

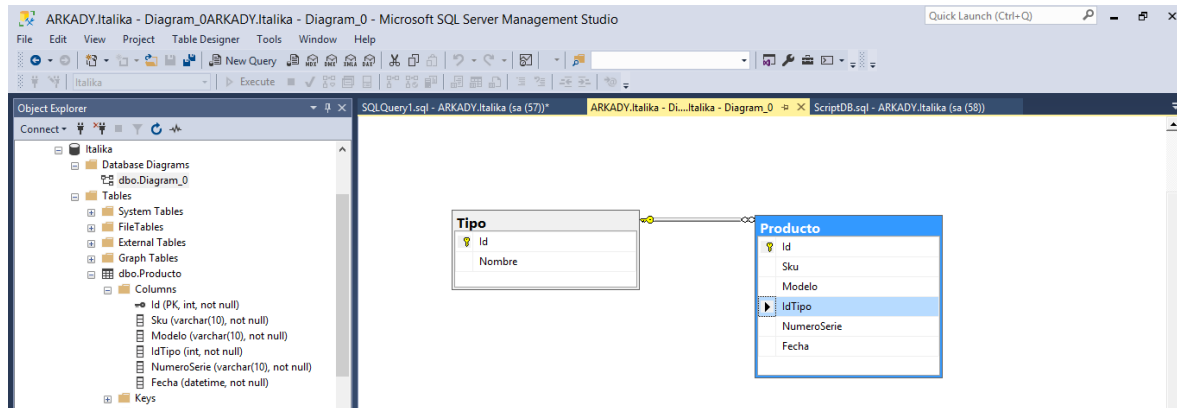
DESARROLLO DE UN PROYECTO WEB (CAPA DE PRESENTACION) CON CONECCIÓN A UNA WEB API (CAPA DE DATOS) Y BASE DE DATOS EN SQL SERVER PARA ITALIKA

Contenido

Desarrollo de la Base de Datos.....	2
Desarrollo de la Web Api.....	3
Desarrollo del portal Web MVC.....	5

Desarrollo de la Base de Datos

Se creo una Base de Datos en Sql Server 2019 una tabla Producto que representa a las Motocicletas de la cadena Italika con las especificaciones básicas como Numero de Serie y SKU además del ID y con su respectiva relación hacia un catalogo de Tipo donde específicamente se generaron 3 tipos de Motocicletas: (TRABAJO,DEPORTIVA,INFANTIL)



Se Setearon algunos datos y se generaron algunos Stored Procedures para que más adelante las consultas fuesen más eficientes.

```
INSERT INTO Tipo(Nombre) VALUES('Trabajo')
INSERT INTO Tipo(Nombre) VALUES('Deportiva')
INSERT INTO Tipo(Nombre) VALUES('Infantil')

Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150REDATH', 'LCXXLD17XE', '2020-01-01', 1)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150BLAKATH', 'LCXXLD18XE', '2020-01-01', 1)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150BLUEATH', 'LCXXLD19XE', '2020-01-01', 1)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150REDSTD', 'LCXXLD20XE', '2020-01-01', 1)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150BLAKSTD', 'LCXXLD21XE', '2020-01-01', 1)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150BLUESTD', 'LCXXLD22XE', '2020-01-01', 1)

Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150REDATH', 'LCXXLD23XE', '2020-01-01', 2)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150BLAKATH', 'LCXXLD24XE', '2020-01-01', 2)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC1', '150BLUEATH', 'LCXXLD25XE', '2020-01-01', 2)

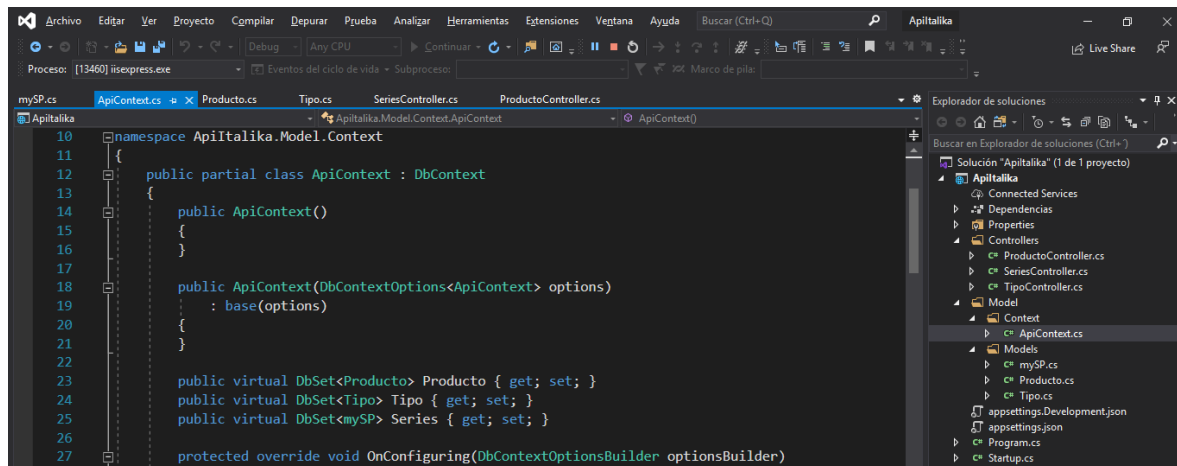
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC2', '150REDSTD', 'LCXXLD26XE', '2020-01-01', 3)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC2', '150BLAKSTD', 'LCXXLD27XE', '2020-01-01', 3)
Insert INTO Producto(Sku, Modelo, NumeroSerie, Fecha, IdTipo) VALUES('CDMXSUC2', '150BLUESTD', 'LCXXLD28XE', '2020-01-01', 3)

CREATE PROC BuscarTodos
as
Begin
    SELECT Id, Sku, Modelo, IdTipo, NumeroSerie, Fecha FROM Producto
end
EXEC BuscarTodos
```

Id	Sku	Modelo	IdTipo	NumeroSerie	Fecha
1	CDMXSUC1	150REDATH	1	LCXXLD17XE	2020-01-01 00:00:00.000
2	CDMXSUC1	150BLUEATH	1	LCXXLD19XE	2020-01-01 00:00:00.000
3	CDMXSUC1	150REDSTD	1	LCXXLD20XE	2020-01-01 00:00:00.000

Desarrollo de la Web Api

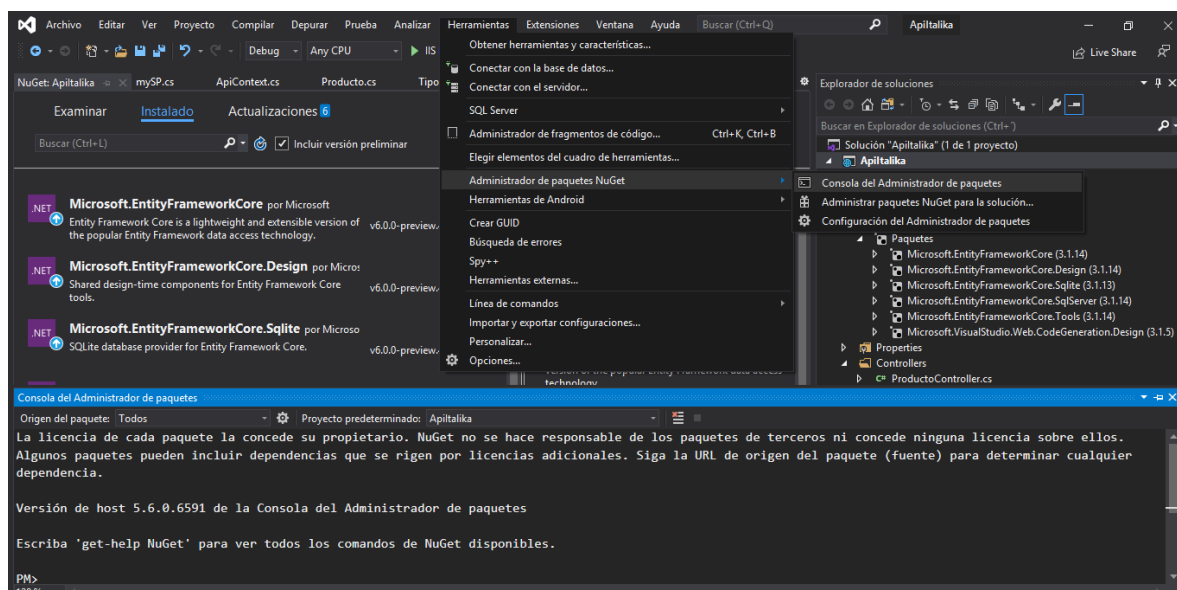
Para la manipulación de los datos de forma asíncrona, se creo un nuevo proyecto Web Api en el Framework de .Net Core para administrar los datos de la base de datos previamente creada. Con ayuda de la herramienta ORM Entity Framework el proceso de construcción de un modelado de datos Cotextual, así como las tablas o 'DBSets', fue rápido y sin errores.



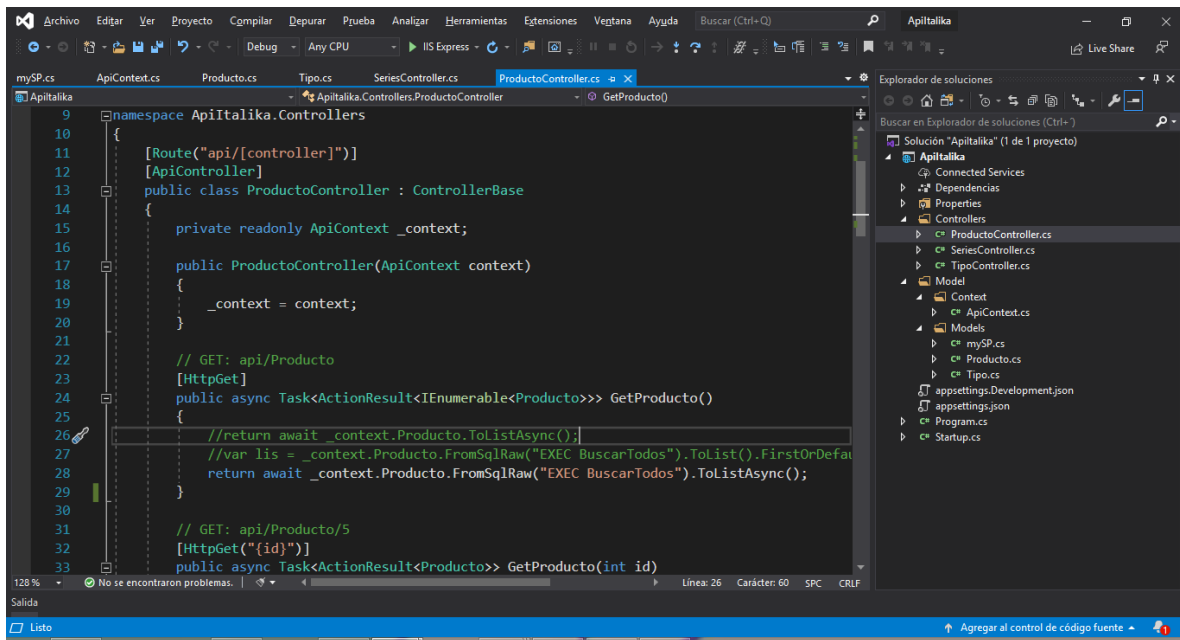
Cabe mencionar la cadena usada para ejecutar el ORM Entity fue la siguiente

```
scaffold-DbContext "Server=ARKADY;Database=Italika;User Id=sa;Password=*****;"  
Microsoft.EntityFrameworkCore.SqlServer -OutputDir Models/Models -ContextDir  
Models/Context -Context SecContext -Force
```

Una vez instalados los paquetes Nuget del Entity Framework Adecuados y apertura la consola de comandos del Administrador de paquetes de Nuget que nos proporciona Visual Studio

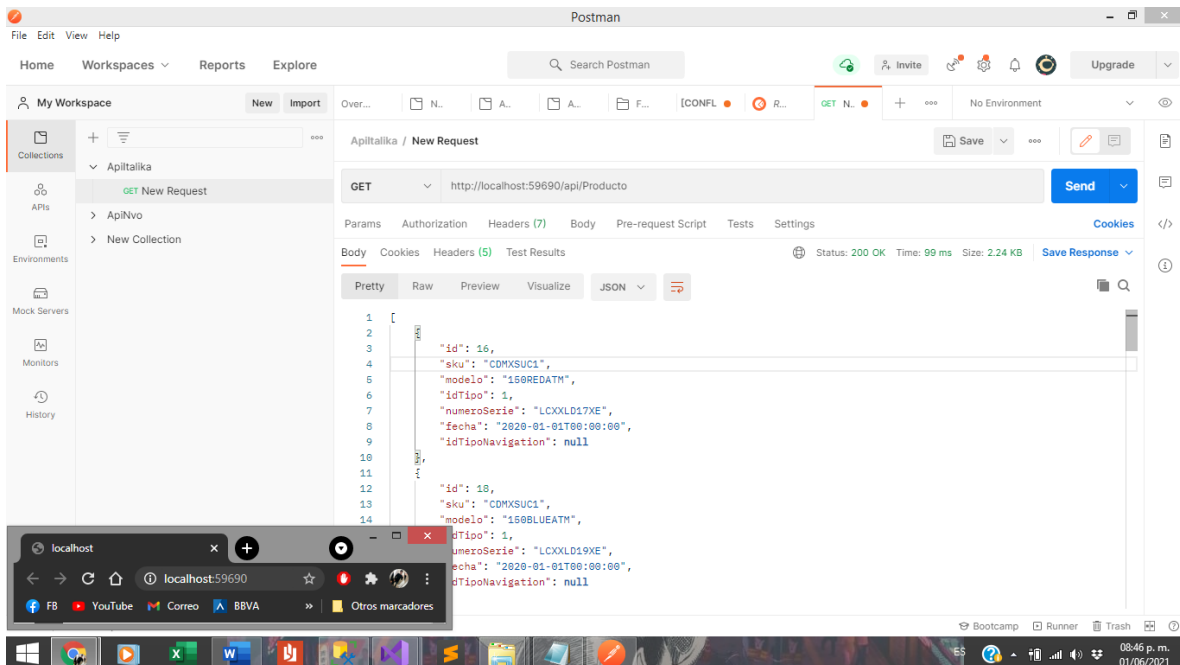


Se generan los Controladores y se hacen los cambios pertinentes para que cada petición funcione de forma correcta.



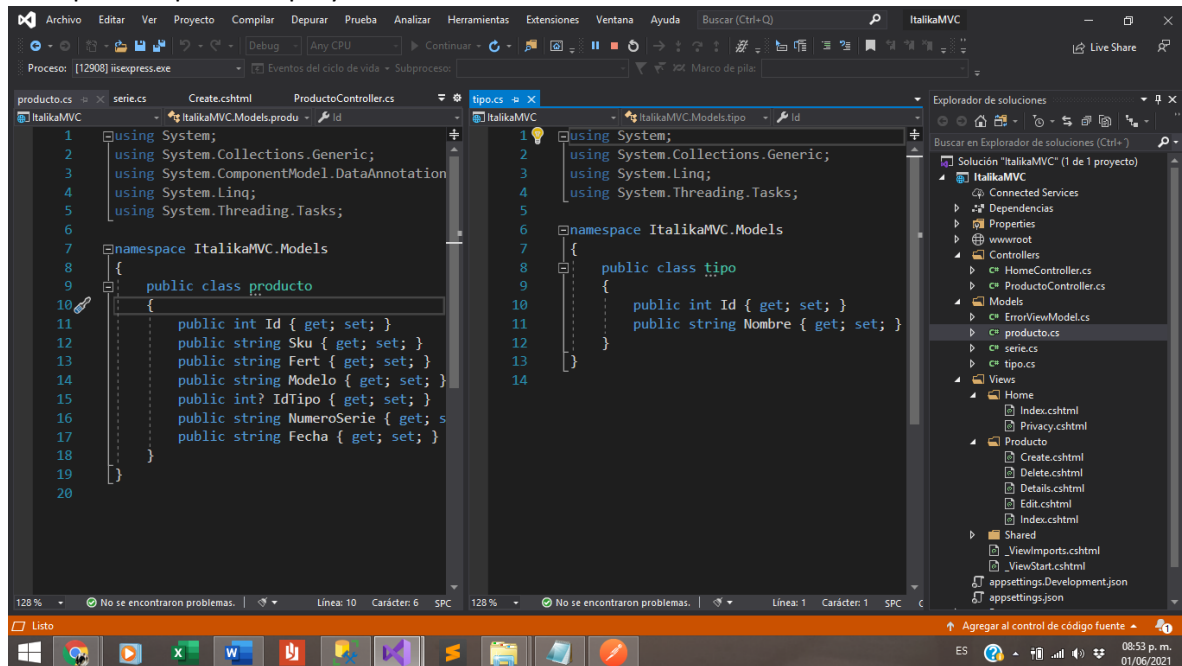
```
9 namespace Apitalika.Controllers
10 {
11     [Route("api/[controller]")]
12     [ApiController]
13     public class ProductoController : ControllerBase
14     {
15         private readonly ApiContext _context;
16
17         public ProductoController(ApiContext context)
18         {
19             _context = context;
20         }
21
22         // GET: api/Producto
23         [HttpGet]
24         public async Task<ActionResult<IEnumerable<Producto>>> GetProducto()
25         {
26             //return await _context.Producto.ToListAsync();
27             //var lis = _context.Producto.FromSqlRaw("EXEC BuscarTodos").ToList().FirstOrDefault();
28             return await _context.Producto.FromSqlRaw("EXEC BuscarTodos").ToListAsync();
29         }
30
31         // GET: api/Producto/5
32         [HttpGet("{id}")]
33         public async Task<ActionResult<Producto>> GetProducto(int id)
```

Ya con la compilación correcta y la ejecución de la web api usamos **Postman** para probar la funcionalidad correcta de cada petición.

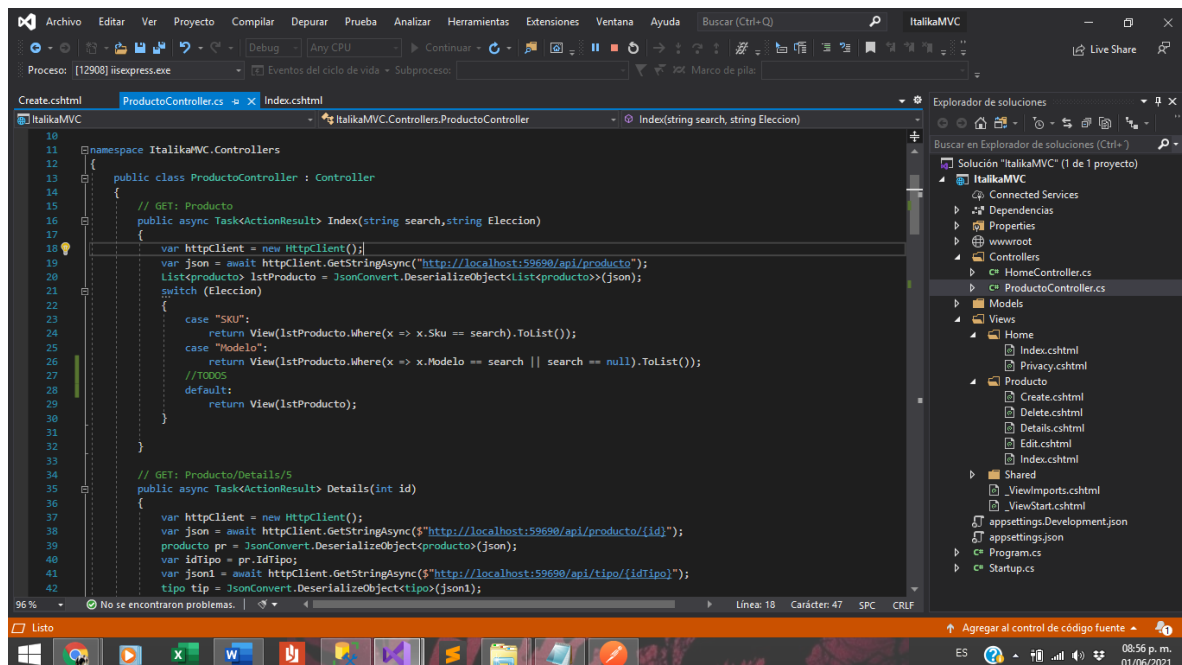


Desarrollo del portal Web MVC

En la primera parte del proyecto se desarrollan los modelos

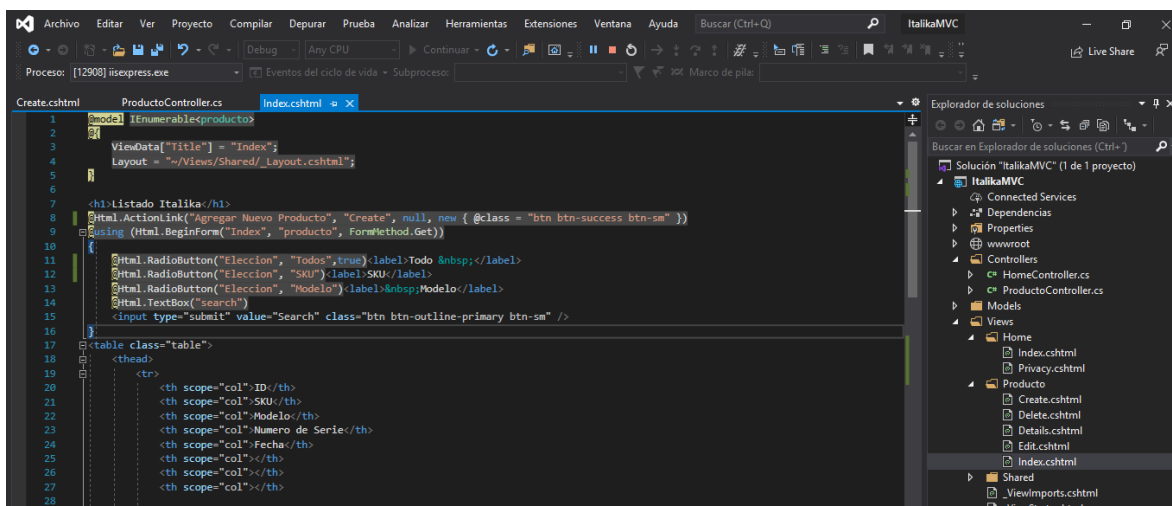


Se generan los Controladores en base a los modelos de la capa de presentación

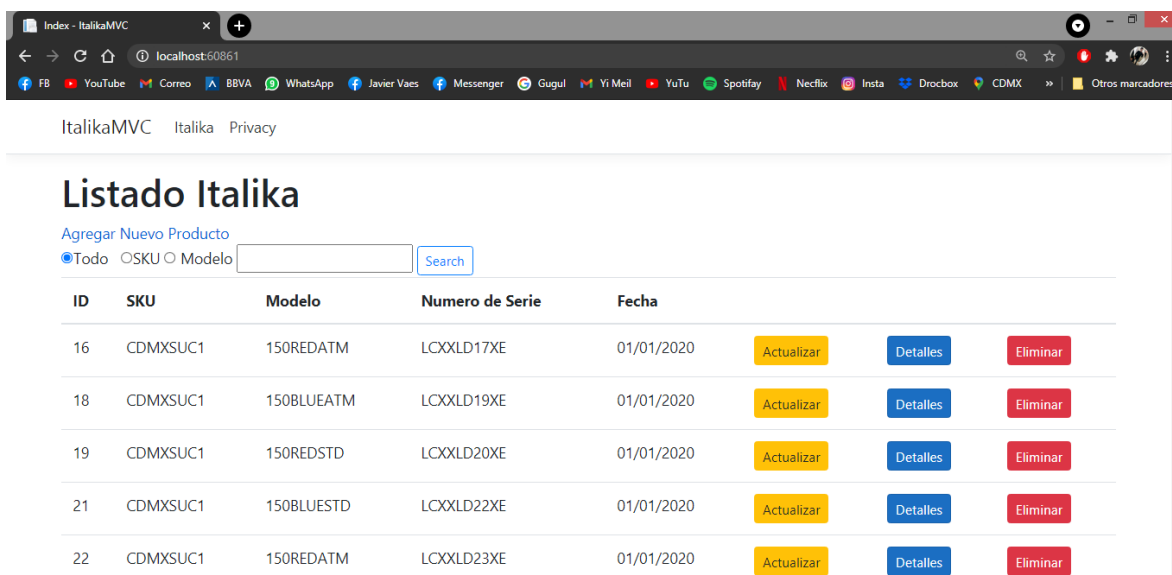


Se generan los métodos para la manipulación de los datos y las vistas uno por uno, en especial detalle con la de editar y la de agregar, para usar el Store Procedure en base a los modelos y presentar los diferentes tipos de motocicletas en el catalogo, así como una edición correcta.

Se generan las vistas nuevamente con ayuda del Scaffolding de Visual Studio



Así se ve finalmente el index, con los botones pertinentes para el funcionamiento del CRUD



Pongo énfasis nuevamente en el código de los Radio Buttons generados para poder cumplir con las características de visualizar solo Motocicletas por modelo, o por SKU

```
@using (Html.BeginForm("Index", "producto", FormMethod.Get))
{
    @Html.RadioButton("Eleccion", "Todos", true) <label> Todo &nbsp;</label>
    @Html.RadioButton("Eleccion", "SKU") <label> SKU </label>
    @Html.RadioButton("Eleccion", "Modelo") <label> &nbsp;Modelo </label>
    @Html.TextBox("search")
    <input type="submit" value="Search" class="btn btn-outline-primary btn-sm" />
}
```

ESO ENVIA AL CONTROLADOR LAS OPCIONES DEL SWITCH para que sepa que imprimir y como presentar la vista del Index.

```

switch (Eleccion)
{
    case "SKU":
        return View(IstProducto.Where(x => x.Sku == search).ToList());
    case "Modelo":
        return View(IstProducto.Where(x => x.Modelo == search || search == null).ToList());
    //TODOS
    default:
        return View(IstProducto);
}

```

Mostrando el Index con el SKU de sucursal 2, solo los registros con ese SKU aparecen

Listado Italika

[Agregar Nuevo Producto](#)

☐ Todo
 ☒ SKU
 ☐ Modelo

ID	SKU	Modelo	Numero de Serie	Fecha			
25	CDMXSUC2	150REDSTD	LCXXLD26XE	01/01/2020	Actualizar	Detalles	Eliminar
27	CDMXSUC2	150BLUESTD	LCXXLD28XE	01/01/2020	Actualizar	Detalles	Eliminar
31	CDMXSUC2	150BLAKSTD	LCXXLD27XE	01/01/2020	Actualizar	Detalles	Eliminar

Vista de la creación de un nuevo Producto

Sku

Modelo

IdTipo

NumeroSerie

Fecha

[Back to List](#)

Redireccionamiento y presentación del nuevo registro al fondo de la tabla

38	CDMXSUC3	150WHTESTD	LCXXLD50XE	01/06/2021	Actualizar	Detalles	Eliminar
----	----------	------------	------------	------------	----------------------------	--------------------------	--------------------------

© 2021 - ItalikaMVC - Privacy

Vista de edición

Edit

Producto

Sku
CDMXSUC3

Modelo
150WHTSTD

IdTipo
Deportiva

NumeroSerie
LCXXLD99XE

Fecha
01/06/2021

Save

37	CDMXSUC3	150GREESTD	LCXXLD35XE	01/06/2021	Actualizar	Detalles	Eliminar
38	CDMXSUC3	150WHTSTD	LCXXLD99XE	01/06/2021	Actualizar	Detalles	Eliminar

DETALLES

Details - ItalikaMVC

localhost:60861/producto/Details/38

ItalikaMVC Italika Privacy

Id	38
Sku	CDMXSUC3
Modelo	150WHTSTD
Tipo	Deportiva
NumeroSerie	LCXXLD99XE

[Regresar a la lista](#)

ELIMINAR

Delete - ItalikaMVC

localhost:60861/producto/Delete/38

ItalikaMVC Italika Privacy

Delete

Sku
CDMXSUC3

Modelo
150WHTSTD

IdTipo
Deportiva

NumeroSerie
LCXXLD99XE

Fecha
2021-06-01T00:00:00

Delete | [Back to List](#)

Así bien vemos como se refresca de forma correcta, dinámicamente y Asíncrona

Link de Repositorio

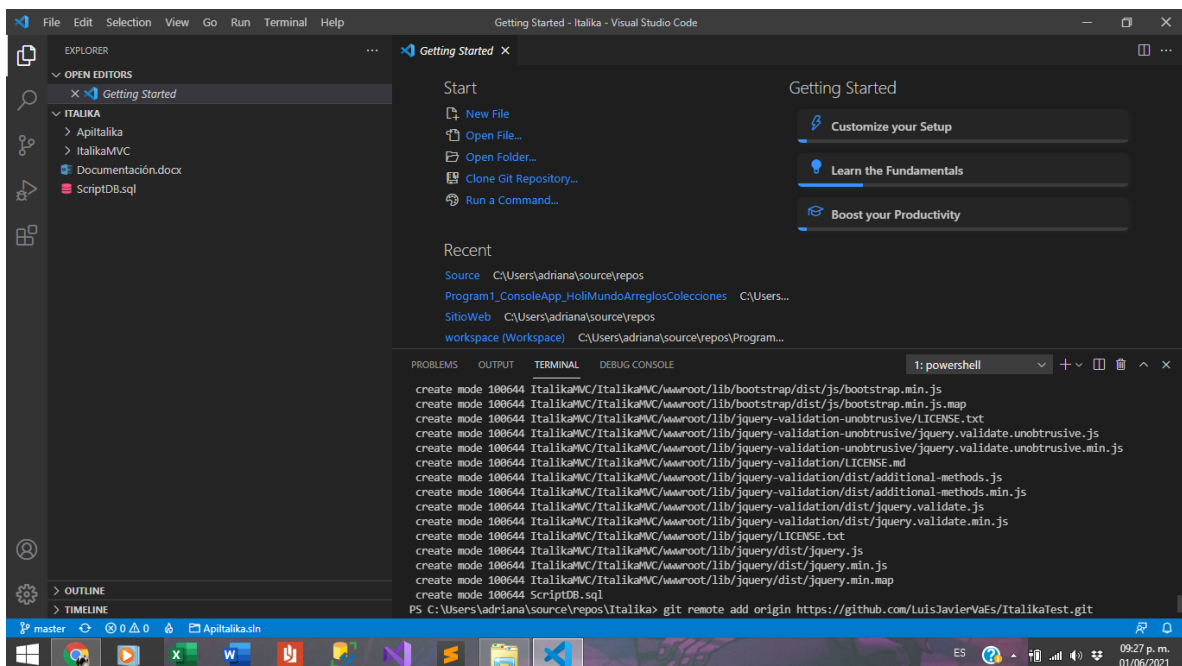
Subida con Visual Studio Code y Git en GitHub

```
PS C:\Users\adriana\source\repos\Italika> git commit
On branch master

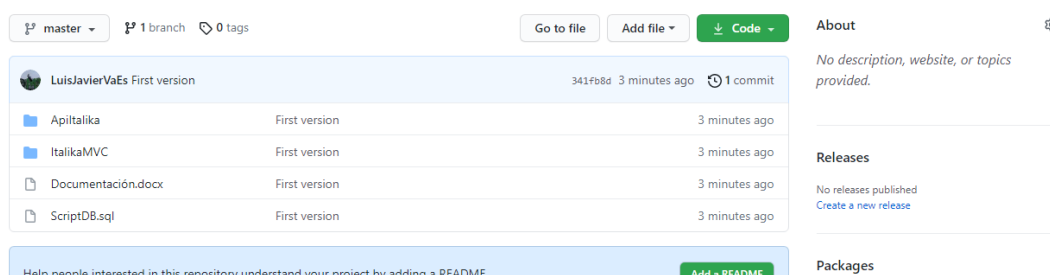
Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)
  ApiItalika/
  "Documentaci383\263n.docx"
  ItalikaMVC/
  ScriptDB.sql
  "~$documentaci383\263n.docx"
  ~$RL2489.tmp

nothing added to commit but untracked files present (use "git add" to track)
PS C:\Users\adriana\source\repos\Italika>
```



GITHUB



Repositorio: <https://github.com/LuisJavierVaEs/ItalikaTest.git>