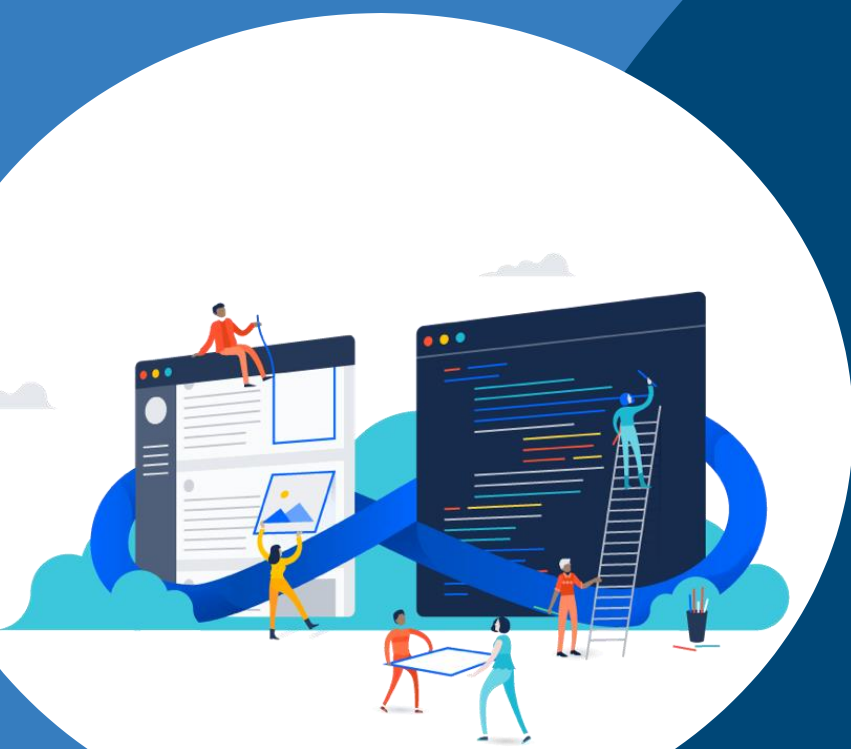
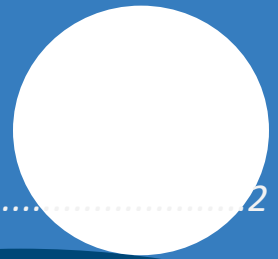


Cahier des charges TP PHP

Loc Auto



Sommaire



<i>Sommaire</i>	2
<i>Consignes</i>	4
Mission 1: étude et cahier des charges.	4
Mission 2: réalisation du site en PHP	4
<i>Conception du site</i>	5
Structure du MCD et du MLD.....	5
MCD	5
.....	6
MLD	6
.....	7
<i>Maquette du site</i>	7
Première maquette	7
<i>Réalisation et fonctionnement du site</i>	8
Frontend.....	9
Les vues.....	9
Home.....	9
Voitures.....	9
Client	10
Location	10
Statistiques	11
Backend	12
Core.....	12
Modele.....	13
Classes.....	13
Dao	14
Vues	14
Controllers	15
Data.....	15

View	16
<i>Outils utilisés.....</i>	<i>17</i>
Logiciels	17
PhpStorm	17
XAMPP	17
JMerise	17
Langages de programmation	18
HTML.....	18
CSS.....	18
JavaScript	19
PHP.....	19

Consignes

Mission 1: étude et cahier des charges.

Votre première mission sera décomposée en deux parties :

Vous devez tout d'abord analyser ce qui est fourni dans l'archive et proposer des améliorations de la base de données proposée. Pour cela, il est demandé de fournir une analyse textuelle dans un document puis d'utiliser la méthode merise et de fournir un MCD et un MLD de la base de données que vous proposez. Vous pouvez chercher des erreurs sur les clefs primaires, étrangères, s'il manque des entités (ou tables).

Puis vous devrez réaliser le cahier des charges d'un site permettant de gérer des locations de voitures, des clients et de locations. Ce cahier des charges devra présenter une maquette des différents écrans que vous proposez.

Cette première partie est à rendre au plus tard le lundi 16 mai 2022 et fera l'objet d'une première note.

Mission 2: réalisation du site en PHP

Votre deuxième mission consiste à réaliser le site dont vous venez de rédiger le cahier des charges. Ce site devra être réalisé avec PHP. Il doit être fonctionnel mais le look (style) n'a pas besoin d'être « exceptionnel ». En option, si vous avez le temps, je vous propose deux améliorations possibles : Améliorer le look/style de votre application avec du CSS ou un Framework de type Bootstrap.

Améliorer l'expérience utilisateur en intégrant du code javascript qui viendra mettre à jour dynamiquement certaines parties de vos pages. Le JSON pourra par exemple être généré par le PHP. Cette deuxième partie est à rendre au plus tard le dimanche 29 mai 2022 et fera l'objet d'une deuxième note.

3

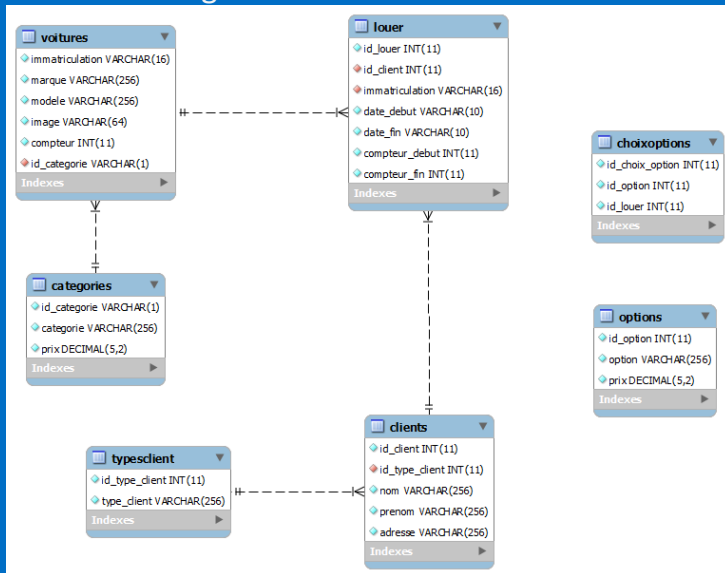


Conception du site

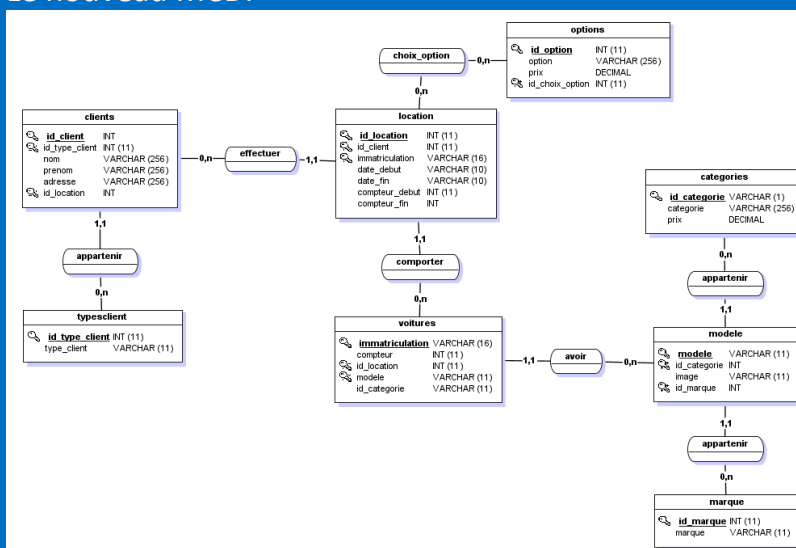
Structure du MCD et du MLD

MCD

Le MCD du stagiaire fournit de base.

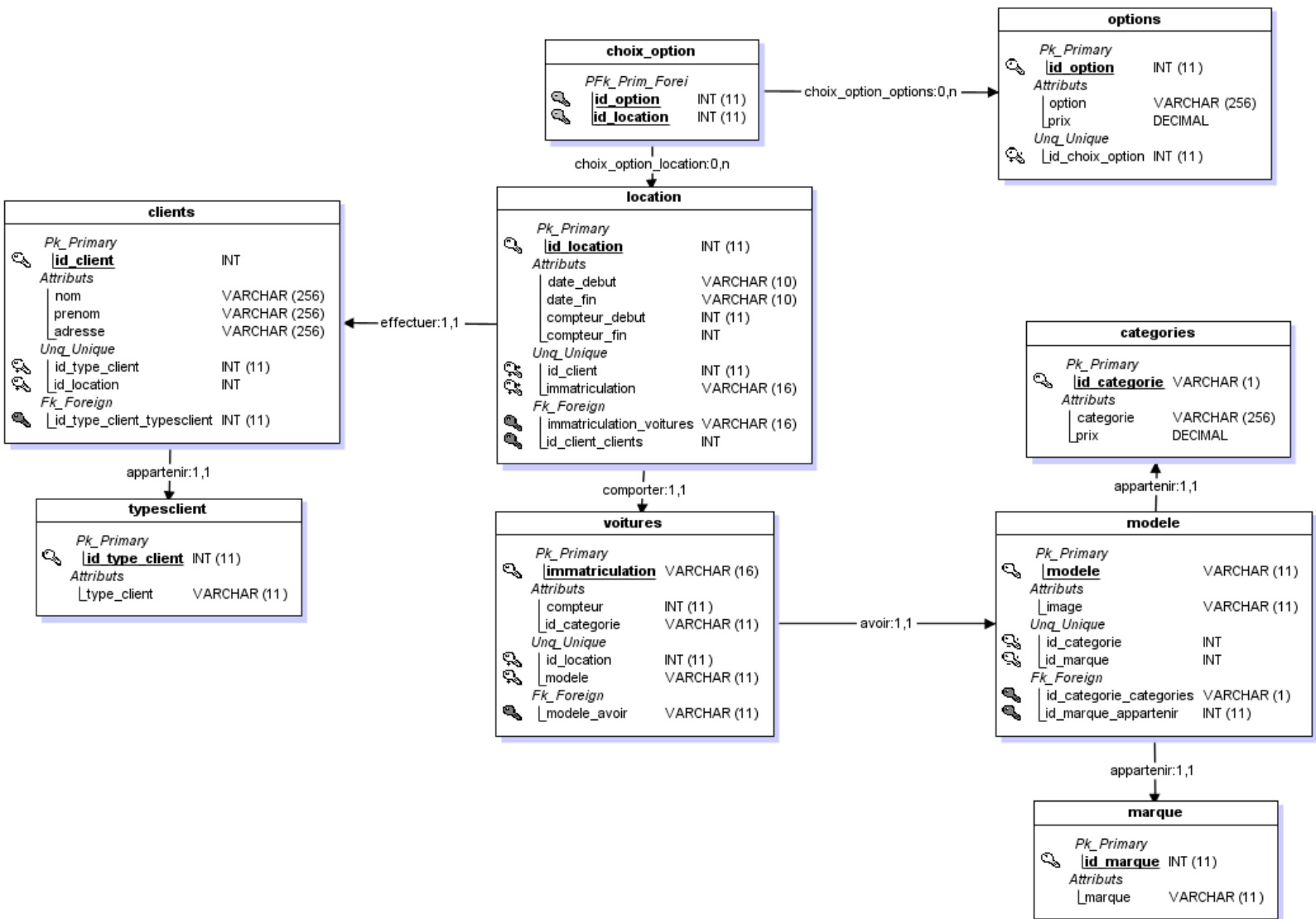


Le nouveau MCD.



MLD

Le MLD généré à partir du MCD



Maquette du site

Première maquette



[Home](#) [Shop](#) [Assistance](#) [Contact](#) [Connexion](#)



MODELE :
CATÉGORIE :
MARQUE :



MODELE :
CATÉGORIE :
MARQUE :



MODELE :
CATÉGORIE :
MARQUE :



MODELE :
CATÉGORIE :



MODELE :
CATÉGORIE :



MODELE :
CATÉGORIE :

[FAQ](#) / [Conseils de sécurité](#) / [Conditions d'utilisation](#) / [Politique relative aux cookies](#) / [Réglages de Confidentialité](#)

© 2022 - LocAuto - Tous droits réservés.

Réalisation et fonctionnement du site

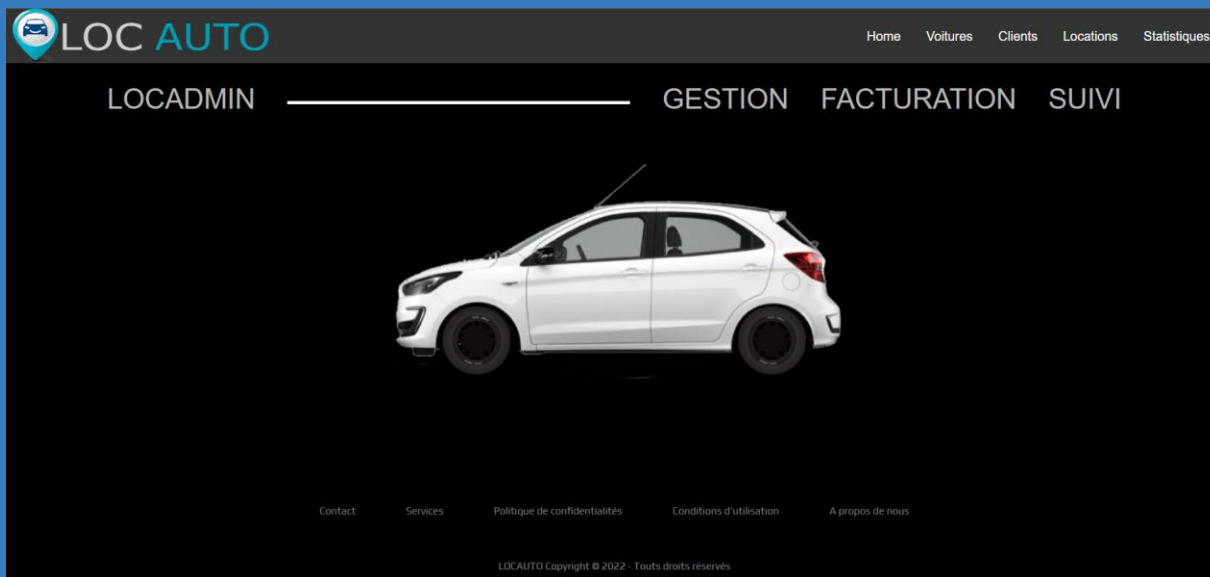


Frontend

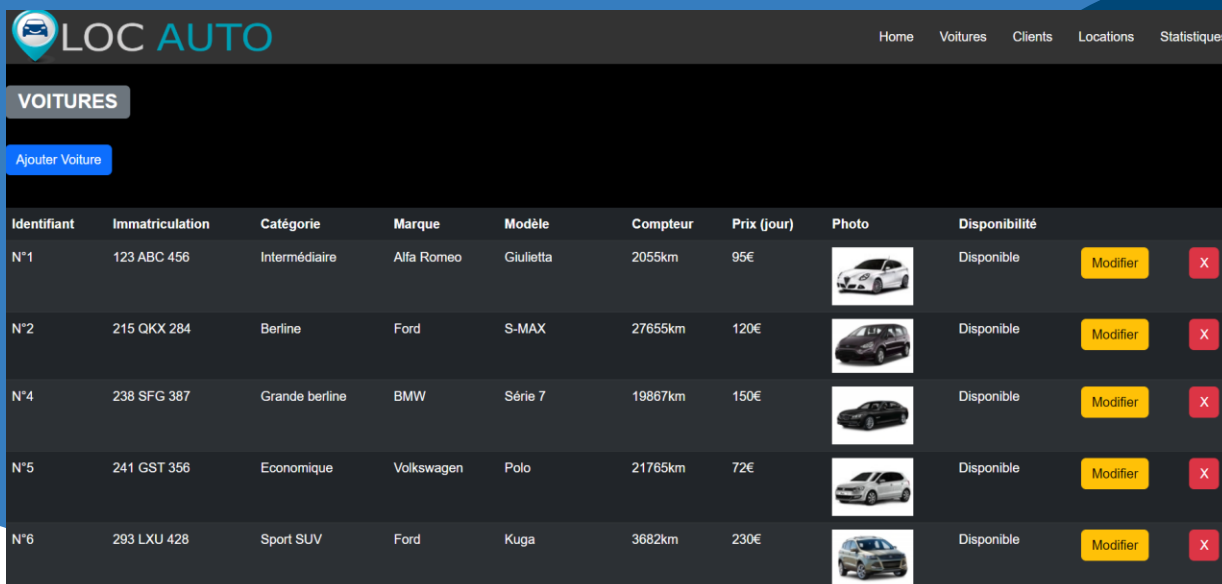
Les vues

Pour le visuel du site, j'ai choisis d'utiliser bootstrap afin d'avoir rapidement un visuel intéressant sans trop de connaissances en css. Je souhaitais également tester un outil que je n'avais pas encore utilisé.


Home



Voitures



Client







HomeVoituresClientsLocationsStatistiques

CLIENTS


Ajouter Client

Identifiant	Nom	Prenom	Adresse	Type de client	Nombre de locations		
N°1	malkovitch	john	paradise street	Particulier	2	Modifier	X
N°2	smith	bill	hell. city	Entreprise	1	Modifier	X
N°3	murray	bill	les fleurs du mal	Administration	0	Modifier	X
N°4	nature	gwendal	rennes	Particulier	1	Modifier	X



ContactServicesPolitique de confidentialitésConditions d'utilisationA propos de nous

Location







HomeVoituresClientsLocationsStatistiques

LOCATIONS

Ajouter Location

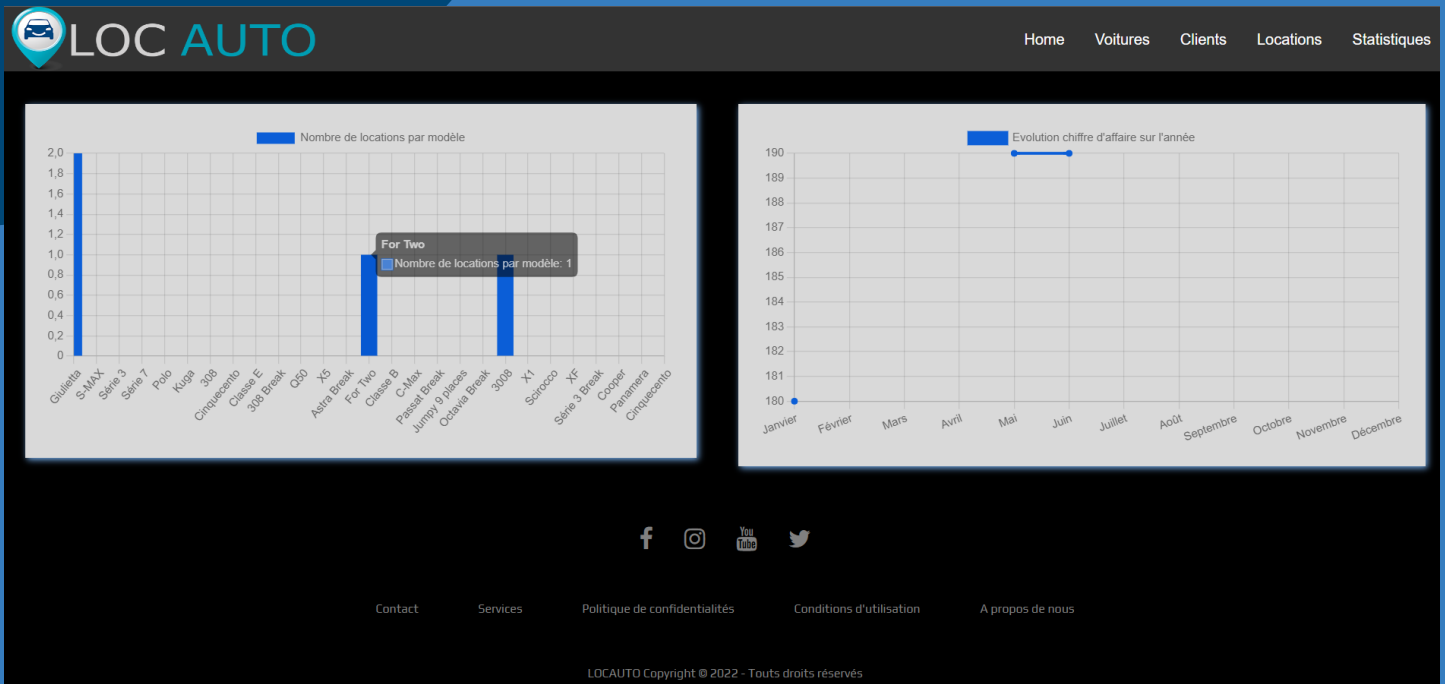
Identifiant Location	Nom	Prénom	Identifiant Client	Modele	Date début location	Date fin location	Durée location	Km parcourus	Prix options	Prix (jour)	Prix total Location	Actuellement en location		
N°1	malkovitch	john	N°1	Giulietta	2022-06-05	2022-06-06	1 jour(s)	1111km	10€	95€	105€	Non	Modifier	X
N°2	malkovitch	john	N°1	Giulietta	1212-12-12	1313-12-13	36890 jour(s)	1km	30€	95€	3504580€	Non	Modifier	X
N°3	nature	gwendal	N°4	3008	2022-06-20	2022-06-21	1 jour(s)	2km	30€	95€	125€	Non	Modifier	X
N°4	smith	bill	N°2	For Two	2022-01-15	2022-01-18	3 jour(s)	198km	250€	60€	430€	Non	Modifier	X



ContactServicesPolitique de confidentialitésConditions d'utilisationA propos de nous

Statistiques

Pour les graphiques j'ai utilisé la librairie javascript chart.js qui se base sur des canvas. J'extraie du json de la base de donnée et je l'envoie dans les graphiques.



Backend

Je ne connaissais ni php, ni l'architecture MVC au début de ce projet, j'ai essayé d'apprendre vite et je n'ai donc pas pu respecter certains principes de l'architecture MVC qui me paraissent aujourd'hui évidents. J'ai recommencé plusieurs fois mon projet avant d'avoir quelque chose qui me satisfasse et même là je vois des améliorations possibles.

La fonction Main se trouve dans l'index. Tout se passe dans l'index, même si l'utilisateur change de vue, il reste en réalité dans l'index et c'est le contrôleur de vues qui lui envoie une vue en fonction des paramètres de l'url. L'index agit donc un peu comme un contrôleur frontal.

```
<?php
// Point d'entrée du site et du programme

class Index {
    public static function main(){
        require_once 'src/core/controller.php';
    }
}

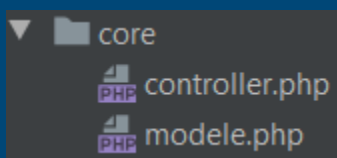
Index::main();
```

Tous les accès au site se font par l'index grâce à un fichier .htaccess qui renvoie une erreur si l'on souhaite accéder à un autre fichier que l'index.

```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteCond %{REQUEST_FILENAME} !-d
RewriteRule ^.*$ index.php [L,QSA]
```

Je n'ai pas encore fait les redirections du router mais de ce que j'ai vu, ça se met en place assez facilement.

Core



Il s'agit du Coeur du programme. C'est ici que l'on traite l'ensemble des requêtes avec la base de données et des interactions de l'utilisateur avec la base données sur le site.

```
<?php
require_once 'src/DbConnexion.php';
require_once 'src/core/modele.php';
require_once 'src/controllers/data.php';
require_once 'src/controllers/view.php';
```

Dans le main controller on initialise la connexion et on importe les controleurs pour traiter les données avec les modèles

Modele

```
<?php

require_once 'src/models/DAO/VoitureDAO.php';
require_once 'src/models/DAO/ModeleDAO.php';
require_once 'src/models/DAO/CategorieDAO.php';
require_once 'src/models/DAO/MarqueDAO.php';
require_once 'src/models/DAO/ClientDAO.php';
require_once 'src/models/DAO/TypeClientDAO.php';
require_once 'src/models/DAO/LocationDAO.php';
require_once 'src/models/DAO/OptionDAO.php';

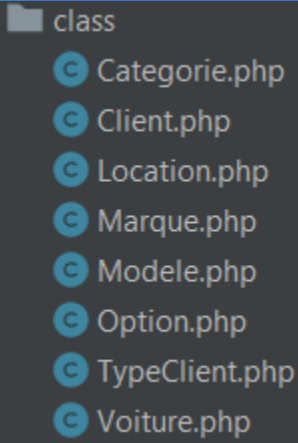
// Voitures
$voitureDAO = VoitureDAO::getInstance();
$modeleDAO = ModeleDAO::getInstance();
$categorieDAO = CategorieDAO::getInstance();
$marqueDAO = MarqueDAO::getInstance();

// Clients
$clientDAO = ClientDAO::getInstance();
$typeClientDAO = TypeClientDAO::getInstance();

// Locations
$locationDAO = LocationDAO::getInstance();
$optionDAO = OptionDAO::getInstance();
```

Classes

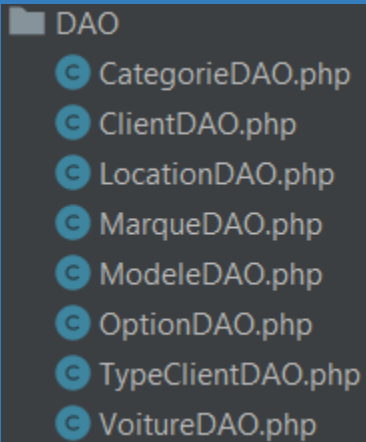
Il s'agit des classes métiers qui représentent chacune une table de la base de données et qui vont avoir les données structures de la database stockées dans les variables de l'objet tout en respectant le schema de la base donnée fourni.



- class
 - Categorie.php
 - Client.php
 - Location.php
 - Marque.php
 - Modele.php
 - Option.php
 - TypeClient.php
 - Voiture.php

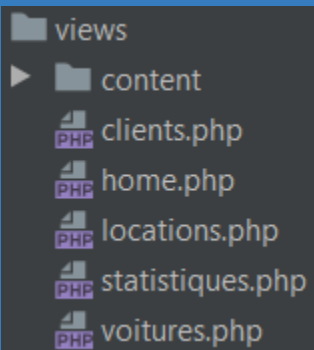
Dao

Les DAO effectuent les requêtes SQL et en génère des objets avec les classes métiers



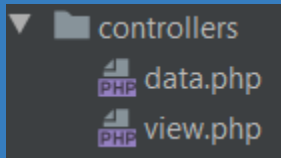
- DAO
 - CategorieDAO.php
 - ClientDAO.php
 - LocationDAO.php
 - MarqueDAO.php
 - ModeleDAO.php
 - OptionDAO.php
 - TypeClientDAO.php
 - VoitureDAO.php

Vues



- views
 - content
 - clients.php
 - home.php
 - locations.php
 - statistiques.php
 - voitures.php

Controllers



Data

Ici on gère les interactions de l'utilisateur avec la base de données.

```
<?php

if (isset($_POST['addClient'])) {
    $clientDAO->addClient($_POST['nom'], $_POST['prenom'], $_POST['adresse'], $_POST['typeClient']);
}

if (isset($_POST['editClient'])) {
    $clientDAO->editClient($_POST['nom'], $_POST['prenom'], $_POST['adresse'], $_POST['typeClient']);
}

if (isset($_POST['deleteClient'])) {
    $clientDAO->deleteClient($_POST['idClient']);
}

if (isset($_POST['addCar'])) {
    $voitureDAO->addVoiture($_POST['modele'], $_POST['immatriculation'], $_POST['compteur']);
}

if (isset($_POST['editCar'])) {
    $voitureDAO->addVoiture($_POST['idCar'], $_POST['immatriculation'], $_POST['compteur'], $_POST['modele']);
}

if (isset($_POST['deleteCar'])) {
    $voitureDAO->deleteVoiture($_POST['idCar']);
}
```

View

Le controller de vue va chercher la vue à afficher en fonction des paramètres d'URL.

```
<?php
require_once 'src/views/content/style.php';
require_once 'src/views/content/header.php';

// Switch de la vue principale

if (isset($_GET['choice'])) {
    $view = $_GET['choice'];
    switch ($view) {
        case "voitures":
            require_once('src/views/voitures.php');
            break;
        case "clients":
            require_once('src/views/clients.php');
            break;
        case "locations":
            require_once('src/views/locations.php');
            break;
        case "statistiques":
            require_once('src/views/statistiques.php');
            break;
        case "home":
            require_once('src/views/home.php');
            break;
    }
} else {
    require_once('src/views/home.php');
}

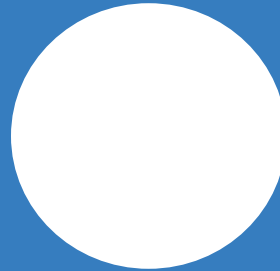
require_once 'src/views/content/footer.php';
```


Outils utilisés

Logiciels

PhpStorm

PhpStorm est un éditeur de code conçu pour le web permettant d'écrire du PHP, HTML, CSS et JavaScript. Ce logiciel est édité par JetBrains.



XAMPP

XAMPP est un ensemble de logiciels permettant de mettre en place un serveur Web local, un serveur FTP et un serveur de messagerie électronique. Il s'agit d'une distribution de logiciels libres offrant une bonne souplesse d'utilisation, réputée pour son installation simple et rapide.



JMerise

Jmerise est un logiciel open source de conception de MCD et MLD.

Langages de programmation

HTML

HyperText Markup Language, généralement abrégé HTML est le langage de balisage conçu pour représenter les pages web. Ce langage permet : d'écrire de l'hypertexte, d'où son nom, de structurer sémantiquement la page, de mettre en forme le contenu.



CSS

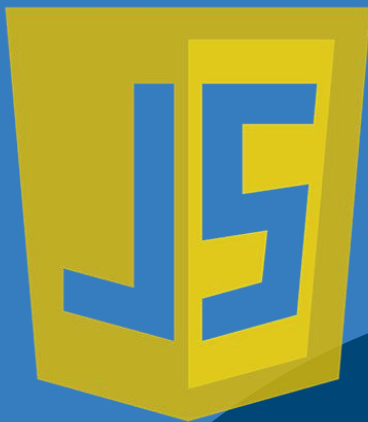
Les feuilles de style en cascade, généralement appelées CSS de l'anglais Cascading Style Sheets, forment un langage informatique qui décrit la présentation des documents HTML et XML. Les standards définissant CSS sont publiés par le World Wide Web Consortium.



JavaScript

JavaScript est un langage de programmation de scripts principalement employé dans les pages web interactives et à ce titre est une partie essentielle des applications web. Avec les langages HTML et CSS, JavaScript est au cœur des langages utilisés par les développeurs web.

JS



PHP

Hypertext Preprocessor, plus connu sous son sigle PHP, est un langage de programmation libre principalement utilisé pour produire des pages Web dynamiques via un serveur HTTP, mais pouvant également fonctionner comme n'importe quel langage interprété de façon locale. PHP est un langage impératif orienté objet.



php

