

MATHYS POMIER

MAXIME VITSE

LOÏC LAMOTE



Sommaire

I. Présentation du projet « Bet4Gifts »

II. Analyse du besoin

III. Répartition des tâches

IV. Réalisation

▶ Première partie : inscription

▶ Deuxième partie : confirmation du compte

V. Diffusion du projet

VI. Perspectives

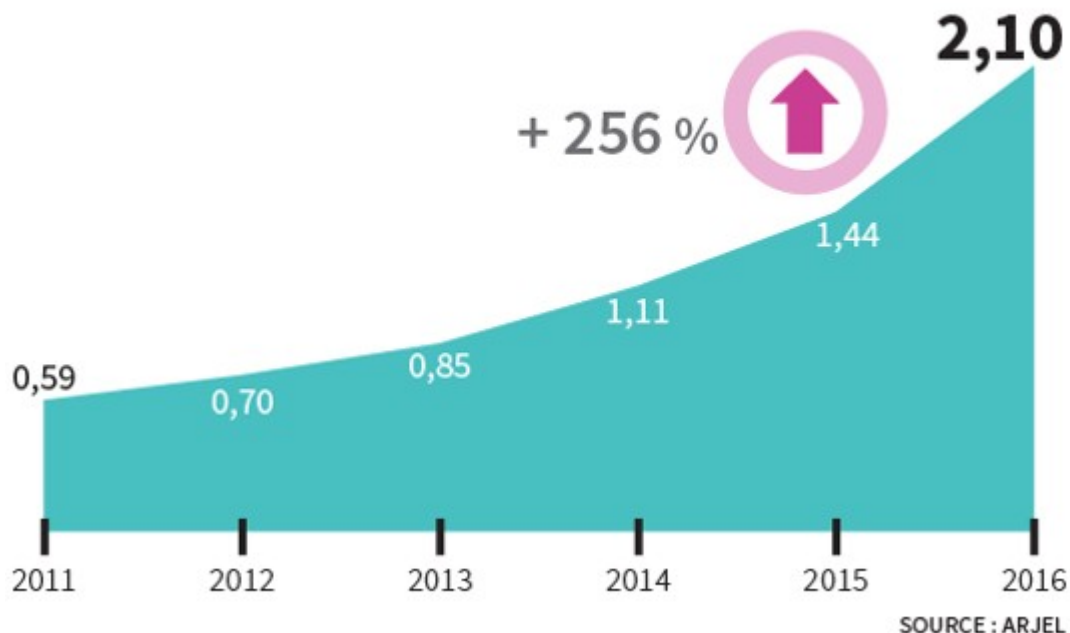
VII. Sources

VIII. Annexes

I. Présentation du projet «Bet4Gifts»

Le marché des paris sportifs en ligne est en pleine expansion depuis les années 2010-2011.

ÉVOLUTION DES MISES SUR LES PARIS SPORTIFS
EN LIGNE EN MILLIARDS D'EUROS



Une manne financière qui peut mener certains joueurs à de véritables problèmes d'addiction et les conduire au surendettement. De nombreux articles sur le web témoignent de cette addiction. Les paris étant réalisés sur internet, le fait d'être seul, loin du regard des autres, peut confiner le joueur dans une atmosphère irréelle, où il n'a plus la notion du temps passé à jouer, de l'argent dépensé et perdu. La possibilité de jouer sur plusieurs sites en même temps et la dématérialisation de l'argent misé augmentent également le risque de perte de contrôle. Dans le cas particulier des paris en direct, l'instantanéité des paris peut provoquer des sensations fortes qui peuvent amener le joueur à perdre le contrôle des sommes dépensées.

Le ministère de la santé publique a bien réalisé l'impact alarmant de ces nouvelles plateformes de jeux d'argent et propose d'ailleurs un forum et une plate-forme d'écoute disponible 7 jours sur 7 aux personnes ayant un problème de jeu excessif.

Notre idée est donc de proposer une plate-forme gratuite de paris-sportifs pour montrer aux joueurs et à leur entourage, le risque encouru en jouant leur argent sur de tels sites. Certains joueurs peuvent penser qu'en consultant de nombreux pronostics et en développant une connaissance fine dans le domaine sportif, ils auront plus de chances de gagner. Mais ce n'est pas le cas, nous souhaitons donc montrer l'importance du hasard tout en proposant un site plaisant et ludique. Notre site a pour but ultime de faire réaliser l'engrenage dans lequel certains joueurs ont été pris pour leur permettre d'arrêter avant une dette financière trop importante.

Nous avons aussi très bien réalisé le plaisir qu'apporte les paris sportifs et l'excitation que peuvent ressentir les joueurs durant les matchs où ils ont parié. C'est pourquoi nous souhaitons proposer la possibilité de gagner des lots en atteignant une certaine somme de points.

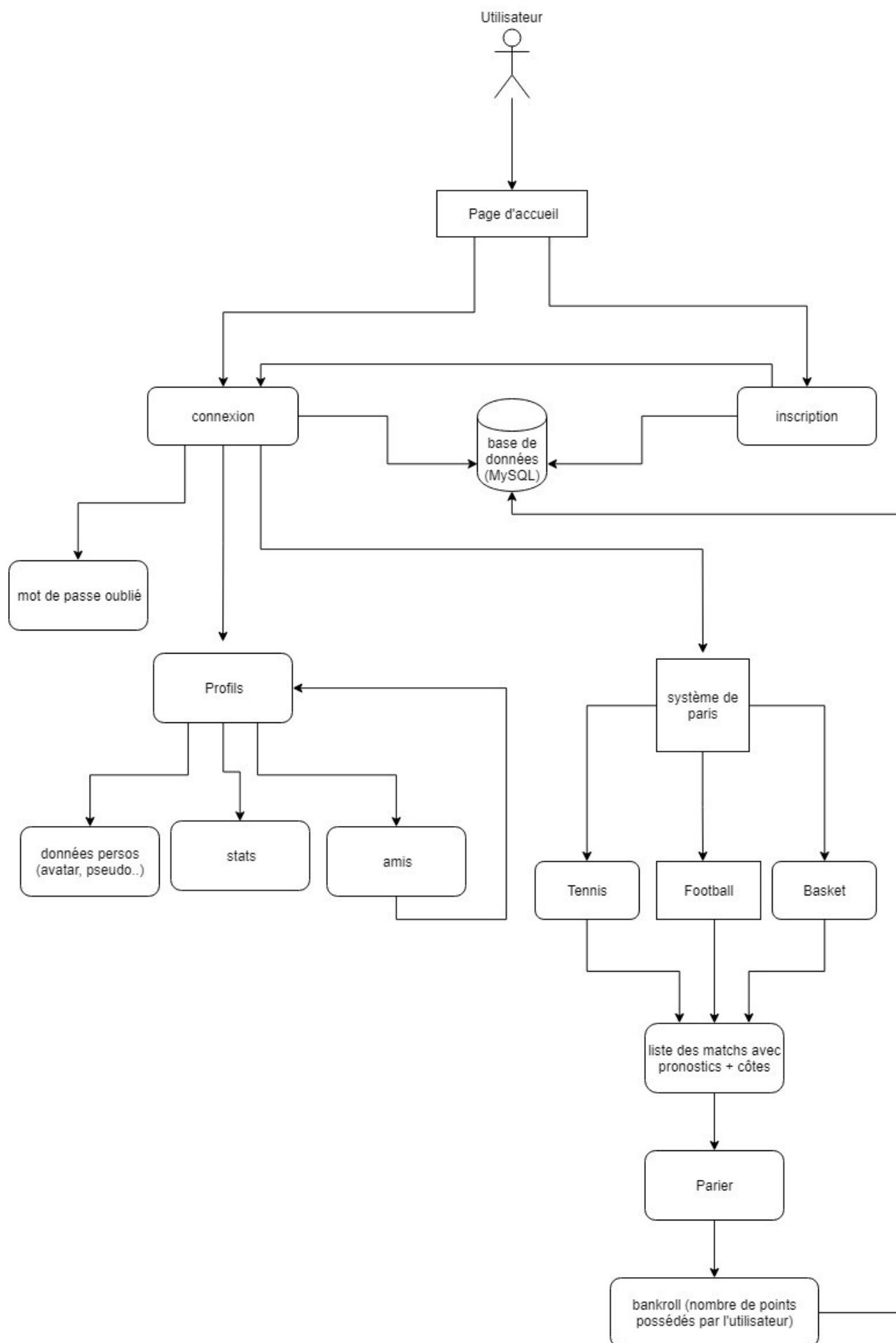
«Les paris sportifs consistent à miser de l'argent sur l'issue d'une rencontre sportive ou sur les événements qui la jalonnent (nombre de buts, score à un moment donné, fautes, etc.). Ils supposent une part de hasard et peuvent nécessiter quelques connaissances dans le domaine sportif, même s'il est tout à fait possible de jouer en étant novice. »

II. Analyse du besoin

Pour bien visualiser notre projet dans sa globalité et bien contextualiser les différentes tâches que nous allons devoir réaliser, nous avons fait le choix de créer un UML pour notre projet.

UML, c'est l'acronyme anglais pour « Unified Modeling Language ». On le traduit par « Langage de modélisation unifié ». L'abréviation UML est un **langage visuel** constitué d'un ensemble de schémas, appelés des **diagrammes**, qui donnent chacun une vision différente du projet à traiter. UML nous fournit donc des diagrammes pour représenter le site à développer : son fonctionnement, sa mise en route, les actions susceptibles d'être effectuées par les utilisateurs, etc.

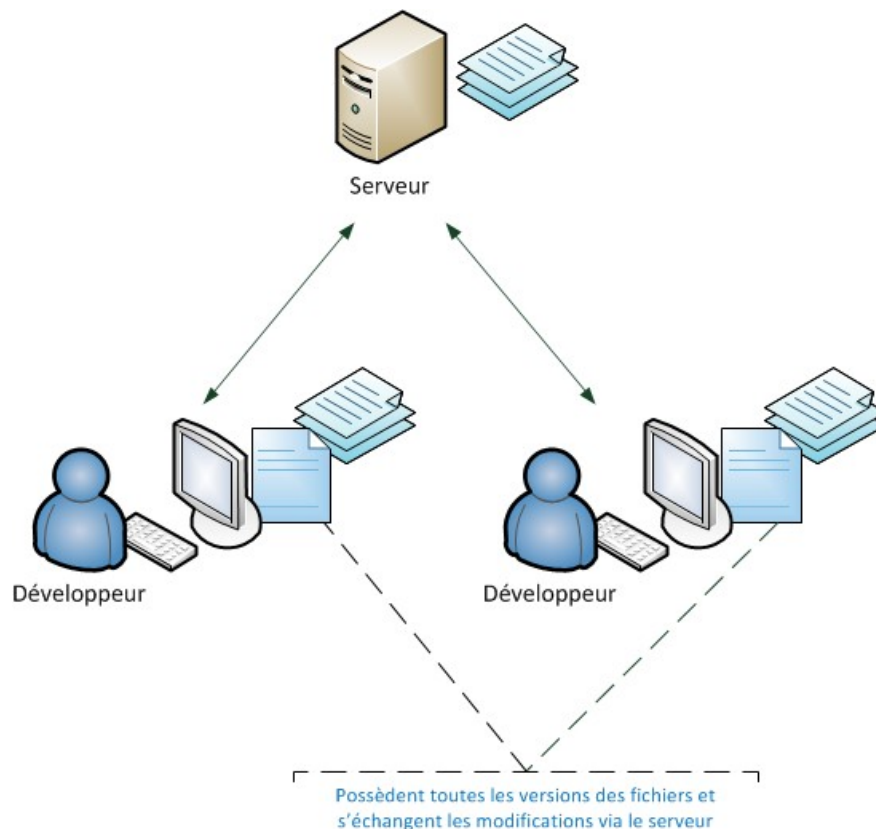
Réaliser ces diagrammes revient donc à modéliser les besoins du site à développer. C'est ce qu'on appelle « **l'analyse des besoins** ».



III. Répartition des tâches

Vitse Maxime	Lamote Loïc	Pomier Mathys
système d'inscription	système de connexion/déconnexion	profil du joueur
système de confirmation de compte	système de mot de passe oublié	système de paris
Boutique de cadeaux	système de changement de mot de passe	design

Nous avons fait le choix de relier notre site hébergé sur Dyjix à un GitHub à l'aide d'un plugin proposé par l'hébergeur. La connexion est faite à l'aide d'une clé SSH. GitHub est un outil gratuit pour héberger du code. Git est un logiciel de contrôle de version, ce qui signifie qu'il gère les modifications d'un projet sans écraser n'importe quelle partie du projet. Nous pouvons donc travailler tous ensemble sur le code sans nous gêner mutuellement.



<http://bet4gifts.web-edu.fr/>

Edit

Add topics

 469 commits

 1 branch

 0 releases

 3 contributors

Branch: master ▾

















New pull request

Create new file

Upload files

Find file

Clone or download ▾

 Elkios update		Latest commit e4e8736 an hour ago
 images	Add files via upload	a month ago
 style	update	an hour ago
 README.md	Update README.md	a day ago
 SSH	Create SSH	a month ago
 bets.php	COMMENTS bets.php	a day ago
 changepassword.php	COMMENTS changepassword.php	a day ago
 confirmation.php	Add files via upload	a day ago
 connexion.php	Update connexion.php	4 hours ago
 deconnexion.php	deconnexion.php	a day ago
 forgottenpassword.php	Update forgottenpassword.php	4 hours ago
 index.php	Update index.php	4 hours ago
 inscription.php	Update inscription.php	4 hours ago
 profil.php	update	an hour ago
 script.js	Update script.js	a month ago
 shop.php	Update shop.php	an hour ago

IV. Réalisation

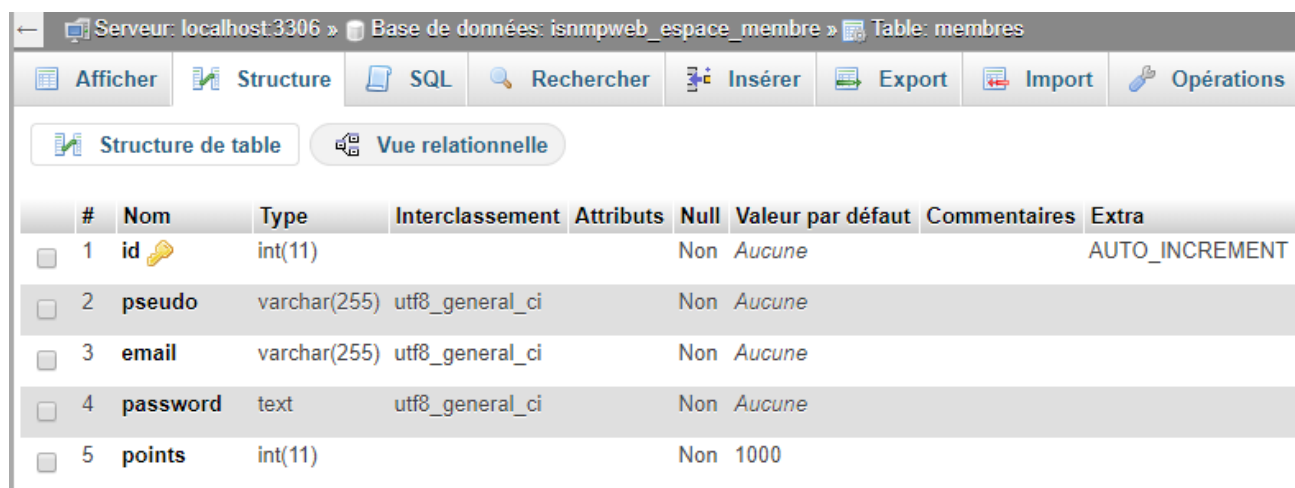
► Première partie

Après la répartition des tâches, je me suis lancé dans la réalisation de ma partie la plus importante pour le début du site : le système d'inscription.

▪ **Premièrement**, j'ai dû me demander de quoi j'avais besoin. Il fallait répertorier nos utilisateurs, il me fallait donc une base de données qui permet de stocker toutes les informations. J'ai donc créé une nouvelle table dans phpMyAdmin nommée « membres » avec 5 champs : id, pseudo, email, password et points.

L'id permet de trier les membres de notre site. Il fallait mettre l'id en index primaire et un auto-increment. L'index primaire sert à accélérer les recherches (quand on veut sélectionner les données par la suite). L'auto-increment est utilisé pour générer un identifiant unique (on rajoute +1 à chaque compte). Par exemple :

id	pseudo
1	weamix
2	Elkios
3	Posseidon



The screenshot shows the phpMyAdmin interface for a database named 'isnmpweb_espace_membre'. The table 'membres' is selected. The 'Structure de table' tab is active, displaying the table's schema. The table has 5 columns: 'id' (int(11), primary key, auto-increment), 'pseudo' (varchar(255), utf8_general_ci), 'email' (varchar(255), utf8_general_ci), 'password' (text, utf8_general_ci), and 'points' (int(11)).

#	Nom	Type	Interclassement	Attributs	Null	Valeur par défaut	Commentaires	Extra
1	id	int(11)			Non	Aucune		AUTO_INCREMENT
2	pseudo	varchar(255)	utf8_general_ci		Non	Aucune		
3	email	varchar(255)	utf8_general_ci		Non	Aucune		
4	password	text	utf8_general_ci		Non	Aucune		
5	points	int(11)			Non	1000		

Pour les champs pseudos et email ce sont des caractères (de 255 caractères maximum) tandis que pour le mot de passe, on choisit un type text parce que selon le hachage/l'encodage du mot de passe qu'on va choisir, cela permettra de mixer les chiffres et les lettres donc le mot de passe sera plus sécurisé.

▪ **Deuxièmement**, il a fallu ensuite réaliser le formulaire d'inscription. On a utilisé la variable « form » avec la méthode « post » ce qui permet de récupérer les données entrées dans le champ de formulaire par l'utilisateur.

On crée les différentes cases nécessaires avec l'élément HTML « input » qui permet à l'utilisateur de rentrer des données : pseudo, email, confirmation, email, mot de passe et

enfin confirmation du mot de passe. J'ai dû ajouter un bouton « submit » qui permet de valider le formulaire.

▪ **Troisièmement**, on s'intéresse à la partie php qui va nous permettre de traiter ce formulaire.

On se connecte à la base de données :

```
1 $bdd = new PDO('mysql:host=localhost;dbname=isnmpweb_espace_membre', 'isnprojet', '01cuz98@');
```

Puis une série de tests et de conditions pour voir si notre utilisateur entre de bonnes valeurs.

1) on vérifie si tous les champs ont bien été complétés (s'ils ne sont pas vides, ils doivent être différents de vide)

```
1 if (isset($_POST['forminscription'])) {  
2   if (!empty($_POST['pseudo']) AND !empty($_POST['email']) AND !empty($_POST['confirmemail'])  
3     AND !empty($_POST['password']) AND !empty($_POST['confirmpassword'])) {  
4
```

2) on stocke l'erreur dans une variable

```
1   else {  
2     $error = "Some fields are empty ! Please complete all fields !";  
3
```

En-dessous du formulaire, on ajoute quelques lignes en php pour afficher le message d'erreur si tous les champs ne sont pas complétés :

```
1 <?php  
2   if (isset($error) OR isset($_SESSION['error'])) {  
3     echo '<span class="errorMessage">'.$error.$_SESSION['error'].'</span>';  
4     $_SESSION["error"] = null;  
5   }  
6 ?>
```

3) on sécurise nos variables et on les simplifie :

```
1 $pseudo = htmlspecialchars($_POST['pseudo']);  
2 $email = htmlspecialchars($_POST['email']);  
3 $confirmemail = htmlspecialchars($_POST['confirmemail']);  
4 $password = sha1($_POST['password']);  
5 $confirmpassword = sha1($_POST['confirmpassword']);
```

Le **sha1** est une fonction de hachage. Si un individu arrivait à entrer dans notre base de données, il ne verrait pas les mots de passe de nos utilisateurs en « clair ».

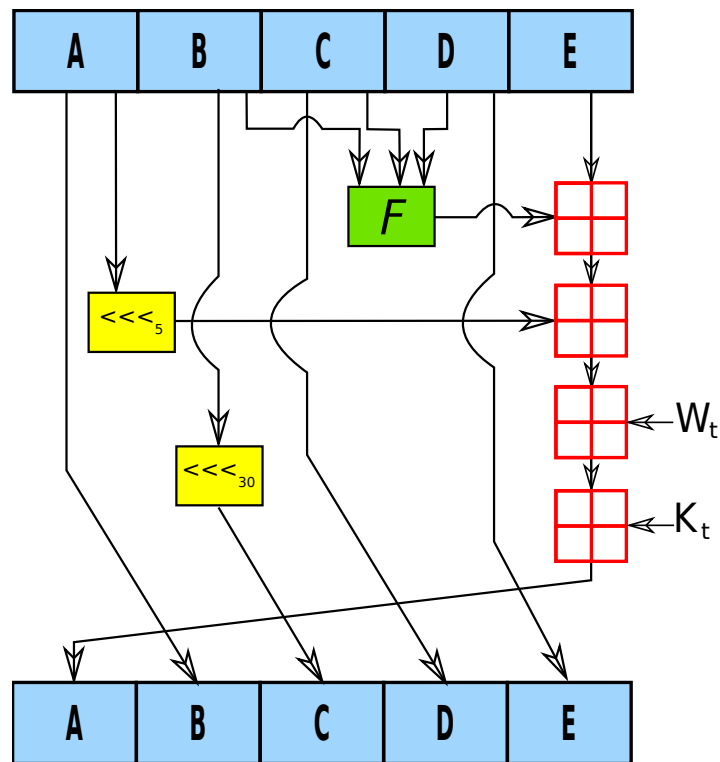


schéma du fonctionnement du SHA1

- 4) On vérifie la concaténation des différentes conditions :
- la taille du pseudo
 - si les mails se correspondent
 - si les mots de passe sont identiques
 - si le pseudo n'est pas déjà utilisé
 - si l'email est sous une forme valide (@ et .com ; .fr ; etc)

```

1  $pseudolength = strlen($pseudo);
2  if ($pseudolength <= 255) {
3      if ($email == $confirmemail) {
4          if(filter_var($email, FILTER_VALIDATE_EMAIL)){
5              $reqemail = $bdd->prepare("SELECT * FROM membres WHERE email = ?");
6              $reqemail->execute(array($email));
7              $emailexist = $reqemail->rowCount();
8              $reqpseudo = $bdd->prepare("SELECT * FROM membres WHERE pseudo = ?");
9              $reqpseudo->execute(array($pseudo));
10             $pseudoexist = $reqpseudo->rowCount();
11             if($pseudoexist == 0){
12                 if($emailexist == 0){
13                     if ($password == $confirmpassword) {
14                         $insertmembre = $bdd->prepare("INSERT INTO membres(pseudo, email, password) VALUES(?, ?, ?)");
15                         $insertmembre->execute(array($pseudo, $email, $password));
16
17                     }
18                     else {
19                         $error = "The passwords don't match !";
20                     }
21                 }else {
22                     $error = "This email adress is already taken ! ";
23                 }
24             }else {
25                 $error = "This pseudo is already taken !";
26             }
27         }
28         else {
29             $error = "Your email adress is invalid !";
30         }
31     }
32     else {
33         $error = "The email addresses don't match !";
34     }
35 }
36 else {
37     $error = "Your pseudo is far too long ! (<255 characters)";
38 }
39 }
40 else {
41     $error = "Some fields are empty ! Please complete all fields !";
42 }
43 }
44 ?>
45

```

5) insertion dans la base de donnée

```

1  $insertmembre = $bdd->prepare("INSERT INTO membres(pseudo, email, password, confirmkey)
   VALUES(?, ?, ?, ?)");
2  $insertmembre->execute(array($pseudo, $email, $password, $key));
3

```

Le système inscription est fonctionnel !

► Deuxième partie

J'ai maintenant voulu faire un système de confirmation par email.

▪ **Premièrement**, nous devons effectuer quelques petites modifications :

On rajoute 2 colonnes à la table membres:

<input type="checkbox"/>	7	confirmkey	varchar(255)	utf8_general_ci	Non	Aucune
<input type="checkbox"/>	8	isconfirm	int(1)		Non	Aucune

On rajoute un script qui génère une clé aléatoirement dans la page inscription.php :

```
1  if ($password == $confirmpassword) {  
2      $lengthkey = 16;  
3      $key = "";  
4      for ($i=0; $i<$lengthkey; $i++) {  
5          $key .= mt_rand(0,9);  
6      }  
}
```

La longueur de la clé est défini à 16 caractères.

On initialise la variable key, puis on crée une boucle for (pour que i soit inférieur ou égale à 16). Enfin, on crée une valeur aléatoire entre 0 et 9.

On modifie l'insertion de l'utilisateur dans la base de données en rajoutant
«confirmkey », « ? » et « \$key »

```
1  $insertmembre = $bdd->prepare("INSERT INTO membres(pseudo, email, password, confirmkey) VALUES(?, ?, ?, ?)");  
2  $insertmembre->execute(array($pseudo, $email, $password, $key));  
3
```

On utilise un template pour l'envoi d'un mail disponible sur internet, qu'on modifie pour notre utilisation :

```
1 $header="MIME-Version: 1.0\r\n"; // on définit le HEADER du mail
2 $header.='From: "Bet4Gifts"<noreply@bet4gifts.web-edu.fr>'. "\n";
3 $header.='Content-Type: text/html; charset="uft-8"'. "\n";
4 $header.='Content-Transfer-Encoding: 8bit';
5 $message='
6 <html>
7     <style>
8         @import url('https://fonts.googleapis.com/css?family=Lato');
9     </style>
10    <body style="margin: 0; padding: 0; background-color: #2c3e50; font-family: 'Lato', sans-serif;">
11        <br>
12        <div style="" align="center"></div>
14        <br>
15        <hr style="width: 25%;">
16        <br>
17        <div align="center">
18            <span style="display: block; color: #FFF; font-size: 25px; font-weight: bold;">Your account was
19            successfully created!</span>
20        </div>
21        <br>
22        <hr style="width: 25%;">
23        <br>
24        <div align="center">
25            <span style="display: block; color: #FFF; font-size: 20px;">Your account was successfully created!</span>
26            <span style="display: block; color: #FFF; font-size: 20px;">You are just one step to confirm your mail !</
27            span>
28        <br>
29        <hr style="width: 25%;">
30        <br>
31        <div align="center">
32            <a style="background-color: #27ae60; border-radius: 20px; padding: 16px 18px; color: #FFF; text-decoration:
33                none; font-size: 15px; text-transform: uppercase; font-weight: bold;" href="https://
34                bet4gifts.web-edu.fr/confirmation.php?pseudo='.urlencode($pseudo).'&key='.$key.'">Click me !</a>
35        </div>
36        <br><br><br>
37    </body>
38 </html>
```

La ligne qui nous intéresse le plus est la suivante :

```
1 <a style="background-color: #27ae60; border-radius: 20px; padding: 16px 18px; color: #FFF; text-decoration: none; font-size:
  15px; text-transform: uppercase; font-weight: bold;" href="https://bet4gifts.web-edu.fr/
  confirmation.php?pseudo='.urlencode($pseudo).'&key='.$key.'">Click me !</a>
```

On crée un lien avec des variables : le pseudo de l'utilisateur, « urlencode » permet de faire transiter les éléments à travers l'url et enfin la clé de l'utilisateur.

- **Deuxièmement**, la mise en place d'une page de confirmation en php, dans laquelle on fait quelques vérifications :

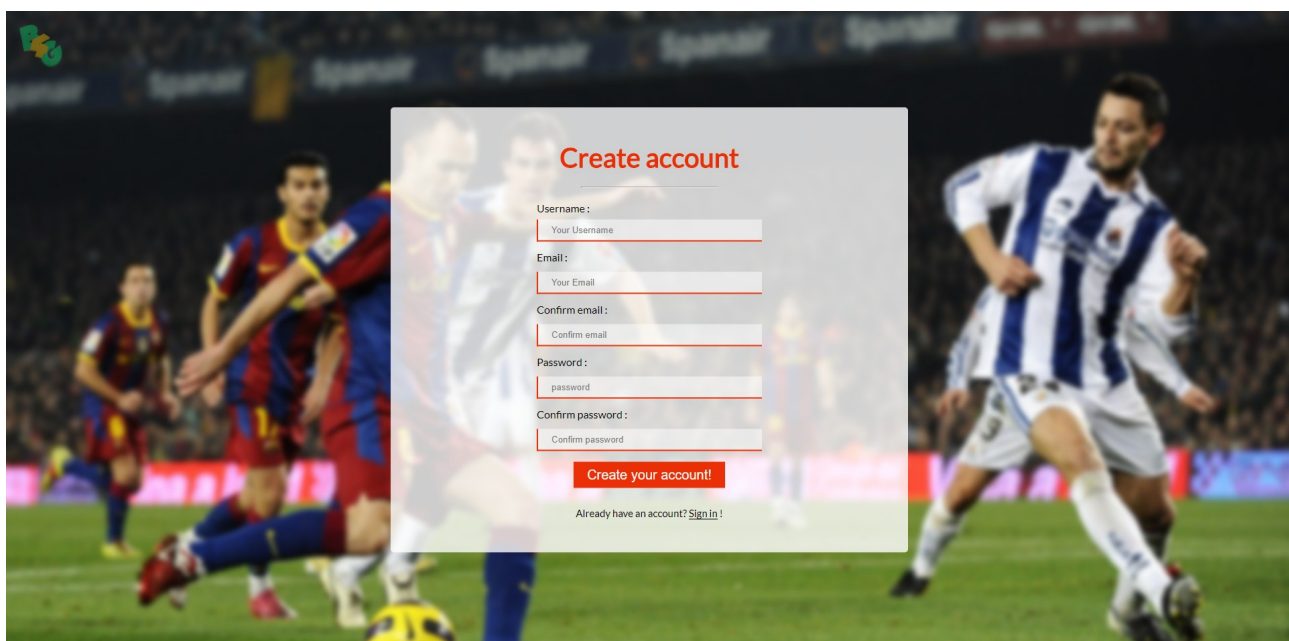
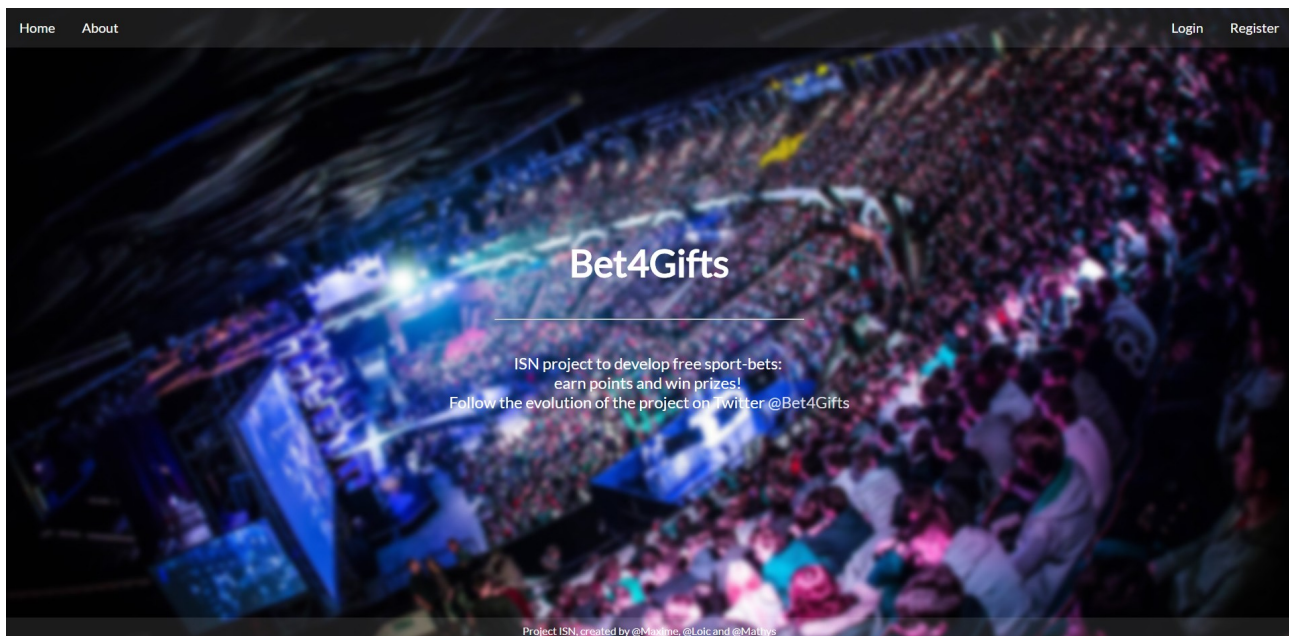
```
1 <?php
2 session_start();
3 $bdd = new PDO('mysql:host=localhost;dbname=ismnpweb_espace_membre', 'isnprojet', '01cuz98@');
4
5 if(isset($_GET['pseudo'], $_GET['key']) AND !empty($_GET['pseudo']) AND !empty($_GET['key'])) {
6 //On vérifie si les informations de l'URL sont remplies
7
8 $pseudo = htmlspecialchars($_GET['pseudo']);
9 //On sécurise les variables
10 $key = htmlspecialchars($_GET['key']);
11 $requirer = $bdd->prepare("SELECT * FROM membres WHERE pseudo = ? AND confirmkey = ?");
12 //On prépare la requête SQL
13 $requirer->execute(array($pseudo, $key));
14 //On l'exécute avec les bonnes valeurs
15 $userexist = $requirer->rowCount();
16 // On regarde le nombre de lignes dans la BDD qui respectent les conditions de la requête
17
18 if($userexist == 1) { //On vérifie si l'utilisateur existe
19 $user = $requirer->fetch(); //Permet de récupérer les informations de la requête
20 if($user['isconfirm'] == 0) { //On vérifie que l'utilisateur n'a pas encore confirmé son compte
21 $updateuser = $bdd->prepare("UPDATE membres SET isconfirm = 1 WHERE pseudo = ? AND confirmkey = ?");
22 $updateuser->execute(array($pseudo,$key)); // on passe l'utilisateur en "confirmé"
23 $_SESSION['valid'] = "Your account has been confirmed !"; //On définit un message de validité
24 header("Location: connexion.php"); //On redirige l'utilistauer vers la page de connexion
25 } else {
26 $_SESSION['error'] = "Your account has already been confirmed !"; //On définit un message d'erreur
27 header("Location: inscription.php");
28 }
29 } else {
30 $_SESSION['error'] = "The user doesn't exist !";
31 header("Location: inscription.php");
32 }
33 }
34 ?>
```

Notre système de confirmation par email est fonctionnel aussi !

V. Diffusion du projet

Notre projet est à retrouver sur bet4gifts.web-edu.fr hébergé chez dyjix.eu qui soutient les projets scolaires. Nous allons essayer de faire connaître notre site à l'aide d'un compte Twitter (@Bet4Gifts).

Aperçu de notre projet :





Login

Email Address :

Password :

LOG IN

Don't have an account? [Create an account](#)

Forgot your password? [Reset password](#)

Login

Email Address :

Password :

LOG IN

Don't have an account? [Create an account](#)

Forgot your password? [Reset password](#)

Create account

Username :

Email :

Confirm email :

Password :

Confirm password :

Create your account!

Already have an account? [Sign in](#) !

Your profile Weamix

Username : Weamix
Email : weamix16@gmail.com
Points : 1001
[Sign out](#)

Matches available

+

team753 VS team159

Categorie : football
Date: 22-05-2018 22:00

[Pariez !](#)

Matches bet upcoming

+

Matches bet in progress

+

Matches bet finished

+

VI. Perspectives

Avec un temps de réalisation plus conséquent il serait tout à fait possible d'améliorer notre projet :

- Mettre en place la boutique de cadeaux
- Améliorer le design du profil
- Système d'amis
- Système de niveaux
- Classement des joueurs

VII. Sources

- Les articles témoignant de l'addiction aux paris sportifs :
 - <https://www.letemps.ch/sport/misere-joueur-compulsif>
 - <https://www.vosgesmatin.fr/edition-d-epinal/2017/01/23/vosges-les-paris-sportifs-ont-detruit-ma-vie>
 - http://www.lemonde.fr/entreprises/article/2016/05/06/l-envolee-des-paris-sportifs-en-ligne_4914638_1656994.html
- Forum et plate-forme d'appel mise en place par le ministère de la santé publique pour les joueurs addictes : <http://www.joueurs-info-service.fr/Forums/Forums-pour-les-joueurs>
- GitHub :
 - <https://www.christopheducamp.com/2013/12/15/github-pour-nuls-partie-1/>
 - <https://openclassrooms.com/courses/gerez-vos-codes-source-avec-git>
- SHA1 : <https://www.plpeeters.com/blog/fr/post/76-stocker-des-mots-de-passe-utilisateur-de-maniere-securisee>

[inscription.php](#)

```
<!DOCTYPE html>

<?php

    session_start(); //permet d'utiliser les variables de SESSION

    $bdd = new PDO('mysql:host=localhost;dbname=isnmpweb_espace_membre', 'isnprojet',
'O1cuz98@'); // connexion à la BDD

    if (isset($_POST['forminscription'])) { // on vérifie si le bouton SUBMIT du formulaire a été
cliqué

        $pseudo = htmlspecialchars($_POST['pseudo']); // on sécurise nos variables et on les simplifie

        $email = htmlspecialchars($_POST['email']);

        $confirmemail = htmlspecialchars($_POST['confirmemail']);

        $password = sha1($_POST['password']); // on crypte le mot de passe

        $confirmpassword = sha1($_POST['confirmpassword']);

        if (!empty($_POST['pseudo']) AND !empty($_POST['email']) AND !
empty($_POST['confirmemail']) AND !empty($_POST['password']) AND !
empty($_POST['confirmpassword'])) { //on vérifie si tous les champs ont bien été complétés (s'ils
ne sont pas vides, ils doivent être différents de vide)

            $pseudolength = strlen($pseudo);

            if ($pseudolength <= 255) { // on vérifie si le pseudo spécifié est inférieur ou égal à 255
caractères

                if ($email == $confirmemail) { // on vérifie que les deux mails sont identiques

                    if(filter_var($email, FILTER_VALIDATE_EMAIL)){ // on vérifie que l'email est sous une
forme valide (@ et .com ; .fr ; etc )

                        $reqemail = $bdd->prepare("SELECT * FROM membres WHERE email = ?"); // on
prépare la requête SQL

                        $reqemail->execute(array($email)); // on l'exécute avec les bonnes valeurs

                        $emailexist = $reqemail->rowCount(); // on regarde le nombre de lignes dans la BDD qui
respectent les conditions de la requête

                        $reqpseudo = $bdd->prepare("SELECT * FROM membres WHERE pseudo = ?");
```



```

$reqpseudo->execute(array($pseudo));

$pseudoexist = $reqpseudo->rowCount();

if($pseudoexist == 0){ // on vérifie si le pseudo n'est pas déjà utilisé

    if($mailexist == 0){ // on vérifie si un compte n'existe pas déjà avec cette adresse mail

        if ($password == $confirmpassword) { // on vérifie si les deux MDP sont identiques

            $lengthkey = 16; // on définit la taille de la clef

            $key = ""; // on initialise la variable key

            for ($i=0; $i<$lengthkey; $i++) {

                $key.= mt_rand(0,9); // on génère 16 fois un nombre aléatoire entre 0 et 9 et on
l'ajoute à la variable $key

            }

            $insertmembre = $bdd->prepare("INSERT INTO membres(pseudo, email, password,
confirmkey) VALUES(?, ?, ?, ?)"); //on prépare la requête SQL

            $insertmembre->execute(array($pseudo, $email, $password, $key)); //on l'exécute avec
les bonnes valeurs (insertion du membre dans la BDD)

            $header="MIME-Version: 1.0\r\n"; // on définit le HEADER du mail

            $header.='From:"Bet4Gifts"<noreply@bet4gifts.web-edu.fr>'. "\n";

            $header.='Content-Type:text/html; charset="uft-8"'. "\n";

            $header.='Content-Transfer-Encoding: 8bit';

            $message='

<html>

<style>

    @import url(\'https://fonts.googleapis.com/css?family=Lato\');

</style>

<body style="margin: 0; padding: 0; background-color:#2c3e50; font-
family: \'Lato\', sans-serif;">

    <br>

    <div style="" align="center"></div>

    <br>

    <hr style="width: 25%;">

    <br>

```

```

        <div align="center">

            <span style="display: block; color: #FFF; font-size: 25px; font-weight:
bold;">Your account was successfully created!</span>

        </div>

        <br>

        <hr style="width: 25%;">

        <br>

        <div align="center">

            <span style="display: block; color: #FFF; font-size: 20px;">Your account was
successfully created!</span>

            <span style="display: block; color: #FFF; font-size: 20px;">You are just one step
to confirm your mail !</span>

            <br>

            <hr style="width: 25%;">

            <br>

        </div>

        <div align="center">

            <br>

            <a style="background-color: #27ae60; border-radius: 20px; padding: 16px 18px;
color: #FFF; text-decoration: none; font-size: 15px; text-transform: uppercase; font-weight: bold;"
href="https://bet4gifts.web-edu.fr/confirmation.php?pseudo='.urlencode($pseudo).'&key='.
$key.">Click me !</a>

        </div>

        <br><br><br>

    </body>

</html>

'; // on définit le message du mail

mail($email, "Confirm your account !", $message, $header); // on envoie un mail avec
les informations précédemment définies

$_SESSION['valid'] = "Your account has been created ! Look at your mails to
confirm ! (and your spams)"; //on définit un message de validité

header("Location: connexion.php"); //on redirige l'utilisateur vers la page de connexion
}

```

```

else {

    $error = "The passwords don't match !"; //on définit un message d'erreur

}

}else {

$error = "This email adress is already taken ! ";

}

}else {

$error = "This pseudo is already taken !";

}

}

else {

$error = "Your email adress is invalid !";

}

}

else {

$error = "The email addresses don't match !";

}

}

else {

$error = "Your pseudo is far too long ! (<255 characters)";

}

}

else {

$error = "Some fields are empty ! Please complete all fields !"; // on stocke l'erreur dans une
variable

}

}

?>

<html lang="fr">

<head>

<meta charset="utf-8">

```

```

<title>Bet4Gifts - Create account</title>

<link rel="icon" type="image/png" href="images/favicon.png" />

<link rel="stylesheet" href="style/style.css">

<link rel="stylesheet" href="style/inscriptionstyle.css">

<link href="https://fonts.googleapis.com/css?family=Lato" rel="stylesheet">

<script type="text/javascript" src="script.js"></script>

</head>

<body onload="resizeHeaderHeight()">

    <a href="index.php" class="logohome">  </a>

    <header class="full-background" id="full-background">

        <div class="container_form_inscription">

            <h2>Create account</h2>

            <hr>

            <form class="" action="" method="post"> <!-- création du formulaire: variable form avec la
méthode post qui permet de récupérer les données entrées dans le champ de formulaire par
l'utilisateur -->

                <label for="pseudo"> Username :

                    <input type="text" name="pseudo" value="<?php if(isset($pseudo)) {echo $pseudo;} ?>"
id="pseudo" placeholder="Your Username">

                </label>

                <label for="email"> Email : </label>

                <input type="email" name="email" value="<?php if(isset($email)) {echo $email;} ?>"
id="email" placeholder="Your Email"> <!-- élément HTML qui permet à l'utilisateur de rentrer des
données dans une case -->

                <label for="confirmemail"> Confirm email : </label>

                <input type="email" name="confirmemail" value="<?php if(isset($confirmemail)) {echo
$confirmemail;} ?>" id="confirmemail" placeholder="Confirm email">

                <label for="password"> Password : </label>

                <input type="password" name="password" value="" id="password"
placeholder="password">

                <label for="confirmpassword"> Confirm password : </label>

                <input type="password" name="confirmpassword" value="" id="confirmpassword"
placeholder="Confirm password">

```


<input type="submit" name="forminscription" value="Create your account!"> <!-- bouton pour valider le formulaire -->

Already have an account? Sign in !

</form>

<?php

if (isset(\$error) OR isset(\$_SESSION['error'])) { // message d'erreur si tous les champs ne sont pas complétés

echo ''.\$error.\$_SESSION['error'].''; // on vérifie si la variable ERROR est SET , si OUI on affiche le message à l'utilisateur

\$_SESSION["error"] = null;

}

if (isset(\$_SESSION['valid'])) {

echo ''.\$_SESSION['valid'].''; // on vérifie si la variable VALID est SET , si OUI on affiche le message à l'utilisateur

\$_SESSION['valid'] = null;

}

?>

</div>

</header>

</body>

</html>

[confirmation.php](#)

```
<?php

session_start();

$bdd = new PDO('mysql:host=localhost;dbname=isnmpweb_espace_membre', 'isnprojet',
'O1cuz98@');

if(isset($_GET['pseudo'], $_GET['key']) AND !empty($_GET['pseudo']) AND !
empty($_GET['key'])) { //On vérifie si les informations de l'URL sont remplies

    $pseudo = htmlspecialchars($_GET['pseudo']); //On sécurise les variables

    $key = htmlspecialchars($_GET['key']);

    $requiser = $bdd->prepare("SELECT * FROM membres WHERE pseudo = ? AND confirmkey =
?"); //On prépare la requête SQL

    $requiser->execute(array($pseudo, $key)); //On l'exécute avec les bonnes valeurs

    $userexist = $requiser->rowCount(); // On regarde le nombre de lignes dans la BDD qui
respectent les conditions de la requête

    if($userexist == 1) { //On vérifie si l'utilisateur existe

        $user = $requiser->fetch(); //Permet de récupérer les informations de la requête

        if($user['isconfirm'] == 0) { //On vérifie que l'utilisateur n'a pas encore confirmé son compte

            $updateuser = $bdd->prepare("UPDATE membres SET isconfirm = 1 WHERE pseudo = ?
AND confirmkey = ?");

            $updateuser->execute(array($pseudo,$key));

            $_SESSION['valid'] = "Your account has been confirmed !"; //On définit un message de
validité

            header("Location: connexion.php"); //On redirige l'utilistauer vers la page de connexion

        } else {

            $_SESSION['error'] = "Your account has already been confirmed !"; //On définit un message
d'erreur

            header("Location: inscription.php");

        }

    } else {

        $_SESSION['error'] = "The user doesn't exist !";

        header("Location: inscription.php");

    }

} ?>
```