Projet 6 : Petites annonces PHP MVC

Ressources :

https://github.com/michelonlineformapro/Projet7_Poo_Mvc_Annonces

Description du projet :

Projet individuel ou en groupe - Temps estimés : 10-15 jours
Technologies, outils et concept travaillés : HTML5, CSS3/SCSS, PHP, POO, Design Pattern MVC, Routeur et/ou Alto Router, Réécritures d'url, WAMP, IDE, PhpMyAdmin, MySQL, Git, Composer

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.

Un visiteur à la possibilité de s'inscrire, ce connecter et manager ses annonces (CRUD)

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(logique metier). 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 et des algorithmes complexes .

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 :

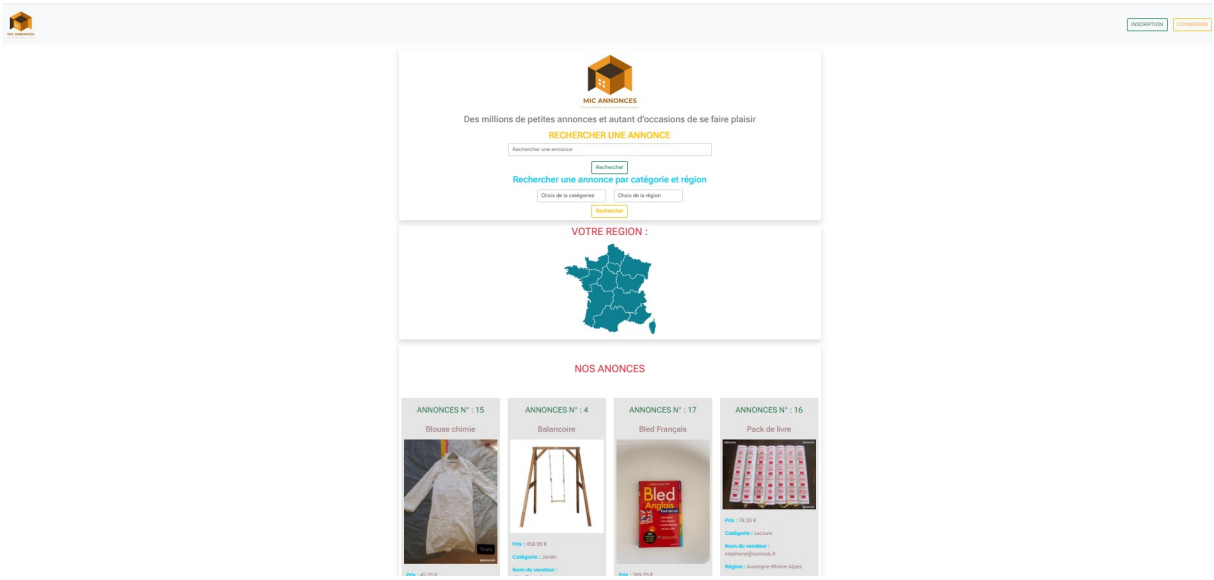
Le visiteur à la possibilité de consulter , rechercher une/des annonces par catégorie et/ou par région ou par des mots clés, à la suite de sa recherche, il est redirigé vers la page concernée ou les résultats seront affiché dans la page d'accueil.

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

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

Libre a vous de modifier le CSS (style.css) et le JavaScript (France-map.js) pour assigner à chaque zone de la carte un id dynamique.

Maquette :



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)
- 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 de connexion 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 + :

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

Tester des packs disponible sur le site :

https://packagist.org/?query=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 :

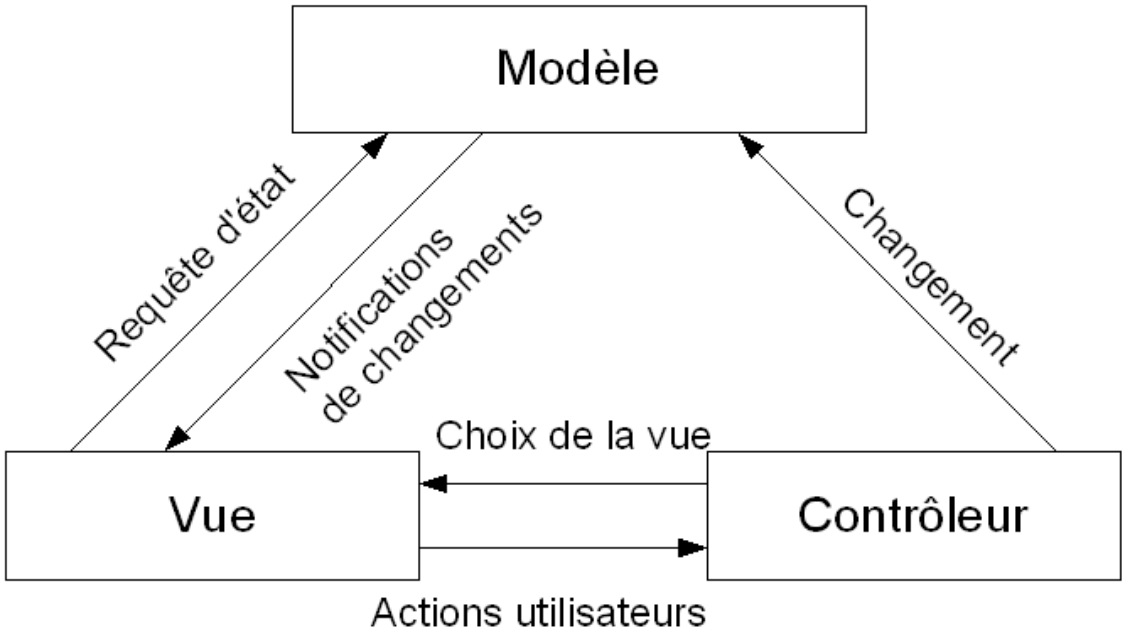
Le design pattern MVC (Model Views Controllers)

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).

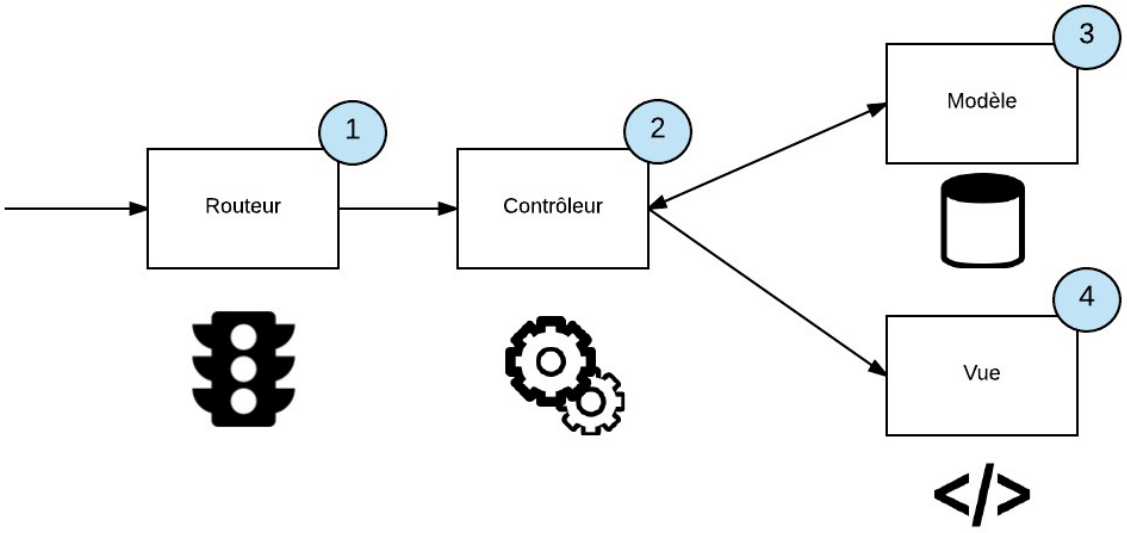
Ce modèle de conception impose donc une séparation en trois couches :

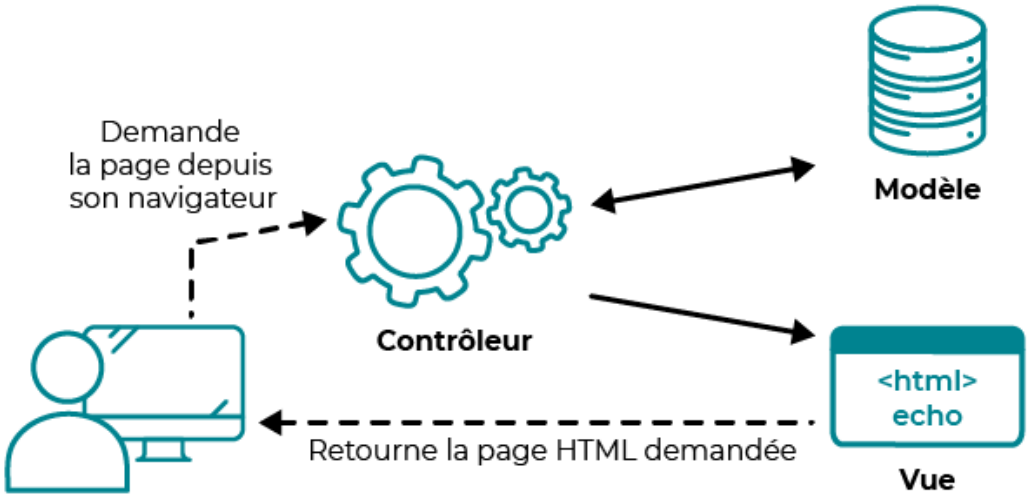
- Le modèle : il représente les données de l'application. Il définit aussi l'interaction avec la base de données et le traitement de ces données ;
- La vue : elle représente l'interface utilisateur, ce avec quoi il interagit. Elle n'effectue aucun traitement, elle se contente d'afficher les données que 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.

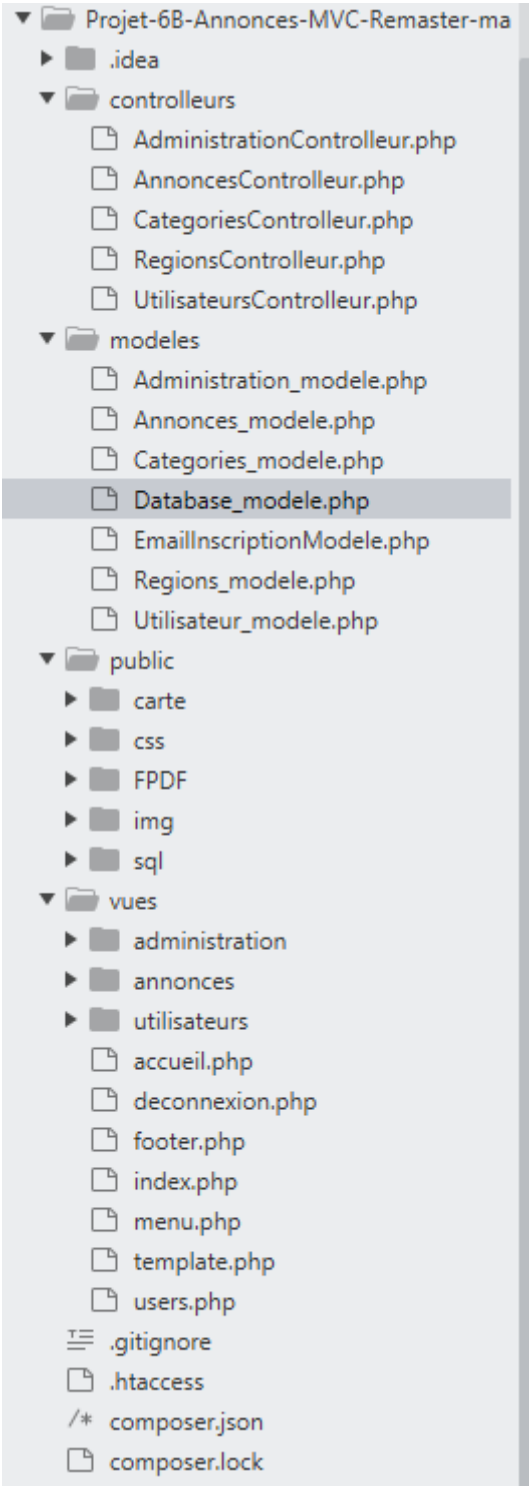


Architecture MVC avec un routeur :





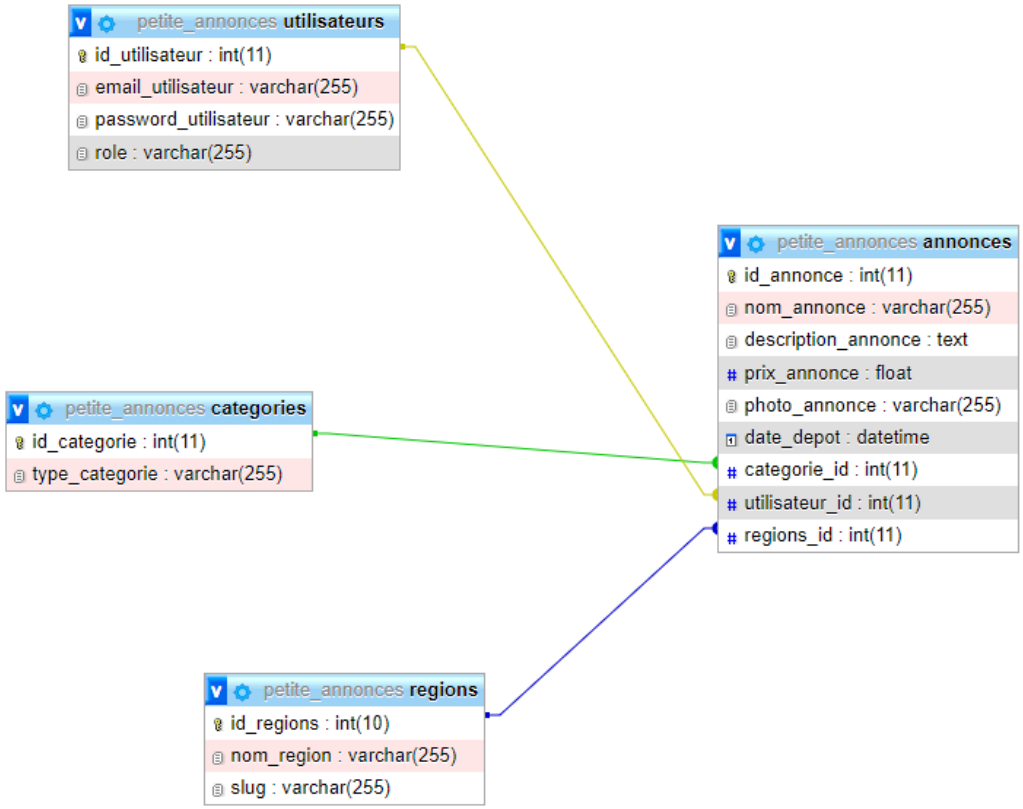
Exemple d'architecture des dossiers :


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

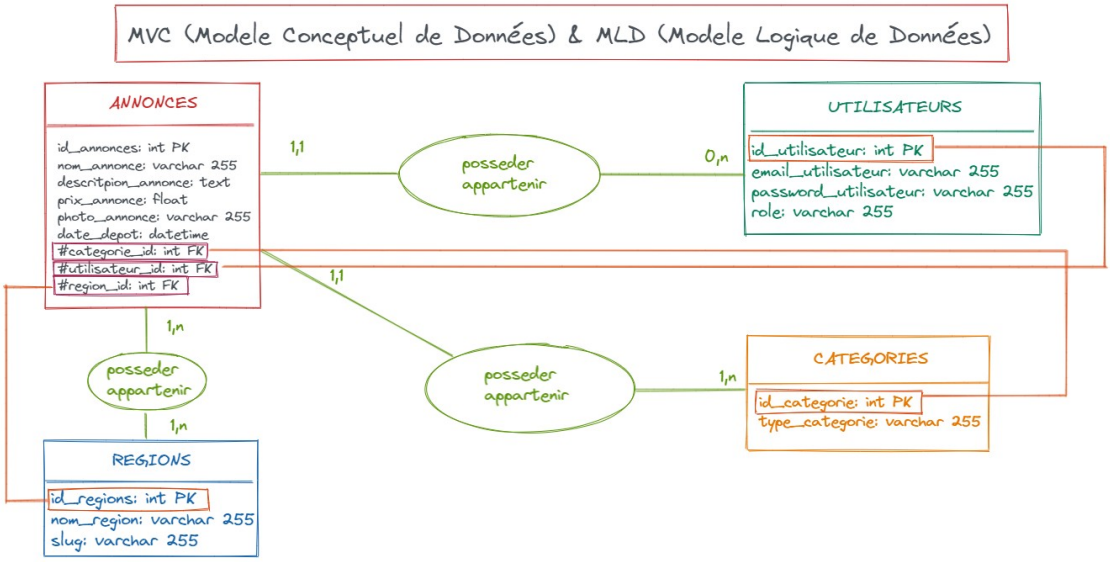
1)Élaborer un dictionnaire de données pour chaque table

2)Élaborer une liste des taches (Products Back Logs) a l'aide Github/projects

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



LE MLD (Modèle Logique de Données)



3)Créer un fichier de configuration pour le serveur Apache :

Extrait du fichier .htaccess qui configure Apache :

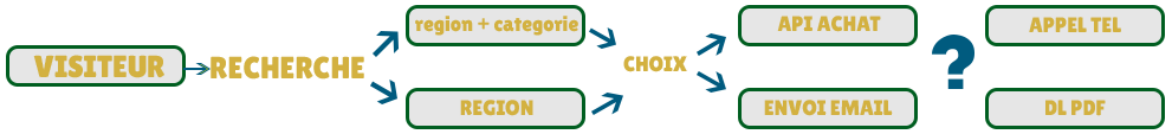
```
RewriteEngine on

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

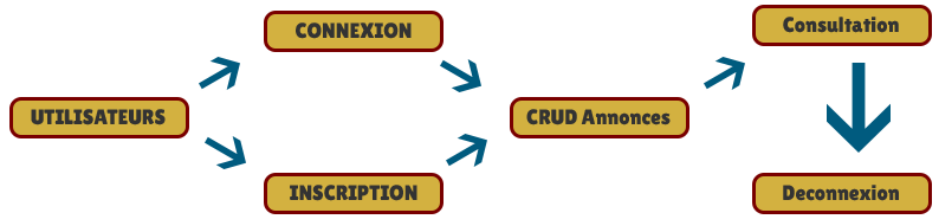
Ici le point d'entré index.php se situe dans le dossier vues

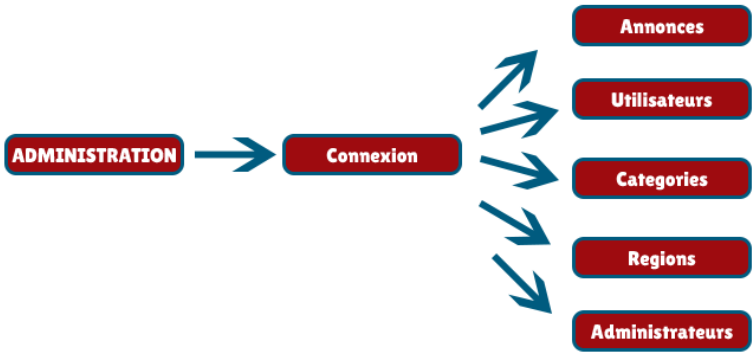
Chaque query_string vues/index.php?url=$_GET[‘ma_route’] sera remplacée par votre roue.

Pour anticiper l'expérience utilisateur :

Diagramme de séquences des différents acteurs :







Les +

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

ÉTAPES 1 : LES BASES DU MVC

1. Créer un fichier de routing index.php dans le dossier vues/
2. Démarrer une session PHP : avec session_start()
3. Démarrer la mise en cache avec ob_start()
4. Créer votre query_string avec la super globale $_GET[‘url’]

```php
//Si dans url, un paramètre url existe
if(isset($_GET['url'])){
    $url = $_GET['url'];
}else{
    $url = 'accueil';
}
//Si index.php?url= null
if($_GET['url'] === ""){
    $url = 'accueil';
}
```

5.

6. Créer une route pour la splash page (page d’accueil ou entrée d’un site web composée d’une animation ou vidéo)
7. Cette route appel une fonction du contrôleur AnnoncesControleur

```php
//******************PAGE SPLASH *********************//

if($url === "mic-location"){
    $title = "-Mic Annonces-";
    afficherSplashPage();
}
```

8.
8.
8.
8.
8.
8.

9. Ajouter en bas de page les regexs et la lecture du cache stocker dans une variable

```php
//Si la route $url n'est pas formée de [#: A-Z a-z 0-9] soit index.php?url=NON VALIDE
//On effectue une redirection
elseif($url !=  '#:@&-[\w]+)#'){
    //On redirige vers la page d'accueil
    header( string: "Location: accueil");
}

/*
 * ob_get_clean – Lit le contenu courant du tampon (du cache)de sortie puis l'efface
 */
//ici $content se situe dans le dossier template.php
$content = ob_get_clean();
require_once "template.php";
```

10. Créer une fichier template.php avec le squelette HTML5, appel de libraries CSS et votre variable de cache echo $content dans le <body>
11. Créer un dossier modèles et une classe annonces.php
12. Créer un dossier contrôleurs et un fichier AnnoncesControleur.php
13. Dans votre fichier modeles/Annonces.php
14. Créer votre classe annonce + une fonction donneesModeleAnnonce() qui retourne un titre HTML

```php
<?php

require_once "../modeles/Database.php";
class Annonces extends Database
{
    ////////////////////////PROPRIETEES PRIVEES//////////////////////////////
    //Intitulé des colones de la tables annonces php myd Admin
    private $id_annonce;
    private $nom_annonce;
    private $description_annonce;
    private $prix_annonce;
    private $date_depot;
    private $photo_annonce;
    private $categorie_id;
    private $utilisateur_id;
    private $region_id;

    ////////////////////////////////////SPlASH PAGE//////////////////////////////////
    public function donneesModeleAnnonce(){
        ?>
            <h1 class="text-center text-secondary">MIC ANNONCES</h1>
        <?php
    }
```

15. Dans votre dossier controleurs/AnnoncesControleur.php
16. Appeler le fichier de modeles Annonce.php
17. Créer une fonction afficherSplashPage() appelée dans le router
18. Créer une variable qui stock une instance de la classe modeles/Annones.php
19. Avec cette variable appelé la méthode donneesModeleAnnonces() ;
20. Appeler la vues/splash.php

21. Retourner la variable $annonces

```php
<?php
require_once "../modeles/Annonces.php";

//////////////////LA SPLSH PAGE//////////////
function afficherSplashPage(){
    $annonceModele = new Annonces();
    $annonce = $annonceModele->donneesModeleAnnonce();
    require_once "../vues/splash.php";
    return $annonce;
}
```

22.
22.
22.
22.
22.
22.
22.
22.
22.
22.

Dans le router index.php => appel le fichier controleurs/AnnoncesControleur.php

```php
ob_start();
//Auto loader de class
require "../vendor/autoload.php";
//Appel des controleurs
require_once "../controleurs/UtilisateursControleur.php";
require_once "../controleurs/AdministrationControleur.php";
require_once "../controleurs/AnnoncesControlleur.php";
require_once "../controleurs/RegionsControlleur.php";
require_once "../controleurs/CategoriesControlleur.php";
```

23. L'appel du contrôleur donne accès a la fonction afficherSplashPage() qui appel la vue et donnes accès aux données du modèle Annonces.php

ÉTAPES 2 : INSCRIRE DES

UTILISATEUR AVEC VALIDATION PAR E-MAIL

LA ROUTE :

```php
//******************INSCRIPTION UTILISATEUR*********************//
elseif ($url === "inscription-utilisateur"){
    $title = "INSCRIPTION -Mic Annonces-";
    require_once "utilisateurs/inscription_utilisateur.php";
    //Appel de la fonction du controlleur
    envoiEmailInscriptionUtilisateur();
}
```

1. Créer une route qui appel une vue et une fonction du contrôleur/UtilisateursControleur.php

LA VUE :

1. Créer votre vues/utilisateurs/inscription_utilisateur.php
2. Créer un formulaire méthode POST avec 3 champs
3. E-mail + mot de passe + répéter mot de passe
4. Ajouter un bouton de validation avec un attribut name= ‘btn-valide-inscription’

LE MODÈLES:

1. Créer un singleton connexion a PHP PDO
2. Créer une classe Database.php dans le dossier modèles
3. Un booléen nous permet supprimer les instances multiple de la classe PDO

```php
<?php

class Database
{
    private $db_host = "localhost";
    private $db_dbname = "petite_annonces";
    private $db_user = "root";
    private $db_pass = "";

    protected $isConnected;

    public function getPDO(){
        //Par defaut la connexion est nul
        $this->isConnected = null;
        //Si c nul
        if($this->isConnected === null){
            //On essaie de ce connecter
            try{
                $this->isConnected = new PDO( dsn: "mysql:host=".$this->db_host.";dbname=".$this->db_dbname.";charset=utf8", $this->db_user, $this->db_pass);
                //DEBUG DE PDO MySQL
                $this->isConnected->setAttribute( attribute: PDO::ATTR_ERRMODE, value: PDO::ERRMODE_EXCEPTION);
                //echo "C bon t connecté";
                //ici avec le bool isConnected : true ou false = 1 instance de la classe PDO
                return $this->isConnected;
            //Sinon on declenche une erreur
            }catch (PDOException $exception){
                echo "Erreur de connexion à PDO";
                die();
            }
        }
    }
}
```

4. **Créer une classe modèles/EmailInscriptionUtilisateur.php**
5. Cette classe hérite de la classe Database.php et donc de la méthode getPDO()

6. Installer phpmailer via composer et votre terminal Windows: https://github.com/PHPMailer/PHPMailer
7. composer require phpmailer/phpmailer
8. Créer 2 propriétés privée e-mail et password
9. Créer une fonction EmailInscriptionUtilisateur()
10. Ajouter les infos mailTrap
11. Nettoyer les champs de votre formulaire avec trim et htmlspecialchar()
12. Vérifié que l'e-mail n'existe pas déjà

```php
//Verifié que le nom et l'email n'existe pas deja
$checkDatas = $db->prepare( statement: "SELECT * FROM utilisateurs WHERE email_utilisateur = ?");
$checkDatas->execute(array(
    $this->email_utilisateur
));
//Parcours des nom et email des utilisateurs (1 a la fois)
$utilisateur = $checkDatas->fetch();
//Si le nom ou email existe deja = on affiche une erreur et stop le process
if($utilisateur){
    ?>
    <div class='container text-center'>
        <div class='alert alert-danger'>Ce email n'est pas disponible</div>
        <a href='inscription-utilisateur' class="btn btn-warning">Recommencer</a>
    </div>
    <?php
}else{
```

13. Sinon
14. Créer un rôle = 'utilisateur'
15. Une requête SQL INSERT INTO
16. Le bind des paramètres
17. Le hash des mots de passe
18. Exécuter la requête et stocker L'URL de retour sur le site a afficher dans un bouton de votre e-mail

```php
}else{
    //Sinon
    $role = "utilisateur";
    //Insertion des utilisateurs dans la table
    $ajouterUtlisateur = $db->prepare( statement: "INSERT INTO `utilisateurs`(`email_utilisateur`, `password_utilisateur`, `role`) VALUES (?,?,?)");
    //Liaison des paramètres
    $ajouterUtlisateur->bindParam( parameter: 1, &variable: $this->email_utilisateur);
    $ajouterUtlisateur->bindParam( parameter: 2, &variable: $this->password_utilisateur);
    $ajouterUtlisateur->bindParam( parameter: 3, &variable: $role);

    //Hash du mot de passe
    $hash_password = password_hash($this->password_utilisateur, algo: PASSWORD_DEFAULT);

    //Execution de la requête = le dernier paramètre est le mot de passe haché
    $ajouterUtlisateur->execute(array($this->email_utilisateur, $hash_password, $role));
    //Si ca marche on appel la redirection
    //Passer les valeur email + password dans url avec get
    //ATTENTION on passe de mailTrap a notre site URL est absolue = localhost/votreprojet/route
    $redirect = "http://localhost/Projet7_Poo_Mvc_Annonces/connexion-utilisateur";
    //Corp de la page HTML5 = ici le css est dans les balises
    $mail->Body = '
        <!DOCTYPE html>
           <html lang="fr">
           <head>
               <meta charset="UTF-8">
               <meta http-equiv="Content-Type" content="text/html">
               <title>Votre inscription sur Mic-Annonce.com</title>
               <meta name="viewport" content="width=device-width, initial-scale=1.0">
           </head>
           <body style="...">
           <div style="...">
```

19. Faite un récapitulatif des infos d'inscription dans le corps HTML du mail

```php
    //Passer les valeur email + password dans url avec get
    //ATTENTION on passe de mailTrap a notre site URL est absolue = localhost/votreprojet/route
    $redirect = "http://localhost/Projet7_Poo_Mvc_Annonces/connexion-utilisateur";
    //Corp de la page HTML5 = ici le css est dans les balises
    $mail->Body = '
        <!DOCTYPE html>
           <html lang="fr">
           <head>
               <meta charset="UTF-8">
               <meta http-equiv="Content-Type" content="text/html">
               <title>Votre inscription sur Mic-Annonce.com</title>
               <meta name="viewport" content="width=device-width, initial-scale=1.0">
           </head>
           <body style="...">
           <div style="...">
               <img src="https://encrypted-tbn0.gstatic.com/images?q=tbn:ANd9GcS6fV5-gvJoErCmW1i-kzcc5C0slzboniFycw&usqp=CAU" width="75px" height="75px"
           </div>
           <div style="...">
               <h1>Annonce.com</h1>
               <h2>Bonjour : '.$this->email_utilisateur.'</h2>
               <p>Vous êtes desormais inscrit sur le site Annonce.com merci de valider votre inscription avec le liens suivant</p><br />
               <p>Recapitulatif de vos information de connexion</p>
               <p>Email :<b style="..."> '.$this->email_utilisateur.'</b></p>
               <p>Mot de passe :<b style="...">'.$this->password_utilisateur.'</p>
               <br /><br />
               <a href="' . $redirect . '" style="...">Confimer votre inscription sur notre site</a><br />
               <br /><br />
               <p style="...">Merci d\'utiliser notre site web</p>
               <p style="...">Cordialement : Annonces.com: Michael MICHEL : Administrateur</p>
           </div>
           </body>
           </html>
           ';
    //Conversion de HTML5
    $mail->body = "MIME-Version: 1.0" . "\r\n";
    $mail->body .= "Content-type:text/html;charset=utf8" . "\r\n";


    //Envoi de email
    $mail->send();
```

20. Afficher un message de succès

```php
        //Envoi de email
        $mail->send();
        //Message de succes + bouton pour aller a la connexion
        echo "<div class='container text-center'>
                    <div class='w-100 alert alert-success text-center'>Merci pour votre inscription,
                    un email de validation vous à été envoyé, merci de validé votre inscription pour acceder à votre tableau de bord.</div>
                    <a href='accueil' class='btn btn-warning'>ACCUEIL< /a>
                </div>";
    }

} catch (Exception $e) {
    echo "<p class='alert alert-danger'>Une erreur est survenue lors de votre inscription, merci de vérifié les champs !</p>";
    die();
}
```

21. **LE CONTRÔLEUR:**

22. Créer un contrôleur : contrôleur/UtilisateursContoleur.php

23. Dans ce fichier : appeler le modèle ../modèles/EmailInscriptionUtilisateur.php (require_once)
24. Créer une fonction envoEmailInscriptionUtilusateur()
25. Stocker une instance de la classe modèles/EmailInscriptionUtilisateur() dans une variables

```php
require_once "../modeles/Utilisateurs.php";

//Inscription et envoi d'email Utilisateurs
function envoiEmailInscriptionUtilisateur(){
    //Instance de la classe Email
    $emailUtilisateur = new EmailInscriptionUtilisateur();

    //Les champs du formulaire

    if(isset($_POST['email_utilisateur']) && !empty($_POST['email_utilisateur'])){
        $email = trim(htmlspecialchars($_POST['email_utilisateur']));
    }else{
        ?>
            <div class="text-center">
                <p class="alert alert-warning">Merci de remplir le champ email !</p>
            </div>
            <?php
    }
    if(isset($_POST['password_utilisateur']) && !empty($_POST['password_utilisateur'])){
        $password = trim(htmlspecialchars($_POST['password_utilisateur']));
    }else{
        ?>
        <div class="text-center">
            <p class="alert alert-warning">Merci de remplir le champ mot de pass !</p>
        </div>
        <?php
    }
```

26. Vérifié les champs du formulaire

```php
//La soumission du formulaire = recup de attribut name du bouton
if(isset($_POST['btn-valide-inscription'])){

    //Check des email et mot de passe avec preg_match php et les regexs
    if (preg_match( pattern: '/[-0-9a-zA-Z.+_]+@[-0-9a-zA-Z.+_]+.[a-zA-Z]{2,4}$/', $email)) {
        echo "<div class='container'>
            <p class='alert alert-warning text-success'>Votre email est valide</p>
            </div>";
    }

    if(preg_match( pattern: '/^(?=.*\d)(?=.*[A-Za-z])[0-9A-Za-z!@#$%]{8,12}$/', $password)) {
        echo "<div class='container'>
            <p class='alert alert-warning text-success'>Votre mot de passe est valide !</p>
            </div>";
    }
```

27. Si on valide le formulaire au clic appliquer des regexs

28. Si les champs sont remplis et que le mot de passe répété match

```php
    //Vérification de l'existance des champs du formulaires d'inscription
    if(isset($email) && isset($password)){
        //var_dump($_POST['nom_utilisateur']);
        //var_dump($_POST['email_utilisateur']);
        //var_dump($_POST['password_utilisateur']);

        //Check mot passe === mot de passe repeter
        if($_POST['password_utilisateur'] === $_POST['password_repeter']){
            $valideForm = true;
            if($valideForm){
                $emailUtilisateur->validerInscriptionUtilisateurEmail();
                ?>
                <style>
                    #form-register-user{
                        display: none;
                    }
                </style>
                <?php
            }
        }else{
            //Erreur si les 2 mot de passe ne sont pas identiques
            echo "<p class='alert alert-danger'>Le mot de passe répeter n'est pas identique au mot de passe.</p>";
        }
    }else{
        //SINON AFFICHE UNE ERREUR :
        echo "<p class='alert alert-danger'>Erreur ! Merci de remplir tous les champs.</p>";
    }
}
```

29. Appeler la méthode validerInscriptionUtilisateurEmail de la classe EmailInscriptionUtilisateur.php
30. Si ça marche : cacher le formulaire et afficher le message de succès du modèles

LE ROUTER:

1. Appeler votre fichier contrôleur/UtilisateurControleur.php
2. Si la route ≡ inscription-utilisateur
3. Appeler la vue et la fonction envoiEmailInscriptionUtilisateur() du contrôleur qui appel la méthode du modèle et la requête SQL

ÉTAPES 3 : CONNECTER DES UTILISATEURS

LE ROUTEUR :

1. Ajouter une route qui appel votre vue et la fonction connexionUtilisateur() du contrôleur

```php
//*****************CONNECTER UTILISATEUR*********************//
elseif ($url === "connexion-utilisateur"){
    $title = "CONNEXION -Mic Annonces-";
    require_once "utilisateurs/connexion_utilisateur.php";
    //Appel de la fonction du controlleur
    connexionUtilisateur();

}
```

LE MODÈLE :

2. Créer une classe modèles/Utilisateurs.php
3. Créer votre classe qui hérite de Database.php
4. Créer vos propriétés privée e-mail + password + rôle
5. Créer une méthode connecterUtilisateur()
6. Appeler la méthode getPDO() de la classe mère
7. Associé les champs du formulaire ($_POST) a vos propriétés privées
8. Écrire votre requête SQL (filtrer par rôle) + créer une requête préparée + bind des paramètres + exécution de la requête

```php
//Connecter un utilisateur
public function connecterUtilisateur(){
    //Connexion a la base de données
    $db = $this->getPDO();
    //On assigne les champs du formulaire aux propriétées privées
    $this->email_utilisateur = $_POST['email_utilisateur'];
    $this->password_utilisateur = $_POST['password_utilisateur'];
    $this->role = $_POST['role'];

    //Requête SQL
    $sql = "SELECT * FROM utilisateurs WHERE email_utilisateur = ? AND role = ?";

    //Requête préparée
    $stmt = $db->prepare($sql);
    //Liés les params du formulaire a la table
    $stmt->bindParam( parameter: 1,  &variable: $this->email_utilisateur);
    $stmt->bindParam( parameter: 2,  &variable: $this->role);

    //Execute la requête
    $stmt->execute(array(
        $this->email_utilisateur,
        $this->role
    ));
```

9. Compter les éléments de la table et vérifié les entrées
10. Vérifié le mot de passe haché et définir une redirection en fonctions des rôles (admin et user)

11. Si ça marche on redirige vers gestion_annonce qui est le tableau de bord de chaque utilisateur et donne accès au CRUD

```php
//Compter les elements dans table et verifié qu'il y a au - une valeur
if($stmt->rowCount() >= 1){
    $row =  $stmt->fetch();

        if($this->email_utilisateur === $row['email_utilisateur'] && password_verify($this->password_utilisateur, $row['password_utilisateur']) && $this->role === $row['role
            if ($row['role'] === "utilisateur") {
                session_start();
                $_SESSION['connecter_utilisateur'] = true;
                //On stock dans une variable de session l'id de chaque utilisateur
                $_SESSION['id_utilisateur'] = $row['id_utilisateur'];
                //Son email
                $_SESSION['email_utilisateur'] = $this->email_utilisateur;
                //La redirection
                header( string: "Location: gestion_annonces");
            }
            if($row['role'] === "administrateur"){
                session_start();
                $_SESSION['connecter_admin'] = true;
                $_SESSION['email_admin'] = $this->email_utilisateur;
                header( string: "Location: espace_administration");

            }
        }else{
            echo "<p class='alert alert-danger'>Erreur ! Merci de vérifié votre email et mot de passe pour les utilisateurs</p>";
        }


}else{
    echo "<p class='alert alert-danger'>Aucun utilisateur ne possèdent cet email et mot de passe: Erreur de fetch</p>";

}
```

LE CONTRÔLEUR :

12. Dans votre fichier contrôleur/UtilisateurControleur.php
13. Créer une fonction connexionUtilisateur()

```php
//Connecter un utilisateurs
function connexionUtilisateur(){
    //Instance de la classe Utilisateur
    $utilisateurClasse = new Utilisateurs();
    //Check des champs
    if(isset($_POST['btn-valide-connexion'])) {
        //Vérification de l'existance des champs du formulaires d'inscription
        if (isset($_POST['email_utilisateur']) && isset($_POST['password_utilisateur']) && isset($_POST['role'])) {
            $utilisateurClasse->connecterUtilisateur();
        }
    }

}
```

14. Cette fonction est appelée quand la route est égale a connexion-utilisateur

```php
//*****************CONNECTER UTILISATEUR*********************//
elseif ($url === "connexion-utilisateur"){
    $title = "CONNEXION -Mic Annonces-";
    require_once "utilisateurs/connexion_utilisateur.php";
    //Appel de la fonction du controlleur
    connexionUtilisateur();


}
```

15. Test

ÉTAPES 4 : LE CRUD DES ANNONCES PAR UTILISATEURS

LE ROUTEUR

1. Créer une une route gestion_annonce et des conditions d'accès

```php
//******************TABLEUR DE BORD UTILISATEUR*************//
elseif ($url === "gestion_annonces" && $_SESSION['connecter_utilisateur'] && $_SESSION['connecter_utilisateur'] === true){
    $title = "TABLEAU DE BORD UTILISATEUR -Mic Annonces-";
    afficherLesAnnoncesParUtilisateur();
}
```

2. Cette routes appel la méthode afficherAnnoncesParUtilisateur() du contrôleur

LE MODÈLE :

1. Créer une classe Annonces qui hérite de Database
2. Créer vos propriétés privées de la table annonces
3. Créer une méthode afficherAnnoncesParUtilisateur()
4. Ajouter la connexion a PDO + la requête SQL filtrer par utilisateur + l'id récupérer dans la session + la requête préparée + le bind des paramètres + exécution de la requête + fetchAll pour récupérer toutes les annonces par utilisateurs

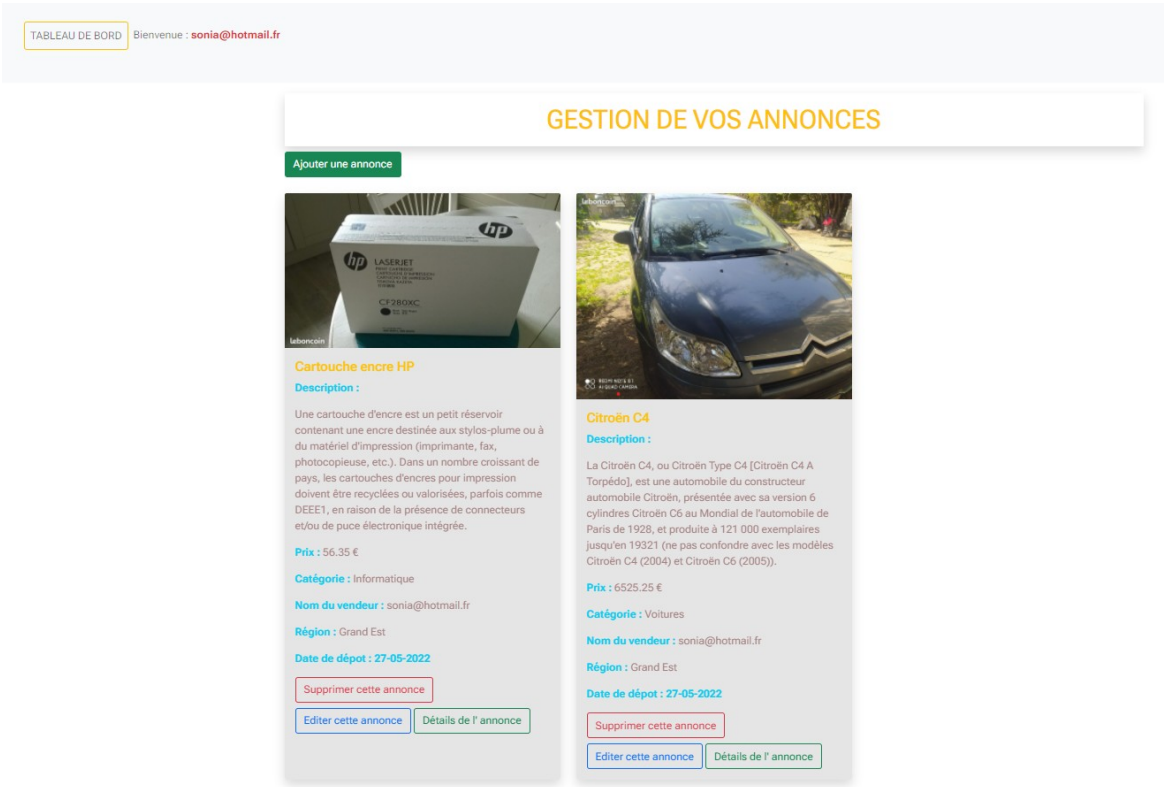
```php
////////////////////////////////////POUR LES UTILISATEURS INSCRITS////////////////////////////////

//AFFICHER TOUTES LES ANNONCES PAR UTILISATEUR////
public function afficherAnnoneParUtilisateur(){
    //Connexion a PDO
    $db = $this->getPDO();
    //Requête SQL + jointure
    $sql = "SELECT * FROM annonces INNER JOIN utilisateurs
            ON annonces.utilisateur_id = utilisateurs.id_utilisateur INNER JOIN categories
            ON annonces.categorie_id = categories.id_categorie INNER JOIN regions
            ON annonces.regions_id = regions.id_regions WHERE utilisateur_id = ?";
    //Recup de id utilisateur
    $this->id_annonce = $_SESSION['id_utilisateur'];
    //Requête préparée
    $request = $db->prepare($sql);
    //Lié les paramètres
    $request->bindParam( parameter: 1,  &variable: $this->id_annonce);

    //Execution de la requête
    $request->execute();
    //Retourne un objet de resultats

    return $request->fetchAll();
}
```

LE CONTRÔLEUR :

1. Créer un contrôleur/AnnoncesControleur.php
2. Appeler le fichier du modèles/Annonces.php via require_once
3. Créer une fonction (appelée dans le routeur) afficherlesAnnoncesParUtilisateurs()
4. Créer une instance du modèle stocké dans une variables
5. Dans une seconde variable = appeler la méthode du modèle
6. Appeler la vue gestion_annonces_utilisateur
7. Retourner la seconde variables

```php
    ////////////////////POUR LES UTILISATEURS/////////////////////
    ///
    //Afficher le annonces par utilisateur
    function afficherLesAnnoncesParUtilisateur()
    {
        //Insatnce du de la classe Annonce
        $annonce = new Annonces();
        $annonceParUtilisateur = $annonce->afficherAnnoneParUtilisateur();
        require_once "../vues/utilisateurs/gestion_annonces_utilisateur.php";
        return $annonceParUtilisateur;
    }
```

8. Cette méthode est appelée dans le routeur index.php qui appel la vue (exemple pour Sonia du Grand Est)



ÉTAPES 5 : AJOUTER UNE ANNONCES PAR UTILISATEUR

1. Dans votre vue : gestion_annonces ajouter un bouton qui appel une vue et un formulaire d'ajout d'annonce
2. `<a href="ajouter_annonce" class="btn btn-success">Ajouter une annonce</a>`

LE ROUTEUR

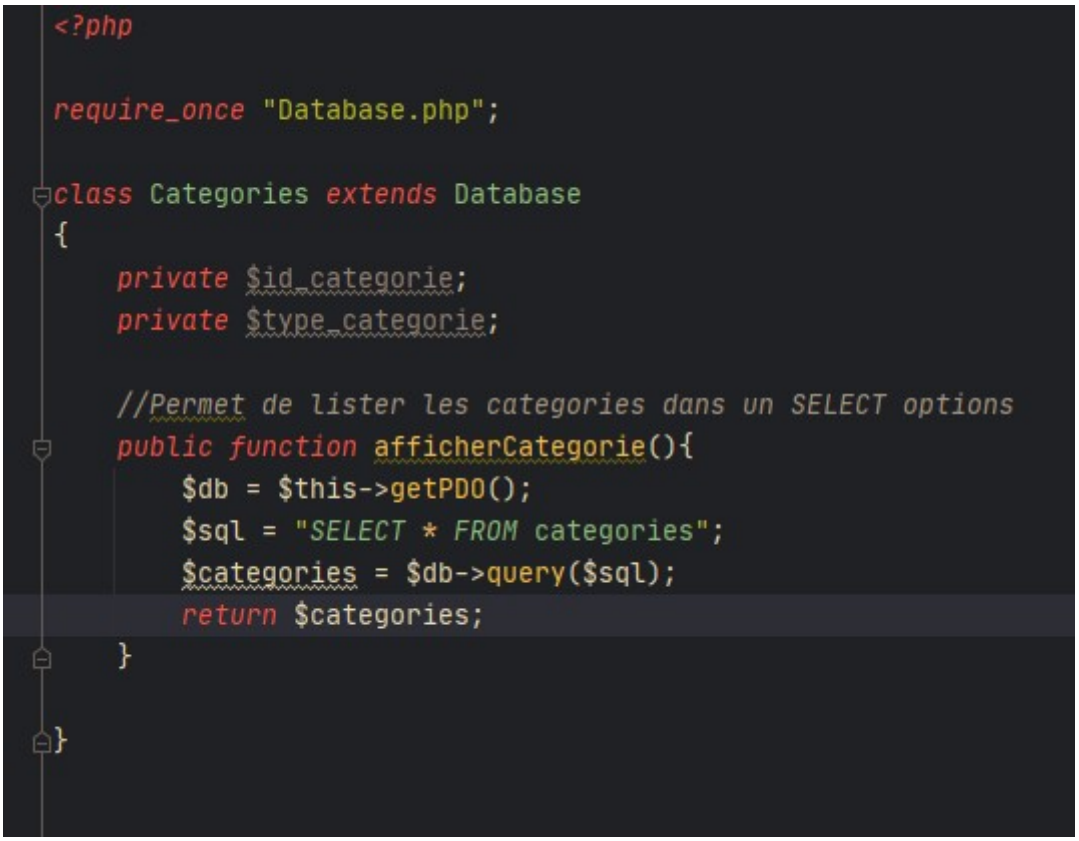
1. Créer une route 'ajouter_annonce' et des condition d'accès de la vue

```php
//**********************AJOUTER ANNONCE UTILISATEUR*************//
elseif ($url === "ajouter_annonce" && $_SESSION['connecter_utilisateur'] && $_SESSION['connecter_utilisateur'] === true){
    $title = "AJOUTER ANNONCES UTILISATEUR -Mic Annonces-";
    ajouterAnnonceParUtilisateur();
}
```

LA VUE

1. Créer un formulaire méthode POST avec les champs de la table Annonces
2. Ajouter les champs : nom, description, prix, date de dépôt catégorie, région , un champ caché id utilisateur (session) et upload de la photos
3. **Pour les catégories :**
4. Créer une classe Catégories dans le dossier modèles
5. Cette classe hérite de Database.php

6. Créer les propriétés privées id + type catégories
7. Créer une méthode afficherCategorie() + appel a PDO via la classe mère + une requête SQL de sélection + appel de la méthode query de PDO + un retour de la variables $categories



```php
<?php

require_once "Database.php";

class Categories extends Database
{
    private $id_categorie;
    private $type_categorie;

    //Permet de lister les categories dans un SELECT options
    public function afficherCategorie(){
        $db = $this->getPDO();
        $sql = "SELECT * FROM categories";
        $categories = $db->query($sql);
        return $categories;
    }

}
```

8. Créer un fichier CategoriesControleur.php
9. Appeler le fichier du modèles (require_once ‘’)
10. Créer une fonction afficherToutesCategories()
11. Dans une variables créer une instance de la classe modèle Catégories
12. Dans une autres variable : appeler la méthode afficherCategorie() du modèle
13. Dans une boucle foreach afficher toutes les catégorie via un alias
14. Dans la balise HTML <option> value=’id categorie’ et on affiche le type_categorie dans une fenêtre déroulante
15. Faite la même opération pour la table Région

```php
<div class="mb-3">
    <label for="categorie_id"></label>
    <select name="categorie_id" class="form-control">
        <?php
        afficherToutesCategories();
        ?>
    </select>
</div>

<input type="hidden" value="<?= $_SESSION['id_utilisateur'] ?>" name="id_utilisateur">

<div class="mb-3">
    <label for="regions_id"></label>
    <select name="regions_id" class="form-control">
        <?php
        listerRegions()
        ?>
    </select>
</div>
```

16. Le champ caché récupère l'id de l'utilisateur grâce a la connexion et la création de la variable de session : $_SESSION['id_utilisateur']
17. Chaque annonces est donc ajoutée pour UN UTILISATEURS

LE MODÈLE

1. Créer une méthode ajouterUneAnnonce()
2. Ajouter la connexion a PDO via la méthode getPDO() de la classe mère
3. Désinfecter chaque champ du formulaire via trim et htmlspecialchar()
4. Écrire la requête SQL INSERT INTO
5. Créer une requête préparée + le bind des paramètres + exécution de la requête
6. On retourne une variable $insert

```php
public function ajouterUneAnnonce(){
    $db = $this->getPDO();
    //Les propriétés privée
    $this->nom_annonce = trim(htmlspecialchars($_POST['nom_annonce']));
    $this->description_annonce = trim(htmlspecialchars($_POST['description_annonce']));
    $this->prix_annonce = trim(htmlspecialchars($_POST['prix_annonce']));
    $this->date_depot = trim(htmlspecialchars($_POST['date_depot']));
    $this->photo_annonce = trim(htmlspecialchars($_POST['photo_annonce']));
    $this->categorie_id = trim(htmlspecialchars($_POST['categorie_id']));
    $this->utilisateur_id = trim(htmlspecialchars($_POST['id_utilisateur']));
    $this->region_id = trim(htmlspecialchars($_POST['regions_id']));

    //Requête SQL :
    $sql = "INSERT INTO `annonces`(`nom_annonce`, `description_annonce`, `prix_annonce`, `photo_annonce`, `date_depot`, `categorie_id`, `utilisateur_id`, `regions_id`) VALUE

    //Requête préparée
    $insert = $db->prepare($sql);

    //Liés les paramètre du formulaire a la table phpmyadmin
    $insert->bindParam( parameter: 1,  &variable: $this->nom_annonce);
    $insert->bindParam( parameter: 2,  &variable: $this->description_annonce);
    $insert->bindParam( parameter: 3,  &variable: $this->prix_annonce);
    $insert->bindParam( parameter: 4,  &variable: $this->photo_annonce);
    $insert->bindParam( parameter: 5,  &variable: $this->date_depot);
    $insert->bindParam( parameter: 6,  &variable: $this->categorie_id);
    $insert->bindParam( parameter: 7,  &variable: $this->utilisateur_id);
    $insert->bindParam( parameter: 8,  &variable: $this->region_id);

    $insert->execute(array(
        $this->nom_annonce,
        $this->description_annonce,
        $this->prix_annonce,
        $this->photo_annonce,
        $this->date_depot,
        $this->categorie_id,
        $this->utilisateur_id,
        $this->region_id
    ));
    return $insert;
```

7. Cette méthode est appelée dans le contrôleur

LE CONTRÔLEUR

1. Dans votre contrôleur : ajouter une fonction ajouterAnnonceParUtilisateur()
2. Dans une variable créer une instance de la classe Annonces
3. Recouper attribut name du bouton qui déclenche un evénement au clic
4. Ajouter le code d'upload de la photo
5. Vérifier les champs du formulaire
6. Si tous les champs existent et ne sont pas vide
7. Appeler la méthode du modèles
8. Si ça marche afficher un message de succès et une redirection après 3 secondes puis une redirection vers le tableau de bord

```php
function ajouterAnnonceParUtilisateur()
{
    require_once "../vues/utilisateurs/ajouter_annonces_utilisateur.php";
    $annonce = new Annonces();

    //Si on click
    if (isset($_POST['btn_ajouter_annonce'])) {
        //Upload de la photo
        if (isset($_FILES['photo_annonce'])) {
            $repertoire = "../assets/img/";
            $photo_annonce = $repertoire . basename($_FILES['photo_annonce']['name']);
            $_POST['photo_annonce'] = $photo_annonce;
            if (move_uploaded_file($_FILES['photo_annonce']['tmp_name'], $photo_annonce)) {
                echo "<p class='alert alert-success'>Le fichier est valide et téléchargé avec succès !</p>";
            } else {
                echo "<p class='alert alert-danger'>Erreur lors du téléchargement de votre fichier !</p>";
            }
        } else {
            echo "<p class='alert alert-danger'>Le fichier est invalide seul les format .png, .jpg, .bmp, .svg, .webp sont autorisé !</p>";
        }
        if (isset($_POST['nom_annonce']) && isset($_POST['description_annonce']) && isset($_POST['prix_annonce']) && isset($_POST['photo_annonce']) && isset($_POST['date_depot'])
            $ajouterAnnonce = $annonce->ajouterUneAnnonce();
            if ($ajouterAnnonce) {
                ?>
                <style>
                    #ajouter-annonce-form {
                        display: none;
                    }
                </style>
                <?php
                echo "<p class='alert alert-success'>Votre annonce a bien été ajouté, veuillez patienter vous allez etre rediriger !</p>";
                header( string: "Refresh:3; url=gestion_annonces", replace: true, http_response_code: 303);
            } else {
                echo "<p class='alert alert-danger'>Une erreur est survenue durant l'ajout de votre annonce merci de réessayé !</p>";
            }
        } else {
            echo "<p class='alert alert-danger'>Erreur : Merci de remplir tous les champs !</p>";
        }
    }
```

9. Cette fonction est appelée dans le router quand la route = /gestion_annonces

ÉTAPES 6 : SUPPRIMER UNE ANNONCES PAR UTILISATEUR

LE ROUTEUR

1. Depuis votre tableau de bord : ajouter un bouton <a> avec la route qui recupere la clé primaire de chaque annonces
2. `<a href="supprimer_annonce&id_suppr=<?= $data['id_annonce'] ?>" type="button" class="btn btn-outline-danger">Supprimer cette annonce</a>`
3. Depuis le routeur : la route appel la fonction supprimerAnnonces du contrôleur

```php
//**************SUPPRIMER ANNONCE UTILISATEUR*************//
elseif ($url === "supprimer_annonce"  && isset($_SESSION['connecter_utilisateur']) && $_SESSION['connecter_utilisateur'] === true && isset($_GET['id_suppr']) && $_GET['id_suppr'] >
    $title = "Annonces.com -SUPPRIMER ANNONCES-";
    supprimerAnnonceUtilisateur();
}
```

LE MODÈLE

1. Créer une méthode supprimerAnnonces()
2. Ajouter votre connexion a PDO via la méthode de la classe mère
3. Créer votre requête SQL + bind de l'id récupérer dans l'URL via $_GET['id']
4. Binder le paramètres + exécuter

```php
/*****************SUPPRIMER ANNONCES UTILISATEURS*************/
//Supprimer une annonce par utilisateur
public function suprimerAnnonce(){
    //Appel de  la classe mere et de la methode PDO
    $db = $this->getPDO();
    //Requête sql
    $sql = 'DELETE FROM `annonces` WHERE `id_annonce` = ?';
    //Creation de la requête péparée
    $stmt = $db->prepare($sql);
    $this->id_annonce = $_GET['id_suppr'];
    //Lié les paramètre (ici id de annonce a $_GET id url)
    $stmt->bindParam( parameter: 1,  &variable: $this->id_annonce);
    //Execution de la requête
    $delete = $stmt->execute();
    //Retourne l'objet avec son id
    return $delete;
}
```

LE CONTRÔLEUR

1. Créer une fonction supprimerAnnoncesParutilisateur()
2. Créer une instance de la classe Annonces
3. Appeler la méthode supprimer du modèle
4. Si ça marche : afficher un message de succès et effectuer une redirection après 3 secondes

```php
//Supprimer une annone d'un utilisateur
function supprimerAnnonceUtilisateur()
{
    //Instance du model (classe) annonce
    $annonce = new Annonces();
    $delete = $annonce->suprimerAnnonce();
    if ($delete) {
        echo "<p class='alert alert-success'>Votre annonce a bien été supprimer, veuillez patienter vous allez etre rediriger !</p>";
        header( string: "Refresh:3; url=gestion_annonces",  replace: true,  http_response_code: 303);
    } else {
        echo "rien a supprimer";
    }

}
```

5. Cette méthode est appelée dans le routeur quand la route = supprimer_annonce

ÉTAPES 7 : ÉDITER UNE ANNONCES PAR UTILISATEUR

LE ROUTEUR

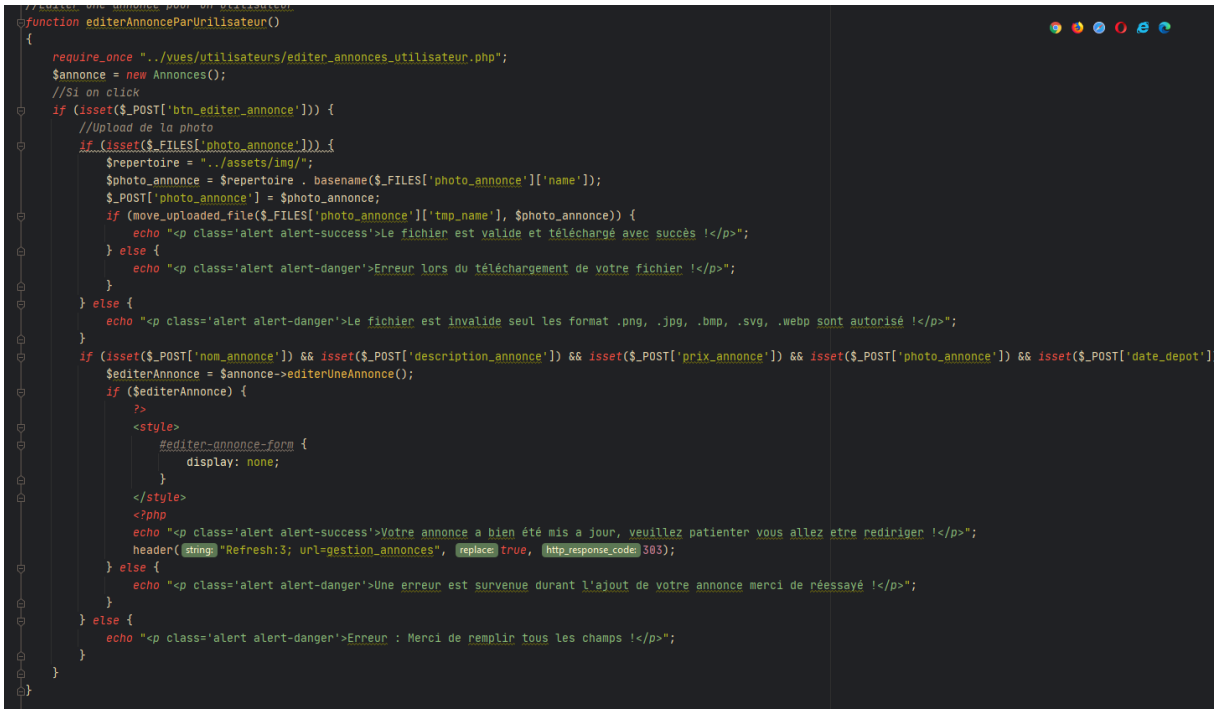
1. Depuis la tableau de bord : ajouter un bouton avec :
2. <a class="mt-2 btn btn-outline-primary" href="editer_annonce&id_details=<?= $data['id_annonce'] ?>">Éditer cette annonce
3. Puis ajouter la route a index.php et les conditions d'accès

```php
}
//***************EDITER ANONCE UTILISATEURS******************//
elseif ($url === "editer_annonce" && $_SESSION['connecter_utilisateur'] && $_SESSION['connecter_utilisateur'] === true && isset($_GET['id_details']) && $_GET['id_details'] > 0){
    $title = "AJOUTER ANNONCES UTILISATEUR -Mic Annonces-";
    editerAnnonceParUrilisateur();
}
```

LE MODÈLE

```php
//***************METTRE A JOUR ANNONCE*********************//
//Editer une annonce par utilisateur
public function editerUneAnnonce(){
    $db = $this->getPDO();
    //Les propriétés privée
    $this->nom_annonce = trim(htmlspecialchars($_POST['nom_annonce']));
    $this->description_annonce = trim(htmlspecialchars($_POST['description_annonce']));
    $this->prix_annonce = trim(htmlspecialchars($_POST['prix_annonce']));
    $this->date_depot = trim(htmlspecialchars($_POST['date_depot']));
    $this->photo_annonce = trim(htmlspecialchars($_POST['photo_annonce']));
    $this->categorie_id = trim(htmlspecialchars($_POST['categorie_id']));
    $this->utilisateur_id = trim(htmlspecialchars($_POST['id_utilisateur']));
    $this->region_id = trim(htmlspecialchars($_POST['regions_id']));
    $this->id_annonce = $_GET['id_details'];

    //La requête sql
    $sql = "UPDATE `annonces` SET `nom_annonce`= ?,`description_annonce`= ?,`prix_annonce`= ?,`photo_annonce`= ?,`date_depot`= ?,`categorie_id`= ?,`utilisateur_id`= ?,`regions
    //La requête préparée
    $update = $db->prepare($sql);
    //Execution de la requête
    $update->execute(array(
        $this->nom_annonce,
        $this->description_annonce,
        $this->prix_annonce,
        $this->photo_annonce,
        $this->date_depot,
        $this->categorie_id,
        $this->utilisateur_id,
        $this->region_id,
        $this->id_annonce));
    return $update;
}
```

LE CONTRÔLEUR



```php
function editerAnnonceParUrilisateur()
{
    require_once "../vues/utilisateurs/editer_annonces_utilisateur.php";
    $annonce = new Annonces();
    //Si on click
    if (isset($_POST['btn_editer_annonce'])) {
        //Upload de la photo
        if (isset($_FILES['photo_annonce'])) {
            $repertoire = "../assets/img/";
            $photo_annonce = $repertoire . basename($_FILES['photo_annonce']['name']);
            $_POST['photo_annonce'] = $photo_annonce;
            if (move_uploaded_file($_FILES['photo_annonce']['tmp_name'], $photo_annonce)) {
                echo "<p class='alert alert-success'>Le fichier est valide et téléchargé avec succès !</p>";
            } else {
                echo "<p class='alert alert-danger'>Erreur lors du téléchargement de votre fichier !</p>";
            }
        } else {
            echo "<p class='alert alert-danger'>Le fichier est invalide seul les format .png, .jpg, .bmp, .svg, .webp sont autorisé !</p>";
        }
        if (isset($_POST['nom_annonce']) && isset($_POST['description_annonce']) && isset($_POST['prix_annonce']) && isset($_POST['photo_annonce']) && isset($_POST['date_depot'])
            $editerAnnonce = $annonce->editerUneAnnonce();
            if ($editerAnnonce) {
                ?>
                <style>
                    #editer-annonce-form {
                        display: none;
                    }
                </style>
                <?php
                echo "<p class='alert alert-success'>Votre annonce a bien été mis a jour, veuillez patienter vous allez etre rediriger !</p>";
                header( string: "Refresh:3; url=gestion_annonces", replace: true, http_response_code: 303);
            } else {
                echo "<p class='alert alert-danger'>Une erreur est survenue durant l'ajout de votre annonce merci de réessayé !</p>";
            }
        } else {
            echo "<p class='alert alert-danger'>Erreur : Merci de remplir tous les champs !</p>";
        }
    }
}
```

1. Cette fonction est appelée dans le routeur quand la route = éditer_annonce

ÉTAPES 8 : AFFICHER TOUTES LES ANNONCES SUR LA PAGE D'ACCUEIL

LE ROUTEUR

1. Quand la route = accueil : on appel la fonction du contrôleur afficherLesAnnonces()
2. On récupérer également les attributs des boutons des formulaire de recherche et au clic on appel les fonctions de recherche du contrôleur

```php
//******************PAGE ACCUEIL *********************//
elseif($url === "accueil"){
    $title = "ACCUEIL -Mic Annonces-";
    afficherLesAnnonces();
    if(isset($_POST['btn-search-text'])){
        //echo $_POST['recherche'];
        rechercheGlobaleMotCle();
        //header("Location: resultat-recherche-texte");
    }
    if(isset($_POST['btn-search-CR'])){
        getAnnonceByCategorieAndRegion();
    }
}
```

LA VUE

1. Ajouter les 2 formulaires méthodes POST de recherche (par mot clé + par catégories et région)
2. Dans une boucle foreach afficher toutes les annonces grâce a la variable retournée par le contrôleur

```php
<div class="container mt-3">
    <div class="text-center">
        <h3 class="text-warning">RECHERCHER UNE ANNONCE</h3>
        <form method="post" class="w-50" style="...">
            <input type="search" name="recherche" class="form-control" required placeholder="Rechercher une annonce"/>
            <button type="submit" name="btn-search-text" class="btn btn-outline-success mt-3">Rechercher</button>
        </form>
    </div>
</div>
```

3. Recherche par nom

```php
<div class="container">
    <div class="text-center">
        <h3 class="text-info">Rechercher une annonce par catégorie et région</h3>

        <div id="search-form" class="d-flex justify-content-center">
            <form class="form-inline text-center mt-2" method="post">
                <div class="row">
                    <div class="col-md-6 sol-sm-12">
                        <div class="form-group mb-2">
                            <select name="categorie_id" id="categorie" class="form-control form-search-item">
                                <?php
                                afficherToutesCategories();
                                ?>
                            </select>
                        </div>
                    </div>
                    <div class="col-md-6 sol-sm-12">
                        <div class="form-group mb-2 ml-2">
                            <select class="form-control" id="stock" name="region_id">
                                <?php
                                listerRegions()
                                ?>
                            </select>
                        </div>
                    </div>
                </div>

                <div class="form-group mb-2 ml-2">
                    <button type="submit" name="btn-search-CR" class="btn btn-outline-warning">Rechercher</button>
                </div>
            </form>
        </div>
    </div>
</div>
iv>
```

4. Recherche par catégorie et région

LE MODÈLE

1. La méthode afficher toutes les annonces coté visiteur

```php
////////////////////////////////////POUR LES SIMPLES VISITEURS////////////////////////////////////
public function afficherToutesAnnonces(){
    $db = $this->getPDO();

    //Requête SQL + limite
    $sql = "SELECT * FROM annonces
            INNER JOIN utilisateurs ON annonces.utilisateur_id = utilisateurs.id_utilisateur
            INNER JOIN categories ON annonces.categorie_id = categories.id_categorie
            INNER JOIN regions ON annonces.regions_id = regions.id_regions ORDER BY Rand() DESC";
    $annonces = $db->query($sql);

    return $annonces;
}
```

LE CONTRÔLEUR

```php
//////////////////////POUR LES SIMPLES VISITEURS//////////////
//POUR LES VISITEURS//////
function afficherLesAnnonces()
{
    //Instance de la classe Annonce
    $annonce = new Annonces();
    //stock dansune variable l'appel de la methode concernée
    $recupAnnonce = $annonce->afficherToutesAnnonces();
    require_once "../vues/accueil.php";
    return $recupAnnonce;
}
```

1. Cette fonction est appelée quand le route = accueil

ÉTAPES 9 : LES RECHERCHES PAR MOT CLÉ + RÉGION ET CATÉGORIE

A a partir de la route accueil on accède au 2 formulaire de recherches qui appel les méthodes rechercheGlobale et afficherAnnonceParCategorieEtRegion

LE MODÈLE

Pour la recherche par mot clé :

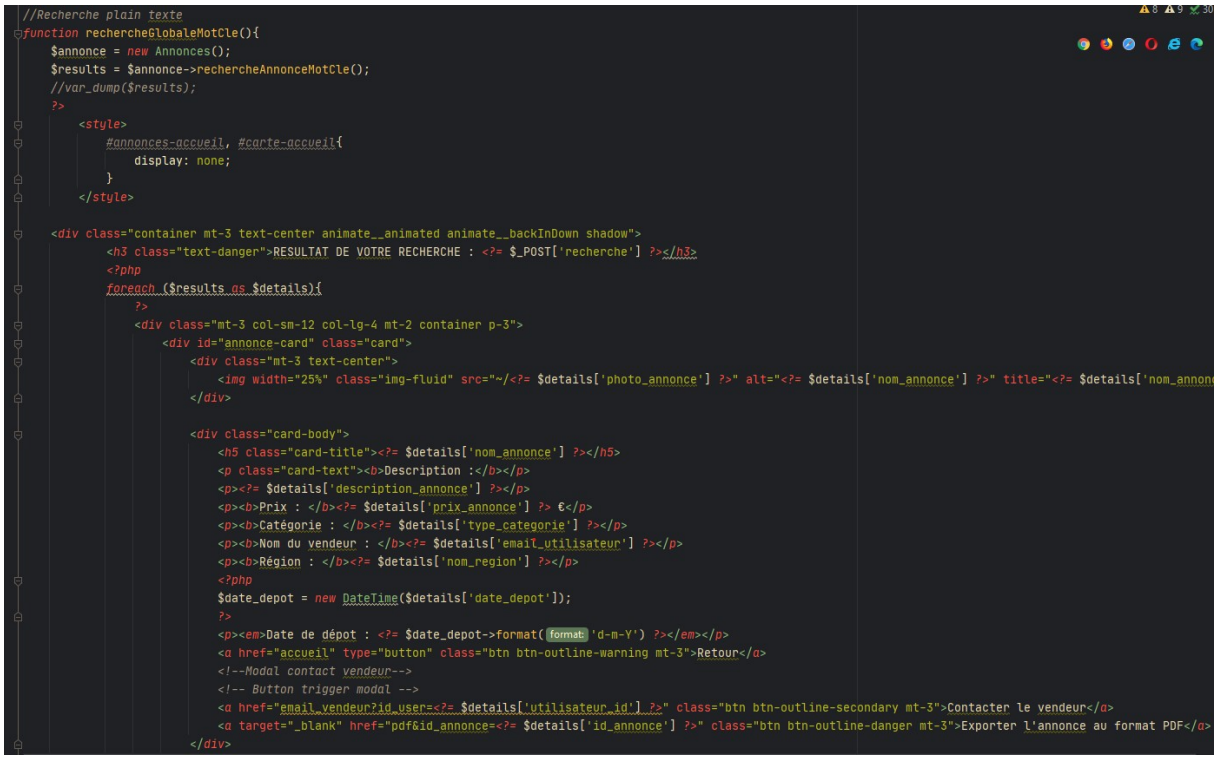
```php
//*****************ANNONCE PAR MOT CLE**************************//
public function rechercheAnnonceMotCle(){
    $db = $this->getPDO();
    //Recup de input recherche
    $recherche = $_POST['recherche'];
    //var_dump($recherche);


    $sql = "SELECT * FROM annonces
            INNER JOIN utilisateurs ON annonces.utilisateur_id = utilisateurs.id_utilisateur
            INNER JOIN  categories ON annonces.categorie_id = categories.id_categorie
            INNER JOIN regions ON annonces.regions_id = regions.id_regions
            WHERE nom_annonce LIKE '%$recherche%'
            OR description_annonce LIKE '%$recherche%'
            OR prix_annonce LIKE '%$recherche%'
            OR type_categorie LIKE '%$recherche%' OR nom_region LIKE '%$recherche%'";
    $res = $db->query($sql);
    if($res){
        return $res;
    }else{
        ?>
            <h3 class="text-danger">PAS DE RESULTAT</h3>
        <?php
    }


}
```

Pour la recherche par catégorie et région :

```php
//*****************ANNONCE PAR CATEGORIE ET REGION*************//

//Afficher les détails de l'annonce par regions et categories
public function getAnnonceByRegionAndCategorie(){
    $db = $this->getPDO();

    //Recup de input recherche
    if(isset($_POST['categorie_id']) && isset($_POST['region_id'])){
        $cat = $_POST['categorie_id'];
        $reg = $_POST['region_id'];
    }else{
        $cat = 1;
        $reg = 1;
        if(empty($cat) || empty($reg)){
            echo "<p class='alert-danger mt-2 p-2'>Merci de remplir les champs Catégorie et Région</p>";
        }
    }


    $sql = "SELECT * FROM annonces
            INNER JOIN utilisateurs ON annonces.utilisateur_id = utilisateurs.id_utilisateur
            INNER JOIN  categories ON annonces.categorie_id = categories.id_categorie
            INNER JOIN regions ON annonces.regions_id = regions.id_regions
            WHERE regions_id = ? AND categorie_id = ? ";
    $stmt = $db->prepare($sql);

    $stmt->bindParam( parameter: 1,  &variable: $_POST['region_id']);
    $stmt->bindParam( parameter: 2,  &variable: $_POST['categorie_id']);
    $stmt->execute();
    return $stmt->fetchAll();
}
```

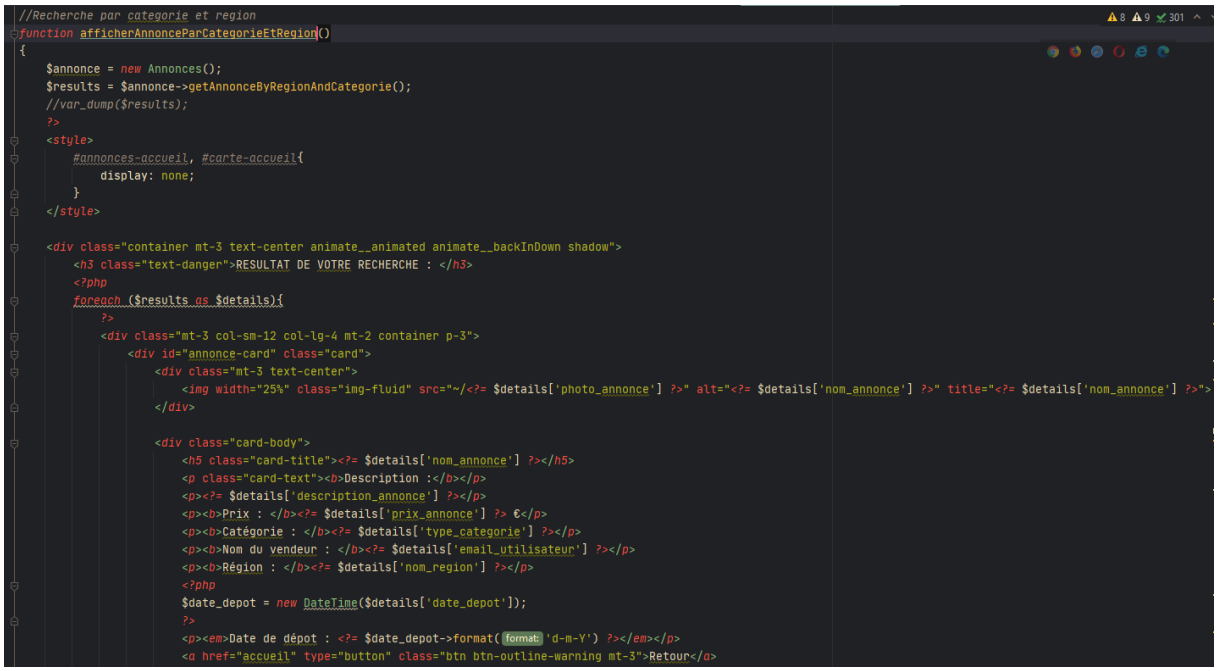
LE CONTRÔLEUR

Pour la recherche par mot clé :



```php
//Recherche plain texte
function rechercheGlobaleMotCle(){
    $annonce = new Annonces();
    $results = $annonce->rechercheAnnonceMotCle();
    //var_dump($results);
    ?>
        <style>
            #annonces-accueil, #carte-accueil{
                display: none;
            }
        </style>

    <div class="container mt-3 text-center animate__animated animate__backInDown shadow">
            <h3 class="text-danger">RESULTAT DE VOTRE RECHERCHE : <?= $_POST['recherche'] ?></h3>
            <?php
            foreach ($results as $details){
                ?>
                <div class="mt-3 col-sm-12 col-lg-4 mt-2 container p-3">
                    <div id="annonce-card" class="card">
                        <div class="mt-3 text-center">
                            <img width="25%" class="img-fluid" src="~/<?= $details['photo_annonce'] ?>" alt="<?= $details['nom_annonce'] ?>" title="<?= $details['nom_annonc
                        </div>

                        <div class="card-body">
                            <h5 class="card-title"><?= $details['nom_annonce'] ?></h5>
                            <p class="card-text"><b>Description :</b></p>
                            <p><?= $details['description_annonce'] ?></p>
                            <p><b>Prix : </b><?= $details['prix_annonce'] ?> €</p>
                            <p><b>Catégorie : </b><?= $details['type_categorie'] ?></p>
                            <p><b>Nom du vendeur : </b><?= $details['email_utilisateur'] ?></p>
                            <p><b>Région : </b><?= $details['nom_region'] ?></p>
                            <?php
                            $date_depot = new DateTime($details['date_depot']);
                            ?>
                            <p><em>Date de dépot : <?= $date_depot->format( format: 'd-m-Y') ?></em></p>
                            <a href="accueil" type="button" class="btn btn-outline-warning mt-3">Retour</a>
                            <!--Modal contact vendeur-->
                            <!-- Button trigger modal -->
                            <a href="email_vendeur?id_user=<?= $details['utilisateur_id'] ?>" class="btn btn-outline-secondary mt-3">Contacter le vendeur</a>
                            <a target="_blank" href="pdf&id_annonce=<?= $details['id_annonce'] ?>" class="btn btn-outline-danger mt-3">Exporter l'annonce au format PDF</a>
                        </div>
```

Pour la recherche par catégorie et région :



```php
//Recherche par categorie et region
function afficherAnnonceParCategorieEtRegion()
{
    $annonce = new Annonces();
    $results = $annonce->getAnnonceByRegionAndCategorie();
    //var_dump($results);
    ?>
    <style>
        #annonces-accueil, #carte-accueil{
            display: none;
        }
    </style>

    <div class="container mt-3 text-center animate__animated animate__backInDown shadow">
        <h3 class="text-danger">RESULTAT DE VOTRE RECHERCHE : </h3>
        <?php
        foreach ($results as $details){
            ?>
            <div class="mt-3 col-sm-12 col-lg-4 mt-2 container p-3">
                <div id="annonce-card" class="card">
                    <div class="mt-3 text-center">
                        <img width="25%" class="img-fluid" src="~/<?= $details['photo_annonce'] ?>" alt="<?= $details['nom_annonce'] ?>" title="<?= $details['nom_annonce'] ?>">
                    </div>

                    <div class="card-body">
                        <h5 class="card-title"><?= $details['nom_annonce'] ?></h5>
                        <p class="card-text"><b>Description :</b></p>
                        <p><?= $details['description_annonce'] ?></p>
                        <p><b>Prix : </b><?= $details['prix_annonce'] ?> €</p>
                        <p><b>Catégorie : </b><?= $details['type_categorie'] ?></p>
                        <p><b>Nom du vendeur : </b><?= $details['email_utilisateur'] ?></p>
                        <p><b>Région : </b><?= $details['nom_region'] ?></p>
                        <?php
                        $date_depot = new DateTime($details['date_depot']);
                        ?>
                        <p><em>Date de dépot : <?= $date_depot->format( format: 'd-m-Y') ?></em></p>
                        <a href="accueil" type="button" class="btn btn-outline-warning mt-3">Retour</a>
```

ÉTAPES 10 : LA CARTE DE FRANCE ET LA RECHERCHES PAR RÉGION

Télécharger l'archive sur le site et décompresser le contenus dans votre projet :

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

LA VUE

1. Appel du fichier css + jQuery + France-map.js

```
<!--CARTE DE FRANCE ET RECHERCHE PAR REGION -->
<div id="carte-accueil" class="mt-3 container text-center animate__animated animate__backInDown shadow">
    <h2 class="text-center text-danger">VOTRE REGION :</h2>
    <link rel="stylesheet" href="assets/carte/cmap/style.css">
    <script src="assets/carte/cmap/jquery-1.11.1.min.js"></script>
    <script src="assets/carte/cmap/France-map.js"></script>
    <script>
        francefree();
    </script>
</div>
```

2. Dans une balise script in appel la fonction francefree()

LE JQUERY

1. Dans la variables objet path : chaque clé url doit posséder une route + la clé primaire de chaque région

```
// MAP LINKS  ////////////////////////////////////////////////////////////////////
//Url = $url du routeur + cle id = valeur
let paths = {
            R0: {
            title: "Grand-Est",
            url: "region&id=11",
            path: "258,68, 257,67, 256,68, 255,66, 253,65, 252,65, 252,66, 251,67, 249,67, 249,66, 248,63, 246,62, 246,61, 245,60, 243,58, 241,58, 238,56, 237,57, 236,57, 236,
            },
```

2. La suite du code rend la carte interactive
3. Au clic sur une région, on est donc redirigé vers la route = region&id= CLÉ PRIMAIRE DE CHAQUE RÉGION

LE ROUTEUR

1. Cette route appel une fonction du contrôleur + un id en paramètres

```php
//******************CARTE DE FRANCE*********************//
elseif ($url === "region"){
    $title = "Annonce -ANNONCE PAR REGION-";
    $id = $_GET['id'];
    annonceParRegion($_GET['id']);
}
```

LE MODÈLE RÉGION

1. La méthode prend un paramètre id

```php
public function afficherAnnonceParRegion($id){
    //Afficher les détails de l'annonce par regions
    $db = $this->getPDO();
    $sql = "SELECT *  FROM annonces
            INNER JOIN utilisateurs ON annonces.utilisateur_id = utilisateurs.id_utilisateur
            INNER JOIN  categories ON annonces.categorie_id = categories.id_categorie
            INNER JOIN regions ON annonces.regions_id = regions.id_regions
            WHERE regions_id = ?";
    $stmt = $db->prepare($sql);
    $stmt->execute(array($id));
    $getRegion = $stmt->fetchAll();
    return $getRegion;

}
```

LE CONTRÔLEUR RÉGION

```php
function annonceParRegion($id){
    $region = new Regions();
    $id = $_GET['id'];
    $annonceParRegion = $region->afficherAnnonceParRegion($id);
    if($annonceParRegion){
        require_once '../vues/annonces/annonce_region.php';
    }else{
        echo "<p class='alert-warning text-center p-2 mt-2'><b>Pas d'annonce pour cette region</b></p>";
    }
}
```

LA VUE

```php
<div class="mt-3 container animate__animated animate__backInDown p-3 shadow">
<div class="row">

    <?php
    foreach ($annonceParRegion as $row){
    ?>
    <h1 class="mt-3 text-center text-warning">ANNONCES DE LA REGION : <br>
        <b class="text-info text-uppercase"><?= $row['nom_region'] ?></b>
    </h1>
    <div class="col-md-12 col-sm-12 mt-3">
        <div class="card"">
        <div class="mt-3 text-center rounded">
            <img width="25%" class="img-fluid" src="~/<?= $row['photo_annonce'] ?>" alt="<?= $row['nom_annonce'] ?>" title="<?= $row['nom_annonce'] ?>">
        </div>

        <div class="card-body">
            <h5 class="card-title"><?= $row['nom_annonce'] ?></h5>
            <p class="card-text"><b>Description :</b></p>
            <p><?= $row['description_annonce'] ?></p>
            <p><b>Prix : </b><?= $row['prix_annonce'] ?> €</p>
            <p><b>Catégorie : </b><?= $row['type_categorie'] ?></p>
            <p><b>Nom du vendeur : </b><?= $row['email_utilisateur'] ?></p>
            <p><b>Région : </b><?= $row['nom_region'] ?></p>
            <?php
            $date_depot = new DateTime($row['date_depot']);
            ?>
            <p><em>Date de dépot : <?= $date_depot->format( format: 'd-m-Y') ?></em></p>
            <a href="accueil" type="button" class="btn btn-outline-warning mt-3">Retour</a>
            <!--Modal contact vendeur-->
            <!-- Button trigger modal -->
            <a href="email_vendeur?id_user=<?= $row['utilisateur_id'] ?>" class="btn btn-outline-secondary mt-3">Contacter le vendeur</a>
            <a target="_blank" href="pdf&id_annonce=<?= $row['id_annonce'] ?>" class="btn btn-outline-danger mt-3">Exporter l'annonce au format PDF</a>
        </div>
    </div>
</div>

<?php
}
?>
</div>
```

ÉTAPES 11 : LES DÉTAILS D'UNE ANNONCES

1. Au clic sur le bouton PLUS D'INFOS appel la route = details-annonce-visiteur

```php
//*****************DETAILS ANNONCES VISITEUR*****************//
elseif ($url === "details-annonce-visiteur" && isset($_GET['id_details']) && $_GET['id_details'] > 0){
    $title = "DETAILS ANNONCES -Mic Annonces-";
    afficherDetailsVisiteurs();
}
```

2. Cette route est accessible depuis le bouton plus d'infos de la page d'accueil
3. `<a href="details-annonce-visiteur&id_details=<?= $data['id_annonce'] ?>" class="btn btn-outline-success mt-2">Détails de l' annonce</a>`

LE MODÈLE

1.

```php
//AFFICHE LES DETAILS 1 ANNONCES PAR UTILISATEUR///
public function afficherDetailsUneAnnonce(){
    //Coonexion PDO
    $db = $this->getPDO();
    $sql = "SELECT * FROM annonces
            INNER JOIN utilisateurs ON annonces.utilisateur_id = utilisateurs.id_utilisateur
            INNER JOIN categories ON annonces.categorie_id = categories.id_categorie
            INNER JOIN regions ON annonces.regions_id = regions.id_regions
            WHERE id_annonce = ?";
    //Recup de id utilisateur
    $this->id_annonce = $_GET['id_details'];
    //Requête préparée
    $request = $db->prepare($sql);
    //Lié les paramètres
    $request->bindParam( parameter: 1,  &variable: $this->id_annonce);

    //Execution de la requête
    $request->execute();
    //Retourne un objet de resultats
    $details = $request->fetch();
    return $details;
}
```

LE CONTRÔLEUR

```php
function afficherDetailsVisiteurs()
{
    //Instance de la classe Annonce
    $annonce = new Annonces();
    $details = $annonce->afficherDetailsUneAnnonce();
    require_once "../vues/annonces/details-annonce-visiteur.php";
    return $details;
}
```

LA VUE

```php
<div class="container w-25 animate__animated animate__backInDown">
    <h1 class="text-success">Annonce numéro: <?= $details['id_annonce'] ?></h1>
    <div id="annonce-card" class="card">
        <div class="mt-3 text-center">
            <img width="50%"  class="" src="~/<?= $details['photo_annonce'] ?>" alt="<?= $details['nom_annonce'] ?>" title="<?= $details['nom_annonce'] ?>">
        </div>

        <div class="card-body">
            <h5 class="card-title"><?= $details['nom_annonce'] ?></h5>
            <p class="card-text"><b>Description :</b></p>
            <p><?= $details['description_annonce'] ?></p>
            <p><b>Prix : </b><?= $details['prix_annonce'] ?> €</p>
            <p><b>Catégorie : </b><?= $details['type_categorie'] ?></p>
            <p><b>Nom du vendeur : </b><?= $details['email_utilisateur'] ?></p>
            <p><b>Région : </b><?= $details['nom_region'] ?></p>
            <?php
            $date_depot = new DateTime($details['date_depot']);
            ?>
            <p><em>Date de dépot : <?= $date_depot->format( format: 'd-m-Y') ?></em></p>
            <a href="accueil" type="button" class="btn btn-outline-warning mt-3">Retour</a>
            <!--Modal contact vendeur-->
            <!-- Button trigger modal -->
            <a href="email_vendeur?id_user=<?= $details['utilisateur_id'] ?>" class="btn btn-outline-secondary mt-3">Contacter le vendeur</a>
            <a target="_blank" href="pdf&id_annonce=<?= $details['id_annonce'] ?>" class="btn btn-outline-danger mt-3">Exporter l'annonce au format PDF</a>
        </div>
    </div>
</div>
```

1. Ici un bouton permet d'exporter l'annonce au format PDF
2. `<a target="_blank" href="pdf&id_annonce=<?= $details['id_annonce'] ?>" class="btn btn-outline-danger mt-3">Exporter l'annonce au format PDF</a>`

ÉTAPES 12 : LES DÉTAILS D'UNE ANNONCES EXPORTER AU FORMAT PDF

LE ROUTEUR

```php
//*****************EXPORT ANNONCE EN PDF************************//
elseif ($url === "pdf" & isset($_GET['id_annonce']) && $_GET['id_annonce'] > 0) {
    annoncePDF($_GET['id_annonce']);
}
```

LE MODÈLE

1. Rendez vous sur le site : http://www.fpdf.org/ et telecharger l'archive a décompresser dans votre projet
2. Consulter la documentation

```php
/*******************EXPORTER ANNONCE AU FORMAT PDF***************/
public function pdfExportParId($annonceID){
    ob_get_clean();
    //Instance de la classe
    require "../assets/FPDF/fpdf.php";
    $db = $this->getPDO();
    $query = "SELECT * FROM annonces
            INNER JOIN utilisateurs ON annonces.utilisateur_id = utilisateurs.id_utilisateur
            INNER JOIN categories ON annonces.categorie_id = categories.id_categorie
            INNER JOIN regions ON annonces.regions_id = regions.id_regions
            WHERE id_annonce = ?";
    $req = $db->prepare($query);
    $_GET['id_annone'] = $annonceID;
    $req->bindParam( parameter: 1, &variable: $annonceID);
    $req->execute();
    $details_annonces = $req->fetch();

    $this->nom_annonce = $details_annonces['nom_annonce'];
    $this->description_annonce = $details_annonces['description_annonce'];
    $this->prix_annonce = $details_annonces['prix_annonce'];
    $this->photo_annonce = $details_annonces['photo_annonce'];
    $this->categorie_id = $details_annonces['type_categorie'];
    $this->utilisateur_id = $details_annonces['email_utilisateur'];
    $this->region_id = $details_annonces['nom_region'];

    $pdf = new FPDF( orientation: 'P', unit: 'mm', size: 'A4');
    //Sortie
    $pdf->AddPage();
    //Header
    $pdf->Image( file: '../assets/img/logo-min1.png');

    $pdf->SetFont( family: 'Arial', style: 'B', size: 16);
    $pdf->Cell( w: 40, h: 10, txt: 'Votre annonces : ');
    $pdf->Ln( h: 10);
    $pdf->Cell( w: 190, h: 10, iconv( in_charset: 'UTF-8', out_charset: 'ISO-8859-2', str: 'Nom du annonce : '.$this->nom_annonce));

    $pdf->Ln( h: 10);
    $pdf->Image( file: ''.$details_annonces['photo_annonce'], x: 100, y: 200, w: 100, h: 100);


    $pdf->Ln( h: 10);
    $pdf->SetFont( family: 'Arial', style: '', size: 12);
    $pdf->MultiCell( w: 190, h: 10, txt: 'Description de l\'annonce : '.utf8_decode($this->description_annonce), border: 1, align: 'J');
    $pdf->Ln( h: 10);
    $pdf->Cell( w: 190, h: 10, txt: 'Prix du produit : '.$this->prix_annonce. ' EUROS');
    $pdf->Ln( h: 10);
    $pdf->Cell( w: 190, h: 10,iconv( in_charset: 'UTF-8', out_charset: 'ISO-8859-2', str: 'Catégorie : '.$this->categorie_id));
    $pdf->Ln( h: 10);
    $pdf->Cell( w: 190, h: 10, txt: 'Nom du vendeur : '.$this->utilisateur_id);
    $pdf->Ln( h: 10);
    $pdf->Cell( w: 190, h: 10,iconv( in_charset: 'UTF-8', out_charset: 'ISO-8859-2', str: 'Région : '.$this->region_id));
    $pdf->Ln( h: 10);
    //$pdf->AddLink("http://localhost/bon_coin_mic/region&id=3");


    $pdf->Output( dest: '', name: 'annonces.pdf', isUTF8: true);
}
```

LE CONTRÔLEUR

```php
//Exporter l'annonce ne PDF
function annoncePDF($id){
    $annonce = new Annonces();
    $_GET['id_annonce'] = $id;
    $afficherPDF = $annonce->pdfExportParId($id);
    return $afficherPDF;
}
```

Chaque PDF s'ouvre dans un nouvel ongle via

ÉTAPES 13 : ENVOYER UN E-MAIL AU VENDEUR

LE ROUTEUR

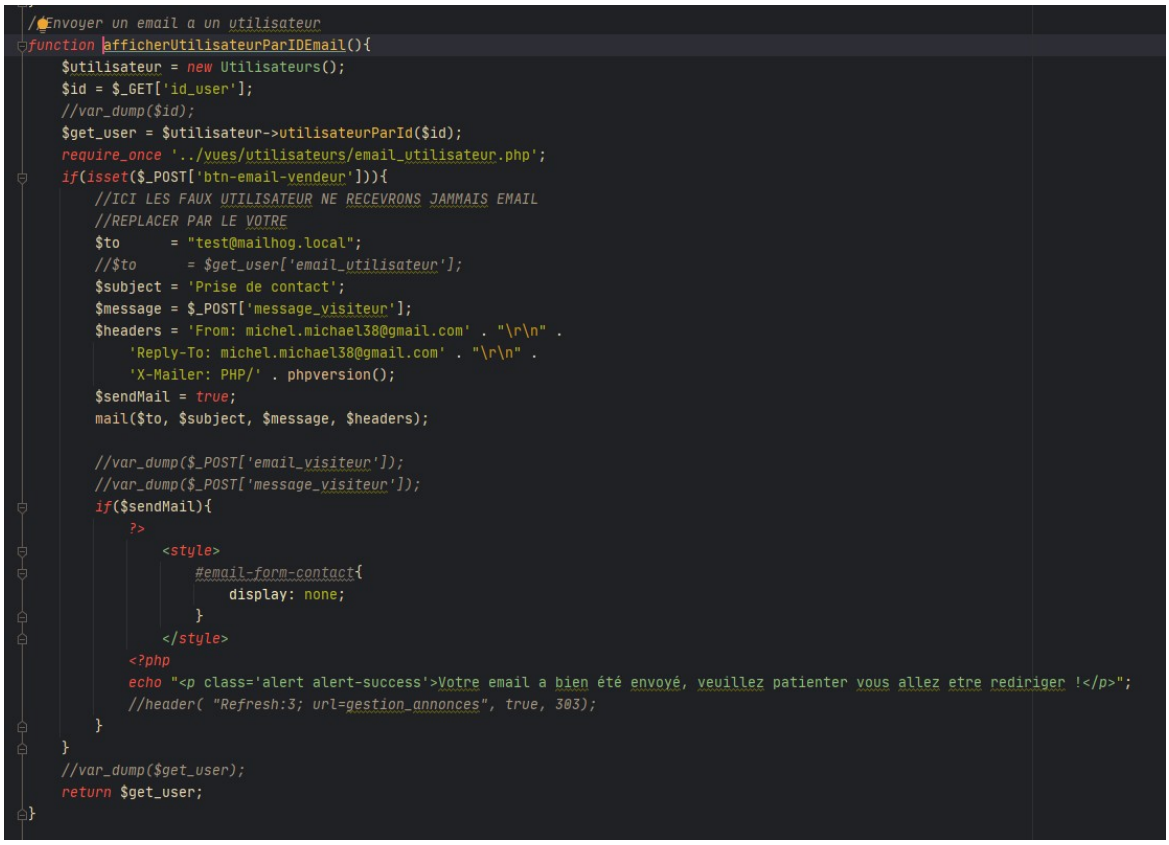
```php
//*************EMAIL VENDEUR*****************************//
elseif ($url === "email_vendeur") {
    $title = "CONTACTER UN VENDEUR -Mic Annonces-";
    afficherUtilisateurParIDEmail();
}
```

LE MODÈLE UTILISATEUR

```php
//Utilisateur par id
public function utilisateurParId($id){
    $db = $this->getPDO();
    $sql = "SELECT *  FROM utilisateurs WHERE id_utilisateur = ?";
    $stmt = $db->prepare($sql);
    $_GET['id_user'] = $id;
    $stmt->bindParam( parameter: 1,  &variable: $id);
    $stmt->execute();
    $get_user = $stmt->fetch();
    return $get_user;
}
```

LE CONTRÔLEUR

1. Ici on utilise la fonction native PHP mail(paramètres)



```php
//Envoyer un email a un utilisateur
function afficherUtilisateurParIDEmail(){
    $utilisateur = new Utilisateurs();
    $id = $_GET['id_user'];
    //var_dump($id);
    $get_user = $utilisateur->utilisateurParId($id);
    require_once '../vues/utilisateurs/email_utilisateur.php';
    if(isset($_POST['btn-email-vendeur'])){
        //ICI LES FAUX UTILISATEUR NE RECEVRONS JAMMAIS EMAIL
        //REPLACER PAR LE VOTRE
        $to      = "test@mailhog.local";
        //$to      = $get_user['email_utilisateur'];
        $subject = 'Prise de contact';
        $message = $_POST['message_visiteur'];
        $headers = 'From: michel.michael38@gmail.com' . "\r\n" .
            'Reply-To: michel.michael38@gmail.com' . "\r\n" .
            'X-Mailer: PHP/' . phpversion();
        $sendMail = true;
        mail($to, $subject, $message, $headers);

        //var_dump($_POST['email_visiteur']);
        //var_dump($_POST['message_visiteur']);
        if($sendMail){
            ?>
                <style>
                    #email-form-contact{
                        display: none;
                    }
                </style>
            <?php
            echo "<p class='alert alert-success'>Votre email a bien été envoyé, veuillez patienter vous allez etre rediriger !</p>";
            //header( "Refresh:3; url=gestion_annonces", true, 303);
        }
    }
    //var_dump($get_user);
    return $get_user;
}
```

2. Après l'envoi on effectue une redirection vers la page d'accueil

BRAVO ! VOUS SAVEZ MAINTENANT RÉALISER UNE APPLICATION WEB PHP AVEC LE PATRON DE CONCEPTION MVC