

OverwatchCollection V.1

Code source

Sven Wikberg

19/06/2017

Table des matières

Page index.php	2
Page des héros	4
Page d'un héros.....	5
Page des événements.....	6
Page d'un événement.....	7
Page des autres objets	8
Page utilisateur (login/sign in et my account).....	9
Page administrateur.....	12
Fonction ajout ou suppression d'objet pour un utilisateur	14
Classe MyPdo	15
Classe Oc_Dao	16
Class Oc_Display.....	27
Fichier de config base de données	38
Style du site web (CSS)	39

Page index.php

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\index.php                                         lundi 19 juin 2017 10:37
1  <!--
2      Auteur: Sven Wikberg
3      Date: 19/06/2017
4      Description: Page d'accueil
5  -->
6  <!doctype html>
7  <?php
8  session_start();
9
10 require_once('class/class.oc_dao.php');
11 require_once('class/class.oc_display.php');
12 ?>
13 <html lang="en">
14     <head>
15         <meta charset="utf-8">
16         <title>OverwatchCollection</title>
17         <link rel="stylesheet" href="css/style-main.css">
18     </head>
19     <body>
20         <header>
21             <?php
22                 OcDisplay::DisplayNavbar();
23             ?>
24         </header>
25         <section id="index">
26             <h1>Overwatch Collection</h1>
27             <div class="flex_row">
28                 <div style="width: 70;">
29                     <p>Bienvenue sur Overwatch Collection! Le site qui vous permet
30                         enfin d'avoir un suivi de vos objets dans Overwatch™.
31                         Vous n'avez pas besoin de créer compte si vous voulez seulement
32                         naviguer parmi les héros et les objets, par contre
33                         si vous voulez pouvoir sélectionner des objets et avoir un
34                         suivi ainsi que des statistiques, vous êtes obligé d'avoir
35                         compte.</p>
36                     <p>Ce site est divisé en plusieurs parties, d'abord il y a la
37                         partie "Heroes", cette partie contient les héros du jeu, pour
38                         chaque
39                         héros il y a un résumé, quelques informations et évidemment la
40                         liste, triée par rareté, des objets du héros en question.
41                         <br>Ensuite, il y a la partie "Events" qui contient les
42                         événements du jeu, comme par exemple Halloween ou Noël, et de la
43                         même manière
44                         que pour les héros, il y a la liste des objets de l'événement.
45                         <br>Après ça il y a la partie "Rewards" qui contient les objets
46                         qui ne sont reliés à aucun héros, comme les icons de joueurs par
47                         exemple.
48                         <br>Et finalement il y a la partie "My Account" qui est
49                         accessible uniquement pour les utilisateurs connectés, qui
50                         affiche
51                         entre autre des statistiques sur les objets sélectionnés et un
52                         classement des utilisateurs.</p>
53                 </div>
54                 <div style="width: 25;">
55                     <h2>Users list</h2>
56                     <?php
57                         OcDisplay::DisplayListUsers();
58                 </div>
59             </div>
60         </section>
61     </body>
62 </html>
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\index.php                                         lundi 19 juin 2017 10:37
46
47          ?>
48      </div>
49  </div>
50  </section>
51  <footer>
52      <p>This site is not affiliated with Blizzard Entertainement. ©2016
53      Blizzard Entertainment, Inc. All rights reserved /
54      Author : Sven Wikberg </p>
55  </footer>
56  </body>
57
58</html>
```

Page des héros

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\heroes.php  
lundi 19 juin 2017 10:45  
1  <!--  
2      Auteur: Sven Wikberg  
3      Date: 19/06/2017  
4      Description: Page des heros  
5  -->  
6  <!doctype html>  
7  <?php  
8  session_start();  
9  
10 require_once ('class/class.oc_dao.php');  
11 require_once ('class/class.oc_display.php');  
12 ?>  
13 <html lang="en">  
14     <head>  
15         <meta charset="utf-8">  
16         <title>OverwatchCollection</title>  
17         <link rel="stylesheet" href="css/style-main.css">  
18     </head>  
19     <body>  
20         <header>  
21             <?php  
22                 OcDisplay::DisplayNavbar ();  
23             ?>  
24         </header>  
25         <section id="heroes">  
26             <?php  
27                 OcDisplay::DisplayHeroesByRole (4);  
28             ?>  
29         </section>  
30         <footer>  
31             <p>This site is not affiliated with Blizzard Entertainement. ©2016  
32                 Blizzard Entertainment, Inc. All rights reserved /  
33                 Author : Sven Wikberg </p>  
34         </footer>  
35     </body>  
36 </html>
```

Page d'un héros

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\hero.php  
lundi 19 juin 2017 10:48  
1 <!--  
2     Auteur: Sven Wikberg  
3     Date: 19/06/2017  
4     Description: Page d'un heros  
5 -->  
6 <!doctype html>  
7 <?php  
8 session_start();  
9  
10 require_once('class/class.oc_dao.php');  
11 require_once('class/class.oc_display.php');  
12 require_once('function/func.user_reward.php');  
13 ?>  
14 <html lang="en">  
15     <head>  
16         <meta charset="utf-8">  
17         <title>OverwatchCollection</title>  
18         <link rel="stylesheet" href="css/style-main.css">  
19     </head>  
20     <body>  
21         <header>  
22             <?php  
23                 OcDisplay::DisplayNavbar();  
24             ?>  
25         </header>  
26         <section id="hero_info">  
27             <?php  
28                 OcDisplay::DisplayHeroInfo($_GET['id']);  
29             ?>  
30         </section>  
31         <section id="hero_abilities">  
32             <?php  
33                 OcDisplay::DisplayHeroAbilities($_GET['id']);  
34             ?>  
35         </section>  
36         <section id="rewards">  
37             <?php  
38                 OcDisplay::DisplayHeroRewards($_GET['id']);  
39             ?>  
40         </section>  
41         <footer>  
42             <p>This site is not affiliated with Blizzard Entertainment. ©2016  
43                 Blizzard Entertainment, Inc. All rights reserved /  
44                 Author : Sven Wikberg </p>  
45         </footer>  
46     </body>  
47 </html>
```

Page des événements

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\events.php  
lundi 19 juin 2017 10:49  
1  <!--  
2      Auteur: Sven Wikberg  
3      Date: 19/06/2017  
4      Description: Page des evenements  
5  -->  
6  <!doctype html>  
7  <?php  
8  session_start();  
9  
10 require_once ('class/class.oc_dao.php');  
11 require_once ('class/class.oc_display.php');  
12 ?>  
13 <html lang="en">  
14     <head>  
15         <meta charset="utf-8">  
16         <title>OverwatchCollection</title>  
17         <link rel="stylesheet" href="css/style-main.css">  
18     </head>  
19     <body>  
20         <header>  
21             <?php  
22                 OcDisplay::DisplayNavbar ();  
23             ?>  
24         </header>  
25         <section id="events">  
26             <?php  
27                 OcDisplay::DisplayEvents (2);  
28             ?>  
29         </section>  
30         <footer>  
31             <p>This site is not affiliated with Blizzard Entertainement. ©2016  
32                 Blizzard Entertainment, Inc. All rights reserved /  
33                 Author : Sven Wikberg </p>  
34         </footer>  
35     </body>  
36 </html>
```

Page d'un événement

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\event.php  
lundi 19 juin 2017 10:50  
1  <!--  
2      Auteur: Sven Wikberg  
3      Date: 19/06/2017  
4      Description: Page d'un evenement  
5  -->  
6  <!doctype html>  
7  <?php  
8  session_start();  
9  
10 require_once('class/class.oc_dao.php');  
11 require_once('class/class.oc_display.php');  
12 require_once('function/func.user_reward.php');  
13  
14 ?>  
15 <html lang="en">  
16     <head>  
17         <meta charset="utf-8">  
18         <title>OverwatchCollection</title>  
19         <link rel="stylesheet" href="css/style-main.css">  
20     </head>  
21     <body>  
22         <header>  
23             <?php  
24                 OcDisplay::DisplayNavbar();  
25             ?>  
26         </header>  
27         <section id="event_info">  
28             <?php  
29                 OcDisplay::DisplayEventInfo($_GET['id']);  
30             ?>  
31         </section>  
32         <section id="rewards">  
33             <?php  
34                 OcDisplay::DisplayEventRewards($_GET['id']);  
35             ?>  
36         </section>  
37         <footer>  
38             <p>This site is not affiliated with Blizzard Entertainement. ©2016  
39                 Blizzard Entertainment, Inc. All rights reserved /  
40                 Author : Sven Wikberg </p>  
41         </footer>  
42     </body>  
43 </html>
```

Page des autres objets

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\rewards.php  
lundi 19 juin 2017 10:51  
1  <!--  
2      Auteur: Sven Wikberg  
3      Date: 19/06/2017  
4      Description: Page des autres objets cosmétiques  
5  -->  
6  <!doctype html>  
7  <?php  
8  session_start();  
9  
10 require_once('class/class.oc_dao.php');  
11 require_once('class/class.oc_display.php');  
12 require_once('function/func.user_reward.php');  
13 ?>  
14 <html lang="en">  
15     <head>  
16         <meta charset="utf-8">  
17         <title>OverwatchCollection</title>  
18         <link rel="stylesheet" href="css/style-main.css">  
19     </head>  
20     <body>  
21         <header>  
22             <?php  
23                 OcDisplay::DisplayNavbar();  
24             ?>  
25         </header>  
26         <section id="others_rewards_info">  
27             <h1>Others Rewards</h1>  
28             <p>Certains des objets cosmétiques dans Overwatch n'ont pas d'héros  
29                 associé, par exemple les icônes d'utilisateurs, étant donnée qu'elle  
                  sont faites pour le compte et pas un héros spécifique. C'est sur cette  
                  page qu'elle seront répertoriées, triées par catégories et par raretés.  
              </p>  
30         </section>  
31         <section id="rewards">  
32             <?php  
33                 OcDisplay::DisplayOtherRewards();  
34             ?>  
35         </section>  
36         <footer>  
37             <p>This site is not affiliated with Blizzard Entertainment. ©2016  
38                 Blizzard Entertainment, Inc. All rights reserved /  
39                 Author : Sven Wikberg </p>  
40         </footer>  
41     </body>  
42 </html>
```

Page utilisateur (login/sign in et my account)

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\user.php
lundi 19 juin 2017 10:53

1  <!--
2   Auteur: Sven Wikberg
3   Date: 19/06/2017
4   Description: Page utilisateur
5 -->
6 <!doctype html>
7 <?php
8 session_start();
9
10 require_once('class/class.oc_dao.php');
11 require_once('class/class.oc_display.php');
12
13 if (isset($_GET['action'])) { // selon l'action, la page recupere, teste ou process
des données différentes
14     $myget = '';
15     if($_GET['action'] == 'login'){ // l'action login teste les données entrées par
l'utilisateur afin de le connecter ou pas
16         if (isset($_POST['username']) && isset($_POST['password'])) {
17             $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_ENCODED);
18             $password = sha1(filter_input(INPUT_POST, 'password',
FILTER_SANITIZE_ENCODED));
19
20             $user = OcDao::SelectUserByUsername($username); // on récupere les
données de l'utilisateur grace a son nom d'utilisateur
21
22             if($user == null) // si l'on ne récupère rien, ca veut dire que ce nom
d'utilisateur n'existe pas, donc on informe l'utilisateur grace a
l'erreur en GET
23                 $myGet = '?msg=wrongUn';
24             elseif ($user['password'] == $password) { // si tout est juste
25                 if($user['is_banned'] == 1) { // si l'utilisateur est banni
26                     $myGet = '?msg=banned';
27                 } else{ // si l'utilisateur n'est pas banni, on le connecte
28                     $_SESSION['id_connected'] = $user['id_user'];
29                 }
30             }
31             else { // si le mot de passe est faux on met l'erreur en GET afin
d'informer l'utilisateur
32                 $myGet = '?msg=wrongPw';
33             }
34         }
35     } elseif ($_GET['action'] == 'signin') { // l'action sign in ajoute un nouvel
utilisateur la dans la base avec les données entrées
36         if (isset($_POST['username']) && isset($_POST['password']) && isset($_POST[
'email'])) {
37             $username = filter_input(INPUT_POST, 'username', FILTER_SANITIZE_ENCODED);
38             $password = sha1(filter_input(INPUT_POST, 'password',
FILTER_SANITIZE_ENCODED));
39             $email = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_EMAIL);
40
41             $tmp = OcDao::InsertUser($username, $password, $email);
42             if($tmp == null){
43                 $myGet = '?msg=signOk';
44             } else {
45                 if($tmp == 1062){
46                     $myGet = '?msg=duplicate';
47                 }
48             }
49         }
50     }
51 }
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
205
206
207
208
209
210
211
212
213
214
215
216
217
218
219
220
221
222
223
224
225
226
227
228
229
230
231
232
233
234
235
236
237
238
239
240
241
242
243
244
245
246
247
248
249
250
251
252
253
254
255
256
257
258
259
259
260
261
262
263
264
265
266
267
268
269
270
271
272
273
274
275
276
277
278
279
279
280
281
282
283
284
285
286
287
288
289
289
290
291
292
293
294
295
296
297
298
299
299
300
301
302
303
304
305
306
307
308
309
309
310
311
312
313
314
315
316
317
318
319
319
320
321
322
323
324
325
326
327
328
329
329
330
331
332
333
334
335
336
337
338
339
339
340
341
342
343
344
345
346
347
348
349
349
350
351
352
353
354
355
356
357
358
359
359
360
361
362
363
364
365
366
367
368
369
369
370
371
372
373
374
375
376
377
378
379
379
380
381
382
383
384
385
386
387
388
389
389
390
391
392
393
394
395
396
397
398
399
399
400
401
402
403
404
405
406
407
408
409
409
410
411
412
413
414
415
416
417
418
419
419
420
421
422
423
424
425
426
427
428
429
429
430
431
432
433
434
435
436
437
438
439
439
440
441
442
443
444
445
446
447
448
449
449
450
451
452
453
454
455
456
457
458
459
459
460
461
462
463
464
465
466
467
468
469
469
470
471
472
473
474
475
476
477
478
479
479
480
481
482
483
484
485
486
487
488
489
489
490
491
492
493
494
495
496
497
498
499
499
500
501
502
503
504
505
506
507
508
509
509
510
511
512
513
514
515
516
517
518
519
519
520
521
522
523
524
525
526
527
528
529
529
530
531
532
533
534
535
536
537
538
539
539
540
541
542
543
544
545
546
547
548
549
549
550
551
552
553
554
555
556
557
558
559
559
560
561
562
563
564
565
566
567
568
569
569
570
571
572
573
574
575
576
577
578
579
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
738
739
739
740
741
742
743
744
745
746
747
748
749
749
750
751
752
753
754
755
756
757
758
759
759
760
761
762
763
764
765
766
767
768
769
769
770
771
772
773
774
775
776
777
778
779
779
780
781
782
783
784
785
786
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
838
839
839
840
841
842
843
844
845
846
847
848
849
849
850
851
852
853
854
855
856
857
858
859
859
860
861
862
863
864
865
866
867
868
869
869
870
871
872
873
874
875
876
877
878
879
879
880
881
882
883
884
885
886
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
918
919
919
920
921
922
923
924
925
926
927
928
929
929
930
931
932
933
934
935
936
937
938
939
939
940
941
942
943
944
945
946
947
948
949
949
950
951
952
953
954
955
956
957
958
959
959
960
961
962
963
964
965
966
967
968
969
969
970
971
972
973
974
975
976
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1027
1028
1029
1029
1030
1031
1032
1033
1034
1035
1036
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1057
1058
1059
1059
1060
1061
1062
1063
1064
1065
1066
1067
1068
1069
1069
1070
1071
1072
1073
1074
1075
1076
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1086
1087
1088
1089
1089
1090
1091
1092
1093
1094
1095
1096
1097
1097
1098
1099
1099
1100
1101
1102
1103
1104
1105
1106
1107
1108
1109
1109
1110
1111
1112
1113
1114
1115
1116
1117
1118
1119
1119
1120
1121
1122
1123
1124
1125
1126
1127
1128
1129
1129
1130
1131
1132
1133
1134
1135
1136
1137
1138
1139
1139
1140
1141
1142
1143
1144
1145
1146
1147
1148
1148
1149
1150
1151
1152
1153
1154
1155
1156
1157
1158
1159
1159
1160
1161
1162
1163
1164
1165
1166
1167
1168
1169
1169
1170
1171
1172
1173
1174
1175
1176
1177
1178
1178
1179
1180
1181
1182
1183
1184
1185
1186
1187
1187
1188
1189
1189
1190
1191
1192
1193
1194
1195
1195
1196
1197
1198
1199
1199
1200
1201
1202
1203
1204
1205
1206
1207
1208
1209
1209
1210
1211
1212
1213
1214
1215
1216
1217
1218
1218
1219
1220
1221
1222
1223
1224
1225
1226
1227
1228
1229
1229
1230
1231
1232
1233
1234
1235
1236
1237
1238
1239
1239
1240
1241
1242
1243
1244
1245
1246
1247
1248
1248
1249
1250
1251
1252
1253
1254
1255
1256
1257
1258
1259
1259
1260
1261
1262
1263
1264
1265
1266
1267
1268
1269
1269
1270
1271
1272
1273
1274
1275
1276
1277
1278
1278
1279
1280
1281
1282
1283
1284
1285
1286
1287
1287
1288
1289
1289
1290
1291
1292
1293
1294
1295
1296
1297
1297
1298
1299
1299
1300
1301
1302
1303
1304
1305
1306
1307
1308
1309
1309
1310
1311
1312
1313
1314
1315
1316
1317
1318
1318
1319
1320
1321
1322
1323
1324
1325
1326
1327
1328
1329
1329
1330
1331
1332
1333
1334
1335
1336
1337
1338
1338
1339
1340
1341
1342
1343
1344
1345
1346
1347
1348
1348
1349
1350
1351
1352
1353
1354
1355
1356
1357
1358
1359
1359
1360
1361
1362
1363
1364
1365
1366
1367
1368
1368
1369
1370
1371
1372
1373
1374
1375
1376
1377
1378
1378
1379
1380
1381
1382
1383
1384
1385
1386
1387
1387
1388
1389
1389
1390
1391
1392
1393
1394
1395
1396
1396
1397
1398
1399
1399
1400
1401
1402
1403
1404
1405
1406
1407
1408
1409
1409
1410
1411
1412
1413
1414
1415
1416
1417
1418
1418
1419
1420
1421
1422
1423
1424
1425
1426
1427
1428
1428
1429
1430
1431
1432
1433
1434
1435
1436
1437
1438
1438
1439
1440
1441
1442
1443
1444
1445
1446
1447
1448
1448
1449
1450
1451
1452
1453
1454
1455
1456
1457
1458
1459
1459
1460
1461
1462
1463
1464
1465
1466
1467
1468
1468
1469
1470
1471
1472
1473
1474
1475
1476
1477
1478
1478
1479
1480
1481
1482
1483
1484
1485
1486
1487
1488
1488
1489
1490
1491
1492
1493
1494
1495
1496
1497
1497
1498
1499
1499
1500
1501
1502
1503
1504
1505
1506
1507
1508
1509
1509
1510
1511
1512
1513
1514
1515
1516
1517
1518
1518
1519
1520
1521
1522
1523
1524
1525
1526
1527
1528
1528
1529
1530
1531
1532
1533
1534
1535
1536
1537
1538
1538
1539
1540
1541
1542
1543
1544
1545
1546
1547
1548
1548
1549
1550
1551
1552
1553
1554
1555
1556
1557
1558
1559
1559
1560
1561
1562
1563
1564
1565
1566
1567
1568
1568
1569
1570
1571
1572
1573
1574
1575
1576
1577
1578
1578
1579
1580
1581
1582
1583
1584
1585
1586
1587
1588
1588
1589
1590
1591
1592
1593
1594
1595
1596
1597
1597
1598
1599
1599
1600
1601
1602
1603
1604
1605
1606
1607
1608
1609
1609
1610
1611
1612
1613
1614
1615
1616
1617
1618
1618
1619
1620
1621
1622
1623
1624
1625
1626
1627
1628
1628
1629
1630
1631
1632
1633
1634
1635
1636
1637
1638
1638
1639
1640
1641
1642
1643
1644
1645
1646
1647
1648
1648
1649
1650
1651
1652
1653
1654
1655
1656
1657
1658
1659
1659
1660
1661
1662
1663
1664
1665
1666
1667
1668
1668
1669
1670
1671
1672
1673
1674
1675
1676
1677
1678
1678
1679
1680
1681
1682
1683
1684
1685
1686
1687
1688
1688
1689
1690
1691
1692
1693
1694
1695
1696
1697
1697
1698
1699
1699
1700
1701
1702
1703
1704
1705
1706
1707
1708
1709
1709
1710
1711
1712
1713
1714
1715
1716
1717
1718
1718
1719
1720
1721
1722
1723
1724
1725
1726
1727
1728
1728
1729
1730
1731
1732
1733
1734
1735
1736
1737
1738
1738
1739
1740
1741
1742
1743
1744
1745
1746
1747
1748
1748
1749
1750
1751
1752
1753
1754
1755
1756
1757
1758
1759
1759
1760
1761
1762
1763
1764
1765
1766
1767
1768
1768
1769
1770
1771
1772
1773
1774
1775
1776
1777
1778
1778
1779
1780
1781
1782
1783
1784
1785
1786
1787
1788
1788
1789
1790
1791
1792
1793
1794
1795
1796
1797
1797
1798
1799
1799
1800
1801
1802
1803
1804
1805
1806
1807
1808
1809
1809
1810
1811
1812
1813
1814
1815
1816
1817
1818
1818
1819
1820
1821
1822
1823
1824
1825
1826
1827
1828
1828
1829
1830
1831
1832
1833
1834
1835
1836
1837
1838
1838
1839
1840
1841
1842
1843
1844
1845
1846
1847
1848
1848
1849
1850
1851
1852
1853
1854
1855
1856
1857
1858
1859
1859
1860
1861
1862
1863
1864
1865
1866
1867
1868
1868
1869
1870
1871
1872
1873
1874
1875
1876
1877
1878
1878
1879
1880
1881
1882
1883
1884
1885
1886
1887
1888
1888
1889
1890
1891
1892
1893
1894
1895
1896
1897
1897
1898
1899
1899
1900
1901
1902
1903
1904
1905
1906
1907
1908
1909
1909
1910
1911
1912
1913
1914
1915
1916
1917
1918
1918
1919
1920
1921
1922
1923
1924
1925
1926
1927
1928
1929
1929
1930
1931
1932
1933
1934
1935
1936
1937
1938
1938
1939
1940
1941
1942
1943
1944
1945
1946
1947
1948
1948
1949
1950
1951
1952
1953
1954
1955
1956
1957
1958
1959
1959
1960
1961
1962
1963
1964
1965
1966
1967
1968
1968
1969
1970
1971
1972
1973
1974
1975
1976
1977
1978
1978
1979
1980
1981
1982
1983
1984
1985
1986
1987
1988
1988
1989
1990
1991
1992
1993
1994
1995
1996
1997
1997
1998
1999
1999
2000
2001
2002
2003
2004
2005
2006
2007
2008
2009
2009
2010
2011
2012
2013
2014
2015
2016
2017
2017
2018
2019
2019
2020
2021
2022
2023
2024
2025
2026
2027
2028
2029
2029
2030
2031
2032
2033
2034
2035
2036
2037
2038
2039
2039
2040
2041
2042
2043
2044
2045
2046
2047
2048
2048
2049
2050
2051
2052
2053
2054
2055
2056
2057
2058
2059
2059
2060
2061
2062
2063
2064
2065
2066
2067
2068
2068
2069
2070
2071
2072
2073
2074
2075
2076
2077
2078
2078
2079
2080
2081
2082
2083
2084
2085
2086
2087
2088
2088
2089
2090
2091
2092
2093
2094
2095
2096
2097
2097
2098
2099
2099
2100
2101
2102
2103
2104
2105
2106
2107
2108
2109
2109
2110
2111
2112
2113
2114
2115
2116
2117
2118
2118
2119
2120
2121
2
```

```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\user.php                                         lundi 19 juin 2017 10:53
48
49
50 } elseif ( $_GET['action'] == 'deco' ) { // l'action deco met la valeur de
l'id_connected a null, id_connected qui donne l'info de l'id de utilisateur
connecté
51     session_destroy();
52 } elseif ( $_GET['action'] == 'update' ) { // l'action update met a jour les
donnée de l'utilisateur avec les données qu'il a entrées
53     if( isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){
// si l'utilisateur est connecter
54         if ( isset($_POST['username']) && isset($_POST['email'])) {
55             $username = filter_input(INPUT_POST, 'username',
FILTER_SANITIZE_ENCODED);
56             $email = filter_input(INPUT_POST, 'email');

57             $tmp = OcDao::UpdateUserByIdNoPwd($_SESSION['id_connected'],
$username, $email);
58             if($tmp == null){
59                 $myGet = '?msg=updateOK';
60             } else {
61                 if($tmp == 1062){
62                     $myGet = '?msg=updateDuplicate';
63                 }
64             }
65         }
66     }
67 }
68 }
69 header('Location: user.php' . $myGet);
70 }
71
72 ?>
73 <html lang="en">
74     <head>
75         <meta charset="utf-8">
76         <title>OverwatchCollection</title>
77         <link rel="stylesheet" href="css/style-main.css">
78     </head>
79     <body>
80         <header>
81             <?php
82                 OcDisplay::DisplayNavbar();
83             ?>
84         </header>
85         <?php
86             if( isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){
// si l'utilisateur est connecter il peut se deconnecter
87                 if(isset($_GET['goto'])){
88                     if($_GET['goto'] == 'updating'){
89                         OcDisplay::DisplayAccountInfoUpdating($_SESSION['id_connected']);
90                     }
91                 } else {
92                     OcDisplay::DisplayAccountInfo($_SESSION['id_connected']);
93
94                     OcDisplay::DisplayAccountStats($_SESSION['id_connected']);
95                 }
96             } else {
97                 if(isset($_GET['msg']))

```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\user.php                                         lundi 19 juin 2017 10:53
98          if($_GET['msg'] == 'signOk') // l'utilisateur a bien ete ajouté a la
99              base de donnees'
100             echo '<h2>Account created</h2>';
101             OcDisplay::DisplayLogin();
102             OcDisplay::DisplaySignin();
103         }
104     ?>
105     <footer>
106         <p>This site is not affiliated with Blizzard Entertainment. ©2016
107             Blizzard Entertainment, Inc. All rights reserved /
108             Author : Sven Wikberg </p>
109     </footer>
110   </body>
111 </html>
```

Page administrateur

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\admin.php
lundi 19 juin 2017 10:55

1  <!--
2      Auteur: Sven Wikberg
3      Date: 19/06/2017
4      Description: Page administrateur
5  -->
6  <!doctype html>
7  <?php
8  session_start();
9
10
11 require_once('class/class.oc_dao.php');
12 require_once('class/class.oc_display.php');
13
14 if(isset($_SESSION['id_connected'])){ // si pas admin, on ne peut pas accéder à
15     cette page
16     $user = OcDao::SelectUserById($_SESSION['id_connected']);
17     if($user['is_admin'] == 0){
18         header('Location: index.php');
19     }
20 } else {
21     header('Location: index.php');
22 }
23 if (isset($_GET['action'])) { // selon l'action, la page récupère, teste ou process
24     des données différentes
25     if($_GET['action'] == 'ban'){ // on banni l'utilisateur sélectionné
26         if(isset($_GET['id'])){
27             OcDao::BanUserById($_GET['id']);
28         }
29     } elseif($_GET['action'] == 'unban'){ // on débanni l'utilisateur sélectionné
30         if(isset($_GET['id'])){
31             OcDao::UnbanUserById($_GET['id']);
32         }
33     } elseif($_GET['action'] == 'delete'){ // on supprime l'utilisateur sélectionné
34         if(isset($_GET['id'])){
35             OcDao::DeleteUserById($_GET['id']);
36         }
37     }
38     header('Location: admin.php');
39 }
40 <html lang="en">
41     <head>
42         <meta charset="utf-8">
43         <title>OverwatchCollection</title>
44         <link rel="stylesheet" href="css/style-main.css">
45     </head>
46     <body>
47         <header>
48             <?php
49             OcDisplay::DisplayNavbar();
50         ?>
51         </header>
52         <section id="admin">
53             <h1>Admin Page</h1>
54             <table border="1" style="width:100%;">
55                 <tr>
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\admin.php                                         lundi 19 juin 2017 10:55
56          <td><h2>Clean Users</h2></td>
57          <td><h2>Banned Users</h2></td>
58      </tr>
59      <tr>
60          <td>
61              <?php
62                  OcDisplay::DisplayListCleanUserWithBan ();
63              ?>
64          </td>
65          <td>
66              <?php
67                  OcDisplay::DisplayListBannedUserWithUnbanDelete ();
68              ?>
69          </td>
70      </tr>
71  </table>
72 </section>
73 <footer>
74     <p>This site is not affiliated with Blizzard Entertainment. ©2016
    Blizzard Entertainment, Inc. All rights reserved /
        Author : Sven Wikberg </p>
75 </footer>
76 </body>
77 </html>
```

Fonction ajout ou suppression d'objet pour un utilisateur

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\function\func.user_reward.php  
lundi 19 juin 2017 10:56  
1 <?php  
2 /*  
3     Auteur: Sven Wikberg  
4     Date: 19/06/2017  
5     Description: Fonction de ajout ou de suppression d'objets pour un utilisateur  
6 */  
7 if (isset($_GET['action'])) {  
8     if($_GET['action'] == 'add_user_reward' && isset($_GET['id_reward'])){  
9         if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){  
10            // si l'utilisateur est connecter  
11            $id_user = $_SESSION['id_connected'];  
12            $id_reward = $_GET['id_reward'];  
13            if(OcDao::IsCreatedUserReward($id_user, $id_reward)){ // test si  
14                l'enregistrement est deja crée  
15                OcDao::DeleteUserReward($id_user, $id_reward); // si c'est le cas on  
16                l'enlève  
17            } else {  
18                OcDao::InsertUserReward($id_user, $id_reward); // si ce n'est pas le  
19                cas on l'ajoute  
20            }  
21        header('Location: '. $_SERVER['PHP_SELF'] . (isset($_GET['id']) ? '?id=' . $_GET[  
22        'id'] : '') . '#rewards');  
23    }  
?>
```

Classe MyPdo

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.my_pdo.php  
lundi 19 juin 2017 10:59  
1 <?php  
2 /*  
3     Auteur: Sven Wikberg  
4     Date: 19/06/2017  
5     Description: classe de connexion a la base de données  
6 */  
7 require_once('config/config_db.php');  
8  
9 class MyPdo{  
10     private static $_myPdo;  
11  
12     public static function GetMyPdo(){  
13         try {  
14             if (!isset($_myPdo)) {  
15                 $dbc = new PDO('mysql:host=' . DB_HOST . ';dbname=' . DB_NAME . '',  
16                             DB_USER, DB_PWD, array(  
17                             PDO::MYSQL_ATTR_INIT_COMMAND => 'SET NAMES utf8',  
18                             PDO::ATTR_PERSISTENT => true,  
19                             PDO::ATTR_ERRMODE => PDO::ERRMODE_EXCEPTION));  
20         }  
21         catch (PDOException $e) {  
22             print "Erreur !: " . $e->getMessage() . "<br/>";  
23             die();  
24         }  
25         return $dbc;  
26     }  
27 }
```

Classe Oc_Dao

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
1 <?php
2 /*
3     Auteur: Sven Wikberg
4     Date: 19/06/2017
5     Description: Classe de récupération des données
6 */
7
8 require_once('class/class.my_pdo.php');
9
10 class OcDao{
11
12     // récupère tous les heros dans l'ordre de base/de l'id
13     static function SelectHeroes() {
14         $req = 'SELECT id_hero, name, description, id_role, health, armour, shield,
15             real_name, age, height, affiliation, base_of_operations, difficulty
16             FROM heroes';
17         $sql = MyPdo::GetMyPdo()->prepare($req);
18         $sql->execute();
19
20         return $sql->fetchAll(PDO::FETCH_ASSOC);
21     }
22
23     // récupère un heros grace a son id
24     static function SelectHeroById($id) {
25         $req = 'SELECT id_hero, name, description, id_role, health, armour, shield,
26             real_name, age, height, affiliation, base_of_operations, difficulty
27             FROM heroes
28             WHERE id_hero = :id';
29         $sql = MyPdo::GetMyPdo()->prepare($req);
30         $sql->bindParam(':id', $id, PDO::PARAM_INT);
31         $sql->execute();
32
33         $tmpReturn = $sql->fetch(PDO::FETCH_ASSOC);
34
35         if(count($tmpReturn) > 0){
36             return $tmpReturn;
37         } else {
38             return false;
39         }
40
41     // récupere les capacités d'un hero grace a son id
42     static function SelectAbilitiesByIdHero($id) {
43         $req = 'SELECT abilities.id_ability, abilities.name, abilities.description,
44             abilities.id_hero, abilities.is_ultimate
45             FROM abilities
46             WHERE abilities.id_hero = :id
47             ORDER BY abilities.is_ultimate';
48         $sql = MyPdo::GetMyPdo()->prepare($req);
49         $sql->bindParam(':id', $id, PDO::PARAM_INT);
50         $sql->execute();
51
52         $tmpReturn = $sql->fetchAll(PDO::FETCH_ASSOC);
53
54         if(count($tmpReturn) > 0){
55             return $tmpReturn;
56         } else {
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
  55         return false;
  56     }
  57 }
  58
  59 // récupère tous les roles dans l'ordre de base/de l'id
 60 static function SelectRoles() {
 61     $req = 'SELECT id_role, name FROM roles';
 62     $sql = MyPdo::GetMyPdo()->prepare($req);
 63     $sql->execute();
 64
 65     return $sql->fetchAll(PDO::FETCH_ASSOC);
 66 }
 67
 68 // récupère toutes les qualités/raretés des objets dans l'ordre de base/de l'id
 69 static function SelectQualities() {
 70     $req = 'SELECT id_quality, name FROM qualities';
 71     $sql = MyPdo::GetMyPdo()->prepare($req);
 72     $sql->execute();
 73
 74     return $sql->fetchAll(PDO::FETCH_ASSOC);
 75 }
 76
 77 // récupère tous les type d'objets dans l'ordre de base/de l'id
 78 static function SelectRewardTypes() {
 79     $req = 'SELECT id_reward_type, name FROM reward_types';
 80     $sql = MyPdo::GetMyPdo()->prepare($req);
 81     $sql->execute();
 82
 83     return $sql->fetchAll(PDO::FETCH_ASSOC);
 84 }
 85
 86 // récupère tous les événements dans l'ordre chronologique de la date de debut
 87 static function SelectEvents() {
 88     $req = 'SELECT id_event, name, start_date, end_date FROM events ORDER BY
 89     events.start_date ASC';
 90     $sql = MyPdo::GetMyPdo()->prepare($req);
 91     $sql->execute();
 92
 93     return $sql->fetchAll(PDO::FETCH_ASSOC);
 94 }
 95
 96 // récupère un evenement grace a son id
 97 static function SelectEventById($id) {
 98     $req = 'SELECT id_event, name, start_date, end_date FROM events WHERE
 99     id_event = :id';
100     $sql = MyPdo::GetMyPdo()->prepare($req);
101     $sql->bindParam(':id', $id, PDO::PARAM_INT);
102     $sql->execute();
103
104     if(count($tmpReturn) > 0) {
105         return $tmpReturn;
106     } else {
107         return false;
108     }
109 }
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php

lundi 19 juin 2017 11:01

```
110 // récupère tous les utilisateurs dans l'ordre alphabétique
111 static function SelectUsers() {
112     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
113     ORDER BY username ASC';
114     $sql = MyPdo::GetMyPdo()->prepare($req);
115     $sql->execute();
116
117     return $sql->fetchAll(PDO::FETCH_ASSOC);
118 }
119
120 // récupère tous les utilisateurs non-bannis dans l'ordre alphabétique (avec les
121 // admin en premier)
121 static function SelectCleanUsers() {
122     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
123     WHERE is_banned = 0 ORDER BY is_admin DESC, username ASC';
124     $sql = MyPdo::GetMyPdo()->prepare($req);
125     $sql->execute();
126
127     return $sql->fetchAll(PDO::FETCH_ASSOC);
128 }
129
130 // récupère tous les utilisateurs non-bannis dans l'ordre alphabétique (qui ne
131 // sont pas admin)
132 static function SelectCleanUsersNoAdmin() {
133     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
134     WHERE is_banned = 0 AND is_admin = 0 ORDER BY username ASC';
135     $sql = MyPdo::GetMyPdo()->prepare($req);
136     $sql->execute();
137
138     return $sql->fetchAll(PDO::FETCH_ASSOC);
139 }
140
141 // récupère tous les utilisateurs bannis dans l'ordre alphabétique (qui ne sont
142 // pas admin)
143 static function SelectBannedUsers() {
144     $req = 'SELECT id_user, username, email, is_banned, is_admin FROM users
145     WHERE is_banned = 1 AND is_admin = 0 ORDER BY username ASC';
146     $sql = MyPdo::GetMyPdo()->prepare($req);
147     $sql->execute();
148
149     return $sql->fetchAll(PDO::FETCH_ASSOC);
150 }
151
152 // modifie le champ is_banned de l'utilisateur sélectionné, en le passant à 1
153 // (pour bannir l'utilisateur)
154 static function BanUserById($id) {
155     $req = 'UPDATE users SET is_banned=1 WHERE id_user = :id';
156     $sql = MyPdo::GetMyPdo()->prepare($req);
157     $sql->bindParam(':id', $id, PDO::PARAM_INT);
158     $sql->execute();
159
160
161 // modifie le champ is_banned de l'utilisateur sélectionné, en le passant à 0
162 // (pour débannir l'utilisateur)
163 static function UnbanUserById($id) {
164     $req = 'UPDATE users SET is_banned=0 WHERE id_user = :id';
```

```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
158     $sql = MyPdo::GetMyPdo()->prepare($req);
159     $sql->bindParam(':id', $id, PDO::PARAM_INT);
160     $sql->execute();
161 }
162
163 // supprime l'utilisateur selectionné
164 static function DeleteUserById($id) {
165     $req = 'DELETE FROM users WHERE id_user = :id';
166     $sql = MyPdo::GetMyPdo()->prepare($req);
167     $sql->bindParam(':id', $id, PDO::PARAM_INT);
168     $sql->execute();
169 }
170
171 // modifie l'enregistrement d'un utilisateur en les replacant par les parametres
de la fonction
172 static function UpdateUserByIdNoPwd($id, $username, $email){
173     try{
174         $req = "UPDATE users SET username='$username', email='$email' WHERE
id_user=:id";
175         $sql = MyPdo::GetMyPdo()->prepare($req);
176         $sql->bindParam(':id', $id, PDO::PARAM_INT);
177         $sql->execute();
178     } catch (PDOException $e) {
179         print_rr($e);
180         return $e->errorInfo[1];
181     }
182     return null;
183 }
184
185 // recuper tous les heros et les range par role dans un tableau
186 static function SelectHeroesInArrayOfRole() {
187     $req = 'SELECT id_hero, name, id_role FROM heroes WHERE id_role = :id';
188     $sql = MyPdo::GetMyPdo()->prepare($req);
189     $tmpReturn = Array();
190
191     foreach (OcDao::SelectRoles() as $role) {
192         $sql->bindParam(':id', $role['id_role'], PDO::PARAM_INT);
193         $sql->execute();
194         $tmpReturn[$role['name']] = $sql->fetchAll(PDO::FETCH_ASSOC);
195     }
196
197     return $tmpReturn;
198 }
199
200 // ajoute un enregistrement a la table de liaison "users_rewards"
201 static function InsertUserReward($id_user, $id_reward) {
202     $req = 'INSERT INTO users_rewards(id_user, id_reward) VALUES
(:id_user,:id_reward)';
203     $sql = MyPdo::GetMyPdo()->prepare($req);
204     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
205     $sql->bindParam(':id_reward', $id_reward, PDO::PARAM_INT);
206     $sql->execute();
207 }
208
209 // supprime un enregistrement a la table de liaison "users_rewards"
210 static function DeleteUserReward($id_user, $id_reward) {
211     $req = 'DELETE FROM users_rewards WHERE id_user = :id_user AND id_reward =

```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
    :id_reward';
212     $sql = MyPdo::GetMyPdo()->prepare($req);
213     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
214     $sql->bindParam(':id_reward', $id_reward, PDO::PARAM_INT);
215     $sql->execute();
216 }
217
218 // test si un enregistrement de la table de liaison "users_rewards" existe, oui
219 // -> true / non -> false
220 static function IsCreatedUserReward($id_user, $id_reward){
221     $req = 'SELECT users_rewards.id_user, users_rewards.id_reward
222             FROM users_rewards
223             WHERE id_user = :id_user
224             AND id_reward = :id_reward';
225     $sql = MyPdo::GetMyPdo()->prepare($req);
226     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
227     $sql->bindParam(':id_reward', $id_reward, PDO::PARAM_INT);
228     $sql->execute();
229
230     if($sql->fetchAll(PDO::FETCH_ASSOC) == NULL) {
231         return false;
232     } else {
233         return true;
234     }
235
236 // recuper les id des objets d'un hero qu'un utilisateur a selectionné
237 static function SelectIdRewardsByIdHeroAndidUser($id_hero, $id_user){
238     $req = 'SELECT rewards.id_reward
239             FROM rewards
240             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
241             WHERE rewards.id_hero = :id_hero
242             AND users_rewards.id_user = :id_user';
243     $sql = MyPdo::GetMyPdo()->prepare($req);
244     $sql->bindParam(':id_hero', $id_hero, PDO::PARAM_INT);
245     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
246     $sql->execute();
247
248     $tmpReturn = NULL;
249
250     $cpt = 0;
251     foreach($sql->fetchAll(PDO::FETCH_ASSOC) as $array) {
252         $tmpReturn[$cpt] = $array['id_reward'];
253
254         $cpt++;
255     }
256
257     return $tmpReturn;
258 }
259
260 // recuper les id des objets d'un evenement qu'un utilisateur a selectionné
261 static function SelectIdRewardsByIdEventAndidUser($id_event, $id_user){
262     $req = 'SELECT rewards.id_reward
263             FROM rewards
264             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
265             WHERE rewards.id_event = :id_event
266             AND users_rewards.id_user = :id_user';

```

```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
267     $sql = MyPdo::GetMyPdo()->prepare($req);
268     $sql->bindParam(':id_event', $id_event, PDO::PARAM_INT);
269     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
270     $sql->execute();
271
272     $tmpReturn = NULL;
273
274     $cmpt = 0;
275     foreach($sql->fetchAll(PDO::FETCH_ASSOC) as $array){
276         $tmpReturn[$cmpt] = $array['id_reward'];
277
278         $cmpt++;
279     }
280
281     return $tmpReturn;
282 }
283
284 // recuper les id des objets d'aucun hero qu'un utilisateur a selectionné
285 static function SelectIdRewardsByNoIdHeroAndidUser($id_user){
286     $req = 'SELECT rewards.id_reward
287             FROM rewards
288             JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
289             WHERE (rewards.id_hero = 0 OR rewards.id_hero = NULL)
290             AND users_rewards.id_user = :id_user';
291     $sql = MyPdo::GetMyPdo()->prepare($req);
292     $sql->bindParam(':id_user', $id_user, PDO::PARAM_INT);
293     $sql->execute();
294
295     $tmpReturn = NULL;
296
297     $cmpt = 0;
298     foreach($sql->fetchAll(PDO::FETCH_ASSOC) as $array){
299         $tmpReturn[$cmpt] = $array['id_reward'];
300
301         $cmpt++;
302     }
303
304     return $tmpReturn;
305 }
306
307 // recuper tous les objets d'un heros et les range d'abord par catégorie et
308 // ensuite par rareté
309 static function SelectRewardsInArrayOfQualityAndTypeByIdHero($id) {
310     $req = 'SELECT rewards.id_reward, rewards.name, rewards.cost,
311             rewards.id_currency, rewards.id_event
312             FROM rewards
313             WHERE rewards.id_quality = :id_quality
314             AND rewards.id_reward_type = :id_reward_type
315             AND rewards.id_hero = :id_hero
316             ORDER BY rewards.name';
317     $sql = MyPdo::GetMyPdo()->prepare($req);
318
319     $rewardTypes = OcDao::SelectRewardTypes();
320     $qualities = OcDao::SelectQualities();
321     $tmpReturn = Array();
322
323     foreach ($rewardTypes as $rewardType) {

```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
322         foreach ($qualities as $quality) {
323             $sql->bindParam(':id_reward_type', $rewardType['id_reward_type'], PDO
324             ::PARAM_INT);
325             $sql->bindParam(':id_quality', $quality['id_quality'], PDO::PARAM_INT
326             );
327             $sql->bindParam(':id_hero', $id, PDO::PARAM_INT);
328             $sql->execute();
329
330             $tmp = $sql->fetchAll(PDO::FETCH_ASSOC); // afin de ne pas mettre du
331             vide dans le tableau
332             if(isset($tmp[0]))
333                 $tmpReturn[$rewardType['name']][$quality['name']] = $tmp;
334         }
335     }
336
337     if(count($tmpReturn) > 0){
338         return $tmpReturn;
339     } else {
340         return false;
341     }
342
343     // récupère tous les objets qui ne sont pas associé a un hero et les range
344     // d'abord par catégorie et ensuite par rareté
345     static function SelectRewardsInArrayOfQualityAndTypeByNoIdHero () {
346         $req = 'SELECT rewards.id_reward, rewards.name, rewards.cost,
347             rewards.id_currency, rewards.id_event
348             FROM rewards
349             WHERE rewards.id_quality = :id_quality
350             AND rewards.id_reward_type = :id_reward_type
351             AND (rewards.id_hero = 0 OR rewards.id_hero = NULL)
352             ORDER BY rewards.name';
353         $sql = MyPdo::GetMyPdo()->prepare($req);
354
355         $rewardTypes = OcDao::SelectRewardTypes();
356         $qualities = OcDao::SelectQualities();
357
358         foreach ($rewardTypes as $rewardType) {
359             foreach ($qualities as $quality) {
360                 $sql->bindParam(':id_reward_type', $rewardType['id_reward_type'], PDO
361                 ::PARAM_INT);
362                 $sql->bindParam(':id_quality', $quality['id_quality'], PDO::PARAM_INT
363                 );
364                 $sql->execute();
365
366                 $tmp = $sql->fetchAll(PDO::FETCH_ASSOC); // afin de ne pas mettre du
367                 vide dans le tableau
368                 if(isset($tmp[0]))
369                     $tmpReturn[$rewardType['name']][$quality['name']] = $tmp;
370             }
371         }
372
373         return $tmpReturn;
374     }
375
376     // récupère tous les objets d'un evenement et les range d'abord par catégorie et
377     // ensuite par rareté
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
370     static function SelectRewardsInArrayOfQualityAndTypeByIdEvent ($id) {
371
372         // il y a une jointure externe (LEFT JOIN) dans la requête car pas tous les
373         // objet qu'on veut récupérer on un hero qui leur est associé
374         $req = 'SELECT rewards.id_reward, rewards.name as r_name, rewards.cost,
375             rewards.id_currency, rewards.id_hero, heroes.name as h_name
376             FROM rewards
377             LEFT JOIN heroes ON heroes.id_hero = rewards.id_hero
378             WHERE rewards.id_quality = :id_quality
379             AND rewards.id_reward_type = :id_reward_type
380             AND rewards.id_event = :id_event
381             ORDER BY rewards.name';
382         $sql = MyPdo::GetMyPdo()->prepare($req);
383
384
385         foreach ($rewardTypes as $rewardType) {
386             foreach ($qualities as $quality) {
387                 $sql->bindParam(':id_reward_type', $rewardType['id_reward_type'], PDO
388                 ::PARAM_INT);
389                 $sql->bindParam(':id_quality', $quality['id_quality'], PDO::PARAM_INT
390                 );
391                 $sql->bindParam(':id_event', $id, PDO::PARAM_INT);
392                 $sql->execute();
393
394                 $tmp = $sql->fetchAll(PDO::FETCH_ASSOC); // afin de ne pas mettre du
395                 vide dans le tableau
396                 if(isset($tmp[0]))
397                     $tmpReturn[$rewardType['name']][$quality['name']] = $tmp;
398             }
399         }
400
401         if(count($tmpReturn) > 0){ // s'il n'y a rien, il y a un probleme, donc on
402             retourne false
403             return $tmpReturn;
404         } else {
405             return false;
406         }
407
408         // récupère les infos d'un utilisateur en fonction de son 'username' (utilisé
409         pour la connection)
410         static function SelectUserByUsername($username) {
411             $req = 'SELECT id_user, username, password, email, is_banned, is_admin FROM
412             users WHERE username = :username';
413             $sql = MyPdo::GetMyPdo()->prepare($req);
414             $sql->bindParam(':username', $username, PDO::PARAM_STR);
415             $sql->execute();
416
417             return $sql->fetch(PDO::FETCH_ASSOC);
418         }
419
420         // récupère les infos d'un utilisateur en fonction de son id
421         static function SelectUserById($id) {
422             $req = 'SELECT id_user, username, password, email, is_banned, is_admin FROM
423             users WHERE id_user = :id';
424
425
426
427
428
429
430
431
432
433
434
435
436
437
438
439
440
441
442
443
444
445
446
447
448
449
450
451
452
453
454
455
456
457
458
459
460
461
462
463
464
465
466
467
468
469
470
471
472
473
474
475
476
477
478
479
480
481
482
483
484
485
486
487
488
489
490
491
492
493
494
495
496
497
498
499
500
501
502
503
504
505
506
507
508
509
510
511
512
513
514
515
516
517
518
519
520
521
522
523
524
525
526
527
528
529
530
531
532
533
534
535
536
537
538
539
540
541
542
543
544
545
546
547
548
549
550
551
552
553
554
555
556
557
558
559
560
561
562
563
564
565
566
567
568
569
570
571
572
573
574
575
576
577
578
579
580
581
582
583
584
585
586
587
588
589
589
590
591
592
593
594
595
596
597
598
599
599
600
601
602
603
604
605
606
607
608
609
609
610
611
612
613
614
615
616
617
618
619
619
620
621
622
623
624
625
626
627
628
629
629
630
631
632
633
634
635
636
637
638
639
639
640
641
642
643
644
645
646
647
648
649
649
650
651
652
653
654
655
656
657
658
659
659
660
661
662
663
664
665
666
667
668
669
669
670
671
672
673
674
675
676
677
678
679
679
680
681
682
683
684
685
686
687
687
688
689
689
690
691
692
693
694
695
696
697
697
698
699
699
700
701
702
703
704
705
706
707
708
709
709
710
711
712
713
714
715
716
717
718
719
719
720
721
722
723
724
725
726
727
728
729
729
730
731
732
733
734
735
736
737
737
738
739
739
740
741
742
743
744
745
746
747
747
748
749
749
750
751
752
753
754
755
756
757
757
758
759
759
760
761
762
763
764
765
766
767
767
768
769
769
770
771
772
773
774
775
776
777
777
778
779
779
780
781
782
783
784
785
786
787
787
788
789
789
790
791
792
793
794
795
796
797
797
798
799
799
800
801
802
803
804
805
806
807
808
809
809
810
811
812
813
814
815
816
817
818
819
819
820
821
822
823
824
825
826
827
828
829
829
830
831
832
833
834
835
836
837
837
838
839
839
840
841
842
843
844
845
846
847
847
848
849
849
850
851
852
853
854
855
856
857
857
858
859
859
860
861
862
863
864
865
866
867
867
868
869
869
870
871
872
873
874
875
876
877
877
878
879
879
880
881
882
883
884
885
886
887
887
888
889
889
890
891
892
893
894
895
896
897
897
898
899
899
900
901
902
903
904
905
906
907
908
909
909
910
911
912
913
914
915
916
917
917
918
919
919
920
921
922
923
924
925
926
927
927
928
929
929
930
931
932
933
934
935
936
937
937
938
939
939
940
941
942
943
944
945
946
947
947
948
949
949
950
951
952
953
954
955
956
957
957
958
959
959
960
961
962
963
964
965
966
967
967
968
969
969
970
971
972
973
974
975
976
977
977
978
979
979
980
981
982
983
984
985
986
987
987
988
989
989
990
991
992
993
994
995
996
997
998
999
999
1000
1000
1001
1002
1003
1004
1005
1006
1007
1008
1009
1009
1010
1011
1012
1013
1014
1015
1016
1017
1017
1018
1019
1019
1020
1021
1022
1023
1024
1025
1026
1026
1027
1028
1028
1029
1030
1031
1032
1033
1034
1035
1036
1037
1037
1038
1039
1039
1040
1041
1042
1043
1044
1045
1046
1046
1047
1048
1048
1049
1050
1051
1052
1053
1054
1055
1056
1056
1057
1058
1058
1059
1060
1061
1062
1063
1064
1065
1066
1066
1067
1068
1068
1069
1070
1071
1072
1073
1074
1075
1075
1076
1077
1077
1078
1079
1079
1080
1081
1082
1083
1084
1085
1085
1086
1087
1087
1088
1089
1089
1090
1091
1092
1093
1093
1094
1095
1095
1096
1097
1097
1098
1099
1099
1100
1101
1101
1102
1103
1103
1104
1105
1105
1106
1107
1107
1108
1109
1109
1110
1111
1111
1112
1113
1113
1114
1115
1115
1116
1117
1117
1118
1119
1119
1120
1121
1121
1122
1123
1123
1124
1125
1125
1126
1127
1127
1128
1129
1129
1130
1131
1131
1132
1133
1133
1134
1135
1135
1136
1137
1137
1138
1139
1139
1140
1141
1141
1142
1143
1143
1144
1145
1145
1146
1147
1147
1148
1149
1149
1150
1151
1151
1152
1153
1153
1154
1155
1155
1156
1157
1157
1158
1159
1159
1160
1161
1161
1162
1163
1163
1164
1165
1165
1166
1167
1167
1168
1169
1169
1170
1171
1171
1172
1173
1173
1174
1175
1175
1176
1177
1177
1178
1179
1179
1180
1181
1181
1182
1183
1183
1184
1185
1185
1186
1187
1187
1188
1189
1189
1190
1191
1191
1192
1193
1193
1194
1195
1195
1196
1197
1197
1198
1199
1199
1200
1201
1201
1202
1203
1203
1204
1205
1205
1206
1207
1207
1208
1209
1209
1210
1211
1211
1212
1213
1213
1214
1215
1215
1216
1217
1217
1218
1219
1219
1220
1221
1221
1222
1223
1223
1224
1225
1225
1226
1227
1227
1228
1229
1229
1230
1231
1231
1232
1233
1233
1234
1235
1235
1236
1237
1237
1238
1239
1239
1240
1241
1241
1242
1243
1243
1244
1245
1245
1246
1247
1247
1248
1249
1249
1250
1251
1251
1252
1253
1253
1254
1255
1255
1256
1257
1257
1258
1259
1259
1260
1261
1261
1262
1263
1263
1264
1265
1265
1266
1267
1267
1268
1269
1269
1270
1271
1271
1272
1273
1273
1274
1275
1275
1276
1277
1277
1278
1279
1279
1280
1281
1281
1282
1283
1283
1284
1285
1285
1286
1287
1287
1288
1289
1289
1290
1291
1291
1292
1293
1293
1294
1295
1295
1296
1297
1297
1298
1299
1299
1300
1301
1301
1302
1303
1303
1304
1305
1305
1306
1307
1307
1308
1309
1309
1310
1311
1311
1312
1313
1313
1314
1315
1315
1316
1317
1317
1318
1319
1319
1320
1321
1321
1322
1323
1323
1324
1325
1325
1326
1327
1327
1328
1329
1329
1330
1331
1331
1332
1333
1333
1334
1335
1335
1336
1337
1337
1338
1339
1339
1340
1341
1341
1342
1343
1343
1344
1345
1345
1346
1347
1347
1348
1349
1349
1350
1351
1351
1352
1353
1353
1354
1355
1355
1356
1357
1357
1358
1359
1359
1360
1361
1361
1362
1363
1363
1364
1365
1365
1366
1367
1367
1368
1369
1369
1370
1371
1371
1372
1373
1373
1374
1375
1375
1376
1377
1377
1378
1379
1379
1380
1381
1381
1382
1383
1383
1384
1385
1385
1386
1387
1387
1388
1389
1389
1390
1391
1391
1392
1393
1393
1394
1395
1395
1396
1397
1397
1398
1399
1399
1400
1401
1401
1402
1403
1403
1404
1405
1405
1406
1407
1407
1408
1409
1409
1410
1411
1411
1412
1413
1413
1414
1415
1415
1416
1417
1417
1418
1419
1419
1420
1421
1421
1422
1423
1423
1424
1425
1425
1426
1427
1427
1428
1429
1429
1430
1431
1431
1432
1433
1433
1434
1435
1435
1436
1437
1437
1438
1439
1439
1440
1441
1441
1442
1443
1443
1444
1445
1445
1446
1447
1447
1448
1449
1449
1450
1451
1451
1452
1453
1453
1454
1455
1455
1456
1457
1457
1458
1459
1459
1460
1461
1461
1462
1463
1463
1464
1465
1465
1466
1467
1467
1468
1469
1469
1470
1471
1471
1472
1473
1473
1474
1475
1475
1476
1477
1477
1478
1479
1479
1480
1481
1481
1482
1483
1483
1484
1485
1485
1486
1487
1487
1488
1489
1489
1490
1491
1491
1492
1493
1493
1494
1495
1495
1496
1497
1497
1498
1499
1499
1500
1501
1501
1502
1503
1503
1504
1505
1505
1506
1507
1507
1508
1509
1509
1510
1511
1511
1512
1513
1513
1514
1515
1515
1516
1517
1517
1518
1519
1519
1520
1521
1521
1522
1523
1523
1524
1525
1525
1526
1527
1527
1528
1529
1529
1530
1531
1531
1532
1533
1533
1534
1535
1535
1536
1537
1537
1538
1539
1539
1540
1541
1541
1542
1543
1543
1544
1545
1545
1546
1547
1547
1548
1549
1549
1550
1551
1551
1552
1553
1553
1554
1555
1555
1556
1557
1557
1558
1559
1559
1560
1561
1561
1562
1563
1563
1564
1565
1565
1566
1567
1567
1568
1569
1569
1570
1571
1571
1572
1573
1573
1574
1575
1575
1576
1577
1577
1578
1579
1579
1580
1581
1581
1582
1583
1583
1584
1585
1585
1586
1587
1587
1588
1589
1589
1590
1591
1591
1592
1593
1593
1594
1595
1595
1596
1597
1597
1598
1599
1599
1600
1601
1601
1602
1603
1603
1604
1605
1605
1606
1607
1607
1608
1609
1609
1610
1611
1611
1612
1613
1613
1614
1615
1615
1616
1617
1617
1618
1619
1619
1620
1621
1621
1622
1623
1623
1624
1625
1625
1626
1627
1627
1628
1629
1629
1630
1631
1631
1632
1633
1633
1634
1635
1635
1636
1637
1637
1638
1639
1639
1640
1641
1641
1642
1643
1643
1644
1645
1645
1646
1647
1647
1648
1649
1649
1650
1651
1651
1652
1653
1653
1654
1655
1655
1656
1657
1657
1658
1659
1659
1660
1661
1661
1662
1663
1663
1664
1665
1665
1666
1667
1667
1668
1669
1669
1670
1671
1671
1672
1673
1673
1674
1675
1675
1676
1677
1677
1678
1679
1679
1680
1681
1681
1682
1683
1683
1684
1685
1685
1686
1687
1687
1688
1689
1689
1690
1691
1691
1692
1693
1693
1694
1695
1695
1696
1697
1697
1698
1699
1699
1700
1701
1701
1702
1703
1703
1704
1705
1705
1706
1707
1707
1708
1709
1709
1710
1711
1711
1712
1713
1713
1714
1715
1715
1716
1717
1717
1718
1719
1719
1720
1721
1721
1722
1723
1723
1724
1725
1725
1726
1727
1727
1728
1729
1729
1730
1731
1731
1732
1733
1733
1734
1735
1735
1736
1737
1737
1738
1739
1739
1740
1741
1741
1742
1743
1743
1744
1745
1745
1746
1747
1747
1748
1749
1749
1750
1751
1751
1752
1753
1753
1754
1755
1755
1756
1757
1757
1758
1759
1759
1760
1761
1761
1762
1763
1763
1764
1765
1765
1766
1767
1767
1768
1769
1769
1770
1771
1771
1772
1773
1773
1774
1775
1775
1776
1777
1777
1778
1779
1779
1780
1781
1781
1782
1783
1783
1784
1785
1785
1786
1787
1787
1788
1789
1789
1790
1791
1791
1792
1793
1793
1794
1795
1795
1796
1797
1797
1798
1799
1799
1800
1801
1801
1802
1803
1803
1804
1805
1805
1806
1807
1807
1808
1809
1809
1810
1811
1811
1812
1813
1813
1814
1815
1815
1816
1817
1817
1818
1819
1819
1820
1821
1821
1822
1823
1823
1824
1825
1825
1826
1827
1827
1828
1829
1829
1830
1831
1831
1832
1833
1833
1834
1835
1835
1836
1837
1837
1838
1839
1839
1840
1841
1841
1842
1843
1843
1844
1845
1845
1846
1847
1847
1848
1849
1849
1850
1851
1851
1852
1853
1853
1854
1855
1855
1856
1857
1857
1858
1859
1859
1860
1861
1861
1862
1863
1863
1864
1865
1865
1866
1867
1867
1868
1869
1869
1870
1871
1871
1872
1873
1873
1874
1875
1875
1876
1877
1877
1878
1879
1879
1880
1881
1881
1882
1883
1883
1884
1885
1885
1886
1887
1887
1888
1889
1889
1890
1891
1891
1892
1893
1893
1894
1895
1895
1896
1897
1897
1898
1899
1899
1900
1901
1901
1902
1903
1903
1904
1905
1905
1906
1907
1907
1908
1909
1909
1910
1911
1911
1912
1913
1913
1914
1915
1915
1916
1917
1917
1918
1919
1919
1920
1921
1921
1922
1923
1923
1924
1925
1925
1926
1927
1927
1928
1929
1929
1930
1931
1931
1932
1933
1933
1934
1935
1935
1936
1937
1937
1938
1939
1939
1940
1941
1941
1942
1943
1943
1944
1945
1945
1946
1947
1947
1948
1949
1949
1950
1951
1951
1952
1953
1953
1954
1955
1955
1956
1957
1957
1958
1959
1959
1960
1961
1961
1962
1963
1963
1964
1965
1965
1966
1967
1967
1968
1969
1969
1970
1971
1971
1972
1973
1973
1974
1975
1975
1976
1977
1977
1978
1979
1979
1980
1981
1981
1982
1983
1983
1984
1985
1985
1986
1987
1987
1988
1989
1989
1990
1991
1991
1992
1993
1993
1994
1995
1995
1996
1997
1997
1998
1999
1999
2000
2001
2001
2002
2003
2003
2004
2005
2005
2006
2007
2007
2008
2009
2009
2010
2011
2011
2012
2013
2013
2014
2015
2015
2016
2017
2017
2018
2019
2019
2020
2021
2021
2022
2023
2023
2024
2025
2025
2026
2027
2027
2028
2029
2029
2030
2031
2031
2032
2033
2033
2034
2035
2035
2036
2037
2037
2038
2039
2039
2040
2041
2041
2042
2043
2043
2044
2045
2045
2046
2047
2047
2048
2049
2049
2050
2051
205
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
418     $sql = MyPdo::GetMyPdo()->prepare($req);
419     $sql->bindParam(':id', $id, PDO::PARAM_INT);
420     $sql->execute();
421
422     return $sql->fetch(PDO::FETCH_ASSOC);
423 }
424
425 // insère un nouvel enregistrement dans la table "users"
426 // retourne le code de l'erreur s'il y en a une, ou null s'il n'y en a pas
427 static function InsertUser($username, $email, $password) {
428     try{
429         $req = 'INSERT INTO users (username, password, email, is_banned) VALUES
430               (:username, :email, :password, 0)';
431         $sql = MyPdo::GetMyPdo()->prepare($req);
432         $sql->bindParam(':username', $username);
433         $sql->bindParam(':email', $email);
434         $sql->bindParam(':password', $password);
435         $sql->execute();
436     } catch (PDOException $e) {
437         print_r($e);
438         return $e->errorInfo[1];
439     }
440     return null;
441 }
442
443 // compte le nombre d'objets en tous
444 static function SelectCountReward(){
445     $req = 'SELECT COUNT(id_reward) AS c FROM rewards';
446     $sql = MyPdo::GetMyPdo()->prepare($req);
447     $sql->execute();
448
449     return $sql->fetch(PDO::FETCH_ASSOC) ['c'];
450 }
451
452 // compte le nombre d'objets d'un utilisateur
453 static function SelectCountRewardByIdUser($id) {
454     $req = 'SELECT COUNT(id_reward) AS c FROM users_rewards WHERE id_user = :id';
455     $sql = MyPdo::GetMyPdo()->prepare($req);
456     $sql->bindParam(':id', $id, PDO::PARAM_INT);
457     $sql->execute();
458
459     return $sql->fetch(PDO::FETCH_ASSOC) ['c'];
460 }
461
462 // compte le nombre d'objets qu'a chaque evenement
463 static function SelectCountRewardEvents() {
464     $req = 'SELECT events.name, COUNT(rewards.id_reward) AS c
465           FROM rewards
466             JOIN events ON rewards.id_event = events.id_event
467               GROUP BY events.id_event
468                 ORDER BY events.start_date';
469     $sql = MyPdo::GetMyPdo()->prepare($req);
470     $sql->execute();
471
472     return $sql->fetchAll(PDO::FETCH_ASSOC);
473 }
```



```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_dao.php                                         lundi 19 juin 2017 11:01
526          WHERE users_rewards.id_user = :id1
527          GROUP BY heroes.id_hero)
528      UNION
529          (SELECT heroes.name, 0 AS c
530           FROM heroes
531          WHERE heroes.id_hero NOT IN
532              (SELECT heroes.id_hero
533               FROM heroes
534               JOIN rewards ON rewards.id_hero = heroes.id_hero
535               JOIN users_rewards ON users_rewards.id_reward = rewards.id_reward
536               WHERE users_rewards.id_user = :id2
537               GROUP BY heroes.id_hero))
538          ORDER BY name ASC';
539 $sql = MyPdo::GetMyPdo()->prepare($req);
540 $sql->bindParam(':id1', $id, PDO::PARAM_INT);
541 $sql->bindParam(':id2', $id, PDO::PARAM_INT);
542 $sql->execute();
543
544     return $sql->fetchAll(PDO::FETCH_ASSOC);
545 }
546 ?
547 ?>
```

Class Oc_Display

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php
lundi 19 juin 2017 11:04

1 <?php
2 /*
3     Auteur: Sven Wikberg
4     Date: 19/06/2017
5     Description: classe d'affichage des donnees
6 */
7 class OcDisplay{
8
9     // affiche la barre de navigation, avec des changement selon si on est connecter
10    ou pas
11    static function DisplayNavbar(){
12        $display = '';
13
14        $display .= '<ul>';
15        $display .= '<li><a href="index.php">Index</a></li>';
16        $display .= '<li><a href="heroes.php">Heroes</a></li>';
17        $display .= '<li><a href="events.php">Events</a></li>';
18        $display .= '<li><a href="rewards.php">Others Rewards</a></li>';
19        if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null){
20            if(fnmatch('*user.php', $_SERVER['PHP_SELF'])){
21                $display .= '<li><a href="user.php?action=deco">Log Out</a></li>';
22            } else {
23                $display .= '<li><a href="user.php">My Account</a></li>';
24            }
25        } else {
26            $display .= '<li><a href="user.php">Login/Sign In</a></li>';
27        }
28
29        $display .= '</ul>';
30
31        echo $display;
32    }
33
34    // affiche une liste des utilisateur non bannis avec des bouton pour bannir
35    // les utilisateurs
36    static function DisplayListCleanUserWithBan(){
37        $clean_users = OcDao::SelectCleanUsersNoAdmin(); // retourne les utilisateur
38        non bannis (sans les admins)
39        $display = '';
40
41        $display .= '<ul style="height:100%; width:300px; overflow:hidden;
42        overflow-y:auto;">';
43        foreach ($clean_users as $clean_user) { // on affiche les utilisateur non
44            bannis
45            $display .= '<li>' . $clean_user['username'] . ' / ' . $clean_user[
46            'email'];
47            $display .= '&nbsp;<a href="admin.php?action=ban&id=' . $clean_user[
48            'id_user'] . '"></a>';
49            $display .= '</li>';
50        }
51        $display .= '</ul>';
52
53        echo $display;
54    }
55
56    // affiche un liste des utilisateurs bannis avec des boutons pour les de-bannir

```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php lundi 19 juin 2017 11:05
ou les supprimer
50 static function DisplayListBannedUserWithUnbanDelete() {
51     $banned_users = OcDao::SelectBannedUsers(); // retourne les utilisateurs
52     // bannis (sans les admins)
53     $display = '';
54
55     $display .= '<ul style="height:100%; width:300px; overflow:hidden;
56     overflow-y:auto;">';
57     foreach ($banned_users as $banned_user) { // on affiche les utilisateur bannis
58         $display .= '<li>' . $banned_user['username'] . ' / ' . $banned_user[
59             'email'];
60         $display .= '&nbsp;<a href="admin.php?action=unban&id=' . $banned_user[
61             'id_user'] . '"></a>';
62         $display .= '&nbsp;<a href="admin.php?action=delete&id=' . $banned_user[
63             'id_user'] . '"></a>';
64         $display .= '</li>';
65     }
66     $display .= '</ul>';
67
68     echo $display;
69 }
70
71 // affiche les objets d'un evenement triés, avec les liens pour les selectionner
72 // si l'utilisateur est connecté
73 static function DisplayEventRewards($id_event) {
74     $rewards_array = OcDao::SelectRewardsInArrayOfQualityAndTypeByIdEvent(
75         $id_event); // retourne un tableau de rewards
76
77     if(isset($_SESSION['id_connected'])) && $_SESSION['id_connected'] != null) // si l'utilisateur est connecter on va chercher l'id des rewards qu'il a
78         $rewards_owned = OcDao::SelectIdRewardsByIdEventAndidUser($id_event,
79             $_SESSION['id_connected']);
80     else
81         $rewards_owned = false;
82
83     if($rewards_array) {
84         $display = '';
85
86         foreach ($rewards_array as $key => $type) {
87             $quality_count = count($type); // on compte le nombre de boite il y
88             // par catégorie afin de definir la taille de cell-ci
89             $display .= '<div class="rewards_type" style="width:' . (
90                 $quality_count == 4 ? '100' : $quality_count * 24 - 0.5) . '%>';
91             $display .= '<h2>' . $key. '</h2>';
92             $display .= '<div>';
93             foreach ($type as $key => $quality) {
94                 $display .= '<div>';
95                 $display .= '<h3 class="" . $key. '">' . $key. '</h3><ul>';
96                 foreach ($quality as $key => $reward) {
97
98                     // si l'id reward correspond a un element du tableau alors
99                     // l'utilisateur a la reward, donc on l'affiche vert
100                     if($rewards_owned && in_array($reward['id_reward'],
101                         $rewards_owned)) {
102                         $css_string = 'style="color: #00c600;"';
103                         $display .= '<li>' . $reward['label'] . '</li>';
104                     } else {
105                         $display .= '<li>' . $reward['label'] . '</li>';
106                     }
107                 }
108             }
109         }
110     }
111 }
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                                         lundi 19 juin 2017 11:04
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
} else{
    $css_string = '';
}

// si un hero est associer a l'objet on l'affiche sinon, on
affiche juste le nom de l'objet
// le lien envoie 3 variables en GET:
// id, qui est l'id de l'événement, qui sert à revenir
sur la bonne page
// action, qui signifie qu'il faut faire une action en
l'occurrence ajouter un "user_reward"
// id_reward, qui est l'id de l'objet sur lequel on a
cliqué

$display .= '<li>';
if(isset($_SESSION['id_connected']) && $_SESSION[
'id_connected'] != null) // si un utilisateur est connecté
on affiche les liens pour ajouter/enlever un "user_reward"
    $display .= '<a ' . $css_string. ' href="event.php?id=' .
$id_event. '&action=add_user_reward&id_reward=' . $reward[
'id_reward']. '">';
    $display .= $reward['r_name'] . ($reward['h_name'] != NULL ?
' / ' . $reward['h_name'] : '');
    if(isset($_SESSION['id_connected']) && $_SESSION[
'id_connected'] != null)
        $display .= '</a>';
    $display .= '</li>';
}
$display .= '</ul>';
$display .= '</div>';

echo $display;
} else {
    header('Location: events.php');
}
}

// affiche les informations de base d'un événement
static function DisplayEventInfo($id_event){
    $event = OcDao::SelectEventById($id_event); // retourne l'enregistrement de
l'événement depuis la base

    if($event){
        $display = '<h1>' . $event['name']. '</h1>';
        $display .= '<div><p>Beginning Date: ' . $event['start_date']. '</p><p>Ending Date: ' . $event['end_date']. '</p></div>';

        echo $display;
    } else {
        header('Location: events.php');
    }
}

// affiche un tableau des événements avec un paramètre pour le nombre de colonnes
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                                         lundi 19 juin 2017 11:04
136     du tableau
137     static function DisplayEvents($nb_col_max) { // $nb_col_max = nombre de colonnes
138         maximum pour les tableaux de heros
139         $events = OcDao::SelectEvents();
140         $display = '';
141         $nb_col_current = 0; // nombre de colonnes actuelle
142
143         $display .= '<table>';
144         foreach ($events as $event) {
145             $nb_col_current++;
146
147             if($nb_col_current == 1) // si le nombre de colonne actuelle vaut 1,
148             c'est qu'on est au debut d'une nouvelle ligne donc on ouvre une balise
149             <tr>
150                 $display .= '<tr>';
151                 $display .= '<td><a href="event.php?id=' . $event['id_event']. '"><div>' .
152                 $event['name']. '</div><div>From: ' .
154                 $event['start_date']. ' To: ' . $event['start_date']. '</div></a></td>';
155             if($nb_col_current == $nb_col_max) { // si le nombre de colonne actuelle
156             vaut le nombre de colonne max, c'est qu'on est a la fin de la ligne donc
157             on ferme une balise <tr>
158                 $display .= '</tr>';
159                 $nb_col_current = 0;
160             }
161         }
162     }
163
164     $display .= '</table>';
165
166     echo $display;
167 }
168
169 // affiche les objets d'un hero triés, avec les liens pour les selectionner si
170 // l'utilisateur est connecté
171 static function DisplayHeroRewards($id_hero) {
172     $rewards_array = OcDao::SelectRewardsInArrayOfQualityAndTypeByIdHero($id_hero
173     ); // retourne un tableau de rewards
174
175     if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null) // si l'utilisateur est connecter on va chercher l'id des rewards qu'il a
176     $rewards_owned = OcDao::SelectIdRewardsByIdHeroAndidUser($id_hero,
177     $_SESSION['id_connected']);
178
179     else
180     $rewards_owned = false;
181
182     if($rewards_array) {
183         $display = '';
184
185         foreach ($rewards_array as $key => $type) {
186             $quality_count = count($type); // on compte le nombre de boite il y
187             par catégorie afin de definir la taille de cell-ci
188             $display .= '<div class="rewards_type" style="width:' . (
189             $quality_count == 4 ? '100' : $quality_count * 24 - 0.5) . '%>';
190             $display .= '<h2>' . $key. '</h2>';
191             $display .= '<div>';
192             foreach ($type as $key => $quality) {
193                 $display .= '<div>';
194                 $display .= '<h3 class="" . $key. '">' . $key. '</h3><ul>';
195             }
196         }
197     }
198 }
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php
lundi 19 juin 2017 11:04

    c'est qu'on est au debut d'une nouvelle ligne donc on ouvre une
    balise <tr>
224        $display .= '<tr>';
225        $display .= '<td width=' . 100 / $nb_col_max . '%>';
226        $display .= '<h2>' . $ability['name'] . '</h2><p>' . $ability[
227            'description'] . '</p>';
228        $display .= ($ability['is_ultimate'] == 0 ? '<h3>Ultimate</h3>' : '');
229        $display .= '</td>';
230
231        if($nb_col_current == $nb_col_max){ // si le nombre de colonne
232            actuelle vaut le nombre de colonne max, c'est qu'on est a la fin de
233            la ligne donc on ferme une balise <tr>
234            $display .= '</tr>';
235            $nb_col_current = 0;
236        }
237
238        $display .= '</table>';
239
240        echo $display;
241    } else {
242        header('Location: heroes.php');
243    }
244
245    // affiche les informations de base d'un hero
246    static function DisplayHeroInfo($id_hero){
247        $hero = OcDao::SelectHeroById($id_hero); // retourne l'enregistrement du
248        heros depuis la base
249
250        if($hero){
251            $display = '<h1>' . $hero['name'] . '</h1>';
252            $display .= '<p>' . $hero['description'] . '</p>';
253            $display .= '<div><p>Real Name: ' . $hero['real_name'] . '</p><p>Base Of
254            Operations: ' . $hero['base_of_operations'] . '</p></div>';
255            $display .= '<div><p>Health: ' . $hero['health'] . '</p><p>Armour: ' . $hero
256            ['armour'] . '</p><p>Shield: ' . $hero['shield'] . '</p></div>';
257            $display .= '<div><p>Affiliation: ' . $hero['affiliation'] .
258            '</p><p>Difficulty: ' . $hero['difficulty'] . '</p></div>';
259
260            echo $display;
261        } else {
262            header('Location: heroes.php');
263        }
264
265        // affiche un tableau de hero pour chaque role, avec un parametre pour le nombre
266        // de colonne de ces tableaux
267        static function DisplayHeroesByRole($nb_col_max){ // $nb_col_max = nombre de
268            colonnes maximum pour les tableaux de heros
269            $heroes_array = OcDao::SelectHeroesInArrayOfRole(); // retourne un tableau,
270            trié par role, de tableau de héros
271            $display = '';
272            $nb_col_current = 0; // nombre de colonnes actuelle
273
274            foreach ($heroes_array as $key => $role) { // pour chaque role on fait un
275                tableau afin d'ordonner et de regrouper les heros

```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                                         lundi 19 juin 2017 11:04
268     $display .= '<div>';
269     $display .= '<h1>' . $key . '</h1>';
270     $display .= '<table>';
271     foreach ($role as $hero) {
272         $nb_col_current++;
273
274         if ($nb_col_current == 1) // si le nombre de colonne actuelle vaut 1,
275         c'est qu'on est au debut d'une nouvelle ligne donc on ouvre une
276         balise <tr>
277         $display .= '<tr>';
278         $display .= '<td><a href="hero.php?id=' . $hero['id_hero'] . '"><div>' . $hero['name'].
281         '</div></a></td>';
282         if ($nb_col_current == $nb_col_max) { // si le nombre de colonne
283         actuelle vaut le nombre de colonne max, c'est qu'on est a la fin de
284         la ligne donc on ferme une balise <tr>
285         $display .= '</tr>';
286         $nb_col_current = 0;
287     }
288
289 // affiche une liste des utilisateurs et adnministrateurs non bannis
290 static function DisplayListUsers(){
291     $users = OcDao::SelectCleanUsers(); // recupere le utilisateur depuis la
292     base de données
293
294     $display = '';
295     $display .= '<ul style="height:100%; width:170px; overflow:hidden;
296     overflow-y:auto;">';
297     foreach ($users as $user) {
298         $display .= '<li>' . $user['username'] . ($user['is_admin'] == 1 ? ' /
299         Admin' : '') . '</li>';
300     }
301     $display .= '</ul>';
302
303     echo $display;
304 }
305
306 // afiche les objet non lié a un heros de facon triee, avec les liens pour les
307 // selectionner si l'utilisateur est connecté
308 static function DisplayOtherRewards(){
309     $rewards_array = OcDao::SelectRewardsInArrayOfQualityAndTypeByNoIdHero(); //
310     retourne un tableau de rewards
311
312     if(isset($_SESSION['id_connected']) && $_SESSION['id_connected'] != null) // si l'utilisateur est connecter on va chercher l'id des rewards qu'il a
313     $rewards_owned = OcDao::SelectIdRewardsByNoIdHeroAndIdUser($_SESSION[
314     'id_connected']);
315     else
316         $rewards_owned = false;
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php

lundi 19 juin 2017 11:04

```

311
312     $display = '';
313
314     foreach ($rewards_array as $key => $type) {
315         $quality_count = count($type); // on compte le nombre de boite il y par
316         // catégorie afin de definir la taille de celle-ci
317         $display .= '<div class="rewards_type" style="width:' . ($quality_count
318         == 4 ? '100' : $quality_count * 24 - 0.5) . '%>';
319         $display .= '<h2>' . $key. '</h2>';
320         $display .= '<div>';
321         $display .= '<h3 class="' . $key. '">' . $key. '</h3><ul>';
322         foreach ($type as $key => $quality) {
323             $display .= '<li>';
324             // si l'id reward correspond a un element du tableau alors
325             // l'utilisateur a la reward, donc on l'affiche vert
326             if($rewards_owned && in_array($reward['id_reward'],
327             $rewards_owned)){
328                 $css_string = 'style="color: #00c600;"';
329             } else{
330                 $css_string = '';
331             }
332             // le lien envoie 2 variables en GET:
333             // action, qui signifie qu'il faut faire une action en
334             // l'occurrence ajouter un "user_reward"
335             // id_reward, qui est l'id de l'objet sur lequel on a
336             // cliqué
337             $display .= '<a ' . $css_string. '
338             href="rewards.php?action=add_user_reward&id_reward=' . $reward
339             ['id_reward']. '"';
340             $display .= $reward['name'];
341             if(isset($_SESSION['id_connected']) && $_SESSION['id_connected']
342             != null) // si un utilisateur est connecté on affiche le liens
343             pour ajouter/enlever un "user_reward"
344             $display .= '<a ' . $css_string. '
345             href="rewards.php?action=remove_user_reward&id_reward=' . $reward
346             ['id_reward']. '"';
347             $display .= '</a>';
348             $display .= '</li>';
349         }
350         $display .= '</ul>';
351         $display .= '</div>';
352     }
353     echo $display;
354 }

// affichage du formulaire de connection avec les message d'erreurs, en GET,
s'il y en a
static function DisplayLogin(){
    $display = '<section id="login"><form action="user.php?action=login"
method="post">
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                                         lundi 19 juin 2017 11:04
355                                     <fieldset>
356                                         <legend><h2>Login</h2></legend>
357                                         <p>Username :</p>
358                                         <input maxlength="25" required type="text" name="username"
359                                         value="">;
360
361     if(isset($_GET['msg'])) {
362         if($_GET['msg'] == 'wrongUn') // le nom d'utilisateur n'est pas valide
363             $display .= 'Username not valid';
364         elseif($_GET['msg'] == 'banned') // l'utilisateur est banni
365             $display .= 'Your account has been banned';
366
367         $display .=    '<br>
368                         <p>Password:</p>
369                         <input required type="password" name="password" value="">';
370
371     if(isset($_GET['msg'])) {
372         if($_GET['msg'] == 'wrongPw') // le mot de passe n'est pas valide
373             $display .= 'Wrong password';
374
375         $display .=    '<br><br>
376                         <input type="submit" value="Submit">
377                         </fieldset>
378                     </form></section>';
379     echo $display;
380 }
381
382 // affichage du formulaire de création de compte avec les message d'erreurs, en
383 // GET, s'il y en a
384 static function DisplaySignin(){
385     $display = '<section id="sign_in">
386         <form action="user.php?action=signin" method="post">
387             <fieldset>
388                 <legend><h2>Sign in</h2></legend>
389                 <p>Username* :</p>
390                 <input maxlength="25" required type="text"
391                 name="username" value="">;
392
393     if(isset($_GET['msg'])) {
394         if($_GET['msg'] == 'duplicate') // nom d'utilisateur ou email déjà utilisé
395             $display .= 'Username or email already used';
396
397         $display .=
398             '<p>Email* :</p>
399             <input maxlength="100" required type="email"
400             name="email" value=""><br>
401             <p>Password* :</p>
402             <input minlength="8" required type="password"
403             name="password" value=""><br><br>
404             <input type="submit" value="Submit">
405             </fieldset>
406         </form>
407     </section>';
408     echo $display;
409 }
410
411 // affiche les informations du compte de l'utilisateur connecté
412 static function DisplayAccountInfo($id_user) {
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                                         lundi 19 juin 2017 11:04
407     $user = OcDao::SelectUserById($id_user);
408
409     $display = '';
410     $display .= '<section id="account_info">';
411     $display .= '<div id="title">';
412     $display .= '<h1>My Account </h1>; // '&nbsp' sert a forcer un
413     espace en html
414     $display .= '<a href="user.php?goto=updating"></a>';
416     $display .= '</div>';
417
418     if(isset($_GET['msg'])) {
419         if($_GET['msg'] == 'updateDuplicate')
420             $display .= '<h3>Username/Email already used</h3>';
421
422         $display .= '<div class="flex_row">';
423         $display .= '<div style="width:70%;">';
424         $display .= '<p>Username: ' . $user['username'] . '</p>';
425         $display .= '<p>Email: ' . $user['email'] . '</p>';
426         $display .= '</div>';
427         if($user['is_admin']) {
428             $display .= '<div>';
429             $display .= '<a href="admin.php"><p>Admin page</p></a>';
430             $display .= '</div>';
431
432         $display .= '</div>';
433         $display .= '</section>';
434
435         echo $display;
436     }
437
438 // affiche le formulaire qui permet de modifier les infos de l'utilisateur
439 static function DisplayAccountInfoUpdating($id_user) {
440     $user = OcDao::SelectUserById($id_user);
441
442     $display = '';
443     $display .= '<section id="account_info_updating">';
444     $display .= '<div id="title">';
445     $display .= '<h1>Updating account info</h1>';
446     $display .= '</div>';
447
448     $display .= '<form action="user.php?action=update" method="post">';
449     $display .= '<p>Username :</p>';
450     $display .= '<input maxlength="25" required type="text" name="username"
451     value="' . $user['username'] . '"><br>';
452     $display .= '<p>Email: </p>';
453     $display .= '<input maxlength="100" required type="email" name="email"
454     value="' . $user['email'] . '"><br><br>';
455     $display .= '<input type="submit" value="Submit">';
456     $display .= '</form>';
457
458     echo $display;
459 }
460
461 // affiche la section des statistiques d'un utilisateur
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                                         lundi 19 juin 2017 11:04
460     static function DisplayAccountStats($id_user) {
461         echo '<section id="account_stats">';
462         OcDisplay::DisplayMainProgressBar($id_user);
463         OcDisplay::DisplayEventsProgressBar($id_user);
464         OcDisplay::DisplayHeroesProgressBar($id_user);
465         echo '</section>';
466     }
467
468     // affiche la barre de progression principale (des tous les objets) d'un
469     // utilisateur
470     static function DisplayMainProgressBar($id_user) {
471         $all_count = OcDao::SelectCountReward(); // retourne le nombre d'objet en
472         tout
473         $user_count = OcDao::SelectCountRewardByIdUser($id_user); // retourne le
474         nombre d'objet d'un utilisateur
475         $display = '';
476
477         $display .= '<div>';
478         $display .= '<div class="flex_row">';
479         $display .= '<h2>All rewards</h2>';
480         $display .= '<h2>' . $user_count . '/' . $all_count . '</h2>';
481         $display .= '</div>';
482         $display .= OcDisplay::GetProgressBar($all_count, $user_count, 100, 30);
483         $display .= '</div>';
484         echo $display;
485     }
486
487     // affiche les barres de progression des evenements d'un utilisateur
488     static function DisplayEventsProgressBar($id_user) {
489         $array_events_count = OcDao::SelectCountRewardEvents(); // retourne le
490         nombre d'objets de chaque evenement
491         $array_events_user_count = OcDao::SelectCountRewardEventsByIdUser($id_user);
492         // retourne le nombre d'objets de l'utilisateur pour chaque evenement
493         $display = '';
494
495         $display .= '<div>';
496         $display .= '<h2>Events</h2>';
497         $display .= '<div class="flex_row">';
498         for ($i=0; $i < count($array_events_count); $i++) {
499             $display .= '<div style="width:48%;>';
500             $display .= '<div class="flex_row">';
501             $display .= '<h4>' . $array_events_count[$i]['name'] . '</h4>';
502             $display .= '<h4>' . $array_events_user_count[$i]['c'] . '/' .
503             $array_events_count[$i]['c'] . '</h4>';
504             $display .= '</div>';
505             $display .= OcDisplay::GetProgressBar($array_events_count[$i]['c'],
506             $array_events_user_count[$i]['c'], 100, 25);
507             $display .= '</div>';
508         }
509         $display .= '</div>';
510         $display .= '</div>';
511         echo $display;
512     }
513
514     // affiche les barres de progression des heros d'un utilisateur
515     static function DisplayHeroesProgressBar($id_user) {
516         $array_heroes_count = OcDao::SelectCountRewardHeroes(); // retourne le
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\class\class.oc_display.php                               lundi 19 juin 2017 11:04
                                                    nombre d'objets de chaque hero
510      $array_heroes_user_count = OcDao::SelectCountRewardHeroesByIdUser($id_user);
      // retourne le nombre d'objets de l'utilisateur pour chaque hero
511      $display = '';
512
513      $display .= '<div>';
514      $display .= '<h2>Heroes</h2>';
515      $display .= '<div class="flex_row">';
516      for ($i=0; $i < count($array_heroes_count); $i++) {
517          $display .= '<div style="width:32%;">';
518          $display .= '<div class="flex_row">';
519          $display .= '<h4>' . $array_heroes_count[$i]['name'] . '</h4>';
520          $display .= '<h4>' . $array_heroes_user_count[$i]['c'] . '/' .
521          $array_heroes_count[$i]['c'] . '</h4>';
522          $display .= '</div>';
523          $display .= OcDisplay::GetProgressBar($array_heroes_count[$i]['c'],
524          $array_heroes_user_count[$i]['c'], 100, 15);
525          $display .= '</div>';
526      }
527      echo $display;
528  }
529
530  // retourne une chaine pour afficher une barre de progression grace aux
parametre de la fonction
531      // max_value est la valeur maximum de la progress bar
532      // current_value est la valur actuelle de la barre
533      // width_percent est la largeur de la barre en pourcentage
534      // height_px est la hauteur de la barre en pixel
535  static function GetProgressBar($max_value, $current_value, $width_percent,
536  $height_px){
537      $display = '';
538      $display .= '<div class="bar_ext" style="width: ' . $width_percent . '%;
539      height: ' . $height_px . 'px; border: 1px solid black;">';
540      $display .=     '<div style="width: ' . ($current_value / $max_value) * 100
541      . '%; background-color: black; height: ' . $height_px . 'px;">';
542      $display .=     '</div>';
543      $display .= '</div>';
544  }
545  ?>
```

Fichier de config base de données

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\config\config_db.php                               lundi 19 juin 2017 11:07
1  <?php
2  /*
3   *      Auteur: Sven Wikberg
4   *      Date: 19/06/2017
5   *      Description: D'affichage des donnees
6  */
7
8  define('DB_HOST', 'localhost');
9  define('DB_NAME', 'overwatchcollection');
10 define('DB_USER', 'oc_admin');
11 define('DB_PWD', 'overwatch123');
12 ?>
```

Style du site web (CSS)

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css                               lundi 19 juin 2017 11:09
1  /*
2   * Auteur: Sven Wikberg
3   * Date: 19/06/2017
4   * Description: Page de style du site
5   */
6 @font-face {
7   font-family: "BigNoodle";
8   src: url('../font/BigNoodleToo.ttf');
9 }
10 @font-face {
11   font-family: "BigNoodle";
12   font-style: italic;
13   src: url('../font/BigNoodleTooOblique.ttf');
14 }
15 @font-face {
16   font-family: "Futura";
17   src: url('../font/Futura.ttf');
18 }
19
20 body{
21   font-family: "BigNoodle", Helvetica, Arial, sans-serif;
22   max-width: 900px;
23   margin-left: auto;
24   margin-right: auto;
25 }
26
27 h1{
28   color: #FF9C1E;
29   font-size: 40px;
30   font-style: italic;
31 }
32
33 h2{
34   font-size: 30px;
35   font-style: italic;
36 }
37
38 h3{
39   font-size: 25px;
40   font-style: italic;
41 }
42
43 h4{
44   font-size: 25px;
45 }
46
47 p{
48   font-family: "Futura", Helvetica, Arial, sans-serif;
49   text-align: justify;
50   font-style: normal;
51   font-weight: normal;
52 }
53
54 footer{
55   padding: 5px;
56   margin: 0px;
57   margin-top: 30px;
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css                                         lundi 19 juin 2017 11:09
58     background-color: #cccccc;
59     border-radius: 15px;
60   }
61 footer p{
62   margin: 0px;
63 }
64
65 header ul{
66   padding: 30px 20px;
67   margin: 0px;
68   list-style-type: none;
69   display: flex;
70   flex-direction: row;
71   justify-content: space-between;
72   flex-wrap: wrap;
73   background-color: #cccccc;
74   border-radius: 15px;
75 }
76
77 header ul li{
78   font-size: 30px;
79 }
80
81 header ul li a{
82   text-decoration: none;
83   color: #606060;
84   padding: 5px;
85 }
86
87 header ul li a:hover{
88   color: #FF9C1E;
89   cursor: pointer;
90   background-color: #a0a0a0;
91   border-radius: 5px;
92 }
93
94 .flex_row{
95   display: flex;
96   flex-direction: row;
97   justify-content: space-between;
98   flex-wrap: wrap;
99 }
100
101 section#index li{
102   font-size: 22px;
103 }
104
105 section#heroes{
106   display: flex;
107   flex-direction: row;
108   justify-content: space-between;
109   flex-wrap: wrap;
110 }
111
112 section#heroes table a div{
113   font-size: 22px;
114 }
```

C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css

lundi 19 juin 2017 11:09

```
115 section#heroes table a img{  
116     border-radius: 5px;  
117 }  
118  
119 section#heroes table td{  
120     border-radius: 5px;  
121 }  
122  
123 section#heroes table td:hover{  
124     background-color: #FF9C1E;  
125 }  
126  
127 section#hero_info div{  
128     display: flex;  
129     flex-direction: row;  
130     justify-content: space-between;  
131     flex-wrap: wrap;  
132 }  
133  
134 section#hero_abilities{  
135     margin-top: 50px;  
136 }  
137  
138 section#hero_abilities table tr td{  
139     border: 3px solid black;  
140     border-radius: 10px;  
141 }  
142  
143  
144 section#rewards{  
145     margin-top: 50px;  
146     display: flex;  
147     flex-direction: row;  
148     justify-content: space-between;  
149     flex-wrap: wrap;  
150 }  
151  
152 section#rewards h2{  
153     margin: 0;  
154 }  
155  
156 section#rewards h3{  
157     margin: 10px 0px;  
158 }  
159  
160 section#rewards ul{  
161     height:200px;  
162     width:170px;  
163     overflow:hidden;  
164     overflow-y:auto;  
165 }  
166  
167 section#rewards div{  
168     display: flex;  
169     flex-direction: column;  
170 }  
171
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css                                         lundi 19 juin 2017 11:09
172    section#rewards div.rewards_type{
173        border: 3px solid black;
174        border-radius: 10px;
175        margin-top: 10px;
176    }
177
178    section#rewards div div{
179        display: flex;
180        flex-direction: row;
181        justify-content: space-between;
182        flex-wrap: wrap;
183    }
184
185    section#rewards div div div{
186        display: flex;
187        flex-direction: column;
188    }
189
190    section#rewards div div div h3.Common{
191        color: black;
192    }
193
194    section#rewards div div div h3.Rare{
195        color: #01C2FD;
196    }
197
198    section#rewards div div div h3.Epic{
199        color: #FE01FE;
200    }
201
202    section#rewards div div div h3.Legendary{
203        color: #FF9C1E;
204    }
205
206    section#rewards div div div li{
207        font-size: 22px;
208        border-radius: 3px;
209    }
210
211    section#events{
212        margin-top: 30px;
213    }
214
215    section#events table a div{
216        font-size: 25px;
217        font-style: italic;
218    }
219
220    section#events table tr td{
221        padding: 10px;
222        border-radius: 10px;
223    }
224
225    section#events table tr td:hover{
226        background-color: #FF9C1E;
227    }
228
```

```
C:\Users\wikbergs_info\Desktop\OverwatchCollection\web\css\style-main.css                                         lundi 19 juin 2017 11:09
229  section#events table tr td img{
230      border-radius: 10px;
231  }
232
233  section#event_info div{
234      display: flex;
235      flex-direction: row;
236      justify-content: space-between;
237      flex-wrap: wrap;
238  }
239
240  section#account_info div#title{
241      display: flex;
242      flex-direction: row;
243      justify-content: flex-start;
244      align-items: center;
245      flex-wrap: wrap;
246  }
247
248  section#account_stats div.bar_ext{
249      border-radius: 5px;
250  }
251
252  section#admin table ul{
253      margin: 10px;
254  }
255
256  section#admin table ul li{
257      font-family: "Futura", Helvetica, Arial, sans-serif;
258      font-size: 18px;
259  }
```