PublicChangelogController

September 4, 2026

PublicChangelogController

Kind: Controller

Source: `atloria-monorepo/apps/api/src/changelog/public-changelog.controller.ts`

Published-site plane for the B5 changelog (`/api/v1/public/p/:slugWithId/changelog...`).

Same guard stack as PublicProjectController: OptionalJwtAuthGuard populates req.user (org-member bypass), ReaderAuthGuard decodes the reader token, and DocsAccessGuard enforces the A3 gate on every :slugWithId surface — a gated project's changelog, RSS and JSON feeds 401 exactly like llms.txt does.

The ONLY exceptions are the emailed token links (confirm / unsubscribe): they must work from an email client with no reader session, the token is the capability, so those two routes carry

`PublicChangelogController` serves published changelog content for public project URLs under `/api/v1/public/p/:slugWithId/changelog` . It applies the same optional JWT, reader-token, and documentation access checks used by public project endpoints, ensuring gated projects protect changelog pages and RSS/JSON feeds. Email confirmation and unsubscribe token routes are capability-based exceptions that work without an active reader session.

Diagram

```
graph LR
  Client[Browser / RSS Reader / API Client] --> Controller[PublicChangelogController]
  Controller --> OptionalJwt[OptionalJwtAuthGuard]
  OptionalJwt --> ReaderAuth[ReaderAuthGuard]
  ReaderAuth --> DocsAccess[DocsAccessGuard]
  DocsAccess --> Changelog[Published changelog content]
  Changelog --> Page[Changelog page]
  Changelog --> RSS[RSS feed]
  Changelog --> JSON[JSON feed]
```

```
  EmailClient[Email client] --> TokenRoutes[Confirm / unsubscribe token routes]
  TokenRoutes --> Subscription[Subscription action]
```

Usage

```
const projectSlugWithId = 'my-project-abc123';

// Fetch published changelog entries.
const response = await fetch(
  `https://api.example.com/api/v1/public/p/${projectSlugWithId}/changelog`,
  {
    headers: {
      // Include a reader token when accessing a gated project.
      Authorization: `Bearer ${readerToken}`,
    },
  },
);

if (response.status === 401) {
  throw new Error('Reader access is required for this project changelog.');
}

const changelog = await response.json();

// RSS and JSON representations use the same public access policy.
const rssResponse = await fetch(
  `https://api.example.com/api/v1/public/p/${projectSlugWithId}/changelog/rss`,
);
```

AI Coding Instructions

- Keep every `:slugWithId` changelog, RSS, and JSON route behind the same `OptionalJwtAuthGuard`, `ReaderAuthGuard`, and `DocsAccessGuard` stack as public project routes.
- Preserve the organization-member bypass behavior provided by `OptionalJwtAuthGuard`; do not duplicate membership checks inside route handlers.
- Treat gated-project feeds exactly like other protected published surfaces: unauthenticated or unauthorized access must return `401`.
- Do not apply reader-session requirements to email confirmation or unsubscribe token endpoints; their signed token is the access capability.
- When adding a new changelog representation or public sub-route, verify it is covered by the project access guard unless it is an explicitly token-authorized email flow.

Relationships

- MODULE_DECLARES → `getChangelog`
- MODULE_DECLARES → `getRss`
- MODULE_DECLARES → `getJsonFeed`
- MODULE_DECLARES → `subscribe`
- MODULE_DECLARES → `confirm`
- MODULE_DECLARES → `unsubscribe`
- DEPENDS_ON → `ChangelogPublicService`
- DEPENDS_ON → `ChangelogSubscriptionsService`

Referenced By

- `ChangelogModule` (MODULE_DECLARES)