**Prof :Elie Hamouche 09/10/2019**

**eliehamouche@hotmail.com**

**Programmation web**

**NB : pour tous les groupes(mardi,mercredi et Jeudi)**

**Coté serveur : php-Mysql**

**TP5- la persistance des données et l’architecture MVC**

***On désire développer une application web php-Mysql effectuant un shopping online .Veuillez développer en PHP ,jQuery (et HTML) les pages de ce site.***

***NB : Vous pouvez résoudre ce TP selon votre choix (en utilisant l’orienté objet MVC, la méthode classique de fonctions, etc.) afn de :***

* ***rendre les objets persistants, en les sauvegardant dans une base de données.***
* ***Utiliser si vous désirez le pattern design MVC qui permet de bien organiser son code source en créant des classes Controller, Model et View pour séparer les parties traitements, données et les pages de vue HTML .***
1. Dans **Workbench** ,créer votre base de données **Ecommerce** qui est l’ensemble de 4 tables : ***(user ,catégorie, produit et Commande)*** puis exporter lavers l’outil PhpMyAdmin**.**

**Schéma de la database Ecommerce sous Mysql**



CREATE TABLE IF NOT EXISTS `Ecommerce`.`commande` (

 `cmdId` VARCHAR(10) NOT NULL,

 `nomCmd` VARCHAR(30) NOT NULL,

 `dateCmd` VARCHAR(6) NOT NULL,

 `QteCmd` INT(3) NOT NULL,

 `UserId` INT(20) NOT NULL,

 `ProdId` INT(10) NOT NULL,

 `NoFactId` INT(10) NOT NULL,

 `confirm` TINYINT(1) NOT NULL,

 PRIMARY KEY (`cmdId`))

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS ` Ecommerce `.`User` (

 `UserId` VARCHAR(30) NOT NULL,

 `nom` VARCHAR(50) NOT NULL,

 `prenom` VARCHAR(50) NOT NULL,

 `profile\_id` INT(3) NOT NULL,

 `email` VARCHAR(30) NOT NULL,

 `pwd` VARCHAR(20) NOT NULL,

 `rpwd` VARCHAR(20) NOT NULL,

 `commande\_cmdId` VARCHAR(10) NOT NULL,

 PRIMARY KEY (`UserId`),

 CONSTRAINT `fk\_User\_commande1`

 FOREIGN KEY (`commande\_cmdId`)

 REFERENCES `mydb`.`commande` (`cmdId`)

 ON DELETE NO ACTION

 ON UPDATE NO ACTION)

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS ` Ecommerce`.`categorie` (

 `catid` INT(10) NOT NULL,

 `nomcat` VARCHAR(50) NOT NULL,

 PRIMARY KEY (`catid`))

ENGINE = InnoDB;

CREATE TABLE IF NOT EXISTS ` Ecommerce`.`produit` (

 `ProdId` INT(10) NOT NULL,

 `nomprod` VARCHAR(45) NOT NULL,

 `prix` DECIMAL(10) NOT NULL,

 `photo` BLOB(30) NOT NULL,

 `catId` INT(10) NOT NULL,

 `NoFact` VARCHAR(10) NOT NULL,

 `categorie\_catid` INT(10) NOT NULL,

 `commande\_cmdId` VARCHAR(10) NOT NULL,

 PRIMARY KEY (`ProdId`, `categorie\_catid`, `commande\_cmdId`),

 CONSTRAINT `fk\_produit\_categorie`

 FOREIGN KEY (`categorie\_catid`)

 REFERENCES `mydb`.`categorie` (`catid`)

 ON DELETE NO ACTION

 ON UPDATE NO ACTION,

 CONSTRAINT `fk\_produit\_commande1`

 FOREIGN KEY (`commande\_cmdId`)

 REFERENCES `mydb`.`commande` (`cmdId`)

 ON DELETE NO ACTION

 ON UPDATE NO ACTION)

ENGINE = InnoDB;

1. **Développer le code php du modèle MVC(Model,View controller) décrit comme suit :**

******



 Architecture MVC

À la racine de l’application, créer 3 répertoires  et un fichier **index.php** ou **routeur.php** comme contrôleur globale ou frontière qui lance controleurUser.php dans ce TP

* **../model/** : qui contient les fichiers gérant l'accès à la base de données sous Mysql de la database E-Commerce ;
* **../view/** : qui contient les fichiers gérant l'affichage ou de vue d’Ecommerce ;
* **…/controller/** : qui  contient les fichiers contrôlant le fonctionnement global d’ Ecommerce , c.à.d. de gérer les actions déclenchées par l’utilisateur a l’envoie de ses requêtes

***Pour répondre a l’ architecture MVC , on recommande de créer a***

1. ***stucture des répertoires suivants :***

 ***Index.php***

Controller

 ***controllerUser.php***

Model

 ***modelUser.php, dbconnect.php***

TP5

View

 ***i viewUser.php , viewSignIn.php , viewAccueil***

 ***style.css***

css

 ***functions.js***

js

1. Développez le code d’un programme ***dbconnect.php*** qui établit la connexion au serveur Mysql en utilisant l’objet **PDO**  et qu’il faudrait l’inclure d’après **required\_once ou include(‘dbconnect’)** dans les modèles de l’application .

<?php

**class dbConnect**

 {

**//1. variables membres**

private $host = '';

private $user = '';

private $pass = '';

private $dbh= '';

**// 2. constructeur plein**

public function \_\_construct($host='localhost',$user='root',$pass='')

{

 $this->host=$host;

 $this->user=$user;

 $this->pass=$pass;

 $dbh = new PDO("mysql:host=$host;dbname=ecommerce", $user , $pass );

 // echo 'Connected to database<br />';

}

**//3. méthode de connexion à la base de données sous Mysql via pdo Object**

 public function connect()

 {

 $host="localhost";

 $user="root";

 $pass="";

 $dbh = new PDO("mysql:host=$host;dbname=ecommerce", $user , $pass );

 return $dbh;

 }

 **// 2eme méthode connexion(classique) a la base de données Mysql**

 public function connect2()

 {

 $host="localhost";

 $user="root";

 $pass="";

 $dbh="ecommerce";

 $con=@mysql\_connect($host,$user,$pass) or die("Connexion échouée");

 $db=@mysql\_select\_db($dbh) or die("La base ne peut pas être selectionnée");

 return $db;

 }

 }

**?>**

1. créer un contrôleur par modèle sous **TP7/Controller/controllerUser.php** quicontrôle le fonctionnement d’un model et d’une vue

 *En effet, le* **controllerUser.php** *de ce TP se décompose en plusieurs actions :*

* On charge la déclaration de la classe ModelUser  d’après la syntaxe :

< ?php

namespace TP7\_MVC;

//a- en entrée on fait appel au modèle des entités , ici **modelUser.php**

 include ('modelUser.php');

//b- on crée une class **ControllerUser** ,c’est le moulin fabrication d'objet entre vue et modèle

 class controllerUser {

 …

}

//c- en sortie , On affiche la page (vue) ,ici **..\View\Viewlogin.php'**

 include\_once '..\View\Viewlogin.php'; //ajout view au controller

$ocont = new ControllerUser();

$res = $ocont->Action1\_login();

 echo "login :".$res;

?>

1. Utiliser le modèle sous **../Model/modelUser.php** pour développer les actions du contrôleur **controllerUser.php** afin de pouvoir :
* récupérer le résultat du login d’après : public function Action1\_login() en appelant la vue associée à cette action qui va générer la page Web avec include ('../view/Viewlogin.php');
* récupérer le résultat d’un utilisateur connu par le service d’après : public function Action\_displayUser() en appelant la vue associée à cette action qui va générer la page Web avec include ('../view/ViewAccueil.php');

* effectuer un **signIn** de l’utilisateur qui n’est pas encore authentifié d’après : public function Action2\_SignIn() ;

en appelant la vue associée à cette action qui va générer la page Web avec include ('../view/ViewSignIn.php');

 4. créer sous **../model/modelUser.php** une

 **class modelUser** dans laquelle , on

- déclare les variables membres (userid,nom,prenom,profile,..)

- définit un constructeur vide et plein

- crée des fonctions getters/setter de userid,…

- une fonction login($userid, $pwd) qui prend 2 paramètres ***userid et Mot de pass*** puis crée une session propre à tout utilisateur qui souhaite devenir un vrai client du site Ecommerce pour pouvoir le reconnaitre à tout moment lors d’une session ouverte .

1. Développer le code php de la **vue sous../View/viewSignIn.php aynat une** interaction avec l’utilisateurcomme suit :

View Authentification

**ViewLogin.php** October 14, 2016, 1:29 pm

# ****TP7-MVC : Site Ecommerce**** ****Authentication****

Top of Form

USERNAME

PASSWORD

 Remember Me



Bottom of Form

​

et lorsque l’utilisateur clique sur le bouton login on appelle le contrôleur sous **../controller/controllerUser.**php qui va activer l’action login définie dans **../Model/modelUser.php** . 2 cas se présentent ici :

1. si le user est connu par le service on le redirige à la vue d’accueil sous **../view/viewAccueil.php** dans laquelle on affiche que le username existe comme suit :

# ViewAccueil.php

# Vous etes le Bienvenue sur Notre site Ecommerce

Top of Form

October 14, 2016, 1:01 pm

Bottom of Form

login : Welcome M/Mme :eliehamouche@hotmail.com a date: 14, 10, 2016

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UserId | Nom | Prénom | profile id | Email |
| eliehamouche@hotmail.com | hamouche | elie | 2 | eliehamouche@hotmail.com |

1. sinon , c.à.d. si l’utilisateur n’est pas encore reconnu par le service en cours(serveur Mysql) , un formulaire d’inscription ou bien un autre Vue  **View/ViewSignIn.php** lui est proposé, lui permettant la saisie de ses propres informations  :

**ViewSignIn.php** October 14, 2016, 1:32 pm

**L’utilisateur en cours n’est pas reconnu**

***Si vous êtes un nouveau client, veuillez-vous enregistrer en remplissant le formulaire ci-dessous***

Inscription

|  |
| --- |
|  |
| Nom :  *svp remplir Nom*  |
| Prenom :  *svp remplir prenom*  |
| Mail :  *svp remplir Email* |
| Login :  *svp remplir pseudo*  |
| Pass : *svp remplir Mot de pass* |
| Verif Pass :  |

1. Développer le code de l’action  **ExecuteSignIn()**  de l’utilisateur dans ..**controller/controllerUser.php** qui :
	1. reçoit une requête contenant des informations saisies par l’utilisateur dans la page **View/viewSignIn.php**
	2. Déclenche l’évènement en appuyant sur le bouton **SignIn** et effectue un required\_once du **Model/modelUser.php** etdans lequel on développe la fonction :

public function ExecuteSignIn() qui réalise l’insertion d’un nouveau utilisateur et l’ enregistre dans une base de données(table User)

***NB :*** *il serait possible de développer cette application a 3 tiers(client,Serveur php et serveur Mysql) en créant les 3 pages d’interfaces web décrites ci-haut qui communiquent et dialoguent avec la classe cachée* ***Service.php*** *définie comme suit :*

<?php

**class Service{**

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*1. propriétés membres*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

 private $host='localhost';

 private $user='root';

 private $pass='NomBase';

 private $db='';

 private $con='';

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*2. Constructeur plein*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

public function \_\_construct($host='localhost',$user='root',$pass='',$db='ecomm'){

 $this->host=$host;

 $this->user=$user;

 $this->pass=$pass;

 $this->db=$db;

 $this->con =@mysql\_connect($this->host,$this->user,$this->pass);

 if(!$this->con ){throw new Exception('Erreur de connexion avec mysql');}

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*3. les fonctions et procédures*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

*// Fonction de connexion a la base de données*

public function connect(){

 $this->con=@mysql\_connect($this->host,$this->user,$this->pass) or die("Connexion échouée");

 $this->db=@mysql\_select\_db('ecomm') or die("La bdd ne peut pas etre selectionnee");

 }

*// 2eme fonction de connexion à la base de données*

public function connect2()

 {

 $host="localhost";

 $user="root";

 $pass="";

 $db="Nombase"; // donner le nom de votre bdd sous mysql

 $con=@mysql\_connect($host,$user,$pass) or die("Connexion échouée");

 $db=@mysql\_select\_db($db) or die("La base ne peut pas être sélectionnée");

}

 *//fonction qui exécute une requête sql en retournant comme résultat un recordset*

public function execute($sql){

 $res=@mysql\_query($sql,$this->con);

 return $res;

}

*//fonction qui exécute une requête sql en consultation et retourne une ligne record*

public function executesql($req){

 $res=@mysql\_query($req,$this->con);

 $row=@mysql\_fetch\_array($res);

 return ($row);

}

// Fonction qui retourne un recordset ,un résultat d'une requête écrite dans un tableau associatif

public function Tablesql($req){

 $i=0;

 $res=@mysql\_query($req,$this->con);

 $tabres=array();

 if(!$res) throw new Exception('La requete est erronnée');

 while($lig=mysql\_fetch\_assoc($res)){

 foreach($lig as $key=>$val)

 $tabres[$i][$key]=$val;

 $i++;

 }

 mysql\_free\_result($res);

 return $tabres;

}

*//procédure qui affiche le résultat d'une requête sous forme d'un tableau HTML*

function tableau($req){

 echo "<table border='1' bordercolor='#CCCC00'>";

 //lecture des fields

 $res=@mysql\_query($req,$this->con) or die("La requete est erronne");

 $rows=@mysql\_num\_rows($res);

 echo "<h2> nb de records :$rows </h2> ";

 $fields=@mysql\_num\_fields($res);

 echo "<tr>";

 for($i=0;$i<$fields;$i++){

 echo "<th>".mysql\_field\_name($res,$i)."</th>";

 }

 echo "</tr>";

 //lecture des donnees

 $res=mysql\_query($req,$this->con);

 while ($row=mysql\_fetch\_array($res)){

 echo "<tr>";

 for($i=0;$i<$fields;$i++)

 {

 echo "<td>$row[$i]</td>";

 }

 echo "</tr>";

 }

 echo "</table>";

 }

 // procédure qui realise les operations Insert/ Update/ Delete

public function insert\_up\_del($sql)

 {

 // $s1=new service();

 // $s1->connect();

 @mysql\_close();

 $res=@mysql\_query($sql,$this->con);

 ECHO $res. ' '.$this->con;

 if(!$res)

 {

 echo "<h3>Erreur d'insertion </h3>";

 }

 else

 {

 echo "<h3>une insertion est realisee</h3> ";

 }

 @mysql\_close();

 }

//fonction qui réalise l'authentification ou (login avec création d’une session)

public function login\_session($sql,$uid,$pwd)

 {

 $res=@mysql\_query($sql);

 $req=@mysql\_num\_rows($res);

 if($req==1)

 {

 session\_start();

 $row=@mysql\_fetch\_array($res);

 $\_SESSION['uid']=$row['userId'];

 return true;

 }

 else

 return false;

}

// 2eme méthode login avec session

public function login\_session2($table\_name,$un,$pwd)

 {

 $os=new service();

 $os->connect();

 $sql="SELECT UserId FROM $table\_name WHERE UserId='".$uid."' AND pwd='".$pwd."'";

 $res=@mysql\_query($sql);

 $req=@mysql\_num\_rows($res);

 if($req==1)

 {

 session\_start();

 $\_SESSION['userId']=$uid;

 $\_SESSION['pwd']=$pwd;

 $\_SESSION['userId']=$res['userId'];

 }

 }

*// fonction qui retourne l' id*

function retour\_id($tablename,$user,$pw)

{

$sql="select id from $tablename where username='$user' and password='$pw'";

$res=@mysql\_query($sql);

$resultat =@mysql\_fetch\_array($res);

return $resultat['id'];

}

*// fonction qui réalise l'exécution d'une requête sql en mise jour*

function executer($sql)

 {

 $this->connect();

 $res=@mysql\_query($sql,$this->con) or die (" Error while executing Query");

 $rows=@mysql\_affected\_rows();

 @mysql\_free\_result($sql);

 return $res;

 }

 *//fonction contre les injections sql*

 function qry($query) {

 $con = @mysql\_connect($this->host, $this->user, $this->pass) or die ('Unable to connect to the database');

 @mysql\_select\_db($this->db) or die ('Unable to select database!');

 echo "hello $this->db " ;a

 $args = func\_get\_args();

 $query = array\_shift($args);

 $query = str\_replace("?", "%s", $query);

 $args = array\_map('mysql\_real\_escape\_string', $args);

 array\_unshift($args,$query);

 $query = call\_user\_func\_array('sprintf',$args);

 echo $query;

 $res = mysql\_query($query) or die(mysql\_error());

 if($res){

 return $res;

 }else{

 $error = "Error";

 return $res;

 }

 }

*//fonction qui consulte et affiche les users en leur envoyant l'id*

public function consult2\_user($tablename,$Id){

 $os=new Service();

$os->connect();

 $sql = "SELECT \* FROM tblename WHERE UserId = $Id";

 $res = @mysql\_query($sql);

 if (!$res)

 {

 die('Invalid query: ' . mysql\_error());

 }

 $num\_rows = mysql\_num\_rows($res);

 if($num\_rows==0)

 {

 echo '<center><font color="red"><b>No record found!!</b></font></center>';

 }

 else

 {

 echo "<table border='1' bordercolor='#aaabbb'>";

 echo"<caption><b>Liste des users...</b></caption>";

 echo"<tr><td> Id </td><td> username </td> <td> Password </td> <td> Prenom </td><td> email </td><td> Adresse </td> </tr>";

 while ($rows = mysql\_fetch\_assoc($Result))

 {

 $id =$rows['UserId'];

 $no =$rows['nom'];

 $pr =$rows['prenom'];

 $pid =$rows['profile\_id'];

 $em =$rows['email'];

 $pw =$rows['pwd'];

 $rpw =$rows['rpwd'];

 echo"<tr><td> $id </td><td> $no </td><td> $pr </td><td> $pid </td><td> $em </td><td> $pw </td><td> $rpw </td></tr>";

 }

 echo"</table>";

 }

 }

*//function that Returns the IP address of the user*

function get\_ip() {

 // IP si internet partagé

 if (isset($\_SERVER['HTTP\_CLIENT\_IP'])) {

 return $\_SERVER['HTTP\_CLIENT\_IP'];

 }

 // IP derrière un proxy

 elseif (isset($\_SERVER['HTTP\_X\_FORWARDED\_FOR'])) {

 return $\_SERVER['HTTP\_X\_FORWARDED\_FOR'];

 }

 // Sinon : IP normale

 else {

 return (isset($\_SERVER['REMOTE\_ADDR']) ? $\_SERVER['REMOTE\_ADDR'] : '');

 }

}

 *//logout function*

 function logout(){

 session\_destroy();

 return;

 }

*// fonction qui clôture la connexion avec le serveur Mysql*

function deconnect()

 {

 @mysql\_close();

 }

} // fin class

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

*4 création instances et appel fonctions*

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

/\*

 echo "<h1> vous etes le bienvenue a la classe service </h1>";

$se = new Service();

$se->connect();

echo "<h3> display table </h3>";

$se->tableau('select \* from etudiant');

echo "<h3> Authentification- login </h3>";

$se->login\_session2('etudiant','xxxx','xxxx');

echo "<h2>une authentification est realisee correctement </h2>";

echo "etc...";

\*/

?>

*Good Job !*

**English**

**Web programming**

**TP5- Data Persistence and MVC Architecture**

**We want to develop a web application php-Mysql performing an online shopping .Please develop the pages of this site in PHP, jQuery (and HTML).**

***NB: You can solve this TP according to your choice (using the object-oriented MVC, the classical method of functions, etc.)***

***• make objects persistent, by saving them in a database.***

***• Use if you want the MVC pattern design that allows you to organize your source code well by creating Controller, Model, and View classes to separate the process, data, and HTML view pages.***

1. In **Workbench,** create your Ecommerce database which is the set of 4 tables: (user, category, product and Order) then export to the PhpMyAdmin tool.

***(See above the Database and MVC architecture).***

2. **Develop the php** code of the model **MVC (Model, View controller)** described as follows:

At the root of the application, create 3 directories and an ***index.php*** file or ***router.php*** as global controller or border that launches ***controllerUser.php*** in this lab

**• ../model/:** which contains the files managing the access to the database under Mysql of the E-Commerce database;

**• ../view/:** which contains the files managing the display or Ecommerce view;

• **... / controller /:** which contains the files controlling the overall functioning of Ecommerce, to manage the actions triggered by the user when sending his requests

To meet the MVC architecture, it is recommended to create a

***See above the structure of the following directories.***

Develop the code of a **dbconnect.php** program that connects to the Mysql server using the PDO and should be included according to ***required\_once*** or include ('dbconnect') in the models of the application.

* ***See above the code of the class dbconnect .***

2. Create a model controller under ***TP5 / Controller / controllerUser.php*** that controls the operation of a model and a view

      Indeed, the ***controllerUser.php*** of this TP is broken down into several actions:

• The declaration of the ModelUser class is loaded according to the syntax:]

<? Php

 namespace TP5\_MVC;

// a- in input we use the model of the entities, here modelUser.php

 include ('modelUser.php');

// b- we create a class ControllerUser, it's the mill making object between view and model

  class controllerUser {

  ...

}

// c- output, We display the page (view), here .. \ View \ Viewlogin.php '

 include\_once '.. \ View \ Viewlogin.php'; // add view to the controller

$ ocont = new ControllerUser ();

$ res = $ ocont-> Action1\_login ();

 echo "login:". $ res;

?>

3. Use the model under***../Model/modelUser.php*** to develop the controllerUser.php controller actions in order to:

* Retrieve the login result from: public function *Action1\_login ()* by calling the view associated with this action that will generate the web page with include ***('../view/Viewlogin.php');***
* Retrieve the result of a user known by the service from: public function *Action\_displayUser ()* by calling the view associated with this action that will generate the web page with include ***('../view/ViewHome.php');***
* Make a sign In of the user who is not yet authenticated according to: public function *Action2\_SignIn ();*
* calling the view associated with this action that will generate the web page with include ('***../view/ViewSignIn.php'***);

    4. Create as .**./model/modelUser.php**   class modelUser in which, one

- declare the member variables (userid, name, first name, profile, ..)

- defines an empty and full constructor

- creates userid getters / setter functions, ...

- a login function ($userid, $pwd) that takes 2 parameters userid and Password and creates a session specific to any user who wants to become a real customer of the Ecommerce site to be able to recognize it at any time during an open session .

5. Develop the php code of the view under..***/ View / viewSignIn.php*** aynat an interaction with the user as follows:

View Authentification

**ViewLogin.php** October 14, 2016, 1:29 pm

# ****TP7-MVC : Site Ecommerce**** ****Authentication****

Top of Form

USERNAME

PASSWORD

 Remember Me



Bottom of Form

​

And when the user clicks on the login button, the controller is called..***/controller/controllerUser.php*** which will activate the login action defined in..***/Model/modelUser.php.*** 2 cases are here:

1. if the user is known by the service it is redirected to the home view under ../view/viewHome.php where it is displayed that the username exists as follows:

# ViewAccueil.php

# Vous etes le Bienvenue sur Notre site Ecommerce

Top of Form

October 14, 2016, 1:01 pm

Bottom of Form

login : Welcome M/Mme :eliehamouche@hotmail.com a date: 14, 10, 2016

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| UserId | Nom | Prénom | profile id | Email |
| eliehamouche@hotmail.com | hamouche | elie | 2 | eliehamouche@hotmail.com |

d) otherwise, ie if the user is not yet recognized by the current service (Mysql server), a registration form or another View ***View / ViewSignIn.php*** is offered, allowing him to enter his own information:

**ViewSignIn.php** October 14, 2016, 1:32 pm

**L’utilisateur en cours n’est pas reconnu**

***Si vous êtes un nouveau client, veuillez-vous enregistrer en remplissant le formulaire ci-dessous***

Inscription

|  |
| --- |
|  |
| Nom :  please fill the LastName |
| Prenom :  please fill the FirstName |
| Mail : please fill ithe email |
| Login :  please fill login |
| Pass : please fill the password |
| Verif Pass : pease fill the confirmation |

7.

Develop the code of the ***ExecuteSignIn ()*** action of the user in ***..controller / controllerUser.php*** which:

a) receives a request containing information entered by the user in the ***View / viewSignIn.php*** page

b. Trigger the event by pressing the Sign In button and perform a required\_once of the ***Model / modelUser.php*** and in which one develops the function:

public function ***ExecuteSignIn ()*** performs the insertion of a new user and saves it in a database (User table)

***NB****: it would be possible to develop this application by 3 tiers (client, php server and Mysql server) by creating the 3 pages of web interfaces described above that communicate and interact with the hidden class Service.php defined as follows:*

***See above the class Service.php***

 **good luck!**