

```

1  <?php
2  /*****
3  AUTEUR      : Constantin Herrmann
4  LIEU        : CFPT Informatique Genève
5  DATE        : 14.06.2018
6
7  TITRE PROJET: KidsGames Geneva Score
8
9  TITRE PAGE  : staff.c
10 DESCRIPTION : Ce script contient toutes les fonctions concernant le staff
11 VERSION     : 1.0
12 *****/
13
14 /**
15  * Récupère un coach en fonction de son id
16  * Si le coach ou l'id n'existe pas alors le return sera égal à FALSE
17  *
18  * @param string id du membre du staff
19  * @return array un tableau avec le membre sélectionné par son id
20  *      ['id'] -> l'id du membre
21  *      ['nom'] -> le nom
22  *      ['prenom'] -> le prénom
23  *      ['age'] -> l'âge
24  *      ['idRole'] -> l'id de son Rôle
25  *      ['phone'] -> le numéro de téléphone
26  */
27 function GetStaffById($id){
28     static $query = null;
29     if ($query == null) {
30         $req = "SELECT `id`, `nom`, `prenom`, `age`, `idRole`, `phone` FROM `Staff`
31         • WHERE `id` = :id";
32     }
33     $query = connecteur()->prepare($req);
34     try {
35         $query->bindParam(':id', $id, PDO::PARAM_STR);
36         $query->execute();
37         $res = $query->fetch(PDO::FETCH_ASSOC);
38     } catch (Exception $e) {
39         error_log($e->getMessage());
40         $res = false;
41     }
42     return $res;
43 }
44
45 /**
46  * Récupère tout les membres du staff
47  *
48  * @return array un tableau avec tout les membres
49  *      [index]
50  *      ['id'] -> l'id du membre
51  *      ['nom'] -> le nom
52  *      ['prenom'] -> le prénom
53  *      ['age'] -> l'âge
54  *      ['idRole'] -> l'id de son Rôle
55  *      ['phone'] -> le numéro de téléphone
56  */
57 function GetAllStaff(){

```

```

58     static $query = null;
59     if ($query == null) {
60         $req = "SELECT `id`, `nom`, `prenom`, `age`, `idRole`, `phone` FROM `Staff`";
61         $query = connecteur()->prepare($req);
62     }
63     try {
64         $query->execute();
65         $res = $query->fetchAll(PDO::FETCH_ASSOC);
66     }
67     catch (Exception $e) {
68         error_log($e->getMessage());
69         $res = false;
70     }
71     return $res;
72 }
73
74 /**
75  * Ajoute un membre à la table staff
76  *
77  * @param string le nom
78  * @param string le prénom
79  * @param string id du Rôle
80  */
81 function AddStaff($nom, $prenom, $idRole){
82     static $query = null;
83
84     if ($query == null) {
85         $req = "INSERT INTO `Staff`(`nom`, `prenom`, `idRole`) VALUES (:nom, :prenom,
      •      :idRole)";
86         $query = connecteur()->prepare($req);
87     }
88     try {
89         $query->bindParam(':nom', $nom, PDO::PARAM_STR);
90         $query->bindParam(':prenom', $prenom, PDO::PARAM_STR);
91         $query->bindParam(':idRole', $idRole, PDO::PARAM_STR);
92         $query->execute();
93         $query->fetch();
94     }
95     catch (Exception $e) {
96         error_log($e->getMessage());
97     }
98 }
99
100 /**
101  * Met à jour un membre du staff
102  *
103  * @param string id du membre à modifier
104  * @param string le nom
105  * @param string le prénom
106  * @param string id du Rôle
107  */
108 function UpdateStaff($id, $nom, $prenom, $idRole){
109     static $query = null;
110
111     if ($query == null) {
112         $req = "UPDATE `Staff` SET `nom`= :nom, `prenom`= :prenom, `idRole`= :role
      •      WHERE `id` = :id";
113         $query = connecteur()->prepare($req);
114     }

```

```

114     }
115     try {
116         $query->bindParam(':id', $id, PDO::PARAM_STR);
117         $query->bindParam(':nom', $nom, PDO::PARAM_STR);
118         $query->bindParam(':prenom', $prenom, PDO::PARAM_STR);
119         $query->bindParam(':role', $idRole, PDO::PARAM_STR);
120         $query->execute();
121         $query->fetch();
122     }
123     catch (Exception $e) {
124         error_log($e->getMessage());
125     }
126 }
127
128 /**
129  * Supprime un membre du staff
130  * Supprime également la liaison entre luo et son équipe si il à un équipe
131  *
132  * @param string id du membre à supprimer
133  */
134 function DeleteStaff($id){
135
136     // d'abord on supprime la liason
137     RemoveCoachFromHisTeam($id);
138     static $query = null;
139
140     if ($query == null) {
141         $req = "DELETE FROM `Staff` WHERE `id` = :id";
142         $query = connecteur()->prepare($req);
143     }
144     try {
145         $query->bindParam(':id', $id, PDO::PARAM_STR);
146         $query->execute();
147         $query->fetch();
148     }
149     catch (Exception $e) {
150         error_log($e->getMessage());
151     }
152 }
153
154 /**
155  * Retourne tout les coaches qui n'ont pas d'équipe
156  *
157  * @return array un tableau avec tout les membres
158  *      [index]
159  *      ['id'] -> l'id du membre
160  *      ['nom'] -> le nom
161  *      ['prenom'] -> le prénom
162  */
163 function GetAllCoachsWithoutTeam(){
164     static $query = null;
165     if ($query == null) {
166         $req = "SELECT `id`, `nom`, `prenom` FROM `Staff` WHERE `idRole` = 2 AND `id`
167         • NOT IN (SELECT idCoach FROM TEAMS WHERE id != 0) ORDER BY `prenom`";
168         $query = connecteur()->prepare($req);
169     }
170     try {
171         $query->execute();
172         $res = $query->fetchAll(PDO::FETCH_ASSOC);

```

```

171     $res = $query->fetchAll(PDO::FETCH_ASSOC),
172 }
173 catch (Exception $e) {
174     error_log($e->getMessage());
175     $res = false;
176 }
177 return $res;
178 }
179
180 /**
181  * Met à -1 l'équipe avec l'id du coach indiqué
182  *
183  * @param string id du coach
184  */
185 function RemoveCoachFromHisTeam($idCoach){
186     if ($query == null) {
187         $req = "UPDATE `Teams` SET `idCoach` = -1 WHERE `idCoach` = :idCoach";
188         $query = connecteur()->prepare($req);
189     }
190     try {
191         $query->bindParam(':idCoach', $idCoach, PDO::PARAM_STR);
192         $query->execute();
193         $query->fetch();
194     }
195     catch (Exception $e) {
196         error_log($e->getMessage());
197     }
198 }
199
200 /**
201  * Retourne tout les arbitres de la table Staff
202  *
203  * @return array un tableau avec tout les arbitres
204  *     [index]
205  *     ['id'] -> l'id du membre
206  *     ['nom'] -> le nom
207  *     ['prenom'] -> le prénom
208  */
209 function GetArbitres(){
210     static $query = null;
211     if ($query == null) {
212         $req = "SELECT `id`, `nom`, `prenom` FROM `Staff` WHERE `idRole` = 3 ORDER BY
        • `prenom`";
213         $query = connecteur()->prepare($req);
214     }
215     try {
216         $query->execute();
217         $res = $query->fetchAll(PDO::FETCH_ASSOC);
218     }
219     catch (Exception $e) {
220         error_log($e->getMessage());
221         $res = false;
222     }
223     return $res;
224 }
225
226 /**
227  * Verifie si un membre es admin
228  *

```

```

229 * @param string id du membre
230 * @return boolean true si il est admin, false si non
231 */
232 function IsAdmin($id){
233     static $query = null;
234
235     if ($query == null) {
236         $req = "SELECT `Staff`.`prenom` FROM `Staff` WHERE `Staff`.`idRole` = (SELECT
        • `Role`.`id` FROM `Role` WHERE `Role`.`intitule` = 'Admin') AND `Staff`.`id` =
        • :id";
237         $query = connecteur()->prepare($req);
238     }
239     try {
240         $query->bindParam(':id', $id, PDO::PARAM_STR);
241         $query->execute();
242         $res = $query->fetch(PDO::FETCH_ASSOC);
243     }
244     catch (Exception $e) {
245         error_log($e->getMessage());
246         $res = false;
247     }
248
249     if ($res != false)
250         return true;
251     else
252         return false;
253 }
254

```