

Ministère D'enseignement Supérieur Et De La
Recherche Scientifique

Université De La Manouba

Projet de fin d'année



Présenté Par

Aymen Samoudi

Intitulé

Conception et développement d'une application Web/Desktop
des annonces de location de voiture

Année universitaire : 2022/2023

Liste des tableaux	
Université De La Manouba	1
<u>Liste des tableaux</u>	<u>2</u>
<u>Liste des figures</u>	<u>3</u>
Liste des tables	4
<u>Partie I : Présentation du projet</u>	<u>5</u>
I.1 Présentation générale du projet	5
I.1.1 Introduction	5
I.1.2 Objectif	5
I.2.3 Etat de l'art	5
I.2.4 La location de voiture	5
I.2.5 La réservation en ligne	6
<u>Partie II : Conception Orientée Objet des Systèmes d'Information</u>	<u>7</u>
<u>II.1- Introduction</u>	<u>7</u>
<u>II.2- Liste des besoins des utilisateurs du système à modéliser</u>	<u>7</u>
<u>II.3 Liste des acteurs du système</u>	<u>8</u>
<u>II.4- Diagramme des cas d'utilisation</u>	<u>8</u>
<u>II.5- Diagramme des classes</u>	<u>9</u>
<u>II.6- Description textuelle des cas d'utilisation</u>	<u>10</u>
5- Diagramme de séquence système pour chaque cas d'utilisation	10
8- Diagramme de séquence détaillé pour chaque cas d'utilisation	11
<u>Partie III : Base de données</u>	<u>26</u>
<u>III.1- Les règles de passage du diagramme de classe au schéma relationnel de la base de données</u>	<u>26</u>
<u>III.2- Le schéma relationnel de la base de données</u>	<u>30</u>
III.3- Normalisation du schéma relationnel (Boyce-Codd)	31
<u>III.4- Les scripts de création de la base de données</u>	<u>32</u>
III.5- Les requêtes d'interrogation et de modification	35
<u>Partie IV : La Réalisation</u>	<u>38</u>
<u>Introduction</u>	<u>39</u>
<u>IV.1- Partie Programmation Web 2</u>	<u>40</u>
<u>IV.2- Partie Environnement de Développement BD</u>	<u>48</u>

Liste des Figures

Figure 1: Diagramme de cas d'utilisation	8
Figure 2: Diagramme des classes	9
Figure 3: Diagramme de séquence de cas d'utilisation "S'authentifier"	10
Figure 4: Diagramme de séquence détaillé "S'authentifier"	11
Figure 5: Diagramme de séquence de cas d'utilisation "S'inscrire"	12
Figure 6: Diagramme de séquence détaillé "S'inscrire"	13
Figure 7: Diagramme de séquence de cas d'utilisation "Réservation"	14
Figure 8: Diagramme de séquence détaillé "Réservation"	14
Figure 9: Diagramme de séquence de cas d'utilisation "Déposer nouveau véhicule"	15
Figure 10: Diagramme de séquence détaillé "Déposer nouveau véhicule"	16
Figure 11: Diagramme de séquence de cas d'utilisation "Modifier véhicule"	17
Figure 12: Diagramme de séquence détaillé "Modifier véhicule"	17
Figure 13: Diagramme de séquence de cas d'utilisation "Supprimer véhicule"	18
Figure 14: Diagramme de séquence détaillé "Supprimer véhicule"	19
Figure 15: Diagramme de séquence de cas d'utilisation "Accepter la location"	20
Figure 16: Diagramme de séquence détaillé "Accepter la location"	20
Figure 17: Diagramme de séquence de cas d'utilisation "Refuser la location"	21
Figure 18: Diagramme de séquence détaillé "Refuser la location"	22
Figure 19: Diagramme de séquence de cas d'utilisation "Mise à jour véhicule"	23
Figure 20: Diagramme de séquence détaillé "Mise à jour véhicule"	23
Figure 21: Diagramme de séquence de cas d'utilisation "Visualise le client réservée"	24
Figure 22: Diagramme de séquence détaillé "Visualise le client réservée"	25
Figure 23: Schéma relationnel de la base de données	30
Figure 24: Les scripts de création de la base de données	35
Figure 25: Interface d'accueil de l'application	41
Figure 26: Interface des voitures de l'application	42
Figure 27: Interface de connexion du client de l'application	42
Figure 28: Interface de détails voiture de l'application	43
Figure 29: Interface de réservation voitures de l'application	43
Figure 30: Interface de suite de réservation voitures de l'application	44
Figure 31: Interface de suite de listes de réservation du client de l'application	44
Figure 32: Interface de connexion agent de l'application	45
Figure 33: Interface d'insertion nouveau véhicule par l'agent l'application	45
Figure 34: Interface des voitures de l'agent	46
Figure 35: Interface de choix d'authentification de différent utilisateur	47
Figure 36: Interface de connexion pour le client	48
Figure 37: Interface de connexion pour l'agent	48
Figure 38: Interface principale pour le client	49
Figure 39: Interface principale de réservation pour le client	49
Figure 40: Interface principale de liste des réservation pour le client	50
Figure 41: Interface principale de liste des réservation pour le client	50
Figure 42: Interface principale de véhicule pour un agent	50
Figure 43: Interface d'inscription pour le client	50

Liste Des Tableau

Tableau 1: Description textuelle de cas d'utilisation "S'authentifier"	10
Tableau 2: Description textuelle de cas d'utilisation "S'inscrire"	11
Tableau 3: Description textuelle de cas d'utilisation "Réservation"	13
Tableau 4: Description textuelle de cas d'utilisation "Déposer nouveau voiture"	15
Tableau 5: Description textuelle de cas d'utilisation "Modifier voiture"	16
Tableau 6: Description textuelle de cas d'utilisation "Supprimer voiture"	18
Tableau 7: Description textuelle de cas d'utilisation "Accepter la location"	19
Tableau 8: Description textuelle de cas d'utilisation "Refuser la location"	21
Tableau 9: Description textuelle de cas d'utilisation "Mise a jour le voiture réservée"	22
Tableau 10: Description textuelle de cas d'utilisation "Visualise le client réservée"	24

Partie I : Présentation du projet

I.1 Introduction :

Nous sommes dans le XXI^e siècle, c'est la mondialisation, la communication entre les hommes devient de plus en plus rapide et par tout, grâce à une nouvelle technologie qui est l'Internet. Aujourd'hui la plus grande partie des personnes communiquent avec Internet parce qu'elle est disponible pour tous et offre des millions d'informations et services.

Pour cela nous avons réalisé un portail web dynamique pour la location des véhicules qui est apparue suite au besoin des clients en Tunisie ou à l'étranger qui peuvent faire la réservation en ligne selon leur besoin et leurs préférences de véhicules avec les agences de location.

Dans ce chapitre on représente l'étude de faisabilité de notre projet de la réalisation d'un portail web pour la réservation des véhicules.

I.2 Objectif :

Pour obtenir une bonne qualité de projet informatique, il faut maîtriser les processus d'élaboration. L'étude de la faisabilité est la première étape de cycle de vie d'un projet après l'idée. Cette étape est une clé de développement, elle permet de déterminer si le développement proposé vaut la peine d'être mis en œuvre et déterminer les fonctionnalités que doit posséder le projet. Grâce à cette phase on peut étudier l'existence de projet, le marché potentiel de produit et ses exigences.

I.3 Etat de l'art :

La phase élémentaire dans l'initiation d'une étude, d'une rédaction, d'un mémoire commence par l'état de l'art. Elle comprend l'état de connaissances, les différentes parties de réalisations, et permet de :

Connaître l'exploitabilité de votre idée.

Vérifier l'originalité de votre idée : elle est réalisée par quelqu'un d'autre ou non.

Donner des nouvelles idées sur le développement de votre projet, et des indications sur les problèmes qui peuvent être trouvés dans la réalisation de projet.

I.3.1 La location de voiture :

La location de voiture est un service offert par des professionnels détenteurs d'automobiles de tourisme ou de véhicules utilitaires. Ce service consiste pour le client (professionnel ou particulier) à réserver et à louer un véhicule pour une période donnée allant de quelques heures à plusieurs mois.

Il existe enfin des systèmes de location de véhicules de luxe avec chauffeurs. Ceux-ci sont réservés pour les événements.

Le marché de la location se partage entre :

La location à courte durée, d'une journée à plusieurs jours.

La location à longue durée, pour un an et plus.

La location à très courte durée, de moins d'une heure à quelques heures de location.

Evolution des usages de location de voiture

La location de voitures est apparue aux Etats-Unis en tant que complément des transports en commun, notamment pour les personnes qui traversaient le pays en avion et qui avaient besoin d'une automobile à leur arrivée. Par la suite, cette pratique s'est diffusée en Europe : la location était proposée dans les gares et les aéroports pour compléter un trajet en train ou en avion. Puis la location s'est développée auprès d'une clientèle d'entreprises locales qui utilisent la location pour leurs besoins ponctuels.

I.3.2 La réservation en ligne :

Définition:

La réservation en ligne est une demande de réservation d'un produit ou d'un service d'une agence par un simple clic (réserver) où le client peut être dans sa maison. Autrement dit, la réservation en ligne est une option rendant les réservations plus rapides, plus sûres, simples et par tout.

Cycle de vie :

1. l'agence ou le vendeur doivent choisir une façon de mettre leur produit sur web:

Crée un site web personnel (une méthode traditionnelle) peut être coûteuse.

S'inscrire dans un portail web qui garantit un grand nombre de visites plus que dans un site web personnel.

2. Mettre ces produits sur le site avec pleins d'informations (nom de produit, le prix, type de vente, location, type de paiement,...).

3. le client doit choisir ses demandes sur web et réserver selon ses besoins en respectant le type de paiement du produit.

Les domaines de réservation en ligne :

Il y a plusieurs domaines qui exploitent la réservation en ligne, parmi ces domaines on cite :

la réservation des voyages.

la réservation des hôtels.

la réservation des voitures (location de voitures).

réservation des mini-services telles : dîner dans un restaurant, nouvelles marques de vêtements, des produits informatiques . . .

L'importance de la réservation en ligne :

Grâce à la réservation en ligne on peut choisir le produit comme dans un magasin réel:
sans se déplacer, et à toute heure de la journée.

sans prendre son téléphone pour communiquer avec une langue étrangère.

sans passer par un intermédiaire physique.

la réservation en ligne influence sur l'indice de Qualité et améliore ainsi le positionnement de l'agence sur le web (la réputation) et augmente le chiffre d'affaire.

Partie II : Conception Orientée Objet des Systèmes d'Information

II.1 Introduction :

La réussite de tout projet dépend de la qualité de son départ. De ce fait, l'étape de spécification des besoins constitue la base de départ de notre travail, elle doit décrire sans ambiguïté le logiciel à développer. Pour assurer les objectifs attendus, il est essentiel que nous parvenions à une vue claire des différents besoins escomptés de notre projet. Au cours de ce partie, nous allons dégager les fonctionnalités attendues du module "location des véhicules" en définissant les différents cas d'utilisation et quelques scénarios qui expliquent ces cas.

II.2 Liste des besoins des utilisateurs du système à modéliser :

II.2.1 Besoins fonctionnels :

Le module de location des véhicules doit permettre aux utilisateurs de consulter les tableaux de bords des services et des missions. Ce module doit mettre à la disposition des acteurs les fonctionnalités suivantes :

Administrateur :

Un administrateur peut visualiser les clients inscrit dans l'application.
Un administrateur peut visualiser et vérifier les nouveau comptes agents par accepter ou refuser.
Un administrateur prend les listes des réservation confirmer entre les clients et les agences.

Agent :

L'agent il peut regarder son véhicules.
Peut crée un compte et attend l'accord par l'administrateur.
Un agent peut crée ou déposer des nouveau voiture et affiche dans la page des véhicules.
Un agent a l'option de supprime des véhicules.
Un agent peut modifier des véhicules aussi.
Si une véhicule réservée par un client, l'agence a le choix d'accord cette réservation.
L'agent permet de visualise le client louer cet véhicule.

Client :

Peut visualise tous les véhicules insérée par les agences.
Le client il peut inscrit et connecté dans l'application.
Ajouter une ou plusieurs véhicules dans l'interface réservation et de choisir le nombre de jour d'une location.
Un client pour le choix d'annuler le réservation.
Le client peut regarder la liste de réservation et attend la réponse oui/non par l'agence possède cette véhicule.

II.2.2 Besoins non-fonctionnels :

Le temps de réponse de l'application doit être le plus rapide possible.
Possibilité d'optimiser l'application plus tard.

II.3- Liste des acteurs du système :

Un acteur représente une abstraction d'un rôle joué par des entités externes au système qui interagissent directement avec lui. Dans cette application, nous identifions trois types d'acteurs : l'administrateur, l'agent et le client.

1. Administrateur : C'est l'acteur qui gère les comptes agents ou clients ainsi que les listes des réservation confirmer. Il se charge aussi de contrôler l'avancement du travail au sein de l'agence. La fonctionnalité la plus importante qu'il a est de vérifier les comptes aux agences et de recevoir les nouveau inscription de chaque agent.

2. Agent : Il s'agit d'un employé de l'agence qui est chargé d'accomplir les tâches qui lui sont assignées. L'application lui offre plusieurs fonctionnalités telles que consulter ses propres véhicules, et de supprimer ou modifier concernant chaque voiture.

3. Client : Le client, qui souhaite éventuellement effectuer des réservation , peut chercher un véhicule avec sont préférence, puis remplir sont informations dans une formulaire pour confirmer la réservation .

II.4 Diagramme des cas d'utilisation :

Le diagramme de la figure 1 illustre le rôle de chaque acteur ainsi que l'étendue de ses responsabilités.

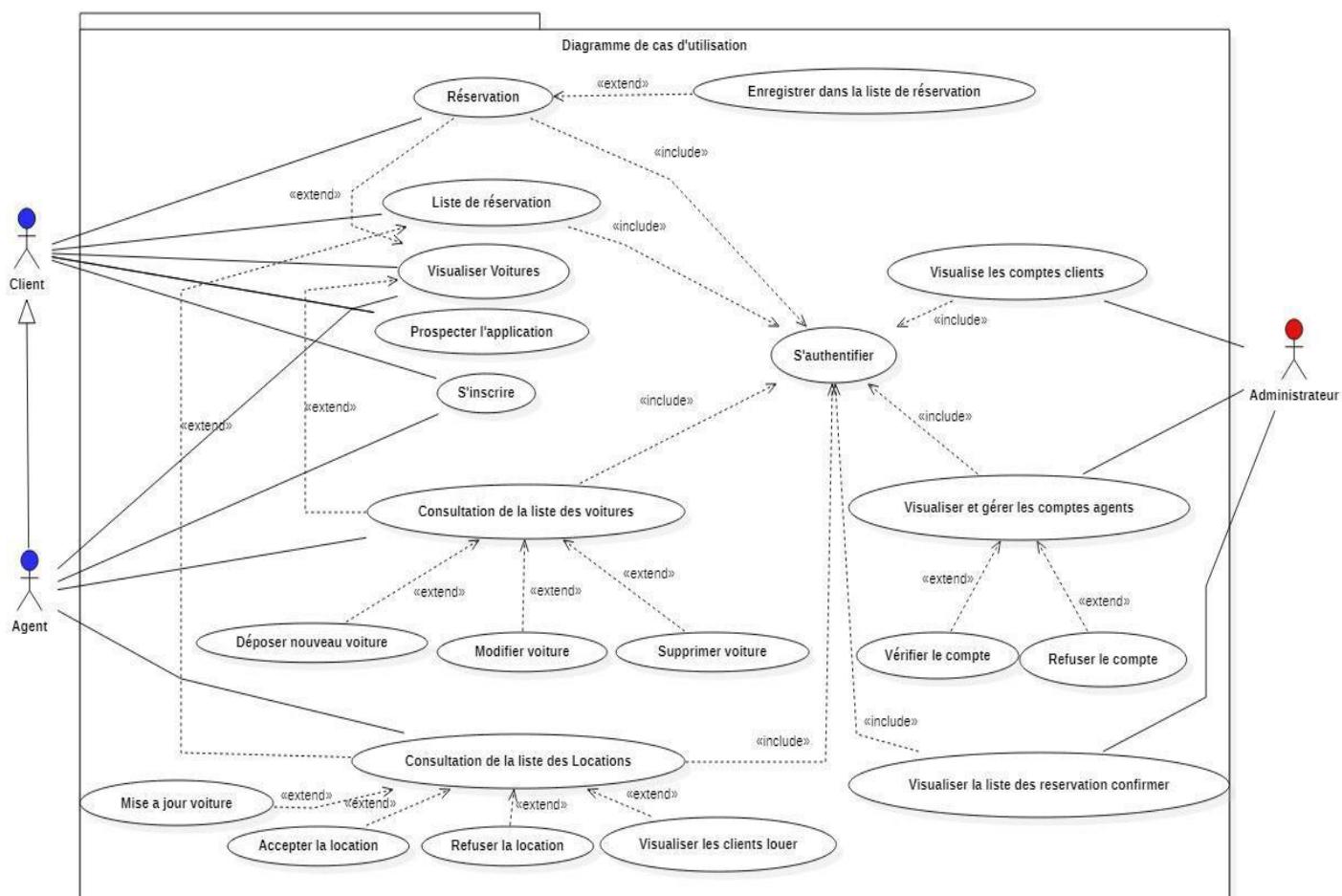


Figure 1: Diagramme de cas d'utilisation

II.5 Diagramme des classes :

La figure ci-dessous récapitule le tableau précédent dans un diagramme de classes qui contient toutes les informations telles que les classes, les attributs, les méthodes et les associations :

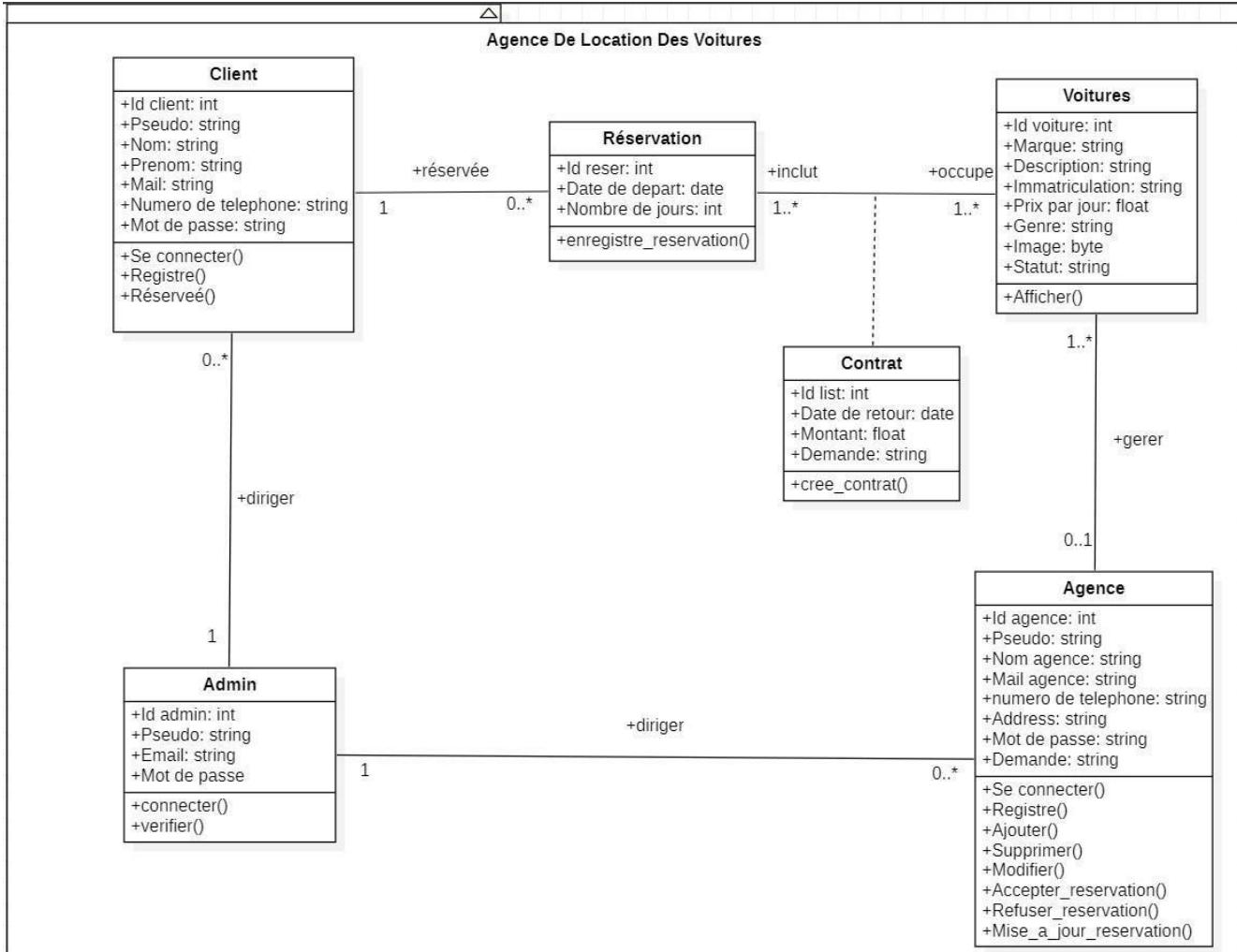


Figure 2: Diagramme des classes

-Client : c'est la classe qui représente les clients de l'application pouvez profiter des services de réservation .

-Agence : c'est la classe qui représente l'agence de l'application il peut profiter de plusieurs services l'ajout, suppression, modification des véhicules et de gérée les clients réservée.

-Réservation : c'est la classe qui permet d'enregistrée chaque location des clients a une date donnée et le nombre de jours saisi.

-Voitures : c'est la classe qui représente les voitures entre par chacune agent et de visualise en publique dans l'application.

-Admin : c'est la classe qui principale qui joue le rôle d'administrateur et gérée les clients el les agent registrée.

-Contrat : c'est une association entre la classe réservation et le voiture, toutes les transaction de client et l'agent doit représentée dans cette association.

II.6 Description textuelle des cas d'utilisation :

Description textuelle de cas d'utilisation : « S'authentifier »

Cas d'utilisation : S'authentifier
Acteurs : Administrateur et autres utilisateurs.
Objectif : Il permet à l'acteur de s'identifier en saisissant son login et mot de passe.
Précondition : L'acteur doit être présent dans la base de données.
Postcondition : <ul style="list-style-type: none"> -Acteur authentifié. -La page d'accueil s'affiche.
Scénario nominal : <ol style="list-style-type: none"> 1. L'acteur ouvre l'application, 2. Le système affiche la page d'authentification, 3. L'acteur saisit le login et le mot de passe, 4. Le système vérifie l'existence des données, 5. Le système affiche la page d'accueil.
Scénario alternatif : <p>A. Erreur d'authentification : login ou mot de passe non valide. Cet enchaînement démarre au point 4. 5. Le système affiche un message d'erreur. Le scénario reprend au point 2.</p> <p>B. Champs obligatoires vides. Cet enchaînement démarre au point 4. Le scénario reprend au point 2.</p>

Tableau 1: Description textuelle de cas d'utilisation "S'authentifier"

Diagramme de séquence système de cas d'utilisation « S'authentifier »

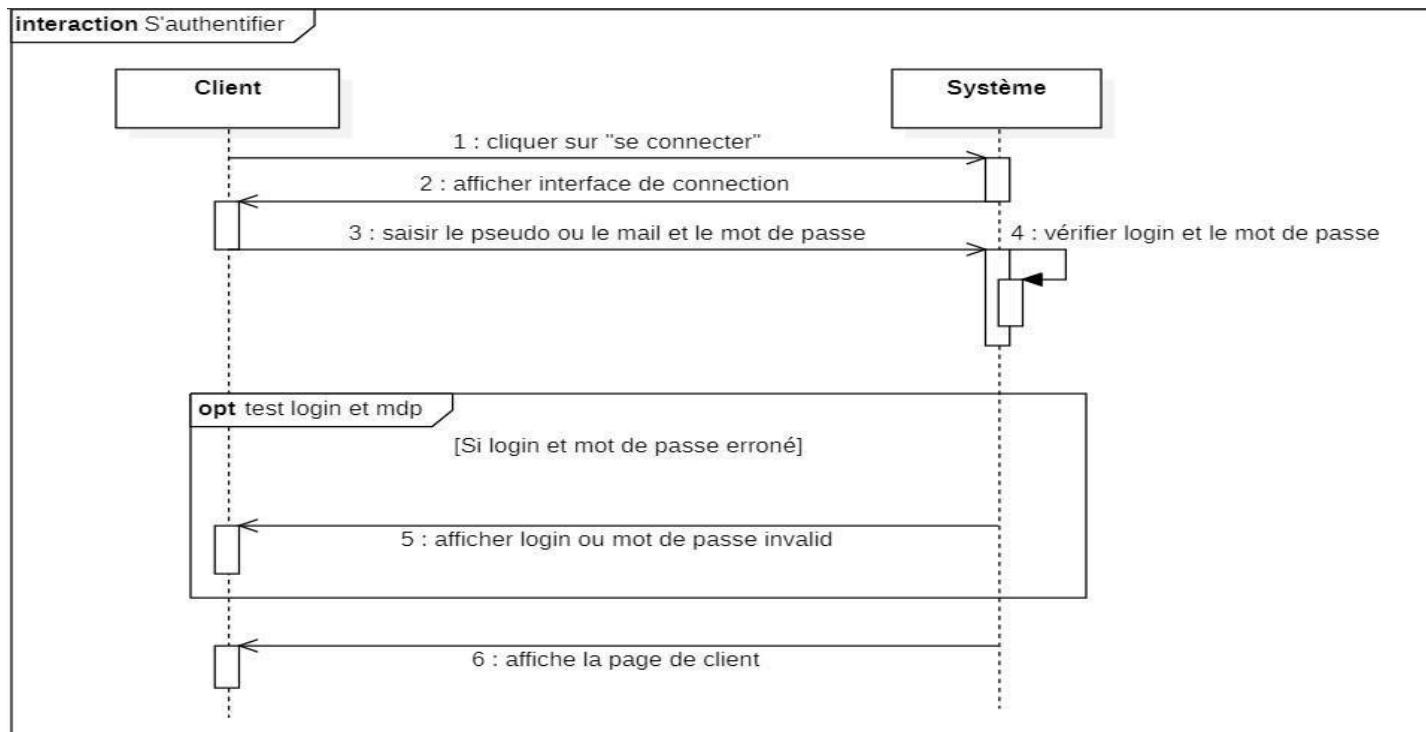


Figure 3: Diagramme de séquence de cas d'utilisation "S'authentifier"

Diagramme de séquence détaillé «S'authentifier»

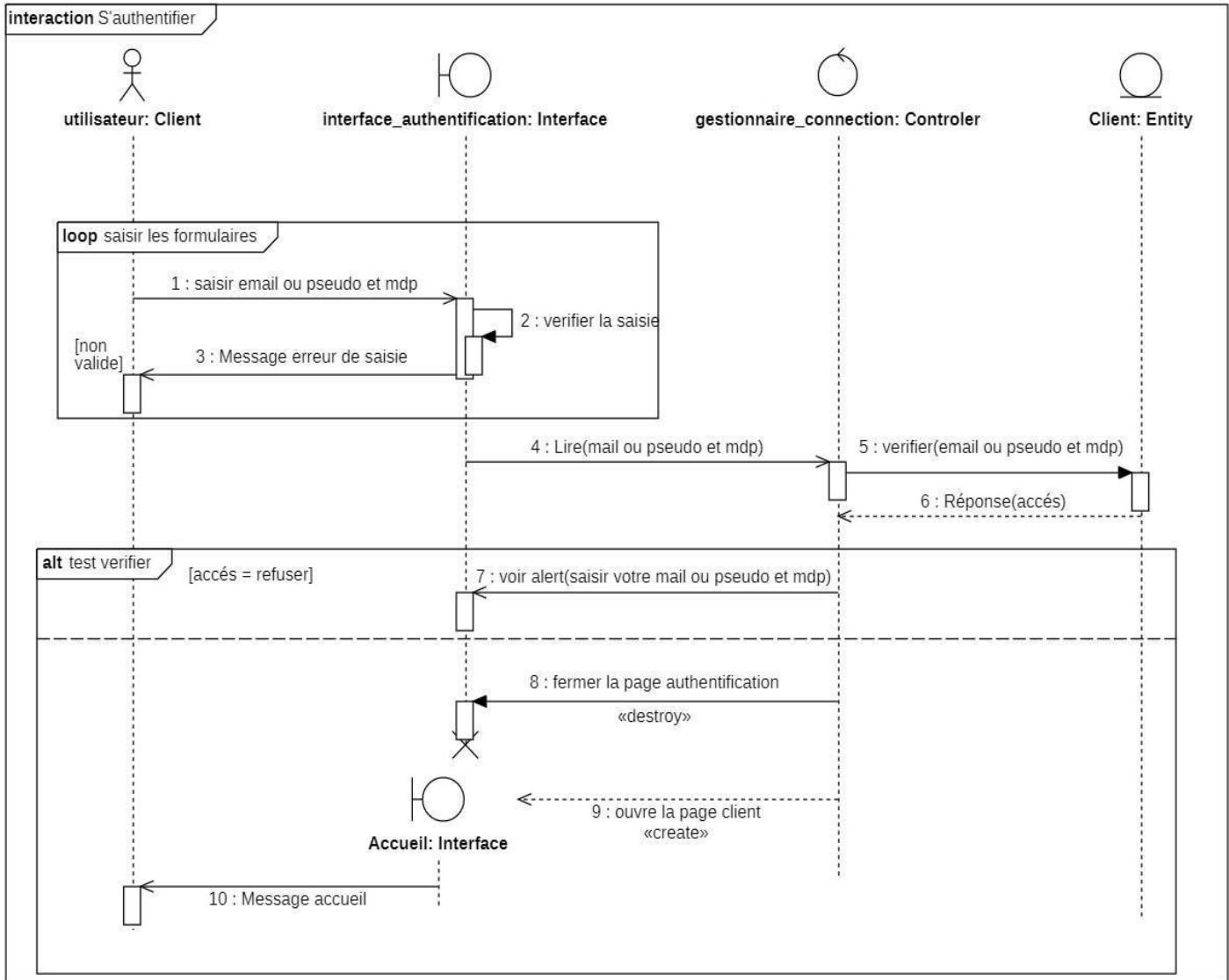


Figure 4: Diagramme de séquence détaillé "S'authentifier"

Description textuelle de cas d'utilisation : « S'inscrire »

Cas d'utilisation : S'inscrire
Acteurs : Client et Agent
Objectif : Il permet à l'acteur de remplir les formulaires .
Précondition : L'acteur doit être créé dans la base de données.
Postcondition : <ul style="list-style-type: none"> -Acteur inscrit. -La page d'authentification s'affiche.
Scénario nominal : <ol style="list-style-type: none"> 1. L'acteur ouvre l'application, 2. Le système affiche la page d'inscription, 3. L'acteur saisit tous les champs demandés, 4. Le système vérifie l'existence des données, 5. Le système affiche la page d'authentification.

Scénario alternatif :

A. Erreur d'inscription : pseudo ou email, mot de passe, etc.., non valide.

Cet enchaînement démarre au point 4.

5. Le système affiche un message d'erreur.

Le scénario reprend au point 2.

B. Champs obligatoires vides.

Cet enchaînement démarre au point 4.

Le scénario reprend au point 2.

Tableau 2: Description textuelle de cas d'utilisation " S'inscrire "

Diagramme de séquence système de cas d'utilisation « S'inscrire »

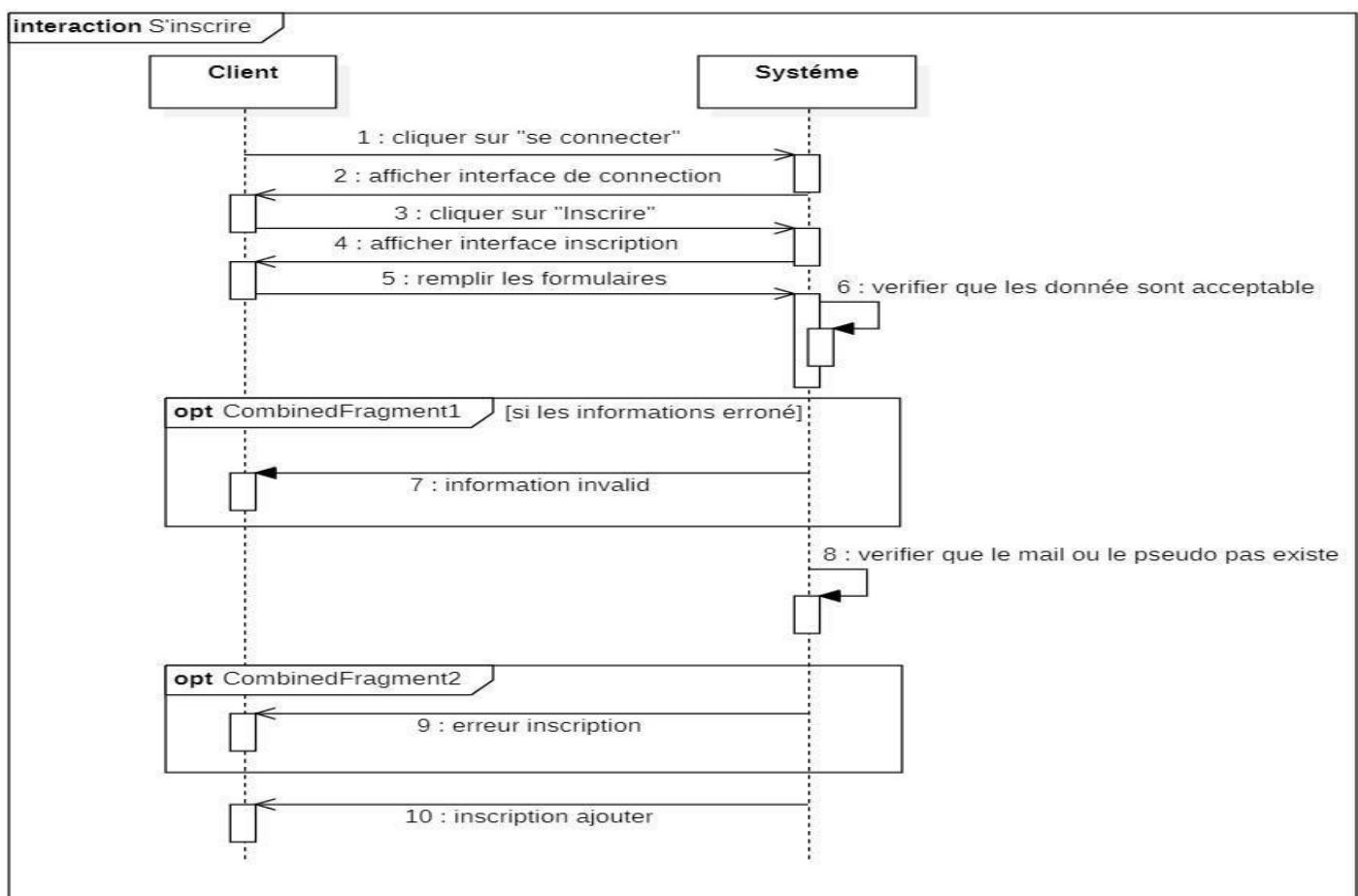


Figure 5: Diagramme de séquence de cas d'utilisation "S'inscrire"

Diagramme de séquence détaillé «S'inscrire»

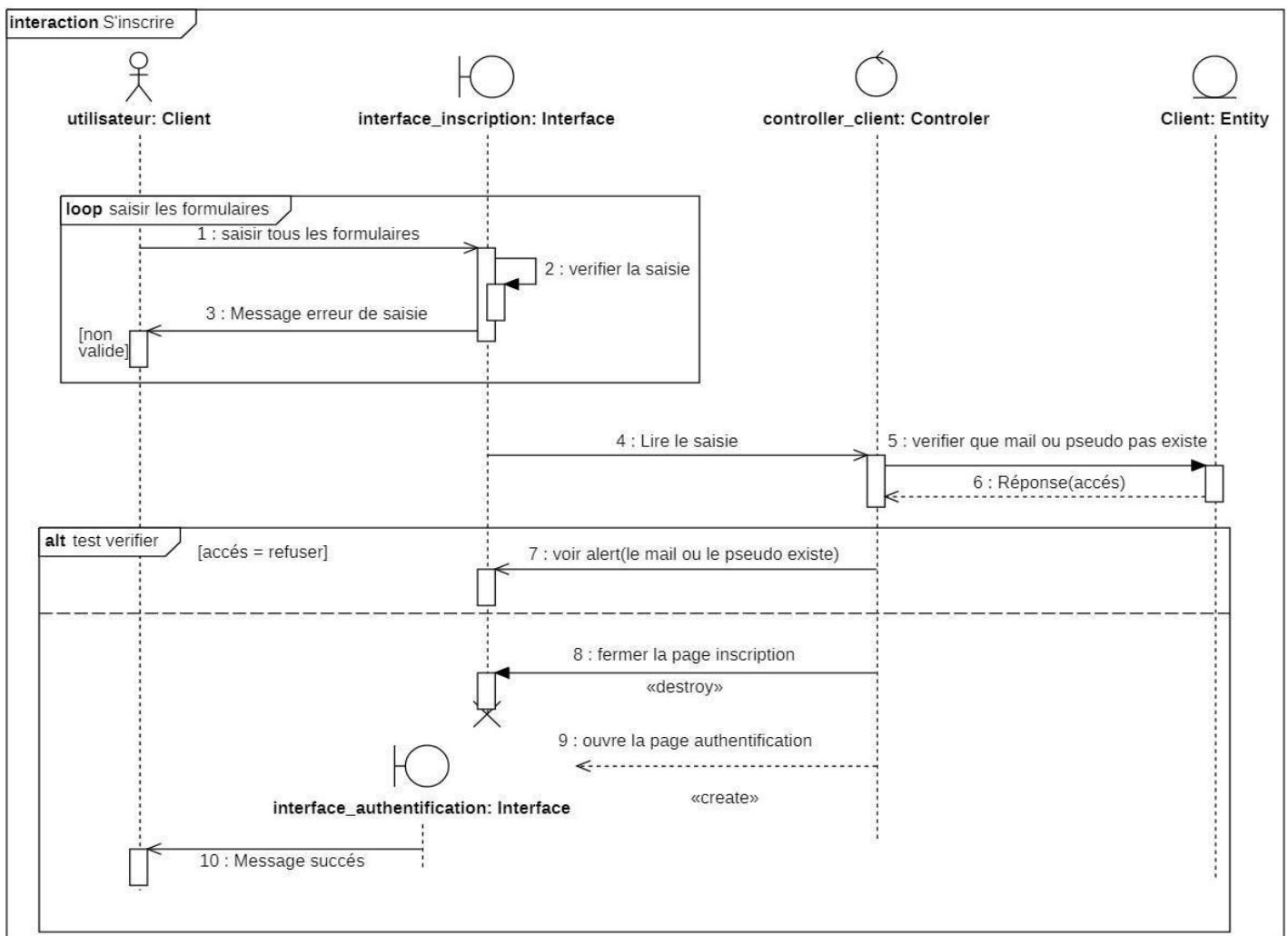


Figure 6: Diagramme de séquence détaillé "S'inscrire"

Description textuelle de cas d'utilisation : « Réservation »

Cas d'utilisation : Réservation

Acteurs : Client

Objectif : Il permet de réservée une véhicules.

Précondition :

- Succès d'authentification.
- Succès de consultation de la liste des véhicules.

Postcondition : Véhicule réservée

Scénario nominal :

1. Le client choisit de réservée une véhicule,
2. Le système affiche le détails de véhicule a réservée,
3. Le client donne le nombre de jour a choisi,
4. Le client clique sur le bouton louer,
5. Le système affiche l'étape de vérification le réservation.
6. Le client cliquer sur soumettre le réservation,
7. Le système affiche une alerte de confirmation (ok/annuler),
8. Le client cliquer sur ok,
9. Le système rendre le véhicule « Occupé » et enregistre dans la base de donnée.

Scénario alternatif :

- A. En cas d'annuler la réservation.
- Cet enchaînement démarre au point 7.

Tableau 3: Description textuelle de cas d'utilisation "Réservation"

Diagramme de séquence système de cas d'utilisation « Réservation »

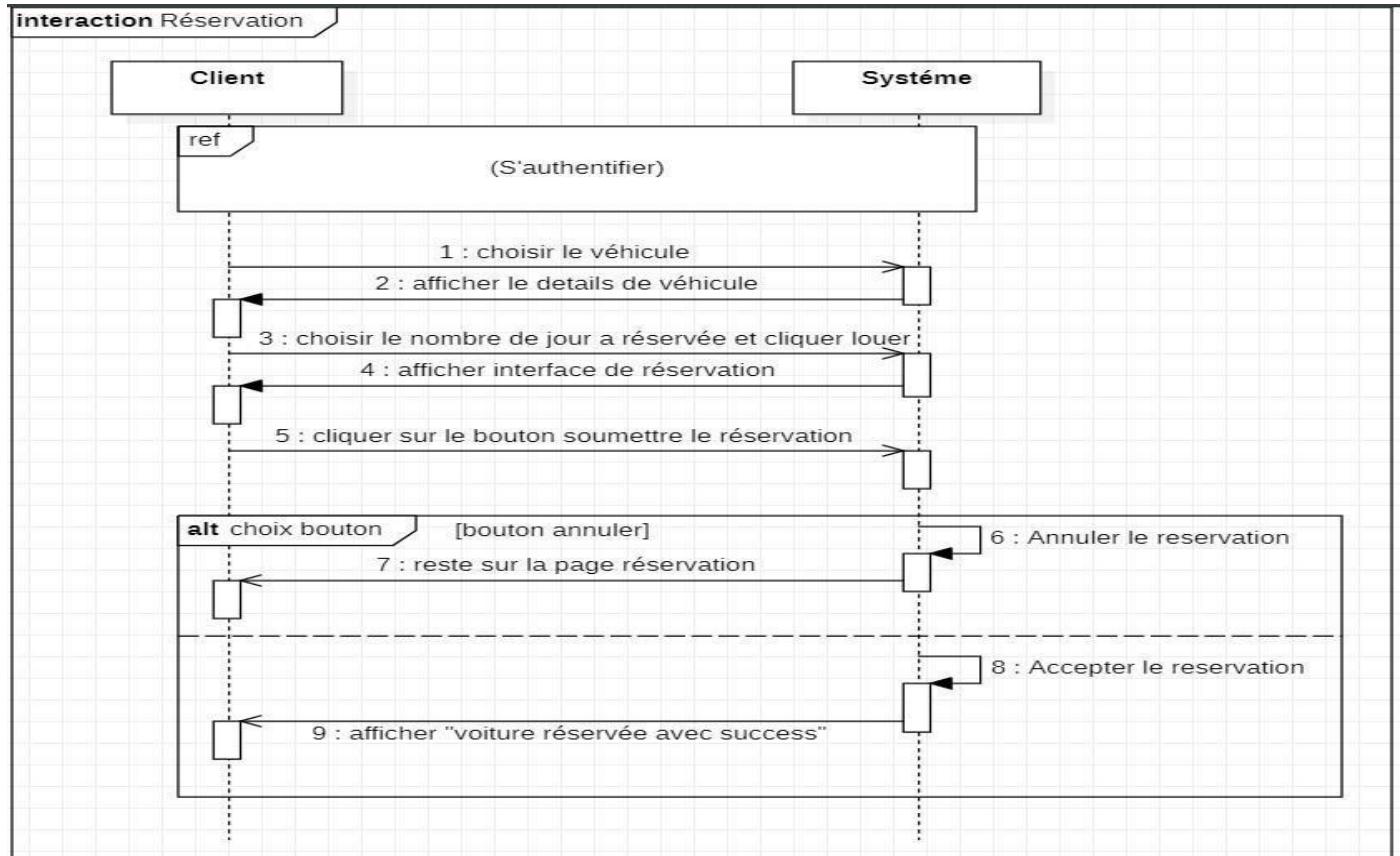


Figure 7: Diagramme de séquence de cas d'utilisation "Réservation"

Diagramme de séquence détaillé «Réservation»

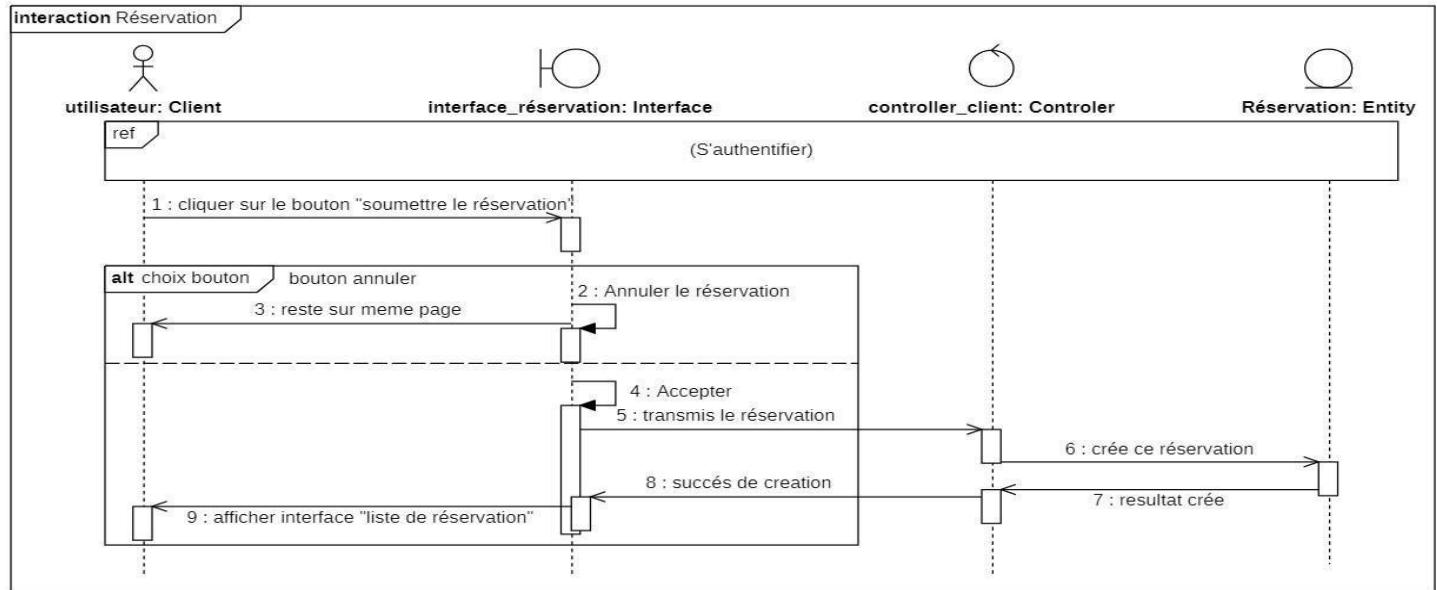


Figure 8: Diagramme de séquence détaillé "Réservatio

Description textuelle de cas d'utilisation : « Déposer nouveau voiture»

Cas d'utilisation : Déposer nouveau voiture
Acteurs : Agent
Objectif : Il permet au agent d'ajouter un véhicule.
Précondition :
- Succès d'authentification. - Succès de consultation de la liste des véhicules.
Postcondition : Véhicule ajouté.
Scénario nominal :
1. L'agent choisit l'ajout d'un nouveau véhicule, 2. Le système affiche le formulaire à remplir, 3. L'agent saisit les informations à remplir sur le nouveau véhicule, 4. Le système vérifie les données, 5. Le système enregistre le véhicule dans la base de données.
Scénario alternatif :
A. Champs obligatoires non valides ou vides. Cet enchaînement démarre au point 4. 5. Le système affiche un message d'erreur. Le scénario reprend au point 2.

Tableau 4: Description textuelle de cas d'utilisation "Déposer nouveau voiture"

Diagramme de séquence système de cas d'utilisation « Déposer nouveau voiture »

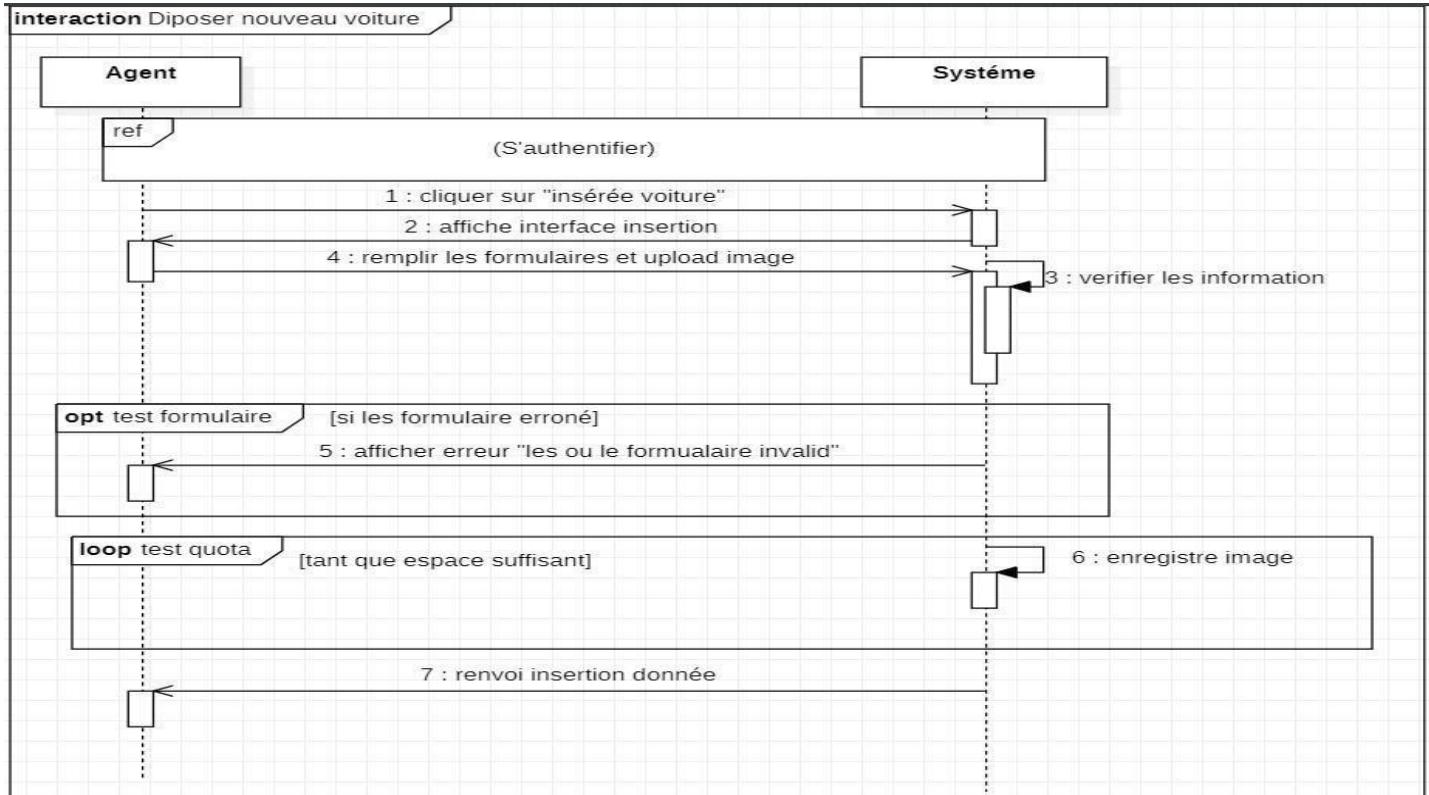


Figure 9: Diagramme de séquence de cas d'utilisation "Déposer nouveau voiture"

Diagramme de séquence détaillé «Déposer nouveau voiture »

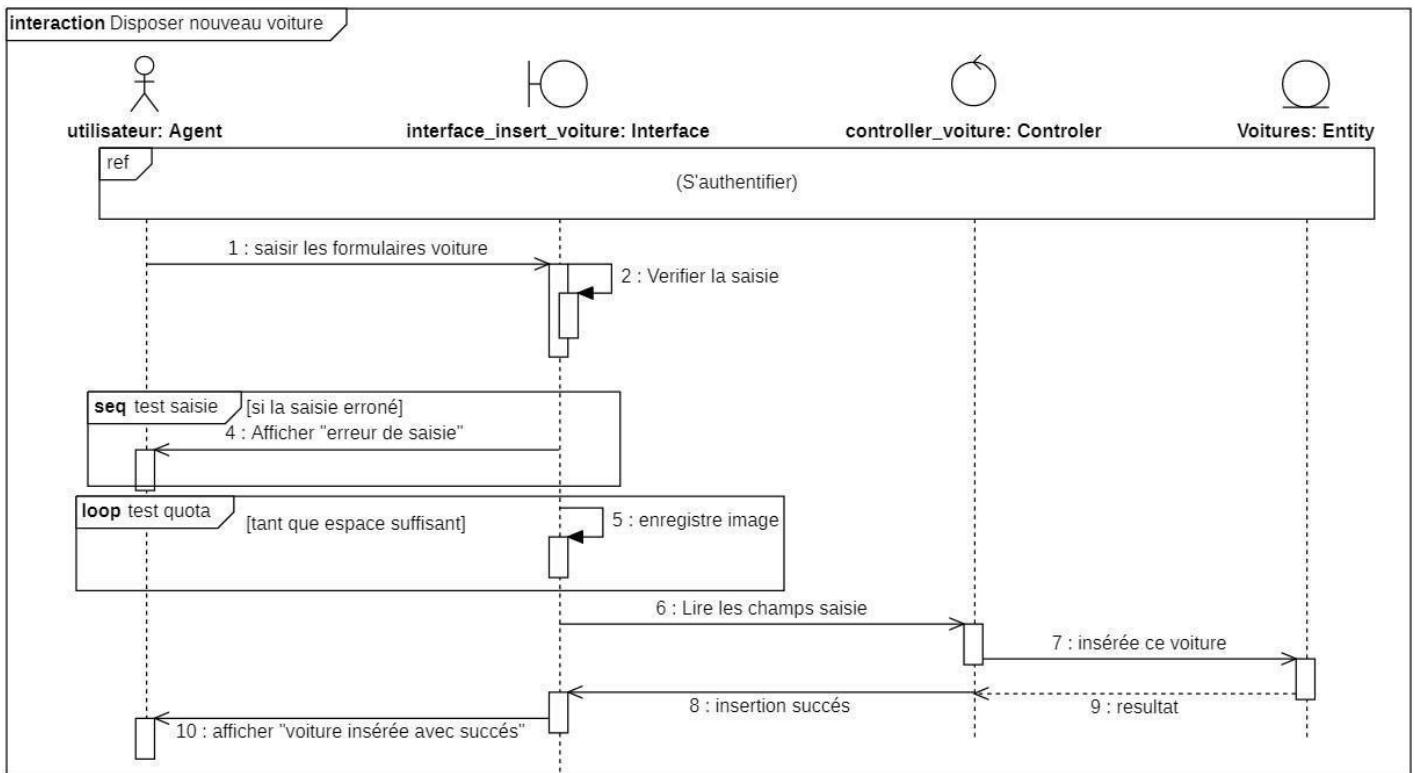


Figure 10: Diagramme de séquence détaillé "Déposer nouveau voiture"

Description textuelle de cas d'utilisation : « Modifier voiture»

Cas d'utilisation : Modifier voiture
Acteurs : Agent
Objectif : Il permet au agent de modifier un véhicule.
Précondition :
-Succès d'authentification. -Succès de consultation de la liste des véhicules.
Postcondition : Véhicule modifié.
Scénario nominal :
1. L'agent choisit d'affiche la « Liste des véhicules », 2. Le système affiche la liste, 3. L'agent choisit la modification d'un véhicule, 4. Le système affiche le formulaire de modification, 5. L'agent modifier les informations de véhicule, 6. Le système demande la validation de modification, 7. L'agent valide la modification, 8. Le système vérifie les données, 9. Le système enregistre la modification dans la base de données.
Scénario alternatif :
A. Champs obligatoires non valides ou vides. Cet enchaînement démarre au point 8. 9. Le système affiche un message d'erreur. Le scénario reprend au point 5.

Tableau 5: Description textuelle de cas d'utilisation "Modifier voiture "

Diagramme de séquence système de cas d'utilisation « Modifier voiture »

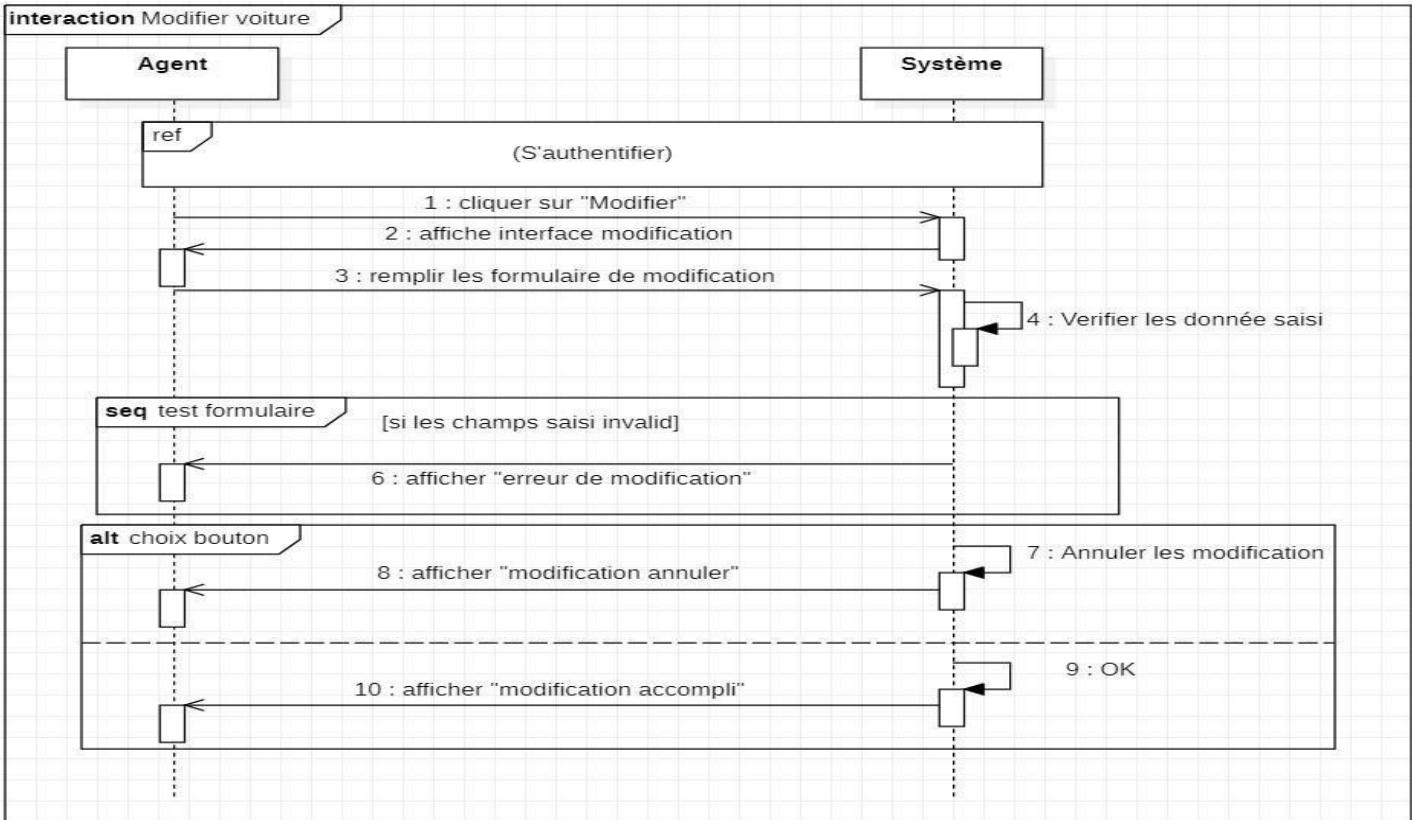


Figure 11: Diagramme de séquence de cas d'utilisation "Modifier voiture"

Diagramme de séquence détaillé «Modifier voiture »

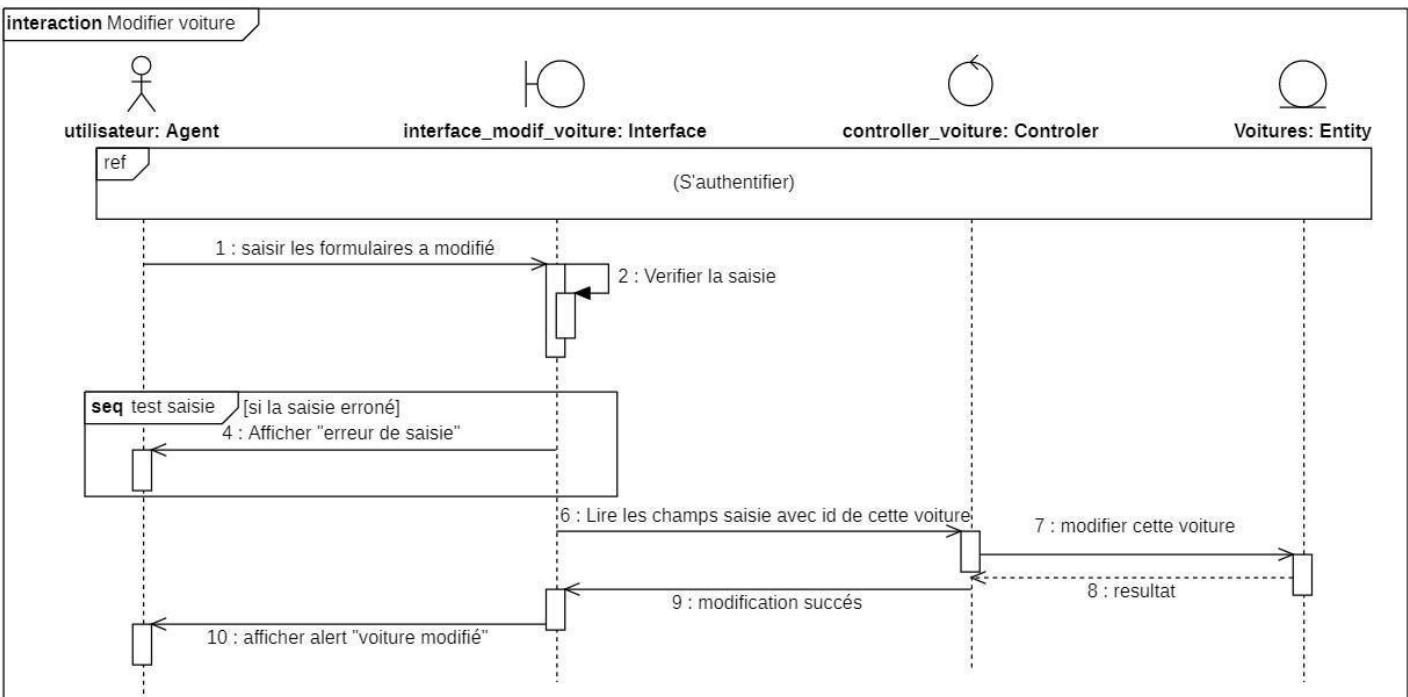


Figure 12: Diagramme de séquence détaillé "Modifier voiture"

Description textuelle de cas d'utilisation : « Supprimer voiture»

Cas d'utilisation : Supprimer voiture
Acteurs : Agent
Objectif : Il permet au agent de supprimer un véhicule.
Précondition :
-Succès d'authentification. -Succès de consultation de la liste des véhicules.
Postcondition : Véhicule supprimé.
Scénario nominal :
1. L'agent choisit d'afficher la « Liste des véhicules », 2. Le système affiche la liste, 3. Un agent choisit la suppression d'un véhicule, 4. Le système demande la validation de la suppression, 5. L'agent valide la suppression, 6. Le système procède à la suppression du véhicule de la base de données.
Scénario alternatif :
A. l'agent annule la suppression. Cet enchaînement démarre au point 4. 9. Le système affiche une notification. Le scénario reprend au point 2.

Tableau 6: Description textuelle de cas d'utilisation " Supprimer voiture"

Diagramme de séquence système de cas d'utilisation « Supprimer voiture »

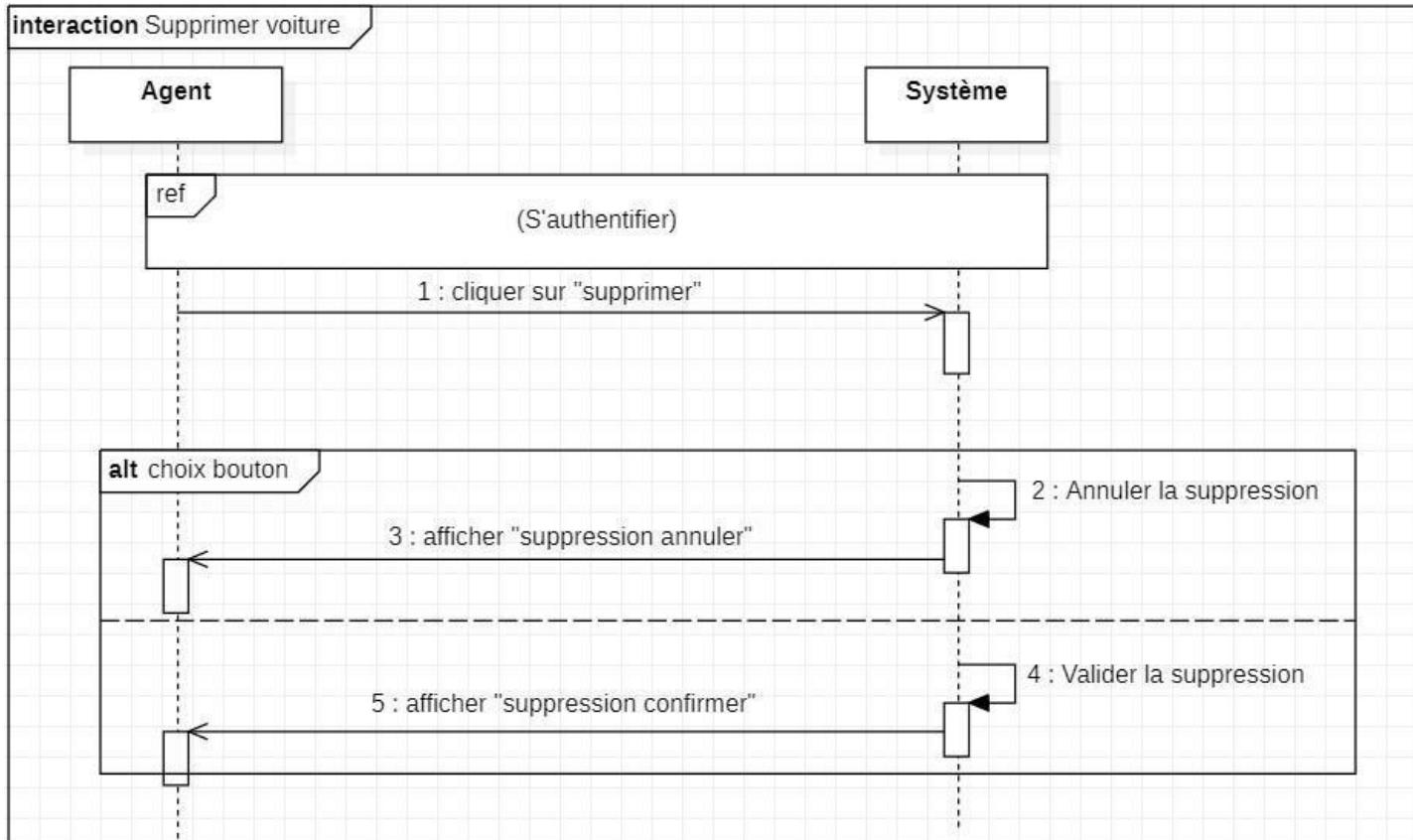


Figure 13: Diagramme de séquence de cas d'utilisation "Supprimer voiture"

Diagramme de séquence détaillé «Supprimer voiture »

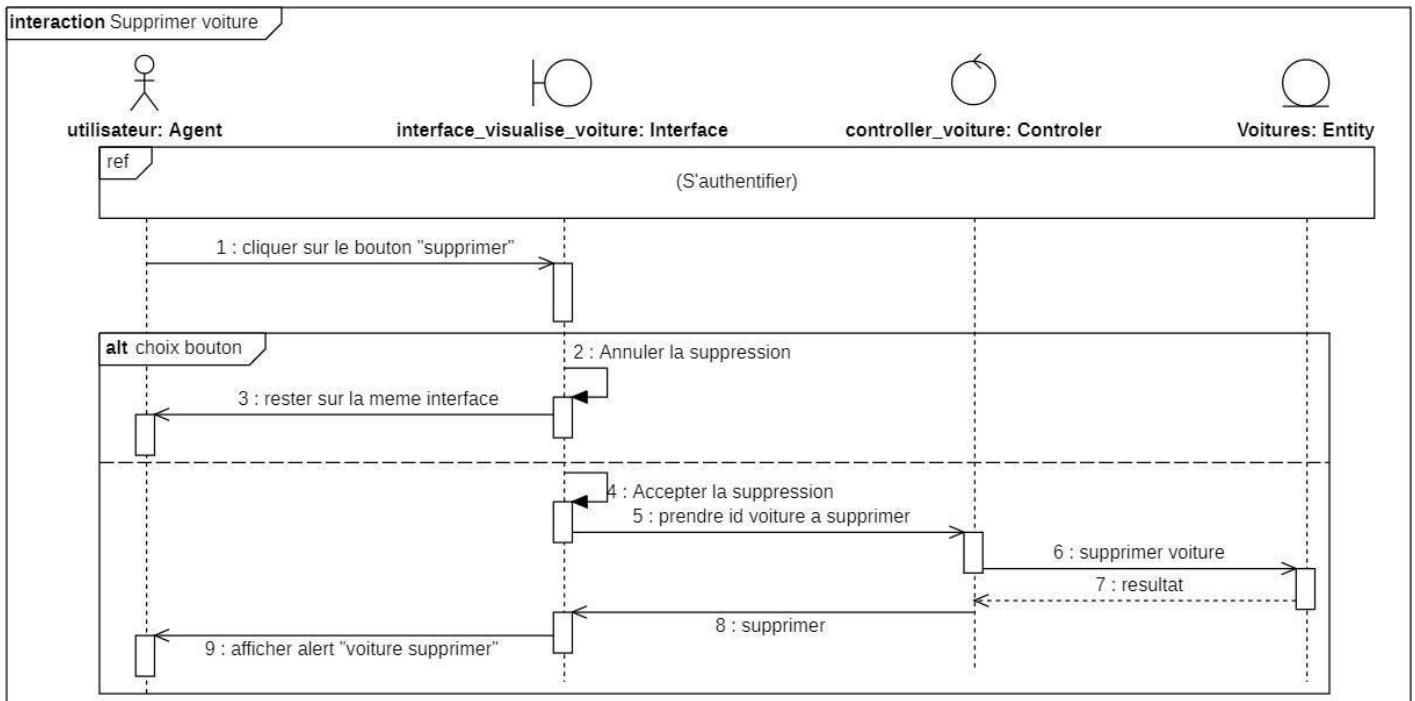


Figure 14: Diagramme de séquence détaillé "Supprimer voiture"

Description textuelle de cas d'utilisation : « Accepter la location»

Cas d'utilisation : Accepter la location
Acteurs : Agent
Objectif : Il permet au agent d'accepter la réservation.
Précondition : Succès d'authentification. Le client réservé cette véhicule et changer l'occupation (Libre par Occupée).
Postcondition : Le véhicule a été accepté.
Scénario nominal : 1. L'agent clique sur le bouton accepter, 2. Le système affiche une confirmation (Vous avez accepté cette réservation), 3. Le système garde le statut (Occupée) de cette véhicule selon la date de retour dans la base de données, 4. Le système affiche une alerte d'acceptation
Scénario alternatif : Aucun.

Tableau 7: Description textuelle de cas d'utilisation "Accepter la location "

Diagramme de séquence système de cas d'utilisation « Accepter la location »

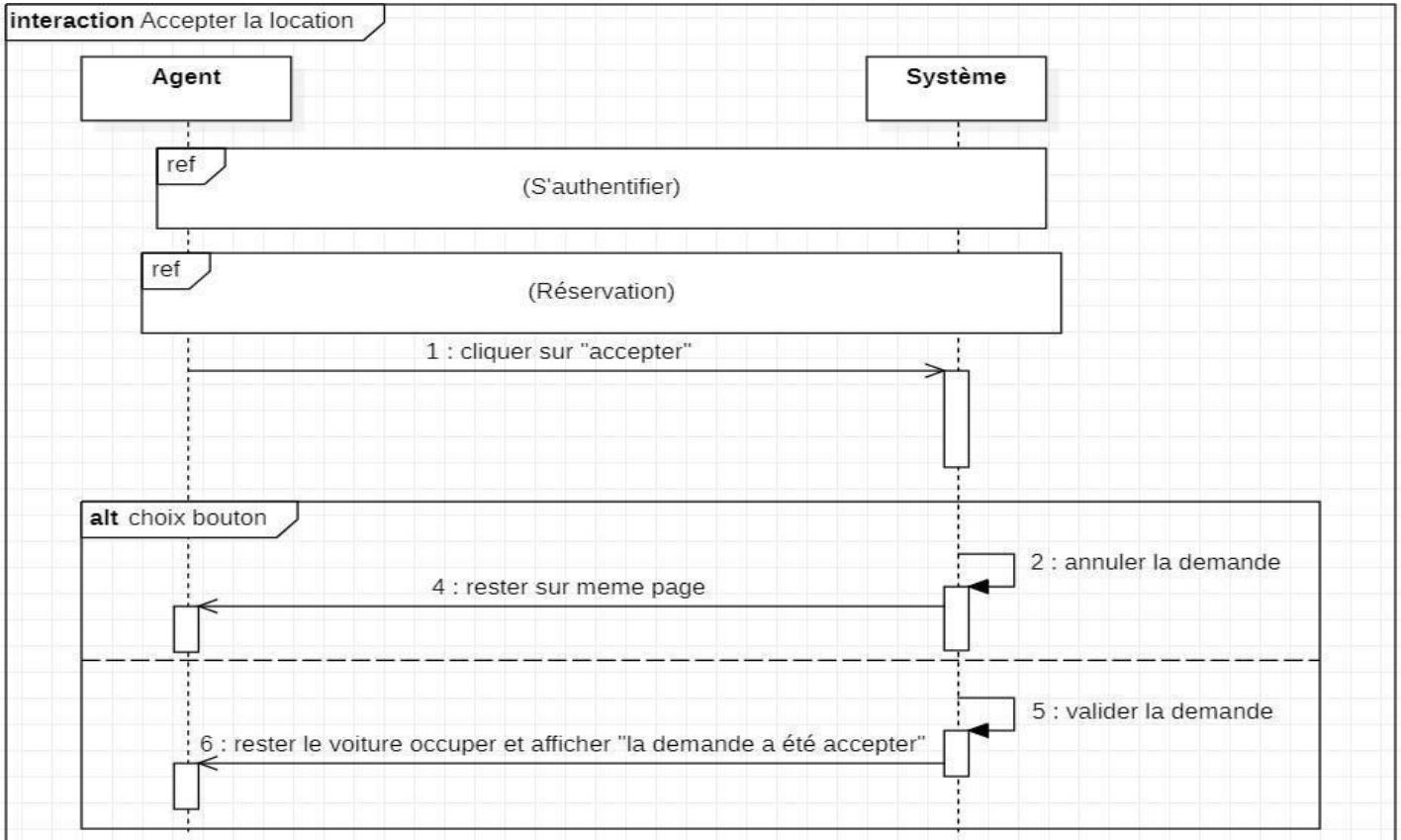


Figure 15: Diagramme de séquence de cas d'utilisation "Accepter la location"

Diagramme de séquence détaillé «Accepter la location»

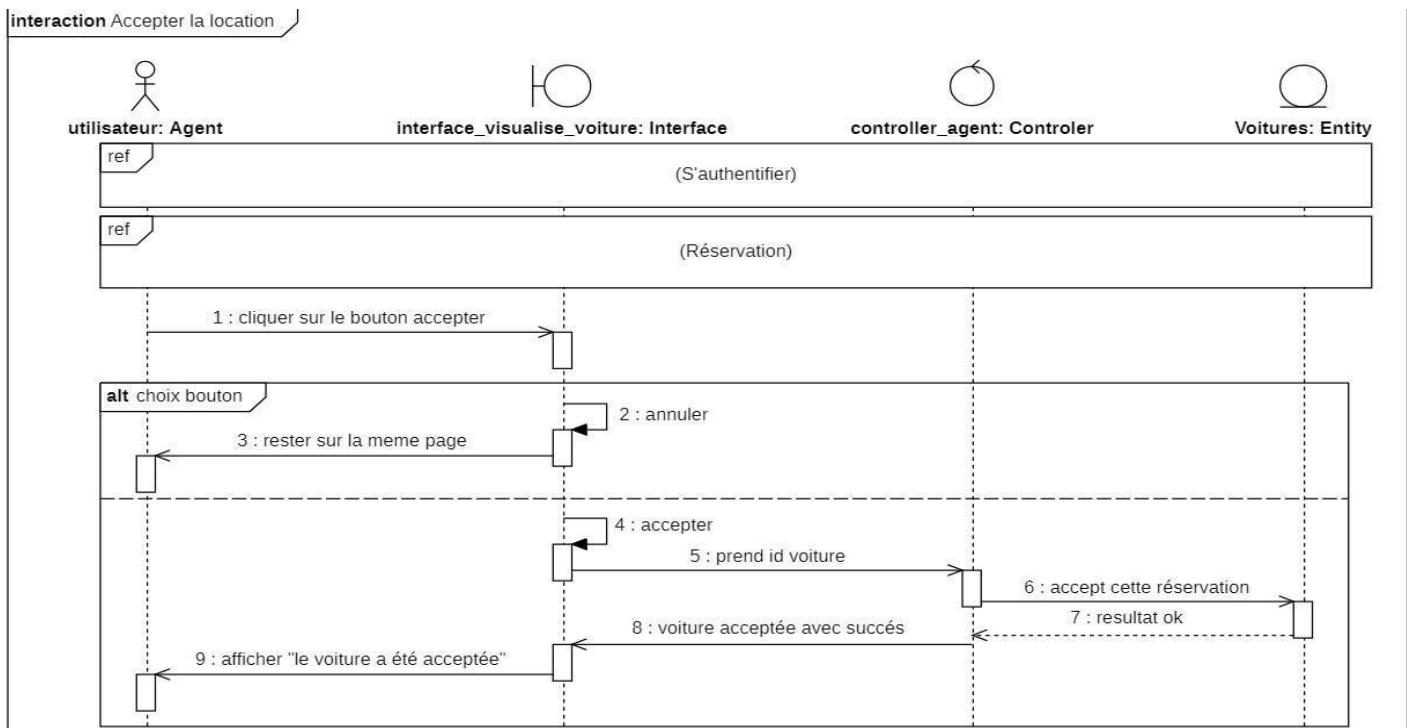


Figure 16: Diagramme de séquence détaillé "Accepter la location"

Description textuelle de cas d'utilisation : « Refuser la location»

Cas d'utilisation : Refuser la location
Acteurs : Agent
Objectif : Il permet au agent de refuser la réservation.
Précondition :
- Succès d'authentification. - Le client a réservé cette voiture et change l'occupation (Libre par Occupée).
Postcondition : La voiture a été refusée.
Scénario nominal :
1. L'agent clique sur le bouton refuser, 2. Le système affiche une confirmation (Vous avez refusé cette réservation), 3. Le système change le statut (Occupée par Libre), 4. Le système affiche une alerte de refus
Scénario alternatif : Aucun.

Tableau 8: Description textuelle de cas d'utilisation "Refuser la location"

Diagramme de séquence système de cas d'utilisation « Refuser la location »

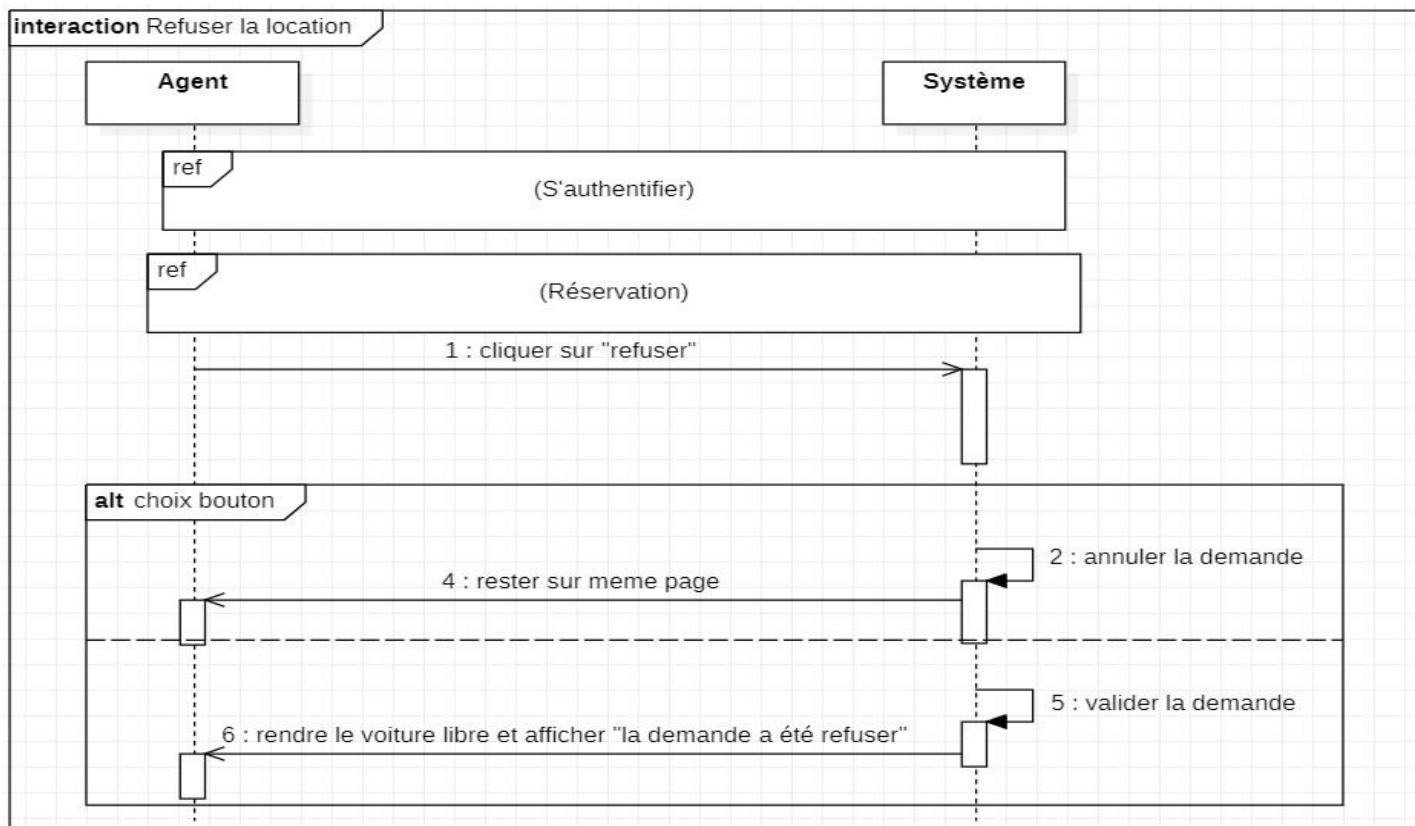


Figure 17: Diagramme de séquence de cas d'utilisation "Refuser la location"

Diagramme de séquence détaillé «Refuser la location»

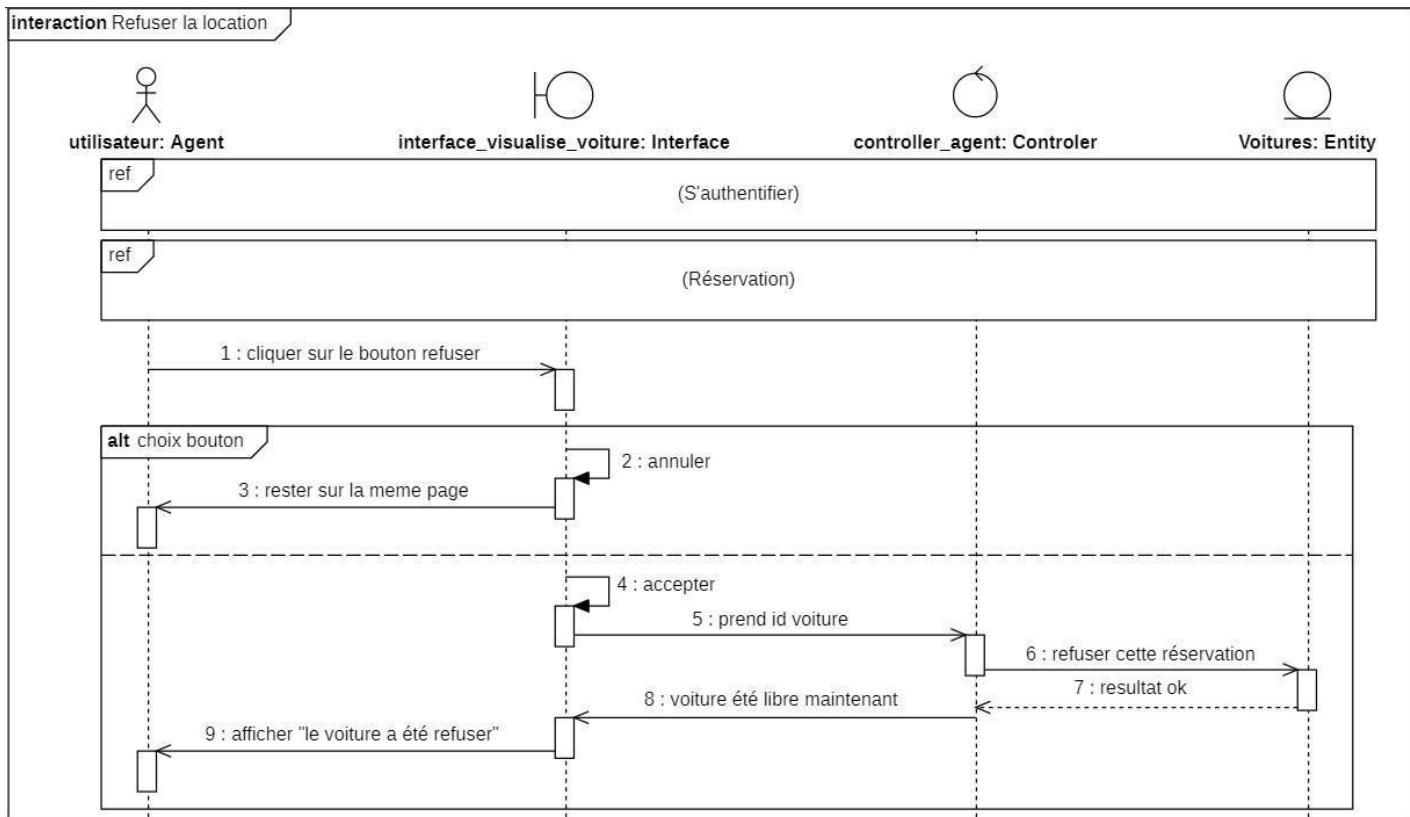


Figure 18: Diagramme de séquence détaillé "Refuser la location"

Description textuelle de cas d'utilisation : « Mise à jour le voiture réservée »

Cas d'utilisation : Mise à jour le voiture réservée
Acteurs : Agent
Objectif : Il permet au agent de mettre à jour la date de retour par rapport au date de départ d'une véhicule.
Précondition : - Succès d'authentification. - Le client a réservé cette véhicule et change l'occupation (Libre par Occupée) pendant une durée de nombre de jour.
Postcondition : Le véhicule a été mis à jour.
Scénario nominal : 1. agent cliquer sur le bouton mise à jour, 2. Le système doit vérifier la période de réservation 3. Le système doit afficher une alerte (Votre voiture a été libre), 4. Le système changer le statut (Occupée par Libre) dans la base de données,
Scénario alternatif : A. Si la période de retour est supérieur à la date actuelle : 1. Le système affiche une alerte contenant le nombre de jour restant Le scénario reprend au point 2.

Tableau 9: Description textuelle de cas d'utilisation "Mise à jour le voiture réservée "

Diagramme de séquence système de cas d'utilisation «Mise a jour voiture»

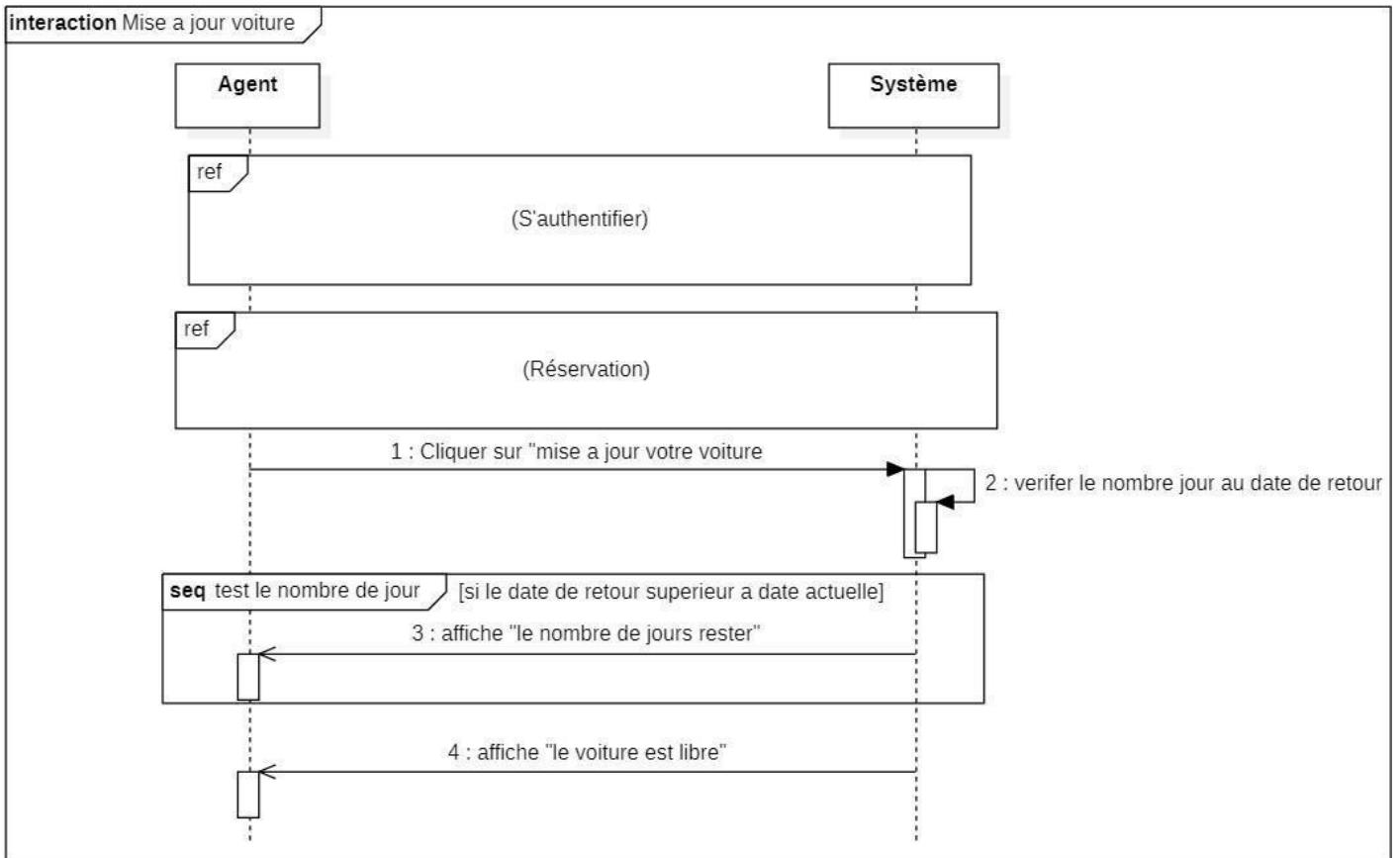


Figure 19: Diagramme de séquence de cas d'utilisation "Mise a jour voiture"

Diagramme de séquence détaillé «Mise a jour voiture»

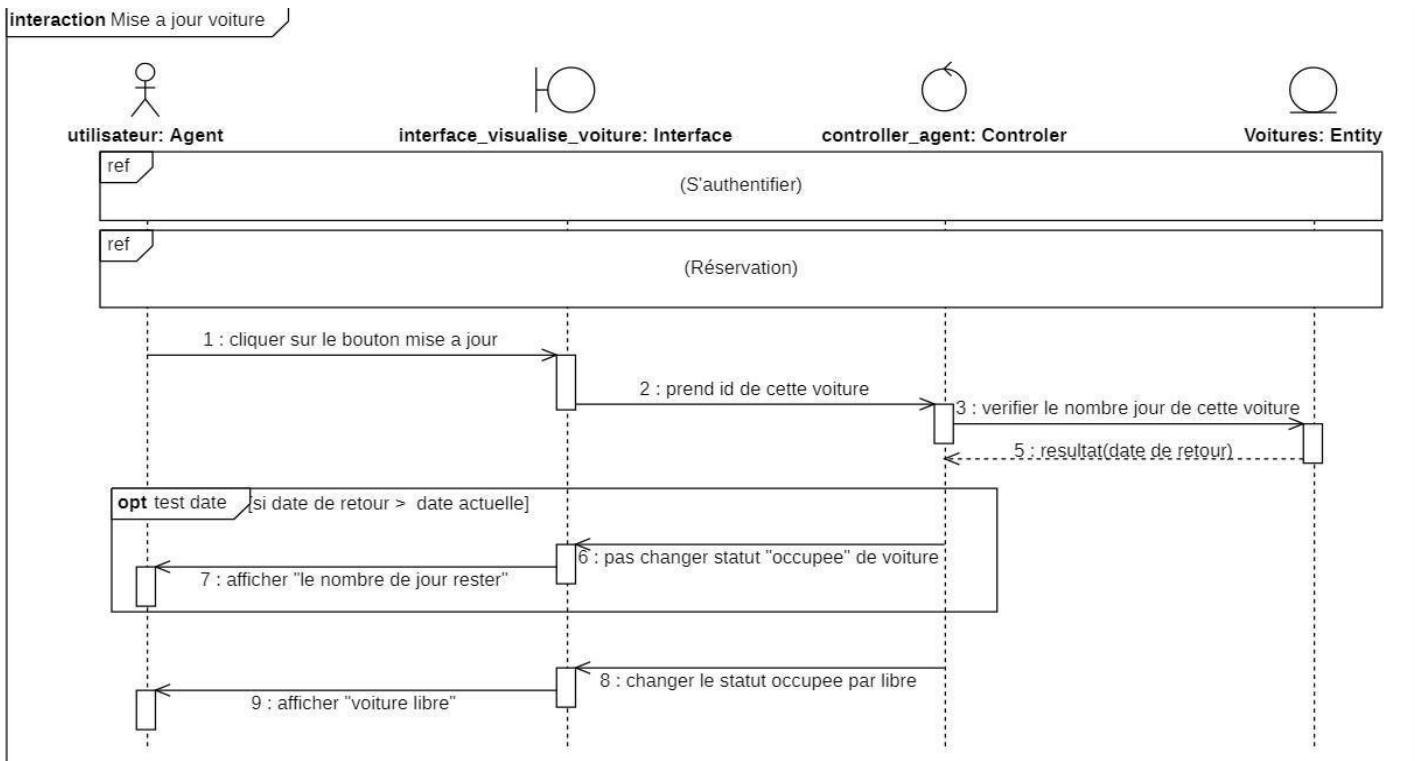


Figure 20: Diagramme de séquence détaillé "Mise a jour voiture"

Description textuelle de cas d'utilisation : « Visualiser le client réservée»

Cas d'utilisation : Visualise le client réservée
Acteurs : Agent
Objectif : Il permet au agent de visualiser la réservation d'un client.
Précondition : - Succès d'authentification. - Le client réservé cette véhicule et changer l'occupation (Libre par Occupée).
Postcondition : Le détails de client.
Scénario nominal : 1. L'agent cliquer sur le bouton détails client, 2. Le système affiche une confirmation (Vous avez refuser cette réservation), 3. Le système changer le statut (Occupée par Libre), 4. Le système affiche une alerte de refuse
Scénario alternatif : Aucun.

Tableau 11: Description textuelle de cas d'utilisation "Visualise le client réservée"

Diagramme de séquence système de cas d'utilisation «Visualise le client réservée»

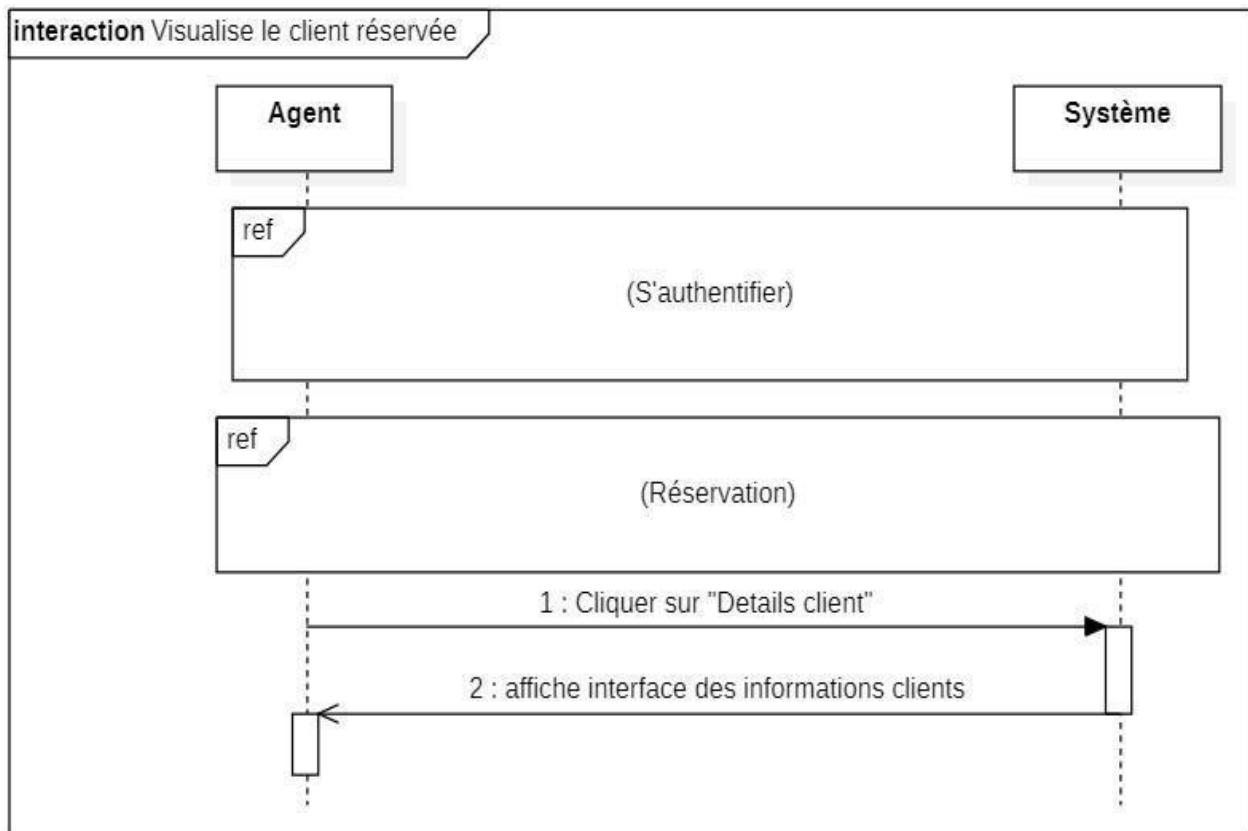


Figure 21: Diagramme de séquence de cas d'utilisation "Visualise le client réservée"

Diagramme de séquence détaillé «Visualise le client réservée»

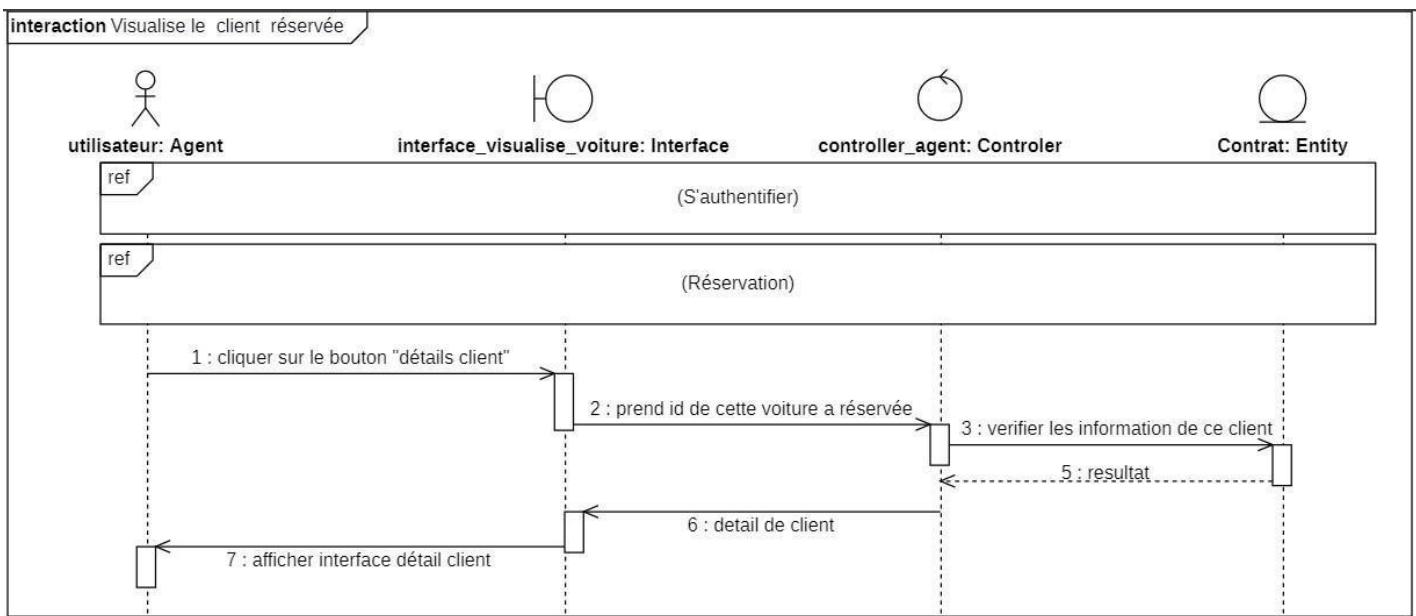


Figure 22: Diagramme de séquence détaillé "Visualise le client réservée"

Conclusion

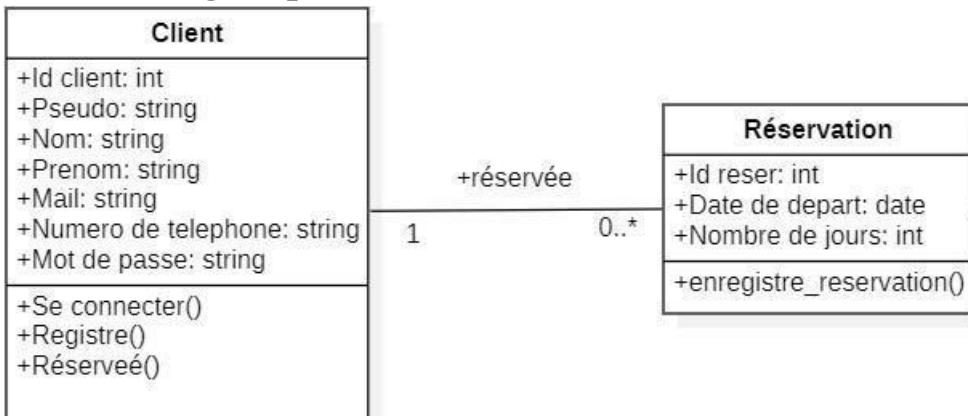
Dans ce partie, nous avons modélisé le fonctionnement de l'application afin d'avoir une vue globale et simplifiée du système. Nous avons aussi détaillé les différents modules de l'application ce qui nous a permis d'organiser le travail et d'avoir une idée claire sur le travail à réaliser. Ce travail est décrit plus précisément dans le partie qui suit.

Partie III : Base de données :

III.1 Les règles de passage du diagramme de classe au schéma relationnel de la base de données.

Dans notre projet, nous avons adapté les règles suivantes pour faire le passage du diagramme de classe vers le modèle relationnel.

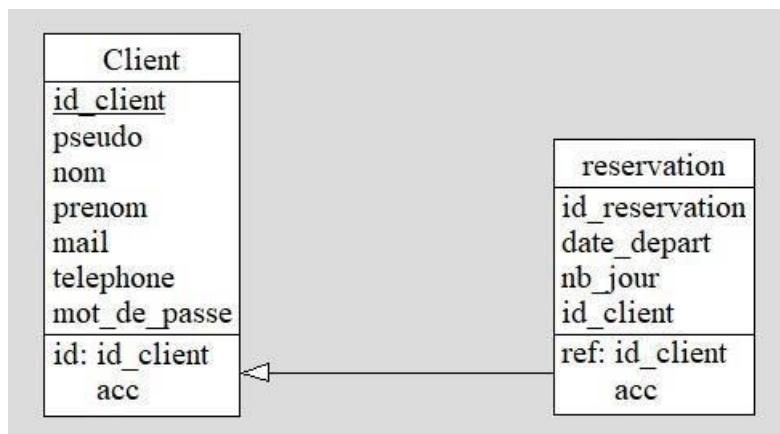
- o Règle 1: présence de la cardinalité (?..1) d'un côté de l'association



Chaque classe se transforme en une table.

Chaque attribut de classe se transforme en un champs de table.

L'identifiant de la classe qui est associée à la cardinalité (?..1) (ex: Client) devient le clé étrangère de l'autre classe (ex: Réservation).



Contrainte d'intégrité référentielle:

Clé Etrangère \subseteq Clé Primaire.

Ex: reservation.id_client \subseteq Client.id_client

Exemple :

Client :

id_client	pseudo	nom	prénom	mai l	telephone	Mot_de_passe
1	AymenSam	Samoudi	Aymen	Aymen.samoudi@esen.tn	54330607	Hash1(MD5)
2	fayezSlimi	Slimi	Fayez	Slimi.fayez@gmail.com	54330607	Hash2(MD5)

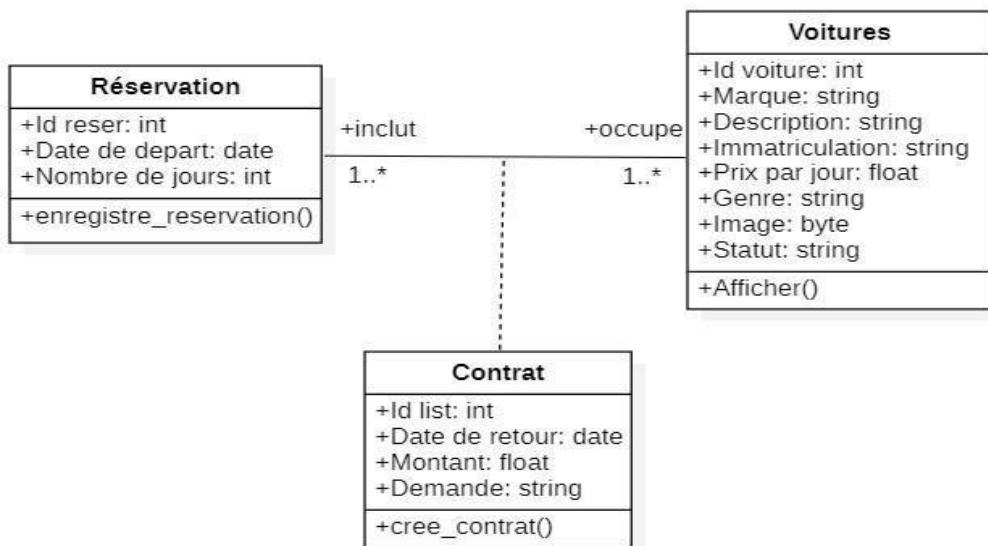
Réservation :

id_reservation	Date_depart	nb_jour	id_client
5	16/05/2020	3	2
9	20/05/2020	5	1
10	20/05/2020	2	3 !!!

- Quel est le client(nom) de réservation dont id_reservation est 5 ?
Slimi
- Quels sont les réservations (id_reservation) du client dont id_client est 1 ?
9
- Quel est le nom de réservation dont id_reservation est 10 ?
 pas de sens !!!

Contrainte d'intégrité référentielle :
 $\text{reservation.id_client} \subseteq \text{client.id_client}$

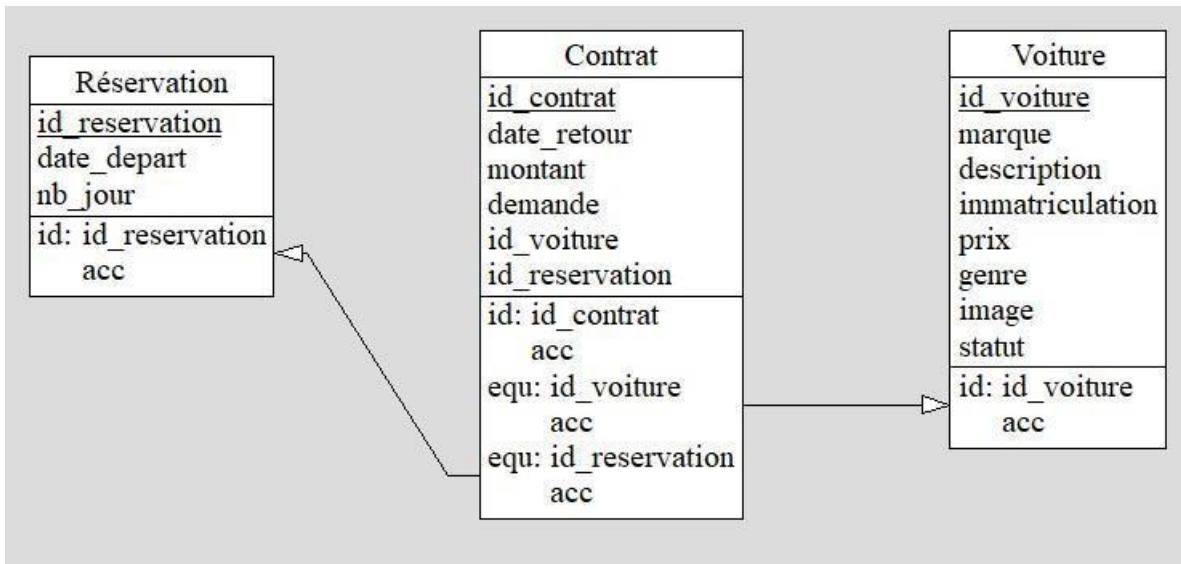
- o **Règle 2: présence de (..N) des deux côtés de l'association :**



Chaque classe se transforme en une table.

Chaque attribut de classe se transforme en un champs de table.

L'association se transforme en une table. Cette table a comme champs l'identifiant de chacune des deux classes, plus d'éventuels autres attributs.



Contrat.id_reservation \subseteq Réservation.id_reservation

Contrat.id_voiture \subseteq Voiture.id_voiture

Exemple :

Réservation

id_reservation	date_depart	nb_jour
1	22-05-2020	3
2	25-05-2020	2
3	23-05-2020	6
4	26-05-2020	4

id_voiture	marque	description	immatriculation	prix	genre	image	statut
1	Ford	Familial	180 Tun 5410	60	Economique	(chemin relative)	Occupée
2	Golf 6	Sport	208 Tun 4211	80	Intermédiaire	(chemin relative)	Occupée
3	Symbol	Familial	200 Tun 4541	61	Economique	(chemin relative)	Occupée

Contrat

id_reservation	id_voiture	id_contrat	date_retour	montant	demande
1	1	1	25-05-2020	180	Accepter
2	1	2	27-05-2020	120	Refuser
3	2	3	29-05-2020	366	Accepter
5 !!!	3	4	22-05-2020 !	100 !	Accepter !
4	4!!!	5	30-05-2020	200 !	Accepter !

Contraintes d'intégrité référentielle :

Contrat.id_reservation \subseteq Réservation.id_reservation

Contrat.id_voiture \subseteq Voiture.id_voiture

Qui a réservée le voiture « Ford » ?

Quels sont les voitures (marque) qui sont réservée 2 jours ?

- Qui a réservée le voiture « Symbol » ? pas de sens .
- Quels sont le voiture (marque) qui sont réservée dont id_reservation 4 ?
pas de sens.

III.2 Le schéma relationnel de la base de données.

La figure ci-dessous récapitule les règles précédent dans un schéma relationnel qui contient toutes les informations telles que les tabes, les champs:

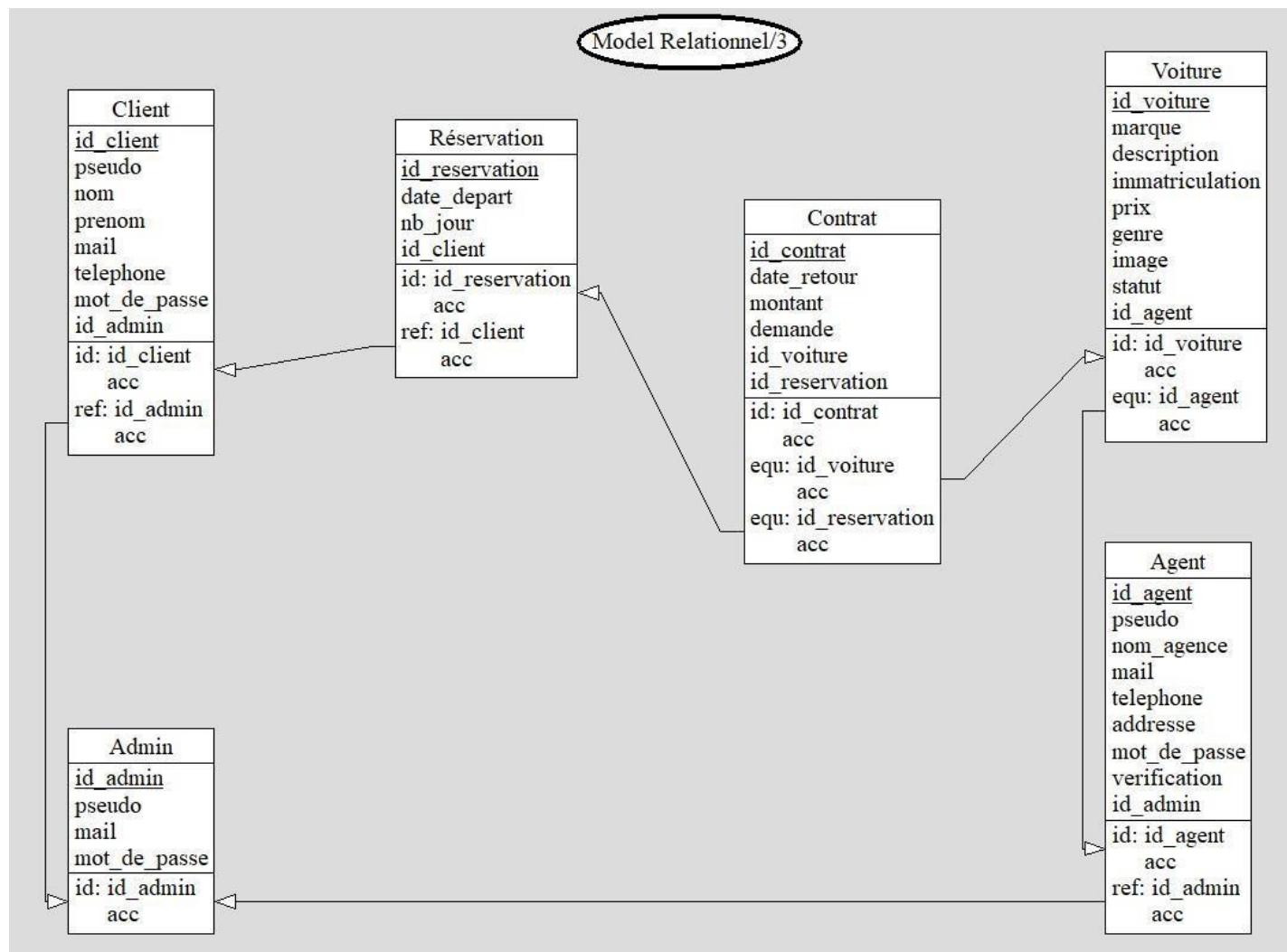


Figure 23: schéma relationnel de la base de données

Modèle logique des données :

Dans ce qui suit, nous présentons le modèle de données

Admin(id_admin, pseudo, mail, mot_de_passe) ;

Client(id_client, pseudo, nom, prénom, mail, téléphone, mot_de_passe, #id_admin) ;

Agent(id_agent, pseudo, nom_agence, mail, telephone, adresse, mot_de_passe, vérification, #id_admin) ;

Réservation(id_reservation, date_depart, nb_jour, #id_client) ;

Voiture(id_voiture, marque, description, immatriculation, prix, genre, image, statut, #id_agent) ;

Contrat(id_contrat, #id_reservation, #id_voiture, date_retour, montant, demande) ;

III.3 Normalisation du schéma relationnel

(Boyce-Codd) Qu'est-ce qu'une BD relationnelle

'incorrecte' ?

Une relation n'est pas correcte si :

- elle implique des répétitions au niveau de sa population
- elle pose des problèmes lors des maj (insertions, modifications et suppressions)

Les conditions pour qu'une relation soit correcte peuvent être définies formellement :

=> règles de normalisation.

Exemple :

Contrat(id_contrat, date_retour, montant, demande, marque, nb_jour, #id_reservation, #id_voiture) ;

La marque du véhicule ne dépend que du véhicule et pas de la réservation.

Le nombre de jour ne dépend que de la réservation et pas du véhicule

□ **REDONDANCES**

□ **Anomalies de mise à jour**

Cette relation n'est pas correcte. Il faut la normaliser.

Normalisation d'un schéma logique

Processus de transformation d'un schéma S1 pour obtenir un schéma S2 :

qui est équivalent (même contenu)

dont les maj assurant la cohérence de la base de données sont simples

maj simple :

un changement élémentaire dans le monde réel se traduit par une mise à jour
d'un tuple.

Exemples de changements élémentaires

- **Contrat**(id_contrat, date_retour, montant, demande, marque, nb_jour, #id_reservation, #id_voiture) ;
- La montant totale pour une réservation et un véhicule est mise à jour => 1 tuple à m.a.j.
- Une voiture change la marque => N tuples à m.a.j.

Normalisation d'une relation

Processus de décomposition d'une relation à maj complexes en plusieurs relations à maj simples.

Processus sur le schéma relationnel formel.

Exemple :

La relation

□ **Contrat**(id_contrat, date_retour, montant, demande, marque, nb_jour, #id_reservation, #id_voiture) ;

sera décomposée en :

Contrat'(id_contrat, #id_reservation, #id_voiture, date_retour, montant, demande)

Voiture(id_voiture, marque, etc...) ;

Réservation(id_reservation, nb_jour, etc...) ;

Normalisation

On mesure la qualité d'une relation par son degré de normalisation :

1FN (première forme normale), 2FN, 3FN, FNBC (forme normale de Boyce Codd), 4FN, etc.

III.4 Les scripts de création de la base de données

```
-- Database Section
-- _____
create database Model Relationnel;
use Model Relationnel;

-- Tables Section
-- _____
create table Client (
    id_client bigint not null,
    pseudo varchar(21) not null,
    nom varchar(21) not null,
    prenom varchar(21) not null,
    mail varchar(128) not null,
    telephone int not null|,
    mot_de_passe varchar(32)* not null,
    id_admin bigint not null,
    constraint ID_Client_ID primary key (id_client));

create table Réservation (
    id_reservation bigint not null,
    date_depart date not null,
    nb_jour int not null,
    id_client bigint not null,
    constraint ID_reservation_ID primary key (id_reservation));
```

```
create table Voiture (
    id_voiture bigint not null,
    marque varchar(32) not null,
    description varchar(512) not null,
    immatriculation varchar(10) not null,
    prix float(3) not null,
    genre varchar(21) not null,
    image varchar(256) not null,
    statut varchar(21) not null,
    id_agent bigint not null,
    constraint ID_voiture_ID primary key (id_voiture));

create table Agent (
    id_agent bigint not null,
    pseudo varchar(21) not null,
    nom_agence varchar(64) not null,
    mail varchar(64) not null,
    telephone int not null,
    adresse varchar(256) not null,
    mot_de_passe varchar(32) not null,
    verification varchar(21) not null,
    id_admin bigint not null,
    constraint ID_agent_ID primary key (id_agent));

create table Admin (
    id_admin bigint not null,
    pseudo varchar(21) not null,
    mail varchar(128) not null,
    mot_de_passe varchar(32) not null,
    constraint ID_admin_ID primary key (id_admin));

create table Contrat (
    id_contrat bigint not null,
    date_retour date not null,
    montant float(4) not null,
    demande varchar(21) not null,
    id_voiture bigint not null,
    id_reservation bigint not null,
    constraint ID_contrat_ID primary key (id_contrat));
```

```
-- Constraints Section
-- _____
alter table Client add constraint FKobtenir_FK
    foreign key (id_admin)
    references Admin (id_admin);

-- Not implemented
-- alter table Réservation add constraint ID_reservation_CHK
--     check(exists(select * from Contrat
--                  where Contrat.id_reservation = id_reservation));

alter table Réservation add constraint FKreservee_FK
    foreign key (id_client)
    references Client (id_client);

-- Not implemented
-- alter table Voiture add constraint ID_voiture_CHK
--     check(exists(select * from Contrat
--                  where Contrat.id_voiture = id_voiture));

alter table Voiture add constraint FKgeree_FK
    foreign key (id_agent)
    references Agent (id_agent);

-- Not implemented
-- alter table Agent add constraint ID_agent_CHK
--     check(exists(select * from Voiture
--                  where Voiture.id_agent = id_agent));

alter table Agent add constraint FKdiriger_FK
    foreign key (id_admin)
    references Admin (id_admin);

alter table Contrat add constraint FKcon_voi_FK
    foreign key (id_voiture)
    references Voiture (id_voiture);
```

```

alter table Contrat add constraint FKcon_res_FK
    foreign key (id_reservation)
    references Réservation (id_reservation);

-- Index Section
-- _____

create unique index ID_Client_IND
    on Client (id_client);

create index FKobtenir_IND
    on Client (id_admin);

create unique index ID_reservation_IND
    on Réservation (id_reservation);

create index FKreservee_IND
    on Réservation (id_client);

create unique index ID_voiture_IND
    on Voiture (id_voiture);

create index FKgeree_IND
    on Voiture (id_agent);

create unique index ID_agent_IND
    on Agent (id_agent);

create index FKdiriger_IND
    on Agent (id_admin);

create unique index ID_admin_IND
    on Admin (id_admin);

create unique index ID_contrat_IND
    on Contrat (id_contrat);

create index FKcon_voi_IND
    on Contrat (id_voiture);

```

Figure 24: Les scripts de création de la base de données

les requêtes d'interrogation et de modification :

- o Pour afficher toutes les voitures dans l'ordre de statut (Libre / Occupée) : **SELECT * FROM voitures ORDER BY statut ;**
- o Pour récupérée les information d'une seul voiture par l'id_voiture .
SELECT * FROM voitures WHERE id_voiture = \$id
- o Pour insérée nouveau véhicule .

```
INSERT INTO voitures(marque, description, immatriculation, prix, genre, image, statut, id_agent)  
VALUES(?, ?, ?, ?, ?, ?, 'Libre', ".$SESSION['agenceld'].") ;  
NB : Pour les ' ?' c'est les valeurs entrée par l'utilisateur(agent).
```

- o L'agent doit afficher votre voitures par l'id.

```
SELECT * FROM voitures WHERE id_agent = ".$SESSION['agenceld'].";
```

- o Pour le registre d'un nouveau agent à besoin 2 requête SQL 1ere requête doit vérifier le pseudo pas existe.

```
SELECT pseudo FROM agent WHERE pseudo=:unameag ;
```

2eme requête si le pseudo entrée est correct par le 1ere requête.

```
INSERT INTO agent (pseudo, nom_agence, mail, tel, adress, demande, mot_de_passe, id_admin)  
VALUES (:unameag, :nomag, :mailag, :telag, :addressag, 'En attent', :pwdag,  
".$SESSION['idAdmin'].");
```

- o Pour la connexion d'agent utilise sont mail ou le pseudo. **SELECT * FROM** agent **WHERE** pseudo=? OR mail =?;

- o Pour récupere la demande de réservation d'un client pour l'agent **SELECT** demande **FROM** contrat **WHERE** id_voiture = \$id ;

\$id c'est le \$_GET['id'] dans l'URL .

- o Pour accepter nouveau réservation du l'agent vers le client.

```
UPDATE contrat SET demande = 'Accepter' WHERE demande = 'En attent' AND id_voiture = $id ;
```

- o Pour refuser la demande de réservation.

1ere requête doit modifier le statut voiture d'Occupée par Libre.

```
UPDATE voitures SET statut='Libre' WHERE id_voiture= $id ;
```

2eme requête doit modifier la demande dans le contrat 'En attent' par 'Annuler'.

```
UPDATE contrat SET demande = 'Annuler' WHERE demande = 'En attent' AND id_voiture= $id ;
```

- o L'agent doit supprimer une voiture.

```
DELETE FROM voitures WHERE id_voiture= $id ;
```

- o Pour recuperer les champs 'input' du formulaire pour modifier une voiture si l'agent doit choisir de changer parmi les données .

```
SELECT id_voiture, marque, description, prix, immatriculation FROM voitures WHERE id_voiture = $idget ;
```

- o L'étape de modification une véhicule par l'agent .

```
UPDATE voitures SET marque= $mq, description= $dsc, immatriculation= $imm, prix= $pr WHERE id_voiture= $idget ;
```

- o Pour recuperer le date de retour d'une véhicule.

```
SELECT date_retour FROM réservation as R, contrat as C, voitures as V WHERE R.id_reser = C.id_reser AND V.id_voiture = C.id_voiture AND V.id_voiture = $idret ;
```

- o Le client doit réservée une véhicule en utilise 3 requête simultanément. 1ere requête doit insérée dans les réservation .

```
INSERT INTO reservation(date_depart, nbjour, id_client) VALUES("".$value['datedep'].",
".$value['nbjour'].", ".$SESSION['clientId'].") ;
```

NB : les \$value["] c'est la récupération du tableau array() qui contient ces valeur.

2eme requete c'est la modification de statut en Libre vers Occupée de véhicule.

```
UPDATE voitures SET statut = 'Occupée' WHERE id_voiture=".$value['id_voit'].";
```

3eme requete c'est l'insertion dans le contrat.

```
INSERT INTO contrat(date_retour, montant, demande, id_reservation,
id_voiture)VALUES("".$date_ret.", ".$value['nbjour'] * $value['prix'].", 'En attent',
".$SESSION['id_reser'].", ".$value['id_voit'].");
```

- o Le client doit visualise le historique de réservation.

```
SELECT id_contrat, date_depart, date_ret, marque, montant, demande FROM contrat as C,
reservation as R, voitures as V WHERE C.id_voiture = V.id_voiture AND C.id_reservation =
R.id_reservation AND R.id_client = ".$SESSION['clientId'].";
```

- o Pour compter le nombre
de client. **SELECT** count(*)
FROM client ;

Partie 4 :

La Réalisation

Introduction

Après avoir terminé la conception détaillée de notre application, nous traitons dans le présent partie les détails liés à la réalisation de l'application. Pour cela, nous exposons tout d'abord les choix de l'environnement de travail (matériel et logiciel) adopté afin de réussir la réalisation de l'application, et nous décrivons les étapes d'implémentation suivies de quelques imprimés d'écrans de l'exécution de certains modules de l'application pour illustrer quelques fonctionnalités de notre système.

1. Environnement de travail

Dans cette partie nous allons présenter l'environnement de travail (matériel et logiciel) utilisé dans le développement notre application.

1.1 Environnement matériel

Pour la réalisation de l'application nous avons utilisé un ordinateur qui nous a permis de traiter et d'installer plusieurs logiciels. Dans ce qui suit, les principales caractéristiques de cet ordinateur

Matériel	Caractéristique
PC ASUS Vivobook	Système d'exploitation: Windows 11 Ecran: 14.6 pouces Processeur: Rayzen5 Mémoire: 16Go Disque Dur: 500 SSD Carte Graphique: AMD Radeon

1.2 Environnement logiciel

Dans cette partie nous allons présenter les logiciels utilisés pour la réalisation de notre application.

StarUML : L'Outil de conception UML StarUML est une plate-forme de modélisation avec le langage UML open source qui gère la plupart des diagrammes. Spécifiés dans la norme UML 3.2.2 qui est caractérisée par une forte flexibilité et une excellente extensibilité de ses fonctionnalités.



XAMPP : est un ensemble de logiciels servant à mettre en place aisément un serveur Web, un serveur FTP et un serveur de messagerie électronique. C'est une distribution de logiciels libres (X Apache MySQL Perl PHP) offrant une bonne souplesse d'utilisation, reconnue pour son installation simple et rapide.

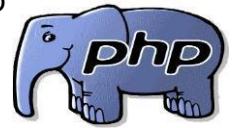


Microsoft Visual Studio 2019 : L'Environnement de développement Intégrés

Microsoft Visual Studio est une suite de logiciels de développement pour Windows conçue par Microsoft. Il représente un ensemble complet d'outils de développement permettant de générer des applications Web ASP.NET, des services web XML, des applications bureautiques et des applications mobiles. [1]



PHP: pour HyperText Preprocessor, désigne un langage informatique, ou un langage de script, utilisé principalement pour la conception de sites web dynamiques. Il s'agit d'un langage de programmation sous licence libre qui peut donc être utilisé par n'importe qui de façon totalement gratuite.



IV.1 Partie Programmation Web 2 :

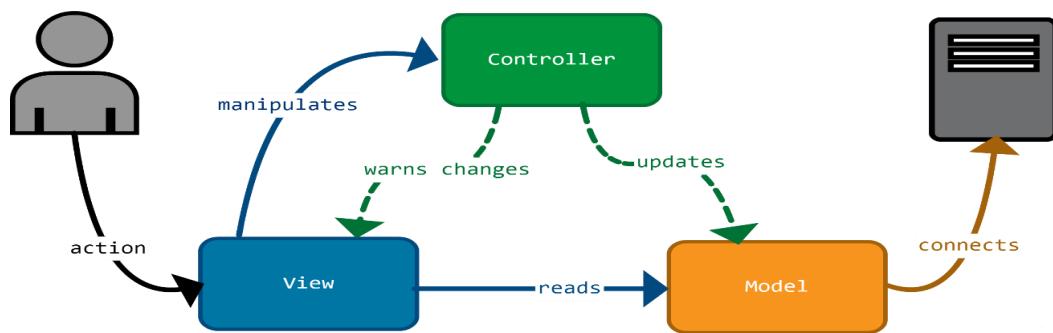
IV.1.1 Présentation détaillée du modèle MVC

Le modèle MVC (Modèle, Vue, Contrôleur) est une architecture qui organise les composants d'une application. Son principe consiste à séparer les données, les interfaces graphiques et les traitements en trois parties distinctes : modèle, vue et contrôleur.

- Modèle: Le rôle du modèle est de décrire le comportement de l'application et de gérer les données qu'elle manipule. Dans le cas d'accès aux bases de données, il prend en charge le trafic des enregistrements.
- Vue :La vue présente les interfaces graphiques qui s'affichent sur la machine d'un client. Le rôle de la vue n'est pas restreint à afficher les fenêtres de l'application, elle intercepte aussi les actions (événements) de l'utilisateur comme les clics de la souris et la saisie des textes avec le clavier. Suite à la réception de ces événements, la vue ne fait pas de traitement, elle les transmet au contrôleur.



Contrôleur :Le contrôleur est un relieur entre le modèle et la vue. Son rôle consiste à analyser les événements provenant des vues et commander au(x) modèle(s) d'exécuter les actions appropriées. La figure 15 suivante représente cette architecture



ainsi que les différentes interactions entre ces composants :

IV.1.2 Présentation et description des interfaces

Nous présentons dans cette partie quelques interfaces de notre application.

IV.1.2.1 Interface d'accueil de l'application.

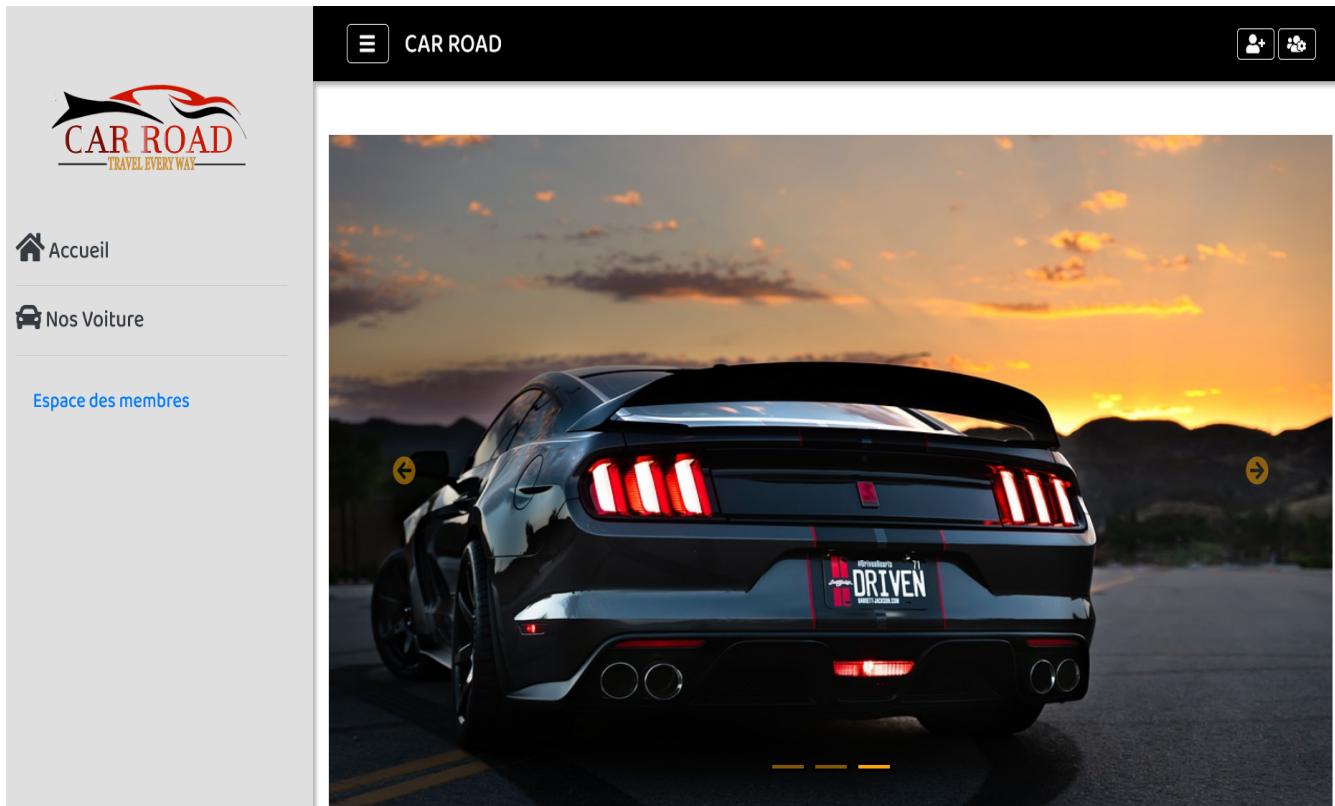


Figure 25: Interface d'accueil de l'application

La figure 18 représente l'interface d'accueil de notre application, l'utilisateur visualise les descriptions de réservation. Il peut accéder aux véhicules et il peut connecter ou s'inscrire à l'application.

IV.1.2.2 Interface des voitures de l'application.

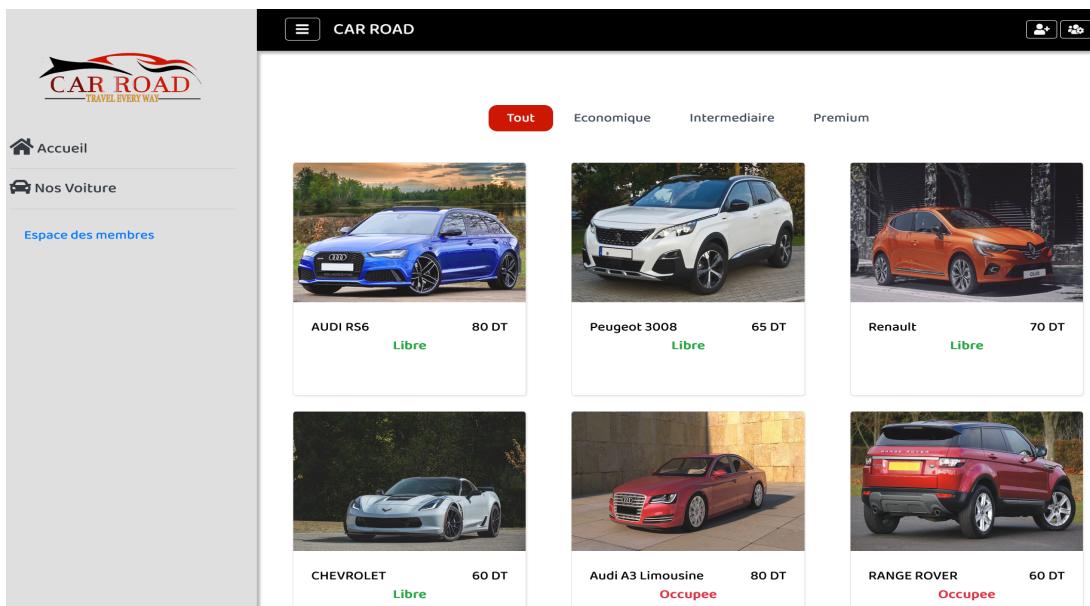


Figure 26: Interface des voitures de l'application

La figure 18 représente l'interface des voitures de notre application, l'utilisateur visualise les voitures disponible de réservation. Il peut accéder aux détails et permet filtrée les genres des véhicule selon les bouton (Economique, intermédiaire, supérieur) , il peut connecter ou s'inscrire à l'application.

IV.1.2.3 Interface des connexion client de l'application.

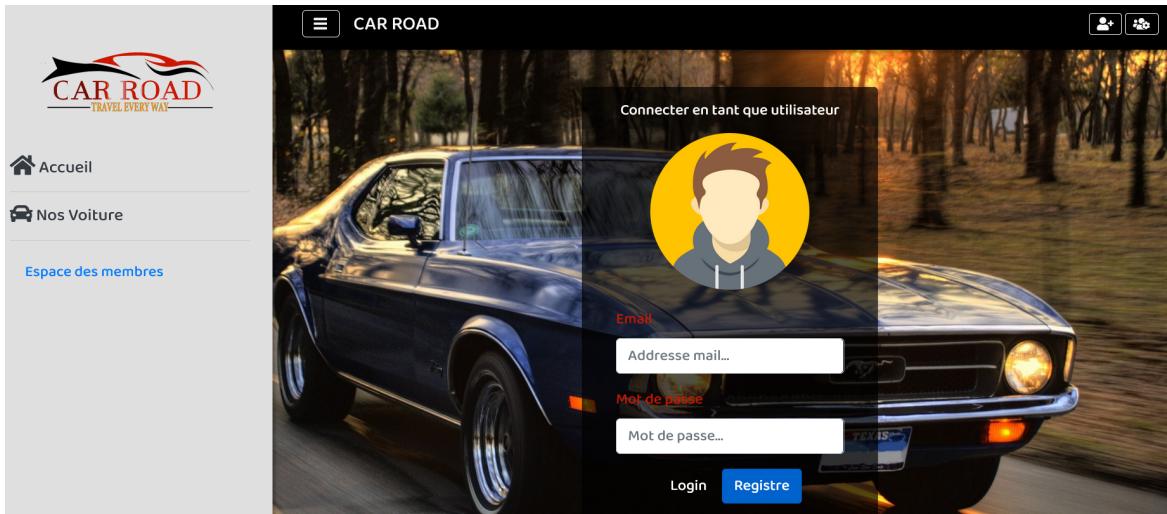


Figure 27: Interface de connexion du client de l'application

IV.1.2.4 Interface des connexion agence de l'application.

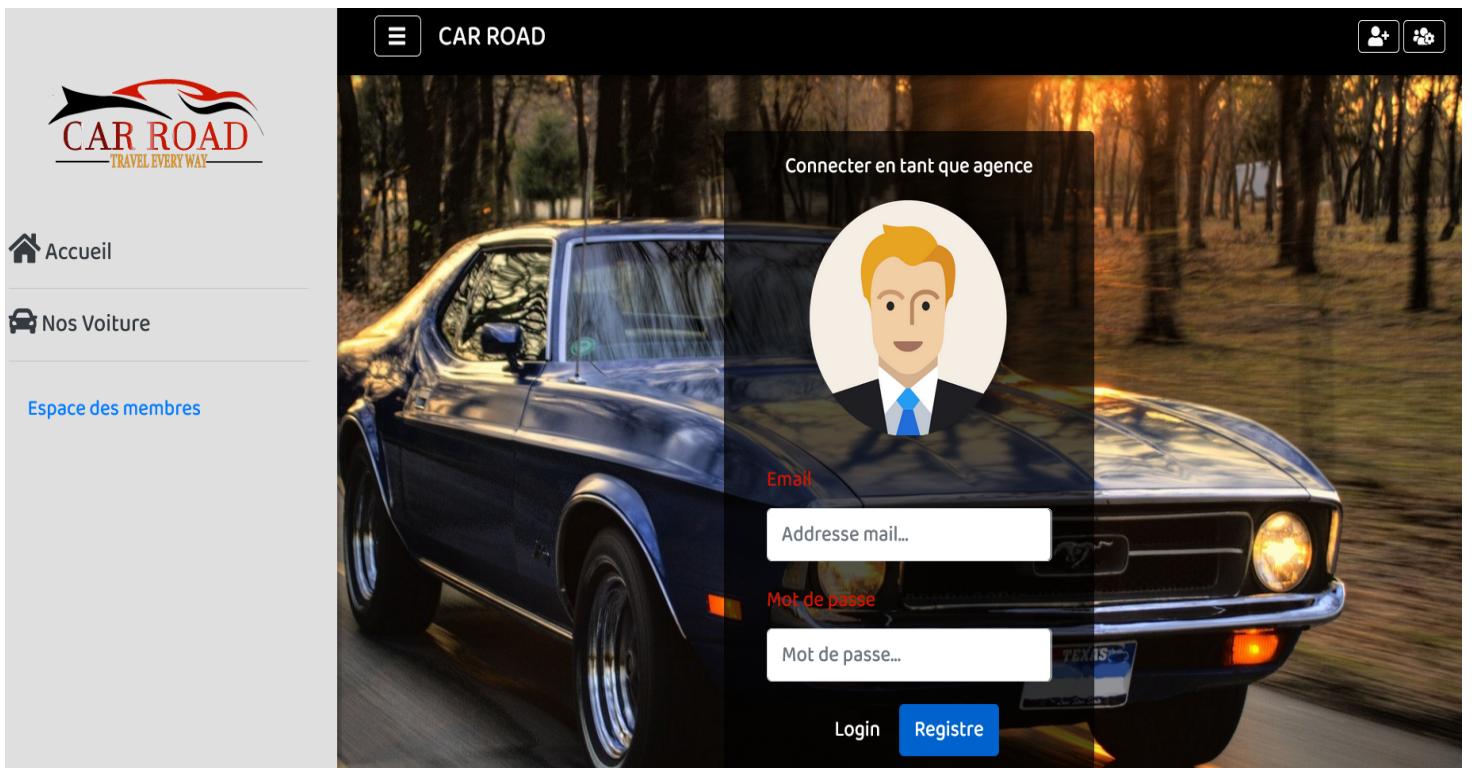


Figure 28: Interface des connexion agence de l'application

Conclusion et perspectives

Durant les derniers deux mois, nous avons tenté de réaliser un portail web et desktop dynamique pour location de véhicules .

L'objectif visé dans ce projet de fin d'année est la mise en place d'un portail Web et Desktop, pour atteindre cet objectif, on a abordé notre problème en s'appuyant sur la démarche de Merise, en ce qui concerne la réalisation, on a utilisé le langage PHP, et C# Cette application a permis de répondre aux besoins des tunisiens d'un espace web qui permet de faciliter les taches de réservations des véhicules en ligne.

Ce projet a fait l'objet d'une expérience intéressante, qui nous a permis d'améliorer nos connaissances et nos compétences dans le domaine de la programmation. Nous avons appris à mieux manipuler les langages C#, PHP, HTML, MYSQL, CSS et Java Script.

En effet, ce travail étant une œuvre humaine, n'est pas un modèle unique et parfait, c'est pourquoi nous restons ouverts à toutes les critiques et nous sommes prêts à recevoir toutes les suggestions et remarques tendant à améliorer d'avantage cette étude. Etant donné que tout travail informatique a été toujours l'œuvre d'une équipe.

Nous espérons que ce travaille, soit d'une utilité pour les citoyens tunisiens.

Quelques perspectives Nombreuses sont les idées dont on pouvait incorporer dans ce projet. Cependant, vu le temps imparti, il a fallu une limite pour cette version première de l'application. Ainsi, en perspective nous pouvons envisager quelques améliorations ci et l'avisant à rendre cette projet de plus en plus performante. Mais aussi à ce stade, l'idéal