Actividad Integradora NestJS: API de Tareas

TASK SCHEMA: {
id: string AUTOGENERADO
name: string OBLIGATORIO
description: string OBLIGATORIO
scheduledTime: date-time OPCIONAL
priority: number OPCIONAL
}

- *CRUD tareas con persistencia en archivo*.
    - Create: debe respetar el schema y devolver la tarea almacenada con el id generado. Cada vez que se cree una tarea de forma exitosa, el sistema debe lanzar un evento interno llamado “TASK-CREATED-SUCCESSFULLY”, y el mismo debe ser escuchado desde otro servicio, que debe loguear el ID de la tarea en la consola, con el mensaje “NUEVA TAREA: <id>” (reemplazar id con el id de la tarea)
    - Read: debe devolver la lista completa de tareas. Debe esperar recibir un query param con el nombre “sort” cuyo valor puede ser “priority” o “scheduledTime”, debiendo devolver la lista de tareas ordenadas de forma descendente por el valor solicitado.
    - Update: debe permitir actualizar una tarea existente, a partir de su id, y, si existe la tarea, devolver el valor actualizado de la misma, y sino lanzar un error 400. No debe ser obligatorio recibir todos los campos, por ejemplo, podríamos solo intentar actualizar el “name”.
    - Delete: debe permitir borrar una tarea a partir de su id y lanzar un error 400 si la tarea no existe.
    - Además se debe implementar un endpoint que devuelva sólo una tarea a partir de su id, y que deberá lanzar error 404 en caso de no encontrarla.

- *Logging*: Registro de solicitudes entrantes en archivos .txt detallando método, endpoints y hora. Estos archivos deberán ser almacenados en una carpeta llamada “logs”, ubicada en el root del proyecto. Un archivo no podrá contener más de 10 logs. En caso de alcanzar los 10 logs, el almacenamiento deberá continuar en otro archivo, siempre dentro de la misma carpeta.

- *Health Check*: Implementar un endpoint abierto que devuelva la hora del sistema. Es el único endpoint que se exceptúa de las reglas de seguridad mencionadas a continuación.

- *Seguridad*:
    - Las rutas de escritura (crear, actualizar, eliminar) solo podrán ser consumidas por solicitudes que tengan el valor “admin” en el header “x-role”.
    - Las rutas de lectura sólo podrán ser consumidas por solicitudes que tengan el valor “admin” o el valor “user” en el header “x-role”.
    - En caso de recibir alguna solicitud que no contenga el header “x-role” o cuyo valor sea diferente de “admin” y “user” la API debe lanzar un error 401.

    - En caso de recibir el valor “user” en una ruta que espera el valor “admin”, lanzar error 403.

- *Control de flujo*: los endpoints de creación y actualización de tarea siempre deben incluir el campo “took” en la respuesta que envían, cuyo valor debe ser el tiempo en milisegundos que tomó el procesamiento del request. Este comportamiento debe realizarse fuera del servicio que devuelve los datos de la tarea creada/actualizada.

- *Cron*: La aplicación debe chequear cada 1 minuto si hay alguna tarea con una fecha dentro de las próximas 12 horas, y escribir en consola el siguiente mensaje “La tarea <**id**> vence en <**tiempo**>” donde id sea el **id** de la tarea y **tiempo** sea las horas y minutos que faltan para el vencimiento de la tarea (scheduledTime).