

Permalink URI:

vscode://joelabenhaim.aicode-ja/chat?sessionId=c035a516410f4d3e9f00896906a87d4f-RS-Affiner-Session-L

<USER>

[Refine]SPECIF FINALISER LIVESESSION POUR LES SESSIONS PROMPT NORMALES 20260712 013420

</USER>

<ASSISTANT>

I'm thinking...

</ASSISTANT>

<ASSISTANT>

🔧  reading project:  [liveSession.programming-doc-readme.md](#),  [isomorphism-invariant-rule-between-streaming-and-rehydration-of-chat-sessions.programming-doc-readme.md](#),  [thinking-bubble-and-hourglass-streaming-rules-invariants.programming-doc-readme.md](#),  [easyAgents.programming-doc-readme.md](#)

🔧  scanning git:  ["Vérifier le commit actuel"](#),  ["Vérifier le diff de base"](#)

🔧  reading project:  h ["Architecture liveSession"](#),  e ["État activité windows"](#),  e ["Ouvertures panneau prompt"](#),

 [LiveSessionRun.type.ts](#),

 [LiveSessionPanel.type.ts](#),

 [LiveSessionOpenResult.type.ts](#),  [LiveSessionRecord.type.ts](#),  [LiveSessionRegistry.service.ts](#) [L1-

L560],  [LiveSessionService.ts](#),  [LiveSessionOpen.service.ts](#),  [LiveSessionReplay.service.ts](#),  h ["Ouverture Prompt coordonnée"](#),  r ["openPanel"](#),  r ["registerPromptThreadCleanup"](#),  [openPanel.util.ts](#) [L1-

L740],

 [PromptThreadCleanup.manager.ts](#),

 [iaStreamHandlers.ts](#),

 [liveSessionStreamHandlers.ts](#),

 [SidebarQueryRunningBridge.ts](#),  [PromptQueryDoneSound.manager.ts](#),  [stream.endpoint.ts](#),  e ["Appels ouverture centrale"](#),  e ["Callbacks stream client"](#),  h ["Persistance continuation"](#),  [AiStreamStartArgs.type.ts](#),  [useAiStream.hook.ts](#) [L1-

L520],  [PromptLiveSession.types.ts](#),  [usePromptLiveSession.hook.ts](#) [L1-

L520],  [usePromptRunController.hook.ts](#) [L1-

L500],  [preparePromptStreamingTurn.util.ts](#),  [usePromptThinkingLifecycle.hook.ts](#) [L1-

L340],  [usePromptRunCommands.hook.ts](#) [L1-

L500],  [MessageRegistry.ts](#),  [SidebarStore.ts](#),  [useAiStream.hook.ts](#) [L500-

L751],  [usePromptRunController.hook.ts](#) [L480-L613],  [usePromptRunCommands.hook.ts](#) [L460-

L865],  [EasyAgentsBatchLiveSession.adapter.ts](#),  [Prompt.tsx](#) [L200-

L460],

 [SidebarUiMessages.ts](#),  [SidebarStore.rehydrate.manager.ts](#),  [EasyAgentsBatchRunManager.service.ts](#) [L220-

L290],  [abort.endpoint.ts](#),  [open.endpoint.ts](#),  e ["Construction child executor"](#),  [EasyAgentsBatchRunChildExecution.service.ts](#) [L1-

L520],

 [aiStopSuffix.util.ts](#),

 [useStreamListener.hook.ts](#),

 [frontMessageBus.util.ts](#),

 [closePromptPanel.util.ts](#),

 [delete.endpoint.ts](#),  [SidebarStore.handlers.manager.ts](#),  [SidebarActivityQuery.tsx](#),  e ["Nettoyage thread enregistré"](#),  [uiAiPromptHandlers.ts](#) [L90-L155],  [AiPromptUiMessages.ts](#),  e ["Cleanup dispose usages"](#),  e ["Sound cleanup usages"](#),  e ["Query window type usages"](#),  e ["Réhydratation query windows"](#),  e ["Running bridge usages"](#),  e ["Son stream usages"](#),  e ["Live run ref literals"](#),  [PromptLiveSessionBanner.tsx](#),  e ["Viewer mode usages"](#),  e ["Live session view model"](#),  e ["Register cleanup bus"](#),  e ["Open chat panel](#)

[bus"](#),  [uiGenericHandlers.ts](#),  [NewChat.command.ts](#),  [registerDeepLinkUriHandler.ts](#) [L110-

L160],  [EasyAgentsBatchLiveSessionAdapter.service.ts](#),  [Prompt.tsx](#) [L620-L760],  [PromptInput.tsx](#) [L320-

L470],  [PromptInput.tsx](#) [L700-L850],  e ["Prompt mode viewer"](#),  e ["Interactions live session"](#),  e ["Thread cleanup register"](#),  [Prompt.tsx](#) [L1230-L1337],  [PromptInput.tsx](#) [L1510-L1660],  e ["Live bootstrap query fields"](#),

 e ["Hydration skip live"](#),  e ["Prompt variables exactes"](#),  [Prompt.tsx](#) [L120-L210],  e ["Flux Slimbk erreurs"](#),  e ["Client stream Slimbk"](#),  [callLocalApi.generated.ts](#) [L80-L160],  e ["Erreur stream extension bus"](#),  e ["Bus streamed handler"](#),  [callLocalApi.generated.ts](#) [L160-L235],  e ["Fixtures fenêtres activité"](#),  e ["Accès windows activité"](#),  e ["Type query state"](#),  e ["isRunning query](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

["](#)

[item](#)", [SerialTaskQueue.ts](#), [e "Usage queues série"](#), [h "Queue par locator"](#), [slimbkBuildHeader.ts](#) [L320-L430], [callLocalApi.generated.ts](#) [L1-L80], [extTypedMessageBus.util.ts](#) [L320-L390]



specifying

"SPECIF_FINALISER_LIVESESSION_POUR_LES_SESSIONS_PROMPT_NORMALES_20260712_014742"

SPECIF_FINALISER_LIVESESSION_POUR_LES_SESSIONS_PROMPT_NORMALES_20260712_014742

Copy

Refine

Code

Verify

Markdown formattedMarkdown source

Spécification — Finaliser `liveSession` pour les sessions Prompt normales

Contexte

La tranche 1 correspond au sous-système fonctionnel présent au

commit `f65adc8e5172cc1e4f9781bf54ffa6dd089d3d33`. La réinspection a été reprise depuis zéro sur le dépôt courant au commit `7a39caa251158f63a6a34d6ca3cf2a0e82993425`. Le diff entre ces commits ne touche que quatre fichiers de génération d'instructions SPECIF et ne modifie aucun contrat ni chemin d'exécution visé ici. La base fonctionnelle `liveSession` reste donc celle de la tranche 1.

Cette fondation fournit déjà :

- un registre process-local de runs exacts ;
- un buffer ordonné complet de `AiStreamedChunk` ;
- des leases de viewer exactes ;
- les barrières distinctes `sourceSettled` et `handoffReady` ;
- l'ouverture live-safe sans reset de la mémoire Orch ;
- le replay backlog + tail ;
- la protection anti-ABA par époque de locator ;
- le terminal exact et la distinction `TOOL_USE` / vraie fin de tour ;
- les transports Slimbk et message bus génériques ;
- un hook frontend générique réutilisant Queue A, Queue B, les locks et les sanitizers Prompt ;
- un adaptateur EasyAgents qui conserve le comportement produit historique.

La tranche finale active ce mécanisme pour les sessions Prompt AICode normales en une seule implémentation cohérente.

Le couplage entre capture de la source, fermeture du panneau, projection Activité, réouverture, replay, contrôle Stop, son de fin et nettoyage terminal doit être livré atomiquement afin qu'aucun contrat intermédiaire incomplet ne soit introduit.

Le problème utilisateur est le suivant : un renderer Prompt peut devenir gris, notamment après une saturation mémoire Electron. Aujourd'hui, fermer ce panneau déclenche explicitement `Abort` → `AwaitStop` → `ClearMemory`, détruit la génération en cours et oblige à relancer le dernier message depuis zéro. Après cette tranche :

- fermer un panneau Prompt normal pendant une génération ne doit plus arrêter cette génération ;
- la génération continue dans l'extension ;
- la session reste visible dans Activité pendant son exécution ;
- cliquer la session dans Activité ou dans la liste des chats rouvre le Prompt sur la source exacte encore active ;
- le nouveau panneau reconstruit l'historique et rattrape le stream sans perte ni duplication ;
- si le panneau reste fermé jusqu'à la fin du processus, l'item disparaît d'Activité ;
- si le panneau est ouvert à la fin, l'item reste en état Idle comme aujourd'hui ;
- un bouton Stop apparaît sur une troisième ligne d'Activité uniquement pour une source Prompt normale, lorsque le panneau est fermé et que cette source est encore active.

La réinspection complète impose les invariants suivants :

- l'Activité distingue les runs `source` des runs `observer`, sinon une fermeture EasyAgents créerait à tort un item détaché avec bouton Stop ;
- le coordinateur de panneau n'importe jamais le low-level opener qui le compose, afin d'éviter un cycle ESM ;
- une query `liveSession` EasyAgents explicitement fournie par le caller est préservée et n'est jamais remplacée par une ouverture classique ;
- `sourceId` identifie la source backend exacte, tandis qu'un `streamInstanceId` frontend-only distingue chaque tentative de transport acceptée, y compris un retry de replay de la même source ;
- les callbacks frontend `onData`, `onComplete` et `onError`, y compris leurs continuations après `await`, sont gardés par `sourceId` + `streamInstanceId` ;
- le handoff direct normal exige un vrai terminal reçu comme donnée et une complétion transport normale ; une vraie erreur source transportée dans le premier header Slimbk doit donc être reconstruite par le handler extension, injectée dans sa

même pipeline FIFO comme chunk terminal, puis considérée comme une fin transport traitée normalement ; elle ne doit pas remonter au frontend comme une erreur de transport ;

- une erreur de transport non encodée ou une complétion sans vrai terminal déclenche la récupération et ne simule jamais un passage à `normal` ;
- `continueResponse` réutilise la bulle thinking agrégée existante et son texte de base ; cette initialisation est corrélée au `streamInstanceId`, y compris lors d'un retry transport de la même source ;
- la baseline de continuation fige les messages ciblés, mais jamais le brouillon, les attachments, `nextTurnId`, le titre ou les autres champs d'enveloppe susceptibles d'évoluer après le début du stream ; chaque Open superpose les messages immuables sur l'enveloppe persistée la plus récente ;
- les erreurs source transformées en header Slimbk utilisent un encodage réversible et versionné du `AiErrorMetaValue` ; le buffer et le forwarding extension reconstruisent exactement le même chunk ;
- le client Slimbk généré n'attend pas les callbacks `onChunk` asynchrones et appelle `onError` pour un header non-200 sans rejeter nécessairement sa promesse si le callback ne throw pas : le handler normal maintient donc une pipeline FIFO unique pour les chunks ordinaires et le terminal source décodé depuis le header, puis l'attend avant ses cleanups ;
- pour une source Prompt normale, `sourceSettled` et `handoffReady` ne sont pas publiés par l'endpoint backend avant que le handler extension ait drainé cette pipeline ;
- la finalisation normale du registre est atomique et exécutée par le handler extension après le drainage : réservation restante annulée comme non démarrée, ou record exact marqué `sourceSettled` puis `handoffReady` uniquement s'il possède baseline et vrai terminal ;
- le son de fin est corrélé au run exact `{ locator, turnId, sourceId }`, y compris pour le terminal source décodé depuis le premier header Slimbk ;
- la libération de lease lors du dispose est dispatchée par adaptateur : EasyAgents conserve `last viewer → kill`, tandis que `normalPrompt` libère seulement la lease et laisse la source continuer ;
- le coordinateur et la réhydratation Activité observent tout run actif, source ou observer, tandis que deletion/Retry conservent l'autorité plus étroite « source normale active ou réservée » ;
- la retraite terminale conserve l'époque terminale du run ; une éviction tardive de l'ancien run après le démarrage d'une source plus récente ne rend jamais l'ancien terminal de nouveau éligible au fallback persistant ;
- les suppressions de chat et le Retry EasyAgents consultent le registre `liveSession` comme autorité, et non uniquement le `SidebarStore` de présentation.

La tranche préserve également :

- aucun changement des algorithmes Queue A, Queue B, `PromptStreamLockStore` ou des sanitizers ;
- aucune duplication du timer Activité ou du son de fin lors d'un replay ;
- aucun second bouton Stop dans le Prompt ;
- aucune notion de panneau « invisible » : le panneau est `open` ou `closed` ;
- restauration du compositeur normal dès que le snapshot de base est installé et le replay armé ; pendant la réponse, seul le verrou Send normal reste actif, la saisie du prochain brouillon et la préparation des attachments restant autorisées ;
- aucun `*.generated.*` édité manuellement ;
- aucun fichier du répertoire documentaire personnel interdit chargé ou modifié ;
- commentaires, tests et identifiants ajoutés ou modifiés en anglais ;
- aucun `any`, `eslint-disable`, TODO, cast inutile, import dynamique de production ni compatibilité implicite non demandée.

Objectifs décrits par l'utilisateur

Besoin initial exprimé par l'utilisateur

- Livrer toute la tranche finale en une seule implémentation.
- Permettre à une génération Prompt normale de continuer lorsque son panneau est fermé.
- Conserver l'item dans Activité si le panneau est ouvert, ou fermé avec processus encore en cours.
- Supprimer l'item d'Activité si le panneau est fermé et le processus terminé.
- Ajouter une troisième ligne avec un bouton Stop uniquement pour `Running/source/closed`.
- Utiliser l'icône Stop existante, `size="small"`, `variant="danger"`, label Stop, avec désactivation immédiate après clic.
- Faire disparaître cette ligne dès la réouverture du panneau ou la fin du processus.
- Réutiliser dans le panneau reconnecté le Stop normal du compositeur.
- Restaurer le Prompt normal au plus tôt, sans read-only prolongé propre à la reconnexion.
- Une fois le replay armé, autoriser saisie et attachments du prochain message, tout en conservant le verrou Send normal.
- Étendre `SidebarStore` pour représenter les activités de session et pas uniquement les fenêtres ouvertes.
- Conserver le buffer raw-live complet, sans compression ni limitation ajoutée.

- Tester tout code créé, déplacé ou modifié, y compris les courses fermeture/réouverture/arrêt/retry transport/sources successives.
 - Mettre à jour la programming doc générique `liveSession` et les documentations d'isomorphisme et de thinking.
 - Respecter l'architecture modulaire, le typage strict et toutes les règles de qualité du projet.
- Clarifications implicites ajoutées par AICode**
- Le changement part fonctionnellement du commit `f65adc8e5172cc1e4f9781bf54ffa6dd089d3d33`; le commit inspecté `7a39caa251158f63a6a34d6ca3cf2a0e82993425` ne contient aucune évolution pertinente du sous-système.
 - `LiveSessionRunRef` devient `{ locator, turnId, sourceId }`. `turnId` reste la corrélation logique persistante ; `sourceId` identifie une invocation backend précise.
 - `streamInstanceId` reste frontend-only, change à chaque tentative acceptée et ne traverse aucun contrat backend.
 - `sourceId` est obligatoire dans toutes les routes/query strings `liveSession` et tous les guards exacts. Aucune opération exacte ne retombe sur le run actif d'un locator après échec d'identité.
 - `EasyAgents` utilise `sourceId = turnId`, son contrat garantissant une source unique par tour.
 - Pour Prompt normal, `useAiStream.start()` génère `sourceId` après acceptation locale du start et génère toujours un nouveau `streamInstanceId`.
 - Les callbacks stale sont ignorés avant toute mutation, après chaque `await` et avant tout reset de queue/lock.
 - Le registre ajoute des réservations de source entre l'entrée du handler extension et le retour réussi de `askStream(...)`. La réservation est la première mutation synchrone du handler.
 - Une réservation n'est pas replayable et ne viole pas « seul `registerRun` crée un record ».
 - Réservation → record conserve `sourceStartedAtMs` et `abortRequested`; un abort précoce est appliqué après priming/enregistrement, avant l'interprétation du premier chunk.
 - Le registre conserve des retraites structurées. Seule une retraite terminale dont l'époque reste courante autorise un fallback absent-record.
 - `baselineAndObservableChunk` reste la policy `EasyAgents` ; `baselineOnly` est utilisée pour les sources Prompt normales.
 - `replayMode` vaut `replaceTurnAssistants` pour prompt/retry/edit/specif/compact-context et `continueAssistant` pour `continueResponse`.
 - `continueResponse` capture sous sérialisation une baseline immuable des messages `OrchMemory` avant `askStream(...)`, valide l'assistant ciblé et ne fige pas l'enveloppe entière.
 - À chaque Open `continueAssistant`, le backend recharge l'enveloppe persistée la plus récente et remplace uniquement `messages` par la baseline défensive.
 - `markBaselineReady` accepte une baseline de messages uniquement pour `continueAssistant`, avec clones défensifs entrée/sortie.
 - Le lifecycle `Readable` lazy est extrait du child executor `EasyAgents` dans un service générique partagé.
 - Le stream normal est `ownership= 'source'` ; un replay n'ouvre jamais une seconde activité et ne joue jamais un second son.
 - Le handler source normal reste l'unique propriétaire du timer `Activité` et du son. Son forwarding ordinaire et le terminal source décodé depuis un header 500 encodé passent par la même pipeline FIFO.
 - Chaque chunk normal est appendu au registre avant exposition.
 - Le premier `meta.error` source est appendu tel quel au buffer. L'endpoint encode son `AiErrorMetaValue` dans le header 500. Le handler extension reconnaît cet encodage dans le `SlimbkCallError`, décode le chunk exact, l'enfile après tous les chunks précédents dans la pipeline FIFO, l'envoie à `Activité/son/frontend`, marque cette erreur comme source-terminal traitée et laisse l'appel stream se terminer normalement. Le frontend reçoit donc ce cas par `onData`, jamais par `onError`.
 - Les erreurs de transport ordinaires non encodées restent canonisées côté frontend en `SYSTEM_ERROR` et donnent `transportError`.
 - Les throws source après baseline et EOF sans terminal deviennent un `meta.error` canonique bufferisé/exposé.
 - Le service source ne retire pas le run de l'index actif. Le handler extension finalise le registre seulement après retour `Slimbk` et drainage FIFO.
 - Tant que cette finalisation n'a pas eu lieu, le run/réservation reste autoritatif pour coordinateur, `Activité`, Stop, deletion et cleanup.
 - Toute ouverture Prompt passe par une façade coordonnée commune. Une query live explicite valide garde priorité ; une query live explicite invalide ne se dégrade jamais silencieusement.
 - Le coordinateur est une classe à dépendances injectées et n'importe pas le low-level opener.
 - Ouverture, dispose et cleanup terminal sont sérialisés par locator avec queues FIFO supprimées à idle.

- Le low-level panneau retire synchroniquement le panneau du registre puis délègue les effets Prompt au coordinateur.
- Le coordinateur réévalue panneau et run actif immédiatement avant tout cleanup destructif.
- Le mapping cleanup Prompt est indexé par locator structurel complet.
- Le panneau reconnecté utilise une policy `normalPrompt`, distincte du viewer `EasyAgents` ; `liveSessionStreaming` n'implique plus automatiquement read-only.
- `not_ready` `normalPrompt` est retryé avec une seule requête Open en vol et un délai fixe annulable. `EasyAgents` conserve son échec actuel.
- `transportError` ou complétion sans vrai terminal conduit à `liveSessionReplayFailed`. Une erreur source encodée et décodée par le handler extension est au contraire un terminal de données avec complétion transport normale et peut suivre le handoff direct.
- Le Stop Prompt normal utilise une route source exacte ; l'ancien endpoint thread-only reste réservé aux nettoyages destructifs.
- Le Stop détaché valide synchroniquement item exact, panneau fermé, source exacte active/réservée, non-settled et claim non consommé, puis appelle Orch sans `await` intermédiaire.
- Le son garde l'option utilisateur par thread, mais corrèle l'état de run et la déduplication par `runRef` exact.
- `SidebarStore` devient une union stricte : Idle/open ; Running/observer/open ; Running/source/open|closed.
- `queryState.windows` devient `queryState.items` et `SidebarActivityQueryWindowItem` devient `SidebarActivityQueryItem`.
- La réhydratation fusionne panneaux ouverts, observers actifs et sources normales actives/réservées ; un observer sans panneau n'est pas projeté.
- Le registre, et non le `SidebarStore`, est l'autorité pour deletion et Retry destructifs.
- `continueResponse` réutilise une bulle assistant et une bulle thinking agrégée uniques, avec `baseThinkingTextRef`, séparateur canonique et reset par `streamInstanceId`.
- Aucun résultat généré n'est listé comme fichier à éditer.

Vue d'ensemble de la méthode choisie

Identité, réservations et machine d'état

Une source normale suit :

1. acceptation du start frontend ;
2. génération de `sourceId` et `streamInstanceId` ;
3. message bus vers l'extension ;
4. réservation exacte `ownership='source'` comme première mutation synchrone ;
5. publication Activité Running/source avec état réel du panneau ;
6. armement du son exact ;
7. persistance éventuelle de la préférence SPECIF Code ;
8. chargement catalogue ;
9. capture sérialisée des messages de continuation si nécessaire ;
10. retour réussi de `askStream(...)` ;
11. priming du `Readable` ;
12. réservation → record et application d'un abort précoce ;
13. append de chaque chunk avant exposition ;
14. baseline durable ;
15. replay possible ;
16. vrai terminal ou synthèse canonique ;
17. fin de consommation backend sans retrait du run actif ;
18. retour Slimbk et drainage de la pipeline FIFO extension ; si le premier résultat est un header 500 contenant le protocole privé, le handler le décode en chunk source terminal, l'enfile dans la même FIFO et ne le transforme pas en erreur transport ;
19. finalisation atomique post-forwarding ;
20. Activité Idle si panneau ouvert, suppression si fermé ;
21. cleanup mémoire/attachments/son/trivial uniquement sans panneau ni nouvelle source.

Replay, erreurs et baselines

`replaceTurnAssistants` retire les assistants du tour exact puis rejoue la source. `continueAssistant` conserve les messages exacts d'avant continuation, tout en rechargeant l'enveloppe persistée la plus récente à chaque Open.

Le protocole du premier `meta.error` est :

- le service source append le chunk original au buffer ;

- l'endpoint encode `AiErrorMetaValue` dans un header `500_SERVER_ERROR` versionné ;
- le client généré transmet ce header au callback `onError` sous forme de `SlimbkCallError` et n'attend pas les callbacks `async` ;
- le handler extension tente le décodage ; si le payload privé est valide, il reconstruit le chunk exact, l'ajoute à la même pipeline FIFO que les `onChunk` , met à jour son/Activité, l'émet au frontend et n'effectue aucun throw pour ce cas traité ;
- l'appel `Slimbk` se termine alors normalement, ce qui permet au frontend d'observer le vrai terminal via `onData` puis une complétion transport normale ;
- si le payload n'est pas le protocole privé valide, le handler conserve la sémantique d'erreur transport ; le frontend crée le `SYSTEM_ERROR` canonique ;
- un EOF source sans terminal devient `INTERRUPTED` avant l'endpoint et suit le chemin de données terminales normal. Le terminal bufferisé ne rend pas le run handoff-ready : la finalisation post-forwarding reste obligatoire.

Cycle de vie du panneau

Le wiring low-level compose le coordinateur, qui reçoit `opener` , panneau registry, `liveSession resolver` , `releases adapter-specific` , transitions `Activité` , `cleanup thread/son/trivial` et `UI chat` . Les releases :

- `easyAgentsBatch + stopOnLastViewerClose` → policy `EasyAgents` dernier viewer = hard stop ;
- `normalPrompt + continueOnPanelClose` → release de lease sans abort ;
- descripteur invalide → fail-fast, sans fallback ni cleanup destructif d'un run actif.

État	Panneau	Ownership	Projection
Idle	open	aucun	deux lignes, aucun Stop détaché
Running	open	observer	deux lignes, UX <code> EasyAgents </code> inchangée
Running	open	source	deux lignes, Stop normal Prompt
Running	closed	source	trois lignes, Stop détaché <code> Activité </code>
terminal	closed	aucun	aucun item

Frontend reconnecté

Après installation du snapshot et armement du replay, le compositeur et les attachments redeviennent visibles. `isStreaming` maintient le verrou `Send` et le `Stop` normal. Chaque retry possède un nouveau `streamInstanceId` . Les callbacks antérieurs ne peuvent plus muter le runtime. Le handoff normal est direct uniquement après vrai terminal de données, drainage `Queue A/Queue B` , finalisation `Prompt` et outcome `completed` de la tentative courante.

Activité, son et Stop

Le `Stop` détaché utilise `VscButton : icon="SquareStop", size="small", variant="danger", label Stop, disableAfterClick, enabled={!stopRequested}, testId` stable basé sur `sourceId` . Le lien de titre reçoit un `testId` stable basé sur le locator. Le son et les transitions `Activité` consomment le `runRef` exact et la même séquence de chunks que le frontend, y compris le terminal reconstruit depuis le premier header d'erreur source.

Liste exacte des fichiers avec détails d'implémentation

Contrats shared

1) `AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionReplay.type.ts` — créé

- Définir `ZLiveSessionReplayMode / LiveSessionReplayMode : replaceTurnAssistants | continueAssistant.`

- Documenter la reconstruction frontend de chaque branche.

2) `AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionRun.type.ts` — modifié

- Ajouter `LiveSessionSourceId` strict trim/non vide et `sourceId` obligatoire au `runRef` .
- Inclure `sourceId` dans la run key, sans modifier la locator key.
- Documenter `turnId` logique et `sourceId` exécutionnel.

3) `AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionPanel.type.ts` — modifié

- Ajouter paramètres source/adaptateur et adaptateur strict `easyAgentsBatch | normalPrompt` .
- Réutiliser le schéma `sourceId` ; rendre `continueOnPanelClose` effectif pour `normalPrompt` .

4) `AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionOpenResult.type.ts` — modifié

- Ajouter `sourceId` aux branches `live/terminal` et `replayMode` uniquement à `live` .
- Étendre transport et `superRefine` , y compris propriété explicitement présente à `undefined` .

5) `AICode/src/sharedlib/src/types/liveSessionTypes/__test__/LiveSessionContracts.test.ts` — modifié

- Couvrir clés, collisions, query params, adaptateurs, replay modes, transports et `sourceId` obligatoire.

6) `AICode/src/sharedlib/src/types/aiTypes/aiStopSuffix.util.ts` — modifié

- Ajouter fabrique canonique des `meta.error` transport/source.
- Ajouter encodeur/décodeur privé, versionné, réversible de `AiErrorMetaValue` .

- Le décodeur accepte `Error`, objet structurel `SlimbKCallError`, chaîne et stack enveloppée ; validation stricte du payload.
- Produire `SYSTEM_ERROR` / `INTERRUPTED` canoniques pour les erreurs non encodées.
- Garantir que le chunk décodé par l'extension est identique au chunk bufferisé.
- 7) `AICode/src/sharedlib/src/types/aiTypes/__test__/aiStopSuffix.util.test.ts` — **modifié**
- Tester `Error/string/status`, réversibilité, `stacks/prefixes`, détails multilignes, invalide, `SYSTEM_ERROR`, `INTERRUPTED` et identité du chunk.
- 8) `AICode/src/sharedlib/src/types/busTypes/ui/SidebarStore.ts` — **modifié**
- Remplacer le type fenêtre par l'union stricte `SidebarActivityQueryItem`.
- Branches `Idle/open`, `Running observer/open`, `Running source/open|closed` avec identités exactes et `stopRequested`.
- Préserver `titre`, `elapsed`, `updatedAt`, `openedAt`; renommer `windows` en `items`.
- 9) `AICode/src/sharedlib/src/types/busTypes/ui/SidebarUiMessages.ts` — **modifié**
- Ajouter `openQueryActivity(locator)` et `stopDetachedQuery(runRef) -> {accepted}`.
- 10) `AICode/src/sharedlib/src/types/busTypes/global/MessageRegistry.ts` — **modifié**
- Ajouter `sourceId` aux streams `normal` et `liveSession` ; ne jamais transporter `streamInstanceId`.
- Backend Orch liveSession**
- 11) `AICode/src/extension/src/aiTools/orch/liveSession/liveSession.programming-doc-readme.md` — **modifié**
- Documenter identité, réservations, `readiness`, `replay`, `continuation`, erreur header décodée dans le handler extension, `ownership`, `lifecycle`, `Activity`, son, retraits et barrière `post-forwarding`.
- 12) `AICode/src/extension/src/aiTools/orch/liveSession/registry/LiveSessionRecord.type.ts` — **modifié**
- Ajouter `baselineOnly`, `replayMode`, `baseline continuation`, `timestamp`, `abortRequested`, `viewerGroup` optionnel, `réservations/snapshots/retraits/époque terminale`.
- Étendre `permits/subscriptions` à `sourceId` et interdire les `baselines` incohérentes.
- 13) `AICode/src/extension/src/aiTools/orch/liveSession/registry/LiveSessionRegistry.service.ts` — **modifié**
- `Réservations/cancel` exacts, transformation atomique vers `record`, `listings` actifs génériques/`source`, existence et `claims` atomiques.
- Supprimer `canAbortSource`.
- Finalisation normale `post-forwarding` `idempotente` et exacte.
- `Retraites` structurées et époques empêchant la résurrection d'un ancien terminal.
- Préserver `leases`, `retention` et `replay` tranche 1.
- 14) `AICode/src/extension/src/aiTools/orch/liveSession/registry/__test__/LiveSessionRegistry.service.test.ts` — **modifié**
- Couvrir sources successives, `réservations`, `baselineOnly`, `baseline messages`, `abort`, `listings`, `claims`, finalisation, `retraites`, époques et `A-terminal/B-start/A-eviction` tardive.
- 15) `AICode/src/extension/src/aiTools/orch/liveSession/core/LiveSessionService.ts` — **modifié**
- Exposer toutes les nouvelles opérations du registre, supprimer `canAbortSource`, conserver `Open/Replay/release/sweep/evict`.
- 16) `AICode/src/extension/src/aiTools/orch/liveSession/core/__test__/LiveSessionService.test.ts` — **modifié**
- Couvrir parcours `normal`, `réservations`, `réouverture`, sources successives, `claims`, finalisation et `retraite` stale.
- 17) `AICode/src/extension/src/aiTools/orch/liveSession/open/LiveSessionOpen.service.ts` — **modifié**
- `not_ready` pour réservation exacte.
- `replaceTurnAssistants` : filtrage assistants du tour sur enveloppe actuelle.
- `continueAssistant` : enveloppe persistée récente + remplacement du seul tableau messages.
- Retourner `sourceId/replayMode` et contrôler le `fallback` terminal par `retraite+époque`.
- 18) `AICode/src/extension/src/aiTools/orch/liveSession/open/__test__/LiveSessionOpen.service.test.ts` — **modifié**
- Couvrir réservation, `baselineOnly`, `continuation/enveloppe` récente, `mismatch`, sources mêmes `turn`, `retraites` et état `live` avant finalisation `post-forwarding`.
- 19) `AICode/src/extension/src/aiTools/orch/liveSession/replay/__test__/LiveSessionReplay.service.test.ts` — **modifié**
- Ajouter `sourceId`, isolation ancienne `subscription/nouvelle source` et attente de finalisation après terminal drainé.

20) `AIcode/src/extension/src/aiTools/orch/liveSession/source/LiveSessionSourceLifecycle.service.ts` — **créé**

- Extraire le lifecycle lazy Readable OOP générique : création, first next, source-start sync, first result/tail, cleanup/abort/return/destroy, settlement produit exact et agrégation déterministe.
- Ne contenir aucune sémantique EasyAgents/Prompt ni publication registre normalPrompt.

21) `AIcode/src/extension/src/aiTools/orch/liveSession/source/__test__/LiveSessionSourceLifecycle.service.test.ts` — **créé**

- Généraliser les tests priming/cleanup et vérifier que le callback produit reste l'unique frontière de settlement.

22) `AIcode/src/extension/src/aiTools/orch/liveSession/source/NormalPromptLiveSessionSource.service.ts` — **créé**

- Mapper AiQuery → replayMode, readiness baselineOnly.
- Capturer sous queue la baseline messages de continuation et valider l'assistant.
- Enregistrer après priming, transmettre timestamp, appliquer abort réservé.
- Parser, append-before-yield, baseline, vrai terminal.
- Premier meta.error appendu inchangé puis exposé à l'endpoint pour encodage header.
- Transformer throw/EOF sans terminal en erreur canonique.
- Ne pas finaliser le registre ; garantir qu'un échec registration ne touche aucun tiers.

23) `AIcode/src/extension/src/aiTools/orch/liveSession/source/__test__/NormalPromptLiveSessionSource.service.test.ts` — **créé**

- Couvrir tous les query types, continuation, append order, abort, premier error/header, throw, EOF, absence de settlement/handoff avant finalisation et isolation registration.

Adaptation EasyAgents

24) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/liveSession/EasyAgentsBatchLiveSessionAdapter.service.ts` — **modifié**

- `replayMode='replaceTurnAssistants'`, `sourceId=turnId`, timestamp.
- Injecter lifecycle partagé ; supprimer `canAbortSource`.
- Préserver settlement/handoff métier EasyAgents.

25) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/liveSession/__test__/EasyAgentsBatchLiveSessionAdapter.service.test.ts` — **modifié**

- Vérifier identité, replayMode, timestamp, nouveau claim et ordre historique.

26) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/EasyAgentsBatchRunExecution.service.ts` — **modifié**

- Construire tous les runRefs avec `sourceId=liveTurnId`, sans changer les transitions Batch.

27) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunExecution.live-activity-and-gating.service.test.ts` — **modifié**

- Étendre assertions identité/replayMode et préserver Watchable/Handoff.

28) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/EasyAgentsBatchRunChildExecution.service.ts` — **modifié**

- Remplacer lifecycle local par service partagé, conserver parsing/classification/callbacks.

29) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunChildExecution.service.test.ts` — **modifié**

- Conserver intégration Batch et déplacer tests lifecycle purs.

30) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/EasyAgentsBatchRunManager.service.ts` — **modifié**

- Injecter lifecycle, migrer items, consulter `hasActiveOrReservedSourceForLocator` pour Retry.

31) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunExecution.testHarness.ts` — **modifié**

- Injecter lifecycle et runRefs sourceId.

32) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunLifecycle.service.test.ts` — **modifié**

- Adapter construction au lifecycle partagé.

33) `AIcode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchLastViewerStop.service.test.ts` — **modifié**

- Ajouter sourceId et préserver courses last-viewer/ancien-nouveau run.

34) `AIcode/src/extension/src/aiTools/easyAgents/easyAgents.programming-doc-readme.md` — **modifié**

- Documenter `sourceId=turnId`, lifecycle partagé, Activity observer/open et zéro viewer=hard stop.

Endpoints et message bus

- 35) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/open.endpoint.ts` — **modifié**
 - Exiger `sourceId` et transporter `sourceId/replayMode`.
- 36) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/open.endpoint.test.ts` — **modifié**
 - Couvrir champs, branches et mismatch.
- 37) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/stream.endpoint.ts` — **modifié**
 - Exiger `sourceId` et relayer `runRef` complet.
- 38) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/stream.endpoint.test.ts` — **modifié**
 - Couvrir exactitude, mismatch, terminal et attente de finalisation post-forwarding.
- 39) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/abort.endpoint.ts` — **modifié**
 - Exiger `sourceId` ; claim viewer atomique avec lease active ; claim et abort dans la même pile.
- 40) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/abort.endpoint.test.ts` — **modifié**
 - Couvrir viewer `EasyAgents/normal`, stale, double Stop, remplacement et idempotence.
- 41) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/abortSource.endpoint.ts` — **créé**
 - Stop exact source Prompt normal par locator+turn+source, claim puis abort sans await.
- 42) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/abortSource.endpoint.test.ts` — **créé**
 - Couvrir réservation, record, stale, double clic, absent et ordre.
- 43) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/query/ask/stream.endpoint.ts` — **modifié**
 - Ajouter `sourceId`, préserver ordre SPECIF/catalogue, déléguer au service source.
 - Premier meta.error : header 500 avec payload privé ; ne jamais bufferiser une autre représentation.
 - Ne pas finaliser le registre ; terminer après consommation/canonisation.
- 44) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/query/ask/__test__/stream.endpoint.test.ts` — **modifié**
 - Tester délégation, ordre, premier chunk, header encodé, absence double forwarding et absence de finalisation prématurée.
- 45) `AICode/src/extension/src/messageBus/handlers/ia/liveSessionStreamHandlers.ts` — **modifié**
 - Relayer `sourceId`.
 - Source ownership : aucune activité/son supplémentaire.
 - Observer : ressources uniquement si panneau exact ouvert.
 - RunRef exact au son ; conserver pipeline async et cleanup partiel.
- 46) `AICode/src/extension/src/messageBus/handlers/ia/__test__/liveSessionStreamHandlers.test.ts` — **modifié**
 - Couvrir `sourceId`, ownership, observer fermé, son exact, absence duplication et ordre.
- 47) `AICode/src/extension/src/messageBus/handlers/ia/iaStreamHandlers.ts` — **modifié**
 - Recevoir `sourceId` et construire `runRef` exact.
 - Réserver synchroniquement avant tout await, publier Activity et armer son exact.
 - Transmettre `sourceId` à l'endpoint.
 - Maintenir une FIFO unique. `onChunk` y enfile le chunk ordinaire. `onError` tente d'abord de décoder le protocole privé : si valide, il enfile le chunk terminal exact dans cette même FIFO, le fait passer par Activity/son/emit, mémorise que l'erreur source a été traitée et retourne sans throw ; sinon il conserve l'erreur transport et la propage.
 - Après retour Slimbk, attendre la FIFO. Une erreur source encodée traitée n'est pas un `transportError`; toute autre erreur l'est.
 - Finaliser atomiquement le registre après FIFO, avant son Stop et Activity Stop.
 - Exécuter tous les cleanups, agréger les erreurs dans l'ordre, puis coordinateur `onSourceSettled`.
- 48) `AICode/src/extension/src/messageBus/handlers/ia/__test__/iaStreamHandlers.queryDoneSound.test.ts` — **modifié**
 - Vérifier `runRef/sourceId`, FIFO avant finalisation/son, fermeture sans cleanup prématuré, source suivante.
 - Ajouter premier meta.error encodé : décodage extension, son d'erreur unique, émission data avant done, aucune erreur transport frontend.
 - Ajouter terminal backend produit mais callback encore en transit.

49) `AIcode/src/extension/src/messageBus/handlers/ia/__test__/iaStreamHandlers.runIdPropagation.test.ts` — **modifié**

- Vérifier réservation, Activity, FIFO, finalisation, cleanup et ordre strict.
- Vérifier que le terminal décodé depuis header rejoint la FIFO après les chunks déjà reçus et avant finalisation.

50) `AIcode/src/extension/src/extension/panels/prompt/PromptQueryDoneSound.manager.ts` — **modifié**

- Corréler état actif/dédup par runRef exact, callbacks stale no-op, pas de création implicite.
- Préserver TOOL_USE/ABORTED/fréquences et traiter l'erreur source décodée comme le même terminal que le buffer.

51) `AIcode/src/extension/src/extension/panels/prompt/__test__/PromptQueryDoneSound.manager.test.ts` — **modifié**

- Couvrir mêmes turns/sources différentes, stale chunk/stop et absence de création tardive.

Activité runtime extension

52) `AIcode/src/extension/src/messageBus/handlers/ia/SidebarQueryRunningBridge.ts` — **modifié**

- Renommer classe interne `SidebarQueryActivityRuntimeService` sans déplacer le fichier.
- Stocker runRef, ownership, timer, startMs, runId par locator ; guards sourceId+runId.
- Source open/closed ; observer open seulement ; piloter transitions de l'union.

53) `AIcode/src/extension/src/messageBus/handlers/ia/__test__/SidebarQueryRunningBridge.race.test.ts` — **modifié**

- Couvrir sources successives, stale, observer fermé et panel source fermé/ouvert.

54) `AIcode/src/extension/src/messageBus/handlers/ia/__test__/SidebarQueryRunningBridge.usage.test.ts` — **modifié**

- Usage exact et transition Idle/suppression.

Coordination panneau Prompt

55) `AIcode/src/extension/src/extension/panels/prompt/liveSession/PromptLiveSessionPanelQuery.util.ts` — **créé**

- Parser locator et descripteur complet, construire query normalPrompt, distinguer classique/explicite valide/invalid, retirer autoRun lors injection.

56) `AIcode/src/extension/src/extension/panels/prompt/liveSession/__test__/PromptLiveSessionPanelQuery.util.test.ts` — **créé**

- Couvrir identité, adapter, ordre, politiques, invalide, suppression autoRun.

57) `AIcode/src/extension/src/extension/panels/prompt/liveSession/PromptLiveSessionPanelCoordinator.service.ts` — **créé**

- Classe OOP injectée, aucun import low-level.
- Queue FIFO par locator supprimée à idle.
- `openPromptPanel`, `onPanelDisposed`, `onSourceSettled` avec réévaluation avant cleanup.
- Préserver live explicite ; injecter normalPrompt seulement sans descripteur et avec source exacte active/réservée.
- Dispatch release EasyAgents/normalPrompt, detach source sans abort, ignorer settlement stale.

58) `AIcode/src/extension/src/extension/panels/prompt/liveSession/__test__/PromptLiveSessionPanelCoordinator.service.test.ts` — **créé**

- Couvrir priorités, releases, query invalide, races open/cleanup, pré-baseline, observer, source suivante, drainage post-terminal et protection mémoire/son.

59) `AIcode/src/extension/src/extension/panels/prompt/liveSession/PromptLiveSessionDetachedStop.service.ts` — **créé**

- Valider synchroniquement closed + Running/source exact + source active/réservée + claim non consommé ; mutation stopRequested puis abort sans await.

60) `AIcode/src/extension/src/extension/panels/prompt/liveSession/__test__/PromptLiveSessionDetachedStop.service.test.ts` — **créé**

- Couvrir accept/refus, reopened, remplacement, double clic, réservation, store stale.

61) `AIcode/src/extension/src/extension/panels/prompt/PromptThreadCleanup.manager.ts` — **modifié**

- Indexer par locator structurel ; séparer register, detach actif, cleanup historique, cleanup terminal sans abort/await et orphan cleanup.
- Ne jamais Abort/ClearMemory une source normale active.

62) `AIcode/src/extension/src/extension/panels/prompt/__test__/PromptThreadCleanup.manager.attachmentsCleanup.test.ts` — **modifié**

- Couvrir scopes, detach, idle, terminal, mapping absent et sérialisation attachments.

63) `AIcode/src/extension/src/extension/panels/core/openPanel.util.ts` — **modifié**

- Composer coordinateur et exposer façade Prompt async, conserver low-level autres pages.
- Capture descriptor initial ; dispose unregister panneau synchronique puis délégation complète.

- Adapter titre Running à `items/status`.
- 64) `AICode/src/extension/src/extension/panels/core/__test__/openPanel.util.test.ts` — **modifié**
- Wiring sans cycle, identité/adaptateur, reveal, dispose unique, unregister avant coordinateur, ports release, pages non-Prompt et fixtures `items`.
- 65) `AICode/src/extension/src/messageBus/handlers/ui/uiGenericHandlers.ts` — **modifié**
- Await façade coordonnée pour Prompt, low-level pour autres pages.
- 66) `AICode/src/extension/src/messageBus/handlers/ui/__test__/uiGenericHandlers.openExternalUrl.test.ts` — **modifié**
- Mockeur façade Prompt et préserver URL.
- 67) `AICode/src/extension/src/extension/commands/NewChat.command.ts` — **modifié**
- Await ouverture coordonnée.
- 68) `AICode/src/extension/src/extension/commands/__test__/NewChat.command.test.ts` — **modifié**
- Vérifier async et propagation erreur.
- 69) `AICode/src/extension/src/extension/uriHandlers/registerDeepLinkUriHandler.ts` — **modifié**
- Deep links chat via façade coordonnée, non-bloquant avec catch explicite.
- 70) `AICode/src/extension/src/extension/uriHandlers/__test__/registerDeepLinkUriHandler.scope.test.ts` — **modifié**
- Couvrir root/nested et guards.
- SidebarStore et Activité**
- 71) `AICode/src/extension/src/extension/panels/sidebar/SidebarStore.queryWindow.listeners.manager.ts` — **modifié**
- Transitions atomiques `panelOpened`, `sourceStarted`, `sourceSettled`; lecture `panelState` et `sourceStarted` sans await intermédiaire.
- 72) `AICode/src/extension/src/extension/panels/sidebar/SidebarStore.queryWindow.mutations.manager.ts` — **modifié**
- `windows→items`, `panelClosed`, `forceRemove`, `elapsed` et `stopRequested` exacts ; conserver seulement Running/source fermé.
- 73) `AICode/src/extension/src/extension/panels/sidebar/SidebarStore.core.manager.ts` — **modifié**
- Initialiser `items` et appeler `rehydrateQueryActivities`.
- 74) `AICode/src/extension/src/extension/panels/sidebar/SidebarStore.rehydrate.manager.ts` — **modifié**
- Fusionner panneaux, observers actifs et sources/réservations ; dédupliquer et interdire observer/closed/idle closed.
- Elapsed source depuis timestamp ; terminal bufferisé non finalisé reste Running.
- 75) `AICode/src/extension/src/extension/panels/sidebar/__test__/SidebarStore.rehydrate.manager.test.ts` — **modifié**
- Couvrir toutes les projections, réservation, observer sans panneau, dédup et fenêtre post-backend/pré-forwarding.
- 76) `AICode/src/extension/src/extension/panels/sidebar/__test__/SidebarStore.manager.queryRunningUpsert.test.ts` — **modifié**
- Tester machine d'état et identités.
- 77) `AICode/src/extension/src/extension/panels/sidebar/SidebarStore.chatTitles.manager.ts` — **modifié**
- Lire `items` et préserver fallback title source fermée/running.
- 78) `AICode/src/extension/src/extension/panels/sidebar/SidebarStore.handlers.manager.ts` — **modifié**
- Enregistrer open activity et Stop détaché, déléguer coordinateur/service.
- 79) `AICode/src/extension/src/extension/panels/sidebar/__test__/SidebarStore.handlers.manager.test.ts` — **créé**
- Vérifier payloads, accepted et délégation.
- 80) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/delete.endpoint.ts` — **modifié**
- Lire `items` ; bloquer si projection Running ou source active/réservée ; fermer/supprimer après validation autoritative.
- 81) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/__test__/delete.endpoint.liveSession.test.ts` — **créé**
- Couvrir open/closed/réservation/store stale/absent et terminal bufferisé avant drainage.
- 82) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/__test__/list.endpoint.windowsLock.eperm.test.ts` — **modifié**
- Adapter fixtures `items`.
- 83) `AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/__test__/diskAliases.lastWins.endpoints.test.ts` — **modifié**
- Adapter `items` et préserver delete disque.

- 84) `AICode/src/extension/src/aiTools/orch/context/persistence/__test__/ChatPersistenceWriter.windowsFallback.test.ts` — **modifié**
- Adapter fixture items.
- 85) `AICode/src/extension/src/aiTools/orch/context/memory/core/mem/__test__/OrchMemory.naive_title.user_prompt_specif_multiline.e2e.repro.test.ts` — **modifié**
- Adapter fixture items.
- 86) `AICode/src/frontend/src/app/sidebar/sidebar-activity/SidebarActivityQuery.tsx` — **modifié**
- Utiliser items/union ; ouvrir via nouveau message ; Stop exact sur troisième ligne source/closed.
- `SquareStop`, small, danger, `disableAfterClick`, `enabled`, `testIds` ; sous-composant/callbacks mémorisés sans hooks dans map.
- 87) `AICode/src/frontend/src/app/sidebar/sidebar-activity/__test__/SidebarActivityQuery.test.tsx` — **modifié**
- Couvrir locators, observer/source, Stop, disparition, payload, désactivation et `testIds` stables sans labels fragiles.
- Frontend liveSession et Prompt**
- 88) `AICode/src/frontend/src/app/prompt/liveSession/adapters/NormalPromptLiveSession.adapter.ts` — **créé**
- Parser `normalPrompt`, `sourceId`, lease, close policy.
- Politiques `not_ready` retry, handoff direct, replay recovery, surface interactive ; aucune action Stop banner.
- 89) `AICode/src/frontend/src/app/prompt/liveSession/adapters/__test__/NormalPromptLiveSession.adapter.test.ts` — **créé**
- Couvrir parsing, politiques et view models.
- 90) `AICode/src/frontend/src/app/prompt/liveSession/adapters/PromptLiveSessionBootstrap.adapter.ts` — **créé**
- Dispatch strict `EasyAgents/normalPrompt`, aucun fallback.
- 91) `AICode/src/frontend/src/app/prompt/liveSession/adapters/__test__/PromptLiveSessionBootstrap.adapter.test.ts` — **créé**
- Couvrir dispatch, partiels et inconnus.
- 92) `AICode/src/frontend/src/app/prompt/liveSession/core/PromptLiveSession.types.ts` — **modifié**
- Ajouter politiques et `liveSessionReplayFailed` ; supprimer `mode⇒viewer` et helper mort ; `runRef sourceId`.
- 93) `AICode/src/frontend/src/app/prompt/liveSession/core/usePromptLiveSession.hook.ts` — **modifié**
- Transporter `sourceId/replayMode`, retry `not_ready` annulable, replay selon mode.
- Handoff direct normal uniquement vrai terminal data + queues/finalisation + outcome `completed`.
- `transportError` ou `completed` sans terminal → `replayFailed`.
- Une erreur source encodée ne passe pas ici par `onError` : elle arrive comme chunk data décodé/forwardé par l'extension et peut donc satisfaire le handoff direct.
- Chaque retry recharge baseline et crée un nouveau `streamInstanceId`.
- 94) `AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.hydration.test.tsx` — **modifié**
- Ajouter identité/replayMode et préserver `EasyAgents`.
- 95) `AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.handoff.test.tsx` — **modifié**
- Vérifier `EasyAgents` terminal, normal direct, refus `transportError/completed` sans terminal et acceptation d'un `meta.error` reçu comme data avec `completed`.
- 96) `AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.normalPrompt.test.tsx` — **créé**
- Couvrir `not_ready`, surface interactive, draft/attachments, Stop, replay failure/retry, direct handoff, terminal et close/reopen.
- 97) `AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.normalPromptIsomorphism.test.tsx` — **créé**
- Comparer stream continu/replay/hydratation pour tous query types, thinking, tools, usage, edits, erreurs canoniques et EOF.
- 98) `AICode/src/frontend/src/app/prompt/liveSession/adapters/EasyAgentsBatchLiveSession.adapter.ts` — **modifié**
- Ajouter `sourceId=turnId/adaptateur`, préserver UX.
- 99) `AICode/src/frontend/src/app/prompt/liveSession/adapters/__test__/EasyAgentsBatchLiveSession.adapter.test.ts` — **modifié**
- Adapter query, `sourceId` et dispatcher.
- 100) `AICode/src/frontend/src/messageBus/AiStreamStartArgs.type.ts` — **modifié**
- Garder `promptQuery` public sans `sourceId` ; état interne `runRef+streamInstanceId` ; outcome `pending|completed|transportError`.

101) AICode/src/frontend/src/messageBus/useAiStream.hook.ts — modifié

- Générer identités après acceptation, callbacks par tentative et guards stale complets.
- Exposer runRef, streamInstanceId, outcome.
- Stop exact selon source/observer.
- Les erreurs source encodées ne sont pas décodées ici : elles sont déjà reçues via onData comme meta.error exact après décodage extension.
- onError est réservé aux erreurs de transport non traitées et utilise la fabrique canonique SYSTEM_ERROR.
- Ne modifier aucune Queue/lock/sanitizer/raw-live.

102) AICode/src/frontend/src/messageBus/__test__/useAiStream.hook.liveSessionObserver.test.ts — modifié

- Ajouter identité, routes Stop, outcome, retry/stale ; vérifier qu'un meta.error reçu par data donne completed, tandis qu'un vrai onError donne transportError.

103) AICode/src/frontend/src/messageBus/__test__/useAiStream.hook.promptSourceIdentity.test.ts — créé

- Vérifier unicité, stabilité, abort exact, callbacks stale et observer retry.

104) AICode/src/frontend/src/app/prompt/Prompt/usePromptRunController.hook.ts — modifié

- Observer reçoit runRef+lease+replayMode ; base thinking + streamInstanceId ; exposer outcome/tentative ; reset complet.

105) AICode/src/frontend/src/app/prompt/Prompt/__test__/usePromptRunController.liveSessionObserver.test.tsx — modifié

- Couvrir identité, modes, base thinking, tentative et outcome.

Continuation et thinking

106) AICode/src/frontend/src/app/prompt/Prompt/preparePromptContinuationStreaming.util.ts — créé

- Trouver assistant non-thinking exact, capturer bases assistant/thinking, réutiliser thinking agrégé ou créer placeholder, réutiliser UID assistant, armer séparateur, reset refs/buffers/notices.

107) AICode/src/frontend/src/app/prompt/Prompt/__test__/preparePromptContinuationStreaming.util.test.ts — créé

- Couvrir assistant, bases, thinking, séparateur, manquant, sources successives, aucune duplication.

108) AICode/src/frontend/src/app/prompt/Prompt/preparePromptStreamingTurn.util.ts — modifié

- Reset explicite base thinking/séparateur pour replace.

109) AICode/src/frontend/src/app/prompt/Prompt/__test__/preparePromptStreamingTurn.util.test.ts — modifié

- Vérifier resets.

110) AICode/src/frontend/src/app/prompt/Prompt/usePromptThinkingLifecycle.hook.ts — modifié

- Recevoir base thinking et streamInstanceId ; initialiser avant effets transport ; séparateur unique ; préserver historique ; reset replace.

111) AICode/src/frontend/src/app/prompt/Prompt/__test__/usePromptThinkingLifecycle.continuation.test.tsx — créé

- Couvrir base, append, tool/text avant thinking, réinsertion, reset et retry même source/nouvelle tentative.

112) AICode/src/frontend/src/app/prompt/Prompt/usePromptToolUse.hook.ts — modifié

- Supprimer bulle vide uniquement si baseThinking vide.

113) AICode/src/frontend/src/app/prompt/Prompt/usePromptRunController.finalization.hook.ts — modifié

- Reset base thinking/séparateur après terminal, ordre inchangé.

114) AICode/src/frontend/src/app/prompt/Prompt/usePromptRunCommands.hook.ts — modifié

- Remplacer bootstrap continue inline par helper ; préserver autres commandes.

115) AICode/src/frontend/src/app/prompt/Prompt/__test__/usePromptRunCommands.continueResponse.test.tsx — créé

- Vérifier helper, même turn, nouvelle source/tentative, thinking unique, assistant exact.

Intégration Prompt et documentation

116) AICode/src/frontend/src/app/prompt/Prompt/Prompt.tsx — modifié

- Capturer source/adaptateur dans query figée, dispatcher, passer outcome/runRef/tentative.
- Déduire interactions de surface/policy ; draft/interactions normaux pendant replay normalPrompt.
- Cleanup ownership différé jusqu'au handoff direct ; hydratation classique off pour tout live bootstrap.

117) AICode/src/frontend/src/app/prompt/Prompt/__test__/Prompt.liveSessionBootstrap.test.tsx — modifié

- Couvrir dispatch, identité, surface, draft, query figée, hydratation off, replay failure, ownership cleanup.

118) `AIcode/src/frontend/src/app/prompt/PromptInput/__test__/PromptInput.batchLiveViewerMode.test.tsx` — **modifié**

- Préserver viewer EasyAgents et ajouter normalPrompt interactif avec Stop normal, sans banner Stop.

119) `AIcode/src/frontend/src/app/easy-agents/easy-agents-batch/core/__test__/EasyAgentsBatch.liveWatch.test.tsx` — **modifié**

- Adapter query exacte source/adaptateur.

120) `AIcode/src/frontend/src/app/easy-agents/easy-agents-item/core/batch/__test__/EasyAgentBatchSurface.test.tsx` — **modifié**

- Adapter query exacte sans changement UX.

121) `AIcode/src/frontend/src/app/prompt/hydration/isomorphism-invariant-rule-between-streaming-and-rehydration-of-chat-sessions.programming-doc-readme.md` — **modifié**

- Ajouter troisième chemin fermeture/réouverture, continuation, thinking, retry, erreurs canoniques, vrai terminal et barrière post-forwarding.
- Documenter que l'erreur source du premier header est transformée en chunk de données exact par le handler extension avant le frontend.

122) `AIcode/src/frontend/src/app/prompt/thinking-bubble-and-hourglass-streaming-rules-invariants.programming-doc-readme.md` — **modifié**

- Documenter placeholder vide, bulle agrégée, séparateur et reset streamInstanceId.

Récapitulatif exact des chemins de fichiers

Créés

- `AIcode/src/sharedlib/src/types/liveSessionTypes/LiveSessionReplay.type.ts`
- `AIcode/src/extension/src/aiTools/orch/liveSession/source/LiveSessionSourceLifecycle.service.ts`
- `AIcode/src/extension/src/aiTools/orch/liveSession/source/__test__/LiveSessionSourceLifecycle.service.test.ts`
- `AIcode/src/extension/src/aiTools/orch/liveSession/source/NormalPromptLiveSessionSource.service.ts`
- `AIcode/src/extension/src/aiTools/orch/liveSession/source/__test__/NormalPromptLiveSessionSource.service.test.ts`
- `AIcode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/abortSource.endpoint.ts`
- `AIcode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/abortSource.endpoint.test.ts`
- `AIcode/src/extension/src/extension/panels/prompt/liveSession/PromptLiveSessionPanelQuery.util.ts`
- `AIcode/src/extension/src/extension/panels/prompt/liveSession/__test__/PromptLiveSessionPanelQuery.util.test.ts`
- `AIcode/src/extension/src/extension/panels/prompt/liveSession/PromptLiveSessionPanelCoordinator.service.ts`
- `AIcode/src/extension/src/extension/panels/prompt/liveSession/__test__/PromptLiveSessionPanelCoordinator.service.test.ts`
- `AIcode/src/extension/src/extension/panels/prompt/liveSession/PromptLiveSessionDetachedStop.service.ts`
- `AIcode/src/extension/src/extension/panels/prompt/liveSession/__test__/PromptLiveSessionDetachedStop.service.test.ts`
- `AIcode/src/extension/src/extension/panels/sidebar/__test__/SidebarStore.handlers.manager.test.ts`
- `AIcode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/__test__/delete.endpoint.liveSession.test.ts`
- `AIcode/src/frontend/src/app/prompt/liveSession/adapters/NormalPromptLiveSession.adapter.ts`
- `AIcode/src/frontend/src/app/prompt/liveSession/adapters/__test__/NormalPromptLiveSession.adapter.test.ts`
- `AIcode/src/frontend/src/app/prompt/liveSession/adapters/PromptLiveSessionBootstrap.adapter.ts`
- `AIcode/src/frontend/src/app/prompt/liveSession/adapters/__test__/PromptLiveSessionBootstrap.adapter.test.ts`

- AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.normalPrompt.test.tsx
- AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.normalPromptIsomorphism.test.tsx
- AICode/src/frontend/src/messageBus/__test__/useAiStream.hook.promptSourceIdentity.test.ts
- AICode/src/frontend/src/app/prompt/Prompt/preparePromptContinuationStreaming.util.ts
- AICode/src/frontend/src/app/prompt/Prompt/__test__/preparePromptContinuationStreaming.util.test.ts
- AICode/src/frontend/src/app/prompt/Prompt/__test__/usePromptThinkingLifecycle.continuation.test.tsx
- AICode/src/frontend/src/app/prompt/Prompt/__test__/usePromptRunCommands.continueResponse.test.tsx

Modifiés

- AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionRun.type.ts
- AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionPanel.type.ts
- AICode/src/sharedlib/src/types/liveSessionTypes/LiveSessionOpenResult.type.ts
- AICode/src/sharedlib/src/types/liveSessionTypes/__test__/LiveSessionContracts.test.ts
- AICode/src/sharedlib/src/types/aiTypes/aiStopSuffix.util.ts
- AICode/src/sharedlib/src/types/aiTypes/__test__/aiStopSuffix.util.test.ts
- AICode/src/sharedlib/src/types/busTypes/ui/SidebarStore.ts
- AICode/src/sharedlib/src/types/busTypes/ui/SidebarUiMessages.ts
- AICode/src/sharedlib/src/types/busTypes/global/MessageRegistry.ts
- AICode/src/extension/src/aiTools/orch/liveSession/liveSession.programming-doc-readme.md
- AICode/src/extension/src/aiTools/orch/liveSession/registry/LiveSessionRecord.type.ts
- AICode/src/extension/src/aiTools/orch/liveSession/registry/LiveSessionRegistry.service.ts
- AICode/src/extension/src/aiTools/orch/liveSession/registry/__test__/LiveSessionRegistry.service.test.ts
- AICode/src/extension/src/aiTools/orch/liveSession/core/LiveSessionService.ts
- AICode/src/extension/src/aiTools/orch/liveSession/core/__test__/LiveSessionService.test.ts
- AICode/src/extension/src/aiTools/orch/liveSession/open/LiveSessionOpen.service.ts
- AICode/src/extension/src/aiTools/orch/liveSession/open/__test__/LiveSessionOpen.service.test.ts
- AICode/src/extension/src/aiTools/orch/liveSession/replay/__test__/LiveSessionReplay.service.test.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/liveSession/EasyAgentsBatchLiveSessionAdapter.service.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/liveSession/__test__/EasyAgentsBatchLiveSessionAdapter.service.test.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/EasyAgentsBatchRunExecution.service.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunExecution.live-activity-and-gating.service.test.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/EasyAgentsBatchRunChildExecution.service.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunChildExecution.service.test.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/EasyAgentsBatchRunManager.service.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunExecution.testHarness.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchRunLifecycle.service.test.ts
- AICode/src/extension/src/aiTools/easyAgents/batch/run/__test__/EasyAgentsBatchLastViewerStop.service.test.ts

- AICode/src/extension/src/aiTools/easyAgents/easyAgents.programming-doc-readme.md
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/open.endpoint.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/open.endpoint.test.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/stream.endpoint.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/stream.endpoint.test.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/abort.endpoint.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/liveSession/__test__/abort.endpoint.test.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/query/ask/stream.endpoint.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/query/ask/__test__/stream.endpoint.test.ts
- AICode/src/extension/src/messageBus/handlers/ia/liveSessionStreamHandlers.ts
- AICode/src/extension/src/messageBus/handlers/ia/__test__/liveSessionStreamHandlers.test.ts
- AICode/src/extension/src/messageBus/handlers/ia/iaStreamHandlers.ts
- AICode/src/extension/src/messageBus/handlers/ia/__test__/iaStreamHandlers.queryDoneSound.test.ts
- AICode/src/extension/src/messageBus/handlers/ia/__test__/iaStreamHandlers.runIdPropagation.test.ts
- AICode/src/extension/src/extension/panels/prompt/PromptQueryDoneSound.manager.ts
- AICode/src/extension/src/extension/panels/prompt/__test__/PromptQueryDoneSound.manager.test.ts
- AICode/src/extension/src/messageBus/handlers/ia/SidebarQueryRunningBridge.ts
- AICode/src/extension/src/messageBus/handlers/ia/__test__/SidebarQueryRunningBridge.race.test.ts
- AICode/src/extension/src/messageBus/handlers/ia/__test__/SidebarQueryRunningBridge.usage.test.ts
- AICode/src/extension/src/extension/panels/prompt/PromptThreadCleanup.manager.ts
- AICode/src/extension/src/extension/panels/prompt/__test__/PromptThreadCleanup.manager.attachmentsCleanup.test.ts
- AICode/src/extension/src/extension/panels/core/openPanel.util.ts
- AICode/src/extension/src/extension/panels/core/__test__/openPanel.util.test.ts
- AICode/src/extension/src/messageBus/handlers/ui/uiGenericHandlers.ts
- AICode/src/extension/src/messageBus/handlers/ui/__test__/uiGenericHandlers.openExternalUrl.test.ts
- AICode/src/extension/src/extension/commands/NewChat.command.ts
- AICode/src/extension/src/extension/commands/__test__/NewChat.command.test.ts
- AICode/src/extension/src/extension/uriHandlers/registerDeepLinkUriHandler.ts
- AICode/src/extension/src/extension/uriHandlers/__test__/registerDeepLinkUriHandler.scope.test.ts
- AICode/src/extension/src/extension/panels/sidebar/SidebarStore.queryWindow.listeners.manager.ts
- AICode/src/extension/src/extension/panels/sidebar/SidebarStore.queryWindow.mutations.manager.ts
- AICode/src/extension/src/extension/panels/sidebar/SidebarStore.core.manager.ts
- AICode/src/extension/src/extension/panels/sidebar/SidebarStore.rehydrate.manager.ts
- AICode/src/extension/src/extension/panels/sidebar/__test__/SidebarStore.rehydrate.manager.test.ts
- AICode/src/extension/src/extension/panels/sidebar/__test__/SidebarStore.manager.queryRunningUpsert.test.ts
- AICode/src/extension/src/extension/panels/sidebar/SidebarStore.chatTitles.manager.ts
- AICode/src/extension/src/extension/panels/sidebar/SidebarStore.handlers.manager.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/delete.endpoint.ts

- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/__test__/list.endpoint.windowsLock.eperm.test.ts
- AICode/src/extension/src/slimbk-local/endpoints/ai/orch/chat/__test__/diskAliases.lastWins.endpoints.test.ts
- AICode/src/extension/src/aiTools/orch/context/persistence/__test__/ChatPersistenceWriter.windowsFallback.test.ts
- AICode/src/extension/src/aiTools/orch/context/memory/core/mem/__test__/OrchMemory.naive_title.user_prompt_specif_multiline.e2e.repro.test.ts
- AICode/src/frontend/src/app/sidebar/sidebar-activity/SidebarActivityQuery.tsx
- AICode/src/frontend/src/app/sidebar/sidebar-activity/__test__/SidebarActivityQuery.test.tsx
- AICode/src/frontend/src/app/prompt/liveSession/core/PromptLiveSession.types.ts
- AICode/src/frontend/src/app/prompt/liveSession/core/usePromptLiveSession.hook.ts
- AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.hydratation.test.tsx
- AICode/src/frontend/src/app/prompt/liveSession/core/__test__/usePromptLiveSession.handoff.test.tsx
- AICode/src/frontend/src/app/prompt/liveSession/adapters/EasyAgentsBatchLiveSession.adapter.ts
- AICode/src/frontend/src/app/prompt/liveSession/adapters/__test__/EasyAgentsBatchLiveSession.adapter.test.ts
- AICode/src/frontend/src/messageBus/AiStreamStartArgs.type.ts
- AICode/src/frontend/src/messageBus/useAiStream.hook.ts
- AICode/src/frontend/src/messageBus/__test__/useAiStream.hook.liveSessionObserver.test.ts
- AICode/src/frontend/src/app/prompt/Prompt/usePromptRunController.hook.ts
- AICode/src/frontend/src/app/prompt/Prompt/__test__/usePromptRunController.liveSessionObserver.test.tsx
- AICode/src/frontend/src/app/prompt/Prompt/preparePromptStreamingTurn.util.ts
- AICode/src/frontend/src/app/prompt/Prompt/__test__/preparePromptStreamingTurn.util.test.ts
- AICode/src/frontend/src/app/prompt/Prompt/usePromptThinkingLifecycle.hook.ts
- AICode/src/frontend/src/app/prompt/Prompt/usePromptToolUse.hook.ts
- AICode/src/frontend/src/app/prompt/Prompt/usePromptRunController.finalization.hook.ts
- AICode/src/frontend/src/app/prompt/Prompt/usePromptRunCommands.hook.ts
- AICode/src/frontend/src/app/prompt/Prompt/Prompt.tsx
- AICode/src/frontend/src/app/prompt/Prompt/__test__/Prompt.liveSessionBootstrap.test.tsx
- AICode/src/frontend/src/app/prompt/PromptInput/__test__/PromptInput.batchLiveViewerMode.test.tsx
- AICode/src/frontend/src/app/easy-agents/easy-agents-batch/core/__test__/EasyAgentsBatch.liveWatch.test.tsx
- AICode/src/frontend/src/app/easy-agents/easy-agents-item/core/batch/__test__/EasyAgentBatchSurface.test.tsx
- AICode/src/frontend/src/app/prompt/hydratation/isomorphism-invariant-rule-between-streaming-and-rehydratation-of-chat-sessions.programming-doc-readme.md
- AICode/src/frontend/src/app/prompt/thinking-bubble-and-hourglass-streaming-rules-invariants.programming-doc-readme.md

Déplacés

Aucun.

Supprimés

Aucun.

Facteur de risque de régression

1. **Risque calculé : 92 %.** Criticité 5, couplage 5, blast radius 5, incertitude 4, faible confiance 3. Base = $10 \times 5 + 3 \times 5 + 2 \times 5 + 2 \times 4 + 3 \times 3 = 92$; risque = $\text{round}(92 \times 5/5) = 92 \%$.

2. **Zones complexes** : sources successives, retry de même source, réservations, callbacks stale, lifecycle lazy, pipeline non attendue, traitement spécial du premier header source-error, finalisation post-forwarding, EOF, continuation, thinking, panneau/cleanup, Activity, son, handoff, draft/attachments, Stop pré-enregistrement, deletion et retraites.
3. **Justification** : la tranche modifie le chemin normal de toutes les requêtes Prompt et plusieurs coordinations asynchrones. Le traitement du premier header d'erreur est particulièrement critique : si le handler le laisse remonter comme erreur transport, le frontend ne peut pas satisfaire la règle « vrai terminal data + complétion normale », le son ne voit pas le terminal exact et le handoff direct devient impossible pour cette classe d'erreurs. La reconstruction extension dans la FIFO unique, les identités exactes, réservations, époques et tests d'isomorphisme sont donc obligatoires.

Résumé

- Étendre `liveSession` aux sources Prompt normales en une implémentation complète.
- Ajouter `sourceId` de bout en bout et `streamInstanceId` frontend.
- Réserver puis enregistrer les sources, `append-before-forward` et partager le lifecycle.
- Drainer une FIFO extension unique avant finalisation.
- Décoder le premier header d'erreur source dans le handler extension, l'enfiler comme chunk terminal exact, l'émettre comme donnée et conserver une complétion transport normale.
- Finaliser atomiquement `sourceSettled` + `handoffReady` seulement après drainage.
- Capturer les messages de continuation sans figer l'enveloppe.
- Réutiliser assistant/thinking sans duplication.
- Ne plus tuer une source normale à la fermeture et différer les cleanups.
- Sérialiser panneau/dispose/cleanup par locator.
- Modéliser Activité avec ownership et panneau open/closed.
- Ajouter uniquement le Stop détaché source/closed.
- Corréler son et contrôles au `runRef` exact.
- Utiliser le registre comme autorité deletion/Retry.
- Préserver EasyAgents, queues, locks et sanitizers.
- Tester stream continu, replay, hydratation, erreurs source encodées et transport réel.

Nom du commit

`feat(live-session): keep prompt runs alive across panel close and reopen`

`SPECIF_FINALISER_LIVESESSION_POUR_LES_SESSIONS_PROMPT_NORMALES_20260712_014742`

Copy

Refine

Code

Verify

Sévérité	Description du changement	Pourquoi
■ majeure	Déplacement du décodage du premier <code>meta.error</code> encodé depuis le frontend vers le handler extension normal. Le chunk reconstruit rejoint désormais la même pipeline FIFO que les callbacks <code>onChunk</code> .	Le client Slimbk généré transmet un header non-200 à <code>onError</code> et n'attend pas les callbacks asynchrones. Si cette erreur remontait jusqu'au frontend comme erreur de transport, le vrai terminal ne serait pas reçu via <code>onData</code> , le son ne l'observerait pas et le handoff direct normal serait impossible. Le handoff normal exige simultanément un vrai terminal de données et un outcome transport <code>completed</code> . Cette distinction ferme une incohérence fonctionnelle de la version précédente.
■ majeure	Distinction explicite entre une erreur source encodée et traitée et une véritable erreur de transport . Le callback extension ne throw pas pour le protocole privé valide, permettant une complétion normale du stream.	Cela évite un double
■ mineure	Alignement des responsabilités de <code>useAiStream</code> : les	

Sévérité	Description du changement	Pourquoi
	erreurs source encodées arrivent comme <code>meta.error</code> via <code>onData</code> ; <code>onError</code> reste réservé aux pannes de transport non traitées et produit le <code>SYSTEM_ERROR</code> canonique.	décodage, une double injection terminale et une divergence entre streaming continu, replay et hydratation. Les tests doivent verrouiller l'ordre <code>chunks ordinaires</code> → <code>terminal décodé</code> → <code>finalisation registre</code> →
■ mineure	Renforcement des tests du handler, du son, de l'ordre FIFO, du handoff et de l'isomorphisme pour couvrir le premier header d'erreur source.	→ <code>son/Activité/cleanup</code> et garantir une émission <code>frontend data</code> → <code>done</code> , sans événement transport <code>error</code> .