

```
1  <?php
2
3  /*
4   * SADEK Serena
5   * Juin 2017
6   * TripTracker
7   * DataInsertModif.php
8   *
9   * Toute les fonctions de cette page sont dédiées à l'insertion des données dans
10  * le serveur et dans la base de donnée. Cette page fait également office de protail
11  * pour les call AJAX commandant l'insertion et la modification d'éléments.
12  */
13
14  session_start();
15
16  //Connexion PDO requise
17  require_once '../connection.php';
18
19  /**
20   * Protocole visant à supprimer un voyage, ses étapes et les média qui leur sont
21   * attribués de la base de donnée. Le protocole supprime également le fichier du
22   * tracé et les images du serveur.
23   */
24  if (isset($_POST["deleteTrip"])) {
25      //Nécessité de récupérer des données en lecture
26      require_once './navigationData.php';
27
28      $idTrip = $_POST["idToDelete"];
29
30      try {
31          $wps = getWps($idTrip);
32
33          if (count($wps > 0)) {
34              $connection = getConnection();
35              $connection->beginTransaction();
36
37              for ($index7 = 0; $index7 < count($wps); $index7++) {
38                  //Suppression des media
39                  removeMediaFile($wps[$index7]["idWaypoint"], $connection);
40                  deleteMediaOfWp($wps[$index7]["idWaypoint"], $connection);
41                  //Suppression de l'étape
42                  deleteWaypoint($wps[$index7]["idWaypoint"], $connection);
43              }
44
45              //Suppression du voyage
46              removePathOfTrip($idTrip, $connection);
47              deleteTrip($idTrip, $connection);
48
49              $connection->commit();
50          }
51      } catch (Exception $ex) {
52          $connection->rollBack();
53          echo json_encode($ex->getTraceAsString());
54      }
55  }
56
57
58  /**
59   * Procédure déclenchée par la présence de la variable insert dans le post
60   * Récupère le path et les informations pour les insérer dans la base de donnée
61   * l'aide d'une transaction.
62   */
63  if (isset($_POST["insert"])) {
64      try {
65          $ids = array();
66
67          $connection = getConnection();
68          $connection->beginTransaction();
69  }
```

```

70     $title = filter_input(INPUT_POST, "title", FILTER_SANITIZE_STRING);
71
72     $stripId = InsertTrip($title, $connection);
73     $fileId = CreatePathTextFile($_POST["path"]);
74     setPath($stripId, $fileId, $connection);
75
76     $path = filter_input(INPUT_POST, "path", FILTER_SANITIZE_STRING);
77
78     $content = $_POST["content"];
79     $content = json_decode($content);
80
81     for ($index = 0; $index < count($content); $index++) {
82
83         $title = $content[$index]->title;
84         $comment = $content[$index]->comment;
85         $date = $content[$index]->date;
86         $date = implode("-", array_reverse(explode("/", $date)));
87
88         $lat = $content[$index]->lat;
89         $lng = $content[$index]->lng;
90         $address = $content[$index]->address;
91
92         $wpId = InsertWaypoint($stripId, $title, $comment, $date, $lat, $lng,
93         $address, $connection);
94         //Cookie destiné au call AJAX d'insertion des media
95         setcookie("WP" . $content[$index]->ref, $wpId, time() + 10);
96
97         array_push($ids, $wpId);
98     }
99
100    $connection->commit();
101
102    echo json_encode($ids);
103 } catch (Exception $ex) {
104     $connection->rollBack();
105     if (isset($fileId)) {
106         unlink("../usersRessources/path/" . $fileId);
107     }
108
109     echo json_encode($ex->getTraceAsString());
110 }
111
112 /**
113  * Protocole visant à éditer un voyage existant. On compare les informations
114  * soumises aux informations existantes, et selon le cas on les met à jour,
115  * on les insère ou on les supprime.
116  */
117 if (isset($_POST["edit"])) {
118     require '../navigationData.php';
119
120     //Réucpération du contenu initial du trip
121
122     $data = array();
123     $data["trip"] = getTrip($_POST["tripId"]);
124
125     if (count($data["trip"]) > 0) {
126         $data["waypoints"] = getWps($_POST["tripId"]);
127
128         for ($index1 = 0; $index1 < count($data["waypoints"]); $index1++) {
129             $media = getMediaOfWp($data["waypoints"][$index1]["idWaypoint"]);
130             $mediaTitle = array();
131             for ($index2 = 0; $index2 < count($media); $index2++) {
132                 $filename = "usersRessources/image/" . $media[$index2]["mediaName"];
133                 array_push($mediaTitle, $filename);
134             }
135             $data["waypoints"][$index1]["wpDate"] = implode("/",
136                 array_reverse(explode("-", $data["waypoints"][$index1]["wpDate"])));

```

```

137         $data["waypoints"][$index1]["media"] = $mediaTitle;
138     }
139
140     //Modification contextuelle du contenu
141
142     try {
143         $connection = getConnection();
144         $connection->beginTransaction();
145
146         $idTrip = $_POST["tripId"];
147         $title = $_POST["title"];
148         $path = $_POST["path"];
149
150         //Mise à jour du voyage
151         updateTrip($idTrip, $title, $connection);
152         updatePathTextFile(getPathName($idTrip, $connection), $path);
153
154         $content = json_decode($_POST["content"]);
155         $contentCopy = json_decode($_POST["content"]);
156
157         for ($index3 = 0; $index3 < count($content); $index3++) {
158
159             $title = $content[$index3]->title;
160             $date = $content[$index3]->date;
161             $date = implode("-", array_reverse(explode("/", $date)));
162             $comment = $content[$index3]->comment;
163             $lat = $content[$index3]->lat;
164             $lng = $content[$index3]->lng;
165             $address = $content[$index3]->address;
166
167             //Les étapes possédant un id (présentes dans la base) sont mises à
            jour
168             if (isset($content[$index3]->id)) {
169
170                 $idWp = $content[$index3]->id;
171                 updateWaypoint($idWp, $idTrip, $title, $comment, $date, $lat,
                    $lng, $address, $connection);
172                 //Cookie destiné au call AJAX d'insertion des media
173                 setcookie("WP" . $content[$index3]->ref, $idWp, time() + 10);
174
175                 for ($index4 = 0; $index4 < count($data["waypoints"]);
                    $index4++) {
176                     if (isset($data["waypoints"][$index4]["idWaypoint"])) {
177                         if ($data["waypoints"][$index4]["idWaypoint"] == $idWp) {
178                             $data["waypoints"][$index4] = null;
179                         }
180                     }
181                 }
182             } else {
183                 //Les étapes ne possédant pas d'id sont nouvelles et donc
                    insérées dans la base
184                 InsertWaypoint($idTrip, $title, $comment, $date, $lat, $lng,
                    $address, $connection);
185             }
186         }
187
188         //Les étapes de la base absentes des étapes mises à jour sont supprimées
189         for ($index5 = 0; $index5 < count($data["waypoints"]); $index5++) {
190             if ($data["waypoints"][$index5] != null) {
191                 $wpid = ($data["waypoints"][$index5]["idWaypoint"]);
192                 removeMediaFile($wpid, $connection);
193                 deleteMediaOfWp($wpid, $connection);
194                 deleteWaypoint($wpid, $connection);
195             }
196         }
197
198         //Les images dans la fille d'attente de suppression sont supprimées.
199         for ($index6 = 0; $index6 < count($_SESSION["picOnDelete"]); $index6++) {
200             removeMedia($_SESSION["picOnDelete"][$index6], $connection);

```

```

201         deleteMedia($_SESSION["picOnDelete"][$index6], $connection);
202     }
203
204     $_SESSION["picOnDelete"] = array();
205
206     $connection->commit();
207
208     echo 'OK';
209 } catch (Exception $exc) {
210     $connection->rollBack();
211
212     echo $exc->getMessage();
213 }
214 } else {
215     echo 'No Result';
216 }
217 }
218
219 /**
220  * Ajoute un voyage dans la base de donnée avec les informations données par le
221  * biais de la connexion donnée
222  * @param type $title
223  * @param type $co
224  * @return type
225  */
226 function InsertTrip($title, $co) {
227     $req = $co->prepare("INSERT INTO trip(idUser, tpTitle) values (:idUser, :title)");
228     $req->bindParam(":idUser", $_SESSION["idUser"], PDO::PARAM_INT);
229     $req->bindParam(":title", $title, PDO::PARAM_STR);
230     $req->execute();
231     return $co->lastInsertId();
232 }
233
234 /**
235  * Attribue le nom du fichier contenant son tracé au voyage donné par le biais
236  * de la connexion donnée
237  * @param type $idTrip
238  * @param type $pathLocation
239  * @param type $co
240  */
241 function setPath($idTrip, $pathLocation, $co) {
242     $req = $co->prepare("UPDATE trip set pathObject = :path WHERE idTrip = :idTrip
243     AND idUser = :idUser");
244     $req->bindParam(":path", $pathLocation, PDO::PARAM_STR);
245     $req->bindParam(":idTrip", $idTrip, PDO::PARAM_INT);
246     $req->bindParam(":idUser", $_SESSION["idUser"], PDO::PARAM_INT);
247     $req->execute();
248 }
249
250 /**
251  * Ajoute une étape par le biais de la connexion donnée
252  * @param int $idTrip
253  * @param string $title
254  * @param string $comment
255  * @param string $date
256  * @param double $lat
257  * @param double $lng
258  * @param string $address
259  * @param PDO $co
260  * @return int $idWp : id du Waypoint ajouté
261  */
262 function InsertWaypoint($idTrip, $title, $comment, $date, $lat, $lng, $address, $co) {
263     $req = $co->prepare("INSERT INTO waypoint(idTrip, wpTitle, wpComment, wpDate,
264     lat, lng, address) values (:idTrip, :title, :comment, :date, :lat, :lng,
265     :address)");
266     $req->bindParam(":idTrip", $idTrip, PDO::PARAM_INT);
267     $req->bindParam(":title", $title, PDO::PARAM_STR);
268     $req->bindParam(":comment", $comment, PDO::PARAM_STR);
269     $req->bindParam(":date", $date, PDO::PARAM_STR);

```

```

267     $req->bindParam(":lat", $lat, PDO::PARAM_STR);
268     $req->bindParam(":lng", $lng, PDO::PARAM_STR);
269     $req->bindParam(":address", $address, PDO::PARAM_STR);
270     $req->execute();
271     return $co->lastInsertId();
272 }
273
274 /**
275  * Génère un fichier au nom aléatoire pour stocker la variable passée en paramètre
276  * @param string $content : contenu à insérer dans un fichier texte
277  * @return string $id : shortName du fichier texte créé
278  */
279 function CreatePathTextFile($content) {
280     $id = uniqid("path", true);
281     $id .= ".txt";
282
283     $stream = fopen("../usersRessources/path/" . $id, 'a+');
284     fwrite($stream, $content);
285     fclose($stream);
286
287     return $id;
288 }
289
290 /**
291  * Ré-écrit le chemin du voyage dans le fichier donné
292  * @param type $name
293  * @param type $content
294  */
295 function updatePathTextFile($name, $content) {
296     $stream = fopen("../usersRessources/path/" . $name, 'w');
297     fwrite($stream, $content);
298     fclose($stream);
299 }
300
301 /**
302  * Efface le voyage donné se sa table par le biais de la connexion donnée
303  * @param type $tripId
304  */
305 function deleteTrip($tripId, $co) {
306     $req = $co->prepare("DELETE FROM trip where idTrip = :id and idUser = :idUser");
307     $req->bindParam(":id", $tripId, PDO::PARAM_INT);
308     $req->bindParam(":idUser", $_SESSION["idUser"], PDO::PARAM_INT);
309     $req->execute();
310 }
311
312 /**
313  * Efface l'étape de sa table par le biais de la connexion donnée
314  * @param type $wpId
315  * @param type $co
316  */
317 function deleteWaypoint($wpId, $co) {
318     $req = $co->prepare("DELETE FROM waypoint where idWaypoint = :id");
319     $req->bindParam(":id", $wpId, PDO::PARAM_INT);
320     $req->execute();
321 }
322
323 /**
324  * Efface les références des Média du voyage donné dans la table par le biais
325  * de la connexion donnée
326  * @param type $wpId
327  * @param type $co
328  */
329 function deleteMediaOfWp($wpId, $co) {
330     $req = $co->prepare("DELETE FROM media where idWaypoint = :id");
331     $req->bindParam(":id", $wpId, PDO::PARAM_INT);
332     $req->execute();
333 }
334
335 /**

```

```

336  * Efface les référénces du média donné par le biais de la connexion donnée
337  * @param type $wpId
338  * @param type $co
339  */
340  function deleteMedia($idMedia, $co) {
341      $req = $co->prepare("DELETE FROM media where idMedia = :id");
342      $req->bindParam(":id", $idMedia, PDO::PARAM_INT);
343      $req->execute();
344  }
345
346  /**
347   * Efface le média du serveur
348   * @param type $wpId
349   * @param type $co
350   */
351  function removeMedia($idMedia, $co) {
352      $name = getMediaName($idMedia, $co);
353      if (file_exists("../usersRessources/image/" . $name)) {
354          unlink("../usersRessources/image/" . $name);
355      }
356  }
357
358  /**
359   * Effece la tracé du serveur
360   * @param type $idTrip
361   * @param type $co
362   */
363  function removePathOfTrip($idTrip, $co) {
364      $pathName = getPathName($idTrip, $co);
365      unlink("../usersRessources/path/" . $pathName);
366  }
367
368  /**
369   * Récupère le nom du média à partir de son id par le biais de la connexion donnée
370   * @param type $wpId
371   * @param type $co
372   */
373  function getMediaName($mediaId, $co) {
374      $req = $co->prepare("SELECT mediaName FROM media where idMedia = :id");
375      $req->bindParam(":id", $mediaId, PDO::PARAM_INT);
376      $req->execute();
377      $result = $req->fetch(PDO::FETCH_ASSOC);
378      return $result["mediaName"];
379  }
380
381  /**
382   * Efface les images de l'étape donnée par le biais de la connexion donnée
383   * @param type $wpId
384   */
385  function removeMediaFile($wpId, $co) {
386      $media = getMediaOfWp($wpId);
387      for ($index = 0; $index < count($media); $index++) {
388          if (file_exists("../usersRessources/image/" . $media[$index]["mediaName"])) {
389              unlink("../usersRessources/image/" . $media[$index]["mediaName"]);
390          }
391      }
392  }
393
394  /**
395   * Met à jour le voyage donnée par le biais de la connexion donnée
396   * @param type $idTrip
397   * @param type $title
398   * @param type $co
399   */
400  function updateTrip($idTrip, $title, $co) {
401      try {
402
403          $req = $co->prepare("UPDATE trip set tpTitle = :title where idTrip = :id");
404          $req->bindParam(":id", $idTrip, PDO::PARAM_INT);

```

```

405         $req->bindParam(":title", $title, PDO::PARAM_STR);
406         $req->execute();
407     } catch (Exception $ex) {
408         echo $ex->getMessage();
409     }
410 }
411
412 /**
413  * Met à jour l'étape donnée par le biais de la connexion donnée
414  * @param type $wpId
415  * @param type $idTrip
416  * @param type $title
417  * @param type $comment
418  * @param type $date
419  * @param type $lat
420  * @param type $lng
421  * @param type $address
422  * @param type $co
423  */
424 function updateWaypoint($wpId, $idTrip, $title, $comment, $date, $lat, $lng,
425 $address, $co) {
426     try {
427         $req = $co->prepare("UPDATE waypoint set wpTitle = :title, wpComment =
428         :comment, wpDate = :date, lat = :lat, lng = :lng, address = :address where
429         idWaypoint = :wpId");
430         $req->bindParam(":wpId", $wpId, PDO::PARAM_INT);
431         $req->bindParam(":title", $title, PDO::PARAM_STR);
432         $req->bindParam(":comment", $comment, PDO::PARAM_STR);
433         $req->bindParam(":date", $date, PDO::PARAM_STR);
434         $req->bindParam(":lat", $lat, PDO::PARAM_STR);
435         $req->bindParam(":lng", $lng, PDO::PARAM_STR);
436         $req->bindParam(":address", $address, PDO::PARAM_STR);
437         $req->execute();
438     } catch (Exception $ex) {
439         echo $ex->getMessage();
440     }
441 }
442
443 /**
444  * Retounre le nom du fichier contenant le tracé do voyage donné
445  * @param type $idTrip
446  * @param type $co
447  * @return type
448  */
449 function getPathName($idTrip, $co) {
450     $req = $co->prepare("SELECT pathObject FROM trip WHERE idTrip = :idTrip");
451     $req->bindParam(":idTrip", $idTrip, PDO::PARAM_INT);
452     $req->execute();
453     $result = $req->fetch(PDO::FETCH_ASSOC);
454     return $result["pathObject"];
455 }

```