Création d'un CRUD PHP PDO

- Connexion à la BDD
- Affichage des données
- Insertion de données
- Modification de données
- Suppression des données
- Création Utilisateur
- Hashage mot de passe
- Formulaire Connexion
- Sécurisation des pages

L’objectif est de créer un site web simple qui permet de gérer des informations sur des voitures

Une voiture contiendra:

- Un modèle (texte)
- Une marque (texte)
- Un nombre de chevaux (Entier)
- Une image (texte, nom du fichier)

L’objectif est de pouvoir:

- consulter la liste des voitures et leurs informations.
- ajouter une voiture
- modifier une voiture
- supprimer une voiture

Par la suite nous sécuriseront les pages d’administration avec une connexion utilisateur

Création d'une base de données avec la table Voiture

* id int	* modele varchar(255)	* marque varchar(255)	* chevau: int	* date_creation date	* image varchar(255)
1	Model S	Tesla	670	2022-01-01	model_s.jpg
2	Civic	Honda	158	2020-05-15	civic.jpg
3	Golf	Volkswagen	150	2021-08-23	golf.jpg
4	Mustang	Ford	450	2019-03-10	mustang.jpg
5	911 Carrera	Porsche	379	2022-02-11	911_carrera.jpg

```
CREATE DATABASE garage12;
USE garage12;
 CREATE TABLE car ( id INT AUTO_INCREMENT
PRIMARY KEY,
model VARCHAR(255) NOT NULL,
brand VARCHAR(255) NOT NULL,
horsePower INT NOT NULL,
image VARCHAR(255) NOT NULL
);
```

Fichiers

Préparation du projet avec les blocs **header footer** dans les pages.

index.php page d'accueil affichage des voitures

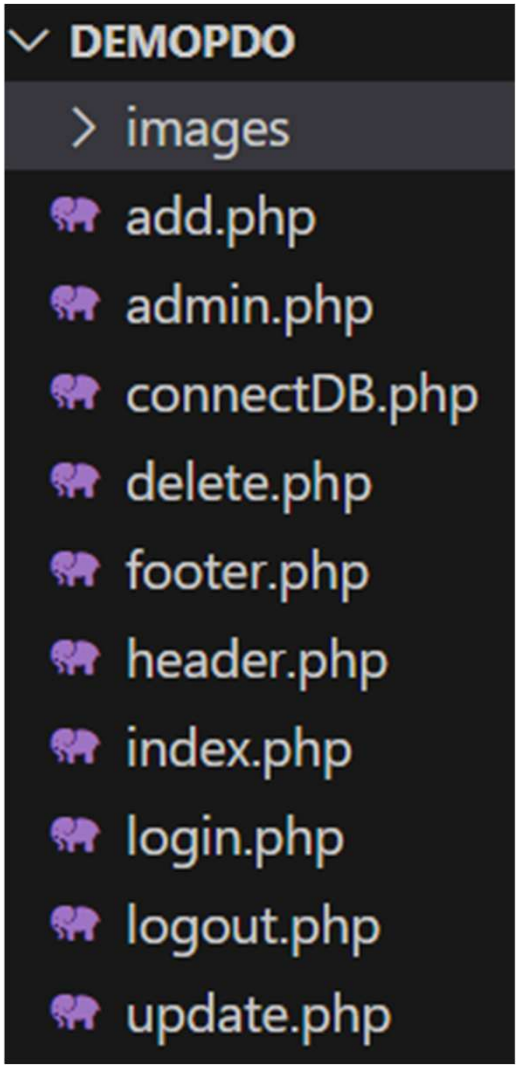
connectDB.php fonction permettant la connexion a la BDD

add.php page contenant le **formulaire d'ajout** de voiture
et son **traitement**

update.php **formulaire d'update** d'une voiture par ID
et son **traitement**

delete.php **formulaire de suppression d'une voiture par ID**
et son traitement

admin,login,logout seront réservés à la partie sécurisation



connectDB

Création de la fonction connectDB()

Modification des variables selon le projet

Import du fichier et appel de la fonction dans les fichiers ayant besoin d'une requête BDD

```php
function connectDB(): PDO
{
    $host = 'localhost';
    $dbName = 'testDB';
    $user = 'root';
    $password = '';

    try {
        $pdo = new PDO(
            'mysql:host=' . $host . ';dbname=' . $dbName . ';charset=utf8',
            $user,
            $password
        );
        $pdo->setAttribute(
            PDO::ATTR_ERRMODE,
            PDO::ERRMODE_EXCEPTION
        );
        $pdo->setAttribute(
            PDO::ATTR_DEFAULT_FETCH_MODE,
            PDO::FETCH_ASSOC
        );

        return $pdo;
```

Affichage des données dans l’index

Index.php

1) Récupération des données

- Création de la connexion à la BDD
- Requête SQL « **selectAll** »
- Récupération des données dans **$cars**

2) Affichage des données

- Boucle foreach pour parcourir le tableau
- Affichage de chaque champ dans la balise associée

Formulaire d’ajout

add.php

Ajout de données

- Ajout d'un lien dans **index.php** vers **add.php**

```
<a href="add.php">Ajouter une voiture</a>
```

add.php contiendra:

- Création d'un formulaire d'ajout
- Validation des données
- Ajout en base données
- Gestion des erreurs

Exemple de validation avec **empty()** des données du formulaire et créations d'erreurs

Le tableau $errors permettra d'afficher une erreur sous chaque champ du formulaire

```php
// stockage des erreurs dans un tableau
$errors = [];
// si la méthod est "POST" ( validation du formulaire ), on passe au traitement
if ($_SERVER['REQUEST_METHOD'] === 'POST') {
    // s'il manque un champ nom/prénom/age/position, il y aura une erreur
    // tester chaque champs un à un, meilleure gestions des erreurs
    if (empty($_POST['model'])) {
        $errors['model'] = 'Le modèle ne peut pas être vide.';
    }
```

1. Créer le formulaire d'ajout de Voiture
2. Ajouter cette logique au dessus du formulaire
3. Créer la vérification des autres champs

Si le tableau contient une erreur celle-ci sera afficher sous le champ correspondant.

Exemple avec le champ **brand:**

```
<input type="text" name="brand" required>
<?php if(isset($errors['brand'])): ?>
    <p class="text-danger"><?=$errors['brand'] ?></p>
<?php endif; ?>
```

1. **Ajouter cette logique sous chaque champ du formulaire**

add.php

1. Vérification de la présence d'erreurs après les tests (tableau $errors vide)
   Après les vérifications et la création des erreurs

```php
// s'il n'y a pas d'erreur...
if (empty($errors)) {
    // On appel maintenant connectDB.php et instanciation de la connexion a la BDD a
    require_once("connectDB.php");
    $pdo = connectDB();
    // ... la requête préparée peut être exécutée
```

2. Ajouter le code d'insertion en BDD avec une requête préparée et les données du formulaire

3. Redirection vers l'index

Formulaire d’édition

update.php

Edition de données

1) Ajout d'un lien sous chaque « données » dans l'index vers la page **update.php** et l'**ID** en paramètre **GET**

```php
<?php foreach ($cars as $car): ?>
    <div class="p-3 d-flex justify-content-center">
        <img src="images/<?php echo $car["image"] ?>" alt="<?php echo $car["model"] ?>">
        <h2><?php echo $car["model"] ?></h2>
        <p><?php echo $car["brand"] ?></p>
        <p><?php echo $car["horsePower"] ?> chevaux</p>
        <a href="update.php?id=<?= $car["id"] ?>" class="btn btn-primary">Modifier</a>
    </div>
<?php endforeach; ?>
```

2) Afficher $_GET dans **update.php** pour vérifier le lien

Formulaire update

1) Vérification de la présence de l'**ID** en paramètre url GET

2) Récupération de la voiture en BDD correspondant à l'**ID** reçu

3) Création du formulaire d'édition, les champs doivent être prérempli avec les données de la voiture récupérée.

4) Ajout du traitement du formulaire, semblable à celui de add.php

- Vérification des champs
- Créations d'erreurs
- Création de la requête UPDATE avec les données du formulaire, pour mettre à jour la voiture

Formulaire de suppression

delete.php

delete.php

Ajout d'un **lien** dans **l'index** sous chaque « données » contenant **l'ID** de la ligne à supprimer en paramètre GET

Même principe que le lien **update**.

delete.php

1. Vérification de la présence de l'ID en paramètre url GET
2. Récupération de la voiture en BDD correspondant à l'**ID** reçu
3. Affichage d'un texte de confirmation avec le modèle et la marque de la voiture

delete.php

Formulaire DELETE:

Utilisation de l'attribut html **formaction** pour avoir une **action supplémentaire** dans le formulaire

Dans notre cas **annuler** la suppression et **rediriger** a l'index

```
<form method="POST" action="delete.php?id=<?= $_GET["id"] ?>">
    <button class="btn btn-danger">Delete</button>
    <button class="btn btn-secondary" formaction="index.php">Cancel</button>
</form>
```

1 . Création du formulaire DELETE

2. Tests de la création, édition, suppression d'une Voiture via les formulaires.

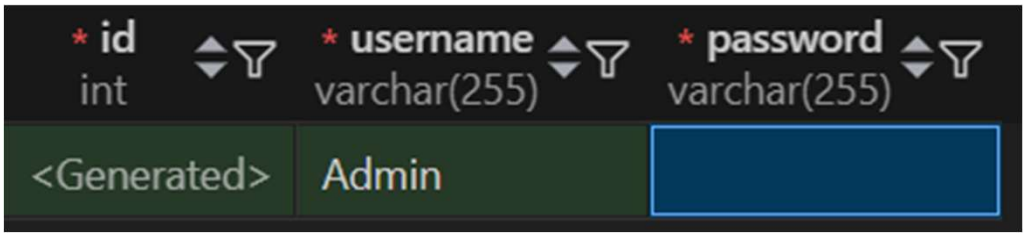
Logique de connexion utilisateur

Création d'un utilisateur

Nous aurons besoin d'une table en base de données pour stocker nos User

- User :id, username (varchar), password (varchar)

```
CREATE TABLE User ( id INT AUTO_INCREMENT PRIMARY KEY,
username VARCHAR(255) NOT NULL,
password VARCHAR(255) NOT NULL);
```



Les mots de passe stockés en BDD doivent toujours être **crypter** ou **hasher**

Encodage d'un mot de passe

La fonction **password_hash()** permet de hasher une chaine de caractère (1er paramètre « admin »)

selon un algorithme de hashage (2ième paramètre PASSWORD_DEFAULT)

```php
$pass = password_hash("admin",PASSWORD_DEFAULT);
var_dump($pass);
```

1) Ajouter ces lignes dans **login.php,**

aller dans la page login.php dans le navigateur et copier $pass

2) Ajouter un utilisateur en BDD et copier coller le mot de passe généré dans le champ password

3) Commenter les lignes garder le mot de passe en clair et le code

Formulaire de connexion **login.php**


```php
<form method="POST" action="login.php">
    <label>Username</label>
    <input required type="text" name="username">
    <label>Password</label>
    <input required type="password" name="password">

    <?php ···
    <button class="btn btn-outline-success">Se connecter</button>
</form>
```

Logique de connexion

Après validation du formulaire:

1. Recherche en base de données de la correspondance d'un utilisateur avec le « username » du formulaire

1. Si un utilisateur avec le même username est trouvé en BDD:

3. Vérification du mot du mot de passe provenant du formulaire et celui en BDD avec cette fonction:

```php
password_verify($_POST["password"], $res["password"])
```

Renvoie TRUE si les mots de passe correspondent

4. Si les vérifications sont correctes, connexion de l'utilisateur:

Enregistrement en **Session** de son **Username**
Redirection **admin.php**

```php
session_start();
$_SESSION["username"] = $user["username"];
header('Location: admin.php');
```

Sécurisation des pages

Sécurisation de pages

Vérification de la l'existence de **$_SESSION[« username »]**, preuve que l'utilisateur est **connecté**

Ce code vérifie la présence de la donnée username et redirige si elle est manquante
Il servira à vérifier si l'utilisateur est connecté ou non et sécurisera les pages d'administration des données

```php
if (!isset($_SESSION["username"])) {
    header("Location: index.php");
}
```

logout.php

Nous ajouterons un lien vers la page **logout.php** dans la navbar

Qui supprimera la Session, ce qui « déconnecte » l'utilisateur.

Et redirigera vers **index.php**

```php
if(isset($_SESSION["username"])){
    //Supprime une variable
    unset($_SESSION["username"]);
}

//Vide les données de la session en conservant la même session
//session_reset();

//Détruit la session totale, une nouvelle session sera créée avec un nouvelle ID
//session_destroy();

header("Location: index.php")
```

Sécurisation de pages

1 - Copier le code de **index.php** dans **admin.php**

2 - Ajouter le code de vérification de Session en haut de la page **admin.php**

```php
if (!isset($_SESSION["username"])) {
    header("Location: index.php");
}
```

3 - Essayer d'aller dans la page **admin.php** en étant connecté et déconnecté

4 - Ajouter également la vérification de Session dans les pages **add.php, update.php, delete.php**

5 - Suppression des liens vers les pages **add.php, update.php, delete.php** dans **index.php**

6 – Vérification de l'accès aux formulaires sans être connecté

Exportation de la base de données en fichier SQL

Améliorations

Création d'un ficher **functions.php** contenant les fonctions suivantes:

selectAllCars()

selectCarByID()

insertCar()

updateCar()

deleteCar()

verifySession() – Qui contiendra la vérification de la session et la redirection

Des fonctions des validation pour le code répétitif dans add et update

Améliorations

Création de templates pour les formulaires pour simplifier la lecture du code des fichiers.

Création d'un formulaire de création d'utilisateur:

- Hashage du mot de passe avant persistance en BDD

Ajout de l'upload de fichiers pour l'image (fin pdf data transmission)

Création d'affichages des données filtrée :

- Un nombre de chevaux minimum
- Formulaire de recherche par marque
- Recherche par marque