ÉTUDE « CRUD COMPLET»

« Julie ESTRUCH »

SOMMAIRE

1.

2.

1.INTRODUCTION 5
2.PRÉ-REQUIS 6
3.LA CONNEXION A LA BASE DE DONNÉES 6
4.LES « TABLES RÉCAPITULATIVES » 7
5.L'AJOUT DE DONNÉES 13
6.LA MODIFICATION DE DONNÉES 16
7.L'AFFICHAGE DES DONNÉES 16
8.LA SUPPRESSION DES DONNÉES 16

PRÉFACE

Ce document est destiné à toute personne appartenant au groupe de développeurs concerné, et uniquement à ces derniers. Cet outil est à utiliser dans le cadre de la mise en place du CRUD de chaque table, aussi appelée « tables récapitulatives ».

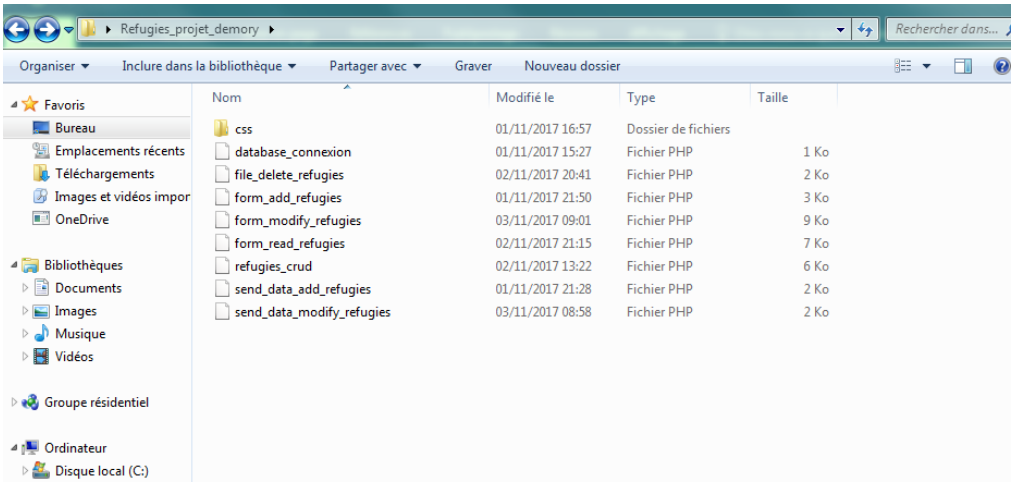
AVANT-PROPOS

Avant toute chose je préfères vous prévenir : cette veille n'est pas évidente. Elle n'a pas été évidente à rédiger, elle ne le sera pas non plus à lire. Pour cause, elle est complète et utilise un niveau de PHP qui n'est pas piqué des hannetons. Je vous demande donc de bien vouloir lire consciencieusement ce document avant de m'appeler à l'aide.

1. INTRODUCTION

Nous allons voir dans cette étude comment faire un CRUD complet. Pour se faire je vais m'appuyer sur certaines choses que Mathis a abordé tout en approfondissant certains autres points. L'objectif de cette étude est qu'à la fin de la lecture de cette dernière, vous soyez capable de créer un CRUD de A à Z (formulaires, boutons et tables récapitulatives comprises).

Cette étude risque d'être longue, en effet, voici le nombre de fichier PHP que nous allons créer (juste pour UNE table).



Pour partir d'un bon pied, j'aimerai que chacun d'entre vous utilise les mêmes noms de fichier qu'ici (ou que l'on se concerte) de sorte à ce que chaque fichier soit parlant et qu'il n'y ait pas besoin de changer leurs noms pour que les codes marchent les uns avec les autres.

Pour une meilleure lisibilité, je vais découper mon étude en 6 parties :

- ⇨ Database_connexion : La connexion à la base de données
- ⇨ Refugies_Crud : La tables récapitulative
- ⇨ L'ajout de donnée (form_add_refugies & send_data_add_refugies)
- ⇨ La modification de données (form_modify_refugies & send_data_modify_refugies)
- ⇨ L'affichage de données (form_read_refugies)
- ⇨ La suppression de données (form_delete_refugies)

2. PRÉ-REQUIS

Comme nous l'expliquais Mathis dans son études, on a la possibilité d'utiliser wamp, mais si (comme moi :p) vous n'êtes pas à l'aise avec ce dernier, vous pouvez utiliser easy-php. Cela ne change strictement rien - juste le fait que (selon moi) easy-php soit plus « user-friendly ».

Il vous faudra également la base de données réalisée par Antoine et Valentin, elle a normalement été commit. Enfin, il vous faudra bien entendu un IDE pour coder en PHP. Et il faudra un peu de bootstrap (je vous renvoie vers l'étude de Thomas)

3. LA CONNEXION A LA BASE DE DONNÉES

Mathis l'avait déjà expliqué dans son étude, mais je pense que cela ne peut pas faire de mal de la remettre également ici. Donc avant toute chose, il vous faudra faire la liaison entre la base de données et le code. Pour se faire, créer un fichier de connexion que vous nommerez « database_connexion.php ».

```php
<?php
class Database
{
  private static $dbName = 'refugies_project' ;
  private static $dbHost = 'localhost' ;
  private static $dbUsername = 'root';
  private static $dbUserPassword = '';
  private static $cont = null;
```

Nous commençons par créer une classe « Database ». Nous initialisons nos variables selon les valeurs en notre possessions. Comme nous aurons tous la même base de données (et c'est pour ça que je vous pousse à ne RIEN changer au niveau des noms) vous pouvez directement récupérer ce début de code.

Une fois notre classe entamée, nous allons créer deux fonctions – plus le constructeur - qui vont respectivement nous servir à nous connecter à la base et à nous déconnecter.

```php
  public function __construct()
  {
    die('Init function is not allowed');
  }
```

Comme dans tout langage objet, nous n'oublions pas le constructeur de notre classe « Database ».

Die : constructeur de langage équivalent à « exit () » pour faire simple, il s'agit d'un simple retour.

```php
public static function connect()
{
  if ( null == self::$cont )
  {
    try
    {
      self::$cont = new PDO( "mysql:host=".self::$dbHost.";"."dbname=".self::$dbName, self::$dbUsername, self::$dbUserPassword);
    }
    catch(PDOException $e)
    {
      die($e->getMessage());
    }
  }
    return self::$cont;
}
```

Nous nous attaquons maintenant à la fonction « connect() » qui comme son nom l'indique… permet la connexion à la base de données.

Cette fonction ne retourne rien et ne prend rien en paramètre et n'est composé « que » d'une condition « if » couplée à un try/catch.

Ce code nous dit :
Si null est égale au $cont (self revient à dire this mais sous sa forme static) alors on essaie d'instancier une référence de PDO avec les paramètres nécessaires bien sûr) on n'oublie pas de catch les erreur potentielles, et on retourne un message avec die.

Après le if on retourne self ::$cont.

```php
    public static function disconnect()
    {
        self::$cont = null;
    }
}
?>
```

Enfin nous créons une fonction disconnect(), ou self ::$cont prendra tout simplement null.

4. LES « TABLES RÉCAPITULATIVES »

J'annonce : Ça va commencer à légèrement piquer alors restez concentrer.

Je créer un nouveau fichier que j'appelle « refugies_crud.php » (je vous conseille de garder cette syntaxe qui vous facilitera la vie soit : « nomTable_crud.php ».

```
<!DOCTYPE html>
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Listes des réfugiés</title>

        <link href="css/bootstrap.min.css" rel="stylesheet">
        <link href="css/responsive.css" rel="stylesheet">
    <img
src="
A7" data-wp-preserve="%3Cscript%20src%3D%22js%2Fbootstrap.js%22%3E%3C%2Fscript%3E"
data-mce-resize="false" data-mce-placeholder="1" class="mce-object" width="20" height="20"
alt="<script>" title="<script>" />


  </head>
```

On commence notre fichier par un peu de mise en forme avec du bootstrap, j'ai choisi ce style là mais on n'est pas du tout obligé de le suivre. Si vous souhaitez modifier cette partie vous le pouvez. – J'ai volontairement fait en sort que vous n'ayez qu'à copier/coller si besoin ☺ - Je ne m'étendrai pas sur le bootstrap car déjà ce n'est pas le sujet de cette étude, et parce que Thomas a déjà réaliser une veille dessus, nous pouvons donc continuer.

```
    <body>

<br />
<div class="container">

<br />
<div class="row">

<br />
<h2>Crud en Php</h2>
</div>


<br />
<div class="row">

                    <a href="form_add_refugies.php" class="btn btn-success">Ajouter un réfugié</a>


<br />
<div class="table-responsive">

<br />
<table class="table table-hover table-bordered">

<br />
```

Là encore ce n'est que de la mise en forme, je pars du principe que nous avons tous un minimum de html dans les pattes et que vous êtes donc capable de jouer avec des « div » si vous le souhaitez.

```
<br />

<thead>

<th>Nom</th>
<p>

<th>Prenom</th>
<p>

<th>DateNaiss</th>
<p>

<th>Illettre</th>
<p>

<th>Blesse</th>
<p>

<th>Conscient</th>
<p>

<th>Nationnalité</th>
<p>

<th>Camp</th>
<p>

<th></th>
<p>

<th></th>
<p>

</thead>
<p>
```

Maintenant nous allons créer un tableau qui contiendra les noms des champs de la table (l'entête des colonnes) sur laquelle nous travaillons. Pour que ce soit concordant, nous ferons en sorte que les champs soient dans le même ordre que dans notre base de données.

Comme vous pouvez le remarquer je n'ai pas indiqué l'ID, c'est normal, le tableau que vous êtes en train de créer sera utiliser certainement pas un novice de l'informatique, l'ID il s'en fout, de plus n'oubliez pas que vu que l'ID est auto incrémentée, il risquera d'y avoir des « trous » dans la base en cas de delete.

Ex : Nous avons cinq enregistrements, leurs ID vont de 1 à 5 et sont auto-incrémenté. Imaginons que nous supprimions l'enregistrement 4, puis que nous ajoutions un enregistrement, la suite d'ID sera donc : 1, 2, 3, 5, 6. L'ID 4 a été delete mais l'auto incrémentation continue. Un utilisateur néophyte ne comprendra donc pas l'absence de l'ID 4.

A partir de maintenant suivez bien.

```
<br />
<tbody>
                        <?php include 'database_connexion.php';
                          $pdo = Database::connect();
                          $pdo_camp =Database::connect();
                          $pdo_natio=Database::connect();
                          $sql = 'SELECT * FROM refugies ORDER BY idRefugies ';
                          $rep = $pdo->query($sql);
                          $camp_sql= 'SELECT * FROM camp';
                          $rep_camp= $pdo->query($camp_sql);
                          $nationalite_sql='SELECT * FROM nationalite';
                          $rep_natio = $pdo->query($nationalite_sql);
```

Nous avons notre tableau, mais nous n'avons aucun lien avec la base, nous allons donc créer la connexion. On inclut le fichier « database_connexion.php » que nous avons réalisé juste avant puis nous initialisons quelques variables :

$pdo = Database :: connect() -> Appel de la fonction connexion de la classe database.

Bien que c'est sale (genre vraiment), nous alons créer deux autres variables qui feront la même chose que $pdo : $pdo_camp et $pdo_natio.

$sql = Requête permettant de sélectionné tous les réfugiés ordonnés avec leur id de manière croissante.

$rep = Exécution de la requête de la variable $sql

On fera la même chose avec $nationalite_sql et $camp_sql. Cela nous servira un peu plus bas.

```php
foreach ($pdo->query($sql) as $row)
{

  foreach($pdo_camp->query($camp_sql) as $row_camp)
  {
    If ($row_camp['IdCamp']==$row['idCamp'])
    {
      $ville_camp=$row_camp['Ville'];
    }
  }

  foreach($pdo_natio->query($nationalite_sql) as $row_natio0)
  {
    If ($row_natio['IdNationalite']==$row['idNationalite'])
    {
      $nom_natio=$row_natio['NomPays'];
    }
  }

  If ( $row['Illetre']==1){
    $illetre="oui";
  }else{
    $illetre="non";
  }

  If ( $row['Blesse']==1){
    $blesse="oui";
  }else{
    $blesse="non";
  }

  If ( $row['Conscient']==1){
    $conscient="oui";
  }else{
    $conscient="non";
  }
  //Nous sommes toujours dans le premier foreach
```

Avant de rentrer les données de notre base dans le tableau, nous devons faire deux trois vérifications :

D'abord il nous faut un premier foreach pour parcourir notre table Réfugié.

Dans cette boucle nous allons créer deux autres foreach, un pour parcourir les camps l'autre pour les nationalités. Ces boucles servent à afficher les noms des villes des camps et des nationalités plutôt que leurs ID (chose qui n'aurait aucun sens pour notre utilisateur) pour se faire nous utilisons des variables « $nom_natio » et « $ville_camp » pour stocker les valeurs lors du parcours des tables concernées. Faites attention aux noms des « IdCamp » et « IdNationalité » en effet parfois le « i » est en minuscule ou en majuscule, référez-vous à la base de données en cas de doute.

Enfin nous allons créer trois conditions « if » qui nous serviront à afficher « oui » ou « non » pour les champs illetrés (oui il manque un « t » mais c'est ainsi dans la base), blessé et conscient. Sans cela l'utilisateur verrait « 1 » ou « 0 » selon l'état du réfugiés, et ça c'est pas très compréhensible pour l'utilisateur.

```
                            echo '

<tr>';
                            echo'

<td>' . $row['Nom'] . '</td>
<p>
';
                            echo'

<td>' . $row['Prenom'] . '</td>
<p>
';
                            echo'

<td>' . $row['DateNaiss'] . '</td>
<p>
';
                            echo'

<td>'. $illetre . '</td>
<p>
';
                            echo'

<td>' . $blesse . '</td>
<p>
';
                            echo'

<td>' . $conscient . '</td>
<p>
';
                            echo'

<td>' . $nom_natio . '</td>
<p>
';
                            echo'

<td>' . $ville_camp . '</td>
<p>
';
```

Toujours dans le premier « foreach », nous allons afficher nos variables. Il n'y a pas grand-chose à dire ici, vous savez tous (j'espère) faire un « echo ».

(On suit TOUJOURS l'ordre des entêtes pour la concordance des données !!!)

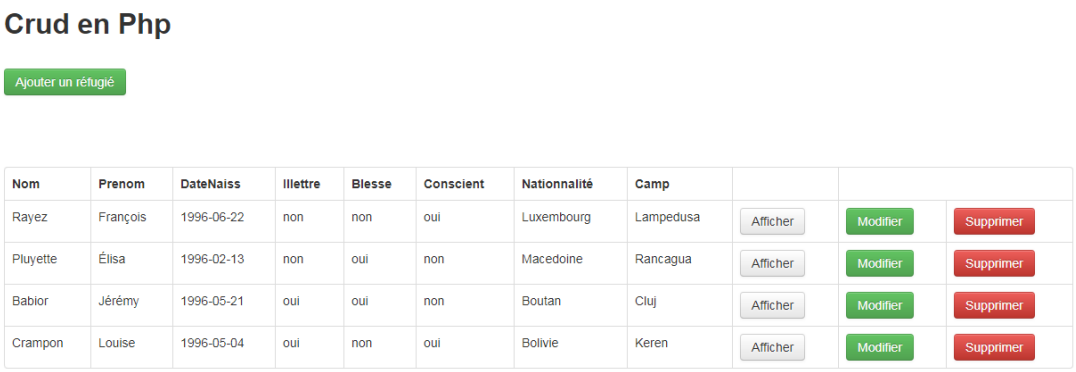
```
                        echo '

<td>';
                        echo '<a class="btn" href="form_read_refugies.php?id=' . $row['IdRefugies'] . '">Afficher</a>';
                        echo '</td>
<p>
';
                        echo '

<td>';
                        echo '<a class="btn btn-success" href="form_modify_refugies.php?id=' . $row['IdRefugies'] . '">Modifier</a>';
                        echo '</td>
<p>
';
                        echo'

<td>';
                        echo '<a class="btn btn-danger" href="file_delete_refugies.php?id=' . $row['IdRefugies'] . ' ">Supprimer</a>';
                        echo '</td>
<p>
';
                        echo '</tr>
<p>
';
}
                                Database::disconnect(); //on se deconnecte de la base

                    ?>
</tbody>
<p>
```

Une fois nos cases remplies, il faut ajouter les boutons sans quoi le « CRUD » perd tout son intérêt.

Le $row[‘IdRefugie’] sert à récupérer l’enregistrement choisit pour son affichage, sa modification ou encore sa suppression.

Nous allons donc créer trois boutons : Afficher , Modifier et Supprimer. Vous n’oublierez juste pas après le « href= » de rentrer le nom du fichier «. php » (bien que comme depuis le début, je vous conseille vivement de garder une syntaxe similaire à la mienne pour que l’on s’y retrouve lors de l’intégration. Il ne vous restera plus, après ça, qu’à fermer les deux trois balises restantes :

```
</tbody>
<p>

</table>
<p>

</div>
<p>


</div>
<p>


</div>
<p>

    </body>
</html>
```



Le CRUD terminé (je n'ai pas les mêmes données que vous car j'ai fait beaucoup de test). Mais en gros, voilà à quoi cela devra ressembler.

5. L'AJOUT DE DONNÉES

A) Le Formulaire d'ajout.

Nous allons à présent nous pencher sur l'ajout de données. Pour se faire, créons un fichier form_add_refugies.php. Ce fichier n'est ni plus ni moins qu'un formulaire. Néanmoins nous allons voir certaines fonctionnalités un peu « tricky » mais qui rendront notre formulaire super.

```php
<!DOCTYPE html>
<html>

    <head>
        <meta charset="utf-8" />
        <title>Titre</title>
    </head>

    <body>

            <?php include 'database_connexion.php'; //on inclut notre fichier de connection $pdo = Database::connect();
            $pdo = Database::connect();
            $sql = 'SELECT * FROM nationalite';
            $rep = $pdo->query($sql);
            $sql2= 'SELECT * FROM camp';
            $res= $pdo->query($sql2);
            $i=1;?>
```

Comme depuis toute à l'heure, nous commençons à initialiser nos variables. On n'oublie pas l'include de notre fichier de connexion. Nous allons utiliser deux requêtes sql, l'une qui recherche toutes les nationalités et l'autre qui recherche tous les camps, cela nous servira pour nos jolies barres déroulantes plus bas ☺. Nous n'oublions pas non plus les variables servant à l'exécution des requêtes sql.

/ ! le $i n'est pas utilisé, j'ai oublié de le supprimer ☺

```
<form name="Ajouter d'un réfugié" action="send_data_add_refugies.php" method="post">
  Nom:<br/>
  <input type="text" name="Nom" value=""/>
<p>

Prenom:<br/>
<input type="text" name="Prenom" value=""/>
<p>

Date de naissance:<br/>
<input type="date" name="DateNaiss" value=""/>
<p>
```

Après l'initialisation de nos variables, nous allons créer notre formulaire. Et tout commence avec cette ligne :

<form name="Ajouter d'un réfugié" action="send_data_add_refugies.php" method="post">

Pour faire simple (car nous ne sommes même pas à la moitié de l'étude et on est déjà à la 13ème page …) : Vous donner un nom à votre formulaire, l'action servira à envoyer les données rentrées dans le formulaire à un autre fichier qui les traitera, on verra ça un peu plus bas. Enfin la « méthod » sert à savoir ce qu'il passera dans l'url lors de l'envoie des données. En clair ne vous prenez pas la tête et mettez « post » ^^.

Notre formulaire est « ouvert » nous allons donc y ajouter des champs. Je vais vous expliquer le premier champ, c'est la même chose pour les deux en dessous.

Nom :
 Ce n'est que le « titre » si on veut du champ, cela permet à préciser à l'utilisateur quelle est la donnée attendue.

<input type= « text » name = « Prenom » value=””/>

Le type sert à définir celui du champ, ici ce sera un champ où il sera possible de saisir un texte. Le nom c'est tout simplement le nom du champs et value sert à définir une donnée par défaut.

```
  Illettrisme ?:<br/>
    <select name="Illetre">
      <option value="1">Oui</option>
      <option value="0">Non</option>
  </select>
  <p>

  Blessé(e) ?:<br/>
  <select name="Blesse">
    <option value="1">Oui</option>
    <option value="0">Non</option>
</select>
<p>

  Conscient(e)?:<br/>
  <select name="Conscient">
    <option value="1">Oui</option>
    <option value="0">Non</option>
</select>
  <p>
```

Ici nous n'allons pas mettre des champs mais des listes déroulantes. Pour les champs Illetre, Blessé et Conscient ce sera la même chose : une liste déroulante contenant un « Oui » ou un « Non ».

Pour se faire nous allons ouvrir des balise select après avoir donné un « libellé » à notre champ. Nous avons deux options à la « question » Illetre, oui ou non, soit 1 ou 0 .
Nous allons donc ouvrir des balises option, l'une aura pour value 1 et l'autre 0. Si c'est 1 c'est Oui et sinon c'est Non. On ferme les balise et c'est tout.

```
            Nationalité:<br/>
            <select name="Nationalite">
            <?php
            foreach ($pdo->query($sql) as $row){
              echo ('<option value='.$row['IdNationalite'].'>'.$row['NomPays'].'</option><br/>');
            }?>

            </select>
          <p>

          Camp:<br/>
          <select name="Camp">
          <?php  foreach ($pdo->query($sql2) as $row){
            echo ('<option value='.$row['IdCamp'].'>'.$row['Ville'].'</option><br/>');
          }?>
        </select>
          <p>
          <input type="submit" value="Envoyer"/>
          <input type="reset" value="Annuler"/>
        </form>
    </body>
</html>
```

Comme au-dessus nous allons utiliser des listes déroulantes, sauf que cette fois nous utiliserons un foreach (oui parce que faut pas déconner, on ne va pas perdre du temps en écrivant X option pour les nationalités et les camps).

Nous allons donc parcourir la table nationalité puis echo Le nom du pays en fonction de son id. Faites attention à la syntaxe car nous utiliser du php et du html en même temps.

Pour le camp ce sera la même chose.

Enfin nous créerons deux boutons, l'un qui envoie le formulaire et le second qui annule la saisie.

Voilà ce à quoi vous devriez arriver si vous avez bien suivit.

B) L'envoie de donnée d'ajout. :

Création du fichier « send_data_add_refugies.php »

```php
<?php


include 'database_connexion.php';


$pdo = Database::connect();


$stmt = $pdo->prepare("INSERT INTO refugies (Nom, Prenom, DateNaiss, Illetre, Blesse, Conscient, IdNationalite, IdCamp) VALUES (:Nom, :Prenom, :DateNaiss, :Illetre, :Blesse, :Conscient, :IdNationalite , :IdCamp)");
$stmt->bindParam(':Nom', $nom);
$stmt->bindParam(':Prenom', $prenom);
$stmt->bindParam(':DateNaiss', $date_naiss);
$stmt->bindParam(':Illetre', $illetre);
$stmt->bindParam(':Blesse', $blesse);
$stmt->bindParam(':Conscient', $conscient);
$stmt->bindParam(':IdNationalite', $idnationalite);
$stmt->bindParam(':IdCamp', $idCamp);
```

Pour ne pas changer, on commence par faire la connexion à la base.
Nous allons ensuite créer une variable stmt qui contient la préparation de la requête d'insertion en base.
Concernant la requête, rien de bien compliqué, il n'y a qu'à écrire les divers champs. Après le values on mettre « : » avant chaque champ car il s'agit ici des valeurs venant du formulaire.

Ensuite, nous allons lier nos paramètres à un nom de variable spécifique. Cela nous sert à préparer la requête. La variable sera liée en tant que référence et ne sera évaluée qu'à l'appel de stmt->execute.

```php
$nom=$_POST['Nom'];
$prenom=$_POST['Prenom'];
$date_naiss=$_POST['DateNaiss'];
$illetre=$_POST['Illetre'];
$blesse=$_POST['Blesse'];
$conscient=$_POST['Conscient'];
$idnationalite=$_POST['Nationalite'];
$idCamp=$_POST['Camp'];
```

Ici c'est comme ce que nous avions vu en cours. Il s'agit de tableau associatif, qui permette de rentrer ce qui a été saisie dans le formulaire dans leurs variables respectives.

```php
try {
$stmt-> execute();
}catch(Exception $e){
  echo $e->getMessage;
}
echo"<div align='center'>";
echo"<font face='Verdana' size='3' >L'élément a bien été inséré !</font>";
echo"</div>";

?>
```

Enfin on finit par un petit try catch pour vérifier que tout se passe bien, auquel cas, stmt->execute() est appelée.

Il faudra surement améliorer ce fichier, par exemple une redirection sur la table récapitulative sera obligatoire.

6. LA MODIFICATION DE DONNÉES

A) Le formulaire de modification

```php
<?php require 'database_connexion.php';
$id = null;
if ( !empty($_GET['id']))
{
  $id = $_REQUEST['id'];
 }
 if ( null==$id )
 {
   header("Location: refugies_crud.php");
 }

                $pdo = Database::connect();
               $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
               $sql = "SELECT * FROM Refugies where IdRefugies = $id";
               $sql_natio="SELECT * FROM Nationalite";
               $sql_camp="SELECT * FROM Camp";
               $q = $pdo->prepare($sql);
               $q->execute(array($id));
               $data = $q->fetch(PDO::FETCH_ASSOC);
               $nom = $data['Nom'];
               $prenom = $data['Prenom'];
               $date_naiss = $data['DateNaiss'];
               $illetre = $data['Illetre'];
               $blesse = $data['Blesse'];
               $conscient = $data['Conscient'];
               $nom_natio = $data['idNationalite'];
               $ville_camp = $data['idCamp'];
               Database::disconnect();
        //  }

      ?>
```

Création du fichier form_modify_refugies.php

Comme pour le formulaire d'ajout, nous allons initialiser nos variables.

Au départ nous allons faire une petite vérification quant à l'id, si la variable $_GET['id'] n'est pas vide alors la variable $id récupérera la requête http du paramètre ['id'].

$pdo nous servant toujours à nous connecter à la base définira le mode d'erreur (il s'agit d'une gestion d'exception).

Nous ferons également trois requêtes sql, l'une pour récupérer les données d'un réfugié en fonction de son id (en effet, il s'agit ici de récupérer ces données dans l'objectif de les modifier), une autre pour récupérer les nationalités et l'autre pour les camps. Ces requêtes retourneront un tableau, nous ferons donc un foreach plus bas.

La variable $q prépare la requête de la variable « sql » et sert aussi à exécuter cette même requête. La variable $data nous permet de retourner un enregistrement de requête sous la forme d'un tableau associatif.

Ensuite nous n'aurons plus qu'à initialiser les variables correspondantes aux champs de la tables qui contiendront la variable $data avec la variable associée à chaque champs.

```
<!DOCTYPE html>
<html>
 <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
    <title>Mofication d'un Refugié</title>
     <link href="css/bootstrap.min.css" rel="stylesheet">
    <img
src="
A7" data-wp-preserve="%3Cscript%20src%3D%22js%2Fbootstrap.js%22%3E%3C%2Fscript%3E"
data-mce-resize="false" data-mce-placeholder="1" class="mce-object" width="20" height="20"
alt="<script>" title="<script>" />

 </head>
 <body>


<br />
<div class="container">

<br />
<div class="row">

<br />
<h3>Modifier un Réfugié</h3>
<p>

</div>
<p>

<br />
```

Comme souvent, nous allons commencer par une mise en page, rien de sorcier, quelques div class et un peu de bootstrap ☺ vous n'avez qu'à copier-coller ce code pour que l'on soit tous sur le même thème css.

```
<br />
<form method="post" action="send_data_modify_refugies.php?id=<?php echo $id ;?>">

<br />
<div class="control-group ">
                <label class="control-label">Nom</label>

<br />
<div class="controls">
                    <input name="Nom" type="text"  placeholder="Nom" value="<?php echo!empty($nom) ? $nom : ''; ?>">

</div>
<p>

</div>
<p>



<br />
<div class="control-group">
                <label class="control-label">Prenom:</label>

<br />
<div class="controls">
                    <input type="text" name="Prenom" value="<?php echo!empty($prenom) ? $prenom : ''; ?>">

</div>
<p>

</div>
<p>
```

Nous allons pouvoir commencer à créer notre formulaire. En soit cela ne change pas beaucoup du formulaire d'ajout, sauf qu'ici le but est de récupérer les données du réfugié que nous souhaitons modifier.

Pour se faire, nous allons faire un « input » de base SAUF que la value prendra en paramètre un if else (ne prenez pas peur à cause de la syntaxe, il ne s'agit que d'un « if/else » en écriture condensée).

Pour le nom nous allons donc récupérer la variable $nom si la donnée n'est pas vide sinon la textbox sera vide.

Nous utiliserons le même procédé avec les variables du prénom et de la date de naissance.

```
<br />
<div class="control-group">
                <label class="control-label">Date de Naissance:</label>

<br />
<div class="controls">
                    <input type="date" name="DateNaiss" value="<?php echo!empty($date_naiss) ? $date_naiss : ''; ?>">

</div>
<p>

</div>
<p>
```

```
<br />
<div class="control-group ">
                  <label class="control-label">Illetre</label>

<br />
<div class="controls">

                      <select name="Illetre">

                      <?php  if( $data['Illetre']==0){?>
                                <option value="0">Non</option>
                                <option value="1">Oui</option>
                      <?php  }else{ ?>
                          <option value="1">Oui</option>
                          <option value="0">Non</option>

                      <?php  } ?>



                      </select>

</div>
<p>

</div>
<p>
```

Pour la valeur de la donnée «illetre » nous ne ferons pas un input mais un « select » comme lors de la création du formulaire d'ajout. Nous allons faire un if else, dans la première partie nous allons vérifier que la valeur d'illetre est égale à 0, dans ce cas nous afficherons une liste déroulante qui proposera D'ABORD Non puis Oui. Si au contraire la valeur d'illetre est différente de 0 nous afficherons une liste déroulante qui proposera D'ABORD Oui puis Non.

Ainsi ce que nous verrons dans la liste déroulante sera la valeur du réfugié quant à son illetrisme.

Nous utiliserons ce même procédé pour les valeurs blesse et conscient.

```
<br />
<div class="control-group ">
                  <label class="control-label">Blesse</label>

<br />
<div class="controls">
  <select name="Blesse">

  <?php  if( $data['Blesse']==0){?>
            <option value="0">Non</option>
            <option value="1">Oui</option>
  <?php  }else{ ?>
      <option value="1">Oui</option>
      <option value="0">Non</option>

  <?php  } ?>



  </select>

</div>
<p>

</div>
<p>
```

```
<br />
<div class="control-group ">
                 <label class="control-label">Conscient</label>

                   <select name="Conscient">

                   <?php  if( $data['Conscient']==0){?>
                             <option value="0">Non</option>
                             <option value="1">Oui</option>
                   <?php  }else{ ?>
                       <option value="1">Oui</option>
                       <option value="0">Non</option>

                   <?php  } ?>



                   </select>

</div>
<p>
```

```
<br />
<div class="control-group ">
                  <label class="control-label">Nationalité:</label>

                    <select name="Nationalite">

                      <?php
                      foreach ($pdo->query($sql_natio) as $row)
                      {
                        if ($data['idNationalite']!=$row['IdNationalite'])
                        {
                          echo('<option value='.$row['IdNationalite'].'>'.$row['NomPays'].'</option>');
                        }else{
                          echo('<option value='.$row['IdNationalite'].' selected>'.$row['NomPays'].'</option>');
                        }
                      }
                      ?>
                    </select>

</div>
<p>

  <br />
  <div class="control-group ">
                  <label class="control-label">Camp:</label>

                    <select name="Camp">

                      <?php
                      foreach ($pdo->query($sql_camp) as $row)
                      {
                        if ($data['idCamp']!=$row['IdCamp'])
                        {
                          echo('<option value='.$row['IdCamp'].'>'.$row['Ville'].'</option>');
                        }else{
                          echo('<option value='.$row['IdCamp'].' selected>'.$row['Ville'].'</option>');
                        }
                      }
                      ?>
                    </select>

  </div>
  <p>


<p>
```

Nous arrivons à quelque chose d'un peu plus tricky, nous allons récupérer le nom de la ville du camp.

Nous allons donc parcourir notre table camp avec un foreach.

Avec une condition if nous vérifierons que la variable retournant la ligne du résultat de la requête « sql_camp » avec en paramètre ['idCamp'] (il s'agit ici de l'idCamp de la table réfugié – soit la clé secondaire) est différente de la ligne ['IdCamp'] (il s'agit ici de l'IdCamp de la table camp – soit la clé primaire).

Si c'est le cas on affiche une liste déroulant semblable à celle du formulaire d'ajout. Sinon on affichera la même liste déroulante MAIS on ajoutera le mot clé « selected » qui permettra de prés-sélectionné la variable utilisée par ce réfugié (en gros on récupère le camp que l'on avait rentré lors de l'ajout du réfugié que nous sommes en train de modifier).

Nous ferons la même chose pour la nationalité.

```
<br />
<div class="form-actions">
                    <input type="submit" class="btn btn-success" name="submit" value="Envoyer">
                    <a class="btn" href="refugies_crud.php">Retour</a>
</div>
<p>

          </form>
<p>



</div>
<p>


  </body>
</html>
```

Nous n'aurons plus qu'à ajouter des boutons « Envoyer » et « Retour ». On n'oubliera pas les redirection – d'ailleurs, ici, il faudrait ajouter une redirection quand on clique sur le bouton « envoyer ».

B) L'envoie de donnée de modification

```
<?php

require 'database_connexion.php';
$id = null;
if ( !empty($_GET['id']))
{
  $id = $_REQUEST['id'];
 }
 if ( null==$id )
 {
   header("Location: refugies_crud.php");
 }
//include 'database_connexion.php';


$pdo = Database::connect();


$stmt = $pdo->prepare("UPDATE  refugies SET Nom = :Nom, Prenom= :Prenom, DateNaiss=:DateNaiss, Illetre=:Illetre, Blesse=:Blesse, Conscient=:Conscient, IdNationalite=:IdNationalite, IdCamp=:IdCamp WHERE IdRefugies = $id");
```

N'HÉSITEZ PAS A CTRL + SCROLL POUR AGRANDIR CETTE CAPTURE.

Ici nous allons envoyer nos données comme lors de l'ajout. -> Création d'un fichier send_data_modify_refugies.php.

On commence par vérifier l'id (voir page 18)

Puis nous nous connectons à la base avec la variable $pdo.A l'instar du formulaire d'ajout, nous créons une variable $stmt qui sera utiliser pour préparer la requête sql « UPDATE » où l'on **N'OUBLIERA PAS LE WHERE** (Sinon tous les réfugiés seront modifiés 😯)

Le reste du code est pareil que celui du formulaire d'ajout (page 16 & 17).

Si vous avez bien suivit, vous arriverez à ce résultat.

7. L'AFFICHAGE DES DONNÉES

Création d'un fichier « form_read_refugies.php »

```php
<?php require('database_connexion.php'); //on appelle notre fichier de confi
$id = null;
if (!empty($_GET['id']))
{
  $id = $_REQUEST['id'];
}
if (null == $id)
{
   header("location:refugies_crud.php");
 } else
 { //on lance la connection et la requete
   $pdo = Database ::connect();
   $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION) .
    $sql = "SELECT * FROM Refugies where IdRefugies=$id";
    $q = $pdo->prepare($sql);
    $q->execute(array($id));
    $data = $q->fetch(PDO::FETCH_ASSOC);

    $pdo_camp =Database::connect();
    $pdo_natio=Database::connect();
    $camp_sql= 'SELECT * FROM camp';
    $rep_camp= $pdo->query($camp_sql);
    $nationalite_sql='SELECT * FROM nationalite';
    $rep_natio = $pdo->query($nationalite_sql);

    Database::disconnect();
}
```

On a presque fini, nous allons à présent faire en sorte que l'on puisse voir la « carte d'identité » d'un réfugié si l'on clique sur « afficher » depuis la table récapitulative.

Je ne représente plus la vérification de l'id ni les variables que j'ai expliquées un peu plus haut, maintenant c'est toujours plus ou moins la même chose que précédemment ☺.

```
<!DOCTYPE html>
<html lang="fr">
   <head>
      <meta charset="utf-8">
          <link href="css/bootstrap.min.css" rel="stylesheet">
      <img
src="
A7" data-wp-
preserve="%3Cscript%20src%3D%22js%2Fbootstrap.min.js%22%3E%3C%2Fscript%3E" data-mce-
resize="false" data-mce-placeholder="1" class="mce-object" width="20" height="20"
alt="<script>" title="<script>" />
   </head>

   <body>

<br />
<div class="container">


<br />
<div class="span10 offset1">

<br />
<div class="row">

<br />
<h3>Edition</h3>
<p>

</div>
<p>
```

Encore et toujours du bootstrap et de la mise en forme ☺

```
<br />
<div class="form-horizontal" >

<br />
<div class="control-group">
                        <label class="control-label">Nom:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php echo $data['Nom']; ?>
                            </label>
</div>
<p>

</div>
<p>


<br />
<div class="control-group">
                        <label class="control-label">Prenom:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php echo $data['Prenom']; ?>
                            </label>
</div>
<p>

</div>
<p>


<br />
<div class="control-group">
                        <label class="control-label">DateNaiss:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php echo $data['DateNaiss']; ?>
                            </label>
</div>
<p>
```

Pour afficher le nom, le prénom et la date de naissance du réfugié, nous allons utiliser quelques classes de bootstrap pour rendre cela plus joli, mais ce n'est pas ce qui nous intéresse le plus vu que selon le fait que l'on garde ce thème ou que l'on en choisisse un autre cela changera.

En revanche il vous faudra « echo » la donnée désirée grâce à la variable $data que vous ai expliquée plus haut.

```
</div>
<p>


<br />
<div class="control-group">
                        <label class="control-label">Illettré(e)?:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php
                                if ($data['Illetre']==1)
                                {
                                    echo('Oui');
                                }else{
                                  echo('Non');
                                }
                                ?>
                            </label>
</div>
<p>

</div>
<p>
```

Pour l'illettrisme, les potentielles blessures et la conscience de notre réfugié nous utiliserons notre amie la condition « if » ainsi nous vérifirons si la valeur de la donnée « illetre » est égale à 1 auquel cas nous afficherons Oui sinon on affichera Non.

```
<br />
<div class="control-group">
                        <label class="control-label">Blessé(e)?:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php
                                if ($data['Blesse']==1)
                                {
                                  echo ('Oui');
                                }else {
                                  echo ('Non');
                                }
                                   ?>
                            </label>
</div>
<p>

</div>
<p>


<br />
<div class="control-group">
                        <label class="control-label">Conscient(e)?:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php
                                if ($data['Conscient']==1)
                                {
                                  echo ('Oui');
                                }else{
                                  echo ('Non');
                                }
                                   ?>
                            </label>
</div>
<p>

</div>
<p>
```

```
<br />
<div class="control-group">
                        <label class="control-label">Nationalité:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                              <?php
                              foreach($pdo_natio->query($nationalite_sql) as $row_natio)
                              {
                                If ($row_natio['IdNationalite']==$data['idNationalite'])
                                {
                                  $nom_natio=$row_natio['NomPays'];
                                }
                              }
                              echo $nom_natio;
                              ?>

                            </label>
</div>
<p>

</div>
<p>


<br />
<div class="control-group">
                        <label class="control-label">Camp:</label>

<br />
<div class="controls">
                            <label class="checkbox">
                                <?php
                                foreach($pdo_camp->query($camp_sql) as $row_camp)
                                {
                                  If ($row_camp['IdCamp']==$data['idCamp'])
                                  {
                                    $ville_camp=$row_camp['Ville'];
                                  }
                                }
                                echo $ville_camp;
                                ?>

                            </label>
</div>
<p>

</div>
<p>
```

Pour l'affichage du camp et de la nationalité de notre réfugié, on ne change pas de méthode : boucle foreach + condition « if » , le code ci-dessus ne diffère pas de d'habitude.

Vous remarquerez néanmoins que pour les données « illetre », « blesse », « conscient », « camp » et « nationalite », nous n'utilisons pas de liste déroulante. En effet on ne cherche qu'à afficher les données qui concernent le réfugié sélectionné.

```
<br />
<div class="form-actions">
                        <a class="btn" href="refugies_crud.php">Retour</a>
</div>
<p>



</div>
<p>

</div>
<p>


</div>
<p>
<!-- /container -->
    </body>
</html>
```

Nous ajouterons enfin un simple bouton « retour »

8. LA SUPPRESSION DES DONNÉES

Ça y est on touche la fin (c'est pas trop tôt Julie, c'est quand même la 32ème page de ton étude …)

Du coup => Création du fichier « file_delete_refugies.php »

```
<?php require 'database_connexion.php';
$id=0;

if(!empty($_GET['id']))
{
  echo($_GET['id']);
  $id=$_REQUEST['id'];
}
if(!empty($_POST))
{ $id= $_POST['id'];
  $pdo=Database::connect();
  $pdo->setAttribute(PDO::ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
  $sql = "DELETE FROM refugies  WHERE IdRefugies = $id";
  $q = $pdo->prepare($sql);
  $q->execute(array($id));
  Database::disconnect();
  header("Location: refugies_crud.php");

}
```

Rien ne change de d'habitude ici SAUF la requête :

On utilise la requête DELETE FROM et on **N'OUBLIE PAS LE WHERE** ! (Sinon on dit bye bye à tous nos pauvres petits réfugiés ☹)

```
<!DOCTYPE html>
<html lang="fr">
<head>
   <meta charset="utf-8">
          <link href="css/bootstrap.min.css" rel="stylesheet">
   <img
src="
A7" data-wp-
preserve="%3Cscript%20src%3D%22js%2Fbootstrap.min.js%22%3E%3C%2Fscript%3E" data-mce-
resize="false" data-mce-placeholder="1" class="mce-object" width="20" height="20"
alt="<script>" title="<script>" />
</head>

<body>

<br />
<div class="container">


<br />
<div class="span10 offset1">

<br />
<div class="row">

<br />
<h3>Supprimer un réfugié</h3>
<p>

</div>
<p>
```

De la simple mise en page.

```
<br />
<form class="form-horizontal" action="file_delete_refugies.php" method="post">
                    <input type="hidden" name="id" value="<?php echo $id;?>"/>

Êtes-vous sûre de vouloir supprimer ce réfugié ?

<br />
<div class="form-actions">
                        <button type="submit" class="btn btn-danger">Oui</button>
                        <a class="btn" href="refugies_crud.php">Non</a>
</div>
<p>

                </form>
<p>
</div>
<p>


</div>
<p>
<!-- /container -->
  </body>
</html>
```

Nous allons créer un formulaire, l'action sera bien sûre sur ce même fichier vu que l'on fait notre delete ici.
L'input est particulier, en effet il sera de type « hidden » (ou caché) cela nous permet d'inclure des données sans que celles-ci puissent être supprimées ou vues lors de l'envoie. La valeur sera la variable $id puisque nous cherchons à l'influencer en la supprimant.

Nous demandons à l'utilisateur s'il est sûr de son choix. Enfin nous ajoutons des boutons « Oui » et «Non » avec la redirection qui va bien ☺.

9. FONCTIONNALITÉS A PRÉVOIR ET CONCLUSION

Cette étude bien que complète n'est pas parfaite, il faudra ajouter des vérifications au niveau des formulaires (bon format de date, bon nombre de caractère insérés etc etc). On devra aussi voir ensemble si ce thème vous va, sinon les personnes en charge du CSS se devront de nous trouver un thème bootstrap plus sympa mais dans tous les cas il faudra rebosser un peu le CSS pour que ce soit plus en accord avec le thème du site web. ☺

Merci encore une fois de bien lire cette étude et de venir me voir que si vous avez des bugs ou que vous souhaitez des précisions.