Elève : Maxime Michel

Promoteur : Monsieur Fabrice Mahieu

Année académique : 2018 - 2019

Application de gestion d’une concession de marque BMW

Travail de fin d’études présenté en vue de l’obtention du diplôme de Bachelier en informatique de gestion

Ecole Industrielle et Commerciale

Rue Ernest Martel 6

7190 Ecaussinnes

Table des matières

[Remerciements 3](#_Toc21269655)

[Concession BMW 4](#_Toc21269656)

[1. Introduction 4](#_Toc21269657)

[1.1. Choix du sujet 4](#_Toc21269658)

[1.2. Situation de l’informatisation 4](#_Toc21269659)

[2. Objectif du travail de fin d’études 5](#_Toc21269660)

[3. Analyse fonctionnelle 6](#_Toc21269661)

[3.1. Applications utilisées pour l’analyse 6](#_Toc21269662)

[3.2. Liste de fonctionnalités 6](#_Toc21269665)

[3.3. Cycle de développement 8](#_Toc21269666)

[3.4. Contraintes matérielles 9](#_Toc21269667)

[3.5. Étude de l’existant 9](#_Toc21269668)

[3.6. Système d’informations 9](#_Toc21269669)

[3.7. Diagramme des uses cases 10](#_Toc21269670)

[4. Descriptif des uses cases 12](#_Toc21269671)

[4.1. Gestion des réparations et des entretiens 12](#_Toc21269672)

[4.2. Gestion des rendez-vous 13](#_Toc21269673)

[4.3. Gestion des ventes 14](#_Toc21269674)

[4.4. Gestion des factures 16](#_Toc21269675)

[4.5. Gestion des employés 17](#_Toc21269676)

[4.6. Gestion des véhicules 18](#_Toc21269677)

[4.7. Gestion des options et des packs d’options 19](#_Toc21269678)

[4.8. Gestion du stock de marchandises 20](#_Toc21269679)

[5. Analyse structurelle 21](#_Toc21269680)

[5.1. Schéma de base de données 21](#_Toc21269681)

[5.2. Le dictionnaire de données 27](#_Toc21269682)

[5.3. Présentation des maquettes d’écrans 36](#_Toc21269683)

[6. Structure du code 49](#_Toc21269684)

[6.1. Applications utilisées pour le développement 49](#_Toc21269685)

[6.2. Description des différents dossiers et de leurs contenus 50](#_Toc21269686)

[6.3. Explications des algorithmes du code 55](#_Toc21269687)

[6.4. La génération des factures 59](#_Toc21269688)

[7. Conclusions 63](#_Toc21269689)

[7.1. Les objectifs atteints 63](#_Toc21269690)

[7.2. Les modifications sur mes fonctionnalités 63](#_Toc21269691)

[7.3. Futur de l’application 63](#_Toc21269692)

[7.4. Ce que m’a apporté ce travail de fin d’études 64](#_Toc21269693)

[8. Annexes 65](#_Toc21269694)

[8.1. Conseils d’utilisations 65](#_Toc21269695)

[8.2. Diagramme de classes complet 0](#_Toc21269696)

[8.3. Bibliographie 0](#_Toc21269697)

# Remerciements

Tout d'abord, j'adresse mes remerciements à Mr Fabrice Albert MAHIEU pour sa disponibilité et ses explications sur les zones floues concernant le travail de fin d’études.

Je remercie également les membres du jury pour le temps investi lors de la lecture de ce PDF et des tests réalisés sur l’application.

Je tiens également à remercier ma famille pour avoir testé mon application une fois terminée afin de chercher des bugs et les améliorations possibles ainsi que pour la correction de ce document.

# Concession BMW

## Introduction

### Choix du sujet

Le sujet que j’ai choisi pour réaliser mon travail de fin d’études s’est porté sur une de mes passions qui est le domaine de l’automobile. Par curiosité, j’ai voulu voir combien coûterait un véhicule qui me plait et à la fin de ma personnalisation, j’ai vu qu’on pouvait imprimer un PDF comprenant les informations de cette personnalisation. De ce fait, je me suis dit que dans la concession ils avaient sûrement des applications permettant l’encodage d’un véhicule afin de le commander.

### Situation de l’informatisation

Afin de jouer le jeu, je vais inventer un client qui souhaite une nouvelle application lui permettant d’avoir des impressions de factures de vente, de bons de commande et un système de prise de rendez-vous pour les réparations et les entretiens à effectuer sur des véhicules avec une gestion des différentes pièces à utiliser pour ces tâches. À l’heure actuelle, ce client ne dispose que d’une vieille application un peu dépassée et pas très esthétique. Il souhaite avoir une distinction entre 3 types d’employés, un administrateur afin de gérer les différentes nouveautés de la concession, des vendeurs et des mécaniciens.

## Objectif du travail de fin d’études

L’objectif sera de réaliser une application permettant la gestion d’une concession BMW en proposant plusieurs services : la vente de véhicules avec une personnalisation, l’entretien de véhicules et la réparation de véhicules. Lors de la vente, le client aura un choix d’options et de packs en fonction du modèle désiré. Lors de la vente par personnalisation d’un véhicule, un bon de commande sera complété automatiquement avec les informations personnelles du client, le choix du véhicule et une description détaillée du choix des options, permettant de garder une trace de la commande du client. Ensuite, lors de l’achat d’un véhicule directement disponible en concession, une facture de vente sera directement disponible à l’impression. La concession proposera comme véhicules des voitures et des motos. Les réparations et les entretiens se prendront par rendez-vous et la gestion des rendez-vous sera réalisée sous forme d’un calendrier permettant ainsi au vendeur de consulter, de modifier, d’ajouter ou de supprimer les rendez-vous.

L’administrateur aura le choix, en plus du reste, d’accéder à une partie de gestion de stock, d’une gestion des nouvelles options, d’une gestion des packs d’options et d’une partie de gestion des employés.

Quant au mécanicien, il aura une interface lui permettant de consulter les tâches qui lui seront demandées en fonction du planning des rendez-vous. À la fin d’une réparation ou d’un entretien, il devra la valider pour que le vendeur puisse avoir l’information que le véhicule est prêt à être repris par le client. Lors de l’entretien ou des réparations d’un véhicule, il devra encoder les différentes pièces utilisées. Il pourra également indiquer au vendeur si une réparation doit être effectuée.

## Analyse fonctionnelle

### Applications utilisées pour l’analyse

* Balsamiq Mockups 3

Pour la création des maquettes, j’ai utilisé le logiciel « Balsamiq Mockups 3 » qui est un programme spécialement conçu pour cette tâche. Elle permet un aspect réaliste sur les maquettes que j’apprécie énormément, si l'application est bien visualisée dans notre tête, il est très aisé de le retranscrire avec le logiciel. De ce fait, j’ai très rarement du faire des modifications sur les maquettes afin qu’elles correspondent au mieux à l’application.

* LucidChart

Pour la création de mon diagramme de classes et des uses cases, j’ai utilisé le site LucidChart. Celui-ci dispose d’une formule gratuite limitant le nombre de projets et d’éléments possibles à mettre sur un projet, mais son aspect visuel et son utilisation m’ont convaincu sur le choix de cet outil malgré cette contrainte.



### Liste de fonctionnalités

Administrateur

* Ajouter un nouveau vendeur ou un nouveau mécanicien.
* Consulter le compte d’un vendeur ou d’un mécanicien.
* Modifier le compte d’un vendeur ou d’un mécanicien.
* Supprimer le compte d’un vendeur ou d’un mécanicien
* Ajouter un véhicule à la concession
* Ajouter un nouveau modèle de véhicule
* Modifier un modèle de véhicule existant
* Supprimer un modèle de véhicule existant
* Consulter les caractéristiques
* Ajouter une caractéristique
* Modifier une caractéristique
* Supprimer une caractéristique
* Consulter les options disponibles pour les véhicules
* Ajouter une nouvelle option
* Modifier une nouvelle option
* Supprimer une option
* Consulter les packs disponibles pour les véhicules
* Créer un pack de personnalisation avec des options existantes
* Supprimer un pack de personnalisation
* Modifier les packs de personnalisation

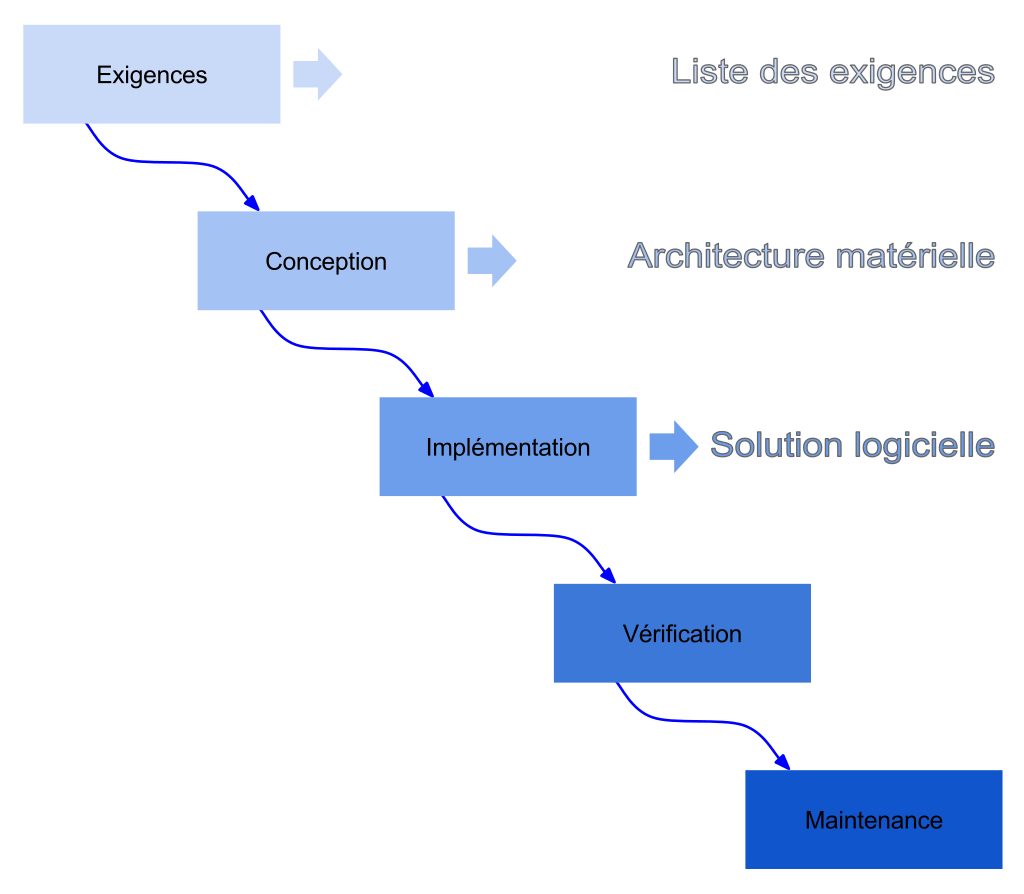
Vendeur

* Enregistrer un client au moment de la vente, d’un rendez-vous ou d’une commande
* Se baser sur les informations d’un client existant pour préparer une vente ou une commande
* Modifier les informations d’un client
* Personnaliser un véhicule avec les options désirées par le client
* Imprimer le bon de commande
* Finaliser la vente après une commande
* Imprimer la facture de la vente, de l’entretien et/ou de la réparation
* Consulter l’inventaire de stock
* Ajouter un article dans l’inventaire
* Supprimer un article dans l’inventaire
* Modifier un article dans l’inventaire
* Consulter les factures
* Ajouter un rendez-vous pour un entretien et/ou une réparation
* Modifier un rendez-vous d’un entretien et/ou d’une réparation
* Supprimer un rendez-vous d’un entretien et/ou d’une réparation

Mécanicien

* Consulter les réparations ou les entretiens qui lui sont assignés
* Indiquer qu’une réparation ou qu’un entretien de véhicule est terminé en renseignant les réparations et/ou l’entretien effectués et les éventuelles réparations supplémentaires effectuées ou à prévoir
* Encoder les articles utilisés pour mettre à jour l’inventaire du stock de marchandises

### Cycle de développement



Lors de la réalisation de mon projet, j’ai utilisé le cycle de développement en cascade. J’ai commencé par analyser les différentes fonctionnalités que le client désirait. Ensuite, j’ai commencé la réalisation des différentes maquettes, des uses cases et du diagramme de classes. Une fois que l’analyse a été terminée, j’ai commencé à réaliser le code. Une fois mon code terminé, j’ai réalisé quelques tests sur les différentes fonctionnalités afin de m’assurer de leurs bons fonctionnements. Cette méthodologie m’aura permis d’être structuré et surtout permis de savoir où aller dans mon application, quelles pages auraient quelles fonctionnalités, quelles personnes auront accès à ces pages, etc…

### Contraintes matérielles

Certaines pages nécessitant une résolution de 1400x700, il est requis d’avoir une résolution supérieure ou égale à 1600x900 afin d’avoir une utilisation optimale de l’application. Il sera nécessaire d’avoir un serveur local ou en ligne afin d’avoir une utilisation globale de l’application afin qu’un administrateur puisse transmettre les nouvelles données à tous les utilisateurs de l’application.

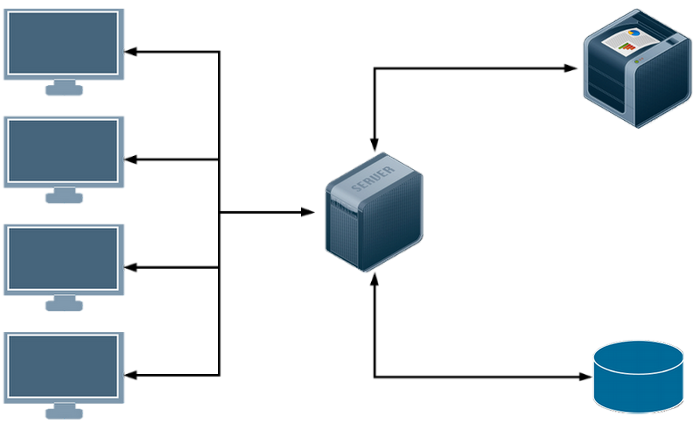
### Étude de l’existant

Afin de me donner une réelle idée de personnalisation possible sur un véhicule et des différentes conditions, je me suis rendu sur le site officiel de BMW et j’ai personnalisé un véhicule et réalisé qu’il était possible de sortir une fiche de personnalisation avec les choix afin de se rendre à la concession et passer une commande. Cela m’a donné l’idée de faire une application afin de pouvoir recevoir des clients ayant ces fiches et les encoder sur l’application afin de sortir une facture.

Ensuite, n’ayant encore jamais acheté de véhicule neuf, je me suis un peu renseigné auprès de proches afin de savoir comment se passait l’achat d’un tel véhicule. J’avais besoin de plus d’informations pour connaître les informations qui seraient importantes à indiquer. Beaucoup d’informations en sont ressorties, j’ai donc fait un tri des informations que j’estimais être les plus importantes afin de ne pas surcharger mon application et la charge de travail.

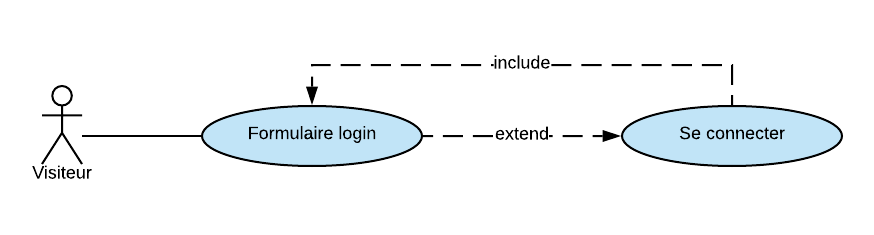
Pour les factures, j’ai effectué quelques recherches sur internet afin de voir les différents modèles de factures d’achat. Durant ces recherches j’ai remarqué que les factures différenciaient beaucoup d’une concession à l’autre, de ce fait, j’ai essayé au maximum de faire une facture contenant les informations requises.

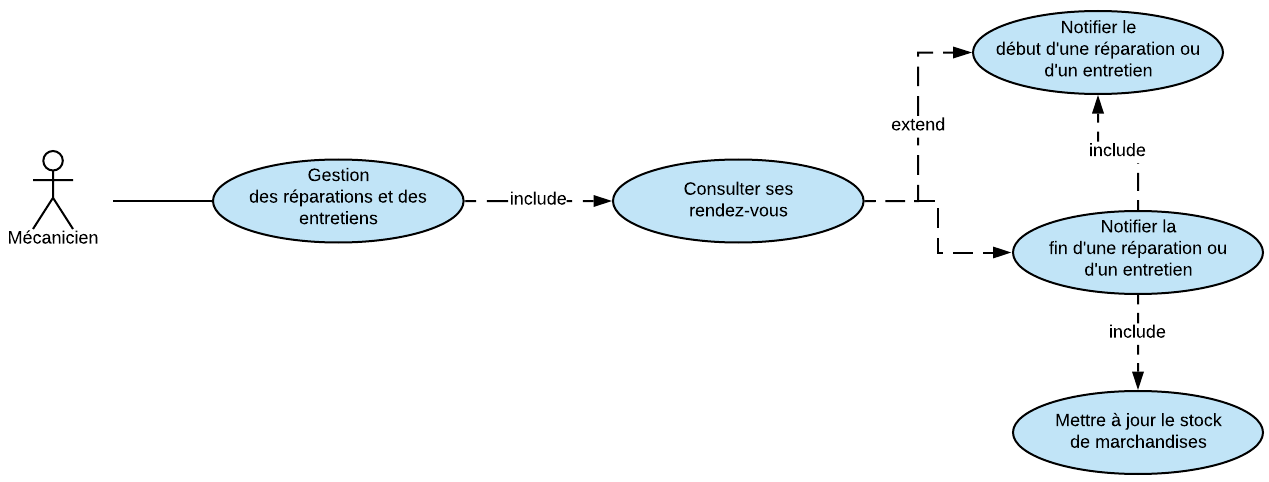
### Système d’informations

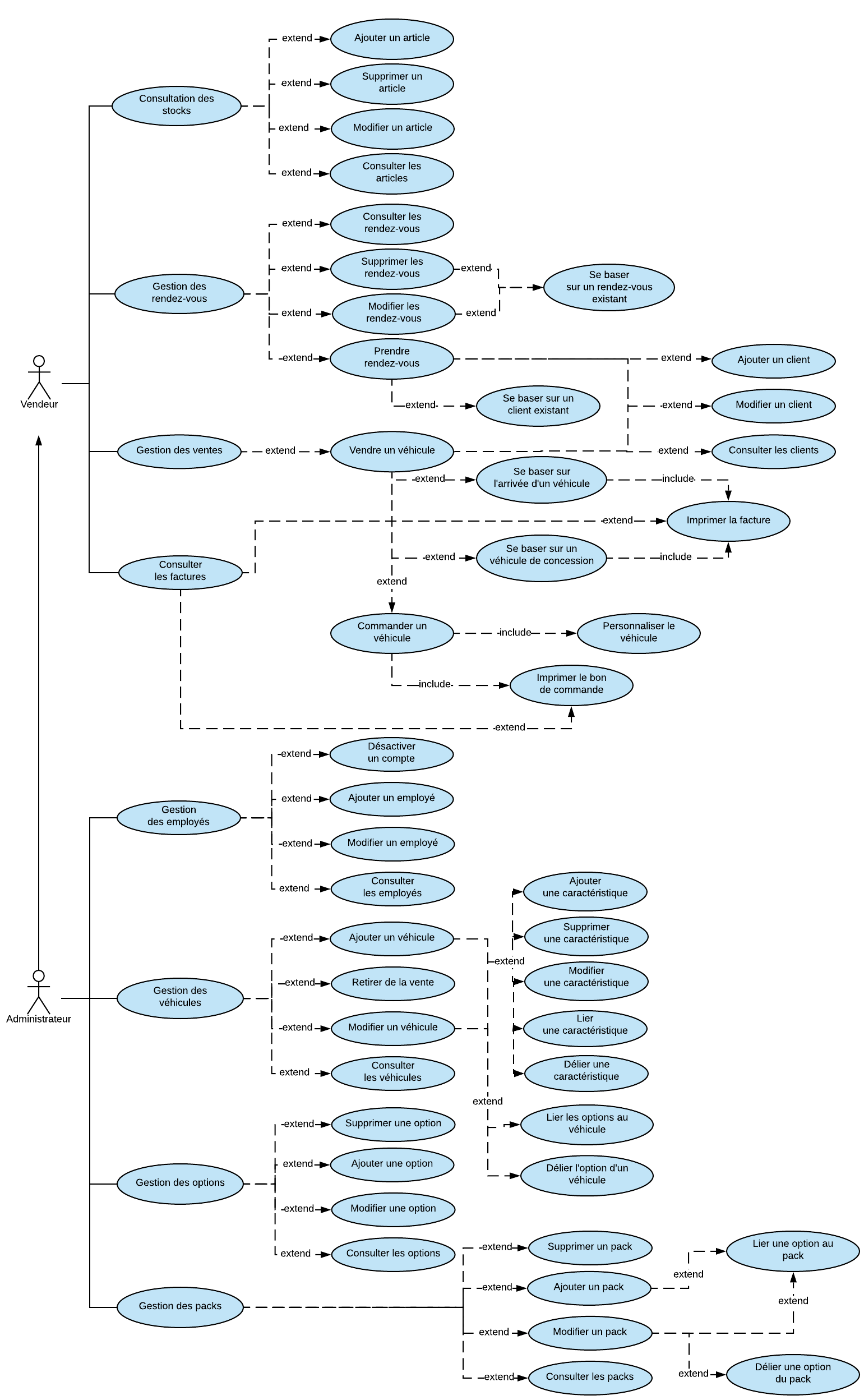


Pour ce système d’informations, j’ai décidé de partir vers un système en réseau local. Chaque pc communiquera avec un serveur qui leur donnera les réponses de la base de données. Chaque pc aura accès à une imprimante qui sera elle aussi connectée au serveur.

### Diagramme des uses cases







## Descriptif des uses cases

### Gestion des réparations et des entretiens

Acteurs principaux

Les mécaniciens.

Description générale

Le mécanicien devra indiquer le véhicule qu’il souhaite prendre en charge afin qu’il passe de « En attente » à « En cours ». Lorsqu’il aura choisi un entretien et/ou une réparation, il devra indiquer les heures prestées sur le véhicule.

Si le véhicule a besoin d’un mécanicien spécialisé dans un domaine spécifique, le mécanicien devra lui aussi indiquer les heures qu’il aura prestées sur le véhicule.

Lors des différents réparations ou entretiens, le mécanicien devra indiquer les articles qu’il aura utilisés afin de garder une trace des différentes quantités des articles. Une fois les réparations et/ou l’entretien terminé(s), le dernier employé devra indiquer que le véhicule est prêt à repartir. Son statut sera automatiquement changé et passera de « En cours » à « En attente ».

Précondition

Les mécaniciens doivent être connectés.

Les articles doivent être existants avant de commencer un entretien ou une réparation.

Post-conditions

Une fois les mécaniciens connectés, ils pourront consulter la liste des rendez-vous. Quand une réparation est terminée, ils devront encoder les différentes pièces utilisées pour la ou les réparation(s) et pour l’entretien si le client en désirait un. Cette manipulation permettra à l’administrateur de garder un œil sur le stock de marchandises en le mettant à jour.

Scénario nominal

1. Le mécanicien consulte ses rendez-vous.
2. Le mécanicien sélectionne un de ses rendez-vous.
3. Passe le statut du rendez-vous à « En cours ».
4. Indique les pièces utilisées pour la réparation ou l’entretien.
5. Indique que la réparation est terminée afin de passer de statut « En cours » à « Terminé ».

### Gestion des rendez-vous

Acteurs principaux

Les vendeurs ou l’administrateur.

Description générale

L’utilisateur aura la possibilité de planifier un rendez-vous pour une réparation et/ou pour un entretien. L’utilisateur précisera alors la date et l’heure de l’entretien avec le client. Si le client n’existe pas encore, il devra alors encoder les informations du client et du véhicule. Il pourra ensuite l’affecter au rendez-vous.

Lors de l’entretien avec le client, l’utilisateur devra indiquer les raisons de son rendez-vous afin que le mécanicien ait une idée du problème concernant le véhicule. Il pourra également supprimer ou modifier la date d’un rendez-vous si celui-ci n’est pas « En cours » ou « Terminé ».

Précondition

L’utilisateur doit avoir les droits d’un vendeur ou d’un administrateur afin de prendre un rendez-vous.

Post-conditions

Les mécaniciens auront accès à la liste des rendez-vous à effectuer, d’une liste d’articles existants et utilisés, du message du vendeur et de l’entretien à effectuer sur le véhicule.

Scénario nominal

1. L’utilisateur inscrit le client.
2. L’utilisateur inscrit la date et l’heure du rendez-vous et les différentes informations concernant ce rendez-vous.

Scénario alternatif

Le client existe déjà :

* 1. Le vendeur inscrit le rendez-vous du client existant.

Le client souhaite supprimer son rendez-vous :

1. L’utilisateur recherche le rendez-vous à supprimer.
2. L’utilisateur supprime le rendez-vous.

Le client souhaite modifier son rendez-vous :

1. L’utilisateur recherche le rendez-vous à modifier.
2. L’utilisateur modifie le rendez-vous.

Scénario d’exception

Il est impossible d’affecter un rendez-vous sans créer ou référencer un client.

Il est impossible de supprimer un rendez-vous qui est en cours, terminé ou avec la date de fin dépassée.

### Gestion des ventes

##### Vendre un véhicule de concession

Acteurs principaux

Les vendeurs et l’administrateur.

Description générale

L’utilisateur aura, lors de sa consultation des différents catalogues, la possibilité de choisir un véhicule disponible en concession. Une fois sur la page des disponibilités, il aura une description complète du véhicule ainsi que les différentes options et les packs que ce véhicule dispose.

Si la concession désire reprendre le véhicule, l’utilisateur devra encoder une ristourne de reprise de véhicule et y indiquer le véhicule à reprendre au client.

Précondition

L’utilisateur doit avoir les droits de vendeur ou d’administrateur.

Le véhicule doit être encodé au préalable pour le voir dans les disponibilités.

Post-conditions

Le vendeur ou l’administrateur dispose de la facture détaillée de la vente en document PDF prêt à l’impression une fois que le client a été choisi.

Scénario nominal

1. L’utilisateur choisit le véhicule que le client aimerait acheter.
2. L’utilisateur inscrit le client ou le sélectionne.
3. L’utilisateur rajoute les différentes informations du véhicule.
4. Le document PDF comprenant la facture détaillée s’ouvre.
5. L’utilisateur peut imprimer la facture.

##### Commander un nouveau véhicule

Acteurs principaux

Les vendeurs et l’administrateur.

Description générale

L’utilisateur aura, lors de sa consultation des différents catalogues, la possibilité de choisir de personnaliser son véhicule. Une fois le véhicule choisi, des options et des packs lui seront proposés en fonction du modèle de véhicule choisi. Un bon de commande sera alors imprimé avec les informations du client et des différents choix du client. Une fois que le véhicule est arrivé et que le client revient avec son bon de commande, le reste des informations du véhicule devront être encodées. Si une reprise de véhicule est effectuée, il devra également indiquer la ristourne faite par le garage sur le nouvel achat.

Précondition

L’utilisateur doit avoir les droits de vendeur ou d’administrateur.

Le client doit avoir commandé un véhicule pour passer à l’étape d’achat.

Post-conditions

1. Le vendeur ou l’administrateur dispose d’un bon de commande en document PDF prêt à l’impression.
2. Le vendeur ou l’administrateur dispose d’une facture détaillée de la vente.

Scénario nominal

1. L’utilisateur choisit le véhicule que le client aimerait acheter.
2. L’utilisateur inscrit le client ou le sélectionne.
3. L’utilisateur rajoute les différentes informations du véhicule.
4. Le document PDF comprenant le bon de commande est prêt à l’impression.
5. Le véhicule est commandé pour la date prévue.
6. Le client revient avec son bon de commande.
7. L’utilisateur encode le reste des informations.
8. L’utilisateur dispose d’un document PDF comprenant la facture détaillée.

### Gestion des factures

Acteurs principaux

L’administrateur et les vendeurs.

Description générale

Une page de consultation de facture sera proposée afin de réimprimer une facture ou un bon de commande si nécessaire. Lors de la suppression d’un article et/ou d’un véhicule, ils ne seront que fictivement supprimés afin de garder les données de la facture correctes. Les articles existants sur des factures ne seront jamais modifiables s’ils sont référencés sur une facture.

Précondition

L’utilisateur devra avoir le rôle d’administrateur ou de vendeur.

Les factures, les bons de commande et les entretiens et/ou les réparations doivent être terminés avant d’être disponibles sur la gestion des factures.

Post-conditions

Les factures sont disponibles à l’impression et la longévité dépendra de la durée de sauvegarde des données.

Scénario nominal

L’utilisateur se trouve sur l’écran des factures de vente :

1. L’utilisateur choisit une facture de vente.
2. Le document de la facture de vente s’ouvre en document PDF et est prêt à l’impression.

Scénario alternatif

L’utilisateur choisit un bon de commande à imprimer :

1. Le document du bon de commande s’ouvre en document PDF et est prêt à l’impression.

L’utilisateur se trouve sur l’écran des entretiens et réparations :

1. L’utilisateur choisit une facture.
2. Le document de la facture s’ouvre en document PDF et est prêt à l’impression.

### Gestion des employés

Acteurs principaux

L’administrateur.

Description générale

Sur cette page, l’administrateur pourra désactiver le compte d’un employé ou y modifier ses informations personnelles (nom, prénom, adresse, code postal, poste, …). Il aura également la possibilité d’ajouter un nouvel employé en sélectionnant son poste. Lors de la création du compte, il devra choisir un mot de passe de connexion pour que l’utilisateur puisse se connecter à l’application.

Précondition

L’utilisateur devra être administrateur pour accéder à la page.

Post-conditions

L’employé aura un compte actif avec ses informations personnelles, un email et un mot de passe pour permettre sa connexion à l’application.

Scénario nominal

1. L’utilisateur encode les informations du nouvel employé.
2. L’employé peut maintenant se connecter.

Scénario alternatif

L’employé s’en va de la concession :

1. L’utilisateur sélectionne le compte de l’employé.
2. L’utilisateur désactive le compte.
3. L’application confirme que le compte de l’employé a été désactivé.

L’employé doit changer une de ses informations personnelles :

1. L’utilisateur sélectionne un employé.
2. L’utilisateur modifie les informations puis valide.
3. L’application valide les modifications.

### Gestion des véhicules

Acteurs principaux

L’administrateur.

Description générale

Sur cette page, l’administrateur pourra rajouter un nouveau véhicule, modifier un véhicule existant ou retirer de la vente un véhicule qu’il ne vendra éventuellement plus. Lors de l’ajout d’un véhicule, il devra choisir parmi une sélection d’options correspondantes à la catégorie du véhicule. Il aura également la possibilité de mettre une image sous un format spécifique afin d’avoir un visuel sur le véhicule en question. Il devra également indiquer les caractéristiques de ce véhicule.

Précondition

L’utilisateur devra avoir le rôle d’administrateur.

Post-condition

Le véhicule sera disponible à la vente avec les choix d’options et de packs disponibles pour ce véhicule.

Le véhicule est lié à des caractéristiques, des options et des packs.

Scénario nominal

1. L’utilisateur choisit les caractéristiques du véhicule.
2. L’utilisateur choisit les options liées au véhicule.
3. L’utilisateur choisit les packs liés au véhicule.

Scénario alternatif

La caractéristique existe déjà :

1. L’utilisateur encode les caractéristiques du véhicule.

La caractéristique n’existe pas :

1. L’utilisateur rajoute une caractéristique.
2. L’utilisateur lie la caractéristique.

### Gestion des options et des packs d’options

Acteurs principaux

L’administrateur.

Description générale

Sur cette page, l’administrateur aura la possibilité d’ajouter les options que la concession de la marque BMW propose pour ensuite les lier au moment de l’ajout d’un véhicule (dans la gestion des véhicules). La suppression d’une option ou d’un pack sera possible, mais n’affectera en rien les véhicules existants avec l’option ou le pack lié à celui-ci. Les factures ne seront également pas impactées par ces suppressions.

Précondition

L’utilisateur devra avoir le rôle d’administrateur.

Post-condition

L’option sera disponible lors de l’ajout d’un nouveau véhicule.

Scénario nominal

1. L’utilisateur encode une option.
2. L’utilisateur valide.
3. L’option apparaît dans la liste des options existantes.

Scénario alternatif

L’option existe déjà :

1. Un message avertit l’utilisateur que celle-ci existe déjà.

L’utilisateur ajoute un pack :

1. L’utilisateur encode le nom d’un pack et valide.
2. L’utilisateur choisit une option et clique sur la flèche pour la lier.
3. L’option apparaît dans la liste des options liées afin d’avertir que l’option est correctement liée.

### Gestion du stock de marchandises

Acteurs principaux

L’administrateur et les vendeurs.

Description générale

Une page de consultation du stock sera disponible pour l’administrateur et les vendeurs dans le but d’y consulter les marchandises manquantes. Ils pourront y ajouter les nouvelles marchandises arrivées ainsi qu’indiquer les marchandises déjà disponibles dans la concession.

Précondition

L’utilisateur devra avoir le rôle d’administrateur ou de vendeur.

Le prix est figé s’il a déjà été utilisé pour une facture.

Post-condition

Le stock de marchandises est affiché ainsi que le filtre sur le nom de la marchandise.

Scénario nominal

1. L’utilisateur ajoute une nouvelle marchandise au stock en indiquant son stock ainsi que son prix unitaire.

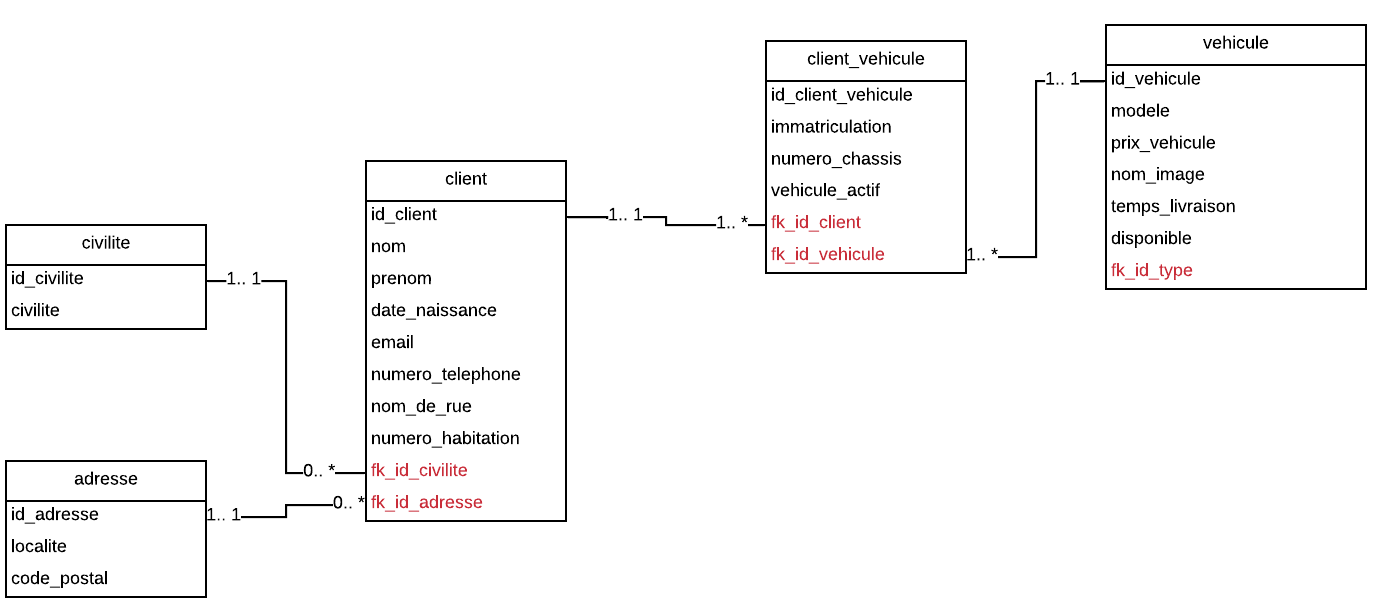
Scénario alternatif

1. L’utilisateur recherche un article, en utilisant le filtre.
2. L’utilisateur sélectionne un article.
3. L’utilisateur modifie les informations comme il le souhaite.

## Analyse structurelle

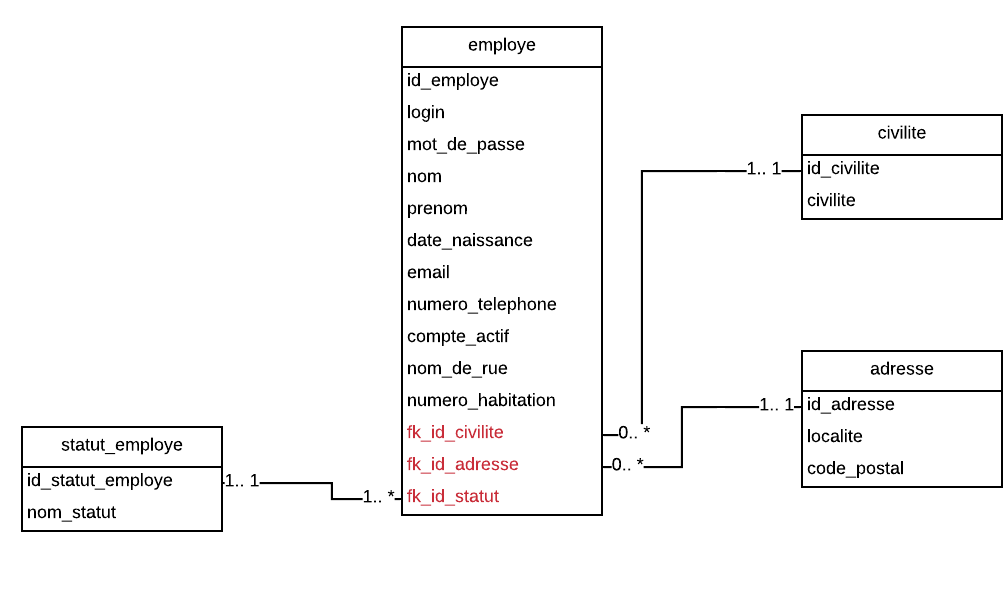
### Schéma de base de données

##### Les clients



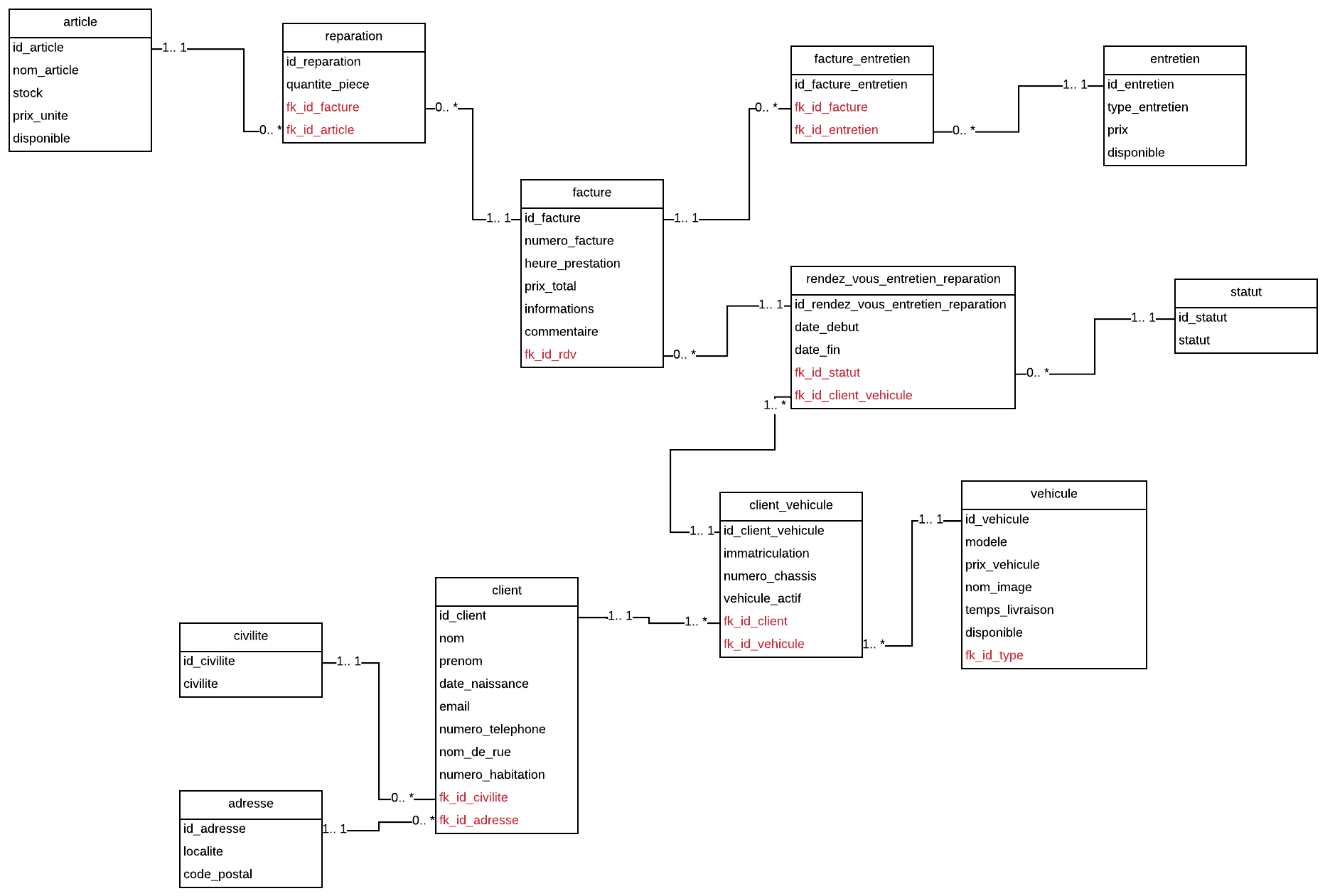
Cette partie du schéma regroupe toutes les données nécessaires pour enregistrer complètement un client. Si le client revend son véhicule à la concession, le champ « vehicule\_actif » entre en ligne de compte, il sera mis à 0 pour dire que ce véhicule ne lui appartient plus. De ce fait, l’unicité du numéro de châssis se fait par code et non pas dans la base de données, car ce véhicule pourrait être revendu à un autre client. Un enregistrement de la table « client\_vehicule » ne pourra exister que si au moins un véhicule et un client existent.

##### Les employés



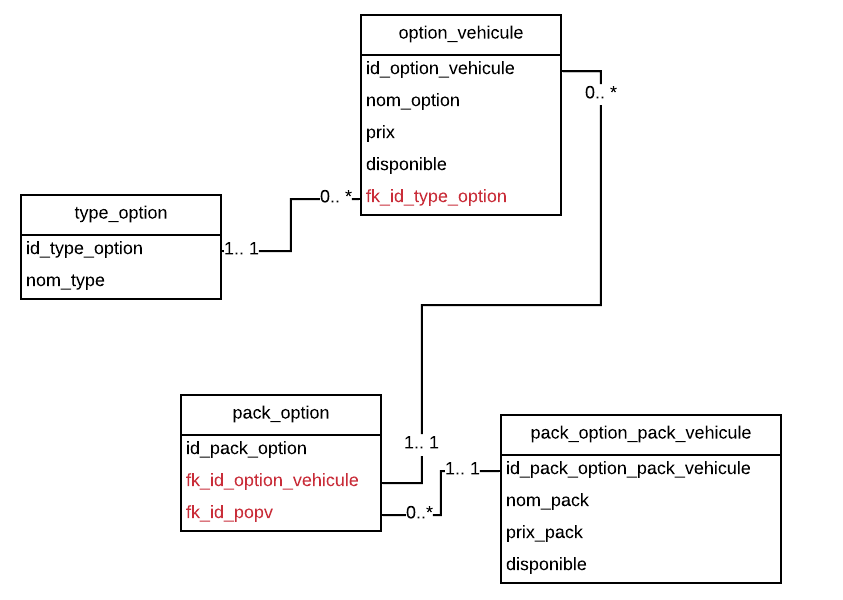
Cette partie du schéma décrit la gestion d’un employé, les statuts correspondent aux différents rôles disponibles : administrateur, vendeur et mécanicien. Une unicité est faite sur l’email et celui-ci doit obligatoirement comporter un nom de domaine spécifique qui est « @BMWEcaussinnes.com ».

##### Les entretiens et les réparations



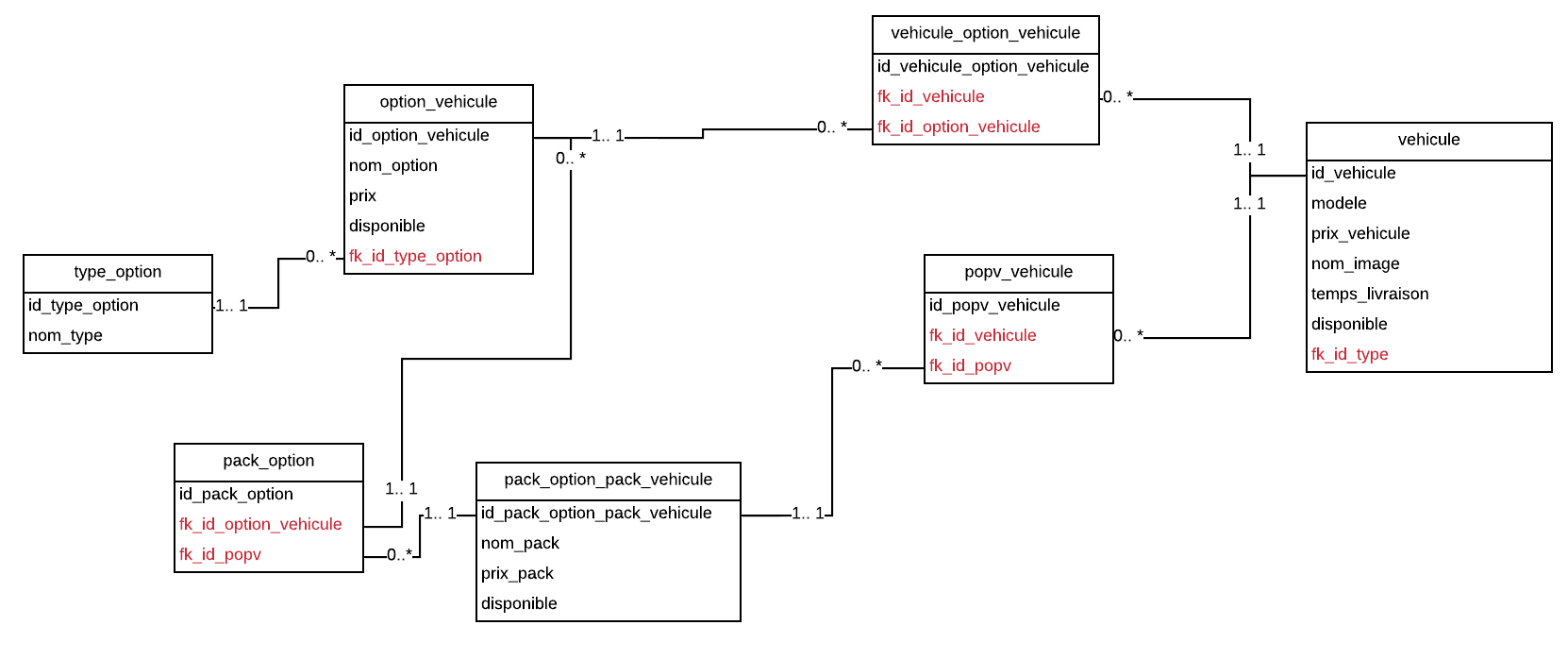
Cette partie du schéma décrit comment se sauvegardent les rendez-vous pour les entretiens et les réparations. Les tables comprenant un champ avec le nom « disponible » me permettent de toujours garder une trace de ces enregistrements si ceux-ci sont « supprimés ». Grâce à cela, la facture restera toujours disponible pour une impression de celle-ci. La partie que je considère un peu particulière sur cette partie du schéma est le fait que la facture se crée au fur et à mesure de l’avancement dans le temps. Une première partie est créée avec les informations de base dont le mécanicien aura besoin afin de réaliser sa tâche. Une fois que le mécanicien commence son travail, le reste des informations sont ajoutées, par exemple, son commentaire sur les éventuels réparations ou entretiens à faire. Dans ce cas-ci, la table statut indique si ce rendez-vous est « En cours » ou « Terminé ».

##### Les packs et les options



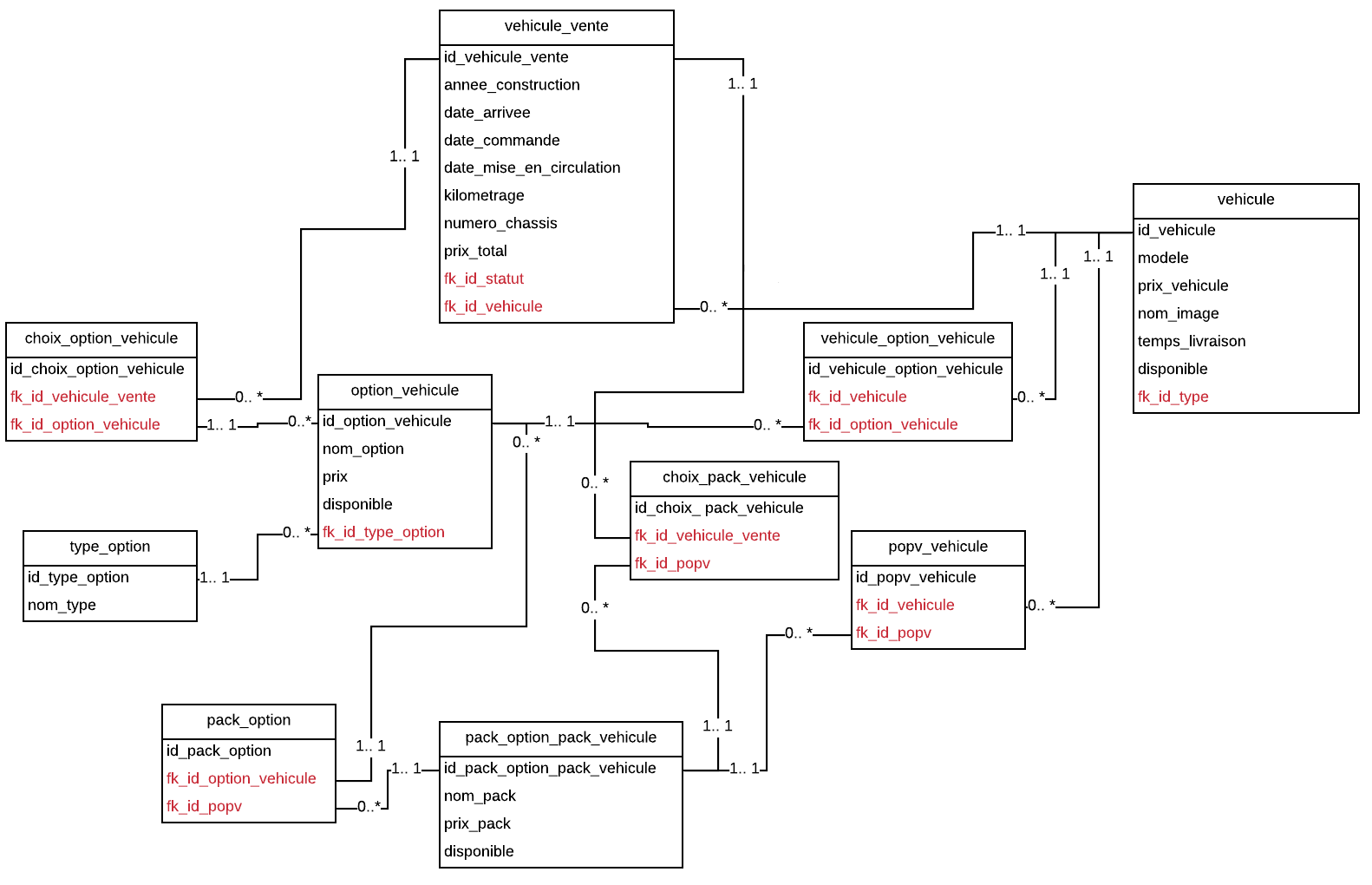
Ce schéma explique comment sont enregistrés les options et les packs, encore une fois, les tables « option\_vehicule » et « pack\_option\_pack\_vehicule » disposent du champ « disponible » afin de toujours garder une trace de ces enregistrements s’ils sont utilisés sur une facture. Les types d’options sont des champs prédéfinis dans la base de données, ils seront les types d’options habituels dans une personnalisation de véhicule tel que la couleur de la voiture ou encore l’habillage. Un type spécifique existe également afin de permettre des options pouvant se cumuler.

##### La liaison entre les véhicules, les packs et les options



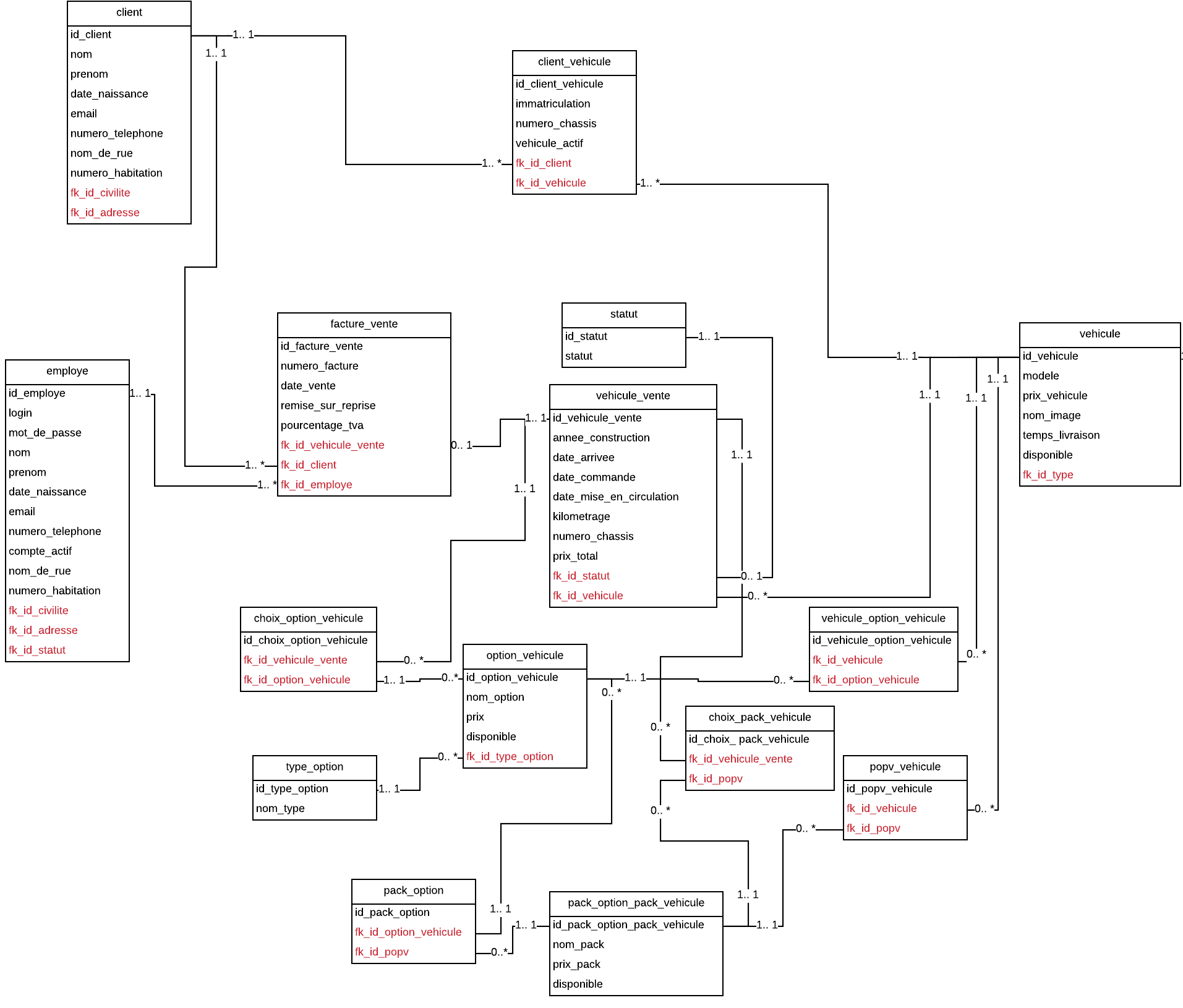
Cette partie du schéma explique la liaison entre les véhicules, les packs d’options et les options.

##### La personnalisation d’un véhicule



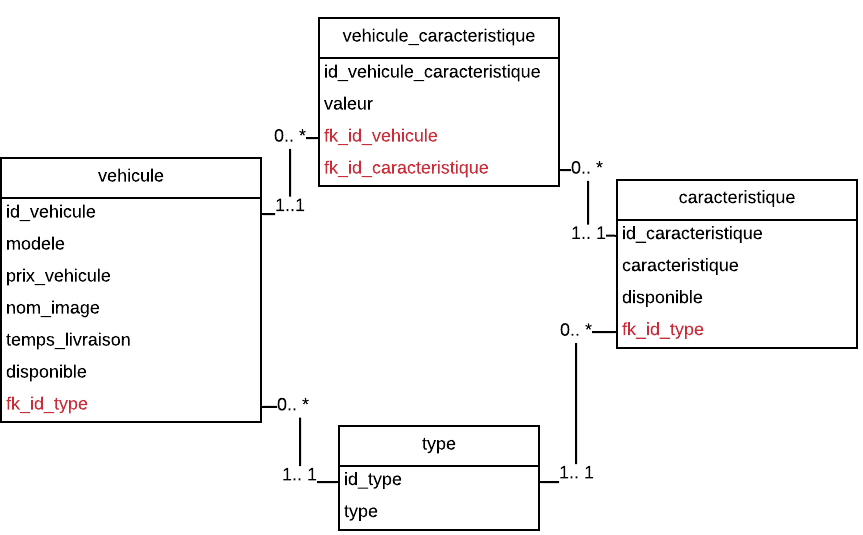
Cette partie du schéma comprend tout ce qui est nécessaire à la personnalisation d’un véhicule. Afin de savoir si un véhicule dispose de cette option, nous devons regarder si cette option est référencée dans la table « vehicule\_option\_vehicule », pour les packs nous devons également vérifier dans la table « popv\_vehicule » s’ils sont référencés. Une fois ces vérifications effectuées, nous pouvons lier les options au véhicule en cours de vente qui est indiquée dans la table « vehicule\_vente ». Ces liaisons seront indiquées dans les tables « choix\_option\_vehicule » et « choix\_pack\_vehicule ». Une unicité par type d’option sera faite par code et une pseudo-unicité privera l’utilisateur d’avoir une option existante dans un pack choisi, cette partie sera plus détaillée lors de l’explication de la maquette de personnalisation.

##### Les factures de ventes



Cette partie du schéma comprend toutes les tables nécessaires à la création d’une facture de vente. Pour commencer, la facture aura besoin d’un employé qui vend le véhicule et d’un client. Ensuite, il personnalisera son véhicule comme il le désire en utilisant les tables expliquées dans la partie « personnalisation d’un véhicule ». La table de statut dans le cas présent utilise les données de la table « En livraison », « Vendu » et « En stock ». Les véhicules disponibles dans la concession seront également encodés dans ces tables et auront le statut « En stock ». La table « statut » et son champ « En stock » me permettront de faire la différence entre les véhicules de la concession et les véhicules personnalisés par un client. La table « client\_vehicule » aura pour but, une fois la livraison complétée, de recueillir les quelques informations importantes pour des éventuels réparations et/ou entretiens à prévoir au sein de la concession. Cette table me permet en plus de savoir si ce véhicule appartient toujours au client grâce à son champ « vehicule\_actif ».

##### Gestion d’un véhicule et de ses caractéristiques



Cette partie du schéma comprend les tables nécessaires à la création d’un véhicule. Encore une fois, ici, les tables affichant un champ « disponible » sont des enregistrements pouvant être supprimés fictivement par l’utilisateur. Un véhicule ne faisant plus partie des ventes actuelles de la concession pourra être supprimé, mais une trace de l’existence de ce véhicule sera sauvegardée afin de compléter les anciennes factures. Même principe pour les caractéristiques, elles pourront être supprimées, mais si le véhicule a été vendu avec cette caractéristique à un moment donné, ce véhicule disposera encore de cette caractéristique afin d’avoir le détail complet sur l’ancienne facture.

Dans la table des types seront compris 3 types de véhicules qui seront des données figées. Ces types sont : voiture, moto et générique. Le mode générique permet d’avoir une caractéristique comprise sur les 2 types de véhicules lors de l’ajout de ceux-ci. La table me permet également de faire un tri sélectif en fonction de la page dans laquelle l’utilisateur se trouve (les pages pour les voitures ou les pages pour les motos).

### Le dictionnaire de données

##### Table « client »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_client | Identifiant d’un client | int(11) | Auto-incrémenté |
| nom\_client | Nom du client | varchar(250) |  |
| prenom\_client | Prénom du client | varchar(250) |  |
| date\_naissance\_client | Date de naissance | datetime | Format :  yyyy-mm-dd hh :mm :ss |
| email\_client | Email du client | varchar(250) | Unicité sur l’email |
| numero\_telephone\_client | Numéro de téléphone | varchar(10) |  |
| nom\_de\_rue\_client | Nom de la rue | varchar(250) | Ce champ deviendra inutile en cas d’adresse complète en base de données |
| numero\_habitation\_client | Numéro d’habitation | varchar(20) |  |
| fk\_id\_civilite | Civilité du client | int(11) | Monsieur, Madame,… |
| fk\_id\_adresse | Adresse du client | int(11) | Localité, code postal |

##### Table « employe »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_client | Identifiant d’un employé | int(11) | Auto-incrémenté |
| mot\_de\_passe | Mot de passe | varchar(500) | Mot de passe de connexion |
| nom | Nom d’un employé | varchar(250) |  |
| prenom | Prénom d’un employé | varchar(250) |  |
| date\_naissance | Date de naissance | datetime | Format :  yyyy-mm-dd hh :mm :ss |
| email | Email de l’employé | varchar(250) | Unicité sur l’email |
| numero\_telephone | Numéro de téléphone | varchar(10) |  |
| compte\_actif | Compte actif | int(1) | Booléen indiquant la permission de l’accès à l’application |
| nom\_de\_rue | Nom de la rue | varchar(250) | Ce champ deviendra inutile en cas d’adresse complète en base de données |
| numero\_habitation | Numéro d’habitation | varchar(20) |  |
| fk\_id\_civilite | Civilité de l’employé | int(11) | Monsieur, Madame,… |
| fk\_id\_adresse | Adresse de l’employé | int(11) | Localité, code postal |

##### Table « statut\_employe »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_statut\_employe | Identifiant d’un statut d’employé | int(11) | Auto-incrémenté |
| nom\_statut | Statut de l’employé | varchar(250) | Données figées en base de données. Les données sont : «Administrateur», «Mécanicien», «Vendeur» |

##### Table « statut »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_statut | Identifiant d’un statut | int(11) | Auto-incrémenté |
| statut | Statut d’une vente ou d’un entretien et/ou de réparations | varchar(250) | Données figées en base de données. Les données sont : « En attente », « En cours » et « Terminé » |

##### Table « civilite »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_civilite | Identifiant d’une civilité | int(11) | Auto-incrémenté |
| civilite | Civilité d’une personne | varchar(100) | Données figées en base de données. Les données sont : « Monsieur », « Madame » et « Mademoiselle ». Unicité sur la civilité. |

##### Table « adresse »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_adresse | Identifiant d’une civilité | int(11) | Auto-incrémenté |
| localite | Localité | varchar(250) | Données figées en base de données |
| code\_postal | Code postal | int(11) | Données figées en base de données |

##### Table « article »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_article | Identifiant des articles | int(11) | Auto-incrémenté |
| nom\_article | Nom des articles | varchar(250) |  |
| stock | Quantité d’articles disponibles | int(11) |  |
| disponible | Disponible en concession | int(1) | Lors d’une suppression, cet enregistrement passe à 0 pour dire qu’il n’existe plus (booléen) |

##### Table « reparation »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_reparation | Identifiant d’une liaison de réparation | int(11) | Auto-incrémenté |
| quantite\_article | Quantité d’articles | int(3) | Quantité de l’article référencé utilisé. |
| fk\_id\_facture | Facture | int(11) | Facture comprenant cette réparation |
| fk\_id\_article | Article | int(11) | Article utilisé pour cette réparation |

##### Table « entretien »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_entretien | Identifiant d’un entretien | int(11) | Auto-incrémenté |
| type\_entretien | Nom de l’entretien | varchar(250) | Unicité |
| prix | Prix de l’entretien | double |  |
| disponible | Disponible dans les choix | int(1) | Lors d’une suppression, cet enregistrement passe à 0 pour dire qu’il n’existe plus (booléen) |

##### Table « facture »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_facture | Identifiant d’une facture | int(11) | Auto-incrémenté |
| numero\_facture | Numéro de facture | text | Ce numéro de facture commence toujours par « er2019 » pour « entretien + réparation + année actuelle » |
| heure\_prestation | Nombre d’heures prestées | int(11) | Heures prestées sur le véhicule en réparation |
| prix\_total | Prix total | double | Prix calculé sur le nombre d’articles utilisés et/ou de l’entretien sélectionné |
| informations | Informations | varchar(500) | Informations pour le mécanicien sur la réparation et/ou l’entretien |
| commentaire | Commentaire | varchar(500) | Commentaire du mécanicien pour avertir le client |
| fk\_id\_rdv | Rendez-vous | int(11) | Rendez-vous lié à cette facture |

##### Table « facture\_entretien»

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_facture\_entretien | Identifiant d’association | int(11) | Auto-incrémenté |
| fk\_id\_facture | Référence de la facture | int(11) |  |
| fk\_id\_entretien | Référence de l’entretien | int(11) |  |

##### Table « reparation »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_rendez\_vous\_  entretien\_reparation | Identifiant d’un rendez-vous | int(11) | Auto-incrémenté |
| date\_debut | Date de début du rendez-vous | datetime | Format :  yyyy-mm-dd hh :mm :ss |
| date\_fin | Date de fin du rendez-vous | int(11) | Format :  yyyy-mm-dd hh :mm :ss |
| fk\_id\_statut | Statut | int(11) | Indique l’état d’avancement de la réparation et/ou de l’entretien |
| fk\_id\_client\_vehicule | Véhicule du client | int(11) |  |

##### Table « client\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_client\_vehicule | Identifiant d’un véhicule appartenant à un client | int(11) | Auto-incrémenté |
| immatriculation | Immatriculation du véhicule | varchar(250) |  |
| numero\_chassis | Numéro de châssis du véhicule | varchar(250) |  |
| vehicule\_actif | Véhicule actif | int(1) | Ce champ permet de savoir si le véhicule appartient toujours au client |
| fk\_id\_client | Client à qui appartient le véhicule | int(11) |  |
| fk\_id\_vehicule | Indique le modèle du véhicule | int(11) |  |

##### Table « facture\_vente »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_facture\_vente | Identifiant d’une facture de vente | int(11) | Auto-incrémenté |
| numero\_facture | Numéro de la facture | longtext | Commence toujours par l’année actuelle suivie d’un tiret et de l’id de la facture |
| date\_vente | Date de vente | datetime |  |
| remise\_sur\_reprise | Remise sur un véhicule repris | double |  |
| pourcentage\_tva | Pourcentage TVA | int(3) |  |
| fk\_id\_vehicule\_vente | Référence vers le véhicule vendu | int(11) |  |
| fk\_id\_client | Référence du client à qui appartient la facture | int(11) |  |
| fk\_id\_employe | Référence de l’employé ayant vendu le véhicule | int(11) |  |

##### Table « vehicule\_vente »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_vehicule\_vente | Identifiant du véhicule en commande, vendu ou en concession | int(11) | Auto-incrémenté |
| annee\_construction | Année de construction du véhicule | int(4) | Année en numérique |
| date\_arrive | Date d’arrivée du véhicule en concession | datetime |  |
| date\_commande | Date de commande du véhicule | datetime |  |
| date\_mise\_en\_circulation | Date de première mise en circulation | datetime |  |
| kilometrage | Indique le kilométrage au compteur | int(11) |  |
| numero\_chassis | Numéro de châssis du véhicule | varchar(250) |  |
| prix\_total | Prix total | double | Ce champ indique le prix de base du véhicule, des packs et des options |
| fk\_id\_statut | Indique l’avancement de la commande et de la facture | int(11) |  |
| fk\_id\_vehicule | Indique le modèle du véhicule vendu | int(11) |  |

##### Table « type\_option»

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_type\_option | Identifiant d’un type d’option | int(11) | Auto-incrémenté |
| type\_option | Type de l’option | varchar(250) | Unicité. Données figées en base de données |

##### Table « option\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_option\_vehicule | Identifiant d’une option | int(11) | Auto-incrémenté |
| nom\_option | Nom de l’option | varchar(250) |  |
| prix | Prix de l’option | double |  |
| disponible | Disponible dans les choix | int(1) | Lors d’une suppression, cet enregistrement passe à 0 pour dire qu’il n’existe plus (booléen) |
| fk\_id\_type\_option | Indique le type de cette option | int(11) |  |

##### Table « pack\_option\_pack\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_pack\_option\_pack\_vehicule | Identifiant du pack | int(11) | Auto-incrémenté |
| nom\_pack | Nom du pack | varchar(250) |  |
| prix\_pack | Prix du pack | double |  |
| disponible | Disponible dans les choix | int(1) | Lors d’une suppression, cet enregistrement passe à 0 pour dire qu’il n’existe plus (booléen) |

##### Table « vehicule\_caracteristique »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_vehicule\_caracteristique | Identifiant de la valeur d’une caractéristique pour un véhicule | int(11) | Auto-incrémenté |
| valeur | Valeur de la caractéristique | varchar(250) |  |
| fk\_id\_vehicule | Véhicule référencé | int(11) |  |
| fk\_id\_caracteristique | Référence de la caractéristique | int(11) |  |

##### Table « pack\_option »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_pack\_option | Identifiant d’association | int(11) | Auto-incrémenté |
| fk\_id\_option\_vehicule | Référence de l’option liée | int(11) |  |
| fk\_id\_popv | Référence du pack | int(11) |  |

##### Table « choix\_pack\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_choix\_pack\_vehicule | Identifiant d’association | int(11) | Auto-incrémenté |
| fk\_id\_vehicule\_vente | Référence le véhicule personnalisé | int(11) |  |
| fk\_id\_popv | Référence un pack choisi | int(11) |  |

##### Table « choix\_option\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_choix\_option\_vehicule | Identifiant d’association | int(11) | Auto-incrémenté |
| fk\_id\_vehicule\_vente | Référence le véhicule personnalisé | int(11) |  |
| fk\_id\_option\_vehicule | Référence une option choisie | int(11) |  |

##### Table « vehicule\_option\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_vehicule\_option\_vehicule | Identifiant d’association | int(11) | Auto-incrémenté |
| fk\_id\_vehicule | Référence le véhicule | int(11) |  |
| fk\_id\_option\_vehicule | Référence une option lié au véhicule | int(11) |  |

##### Table « popv\_vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_popv\_vehicule | Identifiant d’association | int(11) | Auto-incrémenté |
| fk\_id\_vehicule | Référence le véhicule | int(11) |  |
| fk\_id\_popv | Référence un pack d’options | int(11) |  |

##### Table « vehicule »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_vehicule | Identifiant d’une option | int(11) | Auto-incrémenté |
| modele | Modèle du véhicule | varchar(250) |  |
| prix\_vehicule | Prix catalogue du véhicule | double |  |
| nom\_image | Chemin d’accès à l’image du véhicule | varchar(250) |  |
| temps\_livraison | Temps estimé d’une livraison pour ce modèle | int(11) | Temps estimé en jours |
| disponible | Disponible dans les choix | int(1) | Lors d’une suppression, cet enregistrement passe à 0 pour dire qu’il n’existe plus (booléen) |
| fk\_id\_type | Type du véhicule | int(11) | Référence du type de véhicule |

##### Table « caracteristique »

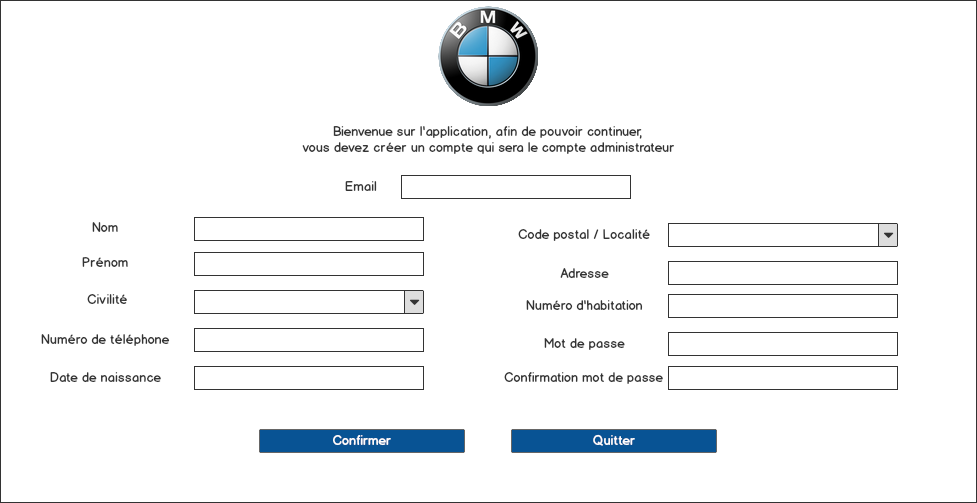
|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_caracteristique | Identifiant de la caractéristique | int(11) | Auto-incrémenté |
| caractéristique | Nom de la caractéristique | varchar(250) |  |
| disponible | Disponible dans les choix | int(1) | Lors d’une suppression, cet enregistrement passe à 0 pour dire qu’il n’existe plus (booléen) |
| fk\_id\_type | Type du véhicule | int(11) | Référence du type de véhicule |

##### Table « type »

|  |  |  |  |
| --- | --- | --- | --- |
| Code mnémonique | Désignation | Type | Remarques |
| id\_type | Identifiant du type de véhicule | int(11) | Auto-incrémenté |
| type | Nom du type de véhicule | varchar(250) | Données figées en base de données |

### Présentation des maquettes d’écrans

##### Page de première connexion



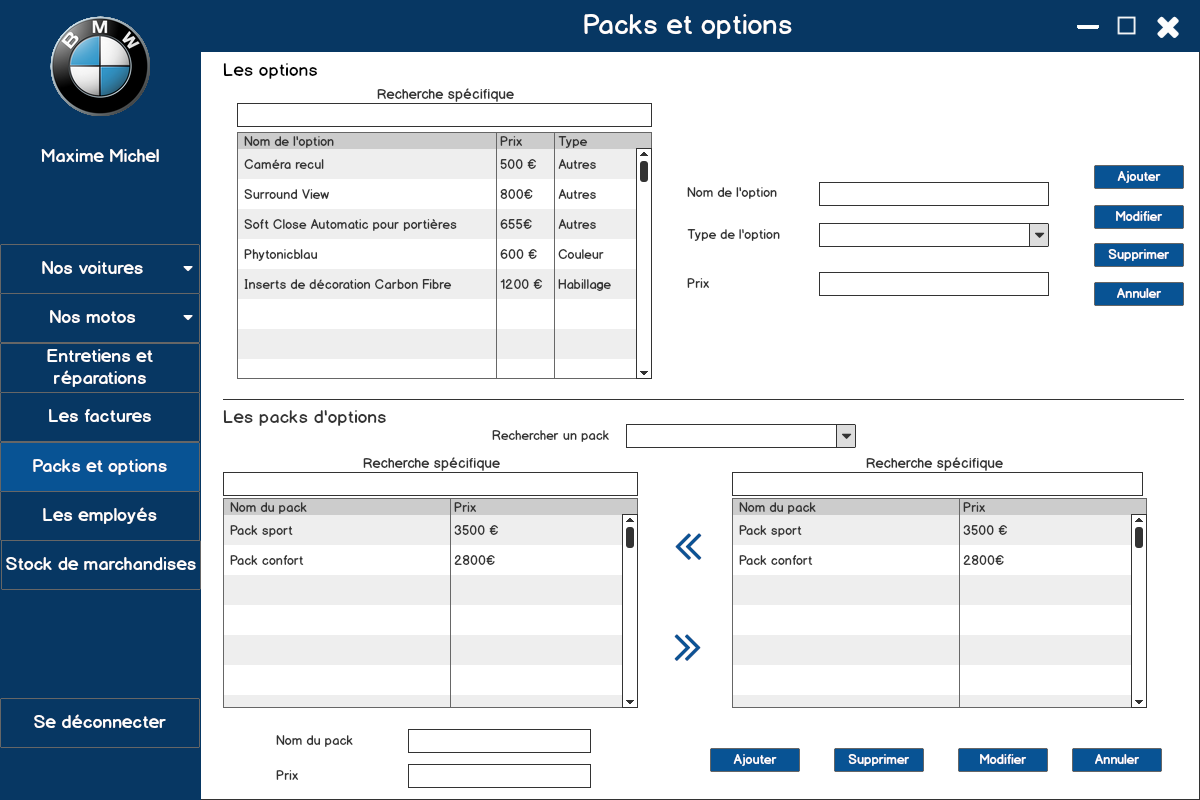
Pour commencer, lors de la première connexion à l’application, l’utilisateur devra créer le tout premier compte qui correspondra au compte de l’administrateur. L’email requis devra comprendre le domaine « @BMWEcaussinnes.com » afin d’être accepté.

##### Page de connexion



Une fois le premier compte créé, l’utilisateur peut ensuite se connecter en y indiquant son adresse email et son mot de passe.

##### Gestion des packs et des options

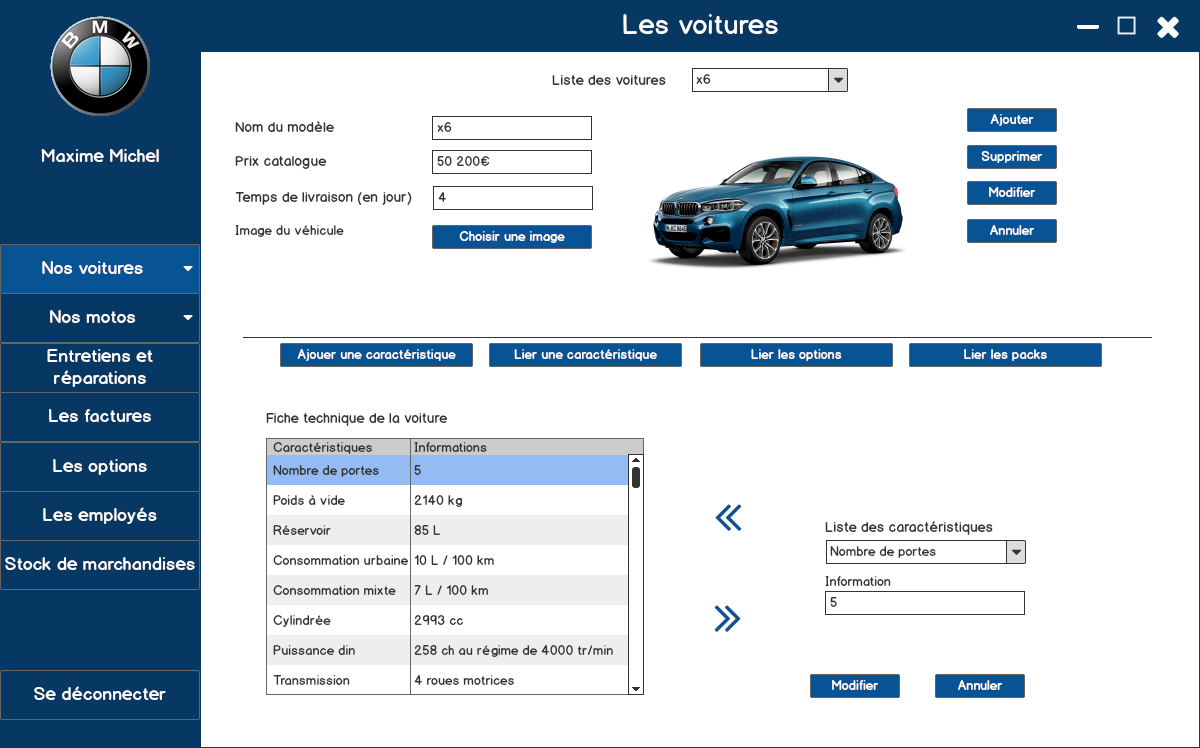


Ensuite, afin de pouvoir créer des voitures, il est conseillé en premier lieu d’ajouter les options et les packs disponibles sur les véhicules de la marque BMW. Pour se faire, l’utilisateur doit y encoder les options sur la première partie de l’écran en y indiquant sa catégorie (jantes, moteur, habillage, etc...). Ensuite, une fois les différentes options ajoutées, l’utilisateur ajoute le pack et son prix. Une fois le pack ajouté, il peut alors lier une option disponible dans la liste des options existantes à son pack grâce aux flèches, il peut également les retirer si ces options ne sont plus disponibles dans ce pack. Si ce pack est déjà utilisé par un véhicule, sans pour autant avoir été vendu, ce pack sera indisponible à la modification. Il faudra soit que l’utilisateur délie les packs du véhicule avant une modification ou qu’il supprime ce pack afin d’en recréer un nouveau.

Si une option ou un pack a été attaché sur un véhicule qui a été vendu et ensuite supprimé de la liste, la modification ne sera plus disponible. S’il désire changer une valeur de l’option ou modifier une option liée à un pack, il devra obligatoirement supprimer l’existant afin de recréer un nouvel enregistrement.

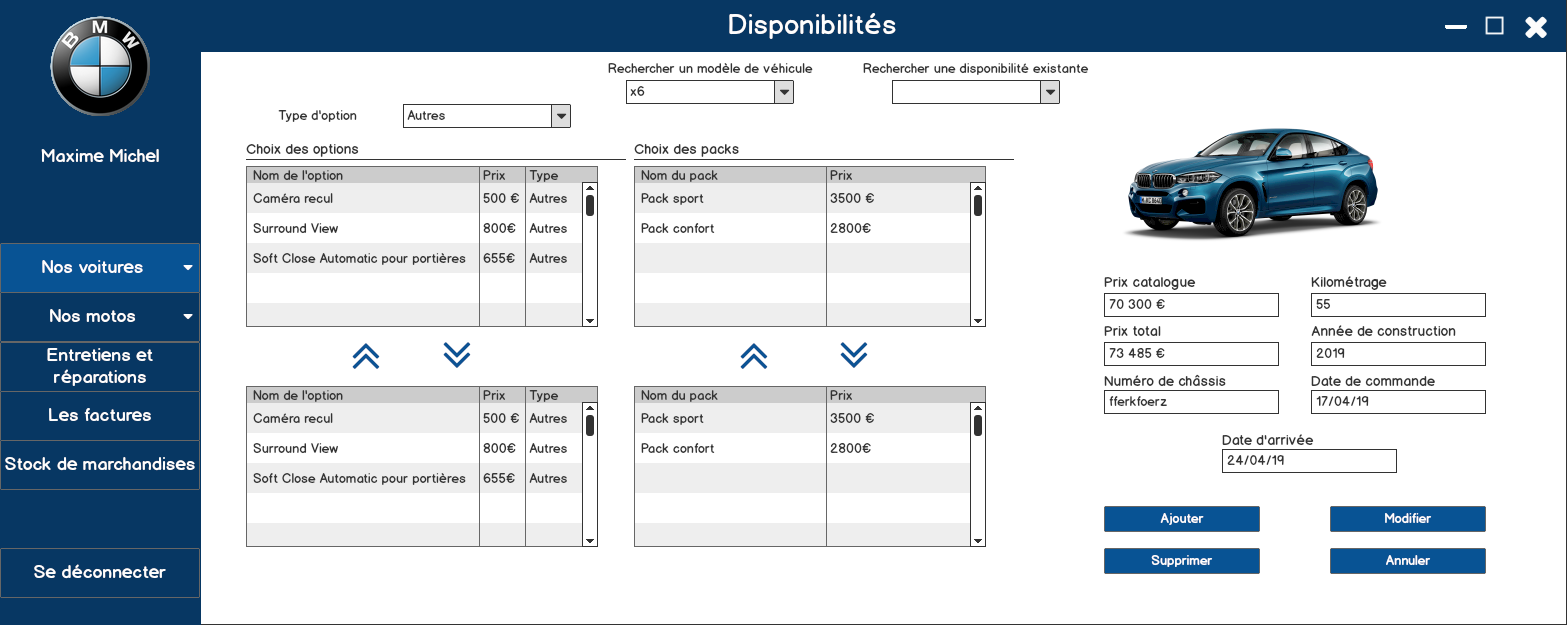
Afin de permettre une certaine souplesse, les options pourront être supprimées même si celles-ci sont disponibles dans un pack. Les options qui ont été supprimées et qui sont disponibles dans les packs ne seront pas retirées du pack, l’utilisateur devra le faire manuellement.

##### Gestion des véhicules



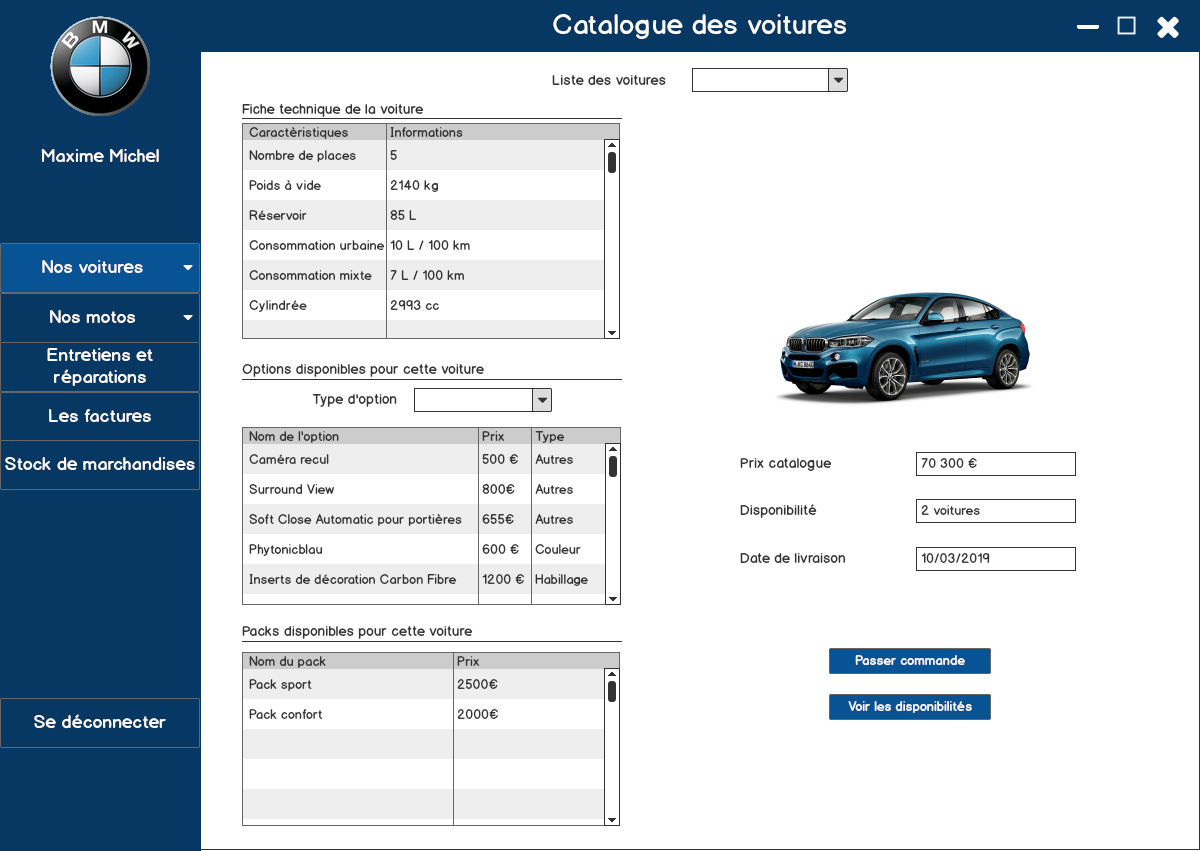
Une fois les options et les packs ajoutés, l’utilisateur peut continuer sur la gestion des véhicules ou sur l’ajout de nouvelle(s) caractéristique(s) à lier. Lors de l’ajout d’un véhicule, l’utilisateur devra choisir une image correspondant au modèle du véhicule. Une fois le véhicule ajouté, il aura plusieurs nouveaux écrans de disponibles. Le premier écran permettra de lier les caractéristiques et pour se faire, il devra choisir une des caractéristiques dans la liste déroulante et y indiquer l’information liée à cette caractéristique. La caractéristique sera ensuite liée au véhicule sélectionné sur la première partie de l’écran. Si la caractéristique n’existe pas encore dans la liste, l’utilisateur pourra toujours retourner sur l’écran d’ajout de caractéristique(s). Les 2 autres nouveaux écrans serviront à lier et délier les options et les packs. Les packs et les options n’étant liés qu’au véhicule, ils pourront grâce à cela retirer et ajouter les packs ou les options à leurs convenances, cette action n’impactera en rien les véhicules vendus utilisant ces options et ces packs.

##### Encodage de véhicule de concession



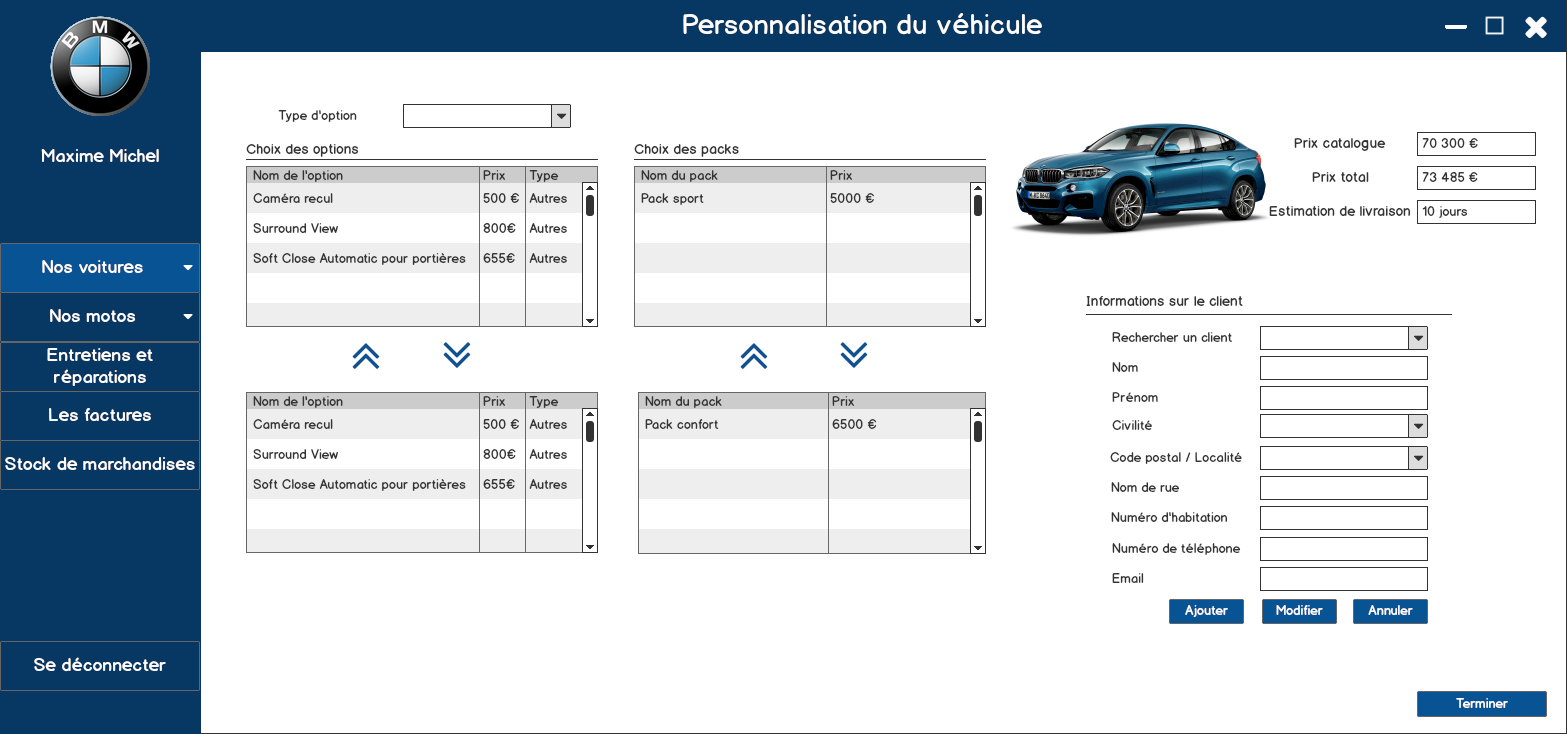
Une fois le véhicule encodé, une nouvelle fonctionnalité sera disponible, à savoir ajouter un véhicule à vendre qui appartiendrait à la concession. Pour personnaliser son véhicule, l’utilisateur aura la possibilité de choisir des options et des packs d’options. Pour commencer, il devra choisir dans les options après avoir sélectionné un type d’option. Ce filtre permet d’empêcher le vendeur de mettre 2 options contradictoires, par exemple, deux couleurs différentes pour un même véhicule. Le seul type d’options permettant une multiplicité d’options est le type « autres ». L’idée derrière ce type d’option est de permettre que les options cumulatives n’entrent pas en conflit avec une autre option, par exemple, les options électroniques. Si l’utilisateur choisit une option disponible dans un pack déjà choisi, un message d’interdiction est affiché expliquant dans quel pack appartient cette option. Quant à l’ajout d’un nouveau pack, si une des options déjà choisies entre en conflit avec une option disponible dans le pack, l’option choisie au préalable sera enlevée afin d’éviter les doublons d’options. Un prix sera affiché et mis à jour à chaque modification (nouvelle option, nouveau pack). Une fois la personnalisation terminée, celui-ci devra indiquer des valeurs propres au véhicule en cours de personnalisation, par exemple, combien de kilomètres le véhicule affiche au compteur. Il pourra en tout temps, modifier un véhicule existant et le supprimer si celui-ci n’a plus lieu d’être dans l’application ou devient erroné.

##### Catalogue des voitures



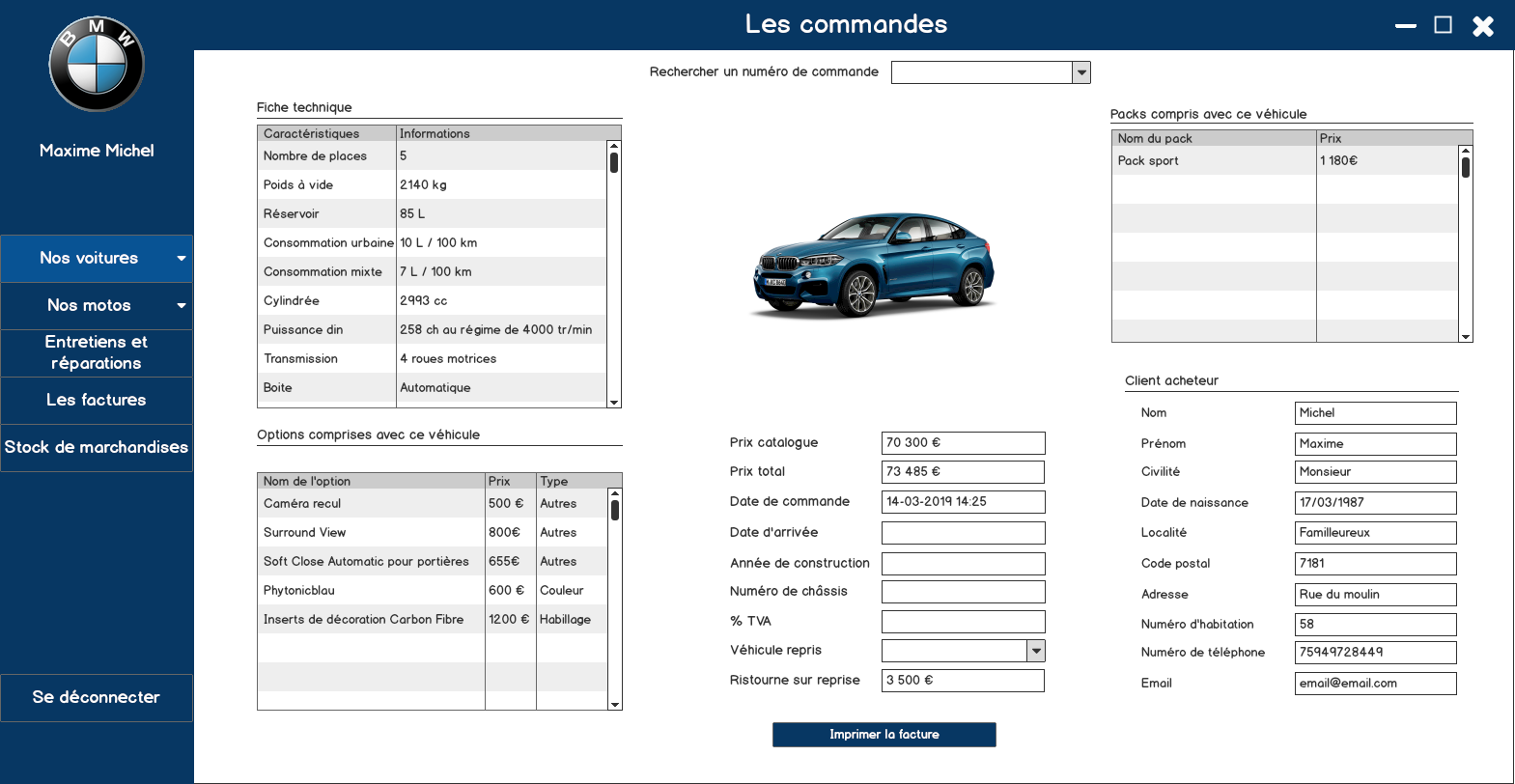
Une fois l’ajout d’un véhicule terminé, le véhicule sera disponible dans le catalogue correspondant à son type (voiture ou moto). Une fois le véhicule sélectionné avec la liste, le vendeur aura la description du véhicule, des packs et des options disponibles pour ce véhicule. Deux nouvelles fonctionnalités seront également disponibles, la première sera de passer une commande et la deuxième sera d’acheter un véhicule disponible en concession, qui aura été encodé au préalable. Un compteur de disponibilité est affiché afin de permettre au vendeur de voir si la concession possède ce véhicule.

##### Personnalisation d’un véhicule pour le client



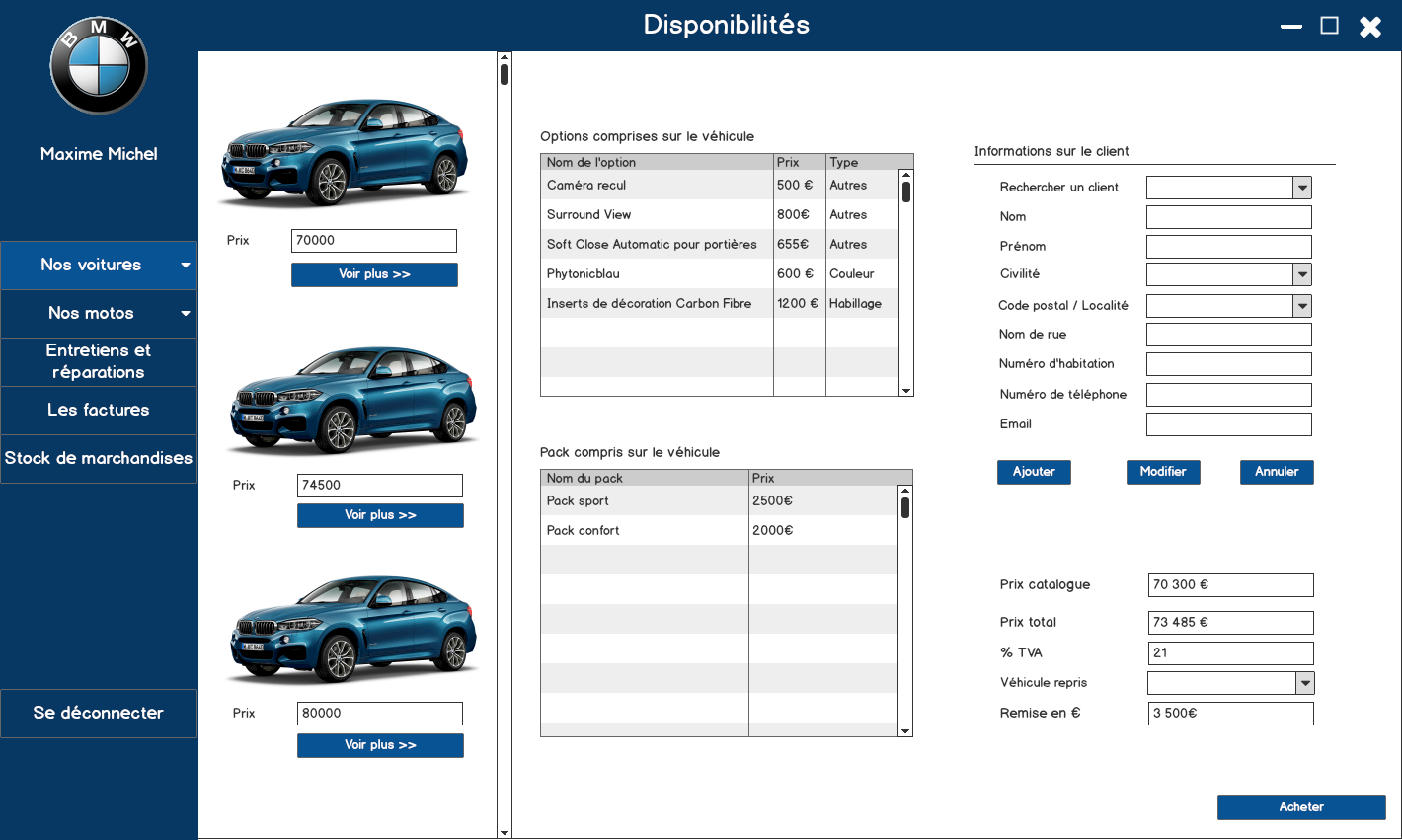
Une fois la sélection d’un véhicule dans le catalogue réalisé, le vendeur pourra personnaliser un véhicule que le client aura personnalisé au préalable ou en direct lors d’un entretien avec un vendeur. Comme pour l’écran de personnalisation d’un véhicule de concession, ce filtre permettra d’empêcher l’utilisateur de mettre 2 options contradictoires. Les seules options qui pourront disposer d’une multiplicité seront les options avec le type d’options « autres ». L’idée derrière ce type d’option sera de permettre le choix de 2 options qui ne rentre pas en conflit. Si le vendeur choisit une option disponible dans un pack déjà choisi, un message d’interdiction est affiché avec un message d’erreur expliquant dans quel pack appartient cette option. Quant à l’ajout d’un nouveau pack, si cette option a déjà été choisie et que le vendeur choisit un nouveau pack, l’option choisie au préalable sera enlevée afin d’éviter les doublons d’options. Un prix sera affiché et mis à jour à chaque modification (nouvelle option, nouveau pack). Ensuite, une fois la personnalisation du véhicule terminée, le vendeur devra choisir un client existant. Si le client n’existe pas encore, il pourra bien évidemment l’ajouter sur le même écran. Ensuite, lorsque le vendeur cliquera sur terminer, un bon de commande en format PDF sera créé avec tous les détails de la personnalisation afin d’avoir une preuve de commande de cette future transaction. Le PDF s’ouvrira alors automatiquement lui permettant d’avoir accès à une impression.

##### Les commandes des clients



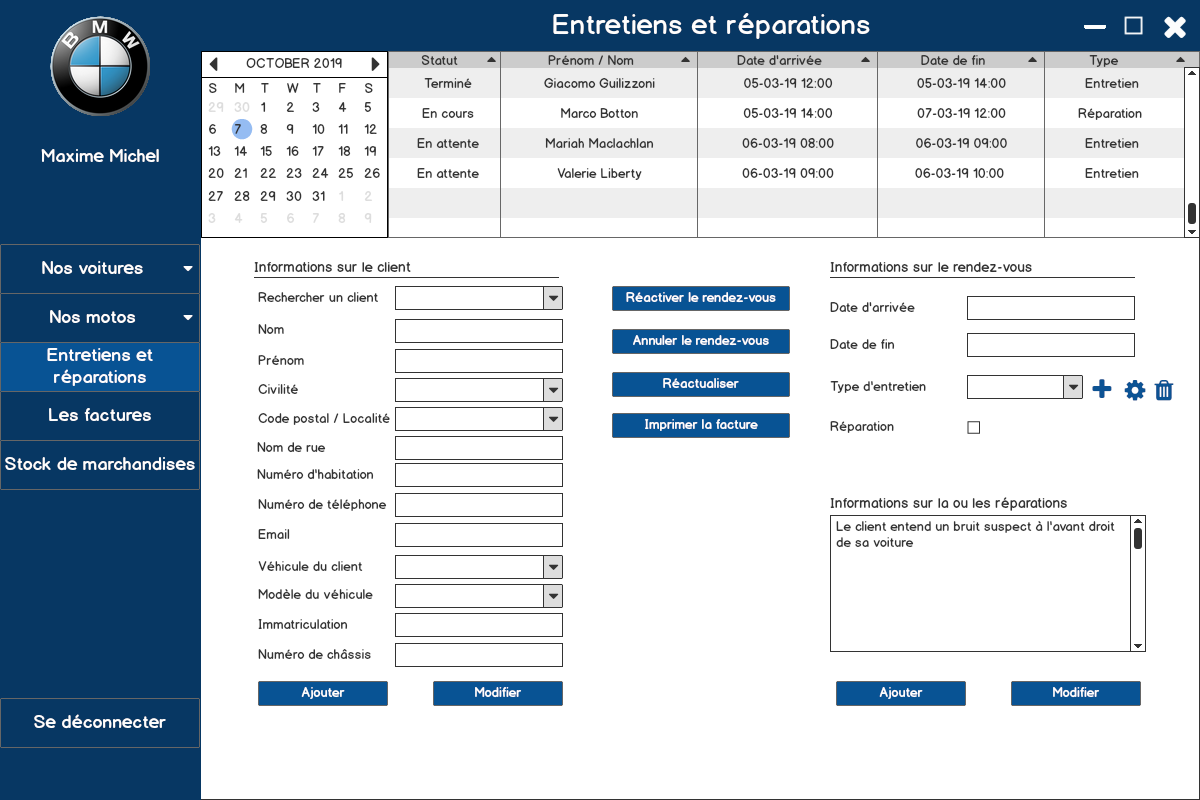
Une fois le véhicule arrivé en concession, le vendeur pourra aller sur la page de commande afin d’y ajouter les dernières informations nécessaires à l’impression du contrat de vente. Il devra également y indiquer si la concession reprend le véhicule du client et y indiquer la remise. Une fois le véhicule repris, celui-ci sera à nouveau disponible à la vente si l’administrateur désire ajouter ce véhicule une nouvelle fois à l’application. Une fois toutes ces données encodées, l’utilisateur n’a plus qu’à cliquer sur le bouton d’impression afin d’avoir un document PDF de la facture de vente.

##### Les disponibilités de la concession



Sur cette page sont affichés tous les véhicules appartenant à la concession. Une description des informations sur la personnalisation du véhicule sera affichée lorsqu’il choisit un véhicule afin de permettre au vendeur de connaître les différents éléments disponibles sur ce véhicule. Si le client désire acheter ce véhicule, le vendeur devra choisir un client existant ou créer un nouveau client. Si ce client dispose d’un véhicule de marque BMW celui-ci pourra être repris par la concession. Une remise lui sera alors proposée. Une fois toutes ces informations encodées, il pourra cliquer sur acheter afin d’avoir le contrat de vente en format PDF.

##### La prise de rendez-vous pour les entretiens et les réparations

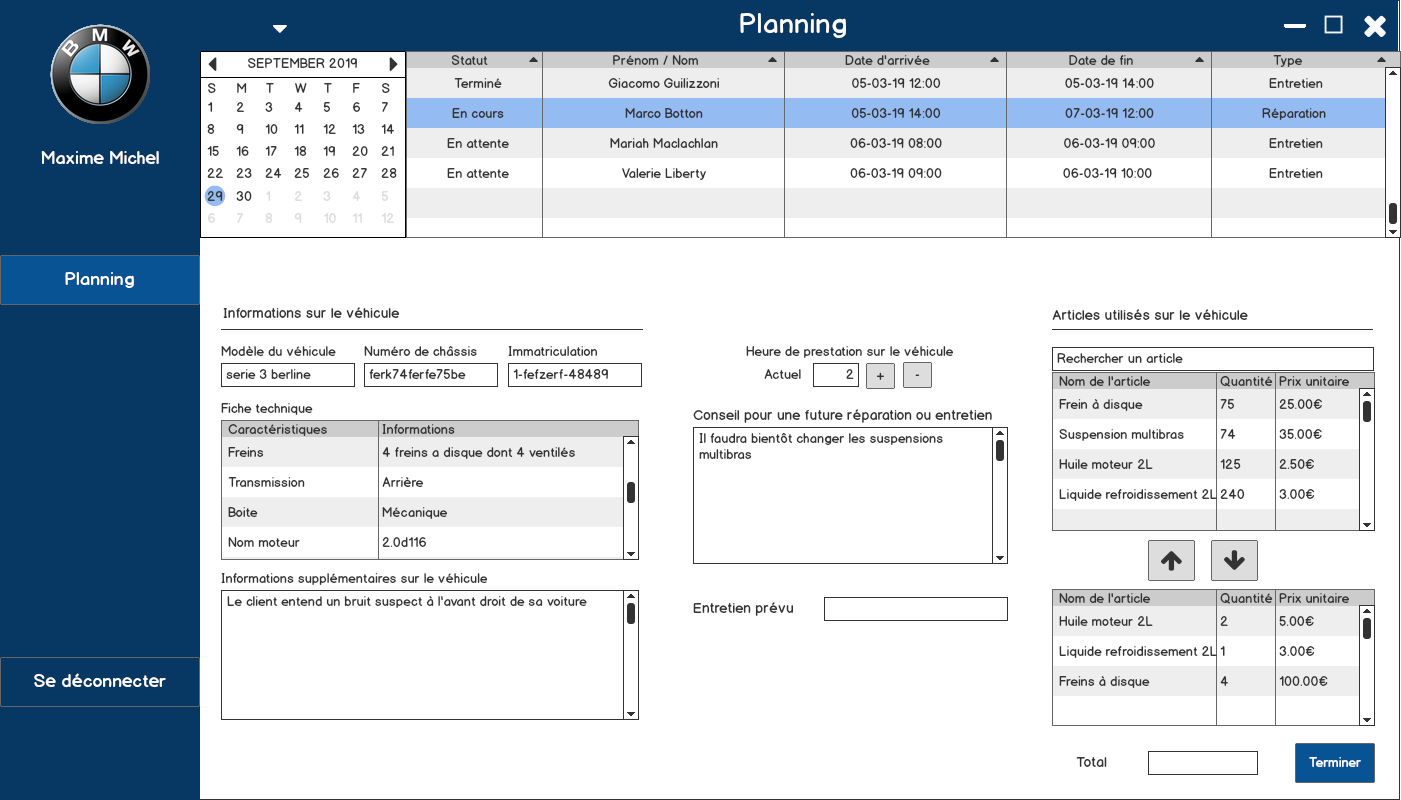


Sur cette page, le vendeur ou l’administrateur pourra prendre les rendez-vous des clients pour des entretiens et/ou des réparations. Pour prendre rendez-vous, il devra choisir un client dans la liste déroulante ou y ajouter un client. Ensuite, il devra choisir un des véhicules appartenant au client ou y ajouter son véhicule. Une fois le véhicule choisi, il devra choisir la date et l’heure du début et de la fin du rendez-vous. Une fois ces données encodées, il aura la possibilité de choisir un entretien dans la liste, si l’utilisateur est l’administrateur, une petite fenêtre apparaitra au moment du clic sur un des boutons lui permettant de gérer les entretiens en les ajoutant, en les modifiant ou en les supprimant. Les rendez-vous déjà pris qui sont en cours ou déjà terminés ne seront pas impactés par cette suppression tant que la modification de l’entretien n’est pas effectuée.

Pour les réparations, l’utilisateur devra cocher le bouton afin d’activer la zone de texte et d’y indiquer les différentes réparations à effectuer ou une indication sur les emplacements à vérifier afin que le mécanicien puisse avoir l’information. Une fois les données encodées, il devra cliquer sur ajouter. Si la date de fin de l’entretien est passée à « Terminé », aucune modification ne sera autorisée et le bouton d’impression sera disponible. Si ces rendez-vous ne sont pas terminés, l’utilisateur pourra annuler le rendez-vous et si les rendez-vous ont été annulés, mais que la date est toujours valide, il pourra alors réactiver le rendez-vous.

Si la modification est possible, une première vérification sera effectuée afin de s’assurer que le véhicule du client n’a pas encore été référencé dans une autre facture. Si une facture est existante, l’utilisateur ne pourra pas modifier les informations concernant le véhicule. Si le véhicule change d’immatriculation, le véhicule sera considéré comme nouveau et devra être ajouté comme tel. Cependant, si le véhicule est référencé une seule fois mais que le rendez-vous a été annulé, les informations du véhicule seront modifiables. Dans le cas où le rendez-vous a été pris pour un autre véhicule que celui concerné, cette modification sera toujours possible via la liste des véhicules du client.

##### La page de planning du mécanicien

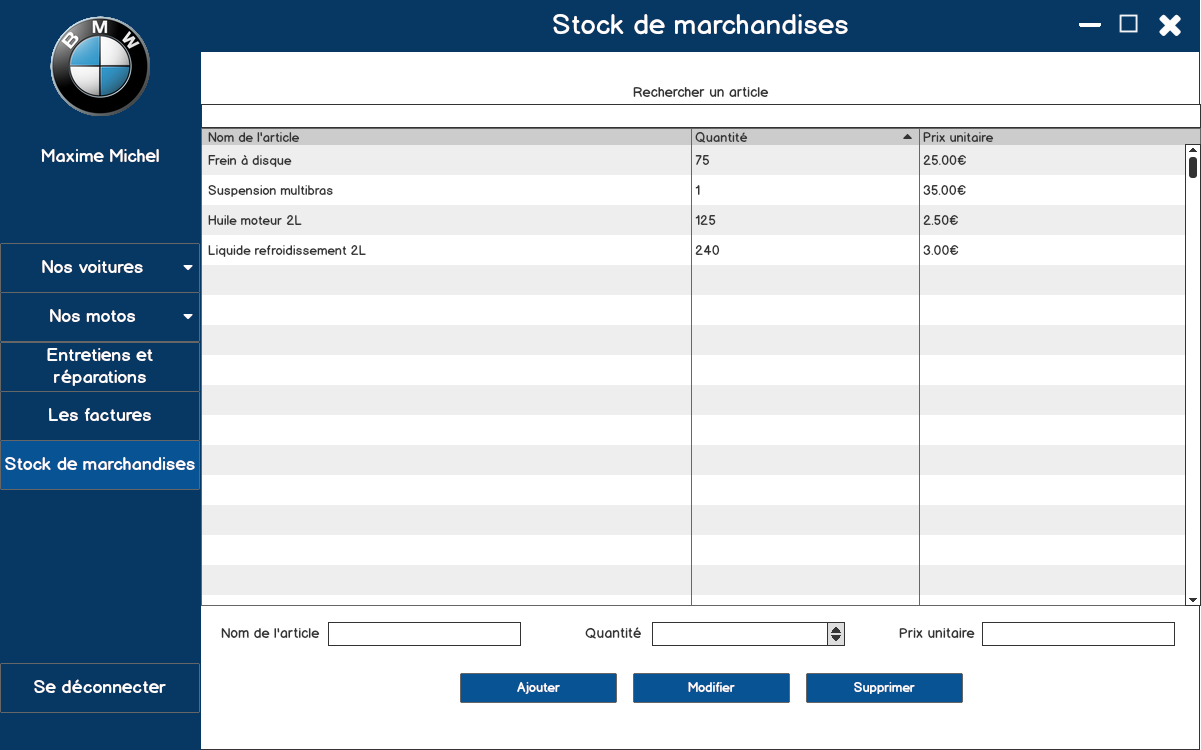


Cette page est exclusivement réservée aux mécaniciens et leur permet d’avoir accès aux différentes tâches à réaliser sur le véhicule ainsi qu’une fiche technique du véhicule en question s’il a besoin d’informations concernant une pièce. L’identification du véhicule y sera également indiquée afin de lui donner l’information du véhicule sur lequel les tâches doivent être effectuées.

La partie encodage de l’utilisateur consiste à indiquer le nombre d’heures qu’il passe à réparer et/ou à entretenir le véhicule. Il devra également y indiquer les produits utilisés afin de pouvoir facturer au client les différents articles. Une fois qu’il estime que la réparation est terminée, il devra cliquer sur le bouton « Terminé ».

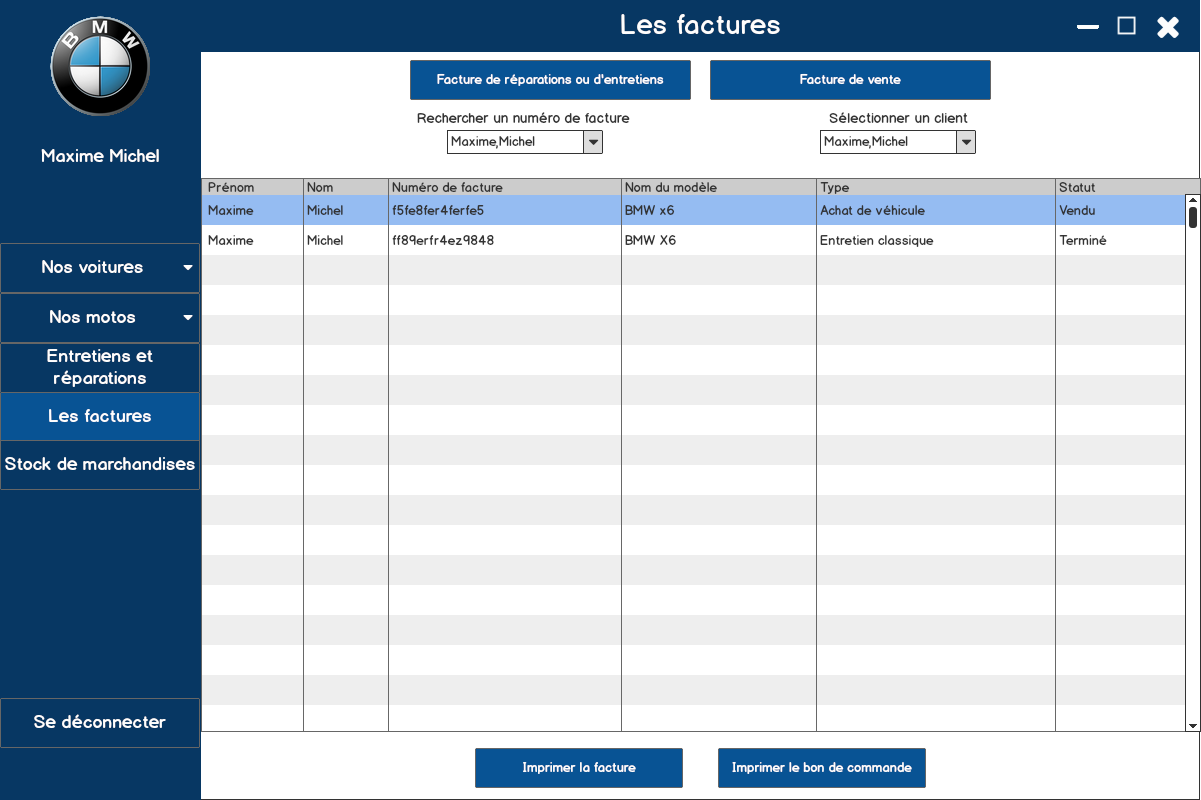
Il aura aussi une partie facultative lui permettant de laisser un message au vendeur afin qu’il avertisse le client des conseils du mécanicien sur des entretiens ou des éventuelles réparations à prévoir.

##### Gestion du stock des articles



Cette page consiste à la gestion du stock des différents articles, autant des pièces que des huiles et autres articles d’entretien de véhicule. Afin de supprimer un article, la quantité de l’article devra être à 0, sinon la suppression sera impossible. La recherche quant à elle, se portera sur le nom de l’article.

##### Les factures

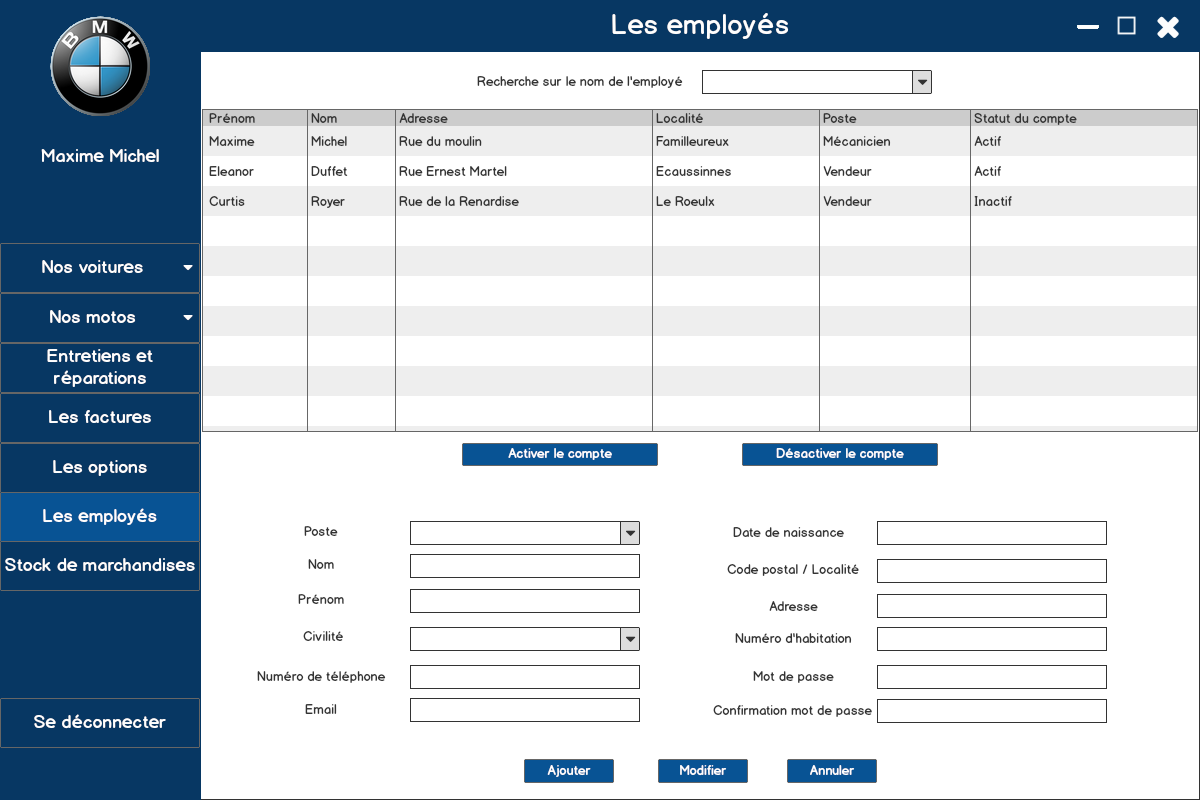


Cette page permet à l’utilisateur de récupérer les factures qui sont sauvegardées en permanence dans la base de données.

Pour la vente d’un véhicule, si la vente n’a pas encore eu lieu, l’impression de la facture ne sera pas disponible, car elle sera inexistante. L’impression du bon de commande restera malgré tout accessible en tout temps si une réimpression est nécessaire. Si le véhicule a été vendu, les 2 boutons seront disponibles.

L’autre fonctionnalité de cette page est l’impression des factures pour les entretiens et/ou les réparations terminés. Les filtres sont configurés pour agir sur les 2 listes, en même temps ou sur une seule liste.

##### Gestion des employés



Sur cette page réservée à l’administrateur, il aura la possibilité d’ajouter un nouvel employé en lui affectant un rôle (vendeur ou mécanicien). Pour cette page, j’ai décidé de ne pas laisser l’administrateur supprimer un employé afin de toujours garder une trace visuelle de la personne. Mais à la place, j’ai mis à disposition la possibilité de désactiver un compte et de le réactiver si, par exemple, l’employé changeait d’entreprise et revenait par la suite, ou pour les contrats temporaires (intérimaire(s)). Pour la modification du mot de passe, il faudra qu’il le confirme afin de vérifier qu’ils sont identiques. Aucun mot de passe ne sera visible. Lors de la sélection, les deux champs seront vides et ne devront être remplis qu’en cas de modification du mot de passe.

## Structure du code

### Applications utilisées pour le développement

* Visual studio

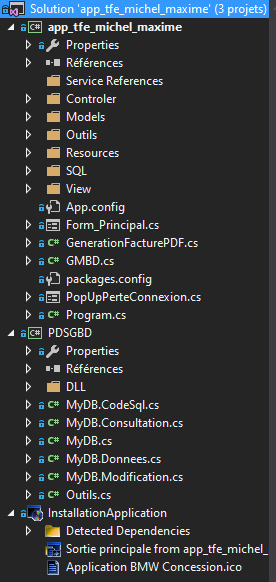
Afin de développer mon application, j’ai fait le choix d’utiliser visual studio. La raison de ce choix est justifiée par ma préférence personnelle et en fonction de mes compétences. De plus, ce choix était beaucoup plus adapté à mon idée d’application, le but de celui-ci étant de faire une application métier. J’ai donc opté pour la technologie WinForms telle que nous l’avons apprise durant ce cursus scolaire. Même si au niveau des effets visuels, il est souvent critiqué par rapport aux possibilités de WPF, il reste un outil très agréable d’utilisation avec beaucoup de choix de personnalisation.

* Wamp server

Pour la gestion de ma base de données, le choix s’est porté sur l’application de WampServer. Lors des différents cours nécessitant une base de données, nous avons appris à utiliser ce gestionnaire qui, de mon point de vue, répond à tous les critères nécessaires au bon fonctionnement de mon application dans la mesure des différents tests à réaliser pour ce mémoire. Lorsque l’application passera en ligne, un autre gestionnaire de données sera nécessaire.

### Description des différents dossiers et de leurs contenus

##### Description générale des dossiers du code

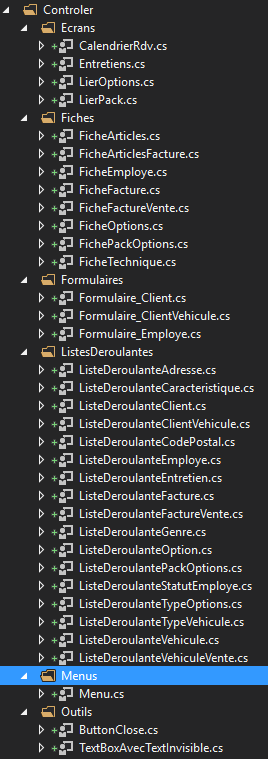
La structure de mes dossiers est composée de 3 dossiers principaux, suivant l’idée des principes du MVC mélangé à du MVVM. Ces 3 dossiers sont « Models », « View » et « Control ». Une description plus détaillée de ces dossiers sera expliquée dans les prochains chapitres.

On peut également voir que mon projet est composé d’un sous projet « PDSGBD ». Ce projet a été réalisé lors du cours ayant le même intitulé et du cours de SGBD. Ce projet permet une connexion sécurisée à la base de données. Chaque classe a sa propre spécification, consulter, modifier, transformation de code en valeurs SQL et un objet de connexion à la base de données personnalisée.

Ce projet permet également de prévenir l’utilisateur si un problème de connexion à la base de données survient. Pour les problèmes de connexion lors de l’utilisation, une autre fenêtre pop-up créée par moi-même apparaitra prévenant l’utilisateur de sa perte de connexion et le forçant ainsi à se reconnecter avant de poursuivre son utilisation.

Le dernier projet a été créer avec une extension de visual studio qui est : « Microsoft Visual Studio 2015 Installer Projects ». Cette extension m’a permis de créer un exécutable d’installation de l’application. Lors de l’installation, l’utilisateur aura le choix de l’emplacement où créer le dossier comprenant les différents éléments nécessaires à son fonctionnement. Un raccourci sera également créé sur le bureau.

##### Description du dossier « Controler »

* Ecrans

Les écrans sont des parties de code permettant une séparation entre le code et leurs pages principales. Le fait de séparer ces contrôles m’a permis d’avoir un code plus éclairci avec moins de ligne. Par exemple, « LierOptions.cs » et « LierPack.cs » sont des écrans se superposant, de ce fait, il n’est pas toujours aisé d’y accéder dans la partie de personnalisation visuelle de visual studio. De plus, la page comprenant beaucoup d’actions possibles, les fonctions d’ajouts, de modifications et de suppressions commençaient à être nombreuses et il était donc difficile de s’y retrouver.

* Fiches

Les fiches sont composées d’une zone de texte afin de permettre un filtre sur un des champs de données visibles de la fiche. Elles disposent également d’un tableau permettant un affichage de plusieurs champs de la base de données. L’espace des différentes colonnes est géré en fonction du nombre de colonnes et du nombre de titres. Ces fiches permettent une sélection afin d’avoir accès à l’enregistrement choisi par l’utilisateur.

* Les listes déroulantes

Les listes déroulantes sont des listes classiques, elles permettent l’affichage de plusieurs enregistrements et une sélection d’un enregistrement. Une recherche spécifique est possible sur ces listes avec une autocomplétion en fonction des champs existants dans la liste.

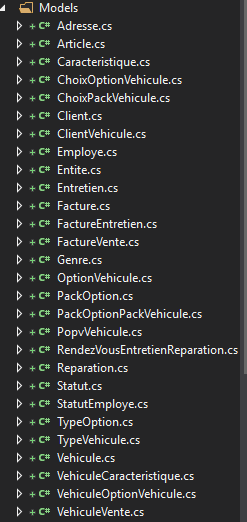
* Menus

Dans le dossier « menus », seront placés les divers menus possibles pour l’application. Dans le cas actuel, il n’existe qu’un seul menu, c’est le menu principal placé sur chaque page sur le côté gauche de celles-ci.

* Outils

Dans les outils seront compris tous les types de boutons personnalisés. Par exemple, le « TextBoxAvecTextInvisible » est une zone de texte avec un texte explicatif en fond et qui disparait une fois que l’utilisateur entre dans celle-ci.

##### Description du dossier « Models »

Dans ce dossier, sont comprises les différentes classes représentatives de la base de données et chacune de ces classes contient une énumération de chaque champ de la table correspondante.

Chaque énumération aura droit à sa propriété publique permettant un accès externe aux membres privés.

Toutes les classes comprennent 3 types de constructeurs :

* + Un constructeur par défaut.
* Un constructeur spécifique construit par paramètre(s).
* Un constructeur spécifique construit par une lecture d’enregistrement en base de données.

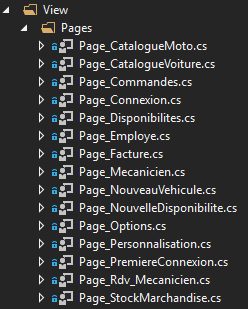
Elles disposent également d’une méthode permettant l’assignation de ses divers membres afin d’effectuer l’ajout d’un nouvel enregistrement dans la base de données.

Chaque modèle hérite de la classe « Entite » permettant l’utilisation des différentes méthodes proposées par la classe, par exemple, les réactions sur certains événements comme l’ajout d’un nouvel enregistrement. Le développeur aura donc la possibilité lors de la construction d’un nouvel objet compris dans la liste des modèles, de lier des méthodes spécifiques à ces objets. Dans ces méthodes, sont comprises : les méthodes de vérifications, avant et après les changements, et elles permettent également de savoir si l’objet est valide. Par valide, est entendu la construction complète de l’objet. À chaque passage dans une assignation de propriété, si elle est valide, elle enverra à la méthode de gestion de validation d’objet, un booléen égal à « true » permettant de spécifier pour chaque propriété de l’objet, que la validation sur l’assignation est valide et qu’elle a bien été enregistrée.

Lors d’un nouvel enregistrement, c’est également cette classe qui permettra la génération du nouvel identifiant grâce à sa valeur de retour de la méthode d’enregistrement.

Cette classe permettra également la préparation des autres requêtes. Parmi ces requêtes, seront comprises : les modifications et les suppressions avec l’utilisation d’une classe généralement appelée « Factory » qui est un principe du modèle MVVM où sont stockées la quasi-totalité des requêtes SQL. Dans l’application, cette classe sera appelée « GMDB », celle-ci sera décrite plus tard dans la description du code.

##### Description du dossier « View »



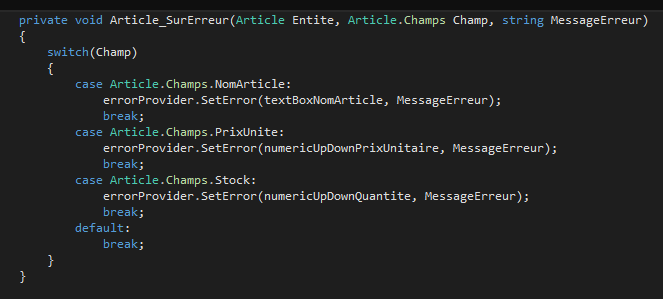
Dans ce dossier sont comprises toutes les pages de mon application. Lors de l’instanciation de mon application, mon formulaire principal fait appel à la page de première connexion ou de connexion à l’application. Une fois le changement de page, une fonction fera appel à une autre page afin de modifier la page courante. Cette méthode permet d’avoir un accès en tout temps à la page qui est actuellement affichée. Elle me permet surtout de remplacer une page courante par une autre afin de sécuriser l’affichage de pages en bloquant l’affichage de plusieurs pages au même moment.

##### Description du dossier « Outil »

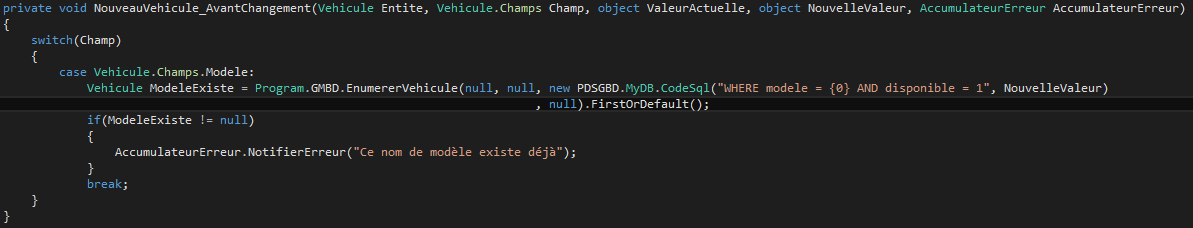
Dans ce dossier sont comprises des classes comprenant différentes méthodes permettant une gestion des erreurs et dispose également d’outils dont on pourrait avoir besoin lors de la programmation, par exemple, une méthode permettant de vérifier la fin de mes emails.

L’accumulateur d’erreur(s) me permet l’affichage des erreurs multiples utilisant le même « errorProvider ». Toutes les erreurs emmagasinées lors de la construction d’un objet seront stockées et affichées l’une à la suite de l’autre dans la même bulle d’indication. Si la méthode héritée de la classe entité utilise la méthode « SurErreur », chaque énumération avec une erreur sera rédigée vers l’outil indiqué dans cette fonction. Cette méthode est donc nécessaire au bon fonctionnement de mes modèles étant donné que chaque assignation de propriété répond à un certain nombre de critères spécifiques à chaque membre.

##### Exemple de méthodes possibles avec le modèle « Entite »



Voici un exemple de méthode possible avec l’héritage de la classe « Entite », cette méthode permet de lier un outil, un message et une énumération. Dans l’exemple ci-dessus, on peut voir que chaque énumération dispose de son propre message. Ces messages viendront en fait des tests effectués lors de l’assignation d’une propriété et seront indiqués à droite de l’outil.



Ci-dessus est un exemple d’utilisation de la méthode avant changement, cette méthode permet de faire des vérifications supplémentaires telles que les tests d’unicité. Dans le cas présent, je vérifie que le nom du modèle n’existe pas encore et s’il existe, que la validité du véhicule existant n’est pas un véhicule supprimé. Si l’erreur survient, le message indiqué sera ajouté au message existant sinon, il sera le seul message indiqué.

### Explications des algorithmes du code

##### Personnalisation des véhicules avec les options et les packs

Gestion des choix d’options

Au moment de l’ajout d’une nouvelle option lors d’une personnalisation, trois conditions sont vérifiées:

1. Une vérification est effectuée dans la liste avec un « foreach » afin de vérifier si cette option a déjà été choisie auparavant et si l’option existe, un message d’erreur est envoyé à l’utilisateur.
2. Une vérification est effectuée dans la liste en fonction des types d’options. Le seul type permettant un choix multiple est le type « Autres », les autres options disposent d’une unicité.
3. Une vérification est effectuée dans la liste des packs. Pour chaque pack de la liste, les options du pack sont vérifiées avec un « foreach » afin de vérifier que l’option n’est pas déjà disponible dans le pack. Si l’option existe dans un pack choisi, un message avertit l’utilisateur en lui précisant le nom du pack disposant de cette option.

* Pseudo code :

Début

Option existe -> recherche dans la liste si le pack y est déjà

Si ((L'option n’existe pas) ou (Nom de l’option est égal à « Autres »))

Si (Pack choisi est supérieur à 0)

On boucle sur chaque pack dans la liste des packs

On boucle sur chaque option dans le pack indicé au-dessus

Si l’option existe -> on retourne le message d’erreur indiquant que l’option est disponible dans le pack où l’on se trouve, et on indique un booléen pour la condition suivante

Si l’option n’existe pas

On ajoute l’option

On actualise les informations

Sinon

On indique que le type d’option a déjà été choisi

Sinon

On indique que l’option est déjà dans la liste des choix d’options

Fin

Gestion des choix de packs

Au moment de l’ajout d’un nouveau pack lors d’une personnalisation, trois conditions sont vérifiées :

1. Une vérification est effectuée sur l’unicité des packs d’options choisis, si ce pack a déjà été choisi, un message l’indiquera à l’utilisateur.
2. Une vérification est effectuée sur la fiche des options choisies au préalable afin d’éviter les doublons entre les options choisies et les options disponibles avec le pack. S’il y a un doublon, les options se retirent automatiquement.
3. La dernière vérification se fait sur les packs. Si un pack choisi comprend une option déjà présente dans un ou plusieurs pack(s) choisi(s) au préalable, le nouveau pack choisi remplacera le pack comprenant cette option.

Pour la page des disponibilités, une vérification supplémentaire est effectuée afin de vérifier si le véhicule a déjà été créer au préalable. De cette façon, un traitement particulier peut être effectué avec une modification directement dans la base de données. Le pseudo code retranscrit en dessous correspond au code des personnalisations pour les clients.

* Pseudo code :

Début

Pack existe -> Recherche dans la liste si le pack y est déjà

Si (Pack n’existe pas)

On boucle sur les options du pack choisi

On boucle sur les options choisies

Si (L’id de l’option du pack choisi est égal à une option choisie)

On diminue le prix affiché

On retire de la liste

On actualise la fiche d’options choisies

Si (Au moins un pack a déjà été choisi)

On boucle sur les packs choisis

On boucle sur les options des packs indicés au-dessus

Si (L’option du pack choisi est égal à l’option d’un des packs indicés et que l’id du pack est différent du pack sélectionné)

On retire le pack avec le doublon

On actualise le prix en retirant le prix des packs retirés

On indique qu’un ou plusieurs pack(s) ont été retiré(s)

On indique que la validation a été acceptée

Si (la validation a été acceptée)

On indique le message de validation

On ajoute le pack à la liste

On rajoute le nouveau prix du pack

On actualise l’affichage des packs

Si (Le pack existe)

On indique que le pack a déjà été choisi

Fin

Changement de sélection sur les rendez-vous

Au moment de la sélection d’un rendez-vous, six conditions sont vérifiées afin de gérer les différentes fonctionnalités possibles à l’utilisateur :

1. On vérifie si le statut du rendez-vous est égal à « En attente », qu’il n’est pas égal à « Terminé » et que la date du début du rendez-vous est supérieure à l’heure actuelle
2. Ensuite, on vérifie que le statut est égal à « Annuler », qu’il est différent de « Terminé » et que la date du début est supérieure à l’heure actuelle
3. Ensuite, on vérifie que la date du début est inférieure à l’heure actuelle
4. Ensuite, on vérifie si le statut est égal à « Terminé »
5. On vérifie si le rendez-vous comprend une réparation
6. Ensuite, on vérifie si le client n’a jamais été référencé afin de bloquer certaines modifications sur les informations du véhicule choisi.

* Pseudo code :

Début

Si ((Statut du rendez-vous est égal à « En attente ») et (Statut du rendez-vous n’est pas égal à « Terminé ») et (Date de début est supérieure à la date et l’heure du jour))

On désactive le bouton de réactivation de rendez-vous

On active le bouton d’annulation de rendez-vous

On désactive le bouton d’ajout de rendez-vous

On active le bouton de modification du rendez-vous sélectionné

On désactive le bouton d’impression

Sinon ((Statut du rendez-vous est égal à « Annuler ») et (Statut du rendez-vous n’est pas égal à « Terminé ») et (Date de début est supérieure à la date et l’heure du jour))

On active le bouton de réactivation de rendez-vous

On désactive le bouton d’annulation de rendez-vous

On désactive le bouton d’ajout de rendez-vous

On active le bouton de modification du rendez-vous sélectionné

On désactive le bouton d’impression

Sinon ((La date du début est inférieure à la date et l’heure actuelle) et (Statut est égal à « Annuler »))

On désactive le bouton de modification des clients

On désactive le bouton de modification de rendez-vous

On désactive le bouton d’ajout de rendez-vous

On désactive le bouton d’annulation du rendez-vous

On désactive le bouton d’impression

Sinon (Si le statut du rendez-vous est égal à « Terminé »)

On désactive le bouton de modification du client

On désactive le bouton de modification du rendez-vous

On désactive le bouton d’ajout de rendez-vous

On désactive le bouton d’annulation de rendez-vous

On active le bouton d’impression de la facture du rendez-vous

Si (Le véhicule sélectionné comprend une réparation)

On active la zone de texte des réparations

On active le bouton d’activation du rendez-vous

Sinon

On désactive la zone de texte

On désactive le bouton d’activation du rendez-vous

On vérifie si le véhicule existe dans une facture déjà « Terminé » ou « En attente »

Si (Une référence existe pour ce véhicule)

On désactive la zone de texte de l’immatriculation

On désactive la zone de texte sur le numéro de châssis

On désactive la liste de choix du modèle de véhicule

Else

On active la zone de texte de l’immatriculation

On active la zone de texte sur le numéro de châssis

On active la liste de choix du modèle de véhicule

Fin

### La génération des factures

##### Description du framework

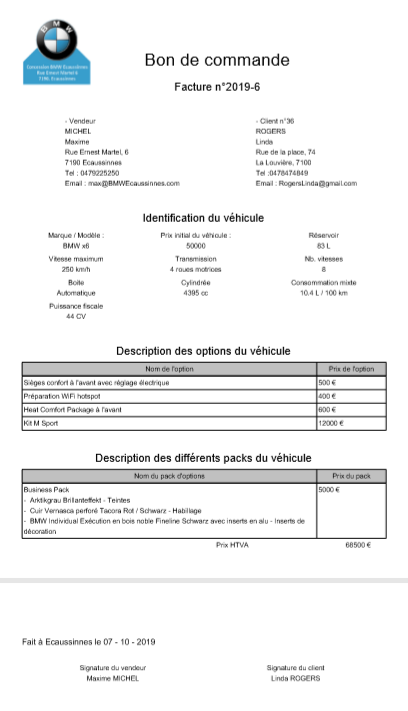


Pour la création de mes factures, j’ai décidé d’utiliser un framework permettant une précision et une propreté de création de documents PDF. IText est une interface de programmation partiellement open source suivant une licence AGPL. De ce fait, si l’application venait à être commercialisée, les frais d’une licence iText devront être pris en compte afin de ne pas avoir un code open source. Il existe une autre solution qui est de suivre la même licence pour le code, ce qui ferait que le code deviendrait open source. Je pense que pour une meilleure souplesse du choix d’une licence, la première solution est la plus adaptée.

##### Description de la création des factures

Pour la création d’une facture, nous devons créer un premier document sur le bureau afin d’y stocker les nouveaux documents PDF. Pour ce faire, une vérification du nom de document sur le bureau est effectuée afin de vérifier son existence. Si ce document est déjà existant, alors les nouveaux fichiers iront dans le document existant, sinon un dossier sera créé. Ensuite, nous devons nous assurer que le nom du fichier n’est pas déjà utilisé afin de ne pas écraser le fichier existant. Pour cette vérification, j’ai utilisé une bibliothèque de visual studio déjà existante, appelée par l’instruction « using system.IO ». Si un des fichiers existe déjà, un comptage du nombre de fichiers avec le nom et l’indice de comptage sera effectué afin d’y placer l’indice du nombre de fichiers entre parenthèses (exemple : facture(4).png). Ensuite, je crée les objets composant la facture et pour ce faire, un principe de tableau sera utilisé afin de placer correctement les éléments. À titre de comparaison, bootstrap utilise le même système. Je place donc les éléments dans des cellules qui sont ensuite ajoutées dans mes différents tableaux. Des tableaux auront la spécificité d’être affichés à l’utilisateur. Ils auront pour but d’afficher les options choisies, les packs choisis ou encore les articles utilisés sur les entretiens et les réparations. Un système d’énumération permet la création de chaque élément avec l’aide d’une boucle « ForEach ». Dans cette boucle, chaque élément fera l’objet d’une création de nouvelles cellules qui seront ensuite ajoutées dans mon tableau. Des exemples de factures seront expliqués à la page suivante.

##### Exemple d’un bon de commande



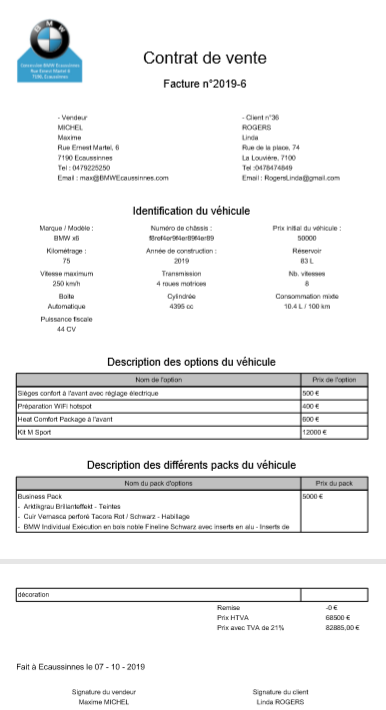
Voici l’exemple d’un bon de commande qui s’ouvre après la personnalisation d’un véhicule. Comme nous pouvons le voir, les informations de la concession et du client sont indiquées sur le dessus du document. Ensuite, une description adaptative en fonction des différentes caractéristiques du véhicule en commande sera décrite.

Pour l’affichage des options, chaque option choisie lors de la personnalisation sera indiquée avec son prix.

Pour l’affichage des packs, chaque pack aura une description des différentes options comprises dans les packs choisis.

Une fois que le client confirme que le document comporte les éléments qu’il avait choisis, il ne restera plus qu’aux 2 parties de signer afin d’acter la commande du véhicule.

##### Exemple d’un contrat de vente



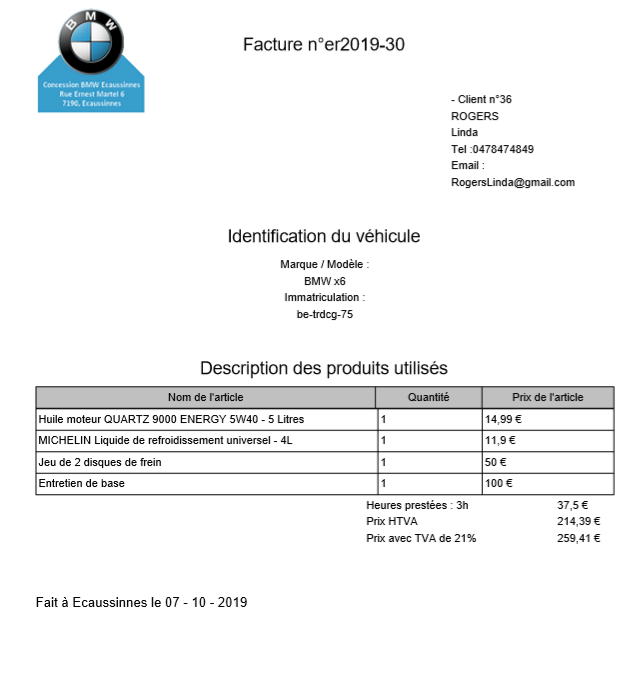
Voici l’exemple d’un contrat de vente après l’encodage de la réception des véhicules sur la page « Les commandes ».

Comme pour le bon de commande les informations concernant les deux personnes sont indiquées sur le dessus.

Même principe pour la description, sauf que pour ce document, des informations fixes seront rajoutées en plus, par exemple, le kilométrage du véhicule au moment de la vente, l’année de construction du véhicule et le numéro de châssis qui était encore inconnu au moment de la commande.

Pour les options et les packs, les mêmes informations que sur le bon de commande seront indiquées.

##### Exemple d’une facture d’entretien et/ou de réparations



Voici l’exemple d’une facture d’un entretien et/ou d’une réparation disponible une fois que le mécanicien a terminé les différentes tâches demandées. On peut voir que les heures prestées sur le véhicule sont également facturées. Cependant, le prix pour une heure prestée est fixé dans le code à 12,50€.

## Conclusions

### Les objectifs atteints

Mes objectifs principaux ont tous été atteints, mes fonctionnalités ont toutes été développées ce qui fait que mon application a pu être terminée. Mes deadlines ont également pu être respectées, cela m’aura permis d’avoir un peu de temps pour effectuer quelques tests sur l’application. Il reste cependant difficile de réaliser tous les tests possibles et imaginables, j’ai donc pu tester les différentes fonctionnalités et tout m’avait l’air fonctionnel. Le seul problème que j’ai pu détecter est situé dans la récupération du compte administrateur, des solutions seront proposées dans le futur de l’application, car il m’était difficile de réaliser ces fonctionnalités gratuitement et par manque de temps.

### Les modifications sur mes fonctionnalités

Afin de me consacrer aux fonctionnalités les plus importantes, j’ai décidé de retirer la fonctionnalité de la consultation des statistiques, je pense que c’était juste une partie supplémentaire inutile, étant donné qu’il y aura quand même un bilan effectué par un comptable en fin d’année comptable.

La consultation des factures est maintenant également disponible au vendeur, il n’y avait aucun intérêt à leur bloquer l’accès à une nouvelle impression de document(s).

3 nouvelles fonctionnalités ont été rajoutées afin de permettre une création, une modification et une suppression de pack d’option.

Une distinction entre les factures de vente et les bons de commande est maintenant faite directement dans les fonctionnalités.

Et les dernières modifications consistent à rendre un peu plus compréhensibles les fonctionnalités déjà indiquées.

### Futur de l’application

Le projet étant fictif, ce projet aura seulement pour but de réussir ma formation, mais différentes améliorations sont envisageables :

1. Un serveur mail afin de récupérer un mot de passe si un des employés ou l’administrateur le perd
2. Une création d’email professionnel avec le domaine « @BMWEcaussinnes.com » qui sera encodé pour l’email des utilisateurs.
3. Une amélioration des factures en fonction d’une réelle demande d’une entreprise en ayant un exemple de leurs factures afin d’être plus précis sur les informations.

### Ce que m’a apporté ce travail de fin d’études

Lors de ce travail de fin d’études, j’ai pu mettre à l’épreuve les différentes matières apprises lors de mon cursus scolaire. Il m’aura permis de voir les domaines où j’ai le plus de lacunes et celles où j’ai un peu plus d’aisance.

Il m’aura également mis à l’épreuve sur la rédaction de ce document. J’ai dû effectuer beaucoup de recherches afin d’être certain des différents éléments importants que ce dossier allait devoir contenir. De ce fait, j’ai emmagasiné de nouvelles matières d’analyse et ça m’a également permis d’approfondir certaines matières vues lors de mes différents cours. Lors de la rédaction du document et du code, j’ai pu me rendre compte que certaines compétences étaient importantes et que prendre un temps de réflexion plus conséquent, me permettait de ne pas revenir sur certains éléments que l’on pensait fonctionnels.

Je suis globalement satisfait des différentes réalisations que j’ai pu réaliser pendant ce travail.

Une application n’est jamais réellement terminée, il y aura toujours de nouvelles fonctionnalités à développer et des bugs à corriger, mais je pense avoir fait un bon travail sans trop de bugs avec un panel de fonctionnalités permettant une utilisation intéressante de mon application.

## Annexes

### Conseils d’utilisations

Afin de vous guider lors des différents tests de mon application, je vais vous indiquer en forme d’étapes le chemin à suivre lors des différents ajouts. Ce chemin a uniquement pour but de faciliter l’utilisation de l’application, tout autre chemin reste envisageable. Le chemin conseillé est le chemin présenté lors des maquettes, mais afin d’être le plus précis possible, je vous propose un petit tutoriel écrit sur les ajouts et la suite logique de ceux-ci.

Une fois votre compte créé, le premier écran qui vous sera proposé, est l’écran des rendez-vous. La première étape sera de créer un client. Une fois le client ajouté, vous pourrez lui ajouter un rendez-vous.

Ensuite, je vous conseille d’ajouter des options et des packs pour les lier au véhicule que vous allez ajouter par la suite. Une fois les options et packs ajoutés, vous pouvez vous diriger vers l’écran des nouveaux véhicules (voitures – motos).

Une fois sur une de ces pages, vous aurez la possibilité d’ajouter votre véhicule. Vous pourrez également y ajouter les différentes caractéristiques. Pour l’image, je vous invite à vous rendre dans le dossier que j’aurais prévu à cet effet ou de vous rendre sur le site officiel de BMW (<https://www.bmw.be/fr/all-models.html>) afin de choisir le véhicule qui vous plait. Quand vos caractéristiques seront ajoutées, vous pourrez les lier au véhicule qui est en cours d’ajout. Sur cette page, vous pourrez également lier des packs et des options qui seront spécifiquement liés à ce véhicule.

Une fois qu’un véhicule a été ajouté, deux pages s’ouvrent à vous. Pour commencer, je vous conseille d’aller sur la page des disponibilités. Sur cette page, vous allez pouvoir y ajouter les véhicules appartenant à la concession. Vous allez maintenant pouvoir indiquer les différents packs et options de ce véhicule. Une fois la personnalisation effectuée, vous allez devoir indiquer les différentes informations demandées pour ensuite valider l’encodage du véhicule. L’autre page qui vous est maintenant accessible est la personnalisation d’un véhicule de client. Pour y accéder, vous allez devoir vous rendre sur la page des catalogues afin d’y retrouver la personnalisation d’un véhicule. Une fois sur cette page, vous allez pouvoir affecter au véhicule les différentes options et packs que le client a choisis. Une fois la personnalisation effectuée, vous devez choisir ou ajouter un client.

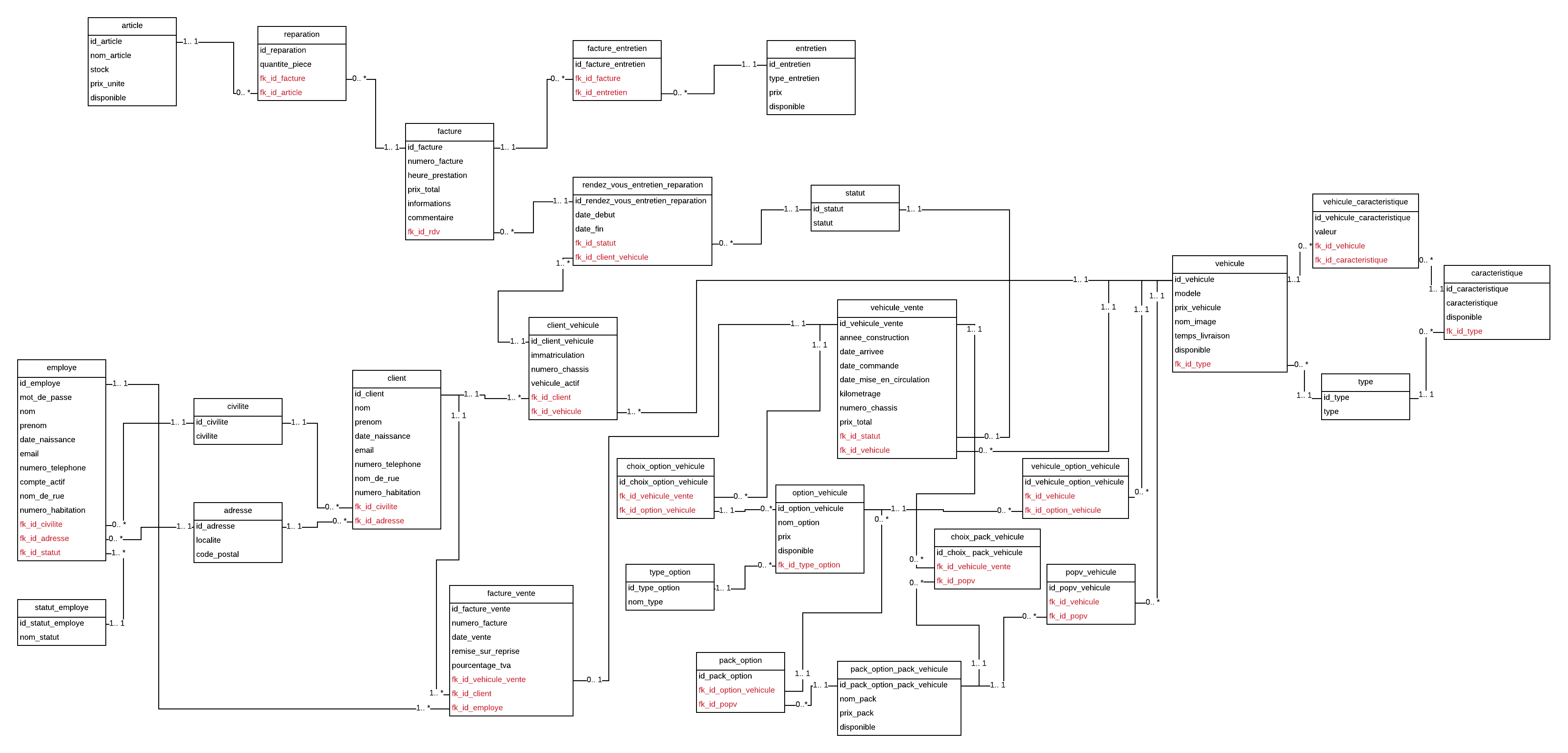
Une fois le véhicule commandé, ce qui vous restera à faire une fois le véhicule arrivé, c’est de terminer la vente. Pour ce faire, vous vous rendez sur la page des commandes. Afin de valider la vente, vous allez devoir y encoder les différentes valeurs demandées. Une fois les valeurs encodées, vous allez pouvoir cliquer sur valider afin de sortir le document PDF de la vente.

Ensuite, nous allons revenir sur la suite logique des véhicules de la concession. Maintenant que nous avons encodé un véhicule, nous allons pouvoir accéder à la page des véhicules disponibles. Vous allez pouvoir choisir un des véhicules proposés sur le côté. Une fois le véhicule choisi, vous n’aurez plus qu’à désigner un client ou en encoder un nouveau pour ensuite valider la vente. Attention à ne pas oublier de modifier la TVA si vous indiquez qu’une reprise de véhicule est effectuée.

Avant d’accéder à la page spécifique des mécaniciens, je vous conseille d’aller sur la page des stocks afin d’y encoder les différentes marchandises que le mécanicien va utiliser. Ensuite, vous allez devoir créer le compte du mécanicien sur la page des employés en lui indiquant son statut de mécanicien.

Une fois ce compte créé, vous allez avoir accès à la page des mécaniciens. Sur cette page, il aura la possibilité de choisir un rendez-vous indiqué (et pris au préalable par un vendeur) afin d’y afficher les informations dont il a besoin. Il devra indiquer les différentes informations présentes sur l’écran afin de valider la vente. Attention, une fois le rendez-vous validé, il n’y a plus de retour en arrière possible.

### Diagramme de classes complet



### Bibliographie

<https://support.microsoft.com/en-us/help/319266/how-to-programmatically-add-controls-to-windows-forms-at-run-time-by-u>

<https://www.nuget.org/packages/itext7/>

<https://stackoverflow.com/questions/2069048/setting-the-filter-to-an-openfiledialog-to-allow-the-typical-image-formats>

<https://stackoverflow.com/questions/12491597/limit-image-size>

<https://docs.microsoft.com/en-us/dotnet/api/system.io.file.copy?view=netframework-4.8>

<https://codes-sources.commentcamarche.net/source/28311-tous-les-codes-postaux-et-villes-de-belgique>

<https://ineumann.developpez.com/tutoriels/merise/initiation-merise/>

<https://balsamiq.com/wireframes/desktop/>

<https://www.lucidchart.com/pages/>

<https://marketplace.visualstudio.com/items?itemName=VisualStudioClient.MicrosoftVisualStudio2015InstallerProjects>

<https://www.bmw.be/fr/all-models.html>