Enunciado práctico – Examen 16-12-24 – Programación III – TUP FRC

Proyecto API:
Desarrollar un proyecto de API con .NET utilizando el enfoque "Code First" de EF Core.
A continuación se detallarán las tablas a crear.

1. **Usuario**:
    - Id: Guid **PK**
    - Nombre: string
    - Apellido: string
    - Email: string
    - Password: string
    - FechaNacimiento: DateTime
    - IdRol: Guid **FK**
    - Activo: Bool
    - FechaAlta: DateTime
    - FechaModificacion: DateTime? (Fecha de la última modificación, puede ser nula si no se ha modificado)
2. **Rol**:
    - Id: Guid **PK**
    - Nombre: string
    - Descripcion: string
    - FechaCreacion: DateTime
    - FechaModificacion: DateTime? (Fecha de la última modificación, puede ser nula si no se ha modificado)
3. **TokenXUsuario**:
    - Id: Guid **PK**
    - Token: string
    - DateTimeValid: DateTime
    - IdUsuario: Guid
4. **Productos**:
    - Id: Guid **PK**
    - Nombre: string
    - Descripción: string
    - IdCategoria: Guid **FK**
    - FechaCreacion: DateTime
    - FechaModificacion: DateTime?
5. **Categoria:**
    - Id: Guid **PK**
    - Nombre: string

Endpoints a desarrollar:

1. UsuarioController:
    - Crear un GET que devuelva un usuario particular por Id
    - Crear un POST que permita realizar login de usuario por mail y password. En caso que el usuario exista, se deberá generar un token (un string de 400 caracteres) el cual se almacene en la tabla correspondiente y se le dé una validez de 15 minutos.
    - Crear un POST que permita registrar un usuario.
2. ProductoController:
    - Crear un GET que permita recibir por Query String la cantidad de productos que desea obtener. Basicamente, se deberá devolver una lista con los primeros X productos registrados en la base de datos (en donde X es el valor recibido por query string).

- Crear un GET que permita obtener un producto específico por su Id.

Aclaraciones para la API:

- Los modelos y relaciones deben estar correctamente mapeados utilizando el enfoque Code First.
- Las claves primarias y foráneas deben estar correctamente configuradas en las tablas.
- Las validaciones para el manejo de datos son obligatorias, especialmente en campos como Email (único), etc.
- El manejo de errores y las respuestas deben ser claras, con códigos HTTP adecuados y mensajes informativos para el cliente de la API.
- **TODOS los endpoints, salvo el de login y registro, deberán verificar si el token recibido por HEADER es válido o no, en caso que no sea válido, se deberá devolver el error http correspondiente.**
- **Si un usuario inicia sesión, se deberá validar, además del password e email, si posee token registrado o no. En caso que si posea token, eliminarlo y cargar el nuevo.**
- **Siempre que se reciba un campo de tipo email, se deberá verificar que sea una dirección de correo electrónica valida.**

Proyecto FrontEnd:

1. Crear una página HTML la cual permita realizar un login a la aplicación.
2. En caso que el login sea exitoso, se deberá redirigir a otra página html la cual me muestre los primero 10 productos cargados en el sistema. Pero tener en cuenta que solo se deberá mostrar, el nombre, la descripción, la fecha de creación en formato “dd/MM/yyyy” y el nombre de su categoría.

Aclaraciones FrontEnd:

- Mostrar mensajes de éxito o error de forma clara y con un diseño amigable.
- La interfaz debe ser **responsiva**, utilizando Bootstrap para adaptarse a diferentes tamaños de pantalla.
- En caso que el token de autenticación no sea válido, se deberá redireccionar al login nuevamente y mostrar el mensaje de error pertinente.