

Voici le livrable d'analyse et de relance du chat [Technique] – 06, construit exclusivement à partir du contenu figé de ce chat, conforme aux règles et orienté pilotage/exécution.

1. Présentation détaillée du contenu du chat

Périmètre et rôle dans le projet Le chat [Technique] – 06 est la référence technique officielle du projet. Il formalise l'architecture, les périmètres backend et frontend, le phasage d'implémentation, les principes de sécurité et d'anti-cheat, ainsi que les arbitrages techniques encore ouverts. Il sert de base d'exécution pour les équipes techniques et de socle stable pour relancer un nouveau cycle de travail sans ambiguïté.

Contenu structurant Architecture macro backend-centric, stateless, DB comme source de vérité. Backend gameplay : Orchestration serveur complète (validation des coups, persistance atomique, fins de partie). Support des tie-breaks (création, rattachement, résolution déterministe). Intégration avec la logique tournoi existante. Phase 6.0.C implémentée, testée et gelée. Frontend : Actuel : affichage et consommation API. À venir : Phase 6.1 (plateau d'échecs, interactions basiques). Sécurité & journalisation : Pas de logique financière directe. Logs et traçabilité serveur, timestamps UTC. Anti-cheat : V1 en place (validation serveur stricte). V2 explicitement hors périmètre. Phasage : Inclus/figé vs reporté clairement listés. Arbitrages restants : Concurrence tie-breaks, WebSockets, frontend chessboard, incrément de temps, tests de concurrence. FIGÉ vs PROVISOIRE FIGÉ

Architecture backend-centric et stateless. Orchestration gameplay serveur. Modèle de tie-breaks et intégration tournoi. Anti-cheat V1. Journalisation serveur et UTC. Phase 6.0.C backend. PROVISOIRE / OUVERT Choix précis des mécanismes de concurrence (option A/B). Implémentation frontend gameplay (Phase 6.1). Passage au temps réel (WebSockets). Anti-cheat V2. Incréments de temps et tie-breaks avancés. Le contenu peut être copié tel quel pour ouvrir un nouveau chat technique vierge, sans perte d'information.

2. Analyse stratégique Forces Périmètre technique clair, cohérent et dédupliqué. Backend gameplay fonctionnel, gelé et déterministe. Séparation nette backend / frontend / finance. Gestion des tie-breaks pensée dès l'architecture, sans hacks. Phasage explicite réduisant le risque de dérive. Faiblesses / limites Frontend gameplay encore non implémenté (dépendance directe à Phase 6.1). Gestion de la concurrence documentée mais non arbitrée définitivement. Absence de temps réel pouvant impacter l'expérience à terme (acceptée mais réelle). Anti-cheat limité à V1 (suffisant pour lancement, mais plafonnant). Opportunités d'amélioration Sécuriser rapidement les arbitrages techniques restants pour fluidifier l'exécution. Formaliser des tests de concurrence ciblés sans étendre le scope. Verrouiller une librairie frontend tôt pour éviter des refontes UX/techniques. Risques principaux Blocage sur les arbitrages de concurrence (tie-breaks). Retard frontend si le périmètre Phase 6.1 n'est pas strictement cadré. Tentation d'introduire du temps réel ou de l'anti-cheat avancé prématûrement. Dépendance implicite au chat 00 pour valider certains choix structurants.

3. Recommandations stratégiques Arbitrer immédiatement la stratégie de concurrence tie-breaks → Recommandation prioritaire, nécessite validation explicite du chat 00. Lancer la Phase 6.1 frontend sans modification backend → Respect strict du périmètre figé pour éviter tout glissement. Formaliser une checklist technique de Phase 6.1 → Limiter le risque d'interprétation côté frontend. Geler explicitement l'absence de WebSockets et d'anti-cheat

V2 → Décision de non-implémentation à court terme pour protéger le planning. 4. Todo list structurée Priorité immédiate (bloquant / critique) Arbitrer la stratégie de gestion de concurrence tie-breaks (Option A vs B). Valider officiellement le périmètre Phase 6.1 côté frontend. Priorité court terme Choisir la librairie de plateau d'échecs frontend. Définir le niveau minimal de tests de concurrence à automatiser. Produire une checklist d'implémentation frontend gameplay. Priorité moyen terme Préparer (sans implémenter) un cadrage WebSockets Phase 6.2. Documenter les scénarios d'extension anti-cheat V2. Optionnel / évolutif Étudier incréments de temps spécifiques. Étudier tie-breaks avancés (Armageddon) hors lancement initial. 5. Dépendances et interactions Chat 00 – Chef de projet : Arbitrages techniques structurants (concurrence, temps réel). Chat 05 – Produit & UX : Alignement strict sur ce que le frontend doit afficher (sans logique métier). Chat 04 – Finance & paiements : Dépendance indirecte (finalisation tournoi), périmètre figé. Chat 03 – Juridique : Aucune dépendance opérationnelle directe à ce stade. Dépendances critiques Validation des arbitrages techniques par le chat 00 avant toute extension backend. 6. Indicateurs de complétion Prêt pour lancement Backend gameplay gelé et sans régression. Frontend Phase 6.1 fonctionnel sans logique métier locale. Arbitrages de concurrence validés et testés. Journalisation complète et exploitable. Prêt pour phase suivante Phase 6.1 stabilisée. Décision formelle sur temps réel (go / no-go). Backlog clair pour Phase 6.2 sans dette technique critique. Conclusion Le domaine Technique – 06 est solide, maîtrisé et prêt à redémarrer un cycle de travail. Les risques sont identifiés, les arbitrages sont localisés, et l'exécution peut reprendre immédiatement sans redéfinition du périmètre, sous réserve de décisions ciblées du chat 00. NB - Prends connaissance de ce prompt tu es le chat 06 dont il est question.

Pris en compte.

Je me positionne désormais **explicitement comme le chat [Technique] – 06**, avec les responsabilités, le périmètre et les contraintes décrits dans ce livrable. Le document que tu fournis est **cohérent, exploitable tel quel et suffisamment verrouillé** pour servir de socle d'exécution sans réinterprétation.

Ce que cela implique immédiatement (cadre d'action du chat 06)

Périmètre strict

- Je n'introduirai **aucune extension fonctionnelle** hors du cadre figé.
- Toute proposition sortant du périmètre (temps réel, anti-cheat V2, logique financière, UX étendue) sera **explicitement signalée comme hors scope** et conditionnée à une validation du chat 00.

Rôle opérationnel

- Clarifier, documenter et préparer l'exécution technique **sans modifier l'architecture backend gelée**.

- Aider à lever les arbitrages ouverts, en particulier ceux bloquants pour l'exécution (concurrence tie-breaks, Phase 6.1 frontend).
- Produire des artefacts techniques orientés pilotage : checklists, options comparées, critères de décision, risques.

État de référence que je considère comme acquis

FIGÉ (non rediscutable ici)

- Architecture backend-centric, stateless, DB source de vérité.
- Orchestration gameplay serveur complète et déterministe.
- Modèle de tie-breaks intégré au moteur tournoi.
- Anti-cheat V1 serveur strict.
- Phase 6.0.C backend implémentée, testée et gelée.

OUVERT / À ARBITRER (zones d'action du chat 06)

- Stratégie de concurrence tie-breaks (Option A / B).
- Implémentation frontend gameplay Phase 6.1 (sans logique métier).
- Choix de la librairie de plateau d'échecs.
- Niveau minimal de tests de concurrence.
- Décision formelle de non-implémentation WebSockets / anti-cheat V2 à court terme.

Prochaine étape logique (si tu valides)

Pour relancer efficacement le cycle, la séquence optimale est la suivante :

1. **Isoler l'arbitrage de concurrence tie-breaks**
→ Présentation option A vs B, critères, impacts, recommandation unique.
2. **Cadrer Phase 6.1 frontend**
→ Checklist d'implémentation exhaustive mais fermée (ce que le frontend fait / ne fait pas).
3. **Proposer un short-list de librairies chessboard**
→ Analyse strictement technique (intégration, maintenance, risques), sans débat UX.

Si tu le souhaites, je peux démarrer immédiatement par le point 1 (arbitrage concurrence tie-breaks), qui est clairement identifié comme **bloquant critique** et dépendant d'une validation du chat 00.