

SE7EN

PROJET LICENCE 3 INFORMATIQUE

PROGRAMMATION WEB



SOMMAIRE

INTRODUCTION	3
I) ORGANISATION DU PROJET	5
1. Modélisation	5
2. Mise en place d'une plateforme Git.....	7
3. Structure du projet.....	8
II) INTERACTION BASE DE DONNÉES	10
1. Formulaires d'inscription et de connexion.....	10
2. Rôle de l'administrateur.....	12
a. Suppression utilisateurs.....	12
b. Création et modification de questionnaire.....	14
3. Gestion des scores	18
III) DÉROULEMENT DU JEU	19
1. Principe du moteur de jeu	19
2. Correction du questionnaire	22
CONCLUSION	25

INTRODUCTION

Dans le cadre de notre 6^{ème} semestre d'étude, nous avons eu à développer un jeu interactif à connotation culturelle. Nous avons choisi d'appeler ce projet « Se7en » en référence au nombre de questions composant un questionnaire.

Issus de la même formation (DUT Informatique), nous avons déjà eu à travailler en binôme, ce qui nous a grandement aidé pour s'organiser et se répartir les tâches. Ainsi, nous avons commencé par modéliser notre projet en deux parties : une partie interface et une partie base de données.

Chaque décision a été prise à deux mais nous nous sommes répartis la réalisation :

	Assan	Dinesh
Modélisation	Création base de données	Réalisation maquettes + interfaces
Interface d'authentification	Requêtes AJAX + PHP	Interface + Requêtes PHP
Menu principal		Travail réalisé presque seul
Page des score	Travail réalisé presque seul	
Plateforme d'administration	Travail partagé	Travail partagé
Moteur de jeu	Partie back-end	Partie front-end
Correction questionnaire	Partie front-end	Partie back-end

Nous avons abordé la réalisation de ce projet comme un défi ce qui nous a, à certains moments, poussé à réaliser chacun de notre côté un même module (notamment le moteur de jeu).

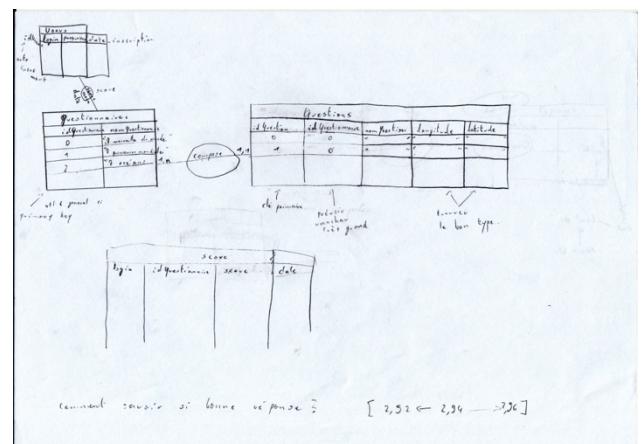
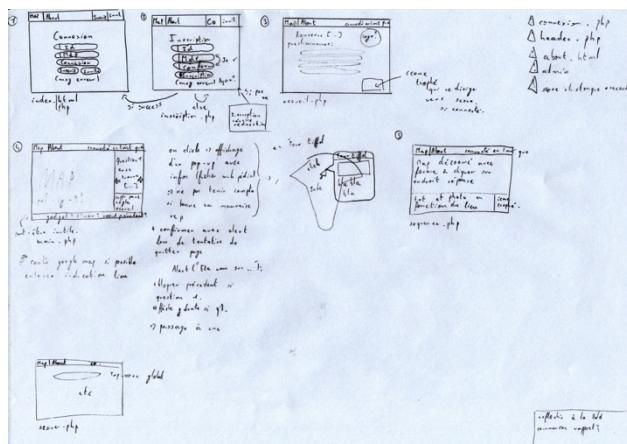
Ainsi, nous avons pris soin de monter en compétences sur les technologies que nous n'avons pas étudiées lors de notre précédente formation (Ajax et Leaflet) car l'acquisition de nouvelles compétences techniques a été notre objectif principal tout au long de ce projet.

L'expérience acquise lors de notre DUT nous a permis de répondre à toutes les impératifs du cahier des charges. Ce rapport détaillera dans une première partie le travail fait en amont du développement puis en seconde partie nous présenterons l'utilité de notre base de données et son utilisation puis nous terminerons par la présentation des pages utilisant la bibliothèque Leaflet : le moteur de jeu et la correction proposée.

I) ORGANISATION DU PROJET

1. Modélisation

Après avoir lu le sujet, nous avons déterminé les points impératifs à développer pour ce projet. Puis, nous avons dessiné des croquis sur feuilles blanche afin d'avoir un fil rouge permettant de répartir les fonctionnalités de notre site.



Maquettes réalisées en amont du développement.

Ces maquettes ont été très importantes pour la réalisation de ce projet, en effet, on remarque que notre interface est similaire à celle prévue.

Nous avons rapidement déterminé une structure pour notre base de données ce qui nous a permis de directement héberger nos données via l'interface PhpMyAdmin de notre serveur web. Ainsi, nous n'avons pas eu besoin d'utiliser de fichiers JSON ou GeoJSON.

La maquette représentant la base de données ne correspond pas exactement à celle qui est utilisée par notre site. Nous avons eu à ajouter quelques attributs au fur du développement mais le nombre de table a été correctement estimé.

Liste des tables composant la BDD du projet

Table ▾	
<input type="checkbox"/>	questionnaires
<input type="checkbox"/>	questions
<input type="checkbox"/>	score
<input type="checkbox"/>	users
4 tables	

SELECT * FROM `users`				
id	login	password	date_inscription	progression
1	admin	admin	2018-03-24 00:55:41	1
2	assan	bafa1b192396b6aceacd52f5708064b1	2018-03-24 00:55:41	1

Structure de la table users

Chaque utilisateur est identifié par une valeur entière ‘id’ correspondant à la clé primaire de la table. Cette valeur est auto-incrémentée ainsi, deux utilisateurs ne peuvent avoir le même id. Le login est unique et sensible à la casse. Le mot de passe est bien crypté en md5 comme souhaité. L’attribut progression permet de débloquer le questionnaire n+1 lorsque le questionnaire n est complété.

SELECT * FROM `questionnaires`		
idQuestionnaire	nomQuestionnaire	statut
1	Les 7 Merveilles du Monde	active
2	Les capitales des 7 pays les plus puissants	active

Structure de la table questionnaires

L’idQuestionnaire est la clé primaire de cette table et le nom est aussi unique. Le statut permet à l’administrateur de bloquer un questionnaire pour le rendre inaccessible à l’utilisateur. Cette fonctionnalité a été implémentée afin de laisser le temps à l’admin de compléter le questionnaire. Ainsi, l’utilisateur ne pourra pas répondre à un questionnaire non complet.

SELECT * FROM `questions`						
idQuestion	idQuestionnaire	nomQuestion	latitude	longitude	description	
1	1	Où se situe la pyramide de Kheops?	29.9789	31.1339	test	
1	2	Où se situe la capitale des Etats-Unis?	38.8951	-77.0364	Washington, la capitale des Etats-Unis, est une vi...	NULL
2	1	Où se situent les jardins suspendus de Babylone?	32.5355	44.4275		
2	2	Où se situe la capitale de la Chine ?	39.9075	116.397	Beijing (ou Pékin), l'immense capitale chinoise, e...	

Structure de la table questions

Un questionnaire est composé de 7 questions. Nous avons donc relié ces deux tables en créant la clé primaire composite (idQuestion, idQuestionnaire). La réponse à une question est sous la forme latitude, longitude. Ces attributs seront expliqués plus en détails dans la partie III 2. Une description est conseillée mais n’est pas obligatoire pour la partie correction.

SELECT * FROM `score`					
id	login	idQuestionnaire	nomQuestionnaire	score	date_partie
6	dinesh	1	Les 7 Merveilles du Monde	0	2018-05-17 09:45:44
6	dinesh	2	Les capitales des 7 pays les plus puissants	30	2018-05-17 09:47:42

Structure de la table score

La clé primaire composite de cette table est composée de 3 attributs ('id', 'idQuestionnaire' et 'date_partie'). On répertorie ainsi le score de chaque utilisateur en fonction du questionnaire. Chaque questionnaire terminé est différent des précédents et des suivants grâce à l'attribut 'date_partie'.

2. Mise en place d'une plateforme Git

L'utilisation d'un logiciel de versionning est indispensable pour tout projet Informatique. C'est pourquoi nous avons ouvert un dépôt GitHub afin d'avoir un cadre de travail optimal.

Branch: master ▾		New pull request	Create new file	Upload files	Find file	Clone or download ▾
121 commits	3 branches	0 releases	2 contributors			
DineshGov suppression fichiers inutiles + modification script insertion sql pou... Latest commit 7650bca 14 seconds ago						
bootstrap	reorganisation des fichiers					2 months ago
css	reorganisation des fichiers					2 months ago
jquery	reorganisation des fichiers					2 months ago
js	details					2 months ago
main	suppression fichiers inutiles + modification script insertion sql pou...					13 seconds ago
database_auth.php	ajout de js et partitioning					2 months ago
database_import.sql	suppression fichiers inutiles + modification script insertion sql pou...					13 seconds ago
database_insertion.sql	suppression fichiers inutiles + modification script insertion sql pou...					13 seconds ago
header.php	gestion du joueur invité, présentation de se7en grâce au bouton about					16 days ago
index.php	changement de noms de fichiers + ajout balises fermantes body et html...					16 days ago
inscription.php	changement de noms de fichiers + ajout balises fermantes body et html...					16 days ago
redirection_connexion.php	changement de noms de fichiers + ajout balises fermantes body et html...					16 days ago
redirection_inscription.php	modif fautes orthographiques et changement nom fichier					7 days ago
redirection_invite.php	upload fichier et formatage nom fichiers fonctionnelle					12 days ago
requete_ajax.php	reorganisation des fichiers					2 months ago

*Aperçu de notre dépôt git.
<https://github.com/DineshGov/MapGame.git>*

3. Structure du projet

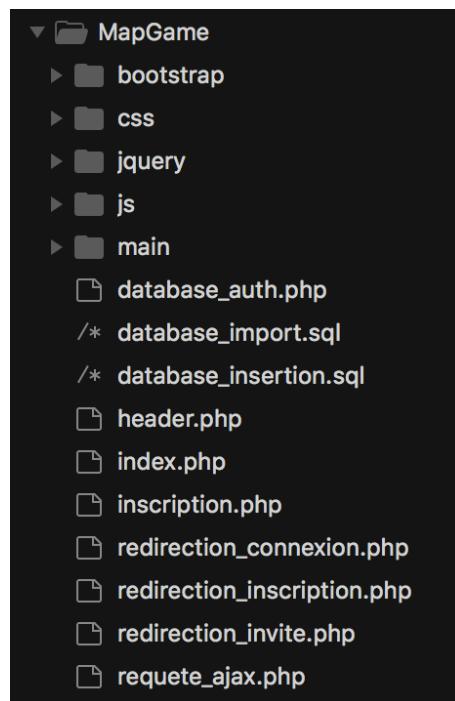
La consigne impose des technologies et les versions de ces dernières. Nous avons pris soin de vérifier que nous répondions à tous ces critères.

```
11  <?php
12  if($page_name=="gestion_questionnaire.php" || $page_name=="jeuLeaflet.php" || $page_name=="correction.php"){
13      echo '<link rel="stylesheet" href="https://unpkg.com/leaflet@1.3.1/dist/leaflet.css"
14          integrity="sha512-Rksm5RenBEKSKFjgI3a41vrjkw4EVPLJ3+0iI65vTjIdo9brLaacEuK0iQ50Fh7c0I1bkDwLqdLw3Zg0cRJAQQ==" crossorigin="" />';
15      echo '<script src="https://unpkg.com/leaflet@1.3.1/dist/leaflet.js" integrity="sha512-
16      Nsx9X4HebavoBvEBuy3I7od5tA0UzAxs+j83KgC8PU0kgB4XiK4Lf4y4cgBtaRJQEIfCW+oC506aPT2L1zw==" crossorigin=""></script>';
17  }
18  ?>
19  <link href="../bootstrap/css/bootstrap.min.css" rel="stylesheet">
20  <script src="../jquery/jquery-3.3.1.js"></script>
<script src="../bootstrap/js/bootstrap.min.js"></script>
```

La version de Bootstrap se vérifie dans le répertoire correspondant.

Nous avons aussi choisi de structurer notre code de manière à placer les fichiers css et js dans des dossiers distincts (css/ et js/). Les bibliothèques jQuery et Bootstrap ont été téléchargées localement.

Une bonne habitude pour un développeur est de fournir un code lisible et facilement compréhensible. Nous avons appliqué ce principe à nos noms de fichiers et les variables du code.

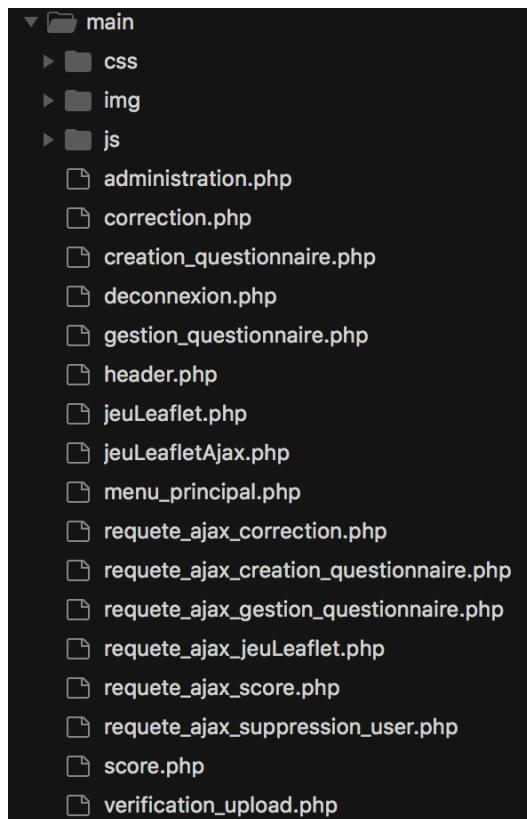


Arborescence du projet.

Cette arborescence comporte le projet dans sa globalité. Les répertoires Bootstrap, jQuery et le fichier database_auth.php sont utilisés dans toutes les pages du projet. On utilise pour cela l'inclusion de code grâce à un fichier header.

Les fichiers database_import.sql et database_insertion.sql servent respectivement à créer la base de données avec ses tables et à insérer des valeurs dans ces dernières.

Les répertoires js/, css/, et les fichiers .php placés dans la racine du projet composent la partie authentification de notre site. Puis, le répertoire main/ comprend l'ensemble des fichiers accessibles à un utilisateur (utilisateur inscrit, invité et administrateur).



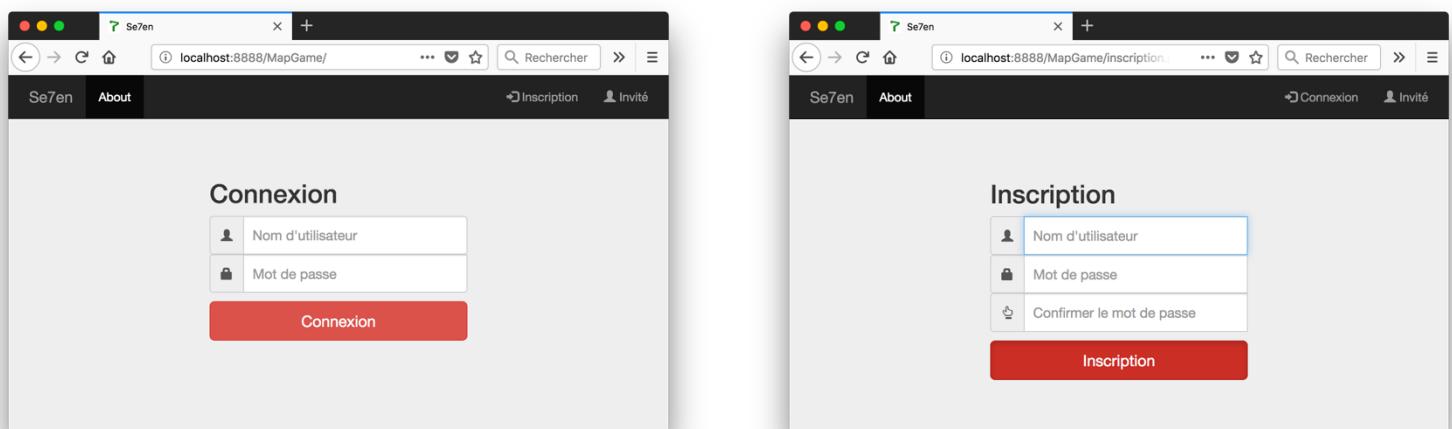
Arborescence du répertoire main/

Ce répertoire à la même structure que la racine du projet. On note la présence d'un répertoire img/ qui comportera les images hébergées par l'admin qui seront utilisées lors de la correction d'un questionnaire.

II) INTERACTION BDD

1. Formulaires d'inscription et de connexion

Pour pouvoir accéder à toutes les fonctionnalités de notre site, l'utilisateur devra s'inscrire et se connecter. Il peut toutefois avoir accès à une interface plus restreinte en se connectant en tant qu'invité ; un message l'invitera à créer un compte pour bénéficier de toutes les fonctionnalités.



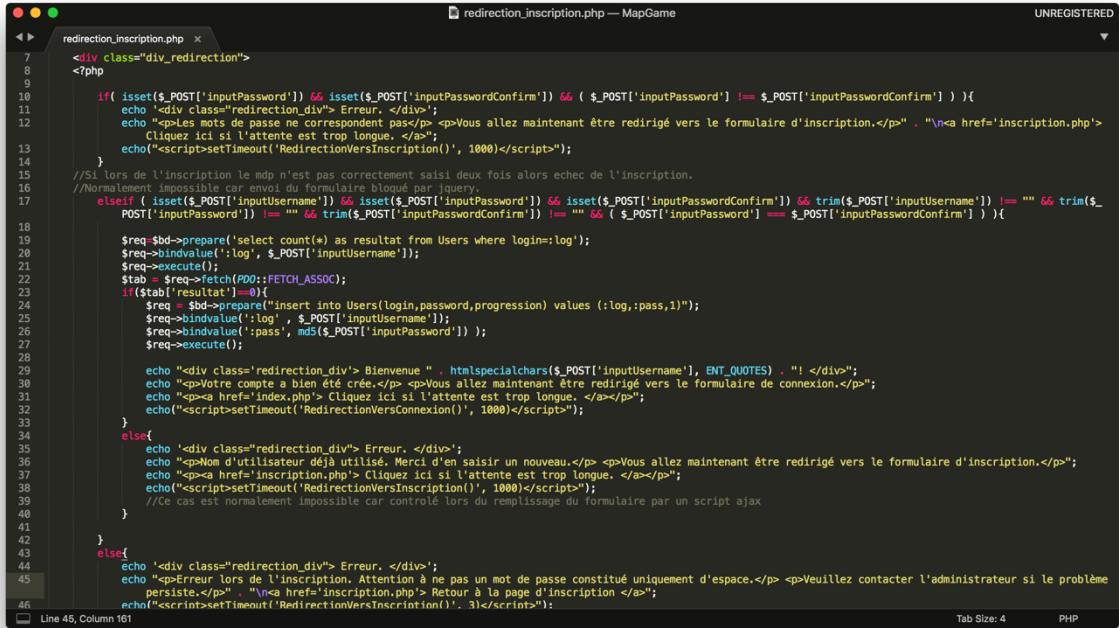
Formulaires d'inscription et de connexion.

À la soumission de ses formulaires, on est conduit vers des pages de transition respectivement : redirection_connexion.php & redirection_inscription.php.

Ces pages comportent le script php permettant de connecter l'utilisateur s'il existe dans la base de données ou de l'inscrire s'il n'existe pas. Bien sûr, nous n'avons pas négliger la partie sécurité de ces formulaires. Ainsi, toute entrée texte de la part de l'utilisateur est vérifiée par la fonction php htmlspecialchars() afin de se protéger contre les attaques XSS.

La page d'inscription permet d'inscrire un utilisateur si le nom d'utilisateur entré n'existe pas dans la base de données. Ce nom est sensible à la casse. Nous avons jugé utile d'améliorer l'ergonomie de l'utilisateur en rajoutant des appels AJAX permettant d'afficher un message et de bloquer la soumission du formulaire si le nom d'utilisateur est déjà pris ou si les deux champs mot de passe ne correspondent pas.

Finalement, les pages de redirection redirigent l'utilisateur en fonction de l'issue des requêtes sql exécutées. Un utilisateur ayant réussi à s'inscrire sera redirigé vers la page de connexion et un utilisateur ayant réussi à se connecter sera redirigé vers le menu_principal situé dans le répertoire main/. Un utilisateur essayant de se connecter avec un nom d'utilisateur inexistant sera redirigé vers le formulaire de connexion.



```

1 // redirection_inscription.php — MapGame
2 <?php
3
4 if( isset($_POST['inputPassword']) && isset($_POST['inputPasswordConfirm']) && ( $_POST['inputPassword'] != $_POST['inputPasswordConfirm'] ) ){
5     echo '<div class="redirection_div"> Erreur. </div>';
6     echo "<p>Les mots de passe ne correspondent pas.</p><p>Vous allez maintenant être redirigé vers le formulaire d'inscription.</p>" . "\n<a href='inscription.php'>
7         Cliquez ici si l'attente est trop longue. </a>" ;
8     echo("<script>setTimeout('RedirectionVersInscription()', 1000)</script>");
9 }
10 //Si lors de l'inscription le mdp n'est pas correctement saisi deux fois alors echec de l'inscription.
11 //Normalement impossible car envoi du formulaire bloqué par jquery.
12 elseif ( isset($_POST['inputUsername']) && isset($_POST['inputPassword']) && isset($_POST['inputPasswordConfirm']) && trim($_POST['inputUsername']) != "" && trim($_POST['inputPassword']) != "" && trim($_POST['inputPasswordConfirm']) != "" && ( $_POST['inputPassword'] == $_POST['inputPasswordConfirm'] ) ){
13
14     $req=$bd->prepare('select count(*) as resultat from Users where login=:log');
15     $req->bindValue(':log', $_POST['inputUsername']);
16     $req->execute();
17     $tab = $req->fetch(PDO::FETCH_ASSOC);
18
19     if($tab['resultat']==0){
20
21         $req = $bd->prepare("insert into Users(login,password,progression) values (:log,:pass,1)");
22         $req->bindValue(':log' , $_POST['inputUsername']);
23         $req->bindValue(':pass' , md5($_POST['inputPassword']));
24         $req->execute();
25
26         echo "<div class='redirection_div'> Bienvenue " . htmlspecialchars($_POST['inputUsername'], ENT_QUOTES) . " ! </div>";
27         echo "<p>Votre compte a bien été créé.</p><p>Vous allez maintenant être redirigé vers le formulaire de connexion.</p>" ;
28         echo "<p><a href='index.php'> Cliquez ici si l'attente est trop longue. </a></p>" ;
29         echo("<script>setTimeout('RedirectionVersConnexion()', 1000)</script>");
30
31     }else{
32
33         echo '<div class="redirection_div"> Erreur. </div>';
34         echo "<p>Nom d'utilisateur déjà utilisé. Merci d'en saisir un nouveau.</p><p>Vous allez maintenant être redirigé vers le formulaire d'inscription.</p>" ;
35         echo "<p><a href='inscription.php'> Cliquez ici si l'attente est trop longue. </a></p>" ;
36         echo "<script>setTimeout('RedirectionVersInscription()', 1000)</script>" ;
37
38         //Ce cas est normalement impossible car contrôlé lors du remplissage du formulaire par un script ajax
39     }
40
41 }
42
43 else{
44
45     echo '<div class="redirection_div"> Erreur. </div>';
46     echo "<p>Erreur lors de l'inscription. Attention à ne pas un mot de passe constitué uniquement d'espace.</p><p>Veuillez contacter l'administrateur si le problème persiste.</p>" . "\n<a href='inscription.php'> Retour à la page d'inscription </a>" ;
47     echo("<script>setTimeout('RedirectionVersInscription()', 1000)</script>");
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

```

Code d'insertion d'un utilisateur dans la base de données.

Ce script php permet, dans un premier temps, de vérifier si les champs du formulaires d'inscription ont été correctement rempli. Cette vérification est n'est pas nécessaire car elle est contrôlée par un code jQuery/AJAX. Si les champs sont bien renseignés, on vérifie que le nom d'utilisateur n'existe pas dans la BDD pour ensuite l'insérer ; ce cas est aussi contrôlé par un script AJAX.

Le formulaire de connexion obéit à des règles similaires, il n'est donc pas utile de détailler son code dans ce rapport.

2. Rôles de l'administrateur

Comme demandé, nous avons inscrit un utilisateur admin (mdp=admin) dans la base de donnée. Cette sous-partie détaillera les actions supplémentaires que peut faire l'admin par rapport à un utilisateur lambda.

a.) Suppression utilisateurs

Les utilisateurs et l'administrateur n'ont pas exactement la même interface après s'être connecté :

The image shows two side-by-side browser screenshots of a web application named 'Se7en'. Both screens have a header with 'Se7en', 'About', and a connection status message ('Connecté en tant que [username]') followed by a 'Déconnexion' button. Below the header is a 'Bienvenue !' message and a grid of three buttons: 'Les 7 Merveilles du Monde', 'Les capitales des 7 pays les plus puissants', and 'Les 7 dernières villes organisatrice des JO'. A large blue 'Score' button is at the bottom. The right screenshot, representing the administrator, includes an additional link labeled 'Panneau de configuration' with a gear icon.

Différence d'interface entre un utilisateur et l'administrateur.

A partir du panneau de configuration, l'admin a accès à la page administration.php. Cette page a été rendue inaccessible via l'url pour un utilisateur lambda.

The image shows a browser screenshot of the 'administration.php' page. The URL is 'localhost:8888/MapGame/main/administration.php'. The page has a header with 'Se7en', 'About', and a connection status message ('Connecté en tant que admin'). Below the header is a section titled 'Administration'. It contains two tables: 'Gestion utilisateurs' and 'Gestion questionnaires'. The 'Gestion utilisateurs' table lists users with columns for Id, Nom d'utilisateur, Date d'inscription, and a delete icon. The 'Gestion questionnaires' table lists questionnaires with columns for Id, Nom questionnaire, and a delete icon. At the bottom is a green 'Création questionnaire' button.

Aperçu de la page d'administration

La partie gestion utilisateurs permet, par un simple clic sur le glyphicon poubelle, de supprimer l'utilisateur correspondant. Il est à la fois supprimé de la base de données via un appel AJAX et la liste est mise à jour grâce à jQuery.

Cette manipulation peut être considérée comme technique car il faut faire correspondre chaque glyphicon à un utilisateur.

```
<table id="table_user" class="table table-striped table-bordered">
<thead>
<tr>
<th>id</th>
<th>Nom d'utilisateur</th>
<th>Date d'inscription</th>
<th><span class="glyphicon glyphicon-remove-sign"></span></th>
</tr>
</thead>
<tbody>
<?php
$req1=$bd->prepare('select * from users order by id');
$req1->execute();
while($tab1 = $req1->fetch(PDO::FETCH_ASSOC)){
    if(htmlspecialchars($tab1['login'], ENT_QUOTES) != "admin"){
        echo "<tr id='ligne_user' . $tab1['id'] . "" >";
        echo "<td>" . $tab1['id'] . "</td>";
        echo "<td>" . htmlspecialchars($tab1['login'], ENT_QUOTES) . "</td>";
        echo "<td>" . $tab1['date_inscription'] . "</td>";
        echo "<td> <span id='user' . $tab1['id'] . '"class='element_supprimable glyphicon glyphicon-trash'"> </td>";
        echo '</tr>';
    }
    //On ne souhaite pas afficher l'utilisateur admin dans la liste des utilisateurs pouvant être supprimé.
}
?>
</tbody>
</table>
```

Code générant le tableau utilisateur.

La technique consiste à attribuer un l'id propre à l'utilisateur dans la base de données au glyphicon concerné. Dans cette page, chaque glyphicon à pour id : user[l'id de l'utilisateur]. Lorsqu'on clique sur une poubelle, l'id est passée en paramètre d'une fonction JavaScript via l'objet event.

```
function supprimer_element_bdd(event){
/*
Suppression d'un utilisateur dans la bdd et retrait de cet
utilisateur dans la liste affichée.
*/
var id = event.target.id;

$.get("requete_ajax_administration.php",
{elementId: id},
function(reponse)
{
    if(reponse=="Supprimé"){
        $('#ligne_' + id).remove();
    } else if(reponse=="Rien à faire"){
        console.log(reponse);
    }
});
?>
<?php
require('../database_auth.php');

if( isset($_GET['elementId']) )
{
    //un élément Id est de la forme userXX avec XX = l'id de l'utilisateur dans la table 'users'.
    $id = $_GET['elementId'];
    if (strpos($id, 'user') != false){
        //strpos retourne false si 'user' n'est pas dans $id, sinon elle retourne la position du 1er char de 'user'.
        //on veut désormais extraire la valeur XX de $id.
        $extract_id = str_replace('user','',$id);
        //on remplace la sous chaîne 'user' par une chaîne vide, $extract_id correspond à l'id de l'utilisateur à supprimer
        $req=$bd->prepare('DELETE FROM users WHERE id=:num');
        $req->bindParam(':num',$extract_id);
        $req->execute();

        echo 'Supprimé';
    } else
        echo 'Rien à faire';
    }
}
```

Fonction JavaScript et script PHP appelé par AJAX

Il nous suffit ainsi d'extraire la valeur de l'id utilisateur de l'id du glyphicon cliqué grâce à la fonction PHP str_replace() et d'utiliser une requête SQL DELETE FROM. Cette technique consistant à concaténer une chaîne de caractère à un id puis de récupérer l'id a été utilisée de nombreuses fois au cours de ce projet, elle ne sera plus détaillée dans ce rapport.

b.) Création et modification de questionnaire

La création d'un questionnaire se fait via un formulaire basique accessible par le bouton situé au pied de la page d'administration. L'idQuestionnaire étant auto-incrémenté et le statut désactivé par défaut, la création d'un questionnaire nécessite de choisir un nom de questionnaire qui n'existe pas dans la base de données. L'administrateur pourra ensuite modifier un questionnaire grâce au glyphicon 'clé à molette' situé à droite du nom du questionnaire sur la page d'administration.

En cliquant sur cette clé à molette, une fonction JavaScript est lancée. Elle sert à soumettre un formulaire caché permettant de transmettre à la page gestion_questionnaire.php des informations spécifiques à un questionnaire (son id et son nom) via la variable PHP \$_POST.

```
<tbody>
<?php
$req2=$bd->prepare('select * from questionnaires order by idQuestionnaire');
$req2->execute();
while($tab2 = $req2->fetch(PDO::FETCH_ASSOC)){
    echo '<tr>';
    echo "<td>" . $tab2['idQuestionnaire'] . "</td>";
    echo "<td>" . $tab2['nomQuestionnaire'] . "</td>";
    echo "<td> <span id='questionnaire' . $tab2['idQuestionnaire'] . ''class='element_modifiable glyphicon glyphicon-wrench'> </td>";
    echo '</tr>';
    /* Formulaire masqué permettant de rediriger vers la page gestion_questionnaire.
    Ce formulaire envoie l'id et le nom du formulaire à modifier à la page. */
    echo "<form method='post' action='gestion_questionnaire.php' id='form_" . $tab2['idQuestionnaire'] . "'>";
    echo '<input type="hidden" name="idQuestionnaire" value="" . $tab2["idQuestionnaire"] . ">';
    echo '<input type="hidden" name="nomQuestionnaire" value="" . $tab2["nomQuestionnaire"] . ">';
    echo "</form>";
}
?>
</tbody>
```

Code générant le tableau Gestion Questionnaires avec la création d'un formulaire caché spécifique à chaque ligne/questionnaire du tableau.

```
function modifie_questionnaire(event){
/*
    L'id d'un questionnaire est de la forme #questionnaireXX
    On cherche à extraire XX et de la concatener à la chaîne de caractère 'form_'
    pour soumettre le formulaire correspondant au questionnaire à modifier.
*/
var questionnaire_a_modifier = event.target.id;
var id_questionnaire = questionnaire_a_modifier.replace('questionnaire', '');
var formulaire_a_soumettre = "form_" + id_questionnaire;

$('#' + formulaire_a_soumettre).submit();
}
```

Fonction JavaScript permettant de soumettre le formulaire correspondant à la clé cliquée.

Après soumission du formulaire, on est redirigé vers la page gestion_questionnaire.php :

idQuestion	nomQuestion	latitude	longitude
1	Où se situe la capitale des Etats-Unis?	38.8951	-77.0364
2	Où se situe la capitale de la Chine ?	39.9075	116.397
3	Où se situe la capitale du Japon ?	35.6895	139.692
4	Où se situe la capitale de l'Allemagne?	52.5244	13.4105
5	Où se situe la capitale de la France?	48.8667	2.33333
6	Où se situe la capitale du Royaume-Uni ?	51.5085	-0.12574
7	Où se situe la capitale de l'Inde?	28.6358	77.2244

Aperçu de la page de gestion questionnaire.

Cette page comprend de nombreuses fonctionnalités interagissant avec la base de données. Nous allons décrire ces outils de haut en bas.

Le champs nomQuestionnaire permet de mettre à jour le nom d'un questionnaire. On rappelle qu'un nomQuestionnaire doit être unique dans la table.

La carte et les deux champs en dessous servent à récupérer les coordonnées d'un point cliqué. Un raccourci a été implémenté afin d'attribuer ces coordonnées à une question après avoir cliqué sur la clé à molette correspondant à la question à modifier.

Latitude: -29.549885 Longitude: 28.322754

Attribuer ces coordonnées à la question à modifier

Go

(Cliquez sur une clé à molette et appuyez sur le bouton Go pour attribuer les coordonnées à la question.)

idQuestion	nomQuestion	latitude	longitude	
1	Où se situe la capitale des Etats-Unis?	38.8951	-77.0364	
1	Où se situe la capitale des Etats-Unis?	-29.549885	28.322754	

Outil permettant d'attribuer des coordonnées après avoir cliqué sur la carte.

Ainsi, après avoir attribué de nouvelles coordonnées à une question (via l'outil ou en tapant une valeur dans le champ correspondant), il est possible de mettre à jour la question via une requête Ajax qui modifiera les valeurs dans la table correspondante.

Puis nous avons développé une interface de correction propre à chaque questionnaire. Cette page sera détaillée dans la partie correspondante. Il est possible d'envoyer au serveur une image qui sera affichée lors de la correction. Une description est aussi recommandée pour que le questionnaire ait un côté culturel.

Une image uploadée doit répondre à deux critères : sa taille doit être inférieure à 4Mo et le fichier doit forcément être un .jpg, .jpeg, .gif ou un .png.

```

else{
    $dossier = "img/";
    $nom_fichier_sans_extension = "" . $_POST["idQuestionnaire"] . "_" . $_POST["idQuestion"] . "";
    $nom_fichier_avec_extension = "" . $nom_fichier_sans_extension . "." . $extension_upload . "";
    //Une image a pour format de nom: idQuestionnaire_idQuestion.extension
    $emplacement = "" . $dossier . "" . $nom_fichier_avec_extension . "";

    $resultat = move_uploaded_file($_FILES['image']['tmp_name'], $emplacement);

    if ($resultat) echo "Transfert réussi";
    else echo "Transfert échoué";
}

```

Code permettant de formater le nom d'une image et de la placer dans le répertoire img.

```

<?php
    $nbr_question = 7;
    $extensions_valides = array( 'jpg' , 'jpeg' , 'gif' , 'png' );
    for ($div_question = 1; $div_question <= $nbr_question; $div_question++) {
        echo '<div class="col-lg-12 col-md-12 col-sm-12">';
        echo '<h4>Question ' . $div_question . '</h4>';

        $indiceQuestionDansTab2 = $div_question - 1;
        $id_image = $tab['idQuestionnaire'] . "_" . $div_question;

        $pwd_image = "img/" . $id_image . "";
        //Chemin vers l'image sans l'extension

        //Ajout des différentes extensions possibles
        $jpg_image = "" . $pwd_image . "." . $extensions_valides[0] . "";
        $jpeg_image = "" . $pwd_image . "." . $extensions_valides[1] . "";
        $gif_image = "" . $pwd_image . "." . $extensions_valides[2] . "";
        $png_image = "" . $pwd_image . "." . $extensions_valides[3] . "";

        if(file_exists($jpg_image)){
            echo '<div class="col-lg-12 col-md-12 col-sm-12 img_container">';
            echo '<img src="" . $jpg_image . "" alt="img" . $id_image . "" height="160" width="100" >';
            echo '</div>';
        }
        else if(file_exists($jpeg_image)){
            echo '<div class="col-lg-12 col-md-12 col-sm-12 img_container">';
            echo '<img src="" . $jpeg_image . "" alt="img" . $id_image . "" height="160" width="100" >';
            echo '</div>';
        }
        else if(file_exists($gif_image)){
            echo '<div class="col-lg-12 col-md-12 col-sm-12 img_container">';
            echo '<img src="" . $gif_image . "" alt="img" . $id_image . "" height="160" width="100" >';
            echo '</div>';
        }
        else if(file_exists($png_image)){
            echo '<div class="col-lg-12 col-md-12 col-sm-12 img_container">';
            echo '<img src="" . $png_image . "" alt="img" . $id_image . "" height="160" width="100" >';
            echo '</div>';
        }
        else
            echo "Pas d'image enregistrée.";
    }

```

Code permettant d'afficher une image précédemment uploadée dans la page de gestion_questionnaire.

Statut du questionnaire

- Questionnaire activé
- Questionnaire désactivé

Outil au pied de la page gestion_questionnaire permettant d'activer ou désactiver un questionnaire lors d'une maintenance par exemple.

3. Gestion des scores

Cette partie ne traitera pas de l'attribution des scores en fonction de la réponse de l'utilisateur mais de la page score.php accessible depuis le menu principal.

The screenshot shows a web browser window with two main sections: 'Records personnels' and 'Score'.

Records personnels (Personal Records):

Nom du questionnaire	Score
Les 7 Merveilles du Monde	0
Les capitales des 7 pays les plus puissants	56

Score:

Nom	Nom du questionnaire	Score	Date
dinesh	Les capitales des 7 pays les plus puissants	56	2018-05-17 19:43:15
dinesh	Les capitales des 7 pays les plus puissants	30	2018-05-17 09:47:42
dinesh	Les 7 Merveilles du Monde	0	2018-05-17 09:45:44

[Retour au menu principal](#)

Page regroupant les scores d'un utilisateur.

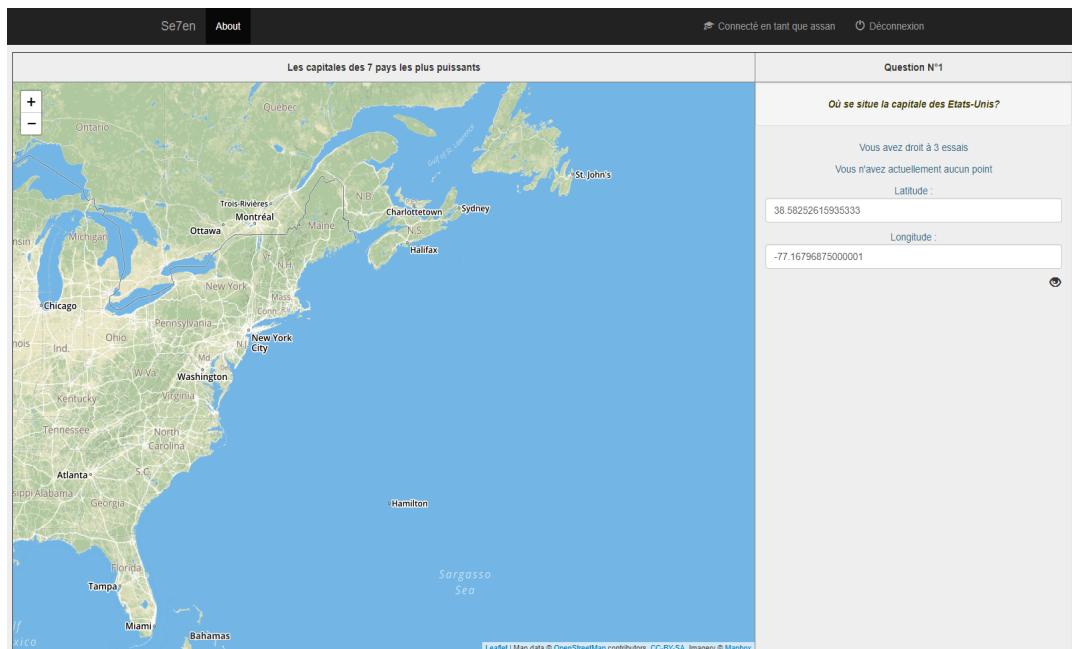
Comme demandé les records ont pris en charge. Le premier tableau affiche le meilleur score du joueur en fonction du questionnaire. La requête sql permettant d'afficher est basique, elle utilise la fonction max() et un group by.

Le tableau affiché dans la partie score est l'historique des parties jouées par l'utilisateur. Il est possible de trier les lignes du tableau en cliquant sur les flèches situées à la première ligne. Ainsi, les noms peuvent être triés dans l'ordre alphabétique, les scores croissants/décroissants, et les parties des plus récentes aux plus anciennes et inversement.

III) DÉROULEMENT DU JEU

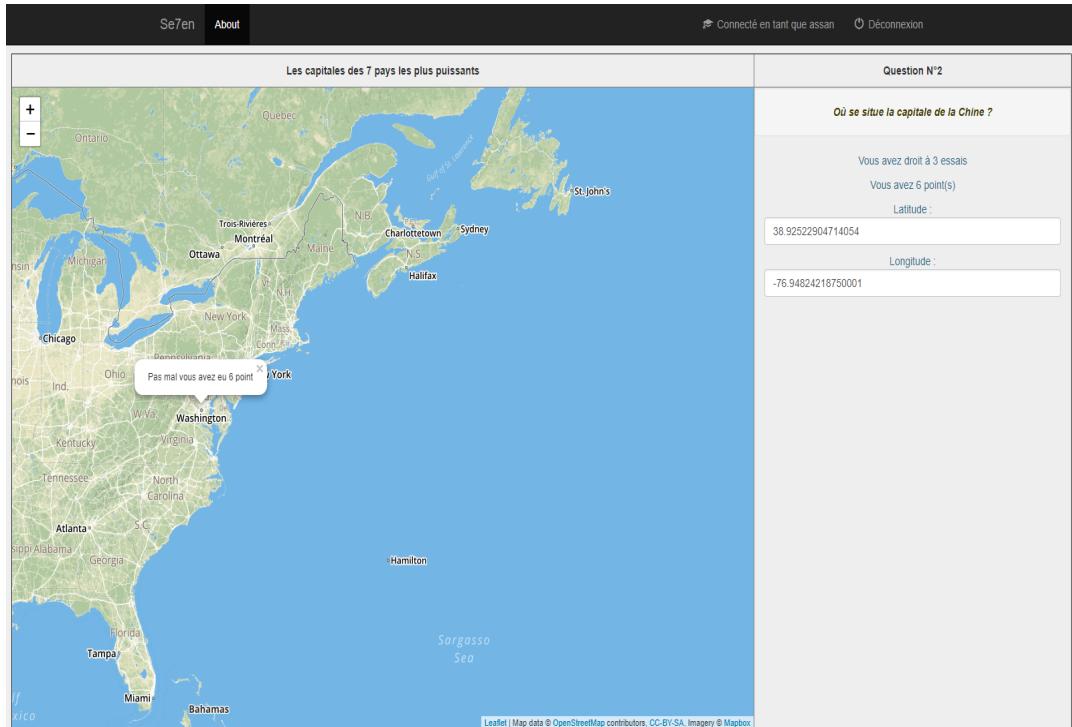
1. Principe du moteur de jeu

Le but du questionnaire est de cliquer le plus précisément possible sur la zone sur laquelle la question porte afin d'obtenir un maximum de point, c'est-à-dire 10. Pour pouvoir créer ce moteur nous avons fait en sorte d'afficher les questions ainsi que la carte dès que le joueur arrive sur la page en question. En ce qui concerne la répartition des points, nous avons choisi de créer des cercles invisibles et concentriques comme au tir à l'arc, et chaque cercle correspond à un nombre de points (de 2 en 2 jusqu'à 10, soit 5 cercles en tout). Le diamètre des cercles augmente de 8km par rapport à l'origine du cercle. Lorsque le joueur s'approche à une certaine distance de la réponse, son curseur entre dans un cercle de 100km de diamètre ayant pour origine les coordonnées présente dans la base de données. Ce cercle permet d'aider le joueur en faisant apparaître une petite image (glyphicon) représentant un œil afin de lui prévenir qu'il est proche de la réponse.



Interface de jeu avec le glyphicon œil affiché. Le curseur est donc à moins de 100 km de la réponse à la question.

Ainsi lorsque le joueur clique sur la réponse, un pop-up s'affiche à l'endroit même de l'action disant s'il a cliqué au mauvais endroit où le nombre de point qu'il a obtenu.



Pop-up affichant le nombre de points obtenus par l'utilisateur. Après une bonne réponse, le nombre d'essai est remis à 3. Un compteur score a été mis en place.

Pour expliquer plus précisément cette page, l'accès à la page JeuLeaflet.php provoque l'appel d'une requête Ajax visant à interroger la base de données puis à récupérer les données concernant un questionnaire sous forme de tableau d'objet (son id, les questions associées, les coordonnées de chaque réponse). On générera également un compteur qui s'incrémentera à chaque fois qu'on passera d'une question à une autre, afin de naviguer dans ce tableau et récupérer les champs nécessaires de chaque objet, ainsi que plusieurs cercles dont chaque rayon permettra de déterminer le nombre de points à chaque réponse.

```

6 Cette erreur n'a aucune conséquence sur le fonctionnement du moteur du jeu.
7 */
8
9 var exoStart=false; //tant que cette valeur reste à false l'exercice ne peut pas commencer
10 var question = []; //tableau d'objet qui contiendra les données de chaque questions
11 var cercle1; //cercles nécessaires à l'attribution des points
12 var cercle2;
13 var cercle3;
14 var cercle4;
15 var cercle5;
16 var cercle6; //cercle nécessaire à l'affichage de l'indice
17 var phase = 0; //permet de naviguer dans le tableau d'objet à chaque question répondue
18 var essai = 3;
19 var point = 0;
20 var exofini = false;
21
22 $.get("requete_ajax_jeuLeaflet.php",
23     {para: "start", idQ: $("#idQuestionnaire").val()}, //requete ajax permettant d'oir les données de chaque questions
24     function(reponse)
25     {
26
27         var i = 0;
28
29         while(i<reponse.length) //on joute chaque objet au tableau crée précédemment
30         {
31             question.push({
32                 idQuestion: reponse[i].idQuestion,
33                 q: reponse[i].nomQuestion,
34                 latitude: reponse[i].latitude,
35                 longitude: reponse[i].longitude
36             })
37             i++;
38         }
39         $('#question_numero').text("Question N°" + question[0].idQuestion);
40         $('#nom_question').text(question[0].q);
41         cercle1 = L.circle([question[0].latitude,question[0].longitude],8000,{color: 'transparent'}).addTo(map); //on est obligé de tout initialiser dans cette fonction
42         cercle2 = L.circle([question[0].latitude,question[0].longitude],16000,{color: 'transparent'}).addTo(map);
43         cercle3 = L.circle([question[0].latitude,question[0].longitude],24000,{color: 'transparent'}).addTo(map);
44         cercle4 = L.circle([question[0].latitude,question[0].longitude],32000,{color: 'transparent'}).addTo(map);
45         cercle5 = L.circle([question[0].latitude,question[0].longitude],40000,{color: 'transparent'}).addTo(map);
46         cercle6 = L.circle([question[0].latitude,question[0].longitude],100000,{color: 'transparent'}).addTo(map);
47
48     }
49
50
51 );

```

Requête Ajax récupérant les questions et réponses propre à un questionnaire et les insérant dans le table question.

Lorsque le joueur clique sur la carte, une fonction jQuery associé aux cliques est appelée. Le but de cette fonction est de comparer la distance entre les coordonnées du clic et celui de la réponse. On établit ainsi le nombre de points en fonction de la valeur de la distance calculée précédemment et celui des rayons des cercles créés. Par exemple si la distance est inférieure au rayon du premier cercle (celui dont le rayon est plus faible) alors le joueur a tous les points, par contre si le joueur clique et que la distance est entre les 2 derniers cercles ayant les plus gros rayons alors le joueur emporte le minimum des points accessibles.

```

143 var pop = L.popup();
144
145 var dist = e.latlng.distanceTo([question[phase].latitude,question[phase].longitude]); //sert à calculer la distance entre 2 points (clique et celui de la réponse)
146
147 if(exoPini!=true)
148 {
149     if(essai==1) //si il ne reste plus que un seul essai au joueur
150     {
151         attrib_pts(); //attribue les points à chaque click (fonction défini en bas)
152
153         miseAJour(); //met à jour les coordonnées de la réponse
154
155     }
156     else
157     {
158         attrib_pts();
159     }
160
161     function attrib_pts() //fonction permettant d'attribuer le score du joueur en fonction de la distance entre le click effectué et la réponse
162     {
163         if(dist<=cercle1.getRadius()) // si la distance calculée est un inférieur au rayon du cercle le plus petit
164         {
165             pop.setLatLng([e.latlng.lat,e.latlng.lng]).setContent("Bonne réponse , vous avez eu 10 point").openOn(map); //affiche au joueur le nombre de point obtenu
166             point+=10;
167             miseAJour(); //met à jour les coordonnées de la réponse
168         }
169
170         else if(dist>cercle1.getRadius() && dist<=cercle2.getRadius())
171         {
172             pop.setLatLng([e.latlng.lat,e.latlng.lng]).setContent("Vous y étiez presque , vous avez eu 8 point").openOn(map);
173             point+=8;
174             miseAJour();
175         }
176         else if(dist>cercle2.getRadius() && dist<=cercle3.getRadius())
177         {
178             pop.setLatLng([e.latlng.lat,e.latlng.lng]).setContent("Pas mal vous avez eu 6 point").openOn(map);
179             point+=6;
180             miseAJour();
181         }
182         else if(dist>cercle3.getRadius() && dist<=cercle4.getRadius())
183         {
184             pop.setLatLng([e.latlng.lat,e.latlng.lng]).setContent("Pas mal vous avez eu 4 point").openOn(map);
185             point+=4;
186             miseAJour();
187         }
188     }
189 }

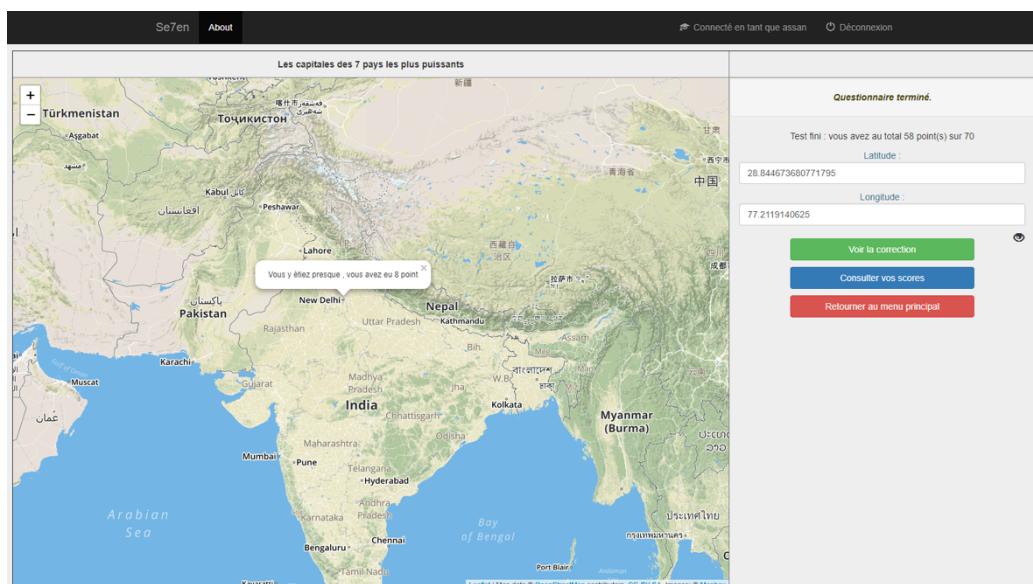
```

Fonction attribuant les points au joueur en fonction de la distance entre le clic et l'origine des cercles concentriques.

2. Correction du questionnaire

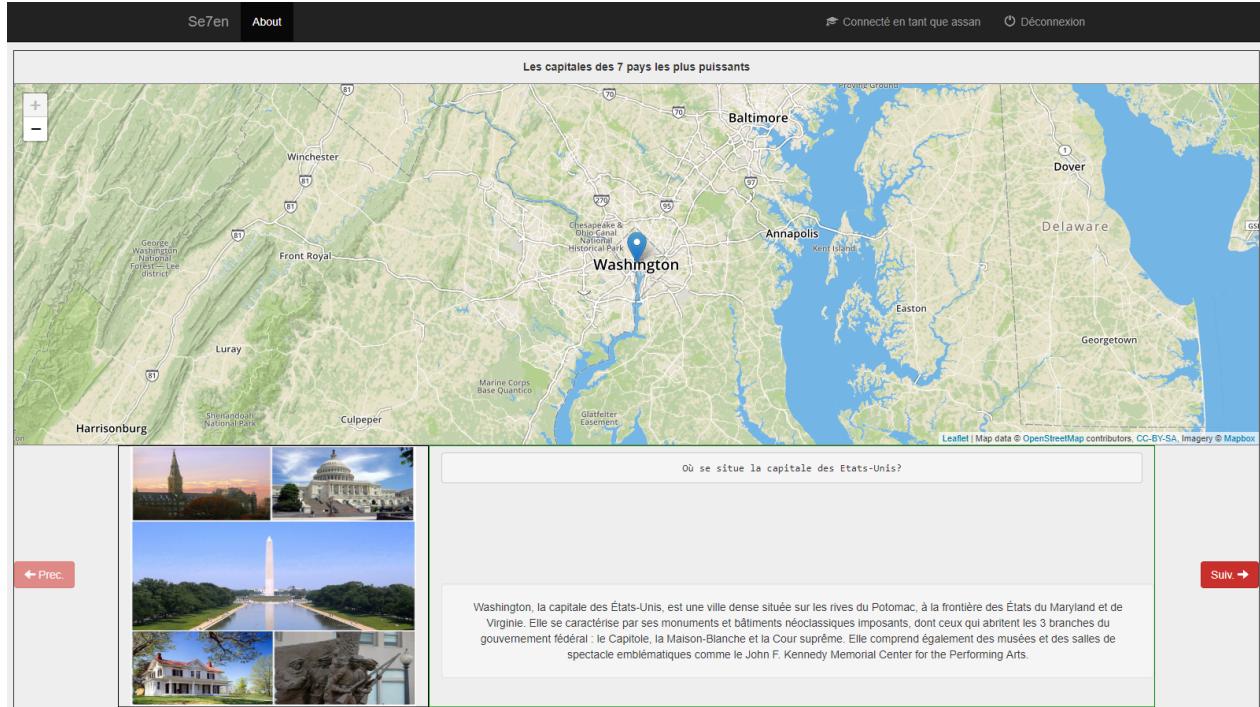
Lorsque le joueur a fini de répondre à l'ensemble du questionnaire, il se voit proposer 3 choix :

- Il peut soit retourner directement au menu principal ;
- Il peut soit aller voir son score ;
- Il peut (et c'est cette partie que l'on va détailler) voir la correction du questionnaire.



Boutons proposés en fin de questionnaire

Lorsque le joueur arrive sur la page de correction, il se voit afficher la carte où un marqueur est placé à l'endroit même de la réponse, avec la possibilité de dézoomer afin qu'il puisse situer précisément l'emplacement.

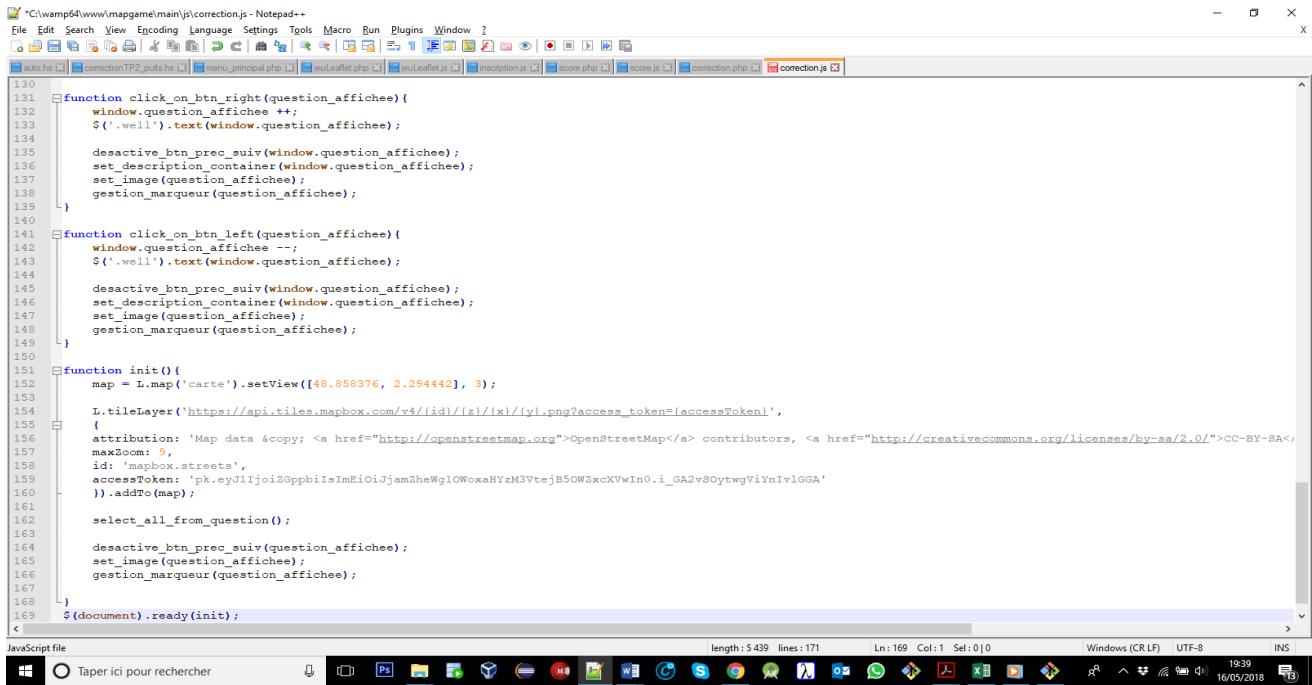


Page de correction d'un questionnaire

En bas de page, le joueur pourra voir des images du lieu en question ainsi que sa description associée. Il pourra par ailleurs naviguer entre les réponses à l'aide des boutons rouges situés sur les coins en bas de page.

Cette page fonctionne en partie comme celui du moteur de jeu, c'est-à-dire que l'accès à cette page provoque l'appel d'une fonction Ajax qui interroge la base de données afin de récolter les données du questionnaire concerné, puis transfère le tout dans un tableau JavaScript. Après avoir effectué le traitement, on initialise la position de la carte en fonction des coordonnées de la première question, puis on affiche l'image et la description associée. On crée aussi une variable permettant de naviguer dans le tableau d'objet afin de récupérer les champs dont on a besoin.

Le fait d'appuyer sur un bouton incrémenté (bouton de droite) ou décrémenté (bouton de gauche) une variable permettant de naviguer dans le tableau d'objet créé pour cette page, et on récupère ensuite les champs nécessaires pour l'affichage de la réponse.



```
/* C:\wamp64\www\mapgame\main\js\correction.js - Notepad++ */
File Edit Search View Encoding Language Settings Tools Macro Run Plugins Window
auto.hdr.js correctionTP2.php.html menu_principal.php.js menuConf.php.js inscription.js.js score.php.js.js score.js.js correction.php.js.js correction.js.js
130     function click_on_btn_right(question_affichee){
131         window.question_affichee++;
132         $('.well').text(window.question_affichee);
133
134         desactive_btn_prec_suiv(window.question_affichee);
135         set_description_container(window.question_affichee);
136         set_image(question_affichee);
137         gestion_marqueur(question_affichee);
138     }
139
140
141     function click_on_btn_left(question_affichee){
142         window.question_affichee--;
143         $('.well').text(window.question_affichee);
144
145         desactive_btn_prec_suiv(window.question_affichee);
146         set_description_container(window.question_affichee);
147         set_image(question_affichee);
148         gestion_marqueur(question_affichee);
149     }
150
151     function init () {
152         map = L.map('carte').setView([48.858376, 2.294442], 3);
153
154         L.tileLayer('https://api.tiles.mapbox.com/v4/{id}/{z}/{x}/{y}.png?access_token={accessToken}',
155         {
156             attribution: 'Map data &copy; <a href="http://openstreetmap.org">OpenStreetMap</a> contributors, <a href="http://creativecommons.org/licenses/by-sa/2.0/">CC-BY-SA</a>',
157             maxZoom: 9,
158             id: 'mapbox.streets',
159             accessToken: 'pk.eyJ1IjoiZGppbiIsImEiOijjamZheWgiOWoxaHYzM3Vtejb50W3xcXVwIn0.i_GA2vSOytwgViYnIvlGGA'
160         }).addTo(map);
161
162         select_all_from_question();
163
164         desactive_btn_prec_suiv(question_affichee);
165         set_image(question_affichee);
166         gestion_marqueur(question_affichee);
167     }
168
169     $(document).ready(init);

```

Fonction JavaScript permettant de créer une sorte de carrousel avec la possibilité de voir la correction des questions passées.

CONCLUSION

Ce projet fut un bon moyen pour nous d'exercer nos compétences dans le domaine du web et de revoir quelques méthodes concernant la gestion de projet informatique. Nous sommes fiers d'avoir pu répondre entièrement au cahier des charges.

Nous pensons aussi que ce projet fut une bonne expérience car il nous a permis de nous remémorer certain acquis dans le domaine de la programmation obtenu lors de notre DUT informatique, acquis qui nous seront certainement utiles pour la suite de nos études.