École Centrale Lyon

INF
Application Web

Application web - X-TREM4

Élèves :
Antoine Edy
Lucas Besson

Enseignant :
Daniel Muller
René Chalon

9 mars 2023

Table des matières

1 Introduction

Node.js et Socket.io sont deux technologies qui permettent de concevoir des applications web performantes et évolutives en temps réel. Dans le cadre de ce rapport, nous allons présenter notre projet de développement d'une application web, dans laquelle nous avons développé un jeu qui reprend les règles du puissance 4 (avec une subtilité de notre conception).

Le jeu de puissance 4 est un jeu de société très classique, dont l'objectif est d'aligner 4 pions de la même couleur. Notre application web permet aux utilisateurs de jouer à ce jeu en ligne, en temps réel et avec des joueurs du monde entier. Nous avons utilisé les fonctionnalités de Node.js et Socket.io pour créer une interface de jeu fluide et intuitive, ainsi qu'un système de chat pour permettre aux joueurs de communiquer entre eux pendant la partie. Nous avons décidé de rajouter une fonctionnalité qui pimentera vos parties de puissance 4.

Dans ce rapport, nous allons rapidement revenir sur nos codes, puis nous illustrerons notre application avec un "parcours utilisateur".

2 Courte explication du code

2.1 app.js

Dans ce fichier, on créé un serveur de jeu en ligne utilisant Express et Socket.IO.
Voici les principales étapes du code :

- On importe les modules nécessaires à l'aide de la commande require, tels que Express, Socket.IO, etc.
- On créé une instance d'Express et un serveur HTTP en utilisant la méthode Server d'Express, puis une instance de Socket.IO en passant le serveur HTTP en paramètre.
- On créé des routes Express pour servir des fichiers statiques et des pages de votre application.
- On définit des événements WebSocket pour gérer la communication entre les clients et le serveur.
- Dans l'événement 'connection', on vérifie si la salle est pleine ou si un joueur peut la rejoindre. Si la salle n'est pas pleine et qu'un joueur peut la rejoindre, on ajoute le joueur à la salle.
- Dans l'événement 'update-state', le serveur reçoit des données du client telles que le joueur actuel, la salle et les coordonnées de la cellule à colorer. Le serveur effectue ensuite les opérations nécessaires pour mettre à jour l'état du jeu.
- Le code comporte également deux fonctions, `checkWinCondition` et `checkGameOver`, qui vérifient si le joueur a gagné ou si le jeu est terminé.

2.2 routes/gameBoard.html

- L'élément `<input>` avec l'id `room-input` permet à l'utilisateur d'entrer le code de la salle qu'il souhaite rejoindre.
- Le bouton `<button>` avec l'id `room-submit` permet de soumettre le code de la salle.

— Les éléments `<h3>` avec les classes `room-full`, `victory`, `defeat` et `tie` sont des messages qui s'affichent à différents moments du jeu (par exemple, si la salle est pleine ou si un joueur gagne).
— L'élément `<div>` avec l'id `board` est l'endroit où la grille de jeu sera affichée.
— La balise `<section>` avec l'id `chat` est l'endroit où le chat en direct sera affiché.
— Les éléments `<script>` font référence à des fichiers JavaScript qui seront utilisés pour le fonctionnement du jeu et de la communication en temps réel avec les autres joueurs grâce à Socket.io.

2.3 routes/home.js

Ce code utilise le framework Node.js Express pour créer un serveur web.
— La deuxième ligne crée un nouvel objet Router en utilisant la méthode Router fournie par Express. Router est utilisé pour définir les routes du serveur web, c'est-à-dire les URL auxquelles le serveur peut répondre.
— La troisième ligne définit une route pour la méthode GET. Cette route correspond à l'URL '/' (la page d'accueil du serveur). Lorsqu'un utilisateur accède à cette page, la fonction callback associée est appelée. Cette fonction envoie le fichier index.html situé dans le répertoire racine du serveur (défini par la variable `__dirname`).
— La quatrième ligne définit une route pour la méthode POST. Cette route correspond également à l'URL '/'. Lorsqu'un utilisateur soumet un formulaire à cette page, la fonction callback associée est appelée. Cette fonction envoie le fichier gameBoard.html situé dans le répertoire racine du serveur.
— La dernière ligne exporte l'objet Router créé pour être utilisé par d'autres fichiers JavaScript.

2.4 public/client.js

Le fichier `client.js` est un fichier qui est exécuté dans le navigateur web du client et qui permet de communiquer avec un serveur socket.io. Il contient des fonctions pour envoyer des messages au serveur, pour recevoir des messages du serveur, pour mettre à jour l'état de l'interface utilisateur en fonction des messages reçus etc...
— Les premières lignes initialisent la connexion de la socket et définissent des variables pour le joueur, la salle et la planche de jeu.
— La fonction `Game` initialise la planche de jeu en créant un tableau de 6x8 cellules. Les éléments HTML correspondants sont créés et ajoutés à la page, puis les événements `click` sont ajoutés à chaque cellule et à un bouton de soumission de la salle.
— La méthode `clicked` est appelée lorsque l'un des événements `click` est déclenché sur une cellule, ce qui envoie un message à travers la socket pour mettre à jour l'état du jeu.
— Les événements sont également définis pour écouter les messages envoyés à travers la socket. La salle est considérée comme pleine lorsque le message 'room-full' est reçu, et les messages 'win', 'lose' et 'tie' sont reçus lorsque la partie est terminée. Le message 'patience' est utilisé pour avertir l'utilisateur de ne pas cliquer quand ce n'est pas son tour. Enfin, les messages 'begin-game', 'update-turn' et 'color-cell' sont utilisés pour initier la partie, afficher le tour du joueur en cours et mettre à jour les couleurs des cellules.

3 Le parcours utilisateur

L'utilisateur se connecte sur le site est arrive sur cette page :

FIGURE 1 – Page d'accueil

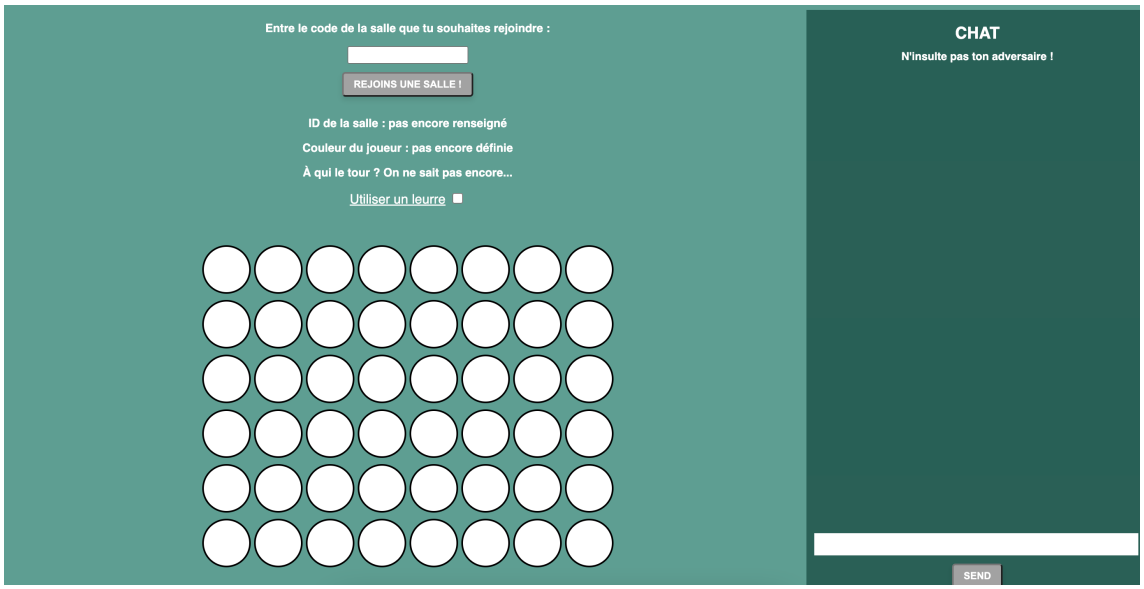

FIGURE 2 – Page de jeu

On arrive sur cette page pour commencer une partie.

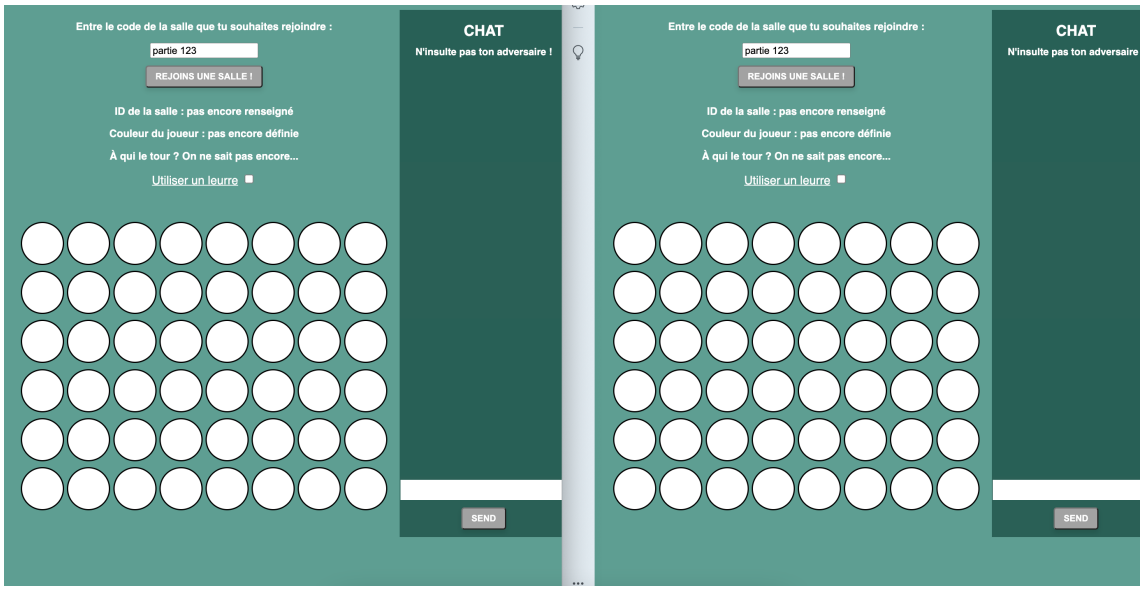

FIGURE 3 – Code de partie

Sur cette page suivante, on entre une "code de partie". Deux joueurs entrant le même code de partie se rejoignent alors.

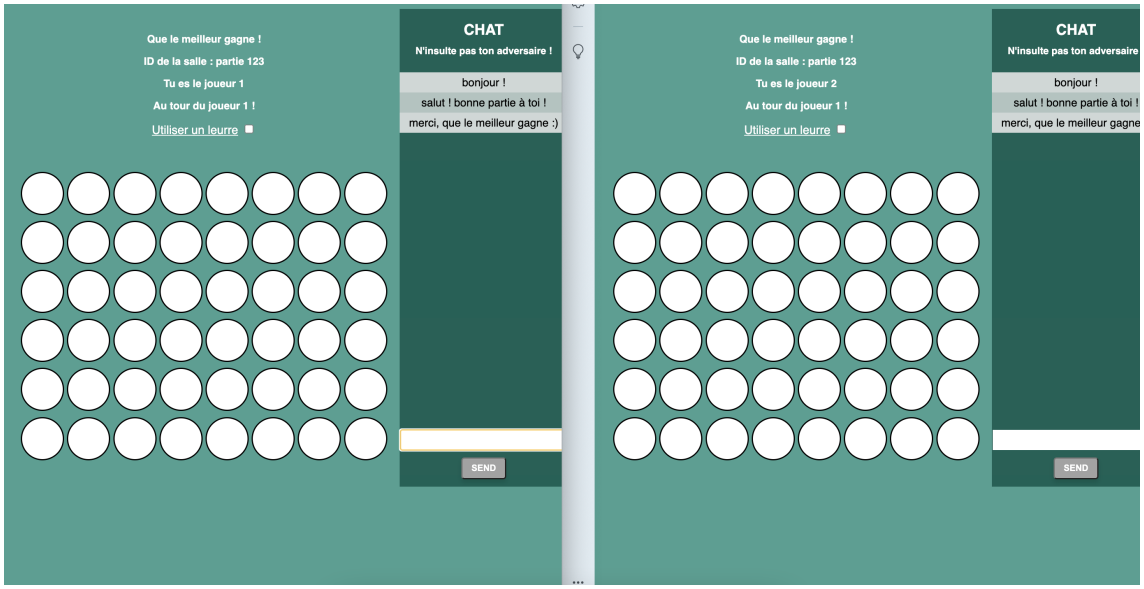

FIGURE 4 – Connectés !

Les deux joueurs se rejoignent : ils peuvent discuter avec le "chat" disponible sur la droite de la page. C'est au joueur 1 de commencer comme indiqué.

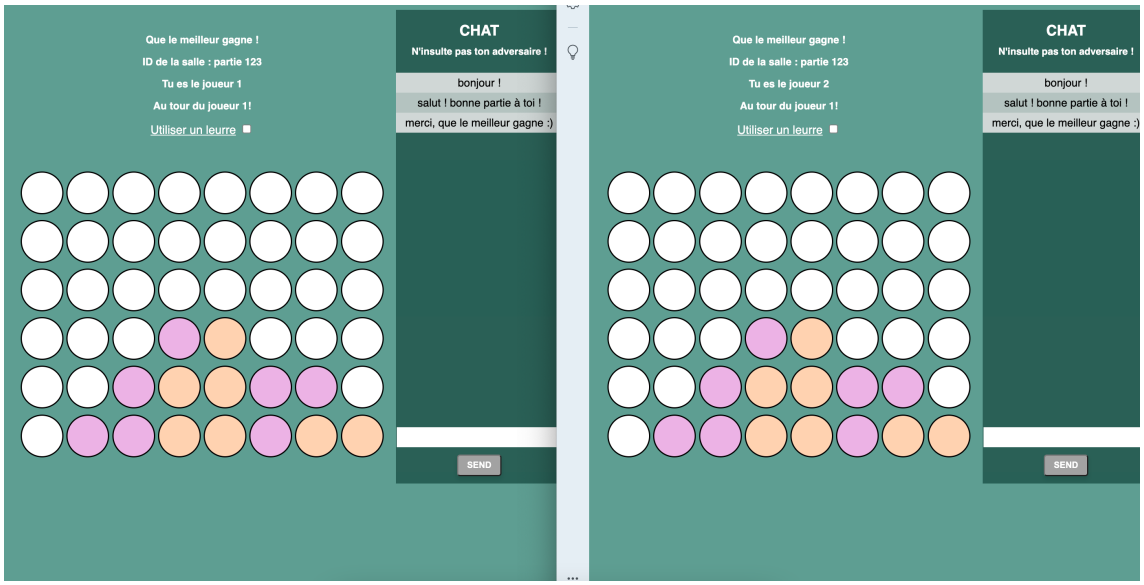

FIGURE 5 – Partie endiablé

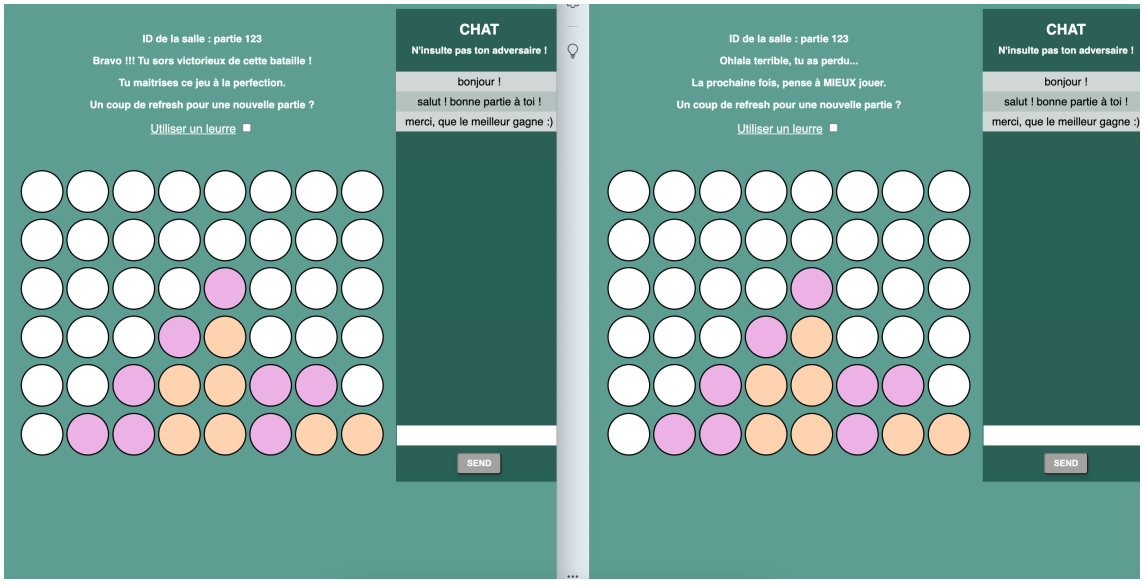

FIGURE 6 – Et c'est gagné !

4 Notre touche personnel

Afin de rendre le jeu très classique du puissance 4 un peu original, nous avons décidé de rajouter une fonctionnalité : celle de poser des "leurres" ! Un leurre est un pion qui a la COULEUR de celui de l'adversaire mais qui COMPTE comme le sien ! De quoi emmêler les pinceaux de l'adversaire... mais aussi les siens !

Une illustration :

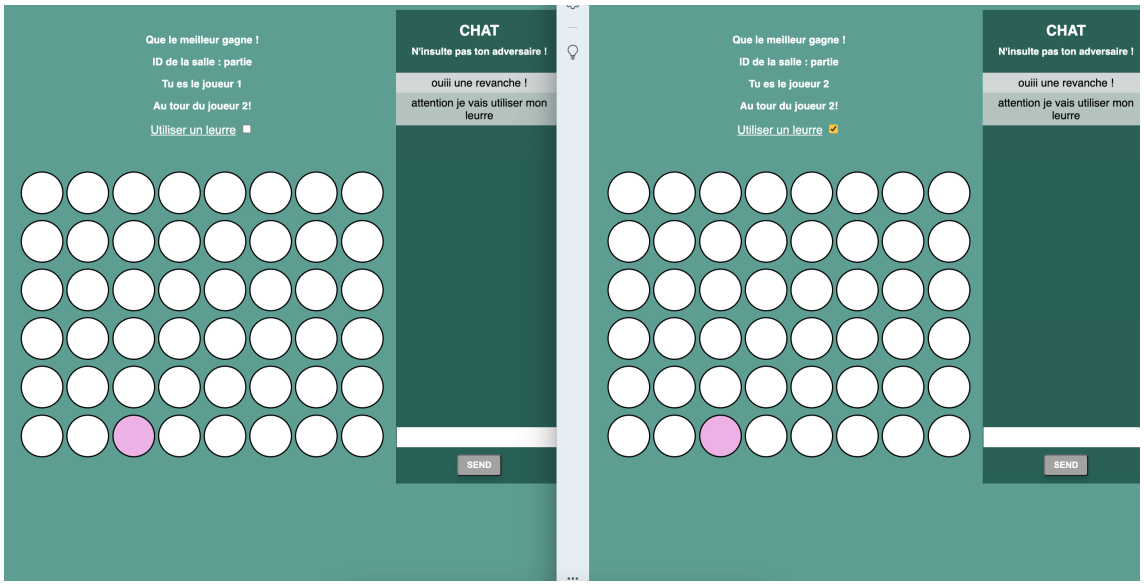

FIGURE 7 – Premier pion

La partie commence normalement, le joueur 1 pose son pion 1...

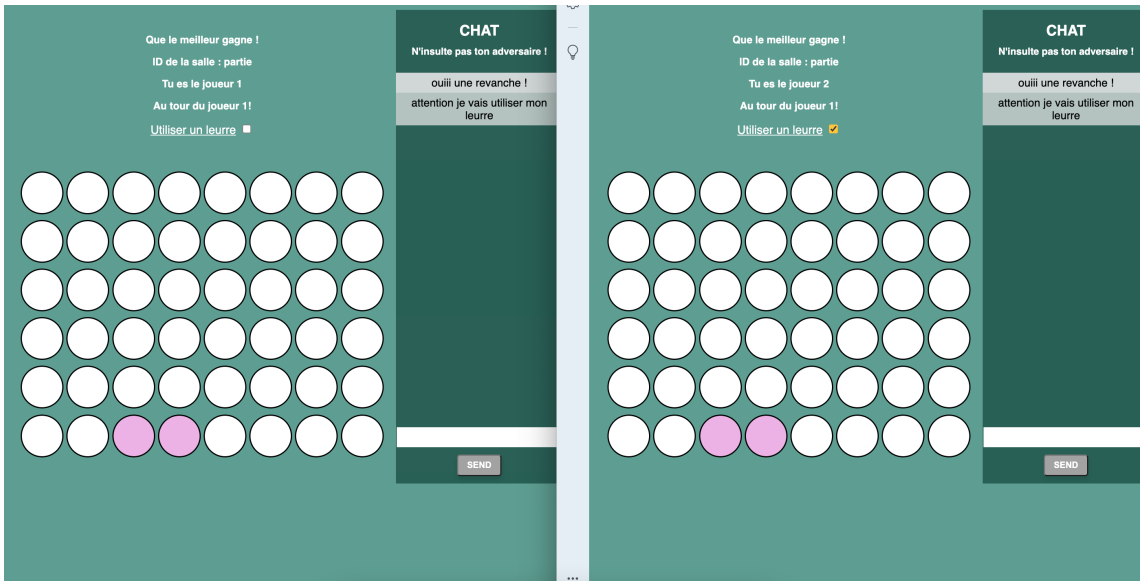

FIGURE 8 – le leurre !

Mais stupeur ! Le joueur 2 pose également un pion rose ! Le joueur 1 est perdu. Cela peut donner scène à des fins de parties... inattendues.

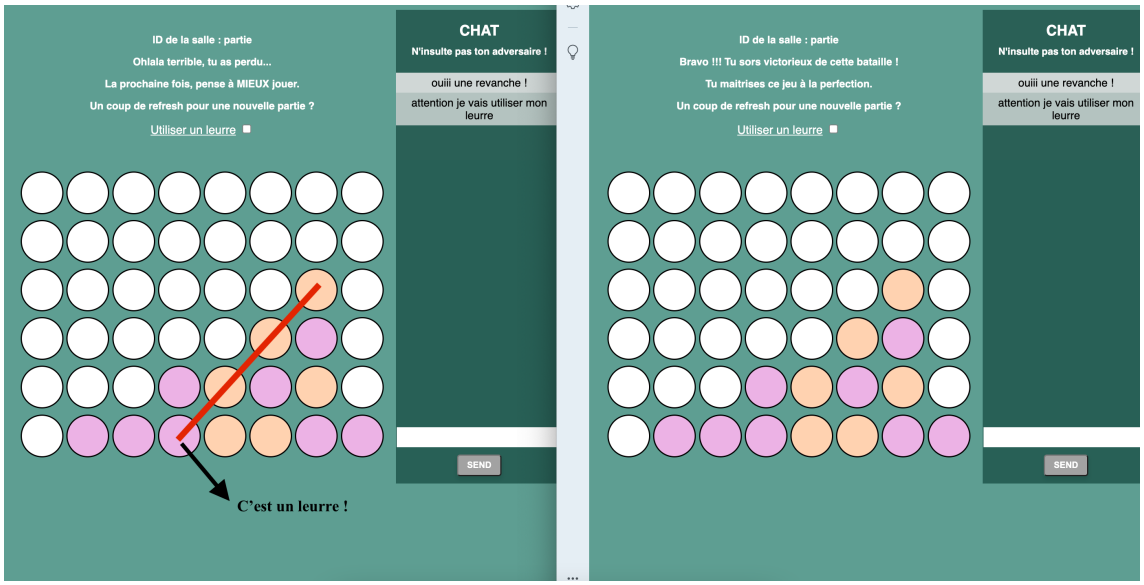

FIGURE 9 – Fin de partie surprenante

Ce nouveau mode de jeu amène à plusieurs nouvelles mécaniques interessante, et fait grandement travailler la mémoire !