Manual Técnico API

Para poder ejecutar el proyecto necesitamos primeramente descargarlo de la plataforma de git: <https://github.com/RonaldoSarahi/exam_cSharp>

1. Tenemos que configurar el archivo appsettings.json donde colocaremos el nombre de nuestro servidor de base de datos local y también el nombre de la base de datos.

"Connection": "Data Source=DELLEFAC\\RONALDO; Initial Catalog=cSharpExam; Integrated Security=True; TrustServerCertificate=True;",

1. La base de datos la crearemos por Code First, es requerido ejecutar los siguientes comandos para poder generar tanto la base de datos como las tablas y sus relaciones; todo esto se realiza en la terminal

Herramientas/Administrador de paquetes NuGet/Consola de administrador de paquetes, los comandos serian:

Add-Migration <Nombre de la migración>

Donde se generará una carpeta de todos los modelos que tenemos creados en el proyecto, así como sus relaciones.

Después de revisar el archivo generado, procedemos a ejecutar el ultimo comando para generar tablas en la base de datos.

Update-DataBase

1. Se modifica el lauchSettings.json para quitar el sslPort en específico de la **csharp**\_api y configurarlo en el program.
2. Para la conexión de la base de datos dentro de nuestro proyecto **csharp\_api** se crea un ExamDbContext donde damos de alta los modelos y la configuración de los modelos y creamos la conexión obteniendo la Connection de nuestro appsettings.json. Despues dicha conexión la inyectaremos a los Services que vallamos a utilizar.
3. Se crea una carpeta para colocar todas las configuraciones de los modelos y las relaciones con las demás tablas en: Data/ EntityTypeConfiguration.
4. Se crea la interfaz para este a su vez ser usado en el Controlador correspondiente.
5. Se configura el cSharpCORSPolicy para que el proyecto web pueda consumir la API.
6. Para realizar las pruebas de consumo de la API se puede utilizar la aplicación Postman usando las siguientes URL.

[GET] Obtener todos los Artículos

[*http://localhost:5002/api/articulos*](http://localhost:5002/api/articulos)

[GET] Obtener un solo Articulo

[*http://localhost:5002/api/articulos/{articulo}*](http://localhost:5002/api/articulos/%7barticulo%7d)

[POST] Crear un Articulo

[*http://localhost:5002/api/articulos/*](http://localhost:5002/api/articulos/)

Pasamos en el body:

 {

        "articulo": "Prueba",

        "descripcion": "Prueba de Articulo"

  }

[PUT] Editar un Articulo

[*http://localhost:5002/api/articulos/*](http://localhost:5002/api/articulos/)

Aquí podemos ya sea mandar la llave primaria de Artículos dentro de la URL para modificar el dato o mandar el Objeto y validar al realizar la modificación.

 {

        "articulo": "Prueba",

        "descripcion": "Cambio en el Articulo"

  }

[DELETE] Eliminar un Articulo

[*http://localhost:5002/api/articulos/{articulo}*](http://localhost:5002/api/articulos/%7barticulo%7d)

Manual Técnico WEB

1. Para ejecutar el proyecto **csharp\_web** se requiere instalar el NuGet siguiente:

Grid.Mvc

Dicho paquete será usado para crear el Grid y mostrar todos los registros de nuestro Modelo.

1. Tenemos un controlador para crear el CRUD de los Artículos donde tenemos un método para poder consumir la API por medio de la URL configurada con el httpClient.
2. Se realiza un \_GridView.cshtml para crear la tabla y podemos usarla dentro del Index.cshtml de Artículos.
3. Creamos la vista de Articulos donde en el Index.cshtml colocaremos las funciones a utilizar para cada una de las funcionalidades de los botones y validaciones necesarias para el CRUD.
4. Para el alta de un nuevo Artículo se utiliza una vista Parcial UpdateModal.cshtml donde se despliega los campos necesarios para la Alta del Articulo

