Introduction

I create projects of CRUD (Create, Read, Update, and Delete) operations in ASP.NET Core MVC (.Net 6.0) with code first migration. We will use Entity Framework Core 6 to interact with the SQL Server database. I use here two Entity for CRUD operation (Category and Product).

**Cover the following points.**

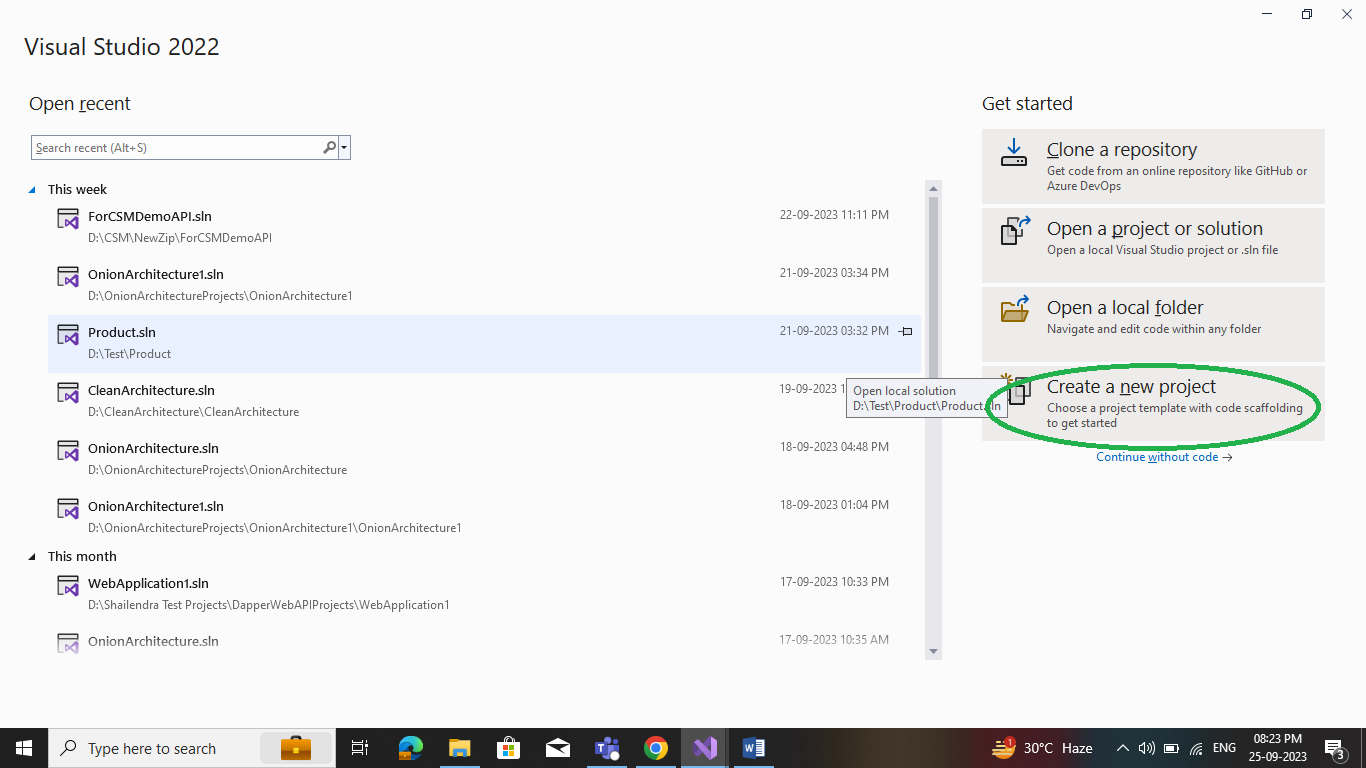
* Create ASP.NET Core MVC 6.0 project in visual studio 2022
* Install all the necessary packages from NuGet.
* Create entities in the Model folder.
* Create BaseEntitie in the Model folder.
* Crete ApplicationDbContext class to interact with Database in Data folder
* Add connection string of SQL Server database into appsettings.json file
* Run migration commands inside Package Manager Console (PMC)
  + Add-Migration "Initial"
  + Update-Database
* Add controllers inside the controller folder
* Add view models in Model folder
* Add DataContext in Data folder
* Add Repository and IRepository in in Repository folder
* Add Views and partial views
* Add Validation
* Updated Bootstrap and Added bootstrap Js and css file online link

**Prerequisites**

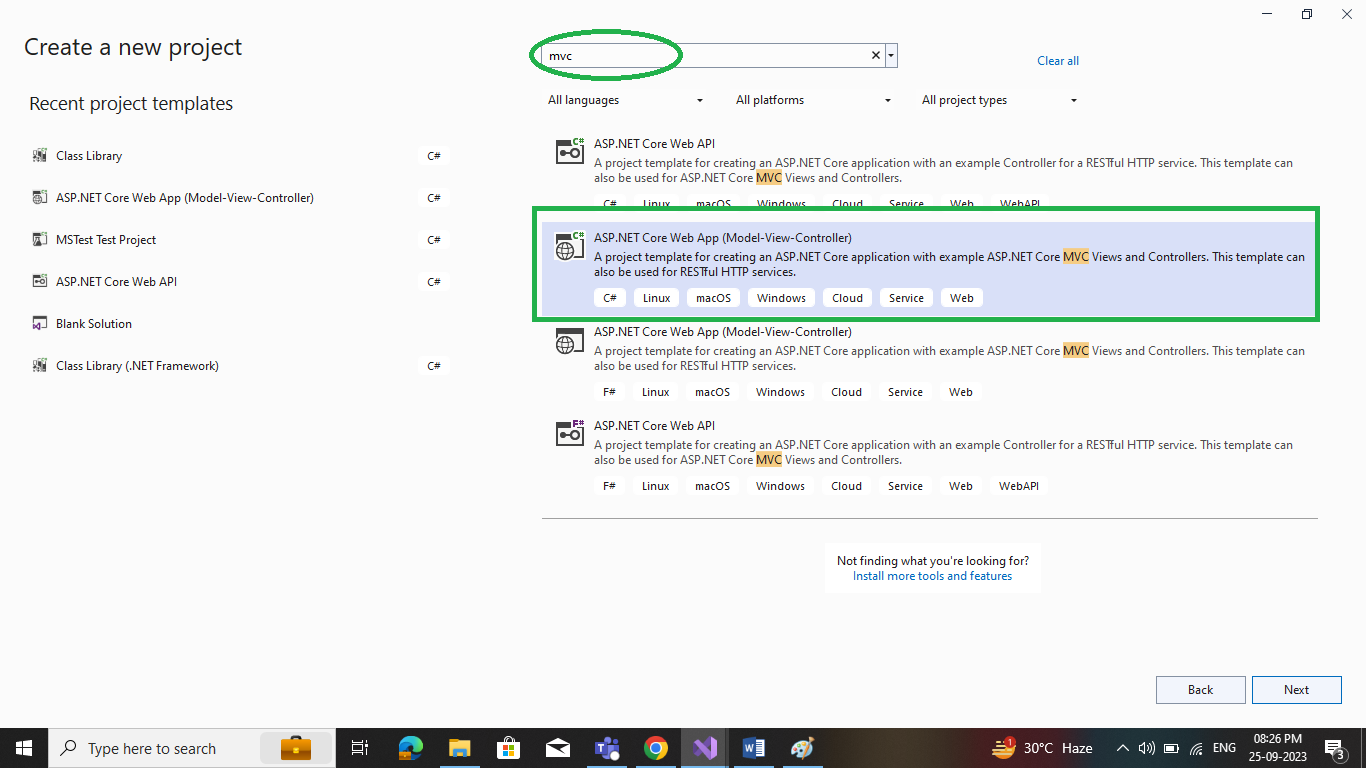
* Visual Studio 2022 (.Net 6.0)
* SQL Server

**Step 1**. Create ASP.NET Core MVC 6 project

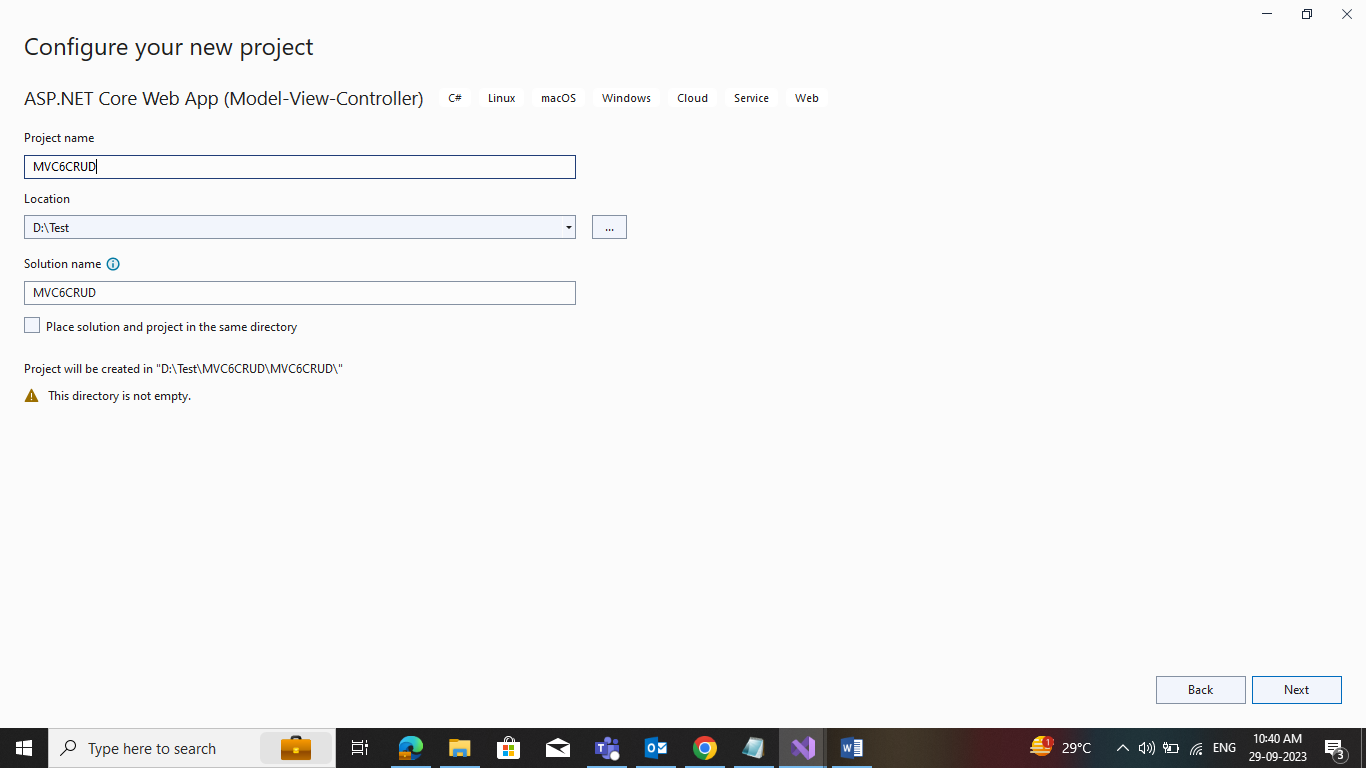
First, open Visual Studio 2022 and click on**"Create a New Project" (**highlighted in green colour circle**)**



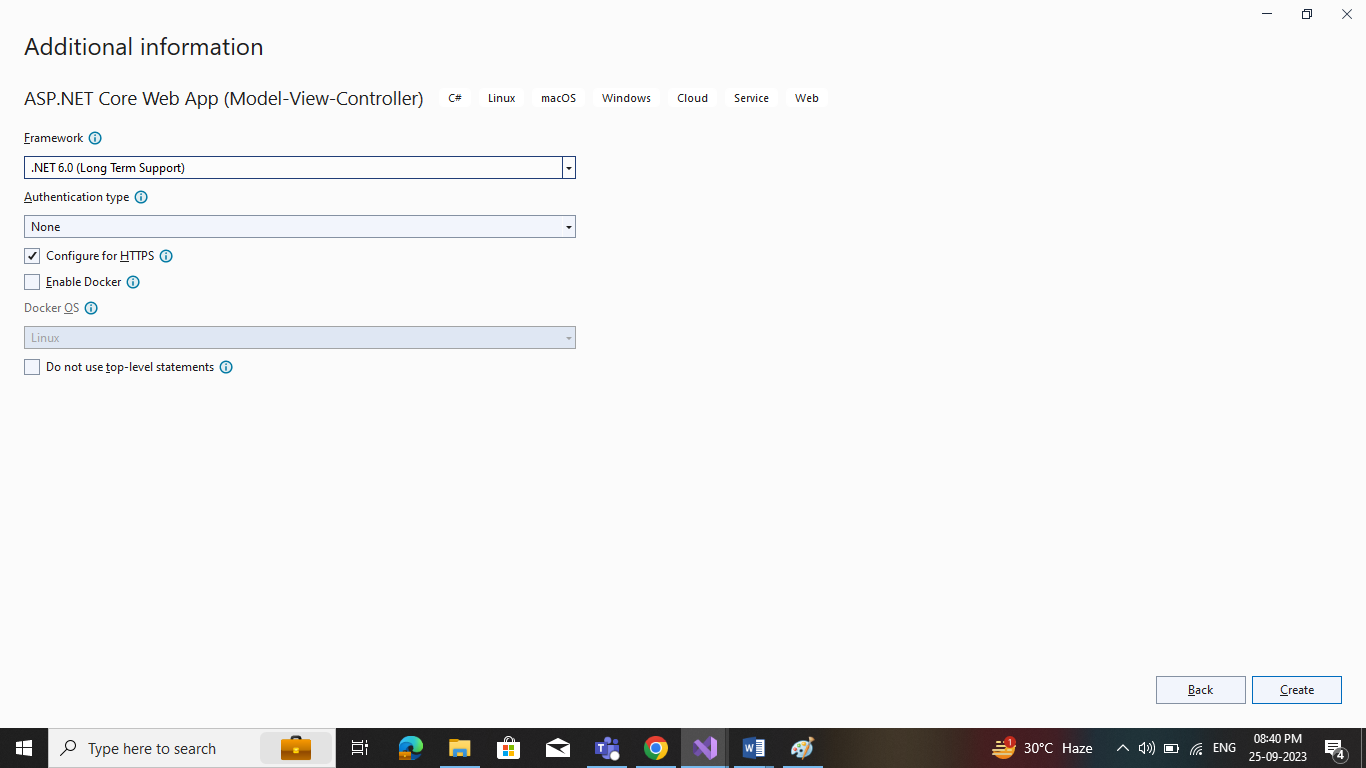
Search "**MVC**" in the Search box**(Search for templates (Alt+s))**and select**ASP.NET Core Web App (Model-View-Controller)**as a project template and click**the Next**button**.**



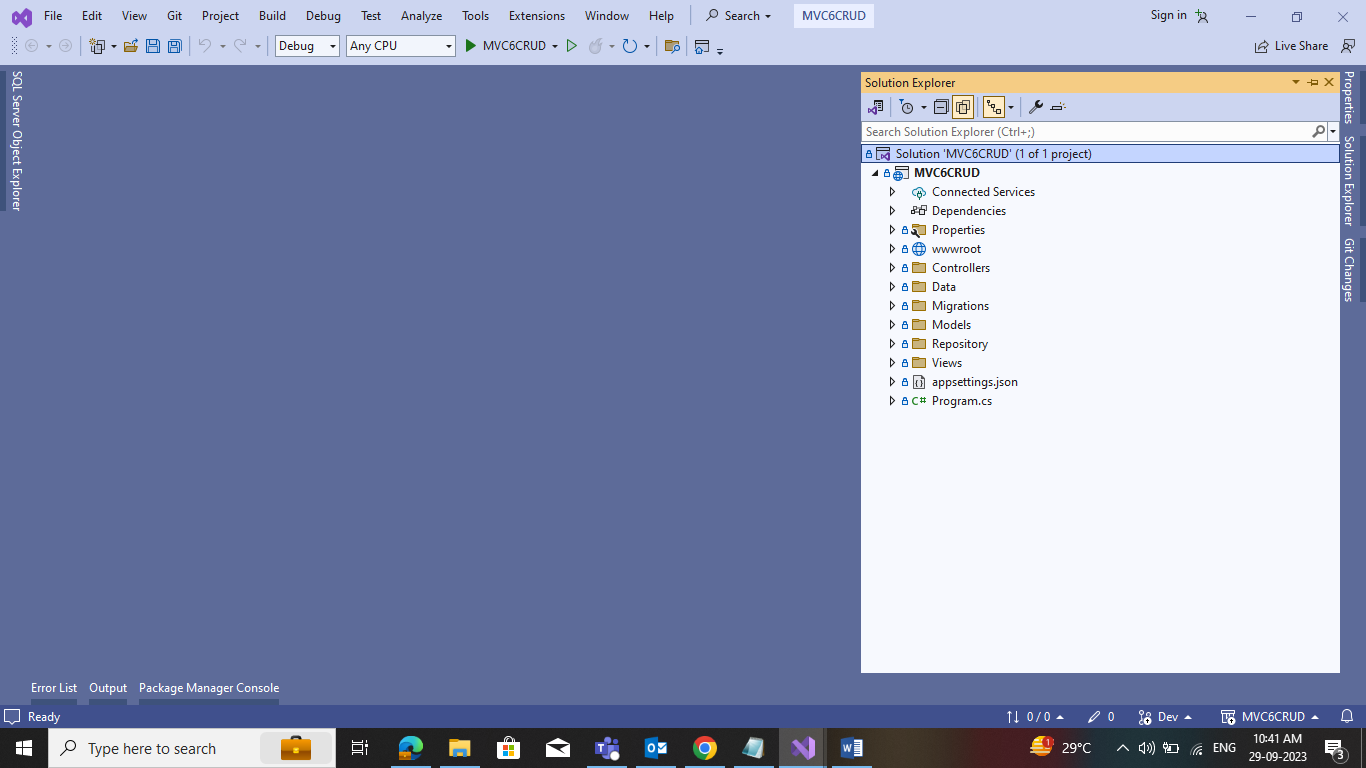
Enter a project name and click next.



Go with the basic configuration as per the below screen; click**Next.**



**ASP.NET Core MVC (.Net 6.0) project structure**



**Step 2**. Install the required package from NuGet

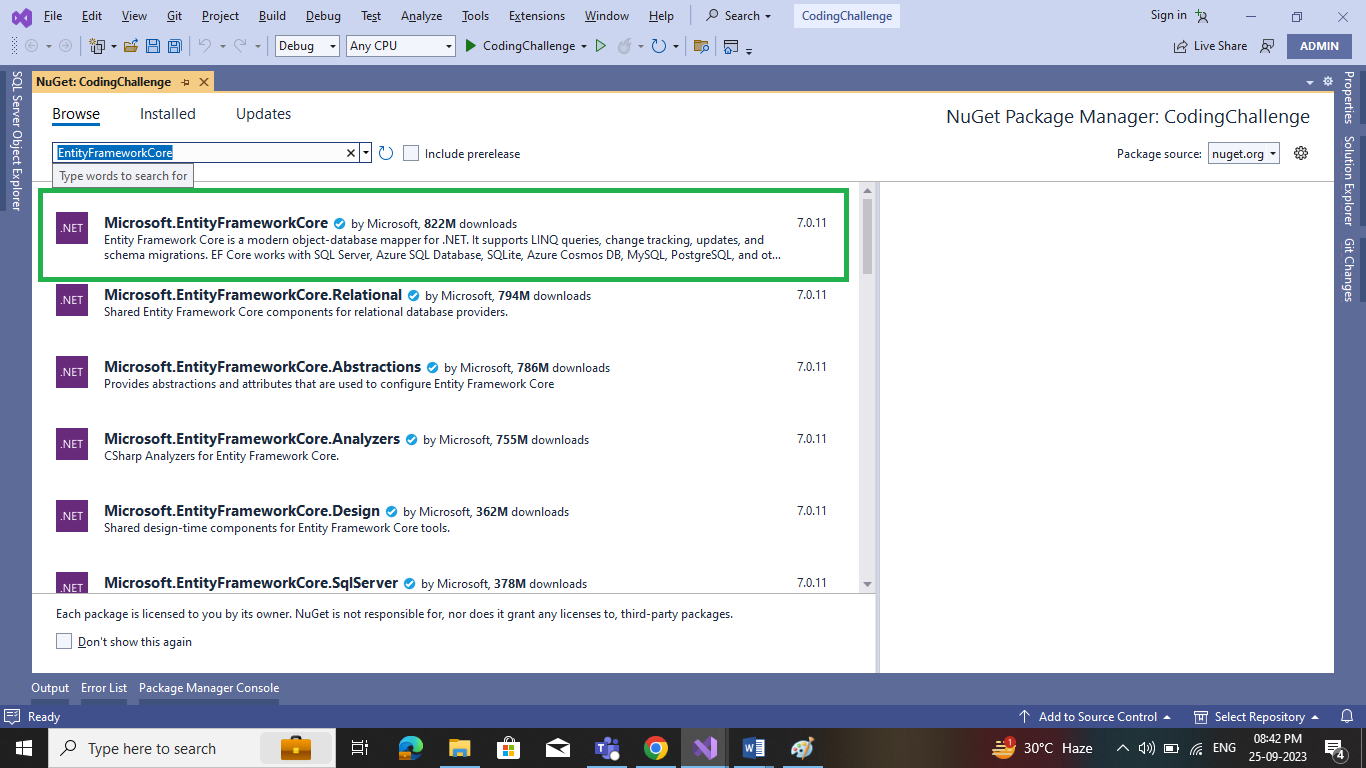
Required package for ASP.NET Core MVC (.Net 6.0) CRUD operation application

* Microsoft.EntityFrameworkCore
* Microsoft.EntityFrameworkCore.SqlServer
* Microsoft.EntityFrameworkCore.Tools

Now, we install the above package one by one**.**Right-click on**Dependencies**and select**"Manage NuGet Packages…"**

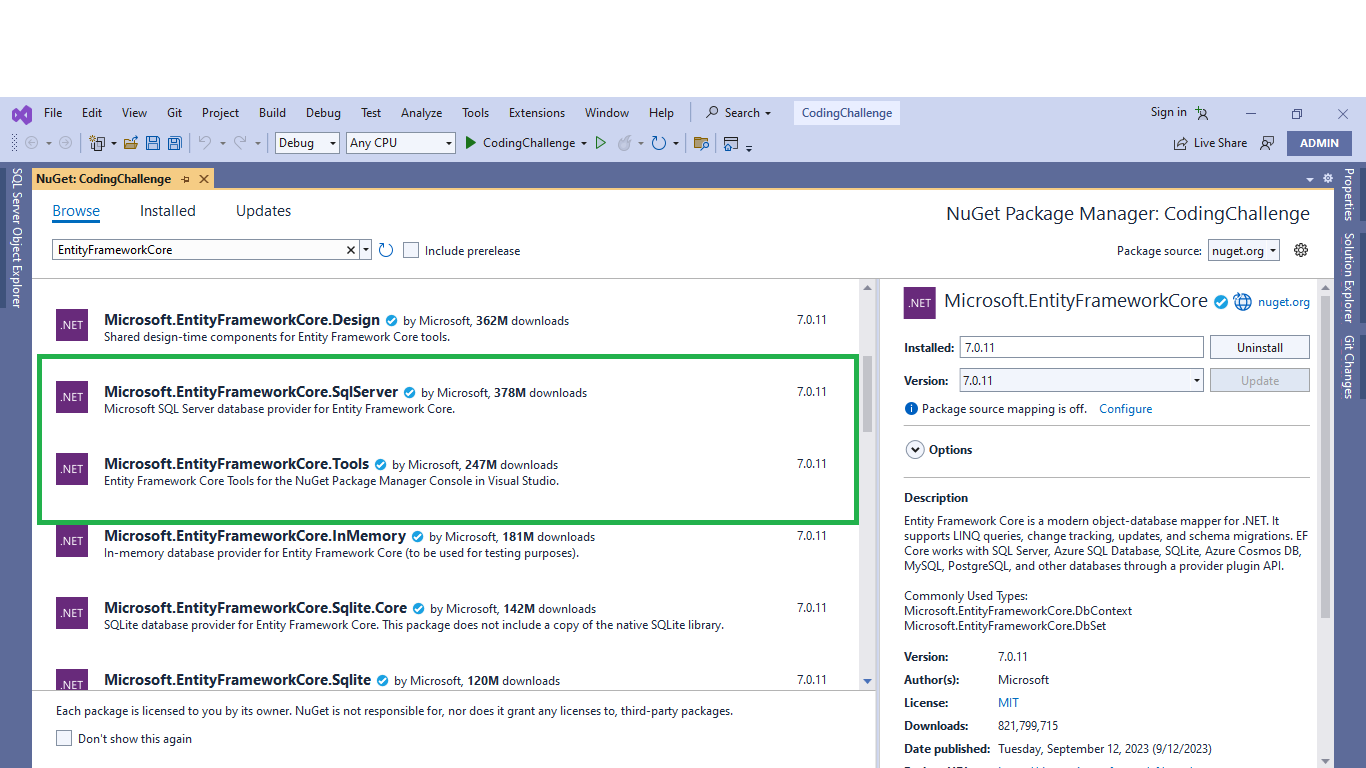


Install **Microsoft.EntityFrameworkCore**package**.**

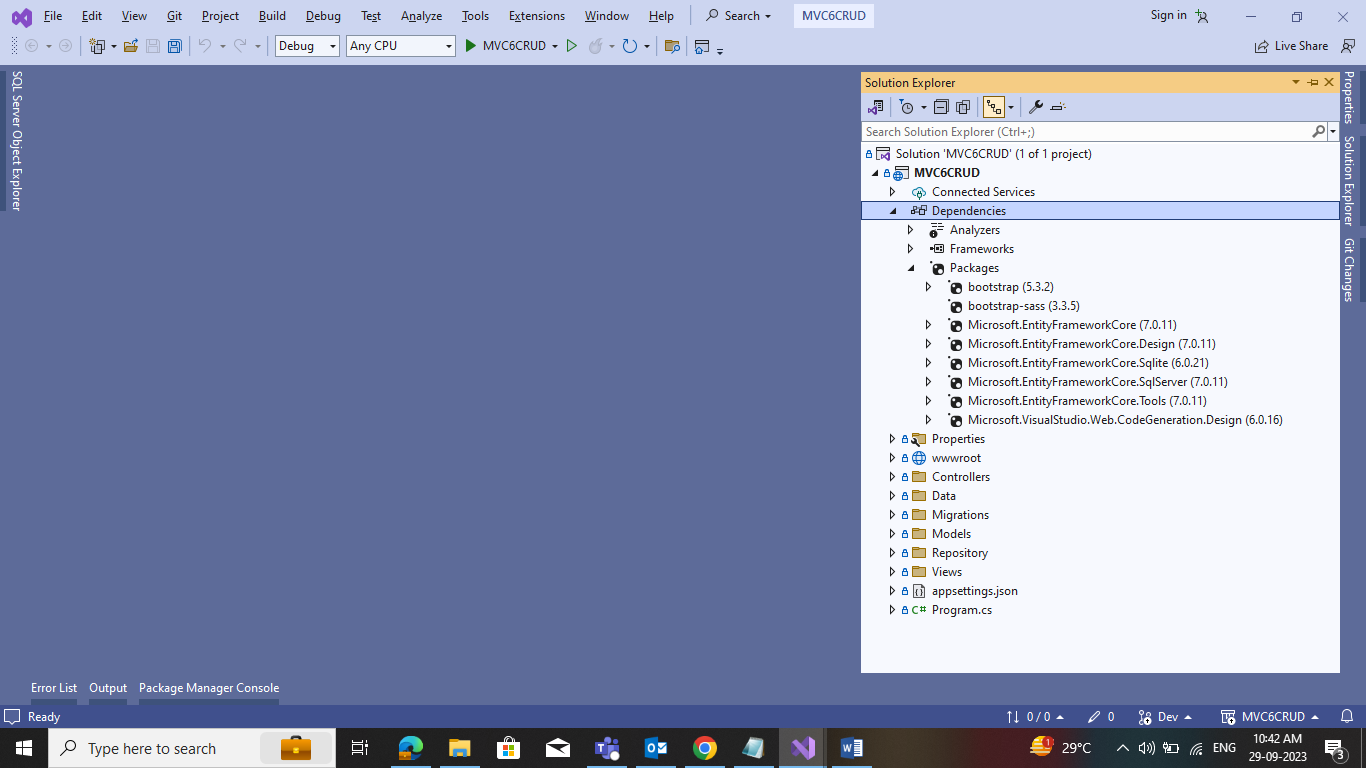


Install **Microsoft.EntityFrameworkCore.SqlServer**package**.**

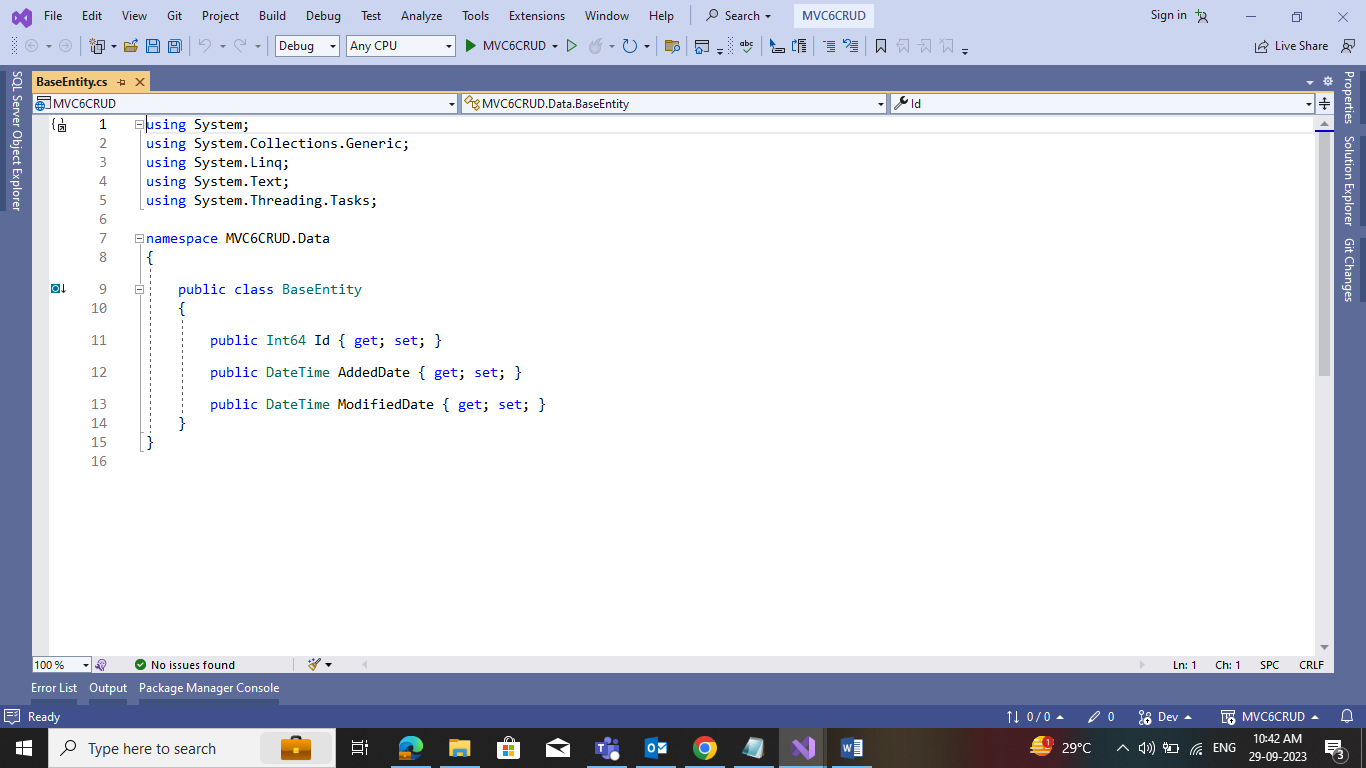
Install **Microsoft.EntityFrameworkCore.Tools**package**.**



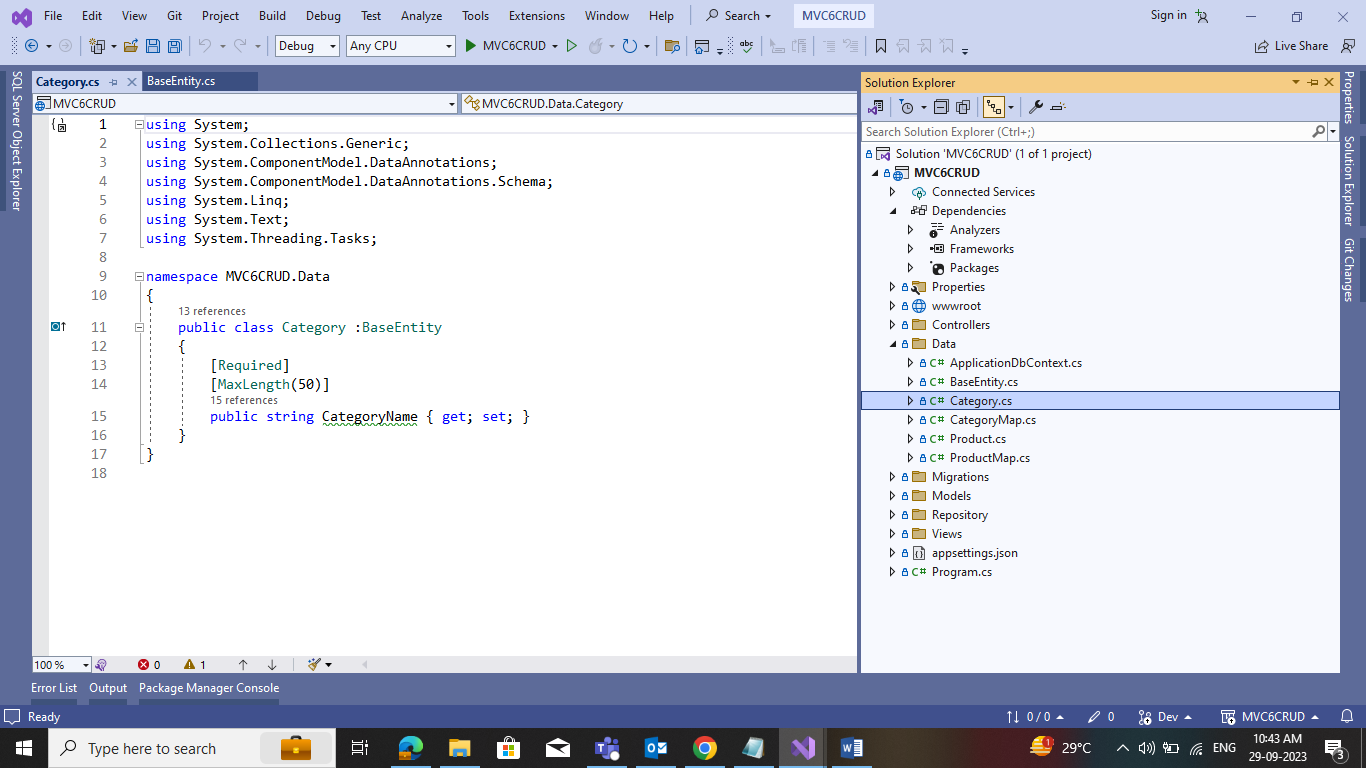
All install packages are showing in **Dependencies -> Packages**

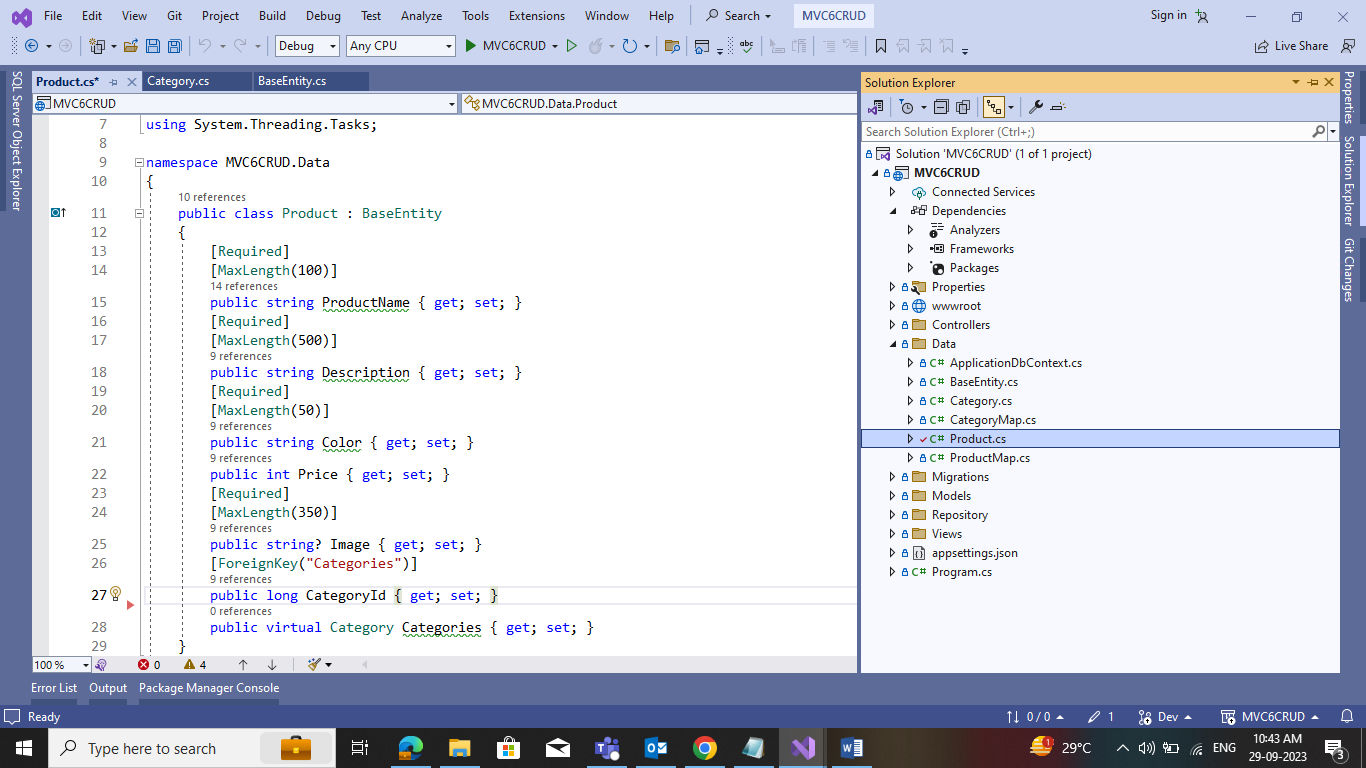


**Step 3**. Add **BaseEntity** entities class into the **Model** folder

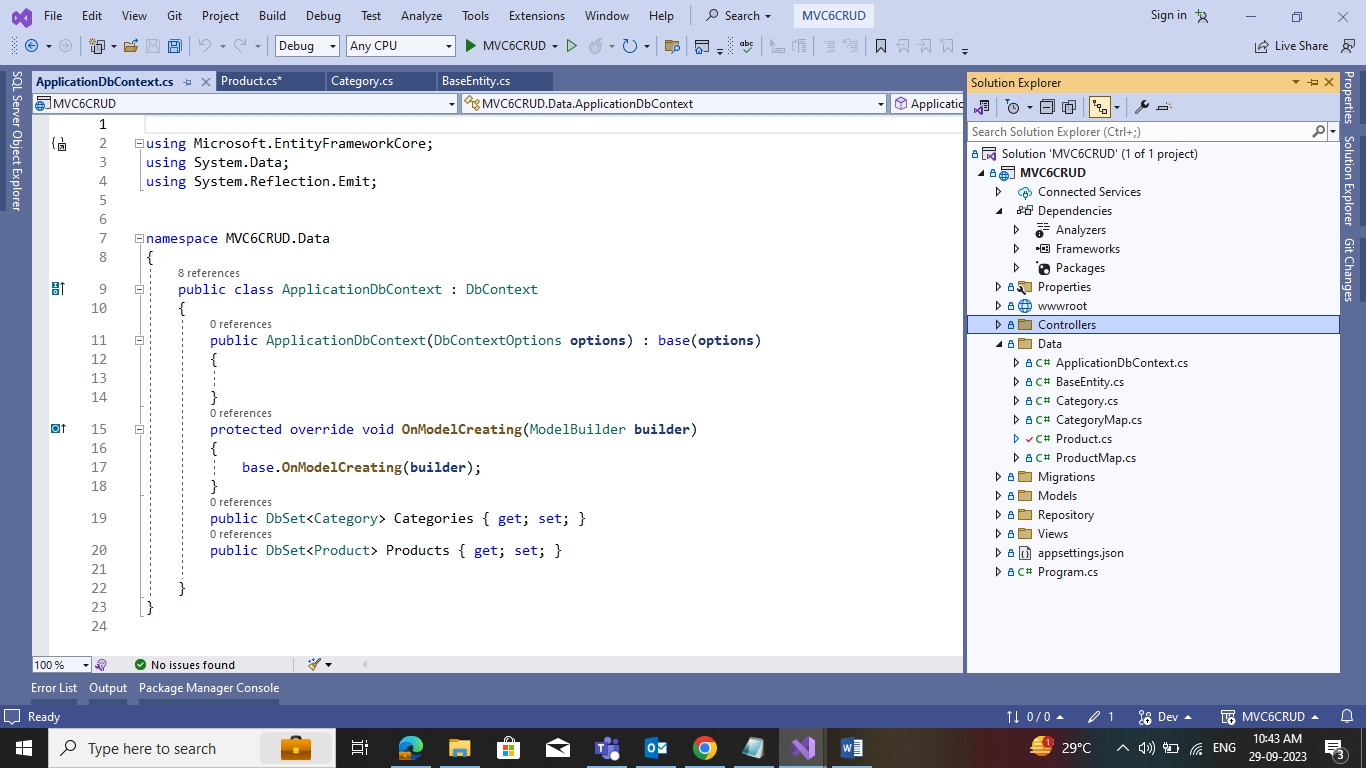


**Step 4**. Add **Category** and **Product** entities class into the **Model** folder and inherit from BaseEntity





**Step 5.**Add **ApplicationDbContext**class inside the **Data** folder and inherit it to DbConext

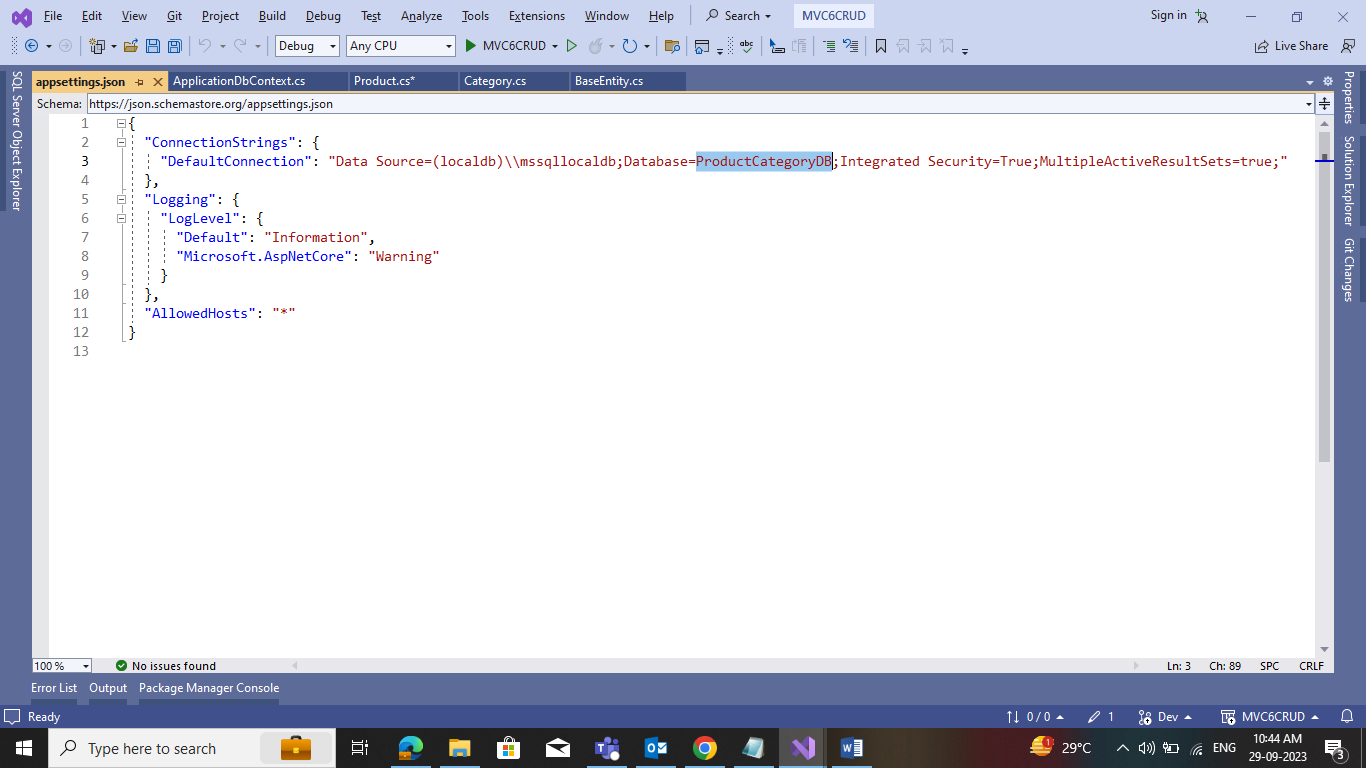


**Step 6.**Add **Connectionstring** into **appsettings.json**

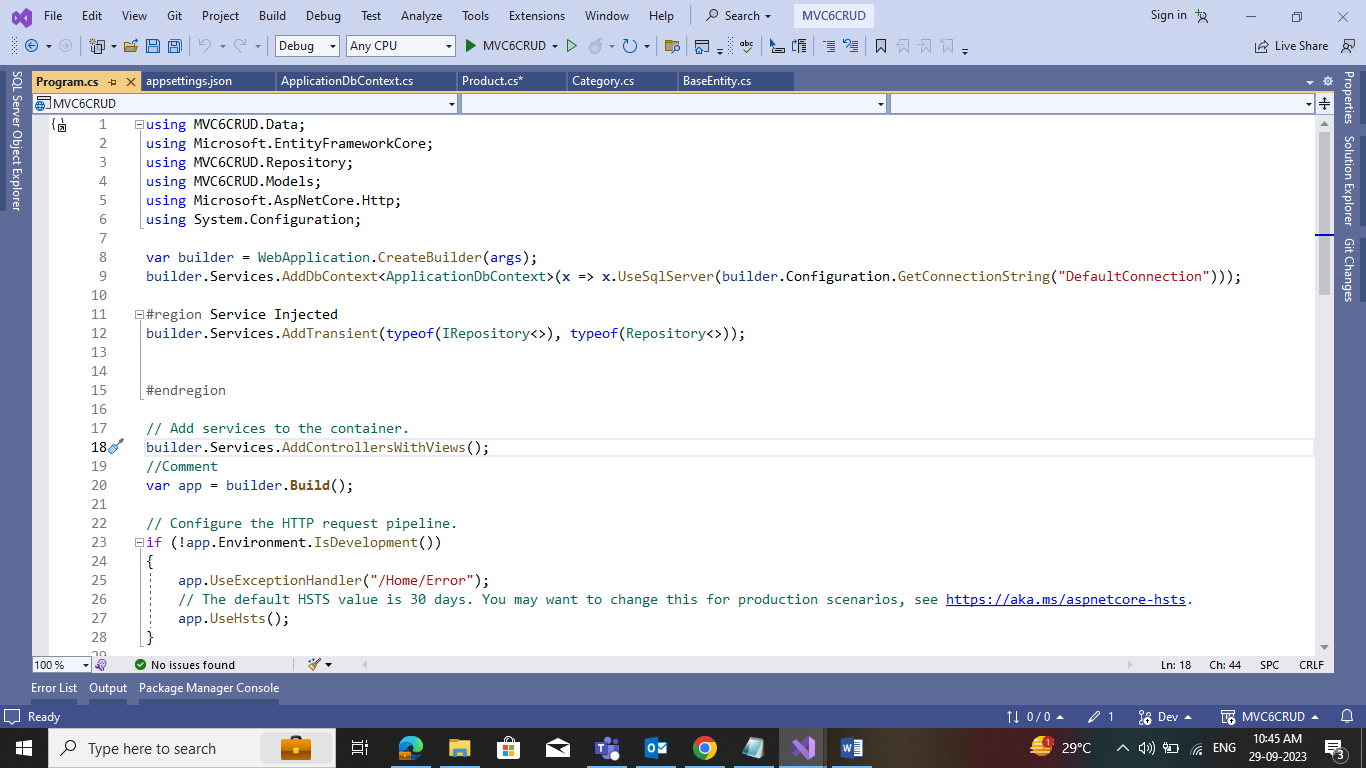
"ConnectionStrings": {

"CodingChallengeConnection": "Data Source=(localdb)\\mssqllocaldb;Database= ProductCategoryDB;Integrated Security=True;MultipleActiveResultSets=true;"

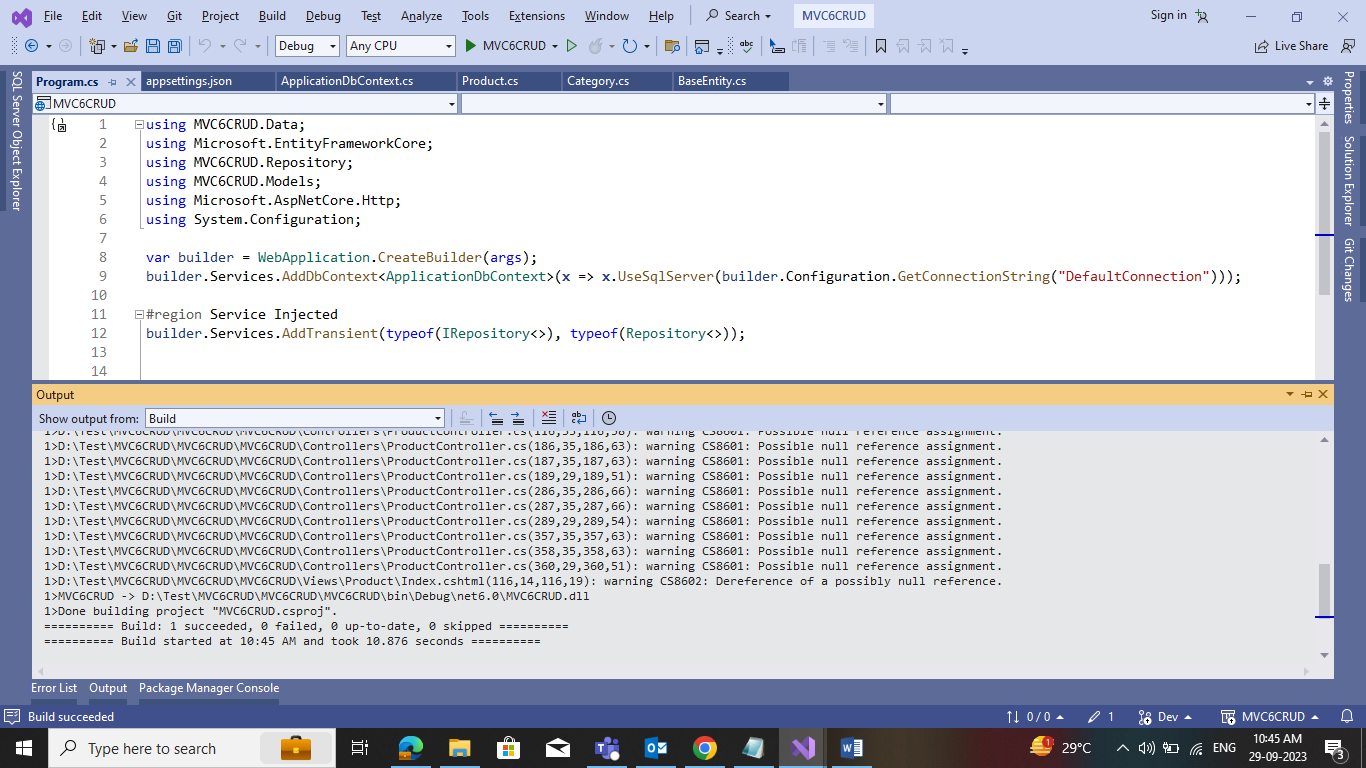
}



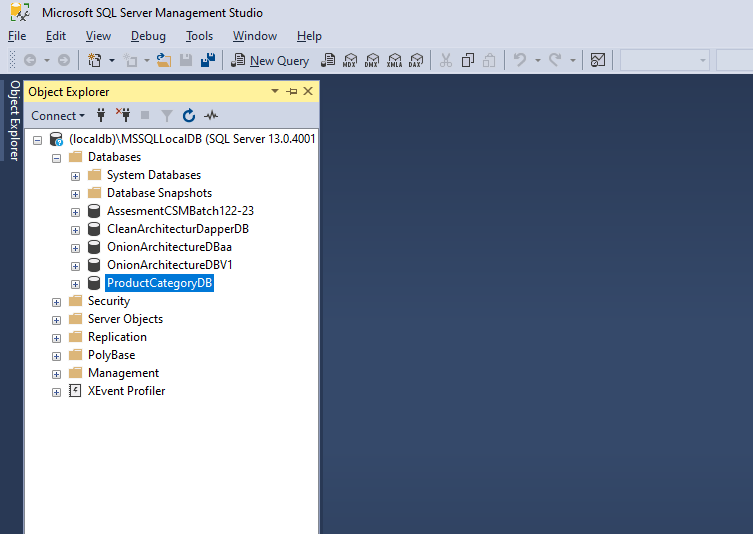
**Step 7.** Add DbContext Service into Program.cs file with the connection string



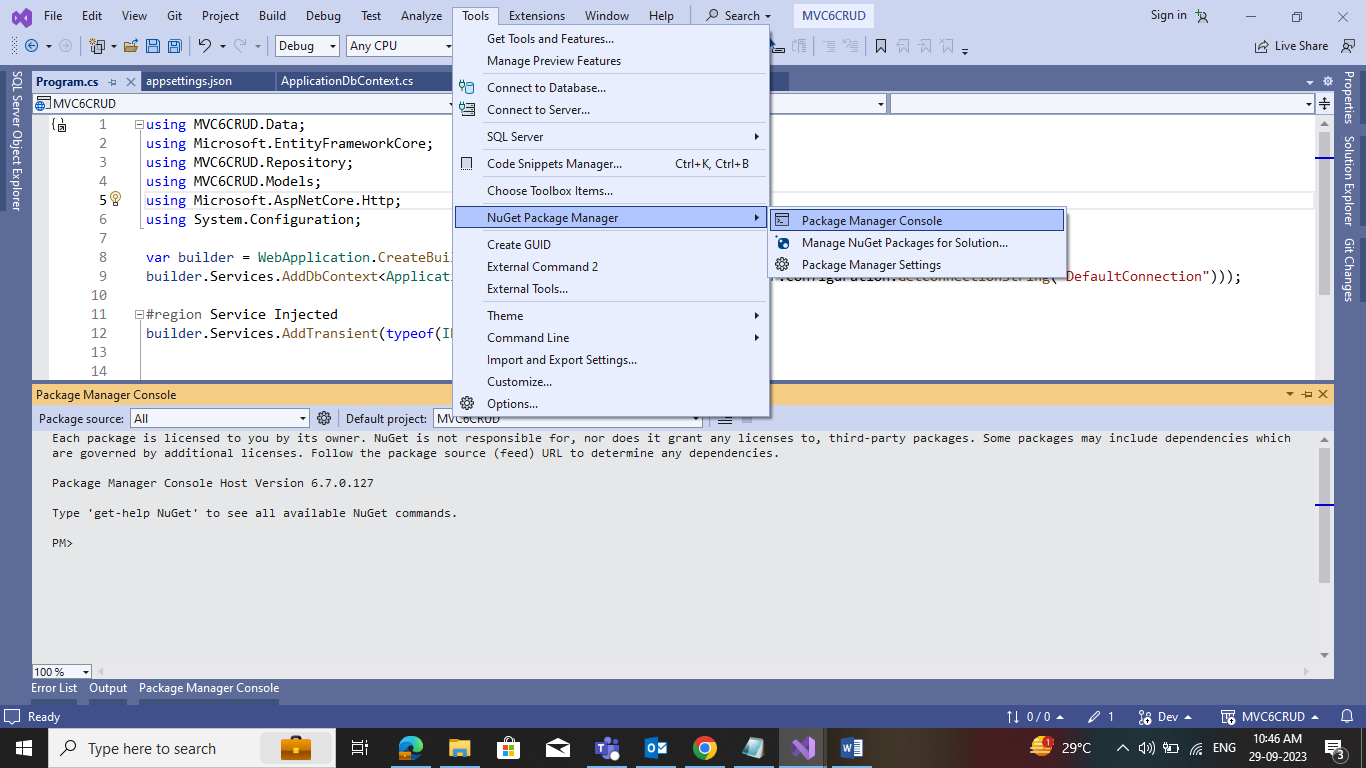
**Build solution**



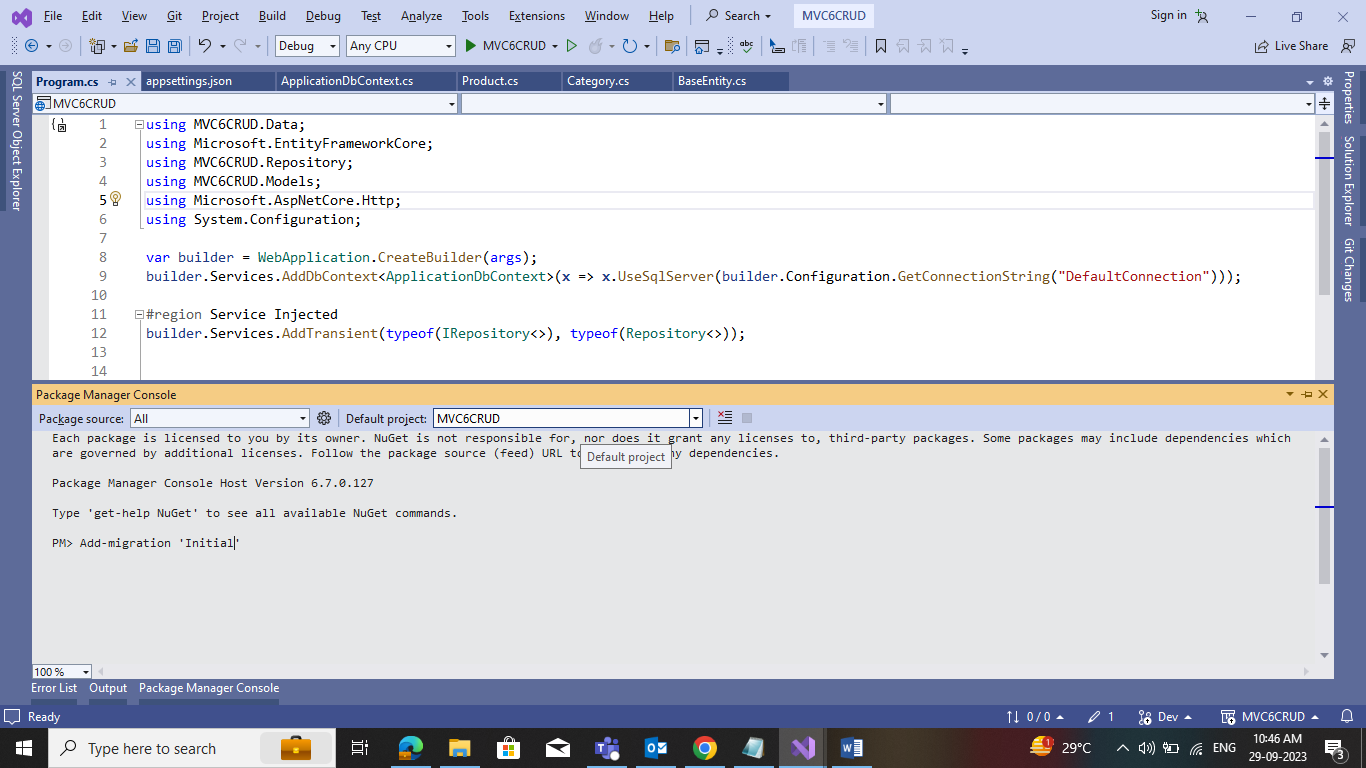
**Step 8.** Run migration commands inside Package Manager Console (PMC) to create a Database



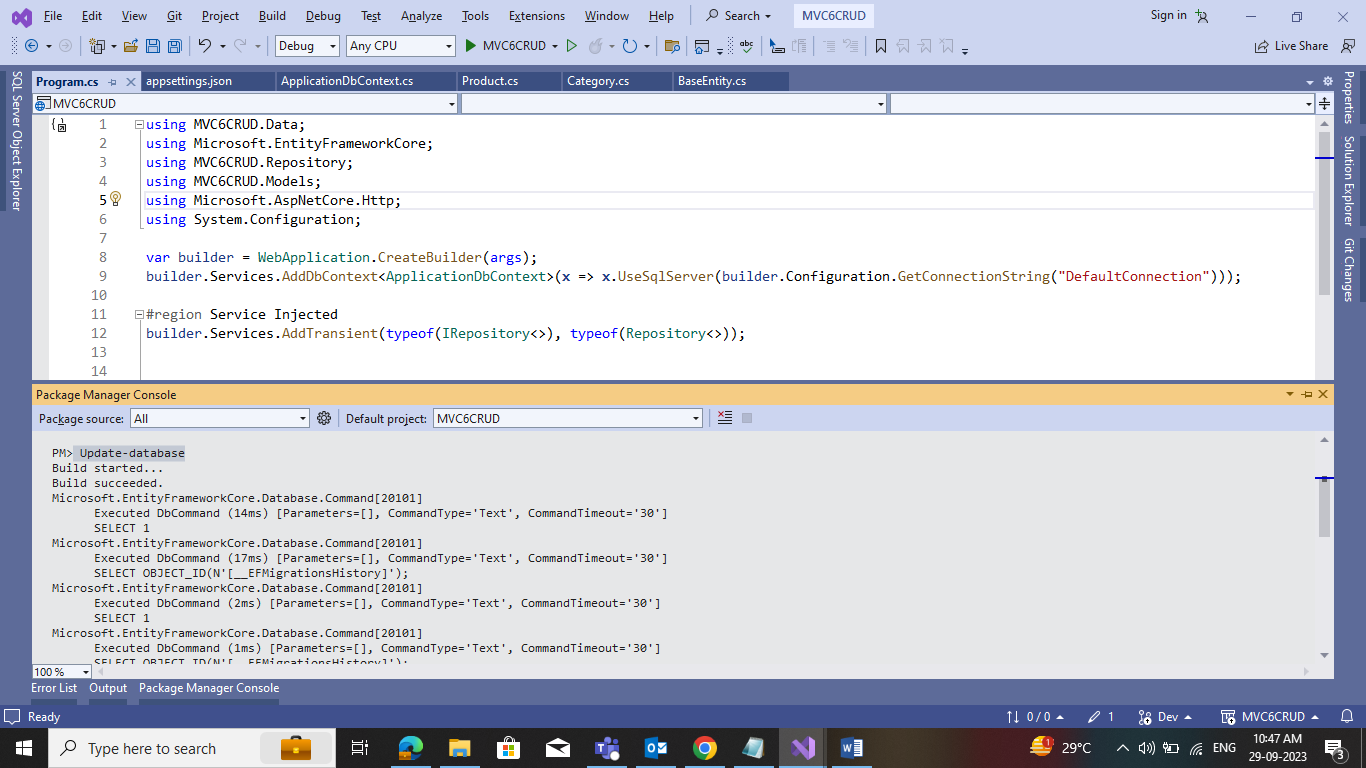
**Package Manager Console (PMC)**



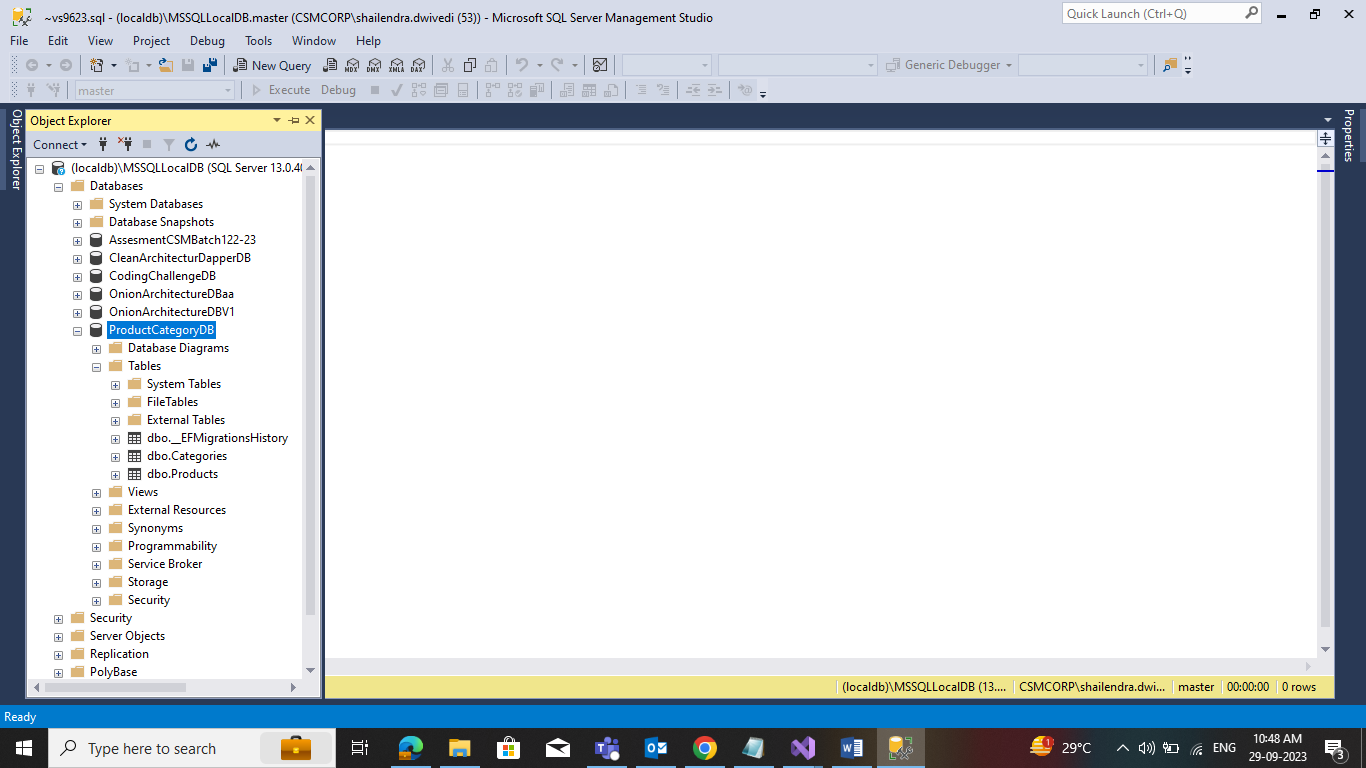
**Add-Migration "Initial" command**



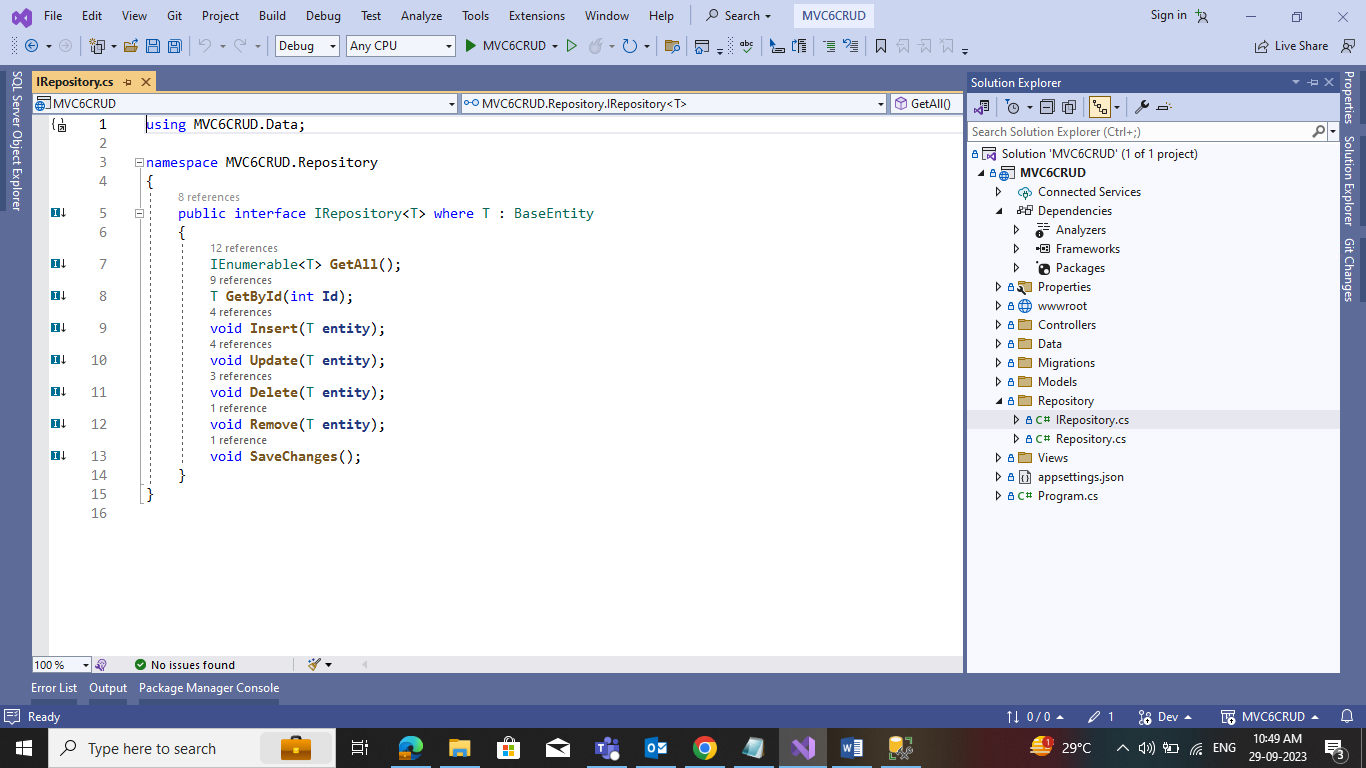
Update-Database command

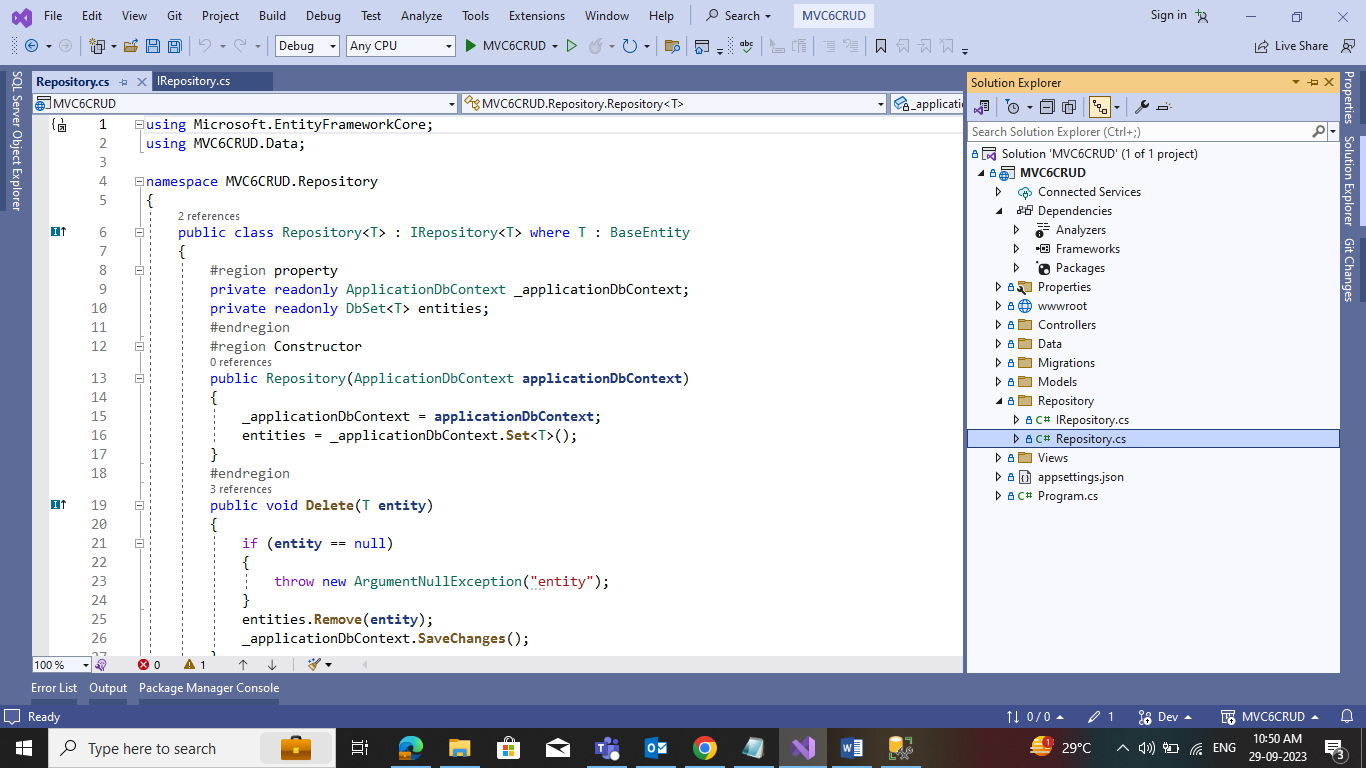


ProductCategoryDB created into SQL Server with Categories and Products table.

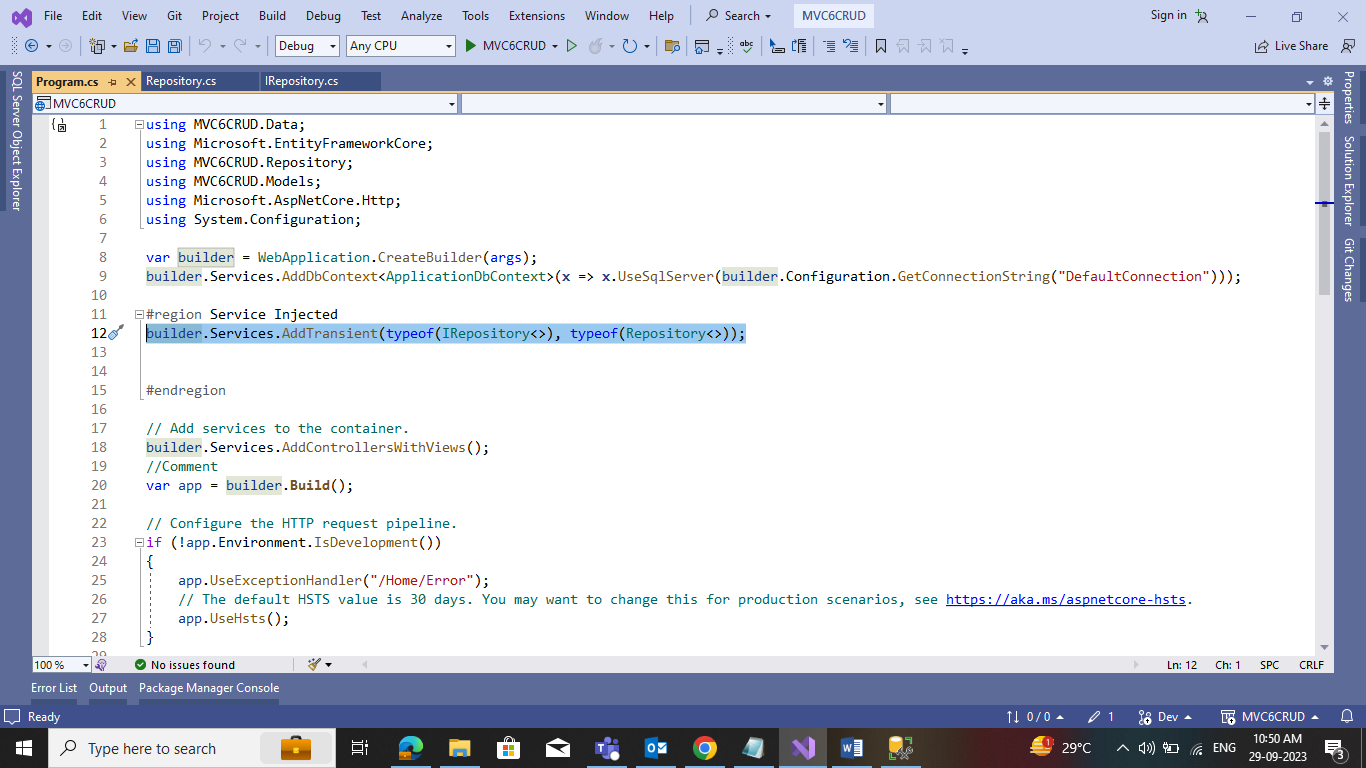


**Step 9**. Add Repository and IRepository in Repository folder

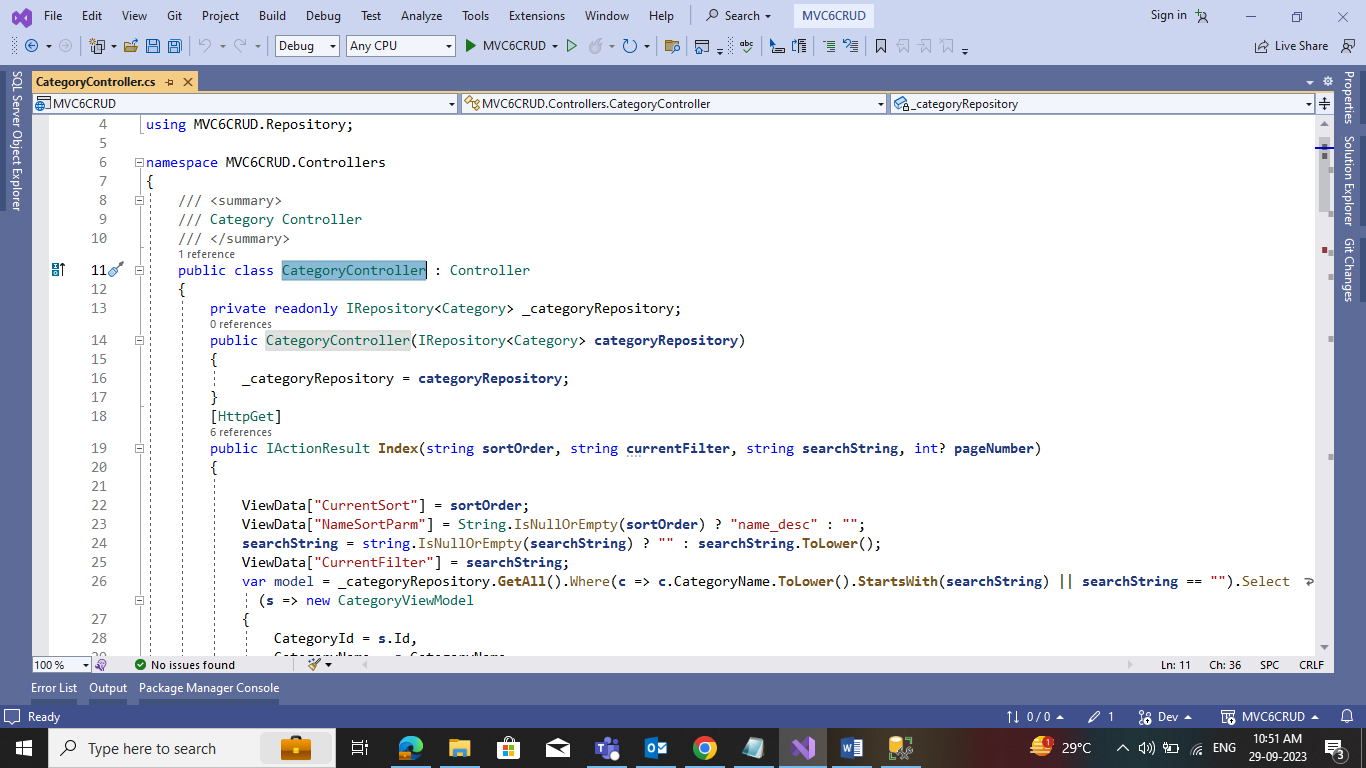


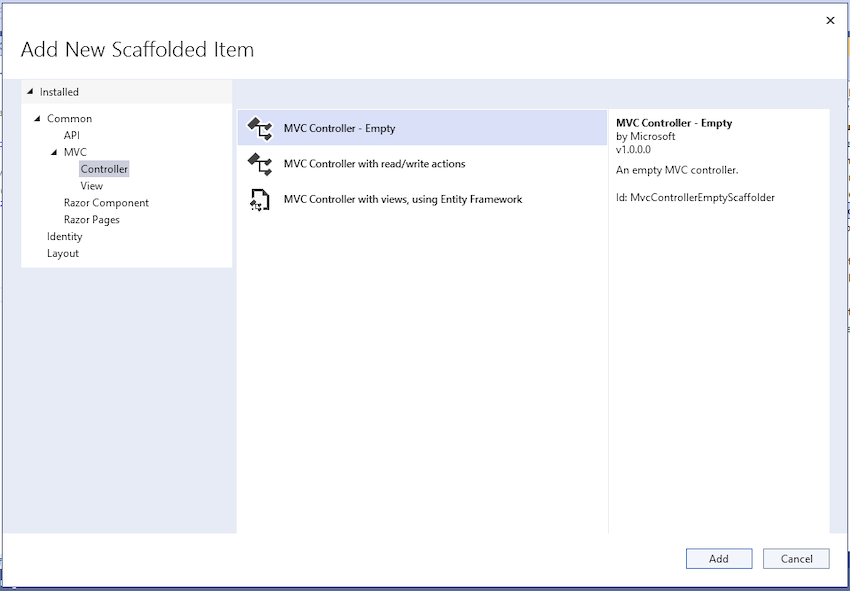


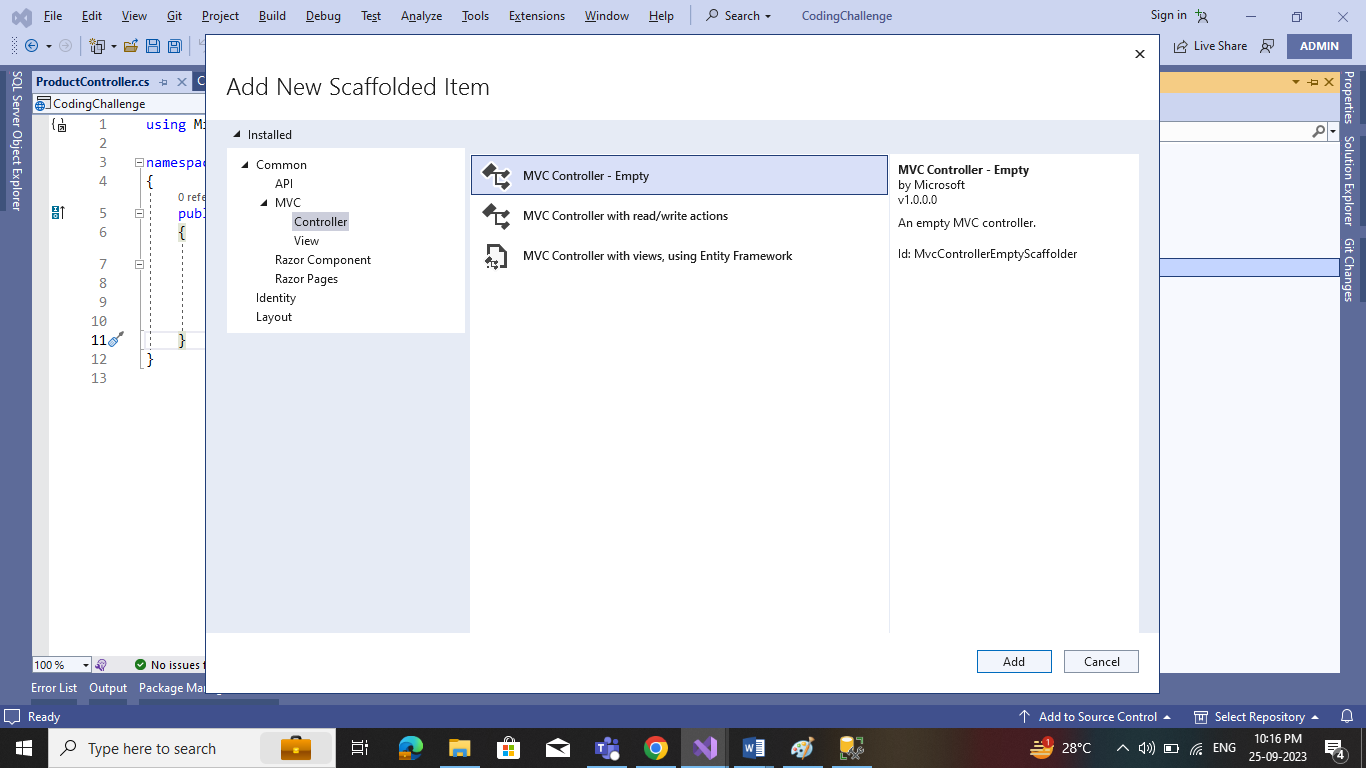
After Repository created the add service references in Program.cs



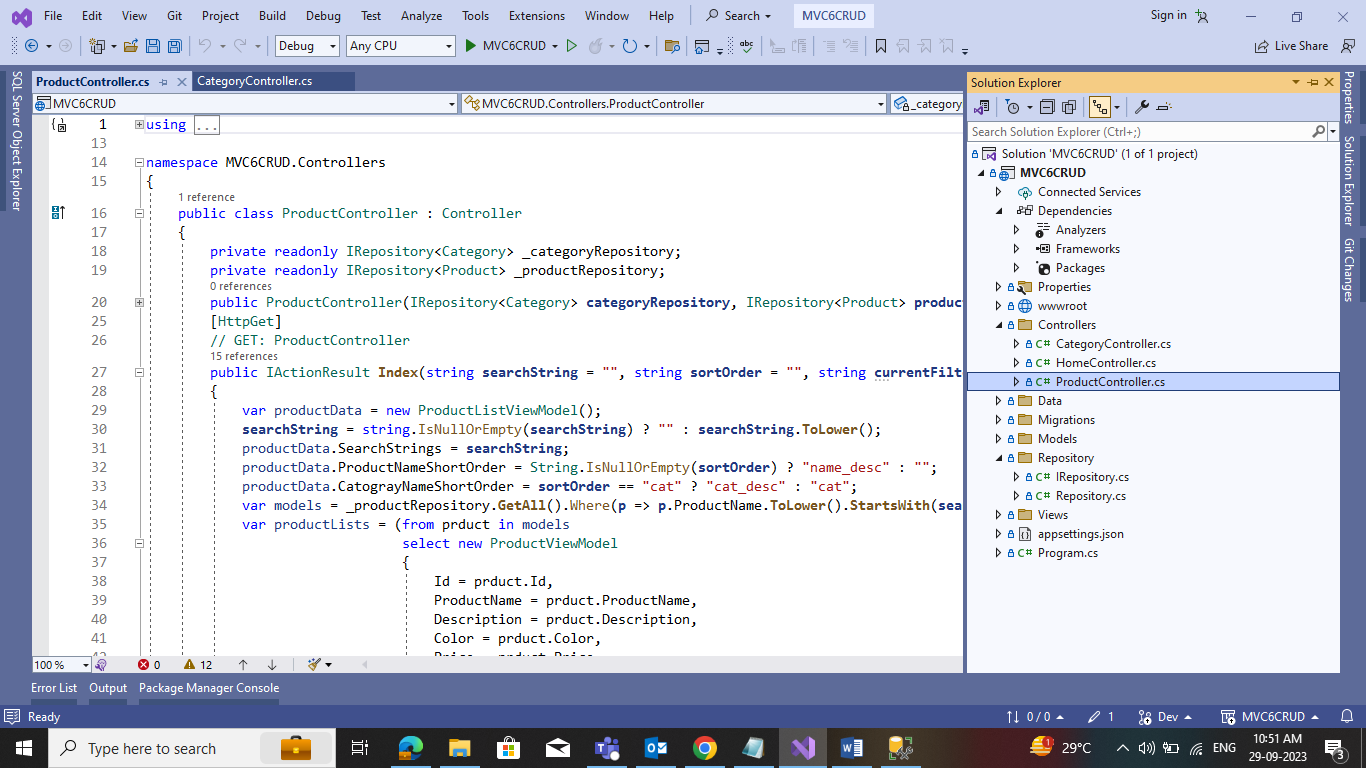
**Step 10.**Add the Category controller and Product controller in the controller folder



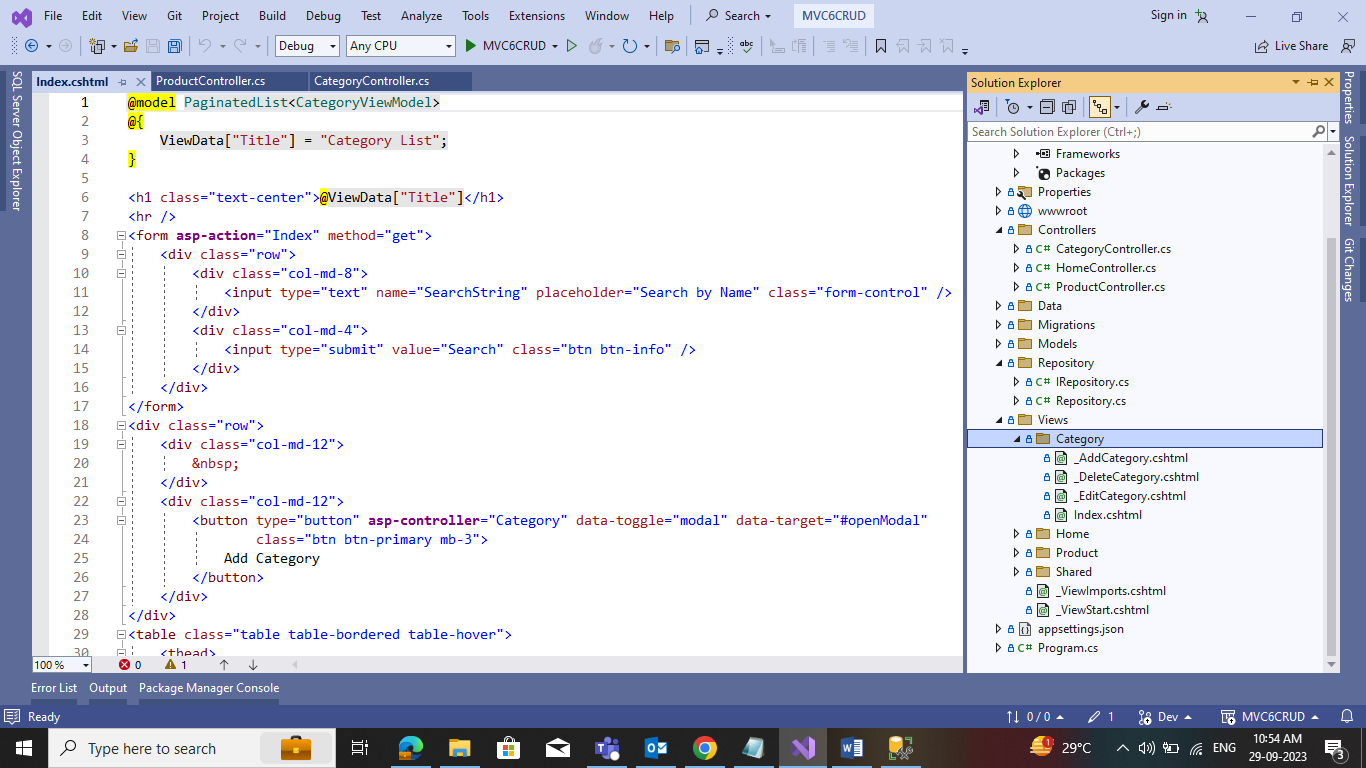


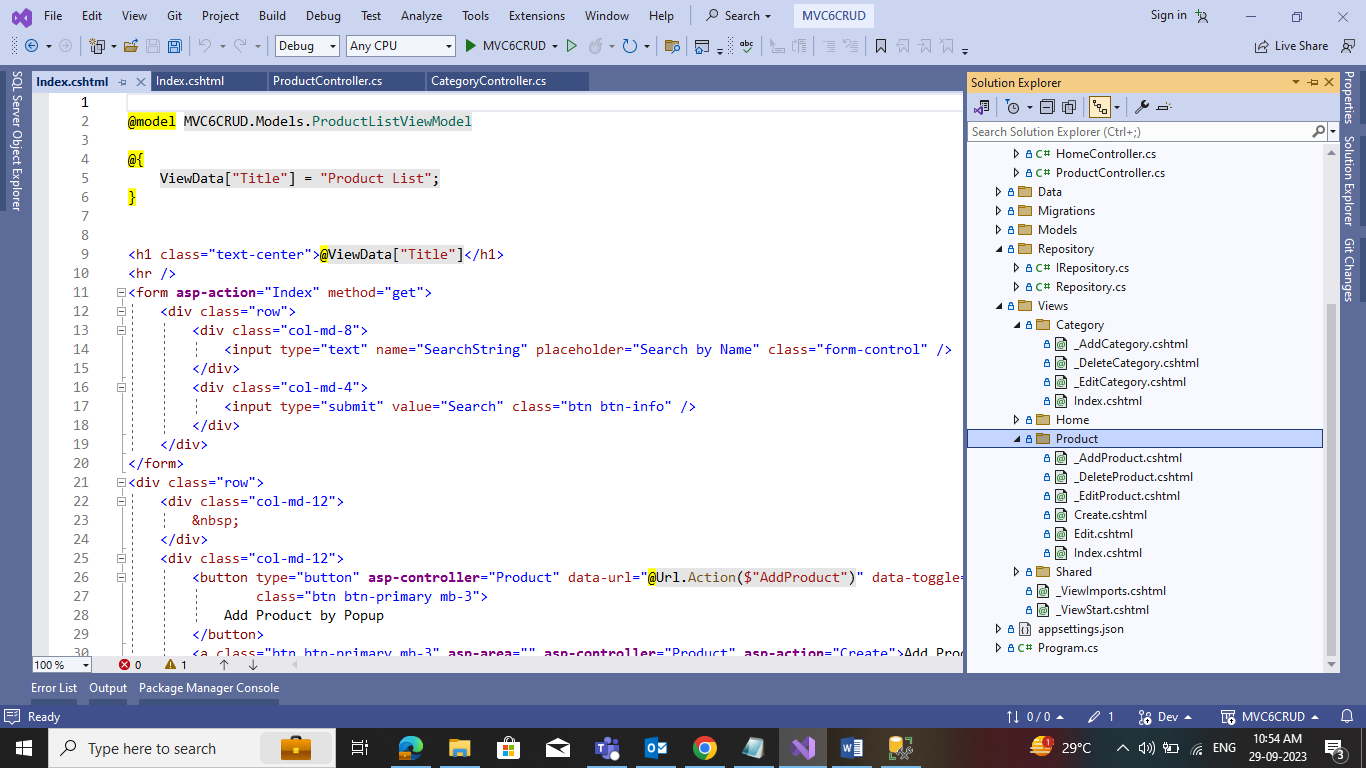


In the same way, we added a Product controller, and now both are showing in the controller folder.



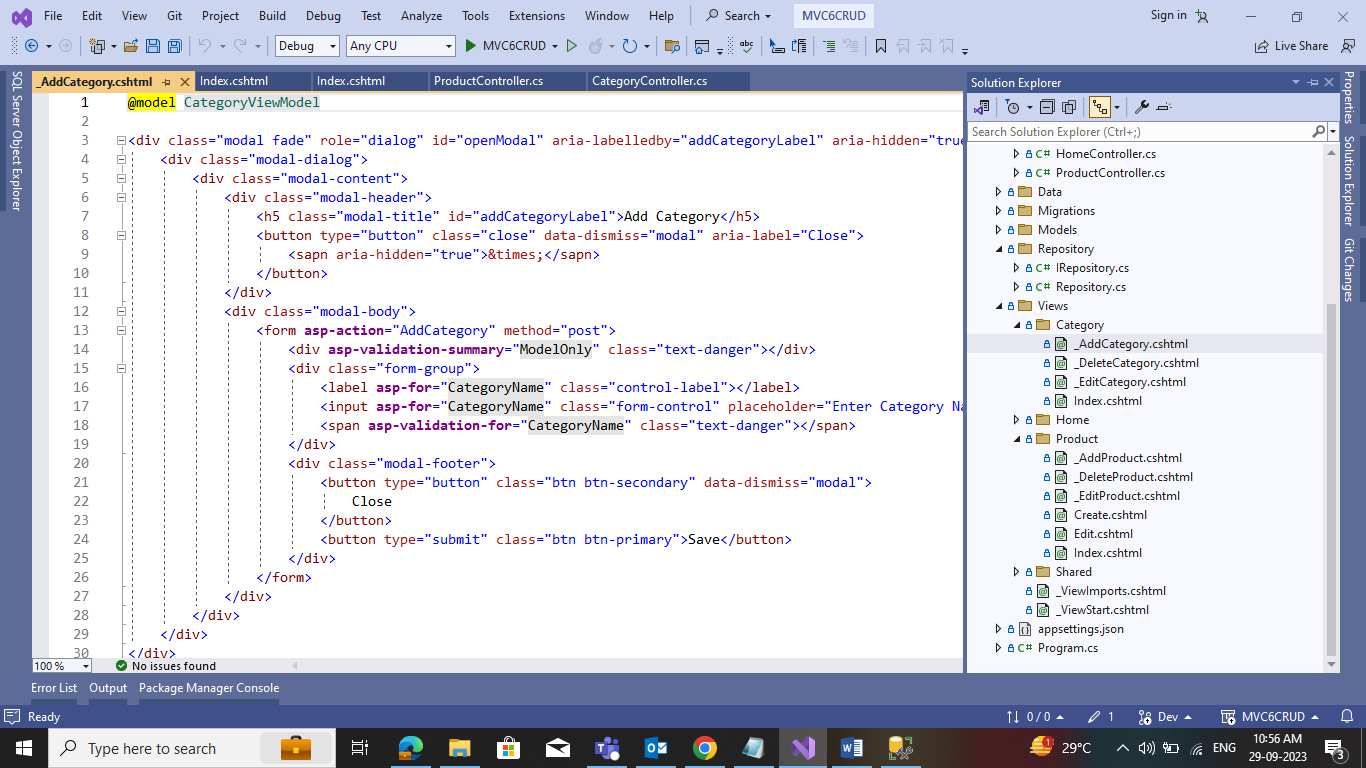
**Step 11.**Add the Category View default(Index) and Product View default(Index) in the View folder

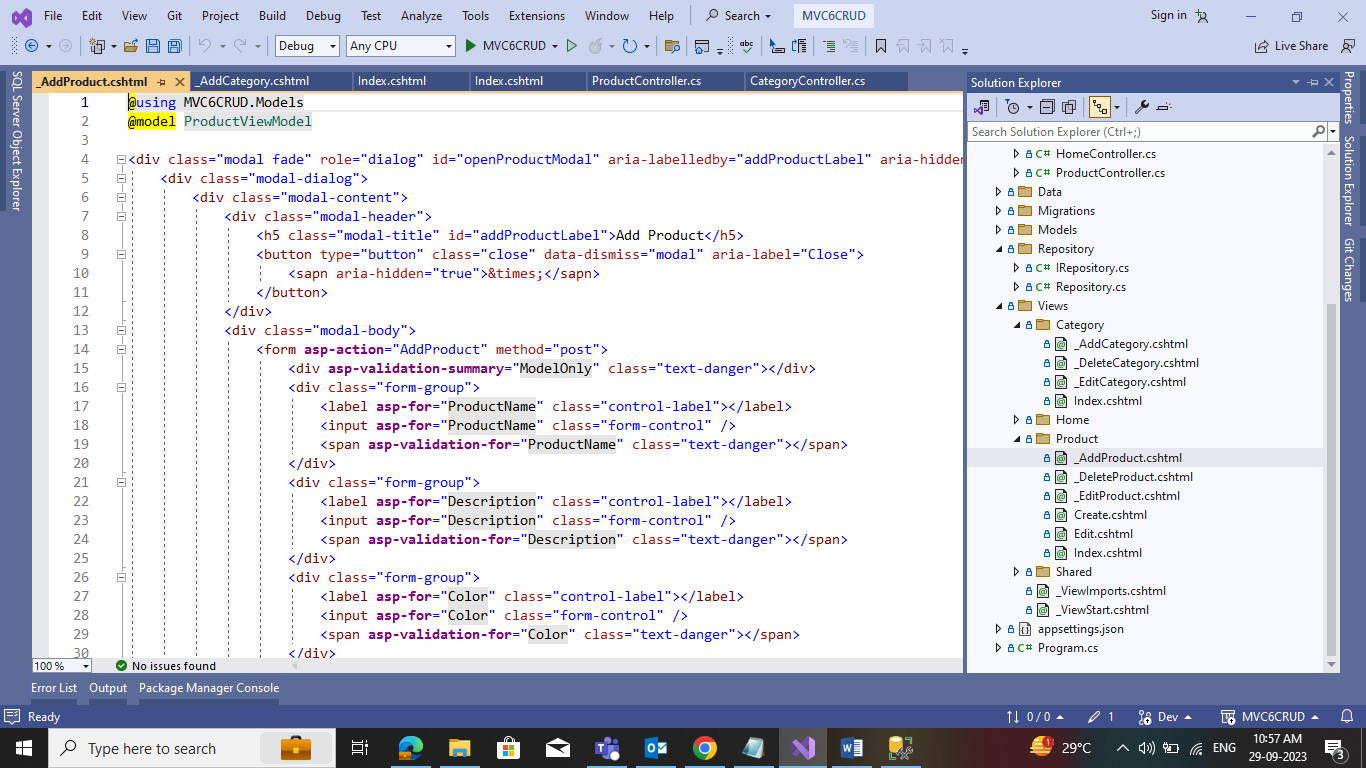




When Category View default(Index) and Product View default(Index) added then get list of Category data from database

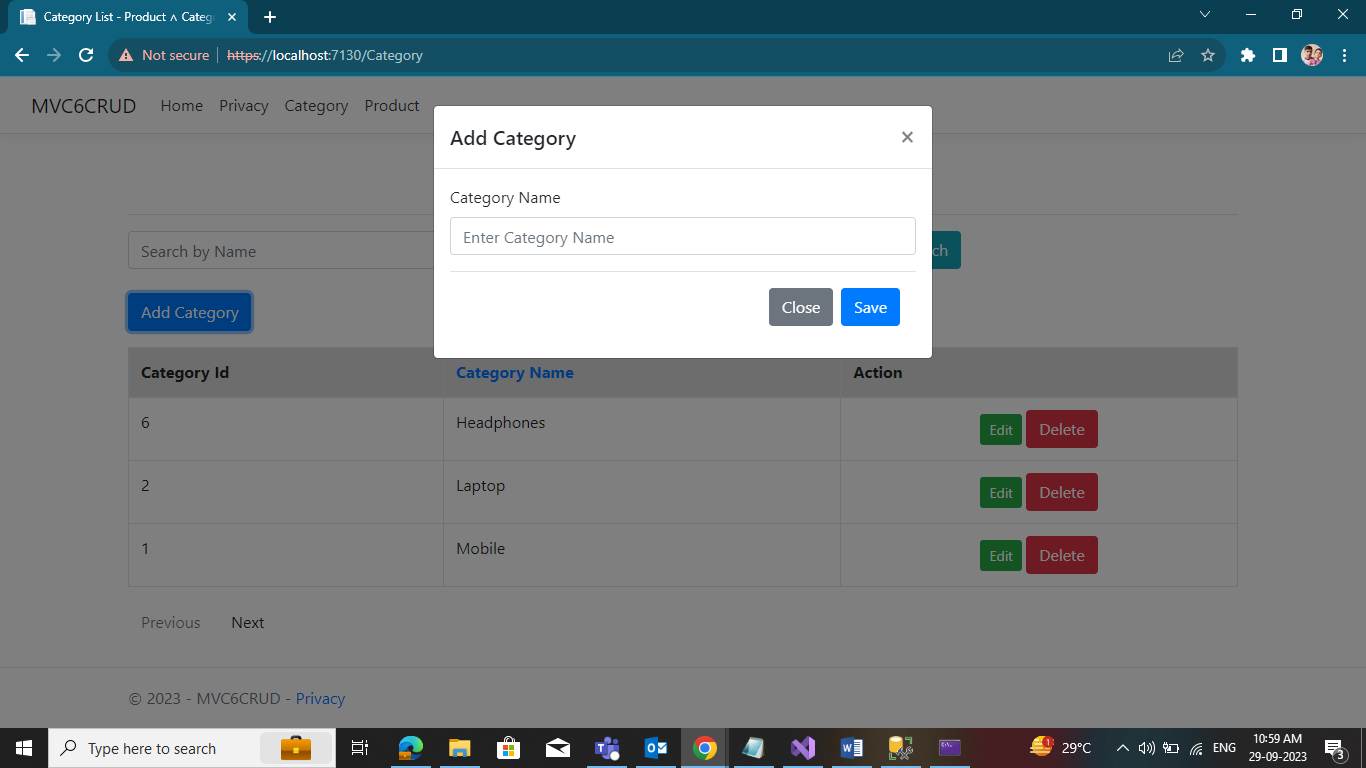
**Step 12:** Add partial view for AddCategory and AddProduct in Category View and Product View folder **and bootstrap Model popup functionality for both partial view.**

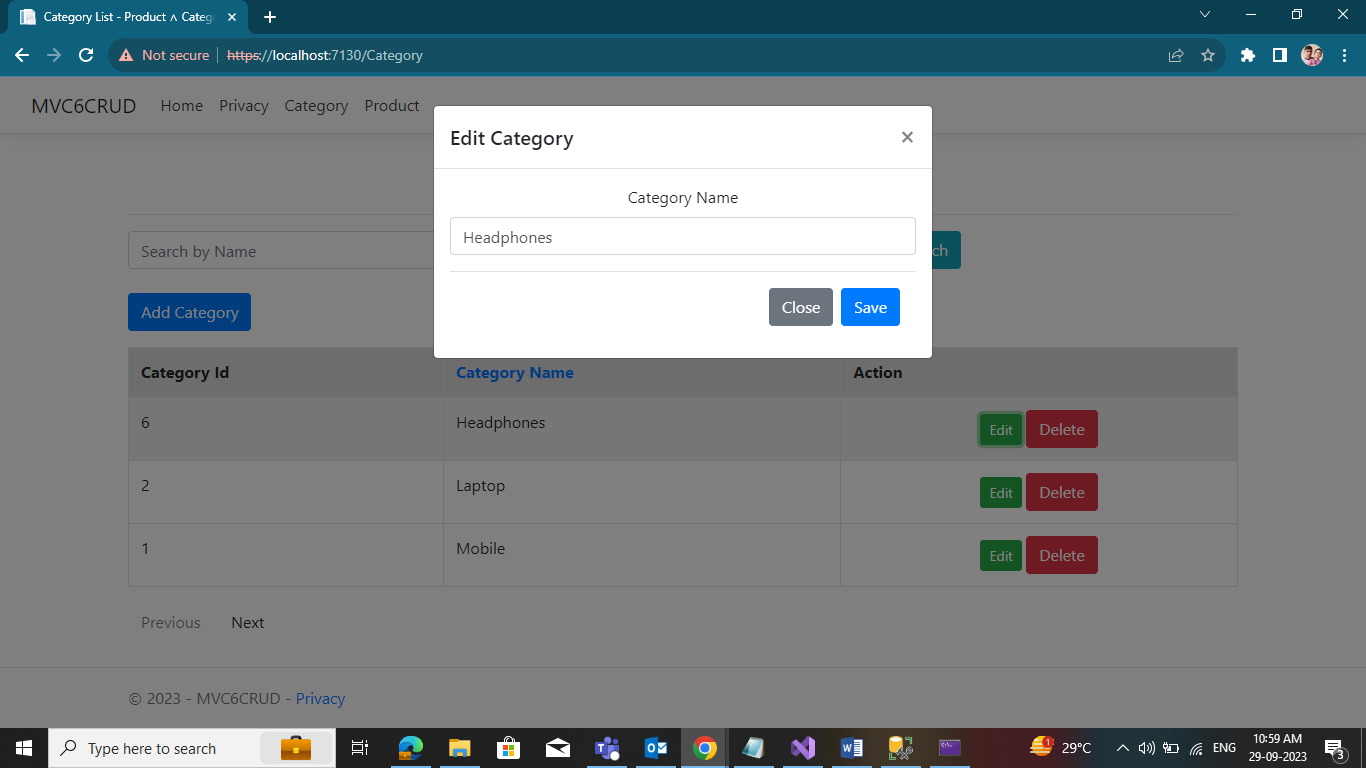


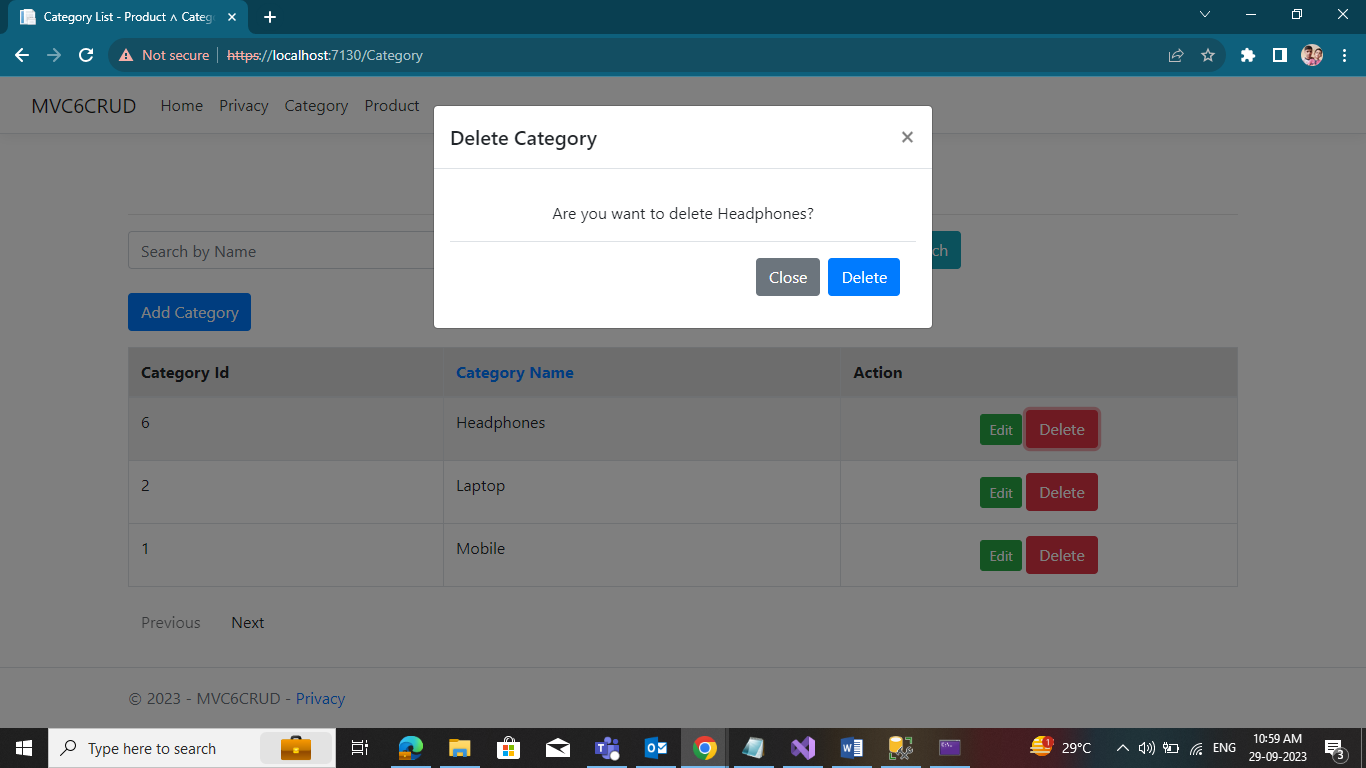


**Category CRUD operation screens**









**Product CRUD operation screens**



