**Rapport en PWEB**

Axel Roche 208

Thomas Pasquier 208

Julie Pessey 205

**SOMMAIRE**

[Présentation du jeu 3](#_Toc97996958)

[Technologies utilisées 4](#_Toc97996959)

[Gestion de la base de données 5](#_Toc97996960)

# Présentation du jeu

L’objectif de notre jeu est de placer différents pays sur un fond de carte du monde.

Le joueur lance une partie. Il commence celle-ci avec 3 vies. Le nom d’un pays s’affiche et il doit le placer sur la carte. Il clique à l’endroit où il pense que le pays se situe. Si sa réponse est correcte, alors il continue de jouer. Si ce n’est pas le cas, il perd une vie.

Lorsque le joueur n’a plus de vie, la partie prend fin. Il peut alors en recommencer une.

Depuis le leaderboard, le joueur peut voir son meilleur score, ainsi que celui des autres joueurs.

Il peut alors comparer ses résultats et s’améliorer pour devenir le meilleur joueur.

Dans notre jeu, il y a plusieurs modes disponible :

* Le mode classique : le joueur doit placer les pays dans le monde entier
* Le mode Europe : le joueur doit placer les pays européens

Pour le moment, ce sont les deux seuls modes de jeu disponible mais cela peut être amené à évoluer dans le futur.

Le joueur peut se connecter : de cette manière ses résultats seront enregistrés et visible dans le leaderboard. Il peut aussi jouer sans être connecté et ses résultats ne seront pas sauvegardés.

|  |  |
| --- | --- |
| Page d’inscription | Page de connexion |

Page d’accueil

Une image contenant carte

Description générée automatiquement

Leaderboard

Une image contenant table

Description générée automatiquement

# Technologies utilisées

* Front-end du projet

Pour notre projet, nous avons défini une page d’accueil, une page de leaderboard, une page de connexion et une page d’inscription.

Ces pages sont codées en HTML/CSS avec Bootstrap.

Le front-end fait partie de l’architecture MVC PHP : dans le dossier view, on retrouve les fichiers .tpl qui sont les différentes pages de notre application.

* Gestion de la carte

*Leaflet + js*

* Architecture MVC PHP

L’architecture de notre projet repose sur le principe MVC PHP.

Les fichiers sont répartis dans trois dossiers : model, controller et view.

Dans le dossier model, on retrouve toutes les requêtes vers la base de données.

Dans le dossier controller, on retrouve la gestion des fonctions définies dans les fichiers du dossier model.

Dans le dossier view, on retrouve les fichiers tpl, qui sont les affichages de notre application après l’appel à un contrôleur.

Un fichier index.php charge les différentes pages php lorsqu’elles sont appelées et gère la création de variable de session.

* Logique du jeu

*Js*

* Gestion des coordonnées

*Fichier geojson*

# Gestion de la base de données

Nous avons mis en place une base de données mySQL.

Dans cette base, on retrouve 3 tables : une table qui stocke les utilisateurs, une table qui stocke les parties jouées et une table qui stocke les meilleurs scores des joueurs.

|  |  |  |
| --- | --- | --- |
| CREATE TABLE joueur (      IdJoueur int NOT NULL AUTO\_INCREMENT,      nomJoueur varchar(50),      MotDePasse varchar(50),      PRIMARY KEY (IdJoueur)  ); | CREATE TABLE partie (      IdPartie int NOT NULL AUTO\_INCREMENT,      IdJoueur int,      score int,      PRIMARY KEY (IdPartie)  ); | CREATE TABLE stats (      IdJoueur int,      meilleurScore int,      nbParties int  ); |

Nous avons également défini un jeu de données afin de pouvoir tester notre application :

INSERT INTO joueur(nomJoueur, MotDePasse) VALUES ("Thomas", "monMotDePasse0");

INSERT INTO joueur(nomJoueur, MotDePasse) VALUES ("Axel", "123456789ABC");

INSERT INTO joueur(nomJoueur, MotDePasse) VALUES ("Julie", "/password12");

INSERT INTO partie(IdJoueur, score) VALUES (1, 19);

INSERT INTO partie(IdJoueur, score) VALUES (2, 16);

INSERT INTO partie(IdJoueur, score) VALUES (3, 22);

INSERT INTO partie(IdJoueur, score) VALUES (2, 25);

INSERT INTO partie(IdJoueur, score) VALUES (1, 20);

INSERT INTO stats(IdJoueur, meilleurScore, nbParties) VALUES (1, 20,3);

INSERT INTO stats(IdJoueur, meilleurScore, nbParties) VALUES (2, 25,2);

INSERT INTO stats(IdJoueur, meilleurScore, nbParties) VALUES (3, 22,1);

Pour la gestion de la connexion :

L’utilisateur renseigne son pseudo et son mot de passe. On récupère ces informations et on les compare avec celles dans la base de données.

S’il existe un couple <pseudo, mot de passe> identique à celui que l’utilisateur a donné, alors l’utilisateur a accès à son espace, sinon, un message d’erreur est renvoyé.

Pour la gestion de l’inscription :

Le nouvel utilisateur renseigne les informations nécessaires à la création d’un compte.

On vérifie si ce joueur n’existe pas déjà. Si ce n’est pas le cas, alors un nouveau joueur est créé.

Leaderboard :

Les scores qu’on peut voir dans le leaderboard sont récupérés depuis la base de données (en particulier, de la table stats).

L’affichage est donc mis à jour à chaque partie effectuée et en cas de nouveau meilleur score, celui-ci est pris en compte automatiquement.