

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : index.php
9 */
10 header("location: ./vue/index.php");
10 exit();
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : index.php
9 */
10 session_start();
11 require_once("../Controleur/controleur.inc.php");
12 require_once("../Controleur/index.inc.php");
13 ?>
14 <!doctype html>
15 <html lang="fr">
16 <head>
17     <meta charset="utf-8">
18     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
19
20     <!-- Bootstrap CSS -->
21     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCBM0v3Xipma34MD+dH/1fQ784/j6cY/iJTQUohcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
22     <link rel="stylesheet" type="text/css" href="MyCss.css">
23     <title>Tales of the tavern</title>
24 </head>
25 <body>
26 <?php include_once("navbar.php");?>
27 <div class="container col-sm-12 col-md-6">
28     <div class="row">
29         <a class="col-3 offset-2 btn btn-primary" href="index.php" role="button">Trier par date</a>
30         <a class="col-3 offset-2 btn btn-primary" href="index.php?ordre=note" role="button">Trier
31             par moyenne</a>
32     </div>
33     <br/>
34 <?php
35 if (isset($_SESSION["utilisateur"])){
36     if(!empty($favoris) || $favoris = ""){
37         echo "<h1>Mes favoris</h1><div class=\"row\">";
38         afficherHistoires($favoris);
39         echo "</div>";
40     }
41     else{
42         echo "<h4>Vous n'avez pas de favoris</h4>";
43     }
44 }
45 }
46 ?>
47 <h1>Les histoires</h1>
48 <div class="row">
49     <?php afficherHistoires($histoires); ?>
50 </div>
51
52 <!--bootstrap-->
53 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
54 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
55 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEAff/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
56 </body>
57 </html>

```

```
1 .card-img-top {  
2     width: 100%;  
3     height: 15vw;  
4     object-fit: cover;  
5 }
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 05.04.19
7 * Version : 1.0
8 * Fichier : compte.php
9 */
10 session_start();
11 {
12     header("location: index.php");
13     exit();
14 }
15 $erreurMessage = "";
16 require_once("../Controleur/controleur.inc.php");
17 require_once("../Controleur/compte.inc.php");
18 ?>
19 <!doctype html>
20 <html lang="fr">
21 <head>
22
23     <meta charset="utf-8">
24     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
25
26     <!-- Bootstrap CSS -->
27
28     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
29     <link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.8.1/css/all.css" integrity="sha384-5oBUHEmvpQ+11W4y57PTFmhCaXp0ML5d60M1M7uH2+pqUivzIebhndOJK28anvf" crossorigin="anonymous">
30     <link rel="stylesheet" type="text/css" href="MyCss.css">
31     <title>Tales of the tavern</title>
32 </head>
33 <body>
34 <?php include_once("./navbar.php");?>
35 <div class="container col-sm-12 col-md-6 c border-1 mb-5">
36     <form action="#" method="post">
37         <div class="form-group">
38             <label>Nom*</label>
39             <input type="text" name="nom" class="form-control" value=<?= $nom ?>" required>
40         </div>
41         <div class="form-group">
42             <label>E-mail</label>
43             <input type="email" class="form-control" name="email" value=<?= $email ?>" required>
44         </div>
45         <div class="form-group">
46             <label>Ancien mot de passe</label>
47             <input type="password" class="form-control" name="motDePasse">
48         </div>
49         <div class="form-group">
50             <label>Nouveau Mot de passe</label>
51             <input type="password" class="form-control" name="nouveauMotDePasse">
52             <small>Le mot de passe doit contenir minimum 8 caractères, un chiffre (0 à 9) et une
53             lettre (a à Z)</small>
54         </div>
55         <div class="form-group">
56             <label>Confirmer Nouveau mot de passe</label>
57             <input type="password" class="form-control" name="confirmerNouveauMotDePasse">
58         </div>
59         <small>*Champs obligatoires</small>
60         <br/>
61         <label style="color: red"><?php if($erreurMessage !== true){echo $erreurMessage;} ?></label>
62         <br/>
63         <button type="submit" class="btn btn-primary">Mettre a jour les informations</button>
64     </form>
65 </div>
66 <div class="container col-12">
67     <h1>Mes histoires (moyenne <?= $moyenneAuteur ?>) <a class="btn btn-primary" title="Ajouter une
68     nouvelle histoire" href="modifierHistoire.php" role="button"><i class="fas fa-plus"></i></span></a></
69     h1>
70     <div >
71
72     </div>
73 </div>

```

```
71 <div class="row">
72     <?php
73         afficherHistoiresUilisateur();
74     ?>
75 </div>
76 <!--bootstrap-->
77 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
78 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
79 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgydOp3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
80 </body>
81 </html>
```

```

1 <!--
2 *   auteur : Raphael Lopes
3 *   Projet : Tales of the Tavern
4 *   description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 *   puissent les noter
6 *   date : 04.04.19
7 *   Version : 1.0
8 *   Fichier : navbar.php
8 -->
9 <nav class="navbar navbar-expand-lg navbar-light bg-light">
10    <a class="navbar-brand" href="index.php" >
11    </a>
11    <button class="navbar-toggler" type="button" data-toggle="collapse" data-target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-expanded="false" aria-label="Toggle navigation">
12      <span class="navbar-toggler-icon"></span>
13    </button>
14
15    <div class="collapse navbar-collapse" id="navbarSupportedContent">
16      <ul class="navbar-nav mr-auto">
17        <li class="nav-item active">
18          <a class="nav-link" href="index.php">Accueil<span class="sr-only">(current)</span></a>
19        </li>
20        <?php
21        if(isset($_SESSION["utilisateur"]))
22        {
23            ?>
24            <li class="nav-item active">
25              <a class="nav-link" href=".//compte.php">Mon compte<span class="sr-only">(current)</span></a>
26            </li>
27            <li class="nav-item active">
28              <a class="nav-link" href="deconnecter.php">Déconnecter<span class="sr-only">(current)</span></a>
29            </li>
30            <?php
31            } else {
32                ?>
33                <li class="nav-item active">
34                  <a class="nav-link" href="connexion.php">Se connecter<span class="sr-only">(current)</span></a>
35                </li>
36                <li class="nav-item active">
37                  <a class="nav-link" href="creerCompte.php">Créer un compte<span class="sr-only">(current)</span></a>
38                </li>
39                <?php
40            }
41            ?>
42        </ul>
43        <form class="form-inline my-2 my-lg-0" action="rechercher.php" method="get">
44          <input class="form-control mr-sm-2" type="search" placeholder="Rechercher" aria-label="Search" name="recherche">
45          <button class="btn btn-outline-success my-2 my-sm-0" type="submit">Rechercher</button>
46        </form>
47    </div>
48 </nav>

```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 11.04.19
7 * Version : 1.0
8 * Fichier : histoire.php
9 */
10 session_start();
11 {
12     header("location: index.php");
13     exit();
14 }
15 require_once("../Controleur/controleur.inc.php");
16 require_once("../Controleur/histoire.inc.php");
17 ?>
18 <!doctype html>
19 <html lang="fr">
20 <head>
21
22     <meta charset="utf-8">
23     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
24
25     <!-- Bootstrap CSS -->
26     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
27
28     <title>Tales of the tavern</title>
29 </head>
30 <body>
31 <?php include_once("./navbar.php");?>
32 <br/>
33 <div class="container col-sm-12 col-md-6 c border-1">
34     <div class="card col-md-12">
35         <?php
36
37             echo '';
38             echo '<h1 class="display-4">' . $titre . ' ('moyenne : ' . $moyenneGlobalHistoire .')</h1>';
39             echo '<p class="lead">' . $auteur . ' ('moyenne de l\'auteur : ' . $moyenneAuteur .')</p>';
40             echo '<p class="lead">' . $categorie . '</p>';
41             echo '<p>' . nl2br($texteHistoire) . '</p>';
42         ?>
43
44
45         <form action="#" method="post">
46             <div class="row">
47                 <div class="col-md-6 col-12">
48                     <div class="form-group">
49                         <label>Style (moyenne : <?= $moyenneStyle ?>)</label>
50                         <select class="form-control" name="noteStyle">
51                             <?php
52                             AfficherNotation();
53                             ?>
54                         </select>
55                     </div>
56                 </div>
57                 <div class="col-md-6 col-12">
58                     <div class="form-group">
59                         <label>Histoire (moyenne : <?= $moyenneHistoire ?>)</label>
60                         <select class="form-control" name="noteHistoire">
61                             <?php
62                             AfficherNotation();
63                             ?>
64                         </select>
65                     </div>
66                 </div>
67                 <div class="col-md-6 col-12">
68                     <div class="form-group">
69                         <label>Orthographe (moyenne : <?= $moyenneOrthographe ?>)</label>
70                         <select class="form-control" name="noteOrthographe">
71                             <?php
72                             AfficherNotation();
73                             ?>
74                         </select>

```

```
75          </div>
76      </div>
77      <div class="col-md-6 col-12">
78          <div class="form-group">
79              <label>Originalité (moyenne : <?= $moyenneOriginalite ?>)</label>
80              <select class="form-control" name="noteOriginalite">
81                  <?php
82                      AfficherNotation();
83                  ?>
84              </select>
85          </div>
86      </div>
87      <button type="submit" class="btn btn-primary col-12">Voter</button>
88      <?php
89          if(isset($_SESSION["utilisateur"]))
90              echo '</br></br><button type="submit" class="btn btn-primary col-12" name="favoris">
91 . $buttonText . '</button>'>
92      ?>
93
94      </form>
95  </div>
96
97  <!--bootstrap-->
98  <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz0rT7abK4lJStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo" crossorigin="anonymous"></script>
99  <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdsJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
100  <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFF/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
101 </body>
102 </html>
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : connexion.php
9 */
10 session_start();
11 {
12     header("Location: index.php");
13     exit();
14 }
15 require_once("../Controleur/controleur.inc.php");
16 require_once("../Controleur/connexion.inc.php");
17 ?>
18 <!doctype html>
19 <html lang="fr">
20 <head>
21
22     <meta charset="utf-8">
23     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
24
25     <!-- Bootstrap CSS -->
26     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
27
28     <title>Tales of the tavern</title>
29 </head>
30 <body>
31 <?php include_once("./navbar.php");?>
32 <br/>
33 <div class="container col-sm-12 col-md-6 c border-1">
34     <form action="#" method="post">
35         <form action="#" method="post">
36             <div class="form-group">
37                 <label>E-mail</label>
38                 <input type="email" class="form-control" name="email" value="=$email?" required>
39             </div>
40             <div class="form-group">
41                 <label>Mot de passe</label>
42                 <input type="password" class="form-control" name="motDePasse" required>
43             </div>
44             <label style="color: red">?= $erreurMessage ?</label>
45             <br/>
46             <button type="submit" class="btn btn-primary">Connexion</button>
47         </form>
48     </div>
49
50 <!--bootstrap-->
51 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965Dz0rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
52 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
53 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFF/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
54 </body>
55 </html>

```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 05.04.19
7 * Version : 1.0
8 * Fichier : supprimer.php
9 */
10 session_start();
11 {
12     header("location: index.php");
13     exit();
14 }
15
16 require_once("../Controleur/controleur.inc.php");
17 require_once("../Controleur/supprimer.inc.php");
18 ?>
19 <!doctype html>
20 <html lang="fr">
21 <head>
22
23     <meta charset="utf-8">
24     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
25
26     <!-- Bootstrap CSS -->
27     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
28
29     <title>Tales of the tavern</title>
30 </head>
31 <body>
32 <?php include_once("./navbar.php");?>
33 <div class="container col-sm-12 col-md-6 c border-1 mb-5">
34     <h1>
35         Êtes-vous sûr de vouloir supprimer cette histoire ?
36     </h1>
37     <form action="#" method="post">
38         <div class="row">
39             <button type="submit" class="btn btn-danger col-12 col-md-6 btn-lg" name="supprimer">Oui</button>
40             <a class="col-6 col-md-6 btn btn-primary btn-lg" href="compte.php" role="button">Non</a>
41         </div>
42     </form>
43 </div>
44
45 <!--bootstrap-->
46 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo" crossorigin="anonymous"></script>
47 <script src="https://cdn.jsdelivr.net/npm/popper.js@1.14.7/dist/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
48 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFF/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
49 </body>
50 </html>
51

```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 12.04.19
7 * Version : 1.0
8 * Fichier : rechercher.php
9 */
10 session_start();
11 if(!isset($_GET["recherche"])){
12     header("location: index.php");
13     exit();
14 }
15 require_once("../Controleur/controleur.inc.php");
16 require_once("../Controleur/rechercher.inc.php");
17 ?>
18 <!doctype html>
19 <html lang="fr">
20 <head>
21     <meta charset="utf-8">
22     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
23
24     <!-- Bootstrap CSS -->
25     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUhcWr7x9JvoRxT2M Zw1T" crossorigin="anonymous">
26     <link rel="stylesheet" type="text/css" href="MyCss.css">
27     <title>Tales of the tavern</title>
28 </head>
29 <body>
30 <?php include_once("navbar.php");?>
31 <h1>Histoire par titre</h1>
32 <?php
33 if(!empty($histoiresParTitre) || $histoiresParTitre = ""){
34     echo "<div class=\"row\">";
35     afficherHistoires($histoiresParTitre);
36     echo "</div>";
37 }
38 else{
39     echo "<h4>Il n' y pas d'histoire avec ce titre</h4>";
40 }
41 ?>
42 <h1>Histoire par nom</h1>
43 <?php
44 if(!empty($histoiresParAuteur) || $histoiresParAuteur = ""){
45     echo "<div class=\"row\">";
46     afficherHistoires($histoiresParAuteur);
47     echo "</div>";
48 }
49 else{
50     echo "<h4>Il n'y a pas d'utilisateur avec ce nom</h4>";
51 }
52 ?>
53
54 <!--bootstrap-->
55 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+965DzO0rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abtTE1Pi6jizo" crossorigin="anonymous"></script>
56 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-UO2eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></script>
57 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JSmVgydOp3pXB1rRib2UAYoIIy6OrQ6VrjIEaFF/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
58 </body>
59 </html>
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : creerCompte.php
9 */
10 session_start();
11 {
12     header("Location: index.php");
13     exit();
14 }
15 require_once("../Controleur/controleur.inc.php");
16 require_once("../Controleur/creerCompte.inc.php");
17 ?>
18 <!doctype html>
19 <html lang="fr">
20 <head>
21
22     <meta charset="utf-8">
23     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
24
25     <!-- Bootstrap CSS -->
26     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
27
28     <title>Tales of the tavern</title>
29 </head>
30 <body>
31 <?php include_once("./navbar.php");?>
32 <br/>
33 <div class="container col-sm-12 col-md-6 c border-1">
34     <form action="#" method="post">
35         <div class="form-group">
36             <label>Nom*</label>
37             <input type="text" name="nom" class="form-control" value="<?= $nom ?>" required>
38         </div>
39         <div class="form-group">
40             <label>E-mail*</label>
41             <input type="email" class="form-control" name="email" value="<?= $email ?>" required>
42         </div>
43         <div class="form-group">
44             <label>Mot de passe*</label>
45             <input type="password" class="form-control" name="motDePasse" required>
46             <small>Le mot de passe doit contenir minimum 8 caractères, un chiffre (0 à 9) et une
lettre (a à Z)</small>
47         </div>
48         <div class="form-group">
49             <label>Confirmer mot de passe*</label>
50             <input type="password" class="form-control" name="confirmerMotDePasse" required>
51         </div>
52         <label style="color: red"><?php if($erreurMessage != true){echo $erreurMessage;} ?></label>
53         <small>*Champs obligatoires</small>
54     <br/>
55     <button type="submit" class="btn btn-primary">Créer un compte</button>
56
57     </form>
58 </div>
59
60 <!--bootstrap-->
61 <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/X+
965Dz00rT7abK41JStQIAqVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo" crossorigin="anonymous"></script>
62 <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="
sha384-U0e2t0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDzWrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></
script>
63 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-
-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFf/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
64 </body>
65 </html>

```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : deconnecter.php
9 */
9 session_start();
10 //Verifie que la session est vide si ça n'est pas le cas la vide et redirige vers la page "index.php"
11 if(isset($_SESSION["utilisateur"]))
12 {
13     session_destroy();
14 }
15 header("Location: index.php");
16 exit();
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 08.04.19
7 * Version : 1.0
8 * Fichier : modifierHistoire.php
9 */
10 session_start();
11 if(!isset($_SESSION["utilisateur"]))
12 {
13     header("Location: index.php");
14     exit();
15 }
16 require_once("../Controleur/controleur.inc.php");
17 require_once("../Controleur/modifierHistoire.inc.php");
18
19 ?>
20 <!doctype html>
21 <html lang="fr">
22 <head>
23
24     <meta charset="utf-8">
25     <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
26
27     <!-- Bootstrap CSS -->
28     <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/css/bootstrap.min.css" integrity="sha384-ggOyR0iXCbMQv3Xipma34MD+dH/1fQ784/j6cY/iJTQUOhcWr7x9JvoRxT2MZw1T" crossorigin="anonymous">
29     <title>Tales of the tavern</title>
30 </head>
31 <body>
32 <?php include_once("./navbar.php");?>
33 <br/>
34 <div class="container col-sm-12 col-md-6 c border-1">
35     <form action="#" method="post" enctype="multipart/form-data">
36         <div class="form-group">
37             <label for="exampleInputEmail1">Titre*</label>
38             <input type="text" name="titre" class="form-control" value=<?= $titreHistoire ?><br/>
39             required>
40             </div>
41             <div class="form-group">
42                 <label>Histoire*</label>
43                 <textarea class="form-control" rows="5" name="histoire" required> <?= $chaineHistoire ?>
44             </textarea>
45             </div>
46             <div class="form-group">
47                 <label>Catégorie*</label>
48                 <select class="form-control" name="categorie">
49                     <?php
50                     if(isset($_GET["id"]))
51                     {
52                         AfficherTouteLesCategorieAvecUneSelectionner($categorieHistoire);
53                     }
54                     else
55                     {
56                         AfficherTouteLesCategorie();
57                     }
58                     ?>
59                 </select>
60             </div>
61             <input type="file" id="image" name="image" accept="image/*">
62             <br/>
63             <label style="color: red"><?php if($erreurMessage !== true){echo $erreurMessage;} ?></label>
64             <small>*Champs obligatoires</small>
65             <br/>
66             <button type="submit" class="btn btn-primary"><?= $texteButton ?></button>
67         </form>
68     </div>
69     <!--bootstrap-->
70     <script src="https://code.jquery.com/jquery-3.3.1.slim.min.js" integrity="sha384-q8i/Xt9
965Dz0OrT7abK41JStQIAgVgRVzbzo5smXKp4YfRvH+8abTE1Pi6jizo" crossorigin="anonymous"></script>
71     <script src="https://cdnjs.cloudflare.com/ajax/libs/popper.js/1.14.7/umd/popper.min.js" integrity="sha384-U02eT0CpHqdSJQ6hJty5KVphtPhzWj9WO1clHTMGa3JDZwrnQq4sF86dIHNDz0W1" crossorigin="anonymous"></

```

```
70 <script>
71 <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.3.1/js/bootstrap.min.js" integrity="sha384-JjSmVgyd0p3pXB1rRibZUAYoIIy6OrQ6VrjIEaFF/nJGzIxFDsf4x0xIM+B07jRM" crossorigin="anonymous"></script>
72 </body>
73 </html>
```

```
1 deny from all
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : fonctionBD.php
9 */
10
11 /**
12 * @param $nomUtilisateur string non utilisateur
13 * @param $emailUtilisateur string email utilisateur
14 * @param $motDePasse string mot de passe
15 */
16
17 function AjouterUtilisateur($nomUtilisateur,$emailUtilisateur,$motDePasse)
18 {
19     $connexion = RecupererConnexion();
20     $requete = $connexion->prepare("INSERT INTO utilisateur(nom, email, motDePasse) VALUES (:nom,:email,:motDePasse)");
21     $requete->bindParam(":nom", $nomUtilisateur, PDO::PARAM_STR);
22     $requete->bindParam(":email", $emailUtilisateur, PDO::PARAM_STR);
23     $requete->bindParam(":motDePasse", $motDePasse, PDO::PARAM_STR);
24     $requete->execute();
25 }
26
27 /**
28 * @param $emailUtilisateur string email de l'utilisateur
29 * @param $nomUtilisateur string nouveau nom de l'utilisateur
30 */
31
32 function ModifierUtilisateurNom($emailUtilisateur,$nomUtilisateur)
33 {
34     $connexion = RecupererConnexion();
35     $requete = $connexion->prepare("UPDATE utilisateur SET nom=:nom WHERE email=:email");
36     $requete->bindParam(":nom", $nomUtilisateur, PDO::PARAM_STR);
37     $requete->bindParam(":email", $emailUtilisateur, PDO::PARAM_STR);
38     $requete->execute();
39 }
40
41 /**
42 * @param $emailUtilisateur string ancien email de l'utilisateur
43 * @param $nomUtilisateur string nouveau nom de l'utilisateur
44 * @param $nouvelEmail string nouvel email de l'utilisateur
45 */
46
47 function ModifierUtilisateurNomEmail($emailUtilisateur,$nomUtilisateur,$nouvelEmail)
48 {
49     $connexion = RecupererConnexion();
50     $requete = $connexion->prepare("UPDATE utilisateur SET nom=:nom,email=:nouvelEmail WHERE email=:
51     $requete->bindParam(":nom", $nomUtilisateur, PDO::PARAM_STR);
52     $requete->bindParam(":nouvelEmail", $nouvelEmail, PDO::PARAM_STR);
53     $requete->bindParam(":email", $emailUtilisateur, PDO::PARAM_STR);
54     $requete->execute();
55 }
56
57 /**
58 * @param $emailUtilisateur string ancien email de l'utilisateur
59 * @param $nomUtilisateur string nouveau nom de l'utilisateur
60 * @param $nouvelEmail string nouvel email de l'utilisateur
61 * @param $motDePasse string nouveau mot de passe de l'utilisateur
62 */
63
64 function ModifierUtilisateurNomEmailMotDePasse($emailUtilisateur,$nomUtilisateur,$nouvelEmail,
65 $motDePasse)
66 {
67     $connexion = RecupererConnexion();
68     $requete = $connexion->prepare("UPDATE utilisateur SET nom=:nom,email=:nouvelEmail,motDePasse=:
69     $requete->bindParam(":nom", $nomUtilisateur, PDO::PARAM_STR);
70     $requete->bindParam(":nouvelEmail", $nouvelEmail, PDO::PARAM_STR);
71     $requete->bindParam(":motDePasse", $motDePasse, PDO::PARAM_STR);
72     $requete->bindParam(":email", $emailUtilisateur, PDO::PARAM_STR);
73     $requete->execute();
74 }
75
76 /**
77 * @param $emailUtilisateur string email de l'utilisateur
78 * @return array|bool retourne un utilisateur si l'utilisateur le trouve | false si il n'y a rien
79 */
80

```

```

73 function RetrouverUtilisateur($emailUtilisateur)
74 {
75     $connexion = RecupererConnexion();
76     $requete = $connexion->prepare("SELECT * FROM utilisateur WHERE email = :email");
77     $requete->bindParam(":email", $emailUtilisateur, PDO::PARAM_STR);
78     $requete->execute();
79     $resultat = $requete->fetch(PDO::FETCH_ASSOC);
80     return $resultat;
81 }
82
83 /** ceci Permet de récupérer toute les catégories
84 * array|bool retourne toute les catégorie | false si il n'y a rien
85 */
86 function RetrouverTouteLesCatégories()
87 {
88     $connexion = RecupererConnexion();
89     $requete = $connexion->prepare("SELECT * FROM categorie");
90     $requete->bindParam(":email", $emailUtilisateur, PDO::PARAM_STR);
91     $requete->execute();
92     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
93     return $resultat;
94 }
95
96 /** ceci permet de rajouter une histoire dans la base de données
97 * @param $titre string titre de l'histoire
98 * @param $histoire string texte de l'histoire
99 * @param $idImage int id de l'image
100 * @param $idCatégorie int id de la catégorie
101 * @param $idUtilisateur int id de l'utilisateur
102 */
103 function AjouterHistoire($titre,$histoire,$idImage,$idCatégorie,$idUtilisateur)
104 {
105     $connexion = RecupererConnexion();
106     $requete = $connexion->prepare("INSERT INTO histoire(titre, DateCreation, histoire, idImage,
107     idCatégorie, idUtilisateur) VALUES (:titre,NOW(),:histoire,:idImage,:idCatégorie,:idUtilisateur)");
108     $requete->bindParam(":titre", $titre, PDO::PARAM_STR);
109     $requete->bindParam(":idImage", $idImage, PDO::PARAM_INT);
110     $requete->bindParam(":idCatégorie", $idCatégorie, PDO::PARAM_INT);
111     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
112     $requete->execute();
113 }
114
115 /** ceci permet de rajouter une image (url) dans la db
116 * @param $urlImage string url de l'image
117 * @return int last inserted id
118 */
119 function AjouterImage($urlImage)
120 {
121     $connexion = RecupererConnexion();
122     $requete = $connexion->prepare("INSERT INTO image(urlImageHistoire) VALUES (:url)");
123     $requete->bindParam(":url", $urlImage, PDO::PARAM_STR);
124     $requete->execute();
125     return $connexion->lastInsertId();
126 }
127
128 /** ceci permet de modifier une histoire
129 * @param $idHistoire int id de l'histoire
130 * @param $titre string titre de l'histoire
131 * @param $histoire string texte de l'histoire
132 * @param $idImage int id de l'image
133 * @param $idCatégorie int id catégorie
134 */
135 function ModifierHistoire($idHistoire, $titre,$histoire,$idImage,$idCatégorie)
136 {
137     $connexion = RecupererConnexion();
138     $requete = $connexion->prepare("UPDATE histoire SET titre=:titre,histoire=:histoire,idImage=:
139     idImage,idCatégorie=:idCatégorie WHERE idHistoire = :idHistoire");
140     $requete->bindParam(":titre", $titre, PDO::PARAM_STR);
141     $requete->bindParam(":histoire", $histoire, PDO::PARAM_STR);
142     $requete->bindParam(":idImage", $idImage, PDO::PARAM_INT);
143     $requete->bindParam(":idCatégorie", $idCatégorie, PDO::PARAM_INT);
144     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
145     $requete->execute();
146 }
147 /** ceci permet de supprimer une histoire

```

```

148 * @param $idHistoire int id de l'histoire
149 */
150 function SupprimerHistoire($idHistoire)
151 {
152     $connexion = RecupererConnexion();
153     $requete = $connexion->prepare("DELETE FROM histoire WHERE idHistoire = :idHistoire");
154     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
155     $requete->execute();
156 }
157
158 /** ceci permet de retrouver une histoire la personne qui l'a créer sa catégorie son email et les 5
   moyennes de l'histoires
159 * @param $idHistoire int id de l'histoire
160 * @return array|bool retourne histoire, utilisateur, catégorie, moyenne | false si il n'y a rien
161 */
162 function RetrouverHistoireParId($idHistoire)
163 {
164     $connexion = RecupererConnexion();
165     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, his.idImage, his.
   idCategorie, email, urlImageHistoire,urlImageCategorie, nomCategorie, nom,
   (AVG(style) + AVG(eva.histoire) + AVG(orthographe) + AVG
   (originalite))/4 as moyenne,
   AVG(style) as moyenneStyle,
   AVG(eva.histoire) as moyenneHistoire,
   AVG(orthographe) as moyenneOrthographe,
   AVG(originalite) as moyenneOriginalite
   FROM histoire as his
   LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti.
   idUtilisateur
   LEFT JOIN categorie as cat ON his.idCategorie = cat.
   idCategorie
   LEFT JOIN image as ima ON his.idImage = ima.idImage
   LEFT JOIN evaluation as eva ON his.idHistoire = eva.
   idHistoire
   WHERE his.idHistoire = :idHistoire
   GROUP BY idHistoire");
178     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
179     $requete->execute();
180     $resultat = $requete->fetch(PDO::FETCH_ASSOC);
181     return $resultat;
182 }
183
184 /** ceci permet de retrouver une histoire la personne qui l'a créer sa catégorie son email et les 5
   moyennes de l'histoires
185 * @param $idUtilisateur int id de l'utilisateur
186 * @return array|bool retourne histoire, utilisateur, catégorie, moyenne | false si il n'y a rien
187 */
188 function RetrouverTouteHistoireParUtilisateur($idUtilisateur)
189 {
190     $connexion = RecupererConnexion();
191     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
   urlImageCategorie, nomCategorie, nom,
   (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
   AVG(originalite))/4 as moyenne
   FROM histoire as his
   LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti.
   idUtilisateur
   LEFT JOIN categorie as cat ON his.idCategorie = cat.
   idCategorie
   LEFT JOIN image as ima ON his.idImage = ima.idImage
   LEFT JOIN evaluation as eva ON his.idHistoire = eva.
   idHistoire
   WHERE his.idUtilisateur = :idUtilisateur
   GROUP BY idHistoire");
200     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
201     $requete->execute();
202     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
203     return $resultat;
204 }
205
206 /** ceci permet de retrouver les favoris d'un utilisateur trier par date
207 * @param $idUtilisateur int id de l'utilisateur
208 * @return array|bool retourne des histoires | false si il n'y a rien
209 */
210 function RetrouverTouteFavorisTrierParDate($idUtilisateur)
211 {
212     $connexion = RecupererConnexion();

```

```

213     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
214                                     urlImageCategorie, nomCategorie, nom,
215                                     (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
216                                     AVG(originalite))/4 as moyenne
217                                     FROM estfavoris as fav
218                                     LEFT JOIN histoire as his ON fav.idHistoire = his.
219                                     idHistoire
220                                     LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti
221                                     .idUtilisateur
222                                     LEFT JOIN categorie as cat ON his.idCategorie = cat.
223                                     idCategorie
224                                     LEFT JOIN image as ima ON his.idImage = ima.idImage
225                                     LEFT JOIN evaluation as eva ON his.idHistoire = eva.
226                                     idHistoire
227                                     WHERE fav.idUtilisateur = :idUtilisateur
228                                     GROUP BY idHistoire
229                                     ORDER By dateCreation DESC");
230
231     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
232     $requete->execute();
233     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
234     return $resultat;
235 }
236 /**
237  * return array|bool retourne des histoires | false si il n'y a rien
238 */
239 function RetrouverTouteHistoireTrierParDate()
240 {
241     $connexion = RecupererConnexion();
242     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
243                                     urlImageCategorie, nomCategorie, nom,
244                                     (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
245                                     AVG(originalite))/4 as moyenne
246                                     FROM histoire as his
247                                     LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti
248                                     .idUtilisateur
249                                     LEFT JOIN categorie as cat ON his.idCategorie = cat.
250                                     idCategorie
251                                     LEFT JOIN image as ima ON his.idImage = ima.idImage
252                                     LEFT JOIN evaluation as eva ON his.idHistoire = eva.
253                                     idHistoire
254                                     GROUP BY idHistoire
255                                     ORDER By dateCreation DESC");
256
257     $requete->execute();
258     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
259     return $resultat;
260 }
261 /**
262  * param $idUtilisateur int id de l'utilisateur
263  * return array|bool retourne des histoires | false si il n'y a rien
264 */
265 function RetrouverTouteFavorisTrierParMoyenne($idUtilisateur)
266 {
267     $connexion = RecupererConnexion();
268     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
269                                     urlImageCategorie, nomCategorie, nom,
270                                     (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
271                                     AVG(originalite))/4 as moyenne
272                                     FROM estfavoris as fav
273                                     LEFT JOIN histoire as his ON fav.idHistoire = his.
274                                     idHistoire
275                                     LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti
276                                     .idUtilisateur
277                                     LEFT JOIN categorie as cat ON his.idCategorie = cat.
278                                     idCategorie
279                                     LEFT JOIN image as ima ON his.idImage = ima.idImage
280                                     LEFT JOIN evaluation as eva ON his.idHistoire = eva.
281                                     idHistoire
282                                     WHERE fav.idUtilisateur = :idUtilisateur
283                                     GROUP BY idHistoire
284                                     ORDER By moyenne DESC");
285
286     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
287     $requete->execute();
288     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
289     return $resultat;
290 }
291 /**
292  * return array|bool retourne des histoires | false si il n'y a rien
293 */

```

```

273 */
274 function RetrouverTouteHistoireTrierParMoyenne()
275 {
276     $connexion = RecupererConnexion();
277     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
278                                     urlImageCategorie, nomCategorie, nom,
279                                     (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
280                                     AVG(originalite))/4 as moyenne
281                                     FROM histoire as his
282                                     LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti
283                                     LEFT JOIN categorie as cat ON his.idCategorie = cat.
284                                     LEFT JOIN image as ima ON his.idImage = ima.idImage
285                                     LEFT JOIN evaluation as eva ON his.idHistoire = eva.
286                                     GROUP BY idHistoire
287                                     ORDER By moyenne DESC");
288
289     $requete->execute();
290     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
291     return $resultat;
292 }
293
294 /**
295  * @param $noteStyle int note sur le style
296  * @param $noteHistoire int note sur le histoire(texte)
297  * @param $noteOrthographe int note sur le Orthographe
298  * @param $noteOriginalite int note sur le Originalité
299  * @param $idHistoire int id de l'histoire
300  */
301 function AjouterEvaluation($noteStyle, $noteHistoire,$noteOrthographe,$noteOriginalite,$idHistoire)
302 {
303     $connexion = RecupererConnexion();
304     $requete = $connexion->prepare("INSERT INTO evaluation(style, histoire, orthographe, originalite
305 , idHistoire) VALUES (:noteStyle,:noteHistoire,:noteOrthographe,:noteOriginalite,:idHistoire)");
306     $requete->bindParam(":noteStyle", $noteStyle, PDO::PARAM_INT);
307     $requete->bindParam(":noteHistoire", $noteHistoire, PDO::PARAM_INT);
308     $requete->bindParam(":noteOrthographe", $noteOrthographe, PDO::PARAM_INT);
309     $requete->bindParam(":noteOriginalite", $noteOriginalite, PDO::PARAM_INT);
310     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
311     $requete->execute();
312
313 /**
314  * @param $idUtilisateur int id de l'utilisateur
315  * @param $idHistoire int id de l'histoire
316  */
317 function AjouterFavoris($idUtilisateur,$idHistoire)
318 {
319     $connexion = RecupererConnexion();
320     $requete = $connexion->prepare("INSERT INTO estfavoris(idHistoire, idUtilisateur) VALUES (:idHistoire,:idUtilisateur)");
321     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
322     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
323     $requete->execute();
324
325 /**
326  * @param $idUtilisateur int id de l'utilisateur
327  * @param $idHistoire int id de l'histoire
328  */
329 function SupprimerFavoris($idUtilisateur,$idHistoire)
330 {
331     $connexion = RecupererConnexion();
332     $requete = $connexion->prepare("DELETE FROM estfavoris WHERE idHistoire = :idHistoire AND
333     idUtilisateur = :idUtilisateur");
334     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
335     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
336     $requete->execute();
337
338 /**
339  * @param $idUtilisateur int id de utilisateur
340  * @param $idHistoire int id de l'histoire
341  * @return array|bool retourne un favoris | false si il n'y a rien
342  */
343 function RetrouverFavoris($idUtilisateur,$idHistoire)
344 {

```

```

342     $connexion = RecupererConnexion();
343     $requete = $connexion->prepare("SELECT * FROM estfavoris WHERE idHistoire = :idHistoire AND
344     idUtilisateur = :idUtilisateur");
345     $requete->bindParam(":idHistoire", $idHistoire, PDO::PARAM_INT);
346     $requete->bindParam(":idUtilisateur", $idUtilisateur, PDO::PARAM_INT);
347     $requete->execute();
348     $resultat = $requete->fetch(PDO::FETCH_ASSOC);
349     return $resultat;
350 }
351 /**
352 * @param $titre string titre de l'histoire
353 * @return array|bool retourne des histoires | false si il n'y a rien
354 */
355 function RetrouverHistoireParTitre($titre)
356 {
357     $connexion = RecupererConnexion();
358     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
359     urlImageCategorie, nomCategorie, nom,
360                                         (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
361                                         AVG(originalite))/4 as moyenne
362                                         FROM histoire as his
363                                         LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti.
364                                         idUtilisateur
365                                         LEFT JOIN categorie as cat ON his.idCategorie = cat.
366                                         idCategorie
367                                         LEFT JOIN image as ima ON his.idImage = ima.idImage
368                                         LEFT JOIN evaluation as eva ON his.idHistoire = eva.
369                                         idHistoire
370                                         WHERE titre LIKE :titre
371                                         GROUP BY idHistoire
372                                         ORDER By dateCreation DESC");
373     $requete->bindParam(":titre", $titre, PDO::PARAM_STR);
374     $requete->execute();
375     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
376     return $resultat;
377 }
378 /**
379 * @param $titre string titre de l'histoire
380 * @return array|bool retourne des histoires | false si il n'y a rien
381 */
382 function RetrouverHistoireParNom($nom)
383 {
384     $connexion = RecupererConnexion();
385     $requete = $connexion->prepare("SELECT his.idHistoire, titre, his.histoire, urlImageHistoire,
386     urlImageCategorie, nomCategorie, nom,
387                                         (AVG(style) + AVG(eva.histoire) + AVG(orthographe) +
388                                         AVG(originalite))/4 as moyenne
389                                         FROM histoire as his
390                                         LEFT JOIN utilisateur as uti ON his.idUtilisateur = uti.
391                                         idUtilisateur
392                                         LEFT JOIN categorie as cat ON his.idCategorie = cat.
393                                         idCategorie
394                                         LEFT JOIN image as ima ON his.idImage = ima.idImage
395                                         LEFT JOIN evaluation as eva ON his.idHistoire = eva.
396                                         idHistoire
397                                         WHERE nom LIKE :nom
398                                         GROUP BY idHistoire
399                                         ORDER By nom ASC, dateCreation DESC");
400     $requete->bindParam(":nom", $nom, PDO::PARAM_STR);
401     $requete->execute();
402     $resultat = $requete->fetchAll(PDO::FETCH_ASSOC);
403     return $resultat;
404 }
405 ?>
```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : connexionBD.php
9 */
10 /* fonction pour connecter la base de données */
11 require_once("../Modele/constanteBD.php");
12 /** permet de se connecter à la base de données
13 * @return PDO retourne la base de données
14 */
15
16 function RecupererConnexion() {
17     static $dbc = null;
18
19     if ($dbc == null) {
20         try {
21             $dbc = new PDO('mysql:host=' . DB_HOST . ';dbname=' . DB_NAME,
22                           DB_USER, DB_PASSWORD, array(PDO::MYSQL_ATTR_INIT_COMMAND => "SET NAMES utf8",
23                                         PDO::ATTR_PERSISTENT => true));
24         }
25         catch (Exception $e) {
26             die('impossible de se connecter à la base de données');
27         }
28     }
29     return $dbc;
30 }
```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : constanteBD.php
9 */
10 DEFINE('DB_USER', 'TalesTPI');
11 DEFINE('DB_PASSWORD', 'Tales19');
12 DEFINE('DB_HOST', 'localhost');
```

```
1 deny from all
```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 */
9 $histoires = "";
10 $favoris = "";
11 if(isset($_GET["ordre"]))
12 {
13     $get = $_GET["ordre"];
14 }
15 else
16 {
17     $get = "default";
18 }
19 $histoires = RetournerTouteHistoire($get);
20 if(isset($_SESSION["utilisateur"]))
21 {
22
23     $favoris = RetournerToutFavoris($get,$_SESSION["utilisateur"]);
24 }
25 /** Affiche une histoire
26 * @param $histoires int id histoire a retourner
27 */
28 function afficherHistoires($histoires)
29 {
30     for($i = 0; $i < count($histoires); $i++)
31     {
32         if($histoires[$i]["urlImageHistoire"] == null)
33         {
34             $histoires[$i]["urlImageHistoire"] = $histoires[$i]["urlImageCategorie"];
35         }
36         echo AfficherHistoire($histoires[$i]["idHistoire"],$histoires[$i]["urlImageHistoire"],
37         $histoires[$i]["titre"],$histoires[$i]["nom"],$histoires[$i]["nomCategorie"],$histoires[$i][
38         "histoire"],false);
39     }
40 }
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 05.04.19
7 * Version : 1.0
8 */
9 $nouveauNom = isset($_POST["nom"]) ? trim(filter_input(INPUT_POST, 'nom', FILTER_SANITIZE_STRING)) : "";
10 $nouvelEmail = isset($_POST["email"]) ? trim(filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL)) :
11 "";
12 $mdp = isset($_POST["motDePasse"]) ? trim(filter_input(INPUT_POST, 'motDePasse', FILTER_SANITIZE_STRING))
13 ) : "";
14 $nouveauMdp = isset($_POST["nouveauMotDePasse"]) ? trim(filter_input(INPUT_POST, 'nouveauMotDePasse',
15 FILTER_SANITIZE_STRING)) : "";
16 $confMdp = isset($_POST["confirmerNouveauMotDePasse"]) ? trim(filter_input(INPUT_POST,
17 'confirmerNouveauMotDePasse', FILTER_SANITIZE_STRING)) : "";
18 $moyenneAuteur = RetournerMoyenneUtilisateur($_SESSION["utilisateur"]);
19 if($nouveauNom != "") {
20     if(strtolower($nouvelEmail) == strtolower($_SESSION["utilisateur"])) || !UtilisateurExiste(
21     $nouvelEmail) {
22         if (filter_var($nouvelEmail, FILTER_VALIDATE_EMAIL)) {
23             if (UtilisateurExisteEtMotDePasseJuste($_SESSION["utilisateur"], $mdp)) {
24                 $MotDePasseEstJuste = VerifieMotDePasse($nouveauMdp, $confMdp);
25                 if ($MotDePasseEstJuste === true) {
26                     ModifierUtilisateurNomEmailMotDePasse($_SESSION["utilisateur"], $nouveauNom,
27 $nouvelEmail,hash("sha256",$nouveauMdp));
28                     $_SESSION["utilisateur"] = $nouvelEmail;
29                     true;
30                 } else {
31                     ModifierUtilisateurNomEmail($_SESSION["utilisateur"],$nouveauNom,$nouvelEmail);
32                     $_SESSION["utilisateur"] = $nouvelEmail;
33                     $erreurMessage = "le mot de passe n'a pas été modifier, car " .
34                     $MotDePasseEstJuste;
35                 }
36             } else {
37                 ModifierUtilisateurNom($_SESSION["utilisateur"],$nouveauNom);
38                 $erreurMessage = "le nom a bien été modifier mais, le mot de passe et email non car,
39 $nouvelEmail n'est pas une adresse email valide ";
40             }
41         } else {
42             ModifierUtilisateurNom($_SESSION["utilisateur"],$nouveauNom);
43             $erreurMessage = "le nom a bien été modifier mais, le mot de passe et email non car, cette
44 adresse email existe déjà";
45         }
46     }
47     $utilisateur = RetournerUtilisateur($_SESSION["utilisateur"]);
48     $nom = $utilisateur["nom"];
49     $email = $utilisateur["email"];
50     /** Affiche toute les histoire d'un utilisateur avec les button supprimer et ajouter
51     * @return array listes histoires
52     */
53     function afficherHistoiresUtilisateur()
54     {
55         $histoire = RetournerTouteHistoireCreerParUnUtilisateur($_SESSION["utilisateur"]);
56         for($i = 0; $i < count($histoire); $i++)
57         {
58             if($histoire[$i]["urlImageHistoire"] == null)
59             {
60                 $histoire[$i]["urlImageHistoire"] = $histoire[$i]["urlImageCategorie"];
61             }
62             echo AfficherHistoire($histoire[$i]["idHistoire"],$histoire[$i]["urlImageHistoire"],
63             $histoire[$i]["titre"],$histoire[$i]["nom"],$histoire[$i]["nomCategorie"],$histoire[$i]["histoire"],
64             true);
65         }
66     }

```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 11.04.19
7 * Version : 1.0
8 * Fichier : histoire.inc.php
9 */
10 $idHistoire = isset($_GET["id"]) ? trim(filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT)) : "";
11 $histoire = RetournerHistoireParId($idHistoire);
12 if($histoire === false){
13     header("location: index.php");
14     exit();
15 }
16 $noteStyle = isset($_POST["noteStyle"]) ? trim(filter_input(INPUT_POST, 'noteStyle',
17 FILTER_SANITIZE_NUMBER_INT)) : "";
18 $noteHistoire = isset($_POST["noteHistoire"]) ? trim(filter_input(INPUT_POST, 'noteHistoire',
19 FILTER_SANITIZE_NUMBER_INT)) : "";
20 $noteOrthographe = isset($_POST["noteOrthographe"]) ? trim(filter_input(INPUT_POST, 'noteOrthographe',
21 FILTER_SANITIZE_NUMBER_INT)) : "";
22 $noteOriginialite = isset($_POST["noteOriginialite"]) ? trim(filter_input(INPUT_POST, 'noteOriginialite
23 ', FILTER_SANITIZE_NUMBER_INT)) : "";
24
25 if(isset($_SESSION["utilisateur"])){
26     if(isset($_POST["favoris"])){
27         if(EstFavoris($_SESSION["utilisateur"], $idHistoire)){
28             SupprimerFavorisParId($_SESSION["utilisateur"], $idHistoire);
29         } else {
30             InsererFavoris($_SESSION["utilisateur"], $idHistoire);
31         }
32     }
33     if(EstFavoris($_SESSION["utilisateur"],$idHistoire))
34     {
35         $buttonText = "Supprimer aux favoris";
36     }
37     else{
38         $buttonText = "Ajouter aux favoris";
39     }
40 }
41 if($noteStyle != "" && $noteHistoire != "" && $noteOrthographe != "" && $noteOriginialite != "" && !
42 isset($_POST["favoris"])){
43     InsererEvaluation($noteStyle,$noteHistoire,$noteOrthographe,$noteOriginialite,$idHistoire);
44     header("location: index.php");
45     exit();
46 }
47 if($histoire["urlImageHistoire"] === null){
48     $urlImage = $histoire["urlImageCategorie"];
49 }
50 else{
51     $titre = $histoire["titre"];
52     $moyenneGlobalHistoire = round($histoire["moyenne"],1);
53     $auteur = $histoire["nom"];
54     $moyenneAuteur = RetournerMoyenneUtilisateur($histoire["email"]);
55     $categorie = $histoire["nomCategorie"];
56     $texteHistoire = $histoire["histoire"];
57     $moyenneStyle = round($histoire["moyenneStyle"],1);
58     $moyenneHistoire = round($histoire["moyenneHistoire"],1);
59     $moyenneOrthographe = round($histoire["moyenneOrthographe"],1);
60     $moyenneOriginalite =round($histoire["moyenneOriginalite"],1);
61
62 /** affiche les options de 1 a 5
63 */
64 function AfficherNotation()
65 {
66     for($i = 1; $i <= 5; $i++)
67     {
68         echo "<option value=\"" . $i . "\" selected > " . $i . "</option>";
69     }
70 }

```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : connexion.inc.php
9 */
9 $erreurMessage = "";
10 $email = isset($_POST["email"]) ? trim(filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL)) : "";
11 $mdp = isset($_POST["motDePasse"]) ? trim(filter_input(INPUT_POST, 'motDePasse', FILTER_SANITIZE_STRING))
12 : "";
12 if( $email != "" && $mdp != "") {
13     if (UtilisateurExisteEtMotDePasseJuste($email, $mdp)) {
14         //Mets l'email dans la session et le redirige à la page index.php
15         $_SESSION["utilisateur"] = $email;
16         header("Location: index.php");
17         exit();
18     } else {
19         $erreurMessage = "Le mot de passe ou l'Email est faux";
20     }
21 }
```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 05.04.19
7 * Version : 1.0
8 * Fichier : supprimer.inc.php
9 */
10 $idHistoire = isset($_GET["id"]) ? trim(filter_input(INPUT_GET, 'id', FILTER_SANITIZE_NUMBER_INT)) : "";
11 $histoire = RetournerHistoireParId($idHistoire);
12 if ($histoire === null || strtolower($histoire["email"]) != strtolower($_SESSION["utilisateur"]))
13 {
14     header("Location: compte.php");
15     exit();
16 if(isset($_POST["supprimer"]))
17 {
18     SupprimerHistoireParId($idHistoire);
19     header("Location: compte.php");
20     exit();
21 }
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : controleur.inc.php
9 */
10 require_once("../Modele/fonctionBD.php");
11 define("CHARMIN", 8);
12 /** appelle la fonction pour ajouter un utilisateur
13 * @param $nomUtilisateur string nom de l'utilisateur
14 * @param $emailUtilisateur string email de l'utilisateur
15 * @param $motDePasse string mot de passe de l'utilisateur
16 */
17 function InsererUtilisateur($nomUtilisateur,$emailUtilisateur,$motDePasse)
18 {
19     AjouterUtilisateur($nomUtilisateur, $emailUtilisateur,$motDePasse);
20 }
21 /** appelle la fonction pour modifier un utilisateur
22 * @param $nomUtilisateur string nom de l'utilisateur
23 * @param $emailUtilisateur string ancien email de l'utilisateur
24 * @param $nouvelEmail string nouvel email de l'utilisateur
25 * @param $motDePasse string mot de passe de l'utilisateur
26 */
27 function ModifierUtilisateurParEmail($nomUtilisateur,$emailUtilisateur,$nouvelEmail,$motDePasse)
28 {
29     ModifierUtilisateur($nomUtilisateur,$emailUtilisateur,$nouvelEmail,hash("sha256", $motDePasse));
30 }
31
32 /** vérifie que les 2 mot de passe sont égale et sont 8 caractère de long et contient au moins une
lettre et un chiffre
33 * @param $motDePasse string mot de passe
34 * @param $motDePasseConfirmation string deuxième mot de passe
35 * @return bool|string true si tout c'est bien passer | le text d'erreur
36 */
37 function VerifieMotDePasse($motDePasse,$motDePasseConfirmation)
38 {
39     if (preg_match('~[0-9]~', $motDePasse) >= 1 && preg_match('~[a-zA-Z]~', $motDePasse) >= 1) {
40         if (strlen($motDePasse) >= CHARMIN) {
41             if ($motDePasse === $motDePasseConfirmation) {
42                 return true;
43             } else {
44                 return "les mots de passe ne correspondent pas";
45             }
46         } else {
47             return "le mot de passe doit contenir au minimum de 8 caractères";
48         }
49     } else {
50         return "Le mot de passe doit contenir au minimum 8 caractères";
51     }
52 }
53
54 /** appelle la fonction pour retourner un utilisateur
55 * @param $emailUtilisateur $email de l'utilisateur
56 * @return array|bool liste contenant un utilisateur | false si il n'existe pas
57 */
58 function RetournerUtilisateur($emailUtilisateur)
59 {
60     $utilisateur = RetrouverUtilisateur($emailUtilisateur);
61     return $utilisateur;
62 }
63 /** appelle la fonction pour retourner toute les catégorie
64 * @return array|bool liste contenant toute les catégories | false si elles n'existent pas
65 */
66 function RetournerTouteLesCategories()
67 {
68     $catégorie = RetrouverTouteLesCatégories();
69     return $catégorie;
70 }
71
72 /** fonction permettant de créer une carte histoire
73 * @param $idHistoire int id de l'histoire
74 * @param $urlImage string url de l'image
75 * @param $titre string titre de l'histoire

```

```

76 * @param $auteur string auteur de l'histoire
77 * @param $categorie string catégorie de l'histoire
78 * @param $histoire string texte de l'histoire
79 * @param $afficherBouton bool afficher les bouton supprimer / modifier
80 * @return string HTML à afficher
81 */
82 function AfficherHistoire($idHistoire,$urlImage,$titre,$auteur,$categorie,$histoire,$afficherBouton)
83 {
84     $histoireHTML = '<a style="color:inherit; text-decoration:none;" href="histoire.php?id=' . $idHistoire . '"><div class="h-100 card col-md-12 col-lg-4 mb-3">' .
85         . '' .
86         . '<h1 class="display-4">' . $titre . '</h1>' .
87         . '<p class="lead">' . $auteur . '</p>' .
88         . '<p class="lead">' . $categorie . '</p>' .
89         . '<p>' . substr($histoire,0,140) . '...' . '</p></a>';
90     if($afficherBouton) {
91         $histoireHTML .= '<div class="row"><a class="col-6 btn btn-primary btn-lg" href="'
92             . 'modifierHistoire.php?id=' . $idHistoire . '" role="button">Modifier</a>' .
93             . '<a class=" col-6 btn btn-danger btn-lg" href="supprimer.php?id=' . $idHistoire . '"'
94             . 'role="button">Supprimer</a></div>';
95     }
96 }
97
98 /** vérifie si l'histoire existe
99 * @param $emailUtilisateur string email de l'utilisateur
100 * @return bool true si il existe | false si il n'existe pas
101 */
102 function UtilisateurExiste($emailUtilisateur)
103 {
104     $utilisateur = RetrouverUtilisateur($emailUtilisateur);
105     if($utilisateur != null)
106     {
107         return true;
108     }else{
109         return false;
110     }
111 }
112 /** Permettant de savoir si un utilisateur existe et le mot de passe est égale a celui
113 * @param $emailUtilisateur string email de l'utilisateur
114 * @param $motDePasse string mot de passe
115 * @return bool true si l'utilisateur existe et le mot de passe est juste | false si le mot de passe
est faux ou l'utilisateur n'existe pas
116 */
117 function UtilisateurExisteEtMotDePasseJuste($emailUtilisateur,$motDePasse)
118 {
119     $utilisateur = RetrouverUtilisateur($emailUtilisateur);
120     if(UtilisateurExiste($emailUtilisateur))
121     {
122         if(hash("sha256",$motDePasse )== $utilisateur["motDePasse"])
123             return true;
124     }
125     return false;
126 }
127
128 /** appelle la fonction pour ajouter une image
129 * @param $urlImage string url de l'image
130 * @return int last inserted id
131 */
132 function InsererImage($urlImage)
133 {
134     return AjouterImage($urlImage);
135 }
136 function InsererHistoire($titre,$histoire,$idImage,$idCatégorie,$emailUtilisateur){
137     $idUtilisateur = $utilisateur = RetrouverUtilisateur($emailUtilisateur)[ "idUtilisateur" ];
138     AjouterHistoire($titre,$histoire,$idImage,$idCatégorie,$idUtilisateur);
139 }
140
141 /** appelle la fonction pour retourner une histoire
142 * @param $idHistoire
143 * @return array|bool liste contenant une histoire | false si elle n'existe pas
144 */
145 function RetournerHistoireParId($idHistoire)
146 {
147     $histoire = RetrouverHistoireParId($idHistoire);
148     return $histoire;

```

```

149 }
150
151 /** appelle la fonction pour retourner toutes les histoires d'un utilisateur
152 * @param $emailUtilisateur string email de l'utilisateur
153 * @return array|bool liste contenant des histoires | false si elles n'existent pas
154 */
155 function RetournerTouteHistoireCreerParUnUtilisateur($emailUtilisateur)
156 {
157     $idUtilisateur = RetrouverUtilisateur($emailUtilisateur) ["idUtilisateur"];
158     return RetrouverTouteHistoireParUtilisateur($idUtilisateur);
159 }
160
161 /** retourne tout les histoire trier d'une manière
162 * @param $trie string trie
163 * @return array|bool liste contenant des histoires | false si elles n'existent pas
164 */
165 function RetournerTouteHistoire($trie)
166 {
167     switch ($trie)
168     {
169         case "note":
170             return RetrouverTouteHistoireTrierParMoyenne();
171             break;
172         default:
173             return RetrouverTouteHistoireTrierParDate();
174             break;
175     }
176 }
177
178 /** retourne les favoris d'un utilisateur
179 * @param $trie string trie
180 * @param $emailUtilisateur string email utilisateur
181 * @return array|bool liste contenant des histoires | false si elles n'existent pas
182 */
183 function RetournerToutFavoris($trie,$emailUtilisateur)
184 {
185     $idUtilisateur = RetrouverUtilisateur($emailUtilisateur) ["idUtilisateur"];
186     switch ($trie)
187     {
188         case "note":
189             return RetrouverTouteFavorisTrierParMoyenne($idUtilisateur);
190             break;
191         default:
192             return RetrouverTouteFavorisTrierParDate($idUtilisateur);
193             break;
194     }
195 }
196
197 /** appelle la fonction pour supprimer une histoire
198 * @param $idHistoire int id de l'histoire
199 */
200 function SupprimerHistoireParId($idHistoire)
201 {
202     SupprimerHistoire($idHistoire);
203 }
204
205 /** appelle la fonction pour inserer une évaluation
206 * @param $noteStyle int note style
207 * @param $noteHistoire int note style
208 * @param $noteOrthographe int note Orthographe
209 * @param $noteOriginalite int note Originalité
210 * @param $idHistoire int id de l'histoire
211 */
212 function InsererEvaluation($noteStyle,$noteHistoire,$noteOrthographe,$noteOriginalite,$idHistoire)
213 {
214     AjouterEvaluation($noteStyle, $noteHistoire,$noteOrthographe,$noteOriginalite,$idHistoire);
215 }
216
217 /** appelle la fonction pour inserer un favoris
218 * @param $emailUtilisateur string email de l'utilisateur
219 * @param $idHistoire int id de l'histoire
220 */
221 function InsererFavoris($emailUtilisateur,$idHistoire)
222 {
223     $idUtilisateur = RetrouverUtilisateur($emailUtilisateur) ["idUtilisateur"];
224     AjouterFavoris($idUtilisateur,$idHistoire);
225 }

```

```

226
227 /** appelle la fonction pour supprimer un favoris
228 * @param $emailUtilisateur string email de l'utilisateur
229 * @param $idHistoire int id de l'histoire
230 */
231 function SupprimerFavorisParId($emailUtilisateur,$idHistoire)
232 {
233     $idUtilisateur = RetrouverUtilisateur($emailUtilisateur)["idUtilisateur"];
234     SupprimerFavoris($idUtilisateur,$idHistoire);
235 }
236
237 /** vérifie si l'utilisateur est favoris ou non
238 * @param $emailUtilisateur string email de l'utilisateur
239 * @param $idHistoire int id de l'histoire
240 * @return bool true si l'utilisateur est favoris a une histoire | false si l'utilisateur n'est pas
favoris a une histoire
241 */
242 function EstFavoris($emailUtilisateur,$idHistoire)
243 {
244     $idUtilisateur = RetrouverUtilisateur($emailUtilisateur)["idUtilisateur"];
245     if(RetrouverFavoris($idUtilisateur,$idHistoire) != false)
246     {
247         return true;
248     }
249     else{
250         return false;
251     }
252 }
253
254 /** retourne la moyenne d'un utilisateur si il n'y en a pas alors retourne 0
255 * @param $emailUtilisateur string email de l'utilisateur
256 * @return float|int moyenne
257 */
258 function RetournerMoyenneUtilisateur($emailUtilisateur)
259 {
260     $idUtilisateur = RetrouverUtilisateur($emailUtilisateur)["idUtilisateur"];
261     $histoires = RetrouverTouteHistoireParUtilisateur($idUtilisateur);
262     $sommeMoyenne = 0;
263     $cptHistoires = 0;
264     if(count($histoires) == 0)
265     {
266         return 0;
267     }
268     for($i = 0; $i < count($histoires); $i++)
269     {
270         $sommeMoyenne += $histoires[$i]["moyenne"];
271         $cptHistoires += 1;
272     }
273
274     return round($sommeMoyenne / $cptHistoires,1);
275 }
276
277 /** Retourne les histoires qui contiennent la recherche
278 * @param $recherche string text a chercher
279 * @return array histoires
280 */
281 function RetournerHistoireParTitre($recherche)
282 {
283     $recherche = "%" . $recherche . "%";
284     return RetrouverHistoireParTitre($recherche);
285 }
286 /** Retourne les histoires des utilisateur qui contiennent la recherche
287 * @param $recherche string text a chercher
288 * @return array histoires
289 */
290 function RetournerHistoireParNom($recherche)
291 {
292     $recherche = "%" . $recherche . "%";
293     return RetrouverHistoireParNom($recherche);
294 }

```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 12.04.19
7 * Version : 1.0
8 * Fichier : rechercher.inc.php
9 */
10 $histoiresParTitre = "";
11 $histoiresParAuteur = "";
12 $recherche = isset($_GET["recherche"]) ? trim(filter_input(INPUT_GET, 'recherche',
13 FILTER_SANITIZE_STRING)) : "";
13 if($recherche == "") {
14 {
15 header("location: index.php");
16 exit();
17 }
18 $histoiresParTitre = RetournerHistoireParTitre($recherche);
19 $histoiresParAuteur = RetournerHistoireParNom($recherche);
20 /** Affiche une histoire
21 * @param $histoires int id histoire a retourner
22 */
23 function afficherHistoires($histoires)
24 {
25 for($i = 0; $i < count($histoires); $i++)
26 {
27 if($histoires[$i]["urlImageHistoire"] == null)
28 {
29     $histoires[$i]["urlImageHistoire"] = $histoires[$i]["urlImageCategorie"];
30 }
31 echo AfficherHistoire($histoires["$i"]["idHistoire"],$histoires["$i"]["urlImageHistoire"],
32 $histoires[$i]["titre"],$histoires[$i]["nom"],$histoires[$i]["nomCategorie"],$histoires[$i][
33 "histoire"],false);
32 }
33 }
```

```
1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 04.04.19
7 * Version : 1.0
8 * Fichier : creerCompte.inc.php
9 */
9 $erreurMessage = "";
10 // Récupère les valeurs en POST et filtre les inputs
11 $nom = isset($_POST["nom"]) ? trim(filter_input(INPUT_POST, 'nom', FILTER_SANITIZE_STRING)) : "";
12 $email = isset($_POST["email"]) ? trim(filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL)) : "";
13 $mdp = isset($_POST["motDePasse"]) ? trim(filter_input(INPUT_POST, 'motDePasse', FILTER_SANITIZE_STRING)) :
14 ) : "";
14 $confMdp = isset($_POST["confirmerMotDePasse"]) ? trim(filter_input(INPUT_POST, 'confirmerMotDePasse',
15 FILTER_SANITIZE_STRING)) : "";
15
16 if($nom != "" && $email != "" && $mdp != "" && $confMdp != "") {
17 {
18
19     $erreurMessage = VerifieMotDePasse($mdp, $confMdp);
20     if ($erreurMessage === true) {
21         if(filter_var($email, FILTER_VALIDATE_EMAIL))
22         {
23             if (!UtilisateurExiste($email)) {
24                 InsererUtilisateur($nom, $email, hash("sha256", $mdp));
25                 $_SESSION["utilisateur"] = $email;
26                 header("Location: index.php");
27                 exit();
28             } else {
29                 $erreurMessage = "Cette adresse e-mail est déjà utilisée.";
30             }
31         }
32     } else {
33         $erreurMessage = "$email n'est pas une adresse email valide";
34     }
35 }
36 }
```

```

1 <?php
2 /* auteur : Raphael Lopes
3 * Projet : Tales of the Tavern
4 * description : Site internet permettant de stocker des histoires et que les autres utilisateurs
5 * puissent les noter
6 * date : 08.04.19
7 * Version : 1.0
8 * Fichier : modifierHistoire.inc.php
9 */
10 $titreHistoire = "";
11 $chaineHistoire = "";
12 $categorieHistoire = "";
13 $imageHistoire = "";
14 $texteButton = "Ajouter une histoire";
15 $idImage = null;
16
17 if(isset($_GET["id"])){
18     $idHistoire = trim(filter_input(INPUT_GET,'id',FILTER_SANITIZE_NUMBER_INT));
19     $texteButton = "Modifier l'histoire";
20     $histoire = RetournerHistoireParId($idHistoire);
21     if ($histoire === null || strtolower($histoire["email"]) != strtolower($_SESSION["utilisateur"])){
22         header("Location: compte.php");
23         exit();
24     }
25     $titreHistoire = $histoire["titre"];
26     $chaineHistoire = $histoire["histoire"];
27     $categorieHistoire = $histoire["idCategorie"];
28     $idImage = $histoire["idImage"];
29 }
30
31 $erreurMessage = "";
32 $titre = isset($_POST["titre"]) ? trim(filter_input(INPUT_POST,'titre',FILTER_SANITIZE_STRING)) : "";
33 $histoire = isset($_POST["histoire"]) ? trim(filter_input(INPUT_POST,'histoire',
34 FILTER_SANITIZE_STRING)) : "";
34 $categorie = isset($_POST["categorie"]) ? trim(filter_input(INPUT_POST,'categorie',
35 FILTER_SANITIZE_NUMBER_INT)) : "";
35 if($titre != "" && $histoire != "" && $categorie != ""){
36     if($_FILES["image"]["error"] != 4) // verifie si utilisateur a mis une image ou non
37     {
38         $Dossier = "Img/";
39         $extensionsAccepter = ['jpeg', 'jpg', 'png'];
40         $nomFichier = $_FILES['image']['name'];
41         $tailleFichier = $_FILES['image']['size'];
42         $NomTemporaire = $_FILES['image']['tmp_name'];
43         $typeDefichier = $_FILES['image']['type'];
44         $extension = explode('.', $nomFichier);
45         $extension = strtolower(end($extension));
46         $nomFichier = uniqid() . $nomFichier;
47         $cheminUpload = $Dossier . basename($nomFichier);
48         if (!in_array($extension, $extensionsAccepter)){
49             $erreurMessage = "ce type d'extension n'est pas accepté (jpeg, jpg, png)";
50         }
51
52         if ($tailleFichier > 5000000) { // 5000000 = 5MB
53             $erreurMessage = "Ce fichier fait plus que 5 Mb. Il doit être moins ou égal à 5 Mb.";
54         }
55         if ($erreurMessage == "" && $_FILES["image"]["error"] == 0) {
56             var_dump($cheminUpload);
57             move_uploaded_file($NomTemporaire, $cheminUpload);
58             $idImage = InsererImage($nomFichier);
59         }
60     }
61
62     //TODO Modifier gestion d'erreur
63     if(isset($_GET["id"])){
64     }
65     ModifierHistoire($idHistoire,$titre,$histoire,$idImage,$categorie);
66     }
67     else
68     {
69         InsererHistoire($titre,$histoire,$idImage,$categorie,$_SESSION["utilisateur"]);
70     }
71     /*if($erreurMessage == "")*/
72     //{
73         header("Location: compte.php");
74         exit();

```

```
75     //)
76
77
78 }
79 /** affiche tout les catégorie dans la base
80 *
81 */
82 function AfficherTouteLesCategorie()
83 {
84     $touteLescategorie = RetournerTouteLesCategories();
85     for ($i = 0; $i < count($touteLescategorie) ; $i++) {
86         echo "<option value="" . $touteLescategorie[$i]["idCategorie"] . "\">" . $touteLescategorie[$i]["nomCategorie"] . "</option>";
87     }
88 }
89
90 /** affiche tout les catégorie dans la base avec un selectioner
91 * @param $id int id de la catégorie
92 */
93 function AfficherTouteLesCategorieAvecUneSelectioner($id)
94 {
95     $touteLescategorie = RetournerTouteLesCategories();
96     for ($i = 0; $i < count($touteLescategorie) ; $i++) {
97         if($id == $touteLescategorie[$i]["idCategorie"])
98         {
99             echo "<option value="" . $touteLescategorie[$i]["idCategorie"] . "\" selected >" .
100             $touteLescategorie[$i]["nomCategorie"] . "</option>";
101         }
102         else
103         {
104             echo "<option value="" . $touteLescategorie[$i]["idCategorie"] . "\">" . $touteLescategorie[$i]["nomCategorie"] . "</option>";
105         }
106     }
107 }
```