**Signatures**

Encadrant Entreprise

Encadrant ISIKef

**Dédicaces**

Je dédie ce modeste travail et ma profonde gratitude :

**À mon père et ma mère**

Mais aucune dédicace ne serait témoin de mon profond amour, mon immense gratitude et mon respect, car je ne pourrais jamais oublier la tendresse et l’amour dévoué par lesquels ils m’ont toujours entouré depuis mon enfance.

**À toute ma famille, frères et sœurs**

Pour leur soutien moral.

À tous mes amis et à tous ceux que j’aime et à toutes les personnes qui m’ont prodigue des encouragements et se sont donné la peine de me soutenir durant ce projet.

**À tous les personnels de LB Consulting**

Qui m’a bien aidé à atteindre mon objectif.

**Hamza Mgaidi**

Remerciements

C’est avec un grand plaisir que je réserve cette page en signe de gratitude et de profonde reconnaissance à tous ceux qui m’ont aidé à la réalisation de ce travail.

Je tiens à remercier, tout d’abord la société LB Consulting au sein de laquelle j’ai eu l’honneur de travailler dans une ambiance dynamique et professionnelle.

Mes remerciements s’adressent particulièrement à :

Mon encadrant universitaire Monsieur Nour Ben Yahia, enseignant à l’Institut Supérieur d’Informatique du Kef, pour son encadrement, ses remarques constructives tout le long de mon travail.

Tout le cadre professoral et administratif de l’Institut Supérieur d’Informatique du Kef (ISIKef).

Monsieur Mahmoud Ben Lamine,

Sénior Manager de LB Consulting, pour m’avoir prodigué l’honneur de travailler au sein de la société.

Monsieur Badreddine Barhoumi,

Manager développement à LB Consulting, pour sa supervision, ses conseils, son support, et les bons moments que nous avons passés ensemble.

Monsieur Safwen Derouich,

Ingénieur recherche et développement à LB Consulting, pour sa disponibilité, sa sympathie, son aide, et l’intérêt qu’il m’a accordé tout au long de mon stage.

Mon dernier mot s’adresse à tous les membres de jury pour l’honneur qu’ils nous font de participer à l’examen de notre mémoire, sans oublier tous nos professeurs pour la formation de qualité qu’ils nous ont prodiguée tout au long de notre cursus universitaire.

Table des matières

[Introduction Générale 1](#_Toc513555011)

[Chapitre 1 : Présentation de l’organisme d’accueil et Etude préalable 3](#_Toc513555012)

[1. Introduction : 3](#_Toc513555013)

[2. Présentation générale : 3](#_Toc513555014)

[2.1. Présentation de l’organisme : 3](#_Toc513555015)

[2.2. Les domaines d’activités de LB Consulting: 3](#_Toc513555016)

[2.3. Les technologies les plus avancées de LB Consulting : 3](#_Toc513555017)

[2.4. Clients et projets réalisé par LB Consulting : 4](#_Toc513555018)

[3. Présentation du thème du stage : 4](#_Toc513555019)

[3.1. Champ de l’étude et objet du stage : 4](#_Toc513555020)

[3.2 Planning prévisionnel du stage : 4](#_Toc513555021)

[4. Etude préalable : 4](#_Toc513555022)

[4.1. Description de l’existant : 4](#_Toc513555023)

[4.2. Critiques de l’existant : 5](#_Toc513555024)

[4.3. Solutions proposées : 7](#_Toc513555025)

[5. Conclusion : 8](#_Toc513555026)

[Chapitre 2 : Analyse et spécification des besoins 9](#_Toc513555027)

[1. Introduction : 9](#_Toc513555028)

[2. Etude des besoins : 9](#_Toc513555029)

[2.1 Besoins fonctionnels : 9](#_Toc513555030)

[2.2 Besoins non fonctionnels : 10](#_Toc513555031)

[3. Identification des acteurs et cas d’utilisation : 11](#_Toc513555032)

[3.1 Présentation des acteurs: 11](#_Toc513555033)

[3.2 Identification des cas d’utilisation : 11](#_Toc513555034)

[3.3 Structuration des cas d’utilisation: 12](#_Toc513555035)

[3.3.1 Identification des acteurs : 12](#_Toc513555036)

[3.3.2 Diagramme de cas d’utilisation global : 12](#_Toc513555037)

[4. Diagramme de séquence : 24](#_Toc513555038)

[5. Conclusion : 30](#_Toc513555039)

[Chapitre 3 : Conception détaillée 31](#_Toc513555040)

[1. Introduction : 31](#_Toc513555041)

[2. Extractions des différentes entités : 31](#_Toc513555042)

[3. Extraction des attributs des différentes entités : 31](#_Toc513555043)

[4. Diagramme de classes : 37](#_Toc513555044)

[4.1 Modèle relationnel : 38](#_Toc513555045)

[5. Diagramme d’interaction : 38](#_Toc513555046)

[5.1 Diagramme de séquence du cas d’utilisation « S’authentifier » 39](#_Toc513555047)

[5.2 Diagramme de séquence du cas d’utilisation « Gérer marques » 40](#_Toc513555048)

[5.3 Diagramme de séquence du cas d’utilisation « Gérer modèles » 43](#_Toc513555049)

[5.4 Diagramme de séquence du cas d’utilisation « Gérer catégories » 46](#_Toc513555050)

[5.5 Diagramme de séquence du cas d’utilisation « Gérer véhicule » 49](#_Toc513555051)

[6. Conclusion : 52](#_Toc513555052)

[Chapitre 4 : Réalisation 53](#_Toc513555053)

[1. Introduction : 53](#_Toc513555054)

[2. Environnement matériel : 53](#_Toc513555055)

[3. Technologies utilisées : 53](#_Toc513555056)

[4. Environnement logiciel de développement : 56](#_Toc513555057)

[5. Architecture technique du projet : 58](#_Toc513555058)

[6. Interfaces réalisées : 59](#_Toc513555059)

[7. Conclusion : 66](#_Toc513555060)

[Conclusion générale 67](#_Toc513555061)

[Webographie 68](#_Toc513555062)

[Bibliographie 68](#_Toc513555063)

Liste des figures

[Figure 1 : Mise à jour d’une marque 5](#_Toc513550843)

[Figure 2 : Mise à jour modèle 6](#_Toc513550844)

[Figure 3 : Mise à jour catégorie 6](#_Toc513550845)

[Figure 4 : Mise à jour d’un véhicule 7](#_Toc513550846)

[Figure 5 : Association entre catégorie et modèle 7](#_Toc513550847)

[Figure 6: Représentation d'un acteur 12](#_Toc513550848)

[Figure 7: Représentation des acteurs de la plateforme 12](#_Toc513550849)

[Figure 8: Diagramme du cas d’utilisation global 13](#_Toc513550850)

[Figure 9: Raffinement du cas d’utilisation « S’authentifier » 14](#_Toc513550851)

[Figure 10: Raffinement du cas d’utilisation « Gérer agent » 15](#_Toc513550853)

[Figure 11: Raffinement du cas d’utilisation « Gérer marque » 17](#_Toc513550855)

[Figure 12: Raffinement du cas d’utilisation « Gérer modèle » 19](#_Toc513550857)

[Figure 13: Raffinement du cas d’utilisation « Gérer catégorie » 21](#_Toc513550859)

[Figure 14: Raffinement du cas d’utilisation « Gérer véhicule » 22](#_Toc513550861)

[Figure 15: Diagramme de séquence authentification 24](#_Toc513550863)

[Figure 16: Diagramme de séquence gérer agents 25](#_Toc513550864)

[Figure 17: Diagramme de séquence gérer marques 26](#_Toc513550865)

[Figure 18: Diagramme de séquence gérer modelés 27](#_Toc513550866)

[Figure 19: Diagramme de séquence gérer catégories 28](#_Toc513550867)

[Figure 20: Diagramme de séquence gérer véhicules 29](#_Toc513550868)

[Figure 21 : diagramme des classes 37](#_Toc513550875)

[Figure 22 : diagramme de séquence du cas d’utilisation « s’authentifier » 39](#_Toc513550882)

[Figure 23 : Diagramme de séquence d'ajout d'une marque de cas gérer marques 40](#_Toc513550884)

[Figure 24 : Diagramme de séquence de modification d'une marque de cas gérer marques 41](#_Toc513550885)

[Figure 25 : Diagramme de séquence de désactivation d'une marque de cas gérer marques 42](#_Toc513550886)

[Figure 26 : Diagramme de séquence d'ajout d'un modèle de cas gérer modèles 43](#_Toc513550894)

[Figure 27 : Diagramme de séquence de modification d'un modèle de cas gérer modèles 44](#_Toc513550895)

[Figure 28 : Diagramme de séquence de désactivation d'un modèle de cas gérer modèles 45](#_Toc513550896)

[Figure 29 : Diagramme de séquence d'ajout d'une catégorie de cas gérer catégories 46](#_Toc513550904)

[Figure 30 : Diagramme de séquence de modification d'une catégorie de cas gérer catégories 47](#_Toc513550905)

[Figure 31 : Diagramme de séquence de désactivation d'une catégorie de cas gérer catégories 48](#_Toc513550906)

[Figure 32 : Diagramme de séquence d'ajout d'un véhicule de cas gérer véhicules 49](#_Toc513550914)

[Figure 33 : Diagramme de séquence de modification d'un véhicule de cas gérer véhicules 50](#_Toc513550915)

[Figure 34 : Diagramme de séquence de désactivation d'un véhicule de cas gérer véhicules 51](#_Toc513550916)

[Figure 35 : Architecture technique de projet 58](#_Toc513550924)

[Figure 37 : Interface d’authentification 59](#_Toc513550925)

[Figure 39 : schéma de login 60](#_Toc513550926)

[Figure 40 : interface marques 60](#_Toc513550927)

[Figure 41 : interface d’ajout d’une marque 61](#_Toc513550928)

[Figure 42 : interface modèles 62](#_Toc513550929)

[Figure 43 : interface d’ajout d’un modèle 63](#_Toc513550930)

[Figure 44 : interface catégories 63](#_Toc513550931)

[Figure 45 : interface d’ajout d’une catégorie 64](#_Toc513550932)

[Figure 46 : interface véhicules 65](#_Toc513550933)

[Figure 47 : interface d’ajout d’un véhicule 66](#_Toc513550934)

Liste des tableaux

[Tableau 1: Description textuelle du cas d’utilisation « s’authentifier » 14](#_Toc513210031)

[Tableau 2: Description textuelle du cas d’utilisation « Gérer agent » 16](#_Toc513210033)

[Tableau 3: Description textuelle du cas d’utilisation « Gérer marque » 18](#_Toc513210035)

[Tableau 4: Description textuelle du cas d’utilisation « Gérer modèle » 20](#_Toc513210037)

[Tableau 5: Description textuelle du cas d’utilisation  « Gérer catégorie » 22](#_Toc513210039)

[Tableau 6: Description textuelle du cas d’utilisation « Gérer véhicule » 23](#_Toc513210041)

[Tableau 7 : Les entités 31](#_Toc513210048)

[Tableau 8 : entité Utilisateur 32](#_Toc513210049)

[Tableau 9 : entité marque 32](#_Toc513210050)

[Tableau 10 : entité modèle 33](#_Toc513210051)

[Tableau 11 : entité catégories 35](#_Toc513210052)

[Tableau 12 : entité véhicule 36](#_Toc513210053)

Liste des abréviations

* API = Application Programming Interface
* CSS = Cascading Style Sheets
* HTML = HyperText Markup Language
* HTTP = HyperText Transfer Protocol
* ISI = Institut Supérieur d’Informatique
* JDBC = Java DataBase Connectivity
* DAO = Data Access Object
* DTO = Data Transfer Object
* JSON = JavaScript Object Notation
* IDE = Integrated Development Environment
* JPA = Java Persistence API
* MVC = Modèle-Vue-Contrôleur
* PDG = Président-Directeur Général
* PFE = Projet de Fin d’Etude
* SGBD = Système de Gestion de Base de Données
* JDBC = Java Database Connectivity
* SQL = Structured Query Language
* UML = Unified Modeling Language
* XML = Extensible Markup Language

# Introduction Générale

De nos jours, la dispersion spatiale de l’habitat et les activités contribuent à une croissance considérable du trafic et de l’utilisation des voitures. La flexibilité, l’efficacité, la rapidité et le confort de la voiture particulière ont fait que ce moyen de transport devient le moyen le plus populaire et préféré des utilisateurs.

Mais et dans un contexte économique d’augmentation du prix du pétrole et de baisse du pouvoir d’achat, les pratiques de déplacement évoluent afin de limiter le coût de la mobilité.

Ainsi, de nouvelles pratiques de la mobilité sont en train de se développer : location à l’heure ou à la journée, covoiturage, auto-partage, voiture en libre-service… Celles-ci réinterrogent l’organisation du secteur professionnel de la location automobile.

Il est indispensable de noter que l’informatique est une discipline très riche et variée qui est devenue inévitable dans tous les domaines et secteur en rendant le travail plus facile, plus précis et surtout bien géré. Elle occupe une grande place dans le domaine du transport et particulièrement, la gestion des parcs automobiles.

La gestion de flotte est la gestion d’utilisation et de maintenance d’un parc de véhicules. Par extension, elle englobe souvent la gestion des conducteurs. Elle désigne une large gamme de solutions appliquées aux véhicules, aidant les entreprises à gérer leurs flottes de véhicules professionnels tels que les voitures utilitaires, légères, bus et poids lourds.

Les entreprises se trouvant dans un univers en perpétuelle évolution, essaient de modifier la culture du papier en numérisant leurs processus. Ce qui est sans doute une garantie de productivité et de gain.

Le stage est une période très importante, c’est le point de départ à la vie professionnelle, il permet à l’étudiant-stagiaire d’appliquer les connaissances et les théories requises pendant son cursus universitaire et d’avoir une large vision de la vie professionnelle.

Ainsi dans le cadre de l’obtention du diplôme en Licence Appliquée en Multimédia et développement web, nous avons réalisé un stage de trois mois au sein de la Société LB Consulting située au pôle technologique el Ghazela.

Ce rapport présente l’ensemble des étapes suivies pour développer l’application. Il contient quatre chapitres organisés comme suit : Le premier chapitre intitulé « Présentation de l’organisme d’accueil et Etude préalable », est consacrée à la présentation du contexte du projet, de l’organisme d’accueil avec les objectifs à atteindre. Nous présentons dans un premier temps le système existant en mettant en œuvre ses défaillances et ses limites. Nous concluons ce chapitre parla présentationde la solution proposée afin de pallier aux limites du système actuel

Le deuxième chapitre, intitulé « Analyse et spécification des besoins », s’articule autour de l’identification des acteurs et description des besoins fonctionnels et non fonctionnels, leurs présentations formelles, la structuration des cas d’utilisation, et les diagrammes de séquence.

Le troisième chapitre, intitulé « Conception détaillée », il s’agit de la phase conceptuelle qui sera matérialisée par l’élaboration du diagramme de classes ainsi que celle des diagrammes d’interaction et les tables de la base de données.

Le dernier chapitre, intitulé « Réalisation », présentera l’environnement du travail, l’architecture et la phase d’implémentation par la description des interfaces de l’application.

Enfin, nous clôturons par une conclusion générale ainsi par la proposition de quelques perspectives sur lesquels peut s’ouvrir le présent travail.

# 

# Chapitre 1 : Présentation de l’organisme d’accueil et Etude préalable

1. Introduction :

Le chapitre présent est dédié à la présentation du projet par la fixation des objectifs à atteindre. Dans une première section, je commence par exposer l'entreprise accueillante, à savoir LB Consulting, l'organisme qui m’a accueilli durant mon stage de fin d'études. Dans la deuxième section l’étude de l'existant ainsi que sa critique et finalement je présente les solutions proposées.

2. Présentation générale :

2.1. Présentation de l’organisme :

Ce stage est déroulé au sein de la société LB Consulting, une société fondée en Mai 2004.

Il s’agitd’une société à responsabilité limitée (SARL) non résidente, totalement exportatrice avec un chiffre d’affaires de 1,5 million de dinars. Elle s’est installée en janvier 2006 au pôle El Ghazala. Elle est composée d’un effectif de 50 personnes réparties sur plusieurs équipes.

2.2. Les domaines d’activités de LB Consulting:

* Conseil en Yield Management hôtelier.
* Conception, développement et maintenance de solutions de Yield Management hôtelier.
* Conception, développement et maintenance de solution de Business Intelligence.
* Conception, développement et maintenance d’application mobiles pour l’hôtellerie.
* Support technique de solution de Yield Management.

2.3. Les technologies les plus avancées de LB Consulting :

* Plateformes mobiles IOS, Android et Windows phones
* JavaEE
* DotNet (Asp.net, c#) c++
* SQL server BI

2.4. Clients et projets réalisé par LB Consulting :

* Projet optims pour le Client Eurotunnel (France)
* Projet Camptats pour le compte de Yelloh village et Sandaya (France)
* Prestation de service pour le compte d’AvailPro (France)
* Divers projet de développement mobile pour le compte des sociétés de Moyen-Orient.

3. Présentation du thème du stage :

3.1. Champ de l’étude et objet du stage :

Dans notre monde moderne les entreprises ont besoin de nombreuses solutions pour gagner du temps et de maximiser leurs potentiels dans le cœur deleurs activités et de sécuriser au summum leurs business.

Donc ma mission a été de créerun module qui appartient à une grande application web qui permet de gérer une flotte automobile.

3.2 Planning prévisionnel du stage :

Dans le but de terminer l’application dans les délais établis par la convention de stage, il a fallu déterminer les différentes phases et estimer le temps à consacrer pour chacune.

Pour cela, et en collaboration avec le responsable de l’entreprise, j’ai élaboré un planning.

Ce dernier contiendra 4 étapes principales à réaliser :

* Etude de l’organisme d’accueil et la structure concernée par le stage
* Etude préalable
* Reformulation des besoins
* Réalisation.

La rédaction du rapport de stage se fera en parallèle avec toutes les phases du projet.

4. Etude préalable :

Dans cette partie je vais étudier l’existant qui va me permettre d’avoir un aperçu général des besoins de l’application.

4.1. Description de l’existant :

LB Consulting, est une boîte de développement qui travaille sur plusieurs projets sur le marché français, l'un de ces projets en collaboration avec Fly Car, une société française spécialisée dans la location des véhicules utilitaires, qui renouvelle complètement son application web. De mon côté je contribue au développement d’une partie de la nouvelle version de l'application.

En effet, la plateforme actuelle consiste en un backoffice destiné aux agences de la société Fly Car et un site web destiné principalement aux réservations en ligne.

4.2. Critiques de l’existant :

Après plusieurs recherches des solutions existantes, l’analyse fonctionnelle et technique de la plateforme Fly Car et d’avoir posé différentes questionsaux membres de l’équipe de l'entreprise, sur la procédure existante, j'ai pu avoir une vision globale à propos de la problématique ainsi que les points faibles et les erreurs qui peuvent affecter le processus de gestion des flottes au niveau des marques, modèles, catégories et véhicules.

La partie configuration de l’application actuelle est basique et sa conception présente des points manquants et d’autres impertinents.

Dans ce qui suit, je mets en évidence ces points dans la partie que j’ai développée.

* Marque :

La configuration des marques ne permet ni d’ajouter le logo de la marque ni de désactiver les marques non utilisées :

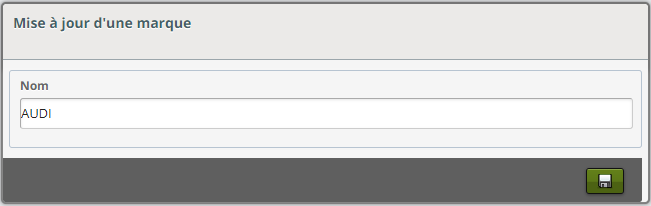


Figure 1 : Mise à jour d’une marque

* Modèle :

La configuration du modèle permet juste de l’associer à une marque :

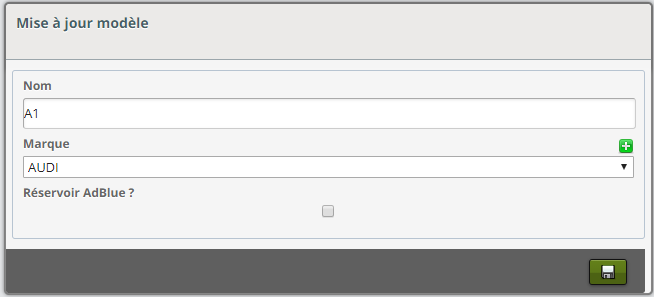


Figure 2 : Mise à jour modèle

En réalité, un modèle est caractérisé par d’autres spécificités qu’on n’a pas besoin de définir pour chaque véhicule (matricule donné). Je cite par exemple la capacité du réservoir, la charge utile, les accessoires de base et l’intervalle de révision en kilomètres.

D’autre part, un modèle de véhicule doit appartenir à une catégorie de véhicule unique. Celle-ci regroupe l’ensemble de tous les modèles (toutes marques confondues) qui ont le même prix de location (donc appartenant à la même « gamme »).

* Catégorie :

La définition actuelle manque entre autres : le positionnement de la catégorie par rapport aux autres catégories de point de vue prix, le nombre de petites et de grandes valises, le type de la boîte vitesse, la photo et le fait de pouvoir désactiver une catégorie.

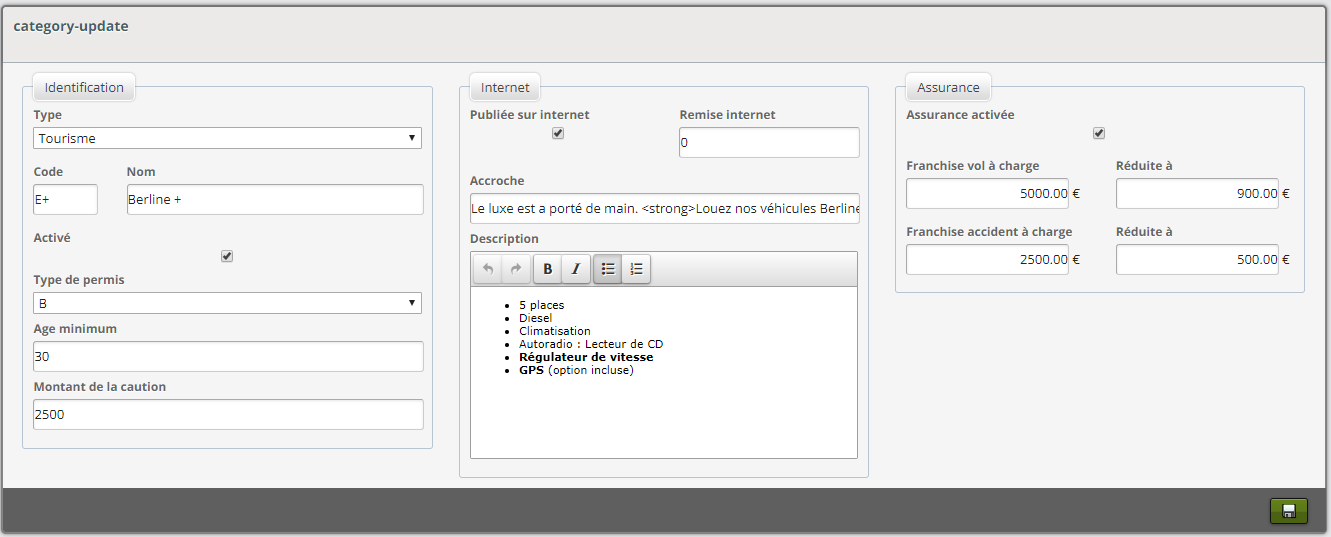


Figure 3 : Mise à jour catégorie

En plus, la partie assurance doit être gérée en tant que produits facturables aux clients finaux donc dans un écran à part (ne fait pas partie de mon projet).

* Véhicule :

La configuration des véhicules met en évidence la redondance de certaines informations qui peuvent être définies une fois pour toute dans la définition du modèle correspondant (capacité réservoir carburant par exemple). Comme dans le cas de la définition de la catégorie, la partie assurance doit être gérée à part.

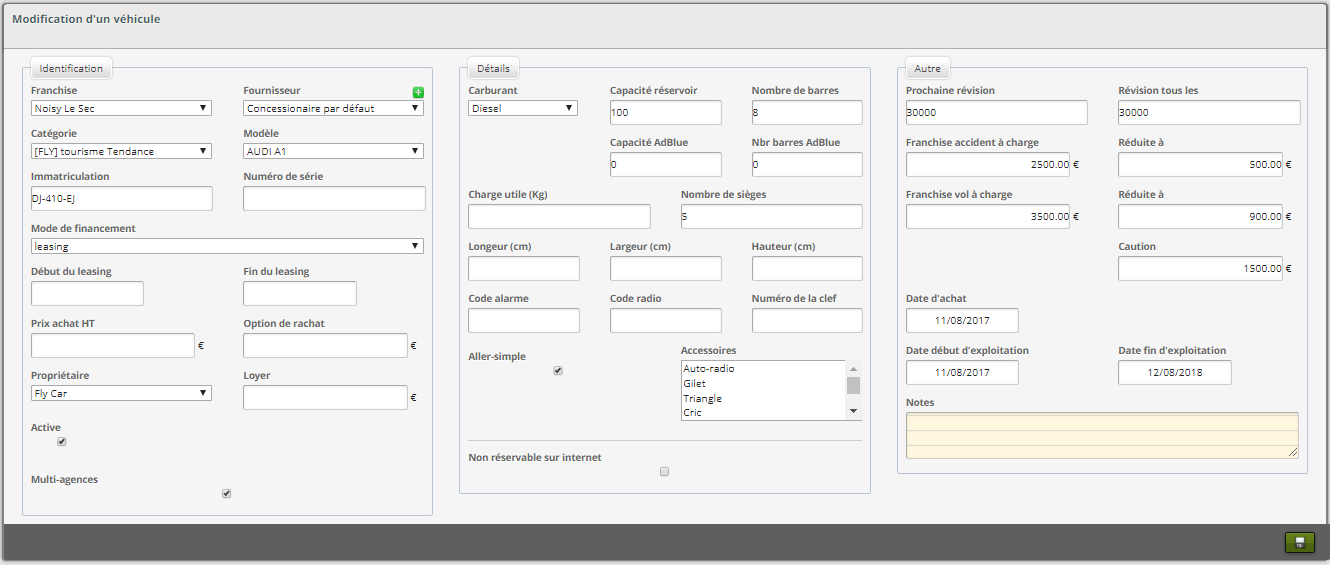


Figure 4 : Mise à jour d’un véhicule

Cette configuration souligne également une impertinence au niveau de l’association entre catégorie et modèle. En effet, dans la définition d’un véhicule, l’utilisateur à le libre choix d’affecter le véhicule à une catégorie quelconque d’une part et à un modèle quelconque d’autre part. Ceci pourrait amener à la configuration suivante :



Figure 5 : Association entre catégorie et modèle

Ce qui est totalement déraisonnable.

4.3. Solutions proposées :

Le nouveau système va présenter des fonctionnalités avancées et garantir la pertinence des liens entre ses différentes entités.

La partie configuration sera très riche et englobera le maximum de paramètres nécessaires pour une meilleure flexibilité du système.

* L’ajout d’une marque de véhicule englobe 3 paramètres où on peut ajouter le logo et désactiver les marques non utilisées
* La définition d’un modèle de véhicule englobe 12 paramètres qui permettent de simplifier la définition du véhicule.
* La définition d’une catégorie de véhicules englobe 14 paramètres qui la définissent totalement.
* La définition d’un véhicule englobe 14 paramètres spécifiques au véhicule en question.
* La partie franchise d’assurance par catégorie de véhicules sera gérée à part (ne fait pas partie de mon projet).
* Un nouveau véhicule doit être associé à un modèle de véhicule par l’utilisateur. Ce modèle est automatiquement associé à la catégorie correspondante.

5. Conclusion :

Ce chapitre a été consacré pour la présentation de l’organisme d’accueil, le cadre général du projet ainsi que la problématique.

Une étude détaillée afin de connaitre les différentes contraintes pour l'élaboration de ce projet.

Compte tenu de cette présentation le prochain chapitre sera dédié à la mise en œuvre des exigences et des besoins de notre application.

Chapitre 2 : Analyse et spécification des besoins

1. Introduction :

L’objectif de la phase d’analyse et de spécification des besoins consiste à détailler l’ensemble des fonctionnalités qui seront offertes par l’application. Les besoins dégagés ont été répartis en deux groupes fonctionnels et non fonctionnels.

1. Etude des besoins :

La spécification des besoins consiste à définir l’ensemble des fonctionnalités que l’application doit fournir. Elle exprime les souhaits et les besoins des acteurs qui n’ont parfois pas de compétences particulières en informatique (sous forme non formelle).

* 1. Besoins non fonctionnels :

En plus des besoins fonctionnels développés ci-dessus nous devons tenir compte des besoins en matière de performance, de type de matériel ou le type de conception. Ces besoins peuvent concerner les contraintes d’implémentation (langage de programmation, type SGBD).

* 1. Aussi, il est im Besoins fonctionnels :

Un besoin fonctionnel est un besoin spécifiant une action qu’un système doit être capable d’effectuer, sans considérer aucune contrainte physique c’est-à-dire un besoin spécifiant un comportement d’entrée/sortie d’un système.

La spécification des besoins fonctionnels consiste donc à recenser et à caractériser les différentes fonctions que le futur système à concevoir doit offrir pour satisfaire les besoins de ses futurs utilisateurs. Pour spécifier ces exigences fonctionnelles, nous allons utiliser deux concepts fondamentaux du langage de modélisation UML ; à savoir les acteurs et les cas d’utilisation.

L’utilisateur qui représente le chef d’agence accède aux fonctionnalités du système après l’authentification par un login et mot de passe fournis par le président-directeur général (P-DG) pour effectuer les tâches suivantes :

* Gérer les agents : il ajoute et supprime des agents et parallèlement gère leur accès aux écrans d’application
* Gérer les marques : il consulte, ajoute, modifie et désactive les marques de voitures.
* Gérer les modèles : il consulte, ajoute, modifie et désactive les modèles de voitures.
* Gérer les catégories : il consulte, ajoute, modifie et désactive les catégories.
* Gérer les véhicules : il consulte, ajoute, modifie et désactive les véhicules.

L’utilisateur qui représente l’agent simple accède aux fonctionnalités du système après l’authentification par un login et mot de passe fournis par l’administrateur (chef d’agence) pour effectuer les tâches suivantes :

* Gérer les marques : il consulte, ajoute, modifie et désactive les marques de voitures.
* Gérer les modèles : il consulte, ajoute, modifie et désactive les modèles de voitures.
* Gérer les catégories : il consulte, ajoute, modifie et désactive les catégories.
* Gérer les véhicules : il consulte, ajoute, modifie et désactive les véhicules.

portant de ne pas perdre de vue ses besoins non fonctionnels qui ont pour but l’établissement d’une ergonomie adaptée aux contenus. Ainsi, dans ce projet, nous allons conserver et renforcer le design graphique en tenant compte des:

* + 1. Contraintes esthétiques :
* Réserver une place pour le logo
* Minimiser le nombre de couleurs
* Contrastes de couleurs
* Assurer le contraste entre le texte et le fond
* La mise en page du texte doit être aérée.
  + 1. Contraintes ergonomiques :
* Développer une interface simple et compréhensible pour l’utilisateur
* Confort visuel et visibilité des caractères
* Permettre un accès rapide à l’information
* Hiérarchie logique des liens
  + 1. Contraintesperformances :
* Le temps de chargement des pages et des traitements des données doit avoir des temps de réponse courts
  + 1. Contraintes de Fiabilité :
* L’application doit toujours être en mesure de fonctionner correctement, il ne doit pas y avoir des problèmes de fonctionnement ou des cas limites qui donnent des résultats non souhaités ou erronés.
  + 1. Contraintes Sécurités :
* L’accès doit être personnalisé, la connexion et les données doivent être complètement sécurisées.

1. Identification des acteurs et cas d’utilisation :
   1. Présentation des acteurs:

* Un acteur est l'idéalisation d'un rôle joué par une personne externe, un processus ou une chose qui interagit avec un système. Il se représente par un petit bonhomme avec son nom inscrit au-dessous.

Les acteurs sont définis comme étant les utilisateurs de cette application. Dans le cadre de ce projet, on distingue deux acteurs principaux:

* L’administrateur : il possède tous les privilèges, à savoir l'accès à l’application et la gestion des agents.
* L’agent: toute personne qui fait son authentification, il peut visiter une ou plusieurs interfaces de l’application, consulter, modifier et désactiver les marques, modèles, catégories et véhicules.
  1. Identification des cas d’utilisation :

Les cas d'utilisation permettent de structurer les besoins des utilisateurs et les objectifs correspondants d'un système. Ils centrent l'expression des exigences du système sur ses utilisateurs.

Dans la présente application j'ai réparti les cas d’utilisation selon les acteurs cités précédemment:

* Assurer l'organisation : l’administrateur peut gérer les agents
* Gérer les mises à jour de contenu : l’utilisateur peut ajouter des nouvelles marques, modèles, catégorie, véhicules ou bien modifier ou désactiver.
* Gérer la base de données : l’administrateur ou bien l’agent peut faire la mise à jour de la base.
  1. Structuration des cas d’utilisation:
     1. Identification des acteurs :

Un acteur représente le rôle d’une entité présentée dans le contexte du système considéré vis-à-vis de ce système.

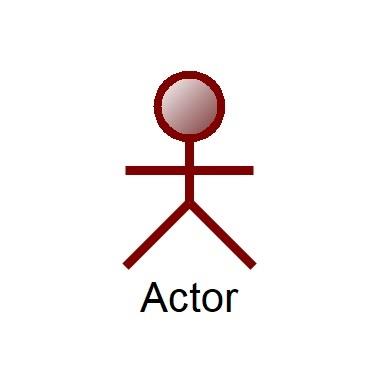


Figure 6: Représentation d'un acteur

Pour notre application, on distingue deux acteurs principaux qui sont :

* **Administrateur** : C’est l’acteur qui est chargé de la plateforme, de la gestion des agents, de la gestion des marques, modèles, catégories et véhicules.
* **Agent**: C’est l’acteur qui s’authentifie pour utiliser la plateforme pour gérer les marques, modèles, catégories et véhicules.



Figure 7: Représentation des acteurs de la plateforme

* + 1. Diagramme de cas d’utilisation global :

Afin de donner une vision globale du comportement fonctionnel de l’application et définir ce qu’elle réalise d’un point de vue externe, nous allons réaliser son diagramme de cas d’utilisation générale et par la suite son diagramme de cas d’utilisation détaillé.

L’ensemble général des fonctionnalités attendues de notre solution est modélisé par le diagramme suivant :



Figure 8: Diagramme du cas d’utilisation global

##### Raffinement du cas d’utilisation « S’authentifier » :

Le raffinement du cas d’utilisation « S’authentifier » est comme suit :



Figure 9: Raffinement du cas d’utilisation « S’authentifier »

* Description textuelle du cas d’utilisation « S’authentifier » :

Ci-dessous la description textuelle du cas d’utilisation « S’authentifier » :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur principal : | Utilisateur |
| Objectif : | Se connecter |
| Pré condition : | L’utilisateur doit être déconnecté de l’application  L’utilisateur doit être déjà inscrit dans l’application |
| Post condition : | Néant |
| Scénario nominal : | 1. L’utilisateur saisie son login et son mot de passe  2. L’utilisateur valide ses données  3. Le système vérifie les données saisies  4. Le système affiche la page demandée (page d’accueil). |
| Exception : | 3. a. Login et/ou mot de passe erroné ou champs vides :  1. Le Système informe l’utilisateur de recommencer la saisie  2. L’enchainement reprend à l’étape 1 |

Tableau 1: Description textuelle du cas d’utilisation « s’authentifier »

##### Raffinement du cas d’utilisation « Gérer Agent » :

Le raffinement du cas d’utilisation « Gérer agent » est comme suit :



Figure 10: Raffinement du cas d’utilisation « Gérer agent »

* Description textuelle du cas d’utilisation «Gérer agent » :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur principal : | Administrateur |
| Objectif : | Gérer les agents |
| Pré condition : | L’administrateur est authentifié |
| Post condition : | Liste des agents affichée |
| Scénario nominal : | 1. L’administrateur consulte l’interface de gestion des agents.  2. Le système affiche la liste des agents.  3. L’administrateur choisit l’opération de mise à jour (Ajout/ modification/suppression).  4. Le système affiche la nouvelle liste d’agent. |
| Scénario alternatif : | 3. a. L’administrateur peut ajouter un nouvel agent à la plateforme.  1. L’administrateur demande l’interface d’ajout d’un agent.  2. Le système affiche l’interface demandée.  3. L’administrateur consulte la liste des agents et confirme l’ajout.  4. b. L’administrateur peut modifier les droits d'accès d’un agent.  1. L’administrateur sélectionne l’agent.  2. L’administrateur gère les modifications.  3. L’administrateur clique sur le bouton « modifier »  4. Le système enregistre la modification.  5. d. L’administrateur peut désactiver un compte agent :  1. L’administrateur sélectionne un agent.  2. L’administrateur clique sur le bouton « Désactiver ».  3. L’administrateurconfirme les modifications  4. Le Système informe l’administrateur que la désactivation est effectuée avec succès. |

Tableau 2: Description textuelle du cas d’utilisation « Gérer agent »

##### Raffinement du cas d’utilisation « Gérer marque » :

Le raffinement du cas d’utilisation « « Gérer marque » » est comme suit :

****

Figure 11: Raffinement du cas d’utilisation « Gérer marque »

* Description textuelle du cas d’utilisation «Gérer marque » :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur principal : | Agent |
| Objectif : | Gérer les marques |
| Pré condition : | L’agent est authentifié |
| Post condition : | La gestion est effectuée avec succès |
| Scénario nominal : | 1. L’agent consulte l’interface de gestion des marques,  2. Le système affiche la liste des marques,  3.L’agent choisit l’opération de mise à jour (Ajout/modification/ désactivation). |
| Scénario alternatif : | 3.a. L’agent peut ajouter une nouvelle marque  1. L’agent clique sur le bouton « ajouter »  2. Le système affiche l’interface d’ajout de la marque  3. L’agent introduit les données de la marque et confirme l’ajout  4. Le système vérifie les données  5. Le système ajout la marque  6. Le système informe l’agent que l’ajout est effectué avec succès  4. b. L’agent peut modifier une marque :  1. L’agent sélectionne la marque de voiture,  2. L’agent gère les modifications  3. L’agent clique sur le bouton «enregistrer».  4. Le système enregistre la modification  4. c. L’agent peut désactiver une marque :  1. L’agent consulte la liste des marques  2. L’agent sélectionne une marque  3. L’agent clique sur la case à cocher « désactiver »  3. L’agent confirme la modification  4. Le Système informe l’agent que la désactivation est effectuée avec succès. |

Tableau 3: Description textuelle du cas d’utilisation « Gérer marque »

##### Raffinement du cas d’utilisation « Gérer modèle » :

Le raffinement du cas d’utilisation « Gérer modèle » est comme suit :



Figure 12: Raffinement du cas d’utilisation « Gérer modèle »

* Description textuelle du cas d’utilisation «Gérer modèle» :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur principal : | Agent |
| Objectif : | Gérer les modèles |
| Pré condition : | L’agent est authentifié |
| Post condition : | La gestion est effectuée avec succès |
| Scénario nominal : | 1. L’agent consulte l’interface de gestion des modèles  2. Le système affiche la liste des modèles  3. L’administrateur choisit l’opération de mise à jour (Ajout/ modification/ désactivation). |
| Scénario alternatif : | 3. a. L’agent peut ajouter un nouveau modèle  1. L’agent clique sur le bouton « ajouter »,  2. Le système affiche l’interface d’ajout du modèle  3. L’agent introduit les données du modèle et confirme l’ajout  4. Le système vérifie les données  5. Le système ajoute le modèle  6. Le système informe l’agent que l’ajout est effectué avec succès.  3. b. L’agent peut modifier un modèle  1. L’agent sélectionne le modèle  2. L’agent gère les modifications  3. L’agent clique sur le bouton « modifier »  4. Le système enregistre la modification.  3. c. L’agent peut désactiver un modèle :  1. L’agent consulte la liste des modèles de voitures  2. L’agent sélectionne un modèle  3. L’agent clique sur la case « désactiver »  4. L’agent confirme la modification  5. Le Système informe l’agent que la désactivation est effectuée avec succès. |

Tableau 4: Description textuelle du cas d’utilisation « Gérer modèle »

##### Raffinement du cas d’utilisation « Gérer catégorie » :

Le raffinement du cas d’utilisation  « Gérer catégorie »  est comme suit :



Figure 13: Raffinement du cas d’utilisation « Gérer catégorie »

* Description textuelle du cas d’utilisation «Gérer catégorie» :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur principal : | Agent |
| Objectif : | Gérer les catégories |
| Pré condition : | L’agent est authentifié |
| Post condition : | La gestion effectuée avec succès |
| Scénario nominal : | 1. L’agent consulte l’interface de gestion des catégories  2. Le système affiche la liste des catégories  3. L’agent choisit l’opération de mise à jour (Ajout/ modification/ désactivation). |
| Scénario alternatif : | 3. a. L’agent peut ajouter une nouvelle catégorie :  1. L’agent clique sur le bouton « ajouter »  2. Le système affiche l’interface d’ajout de la catégorie  3. L’agent introduit les données de la catégorie et confirme l’ajout  4. Le système vérifie les données  5. Le système ajoute la catégorie.  6. Le système informe l’agent que l’ajout est effectué avec succès.  3. b. L’agent peut modifier une catégorie :  1. L’agent sélectionne la catégorie  2. L’agent gère les modifications  3. L’agent clique sur le bouton « modifier »  4. Le système enregistre les modifications  3. c. L’agent peut désactiver une catégorie :  1. L’administrateur consulte la liste des catégories  2. L’agent sélectionne une catégorie  3. L’administrateur clique sur la case à cocher « désactiver »  4. L’agent confirme la modification  5. Le Système informe l’agent que la désactivation est effectuée avec succès. |

Tableau 5: Description textuelle du cas d’utilisation  « Gérer catégorie »

##### Raffinement du cas d’utilisation « Gérer véhicule» :

Le raffinement du cas d’utilisation « Gérer véhicule » est comme suit :



Figure 14: Raffinement du cas d’utilisation « Gérer véhicule »

* Description textuelle du cas d’utilisation «Gérer véhicule» :

|  |  |
| --- | --- |
| Cas d’utilisation | S’authentifier |
| Acteur principal : | Agent |
| Objectif : | Gérer les véhicules |
| Pré condition : | L’agent est authentifié |
| Post condition : | La gestion effectuée avec succès |
| Scénario nominal : | 1. L’agent consulte l’interface de gestion des véhicules  2. Le système affiche la liste des véhicules  3. L’agent choisit l’opération de mise à jour (Ajout/ modification/ désactivation). |
| Scénario alternatif : | 3. a. L’agent peut ajouter un nouveau véhicule :  1. L’agent clique sur le bouton (ajouter)  2. Le système affiche l’interface d’ajout du véhicule  3. L’agent introduit les données du véhicule et confirme l’ajout  4. Le système vérifie les données  5. Le système ajoute le véhicule.  6. Le système informe l’agent que l’ajout est effectué avec succès.  3. b. L’agent peut modifier un véhicule :  1. L’agent sélectionne le véhicule  2. L’agent gère les modifications  3. L’agent clique sur le bouton « modifier »  4. Le système enregistre les modifications.  3. c. L’agent peut désactiver un véhicule :  1. L’agent consulte la liste des véhicules  2. L’agent sélectionne un véhicule  3. L’administrateur clique sur la case à cocher « désactiver »  4. L’agent confirme la modification  5. Le système informe l’agent que la désactivation est effectuée avec succès. |

Tableau 6: Description textuelle du cas d’utilisation « Gérer véhicule »

1. Diagramme de séquence :

Un diagramme de séquence représente la succession chronologique des opérations réalisées par un acteur. Il indique les objets que l’acteur va manipuler et les opérations qui font passer d’un objet à l’autre.

* 1. Diagramme de séquence du cas d’utilisation « S’authentifier »

Le diagramme de séquence du cas d’utilisation « S’authentifier » est comme suit :



Figure 15: Diagramme de séquence authentification

* 1. Diagramme de séquence du cas d’utilisation « Gérer agent »

Le diagramme de séquence du cas d’utilisation « Gérer agent » est comme suit :



Figure 16: Diagramme de séquence gérer agents

* 1. Diagramme de séquence du cas d’utilisation « Gérer marque »

Le diagramme de séquence du cas d’utilisation « Gérer marque » est comme suit :

e

Figure 17: Diagramme de séquence gérer marques

* 1. Diagramme de séquence du cas d’utilisation « Gérer modèle »

Le diagramme de séquence du cas d’utilisation « Gérer modèle » est comme suit :



Figure 18: Diagramme de séquence gérer modelés

* 1. Diagramme de séquence du cas d’utilisation « Gérer catégorie »

Le diagramme de séquence du cas d’utilisation « Gérer catégorie » est comme suit :



Figure 19: Diagramme de séquence gérer catégories

* 1. Diagramme de séquence du cas d’utilisation « Gérer véhicule »

Le diagramme de séquence du cas d’utilisation « Gérer véhicule » est comme suit :



Figure 20: Diagramme de séquence gérer véhicules

1. Conclusion :

Dans ce chapitre, nous avons présenté les exigences et les besoins de notre application. Pour illustrer les interactions entre les différents acteurs et le système, nous avons eu recours aux diagrammes de cas d'utilisation et les diagrammes de séquence préconisés par UML.

Chapitre 3 : Conception détaillée

1. Introduction :

Il s’agit de la présentation d’une description logique de la façon dont le système va fonctionner.

Ce chapitre contient une présentation des entités, le diagramme de classes et les diagrammes de séquences détaillées

1. Extractions des différentes entités :

Le tableau ci-dessous décrit les entités utilisées dans la réalisation de l’application.

|  |  |
| --- | --- |
| Entités | Description |
| Utilisateur | Les informations de l’administrateur |
| Marque | Liste des marques |
| Modèle | Liste des modèles |
| Catégorie | Liste des catégories |
| Véhicule | Liste des véhicules |

Tableau 7 : Les entités

1. Extraction des attributs des différentes entités :

Le tableau ci-dessous décrit les différents attributs de l’entité « utilisateur »

|  |  |  |  |
| --- | --- | --- | --- |
| Entités | Les attributs | Désignations | Type |
| Utilisateur | userId | Identification de l’utilisateur | Int |
| userName | Nom de l’utilisateur | String |
| login | Le login de l’utilisateur | String |
| password | Le mot de passe de l’utilisateur | String |
| enabled | Le mot de passe de l’utilisateur | Booléen |
| dateModifyPassword | La date de modification de mot de passe | Date |
| nbrAttemptsConnection | Le nombre | Int |
| local | Le pays de l'utilisateur | String |
| userLocked | L'état de l'utilisateur | Booléen |

Tableau 8 : entité Utilisateur

Le tableau ci-dessous décrit les différents attributs de l’entité « marque »

|  |  |  |  |
| --- | --- | --- | --- |
| Entités | Les attributs | Désignations | Type |
| Marque | brandId | Identification de la marque | Int |
| brandName | Nom de la marque | String |
| brandLogo | Logo de la marque | Byte |
| Active | L’état de la marque | Booléen |

Tableau 9 : entité marque

Le tableau ci-dessous décrit les différents attributs de l’entité « modèle »

|  |  |  |  |
| --- | --- | --- | --- |
| Entités | Les attributs | Désignations | Type |
| Modèle | modelId | Identification du modèle | Int |
| modelName | Nom du modèle | String |
| modelYear | L’année de modèle | Int |
| brandId | Identification de la marque modèle | Int |
| categoryId | Identification des catégories du modèle | Int |
| tankCapacity | Capacité du réservoir | Int |
| tankGraduationLevel | Niveau de graduation du réservoir | Int |
| servicesPeriod | Intervalle de révision | Int |
| payload | Charge utile | Int |
| seatsNumber | nombre de sièges | Int |
| accessories | Accessoires du modèle | String |
| active | L’état du modèle | Booléen |

Tableau 10 : entité modèle

Le tableau ci-dessous décrit les différents attributs de l’entité « catégorie »

|  |  |  |  |
| --- | --- | --- | --- |
| Entités | Les attributs | Désignations | Type |
| Catégorie | categoryId | Identification de la catégorie | Int |
| categoryCode | Code de la catégorie | String |
| categoryName | Nom de la catégorie | String |
| type | Type de la catégorie | String |
| group | Groupe de la catégorie | String |
| transmission | Transmission de la vitesse | String |
| ranking | Classement de la catégorie | String |
| upperUpgrade | Mise à niveau supérieure de la catégorie | String |
| numberOfSmallSuitCases | Nombre de petites valises | Int |
| numberOfBigSuitCases | Nombre de grandes valises | Int |
| drivingLicenseCategoryId | Identification de la catégorie de permis de conduire | String |
| picture | Une image de catégorie | Byte |
| active | L’état de catégorie | Booléen |
| master | Sert à définir la valeur par défaut du filtre de la catégorie du véhicule dans des futurs écrans | Booléen |

Tableau 11 : entité catégories

Le tableau ci-dessous décrit les différents attributs de l’entité « véhicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Entités | Les attributs | Désignations | Type |
| Véhicule | vehiculeId | Identification du véhicule | Int |
| agencyId | Identification de l’agence | Int |
| brandId | Identification de la marque | Int |
| modelId | Identification du modèle | Int |
| categoryId | Identification de la catégorie | Int |
| fuelType | Type du carburant | String |
| registrationNumber | Numéro d'enregistrement | Int |
| identificationNumber | Numéro d'identification | Int |
| startOfServiceDate | Date de début de service | Date |
| expectedEndOfServiceDate | Date prévue de la fin du service | Date |
| active | L’état du véhicule | Booléen |
| disablingReason | Raison d’immobilisation | String |
| multiAgency | Définir si le véhicule peut être prêté par une autre agence | Booléen |
| oneWay | Définir si le véhicule peut être loué pour un aller simple (sans retour) | Booléen |

Tableau 12 : entité véhicule

1. Diagramme de classes :

C’est un diagramme représentant la structure statique d’un modèle, à savoir les éléments (classes et types), la structure interne des éléments et leurs relations les uns par rapport aux autres. Les diagrammes de classe représentent les classes, les paquetages ou les interfaces connectés par des relations statiques.

Le diagramme de classe de notre application est décrit comme suit :



Figure 21 : diagramme des classes

4.1 Modèle relationnel :

* Utilisateur (userId, userName, login, password, enabled, dateModifyPassword, nbrAttemptsConnection, local, userLocked)
* Marque (BrandId, BrandName, Active, BrandLogo)
* Modèle (ModelId, ModelName, ModelYear, #BrandId ,# CategoryId, TankCapacity, TankGraduationLevel, ServicesPeriod, Payload, SeatsNumber, Accessoires, Active)
* Catégorie (CategoryId, CategoryCode, CategoryName, Type, Ranking, UpperUpgrade, NumberOfSmallSuitCases, NumberOfBigSuitCases, DrivingLicenceCategoryId, Picute, Active, IsMaster)
* Véhicule (VehiculeId, AgencyId, #BrandId, #ModelId, #CategoryId, FuelType, RegistrationNumber, StartOfServiceDate, ExpectedEndOfServiceDate, Active, MultiAgency, OneWay)

1. Diagramme d’interaction :

L’expression diagramme d’interactions englobe principalement deux types de diagrammes UML spécialisés, qui peuvent servir tous les deux à exprimer des interactions de messages similaires :

* Les diagrammes de communication.
* Les diagrammes de séquence.
* Le diagramme d’interaction.
* Le diagramme de temps.

Pour mon projet j’ai choisi le diagramme de séquence comme diagramme d’interaction. Par rapport aux diagrammes de séquence système du chapitre précédent, nous allons remplacer le système vu comme une boite noire par un ensemble d’objets en interaction. Pour cela, nous utiliserons encore dans ce chapitre les trois types de classes

D’analyse, à savoir :

* Les dialogues.
* Les contrôles.
* Les entités.
  1. Diagramme de séquence du cas d’utilisation « S’authentifier »

Nous présentons le diagramme de séquence du cas « s’authentifier » pour illustrer les interactions d’un utilisateur avec le système.



Figure 22 : diagramme de séquence du cas d’utilisation « s’authentifier »

L’authentification doit être effectuée avant chaque traitement, L’utilisateur fournit au système son login et mot de passe. Le système consulte la base de donnéesvia le service de l’authentification dès l’authentification assurée, la page d’accueil sera chargée.

5.2 Diagramme de séquence du cas d’utilisation « Gérer marques »

Dans cette section, nous décrivons les principales opérations du cas d’utilisation « gérer marques » sous la forme de diagramme de séquence.

* Ajout d’une nouvelle marque :



Figure 23 : Diagramme de séquence d'ajout d'une marque de cas gérer marques

* Modification d’une marque :



Figure 24 : Diagramme de séquence de modification d'une marque de cas gérer marques

* Désactivation d’une marque :



Figure 25 : Diagramme de séquence de désactivation d'une marque de cas gérer marques

La tâche s'active par l'accès de l'utilisateur à l'interface d'accueil à partir de laquelle il va lancer une demande d'accès à l'interface dédiée à la gestion des marques.

Afin de créer une nouvelle marque l'utilisateur doit dans un premier temps cliquer sur le bouton dédié à l’ajout (une icône) et remplir le formulaire.

Quand il finit il valide en cliquant sur le bouton « enregistrer ».

La nouvelle marque sera ajoutée dans le tableau qui s'affiche dans l'interface de gestion des marques.

L'utilisateur peut aussi modifier une marque.

Pour ceci il doit cliquer sur l'icône dédiée à la modification et mettre à jour le formulaire puis il clique sur le bouton ''enregistrer'' pour sauvegarder les modifications.

Pour désactiver une marque l'utilisateur doit décocher la case à cocher nommée « active ».

5.3 Diagramme de séquence du cas d’utilisation « Gérer modèles »

Dans cette section, nous décrivons les principales opérations du cas d’utilisation « gérer modèles » sous la forme de diagramme de séquence.

* Ajout d’un nouveau modèle :



Figure 26 : Diagramme de séquence d'ajout d'un modèle de cas gérer modèles

* Modification d’un modèle :



Figure 27 : Diagramme de séquence de modification d'un modèle de cas gérer modèles

* Désactivation d’un modèle :



Figure 28 : Diagramme de séquence de désactivation d'un modèle de cas gérer modèles

La tâche s'active par l'accès de l'utilisateur à l'interface d'accueil à partir de laquelle il va lancer une demande d'accès à l'interface dédiée à la gestion des modèles.

Afin de créer un nouveau modèle l'utilisateur doit dans un premier temps cliquer sur le bouton dédié à l’ajout (une icône) et remplir le formulaire.

Quand il finit il valide en cliquant sur le bouton « enregistrer ».

Le nouveau modèle sera ajouté dans le tableau qui s'affiche dans l'interface de gestion des modèles.

L'utilisateur peut aussi modifier un modèle.

Pour ceci il doit cliquer sur l'icône dédiée à la modification et mettre à jour le formulaire puis il clique sur le bouton « enregistrer » pour sauvegarder les modifications.

Pour désactiver un modèle l'utilisateur doit décocher la case à cocher nommée « active ».

5.4 Diagramme de séquence du cas d’utilisation « Gérer catégories »

Dans cette section, nous décrivons les principales opérations du cas d’utilisation « gérer catégories » sous la forme de diagramme de séquence.

* Ajout d’une nouvelle catégorie :



Figure 29 : Diagramme de séquence d'ajout d'une catégorie de cas gérer catégories

* Modification d’une catégorie :



Figure 30 : Diagramme de séquence de modification d'une catégorie de cas gérer catégories

* Désactivation d’une catégorie :



Figure 31 : Diagramme de séquence de désactivation d'une catégorie de cas gérer catégories

La tâche s'active par l'accès de l'utilisateur à l'interface d'accueil à partir de laquelle il va lancer une demande d'accès à l'interface dédiée à la gestion des catégories.

Afin de créer une nouvelle catégorie l'utilisateur doit dans un premier temps cliquer sur le bouton dédié à l’ajout (une icône) et remplir le formulaire.

Quand il finit il valide en cliquant sur le bouton « enregistrer ».

La nouvelle catégorie sera ajoutée dans le tableau qui s'affiche dans l'interface de gestion des catégories.

L'utilisateur peut aussi modifier une catégorie.

Pour ceci il doit cliquer sur l'icône dédiée à la modification et mettre à jour le formulaire puis il clique sur le bouton « enregistrer » pour sauvegarder les modifications.

Pour désactiver une catégorie l'utilisateur doit décocher la case à cocher nommée « active ».

5.5 Diagramme de séquence du cas d’utilisation « Gérer véhicule »

Dans cette section, nous décrivons les principales opérations du cas d’utilisation « gérer véhicules » sous la forme de diagramme de séquence.

* Ajout d’un nouveau véhicule :



Figure 32 : Diagramme de séquence d'ajout d'un véhicule de cas gérer véhicules

* Modification d’un véhicule :



Figure 33 : Diagramme de séquence de modification d'un véhicule de cas gérer véhicules

* Désactivation d’un véhicule :



Figure 34 : Diagramme de séquence de désactivation d'un véhicule de cas gérer véhicules

La tâche s'active par l'accès de l'utilisateur à l'interface d'accueil à partir de laquelle il va lancer une demande d'accès à l'interface dédiée à la gestion des véhicules.

Afin de créer un nouveau véhicule l'utilisateur doit dans un premier temps cliquer sur le bouton dédié à l’ajout (une icône) et remplir le formulaire.

Quand il finit il valide en cliquant sur le bouton « enregistrer ».

Le nouveau véhicule sera ajouté dans le tableau qui s'affiche dans l'interface de gestion des véhicules.

L'utilisateur peut aussi modifier un véhicule.

Pour ceci il doit cliquer sur l'icône dédiée à la modification et mettre à jour le formulaire puis il clique sur le bouton « enregistrer » pour sauvegarder les modifications.

Pour désactiver un véhicule l'utilisateur doit décocher la case à cocher nommée « active ».

1. Conclusion :

La modélisation conceptuelle est une étape très importante dans la conception de tout système et son élaboration doit être bien méticuleuse afin d’aboutir à un résultat pouvant engendrer une étape de réalisation pertinente que nous représentons dans le chapitre suivant.

Chapitre 4 : Réalisation

1. Introduction :

La réalisation est l’une des étapes de cycles de vie d'un projet, aussi importante que la conception. Cette étape constitue la phase d'achèvement et aboutissement du site. Dans ce qui suit, nous commencerons par présenter l'environnement matériel et logiciel utilisé et par la suite nous allons présenter les interfaces de site.

1. Environnement matériel :

L’environnement matériel utilisé lors de la réalisation de l’application, est :

* PC : Dell, processeur : Intel® Core™ i7-7500U, RAM : 8Go
* Disque dur : 1 To
* Système d’exploitation : Windows 10 professionnel 64 bits

1. Technologies utilisées :

Cette partie se consacre à la présentation des langages de développement et les technologies utilisées pour réaliser ce projet.

* Back-End

Cette sous-partie est basée sur les langages de développement et les technologies utilisées pour réaliser la partie Back-End de ce projet



**Spring :**

Spring est un framework libre dédié à la construction et la définition de l'infrastructure d'une application java, en facilitant le développement et les tests. En 2004, Rod Johnson a écrit le livre Expert One-on-One J2EE Design and Development5 qui explique les raisons de la création de Spring.



**Spring Boot :**

Spring Boot est un framework libre bien connu par des développeurs Java pour les nombreuses fonctionnalités qu’il apporte sur les aspects web, sécurité ou encore accès aux données dans le cadre du développement d’une application java.

****

**Spring Data :**

Spring Data a pour mission de fournir un modèle de programmation Spring familier et cohérent pour l'accès aux données tout en conservant les caractéristiques spéciales du data store.

Il facilite l'utilisation des technologies d'accès aux données, des bases de données relationnelles et non relationnelles, des cadres de réduction de la carte et des services de données en nuage. C'est un projet parapluie qui contient de nombreux sous-projets spécifiques à une base de données donnée. Les projets sont développés en travaillant avec de nombreuses entreprises et développeurs qui sont derrière ces technologies passionnantes.



**Spring MVC :**

Spring MVC « Model View Controller » offre une meilleure structuration du tiers de présentation des applications Java EE

Le framework Spring offre une implémentation innovante du modèle MVC par le biais d’un framework nommé Spring MVC, qui profite des avantages de l’injection de dépendances et qui, depuis la version 2.5, offre une intéressante flexibilité grâce aux annotations Java 5. Ce module permet dès lors de s’abstraire de l’API Servlet de Java EE, les informations souhaitées étant automatiquement mises à disposition en tant que paramètres des méthodes des contrôleurs.

****

**Spring Security :**

Spring Security est un cadre d'authentification et de contrôle d'accès puissant et hautement personnalisable. C'est la norme de facto pour la sécurisation des applications Spring.



**Hibernate :**

Hibernate est un framework open source gérant la persistance des objets en base de données relationnelle, il est adaptable en termes d'architecture, il peut donc être utilisé aussi bien dans un développement client lourd, que dans un environnement web léger de type Apache Tomcat ou dans un environnement Java EE complet : WebSphere, JBoss Application Server et Oracle WebLogic Server.

Hibernate apporte une solution aux problèmes d'adaptation entre le paradigme objet et les SGBD en remplaçant les accès à la base de données par des appels à des méthodes objet de haut niveau.

* Front End

Cette sous-partie est basée sur les langages de développement et les technologies utilisées pour réaliser la partie Front-End de ce projet



**Angular 5 :**

Angular 5 est un framework d'application Web open-source basé sur TypeScript et dirigé par l'équipe Angular de Google et par une communauté de particuliers et d'entreprises.

Angular5 est une réécriture complète de la même équipe qui a construit AngularJS.



**Angular Material :**

Est un ensemble de règles de design qui s'appliquent à l'interface graphique des logiciels et applications. Il implémente la spécification Material Design de Google.

**ag-Grid :**

ag-Grid a évolué d'un projet parallèle à devenir la principale DataGrid JavaScript sur le marché.

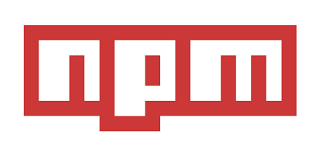
Il s’agit d’une entreprisequi offre **ag-Grid communauté** : un projet gratuit et open source qui offre des performances de grille de classe mondiale, et **ag-Grid Enterprise** est une offre commerciale sous licence qui a été largement adoptée en facilitant la réalisation de la mission.



**Angular CLI :**

Angular-Cli « Angular Command Line Interface » est un outil en ligne de commande puissant qui permet de créer un projet Angular contenant un ensemble plus que complet d'outils en vogue dans le monde JavaScript.Car vous n'avez pas besoin de passer du temps à installer et à configurer toutes les dépendances requises et à tout câbler ensemble.

Il utilise l'exécutable nommé ng pour réaliser ses différentes fonctionnalités offertes.



**Npm :**

Npm « Node Package Manager » est le gestionnaire de paquets officiel de Node.js. Depuis la version 0.6.3 de Node.js, npm fait partie de l'environnement et donc installé par défaut.

Npm fonctionne avec un terminal et gère les dépendances pour une application.

****

**Node.js :**

Node.js est une plateforme logicielle libre et événementielle en JavaScript orientée vers les applications réseau qui doivent pouvoir monter en charge.

Parmi les modules de Node.js, on retrouve http qui permet le développement de serveur HTTP. Il est donc possible de se passer de serveurs web tels que Nginx ou Apache lors du déploiement de sites et d'applications web développés avec Node.js.

Concrètement, Node.js est un environnement bas niveau permettant l’exécution de JavaScript côté serveur.

Node.js est de plus en plus populaire comme plateforme serveur4, elle est utilisée par Groupon5, Vivaldi, SAP6, LinkedIn7,8, Microsoft9,10, Yahoo!11, Walmart12, Rakuten, Sage et PayPal13,14.

## Environnement logiciel de développement :

Pour la réalisation de notre projet nous avons eu recours à plusieurs outils et langages de programmation. Une liste détaillée est recensée dans ce qui suit

****

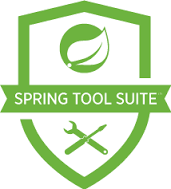
**Apache Tomcat :**

Apache Tomcat est un conteneur web libre de servlets et JSP Java EE. Issu du projet Jakarta, c'est un des nombreux projets de l’Apache Software Foundation. Il implémente les spécifications des servlets et des JSP du Java Community Process, est paramétrable par des fichiers XML et des propriétés, et inclut des outils pour la configuration et la gestion. Il comporte également un serveur HTTP.



**PostgreSQL :**

PostgreSQL est un système de gestion de base de données relationnelle et objet (SGBDRO). C'est un outil libre disponible selon les termes d'une licence de type BSD. Ce système est concurrent d'autres systèmes de gestion de base de données, qu'ils Soient libres (comme MariaDB, MySQL et Firebird), ou propriétaires (comme Oracle, Sybase, DB2, Informix et Microsoft SQL Server). Comme les projets libres Apache et Linux, PostgreSQL n'est pas contrôlé par une seule entreprise, mais est fondé sur une communauté mondiale de développeurs et d'entreprises.



**Spring Tool Suite :**

Spring Tool Suite est un environnement de développement basé sur Eclipse qui est personnalisé pour le développement d'applications Spring. Il fournit un environnement prêt à l'emploi pour implémenter, déboguer, exécuter et déployer vos applications Spring, y compris les intégrations pour Pivotal tc Server, Pivotal Cloud Foundry, Git, Maven, AspectJ et vient s'ajouter aux dernières versions d'Eclipse.



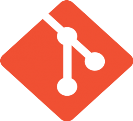
**Apache Maven :**

Apache Maven est un outil de gestion et d'automatisation de production des projets logiciels Java en général et Java EE en particulier. Maven est géré par l'organisation Apache Software Foundation. L'outil était précédemment une branche de l'organisation Jakarta Project.

Maven utilise un paradigme connu sous le nom de Project Object Model « POM » afin de décrire un projet logiciel, ses dépendances avec des modules externes et l'ordre à suivre pour sa production. Il est livré avec un grand nombre de tâches pré-définies, comme la compilation de code Java ou encore sa modularisation.

**Postman :**

Une application moderne est construite sur des APIs, qui permettent de construire et de tester rapidement des requêtes HTTP.

****

**Git :**

Assurer un versionnage (équivalent francophone de versionning) du code de notre application est nécessaire pour garder une trace de notre travail afin de retrouver facilement une ancienne version du code développé. Pour cela, nous avons employé Git c’est un outil qui se veut simple et performant, dont la principale tâche est de gérer l’évaluation de contenu d’une arborescence.



**WebStorm :**

WebStorm est un IDE pour les langages Web (HTML, CSS et JavaScript), développé par l'entreprise JetBrains et basé sur la plateforme IntelliJ IDEA.

1. Architecture technique du projet :

L’architecture technique désigne la structure générale inhérente à un système informatique, l'organisation des différents éléments du système et des relations entre les éléments. Cette structure se fait suite à un ensemble de décisions stratégiques prises durant la conception de tout ou une partie du système informatique, par l'exercice d'une discipline technique et industrielle du secteur informatique.

L’adoption d’une bonne architecture en couches facilite la compréhension des besoins et des fonctions. En effet, celle-ci assure un niveau d’abstraction qui facilite la compréhension d’un système informatique puisque les détails ne sont pas représentés. D’une part elle assure la réutilisation des composants, d’autre part elle nous permet de dénombrer les composants du système informatique et de modéliser les relations et les interactions entre elles.

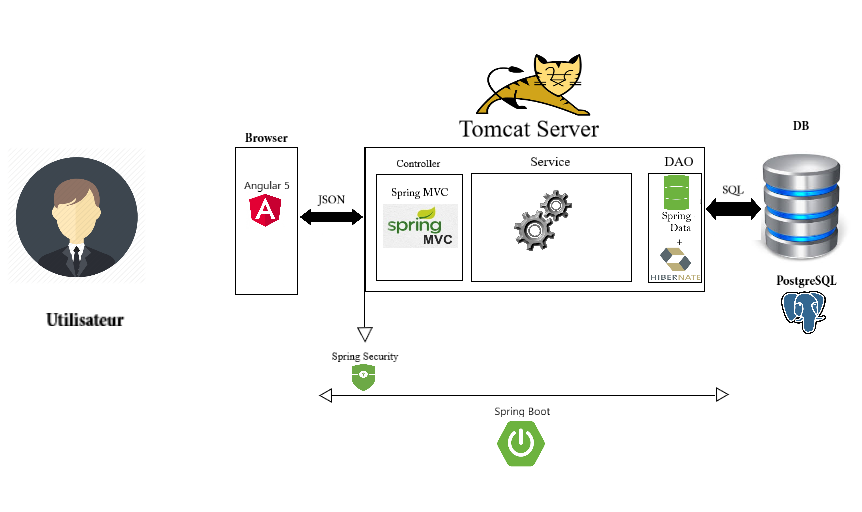
****

Figure 35 : Architecture technique de projet

Cette figure représente le fonctionnement de notre application : les utilisateurs sont interconnectés à une application Front-End développé avec la technologie Angular5.

Cette application est connectée à une application Back-End déployée sur le serveur Tomcat qui est développé avec la technologie Spring Boot, L’échange des données se fait sous format JSON.

L'application Back-End est connectée à la base de données avec les technologies Spring Data et Hibernate pour manipuler les données.

1. Interfaces réalisées :

* Interface d’authentification

C’est l’interface dans laquelle l’agent ou l’administrateur s’authentifie en introduisant le login et le mot de passe.

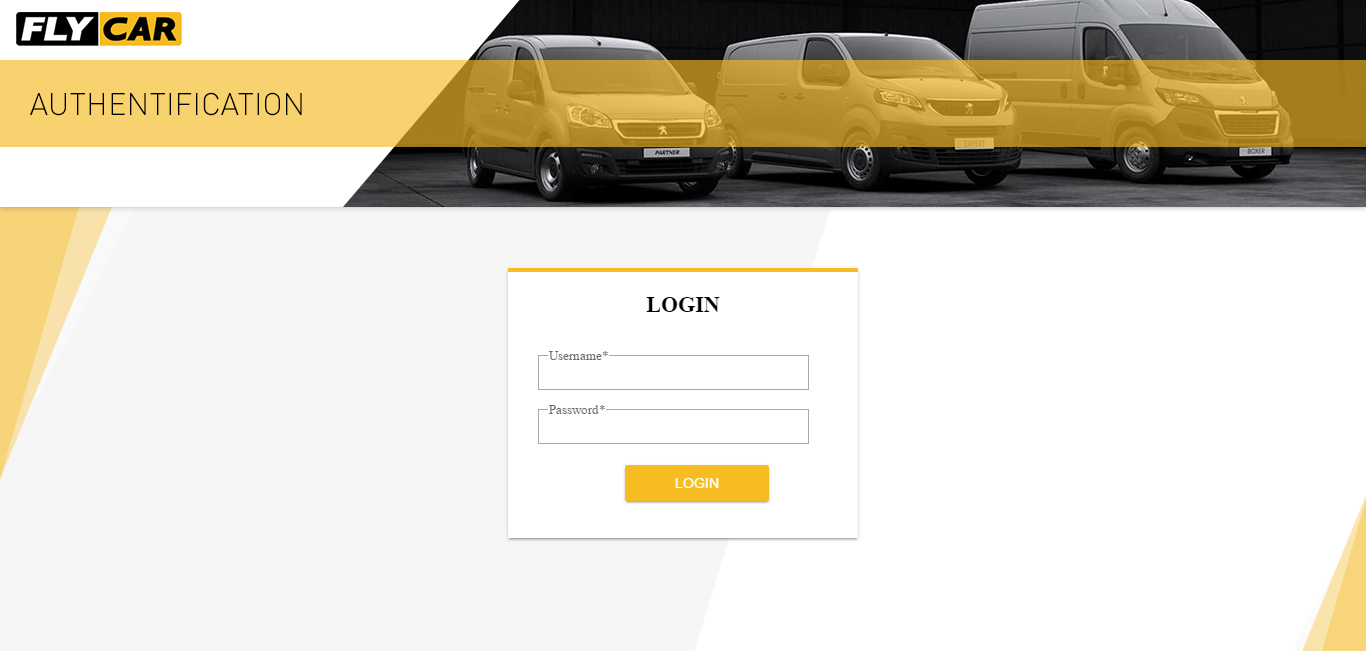


Figure 37 : Interface d’authentification

L’utilisateur va saisir son login et mot de passe puis, dés qu’il clique sur le bouton « Login » l’application FrontEnd va envoyer une requête http a l’application BackEnd, donc cette dernière va faire le traitement nécessaire et va envoyer une requête SQL à la base de données. Si le login et le mot de passe sont corrects une requête sera envoyée à l’application BackEnd contenant toutes les informations de cet utilisateur si non il n’y aura aucun retour.

Au cas où l’utilisateur existe l’application BackEnd va envoyer une requête http avec toutes les informations de cet utilisateur et la page d’accueil sera chargée sinon une erreur s’affiche et la page login sera actualisée.

Cette démarche sera schématisée dans la figure ci-dessous :

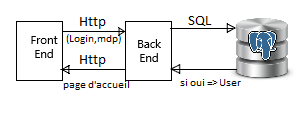


Figure 39 : schéma de login

* Interface de gestion des marques

C’est l’interface dédiée à la gestion des marques, via cette page l’agent ou l’administrateur trouve tous les marques insérées dans la base de données afin de pouvoir les gérer.

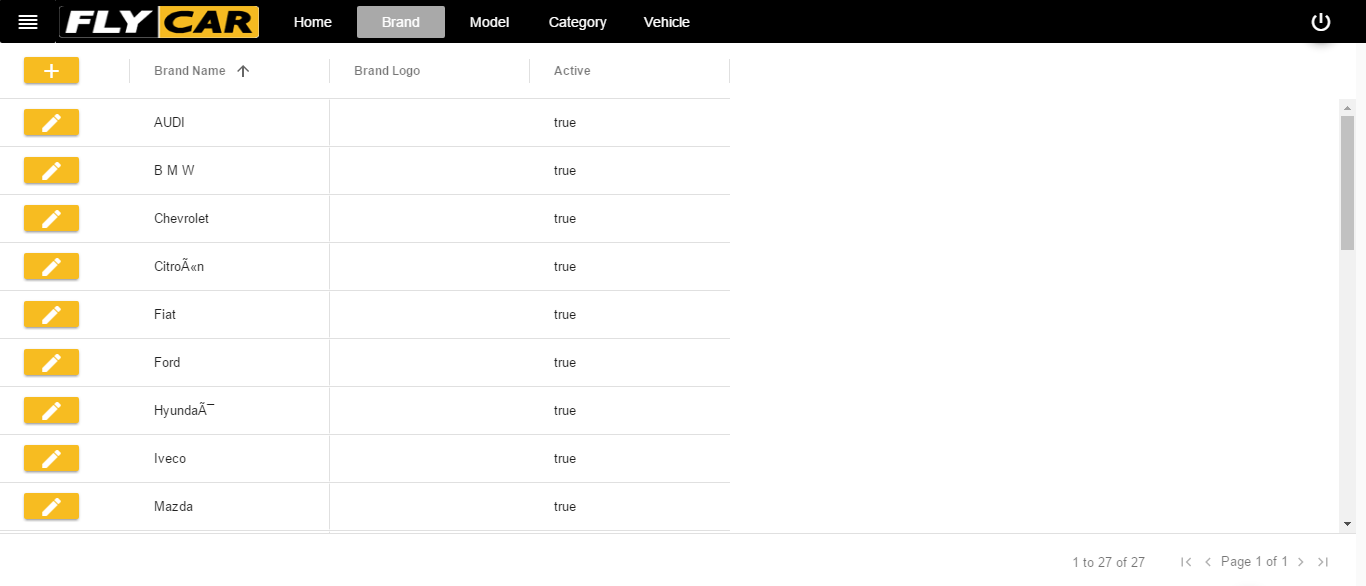


Figure 40 : interface marques

Après avoir s’authentifier l’utilisateur va choisir le bouton Marque dans le menu. Il sera redirigé vers l’interface de gestion des marques. Un tableau contenant toutes les marques enregistrées dans la base de données sera affiché.

A partir de ce tableau l’utilisateur aura la main pour ajouter, modifier ou désactiver une marque.

* Interface d’ajout d’une marque

C’est l’interface qui va permettre à l’agent ou l’administrateur d’ajouter une marque de véhicule qui sera inséré dans la base de données.

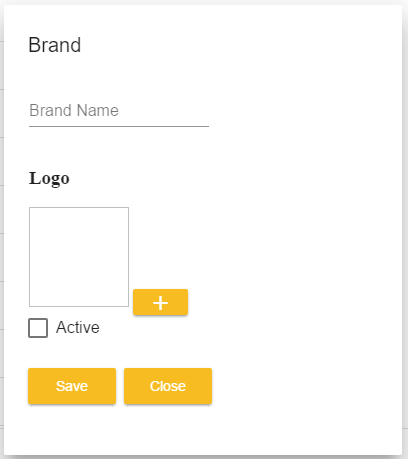


Figure 41 : interface d’ajout d’une marque

Cette interface sera affichée après avoir cliqué sur l’icône d’ajout d’une marque.

L’utilisateur va remplir les champs du formulaire ensuite il valide en cliquant sur le bouton « enregistrer ». Une requête http sera envoyée à l’application BackEnd contenant toutes les informations de cette marque, celle-ci va traiter les informations appropriées et envoyer une requête SQL a la base de données afin de l’enregistrer.

L’utilisateur sera redirigé vers l’interface de gestion des marques dans laquelle la nouvelle marque sera affichée.

* Interface de gestion des modèles

C’est l’interface dédiée à la gestion des modèles, via cette page l’agent ou l’administrateur trouve tous les modèles insérés dans la base de données afin de pouvoir les gérer.

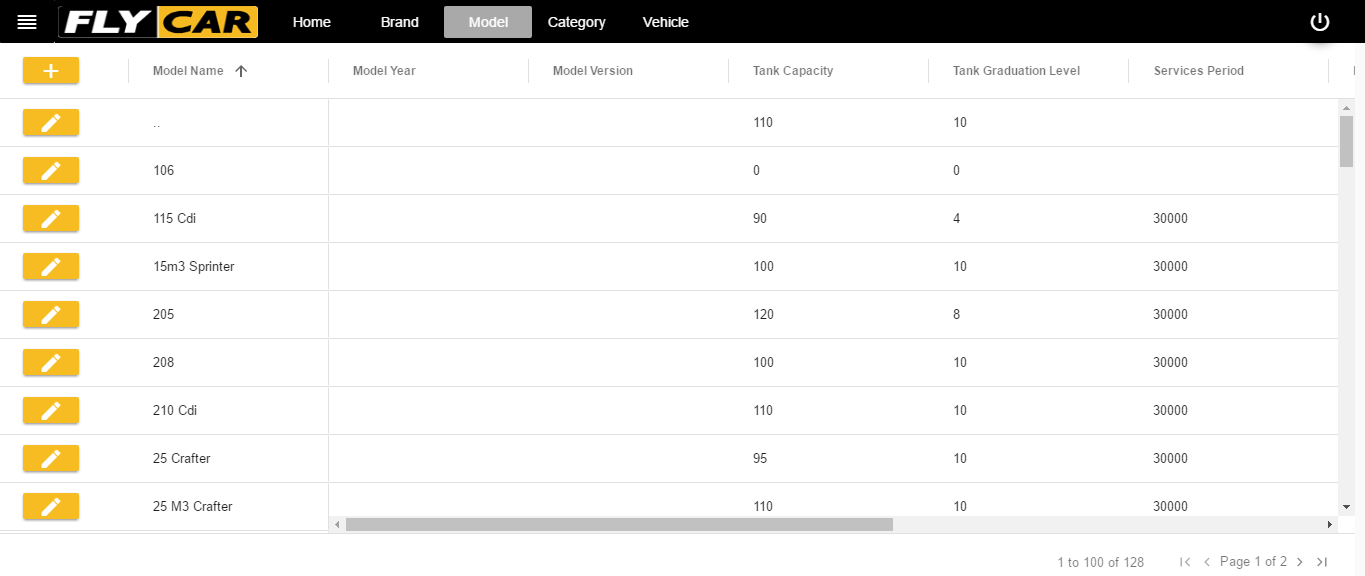


Figure 42 : interface modèles

Après avoir s’authentifier l’utilisateur va choisir le bouton Modèle dans le menu. Il sera redirigé vers l’interface de gestion des modèles. Un tableau contenant toutes les modèles enregistrées dans la base de données sera affiché.

A partir de ce tableau l’utilisateur aura la main pour ajouter, modifier ou désactiver un modèle.

* Interface d’ajout d’un modèle

C’est l’interface qui va permettre à l’agent ou l’administrateur d’ajouter un modèle de véhicule qui sera inséré dans la base de données.

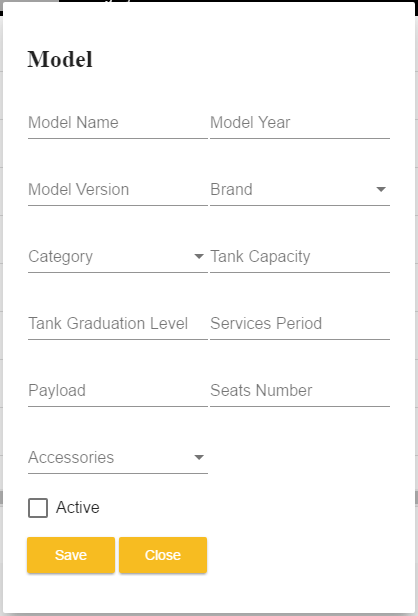


Figure 43 : interface d’ajout d’un modèle

Cette interface sera affichée après avoir cliqué sur l’icône d’ajout d’un modèle.

L’utilisateur va remplir les champs du formulaire ensuite il valide en cliquant sur le bouton « enregistrer ». Une requête http sera envoyée à l’application BackEnd contenant toutes les informations de ce modèle, celle-ci va traiter les informations appropriées et envoyer une requête SQL a la base de données afin de l’enregistrer.

L’utilisateur sera redirigé vers l’interface de gestion des modèles dans laquelle le nouveau modèle sera affiché.

* Interface de gestion des catégories

C’est l’interface dédiée à la gestion des catégories, via cette page l’agent ou l’administrateur trouve tous les catégories insérées dans la base de données afin de pouvoir les gérer.

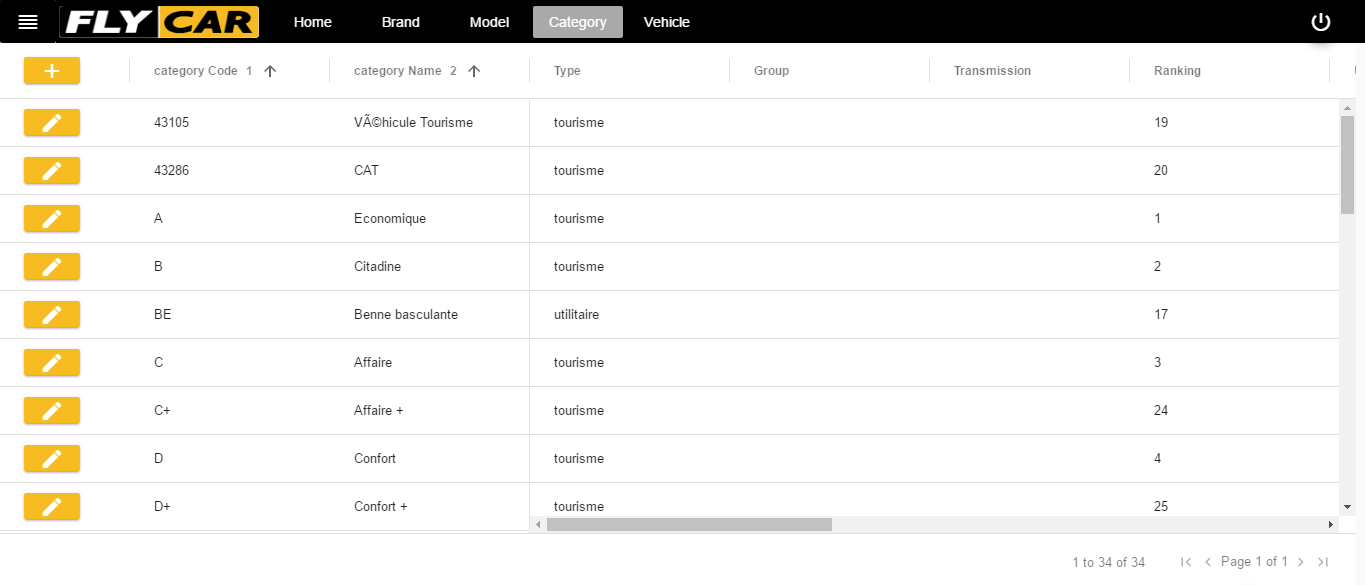


Figure 44 : interface catégories

Après avoir s’authentifier l’utilisateur va choisir le bouton Catégorie dans le menu. Il sera redirigé vers l’interface de gestion des catégories. Un tableau contenant toutes les catégories enregistrées dans la base de données sera affiché.

A partir de ce tableau l’utilisateur aura la main pour ajouter, modifier ou désactiver une catégorie.

* Interface d’ajout d’une catégorie

C’est l’interface qui va permettre à l’agent ou l’administrateur d’ajouter un modèle de véhicule qui sera inséré dans la base de données.

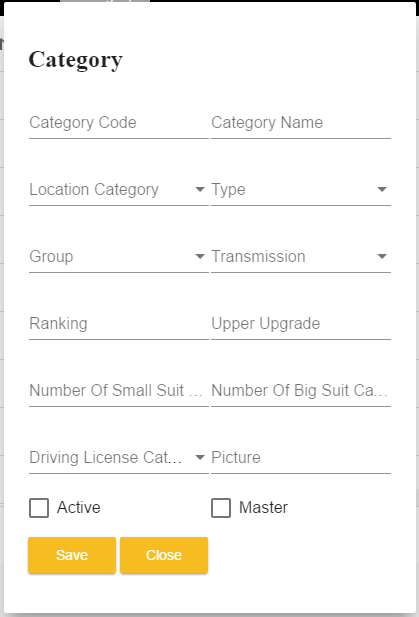


Figure 45 : interface d’ajout d’une catégorie

Cette interface sera affichée après avoir cliqué sur l’icône d’ajout d’une catégorie.

L’utilisateur va remplir les champs du formulaire ensuite il valide en cliquant sur le bouton « enregistrer ». Une requête http sera envoyée à l’application BackEnd contenant toutes les informations de cette catégorie, celle-ci va traiter les informations appropriées et envoyer une requête SQL a la base de données afin de l’enregistrer.

L’utilisateur sera redirigé vers l’interface de gestion des catégories dans laquelle la nouvelle catégorie sera affichée.

* Interface de gestion des véhicules

C’est l’interface dédiée à la gestion des véhicules, via cette page l’agent ou l’administrateur trouve tous les véhicules insérés dans la base de données afin de pouvoir les gérer.

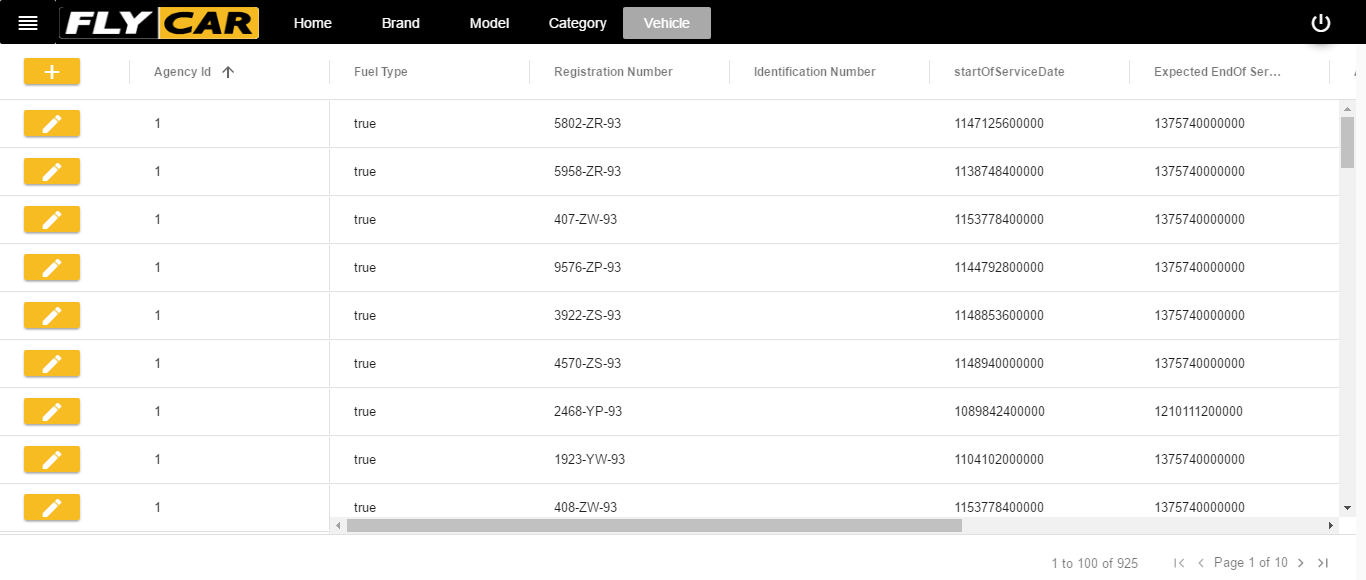


Figure 46 : interface véhicules

Après avoir s’authentifier l’utilisateur va choisir le bouton Véhicule dans le menu. Il sera redirigé vers l’interface de gestion des modèles. Un tableau contenant toutes les véhicules enregistrés dans la base de données sera affiché.

A partir de ce tableau l’utilisateur aura la main pour ajouter, modifier ou désactiver un véhicule.

* Interface d’ajout d’un véhicule

C’est l’interface qui va permettre à l’agent ou l’administrateur d’ajouter un véhicule qui sera inséré dans la base de données.

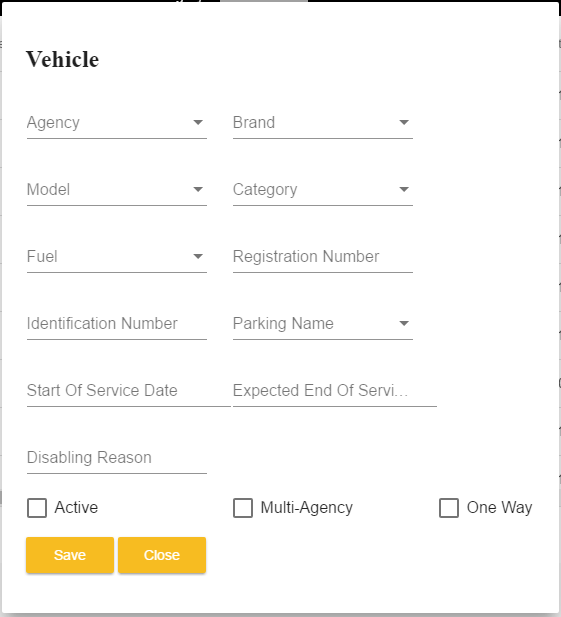


Figure 47 : interface d’ajout d’un véhicule

Cette interface sera affichée après avoir cliqué sur l’icône d’ajout d’un véhicule.

L’utilisateur va remplir les champs du formulaire ensuite il valide en cliquant sur le bouton « enregistrer ». Une requête http sera envoyée à l’application BackEnd contenant toutes les informations de ce véhicule, celle-ci va traiter les informations appropriées et envoyer une requête SQL a la base de données afin de l’enregistrer.

L’utilisateur sera redirigé vers l’interface de gestion des véhicules dans laquelle le nouveau véhicule sera affiché.

1. Conclusion :

Avec la fin de ce chapitre, nous clôturons ce projet de fin d’études tout en ayant détaillé toutes les étapes de sa réalisation. Sachant que, la réalisation de cette phase nous a permis de mettre en œuvre toutes les compétences acquises pendant la formation que nous avons eue à Spring et Angular 5. Nous avons essayé de montrer dans ce chapitre la quantité et la qualité de la tâche de la programmation et les connaissances mises en œuvre pour l’élaboration.

# Conclusion générale

Pour conclure, j’ai eu l’honneur d’effectuer ce projet au sein de la société LB Consulting. Il s’agit de contribuer à réaliser une Application de Gestion de Flotte automobile.

Ce projet a été mené dans le but de gérer l’utilisation et la maintenance d’un parc de véhicules de façon optimale et organisée. Le projet s’est déroulé selon trois principales phases ; de façon à passer par les étapes essentielles de tout projet : l’analyse, la conception et la réalisation.

La phase d’analyse touche plusieurs points, d’un côté les besoins fonctionnels, il s’agit des fonctions du futur système permettant de satisfaire les besoins des futurs utilisateurs, et d’un autre côté les besoins non fonctionnels, il est question des besoins en matière de performance, le type de matériel ou le type de conception.

J’ai eu recours aux diagrammes de cas d'utilisation et les diagrammes de séquence préconisés par UML pour effectuer cette première étape.

En ce qui concerne la seconde phase, à savoir la conception, il s’agit d’une étape très importante et son élaboration doit être bien méticuleuse afin d’aboutir à un résultat pouvant engendrer une étape de réalisation pertinente.

Au moment de la réalisation, j’ai utilisé le PostgreSQL en ce qui concerne la couche base de données, Spring pour l’application BackEnd et quant à FrontEnd, j’ai utilisé Angular 5.

Ce projet m’a permis d’être plus proche de la vie professionnelle, de suivre les différentes étapes de développement d’un projet et surtout d’enrichir mes connaissances au niveau de la conception.

Ainsi, cette formation m’a permis de confronter plusieurs contraintes que ce soient celles du temps, de la qualité ou des technologies.

Finalement, j’ai eu la chance de beaucoup apprendre au cours de cette expérience enrichissante néanmoins on ne s’arrête jamais d’apprendre.

# Webographie

[1] [www.etudiant.aujourdhui.fr](http://www.etudiant.aujourdhui.fr)

[2] <https://spring.io/>

[3] <https://projects.spring.io/spring-boot/>

[4] <https://projects.spring.io/spring-data/>

[5] <https://projects.spring.io/spring-security/>

[6] <http://www.baeldung.com/role-and-privilege-for-spring-security-registration>

[7] <https://docs.spring.io/spring-security/site/docs/2.0.7.RELEASE/apidocs/org/springframework/security/userdetails/UserDetails.html>

[8] <https://medium.com/>

# Bibliographie

[9] « UML 2 en action »: De l'analyse des besoins à la conception, Livre de Franck Vallée et Pascal Roques.

[10] « Les Cahiers du Programmeur UML » : Modéliser une application Web, de Pascal Roques.