint en réalisant une plateforme Web de gestion d’assurance fonctionnelle qui satisfait les besoins des futurs utilisateurs et qui répond à leurs attentes.

Notre projet reste extensible à tout type d’ajout de nouvelles fonctionnalités à fin d’améliorer la qualité de l’application. Nous pouvons ajouter nos perspectives :

* Une partie qui permet d’estimer le prix d’assurance.
* Trier les clients par catégories.
* Mettre un système de communication entre l’administrateur et le client avant l’échéance de l’assurance l’administrateur va envoyer un message au client.

# BIBLIOGRAPHIE

1. : <https://www.lloyd.com.tn/presentation> Consulté le 14/03/2022
2. : <https://www2.stardust-testing.com/blog-fr/cascade-scrum-testing-qa> Consulté le 26/04/2022
3. : <https://blog.hubspot.fr/marketing/cycle-en-v> Consulté le 12/05/2022
4. : <https://fr.wikipedia.org/wiki/HTML5> Consulté le 20/03/2022
5. : <https://www.futura-sciences.com/tech/definitions/internet-css-4050/> Consulté le 20/03/2022
6. : <https://developer.mozilla.org/fr/docs/Web/JavaScript> Consulté le 20/03/2022
7. : <https://sql.sh/sgbd> Consulté le 08/05/2022
8. : <https://laravel.com/docs/5.4> Consulté le 08/05/2022
9. : <https://fr.wikipedia.org/wiki/Bootstrap_(framework)>

Consulté le 09/05/2022

1. : <https://fr.wikipedia.org/wiki/Visual_Studio_Code> Consulté le 09/03/2022

**Résumé**

Ce projet est réalisé de projet de fin d'études, pour l’obtention du diplôme de licence appliquée en télécommunication à l’Institut supérieur de l’informatique et des technologies de la communication (ISITCom).

Ce travail Consiste a réalisé une plateforme web de gestion d’assurance qui nous permet de bien organiser les données des clients et leurs véhicules et de faire des contrats d’assurance.

Nous avons utilisé pour ce faire la technologie Laravel.

**Mots clés** : Laravel, JavaScript

**Abstract**

This project is realized of end of studies project, for the obtaining of the diploma of applied licence in telecommunication in the Higher Institute of the computing and the technologies of the communication (ISITCom).

This work consists in the realization of a web platform of insurance management which allows us to organize well the data of the customers and their vehicles and to make insurance contracts.

We used for this purpose the Laravel technology.

**Keywords :** Lavael , JavaScript