OutboxPollerService

September 4, 2026

OutboxPollerService

Kind: Service

Source: `atloria-monorepo/apps/api/src/docs-repo/outbox-poller.service.ts`

OutboxPollerService — the PRODUCER that actually runs the commit loop.

The outbox rows are only useful if something drains them; nothing else does. Every minute
(on the worker only) this reads the pending outbox grouped by project and enqueues ONE
coalesced commit job per project with pending work. The CommitterWorker then drains + commits.

GATED on DOCS_REPO_WORKER==='true' (mirrors technical-docs' worker gate) so the many api
replicas don't all poll and double-enqueue — only the dedicated 1-replica worker deployment
produces. ScheduleModule.forRoot() is registered app-wide, so

`OutboxPollerService` is the producer side of the docs repository outbox workflow. Running only when `DOCS_REPO_WORKER === 'true'`, it periodically finds pending outbox records, groups them by project, and enqueues one coalesced commit job per project. The `CommitterWorker` consumes those jobs to drain and commit the pending documentation changes.

Methods

Method	Signature	Returns	Description
`poll`	`poll()`	`Promise<void>`	Cron entrypoint — gated so only the worker deployment produces commit jobs.

Method	Signature	Returns	Description
`drainAndEnqueue`	`drainAndEnqueue()`	`Promise<number>`	Read the pending outbox grouped by project and enqueue a coalesced commit for each project that has work.

Dependencies

- `ConfigService`
- `OutboxService`
- `DocsRepoQueue`

Where it refuses work

- `OutboxPollerService` stops the work with an early return when `this.config.get<string>('DOCS_REPO_WORKER') !== 'true'`.

When something fails

- `OutboxPollerService` handles failure in 1 place: it logs it and continues in all 1.

Diagram

```
sequenceDiagram
    participant Scheduler as NestJS Scheduler
    participant Poller as OutboxPollerService
    participant Outbox as Docs Outbox
    participant Queue as Commit Job Queue
    participant Worker as CommitterWorker
    participant Repo as Documentation Repository

    Scheduler->>Poller: poll() every minute
    alt DOCS_REPO_WORKER === "true"
        Poller->>Outbox: Read pending rows grouped by project
        loop Each project with pending work
            Poller->>Queue: Enqueue one coalesced commit job
        end
        Queue->>Worker: Deliver commit job
        Worker->>Outbox: Drain project outbox rows
        Worker->>Repo: Commit documentation changes
    else Worker mode disabled
        Poller-->>Scheduler: Skip polling
    end
```

Usage

```
import { Injectable } from '@nestjs/common';
import { OutboxPollerService } from './outbox-poller.service';

@Injectable()
export class DocsRepoMaintenanceService {
  constructor(private readonly outboxPoller: OutboxPollerService) {}

  async processPendingDocsChanges(): Promise<void> {
    // Normally invoked by the service's scheduled poll.
    // Useful for an explicit worker-triggered run or integration test.
    const projectsEnqueued = await this.outboxPoller.drainAndEnqueue();

    console.log(`Enqueued commit work for ${projectsEnqueued} project(s).`);
  }
}
```

AI Coding Instructions

- Keep polling and job production gated behind `DOCS_REPO_WORKER === 'true'`; API replicas must not independently enqueue duplicate commit work.
- Preserve project-level coalescing: enqueue at most one commit job per project even when multiple outbox rows are pending.
- Treat this service as a producer only; the `CommitterWorker` is responsible for draining outbox entries and creating repository commits.
- When adding new outbox-producing features, ensure their rows can be discovered by the poller’s pending-work query.
- Prefer idempotent queueing and commit handling because scheduled polling may run again before previously enqueued work is completed.

Relationships

- DEPENDS_ON → `configservice`
- DEPENDS_ON → `OutboxService`
- DEPENDS_ON → `DocsRepoQueue`

Referenced By

- `DocsRepoModule` (MODULE_PROVIDES)