

DESCRIPTIF DES MISSIONS REALISEES

Application de gestion des oraux de langue

BTS SIO option SLAM

Gwendoline Bassette Chancereul

Du 21 mars au 7 octobre 2019

Table des matières

I-	PRESENTATION DU PPE	2
II-	CONTEXTE	2
1.	CONTEXTE GENERAL	2
2.	PROCESSUS DE GESTION DES ORAUX DE LANGUE	2
III-	MEA	6
IV-	MODELE PHYSIQUE DES DONNEES	7
V-	DESCRIPTIF DETAILLE DES ACTIVITES REALISEES PENDANT LE PPE.....	8
1.	GESTION DES PROFESSEURS	8
1.1.	<i>Consultation des informations des professeurs</i>	<i>8</i>
1.2.	<i>Ajout des informations d'un professeur</i>	<i>9</i>
1.3.	<i>Modification des informations d'un professeurs.....</i>	<i>15</i>
1.4.	<i>Suppression des informations d'un professeur</i>	<i>18</i>
2.	MODIFICATION DES AFFECTATIONS PAR L'ADMINISTRATEUR	21

I- Présentation du PPE

Conformément à la RGPD les données ont été anonymisées.

Équipe : 15 personnes

Logiciels :

- Atom
- Visual Studio Code
- Php My Admin
- MySql
- WampServer

Langages :

- Php
- SQL
- Javascript
- Ajax
- CSS

Projet réalisé du 21 mars 2019 au 7 octobre 2019

II- Contexte

1. Contexte général

Madame R., professeur d'espagnol, organise les oraux de langues des élèves de terminales. Ces oraux sont des oraux blancs ou les oraux du BAC.

Il y a deux oraux par élève : celui de **LV1** et celui de **LV2**.

Pour réaliser le planning des oraux, elle reçoit un fichier Excel par le secrétariat élève, qui contient tous les élèves de terminale en filière générale.

Actuellement, Mme R. doit se renseigner auprès des professeurs pour connaître les créneaux auxquels ils sont disponibles pour faire passer les oraux. Puis, elle doit s'organiser avec le fichier Excel, en ajoutant des onglets, des colonnes, répartir les élèves en fonction des créneaux horaires et des langues, etc. de manière à organiser les oraux. En plus de gérer les dates et horaires de passage de chacun des élèves, elle doit réaliser les convocations des élèves, les fiches d'émargements et le planning des professeurs.

Mme R. souhaiterait ainsi une application permettant la gestion des oraux de langue.

2. Processus de gestion des oraux de langue

À la suite d'une réunion avec Mme R., nous avons eu connaissance du cahier des charges.

Un oral de langue dure 15 minutes ou 18 minutes pour les élèves disposant d'un tiers temps.

Les élèves de série L ne sont pas concernés par les oraux que ce soit la LV1 ou la LV2. Ils ne seront donc pas à prendre en compte lors de l'extraction du fichier Excel.

Les élèves en section internationale ne sont pas concernés par l'oral de LV1 mais sont concernés par l'oral de LV2.

Quatre élèves sont convoqués chaque heure. Par exemple quatre élèves sont convoqués à 8 heures, quatre à 9 heures etc. Le tiers-temps d'un élève n'a aucune incidence sur les horaires des convocations ou le nombre d'élèves à interroger sur l'heure. Le professeur doit juste savoir que l'élève concerné a un tiers temps.

Les professeurs ont une pause tous les 8 élèves.

Le planning pour une journée d'oraux est le suivant :

Horaires
8h-9h (4 élèves)
9h-10h (4 élèves)
Pause
10h30-11h (2 élèves)
11h-12h (4 élèves)
Pause
13h30-14h (4 élèves)
14h-15h (4 élèves)
Pause
15h30-16h (2 élèves)
16h-17h (4 élèves)

Un professeur n'enseigne qu'une seule langue et n'interroge que dans la matière qu'il enseigne.

Il peut y avoir des élèves qui choisissent une LV1 ou LV2 qui n'est pas enseignée au lycée, l'élève sera quand même enregistré dans la base de données, mais ne passera au lycée que la langue qui y est enseigné.

Le fichier envoyé par l'administration contient la série de l'élève (S, ES, L), la division, le titre (Mme, M.), le nom et le prénom de l'élève, sa date de naissance, s'il a le droit à un tiers temps, la section de langue (Européenne, internationale) et pour la LV1 et la LV2, la position (inscrit ou bénéficiaire, c'est à dire s'il passe cette épreuve ou non en cas de redoublement), le choix de la langue vivante, et s'il a ou non effectué une demande de dérogation.

Madame R. définit la période des oraux (par exemple du 10 au 15 juin) et les salles attribuées de certains professeurs, ainsi que les salles où il est possible d'interroger.

Actuellement, elle envoie un mail aux professeurs qui interrogent pour qu'ils choisissent les dates et horaires, leur salle etc.

Chaque professeur sélectionne au minimum une demi-journée pour faire passer des oraux. (Exemple : lundi matin, mardi après-midi). Ce choix est définitif pour la demi-journée.

Chaque professeur donne le nombre maximum d'élèves qu'il souhaite ou peut faire passer par jour. Pour gérer cela il faudra calculer automatiquement le nombre d'élèves moyen par professeur et mettre un avertissement lors du choix des élèves (mais pas le bloquer) (par exemple : vous avez sélectionné 10 élèves sur une moyenne de 45 par professeur.)

Les professeurs qui n'ont pas de salle attitrée choisissent leur salle. Ce vœu est définitif pour la demi-journée, c'est à dire qu'un professeur ne peut pas changer de salle d'une heure à une autre.

Les professeurs ayant une salle attitrée n'ont pas à choisir de salle pour faire passer les oraux, mais Mme R. peut changer cette salle. C'est elle qui fera le choix définitif.

Le professeur choisit les élèves qu'il souhaite interroger. Les professeurs qui n'auront pas saisi leurs demi-journées, salles, ou sélectionné d'élèves devront recevoir un avertissement, par exemple un mail d'information.

Dans l'application, il faudra ainsi réaliser un formulaire. Sur la droite un compteur avec le nombre d'élèves que le professeur interroge (LV1 et LV2 séparés) devra être affiché avec éventuellement le nombre d'élèves à interroger. Il faut aussi indiquer le nombre d'élèves TOTAL sélectionnés (donc par tous les professeurs).

Différentes consultations devront être possibles :

- La liste de tous les élèves qui doivent passer les oraux (LV1/LV2) pour le professeur connecté mais seulement dans la langue qu'il enseigne (ou deux consultations : une pour la LV1 et l'autre pour la LV2)
- La liste des élèves déjà affectés à ce professeur en mettant les élèves déjà sélectionnés dans une autre couleur par exemple
- Le nombre d'élèves devant passer les oraux par langue et par niveaux
- Le nombre d'élèves en attente d'affectation

L'édition du planning doit prendre en compte les souhaits des professeurs (salle, élèves, jours, horaires). Les affectations se font par classe et par élève, avec possibilité de modifications sur le planning.

Un PDF avec toutes les convocations devra être généré. Un bouton devra permettre de ne choisir qu'une seule convocation

En plus des convocations il faut éditer une liste d'émargement pour chaque professeur avec le nom de tous les élèves qu'il doit faire passer. Ainsi qu'un planning avec les professeurs par salle et par horaires.

Si un élève est absent il faut pouvoir générer une nouvelle convocation.

Un formulaire d'ajout, modification, suppression devra être prévu pour les informations des élèves et des professeurs.

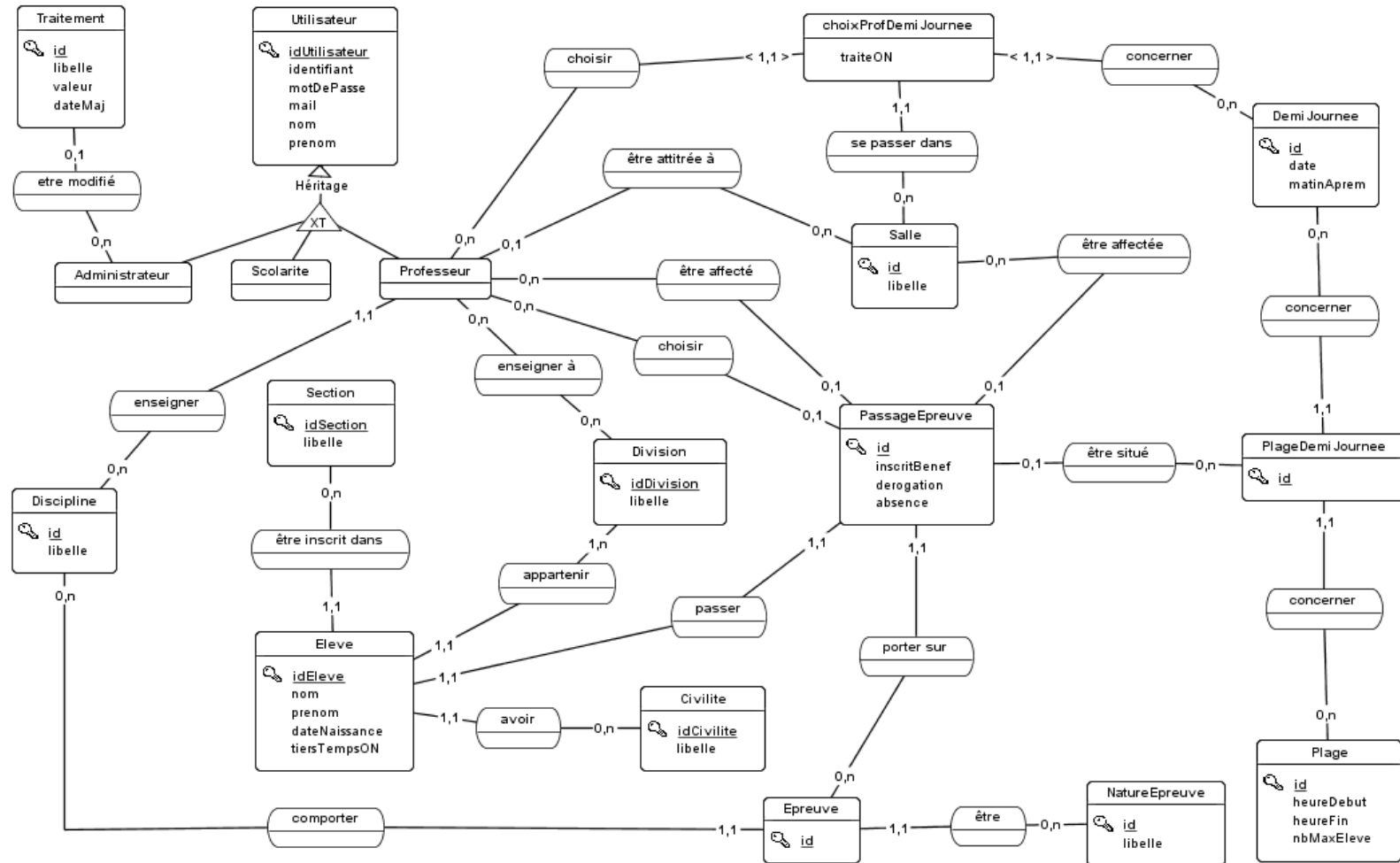
L'administration doit pouvoir faire des consultations. Ainsi, il faut qu'elle puisse consulter la convocation d'un élève avec la salle, le professeur et l'horaire et les convocations d'une classe.

Il faudrait aussi que le professeur puisse noter en temps réel les élèves absents (Afin de pouvoir récupérer tous les absents pour le professeur et l'administration (liste par ordre alphabétique et liste par demi-journée)

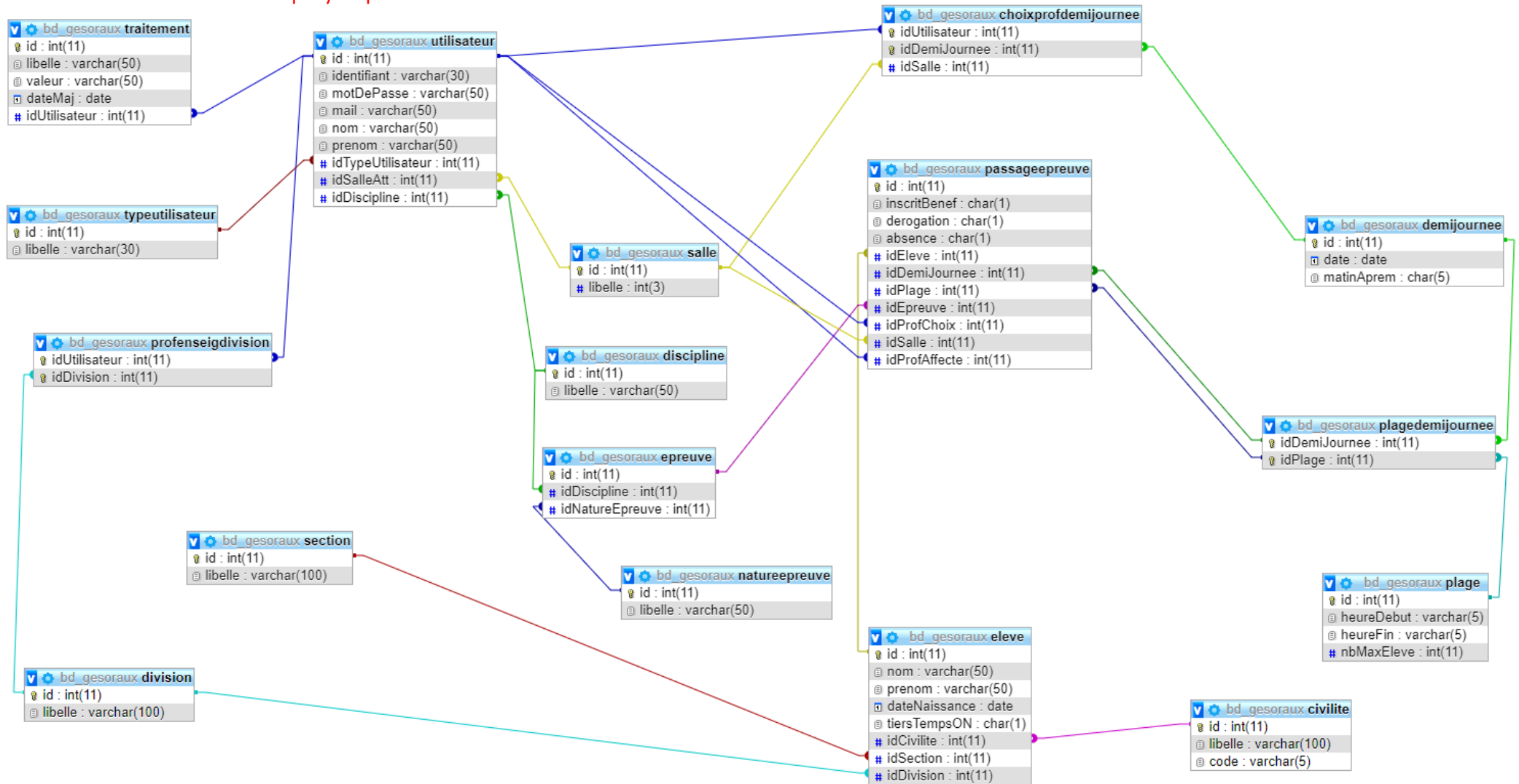
Les comptes utilisateur doivent être créés automatiquement. Les identifiants sont prénom.nom.

L'année suivante on écrase complètement les données. C'est Mme R. qui décide de lancer la suppression des données.

III- MEA



IV- Modèle physique des données

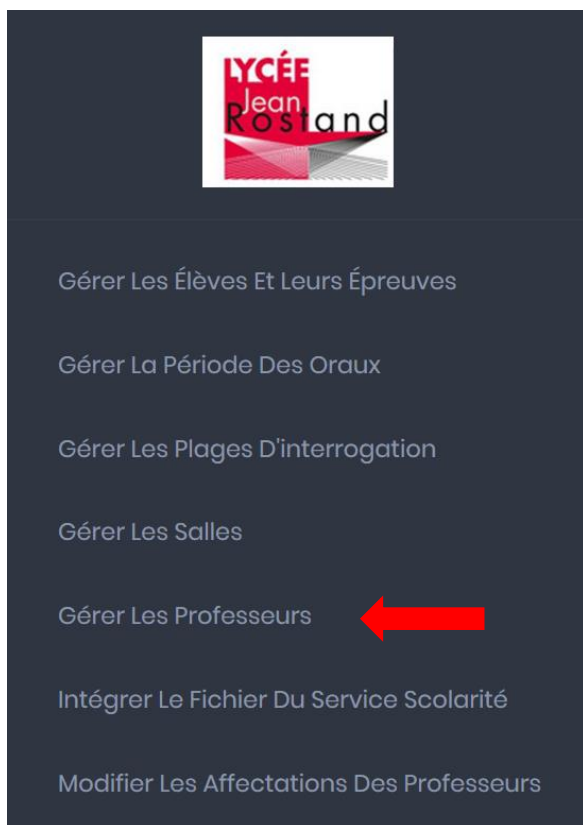


V- Descriptif détaillé des activités réalisées pendant le PPE

1. Gestion des professeurs

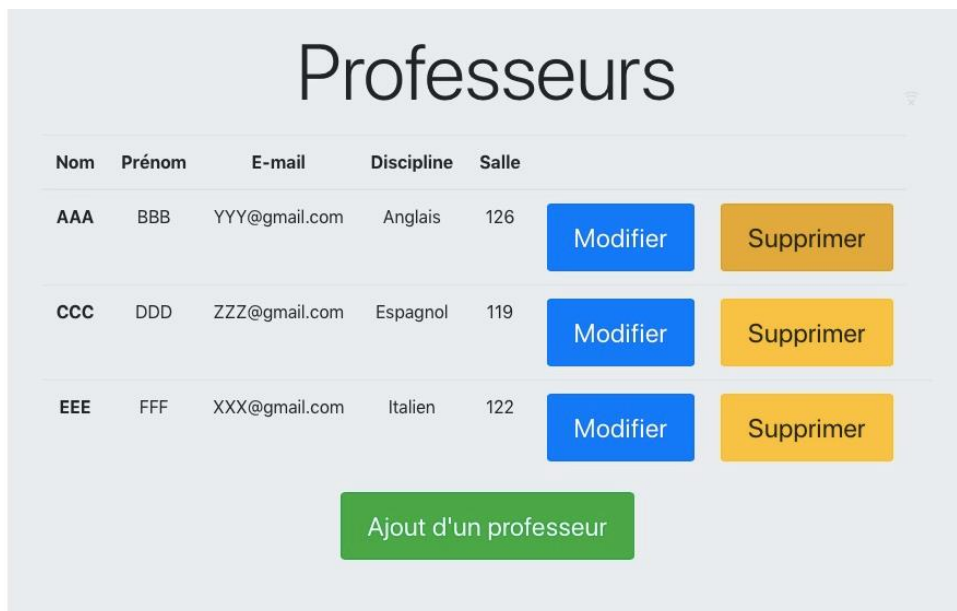
Afin de pouvoir gérer les informations des professeurs un formulaire doit permettre leur ajout, modification et suppression. Cette fonctionnalité n'est accessible qu'aux administrateurs.

Pour accéder à cette fonctionnalité l'utilisateur doit cliquer sur l'onglet « Gérer les professeurs » du menu, il obtiendra alors l'écran de consultation des professeurs.



1.1. Consultation des informations des professeurs

L'administrateur doit pouvoir consulter les informations de tous les professeurs classés par ordre alphabétique. On doit y retrouver le nom du professeur, son prénom, son adresse mail, la discipline qu'il enseigne et sa salle attitrée s'il en a une. Voici la maquette ayant été présentée à Mme R. pour cette fonctionnalité :





Les boutons « Ajouter un professeur », « Modifier », et « Supprimer » permettent de gérer les informations concernant les professeurs.

Pour obtenir ces informations j’ai effectué la requête SQL suivante :

```
SELECT utilisateur.id as 'idUtilisateur', nom, prenom, mail, discipline.libelle as 'discipline', salle.libelle as 'salle'
FROM utilisateur
LEFT OUTER JOIN discipline on idDiscipline = discipline.id
LEFT OUTER JOIN salle on idSalleAtt = salle.id
WHERE utilisateur.idTypeUtilisateur = 2
ORDER BY nom
```

Voici la fonctionnalité une fois réalisée :



 admin admin

Consultation des professeurs						
<div>Ajouter un professeur</div>						
Nom	Prénom	E-mail	Discipline	Salle	Modifier	Supprimer
██████	Sabine	██████@gmail.com	Allemand	109	<div>Modifier</div>	<div>Supprimer</div>
██████	Caroline	██████@gmail.com	Anglais	127	<div>Modifier</div>	<div>Supprimer</div>
██████	Benjamin	██████@gmail.com	Anglais	119	<div>Modifier</div>	<div>Supprimer</div>
██████	Laetitia	██████@gmail.com	Anglais	121	<div>Modifier</div>	<div>Supprimer</div>

1.2. Ajout des informations d’un professeur

L’administrateur doit pouvoir ajouter un professeur dans la base de données. Pour cela il a fallu réaliser un formulaire dont voici la maquette :

Ajout d'un professeur

Nom

Veillez saisir un nom

Prénom

Veillez saisir un prénom

E-mail

Veillez saisir l'e-mail

Discipline

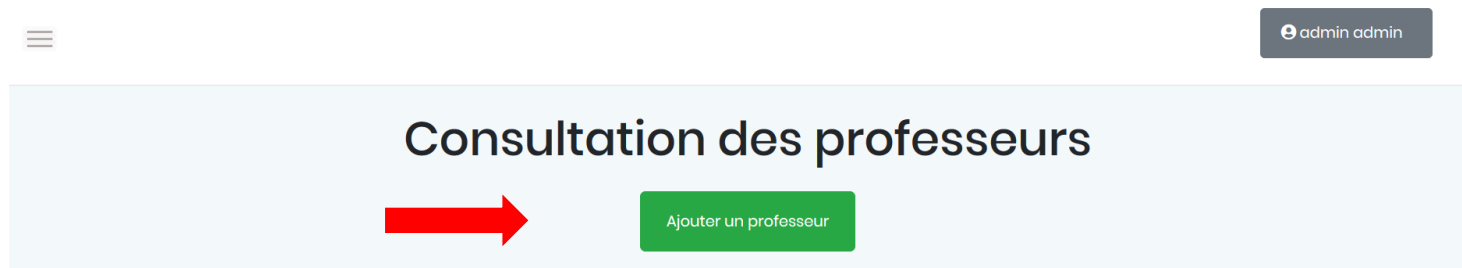
Anglais

Salle attitrée

126

Valider

Pour ajouter les informations d'un professeur qui n'existe pas encore dans la base de données, l'utilisateur doit cliquer sur le bouton « Ajouter un professeur » :



Lorsque l'utilisateur clique sur ce bouton, il obtient le formulaire suivant :

A screenshot of a web application interface showing a form titled 'Ajout d'un professeur'. The form is contained within a light blue box. At the top left of the page is a hamburger menu icon, and at the top right is a user profile button labeled 'admin admin'. The form fields are: 'Civilité' with radio buttons for 'Monsieur' and 'Madame' (selected); 'Nom' with a text input field containing the placeholder 'Saisissez le nom'; 'Prénom' with a text input field containing the placeholder 'Saisissez le prénom'; 'Mail' with a text input field containing the placeholder 'Saisissez le mail'; 'Discipline' with a dropdown menu showing 'Anglais'; and 'Salle Attitrée' with a dropdown menu showing 'Aucune'. At the bottom of the form is a green button labeled 'Valider'.

Pour réaliser ce formulaire, j'ai créé un fichier `admin_gestion_prof_comp_graph.php` qui a pour seul contenu le formulaire. De cette façon le formulaire peut être réutilisé lors de la modification des informations d'un professeur.

Le choix de la civilité se fait avec un radio bouton dont voici les instructions :

```
<label class="col-md-4" for="mr"><b>Civilité</b></label>
<div class="col-md-12">
    <?php
    include "connexion_bd_gesoraux.php";
    try{
        $lesEnregs=$bdd->query("SELECT id,libelle from civilite");
    } catch(PDOException $e) {
        die("ErrSeleCiv : erreur lors de la sélection des civilités dans admin_gestion_eleves_composant_graph.php<br>
        Message d'erreur : ".$e->getMessage());
    }
    if($lesEnregs->rowCount()>0) {
        foreach ($lesEnregs as $enreg) {
            echo"<div class='form-check form-check-inline'>";
            if ($rbt_civilite == $enreg->id) {
                echo "<input class='form-check-input' type='radio' checked name='rbt_civilite' id='$enreg->id' value='$enreg->id' />
                $enreg->libelle";
            }else{
                echo "<input class='form-check-input' type='radio' name='rbt_civilite' id='$enreg->id' value='$enreg->id' />
                $enreg->libelle";
            }
        }
    }
}
```

On effectue tout d'abord une requête dans la base de données pour obtenir toutes les civilités, si la requête ne peut pas être exécutée correctement on affiche un message d'erreur.

Si des lignes sont retournées par la requête SQL alors on crée les radio boutons. L'id du radio bouton sera l'id de la civilité récupérée par la requête et le texte affiché sera le libellé de la civilité. Le radio bouton est sélectionné si on modifie les informations d'un professeur.

Le nom, le prénom et le mail sont saisis dans des textbox.

La discipline est sélectionnée dans une liste déroulante dont voici les instructions :

```
<select class="custom-select custom-select" required name="lst_discipline" id="discipline">

    <?php
    include "connexion_bd_gesoraux.php";
    //exécution de la requête (avec la méthode query) pour obtenir le contenu de la table
    // on récupère le résultat de la requête dans le tableau $lesenregs
    try{
        $lesEnregs=$bdd->query("SELECT id, libelle from discipline");
    }catch(PDOException $e) {
        echo("err BDSelect : erreur de lecture table fonction dans admin_gestion_prof_comp_graph.php<br>
        Message d'erreur : ".$e->getMessage());
    }
    //on teste si le select a retourné des enregistrement
    if($lesEnregs->rowCount() > 0)
    {
        //pour chaque enregistrement retourné par la requête SQL, on crée une option dans la liste
        //l'attribut value contiendra l'id (l'identifiant de la discipline)
        // et le libellé de la discipline sera affiché

        foreach($lesEnregs as $enreg) {
            if($lst_discipline == $enreg->id){
                echo "<option class='form-group' selected value='$enreg->id'>$enreg->libelle</option>";
            }else{
                echo"<option class='form-group' value='$enreg->id'>$enreg->libelle</option>";
            }
        }
    }
    ?>
</select>
```

On effectue une requête dans la base de données pour obtenir toutes les disciplines et on affiche un message d'erreur si la requête ne peut être effectuée.

Puis pour chaque ligne retournée on crée une option dans la liste déroulante dont la valeur est l'id de la discipline et le libellé de la discipline est affiché. Si on modifie les informations d'un professeur alors on sélectionne la discipline du professeur.

La salle attribuée du professeur est sélectionnée dans une liste déroulante. La méthode est exactement la même que pour la liste des disciplines, on récupère toutes les salles présentes dans la base de données puis on crée des options pour chacune des salles.

Lorsque l'utilisateur clique sur le bouton valider à la fin du formulaire, les instructions qui suivent sont exécutées.

```
//si le tableau $_POST contient le bouton valider
//alors cela signifie que le formulaire à été soumis
if(isset($_POST['btn_valider'])==true){
    //appel de la fonction extract qui crée automatiquement les variables
    //dont les noms sont les index de $_POST
    // et leurs affecte la valeur associé
    extract($_POST);
    //si le nom n'existe pas dans le tableau $_POST
    //ou s'il n'est pas renseigné : on ajoute un message d'erreur
    if(isset($txt_nom)==false || trim($txt_nom)==""){
        $msg=$msg."Le nom est obligatoire<br>";
    }
    if(isset($txt_prenom)==false || trim($txt_prenom)==""){
        $msg=$msg."Le prénom est obligatoire";
    }
    if(isset($txt_mail)==false || trim($txt_mail)==""){
        $msg=$msg."<br> Le mail est obligatoire";
    }
    if(isset($lst_discipline)==false || trim($lst_discipline)==""){
        $msg=$msg."<br> La discipline est obligatoire";
    }
    if(isset($rbt_civilite)==false || trim($rbt_civilite)==""){
        $msg=$msg."<br> La civilité est obligatoire";
    }
}
```

On fait un extract de \$_POST afin de récupérer les informations saisies par l'utilisateur dans le formulaire. On vérifie ensuite pour la civilité, le nom, le prénom, le mail et la discipline, qu'ils ont été renseignés. Si ce n'est pas le cas on affiche un message d'erreur de cette façon :

Le nom est obligatoire
Le prénom est obligatoire
Le mail est obligatoire
La civilité est obligatoire



admin admin

Ajout d'un professeur

Civilité

☐ Monsieur ☐ Madame ☐ Autre

Nom

S'il n'y a pas de message d'erreur alors on va pouvoir ajouter l'enregistrement. Les professeurs sont des utilisateurs de l'application, il faut donc leur créer un identifiant et un mot de passe qui seront enregistrés dans la table utilisateur de la base de données avec les autres informations concernant le professeur.

Il a été demandé par Mme R. que l'identifiant soit comme celui de l'ENT, c'est-à-dire le prénom et le nom séparé par un point. Pour l'identifiant, le nom et le prénom doivent être en minuscule, on utilise donc la fonction `strtolower` qui permet de mettre en minuscule le nom et le prénom saisi, qu'ils soient totalement en majuscule, avec des majuscules et des minuscules ou totalement en minuscule. On crée ensuite une variable qui contiendra l'identifiant pour qu'on puisse par la suite l'utiliser lors de l'insert dans la base de données.

Aucune indication n'avait été donnée pour la création du mot de passe on a donc choisi que par défaut le mot de passe serait les deux premières lettres du prénom, l'année en cours puis les deux premières lettres du nom. Pour obtenir l'année en cours j'ai utilisé la fonction `date`. La fonction `substr($variable, 0,2)` permet de récupérer les deux premiers caractères de la variable. Le 0 indique à quel caractère on commence et le 2 le nombre de caractère à retourner. Les mots de passe ne peuvent pas être stockés en clair dans la base de données, il faut les crypter. Pour cela j'ai utilisé la fonction `sha1()`. Pour plus de sécurité il faudrait utiliser un grain de sable lorsqu'on crypte le mot de passe, je ne l'ai pas fait car l'étudiant qui a réalisé le script de connexion à l'application n'avait pas pris en compte le grain de sable.

L'utilisateur peut saisir le nom et le prénom du professeur tout en minuscule, tout en majuscule ou en mélangeant minuscules et majuscule. Pour que dans la base de données tous les noms et tous les prénom soit enregistrés sous le même format, j'ai utilisé la fonction `ucfirst()`. Cette fonction permet de mettre la première lettre de chaque mot en majuscule et toutes les autres en minuscule. De cette façon lors de l'affichage des noms des professeurs tous seront écrits de la même façon et cela facilite la saisie aux utilisateurs qui n'ont pas à faire attention aux majuscules ou aux minuscules.

Voici ces instructions :

```
//s'il n'y a pas d'erreur de saisie on va ajouter l'enregistrement
if($msg=="")
{
    //on passe le nom et le prénom en minuscule
    $nom_min = strtolower($txt_nom);
    $prenom_min = strtolower($txt_prenom);

    //on récupère l'année en cours
    $annee = date("Y");

    //on génère le compte composé du prénom suivi du nom
    $identifiant = $prenom_min." ".$nom_min;

    //on génère le mot de passe composé des 2 premiers caractères du nom
    //suivi de l'année en cours suivi de 2 premiers caractères du prénom
    $mot_de_passe_en_clair = substr($nom_min, 0,2).$annee.substr($prenom_min, 0,2);

    //on appelle la fonction sha1 pour crypter le mot de passe
    $mot_de_passe_crypte = sha1($mot_de_passe_en_clair);

    //Mettre la première lettre du prénom et du nom en majuscule
    $nom_prem_maj = ucfirst($txt_nom);
    $prenom_prem_maj = ucfirst($txt_prenom);
```

Il est maintenant possible d'effectuer la requête SQL permettant l'ajout du professeur dans la base de données. On prépare donc la requête SQL paramétrée pour plus de sécurité. L'identifiant est la variable identifiant créée précédemment, le mot de passe est la variable contenant le mot de passe crypté, le mail est récupéré lors de l'extract du \$_POST, le nom et le prénom sont les variables créés pour avoir le nom et le prénom avec la première lettre en majuscule, le typeUtilisateur est 2 ce qui correspond au typeUtilisateur professeur, la salle attirée sera nulle si aucune salle n'a été sélectionné dans la liste déroulant ou l'id de la salle si elle a été sélectionnée, la discipline sera l'id de la discipline sélectionnée dans la liste déroulante et la civilité sera l'id de la civilité cochée. On exécute ensuite la requête SQL.

```
// on prépare la requête insert
try {
    $req=$bdd->prepare("INSERT into utilisateur values(0, :par_ident,:par_mdp,:par_mail, :par_nom, :par_prenom,
    :par_typeUtili,:par_salleAtt, :par_discipline,:par_civilite)");
    $req->bindValue (':par_ident', $identifiant, PDO::PARAM_STR);
    $req->bindValue (':par_mdp',$mot_de_passe_crypte , PDO::PARAM_STR);
    $req->bindValue (':par_mail',$txt_mail, PDO::PARAM_STR);
    $req->bindValue (':par_nom', $nom_prem_maj, PDO::PARAM_STR);
    $req->bindValue (':par_prenom', $prenom_prem_maj, PDO::PARAM_STR);
    $req->bindValue (':par_typeUtili','2', PDO::PARAM_INT);
    if($1st_salle==0){
        $req->bindValue (':par_salleAtt', null, PDO::PARAM_INT);
    }
    else{
        $req->bindValue (':par_salleAtt', $1st_salle, PDO::PARAM_INT);
    }
    $req->bindValue (':par_discipline', $1st_discipline, PDO::PARAM_INT);
    $req->bindValue (':par_civilite', $rbt_civilite, PDO::PARAM_INT);
    $req->execute();
    $msg="Le professeur a bien été ajouté";

    //on se redirige vers l'affichage des employés en fournissant le message d'information
    header('Location: admin_gestion_prof_consultation.php?msg='.$msg);
}
```

Si la requête SQL s'exécute sans erreur alors la variable msg contiendra un message d'information pour l'utilisateur, et on redirige l'utilisateur sur la page de consultation des professeurs mise à jour sur laquelle le message s'affiche :

Consultation des professeurs

Ajouter un professeur

Le professeur a bien été ajouté

Nom	Prénom	E-mail	Discipline	Salle	Modifier	Supprimer
B	Sabine	@gmail.com	Allemand	109	Modifier	Supprimer

1.3. Modification des informations d'un professeur

Il doit être possible de modifier les informations d'un professeur. Le formulaire de modification est le même que le formulaire d'ajout mais les champs sont préremplis avec les informations du professeur en modification. Voici la maquette qui a été réalisée :

Modification des informations d'un professeur

Nom

Prénom

E-mail

Langue

Anglais

Salle Attitrée

126

Modifier

Pour modifier les informations d'un professeur il faut cliquer sur le bouton « Modifier » sur la ligne correspondant au professeur à modifier. Par exemple pour modifier les informations de Caroline, il faut cliquer sur le bouton « modifier » entouré en rouge :

admin admin

Consultation des professeurs

Ajouter un professeur

Nom	Prénom	E-mail	Discipline	Salle	Modifier	Supprimer
██████	Sabine	██████@gmail.com	Allemand	109	Modifier	Supprimer
██████	Caroline	██████@gmail.com	Anglais	127	Modifier	Supprimer
██████	Benjamin	██████@gmail.com	Anglais	119	Modifier	Supprimer
██████	Laetitia	██████@gmail.com	Anglais	121	Modifier	Supprimer

Lorsque l'utilisateur clique sur ce bouton, le script de modification est appelé et on passe l'id du professeur à modifier en paramètre. Voici l'instruction correspondant au bouton :

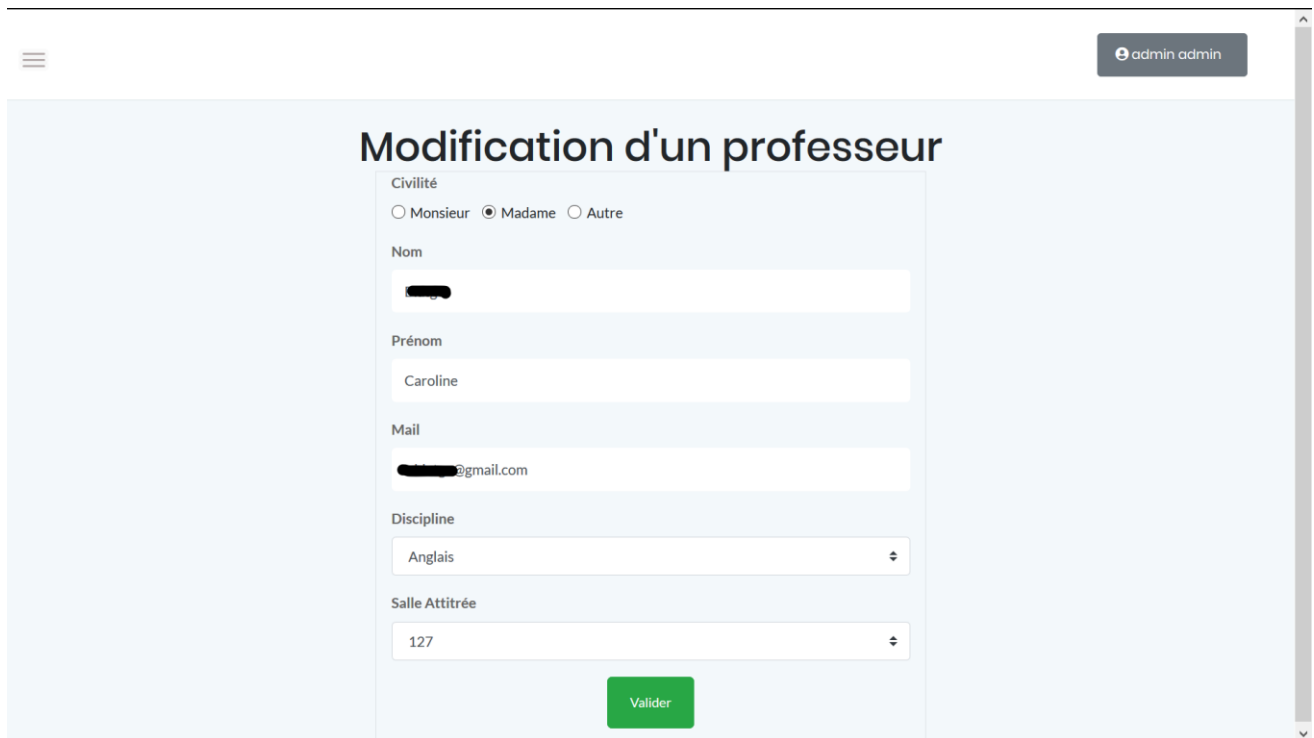
```
echo"<td> <a href='admin_gestion_prof_modif.php?id=$enreg->idUtilisateur'>  
<input class='btn btn-info' type='button' name='btn_modif_prof' value='Modifier'> </a> </td>";
```

De cette façon l'url est le suivant :

localhost/AppSlamOrauxV2-master/admin_gestion_prof_modif.php?id=2

Le script appelé est `admin_gestion_prof_modif.php` et l'id du professeur à modifier est 2.

On obtient alors la page suivante :



The screenshot shows a web interface for modifying a teacher's profile. At the top right, there is a user login bar showing 'admin admin'. The main heading is 'Modification d'un professeur'. Below this is a form with the following fields: 'Civilité' with radio buttons for 'Monsieur', 'Madame' (selected), and 'Autre'; 'Nom' with a text input field containing a redacted name; 'Prénom' with a text input field containing 'Caroline'; 'Mail' with a text input field containing a redacted email address followed by '@gmail.com'; 'Discipline' with a dropdown menu showing 'Anglais'; and 'Salle Attitrée' with a dropdown menu showing '127'. A green 'Valider' button is positioned below the form fields.

Les informations du professeur sont pré-remplies à partir des informations présentes dans la base de données.

Pour cela j'ai réalisé les instructions suivantes qui vérifient que la variable \$_GET['id'] existe puis qui récupère la valeur qu'elle contient dans \$id. Ensuite un select est réalisé dans la base de données pour obtenir le nom, le prénom, l'adresse mail du professeur dont l'id est \$id, ainsi que l'identifiant de la discipline qu'il enseigne, l'identifiant de sa salle attitrée si elle existe et l'identifiant de sa civilité. Si la requête a retourné des lignes alors on attribue aux éléments du formulaire les valeurs correspondantes retournées par le select, sinon on affiche un message à l'utilisateur pour l'avertir qu'il n'existe pas d'enregistrement avec cet identifiant. Si la requête n'a pas pu être exécutée un message d'erreur est affiché à l'utilisateur.

Voici ces instructions :

```
// le script est-il appelé lors du clic sur le bouton Modifier situé sur la page
//d'affichage des professeur (script gestion_prof_admin.php)
```

```
$msg="";
if(isset($_GET['id'])==true && $_GET['id']>0){
    // on récupère l'identifiant du professeur passé avec la méthode GET dans la variable $id
    $id=$_GET['id'];
    //connexion à la base de données
    include "connexion_bd_gesoraux.php";
    try{
        //on récupère les caractéristiques du professeur
        //dont l'identifiant est contenu dans la variable $id
        $lesEnregs=$bdd->query("SELECT nom,prenom,mail,idDiscipline,idSalleAtt,idCivillite FROM utilisateur where id=$id");
        if ($lesEnregs->rowCount() == 0) {
            echo "Aucun enregistrement";
        } else {
            $unEnreg=$lesEnregs->fetch();
            // on stocke les caractéristiques du professeur
            // dans les variables ayant le même nom
            //que les contrôles du formulaire.
            //par exemple : la variable $txt_nom contiendra le nom du professeur
            $txt_nom = $unEnreg->nom;
            $txt_prenom = $unEnreg->prenom;
            $txt_mail = $unEnreg->mail;
            $lst_discipline = $unEnreg->idDiscipline;
            $lst_salle = $unEnreg->idSalleAtt;
            $rbt_civillite =$unEnreg->idCivillite;
        }
    }catch(PDOException $e){
        echo("Err BDAlec01Erreur : erreur de SELECT dans admin_gestion_prof_modif.php <br>Message d'erreur : ".$e->getMessage());
    }
}
```

Lorsque l'utilisateur clique sur valider après avoir modifié les informations du professeur on vérifie que tous les champs sont remplis à l'exception de la salle qui peut être null. Dès que l'un des champs est vide, on ajoute un message dans la variable \$msg qu'on affichera à l'utilisateur. Ces instructions sont les suivantes :

```
//si le tableau $_POST contient le bouton valider
// alors cela signifie que le formulaire a été soumis
//le traitement de mise à jour va avoir lieu
if(isset($_POST['btn_valider'])==true){
    //on appelle la fonction extract qui crée automatiquement les variables
    //dont les noms sont les index de $_POST
    //et leur affecte la valeur associée
    extract($_POST);
    //si le nom n'existe pas dans le tableau $_POST//ou s'il n'est pas renseigné : on ajoute un message d'erreur
    if(isset($txt_nom)==false || trim ($txt_nom)==""){
        $msg=$msg."Le nom est obligatoire<br>";
    }
    if(isset($txt_prenom)==false|| trim ($txt_prenom)==""){
        $msg=$msg."Le prénom est obligatoire";
    }
    if(isset ($txt_mail)==false || trim ($txt_mail)==""){
        $msg=$msg."<br> Le mail est obligatoire";
    }
    if(isset ($lst_discipline)==false || trim ($lst_discipline)==""){
        $msg=$msg."<br> La discipline est obligatoire";
    }
}
```

S'il n'y a rien dans la variable \$msg alors on va essayer d'exécuter la requête SQL de mise à jour dans la base de données. On prépare d'abord la requête paramétrée pour plus de sécurité. Les paramètres sont les mêmes que lors de l'ajout d'un professeur à l'exception de l'identifiant et du mot de passe qui ne peuvent pas être modifiés par l'administrateur. On attribue à la variable \$msg un message pour informer l'utilisateur que la modification a bien été réalisée.

```
// s'il n'y a pas d'erreur on va modifier l'enregistrement
if($msg==""){
    //connexion à la base de données
    include "connexion_bd_gesoraux.php";
    try{
        //on réalise la requête de mise à jour (update)
        $req=$bdd->prepare("UPDATE utilisateur SET nom=:par_nom, prenom=:par_prenom, mail=:par_mail,
        idSalleAtt=:par_salle,idDiscipline=:par_discipline,
        idCivillite=:par_civillite where id=$id");
        //Mettre la première lettre du prénom et du nom en majuscule
        $nom_prem_maj = ucfirst($txt_nom);
        $prenom_prem_maj = ucfirst($txt_prenom);
        $req->bindValue (':par_nom', $nom_prem_maj, PDO::PARAM_STR);
        $req->bindValue (':par_prenom', $prenom_prem_maj, PDO::PARAM_STR);
        $req->bindValue (':par_mail', $txt_mail, PDO::PARAM_STR);

        if($lst_salle==0){
            $req->bindValue (':par_salle', null, PDO::PARAM_INT);
        }
        else{
            $req->bindValue (':par_salle', $lst_salle, PDO::PARAM_INT);
        }

        $req->bindValue (':par_discipline', $lst_discipline, PDO::PARAM_INT);
        $req->bindValue (':par_civillite', $rbt_civillite, PDO::PARAM_INT);

        $req->execute();
        //on indique dans la variable $msg que tout s'est bien passé
        $msg="la modification a bien été prise en compte";
    }
```

On se redirige ensuite vers la page de consultation des professeurs sur laquelle le message d'information est affiché. Si la modification n'a pas pu être réalisée, alors on affiche un message d'erreur. S'il y a un message d'erreur parce que tous les champs n'ont pas été saisis on l'affiche de la même manière que pour la modification

```
//on se redirige vers l'affichage des employés en fournissant le message d'information
header('Location: admin_gestion_prof_consultation.php?msg='.$msg);
} catch(PDOException $e){
    echo("Err BDALec01Erreur : erreur de modification dans admin_gestion_prof_modif.php<br>Message d'erreur : ".$e->getMessage());
}
}
else{
    echo("<div class='alert alert-danger'>");
    echo $msg;
    echo("</div>");
}
}
```

Voici comment s'affiche le message informant l'utilisateur de la réussite de la modification :

Consultation des professeurs

Ajouter un professeur

La modification a bien été prise en compte

Nom	Prénom	E-mail	Discipline	Salle	Modifier	Supprimer
████████	Sabine	████████@gmail.com	Allemand	109	Modifier	Supprimer

1.4. Suppression des informations d'un professeur

Il est possible de supprimer les informations d'un professeur. Voici la maquette de la suppression présentée à Mme R. :

Suppression des informations d'un professeur

Nom : XXX
 Prénom : YYY
 E-mail : XXX@gmail.com
 Professeur d'Anglais
 Salle attitrée : 128

Supprimer

Lorsque l'utilisateur clique sur « supprimer », il devrait obtenir cette page récapitulant les informations du professeur qui va être supprimé puis le bouton de suppression à la fin.

Pour réaliser cette suppression, l'administrateur doit cliquer sur le bouton « supprimer » sur la ligne du professeur dont on souhaite supprimer les informations. Par exemple pour Caroline :

Consultation des professeurs						
Ajouter un professeur						
Nom	Prénom	E-mail	Discipline	Salle	Modifier	Supprimer
Berruyer	Sabine	s.berruyer@gmail.com	Allemand	109	Modifier	Supprimer
Blatge	Caroline	c.blatge@gmail.com	Anglais	127	Modifier	Supprimer

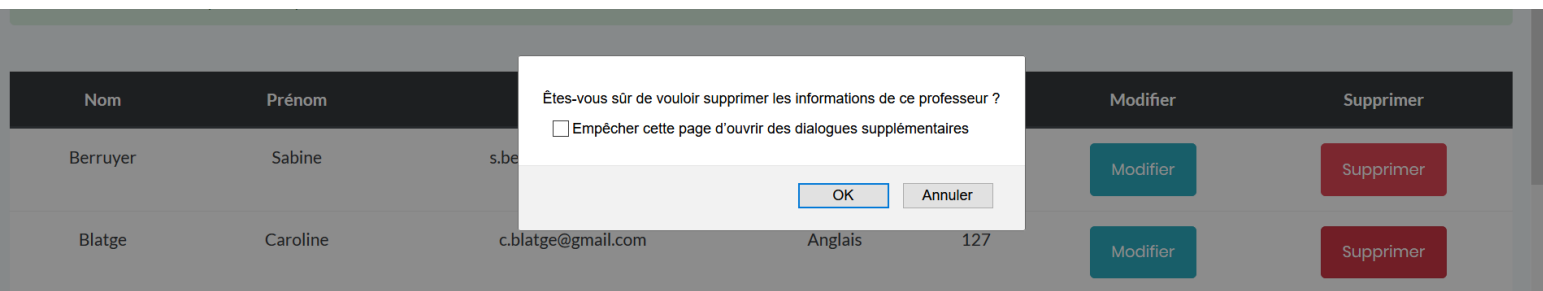
Voici l'instruction du bouton de suppression :

```
echo"<td> <a href='admin_gestion_prof_supp.php?id=$enreg->idUtilisateur' onclick='return confirmer_suppres();'>  
<input class='btn btn-danger' type='button' name='btn_supp_prof' value='Supprimer'> </a></td>";
```

Lorsqu'on clique sur le bouton de suppression, la méthode confirmer_suppres est appelée, elle demande la confirmation de l'utilisateur avant d'appeler le script de suppression en passant en paramètre l'id du professeur dont on souhaite supprimer les informations.

```
function confirmer_suppres()  
{  
    return(confirm('Êtes-vous sûr de vouloir supprimer les informations de ce professeur ?'));  
}
```

Voici comment s'affiche ce message :



Si l'utilisateur clique sur « annuler », la suppression n'aura pas lieu, sinon le script `admin_gestion_prof_supp.php` est appelé, voici ce qu'il contient :

```
//on initialise la variable $msg qui contiendra la liste des erreurs
$msg="";
//-----
//est ce que l'id du professeur à supprimer à été passé en GET ?
//-----
if(isset($_GET['id'])==true && $_GET['id']>0){
    //connexion à la base de données
    include "connexion_bd_gesoraux.php";
    $id=$_GET['id'];
    try{
        //-----
        //on supprime (requête delete) le professeur dont l'identifiant
        //est dans $_GET['id']
        //-----
        $id=$_GET['id'];
        $req=$bdd->prepare("DELETE FROM utilisateur where id=:par_id");
        $req->bindValue (':par_id', $id, PDO::PARAM_INT);
        $req->execute();
        //on indique dans la variable $msg que tout s'est bien passé
        //et on fait une redirection(header) vers gestion_prof_admin.php
        //en passant la variable $msg

        $msg="Le professeur a bien été supprimé";
        header('Location: admin_gestion_prof_consultation.php?msg='.$msg);
    }catch(PDOException $e){
        echo("Err BDInsert : erreur suppression table employe dans admin_gestion_prof_supp.php<br>
        Message d'erreur : ".$e->getMessage());
    }
}
```

Si l'id du professeur dont on souhaite supprimer les informations existe dans le `$_GET`, alors on peut préparer la requête SQL de delete. Cette requête est paramétrée pour assurer la sécurité des données. Le seul paramètre est l'identifiant du professeur récupéré avec le `$_GET['id']`. Si la suppression a été réalisée un message d'information sera affiché à l'utilisateur sur la page de consultation des professeurs, sinon on affiche un message d'erreur.

Voici comment s'affiche le message d'information :

Consultation des professeurs

Ajouter un professeur

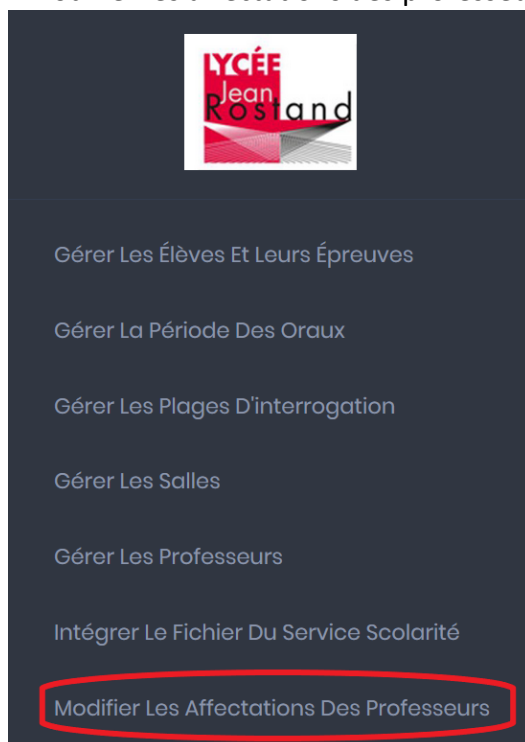
Les informations du professeur ont bien été supprimées

Nom	Prénom	E-mail	Discipline	Salle	Modifier	Supprimer
B	Sabine	@gmail.com	Allemand	109	Modifier	Supprimer

2. Modification des affectations par l'administrateur

Le but de cette application est de réaliser automatiquement les convocations des élèves. Les professeurs sélectionnent les demi-journées où ils souhaitent interroger, leur salle si besoin et les élèves qu'ils souhaitent interroger. Lorsque tous les professeurs se sont positionnés, Mme R. lance le traitement automatique qui répartit les élèves sur les demi-journées selon les règles de gestion qu'elle nous a données.

Lorsque tous les élèves ont été affectés, Mme R. peut accéder à une fonctionnalité permettant de modifier ces affectations. Cela peut lui être utile en cas d'absence des élèves par exemple. Cette fonctionnalité est accessible par l'onglet « Modifier les affectations des professeurs » du menu :



Cette fonctionnalité se présente sous la forme d'un tableau avec le nom et le prénom de l'élève, sa division, la discipline de l'épreuve qu'il passe, s'il s'agit de la LV1 ou la LV2 et la date l'horaire de l'épreuve avec la salle et le professeur qui le fait passer. Il y a deux lignes par élève l'une avec son épreuve de LV1, l'autre avec son épreuve de LV2.

Nom	Prénom	Division	Discipline	LV1/LV2	Plage Epreuve	Salle	Professeur
A [REDACTED]	SALMA	TS2	Espagnol	LV2	25/02/2019 11:00 ▾	120 ▾	[REDACTED] ▾
A [REDACTED]	SALMA	TS2	Anglais	LV1	26/02/2019 10:30 ▾	120 ▾	[REDACTED] ▾
A [REDACTED]	ADAM AHMED	TES2	Espagnol	LV2	28/02/2019 10:30 ▾	120 ▾	[REDACTED] ▾
A [REDACTED]	ADAM AHMED	TES2	Anglais	LV1	26/02/2019 16:00 ▾	119 ▾	[REDACTED] ▾
F [REDACTED]	JULIEN JOSE DENIS	TES3	Espagnol	LV2	28/02/2019 13:30 ▾	120 ▾	[REDACTED] ▾

Pour obtenir ces informations j'ai réalisé la requête SQL suivante :

```
SELECT
passageepreuve.id as idEpreuve,
eleve.nom as 'nom',
eleve.prenom as 'prenom',
division.libelle as 'division',
natureepreuve.libelle as 'natureepreuve',
utilisateur.nom as 'professeur',
discipline.libelle as discipline,
epreuve.idDiscipline,
passageepreuve.idProfAffecte as 'idProfAffecte',
demijournee.date,
demijournee.matinAprem,
passageepreuve.idSalle,
plage.heureDebut
FROM passageepreuve
LEFT OUTER JOIN utilisateur on idProfAffecte=utilisateur.id
LEFT OUTER JOIN eleve on idEleve=eleve.id
JOIN division on idDivision=division.id
JOIN epreuve on idEpreuve=epreuve.id
JOIN natureepreuve on idNatureEpreuve=natureepreuve.id
JOIN discipline on epreuve.idDiscipline = discipline.id
LEFT OUTER JOIN demijournee on demijournee.id = passageepreuve.idDemiJournee
LEFT OUTER JOIN plage on idPlage = plage.id
WHERE ((eleve.idSection=1 OR eleve.idSection IS NULL) OR (eleve.idSection = 2 AND idNatureEpreuve=2))
ORDER BY eleve.nom ASC
```

La clause WHERE de la requête SQL permet de ne pas afficher l'épreuve de LV1 pour les élèves en section internationale. Les élèves en section internationale ne sont pas concernés par l'oral de LV1 mais le sont par l'oral de LV2, ainsi on affiche seulement les informations concernant ce dernier.

Voici les instructions du tableau :

```
echo"<form action='admin_selection_eleves_valider.php' method='POST'>";
    echo" <table class ='table table-striped text-center'>";
        echo" <thead class='thead-dark'>";
            echo" <tr>";
                echo" <th>Nom</th>";
                echo" <th>Prénom</th>";
                echo" <th>Division</th>";
                echo" <th>Discipline</th>";
                echo" <th>LV1/LV2</th>";
                echo" <th>Plage Epreuve</th>";
                echo" <th>Salle</th>";
                echo" <th>Professeur</th>";
            echo" </tr>";
        echo"</thead>";
        foreach($lesEnregs as $enreg){
            $compteur++;
            echo" <tr>";
                echo" <td> $enreg->nom</td>";
                echo" <td> $enreg->prenom</td>";
                echo" <td> $enreg->division</td>";
                echo" <td> $enreg->discipline</td>";
                echo" <td> $enreg->natureepreuve <input type='hidden' name='idEpreuve$compteur' value=$enreg->idEpreuve>";
```

Le tableau est dans un formulaire parce qu'il permet de modifier les affectations des élèves grâce aux trois listes déroulantes de droite.

On crée donc le tableau avec les en-têtes puis on crée les lignes du tableau avec les informations retournées par la requête Select. Il a fallu ajouter un champ caché pour pouvoir effectuer la requête Update par la suite, l'idEpreuve est la clé primaire de la table passageepreuve dans laquelle on retrouve toutes les affectations. Le name est idEpreuve\$compteur car il doit être unique pour pouvoir récupérer tous les idEpreuve à modifier dans la base de données.

Les instructions ci-dessous permettent de sélectionner dans les listes déroulantes les informations concernant la convocation de l'élève. Cela sélectionne donc le professeur faisant passer l'épreuve à l'élève, la salle dans laquelle aura lieu son épreuve et la date ainsi que l'horaire de convocation. Ces valeurs sont celles présentes dans la base de données.

```
//dans le tableau on affiche une liste déroulante avec le nom de tous les professeurs de
//la discipline de l'épreuve |si l'élève a déjà été choisi le nom du professeur est affiché en premier.
?>
<?php
$lst_prof = $enreg->idProfAffecte;
$lst_salle = $enreg->idSalle;
$lst_plage = $enreg->date . $enreg->heureDebut;
```

On réalise et affiche ensuite la liste déroulante avec toutes les dates et plages d'interrogation. Les dates sont séparées entre les dates hors période d'interrogation et les dates dans la période d'interrogation. Les dates hors période sont des dates ajoutées après la période d'interrogation pour les élèves ayant été absents. J'ai donc effectué la requête SQL suivante qui retourne toutes les plages hors période, si la requête SQL n'a pas pu être exécuté alors on affiche un message d'erreur

```
//Liste déroulante avec la date et les plages
//-----
echo "<td>";
echo"<select class='custom-select custom-select' name='lst_plage$compteur' id='plage'><br>";
echo"<option value='0'>Aucune</option>";
try{
    //On récupère toutes les plages hors période
    $lesEnregsPlagesHP=$bdd->query("SELECT date, plage.heureDebut, idDemiJournee, idPlage
    from plagedemijournee
    join demijournee on idDemiJournee = demijournee.id
    join plage on idPlage = plage.id
    where demijournee.pperiode = '0'");
} catch(PDOException $e) {
    echo("err BDSelect : erreur de lecture tables plagedemijournee, demijournee, plage dans admin_selection_eleves_crit.php<br>
    Message d'erreur : ".$e->getMessage());
}
```


Si le select a retourné des enregistrements alors pour chacun d'entre eux on met la date retournée au format français c'est-à-dire le jour suivi du mois puis de l'année. Ensuite on crée les options de la liste déroulante en sélectionnant celui correspondant à la convocation de l'élève. La valeur de l'option sera l'id de la demi-journée et l'id de la plage séparés par un espace. Voici ces instructions :

```
//on teste si le select a retourné des enregistrement
if($lesEnregsPlagesHP->rowCount() > 0)
{
    //on affiche hors période au dessus des dates et horaires
    echo " <optgroup label='Hors Période'>";
    foreach($lesEnregsPlagesHP as $enregPlage) {
        //On met chaque date au format français JJ/MM/AAAA
        list($year, $month, $day) = explode("-", $enregPlage->date);
        $date_fr = $day."/".$month."/".$year;
        //On crée la liste et on sélectionne la date et l'horaire
        //de convocation de l'élève
        if($1st_plage == $enregPlage->date . $enregPlage->heureDebut){
            echo "<option class='form-group' selected value='$enregPlage->idDemiJournee $enregPlage->idPlage'>
                $date_fr $enregPlage->heureDebut</option>";
        }
        else{
            echo "<option class='form-group' value='$enregPlage->idDemiJournee $enregPlage->idPlage'>
                $date_fr $enregPlage->heureDebut</option>";
        }
    }
}
```

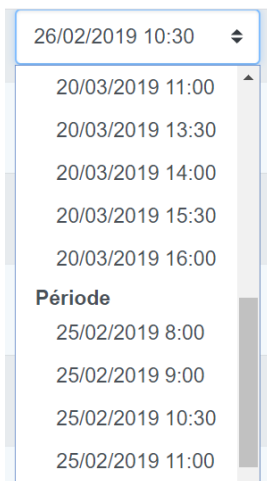
On fait ensuite la même chose pour les plages faisant parti la période des oraux :

```
try{
    $lesEnregsPlagesP=$bdd->query("SELECT date, plage.heureDebut, idDemiJournee, idPlage
    from plagedemijournee
    join demijournee on idDemiJournee = demijournee.id
    join plage on idPlage = plage.id
    where demijournee.periode = 'N'");
}catch(PDOException $e) {
    echo("err BDSelect : erreur de lecture tables plagedemijournee, demijournee, plage dans admin_selection_eleves_crit.pl
    Message d'erreur : ".$e->getMessage());
}

//on teste si le select a retourné des enregistrement
if($lesEnregsPlagesP->rowCount() > 0)
{
    echo " <optgroup label='Période'>";
    foreach($lesEnregsPlagesP as $enregPlage) {
        list($year, $month, $day) = explode("-", $enregPlage->date);
        $date_fr = $day."/".$month."/".$year;
        if($1st_plage == $enregPlage->date . $enregPlage->heureDebut){
            echo "<option class='form-group' selected value='$enregPlage->idDemiJournee $enregPlage->idPlage'>
                $date_fr $enregPlage->heureDebut</option>";
        }
        else{
            echo "<option class='form-group' value='$enregPlage->idDemiJournee $enregPlage->idPlage'>
                $date_fr $enregPlage->heureDebut</option>";
        }
    }
}

echo "</select>";
echo "</td>";
```

Voici un extrait de cette liste déroulante :



26/02/2019 10:30

20/03/2019 11:00

20/03/2019 13:30

20/03/2019 14:00

20/03/2019 15:30

20/03/2019 16:00

Période

25/02/2019 8:00

25/02/2019 9:00

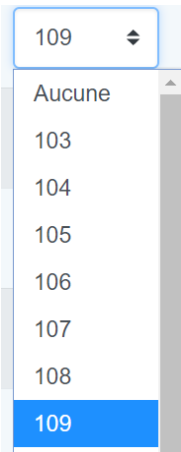
25/02/2019 10:30

25/02/2019 11:00

J'ai ensuite réalisé la liste déroulante contenant les salles dans lesquelles peuvent se dérouler les épreuves. Pour cela j'ai effectué un select dans la base de données qui retourne tous les libellés de salle et leur id. Si le select n'a pas pu être exécuté alors on affiche un message d'erreur. Si le select a retourné au moins une ligne alors on crée les options en sélectionnant la salle dans laquelle se déroule l'épreuve de l'élève. Voici les instructions permettant cela :

```
//-----  
//Liste déroulante contenant les salles  
//-----  
echo "<td>";  
echo"<select class='custom-select custom-select' name='lst_salle$compteur' id='salle'><br>";  
echo"<option value='0'>Aucune</option>";  
try{  
    $lesEnregsSalles=$bdd->query("SELECT id, libelle from salle order by libelle");  
}catch(PDOException $e) {  
    echo("err BDSelect : erreur de lecture table salle dans admin_selection_eleves_crit.php<br>  
    Message d'erreur : ".$e->getMessage());  
}  
  
    //on teste si le select a retourné des enregistrement  
if($lesEnregsSalles->rowCount() > 0)  
{  
    foreach($lesEnregsSalles as $enregSalle) {  
        if($lst_salle == $enregSalle->id){  
            echo "<option class='form-group' selected value='$enregSalle->id'$enregSalle->libelle</option>";  
        }  
        else{  
            echo"<option class='form-group' value='$enregSalle->id'$enregSalle->libelle</option>";  
        }  
    }  
}  
  
echo "</select>";  
echo"</td>";
```

Voici un extrait de cette liste déroulante :



A screenshot of a web form's dropdown menu. The menu is open, showing a list of numbers: 103, 104, 105, 106, 107, 108, and 109. The number 109 is highlighted in blue, indicating it is the selected option. Above the list, the number 109 is also displayed in a small box with a downward arrow, representing the current selection.

J'ai ensuite réalisé la liste déroulante des professeurs. Cette liste déroulante comporte seulement les professeurs enseignant la discipline de l'épreuve. C'est-à-dire que s'il s'agit d'une épreuve d'anglais alors seuls les professeurs d'anglais seront affichés. J'ai donc commencé par réaliser une requête Select sur la base de données qui affiche seulement les professeurs dont la discipline correspond à celle de l'épreuve. Si la requête n'a pas pu être exécutée on affiche un message d'erreur à l'utilisateur.

```
//-----  
//Liste déroulantes des professeurs  
//-----  
echo"<td>";  
echo"<select class='custom-select custom-select' required name='lst_prof$compteur' id='prof'>"; ?>  
    <option value="0">Aucun</option>  
    <?php  
  
        //exécution de la requête (avec la méthode query) pour obtenir le contenu de la table  
        // on récupère le résultat de la requête dans le tableau $lesenregs  
        try{  
            $lesenregsProfs=$bdd->query("SELECT utilisateur.id as idProf, utilisateur.nom as nomProf  
from utilisateur  
Join discipline on utilisateur.idDiscipline = Discipline.id  
where utilisateur.idTypeUtilisateur = 2 and (utilisateur.idDiscipline= $enreg->idDiscipline )");  
        }catch(PDOException $e) {  
            die("err BDSelect : erreur de lecture table dans admin_selection_eleve-crit.php<br>  
            Message d'erreur : ".$e->getMessage());  
        }  
    }  
    </td>";  
}
```

Si le select a retourné au moins un enregistrement, on crée les options de la liste déroulante en sélectionnant le professeur qui doit faire passer l'épreuve à l'élève. Voici ces instructions :

```
//on teste si le select a retourné des enregistrement  
if($lesenregsProfs->rowCount() > 0)  
{  
  
    //pour chaque enregistrement retourné par la requête SQL, on crée une option dans la liste  
    //l'attribut value contiendra l'id (l'identifiant du prof )et son nom  
    //on limite le tableau retourné et pour chaque enregistrement, on affiche le nom  
    foreach($lesenregsProfs as $prof) {  
        if($lst_prof == $prof->idProf){  
            echo "<option class='form-group' selected value='$prof->idProf'>$prof->nomProf</option>";  
        }else{  
            echo"<option class='form-group' value='$prof->idProf'>$prof->nomProf</option>";  
        }  
    }  
}
```

Lorsque l'utilisateur a terminé ses modifications il doit cliquer sur le bouton « Valider » en dessous du tableau.

WY [REDACTED]	ILANA FRANCOISE LUCIENNE	TS2	Espagnol	LV2	01/03/2019 8:00	105	Aucun
WY [REDACTED]	ILANA FRANCOISE LUCIENNE	TS2	Anglais	LV1	25/02/2019 10:30	128	[REDACTED]



Le script de validation est alors appelé. On vérifie tout d'abord que le formulaire a bien été validé. Ensuite pour chaque ligne du tableau on récupère l'id de l'épreuve récupéré à partir du champ caché, la valeur de lst_salle qui est l'id de la salle sélectionnée dans la liste déroulante et la valeur de lst_plage. Dans la liste déroulante cette valeur est l'id de la demi-journée et l'id de la plage séparés par un espace, il faut donc séparer les deux valeurs à l'aide de la fonction explode. Voici les instructions permettant cela :

```
if(isset($_POST['btn_valider'])==true){  
    foreach($_POST as $cle=>$valeur){  
        try{  
            //On récupère la valeur de l'idEpreuve  
            if (strpos($cle, "idEpreuve")==0) {  
                $val_idEpreuve=$valeur;  
            }  
            //on récupère la valeur de l'idSalle  
            if(strpos($cle, "lst_salle")==0)  
            {  
                $val_idSalle=$valeur;  
            }  
            //on récupère la valeur sélectionné pour la plage  
            if(strpos($cle, "lst_plage")==0)  
            {  
                $val_plage=$valeur;  
  
                // on sépare la date et l'heure de l'épreuve  
                $lesPlages = explode(" ", $val_plage);  
                $idDemiJournee = $lesPlages[0];  
                $idPlage = $lesPlages[1];  
            }  
        }  
    }  
}
```

Enfin, on récupère l'id du professeur sélectionné dans la liste déroulante puis on peut préparer la requête Update paramétrée pour assurer la sécurité des données. Les nouvelles valeurs sont celles récupérées précédemment. On exécute la requête puis on redirige la page vers `admin_selection_eleve.php` en affichant un message à l'utilisateur pour l'informer que la modification a bien eu lieu. S'il y a eu un problème lors de l'update on affiche un message d'erreur à l'utilisateur. Voici ces instructions :

```

}
    ///On récupère l'id du prof sélectionné
    if(strpos($cle, "lst_prof") ===0) {
        $val_prof=$valeur;
        $req=$bdd->prepare("UPDATE passageepreuve set idProfAffecte =:par_idProf, idSalle = :par_Salle,
        idDemiJournee = :par_idDemiJournee, idPlage = :par_idPlage
        where id=:par_idEpreuve");
        //si l'id prof est 0 alors on met null dans la bdd
        if ($val_prof==0){
            $req->bindValue (':par_idProf', null, PDO::PARAM_INT);
        }else {
            //sinon on met l'id du professeur sélectionné
            $req->bindValue(':par_idProf', $val_prof,PDO::PARAM_INT);
        }
        $req->bindValue(':par_Salle', $val_idSalle,PDO::PARAM_INT);
        $req->bindValue(':par_idDemiJournee', $idDemiJournee,PDO::PARAM_INT);
        $req->bindValue(':par_idPlage', $idPlage,PDO::PARAM_INT);
        $req->bindValue(':par_idEpreuve', $val_idEpreuve,PDO::PARAM_INT);

        $req->execute();
        //on se redirige vers la sélection des élèves en fournissant le message d'information
        $msg="Les modifications ont été prises en compte";
        header('Location:admin_selection_eleves.php?msg='.$msg);
    }
}
}
catch(PDOException $e){
    die("err BDUpdate : erreur d'update table passageepreuve dans admin_selection_eleves_valider.php<br>
    Message d'erreur : ".$e->getMessage());
}

```

Le nombre d'élèves étant important, il y a énormément de lignes dans le tableau et il n'est pas facile de trouver un élève en particulier. Pour rendre plus simple la recherche, j'ai réalisé trois filtres, on peut ainsi choisir d'afficher une division par exemple tous les élèves de TS1, toutes les épreuves auxquelles l'élève a été absent ou afficher toutes les épreuves de LV1 ou de LV2. Il est également possible d'utiliser deux ou trois filtres, par exemple avoir toutes les épreuves auxquelles l'élève a été absent en TS2 ou toutes les épreuves de LV1 auxquelles l'élève a été absent en TS2.

Le choix de la division se fait dans une liste déroulante et le choix de la nature d'épreuve et des absences se fait avec des radios boutons.

Sélection des élèves

Choix de la division

TS2 ▼

Choix élèves

Tous les élèves ☒

Elèves absents ☐

Choix langue vivante

Toutes les natures d'épreuves ☒

LV1 ☐

LV2 ☐

Pour réaliser cette liste déroulante j'ai réalisé les instructions suivantes :

```
<label><b>Choix de la division</b></label></b><br>
<select required name="lst_division" id="lst_division" onchange="reset_rbt();">
  <option value='0'>Veuillez sélectionner une division</option>
</php>
$msg="";
if(isset ($_GET['msg'])==true){
  $msg=$_GET['msg'];
}

//exécution de la requête (avec la méthode query) pour obtenir le contenu de la table lst_division
// on récupère le résultat de la requête dans le tableau $lesenregs
try{
  $lesenregs=$bdd->query("SELECT id, libelle from division");
}catch(PDOException $e) {
  echo("err BDSelect : erreur de lecture table division dans admin_selection_eleves.php.php<br>
  Message d'erreur : ".$e->getMessage());
}
```

On crée la liste déroulante avec pour première option « Veuillez sélectionner une division ». Au lancement de la page on sera positionné sur cette option la valeur sera donc 0 ce qui nous permettra d'afficher toutes les divisions. On essaye ensuite d'exécuter la requête Select qui retourne toutes les divisions présentes dans la base de données. Si la requête ne peut pas être exécutée on affiche un message d'erreur.

```
//on teste si le select a retourné des enregistrement
if($lesenregs->rowCount() > 0)
{
  //pour chaque enregistrement retourné par la requête SQL, on crée une option dans la liste
  //l'attribut value contiendra l'id (l'identifiant de la division)
  // et le libellé de la division sera affiché
  //on limite le tableau retourné et pour chaque enregistrement, on affiche la division
  foreach($lesenregs as $enreg) {
    if($lst_division == $enreg->id){
      echo "<option selected value='$enreg->id' id='idDivision'>$enreg->libelle</option>";
    }else{
      echo "<option value='$enreg->id' id='idDivision'>$enreg->libelle</option>";
    }
  }
}
echo("</select>");
```

Ensuite si le select a retourné des enregistrements, on crée les options de la liste déroulante à partir des divisions retournées. Si une division a déjà été sélectionnée précédemment, on la sélectionne dans la liste déroulante.

Pour choisir si on souhaite afficher tous les élèves ou seulement les élèves ayant été absents j'ai réalisé deux radio-boutons dont voici les instructions :

```
<label><b>Choix élèves</b></label>
<br>
Tous les élèves &nbsp;<input type='radio' class='rbt_absence' name='rbt_absence' checked = 'checked' value='T'id="idelevesT"/>
<br>
Elèves absents &nbsp;<input type='radio' class='rbt_absence' name='rbt_absence' value='0'/>
<br>
<br><br>
```

Pour le choix de la nature d'épreuve j'ai réalisé les radios boutons en fonction des données de la base de données. Ainsi je réalise une requête Select qui retourne toutes les natures d'épreuves et j'affiche un message d'erreur s'il y a eu un problème. Si la requête SQL a retourné au moins un enregistrement j'affiche un premier radio bouton pour afficher toutes les natures d'épreuves qui est sélectionné lors du premier affichage de la page, ensuite je crée les radio-boutons à partir des enregistrements retournés. Voici les instructions réalisant cela :

```
<label for="Toutes les natures d'épreuve "><b>Choix langue vivante</b></label><br>
<?php
//execution de la requête qui récupère le contenu de la table
// on récupère le résultat de la requête dans le tableau $lesEnregs
try{
    $lesEnregs=$bdd->query("SELECT id, libelle from natureepreuve");
}catch(PDOException $e) {
    echo("Err BDSelect : erreur de lecture table natureepreuve dans admin_selection_eleves_crit.php.php<br>
        Message d'erreur : " . $e->getMessage());
}
//on teste si le select a retourné des enregistrements
if($lesEnregs->rowCount()>0){

    //radio bouton pour obtenir si l'epreuve est LV1 ou LV2
    echo("<b>Toutes les natures d'épreuves <input type='radio' class='rbt_natep' name='rbt_natep'
checked = 'checked' value='T' id='idnatepT'/></b>");
    echo"<br>";
    //on lit le tableau retourné par la requête SELECT
    //et pour chaque enregistrement génère un radio bouton
    //l'attribut value contient l'id de la nature de l'épreuve
    foreach($lesEnregs as $enreg){
        echo"$enreg->libelle <input type='radio' class='rbt_natep' name='rbt_natep' value='".$enreg->id."' />";

        echo"<br>";
    }
}else{
    echo"<br>Consultation impossible : aucune nature d'épreuve n'a été enregistrée";
}
```


J'ai utilisé AJAX pour que les filtres soient pris en compte lors de l'affichage, ainsi la page est automatiquement rechargée avec les filtres cochés sans avoir besoin d'un formulaire et d'un bouton « valider ».

```
$(document).ready(function(){
    $("input[name=rbt_natep]").click(function() {
        $(".resultat").load("admin_selection_eleves_crit.php",{
            'idabsence' : $('input[type=radio][name=rbt_absence]:checked').attr('value'),
            'idnatep' : $('input[type=radio][name=rbt_natep]:checked').attr('value'),
            'idDivision' : $("#1st_division").val()
        });
    });
});

$(document).ready(function(){
    $("input[name=rbt_absence]").click(function() {
        $(".resultat").load("admin_selection_eleves_crit.php",{
            'idabsence' : $('input[type=radio][name=rbt_absence]:checked').attr('value'),
            'idnatep' : $('input[type=radio][name=rbt_natep]:checked').attr('value'),
            'idDivision' : $("#1st_division").val()
        });
    });
});

//-----
//Exécuté au lancement du formulaire
$(document).ready(function(){
    $.ajax({
        url : 'admin_selection_eleves_crit.php', // script appelé
        type : 'POST', // Le type de la requête HTTP est POST
        data : 'idDivision='+ 0 +'&idnatep=T'+'&idabsence=T', //l'id sélectionné est passé au script
        dataType : 'html', //resultat sera en HTML
        success : function(code_html, statut){
            //l'appel s'est bien passé : on met le résultat dans la div resultat
            $(".resultat").html (code_html);
        },
        error :function(resultat, statut, erreur){
            //si l'appel ne se passe pas bien on affiche l'erreur
            $(".resultat").html("Erreur : " + resultat.reponseText);
        }
    });
});

$("#1st_division").change(function(){
    $.ajax({
        url : 'admin_selection_eleves_crit.php', // script appelé
        type : 'POST', // Le type de la requête HTTP est POST
        data : 'idDivision=' + $("#1st_division").val() +
            '&idnatep=' +$("#.rbt_natep").val()+ //l'id sélectionné est passé au script
            '&idabsence=' +$("#.rbt_absence").val(),
        dataType : 'html', //resultat sera en HTML
        success : function(code_html, statut){
            //l'appel s'est bien passé : on met le résultat dans la div resultat
            $(".resultat").html (code_html);
        },
        error :function(resultat, statut, erreur){
            //si l'appel ne se passe pas bien on affiche l'erreur
            $(".resultat").html("Erreur : " + resultat.reponseText);
        }
    });
});
});
```


Ainsi lorsque les filtres sont utilisés la clause where de la requête SQL permettant d'afficher les informations du tableau change.

Lorsqu'**aucun filtre** n'est appliqué ou que le filtre ne se fait que sur la **division**, la clause where permet d'afficher toutes les épreuves des élèves qui ne sont pas en section internationale et l'épreuve de LV2 des élèves en section internationale dont la division est \$idDivision. Si aucune division n'a été sélectionnée dans la liste \$idDivision = 0 donc toutes les divisions seront affichées. Voici la requête SQL :

```
SELECT
passageepreuve.id as idEpreuve,
eleve.nom as 'nom',
eleve.prenom as 'prenom',
division.libelle as 'division',
natureepreuve.libelle as 'natureepreuve',
utilisateur.nom as 'professeur',
discipline.libelle as discipline,
epreuve.idDiscipline,
passageepreuve.idProfAffecte as 'idProfAffecte',
demijournee.date,
demijournee.matinAprem,
passageepreuve.idSalle,
plage.heureDebut
FROM passageepreuve
LEFT OUTER JOIN utilisateur on idProfAffecte=utilisateur.id
LEFT OUTER JOIN eleve on idEleve=eleve.id
JOIN division on idDivision=division.id
JOIN epreuve on idEpreuve=epreuve.id
JOIN natureepreuve on idNatureEpreuve=natureepreuve.id
JOIN discipline on epreuve.idDiscipline = discipline.id
LEFT OUTER JOIN demijournee on demijournee.id = passageepreuve.idDemiJournee
LEFT OUTER JOIN plage on idPlage = plage.id
WHERE ((eleve.idSection=1 OR eleve.idSection IS NULL) OR (eleve.idSection = 2 AND
idNatureEpreuve=2)) AND((idDivision=$idDivision AND $idDivision !=0) OR $idDivision =0)
ORDER BY eleve.nom ASC
```

Lorsqu'on filtre sur une **division** et sur les élèves **absents** ou **seulement les élèves absents**, la clause where permet d'afficher toutes les épreuves des élèves qui ne sont pas en section internationale et l'épreuve de LV2 des élèves en section internationale dont les élèves ont été notés absents et dont la division est \$idDivision. Si aucune division n'a été sélectionnée dans la liste \$idDivision = 0 donc toutes les divisions seront affichées. Voici la clause where de la requête SQL :

```
WHERE ((eleve.idSection=1 OR eleve.idSection IS NULL) OR (eleve.idSection = 2 AND
idNatureEpreuve=2))AND((idDivision=$idDivision AND $idDivision !=0) OR $idDivision =0) AND absence='O'
```

Lorsqu'on filtre sur une **division** et une **nature d'épreuve** ou **seulement une nature d'épreuve** la clause where permet d'afficher toutes les épreuves des élèves qui ne sont pas en section internationale et l'épreuve de LV2 des élèves en section internationale dont l'id de la nature d'épreuve est \$idnaturep et dont la division est \$idDivision. Si aucune division n'a été sélectionnée dans la liste \$idDivision = 0 alors toutes les divisions seront affichées. Voici la clause where de la requête SQL :

WHERE ((eleve.idSection=1 OR eleve.idSection IS NULL) OR (eleve.idSection = 2 AND idNatureEpreuve=2))
AND ((idDivision=\$idDivision AND \$idDivision !=0) OR \$idDivision =0) AND(natureepreuve.id= \$idnatep)

Lorsqu'on filtre sur une **division**, une **nature d'épreuve** et les élèves ayant été **absents** ou une nature d'épreuve et les élèves absents la clause where permet d'afficher toutes les épreuves des élèves qui ne sont pas en section internationale et l'épreuve de LV2 des élèves en section internationale dont les élèves ont été notés absents et dont l'id de la nature d'épreuve est \$idnatep et la division est \$idDivision. Si aucune division n'a été sélectionnée dans la liste \$idDivision = 0 alors toutes les divisions seront affichées. Voici la clause where de la requête SQL :

where ((eleve.idSection=1 OR eleve.idSection IS NULL) OR (eleve.idSection = 2 AND idNatureEpreuve=2))AND ((idDivision=\$idDivision AND \$idDivision !=0) OR \$idDivision =0) AND (natureepreuve.id= \$idnatep) AND absence='O'