AgentService

September 4, 2026

AgentService

Kind: Service

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

`AgentService` is a NestJS backend service responsible for creating, retrieving, and destroying managed agent sessions. It maintains the currently active session state and exposes utilities for checking how many sessions are active, allowing other application components to coordinate agent lifecycle operations.

Methods

Method	Signature	Returns	Description
`createSession`	`createSession(socketId: string, payload: AgentStartPayload)`	`Promise<ManagedSession>`	Create a new agent session for a socket connection.
`getSession`	`getSession(socketId: string)`	`ManagedSession	undefined`	
`destroySession`	`destroySession(socketId: string)`	`void`	Destroy a session and clean up resources.
`getActiveSessionCount`	`getActiveSessionCount()`	`number`	Get active session count (for monitoring).

Dependencies

- `ConfigService`

Where it refuses work

- `AgentService` stops the work with `Error` when `!endpoint || !apiKey` — “Missing Azure Responses API configuration”.
- `AgentService` stops the work with an early return when `!response.ok`.

When something fails

- `AgentService` handles failure in 4 places: it discards it silently in 2, logs it and continues in 1, and turns it into a return value in 1.

Diagram

```
sequenceDiagram
    participant Controller
    participant AgentService
    participant ManagedSession

    Controller->>AgentService: createSession()
    AgentService->>ManagedSession: initialize session
    ManagedSession-->>AgentService: session instance
    AgentService-->>Controller: Promise<ManagedSession>

    Controller->>AgentService: getSession()
    AgentService-->>Controller: ManagedSession | undefined

    Controller->>AgentService: getActiveSessionCount()
    AgentService-->>Controller: number

    Controller->>AgentService: destroySession()
    AgentService->>ManagedSession: cleanup / terminate
    AgentService-->>Controller: void
```

Usage

```
import { Injectable } from '@nestjs/common';
import { AgentService } from './agent.service';

@Injectable()
export class AgentController {
  constructor(private readonly agentService: AgentService) {}

  async startAgentSession() {
    const existingSession = this.agentService.getSession();

    if (existingSession) {
```

```
      return {
        session: existingSession,
        activeSessions: this.agentService.getActiveSessionCount(),
      };
    }

    const session = await this.agentService.createSession();

    return {
      session,
      activeSessions: this.agentService.getActiveSessionCount(),
    };
  }

  stopAgentSession() {
    this.agentService.destroySession();

    return {
      activeSessions: this.agentService.getActiveSessionCount(),
    };
  }
}
```

AI Coding Instructions

- Inject `AgentService` through NestJS dependency injection instead of constructing it directly.
- Call `getSession()` before creating a new session when the caller should reuse an existing managed session.
- Await `createSession()` because session initialization is asynchronous and returns `Promise<ManagedSession>`.
- Handle the `undefined` result from `getSession()` before accessing session properties or methods.
- Call `destroySession()` during explicit shutdown, disconnect, or cleanup flows to avoid retaining active session resources.

Relationships

- DEPENDS_ON → `configservice`

Referenced By

- `AgentController` (DEPENDS_ON)
- `AgentModule` (MODULE_PROVIDES)

- `AgentModule` (MODULE_EXPORTS)