**Tech Stack Used for the application**

**Front end**: Angular6

**Backend** : c# MVC with EF, repository pattern

**Database** : Sql Server 2014

**Product Module**

**1. Product Business class**

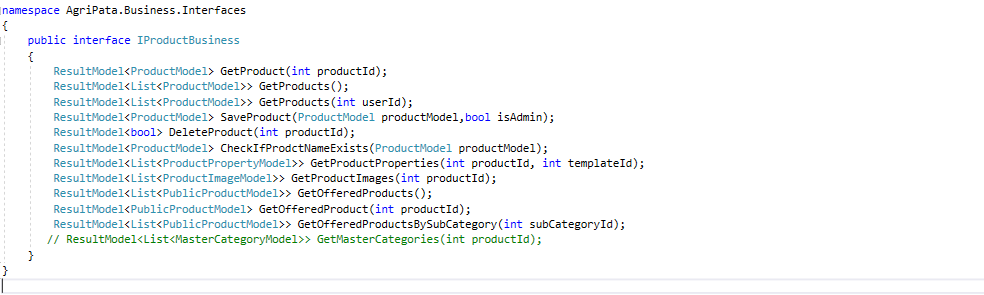


Above is the business logic for the Product module which is inherit the Interface named **IProductBusiness** and implementing the defined methods in the interface.

In the constructor of the class **(ProductBusiness)** we have inject the UnitofWork object, to access the repositories and functions for Commit the DB Changes.

**GetProduct(int ProductId) :** This function is used to get the product based on the productid , we used the Automapper to copy data from entity to Object Model then as response we will return the ProductModel

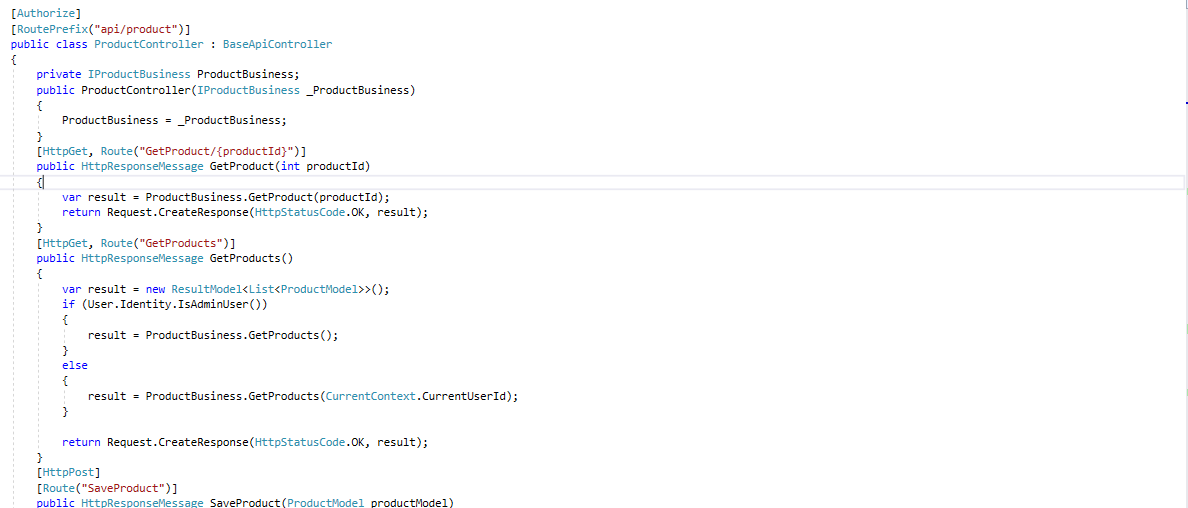
**2. Product Business Interface**



**Interface** can have methods, properties, events, and indexers as its members. But interfaces will contain only the declaration of the members. The implementation of interface’s members will be given by class who implements the interface implicitly or explicitly.

**IProductBusiness** holds the declaration of the methods which are needed to be implemented in the **ProductBusiness**

**3. Product API Controller**



Product API controller is used to receive the call from the end **Authorized User**, we also used the Attribute routing here for making the URL's User Friendly

The API controller also have a constructor in which the interface **IProductBusiness** is injected in order to use it to call the business methods

**4. Product Component in Angular6**



On the front end Angular6 is used .

Angular 6 is a JavaScript framework for building web applications and apps in JavaScript, html, and TypeScript, which is a superset of JavaScript. Angular provides built-in features for animation, http service, and materials which in turn has features such as auto-complete, navigation, toolbar, menus, etc

Here Datatable Js Library is Used to display the array of data on UI

**ngOnInit** it runs when the component is initialized

in this function default values are set for the DtOption object, and called getProduct function to get the list of all the products in order to display on the UI

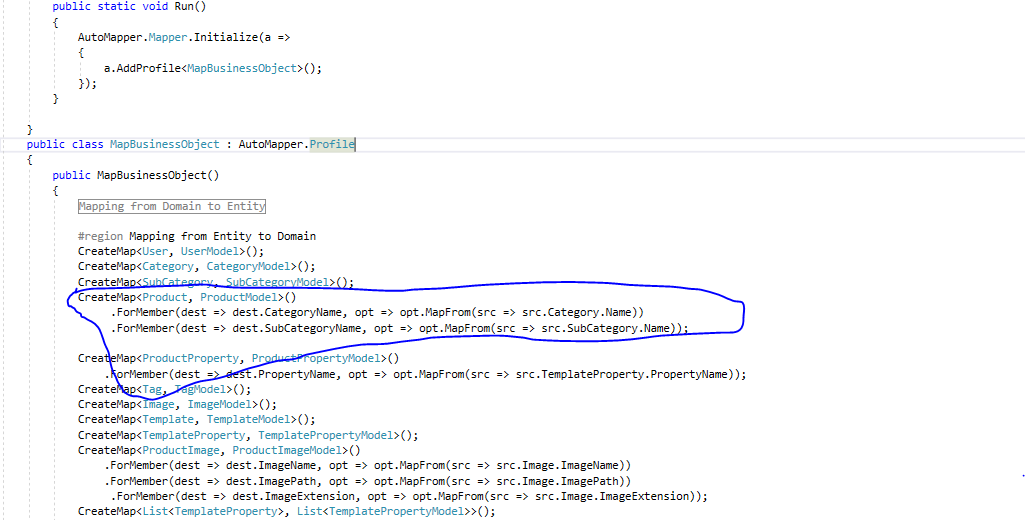
**Constructor** : Different types of services are injected in the constructor, it works similar to the server side class constructor

**5.Product Client Side Service**



Above is the Service which is used to interact with the API to get and post the data.

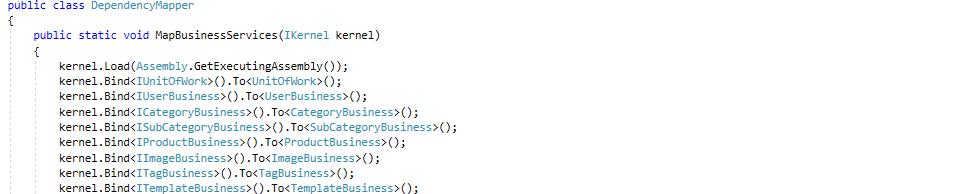
**6. Mapper Config**



AutoMapper is a simple little library built to solve a deceptively complex problem - getting rid of code that mapped one object to another.

As marked area above is the mapping for the **product (Entity)** into **productModel (model Object)**.

**7.Dependency Mapper**



Ninject helps you use the technique of dependency injection to break your applications into loosely-coupled, highly-cohesive components, and then glue them back together in a flexible manner.