

PROJET ANALIM CONGRESSISTE

Mission Hôtellerie et
Session

Equipe Sans nom
Fatima ABOUDA

Projet de Développement d'Application pour la Gestion du Congrès [ANALIM] par la Maison des Ligues de Lorraine

Contexte

La Maison des Ligues de Lorraine (M2L) est un établissement du Conseil Régional dédié à fournir des salles et des services aux ligues sportives régionales et autres structures. Financée entièrement par le Conseil régional, la M2L héberge divers comités départementaux, allant de grandes ligues employant plusieurs personnes à des ligues plus petites sans employés permanents. Dans le cadre de ses activités, la M2L organise en collaboration avec d'autres régions françaises le [Congrès National des Maisons des Ligues] à Limoges, appelé [ANALIM].

Objectifs du Projet

L'objectif principal est de développer une application web pour faciliter l'organisation du congrès [ANALIM]. Cette application doit offrir une souplesse accrue dans la gestion des inscriptions, des sessions, des activités culturelles, de la facturation, et autres aspects liés à l'événement.

Organisation du Congrès [ANALIM] :

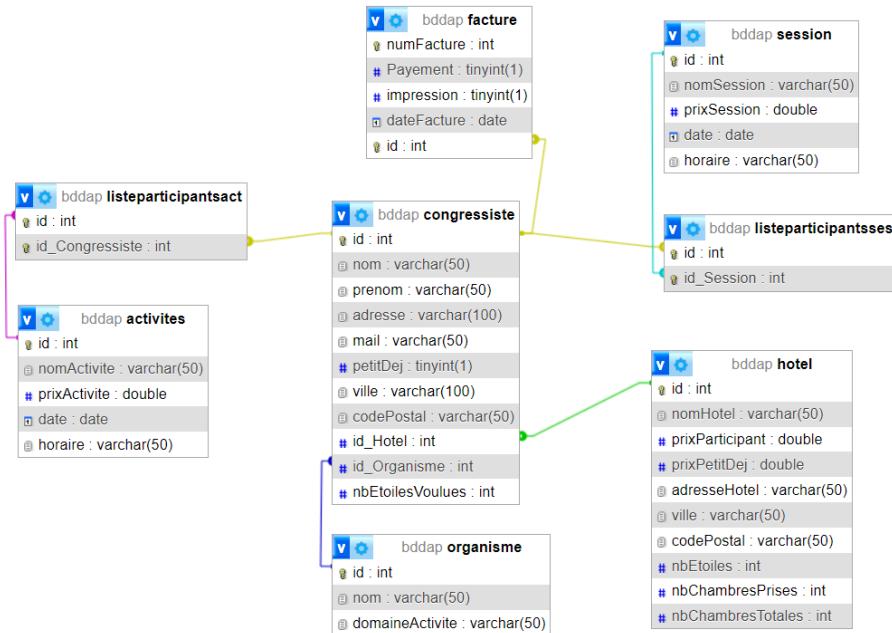
- Les participants s'inscrivent en remplissant une fiche d'inscription.
- Une fois inscrits, les participants reçoivent une confirmation par e-mail.
- Les inscriptions génèrent un dossier pour chaque congressiste.
- Les hôtels proposent des tarifs par participant.
- Chaque session du congrès a un coût particulier.
- Un service d'accueil remet un badge et un dossier aux participants.
- Des activités culturelles sont proposées par le bureau d'animation.
- Les participants peuvent s'inscrire à ces activités jusqu'à 30 minutes avant.
- Les factures doivent être envoyées aux organismes payeurs ou remises aux participants.
- Les règlements par courrier sont enregistrés, et un suivi des factures non réglées est nécessaire.

DESCRIPTION DU PROJET

La Maison des Ligues de Lorraine a confié aux étudiants du BTS SIO du Lycée Suzanne Valadon le développement d'une application complète pour coordonner tous les aspects du congrès [ANALIM]. Chaque membre du groupe s'est vu attribuer une mission spécifique. J'ai opté initialement pour la gestion des Hôtels, et, suite à l'avancement rapide, j'ai également pris en charge la mission suivante, celle de la gestion des Sessions. Ce document technique offre une vue détaillée des étapes suivies, des résultats obtenus, des fonctions implémentées, et plus encore.

Base de Données

Avant de débuter les missions assignées, une étape cruciale a été franchie : la conception et la création d'une base de données. Cette base de données a été élaborée conjointement sur JMerise, puis transférée avec succès sur PHPMyAdmin pour permettre une gestion dynamique des données du congrès [ANALIM]. Voici un aperçu des entités principales et de leurs relations :



1. Table activites :

- Stocke les détails sur les activités proposées pendant le congrès.
- Champs : id, nomActivite, prixActivite, date, horaire.

2. Table congressiste :

- Contient les informations relatives aux congressistes inscrits au congrès.
- Champs : id, nom, prenom, adresse, mail, petitDej, ville, codePostal, id_Hotel, id_Organisme, nbEtoilesVoulues.

3. Table facture :

- Enregistre les détails des factures générées pour les congressistes.
- Champs : numFacture, Payement, impression, dateFacture, id.

4. Table hotel :

- Stocke les informations sur les hôtels disponibles pour l'hébergement des congressistes.
- Champs : id, nomHotel, prixParticipant, prixPetitDej, adresseHotel, ville, codePostal, nbEtoiles, nbChambresPrises, nbChambresTotales.

5. Table listparticipantsact :

- Associe les congressistes aux activités auxquelles ils sont inscrits.
- Champs : id, id_Congressiste, id_Activite.

6. Table listparticipantsses :

- Associe les congressistes aux sessions auxquelles ils sont inscrits.
- Champs : id, id_Congressiste, id_Session.

7. Table organisme :

- Stocke les informations sur les organismes qui paient la participation de certains congressistes.
- Champs : id, nom, domaineActivite.

8. Table session :

- Contient les détails sur les différentes sessions planifiées pour le congrès.

- Champs : id, nomSession, prixSession, date, horaire.

Avant d'entamer ma mission, j'ai créé le contrôleur principal et inclus en tête de la vue le code suivant :

```
<?php
function controleurPrincipal($action)
{
    $lesActions = array();
    $lesActions["defaut"] = "accueil.php";
    $lesActions["hotel"]="hotel.php";
    $lesActions["session"]="session.php";
    if (array_key_exists($action, $lesActions)) {
        return $lesActions[$action];
    } else {
        return $lesActions["defaut"];
    }
}
```

La fonction **controleurPrincipal** agit comme un aiguilleur (router) principal, orientant le flux de l'application vers le fichier approprié en fonction de l'action demandée. Voici quelques corrections et améliorations dans la description :

- La fonction crée un tableau **\$lesActions** associant des actions aux fichiers PHP correspondants. Par exemple, l'action "hotel" est liée au fichier "hotel.php", tandis que l'action "session" est associée à "session.php".
- Elle prend en paramètre une action, généralement fournie dans l'URL ou une autre source, et vérifie si cette action existe dans le tableau **\$lesActions**.
- Si l'action existe, la fonction renvoie le nom du fichier correspondant à cette action. Sinon, elle renvoie le fichier par défaut, généralement "accueil.php".

En résumé, cette fonction **controleurPrincipal** joue un rôle essentiel dans le routage de l'application, en assurant une navigation structurée vers les différents fichiers en fonction des actions spécifiées.

Hôtellerie

Le contrôleur principal envoie au contrôleur hôtel et session qui le plus important pour la suite des explications quand on va au hotel.php cela affiche ceci :

```
<?php
include_once "modele/dataBase.php";
include_once "modele/classHotel.php";

include_once "vue/vueHotel.php";
?>
```

Ce code, inclus dans le fichier **hotel.php**, a pour but d'organiser l'environnement nécessaire à la gestion des hôtels. Voici une description plus détaillée :

- **dataBase.php**: Ce fichier contient les paramètres de connexion à la base de données, assurant ainsi la disponibilité des informations nécessaires pour interagir avec la base de données.
- **classHotel.php**: Il s'agit probablement d'une classe définissant la structure et le comportement des objets "Hôtel". Cette classe peut inclure des méthodes pour récupérer des informations spécifiques sur les hôtels.
- **vueHotel.php**: Ce fichier représente la vue associée aux hôtels. Il pourrait contenir le code HTML et PHP nécessaire pour afficher les détails des hôtels

CLASSE HOTEL

La classe Hotel représente un modèle d'hôtel avec des attributs associés tels que le nom, le prix par participant, le prix du petit déjeuner, l'adresse, la ville, le code postal, le nombre d'étoiles, le nombre de chambres prises, et le nombre total de chambres. Elle offre des méthodes pour accéder et manipuler ces attributs ainsi que pour interagir avec une base de données.

Attributs

- **id (int|null)**: Identifiant unique de l'hôtel.
- **nomHotel (string)**: Nom de l'hôtel.
- **prixParticipant (float)**: Prix par participant.
- **prixPetitDej (float)**: Prix du petit déjeuner.
- **adresseHotel (string)**: Adresse de l'hôtel.
- **ville (string)**: Ville où se situe l'hôtel.
- **codePostal (string)**: Code postal de la ville.
- **nbEtoiles (int|null)**: Nombre d'étoiles attribuées à l'hôtel.
- **nbChambresPrises (int)**: Nombre de chambres actuellement occupées.
- **nbChambresTotales (int)**: Nombre total de chambres dans l'hôtel.

Méthodes

1. Constructeur `__construct`

- **Description** : Initialise une instance de la classe **Hotel** avec les valeurs fournies ou des valeurs par défaut.
- **Paramètres** :

- **\$nomHotel (string)**: Nom de l'hôtel.
- **\$prixParticipant (float)**: Prix par participant.
- **\$prixPetitDej (float)**: Prix du petit déjeuner.
- **\$adresseHotel (string)**: Adresse de l'hôtel.
- **\$ville (string)**: Ville où se situe l'hôtel.
- **\$codePostal (string)**: Code postal de la ville.
- **\$nbEtoiles (int)**: Nombre d'étoiles attribuées à l'hôtel.
- **\$nbChambresPrises (int)**: Nombre de chambres actuellement occupées.

- **\$nbChambresTotales (int)**: Nombre total de chambres dans l'hôtel.

```
class Hotel {
    private ?int $id;
    private string $nomHotel;
    private float $prixParticipant;
    private float $prixPetitDej;
    private string $adresseHotel;
    private string $ville;
    private string $codePostal;
    private ?int $nbEtoiles;
    private int $nbChambresPrises;
    private int $nbChambresTotales;
```

2. Getter et Setter pour chaque attribut

- **Description** : Permet d'accéder et de modifier chaque attribut de la classe.

3. getAllHotels

- Description :** Récupère tous les hôtels depuis la base de données.
- Paramètres :** \$conn (PDO): Connexion à la base de données.
- Retour :** Tableau associatif contenant les détails de tous les hôtels.
- Requête :** SELECT * FROM hotel

```
4. public function
getAllHotels($conn) {

include_once('dataBase.php');
$conn = (new Database())-
>getConnexion();
$result = $conn-
>prepare("SELECT * FROM
hotel");
$result->execute();
return $result-
>fetchAll(PDO::FETCH_ASSOC);
}
```

5. getHotelById

- Description :** Récupère un hôtel par son identifiant depuis la base de données
- Paramètres :**
 - \$conn (PDO): Connexion à la base de données.
 - \$id (int): Identifiant de l'hôtel à récupérer
- Retour :** Tableau associatif contenant les détails de l'hôtel correspondant à l'identifiant.

```
6. public function
getHotelById($conn, $id)

{
include_once('dataBase.php');
$conn = (new Database())-
>getConnexion();

$stmt = $conn-
>prepare("SELECT * FROM hotel
WHERE id = ?");
$stmt->bindValue(1, $id);
$stmt->execute();
return $stmt-
>fetch(PDO::FETCH_ASSOC);
}
```

7. addHotel

- Description :** Ajoute un nouvel hôtel à la base de données.
- Retour :** Message de succès ou d'échec.
- Requête :** INSERT INTO hotel (nomHotel, prixParticipant, prixPetitDej, adresseHotel, ville, codePostal, nbEtoiles, nbChambresTotales) VALUES (?, ?, ?, ?, ?, ?, ?, ?)

```
8. public function addHotel() {
include_once('dataBase.php');
$conn = (new Database())->getConnexion();

// Insérer un nouvel hôtel dans la base de données
$sql = "INSERT INTO hotel (nomHotel, prixParticipant, prixPetitDej,
adresseHotel, ville, codePostal, nbEtoiles, nbChambresTotales) VALUES (?, ?, ?, ?, ?, ?, ?, ?)";
$stmt = $conn->prepare($sql);
$stmt->bindValue(1, $this->nomHotel);
$stmt->bindValue(2, $this->prixParticipant);
$stmt->bindValue(3, $this->prixPetitDej);
$stmt->bindValue(4, $this->adresseHotel);
$stmt->bindValue(5, $this->ville);
$stmt->bindValue(6, $this->codePostal);
$stmt->bindValue(7, $this->nbEtoiles);
$stmt->bindValue(8, $this->nbChambresTotales);

if ($stmt->execute()) {
    return "Succès"; // Ou renvoyez un message de succès
}
```

```

    } else {
        return "Échec"; // Ou renvoyez un message d'erreur
    }
}

```

9. updateHotel

- Description :** Met à jour les informations d'un hôtel dans la base de données.
- Paramètres :** \$id (int): Identifiant de l'hôtel à mettre à jour.
- Retour :** Message de succès ou d'échec.
- Requête :** UPDATE hotel SET nomHotel = ?, prixParticipant = ?, prixPetitDej = ?, adresseHotel = ?, ville = ?, codePostal = ?, nbEtoiles = ? WHERE id = ?

```

public function updateHotel($id)
{
    include_once('dataBase.php');
    $conn = (new Database())->getConnexion();
    $req = $conn->prepare("UPDATE hotel SET nomHotel = ?, prixParticipant = ?, prixPetitDej = ?, adresseHotel = ?, ville = ?, codePostal = ?, nbEtoiles = ? WHERE id = ?");
    $req->bindValue(1, $this->nomHotel);
    $req->bindValue(2, $this->prixParticipant);
    $req->bindValue(3, $this->prixPetitDej);
    $req->bindValue(4, $this->adresseHotel);
    $req->bindValue(5, $this->ville);
    $req->bindValue(6, $this->codePostal);
    $req->bindValue(7, $this->nbEtoiles);
    $req->bindValue(8, $id);
    $req->execute();
}

```

10. deleteHotel

- Description :** Supprime un hôtel de la base de données.
- Paramètres :** \$id (int): Identifiant de l'hôtel à supprimer.
- Retour :** Message de succès ou d'échec.
- Requête :** DELETE FROM hotel WHERE id = ?

```

11. public function deleteHotel($id) {
    include_once('dataBase.php');
    $conn = (new Database())->getConnexion();
    $sql = "DELETE FROM hotel WHERE id = ?";
    $stmt = $conn->prepare($sql);
    $stmt->bindValue(1, $id);

    if ($stmt->execute()) {
        return "Succès"; // Ou renvoyez un message de succès
    } else {
        return "Échec"; // Ou renvoyez un message d'erreur
    }
}

```

Page Web Hotelerie (index.php)

Description

La page web "Hotelerie" (index.php) est une interface utilisateur permettant de gérer des hôtels. Elle affiche une liste d'hôtels existants, offre la possibilité d'ajouter, de supprimer et de modifier des hôtels, et fournit des formulaires pour interagir avec la base de données.

Logique PHP

1. Instanciation des objets :

- Création d'objets **Hotel** et **Database** pour interagir avec les données.

```

$hotels = new Hotel();
$conn = new Database();

```

2. Ajout d'un Nouvel Hôtel :

- Si le formulaire "ajouter" est soumis, récupération des données du formulaire, validation et création d'une nouvelle instance d'**Hotel**.
- Appel de la méthode **addHotel()** pour ajouter l'hôtel à la base de données.

```
3. if (isset($_POST["ajouter"])) {
    $nomHotel = $_POST["nomHotel"];
    $prixParticipant = $_POST["prixParticipant"];
    $prixPetitDej = $_POST["prixPetitDej"];
    $adresseHotel = $_POST["adresseHotel"];
    $codePostal = $_POST["codePostal"];
    $ville = $_POST["ville"];
    $nbEtoiles = $_POST["nbEtoiles"];
    if (!is_numeric($nbEtoiles) || $nbEtoiles < 1 || $nbEtoiles > 5) {
        echo "Nombre d'étoiles invalide. Veuillez choisir un nombre entre
1 et 5.";
        exit();
    }
    $nbChambresPrises = 0;
    $nbChambresTotales = $_POST["nbChambresTotales"];
    $hotel = new Hotel($nomHotel, $prixParticipant, $prixPetitDej,
$adresseHotel, $ville, $codePostal, $nbEtoiles, $nbChambresPrises,
$nbChambresTotales);
    $hotel->addHotel();
}
```

4. Suppression d'un Hôtel :

- Si le formulaire "supprimer" est soumis, récupération de l'ID de l'hôtel à supprimer.
- Appel de la méthode **deleteHotel()** pour supprimer l'hôtel de la base de données.

```
if (isset($_POST["supprimer"])) {
    $id = $_POST["hotelId"];
    $hotel = new Hotel();
    $hotel->deleteHotel($id);
}
```

5. Modification d'un Hôtel :

- Si le formulaire "modifier" est soumis, récupération des données du formulaire, validation et création d'une nouvelle instance d'**Hotel**.
- Appel de la méthode **updateHotel()** pour mettre à jour les informations de l'hôtel dans la base de données.

```
6. if (isset($_POST["modifier"])) {
    $id = $_POST["hotelId"];
    $nomHotel = $_POST["nomHotel"];
    $prixParticipant = $_POST["prixParticipant"];
    $prixPetitDej = $_POST["prixPetitDej"];
    $adresseHotel = $_POST["adresseHotel"];
    $codePostal = $_POST["codePostal"];
    $ville = $_POST["ville"];
    $nbEtoiles = $_POST["nbEtoiles"];

    $hotel = new Hotel($nomHotel, $prixParticipant, $prixPetitDej,
$adresseHotel, $ville, $codePostal, $nbEtoiles);
    $hotel->updateHotel($id);
}
```

7. Affichage du Formulaire de Modification :

- Si le bouton "Modifier" est cliqué pour un hôtel spécifique, affichage d'un formulaire pré-rempli avec les détails de l'hôtel pour modification.

```
8. if (isset($_POST["afficherDetails"])) {
    $id = $_POST["afficherDetails"];
    $hotelDetails = $hotels->getHotelById($conn, $id); // Change $hotel
    to $hotels here
    // Display the details for modification
    if ($hotelDetails) {?>

        <h1>Modifier un hôtel</h1>
        <form action="" method="post">
            <label for="nomHotel">Nom de l'hôtel</label>
            <input type="text" name="nomHotel" id="nomHotel" value="<?=
$hotelDetails['nomHotel'] ?>"><br>
            <label for="prixParticipant">Prix participant</label>
            <input type="text" name="prixParticipant" id="prixParticipant"
value="<?= $hotelDetails['prixParticipant'] ?>"><br>
            <label for="prixPetitDej">Prix petit déjeuner</label>
            <input type="text" name="prixPetitDej" id="prixPetitDej"
value="<?= $hotelDetails['prixPetitDej'] ?>"><br>
            <label for="adresseHotel">Adresse</label>
            <input type="text" name="adresseHotel" id="adresseHotel"
value="<?= $hotelDetails['adresseHotel'] ?>"><br>
            <label for="codePostal">Code postal</label>
            <input type="text" name="codePostal" id="codePostal"
value="<?= $hotelDetails['codePostal'] ?>"><br>
            <label for="ville">Ville</label>
            <input type="text" name="ville" id="ville" value="<?=
$hotelDetails['ville'] ?>"><br>
            <label for="nbEtoiles">Nombre d'étoiles</label>
            <select name="nbEtoiles" id="nbEtoiles">
                <?php for ($i = 1; $i <= 5; $i++) : ?>
                    <option value="<?= $i ?>" <?=
isset($hotelDetails['nbEtoiles']) && $hotelDetails['nbEtoiles'] == $i ?
'selected' : '' ?><?= $i ?> étoiles</option>
                <?php endfor; ?>
            </select>
            <br>
            <input type="hidden" name="hotelId" value="<?=
$hotelDetails['id'] ?>">
            <input type="submit" name="modifier" value="Modifier un
hôtel">
        </form>
        <a href="index.php?action=hotel">Retour à l'ajout</a>
    <?php
}
```

Fatima Abouda	90	1	101 rue babylone	87410	limoges		0	1000	Supprimer
---------------	----	---	------------------	-------	---------	--	---	------	---------------------------

Modifier un hôtel

Nom de l'hôtel	<input type="text" value="Fatima Abouda"/>
Prix participant	<input type="text" value="90"/>
Prix petit déjeuner	<input type="text" value="1"/>
Adresse	<input type="text" value="101 rue babylone"/>
Code postal	<input type="text" value="87410"/>
Ville	<input type="text" value="limoges"/>
Nombre d'étoiles	<input type="text" value="2 étoiles"/>

[Retour à l'ajout](#)

9. Affichage du Formulaire d'Ajout :

- Si aucun formulaire n'est soumis, affichage du formulaire d'ajout d'un nouvel hôtel.

```

} else {?>
    <h1>Ajouter un hôtel</h1>
    <form action="" method="post">
        <label for="">Nom de l'hôtel</label>
        <input type="text" name="nomHotel" id="nomHotel">
        <br>
        <label for="">Prix participant</label>
        <input type="text" name="prixParticipant" id="">
        <br>
        <label for="">Prix petit déjeuner</label>
        <input type="text" name="prixPetitDej" id="">
        <br>
        <label for="">Adresse</label>
        <input type="text" name="adresseHotel" id="">
        <br>
        <label for="">Code postal</label>
        <input type="text" name="codePostal" id="">
        <br>
        <label for="">Ville</label>
        <input type="text" name="ville" id="">
        <br>
        <label for="nbEtoiles">Nombre d'étoiles</label>
        <select name="nbEtoiles" id="nbEtoiles">
            <?php for ($i = 1; $i <= 5; $i++) : ?>
                <option value="<?= $i ?><?= $i ?> étoiles</option>
            <?php endfor; ?>
        </select>
        <br>
        <label for="nbChambresTotales">Nombre de chambres totales</label>
        <input type="text" name="nbChambresTotales" id="">
        <br>
        <input type="submit" name="ajouter" value="Ajouter un hôtel">
    </form>

```

Ajouter un hôtel

Nom de l'hôtel	<input type="text"/>
Prix participant	<input type="text"/>
Prix petit déjeuner	<input type="text"/>
Adresse	<input type="text"/>
Code postal	<input type="text"/>
Ville	<input type="text"/>
Nombre d'étoiles	<input type="text" value="1 étoiles"/>
Nombre de chambres totales	<input type="text"/>
<input type="button" value="Ajouter un hôtel"/>	

Affichage des Hôtels

- Utilisation d'une boucle foreach pour parcourir la liste des hôtels récupérée depuis la base de données.
- Affichage des détails de chaque hôtel dans une ligne du tableau HTML.

```
$hotelList = $hotels->getAllHotels ($conn) ;?>
<h1>Liste des Hôtels</h1>
<table>    <thead>      <tr>
    <th>Nom</th>
    <th>prix du Participant</th>
    <th>prix du Petit Déjeuner</th>
    <th>Adresse</th>
    <th>Code Postal</th>
    <th>Ville</th>
    <th>Nombre d'étoiles</th>
    <th>nombre de chambres prises</th>
    <th>nombre de chambres totales</th>
    <th>Action</th>
</tr>    </thead>    <tbody>
<?php foreach ($hotelList as $hotel) : ?>      <tr>
    <td><?= $hotel['nomHotel'] ?></td>
    <td><?= $hotel['prixParticipant'] ?></td>
    <td><?= $hotel['prixPetitDej'] ?></td>
    <td><?= $hotel['adresseHotel'] ?></td>
    <td><?= $hotel['codePostal'] ?></td>
    <td><?= $hotel['ville'] ?></td>
    <td>
        <?php
            for ($i = 0; $i < $hotel['nbEtoiles']; $i++) {
                echo "★";
            }
        ?>
        </td>
    <td><?= $hotel['nbChambresPrises'] ?></td>
    <td><?= $hotel['nbChambresTotales'] ?></td>
    <td>
        <form action="" method="post">
            <input type="hidden" name="hotelId" value=<?=
$hotel['id'] ?>>
            <button type="submit" name="supprimer" value=<?=
$hotel['id'] ?>>Supprimer</button>
            <button type="submit" name="afficherDetails" value=<?=
$hotel['id'] ?>>Modifier</button>
        </form>
    </td>      </tr>      <?php endforeach ?>      </tbody></table>
```

Liste des Hôtels

Nom	prix du Participant	prix du Petit Déjeuner	Adresse	Code Postal	Ville	Nombre d'étoiles	nombre de chambres prises	nombre de chambres totales	Action
Hotel_Limoges	50	12	12 rue de limoges	87000	Limoges	★★★★★	50	500	Supprimer Modifier
fati	90	12	101 rue babylone	87000	le palais sur vienne	★★★★★	50	500	Supprimer Modifier
Limoges	7	7	7	87410	limoges	★ ★	0	0	Supprimer Modifier
FATIMA	888	1	12 rue de limoges	87000	Limoges	★★★★★	0	400	Supprimer Modifier

Remarques

- Les validations de formulaire sont effectuées côté serveur avec des messages d'erreur appropriés.
- La classe **Hotel** est utilisée pour représenter les données des hôtels et interagir avec la base de données.
- La classe **Database** est utilisée pour obtenir une connexion à la base de données.

Exemples d'Utilisation

1. Ajout d'un Nouvel Hôtel :

- Remplir le formulaire d'ajout avec les détails de l'hôtel et cliquer sur "Ajouter un hôtel".

2. Suppression d'un Hôtel :

- Cliquer sur le bouton "Supprimer" correspondant à l'hôtel à supprimer.

3. Modification d'un Hôtel :

- Cliquer sur le bouton "Modifier" correspondant à l'hôtel à modifier.
- Remplir le formulaire de modification et cliquer sur "Modifier un hôtel".

Conclusion

La page "Hotelerie" offre une interface conviviale pour gérer les données des hôtels, en permettant des opérations telles que l'ajout, la suppression et la modification. Elle utilise une approche orientée objet pour structurer le code et garantir une interaction efficace avec la base de données.

```
<?php
include_once "modele/dataBase.php";
include_once "modele/classHotel.php";

include_once "vue/vueHotel.php";
?>
```

Inclure les fichiers nécessaires (**dataBase.php**, **classSession.php**) et ensuite inclut un fichier de vue (**vueSession.php**). Les fichiers inclus semblent être liés à la gestion des sessions, probablement dans le contexte d'une application web.

1. Inclusion des Fichiers Nécessaires :

- **dataBase.php** : Ce fichier semble être responsable de la connexion à la base de données. Il est probable qu'il contienne la définition d'une classe ou des fonctions pour établir et gérer la connexion à la base de données.
- **classSession.php** : Ce fichier semble contenir la définition de la classe **Session** ou des fonctions associées à la gestion des sessions dans votre application.

2. Inclusion du Fichier de Vue (**vueSession.php**) :

- Le fichier **vueSession.php** est inclus après les fichiers nécessaires. Cela suggère qu'il est utilisé pour la présentation des données liées aux sessions dans l'interface utilisateur.

CLASSE SESSION

Description

La classe **Session** modélise une session d'un événement. Elle gère les informations telles que le nom de la session, le prix, la date, et l'horaire. La classe permet également d'associer des congressistes à une session.

Propriétés

Privées

- **\$id** : Identifiant de la session.
- **\$nomSession** : Nom de la session.
- **\$prixSession** : Prix de la session.
- **\$date** : Date de la session.
- **\$horaire** : Horaire de la session.
- **\$sessionId** : Identifiant de la session (utilisé dans certaines méthodes).
- **\$congressisteId** : Identifiant du congressiste (utilisé dans certaines méthodes).

```
class Session
{
    private $id;
    private $nomSession;
    private $prixSession;
    private $date;
    private $horaire;
    private $sessionId;
    private $congressisteId;
```

Méthodes

Getters

- **getId()** : ?int : Récupère l'identifiant de la session.
- **getNomSession()** : string : Récupère le nom de la session.
- **getPrixSession()** : float : Récupère le prix de la session.
- **getDate()** : string : Récupère la date de la session.
- **getHoraire()** : string : Récupère l'horaire de la session.

Setters

- **setId(\$id)** : Définit l'identifiant de la session.
- **setNomSession(\$nomSession)** : Définit le nom de la session.
- **setPrixSession(\$prixSession)** : Définit le prix de la session.
- **setDate(\$date)** : Définit la date de la session.
- **setHoraire(\$horaire)** : Définit l'horaire de la session.

Constructeur

- **__construct(\$nomSession="", \$prixSession=0, \$date="00-00-000", \$horaire="00-00")**
: Constructeur de la classe **Session**.

Méthodes de Récupération de Données

- **getAllSessions() : array** : Récupère toutes les sessions depuis la base de données.

```
public static function getAllSessions() {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();
    $req = $bdd->prepare("SELECT * FROM session");
    $req->execute();
    $sessions = $req->fetchALL(PDO::FETCH_ASSOC);
    return $sessions;
}
```

- **getSessionById(\$id) : ?array** : Récupère une session par son identifiant.

```
public static function getSessionById($id) {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();
    $req = $bdd->prepare("SELECT * FROM session WHERE id = :id");
    $req->bindParam(':id', $id);
    $req->execute();
    $session = $req->fetch(PDO::FETCH_ASSOC);
    return $session;
}
```

Méthodes de Gestion des Sessions

- **addSession(\$nomSession, \$prixSession, \$date, \$horaire)** : Ajoute une session à la base de données.

```
public static function addSession($nomSession, $prixSession, $date,
$horaire)
{
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();
    $req = $bdd->prepare("INSERT INTO session (nomSession, prixSession,
date, horaire) VALUES (:nomSession, :prixSession, :date, :horaire)");
    $req->bindParam(':nomSession', $nomSession);
    $req->bindParam(':prixSession', $prixSession);
    $req->bindParam(':date', $date);
    $req->bindParam(':horaire', $horaire);
    $req->execute();
}
```

- **updateSession(\$id, \$nomSession, \$prixSession, \$date, \$horaire)** : Met à jour les informations d'une session existante.

```
• public static function updateSession($id, $nomSession, $prixSession,
$date, $horaire) {
    include_once('dataBase.php');

    $bdd = (new Database())->getConnexion();

    $req = $bdd->prepare("UPDATE session SET nomSession = :nomSession,
prixSession = :prixSession, date = :date, horaire = :horaire WHERE id =
:id");
    $req->bindParam(':nomSession', $nomSession);
    $req->bindParam(':prixSession', $prixSession);
    $req->bindParam(':date', $date);
    $req->bindParam(':horaire', $horaire);
    $req->bindParam(':id', $id);
    $req->execute();

}
```

- **deleteSession(\$id)** : Supprime une session de la base de données.

```
• public static function deleteSession($id) {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();
    $req = $bdd->prepare("DELETE FROM session WHERE id = :id");
    $req->bindParam(':id', $id);
    $req->execute();
}
```

Méthodes de Gestion des Participants à une Session

- **addParticipantToSession(\$congressisteId, \$sessionId)** : Ajoute un participant à une session.

```
• public static function addParticipantToSession($congressisteId,
$sessionId) {
    include_once('dataBase.php');

    $bdd = (new Database())->getConnexion();

    $checkReq = $bdd->prepare("SELECT * FROM listeparticipantsses WHERE id =
:congressisteId AND id_Session = :sessionId");
    $checkReq->bindParam(':congressisteId', $congressisteId);
    $checkReq->bindParam(':sessionId', $sessionId);
    $checkReq->execute();

    if ($checkReq->rowCount() == 0) {
        $req = $bdd->prepare("INSERT INTO listeparticipantsses (id,
id_Session) VALUES (:congressisteId, :sessionId)");
        $req->bindParam(':congressisteId', $congressisteId);
        $req->bindParam(':sessionId', $sessionId);
        $req->execute();

        echo "<h3>Enregistrer avec succès</h3>";
    } else {
        echo "<h3>Vous êtes déjà inscrit à cette session</h3>";
    }
}
```

- **cancelInscription(\$congressisteId, \$sessionId)** : Annule l'inscription d'un participant à une session.

```
public static function cancelInscription($congressisteId, $sessionId) {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();

    $checkReq = $bdd->prepare("SELECT * FROM listeparticipantsses WHERE id =
:congressisteId AND id_Session = :sessionId");
    $checkReq->bindParam(':congressisteId', $congressisteId);
    $checkReq->bindParam(':sessionId', $sessionId);
    $checkReq->execute();

    if ($checkReq->rowCount() > 0) {
        $req = $bdd->prepare("DELETE FROM listeparticipantsses WHERE id =
:congressisteId AND id_Session = :sessionId");
        $req->bindParam(':congressisteId', $congressisteId);
        $req->bindParam(':sessionId', $sessionId);
        $req->execute();

        echo "<h3>Annuler avec succès</h3>";
    } else {
        echo "<h3>Vous n'êtes pas inscrit à cette session</h3>";
    }
}
```

Méthodes de Récupération des Congressistes

- **getAllCongressite()** : array : Récupère tous les congressistes.

```
• public static function getAllCongressite() {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();
    $req = $bdd->prepare("SELECT * FROM congressiste");
    $req->execute();
    $congressistes = $req->fetchAll(PDO::FETCH_ASSOC);
    return $congressistes;
}
```

- **getAllCongressistesWithoutInvoice()** : array : Récupère tous les congressistes sans facture associée.

```
• public static function getAllCongressistesWithoutInvoice() {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();

    $req = $bdd->prepare("
        SELECT c.*
        FROM congressiste c
        LEFT JOIN facture f ON c.id = f.id
        WHERE f.numFacture IS NULL
    ");

    $req->execute();
    $congressistes = $req->fetchAll(PDO::FETCH_ASSOC);

    return $congressistes;
}
```

- **getListeParticipants()** : array : Récupère la liste des participants à une session.

```
public static function getListeParticipants() {
    include_once('dataBase.php');
    $bdd = (new Database())->getConnexion();
    $req = $bdd->prepare("SELECT * FROM listeparticipants inner join
        congressiste on listeparticipants.id = congressiste.id inner join
        session on listeparticipants.id_Session = session.id where
        listeparticipants.id = congressiste.id");
    $req->execute();
    $listeparticipants = $req->fetchAll(PDO::FETCH_ASSOC);
    return $listeparticipants;
}
```

Vue Session

Introduction

Cette documentation explique le fonctionnement de la vue **Session**. La vue est destinée à afficher et gérer les sessions d'un événement, ainsi que les inscriptions des congressistes à ces sessions.

Gestion des Soumissions de Formulaire

- Les soumissions de formulaire sont traitées en vérifiant la méthode de requête (POST).
- Les actions (ajouter, supprimer, modifier) sont déclenchées selon les boutons de formulaire correspondants.
- L'objet **Session** est utilisé pour effectuer des opérations sur les sessions en fonction des actions.

```
if ($_SERVER["REQUEST_METHOD"] == "POST") {
    $sessions = new Session("", "", "", "");

    if (isset($_POST["ajouter"])) {
        $sessions->addSession($_POST["nomSession"], $_POST["prixSession"],
        $_POST["date"], $_POST["horaire"]);
    }

    if (isset($_POST["supprimer"])) {
        $sessions->deleteSession($_POST["sessionId"]);
    }

    if (isset($_POST["modifier"])) {
        $id = $_POST["sessionId"];
        $nomSession = $_POST["nomSession"];
        $prixSession = $_POST["prixSession"];
        $date = $_POST["date"];
        $horaire = $_POST["horaire"];
        $sessions->updateSession($id, $nomSession, $prixSession, $date,
        $horaire);
    }
}

// Fetch sessions
$sessions = new Session("", "", "", "");
$sessionList = $sessions->getAllSessions();
$congressisteList = $sessions->getAllCongressistesWithoutInvoice();
$listeInscrits = $sessions->getListeParticipants();
?>
```

Affichage des Sessions

- Les sessions sont récupérées à partir de la base de données en utilisant la méthode **getAllSessions** de la classe **Session**.
- Les sessions sont affichées dans un tableau avec les colonnes : Nom, Prix, Date, Horaire.
- Pour chaque session, des boutons "Modifier" et "Supprimer" sont fournis pour effectuer des actions sur la session.

```

• <h1>Session</h1>
<table>
    <tr>
        <th>Nom</th>
        <th>Prix</th>
        <th>Date</th>
        <th>Horaire</th>
    </tr>
    <?php foreach ($sessionList as $session) { ?>
        <tr>
            <td> <?= $session['nomSession'] ?></td>
            <td> <?= $session['prixSession'] ?></td>
            <td> <?= $session['date'] ?></td>
            <td> <?= $session['horaire'] ?></td>
            <td>
                <form action="" method="post">
                    <input type="hidden" name="sessionId" value="<?=
$session['id'] ?>">
                    <input type="submit" name="AfficherDetail"
value="Modifier">
                    <input type="submit" name="supprimer"
value="Supprimer">
                </form>
            </td>
        </tr>
    <?php } ?>
</table>

```

Session

Nom	Prix	Date	Horaire	
Conférence_1	12	2023-10-18	10:00	<div style="background-color: #f0f0f0; padding: 5px; text-align: center;"> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px; margin-bottom: 5px;" type="button" value="Modifier"/> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px;" type="button" value="Supprimer"/> </div>
Conférence_2	15	2023-12-12	10:49	<div style="background-color: #f0f0f0; padding: 5px; text-align: center;"> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px; margin-bottom: 5px;" type="button" value="Modifier"/> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px;" type="button" value="Supprimer"/> </div>
Conférence_3	20	2023-12-28	15:30	<div style="background-color: #f0f0f0; padding: 5px; text-align: center;"> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px; margin-bottom: 5px;" type="button" value="Modifier"/> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px;" type="button" value="Supprimer"/> </div>
Conférence_4	15	2023-12-23	01:00	<div style="background-color: #f0f0f0; padding: 5px; text-align: center;"> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px; margin-bottom: 5px;" type="button" value="Modifier"/> <input style="width: 100%; height: 30px; background-color: #800080; color: white; border: none; font-size: 10px;" type="button" value="Supprimer"/> </div>

Modification d'une Session

- En cliquant sur le bouton "Modifier" d'une session, les détails de cette session sont récupérés et affichés dans un formulaire pré-rempli.
- Le formulaire permet de modifier les détails de la session sélectionnée.

```
<?php
if (isset($_POST["AfficherDetail"])) {
    $id = $_POST["sessionId"];
    $sessionDetails = $sessions->getSessionById($id);
    if ($sessionDetails) {
?>
        <h1>Modifier une session</h1>
        <form action="" method="post">
            <input type="text" name="nomSession" placeholder="Nom"
value=<?= $sessionDetails['nomSession'] ?>">
            <input type="text" name="prixSession" placeholder="Prix"
value=<?= $sessionDetails['prixSession'] ?>">
            <input type="date" name="date" placeholder="Date" value=<?=
$sessionDetails['date'] ?>">
            <input type="time" name="horaire" placeholder="Horaire"
value=<?= $sessionDetails['horaire'] ?>">
            <input type="hidden" name="sessionId" value=<?=
$sessionDetails['id'] ?>">
            <input type="submit" name="modifier" value="Modifier">
        </form>
        <a href="index.php?action=session">Retour à l'ajout</a>
```

Modifier une session

The screenshot shows a form titled 'Modifier une session'. It contains four input fields: 'Nom' (placeholder 'Conférence_1'), 'Prix' (placeholder '12'), 'Date' (placeholder '18/10/2023'), and 'Horaire' (placeholder '10 : 00'). Below the form is a purple 'Modifier' button.

[Retour à l'ajout](#)

Ajout d'une Session

- Un formulaire est fourni pour ajouter une nouvelle session avec les détails tels que Nom, Prix, Date, Horaire.

```
else {
?>

<h1>Ajouter une session</h1>
<form action="" method="post">
    <input type="text" name="nomSession" placeholder="Nom">
    <input type="text" name="prixSession" placeholder="Prix">
    <input type="date" name="date" placeholder="Date">
    <input type="time" name="horaire" placeholder="Horaire">
    <input type="submit" name="ajouter" value="Ajouter">

    <input type="hidden" name="sessionId" value="">
</form>
<?php } ?>
```

Ajouter une session

The screenshot shows a form titled 'Ajouter une session'. It contains four input fields: 'Nom', 'Prix', 'Date' (placeholder 'jj/mm/aaaa'), and 'Horaire' (placeholder '-- : --'). Below the form is a purple 'Ajouter' button.

Inscription et Annulation d'une Session

- Un formulaire est fourni pour inscrire un congressiste à une session ou annuler son inscription.
- Les congressistes et les sessions disponibles sont affichés dans des listes déroulantes.

```
<h1>Inscription à une Session</h1>

<form action="" method="post">

    <select name="congressisteId">
        <?php foreach ($congressisteList as $congressiste) { ?>
            <option value="= $congressiste['id'] ?&gt;"&gt;&lt;?=
$congressiste['nom'] ?&gt;&lt;/option&gt;
        &lt;?php } ?&gt;
    &lt;/select&gt;
    &lt;select name="sessionId"&gt;
        &lt;?php foreach ($sessionList as $session) { ?&gt;
            &lt;option value="<?= $session['id'] ?&gt;"&gt;&lt;?=
$session['nomSession'] ?&gt;&lt;/option&gt;
        &lt;?php } ?&gt;
    &lt;/select&gt;
    &lt;input type="submit" name="inscrire" value="Inscrire"&gt;
    &lt;input type="submit" name="annuler" value="Annuler Inscription"&gt;

&lt;/form&gt;</pre

```

Inscription à une Session

Liste des Inscrits

- Une table affiche la liste des congressistes inscrits à une session avec les détails de la session.

```
<h2>La liste des inscrits</h2>
<table>
    <tr>
        <th>Nom congressiste</th>
        <th>Nom</th>
        <th>Prix</th>
        <th>Date</th>
        <th>Horaire</th>
    </tr>
    <?php foreach ($listeInscrits as $inscrit) { ?>
        <tr>
            <td> <?= $inscrit['nom'] ?></td>
            <td> <?= $inscrit['nomSession'] ?></td>
            <td> <?= $inscrit['prixSession'] ?></td>
            <td> <?= $inscrit['date'] ?></td>
            <td> <?= $inscrit['horaire'] ?></td>
        </tr>
    <?php } ?>
```

La liste des inscrits

Nom congressiste	Nom	Prix	Date	Horaire
Abric	Conférence_1	12	2023-10-18	10:00
Dalton	Conférence_1	12	2023-10-18	10:00
Abric	Conférence_2	15	2023-12-12	10:49
ABOUDA	Conférence_2	15	2023-12-12	10:49
ABOUDA	Conférence_3	20	2023-12-28	15:30
Abric	Conférence_4	15	2023-12-23	01:00
ABOUDA	Conférence_4	15	2023-12-23	01:00
ABOUDA	Conférence_5	53	2023-12-24	17:21
ABOUDA	Conférence_6	5	2024-01-08	10:44

Ajout d'une Session

- Remplissez le formulaire sous "Ajouter une session" avec les détails appropriés.
- Cliquez sur le bouton "Ajouter".

Modification d'une Session

- Cliquez sur le bouton "Modifier" de la session que vous souhaitez modifier.
- Modifiez les détails dans le formulaire.
- Cliquez sur le bouton "Modifier" dans le formulaire.

Suppression d'une Session

- Cliquez sur le bouton "Supprimer" de la session que vous souhaitez supprimer.
- La session sera supprimée de la base de données.

Inscription à une Session

- Sélectionnez un congressiste et une session dans les listes déroulantes sous "Inscription à une Session".
- Cliquez sur le bouton "Inscrire".

```
if (isset($_POST["inscrire"])) {
    $sessions->addParticipantToSession($_POST["congressisteId"],
    $_POST["sessionId"]);
}
```

Annulation d'une Inscription à une Session

- Sélectionnez un congressiste et une session dans les listes déroulantes sous "Inscription à une Session".
- Cliquez sur le bouton "Annuler Inscription".

```
if (isset($_POST["annuler"])) {
    $sessions->cancelInscription($_POST["congressisteId"],
    $_POST["sessionId"]);
}
```

Conclusion

La vue **Session** offre une interface conviviale pour la gestion des sessions d'un événement, y compris l'ajout, la modification, la suppression et l'inscription des congressistes. Les interactions avec la base de données sont gérées par la classe **Session**. Pour une compréhension approfondie, veuillez-vous référer aux commentaires dans le code source.