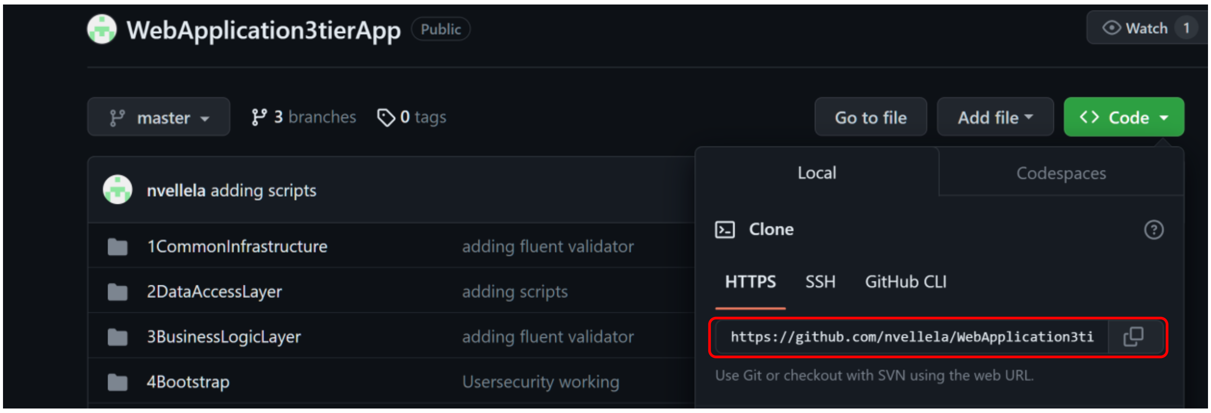
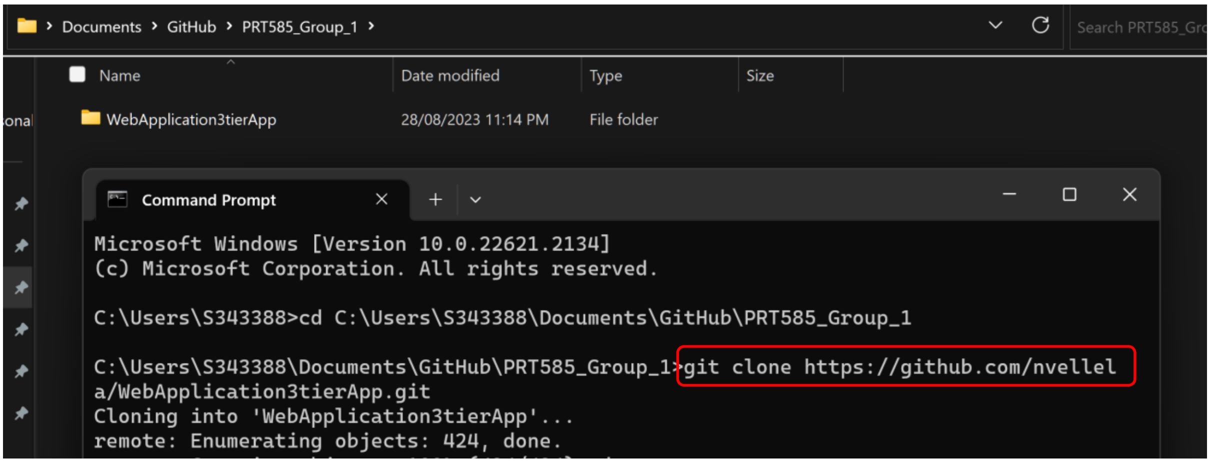
Run 3-Tier Web Application

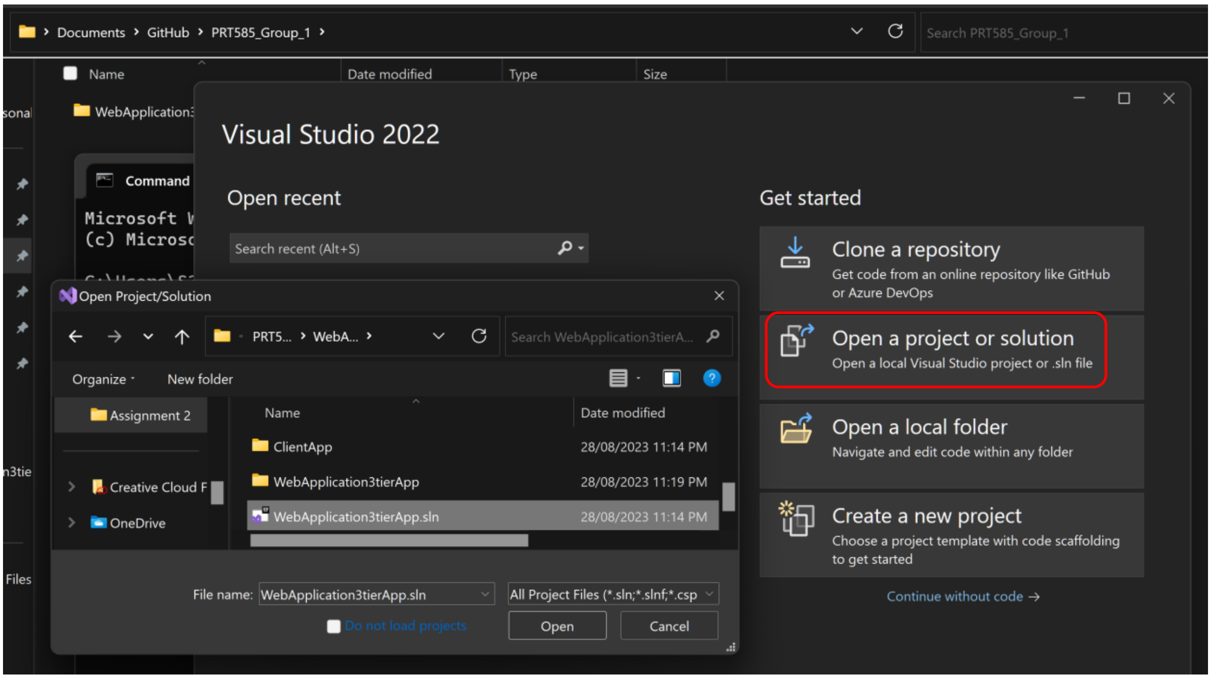
1. Clone the repository.
   https://github.com/nvellela/WebApplication3tierApp



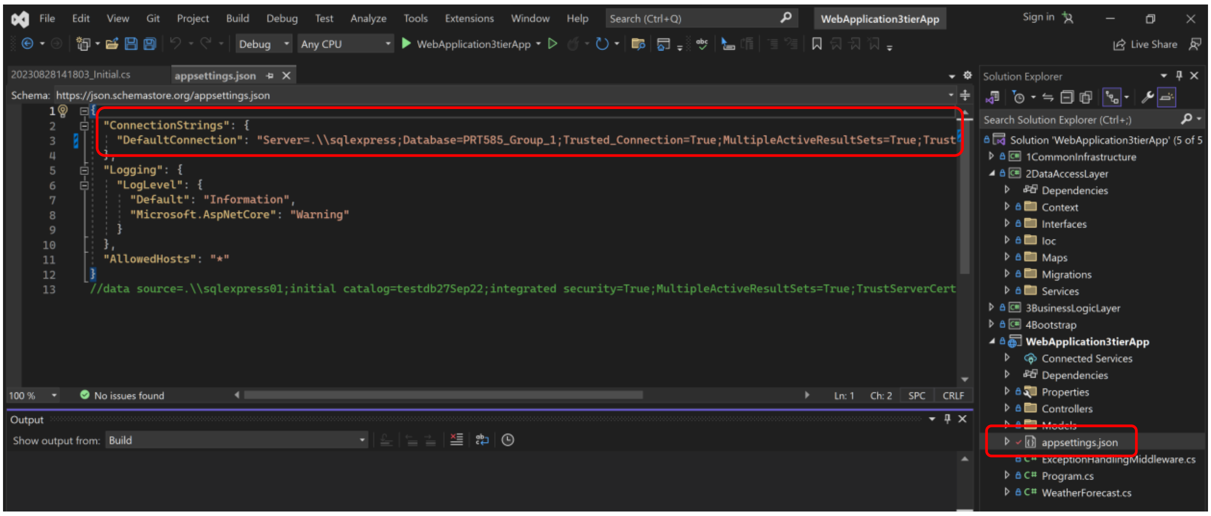


```
Microsoft Windows [Version 10.0.22621.2134]
(c) Microsoft Corporation. All rights reserved.

C:\Users\S343388>cd C:\Users\S343388\Documents\GitHub\PRT585_Group_1

C:\Users\S343388\Documents\GitHub\PRT585_Group_1>git clone https://github.com/nvellel
a/WebApplication3tierApp.git
Cloning into 'WebApplication3tierApp'...
remote: Enumerating objects: 424, done.
```

2. Open Visual Studio and open the 3-Tier Web Application project.

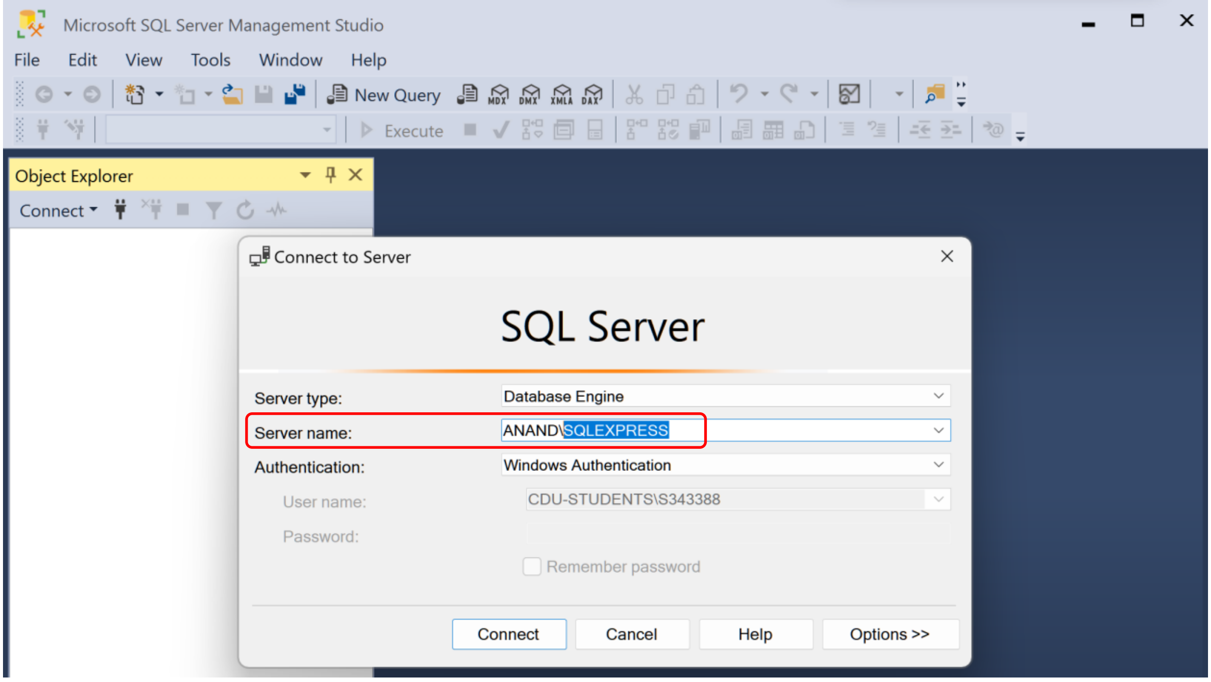


3. Open “**appsettings.json**” and update the connection string.
   - The server’s name in the connection string should match the SQL server name.
   - Update the database name. For example, PRT585_Group _1



Schema: https://json.schemastore.org/appsettings.json

```
{
  "ConnectionStrings": {
    "DefaultConnection": "Server=.\\sqlexpress;Database=PRT585_Group_1;Trusted_Connection=True;MultipleActiveResultSets=True;Trust
  },
  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft.AspNetCore": "Warning"
    }
  },
  "AllowedHosts": "*"
}
//data source=.\\sqlexpress01;initial catalog=testdb27Sep22;integrated security=True;MultipleActiveResultSets=True;TrustServerCert
```



Connect to Server

SQL Server

Server type: Database Engine

Server name: ANAND\SQLEXPRESS

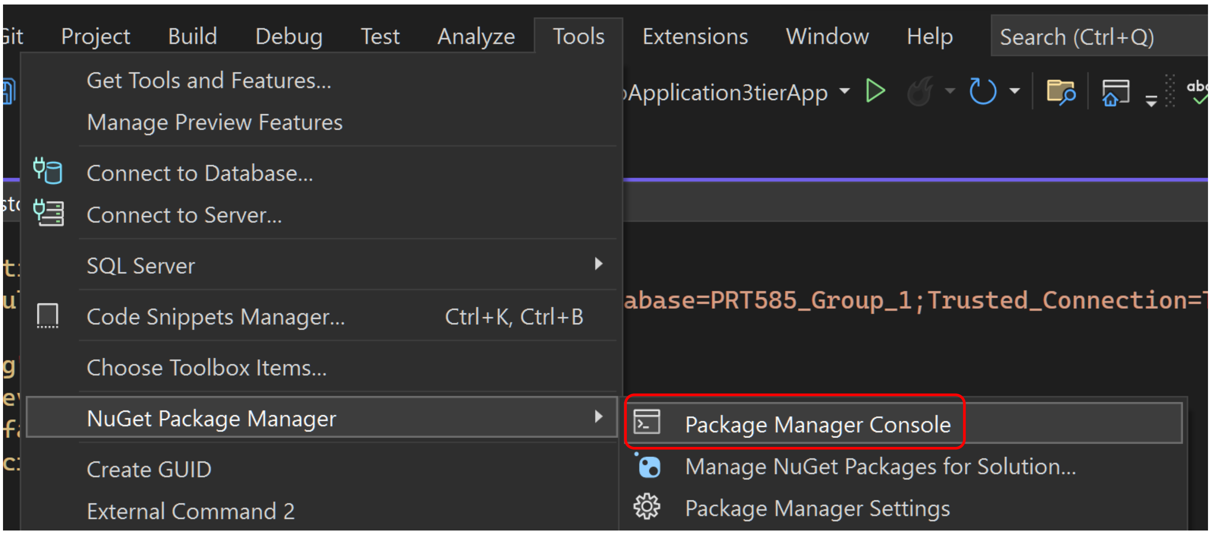
Authentication: Windows Authentication

User name: CDU-STUDENTS\S343388

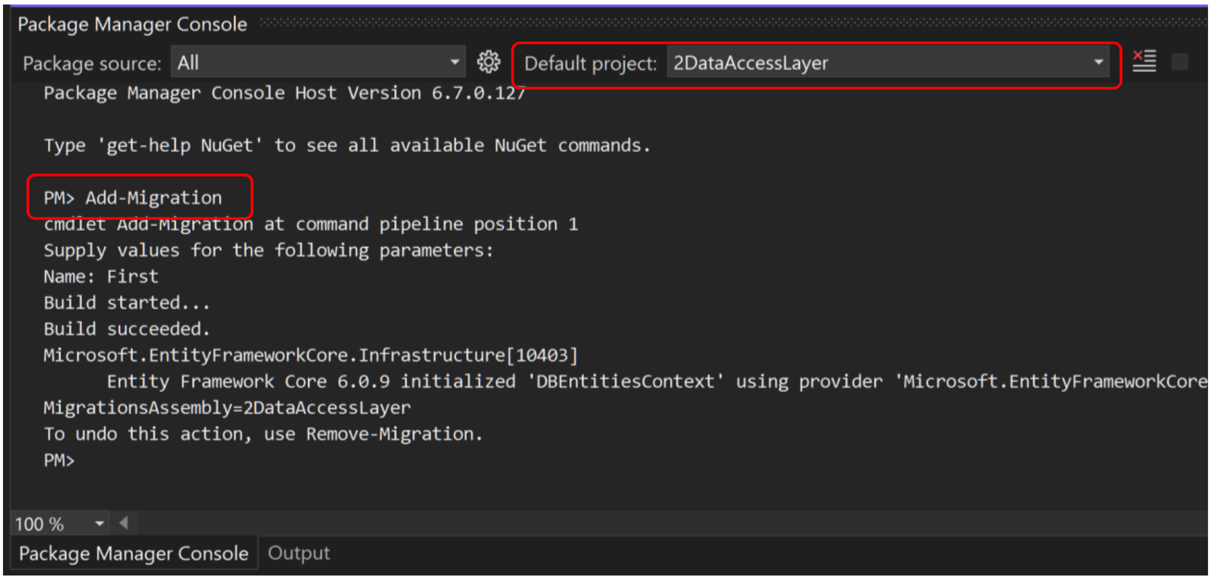
Password:

□ Remember password

4. Go to Tools > NuGet Package Manager > **Package Manager Console**

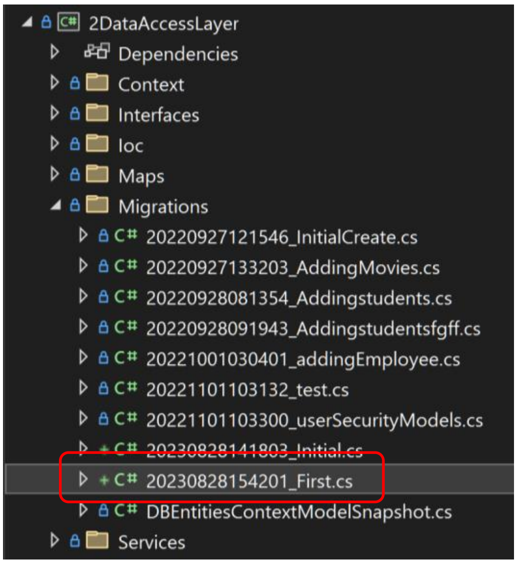


5. Add a new migration using the syntax "**Add-Migration**" in the Data Access Layer (DAL).



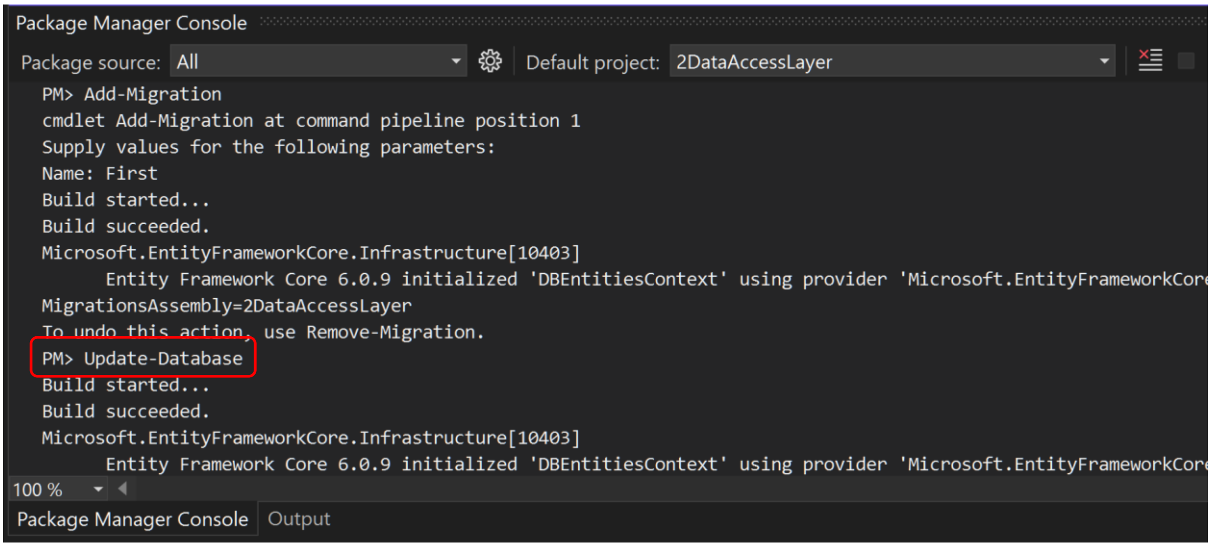
```
Package Manager Console Host Version 6.7.0.127

Type 'get-help NuGet' to see all available NuGet commands.

PM> Add-Migration
cmdlet Add-Migration at command pipeline position 1
Supply values for the following parameters:
Name: First
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCore
MigrationsAssembly=2DataAccessLayer
To undo this action, use Remove-Migration.
PM>
```

A new migration is added.



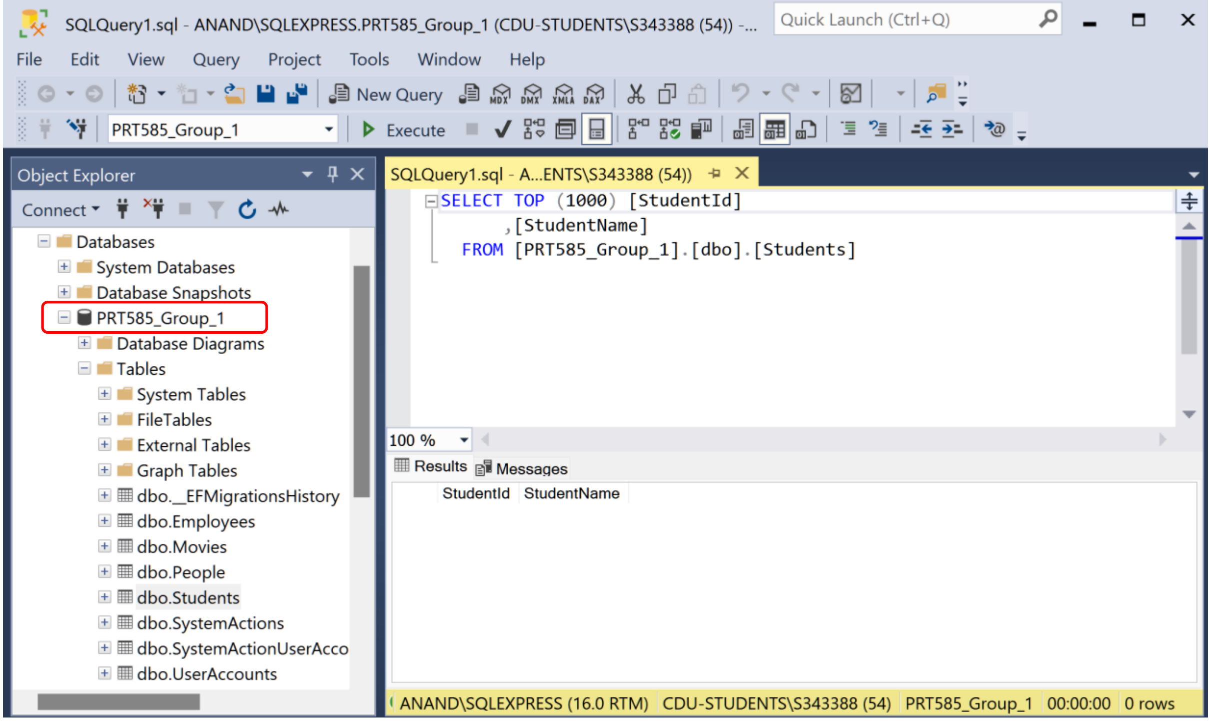
```
2DataAccessLayer
    Dependencies
    Context
    Interfaces
    Ioc
    Maps
    Migrations
        20220927121546_InitialCreate.cs
        20220927133203_AddingMovies.cs
        20220928081354_Addingstudents.cs
        20220928091943_Addingstudentsfgff.cs
        20221001030401_addingEmployee.cs
        20221101103132_test.cs
        20221101103300_userSecurityModels.cs
        20230828141803_Initial.cs
        20230828154201_First.cs
        DBEntitiesContextModelSnapshot.cs
    Services
```

6. Use the syntax “**Update-Database**” to update the database in the SQL server.



```
PM> Add-Migration
cmdlet Add-Migration at command pipeline position 1
Supply values for the following parameters:
Name: First
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCor
MigrationsAssembly=2DataAccessLayer
To undo this action, use Remove-Migration.
PM> Update-Database
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCor
```

This will update the database to the specified migration.

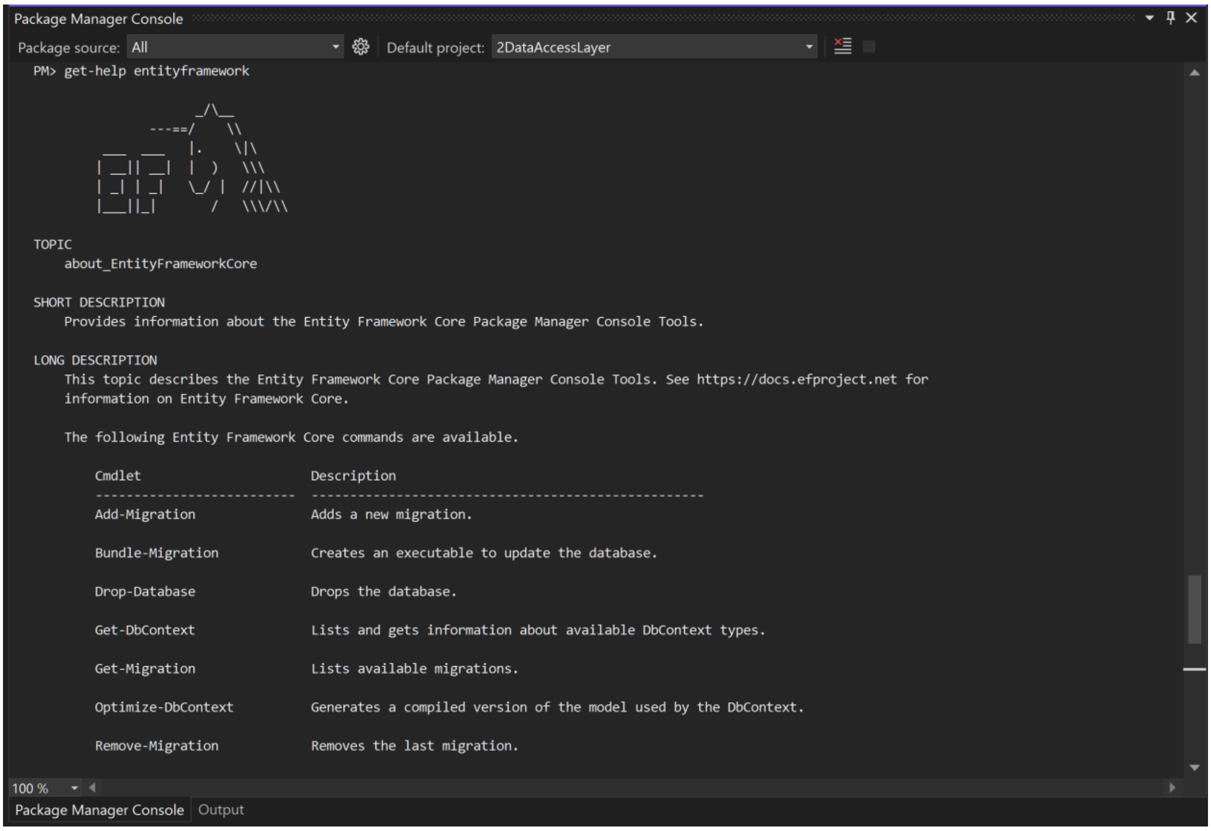


```
SELECT TOP (1000) [StudentId]
      ,[StudentName]
  FROM [PRT585_Group_1].[dbo].[Students]
```

StudentId	StudentName

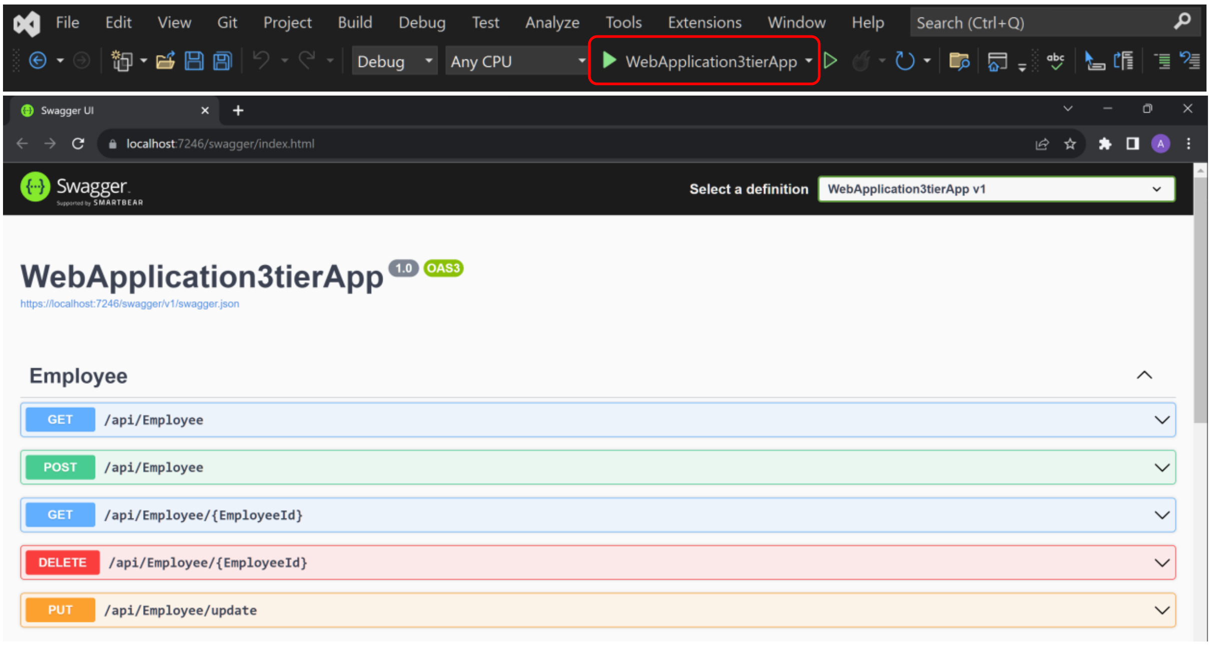
ANAND\SQLEXPRESS (16.0 RTM) | CDU-STUDENTS\S343388 (54) | PRT585_Group_1 | 00:00:00 | 0 rows

7. Use the syntax “**get-help entityframework**” to see other commands to see Package Manager Console (PMC) commands for migration.
   - Use the syntax “**Remove-Migration**” to delete the last migration.
   - Use the syntax “**Drop-Database**” to delete the database.

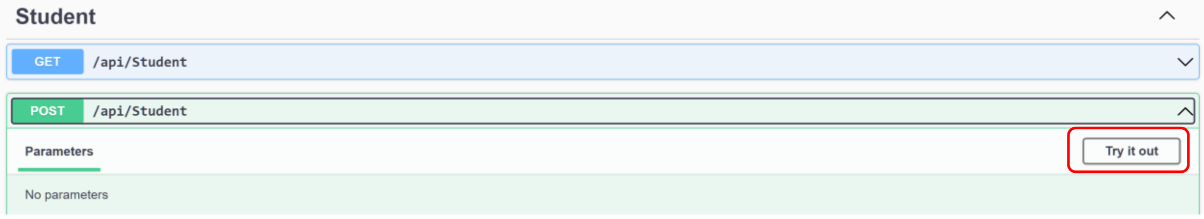


```
PM> get-help entityframework

                     _/\__
               ---==/    \\
         ___  ___   |.    \|\
        | __|| __|  |  )   \\\
        | _| | _|   \_/ |  //|\\
        |___||_|       /   \\\/\\

TOPIC
    about_EntityFrameworkCore

SHORT DESCRIPTION
    Provides information about the Entity Framework Core Package Manager Console Tools.

LONG DESCRIPTION
    This topic describes the Entity Framework Core Package Manager Console Tools. See https://docs.efproject.net for
    information on Entity Framework Core.

    The following Entity Framework Core commands are available.

        Cmdlet                      Description
        --------------------------  ---------------------------------------------------
        Add-Migration               Adds a new migration.

        Bundle-Migration            Creates an executable to update the database.

        Drop-Database               Drops the database.

        Get-DbContext               Lists and gets information about available DbContext types.

        Get-Migration               Lists available migrations.

        Optimize-DbContext          Generates a compiled version of the model used by the DbContext.

        Remove-Migration            Removes the last migration.
```

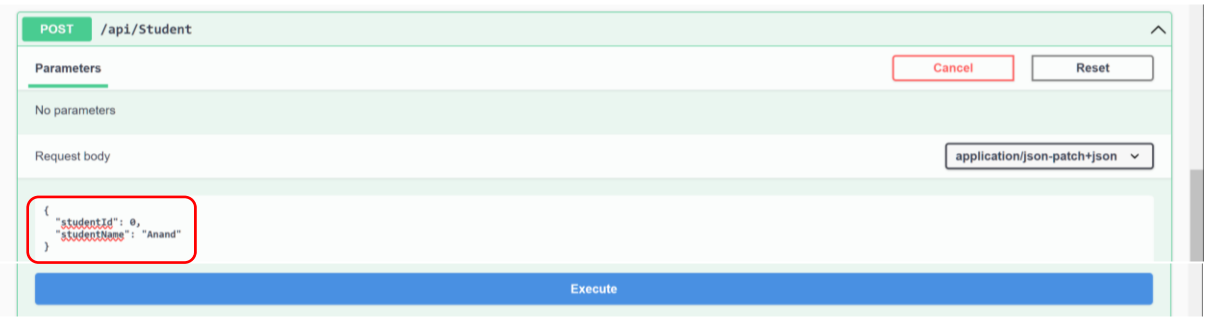
8. Run the app.



9. Make a request.
   - Expand the **POST Student** endpoint.
   - Click **Try it out**.

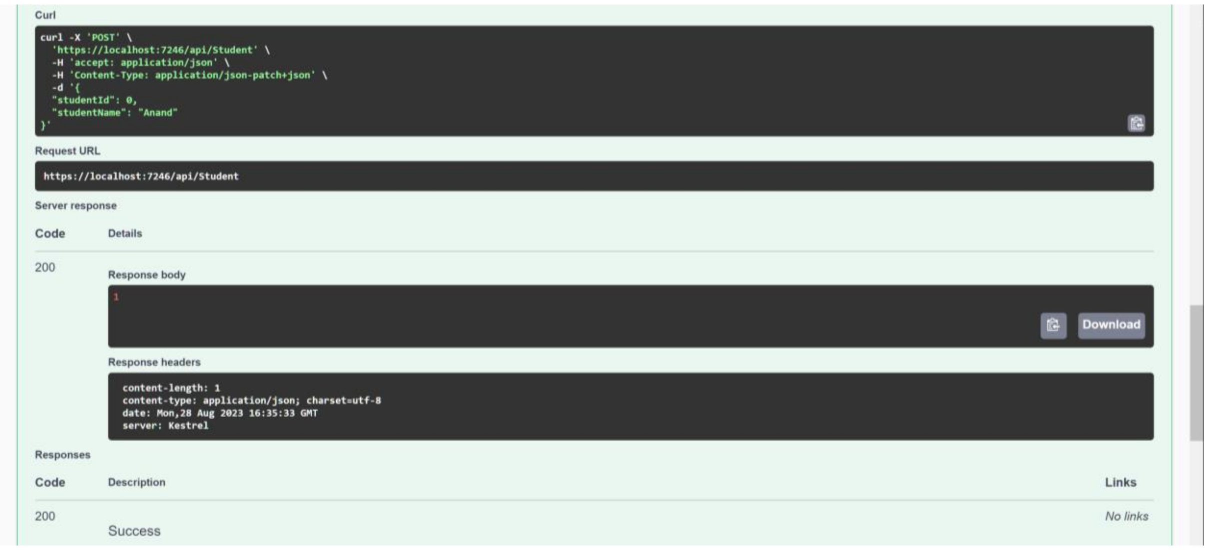


- Change the example value (**studentName**) in the Request Body field.
- Click **Execute**.



```
{
  "studentId": 0,
  "studentName": "Anand"
}
```

Swagger UI submits the request and shows the response.



```
curl -X 'POST' \
  'https://localhost:7246/api/Student' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json-patch+json' \
  -d '{
  "studentId": 0,
  "studentName": "Anand"
}'
```

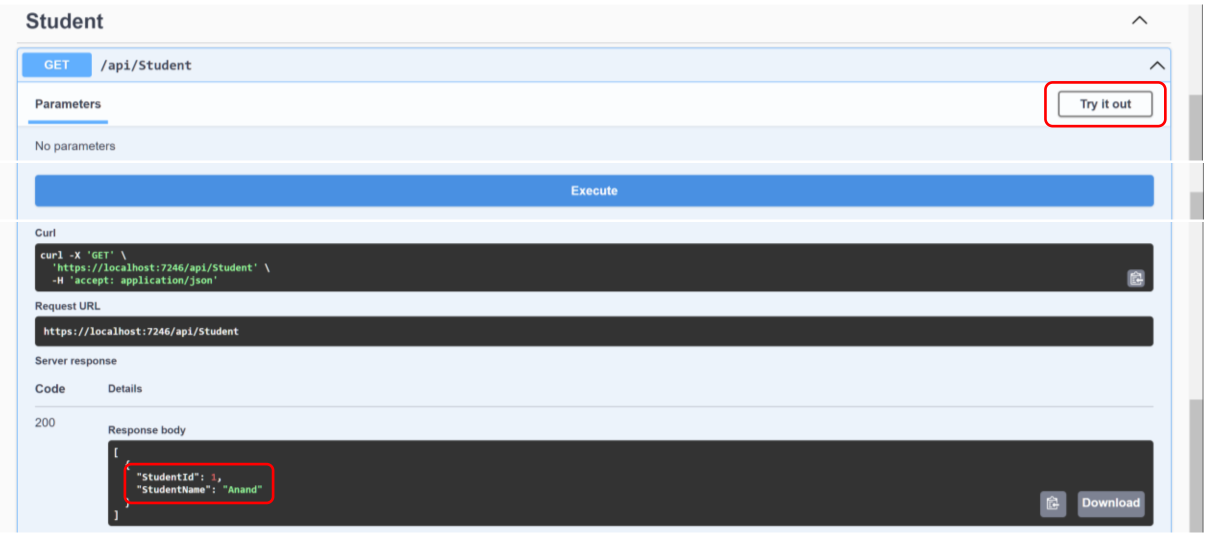
```
https://localhost:7246/api/Student
```

Code	Details
200	Response body: `1` Response headers: content-length: 1 content-type: application/json; charset=utf-8 date: Mon,28 Aug 2023 16:35:33 GMT server: Kestrel

Code	Description	Links
200	Success	No links

10. Verify that the student was created.

- Expand the **GET Student** endpoint.
- Click **Try it out**.
- Click **Execute**.
- The new student’s name will be returned in the Response section.



No parameters

Curl

```
curl -X 'GET' \
  'https://localhost:7246/api/Student' \
  -H 'accept: application/json'
```

Request URL

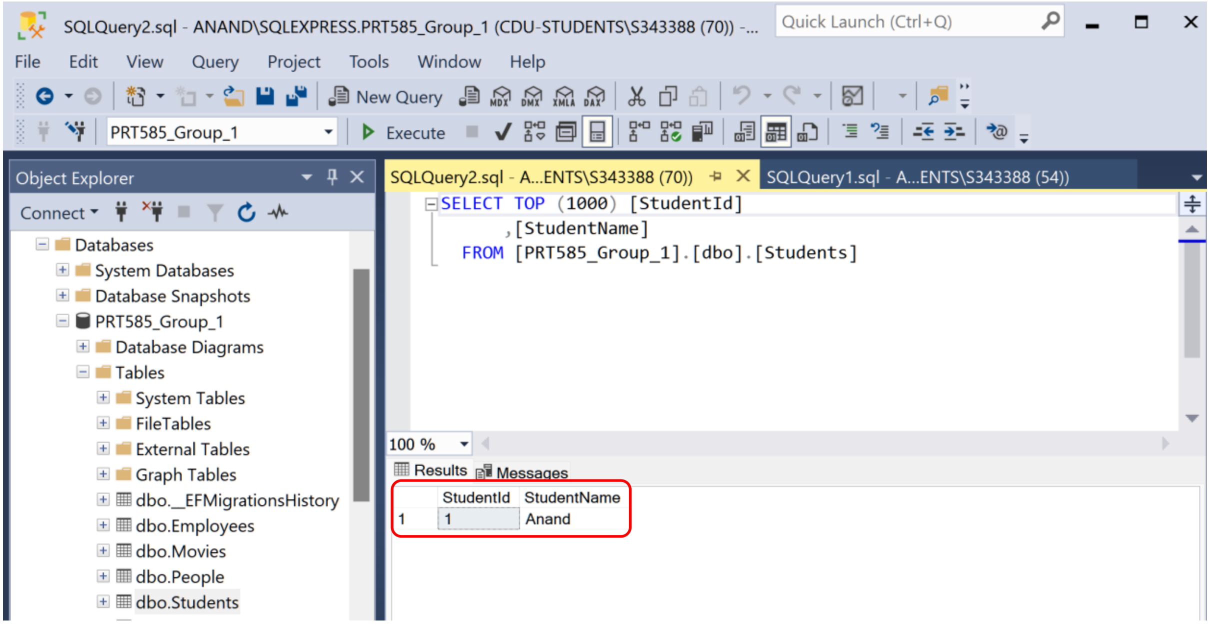
```
https://localhost:7246/api/Student
```

Server response

Code	Details
200	Response body

```
[
  {
    "StudentId": 1,
    "StudentName": "Anand"
  }
]
```

Check the database to see the new student.



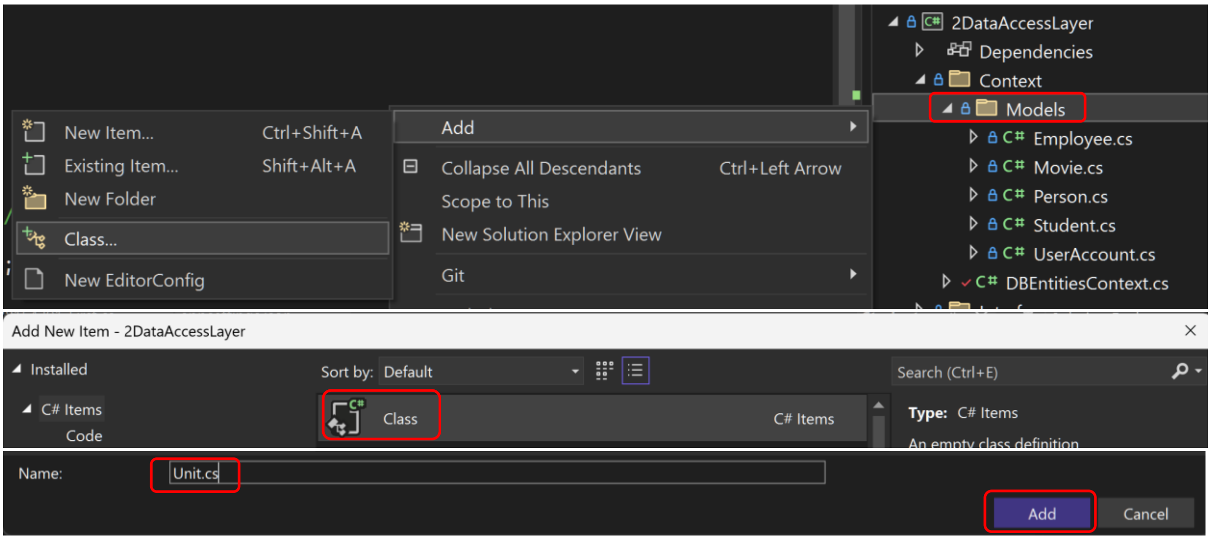
SQLQuery2.sql - ANAND\SQLEXPRESS.PRT585_Group_1 (CDU-STUDENTS\S343388 (70)) -...

```
SELECT TOP (1000) [StudentId]
      ,[StudentName]
  FROM [PRT585_Group_1].[dbo].[Students]
```

	StudentId	StudentName
1	1	Anand

Steps to add end-to-end.

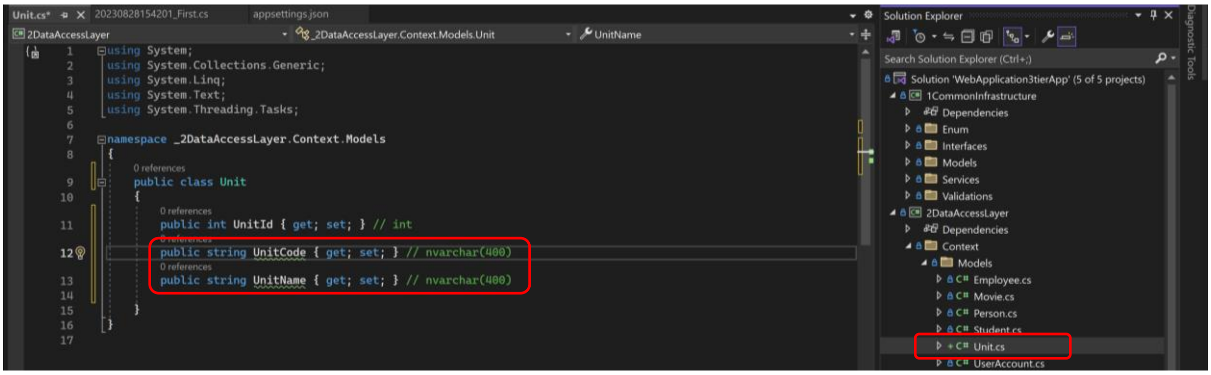
1. Add a new entity class in the Models folder under the Data Access Layer.
   - Right-click on the Models folder.
   - Add > **Class**
   - Change the name to "**Unit.cs**" and click **Add**.



Name: Unit.cs

2. Add the following -

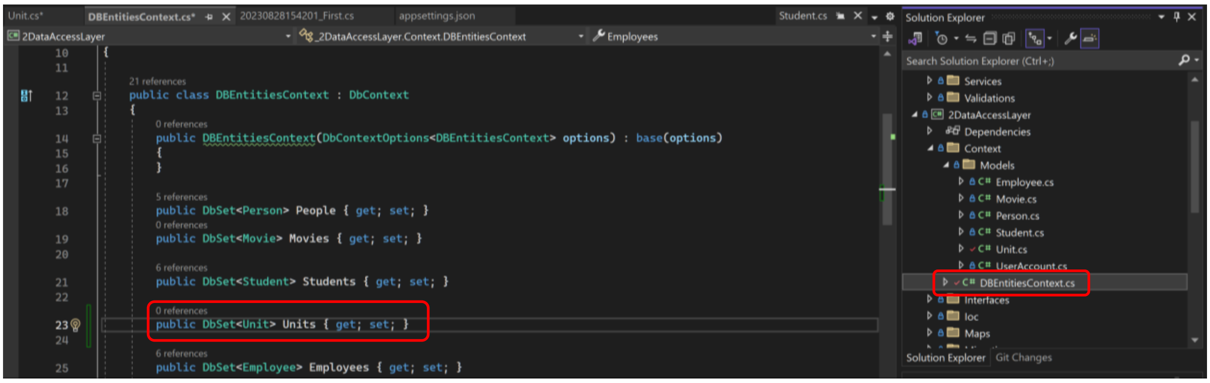
```
public class Unit
{
    public int UnitId { get; set; } // int
    public string UnitCode { get; set; } // nvarchar(400)
    public string UnitName { get; set; } // nvarchar(400)
}
```



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

namespace _2DataAccessLayer.Context.Models
{
    public class Unit
    {
        public int UnitId { get; set; } // int
        public string UnitCode { get; set; } // nvarchar(400)
        public string UnitName { get; set; } // nvarchar(400)

    }
}
```

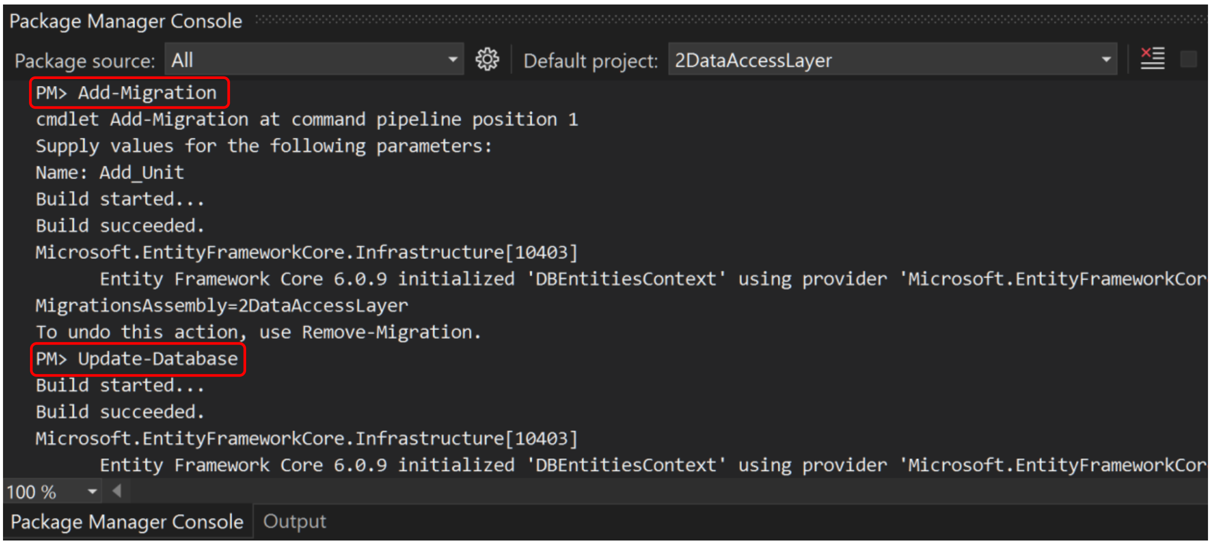
3. Add context.

```
public DbSet<Unit> Units { get; set; }
```

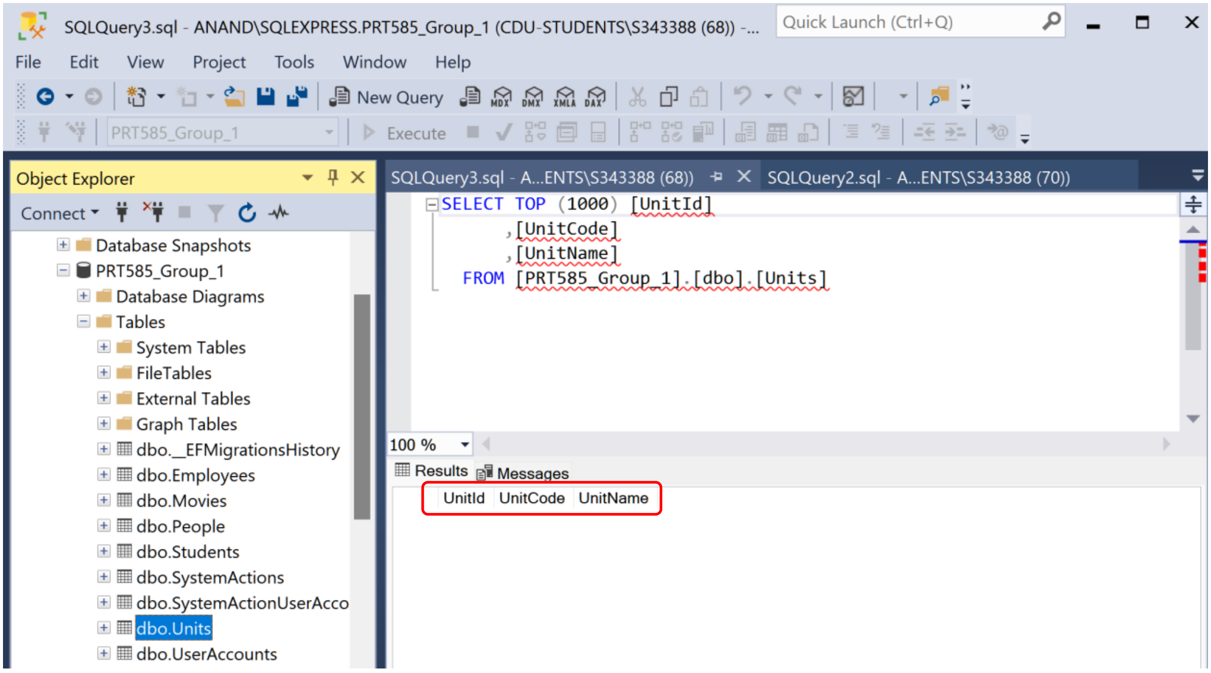


```
{

    21 references
    public class DBEntitiesContext : DbContext
    {
        0 references
        public DBEntitiesContext(DbContextOptions<DBEntitiesContext> options) : base(options)
        {
        }

        5 references
        public DbSet<Person> People { get; set; }
        0 references
        public DbSet<Movie> Movies { get; set; }

        6 references
        public DbSet<Student> Students { get; set; }

        0 references
        public DbSet<Unit> Units { get; set; }

        6 references
        public DbSet<Employee> Employees { get; set; }
```

4. Add a new migration and update the database.



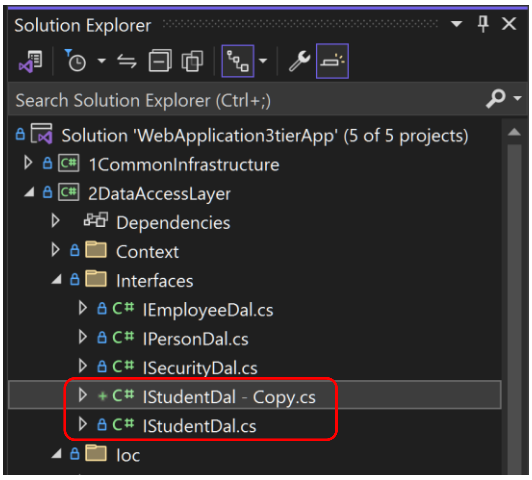
```
PM> Add-Migration
cmdlet Add-Migration at command pipeline position 1
Supply values for the following parameters:
Name: Add_Unit
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCor
MigrationsAssembly=2DataAccessLayer
To undo this action, use Remove-Migration.
PM> Update-Database
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Infrastructure[10403]
      Entity Framework Core 6.0.9 initialized 'DBEntitiesContext' using provider 'Microsoft.EntityFrameworkCor
```

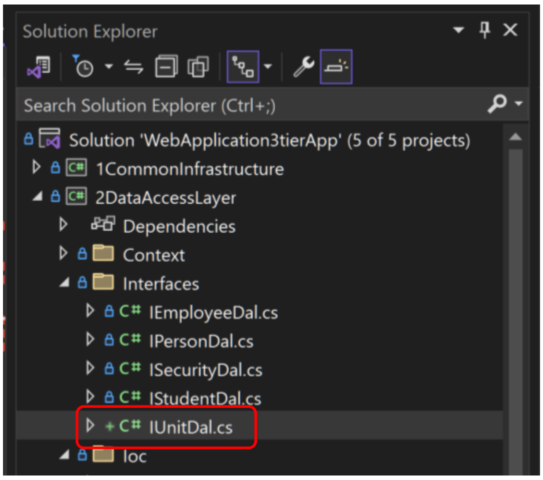


```
SELECT TOP (1000) [UnitId]
      ,[UnitCode]
      ,[UnitName]
  FROM [PRT585_Group_1].[dbo].[Units]
```

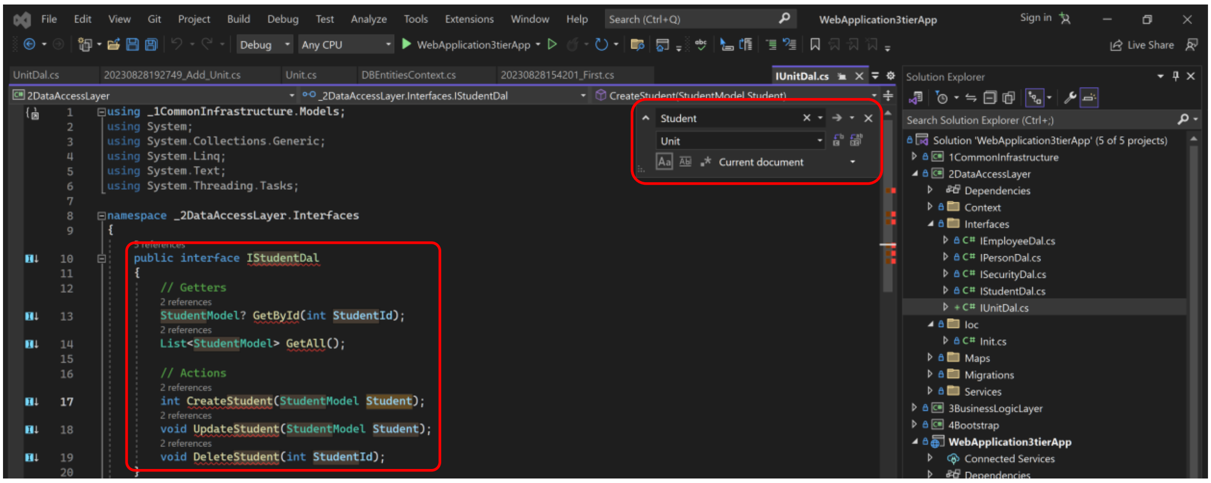
UnitId	UnitCode	UnitName

5. Create a copy of “**IStudentDal.cs**” in the Interfaces folder under DAL and rename it “**IUnitDal.cs**”.





Replace all the words “**Student**” with “**Unit**”.



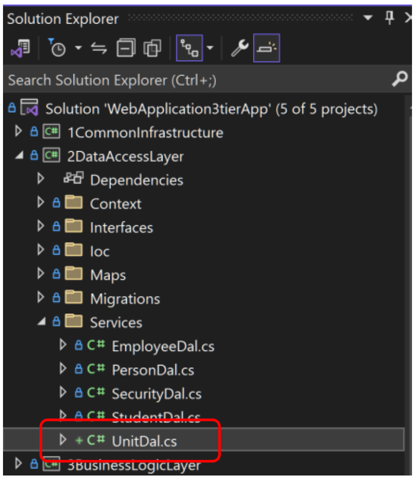
Find: Student

Replace: Unit

Current document

```
using _1CommonInfrastructure.Models;
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace _2DataAccessLayer.Interfaces
{
    public interface IStudentDal
    {
        // Getters
        StudentModel? GetById(int StudentId);
        List<StudentModel> GetAll();

        // Actions
        int CreateStudent(StudentModel Student);
        void UpdateStudent(StudentModel Student);
        void DeleteStudent(int StudentId);
```

6. Create a copy of “**StudentDal.cs**” in the Services folder under DAL and rename it “**UnitDal.cs**”.

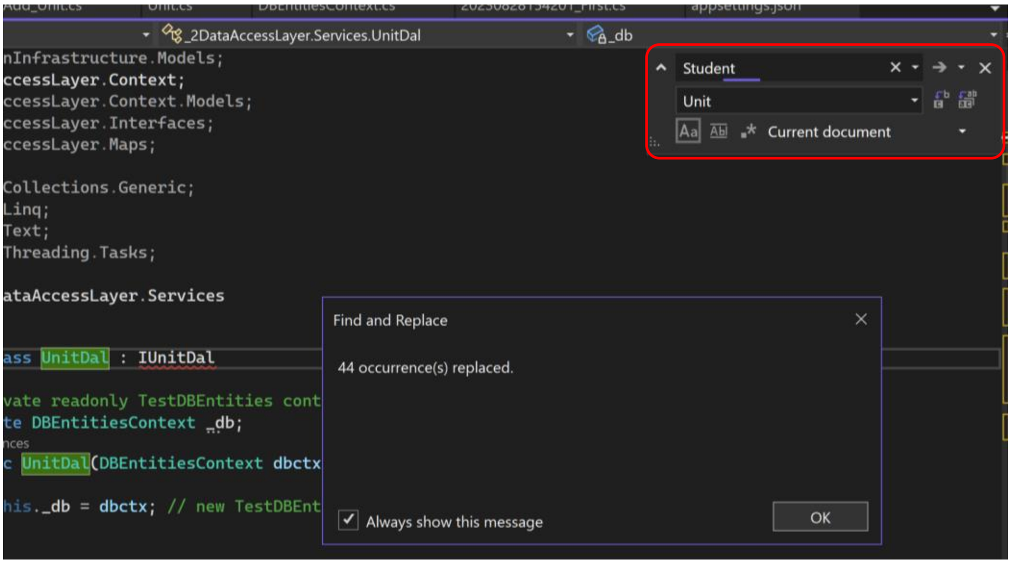


Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'WebApplication3tierApp' (5 of 5 projects)
  1CommonInfrastructure
  2DataAccessLayer
    Dependencies
    Context
    Interfaces
    Ioc
    Maps
    Migrations
    Services
      EmployeeDal.cs
      PersonDal.cs
      SecurityDal.cs
      StudentDal.cs
      UnitDal.cs
  3BusinessLogicLayer
```

Replace all the words “**Student**” with “**Unit**”.



_2DataAccessLayer.Services.UnitDal   _db

```
nInfrastructure.Models;
ccessLayer.Context;
ccessLayer.Context.Models;
ccessLayer.Interfaces;
ccessLayer.Maps;

Collections.Generic;
Linq;
Text;
Threading.Tasks;

ataAccessLayer.Services

ass UnitDal : IUnitDal

vate readonly TestDBEntities cont
te DBEntitiesContext _db;
nces
c UnitDal(DBEntitiesContext dbctx

his._db = dbctx; // new TestDBEnt
```

Find: Student

Replace: Unit

Current document

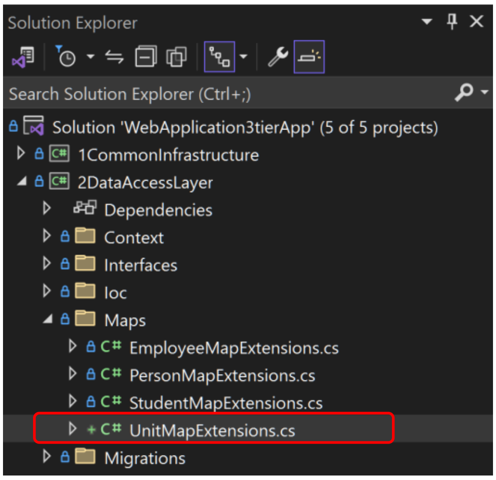
Find and Replace

44 occurrence(s) replaced.

☑ Always show this message

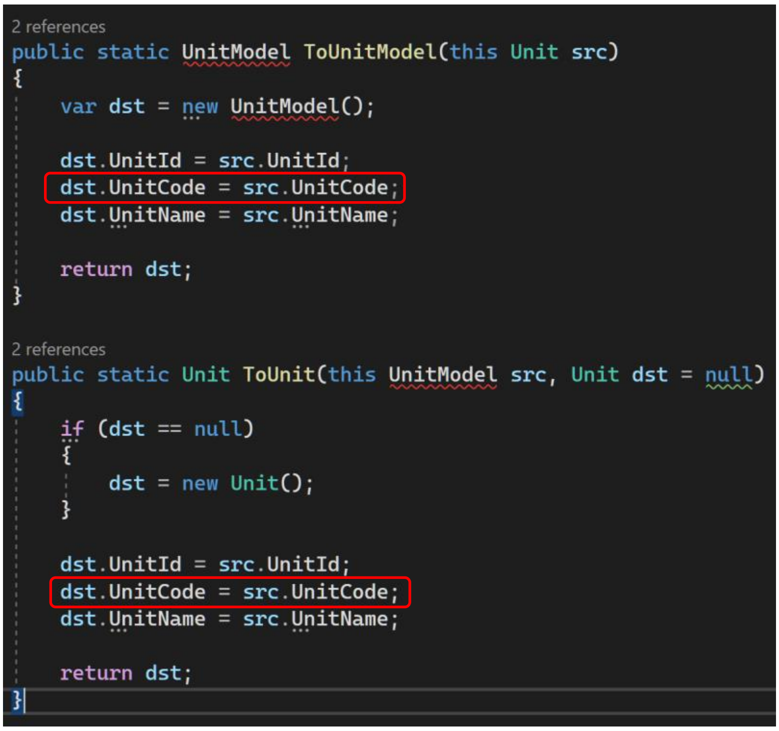
OK

7. Create a copy of “**StudentMapExtensions.cs**” in the Maps folder under DAL and rename it “**UnitMapExtensions.cs**”.



Replace all the words “**Student**” with “**Unit**”. And add the following -

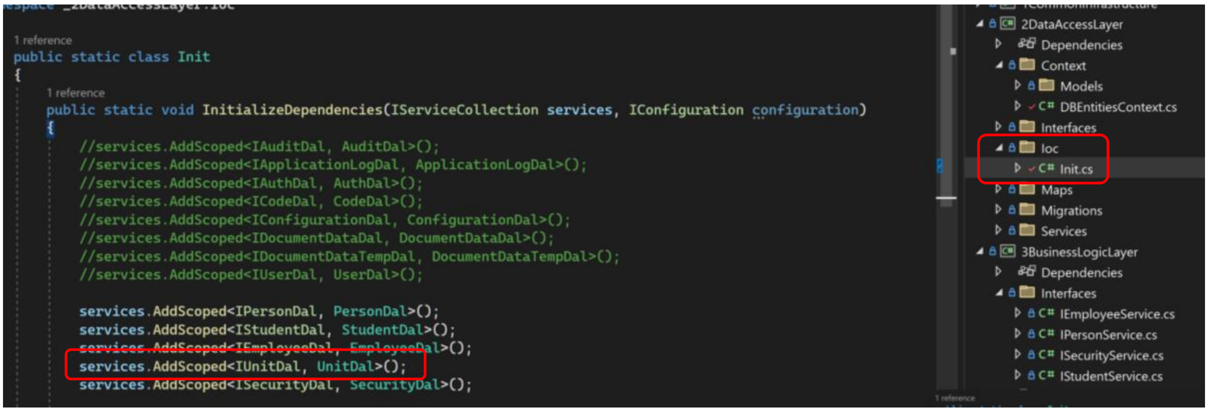
```
dst.UnitCode = src.UnitCode;
```



```
2 references
public static UnitModel ToUnitModel(this Unit src)
{
    var dst = new UnitModel();

    dst.UnitId = src.UnitId;
    dst.UnitCode = src.UnitCode;
    dst.UnitName = src.UnitName;

    return dst;
}

2 references
public static Unit ToUnit(this UnitModel src, Unit dst = null)
{
    if (dst == null)
    {
        dst = new Unit();
    }

    dst.UnitId = src.UnitId;
    dst.UnitCode = src.UnitCode;
    dst.UnitName = src.UnitName;

    return dst;
}
```

8. Open “**Init.cs**” from the Ioc folder under DAL. Add the following -

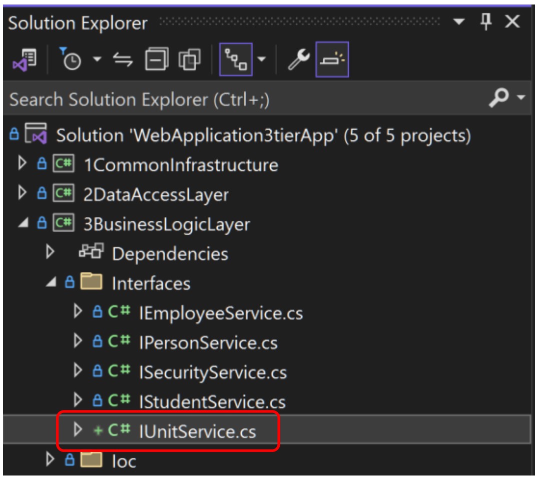
```
services.AddScoped<IUnitDal, UnitDal>();
```



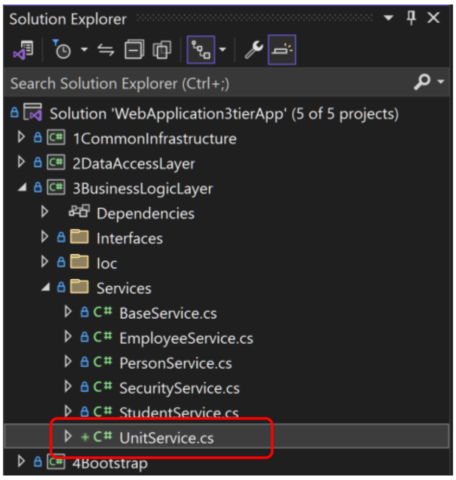
```
espace _2DataAccessLayer.Ioc

1 reference
public static class Init
{
    1 reference
    public static void InitializeDependencies(IServiceCollection services, IConfiguration configuration)
    {
        //services.AddScoped<IAuditDal, AuditDal>();
        //services.AddScoped<IApplicationLogDal, ApplicationLogDal>();
        //services.AddScoped<IAuthDal, AuthDal>();
        //services.AddScoped<ICodeDal, CodeDal>();
        //services.AddScoped<IConfigurationDal, ConfigurationDal>();
        //services.AddScoped<IDocumentDataDal, DocumentDataDal>();
        //services.AddScoped<IDocumentDataTempDal, DocumentDataTempDal>();
        //services.AddScoped<IUserDal, UserDal>();

        services.AddScoped<IPersonDal, PersonDal>();
        services.AddScoped<IStudentDal, StudentDal>();
        services.AddScoped<IEmployeeDal, EmployeeDal>();
        services.AddScoped<IUnitDal, UnitDal>();
        services.AddScoped<ISecurityDal, SecurityDal>();
```

9. Repeat the steps in the Business Logic Layer (BLL).

- Create a copy of “**IStudentService.cs**” in the Interfaces folder under BLL and rename it “**IUnitService.cs**”.
- Replace all the words “**Student**” with “**Unit**”.

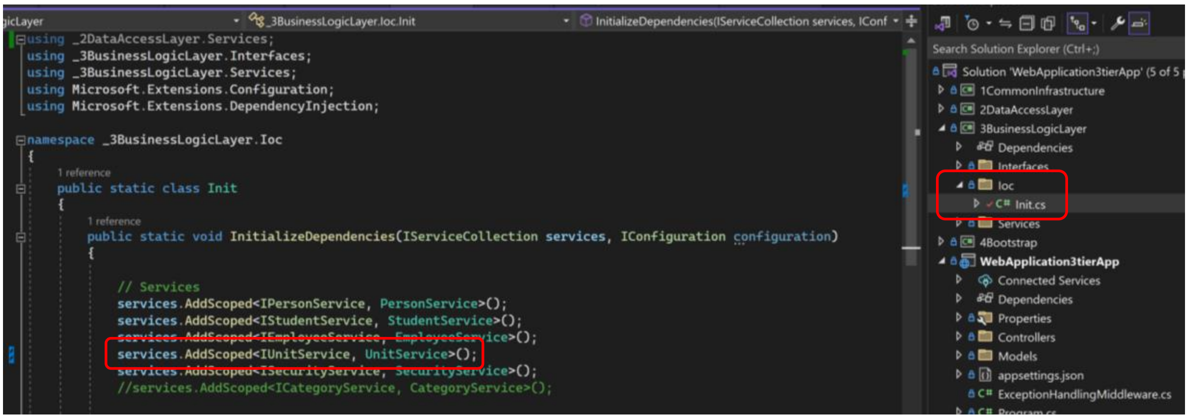


- Create a copy of “**StudentService.cs**” in the Services folder under BLL and rename it “**UnitService.cs**”.
- Replace all the words “**Student**” with “**Unit**”.



- Open “**Init.cs**” from the Ioc folder under BLL. Add the following -

  `services.AddScoped<IUnitService, UnitService>();`



```
using _2DataAccessLayer.Services;
using _3BusinessLogicLayer.Interfaces;
using _3BusinessLogicLayer.Services;
using Microsoft.Extensions.Configuration;
using Microsoft.Extensions.DependencyInjection;

namespace _3BusinessLogicLayer.Ioc
{
    1 reference
    public static class Init
    {
        1 reference
        public static void InitializeDependencies(IServiceCollection services, IConfiguration configuration)
        {

            // Services
            services.AddScoped<IPersonService, PersonService>();
            services.AddScoped<IStudentService, StudentService>();
            services.AddScoped<IEmployeeService, EmployeeService>();
            services.AddScoped<IUnitService, UnitService>();
            services.AddScoped<ISecurityService, SecurityService>();
            //services.AddScoped<ICategoryService, CategoryService>();
```

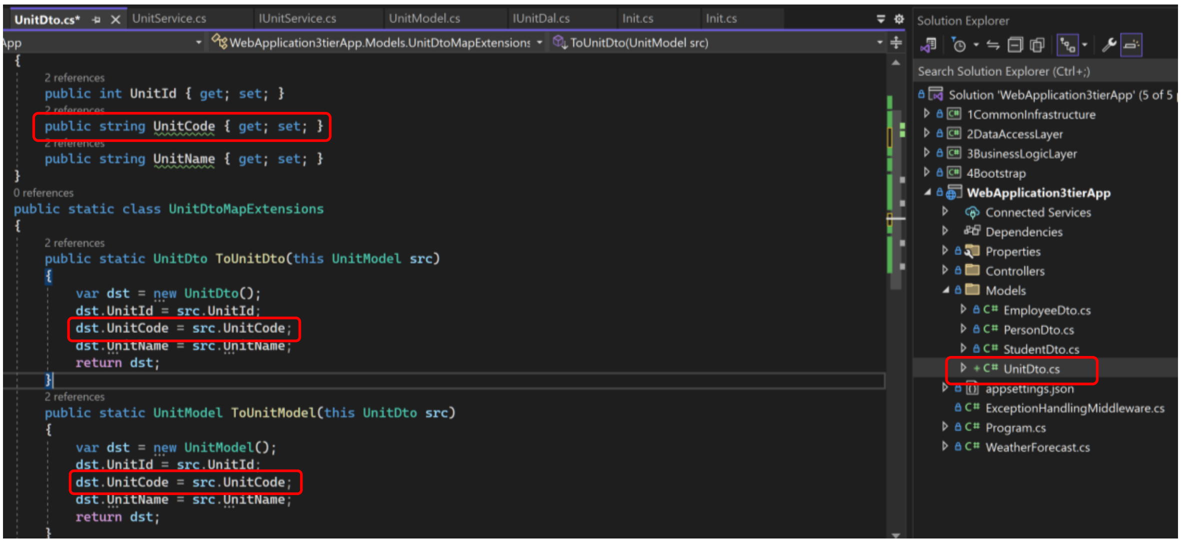
10.Create a copy of “**StudentDto.cs**” in the Models folder under WebApplication3tierApp and rename it “**UnitDto.cs**”.

- Replace all the words “**Student**” with “**Unit**”.
- Add the following in `public class UnitDto`

  `public string UnitCode { get; set; }`

- Add the following in `public static class UnitDtoMapExtensions`

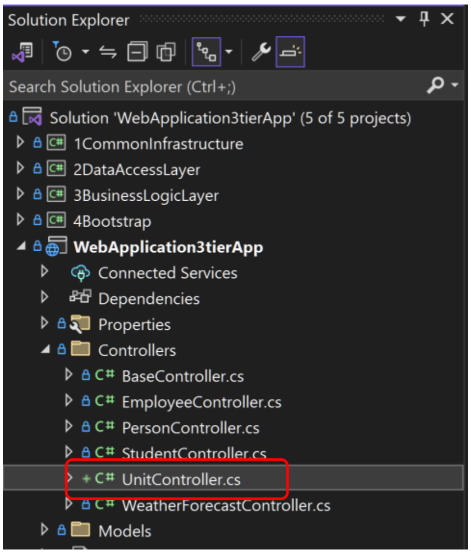
  `dst.UnitCode = src.UnitCode;`



```
{
    2 references
    public int UnitId { get; set; }
    2 references
    public string UnitCode { get; set; }
    2 references
    public string UnitName { get; set; }
}
0 references
public static class UnitDtoMapExtensions
{
    2 references
    public static UnitDto ToUnitDto(this UnitModel src)
    {
        var dst = new UnitDto();
        dst.UnitId = src.UnitId;
        dst.UnitCode = src.UnitCode;
        dst.UnitName = src.UnitName;
        return dst;
    }
    2 references
    public static UnitModel ToUnitModel(this UnitDto src)
    {
        var dst = new UnitModel();
        dst.UnitId = src.UnitId;
        dst.UnitCode = src.UnitCode;
        dst.UnitName = src.UnitName;
        return dst;
    }
```

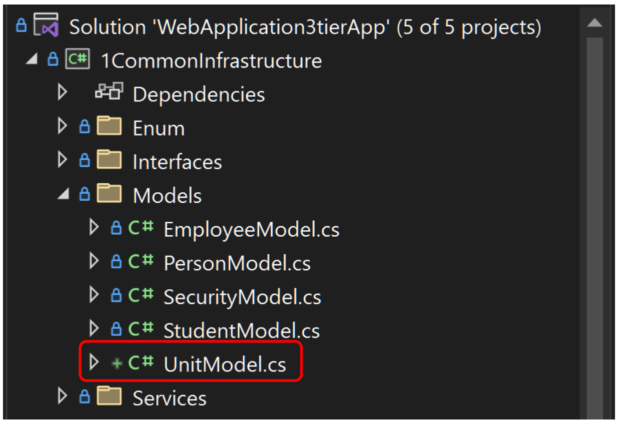
11. Create a copy of “**StudentController.cs**” in the Controllers folder under WebApplication3tierApp and rename it “**UnitController.cs**”.

    - Replace all the words “**Student**” with “**Unit**”.



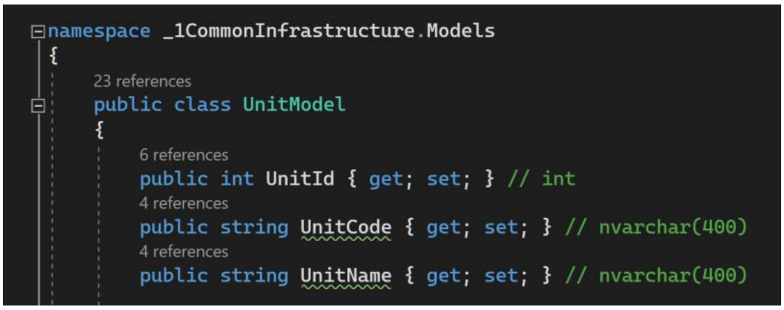
12. Create a copy of “**StudentModel.cs**” in the Models folder under the Common Infrastructure Layer

    and rename it “**UnitModel.cs**”.



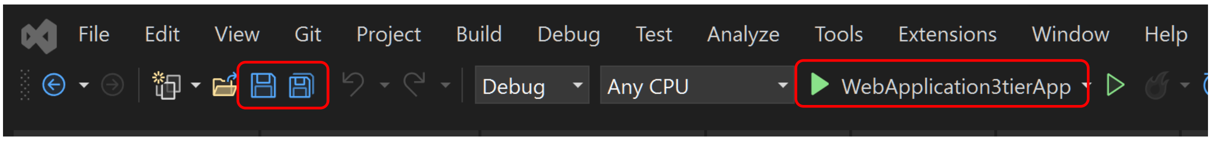
Add the following –

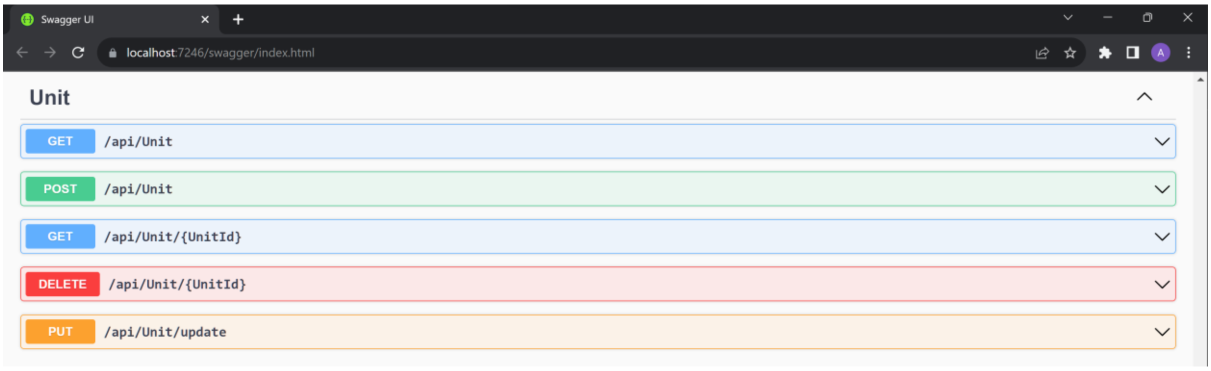
```
public class UnitModel
{
    public int UnitId { get; set; } // int
    public string UnitCode { get; set; } // nvarchar(400)
    public string UnitName { get; set; } // nvarchar(400)
}
```



```
namespace _1CommonInfrastructure.Models
{
    23 references
    public class UnitModel
    {
        6 references
        public int UnitId { get; set; } // int
        4 references
        public string UnitCode { get; set; } // nvarchar(400)
        4 references
        public string UnitName { get; set; } // nvarchar(400)
```

13.Save the project and run the app.





localhost:7246/swagger/index.html

Unit

GET /api/Unit

POST /api/Unit

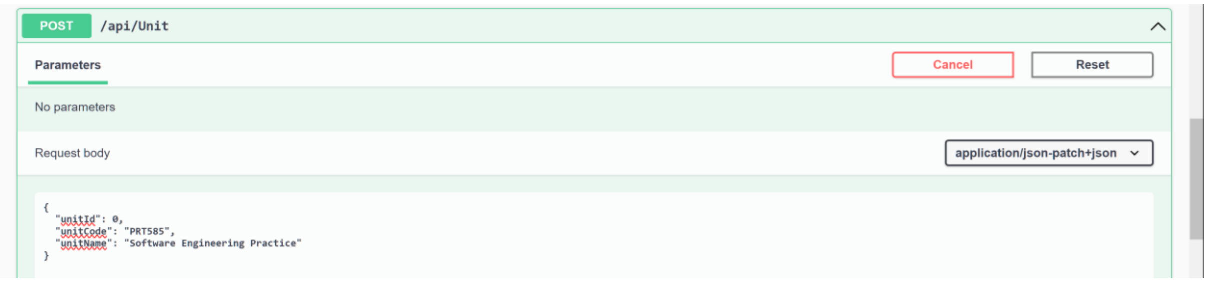
GET /api/Unit/{UnitId}

DELETE /api/Unit/{UnitId}

PUT /api/Unit/update

14.Make a request.

- Expand the **POST Student** endpoint.
- Click **Try it out**.
- Add unit code and unit name.
- Execute.



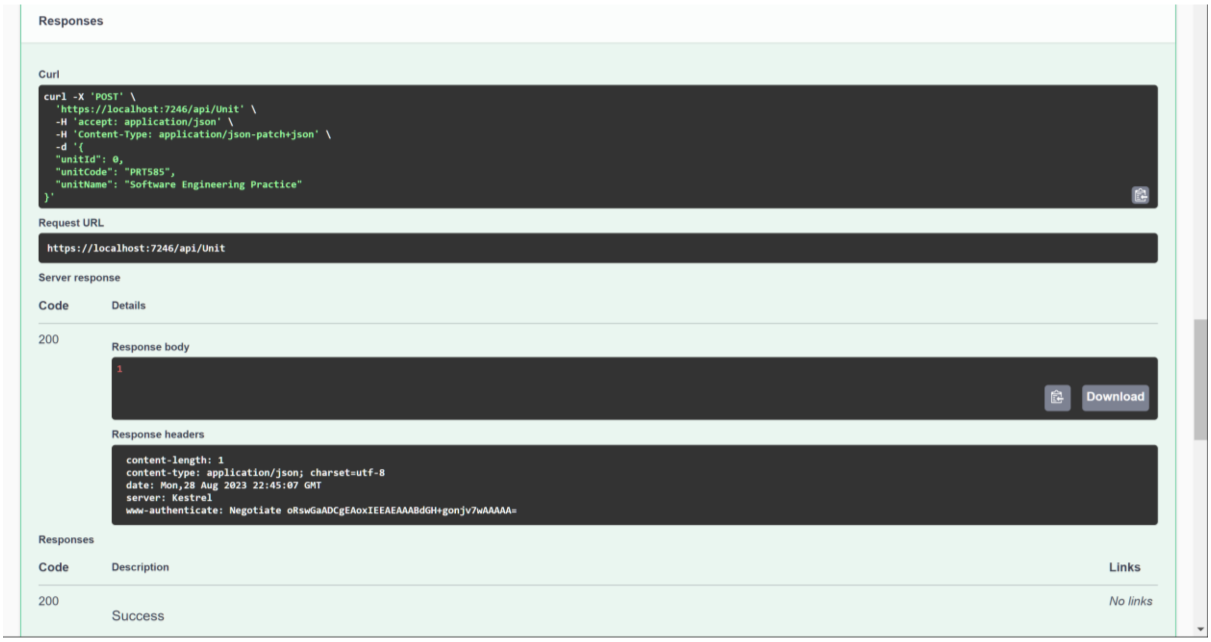
POST /api/Unit

Parameters

No parameters

Request body: application/json-patch+json

```
{
  "unitId": 0,
  "unitCode": "PRT585",
  "unitName": "Software Engineering Practice"
}
```



Responses

Curl

```
curl -X 'POST' \
  'https://localhost:7246/api/Unit' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json-patch+json' \
  -d '{
  "unitId": 0,
  "unitCode": "PRT585",
  "unitName": "Software Engineering Practice"
}'
```

Request URL

```
https://localhost:7246/api/Unit
```

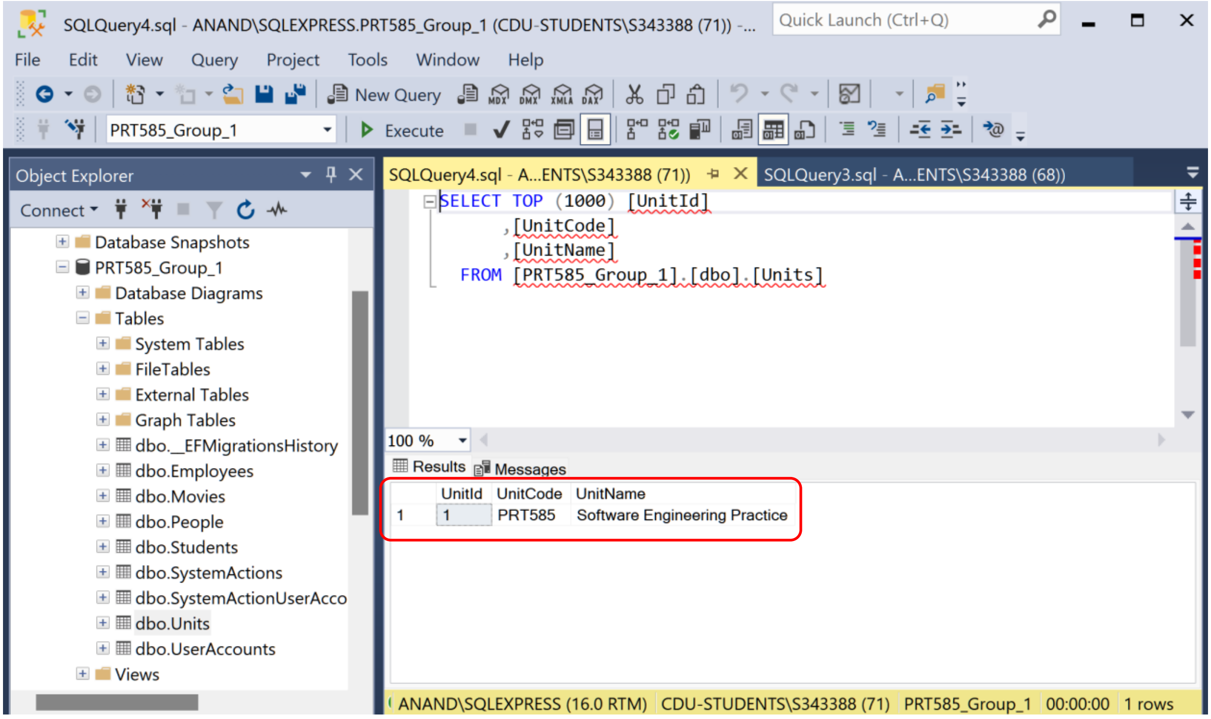
Server response

Code	Details
200	Response body 1 Response headers content-length: 1 content-type: application/json; charset=utf-8 date: Mon,28 Aug 2023 22:45:07 GMT server: Kestrel www-authenticate: Negotiate oRswGaADCgEAoxIEEAEAAABdGH+gonjv7wAAAAA=

Responses

Code	Description	Links
200	Success	*No links*

15.Check the database to see the new unit.



```sql
SELECT TOP (1000) [UnitId]
      ,[UnitCode]
      ,[UnitName]
  FROM [PRT585_Group_1].[dbo].[Units]
```

	UnitId	UnitCode	UnitName
1	1	PRT585	Software Engineering Practice

ANAND\SQLEXPRESS (16.0 RTM) | CDU-STUDENTS\S343388 (71) | PRT585_Group_1 | 00:00:00 | 1 rows

16.**Next step** – Use Angular on the client side to perform CRUD operations.

https://www.youtube.com/watch?v=CdE6rVfPJ9I&t=130s