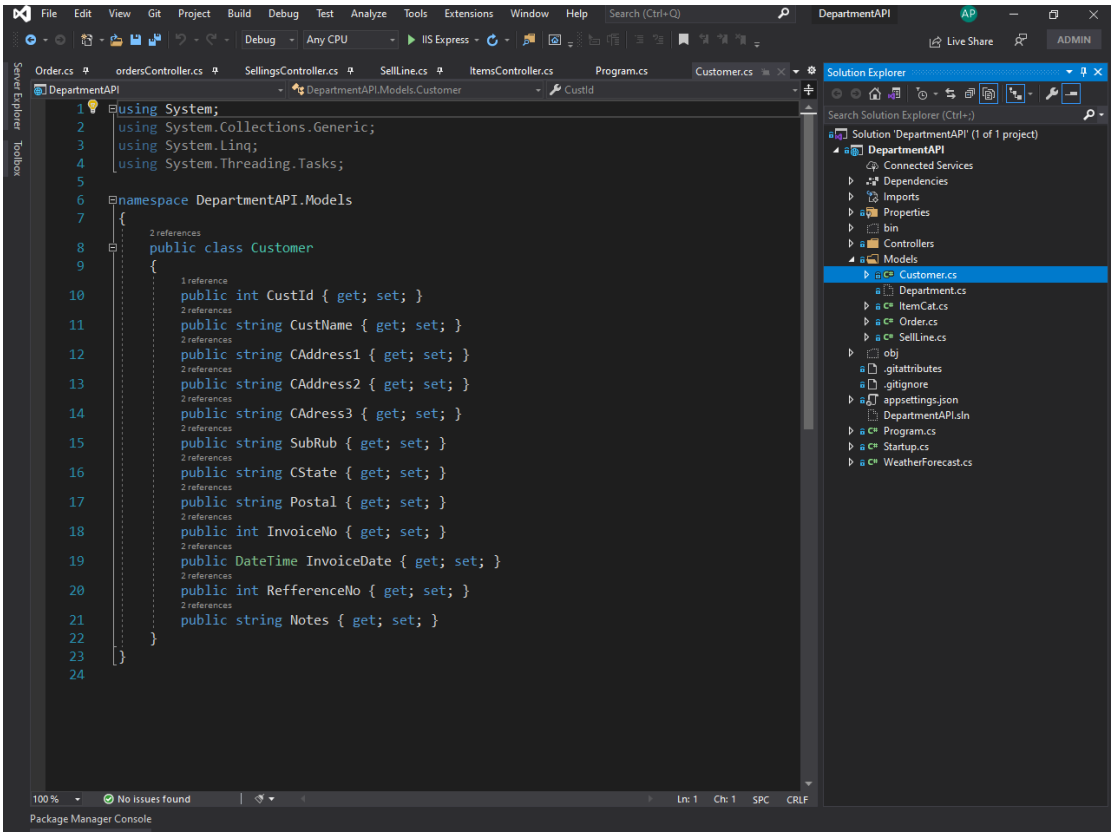
This sample project has been implemented per requirenments that given by spill software company…

Documentation.

In this simple project I have implemented following functions and procedures.

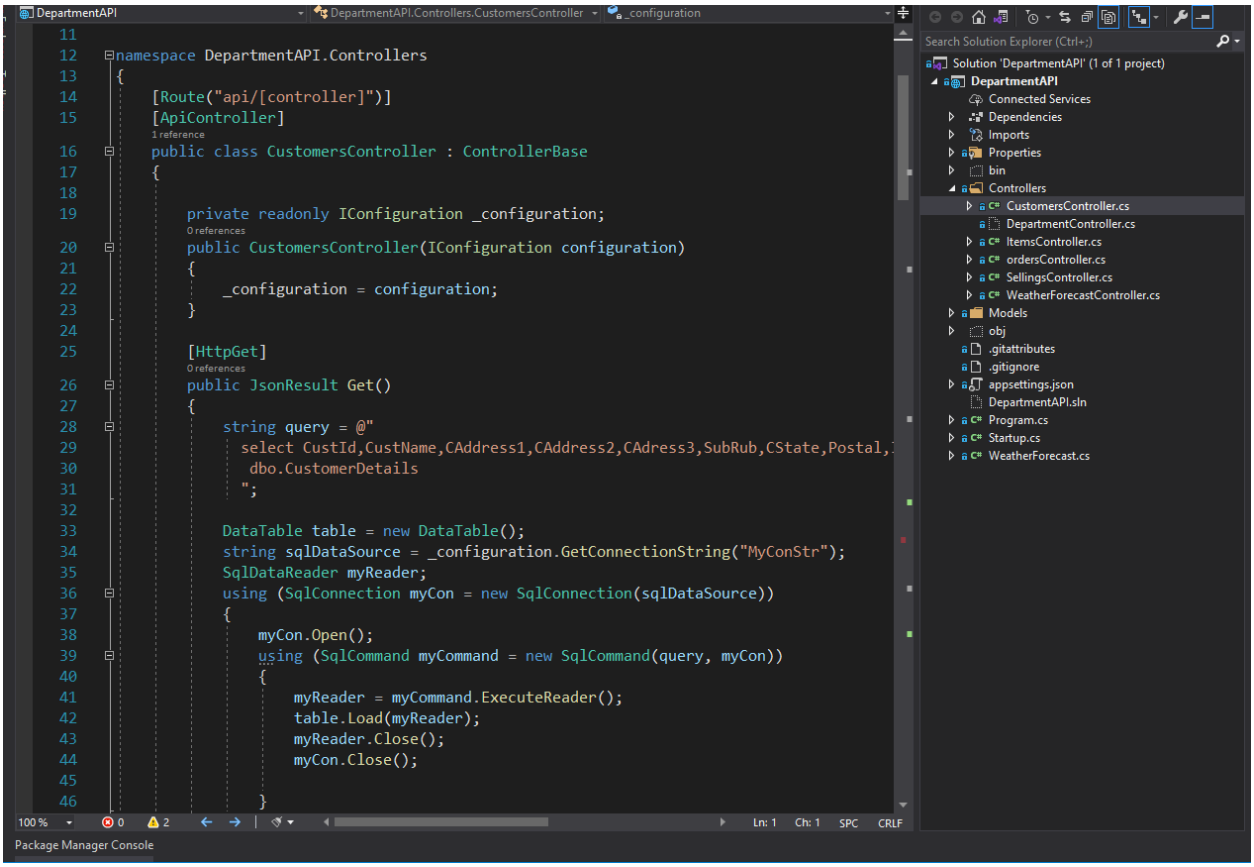
1. End points to INSERT,EDIT, VIEW and DELETE a customer to database through and API,

The Model folder is called Models and there is a class file called “Customer.cs” to view all requirements, it can now extend to catch the data coming from an UI.

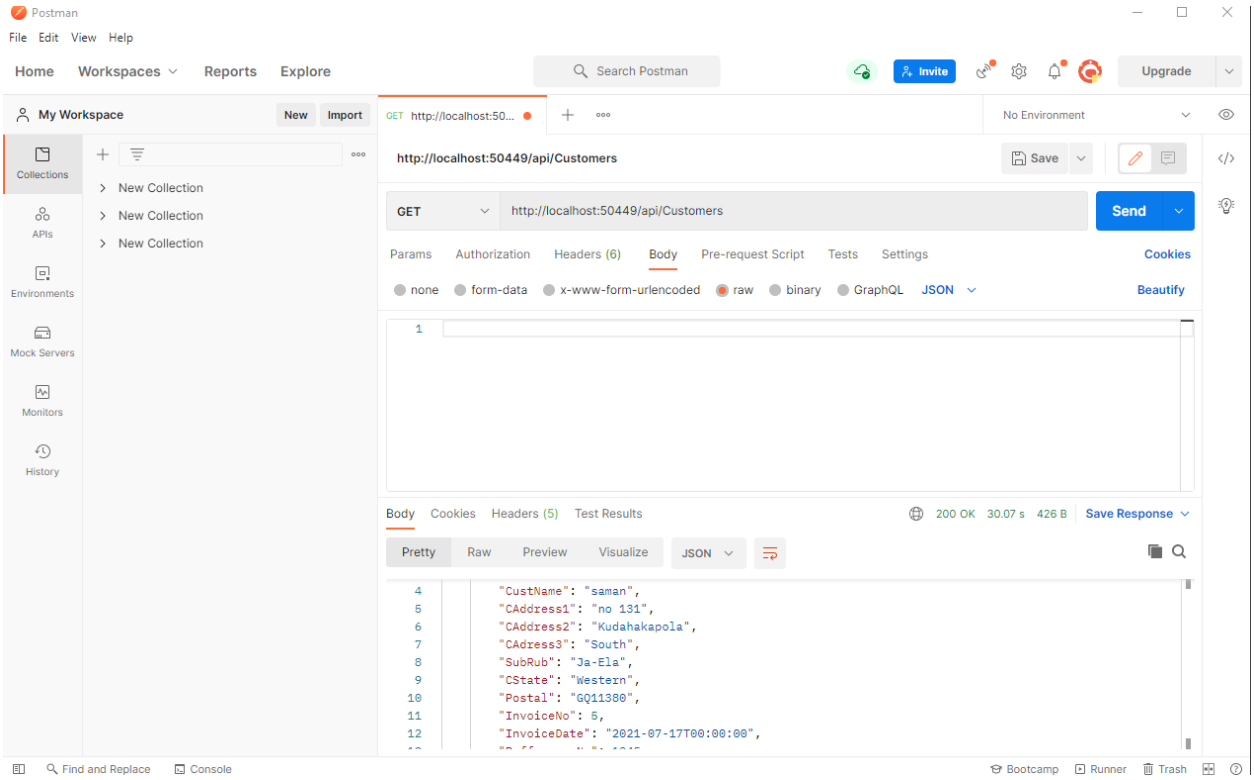


2. There is a controller called “CustomersController” it performs the all CRUD operation for customer entity related queries.

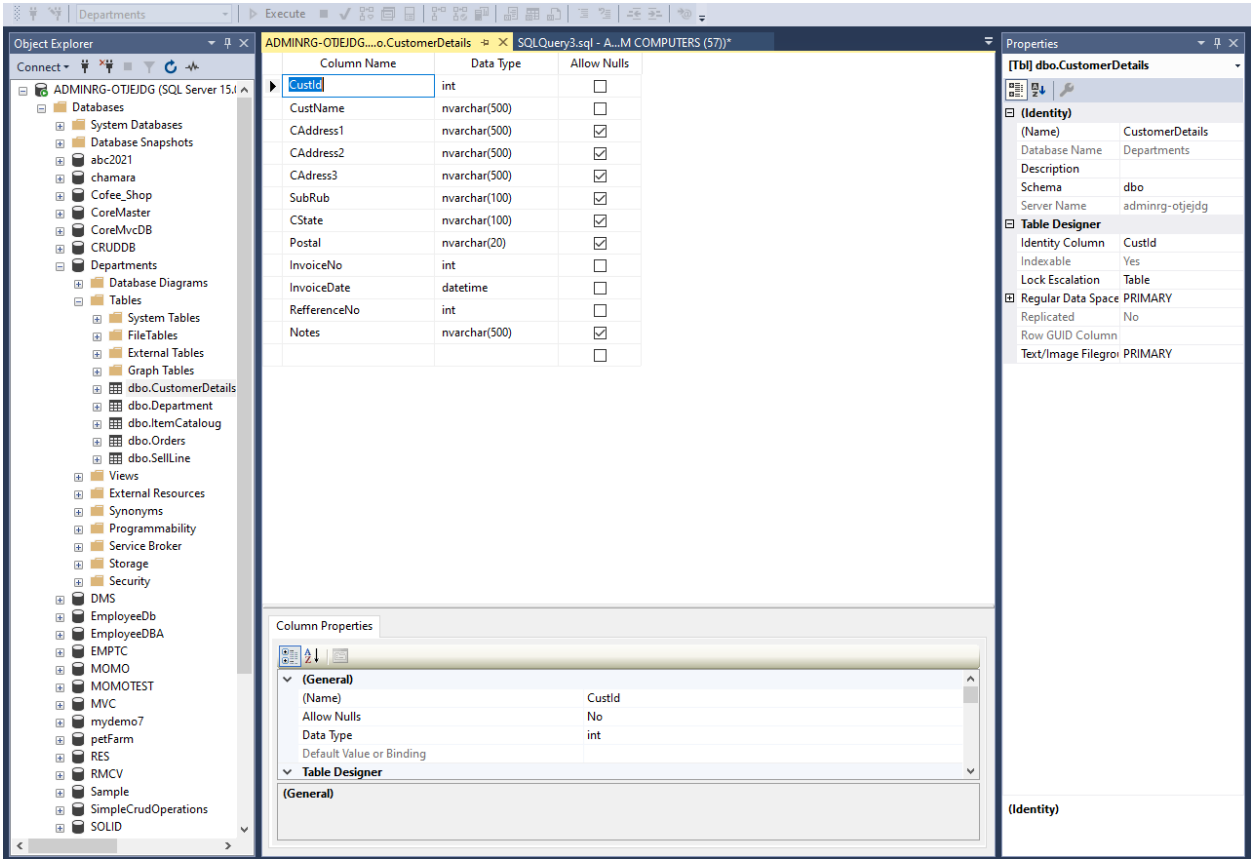
Attached Photo no 1 show the Screen capture of Controller and photo no 2 shows the checked output through Postman

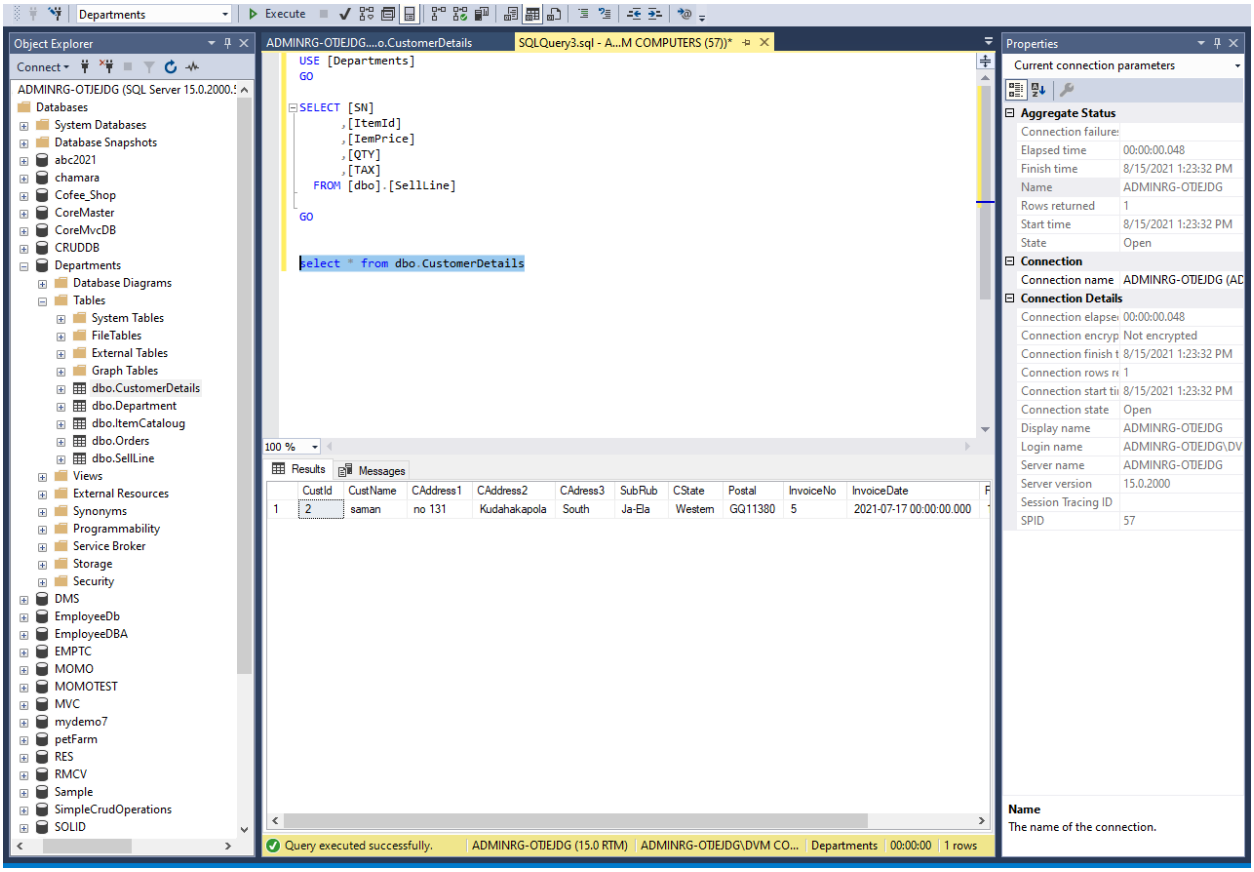


```
DepartmentAPI | DepartmentAPI.Controllers.CustomersController | _configuration

namespace DepartmentAPI.Controllers
{
    [Route("api/[controller]")]
    [ApiController]
    1 reference
    public class CustomersController : ControllerBase
    {

        private readonly IConfiguration _configuration;
        0 references
        public CustomersController(IConfiguration configuration)
        {
            _configuration = configuration;
        }

        [HttpGet]
        0 references
        public JsonResult Get()
        {
            string query = @"
              select CustId,CustName,CAddress1,CAddress2,CAdress3,SubRub,CState,Postal,
               dbo.CustomerDetails
              ";

            DataTable table = new DataTable();
            string sqlDataSource = _configuration.GetConnectionString("MyConStr");
            SqlDataReader myReader;
            using (SqlConnection myCon = new SqlConnection(sqlDataSource))
            {
                myCon.Open();
                using (SqlCommand myCommand = new SqlCommand(query, myCon))
                {
                    myReader = myCommand.ExecuteReader();
                    table.Load(myReader);
                    myReader.Close();
                    myCon.Close();

                }
```



Below shows the table design and the validated output from SQL Server database.


Departments
Execute
Object Explorer
Connect
ADMINRG-OTJEJDG (SQL Server 15.
Databases
System Databases
Database Snapshots
abc2021
chamara
Cofee_Shop
CoreMaster
CoreMvcDB
CRUDDB
Departments
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.CustomerDetails
dbo.Department
dbo.ItemCataloug
dbo.Orders
dbo.SellLine
Views
External Resources
Synonyms
Programmability
Service Broker
Storage
Security
DMS
EmployeeDb
EmployeeDBA
EMPTC
MOMO
MOMOTEST
MVC
mydemo7
petFarm
RES
RMCV
Sample
SimpleCrudOperations
SOLID
ADMINRG-OTJEJDG....o.CustomerDetails
SQLQuery3.sql - A...M COMPUTERS (57))*
Column Name
Data Type
Allow Nulls
CustId int
CustName nvarchar(500)
CAddress1 nvarchar(500)
CAddress2 nvarchar(500)
CAdress3 nvarchar(500)
SubRub nvarchar(100)
CState nvarchar(100)
Postal nvarchar(20)
InvoiceNo int
InvoiceDate datetime
RefferenceNo int
Notes nvarchar(500)
Column Properties
(General)
(Name) CustId
Allow Nulls No
Data Type int
Default Value or Binding
Table Designer
(General)
Properties
[Tbl] dbo.CustomerDetails
(Identity)
(Name) CustomerDetails
Database Name Departments
Description
Schema dbo
Server Name adminrg-otjejdg
Table Designer
Identity Column CustId
Indexable Yes
Lock Escalation Table
Regular Data Space PRIMARY
Replicated No
Row GUID Column
Text/Image Filegro PRIMARY
(Identity)



As same manner remaining functions also are working properly…

2. Let Move for core of Project.

There is a class called “ItemCat.cs” in Model Folder that is able to catch the data that coming from front end Application.

DepartmentAPI | DepartmentAPI.Models.ItemCat | ItemId

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

namespace DepartmentAPI.Models
{
    2 references
    public class ItemCat
    {
        1 reference
        public int ItemId { get; set; }   //this help to add/edit/view and delete item
        2 references
        public string ItemCode { get; set; }
        2 references
        public decimal IemPrice { get; set; }
        2 references
        public string ItemNote { get; set; }


    }
}
```

100 % | No issues found | Ln: 1 | Ch: 1 | SPC | CRLF

Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'DepartmentAPI' (1 of 1 project)
  - DepartmentAPI
    - Connected Services
    - Dependencies
    - Imports
    - Properties
    - bin
    - Controllers
    - Models
      - Customer.cs
      - Department.cs
      - ItemCat.cs
      - Order.cs
      - SellLine.cs
    - obj
    - .gitattributes
    - .gitignore
    - appsettings.json
    - DepartmentAPI.sln
    - Program.cs
    - Startup.cs
    - WeatherForecast.cs

Order.cs | ordersController.cs | SellingsController.cs | SellLine.cs | CustomersController.cs | ItemsController.cs

DepartmentAPI | DepartmentAPI.Controllers.ItemsController | Post(ItemCat itm)

```
52
53          [HttpPost]
            0 references
54          public JsonResult Post(ItemCat itm)
55          {
56              string query = @"
57                insert into  dbo.ItemCataloug
58                values (@ItemCode,@IemPrice,@ItemNote)
59                ";
60
61              DataTable table = new DataTable();
62              string sqlDataSource = _configuration.GetConnectionString("MyConStr");
63              SqlDataReader myReader;
64              using (SqlConnection myCon = new SqlConnection(sqlDataSource))
65              {
66                  myCon.Open();
67                  using (SqlCommand myCommand = new SqlCommand(query, myCon))
68                  {
69                      myCommand.Parameters.AddWithValue("@ItemCode", itm.ItemCode);
70                      myCommand.Parameters.AddWithValue("@IemPrice", itm.IemPrice);
71                      myCommand.Parameters.AddWithValue("@ItemNote", itm.ItemNote);
72                      myReader = myCommand.ExecuteReader();
73                      table.Load(myReader);
74                      myReader.Close();
75                      myCon.Close();
76
77                  }
78              }
79              return new JsonResult("Record Posted!!");
80
81          }
82
83
84
85          [HttpPut]
            0 references
86          public JsonResult Put(ItemCat itm)
87          {
88              string query = @"
```

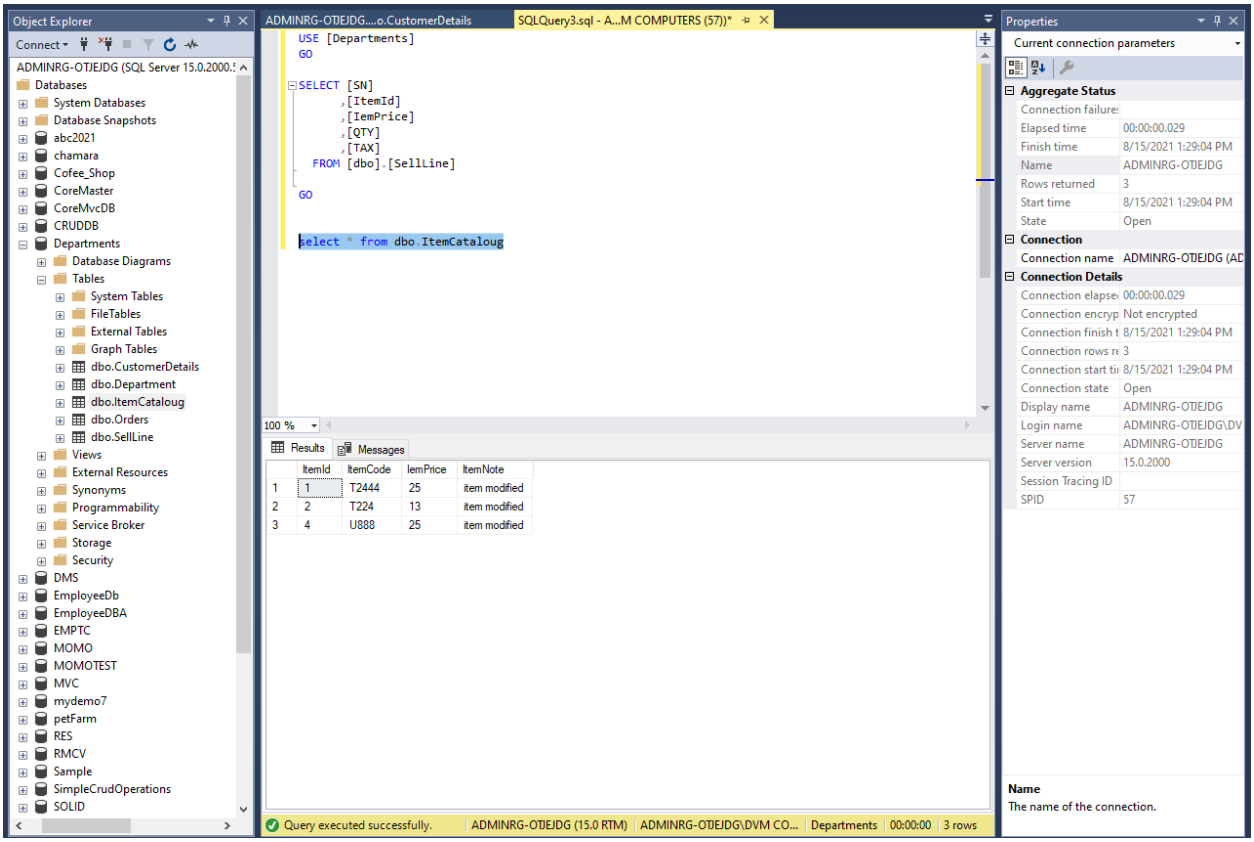
100 % | No issues found | Ln: 54 | Ch: 42 | SPC | CRLF

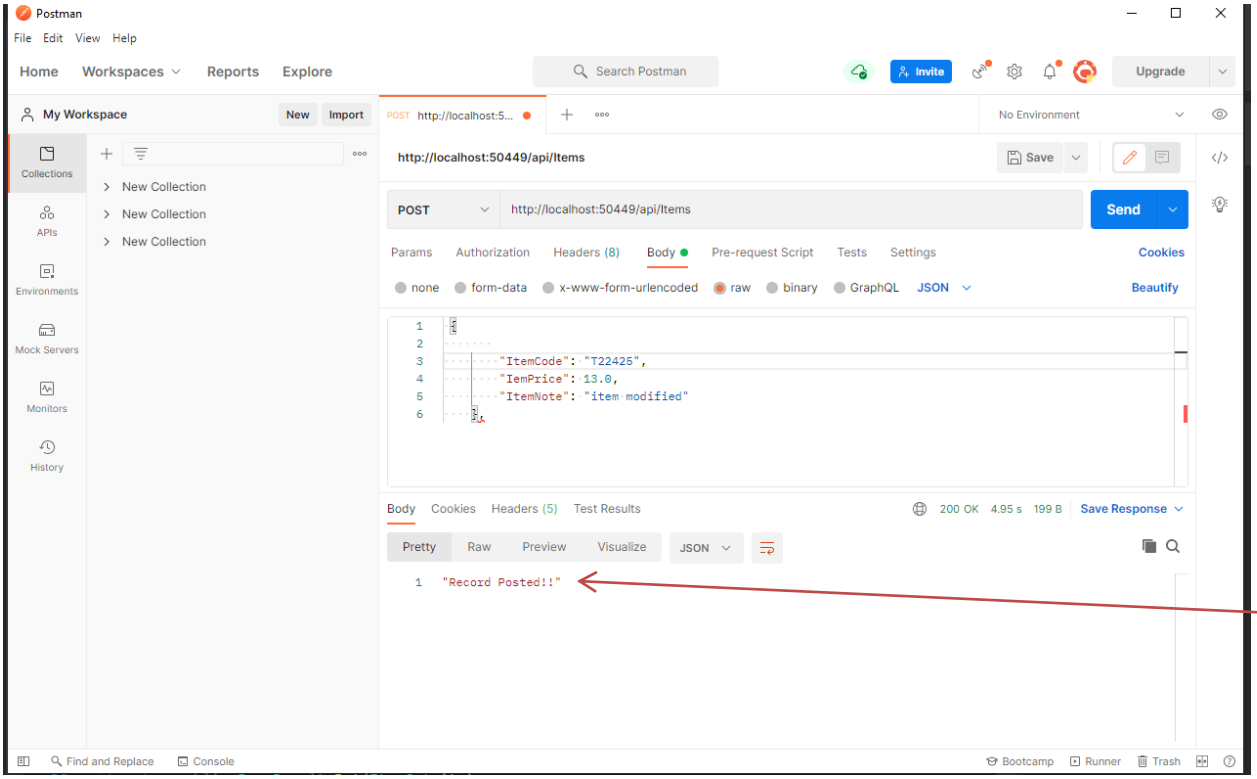
Package Manager Console

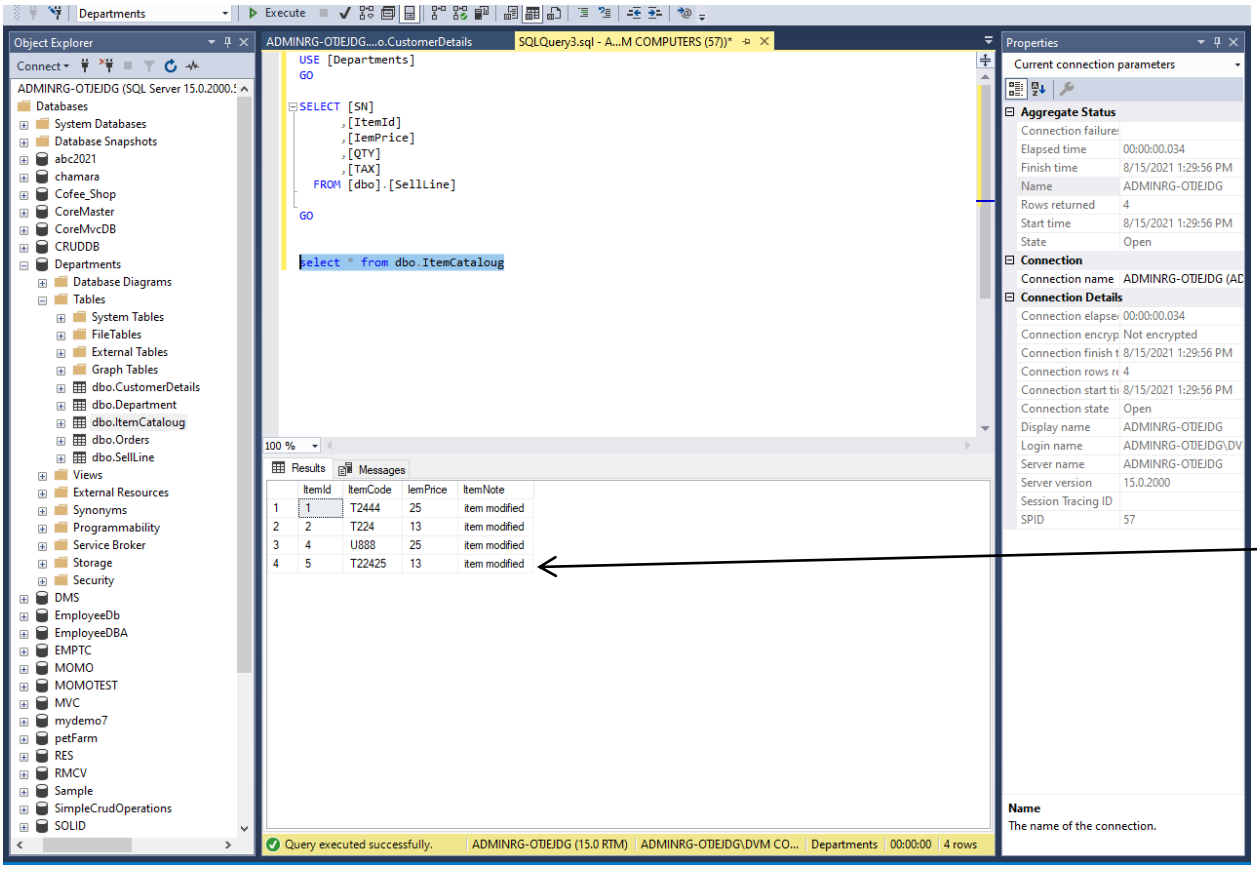
Solution Explorer

Search Solution Explorer (Ctrl+;)

- Solution 'DepartmentAPI' (1 of 1 project)
  - DepartmentAPI
    - Connected Services
    - Dependencies
    - Imports
    - Properties
    - bin
    - Controllers
      - CustomersController.cs
      - DepartmentController.cs
      - ItemsController.cs
      - ordersController.cs
      - SellingsController.cs
      - WeatherForecastController.cs
    - Models
      - Customer.cs
      - Department.cs
      - ItemCat.cs
      - Order.cs
      - SellLine.cs
    - obj
    - .gitattributes
    - .gitignore
    - appsettings.json
    - DepartmentAPI.sln
    - Program.cs
    - Startup.cs
    - WeatherForecast.cs


Object Explorer
Connect
ADMINRG-OTJEJDG (SQL Server 15.0.2000.
Databases
System Databases
Database Snapshots
abc2021
chamara
Cofee_Shop
CoreMaster
CoreMvcDB
CRUDDB
Departments
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.CustomerDetails
dbo.Department
dbo.ItemCataloug
dbo.Orders
dbo.SellLine
Views
External Resources
Synonyms
Programmability
Service Broker
Storage
Security
DMS
EmployeeDb
EmployeeDBA
EMPTC
MOMO
MOMOTEST
MVC
mydemo7
petFarm
RES
RMCV
Sample
SimpleCrudOperations
SOLID
ADMINRG-OTJEJDG....o.CustomerDetails
SQLQuery3.sql - A...M COMPUTERS (57))*
USE [Departments]
GO
SELECT [SN]
,[ItemId]
,[IemPrice]
,[QTY]
,[TAX]
FROM [dbo].[SellLine]
GO
select * from dbo.ItemCataloug
100 %
Results
Messages
ItemId ItemCode IemPrice ItemNote
1 1 T2444 25 item modified
2 2 T224 13 item modified
3 4 U888 25 item modified
Query executed successfully.
ADMINRG-OTJEJDG (15.0 RTM)
ADMINRG-OTJEJDG\DVM CO...
Departments
00:00:00
3 rows
Properties
Current connection parameters
Aggregate Status
Connection failures
Elapsed time 00:00:00.029
Finish time 8/15/2021 1:29:04 PM
Name ADMINRG-OTJEJDG
Rows returned 3
Start time 8/15/2021 1:29:04 PM
State Open
Connection
Connection name ADMINRG-OTJEJDG (AD
Connection Details
Connection elapse 00:00:00.029
Connection encryp Not encrypted
Connection finish t 8/15/2021 1:29:04 PM
Connection rows re 3
Connection start ti 8/15/2021 1:29:04 PM
Connection state Open
Display name ADMINRG-OTJEJDG
Login name ADMINRG-OTJEJDG\DV
Server name ADMINRG-OTJEJDG
Server version 15.0.2000
Session Tracing ID
SPID 57
Name
The name of the connection.


Postman
File Edit View Help
Home Workspaces Reports Explore
Search Postman
Invite
Upgrade
My Workspace
New Import
POST http://localhost:5...
No Environment
Collections
APIs
Environments
Mock Servers
Monitors
History
New Collection
New Collection
New Collection
http://localhost:50449/api/Items
Save
POST
http://localhost:50449/api/Items
Send
Params Authorization Headers (8) Body Pre-request Script Tests Settings
Cookies
none form-data x-www-form-urlencoded raw binary GraphQL JSON
Beautify
{
"ItemCode": "T22425",
"IemPrice": 13.0,
"ItemNote": "item modified"
}
Body Cookies Headers (5) Test Results
200 OK 4.95 s 199 B
Save Response
Pretty Raw Preview Visualize JSON
"Record Posted!!"
Find and Replace
Console
Bootcamp
Runner
Trash



After that there is Class called “Order.cs” to Query the operations which comes from front end application,

It has below properties,

```
public int orderId { get; set; }
        public int orderNo { get; set; }
        public int CustId { get; set; }
        public decimal FinalValue { get; set; }
```

By using this class and controller according with that, An User will able to perform all crud operations as per their need,

Here I have highlighted the field “`FinalValue`” that is taken from front end application,

To get this it has been asked me to make 3 calculations, I have written a query to get these values like below,

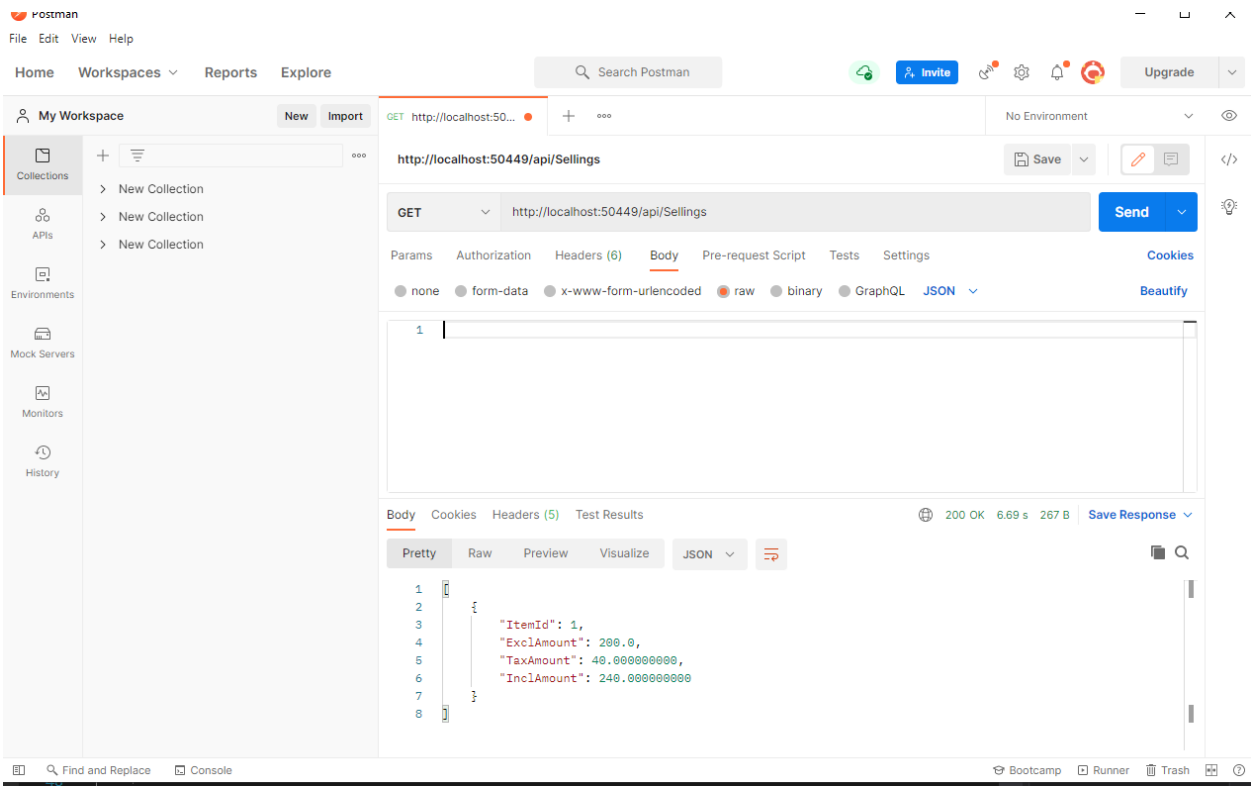
```
SELECT ItemId,
                  IemPrice *QTY   AS ExclAmount,(IemPrice * QTY)/ TAX   AS
TaxAmount,(IemPrice * QTY + (IemPrice * QTY) / TAX )  AS InclAmount
                  FROM dbo.SellLine"
```

By using this query an user will able to calculate the,

Excl Amount = Quantity * Price

Tax Amount = Excl Amount * Tax Rate /100

Incl Amount = Excl Amount + Tax Amount values and display them, below screen shot is taken from when this is running,



So Final value variable in my model class should get the value "`InclAmount`".

But Since I have no front end application, it is harder to show POST method here,

So I have wriiten GET method to calculate these functions and they Can see under “SellingsController.cs”

```
public class SellingsController : ControllerBase
{
    private readonly IConfiguration _configuration;
    0 references
    public SellingsController(IConfiguration configuration)
    {
        _configuration = configuration;
    }


    [HttpGet]
    0 references
    public JsonResult Get()
    {
        string query = @"
                SELECT ItemId,
                IemPrice *QTY   AS ExclAmount,(IemPrice * QTY)/ TAX   AS TaxAmount,(I
                FROM dbo.SellLine"
                ;
        DataTable table = new DataTable();
        string sqlDataSource = _configuration.GetConnectionString("MyConStr");
        SqlDataReader myReader;
        using (SqlConnection myCon = new SqlConnection(sqlDataSource))
        {
            myCon.Open();
            using (SqlCommand myCommand = new SqlCommand(query, myCon))
            {
                myReader = myCommand.ExecuteReader();
                table.Load(myReader);
                myReader.Close();
                myCon.Close();

            }
        }
        return new JsonResult(table);

    }
}
```

The logic what I used here is, the class file called “SellLine.cs” takes below properties,

```
public int SN { get; set; }
        public int ItemId { get; set; }
        public decimal IemPrice { get; set; }
        public int QTY { get; set; }
        public int TAX { get; set; }
```

By using these properties we will able to calculate those 3 equations and we need to pass that value to front end application, (The API method is for this function is OrderController.cs)

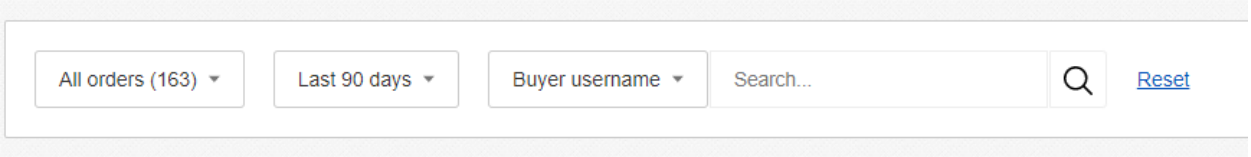
****

Conclusion!

We can enhance the performance of this application if we use below simple technique,

If we develop this system to get all required things from one end this will be a help for improve end user experience and also make our code easy too,

Below I have attached sample design.



This will help to avoid fill long tables and also make nice look and feel too.

Thank you so much!

Akila Udana….