Caso Práctico: Sistema de Gestión de Tareas

Descripción:

Estás trabajando en el desarrollo de un sistema de gestión de tareas (to-do list) utilizando NodeJS – VueJS o JavaScript. El objetivo es crear una aplicación que permita a los usuarios agregar, eliminar y listar tareas pendientes. Puedes utilizar el repositorio de información que desees para almacenar la información de las tareas.

Requisitos:

1. Crear una estructura de proyecto Node.js con los siguientes archivos:

* index.js: Archivo principal que manejará la interacción con el usuario y llamará a las funciones adecuadas.
* tasks.js: Archivo donde implementarás las funciones para agregar, eliminar, listar y cambiar de estado a tareas.

1. Implementar las siguientes funcionalidades en el archivo tasks.js:
2. addTask(taskDescription): Esta función deberá tomar la descripción de una tarea como parámetro y agregarla a un archivo de tareas pendientes.
3. RemoveTask(taskIndex): Esta función deberá tomar el índice de la tarea a eliminar y actualizar la lista de tareas pendientes en el archivo.
4. ListTasks(): Esta función deberá mostrar una lista numerada de todas las tareas pendientes.
5. updateTaskStatus(taskIndex, newStatus): Esta función deberá tomar el índice de la tarea y el nuevo estado como parámetros, y actualizar el estado de la tarea, los posibles estados de la tarea pueden ser Pendiente, en proceso y finalizado.
6. En el archivo index.js, implementa la lógica para que el usuario pueda interactuar con las funciones que has creado. Deberías proporcionar un menú de opciones que permita al usuario elegir entre agregar, eliminar, cambiar de estado o listar tareas.
7. Utiliza manejo de errores y validación de entrada para asegurarte de que el programa no falle ante entradas incorrectas o situaciones inesperadas.
8. Comenta tu código de manera clara para explicar cada paso y función que has implementado.

**Nota: Se evaluarán los siguientes aspectos:**

Organización del código y estructura del proyecto.

Uso adecuado de funciones y modularidad.

Uso del repositorio para el almacenamiento de tareas.

Implementación de manejo de errores y validación de entrada.

Interfaz amigable para el usuario.

Comentarios y claridad del código.