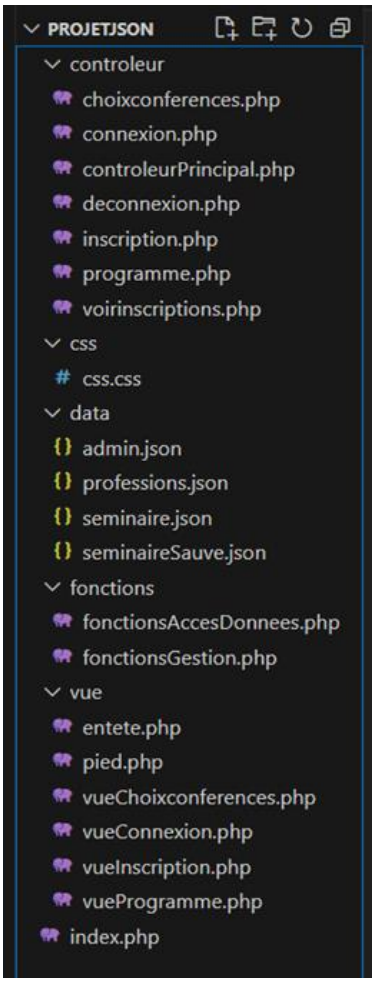
Documentation ProjetJson Séminaire

Atelier Pro

Célia Baylet 2ème année BTS SIO SLAM

[Projet Gestion SEMINAIRE] PHP-MVC-JSON - Projet "Gestion de conférences"

https://wikiwebagnes.fr/doku.php/ap:sio2:ap3.1.1



Présentation du contexte

Le Centre de conférence de la ville de Poitiers accueille régulièrement des conférences ou des séminaires dont les thèmes sont variés.

Un séminaire est prévu le 12 octobre 2222 autour des questions urbaines, il est ouvert aux acteurs des seules municipalités, qui peuvent venir de différents départements. Ce séminaire, durant une seule journée, prévoit différentes conférences qui pour certaines ont lieu sur le même créneau horaire.

Gestion du Projet

Les données sont stockées dans des fichiers JSON ; ainsi, pour utiliser l'application, il ne sera pas nécessaire d'installer une base de données.
Le site Web sera développé avec du code PHP et permet d'exploiter les formulaires de présentation.
Des fonctions permettront d'accéder aux données dans un répertoire dédié. Leurs spécifications (signatures, services) sont présentées sous un format HTML.

Organisation du site

Les participants peuvent s'inscrire et voir le programme. Lorsqu'ils sont inscrits ils peuvent choisir les conférences auxquelles ils souhaitent participer. Ensuite ils peuvent se déconnecter. Il existe une page de connexion pour les administrateurs qui peuvent voir les personnes inscrites aux conférences.

index.php

```php
<?php
include "./controleur/controleurPrincipal.php";

if (isset($_GET["action"])){
    $action = $_GET["action"];
}
else{

    $action = "defaut";
}

$fichier = controleurPrincipal($action);
include "./controleur/$fichier";

?>
```

Index.php - En résumé

- Ce code vérifie si un utilisateur a spécifié une action dans l'URL.
- Si oui, il l'utilise ; sinon, il définit une action par défaut.
- Ensuite, il appelle une fonction de contrôle qui détermine quel traitement exécuter en fonction de l'action.

controleur > controleurPrincipal.php

```php
<?php
function controleurPrincipal($action){
    $lesActions = array();
    $lesActions["defaut"] = "inscription.php";
    $lesActions["programme"] = "programme.php";
    $lesActions["choixconferences"] = "choixconferences.php";
    $lesActions["deconnexion"] = "deconnexion.php";
    $lesActions["validerConnexion"] = "connexion.php";
    $lesActions["voirinscriptions"] = "voirinscriptions.php";

    if (array_key_exists ( $action , $lesActions )){
        return $lesActions[$action];
    }
    else{
        return $lesActions["defaut"];
    }

}

?>
```

controleurPrincipal.php - En résumé

- La fonction controleurPrincipal prend une action en paramètre et vérifie si elle est définie dans un tableau d'actions.

- Si l'action est trouvée, elle retourne le fichier associé. Sinon, elle retourne un fichier par défaut.
- Cela permet de gérer la navigation dans l'application en redirigeant l'utilisateur vers le bon fichier en fonction de l'action qu'il souhaite réaliser.

vue > entete.php

```php
<!DOCTYPE html>
<html>
    <head>
        <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
        <link href="css/css.css" rel="stylesheet" type="text/css">
        <title></title>
    </head>
<?php
include_once "fonctions/fonctionsAccesDonnees.php";
include_once "fonctions/fonctionsGestion.php";
?>
<body>
    <h1><?php echo donnerIntituleSeminaire()?></h1>
    <table>
        <tr>
            <td><a href="./?action=inscription">Inscription</a></td>
            <td><a href="./?action=programme">Voir le programme</a></td>
            <td><a href="./?action=choixconferences">Choisir ses conférences</a></td>
            <td><a href="./?action=deconnexion">Deconnexion</a></td>
            <td><a href="./?action=validerConnexion">Connexion admin</a></td>
            <td><a href="./?action=voirinscriptions">Voir les inscrits aux conférences</a></td>
        </tr>
    </table>
```

Entete.php - En résumé

L'entête prépare le document pour être correctement affiché et stylisé. Les inclusions PHP permettent d'accéder à des fonctions utiles. Les liens dans le corps de la page utilisent le paramètre action dans l'URL pour diriger l'utilisateur vers différentes pages ou fonctionnalités de l'application. Chaque lien correspond à une action qui déclenche une logique spécifique dans le système, facilitant ainsi la navigation.

Les liens dans l'entête génèrent des URL contenant des paramètres d'action qui dirigent l'utilisateur vers différentes pages ou fonctionnalités de l'application.

→Partie 1 Gestion des inscriptions à un séminaire (Programme du séminaire et saisie des inscriptions)

Il fallait tout d'abord écrire la ***function donnerIntituleSeminaire()*** qui retourne l'intitulé du séminaire (dans la page fonctionsAccesDonnees.php).

```
function donnerIntituleSeminaire()
{
    $document = chargeJSONseminaire();
    return $document->seminaire->intitule;
}
```

La ligne 3 appelle la fonction chargeJSONseminaire(), qui charge le document JSON et le décode en un objet PHP. Le résultat de cette fonction est stocké dans la variable $document.

*La ligne 4 accède à la propriété intitule de l'objet seminaire contenu dans $document. Elle retourne la valeur de intitule, qui est le titre du séminaire. («**Seminaire du 12 octobre 2222, la ville et ses enjeux**»)*

-Etape1 Programme du séminaire

10h	Intervenant	Salle
Enjeux émergents, enjeux à construire pour la recherche urbaine	Alain Lecastel	salle Mozart
L'activité urbaine et l'environnement: de nouveaux modèles à construire	Yves Renard	salle Berlioz
Des infrastructures et des services urbains pour une ville économe?	Anne-Marie Brelinsky	salle Beethoven

11h30	Intervenant	Salle
Métropoles et aménagements métropolitains	Joseph Demongeot	salle Berlioz
La ville pour tous	Marc Touati	salle Mozart
Quelles dynamiques d'innovations pour les villes ?	Hervé Bachelard	salle Beethoven

14h	Intervenant	Salle
Urbanisme et environnement: des exemples en Europe	Yann Berlinne	salle Berlioz
Nouvelles vulnérabilités, nouvelles gouvernances	Hamed Djalaoui	salle Beethoven

16h	Intervenant	Salle
La ville et ses périphéries, quelles relations ?	Omar Azdine	salle Berlioz
La ville, quel avenir, quelles contraintes ?	Sophie Radon	salle Beethoven

On cherche à afficher le programme sous forme de tableau.

Il faut ecrire la fonction ***function donnerLesHeuresCreneaux()*** qui retourne la liste de tous les créneaux horaires (heures de début) sous forme d'un tableau (le tableau retourné commence à l'indice 0). (dans la page fonctionsAccesDonnees.php).

```
function donnerLesHeuresCreneaux(){
    $document = chargeJSONseminaire();
    foreach($document->seminaire->creneau as $valeur){
    $ListeHeure[] = $valeur->heure;
    }
    return $ListeHeure;
}
```

Ligne 3 : La boucle foreach, permet de parcourir chaque élément du tableau $document->seminaire->creneau. Pour chaque élément, on le nomme $valeur.

Ligne 4 : Dans la boucle, on extrait la propriété heure de chaque $valeur (qui représente un créneau) et on l'ajoute à un tableau $ListeHeure.

A la ligne 6, la fonction renvoie le tableau $ListeHeure, qui contient toutes les heures des créneaux.

Dans la page fonctionsAccesDonnees.php, il faut écrire la fonction **function donnerLesConferences($heure)** qui retourne toutes les conférences (id, description, intervenant, salle et nbplaces) commençant à l'heure donnée sous forme d'un tableau.

```
function donnerLesConferences($heure){
    $document = chargeJSONseminaire();
    foreach($document->seminaire->creneau as $creneau){
        if ($creneau->heure == $heure){
            foreach($creneau->conference as $conference){
                $tabConf[] = $conference;
            }
        }
    }
    return $tabConf;
```

Ligne 3 : boucle foreach pour parcourir chaque élément du tableau $document->seminaire->creneau. Chaque élément de la boucle est désigné par la variable $creneau.

Ligne 4 : Ici, on vérifie si l'heure du créneau ($creneau->heure) est égale à l'heure passée en paramètre à la fonction ($heure). Si c'est le cas, le bloc de code suivant sera exécuté.

Ligne 5 : Une nouvelle boucle foreach commence, parcourant chaque élément de $creneau->conference. Chaque élément de la boucle est désigné par la variable $conference.

Ligne 6 : Dans cette boucle, chaque $conference est ajouté au tableau $tabConf.

Ligne 10 : La fonction renvoie le tableau $tabConf, qui contient toutes les conférences correspondant à l'heure spécifiée.

Il faut maintenant compléter la page programme.php qui permet d'afficher le programme du séminaire avec la liste de toutes les conférences en utilisant les fonctions écrites précédemment.

```php
<?php
$heures = donnerLesHeuresCreneaux();


foreach($heures as $heure){
    echo "<table>";
        echo"<th>$heure</th>";
        echo"<th>Intervenant</th>";
        echo"<th>Salle</th>";
        $LesConferences = donnerLesConferences($heure);
        foreach ($LesConferences as $UneConference){
        echo"<tr>";
            echo"<td>$UneConference->description</td>";
            echo"<td>$UneConference->intervenant</td>";
            echo"<td>$UneConference->salle</td>";
        echo"</tr>";
        }
    echo"</table>";
}
?>
```

Ligne 2 : boucle foreach pour parcourir chaque élément du tableau $heures. Chaque heure est désignée par la variable $heure.

On fait un tableau avec <table>. On crée trois en-têtes de colonne (<th>) dont une particulière avec la valeur de $heure. Cela affichera l'heure dans le tableau.

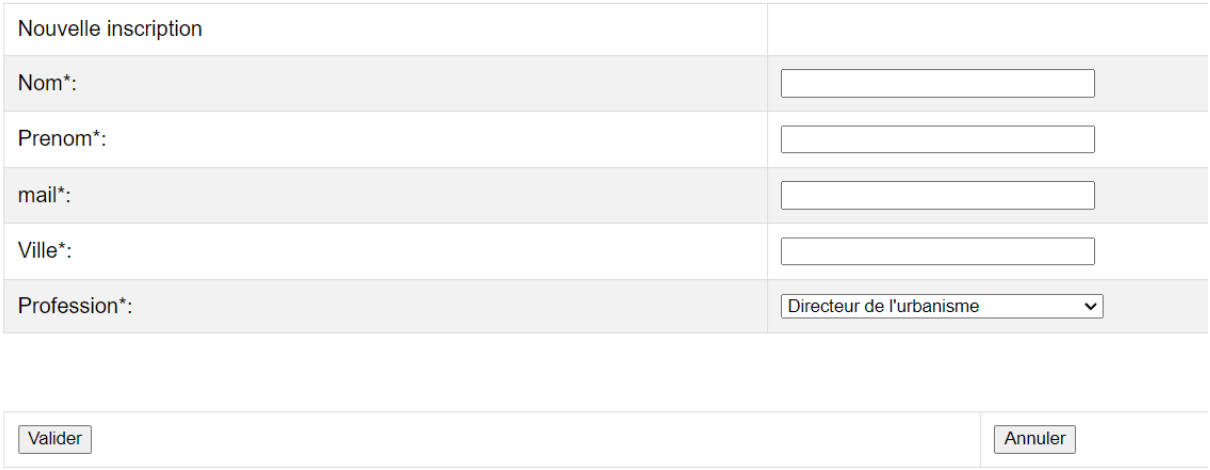
On appelle la fonction créée précédemment donnerLesConferences avec l'heure ($heure). Cette fonction retourne un tableau de conférences qui se déroulent à cette heure. Ce tableau est stocké dans la variable $LesConferences.

Ligne 8 : On commence une nouvelle boucle foreach pour parcourir chaque élément de $LesConferences. Chaque conférence est désignée par la variable $UneConference.

On ouvre une ligne du tableau <tr>. On crée les cellules du tableau (<td>) contenant description, intervenant et salle de la conférence.

-Etape2 Inscription des visiteurs

Pour s'inscrire, le visiteur doit, dans un premier temps répondre à un questionnaire.



Page entete.php : **Inscription**

Dans la page fonctionsGestion.php on ajoute une fonction de gestion des erreurs. Les champs suivis de * sont obligatoires. Un message d'erreur apparait si le format de l'email est invalide.

On complète la page inscription.php afin d'y intégrer la gestion des erreurs et la vérification des données saisies. Il faut intégrer les fonctions 'ajouterErreur', 'donnerNbErreurs' et afficherErreurs'.

Ces 3 fonctions sont dans fonctionsGestion.php :

```php
/**
 * @access private
 * @param type $msg
 */
function ajouterErreur($msg)
{
   if (! isset($_GET['erreurs']))
      $_GET['erreurs']=array();
   $_GET['erreurs'][]=$msg;
}
/**
 * Retourne le nombre de messages d'erreurs de saisie
 * @return entier
 */
function donnerNbErreurs()
{
   if (!isset($_GET['erreurs']))
   {
      return 0;
   }
   else
   {
      return count($_GET['erreurs']);
   }
}
/**
 * Affiche toutes les erreurs de saisie
 */
function afficherErreurs()
{
   echo '<div>';
   echo '<ul>';
   foreach($_GET['erreurs'] as $erreur)
   {
      echo "<li>$erreur</li>";
   }
   echo '</ul>';
   echo '</div>';
}
```

Fonction ajouterErreur :

Prend un paramètre $msg, qui représente un message d'erreur à ajouter.
On vérifie si la clé 'erreurs' n'existe pas dans la superglobale $_GET. Si elle n'existe pas, on l'initialise en tant que tableau vide.
On ajoute le message d'erreur $msg à la fin du tableau $_GET['erreurs']

Fonction donnerNbErreurs :

On vérifie si la clé 'erreurs' n'existe pas dans la superglobale $_GET. La fonction isset() renvoie false si la variable n'est pas définie. Si 'erreurs' n'existe pas, la fonction retourne 0, indiquant qu'il n'y a aucune erreur à signaler. Sinon la fonction utilise count() pour compter le nombre d'éléments dans le tableau $_GET['erreurs'] et retourne ce nombre. Cela donne le nombre d'erreurs enregistrées.

Fonction afficherErreurs :

Affiche une balise HTML , qui sert à regrouper les éléments à l'intérieur. Affiche une balise HTML <ul> (liste non ordonnée) pour contenir les messages d'erreur sous forme de liste.
Une boucle foreach pour parcourir chaque message d'erreur dans le tableau $_GET['erreurs']. Chaque message d'erreur est stocké dans la variable $erreur.
Pour chaque erreur, elle affiche un élément de liste <li> contenant le message d'erreur.
Cela place chaque message d'erreur sur une nouvelle ligne dans la liste.

```
$btn = "Inscription";
if (isset($_POST["btn"])){
   $btn = $_POST["btn"];
   $nom = $_POST["nom"];
   $prenom = $_POST["prenom"];
   $mail = $_POST["mail"];
   $ville = $_POST["ville"];
   $profession = $_POST["profession"];
}
switch ($btn){
   case "Annuler" :
      $nom = ' ';
      $prenom = ' ';
      $mail = ' ';
      $ville = ' ';
      break;

   case "Valider" :
      verifierDonneesInscription($nom, $prenom, $mail, $ville);
      if (donnerNbErreurs()  !=  0) {
         afficherErreurs();
         break;
      }

      sauverDonneesInscription($nom, $prenom, $mail, $ville, $profession);
      echo "<h2>Votre inscription a été prise en compte, il faut procéder au
choix des conférences</h2>";
         break;
}
```

Ligne 2 : On vérifie si le bouton nommé "btn" a été soumis via un formulaire. La fonction isset() détermine si la variable existe et n'est pas nulle.

Ligne 3 : Si le bouton a été soumis, on affecte sa valeur à la variable $btn

Ligne 4 à 8 : Ces lignes récupèrent les valeurs soumises par l'utilisateur via le formulaire. Chaque champ (nom, prénom, mail, ville, profession) est stocké dans sa propre variable.

Ligne 10 : On commence une structure de contrôle switch qui va exécuter différents blocs de code en fonction de la valeur de $btn.

Ligne 11 à 16 : Ce bloc est exécuté si $btn est égal à "Annuler". Les variables $nom, $prenom, $mail, et $ville sont réinitialisées à des chaînes vides. Cela est utilisé pour effacer les informations du formulaire. Le break termine ce cas.

Ligne 17 : Ce bloc est exécuté si $btn est égal à "Valider".

Ligne 18 : Appelle la fonction verifierDonneesInscription pour vérifier les données fournies par l'utilisateur.

Ligne 19-20 : On vérifie s'il y a des erreurs en appelant la fonction donnerNbErreurs(). Si le nombre d'erreurs est différent de 0, cela signifie qu'il y a des erreurs de validation. Si des erreurs sont trouvées la fonction afficherErreurs() va afficher ces erreurs.

La fonction sauverDonneesInscription doit d'abord être créée avant d'être inclue dans la page inscription.

Si aucune erreur n'est trouvée, on appelle la fonction sauverDonneesInscription pour sauvegarder les données de l'utilisateur (nom, prénom, mail, ville et profession). On affiche un message de confirmation à l'utilisateur.

Les données d'inscription saisies dans le formulaire seront stockées dans des variables de session (rappel $_SESSION) afin de pouvoir gérer la saisie du choix des conférences.

Dans la page FonctionsAccesDonnees .php il faut ecrire la fonction **function sauverDonneesInscription($nom, $prenom,$mail,$ville, $profession)**.

```
function sauverDonneesInscription($nom, $prenom,$mail,$ville, $profession){
    session_start();
    $_SESSION['nom'] = $nom;
    $_SESSION['prenom'] = $prenom;
    $_SESSION['mail'] = $mail;
    $_SESSION['ville'] = $ville;
    $_SESSION['profession'] = $profession;
    print_r($_SESSION);
}
```

La fonction prend cinq paramètres : $nom, $prenom, $mail, $ville et $profession. Ils représentent les informations d'inscription d'un utilisateur.

La fonction session_start() est appelée pour démarrer une nouvelle session ou reprendre une session existante. Cela permet de stocker des données spécifiques à l'utilisateur sur le serveur.

Chaque paramètre d'entrée est stocké dans la superglobale $_SESSION, ce qui permet de conserver ces valeurs pendant la durée de la session de l'utilisateur. Le nom, le prénom, l'adresse mail, la ville et la profession de l'utilisateur sont stocké.

La fonction print_r($_SESSION) est utilisée pour afficher le contenu du tableau $_SESSION. Cela sert pour le débogage, car cela permet de visualiser les données stockées dans la session.

```
Array ( [nom] => durand [prenom] => marcel [mail] => marcel.durand@gmail.com [ville] => Bordeaux [profession] => Technicien territorial )
```

Il faut ecrire la fonction **function donnerLesProfessions()** qui retourne toutes les professions possibles dans un tableau (les professions sont inscrites dans le fichier professions.json). Cette fonction permettra de charger la liste déroulante des professions dans le formulaire.


```
function donnerLesProfessions(){
    $document = chargeJSONprofessions();
    return $document->professions;
}
```

On appelle une fonction nommée chargeJSONprofessions(), qui charge la liste de professions. Le résultat de cette fonction est stocké dans la variable $document. La fonction renvoie une liste des professions qui ont été chargées.

On écrit la fonction sauverDonneesInscription dans la page inscription.php (voir ci-dessus)

-Etape3 Choix des conférences

Le visiteur inscrit peut ensuite procéder à la sélection de ses conférences.

Page entete.php : **Choisir ses conférences**

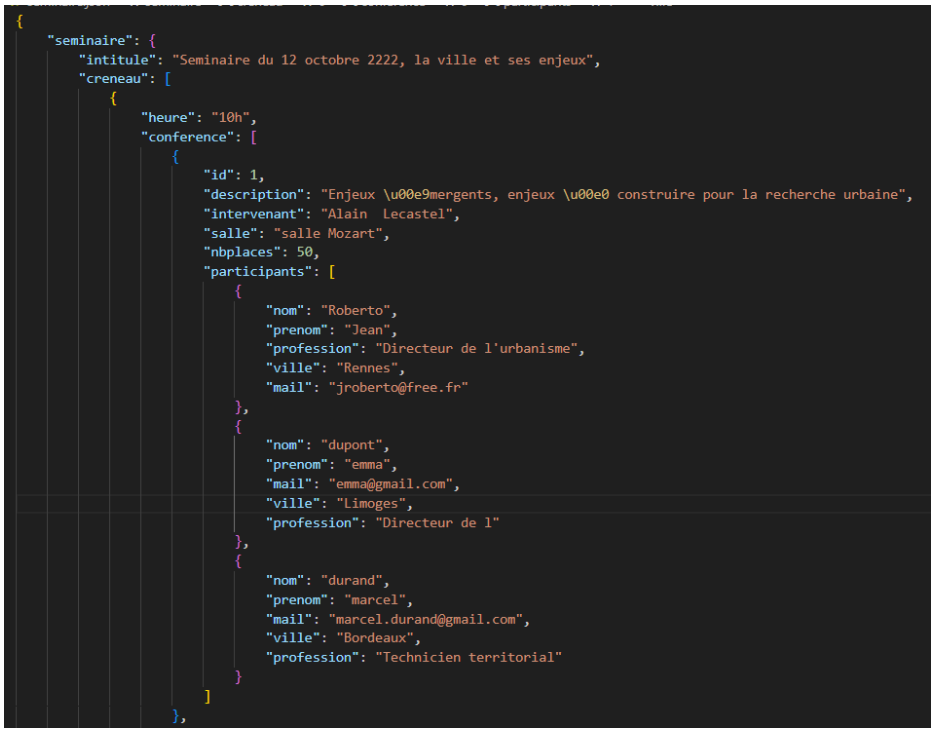
10h		
Enjeux émergents, enjeux à construire pour la recherche urbaine	salle Mozart	◉
L'activité urbaine et l'environnement: de nouveaux modèles à construire	salle Berlioz	○
Des infrastructures et des services urbains pour une ville économe?	salle Beethoven	○
11h30		
Métropoles et aménagements métropolitains	salle Berlioz	○
La ville pour tous	salle Mozart	◉
Quelles dynamiques d'innovations pour les villes ?	salle Beethoven	○
14h		
Urbanisme et environnement: des exemples en Europe	salle Berlioz	○
Nouvelles vulnérabilités, nouvelles gouvernances	salle Beethoven	◉
16h		
La ville et ses périphéries, quelles relations ?	salle Berlioz	◉
La ville, quel avenir, quelles contraintes ?	salle Beethoven	○

valider annuler

Il peut seulement choisir 1 conférence par heure. Il ne peut pas assister à 2 conférences qui ont lieu en même temps.

Il faut compléter la page **choixconferences.php** afin de réaliser l'affichage de l'ensemble des conférences et des salles pour chacun des créneaux suivant le modèle ci-dessous. (à ce moment la les lignes 5 a 10 nous intéressent).

Array ([nom] => durand [prenom] => marcel [mail] => marcel.durand@gmail.com [ville] => Bordeaux [profession] => Technicien territorial) Array ([action] => choixconferences [10h] => 1 [11h30] => 5 [14h] => 8 [16h] => 9 [bouton] => valider)



Le participant doit être enregistré dans le fichier seminaire.json.

```
session_start();


if(estInscrit() == false){
    header("Location: ./?action=inscription.php");
}


print_r($_SESSION);
$btn = "choixconferences";
if (isset($_POST["bouton"])){
   $btn = $_POST["bouton"];
   $lesChoix = $_POST;
}


switch ($btn){
    case "annuler" :
       break;


    case "valider" :
        print_r($_REQUEST);
        enregistre($lesChoix);
        break;
}
```

La fonction session_start() est appelée pour démarrer une session. Cela permet d'accéder aux données stockées dans la superglobale $_SESSION.

Ligne 2 à 4 : On vérifie si l'utilisateur est inscrit en appelant la fonction estInscrit(). Si la fonction retourne false, l'utilisateur n'est pas inscrit. Dans ce cas, on redirige l'utilisateur vers la page d'inscription en utilisant header(). Cela envoie un en-tête HTTP pour rediriger l'utilisateur vers une autre URL.

//fonction estInscrit() est écrite un peu plus tard. (ci-dessous)

print_r($_SESSION) est utilisée pour afficher le contenu de la variable de session, pour le débogage.

Ligne 6 : Une variable $btn est initialisée avec la valeur "choixconferences". Cela sert de valeur par défaut au cas où le formulaire n'aurait pas été soumis.

Ligne 7 : On vérifie si un bouton nommé "bouton" a été soumis via un formulaire. La fonction isset() détermine si la variable existe et n'est pas nulle.

Ligne 8-10 : Si le bouton a été soumis, on affecte sa valeur à la variable $btn et on stocke toutes les données du formulaire dans la variable $lesChoix.

On commence une structure de contrôle switch, qui va exécuter différents blocs de code en fonction de la valeur de $btn.

Si $btn est égal à « annuler », le break termine ce cas du switch, aucune action n'est effectuée, si l'utilisateur choisit « annuler » rien ne se passe.

Si par contre l'utilisateur choisit le bouton « valider » :

La fonction print_r($_REQUEST) affiche le contenu de la superglobale $_REQUEST, qui contient les données des formulaires envoyées via $_POST. Cela est utile pour le débogage.

On appelle la fonction enregistre, en lui passant $lesChoix, qui contient les choix faits par l'utilisateur dans le formulaire. Cette fonction enregistre les choix de l'utilisateur dans le fichier json : seminaire.json

Dans la page fonctionsAccesDonnees.php : Il faut ecrire la fonction **function enregistre($lesChoix)** qui enregistre un participant et ses choix de conférences dans le fichier seminaire.json.

```
function enregistre($lesChoix){
    $document = chargeJSONseminaire();
    foreach ($lesChoix as $choix) {
        foreach ($document->seminaire->creneau as $creneau)
{
            foreach ($creneau->conference as $conference) {
                if ($conference->id == $choix) {
                    if (!isset($conference->participants)) {
                        $conference->participants = [];
                    }
                    $participant = [
                        'nom' => $_SESSION['nom'],
                        'prenom' => $_SESSION['prenom'],
                        'mail' => $_SESSION['mail'],
                        'ville' => $_SESSION['ville'],
                        'profession' => $_SESSION['profession']
                    ];
                    $conference->participants[] = $participant;
                }
            }
        }
    }
    sauveJSONseminaire($document);
```

On commence une boucle foreach pour parcourir chaque élément dans $lesChoix. Chaque élément (l'identifiant d'une conférence) est désigné par la variable $choix.

On commence une nouvelle boucle pour parcourir chaque créneau ($creneau) dans la propriété creneau de l'objet $document->seminaire.

On commence une troisième boucle pour parcourir chaque conférence ($conference) dans le créneau actuel.

On vérifie si l'identifiant de la conférence ($conference->id) correspond au choix actuel ($choix). Si c'est le cas, le bloc suivant sera exécuté.

Ligne 7-8 : On vérifie si la propriété participants n'est pas déjà définie pour la conférence. Si elle n'existe pas, on l'initialise comme un tableau vide. Cela permet de s'assurer qu'on a un tableau pour stocker les participants.

Ligne 10-15 : On crée un tableau associatif $participant contenant les informations du participant, extraites de la session ($_SESSION). Cela comprend le nom, prénom, mail, ville et profession de l'utilisateur.

Ligne 16 : On ajoute le tableau $participant au tableau participants de la conférence. Cela enregistre le participant à la conférence en cours. (tableau dans le fichier seminaire.json)

Après avoir ajouté tous les participants aux conférences, on appelle la fonction sauveJSONseminaire($document) pour sauvegarder les données mises à jour dans le fichier ou la source JSON.

Il faut ecrire la fonction **function estInscrit()** qui vérifie si le visiteur a déjà rempli son formulaire d'inscription et retourne un booléen (test de l'existence d'une variable SESSION).

```
function estInscrit(){
    if (isset($_SESSION['nom']) && isset($_SESSION['prenom']) && isset($_SESSION['mail']) &&
isset($_SESSION['ville']) && isset($_SESSION['profession'])){
        return true;
    }
    else{
        return false;
    }
}
```

On utilise une instruction if pour vérifier si toutes les clés nécessaires (nom, prénom, mail, ville, profession) existent dans la superglobale $_SESSION. La fonction isset() détermine si une variable est définie et n'est pas nulle. L'utilisation de && signifie que toutes ces conditions doivent être vraies pour que le bloc suivant soit exécuté.

Si toutes les conditions de l'instruction if sont vraies, la fonction retourne true, indiquant que l'utilisateur est inscrit.

Si l'une des conditions de l'instruction if n'est pas remplie, le bloc else est exécuté et la fonction retourne false, indiquant que l'utilisateur n'est pas inscrit.

Il fallait compléter la page choixconferences.php afin d'y intégrer l'enregistrement des choix du visiteur dans le fichier seminaire.json. Cette page est complétée et expliquer ci-dessus (au début de l'étape3).

→Partie 2 Exploitation des inscriptions à un séminaire (Connexion et visualisation des inscriptions)

On doit commencer par compléter le fichier entete.php pour ajouter un bouton de Connexion admin et un autre pour visualiser les personnes inscrites aux conférences.

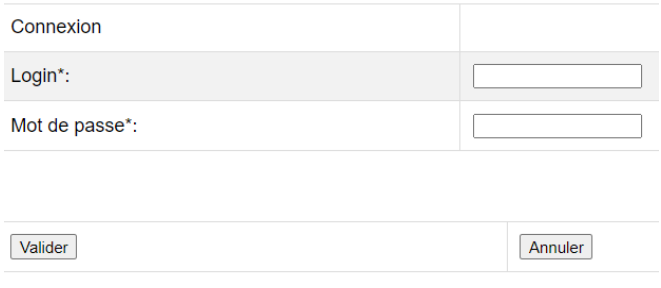
```
<td><a href="./?action=validerConnexion">Connexion admin</a></td>
<td><a href="./?action=voirinscriptions">Voir les inscrits aux conférences</a></td>
```

Connexion admin	Voir les inscrits aux conférences

-Etape1 Connexion

Pour visualiser les inscriptions, il faut être autorisé et commencer par s'identifier : seuls certains utilisateurs peuvent accéder à cette partie du site.
Actuellement, deux comptes permettent de se connecter : admin/admin et btssio/btssio.

fichier admin.json



Page de connexion administrateur

Dans la page fonctionsAccesDonnees.php : Ecrire la fonction function chargeJSONadmin() qui permet de tranférer le contenu du fichier admin.json dans un objet contenant l'arborescence du fichier JSON.

```
function chargeJSONadmin()
{
    $json_source = file_get_contents('data/admin.json');
    $document = json_decode($json_source);
    return $document;
}
```

On utilise la fonction file_get_contents() pour lire le contenu du fichier admin.json, qui est situé dans le répertoire data. Le contenu du fichier JSON est stocké dans la variable $json_source. Cette fonction retourne le contenu du fichier sous forme de chaîne de caractères.

La fonction json_decode() est appelée pour convertir la chaîne JSON contenue dans $json_source en un objet PHP. Le résultat est stocké dans la variable $document.

La fonction retourne l'objet $document qui contient les données chargées à partir du fichier JSON.

Dans la page fonctionsGestions.php :

Ecrire la fonction **function verifier($login,$mdp)** qui permet de vérifier le login et le mot de passe afin d'autoriser ou non le visiteur à visualiser les inscriptions.

Si le login et mot de passe correspondent à l'admin, on chargera une variable de session $_SESSION['admin']=1.

```
function verifier($login,$mdp){
   $document = chargeJSONadmin();
   foreach($document->users as $users){
      if ($users->login == $login && $users->mdp == $mdp){
         $_SESSION['admin']=1;
         return true;
      }
}
}
```

Elle prend deux paramètres : $login et $mdp.

On appelle la fonction chargeJSONadmin() pour charger les données à partir du fichier JSON contenant les informations administratives. Le résultat est stocké dans la variable $document.

On commence une boucle foreach pour parcourir chaque utilisateur ($users) dans la propriété users de l'objet $document. Cela permet d'examiner chaque utilisateur stocké dans le fichier JSON.

On vérifie si le login et le mdp de l'utilisateur actuel correspondent aux valeurs fournies en paramètres.

Si les identifiants sont corrects, on définit une variable de session $_SESSION['admin'] avec la valeur 1. Cela indique que l'utilisateur est connecté en tant qu'administrateur.

La fonction retourne true, ce qui signifie que l'authentification a réussi.

Toujours dans la page fonctionsGestions.php il faut ecrire la fonction **function estAdmin()** qui retourne vrai si le visiteur est un administrateur connecté et autorisé à visualiser les inscriptions.

```
function estAdmin(){
    if (isset($_SESSION['admin']) && $_SESSION['admin']==1){
        return true;
    }
    else{
        return false;
    }
}
```

On utilise une instruction if pour vérifier deux conditions :

- ➔ *isset($_SESSION['admin']) : Cela vérifie si la variable de session admin est définie.*
- ➔ *$_SESSION['admin'] == 1 : Cela vérifie si la valeur de $_SESSION['admin'] est égale à 1. Cela signifie que l'utilisateur est identifié comme administrateur.*

Si les deux conditions de l'instruction if sont vraies, la fonction retourne true, indiquant que l'utilisateur a des privilèges d'administrateur.

Il faut compléter la page connexion.php afin d'y intégrer la gestion de la connexion et la vérification des données saisies. Vous utiliserez les fonctions utiles écrites précédemment.

```
if (isset($_POST["btn"])){
    $login = $_POST["login"];
    $mdp = $_POST["mdp"];
    $admin = verifier($login,$mdp);

    if ($admin != false){
        session_start();
        $_SESSION["admin"] = $admin;
        $_SESSION["login"] = $login;
        $_SESSION["mdp"] = $mdp;

        echo "Vous êtes connectés en tant qu'administrateur";
        print_r($_SESSION);
    }
    else{
        echo "Login ou mot de passe incorrect";
    }
}
```

Ligne 1 :On vérifie si le bouton a été soumis via un formulaire. Cela permet de déterminer si l'utilisateur a essayé de se connecter.

Ligne 2-3 : Si le bouton a été soumis, les valeurs des champs de formulaire "login" et "mdp" (mot de passe) sont extraites de la superglobale $_POST et stockées dans les variables $login et $mdp.

Ligne 4 : La fonction verifier est appelée avec les identifiants fournis ($login et $mdp). Cette fonction vérifie si les identifiants correspondent à un utilisateur dans le fichier JSON. Le résultat (qui est soit true, soit false) est stocké dans la variable $admin.

Ligne 5 : On vérifie si la variable $admin n'est pas false. Si $admin est vrai (c'est-à-dire que les identifiants sont corrects), le bloc suivant sera exécuté.

Ligne 6 : La fonction session_start() est appelée pour démarrer une nouvelle session ou reprendre une session existante. Cela permet d'accéder aux données de session.

On stocke dans la variable de session :

$_SESSION["admin"] : La valeur de $admin (qui devrait être true).

$_SESSION["login"] : Le nom d'utilisateur fourni.

$_SESSION["mdp"] : Le mot de passe fourni.

Un message indique à l'utilisateur qu'il est connecté en tant qu'administrateur

print_r($_SESSION) est toujours utilisé pour le débogage :

rArray ([admin] => 1 [login] => admin [mdp] => admin)

Si $admin est false, cela signifie que les identifiants fournis sont incorrects. Un message est affiché à l'utilisateur indiquant que le login ou le mot de passe est incorrect.

-Etape2 Visualisation des inscriptions

Lorsque l'utilisateur est connecté, il peut accéder au formulaire de visualisation des inscriptions par conférence.

Dans la page fonctionsAccesDonnees.php :

Ecrire la fonction function donnerToutesLesConferences() qui retourne toutes conférences sous forme d'un tableau.

Chaque ligne du tableau contient les informations sur une conférence : son id, son créneau, sa description, son intervenant, sa salle et son nombre de places.

```
function donnerToutesLesConferences(){
    $document = chargeJSONseminaire();
    foreach($document->seminaire->creneau as $creneau){
        foreach($creneau->conference as $conference){
            $tabConf[] = $conference;
        }
    }
    return $tabConf;
}
```

Ligne 2 : On appelle la fonction chargeJSONseminaire() pour charger les données des séminaires à partir d'un fichier ou d'une source JSON. Le résultat est stocké dans la variable $document.

Ligne 3 : On commence une boucle foreach pour parcourir chaque créneau ($creneau) dans la propriété creneau de l'objet $document->seminaire. Chaque créneau contient des informations sur les conférences qui se déroulent à ce moment-là.

Ligne 4 : On commence une autre boucle foreach pour parcourir chaque conférence ($conference) dans le créneau actuel.

Ligne 5 : Pour chaque conférence trouvée, on l'ajoute au tableau $tabConf. Cela permet de stocker toutes les conférences dans un tableau.

Page déconnexion :

```
<?php
session_start();
session_unset();
session_destroy();
header('Location: ./?action = index.php');
?>
```

Ligne 2 : Démarre la session ou reprend une session existante. Cela permet d'accéder aux variables de session définies précédemment, comme celles stockant des informations sur l'utilisateur.

Ligne 3 : Efface toutes les variables de session. Cela signifie que toutes les informations stockées dans la session actuelle (comme le nom d'utilisateur, les rôles, etc.) sont supprimées.

Ligne 4 : Détruit complètement la session. Cela signifie que la session est terminée et que toutes les informations de session sont définitivement perdues.

Ligne 5 : Envoie un en-tête HTTP pour rediriger le navigateur vers une nouvelle URL. Dans ce cas, l'URL ./?action=index.php est utilisée pour rediriger l'utilisateur vers la page d'accueil ou l'index.