ScreenshotStalenessService

September 4, 2026

ScreenshotStalenessService

Kind: Service

Source: `atloria-monorepo/apps/api/src/doc-version/screenshot-staleness.service.ts`

`ScreenshotStalenessService` evaluates how current a document-version screenshot is and exposes its staleness classification, age in days, and a human-readable explanation. It is used by backend document-version workflows to consistently determine whether screenshots should be considered fresh, aging, or outdated.

Methods

Method	Signature	Returns	Description
`getStalenessLevel`	`getStalenessLevel(capturedAt: string | Date, isCurrentVersion: boolean)`	`number`	Get human-readable reason for the staleness level.
`getDaysAge`	`getDaysAge(date: string | Date)`	`string`	
`getStalenessReason`	`getStalenessReason(level: StalenessLevel, age: number)`		

Where it refuses work

- `ScreenshotStalenessService` stops the work with an early return when `!capturedAt`.
- `ScreenshotStalenessService` stops the work with an early return when `isCurrentVersion`.
- `ScreenshotStalenessService` stops the work with an early return when `age <= 30`.
- `ScreenshotStalenessService` stops the work with an early return when `age <= 90`.

Diagram

```
sequenceDiagram
  participant Caller as Document Version Workflow
  participant Service as ScreenshotStalenessService

  Caller->>Service: getDaysAge()
  Service-->>Caller: Screenshot age (days)

  Caller->>Service: getStalenessLevel()
  Service-->>Caller: StalenessLevel

  Caller->>Service: getStalenessReason()
  Service-->>Caller: Human-readable reason

  Caller->>Caller: Display status or trigger refresh workflow
```

Usage

```
import { Injectable } from '@nestjs/common';
import { ScreenshotStalenessService } from './screenshot-staleness.service';

@Injectable()
export class DocumentVersionStatusService {
  constructor(
    private readonly screenshotStalenessService: ScreenshotStalenessService,
  ) {}

  getScreenshotStatus() {
    return {
      level: this.screenshotStalenessService.getStalenessLevel(),
      daysAge: this.screenshotStalenessService.getDaysAge(),
      reason: this.screenshotStalenessService.getStalenessReason(),
    };
  }
}
```

AI Coding Instructions

- Use `getStalenessLevel()` as the canonical value for conditional backend behavior, such as flagging or refreshing outdated screenshots.
- Use `getDaysAge()` for numeric display, sorting, or threshold-related UI/API fields rather than recalculating age elsewhere.
- Return `getStalenessReason()` alongside the level when exposing staleness data to clients so status decisions remain explainable.

- Keep staleness thresholds and classification logic centralized in this service; do not duplicate date-comparison logic in controllers or consumers.
- Ensure consumers handle all available `StalenessLevel` values rather than assuming only fresh and stale states.

Referenced By

- `DocVersionModule` (MODULE_PROVIDES)
- `DocVersionModule` (MODULE_EXPORTS)