

Part 1: Front-End Development (ASP.NET MVC and JavaScript):

ASP.NET MVC web application Prerequisites:-

Software's required:

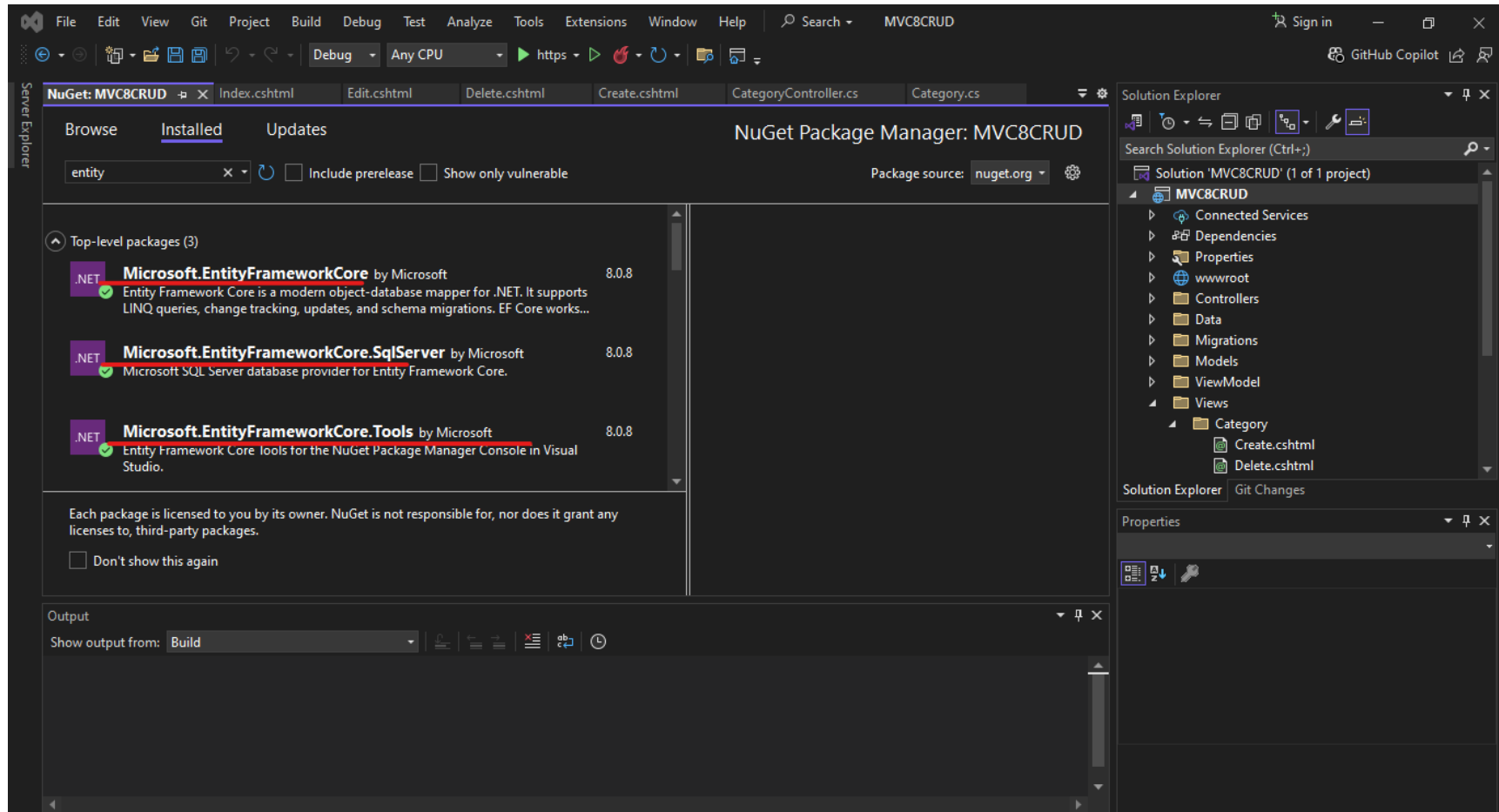
- 1. Visual studio**
- 2. SQL server management studio**

Install EF Core DB Providers

Go to Tools > NuGet Package Manager > Manage NuGet Packages for Solution and search for

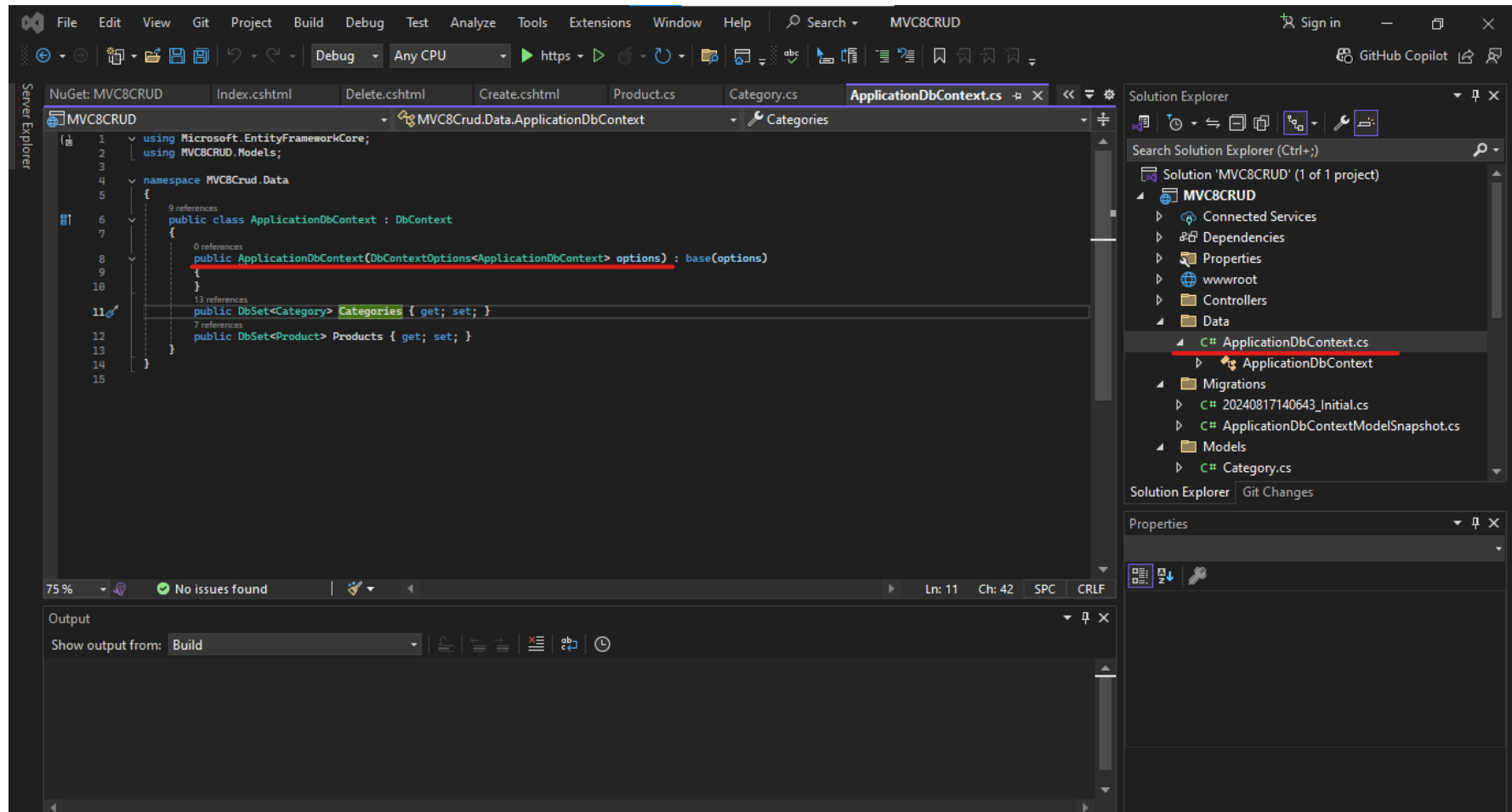
- 1. Microsoft.EntityFrameworkCore.**
- 2. Microsoft.EntityFrameworkCore.SqlServer.**
- 3. Microsoft.EntityFrameworkCore.Tools**

1. Install EF Code tools packages in the solution created in visual studio as highlighted below.



2. Adding EF Core DbContext File

We now have to add Entity Framework Core Database Context file for defining your database and tables. So, created a new class called ApplicationDbContext.cs in data folder



3. Creating a database using EF core migration

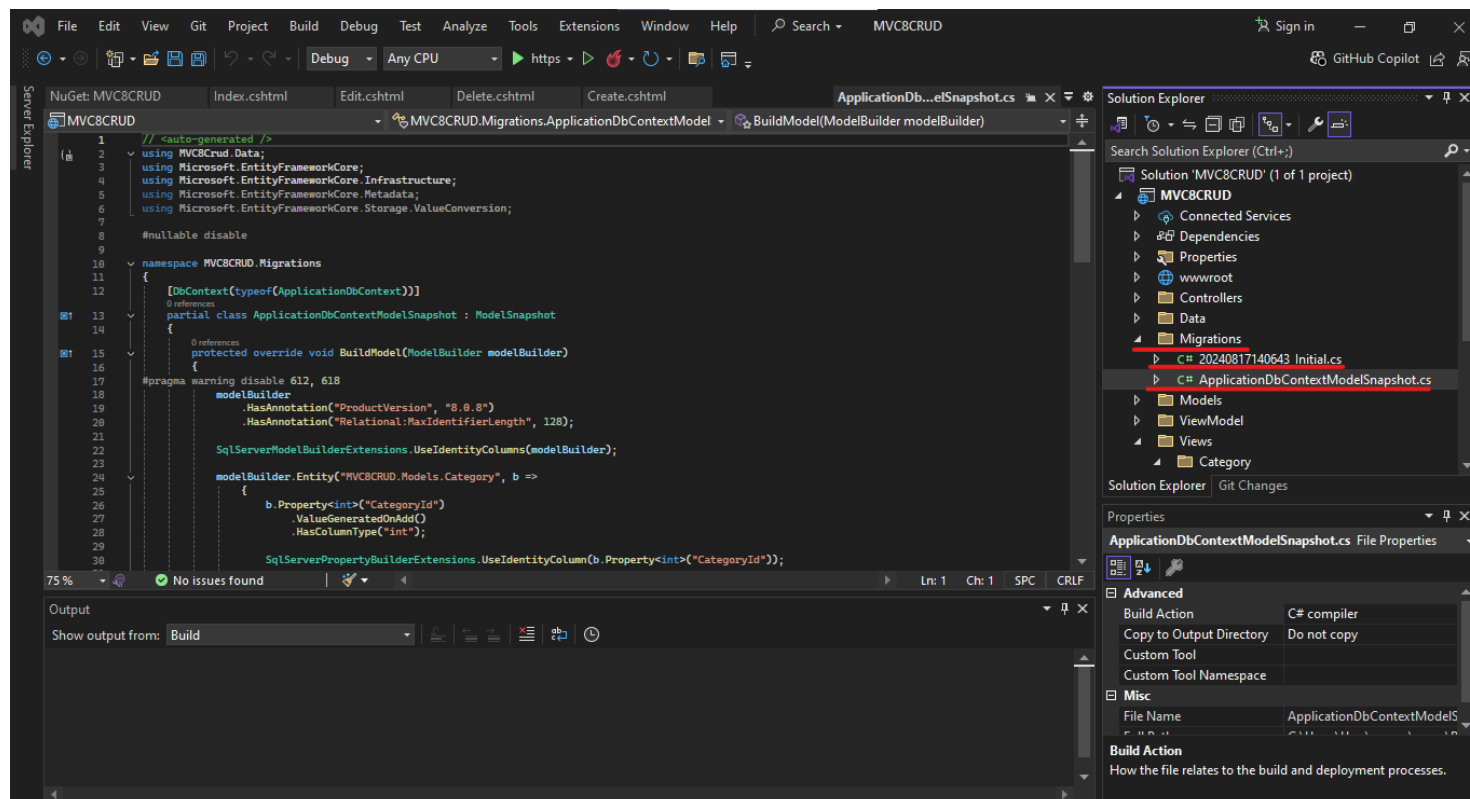
Now run the Migrations in Entity Framework Core so that the '[MVC8CrudDb]' database and '[dbo].[Categories]' table are created. So open the Package Manager Console window in Visual Studio then run this below command,

PM> add-migration Initial

C#

This command will create Migration classes inside Migrations folder in the root of the app. Next, execute the migrations by the below command,

PM> Update-Database



This will create the database and table in your SQL Server LocalDB database.

The screenshot shows the Microsoft SQL Server Enterprise Manager interface. The Object Explorer on the left displays the database structure for 'DESKTOP-6JBC82E (SQL Server 15.0.2000.5 - DESKTOP-6JBC82E\User (60))'. The 'dbo' schema is expanded, and the tables 'dbo.EFMigrationsHistory' and 'dbo.Categories' are highlighted with red boxes. The central query editor shows the following SQL query:

```
SELECT TOP (1000) [MigrationId]
, [ProductVersion]
FROM [MVC8CrudDb].[dbo].[__EFMigrationsHistory]
```

The Results pane at the bottom displays the query output in a table format:

	MigrationId	ProductVersion
1	20240817140643_Initial	8.0.8

The status bar at the bottom indicates 'Query executed successfully.' and '1 rows'.

ASP.NET MVC web application with two views: one for displaying a list of items and another for adding a new item.

Displaying category

← ↻ Not secure | https://localhost:7128/Category

MVC8CRUD Home Privacy

Category (test) added successfully.

Category List

Add Category

Category Id	Category Name	Action
2	rukaiyya	<div>EditDelete</div>
3	test	<div>EditDelete</div>

Adding category

[←](#) [↻](#) Not secure | <https://localhost:7128/Category/Create> [A](#) [☆](#) [📄](#) [📑](#) [🔍](#) [🔒](#) [⋮](#)

MVC8CRUD [Home](#) [Privacy](#)

Add Category

Category Name

test3

Save

Back

Database Interaction entry.

Connect

DESKTOP-6JBC82E (SQL Server 15.0.2000.5 - DESKTOP)

Databases

System Databases

Database Snapshots

MVC8CrudDb

Database Diagrams

Tables

System Tables

FileTables

External Tables

Graph Tables

dbo._EFMigrationsHistory

dbo.Categories

Columns

Keys

Constraints

Triggers

Indexes

Statistics

dbo.Products

Views

External Resources

Synonyms

Programmability

Service Broker

Storage

Security

Security

SELECT TOP (1000) [CategoryId]
,[CategoryName]
FROM [MVC8CrudDb].[dbo].[Categories]

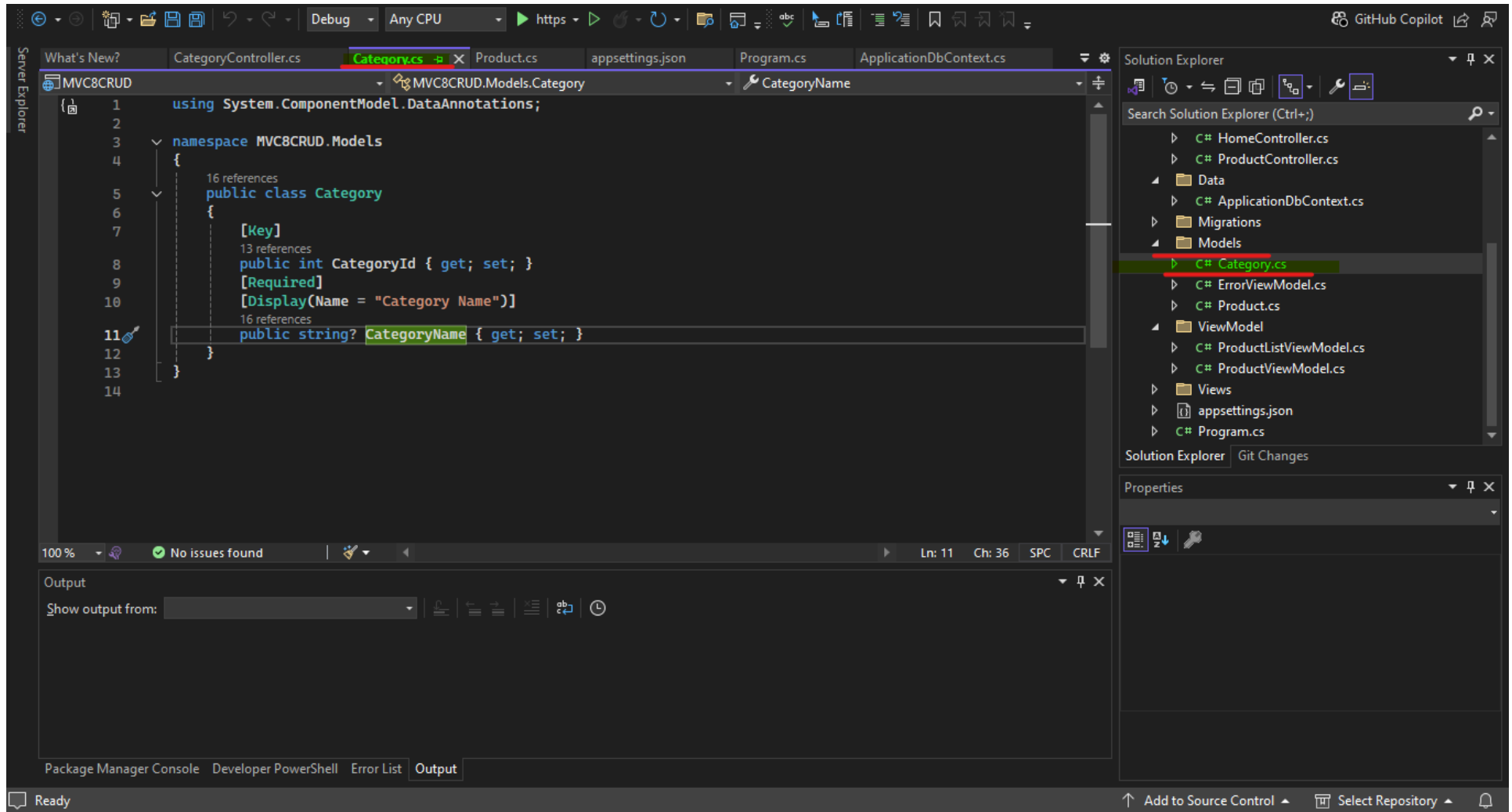
100 %

Results

Messages

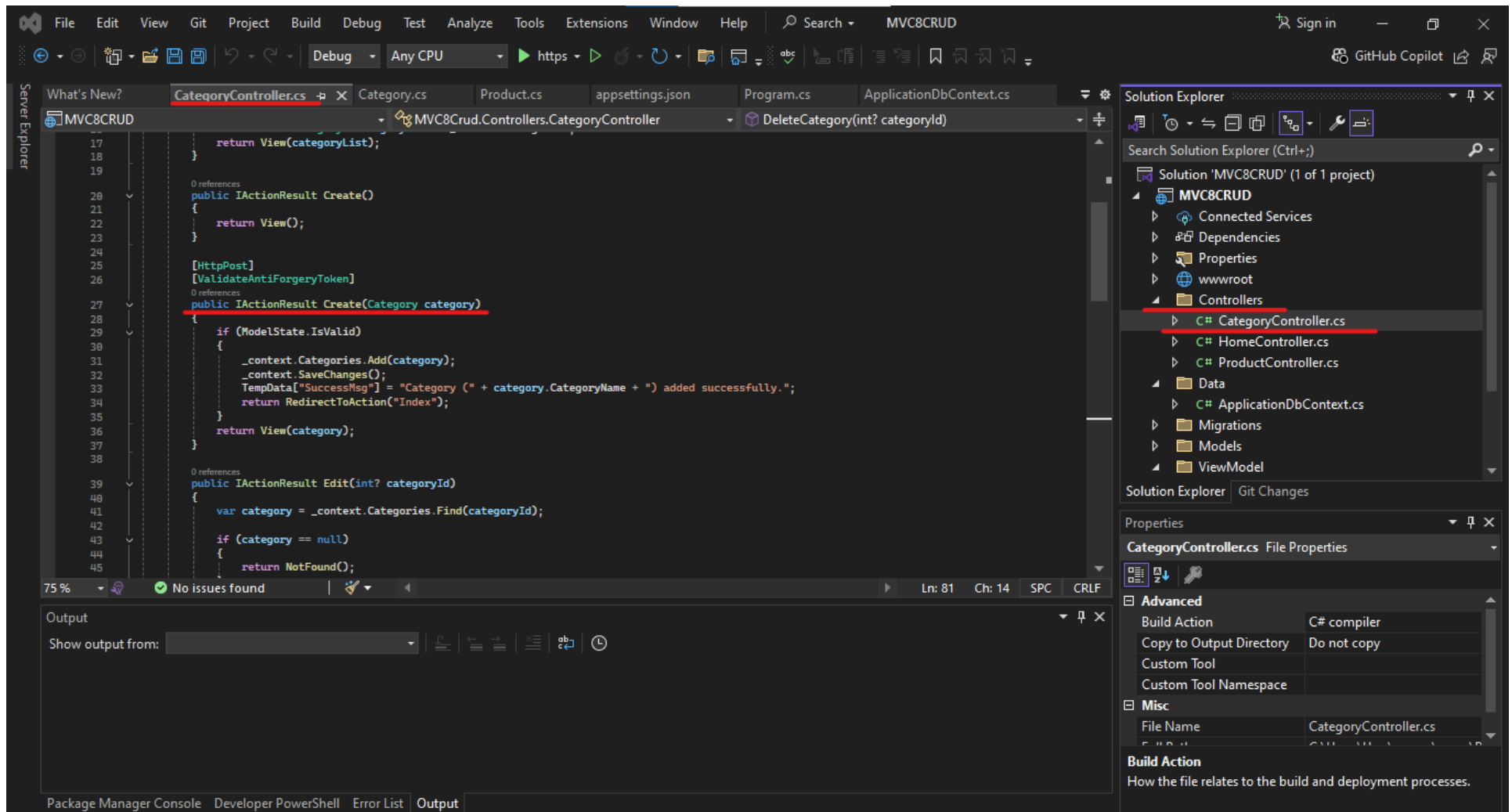
	CategoryId	CategoryName
1	2	rukaiyya
2	3	test

Model to represent the items.

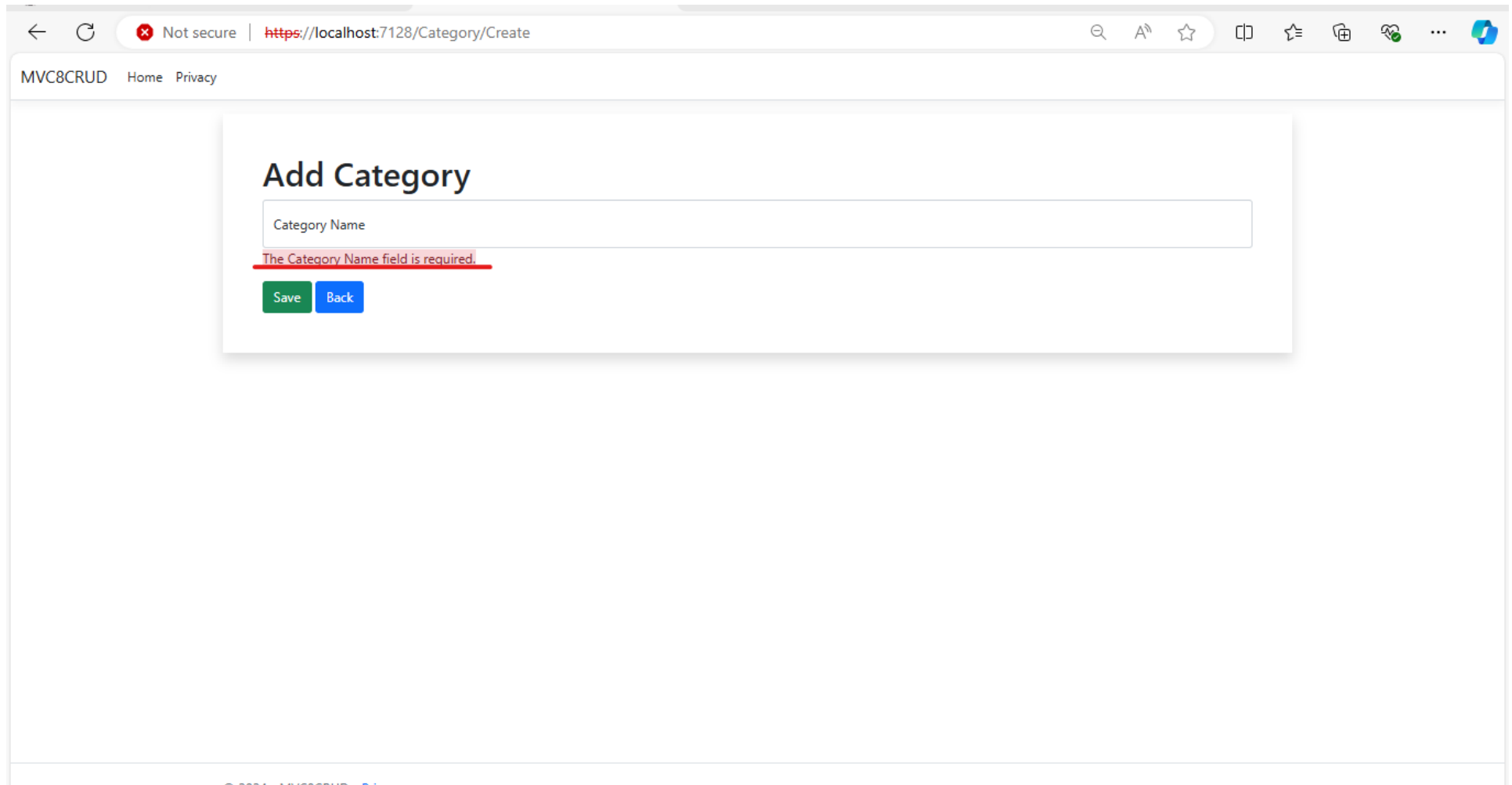


Implementing a controller to handle CRUD operations for the items.

Controller with create code.



Create code view and Client Side validation for input fields.



The screenshot shows a web browser window with the address bar displaying "https://localhost:7128/Category/Create". The page title is "MVC8CRUD" with links for "Home" and "Privacy". The main content area features a form titled "Add Category". Inside the form, there is a text input field labeled "Category Name". Below the input field, a red error message is displayed: "The Category Name field is required." At the bottom of the form, there are two buttons: a green "Save" button and a blue "Back" button.

← ↻ Not secure | https://localhost:7128/Category/Create 🔍 🔊 ☆ 📄 ⌵ 📁 📌 ... 🌐

MVC8CRUD Home Privacy

Add Category

Category Name

The Category Name field is required.

Save Back

© 2024 MVC8CRUD - All rights reserved.

Controller with delete code.

The screenshot displays the Visual Studio IDE with the `CategoryController.cs` file open. The `DeleteCategory` method is implemented as follows:

```
69     {
70         return NotFound();
71     }
72     return View(category);
73 }
74
75 [HttpPost]
76 [ValidateAntiForgeryToken]
77 0 references
78 public IActionResult DeleteCategory(int? categoryId)
79 {
80     var category = _context.Categories.Find(categoryId);
81     if (category == null)
82     {
83         return NotFound();
84     }
85     _context.Categories.Remove(category);
86     _context.SaveChanges();
87     TempData["SuccessMsg"] = "Category (" + category.CategoryName + ") deleted successfully.";
88     return RedirectToAction("Index");
89 }
90
91 }
```

The `Solution Explorer` on the right shows the project structure for `MVC8CRUD`, with `CategoryController.cs` selected under the `Controllers` folder. The `Properties` window at the bottom right shows the `CategoryController.cs` file properties, including the `Build Action` set to `C# compiler` and the `File Name` set to `CategoryController.cs`.

Controller with read code

The screenshot displays the Visual Studio IDE with the following components:

- Editor:** Shows the `CategoryController.cs` file with the following code:

```
9 private readonly ApplicationDbContext _context;
10
11 public CategoryController(ApplicationDbContext context)
12 {
13     _context = context;
14 }
15
16 public IActionResult Index()
17 {
18     IEnumerable<Category> categoryList = _context.Categories;
19     return View(categoryList);
20 }
21
22 public IActionResult Create()
23 {
24     return View();
25 }
26
27 [HttpPost]
28 [ValidateAntiForgeryToken]
29 public IActionResult Create(Category category)
30 {
31     if (ModelState.IsValid)
32     {
33         _context.Categories.Add(category);
34         _context.SaveChanges();
35         TempData["SuccessMsg"] = "Category (" + category.CategoryName + ") added successfully.";
36         return RedirectToAction("Index");
37     }
38     return View(category);
39 }
```
- Solution Explorer:** Shows the project structure for `MVC8CRUD`, with `Controllers` expanded to show `CategoryController.cs`, `HomeController.cs`, and `ProductController.cs`.
- Properties Window:** Shows the `CategoryController.cs` file properties, including the `Build Action` set to `C# compiler` and the `File Name` set to `CategoryController.cs`.
- Output Window:** Shows the `Output` tab, which is currently empty.

View using razor syntax

The screenshot displays the Visual Studio Code interface for a project named MVC8CRUD. The main editor window shows the `Create.cshtml` file, which contains the following Razor code:

```
1 @model Category
2 @{
3     ViewData["Title"] = "Create Category";
4 }
5
6 <div class="container shadow p-5">
7     <div class="row">
8         <h1>Add Category</h1>
9     </div>
10    <form method="post">
11        @*<div asp-validation-summary="All" class="alert-danger"></div> *@
12        <div class="form-floating mb-3">
13            <input type="text" class="form-control" asp-for="CategoryName" placeholder="Enter category name" />
14            <label for="CategoryName">Category Name</label>
15            <span asp-validation-for="CategoryName" class="alert-danger"></span>
16        </div>
17        <button type="submit" class="btn btn-success">Save</button>
18        <a asp-controller="Category" asp-action="Index" class="btn btn-primary">Back</a>
19    </form>
20 </div>
```

The Solution Explorer on the right shows the project structure, with the `Views` folder expanded to show the `Category` subfolder containing `Create.cshtml`, `Delete.cshtml`, `Edit.cshtml`, and `Index.cshtml`. The Output window at the bottom is empty, showing "No issues found".

CRUD OPERATIONS PERFORMED HERE

1. Read Operation:

← ↻ Not secure | https://localhost:7128/Category

MVC8CRUD Home Privacy

Category (test) added successfully.

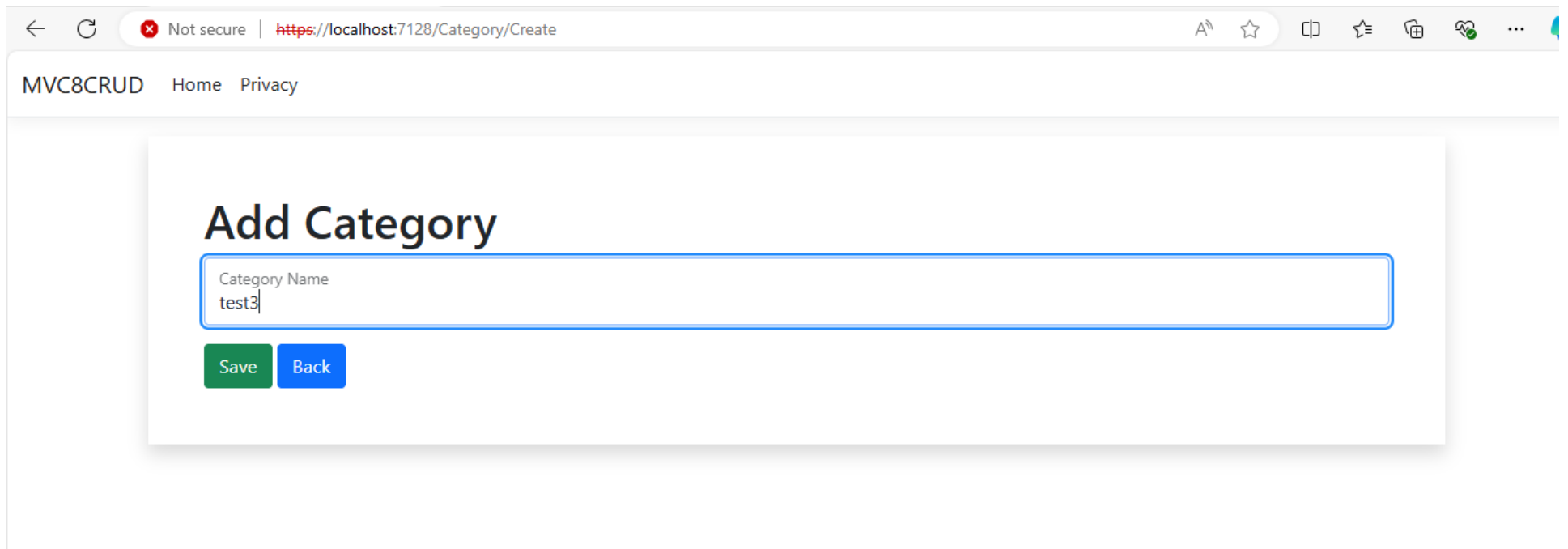
Category List

Add Category

Category Id	Category Name	Action
2	rukaiyya	<div>EditDelete</div>
3	test	<div>EditDelete</div>

2. Add Operation:-

Adding category test3



← ↻ Not secure | <https://localhost:7128/Category/Create> 🔍 ☆ 📄 ⚙️ 🛡️ ...

MVC8CRUD [Home](#) [Privacy](#)

Add Category

Category Name
test3

[Save](#) [Back](#)

Category test3 with ID 4 added successfully

← ↻ Not secure | https://localhost:7128/Category

MVC8CRUD Home Privacy

Category (test3) added successfully.

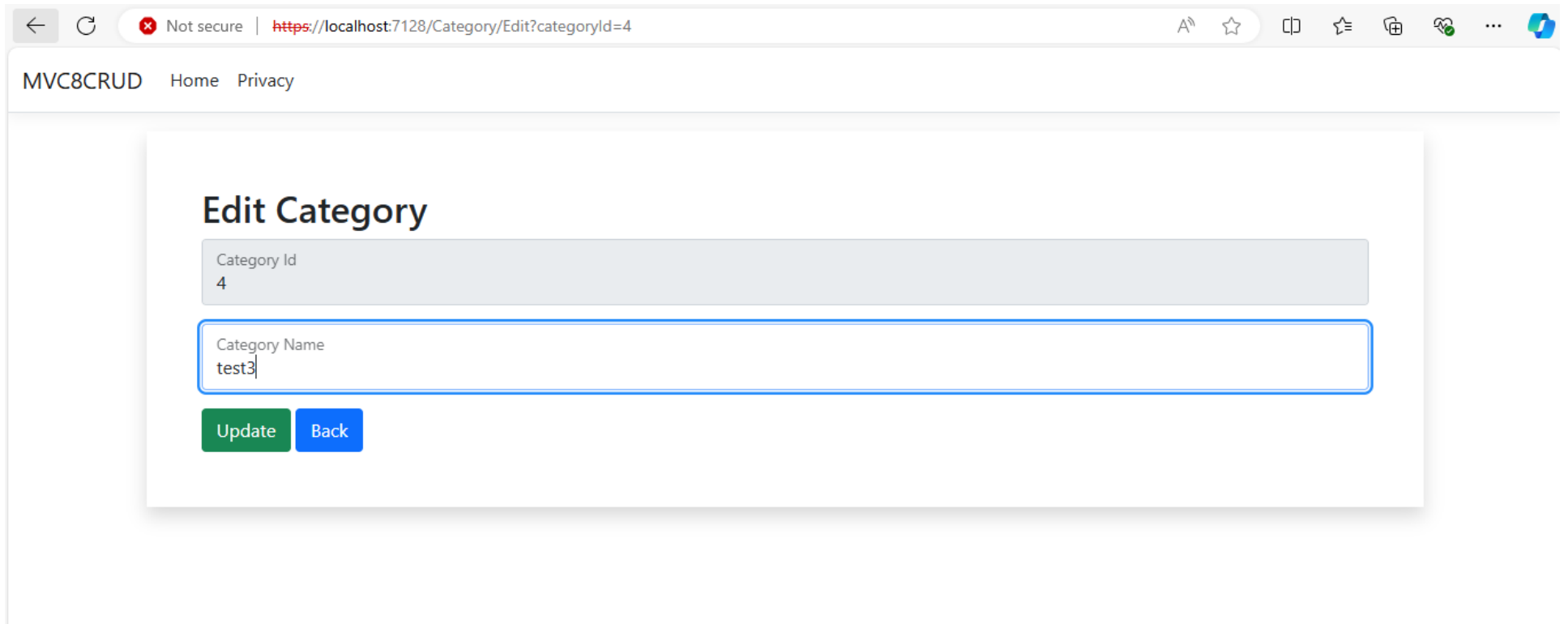
Category List

Add Category

Category Id	Category Name	Action
2	rukaiyya	Edit Delete
3	test	Edit Delete
4	test3	Edit Delete

3. Edit/Update Operation:-

Category field before update (test 3)



The screenshot shows a web browser window with the address bar displaying 'https://localhost:7128/Category/Edit?categoryId=4'. The page title is 'MVC8CRUD' with links for 'Home' and 'Privacy'. The main content area is titled 'Edit Category'. It contains two input fields: 'Category Id' with the value '4' and 'Category Name' with the value 'test3'. Below the input fields are two buttons: 'Update' (green) and 'Back' (blue).

Category Id
4

Category Name
test3

Update Back

Category field after update (test3 to test5)

←

↺

Not secure | https://localhost:7128/Category/Edit?categoryId=4

A¹¹

☆

📄

☰

🔍

🔒

⋮

🌐

MVC8CRUD Home Privacy

Edit Category

Category Id

4

Category Name

test5

Update

Back

Category with ID 4 updated successfully to (test5)

Category (test5) updated successfully.

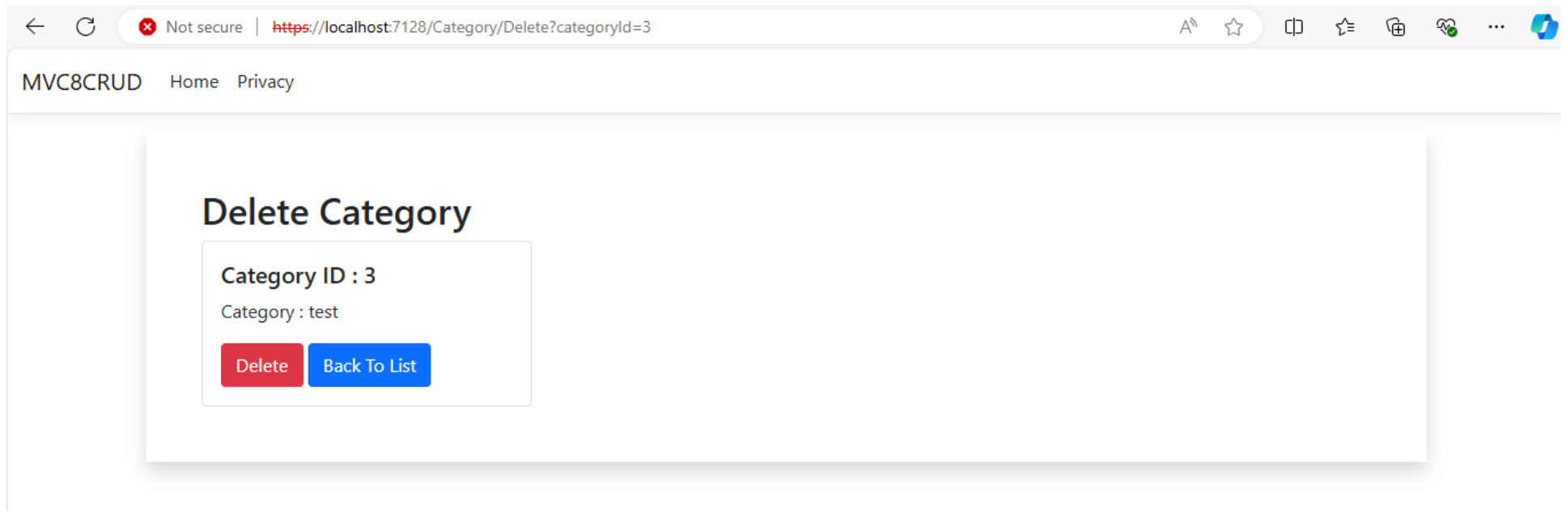
Category List

Add Category

Category Id	Category Name	Action
2	rukaiyya	Edit Delete
3	test	Edit Delete
4	test5	Edit Delete

4. Delete Operation:-

Deleting Category with ID 3



Category with ID 3 (test) deleted successfully

← ↻

Not secure | https://localhost:7128/Category

A ☆ 📄 ☆ 🗂 🛡 ...

MVC8CRUD Home Privacy

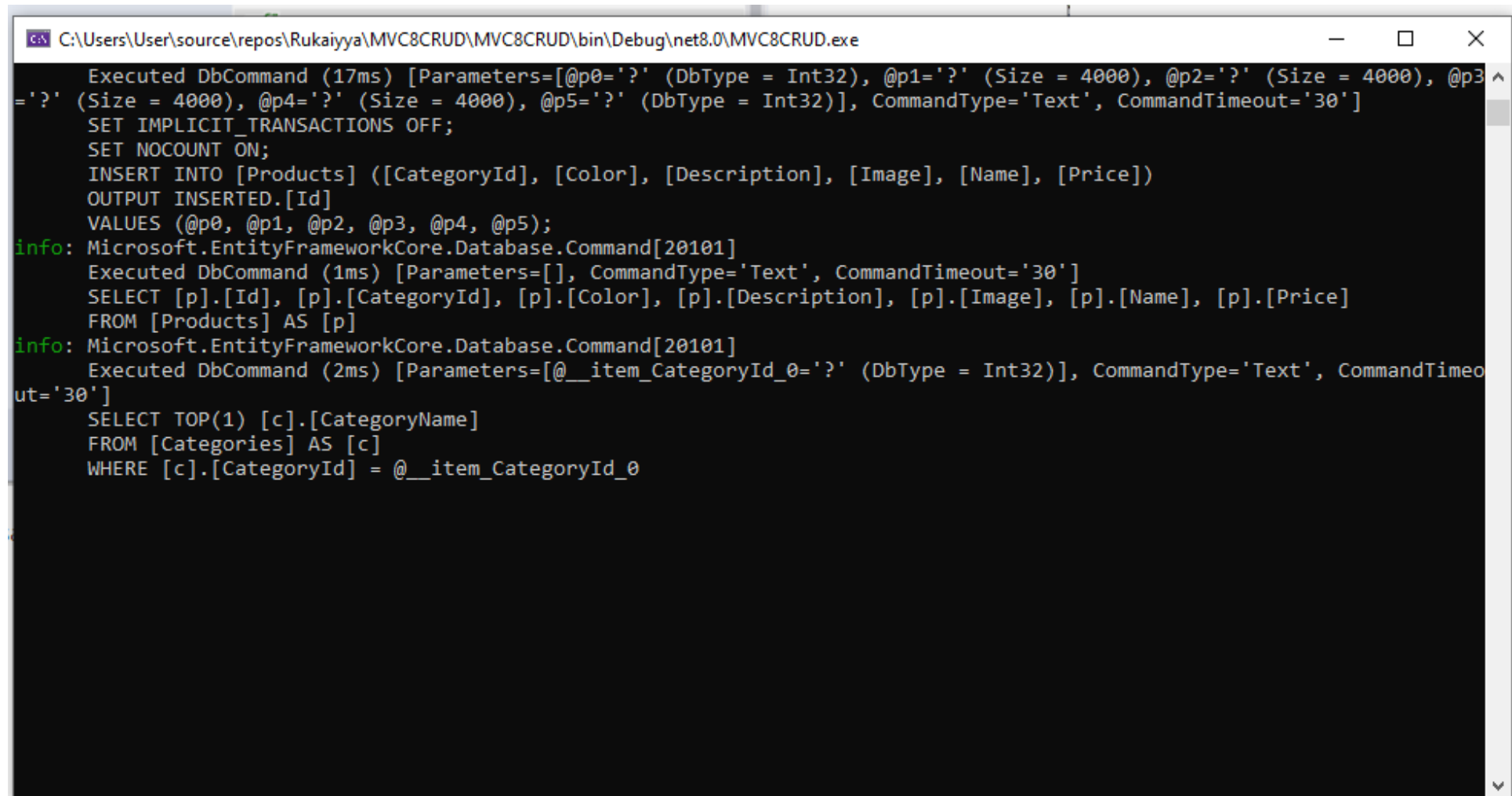
Category (test) deleted successfully.

Category List

Add Category

Category Id	Category Name	Action
2	rukaiyya	<div>EditDelete</div>
4	test5	<div>EditDelete</div>

Backend connectivity



```
C:\Users\User\source\repos\Rukaiyya\MVC8CRUD\MVC8CRUD\bin\Debug\net8.0\MVC8CRUD.exe

Executed DbCommand (17ms) [Parameters=[@p0='?' (DbType = Int32), @p1='?' (Size = 4000), @p2='?' (Size = 4000), @p3='?' (Size = 4000), @p4='?' (Size = 4000), @p5='?' (DbType = Int32)], CommandType='Text', CommandTimeout='30']
SET IMPLICIT_TRANSACTIONS OFF;
SET NOCOUNT ON;
INSERT INTO [Products] ([CategoryId], [Color], [Description], [Image], [Name], [Price])
OUTPUT INSERTED.[Id]
VALUES (@p0, @p1, @p2, @p3, @p4, @p5);
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (1ms) [Parameters=[], CommandType='Text', CommandTimeout='30']
      SELECT [p].[Id], [p].[CategoryId], [p].[Color], [p].[Description], [p].[Image], [p].[Name], [p].[Price]
      FROM [Products] AS [p]
info: Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (2ms) [Parameters=[@__item_CategoryId_0='?' (DbType = Int32)], CommandType='Text', CommandTimeout='30']
      SELECT TOP(1) [c].[CategoryName]
      FROM [Categories] AS [c]
      WHERE [c].[CategoryId] = @__item_CategoryId_0
```

Another ASP.net MVC web application is created for products, Reference attached in the ZIP archive
File name is Product.

Part 2: Back-End Development (APIs and Database Interaction):

Web API controller

The screenshot displays the Visual Studio IDE with the following components:

- Menu Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Toolbar:** Includes icons for undo, redo, save, and other standard editing functions. The 'Debug' button is highlighted.
- Server Explorer:** Located on the left, showing the project structure.
- Code Editor:** Displays the `CategoryController.cs` file. The code includes:
 - Using statements: `using Microsoft.EntityFrameworkCore;`, `using System.Collections.Generic;`, `using System.Threading.Tasks;`
 - Namespace: `namespace WebAPIMVC8CRUD.Controllers`
 - Route attribute: `[Route("api/[controller]")]`
 - ApiController attribute: `[ApiController]`
 - Class: `public class CategoryController : ControllerBase`
 - Property: `private readonly Mvc8crudDbContext _context;`
 - Constructor: `public CategoryController(Mvc8crudDbContext context)`
 - HttpGet method: `[HttpGet] public async Task<IEnumerable<Category>> Get()`
 - HttpGet method: `[HttpGet("{CategoryId}")] public async Task<IActionResult> Get(int CategoryId)`
- Solution Explorer:** Located on the right, showing the project structure. The `CategoryController.cs` file is selected.
- Properties Window:** Located on the right, showing the properties of the selected file. The 'Advanced' tab is active, showing build action and copy settings.
- Output Window:** Located at the bottom, showing the output of the build process.

WEB API Controller-Asynchronous-GET-PUT-POST method code

The screenshot displays the Visual Studio IDE with the **CategoryController.cs** file open. The code implements three asynchronous methods: **Get()**, **Get(int CategoryId)**, **Post(Category category)**, and **Put(Category categoryData)**. The **Get(int CategoryId)** method includes a validation check for **CategoryId < 1** and a query for the first category matching the ID. The **Post** and **Put** methods handle adding and updating categories in the database context.

```
23 public async Task<IEnumerable<Category>> Get()
24 {
25     return await _context.Categories.ToListAsync();
26 }
27
28 [HttpGet("{CategoryId}")]
29 public async Task<ActionResult> Get(int CategoryId)
30 {
31     if (CategoryId < 1)
32         return BadRequest();
33     var Category = await _context.Categories.FirstOrDefaultAsync(m => m.CategoryId == CategoryId);
34     if (Category == null)
35         return NotFound();
36     return Ok(Category);
37 }
38
39 [HttpPost]
40 public async Task<ActionResult> Post(Category category)
41 {
42     _context.Add(category);
43     await _context.SaveChangesAsync();
44     return Ok();
45 }
46
47 [HttpPut]
48 public async Task<ActionResult> Put(Category categoryData)
49 {
50     if (categoryData == null || categoryData.CategoryId == 0)
51         return BadRequest();
52     _context.Update(categoryData);
53     await _context.SaveChangesAsync();
54     return Ok(categoryData);
55 }
```

The **Solution Explorer** on the right shows the project structure for **WebAPIMVC8CRUD**, including **Controllers**, **Models**, and **Properties**. The **Properties** window shows the **CategoryController.cs** file properties, and the **Advanced** tab is selected, showing build action settings.

WEB API Controller-Asynchronous-DELETE method code

The screenshot displays the Visual Studio IDE with the following components:

- Menu Bar:** File, Edit, View, Git, Project, Build, Debug, Test, Analyze, Tools, Extensions, Window, Help.
- Toolbar:** Includes icons for file operations, a search bar, and a status bar showing 'WebAPIMVC8CRUD'.
- Server Explorer:** Located on the left, showing the project structure.
- Code Editor:** Displays the `CategoryController.cs` file. The `Delete(int CategoryId)` method is highlighted in red. The code includes logic for handling null category data, finding the category, and removing it from the database.
- Solution Explorer:** Located on the right, showing the project structure. The 'Controllers' folder is expanded, showing `CategoryController.cs` and `WeatherForecastController.cs`.
- Properties Window:** Located on the right, showing the 'File Properties' for `CategoryController.cs`. The 'Advanced' tab is selected, showing build actions and custom tool settings.
- Output Window:** Located at the bottom, showing the 'Output' tab with a message 'No issues found'.

```
50
51
52     if (categoryData == null || categoryData.CategoryId == 0)
53         return BadRequest();
54
55     var category = await _context.Categories.FindAsync(categoryData.CategoryId);
56     if (category == null)
57         return NotFound();
58     category.CategoryName = categoryData.CategoryName;
59     //category.Description = productData.Description;
60     //product.Price = productData.Price;
61     await _context.SaveChangesAsync();
62     return Ok();
63 }
64
65 [HttpDelete("{CategoryId}")]
66 public async Task<IActionResult> Delete(int CategoryId)
67 {
68     if (CategoryId < 1)
69         return BadRequest();
70     var Category = await _context.Categories.FindAsync(CategoryId);
71     if (Category == null)
72         return NotFound();
73     _context.Categories.Remove(Category);
74     await _context.SaveChangesAsync();
75     return Ok();
76 }
77
78
79
80
```

75 % | No issues found | Ln: 76 | Ch: 10 | SPC | CRLF

Output

Show output from: [dropdown]

Package Manager Console | Developer PowerShell | Error List | Output

Solution Explorer

Search Solution Explorer (Ctrl+):

Solution 'WebAPIMVC8CRUD' (1 of 1 project)

- WebAPIMVC8CRUD
 - Connected Services
 - Dependencies
 - Properties
 - Controllers
 - CategoryController.cs
 - WeatherForecastController.cs
 - Models
 - Category.cs
 - Mvc8crudDbContext.cs
 - Product.cs
 - appsettings.json
 - Program.cs
 - WeatherForecast.cs

Properties

CategoryController.cs File Properties

Advanced

Build Action	C# compiler
Copy to Output Directory	Do not copy
Custom Tool	
Custom Tool Namespace	

Misc

File Name	CategoryController.cs
-----------	-----------------------

Build Action

How the file relates to the build and deployment processes.

CRUD operation in web API

GET method output

The screenshot displays the Swagger UI interface for a web API. The browser address bar shows the URL `localhost:5126/swagger/index.html`. The main interface is titled `GET /api/Category/{CategoryId}`. Under the **Parameters** tab, there is a single path parameter `CategoryId` of type `Integer($int32)` with a required flag and a value of `4` entered in the input field. Below the parameters, there are `Execute` and `Clear` buttons. The **Responses** section shows a `200` status code. The **Response body** is displayed as a JSON object: `{ "categoryId": 4, "categoryName": "EARPODS" }`. The **Response headers** section shows the following headers: `content-type: application/json; charset=utf-8`, `date: Sat, 17 Aug 2024 21:56:31 GMT`, `server: Kestrel`, and `transfer-encoding: chunked`.

GET /api/Category/{CategoryId}

Parameters

Name Description

CategoryId * required
Integer(\$int32)
(path)

4

Execute Clear

Responses

Curl

```
curl -X 'GET' \
  'http://localhost:5126/api/Category/4' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5126/api/Category/4
```

Server response

Code Details

200

Response body

```
{
  "categoryId": 4,
  "categoryName": "EARPODS"
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Sat, 17 Aug 2024 21:56:31 GMT
server: Kestrel
transfer-encoding: chunked
```

GET Method response

Responses

Curl

```
curl -X 'GET' \
'http://localhost:5126/api/Category' \
-H 'accept: application/json'
```

Request URL

```
http://localhost:5126/api/Category
```

Server response

Code	Details
200	<div><div>Response body</div><div><pre>[{ "categoryId": 2, "categoryName": "rukaiyya" }, { "categoryId": 4, "categoryName": "headset" }]</pre></div><div><div>Download</div></div></div>
	<div><div>Response headers</div><div><pre>content-type: application/json; charset=utf-8 date: Sat,17 Aug 2024 21:40:47 GMT server: Kestrel transfer-encoding: chunked</pre></div></div>

Responses

Code	Description	Links
------	-------------	-------

GET method data in Database.

The screenshot displays the Microsoft SQL Server Management Studio interface. The title bar indicates the active window is 'SQLQuery3.sql - DESKTOP-6JBC82E.MVC8CrudDb (DESKTOP-6JBC82E\User (53)) - Microsoft SQL Server Management Studio'. The menu bar includes File, Edit, View, Project, Tools, Window, and Help. The toolbar contains various icons for file operations, query execution, and formatting. The Object Explorer on the left shows the database structure for 'DESKTOP-6JBC82E (SQL Server 15.0.2000.5 - DESKTOP-6JBC82E)'. The database 'MVC8CrudDb' is expanded, showing 'Tables' with 'dbo.Categories' and 'dbo.Products' selected. The main query editor displays the following SQL query:

```
SELECT TOP (1000) [CategoryId]
, [CategoryName]
FROM [MVC8CrudDb].[dbo].[Categories]
```

The query results are shown in the 'Results' tab at the bottom. The results are displayed in a table with two columns: 'CategoryId' and 'CategoryName'. The first two rows are highlighted in yellow:

	CategoryId	CategoryName
1	2	rukaiyya
2	4	headset

The status bar at the bottom indicates 'Query executed successfully.' and provides details about the server, user, database, and execution time: 'DESKTOP-6JBC82E (15.0 RTM) | DESKTOP-6JBC82E\User (53) | MVC8CrudDb | 00:00:00 | 2 rows'.

PUT method output

PUT

/api/Category

^

Parameters

Cancel

Reset

No parameters

Request body

application/json

⌵

```
{
  "categoryId": 4,
  "categoryName": "EARPODS"
}
```

PUT method changes in Database

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the Object Explorer shows the database structure for 'DESKTOP-6JBC82E (SQL Server 15.0.2000.5 - DESKTOP-6JBC82E)'. The 'Categories' table is highlighted under the 'dbo' schema. The main window shows a SQL query executed in 'SQLQuery3.sql'.

```
SELECT TOP (1000) [CategoryId]  
                , [CategoryName]  
FROM [MVC8CrudDb].[dbo].[Categories]
```

The query results are displayed in a table with two columns: 'CategoryId' and 'CategoryName'. The results are as follows:

	CategoryId	CategoryName
1	2	rukaiyya
2	4	EARPODS

The status bar at the bottom indicates 'Query executed successfully.' and shows the connection details: 'DESKTOP-6JBC82E (15.0 RTM) | DESKTOP-6JBC82E\User (53) | MVC8CrudDb | 00:00:00 | 2 rows'.

DELETE method output

localhost:5126/swagger/index.html

DELETE /api/Category/{CategoryId}

Parameters

Name	Description
CategoryId * required	

integer(\$int32)
(path)

2

Execute

Clear

Responses

Curl

curl -X 'DELETE' \n'http://localhost:5126/api/Category/2' \n-H 'accept: */*'

Request URL

http://localhost:5126/api/Category/2

Server response

Code	Details
200	<div>Response headers</div> <div>content-length: 0 date: Sat, 17 Aug 2024 21:58:05 GMT server: Kestrel</div>

Responses

Code	Description	Links
200	Success	No links

DELETE method changes in database

The screenshot displays the SQL Server Enterprise Manager interface. On the left, the Object Explorer shows the database structure for 'DESKTOP-6JBC82E (SQL Server 15.0.2000.5 - DESKTOP-6JBC82E)'. The 'Tables' folder is expanded, showing 'dbo.Categories' and 'dbo.Products'. The 'Messages' tab is active in the bottom pane, showing the results of a DELETE query. The query is a 'SELECT TOP (1000) [CategoryId], [CategoryName] FROM [MVC8CrudDb].[dbo].[Categories]'.

SQLQuery3.sql - DE...-6JBC82E\User (53))* X SQLQuery2.sql - DE...-6JBC82E\User (60)) SQLQuery1.sql - DE...-6JBC82E\User (54))

```
SELECT TOP (1000) [CategoryId]
, [CategoryName]
FROM [MVC8CrudDb].[dbo].[Categories]
```

100 %

Results Messages

	CategoryId	CategoryName
1	4	EARPODS

WEB API Database Interaction.

localhost:5126/swagger/index.html

DELETE /api/Category/{CategoryId}

```
Microsoft.EntityFrameworkCore.Database.Command[20101]
info: Executed DbCommand (2ms) [Parameters=[@__CategoryId_0='?' (DbType = Int32)], CommandType='Text', CommandTimeout='30']
SELECT TOP(1) [c].[CategoryId], [c].[CategoryName]
FROM [Categories] AS [c]
WHERE [c].[CategoryId] = @__CategoryId_0
Microsoft.EntityFrameworkCore.Database.Command[20101]
info: Executed DbCommand (4ms) [Parameters=[@__p_0='?' (DbType = Int32)], CommandType='Text', CommandTimeout='30']
SELECT TOP(1) [c].[CategoryId], [c].[CategoryName]
FROM [Categories] AS [c]
WHERE [c].[CategoryId] = @__p_0
Microsoft.EntityFrameworkCore.Database.Command[20101]
info: Executed DbCommand (9ms) [Parameters=[@p0='?' (DbType = Int32)], CommandType='Text', CommandTimeout='30']
SET IMPLICIT_TRANSACTIONS OFF;
SET NOCOUNT ON;
DELETE FROM [Categories]
OUTPUT 1
WHERE [CategoryId] = @p0;
```

Code	Description	Links
200	Success	No links

Part 3: Integration and Testing

Controller class with set of integration test

The screenshot displays the Visual Studio IDE with the following components:

- Code Editor:** Shows the `CategoryControllerTest.cs` file. The code defines a test class `CategoryControllerTest` within the `UnitTestCaseDemo.Test` namespace. It includes a constructor that initializes `_controller` and `_service`, and a test method `GetAll_Category_Success()` using the `[Fact]` attribute. The test method includes `Arrange`, `Act`, and `Assert` sections.
- Solution Explorer:** Located on the right, it shows the project structure for `WebAPIMVC8CRUD`. The file `CategoryControllerTest.cs` is highlighted under the `Controllers` folder.
- Properties Window:** Below the Solution Explorer, it shows the properties for `CategoryControllerTest.cs`, including the `Build Action` set to `C# compiler`.
- Output Window:** At the bottom, it shows the build output for the project `WebAPIMVC8CRUD.csproj`. The output indicates that the build was successful, with 2 tests succeeded, 0 failed, and 0 up-to-date or skipped.

```
namespace UnitTestCaseDemo.Test
{
    public class CategoryControllerTest
    {
        CategoryController _controller;
        ICategoryService _service;

        public CategoryControllerTest()
        {
            _service = new CategoryService();
            _controller = new CategoryController(_service);
        }

        [Fact]
        public void GetAll_Category_Success()
        {
            //Arrange

            //Act
            var result = _controller.Get();
            var resultType = result as OkObjectResult;
            var resultList = resultType.Value as List<Category>;

            //Assert
            Assert.NotNull(result);
            Assert.IsType<List<Category>>(resultType.Value);
            Assert.Equal(3, resultList.Count);
        }
    }
}
```

Build Output:

```
1>Done building project "WebAPIMVC8CRUD.csproj".
2>----- Build started: Project: UnitTestWebAPIMVC8CRUD, Configuration: Debug Any CPU -----
2>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.
2>UnitTestWebAPIMVC8CRUD -> C:\Users\User\source\repos\WebAPIMVC8CRUD\UnitTestWebAPIMVC8CRUD\bin\Debug\net8.0\UnitTestWebAPIMVC8CRUD.dll
===== Build: 2 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Build completed at 7:18 PM and took 15.930 seconds =====
```

Xunit test result.

FileEditViewGitProjectBuildDebugTestAnalyzeToolsExtensionsWindowHelp

WebAPIMVC8CRUD

DebugAny CPUWebAPIMVC8CRUDhttps

Server Explorer

UnitTestWebAPIMVC8CRUDCategoryControllerTest.csUnitTest1.csICategoryService.csCategoryService.csCategory.cs

Test Explorer

Ready

550

0 Warnings0 Errors

Test	Duration	Traits	Error Message
✔ UnitTestWebAPIMVC8CRUD (5)	379 ms		
✔ UnitTestCaseDemo.Test (4)	374 ms		
✔ CategoryControllerTest (4)	374 ms		
✔ Add_Category_Success	10 ms		
✔ Delete_Category_Success	< 1 ms		
✔ GetAll_Category_Success	< 1 ms		
✔ GetById_Category_Success	364 ms		
✔ UnitTestWebAPIMVC8CRUD (1)	5 ms		
✔ UnitTest1 (1)	5 ms		

RunDebug

Group Summary

UnitTestWebAPIMVC8CRUD

Tests in group: 5

Total Duration: 379 ms

Outcomes

5 Passed

Solution Explorer

Search Solution Explorer (Ctrl+)

Solution 'WebAPIMVC8CRUD' (2 of 2 projects)

✔ UnitTestWebAPIMVC8CRUD

Dependencies

C# CategoryControllerTest.cs

C# UnitTest1.cs

✔ WebAPIMVC8CRUD

Connected Services

Dependencies

Properties

Controllers

C# CategoryController.cs

C# WeatherForecastController.cs

Model

C# Category.cs

Services

Solution ExplorerGit Changes

Properties

CategoryControllerTest.cs File Properties

Advanced

Build ActionC# compiler

Copy to Output DirectoryDo not copy

Custom Tool

Custom Tool Namespace

Misc

File NameCategoryControllerTest.cs

Build Action

How the file relates to the build and deployment processes.

75%

Ln: 82Ch: 35SPC CRLF

Output

Show output from: Build

1>Done building project "WebAPIMVC8CRUD.csproj".

2>----- Build started: Project: UnitTestWebAPIMVC8CRUD, Configuration: Debug Any CPU -----

2>Skipping analyzers to speed up the build. You can execute 'Build' or 'Rebuild' command to run analyzers.

2>UnitTestWebAPIMVC8CRUD -> C:\Users\User\source\repos\WebAPIMVC8CRUD\UnitTestWebAPIMVC8CRUD\bin\Debug\net8.0\UnitTestWebAPIMVC8CRUD.dll

===== Build: 2 succeeded, 0 failed, 0 up-to-date, 0 skipped =====

===== Build completed at 7:18 PM and took 15.930 seconds =====

Error ListOutput

Test run finished: 5 Tests (5 Passed, 0 Failed, 0 Skipped) run in 610 ms

Add to Source ControlSelect Repository

Here are the details for running the solutions locally:

Note: Please update **appsettings.json** file with the database string connection of your local SQL server

MVC8CRUD: This is an ASP.NET MVC web application solution. Open the `.sln` file in the ZIP archive to run it.

WEBAPIMVC8CRUD: This is an ASP.NET Web API solution. Open the `.sln` file in the ZIP archive to run it.

UnitTestWebAPIMVC8CRUD: This uses the XUnit testing framework. Open the `.sln` file in the ZIP archive to run it.