Asp.Net With Web API

UNIT-6

Introduction To REST

REST (Representational State Transfer) is an architectural style for designing networked applications. It was introduced by Roy Fielding in his doctoral dissertation in 2000. RESTful systems typically use HTTP for communication and adhere to a set of principles that define how resources are identified and addressed.

In ASP.NET Web API, you can create RESTful services easily. Here's an introduction to REST principles in the context of ASP.NET Web API:

1. Resource Identification: In REST, everything is considered a resource. Resources are identified by URIs (Uniform Resource Identifiers). For example, `/api/products` could represent a collection of products, and `/api/products/{id}` could represent a specific product identified by its ID.

1. HTTP Verbs: HTTP verbs (GET, POST, PUT, DELETE, etc.) are used to perform actions on resources. Each HTTP verb corresponds to a CRUD (Create, Read, Update, Delete) operation on resources.
2. Stateless Communication: RESTful APIs are stateless, meaning each request from a client must contain all the information necessary to understand and process the request. The server should not store any client state between requests.
3. Representation: Resources can have different representations. Common representations include JSON, XML, HTML, and plain text. Clients can specify their preferred representation using HTTP headers (e.g., `Accept` header).
4. Uniform Interface: RESTful APIs should have a uniform interface to promote simplicity and scalability. This includes using standard HTTP methods, following resource naming conventions, and providing consistent response formats.

INTRODUCTION TO ASP.NET WEB API

ASP.NET Web API is a framework for building HTTP services that can reach a broad range of clients, including browsers and mobile devices. It's part of the ASP.NET platform and provides a simple, yet powerful, way to build APIs that follow RESTful principles.

1. HTTP-Centric: ASP.NET Web API is built around HTTP and provides full support for HTTP features, including content negotiation, request and response headers, caching, and more. This makes it ideal for building RESTful APIs.
2. Routing: Web API uses routing to map HTTP requests to controller actions. Routing in Web API is similar to ASP.NET MVC routing but optimized for HTTP services. You can define routes using attributes or by configuring routes in a routing table.
3. Content Negotiation: Web API supports content negotiation, allowing clients to request data in various formats such as JSON, XML, or even custom media types. This is achieved through the use of HTTP headers like `Accept` and `Content-Type`.

INTRODUCTION TO ASP.NET WEB API

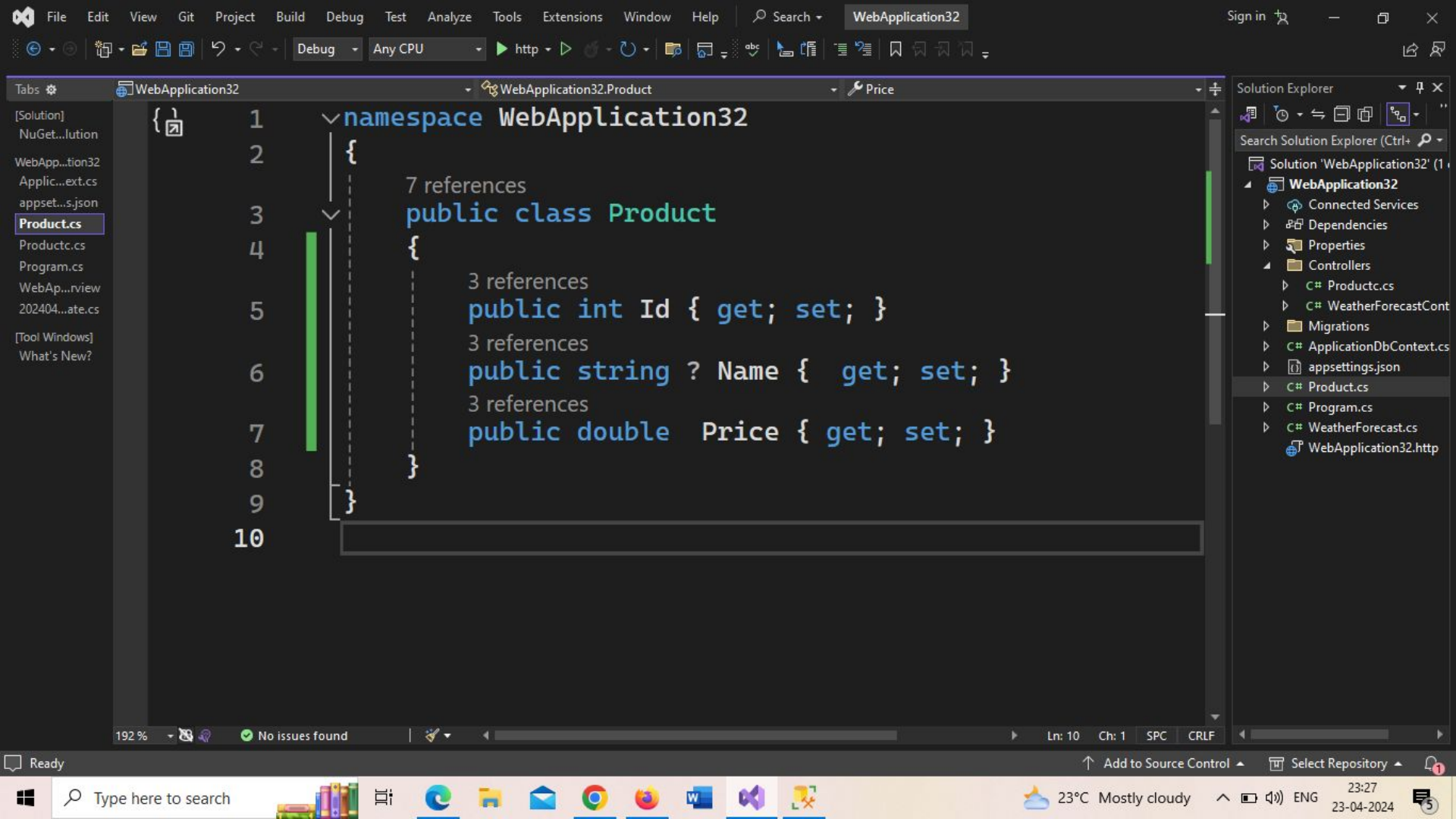
4. Model Binding and Validation: Web API provides powerful model binding capabilities, allowing you to bind data from HTTP requests directly to action method parameters. It also includes built-in support for model validation, making it easy to validate incoming data.
5. ActionResult: In Web API, action methods return instances of `IHttpActionResult`, which represent the result of an HTTP operation. This allows you to return different types of responses, such as `Ok`, `NotFound`, `BadRequest`, etc., based on the outcome of the operation.
6. Filters and Middleware: Web API includes a rich set of filters and middleware that allow you to add cross-cutting concerns such as authentication, authorization, logging, exception handling, and caching to your API.
7. Dependency Injection: Web API supports dependency injection out of the box, allowing you to inject dependencies into your controllers, filters, and other components using constructor injection.
8. Testability: Web API is designed with testability in mind. You can easily write unit tests for your controllers and integration tests for your API using frameworks like NUnit, xUnit, or MSTest.

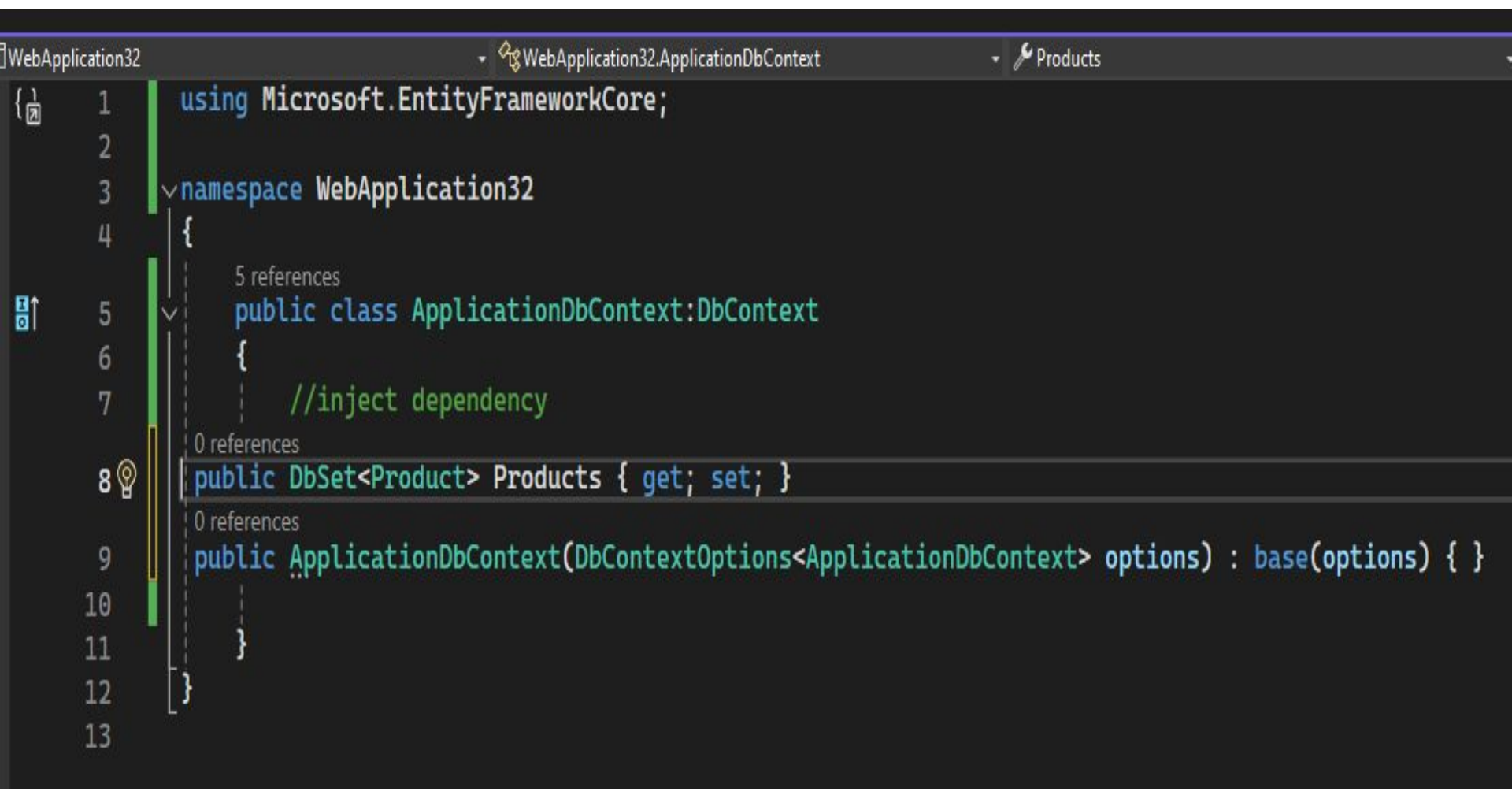
Code First Approach

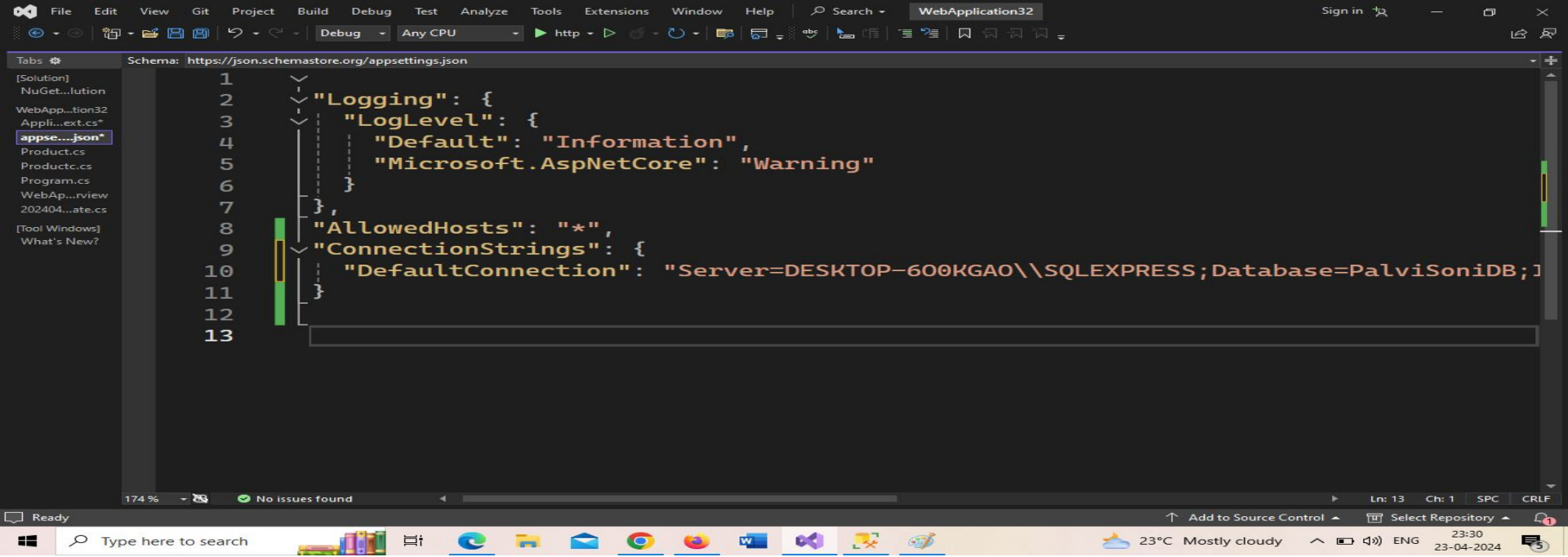
a) First of all, you have to create one model class with name “Product.cs”.

   Add Id,Name and price as entities.

b) Secondly, you can create one controller named as “Productc.cs”.
c) Now again you have to create one class named as “ApplicationDbContext”, here you will creating dependencies.
d) Now in appsetting.json, you will have to write connection string.
e) One important thing, you will have to install nuget packages for ef core, ef tools, and sqlserver(that already we have discussed in Unit4)


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
WebApplication32
Sign in
Debug
Any CPU
http
Tabs
[Solution]
NuGet...lution
WebApp...tion32
Applic...ext.cs
appset...s.json
Product.cs
Productc.cs
Program.cs
WebAp...rview
202404...ate.cs
[Tool Windows]
What's New?
WebApplication32
WebApplication32.Product
Price
1 namespace WebApplication32
2 {
7 references
3 public class Product
4 {
3 references
5 public int Id { get; set; }
3 references
6 public string ? Name { get; set; }
3 references
7 public double Price { get; set; }
8 }
9 }
10
Solution Explorer
Search Solution Explorer (Ctrl+
Solution 'WebApplication32' (1
WebApplication32
Connected Services
Dependencies
Properties
Controllers
Productc.cs
WeatherForecastCont
Migrations
ApplicationDbContext.cs
appsettings.json
Product.cs
Program.cs
WeatherForecast.cs
WebApplication32.http
192 %
No issues found
Ln: 10
Ch: 1
SPC
CRLF
Ready
Add to Source Control
Select Repository
Type here to search
23°C Mostly cloudy
ENG
23:27
23-04-2024

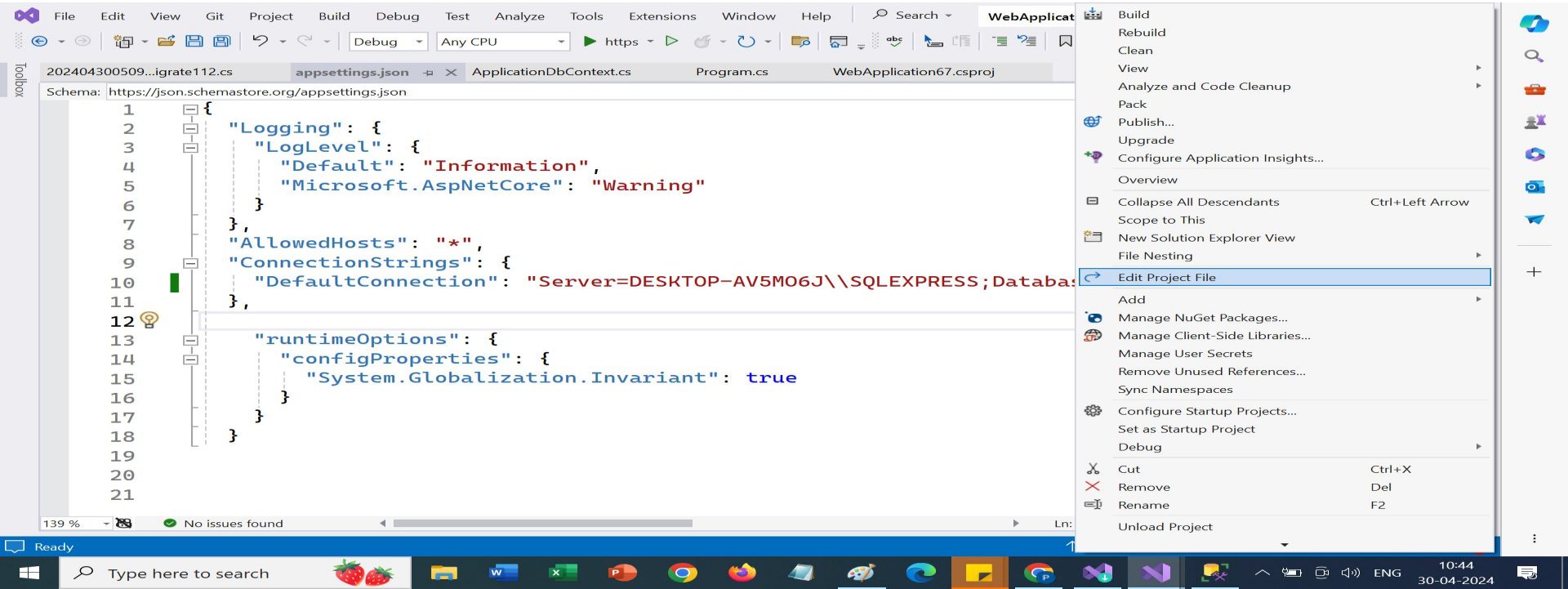

WebApplication32
WebApplication32.ApplicationDbContext
Products
using Microsoft.EntityFrameworkCore;
namespace WebApplication32
{
5 references
public class ApplicationDbContext:DbContext
{
//inject dependency
0 references
public DbSet<Product> Products { get; set; }
0 references
public ApplicationDbContext(DbContextOptions<ApplicationDbContext> options) : base(options) { }
}
}



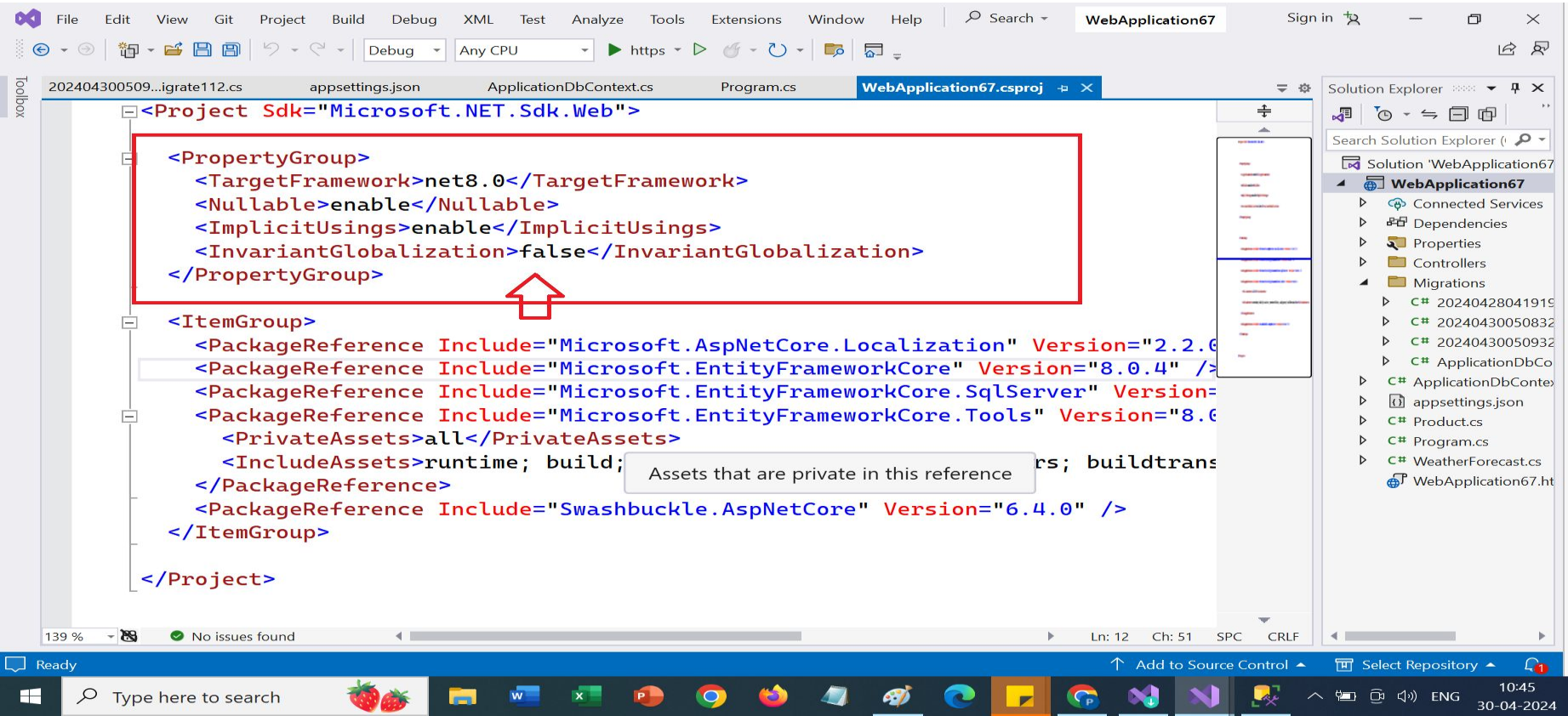

```
"ConnectionStrings": {
  "DefaultConnection": "Server=DESKTOP-6O0KGAO\\SQLEXPRESS;Database=PalviSoniDB;Integrated Security=true;TrustServerCertificate=true"
}
```

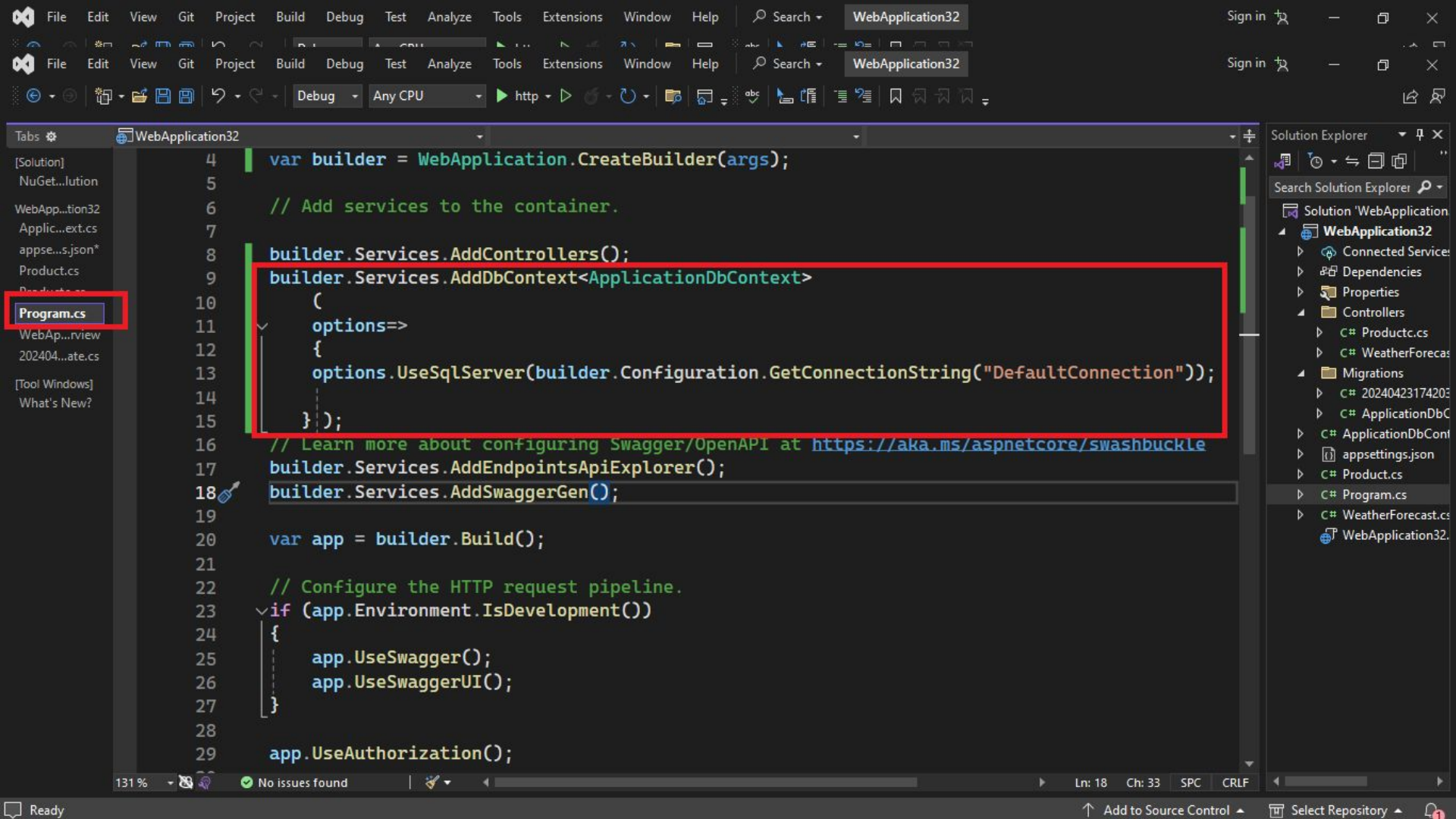
If your database is not updating and giving you globalization error

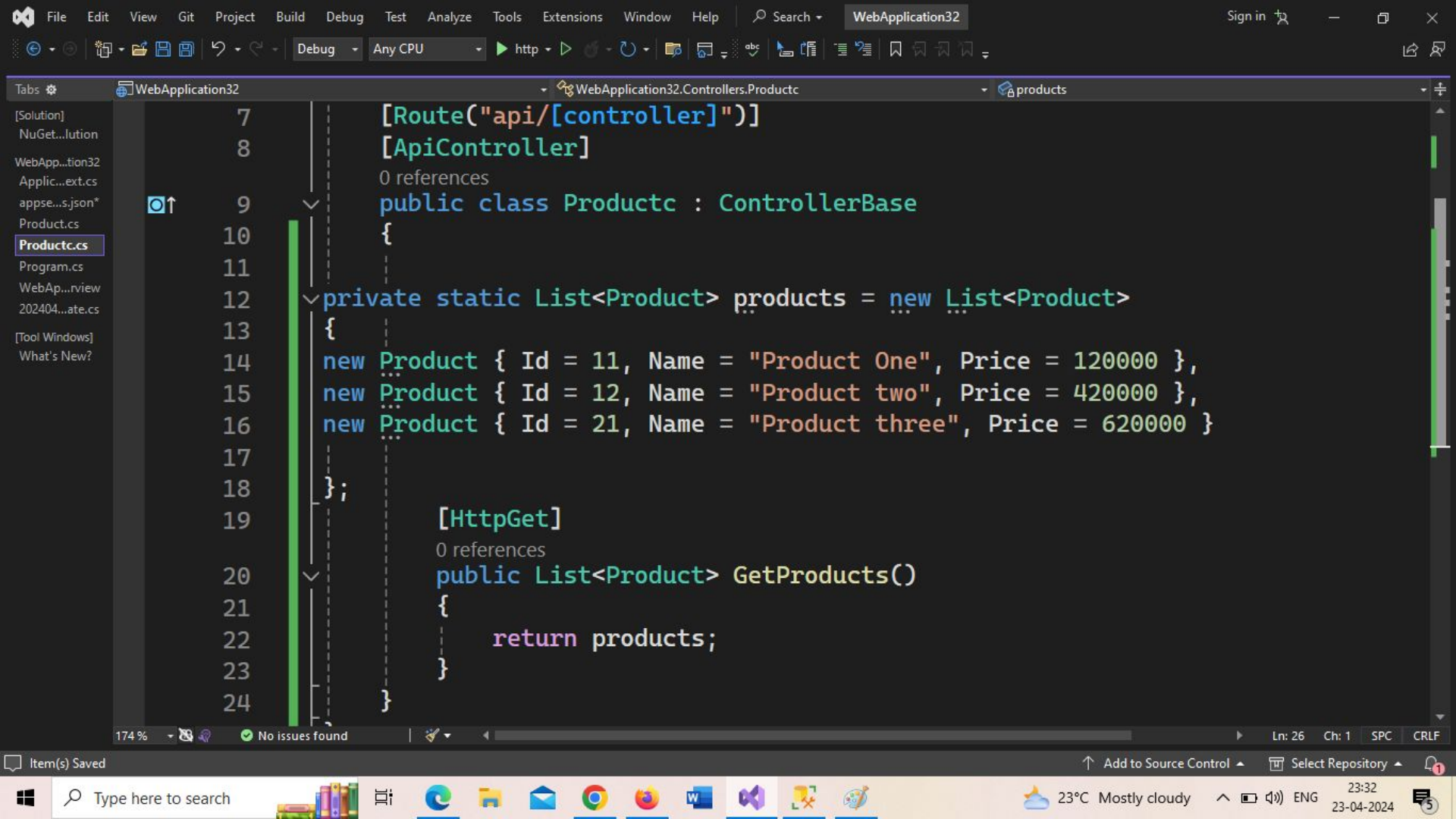
MAKE A RIGHT CLICK ON YOUR PROJECT AND CLICK EDIT PROJECT FILE



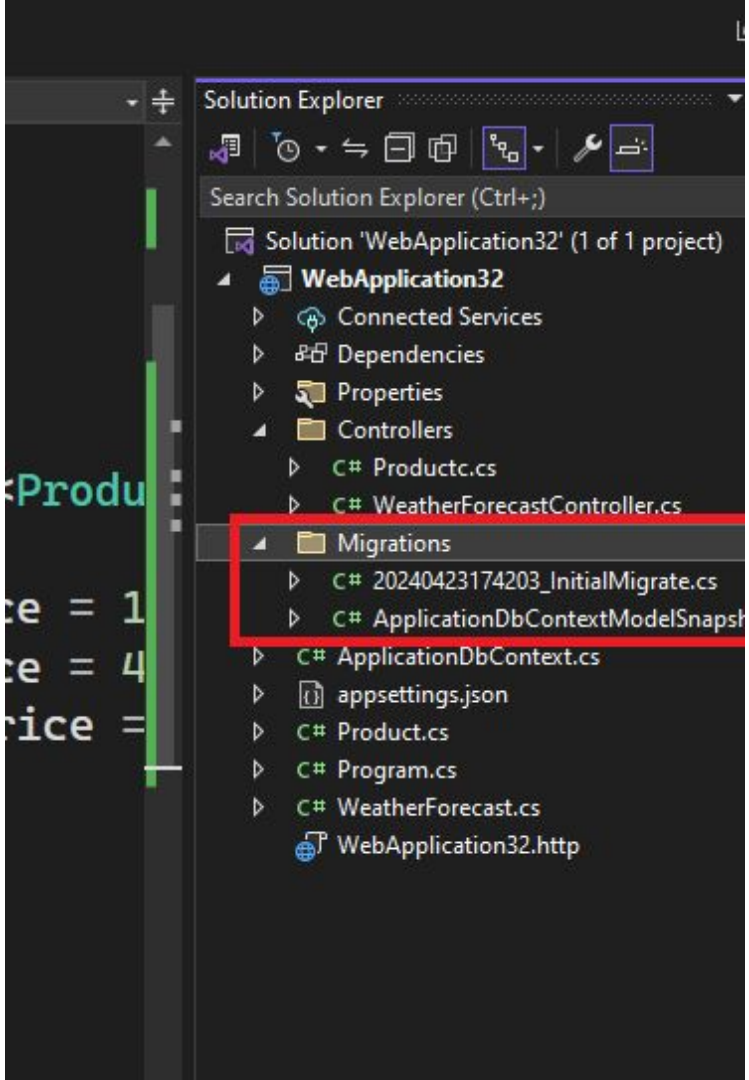
SetInvariantGlobalization false(initially it is true)-even if its not reflecting then also you can paste this in the project file



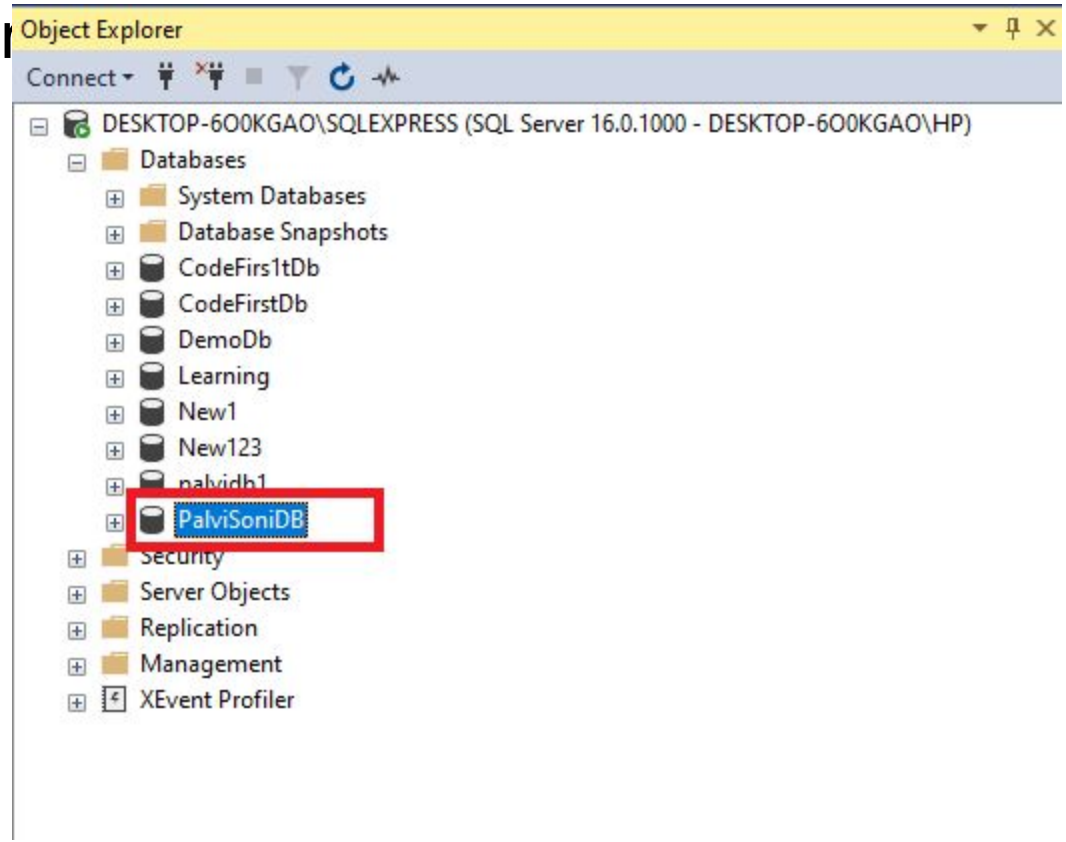

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
WebApplication32
Sign in
Debug
Any CPU
http
Tabs
[Solution]
NuGet...lution
WebApp...tion32
Applic...ext.cs
appse...s.json*
Product.cs
Program.cs
WebAp...rview
202404...ate.cs
[Tool Windows]
What's New?
var builder = WebApplication.CreateBuilder(args);
// Add services to the container.
builder.Services.AddControllers();
builder.Services.AddDbContext<ApplicationDbContext>
(
options=>
{
options.UseSqlServer(builder.Configuration.GetConnectionString("DefaultConnection"));
});
// Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle
builder.Services.AddEndpointsApiExplorer();
builder.Services.AddSwaggerGen();
var app = builder.Build();
// Configure the HTTP request pipeline.
if (app.Environment.IsDevelopment())
{
app.UseSwagger();
app.UseSwaggerUI();
}
app.UseAuthorization();
Solution Explorer
Search Solution Explorer
Solution 'WebApplication
WebApplication32
Connected Service
Dependencies
Properties
Controllers
Productc.cs
WeatherForeca
Migrations
ApplicationDbC
ApplicationDbCont
appsettings.json
Product.cs
Program.cs
WeatherForecast.cs
WebApplication32.
131 %
No issues found
Ln: 18
Ch: 33
SPC
CRLF
Ready
Add to Source Control
Select Repository

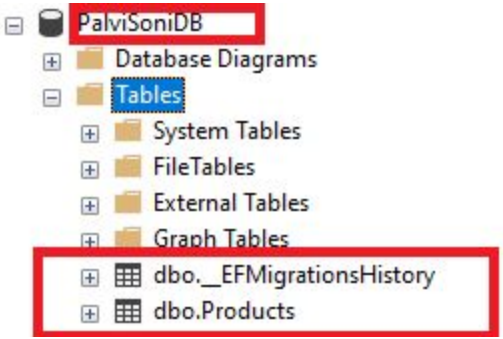

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
WebApplication32
Sign in
Debug
Any CPU
http
Tabs
WebApplication32
WebApplication32.Controllers.Productc
products
[Solution]
NuGet...lution
WebApp...tion32
Applic...ext.cs
appse...s.json*
Product.cs
Productc.cs
Program.cs
WebAp...rview
202404...ate.cs
[Tool Windows]
What's New?
[Route("api/[controller]")]
[ApiController]
0 references
public class Productc : ControllerBase
{
private static List<Product> products = new List<Product>
{
new Product { Id = 11, Name = "Product One", Price = 120000 },
new Product { Id = 12, Name = "Product two", Price = 420000 },
new Product { Id = 21, Name = "Product three", Price = 620000 }
};
[HttpGet]
0 references
public List<Product> GetProducts()
{
return products;
}
}
174 %
No issues found
Ln: 26
Ch: 1
SPC
CRLF
Item(s) Saved
Add to Source Control
Select Repository
Type here to search
23°C Mostly cloudy
ENG
23:32
23-04-2024

```
PM> add-migration InitialMigrate
Build started...
Build succeeded.
To undo this action, use Remove-Migration.
PM> update-database
Build started...
Build succeeded.
Microsoft.EntityFrameworkCore.Database.Command[20101]
      Executed DbCommand (557ms) [Parameters=[], CommandType='Text', Comma
      CREATE DATABASE [PalviSoniDB];
```

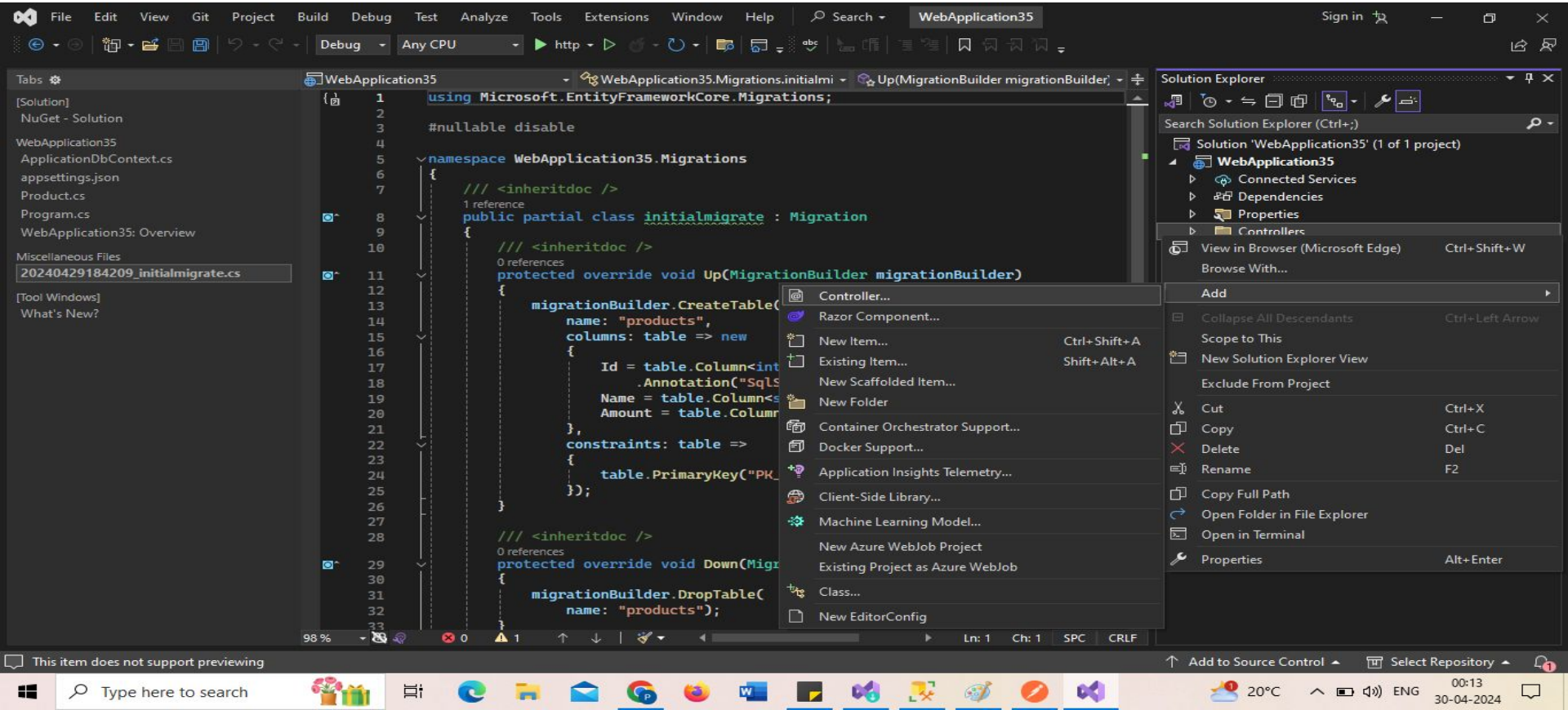

Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'WebApplication32' (1 of 1 project)
WebApplication32
Connected Services
Dependencies
Properties
Controllers
Productc.cs
WeatherForecastController.cs
Migrations
20240423174203_InitialMigrate.cs
ApplicationDbContextModelSnapsh
ApplicationDbContext.cs
appsettings.json
Product.cs
Program.cs
WeatherForecast.cs
WebApplication32.http

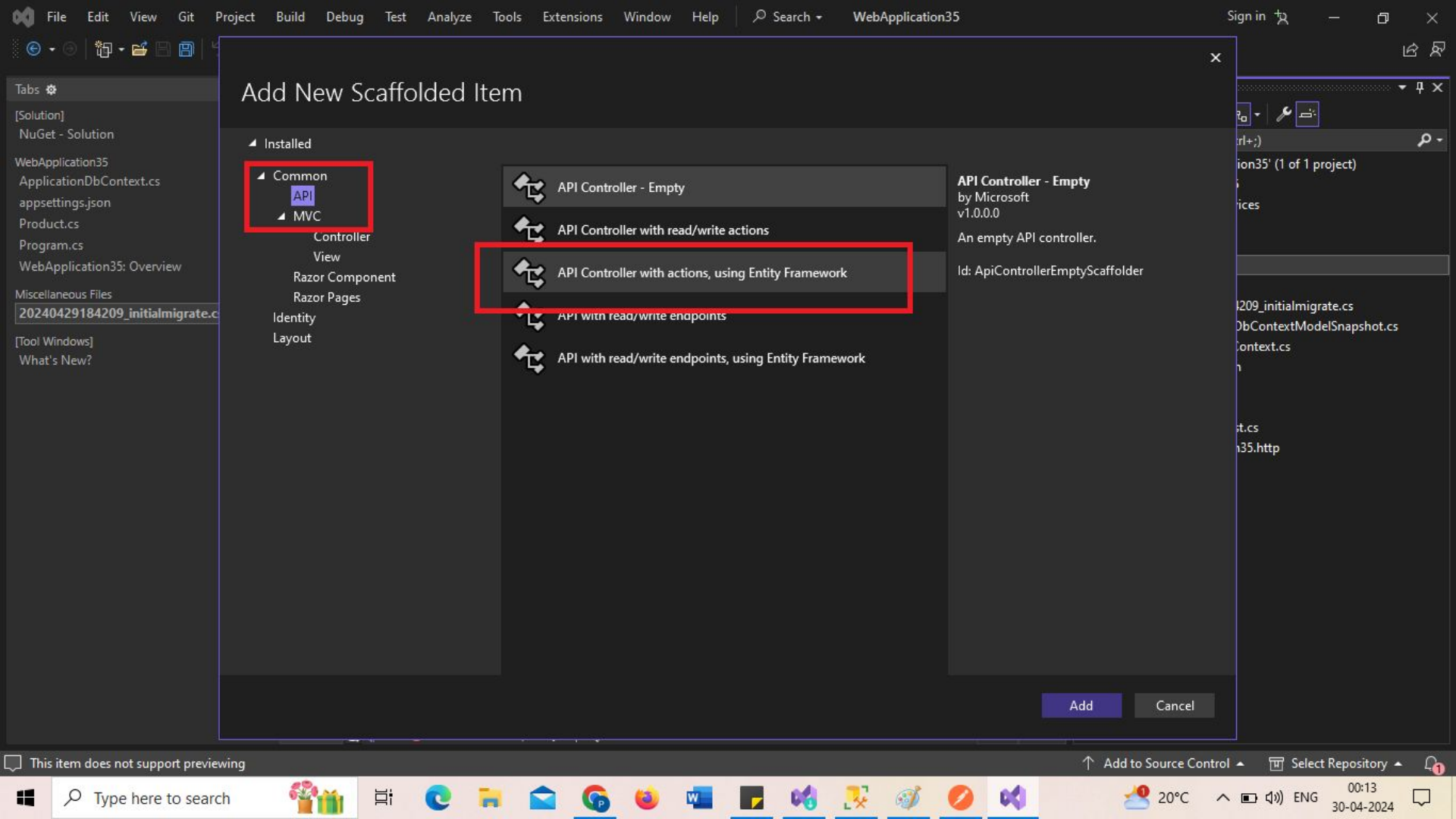
Database cr

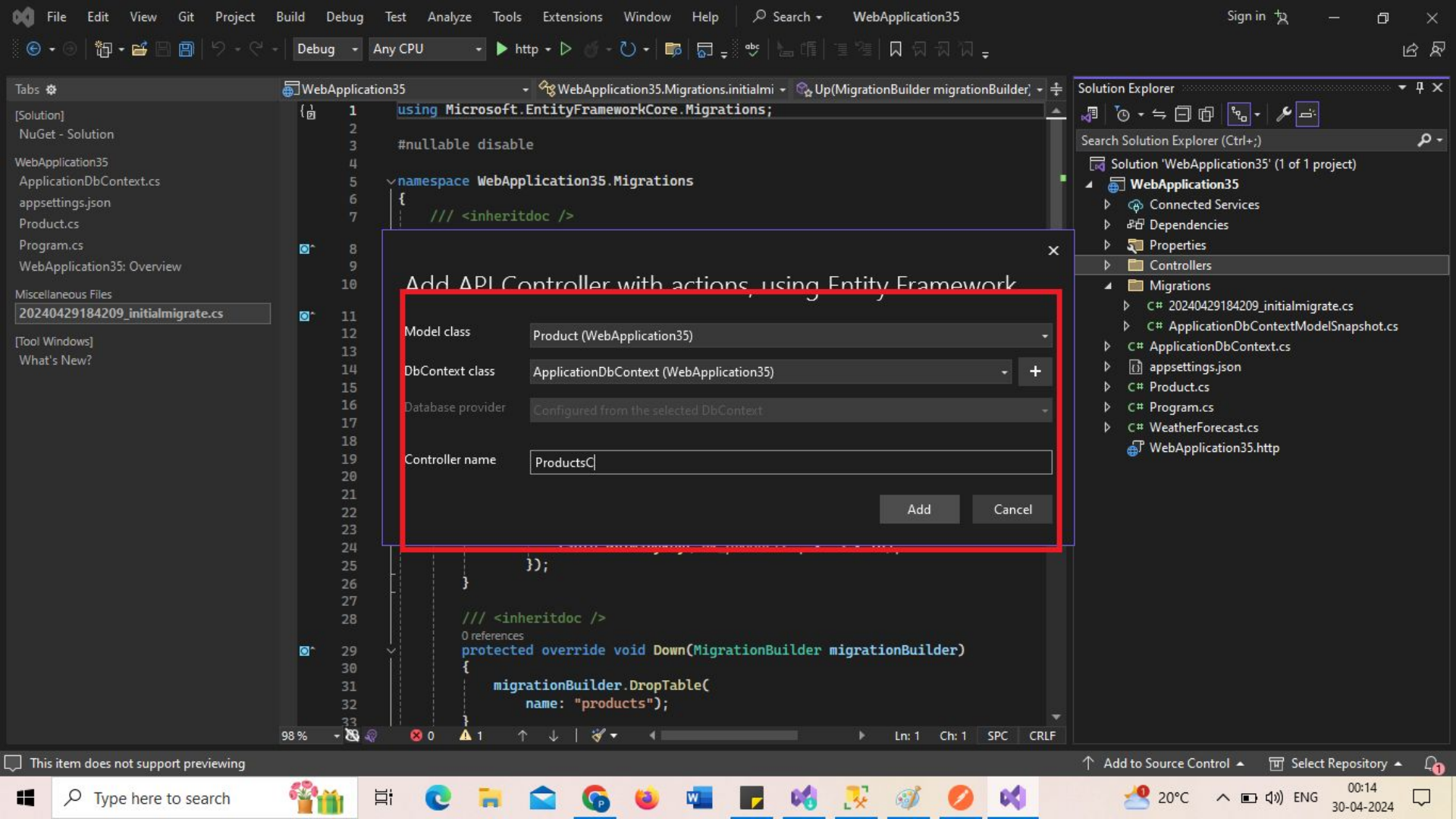

Object Explorer
Connect
DESKTOP-6O0KGAO\SQLEXPRESS (SQL Server 16.0.1000 - DESKTOP-6O0KGAO\HP)
Databases
System Databases
Database Snapshots
CodeFirs1tDb
CodeFirstDb
DemoDb
Learning
New1
New123
palvidb1
PalviSoniDB
Security
Server Objects
Replication
Management
XEvent Profiler


PalviSoniDB
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.__EFMigrationsHistory
dbo.Products

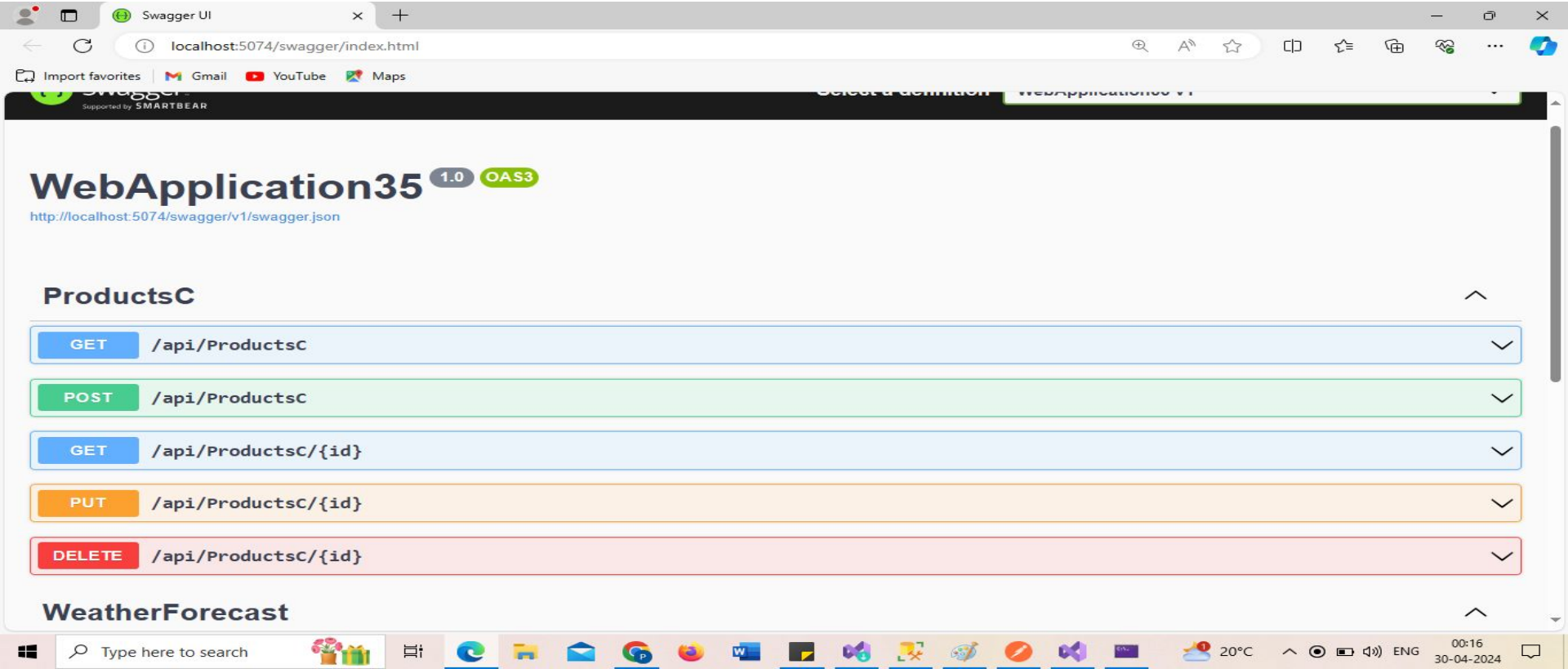
CRUD operations with Web Api



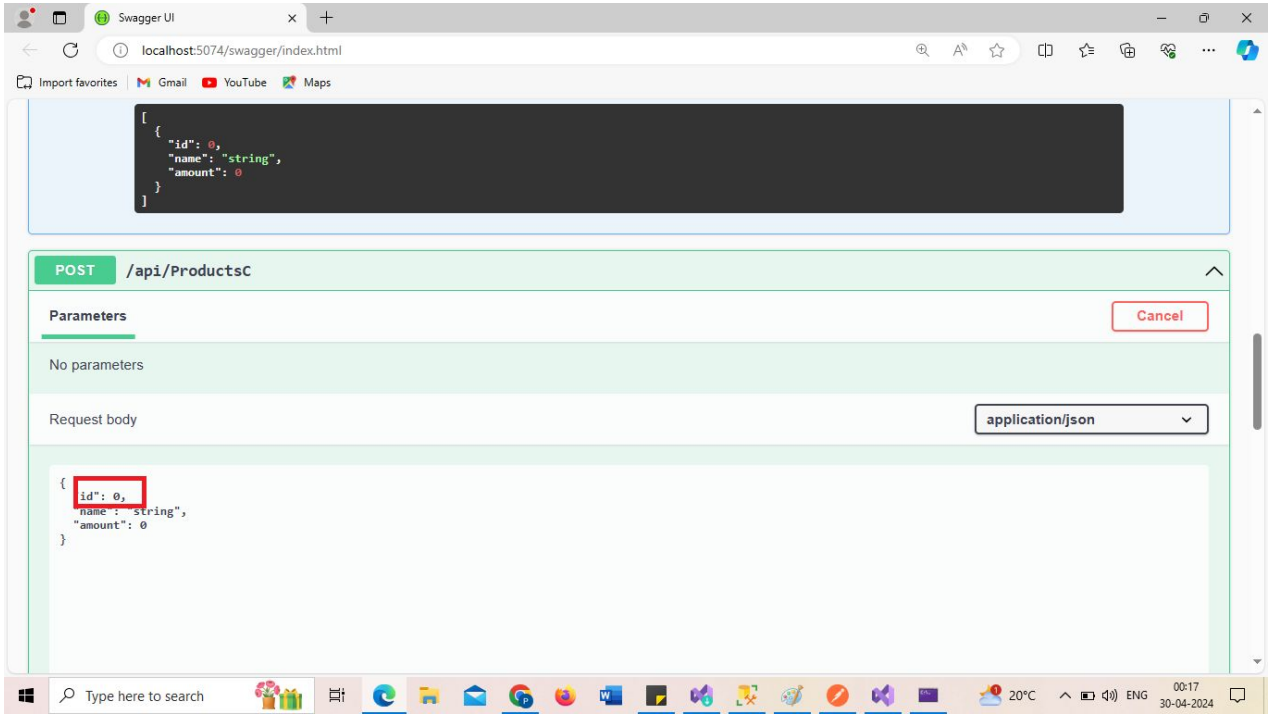

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
WebApplication35
Sign in
Tabs
[Solution]
NuGet - Solution
WebApplication35
ApplicationDbContext.cs
appsettings.json
Product.cs
Program.cs
WebApplication35: Overview
Miscellaneous Files
20240429184209_initialmigrate.c
[Tool Windows]
What's New?
Add New Scaffolded Item
Installed
Common
API
MVC
Controller
View
Razor Component
Razor Pages
Identity
Layout
API Controller - Empty
API Controller with read/write actions
API Controller with actions, using Entity Framework
API with read/write endpoints
API with read/write endpoints, using Entity Framework
API Controller - Empty
by Microsoft
v1.0.0.0
An empty API controller.
Id: ApiControllerEmptyScaffolder
Add
Cancel
This item does not support previewing
Add to Source Control
Select Repository
Type here to search
20°C
ENG
00:13
30-04-2024

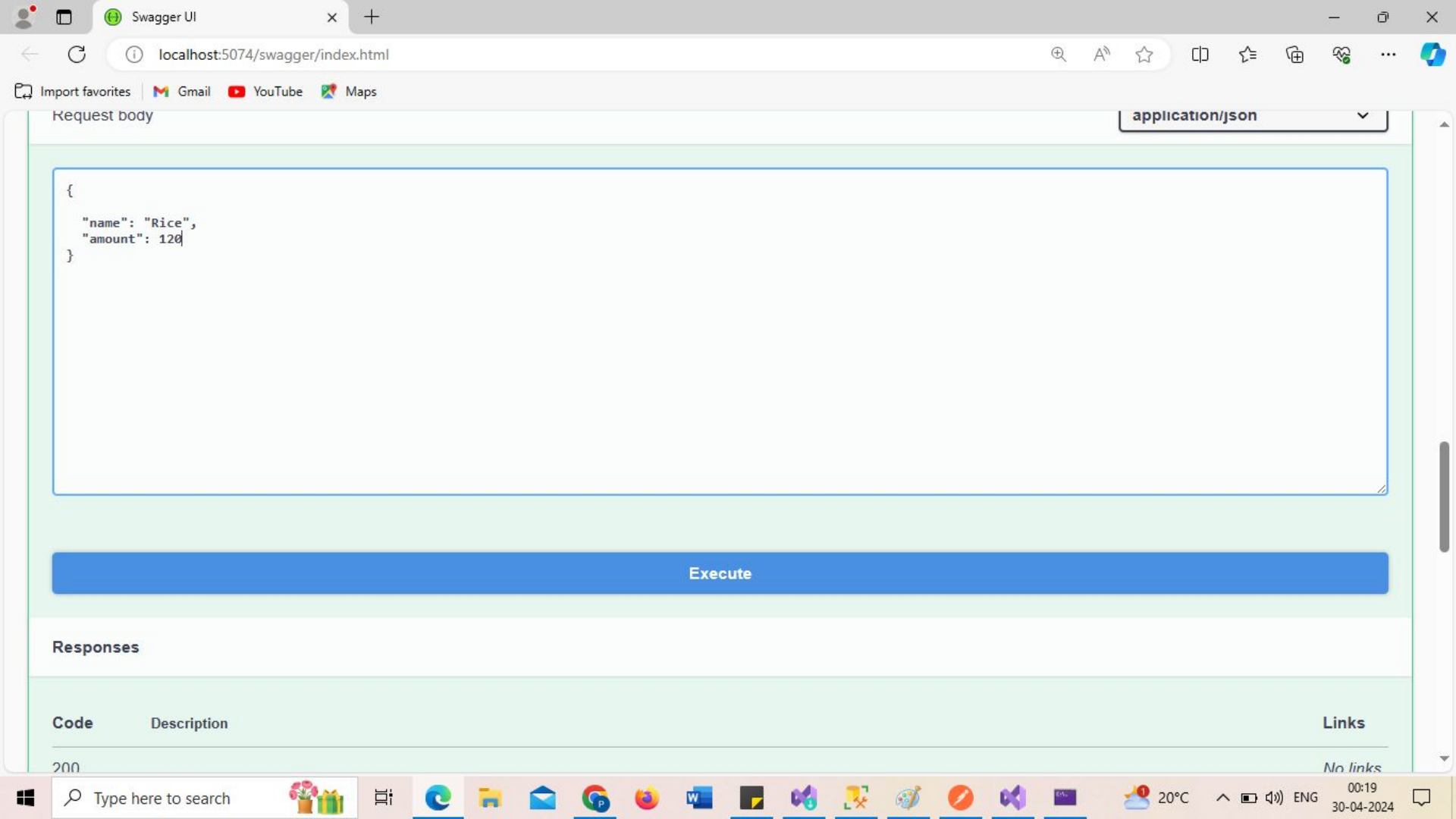

File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
Search
WebApplication35
Sign in
Debug
Any CPU
http
Tabs
[Solution]
NuGet - Solution
WebApplication35
ApplicationDbContext.cs
appsettings.json
Product.cs
Program.cs
WebApplication35: Overview
Miscellaneous Files
20240429184209_initialmigrate.cs
[Tool Windows]
What's New?
WebApplication35
WebApplication35.Migrations.initialmi
Up(MigrationBuilder migrationBuilder)
using Microsoft.EntityFrameworkCore.Migrations;
#nullable disable
namespace WebApplication35.Migrations
{
/// <inheritdoc />
Add API Controller with actions, using Entity Framework
Model class
Product (WebApplication35)
DbContext class
ApplicationDbContext (WebApplication35)
Database provider
Configured from the selected DbContext
Controller name
ProductsC
Add
Cancel
});
}
/// <inheritdoc />
0 references
protected override void Down(MigrationBuilder migrationBuilder)
{
migrationBuilder.DropTable(
name: "products");
}
98 %
0
1
Ln: 1
Ch: 1
SPC
CRLF
Solution Explorer
Search Solution Explorer (Ctrl+;)
Solution 'WebApplication35' (1 of 1 project)
WebApplication35
Connected Services
Dependencies
Properties
Controllers
Migrations
20240429184209_initialmigrate.cs
ApplicationDbContextModelSnapshot.cs
ApplicationDbContext.cs
appsettings.json
Product.cs
Program.cs
WeatherForecast.cs
WebApplication35.http
This item does not support previewing
Add to Source Control
Select Repository
Type here to search
20°C
ENG
00:14
30-04-2024

After running the project, you will see the following screen



In beginning, we have added nothing so in get method no values will be displayed.So try executing put




Swagger UI
localhost:5074/swagger/index.html
Import favorites
Gmail
YouTube
Maps
Request body
application/json
{
"name": "Rice",
"amount": 120
}
Execute
Responses
Code
Description
Links
200
No links
Type here to search
20°C
ENG
00:19
30-04-2024

Server response

Code	Details

201 *Undocumented*	**Response body**
	`{ "id": 1, "name": "Rice", "amount": 120 }` Download
	Response headers
	`content-type: application/json; charset=utf-8`
	`date: Mon,29 Apr 2024 18:49:33 GMT`
	`location: http://localhost:5074/api/ProductsC/1`
	`server: Kestrel`
	`transfer-encoding: chunked`

Responses

Code	Description	Links
200	Success Media type: text/plain Controls `Accept` header. Example Value | Schema	*No links*

Request body

application/json

```
{

  "name": "Wheat",
  "amount": 100
}
```

Execute | Clear

Responses

Curl

```
{
  "name": "Oats",
  "amount": 200
}
```

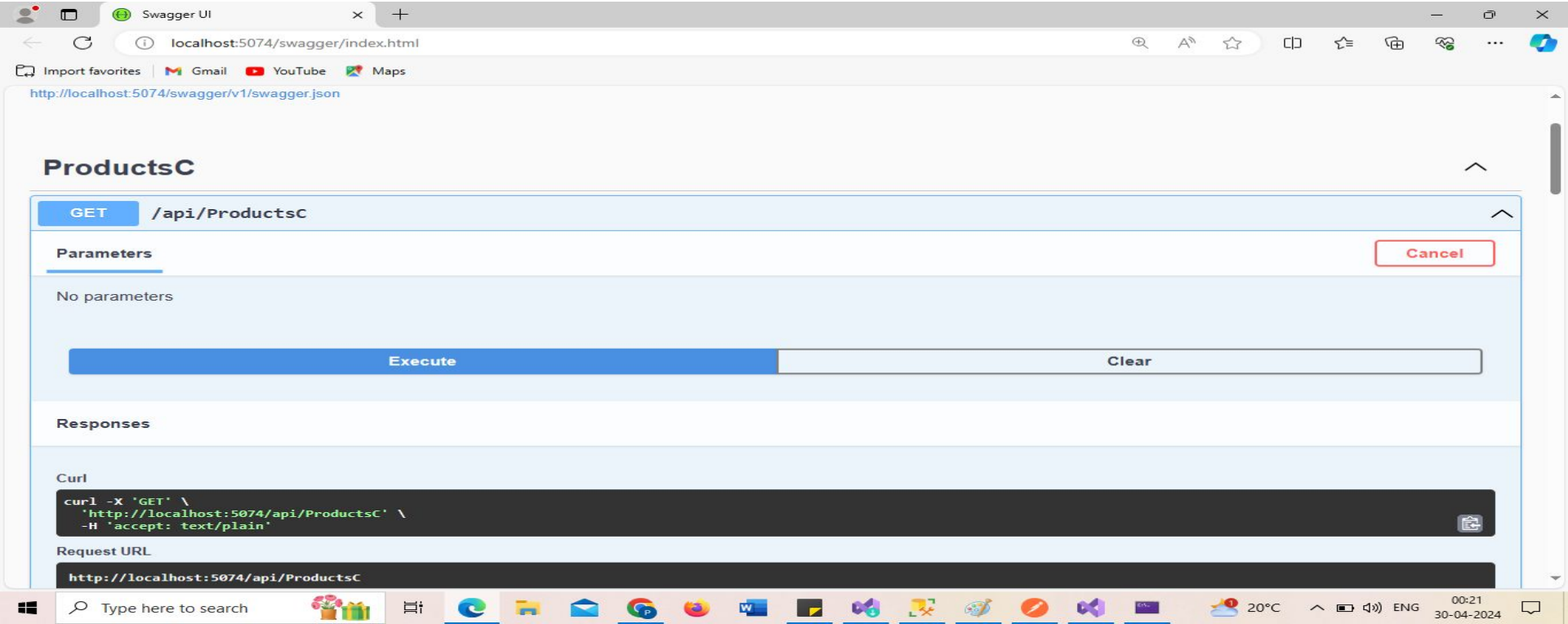
Execute | Clear

Responses

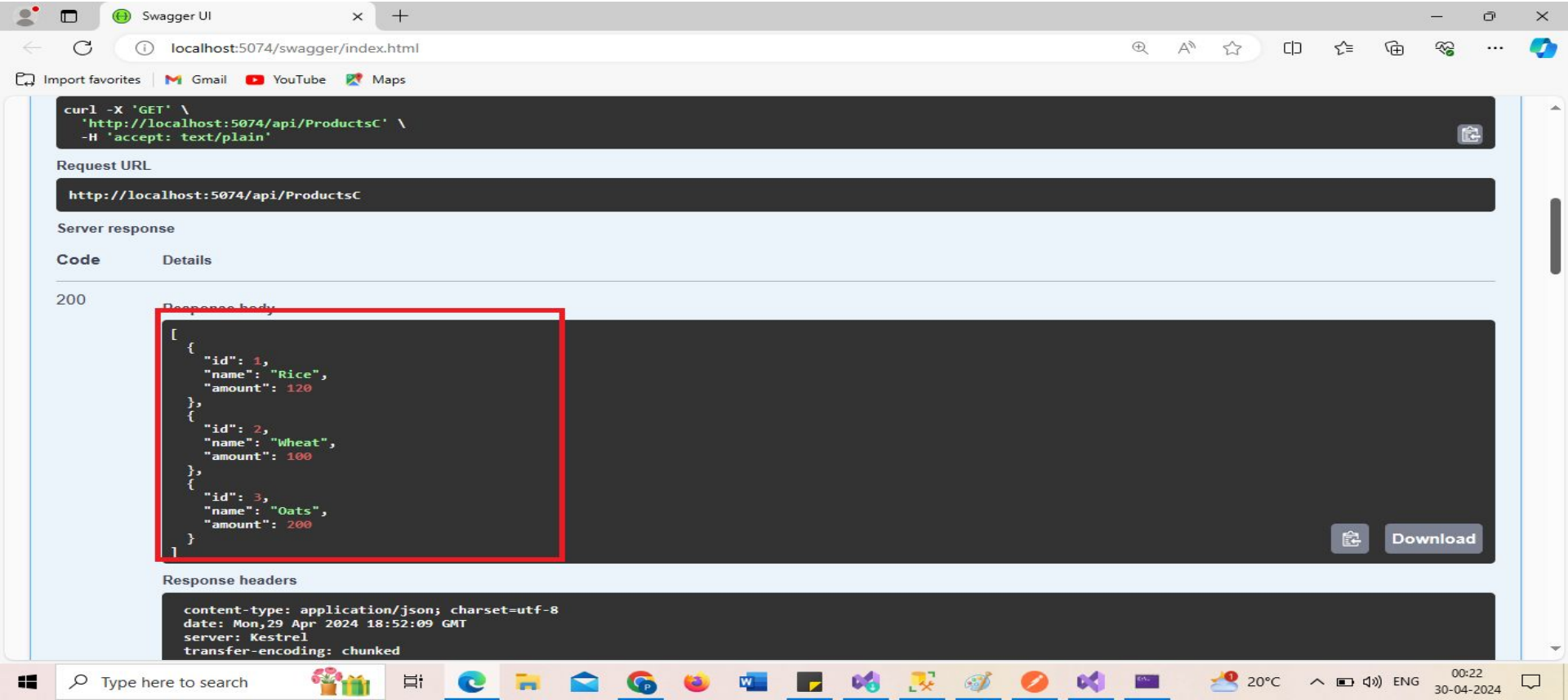
Curl

```
curl -X 'POST' \
  'http://localhost:5074/api/ProductsC' \
  -H 'accept: text/plain' \
  -H 'Content-Type: application/json' \
```

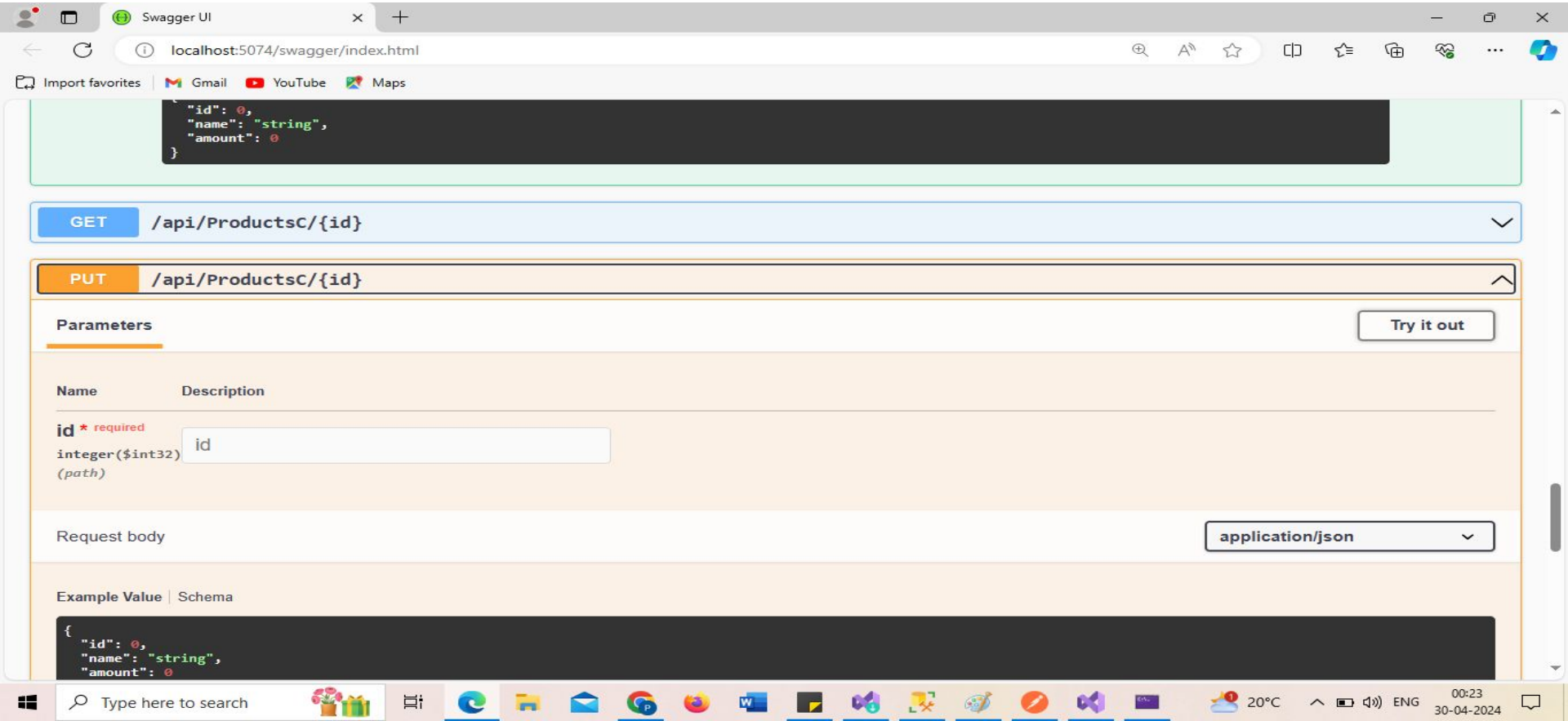
Now, execute get

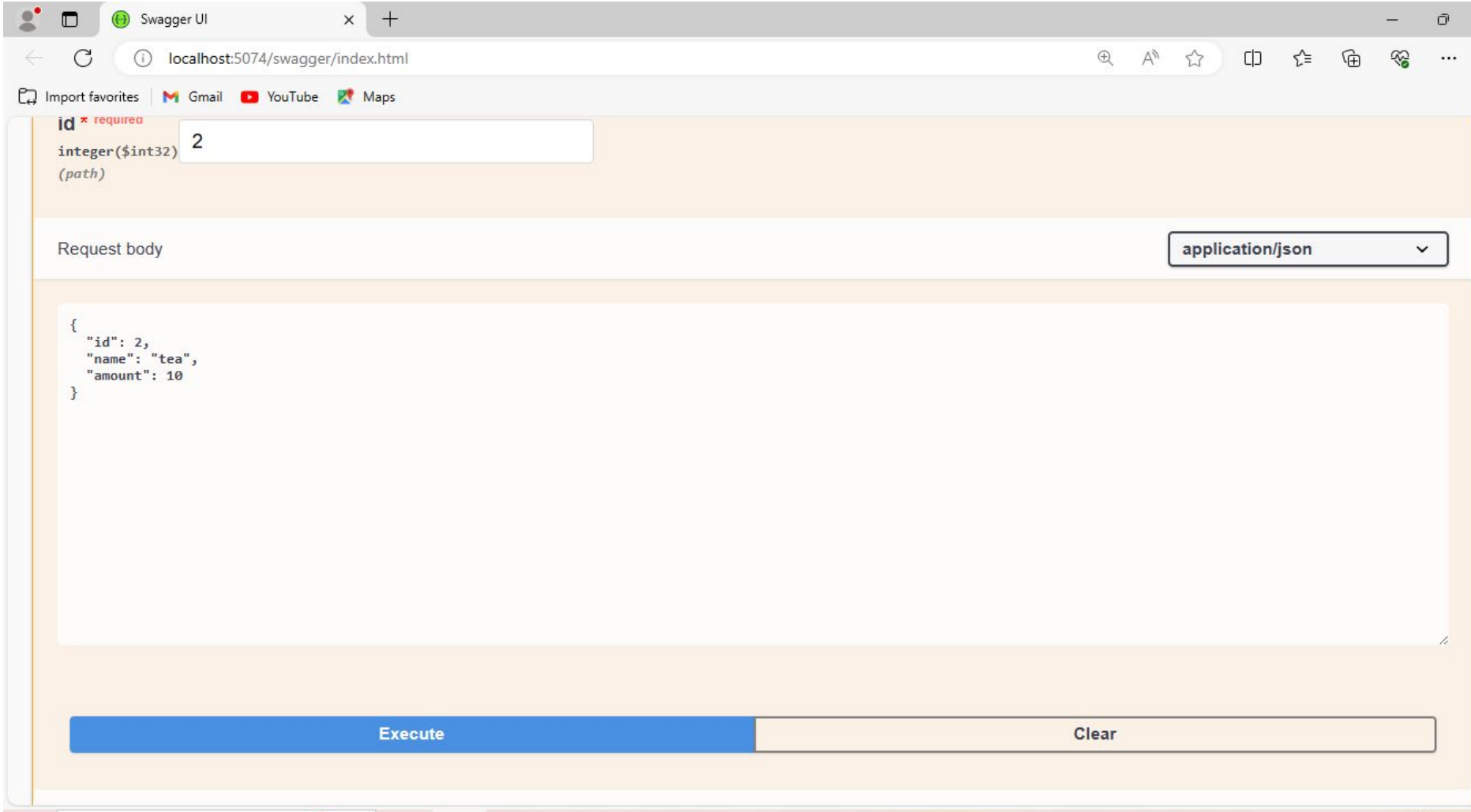


Now all items we are getting with get method



Now, we will use put method,click on try it out




Swagger UI
localhost:5074/swagger/index.html
Import favorites
Gmail
YouTube
Maps
id * required
integer($int32)
(path)
2
Request body
application/json
{
"id": 2,
"name": "tea",
"amount": 10
}
Execute
Clear

```
curl -X 'PUT' \
  'http://localhost:5074/api/ProductsC/2' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 2,
  "name": "tea",
  "amount": 10
}'
```

Request URL

```
http://localhost:5074/api/ProductsC/2
```

Server response

Code	Details

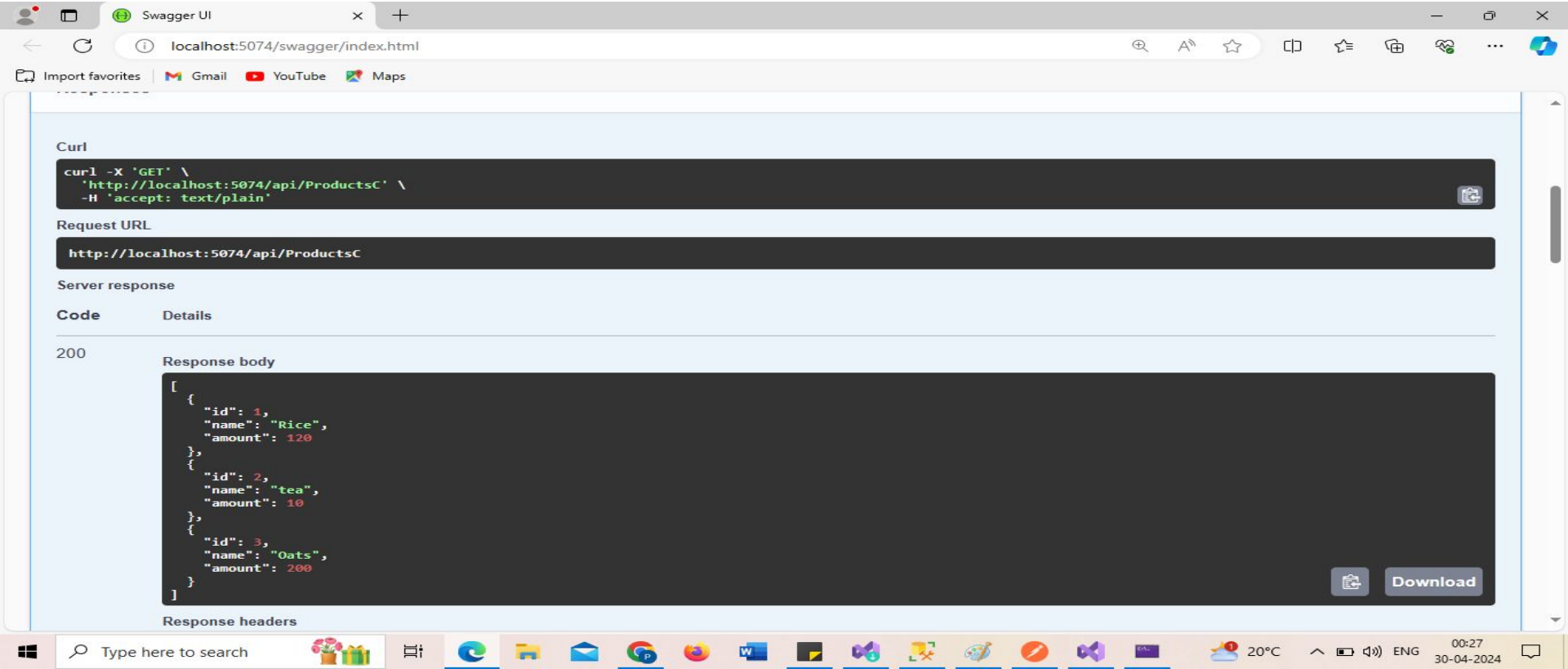
204 *Undocumented*	**Response headers**
	`date: Mon,29 Apr 2024 18:56:59 GMT`
	`server: Kestrel`

Responses

Code	Description	Links
200	Success	*No links*

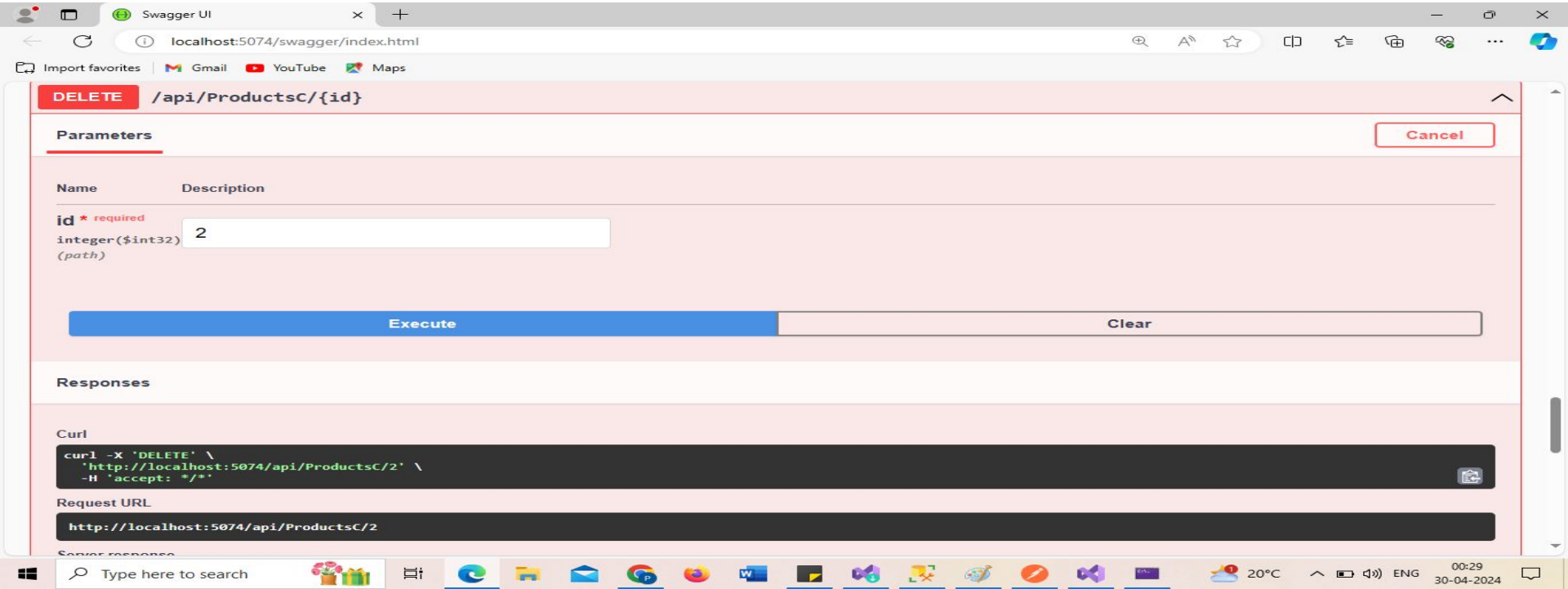
DELETE `/api/ProductsC/{id}`

Again you have to go to get method, for checking whether output is updated or not



Now we will perform delete operation

Just mention id that you want to delete



Responses

Curl

```
curl -X 'DELETE' \
  'http://localhost:5074/api/ProductsC/2' \
  -H 'accept: */*'
```

Request URL

```
http://localhost:5074/api/ProductsC/2
```

Server response

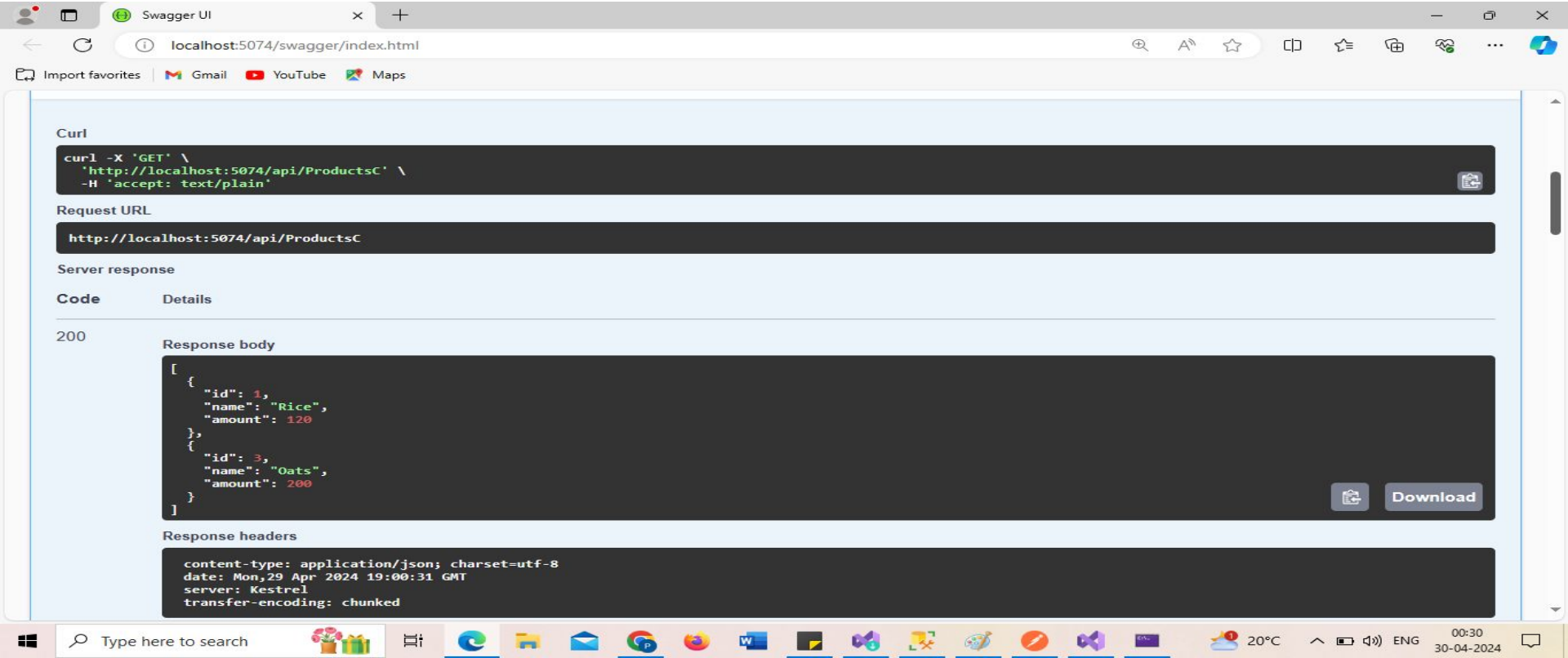
Code	Details
204 *Undocumented*	Response headers

```
date: Mon,29 Apr 2024 18:58:57 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	Success	*No links*

Again we will check get method, to know whether its deleted or not.Now we will see id 2 is deleted.

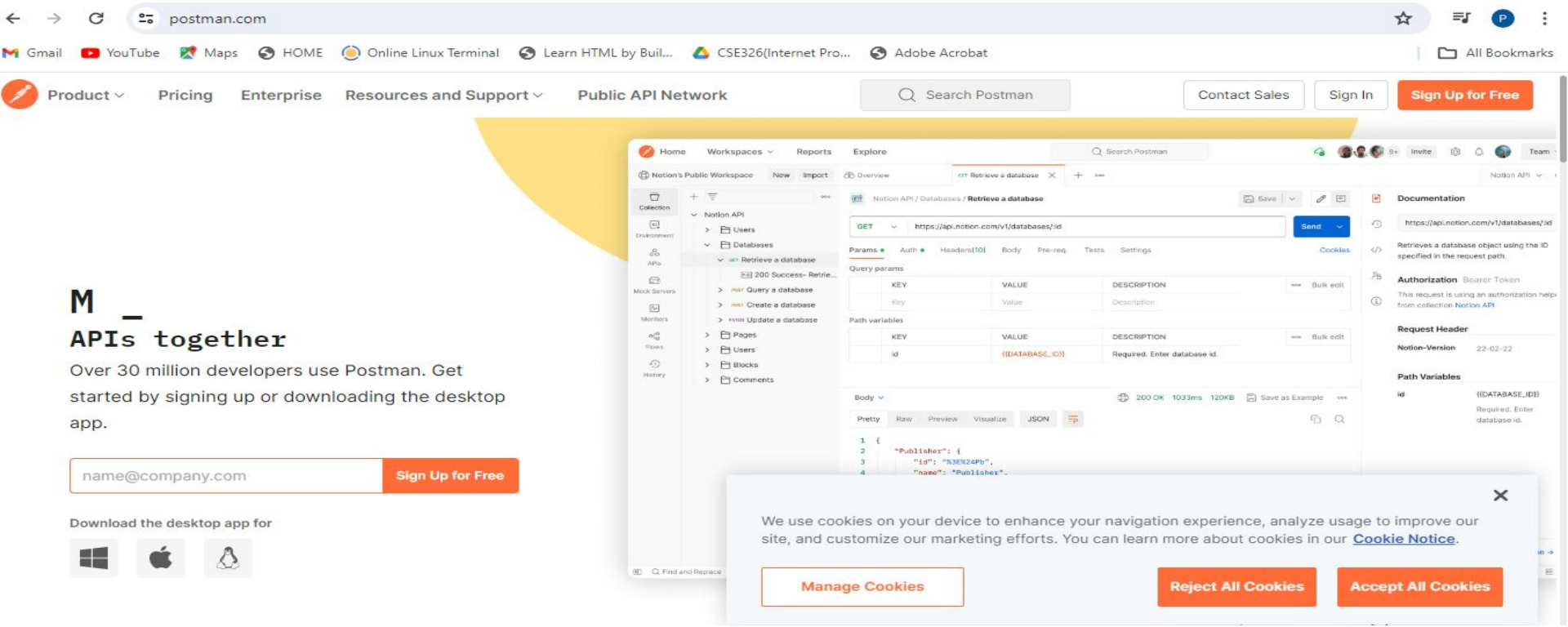


Testing Web API Using Postman-Used for basically testing API's, you project must be running first for testing whether API's are working or not.

POSTMAN

- APIs are commonly used in development.
- Postman is a most powerful HTTP Client that can be used for API Testing.
- It provides a user-friendly interface for developers to create, test, and document APIs.
- POSTMAN is used to send HTTP requests to web servers and receive responses.
- It supports various HTTP request methods such as GET, POST, PUT, DELETE, and others.
- With POSTMAN, developers can test the functionality and performance of APIs and diagnose any issues with them.

Testing Web API Using Postman



postman.com/downloads/?utm_source=postman-home

Gmail YouTube Maps HOME Online Linux Terminal Learn HTML by Buil... CSE326(Internet Pro... Ad

Product Pricing Enterprise Resources and Support Public API Network

Download the app to get started using the Postman API P
browser experience, you can try the web v

The Postman app

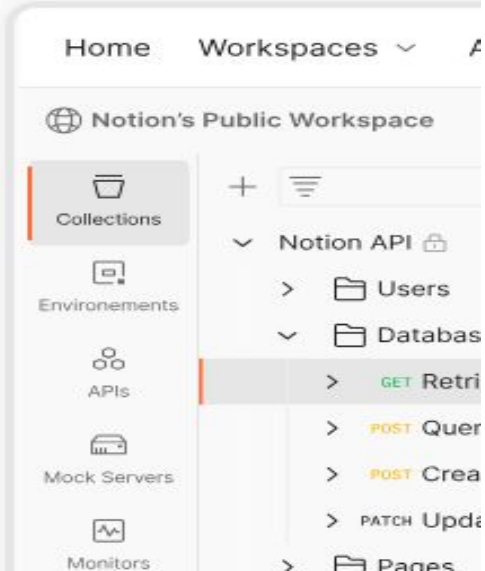
Download the app to get started with the Postman API Platform.

Windows 64-bit

By downloading and using Postman, I agree to the Privacy Policy and Terms.

Release Notes

Not your OS? Download for Mac (Intel Chip, Apple Chip) or Linux (x64, arm64)



POSTMAN

Join 25 million developers who use Postman

Simplify your API design, development and testing workflows with Postman.

- Save and organize your work in collections and workspaces
- Share your work with clients and collaborate with teammates
- Integrate with tools like VS Code, GitHub and BitBucket

Create a free Postman account

Work email

name@company.com

Create Free Account

Already have an account? Log In

Or continue with the lightweight API client.

Home Workspaces Explore Search Postman Sign In Create Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and **unlock all free features in Postman.**

History New Import

GET Untitled Request

Untitled Request Save

GET Enter URL or paste text Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Time to send your first request

All the requests you send will be stored in History. Sign in or create an account to organize them in collections.

Show me how

Response

Enter the URL and click Send to get a response

Console Not connected to a Postman account

Type here to search 29°C ENG 16:12 29-04-2024

Home Workspaces Explore Search Postman Sign In Create Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and **unlock all free features in Postman.**

History New Import

Today

POST http://localhost:5055/api/CustomersC
POST http://localhost:5055/api/CustomersC
POST http://localhost:5055/api/CustomersC
POST http://localhost:5055/api/CustomersC
GET http://localhost:5055/api/CustomersC

POST http://localhost:5055/ap GET http://localhost:5074/api/

http://localhost:5074/api/ProductsC Save

GET http://localhost:5074/api/ProductsC Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

	Key	Value	Bulk Edit
	Key	Value	

Response

Console Not connected to a Postman account

Type here to search 20°C ENG 00:52 30-04-2024

Home Workspaces Explore Search Postman Sign In Create Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and **unlock all free features in Postman.**

History New Import

Today
- GET http://localhost:5074/api/ProductsC

Yesterday
- POST http://localhost:5055/api/CustomersC
- POST http://localhost:5055/api/CustomersC
- POST http://localhost:5055/api/CustomersC
- POST http://localhost:5055/api/CustomersC
- GET http://localhost:5055/api/CustomersC

POST http://localhost:5055/ap | GET http://localhost:5074/api/

http://localhost:5074/api/ProductsC

Save

GET http://localhost:5074/api/ProductsC Send

Params Authorization Headers (6) Body Pre-request Script Tests Settings Cookies

Query Params

Key	Value	Bulk Edit

Body Cookies Headers (4) Test Results Status: 200 OK Time: 93 ms Size: 221 B Save Response

Pretty Raw Preview Visualize JSON


```
[
    {
        "id": 1,
        "name": "Rice",
        "amount": 120
    },
    {
        "id": 3,
        "name": "Oats",
        "amount": 200
    }
]
```


Console Not connected to a Postman account

Type here to search 20°C ENG 00:57 30-04-2024

Home Workspaces Explore Search Postman Sign In Create Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and **unlock all free features in Postman.**

History New Import

POST http://localhost:5055/ap GET http://localhost:5074/api/ POST http://localhost:5074/ap

Today

GET http://localhost:5074/api/ProductsC

Yesterday

POST http://localhost:5055/api/CustomersC

POST http://localhost:5055/api/CustomersC

POST http://localhost:5055/api/CustomersC

POST http://localhost:5055/api/CustomersC

GET http://localhost:5055/api/CustomersC

http://localhost:5074/api/ProductsC Save

POST http://localhost:5074/api/ProductsC Send

Params Authorization Headers (8) Body Pre-request Script Tests Settings Cookies

☑	Connection	keep-alive
☑	Content-Type	application/json
	Key	Value

Response

Click Send to get a response

Console Not connected to a Postman account

Type here to search 20°C ENG 00:59 30-04-2024

Home Workspaces Explore Search Postman Sign In Create Account

You are using the Lightweight API Client, sign in or create an account to work with collections, environments and **unlock all free features in Postman.**

History New Import

Today
- GET http://localhost:5074/api/ProductsC

Yesterday
- POST http://localhost:5055/api/CustomersC
- POST http://localhost:5055/api/CustomersC
- POST http://localhost:5055/api/CustomersC
- POST http://localhost:5055/api/CustomersC
- GET http://localhost:5055/api/CustomersC

POST http://localhost:5055/ap | GET http://localhost:5074/api/ | POST http://localhost:5074/ap

http://localhost:5074/api/ProductsC

Save

POST http://localhost:5074/api/ProductsC Send

Params Authorization Headers (9) Body Pre-request Script Tests Settings Cookies

none form-data x-www-form-urlencoded raw binary JSON Beautify

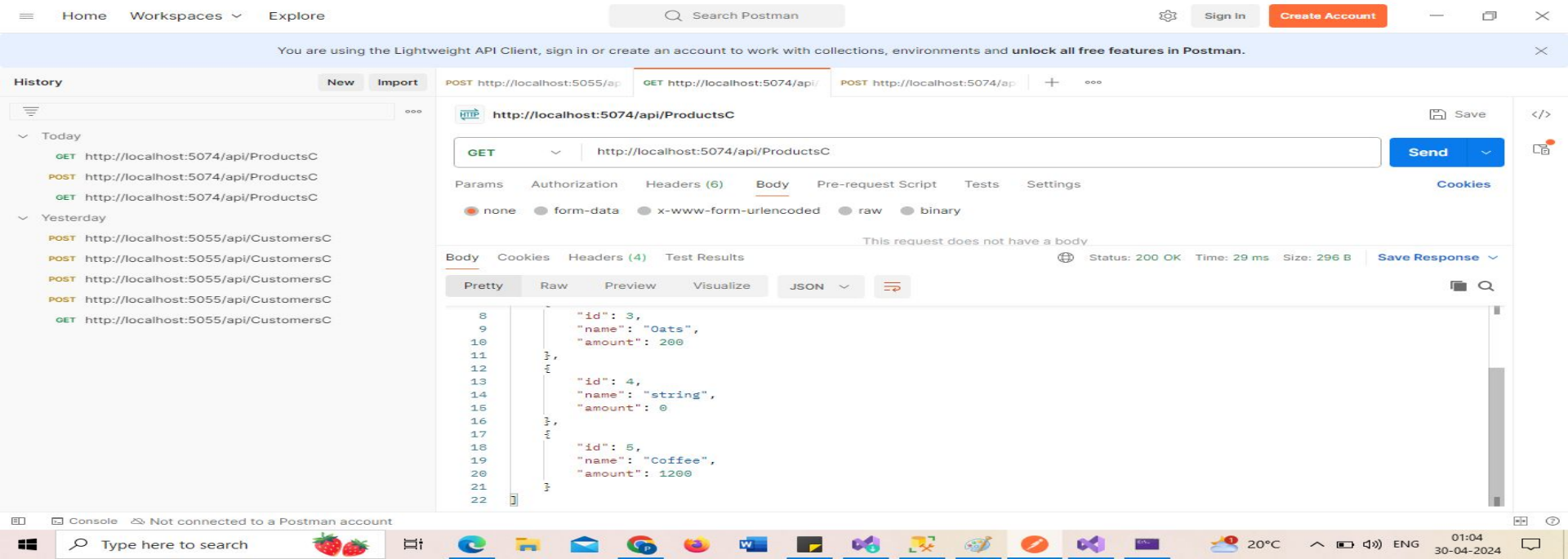
```
1 {
2
3     "name": "Coffee",
4     "amount": 1200
5 }
```

Response

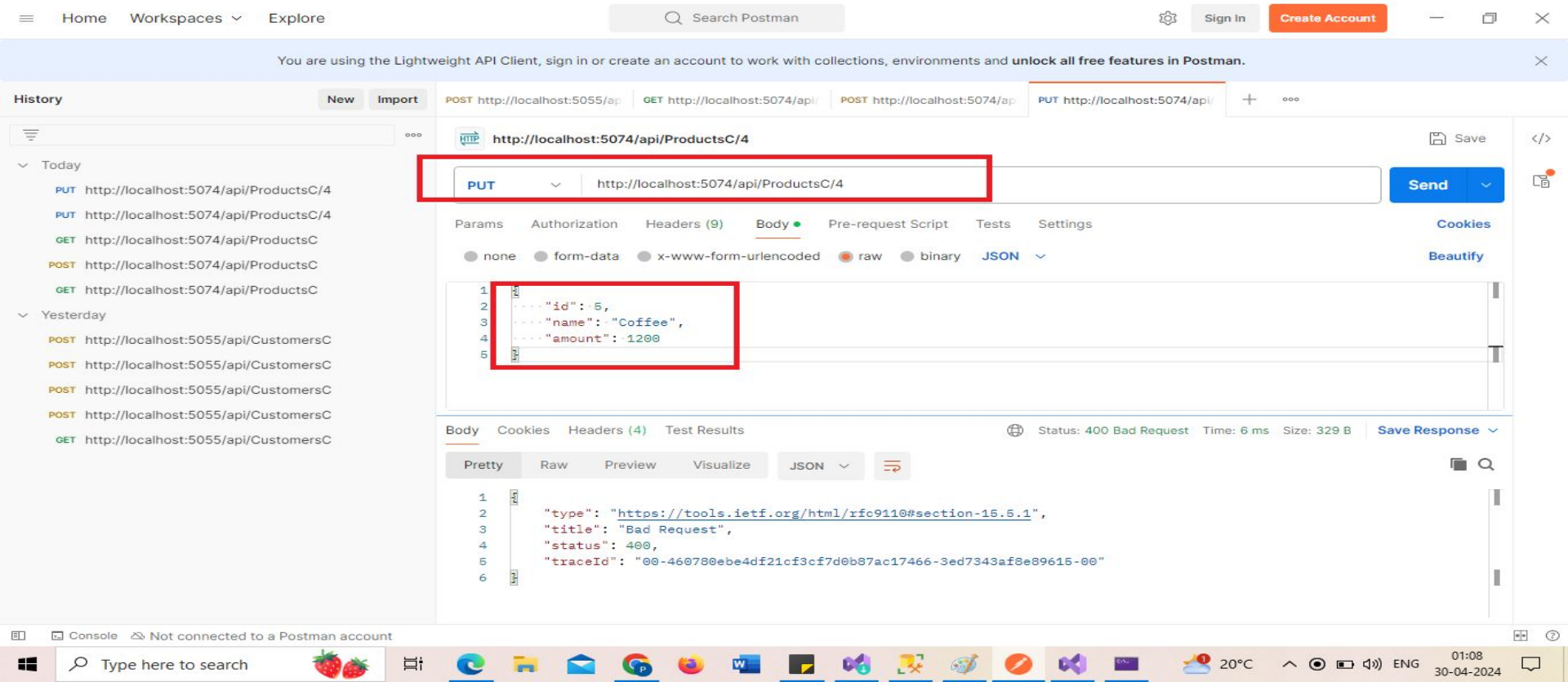
Console Not connected to a Postman account

Type here to search 20°C ENG 01:03 30-04-2024

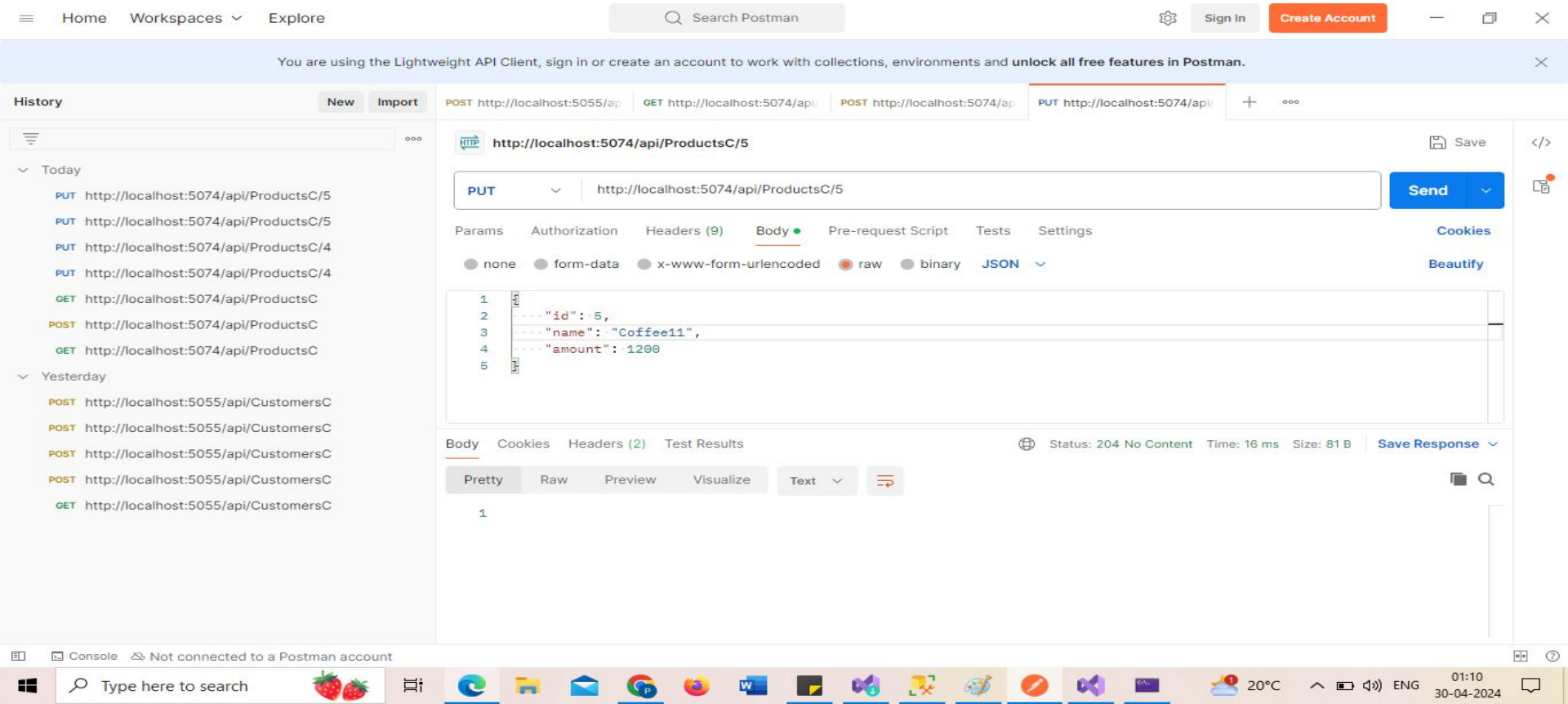
Now again go to the get request for checking whether product has added or not



Put


Home
Workspaces
Explore
Search Postman
Sign In
Create Account
You are using the Lightweight API Client, sign in or create an account to work with collections, environments and unlock all free features in Postman.
History
New
Import
Today
PUT http://localhost:5074/api/ProductsC/4
PUT http://localhost:5074/api/ProductsC/4
GET http://localhost:5074/api/ProductsC
POST http://localhost:5074/api/ProductsC
GET http://localhost:5074/api/ProductsC
Yesterday
POST http://localhost:5055/api/CustomersC
POST http://localhost:5055/api/CustomersC
POST http://localhost:5055/api/CustomersC
POST http://localhost:5055/api/CustomersC
GET http://localhost:5055/api/CustomersC
POST http://localhost:5055/ap
GET http://localhost:5074/api/
POST http://localhost:5074/ap
PUT http://localhost:5074/api/
http://localhost:5074/api/ProductsC/4
Save
PUT
http://localhost:5074/api/ProductsC/4
Send
Params
Authorization
Headers (9)
Body
Pre-request Script
Tests
Settings
Cookies
none
form-data
x-www-form-urlencoded
raw
binary
JSON
Beautify
{
"id": 5,
"name": "Coffee",
"amount": 1200
}
Body
Cookies
Headers (4)
Test Results
Status: 400 Bad Request
Time: 6 ms
Size: 329 B
Save Response
Pretty
Raw
Preview
Visualize
JSON
{
"type": "https://tools.ietf.org/html/rfc9110#section-15.5.1",
"title": "Bad Request",
"status": 400,
"traceId": "00-460780ebe4df21cf3cf7d0b87ac17466-3ed7343af8e89615-00"
}
Console
Not connected to a Postman account
Type here to search
20°C
ENG
01:08
30-04-2024

Now at id 5, I am adding coffee11, you can check using get



```
http://localhost:5074/api/ProductsC
```

Server response

Code	Details

200

Response body


```
[
  {
    "id": 1,
    "name": "Rice",
    "amount": 120
  },
  {
    "id": 3,
    "name": "Oats",
    "amount": 200
  },
  {
    "id": 4,
    "name": "string",
    "amount": 0
  },
  {
    "id": 5,
    "name": "Coffee11",
    "amount": 1200
  },
  {
    "id": 6,
    "name": "string",
    "amount": 0
  }
]
```


Download

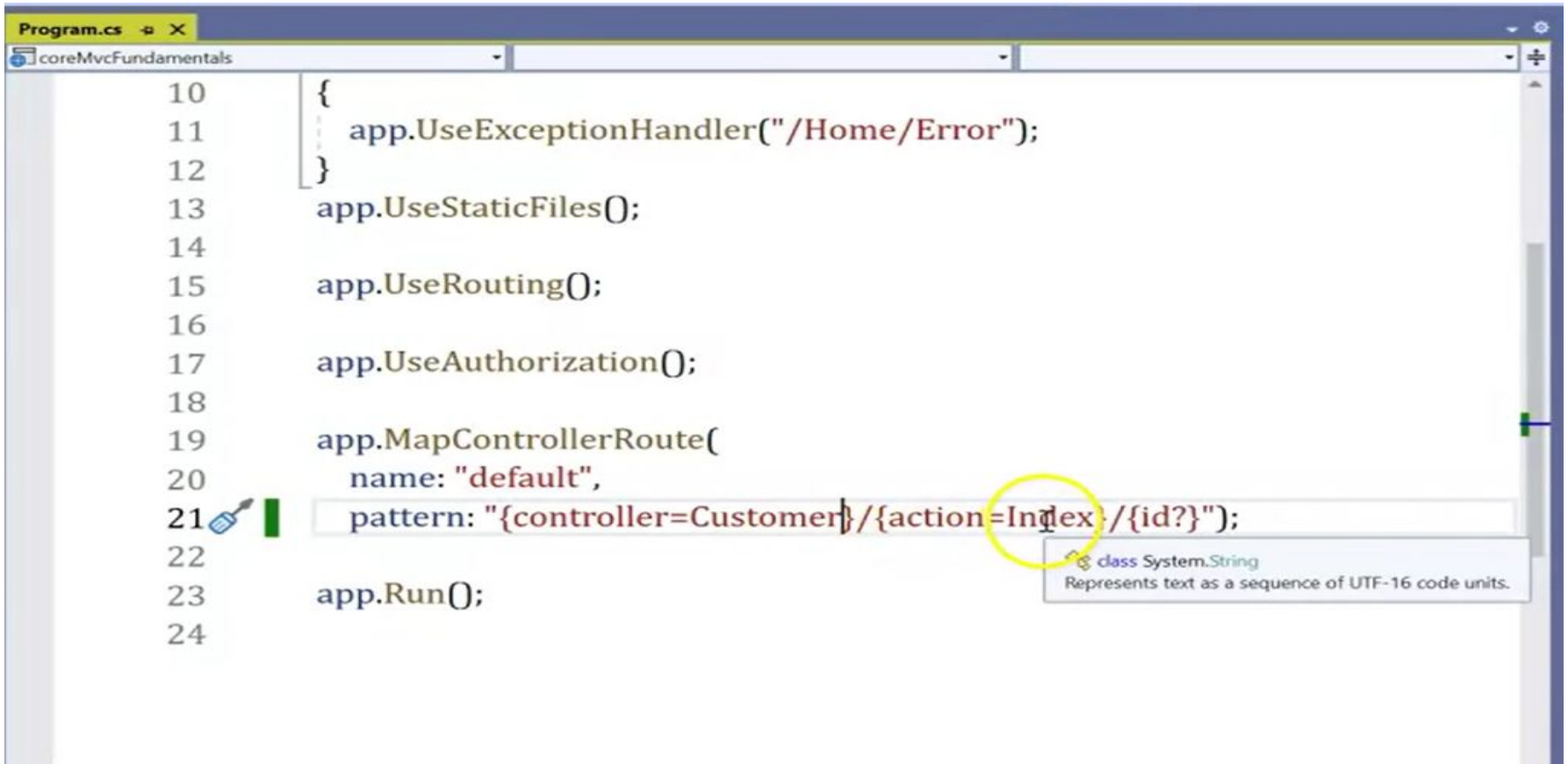
Response headers

```
content-type: application/json; charset=utf-8
```

Conventional Routing

CONVENTION ROUTING

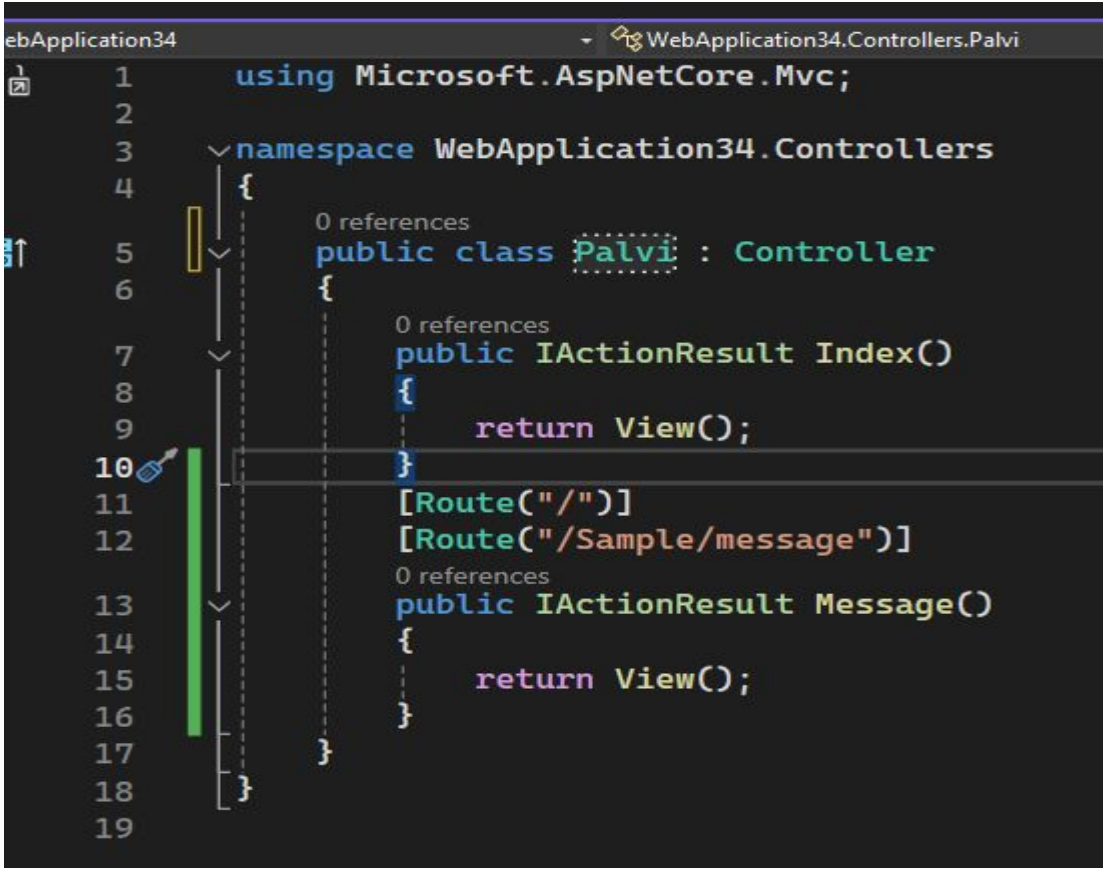
- It is used for defining routes based on conventions and defaults,
- rather than explicitly defining each route.
- Developers define a set of conventions that are used to generate routes for different types of controllers and actions.
- It provides a simple and flexible way to define routes in an ASP.NET Core application.
- The URL pattern is generated based on the names of the controller and action methods.

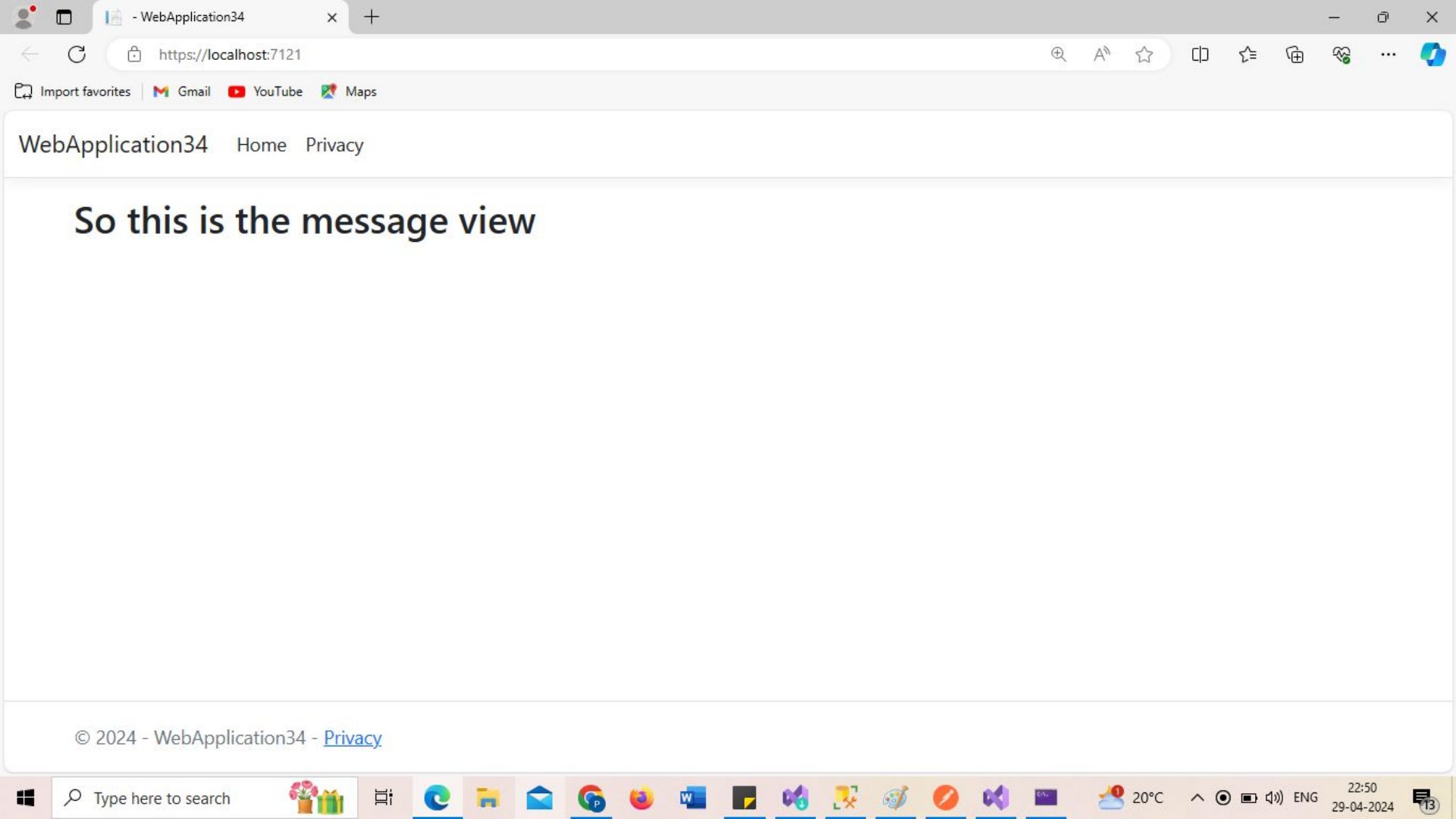

Program.cs
coreMvcFundamentals
{
app.UseExceptionHandler("/Home/Error");
}
app.UseStaticFiles();
app.UseRouting();
app.UseAuthorization();
app.MapControllerRoute(
name: "default",
pattern: "{controller=Customer}/{action=Index}/{id?}");
app.Run();
class System.String
Represents text as a sequence of UTF-16 code units.

Attribute Routing

ATTRIBUTE ROUTING

- It is a technique used for defining routes directly in the controller code using attributes.
- Developers can specify the URL pattern for a particular action method using attributes,
- instead of relying on conventions or explicit route definitions.
- In attribute routing, [Route] attribute is used to specify the URL pattern for an action method.
- By specifying the route directly in the controller code, developers can have fine-grained control over the URL patterns for each action method.


ebApplication34
WebApplication34.Controllers.Palvi
using Microsoft.AspNetCore.Mvc;
namespace WebApplication34.Controllers
{
0 references
public class Palvi : Controller
{
0 references
public IActionResult Index()
{
return View();
}
[Route("/")]
[Route("/Sample/message")]
0 references
public IActionResult Message()
{
return View();
}
}
}


- WebApplication34
https://localhost:7121
Import favorites
Gmail
YouTube
Maps
WebApplication34
Home
Privacy
So this is the message view
© 2024 - WebApplication34 - Privacy
Type here to search
20°C
ENG
22:50
29-04-2024

If I want that by default message method will open, then use [“/Route”], and if I want some specific URL will open that path, then add [Route(“Jai/Message”)]