

NOM	GUTIERREZ
Prénom	Morgane
Date de naissance	13/06/1995

Copie à rendre

Graduate Développeur

(Android, Angular, Flutter, Front End, Full Stack, IOS, PHP/Symfony)

Documents à compléter et à rendre

Lien du site : <https://garage-v-parrot.codecreator.fr/>

L'espace pro est accessible :

- Admin : v.parrot@example.com et mdp : Jaimelesvoituresdu31100
- User : studi@example.com et mdp : NR7dxs642dZZ9

Lien du dépôt Git : [https://github.com/Morgane33127/Garage Parrot](https://github.com/Morgane33127/Garage_Parrot)

Lien du Trello : <https://trello.com/b/0GWKyqaB/garage-v-parrot>

Liste des compétences couvertes par le projet

Ce projet couvre les compétences professionnelles suivantes :

Activité – Type 1 : Développer la partie front-end d’une application web ou web mobile en intégrant les recommandations de sécurité

- 1 Maquetter une application
- 2 Réaliser une interface utilisateur web statique et adaptable
- 3 Développer une interface utilisateur web dynamique
- 4 Réaliser une interface utilisateur avec une solution de gestion de contenu ou e-commerce

Activité – Type 2 : Développer la partie back-end d’une application web ou web mobile en intégrant les recommandations de sécurité

- 5 Créer une base de données
- 6 Développer les composants d’accès aux données
- 7 Développer la partie back-end d’une application web ou web mobile
- 8 Élaborer et mettre en œuvre des composants dans une application de gestion de contenu ou e-commerce

Résumé du projet

Mr Parrot, chef du garage toulousain « Garage V. Parrot », souhaite développer sa visibilité sur internet afin de se démarquer de la concurrence grâce à un nouveau site vitrine. Fort de 15 ans d’expérience, le garage de Mr Parrot propose depuis 2 ans une large gamme de prestations allant de la réparation à la vente de véhicules d’occasions. Le projet consiste à créer une application web vitrine pour le Garage V. Parrot et se divise en deux axes de développement.

Le premier axe consiste à faire ressortir sur le site les aspects « confiance », « professionnalisme » et « qualité » chers à Mr Parrot, et à mettre en avant la gamme de prestations proposées par le garage, notamment la vente de véhicules d’occasion. La mise en avant des avis clients sur le site et la possibilité de laisser un avis constituent donc un point essentiel afin de mettre en avant les valeurs du garage. De plus, afin de développer le chiffre d’affaires de Mr Parrot, il est indispensable de faciliter la recherche d’un véhicule. Pour ce faire, l’application doit permettre d’appliquer différents filtres dont le prix, le kilométrage et l’année de mise en circulation.

Le second axe consiste à mettre en place un espace professionnel sécurisé avec des accès distincts pour les utilisateurs (employés et administrateur) permettant de gérer les différents contenus de l’application en fonction de leurs droits. Cet espace doit donc être accessible par un formulaire de connexion robuste afin d’éviter toute faille potentielle.

Le projet a été réalisé sur une période de deux mois et demi s’étendant de fin octobre à mi-janvier par l’agence de création de sites web missionnée par Mr Parrot et dont je fais partie.

Cahier des charges (CDC)

Mr Parrot désirait sur son site les fonctionnalités suivantes :

- Se connecter à un espace professionnel grâce à l’association email/mot de passe avec une restriction des fonctionnalités pour les employés et un contrôle total par l’administrateur (Mr Parrot).
- Ajouter un utilisateur à l’espace professionnel par uniquement le rôle administrateur.
- Mettre en avant sur la page d’accueil les différentes prestations du garage.

- Pouvoir gérer les différentes prestations du garage depuis l'espace pro exclusivement par l'admin.
- Mettre en avant les horaires du garage sur le pied de page de toutes les pages.
- Pouvoir mettre à jour les horaires du garage depuis l'espace pro exclusivement par l'admin.
- Présenter sur le site les voitures à la vente avec à minima pour chaque véhicule : le prix, l'image, le kilométrage et l'année de mise en circulation.
- Pouvoir gérer les véhicules à la vente depuis l'espace pro par n'importe quel employé du garage et l'admin.
- Faciliter la recherche d'un véhicule grâce à un système de filtres fonctionnant sans rechargement de la page. Les filtres applicables devront être : une fourchette de prix, un nombre de kilomètres parcourus, une année de mise en circulation.
- Créer une page de contact permettant de contacter facilement l'atelier par téléphone ou via un formulaire de contact nécessitant nom, prénom, adresse e-mail, numéro de téléphone et un message.
- Pouvoir contacter l'atelier sous chaque annonce d'un véhicule d'occasion avec le sujet reprenant automatiquement le titre de l'annonce.
- Mettre en avant les avis clients approuvés sur la page d'accueil.
- Permettre à un visiteur de laisser un avis grâce à son nom, prénom, commentaire, et une note.
- Permettre aux employés et à Mr Parrot de modérer et d'ajouter un avis depuis l'espace pro.

Spécifications techniques

Architecture logicielle

Mr Parrot n'étant pas du tout développeur la conception du site devait être claire et facile à maintenir, c'est pourquoi j'ai choisi l'architecture en Modèle Vue Contrôleur.

Accessibilité

Le site vitrine doit parfaitement être responsive. J'ai donc utilisé le Framework Bootstrap et des media queries.

Sécurité

Le site vitrine et la partie « espace pro » devaient être robuste. J'ai donc mis en place une validation et un nettoyage des entrées côté serveur grâce à des fonctions dédiées, limité les champs par l'utilisation d'attributs « max » et « min » et mis en place des requêtes préparées.

Pour accéder à l'espace pro j'ai mis en place : des UUID plus sécurisés que des ID classiques, un token CSRF, des mots de passe forts obligatoires et une vérification préalable de l'email dans la base de données.

Spécifications tierces

Lorsqu'une action est effectuée ou une erreur rencontrée une alerte doit s'afficher. Pour cela j'ai mis en place des messages de session et un fichier de log.

Veille sécurité

Afin de mettre en place le jeton CSRF j'ai réalisé des recherches sur les sites anglophones suivants :

- <https://www.phptutorial.net/php-tutorial/php-csrf/>

Afin de mettre en place les UUID j'ai utilisé la documentation de MariaDB :

- <https://mariadb.com/kb/en/uuid-data-type/>

Afin de vérifier la solidité d'un mot de passe j'ai suivi les recommandations de la CNIL et je me suis inspirée du code ci-dessous (site non anglophone):

- <https://m-gut.developpez.com/tutoriels/php/verif-password/>

Problématique rencontrée et veille effectuée

Lors de la conception du site j'ai rencontré des difficultés à afficher les données filtrées de manière asynchrone. J'ai donc réalisé une veille sur l'utilisation de l'API Fetch en javascript. En effet je recherchais comment afficher mes voitures en récupérant le contenu de mes vues en JSON. Je n'avais que très peu utilisé l'asynchrone jusque-là. J'ai donc recherché « response.json() return a text fetch php ». La problématique « Getting a text respons using Fetch API and a PHP file » rencontrée à l'adresse suivante correspondait donc parfaitement à ma problématique : <https://stackoverflow.com/questions/67864999/getting-a-text-respons-using-fetch-api-and-a-php-file>

EN

Q : « I am hitting a php file and the response returns a string. I can't find the documentation on how to deal with this.

From the google developer doc google developer doc, my fetch is simple. [...] This fails on the response.json() assumedly because I am returning just a string. If I log out just the response (attached screen shot) I get something I have no idea how to work with, and I cannot convert it. Where is my return value from the PHP here?»

R : « Your PHP isn't returning JSON, it's just returning a string so use .text() instead. »

FR

Q : « La réponse de mon fichier PHP est une chaîne et je n'arrive pas à trouver de documentation sur ce point. À partir de la documentation Google, ma récupération est simple. L'échec provient de la réponse.json() probablement parce que je ne renvoie qu'une chaîne. Si je regarde la réponse (capture d'écran ci-jointe), j'obtiens quelque chose avec lequel je ne sais pas comment travailler et je ne peux pas le convertir. Où est ma valeur de retour du PHP ici ? ».

R : « Votre PHP ne renvoie pas de JSON, il renvoie simplement une chaîne, alors utilisez plutôt .text(). »

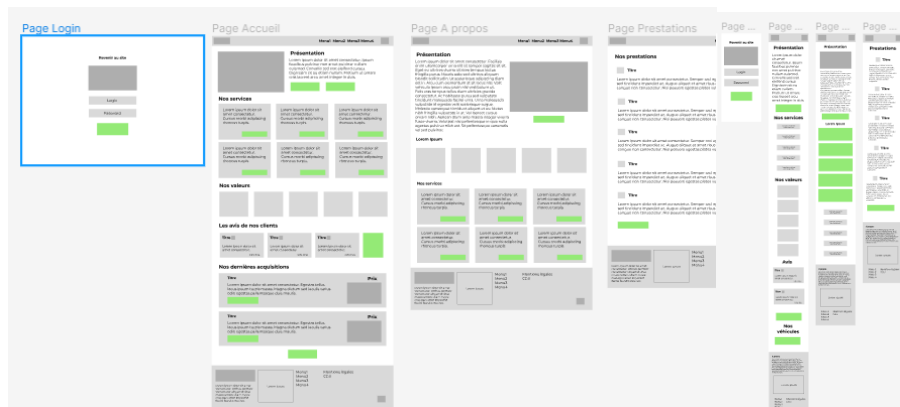
J'ai donc simplement transformé mon response.json() en response.text() pour afficher mes « data » dans ma div.

Réalisations

Activité – Type 1 : Développer la partie front-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Après avoir compris le besoin client et réalisé les différents diagrammes nécessaires (voir documentation technique) j'ai pu poursuivre le processus de design thinking en réalisant les interfaces utilisateurs du site. Pour cela j'ai utilisé le site Figma pour réaliser les wireframes et mockups du site pour les équipements desktops et mobile en tenant compte des règles d'accessibilité. J'ai notamment vérifié mes scores de contrastes grâce au « color-contrast-analyzer » de Adobe. J'ai décidé de conserver le logo et la charte fournie car je les trouvais très adaptés à l'activité automobile. En effet, j'ai constaté de mes recherches sur la concurrence que le rouge, blanc, gris et noir étaient souvent choisis dans ce type de métier. De plus, dans la signification des couleurs, le rouge est symbole de passion, comme celle de Mr Parrot, et le blanc et noir symboles de qualité. Mais, j'aurai pu choisir du bleu pour la confiance.

Wireframes :



Une fois la maquette terminée j'ai pu commencer la phase de codage avec mon IDE VSCode. Afin de réaliser mes pages je les ai structurées selon les recommandations du W3C et j'ai utilisé le framework Bootstrap pour faciliter leur adaptation sur mobile. J'ai téléchargé les fichiers minifiés du CDN pour ne pas empêcher l'exécution du site si des mises à jour Bootstrap venaient à être réalisées. J'ai aussi utilisé le langage de script préprocesseur SASS afin de réaliser mon fichier de styles personnels. J'ai notamment utilisé les propriétés de CSS3 afin de réaliser les border radius et les media queries qui permettent d'appliquer des styles en fonction de la taille de l'écran.

```

1  <!DOCTYPE html>
2  <html lang="fr">
3
4  <head>
5      <meta charset="UTF-8">
6      <meta name="description" content="Garage V. Parrot :">
7      <meta name="keywords" content="Garage, V.Parrot, Entretien, Réparation, Révision, Vidange, Véhicules d'occasions">
8      <meta name="author" content="MG">
9      <meta name="viewport" content="width=device-width, , initial-scale=1.0">
10     <title>Garage V. Parrot</title>
11     <link rel="stylesheet" href="public/assets/css/bootstrap.min.css">
12     <link rel="stylesheet" href="node_modules/bootstrap-icons/font/bootstrap-icons.min.css">
13     <link rel="stylesheet" href="public/assets/css/style.css">
14     <script src="public/assets/js/bootstrap.bundle.min.js" type="text/javascript"></script>
15     <script src="public/assets/js/jquery-3.7.1.min.js" type="text/javascript"></script>
16     <script src="public/assets/js/script.js" type="text/javascript" defer></script>
17
18 </head>
19
20 <body>

```

Afin d'apporter de l'interaction coté client au niveau de l'affichage des voitures j'ai mis en place des inputs de type range pour borner les prix et les kilomètres et un select pour l'année maximale de mise en circulation. Afin d'afficher les requêtes de manière asynchrone j'ai utilisé l'API Fetch de javascript. Comme je ne savais pas m'en servir j'ai dû mener une veille. Pour tester mon script j'ai utilisé la commande console.log() ainsi que l'inspecteur du navigateur, plus particulièrement l'onglet « Réseau ». J'ai rédigé mes scripts en suivant les recommandations ECMAScript 6.

```
function changeVoitures(priceMinValue, priceMaxValue, kmMinValue, kmMaxValue, dateValue) {
```


```
    if(dateValue === ''){
        const d = new Date();
        dateValue = d.getFullYear();
    }

    fetch('src/affichageVoitures.php', {
        method: 'POST',
        headers: {
            'Content-Type': 'application/json'
        },
        body: JSON.stringify({ 'prixmin': priceMinValue, 'prixmax': priceMaxValue, 'kmmin' : kmMinValue,
            'kmmax' : kmMaxValue, 'date': dateValue })
    })
    .then(response => {
        if (!response.ok) {
            throw new Error('Erreur lors de la requête.');
```

Name	Sta...	Ty...	Initiator	Size	Ti...	Waterfall
affichageVoitu...	200	fet...	script.js...	76...	23 ...	

Le cahier des charges explicitait la gestion du contenu du site grâce à un espace professionnel uniquement accessible aux employés du garage et à Mr Parrot. Pour ce faire j'ai créé une page de connexion permettant la redirection de l'utilisateur vers l'interface d'administration du site. Grâce à cet espace, et à la mise en place de droits et de rôles, l'utilisateur peut en fonction de son rôle créer, gérer et publier du contenu sur le site. De cette manière, avec son rôle admin, Mr Parrot dispose d'un contrôle total du site contrairement à ses employés. Mais tous peuvent par exemple modérer les avis et ajouter une voiture :

Revenir au site >>



Garage V. Parrot

Mot de passe oublié ?
Changement de mot de passe ?

Garage V. Parrot

Tableau de bord

- Voitures
 - Liste
 - Ajouter
- Prestations
- Avis
- Horaires
- Utilisateurs

Avis

Première visite. ✓ ✗

2024-01-14 13:35:24

★★★★★

Rdv rapide, prix corrects et le patron connaît son travail. J'y retournerai.

Petit Sandrine

Super garage ✓ ✗

2024-01-14 13:35:24

★★★★★

Super garage, très pro, personnel et patron au top.

Derat Florient

Activité – Type 2 : Développer la partie back-end d'une application web ou web mobile en intégrant les recommandations de sécurité

Une des premières étapes de codage a été de créer la base de données à partir du schéma physique établi. En effet cela était indispensable pour afficher les contenus : voitures, avis, prestations... Pour cela j'ai généré et exécuté un script de création (« create_database.php ») et un script d'alimentation de la base de données (« insert_data.php ») avec mon IDE VSCode. Pour faciliter la création de la base j'appelle le script d'alimentation directement dans mon script de création si toutes les conditions sont remplies. Afin de faciliter une exécution en local j'ai rédigé un fichier README explicitant la démarche.

```
<?php
include_once '../config/Database.php';
require '../config/functions.php';

/* EN : Database and tables creation and add data
FR : Creation de la base de donnée, creation des tables et ajouts de données
*/

try {
    $db = new Database();
    $pdo = $db->createDatabase();

    if ($pdo->exec('DROP DATABASE IF EXISTS garageParrot') !== false) {
        if ($pdo->exec('CREATE DATABASE garageParrot') !== false) {
            $newpdo = $db->getConnection();

            $uti = 'CREATE TABLE utilisateurs (
id_u UUID NOT NULL,
prenom_u VARCHAR(30) NOT NULL,
nom_u VARCHAR(30) NOT NULL,
role_u CHAR (3) NOT NULL,
login_u VARCHAR(150) NOT NULL,
mdp_u VARCHAR(100) NOT NULL
)';

            $l1 = 'CREATE TABLE l1 (
id_l1 INT PRIMARY KEY AUTO_INCREMENT,
code_l1 CHAR(3) NOT NULL,
l1 VARCHAR(100) NOT NULL
)';

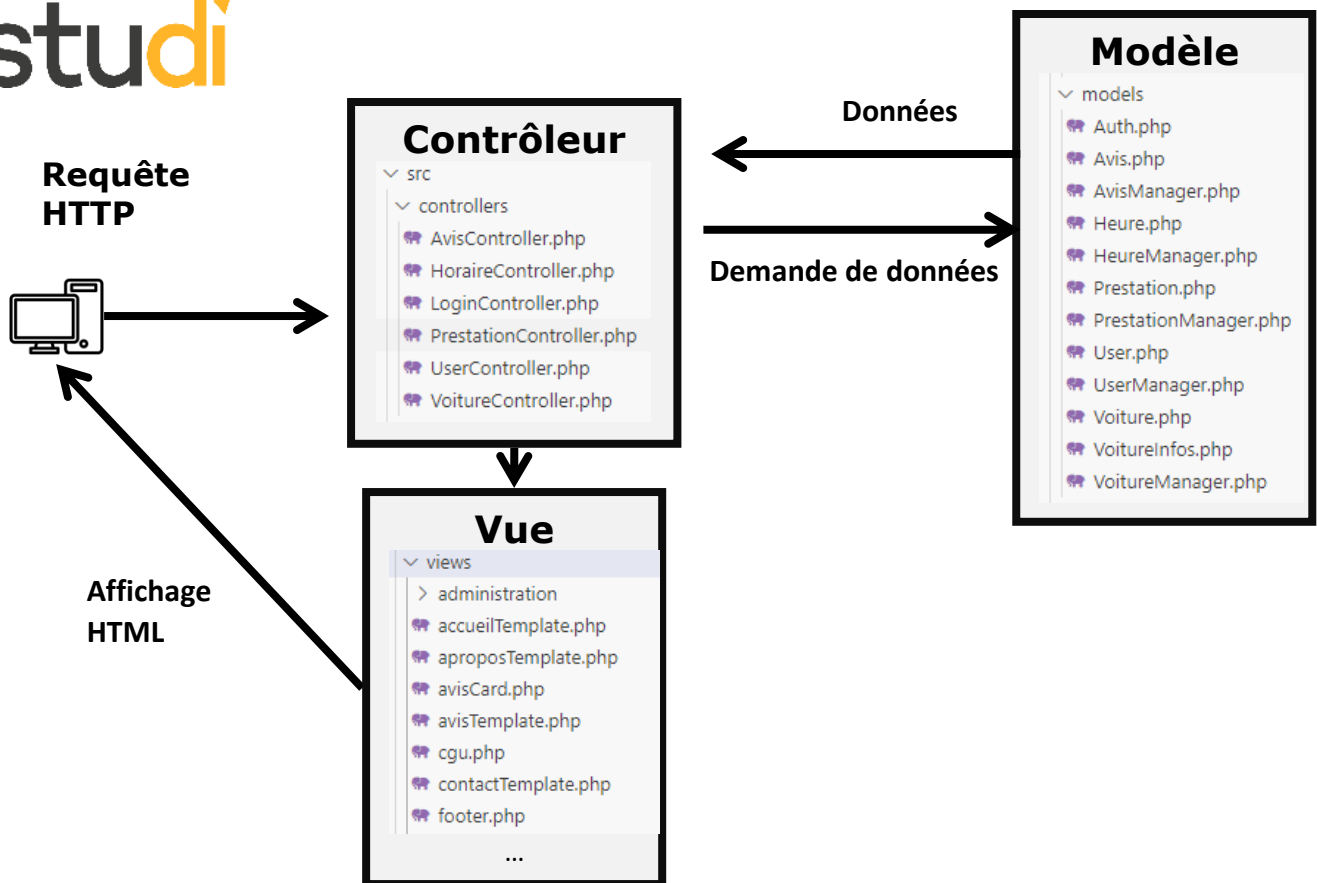
            $heures = 'CREATE TABLE heures (
id_h INT PRIMARY KEY AUTO_INCREMENT,
jour CHAR(3) NOT NULL,
hr_debut CHAR(5) NOT NULL,
hr_fin CHAR(5) NOT NULL
)';

            $pdo->exec("INSERT INTO utilisateurs (id_u, prenom_u, nom_u, role_u, login_u, mdp_u)
VALUES ('$suid', 'Vincent', 'Parrot', 'ADM', 'v.parrot@example.com', '$mdp')") &&

            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('LUN', '08:00', '18:00')") &&
            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('MAR', '08:00', '18:00')") &&
            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('MER', '08:00', '18:00')") &&
            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('JEU', '08:00', '18:00')") &&
            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('VEN', '08:00', '18:00')") &&
            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('SAM', '08:00', '18:00')") &&
            $pdo->exec("INSERT INTO heures (jour, hr_debut, hr_fin) VALUES ('DIM', 'Ferme', 'Ferme')") &&
```

Une fois la base de données créée j'ai choisis un motif d'architecture logicielle en Modèle Vue Contrôleur (MVC) pour séparer la logique métier de l'affichage. Pour ce faire, j'ai créé une classe Database et mes différentes classes (Avis, Voiture, Heure, Prestation, Utilisateur). La classe Database permettant, d'une part, la création de la base de données, et d'autre part, la connexion à la base de données. Pour accéder à la base de données j'ai mis en place les modèles « ClassManager » (ex : AvisManager) qui s'articulent avec les vues grâce aux contrôleurs « ClassController » (Ex : AvisController). Grâce à mes managers j'ai créé mes requêtes CRUD de manière à ne pas introduire de vulnérabilité dans le système d'information :

- Des requêtes de sélection pour afficher les voitures, les avis, les prestations
- Des requêtes de création pour ajouter des avis, des voitures, des utilisateurs, ...
- Des requêtes de mise à jour coté administration pour changer les horaires, les informations d'un véhicule, les prestations
- Des requêtes de suppression coté administration pour supprimer une voiture, un utilisateur, une prestation



Par exemple, pour accéder à l'espace pro du garage V. Parrot j'ai créé un contrôleur « LoginController » permettant de lier la classe « Auth » et la vue de login « loginForm ». La connexion à l'interface est codé dans un style défensif grâce au contrôleur « LoginController » permettant en premier de lieu une vérification des entrées utilisateurs grâce aux fonctions dédiées : `verifMail()` et `verifData()`. Une fois les données vérifiées la classe « Auth » permet de vérifier l'existence de l'utilisateur en base de données puis la conformité du mot de passe saisie. La classe « Auth » renvoie un objet « User » afin de générer le lien de redirection et la gestion des sessions. Afin de renforcer la sécurité j'ai choisi de mettre en place des UUID et un token CSRF. Les UUID étant moins « devinables » que les id incrémentales classiques. Pour les utiliser j'ai dû installer la librairie PHP « ramsey/uuid » grâce à Composer. Lorsque les données entrées sont erronées ou qu'une requête n'aboutit pas sur le site une exception est lancée puis saisie par un bloc « try/catch » avant d'être affichée comme message de session.

```
models > Auth.php > PHP Intelephense > Auth > login

/**
 * Verifications of email and password to enter to the administration area.
 *
 * @param string $mdp password
 * @param string $login email
 * @return object User
 * @access public
 */
public function login(string $mdp, string $login): User
{
    $stmt = $this->conn->prepare('SELECT * FROM utilisateurs WHERE login_u = :email');
    $stmt->bindValue(':email', $login);
    $stmt->execute();
    $user = $stmt->fetchAll(PDO::FETCH_ASSOC);
    if (count($user) > 0) {
        $newUser = new User($user[0]['id_u'], $user[0]['prenom_u'], $user[0]['nom_u'], $user[0]['role_u'], $user[0]['login_u'], $user[0]['mdp_u']);
        $id_u = $newUser->getId();
        if (!empty($newUser) && password_verify($mdp, $newUser->getPassword()) == true) {
            $event = $this->conn->prepare('INSERT INTO evenements (id_u, info_e) VALUES (:id_u, :info_e)');
            $event->bindValue(':id_u', $id_u);
            $event->bindValue(':info_e', 'Nouvelle connexion');
            $event->execute();
            return ($newUser);
        } else {
            throw new Exception('Mot de passe incorrect.');
```


L'espace pro du site permet la gestion du contenu grâce à des formulaires et tableaux comportant tous les champs nécessaires pour ajouter un nouveau véhicule, un avis, une prestation... Une fois un formulaire soumis les informations sont récupérées par le fichier « demande.php » qui traite les données recueillies et les ajoute dans la base de données grâce à la méthode spécifique comme « ajouterVoiture » du composant « VoitureController ». La méthode « affichageCard » de ce même composant permet alors d'afficher le véhicule sur le site. Coté espace pro l'utilisateur a la possibilité de modifier ou supprimer un enregistrement d'une entité grâce à des méthodes dédiées comme « supprimerPrestation » ou « modifierPrestation ». Afin de fonctionner, les composants nécessitent d'accéder à la base de données et sont donc reliés à un composant externe d'accès à la base de données : la classe Database.

```
//Add a car
try {
    if (isset($_POST['ajouterVoiture'])) {
        if (!empty($_FILES['img']['name'])) {
            uploadFile('img');
            $img = $_FILES['img']['name'];
            if (file_exists("../public/assets/img/$img" === true)) {
                $img = $_FILES['img']['name'];
            }
        } else {
            $img = 'gvplogo.svg';
        }
        $titre_v = $_POST['titre_v'];
        $petite_description_v = $_POST['petite_description'];
        $large_description_v = $_POST['large_description'];
        $marque = $_POST['marque_v'];
        $modele = $_POST['modele_v'];
        $prix = $_POST['prix_v'];
        $annee = $_POST['annee_v'];
        $kilometre = $_POST['km_v'];
        $statut = 'Dispo';
        $type = $_POST['type_v'];
        $carburant = $_POST['carburant'];
        $couleur = $_POST['couleur_v'];
        $nbportes = $_POST['nbportes'];
        $nbplaces = $_POST['nbplaces'];
        $puissance_fiscale = $_POST['cv'];

        $donnees = array(
            0 => $titre_v, 1 => $petite_description_v, 2 => $large_description_v, 3 => $marque, 4 => $modele, 5 => $prix, 6 => $img, 7 => $annee,
            8 => $kilometre, 9 => $statut, 10 => $type, 11 => $carburant, 12 => $couleur, 13 => $nbportes, 14 => $nbplaces, 15 => $puissance_fiscale
        );
        $connection = new VoitureController();
        $voiture = $connection->ajouterVoiture($donnees);
        sessionAlert('success', 'Voiture ajoutée avec succès!');
        header('Location: index.php?page=administr&div=voitures');
    }
} catch (Exception $e) {
    error($e->getMessage());
    sessionAlert('danger', $e->getMessage());
    header('Location: index.php?page=administr&div=voitures');
}
```

Tout le long de la phase de développement le logiciel Git m'a permis de créer des versions grâce aux commits et des branches pour créer les fonctionnalités. Les classes, requêtes et scripts ont été documentés afin de maintenir facilement le site.

Afin de vérifier que l'« espace pro » soit bien sécurisé j'ai effectué un test d'intrusion pour chaque système de protection mis en place. Ainsi, j'ai testé l'utilisation d'un mauvais mail, puis d'un mauvais mot de passe, de changer un UUID dans la barre URL et de changer le jeton CSRF.

Conclusion

Le site proposé à Mr Parrot est un site sécurisé au design épuré proposant toutes les fonctionnalités attendues et mettant en avant les valeurs du garage. La navigation est facile, ainsi que sa maintenance grâce à l'architecture en MVC, aux technologies utilisées et aux commentaires. On peut cependant noter dans le processus d'amélioration continue un axe d'amélioration majeur : la mise en place d'une table d'événements permettant de tracer l'activité sur le site.