ASP.NET Core MVC 3.1

Tahaluf Training Center 2021

شركـــة تحالــف الإمـــارات للحـــلـــول التـقـنـيــة ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Chapter 2

1 Generate ASP.NET Core MVC Model Classes

2 Route

3 CRUD Operation

4 Controller and View

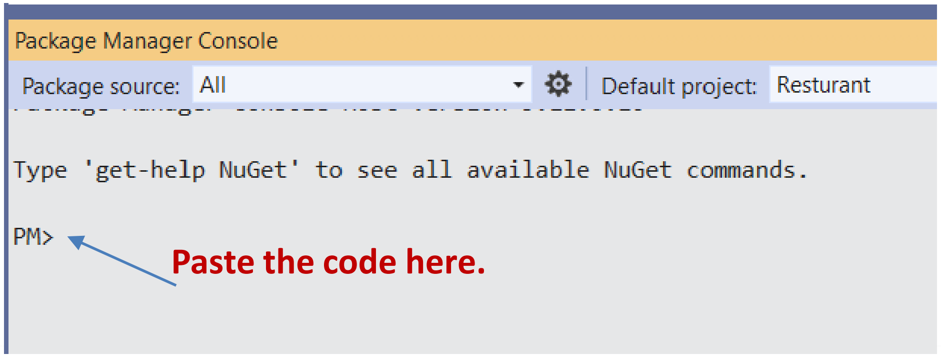
Generate ASP.NET Core MVC Model Classes

- Tools => NuGet Package Manager => Package Manager Console => Paste the following command:

Code:

```
Scaffold-DbContext "USER ID=****;PASSWORD=****;DATA
SOURCE=94.56.229.181:3488/traindb"
Oracle.EntityFrameworkCore -outputdir Models.
```

**** :put your username and password in Sql oracle server



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

PM>
```

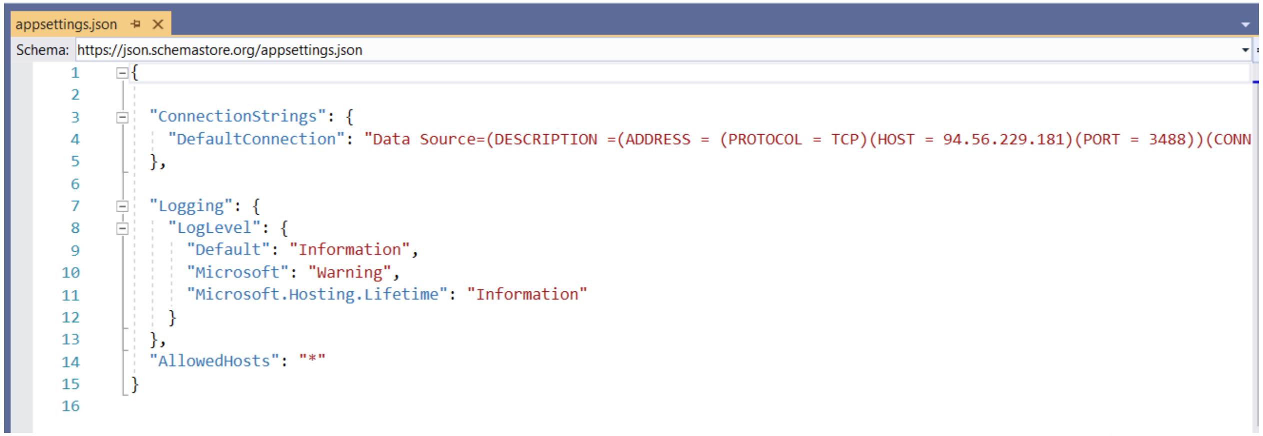
Paste the code here.

Generate ASP.NET Core MVC Model Classes

TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

In appsettings.json:

```
"ConnectionStrings": {
"DefaultConnection": "Data Source=(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = 94.56.229.181)(PORT = 3488))(CONNECT_DATA =(SERVER = DEDICATED)(SERVICE_NAME = traindb)));User Id=****;Password=****;Persist Security Info=True;"
  },
```



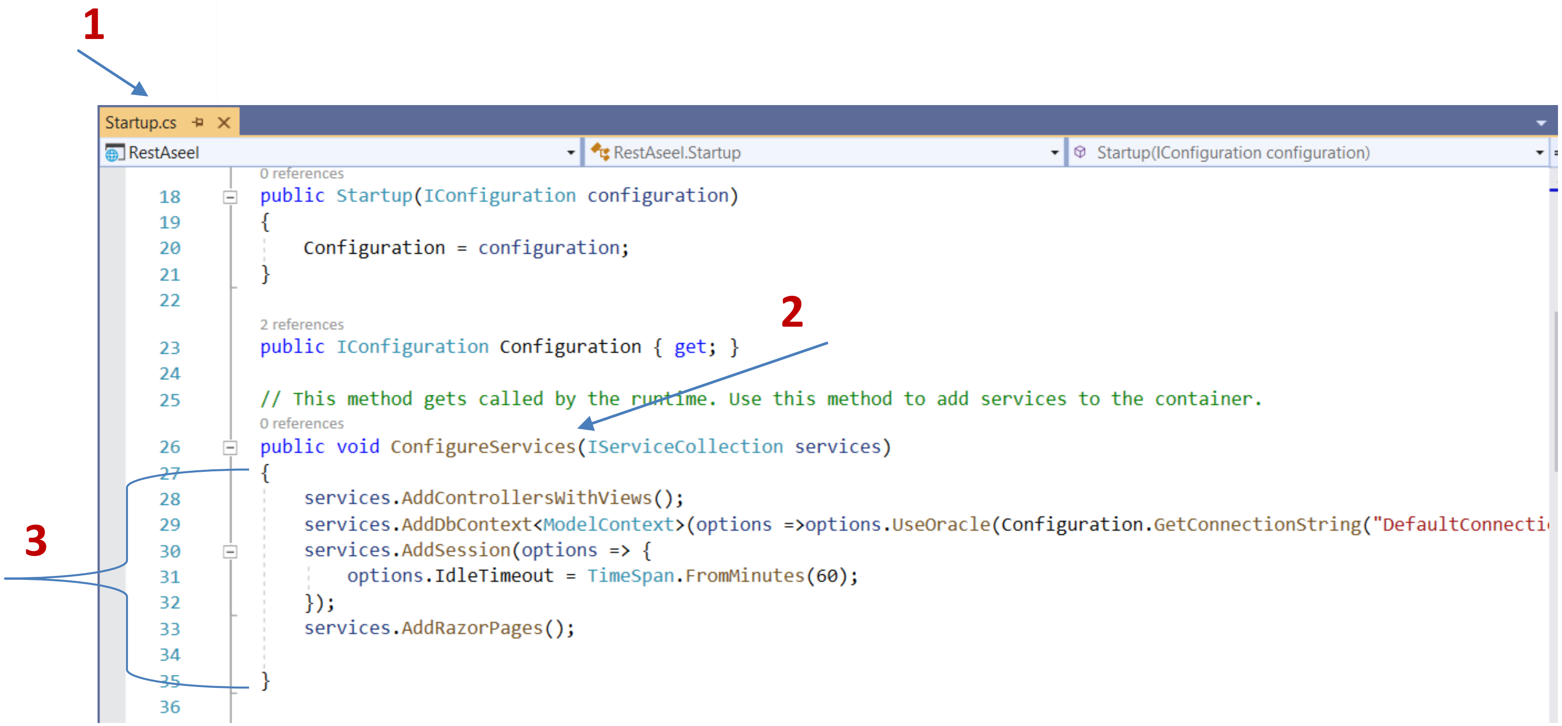
```
{

  "ConnectionStrings": {
    "DefaultConnection": "Data Source=(DESCRIPTION =(ADDRESS = (PROTOCOL = TCP)(HOST = 94.56.229.181)(PORT = 3488))(CONN
  },

  "Logging": {
    "LogLevel": {
      "Default": "Information",
      "Microsoft": "Warning",
      "Microsoft.Hosting.Lifetime": "Information"
    }
  },
  "AllowedHosts": "*"
}
```

Generate ASP.NET Core MVC Model Classes

In Startup.cs:

```
public void ConfigureServices(IServiceCollection services)
{
    services.AddDbContext<ModelContext>(options =>
    options.UseOracle(
    Configuration.GetConnectionString("DefaultConnection")));
    services.AddSession(options => {
    options.IdleTimeout = TimeSpan.FromMinutes(60);});
    services.AddControllersWithViews();
    services.AddRazorPages();
}
```

Generate ASP.NET Core MVC Model Classes



```
0 references
public Startup(IConfiguration configuration)
{
    Configuration = configuration;
}

2 references
public IConfiguration Configuration { get; }

// This method gets called by the runtime. Use this method to add services to the container.
0 references
public void ConfigureServices(IServiceCollection services)
{
    services.AddControllersWithViews();
    services.AddDbContext<ModelContext>(options =>options.UseOracle(Configuration.GetConnectionString("DefaultConnecti
    services.AddSession(options => {
        options.IdleTimeout = TimeSpan.FromMinutes(60);
    });
    services.AddRazorPages();

}
```

Chapter 2

1 Generate ASP.NET Core MVC Model Classes

2 Route

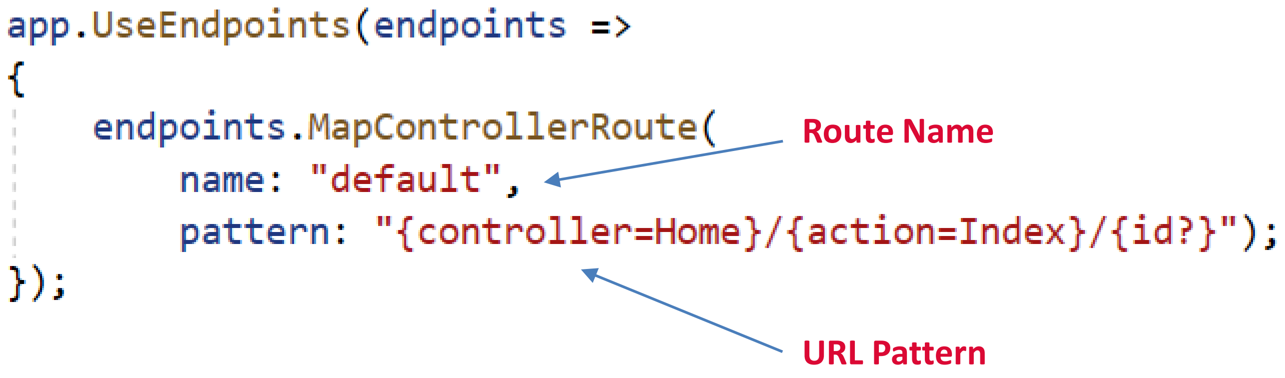
3 CRUD Operation

4 Controller and View

Route

- Route defines the URL pattern and handler information, Routing maps URL to physical file or class (controller class in MVC).

- Configure routes in startup.cs :



```
app.UseEndpoints(endpoints =>
{
    endpoints.MapControllerRoute(
        name: "default",
        pattern: "{controller=Home}/{action=Index}/{id?}");
});
```

Route Name

URL Pattern

Chapter 2

1 Generate ASP.NET Core MVC Model Classes

2 Route

3 CRUD Operation

4 Controller and View

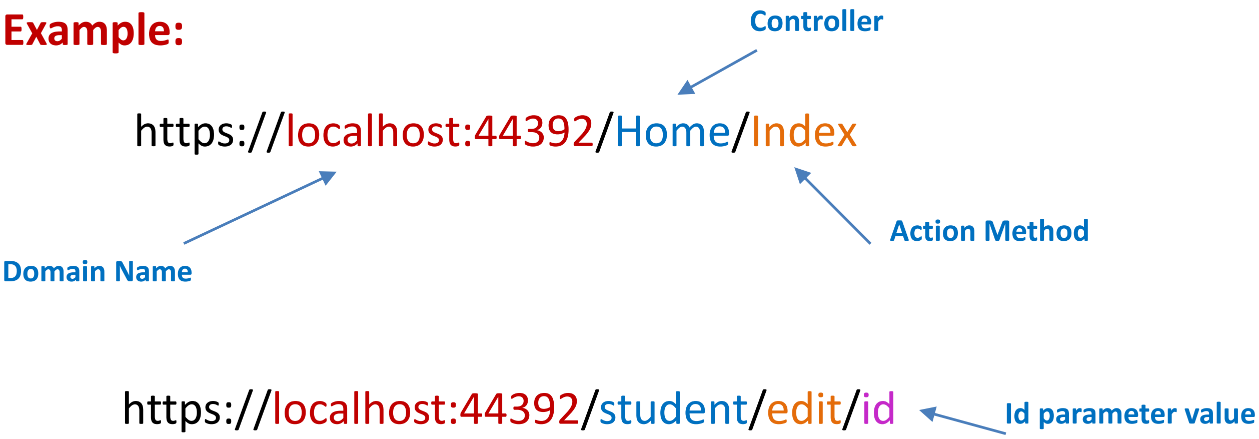
URL Pattern

- The URL pattern is considered only after the domain name part in the URL.

- would look like localhost:xxxx/{controller}/{action}/{id?}.

  ? : mean it is optional

Example:



https://localhost:44392/Home/Index

https://localhost:44392/student/edit/id

CRUD Operation

Overview of CRUD Operation:

CRUD is an acronym that comes from the computer programming world. It refers to the four functions that are represented necessary to implement a persistent storage application.

CRUD: Create, Read, Update and **Delete**.

Chapter 2

1 Generate ASP.NET Core MVC Model Classes

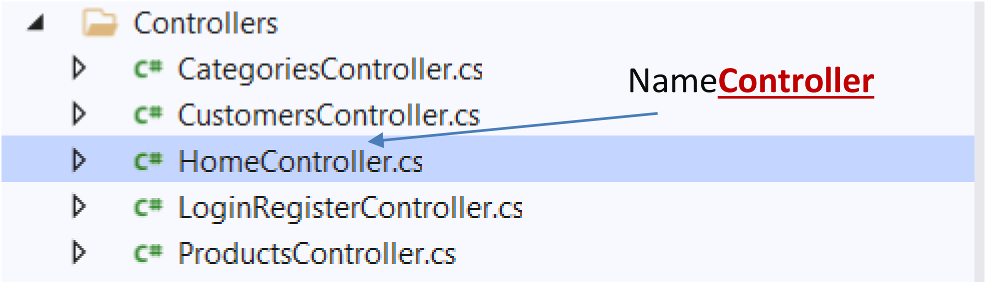
2 Route

3 CRUD Operation

4 Controller and View

Controller

- The Controller handles any incoming URL request.
- Contains public methods called Action methods.
- Handles incoming browser requests, retrieves necessary model data, and returns appropriate responses.
- Every controller class name must end with the word "Controller“.



Controllers
- CategoriesController.cs
- CustomersController.cs
- HomeController.cs
- LoginRegisterController.cs
- ProductsController.cs

Name**Controller**

Controller

- Action method must be public. It cannot be private or protected , overloaded , static method.

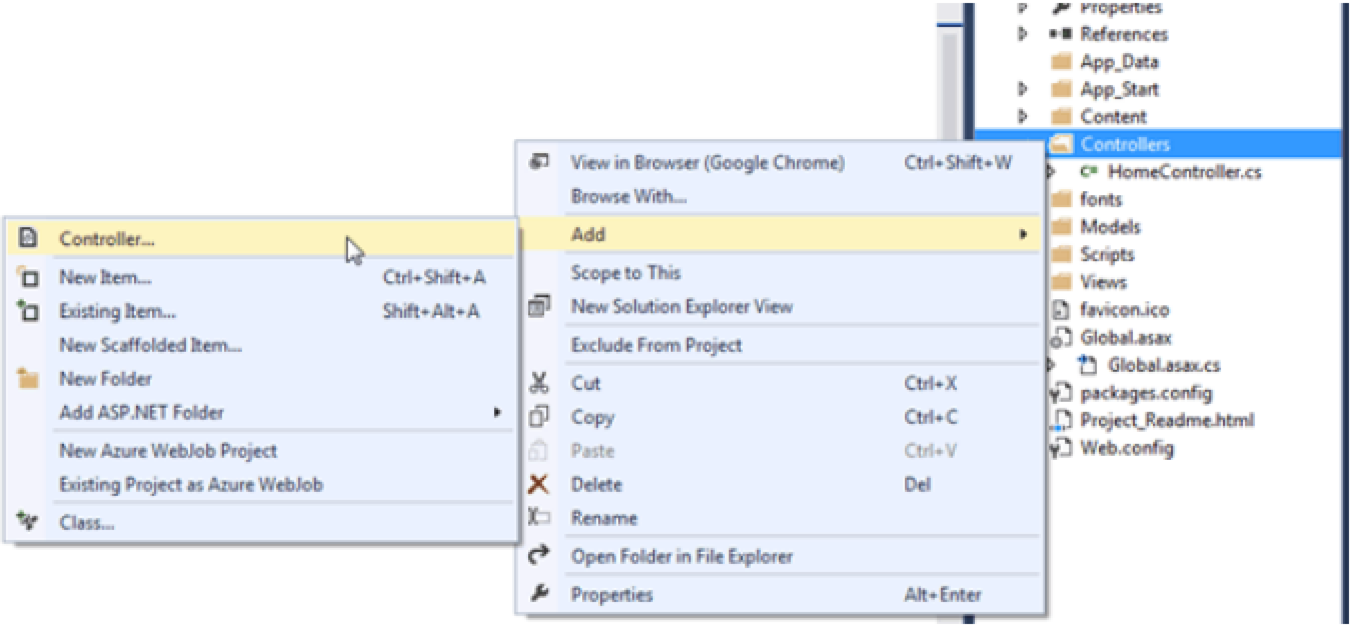
- ActionResult is a base class of all the result type which returns from Action method.

- The base Controller class contains methods that returns appropriate result type.

- ActionVerbs: Handle different type of http request { HttpGet, HttpPost, HttpPut}.

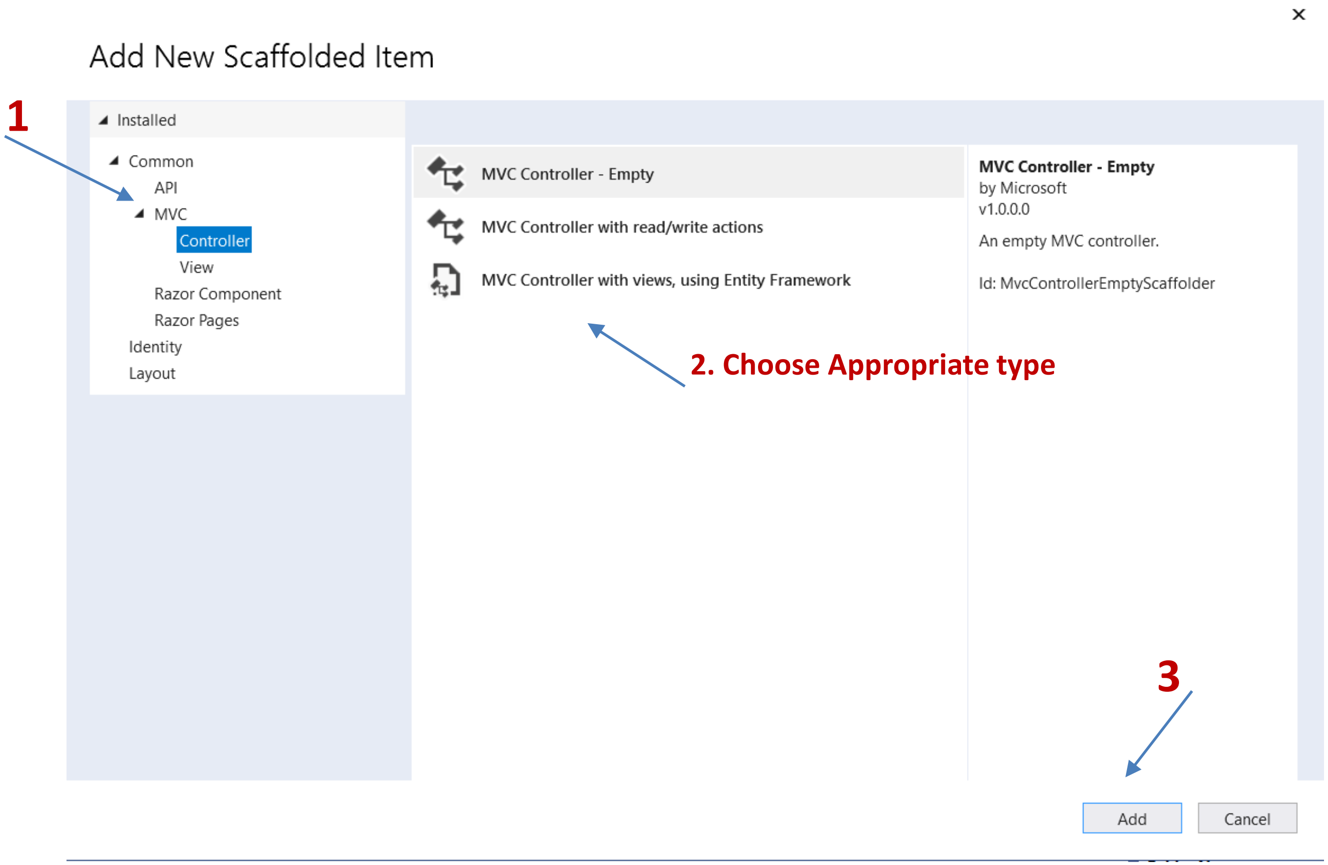
Controller

- Add new Controller
  - ➢ In the Visual Studio, right click on the Controller folder -> select Add -> click on Controller.



شركة تحالف الإمارات للحلول التقنية ذ.م.م.
TAHALUF AL EMARAT TECHNICAL SOLUTIONS L.L.C.

Controller



Add New Scaffolded Item

MVC Controller - Empty
by Microsoft
v1.0.0.0

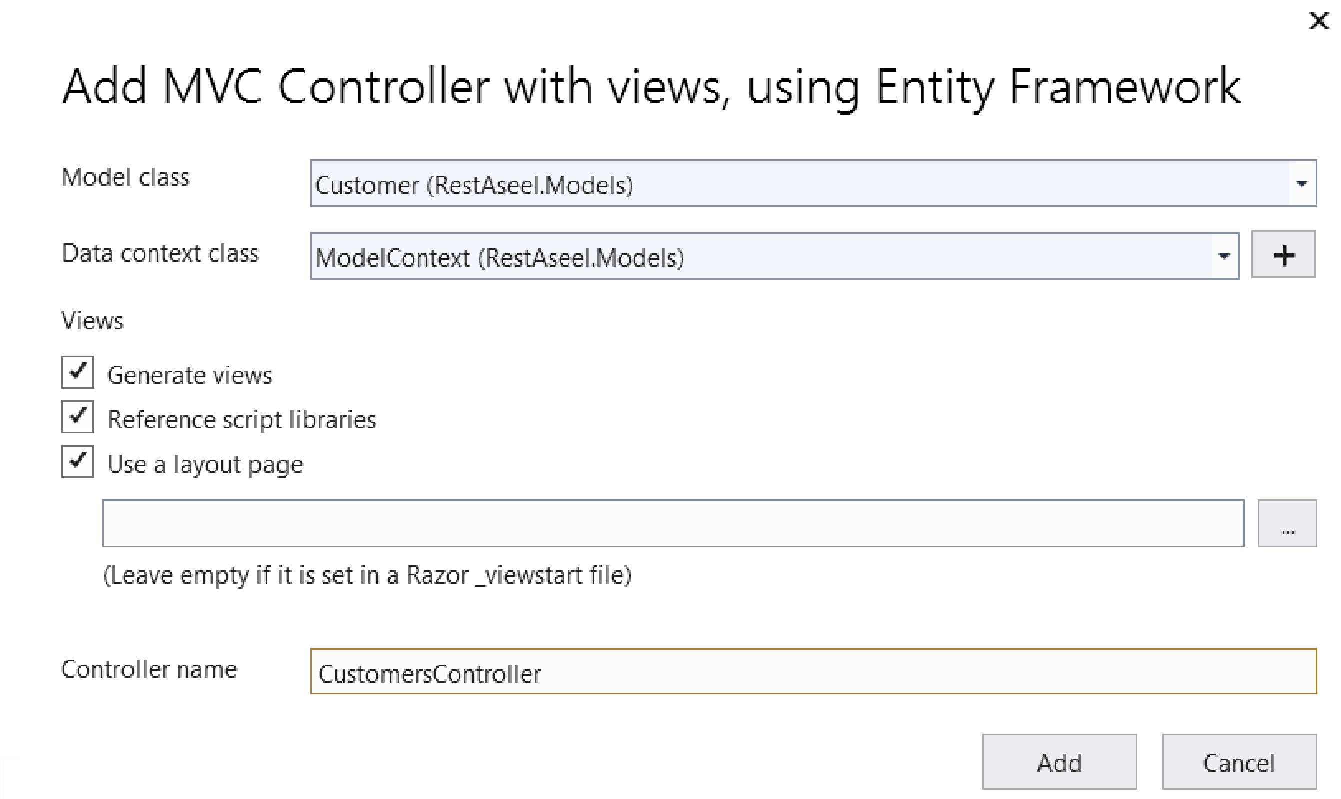
An empty MVC controller.

Id: MvcControllerEmptyScaffolder

2. Choose Appropriate type

Controller

- Choose which **model class** you need to create the controller for, and the **data context** class.



Add MVC Controller with views, using Entity Framework

Model class: Customer (RestAseel.Models)

Data context class: ModelContext (RestAseel.Models)

Views

☑ Generate views

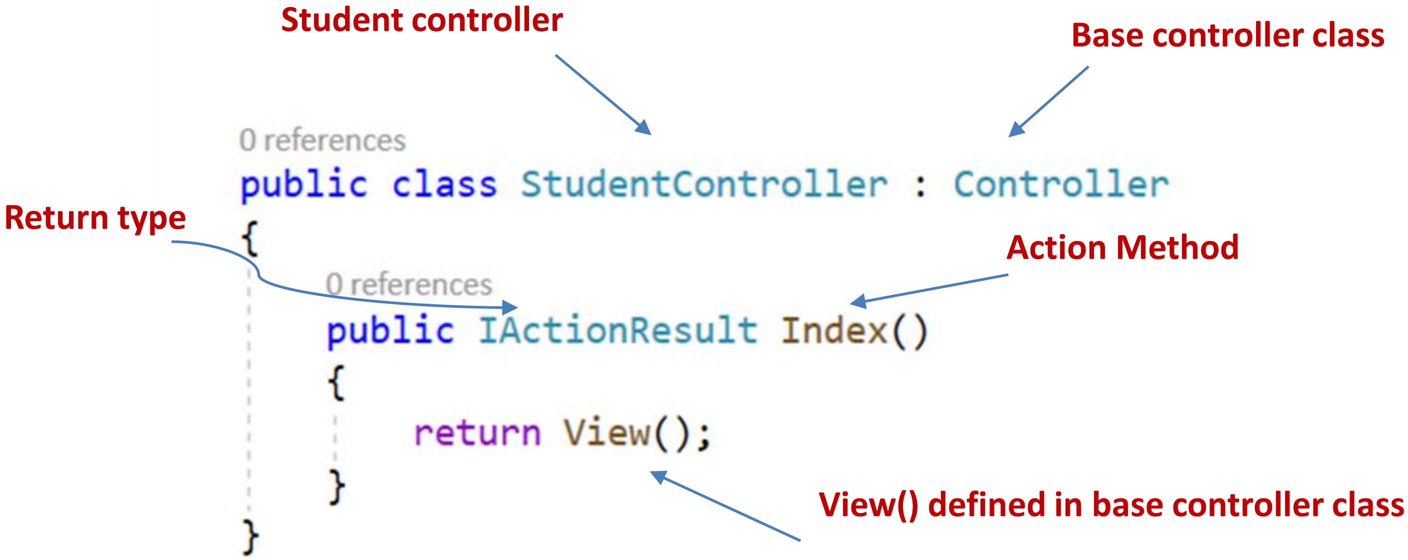
☑ Reference script libraries

☑ Use a layout page

(Leave empty if it is set in a Razor _viewstart file)

Controller name: CustomersController

Controller



```
0 references
public class StudentController : Controller
{
    0 references
    public IActionResult Index()
    {
        return View();
    }
}
```

Student controller

Base controller class

Return type

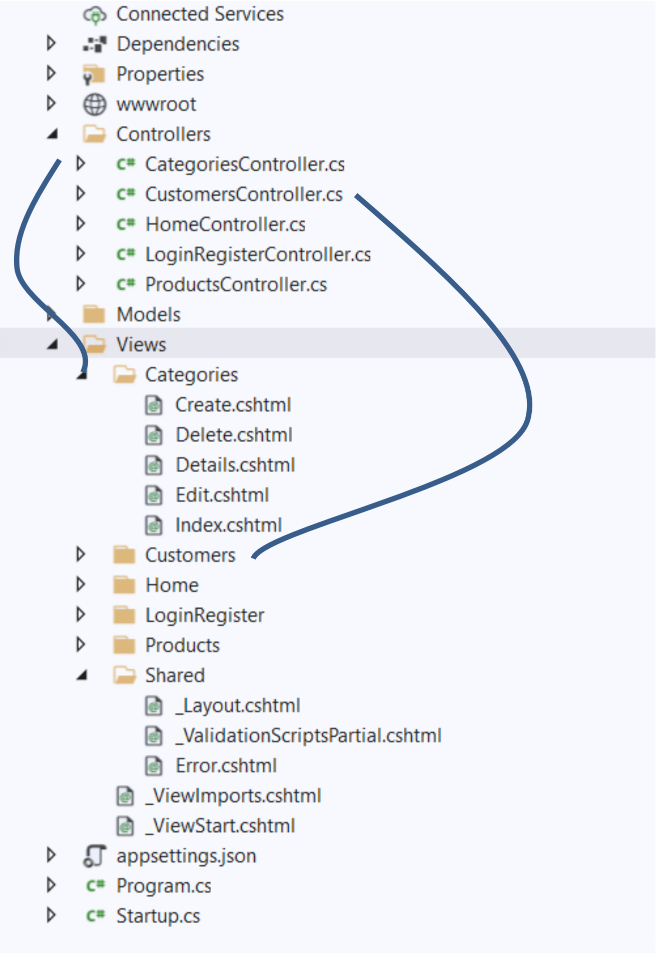
Action Method

View() defined in base controller class

View

- View is used to display data using the model class object.
- The Shared folder contains views, layout views, and partial views, which will be shared among multiple controllers.
- A controller can render one or more views.

View



```
  Connected Services
▷ Dependencies
▷ Properties
▷ wwwroot
◢ Controllers
    ▷ CategoriesController.cs
    ▷ CustomersController.cs
    ▷ HomeController.cs
    ▷ LoginRegisterController.cs
    ▷ ProductsController.cs
▷ Models
◢ Views
    ◢ Categories
        Create.cshtml
        Delete.cshtml
        Details.cshtml
        Edit.cshtml
        Index.cshtml
    ▷ Customers
    ▷ Home
    ▷ LoginRegister
    ▷ Products
    ◢ Shared
        _Layout.cshtml
        _ValidationScriptsPartial.cshtml
        Error.cshtml
      _ViewImports.cshtml
      _ViewStart.cshtml
▷ appsettings.json
▷ Program.cs
▷ Startup.cs
```

- **Each controller has a view with the same name.**

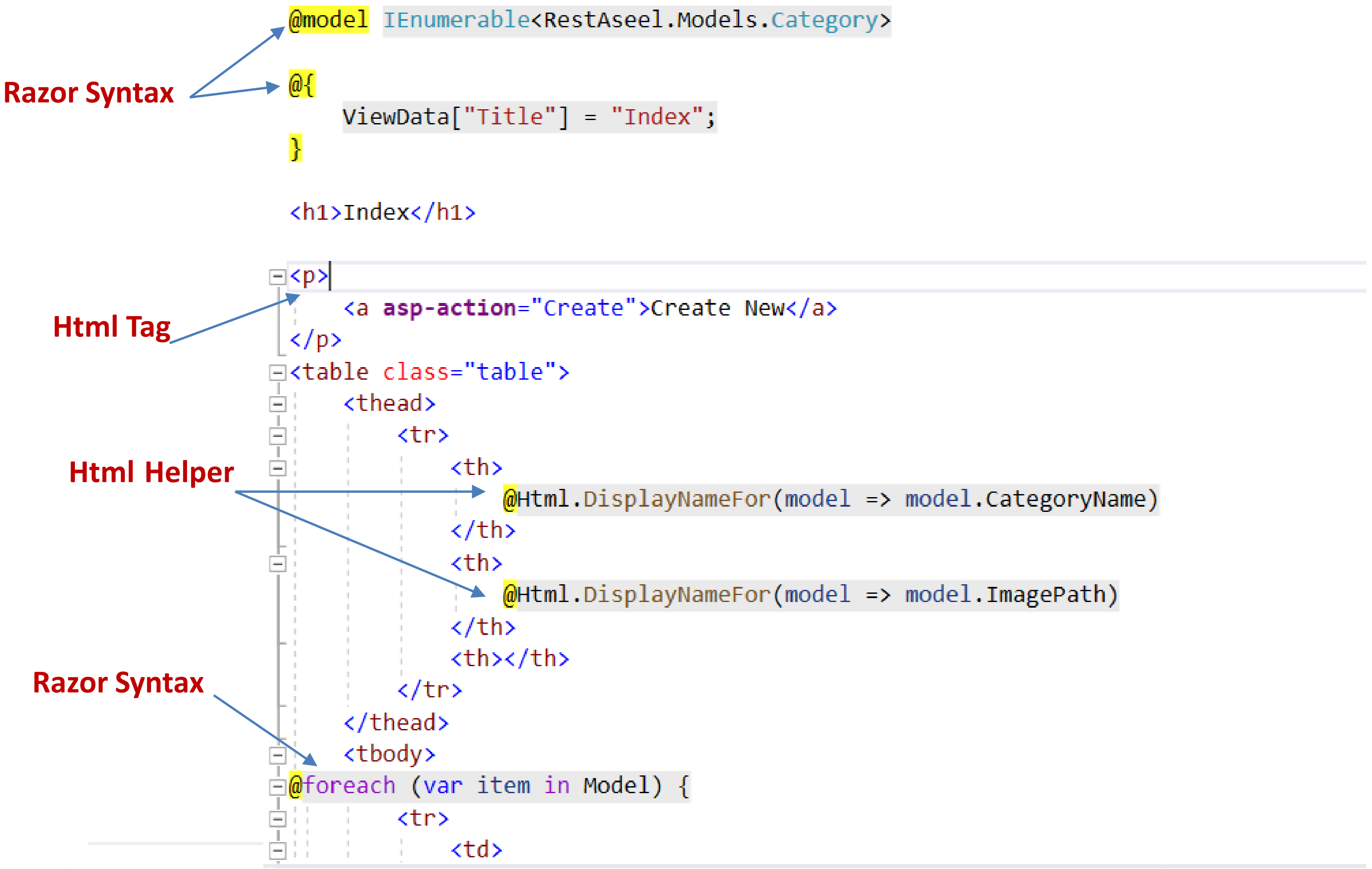
View

- Razor View: allows you to write a mix between Html tags and C#|VB code (viewname.**vbhtml** |viewname.**cshtml**).

- Razor syntax is Compact, Easy to Learn, Intellisens.

- Razor Inline expression using @ with C# code, Multi-statement Code block using @{ ……. } with C# code.

View



Razor Syntax

Html Tag

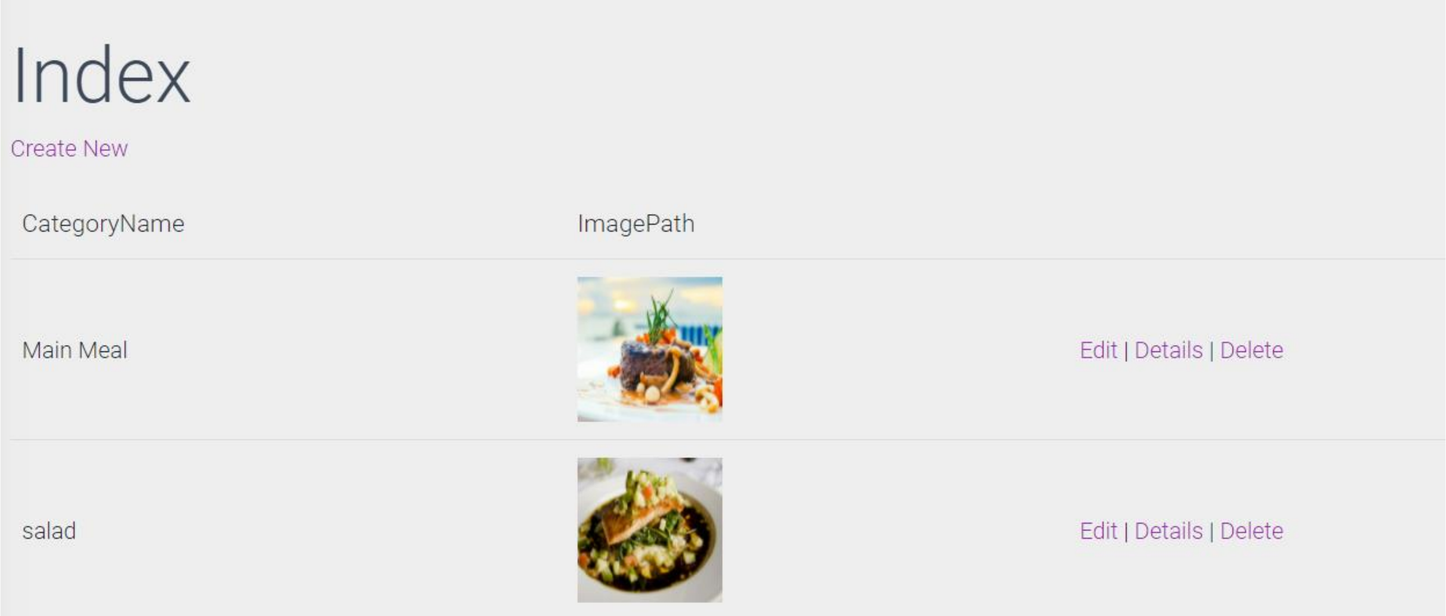
Html Helper

Razor Syntax

```
@model IEnumerable<RestAseel.Models.Category>

@{
    ViewData["Title"] = "Index";
}

<h1>Index</h1>

<p>
    <a asp-action="Create">Create New</a>
</p>
<table class="table">
    <thead>
        <tr>
            <th>
                @Html.DisplayNameFor(model => model.CategoryName)
            </th>
            <th>
                @Html.DisplayNameFor(model => model.ImagePath)
            </th>
            <th></th>
        </tr>
    </thead>
    <tbody>
@foreach (var item in Model) {
        <tr>
            <td>
```

View

Result:



Index

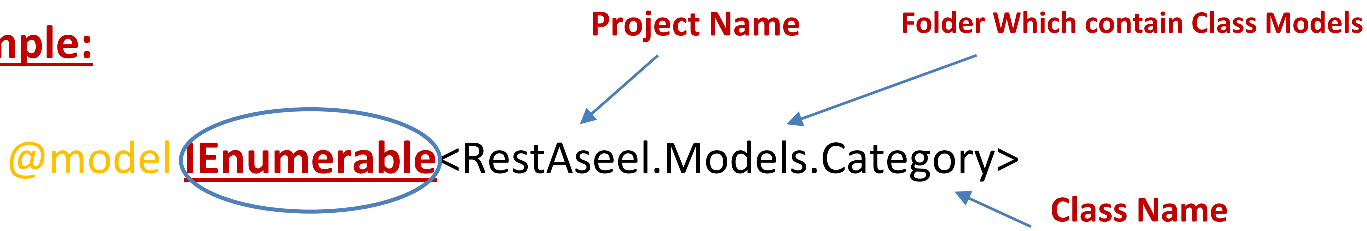
Create New

CategoryName	ImagePath	
Main Meal		Edit | Details | Delete
salad		Edit | Details | Delete

View

- To define a List of objects use **IEnumerable,** usually use it in the index view page to display all data in the table.

Example:



Project Name

Folder Which contain Class Models

@model **IEnumerable**<RestAseel.Models.Category>

Class Name

- In Edit, Details, delete and create view page which returns the data from one object (row) declare :

@model <RestAseel.Models.Category>

Add MVC Controller with views, using Entity Framework

Task (10 Marks):

Create a Model-View-Controller for all tables in the database:

1. Category.
2. Product.
3. Customer.
4. UserLogin.