Buenas tardes.

A continuación, realizo una breve explicación de la prueba realizada.

A pesar de que el eventDate debe ser de forma automática mediante el DateTime.Now Opte por ingresarlo para asi poder ingresar distintas fechas y probar los filtros

1. Controladores, Prueba Back end
   1. **Get-Events**: este controlador retorna todos los eventos si alguno de los parámetros llega null (type, startDate, endDate)

**Con parámetros:**

Captura de pantalla de un celular

Descripción generada automáticamente

**Sin parámetros:**

Texto

Descripción generada automáticamente

* 1. **Add-Log**: este controlador agrega un nuevo evento y retorna un mensaje de respuesta si es exitoso o si es fallido, controlando la excepción

**Captura de pantalla de un celular

Descripción generada automáticamenteRespuesta exitosa**

**respuesta Fallida:** en este caso el try catch captura la excepción ya que el “Id”:1 existe y es único ya que el la clave primaria y auto incrementable por lo tanto se muestra un mensaje personalizado.

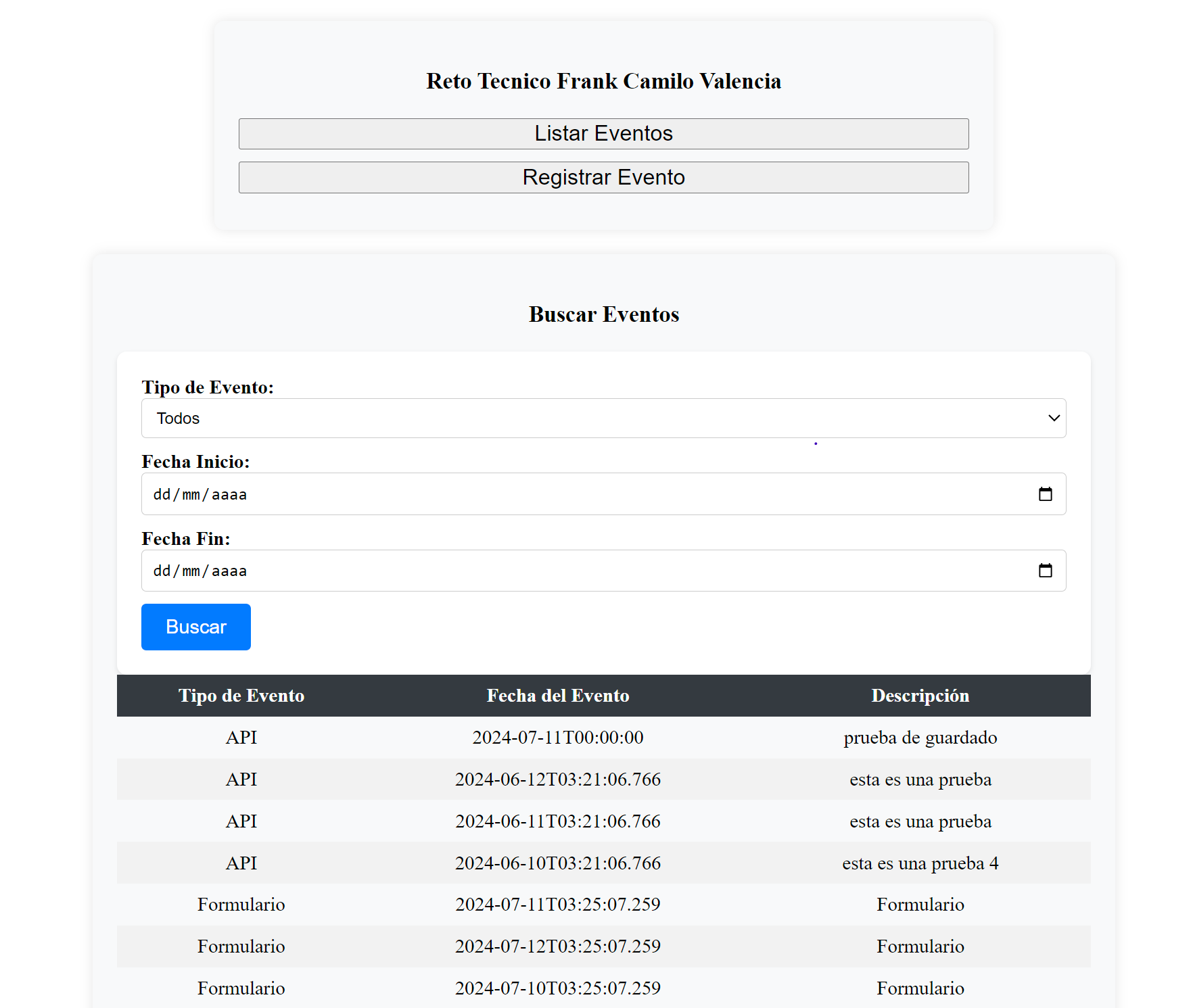
Captura de pantalla de computadora

Descripción generada automáticamente

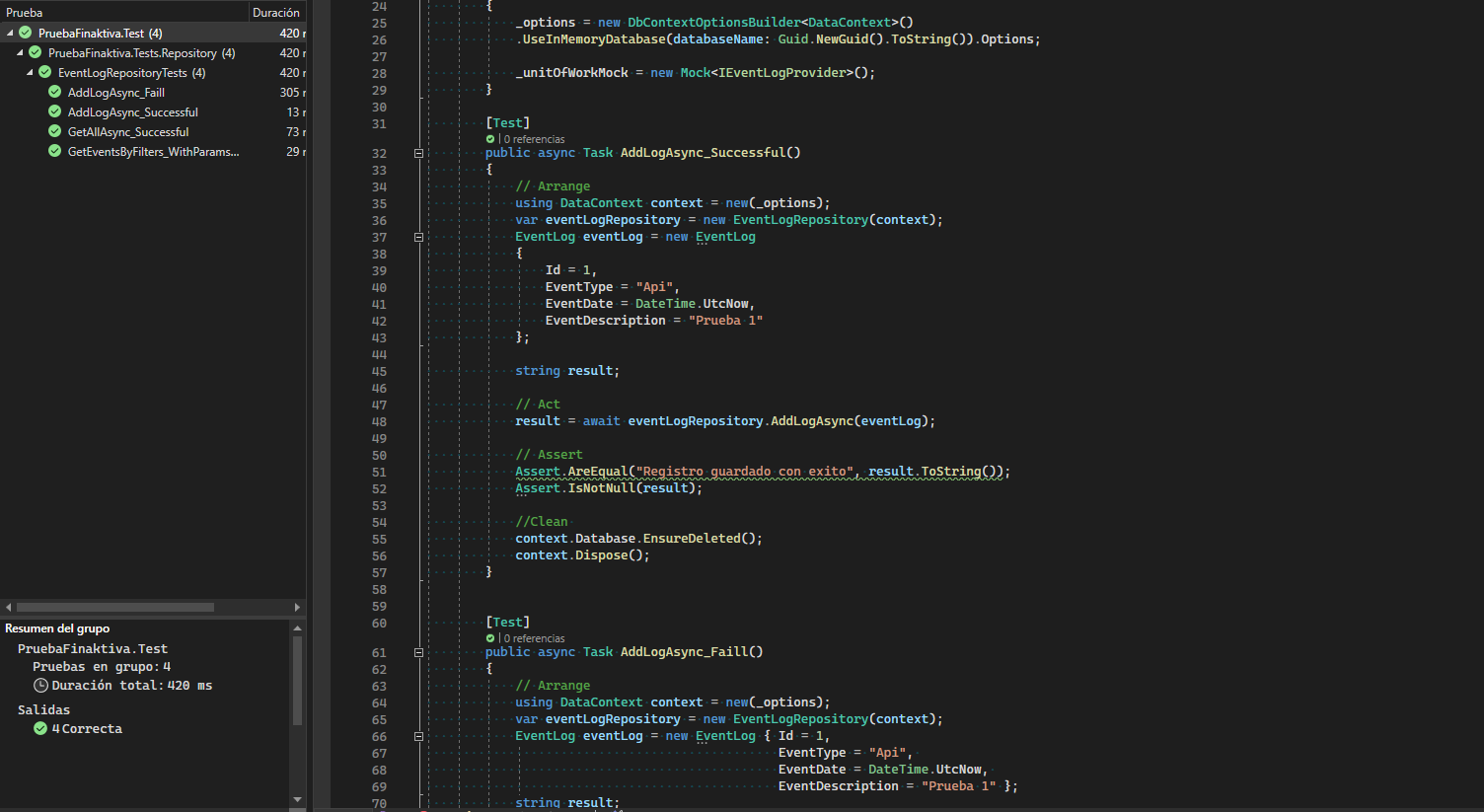
1. **Base de datos en SQL**



1. Desde angular el formulario se ve asi:
   1. Interfaz de usuario gráfica, Aplicación

      Descripción generada automáticamenteRegistro de eventos:
   2. Búsqueda sin filtros:
   3. Interfaz de usuario gráfica, Texto, Aplicación, Correo electrónico

      Descripción generada automáticamenteBusqueda con filtros
2. Pruebas unitarias



1. Nota: estos son algunas consideraciones sobre la prueba:
   1. Se implementa una interfaz para la lógica del negocio asi los métodos dependen de la implementación de esta
   2. se implementa inyección de dependencias para la interfaz y la conexión a la base de datos
   3. se realiza el back end en .Net CORE 6 y el front end en Angular 17
   4. se siguen lineamientos SOLID y clean code
   5. Se siguen lineamientos standard para api Rest al igual que para .Net
   6. Se agregan pruebas unitarias en el back end