Projet 6 : Petites annonces PHP MVC

Ressources :

https://github.com/michelonlineformapro/Projet-6B-Annonces-MVC-Remaster

Description du projet :

Projet individuel ou en groupe - Temps estimés : 15 jours
Technologies, outils et concept travaillés : HTML / CSS, PHP, POO, Design Pattern MVC, Routeur et/ou Alto Router, Réécritures d'url, Serveur web local, IDE, PHP MyAdmin, MySQL, Git, Composer, CSS/SCSS

Vous allez devoir créer un site ou application web de qui permet à un visiteur de consulté et éventuellement acheté (pas obligatoire) des produits grâce des petites annonces.

La structure du projet doit respecter les règles du Design Pattern « Modèles – Vues – Contrôleurs » et de la programmation orientée objet PHP.

Modèle : cette partie gère les *données* de votre site. Son rôle est d'aller récupérer les informations « brutes » dans la base de données, de les organiser et de les assembler pour qu'elles puissent ensuite être traitées par le contrôleur. On y trouve donc entre autres les requêtes SQL.

Vue : cette partie se concentre sur l'*affichage*. Elle ne fait presque aucun calcul et se contente de récupérer des variables pour savoir ce qu'elle doit afficher. On y trouve essentiellement du code HTML mais aussi quelques boucles et conditions PHP très simples, pour afficher par exemple une liste de messages.

Contrôleur : cette partie gère la logique du code qui prend des *décisions*. C'est en quelque sorte l'intermédiaire entre le modèle et la vue : le contrôleur va demander au modèle les données, les analyser, prendre des décisions et renvoyer le texte à afficher à la vue. Le contrôleur contient exclusivement du PHP. C'est notamment lui qui détermine si le visiteur a le droit de voir la page ou non (gestion des droits d'accès).

Coté visiteur :

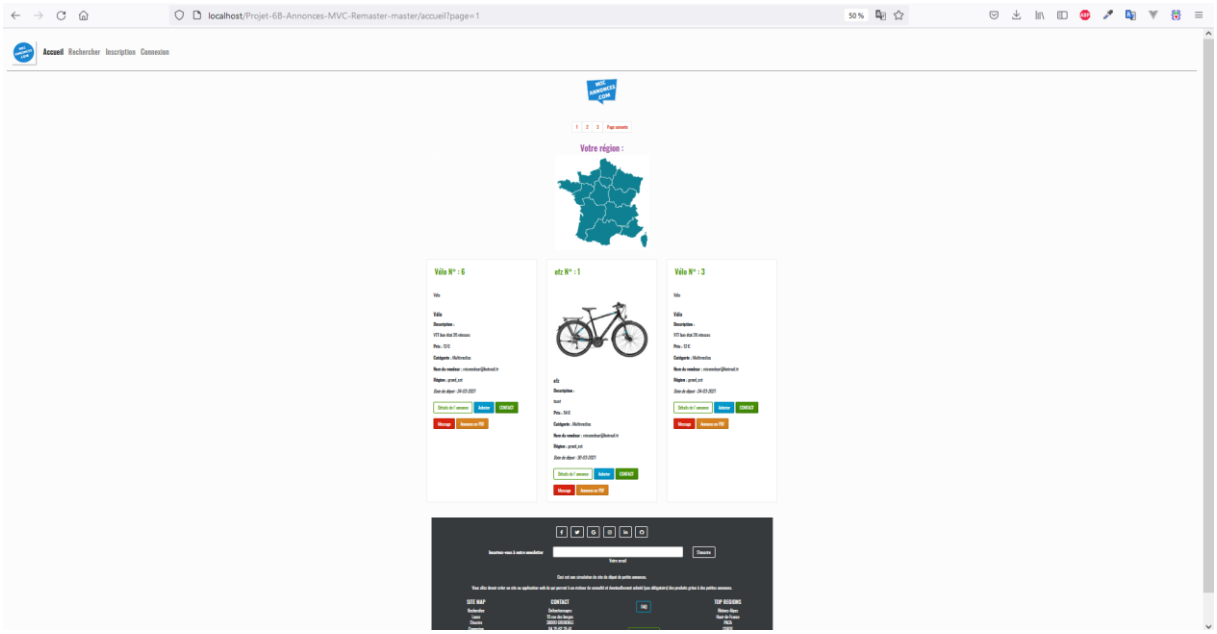
Le visiteur à la possibilité de rechercher une/des annonces par catégorie et/ou par région ou par des mots clés, à la suite de sa recherche, il est redirigé vers la page concernée ou les résultats seront affiché dans la page d'accueil.

Pour la recherche par région utilisé la carte de France interactive :

https://cmap.comersis.com/carte-France-interactive-HTML5-gratuite-cm6z59f89o4.html

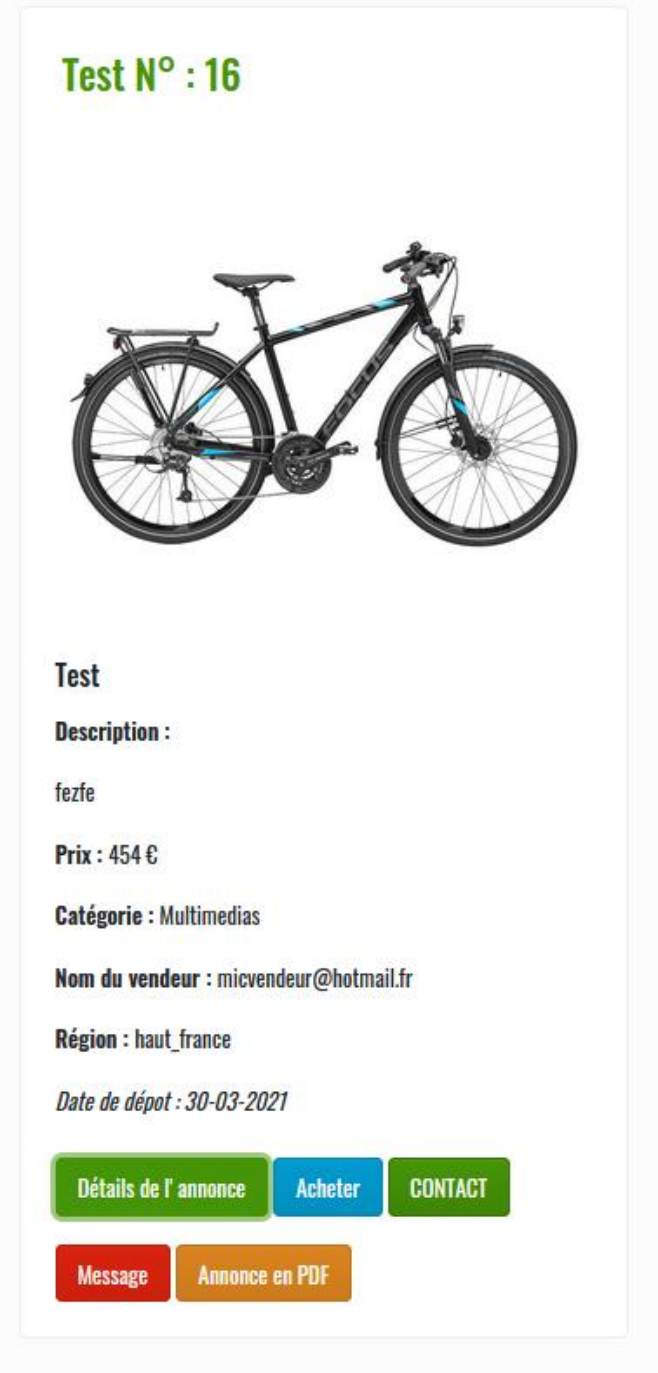
A vous de modifier le CSS (style.css) et le JavaScript (France-map.js) pour assigner à chaque région un id dynamique.

Une seconde page de recherche propose une barre avec des inputs et une recherche par catégorie et région (les 2 critères sont indépendant)



Les annonces sont composées de :

- Un titre
- Une description
- Une image (ou plusieurs)
- Un prix
- La catégorie
- La région
- L’email du vendeur
- La date de dépôt
- Un bouton détails de l’annonces
- Un bouton acheter
- Un bouton envoyer un email au vendeur
- Un bouton contact qui affiche le n° de téléphone
- Un bouton exporter l’annonce au format PDF



Lors d'un clic sur une annonces le visiteur à la possibilité :

- Acheter le produit (API de paiement pas obligatoire)
- Consulter le numéro de téléphone
- Envoyer un email au vendeur
- Télécharger l'annonces au format PDF

Coté utilisateur : (demande un système d'inscription et de connexion)

Lorsqu'une personne souhaite poster une annonce, il doit s'inscrire, l'utilisateur accède à un formulaire d'inscription, un email de validation lui est envoyé (PHP Mailer + MailTrap) pour valider cette dernière.

Dans l'email un bouton permet de valider l'inscription et le redirige vers une page récapitulative de ses informations client et l'invite à se connecter à son tableau de bord.

Le tableau de bord est un CRUD (Create, Read, Update, Delete) des petites annonces enregistrées par id utilisateur et sa région, lors de la création tous les champs sont obligatoires.

Attention l'id de l'utilisateur doit correspondre à une région (il est également possible qu'un utilisateur est plusieurs région (ex : s'il possède une résidence secondaire))

Les pages d'annonces doivent lister les 10 premières annonces du site et permettre d'afficher les suivantes 10 par 10. (Pagination)

Coté Administration : (demande un système d'inscription et de connexion)

Lors de la connexion en tant qu'administrateur, le tableau de bord permet d'accéder a toutes les données des utilisateurs, soit :

- Liste des administrateurs + CRUD
- Liste des utilisateurs + CRUD
- Liste des annonces + CRUD
- Liste des catégories + CRUD
- Liste des régions + CRUD

L'administrateur peu :

- Ajouter supprimer des administrateurs (éventuellement les édités)
- Supprimer des utilisateurs
- Supprimer des articles non désirés
- Ajouter et/ou supprimer des catégories
- Les régions n'ont pas de CRUD et sont aux nombre de 13, ce sont des constantes elles ne sont pas vouées a changé

Structure du projet :

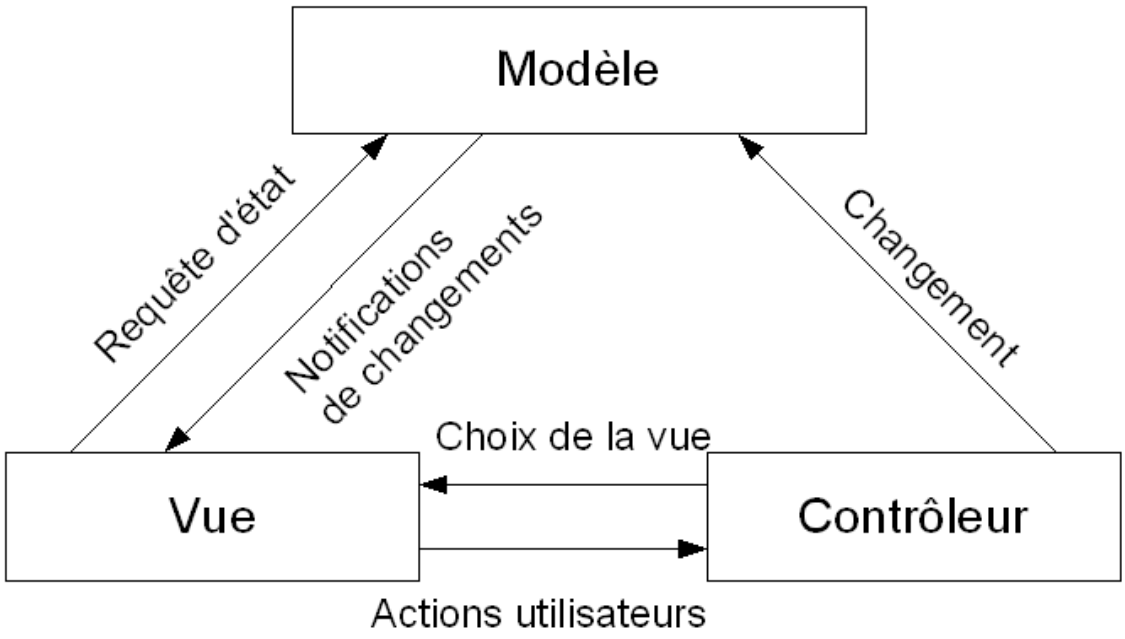
Le design pattern MVC (Model Views Controllers)

Le design pattern Modèle-Vue-Contrôleur (MVC) est un pattern architectural qui sépare les données (le modèle), l'interface homme-machine (la vue) et la logique de contrôle (le contrôleur).

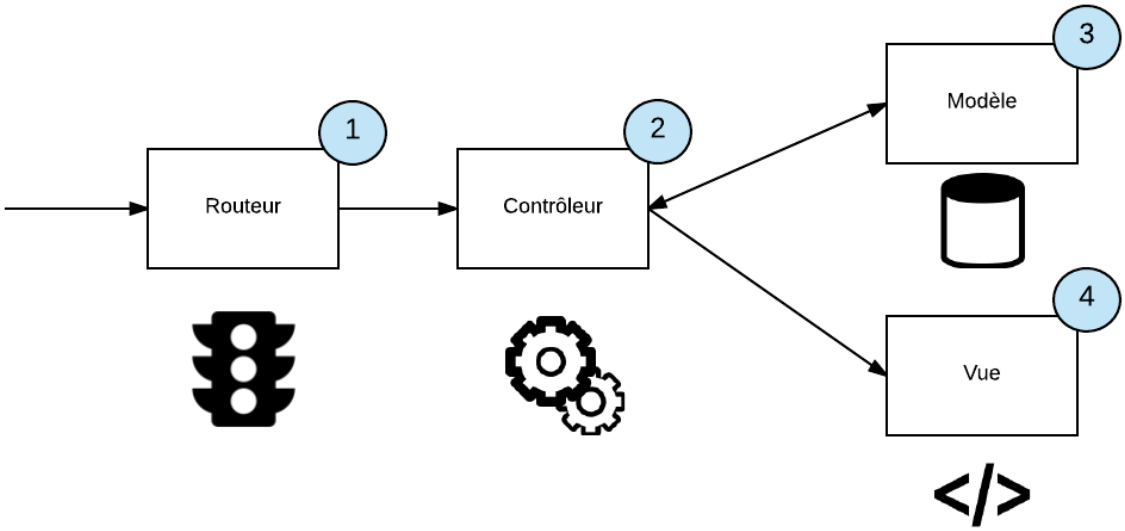
Ce modèle de conception impose donc une séparation en trois couches :

- Le modèle : il représente les données de l'application. Il définit aussi l'interaction avec la base de données et le traitement de ces données ;
- La vue : elle représente l'interface utilisateur, ce avec quoi il interagit. Elle n'effectue aucun traitement, elle se contente d'afficher les données que lui fournit le modèle. Il peut tout à fait y avoir plusieurs vues qui présentent les données d'un même modèle ;
- Le contrôleur : il gère l'interface entre le modèle et le client. Il va interpréter la requête de ce dernier pour lui envoyer la vue correspondante. Il effectue la synchronisation entre le modèle et les vues.

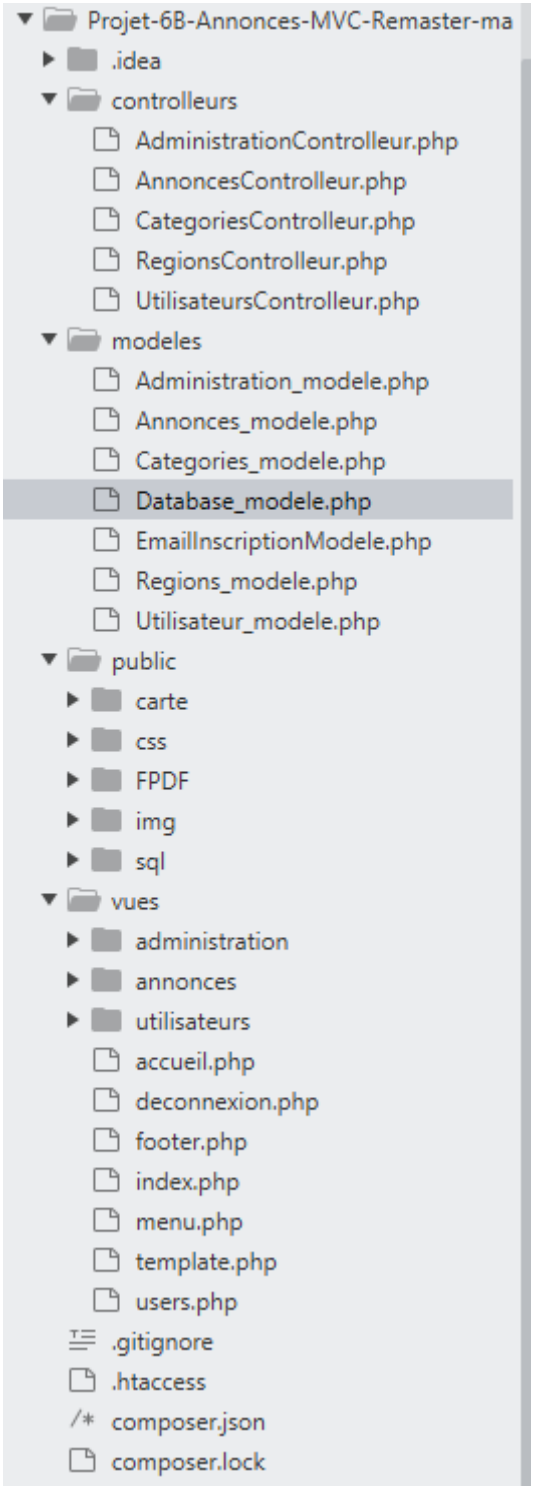
La synchronisation entre la vue et le modèle se passe avec le pattern Observer. Il permet de générer des événements lors d'une modification du modèle et d'indiquer à la vue qu'il faut se mettre à jour.



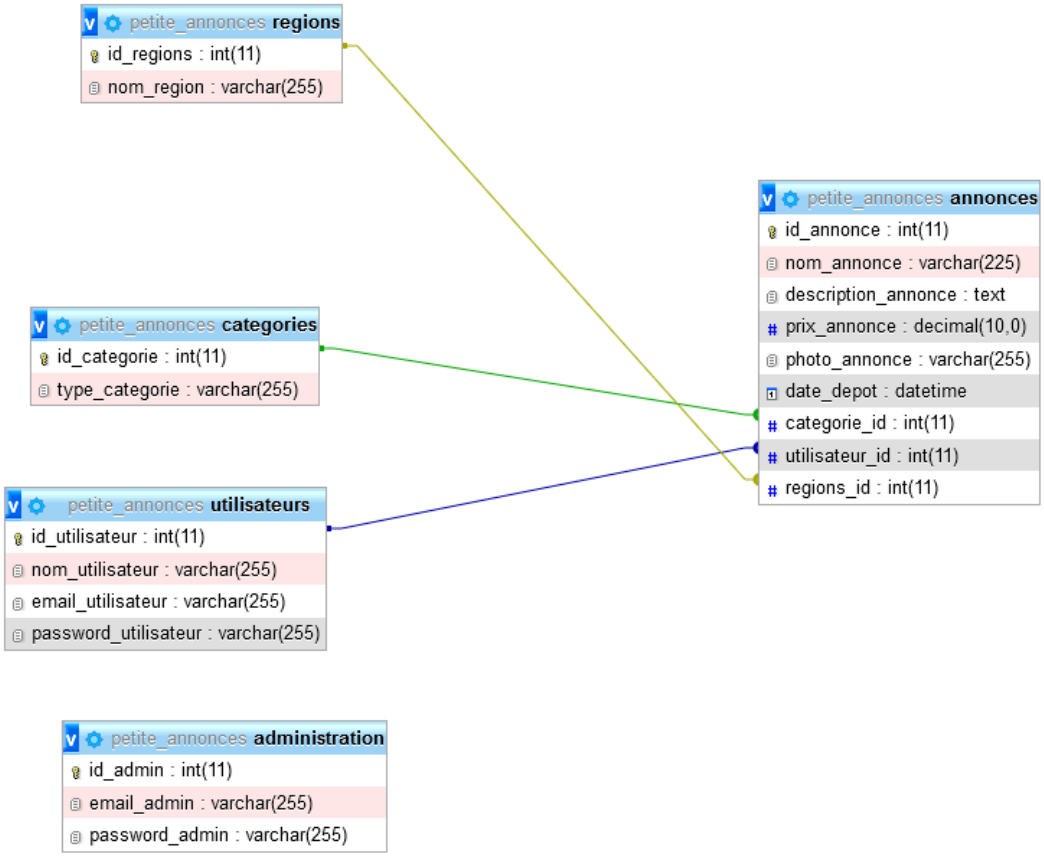
Architecture MVC avec un routeur :



Exemple d'architecture des dossiers :


Projet-6B-Annonces-MVC-Remaster-ma
.idea
controlleurs
AdministrationControlleur.php
AnnoncesControlleur.php
CategoriesControlleur.php
RegionsControlleur.php
UtilisateursControlleur.php
modeles
Administration_modele.php
Annonces_modele.php
Categories_modele.php
Database_modele.php
EmailInscriptionModele.php
Regions_modele.php
Utilisateur_modele.php
public
carte
css
FPDF
img
sql
vues
administration
annonces
utilisateurs
accueil.php
deconnexion.php
footer.php
index.php
menu.php
template.php
users.php
.gitignore
.htaccess
composer.json
composer.lock

Concevoir au préalable un MCD (Modèle conceptuelle de données) solide :



Créer un routeur qui oriente le ou les contrôleurs :

Extrait du fichier .htaccess qui configure Apache :

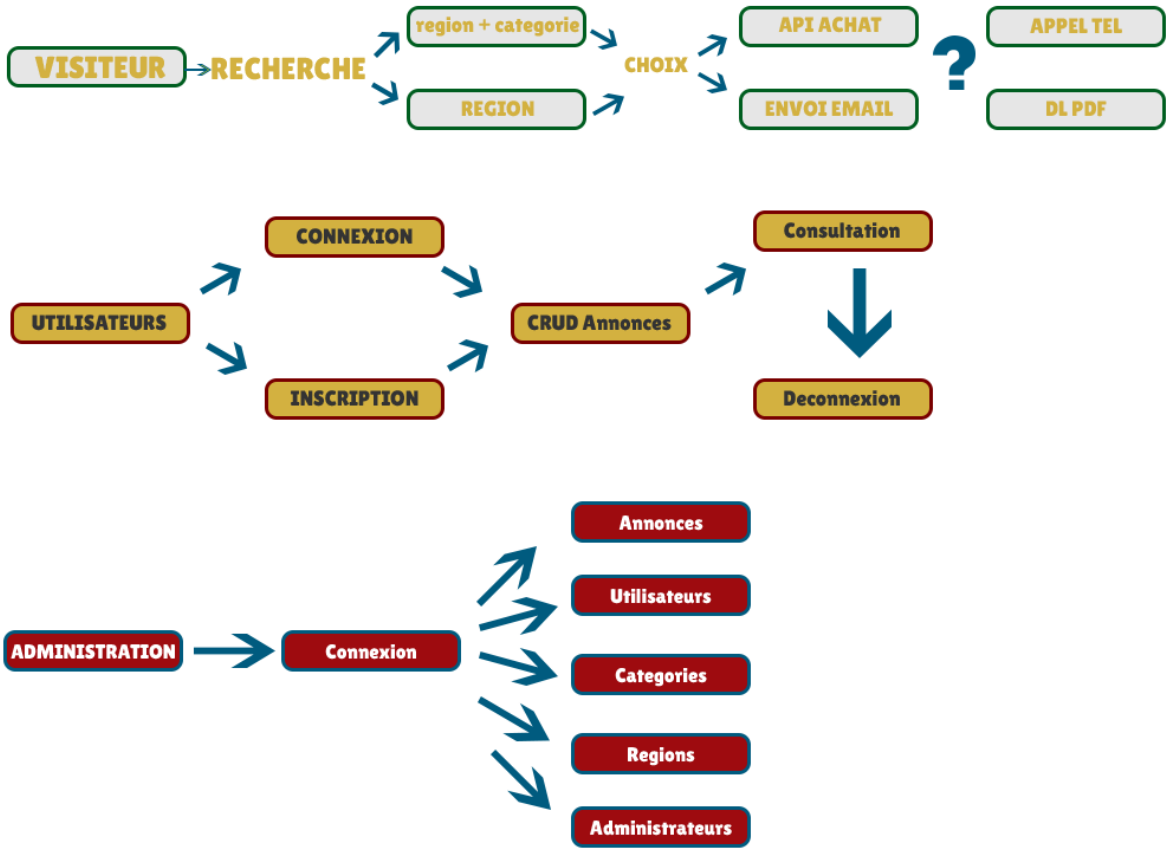
```
RewriteEngine on
RewriteCond %{REQUEST_FILENAME} !-f
RewriteRule ^(.*)$ views/index.php?url=$1 [QSA,L]
```

Ici le point d'entré index.php se situe dans le dossier views et focus sur le routing :

Views/index.php ?url=routeur

Pour anticiper l'expérience utilisateur :

Diagramme de séquences des différents acteurs :



Les +

1. Randomiser les annonces avec un bouton.
2. Ajouter les dates de dépôt.
3. Un système de commentaire sur les vendeurs et fiabilité (système de note sur 5)
4. Interdire des dépôt d'annonces avec des mots clés sensible (homophobie, haine, sexisme, etc…).
5. Système de réservation et click & collecte et – ou de livraison ?
6. Système de rappel de mot de passe oublier.
7. Tâche cron qui supprime les annonces qui sont publiées à n+15 jours de la date de création.
8. Envoyer un mail à la personne de la suppression de son annonce
9. Créer un infinite-scroll pour la pagination des annonces
10. Utilisé cURL et Postman pour consommer des API 's pour lister les communes de France et améliorer le formulaire de recherche
11. Créer des requête Ajax pour des résultats asynchrones des demandes.

Les étapes :

- Créer le MCD ensemble et confirmer votre MCD par votre coach
- Créer le MLD ensemble et confirmer votre MLD par votre coach
- Créer le modèle Physique de données ensemble et confirmer votre Modèle Physique par votre coach
- Installer Composer sous Windows ensemble
- Installer vos dépendances PHP : (au choix) Alto Router, MJML (outils de création email responsive avec une syntaxe alternative à HTML 5), MJML-PHP, FPDF /mPDF ensemble
- Installer nodejs et npm ou yarn ensemble si besoin API
- Installer vos dépendances front : sass (SCSS) si besoin
- Définir l'arborescence de vos dossiers ensemble et confirmer par votre coach
- Mettre en place le .htaccess pour Apache ensemble et confirmer votre coach
- Avec Composer généré l'Autoload (PSR-4) ensemble
- Définir les routes nécessaires à votre projet
- Mettre en place le Router et ajouter vos routes
- Créer les contrôleurs et méthodes nécessaire à vos routes ensemble et confirmer par le coach
- Pour la partie back :
    - Créer les modèles
    - Pour chaque méthode définir les variables nécessaires à la vue
- Pour la partie front
    - Réaliser les wireframe de toutes vos pages et emails
    - Réaliser les vues nécessaires pour vos différentes pages
    - Réaliser le MJML de vos mails si besoin
    - Réaliser le code pour générer le PDF (FPDF et/ou mPDF)
    - Réaliser le CSS et responsive

Durant toutes les étapes Essayer par vous-même

- Fait un point régulier avec le coach et/ou les autres apprenants participant au projet
- Faite un Trello de vos différentes tâches
- Vous êtes en difficultés ? demander à un apprenant de votre groupe de vous expliquez les bases
- Tu as compris ? prend le temps d'aider un autre apprenant
- Tu as fini ? ajoute des fonctionnalités
- Tous le long du projet commit et push ton projet

Ressources :

- Installer composer sous windows
- composer guide : Ajouter une dépendance
- composer Autoload PSR-4
- Exemple d'une structure des dossiers en mvc
- .htaccess exemple
- SASS
- Alto router
- Alto router exemple mapping
- MJML
- MJML exemple de base
- MJML-PHP
- MPDF & Fpdf
