Learn Asp.Net Core Web API

Repository Pattern

Step 1:

Create a new Project and name it Module2.

Add Models Folder, create a Product Class, and put the following code in this class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;

namespace Module2.Models
{
    public class Product
    {
        public int Id { get; set; }
        public string ProductName { get; set; }
        public int Price { get; set; }
    }
}
```

Step 2:

First, you will need to create a folder and name this folder to Services.

Step 3:

Then in the services folder just create an interface, name it IProduct, and make sure to make it public and just add some methods in this IProduct interface like

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Module2.Models;

namespace Module2.Services
{
    public interface IProduct
    {
        IEnumerable<Product> GetProducts();
        Product GetProduct(int id);
        void AddProduct(Product product);
        void UpdateProduct(Product product);
        void DeleteProduct(int id);
    }
}
```

Step 4:

Create a class in the services folder, name it ProductRepository, and put the following code

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Module2.Data;
using Module2.Models;

namespace Module2.Services
{
    public class ProductRepository : IProduct
    {
        private ProductsDbContext productsDbContext;

        public ProductRepository(ProductsDbContext _productsDbContext)
        {
            productsDbContext = _productsDbContext;
        }
        public IEnumerable<Product> GetProducts()
        {
            return productsDbContext.Products;

        }

        public Product GetProduct(int id)
        {
            var product = productsDbContext.Products.SingleOrDefault(m => m.Id == id);
            return product;
        }

        public void AddProduct(Product product)
        {
            productsDbContext.Products.Add(product);
            productsDbContext.SaveChanges(true);
        }

        public void UpdateProduct(Product product)
        {
            productsDbContext.Products.Update(product);
            productsDbContext.SaveChanges(true);
        }

        public void DeleteProduct(int id)
        {
            var product = productsDbContext.Products.Find(id);
            productsDbContext.Products.Remove(product);
            productsDbContext.SaveChanges(true);
        }
    }
}
```

Step 5:

Create a folder and name it Data and then add a class in this Data Folder and name it ProductsDbContext and add the following code in this class

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.EntityFrameworkCore;
using Module2.Models;

namespace Module2.Data
{
    public class ProductsDbContext : DbContext
    {
        public
ProductsDbContext(DbContextOptions<ProductsDbContext>options):base(options)
        {

        }
        public DbSet<Product> Products { get; set; }
    }
}
```

Step 6:

In the controller folder add a controller and name it ProductsController and then put the following code in this ProductsController

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;
using Microsoft.AspNetCore.Mvc.ModelBinding;
using Module2.Data;
using Module2.Models;
using Module2.Services;

namespace Module2.Controllers
{
    [Produces("application/json")]
    [Route("api/Products")]
    public class ProductsController : Controller
    {
```

```csharp
private IProduct productRepository;

public ProductsController(IProduct _productRepository)
{

    productRepository = _productRepository;
}

// GET: api/Products
[HttpGet]
public IEnumerable<Product> Get()
{
    return productRepository.GetProducts();
}

// GET: api/Products/5
[HttpGet("{id}", Name = "Get")]
public IActionResult Get(int id)
{
    var product = productRepository.GetProduct(id);
    if (product == null)
    {
        return NotFound("No Record Found...");
    }

    return Ok(product);
}

// POST: api/Products
[HttpPost]
public IActionResult Post([FromBody]Product product)
{

    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    productRepository.AddProduct(product);
    return StatusCode(StatusCodes.Status201Created);

}

// PUT: api/Products/5
[HttpPut("{id}")]
public IActionResult Put(int id, [FromBody]Product product)
{
    if (!ModelState.IsValid)
    {
        return BadRequest(ModelState);
    }

    if (id != product.Id)
    {
        return BadRequest();
    }

    try
```

```
            {
                productRepository.UpdateProduct(product);
            }
            catch (Exception e)
            {
                Console.WriteLine(e);
                return NotFound("No Record Found against this Id...");
            }
            return Ok("Product Updated...");

        }

        // DELETE: api/ApiWithActions/5
        [HttpDelete("{id}")]
        public IActionResult Delete(int id)
        {
            productRepository.DeleteProduct(id);
            return Ok("Product Deleted...");
        }
    }
}
```

Step 7:

Open the startup.cs file and in the configure service methods let's add replace the configureservice method with this code

```
public void ConfigureServices(IServiceCollection services)
        {
            services.AddMvc();
            services.AddDbContext<ProductsDbContext>(option => option.UseSqlServer(@"Data
Source=(localdb)\MSSQLLocalDB;Initial Catalog=ProductsDb;"));
            services.AddScoped<IProduct, ProductRepository>();
        }
```

Step 8:

Right after that just go to the configure methods let's add replace the configure method with this code

```
public void Configure(IApplicationBuilder app, IHostingEnvironment env,ProductsDbContext productsDbContext)
        {
            if (env.IsDevelopment())
            {
                app.UseDeveloperExceptionPage();
            }

            app.UseMvc();
            productsDbContext.Database.EnsureCreated();


           }
```