

```

1  <?php
2  session_start();
3  /**
4   *
5   *
6   *
7   *
8   *
9   *
10  */
11
12  require_once $_SERVER['DOCUMENT_ROOT'].'/controller/
ControllerSquash.php';
13
14  $action = (isset($_GET['action']))? $_GET['action'] : null
;
15
16  //Si l'utilisateur est connecté, récupère son role. Sinon
mets le role à null
17  $u = ESession::getUser();
18  $role = null;
19  if($u !== false)
20      $role = ($u->Role === false) ? ControllerSquash::
GetRole($u) : $u->Role;
21
22
23  if($role === null)
24  {
25      switch($action)
26      {
27          case 'Register' :
28              if(isset($_POST) && count($_POST) == 6)
29              {
30                  $user = new EUser();
31                  $user->Nickname = filter_input(INPUT_POST,
'nickname', FILTER_SANITIZE_SPECIAL_CHARS);
32                  $user->Password = filter_input(INPUT_POST,
'password', FILTER_SANITIZE_SPECIAL_CHARS);
33                  $user->Email = filter_input(INPUT_POST, '
email', FILTER_SANITIZE_SPECIAL_CHARS);
34                  $user->Name = filter_input(INPUT_POST, '
name', FILTER_SANITIZE_SPECIAL_CHARS);
35                  $user->Firstname = filter_input(INPUT_POST
, 'firstname', FILTER_SANITIZE_SPECIAL_CHARS);
36                  $user->Phone = filter_input(INPUT_POST, '

```

```

36 phone', FILTER_SANITIZE_SPECIAL_CHARS);
37     $user->Password = sha1($user->Password);
38     ControllerSquash::Register($user);
39 }
40     include $_SERVER['DOCUMENT_ROOT'].'/view/
register.html';
41     break;
42     case 'EmailRegisterSend' :
43         ControllerSquash::EmailRegisterSend();
44         break;
45     case 'EmailConfirmation' :
46         $token = (isset($_GET['token'])) ?
filter_input(INPUT_GET, 'token',
FILTER_SANITIZE_SPECIAL_CHARS) : null;
47         if($token === null)
48             break;
49         ControllerSquash::EmailValidation($token);
50         break;
51     default :
52         ControllerSquash::Login();
53 }
54 }
55 }
56
57 if(isset($role) && $role->CodeRole == "1")
58 {
59     switch($action)
60     {
61         case 'Logout' :
62             ControllerSquash::Logout();
63             break;
64         case 'Accueil' :
65             ControllerSquash::Accueil($role->CodeRole);
66             break;
67         case 'ShowAllUsers' :
68             ControllerSquash::ShowUsers();
69             break;
70         case 'UserProfil' :
71             $nickname = filter_input(INPUT_GET, 'Nickname'
, FILTER_SANITIZE_STRING);
72             ControllerSquash::showUserProfil($nickname);
73             break;
74         case 'DeleteReservation' :
75             $user = ESession::GetUser();
76             $reservation = new EReservation();

```

```

77         $reservation->Court = ModelCourts::
GetCourtById(filter_input(INPUT_GET, 'idCourt',
FILTER_SANITIZE_SPECIAL_CHARS));
78         $reservation->User = ModelUsers::
GetUserByNickname(filter_input(INPUT_GET, 'Nickname',
FILTER_SANITIZE_SPECIAL_CHARS));
79         $reservation->Date = filter_input(INPUT_GET,
'date', FILTER_SANITIZE_SPECIAL_CHARS);
80         ControllerSquash::DeleteReservation(
$reservation, $user);
81         break;
82         case 'Preferences' :
83             ControllerSquash::ManagePreferences();
84             break;
85         case 'ModifyPreference' :
86             $preferences = new EPreference();
87             $preferences->BeginTime = filter_input(
INPUT_POST, 'openingTime', FILTER_SANITIZE_STRING);
88             $preferences->EndTime = filter_input(
INPUT_POST, 'closingTime', FILTER_SANITIZE_STRING);
89             $preferences->NbReservation = filter_input(
INPUT_POST, 'nbReservationByUser', FILTER_SANITIZE_STRING
);
90             ControllerSquash::updatePreferences(
$preferences);
91             header("Location: ?action=Preferences");
92             break;
93         case null :
94             ControllerSquash::Accueil();
95             break;
96     }
97 }
98 else if(isset($role) && $role->CodeRole == "2")
99 {
100     switch($action)
101     {
102         case 'Logout' :
103             ControllerSquash::Logout();
104             break;
105         case 'Accueil' :
106             ControllerSquash::Accueil();
107             break;
108         case 'MyReservation' :
109             ControllerSquash::MyReservations($u);
110             break;

```

```
111         case 'DeleteReservation' :
112             $user = ESession::getUser();
113             $reservation = new EReservation();
114             $reservation->Court = ModelCourts::
GetCourtById(filter_input(INPUT_GET, 'idCourt',
FILTER_SANITIZE_SPECIAL_CHARS));
115             $reservation->User = ModelUsers::
getUserByNickname(filter_input(INPUT_GET, 'Nickname',
FILTER_SANITIZE_SPECIAL_CHARS));
116             $reservation->Date = filter_input(INPUT_GET,
'date', FILTER_SANITIZE_SPECIAL_CHARS);
117             ControllerSquash::DeleteReservation(
$reservation, $user);
118             break;
119         case null :
120             ControllerSquash::Accueil();
121             break;
122     }
123 }
124
125
```

```

1 <?php
2 /**
3  * @remark Mettre le bon chemin d'accès à votre fichier
   contenant les constantes
4  */
5 require_once $_SERVER['DOCUMENT_ROOT'].'/config/conparam.
   php';
6
7 /**
8  * @brief   Helper class encapsulating
9  *           the PDO object
10 * @author  dominique.aigroz@kadeo.net
11 * @remark
12 */
13 class EDatabase {
14     private static $objInstance;
15     /**
16      * @brief   Class Constructor - Create a new database
   connection if one doesn't exist
17      *           Set to private so no-one can create a new
   instance via ' = new EDatabase();'
18      */
19     private function __construct() {}
20     /**
21      * @brief   Like the constructor, we make __clone
   private so nobody can clone the instance
22      */
23     private function __clone() {}
24     /**
25      * @brief   Returns DB instance or create initial
   connection
26      * @return $objInstance;
27      */
28     public static function getInstance() {
29         if(!self::$objInstance){
30             try{
31
32                 $dsn = EDB_DBTYPE.':host='.EDB_HOST.':port
   ='.EDB_PORT.':dbname='.EDB_DBNAME;
33                 self::$objInstance = new PDO($dsn,
   EDB_USER, EDB_PASS, array('charset'=>'utf8'));
34                 self::$objInstance->setAttribute(PDO::
   ATTR_ERRMODE, PDO::ERRMODE_EXCEPTION);
35             }catch(PDOException $e ){
36                 echo "EDatabase Error: ".$e;

```

```
37         }
38     }
39     return self::$objInstance;
40 } # end method
41 /**
42  * @brief    Passes on any static calls to this class
43  onto the singleton PDO instance
44  * @param    $chrMethod    The method to call
45  * @param    $arrArguments    The method's parameters
46  * @return    $mix          The method's return value
47  */
48 final public static function __callStatic( $chrMethod,
49 $arrArguments ) {
50     $objInstance = self::getInstance();
51     return call_user_func_array(array($objInstance,
52 $chrMethod), $arrArguments);
53 } # end method
54 }
```

```
1 #DivEmailSend
2 {
3     margin-top: 30vh;
4 }
5
6 #selectCourts{
7     float: right;
8 }
9
10 .modal-body {
11     padding: 2px 16px;
12     height: 20vh;
13 }
14
15 .fc-time-grid .fc-slats td {
16     height: 4em;
17     border-bottom: 0;
18     /* each cell is responsible for its top border */
19 }
20
21 /* Modal Background */
22 .modal {
23     display: none; /* Hidden by default */
24     position: fixed; /* Stay in place */
25     z-index: 1; /* Sit on top */
26     left: 0;
27     top: 0;
28     width: 100%; /* Full width */
29     height: 100%; /* Full height */
30     overflow: auto; /* Enable scroll if needed */
31     background-color: rgb(0,0,0); /* Fallback color */
32     background-color: rgba(0,0,0,0.4); /* Black w/ opacity
33     */
34 }
35 /* Modal Content/Box */
36 .modal-content {
37     background-color: #fefefe;
38     margin: 15% auto; /* 15% from the top and centered */
39     padding: 20px;
40     border: 1px solid #888;
41     width: 80%; /* Could be more or less, depending on
42     screen size */
43 }
```

```
44 /* The Close Button */
45 .close {
46     color: #aaa;
47     float: right;
48     font-size: 28px;
49     font-weight: bold;
50     width: 20%
51 }
52
53 .close:hover,
54 .close:focus {
55     color: black;
56     text-decoration: none;
57     cursor: pointer;
58 }
59
60 /* Modal Header */
61 .modal-header {
62     padding: 2px 16px;
63     background-color: #767776;
64     color: white;
65 }
66
67 /* Modal Body */
68 .modal-body {
69     padding: 2px 16px;
70     height: 30vh;
71     width: 100%;
72 }
73
74 /* Modal Footer */
75 .modal-footer {
76     padding: 2px 16px;
77     background-color: #767776;
78     color: white;
79 }
80
81 /* Modal Content */
82 .modal-content {
83     position: relative;
84     background-color: #fefefe;
85     margin: auto;
86     padding: 0;
87     border: 1px solid #888;
88     width: 80%;
```



```
89   box-shadow: 0 4px 8px 0 rgba(0,0,0,0.2), 0 6px 20px 0
      rgba(0,0,0,0.19);
90   animation-name: animatetop;
91   animation-duration: 0.4s
92 }
93
94 /* Add Animation */
95 @keyframes animatetop {
96   from {top: -300px; opacity: 0}
97   to {top: 0; opacity: 1}
98 }
99
100 /***** https://bootsnipp.com/snippets/dldxB
      *****/
101 /* BASIC */
102
103 html {
104     background-color: #56baed;
105 }
106
107 body {
108     font-family: "Poppins", sans-serif;
109     height: 100vh;
110 }
111
112 a {
113     color: #92badd;
114     display:inline-block;
115     text-decoration: none;
116     font-weight: 400;
117 }
118
119 h2 {
120     text-align: center;
121     font-size: 16px;
122     font-weight: 600;
123     text-transform: uppercase;
124     display:inline-block;
125     margin: 40px 8px 10px 8px;
126     color: #cccccc;
127 }
128
129
130
131 /* STRUCTURE */
```

```
132
133  .wrapper {
134      display: flex;
135      align-items: center;
136      flex-direction: column;
137      justify-content: center;
138      width: 100%;
139      min-height: 100%;
140      padding: 20px;
141  }
142
143  #formContent {
144      -webkit-border-radius: 10px 10px 10px 10px;
145      border-radius: 10px 10px 10px 10px;
146      background: #fff;
147      padding: 30px;
148      width: 90%;
149      max-width: 450px;
150      position: relative;
151      padding: 0px;
152      -webkit-box-shadow: 0 30px 60px 0 rgba(0,0,0,0.3);
153      box-shadow: 0 30px 60px 0 rgba(0,0,0,0.3);
154      text-align: center;
155  }
156
157  #formFooter {
158      background-color: #f6f6f6;
159      border-top: 1px solid #dce8f1;
160      padding: 25px;
161      text-align: center;
162      -webkit-border-radius: 0 0 10px 10px;
163      border-radius: 0 0 10px 10px;
164  }
165
166
167
168  /* TABS */
169
170  h2.inactive {
171      color: #cccccc;
172  }
173
174  h2.active {
175      color: #0d0d0d;
176      border-bottom: 2px solid #5fbae9;
```

```
177     }
178
179
180
181     /* FORM TYPOGRAPHY*/
182
183     input[type=button], input[type=submit], input[type=
reset] {
184         background-color: #56baed;
185         border: none;
186         color: white;
187         padding: 15px 80px;
188         text-align: center;
189         text-decoration: none;
190         display: inline-block;
191         text-transform: uppercase;
192         font-size: 13px;
193         -webkit-box-shadow: 0 10px 30px 0 rgba(95,186,233,0.4
);
194         box-shadow: 0 10px 30px 0 rgba(95,186,233,0.4);
195         -webkit-border-radius: 5px 5px 5px 5px;
196         border-radius: 5px 5px 5px 5px;
197         margin: 5px 20px 40px 20px;
198         -webkit-transition: all 0.3s ease-in-out;
199         -moz-transition: all 0.3s ease-in-out;
200         -ms-transition: all 0.3s ease-in-out;
201         -o-transition: all 0.3s ease-in-out;
202         transition: all 0.3s ease-in-out;
203     }
204
205     input[type=button]:hover, input[type=submit]:hover,
input[type=reset]:hover {
206         background-color: #39ace7;
207     }
208
209     input[type=button]:active, input[type=submit]:active,
input[type=reset]:active {
210         -moz-transform: scale(0.95);
211         -webkit-transform: scale(0.95);
212         -o-transform: scale(0.95);
213         -ms-transform: scale(0.95);
214         transform: scale(0.95);
215     }
216
217     input[type=text] {
```

```
218     background-color: #f6f6f6;
219     border: none;
220     color: #0d0d0d;
221     padding: 15px 32px;
222     text-align: center;
223     text-decoration: none;
224     display: inline-block;
225     font-size: 16px;
226     margin: 5px;
227     width: 85%;
228     border: 2px solid #f6f6f6;
229     -webkit-transition: all 0.5s ease-in-out;
230     -moz-transition: all 0.5s ease-in-out;
231     -ms-transition: all 0.5s ease-in-out;
232     -o-transition: all 0.5s ease-in-out;
233     transition: all 0.5s ease-in-out;
234     -webkit-border-radius: 5px 5px 5px 5px;
235     border-radius: 5px 5px 5px 5px;
236 }
237
238 input[type=password]
239 {
240     background-color: #f6f6f6;
241     border: none;
242     color: #0d0d0d;
243     padding: 15px 32px;
244     text-align: center;
245     text-decoration: none;
246     display: inline-block;
247     font-size: 16px;
248     margin: 5px;
249     width: 85%;
250     border: 2px solid #f6f6f6;
251     -webkit-transition: all 0.5s ease-in-out;
252     -moz-transition: all 0.5s ease-in-out;
253     -ms-transition: all 0.5s ease-in-out;
254     -o-transition: all 0.5s ease-in-out;
255     transition: all 0.5s ease-in-out;
256     -webkit-border-radius: 5px 5px 5px 5px;
257     border-radius: 5px 5px 5px 5px;
258 }
259
260 input[type=text]:focus {
261     background-color: #fff;
262     border-bottom: 2px solid #5fbae9;
```

```
263     }
264
265     input[type=text]:placeholder {
266         color: #cccccc;
267     }
268
269
270
271     /* ANIMATIONS */
272
273     /* Simple CSS3 Fade-in-down Animation */
274     .fadeInDown {
275         -webkit-animation-name: fadeInDown;
276         animation-name: fadeInDown;
277         -webkit-animation-duration: 1s;
278         animation-duration: 1s;
279         -webkit-animation-fill-mode: both;
280         animation-fill-mode: both;
281     }
282
283     @-webkit-keyframes fadeInDown {
284         0% {
285             opacity: 0;
286             -webkit-transform: translate3d(0, -100%, 0);
287             transform: translate3d(0, -100%, 0);
288         }
289         100% {
290             opacity: 1;
291             -webkit-transform: none;
292             transform: none;
293         }
294     }
295
296     @keyframes fadeInDown {
297         0% {
298             opacity: 0;
299             -webkit-transform: translate3d(0, -100%, 0);
300             transform: translate3d(0, -100%, 0);
301         }
302         100% {
303             opacity: 1;
304             -webkit-transform: none;
305             transform: none;
306         }
307     }
```

```
308
309  /* Simple CSS3 Fade-in Animation */
310  @-webkit-keyframes fadeIn { from { opacity:0; } to {
    opacity:1; } }
311  @-moz-keyframes fadeIn { from { opacity:0; } to {
    opacity:1; } }
312  @keyframes fadeIn { from { opacity:0; } to { opacity:1;
    } }
313
314  .fadeIn {
315      opacity:0;
316      -webkit-animation:fadeIn ease-in 1;
317      -moz-animation:fadeIn ease-in 1;
318      animation:fadeIn ease-in 1;
319
320      -webkit-animation-fill-mode:forwards;
321      -moz-animation-fill-mode:forwards;
322      animation-fill-mode:forwards;
323
324      -webkit-animation-duration:1s;
325      -moz-animation-duration:1s;
326      animation-duration:1s;
327  }
328
329  .fadeIn.first {
330      -webkit-animation-delay: 0.4s;
331      -moz-animation-delay: 0.4s;
332      animation-delay: 0.4s;
333  }
334
335  .fadeIn.second {
336      -webkit-animation-delay: 0.6s;
337      -moz-animation-delay: 0.6s;
338      animation-delay: 0.6s;
339  }
340
341  .fadeIn.third {
342      -webkit-animation-delay: 0.8s;
343      -moz-animation-delay: 0.8s;
344      animation-delay: 0.8s;
345  }
346
347  .fadeIn.fourth {
348      -webkit-animation-delay: 1s;
349      -moz-animation-delay: 1s;
```

```
350     animation-delay: 1s;
351 }
352
353 /* Simple CSS3 Fade-in Animation */
354 .underlineHover:after {
355     display: block;
356     left: 0;
357     bottom: -10px;
358     width: 0;
359     height: 2px;
360     background-color: #56baed;
361     content: "";
362     transition: width 0.2s;
363 }
364
365 .underlineHover:hover {
366     color: #0d0d0d;
367 }
368
369 .underlineHover:hover:after{
370     width: 100%;
371 }
372
373
374
375 /* OTHERS */
376
377 *:focus {
378     outline: none;
379 }
380
381 #icon {
382     width: 60%;
383 }
```



```
1 <?php
2 session_start();
3 require_once $_SERVER['DOCUMENT_ROOT'].'/config/mailparam.
  php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
  ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ESession.
  php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EEmail.php'
  ;
10 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelEmailSender.php';
13
14 require_once $_SERVER['DOCUMENT_ROOT'].'/swiftmailer5/lib/
  swift_required.php';
15
16
17 $Name = filter_input(INPUT_POST, 'CourtName',
  FILTER_SANITIZE_STRING);
18
19 $r = new ECourt();
20 $r->Name = $Name;
21 $r->Desc = "";
22 $r->Deleted = false;
23
24
25 if(ModelCourts::AddCourt($r))
26     echo '{ "ReturnCode": 0, "Message": ""}';
27 else
28     echo '{ "ReturnCode": 1, "Message": "Problème dans l\'
  enregistrement des données"}';
29
```


1 <?php

2


```

1 <?php
2 session_start();
3 require_once $_SERVER['DOCUMENT_ROOT'].'/config/mailparam.
  php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
  ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ESession.
  php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EEmail.php'
  ;
10 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelEmailSender.php';
13
14 require_once $_SERVER['DOCUMENT_ROOT'].'/swiftmailer5/lib/
  swift_required.php';
15
16
17 $id = filter_input(INPUT_POST, 'IdCourt',
  FILTER_SANITIZE_NUMBER_INT);
18 $dt = filter_input(INPUT_POST, 'Date',
  FILTER_SANITIZE_STRING);
19
20 $r = new EReservation();
21 $r->Court = ModelCourts::GetCourtById($id);
22 $r->User = ESession::GetUser();
23 $r->Date = date_parse($dt);
24
25 $r->Date = $r->Date["year"] . "-" . $r->Date["month"] . "-"
  . $r->Date["day"] . " " . $r->Date["hour"] . ":00:00";
26
27 if(ModelReservations::AddReservation($r))
28 {
29     echo '{ "ReturnCode": 0, "Message": ""}';
30     $Email = new EEmail($r->User->Email, "Nouvelle
  reservation");
31     $Email->Body = '<html>' .
32         ' <head></head>' .

```

```
33         ' <body>'.
34         ' <p>Vous avez réserver le ' . $r->Court->
    Name .' pour le : ' . $r->Date .
35         '</p>' .
36         ' </body>' .
37         '</html>';
38         ModelEmailSender::SendEmail($Email);
39     }
40     else
41         echo '{ "ReturnCode": 1, "Message": "Problème dans l\'
    enregistrement des données"}';
42
```

```
1 <?php
2 session_start();
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
6 ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
ModelReservations.php';
10 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
.php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
.php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
.php';
13
14
15 $reservations = ModelReservations::GetAllReservations();
16 $reservations = json_encode($reservations);
17 echo $reservations;
```



```
1 <?php
2 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 // Initialisation des variables et Récupération du contenu
  des champs passés en paramètres
5 $nickname = filter_input(INPUT_POST, 'Nickname',
  FILTER_SANITIZE_STRING);
6 $isConfirmed = filter_input(INPUT_POST, 'isConfirmed',
  FILTER_SANITIZE_NUMBER_INT);
7
8 // Validation des données avec base de données ou autre
9 if (ModelUsers::UpdateConfirmation($nickname, (int)
  $isConfirmed))
10     // si c'est valide
11     echo '{ "ReturnCode": 1, "Message": "Update réussie."}
  ';
12 else
13     // si c'est invalide on renvoie le code d'erreur et le
  message d'erreur
14     echo '{ "ReturnCode": 0, "Message": "Update invalide
  ."}';
15
16
```



```

1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwblQTTN6wy59ffFlBuGJpLSa9DkKmp0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12  </head>
13  <body>
14    <div class="wrapper fadeInDown">
15      <div id="formContent">
16        <!-- Tabs Titles -->
17
18        <!-- Icon -->
19        <div class="fadeIn first">
20          <h1>Squash reservation</h1>
21        </div>
22
23        <!-- Login Form -->
24        <form action="#" method="POST">
25          <input type="text" id="login" class="fadeIn
second" name="login" placeholder="login"><br/>
26          <input type="password" id="password" class="
fadeIn third" name="password" placeholder="password">
27          <input type="submit" class="fadeIn fourth"
value="Log In">
28        </form>
29
30        <!-- Remind Passowrd -->
31        <div id="formFooter">
32          <a class="underlineHover" href="?action=Register">
Register</a>
33        </div>
34
35      </div>
36    </div>

```

```
37         <!-- Optional JavaScript -->
38         <!-- jQuery first, then Popper.js, then Bootstrap JS
-->
39         <script src="https://code.jquery.com/jquery-3.3.1.min.
js"></script>
40         <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
wHAIffRlMFy6i5SRaxvfOCifBUQy1xHdJ/
yoi7FRNXMRBu5WHdZYu1hA6ZOblgut" crossorigin="anonymous"></
script>
41         <script src="https://stackpath.bootstrapcdn.com/
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
42     </body>
43 </html>
```

```

1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKMP0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12
13
14    <link href='./FullCalendar/core/main.css' rel='
stylesheet' />
15    <link href='./FullCalendar/daygrid/main.css' rel='
stylesheet' />
16    <link href='./FullCalendar/timegrid/main.css' rel='
stylesheet' />
17    <link href='./FullCalendar/list/main.css' rel='
stylesheet' />
18    <script src='./FullCalendar/core/main.js'></script>
19    <script src='./FullCalendar/interaction/main.js'></
script>
20    <script src='./FullCalendar/daygrid/main.js'></script>
21    <script src='./FullCalendar/timegrid/main.js'></script
>
22    <script src='./FullCalendar/list/main.js'></script>
23    <script src='./FullCalendar/interaction/main.js'></
script>
24
25    <link rel="stylesheet" href="http://cdn.dhtmlx.com/
edge/dhtmlx.css"
26    type="text/css">
27    <script src="http://cdn.dhtmlx.com/edge/dhtmlx.js"
28    type="text/javascript"></script>
29
30    <script>
31
32    var cal = null;

```

```

33  var mod = document.getElementById("myModal");
34  var span = document.getElementsByClassName("close")[0];
35  var myCalendar = null;
36
37  document.addEventListener('DOMContentLoaded', function()
{
38      var calendarEl = document.getElementById('calendar');
39      mod = document.getElementById("myModal");
40      span = document.getElementsByClassName("close")[0];
41      myCalendar = new dhtmlXCalendarObject("box");
42      myCalendar.show();
43      cal = new FullCalendar.Calendar(calendarEl, {
44          plugins: [ 'timeGrid', 'interaction' ],
45          header: {
46              left: 'prev,next today',
47              center: 'title',
48              right: 'timeGridWeek'
49          },
50          titleFormat: {
51              year: 'numeric', month: 'numeric', day: 'numeric'
52          },
53          defaultDate: '<?= date("o-m-t") ?>',
54          minTime: "<?= $BeginTime ?>:00:00",
55          maxTime: "<?= $EndTime + 1 ?>:00:00",
56          slotDuration: "01:00:00", // Ne pas changer, c'est
bon
57          navLinks: true, // can click day/week names to
navigate views
58          businessHours: true, // display business hours
59          editable: true,
60          dateClick: CalendarDateOnClicked,
61          events: [<?php
62              if(isset($reservations))
63              {
64                  for($i = 0; $i < count($reservations); $i++)
65                  {
66                      echo "{
67                          title: '" . $reservations[$i]->Court->Name .
68                          ",
69                          start: '" . $reservations[$i]->Date . "',
70                      }";
71                      if($i < count($reservations)-1)
72                          echo ",";
73                  }
74              }

```

```

74     ?>]
75     });
76     myCalendar.loadUserLanguage('fr');
77
78     $("#btnReservation").click(btnReservationOnClicked);
79
80     $.ajax({
81         method: 'POST',
82         url: './ajax/loadReservations.php',
83         dataType: 'json',
84         success: function (data) {
85             switch (data.ReturnCode)
86             {
87                 case 1:
88                     break;
89                 case 0:
90
91                     cal.addEvent(event);
92                     break;
93             }
94             }, // #end success
95         error: function (jqXHR) {
96             msg = "Une erreur est survenue. Error : "
97             switch(jqXHR.status){
98                 case 200 :
99                     msg = msg + jqXHR.status + " Le json
100                 retourné est invalide.";
101                     break;
102                 case 404 :
103                     msg = msg + jqXHR.status + " La page
104                 checklogin.php est manquante.";
105                     break;
106             } // #end switch
107             alert(msg);
108             } // #end error
109     });
110
111     span.onclick = function() {
112         mod.style.display = "none";
113     }
114     window.onclick = function(event) {
115         if (event.target == mod) {
116             mod.style.display = "none";
117         }
118     }

```

```

117     cal.render();
118 });
119
120
121 function CalendarDateOnClicked(info)
122 {
123
124     var event={id:1 , title: 'Test event', start: info.
dateStr};
125     var el = $("#calendar");
126     myCalendar.setDate(info.dateStr);
127     var time = info.date.getHours();
128     $("#selectTime").val(time);
129     myCalendar.show();
130     mod.style.display = "block";
131     //alert('Clicked on: ' + info.dateStr);
132     //alert('Coordinates: ' + info.jsEvent.pageX + ',' +
info.jsEvent.pageY);
133     //alert('Current view: ' + info.view.type);
134     //change the day's background color just for fun
135     //info.dayEl.style.backgroundColor = 'red';
136
137 }
138
139 function btnReservationOnClicked()
140 {
141     var Id = 1;
142     var date = myCalendar.getDate();
143     var hours = $("#selectTime option:selected").val();
144     date.setHours(hours);
145     var Title = $("#selectCourts option:selected").text();
146     var idCourt = $("#selectCourts option:selected").val();
147     //var reservation = '{"IdCourt":"' + idCourt + ', "Date
": "' + date + '"}';
148     //reservation = JSON.parse(reservation);
149     var event = {id: Id, title: Title, start: date};
150     $.ajax({
151         method: 'POST',
152         url: './ajax/saveReservation.php',
153         data: {'IdCourt': idCourt, 'Date': date},
154         dataType: 'json',
155         success: function (data) {
156             switch (data.ReturnCode)
157             {
158                 case 1:

```



```

159             break;
160             case 0:
161                 cal.addEvent(event);
162                 break;
163         }
164     }, // #end success
165     error: function (jqXHR) {
166         msg = "Une erreur est survenue. Error : "
167             switch(jqXHR.status){
168                 case 200 :
169                     msg = msg + jqXHR.status + " Le
170 json retourné est invalide.";
171                     break;
172                 case 404 :
173                     msg = msg + jqXHR.status + " La
174 page checklogin.php est manquante.";
175                     break;
176             } // #end switch
177         alert(msg);
178     } // #end error
179 });
180 mod.style.display = "none";
181 }
182
183 </script>
184 </head>
185 <body>
186     <?php if(isset($navBar)) echo $navBar; ?>
187
188     <div id='calendar'></div>
189     <div id="myModal" class="modal" tabindex="-1" role="
190 dialog">
191         <div class="modal-dialog" role="document">
192             <div class="modal-content">
193                 <div class="modal-header">
194                     <h5 class="modal-title">Modal title</h5>
195                     <button type="button" class="close" data-
196 dismiss="modal" aria-label="Close">
197                         <span aria-hidden="true">&times;</span>
198                     </button>
199                 </div>
200                 <div class="modal-body" style="width: 40vh">
201                     <div id="box" style="position:absolute; height:
202 250px;"></div>
203                     <select class="form-control" id="selectCourts"

```

```

198 style="float:right">
199     <?php
200         if(isset($courts))
201         {
202             for($i = 0; $i < count($courts); $i++)
203             {
204                 echo "<option value=\"" . $courts[$i]->Id
205                 . "\">" . $courts[$i]->Name . "</option>";
206             }
207         }
208     ?>
209 </select>
210 <select class="form-control" id="selectTime"
211 style="float:right">
212     <?php
213         if(isset($BeginTime) && isset($EndTime))
214         {
215             for($i = $BeginTime; $i <= $EndTime; $i++)
216             {
217                 echo "<option value=\"" . $i . "\">" .
218                 $i . "h</option>";
219             }
220         }
221     ?>
222 </select>
223 </div>
224 <div class="modal-footer">
225     <button type="button" class="btn btn-primary"
226     id="btnReservation">Reservation</button>
227     <button type="button" class="btn btn-secondary"
228     data-dismiss="modal">Close</button>
229 </div>
230 <!-- Optional JavaScript -->
231 <!-- jQuery first, then Popper.js, then Bootstrap JS
232 -->
233 <script src="https://code.jquery.com/jquery-3.3.1.min
234 .js"></script>
235 <script src="https://cdnjs.cloudflare.com/ajax/libs/

```

```
234 popper.js/1.14.6/umd/popper.min.js" integrity="sha384-  
wHAiFfRlMFy6i5SRaxvfOCifBUQy1xHdJ/  
yoi7FRNXMRBu5WHdZYulhA6ZOblgut" crossorigin="anonymous">  
  </script>  
235 <script src="https://stackpath.bootstrapcdn.com/  
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-  
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/  
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>  
236 </body>  
237 </html>
```



```

1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwblQTTN6wy59ffFlBuGJpLSa9DkKmp0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12  </head>
13  <body>
14    <div class="wrapper fadeInDown">
15      <div id="formContent">
16        <!-- Tabs Titles -->
17
18        <!-- Icon -->
19        <div class="fadeIn first">
20          <h1>Register</h1>
21        </div>
22
23        <!-- Login Form -->
24        <form action="?action=Register" method="POST">
25          <input type="text" id="nickname" class="fadeIn
second" name="nickname" placeholder="nickname">
26          <input type="password" id="password" class="
fadeIn third" name="password" placeholder="password">
27          <input type="text" id="email" class="fadeIn
fourth" name = "email" placeholder="email">
28          <input type="text" id="name" class="fadeIn
fifth" name = "name" placeholder="name">
29          <input type="text" id="firstname" class="
fadeIn seventh" name = "firstname" placeholder="firstname"
>
30          <input type="text" id="phone" class="fadeIn
eigth" name="phone" placeholder="phone">
31          <input type="submit" class="fadeIn fourth"
value="Register">
32        </form>

```

```
33
34         <!-- Remind Passowrd -->
35         <div id="formFooter">
36             <a class="underlineHover" href="?action=Login">Log
In</a>
37         </div>
38
39     </div>
40 </div>
41     <!-- Optional JavaScript -->
42     <!-- jQuery first, then Popper.js, then Bootstrap JS
-->
43     <script src="https://code.jquery.com/jquery-3.3.1.min.
js" integrity="sha384-q8i/X+
965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo
" crossorigin="anonymous"></script>
44     <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
wHAIffRlMFy6i5SRaxvfOCifBUQy1xHdJ/
yoi7FRNXMRBu5WHdZYulhA6ZOblgut" crossorigin="anonymous"></
script>
45     <script src="https://stackpath.bootstrapcdn.com/
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
46 </body>
47 </html>
```

```
1 <div class="myReservation table table-bordered">
2     <table>
3         <tr>
4             <td><?= $reservation->Court->Name ?></td>
5             <td><?= $reservation->Date ?></td>
6             <th><a href="?action=DeleteReservation&idCourt
=<?=$reservation->Court->Id?>&Nickname=<?=$reservation->
User->Nickname?>&date=<?=$reservation->Date?>">X</a></th>
7         <tr>
8             <table>
9 </div>
```



```

1 <!-- Navigation -->
2 <nav class="navbar navbar-expand-lg navbar-dark bg-dark
  static-top">
3   <div class="container">
4     <ul class="navbar-nav ml-auto">
5       <li class="nav-item active">
6         <a class="nav-link" href="?action=Accueil">
Home
7           <span class="sr-only">(current) </span>
8         </a>
9       </li>
10      <li class="nav-item">
11        <a class="nav-link" href="?action=
MyReservation">Mes réservations</a>
12      </li>
13    </ul>
14    <button class="navbar-toggler" type="button" data-
toggle="collapse" data-target="#navbarResponsive" aria-
controls="navbarResponsive" aria-expanded="false" aria-
label="Toggle navigation">
15      <span class="navbar-toggler-icon"></span>
16    </button>
17    <div class="collapse navbar-collapse" id="
navbarResponsive">
18      <ul class="navbar-nav ml-auto">
19        <li class="nav-item">
20          <a class="nav-link" href="#"><?= (
ESession::GetUser() == false) ? "" : ESession::GetUser()
->Nickname ?></a>
21        </li>
22        <li class="nav-item">
23          <a href="?action=Logout"><button class="btn
btn-light">Log out</button></a>
24        </li>
25        <?php
26          if(isset($doc))
27          {
28            ?>
29            <li>
30              <a href="<?=$doc?>" target="_blank">?</a>
31            </li>
32          <?php
33          }
34          else{
35            ?>

```

```
36         <li>
37             <a href="/view/DocUtilisateur/DocAccueil.
html" target="_blank">?</a>
38         </li>
39     <?php
40     }
41     ?>
42 </ul>
43 </div>
44 </div>
45 </nav>
```

```

1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwb1QTTN6wy59ffFlBuGJpLSa9DkKmp0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12    <script src="https://code.jquery.com/jquery-3.3.1.min.
js"></script>
13  </head>
14  <body>
15    <?php include $_SERVER['DOCUMENT_ROOT'].'/view/User/
NavBar.php'; ?>
16    <div class="wrapper fadeInDown">
17      <div id="formContent">
18        <!-- Tabs Titles -->
19
20        <!-- Icon -->
21        <div class="fadeIn first">
22          <h1>Mes réservations</h1>
23        </div>
24
25        <!-- Login Form -->
26        <?php
27          if(isset($reservations))
28            for($i = 0; $i < count($reservations); $i++)
29              {
30                $reservation = $reservations[$i];
31                include $_SERVER['DOCUMENT_ROOT'].'/view/
ReservationTemplate.php';
32              }
33
34        ?>
35
36      </div>
37    </div>

```

```
38         <!-- Optional JavaScript -->
39         <!-- jQuery first, then Popper.js, then Bootstrap JS
-->
40         <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
wHAIffRlMFy6i5SRaxvfOCifBUQy1xHdJ/
yoi7FRNXMRBu5WHdZYulhA6ZOblgut" crossorigin="anonymous"></
script>
41         <script src="https://stackpath.bootstrapcdn.com/
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
42     </body>
43 </html>
```

```

1 <!-- Navigation -->
2 <nav class="navbar navbar-expand-lg navbar-dark bg-dark
  static-top">
3   <div class="container">
4     <ul class="navbar-nav ml-auto">
5       <li class="nav-item active">
6         <a class="nav-link" href="?action=Accueil">
          Home
7           <span class="sr-only">(current) </span>
8         </a>
9       </li>
10      <li class="nav-item">
11        <a class="nav-link" href="?action=ShowAllUsers
12        ">Utilisateurs</a>
13      </li>
14      <li class="nav-item">
15        <a class="nav-link" href="?action=
16        Preferences">Paramètres</a>
17      </li>
18    </ul>
19    <button class="navbar-toggler" type="button" data-
20    toggle="collapse" data-target="#navbarResponsive" aria-
21    controls="navbarResponsive" aria-expanded="false" aria-
22    label="Toggle navigation">
23      <span class="navbar-toggler-icon"></span>
24    </button>
25    <div class="collapse navbar-collapse" id="
26    navbarResponsive">
27      <ul class="navbar-nav ml-auto">
28        <li class="nav-item">
29          <a class="nav-link" href="#"><?= (ESession
30          ::getUser() == false) ? "" : ESession::getUser()->
31          Nickname ?></a>
32        </li>
33        <li class="nav-item">
34          <a href="?action=Logout"><button class="btn
35          btn-light">Log out</button></a>
36        </li>
37        <li class="nav-item">
38          <?php
39            if(isset($doc))
40            {
41              <?
42            </li>
43            <a href="<?=$doc?>" target="_blank">?</a>
44          </li>

```

```
35         <?php
36     }
37     ?>
38     </ul>
39     </div>
40 </div>
41 </nav>
```

```

1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwb1QTTN6wy59ffFlBuGJpLSa9DkKMP0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <script src="https://code.jquery.com/jquery-3.3.1.min.
js"></script>
12    <title>Squash</title>
13    <script>
14      $(document).ready(function(){
15        $('input[type="checkbox"]').click(function(){
16          var nickname = $(this).val();
17          var checked = $(this).prop("checked");
18          $.ajax({
19            method: 'POST',
20            url: './ajax/changeConfirmation.php',
21            data: {'Nickname': nickname , 'isConfirmed':
Number(checked)},
22            dataType: 'json',
23            success: function (data) {
24              switch (data.ReturnCode)
25              {
26                case 1:
27                  break;
28                case 0:
29                  alert(data.Message);
30                  break;
31              }
32            }, // #end success
33            error: function (jqXHR) {
34              msg = "Une erreur est survenue. Error : "
35              switch(jqXHR.status){
36                case 200 :
37                  msg = msg + jqXHR.status + " Le
json retourné est invalide.";

```

```

38             break;
39             case 404 :
40                 msg = msg + jqXHR.status + " La
page checklogin.php est manquante.";
41                 break;
42             } // #end switch
43             alert(msg);
44             } // #end error
45         });
46     });
47 });
48 </script>
49 </head>
50 <body>
51     <?php include $_SERVER['DOCUMENT_ROOT'].'/view/Admin/
NavBar.php'; ?>
52     <table class="table">
53     <tr>
54         <th scope="col">Nickname</th>
55         <th scope="col">Name + First name</th>
56         <th scope="col">Phone</th>
57         <th scope="col">Email</th>
58         <th scope="col">Confirmé</th>
59         <th scope="col"></th>
60     </tr>
61     <?php
62     if(isset($users))
63     {
64         for($i = 0; $i < count($users); $i++)
65         {
66             $user = $users[$i];
67             include $_SERVER['DOCUMENT_ROOT'].'/view/Admin
/OneListUser.php';
68         }
69     }
70     ?>
71     </table>
72     <!-- Optional JavaScript -->
73     <!-- jQuery first, then Popper.js, then Bootstrap JS
-->
74     <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
wHAIffRlMFy6i5SRaxvfOCifBUQy1xHdJ/
yoi7FRNXMRBu5WHdZYulhA6ZOblgut" crossorigin="anonymous"></
script>

```



```
75     <script src="https://stackpath.bootstrapcdn.com/
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
76 </body>
77 </html>
```



```

1 <!doctype html>
2 <html lang="en">
3 <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
    initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
    bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
    integrity="sha384-
    GJzZqFGwblQTTN6wy59ffFlBuGJpLSa9DkKMP0DgiMDm4iYMj70gZWKYbI
    706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12 </head>
13 <body>
14     <?php include $_SERVER['DOCUMENT_ROOT'].'/view/Admin/
    NavBar.php'; ?>
15     <div class="wrapper fadeInDown">
16     <div id="formContent">
17         <!-- Tabs Titles -->
18
19         <!-- Icon -->
20         <div class="fadeIn first">
21             <h1>Profil</h1>
22         </div>
23
24         <table class="table">
25             <tr>
26                 <td> Name : <?= $user->Name ?> </td>
27                 <td> Firstname : <?= $user->FirstName ?> </td>
28             </tr>
29             <tr>
30                 <td> Email : <?= $user->Email ?> </td>
31                 <td> Phone : <?= $user->Phone ?> </td>
32             </tr>
33         </table>
34         <!-- Login Form -->
35         <?php
36             if(isset($reservations))
37             for($i = 0; $i < count($reservations); $i++)
38             {
39                 $reservation = $reservations[$i];

```

```
40         include $_SERVER['DOCUMENT_ROOT'].'/view/
ReservationTemplate.php';
41     }
42
43     ?>
44
45     </div>
46     </div>
47     <!-- Optional JavaScript -->
48     <!-- jQuery first, then Popper.js, then Bootstrap JS
-->
49     <script src="https://code.jquery.com/jquery-3.3.1.min.
js" integrity="sha384-q8i/X+
965DzO0rT7abK41JStQIAqVgRVzpbzo5smXKp4YfRvH+8abtTE1Pi6jizo
" crossorigin="anonymous"></script>
50     <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
wHAiFfRlMFy6i5SRaxvfOCifBUQy1xHdJ/
yoi7FRNXMRBu5WHdZYulhA6ZOblgut" crossorigin="anonymous"></
script>
51     <script src="https://stackpath.bootstrapcdn.com/
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
52 </body>
53 </html>
```

```
1 <tr>
2   <td scope="row"><?= isset($user->Nickname) ? $user->
  Nickname : "" ?> </td>
3   <td><?= (isset($user->Name) && isset($user->FirstName)
  ) ? $user->Name . $user->FirstName : "" ?> </td>
4   <td><?= isset($user->Phone) ? $user->Phone : "" ?> </
  td>
5   <td><?= isset($user->Email) ? $user->Email : "" ?> </
  td>
6   <td><input type="checkbox" value="<?= isset($user->
  Nickname) ? $user->Nickname : "" ?>" id="scales" name="
  scales" <?= (isset($user->IsConfirmed) && $user->
  IsConfirmed == 1) ? "checked" : "" ?>></td>
7   <td><a href="?action=UserProfil&Nickname=<?= $user->
  Nickname?>"> Paramètres </a></td>
8 </tr>
```



```

1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwblQTTN6wy59ffFlBuGJpLSa9DkKmp0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <script src="https://code.jquery.com/jquery-3.3.1.min.
js"></script>
12    <title>Squash</title>
13    <script>
14      var listItems = [];
15      $(document).ready(function() {
16        $('#btnDeleteCourt').click(function() {
17          var courtName = $('#selectCourt option:selected'
).val();
18          $.ajax({
19            method: 'POST',
20            url: './ajax/deleteCourt.php',
21            data: {'Nickname': courtName},
22            dataType: 'json',
23            success: function (data) {
24              switch (data.ReturnCode)
25              {
26                case 1:
27                  break;
28                case 0:
29                  alert(data.Message);
30                  break;
31              }
32            }, // #end success
33            error: function (jqXHR) {
34              msg = "Une erreur est survenue. Error : "
35              switch(jqXHR.status){
36                case 200 :
37                  msg = msg + jqXHR.status + " Le
json retourné est invalide.";

```

```

38             break;
39         case 404 :
40             msg = msg + jqXHR.status + " La
page checklogin.php est manquante.";
41             break;
42         } // #end switch
43         alert(msg);
44         } // #end error
45     });
46 }); //#end delete court click
47
48 $('#btnAddCourt').click(function(){
49
50     // On remplit le tableau avec les courts
existants
51     listItems = [];
52     $('#selectCourt').children().each(function () {
53         listItems.push($(this).val());
54     });
55     $('#edtCourtName').val("");
56     $('#edtCourtName').show();
57
58     $('#edtCourtName').focus();
59
60 }); //#end add court click
61
62 $('#btnDeleteCourt').click(function(){
63
64     var s = $('#selectCourt option:selected').val();
65
66 }); //#end add court click
67
68 $('#edtCourtName').keyup(function() {
69
70     var s = $(this).val();
71
72     // Return pressé?
73     if ( event.which == 13 ) {
74         event.preventDefault();
75         // Ajouter le court en ajax si pas encore
utilisé
76         if (listItems.indexOf(s) < 0){
77             // Appel fonction ajout cour
78             $.ajax({
79                 method: 'POST',

```



```

80         url: './ajax/addCourt.php',
81         data: {'CourtName': s},
82         dataType: 'json',
83         success: function (data) {
84             switch (data.ReturnCode)
85             {
86                 case 1:
87                     break;
88                 case 0:
89                     alert(data.Message);
90                     break;
91             }
92             }, // #end success
93         error: function (jqXHR) {
94             msg = "Une erreur est survenue.
95             Error : "
96
97             switch(jqXHR.status){
98                 case 200 :
99                     msg = msg + jqXHR.status
100             + " Le json retourné est invalide.";
101                     break;
102                 case 404 :
103                     msg = msg + jqXHR.status
104             + " La page checklogin.php est manquante.";
105                     break;
106             } // #end switch
107             alert(msg);
108             } // #end error
109         });
110         // On cache l'edit
111         $('#edtCourtName').hide();
112     }
113
114     // Escape pressé?
115     if ( event.which == 27 ) {
116         event.preventDefault();
117
118         $('#edtCourtName').hide();
119     }
120
121     // Est-ce vide ?
122     if (s.length > 0)

```

```

122         $(this).css('background-color', '');
123     else
124         $(this).css('background-color', 'red');
125
126     // Est-ce déjà utilisé ?
127     if (listItems.indexOf(s) < 0)
128         $(this).css('color', 'black');
129     else
130         $(this).css('color', 'red');
131
132     });
133
134
135     }); // #end document ready
136 </script>
137 </head>
138 <body>
139     <?php include $_SERVER['DOCUMENT_ROOT'].'/view/Admin/
140     NavBar.php'; ?>
141     <div class="wrapper fadeInDown">
142         <div id="formContent">
143             <!-- Tabs Titles -->
144
145             <!-- Icon -->
146             <div class="fadeIn first">
147                 <h1>Préférences</h1>
148
149             <table class="table">
150                 <tr>
151                     <td colspan = "3">
152                         <select id="selectCourt" name="Courts"
153                         multiple>
154                             <?php
155                             if(isset($courts))
156                             {
157                                 for($i = 0; $i < count($courts); $i
158                                 ++){
159                                     {
160                                         echo "<option>" . $courts[$i]->
161                                         Name . "</option>";
162                                     }
163                                 }
164                             }
165                             ?>
166                         </select>
167                     </td>

```

```

163         </tr>
164     <tr>
165         <td colspan="2"><input type="button" id
166         ="btnAddCourt" value="Add" style="width:30px"/><input
167         type="text" id="edtCourtName" style="display:none"></td>
168         <td><input type="button" id="
169         btnDeleteCourt" value="Delete" style="width:30px"/></td>
170     </tr>
171     <tr>
172         <td colspan="3"><h3>Paramètres généraux
173         <h3></td>
174     </tr>
175     <form method="POST" action="?action=
176     ModifyPreference" >
177     <tr>
178         <td colspan="2"> Horaire d'ouverture
179         : </td>
180         <td> <input type="time" name="
181         openingTime" value="<?= $preferences->BeginTime ?>"></td>
182     </tr>
183     <tr>
184         <td colspan="2"> Horaire de fermeture
185         : </td>
186         <td> <input type="time" name="
187         closingTime" value="<?= $preferences->EndTime ?>"></td>
188     </tr>
189     <tr>
190         <td colspan="2"> Nombre de
191         réservations par personne : </td>
192         <td> <input type="number" name="
193         nbReservationByUser" value="<?= $preferences->
194         NbReservation ?>"></td>
195     </tr>
196     <tr>
197         <td colspan="2"> </td>
198         <td></td>
199     </tr>
200     <td colspan="2"><input type="submit"></
201     td>
202 </tr>
203 </form>
204 </table>
205 </div>
206 </div>
207 <!-- Optional JavaScript -->
208 <!-- jQuery first, then Popper.js, then Bootstrap JS

```

```
194 -->
195
196     <script src="https://cdnjs.cloudflare.com/ajax/libs/
popper.js/1.14.6/umd/popper.min.js" integrity="sha384-
wHAIffRlMFy6i5SRaxvfOCifBUQy1xHdJ/
yoi7FRNXMRBu5WHdZYulhA6ZOblgut" crossorigin="anonymous">
</script>
197     <script src="https://stackpath.bootstrapcdn.com/
bootstrap/4.2.1/js/bootstrap.min.js" integrity="sha384-
B0UglyR+jN6CkvvICOB2joaf5I413gm9GU6Hc1og6Ls7i6U/
mkkaduKaBhlAXv9k" crossorigin="anonymous"></script>
198 </body>
199 </html>
```

```
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKmp0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12  </head>
13  <body>
14    <div class="text-center mb-4" id="DivEmailSend"><
h1>Your account has been confrimed</h1><br><p>You can now
connect to our website : <a href="index.php">login</a></p>
</div>
15  </body>
16 </html>
```



```
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <!-- Required meta tags -->
5     <meta charset="utf-8">
6     <meta name="viewport" content="width=device-width,
initial-scale=1, shrink-to-fit=no">
7
8     <!-- Bootstrap CSS -->
9     <link rel="stylesheet" href="https://stackpath.
bootstrapcdn.com/bootstrap/4.2.1/css/bootstrap.min.css"
integrity="sha384-
GJzZqFGwb1QTTN6wy59ffF1BuGJpLSa9DkKmp0DgiMDm4iYMj70gZWKYbI
706tWS" crossorigin="anonymous">
10    <link rel="stylesheet" href="css/css.css">
11    <title>Squash</title>
12  </head>
13  <body>
14    <div class="text-center mb-4" id="DivEmailSend"><
h1>An email has been send</h1><br><p>To complete your
inscrption, we have send you an email. Please check your
email to validate your account</p></div>
15  </body>
16 </html>
```



```
1 <html>
2   <head>
3     <title>Accueil documentation</title>
4   </head>
5   <body>
6
7     </src>
8     </src>
9   </body>
10 </html>
```



```
1 <html>
2   <head>
3     <title>Documentation Profils</title>
4   </head>
5   <body>
6     </src>
7   </body>
8 </html>
```



```
1 <html>
2   <head>
3     <title>Documentation Mes Reservations</title>
4   </head>
5   <body>
6     </src>
7   </body>
8 </html>
```



```
1 <html>
2   <head>
3     <title>Paramètre documentation</title>
4   </head>
5   <body>
6     </src>
7   </body>
8 </html>
```



```
1 <html>
2   <head>
3     <title>Documentation Profil user</title>
4   </head>
5   <body>
6     </src>
7   </body>
8 </html>
```



```

1  <?php
2  /**
3   * @brief    Objet Role
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new ERole();
8   *          $u->IdRole = 1;
9   *          $u->Name = "Admin";
10  *
11  *          Exemple d'utilisation 2
12  *          $u = new ERole(1, "Admin");
13  */
14  class ERole {
15      /**
16       * @brief    Le Constructor appelé au moment de la
17       * création de l'objet. Ie. new ERole();
18       * @param InCodeRole Le code du rôle
19       * @param InName      Le nom du rôle. (Optionel) Defaut
20       * ""
21       */
22      public function __construct($InCodeRole, $InName = "")
23      {
24          $this->CodeRole = $InCodeRole;
25          $this->Label = $InName;
26      }
27
28      /**
29       * @var int code du Role
30       */
31      public $CodeRole;
32
33      /**
34       * @var string Le nom du role
35       */
36      public $Label;
37  }
38
39  ?>

```



```

1  <?php
2  /**
3   * @brief    Objet User
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new EUser();
8   *          $u->Email = "loic@burnand.com";
9   *          $u->Nickname = "Test";
10  *          $u->Name = "BURNAND";
11  *          $u->FirstName = "Loïc";
12  *          $u->Phone = "0767756644";
13  *          $u->Role = new ERole();
14  *          $u->IsConfirmed = false;
15  *
16  *          Exemple d'utilisation 2
17  *          $u = new EUser("loic@burnand.com", "Test", "
BURNAND", "Loïc", "0767756644", new ERole(), false);
18  */
19  class EUser {
20      /**
21       * @brief Le Constructor appelé au moment de la création
de l'objet. Ie. new EUser();
22       * @param InEmail      L'email de l'utilisateur. (
Optional) Default ""
23       * @param InNickname    Le nickname de l'utilisateur.
(Optional) Default ""
24       * @param InName        Le nom de l'utilisateur. (
Optional) Default ""
25       * @param InFirstName    Le prénom de l'utilisateur. (
Optional) Default ""
26       * @param InPhone        Le numéro de téléphone de l'
utilisateur. (Optional) Default ""
27       * @param InRole        Le role de l'utilisateur. (
Optional) Default null
28       * @param InCofirmation  Est-ce que l'utilisateur est
confirmé. (Optional) Default false
29       */
30       public function __construct($InEmail = "", $InNickname
= "", $InName = "", $InFirstName = "", $InPhone = "",
$InRole = null, $InCofirmation = false)
31       {
32           $this->Email = $InEmail;
33           $this->Nickname = $InNickname;
34           $this->Name = $InName;

```

```

35         $this->FirstName = $InFirstName;
36         $this->Phone = $InPhone;
37         $this->Role = $InRole;
38         $this->IsConfirmed = $InConfirmation;
39     }
40     /**
41      * @var string L'email de l'utilisateur
42      */
43     public $Email;
44     /**
45      * @var string Le nickname de l'utilisateur
46      */
47     public $Nickname;
48     /**
49      * @var string Le nom de l'utilisateur
50      */
51     public $Name;
52
53     /**
54      * @var string Le prénom de l'utilisateur
55      */
56     public $FirstName;
57
58
59     /**
60      * @var string Le mot de passe de l'utilisateur
61      * cette propriété sert principalement pour l'ajout d'
utilisateur
62      * et jamais le mot de passe est stocké en clair
63      */
64     public $Password;
65
66     /**
67      * @var string le numéro de téléphone
68      */
69     public $Phone;
70
71     /**
72      * @var int le rôle de l'utilisateur
73      */
74     public $Role;
75
76     /**
77      * @var bool Confirmation de l'inscription de l'
utilisateur

```

```
78      */
79      public $IsConfirmed;
80  }
81
82
83
84  ?>
```



```

1  <?php
2  /**
3   * @brief    Objet Court
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new ECourt();
8   *          $u->Name = "Court Test";
9   *          $u->Desc = "Ce terrain est un terrain reserver
    au entrainement";
10  *          $u->Deleted = true;
11  *
12  *          Exemple d'utilisation 2
13  *          $u = new ECourt("Court Test", "Ce terrain est
    un terrain reserver au entrainement", true);
14  */
15  class ECourt {
16      /**
17       * @brief    Le Constructor appelé au moment de la création
    de l'objet. Ie. new ECourt();
18       * @param InEmail    L'email du Court. (Optionel)
    Default ""
19       * @param InDesc    La description du court. (Optionel)
    Default ""
20       * @param InDeleted Est-ce que le court est fermé ou pas.
    (Optionel) Default false
21       */
22       public function __construct($InId = "", $InName = "",
    $InDesc = "", $InDeleted = false)
23       {
24           $this->Id = $InId;
25           $this->Name = $InName;
26           $this->Desc = $InDesc;
27           $this->Deleted = $InDeleted;
28       }
29
30      /**
31       * @var int L'id du court
32       */
33       public $Id;
34
35      /**
36       * @var string Le nom du court
37       */
38       public $Name;

```

```
39
40     /**
41      * @var string La description du court
42      */
43     public $Desc;
44
45     /**
46      * @var bool Est-ce que le court est fermé ou pas
47      */
48     public $Deleted;
49 }
50
51
52 ?>
```

```

1  <?php
2  /**
3   * @brief    Objet Email
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new EEmail();
8   *          $u->Subject = "Sujet de l'email";
9   *          $u->From = "tpiSquash@gmail.com";
10  *          $u->To = "test@example.com";
11  *          $u->Body = "<p>Corps du message</p>"
12  *
13  *          Exemple d'utilisation 2
14  *          $u = new ECourt("test@example.com", "Sujet de
15  l'email", "<p>Corps du message</p>", "tpiSquash@gmail.com"
16  );
17  */
18  class EEmail {
19      /**
20       * @brief    Le Constructor appelé au moment de la
21       création de l'objet. Ie. new EEmail();
22       * @param InSubject    Le sujet de l'email. (Optionel
23       ) Default ""
24       * @param InFrom      L'email de l'envoyeur. (Optionel)
25       Default ""
26       * @param InTo        Le destinataire du mail
27       * @param InBody      Le corps du message. (Optionel
28       ) Default ""
29       */
30      public function __construct($InTo, $InSubject = "",
31      $InBody = "", $InFrom = "")
32      {
33          $this->Subject = $InSubject;
34          $this->From = $InFrom;
35          $this->To = $InTo;
36          $this->Body = $InBody;
37      }
38
39      /**
40       * @var string Le sujet de l'email
41       */
42      public $Subject;
43
44      /**
45       * @var string L'email de l'envoyeur

```

```
39     */
40     public $From;
41
42     /**
43      * @var string L'email du destinataire
44      */
45     public $To;
46
47     /**
48      * @var string Le corps de l'email
49      */
50     public $Body;
51 }
52 ?>
```

```

1  <?php
2  /**
3   * @brief    Objet Token
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new EToken();
8   *          $u->Nickname = "Test";
9   *          $u->ValidateTill = "2000-00-00 00:00:00";
10  *
11  *          Exemple d'utilisation 2
12  *          $u = new EToken("Test", "2000-00-00 00:00:00")
13  */
14
15  class EToken
16  {
17      /**
18       * @brief    Le Constructor appelé au moment de la
19       création de l'objet. Ie. new EToken();
20       * @param InNickname      Le nickname stocké dans le
21       token
22       * @param InValidateTill    La date d'expiration. (
23       Optionel) Défaut ""
24       * @param InCode            Le code du token. (
25       Optionel) Défaut ""
26       */
27      public function __construct($InNickname,
28      $InValidateTill = "", $InCode = "")
29      {
30          $this->Nickname = $InNickname;
31          $this->ValidateTill = $InValidateTill;
32          $this->Code = $InCode;
33      }
34
35      /**
36       * @var string Le nickname du user associé à ce token
37       */
38      public $Nickname;
39
40      /**
41       * @var string La date d'expiration du token
42       */
43      public $ValidateTill;
44
45      /**
46       * @var string Le code du token
47       */
48      public $Code;
49  }

```

```
40     /**
41      * @var string Le code du token
42      */
43     public $Code;
44 }
```

```
1 <?php
2
3 class ESession {
4
5     /**
6      * @brief Assigner un objet EUser dans la session
7      * @param EUser user l'objet user
8      */
9     public static function SetUser($user)
10    {
11        $_SESSION['User'] = serialize($user);
12    }
13
14    /**
15     * @brief Retourne l'objet EUser stocké en session
16     * @return EUser user l'objet user
17     */
18    public static function GetUser()
19    {
20        if (isset($_SESSION['User']))
21        {
22            return unserialize($_SESSION['User']);
23        }
24        return false;
25    }
26 }
27
```



```

1  <?php
2
3  // Projet: TPI
4  // Script: Modèle roles.php
5  // Description: Classe de gestions de la table roles
6  // Auteur: Loïc Burnand
7  // Version 1.0 PC 9.5.2019 / Codage initial
8
9
10 class ModelRoles
11 {
12     /**
13      * @brief Retourne le tableau des roles de type ERole
14      * @return [array of ERole] Le tableau de ERole
15      */
16     static function GetAllRoles()
17     {
18         // On crée un tableau qui va contenir les objets
19         ERole $arr = array();
20
21         $s = "SELECT `Code`, `Label` FROM `tpi`.`ROLES`";
22         $statement = EDatabase::prepare($s,array(PDO::
23 ATTR_CURSOR => PDO::CURSOR_FWDONLY));
24         try {
25             $statement->execute();
26         }
27         catch (PDOException $e) {
28             echo 'Problème de lecture de la base de
29 données: '.$e->getMessage();
30             return false;
31         }
32         // On parcourt les enregistrements
33         while ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO
34 ::FETCH_ORI_NEXT)){
35             // On crée l'objet ERole en l'initialisant
36             avec les données provenant
37             // de la base de données
38             $u = new ERole($row['Code'], $row['Label']);
39             // On place l'objet ERole créé dans le tableau
40             array_push($arr, $u);
41         }
42         // On retourne le tableau contenant les
43         utilisateurs sous forme EUser
44         return $arr;

```

```

40     }
41
42     /**
43      * @brief Chercher un role par son Code dans la table
44      * @param Code Le code a recherché dans la base de
donnée
45      * @return [ERole] Retourne le premier rôle du
résultat de la requête
46      */
47     static function GetRoleByCode ($Code)
48     {
49         $s = "SELECT `Code`, `Label` FROM `tpi`.`ROLES`
WHERE `Code` = :e";
50
51         $statement = EDatabase::prepare($s,array(PDO::
ATTR_CURSOR => PDO::CURSOR_FWDONLY));
52         try {
53             $statement->execute(array(':e' => $Code));
54         }
55         catch (PDOException $e) {
56             echo 'Problème de lecture de la base de
données: '.$e->getMessage();
57             return false;
58         }
59         // On parcourt les enregistrements
60         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
FETCH_ORI_NEXT)){
61             // On crée l'objet ERole en l'initialisant
avec les données provenant
62             // de la base de données et on le retourne
63             return new ERole($row['Code'], $row['Label']);
64         }
65         // On retourne null si il n'y a pas de role
66         return null;
67     }
68 }
69
70

```

```

1  <?php
2
3  // Projet: TPI
4  // Script: Modèle users.php
5  // Description: Classe de gestions de la table users
6  // Auteur: Loïc Burnand
7  // Version 1.0 PC 9.5.2019 / Codage initial
8
9
10 class ModelUsers
11 {
12     /**
13      * @brief Retourne le tableau des utilisateurs de type
14      EUser
15      * @return [array of EUser] Le tableau de EUser
16      */
17     static function GetAllUsers()
18     {
19         // On crée un tableau qui va contenir les objets
20         EUser
21         $arr = array();
22
23         $s = "SELECT `Nickname`, `Name`, `Firstname`, `
24 Phone`, `email`, `IsConfirmed`, `CodeRole` FROM `tpi`.`
25 USERS`";
26         $statement = EDatabase::prepare($s,array(PDO::
27 ATTR_CURSOR => PDO::CURSOR_FWDONLY));
28         try {
29             $statement->execute();
30         }
31         catch (PDOException $e) {
32             echo 'Problème de lecture de la base de
33 données: '.$e->getMessage();
34             return false;
35         }
36         // On parcourt les enregistrements
37         while ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO
38 ::FETCH_ORI_NEXT)){
39             // On crée l'objet EUser en l'initialisant
40             avec les données provenant
41             // de la base de données
42             $u = new EUser($row['email'], $row['Nickname']
43 , $row['Name'], $row['Firstname'], $row['Phone'], new
44 ERole($row['CodeRole'], $row['IsConfirmed']));
45             // On place l'objet EUser créé dans le tableau

```

```

36         array_push($arr, $u);
37     }
38     // On retourne le tableau contenant les
    utilisateurs sous forme EUser
39     return $arr;
40 }
41
42 /**
43  * @brief Cherche un utilisateur par son Id dans la
    table
44  * @param nickname Le nickname a recherché dans la
    base de donnée
45  * @return [EUser] Retourne le premier utilisateur du
    résultat de la requête
46  */
47     static function GetUserByNickname($nickname)
48     {
49         $s = "SELECT `Nickname`, `Name`, `Firstname`, `
    Phone`, `email`, `IsConfirmed`, `CodeRole` FROM `tpi`.`
    USERS` WHERE `Nickname` = :e";
50
51         $statement = EDatabase::prepare($s,array(PDO::
    ATTR_CURSOR => PDO::CURSOR_FWDONLY));
52         try {
53             $statement->execute(array(':e' => $nickname));
54         }
55         catch (PDOException $e) {
56             echo 'Problème de lecture de la base de
    données: '.$e->getMessage();
57             return false;
58         }
59         // On parcourt les enregistrements
60         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
    FETCH_ORI_NEXT)){
61             // On crée l'objet EUser en l'initialisant
    avec les données provenant
62             // de la base de données et on le retourne
63             return new EUser($row['email'], $row['Nickname
    '], $row['Name'], $row['Firstname'], $row['Phone'],
    ModelRoles::GetRoleByCode($row['CodeRole']), $row['
    IsConfirmed']);
64         }
65         // On retourne null si il n'y a pas d'utilisateur
66         return null;
67     }

```

```

68
69     /**
70      * @brief Ajoute un utilisateur dans la base de
donnée
71      * @param User l'utilisateur a ajouté sous forme de
EUser
72      * @return [bool] Retourne true si l'ajout est réussi
, sinon retourne false
73      */
74     static function AddUser($user)
75     {
76         if($user->Role == null)
77             $user->Role = new ERole(2);
78         $s = "INSERT INTO `tpi`.`users`(`Nickname`, `Name`
, `Firstname`, `Password`, `Phone`, `email`, `
IsConfirmed`, `CodeRole`) VALUES (:nc, :na, :fn, :pw , :
ph, :e, 0, :cr);";
79         $statement = EDatabase::prepare($s);
80         try {
81             $statement->execute(array(':nc' => $user->
Nickname, ':na' => $user->Name, ':fn' => $user->FirstName
, ':pw' => $user->Password, ':ph' => $user->Phone, ':e'
=> $user->Email, ':cr' => $user->Role->CodeRole ));
82         }
83         catch (PDOException $e) {
84             echo 'Problème d\'insertion dans la base de
données: '. $e->getMessage();
85             return false;
86         }
87         return ModelTokens::CreateToken($user->Nickname);
88     }
89
90     /**
91      * @brief Permet de mettre à jours les donnée d'un
utilisateur
92      * @param User l'utilisateur que l'on souhaite
modifié
93      * @return [bool] Retourne true si l'update est
réussi, sinon retourne false
94      */
95     static function UpdateUser($user)
96     {
97         $s = "UPDATE `TPI`.`USERS` SET `Name` = :na, `
Firstname` = :fn, `Phone` = :ph, `email` = :e, `
IsConfirmed` = :ic, `CodeRole` = :cr WHERE `Nickname` = :

```

```

97  nc";
98      $statement = EDatabase::prepare($s);
99      try {
100          $statement->execute(array(':nc' => $user->
Nickname, ':na' => $user->Name, ':fn' => $user->FirstName
, ':ph' => $user->Phone, ':e' => $user->Email, ':ic' =>
$user->IsConfirmed, ':cr' => $user->Role->CodeRole ));
101      }
102      catch (PDOException $e) {
103          echo 'Problème de mise à jour dans la base de
données: '.$e->getMessage();
104          return false;
105      }
106      // Ok
107      return true;
108  }
109
110  /**
111   * @brief Retourne le nom du rôle de l'utilisateur
112   * @param User L'utilisateur du quel on souhaite
récupérer le rôle
113   * @return [string] Le nom du rôle
114   */
115  static function GetUserRole($User)
116  {
117      if(!isset($User->Role->CodeRole))
118      {
119          // si le code du role n'est pas renseigné on
essaie d'utiliser le
120          // nickname pour récupérer toutes les infos
sur l'utilisateurs (dont l'id du role)
121          if(isset($User->Nickname))
122          {
123              $User = ModelUsers::GetUserByNickname(
$user->Nickname);
124          }
125          else
126          {
127              return false;
128          }
129      }
130      return ModelRoles::GetRoleByCode($User->Role->
CodeRole);
131  }
132

```

```

133     /**
134      * @brief Retourne true si le password et le nickname
        sont dans la base
135      * @param User De type EUser, doit contenir le
        nickname et le password
136      * @return [bool] Retourne true si l'utilisateur et
        le password correspondent
137     */
138     static function CheckLogin($User)
139     {
140         $s = "SELECT COUNT(*) FROM `tpi`.`USERS` WHERE `
        Nickname` = :e AND `Password` = :p AND `IsConfirmed` = 1"
        ;
141
142         $statement = EDatabase::prepare($s,array(PDO::
        ATTR_CURSOR => PDO::CURSOR_FWDONLY));
143         try {
144             $statement->execute(array(':e' => strtolower(
        $User->Nickname), ':p' => $User->Password));
145         }
146         catch (PDOException $e) {
147             echo 'Problème de lecture de la base de
        données: '.$e->getMessage();
148             return false;
149         }
150         return ($statement->fetch()["COUNT(*)"] > 0) ?
        true : false;
151     }
152
153
154     /**
155      * @brief Update le champ IsConfirmed d'un
        utilisateur
156      * @param nickname De type string, contient le
        nickname de l'utilisateur à update
157      * @param isConfirmed De type Boolean, contient la
        valeur du champ à update
158      * @return [bool] Retourne true si l'update a réussie
159     */
160     static function UpdateConfirmation($nickname,
        $isConfirmed)
161     {
162         $s = "UPDATE `TPI`.`USERS` SET `IsConfirmed` = :
        ic WHERE `Nickname` = :nc";
163         $statement = EDatabase::prepare($s);

```

```
164         try {
165             $statement->execute(array(':nc' => $nickname,
166             ':ic' => $isConfirmed ));
167         }
168         catch (PDOException $e) {
169             echo 'Problème de mise à jour dans la base de
170 données: '.$e->getMessage();
171             return false;
172         }
173         // Ok
174         return true;
175     }
176
177     /**
178      * @brief Update le champ IsConfirmed d'un
179      utilisateur à 1
180      * @param nickname De type string, contient le
181      nickname de l'utilisateur à update
182      * @return [bool] Retourne true si l'update a réussi
183      */
184     static function ConfirmUser($nickname)
185     {
186         return ModelUsers::UpdateConfirmation($nickname,
187         1);
188     }
189 }
```



```

1  <?php
2  /**
3   * @brief    Objet EPreferences
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new EPreferences();
8   *          $u->Updated = new DateTime('2011-01-01T15:03')
9   *          $u->BeginTime = new DateTime('2019-01-01T15:00
10  *          $u->EndTime = new DateTime('2019-01-01T16:00')
11  *          $u->NbReservation = 2;
12  *
13  */
14  class EPreference {
15      /**
16       * @brief    Le Constructor appelé au moment de la
17       *          création de l'objet. Ie. new EPreferences();
18       * @param InUpdated          L'Id du rôle
19       * @param InBeginTime        Le nom du rôle. (Optionel)
20       *          Defaut ""
21       *          ***** A FAIRE *****
22       */
23      public function __construct($InBeginTime=null,
24      $InEndTime=null, $InNbReservation=null)
25      {
26          $this->BeginTime = $InBeginTime;
27          $this->EndTime = $InEndTime;
28          $this->NbReservation = $InNbReservation;
29      }
30      /**
31       * @var DateTime Date et heure de la dernière update
32       */
33      public $Updated;
34      /**
35       * @var DateTime L'heure d'ouverture des courts
36       */
37      public $BeginTime;
38

```

```
39     /**
40      * @var DateTime l'heure de fermeture des courts
41      */
42     public $EndTime;
43
44     /**
45      * @var int Le nombre de réservation que l'utilisateur
46      peut faire par jour
47      */
48     public $NbReservation;
49 }
50
```

```

1  <?php
2  // Projet: TPI
3  // Script: Modèle users.php
4  // Description: Classe de gestions de la table users
5  // Auteur: Loïc Burnand
6  // Version 1.0 PC 13.5.2019 / Codage initial
7
8  class ModelCourts
9  {
10     /**
11      * @brief Retourne le tableau des utilisateurs de type
12      ECourt
13      * @return [array of ECourt] Le tableau de ECourt
14      */
15     static function GetAllCourts()
16     {
17         // On crée un tableau qui va contenir les objets
18         ECourt
19         $arr = array();
20
21         $s = "SELECT `IdCourt`, `Name`, `Desc`, `Deleted`
22 FROM `tpi`.`COURTS`";
23         $statement = EDatabase::prepare($s,array(PDO::
24 ATTR_CURSOR => PDO::CURSOR_FWDONLY));
25         try {
26             $statement->execute();
27         }
28         catch (PDOException $e) {
29             echo 'Problème de lecture de la base de
30 données: '.$e->getMessage();
31             return false;
32         }
33         // On parcourt les enregistrements
34         while ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO
35 ::FETCH_ORI_NEXT)){
36             // On crée l'objet ECourt en l'initialisant
37             avec les données provenant
38             // de la base de données
39             $u = new ECourt($row['IdCourt'], $row['Name'],
40 $row['Desc'], $row['Deleted']);
41             // On place l'objet ECourt créé dans le
42             tableau
43             array_push($arr, $u);
44         }
45         // On retourne le tableau contenant les courts

```

```

36 sous forme ECourt
37     return $arr;
38 }
39
40 /**
41  * @brief Cherche un court par son Id dans la table
42  * @param Id      L'id a recherché dans la base de
donnée
43  * @return [ECourt] Retourne le premier Court du
résultat de la requête
44  */
45 static function GetCourtById($Id)
46 {
47     $s = "SELECT `IdCourt`, `Name`, `Desc`, `Deleted`
FROM `tpi`.`COURTS` WHERE `IdCourt` = :e";
48
49     $statement = EDatabase::prepare($s,array(PDO::
ATTR_CURSOR => PDO::CURSOR_FWDONLY));
50     try {
51         $statement->execute(array(':e' => $Id));
52     }
53     catch (PDOException $e) {
54         echo 'Problème de lecture de la base de
données: '.$e->getMessage();
55         return false;
56     }
57     // On parcourt les enregistrements
58     if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
FETCH_ORI_NEXT)){
59         // On crée l'objet ECourt en l'initialisant
avec les données provenant
60         // de la base de données et on le retourne
61         return new ECourt($row['IdCourt'], $row['Name'
], $row['Desc'], $row['Deleted']);
62     }
63     // On retourne null si il n'y a pas de court
64     return null;
65 }
66
67 /**
68  * @brief Ajoute un court dans la base de donnée
69  * @param Court le court a ajouté sous forme de ECourt
70  * @return [bool] Retourne true si l'ajout est réussi,
sinon retourne false
71  */

```

```

72     static function AddCourt($Court)
73     {
74         $s = "INSERT INTO `tpi`.`courts`(`Name`, `Desc`
75         `, `Deleted`) VALUES (:na, :dc, :de);";
76         $statement = EDatabase::prepare($s);
77         try {
78             $statement->execute(array(':na' => $Court->
79 Name, ':dc' => $Court->Desc, ':de' => (int)$Court->
80 Deleted));
81         }
82         catch (PDOException $e) {
83             echo 'Problème d\'insertion dans la base de
84 données: '. $e->getMessage();
85             return false;
86         }
87         return true;
88     }
89     /**
90      * @brief Permet de mettre à jours les donnée d'un
91      court
92      * @param Court      le court que l'on souhaite
93      modifié
94      * @return [bool]    Retourne true si l'update est
95      réussi, sinon retourne false
96      */
97     static function UpdateCourt($Court)
98     {
99         $s = "UPDATE `TPI`.`COURTS` SET `Name` = :na, `
100 Desc` = :dc, `Deleted` = :de WHERE `IdCourt` = :id";
101         $statement = EDatabase::prepare($s);
102         try {
103             $statement->execute(array(':id' => $Court->Id
104 , ':na' => $Court->Name, ':dc' => $Court->Desc, ':de' =>
105 (int)$Court->Deleted));
106         }
107         catch (PDOException $e) {
108             echo 'Problème de mise à jour dans la base de
109 données: '. $e->getMessage();
110             return false;
111         }
112         // Ok
113         return true;
114     }
115

```

```

106      /**
107       * @brief Supprime un Court
108       * @param Court le court que l'on souhaite
supprimer
109       * @return [bool] Retourne true si la suppression
est réussie, sinon retourne false
110       */
111       static function DeleteCourt($Court)
112       {
113           if(ModelCourts::IsCourtReference($Court))
114               return ModelCourts::MarkCourtDeleted(
$Court);
115           else
116           {
117               $s = "DELETE FROM `TPI`.`COURTS`
WHERE `IdCourt` = :id";
118               $statement = EDatabase::prepare($s);
119               try {
120                   $statement->execute(array(':id'
=> $Court->Id));
121               }
122               catch (PDOException $e) {
123                   echo 'Problème de mise à jour
dans la base de données: '.$e->getMessage();
124                   return false;
125               }
126               // Ok
127               return true;
128           }
129       }
130
131      /**
132       * @brief Retourne true si le court est lié à une
ou plusieurs reservation sinon false
133       * @param court de type ECourt, le court à
recherché
134       */
135       static function IsCourtReference($court)
136       {
137           $s = "SELECT COUNT(*) FROM `tpi`.`
RESERVATIONS` WHERE `IdCourt` = :e";
138
139           $statement = EDatabase::prepare($s, array(PDO
::ATTR_CURSOR => PDO::CURSOR_FWDONLY));
140           try {

```

```

141         $statement->execute(array(':e' => $court
->Id));
142     }
143     catch (PDOException $e) {
144         echo 'Problème de lecture de la base de
données: '.$e->getMessage();
145         return false;
146     }
147     $result = $statement->fetch();
148     return ($result['COUNT(*)'] > 0) ? true :
false;
149 }
150
151     static function MarkCourtDeleted($court)
152     {
153         $s = "UPDATE `tpi`.`courts` SET `Deleted` = 1
WHERE `IdCourt` = :id";
154         $statement = EDatabase::prepare($s);
155         try {
156             $statement->execute(array(':id' => $court
->Id));
157         }
158         catch (PDOException $e) {
159             echo 'Problème de mise à jour dans la
base de données: '.$e->getMessage();
160             return false;
161         }
162         // Ok
163         return true;
164     }
165
166
167 }

```



```

1  <?php
2
3  // Projet: TPI
4  // Script: Modèle ModelTokens.php
5  // Description: Classe de gestions de la table Tokens
6  // Auteur: Loïc Burnand
7  // Version 1.0 PC 13.5.2019 / Codage initial
8
9
10 class ModelTokens
11 {
12     /**
13      * @brief Cherche un Token par son nickname dans la
14      table
15      * @param nickname Le nickname a recherché dans la
16      base de donnée
17      * @return [EToken] Retourne le premier token du
18      résultat de la requête
19      */
20     static function GetTokenByNickname($nickname)
21     {
22         $s = "SELECT `Nickname`, `ValidateTill`, `Code`
23 FROM `tpi`.`TOKENS` WHERE `Nickname` = :e";
24
25         $statement = EDatabase::prepare($s,array(PDO::
26 ATTR_CURSOR => PDO::CURSOR_FWDONLY));
27
28         try {
29             $statement->execute(array(':e' => $nickname));
30         }
31         catch (PDOException $e) {
32             echo 'Problème de lecture de la base de
33 données: '.$e->getMessage();
34             return false;
35         }
36         // On parcourt les enregistrements
37         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
38 FETCH_ORI_NEXT)){
39             // On crée l'objet EToken en l'initialisant
40             avec les données provenant
41             // de la base de données et on le retourne
42             return new EToken($row['Nickname'], $row['
43 ValidateTill'], $row['Code']);
44         }
45         // On retourne null si il n'y a pas d'utilisateur
46         return null;

```

```

37     }
38
39     /**
40      * @brief Cherche un Token par son code dans la table
41      * @param code Le code a recherché dans la base de
donnée
42      * @return [EToken] Retourne le premier token du
résultat de la requête
43      */
44     static function GetTokenByCode($code)
45     {
46         $s = "SELECT `Nickname`, `ValidateTill`, `Code`
FROM `tpi`.`TOKENS` WHERE `Code` = :e";
47
48         $statement = EDatabase::prepare($s,array(PDO::
ATTR_CURSOR => PDO::CURSOR_FWDONLY));
49         try {
50             $statement->execute(array(':e' => $code));
51         }
52         catch (PDOException $e) {
53             echo 'Problème de lecture de la base de
données: '.$e->getMessage();
54             return false;
55         }
56         // On parcourt les enregistrements
57         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
FETCH_ORI_NEXT)){
58             // On crée l'objet EUser en l'initialisant
avec les données provenant
59             // de la base de données et on le retourne
60             return new EToken($row['Nickname'], $row['
ValidateTill'], $row['Code']);
61         }
62         // On retourne null si il n'y a pas d'utilisateur
63         return null;
64     }
65
66
67     /**
68      * @brief Créer un token et appelle une fonction pour
l'insérer en bd
69      * @param nickname Le nickname de l'utilisateur
possédant le token
70      * @return [bool] Retourne true si la création est
réussie, sinon retourne false

```

```

71     */
72     static function CreateToken($nickname)
73     {
74         $token = new EToken($nickname);
75         $token->Code = ModelTokens::generateToken();
76         return ModelTokens::InsertToken($token);
77     }
78
79     /**
80      * @brief Ajoute un token dans la base de donnée
81      * @param Token le token a ajouté sous forme de
EToken
82      * @return [bool] Retourne true si l'ajout est réussi
, sinon retourne false
83     */
84     static function InsertToken($Token)
85     {
86         $s = "INSERT INTO `tpi`.`tokens`(`Nickname`, `
ValidateTill`, `Code`) VALUES (:n, NOW() + INTERVAL 1 DAY
, :c);";
87         $statement = EDatabase::prepare($s);
88         try {
89             $statement->execute(array(':n' => $Token->
Nickname, ':c' => $Token->Code));
90         }
91         catch (PDOException $e) {
92             echo 'Problème d\'insertion dans la base de
données: '. $e->getMessage();
93             return false;
94         }
95         return true;
96     }
97
98     /**
99      * Generate a random token
100     * @param number $len The token length. Default is
20.
101     * @return string The generated token string.
102     */
103     static function generateToken($len = 20) {
104         $keys = array_merge(range(0,9), range('a', 'z'),
range('A', 'Z'));
105         $c = count($keys);
106         $token = '';
107         for($i=0; $i < $len; $i++) {

```

```

108         $token .= $keys[mt_rand(0, $c - 1)];
109     }
110     return $token;
111 }
112
113
114 /**
115  * @brief Accepte un utilisateur grâce à son token
116  * @param token Le token a validé
117  * @return bool Retourne true si l'acceptation est
118  réussie, sinon retourne false
119  */
119     static function acceptToken($code)
120     {
121         if(ModelTokens::isTokenValable($code))
122         {
123             ModelUsers::ConfirmUser(ModelTokens::
124 GetTokenByCode($code)->Nickname);
125             return true;
126         }
127         return false;
128     }
129
130 /**
131  * @brief Vérifie si le token est toujours valable
132  * @param Code le code du token a vérifié
133  * @return bool Retourne true si le token est valable
134  sinon false
135  */
136     static function isTokenValable($code)
137     {
138         $s = "SELECT `Nickname`, `ValidateTill`, `Code`
139 FROM `tpi`.`TOKENS` WHERE `Code` = :e AND DATEDIFF(`
140 ValidateTill`, now()) >= 0";
141
142         $statement = EDatabase::prepare($s,array(PDO::
143 ATTR_CURSOR => PDO::CURSOR_FWDONLY));
144         try {
145             $statement->execute(array(':e' => $code));
146         }
147         catch (PDOException $e) {
148             echo 'Problème de lecture de la base de
149 données: '.$e->getMessage();
150             return false;
151         }
152     }

```

```
146         // On parcourt les enregistrements
147         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
FETCH_ORI_NEXT)) {
148             //
149             return true;
150         }
151         // On retourne null si il n'y a pas d'utilisateur
152         return false;
153     }
154 }
```



```

1  <?php
2  /**
3   * @brief    Objet Reservation
4   * @remark  Cet objet est utilisé comme conteneur
5   *
6   *          Exemple d'utilisation 1
7   *          $u = new EResevation();
8   *          $u->Court = new ECourt();
9   *          $u->User = new EUser();
10  *          $u->IsConfirmed = false;
11  *          $u->Date = "";
12  *
13  *          Exemple d'utilisation 2
14  *          $u = new EResevation(new ECourt(), new EUser()
15  , false, "");
16  */
17
18  class EReservation {
19      /**
20       * @brief    Le Constructor appelé au moment de la
21       création de l'objet. Ie. new EResevation();
22       * @param InCourt      Le terrain réservé. (Optionel)
23       Default ""
24       * @param InUser      L'utilisateur qui a réservé. (
25       Optionel) Default ""
26       * @param InConfirmed  La confirmation de la
27       reservation. (Optionel) Default false
28       * @param InDate      La date et l'heure de la
29       reservation. (Optionel) Default ""
30       */
31
32      public function __construct($InCourt = "", $InUser = "
33  ", $InDate = "")
34      {
35          $this->Court = $InCourt;
36          $this->User = $InUser;
37          $this->Date = $InDate;
38      }
39
40      /**
41       * @var ECourt Le terrain qui a été réservé
42       */
43
44      public $Court;
45
46      /**
47       * @var EUser L'utilisateur qui a réservé
48       */
49
50      public $User;

```

```
39
40     /**
41      * @var DateTime La date et l'heure de la reservation
42      */
43     public $Date;
44 }
45
46
47
48 ?>
```



```

1  <?php
2  // Projet: TPI
3  // Script: Modèle ModelEmailSender.php
4  // Description: Classe de gestions des envoient d'email
5  // Auteur: Loïc Burnand
6  // Version 1.0 PC 13.5.2019 / Codage initial
7
8  class ModelEmailSender
9  {
10     /**
11      * @brief Permet d'envoyer un email grâce à l'api
Swiftmailer
12      * @param Email Un objet EEmail contenant les données
pour envoyer un email
13      */
14     static function SendEmail($Email)
15     {
16         // On doit créer une instance de transport smtp
avec les constantes
17         // définies dans le fichier mailparam.php
18         $transport = Swift_SmtpTransport::newInstance(
EMAIL_SERVER, EMAIL_PORT, EMAIL_TRANS)
19         ->setUsername(EMAIL_USERNAME)
20         ->setPassword(EMAIL_PASSWORD);
21
22         try {
23             // On crée un nouvelle instance de mail en
utilisant le transport créé précédemment
24             $mailer = Swift_Mailer::newInstance($transport
);
25             // On crée un nouveau message
26             $message = Swift_Message::newInstance();
27             // Le sujet du message
28             $message->setSubject($Email->Subject);
29             // Qui envoie le message
30             $message->setFrom(array(EMAIL_USERNAME =>
EMAIL_USERNAME));
31             // A qui on envoie le message
32             $message->setTo(array($Email->To));
33
34             // Un petit message html
35             // On peut bien évidemment avoir un message
texte
36             /*$body =
37             '<html>' .

```

```
38         ' <head></head>' .
39         ' <body>'.
40         ' <p>Un petit message envoyé avec Swift
Mailer 5.</p>' .
41         ' </body>' .
42         '</html>';*/
43         // On assigne le message et on dit de quel
type. Dans notre exemple c'est du html
44         $message->setBody($Email->Body, 'text/html');
45         // Maintenant il suffit d'envoyer le message
46         $result = $mailer->send($message);
47
48     } catch (Swift_TransportException $e) {
49         return "Problème d'envoi de message: ".$e->
getMessage();
50     }
51     return true;
52 }
53 }
```

```

1  <?php
2  // Projet: TPI
3  // Script: Modèle ModelPreferences.php
4  // Description: Classe de gestions de la table Preferences
5
6  // Auteur: Loïc Burnand
7  // Version 1.0 PC 13.5.2019 / Codage initial
8
9  class ModelPreferences
10 {
11     /**
12      * @brief Retourne un objet EPreferences contenant les
13      préférences
14      * @return [EPreferences] Les préférences dans un
15      objet
16      */
17     static function GetPreferences ()
18     {
19         // On créer la requête
20         $s = "SELECT `Updated`, `BeginTime`, `EndTime`, `
21 NbReservationsByUser` FROM `tpi`.`Preferences`";
22         $statement = EDatabase::prepare($s,array(PDO::
23 ATTR_CURSOR => PDO::CURSOR_FWDONLY));
24         try {
25             $statement->execute();
26         }
27         catch (PDOException $e) {
28             echo 'Problème de lecture de la base de
29 données: '.$e->getMessage();
30             return false;
31         }
32         // On parcourt les enregistrements
33         while ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO
34 ::FETCH_ORI_NEXT)) {
35             // On crée l'objet EPreferences en l'
36             initialisant avec les données provenant
37             // de la base de données et on le retourne
38             return new EPreference($row['BeginTime'], $row
39 ['EndTime'], $row['NbReservationsByUser']);
40         }
41         // On retourne null si aucun enregistrement n'a
42         été trouvé
43         return $null;
44     }
45 }

```

```

36     /**
37      * @brief Permet de mettre à jours les préférences
38      * @param Preferences Les nouvelles préférences dans
un objet EPreference
39      * @return [bool] Retourne true si l'update est réussi
, sinon retourne false
40      */
41      static function UpdatePreferences($Preferences)
42      {
43          $s = "UPDATE `tpi`.`PREFERENCES` SET `Updated` =
now(), `BeginTime` = :bt, `EndTime` = :et, `
NbReservationsByUser` = :ru WHERE `Updated` != now()";
44          $statement = EDatabase::prepare($s);
45          try {
46              $statement->execute(array(':bt' =>
$Preferences->BeginTime, ':et' => $Preferences->EndTime, '
:ru' => $Preferences->NbReservation));
47          }
48          catch (PDOException $e) {
49              echo 'Problème de mise à jour dans la base de
données: ' . $e->getMessage();
50              return false;
51          }
52          // Ok
53          return true;
54      }
55
56     /**
57      * @brief Retourne le débuts des horaires d'ouvertur
dans le format "H"
58      * @return [string] Retourne le début des horaires d'
ouverture
59      */
60      static function GetBeginTime()
61      {
62          $Preferences = ModelPreferences::GetPreferences();
63          $BeginTime = $Preferences->BeginTime;
64          $BeginTime = strtotime($BeginTime);
65          return date('H', $BeginTime);
66      }
67
68     /**
69      * @brief Retourne la fin des horaires d'ouverture
dans le format "H"
70      * @return [string] Retourne la fin des horaires d'

```

```
70  ouverture
71      */
72      static function GetEndTime()
73      {
74          $Preferences = ModelPreferences::GetPreferences()
75      ;
76          $EndTime = $Preferences->EndTime;
77          $EndTime = strtotime($EndTime);
78          return date('H', $EndTime);
79      }
```



```

1  <?php
2
3  // Projet: TPI
4  // Script: Modèle Reservation.php
5  // Description: Classe de gestions de la table Reservation
6
7  // Auteur: Loïc Burnand
8  // Version 1.0 PC 9.5.2019 / Codage initial
9
10 class ModelReservations
11 {
12     /**
13      * @brief Retourne le tableau des reservations de type
14      EReservation
15      * @return [array of EReservation] Le tableau de
16      EReservation
17      */
18     static function GetAllReservations()
19     {
20         // On crée un tableau qui va contenir les objets
21         EReservation
22         $arr = array();
23
24         $s = "SELECT `IdCourt`, `Nickname`, `Date` FROM `
25         tpi`.`RESERVATIONS`";
26         $statement = EDatabase::prepare($s,array(PDO::
27         ATTR_CURSOR => PDO::CURSOR_FWDONLY));
28         try {
29             $statement->execute();
30         }
31         catch (PDOException $e) {
32             echo 'Problème de lecture de la base de
33             données: '.$e->getMessage();
34             return false;
35         }
36         // On parcourt les enregistrements
37         while ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO
38         ::FETCH_ORI_NEXT)){
39             // On crée l'objet EReservation en l'
40             initialisant avec les données provenant
41             // de la base de données
42             $u = new EReservation(ModelCourts::
43             GetCourtById($row['IdCourt']), ModelUsers::
44             GetUserByNickname($row['Nickname'], $row['Date']);

```

```

35         // On place l'objet EReservation créé dans le
    tableau
36         array_push($arr, $u);
37     }
38     // On retourne le tableau contenant les
    utilisateurs sous forme EReservation
39     return $arr;
40 }
41
42
43 /**
44  * @brief Retourne un tableau de EReservation qui ont
    lieu à la date donnée
45  * @param date La date des reservations que l'on
    souhaite récupérer
46  * @return [array of EReservation] Le tableau de
    EReservation
47  */
48     static function GetReservationsForADate($date)
49     {
50         // On crée un tableau qui va contenir les objets
    EReservation
51         $arr = array();
52
53         $s = "SELECT `IdCourt`, `Nickname`, `Date` FROM `
    tpi`.`reservations` WHERE `Date` between :e and :en ";
54
55         $statement = EDatabase::prepare($s,array(PDO::
    ATTR_CURSOR => PDO::CURSOR_FWDONLY));
56         try {
57             $statement->execute(array(':e' => $date, ':en'
    => $date . ' 23:59:59'));
58         }
59         catch (PDOException $e) {
60             echo 'Problème de lecture de la base de
    données: ' . $e->getMessage();
61             return false;
62         }
63         // On parcourt les enregistrements
64         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
    FETCH_ORI_NEXT)) {
65             // On crée l'objet EReservation en l'
    initialisant avec les données provenant
66             // de la base de données
67             $u = new EReservation(ModelCourts::

```



```

67 GetCourtById($row['IdCourt']), ModelUsers::
    GetUserByNickname($row['Nickname'], $row['Date']);
68     // On place l'objet EReservation créé dans le
    tableau
69     array_push($arr, $u);
70 }
71     // On retourne le tableau contenant les
    utilisateurs sous forme EReservation
72     return $arr;
73 }
74
75 /**
76  * @brief Cherche les réservations pour un
    utilisateur
77  * @param User L'utilisateur (le nickname est
    obligatoire) a recherché dans la base de donnée
78  * @return [EReservation] Retourne les réservations
    du résultat de la requête
79  */
80 static function GetReservationsByUser($User)
81 {
82     // On crée un tableau qui va contenir les objets
    EReservation
83     $arr = array();
84
85     $s = "SELECT `IdCourt`, `Nickname`, `Date` FROM `
    tpi`.`RESERVATIONS` WHERE `Nickname` = :e";
86
87     $statement = EDatabase::prepare($s,array(PDO::
    ATTR_CURSOR => PDO::CURSOR_FWDONLY));
88     try {
89         $statement->execute(array(':e' => $User->
    Nickname));
90     }
91     catch (PDOException $e) {
92         echo 'Problème de lecture de la base de
    données: '.$e->getMessage();
93         return false;
94     }
95     // On parcourt les enregistrements
96     while ($row=$statement->fetch(PDO::FETCH_ASSOC,
    PDO::FETCH_ORI_NEXT)) {
97         // On crée l'objet EReservation en l'
    initialisant avec les données provenant
98         // de la base de données

```

```

99         $u = new EReservation(ModelCourts::
GetCourtById($row['IdCourt']), ModelUsers::
GetUserByNickname($row['Nickname']), $row['Date']);
100         // On place l'objet EReservation créé dans le
tableau
101         array_push($arr, $u);
102     }
103     // On retourne le tableau contenant les
utilisateurs sous forme EReservation
104     return $arr;
105 }
106
107 /**
108  * @brief Cherche les réservations pour un court
109  * @param Court Le court (l'id est obligatoire) a
recherché dans la base de donnée
110  * @return [EReservation] Retourne les réservations
du résultat de la requête
111  */
112     static function GetReservationsByCourt($Court)
113     {
114         // On crée un tableau qui va contenir les objets
EReservation
115         $arr = array();
116
117         $s = "SELECT `IdCourt`, `Nickname`, `Date` FROM `
tpi`.`RESERVATIONS` WHERE `IdCourt` = :e";
118
119         $statement = EDatabase::prepare($s,array(PDO::
ATTR_CURSOR => PDO::CURSOR_FWDONLY));
120         try {
121             $statement->execute(array(':e' => $Court->Id)
);
122         }
123         catch (PDOException $e) {
124             echo 'Problème de lecture de la base de
données: '.$e->getMessage();
125             return false;
126         }
127         // On parcourt les enregistrements
128         if ($row=$statement->fetch(PDO::FETCH_ASSOC,PDO::
FETCH_ORI_NEXT)){
129             // On crée l'objet EReservation en l'
initialisant avec les données provenant
130             // de la base de données

```

```

131         $u = new EReservation(ModelCourts::
GetCourtById($row['IdCourt']), ModelUsers::
GetUserByNickname($row['Nickname']), $row['Date']);
132         // On place l'objet EReservation créé dans le
tableau
133         array_push($arr, $u);
134     }
135     // On retourne le tableau contenant les
utilisateurs sous forme EReservation
136     return $arr;
137 }
138
139
140 /**
141  * @brief Supprime une reservation
142  * @param Reservation (EReservation) La reservation
a supprimer
143  * @return [bool] Retourne true si la suppression est
réussie, sinon retourne false
144  */
145 static function DeleteReservation($Reservation)
146 {
147     $s = "DELETE FROM `TPI`.`RESERVATIONS` WHERE `
IdCourt` = :ic AND `Nickname` = :nc AND `Date` = :d";
148     $statement = EDatabase::prepare($s);
149     try {
150         $statement->execute(array(':ic' =>
$Reservation->Court->Id, ':nc' => $Reservation->User->
Nickname, ':d' => $Reservation->Date));
151     }
152     catch (PDOException $e) {
153         echo 'Problème de mise à jour dans la base de
données: ' . $e->getMessage();
154         return false;
155     }
156     // Ok
157     return true;
158 }
159
160 /**
161  * @brief Ajoute une réservation dans la base de
donnée
162  * @param Reservation la réservation a ajouté sous
forme de EReservation
163  * @return [bool] Retourne true si l'ajout est réussi

```

```
163 , sinon retourne false
164 */
165 static function AddReservation($Reservation)
166 {
167     $s = "INSERT INTO `tpi`.`reservations` (`IdCourt
168     `, `Nickname`, `Date`) VALUES (:ic, :nc, :d);";
169     $statement = EDatabase::prepare($s);
170     try {
171         $statement->execute(array(':ic' =>
172         $Reservation->Court->Id, ':nc' => $Reservation->User->
173         Nickname, ':d' => $Reservation->Date));
174     }
175     catch (PDOException $e) {
176         echo 'Problème d\'insertion dans la base de
177         données: '. $e->getMessage();
178         return false;
179     }
180     return true;
181 }
```

```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
;
10 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
.php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
php';
13
14 $u = new EUser("loic@burnand.com", "Test3", "BURNAND", "
Loïc", "0767756644", new ERole(1), 1);
15 $u->Password = "f6889fc97e14b42dec11a8c183ea791c5465b658";
16 ModelUsers::AddUser($u);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
15
16 $u = new ECourt();
17 $u->Name = "Court 2";
18 $u->Desc = "Le deuxième court";
19 $u->Deleted = false;
20 ModelCourts::AddCourt($u);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
12 .php';
13
14 $u = new EUser("loic@burnand.com", "Test3", "Burnand2", "
  Loic", "0767756644", new ERole(1), 1);
15 $u->Password = "f6889fc97e14b42dec11a8c183ea791c5465b658";
16 ModelUsers::updateUser($u);
```



```
1 <?php
2
3 /**
4  * @remark Mettre le bon chemin d'accès à votre fichier
   contenant les constantes
5  */
6 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
   ;
10 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
   EReservation.php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
   ;
12 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
   .php';
13
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
   php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
   php';
16 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
   .php';
17 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelReservations.php';
18 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelPreferences.php';
19 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelTokens
   .php';
20
21 ModelTokens::CreateToken("Test3");
```



```
1 <?php
2
3 /**
4  * @remark Mettre le bon chemin d'accès à votre fichier
   contenant les constantes
5  */
6 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
   ;
10 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
   EReservation.php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
   ;
12 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
   .php';
13
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
   php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
   php';
16 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
   .php';
17 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelReservations.php';
18 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelPreferences.php';
19 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelTokens
   .php';
20
21 $c = new ECourt();
22 $c->Id = 2;
23 ModelCourts::DeleteCourt($c);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
  ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
  ;
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
10
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13
14 $users = ModelUsers::GetAllUsers();
15
16 print_r($users);
```



```
1 <?php
2 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
3 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
6 ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
9
10 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
12
13 $role = ModelUsers::GetUserRole(new EUser("loic@burnand.
  com", "Test3", "Burnand2", "Loic", "0767756644", new ERole
  (1), 1));
14 echo "Si le code est déjà rempli, requête retourne : " .
  $role->Label;
15
16 $role = ModelUsers::GetUserRole(new EUser("loic@burnand.
  com", "Test3", "Burnand2", "Loic", "0767756644"));
17 echo "Si le code n'est pas rempli mais le nickname est
  rempli : " . $role->Label;
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
16
17 $u = new ECourt("2", "Court 2", "Le deuxième court modifié", false);
18 ModelCourts::UpdateCourt($u);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
15
16 $courts = ModelCourts::GetAllCourts();
17
18 print_r($courts);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
15
16 print_r(ModelCourts::GetCourtById(1));
```



```
1 <?php
2 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
3 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
6 ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
9
10 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
12
13 echo ModelRoles::GetRoleByCode("1")->Label;
14
15
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
15
16 $reservation = new EReservation();
17 $reservation->Court = new ECourt(1);
18 $reservation->User = new EUser();
19 $reservation->User->Nickname = "Test2";
20 $reservation->Date = "2010-04-02 16:00:00";
21
22 ModelReservations::AddReservation($reservation);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelPreferences.php';
16
17 print_r(ModelPreferences::GetPreferences());
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
.php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
.php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
.php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
.php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
ModelReservations.php';
16
17 $reservation = new EReservation();
18 $reservation->Court = new ECourt(2);
19 $reservation->User = new EUser();
20 $reservation->User->Nickname = "Test";
21 $reservation->Date = "2010-04-02 16:00:00";
22
23 ModelReservations::DeleteReservation($reservation);
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
  ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
  ;
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
10
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13
14 ModelUsers::getUserByNickname("Test");
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelPreferences.php';
16
17 $r = new EPreference("2000-00-00 08:00:00", "2000-00-00 17
  :00:00", 4);
18
19 ModelPreferences::UpdatePreferences($r);
20
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
12 .php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
14 php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
16 php';
17 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
18 .php';
19 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
20 ModelReservations.php';
21
22 print_r(ModelReservations::GetAllReservations());
23
```



```
1 <?php
2 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
3 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
6 ;
7 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
9
10 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
11 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelPreferences.php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelTokens
  .php';
16
17 print_r(ModelTokens::GetTokenByNickname("Test"));
18
19
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
12 .php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
14 php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
16 php';
17 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
18 .php';
19 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
20
21 $u = new EUser();
22 $u->Nickname = "Test";
23
24 print_r(ModelReservations::GetReservationsByUser($u));
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
12 .php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
14 php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
16 php';
17 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
18 .php';
19 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
20
21 $u = new ECourt();
22 $u->Id = 1;
23
24 print_r(ModelReservations::GetReservationsByCourt($u));
```



```
1 <?php
2
3 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
4 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
5 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
6 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
7 ;
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
  EReservation.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
10 ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
  .php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
  php';
13 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
  php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
  .php';
15 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
  ModelReservations.php';
16 print_r(ModelReservations::GetReservationsForADate("2010/
  04/02"));
```



```
1 <?php
2 /**
3  * @remark Remplir correctement les constantes ci-dessous
4  * en fonction de votre base de données
5  */
6 /**
7  * @brief Connection constants
8  */
9 define('EDB_DBTYPE', "mysql");
10 define('EDB_DBNAME', "tpi");
11 define('EDB_HOST', "127.0.0.1");
12 define('EDB_PORT', "3306");
13 define('EDB_USER', "root");
14 define('EDB_PASS', "Super");
15
16
17
```



```
1 <?php
2 /**
3  * @remark Remplir correctement les constantes ci-dessous
4  *          d'email. On utilise un compte google et le
5  *          serveur smtp de google.
6  *          Si vous utilisez un compte google, il faut
7  *          aller dans les paramètres du compte google
8  *          sous Security
9  *          il faut turn on "Less secure app access"
10 *          Sans quoi vous ne pourrez pas envoyer des
11 *          emails, vous recevrez le message
12 *          -----
13 *          Expected response code 250 but got code "535",
14 *          with message "535-5.7.8 Username and Password not
15 *          accepted.
16 *          Learn more at 535 5.7.8 https://support.google.com/mail/?p=BadCredentials w16sm74990200wrt.84 - gsmtpt "
17 *          -----
18 */
19
20 /**
21  * @brief Email constants
22  */
23
24 define('EMAIL_SERVER', "smtp.gmail.com");
25 define('EMAIL_PORT', 465);
26 define('EMAIL_TRANS', "ssl");
27 define('EMAIL_USERNAME', "tpiSquash@gmail.com");
28 define('EMAIL_PASSWORD', "Super2018");
29
```



```

1 <?php
2
3 /**
4  * @remark Mettre le bon chemin d'accès à votre fichier
   contenant les constantes
5  */
6 require_once $_SERVER['DOCUMENT_ROOT'].'/config/mailparam.
   php';
7 require_once $_SERVER['DOCUMENT_ROOT'].'/db/database.php';
8 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EUser.php';
9 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ERole.php';
10 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ECourt.php'
   ;
11 require_once $_SERVER['DOCUMENT_ROOT'].'/model/
   EReservation.php';
12 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EToken.php'
   ;
13 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EPreference
   .php';
14 require_once $_SERVER['DOCUMENT_ROOT'].'/model/EEmail.php'
   ;
15 require_once $_SERVER['DOCUMENT_ROOT'].'/model/ESession.
   php';
16
17 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelUsers.
   php';
18 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelRoles.
   php';
19 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelCourts
   .php';
20 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelReservations.php';
21 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelPreferences.php';
22 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/ModelTokens
   .php';
23 require_once $_SERVER['DOCUMENT_ROOT'].'/Model/
   ModelEmailSender.php';
24
25 // Inclure le fichier swift_required.php localisé dans le
   répertoire swiftmailer5
26 require_once $_SERVER['DOCUMENT_ROOT'].'/swiftmailer5/lib/
   swift_required.php';
27
28 class ControllerSquash

```

```

29 {
30     static function Login()
31     {
32         if(isset($_POST['login']) && isset($_POST['
password']))
33         {
34             $User = new EUser();
35             $User->Nickname = filter_input(INPUT_POST, '
login', FILTER_SANITIZE_SPECIAL_CHARS);
36             $User->Password = filter_input(INPUT_POST, '
password', FILTER_SANITIZE_SPECIAL_CHARS);
37             $User->Password = sha1($User->Password);
38             if(ModelUsers::CheckLogin($User))
39             {
40                 ESession::SetUser(ModelUsers::
getUserByNickname($User->Nickname));
41                 header("Location: index.php?action=Accueil
");
42             }
43         }
44         include $_SERVER['DOCUMENT_ROOT'].'/view/login.
html';
45     }
46
47     static function Register($user)
48     {
49         ModelUsers::AddUser($user);
50         $Email = new EEmail($user->Email, "Confirmation de
votre compte");
51         $Email->Body = '<html>' .
52             ' <head></head>' .
53             ' <body>' .
54             ' <p>Veuillez ouvrir ce lien pour
confirmer votre compte : http://localhost/index.php?action
=EmailConfirmation&token=' .
55             ModelTokens::GetTokenByNickname($user
->Nickname)->Code.
56             '</p>' .
57             ' </body>' .
58             '</html>';
59         ModelEmailSender::SendEmail($Email);
60         header("Location: index.php?action=
EmailRegisterSend");
61         exit();
62     }

```

```

63
64     static function EmailRegisterSend()
65     {
66         include $_SERVER['DOCUMENT_ROOT'].'/view/Email/
emailRegistrationSend.html';
67     }
68
69     static function EmailValidation($token)
70     {
71         if(ModelTokens::acceptToken($token))
72         {
73             include $_SERVER['DOCUMENT_ROOT'].'/view/
Email/emailValidate.html';
74         }
75         else
76         {
77             include $_SERVER['DOCUMENT_ROOT'].'/view/
Email/emailError.html';
78         }
79     }
80 }
81
82 static function Logout()
83 {
84     session_destroy();
85     header("Location: index.php");
86 }
87
88 static function Accueil($role=2)
89 {
90     $doc = $_SERVER['DOCUMENT_ROOT'].'/view/
DocUtilisateur/DocAccueil.html';
91     $courts = ModelCourts::GetAllCourts();
92     $users = ModelUsers::GetAllUsers();
93     $reservations = ModelReservations::
GetAllReservations();
94     $BeginTime = ModelPreferences::GetBeginTime();
95     $EndTime = ModelPreferences::GetEndTime();
96     $navBar = null;
97     if($role==2)
98         $navBar = ControllerSquash::requireToVar(
$_SERVER['DOCUMENT_ROOT'].'/view/User/NavBar.php');
99     else
100         $navBar = ControllerSquash::requireToVar(
$_SERVER['DOCUMENT_ROOT'].'/view/Admin/NavBar.php');

```

```

101         include $_SERVER['DOCUMENT_ROOT'].'/view/listAll
.php';
102     }
103
104     static function MyReservations($user)
105     {
106         $doc = '/view/DocUtilisateur/DocMyProfil.html';
107         $reservations = ModelReservations::
GetReservationsByUser($user);
108         include $_SERVER['DOCUMENT_ROOT'].'/view/User/
myReservations.php';
109     }
110
111     static function DeleteReservation($reservation, $user
)
112     {
113         if(!isset($user->Role))
114         {
115             $user = ModelUsers::getUserByNickname($user->
Nickname);
116         }
117         if($user->Role->Label == "Admin" || $reservation
->User->Nickname == $user->Nickname)
118             ModelReservations::DeleteReservation(
$reservation);
119         header("Location: ?action=Accueil");
120     }
121
122     static function ShowUsers()
123     {
124         $doc = "/view/DocUtilisateur/DocProfils.html";
125         $users = ModelUsers::GetAllUsers();
126         include $_SERVER['DOCUMENT_ROOT'].'/view/Admin/
listUsers.php';
127     }
128
129     static function ManagePreferences()
130     {
131         $doc = "/view/DocUtilisateur/DocParametre.html";
132         $courts = ModelCourts::GetAllCourts();
133         $preferences = ModelPreferences::GetPreferences()
;
134         $preferences->BeginTime = ModelPreferences::
GetBeginTime() . ":00";
135         $preferences->EndTime = ModelPreferences::

```

```

135 GetEndTime() . ":00";
136     include $_SERVER['DOCUMENT_ROOT'].'/view/Admin/
    ManagePreferences.php';
137 }
138
139     static function GetRole($user)
140     {
141         return ModelUsers::GetUserRole($user);
142     }
143
144     static function requireToVar($file){
145         ob_start();
146         require($file);
147         return ob_get_clean();
148     }
149
150     static function showUserProfil($nickname)
151     {
152         $doc = '/view/DocUtilisateur/DocProfilUser.html';
153         $user = ModelUsers::GetUserByNickname($nickname);
154         $reservations = ModelReservations::
GetReservationsByUser($user);
155         include $_SERVER['DOCUMENT_ROOT'].'/view/Admin/
    userProfil.php';
156     }
157
158     static function updatePreferences($preferences)
159     {
160         $preferences->BeginTime = "2000-00-00 " .
$preferences->BeginTime . ":00";
161         $preferences->EndTime = "2000-00-00 " .
$preferences->EndTime . ":00";
162         ModelPreferences::UpdatePreferences($preferences)
;
163     }
164 }

```