

Site Gestion Inventaires Automobile

Prestige Motors :

10/07/2023



Auteurs :

BINET Florentin

DIAW Ahmadou

MAGNETTE Arthur

Table des matières

I.	Objectifs du Site Internet Prestige Motors	3
a)	Fonctionnement du site internet général	3
b)	Outils utilisés pour le site web	3
c)	Utilisation de GIT	4
II.	Spécifications du site internet	5
a)	Diagramme de cas d'utilisation du site internet	5
b)	Maquettes des différentes pages du site internet	5
c)	Schéma relationnel de la base de données	8
III.	Roadmap du Projet	8
IV.	Détails techniques des éléments du site internet	8
a)	Formulaire du client pour la réservation, BACKEND	9
b)	Fonctionnement du filtre, modification et suppression, BACKEND	14
	Conclusion et Remerciements	19

I. Objectifs du Site Internet Prestige Motors

L'objectif premier du site Internet est la réservation et la location de voitures disponibles dans une liste. Les clients peuvent prendre rendez-vous pour la récupérer et devront la rendre après un nombre de jours payés.

a) Fonctionnement du site internet général

L'utilisateur arrive sur la page d'accueil du site internet. Il verra le logo, l'en-tête, une barre de navigation pour pouvoir se déplacer entre les différentes pages.

L'utilisateur pourra entrer un formulaire pour pouvoir réserver un véhicule de la catégorie qu'il veut, par exemple collection, urbaine, sport, berline, SUV. Il pourra choisir le véhicule selon une liste affichée avec le filtre défini. Il prendra rendez-vous pour pouvoir récupérer le véhicule au garage.

L'employé pourra accéder à une liste de tous les véhicules spécifiés et pourra voir, modifier et supprimer manuellement les réservations.

L'utilisateur pourra ensuite naviguer sur les pages A propos et avoir des renseignements approfondis sur le garage et les auteurs du site internet.

b) Outils utilisés pour le site web

Le site web sera écrit en html et php et sera stylisé en CSS. Il tournera de manière locale sur XAMPP. Il sera animé par Javascript sur certaines pages pour permettre certaines actions possibles.

L'accès à la base de données MySQL se fera sur le serveur distant hébergé par Florentin sur instance OVH. Nous sommes venus avec cette solution car il fallait centraliser la base pour éviter d'avoir des différences d'enregistrement dans la base de données entre les différents membres du groupe. Nous avons géré et ajouté les données, création de tables grâce à My SQL Workbench.

Schéma du fonctionnement du site internet :



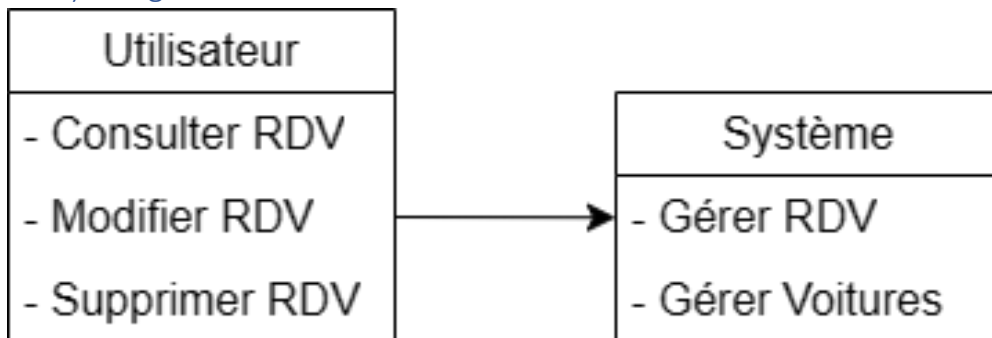
c) Utilisation de GIT

Pour le projet, nous avons mis en place un référentiel git pour pouvoir avoir une trace de l'historique et faire des modifications et les envoyer au serveur et avoir différentes versions. Nous avons fait différentes branches et mis en place une protection sur la branche Master pour éviter les erreurs. Il fallait faire une pull request, faire une review change pour accepter et enfin merge. On a une deuxième branche où l'on a fait la majorité des développements et une branche test pour tester certains ajouts.

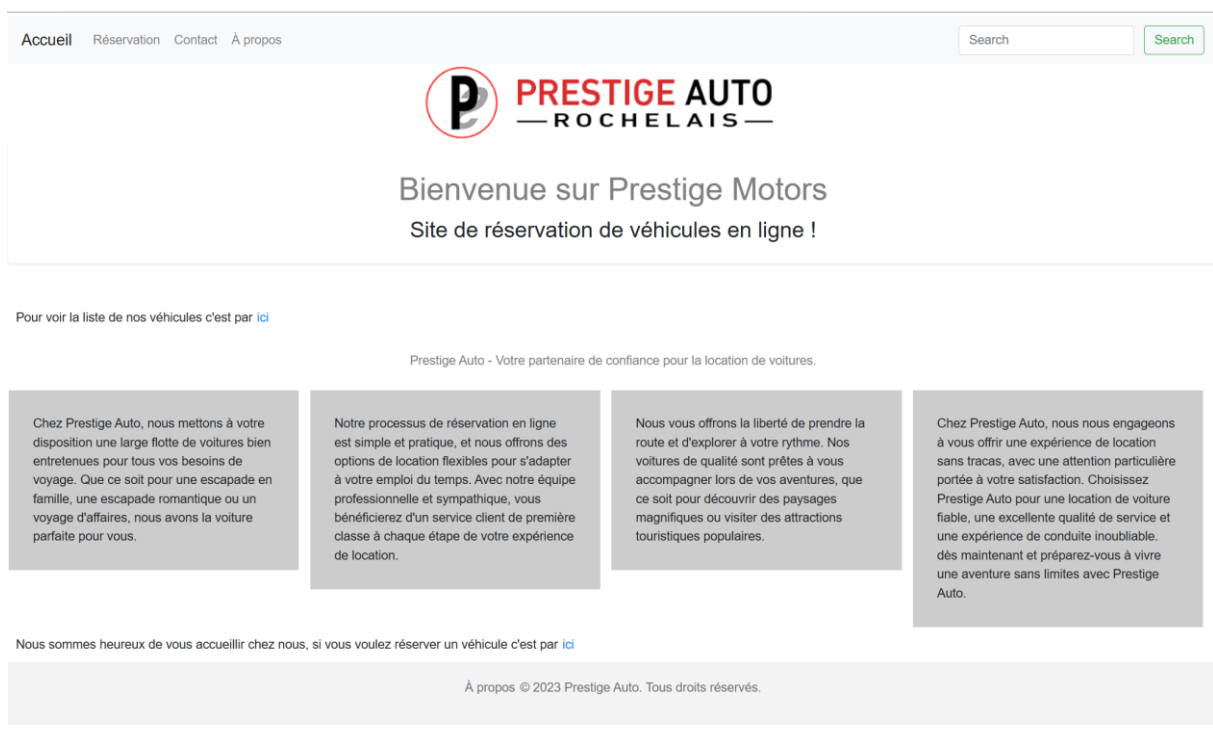
Graph	Description	Date	Author	Commit
	Update apropos.php	9 Jul 2023 21:24	ARTY-cpu	6f35a2c8
	Merge pull request #5 from ARTY-cpu/develop	9 Jul 2023 21:20	ARTY-cpu	458b9c8b
	correctif final	9 Jul 2023 21:18	Arthur Magnette	5ecdad14
	FIN DU SITE (VERSION POUR LUNDI)	9 Jul 2023 20:06	Kanoox	2c991ed0
	modificaton	9 Jul 2023 18:19	Ahmadou0825	4bb01142
	Merge 4	9 Jul 2023 17:59	ARTY-cpu	330af642
	good	9 Jul 2023 17:40	Florentin	6bb2fa60
	Fusion dev master	8 Jul 2023 20:37	ARTY-cpu	32842ac1
	merge resolu	8 Jul 2023 18:29	Arthur Magnette	d3a3a537
	modification	8 Jul 2023 18:26	Ahmadou0825	a6cf1c7e
	Merge branch 'develop' of https://github.com/ARTY-cpu/garage into develop	8 Jul 2023 18:23	Kanoox	eccd6a06
	bug fix voiture dispo 0	8 Jul 2023 18:23	Arthur Magnette	1b1919ae
	test push	8 Jul 2023 18:16	Kanoox	77a1a48e
	finition sur backend cote code	8 Jul 2023 16:47	Arthur Magnette	45fd0c8
	ajout bouton page precedente	8 Jul 2023 10:43	Arthur Magnette	903162eb
	correction index	8 Jul 2023 10:33	Arthur Magnette	e2091d0a
	Merge pull request #2 from ARTY-cpu/develop	6 Jul 2023 21:51	ARTY-cpu	b5e80d45
	Restauration purifiée	6 Jul 2023 21:46	Arthur Magnette	e2520bea
	Revert "Merge branch 'develop' of https://github.com/ARTY-cpu/garage into develop ajout fchiers"	6 Jul 2023 20:59	Arthur Magnette	5c6d107c
	Merge branch 'develop' of https://github.com/ARTY-cpu/garage into develop ajout fchiers	6 Jul 2023 00:00	Ahmadou0825	0df6e04a
	Supprimer des fichiers	5 Jul 2023 23:44	Ahmadou0825	75bbdc3b
	Supprimer des fichiers	5 Jul 2023 23:40	Ahmadou0825	1827a879
	ajout du fichier portrait.html	5 Jul 2023 22:37	Ahmadou0825	60a0a2f0
	backend termine	5 Jul 2023 22:04	Arthur Magnette	507cdd88
	good	5 Jul 2023 21:47	Kanoox	3031e255
	update delete working	4 Jul 2023 21:05	Arthur Magnette	3450ad66
	ajout de portrait.html et portrait.css	4 Jul 2023 17:45	Ahmadou0825	d459d35e
	modification works	4 Jul 2023 16:59	Arthur Magnette	f04323dc
	BOOTSTRAP MAJ V1.0.2	4 Jul 2023 16:59	Kanoox	a4c2d572
	Filtre fonctionne	4 Jul 2023 16:25	Arthur Magnette	e4dba599
	Bootstrap MAJ V1.0.1	4 Jul 2023 14:19	Kanoox	48725024
	Ajout bootstrap + commencement css	4 Jul 2023 12:16	Kanoox	3d87fc9c
	test	4 Jul 2023 11:04	Kanoox	6f776990
	filtre presque pret, supprimer fonctionne	3 Jul 2023 22:34	Arthur Magnette	6de3ab9e
	update_delete en cours	3 Jul 2023 17:56	Arthur Magnette	c0ecfcac
	update RDV en cours	3 Jul 2023 16:16	Arthur Magnette	9489c9f2
	.gitignore ajoute	3 Jul 2023 13:46	Arthur Magnette	bdce59b9
	update css	3 Jul 2023 11:55	Kanoox	debc131d
	Test upload	2 Jul 2023 15:29	Florentin	6f86c0b7
	modif msg clarifie	2 Jul 2023 11:53	Arthur Magnette	02b0f46a
	update bdd test	1 Jul 2023 22:58	Arthur Magnette	179fe0e4
	lecture BDD, prise de rdv BETA terminee	1 Jul 2023 17:57	Arthur Magnette	9d4efcd3
	liaison bdd php presque terminee pour rdv	1 Jul 2023 15:43	Arthur Magnette	508ba7e8
	formulaire fini	1 Jul 2023 15:13	Arthur Magnette	be375fa6
	formulaire combo box termine	1 Jul 2023 13:47	Arthur Magnette	f754ba9c
	get modele voiture formulaire	1 Jul 2023 09:39	Arthur Magnette	9f11d353
	formulaire genereee	1 Jul 2023 09:27	Arthur Magnette	3a04fbfc
	Ajout formulaire simple	28 Jun 2023 15:37	Arthur Magnette	dfe67a8e
	html test ajoute	28 Jun 2023 12:34	Arthur Magnette	0d694204
	test trie & script php test	28 Jun 2023 12:29	Arthur Magnette	be59a850
	ajout index et apropos	28 Jun 2023 11:40	Ahmadou0825	4bfaa62a
	message	28 Jun 2023 10:01	Ahmadou0825	527c7b57
	test	27 Jun 2023 16:00	Kanoox	a3432feb
	test 1e commit	27 Jun 2023 15:53	Arthur Magnette	f02859cd
	Initial commit	25 Jun 2023 11:02	ARTY-cpu	6b79a9f7

II. Spécifications du site internet

a) Diagramme de cas d'utilisation du site internet




b) Maquettes des différentes pages du site internet



Page d'accueil index.php 1

Ici, l'utilisateur est accueilli sur la page d'accueil. Il peut lire le descriptif de ce site web. Il peut naviguer dans la barre de navigation pour voir les différentes pages. Il peut regarder la liste des véhicules en images. Il peut aussi réserver son véhicule dans l'onglet Réservation ou il peut le faire en cliquant sur « ici ».

[Accueil](#) [Réservation](#) [Contact](#) [À propos](#)



Bienvenue sur Prestige Motors

Vous êtes actuellement sur la page de réservation de véhicules

[Voir la liste des véhicules en image](#)

Formulaire de réservation

Name

Email

Phone:

Address:

Appointment Date:

Vehicle Category:

Citadine

Vehicle Model:

Renault Clio Noir 2004

Si vous voulez modifier un de vos rendez-vous c'est par [ici](#)

À propos © 2023 Prestige Auto. Tous droits réservés.

Page de Réservation reservation.php 1

L'utilisateur devra remplir tous les champs du formulaire. Il sera amené à choisir la catégorie de véhicule qu'il veut obtenir et son modèle de véhicule. En cliquant sur « J'ai fini ! », son véhicule sera réservé.

[Accueil](#) [Réservation](#) [Contact](#) [À propos](#)



Bienvenue sur Prestige Motors

Vous êtes actuellement sur la page de modification de votre réservation de véhicules

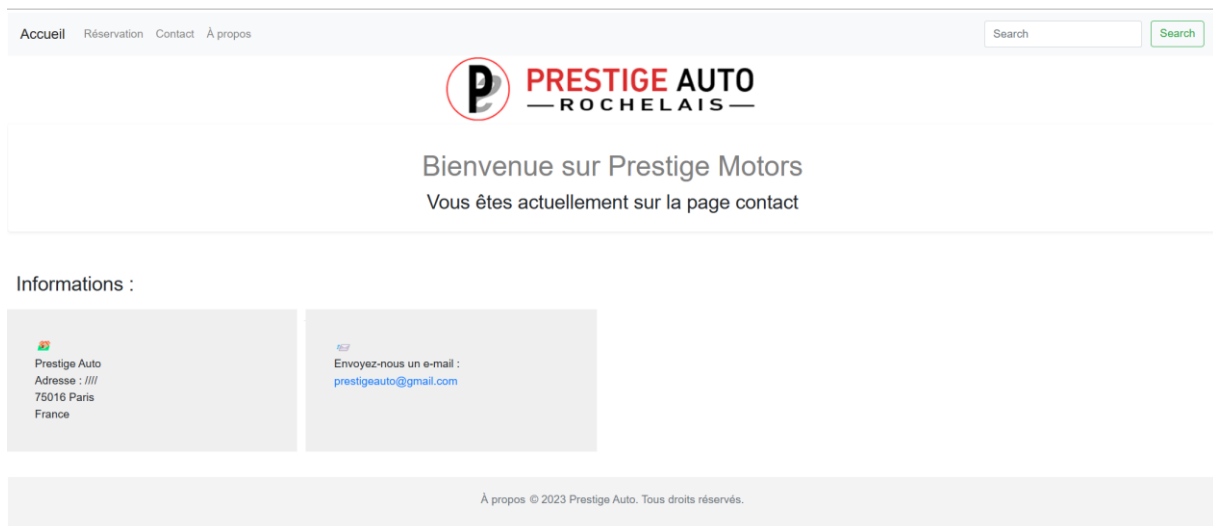
Liste de vos rendez-vous :

Filtre:

À propos © 2023 Prestige Auto. Tous droits réservés.

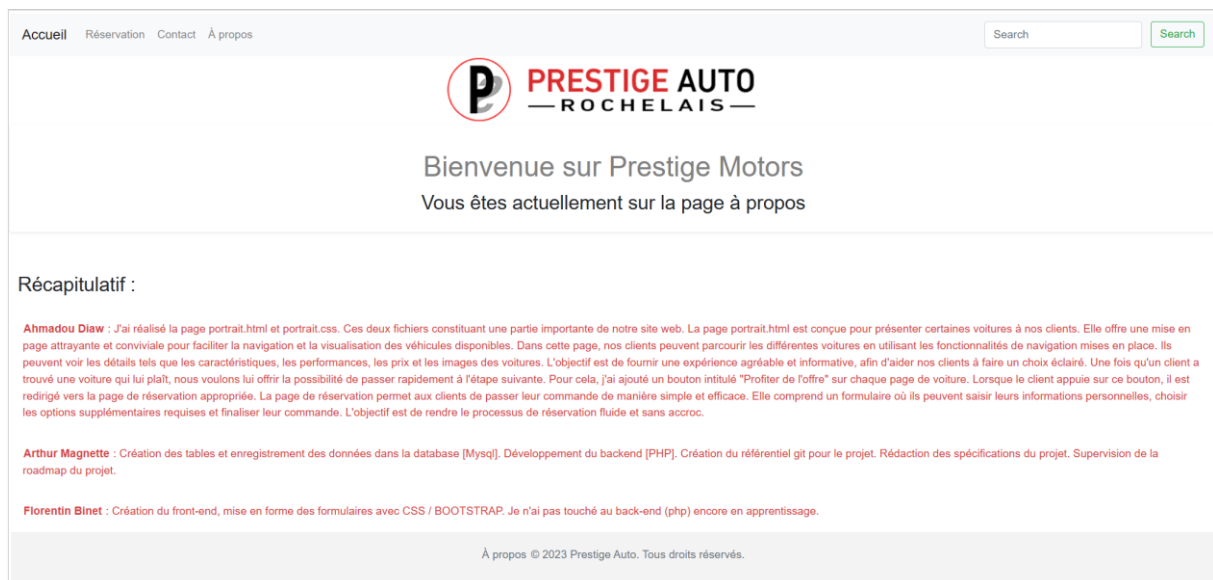
Page de changement de RDV modifrdv.php 1

Ici, le client sera amené à entrer son adresse mail considéré comme unique pour pouvoir modifier sa ou ses réservations de véhicules. Il pourra aussi les annuler.



Page de contact *contact.php 1*

C'est la page contact du site Internet. Dès qu'on clique sur l'adresse mail, on est amené à sélectionner le client de mail pour que l'on puisse envoyer un mail à Prestige Auto.



Page *apropos.php 1*

Ici, c'est du HTML et CSS classique et sur cette page, on peut en apprendre plus sur les tâches qu'ont effectué les membres du groupe.



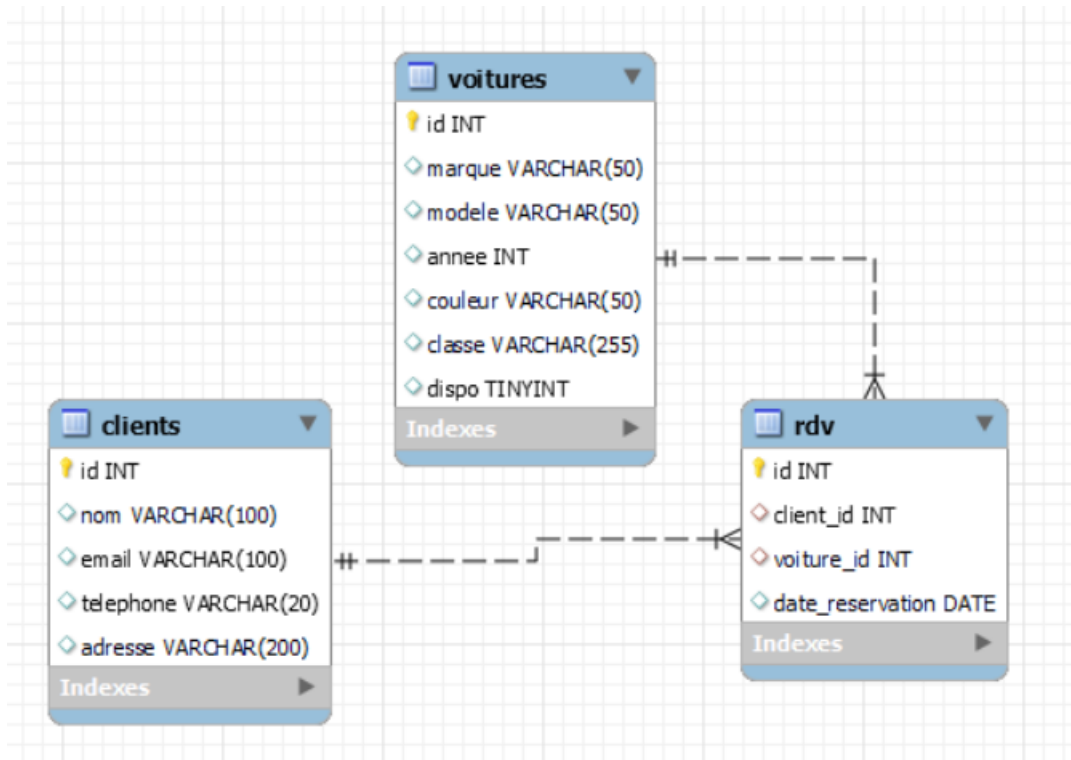
Page portrait



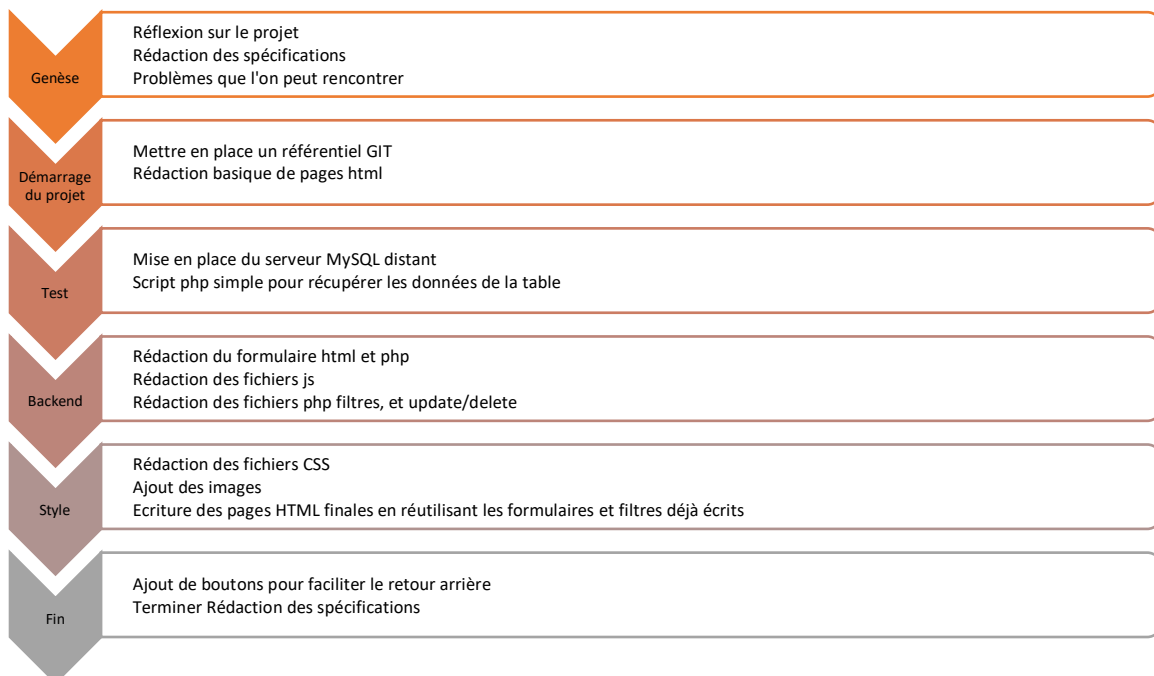
Page portrait

La page "Portrait" offre une galerie visuelle des véhicules disponibles à la location ou à la vente. Chaque véhicule est présenté avec des détails complets, y compris les dates de disponibilité, les prix de location ou de vente, ainsi que les garanties et assurances associées.

c) Schéma relationnel de la base de données



III. Roadmap du Projet



IV. Détails techniques des éléments du site internet

a) Formulaire du client pour la réservation, BACKEND

```
1 <head>
2 <title>Form TEST</title>
3 <script src="https://code.jquery.com/jquery-3.6.0.min.js"></script>
4 <script src="../../js/detect_model.js"></script>
5 </head>
6 <body>
7 <h1>Form</h1>
8
9 <form id="main-form" action="../../php/formulaire.php" method="POST">
10 <div class="main-component">
11 <label for="name">Name:</label>
12 <input type="text" name="name" id="name" required>
13
14 <label for="email">Email:</label>
15 <input type="email" name="email" id="email" required>
16
17 <label for="phone">Phone:</label>
18 <input type="text" name="phone" id="phone" required>
19
20 <label for="address">Address:</label>
21 <input type="text" name="address" id="address" required>
22
23 <label for="appointment-date">Appointment Date:</label>
24 <input type="date" name="appointment-date" id="appointment-date" required>
25
26 <div class="sub-component">
27 <label for="vehicle-category">Vehicle Category:</label>
28 <select name="vehicle-category" id="vehicle-category" required>
29 <option value="citadine">Citadine</option>
30 <option value="suv">SUV</option>
31 <option value="ancienne">Ancienne</option>
32 <option value="luxe">Luxe</option>
33 </select>
34
35 <label for="vehicle-model">Vehicle Model:</label>
36 <select name="vehicle-model" id="vehicle-model" required>
37 <!-- Les options pour la sélection de la marque et du modèle seront ajoutées dynamiquement ici -->
38 </select>
39
40 <!-- Champ de formulaire caché pour l'ID de la voiture -->
41 <input type="hidden" name="vehicle-id" id="vehicle-id">
42 </div>
43
44 <input type="submit" name="main-save" value="Submit">
45 </div>
46 </form>
47
48 <form id="sub-form" action="../../php/get_model.php" method="POST"></form>
49 </body>
```

Ce fichier HTML contient un formulaire qui permet aux utilisateurs de soumettre leurs informations.

- Les scripts jQuery et **detect_model.js** sont inclus via les balises **<script>**.
- Le formulaire principal est défini avec l'ID **main-form** et il envoie les données à **../php/formulaire.php** en utilisant la méthode POST.
- Les champs de formulaire tels que le nom, l'email, le téléphone, l'adresse et la date de rendez-vous sont requis (**required**).
- Il y a un composant secondaire qui contient des champs de sélection pour la catégorie de véhicule et le modèle de véhicule. L'élément **select** avec l'ID **vehicle-model** n'a pas d'options statiques. Les options seront ajoutées dynamiquement en utilisant le script **detect_model.js**. Quand on sélectionne une catégorie de voitures, par exemple SUV, le modèle du véhicule est dynamique. Les véhicules seront affichés dans la catégorie SUV. Il y a un deuxième formulaire qui est en commentaire sur la page HTML (**sub-form**) qui n'a pas de contenu à l'intérieur au démarrage, mais il enverra les données à **../php/get_model.php** lorsqu'il sera utilisé.

r: Citadine ▾ Vehicle Model: Peugeot 208 Gris 2020 ▾

Citadine
SUV
Ancienne
Luxe

Avant Sélection

: SUV ▾ Vehicle Model: Peugeot 3008 Gris 2018 ▾

Peugeot 3008 Gris 2018
Dacia Duster Marron 2017
Peugeot 2008 Vert 2016
Renault Captur Blanche 2015
Land-Rover Discovery Sport Gris 2015
Land-Rover Discovery Sport Noir 2017
Jeep Compass Gris 2015
Ford Ranger Gris 2013
Dodge Ram Orange 2014

Après sélection

Quand on sélectionne une catégorie de voitures, par exemple SUV, le modèle du véhicule est dynamique. Les véhicules seront affichés dans la catégorie SUV.

- Il y a également un champ de formulaire caché (**input** de type **hidden**) avec l'ID **vehicle-id**, qui stockera l'ID du véhicule sélectionné.

Cette page html est écoutée par un script Javascript et la technologie jQuery pour détecter quel est la catégorie de véhicule affichée à l'écran.

```

function updateVehicleModels() {
    // Récupération de la classe de véhicule sélectionnée
    var vehicleCategory = $("#vehicle-category").val();

    // Envoi de la requête AJAX
    $.ajax({
        type: "POST",
        url: "../php/get_model.php",
        data: { "vehicle-category": vehicleCategory },
        dataType: "json",
        success: function(models) {
            // Réinitialisation de la combobox de la marque et du modèle
            var vehicleModelSelect = $("#vehicle-model");
            vehicleModelSelect.empty();

            // Ajout des options pour chaque modèle récupéré
            for (var i = 0; i < models.length; i++) {
                var option = $("<option></option>");
                option.val(models[i].id);
                option.text(models[i].marque + " " + models[i].modele + "
" + models[i].couleur + " " + models[i].annee);
                vehicleModelSelect.append(option);
            }

            // Déclencher l'événement change manuellement pour mettre à
jour l'ID avec la première valeur affichée
            vehicleModelSelect.trigger('change');
        },
        error: function(xhr, status, error) {
            // Gérer les erreurs éventuelles
            console.log("Erreur lors de la récupération des modèles de
véhicule.");
        }
    });
}

```

Ce script utilise jQuery et repose sur l'événement "document ready" pour s'assurer que le code JavaScript s'exécute une fois que le document HTML est complètement chargé. Les points clés du script sont :

- La fonction **updateVehicleModels()** est définie pour mettre à jour la liste des modèles de véhicules en fonction de la catégorie de véhicule sélectionnée.
- Lorsque la catégorie de véhicule est modifiée (**change**), la fonction **updateVehicleModels()** est appelée pour récupérer les modèles correspondants via une requête AJAX.
- La requête AJAX est envoyée au fichier **../php/get_model.php** en utilisant la méthode POST. La valeur de la catégorie de véhicule est envoyée en tant que paramètre dans l'objet de données.

- En cas de succès, la fonction **success** est exécutée. Elle réinitialise d'abord la liste des modèles de véhicules, puis ajoute des options pour chaque modèle récupéré à partir des données JSON renvoyées par le fichier PHP.
- Lorsqu'un modèle de véhicule est sélectionné (**change**), la fonction anonyme associée est appelée. Elle récupère l'ID du modèle sélectionné et met à jour la valeur du champ de formulaire caché (**#vehicle-id**).
- Un message est également affiché dans la console pour afficher l'ID du véhicule sélectionné à des fins de débogage.
- Enfin, l'appel initial de **updateVehicleModels()** est effectué pour mettre à jour les modèles de véhicules en fonction de la catégorie de véhicule par défaut.

Le fonctionnement du script php `get_model` est le suivant :

- Une connexion à la base de données est établie en utilisant les informations d'identification appropriées (serveur, nom d'utilisateur, mot de passe et nom de la base de données).
- Si la connexion échoue, un message d'erreur est affiché et le script se termine.
- La catégorie de véhicule sélectionnée est récupérée à partir de la requête AJAX POST avec la clé "vehicle-category".
- Une requête SQL est exécutée pour sélectionner les modèles de véhicules correspondant à la catégorie spécifiée. La requête utilise la clause WHERE pour filtrer les véhicules disponibles (**dispo = 1**) et la catégorie de véhicule.

```
SELECT id,marque, modele, annee, couleur FROM voitures WHERE dispo = 1
AND classe = '$vehicleCategory'
```

Les résultats de la requête sont parcourus, et pour chaque modèle de véhicule, les informations pertinentes (ID, marque, modèle, année, couleur) sont extraites et ajoutées à un tableau.

```
// Création d'un tableau pour stocker les modèles
$models = array();

// Parcours des résultats et ajout des modèles au tableau
while ($row = mysqli_fetch_assoc($result)) {
    $modelData = array(
        'id' => $row['id'],
        'marque' => $row['marque'],
        'modele' => $row['modele'],
        'annee' => $row['annee'],
        'couleur' => $row['couleur']
    );
    $models[] = $modelData;
}
```

- La connexion à la base de données est fermée.
- Les données des modèles de véhicules sont renvoyées au format JSON en utilisant la fonction **json_encode()**.
- Le résultat JSON est renvoyé en tant que réponse à la requête AJAX effectuée dans le script JavaScript.

Ce fichier PHP (**formulaire.php**) est responsable de la gestion du formulaire de réservation. Voici les points clés du script :

- Les informations de connexion à la base de données sont configurées (**\$servername**, **\$username**, **\$password** et **\$database**).
- Une connexion à la base de données est établie en utilisant les informations d'identification fournies.
- Si la connexion échoue, un message d'erreur est affiché et le script se termine.
- Les données du formulaire sont récupérées à l'aide de la méthode POST (**\$_POST**).
- Une transaction est démarrée pour garantir l'intégrité des données lors de l'insertion dans les tables. Le script effectue les opérations suivantes :
 - Vérifie si l'email existe déjà dans la table 'clients'. Si oui, récupère l'ID correspondant.
 - Sinon, insère les données du client dans la table 'clients' et récupère l'ID généré.
`INSERT INTO clients (nom, email, telephone, adresse) VALUES ('$name', '$email', '$phone', '$address')`
 - Insère les données de réservation dans la table 'rdv' avec l'ID du client et l'ID du véhicule sélectionné.
`INSERT INTO rdv (client_id, voiture_id, date_reservation) VALUES ('$clientId', '$vehicleId', '$appointmentDate')`
 - Met à jour la disponibilité du véhicule dans la table 'voitures' en le marquant comme non disponible (**dispo = 0**).
`UPDATE voitures SET dispo = 0 WHERE id = '$vehicleId'`
- Si toutes les opérations sont réussies, la transaction est validée et un message de réussite est affiché. Un bouton est également affiché pour revenir à la page précédente.
- Si une erreur se produit à tout moment, la transaction est annulée et un message d'erreur approprié est affiché. Un bouton est également affiché pour revenir à la page précédente.

b) Fonctionnement du filtre, modification et suppression, BACKEND

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>FILTRE TEST</title>
</head>
<body>
  <h1>Liste de vos rdv:</h1>

  <form action="../php/update_bdd.php" method="GET">
    <label for="Filtre">Filtre:</label>
    <input type="text" name="Filtre" id="Filtre" required
placeholder="Veuillez entrer votre email.">

    <input type="hidden" name="list_records" value="true">
    <input type="submit" value="Vos RDV">
  </form>
</body>
</html>
```

Ce fichier HTML contient un formulaire de filtre permettant de sélectionner les rendez-vous d'un client spécifique en entrant son adresse e-mail. Voici les points clés à noter :

- Le formulaire envoie les données via la méthode GET vers le fichier `../php/update_bdd.php`.
- Un champ de saisie (**input** de type **text**) est utilisé pour entrer l'adresse e-mail du client à filtrer.
- Le champ de saisie est requis (**required**) et est identifié par l'ID **Filtre**.
- Un champ de formulaire caché (**input** de type **hidden**) avec le nom **list_records** est inclus et défini sur la valeur "true". Cela permettra au script PHP de reconnaître que la demande concerne l'affichage des rendez-vous du client.
- Un bouton de soumission (**input** de type **submit**) est affiché avec l'étiquette "Vos RDV" pour envoyer le formulaire.

Le fichier php `update_bdd.php` est appelé dès que l'on clique sur le bouton submit. Sa logique est :

- La session est démarrée en utilisant `session_start()` pour stocker l'e-mail du client dans une variable de session (`$_SESSION['email']`).
- Une connexion à la base de données est établie en utilisant les informations d'identification fournies.
- Si la connexion échoue, un message d'erreur est affiché et le script se termine.
- L'e-mail du client est récupéré à partir de la variable `$_GET['Filtre']` ou est initialisé à une chaîne vide si aucune valeur n'est fournie.
- Une requête SQL est préparée pour récupérer les données des rendez-vous du client correspondant à l'e-mail.

```
$sql = "SELECT clients.id AS client_id, voitures.id AS voiture_id, rdv.id AS  
rdv_id, clients.nom, clients.adresse, rdv.date_reservation,  
CONCAT(voitures.marque, ' ', voitures.modele) AS voiture  
FROM rdv  
JOIN clients ON clients.id = rdv.client_id  
JOIN voitures ON voitures.id = rdv.voiture_id  
WHERE clients.email = ?";
```

- La valeur de l'e-mail est liée au paramètre de la requête préparée et la requête est exécutée.
- Les résultats de la requête sont récupérés à l'aide de la méthode `get_result()`.

- Si des rendez-vous sont trouvés, les données sont affichées sous forme de tableau.

```
<table>
    <tr>
        <th>Nom</th>
        <th>Adresse</th>
        <th>Date de réservation</th>
        <th>Voiture</th>
        <th>Actions</th>
    </tr>';

while ($row = $result->fetch_assoc()) {
    echo '<tr>';
    echo '<td>' . $row["nom"] . '</td>';
    echo '<td>' . $row["adresse"] . '</td>';
    echo '<td>' . $row["date_reservation"] . '</td>';
    echo '<td>' . $row["voiture"] . '</td>';
    [...]
}

echo '</table>';
```

- Pour chaque ligne de rendez-vous, des boutons "Modifier" et "Supprimer" sont affichés. Les boutons sont associés à des fonctions JavaScript (**modifyRow()** et **deleteRow()**) définies dans le fichier JavaScript **update_delete.js**.

```
<button onclick="modifyRow(' . $row["rdv_id"] . ')">Modifier</button>
<button onclick="deleteRow(' . $row["rdv_id"] . ')">Supprimer</button>
```

- Des champs de formulaire cachés sont inclus pour stocker les informations nécessaires à la modification ou à la suppression d'un rendez-vous spécifique (ID du rendez-vous, ID de la voiture, date de réservation et e-mail du client).

```
echo '<input type="hidden" id="voiture_id_' . $row["rdv_id"] . '" value="' .
$row["voiture_id"] . '">';
echo '<input type="hidden" id="date_reservation_id_' . $row["rdv_id"] . '"
value="' . $row["date_reservation"] . '">';
echo '<input type="hidden" id="email" . '" value="' . $email . '">';
```

- Un bouton est également affiché pour revenir à la page précédente.
- Si aucun rendez-vous n'est trouvé, un message approprié est affiché.
- Si une modification a été effectuée via un formulaire de modification, un message de réussite est affiché.
- Les ressources JavaScript sont incluses à la fin du fichier.

```
<script src="../../js/update_delete.js"></script>
```

Le formulaire de modification est géré par le fichier `update_delete.js`. Ce fichier gère aussi deux scripts php qui font la modification des lignes et la suppression des lignes.

Le résultat général du filtre sans CSS est :

▼

12/07/2023

📅

Modifier

Annuler

Nom	Adresse	Date de réservation	Voiture	Actions
Jean	11 rue aaaa	2023-07-12	Peugeot 208	<div>Modifier</div> <div>Supprimer</div>

Revenir à la page précédente

Ce fichier JavaScript (**update_delete.js**) contient deux fonctions :

1. La fonction **modifyRow(id)** est utilisée pour modifier une ligne de rendez-vous. Voici comment elle fonctionne :
 - La fonction prend en paramètre l'ID de la ligne de rendez-vous à modifier.
 - Tout d'abord, elle vérifie si un formulaire de modification existe déjà en recherchant un élément avec la classe **.modification-form**. Si un formulaire existe, il est supprimé.
 - Les valeurs actuelles de la ligne de rendez-vous (voiture, date de réservation, e-mail) sont récupérées à partir des éléments correspondants dans le document.
 - Ensuite, un formulaire est créé dynamiquement en utilisant la méthode **createElement()** et les attributs appropriés sont définis (classe, méthode, action).
 - Un champ caché est ajouté au formulaire pour stocker l'ID du rendez-vous à modifier.
 - Une requête AJAX est effectuée pour récupérer les véhicules disponibles. Le résultat de la requête est analysé en tant que JSON.
 - Une combobox est créée pour permettre la modification du véhicule. Les options correspondant aux véhicules disponibles sont ajoutées à la combobox.
 - La valeur de la combobox est définie sur le véhicule actuel de la ligne de rendez-vous.
 - Un champ de date est ajouté au formulaire pour permettre la modification de la date de réservation.
 - Un champ caché est ajouté pour stocker l'e-mail du client.
 - Un bouton de soumission et un bouton d'annulation sont ajoutés au formulaire.
 - Enfin, le formulaire est inséré au début du document.
2. La fonction **deleteRow(id)** est utilisée pour supprimer une ligne de rendez-vous. Voici comment elle fonctionne :
 - La fonction prend en paramètre l'ID de la ligne de rendez-vous à supprimer.
 - Elle demande une confirmation à l'utilisateur à l'aide de la fonction **confirm()**.
 - Si l'utilisateur confirme, un formulaire est créé dynamiquement avec les attributs appropriés (méthode, action).
 - Un champ caché est ajouté au formulaire pour stocker l'ID du rendez-vous à supprimer.
 - Un champ caché est ajouté pour stocker l'e-mail du client.
 - Le formulaire est ajouté à la page et soumis.

La fonction **modifyRow(id)** après avoir construit le formulaire appelle **update.php**.

- Tout d'abord, il démarre une session et utilise la fonction **ob_start()** pour mettre en mémoire tampon la sortie.
- Ensuite, il établit une connexion à la base de données en utilisant les informations de connexion.
- Les données du formulaire sont récupérées à l'aide de la superglobale **\$_POST**.
- L'e-mail du client est récupéré depuis la session créée sur **update_bdd.php**.
- Une requête SQL est exécutée pour désactiver l'ancien véhicule associé au rendez-vous, en mettant à jour la colonne **dispo** de la table **voitures**.

```
// Désactiver l'ancien véhicule
:disable_sql = "UPDATE voitures SET dispo = 1 WHERE id = (
    SELECT voiture_id FROM rdv WHERE id = ?
)";
:disable_stmt = $connection->prepare($disable_sql);
:disable_stmt->bind_param("i", $id);
:disable_stmt->execute();
:disable_stmt->close();

// Mettre à jour les données
$update_sql = "UPDATE rdv SET voiture_id = ?, date_reservation = ? WHERE id = ?";
$update_stmt = $connection->prepare($update_sql);
$update_stmt->bind_param("iss", $voiture, $date_reservation, $id);
$update_stmt->execute();
$rows_affected = $update_stmt->affected_rows;
$update_stmt->close();
```

- Les données du rendez-vous sont mises à jour dans la table **rdv** en utilisant une autre requête SQL.

```
// Requête pour récupérer les données mises à jour
$select_sql = "SELECT clients.id AS client_id, voitures.id AS voiture_id,
rdv.id AS rdv_id, clients.nom, clients.adresse, rdv.date_reservation,
CONCAT(voitures.marque, ' ', voitures.modele) AS voiture
FROM rdv
JOIN clients ON clients.id = rdv.client_id
JOIN voitures ON voitures.id = rdv.voiture_id
WHERE rdv.id = ?";
```

- Le nombre de lignes affectées par la mise à jour est récupéré.
- Une requête est effectuée pour récupérer les données mises à jour à partir des tables **clients**, **voitures** et **rdv**.
- On vérifie si une ligne a été mise à jour avec succès et si des résultats ont été obtenus.
- Si la mise à jour a été effectuée avec succès et qu'il y a des résultats, un message "Data Inserted and modified." est affiché.
- Si aucune ligne n'a été mise à jour ou s'il n'y a pas de résultats, un message "Data Not Inserted." est affiché.
- Ensuite, tout contenu HTML précédent est nettoyé à l'aide de **ob_end_clean()** pour éviter toute sortie non souhaitée.

- Enfin, l'utilisateur est redirigé vers la page précédente (**update_bdd.php**) avec l'e-mail du client en tant que paramètre GET.

```
// Rediriger vers la page précédente
header("Location: ../php/update_bdd.php?Filtre=" . urlencode($email));
exit();
```

La fonction **deleteRow(id)** après avoir construit le formulaire appelle **update.php**.

- Tout d'abord, il démarre une session et utilise la fonction **ob_start()** pour mettre en mémoire tampon la sortie.
- Ensuite, il établit une connexion à la base de données en utilisant les informations de connexion.
- Les données de la session sont récupérées pour obtenir le courriel du client.
- La variable POST **id** est vérifiée pour s'assurer qu'elle est définie.
- Ensuite, une requête SELECT est exécutée pour vérifier la correspondance de l'identifiant de rendez-vous avec une ligne dans la table **rdv**. Si aucune correspondance n'est trouvée, un message d'erreur est affiché.

```
// Requête SELECT pour vérifier la correspondance de l'identifiant de rendez-vous
$select_sql = "SELECT * FROM rdv WHERE id = ?";
$select_stmt = $connection->prepare($select_sql);
$select_stmt->bind_param("i", $rdv_id);
$select_stmt->execute();
$result = $select_stmt->get_result();

if ($result->num_rows === 0) {
    die("Aucun résultat trouvé pour l'identifiant de rendez-vous : " .
$rdv_id);
}

$select_stmt->close();
```

- Les valeurs de l'identifiant de rendez-vous sont affichées à des fins de débogage.
- Une autre requête SELECT est exécutée pour récupérer l'identifiant du véhicule associé au rendez-vous.

```
SELECT voiture_id FROM rdv WHERE id = ?
```

- Si aucun véhicule n'est associé à l'identifiant de rendez-vous, un message d'erreur est affiché.
- La disponibilité du véhicule est mise à jour dans la table **voitures** en utilisant une requête UPDATE.

```
UPDATE voitures SET dispo = 1 WHERE id = ?
```

- Une requête DELETE est utilisée pour supprimer la ligne de rendez-vous de la table **rdv**.

```
DELETE FROM rdv WHERE id = ?
```

- Si une erreur se produit lors de la mise à jour de la disponibilité du véhicule ou de la suppression du rendez-vous, un message d'erreur est affiché.
- Si la suppression est effectuée avec succès, un message "Suppression effectuée avec succès." est affiché.
- La connexion au serveur de base de données est fermée.
- La fonction **ob_end_clean()** est utilisée pour s'assurer qu'aucun contenu HTML n'est envoyé avant la redirection.

- Enfin, l'utilisateur est redirigé vers la page précédente (**update_bdd.php**) avec l'e-mail du client en tant que paramètre GET.

```
// Rediriger vers la page précédente
header("Location: ../php/update_bdd.php?Filtre=" . urlencode($email));
exit();
```

Conclusion et Remerciements

Au cours de ce projet, nous avons beaucoup appris en termes de programmation, d'interrogation en base de données, de travail en groupe et de pouvoir gérer son projet sur GitHub. Nous vous remercions d'avoir lu ce document pour mieux comprendre l'élaboration de ce projet.