

Projet BDW1

Application de gestion des courses à pied

Descriptif du projet

Une association sportive souhaite proposer à ses adhérents un outil de gestion des courses de manière à conserver une trace des différentes performances de ses adhérents.

Cette application est utilisée par :

- un administrateur de l'association qui aura la charge de gérer la base de données
- les différents adhérents qui pourront visualiser les informations les concernant.

Pour chaque course, on stocke le nom de la course (e.g. 'Marathon de Paris'), l'année de création de la course (e.g. 1976), le ou les types d'épreuves qu'il est possible de réaliser durant la course (e.g. 10km, semi-marathon, marathon...). Pour chaque épreuve, on dispose de sa distance en kilomètres (e.g. 10, 21.1, 42.195) et du dénivelé. On stocke également le mois où la course a lieu en général (e.g. le 'Marathon de Paris' est en avril, la 'Run in Lyon' en octobre...).

Une course se décline chaque année en édition. Une édition de course est identifiée relativement par la course et l'année où celle-ci se déroule. Pour chaque édition, on stocke le nombre total de participants, un plan du parcours (au format .jpg ou .png), une adresse de départ, les dates d'inscription, les dates de dépôt des certificats et de récupération des dossards. Pour finir, on stocke l'url du site officiel de la course et les différents tarifs d'inscription selon l'âge et les différentes épreuves possibles.

Pour chaque édition de course, les organisateurs fournissent à l'association les résultats sous la forme d'un script SQL contenant le classement général défini selon le schéma suivant :

Resultat(dossard, rang, nom, prenom, sexe)

TempsPassage(dossard, km, temps)

A noter que si au moins un temps de passage est manquant (attribut temps non valué pour un dossard et un km donnés), le coureur est considéré comme défaillant ce qui se traduit par une absence de valeur pour l'attribut rang.

L'association dispose d'une liste d'adhérents. Pour chaque adhérent, on stocke les informations personnelles concernant le nom, le prénom, la date de naissance, le sexe, l'adresse (composée d'une rue, d'un code postal et d'une ville) de l'adhérent. L'adhérent est identifié par un numéro d'adhérent qui lui est attribué à la création de sa fiche. Un numéro d'adhérent est composé de sept chiffres. Les quatre premiers représentent l'année de création de la fiche et les trois derniers représentent un compteur incrémenté à chaque inscription. Ainsi le 46^{ième} inscrit en 2013, aura pour numéro d'adhérent 2013046. On stocke également des informations sportives concernant la date de validité du dernier certificat médical en cours et le nom du club d'athlétisme si l'adhérent est licencié dans un club.

Rendu et notation du projet

Rendu du projet : ce projet est à rendre avec les consignes de rendu suivantes :

Le fichier rendu (un seul par binôme) aura une extension **.zip** et il doit inclure votre/vos noms à la fois **dans le nom de fichier** et **à l'intérieur des fichiers**

Ce fichier **zip** contiendra :

- le répertoire **htdocs** contenant les pages de votre site (code commenté et indenté)
- un fichier **baselif4.sql** qui contiendra un export SQL de votre base de données (obtenu via l'onglet 'export' de phpmyadmin)
- le script SQL "exécutable" de création de votre base de données ainsi que les requêtes demandées (en partie 2)
- un fichier **requetes.sql** qui contiendra le script SQL "exécutable" des requêtes demandées (en partie 2)
- un rapport **rapport.pdf** de 10 pages maximum (annexes incluses), qui détaillera vos choix de conception pour la BD, les problèmes rencontrés aussi bien au niveau organisationnel (travail en équipe) qu'au niveau technique, d'éventuelles explications sur les choix de votre MCD (schéma) et MLD (textuel) en annexe, etc.

Ce fichier .zip est à rendre le 21/04/19 avant minuit dans la zone de dépôt Projet sur SPIRAL

Notation du projet : le projet compte pour 30% dans la note finale de l'UE, et la note de projet se calcule ainsi : 1/2 code + 1/2 rapport.

De plus, les conditions suivantes s'appliquent :

- *Concernant le code* : **toute ressemblance trop importante entre des projets sera sanctionnée par un 0 pour la note de code et ce pour chacun des protagonistes.**
- *Concernant le rapport* : il doit être organisé, clair, au format A4 et en police de taille 12, et l'orthographe sera prise en compte.

Environnement de travail

XAMPP

XAMPP (X Apache MySQL Perl PHP) est un environnement qui transforme votre ordinateur en serveur Web et qui fournit les plateformes de développement (Perl et PHP) ainsi qu'une base de données MySQL.

L'accès à la BD avec XAMPP s'effectue via l'interface Web PhpMyAdmin accessible à l'adresse suivante : <http://localhost/phpmyadmin/> ou <http://localhost:8080/phpmyadmin> en fonction de votre configuration.

L'accès à votre site avec XAMPP s'effectue via un navigateur Web (e.g., Firefox, Chrome) en allant à l'adresse : http://localhost/votre_site/ ou http://localhost:8080/votre_site/ où votre_site correspond à un sous-répertoire du répertoire *htdocs* sur Xampp.

A noter qu'il est possible d'utiliser d'autres solutions équivalentes WAMP, LAMP et MAMP.

Documentation

Documentation de MySQL (et les commandes SQL implémentées dans ce SGBD) :

<http://dev.mysql.com/doc/refman/5.0/fr/index.html>

Documentation de PHP et index des fonctions de PHP :

<http://www.php.net/manual/fr/> et <http://www.php.net/manual/fr/funcref.php>

Partie 1 : Conception de la base de données

Objectif

- Produire le modèle conceptuel (schéma entité/association) à partir des spécifications.
- Produire le modèle logique (modèle Relationnel)
- Produire le script de création de la base
- Importer ce script dans MySQL (avec l'interface PhpMyAdmin)

Produire le schéma entité/association

Vous êtes libre(s) d'utiliser le logiciel de votre choix pour concevoir le schéma entité association, avec le formalisme Merise (liste non-exhaustive d'outils dans les liens utiles). Vous incluez en annexe de votre rapport une exportation de votre schéma entité/association.

Liens utiles :

- http://en.wikipedia.org/wiki/Entity-relationship_model#ER_diagramming_tools
- http://en.wikipedia.org/wiki/List_of_UML_tools
- http://fr.wikipedia.org/wiki/Liste_de_lecture

Les spécifications de l'application Web à réaliser sont les suivantes :

Nous distinguons deux profils utilisateur :

- Le profil '**administrateur**' correspondant à un membre responsable de l'association qui a la possibilité :
 - de gérer (ajouter/modifier/supprimer) des adhérents, des courses, des éditions de courses
 - d'importer des résultats dans la base
- Le profil '**adhérent**' correspondant à un adhérent de l'association qui a la possibilité :
 - de gérer ses données personnelles
 - de visualiser les éditions de courses effectuées, ses classements et ses temps de passage

Fonctionnalités du profil 'administrateur'

Le profil 'administrateur' permet d'accéder à un espace personnalisé dédié à la gestion des données dans la base. L'administrateur peut visualiser la liste de toutes les courses et de leurs différentes éditions. Au niveau des noms de courses, l'administrateur peut voir le nombre moyen d'adhérents ayant couru les différentes éditions de la course et au niveau d'une édition de course, l'administrateur peut voir le nombre d'adhérents ayant couru l'édition.

De plus, il a la possibilité :

- d'ajouter une course
- d'ajouter une édition à une course

- de supprimer une course
- de supprimer une édition de course

En cliquant sur le nom d'une course, l'administrateur peut visualiser les informations relatives à la course. Il a également la possibilité de modifier les informations hormis le nom de la course qui lui ne peut changer. Si le nom de la course change, il faut alors créer une nouvelle course.

En cliquant sur le nom d'une édition de course, l'administrateur peut visualiser les informations relatives à l'édition de course, à la liste des adhérents ayant couru l'édition et les informations suivantes :

- le nombre total d'adhérents ayant couru l'édition.
- le nombre d'adhérents ayant couru l'édition en tant que licencié dans un club d'athlétisme
- le nombre de clubs d'athlétisme représentés durant l'édition de la course
- le temps du vainqueur de l'édition
- le meilleur et le pire rang/temps obtenu par un adhérent de l'association
- la moyenne des temps réalisés par les adhérents de l'association
 - o La moyenne des temps réalisés par sexe pourra être affichée si les deux catégories sont représentées dans l'édition de la course
- le nombre d'abandons d'adhérents de l'association

La liste des adhérents ayant participé à l'édition de la course sélectionnée est triée :

- par ordre lexicographique sur le nom et le prénom, si l'édition de la course n'a pas encore été courue.
- Par rang, si les résultats de la course sont connus

L'administrateur a également la possibilité de modifier les informations relatives à l'édition hormis l'année qui ne peut pas être changée.

L'administrateur a la possibilité de créer et de supprimer le compte d'un adhérent. Il dispose d'une interface permettant de créer un identifiant et un mot de passe dans la base. On suppose que l'information est transmise à l'adhérent par mail (indépendamment de l'application). Cette interface permet également de visualiser la liste des adhérents, de sélectionner un adhérent et de supprimer toutes les informations concernant cet adhérent dans la base.

Le profil 'administrateur' permet également d'importer les résultats fournis par les organisateurs d'une édition de course. [**Remarque pour les développeurs** : il sera possible de coupler la notion de création d'édition de course et la phase d'import de manière à ce que l'importation des résultats engendre la création de l'édition de course]. L'administrateur dispose d'une interface permettant de sélectionner un script SQL pour charger les résultats. Cependant toute l'information contenue dans le fichier n'est pas considérée comme pertinente puisqu'elle concerne tous les participants à l'édition de la course. Il est donc nécessaire de filtrer les informations pour ne garder que les données sur les adhérents de l'association. Pour cela, il sera nécessaire d'effectuer une tâche d'intégration de données afin de retrouver parmi tous les participants ceux qui sont adhérents à l'association. Toutefois, les informations concernant le vainqueur de la course et le nombre total de participants devront être conservées.

Fonctionnalités du profil 'adhérent'

Le profil 'adhérent' permet d'accéder à un espace personnalisé dédié aux informations relatives à l'adhérent.

L'adhérent peut modifier ses informations personnelles (le nom, le prénom, la date de naissance, le sexe, l'adresse). On suppose qu'à la première utilisation de l'application, l'adhérent se connecte *via* l'identifiant et le mot de passe fournis par l'administrateur et ce, par

mail, et que l'application l'invite à remplir les informations personnelles le concernant. Pour pouvoir enregistrer des informations dans la base, l'application devra s'assurer que l'adhérent a au moins spécifié son nom, son prénom et son sexe.

L'adhérent peut modifier ses informations sportives (date de validité du certificat médical en cours, nom du club d'athlétisme où l'adhérent est licencié).

L'adhérent dispose d'une interface permettant de visualiser la liste des éditions de courses qu'il a réalisées. La liste est triée par année, puis par distance de course, puis par temps de course. Par exemple :

- 2013
 - 42.195
 - Marathon de Paris (3h56m)
 - Run in Lyon (4h06)
 - 10
 - Run in Lyon (51m)
- 2012
 - ...

Pour des raisons de simplification, on suppose qu'un utilisateur ne peut pas avoir à la fois le profil *administrateur* et le profil *adhérent*.

Gestion de l'authentification

L'application repose sur le principe d'espaces personnalisés, il est donc nécessaire de mettre en place une phase d'authentification qui consiste à demander aux administrateurs et aux adhérents leur identifiant et leur mot de passe pour pouvoir interroger la base de données et si les informations saisies sont correctes alors l'utilisateur se retrouve face à son espace personnalisé. Sinon, une page d'erreur avertit l'utilisateur que les informations saisies sont erronées et un lien lui permet de revenir à la page d'accueil contenant le formulaire de saisie de l'identifiant et du mot de passe.

L'application web que vous avez à développer devra implémenter les fonctionnalités stipulées dans le descriptif du projet et la spécification ci-dessus.

Concernant la partie importation des résultats, vous trouverez **ici (à venir)** le script correspondant aux résultats de la course 'Marathon du LIF4' pour l'édition 2014.

Produire le modèle Relationnel

Selon l'outil de modélisation choisi à l'étape précédente, il est possible de générer le modèle Relationnel automatiquement, mais il est recommandé de bien vérifier le résultat obtenu et de le modifier si nécessaire. Sinon, la dérivation du MCD vers le MLD se fait à partir des règles énoncées en cours. Vous inclurez une exportation de votre modèle relationnel dans le rapport.

Produire le script de création de la base

Selon l'outil de modélisation choisi à l'étape précédente, il est possible de générer le script de création des tables automatiquement, mais il est recommandé de bien vérifier le résultat obtenu

et de le modifier si nécessaire. Sinon, rédigez les requêtes de création de vos tables. Vous incluez une exportation de votre base de données dans le fichier *baselif4.sql* de rendu.

@TODO : produire le schéma Entité/Association de la base de données dédiée à votre application. Puis produire le code SQL pour implémenter votre base. Vous peuplerez votre base à partir des utilisateurs mis à votre disposition ici : <http://liris.cnrs.fr/nicolas.lumineau/teaching/LIF4/2014/projet/adherentCoureur.csv>

A venir

Vous avez également les résultats de trois courses : 10km LIF4 2013, Marathon LIF4 2013 et Marathon LIF4 2012. Les jeux de données sont disponibles ici :

<http://liris.cnrs.fr/nicolas.lumineau/teaching/LIF4/2014/projet/resultats>

A venir

Partie 2 : Quelques Requêtes SQL

Objectifs

- Identifier et écrire les requêtes SQL nécessaires à votre application.

A partir du descriptif du projet, vous allez pouvoir identifier les requêtes dont vous avez besoin pour chacun des profils *administrateur* et *adhérent*. Les besoins sont explicites. Vous êtes invités à réfléchir à d'autres besoins qui ne seraient pas forcément explicites dans le sujet (e.g. requête pour la gestion de l'authentification).

@TODO : écrire le script **requetes.sql** répertoriant l'ensemble des requêtes qui interrogerons votre base de données. Le fichier pourra être complété au fur et à mesure que vous avancerez dans le projet.

Partie 3 : Architecture du site

Objectif :

- Créer la hiérarchie de fichiers de votre site
- Créer des liens hypertexte de navigation entre les fichiers

Hiérarchie de fichiers de votre site

Dans le répertoire de XAMPP, vous trouverez un sous-répertoire *htdocs*. Ce sous-répertoire contient les différents sites qui sont hébergés par votre ordinateur. Pour accéder à l'un de ces sites, par exemple, celui dans le sous-répertoire *exempleSite*, il suffit de saisir dans un navigateur : <http://localhost/exempleSite/>

En saisissant cette URL, le navigateur cherche, conformément à la configuration du serveur, à afficher un fichier *index.** présent à la racine du sous-répertoire *exempleSite*. En général, c'est le fichier *index.html*, *index.php*, *index.jsp* ou *index.asp* (L'extension des fichiers *.php* *.jsp* ou *.asp* dépend de la technologie utilisée)

La première étape va consister à trouver un nom à votre application. A défaut de nom original, vous pourrez prendre le nom **projetlif4**.

Le fichier **index.php** sera dédié à l'accueil et l'authentification des utilisateurs. Sur cette page apparaîtra le nom de votre application, éventuellement un logo et un formulaire permettant de saisir un identifiant, un mot de passe et de cliquer sur un bouton pour soumettre le formulaire. Pour atteindre cette page, l'utilisateur devra saisir directement l'url <http://localhost/projetlif4/index.html>

Une fois les informations d'authentification (identifiant et mot de passe) saisies *via* le formulaire du fichier précédent **index.html**, l'utilisateur se retrouve sur la page **espaceperso.php**.

Cette page affichera :

- une interface dédiée à l'administrateur de l'association s'il s'agit d'un utilisateur correctement identifié avec un profil 'administrateur',
- une interface dédiée à l'adhérent de l'association s'il s'agit d'un utilisateur correctement identifié avec un profil 'adhérent',
- une interface avec un message d'erreur d'authentification (si l'authentification n'a pu se faire correctement) avec un lien vers la page index pour pouvoir réessayer la phase d'authentification.

Le principe qui sera utilisé par la suite est que le fichier **espaceperso.php** affichera le contenu d'autres fichiers correspondant à différentes fonctionnalités par le principe d'inclusion en fonction des paramètres fournis. On pourra trouver dans le fichier **espaceperso.php** du code de la forme :

```
<?php
$page = $_GET['page'];
if($page == 'courses'){
    include ('../adm/courses.php') ;
}else if($page == 'adherents')
    include ('../adm/adherents.php');
...
}else{
    include ('../erreur.php');
}
```

Ce code permet d'afficher le contenu du fichier :

- **courses.php** du répertoire *adm* si l'utilisateur accède à la page <http://localhost/projetlif4/espaceperso.php?page=course>,
- **adherents.php** du répertoire *adm* si l'utilisateur accède à la page <http://localhost/projetlif4/espaceperso.php?page=adherents>,
- **erreur.php** du répertoire racine dans tous les autres cas.

Listons à présent les fichiers dont vous aurez besoin. (liste non exhaustive)

Fichiers pour le profil administrateur

Pour le profil administrateur, la page **espaceperso.php**, permet d'accéder au contenu des fichiers suivants contenus dans le répertoire *adm* :

- **accueil.php** : permet de visualiser la liste de toutes les courses et de leurs différentes éditions. Au niveau des noms de courses, l'administrateur peut voir le nombre moyen d'adhérents ayant couru les différentes éditions de la course et au niveau d'une édition de course, l'administrateur peut voir le nombre d'adhérents ayant couru l'édition. Le fichier **accueil.php** dispose également d'un menu permettant d'afficher dans la page **espaceperso.php** le contenu des fichiers suivants.
- **courses.php** : permet d'ajouter et supprimer des courses et des éditions de courses.
- **course.php** : permet de modifier les informations relatives à une course ou à une édition de course.
- **import.php** : permet d'importer les résultats avec temps de passage pour une course
- **resultats.php** : permet de visualiser les résultats d'une édition de course ()
- **adhérents.php** : permet de visualiser la liste des adhérents, ajouter, et supprimer un adhérent de l'association.
- **adherent.php** : permet de visualiser et modifier la fiche d'un adhérent de l'association.

Fichiers pour le profil *adhérent*

Pour le profil adhérent, la page **espaceperso.php**, permet d'accéder au contenu des fichiers suivants contenus dans le répertoire *adh* :

- **accueil.php** : dispose d'un menu permettant d'afficher dans la page **espaceperso.php** le contenu des fichiers suivants.
- **courses.php** : permet de visualiser la liste des éditions de courses réalisées par l'adhérent.
- **course.php** : permet de visualiser les informations relatives à une course ou à une édition de course (hors résultats). Pour les éditions de courses, le fichier permet de visualiser l'ensemble des commentaires fait sur l'édition.
- **resultat.php** : permet de visualiser les temps de parcours d'une édition de course donnée.
- **fiche.php** : permet de visualiser et modifier la fiche de l'adhérent.

@TODO : A partir du descriptif précédent, créer dans *htdocs* les répertoires et les fichiers nécessaires à votre application.

Partie 4 : Connexion à la base de données

Objectif :

- Factoriser les instructions de connexion à la base
- Afficher les données

Factoriser les instructions de connexion à la base

Les instructions de connexion à la base de données vont être regroupées dans le fichier **connexionBD.php** que vous mettrez dans un répertoire nommé **includes**.

Pour mémoire, l'exécution d'une requête dans la base de données s'effectue en trois étapes :

- Ouverture de la connexion

```
$user = 'un_nom' ;  
$mdp = 'un_mdp' ;
```



```

$machine = '127.0.0.1' ; // machine locale
$bd = 'une_bd' ;
$connexion = mysqli_connect($machine,$user,$mdp, $bd) ;
if(mysqli_connect_errno()) // erreur si > 0
printf("Échec de la connexion :
        %s",mysqli_connect_error()) ;

else {
// utilisation de la base
}
- Traitement de la requête
$req=' SELECT...FROM...WHERE...' ;
$tableauRetourne = array() ;
$resultat = mysqli_query($connexion, $req) ;
if($resultat == FALSE) // échec si FALSE
    printf("Échec de la requête" ;
else {
    // collecte des métadonnées
    $finfo = mysqli_fetch_fields($resultat);
    $entete=array() ;
    foreach ($finfo as $val) {
        $entete($val->name);
    }
    $tableauRetourne[0]=$entete ;
    $cpt=1 ;
    while ($ligne = mysqli_fetch_array($resultat)) {
        $tableauRetourne[$cpt++]= $ligne ;
    }
}
- Fermeture de la connexion
mysqli_close($connexion) ;

```

Ces étapes seront regroupées dans une fonction php que vous nommerez **traiterRequete** dans le fichier **connexionBD.php**. Cette fonction prendra en paramètre une chaîne de caractères (représentant une requête SQL) et retournera un tableau contenant les tuples résultats de la requête.

A partir du cours, écrivez le corps de la fonction **traiterRequete** en respectant les 3 étapes rappelées ci-dessus.

Comme vous aurez *a priori* besoin d'interroger la base de données dans chaque fichier se trouvant à la racine, modifier le fichier **entete.php** pour y inclure le fichier **connexionBD.php**.

Afficher les données

Votre fonction *traiterRequete* retournera un tableau que vous devrez parcourir (en utilisant une boucle FOR) pour pouvoir afficher son contenu.

Créer dans le fichier **connexionBD.php**, la fonction `Array2Table` qui prend en paramètre un tableau de tableau (retourné par la fonction *traiterRequete*) et retourne le code HTML représentant le tableau sous la forme d'un tableau HTML.

```
$leTableau = '<table>';
For($tableauRetourne as $tuple) {
    $leTableau .= '<tr>';
    For($tuple as $attribut) {
        $leTableau .= '<td>' . $attribut . '</td>';
    }
    $leTableau .= '</tr>';
}
$leTableau = '</table>';
echo $leTableau ;
```

@TODO : Ecrire la fonction *traiterRequete* et tester. Pour cela, modifier le fichier **courses.php** du profil adhérent pour afficher la liste des courses. Ensuite, modifier à nouveau le même fichier pour faire apparaître les différentes éditions de chaque course.

Partie 5 : L'authentification, d'accueil et d'inscription

Objectif :

- Créer un formulaire d'authentification qui pointera sur la page **espaceperso.php**.
- Rediriger l'utilisateur si l'authentification n'est pas correcte
- Mémoriser l'utilisateur courant

Créer un formulaire d'authentification

Modifier le fichier **index.php** pour ajouter un formulaire contenant trois inputs :

- pour saisir un login (*input* nommé *pLogin* de type *text*)
- Pour saisir un mot de passe (*input* nommé *pPwd* de type *password*)
- Pour soumettre le formulaire (*input* nommé *pEnvoyer* de type *submit*)

La soumission du formulaire permettra d'accéder à la page **espaceperso.php**.

La vérification du *login/password* doit se faire au niveau du SGBD. Autrement dit, il n'est pas question de sélectionner tous les *logins* dans la base et de vérifier dans votre code php si le login saisi correspond. Vous devrez exécuter une seule requête SQL qui vérifiera s'il existe dans votre base le couple *login/password* saisi respectivement dans les onglets *login* et *password*.

Dans un premier temps, la page **espaceperso.php** :

- Lance une redirection vers la page **erreur.php**, si l'utilisateur essaie de se connecter directement à la page via <http://localhost/projetlif4/espaceperso.php> sans passer par le formulaire. Ce cas est détectable si l'expression `isset($_POST['pEnvoyer'])`

retourne fausse. Une redirection se déclenche via la commande suivante dans le header du fichier **espaceperso.php** :

```
<?php

header( 'Location: http://localhost/projetlif4/erreur.php' ) ;

?>
```

Un lien hypertexte sera proposé à l'utilisateur sur la page **erreur.php** pour revenir à la page **index.php**.

- Lance une redirection vers la page **erreur.php**, si le login ou le password saisis ne sont pas corrects.
- Si le formulaire a bien été saisi et si le login et le password saisis sont corrects, un message de bienvenue et personnalisé (incluant le login, le prénom et le profil de l'utilisateur) sera affiché. Pour cela, vous devrez récupérer les informations en interrogeant la base de données.

Mémorisation de l'utilisateur courant

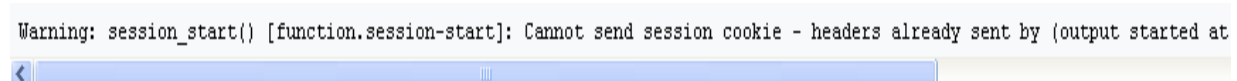
Une fois que le bon login et le bon mot de passe ont été saisis, il serait intéressant de les mémoriser durant le reste de la navigation dans l'application. Cependant, ce n'est pas le cas avec le code que vous avez produit à l'étape précédente.

Pour pouvoir conserver le login, il est nécessaire de le mémoriser sous la forme d'une variable de session.

Pour utiliser les sessions, il faut exécuter l'instruction suivante *avant* d'afficher la moindre chose dans la page PHP, même pas un espace ou un retour à la ligne:

```
<?php session_start() ; ?>
```

En particulier, laisser une ligne blanche ou un espace avant provoquera l'erreur suivante:



Warning: session_start() [function.session-start]: Cannot send session cookie - headers already sent by (output started at ...)

En fait, l'exécution de cette instruction créera le tableau associatif `$_SESSION` dont le contenu est sauvé à chaque fin d'exécution d'une page PHP avec session et restauré au moment de l'exécution de `session_start()`.

Par exemple, pour sauvegarder le login de l'utilisateur courant, il suffit d'affecter la valeur dans le tableau associatif `$_SESSION` via le code suivant :

```
<?php $_SESSION['slogin'] = $_POST['pLogin']; ?>
```

Vous pourrez ensuite accéder au login de l'utilisateur depuis n'importe quelle page de votre site (tant que la session n'est pas clôturée).

Vous pourrez bien évidemment stocker dans le tableau associatif n'importe quel couple attribut/valeur dont vous pourriez avoir besoin durant la navigation dans le site.

Gestion de la déconnexion

Pour permettre à chaque utilisateur de pouvoir se déconnecter à tout moment, il est nécessaire d'associer à chacune de vos page un lien « Déconnexion » qui permet de supprimer la session via :

```
<?php session_destroy() ; ?>
```

et pour qu'ensuite l'utilisateur se retrouve à la page **index.php**.

Partie 6 : Interface profil administrateur

En fonction de la spécification de l'application, définir les interfaces permettant à l'administrateur de visualiser et manipuler les données de la base selon les fonctionnalités associées à son profil.

@TODO : Modifier les fichiers du répertoire *adm* pour répondre aux fonctionnalités attendues pour ce profil.

Partie 7 : Interface profil adhérent

En fonction de la spécification de l'application, définir les interfaces permettant aux adhérents de visualiser et manipuler les données de la base selon les fonctionnalités associées à leur profil.

@TODO : Modifier les fichiers du répertoire *adh* pour répondre aux fonctionnalités attendues pour ce profil.