

```
1 <?php
2 //Fichier contenant les fonctions
3 include_once '../php/functions.php';
4
5 //Résultat à retourner
6 $res = array();
7
8 //Récupérer les données
9 $fav['idUser'] = filter_input(INPUT_POST, 'idUser', FILTER_VALIDATE_INT);
10 $fav['idImage'] = filter_input(INPUT_POST, 'idImage', FILTER_SANITIZE_STRING);
11
12 //Vérifier si le favoris existe
13 $res = checkFavExist($fav);
14
15 //Le favoris existe
16 if($res !== false){
17     $res = true;
18 }
19
20 //JSON
21 header('Content-type: application/json');
22 echo json_encode($res);
23 ?>
24
```

```
1 <?php
2 //Constantes
3 //Connexion à la db
4 define("DB_HOST", "yd6zf.myd.infomaniak.com");
5 define("DB_NAME", "wpheaven");
6 define("DB_USER", "user");
7 define("DB_PASSWORD", "super2012");
8 //Lien pour les détails d'une image de Wallhaven
9 define("URL_API_DETAILS_WP", "https://wallhaven.cc/api/v1/w/");
10
11 //Connecteur pour ouvrir la connexion à la DB
12 function connecteur(){
13     static $dbc = null;
14
15     if ($dbc == null) {
16         try{
17             $dbc = new PDO('mysql:host=' . DB_HOST . ';dbname=' . DB_NAME,
18 DB_USER, DB_PASSWORD);
19         } catch (Exception $e) {
20             echo 'Erreur : ' . $e->getMessage() . '<br/>';
21             echo 'N° : ' . $e->getCode();
22             die('Could not connect to MySQL');
23         }
24     }
25     return $dbc;
26 }
27 ?>
28
```

```
1 <?php
2 //Fichier contenant le connecteur pour la DB
3 include_once 'dbconnect.php';
4
5 //Début des fonctions SQL
6
7 //Fonctions SELECT
8
9 //Récupère le salt de l'utilisateur
10 function getSaltUser($pseudo){
11     $req = "SELECT salt FROM `user`
12     WHERE pseudo = :pseudo";
13     $sth = connecteur()->prepare($req);
14     $sth->execute(array(':pseudo' => $pseudo));
15     return $sth->fetch(PDO::FETCH_ASSOC)['salt'];
16 }
17
18 //Essaie de connecter l'utilisateur avec les informations de connexion
19 //données
20 function login($user){
21     $req = "SELECT idUser
22     FROM user
23     WHERE pseudo = :pseudo
24     AND pwd = :pwd";
25     $sth = connecteur()->prepare($req);
26     $sth->execute(array(':pseudo' => $user['pseudo'], ':pwd' =>
27 $user['pwd']));
28     $array = $sth->fetch(PDO::FETCH_ASSOC);
29     return ($array == false ? false : $array['idUser']);
30 }
31
32 //Checker que l'email donné n'existe pas déjà
33 function checkEmailNotExist($email){
34     $req = "SELECT email
35     FROM user
36     WHERE email = :email";
37     $sth = connecteur()->prepare($req);
38     $sth->execute(array(':email' => $email));
39     $array = $sth->fetch(PDO::FETCH_ASSOC);
40     return $array === false;
41 }
42
43 //Checker que le pseudo donné n'existe pas déjà
44 function checkPseudoNotExist($pseudo){
45     $req = "SELECT pseudo
46     FROM user
47     WHERE pseudo = :pseudo";
48     $sth = connecteur()->prepare($req);
49     $sth->execute(array(':pseudo' => $pseudo));
50     $array = $sth->fetch(PDO::FETCH_ASSOC);
51     return $array === false;
52 }
53
54 //Checker si l'image donnée est dans les favoris de l'utilisateur
55 function checkFavExist($fav){
```

```

54     $req = "SELECT idFavorite
55     FROM `favorite`
56     WHERE idImage = :idImage
57     AND idUser = :idUser";
58     $sth = connecteur()->prepare($req);
59     $sth->execute(array(':idImage' => $fav['idImage'], ':idUser' =>
60 $fav['idUser']));
61     $array = $sth->fetch(PDO::FETCH_ASSOC);
62     return ($array == false ? false : $array['idFavorite']);
63 }
64 //Récupérer les favoris de l'utilisateur
65 function getFavUser($idUser){
66     $req = "SELECT idImage
67     FROM `favorite`
68     WHERE idUser = :idUser
69     ORDER BY dateFavorite";
70     $sth = connecteur()->prepare($req);
71     $sth->execute(array(':idUser' => $idUser));
72     return $sth->fetchAll(PDO::FETCH_ASSOC);
73 }
74
75 //Fin fonctions SELECT
76
77 //Fonctions CREATE
78
79 //Création d'un utilisateur
80 function createUser($user){
81     $req = "INSERT INTO `user` (`idUser`, `pseudo`, `email`, `pwd`, `salt`)
82     VALUES (null, :pseudo, :email, :pwd, :salt)";
83     $sth = connecteur()->prepare($req);
84     $sth->execute(array(':pseudo' => $user['pseudo'], ':email' =>
85 $user['email'],
86 ':pwd' => $user['pwd'], ':salt' => $user['salt']));
87     return connecteur()->lastInsertId();
88 }
89
90 //Ajout d'un favoris
91 function addFav($fav){
92     $req = "INSERT INTO `favorite` (`idFavorite`, `idImage`, `idUser`)
93     VALUES (null, :idImage, :idUser)";
94     $sth = connecteur()->prepare($req);
95     $sth->execute(array(':idImage' => $fav['idImage'], ':idUser' =>
96 $fav['idUser']));
97     return connecteur()->lastInsertId();
98 }
99
100 //Fin fonctions CREATE
101
102 //Fonctions EDIT
103
104 //Fin fonctions EDIT
105
106 //Fonctions DELETE

```

```
106 //Suppression d'un favoris
107 function delFav($fav){
108     $req = "DELETE FROM `favorite` WHERE idImage = :idImage AND idUser =
:idUser";
109     $sth = connecteur()->prepare($req);
110     $sth->execute(array(':idImage' => $fav['idImage'], ':idUser' =>
$fav['idUser']));
111 }
112
113 //Fin fonctions DELETE
114
115 ?>
116
```

```
1 <?php
2 //Fichier contenant les fonctions
3 include_once '../php/functions.php';
4
5 //Résultat à retourner
6 $res = array();
7
8 //Récupérer les données
9 $idUser = filter_input(INPUT_POST, 'idUser', FILTER_VALIDATE_INT);
10
11 $res = getFavUser($idUser);
12 $currentFav = array();
13
14 for ($i=0; $i < count($res); $i++) {
15     //Récupérer les infos du wallpaper
16     $currentFav = json_decode(file_get_contents(URL_API_DETAILS_WP . $res[$i]
17 ['idImage']))->data;
18     //Supprimer les champs inutiles
19     unset($currentFav->uploader);
20     unset($currentFav->tags);
21     //L'ajouter au résultat
22     $res[$i] = $currentFav;
23 }
24
25 //JSON
26 header('Content-type: application/json');
27 echo json_encode($res);
28 ?>
```

```
1 <?php
2 //Fichier contenant les fonctions
3 include_once '../php/functions.php';
4
5 //Résultat à retourner
6 $res = array();
7
8 //Récupérer les données
9 $user['pseudo'] = filter_input(INPUT_POST, 'pseudo', FILTER_SANITIZE_STRING);
10 $salt = getSaltUser($user['pseudo']);
11
12 if($salt == null){
13     $res = false;
14 }else{
15     $user['pwd'] = hash('sha256', $salt . filter_input(INPUT_POST, 'pwd',
16 FILTER_SANITIZE_STRING));
17     //Essaie de connecter l'utilisateur
18     $res = login($user);
19 }
20
21 //JSON
22 header('Content-type: application/json');
23 echo json_encode($res);
24 ?>
25
```

```
1 <?php
2 //Fichier contenant les fonctions
3 include_once '../php/functions.php';
4
5 //Résultat à retourner
6 $res = array('done' => false, 'msg' => false);
7
8 //Récupérer les données
9 $user['pseudo'] = filter_input(INPUT_POST, 'pseudo', FILTER_SANITIZE_STRING);
10 $user['email'] = filter_input(INPUT_POST, 'email', FILTER_SANITIZE_STRING);
11 $user['pwd'] = filter_input(INPUT_POST, 'pwd', FILTER_SANITIZE_STRING);
12
13 //Vérifier que l'email existe pas
14 if(checkEmailNotExist($user['email'])) {
15     //Vérifier que le pseudo existe pas
16     if(checkPseudoNotExist($user['pseudo'])) {
17         //Créer le salt et le mdp crypté
18         $user['salt'] = hash('sha256', random_bytes(100));
19         $user['pwd'] = hash('sha256', $user['salt'] . $user['pwd']);
20
21         //Créer l'user
22         $res['msg'] = createUser($user);
23
24         //Tout s'est bien passé
25         $res['done'] = true;
26     } else {
27         $res['msg'] = "Username is already in use";
28     }
29 } else {
30     $res['msg'] = "The email is already in use";
31 }
32
33 //JSON
34 header('Content-type: application/json');
35 echo json_encode($res);
36 ?>
37
```



```
1 <?php
2 //Fichier contenant les fonctions
3 include_once '../php/functions.php';
4
5 //Résultat à retourner
6 $res = array();
7
8 //Récupérer les données
9 $fav['idUser'] = filter_input(INPUT_POST, 'idUser', FILTER_VALIDATE_INT);
10 $fav['idImage'] = filter_input(INPUT_POST, 'idImage', FILTER_SANITIZE_STRING);
11
12 //Vérifier si le favoris existe
13 $res = checkFavExist($fav);
14
15 //Le favoris n'existe pas
16 if($res === false){
17     addFav($fav);
18     $res = true;
19 }else{
20     delFav($fav);
21     $res = false;
22 }
23
24 //JSON
25 header('Content-type: application/json');
26 echo json_encode($res);
27 ?>
28
```