



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)

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 PROJET

HOFFMANN MICHEL



DEVELOPPEUR WEB FULL STACK



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)

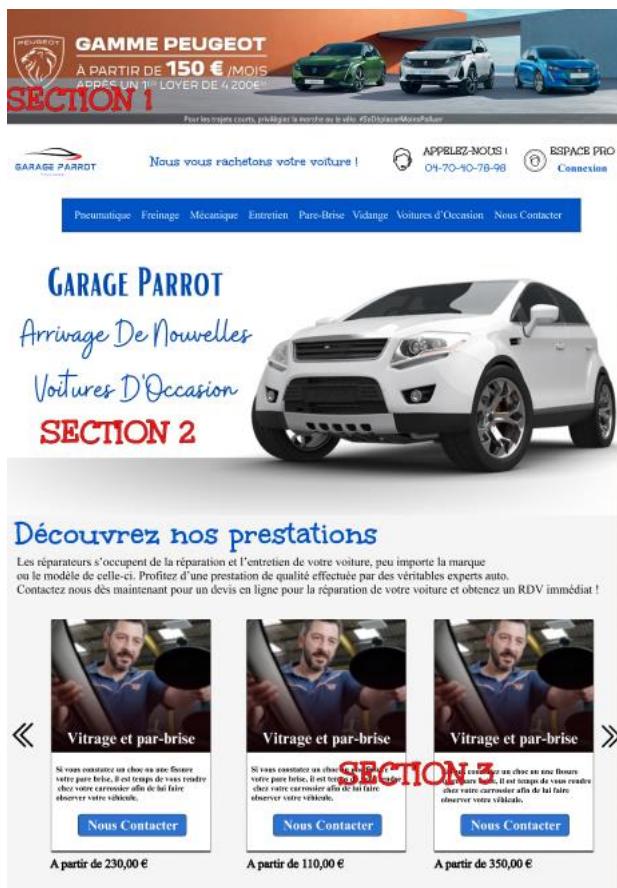
Exemple n°1 ▶ PREPARATION DU PROJET (cahier des charges, maquette...)

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.

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.

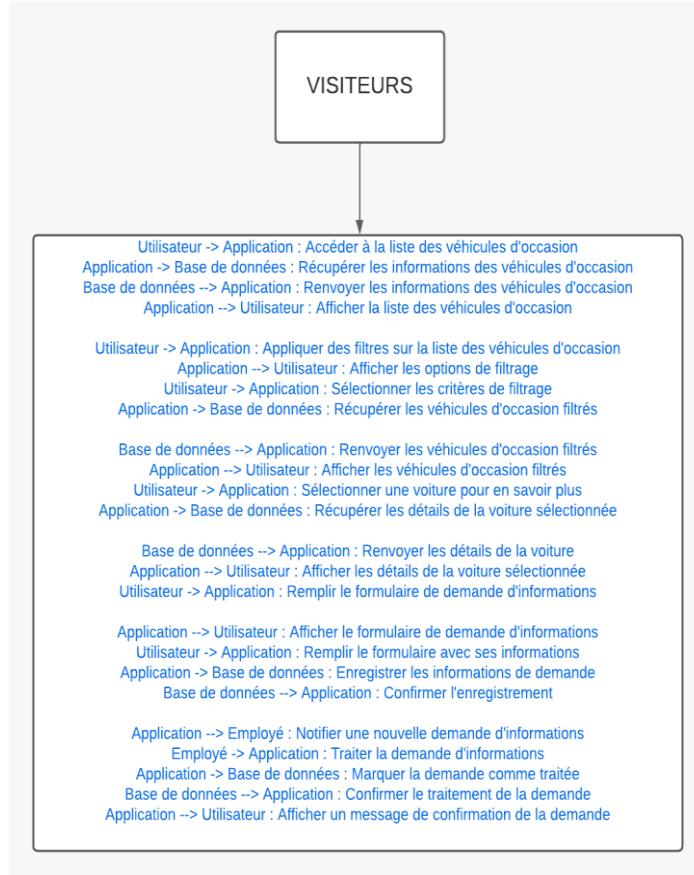
1.0 Diagrammes de Séquence

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 :

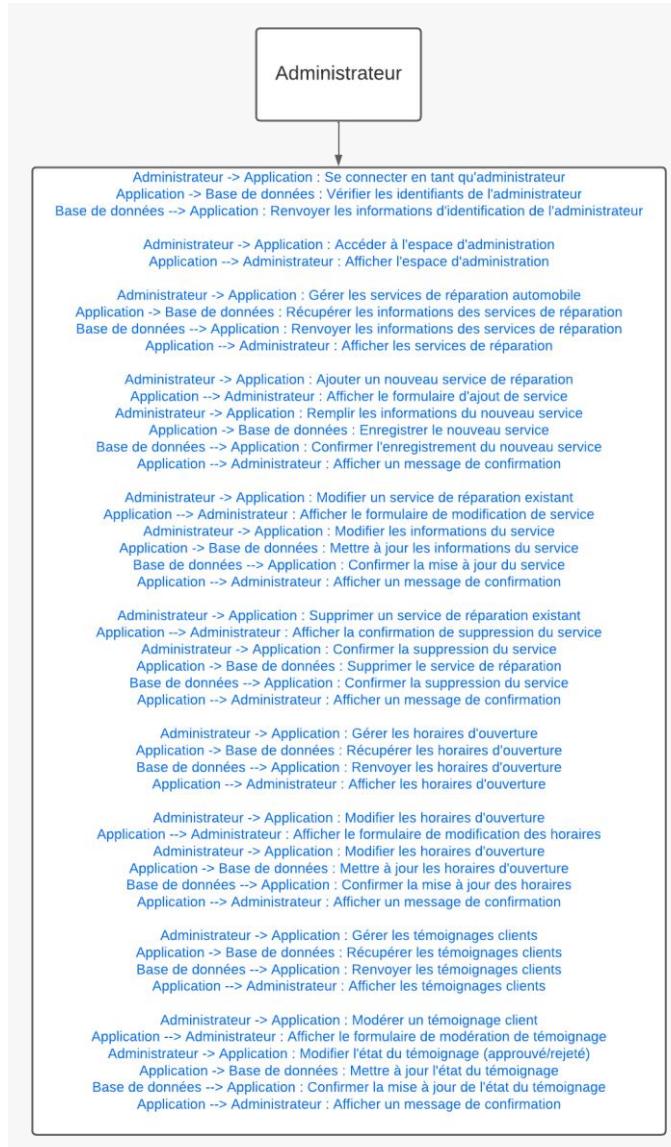


DOSSIER PROFESSIONNEL (DP)

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.

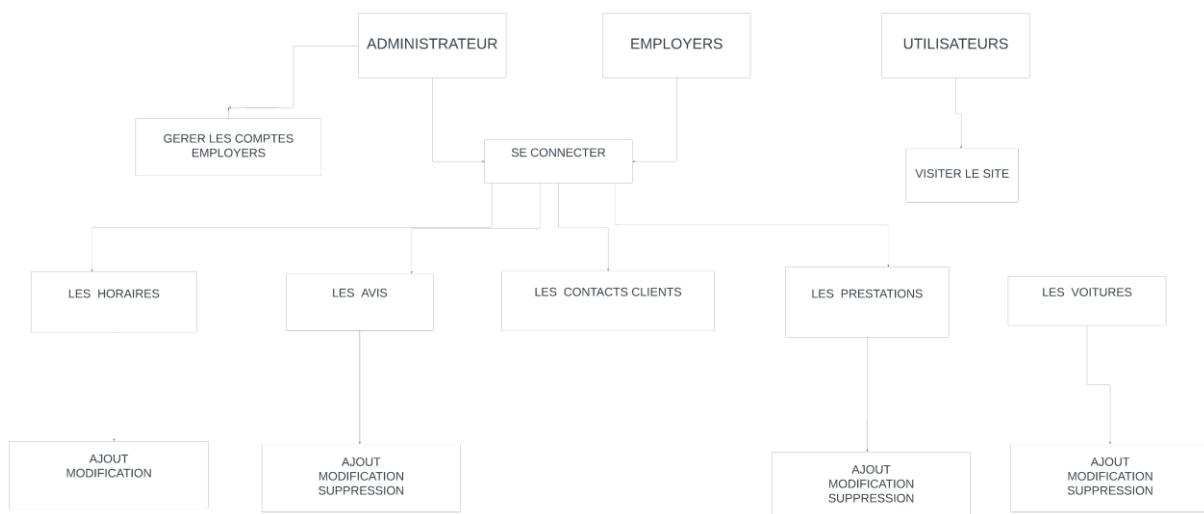
Mon application interagit avec la base de données pour récupérer et mettre à jour les informations pertinentes, et elle affiche des messages de confirmation pour m'indiquer le succès des opérations effectuées.

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.

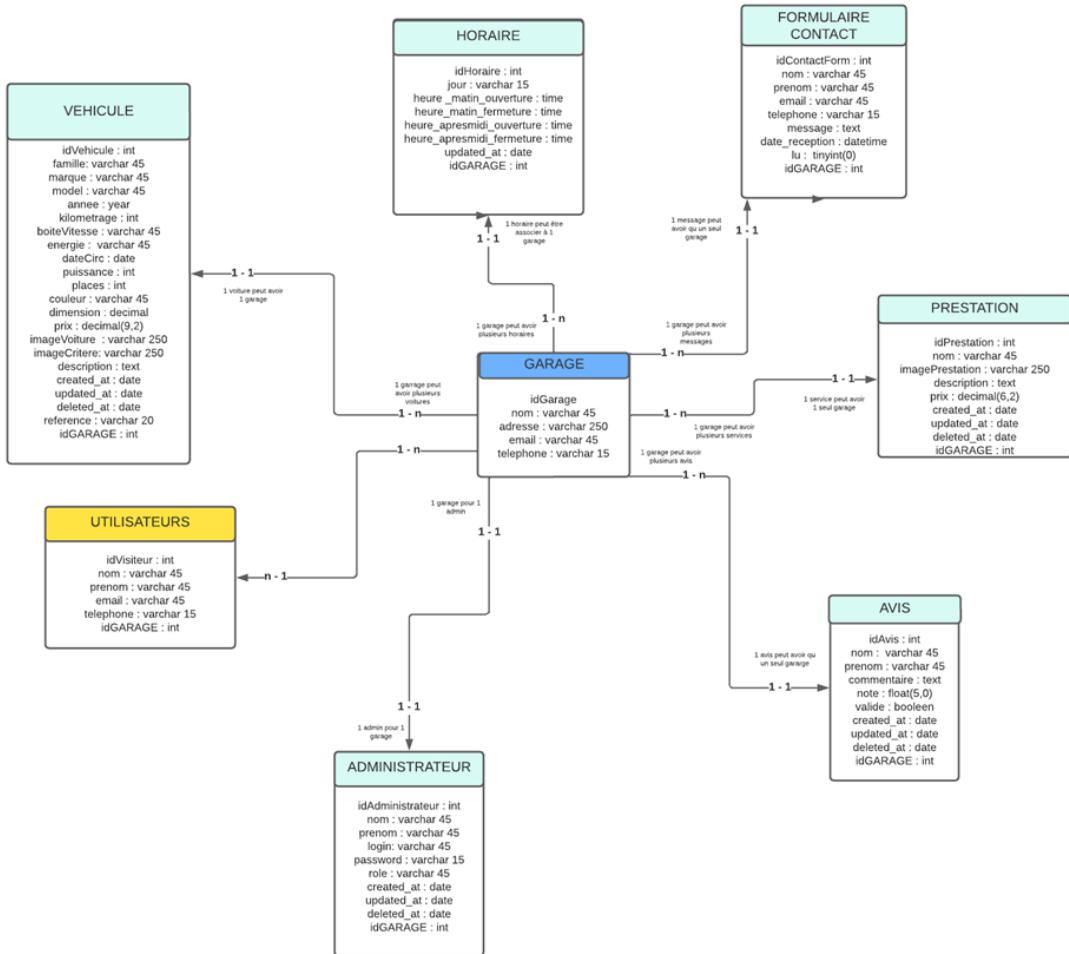
3.0 Diagrammes de Classe

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.

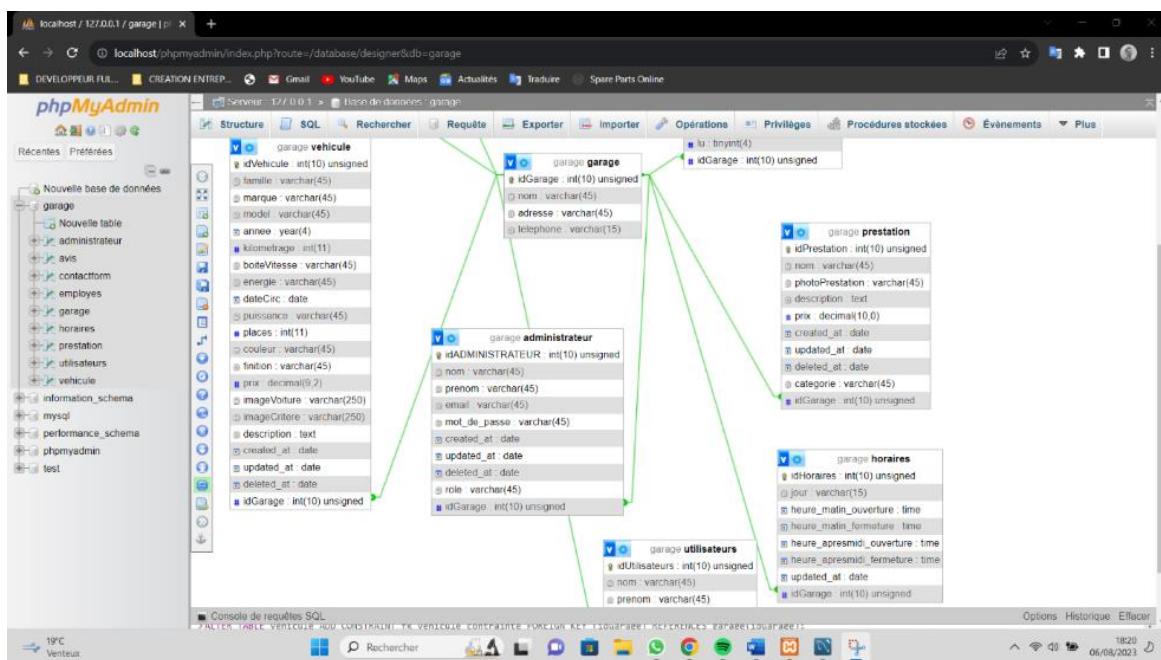
Bien que cette solution ait permis de créer les relations entre les tables conformément à mes besoins, je reste conscient que cela peut être considéré comme une approche moins automatisée et plus sujette aux erreurs.

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.

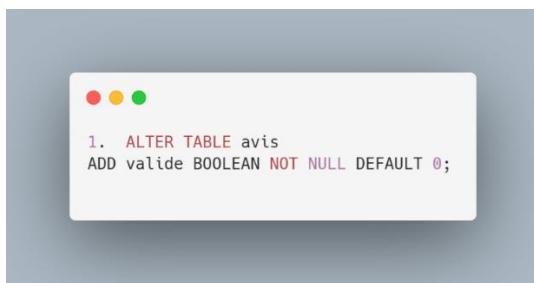
J'ai rapidement rectifié cette omission en ajoutant une colonne supplémentaire appelée "ferme" à la table.

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.

Activité-type 1

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

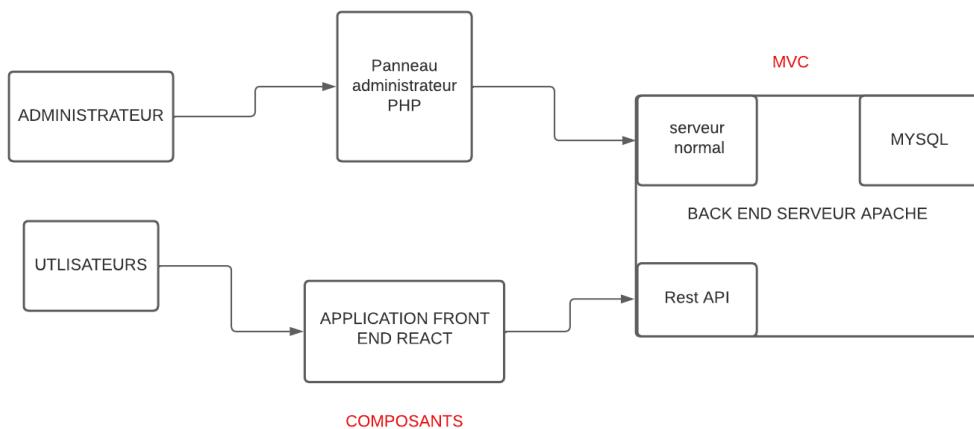
Exemple n°3 ► CONNEXION / DECONNEXION ET MOT DE PASSE

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 cliquerais 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.lPybAhglaiixJMgtyq3...	NULL	

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

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(0,9999); //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](#)

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"}]
```

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.

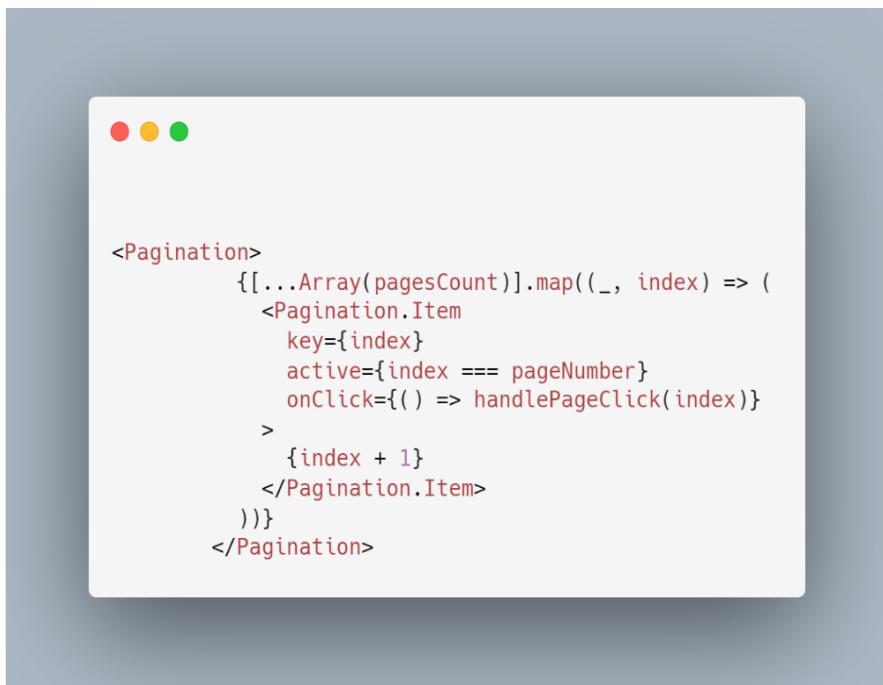
Cette limitation permettra d'améliorer significativement les performances de l'application et éviter que tous les véhicules de la base de données s'affichent en même temps.

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.

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.

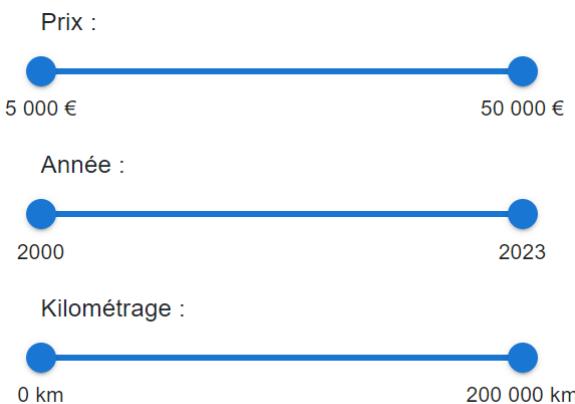
Je la passerai en tant que **props** aux composants fils pour qu'ils puissent **mettre à jour** l'état parent. Chaque composant a été créé en utilisant la bibliothèque **Material-UI** (<https://mui.com/>).

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)

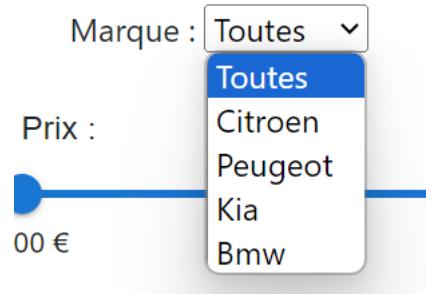
```
return (
  <>
  <Typography id="input-slider" gutterBottom>
    {label}
  </Typography>

  <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.

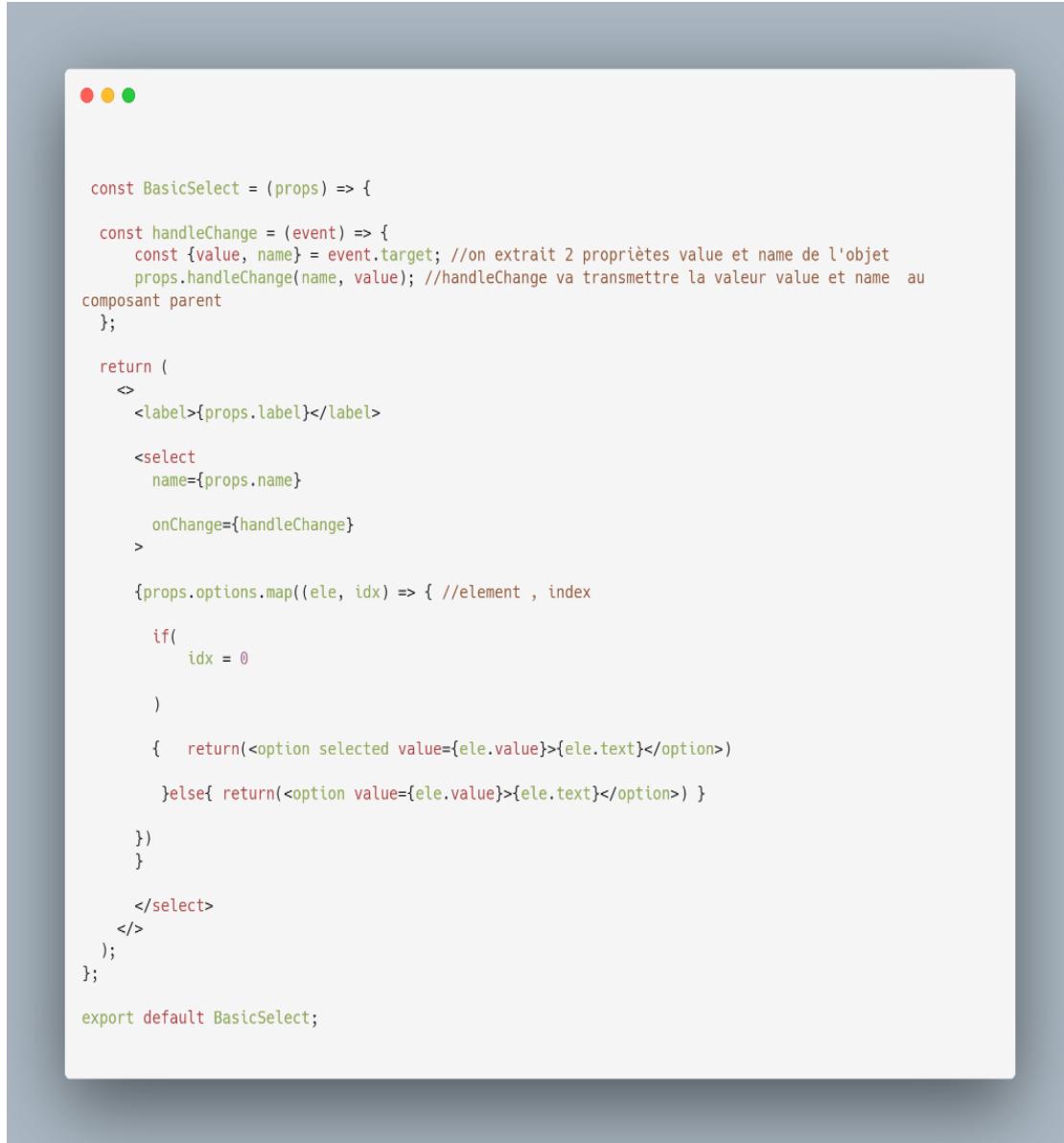
Lorsque l'utilisateur sélectionne une option, la fonction **handleChange** sera utilisée pour communiquer avec le composant parent.

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.

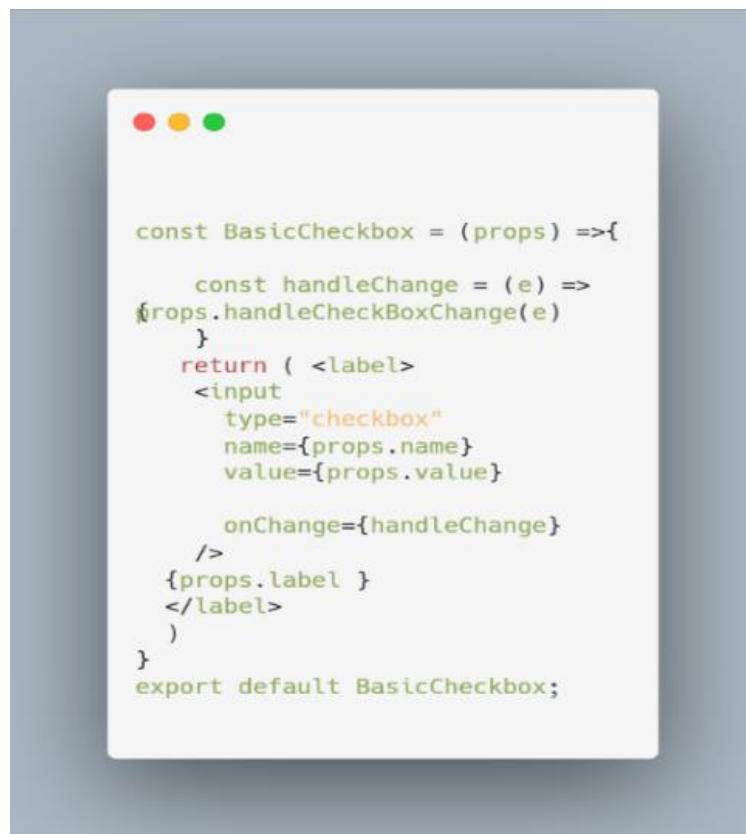
Utilitaire Berline Familiale Citadine SUV

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

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.

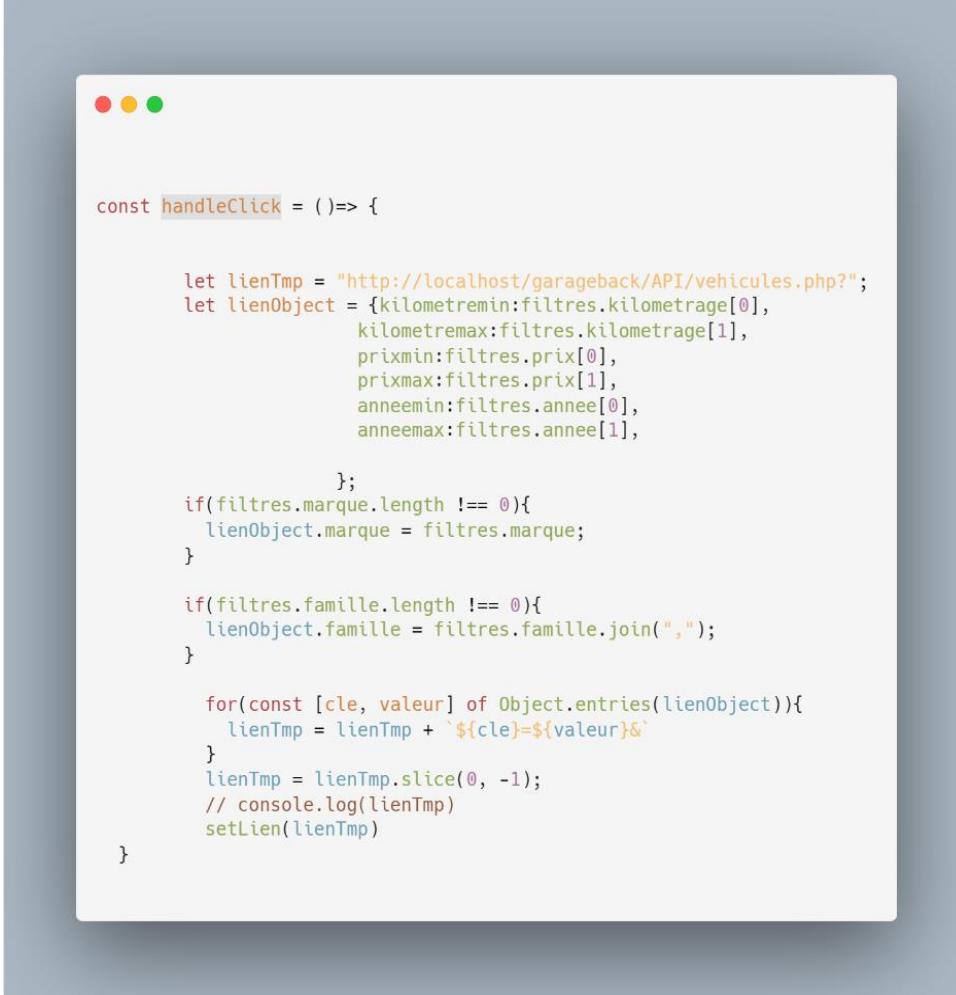


```
useEffect(() => {
  fetch(
    //fetch effectue une requête http, si reponse, elle sera encapsulé dans une promesse
    lien
    // "http://localhost/GarageBack/API/vehicule.php?
    kilometremin=0&kilometremax=200000&anneemin=2000&anneemax=2023&prixmin=5000&prixmax=50000"
  )
  //Si reponse reçu 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:

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

The bottom status bar indicates the response was successful (200 OK) with 11 ms latency and 1.82 KB size.

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**

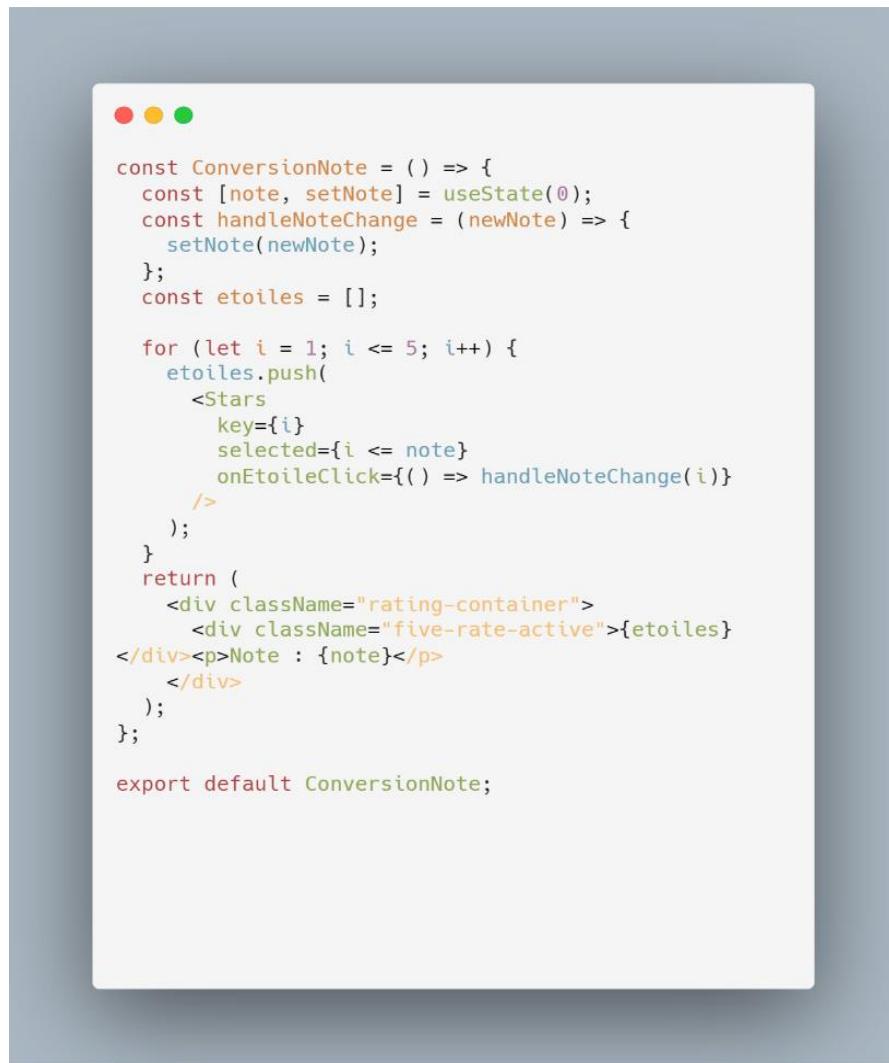
Il n'y a même pas besoins de cliquer sur les étoiles pour les sélectionner, un survol de la souris suffira.



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.

```
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

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.

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

