Rapport de Progrès - Projet "6 qui prend" (du 14 avril au 20 avril)

Projet de jeu multijoueur "6 Takes!" ("6 qui prend") développé en Node.js (backend) avec frontend Godot. Ce rapport synthétise les évolutions majeures du backend WebSocket depuis le 14 avril.

1. Gestion du Timer par Room

- Implémentation d'un timer de 45 secondes à chaque début de tour.
- Lorsque tous les joueurs ont joué ou que le timer expire, on passe à la phase de traitement.
- Timer visible par les clients via `io.to(roomId).emit("temps-room")`.

2. Traitement séquentiel des cartes

- Refactorisation du traitement via une file d'attente `fileTraitementParRoom[roomId]`.
- Traitement carte par carte en ordonnant par valeur croissante (pour respecter les règles du jeu).
- Blocage automatique du traitement si un joueur doit choisir une rangée (cas spécial), reprise une fois le choix reçu.

3. Refonte des événements socket clients

- Organisation logique : `start-game` -> `tour` -> `play-card` -> `fin-tour` -> `update-table`, `update-scores`.
- Emission ciblée d'événements : `choix-rangee` à un seul joueur + `attente-choix-rangee` aux autres.

4. Gestion des Manches

- Ajout de la mécanique multi-manche : une partie se poursuit tant que
- Aucun joueur n'a atteint le score max (par défaut : 66).
- Le nombre de manches défini n'est pas atteint.

- Nouvelle distribution à chaque manche, tout en conservant les scores.
- Emission possible d'un événement `manche-info` pour informer les clients.

1. Traitement parallèle des cartes

- Désynchronisation quand plusieurs joueurs jouaient en même temps une carte inférieure à toutes les rangées.
- Solution : passage au traitement par file + tri des cartes avant traitement.

2. Blocage lors des cas "choix de rangée"

- Complexité à gérer le "wait" en JS sans bloquer le thread.
- Solution adoptée : `await` dans `traiterProchaineCarte()` avec `Promise` et handler d'événement.

3. Timer relancé trop tôt

- `notifierCarteJouee()` et `lancerTimer()` étaient appelés avant la fin réelle de `traiterCartesJouees`.
- Solution : rendre le handler `play-card` `async` et attendre la fin réelle du traitement.

4. Erreur `rooms.find is not a function`

- Passer `rooms` comme tableau et non comme objet lors d'appels `find()`.

5. Incohérence entre moteur de jeu et serveur

- Utilisation de `jeu.checkEndManche()` prévue, mais la méthode n'était pas intégrée ou utilisable dans le bon contexte.
- Solution : retour temporaire au test classique `joueurs.every(...)`.

- Structure du serveur de plus en plus robuste.
- Traitement centralisé des cas spéciaux.
- Architecture des sockets stabilisée.

- Communication client/serveur plus lisible et contrôlable.

- Intégration de `jeu.checkEndManche()` proprement dans le moteur.

- Affichage dynamique de la manche actuelle dans le client.

- Gestion de la déconnexion d'un joueur en cours de manche.

- Ajout d'un bot pour le mode solo (idée déjà évoquée).

- 14 avril : Implémentation des timers.

- 15 avril : Introduction du traitement automatique pour joueurs inactifs.

- 16-17 avril : Refonte totale du traitement en cas de choix de rangée.

- 18-19 avril : Ajout de la logique de multi-manche.

- 20 avril : Finalisation de la transition vers `traiterProchaineCarte()` et vérification des fins de manche.