FlowDraftStitcherService

September 4, 2026

FlowDraftStitcherService

Kind: Service

Source: `atloria-monorepo/apps/api/src/capture/flow-draft-stitcher.service.ts`

FlowDraftStitcherService (WS-A, api side).

Background poll loop: watches a flow-replay job on the screenshot-worker and, as each step's screenshot becomes available, replaces that step's `[SCREENSHOT: flow-step-N]` placeholder in the draft Document's markdown with a real annotated image `![Step N](<blobUrl>#atloria-annotations=…)`. When the job finishes it clears `metadata.screenshotsPending` and strips any placeholders for steps that never produced a screenshot (they degrade to text via the reader's remarkStripScreenshots). Runs HTTP-only — no browser on the api pod.

`FlowDraftStitcherService` runs on the API side to poll screenshot-worker flow-replay jobs and stitch completed step screenshots into a draft document. As screenshots become available, it replaces `[SCREENSHOT: flow-step-N]` markdown placeholders with annotated image URLs; when the replay completes, it clears `metadata.screenshotsPending` and removes unresolved placeholders.

Methods

Method	Signature	Returns	Description
`start`	`start(params: StartStitchParams)`	`void`	Fire-and-forget: kick off the poll loop (never rejects into the caller).

Dependencies

- `PrismaService`

Where it refuses work

- `FlowDraftStitcherService` stops the work with an early return when `!doc`, in 2 places.
- `FlowDraftStitcherService` stops the work with an early return when `!doc.content.includes(token)`.

When something fails

- `FlowDraftStitcherService` handles failure in 1 place: it turns it into a return value in all 1.

Diagram

```
sequenceDiagram
  participant API as FlowDraftStitcherService
  participant Worker as screenshot-worker
  participant Draft as Draft Document Store

  loop While flow-replay job is pending
    API->>Worker: Poll flow-replay job status
    Worker-->>API: Completed steps and screenshot blob URLs

    loop For each newly completed step
      API->>Draft: Load draft markdown
      API->>API: Replace [SCREENSHOT: flow-step-N]
      API->>Draft: Save annotated image markdown
    end
  end

  Worker-->>API: Job finished
  API->>Draft: Clear metadata.screenshotsPending
  API->>Draft: Remove unresolved screenshot placeholders
```

Usage

```
import { Injectable } from '@nestjs/common';
import { FlowDraftStitcherService } from './flow-draft-stitcher.service';

@Injectable()
export class FlowReplayCoordinator {
  constructor(
    private readonly flowDraftStitcher: FlowDraftStitcherService,
  ) {}

  async startStitching(params: {
    draftId: string;
```

```
    replayJobId: string;
  }): Promise<void> {
    await this.flowDraftStitcher.stitchFlowDraft({
      draftId: params.draftId,
      replayJobId: params.replayJobId,
    });
  }
}
```

AI Coding Instructions

- Keep this service HTTP-only: communicate with screenshot-worker through its API and do not introduce browser, Playwright, or local screenshot dependencies.
- Preserve the exact placeholder format, `[SCREENSHOT: flow-step-N]`, so replacements remain deterministic and idempotent.
- Only replace placeholders for newly available screenshots; avoid rewriting already stitched image markdown on subsequent poll iterations.
- When a replay job reaches a terminal state, clear `metadata.screenshotsPending` and remove unresolved placeholders so the document reader can degrade remaining content safely.
- Ensure annotated image URLs use the expected `#atloria-annotations=…` fragment format consumed by the document renderer.

Relationships

- DEPENDS_ON → `PrismaService`

Referenced By

- `CaptureModule` (MODULE_PROVIDES)
- `CaptureService` (DEPENDS_ON)