WorkflowDiscoveryService

September 4, 2026

WorkflowDiscoveryService

Kind: Service

Source: `atloria-monorepo/apps/api/src/workflow/workflow-discovery.service.ts`

`WorkflowDiscoveryService` is a NestJS backend service responsible for discovering available workflows and returning a `WorkflowDiscoveryResult`. It also provides a token/cost estimate for discovery operations and exposes cache invalidation so callers can refresh previously discovered workflow data when the underlying workflow configuration changes.

Methods

Method	Signature	Returns	Description
`discoverWorkflows`	`discoverWorkflows(projectId: string, workflowSignals: WorkflowSignals, options: {		

```
  model?: string;
  force?: boolean;
  businessContext?: BusinessContext;
})` | `Promise<WorkflowDiscoveryResult>` | Discover workflows from code signals using LLM |
```

| `estimateCost` | `estimateCost(workflowSignals: WorkflowSignals, model: string)` | `{ inputTokens: number; outputTokens: number; costUSD: number; }` | Estimate cost before running discovery |
| `clearCache` | `clearCache(projectId: string)` | `void` | Clear cache for a project |

Dependencies

- `ConfigService`
- `AzureClaudeProvider`

When something fails

- `WorkflowDiscoveryService` handles failure in 4 places: it lets it reach the caller in 3, and turns it into a return value in 1.

Diagram

```
sequenceDiagram
    participant Consumer as API Consumer
    participant Service as WorkflowDiscoveryService
    participant Cache as Discovery Cache
    participant Source as Workflow Sources

    Consumer->>Service: discoverWorkflows()
    Service->>Cache: Check cached discovery result

    alt Cache hit
        Cache-->>Service: Cached WorkflowDiscoveryResult
    else Cache miss
        Service->>Source: Discover workflows
        Source-->>Service: Workflow metadata
        Service->>Cache: Store discovery result
    end

    Service-->>Consumer: WorkflowDiscoveryResult

    Consumer->>Service: estimateCost()
    Service-->>Consumer: Token and USD estimate

    Consumer->>Service: clearCache()
    Service->>Cache: Remove cached discovery data
```

Usage

```
import { Injectable } from '@nestjs/common';
import { WorkflowDiscoveryService } from './workflow-discovery.service';

@Injectable()
export class WorkflowCatalogService {
  constructor(
    private readonly workflowDiscoveryService: WorkflowDiscoveryService,
  ) {}

  async listAvailableWorkflows() {
    const workflows = await this.workflowDiscoveryService.discoverWorkflows();
    const estimate = this.workflowDiscoveryService.estimateCost();

    return {
```

```
      workflows,
      discoveryEstimate: {
        inputTokens: estimate.inputTokens,
        outputTokens: estimate.outputTokens,
        costUSD: estimate.costUSD,
      },
    };
  }

  refreshWorkflowCatalog(): void {
    this.workflowDiscoveryService.clearCache();
  }
}
```

AI Coding Instructions

- Inject `WorkflowDiscoveryService` through NestJS dependency injection; do not instantiate it directly with `new`.
- Call `discoverWorkflows()` asynchronously and treat its result as the canonical workflow discovery payload.
- Use `estimateCost()` for display, logging, or budget checks; it returns an estimate and should not be treated as actual billed usage.
- Call `clearCache()` after workflow definitions, integrations, or discovery inputs change to prevent stale results.
- Avoid clearing the cache on every request, since doing so bypasses the service's caching benefits and may increase discovery cost.

Relationships

- DEPENDS_ON → `configservice`
- DEPENDS_ON → `AzureClaudeProvider`

Referenced By

- `GraphModule` (MODULE_PROVIDES)
- `GraphModule` (MODULE_EXPORTS)
- `KgSyncController` (DEPENDS_ON)
- `SyncController` (DEPENDS_ON)
- `WorkflowModule` (MODULE_PROVIDES)
- `WorkflowModule` (MODULE_EXPORTS)