

DOSSIER PROJET

HOFFMANN MICHEL



DEVELOPPEUR WEB FULL STACK

Michel Hoffmann

Développeur Web Full Stack

En Développement web, comme en tout, les connaissances ne sont rien, sans curiosité, intelligence, réflexion et imagination.

C'est la curiosité et la passion pour ce métier qui m'ont amené à une reconversion professionnelle et à reprendre des études.

De nature autodidacte et toujours motivé, je mettrai à profit mes expériences pour répondre au mieux à l'exigence du client. Je suis flexible quant à mes déplacements pour satisfaire aux exigences de l'emploi.

Je possède naturellement le sens du contact, je suis à l'aise pour travailler en équipe ou en autonomie et prêt à relever de nouveaux challenges.



EMPLOI

2006 - 2022

STELLANTIS Conducteur D'installation

-Gestion d'un parc matière.
-Respect de la réglementation en matière de sécurité, des exigences de contrat, des politiques d'entreprise et des procédures opérationnelles.

2008 - 2016

BIGBANG Dj / Animateur / Gérant

-Organisation et animation d'évènements.
-relations clients

2003 - 2008

TENDANCE Responsable de salle

-Satisfaction des clients grâce à une mise en place immédiate afin de prendre rapidement les commandes de boissons et de répondre à tous leurs besoins.
-Responsabilités complètes d'ouverture, de fermeture et de changement d'équipe pour favoriser l'efficacité du restaurant et des équipes afin de répondre aux besoins des clients.
-Applications des procédures adéquates de nettoyage, d'hygiène et de manipulation des aliments dans le respect des réglementations du ministère de la Santé.
-Présentation au moment opportun auprès des clients pour prendre les commandes et confirmer leur satisfaction par rapport aux repas après le service, en veillant à rectifier tout problème éventuel.

2002 - 2004

SHINE Responsable événementiel

-Gestion d'une équipe, planning de 12 personnes.
-Organisation d'évènements.
-Supervision de l'embauche, de la formation et de l'évolution professionnelle de chaque employé.
-Relations clients

1999 - 2002

PINO Chef de rang et maître d'hôtel

-Gestion d'une vingtaine de salariés, supervision de l'embauche, de la formation et de l'évolution professionnelle de chaque salarié.
-A épauler le directeur d'exploitation dans le cadre des décisions opérationnelles quotidiennes.

2022 - 2023

WILD CODE SCHOOL

Diplôme de niveau bac +2
Développeur Web Full Stack

ÉDUCATION



1996 - 1999

SEVIGNE

Cap, Bep et Bac Pro
Restauration + Bepc

michel.hof@hotmail.fr

06 66 33 25 88

www.cvmichel-hoffmann.fr



COMPÉTENCES

Javascript / React

PHP / Symfony

Base De Données MySql

Git Et GitHub

Marketing Digital

Référencement Seo

Réalisation Maquettes et Logos

HTML5 Et CSS3

Bootstrap

LANGUES

ANGLAIS ET ALLEMAND

Lire et écrire (scolaire)

ACTIVITÉS

PERSONNELLES

ECOLOGIE

Militant, respect de la nature

ENTRAIDE

Aux plus démunis

RANDONNEES

Marche + Vélo

MUSIQUE

Ecoute + Concert

SURF

Océan

Sommaire

Exemples de pratique professionnelle

1/Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)	p. 5
▶ Intitulé de l'exemple n° 1 PREPARATION DU PROJET (cahier des charges, maquette...) ...p.	p. 8
▶ Intitulé de l'exemple n° 2 CONCEPTION DE LA BASE DE DONNEESp.	p. 11
▶ Intitulé de l'exemple n° 3 CONNEXION / DECONNEXION ET MOT DE PASSEp	p. 19
2/Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)	p.
▶ Intitulé de l'exemple n° 1 ESPACE ADMINISTRATEUR CRUD VEHICULESp.	p. 25
▶ Intitulé de l'exemple n° 2 API VEHICULE AVEC INTEGRATION DES FILTRESp.	p. 33
▶ Intitulé de l'exemple n°p	p.
3/Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)	p.
▶ Intitulé de l'exemple n° 1 CREATION DES COMPOSANTS CARD VEHICULEp.	p. 40
▶ Intitulé de l'exemple n° 2 DEVELOPPEMENT DES COMPOSANTS FILTRESp.	p. 46
▶ Intitulé de l'exemple n° 3 CONCEPTION DE LA PAGE DE RECHERCHE PAR FILTRESp	p. 53
4/Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)	p.
▶ Intitulé de l'exemple n° 1 AVIS CLIENTSp.	p. 58
▶ Intitulé de l'exemple n° 2 FORMULAIRE DE CONTACTp.	p. 62
▶ Intitulé de l'exemple n° 3p	p.
Titres, diplômes, CQP, attestations de formation (facultatif)	p. 66
66Déclaration sur l'honneur	p. 67
Documents illustrant la pratique professionnelle (facultatif)	p. 68
Annexes (Si le RC le prévoit)	p.



DOSSIER PROFESSIONNEL (DP)

Nom de naissance

▶ HOFFMANN

Nom d'usage

▶ Entrez votre nom d'usage ici.

Prénom

▶ MICHEL

Adresse

▶ 4 LOTISSEMENT LA CROIX ROUGE 03 230 GARNAT SUR
ENGIEVRE

Titre professionnel visé

DEVELOPPEUR WEB FULL STACK

MODALITE D'ACCES :

- Parcours de formation
- Validation des Acquis de l'Expérience (VAE)

Présentation du dossier

Le dossier professionnel (DP) constitue un élément du système de validation du titre professionnel.
Ce titre est délivré par le Ministère chargé de l'emploi.

Le DP appartient au candidat. Il le conserve, l'actualise durant son parcours et le présente **obligatoirement à chaque session d'examen.**

Pour rédiger le DP, le candidat peut être aidé par un formateur ou par un accompagnateur VAE.

Il est consulté par le jury au moment de la session d'examen.

Pour prendre sa décision, le jury dispose :

1. des résultats de la mise en situation professionnelle complétés, éventuellement, du questionnaire professionnel ou de l'entretien professionnel ou de l'entretien technique ou du questionnement à partir de productions.
2. du **Dossier Professionnel (DP)** dans lequel le candidat a consigné les preuves de sa pratique professionnelle.
3. des résultats des évaluations passées en cours de formation lorsque le candidat évalué est issu d'un parcours de formation
4. de l'entretien final (dans le cadre de la session titre).

[Arrêté du 22 décembre 2015, relatif aux conditions de délivrance des titres professionnels du ministère chargé de l'Emploi]

Ce dossier comporte :

- ▶ pour chaque activité-type du titre visé, un à trois exemples de pratique professionnelle ;
- ▶ un tableau à renseigner si le candidat souhaite porter à la connaissance du jury la détention d'un titre, d'un diplôme, d'un certificat de qualification professionnelle (CQP) ou des attestations de formation ;
- ▶ une déclaration sur l'honneur à compléter et à signer ;
- ▶ des documents illustrant la pratique professionnelle du candidat (facultatif)
- ▶ des annexes, si nécessaire.

Pour compléter ce dossier, le candidat dispose d'un site web en accès libre sur le site.

 <http://travail-emploi.gouv.fr/titres-professionnels>



Lien TRELLO : <https://trello.com/b/jdyXm3Ws/conduite-de-projet>

Lien github côté Front : <https://github.com/Michelhof1978/GarageParrot>

Lien github côté back : <https://github.com/Michelhof1978/GarageBack>

Lien Figma : <https://www.figma.com/file/rhs91pTc1st89ApCbMHoel/GARAGE-PARROT?type=design&node-id=0-1&mode=design&t=b8DACnrXU32mmG6U-0>

J'ai élaboré Le projet du Garage Parrot, il s'agit d'un client fictif.

Le projet vide à développer une application web vitrine pour le **Garage V.Parrot**, un établissement automobile situé à Toulouse, qui propose une gamme de services incluant la réparation automobile, l'entretien et la vente de véhicules d'occasion.

L'objectif principal de cette application est de mettre en avant la qualité des services proposés par le garage.

Ma situation personnelle ne me permettait pas de réaliser un stage en entreprise donc, j'ai réalisé ce projet seul de A à Z.

J'ai décidé de faire 2 dossiers Github séparés un côté Back et de l'autre en Front de peur à me perdre lorsque plus tard, que je sois envahi par tout ce code et plus m'y retrouver.

DOSSIER PROFESSIONNEL (DP)

Je vous propose aussi 2 autres projets que j'ai réalisé, 1 pour un client (Site offert pour un ami, ce qui m'a permis de m'entraîner en temps réel) et l'autre, il s'agit de mon Cv en ligne.



Projet Client Les Caravanes De La Besbre : <https://lescaravanesdelabesbre.fr/>

Lien github : <https://github.com/Michelhof1978/lesCaravanesDeLaBesbre>

Technologies Utilisées : Html, Css, Bootstrap, javascript et php sans base de données pour l'instant.

Sur ce projet, il y a plus de front que du back, ici le formulaire de contact fonctionne, le site est en ligne, j'ai fait le référencement naturel (Le site a eu 3000 visites en 4 mois, ce fût un succès, intégration d'une API météo)

Ce projet est évolutif, les prochaines étapes seront un système de réservation en ligne, un espace administrateur et un système de paiement et bien sûr avec une base de données.



Projet Cv en ligne : <https://cvmichel-hoffmann.fr/>

Lien github : <https://github.com/Michelhof1978/cvMichel>

Technologies Utilisées : Html, Css, Bootstrap, javascript et php sans base de données.

BACK => Dans le cadre de la mise en place de la partie Back-end de ce projet, j'ai choisi d'utiliser des technologies telles que PHP, MySQL, ainsi que les environnements de développement Wamp.

Une décision importante a été de ne pas recourir à Symfony pour ce projet, préférant opter pour une approche de développement en PHP pur.

On m'a fréquemment conseillé, en tant que débutant, de maîtriser les bases de la programmation en travaillant directement avec du code non abstrait.

Cela m'a aidé à mieux apprêhender la logique sous-jacente du code. Toutefois, l'absence d'un framework pour accélérer le développement a entraîné une augmentation du temps nécessaire à la réalisation du projet, qui reste partiellement achevé à ce stade

FRONT => React, Javascript et Bootstrap, Html et Css .

Le projet ‘Garage Parrot’ a été réalisé seul à 100 % , de la maquette jusqu’à la réalisation du site.

A ce stade, il y a 4 mois de travail avec énormément de problèmes, je me suis bien sentit bien souvent seul et j'aurais bien voulu être accompagné de temps en temps.

J'ai pu transformer ce négatif en positif au fil du temps car tous ces problèmes m'ont permis d'avancer et de comprendre certaines choses.

DOSSIER PROFESSIONNEL (DP)

Activité-type 1

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)

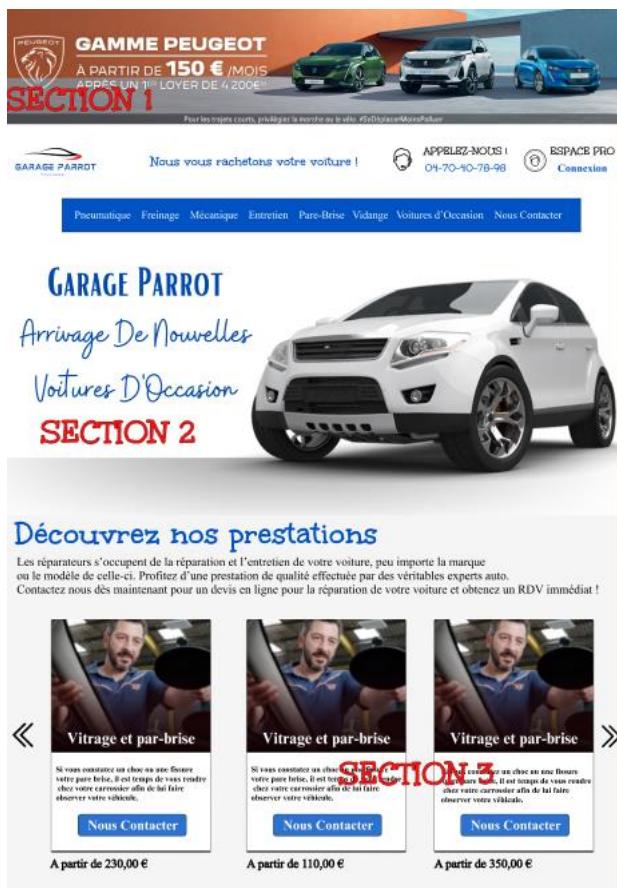
Exemple n°1 ▶ PREPARATION DU PROJET (cahier des charges, maquette...)

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

- J'ai identifié les besoins du client en réalisant un cahier des charges (joint en annexe).

- J'ai réalisé les **Users Stories** pour comprendre les besoins de l'administrateur et de l'utilisateurs et mettre l'accent de ce qu'ils souhaitent accomplir (joint en annexe).

- J'ai réalisé la **maquette** en sachant bien qu'il faudra présenter au client une partie en mode mobile et l'autre en mode desktop (joint en annexe ou directement sur le lien ci-dessus).



- Les **diagrammes de classe, de séquence et de cas d'utilisation** que j'expliquerai dans la section suivante sur le chapitre de la base de données ci-dessous.

J'ai aussi réalisé le logo de l'entreprise et quelques illustrations pour le site.



1er distributeur automobile



Véhicules certifiés et garantis jusqu'à 24 mois



Satisfait ou remboursé 14 jours ou 1 000km



Service client du lundi au samedi de 9h00 à 19h00



2. Précisez les moyens utilisés :

Logiciels :

- **Word**, traitement de texte pour cahier des charges, Users stories....
- **Figma** pour la réalisation de la maquette
- **Lucid** pour les diagrammes de la base de données (Outil de modalisation)
- **Trello** pour l'organisation des tâches
- **Canva** pour la réalisation du logo et quelques illustrations

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

DOSSIER PROFESSIONNEL (DP)

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (facultatif)

Activité-type 1

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)

Exemple n°2 ► CONCEPTION DE LA BASE DE DONNEES

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

J'ai entamé la création de trois diagrammes distincts, des outils de modélisation visant à clarifier les besoins du site pour le client en ce qui concerne son fonctionnement. (voir annexe).

Au fil de la conception du projet, j'ai apporté de nombreuses modifications, car il est devenu évident que certaines parties n'étaient pas logiques, et j'ai commis plusieurs erreurs.

Il est vrai que concevoir l'ensemble dès le départ s'est avéré être une tâche assez complexe en ce qui me concerne.

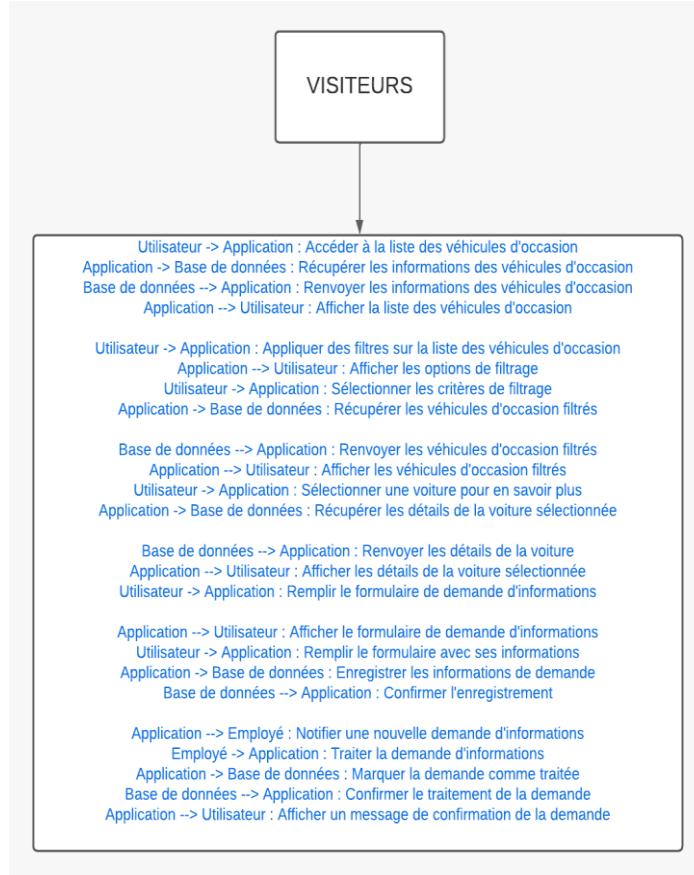
1.0 Diagrammes de Séquence

Les diagrammes de séquence me permettent de visualiser comment les objets interagissent dans le temps pour accomplir une action ou une série d'actions.

Ils mettent en évidence la séquence d'appels de méthodes ou de messages entre les objets d'un système.

Cela m'aide à comprendre comment les différentes parties de mon projet collaborent et se coordonnent pour atteindre un objectif spécifique.

1.1 Diagramme de Séquence Utilisateurs :

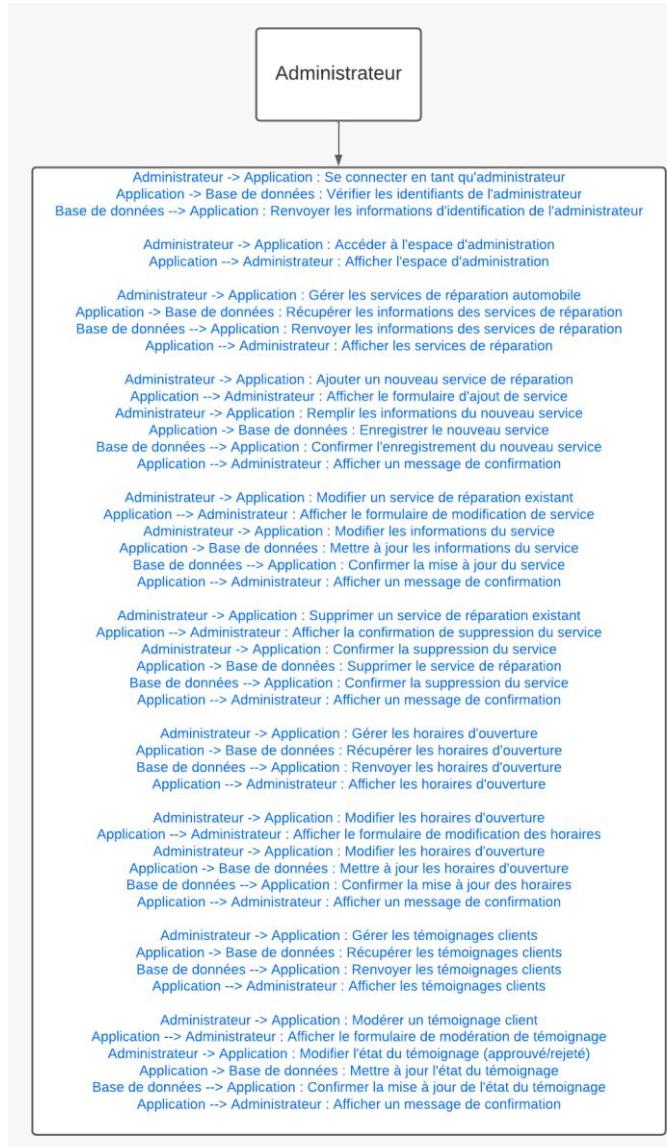


DOSSIER PROFESSIONNEL (DP)

Dans ce diagramme de séquence, je décris comment un nouveau visiteur peut découvrir la liste des véhicules d'occasion, appliquer des filtres pour affiner les résultats, sélectionner une voiture spécifique, et remplir le formulaire de demande d'informations pour en savoir plus sur cette voiture.

Mon application enregistre ensuite la demande et la notifie à un employé du garage, qui la traitera et la marquera comme traitée dans la base de données.

1.2 Diagramme de Séquence Administrateur :



Dans ce diagramme de séquence, je décris comment je peux me connecter à l'application, accéder à l'espace d'administration, gérer les services de réparation automobile (ajouter, modifier, supprimer), gérer les horaires d'ouverture, et modérer les témoignages clients.

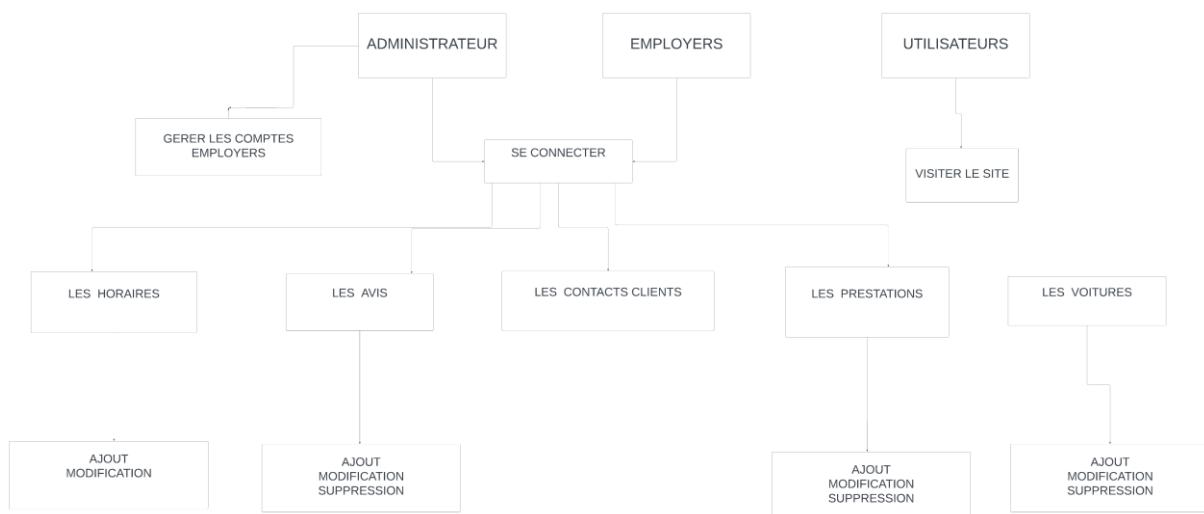
Mon application interagit avec la base de données pour récupérer et mettre à jour les informations pertinentes, et elle affiche des messages de confirmation pour m'indiquer le succès des opérations effectuées.

2.0 Diagrammes de Cas D'Utilisation

Dans le diagramme, je décris les interactions entre un système et ses utilisateurs (acteurs) en mettant l'accent sur ce que le système fait plutôt que sur comment il le fait.

J'ai mis en évidence les fonctionnalités ou les services offerts par le système du point de vue des utilisateurs.

Cela m'a aidé à mieux comprendre les besoins des utilisateurs et à définir les exigences fonctionnelles du système.



Le projet contiendra 2 parties distinctes :

-La partie back end qui sera réalisé en **Php** et placé sur un serveur **Apache** avec une base de données **mysql/apache**.

-Le panneau administrateur alimenteur et générera cette base de données en PHP.

Toute la partie Serveur utilisera l'architecture modèle du contrôleur et sera programmée en POO.

-L'utilisateur utilisera les données de **l'API REST PHP** qui devra être programmé au niveau serveur et qui seront en format **JSON**.

-Le serveur devra renvoyer uniquement les données en **Json** et ça sera **React** qui se chargera de les afficher.

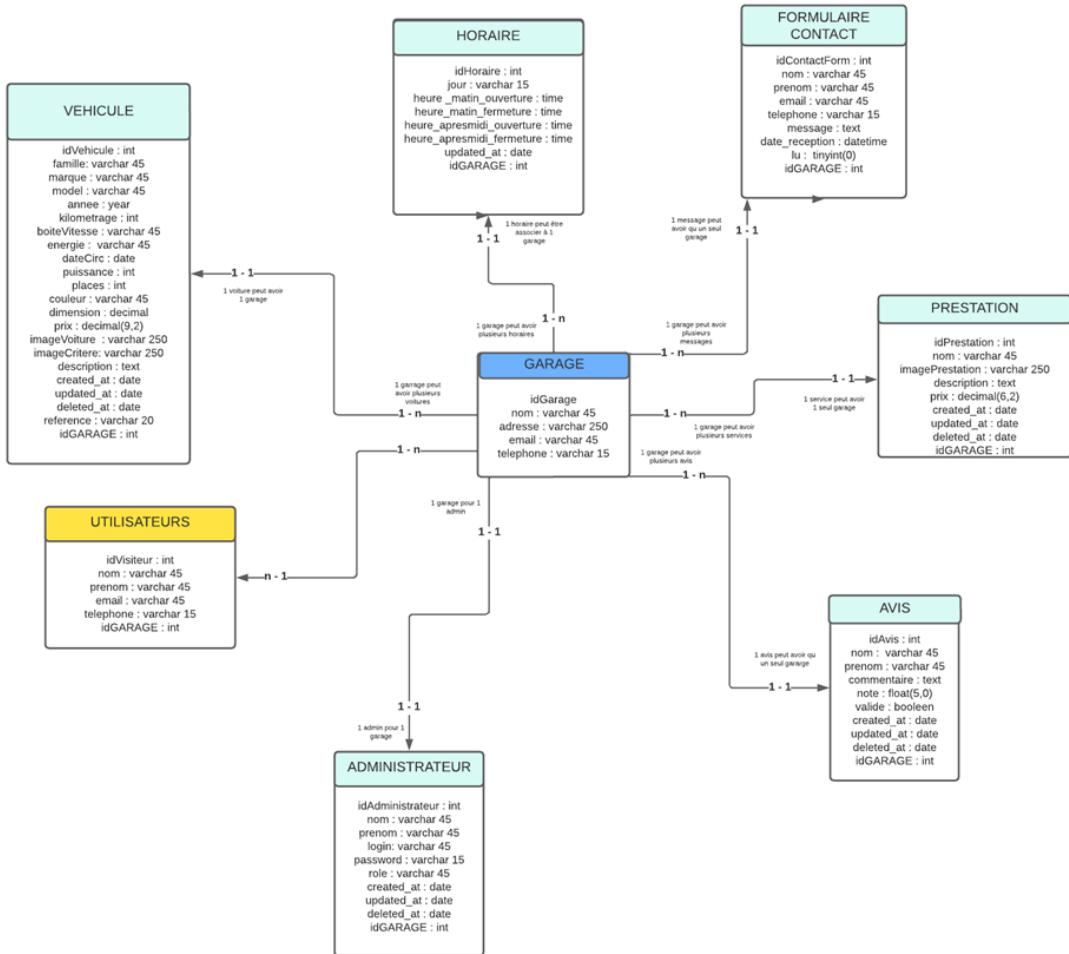
3.0 Diagrammes de Classe

Ce diagramme de classes illustre la structure de base du modèle de données pour le **Garage V. Parrot**.

J'ai créé ce diagramme de classe comme un moyen de représenter la structure statique d'un système en mettant l'accent sur les classes, leurs attributs, leurs méthodes et les relations entre les classes. Ils me permettront de visualiser la structure du système plutôt que son comportement.

Ce diagramme va modéliser les données (tables et champs avec leurs attributs) et la conception de l'architecture logicielle, ce qui facilite la création d'une base solide pour le développement de mon projet.

DOSSIER PROFESSIONNEL (DP)



Dans ce diagramme de classes que je viens de réaliser, le Garage est la classe principale et je lui ai attribué des méthodes pour gérer les **services**, les **horaires**, les **véhicules d'occasion**, les **contacts** avec le service client ainsi que les **avis clients**.

-**La classe ‘Service’** représente un service de réparation automobile, et j'ai défini des méthodes pour y accéder et modifier ses attributs.

-**La classe ‘Véhicule’** représente un véhicule d'occasion, et j'ai également ajouté des méthodes pour accéder et modifier ses attributs.

-**La classe ‘Message’** est utilisée pour stocker les informations de contact, telles que les demandes de contact de clients.

-**La classe ‘Horaire’** est dédiée à la gestion des horaires d'ouverture du garage, et elle peut avoir des méthodes pour accéder et modifier ces attributs.

-**La classe ‘Avis’** est conçue pour recueillir les avis ou les témoignages des clients concernant le garage ou ses services. J'y ai également inclus des méthodes pour accéder et modifier les attributs des avis, ainsi que pour gérer les opérations liées à ces avis.

4.0 Création des tables de la base de données avec MySqlWorkBench

Pour commencer, je vais configurer la base de données et vous citer tous les défis que j'ai pu rencontrer tout au long du processus.

L'objectif principal était de concevoir une base de données fonctionnelle et efficace pour pouvoir stocker les données de mon application.

J'ai plutôt décidé de faire directement le schéma et ensuite l'exporter vers la Bdd au lieu des commandes Sql.

Importation des Tables depuis MySQL Workbench

Au départ, j'ai utilisé MySQL Workbench pour concevoir le modèle conceptuel de la base de données, y compris les relations entre les tables.

Cependant, lors de l'exportation vers MySQL PHPMyAdmin, j'ai rencontré des problèmes liés aux cardinalités entre les tables.

Malgré mes efforts pour configurer les relations correctement dans MySQL Workbench, les contraintes de clé étrangère n'ont pas été prises en compte lors de l'exportation.

Table	Action	Lignes	Type	Interclassement	Taille	Perte
administrateur	Parcourir Structure Rechercher Insérer Vider Supprimer	1	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
avis	Parcourir Structure Rechercher Insérer Vider Supprimer	8	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
contactform	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
garage	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
horaires	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
prestation	Parcourir Structure Rechercher Insérer Vider Supprimer	9	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
utilisateurs	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
vehicule	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-

Ajout Manuel des Contraintes de Clé Étrangère

Pour résoudre ce problème de cardinalités, j'ai dû ajouter manuellement les contraintes de clé étrangère pour chaque table à l'aide de commandes SQL. Cela a permis d'établir les relations souhaitées entre les tables, enfin, c'est que je croyais au départ.

Voici un exemple de commande que j'ai utilisée pour ajouter une contrainte de clé étrangère :

```
1. ALTER TABLE table_enfant
2. ADD CONSTRAINT fk_nom_contrainte
3. FOREIGN KEY (colonne_etrangere) REFERENCES table_parente(colonne_primaire);
```

Cette solution a normalement permis l'intégrité des données dans ma base de données.

DOSSIER PROFESSIONNEL (DP)

Suite à la résolution initiale de mon problème, j'ai vérifié le modèle conceptuel dans PHPMyAdmin et j'ai constaté que les cardinalités n'avaient toujours pas été prises en compte.

Face à cette situation, j'ai dû prendre une décision pour avancer dans le projet.

J'ai choisi de passer par le panneau de commande SQL pour ajouter manuellement les contraintes de clé étrangère, table par table.

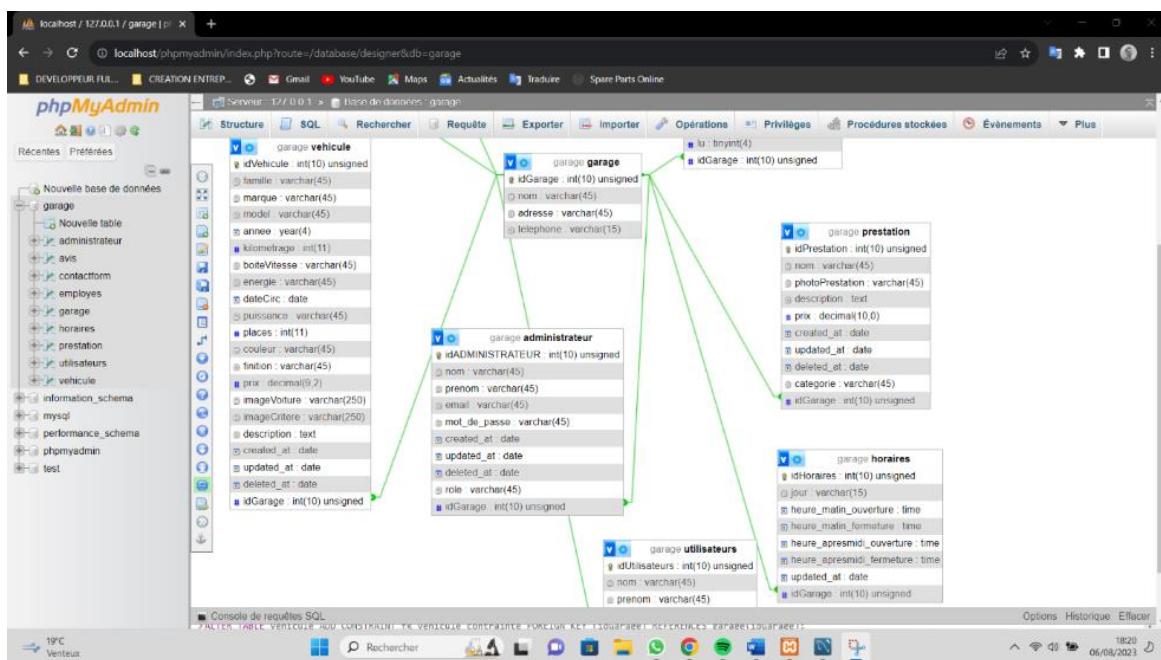
Bien que cette solution ait permis de créer les relations entre les tables conformément à mes besoins, je reste conscient que cela peut être considéré comme une approche moins automatisée et plus sujette aux erreurs.

Cependant, dans le contexte de mon projet et compte tenu du temps limité, c'était la solution la plus pratique pour pouvoir avancer.

Pour l'avenir, je vais continuer à travailler sur l'optimisation de notre base de données et des contraintes de clé étrangère, et je réévaluerai la possibilité de les réactiver une fois que je serai certain que toutes les données sont conformes aux contraintes.

Cette approche garantira l'intégrité des données à long terme tout en me permettant de poursuivre le développement sans interruption.

En fin de compte, cette expérience m'a montré l'importance de la flexibilité et de l'adaptabilité dans le processus de développement, ainsi que la nécessité de prendre des décisions pragmatiques pour faire progresser un projet, même en cas de difficultés imprévues.



Création de Données Temporaires dans PHPMyAdmin

Après avoir résolu les problèmes liés aux contraintes de clé étrangère, j'ai entrepris de saisir de fausses données temporaires dans les tables de ma base de données.

L'objectif était de créer un environnement de test pour vérifier le fonctionnement de mon application.

Cependant, j'ai de nouveau rencontré des erreurs liées aux clés primaires lors de l'insertion de ces données. Après une analyse plus approfondie, j'ai pu identifier la source du problème.

J'avais mal interprété les cardinalités des relations entre les tables.

J'avais créé manuellement les clés étrangères alors que si j'avais choisi les bonnes cardinalités, les clés auraient été créées automatiquement dans chaque table.

Une fois que j'ai ajusté les cardinalités correctement, j'ai pu exporter les données sans problème.

Ajout de Données dans la Table Horaire

Lorsque j'ai commencé à ajouter des données dans la table "horaire", j'ai réalisé que j'avais initialement considéré uniquement les jours d'ouverture, négligeant les jours de fermeture.

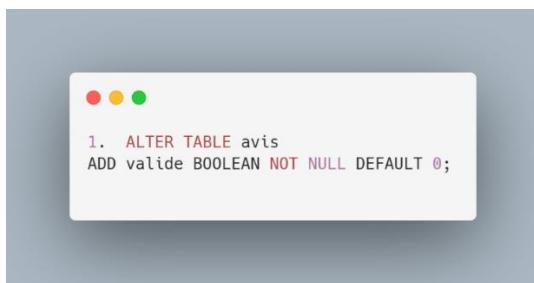
J'ai rapidement rectifié cette omission en ajoutant une colonne supplémentaire appelée "ferme" à la table.

Cette colonne prendra la valeur 0 si l'établissement est ouvert et 1 si c'est fermé, me permettant ainsi de gérer correctement les jours de fermeture.

Création de la table avis

Je vais initialiser dans la table 'avis' un état **valide(1) ou non valide(0)** pour l'espace administrateur sous forme de **booléen**.

Je vais l'initialiser au départ dans la base de données à 0 comme non valide avec une commande Sql



Chaque **Avis** sera donc initialisé à **0** automatiquement comme non valide car après, l'administrateur devra le valider ou pas dans son espace admin avant la mise en ligne.

J'ai enfin réussi à relier par la suite les champs à la table **Garage** en utilisant des clés étrangères. Initialement, j'ai rencontré un problème où il ne semblait pas être capable de localiser la table "**Garage**" avec son ID.

Après une analyse plus approfondie, j'ai découvert que le problème résidait dans les cardinalités que j'avais choisies. Une fois que j'ai ajusté les cardinalités appropriées, le problème de la jointure a été résolu.

2. Précisez les moyens utilisés :

-**Lucid** pour les diagrammes de la base de données (Outil de modalisation)

-**MySQLWorkBench**

-**PhpMyAdmin** avec **MySQL** intégrés tous les 2 dans **Wamp**

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

DOSSIER PROFESSIONNEL (DP)

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (*facultatif*)

Activité-type 1

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)

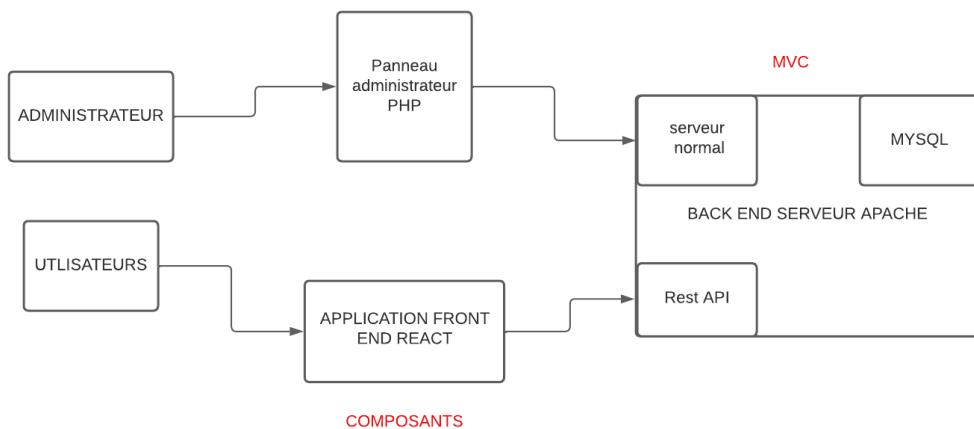
Exemple n°3 ▶ CONNEXION / DECONNEXION ET MOT DE PASSE

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1.0 La Structure MVC (Modèle-Vue-Contrôleur)

J'ai adopté pour l'architecture MVC (Modèle-Vue-Contrôleur) pour organiser ainsi mon code de manière efficace et modulaire.

Cette structure est le moteur dans la séparation des responsabilités et la gestion de l'application.



Les utilisateurs n'ont pas besoin de se connecter, il y aura uniquement du côté administrateur.

2.0 Mise en Place du Système de Connexion pour l'Espace Professionnel

Une étape de mon travail qui a été la mise en place du système de connexion pour l'espace professionnel de l'application.

Voici les étapes une par une pour créer ce système :

2.1 Création des Champs de Connexion

Pour gagner un peu de temps, j'ai utilisé des champs Bootstrap préexistants pour créer les champs de connexion. Cela m'a permis de concevoir plus rapidement l'interface de connexion administrateur.

2.2 Utilisation de password_hash() pour la Sécurité des Mots de Passe

La sécurité sur internet est primordiale, je dois donc faire mon maximum pour que le site ne soit pas piratable. Je vais donc utiliser la fonction **password_hash()** pour hacher (crypter) les mots de passe des utilisateurs.

Cette fonction génère un hachage sécurisé qui peut être stocké en toute sécurité dans la base de données. J'ai opté pour l'option '**PASSWORD_DEFAULT**', considérée comme je pense d'après mes recherches, la plus sécurisée à ce jour.

J'aurais qu'à plus ajouter un mot de passe qui sera haché automatiquement que j'insérerais dans la Bdd par la suite.

```
○ ○ ○  
  
public function GetPageLogin() {  
    require_once(__ROOT__.\views\login_view.php');  
}  
  
public function connexion() {  
  
    echo password_hash("michelaquiche", PASSWORD_DEFAULT);  
//    echo "connexion";  
}
```

2.3 Modification des Tables "Employés" et "Admin"

En voulant supprimer certaines informations inutiles dans les tables "Employés" et "Admin", notamment l'e-mail que j'avais noté auparavant, je vais pour l'instant, conserver uniquement les champs "**Login**" et "**Password**".

A ce stade, le login sera généré manuellement, tandis que le mot de passe sera sécurisé par `password_hash()`.

J'ai pensé ensuite, qu'une seule table devrait suffire pour l'espace admin au lieu de créer 2 tables admin et employés, je vais plutôt attribuer des rôles à chacun des administrateurs que je ferai plus tard.

2.4 Tests du Système de Connexion

J'ai effectué des tests en appuyant sur le bouton "Valider". Le système a généré automatiquement un mot de passe sécurisé sur la page de Login, que j'ai pu vérifier en le comparant avec le mot de passe haché stocké dans la base de données.

À chaque régénération de la page, un nouveau mot de passe haché sera généré mais mon vrai mot de passe ne changera pas.

Ce mot de passe haché se générera à chaque fois que je cliquerai sur le bouton ce qui ajoutera une sécurité en plus. Dans ces conditions-là, il sera sans doute impossible de le pirater.

2.5 Insertion de Données Factices

Pour effectuer des tests plus poussés, j'ai inséré de fausses données dans la table, en copiant le mot de passe généré automatiquement et en l'insérant dans le champ "**password**" et admin pour "Login "de ma table "**Administrateur**".

La mise en place de ce système de connexion sera, je pense une étape qui garantira la sécurité des données sensibles des administrateurs et ainsi respecter les conditions de **RGPD** que l'on nous impose sous peine de grosses amandes par la suite.

A screenshot of the phpMyAdmin interface. The left sidebar shows the database structure with the 'garage' database selected. The main area displays the 'administrateur' table. The table has the following structure:

	idADMINISTREUR	nom	prenom	login	password	created_at
<input type="checkbox"/>	1	admin		\$2y\$10\$uXUsPCy7ZUNVDC0AV/Tui0.lPybAhglaiixJMcgtqj3...	NULL	2023-10-10 14:45:10

Below the table, there are buttons for 'Tout afficher' (Show all), 'Nombre de lignes' (Number of rows) set to 25, and 'Filtrer les lignes' (Filter rows). There is also a search bar labeled 'Chercher dans cette table' (Search in this table).

2.6 Validation des Tests et Conclusions

En conclusion de cette phase de développement, j'ai réussi à mettre en place un système de gestion de mot de passe sécurisé pour l'espace professionnel de l'application.

Le processus de validation a été concluant, confirmant l'efficacité du hachage de mon mot de passe.

Pour valider le système, j'ai utilisé la fonction `'password_hash()'` pour hacher un mot de passe spécifique, en l'occurrence "**"michelaquiche"**". Le test a été un succès, puisque le mot de passe généré automatiquement correspondait au hachage attendu.

2.7 Sécurisation et Vérification des Informations de Connexion



```
public function connexion(){
    if (!empty($_POST["login"]) && !empty($_POST["password"])) {
        $login = Securite::secureHtml($_POST["login"]); //securite en lien avec security.class.php
        $password = Securite::secureHtml($_POST["password"]);

        if($this->AdminManager->isConnexionValid($login, $password)) {
            $_SESSION['access'] = "admin"; // Pour activer les variables de session, il va falloir
que je les active en début de page ds index.php
            header('Location: '.URL."back/admin");
            exit();
        } else {
            header('Location: '.URL."back/login");
            exit();
        }
    }
}
```

La sécurité de l'application est très importante, notamment lorsqu'il s'agit de gérer les informations sensibles dont les mots de passe.

Voici les étapes que j'ai suivies pour renforcer la sécurité de la gestion des comptes professionnels :

Ajustement de la Longueur des Mots de Passe

Lors de l'insertion des données dans la table, j'ai rencontré une erreur liée à la longueur du mot de passe haché.

En effet, j'ai pensé à mettre assez de place pour un mot de passe ordinaire mais je n'avais pas pensé au cryptage qui est largement plus long.

Pour remédier à cela, j'ai ajusté la longueur du champ VARCHAR, en veillant à ce qu'il soit suffisamment long pour accueillir le hachage dans le champ.

J'ai également constaté que l'utilisation de caractères spéciaux dans le mot de passe pouvait entraîner des problèmes, j'ai donc préféré n'utiliser que des caractères alphanumériques pour éviter d'éventuels problèmes même si je sais que je ne devrais pas laisser en l'état.

Je m'en occuperais sans doute plus tard si le temps me le permet.

Mise en Place de Vérifications

J'ai créé une page dédiée à la sécurité où je vais vérifier notamment ce qui se passe au niveau des formulaire notamment il vérifiera si les utilisateurs ont bien accès au site.

J'ai également converti en HTML les caractères spéciaux pour éviter certains problèmes de sécurité.

J'ai aussi mis en place un système de vérification pour m'assurer que les champs de connexion sont correctement remplis.

J'ai rajouté une sécurité en plus où j'extrais les valeurs soumises pour les champs "login" et "password" que je vais faire passer par une fonction nommée **secureHtml**, elle effectuera des opérations de nettoyage pour éviter les attaques de sécurité, comme l'injection de code malveillant.

J'ai également prévu de créer un lien de déconnexion qui supprimera la variable de **session** lorsque l'utilisateur se déconnectera.

Je rajouterais une fonction dans le manager qui fera des actions de vérifications de connexion en renvoyant true ("Authentification réussi") ou false ("Authentification échouée").

Gestion des Sessions

Je vais créer dans le controller des variables de sessions et que si les informations de connexion sont correctes (vérifiées en utilisant des identifiants préalablement créés), une session est générée, et l'utilisateur est redirigé vers la page administrative, en n'oubliant pas de le déclarer en début de la page **index.php** pour que les pages puissent par la suite communiquer entre elles.



```
$_SESSION['access'] = "admin";
```

Validation des Informations de Connexion

J'ai développé une fonction de vérification des informations de connexion pour m'assurer que l'utilisateur a rempli correctement les champs requis.

Cela garantit également que lors de la déconnexion, l'accès à la page d'administration est désactivé donc quitté la Session.

DOSSIER PROFESSIONNEL (DP)

```
public function connexion(){
    if (!empty($_POST["login"]) && !empty($_POST["password"])) {
        $login = Securite::secureHtml($_POST["login"]); //securite en lien avec security.class.php
        $password = Securite::secureHtml($_POST["password"]);

        if($this->AdminManager->isConnexionValid($login, $password)) {
            $_SESSION['access'] = "admin"; // Pour activer les variables de session, il va falloir que je les active en début de page ds index.php
            header('Location: '.URL."back/admin");
            exit();
        } else {
            header('Location: '.URL."back/login");
            exit();
        }
    }
}
```

Les étapes que j'ai effectuées ont garanti que les informations de connexion soient bien sécurisées et que seuls les utilisateurs autorisés auront accès à la page d'administration

Garage Parrot - Espace Pro Connexion

Connexion Espace Pro

Identifiant

Mot De Passe

Valider

2. Précisez les moyens utilisés :

- Vscode
- Wamp
- MySql
- PhpMyAdmin

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (*facultatif*)

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)

Exemple n° 1 ▶ ESPACE ADMINISTRATEUR CRUD VEHICULES

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Dans cette section, je vais élaborer la **vue** de l'espace administrateur sous forme de tableau Bootstrap en offrant une interface utilisateur plus conviviale et je vais expliquer comment l'administrateur peut gérer son inventaire de véhicules en utilisant les opérations **CRUD** (Create, Read, Update, Delete).

Je vais également mettre en œuvre le modèle **MVC** (Modèle-Vue-Contrôleur) et introduire des données fictives en utilisant les commandes **SQL**, ce qui sera particulièrement utile pour effectuer des tests ultérieurement dans la base de données.

Ces données devront être converties au format **Json**. Je vais donc devoir créer une API véhicules que j'expliquerai par la suite.

1.0 Mise en place du Modèle

Pour pouvoir se connecter à la Bdd en utilisant **PDO**, une extension PHP et qui fournit une interface uniforme pour interagir avec différentes bases de données relationnelles.

```
class Model {
    private static $pdo;

    private static function setBdd(){
        self::$pdo = new PDO("mysql:host=localhost;dbname=garage;charset=utf8","root","");
        self::$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
    }

    protected function getBdd(){
        if(self::$pdo === null){
            self::setBdd();
        }
        return self::$pdo;
    }

    public static function sendJSON($info){
        header("Access-Control-Allow-Origin: *"); /site sera en ligne
        header("Content-Type: application/json");
        echo json_encode($info);
    }
}
```

L'affichage des données dans l'espace professionnel va pouvoir fournir aux utilisateurs un accès rapide et efficace aux informations dont ils ont besoin.

Les prochaines étapes consisteront à mettre en place les fonctionnalités de modification et de suppression.

2.0 Implémentation du Bouton de Suppression

J'ai créé le bouton de suppression pour garantir la sécurité des données et éviter toute suppression accidentelle. Voici comment j'ai mis en place ce mécanisme dans l'espace professionnel :

2.1 Création d'une Nouvelle Route

Tout d'abord, j'ai créé une nouvelle route pour gérer l'action de suppression. Cette route permettra de transmettre l'identifiant du véhicule à supprimer depuis l'URL.

2.2 Conversion de l'ID en Entier

Étant donné que l'URL transmet l'identifiant en tant que chaîne de caractères, j'ai dû convertir cette chaîne en entier dans le contrôleur correspondant.

Cela a permis d'éviter les erreurs et les problèmes de sécurité potentiels.

2.3 Mise en Place d'Alertes JavaScript

J'ai préparé des alertes en JavaScript pour confirmer l'action de suppression.

Ces alertes sont conçues pour n'apparaître qu'une seule fois par page, et j'ai veillé à supprimer la variable de session associée une fois que le message a été affiché.

3.4 Gestion des Redirections

Cependant, j'ai rencontré un problème de redirection après la suppression.

Bien que le chemin de redirection soit correct, j'ai dû le commenter temporairement pour résoudre le problème.

La mise en place de ce mécanisme de suppression sécurisé que j'ai effectué servira à protéger les données de l'application et garantir que les opérations de suppression sont effectuées de manière intentionnelle.

4.0 Bouton modifier NON OPERATIONNEL

Dans la vue, je vais créer des champs de saisie qui vont s'afficher sous la ligne de l'objet à modifier pour chaque attribut de la table "véhicule", permettant ainsi la modification de tous les détails, à l'exception de l'identifiant (ID) bien sûr. Par la suite, un bouton "Valider" sera ajouté.

Ensuite, je vais m'occuper de la partie côté serveur lorsque l'utilisateur appuiera sur le bouton "Valider" par rapport aux nouvelles informations insérées.

5.0 Bouton création d'un véhicule

En ce qui concerne le view, et pour une meilleure expérience utilisateur, j'ai préparé des propositions par rapport aux caractéristiques de la voiture sous forme de dropdown sauf pour la famille des véhicules qui seront en checkboxs pour mettre un peu de piquant, je mettrai une règle pour que l'admin puisse uniquement cocher une seule case pour éviter certaines erreurs.

6.0 Difficultés rencontrées

À deux reprises, j'ai rencontré des problèmes d'accès au serveur SQL, où l'accès m'a été complètement refusé sans raison apparente, malgré les tests que j'avais effectués.

J'ai dû réinstaller XAMPP encore une fois, mais le problème est que j'ai perdu mes données de table, bien que j'aie sauvegardé le dossier SQL auparavant.

Il doit exister une solution pour restaurer ces données, donc je vais effectuer des recherches à ce sujet.

Par la suite, j'ai appris à les sauvegarder en exportant la Bdd et me le renvoi en **nonDuFichier.sql** mais je devrais par la suite le compresser en fichier ZIP pour pouvoir de nouveau l'importer.

J'ai aussi un problème mineur que je n'ai pas encore trouvé, il faut que je clique 2 fois sur les boutons modifier, supprimé et déconnexion pour que l'opération s'effectue, peut être un problème de rafraîchissement de page, à suivre... (**J'ai trouvé plus tard le problème, il s'agissait d'un inversement de codes qui n'étaient pas à la bonne place).**

DOSSIER PROFESSIONNEL (DP)



```
<div class="form-check">
  <input type="radio" id="utilitaire" name="famille" value="Utilitaire" class="form-check-input"> <label for="utilitaire" class="form-check-label">Utilitaire</label>
</div>
```

J'ai récemment créé une nouvelle route et ajouté un modèle Bootstrap à mon application, en incorporant des cellules de remplissage pour chaque caractéristique du véhicule.

J'ai choisi de ne pas insérer manuellement l'ID lors de l'ajout d'un véhicule, car j'ai configuré la base de données pour qu'elle utilise l'auto incrémentation, ce qui simplifiera le processus.

Pour gérer cette fonctionnalité, j'ai adapté mon gestionnaire (manager) en utilisant une approche similaire à celle que j'avais employé pour la modification.

Cette fois-ci, j'ai élaboré la commande **SQL INSERT INTO** pour l'ajout des données.

Pour obtenir l'**ID** généré par la base de données, j'ai fait appel à la fonction `lastInsertId()`, qui est une fonctionnalité de l'extension **PDO**.



```
return $this->getBdd()->lastInsertId();
```

Ensuite, j'ai pris soin de transmettre cet ID au contrôleur responsable de la récupération du nouvel ID.

Pour garantir une expérience utilisateur fluide, j'ai ajouté un message de confirmation à l'intention de l'administrateur, l'informant du nouvel ID du véhicule.



```
$_SESSION['alert'] = [
    "message" => "Le véhicule a bien été créé sous l'identifiant : " .
    $idVehicule,    "type" => "alert-success"
];
```



```
public function createVehicule($imageVoiture, $famille, $marque, $modele, $annee,
    $kilometrage, $boitevitesse, $energie, $datecirculation,
    $puissance, $places, $couleur, $description, $prix, $imageCritere, $created_at)
{
    $req = "INSERT INTO vehicule (imageVoiture, famille, marque, modele, annee, kilometrage,
        boitevitesse, energie, datecirculation, puissance, places, couleur, description, prix,
        imageCritere, created_at)
        VALUES (:imageVoiture, :famille, :marque, :modele, :annee, :kilometrage, :boitevitesse,
            :energie, :datecirculation, :puissance, :places, :couleur, :description, :prix, :imageCritere,
            :created_at)";
```

Cependant, j'ai rencontré un problème lors de la saisie des informations dans les champs de formulaire. Les données étaient bien enregistrées dans la base de données, mais au lieu de mes entrées, elles étaient enregistrées sous la forme de '**000000**'.

La source du problème résidait dans l'utilisation de la fonction `Securite::secureHTML`, que j'ai dû temporairement désactiver pour résoudre ce problème.

Bien que cette désactivation ait permis de corriger l'anomalie, je suis conscient que cela pourrait potentiellement compromettre la sécurité de mon code.

Je cherche actuellement une solution plus sécurisée pour résoudre ce problème tout en maintenant la protection de mon application."

Malheureusement, même après avoir retiré l'utilisation de cette sécurité, mon bouton de modification continue de ne pas fonctionner

Cela signifie que la désactivation de cette fonction n'est pas la cause du problème.

Comme la sécurité est importée pour mon application, j'envisage plus tard de trouver des solutions alternatives pour résoudre ce problème tout en préservant la sécurité de mon code.

Ce passage a été marqué par pas mal de défis inattendus et des problèmes techniques mais je pense avoir assez bien géré sur ce coup-là.

Ajouter un Vehicule

Image Voiture

Aucun fichier choisi

Famille

- Utilitaire
- Berline
- Familiale
- Citadine
- SUV

marque

citroen

Modele

6.0 Ajout d'images lors de la création

Dans la vue, je vais modifier le type de champ en "**file**" pour permettre le téléchargement d'image.
Ensuite, je vais créer un nouveau fichier de contrôleur que je nommerai "**regles_utiles.php**".

C'est dans ce fichier que je vais définir les règles de téléchargement, telles que la taille et le format des images qui devront être respectées, je fais cela pour éviter d'alourdir le site et ainsi le ralentir, ce qui pourrait faire fuir les utilisateurs si la page met trop longtemps à charger.



```
function ajoutImage($file, $dir){  
    if(!isset($file['name']) || empty($file['name']))//vérification si l image a bien été saisie  
        throw new Exception("Vous devez indiquer une image");  
  
    if(!file_exists($dir)) mkdir($dir,0777);//Si pas de répertoire de crée alors il va en créer un sur  
    le serveur  
  
    $extension = strtolower(pathinfo($file['name'],PATHINFO_EXTENSION));  
    $random = rand(0,99999); //on va générer un nombre aléatoire pour donner un nom unique au fichier.  
    $target_file = $dir.$random."_".$file['name'];  
  
    if(!getimagesize($file["tmp_name"]))//vérifier que le fichier est bien une image  
        throw new Exception("Le fichier n'est pas une image");  
    //Vérification de la bonne extension  
    if($extension !== "jpg" && $extension !== "jpeg" && $extension !== "png" && $extension !== "gif")  
        throw new Exception("L'extension du fichier n'est pas reconnu");  
    if(file_exists($target_file))  
        throw new Exception("Le fichier existe déjà");  
    if($file['size'] > 900000)//De préférence, mettre 500000  
        throw new Exception("Le fichier est trop gros");  
    if(!move_uploaded_file($file['tmp_name'], $target_file))  
        throw new Exception("l'ajout de l'image n'a pas fonctionné");  
    else return ($random."_".$file['name']);  
}
```

Je vais aussi générer un nombre aléatoire pour m'assurer que le nom du fichier téléchargé sera unique.

Je vais également apporter quelques modifications au fichier "**espacepro_controller.php**" pour lier ces règles que j'ai définies en utilisant des fonctions appropriées. Je n'oublierai pas de spécifier le répertoire dans lequel je souhaite stocker les images téléchargées.

7.0 Suppression d'images :

Lors de la suppression d'un identifiant de véhicule, il est nécessaire que je prenne des mesures pour supprimer les images associées qui sont encore présentes dans le répertoire que j'ai sélectionné.

Pour accomplir cette tâche, je vais ajouter quelques lignes de code supplémentaires à mon contrôleur de suppression dans l'espace professionnel et utiliser la fonction `unlink()`.

Cela permettra de nettoyer efficacement les fichiers image associés au véhicule supprimé et ainsi alléger le site.

DOSSIER PROFESSIONNEL (DP)



```
public function getimageVoiture($idVehicule){  
    $req = "SELECT imageVoiture from vehicule where idVehicule = :idVehicule";  
    $stmt = $this->getBdd()->prepare($req);  
    $stmt->bindValue(":idVehicule", $idVehicule, PDO::PARAM_INT);  
    $stmt->execute();  
    $image = $stmt->fetch(PDO::FETCH_ASSOC);  
    $stmt->closeCursor();  
    return $image['imageVoiture'];  
}  
  
public function getimageCritere($idVehicule){  
    $req = "SELECT imageCritere from vehicule where idVehicule = :idVehicule";  
    $stmt = $this->getBdd()->prepare($req);  
    $stmt->bindValue(":idVehicule", $idVehicule, PDO::PARAM_INT);  
    $stmt->execute();  
    $image = $stmt->fetch(PDO::FETCH_ASSOC);  
    $stmt->closeCursor();  
    return $image['imageCritere'];  
}
```

RESULTAT

1^{ère} partie :

Référence	Image Véhicule	Famille	Marque	Modèle	Année	Kilométrage	Boîte de Vitesse	Énergie	1 ^{ère} mise en Circulation	Puissance
128		Berline	citroen	2222	2010	222222	manuel	essence	0000-00-00	5
129		Citadine	citroen	20008	2015	120000	automatique	diesel	0000-00-00	4
130		Berline	citroen	gg	2019	111111	manuel	essence	0000-00-00	5

2ème partie :

Puissance	Places	Couleur	Description	Prix	Image Critère	Actions
5	5	blanc	<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua </p>	12345.00	 Emissions de CO ₂ : faible <ul style="list-style-type: none"> A B C D E F G Emissions de CO ₂ : élevées	Modifier Supprimer
4	6	vert	<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua </p>	13500.00	 Emissions de CO ₂ : faible <ul style="list-style-type: none"> A B C D E F G Emissions de CO ₂ : élevées	Modifier Supprimer
5	4	bleu	DDDDDDDDDDDDDDDDDDDDDDDDDD	12000.00	 Emissions de CO ₂ : faible <ul style="list-style-type: none"> A B C D E F G Emissions de CO ₂ : élevées	Modifier Supprimer

2. Précisez les moyens utilisés :

-Vscode

-Wamp

-MySql

-PhpMyAdmin

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ►

Cliquez ici pour taper du texte.

Chantier, atelier, service

Cliquez ici pour taper du texte.

Période d'exercice

► Du : [Cliquez ici](#)

au : [Cliquez ici](#)

DOSSIER PROFESSIONNEL (DP)

5. Informations complémentaires (*facultatif*)

Activité-type 2

Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité (PHP)

Exemple n° 2 ▶ API VEHICULE AVEC INTEGRATION DES FILTRES

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1.0 Mise en place de l'API 'véhicule'

Je vais utiliser l'architecture du modèle MVC en continuité de la logique de mon code.
vehicule_controller.php

Je vais créer le contrôleur qui va gérer les requêtes des filtres pour récupérer les données de la table véhicule à partir du modèle '**'vehiculeModel'**' .

A ce contrôleur, je vais lui créer une classe '**'VehiculeModel'**' dans la propriété '**'\$apiManager'**'.

```
● ● ●  
public function __construct() {  
    $this->espaceproManager = new EspaceproManager();  
}
```

J'ai ajouté à cette méthode un constructeur de classe. Je vais utiliser cette méthode pour initialiser chaque objet lorsqu'une instance de cette classe sera créée.

Elle sera appelée automatiquement à chaque fois que je créerais un nouvel objet, ici les filtres, à partir de cette classe.

J'ai ajouté **\$this** qui est un mot clé en Php et qui fait référence à l'objet de lui-même (instance de classe dans laquelle le code est en cours d'exécution).

Il s'agit donc d'un opérateur de sélection de propriété et qui va accéder à chaque propriété de l'objet filtre.

```
● ● ●  
public function getCarsByFilters($filtres)
```

Cette instance sera utilisée pour interagir avec **le modèle de données**.

Ensuite, je vais définir une méthode pour récupérer les véhicules en fonction des filtres passés en paramètre et après, je ferais appel au modèle pour récupérer les données.

Pour finaliser le paramétrage du contrôleur, je vais configurer les en-têtes http avec ‘**header()**’ pour spécifier le type de contenu **JSON** et permettre les requêtes en spécifiant les domaines autorisés comme les méthodes acceptées ou les en-têtes.



```
//Configurez les entêtes avant d'envoyer la réponse
header('Content-Type: application/json');
header("Access-Control-Allow-Origin: http://localhost:3000");
header("Access-Control-Allow-Methods: GET, POST, OPTIONS");
header("Access-Control-Allow-Headers: Content-Type");
```

2.0 API véhicules en intégrant la recherche par filtre

C'est dans ce code, que je vais gérer les requêtes http entrantes en fonction de la méthode qui sera ici en **GET** pour pouvoir récupérer les **paramètres sous format Json** fournis dans l'**Url**.

Je vais vérifier que si la méthode de la requête est **GET**, je ferais en sorte de continuer à traiter la requête.

Je vais faire vérifier les paramètres que j'ai effectué sur mon système de recherche par filtre et que je stockerais dans un tableau nommé **\$filters**, s'ils sont bien sûr définis par l'utilisateur dans la requête.

```
● ● ●

$filters = array();

if (isset($_GET['famille'])) {
    $filters["famille"] = $_GET['famille'];
}

if (isset($_GET['marque'])) {
    $filters["marque"] = $_GET['marque'];
}

if (isset($_GET['kilometremin'])) {
    $filters['kilometremin'] =
}intval($_GET['kilometremin']);
```

Ensuite je vais créer une instance de classe ‘**VehiculeController**’ et je vais appeler la méthode ‘**getCarsByFilters(\$filters)**’ sur cette instance.

Cela me permettra d’interagir entre le contrôleur des véhicules pour récupérer toute leurs données en fonction des filtres choisi par l’utilisateur.

Je vais utiliser la fonction de php ‘**intval()**’ pour pouvoir convertir une valeur en un entier que je stockerais ensuite dans une variable, cela me permettra de m’assurer qu’elle contienne uniquement une valeur numérique en ignorant tout autre caractère non numérique présent dans la chaîne d’origine.

```
● ● ●

if (isset($_GET['kilometremin'])) {
    $filters['kilometremin'] =
}intval($_GET['kilometremin']);
```

Cela me permettra de traiter les données provenant d’une source externe pour assurer que la valeur entrée ici, par la recherche par filtre, sera interprété uniquement comme un entier.

Comme pour le reste de mon projet, en ajoutant une condition, s’il y a une erreur comme une méthode qui n’est pas **GET**, il renverra une **réponse d’erreur 404**.



```
http_response_code(404);
echo json_encode(["error" => "endpoint not found"]);
```

3.0 vehicule_model.php

Je vais créer la classe ‘**VehiculeModel**’ qui va être utilisée pour effectuer des recherches de véhicules dans la Bdd.

Dans le constructeur de la classe, je vais préparer la connexion à la **Bdd** en utilisant **PDO** de Php. J’ai défini les identifiants de connexion pour avoir accès à la Bdd.

Je définie les infos de connexion dans des variables et si une erreur de connexion se passe, j’afficherais un message d’erreur.



```
public function __construct()
{
    $dsn = 'mysql:host=localhost;dbname=garage;charset=utf8';
    $user = 'root';
    $password = '';

    try {
        // Connexion à la base de données
        $this->dbh = new PDO($dsn, $user, $password);
    } catch (PDOException $e) {
        die('Erreur de connexion : ' . $e->getMessage());
    }
}
```

Je vais créer une autre classe ‘**getCarsByFilters**’ qui sera un **tableau associatif** contenant les filtres pour la recherche de véhicules.

La méthode va construire une requête Sql de base qui va sélectionner toutes les colonnes de la table ‘**vehicule**’ avec une condition **WHERE 1**, cela signifie que la requête devra toujours être vrai même s’il n’y a pas de filtres spécifiés.

```
● ○ ●  
$sql = "SELECT * FROM vehicule WHERE 1";
```

En fonction des filtres choisi par l’utilisateur dans le tableau **\$filters**, la méthode va ajouter des conditions supplémentaires à la requête Sql.

Je vais donner un exemple, si le filtre ‘**famille**’ est choisi, la méthode ajoutera une clause ‘**AND famille = :famille**’ à la requête et ainsi de suite pour les autres filtres.

La fonction **explode()** va diviser la valeur du filtre en un tableau en utilisant la virgule comme séparateur. Elles seront ensuite combiné en une chaîne unique séparée par des virgules à l’aide de la fonction **implode()**.

Str_replace va supprimer les virgules de chaque valeur et va les transformer en chaîne de caractère et la valeur sera ensuite concaténée.

```
● ○ ●  
if (isset($filters['famille'])) {les AND 1 par 1  
    $values = explode(",", $filters['famille']);  
    $namedPlaceholders = implode(' ', array_map(function ($value) {  
        return ':value_' . str_replace(',', '', $value);  
    }, $values));  
    $sql .= " AND famille IN ($namedPlaceholders)";
```

J’ai ajouté un filtre limite qui permettra de limiter le nombre de résultats retournés pour éviter que toutes les données soient retournées d’un seul coup et ne serait pas sympa visuellement même si je l’ai déjà aussi paramétré du côté React en limitant le nombre d’objet à afficher et aussi en incluant une pagination.

Au départ, j’ai préparé la requête Sql en liant les valeurs des filtres aux paramètres de la requête en utilisant **bindParam()** qui est une méthode de classe de **PDO**.

DOSSIER PROFESSIONNEL (DP)

```
● ● ●  
  
if (isset($filters['marque'])) {  
    $sql .= " AND marque = :marque";  
}  
  
if (isset($filters['kilometremin'])) {  
    $sql .= " AND kilometrage >= :kilometremin";  
}
```

J'avais un doute si je devais utiliser la fonction **bindParam()** ou **bindValue()** car apparemment, ces 2 fonctions sont assez similaires.

Après quelques recherches, j'ai décidé d'utiliser plutôt la fonction **bindParam()** car elle permet de lier une variable par référence, ce qui signifie que toute modifications de filtres apportés à la variable ‘filtres’ affectera automatiquement à la requête préparée.

Dans les 2 cas, je pense que cela aurait sans doute fonctionnée car le plus important, c'est de bien spécifier le type de données des champs qui sont dans la table de la Bdd pour éviter les certaines erreurs.

Je vais ensuite exécuter la requête en récupérant les résultats sous forme de tableau associatif en utilisant **fetchAll()** qui est aussi une classe de PDO.

Je vais par la suite stocker ces résultats dans une variable et renvoyer la méthode.

Résultat en GET :

```
[{"idVehicule": "25", "famille": "utilitaire", "marque": "citroen", "modele": "456", "annee": "2010", "kilometrage": "145000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-07-28", "puissance": "5", "places": "5", "couleur": "blanc", "reference": "", "prix": "12000.00", "imageVoiture": "40742_voiture1.png", "imageCritere": "85083_etaquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptatem ullam id dolore atque qui magnam magni. Aut magni voluptatem non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "26", "famille": "berline", "marque": "peugeot", "modele": "234", "annee": "2010", "kilometrage": "21000", "boitevitesse": "automatique", "energie": "electrique", "datecirculation": "2023-10-10", "puissance": "4", "places": "4", "couleur": "vert", "reference": "", "prix": "33000.00", "imageVoiture": "21752_voiture2.png", "imageCritere": "64624_etaquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptatem ullam id dolore atque qui magnam magni. Aut magni voluptatem non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "28", "famille": "citadine", "marque": "kia", "modele": "456", "annee": "2011", "kilometrage": "78000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-10-06", "puissance": "5", "places": "5", "couleur": "blanc", "reference": "", "prix": "12000.00", "imageVoiture": "94757_voiture3.png", "imageCritere": "80972_etaquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptatem ullam id dolore atque qui magnam magni. Aut magni voluptatem non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "29", "famille": "berline", "marque": "kia", "modele": "900", "annee": "2022", "kilometrage": "34000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-09-09"}]
```

2. Précisez les moyens utilisés :

- Vscode
- Wamp
- MySql
- PhpMyAdmin

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (*facultatif*)

Activité-type 3

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)

Exemple n° 1 ▶ CREATION DES COMPOSANTS CARD VEHICULE

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1.0 Crédit du Composant Card pour afficher les Véhicules

J'ai voulu présenter les véhicules aux utilisateurs sous forme de cartes Bootstrap, mettant en avant quelques informations essentielles concernant chaque véhicule.

Pour réaliser cela, j'ai connecté ce composant directement à la base de données, extrayant les données de la table '**véhicule**' à l'aide **d'Axios**.

Cette approche me permet de récupérer toutes les informations relatives de chaque véhicule.

Le composant '**Card**' a été conçu pour être réutilisable, affichant de manière individualisée les détails d'un véhicule à la fois, sous forme d'une carte distincte.



BMW

Modèle: 456

Energie: électrique

Prix: 23000.00 €

[En savoir plus](#)

```

const Card = (props) => {
  const image = `http://localhost/garageback/public/images/${props.image}`;
  return (
    <>
      <div className="card">
        <div className="card-body">
          <a
            href={props.image}
            target="_blank"
            rel="noopener noreferrer"
          >
            <img src={image} alt={props.marque} className="img-fluid rounded mx-auto d-block mb-3"/>
          </a>

          <h5 className="card-title text-primary">{props.marque.toUpperCase()}</h5>
          <p className="card-text">Modèle: {props.modele}</p>
          <p className="card-text">Energie: {props.energie}</p>
          <p className="card-text fw-bold text-primary">Prix: {props.prix} €</p>
        </div>
        <div className="card-footer">
          <Link
            to={`/vehiculefiche/${props.id}`}
            className="btn btn-primary"
          >
            En savoir plus
          </Link>
        </div>
      </div>
    </>
  );
};

export default Card;

```

Je vais ajouter des paramètres aux propriétés du composant '**Card**' pour représenter les informations du véhicule, telles que l'image, la marque, le modèle, etc.

Pour obtenir l'image, je vais construire le chemin en utilisant l'**URL** de base, indiquant où sont stockées les images.

Ensuite, je vais transmettre cette URL via la propriété '**props.image**', ce qui me permettra d'éviter de réécrire l'URL chaque fois que j'aurai besoin de récupérer une image .

Composant '**vehiculeCard**'

Pour ce composant, je vais importer plusieurs dépendances, notamment '**useState**', qui me permettra de gérer l'état local pour stocker la liste des véhicules, le numéro de la page actuelle '**currentPage**', et le nombre de cartes affichées par page que je vais définir ultérieurement '**cardsPerPage**'.

DOSSIER PROFESSIONNEL (DP)



```
const VehiculesCard = () => {
  const [vehicules, setVehicules] = useState([]);
  const [currentPage, setCurrentPage] = useState(1);
  const cardsPerPage = 6;

  useEffect(() => {
    axios
      .get("http://localhost/garageback/front/voiturefiche/all")
      .then((response) => {
        const jsonData = response.data;
        const sortedVehicles = [...jsonData];
        const sortByCreatedAt = (a, b) => {
          const dateA = new Date(a.created_at).getTime();
          const dateB = new Date(b.created_at).getTime();
          return dateA - dateB;
        };
        sortedVehicles.sort(sortByCreatedAt);
        setVehicules(sortedVehicles);
      })
      .catch((error) => {
        console.error("Erreur lors de la récupération des véhicules :",
          error));
  });
}
```

Mon objectif est de permettre au composant d'effectuer une requête à partir de l'**API** 'véhicule' que j'ai créée du côté serveur, afin d'obtenir la liste de tous les véhicules à afficher.

Pour cela, je vais utiliser '**axios**' dans la fonction '**useEffect**'. Mon intention est de faire en sorte que les cartes s'affichent sur la page d'accueil en fonction de leur date de création la plus récente.

Cela permettra de renouveler régulièrement les cartes et de proposer aux utilisateurs les derniers véhicules en stock.

Le composant va ensuite mapper les véhicules actuellement affichés dans une liste de composants '**Card**' créés précédemment, en transmettant les informations du véhicule en tant que propriétés à chaque composant '**Card**'.

```
<div className="row">
  {currentCards.map((vehicule) => (
    <div
      key={vehicule.idVehicule}
      className="col-lg-4 col-md-4 col-sm-6 col-6 mt-3"
    >
      <Card
        image={vehicule.imageVoiture}
        marque={vehicule.marque}
        nom={vehicule.nom}
        modele={vehicule.modele}
        energie={vehicule.energie}
        prix={vehicule.prix}
        id={vehicule.idVehicule}
      />
    </div>
```

2.0 Mise en Place de la Pagination des cartes

Je vais mettre en œuvre un système de pagination pour diviser les véhicules en groupes, limitant ainsi le nombre de véhicules affichés par page grâce à la variable '**'cardsPerPage'**'.

Ce mécanisme permettra aux utilisateurs de naviguer d'une page à l'autre.

Le nombre total de pages sera calculé en fonction du nombre total de véhicules et du nombre de cartes par page, une valeur que je vais définir ultérieurement.

```
const indexOfLastCard = currentPage * cardsPerPage;
const indexOfFirstCard = indexOfLastCard - cardsPerPage;
const currentCards = vehicules.slice(indexOfFirstCard, indexOfLastCard);

const paginate = (pageNumber) => {
  setCurrentPage(pageNumber);
};

console.log(currentCards);
  console.error("Erreur lors de la récupération des véhicules :",
error));};
```

DOSSIER PROFESSIONNEL (DP)



```
<Pagination>
  {Array.from(
    { length: Math.ceil(vehicules.length / cardsPerPage) },
    (_, index) => (
      <Pagination.Item
        key={index}
        active={index + 1 === currentPage}
        onClick={() => paginate(index + 1)}
      >
        {index + 1}
      </Pagination.Item>
    )
  )}
</Pagination>
```

1 2

Le nombre maximal de cartes affichées sera de 9 véhicules, ce qui correspond à 3 cartes par ligne en mode desktop et 2 en mode mobile, tant sur la page d'accueil que sur la page de voitures d'occasion résultant d'une recherche par filtres ou d'un classement par ordre de date de création.

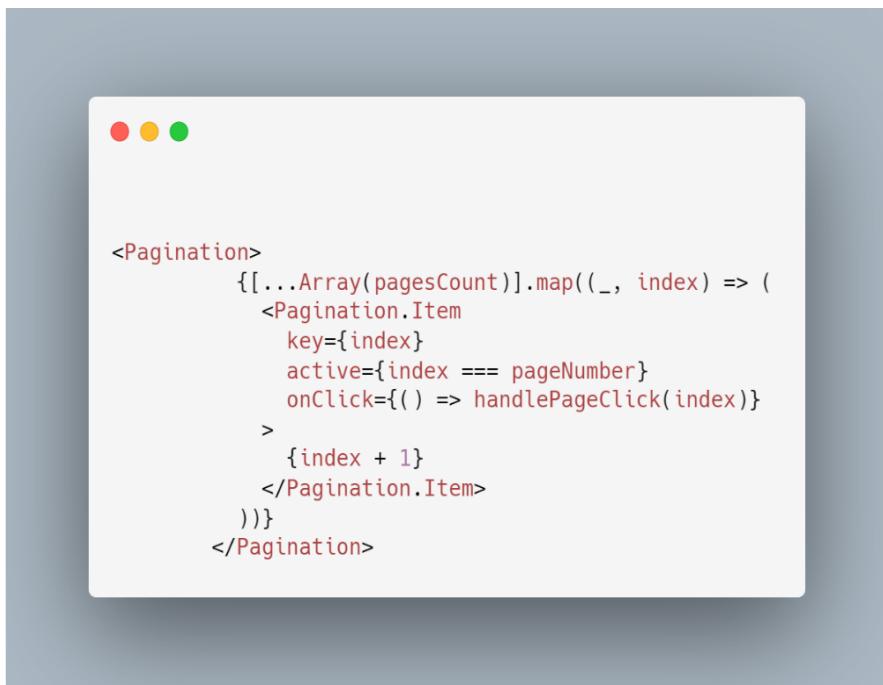
Cette limitation permettra d'améliorer significativement les performances de l'application et éviter que tous les véhicules de la base de données s'affichent en même temps.

Affichage des informations complète du véhicule

Lorsque l'utilisateur clique sur "**plus d'infos**" sur la carte, une nouvelle page s'affichera, présentant toutes les informations du véhicule de manière plus détaillée, toujours sous forme de carte, mais de manière plus large.

Si l'utilisateur souhaite obtenir davantage d'informations sur un produit, je veillerai à le rediriger vers la page de contact.

Dans une étape future, que je n'ai pas encore eu l'occasion de mettre en place, je prévois d'automatiquement inclure l'ID du véhicule dans le formulaire de contact. Cela permettra à l'administrateur de retrouver plus facilement le véhicule en se référant à son identifiant, ce qui entraînera un gain de temps considérable pour lui.



2. Précisez les moyens utilisés :

- Vscode**
- Wamp**
- MySql**
- PhpMyAdmin**
- React**

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

DOSSIER PROFESSIONNEL (DP)

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (facultatif)

Activité-type 3

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)

Exemple n° 2 ► DEVELOPPEMENT DES COMPOSANTS FILTRES

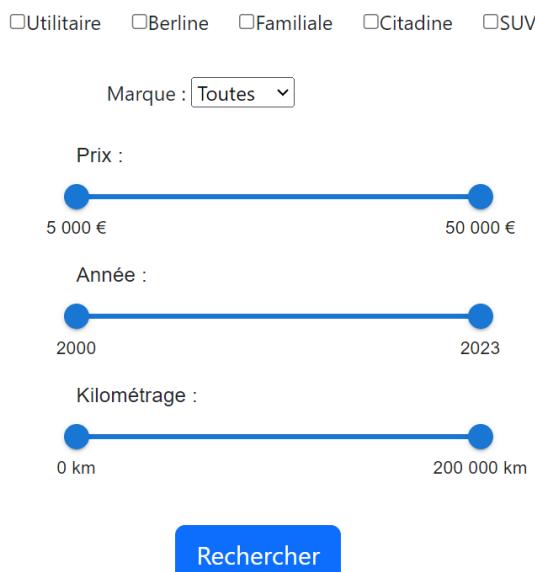
1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1.0 Mise en Place des Composants de Recherche et de Filtrage

Au cours de cette phase de développement, j'ai conçu des éléments d'interface utilisateur (composants), notamment des **sliders**, des sélecteurs (**dropdowns**), et des cases à cocher (**checkboxs**), visant à faciliter la recherche et le filtrage des données.

Pour améliorer l'expérience utilisateur, je vais créer des **sliders** prix, km et année en offrant la possibilité à l'utilisateur de définir à la fois une valeur **minimale et maximale** pour chaque paramètre.

Cette approche permettra aux utilisateurs de mieux cibler leurs critères de recherche.



J'ai structuré le code en créant des composants distincts pour chaque élément de filtrage, qu'il s'agisse de **cases à cocher, de menus déroulants, ou de sliders**.

Ces composants individuels communiquent avec un composant parent centralisé nommé "**VehiculeFilters**" pour gérer l'état des filtres de manière cohérente.

Ne pouvant directement communiquer entre eux, je vais créer pour les composants parents et fils une fonction **handlechange** qui prendra en paramètre un événement lorsque l'utilisateur cliquera sur le bouton rechercher.

Elle sera définie pour mettre à jour l'état des filtres lorsque l'utilisateur effectuera une sélection dans l'un des composants filtre.

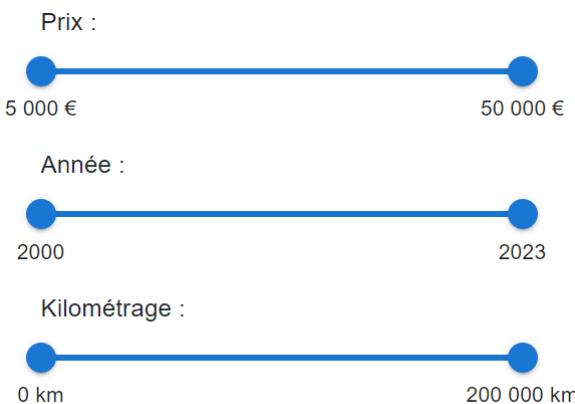
Je la passerai en tant que **props** aux composants fils pour qu'ils puissent **mettre à jour** l'état parent. Chaque composant a été créé en utilisant la bibliothèque **Material-UI** (<https://mui.com/>).

J'ai installé les dépendances nécessaires pour garantir leur bon fonctionnement ce qui m'a beaucoup aidé car au départ, j'ai voulu créer les composants de filtres par moi-même, ça fonctionnait mais visuellement, ce n'était pas terrible.

2.0 Composant ‘BasicRange’

Je vais faire en sorte de créer la fonction qui attendra en paramètre un objet et qui utilisera le **destructuring** en extrayant les propriétés de l'objet dans des variables locales.

Si l'objet passé en paramètre ne contient pas de propriétés spécifiques, alors je mettrai une valeur par défaut.



Après avoir effectué des recherches, j'ai conclu que le ‘**destructuring**’ est une technique efficace pour extraire des valeurs à partir d'objets ou de tableaux, ce qui a motivé mon choix.

Pour ce qui est du tri d'un tableau, j'ai opté pour l'utilisation de la fonction `sort()` sans recourir au ‘destructuring’.

Cette fonction trie les éléments du tableau en les comparant les uns aux autres, ce qui me permet de réorganiser les éléments du tableau de manière à les placer dans un ordre spécifique.

```
range.sort((a, b) => a - b);
```

```

const BasicRange = ({ name = "slider", range = [0, 100], marks, label = "", handleChange }) => {
    // Utilisation de destructuring pour extraire les valeurs des props avec des valeurs par défaut

    // Tri du tableau range si nécessaire, de sorte que la valeur la plus petite soit toujours en 1ère
    // position et la plus grande en dernière
    // On l'utilisera car pour les 3 composants prix, km et année, leurs valeurs sont des entiers et que
    // sort trie uniquement des chaînes de caractères.

    range.sort((a, b) => a - b);

    // Utilisation de useState pour gérer la valeur actuelle du slider
    const [value, setValue] = useState(range);

    // Fonction de gestion du changement de valeur du slider
    const handleSliderChange = (event, newValue) => {
        setValue(newValue);
        handleChange(name, newValue); // Appel de la fonction handleChange du parent avec le nom et la
        // nouvelle valeur
    };
}

```

Le composant ‘BasicRange’ comporte les éléments suivants :

- **handleChange** : Cette fonction sera appelée à chaque modification de la valeur du slider et recevra deux paramètres : le nom du paramètre (comme prix, kilométrage, année) et la nouvelle valeur sélectionnée.

- **label** : Un libellé qui s'affiche au-dessus du slider pour aider l'utilisateur à comprendre les valeurs sélectionnées.

- **name** : Le nom du slider.

- **marks** : Un tableau d'objets spécifiant les marques (valeurs) sur le slider, chaque objet ayant une valeur et une étiquette (label). Par exemple, pour le prix : "5 000 - 50 000".

- **range** : La plage de valeurs possibles du slider. J'ai utilisé une plage de base de 0 à 100, qui convient à de nombreuses situations.

J'ai veillé à ce que la plus petite valeur soit toujours en première position en utilisant une fonction de tri pour maintenir l'ordre correct.

DOSSIER PROFESSIONNEL (DP)

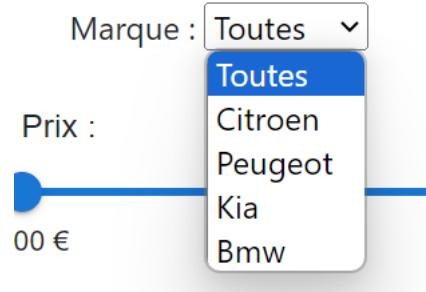
```
return (
  <>
  <Typography id="input-slider" gutterBottom>
    {label}
  </Typography>

  <Slider
    name={name}
    min={range[0]}
    max={range[1]}
    onChange={handleSliderChange}
    size="medium"
    valueLabelDisplay="auto" // Afficher la valeur actuelle
    marks={marks || [{ value: range[0], label: range[0].toString() }, { value: range[1], label: range[1].toString() }]}
    value={value}

  />
  </>
);
};

export default BasicRange;
```

3.0 Composant 'BasicSelect'



J'ai créé ce composant en un menu déroulant avec des options configurables, telles que les marques de voitures.

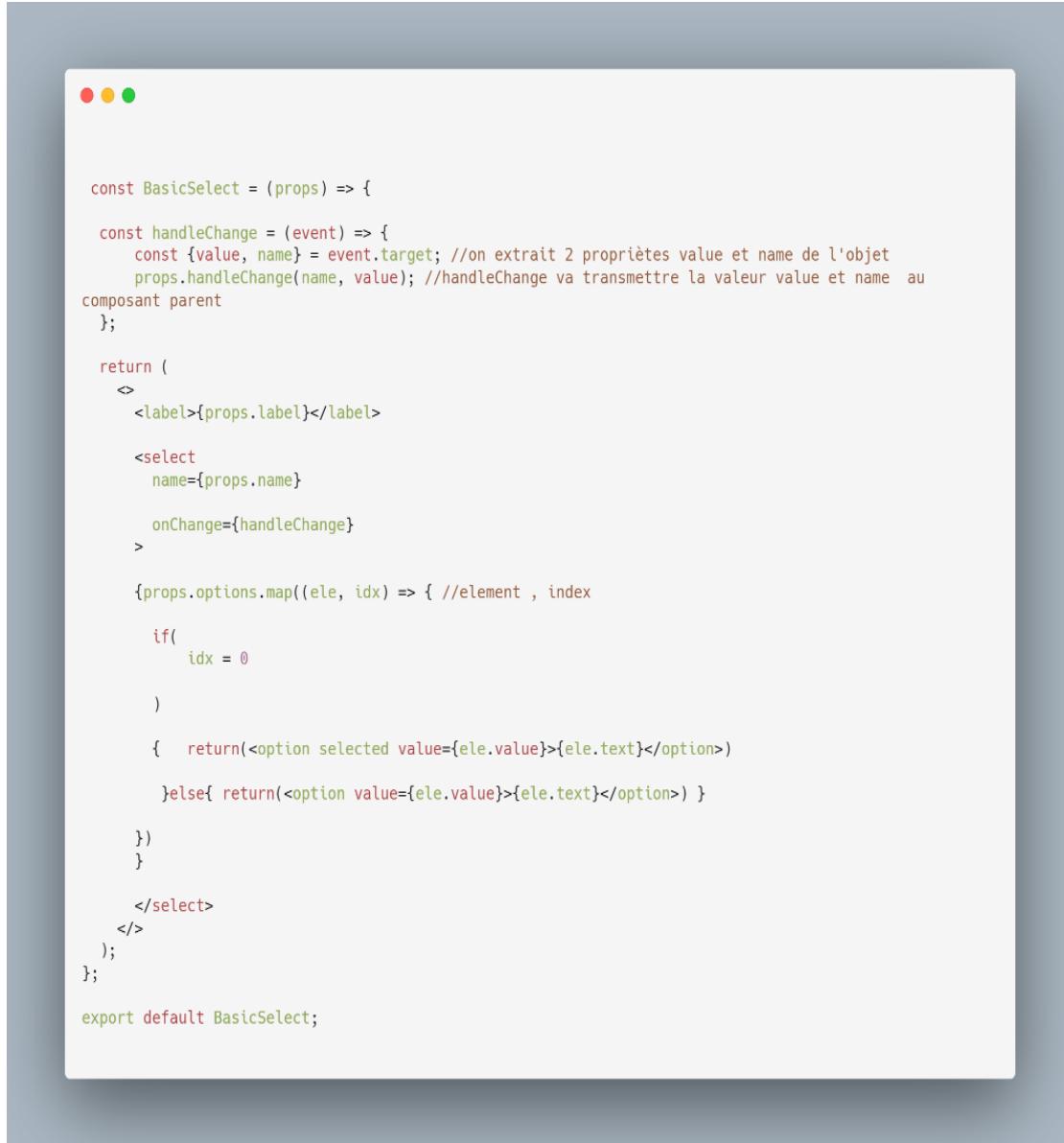
Lorsque l'utilisateur sélectionne une option, la fonction **handleChange** sera utilisée pour communiquer avec le composant parent.

Cette fonction sera appelée à chaque fois que l'utilisateur modifie la sélection dans le menu déroulant. Elle prendra en paramètre l'objet **event**.

Marque : **Toutes**

Je vais utiliser le **destructuring** pour extraire les propriétés **value** et **name** de l'objet **event**.

Je vais ensuite appeler la fonction **props.handleChange(name, value)** ; pour transmettre les valeurs au composant parent ce qui permettra à ce composant de réagir à la sélection effectuée dans le menu déroulant.



```
const BasicSelect = (props) => {
  const handleChange = (event) => {
    const {value, name} = event.target; //on extrait 2 propriétés value et name de l'objet
    props.handleChange(name, value); //handleChange va transmettre la valeur value et name au
    composant parent
  };

  return (
    <>
      <label>{props.label}</label>

      <select
        name={props.name}

        onChange={handleChange}
      >

        {props.options.map((ele, idx) => { //element , index

          if(
            idx = 0
          )

          {  return(<option selected value={ele.value}>{ele.text}</option>)

            }else{ return(<option value={ele.value}>{ele.text}</option>) }

          }
        })

        </select>
    </>
  );
};

export default BasicSelect;
```

4.0 Composant ‘BasicCheckbox’

Ce composant servira à sélectionner une famille de véhicule. L'utilisateur pourra en sélectionner plusieurs s'il le souhaite ou si rien de cocher, je ferais en sorte de lui présenter tout type de véhicules.

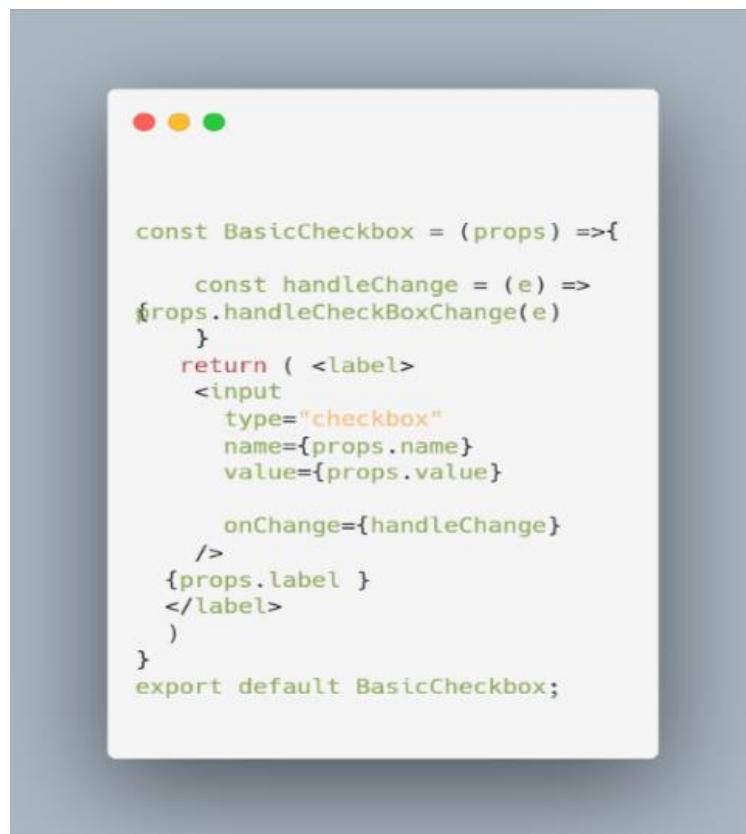
Utilitaire Berline Familiale Citadine SUV

Comme pour les autres composants filtres, la fonction **handleChange** sera appelée chaque fois que l'utilisateur coche ou décoche une case.

La fonction prendra en charge un objet **événement** en tant que paramètre.

DOSSIER PROFESSIONNEL (DP)

Je vais par la suite transmettre cet événement au composant parent grâce à la fonction **props.handleCheckBoxChange(e)**, cela permettra au composant parent de réagir à l'état de la case si cochée ou pas.



```
const BasicCheckbox = (props) =>{  
  const handleChange = (e) =>  
    props.handleCheckBoxChange(e)  
  }  
  return (  
    <label>  
      <input  
        type="checkbox"  
        name={props.name}  
        value={props.value}  
        onChange={handleChange}  
      />  
      {props.label}  
    </label>  
  )  
}  
export default BasicCheckbox;
```

2. Précisez les moyens utilisés :

-Vscode

-Wamp

-MySql

-PhpMyAdmin

-React

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (facultatif)

Activité-type 3

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)

Exemple n° 3 ▶ CONCEPTION DE LA PAGE DE RECHERCHE PAR FILTRES

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1.0 Composant Parent 'VehiculeFilters'

La page nommée "Voitures d'occasion" servira donc de parent pour intégrer les composants enfants que j'ai créés précédemment, à savoir `BasicCheckbox`, `BasicSelect`, et `BasicRange`.

C'est effectivement sur cette page que l'utilisateur effectuera sa recherche en ajustant les filtres en fonction de ses choix.

Pour le slider "année", j'ai créé une fonction permettant de mettre à jour la date de fin du slider en temps réel, évitant ainsi une mise à jour manuelle chaque année.



J'ai implémenté la fonction **handleChange**, qui est transmise aux composants enfants via les **props** et qui met à jour l'état local des filtres en fonction des modifications apportées par l'utilisateur.



```
const handleChange = (name, newValue) => {
  setFiltres({ ...filtres, [name]: newValue });
  //prendra 2 paramètres name (le nom du filtre à mettre à jour et newValue, la nouvelle valeur du filtre.
};
```

La fonction **handleCheckboxChange** sera utilisée pour gérer les cases à cocher. Elle mettra à jour l'état des filtres en fonction de ce qui est coché ou pas.

Dans cette fonction, j'extrais les 3 propriétés de l'objet **événement** : **name** qui est le nom de la case à cocher, **value** qui est la valeur associée à la case à cocher et **checked** qui sera un booléen qui indiquera (**true**) si cochée ou (**false**) si décochée.



```
const handleCheckBoxChange = (e) => {
  const { name, value, checked } = e.target;
  if (checked) {
    setFiltres({ ...filtres, [name]: [...filtres.famille, value] });
  } else {
    setFiltres({
      ...filtres,
      [name]: filtres.famille.filter((ele) => ele !== value),
    });
  }
};
```

2.0 Hook UseEffect

Je vais ensuite utiliser le **hook useEffect** pour effectuer les requêtes http grâce à **Fetch** lorsqu'il sera monté (affiché pour la première fois) grâce à **AXIOS** qui effectuera ces requêtes et ainsi récupérer les données des véhicules en fonction des paramètres de filtres choisi par l'utilisateur.



```
useEffect(() => {
  fetch(
    //fetch effectue une requête http, si reponse, elle sera encapsulé dans une promesse
    lien
    // "http://localhost/GarageBack/API/vehicule.php?
    kilometremin=0&kilometremax=200000&anneemin=2000&anneemax=2023&prixmin=5000&prixmax=50000"
  )
  //Si reponse reçu de la requête http, then va gérer la réponse de cette promesse et va prendre une
  fonction de rappel en argument:

  .then((res) => res.json())// Va extraire les données de l'API sous format json
  .then((data) => {
    setCards(data)
    console.log();
  })//data ou on aurait pu mettre un nom représente la réponse de la requête http
  .catch((err) => console.log(err));//Si erreur de la requête, catch retourne une erreur
}, [lien]);
```

3.0 Fonction handleClick

Je vais créer une fonction qui va construire une Url dynamique en fonction des filtres sélectionnés dans l'objet **lienObjet**. Il supprimera le dernier caractère ‘&’ pour obtenir **une Url propre**.



```
const handleClick = ()=> {

    let lienTmp = "http://localhost/garageback/API/vehicules.php?";
    let lienObject = {kilometremin:filtres.kilometrage[0],
                     kilometremax:filtres.kilometrage[1],
                     prixmin:filtres.prix[0],
                     prixmax:filtres.prix[1],
                     anneemin:filtres.annee[0],
                     anneemax:filtres.annee[1],

    };
    if(filtres.marque.length !== 0){
        lienObject.marque = filtres.marque;
    }

    if(filtres.famille.length !== 0){
        lienObject.famille = filtres.famille.join(",");
    }

    for(const [cle, valeur] of Object.entries(lienObject)){
        lienTmp = lienTmp + `${cle}=${valeur}&`;
    }
    lienTmp = lienTmp.slice(0, -1);
    // console.log(lienTmp)
    setLien(lienTmp)
}
```

4.0 Postman

Avant de créer la variable `Lien`, j'ai effectué des tests sur mes URLs de recherche par filtres en utilisant **Postman**, ce qui m'est avéré être d'une grande aide pour la gestion des erreurs.

Je renseignais dans mon URL les valeurs **minimales et maximales** des filtres que j'avais précédemment définies.

Exemples de Liens de test en méthode GET:

Pour tester la recherche par marque :

<http://localhost/GarageBack/API/vehicule.php?marque=citroen>

Pour tester la recherche de tous les filtres avec les valeurs de début et de fin renseignées :

<http://localhost/GarageBack/API/vehicule.php?kilometremin=0&kilometremax=200000&anneemin=2000&anneemax=2023&prixmin=5000&prixmax=50000>

DOSSIER PROFESSIONNEL (DP)

The screenshot shows the Postman interface with a collection named "garage" containing a "vehicle" folder. A specific API endpoint is selected: `http://localhost/garageback/API/vehicules.php?marque=citroen&kilometremin=0&kilometremax=200000`. The "Params" tab is active, showing the following query parameters:

Key	Value	Description	... Bulk Edit
marque	citroen		
kilometremin	0		
kilometremax	200000		
anneemin	2000		

The "Body" tab shows the raw JSON response from the API call:

```
52 |     "marque": "CITROEN",
53 |     "modele": "333",
54 |     "annee": "2010",
55 |     "kilometrage": 3333,
56 |     "boitevitesse": "manuel",
57 |     "energie": "essence",
58 |     "datecirculation": "2023-10-14",
59 |     "puissance": "5",
60 |     "---.c
```

The Postman toolbar at the bottom indicates the request was made at 11:19 on 11/10/2023.

Bien sûr après que tous ces paramétrages ont été réalisé, je retourne tous les composants de filtrage.

5.0 Difficultés rencontrées ‘Recherches par filtres’ :

Cette partie du développement a été l'une des plus complexes pour moi. Tout d'abord, j'ai initialement essayé de créer mes propres composants de sliders, de cases à cocher, etc., alors que des bibliothèques existaient pour simplifier ce processus mais au départ, je n'en avais pas eu connaissance, n'étant pas satisfait du résultat, j'ai réalisé plusieurs recherches pour trouver une solution et proposer au client un front plus présentable.

J'ai aussi éprouvé des difficultés à bien cibler mes recherches, mais j'ai heureusement bénéficié de conseils avisés qui m'ont facilité la tâche. Mon apprentissage s'améliore au fil du temps que je pratique, et je suis désormais conscient des ressources disponibles pour résoudre de tels problèmes techniques.

De plus, j'ai rencontré des défis lors de la mise en place de la fonction `handleChange` et de sa transmission en tant que `props` aux composants enfants, mais j'ai compris l'importance de cette fonction pour établir la communication entre les composants.

J'ai également été confronté à plusieurs erreurs liées à des noms de composants mal orthographiés, tant au niveau des noms de fichiers que des imports.

Je suis conscient de la sensibilité à la casse dans ce contexte, mais dans ce cas précis, j'ai eu du mal à identifier l'origine de l'erreur, car tous les noms semblaient correctement écrits.

Pour résoudre ce problème, j'ai dû supprimer tous les fichiers des composants et de les recréer un à un afin que l'erreur cesse de s'afficher.

C'est l'un des mystères de l'informatique qui peut parfois entraîner une perte de temps considérable pour ce qui semble être une petite erreur en apparence.

La convention stipule que les fichiers doivent commencer par une lettre majuscule. Cependant, de manière inexplicable, il arrive parfois que certains fichiers se renomment automatiquement, perdant ainsi leur lettre majuscule initiale.

Cela provoque des erreurs lors de l'importation de ces fichiers. Je ne parviens pas à expliquer pourquoi cela se produit, mais cette situation devient de plus en plus frustrante à chaque occurrence.

2. Précisez les moyens utilisés :

- Vscode**
- Wamp**
- MySql**
- PhpMyAdmin**
- React**
- Postman**

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (*facultatif*)

DOSSIER PROFESSIONNEL (DP)

Activité-type 4

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)

Exemple n° 1 ▶ AVIS CLIENTS

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

Avis Clients

Cette partie représente la dernière étape que j'ai pu accomplir dans ce projet, mais elle est encore loin d'être achevée.

Je récupère bien mes données en **Get** mais en Post, je n'y suis pas encore parvenu, j'ai des problèmes au niveau de la requête.

J'ai créé un composant d'étoiles pour la notation et j'ai installé la bibliothèque '**npm install react-simple-star-rating**'. **Opérationnel**

Il n'y a même pas besoins de cliquer sur les étoiles pour les sélectionner, un survol de la souris suffira.



J'ai décidé ici de ne pas inclure les demis étoile pour l'instant, je verrais cela plus tard car j'aimerais bien savoir comment faire.

The screenshot shows a code editor window with a dark theme. At the top, there are three small colored circles (red, yellow, green) which are part of the Mac OS X window control buttons. The main area contains a piece of JavaScript code:

```
const Stars = () => {
  const [rating, setRating] = useState(100) // Va jusqu'à 100 donc la 5 étoiles
  const handleRating = (rate) => {
    console.log(rate)
    setRating(rate)
  }
  return (
    <Rating
      fillColor="#F0C300"
      //allowHalfIcon //Pour les demi étoile, là, on est à true
      // tooltipArray={[ 'nul', 'baf', 'moyen', 'top', 'génial' ]}
      transition
      // showTooltip
      onClick={handleRating}
      ratingValue={rating}
    />
  )
}
export default Stars;
```

Je vais maintenant convertir les étoiles en note en créant un nouveau composant ‘ConversionNote’
A l’intérieur de la fonction, je vais créer une variable d’état ‘note’ à l’aide du Hook ‘useState’ et je vais l’initialiser à 0.



```
const ConversionNote = () => {
  const [note, setNote] = useState(0);
  const handleNoteChange = (newNote) => {
    setNote(newNote);
  };
  const etoiles = [];

  for (let i = 1; i <= 5; i++) {
    etoiles.push(
      <Stars
        key={i}
        selected={i <= note}
        onEtoileClick={() => handleNoteChange(i)}
      />
    );
  }
  return (
    <div className="rating-container">
      <div className="five-rate-active">{etoiles}</div>
    </div><p>Note : {note}</p>
  );
};

export default ConversionNote;
```

J’ai testé dans la console en y ajoutant un echo, ça fonctionne parfaitement

Note :



Download the React DevTools for a better development experience: <https://reactjs.org/link/react-devtools>

✖ Error while trying to use the following icon from the Manifest: <http://localhost:3000/logo192.png> (Download error or resource isn't a valid image)

4

Rating.js:8

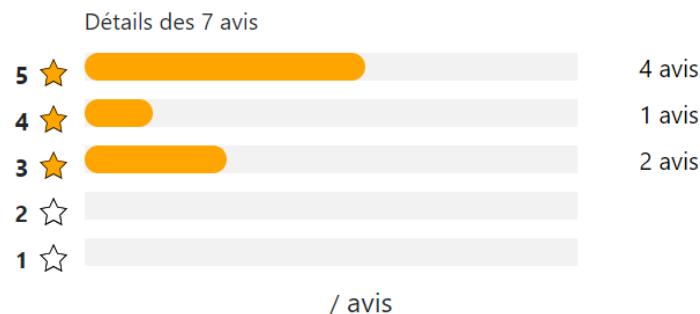
4

Rating.js:8

>

DOSSIER PROFESSIONNEL (DP)

J'ai créé un composant destiné à informer l'utilisateur et à générer un bilan des avis clients (bien que non opérationnel pour le moment).



Par la suite, sur la page d'accueil, j'ai l'intention d'afficher les avis de manière dynamique. Pour ce faire, je vais générer un nombre aléatoire qui sélectionnera aléatoirement un avis à afficher.

Ce garage a fait un excellent travail pour réparer ma voiture. Ils m'ont expliqué en détail ce qui devait être fait et ont répondu à toutes mes questions. J'apprécie leur honnêteté et leur professionnalisme. Mon véhicule fonctionne parfaitement maintenant.

Pauline
 ★★★★★

Laisser Un Avis !

2. Précisez les moyens utilisés :

-Vscode

-React

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (facultatif)

Activité-type 4

Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité (REACT, JS et BOOTSTRAP)

Exemple n° 2 ▶ FORMULAIRE DE CONTACT

1. Décrivez les tâches ou opérations que vous avez effectuées, et dans quelles conditions :

1.0 Formulaire de contact

Je viens d'ajouter une route vers le formulaire de contact qui a été créée dans le fichier `Site.js` , avec son composant associé.

The form is titled "Nous écrire". It contains the following fields:

- Prénom
- Nom
- Téléphone
- Email (with a placeholder '@' and a text input field)
- Objet : Renseignements (dropdown menu)
- Message (large text area)
- Envoyer (blue button)

J'ai développé la **view** du formulaire de la page de contact sous bootstrap dans un conteneaire.

J'ai utilisé la bibliothèque **Formik** pour faciliter la gestion des formulaires et la bibliothèque **Yup** pour imposer des restrictions lors du remplissage des champs du formulaire.

J'ai ajouté certaines restrictions pour les champs de formulaire pour éviter d'éventuelles erreurs que pourrait faire l'utilisateur.

```

export default withFormik({
  mapPropsToValues: () => ({
    firstName: "",
    lastName: "",
    phoneNumber: "",
    email: "",
    message: ""
  }),
  validationSchema: Yup.object().shape({
    firstName: Yup.string()
      .min(2, "Trop court !")
      .max(50, "Trop long !")
      .required("Veuillez saisir votre prénom."),
    lastName: Yup.string()
      .min(2, "Trop court !")
      .max(50, "Trop long !")
      .required("Veuillez saisir votre nom."),
    phoneNumber: Yup.string()
      .min(10, "Trop court !")
      .max(10, "Trop long !")
      .required("Veuillez saisir votre numéro de téléphone."),
    email: Yup.string()
      .email("Veuillez saisir un email valide.")
      .required("Veuillez saisir votre email."),
    message: Yup.string()
      .min(10, "Trop court !")
      .max(1000, "Trop long !")
      .required("Veuillez saisir votre message."),
  }),
  handleSubmit: (values, { props }) => {
    const message = {
      firstName: values.firstName,
      lastName: values.lastName,
      phoneNumber: values.phoneNumber,
      email: values.email,
      message: values.message,
    };
    props.sendMail(message);
  },
})(Form);

```

Comme le formulaire n'est pas en ligne pour l'instant, je ne peux pas savoir s'il est fonctionnel ou pas mais dans mes autres projets que j'ai réalisé pour mon Cv en ligne ou mon client 'Les caravanes de le besbre', ils le sont.

2.0 Ajout de Google reCAPTCHA

Veuillez cocher la case ci-dessous pour continuer.

Je ne suis pas un robot
 
 reCAPTCHA
Confidentialité · Conditions

DOSSIER PROFESSIONNEL (DP)

Afin de renforcer la sécurité du formulaire de contact et prévenir les spams, j'ai intégré Google reCAPTCHA.

L'installation du reCAPTCHA a été effectuée en utilisant la commande `npm install react-google-recaptcha`, puis je l'ai importé par la suite dans le projet.

J'ai généré un identifiant unique de sécurité grâce à Google reCAPTCHA. Cet identifiant doit être stocké dans un fichier caché, de manière à ce qu'il ne soit pas accessible en ligne, renforçant ainsi la sécurité de l'application.

```
import ReCAPTCHA from "react-google-recaptcha";

const Recaptcha = ({ onChange }) => {
  return (
    <ReCAPTCHA
      sitekey="6LcAhge-NEXrK24tmIgsSaL0_ZdUaJzXJ"
      />
  );
}

export default Recaptcha;
```

2. Précisez les moyens utilisés :

3. Avec qui avez-vous travaillé ?

Ce projet a été réalisé seul.

4. Contexte

Nom de l'entreprise, organisme ou association ► *Cliquez ici pour taper du texte.*

Chantier, atelier, service ► *Cliquez ici pour taper du texte.*

Période d'exercice ► Du : *Cliquez ici* au : *Cliquez ici*

5. Informations complémentaires (*facultatif*)

Conclusion

En fin de compte, la mise en place des composants en ce qui concerne la recherche par filtre a été une étape assez instructive en ce qui me concerne par rapport à mon apprentissage.

J'ai été confronté à de nombreux problèmes, que ce soit avec les commandes Git et Github avec beaucoup de conflits, la connexion à la base de données, la création des tables et bien entendu le code.

Souvent, je passais trop de temps à me battre avec un problème, mais j'ai fini par réaliser que cela ne servait à rien de rester des jours à essayer de le résoudre.

Maintenant, je préfère passer à autre chose temporairement, le temps de réfléchir à la meilleure approche pour le résoudre.

Cette méthode me convient bien. Cette expérience m'a permis de développer mes compétences en gestion de bases de données et d'acquérir une compréhension plus approfondie des aspects pratiques du développement.

Mon apprentissage continu dans ce domaine m'aidera à aborder de futurs projets avec plus de confiance et de compétence.

J'ai pris conscience que la mémorisation forcée ne mène à rien, car il est impossible d'absorber l'ensemble des informations par cœur.

La méthode qui fonctionne le mieux pour moi consiste à pratiquer, pratiquer et pratiquer de manière intensive tout en effectuant des recherches constantes.

DOSSIER PROFESSIONNEL (DP)

Titres, diplômes, CQP, attestations de formation

(facultatif)

Déclaration sur l'honneur

Je soussigné(e) [prénom et nom] **HOFFMANN MICHEL**,

Déclare sur l'honneur que les renseignements fournis dans ce dossier sont exacts et que je suis l'auteur(e) des réalisations jointes.

Fait à **GARNAT** le **10/11/2023**

pour faire valoir ce que de droit.

Signature :



Documents illustrant la pratique professionnelle

(*facultatif*)

Intitulé

Cliquez ici pour taper du texte.

ANNEXES

-le schéma de la Base de données réalisé avec MySqlWorkbench.

-la maquette figma

-Les diagrammes de classes, cas d'utilisation et de séquences.

-Le cahier des charges

-Les Users Stories Admins et Utilisateurs

-Git et Github



CAHIER DES CHARGES



Projet Principal

Lien TRELLO : <https://trello.com/b/jdyXm3Ws/conduite-de-projet>

Lien github côté Front : <https://github.com/Michelhof1978/GarageParrot>

Lien github côté back : <https://github.com/Michelhof1978/GarageBack>

Lien Figma : <https://www.figma.com/file/rhs91pTc1st89ApCbMHoel/GARAGE-PARROT?type=design&node-id=0-1&mode=design&t=b8DACnrXU32mmG6U-0>



Projet Client Les Caravanes De La Besbre : <https://lescaravanesdelabesbre.fr/>

Lien github : <https://github.com/Michelhof1978/lesCaravanesDeLaBesbre>

Technologies Utilisées : Html, Css, Bootstrap, javascript et php sans base de données pour l'instant.



Projet Cv en ligne : <https://cvmichel-hoffmann.fr/>

Lien github : <https://github.com/Michelhof1978/cvMichel>

Technologies Utilisées : Html, Css, Bootstrap, javascript et php sans base de données.

Table de matière

1.0	Description du projet	3
1.1	Connexion	3
1.2	Présentation des services	3
1.3	Définir les horaires d'ouverture	4
1.4	Présentation des voitures d'occasion	4
1.5	Filtrage de la liste des véhicules d'occasion	4
1.6	Contact avec l'atelier	4
1.7	Collecte des témoignages clients	4
1.8	Affichage	5
2.0	Diagrammes	5
3.0	Conception et Architecture	10
4.0	Code référence véhicule	11
5.0	UI	12

Documents joints avec le projet :

- Maquette Figma
- User stories admin + Utilisateurs
- Cahier des charges
- Gestion des tâches avec Trello
- Diagramme de cas d'utilisation
- Diagramme de séquences
- Diagramme de classes avec Mysql

TECHNOLOGIES UTILISEES POUR LE PROJET

- Front : React, Bootstrap, Html et Css
- Back : Php 8, MySqlworkbench et Xampp

1.0 Description du projet :

DESCRIPTION DU PROJET GARAGE PARROT

Le projet vise à développer une application web vitrine pour le **Garage V. Parrot**, un établissement automobile situé à Toulouse, qui propose une gamme de services incluant la réparation automobile, l'entretien et la vente de véhicules d'occasion. L'objectif principal de cette application est de mettre en avant la qualité des services offerts par le garage.

Voici un aperçu des principales fonctionnalités à implémenter :

1.1 Connexion : L'application doit permettre à l'administrateur et aux employés de se connecter via un formulaire d'accès à l'espace professionnel. Les identifiants requis seront le nom d'utilisateur et un mot de passe sécurisé. Seul l'administrateur principal sera autorisé à créer, modifier ou supprimer des comptes pour ses employés. Dans le tableau de bord, toutes les sections seront accessibles à tous les utilisateurs, à l'exception de la section "gestion des comptes employés", qui sera réservée à l'administrateur en chef pour gérer les comptes de ses employés.

1.2 Présentation des services : La page d'accueil de l'application devra clairement présenter les différents services de réparation automobile proposés par le garage, tant dans la barre de navigation que sous forme de diaporama sur la page d'accueil. L'administrateur devra avoir la capacité d'ajouter, de modifier ou de supprimer ces informations depuis son espace d'administration. De plus, seul le compte administrateur pourra consulter l'historique des modifications apportées par les employés, afin de retracer toute manipulation effectuée par un employé en cas de problème.

1.3 Définir les horaires d'ouverture : Les horaires d'ouverture du garage seront clairement affichés sur le site, de préférence dans le pied de page. L'administrateur aura la possibilité de spécifier ces horaires depuis son espace dédié. Les horaires seront affichés avec une coupure entre **12h et 14h** pour indiquer les heures de fermeture pendant cette période.

1.4 Présentation des voitures d'occasion : L'application devra afficher les véhicules disponibles à la vente, accompagnés d'une galerie de photos, de descriptions détaillées et d'informations techniques telles qu'un tableau de caractéristiques, la liste des équipements et options installés. Chaque véhicule devra présenter son prix, une image principale, l'année de mise en circulation et le kilométrage. Les employés auront la possibilité d'ajouter de nouvelles voitures depuis leur espace dédié. Chaque nouvelle voiture ajoutée sera automatiquement mise en avant sur la page d'accueil sous forme de carte, permettant ainsi de présenter les nouveautés aux utilisateurs. Les voitures seront triées automatiquement par ordre d'arrivée.

1.5 Filtrage de la liste des véhicules d'occasion : Pour simplifier la recherche des visiteurs, un système de filtres sera mis en place. Les visiteurs pourront affiner leurs résultats en fonction d'une fourchette de prix (**slider**), d'un nombre de kilomètres parcourus ou de l'année de mise en circulation. Pour offrir la meilleure expérience utilisateur possible, les filtres seront appliqués en temps réel, **sans nécessiter le rechargeement de la page**. Les voitures seront classées en cinq catégories : **Berline, SUV, Familiale, Utilitaire et Citadine**, avec des options de mise en avant sous forme de **boutons** (cases à cocher) pour une expérience utilisateur optimale.

1.6 Contact avec l'atelier : Les visiteurs devront avoir la possibilité de contacter facilement le garage, que ce soit par téléphone ou en remplissant un formulaire de contact qui sera ensuite redirigé vers le service approprié. L'utilisateur devra fournir son **nom, prénom, adresse e-mail, numéro de téléphone et un message**. De plus, une **liste déroulante** permettra de choisir vers quel service l'utilisateur souhaite diriger sa demande. En cas de contact concernant un véhicule en vente, **l'ID ou la référence du véhicule et la fiche du véhicule seront automatiquement inclus** dans le formulaire, accompagnés du message de l'utilisateur. Cela permettra à l'administrateur de retrouver la voiture plus facilement et de répondre de manière plus efficace à la demande de l'utilisateur, ce qui permettra de gagner du temps. L'application intégrera également des mécanismes de conformité au **RGPD** pour la protection des informations personnelles et un **captcha** de Google pour renforcer la sécurité. pour prouver que c'est un humain et éviter que le message tombe dans les spams. Un message s'ouvrira dans une autre page pour confirmation d'envois.

1.7 Collecte des témoignages clients : Une section dédiée sera créée pour permettre aux visiteurs de laisser des témoignages. Ces témoignages devront être soumis à une modération préalable par un employé du garage avant d'être affichés sur la page d'accueil. Les employés auront également la possibilité de laisser des avis client via l'espace Pro. Un système de notation à cinq étoiles sera mis en place, permettant aux utilisateurs de sélectionner leur note, d'ajouter leur nom et de laisser un commentaire. Les informations personnelles ne seront pas affichées publiquement pour préserver la vie privée des utilisateurs. Diffusion des avis sous forme de **carrousel**.

1.8 Affichage :

-**Cookies** (Acceptation **RGPD**, affichage page d'accueil lors de la première connexion)

Principales utilisations des cookies :

- 1. Maintenir la session de l'utilisateur**
- 2. Améliorer l'expérience utilisateur**
- 3. Suivi de l'activité et analyse**
- 4. Publicité ciblée**
- 5. Mesure du trafic**

-**Mentions légales**

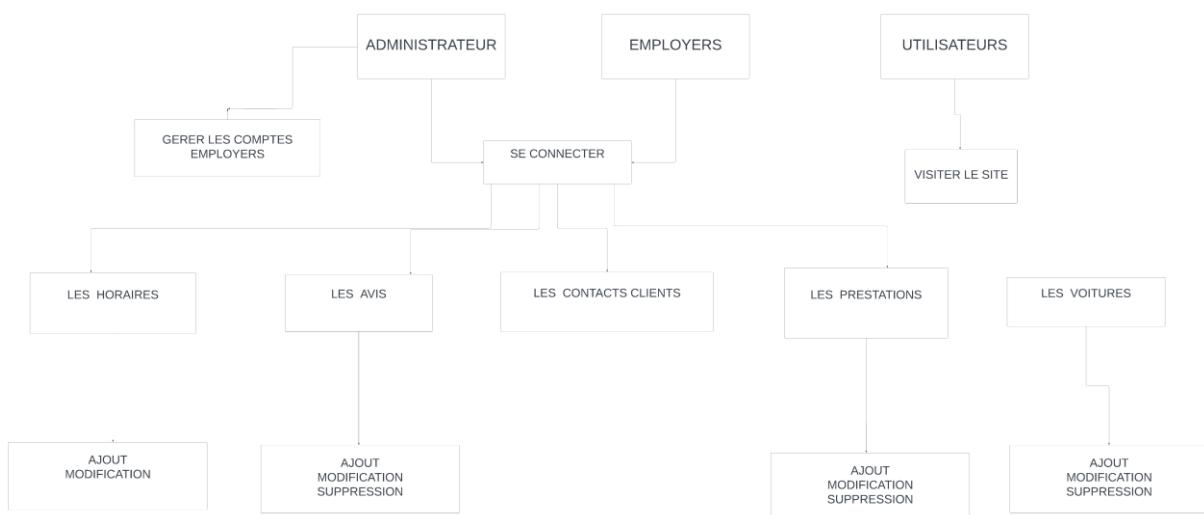
-**Condition générales de ventes**

-**Politique de confidentialité**

2.0 Diagrammes :

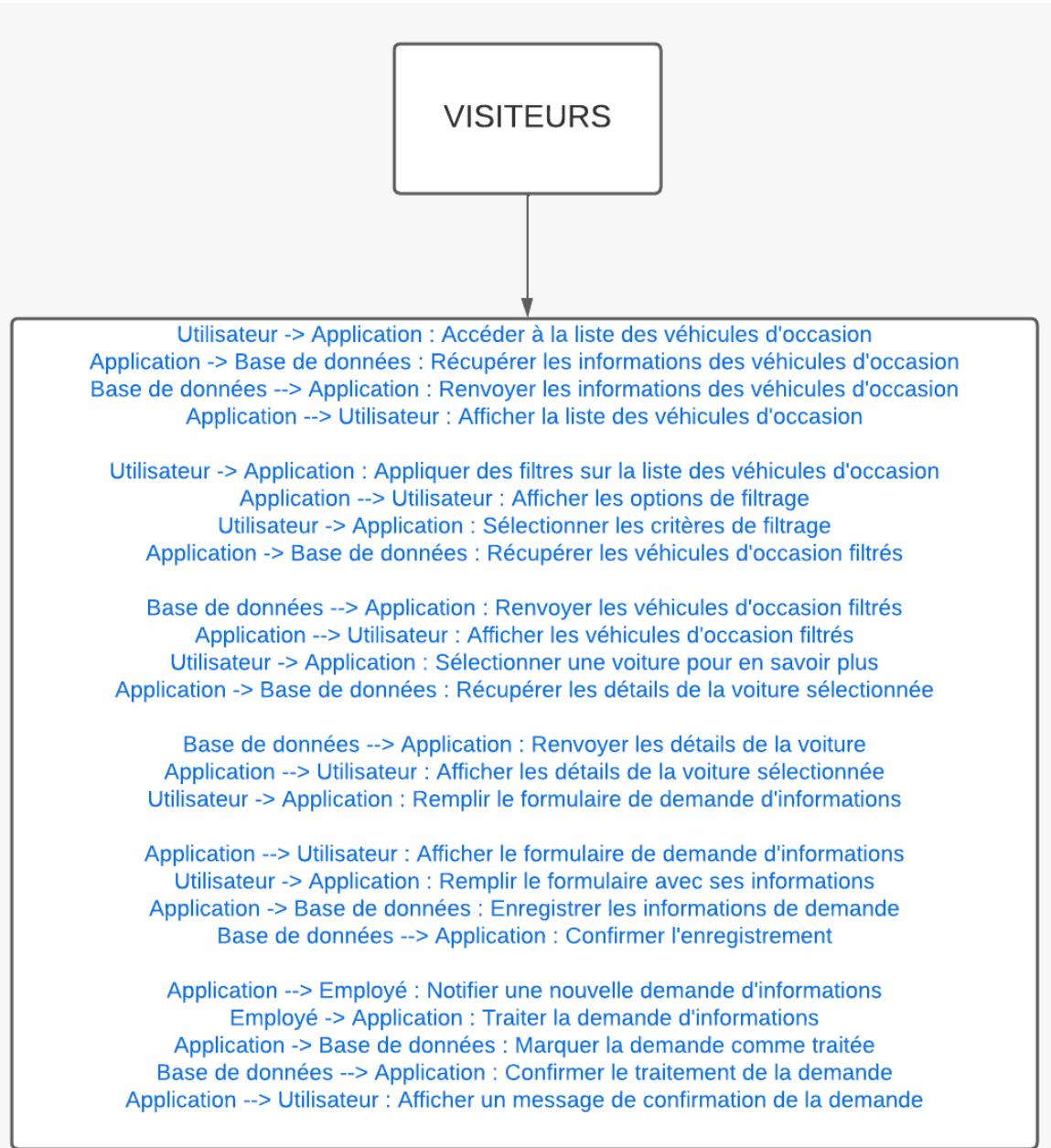
DIAGRAMMES

DIAGRAMME DE CAS D'UTILISATION



DIAGRAMMES DE SEQUENCES

Ces diagrammes de modélisation utilisés représentent les interactions entre différents objets ou acteurs dans l'application. Ils permettent de visualiser la séquence des messages échangés entre ces objets ou acteurs dans le temps. Ils sont particulièrement utiles pour décrire le flux de contrôle entre les différentes parties d'un système et pour comprendre le comportement dynamique d'un processus comme ci-dessous.



Dans ce diagramme de séquence, je décris comment un nouveau visiteur peut découvrir la liste des véhicules d'occasion, appliquer des filtres pour affiner les résultats, sélectionner une voiture spécifique, et remplir le formulaire de demande d'informations pour en savoir plus sur cette voiture. Mon application enregistre ensuite la demande et la notifie à un employé du garage, qui la traitera et la marquera comme traitée dans la base de données..

Administrateur

```
Administrateur -> Application : Se connecter en tant qu'administrateur
Application -> Base de données : Vérifier les identifiants de l'administrateur
Base de données --> Application : Renvoyer les informations d'identification de l'administrateur

Administrateur -> Application : Accéder à l'espace d'administration
Application --> Administrateur : Afficher l'espace d'administration

Administrateur -> Application : Gérer les services de réparation automobile
Application -> Base de données : Récupérer les informations des services de réparation
Base de données --> Application : Renvoyer les informations des services de réparation
Application --> Administrateur : Afficher les services de réparation

Administrateur -> Application : Ajouter un nouveau service de réparation
Application --> Administrateur : Afficher le formulaire d'ajout de service
Administrateur -> Application : Remplir les informations du nouveau service
Application -> Base de données : Enregistrer le nouveau service
Base de données --> Application : Confirmer l'enregistrement du nouveau service
Application --> Administrateur : Afficher un message de confirmation

Administrateur -> Application : Modifier un service de réparation existant
Application --> Administrateur : Afficher le formulaire de modification de service
Administrateur -> Application :Modifier les informations du service
Application -> Base de données : Mettre à jour les informations du service
Base de données --> Application : Confirmer la mise à jour du service
Application --> Administrateur : Afficher un message de confirmation

Administrateur -> Application : Supprimer un service de réparation existant
Application --> Administrateur : Afficher la confirmation de suppression du service
Administrateur -> Application : Confirmer la suppression du service
Application -> Base de données : Supprimer le service de réparation
Base de données --> Application : Confirmer la suppression du service
Application --> Administrateur : Afficher un message de confirmation

Administrateur -> Application : Gérer les horaires d'ouverture
Application -> Base de données : Récupérer les horaires d'ouverture
Base de données --> Application : Renvoyer les horaires d'ouverture
Application --> Administrateur : Afficher les horaires d'ouverture

Administrateur -> Application : Modifier les horaires d'ouverture
Application --> Administrateur : Afficher le formulaire de modification des horaires
Administrateur -> Application : Modifier les horaires d'ouverture
Application -> Base de données : Mettre à jour les horaires d'ouverture
Base de données --> Application : Confirmer la mise à jour des horaires
Application --> Administrateur : Afficher un message de confirmation

Administrateur -> Application : Gérer les témoignages clients
Application -> Base de données : Récupérer les témoignages clients
Base de données --> Application : Renvoyer les témoignages clients
Application --> Administrateur : Afficher les témoignages clients

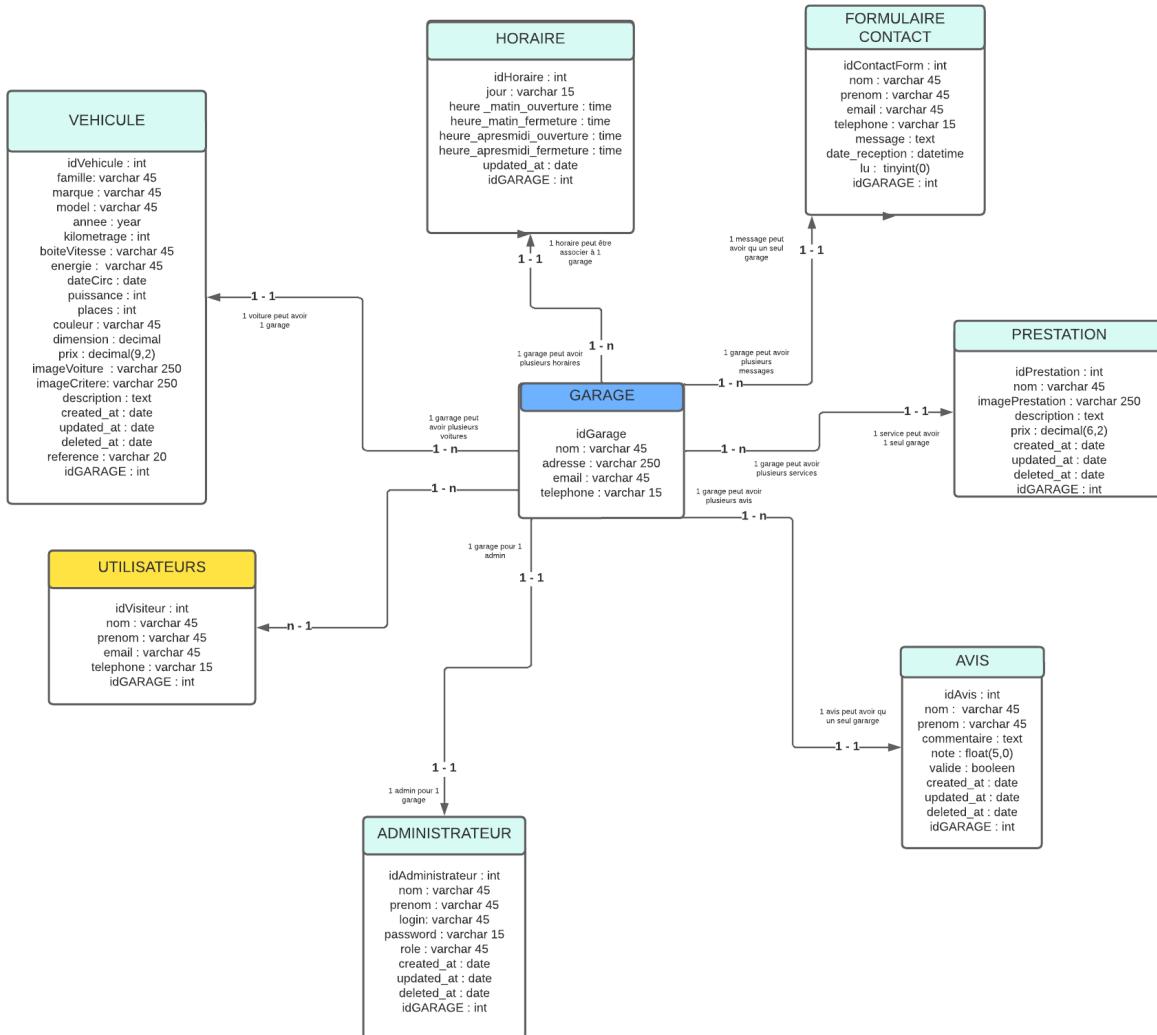
Administrateur -> Application : Modérer un témoignage client
Application --> Administrateur : Afficher le formulaire de modération de témoignage
Administrateur -> Application : Modifier l'état du témoignage (approuvé/rejeté)
Application -> Base de données : Mettre à jour l'état du témoignage
Base de données --> Application : Confirmer la mise à jour de l'état du témoignage
Application --> Administrateur : Afficher un message de confirmation
```

Dans ce diagramme de séquence, je décris comment je peux me connecter à l'application, accéder à l'espace d'administration, gérer les services de réparation automobile (ajouter, modifier, supprimer), gérer les horaires d'ouverture, et modérer les témoignages clients.

Mon application interagit avec la base de données pour récupérer et mettre à jour les informations pertinentes, et elle affiche des messages de confirmation pour m'indiquer le succès des opérations effectuées.

DIAGRAMME DE CLASSES

Ce diagramme de classes illustre la structure de base du modèle de données pour le **Garage V. Parrot**



Dans ce diagramme de classes que je viens de réaliser, le Garage est la classe principale et je lui ai attribué des méthodes pour gérer les **services**, les **horaires**, les **véhicules d'occasion**, les **contacts** avec le service client ainsi que les **avis clients**.

-**La classe Service** représente un service de réparation automobile, et j'ai défini des méthodes pour y accéder et modifier ses attributs.

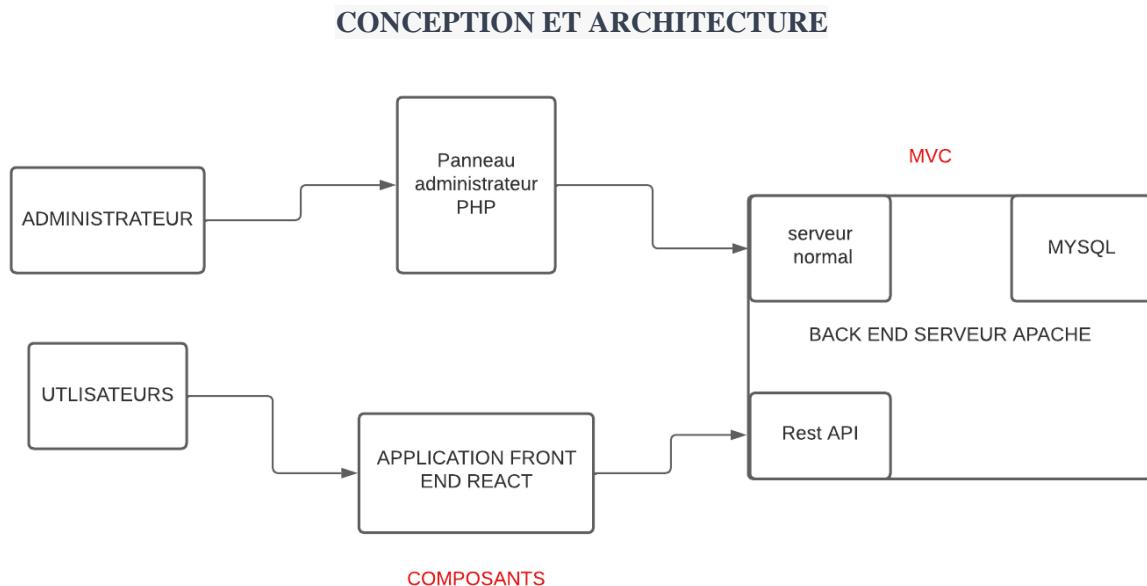
-**La classe Véhicule** représente un véhicule d'occasion, et j'ai également ajouté des méthodes pour accéder et modifier ses attributs.

-**La classe Message** est utilisée pour stocker les informations de contact, telles que les demandes de contact de clients.

-**La classe Horaire** est dédiée à la gestion des horaires d'ouverture du garage, et elle peut avoir des méthodes pour accéder et modifier ces attributs.

-**La classe Avis** est conçue pour recueillir les avis ou les témoignages des clients concernant le garage ou ses services. J'y ai également inclus des méthodes pour accéder et modifier les attributs des avis, ainsi que pour gérer les opérations liées à ces avis.

3.0 Conception et Architecture :



Le projet contiendra 2 parties distinctes :

-La **partie back end** qui sera réalisé en Php et placé sur un serveur Apache avec une base de données mysql/apache.

-Le **panneau administrateur** alimentateur et générera cette base de données en PHP. Toute la partie Serveur utilisera l'architecture modèle du contrôleur et sera programmée en **POO**.

-L'utilisateur utilisera les données de **l'API REST PHP** qui devra être programmé au niveau serveur et qui seront en format **JSON**.

Le serveur devra renvoyer uniquement les données en **Json** et ça sera **React** qui se chargera de les afficher.

4.0 Code référence véhicule :

CODE REFERENCE VEHICULE

Référence id unique par véhicule : exemple = **P-C-208-année**

Code référence véhicule par [marque](#):

Code référence par [famille](#) :

Peugeot = **P**

Utilitaire = **U**

Renault = **R**

Berline = **B**

Citroen = **C**

familiale = **F**

Wolkswagen = **w**

Citadine = **C**

Fiat = **F**

Suv = **S**

Code référence véhicule par marque/model

Peugeot = **208/405/2008/3008/5008**

Renault = **clio/megane/zoe/twingo/kadjar**

Citroen = **C3/C4/C5/berlingo/spacetourer**

Wolkswagen = **golf/polo/passat/tiguan/touareg**

Fiat = **500/panda/ducato/punto/tipo**

5.0 UI :

PALETTE DE COULEURS



#04BBFF



#04BBFF



#04BBFF



#003C57



#003C57

POLICES

- **LEAD** de Bootstrap

- LEAD n'est pas reconnu par Figma, je mettrai en remplacement sur la maquette la police 'Time New Roman'

-Unkempt

User story

GARAGE PARROT



Administrateur
2023

Table de matière

1.0	Création de compte.....	3
1.1	Se connecter.....	3
1.2	Se déconnecter.....	3
2.0	Page d'accueil et prestations.....	4
3.0	Page fiche voiture	4
4.0	Formulaire de contact (navbar + prestations).....	5
4.1	Formulaire de contact (fiche voiture).....	5
5.0	Dashboard - Ajout, modifier, supprimer Page d'accueil et Fiche Prestations.....	5
5.1	Dashboard - Ajout, modifier, supprimer Fiche Voiture	6
5.2	Dashboard - Ajout, supprimer Avis clients.....	6
5.3	Dashboard - Ajout, modifier, supprimer Horaires.....	7

Préambule

Ce document décrit les fonctionnalités du rôle pour chaque administrateur de notre plateforme web et mobile.

CRUD (**c**reate, **r**ead, **u**pdate, **d**elete) (créer, lire, mettre à jour, supprimer) est un acronyme pour les façons dont on peut fonctionner sur des données stockées. C'est un moyen némotechnique pour les quatre fonctions de base du stockage persistant.

1.0 Crédation de compte

CREATION DE COMPTE	
Utilisateurs cibles	ADMINISTRATEUR
Description	En tant qu'administrateur, je souhaite pouvoir créer ou supprimer un espace admin pour mes employés.
Règles de gestion	-uniquement l'admin pourra le faire -un mot de passe aléatoire sera créer -à voir si l'employé pourra modifier son mdp -pouvoir donner un rôle à chaque admin (autorisation ou pas) -l'admin pourra ensuite modifier son mdp par sécurité en 2 étapes
Critères d'acceptation (conditions)	Etant donné que je suis le seul à pouvoir prendre les décisions à gérer le contenu du site, je déleste en parti mon travaille à certains de mes employés en leur créant un espace unique pour chaque personne

1.1 Se connecter

SE CONNECTER	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir gérer le site en me connectant sur l'espace pro.
Règles de gestion	-avoir un compte admin -email et mdp obligatoire

	<ul style="list-style-type: none"> -modification de mot de passe affiché sous la connexion, un lien sera envoyé pour réinitialiser le mdp -création d'un journal des connexions journalière date + heure dans l'espace Admin uniquement -création d'un journal de toutes les tâches effectuées par les employés dans l'espace Admin uniquement
Critères d'acceptation (conditions)	Etant donné que je dois gérer le contenu du site, j'ai le moyen de le faire en me connectant sur la plateforme pro

1.2 Déconnexion

DECONNEXION	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir me déconnecter à tout moment par sécurité.
Règles de gestion	<ul style="list-style-type: none"> -avoir un compte admin -se connecter sera remplacé par déconnexion lorsqu'il sera connecté -déconnexion automatique après 5mn d'inactivité
Critères d'acceptation (conditions)	Etant donné que j'ai terminé ce que je voulais faire ds l'espace pro, j'ai le moyen de me déconnecter à tout moment par sécurité.

1.3 Règles de gestion du Mot de passe

REGLES DE GESTION DU MOT DE PASSE	
Utilisateurs cibles	ADMINISTRATEUR
Description	En tant qu'administrateur, je souhaite pouvoir modifier mon mdp si j'en ai besoins
Règles de gestion	<ul style="list-style-type: none"> -être admin -mdp associé au mail -réinitialisation du mdp et valider
Critères d'acceptation (Conditions)	Etant donné que j'ai oublié mon mdp, je clique sur le lien mdp oublié qui est sur la page connexion, j'inscris mon mail pour pouvoir recevoir une réinitialisation de mdp au compte associé au mail

4.0 Formulaire de contact (navbar + prestations)

FORMULAIRE DE CONTACT (navbar + prestations)	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir répondre aux questions des utilisateurs
Règles de gestion	-avoir un compte admin -pouvoir répondre aux utilisateurs -pouvoir transférer le message au service concerné -avoir un historique de la conversation -notification de modification, supprimer ou envoyé
Critères d'acceptation (conditions)	Etant donné que je dois fournir un service de communication pour la satisfaction du client, je dois pouvoir lui répondre ds les plus brefs délais.

4.1 Formulaire de contact (fiche voiture)

FORMULAIRE DE CONTACT (fiche voiture)	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir ajouter, modifier ou supprimer certains contenus comme des images, textes
Règles de gestion	-avoir un compte admin - L'ID ou référence du véhicule accompagné du message de l'utilisateur devront être afficher pour une meilleure expérience -notification de modification, supprimer ou envoyé
Critères d'acceptation (conditions)	Etant donné que je dois fournir un service de communication pour la satisfaction du client, je dois pouvoir lui répondre ds les plus brefs délais.

5.0 Dashboard - Page d'accueil et prestations

DASHBOARD - PAGE D'ACCUEIL ET PRESTATIONS	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir ajouter, modifier ou supprimer certains contenus comme images, textes...
Règles de gestion	<ul style="list-style-type: none"> - avoir un compte admin - Section 1-> photo modifiable - Section 2-> photo modifiable - Section 3-> photos + texte modifiable - Section 4-> photos modifiables + texte modifiable - Section 5-> photo modifiable - Section 6-> + texte modifiable <p>-notification de modification, supprimer ou création</p> <p>- historique des modifications pour chaque employé.</p>
Critères d'acceptation (conditions)	Etant donné que je dois gérer le contenu de mon site, j'ai le moyen de le faire en tant qu'administrateur dans le Dashboard dédié.

5.1 Dashboard - Page Fiche Voiture

DASHBOARD - PAGE FICHE VOITURE	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir ajouter, modifier ou supprimer certains contenus comme des images, textes.
Règles de gestion	-avoir un compte admin - Section 4 -> photos modifiables + texte modifiable en ajoutant les caractéristiques de chaque véhicule. -notification de modification, supprimer ou création - historique des modifications pour chaque employé.
Critères d'acceptation (conditions)	Etant donné que je dois gérer le contenu de mon site, j'ai le moyen de le faire en tant qu'administrateur dans le Dashboard dédié

5.2 Dashboard - Avis Clients

DASHBOARD – AVIS CLIENTS	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir gérer les avis clients et pouvoir aussi les publier en ligne. J'aurais le choix ou pas de les publier.
Règles de gestion	-avoir un compte admin -ajout, supprimer avis client en ligne -possibilité d'ajouter avis client écrit sur papier -répondre aux avis -les infos perso ne devront pas être publié en ligne, uniquement le prénom
Critères d'acceptation (conditions)	Etant donné que je dois attirer un maximum de clientèle, je peux exposer sur le site quelques avis clients.

5.3 Dashboard - Horaires

DASHBOARD - HORAIRES	
Utilisateurs cibles	ADMINISTRATEUR ET EMPLOYES
Description	En tant qu'administrateur, je souhaite pouvoir modifier les horaires d'ouverture de l'entreprise.
Règles de gestion	-avoir un compte admin -affichage dans footer -notification de modification, supprimer ou création -ajout, modifié, supprimer horaires -heure, fermé, vacances -coupure entre 12h et 14h
Critères d'acceptation (conditions)	Etant donné que je dois informer la clientèle des horaires d'ouverture du garage, j'ai le moyen de les informer par le site.

User story

GARAGE PARROT



Utilisateurs
2023

Table des matières

1.0	Header	3
1.1	Header - Navbar.....	3
2.0	Formulaire de contact	3
3.0	Footer.....	4
4.0	Page d'accueil.....	4
4.1	Page d'accueil - Laisser un avis.....	4
4.3	Page d'accueil - Lire un avis.....	5
4.4	Page d'accueil - Cookies.....	5
5.0	Page Prestations.....	5
5.1	Page Prestations - Contact.....	6
6.0	Page Voitures d'Occasion	6
6.1	Page Voitures d'Occasion - Filtres.....	6
7.0	Page Fiche Voiture.....	6
7.1	Page Fiche Voiture - Contact.....	7

Préambule

Ce document décrit les fonctionnalités par rôle pour chaque utilisateur ou système de notre plateforme web et mobile.

Voici les rôles :

Les utilisateurs bénéficient de tous les services que propose le site sans devoir être authentifiés.

EXEMPLE A VIDE :

TITRE	
Utilisateurs cibles	Utilisateurs et/ou Anonymes
Description	En tant que..... , Je souhaite...., afin de(facultatif)....
Règles de gestion	-fonctionnalités
Critères d'acceptation (conditions)	Etant donné que....(situation de départ),..., que ..., Quand....(action) et que.... Alors....(resultat)

1.0 Header

HEADER	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'Utilisateur, j'ai accès en premier plan à des offres ou promotions sous forme de bannière.
Règles de gestion	
Critères d'acceptation (conditions)	Etant donné que je suis sur le site pour un sujet bien précis, je peux éventuellement profiter d'offres qui pourrait être décisif dans mes futurs choix.

1.1 Header - Navbar

HEADER - NAVBAR	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'Utilisateur, je veux pouvoir accéder facilement aux différentes sections du site depuis la navbar, afin de naviguer rapidement vers les pages qui m'intéressent.
Règles de gestion	-cliquer sur le service désiré -menu Accueil/logo)/Pneumatique/ Mécanique/Contact.....
Critères d'acceptation (conditions)	Etant donné que j'aimerais naviguer facilement sur le site, j'ai le moyen de le faire en cliquant sur le menu qui m'est proposé

2.0 Formulaire De Contact

FORMULAIRE DE CONTACT	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je veux pouvoir contacter le service clientèle via un formulaire de contact pour poser des questions ou signaler un problème.
Règles de gestion	-remplir tous les champs du formulaire -email : invalide si pas @, .fr .com -numéro de téléphone : minimum 10 chiffres -nom, prénom : minimum 3 lettres -prouver que je suis bien un humain en cochant la checkbox captcha
Critères d'acceptation (conditions)	Etant donné que je souhaite un renseignement sur un produit ou service, prendre rdv ou demander un devis, je le fais avec le formulaire de contact qui m'est proposé par le site.

3.0 Footer (pied de page)

FOOTER	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, j'aimerais suivre via les réseaux sociaux l'entreprise et connaître les horaires d'ouverture. Je suis aussi avisé des règles légales de l'entreprise.
Règles de gestion	-Doit avoir une page les mentions légales -Doit avoir une page de confidentialité -Doit avoir condition générale de vente -Doit avoir réseaux sociaux -Doit avoir les horaires d'ouverture -Doit avoir les coordonnées du site pour avoir meilleur référencement
Critères d'acceptation (conditions)	Etant donné que je souhaite me renseigner un peu plus sur l'entreprise et connaître mes droits en tant qu'utilisateur, je peux y accéder directement via le footer

4.0 Page d'accueil

PAGE D'ACCUEIL	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je veux voir une interface claire et attrayante sur la page d'accueil pour faciliter la navigation et susciter mon intérêt dès le premier regard.
Règles de gestion	
Critères d'acceptation (conditions)	-Etant donnée que je cherche à consommer, la page d'accueil influencera mes choix et d'y rester ou pas.

4.1 Page d'accueil – Laisser un avis

PAGE D'ACCUEIL- LAISSER UN AVIS	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je souhaite pouvoir laisser des commentaires et des évaluations sur la qualité des services de l'entreprise pour aider les autres utilisateurs dans leur décision d'achat.
Règles de gestion	-remplir tous les champs du formulaire -email : invalide si pas @, .fr .com -numéro de téléphone : minimum 10 chiffres -nom, prénom : minimum 3 lettres -prouver que je suis bien un humain en cochant la checkbox captcha
Critères d'acceptation (conditions)	Etant donné que j'ai été client du site, je laisse un avis pour aider les futurs clients dans leurs choix

4.3 Page d'accueil – Lire un avis

PAGE D'ACCUEIL – LIRE UN AVIS	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'Utilisateur, je souhaite pouvoir lire les avis d'autres clients pour savoir si l'entreprise est sérieuse ou pas.
Règles de gestion	-Défilement des avis par flèches gauche ou droite.
Critères d'acceptation (conditions)	Etant donné que je souhaite acheter un service ou produit, je me renseigne déjà sur les avis clients pour savoir si l'entreprise est honnête ou pas.

4.4 Page d'accueil - Cookies

PAGE D'ACCUEIL - COOKIES	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, j'ai le choix d'accepter ou pas que l'on utilise mes informations pour une meilleure navigation dans le futur.
Règles de gestion	-Affichage : Accepter ou refuser les cookies lors de la connexion sur le site
Critères d'acceptation (conditions)	Etant donné que je souhaite me connecter au site, j'ai l'obligation d'accepter ou pas les cookies pour pouvoir faire disparaître le popup et pouvoir visiter le site

5.0 Page Prestations

PAGE – PRESTATIONS	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je souhaite pouvoir parcourir facilement les différentes prestations offertes par l'entreprises afin de trouver celles qui correspondent le mieux à mes besoins.
Règles de gestion	-Aucunes
Critères d'acceptation (conditions)	Etant donné que j'ai un but précis dans mes recherches, je vais directement et rapidement au service voulu.

5.1 Page Prestations - Contact

PAGE PRESTATIONS - CONTACT	
Utilisateurs cibles	Utilisateurs

Description	En tant qu'utilisateur, je veux pouvoir contacter le service clientèle via un formulaire de contact pour avoir des renseignements sur une prestation et obtenir un devis
Règles de gestion	-remplir tous les champs du formulaire -email : invalide si pas @, .fr .com -numéro de téléphone : minimum 10 chiffres -nom, prénom : minimum 3 lettres -prouver que je suis bien un humain en cochant la checkbox captcha
Critères d'acceptation (conditions)	Etant donné que je souhaite un renseignement ou un devis sur un produit ou service, prendre rdv, je le fais via le formulaire de contact qui m'est proposé par le site.

6.0 Page Voitures d'Occasion

PAGE VOITURES D'OCCASION	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je souhaite me renseigner sur les véhicules que propose le site pour un éventuel achat.
Règles de gestion	-Aucunes
Critères d'acceptation (conditions)	Etant donné que je souhaite acheter une voiture, j'ai le choix de faire ma sélection par rapport au catalogue qui m'est proposé.

6.1 Page Voitures d'Occasion - Filtres

PAGE VOITURES D'OCCASION - FILTRES	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur et ayant un but précis, je souhaite pouvoir trouver des véhicules plus facilement via le système de filtres.
Règles de gestion	-choix par famille -choix par kilométrage, marque, modèle, prix et année
Critères d'acceptation (conditions)	Etant donné que je souhaite acheter un véhicule, j'ai le moyen de le faire grâce aux filtres pour une recherche plus rapide.

7.0 Page Fiche Voiture

PAGE FICHE VOITURE	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je peux voir toutes les caractéristiques de chaque véhicules
Règles de gestion	-Aucunes

Critères d'acceptation (conditions)	Etant donné que je souhaite acheter un véhicule et que je sais quel genre de véhicule je veux, j'ai le moyen de le faire grâce à la fiche technique du véhicule.
--	--

7.1 Page Fiche Voiture - Contact

PAGE FICHE VOITURE - CONTACT	
Utilisateurs cibles	Utilisateurs
Description	En tant qu'utilisateur, je peux demander des renseignements complémentaires sur le véhicule, le réserver ou obtenir un rdv pour le voir.
Règles de gestion	-remplir tous les champs du formulaire -email : invalide si pas @, .fr .com -numéro de téléphone : minimum 10 chiffres -nom, prénom : minimum 3 lettres -prouver que je suis bien un humain en cochant la checkbox captcha -affichage automatique de l'Id du véhicule ds le formulaire pour que l'admin puisse trouver le véhicule plus rapidement.
Critères d'acceptation (conditions)	Etant donné que je souhaite me renseigner de ce que propose le site et savoir si c'est adapté à mes exigences, je vais me renseigner sur la page en question.

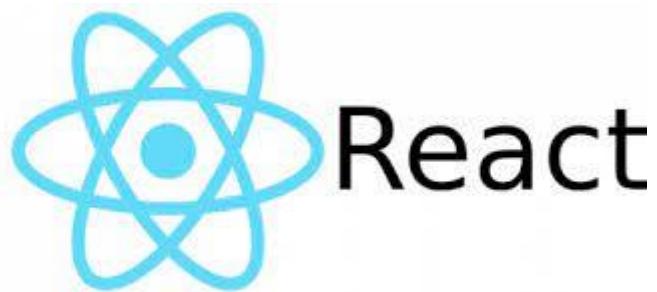
PROJET



Made in



PARTIE FRONT END



Ce rapport décrit les principales étapes et réalisations effectuées lors du développement Front-End, en mettant particulièrement l'accent sur la mise en place de l'environnement de travail, l'utilisation de React, la gestion des dossiers, et l'intégration de Bootstrap via les CDN.



Table de matière

1.0 Mise en Place de l'Environnement de Travail.....	3
1.1 React Router	3
1.2 Gestion des Dossiers et Problèmes d'Importation	3
1.3 Intégration de Bootstrap via les CDN.....	3
1.4 Création de Composants SF et SL	4
1.5 Utilisation d'Axios pour les Requêtes	4
1.6 Optimisation du Flux de Travail avec des Snippets	4
2.0 Création du Composants	5
2.1 Création du Composant Navbar avec Bootstrap	5
2.2 Création du composant de la Page d'Erreur 404	6
2.3 Création du Composant Card pour les Véhicules	6
2.4 Mise en Place de la Pagination des cartes	9
2.5 Création des composants UI	11
3.0 Page D'accueil	11
4.0 Page de contact	12
4.1 Ajout de Google reCAPTCHA	13
5.0 Page voiture d'occasion	14
5.1 Composant BasicRange.....	15
5.2 Composant BasicSelect	17
5.3 Composant BasicCheckbox	19
5.4 Composant VehiculeFilters.....	19
5.5 Hook UseEffect	20
5.6 Fonction handleClick	21
6.0 Postman.....	21
7.0 Prestations	23
8.0 Avis Clients	23
9.0 Conclusion	25

1.0 Mise en Place de l'Environnement de Travail

L'une des premières tâches du projet a été la mise en place de l'environnement de travail.

Mon choix s'est porté sur la bibliothèque JavaScript **React** pour le développement des interfaces utilisateur.

J'ai configuré les outils et dépendances nécessaires pour pouvoir démarrer mon projet.

1.1 React Router

React Router a été installé à l'aide de la commande `npm install react-router-dom`.

Une fois installé, J'ai importé dans le fichier `App.js` pour être utilisé dans la gestion des routes de l'application.

Le composant `Site` a été configuré pour contenir toutes les routes de l'application.

1.2 Gestion des Dossiers et Problèmes d'Importation

Une des premières difficultés rencontrées a été liée à la gestion des dossiers et à l'importation de composants.

J'ai observé que React ne reconnaissait pas automatiquement les majuscules dans les noms de dossiers, obligeant à les saisir manuellement.

Ce problème a également été rencontré précédemment sur un autre de mes projets, il doit s'agir d'un problème de React.

1.3 Intégration de Bootstrap via les CDN

Pour améliorer l'aspect visuel de l'application, Bootstrap a été intégré en utilisant des CDN (Content Delivery Network), par la suite, je l'ai installé via le terminal par rapport à React.

Cette approche a permis d'inclure rapidement Bootstrap dans le projet sans avoir à télécharger et à configurer les fichiers localement.

1.4 Crédit de Composants SF et SL

J'ai créé 2 types de composants à l'avance pour pouvoir gagner un peu plus de temps :ce sont les composants **Stateless** et les composants **Stateful**.

Les composants SF sont des composants React simples qui ne gèrent pas d'état interne.

Les composants SL, en revanche, gèrent un état interne et sont utilisés pour des éléments interactifs de l'interface utilisateur.

1.5 Utilisation d'Axios pour les Requêtes

J'ai installé Axios, une bibliothèque JavaScript populaire pour effectuer des requêtes HTTP, à l'aide de la commande `npm install axios`.

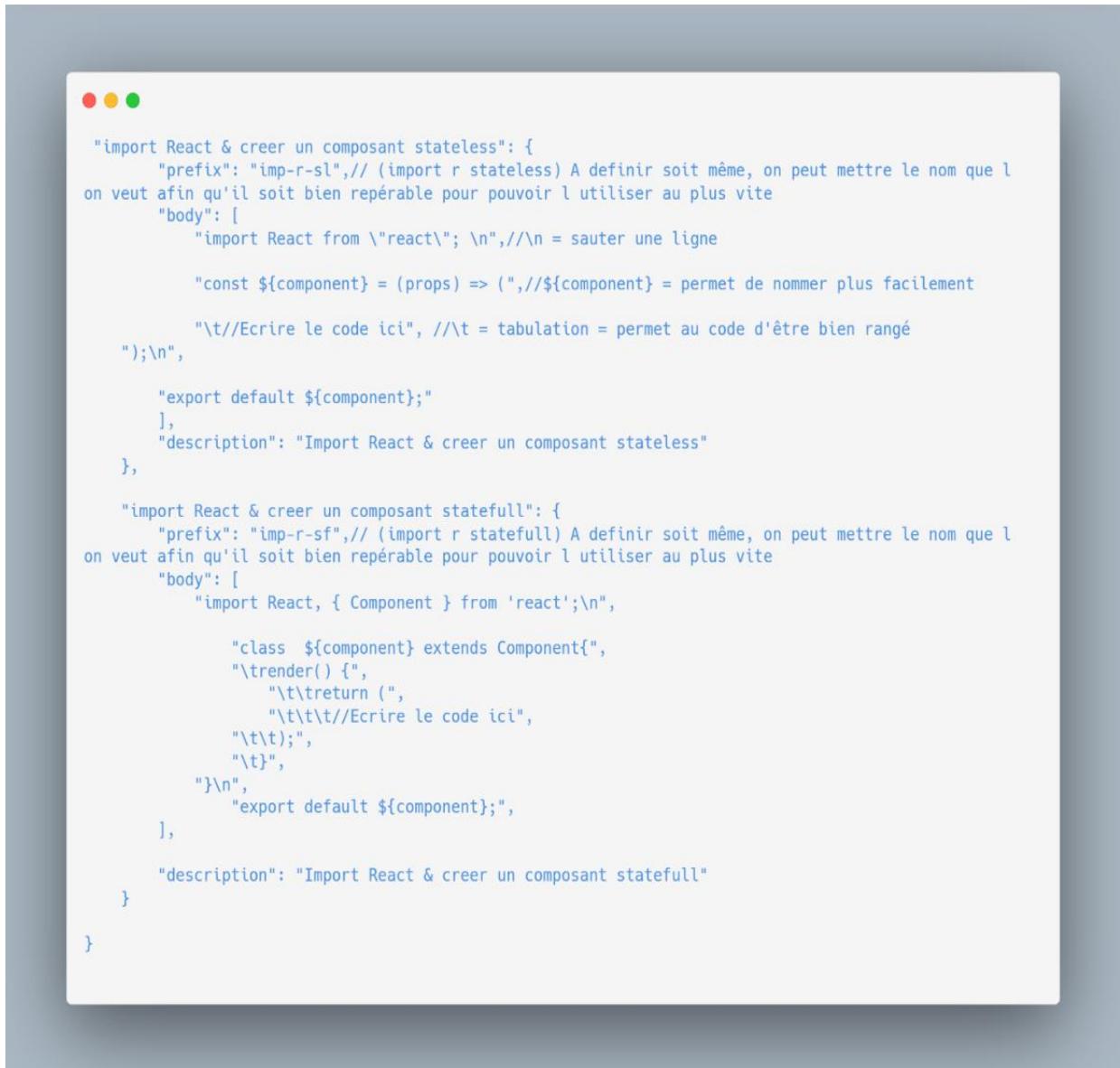
Axios sera utilisé pour effectuer des requêtes vers le backend ou d'autres services par la suite.

1.6 Optimisation du Flux de Travail avec des Snippets

Pour faciliter le développement, j'ai créé dans Visual Studio Code des extraits de code (**snippets**).

Ces extraits permettent de générer automatiquement la structure de base des composants **SF** et **SL**, accélérant ainsi le processus de création de composants.

Pour utiliser ces extraits, Je suis allé dans le menu "Fichier" (File), puis "Préférences" (Préférences), et enfin "Extraits d'utilisateur" (User Snippets), puis de sélectionner le langage (dans ce cas là, JavaScript).



The screenshot shows a Visual Studio Code window with a dark theme. A floating JSON file is open, titled 'User Snippets'. The content of the file is as follows:

```
"import React & creer un composant stateless": {  
    "prefix": "imp-r-sl", // (import r stateless) A definir soit même, on peut mettre le nom que l'on veut afin qu'il soit bien repérable pour pouvoir l'utiliser au plus vite  
    "body": [  
        "import React from \"react\"; \n", // \n = sauter une ligne  
  
        "const ${component} = (props) => (", // ${component} = permet de nommer plus facilement  
  
        "\t//Ecrire le code ici", // \t = tabulation = permet au code d'être bien rangé  
    ");\n",  
  
        "export default ${component};"  
    ],  
    "description": "Import React & creer un composant stateless"  
},  
  
"import React & creer un composant statefull": {  
    "prefix": "imp-r-sf", // (import r statefull) A definir soit même, on peut mettre le nom que l'on veut afin qu'il soit bien repérable pour pouvoir l'utiliser au plus vite  
    "body": [  

```

Je détaille ici les activités effectuées au cours de la phase de développement côté Front. Dans cette section, je me pencherai sur les éléments suivants :

la conception du composant Navbar en utilisant Bootstrap, la création d'une page d'erreur 404, l'intégration d'Axios pour gérer les requêtes, la mise en place du composant de cartes pour afficher les véhicules, l'implémentation d'un système de pagination pour les cartes, ainsi que le développement d'un système de recherche de véhicules basé sur des filtres....

2.0 Crédation des composants

2.1 Crédation du Composant Navbar avec Bootstrap

J'ai créé un composant **Stateless** nommé Navbar.js pour pouvoir afficher la barre de navigation de l'application.

-Mode desktop



-Mode Mobile (Hamburger)



J'ai inséré dans le composant des images et du code **JSX**, avec des classes `className` pour pouvoir mettre du style en majorité Bootstrap.

J'ai présenté les services de réparation sous forme de menus déroulants (**dropdowns**) dans la barre de navigation.

Le tout en responsive bien sûr, un menu de type ‘hamburger’ sera affiché en mode mobile.

2.2 Crédation du composant de la Page d'Erreur 404

J'ai ajouté une route de type Error 404 pour pouvoir gérer les cas où une page n'existe pas.

Cette page d'erreur fournira une réponse aux utilisateurs en cas d'accès à des pages inexistantes.

```

const error = (props) => (
  <div style={{ backgroundColor: "#FF0000", color: "#FFFFFF", padding: "20px"
}}> <TitreH1>Erreur{props.type}</TitreH1>
  <div>
    {props.children}
  </div>
</div>
)
export default error

```

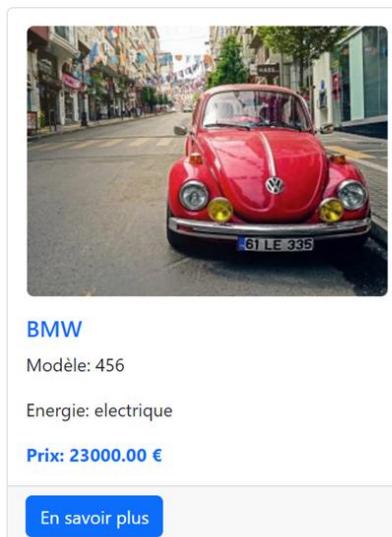
2.3 Création du Composant Card pour afficher les Véhicules

J'ai voulu présenter les véhicules aux utilisateurs sous forme de cartes Bootstrap, mettant en avant quelques informations essentielles concernant chaque véhicule.

Composant 'Card'

Pour réaliser cela, j'ai connecté ce composant directement à la base de données, extrayant les données de la table 'véhicule' à l'aide d'Axios.

Cette approche me permet de récupérer toutes les informations relatives à chaque véhicule. Le composant 'Card' a été conçu pour être réutilisable, affichant de manière individualisée les détails d'un véhicule à la fois, sous forme d'une carte distincte.





```
const Card = (props) => {
  const image = `http://localhost/garageback/public/images/${props.image}`;
  return (
    <>
      <div className="card">
        <div className="card-body">
          <a
            href={props.image}
            target="_blank"
            rel="noopener noreferrer"
          >
            <img src={image} alt={props.marque} className="img-fluid rounded mx-auto d-block mb-3"/> </a>

          <h5 className="card-title text-primary">{props.marque.toUpperCase()}</h5>
          <p className="card-text">Modèle: {props.modele}</p>
          <p className="card-text">Energie: {props.energie}</p>
          <p className="card-text fw-bold text-primary">Prix: {props.prix} €</p>
        </div>
        <div className="card-footer">
          <Link
            to={`/vehiculefiche/${props.id}`}
            className="btn btn-primary"
          >
            En savoir plus
          </Link>
        </div>
      </div>
    );
};

export default Card;
```

Je vais ajouter des paramètres aux propriétés du composant '**Card**' pour représenter les informations du véhicule, telles que l'image, la marque, le modèle, etc.

Pour obtenir l'image, je vais construire le chemin en utilisant l'URL de base, indiquant où sont stockées les images. Ensuite, je vais transmettre cette URL via la propriété `props.image`, ce qui me permettra d'éviter de réécrire l'URL chaque fois que j'aurai besoin de récupérer une image .

Composant 'vehiculeCard'

Pour ce composant, je vais importer plusieurs dépendances, notamment `useState`, qui me permettra de gérer l'état local pour stocker la liste des véhicules, le numéro de la page actuelle `currentPage`, et le nombre de cartes affichées par page que je vais définir ultérieurement `cardsPerPage`.



```
const VehiculesCard = () => {
  const [vehicules, setVehicules] = useState([]);
  const [currentPage, setCurrentPage] = useState(1);
  const cardsPerPage = 6;

  useEffect(() => {
    axios
      .get("http://localhost/garageback/front/voiturefiche/all")
      .then((response) => {
        const jsonData = response.data;
        const sortedVehicules = [...jsonData];
        const sortByCreatedAt = (a, b) => {
          const dateA = new Date(a.created_at).getTime();
          const dateB = new Date(b.created_at).getTime();
          return dateA - dateB;
        };
        sortedVehicules.sort(sortByCreatedAt);
        setVehicules(sortedVehicules);
      })
      .catch((error) => {
        console.error("Erreur lors de la récupération des véhicules :",
          error));
  });
}
```

Mon objectif est de permettre au composant d'effectuer une requête à partir de l'**API** 'véhicule' que j'ai créée du côté serveur, afin d'obtenir la liste de tous les véhicules à afficher.

Pour cela, je vais utiliser '**axios**' dans la fonction '**useEffect**'. Mon intention est de faire en sorte que les cartes s'affichent sur la page d'accueil en fonction de leur date de création la plus récente.

Cela permettra de renouveler régulièrement les cartes et de proposer aux utilisateurs les derniers véhicules en stock.

Le composant va ensuite mapper les véhicules actuellement affichés dans une liste de composants '**Card**' créés précédemment, en transmettant les informations du véhicule en tant que propriétés à chaque composant 'Card'.

```
● ○ ●

<div className="row">
  {currentCards.map((vehicule) => (
    <div
      key={vehicule.idVehicule}
      className="col-lg-4 col-md-4 col-sm-6 col-6 mt-3"
    >
      <Card
        image={vehicule.imageVoiture}
        marque={vehicule.marque}
        nom={vehicule.nom}
        modele={vehicule.modele}
        energie={vehicule.energie}
        prix={vehicule.prix}
        id={vehicule.idVehicule}
      />
    </div>
```

2.4 Mise en Place de la Pagination des cartes

Je vais mettre en œuvre un système de pagination pour diviser les véhicules en groupes, limitant ainsi le nombre de véhicules affichés par page grâce à la variable '**cardsPerPage**'.

Ce mécanisme permettra aux utilisateurs de naviguer d'une page à l'autre.

Le nombre total de pages sera calculé en fonction du nombre total de véhicules et du nombre de cartes par page, une valeur que je vais définir ultérieurement.



```
const indexOfLastCard = currentPage * cardsPerPage;
const indexOfFirstCard = indexOfLastCard - cardsPerPage;
const currentCards = vehicules.slice(indexOfFirstCard, indexOfLastCard);

const paginate = (pageNumber) => {
    setCurrentPage(pageNumber);
};

console.log(currentCards);
console.error("Erreur lors de la récupération des véhicules :",
error});
```



```
<Pagination>
  {Array.from(
    { length: Math.ceil(vehicules.length / cardsPerPage) },
    (_, index) => (
      <Pagination.Item
        key={index}
        active={index + 1 === currentPage}
        onClick={() => paginate(index + 1)}
      >
        {index + 1}
      </Pagination.Item>
    )
  )}
</Pagination>
```

Le nombre maximal de cartes affichées sera de 9 véhicules, ce qui correspond à 3 cartes par ligne en mode desktop et 2 en mode mobile, tant sur la page d'accueil que sur la page de voitures d'occasion résultant d'une recherche par filtres ou d'un classement par ordre de date de création.

Cette limitation permettra d'améliorer significativement les performances de l'application et éviter que tous les véhicules de la base de données s'affichent en même temps.

Affichage des informations complète du véhicule

Lorsque l'utilisateur clique sur "plus d'infos" sur la carte, une nouvelle page s'affichera, présentant toutes les informations du véhicule de manière plus détaillée, toujours sous forme de carte, mais de manière plus large.

Si l'utilisateur souhaite obtenir davantage d'informations sur un produit, je veillerai à le rediriger vers la page de contact.

Dans une étape future, que je n'ai pas encore eu l'occasion de mettre en place, je prévois d'automatiquement inclure l'ID du véhicule dans le formulaire de contact. Cela permettra à l'administrateur de retrouver plus facilement le véhicule en se référant à son identifiant, ce qui entraînera un gain de temps considérable pour lui.



2 .5 Crédation des composants UI

-Création de tous les composant boutons, cards

3.0 Page d'Accueil

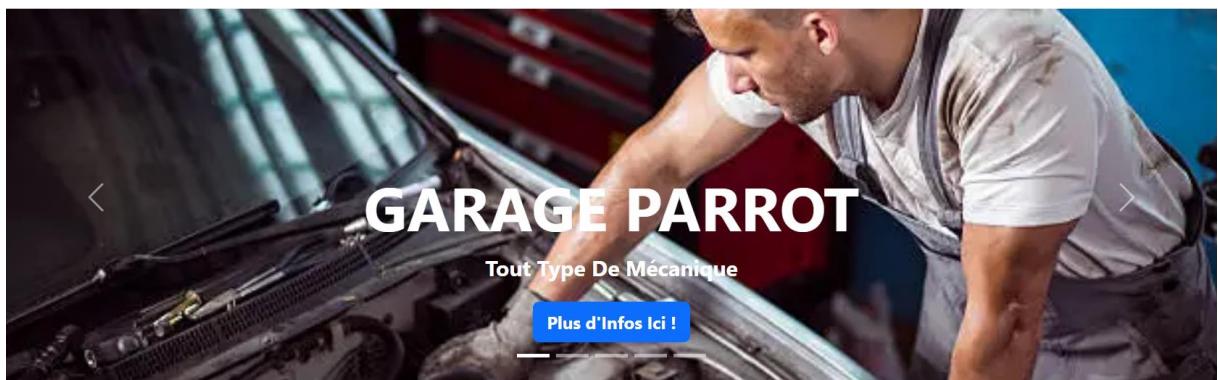
- La page d'accueil a été mise en place avec l'ajout d'un composant `h1` pour afficher le titre.
- J'ai ajouté le composant **Navbar**.
- Pour améliorer la présentation, j'ai installé la bibliothèque **Font Awesome** à l'aide de la commande `npm install --save font-awesome`.

Composant carroussel

J'ai intégré le composant carrousel pour présenter les services que le garage propose.

Lorsque l'utilisateur cliquera sur le bouton du carrousel, il sera redirigé vers une autre route que j'aurai créée au préalable.

Cette nouvelle route affichera une présentation des prestations du garage, également sous forme de cartes.



J'ai ajouté le composant **cards** pour pouvoir présenter à l'utilisateur les derniers véhicules en stock arrivés .

4.0 Page de Contact

Je viens d'ajouter une route vers le formulaire de contact qui a été créée dans le fichier `Site.js`, avec son composant associé.

Nous écrire

Prénom

Nom

Téléphone

@ Email

Objet :

Message

J'ai développé la view du formulaire de la page de contact sous bootstrap dans un conteneaire.

J'ai utilisé la bibliothèque **Formik** pour faciliter la gestion des formulaires et la bibliothèque **Yup** pour imposer des restrictions lors du remplissage des champs du formulaire.

J'ai ajouté certaines restrictions pour les champs de formulaire pour éviter d'éventuelles erreurs que pourrait faire l'utilisateur.

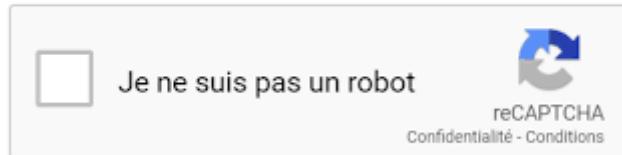
```
export default withFormik({
  mapPropsToValues: () => ({
    firstName: "",
    lastName: "",
    phoneNumber: "",
    email: "",
    message: "",
  }),
  validationSchema: Yup.object().shape({
    firstName: Yup.string()
      .min(2, "Trop court !")
      .max(50, "Trop long !")
      .required("Veuillez saisir votre prénom."),
    lastName: Yup.string()
      .min(2, "Trop court !")
      .max(50, "Trop long !")
      .required("Veuillez saisir votre nom."),
    phoneNumber: Yup.string()
      .min(10, "Trop court !")
      .max(10, "Trop long !")
      .required("Veuillez saisir votre numéro de téléphone."),
    email: Yup.string()
      .email("Veuillez saisir un email valide.")
      .required("Veuillez saisir votre email."),
    message: Yup.string()
      .min(10, "Trop court !")
      .max(1000, "Trop long !")
      .required("Veuillez saisir votre message."),
  }),

  handleSubmit: (values, { props }) => {
    const message = {
      firstName: values.firstName,
      lastName: values.lastName,
      phoneNumber: values.phoneNumber,
      email: values.email,
      message: values.message,
    };
    props.sendMail(message);
  },
})(Form);
```

Comme le formulaire n'est pas en ligne pour l'instant, je ne peux pas savoir s'il est fonctionnel ou pas mais dans mes autres projets que je vais présenter, ils le sont.

4.1 Ajout de Google reCAPTCHA

Veuillez cocher la case ci-dessous pour continuer.



Afin de renforcer la sécurité du formulaire de contact et prévenir les spams, j'ai intégré Google **reCAPTCHA**.

L'installation du reCAPTCHA a été effectuée en utilisant la commande `npm install react-google-recaptcha`, puis je l'ai importé par la suite dans le projet.

J'ai généré un identifiant unique de sécurité grâce à Google reCAPTCHA. Cet identifiant doit être stocké dans un fichier caché, de manière à ce qu'il ne soit pas accessible en ligne, renforçant ainsi la sécurité de l'application.

```
import ReCAPTCHA from "react-google-recaptcha";

const Recaptcha = ({ onChange }) => {
  return (
    <ReCAPTCHA
      sitekey="6LcHahge-[REDACTED]NExrK24tmIgsSaL0_ZdUaJzXJ"
    />
  );
}

export default Recaptcha;
```

5.0 Page Voitures d'Occasions

Mise en Place des Composants de Recherche et de Filtrage

Au cours de cette phase de développement, j'ai conçu des éléments d'interface utilisateur (composants), notamment des **sliders**, des sélecteurs (**dropdowns**), et des cases à cocher (**checkboxs**), visant à faciliter la recherche et le filtrage des données.

Pour améliorer l'expérience utilisateur, je vais créer des **sliders** prix, km et année en offrant la possibilité à l'utilisateur de définir à la fois une valeur **minimale et maximale** pour chaque paramètre. Cette approche permettra aux utilisateurs de mieux cibler leurs critères de recherche.

The screenshot shows a user interface for filtering vehicle search results. At the top, there are five checkboxes labeled: Utilitaire, Berline, Familiale, Citadine, and SUV. Below these is a dropdown menu labeled "Marque : Toutes". The main area contains three horizontal sliders. The first slider is labeled "Prix :" and has tick marks at 5 000 € and 50 000 €. The second slider is labeled "Année :" and has tick marks at 2000 and 2023. The third slider is labeled "Kilométrage :" and has tick marks at 0 km and 200 000 km. At the bottom right is a blue button labeled "Rechercher".

J'ai structuré le code en créant des composants distincts pour chaque élément de filtrage, qu'il s'agisse de **cases à cocher, de menus déroulants, ou de sliders**.

Ces composants individuels communiquent avec un composant parent centralisé nommé "**VehiculeFilters**" pour gérer l'état des filtres de manière cohérente.

Ne pouvant directement communiquer entre eux, je vais créer pour les composants parents et fils une fonction **handlechange** qui prendra en paramètre un évènement lorsque l'utilisateur cliquera sur le bouton rechercher.

Elle sera définie pour mettre à jour l'état des filtres lorsque l'utilisateur effectuera une sélection dans l'un des composant filtre.

Je la passerai en tant que **props** aux composant fils pour qu'ils puissent **mettre à jour** l'état parent.

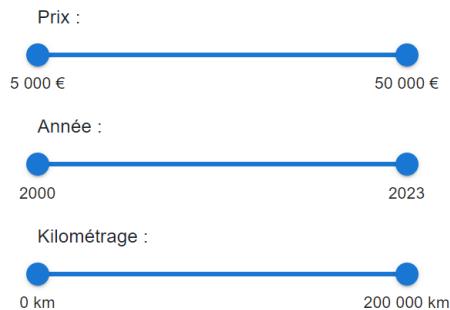
Chaque composant a été créé en utilisant la bibliothèque **Material-UI** (<https://mui.com/>).

J'ai installé les dépendances nécessaires pour garantir leur bon fonctionnement ce qui m'a beaucoup aidé car au départ, j'ai voulu créer les composants de filtres par moi-même, ça fonctionnait mais visuellement, ce n'était pas terrible.

5.1 Composant BasicRange

Je vais faire en sorte de créer la fonction qui attendra en paramètre un objet et qui utilisera le **destructuring** en extrayant les propriétés de l'objet dans des variables locales.

l'objet passé en paramètre ne contient pas de propriétés spécifiques, alors je mettrai une valeur par défaut.



Après avoir effectué des recherches, j'ai conclu que le **destructuring** est une technique efficace pour extraire des valeurs à partir d'objets ou de tableaux, ce qui a motivé mon choix.

Pour ce qui est du tri d'un tableau, j'ai opté pour l'utilisation de la fonction `sort()` sans recourir au destructuring.

Cette fonction trie les éléments du tableau en les comparant les uns aux autres, ce qui me permet de réorganiser les éléments du tableau de manière à les placer dans un ordre spécifique.

```
range.sort((a, b) => a - b);
```



```
const BasicRange = ({ name = "slider", range = [0, 100], marks, label = "", handleChange }) => {
    // Utilisation de destructuring pour extraire les valeurs des props avec des valeurs par défaut

    // Tri du tableau range si nécessaire, de sorte que la valeur la plus petite soit toujours en 1ère
    // position et la plus grande en dernière
    // On l'utilisera car pour les 3 composants prix, km et année, leurs valeurs sont des entiers et que
    // sort trie uniquement des chaînes de caractères.

    range.sort((a, b) => a - b);

    // Utilisation de useState pour gérer la valeur actuelle du slider
    const [value, setValue] = useState(range);

    // Fonction de gestion du changement de valeur du slider
    const handleSliderChange = (event, newValue) => {
        setValue(newValue);
        handleChange(name, newValue); // Appel de la fonction handleChange du parent avec le nom et la
        // nouvelle valeur
    };
}
```

Le composant BasicRange comporte les éléments suivants :

- **handleChange** : Cette fonction sera appelée à chaque modification de la valeur du slider et recevra deux paramètres : le nom du paramètre (comme prix, kilométrage, année) et la nouvelle valeur sélectionnée.

- **label** : Un libellé qui s'affiche au-dessus du slider pour aider l'utilisateur à comprendre les valeurs sélectionnées.

- **name** : Le nom du slider.

- **marks** : Un tableau d'objets spécifiant les marques (valeurs) sur le slider, chaque objet ayant une valeur et une étiquette (label). Par exemple, pour le prix : "5 000 - 50 000".

- **range** : La plage de valeurs possibles du slider. J'ai utilisé une plage de base de 0 à 100, qui convient à de nombreuses situations.

J'ai veillé à ce que la plus petite valeur soit toujours en première position en utilisant une fonction de tri pour maintenir l'ordre correct.

```

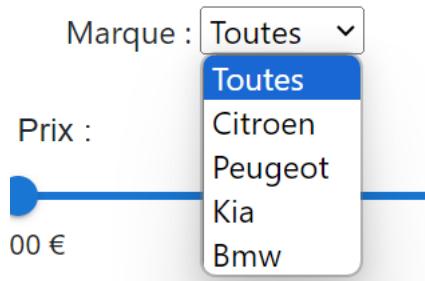
return (
  <>
    <Typography id="input-slider" gutterBottom>
      {label}
    </Typography>

    <Slider
      name={name}
      min={range[0]}
      max={range[1]}
      onChange={handleSliderChange}
      size="medium"
      valueLabelDisplay="auto" // Afficher la valeur actuelle
      marks={marks || [{ value: range[0], label: range[0].toString() }, { value: range[1], label: range[1].toString() }]}
      value={value}
    />
  </>
);
};

export default BasicRange;

```

5.2 Composant BasicSelect



J'ai créé ce composant en un menu déroulant avec des options configurables, telles que les marques de voitures.

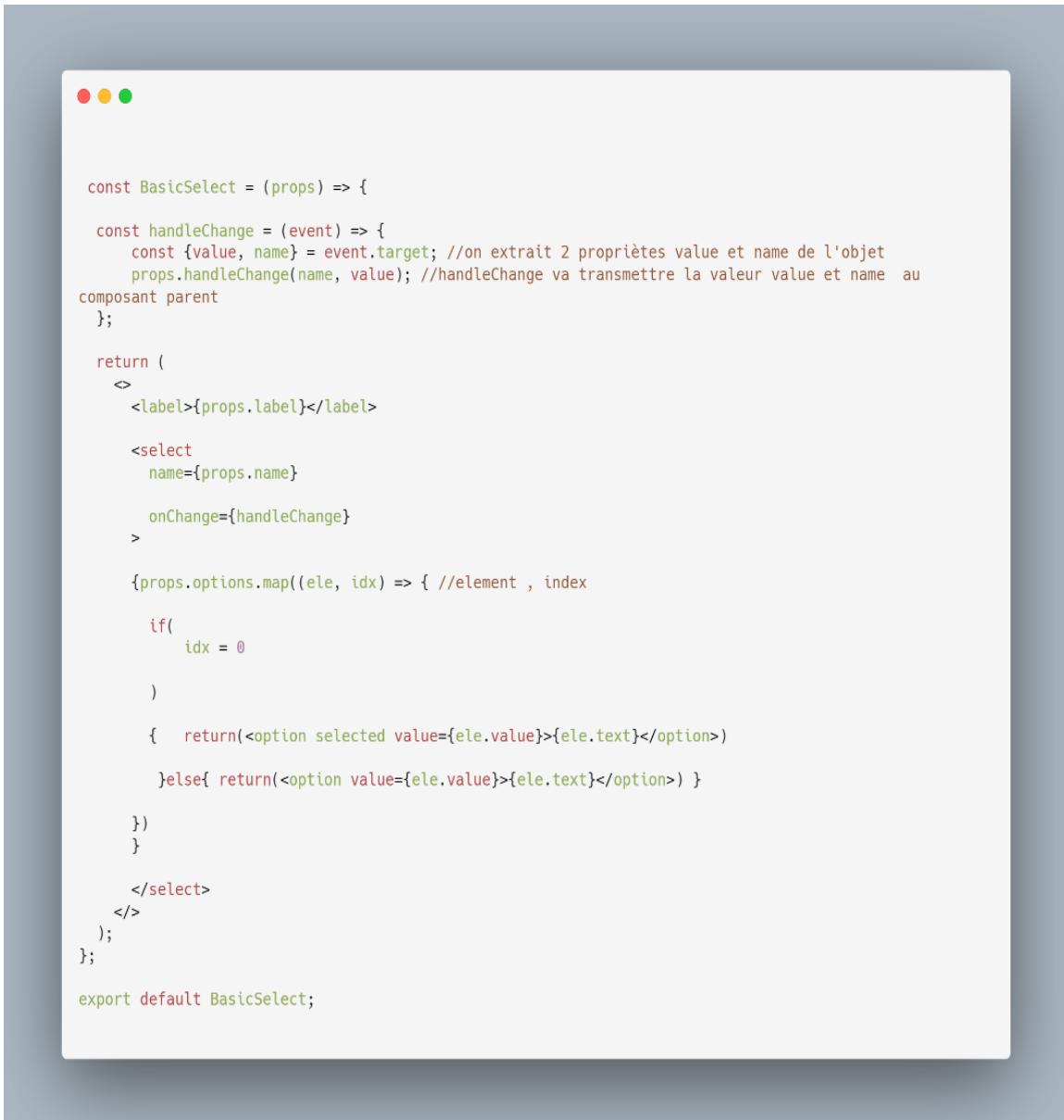
Lorsque l'utilisateur sélectionne une option, la fonction **handleChange** sera utilisée pour communiquer avec le composant parent.

Cette fonction sera appelée à chaque fois que l'utilisateur modifie la sélection dans le menu déroulant. Elle prendra en paramètre l'objet **event**.

Marque : **Toutes**

Je vais utiliser le **destructuring** pour extraire les propriétés **value** et **name** de l'objet **event**.

Je vais ensuite appeler la fonction `props.handleChange(name, value)` ; pour transmettre les valeurs au composant parent ce qui permettra à ce composant de réagir à la sélection effectuée dans le menu déroulant.



```
const BasicSelect = (props) => {

  const handleChange = (event) => {
    const {value, name} = event.target; //on extrait 2 propriétés value et name de l'objet
    props.handleChange(name, value); //handleChange va transmettre la valeur value et name au
  }

  return (
    <>
      <label>{props.label}</label>

      <select
        name={props.name}
        onChange={handleChange}
      >

        {props.options.map((ele, idx) => { //element , index
          if(
            idx = 0
          )
          {   return(<option selected value={ele.value}>{ele.text}</option>)
            }else{ return(<option value={ele.value}>{ele.text}</option>)}
          }
        })
      </select>
    </>
  );
}

export default BasicSelect;
```

5.3 Composant BasicCheckbox

Ce composant servira à sélectionner une famille de véhicule. L'utilisateur pourra en sélectionner plusieurs s'il le souhaite ou si rien de cocher, je ferais en sorte de lui présenter tout type de famille de véhicules.

Utilitaire Berline Familiale Citadine SUV

Comme pour les autres composants filtres, la fonction `handleChange` sera appelée chaque fois que l'utilisateur coche ou décoche une case. La fonction prendra en charge un objet **événement** en tant que paramètre.

Je vais par la suite transmettre cet événement au composant parent grâce à la fonction `props.handleCheckBoxChange(e)`, cela permettra au composant parent de réagir à l'état de la case si cochée ou pas.



5.4 Composant Parent VehiculeFilters

La page nommée "Voitures d'occasion" servira donc de parent pour intégrer les composants enfants que j'ai créés précédemment, à savoir `'BasicCheckbox'`, `'BasicSelect'`, et `'BasicRange'`.

C'est effectivement sur cette page que l'utilisateur effectuera sa recherche en ajustant les filtres en fonction de ses choix.

Pour le slider "année", j'ai créé une fonction permettant de mettre à jour la date de fin du slider en temps réel, évitant ainsi une mise à jour manuelle chaque année.



```
//Fonction pour obtenir l'année actuelle en utilisant l'objet date.  
const getCurrentYear = () => {  
    const dateActuelle = new Date();  
    const anneeActuelle = dateActuelle.getFullYear();  
    return anneeActuelle;  
};
```

J'ai implémenté la fonction **handleChange**, qui est transmise aux composants enfants via les props et qui met à jour l'état local des filtres en fonction des modifications apportées par l'utilisateur.



```
const handleChange = (name, newValue) => {  
    setFiltres({ ...filtres, [name]: newValue });  
    //prendra 2 paramètres name (le nom du filtre à mettre à jour et newValue, la nouvelle valeur du  
filtre.  
};
```

La fonction **handleCheckboxChange** sera utilisée pour gérer les cases à cocher. Elle mettra à jour l'état des filtres en fonction de ce qui est coché ou pas.

Dans cette fonction, j'extrais les 3 propriétés de l'objet **événement : name** qui est le nom de la case à cocher, **value** qui est la valeur associée à la case à cocher et **checked** qui sera un booléen qui indiquera (**true**) si cochée ou (**false**) si décochée.



```
const handleCheckBoxChange = (e) => {
  const { name, value, checked } = e.target;
  if (checked) {
    setFiltres({ ...filtres, [name]: [...filtres.famille, value] });
  } else {
    setFiltres({
      ...filtres,
      [name]: filtres.famille.filter((ele) => ele !== value),
    });
  }
};
```

5.5 Hook UseEffect

Je vais ensuite utiliser le **hook useEffect** pour effectuer les requêtes http grâce à **Fetch** lorsqu'il sera monté (affiché pour la première fois) grâce à **AXIOS** qui effectuera ces requêtes et ainsi récupérer les données des véhicules en fonction des paramètres de filtres choisi par l'utilisateur.



```
useEffect(() => {
  fetch(
    //fetch effectue une requête http, si reponse, elle sera encapsulé dans une promesse
    lien
    // "http://localhost/GarageBack/API/vehicule.php?
    kilometremin=0&kilometremax=200000&anneemin=2000&anneemax=2023&prixmin=5000&prixmax=50000"
  )
  //Si reponse reçu de la requête http, then va gérer la réponse de cette promesse et va prendre une
  //fonction de rappel en argument:

  .then((res) => res.json())// Va extraire les données de l'API sous format json
  .then((data) => {
    setCards(data)
    console.log();
  })//data ou on aurait pu mettre un nom représente la réponse de la requête http
  .catch((err) => console.log(err));//Si erreur de la requête, catch retourne une erreur
}, [lien]);
```

5.6 Fonction handleClick

Je vais créer une fonction qui va construire une Url dynamique en fonction des filtres sélectionnés dans l'objet **lienObjet**. Il supprimera le dernier caractère ‘&’ pour obtenir **une Url propre**.



```
const handleClick = ()=> {

    let lienTmp = "http://localhost/garageback/API/vehicules.php?";
    let lienObject = {kilometremin:filtres.kilometrage[0],
                     kilometremax:filtres.kilometrage[1],
                     prixmin:filtres.prix[0],
                     prixmax:filtres.prix[1],
                     anneemin:filtres.annee[0],
                     anneemax:filtres.annee[1],

    };
    if(filtres.marque.length !== 0){
        lienObject.marque = filtres.marque;
    }

    if(filtres.famille.length !== 0){
        lienObject.famille = filtres.famille.join(",");
    }

    for(const [cle, valeur] of Object.entries(lienObject)){
        lienTmp = lienTmp + `${cle}=${valeur}&`
    }
    lienTmp = lienTmp.slice(0, -1);
    // console.log(lienTmp)
    setLien(lienTmp)
}

}
```

6.0 Postman

Avant de créer la variable `Lien`, j'ai effectué des tests sur mes URLs de recherche par filtres en utilisant **Postman**, ce qui m'est avéré être d'une grande aide pour la gestion des erreurs.

Je renseignais dans mon URL les valeurs **minimales et maximales** des filtres que j'avais précédemment définies.

Exemples de Liens de test en méthode GET

Pour tester la recherche par marque :

<http://localhost/GarageBack/API/vehicule.php?marque=citroen>

Pour tester la recherche de tous les filtres avec les valeurs de début et de fin renseignées :

<http://localhost/GarageBack/API/vehicule.php?kilometremin=0&kilometremax=200000&anneemin=2000&anneemax=2023&prixmin=5000&prixmax=50000>

The screenshot shows the Postman application interface. On the left, there's a sidebar with 'My Workspace' containing a 'garage' collection which includes a 'vehicule' folder. The main area shows a 'garage / vehicule / http://localhost/GarageBack/back/login' endpoint with a 'GET' method. In the 'Params' tab, several filters are set: 'marque' (citroen), 'kilometremin' (0), 'kilometremax' (200000), and 'anneemin' (2000). Below the table, the response status is '200 OK' with a response time of 11 ms and a size of 1.82 KB. The response body is displayed in JSON format, showing a list of vehicles matching the filters.

Bien sûr après que tous ces paramétrages ont été réalisé, je retourne tous les composants de filtrage.

Difficultés rencontrées ‘Recherches par filtres’ :

Cette partie du développement a été l'une des plus complexes pour moi.

Tout d'abord, j'ai initialement essayé de créer mes propres composants de sliders, de cases à cocher, etc., alors que des bibliothèques existaient pour simplifier ce processus mais au départ, je n'en avais pas eu connaissance, n'étant pas satisfait du résultat, j'ai réalisé plusieurs recherches pour trouver une solution et proposer au client un front plus présentable.

J'ai aussi éprouvé des difficultés à bien cibler mes recherches, mais j'ai heureusement bénéficié de conseils avisés qui m'ont facilité la tâche.

Mon apprentissage s'améliore au fil du temps que je pratique, et je suis désormais conscient des ressources disponibles pour résoudre de tels problèmes techniques.

De plus, j'ai rencontré des défis lors de la mise en place de la fonction handleChange et de sa transmission en tant que « props » aux composants enfants, mais j'ai compris l'importance de cette fonction pour établir la communication entre les composants.

J'ai également été confronté à plusieurs erreurs liées à des noms de composants mal orthographiés, tant au niveau des noms de fichiers que des imports.

Je suis conscient de la sensibilité à la casse dans ce contexte, mais dans ce cas précis, j'ai eu du mal à identifier l'origine de l'erreur, car tous les noms semblaient correctement écrits.

Pour résoudre ce problème, j'ai dû supprimer tous les fichiers des composants et de les recréer un à un afin que l'erreur cesse de s'afficher.

C'est l'un des mystères de l'informatique qui peut parfois entraîner une perte de temps considérable pour ce qui semble être une petite erreur en apparence.

La convention stipule que les fichiers doivent commencer par une lettre majuscule. Cependant, de manière inexplicable, il arrive parfois que certains fichiers se renomment automatiquement, perdant ainsi leur lettre majuscule initiale.

Cela provoque des erreurs lors de l'importation de ces fichiers. Je ne parviens pas à expliquer pourquoi cela se produit, mais cette situation devient de plus en plus frustrante à chaque occurrence.

7.0 Prestations

En ce qui concerne la présentation des services du garagiste, le système est presque identique à celui utilisé pour afficher les informations des véhicules. Les services sont affichés sous forme de "cartes" Bootstrap.

8.0 Avis Clients

Cette partie représente la dernière étape que j'ai pu accomplir dans ce projet, mais elle est encore loin d'être achevée.

Je récupère bien mes données en Get mais en Post, je n'y suis pas encore parvenu, j'ai des problèmes au niveau de la requête.

J'ai créé un composant d'étoiles pour la notation et j'ai installé la bibliothèque '**npm install react-simple-star-rating**'. **Opérationnel**

Il n'y a même pas besoins de cliquer sur les étoiles pour les sélectionner, un survol de la souris suffira.



J'ai décidé ici de ne pas inclure les demi-étoiles pour l'instant, je verrais cela plus tard.



```
const Stars = () => {
  const [rating, setRating] = useState(100) // Va jusqu'à 100 donc la 5 étoiles
  const handleRating = (rate) => {
    console.log(rate)
    setRating(rate)
  }
  return (
    <Rating
      fillColor="#F0C300"
      //allowHalfIcon //Pour les demi étoile, là, on est à true
      // tooltipArray={['nul', 'bof', 'moyen', 'top', 'génial']}
      transition
      // showTooltip
      onClick={handleRating}
      ratingValue={rating}
    />
  )
}
export default Stars;
```

Je vais maintenant convertir les étoiles en note en créant un nouveau composant ‘ConversionNote’

A l’intérieur de la fonction, je vais créer une variable d’état ‘note’ à l’aide du Hook ‘**useState**’ et je vais l’initialiser à **0**.

```

    const ConversionNote = () => {
      const [note, setNote] = useState(0);
      const handleNoteChange = (newNote) => {
        setNote(newNote);
      };
      const etoiles = [];

      for (let i = 1; i <= 5; i++) {
        etoiles.push(
          <Stars
            key={i}
            selected={i <= note}
            onEtoileClick={() => handleNoteChange(i)}
          />
        );
      }
      return (
        <div className="rating-container">
          <div className="five-rate-active">{etoiles}</div>
        </div><p>Note : {note}</p>
      );
    }

    export default ConversionNote;

```

J'ai testé dans la console en y ajoutant un echo, ça fonctionne parfaitement

Note :



Download the React DevTools for a better development experience: <https://reactjs.org/link/react-devtools>

✖ Error while trying to use the following icon from the Manifest: <http://localhost:3000/logo192.png> (Download error or resource isn't a valid image)

4

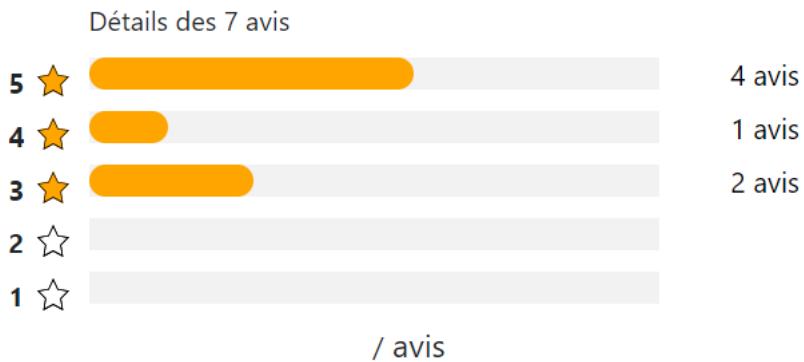
[Rating.js:8](#)

4

[Rating.js:8](#)

>

J'ai créé un composant destiné à informer l'utilisateur et à générer un bilan des avis clients (bien que non opérationnel pour le moment).



Par la suite, sur la page d'accueil, j'ai l'intention d'afficher les avis de manière dynamique. Pour ce faire, je vais générer un nombre aléatoire qui sélectionnera aléatoirement un avis à afficher.

Ce garage a fait un excellent travail pour réparer ma voiture. Ils m'ont expliqué en détail ce qui devait être fait et ont répondu à toutes mes questions. J'apprécie leur honnêteté et leur professionnalisme. Mon véhicule fonctionne parfaitement maintenant.

Pauline

★★★★★

Laisser Un Avis !

9.0 Conclusion

En fin de compte, la mise en place des composants en ce qui concerne la recherche par filtre a été une étape assez instructive en ce qui me concerne par rapport à mon apprentissage.

J'ai été confronté à de nombreux problèmes, que ce soit avec Git et Github, la connexion à la base de données, la création des tables et bien entendu le code.

Souvent, je passais trop de temps à me battre avec un problème, mais j'ai fini par réaliser que cela ne servait à rien de rester des jours à essayer de le résoudre.

Maintenant, je préfère passer à autre chose temporairement, le temps de réfléchir à la meilleure approche pour le résoudre. Cette méthode me convient bien.

Cette expérience m'a permis de développer mes compétences en gestion de bases de données et d'acquérir une compréhension plus approfondie des aspects pratiques du développement.

Mon apprentissage continu dans ce domaine m'aidera à aborder de futurs projets avec plus de confiance et de compétence.

J'ai pris conscience que la mémorisation forcée ne mène à rien, car il est impossible d'absorber l'ensemble des informations par cœur.

La méthode qui fonctionne le mieux pour moi consiste à pratiquer, pratiquer et pratiquer de manière intensive tout en effectuant des recherches constantes.



**HARMONY
DIGITAL**

PARTIE BACK END



Introduction

Ce projet a été élaboré pour un client fictif, le propriétaire d'un garage qui a exprimé le souhait de vendre des voitures d'occasion et de proposer ses services de réparation de véhicules à travers un site web.

Dans le cadre de la mise en place de la partie Back-end de ce projet, j'ai choisi d'utiliser des technologies telles que PHP, MySQL, ainsi que les environnements de développement Wamp et XAMPP au début.

Toutefois, en raison des problèmes rencontrés avec Xampp, j'ai décidé de migrer vers une autre solution plus adaptée à mes besoins qui est Wampp.

Une décision importante a été de ne pas recourir à Symfony pour ce projet, préférant opter pour une approche de développement en PHP pur.

On m'a fréquemment conseillé, en tant que débutant, de maîtriser les bases de la programmation en travaillant directement avec du code non abstrait.

Cette approche m'a grandement aidé à mieux appréhender la logique sous-jacente du code. Toutefois, l'absence d'un framework pour accélérer le développement a entraîné une augmentation du temps nécessaire à la réalisation du projet, qui reste partiellement achevé à ce stade.

Aujourd'hui, je suis prêt à vous présenter le résultat de mes efforts de 4 mois de travail.

Pour chaque fonctionnalité de l'appli, une branche sera créée vers Git.

Table de matière

1.0 Base de données.....	3
1.1 Schéma de la base de données.....	3
1.2 Configuration de la Base de D.....	3
1.3 Importation des Tables depuis MySQL Workbench	4
1.4 Ajout Manuel des Contraintes de Clé Étrangère	4
1.5 Création de Données Temporaires dans PHPMyAdmin.....	5
1.6 Ajout de Données dans la Table Horaire.....	6
1.7 Création de la table avis.....	6
2.0 Mise en place de l'environnement de travail.....	6
2.1 Création du fichier index.php	6
2.2 Création du fichier .htaccess	6
2.3 Bloc "try" "catch": Gestion des Exceptions.....	7
2.3.1 Affichage de messages d'erreur conviviaux pour les utilisateurs.....	7
2.3.2 Journalisation des erreurs.....	8
2.3.3 Gestion spécifique des différents types d'exceptions.....	8
2.3.4 Propagation de l'exception.....	8
3.0 Mise en Place du Système de Routage.....	8
3.1 Création du fichier index.php.....	8
3.2 Tests du Système de Routage.....	9
3.3 Personnalisation des URLs.....	10
4.0 Le Rôle du Routeur.....	10
5.0 La Structure MVC.....	10
5.1 Le Contrôleur.....	11
5.2 Le Modèle.....	11
5.3 La Vue.....	11

6.0 Mise en Place du Contrôleur et du modèle Front-End	11
7.0 Utilisation de PDO.....	12
8.0 Tests avec de Faux Données.....	13
9.0 Liaison de toutes les champs à la base de données "Garage"	14
10.0 Mise en Place des Filtres pour les véhicules.....	14
11.0 Gestion des Problèmes Techniques.....	14
12.0 Développement du Contrôleur Front-End véhicules.....	15
13.0 Espace Administrateur.....	15
13.1.0 : Page Login.....	16
13.1.1 Mise en Place du Système de Connexion.....	16
13.1.2 Création des Champs de Connexion.....	16
13.1.3 Utilisation de password_hash().....	16
13.1.4 Modification des Tables "Employés" et "Admin".....	16
13.1.5 Tests du Système de Connexion.....	17
13.1.6 Insertion de Données Factices.....	17
13.1.6 Validation des Tests et Conclusions.....	17
13.1.7 Sécurisation et Vérification des Informations de Connexion.....	18
14.0 Tests et affichage des données factices dans la table administrateur.....	19
14.1 Correction des Erreurs de Connexion.....	20
14.2 Tests de Données.....	21
14.3 Mise en Forme des Données.....	21
15.0 Implémentation du Bouton de Suppression.....	21
15.1 Création d'une Nouvelle Route.....	21
15.2 Conversion de l'ID en Entier.....	21
15.3 Mise en Place d'Alertes JavaScript.....	21
15.4 Gestion des Redirections.....	22
16.0 Bouton modifier.....	23
17.0 Bouton création d'un véhicule.....	24
18.0 Ajout d'images lors de la création.....	26

19.0 Suppression d'images.....	27
20.0 Gestion des dates.....	28
21.0 Mise en place du back recherche par filtres.....	29
22.0 API vehicules.php.....	31
23.0 vehicule_model.php.....	32
24.0 Avis Clients.....	34
25.0 Prestation du garage.....	34
26.0 Conclusion.....	35

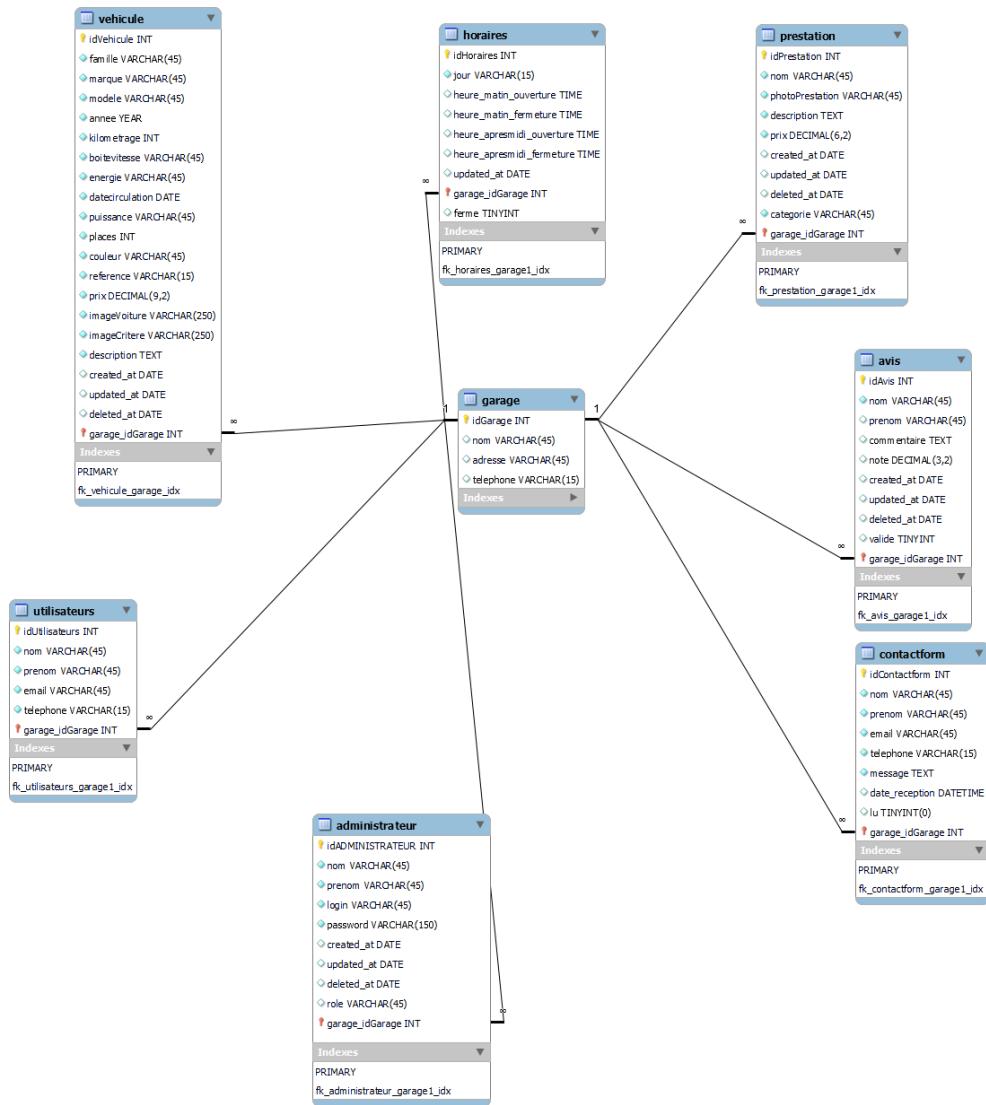
1.0 Base de données

Base De Données

1.1 Schéma de la base de données

En accord avec les spécifications que j'ai élaborées, j'ai conçu le schéma de la base de données, un processus qui a connu diverses évolutions au fil de l'avancement du projet.

Il est fréquemment complexe d'anticiper tous les aspects dès le départ, d'où la nécessité d'ajustements progressifs en réponse aux changements et aux besoins qui émergent au cours du développement.



1.2 Configuration de la Base de Données

Pour commencer, je vais configurer la base de données et vous citer tous les défis que j'ai pu rencontrer tout au long du processus.

L'objectif principal était de concevoir une base de données fonctionnelle et efficace pour pouvoir stocker les données de mon application.

1.3 Importation des Tables depuis MySQL Workbench

Au départ, j'ai utilisé MySQL Workbench pour concevoir le modèle conceptuel de la base de données, y compris les relations entre les tables.

Cependant, lors de l'exportation vers MySQL PHPMyAdmin, j'ai rencontré des problèmes liés aux cardinalités entre les tables.

Malgré mes efforts pour configurer les relations correctement dans MySQL Workbench, les contraintes de clé étrangère n'ont pas été prises en compte lors de l'exportation.

Table	Action	Lignes	Type	Interclassement	Taille	Perte
administrateur	Parcourir Structure Rechercher Insérer Vider Supprimer	1	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
avis	Parcourir Structure Rechercher Insérer Vider Supprimer	8	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
contactform	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
garage	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	16,0 kio	-
horaires	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
prestation	Parcourir Structure Rechercher Insérer Vider Supprimer	9	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
utilisateurs	Parcourir Structure Rechercher Insérer Vider Supprimer	0	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
vehicule	Parcourir Structure Rechercher Insérer Vider Supprimer	7	InnoDB	utf8mb4_0900_ai_ci	32,0 kio	-
8 tables	Somme		25	MyISAM	utf8mb4_0900_ai_ci	240,0 kio

1.4 Ajout Manuel des Contraintes de Clé Étrangère

Pour résoudre, enfin, je croyais, ce problème, j'ai dû ajouter manuellement les contraintes de clé étrangère pour chaque table à l'aide de commandes SQL.

Cela a permis d'établir les relations souhaitées entre les tables. Voici un exemple de commande que j'ai utilisée pour ajouter une contrainte de clé étrangère :

```
1. ALTER TABLE table_enfant
2. ADD CONSTRAINT fk_nom_contrainte
3. FOREIGN KEY (colonne_etrangere) REFERENCES table_parente(colonne_primaire);
```

Cette solution a normalement permis l'intégrité des données dans ma base de données.

Suite à la résolution initiale de mon problème, j'ai vérifié le modèle conceptuel dans PHPMyAdmin et j'ai constaté que les cardinalités n'avaient toujours pas été prises en compte.

Face à cette situation, j'ai dû prendre une décision pour avancer dans le projet. J'ai choisi de passer par le panneau de commande SQL pour ajouter manuellement les contraintes de clé étrangère, table par table.

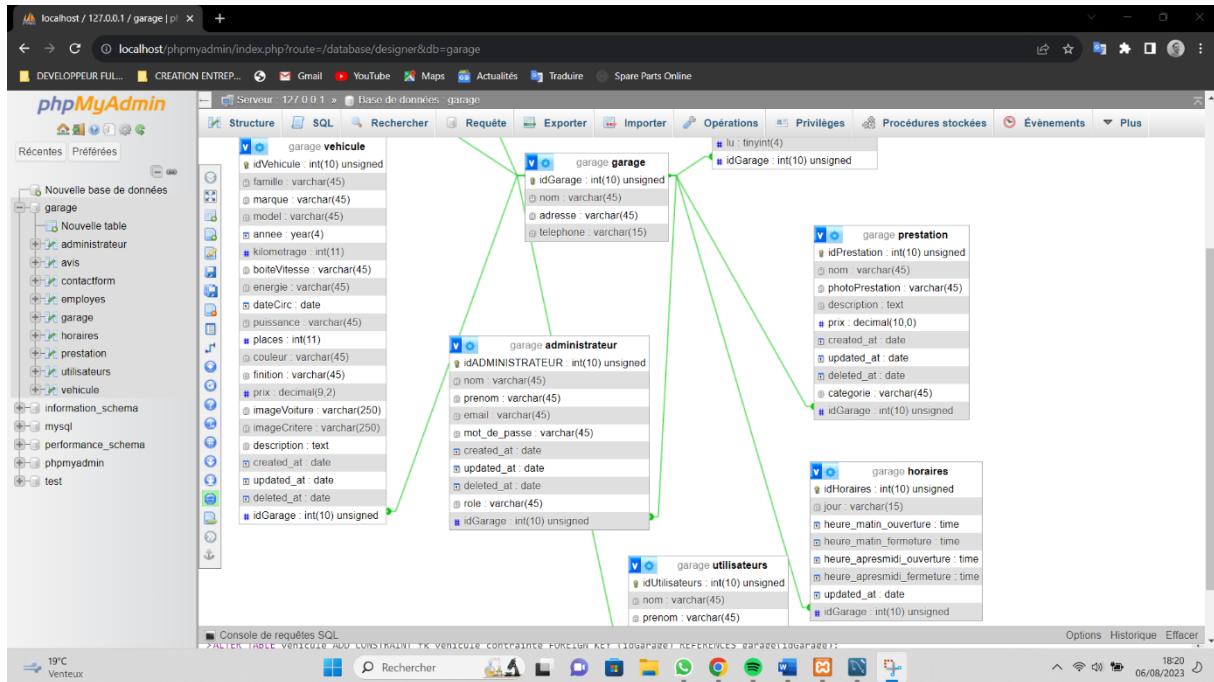
Bien que cette solution ait permis de créer les relations entre les tables conformément à nos besoins, je reste conscient que cela peut être considéré comme une approche moins automatisée et plus sujette aux erreurs.

Cependant, dans le contexte de mon projet et compte tenu du temps limité, c'était la solution la plus pratique pour pouvoir avancer.

Pour l'avenir, je vais continuer à travailler sur l'optimisation de notre base de données et des contraintes de clé étrangère, et je réévaluerai la possibilité de les réactiver une fois que je serai certain que toutes les données sont conformes aux contraintes.

Cette approche garantira l'intégrité des données à long terme tout en me permettant de poursuivre le développement sans interruption.

En fin de compte, cette expérience m'a montré l'importance de la flexibilité et de l'adaptabilité dans le processus de développement, ainsi que la nécessité de prendre des décisions pragmatiques pour faire progresser un projet, même en cas de difficultés imprévues.



1.5 Création de Données Temporaires dans PHPMyAdmin

Après avoir résolu les problèmes liés aux contraintes de clé étrangère, j'ai entrepris de saisir de fausses données temporaires dans les tables de ma base de données.

L'objectif était de créer un environnement de test pour vérifier le fonctionnement de mon application.

Cependant, j'ai de nouveau rencontré des erreurs liées aux clés primaires lors de l'insertion de ces données.

Après une analyse plus approfondie, j'ai pu identifier la source du problème.

J'avais mal interprété les cardinalités des relations entre les tables.

J'avais créé manuellement les clés étrangères alors que si j'avais choisi les bonnes cardinalités, les clés auraient été créées automatiquement dans chaque table.

Une fois que j'ai ajusté les cardinalités correctement, j'ai pu exporter les données sans problème.

1.6 Ajout de Données dans la Table Horaire

Lorsque j'ai commencé à ajouter des données dans la table "horaire", j'ai réalisé que j'avais initialement considéré uniquement les jours d'ouverture, négligeant les jours de fermeture.

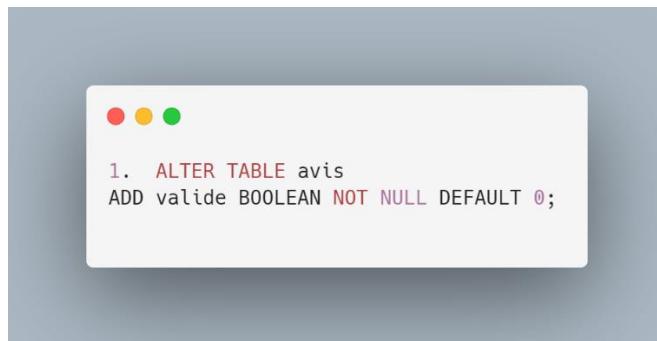
J'ai rapidement rectifié cette omission en ajoutant une colonne supplémentaire appelée "ferme" à la table.

Cette colonne prendra la valeur 0 si l'établissement est ouvert et 1 si c'est fermé, me permettant ainsi de gérer correctement les jours de fermeture.

1.7 Création de la table avis

Je vais initialiser dans la table ‘avis’ un état **valide(1) ou non valide(0)** pour l’espace administrateur sous forme de **booléen**.

Je vais l’initialiser au départ dans la base de données à 0 comme non valide avec une commande Sql :



Chaque **Avis** sera donc initialisé à **0** automatiquement comme non valide car après, l’administrateur devra le valider ou pas dans son espace admin après vérification.

2.0 Mise en place de l’environnement de travail

Mise en Place de l’Environnement de Travail

Pour optimiser mon environnement de développement, j’ai réalisé plusieurs étapes clés :

2.1 Création du fichier index.php

Ce fichier joue un rôle central dans le modèle MVC (Modèle-Vue-Contrôleur) que j’ai créé.

Toutes les pages qui seront réalisées ultérieurement seront redirigées vers index.php grâce à la méthode GET.

Cela permettra une gestion centralisée des requêtes et des vues.

2.2 Création du fichier .htaccess

J’ai également créé un fichier .htaccess pour configurer le serveur Apache de XAMPP.

Ce fichier va mettre en place un système de réécriture des URLs, ce qui rendra les URLs de l’application plus compréhensibles pour les utilisateurs.

Ce fichier est apparemment inaccessible au public, c’est un fichier caché comme on dit.

Ce fichier a permis de configurer le serveur Apache de XAMPP pour la réécriture des URLs, rendant ainsi les URLs plus compréhensibles et lisibles.

Dans l’ensemble, ces étapes m’ont permis de mettre en place un environnement de développement je pense, robuste et de progresser dans la création de l’application.

Je continue à travailler sur l'ajout de données factices dans nos tables pour tester et affiner le fonctionnement de l'application.

2.3 Bloc "try" "catch": Gestion des Exceptions

Je vais gérer le bloc "catch" dans le cadre du développement de l'application, car il permet de gérer les exceptions de manière efficace.

Voici comment ce mécanisme fonctionne :

Dans le bloc "**try**", j'ai entouré le code susceptible de générer des exceptions, telles que des opérations de base de données ou des appels à des services externes.

L'objectif était de détecter et de capturer toute erreur ou exception qui pourrait être générée lors de l'exécution de ces opérations potentiellement risquées.

```
try {
    $pdo = new PDO("mysql:host=$host;dbname=$dbname;charset=utf8", $username,
    $password);$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);

} catch (PDOException $e) {
    die("Erreur de connexion à la base de données: " . $e->getMessage());
}
```

Le bloc "catch" est l'endroit où je défini comment gérer précisément chaque type d'exception qui a été lancé dans le bloc "**try**".

Cette personnalisation me permet de répondre de manière adaptée à chaque situation.

Voici quelques exemples de ce que j'ai pu faire dans le bloc "catch" en fonction de nos besoins spécifiques :

2.3.1 Affichage de messages d'erreur conviviaux pour les utilisateurs

J'ai pu capturer l'exception, extraire les informations pertinentes et afficher un message d'erreur clair et compréhensible pour les utilisateurs.

Cela améliore l'expérience utilisateur en les aidant à comprendre ce qui s'est mal passé.

2.3.2 Journalisation des erreurs

Pour un débogage ultérieur, j'ai pu enregistrer les détails de l'exception dans un fichier journal.

Cette journalisation permet de suivre les erreurs qui se produisent en production, ce qui est essentiel pour diagnostiquer et résoudre les problèmes.

2.3.3 Gestion spécifique des différents types d'exceptions

En fonction de la nature de l'exception (par exemple, une erreur de base de données ou une erreur de fichier), j'ai pu avoir plusieurs blocs "catch" spécifiques pour chaque type d'exception. Chacun de ces blocs peut gérer l'exception de manière appropriée en fonction du contexte.

2.3.4 Propagation de l'exception

Dans certains cas, j'ai pu choisir de propager l'exception vers un niveau supérieur de l'application, où elle peut être traitée de manière plus globale.

Cela peut être utile lorsque nous ne savons pas comment gérer l'exception à l'endroit précis où elle a été lancée.

En utilisant judicieusement les blocs "**try**" et "**catch**", j'ai pu rendre l'application plus robuste et résiliente aux erreurs, ce qui sera je pense essentiel pour assurer un bon fonctionnement, fournir une bonne expérience utilisateur et faciliter le débogage en cas de problèmes.

3.0 Mise en Place du Système de Routage

Une autre étape a été la mise en place d'un système de routage pour organiser les URL de l'application.

Mon objectif était de distinguer clairement la partie "**front**" pour l'espace administrateur de la partie "**back**" qui va faire tourner mon application, tout en permettant une gestion flexible des pages.

Je souhaitais que l'URL contienne deux informations après le "/", ce qui permettait une meilleure organisation du système de routage. Par exemple, notre URL ressemblait à ceci :
``http://localhost/garageback/back/nom_de_la_page``.

Je rajouterais à la fin de l'url quand ça sera nécessaire, l'Id de l'objet sélectionné.

Pour mettre en place ce système de routage, j'ai suivi les étapes suivantes :

3.1 Création du fichier index.php :

J'ai créé un fichier **index.php** qui joue un rôle central dans notre modèle MVC (Modèle-Vue-Contrôleur).

Toutes les pages que j'ai développées ultérieurement ont été redirigées vers **index.php** grâce à la méthode GET.

Ce système de routage a contribué à l'efficacité de l'application en permettant une gestion plus claire des pages front-end et back-end, tout en offrant une expérience utilisateur améliorée.

```
<?php

error_reporting(E_ALL);
ini_set('display_errors', '1');

session_start();

define("URL", str_replace("index.php", "", (isset($_SERVER['HTTPS']) ? "https" : "http") .
"://$_SERVER[HTTP_HOST]$_SERVER[PHP_SELF"]));

define('__ROOT__', dirname(__FILE__));

require_once("controllers/front/vehicule_controller.php");
$apiController = new VehiculeController();

require_once("controllers/back/espacepro_controller.php");
$espacepro_controller = new EspaceproController();

try {
    if (empty($_GET['page'])) {
        throw new Exception("La page n'existe pas"); // Si l'URL est vide ou faussée, on lève une
exception et on affiche une page d'erreur.
    } else {
        $url = explode("/", filter_var($_GET['page'], FILTER_SANITIZE_URL)); // On récupère l'URL et on
la filtre pour pouvoir la mieux sécuriser.
        if (count($url) < 2) {
            throw new Exception("La page n'existe pas");
        }
        switch ($url[0]) {
            case "front":
                switch ($url[1]) {
                    case "voiturefiche":
                        if (count($url) < 3) {
                            throw new Exception("L'identifiant de la voiture est manquant");
                        }
                        $apiController->getCarsByFilters($url[2]);
                        break;

                    default:
                        throw new Exception("La page n'existe pas");
                }
                break;
            case "back":
                switch ($url[1]) {
                    case "login":
                        $admin_controller->getPageLogin();
                        break;
                    case "connexion":
                        $admin_controller->connexion();
                        break;
                    case "espacepro":
                        if (count($url) < 3) {
                            throw new Exception("La page n'existe pas");
                        }
                        switch ($url[2]) {
                            case "visualisationprestation":
                                $espacepro_controller->visualisationprestation();
                                break;
                            case "modificationprestation":
                                $espacepro_controller->modificationprestation();
                                break;
                            case "suppressionprestation":
                                $espacepro_controller->suppressionprestation();
                                break;
                        }
                        break;
                default:
                    throw new Exception("La page n'existe pas");
                }
        }
    } catch (Exception $e) {
    $msg = $e->getMessage();
}
?>
```

3.2 Tests du Système de Routage

Après avoir mis en place le système de routage pour distinguer clairement les parties "front" et "back" de mon application, j'ai entrepris de tester ce système pour m'assurer de son bon fonctionnement.

J'ai dû faire plusieurs tests pour m'assurer que les pages étaient correctement acheminées vers les contrôleurs appropriés.

Pour effectuer ces tests, j'ai utilisé la fonction `\echo` pour afficher des informations sur l'URL et vérifier comment le système de routage réagissait.

Cela m'a permis de diagnostiquer rapidement les éventuels problèmes de routage et de les corriger en conséquence.

3.3 Personnalisation des URLs pour les Pages "voituresFiche" et "prestations" et ainsi de suite

Pour les pages "**voituresFiche**" et "**prestations**", j'ai mis en place une personnalisation supplémentaire de l'URL pour une expérience utilisateur améliorée.

Mon objectif était de permettre l'accès direct aux fiches de voitures et aux informations sur les prestations en utilisant l'ID unique de chaque objet.

Concrètement, j'ai ajouté l'ID unique à l'indice 2 de l'URL.

Par exemple, l'URL ressemblait à ceci :

`\http://localhost/garageback/back/voituresFiche/123` ou
`\http://localhost/garageback/back/prestations/456`.

Cette personnalisation des URLs permettait aux utilisateurs d'accéder rapidement aux informations spécifiques qu'ils recherchaient, améliorant ainsi l'expérience de navigation et la convivialité de l'application.

4.0 Le Rôle du Routeur

J'ai dû ensuite mettre en place un système de routage.

Le permet de faire l'association entre les demandes de l'utilisateur et la logique du site. Cela signifie qu'il décidera comment chaque demande HTTP doit être gérée et quel contrôleur doit être appelé pour traiter la demande qui sera faîtes.

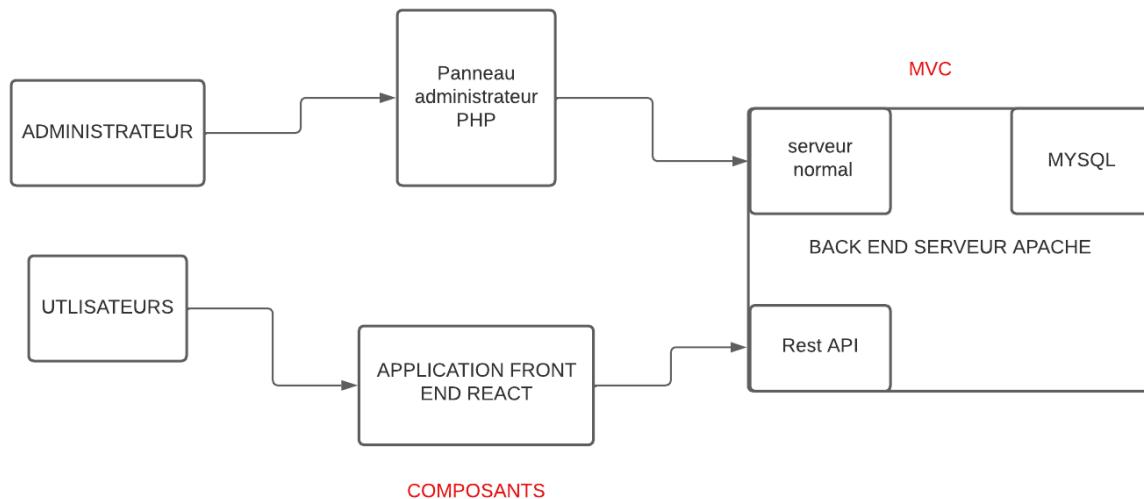
Le routeur permet d'acheminer les utilisateurs vers les bonnes pages en fonction de l'URL demandée.

Il joue un rôle crucial dans la navigation et l'expérience utilisateur.

5.0 La Structure MVC (Modèle-Vue-Contrôleur)

J'ai adopté pour l'architecture MVC (Modèle-Vue-Contrôleur) pour organiser ainsi mon code de manière efficace et modulaire.

Cette structure est le moteur dans la séparation des responsabilités et la gestion de l'application.



Voici comment elle fonctionne :

5.1 Le Contrôleur

Le contrôleur est le cœur de l'application. Il gère la partie logique et pilote la demande du client.

C'est lui qui décide quels modèles doivent être sollicités pour récupérer les données nécessaires, puis il transmet ces données à la vue.

Le contrôleur coordonne ce qui doit être mis en place pour renvoyer le résultat attendu.

5.2 Le Modèle

Le modèle est responsable de la récupération des données demandées par le contrôleur. Il interagit avec la base de données ou d'autres sources de données pour collecter les informations nécessaires.

5.3 La Vue

La vue est chargée de l'affichage des pages demandées. Elle utilise les données transmises par le contrôleur pour créer des pages web dynamiques.

Cette architecture MVC a permis une organisation claire et modulaire du code, facilitant ainsi la maintenance de l'application et le travail collaboratif entre les membres de l'équipe.

Ce fut pour moi, l'étape la plus facile de ce projet.

6.0 Mise en Place du Contrôleur et du modèle Front-End

La gestion des données est assez importante dans ce projet et je n'ai pas trop le droit à l'erreur.

À cette fin, j'ai déployé une architecture modèle-vue-contrôleur (MVC) qui permet de clairement séparer les responsabilités.

Après avoir élaboré le contrôleur front-end, l'étape suivante consiste à mettre en place le modèle afin de gérer les données et de pouvoir les récupérer en utilisant des requêtes GET.

Pour vérifier la récupération des données, je peux simplement saisir l'URL suivante :

<http://localhost/GarageBack/API/vehicules.php>

```
[{"idVehicule": "25", "famille": "utilitaire", "marque": "citroen", "modele": "456", "annee": "2010", "kilometrage": "145000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-07-28", "puissance": "5", "places": "", "couleur": "blanc", "reference": "", "prix": "12000.00", "imageVoiture": "40742_voiture1.png", "imageCritere": "85083_etiquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptatem ullam id dolore atque qui magnam magni. Aut magni voluptatem non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "26", "famille": "berline", "marque": "peugeot", "modele": "234", "annee": "2010", "kilometrage": "21000", "boitevitesse": "automatique", "energie": "electrique", "datecirculation": "2023-10-10", "puissance": "4", "places": "4", "couleur": "vert", "reference": "", "prix": "33000.00", "imageVoiture": "21752_voiture2.png", "imageCritere": "64624_etiquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptates ullam id dolores atque qui magnam magni. Aut magni voluptates non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "28", "famille": "citadine", "marque": "kia", "modele": "456", "annee": "2011", "kilometrage": "78000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-10-06", "puissance": "5", "places": "5", "couleur": "blanc", "reference": "", "prix": "12000.00", "imageVoiture": "44757_voiture3.png", "imageCritere": "80972_etiquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptatem ullam id dolore atque qui magnam magni. Aut magni voluptatem non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "29", "famille": "berline", "marque": "kia", "modele": "900", "annee": "2022", "kilometrage": "34000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-09-30", "puissance": "5", "places": "5", "couleur": "bleu", "reference": "", "prix": "34000.00", "imageVoiture": "60087_voiture4.png", "imageCritere": "78837_etiquetteEnergie.png", "description": "Lorem ipsum dolor sit amet. Est voluptatem ullam id dolore atque qui magnam magni. Aut magni voluptatem non illo maiores est nisi tempora sed aliquam galisum 33 Quis galisum id totam Quis aut impedit illio.", "created_at": "2023-10-28", "updated_at": null, "deleted_at": null, "garage_idGarage": "0"}, {"idVehicule": "31", "famille": "berline", "marque": "citroen", "modele": "567", "annee": "2023", "kilometrage": "189000", "boitevitesse": "manuel", "energie": "essence", "datecirculation": "2023-10-10"}]
```

Cette étape me permet de confirmer que j'ai bien réussi à obtenir toutes les données nécessaires, ce qui sera important pour la création ultérieure de la partie front-end en React.

J'utiliserai cette URL pour afficher les objets que j'aurai récupérés, facilitant ainsi le développement de la partie visible de l'application.

J'ai créé un fichier manager dans la partie front du modèle. Ce manager a été chargé de gérer les données en utilisant une API (Interface de Programmation Applicative).

L'API a été un élément clé pour interagir avec la base de données et récupérer les informations nécessaires pour l'application.

7.0 Utilisation de PDO pour une Liaison Sécurisée avec la Base de Données

Pour établir une connexion sécurisée entre l'application et la base de données, j'ai choisi d'utiliser **PDO (PHP Data Objects)**.

PDO est une interface de programmation permettant d'interagir avec différentes bases de données de manière sécurisée et portable.

L'utilisation de PDO a permis de prévenir les failles de sécurité potentielles telles que les injections SQL.

En utilisant des requêtes préparées, j'ai pu garantir que les données entrantes étaient correctement traitées et échappées, réduisant ainsi les risques liés à la sécurité de l'application.

La liaison sécurisée entre l'application et la base de données est une étape pour garantir l'intégrité des données et la confidentialité des informations stockées.

8.0 Tests avec de Fausses Données et la Page "voituresFiche"

Pour effectuer ces tests, j'ai utilisé des données factices que j'avais préalablement créées dans la base de données. Cette approche m'a permis de simuler un environnement réel et de vérifier si tout fonctionnait comme prévu.

Lors de ces tests, j'ai spécifié un ID dans l'URL, ce qui a permis d'identifier la voiture spécifique que je souhaitais afficher. Les données de cette voiture ont été récupérées à partir de la base de données et renvoyées sous forme de tableau.

Je vais par la suite formater ces données au format JSON pour une meilleure manipulation et un affichage dynamique sur la page. Cette étape va me permettre de garantir que l'application fonctionne correctement et renvoie les informations attendues aux utilisateurs.

```
Array
(
[0] => Array
(
[idVehicule] => 1
[famille] => Berline
[marque] => Peugeot
[model] => 208
[annee] => 2022
[kilometrage] => 23000
[boiteVitesse] => Automatique
[energie] => Essence
[dateCirc] => 2023-08-21
[puissance] => 5
[places] => 5
[couleur] => Rouge
[reference] => 2023-lk-12
[prix] => 12000.00
[imageVoiture] => peugeot.webp
[imageCritere] => critereA.webp
[description] => 1 ligne insérée.
Identifiant de la ligne insérée : 1
Warning: #1265 Data truncated for column 'prix' at row 1
Warning: #1366 Incorrect integer value: " for column `garage`.`vehicule`.`garage_idGarage` at row 1
[created_at] =>
[updated_at] =>
[deleted_at] =>
[garage_idGarage] => 1
)
```

9.0 Liaison de toutes les champs à la base de données "Garage"

J'ai relié les champs à la table **Garage** en utilisant des clés étrangères.

Initialement, j'ai rencontré un problème où il ne semblait pas être capable de localiser la table "**Garage**" avec son ID.

Après une analyse plus approfondie, j'ai découvert que le problème résidait dans les cardinalités que j'avais choisies.

Une fois que j'ai ajusté les cardinalités appropriées, le problème de la jointure a été résolu.

10.0 Mise en Place des Filtres pour les véhicules

J'ai développé la mise en place de filtres pour trier et afficher les données de manière selective. J'ai suivi une approche basée sur le modèle **MVC** en utilisant du code pur comme cité ci-dessus.

Le modèle **MVC** impliquait la connexion à la base de données dans le modèle, la création d'un contrôleur qui renvoyait les données au format **JSON**, et la mise en place de l'API pour gérer les filtres.

J'ai utilisé Postman pour effectuer des tests d'URL et résoudre les problèmes qui me sont présentés. J'ai pu ainsi développer une fonctionnalité de filtrage robuste, enfin, j'espère, pour mon application.

11.0 Gestion des Problèmes Techniques

Je me suis confronté à plusieurs problèmes techniques.

L'un d'entre eux concernait **MariaDB de XAMPP**, où je ne pouvais plus me connecter et où de nombreuses erreurs s'affichaient dans PHPMyAdmin.

Après avoir cherché une solution, j'ai remarqué que MySQL ne fonctionnait pas correctement dans le gestionnaire de tâches.

J'ai donc pris la décision de réinstaller XAMPP en sauvegardant mes fichiers, ce qui a permis de résoudre ce problème et de rétablir la connexion à la base de données.

12.0 Développement du Contrôleur Front-End véhicules

Avec la gestion des données et des filtres en place, j'ai pu me concentrer sur le développement du contrôleur front-end de l'application.

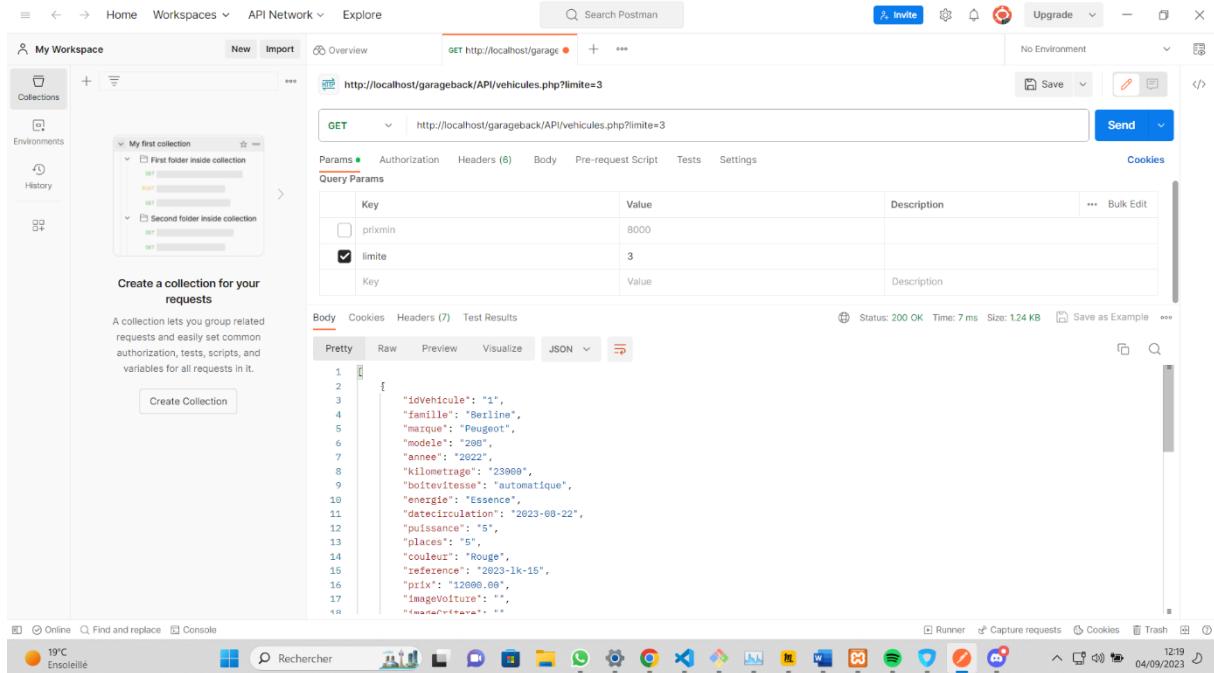
Ce contrôleur a été chargé de manipuler les données des véhicules dans l'application.

J'ai créé un fichier séparé du modèle (**VehiculeModel**) pour accéder aux données et gérer les opérations liées aux véhicules.

En utilisant des tests d'URL et en traitant les requêtes HTTP entrantes, j'ai pu mettre en place différentes actions en fonction de la méthode de la requête et des paramètres fournis dans l'URI.

J'ai également pris en compte la gestion des erreurs (par exemple, une réponse 404 en cas de problème).

Cette phase du projet a permis de préparer le terrain pour le développement du front-end de l'application.



13.0 Espace Administrateur

ESPACE ADMINISTRATEUR

13.1.0 : Page Login

PAGE LOGIN

13.1.1 Mise en Place du Système de Connexion pour l'Espace Professionnel

Une étape de mon travail qui a été la mise en place du système de connexion pour l'espace professionnel de l'application.

Voici les étapes une par une pour créer ce système :

13.1.2 Création des Champs de Connexion

Pour gagner un peu temps, j'ai utilisé des champs Bootstrap préexistants pour créer les champs de connexion.

Cela m'a permis de concevoir plus rapidement l'interface de connexion administrateur.

13.1.3 Utilisation de password_hash() pour la Sécurité des Mots de Passe

La sécurité sur internet est primordiale, je dois donc faire mon maximum pour que le site ne soit pas piratable.

Je vais donc utiliser la fonction **password_hash()** pour hacher (crypter) les mots de passe des utilisateurs.

Cette fonction génère un hachage sécurisé qui peut être stocké en toute sécurité dans la base de données.

J'ai opté pour l'option `PASSWORD_DEFAULT`, considérée comme je pense d'après mes recherches, la plus sécurisée à ce jour.

J'aurais qu'à plus ajouter un mot de passe qui sera haché automatiquement que j'insérerai dans la Bdd par la suite.

```
○ ○ ○

public function GetPageLogin() {
    require_once(__ROOT__.'\views\login_view.php');

}

public function connexion() {

    echo password_hash("michelaquiche", PASSWORD_DEFAULT);
//    echo "connexion";
}
```

13.1.4 Modification des Tables "Employés" et "Admin"

En voulant supprimer certaines informations inutiles dans les tables "Employés" et "Admin", notamment l'e-mail que j'avais noté auparavant, je vais pour l'instant, conserver uniquement les champs "Login" et "Password".

A ce stade, le login sera généré manuellement, tandis que le mot de passe sera sécurisé par `password_hash()`.

J'ai pensé ensuite, qu'une seule table devrait suffire pour l'espace admin au lieu de créer 2 tables admin et employés, je vais plutôt attribuer des rôles à chacun des administrateurs que je ferai plus tard.

13.1.5 Tests du Système de Connexion

J'ai effectué des tests en appuyant sur le bouton "Valider".

Le système a généré automatiquement un mot de passe sécurisé sur la page de Login, que j'ai pu vérifier en le comparant avec le mot de passe haché stocké dans la base de données.

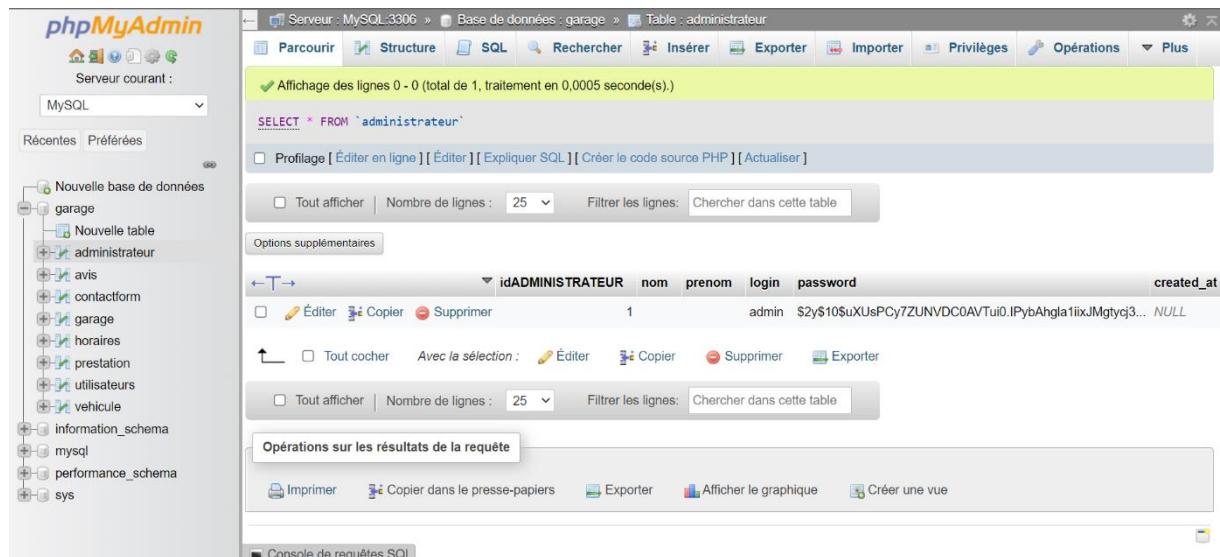
À chaque régénération de la page, un nouveau mot de passe haché sera généré mais mon vrai mot de passe ne changera pas.

Ce mot de passe haché se générera à chaque fois que je cliquerais sur le bouton ce qui ajoutera une sécurité en plus. Dans ces conditions-là, il sera sans doute impossible de le pirater.

13.1.6 Insertion de Données Factices

Pour effectuer des tests plus poussés, j'ai inséré de fausses données dans la table, en copiant le mot de passe généré automatiquement et en l'insérant dans le champ "**password**" et admin pour "Login "de ma table"**Administrateur**".

La mise en place de ce système de connexion est une étape importante qui garantira la sécurité des données sensibles des administrateurs et ainsi respecter les conditions de RGPD que l'on nous impose sous peine de grosses amandes par la suite.



13.1.6 Validation des Tests et Conclusions

En conclusion de cette phase de développement, j'ai réussi à mettre en place un système de gestion de mot de passe sécurisé pour l'espace professionnel de l'application.

Le processus de validation a été concluant, confirmant l'efficacité du hachage de mon mot de passe.

Pour valider le système, j'ai utilisé la fonction `password_hash()` pour hacher un mot de passe spécifique, en l'occurrence "**michelaquiche**".

Le test a été un succès, puisque le mot de passe généré automatiquement correspondait au hachage attendu.

13.1.7 Sécurisation et Vérification des Informations de Connexion

```
public function connexion(){
    if (!empty($_POST["login"]) && !empty($_POST["password"])) {
        $login = Securite::secureHtml($_POST["login"]); //sécurité en lien avec security.class.php
        $password = Securite::secureHtml($_POST["password"]);

        if($this->AdminManager->isConnexionValid($login, $password)) {
            $_SESSION['access'] = "admin"; // Pour activer les variables de session, il va falloir
que je les active en début de page ds index.php
            header('Location: '.URL."back/admin");
            exit();
        } else {
            header('Location: '.URL."back/login");
            exit();
        }
    }
}
```

La sécurité de l'application est très importante, notamment lorsqu'il s'agit de gérer les informations sensibles dont les mots de passe.

Voici les étapes que j'ai suivies pour renforcer la sécurité de la gestion des comptes professionnels

Ajustement de la Longueur des Mots de Passe

Lors de l'insertion des données dans la table, j'ai rencontré une erreur liée à la longueur du mot de passe haché.

En effet, j'ai pensé à mettre assez de place pour un mot de passe ordinaire mais je n'avais pas pensé au cryptage qui est largement plus long.

Pour remédier à cela, j'ai ajusté la longueur du champ VARCHAR, en veillant à ce qu'il soit suffisamment long pour accueillir le hachage dans le champ.

J'ai également constaté que l'utilisation de caractères spéciaux dans le mot de passe pouvait entraîner des problèmes, j'ai donc préféré n'utiliser que des caractères alphanumériques pour éviter d'éventuels problèmes même si je sais que je ne devrais pas le laisser en l'état.

Je m'en occuperai sans doute plus tard si le temps me le permet.

Mise en Place de Vérifications

J'ai créé une page dédiée à la sécurité où je vais vérifier notamment ce qui se passe au niveau des formulaire notamment il vérifiera si les utilisateurs ont bien accès au site. J'ai également converti en HTML les caractères spéciaux pour éviter certains problèmes de sécurité.

J'ai aussi mis en place un système de vérification pour m'assurer que les champs de connexion sont correctement remplis.

J'ai rajouté une sécurité en plus où j'extrais les valeurs soumises pour les champs "login" et "password" que je vais faire passer par une fonction nommée '**secureHtml**', elle effectuera des opérations de nettoyage pour éviter les attaques de sécurité, comme l'injection de code malveillant.

J'ai également prévu de créer un lien de déconnexion qui supprimera la variable de **session** lorsque l'utilisateur se déconnectera.

Je rajouterais une fonction dans le manager qui fera des actions de vérifications de connexion en renvoyant '**true**' ("Authentification réussi") ou false ("Authentification échouée").

Gestion des Sessions

Je vais créer dans le controller des variables de sessions et que si les informations de connexion sont correctes (vérifiées en utilisant des identifiants préalablement créés), une session est générée, et l'utilisateur est redirigé vers la page administrative, en n'oubliant pas de le déclarer en début de la page **index.php** pour que les pages puissent par la suite communiquer entre elles.



Validation des Informations de Connexion

J'ai développé une fonction de vérification des informations de connexion pour m'assurer que l'utilisateur a rempli correctement les champs requis.

Cela garantit également que lors de la déconnexion, l'accès à la page d'administration est désactivé donc quitté la Session.

```

public function connexion(){
    if (!empty($_POST["login"]) && !empty($_POST["password"])) {
        $login = Securite::secureHtml($_POST["login"]); //sécurité en lien avec security.class.php
        $password = Securite::secureHtml($_POST["password"]);

        if($this->AdminManager->isConnexionValid($login, $password)) {
            $_SESSION['access'] = "admin"; // Pour activer les variables de session, il va falloir
que je les active en début de page ds index.php
            header('Location: '.URL."back/admin");
            exit();
        } else {
            header('Location: '.URL."back/login");
            exit();
        }
    }
}

```

Mise en Place de la ‘Navbar’ Admin

J'ai préparé une navbar pour l'administration, présentant toutes les fonctionnalités autorisées sous forme de menus déroulants.

La navbar est visible uniquement lorsque l'administrateur est connecté.

Pour ce faire, j'ai intégré du code PHP dans le code HTML existant.

Résolution des Problèmes de Connexion

J'ai rencontré des problèmes de connexion, mais en examinant attentivement le code, j'ai identifié une erreur de syntaxe avec les instructions `require_once`.

Après correction, la connexion a fonctionné correctement.

Ces étapes ont garanti que les informations de connexion soient bien sécurisées et que seuls les utilisateurs autorisés auront accès à la page d'administration

Garrage Parrot - Espace Pro Connexion

Connexion Espace Pro

Identifiant

Mot De Passe

Valider

14.0 Tests des données factices dans la table administrateur et Affichage des Données

Voici comment j'ai procédé pour afficher la table "Véhicule" dans l'espace professionnel :

```
● ● ●

class Model {
    private static $pdo;

    private static function setBdd(){
        self::$pdo = new PDO("mysql:host=localhost;dbname=garage;charset=utf8","root","");
        self::$pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_WARNING);
    }

    protected function getBdd(){
        if(self::$pdo === null){
            self::setBdd();
        }
        return self::$pdo;
    }

    public static function sendJSON($info){
        header("Access-Control-Allow-Origin: *"); /site sera en ligne
        header("Content-Type: application/json");
        echo json_encode($info);
    }
}
```

14.1 Correction des Erreurs de Connexion

Tout d'abord, j'ai résolu les problèmes de connexion qui empêchaient l'accès à la page. Une fois la connexion réussie, les données se sont affichées.

Pour une meilleure lisibilité, je vais faire en sorte de les afficher sous forme de tableau car assez compliqué de s'y retrouver surtout s'il y a des centaines de données à afficher. Je vais utiliser un tableau Bootstrap pour plus de simplicité.

14.2 Tests de Données

Pour vérifier que tout fonctionnait correctement, j'ai ajouté des données factices à la table et j'ai effectué des tests en utilisant le contrôleur correspondant.

Cela m'a permis de m'assurer que les données étaient correctement récupérées depuis la base de données.

Ces tests ont été fait pour chaque table.

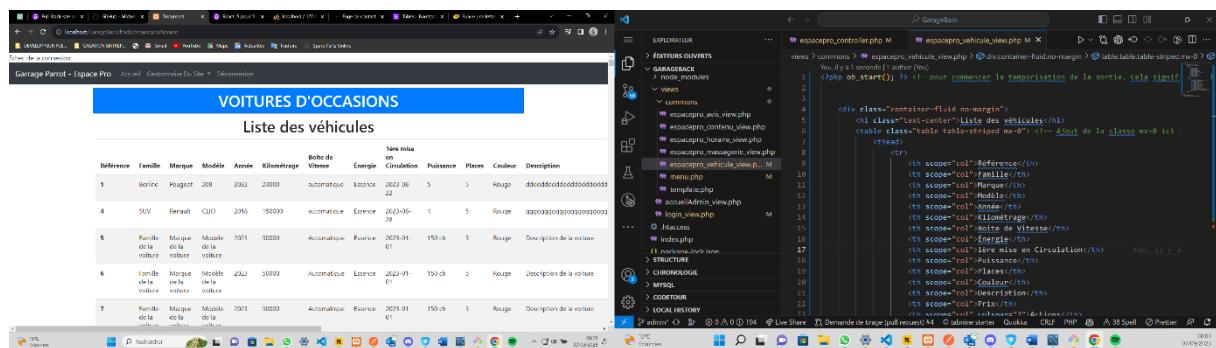
14.3 Mise en Forme des Données

Pour améliorer la présentation des données, j'ai utilisé une table Bootstrap prête à l'emploi. Cela m'a permis de gagner du temps tout en offrant une interface utilisateur plus conviviale.

J'ai également préparé des boutons pour les opérations de base et je vais ajouter les CRUD (**Create, Read, Update, Delete**) sous format date actuelle et non américain.

L'affichage des données dans l'espace professionnel va pouvoir fournir aux utilisateurs un accès rapide et efficace aux informations dont ils ont besoin.

Les prochaines étapes consisteront à mettre en place les fonctionnalités de modification et de suppression.



15.0 Implémentation du Bouton de Suppression

J'ai créé le bouton de suppression pour garantir la sécurité des données et éviter toute suppression accidentelle. Voici comment j'ai mis en place ce mécanisme dans l'espace professionnel :

15.1 Crédation d'une Nouvelle Route

Tout d'abord, j'ai créé une nouvelle route pour gérer l'action de suppression. Cette route permettra de transmettre l'identifiant du véhicule à supprimer depuis l'URL.

15.2 Conversion de l'ID en Entier

Étant donné que l'URL transmet l'identifiant en tant que chaîne de caractères, j'ai dû convertir cette chaîne en entier dans le contrôleur correspondant. Cela a permis d'éviter les erreurs et les problèmes de sécurité potentiels.

15.3 Mise en Place d'Alertes JavaScript

J'ai préparé des alertes en JavaScript pour confirmer l'action de suppression.

Ces alertes sont conçues pour n'apparaître qu'une seule fois par page, et j'ai veillé à supprimer la variable de session associée une fois que le message a été affiché.

15.4 Gestion des Redirections

Cependant, j'ai rencontré un problème de redirection après la suppression. Bien que le chemin de redirection soit correct, j'ai dû le commenter temporairement pour résoudre le problème.

La mise en place de ce mécanisme de suppression sécurisé que j'ai effectué servira à protéger les données de l'application et garantir que les opérations de suppression sont effectuées de manière intentionnelle.

16.0 Bouton modifier NON OPERATIONNEL

Dans la vue, je vais créer des champs de saisie qui vont s'afficher sous la ligne de l'objet à modifier pour chaque attribut de la table "véhicule", permettant ainsi la modification de tous les détails, à l'exception de l'identifiant (ID) bien sûr. Par la suite, un bouton "Valider" sera ajouté.

Ensuite, je vais m'occuper de la partie côté serveur lorsque l'utilisateur appuiera sur le bouton "Valider" par rapport aux nouvelles informations insérées.

Difficultés rencontrées

À deux reprises, j'ai rencontré des problèmes d'accès au serveur SQL, où l'accès m'a été complètement refusé sans raison apparente, malgré les tests que j'avais effectués.

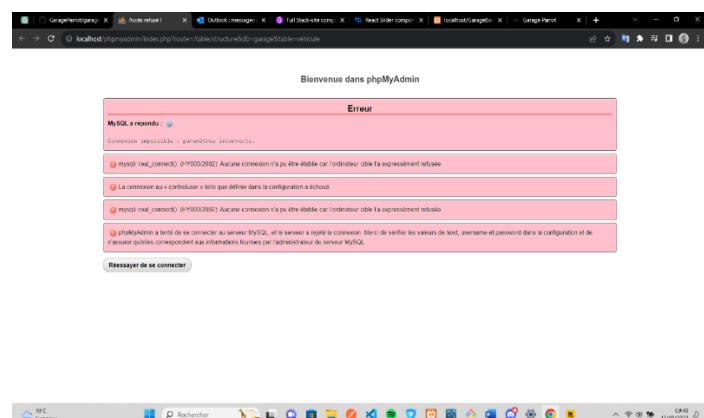
J'ai dû réinstaller XAMPP encore une fois, mais le problème est que j'ai perdu mes données de table, bien que j'aie sauvegardé le dossier SQL auparavant.

Il doit exister une solution pour restaurer ces données, donc je vais effectuer des recherches à ce sujet.

Par la suite, j'ai appris à les sauvegarder en exportant la Bdd et me le renvoi en **nonDuFichier.sql** mais je devrais par la suite le compresser en fichier ZIP pour pouvoir de nouveau l'importer.

J'ai aussi un problème mineur que je n'ai pas encore trouvé, il faut que je clique 2 fois sur les boutons modifier, supprimé et déconnexion pour que l'opération s'effectue, peut être un problème de rafraîchissement de page, à suivre...

(J'ai trouvé plus tard le problème, il s'agissait d'un inversement de codes qui n'étaient pas à la bonne place).



Je me suis retrouvé dans une impasse après la réinstallation de XAMPP, avec différents problèmes.

J'ai souhaité générer un nouveau mot de passe pour l'insérer dans la base de données, mais j'ai rencontré des difficultés et des messages d'erreur liés à un ancien code du contrôleur qui n'existe pas.

J'ai vidé le cache, redémarré XAMPP, isolé tout le reste de mon code, mais rien n'a changé.

En cherchant des solutions, j'ai découvert que je pouvais consulter d'autres messages d'erreur dans un fichier journal (log) de XAMPP, en désactivant d'abord Apache pour éviter de me tromper et ainsi cibler le message d'erreur spécifique.

Cependant, les avertissements (warnings) que j'ai trouvés ne semblaient concerner que le certificat SSL, qui doit être en HTTPS, mais étant donné que je travaille en local, cela ne me concernait pas directement.

Par la suite, mon ordinateur a subi une panne et j'ai dû tout réinstaller, puis récupérer mon travail en utilisant "git clone".

J'ai exporté à nouveau mes tables vers la base de données et généré un nouveau mot de passe que j'ai inséré dans la base de données car je ne savais pas encore comment enregistrer ma Bdd. Cependant, lors de la saisie du nom d'utilisateur (login) et du mot de passe, il était impossible de se connecter.

J'ai mis 6 jours à identifier le problème. Dans le gestionnaire (manager), il était noté "login" et "password", tandis que dans la base de données, c'était écrit "login" et "mot_de_passe", d'où l'erreur. C'était comme chercher une aiguille dans une botte de foin.

Après plusieurs bugs de Xampp, je suis passé à Wamp et j'ai cette fois-ci bien enregistré ma Bdd au format Zip au cas où j'aurais d'autres problèmes de fichiers corrompus.

17.0 Bouton création d'un véhicule

En ce qui concerne le view, et pour une meilleure expérience utilisateur, j'ai préparé des propositions par rapport aux caractéristiques de la voiture sous forme de dropdown sauf pour la famille des véhicules qui seront en checkboxs pour mettre un peu de piquant, je mettrai une règle pour que l'admin puisse uniquement cocher une seule case pour éviter certaines erreurs.



```
<div class="form-check">
    <input type="radio" id="utilitaire" name="famille" value="Utilitaire" class="form-check-input">
    <label for="utilitaire" class="form-check-label">Utilitaire</label>
</div>
```

J'ai récemment créé une nouvelle route et ajouté un modèle Bootstrap à mon application, en incorporant des cellules de remplissage pour chaque caractéristique du véhicule.

J'ai choisi de ne pas insérer manuellement l'ID lors de l'ajout d'un véhicule, car j'ai configuré la base de données pour qu'elle utilise l'auto incrémentation, ce qui simplifiera le processus.

Pour gérer cette fonctionnalité, j'ai adapté mon gestionnaire (manager) en utilisant une approche similaire à celle que j'avais employé pour la modification.

Cette fois-ci, j'ai élaboré la commande SQL `INSERT INTO` pour l'ajout des données.

Pour obtenir l'ID généré par la base de données, j'ai fait appel à la fonction ``lastInsertId``, qui est une fonctionnalité de l'extension PDO.

```
return $this->getBdd()->lastInsertId();
```

Ensuite, j'ai pris soin de transmettre cet ID au contrôleur responsable de la récupération du nouvel ID.

Pour garantir une expérience utilisateur fluide, j'ai ajouté un message de confirmation à l'intention de l'administrateur, l'informant du nouvel ID du véhicule.

```
$_SESSION['alert'] = [
    "message" => "Le véhicule a bien été créé sous l'identifiant : " .
$idVehicule,    "type" => "alert-success"
];
```



```
public function createVehicule($imageVoiture, $famille, $marque, $modele, $annee,
    $kilometrage, $boitevitesse, $energie, $datecirculation,
    $puissance, $places, $couleur, $description, $prix, $imageCritere, $created_at)
{
    $req = "INSERT INTO vehicule (imageVoiture, famille, marque, modele, annee, kilometrage,
        boitevitesse, energie, datecirculation, puissance, places, couleur, description, prix,
        imageCritere, created_at)
        VALUES (:imageVoiture, :famille, :marque, :modele, :annee, :kilometrage, :boitevitesse,
        :energie, :datecirculation, :puissance, :places, :couleur, :description, :prix, :imageCritere,
        :created_at);
```

Cependant, j'ai rencontré un problème lors de la saisie des informations dans les champs de formulaire. Les données étaient bien enregistrées dans la base de données, mais au lieu de mes entrées, elles étaient enregistrées sous la forme de '**000000**'.

La source du problème résidait dans l'utilisation de la fonction `\Securite::secureHTML`, que j'ai dû temporairement désactiver pour résoudre ce problème.

Bien que cette désactivation ait permis de corriger l'anomalie, je suis conscient que cela pourrait potentiellement compromettre la sécurité de mon code.

Je cherche actuellement une solution plus sécurisée pour résoudre ce problème tout en maintenant la protection de mon application."

Malheureusement, même après avoir retiré l'utilisation de cette sécurité, mon bouton de modification continue de ne pas fonctionner.

Cela signifie que la désactivation de cette fonction n'est pas la cause du problème.

Comme la sécurité est importé pour mon application, j'envisage plus tard de trouver des solutions alternatives pour résoudre ce problème tout en préservant la sécurité de mon code.

Ce passage a été marqué par pas mal de défis inattendus et des problèmes techniques mais je pense avoir assez bien géré sur ce coup-là.

Ajouter un Véhicule

Image Voiture

Aucun fichier choisi

Famille

- Utilitaire
- Berline
- Familiale
- Citadine
- SUV

marque

citroen

Modèle

18.0 Ajout d'images lors de la création

Dans la vue, je vais modifier le type de champ en "file" pour permettre le téléchargement d'image.

Ensuite, je vais créer un nouveau fichier de contrôleur que je nommerai "[regles_utiles.php](#)".

C'est dans ce fichier que je vais définir les règles de téléchargement, telles que la taille et le format des images qui devront être respectées, je fais cela pour éviter d'alourdir le site et ainsi le ralentir, ce qui pourrait faire fuir les utilisateurs si la page met trop longtemps à charger.



```
function ajoutImage($file, $dir){
    if(!isset($file['name']) || empty($file['name']))//vérification si l image a bien été saisie
        throw new Exception("Vous devez indiquer une image");

    if(!file_exists($dir)) mkdir($dir,0777);//Si pas de répertoire de crée alors il va en créer un sur
    le serveur

    $extension = strtolower(pathinfo($file['name'],PATHINFO_EXTENSION));
    $random = rand(0,99999); //on va générer un nombre aléatoire pour donner un nom unique au fichier.
    $target_file = $dir.$random."_".$file['name'];

    if(!getimagesize($file["tmp_name"]))//vérifier que le fichier est bien une image
        throw new Exception("Le fichier n'est pas une image");
    //Vérification de la bonne extension
    if($extension !== "jpg" && $extension !== "jpeg" && $extension !== "png" && $extension !== "gif")
        throw new Exception("L'extension du fichier n'est pas reconnu");
    if(file_exists($target_file))
        throw new Exception("Le fichier existe déjà");
    if($file['size'] > 900000)//De préférence, mettre 500000
        throw new Exception("Le fichier est trop gros");
    if(!move_uploaded_file($file['tmp_name'], $target_file))
        throw new Exception("l'ajout de l'image n'a pas fonctionné");
    else return ($random."_".$file['name']);
}
```

Je vais aussi générer un nombre aléatoire pour m'assurer que le nom du fichier téléchargé sera unique.

Je vais également apporter quelques modifications au fichier "**"espacepro_controller.php"** pour lier ces règles que j'ai définies en utilisant des fonctions appropriées.

Je n'oublierai pas de spécifier le répertoire dans lequel je souhaite stocker les images téléchargées.

19.0 Suppression d'images :

Lors de la suppression d'un identifiant de véhicule, il est nécessaire que je prenne des mesures pour supprimer les images associées qui sont encore présentes dans le répertoire que j'ai sélectionné.

Pour accomplir cette tâche, je vais ajouter quelques lignes de code supplémentaires à mon contrôleur de suppression dans l'espace professionnel et utiliser la fonction **`unlink()`**.

Cela permettra de nettoyer efficacement les fichiers image associés au véhicule supprimé et ainsi alléger le site.



```
public function getImageVoiture($idVehicule){  
    $req = "SELECT imageVoiture from vehicule where idVehicule = :idVehicule";  
    $stmt = $this->getBdd()->prepare($req);  
    $stmt->bindValue(":idVehicule",$idVehicule,PDO::PARAM_INT);  
    $stmt->execute();  
    $image = $stmt->fetch(PDO::FETCH_ASSOC);  
    $stmt->closeCursor();  
    return $image['imageVoiture'];  
}  
  
public function getImageCritere($idVehicule){  
    $req = "SELECT imageCritere from vehicule where idVehicule = :idVehicule";  
    $stmt = $this->getBdd()->prepare($req);  
    $stmt->bindValue(":idVehicule",$idVehicule,PDO::PARAM_INT);  
    $stmt->execute();  
    $image = $stmt->fetch(PDO::FETCH_ASSOC);  
    $stmt->closeCursor();  
    return $image['imageCritere'];  
}
```

RESULTAT

1^{ère} partie :

Référence	Image Véhicule	Famille	Marque	Modèle	Année	Kilométrage	Boîte de Vitesse	Énergie	1 ^{ère} mise en Circulation	Puissance
128		Berline	citroen	2222	2010	222222	manuel	essence	0000-00-00	5
129		Citadine	citroen	20008	2015	120000	automatique	diesel	0000-00-00	4
130		Berline	citroen	gg	2019	111111	manuel	essence	0000-00-00	5

2ème partie :

Puissance	Places	Couleur	Description	Prix	Image Critère	Actions
5	5	blanc	<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua </p>	12345.00	 Emissions de CO ₂ faible	Modifier Supprimer
4	6	vert	<p> Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua </p>	13500.00	 Emissions de CO ₂ faible	Modifier Supprimer
5	4	bleu	DDDDDDDDDDDDDDDDDDDDDDDDDD	12000.00	 Emissions de CO ₂ faible	Modifier Supprimer

20.0 Gestion des dates

J'ai identifié deux problèmes liés à la gestion des dates :

1. Le format de la date (0000-00-00) n'était pas adéquat, j'ai dû le convertir au format européen (00-00-0000). Pour ce faire, j'ai effectué quelques modifications du côté du gestionnaire (manager) afin de garantir la conversion correcte de la date au format souhaité.

2. Une fois que le format de la date était corrigé, je me suis heurté à un deuxième problème : l'incapacité à enregistrer la date dans la base de données car elle sera toujours au format 00-00-000.

La solution était, lors du traitement du formulaire côté serveur, j'ai récupéré la date au format '**d-m-y**' et je l'ai préalablement formatée en '**y-m-d**' avant de la stocker dans la base de données.

Problèmes à résoudre ultérieurement : prévisualisation des images en création + Impossibilité de modifier un véhicule

21.0 Mise en place de l'API ‘véhicule’ par rapport à la recherche par filtres

Je vais utiliser l'architecture du modèle MVC en continuité de la logique de mon code.

vehicule_controller.php

Je vais créer le contrôleur qui va gérer les requêtes des filtres pour récupérer les données de la table véhicule à partir du modèle '**vehiculeModel**' .

A ce contrôleur, je vais lui créer une classe '**VehiculeModel**' dans la propriété '**\$apiManager**'.



```
public function __construct() {  
    $this->espaceproManager = new EspaceproManager();  
}
```

J'ai ajouté à cette méthode un constructeur de classe. Je vais utiliser cette méthode pour initialiser chaque objet lorsqu'une instance de cette classe sera créée.

Elle sera appelée automatiquement à chaque fois que je créerais un nouvel objet, ici les filtres, à partir de cette classe.

J'ai ajouté **\$this** qui est un mot clé en Php et qui fait référence à l'objet de lui-même (instance de classe dans laquelle le code est en cours d'exécution).

Il s'agit donc d'un opérateur de sélection de propriété et qui va accéder à chaque propriété de l'objet filtre.



```
public function getCarsByFilters($filtres)
```

Cette instance sera utilisée pour interagir avec **le modèle de données**.

Ensuite, je vais définir une méthode pour récupérer les véhicules en fonction des filtres passés en paramètre et après, je ferais appel au modèle pour récupérer les données.

Pour finaliser le paramétrage du contrôleur, je vais configurer les en-têtes http avec '**header()**' pour spécifier le type de contenu **JSON** et permettre les requêtes en spécifiant les domaines autorisés comme les méthodes acceptées ou les en-têtes.



```
//Configurez les entêtes avant d'envoyer la réponse
header('Content-Type: application/json');
header("Access-Control-Allow-Origin: http://localhost:3000");
header("Access-Control-Allow-Methods: GET, POST, OPTIONS");
header("Access-Control-Allow-Headers: Content-Type");
```

Header() est une fonction en Php qui permet de contrôler les en-têtes http comme expliqué ci-dessus pour pouvoir personnaliser la manière dont le serveur communique avec le navigateur et permettre ainsi d'effectuer **des redirections, définir des cookies, spécifier le type de contenu comme ici du JSON** et apparemment encore beaucoup d'autres choses mais que je ne suis pas encore assez expérimenté pour en dire plus.

22.0 API vehicules.php

C'est dans ce code, que je vais gérer les requêtes http entrantes en fonction de la méthode qui sera ici en **GET** pour pouvoir récupérer les **paramètres sous format Json** fournis dans l'**Url**.

Je vais vérifier que si la méthode de la requête est GET, je ferais en sorte de continuer à traiter la requête.

Je vais faire vérifier les paramètres que j'ai effectué sur mon système de recherche par filtre et que je stockerais dans un tableau nommé \$filters, s'ils sont bien sûr définis par l'utilisateur dans la requête.



```
$filters = array();

if (isset($_GET['famille'])) {
    $filters["famille"] = $_GET['famille'];
}

if (isset($_GET['marque'])) {
    $filters["marque"] = $_GET['marque'];
}

if (isset($_GET['kilometremin'])) {
    $filters['kilometremin'] =
}ntval($_GET['kilometremin']);
```

Ensuite je vais créer une instance de classe ‘**VehiculeController**’ et je vais appeler la méthode ‘**getCarsByFilters(\$filters)**’ sur cette instance.

Cela me permettra d’interagir entre le contrôleur des véhicules pour récupérer toute leurs données en fonction des filtres choisi par l’utilisateur.



```
$controller = new VehiculeController();
$controller->getCarsByFilters($filters);
```

Je vais utiliser la fonction de php ‘**intval()**’ pour pouvoir convertir une valeur en un entier que je stockerais ensuite dans une variable, cela me permettra de m’assurer qu’elle contienne uniquement une valeur numérique en ignorant tout autre caractère non numérique présent dans la chaîne d’origine.



```
if (isset($_GET['kilometremin'])) {  
    $filters['kilometremin'] =  
    intval($_GET['kilometremin']);
```

Cela me permettra de traiter les données provenant d'une source externe pour assurer que la valeur entrée ici, par la recherche par filtre, sera interprété uniquement comme un entier.

Comme pour le reste de mon projet, en ajoutant une condition, s'il y a une erreur comme une méthode qui n'est pas **GET**, il renverra une **réponse d'erreur 404**.



```
http_response_code(404);  
echo json_encode([ "error" => "endpoint not found" ]);
```

23.0 vehicule_model.php

Je vais créer la classe '**VehiculeModel**' qui va être utilisée pour effectuer des recherches de véhicules dans la Bdd.

Dans le constructeur de la classe, je vais préparer la connexion à la Bdd en utilisant le **PDO** de Php. J'ai défini les identifiants de connexion pour avoir accès à la Bdd.

Je définie les infos de connexion dans des variables et si une erreur de connexion se passe, j'afficherais un message d'erreur.

```

public function __construct()
{
    $dsn = 'mysql:host=localhost;dbname=garage;charset=utf8';
    $user = 'root';
    $password = '';

    try {
        // Connexion à la base de données
        $this->dbh = new PDO($dsn, $user, $password);
    } catch (PDOException $e) {
        die('Erreur de connexion : ' . $e->getMessage());
    }
}

```

Je vais créer une autre classe ‘**getCarsByFilters**’ qui sera un **tableau associatif** contenant les filtres pour la recherche de véhicules.

La méthode va construire une requête Sql de base qui va sélectionner toutes les colonnes de la table ‘**vehicule**’ avec une condition **WHERE 1**, cela signifie que la requête devra toujours être vrai même s’il n’y a pas de filtres spécifiés.

```
$sql = "SELECT * FROM vehicule WHERE 1";
```

En fonction des filtres choisi par l’utilisateur dans le tableau **\$filters**, la méthode va ajouter des conditions supplémentaires à la requête Sql.

Je vais donner un exemple, si le filtre ‘**famille**’ est choisi, la méthode ajoutera une clause ‘**AND famille = :famille**’ à la requête et ainsi de suite pour les autres filtres.

La fonction **explode()** va diviser la valeur du filtre en un tableau en utilisant la virgule comme séparateur.

Elles seront ensuite combiné en une chaîne unique séparée par des virgules à l'aide de la fonction **implode()**.

Str_replace va supprimer les virgules de chaque valeur et va les transformer en chaîne de caractère et la valeur sera ensuite concaténée.



```
if (isset($filters['famille'])) {les AND 1 par 1

$values = explode(",", $filters['famille']);

$namedPlaceholders = implode(' ', array_map(function ($value) {
    return ':value_' . str_replace(',', '', $value);
}, $values));

$sql .= " AND famille IN ($namedPlaceholders)";
```

J'ai ajouté un filtre limite qui permettra de limiter le nombre de résultats retournés pour éviter que toutes les données soient retournées d'un seul coup et ne serait pas joli visuellement même si je l'ai déjà aussi paramétré du côté React en limitant le nombre d'objet à afficher et aussi en incluant une pagination.

Au départ, j'ai préparé la requête Sql en liant les valeurs des filtres aux paramètres de la requête en utilisant **bindParam()** qui est une méthode de classe de PDO.



```
if (isset($filters['marque'])) {
    $sql .= " AND marque = :marque";
}

if (isset($filters['kilometremin'])) {
    $sql .= " AND kilometrage >= :kilometremin";
}
```

J'avais un doute si je devais utiliser la fonction **bindParam()** ou **bindValue()** car apparemment, ces 2 fonctions sont assez similaires.

Après quelques recherches, j'ai décidé d'utiliser plutôt la fonction **bindParam()** car elle permet de lier une variable par référence, ce qui signifie que toute modifications de filtres apportés à la variable « **filtres** » affectera automatiquement à la requête préparée.

Dans les 2 cas, je pense que cela aurait sans doute fonctionnée car le plus important, c'est de bien spécifier le type de données des champs qui sont dans la table de la Bdd pour éviter les certaines erreurs.

Je vais ensuite exécuter la requête en récupérant les résultats sous forme de tableau associatif en utilisant **fetchAll()** qui est aussi une classe de PDO.

Je vais par la suite stocker ces résultats dans une variable et renvoyer la méthode.

24.0 Avis Clients

J'ai utilisé le modèle MVC comme référence pour la gestion des avis des clients dans la section réservée aux professionnels.

J'ai mis en place un mécanisme où, par défaut, le champ "valide" dans la table des avis est initialisé à "O", ce qui indique que l'avis est en attente de validation.

Lorsque l'administrateur valide un avis, son état passe automatiquement à 1, ce qui signifie qu'il est validé.

J'ai également ajouté une fonctionnalité supplémentaire sous la forme d'un bouton de validation.

Cela donne à l'administrateur le choix de mettre en ligne un avis ou non. Une fois qu'un avis est validé, le bouton de validation disparaît pour améliorer l'expérience utilisateur dans cet espace.

							Supprimer
5	Müller	Hans	4	Sehr zufrieden	29-10-2023	Validé	Modifier Supprimer
18	Martin	Sophie	5	Service exceptionnel ! Le personnel était très professionnel et ma voiture est comme neuve.	28-10-2023	Validé	Modifier Supprimer
19	Tremblay	Pierre	2	Très déçu de l'expérience. La réparation a pris beaucoup plus de temps que prévu et le coût était élevé.	27-10-2023	Non Validé	Modifier Supprimer Valider
20	Dubois	Marie	4	J'ai été impressionnée par la qualité du service. Ils ont réparé ma voiture rapidement et à un prix raisonnable.	26-10-2023	Non Validé	Modifier Supprimer Valider

25.0 Prestation du garage

La seule difficulté était du côté de la création de l'API, cela, m'a fait planter tout mon back end. J'ai dû faire quelques modifications au niveau du MVC. Le problème est résolu.

Liste des prestations

Référence	Image prestation	Nom	Description	Prix à partir de	Date de création	Date de mise à jour	Actions
4		amortisseurs	Lorem ipsum dolor sit amet. Quo officia corporis aut voluptatem ratione et deserunt rerum. Nam laudantium distinctio id debitis culpa non blanditiis deleniti.	120.00	29-10-2023		Modifier Supprimer
6		embrayage	Lorem ipsum dolor sit amet. Quo officia corporis aut voluptatem ratione et deserunt rerum. Nam laudantium distinctio id debitis culpa non blanditiis deleniti.	200.00	29-10-2023		Modifier Supprimer

26.0 Conclusion

Dans la conception de l'interface en HTML, j'ai élaboré un tableau destiné à afficher la liste complète des véhicules.

J'ai aussi implémenté une boucle "**foreach**" pour itérer à travers les données du tableau, permettant ainsi l'affichage détaillé de chaque véhicule.

Ici, je parle de la table véhicule mais dès que j'ai trouvé la solution au départ comment savoir gérer les données, j'ai pu facilement le reproduire sur les autres tables.

De plus, j'ai associé les fonctionnalités de modification et de suppression à des formulaires HTML dédiés.

Ces formulaires fournissent une interface utilisateur intuitive pour gérer les opérations de mise à jour et de suppression des véhicules de manière efficace.

Pour renforcer la sécurité et éviter toute suppression accidentelle, j'ai intégré une demande de confirmation en JavaScript pour le bouton "Supprimer".

Cette mesure préventive assure une expérience utilisateur plus fiable.

J'ai mis en place une condition pour vérifier si un formulaire de modification a été soumis pour un véhicule spécifique.

En cas de soumission, un formulaire de modification est généré pour le véhicule concerné, avec des champs préremplis (inputs) facilitant ainsi la modification des détails mais qui ne fonctionne pas à ce jour.

Je pense que cette approche permet une gestion fluide et sécurisée des véhicules au sein de l'application, tout en offrant une expérience utilisateur optimale, enfin, je l'espère.



GAMME PEUGEOT
A PARTIR DE **150 € /MOIS**
APRÈS UN 1^{ER} LOYER DE 4 200€⁽¹⁾

SECTION 1

Pour les trajets courts, priviliez la marche ou le vélo. #SeDéplacerMoinsPolluer

GARAGE PARROT

Nous vous rachetons votre voiture !

APPElez NOUS ! 04-70-40-78-98 Espace PRO Connexion

Pneumatique Freinage Mécanique Entretien Pare-Brise Vidange Voitures d'Occasion Nous Contacter

GARAGE PARROT

Arrivage De Nouvelles
Voitures D'Occasion
SECTION 2



Découvrez nos prestations

Les réparateurs s'occupent de la réparation et l'entretien de votre voiture, peu importe la marque ou le modèle de celle-ci. Profitez d'une prestation de qualité effectuée par des véritables experts auto. Contactez nous dès maintenant pour un devis en ligne pour la réparation de votre voiture et obtenez un RDV immédiat !

Vitrage et par-brise

Si vous constatez sur place une fissure votre pare-brise il est temps de faire rendre chez votre carrossier afin de lui faire observer votre véhicule.

Nous Contacter

A partir de 230,00 €

Vitrage et par-brise

Si vous constatez sur place une fissure votre pare-brise il est temps de faire rendre chez votre carrossier afin de lui faire observer votre véhicule.

Nous Contacter

A partir de 110,00 €

Vitrage et par-brise

Si vous constatez sur place une fissure votre pare-brise il est temps de faire rendre chez votre carrossier afin de lui faire observer votre véhicule.

Nous Contacter

A partir de 350,00 €

SECTION 3

Nos nouveautés du mois peuvent vous intéresser

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

Occasion SPOTICAR

Peugeot 2008
Pure tech 100

Essence 12 500 Km 2020

12 600 € **Voir**

SECTION 4

1 - 2 - 3 - 4 ...



SECTION 5

Avis Clients



Service client au top.
Je viens depuis des années
et toujours été satisfait

Michel Hoffmann

Avis Clients



Nous vous rachetons votre voiture !

APPELEZ-NOUS !
04-70-40-78-98

ESPACE PRO
Connexion

Pneumatique Freinage Mécanique Entretien Par-Brise Vidange Voitures d'Occasion Nous Contacter

GARAGE PARROT

Arrivage De Nouvelles
Voitures D'Occasion



1er distributeur automobile



Véhicules certifiés et garantis jusqu'à 24 mois



Satisfait ou remboursé 14 jours



Service client du lundi au samedi de 9h00 à 19h00

Sélection par filtres



Utilitaire



Berline



Familiale



Citadine



Suv

Marque / Modèle

Toutes les marques ▾

Prix

5000 50 000 2000 2023

Tous les modèles ▾

Kilométrage

0 200 000

Rechercher

Résultats Recherche



COMMANDES GENERALES GIT UTILISEES POUR LES PROJETS



Ce rapport présente les principaux concepts et commandes liés à l'utilisation de Git, un système de gestion de version distribué, pour le suivi et la gestion des modifications de code source dans un projet.

1. Création d'un Projet et Communication avec GitHub :

- Pour commencer, un projet est créé sur GitHub, une plateforme d'hébergement de code en ligne.
- Les fichiers du projet sont copiés localement en utilisant Git, établissant ainsi une connexion entre le référentiel local et GitHub.

2. Gestion des Branches :

- Git permet une gestion flexible des branches, que ce soit en local ou à distance.
- Les branches sont créées localement avec la commande `git branch` et on bascule sur une branche spécifique avec `git checkout nom-de-la-branche`.
- Utilisation de `git pull` pour récupérer les dernières modifications d'une branche à distance.

3. Fusion de Branches :

- Pour fusionner une branche avec la branche principale (par exemple `main`), on suit ces étapes :
 1. Se positionner sur la branche à fusionner avec `git checkout nom-de-la-branche`.
 2. Mettre à jour la branche principale avec `git pull origin main`.

3. Utiliser `git merge` pour fusionner les modifications de la branche en étant sur la branche principale.

4. En cas de conflits, ajouter les fichiers modifiés avec `git add`, puis valider les modifications avec `git commit`.

4. Revenir en Arrière après une Fusion Incorrecte :

- Si une fusion a été effectuée incorrectement, on peut revenir en arrière avec `git reset --hard HEAD^`.

5 .Suppression des Branches :

- Suppression d'une branche en local : `git branch -d nom-de-la-branche`.
- Suppression d'une branche à distance : `git push origin --delete nom-de-la-branche`.

6.Suivi des Modifications :

- Utilisation de `git log` pour visualiser toutes les opérations effectuées uniquement sur la branche courante.
- Utilisation de `git branch -a` pour voir toutes les branches locales et à distance.

7.Enregistrement des Modifications :

- `git add .` pour préparer les modifications à être incluses dans le prochain commit.
- `git commit -m "message de commit"` pour enregistrer les modifications avec un message descriptif.
- `git push` pour envoyer les modifications vers GitHub.

En résumé, Git est un outil puissant pour la gestion de version, permettant le suivi efficace des modifications, la collaboration en équipe et le contrôle des branches de développement. L'utilisation de commandes spécifiques permet de maintenir un historique clair et de gérer les modifications de manière méthodique dans un projet de développement logiciel malgré plusieurs frayeurs au départ suite à différents conflits. Que j'ai eu du mal à gérer.

