ContentTasksService

September 4, 2026

ContentTasksService

Kind: Service

Source: `atloria-monorepo/apps/api/src/content-tasks/content-tasks.service.ts`

Analytics action queue — CRUD over ContentTask.

Org tenancy is enforced by the controller's ResourceOrgGuard(kind: 'project'); every query here additionally filters by projectId (defense in depth). The service NEVER recomputes insight data — titles/detail arrive from the analytics page, which already loaded them. Identity is (projectId, sourceRef): creating a task for an insight that is already tracked returns the existing row (idempotent), which is what lets the page dedupe insights to a "tracked" state by sourceRef.

`ContentTasksService` manages analytics action-queue tasks backed by `ContentTask` records. It provides CRUD operations scoped to a `projectId`, treats `(projectId, sourceRef)` as the task identity for idempotent creation, and preserves analytics-provided titles and details rather than recomputing insight data.

Methods

Method	Signature	Returns	Description
`list`	`list(projectId: string, filters: ListContentTasksFilters)`	`unknown`	Open first, then most-recently-updated.
`create`	`create(projectId: string, organizationId: string, createdById: string, dto: CreateContentTaskDto)`	`unknown`	Create a task from an insight, or return the existing task if this insight is already tracked (dedupe by the (projectId, sourceRef) unique).
`update`	`update(projectId: string, taskId: string, dto: UpdateContentTaskDto)`	`unknown`	Change status / assignee / title.

Method	Signature	Returns	Description
`remove`	`remove(projectId: string, taskId: string)`	`unknown`	
`members`	`members(organizationId: string)`	`unknown`	Active org members for the assignee picker (scoped to the caller's org).

Dependencies

- `PrismaService`

Where it refuses work

- `ContentTasksService` stops the work with `NotFoundException` when `!task` — “Content task not found”, in 2 places.
- `ContentTasksService` stops the work with `BadRequestException` when `!member` — “Assignee is not a member of this organization”.
- `ContentTasksService` stops the work with an early return when `existing`.

Diagram

```
sequenceDiagram
  participant Client as Analytics Page
  participant Controller as ContentTasksController
  participant Guard as ResourceOrgGuard
  participant Service as ContentTasksService
  participant DB as Database

  Client->>Controller: POST /content-tasks<br/>{ projectId, sourceRef, title, detail }
  Controller->>Guard: Validate org access for project
  Guard-->>Controller: Authorized
  Controller->>Service: create(projectId, dto)

  Service->>DB: Find task by projectId + sourceRef
  alt Existing task found
    DB-->>Service: Existing ContentTask
    Service-->>Controller: Return existing task
  else No matching task
    Service->>DB: Create ContentTask
    DB-->>Service: Created ContentTask
    Service-->>Controller: Return created task
  end
```

```
  Controller-->>Client: ContentTask response
```

Usage

```
import { ContentTasksService } from './content-tasks.service';

@Injectable()
export class AnalyticsActionsService {
  constructor(
    private readonly contentTasksService: ContentTasksService,
  ) {}

  async trackInsight(projectId: string) {
    // These values should come from the analytics page/data already loaded.
    const task = await this.contentTasksService.create(projectId, {
      sourceRef: 'insight:high-bounce-rate:landing-page',
      title: 'Review landing page bounce rate',
      detail: 'Bounce rate increased by 18% compared with the previous period.',
    });

    // Calling create again with the same projectId and sourceRef
    // returns the existing task rather than creating a duplicate.
    return task;
  }

  async listTrackedInsights(projectId: string) {
    return this.contentTasksService.list(projectId);
  }
}
```

AI Coding Instructions

- Always pass and filter by `projectId`; controller-level `ResourceOrgGuard(kind: 'project')` enforces tenancy, while service-level scoping is required as defense in depth.
- Preserve `(projectId, sourceRef)` as the idempotency key when creating tasks; return the existing row when a matching task is already tracked.
- Do not recompute analytics insights, titles, or details in this service—persist the values supplied by the analytics page.
- Ensure `update`, `remove`, and `members` queries also constrain records by `projectId` before reading or mutating data.
- When adding task fields or query behavior, keep the analytics page integration compatible with source-reference-based “tracked” deduplication.

Relationships

- DEPENDS_ON → `PrismaService`

Referenced By

- `ContentTasksController` (DEPENDS_ON)
- `ContentTasksModule` (MODULE_PROVIDES)
- `ContentTasksModule` (MODULE_EXPORTS)