ÉQUIPE E

Compte Rendu Final

Projet n°4

Sommaire

Introduction

CONTEXTE

Les deux responsables du service « Système d'informations » de la MS2R, M. OPALINE pour la partie système et réseau et MME MIKOLAS pour la partie développement d'applications et bases de données, nous proposent un dernier projet sur la gestion de l'environnement web de la société.

ORGANISATION

L'équipe E est composée de 6 personnes :

- M. WAÏ-LUNE Nathan, Chef de projet ;
- M. VINGADASSALOM Brian ;
- M. WAN-TANG-KIVAN Lucas ;
- M. HOARAU Florian ;
- M. MAILLOT Lilian ;
- M. BARRET Toma ;

Nous avons dès le départ séparer l'équipe en 2 ; les personnes travaillant sur la gestion de l'environnement web et les personnes travaillant sur la gestion du Back Office du site Web.

Il n'était cependant pas impossible que l'une travaille avec l'autre.

Le projet de la MS2R

Il est composé de 2 grandes parties abordant différents thèmes, eux mêmes fractionner en plusieurs sections :

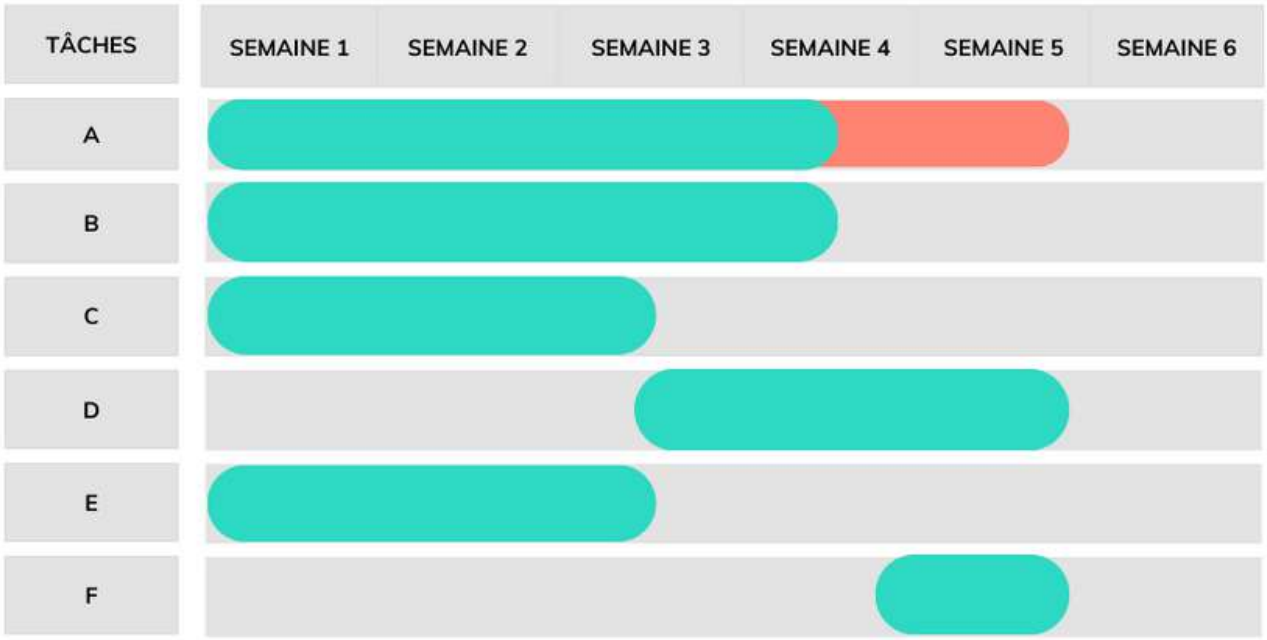
La Gestion de l'environnement système :

- La mise en place du nouveau serveur web ;
- La reprise sur incident ;

La Gestion du Back Office du site web :

- La gestion des membres de la MS2R ;
- La gestion des actualités ;
- La gestion des droits d'accès au Front Office ;
- l'hébergement du site web ;

Gestion du projet



Tâche A : Mise en place du nouveau serveur web ;
Tâche B ; Reprise sur incident ;
Tâche C : Gestion des membres de la MS2R ;
Tâche D : Gestion des actualités ;
Tâche E : Gestion des droits d'accès au Front Office ;
Tâche F : Hébergement du site web ;

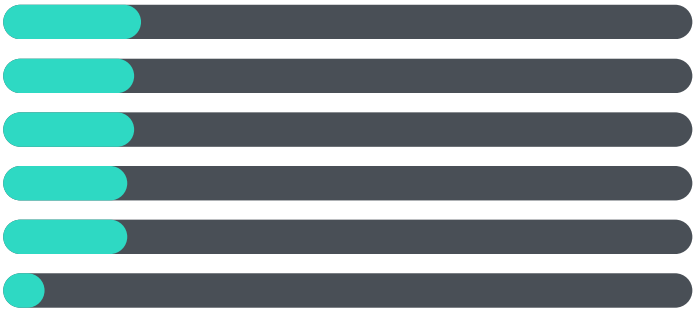
Chaque semaine, les membres de l'équipe reprenaient la tâche qu'ils n'avaient pas fini lors de la séance précédente. Du travail en dehors de ces séances a évidement été nécessaire. La tâche A avait pris du retard à cause de différents problèmes que nous avons eus avec FTP, mais cela n'a eu aucune répercurtion sur le déroulement du projet

Investissement & Charge de travail

INVESTISSEMENT :

17 %	VINGADASSALOM
17 %	WAÏ-LUNE
17 %	WAN-TANG-KIVAN
17 %	HOARAU
17 %	BARRET
15 %	MAILLOT

CHARGE DE TRAVAIL :



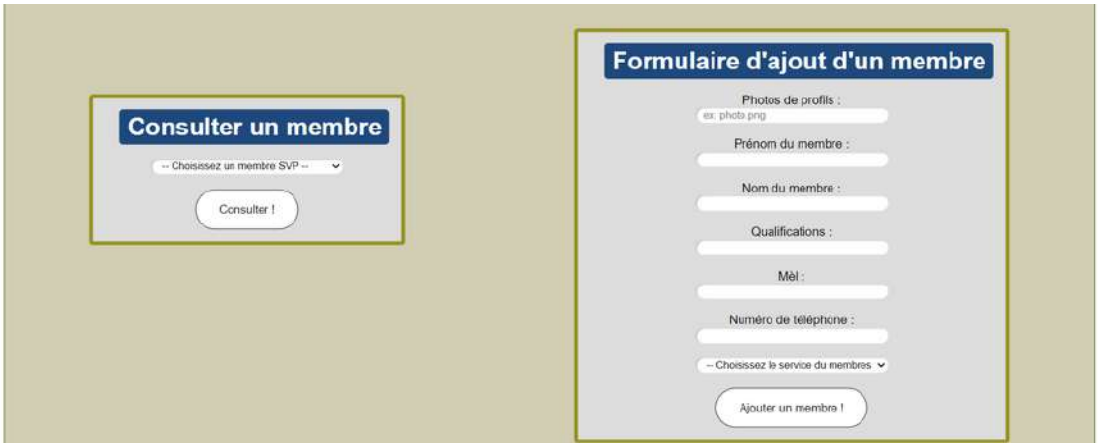
20 %	VINGADASSALOM
20 %	HOARAU
18 %	BARRET
18 %	WAÏ-LUNE
18 %	WAN-TANG-KIVAN
6 %	MAILLOT

GESTION DU BACK OFFICE DU SITE WEB

PARTIE A ET B : GESTIONS MEMBRES ET NEWS

Pour ce projet, M.HOARAU a repris un site existant afin de continuer le travail déjà entamé.
Notre but était de mettre en place un BackOffice fonctionnel en fonction des besoins de notre client.

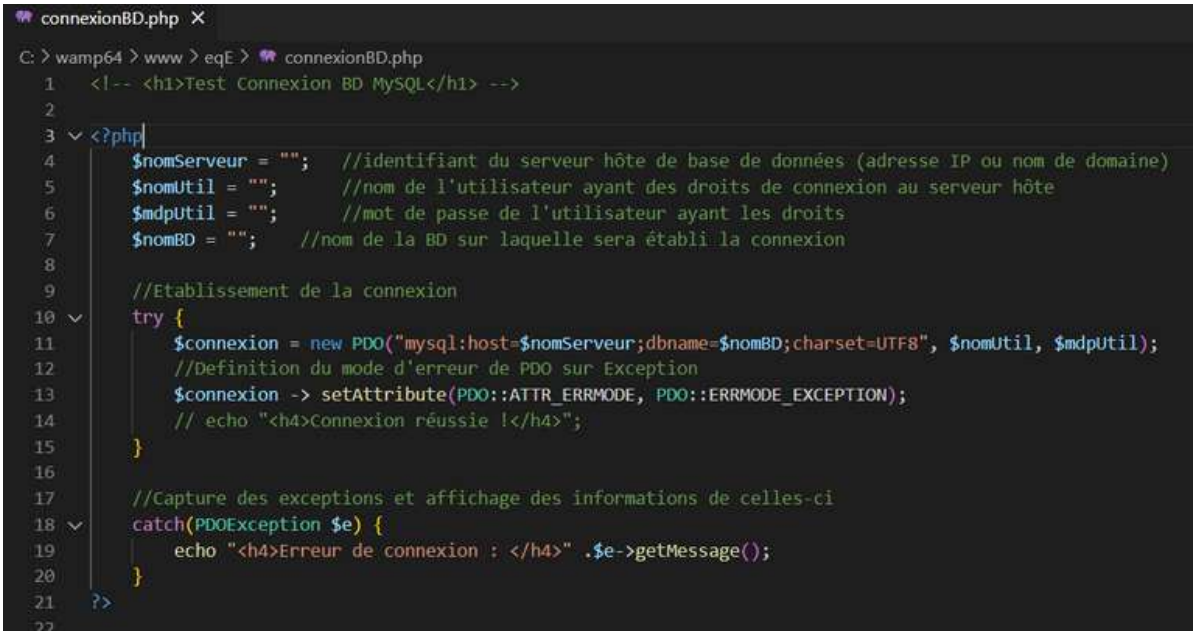
Il a donc pris en charge la création de différents formulaires qui permettent la gestion des membres et des News.
Ces formulaires pourront ajouter, supprimer et modifier des enregistrements sans que l'utilisateur effectue de lignes de commande.



GESTION DU BACK OFFICE DU SITE WEB

Ceci permettra aux clients de gérer les différents enregistrements de la base de données simplement en appuyant sur un bouton ou en sélectionnant des informations clairement afficher sur les formulaires.

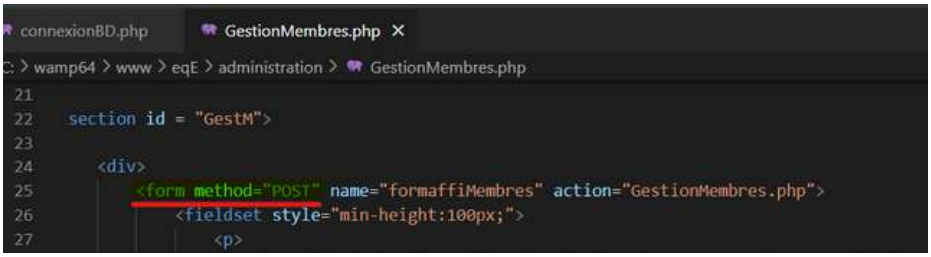
La création des différentes pages a nécessité une connexion avec la base de données qui a pour but de récupérer les enregistrements des gestions des membres et des News, et de rendre le site de la MS2R dynamique.



GESTION DU BACK OFFICE DU SITE WEB

Il a dû utiliser plusieurs langages (informatique et programmation), comme le PHP pour la connexion, la sécurisation du site web et le transfère des diférentes informations qui mets à jour la partie front Office du site en temps réelle.
L'utilisation de la méthode "POST" est vivement conseiller pour la transmission des données entre informations saisies dans le formulaire et le serveur (ou la base de données).

Exemple d'un formulaire utilisant la méthode "POST":



Lorsque les données sont envoyées via à la méthode "POST", elles sont incluses dans le corps de la requête HTTP plutôt que dans l'URL, ce qui signifie que les données ne sont pas visibles dans l'adresse URL et ne sont donc pas facilement accessibles par des tiers.

GESTION DU BACK OFFICE DU SITE WEB

L'ajout de deux pages membresManager et newsManager implique la création de deux nouvelles classes. Dans chacune de ces classes, nous allons déclarer un constructeur, qui est une méthode spéciale appelée automatiquement lors de la création d'un nouvel objet de cette classe.
Le constructeur permet d'initialiser les propriétés de l'objet dès sa création. Nous allons également déclarer des attributs pour chaque classe, qui sont des variables utilisées pour stocker des données spécifiques à chaque objet de la classe. Ces attributs peuvent être définis comme publics ou privés
Dans notre cas, l'accès que nous souhaitons donner aux autres parties du code.sera privés.

```php
<?php

class MembresManager
{
    //Déclaration des attributs de la classe
    private $_id;                   //l'identifiant des membres
    private $_photos;               //photos des membres
    private $_prenom;               //prenom des membres
    private $_nom;                  //nom des membres
    private $_qualifications;       //qualifications des membres
    private $_mail;                 //mail des membres
    private $_telephone;            //Numéro de téléphone des membres
    private $_idService;            //Identifiant service des membres




    //Déclaration du constructeur
    public function __construct($idMembres,$photosMembres,$prenomMembres,$nomMembres,$qualific
    {
        $this->_id = $idMembres;        // Initialisation de l'identifiant de cet objet
        $this->_photos = $photosMembres;
        $this->_prenom = $prenomMembres;
        $this->_nom = $nomMembres;
        $this->_qualifications = $qualificationsMembres;
        $this->_mail = $mailMembres;
        $this->_telephone = $telephoneMembres;
        $this->_idService = $idServiceMembres;

    }
```

GESTION DU BACK OFFICE DU SITE WEB

Les méthodes sont des fonctions définies à l'intérieur d'une classe qui permettent de réaliser des actions spécifiques sur les objets de cette classe. Elles sont utilisées pour manipuler les propriétés des objets et pour effectuer des opérations plus complexes sur ces derniers.

La méthode "create" est utilisée pour ajouter un nouveau membre à la base de données. Elle permet d'insérer les informations d'un membre (nom, prénom, qualifications, etc.) dans la table "Membres" en utilisant une requête SQL. Cette méthode est utilisée chaque fois qu'un nouvel utilisateur est créé dans le système.

```php
public function create()
{
    require_once "../connexionBD.php";
    $connexion->exec("SET NAMES utf8");
    $sql = "INSERT INTO Membres (photos, prenom, nom, qualifications, email, telephone, idService) VALUES ('".$this->
    // echo $sql;
    $connexion->exec($sql) or die(print_r($connexion->errorInfo(), true));
}
```

La méthode "retrieve" est utilisée pour récupérer les informations d'un membre à partir de son identifiant (id). Elle permet d'effectuer une requête SQL pour obtenir les données associées à l'id du membre. Les informations récupérées sont ensuite stockées dans les variables de l'objet "Membre". Cette méthode est utilisée chaque fois qu'un membre est sélectionné pour afficher ou modifier ses informations.

GESTION DU BACK OFFICE DU SITE WEB

```php
public function retrieve()

{
    require "../connexionBD.php";
    $connexion->exec("SET NAMES utf8");
    $sql = "SELECT * FROM membres WHERE id ='".$this->_id."'";
    $resultat = $connexion->query($sql);
    $ligne = $resultat->fetch();
    $this->_id = $ligne['id'];
    $this->_photos = $ligne['Photos'];
    $this->_prenom = $ligne['Prenom'];
    $this->_nom = $ligne['Nom'];
    $this->_qualifications = $ligne['Qualifications'];
    $this->_mail = $ligne['email'];
    $this->_telephone = $ligne['Telephone'];
    $this->_idService = $ligne['idService'];


}
```

La méthode "delete" est utilisée pour supprimer un membre de la base de données. Elle utilise une requête SQL pour supprimer le membre correspondant à l'identifiant (id) spécifié. Cette méthode est utilisée chaque fois qu'un utilisateur doit être supprimé du système dans le back office.

```php
public function delete()
{
    require_once "../connexionBD.php";
    $connexion->exec("SET NAMES utf8");
    $sql3 = "DELETE FROM membres WHERE id = '".$this->_id."'; ";
    // echo $sql3;
    $connexion->exec($sql3) or die(header("Location: ./echecMembres.php"));
}
```

GESTION DU BACK OFFICE DU SITE WEB

La méthode "update" est utilisée pour mettre à jour les informations d'un membre dans la base de données. Elle utilise une requête SQL pour modifier les informations du membre correspondant à l'identifiant (id) spécifié. Cette méthode est utilisée chaque fois qu'un utilisateur modifie ses informations.

```php
public function update()
{
    require_once "../connexionBD.php";
    $connexion->exec("SET NAMES utf8");
    $sql1 = "UPDATE membres SET photos = '".$this->_photos."', prenom = '".$this->_preno
    $connexion->exec($sql1) or die(header("Location: ./echecMembres.php"));
}
```

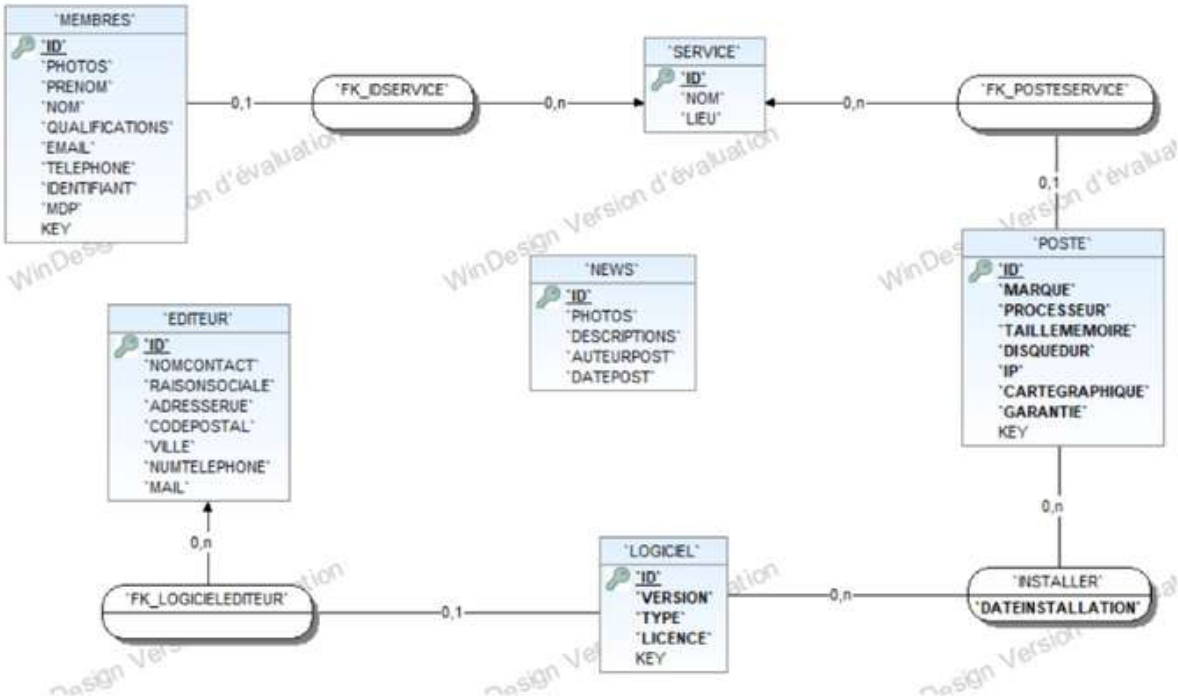
La méthode "get" est utilisée pour obtenir les valeurs des différentes propriétés de l'objet "Membre". Elle est utilisée chaque fois qu'un développeur a besoin d'accéder aux informations d'un membre dans le code source du système.

```php
public function getId()
{
    return $this->_id;
}
```

```php
<center>
    <input type="hidden" name="id" maxlength="100" value="'.$consulterMembres->getId().'">
    <p>Photos : <br> <input type="text" name="photos" maxlength="100" value="'.$consulterMembres->getPhotos().'" ></p><br>
    <p>Prénom : <br> <input type="text" name="prenom" value="'.$consulterMembres->getPrenom().'" ></p><br>
    <p>Nom : <br> <input type="text" name="nom" value="'.$consulterMembres->getNom().'" ></p><br>
    <p>Qualifications : <br> <input type="text" name="qualifications" value="'.$consulterMembres->getQualifications().'" ></p><br>
    <p>Le mèl : <br> <input type="text" name="email" value="'.$consulterMembres->getMail().'" ></p><br>
    <p>Numéro de téléphone : <br> <input type="text" name="telephone" value="'.$consulterMembres->getTelephone().'" ></p><br>
    <p>Service : <br> <select name="idService">
        <option value="'.$consulterMembres->getidService().'">'.$donnees2['nom'].'</option>';
```

GESTION DU BACK OFFICE DU SITE WEB

PARTIE B : SCHÉMA ENTITÉ ASSOCIATION



Le schéma entité association décrit la relation entre plusieurs entités : MEMBRES, SERVICES, POSTE, LOGICIEL et EDITEUR. On peut observer plusieurs types de relations entre ces entités.

La première relation se situe entre MEMBRES et SERVICES. Chaque MEMBRE peut être associé à zéro ou un SERVICE, tandis qu'un SERVICE peut être associé à zéro ou plusieurs MEMBRES. Cette relation est matérialisée par une clé étrangère FK_IDSERVICE dans la table MEMBRES.

GESTION DU BACK OFFICE DU SITE WEB

La seconde relation concerne les entités SERVICES et POSTE. Un SERVICE peut être associé à zéro ou plusieurs POSTE, tandis qu'un POSTE peut être associé à zéro ou un SERVICES. Cette relation est représentée par une clé étrangère FK_POSTERSERVICE dans la table SERVICES.

La troisième relation se situe entre POSTE et LOGICIEL. Un POSTE peut avoir zéro ou plusieurs LOGICIELS installés, et un LOGICIEL peut être installé sur zéro ou plusieurs POSTES. Cette relation est représentée par une table intermédiaire appelée INSTALLER.

Enfin, la quatrième relation concerne l'entité LOGICIEL et EDITEUR. Chaque LOGICIEL peut être édité par zéro ou un EDITEUR, tandis qu'un EDITEUR peut éditer zéro ou plusieurs LOGICIELS. Cette relation est matérialisée par une clé étrangère FK_LOGICIELEDITEUR dans la table LOGICIEL.

GESTION DU BACK OFFICE DU SITE WEB

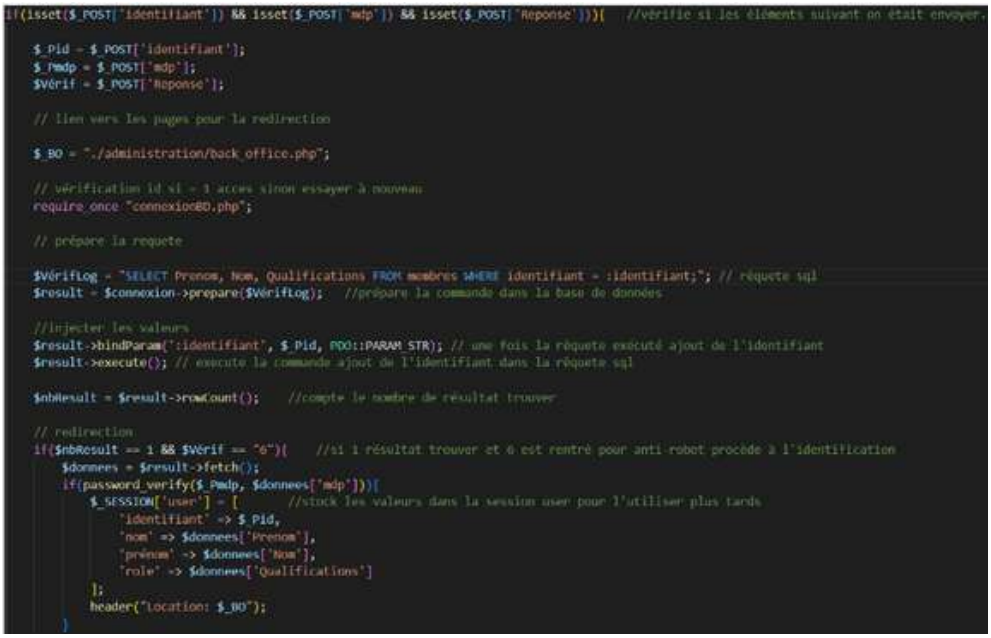
PARTIE C – SÉCURITÉ – GESTION DES DROITS D'ACCÈS AU FRONT-OFFICE

Mise en place des sessions pour la sécurité du back office. (Refonte de la page login par la même occasion)
Si la connexion est réussie, les informations de la base de données sur l'utilisateur seront enregistrées dans des variables différentes afin de les réutiliser plus tard.

```
$_SESSION['user'] = [       //stock les valeurs dans la session user pour l'utiliser plus tards
    'identifiant' => $_Pid,
    'nom' => $donnees['Prenom'],
    'prénom' => $donnees['Nom'],
    'role' => $donnees['Qualifications']
];
header("Location: $_BO");
```

Mise en place de la vérification pour la connexion :
Après la récupération des valeurs saisies par l'utilisateur via le formulaire de connexion.
Une requête SQL sera en préparation pour vérifier l'utilisateur, avec l'utilisateur trouvé on vérifiera le mot de passe s'il correspond bien au mot de passe dans la base de donnée avec password_verify().
Si les conditions suivantes "si 1 utilisateur est trouvé", "vérifie si le calcule est correcte", "vérifie si le mot de passe entré par l'utilisateur correspond à celui haché dans la base de donnée", sont remplies alors connexion au back-office est faite.

GESTION DU BACK OFFICE DU SITE WEB



```php
if(isset($_POST['identifiant']) && isset($_POST['mdp']) && isset($_POST['Reponse'])){   //vérifie si les éléments suivant on était envoyer.

    $_Pid = $_POST['identifiant'];
    $_Pmdp = $_POST['mdp'];
    $Vérif = $_POST['Reponse'];

    // lien vers les pages pour la redirection

    $_BO = "./administration/back_office.php";

    // verification id si = 1 acces sinon essayer à nouveau
    require_once "connexionBD.php";

    // prépare la requete

    $VérifLog = "SELECT Prenom, Nom, Qualifications FROM membres WHERE identifiant = :identifiant;"; // requete sql
    $result = $connexion->prepare($VérifLog);   //prépare la commande dans la base de données

    //injecter les valeurs
    $result->bindParam(':identifiant', $_Pid, PDO::PARAM_STR); // une fois la requete exécuté ajout de l'identifiant
    $result->execute(); // execute la commande ajout de l'identifiant dans la requete sql

    $nbResult = $result->rowCount();    //compte le nombre de résultat trouver

    // redirection
    if($nbResult == 1 && $Vérif == "6"){    //si 1 résultat trouver et 6 est rentré pour anti robot procède à l'identification
        $donnees = $result->fetch();
        if(password_verify($_Pmdp, $donnees['mdp'])){
            $_SESSION['user'] = [       //stock les valeurs dans la session user pour l'utiliser plus tards
                'identifiant' => $_Pid,
                'nom' => $donnees['Prenom'],
                'prénom' => $donnees['Nom'],
                'role' => $donnees['Qualifications']
            ];
            header("Location: $_BO");
        }
```

Pour la sécurité du mot de passe, ce dernier est crypté depuis le navigateur puis injecté dans la base de données. Moi et mes collègues SLAM avons opté pour la méthode suivante qui consiste à récupérer tous les membres ayant un mot de passe non nul et qui ne commence pas par %, afin d'éviter de re hacher ce dernier.
Pour le hash, nous avons utilisé “PASSWORD_DEFAULT” car ce dernier permet d’avoir un hachage à jour en terme d’algorithme bcrypt c’est-à-dire qu’il reçoit des mises à jours régulières afin de renforcer le hash.

GESTION DU BACK OFFICE DU SITE WEB

Hachage des mots de passes des différents membres habilités au back-office réalisé par M.MAILLOT et M.BARRET :

identifiant	mdp
pane	$2y$10$hD5vhZUzVc/r0RSSpbbopeXN8Ji1xg9n...
NULL	NULL
thieumac	$2y$10$lzEUDxPcNW4KGGPb22Q7ce9lCfPX9Gp...
NULL	NULL
NULL	NULL
lenfantm	$2y$10$1tx5aLyyK0PaGZ5e5Hhz0udjZs4T8qzO...

```php
// Se connecter à la base de données
require_once "../connexionBD.php";
$connexion->exec("SET NAMES utf8");

// Récupérer les enregistrements de la table membres
$query = "SELECT id,mdp FROM membres WHERE mdp is not null and mdp not like '\$%'";
$result = $connexion->query($query);
$nbResult = $result->rowCount();

if($nbResult > 0 ){
    // Boucler sur les enregistrements et crypter les mots de passe
    while ($ligne = $result->fetch()) {
    // Générer un nouveau mot de passe haché avec la fonction password_hash
    $nouvPasswordHash = password_hash($ligne['mdp'], PASSWORD_DEFAULT);

    // Mettre à jour la base de données avec le nouveau mot de passe haché
    $updateQuery = "UPDATE membres SET mdp = '$nouvPasswordHash' WHERE id = " . $ligne['id'];
    $connexion->query($updateQuery);
```

Afin d'empêcher des utilisateurs non habilité de rejoindre le back-office le code suivant sera placé dans chaque page afin de vérifier si l'utilisateur possède une session 'user', sinon il sera redirigé sur la page du formulaire de connexion

GESTION DU BACK OFFICE DU SITE WEB

```php
<?php
    session_start();
    if(!isset($_SESSION['user'])){
        header("Location: ../formAuth.php");
        exit();
    }
?>
```

Une fois connecté au back-office, on arrive sur la page suivante :

Ici la session nous permet de véhiculer les informations de l'utilisateur récupérer lors de la connexion, afin d'afficher son nom et prénom.

```php
<?php echo $_SESSION['user']['nom']. " ". $_SESSION['user']['prénom'];?> Bienvenue sur le <br> Back-Office</h1>
```

GESTION DU BACK OFFICE DU SITE WEB

Concernant les accès aux pages du back-office ces derniers sont gérés par les sessions et si le rôle correspond à celui qui est attendu, alors l'utilisateur aura accès aux pages suivantes.

```php
<ul>
    <li><a href="back_office.php">Accueil</a></li>
    <?php
    if($_SESSION['user']['role'] == 'Formation Commmunication Secrétariat'){
        echo '<li><a href="GestionMembres.php">Gestion des Membres</a></li>';
    }
    if($_SESSION['user']['role'] == 'Accueil'){
        echo '<li><a href="GestionNews.php">Gestion des News</a></li>';
    }
    ?>
    <li><a href="logout.php">Exit</a></li>
</ul>
```

Exemple de connexion avec Thieuma:

Exemple de connexion avec Lenfant:

GESTION DU BACK OFFICE DU SITE WEB

Durant le projet, on nous a demandé de réaliser des modes opératoires.
M.MAILLOT et M.HOARAU ont été chargés de réaliser le mode opératoire
Les modes opératoires sont des documents importants qui aident les membres habilités à mieux utiliser les fonctionnalités du back-office. Ils fournissent des instructions étape par étape pour effectuer les tâches courantes, ce qui permet de mieux comprendre les processus et les outils mis à disposition.

Pour aider les membres de la MS2R à prendre en main le back-office, nous avons créé un mode opératoire clair et facile à suivre en utilisant la plateforme Canva. Nous avons pris des captures d'écran pour illustrer les étapes du processus et rendre le document visuellement attrayant.

Nous avons pris le temps de rechercher et comprendre les différentes composantes d'un mode opératoire avant de nous lancer dans la création. Nous voulions nous assurer que le document répondait aux exigences du cahier des charges tout en étant facile à utiliser pour les membres habilités. Donc nous avons crée un seul mode opératoire regroupant les membres habilités qui possèdent différentes fonctions.

Conclusion

DIFFICULTÉS RENCONTRÉES :

Nous avons eu la chance de pouvoir former une équipe, complète il n'y donc que très peu de difficultés techniques. Du côté de la gestion de l'environnement système, nous avons rencontré des problèmes majoritairement avec FTP qui ne fonctionnait que temporairement. Quelques difficultés également sur la réalisation du script, car c'est quelque chose que nous découvrons totalement dans ce projet. Nous avons surtout des difficultés au niveau "Humain". On notera des difficultés pour communiquer entre les deux groupes, voire une certaine incompréhension sur certains points.