TORRI Lucas 3A1

Rapport du projet Punto

Table des matières

Présentation du projet :

Punto est un jeu de cartes dans lequel les joueurs doivent poser 4 cartes de la même couleur dans le but de former une suite logique : une rangée, une colonne ou une diagonale.

Règles complètes du Punto : https://montvalsurloir.bibli.fr/doc_num.php?explnum_id=4140

Ce projet a été réalisé dans le cadre de la formation BUT3 Informatique à l'IUT de Vannes, dans le module "Nouveaux paradigmes de bases de données". Il a été réalisé par : Lucas TORRI, 3ème année.

Le framework utilisé pour ce projet est React.js, et l'API est réalisée en Node.js. Les bases de données utilisées sont MySQL, MongoDB et SQLite. Les ports utilisés sont le 3000 pour l'application et le 5000 pour l'API.

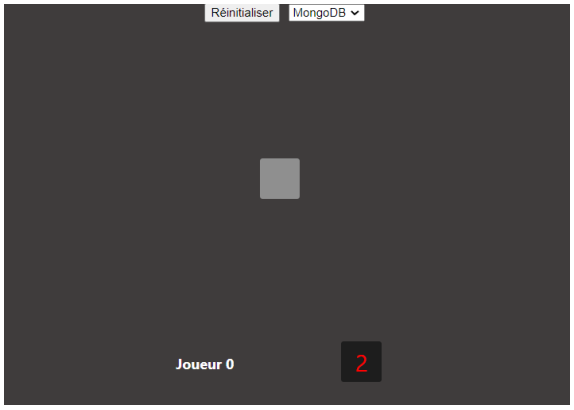
Pour observer et gérer mes bases de données, j'ai utilisé *DBeaver***,** cela m'a permis de gagner beaucoup de temps car je n'avais pas besoin d'effectuer des requêtes SELECT à chaque fois que je voulais vérifier le bon fonctionnement de mon projet.

J'ai également utilisé Github (https://github.com/SadPeanut/Punto/tree/main) pour pouvoir sauvegarder mon code sur le cloud. Cela m'a également permis de récupérer des anciennes versions de mon projet quand une mauvaise manipulation a été faite ou de trop gros changements à annuler.

Pour la gestion de projet, j'ai utilisé le site https://trello.com/ qui permet de créer un tableau qui sert a créer, afficher et classer des tâches afin de mieux s'organiser.

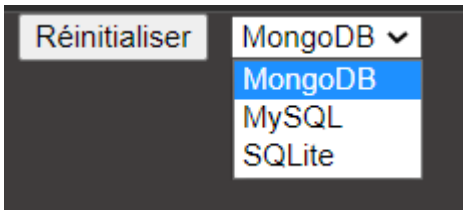
Une partie :

Une partie vierge :



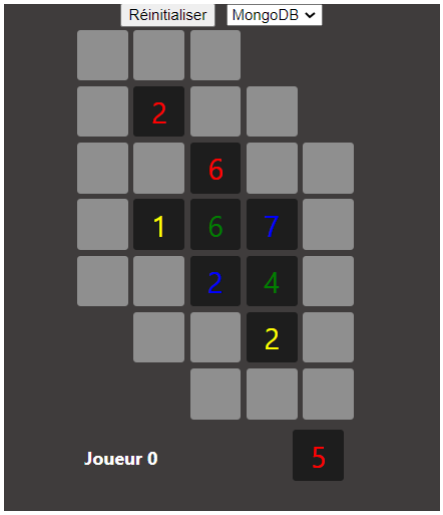
On peut voir la carte qui doit être jouée en bas du tableau de jeu. Ici, le joueur 0 va devoir placer un 2 rouge.

Le menu de choix de la BDD (et aussi le bouton réinitialiser) :



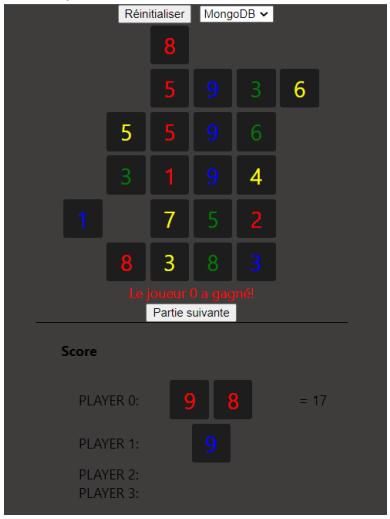
Permettent de réinitialiser le jeu et de choisir la base de données à utiliser lors de la partie.

Une partie en cours :



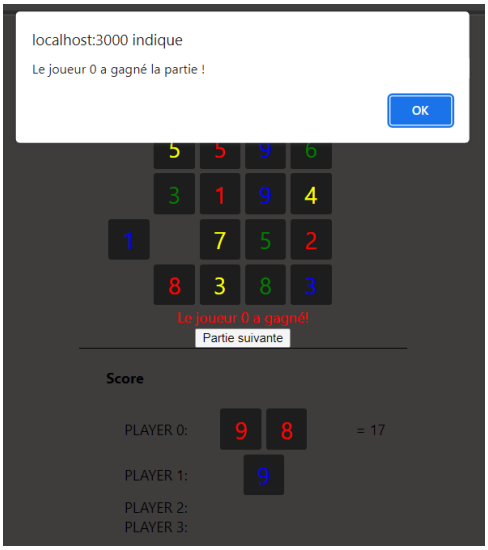
On peut constater que le tableau de jeu s'adapte en fonction de l'état de la partie. Dans tout les cas, le tableau ne permet une surface de jeu que de la taille maximum de 6 en longueur et 6 en largeur (en partant d'un coin).

Une partie finie :



La partie est terminée lorsqu'un joueur gagne 2 manches. Ici, le joueur 0 (rouge) a gagné. De plus, il obtient un score de 9+8=17. Le joueur peut ensuite cliquer sur partie suivante pour passer à la partie suivante.

L’alerte de la partie finie :



On peut voir dans le tableau des scores que quand un joueur gagne, sa carte la plus forte présente sur le jeu est ajoutée à son score.

Fonctionnalités :

L'application respecte les règles du Punto.

Il y a un tableau de jeu qui s'adapte en fonction des choix de placement des joueurs. Il permet d'évoluer dans un espace 6x6 maximum.

On peut réinitialiser la partie en cliquant sur le bouton réinitialiser.

On peut choisir la BDD souhaitée dans le menu déroulant adapté, on peut le changer tout au long de la partie.

Une API express permet de faire le lien entre l'application React Punto et les 3 bases de données.

L'API comporte 2 routes :

- */Partie* : Cette route envoie les données d'une partie (1 fois par partie). Ces données comportent l'id du joueur gagnant, en combien de tours il a gagné, en combien de manches, les points du joueur1,2,3,4.
- */Plays* : Cette route envoie les données d'une action à chaque fois que le joueur clique sur une case. Cela permet de stocker l'historique des coups de chaque partie. Ces données comportent l'id du joueur qui a joué, le x de la carte, le y de la carte, la carte (numéro).

Ensuite, les bases de données récupèrent automatiquement ces données pour les stocker.

MongoDB combine les 2 routes afin d'obtenir une donnée complète, par exemple les plays sont stockés comme cela : plays_joueur1 : []. Cela ne respecte pas la 1NF mais ce n'est pas un problème pour MongoDB.

MySQL et SQLite stockent séparément les 2 données. En effet, il y a 2 tables dans la structure de ces bases (*cf. Documentation*). Une table Partie qui stocke les données de la route */Partie* et une table play qui stocke les données de la route */Plays*. La table Play est liée une et une seule table Partie par une clé étrangère id_partie. Cela permet à plusieurs Play d'être liés à une seule Partie.

Dans ce projet, il y a également un fichier *deleteFrom.js* permettant lors de son exécution (*node deleteFrom*) de supprimer l'intégralité des données de toutes les bases.

Il y a aussi un fichier *monteeEnCharge.js* qui permet, lors de son exécution (*node* monteeEnCharge) de tester la solidité des bases en insérant un volume important de données générées aléatoirement dans les 3 bases. On peut modifier la quantité de données insérées dans le code (par défaut : 20).

Documentation :

https://github.com/SadPeanut/Punto/tree/main/Documentation

Readme.md :

https://github.com/SadPeanut/Punto/blob/main/README.md

Jeu de Punto

Punto est un jeu de cartes dans lequel les joueurs doivent poser 4 cartes de la même couleur dans le but de former une suite logique : une rangée, une colonne ou une diagonale.

Règles complètes du Punto: https://montvalsurloir.bibli.fr/doc_num.php?explnum_id=4140

Installation

Présentation du projet

Ce projet a été réalisé dans le cadre de la formation BUT3 Informatique à l'IUT de Vannes, dans le module "Nouveaux paradigmes de bases de données". Il a été réalisé par : Lucas TORRI, 3ème année.

Le framework utilisé pour ce projet est React.js, et l'API est réalisée en Node.js. Les bases de données utilisées sont MySQL, MongoDB et SQLite. Les ports utilisés sont le 3000 pour l'application et le 5000 pour l'API.

Prérequis

- Node.js
- Git
- MongoDB
- MySQL
- SQLite

Étapes d'installation

1. Installation et création des bases de données :

Noms des BDDs :

- MySQL : *BDD_Punto_MySQL*
- MongoDB : *BDD_Punto_MongoDB*
- SQLite : *BDD_Punto_SQLite*

MySQL : https://dev.mysql.com/doc/

MongoDB : https://www.mongodb.com/docs/

SQLite : https://www.sqlite.org/docs.html

Après avoir créé les 3 bases de données, créez dans chacune une table :

SQLite :

```
CREATE  TABLE Partie ( id INTEGER  PRIMARY KEY AUTOINCREMENT, id_joueur_gagnant INTEGER, manches_gagnees INTEGER, nbTours INTEGI
```

Pour SQLite il faudra changer le chemin de la BDD (dbPath) dans le fichier `punto/src/api/api.js` dans la méthode `insertPartiesQLite(gamedata)`.

2. Installation des fichiers de l'application

Dans votre éditeur de code, clonez le dépôt dans le répertoire de votre choix :

```
git clone https://github.com/SadPeanut/Punto.git
```

Toutes les commandes restantes seront exécutées depuis un terminal ouvert en `/punto`

Installez l'application :

```
npm install
```

3. Lancement de l'application

Lancez l'application :

```
npm start
```

L'application devrait ouvrir automatiquement un onglet dans votre navigateur par défaut sur le port 3000 de localhost.

4. Lancement de l'API

Pour lancer l'API, exécutez cette commande dans un nouveau terminal (au même endroit) :

```
node src/api/api.js
```

L'API devrait se lancer, en affichant : *API Punto listening at http://localhost:5000*

5. Jouez!

Vous avez maintenant tout configuré pour pouvoir jouer à Punto!

Veillez à laisser les 2 terminaux ouverts lors de votre utilisation pour garantir le fonctionnement de l'application.

Trello :

https://github.com/SadPeanut/Punto/blob/main/Documentation/Trello_Punto.png

TORRI Lucas 3A1

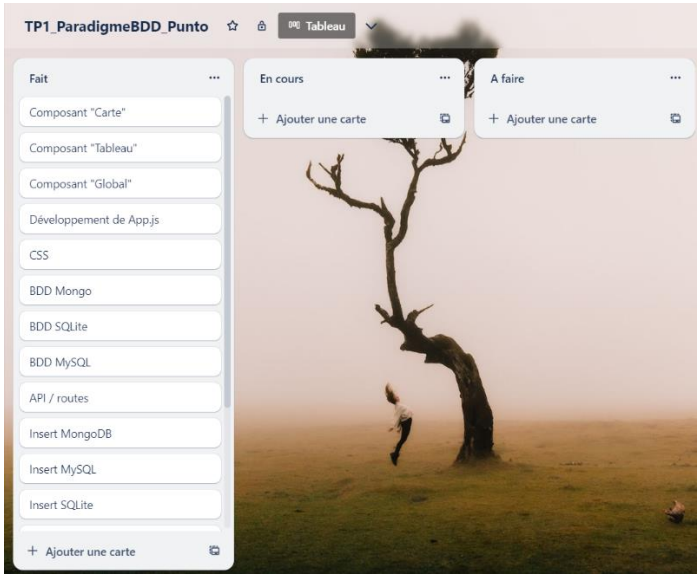
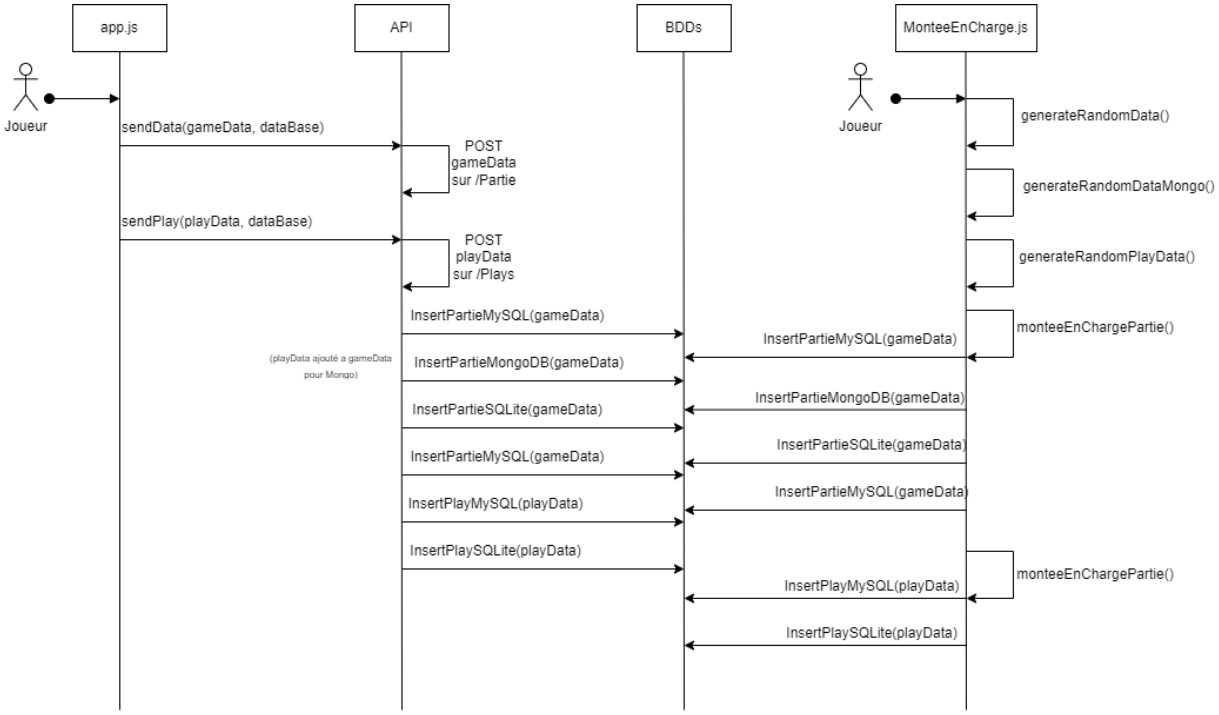


Diagramme de séquence :

https://github.com/SadPeanut/Punto/blob/main/Documentation/S%C3%A9quence_Punto.drawio.png



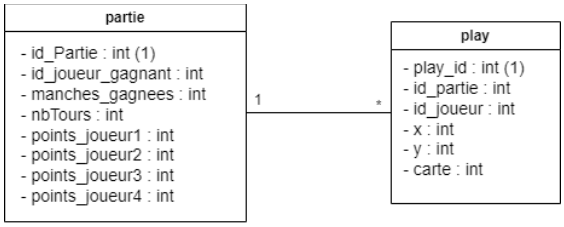
TORRI Lucas 3A1

Diagrammes UML :

https://github.com/SadPeanut/Punto/blob/main/Documentation/UML_MySQL_SQLite.png

https://github.com/SadPeanut/Punto/blob/main/Documentation/UML_MongoDB.png

MySQL & SQLite



partie:

- id_Partie : AUTO INCREMENT

play :

- play_id : AUTO INCREMENT
- id_partie : FOREIGN KEY (partie : id_Partie)

MongoDB