IGRTEC

Santiago de Querétaro, Querétaro
México

Ejercicio Práctico: Implementación de una API RESTful para Gestión de Tareas

Descripción del Ejercicio:

Desarrolla una API RESTful en C# que permita a los usuarios gestionar tareas. La API debe permitir crear, leer, actualizar y eliminar tareas (operaciones CRUD). Además, debe permitir marcar las tareas como completadas.

Requisitos:

1. Modelo de Datos:

- **Task (Tarea):**
  - `Id` (int): Identificador único de la tarea.
  - `Title` (string): Título de la tarea.
  - `Description` (string): Descripción detallada de la tarea.
  - `IsCompleted` (bool): Indica si la tarea ha sido completada.
  - `CreatedAt` (DateTime): Fecha y hora en la que se creó la tarea.
  - `UpdatedAt` (DateTime): Fecha y hora en la que se actualizó por última vez la tarea.

2. Rutas de la API:

- `GET /api/tasks`: Obtener la lista de todas las tareas.
- `GET /api/tasks/{id}`: Obtener una tarea específica por su `Id`.
- `POST /api/tasks`: Crear una nueva tarea.
- `PUT /api/tasks/{id}`: Actualizar una tarea existente.
- `DELETE /api/tasks/{id}`: Eliminar una tarea existente.
- `PATCH /api/tasks/{id}/complete`: Marcar una tarea como completada.

3. Validaciones:

- El título de la tarea no debe estar vacío.
- La descripción es opcional.
- `IsCompleted` debe ser `false` por defecto al crear una nueva tarea.
- Al actualizar una tarea, se debe actualizar también la propiedad `UpdatedAt` con la fecha y hora actual.

4. Manejo de Errores:

- Retornar un código HTTP `404 Not Found` si se intenta acceder a una tarea que no existe.
- Retornar un código HTTP `400 Bad Request` si los datos enviados para crear o actualizar una tarea son inválidos.

5. Requerimientos Técnicos:

- Utiliza ASP.NET Core como framework.
- Organiza el código en capas (Controladores, Servicios, Modelos).
- Implementa la API de manera que sea fácil de extender y mantener.
- Escribe pruebas unitarias para los métodos de la API.

Criterios de Evaluación:

- Estructura y organización del código.
- Correcto uso de patrones de diseño como la inyección de dependencias.
- Manejo adecuado de las rutas y métodos HTTP.
- Implementación de validaciones y manejo de excepciones.
- Documentación y claridad del código.
- Cobertura y efectividad de las pruebas unitarias.

Extensiones Opcionales:

- Implementar autenticación JWT para proteger las rutas de la API.
- Permitir la paginación y búsqueda de tareas.
- Implementar un sistema de categorías o etiquetas para las tareas.

Datos de Acceso a la Base de Datos:

- **Usuario:** Evaluado
- **Contraseña:** Mgjx76^44
- **Base de datos:** Evaluacion_Apis
- **Acceso:** 216.219.86.206

Utiliza estos datos para crear tu propia tabla en la base de datos con tu nombre. Asegúrate de que la tabla tenga un esquema que refleje los datos de la API implementada.

Al finalizar tu evaluación mandar al siguiente correo el repositorio en GitHub cargado:

Jahir.hernandez@igrtec.com

Jose.orzuna@igrtec.com

Jair.guzman@igrtec.com