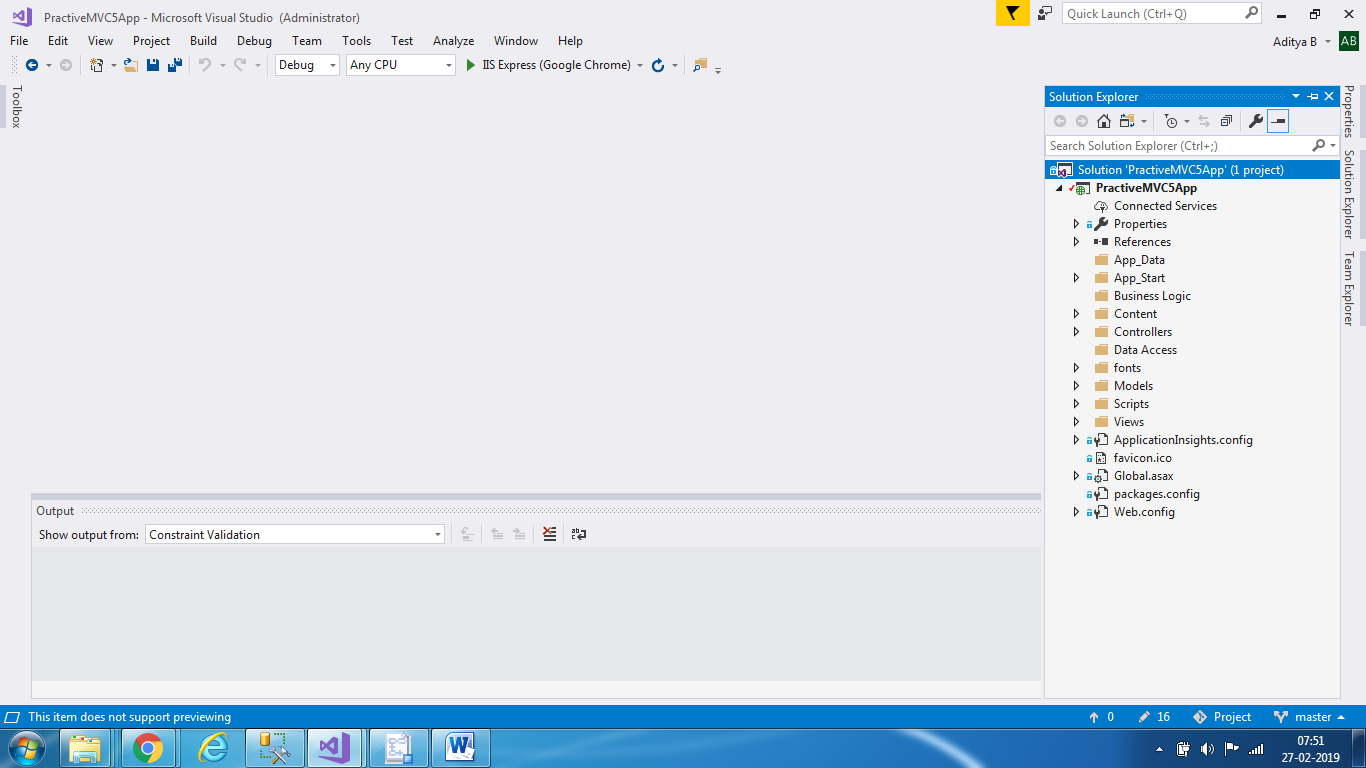
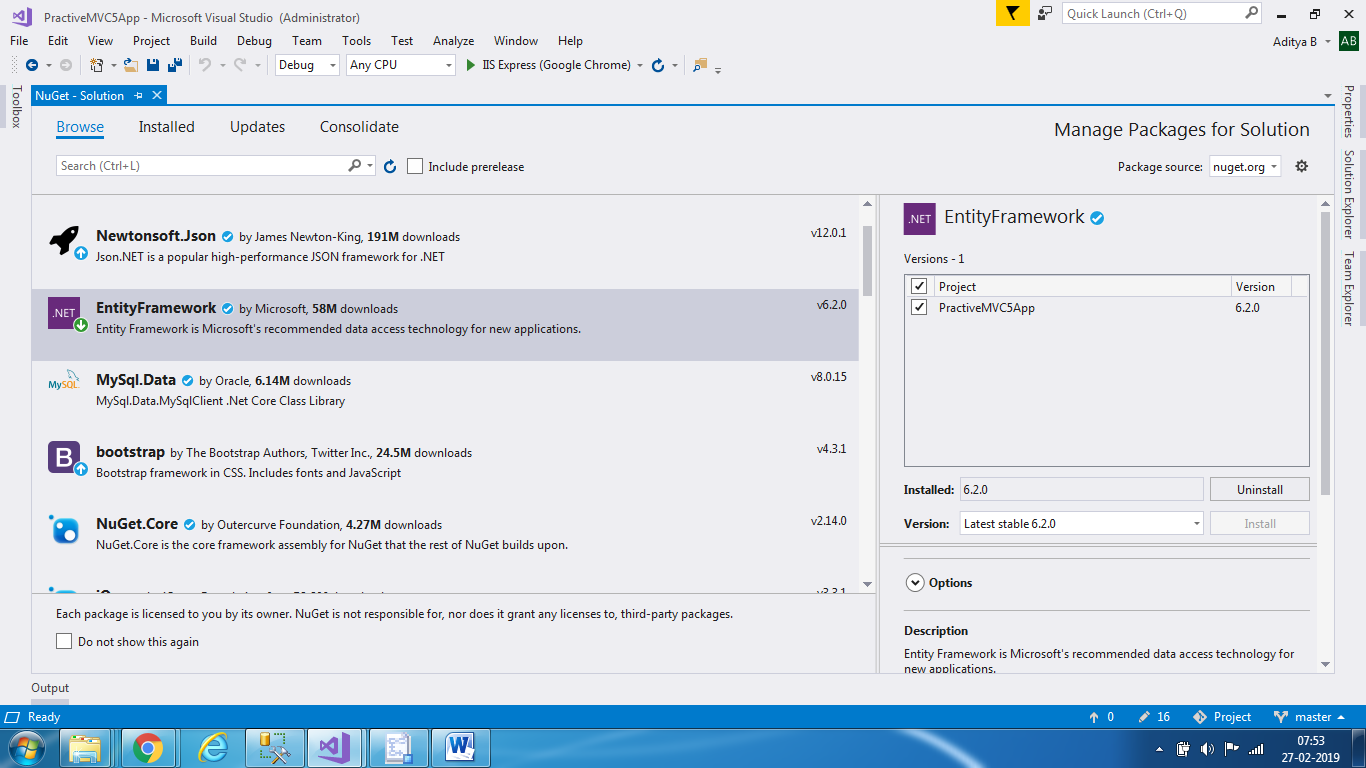
**Generating Entities from CSV**

Here are the steps followed for implementing this project -

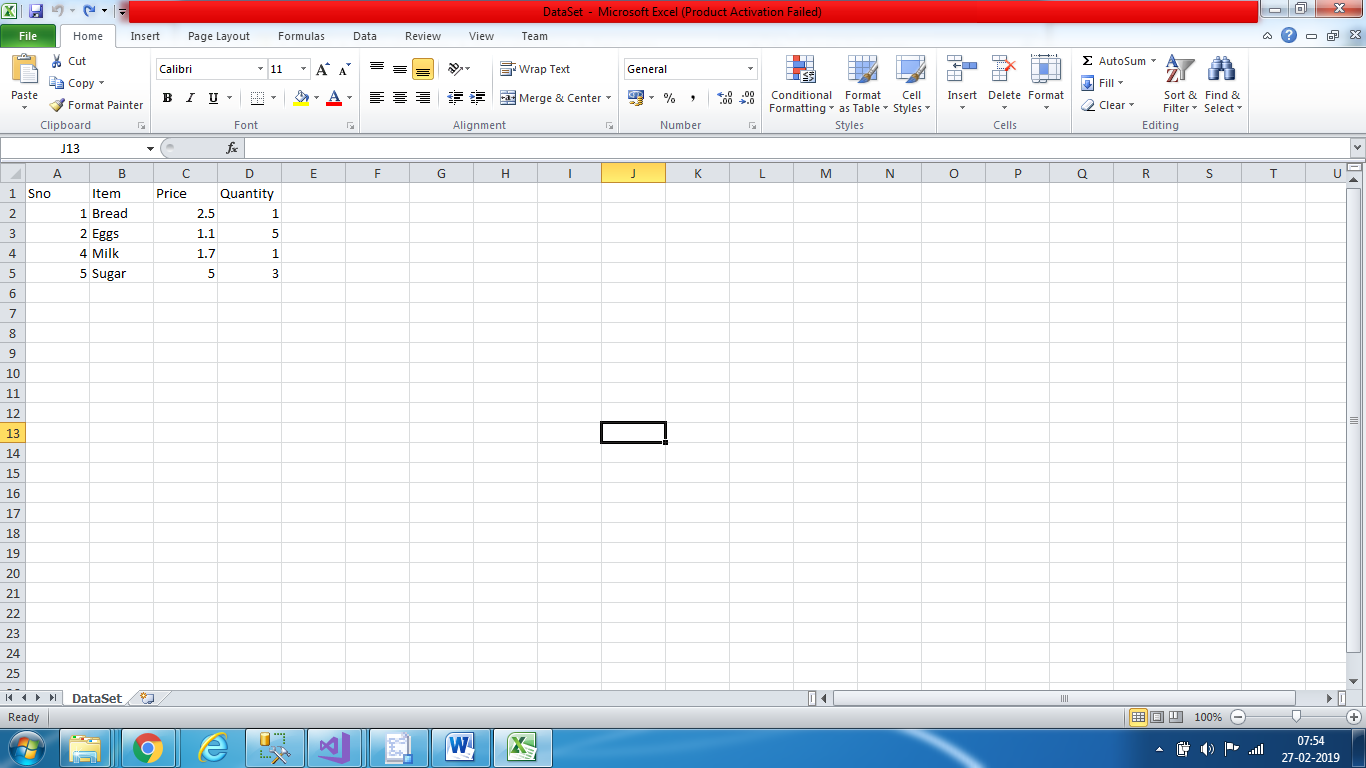
1. Create an MVC project using visual studio



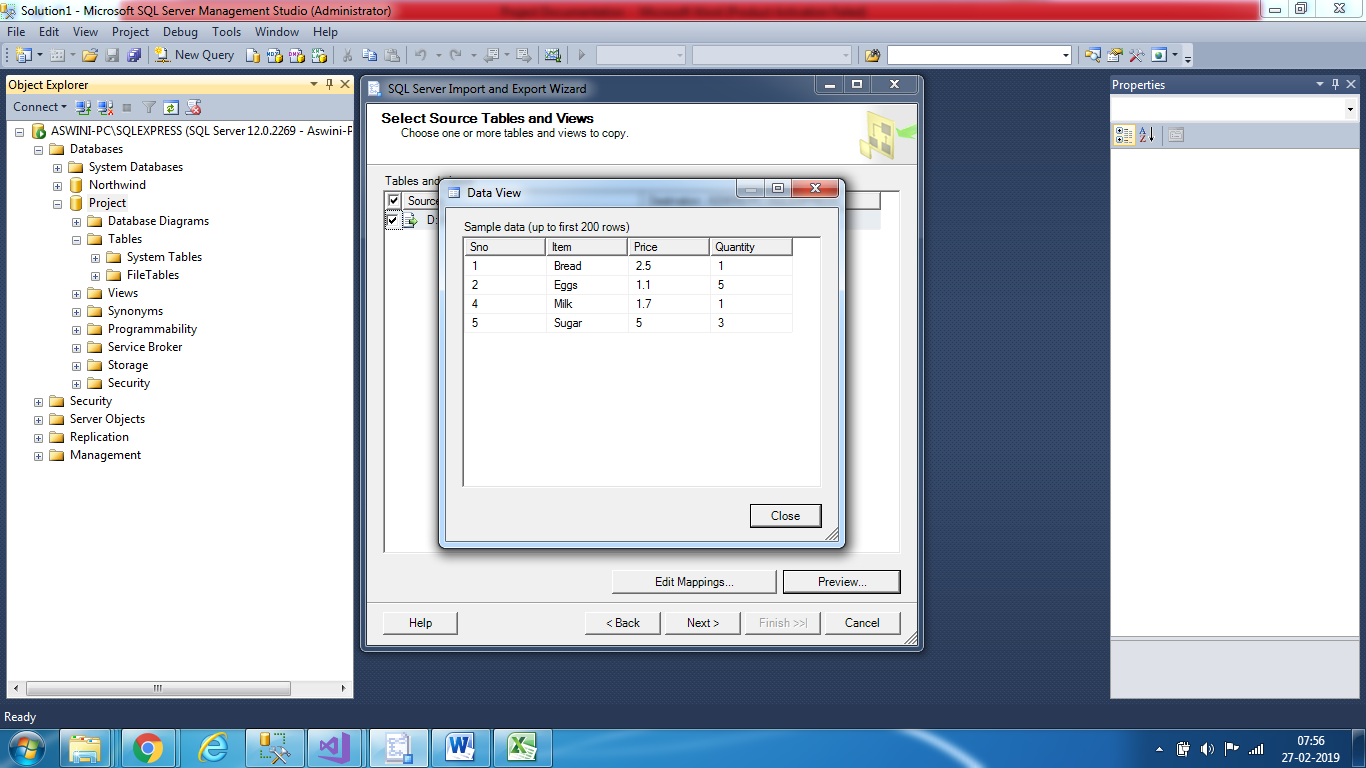
1. Add Entity Framework 6 using Nuget Package Manager



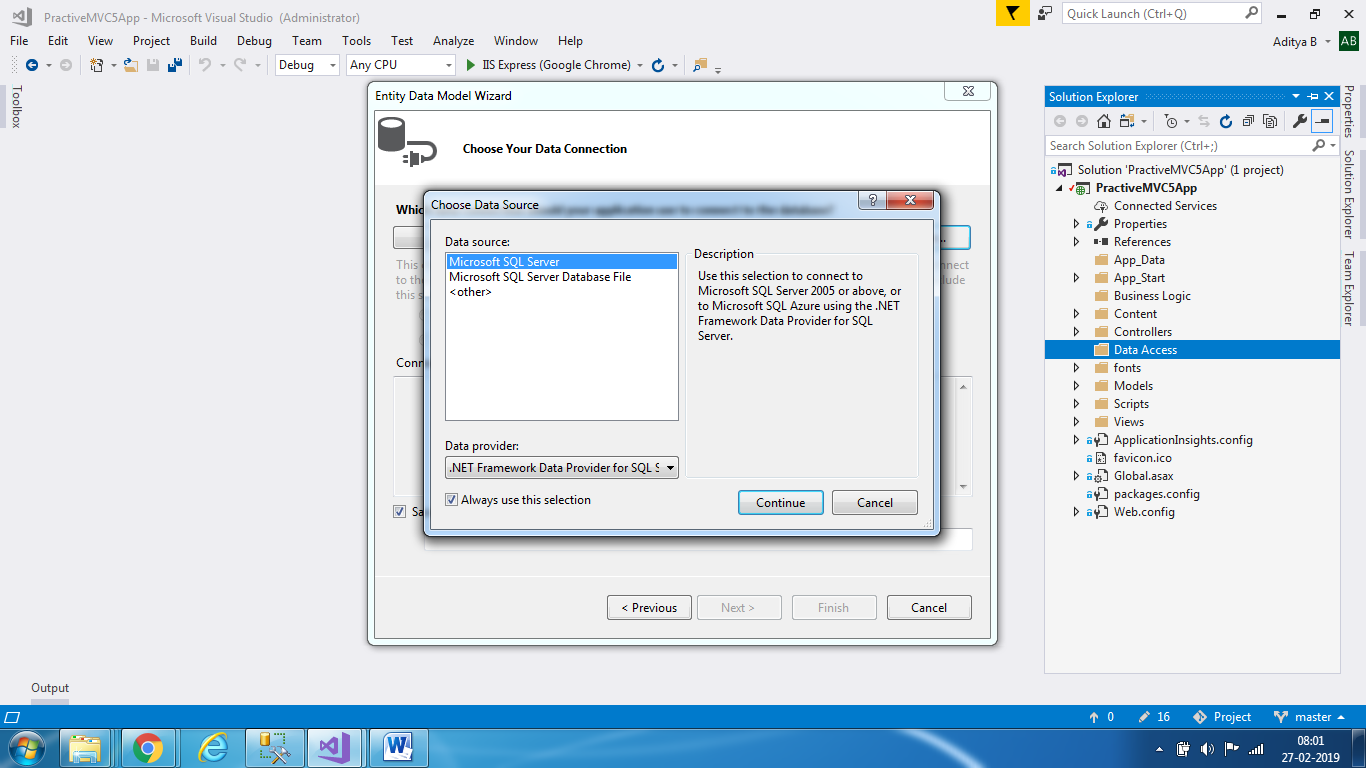
1. Create a sample CSV file with sample data



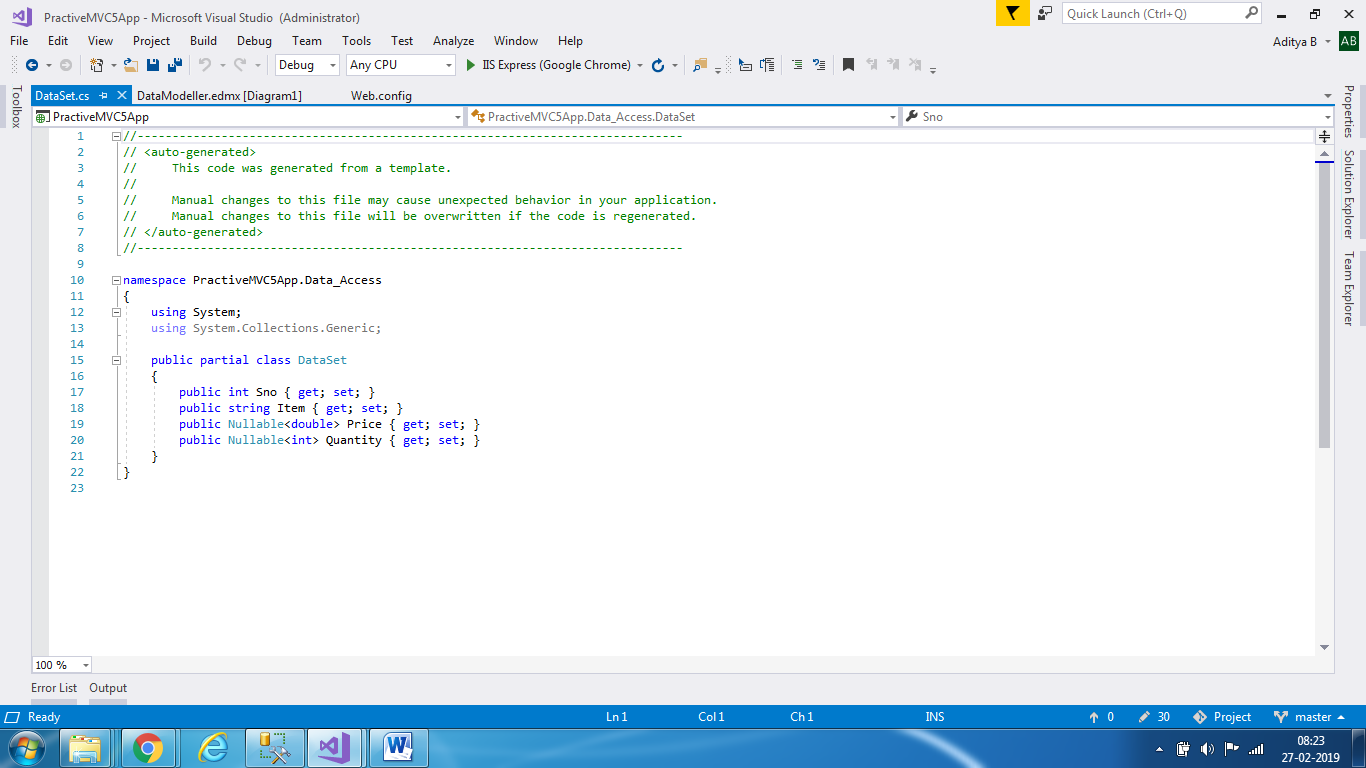
1. Import data from the CSV into a table in SQL Server using the import wizard



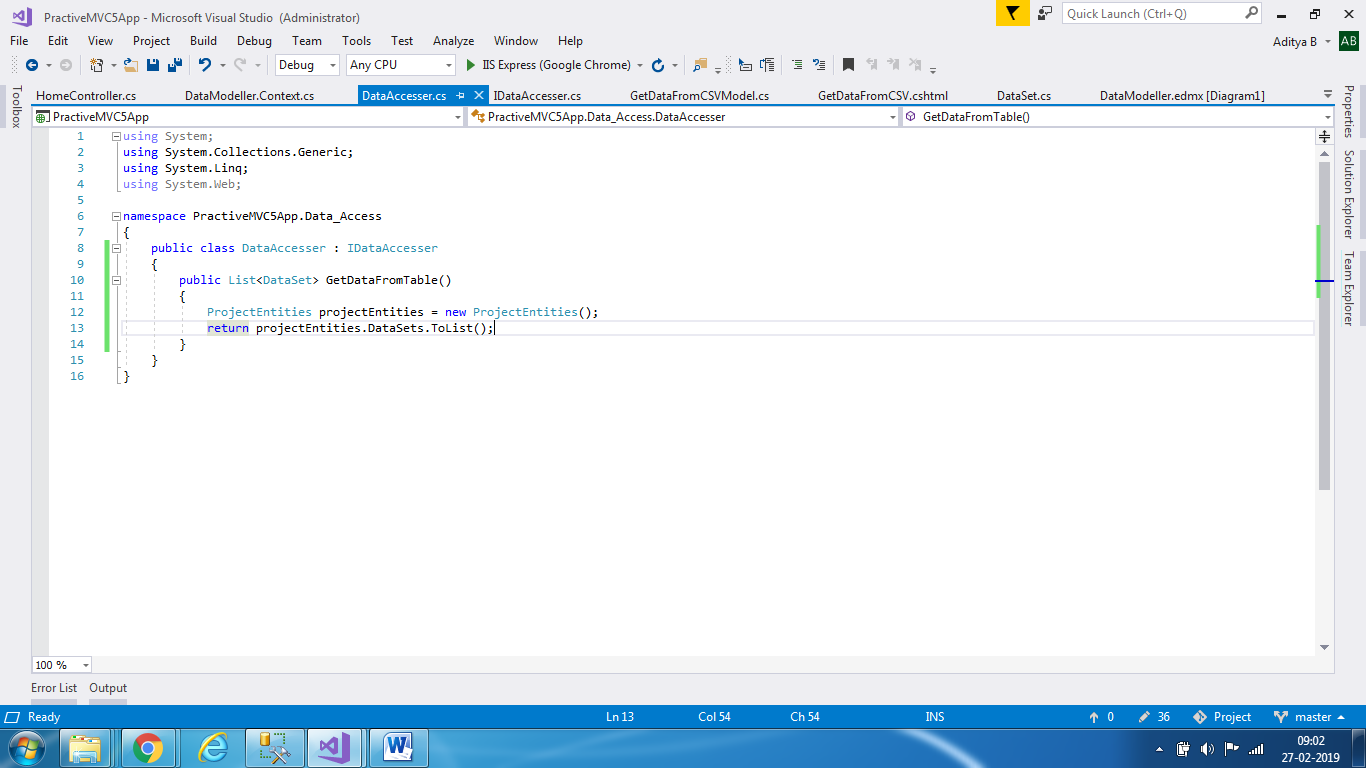
1. Create an edmx (entity data model) file in visual studio by connecting to the database server and selecting the database and tables to import



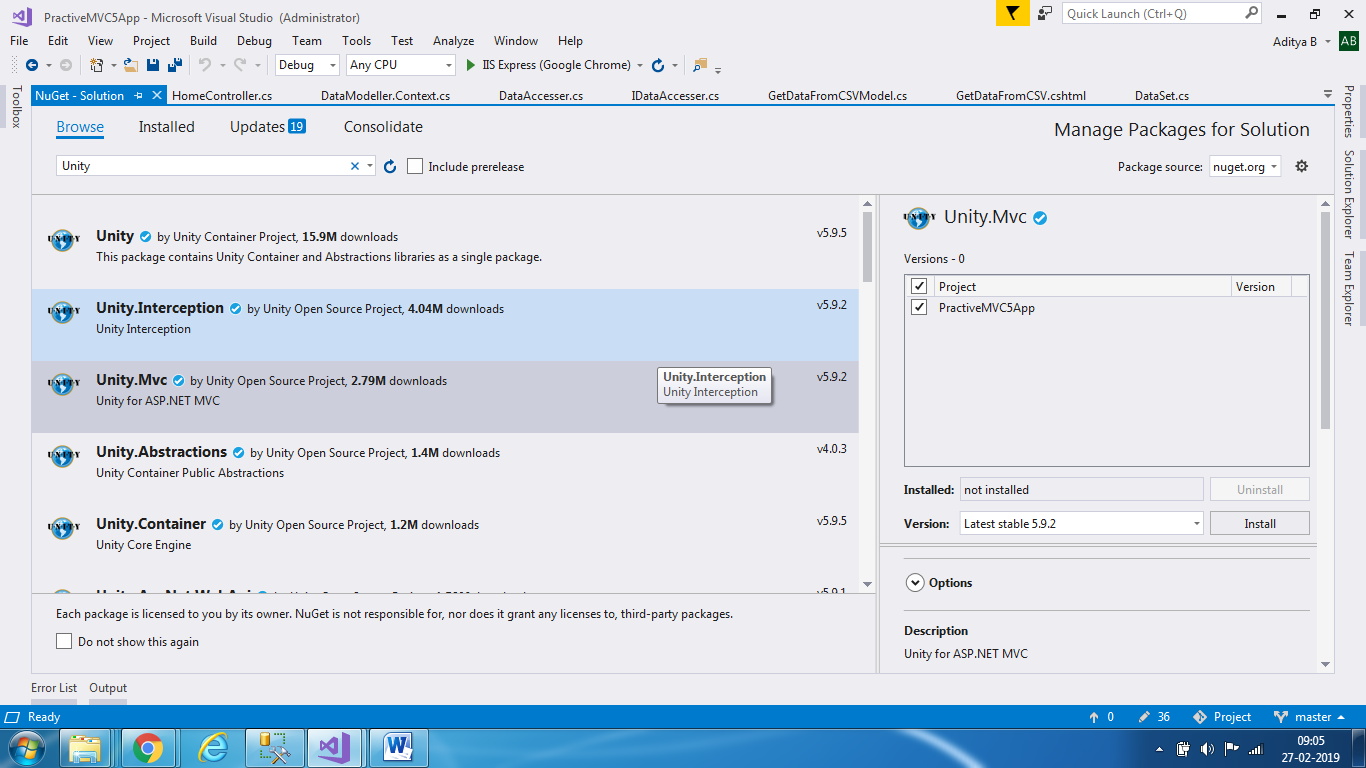
1. After entering the connection details, the entity framework will pick up the table and metadata regarding column types and auto generate the class file with the required properties and their datatypes



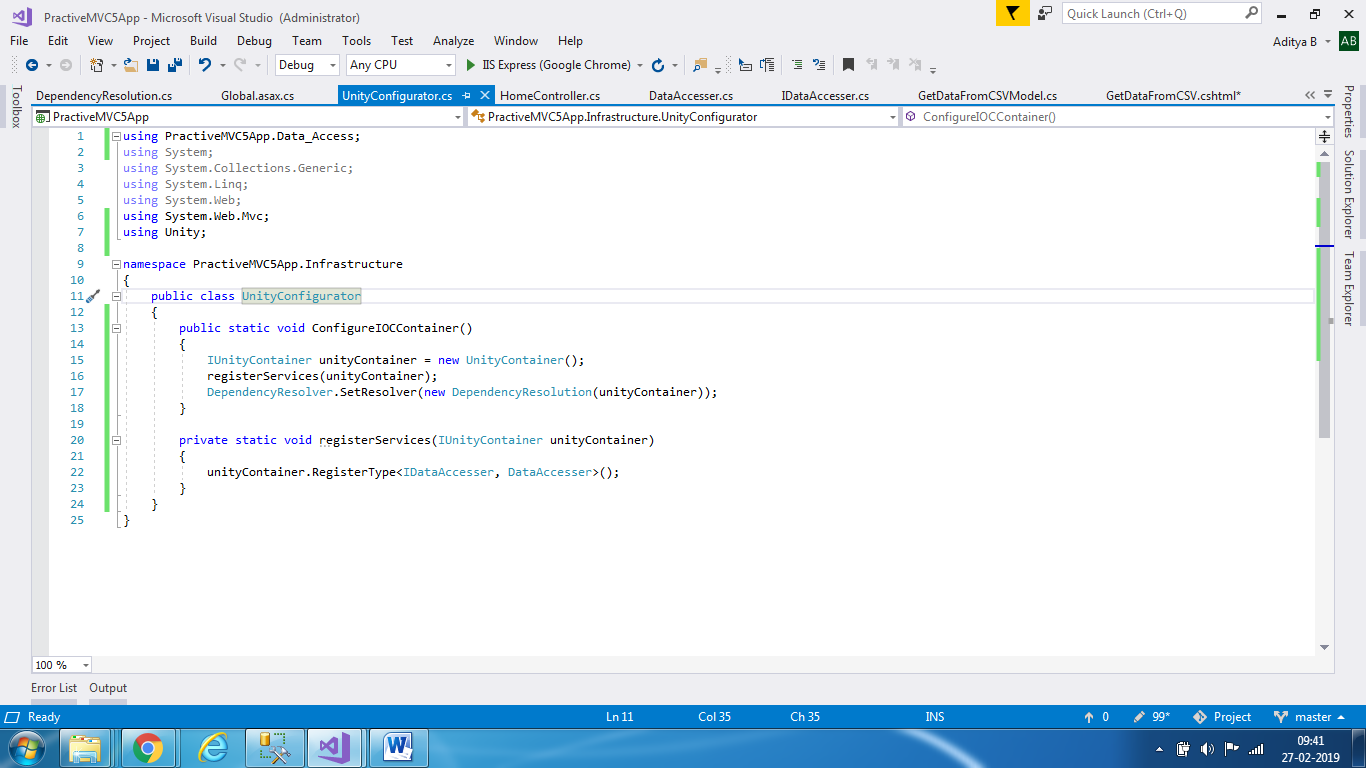
1. Now we can use this data and display it to a user in the form of a view in MVC. First we create an interface which exposes a method which gets the data from the data base. Then we create a class which implements this method to return the data from the DB. This method can be called in our controller using Unity Dependency injection for inversion of control.

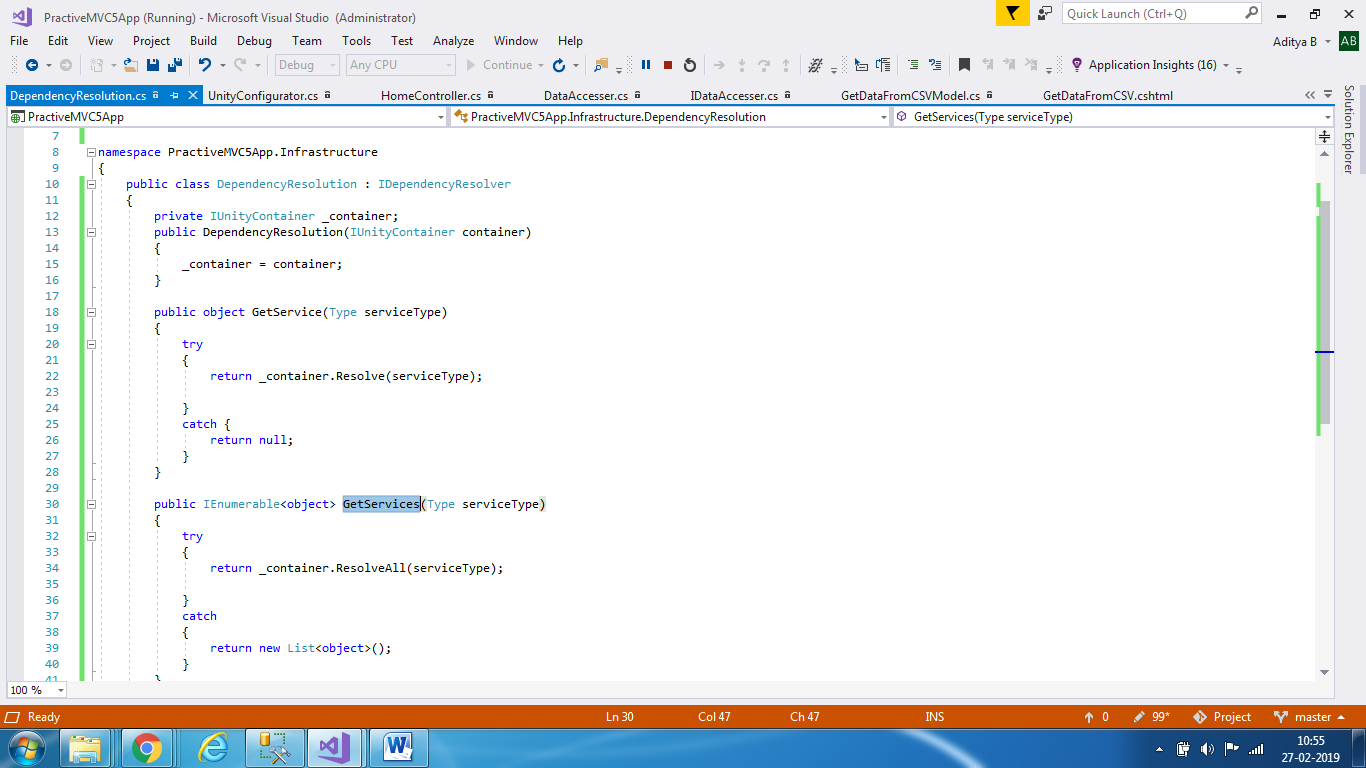


1. Install Unity from Nuget Package manager

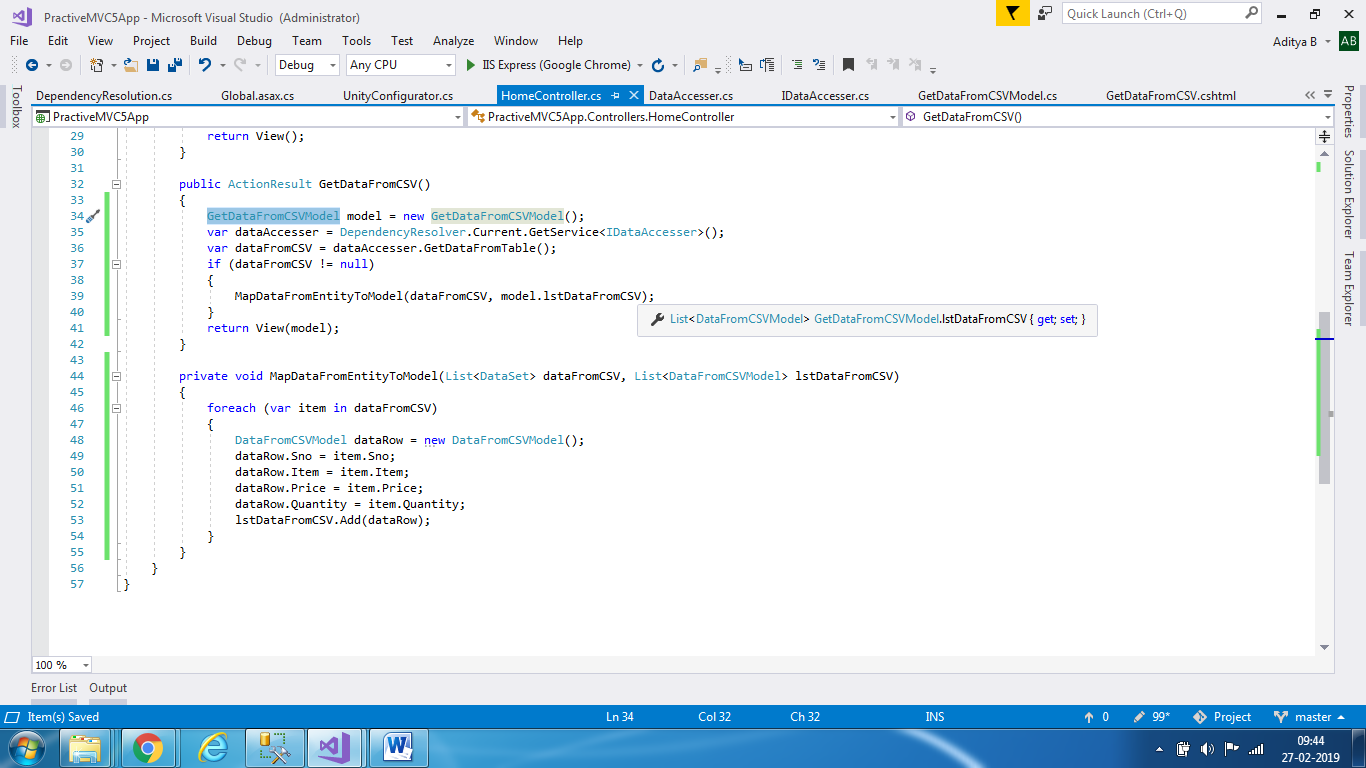


1. Setup configuration for Unity Container and Dependency Resolvers

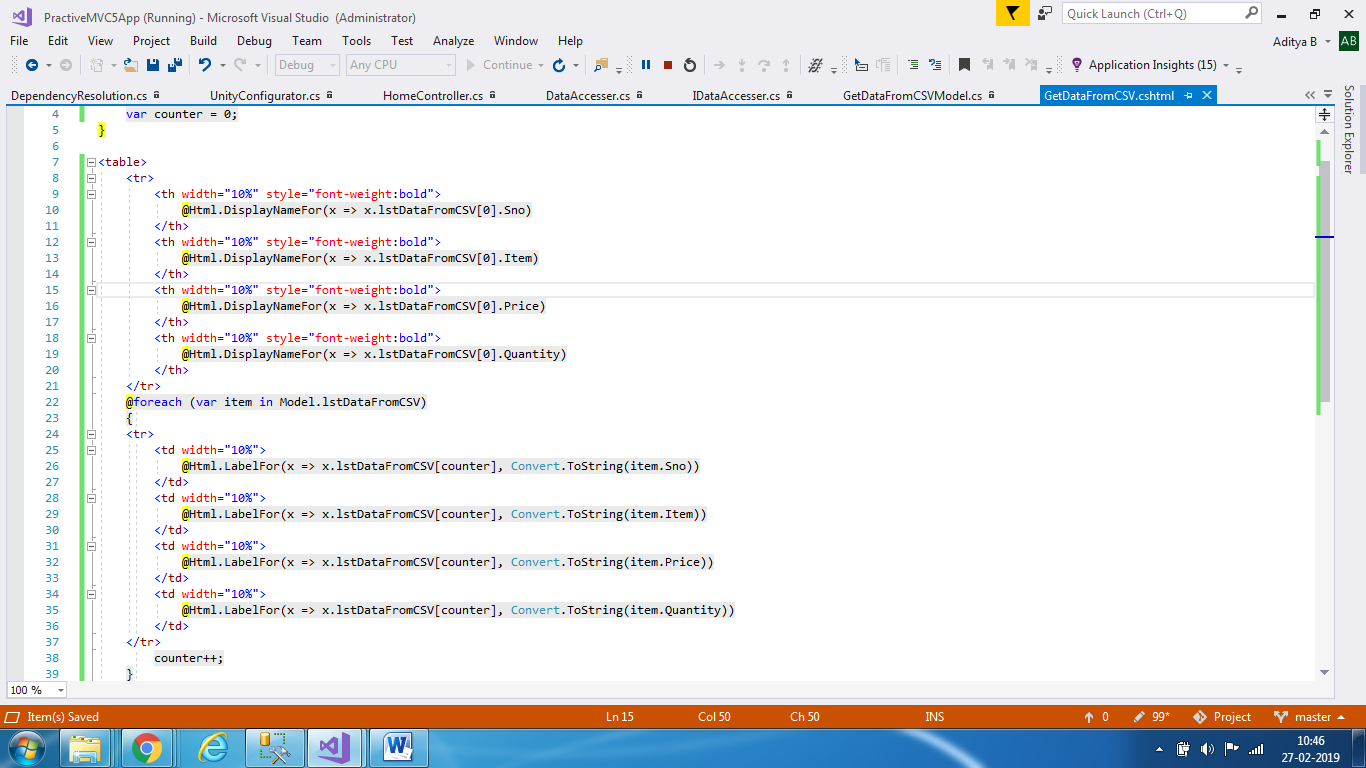




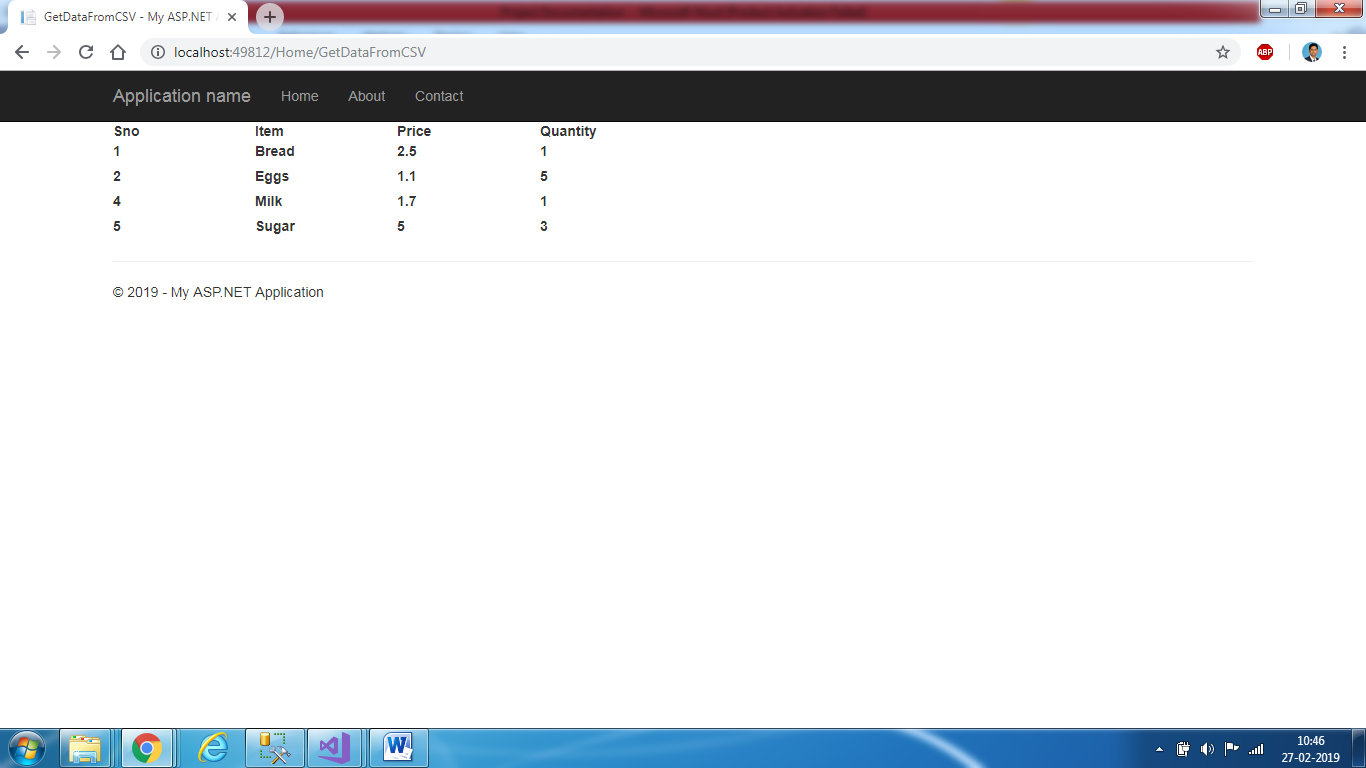
1. In our controller, we can now call the methods to get the data from the database and send it to the view for display using our model



1. Display the data in the view



1. Final View in the web browser –



**Coding Principles Used –**

1. Interfaces – For Creating a service which gets data from the database and returns it to a user
2. Entity Framework – Database first approach for generating Entity types from the database
3. C# - Used C# in the MVC framework as the primary language for executing the project
4. SOLID – Wrote code using solid principles and guidelines. Classes have single dependency, are open for extension but closed for modification, are perfectly substitutable for derived classes, the interfaces are segregated and classes are only implementing methods that they need to and dependency inversion is obtained through dependency injection using Unity
5. Inversion of control is obtained through dependency injection using Unity and loose coupling of code in MVC
6. SQL was used for creating the database and table from the original CSV file