La boule qui tombe... qui tombe...

Raed Abdennadher

Développement mobile android - Prof Stephane Malandain

Hepia ITI 3ème année

23 avril 2018

1 Introduction

Le but de ce tp est de réaliser un jeu android :

— L'objectif du jeu est de gagner le plus de points possible.

— Une boule apparaît en haut de l'écran et commence à tomber sous l'éffet de la gravité tant qu'il n'y a pas de plateformes horizontaux qui s'opposent à son mouvement vertical (du haut vers le bas). A noter que la masse de la boule est variable selon le mode du jeu. Par exemple, dans le mode facile, la boule est relativement légère, et donc sa vitesse de chute est relativement moins rapide que celle dans le mode difficile.

— Dans les plateformes il y a des trous à travers desquels la boule peut tomber.

— En commençant une partie, un timer se lance. Il faut que la boule arrive en bas de l'écran avant que le temps échu.

— Sur l'espace du jeu, il y a des bonus et des malus (selon la difficulté du jeu) : un bonus (en verts) ajoute 5 secondes au timer, alos qu'un malus (en rouge) enlève 5 secondes du timer. Quand l'utilisateur choisit le mode facile, il y aura que des bonus dans le jeu. Pour le mode intermédière il y a des bonus et des malus. Dans le mode difficile, il y a que des malus.

— En bas de l'écran, il y a des cases. Chaque case contient un nombre de points différent et aléatoire.

2 Architecture du projet

2.1 Architecture générale

Pour notre site web, nous avons choisi une architecture 3 tiers qui comporte 3 couches :

— Couche "Présentation" : c'est l'interface graphique à traver de laquelle l'utilisateur va pouvoir intéragir avec le site web.

— Couche "Métier" : c'est la partie responsable de tous les calculs et les traitements de données qui se passent en arrière plan.

— Couche "Base de données" : ici on a les scripts postgreSQL qui permettent l'interrogation avec la base de données.

La figure 1 représente l'architecture du projet.

2.2 Base de données

La figure 2 représente le modèle relationnel de la base de données de notre site web.

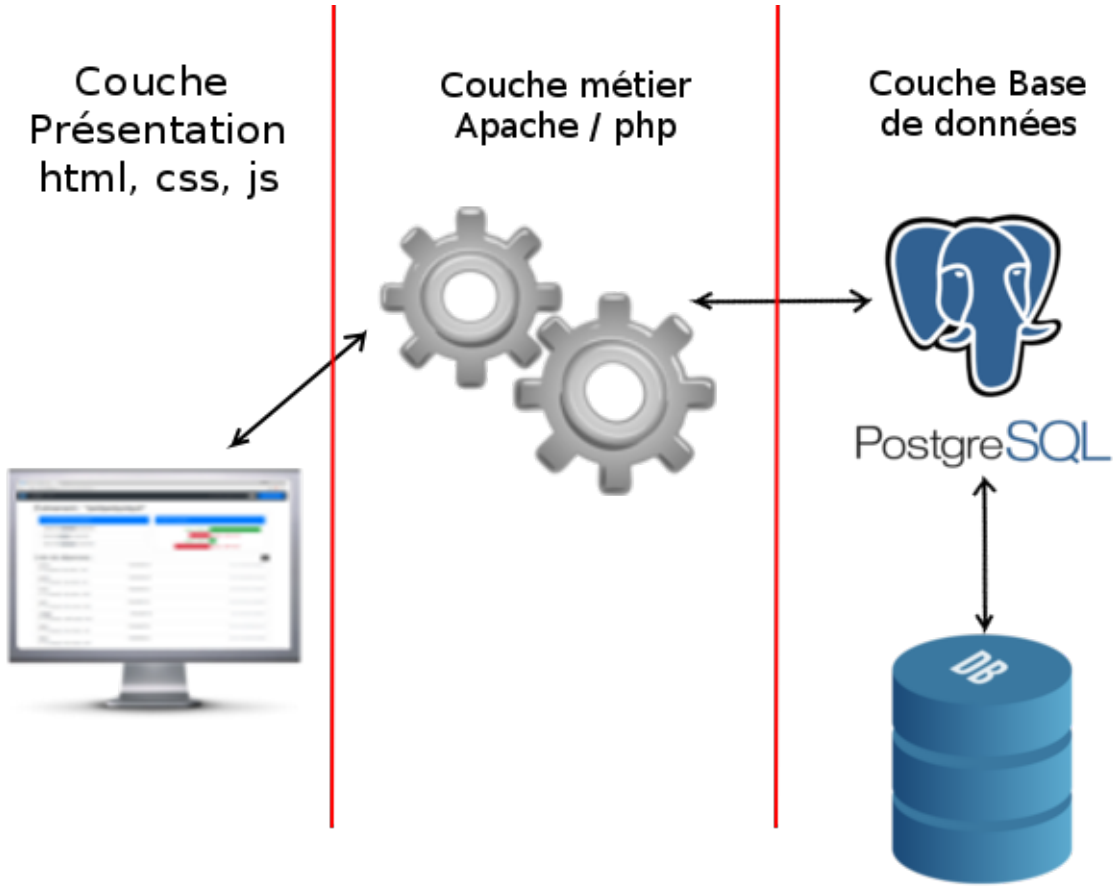

FIGURE 1 – Modèle relationnel de la base de données

3 Technologies et outils

3.1 Technologies utilisées

Nous avons utilisé les technologies suivantes :

— HTML5, CSS3, Bootstrap V4, jQuery 3.2, jQuery UI 1.12, jquery-toastmessage-plugin pour la couche présentation et construction de l'interface graphique de notre site web.

— php7.0 et JavaScript(ES6) pour la couche métier de notre site web (les connexions à la base de données, les calculs, affichage dynamique en fonction des données...)

— PostgreSQL 9.6.4 pour la couche base de données

3.2 Explication des choix

Dans cette section, nous allons expliquer pourquoi nous avons choisi PostgreSQL pour la base de données et Bootstrap pour le design du site.

Pendant la phase de conception de notre site web, nous avons remarqué que nous aurons besoin d'utiliser les triggers pour la cause suivante : au moment de l'insertion d'une nouvelle dépense, il faut vérifier si les utilisateurs concernés appartiennent bien au groupe de l'évènement. Et vu que nous avons déjà eu un cours de base de données en PostgreSQL l'année dernière, nous voulions utiliser cette technologie pour appliquer ce qu'on avait appris.

En ce qui concerne Bootstrap, nous trouvons que c'est un très bon outil qui permet de faciliter la création du design du site. Il offre des mises en forme responsive en ajoutant simplement une classe

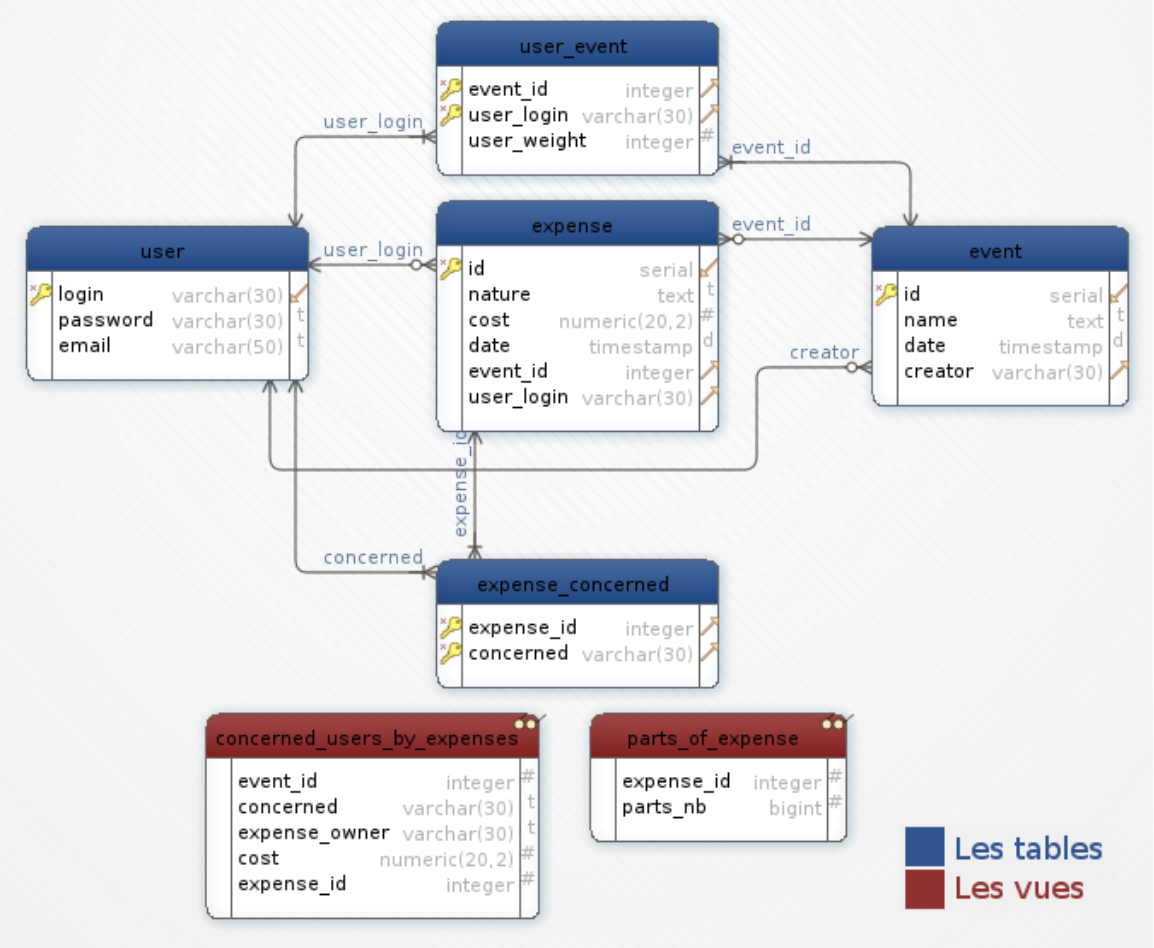

FIGURE 2 – Modèle relationnel de la base de données

css à une balise html par exemple.

4 Structure du projet

Nous avons organisé notre projet de la façon suivante :

- Le script de la création de la base de donées se trouve sous le dossier $psql$. Une documentation pour la base de données est aussi disponible (sous forme html) sous le même dossier.
- Les fichiers CSS se trouvent sous de dossier css
- Les fichiers JavaScript se trouvent sous de dossier js
- Les icons se trouvent sous de dossier $icons$
- Les images se trouvent sous de dossier $images$
- Les fichiers PHP se trouvent directement sous la racine

Dans le fichier $Session.php$ on a la gestion de session : connecter un utilisateur et ouvrir une session, fermer une session et récupérer la session en cours. Les fonctions pour la connexion à la base de données se trouvent dans le fichier $Connexion.php$ (Remarque : en lisant la documentation de php pour la fonction $pg\_connect$, on a constaté que nous avons pas besoin d'appliquer le parton de conception $singletant$ car cette fonction offre la même logique. C'est à dire, si une connexion à la même base de données est ouverte, elle va pas créer une nouvelle). Les fonction qui permettent de communiquer directement avec la base de données sont dans le fichier $Functions.php$.
Au début de chaque page affichée dans le navigateur, toutes ces fonctions sont chargées via le bout de code dans le fichier $includes.php$.

5 Procédure d'installation

Afin de reprendre le projet, il faut :

— Installer un serveur appache (version 2.4.25)

— Installer php (version 7)

— Installer PostgreSQL (version 9.6.4). Remarque : l'installation de PostgreSQL crée automatique un utilisateur unix "postgres" qui va être le super utilisateur du serveur PostgreSQL.

— Une fois ces outils sont installés, on doit créer un utilisateur postgres pour la connexion à la base de données. Dans la console, entrez les commandes suivantes :

  — Se logger en tant que super utilisateur postgres :
  $\#$`su - postgres`

  — Se connecter su serveur PostgreSQL :
  $\$$`psql template1`

  — Ajouter un utilisateur ayant le même nom que votre utilisateur unix courant (Attention, si vous saisissez un nom différent, la connexion aura pas lieu après) :
  $template1 =$ `#CREATE USER 'myUserName' WITH PASSWORD 'myPassword';`

  — Créer une base de données nommée "trip" :
  $template1 =$ `#CREATE DATABASE trip;`

  — Donner tout les droits à l'utilisateur qu'on vient de créer :
  $template1 =$ `#GRANT ALL PRIVILEGES ON DATABASE trip to 'myUserName';`

  — Quitter :
  $template1 =$ `#\q`

  — Après avoir préparé l'environnement de PostgreSQL, vous devez changer les variables de connexion dans le fichier $Connexion.php$ (ligne 7 et 8) :
  ```
  define('USER', 'myUserName');
  define('PASSWORD', 'myPassword');
  ```

  — Maintenant, se connecter en tant que "myUserName" à la base de données via la console :
  $\$$`su - myUserName`
  $\$$`psql -d trip -U myUserName`

  — Créer un schéma sous le nom "trip" :
  $myUserName =>$`CREATE SCHEMA trip;`

  — Finalement, excécuter le script $script\_psql.sql$:
  $myUserName =>$`\i script_psql.sql`

6 Conclusion

Grace à ce mini-projet, nous avons pu appliquer beaucoup de connaissances théorique qu'on avait vu en cours pendant l'année dernière, et nous a permet de bien métriser l'intégration des différentes technologies afin d'aboutir à un produit qui fonctionne.