Escuela de Ingeniería Informática

Dpto. de Lógica y Programación

Tópicos Especiales de Programación

NRC 15997

Proyecto: UCAB Tasks

Actualmente en el entorno tecnológico que vivimos, las aplicaciones de gestión de notas de texto están ganando mucho terreno entre los usuarios. Desde aplicaciones básicas como el editor de notas que trae por defecto cualquier dispositivo móvil hasta aplicaciones más complejas con Notion, están siendo adoptadas de gran manera y sustituyendo de a poco el uso de notas tradicionales escritas en papel y lápiz.

En este contexto, se te ha asignado a un equipo que planea sacar una nueva aplicación para la gestión de notas de texto. Para esto, se requiere que se haga un pequeño prototipo para evaluar su factibilidad y ver posibles mejoras a incorporar durante el desarrollo de la aplicación.

Requisitos funcionales del prototipo

- No se requiere una interfaz gráfica, con que se haga una API REST donde se puedan obtener, crear, eliminar y modificar las notas (es decir, un CRUD) es suficiente.
- Las notas deben tener un título, el contenido que la conforma, la fecha en que se creó, la fecha de la última modificación y un identificador único.
- Para la obtención de las notas habrán dos (2) endpoints. Uno que traiga un listado de todas las notas creadas (listado general) y uno que permita traer una sola nota en específico (búsqueda específica). Para el listado general, las notas pueden ser filtradas u ordenadas por orden alfabético según el título, por fecha de creación o por fecha de modificación y la respuesta del endpoint son las notas obtenidas junto con todos sus datos, excepto el contenido de la nota. En cambio, la búsqueda específica solo trae una nota si se le indica su identificador único y la respuesta incluye todos los datos incluyendo el contenido.
- En vista que es un prototipo y no se ha levantado mucha parte de la lógica de negocio (aparte de lo comentado previamente), se desea implementar una fuente de datos distinta a una base de datos SQL. Para esto se puede usar archivos de texto o base de datos NoSQL enfocadas a documentos. Sin embargo, el servidor donde se implemente la API REST debe ser programado de tal forma que, en caso de cambiar a otra fuente de datos, no se vea afectada la lógica de negocio implementada en el servidor.
- El atributo fecha de modificación debe ser actualizado automáticamente cada que haya una modificación a la nota donde pertenezca. Los atributos que se pueden editar son el título y el contenido de la nota. El resto de atributos no son modificables.
- El endpoint de eliminación puede aceptar uno o varios identificadores de las notas que quieran ser eliminadas.

Requisitos No Funcionales

1. **Tecnologías a utilizar**
    - Para el desarrollo del servidor se requiere el uso de NestJS.
    - Para las pruebas unitarias y de integración se recomienda utilizar Jest.
    - Se debe hacer uso de Git para el control de versiones siguiendo los lineamientos dictados por GitFlow. Además, deben subir su repositorio a GitHub para realizar la defensa de su proyecto.
    - Para documentar los endpoints se debe integrar Swagger, además de utilizar JSDoc en donde sea necesario.
    - En caso de utilizar una base de datos NoSQL para persistencia de los datos del prototipo, se recomienda el uso de MongoDB.
2. **Pruebas unitarias e integración:**
    - El servidor debe poseer las pruebas automatizadas necesarias para garantizar que esté funcionando según lo requerido.
    - En caso de que no haya dependencias, sólo debe poseer pruebas unitarias. En caso de que haya dependencias, deben haber pruebas unitarias y de integración.
    - Se priorizan las pruebas sobre la lógica de negocio implementada sobre el servidor.
3. **Documentación:**
    - El repositorio de GitHub debe tener un archivo README.md que contenga los siguientes puntos:
        - Requisitos para ejecutar el prototipo (verisón de Node, manejador de base de datos, etc.)
        - Explicación paso a paso para poder ejecutar el prototipo en un entorno local
        - **Consejo**: Revisen los archivos README.md de varias librerías reconocidas para que se den una idea de cómo tienen que hacer el suyo
    - Cada función o método implementado debe poseer un comentario bajo el formato JSDoc.
4. **Patrones de diseño y Programación Orientada a Aspectos**
    - Aunque el enfoque principal del proyecto es la creación de una API REST, también se tomarán en cuenta los patrones de diseño y los aspectos adicionales que identifiquen e implementen para mantener la modularidad del servidor.

Evaluación del proyecto

Para la evaluación del proyecto se realizará una revisión durante la semana 16 (a partir del 15/01/2026), donde se les pide lo siguiente:

- El servidor disponible en GitHub para poder descargarlo
- El servidor funcionando de forma local en una computadora del equipo

Además de probar el funcionamiento de lo que implementaron, el equipo puede ser sometido a preguntas sobre el desarrollo del proyecto.

Detalles a tomar en cuenta

- El lenguaje de programación que debe utilizar para desarrollar todo el proyecto debe ser **TypeScript**.
- Los equipos deben estar compuestos por 2 o 3 personas. No se permiten proyectos individuales. A la hora de defender, todos los miembros deben estar presentes.
- En las revisiones del proyecto, el mismo deberá ser transpilado y ejecutado usando el editor de código Visual Studio Code. Se va a evitar el uso de cualquier otro entorno.
- No se requiere que se despliegue en la nube ningún componente del servidor o la base de datos, basta con una ejecución en su entorno local.
- Se tomará en cuenta en la evaluación general el uso de los conceptos vistos a lo largo del semestre.
- Para las revisiones del proyecto, solo se tomará en cuenta el código que se encuentre en la rama principal de los repositorios y que haya sido subido a la misma antes del día que comiencen las revisiones. NO SE REVISARÁ CÓDIGO EN OTRA RAMA FUERA DE LA PRINCIPAL NI QUE HAYA SIDO SUBIDO DESPUÉS DE LA FECHA TOPE (14/01/2026).
- También se tomará en cuenta las buenas prácticas aplicadas al código desarrollado, como la documentación del mismo, el estándar utilizado para nombrar variables y funciones y el buen uso de las funciones y librerías.
- En caso de no presentarse a la defensa del proyecto, la calificación del proyecto y del quiz será **NP** (No presentó).