Compte Rendu de la Mission 4

Maëva Fluteau et Rémi Dubois

Objectifs : Réalisation des tâches suivantes : T1.4, T4.1

1. T1.4 Gestion des types de championnat : Cette tâche doit permettre la création, la modification et la suppression d'un type de championnat.
2. T4.1 Saisie de l'indisponibilité d'un arbitre : Cette tâche doit permettre la saisie des créneaux d'indisponibilité d'un arbitre. Prévoir un formulaire qui permet à l'arbitre connecté de saisir le fait qu'il est indisponible pour une date et une demi-journée donnée (matin ou après-midi). Cette saisie doit être accompagnée de l'envoi d'un mail au responsable.

T1.4 Gestion des types de championnat

La gestion des types de Championnat nécessite un travail en plusieurs étapes :

- L'affichage des types de championnats contenus dans la base de donnée
- La possibilité d'ajouter un type de championnat dans la base donnée
- La possibilité de modifier une des entrées
- La possibilité de supprimer définitivement une des entrées

L'affichage

Le résultat de ce travail se présente ainsi :

Notre but a été d'obtenir un affichage sous forme de tableau. Pour y parvenir, il nous fallait récupérer les données de la base de donnée "bdarbitre" et nous nous y sommes connecté en utilisant le langage php :

```
$connexion=new PDO('mysql:host=localhost;dbname=bdarbitres', 'root', '');
```

Nous avons ensuite fait une requête SQL pour n'afficher que les éléments qui nous intéressaient, à savoir le numéro de type de championnat ainsi que son nom

```
$resultat = $connexion->query('SELECT * FROM type_championnat ORDER BY
NUM_TYPE_CHAMPIONNAT');
```

Pour afficher le résultat de cette requête, nous avons utilisé la méthode fetch

```
<?php
while($ligne=$resultat->fetch()){
?>
```

Ainsi, la liste des enregistrements s'affichent dans le code du tableau ci-dessous, grâce à "echo

$ligne" qui va récupérer les informations correspondants au nom du champs voulu.

```
    <td><?php echo $ligne['NUM_TYPE_CHAMPIONNAT'];?></td>
    <td><?php echo $ligne['NOM_TYPE_CHAMPIONNAT'];?></td>
```

L'ajout

à la fin de la page, pour une meilleure navigation entre les différentes possibilités de gestions de type de championnat, nous avons ajouté un lien qui renvoie vers la page d'ajout de type de championnat.

```
<a href="/ppe/form/typechampionnat_ajouter.php">Ajouter un nouveau
championnat</a>
```

En cliquant dessus, on arrive sur une page qui se présente ainsi :

Pour que l'ajout de type de championnat à la base de donnée fonctionne, il faut ajouter la suite de codes suivants. Ici, il s'agit d'indiquer que ce qui sera contenu dans le tableau correspondra aux variables correspondantes. Donc tout numéro et nom entrés dans les deux champs prendront respectivement les valeurs de "numform" et "nomform"

```
<td><input type="text" name="numform" placeholder="Num"</td>
<td><input type="text" name="nomform" placeholder="Nom"</td>
```

Le bouton ajouter permet de valider la saisie des donnée.

```
<input type="submit" value="Ajouter">
```

Pour se faire, il sera lié avec le code ci-dessous, placé en début du fichier, qui lui envoie les données sur le fichier action

```
<form action="/ppe/action/typechampionnat_action_ajouter.php" method="post">
```

Ce code dans le fichier "typechampionnat*action*ajouter.php" sert à récupérer données contenues dans le formulaire et à les envoyer dans la base de donnée, d'où l'insertion dans la requête. Il va récupérer les données contenues dans les variables "numform" et "nomform" et va les insérer dans les champs NOM*TYPE*CHAMPIONNAT et NUM*TYPE*CHAMPIONNAT à travers les variables :Nom et :Num. L'echo est le message qui est affichée en même temps que la bonne exécution de l'action.

```
$resultat = $connexion->prepare ('insert into
type_championnat(NOM_TYPE_CHAMPIONNAT,NUM_TYPE_CHAMPIONNAT)values(:Nom,:Num)
');
$resultat->execute (array(
'Nom'=>$_POST['nomform'],
'Num'=>$_POST['numform'],));
echo 'le type de championnat a bien ete ajoute';
```

La modification

En revenant sur l'affichage du tableau initial, deux liens sont présent pour permettre d'effectuer la modifications et la suppression de ces données.

```
    <td><a
href="/ppe/form/typechampionnat_modifier.php?NUM_TYPE_CHAMPIONNAT=<?php echo
$ligne['NUM_TYPE_CHAMPIONNAT'];?>">Modifier</a></td>
```

Lorsque l'on clique sur “Modifier”, le lien renvoie vers la page du formulaire de modification, similaire à celui de l'ajout. La différence se trouve dans le code. Ici, on souhaite récupérer les données des informations à modifier qui se trouvent dans la base de donnée. Ce qui se trouve dans NUM*TYPE*CHAMPIONNAT et NOM*TYPE*CHAMPIONNAT doit apparaître dans les variables numform et nom form. Il n'y a plus à modifier l'information voulue.

```
<tr> <!-- champs à remplir -->
<td><input type="text" name="numform" value="<?php echo
$_GET['NUM_TYPE_CHAMPIONNAT'] ?>"</td>
<td><input type="text" name="nomform" value="<?php echo
$ligne['NOM_TYPE_CHAMPIONNAT'] ?>"</td>
</tr>
```

Là encore, le bouton modifier validera ces modification en renvoyant vers le fichier “typechampionnat*action*modifier.php”.

```
<form action="/ppe/action/typechampionnat_action_modifier.php"
method="post">
```

Comme pour l'ajout, les données qui se trouvent dans nomform et numform vont être “update”, “éditée” dans les tables de la base de données.

```
$resultat = $connexion->prepare ('update type_championnat set
NOM_TYPE_CHAMPIONNAT=:Nom WHERE NUM_TYPE_CHAMPIONNAT=:Num');
$resultat->execute (array(
'Nom'=>$_POST['nomform'],
'Num'=>$_POST['numform'],));
```

La supression

La suppression est quant à elle un peu différente. Le lien renvoie directement à l'action de suppression codée, aucun formulaire intermédiaire n'est nécessaire.

```
    <td><a
href="/ppe/action/typechampionnat_action_supprimer.php?NUM_TYPE_CHAMPIONNAT=
<?php echo $ligne['NUM_TYPE_CHAMPIONNAT'];?>">Supprimer</a></td>
```

On récupère Num qui est l'identifiant pour permettre sa suppression totale de la base de donnée dans la table TYPE*CHAMPIONNAT. Toutes les données liées aux “Num”, c'est à dire son nom, seront supprimées. C'est une requête préparée avec un DELETE. <code php> $resultat = $connexion→prepare ('DELETE FROM type*championnat WHERE NUM*TYPE*CHAMPIONNAT=:Num'); $resultat→execute (array('Num'⇒$*GET['NUM*TYPE_CHAMPIONNAT'],)); </code>

T4.1 Saisie de l'indisponibilité d'un arbitre

Le formulaire

Le travail se présente de cette façon :

Il s'agit donc d'un formulaire sous forme de tableau. Le Nom d'arbitre s'affiche sous la forme d'une liste déroulante qui récupère en réalité son numéro mais affichera le nom correspondant

```
<?php
echo '<select name="numarb">';
while($ligne = $resultat->fetch())
{
echo '<option
value="'.$ligne["NUM_ARBITRE"].'">'.$ligne["NOM_ARBITRE"].'</option>';
}
echo '</select>';
```

Lorsque l'on clic dans le champs de saisie pour la date, un calendrier apparait Il s'agit d'un java scripte que nous avons récupéré d'Internet et intégré dans notre code.

```
<link rel="stylesheet"
href="http://code.jquery.com/ui/1.10.2/themes/smoothness/jquery-ui.css" />
<script src="http://code.jquery.com/jquery-1.9.1.js"></script>
<script src="http://code.jquery.com/ui/1.10.2/jquery-ui.js"></script>
<link rel="stylesheet" href="/resources/demos/style.css" />
<script>
$(function() {
$( "#datepicker" ).datepicker();
});
</script>
<code php>
Dans le champs date, on inclue donc le javascript du calendrier dans le
datepicker qui est l'id de "date".
<code php>
<p>Date: <input type="text" id="datepicker" name="datepicker" /></p>
```

Dans le formulaire, on permet à l'utilisateur de préciser si son absence n'aura lieu que le matin, l'après midi, ou la journée entière. Pour cela, nous avons donné une valeur à “matin” et “après midi”. Ainsi, si le matin est coché, le nom de la demie journée d'absence correspondra à “am” et sa veleur à “1”, si c'est l'après midi, son nom est “pm” et sa valeur “2” et cocher les deux prendra la valeur “journée entière” que nous développerons plus tard.

```
<tr>
<td>Matin :</td>
<td><input type="checkbox" name="am" value="1"></td>
</tr>
<tr>
<td>Après-midi :</td>
<td><input type="checkbox" name="pm" value="2"></td>
```

```
</tr>
```

Une fois le bouton validé, l'action est renvoyé vers le fichier indispoarbitre*action*valider.php qui va lui aussi agir sur la base de donnée.

Dans un premier temps, nous avons dû mettre en place quelques ajustements.

Il a fallu préciser que le fait de cocher les deux demis journées prendraient pour valeur “3”, ce qui permet d'obtenir la possibilité d'entrer une absence pour la journée entière. On ajoute pour se faire un +2.

```
$codedj=0;
if (isset($_POST['am'])) $codedj=1;
if (isset($_POST['pm'])) $codedj=$codedj+2;
```

Nous avons aussi dû changer l'ordre des éléments de la date repris par “datepicker” dans la mesure où le calendrier est en format américain, ce qui incompatible avec celle de la base de donnée. Nous avons donc changés leur emplacement pour les faire correspondre. Pour finir, le calendrier sépare le jour, le mois et l'année par un “/” alors que la base de donnée le fait par un “-”.

```
$date=$_POST['datepicker'];
$annee = substr($date, 6, 4);
 $jour = substr($date, 3, 2);
 $mois = substr($date, 0, 2);
 $date1 = $annee . '-' . $mois . '-' . $jour;
```

Pour finir, on envoie toutes ces informations sur la base des données via “l'insert into”. Chaque valeur est transformée pour pour correspondre à celle de la base de donnée

```
$resultat = $connexion->prepare ('insert into indisponibilite
(NUM_ARBITRE,DATE_JOUR,CODE_DEMI_JOURNEE) values (:Num,:Date,:CodeDemiJ)');
$resultat->execute (array(
'Num'=>$_POST['numarb'],
'Date'=>$date1,
'CodeDemiJ'=>$codedj));
```

Pour ce qui concerne l'envoie d'email à chaque nouvelle absence enregistrée, nous n'avons pas pu réaliser ce travail dans la mesure où nous n'avions pas de compte SMTP d’envoi.