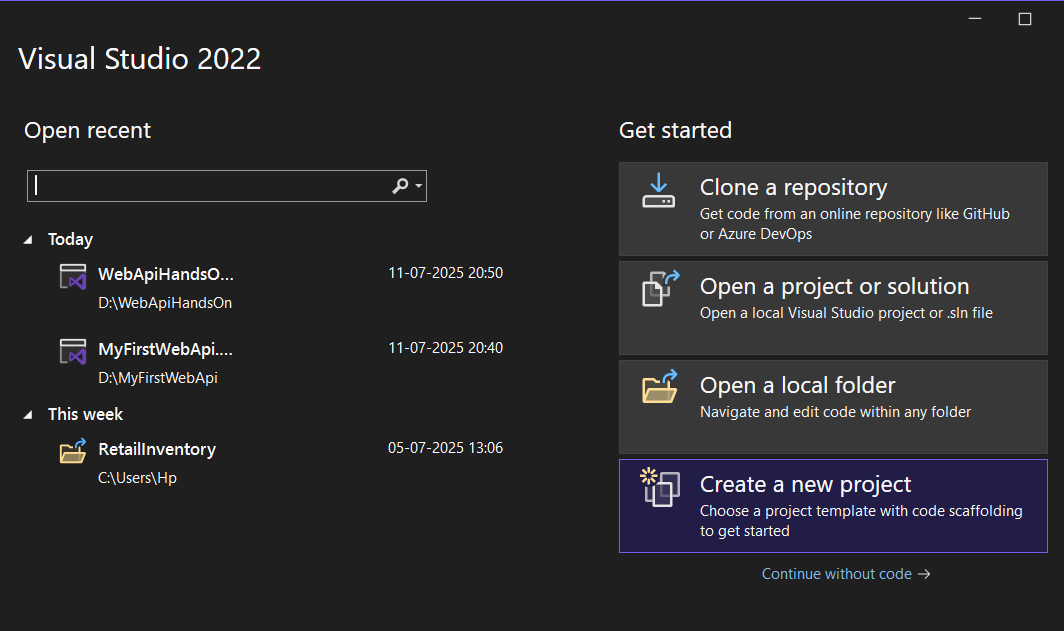
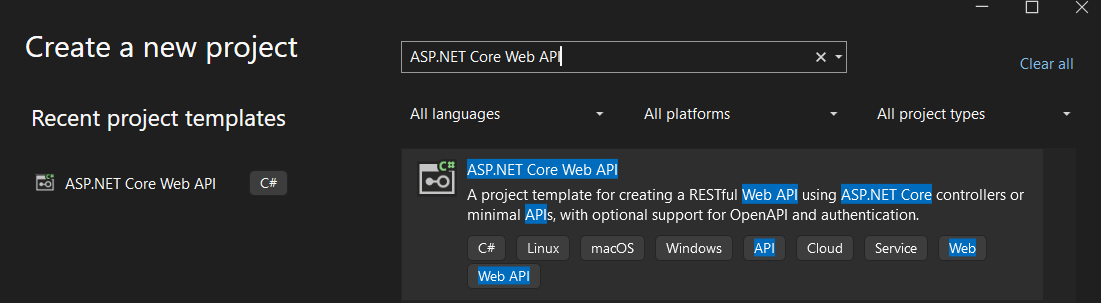
**1.WebApi - Hands\_on**

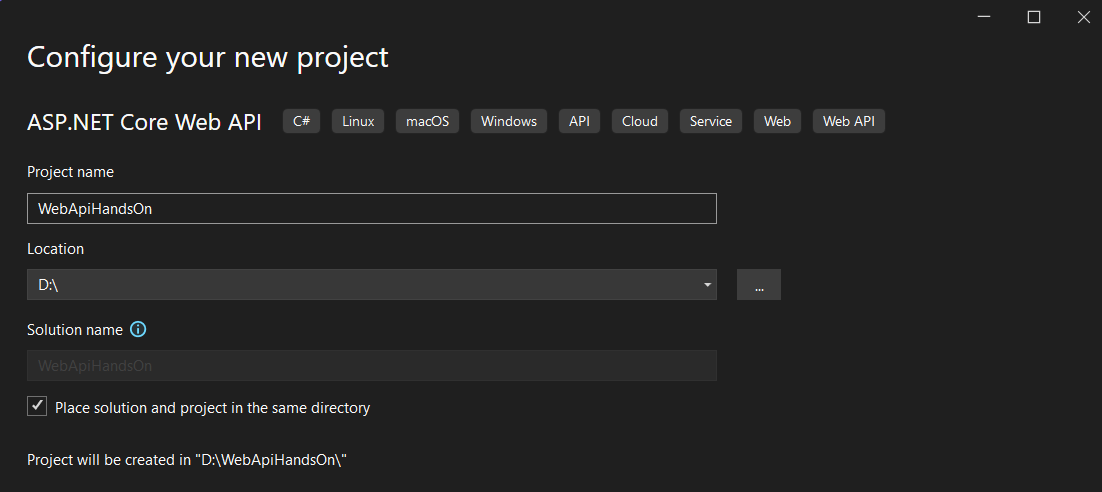
**First Web Api using .Net core:**

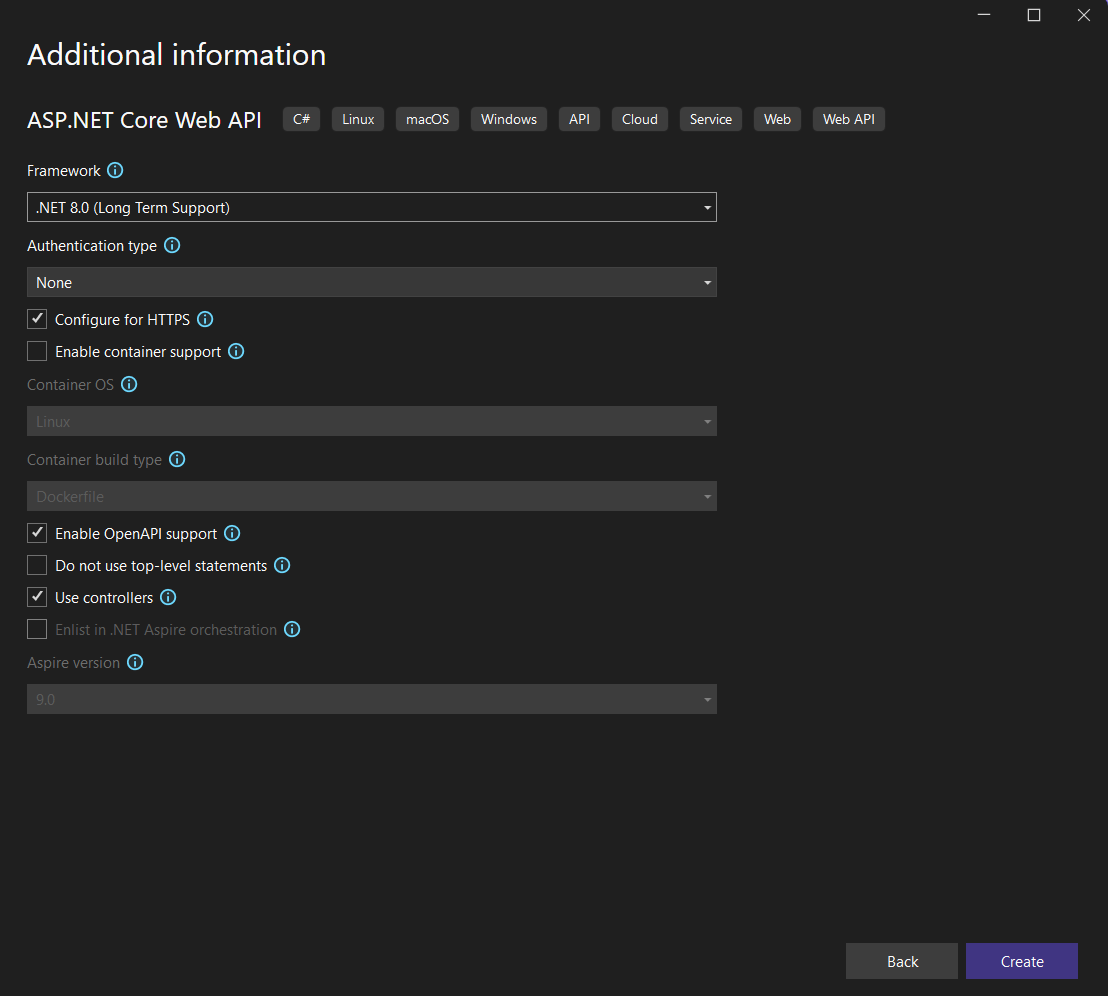
Create a .Net core web application with API template. Use the option to create controller with Read Write permissions. Notice the ValuesController creation with Action methods corresponding to the Action verbs.On creation of the Web API, execute the application and check if the GET action method result is returned as expected.

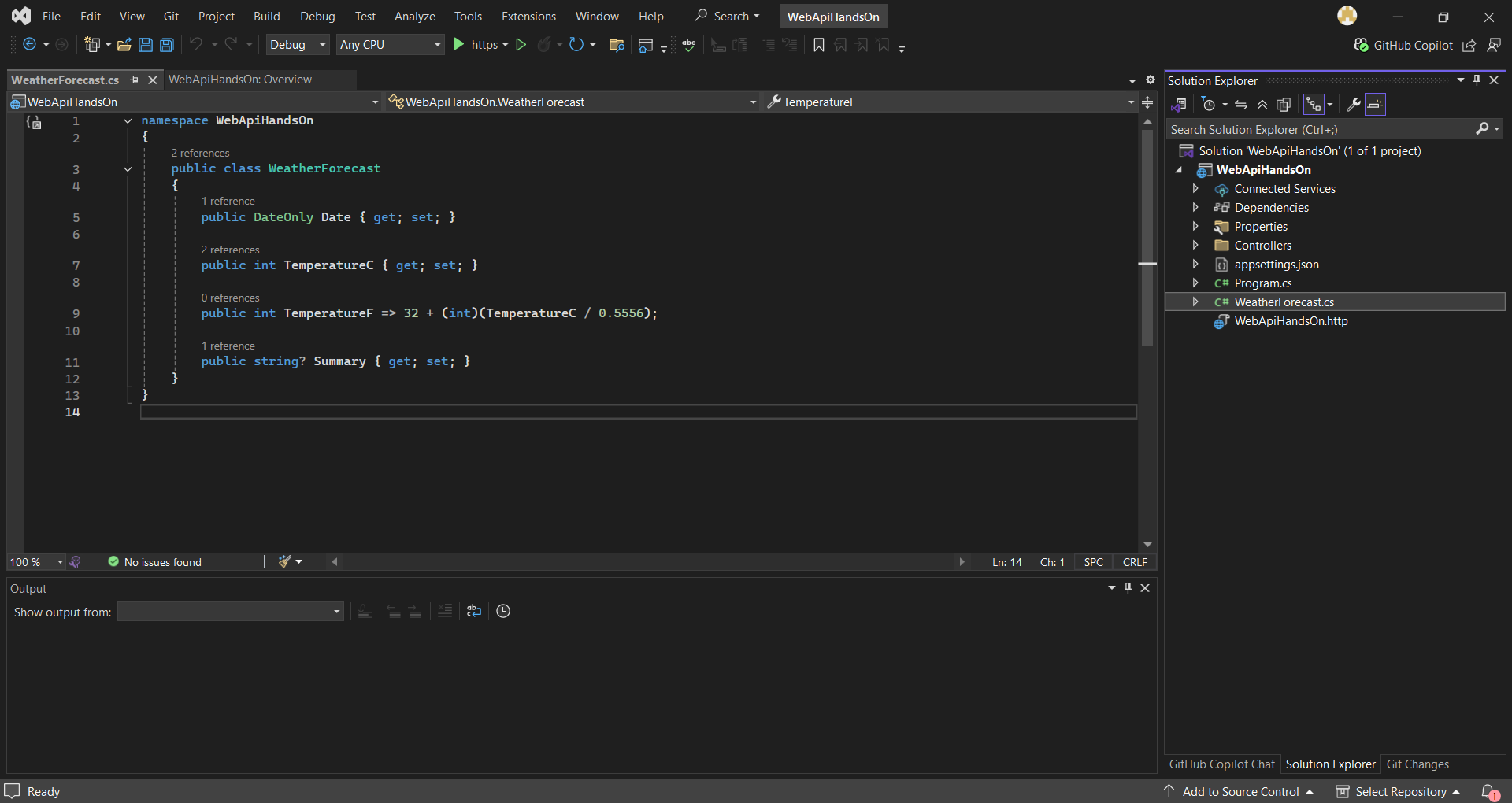
**Steps:**

****

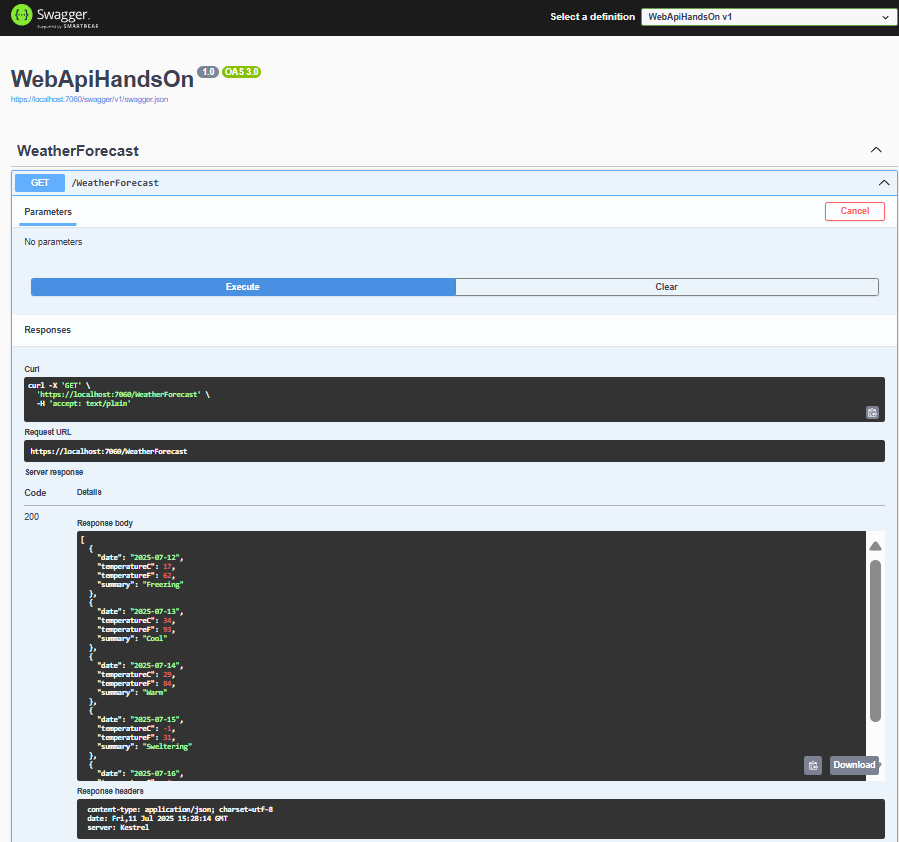
****

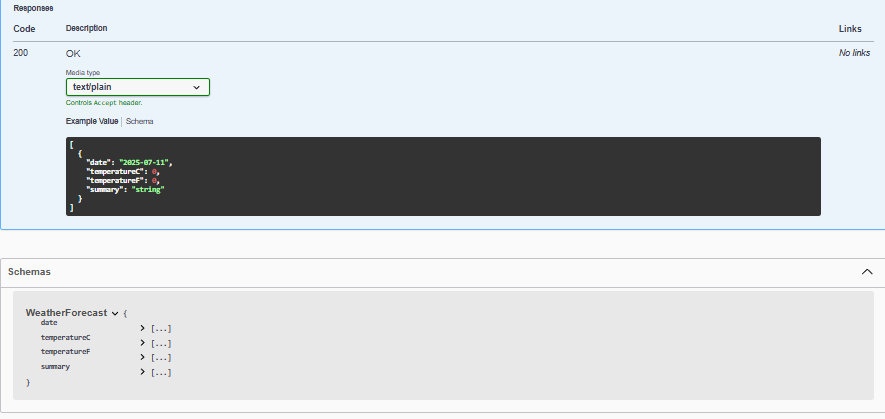
****

****

****

**OUTPUT:**

****

****

**2.WebApi - Hands\_On**

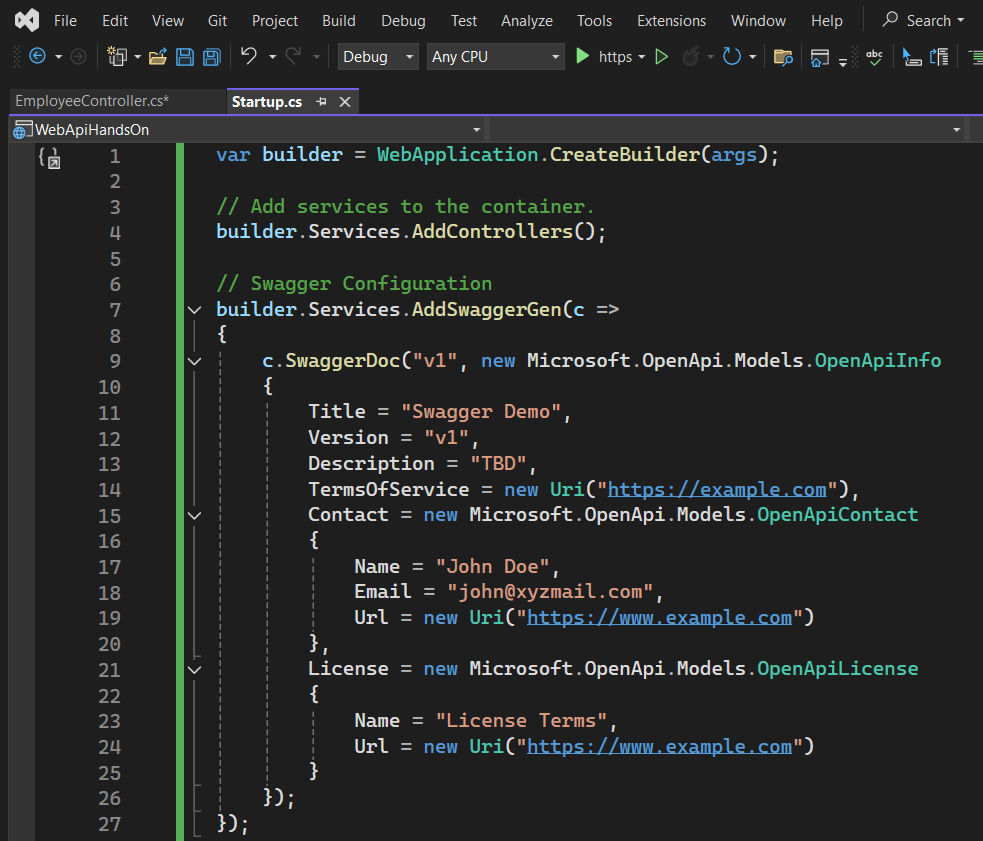
**TASK 1 . Web Api using .Net core with Swagger**

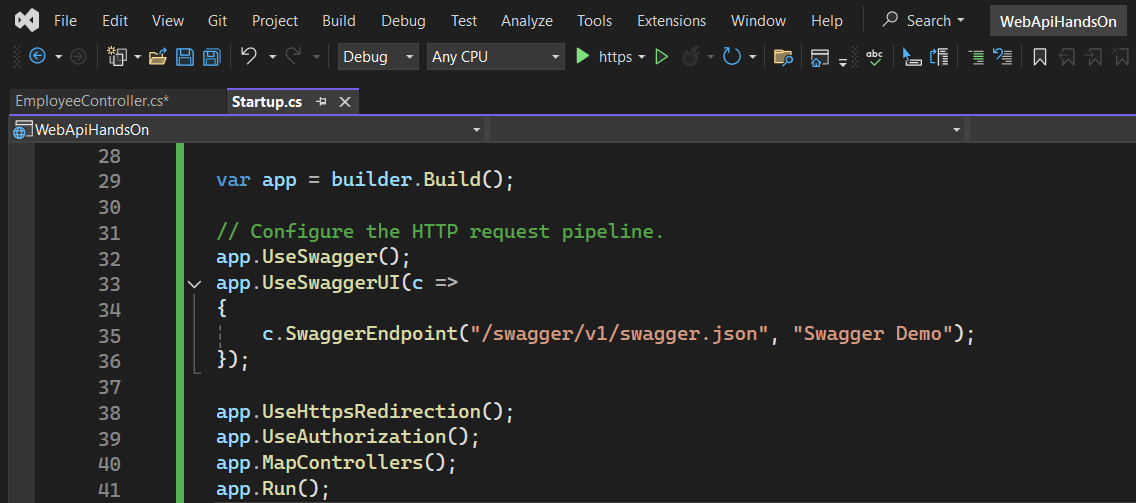
* Create a .Net core web application with API template. (Use existing application if created). Install Swashbuckle.AspNetCore Nuget package. Post this do the following steps in Startup.cs
* Execute the application which will load the default ‘Values’ controller(Settings as per launchSettings.json) GET action method. Change the url to <https://localhost:[port> number]/swagger
* Notice the Title, Version, Contact detail provided shown on the top of the page
* Notice the Values controller HttpVerb action methods getting listed.
* Click the ‘GET’ action verb method(Without the parameter).
* It opens a panel which has ‘Try it out’ button. Click that and Click ‘Execute’ button.

**STEPS:**

Step 1 : Update Startup.cs

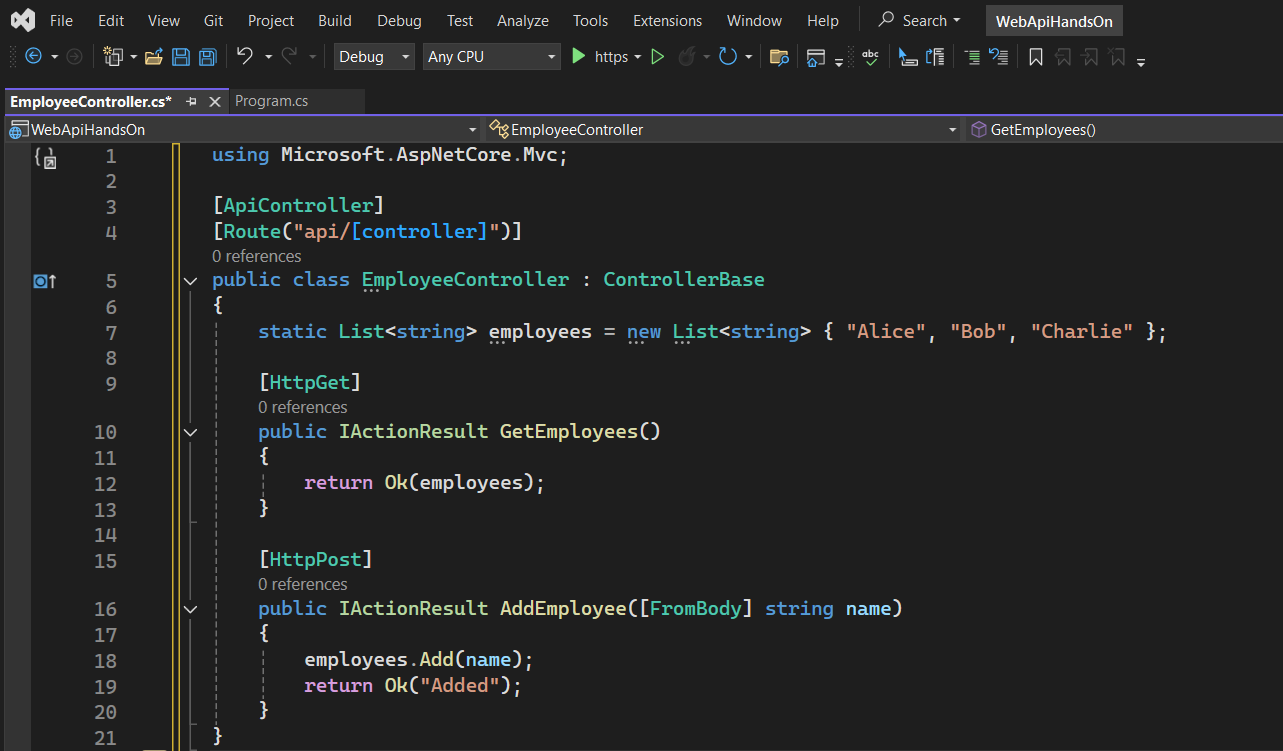
**Startup.cs**

****

****

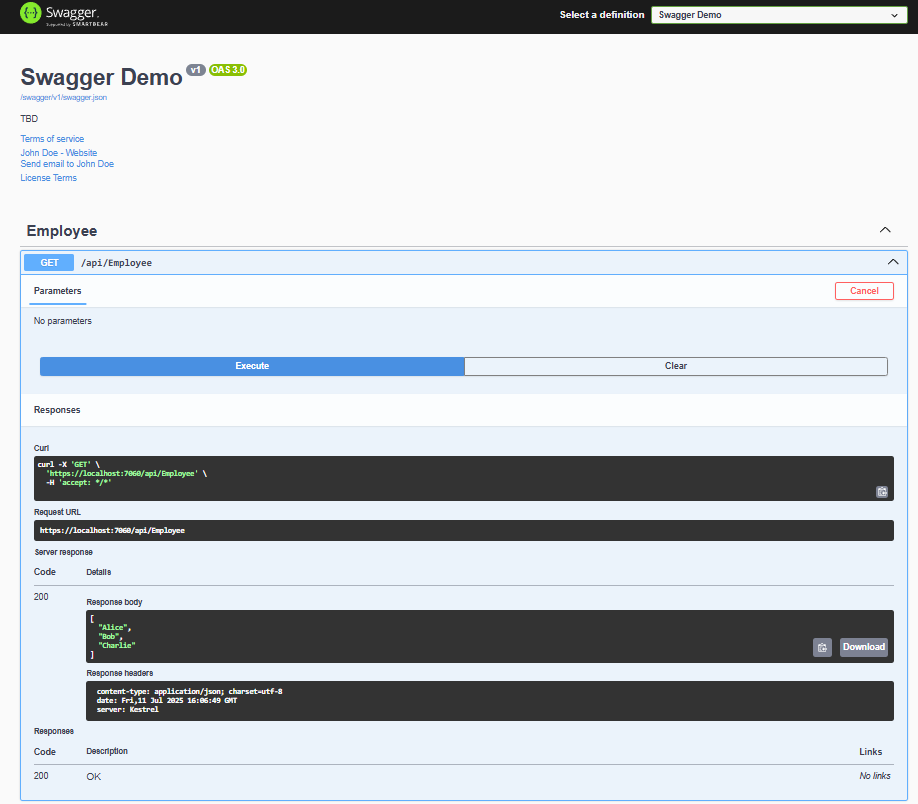
Step 2: Add EmployeeController.cs

**EmployeeController.cs**

****

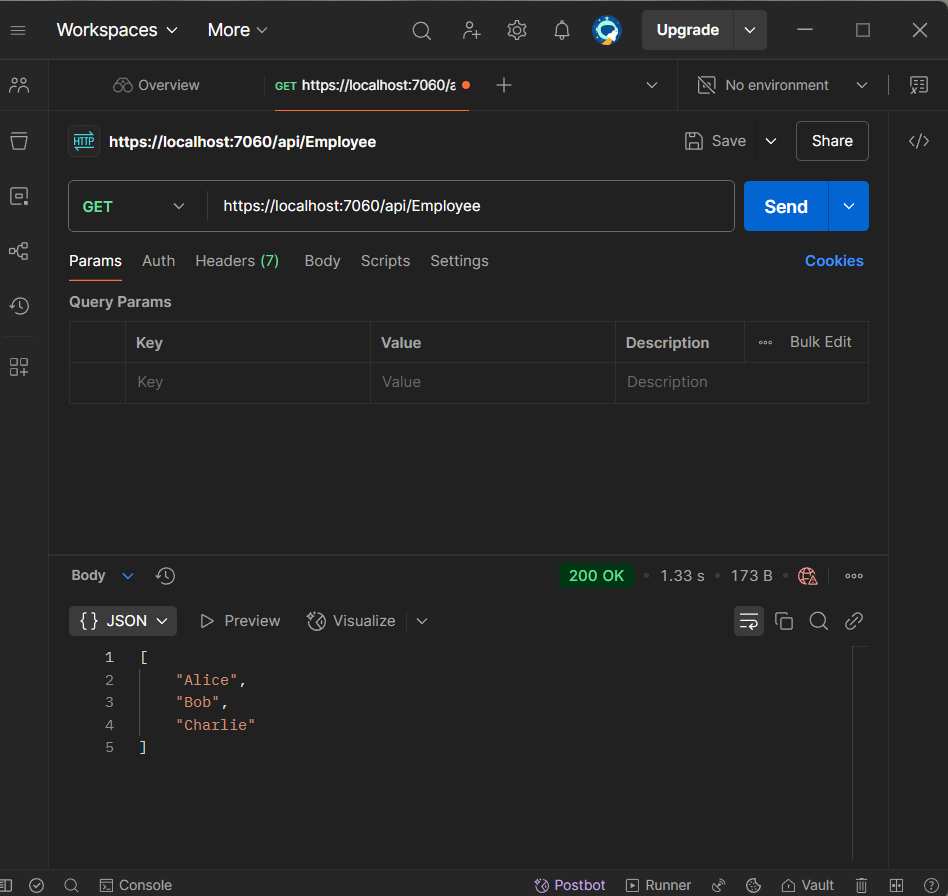
Step 3: Run and Test in Swagger

**OUTPUT:**

****

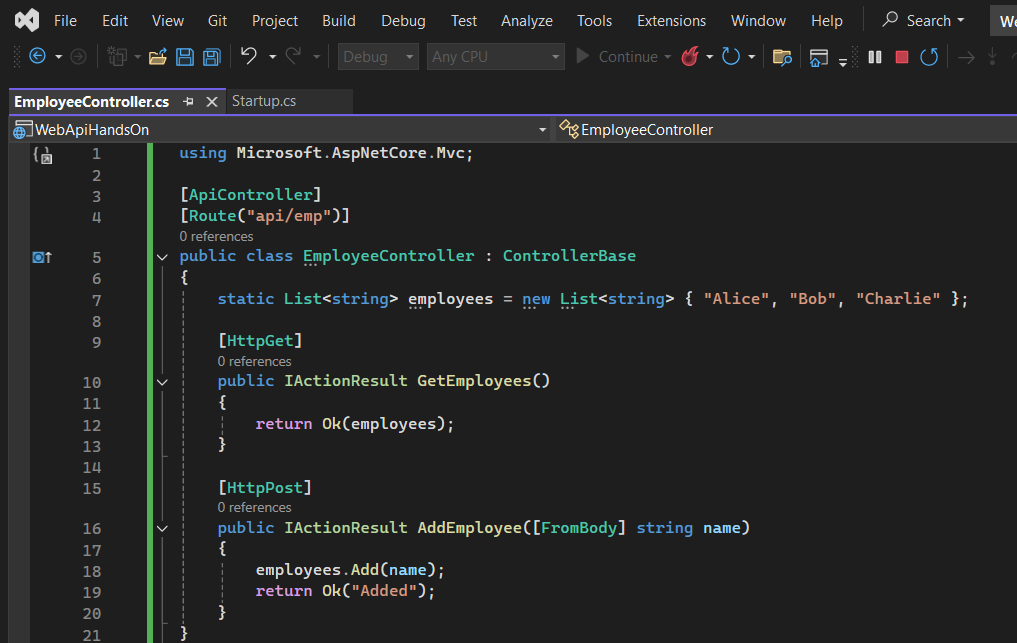
**TASK 2:** Use POSTMAN tool, to point to the local Web API that was created with Employee controller. Test the GET action method using POSTMAN.Verify the output if the List of employees are listed in the ‘Body’ part of the GET window on POSTMAN tool.Verify the Status on the right side of the output pane on POSTMAN tool.

**OUTPUT:**

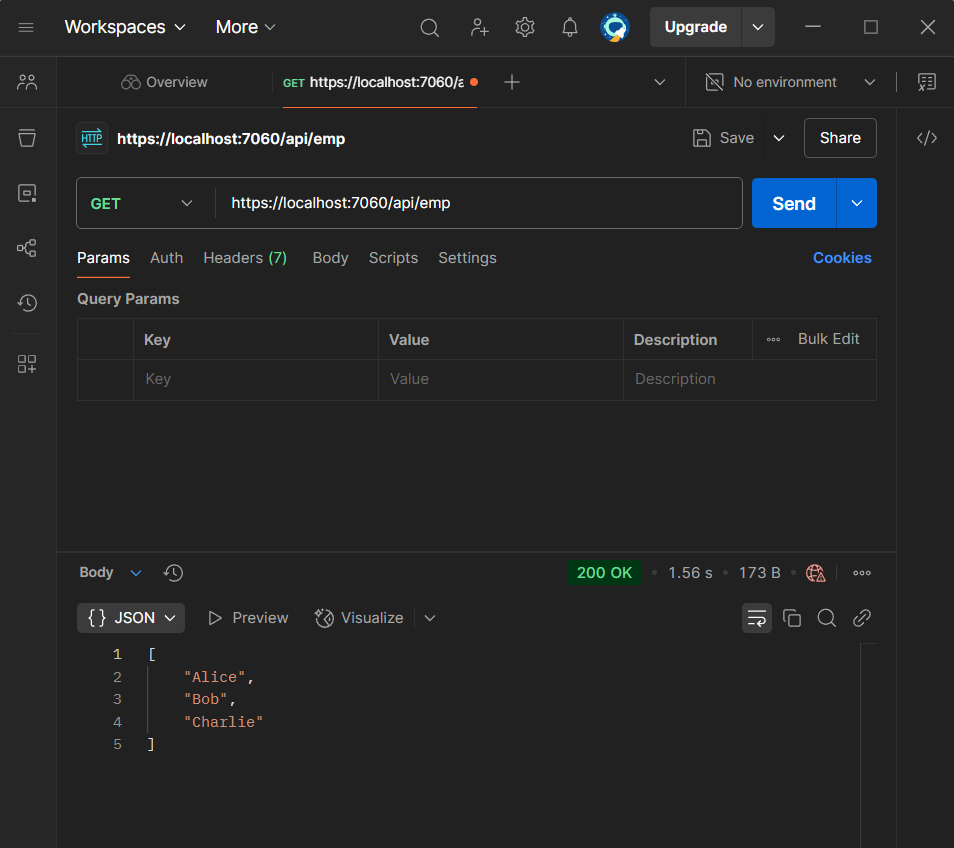


**TASK 3:** Modify the Controller name in the Route attribute of the Employee controller to ‘Emp’ and check its access thru POSTMAN

**EmployeeController.cs**

****

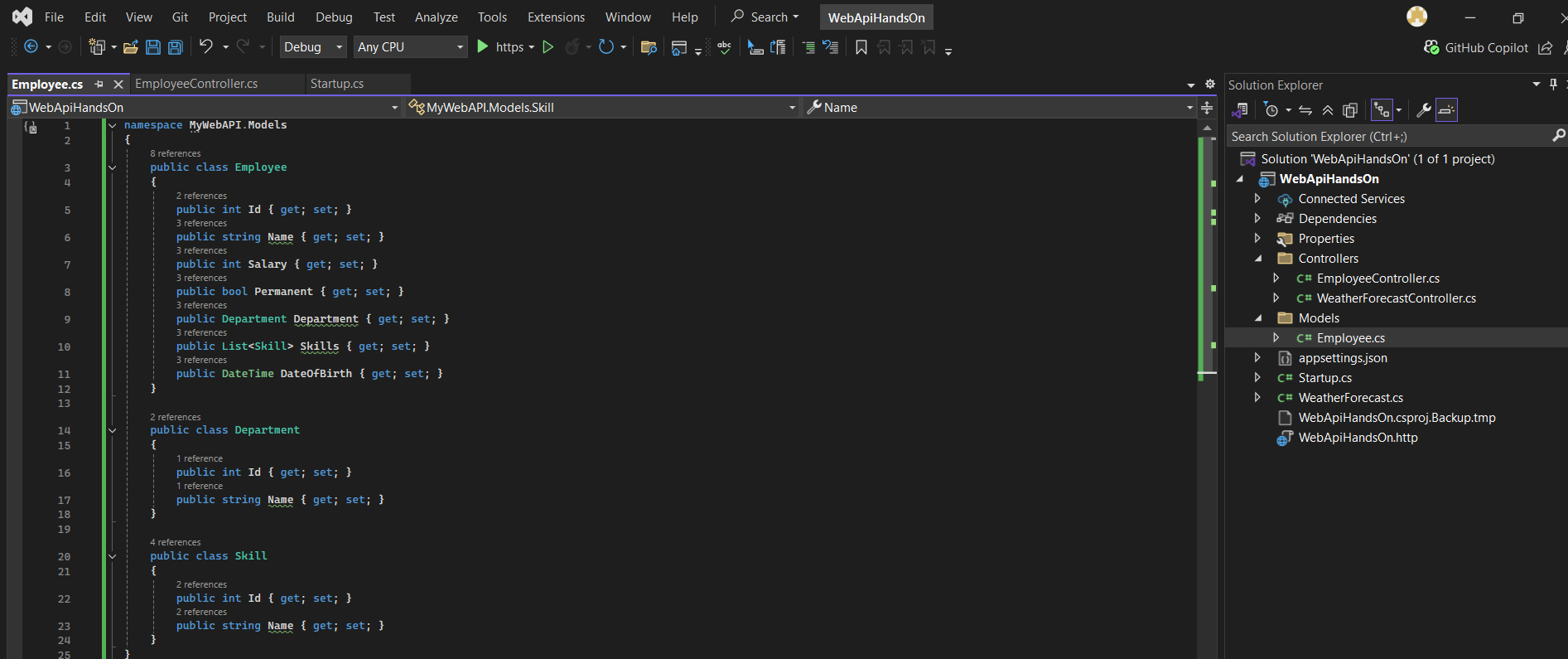
**OUTPUT:**

****

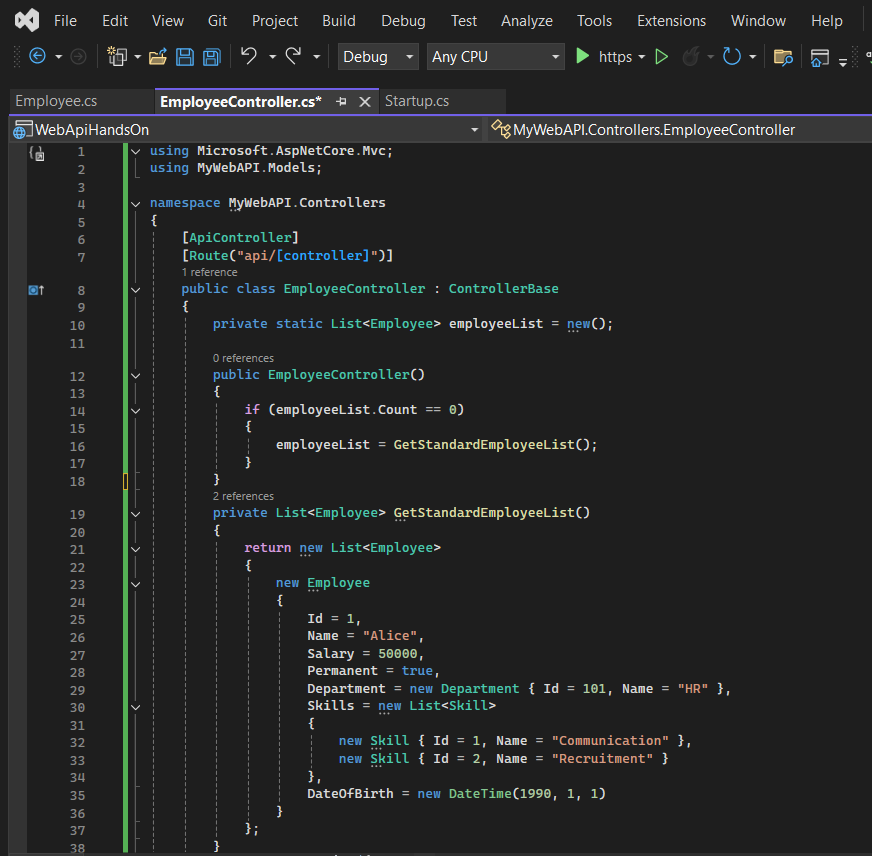
**3. WebApi - Hands\_On**

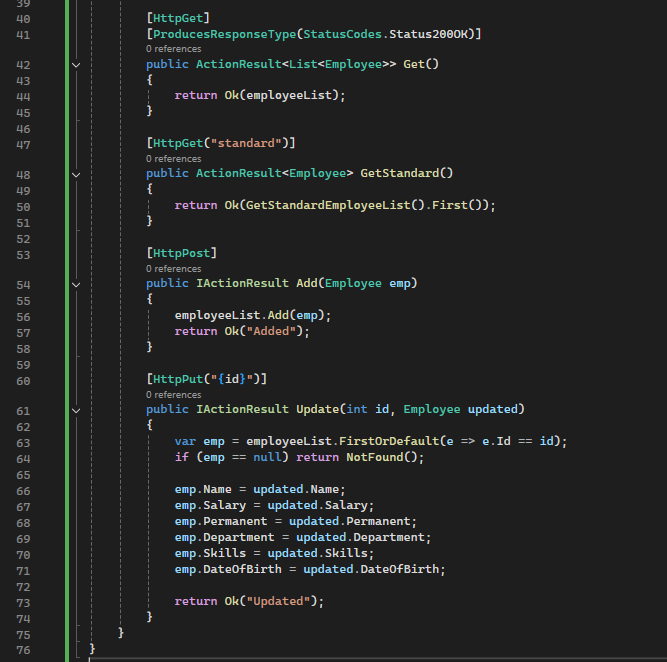
**TASK 1: Web Api using custom model class**

1. Step 1: Create Employee.cs Model

****

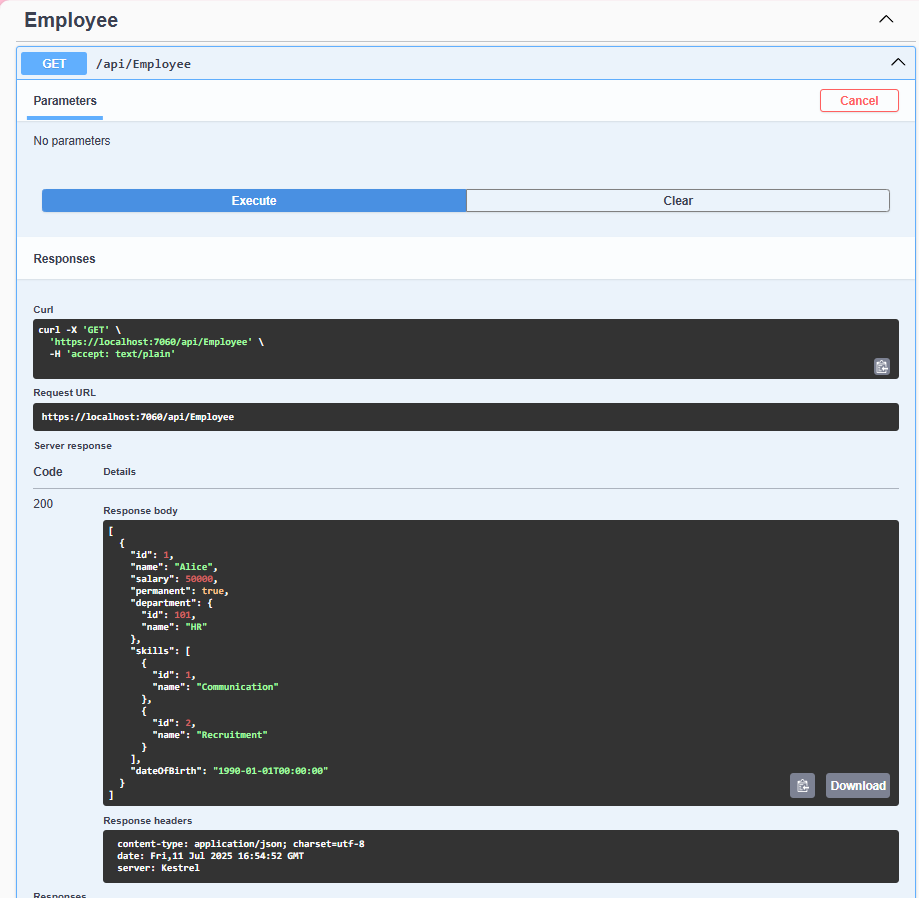
Step 2: Create EmployeeController.cs

****

****

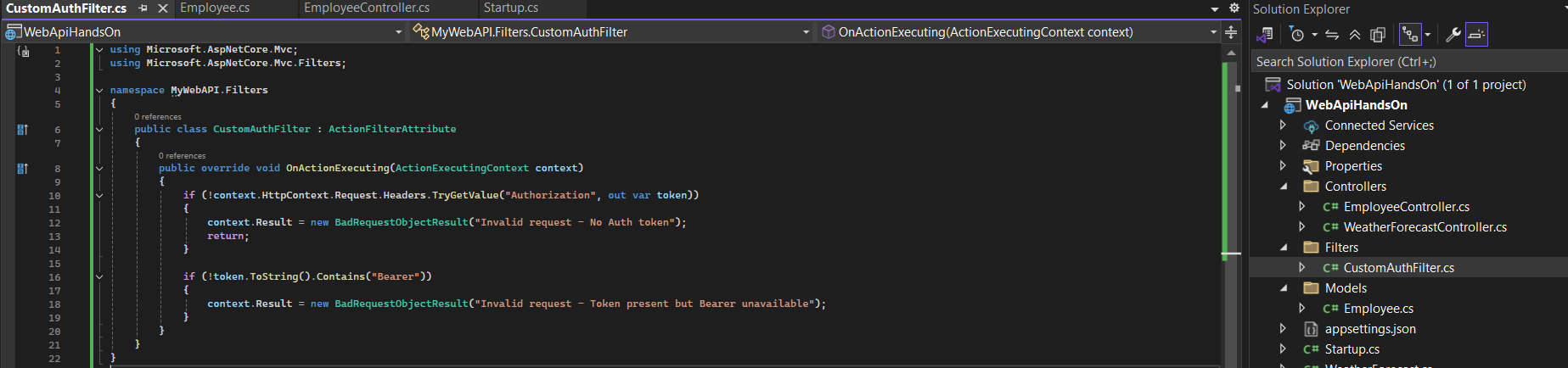
**Step 3:** Run & Test with Swagger

**OUTPUT:**

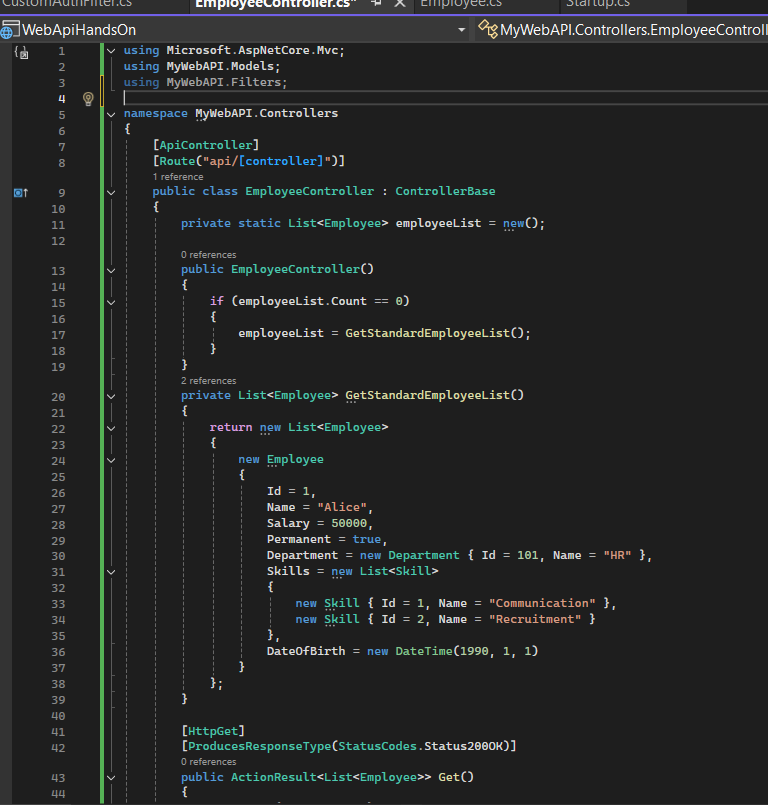
****

**TASK 2: Create a Custom action filter for Authorization.**

Step 1: Add Folder and Class

****

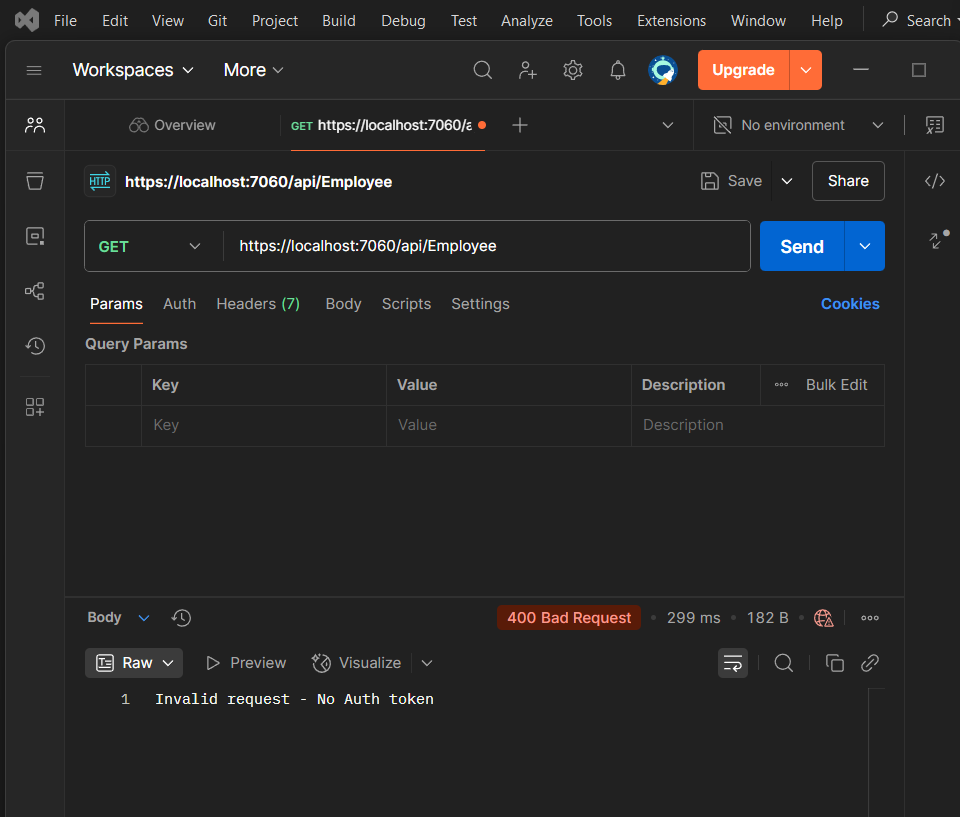
Step 2: Apply Filter to Controller

****

Step 3: Test in Postman

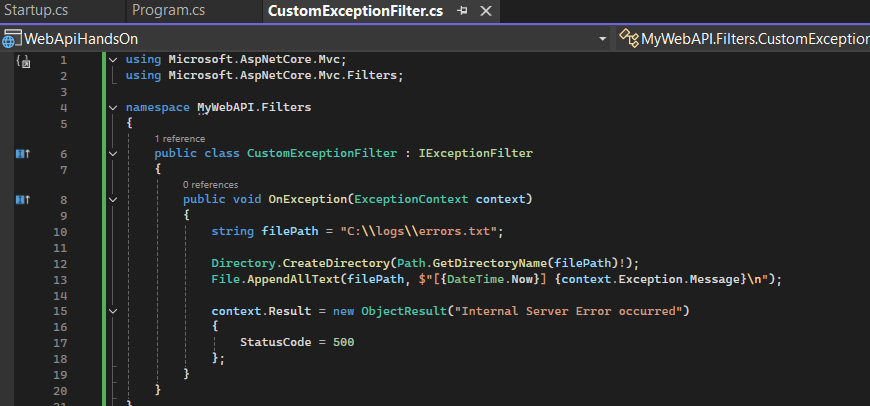
**OUTPUT:**

Without header

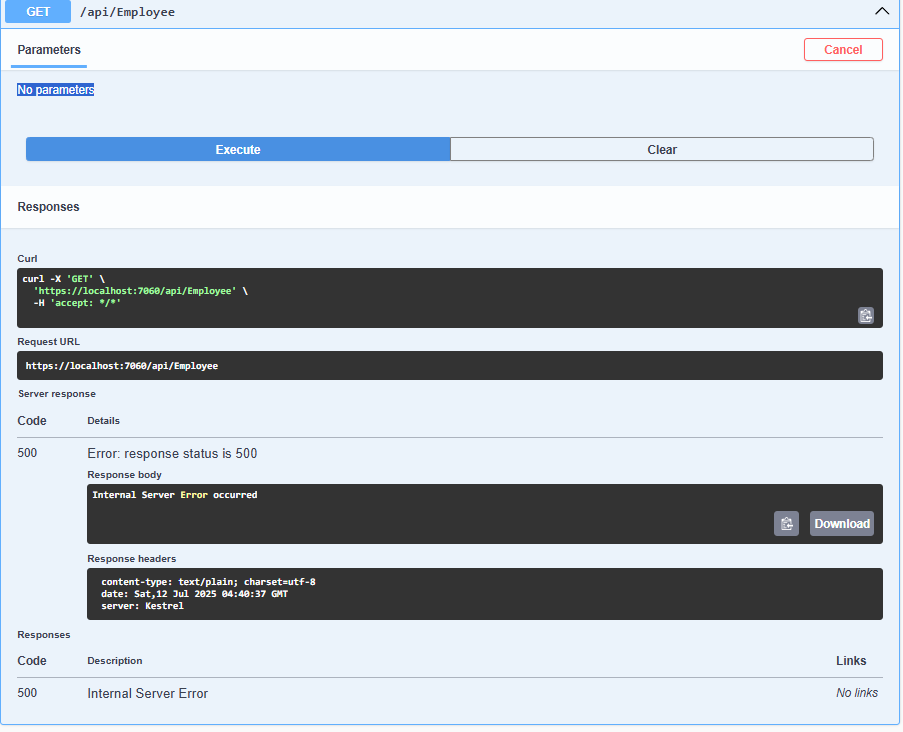
****

**TASK 3: Custom Exception filter**

CustomExceptionFilter.cs

****

**OUTPUT:**

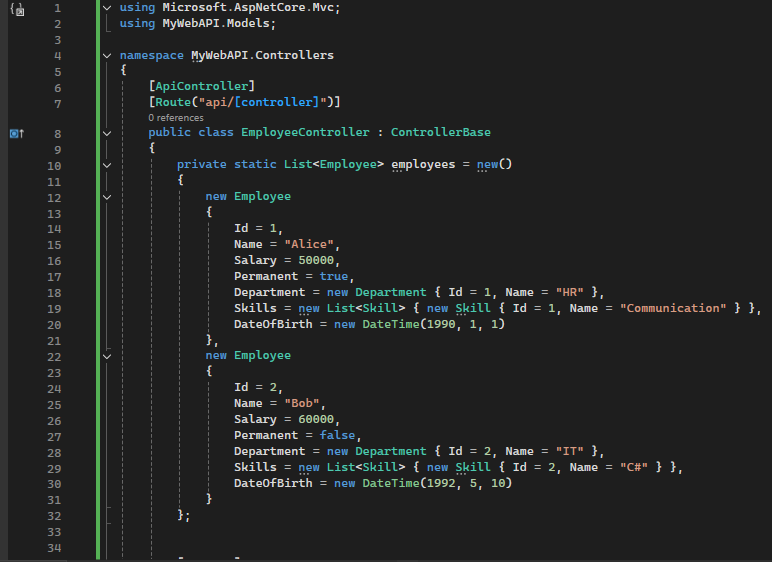
****

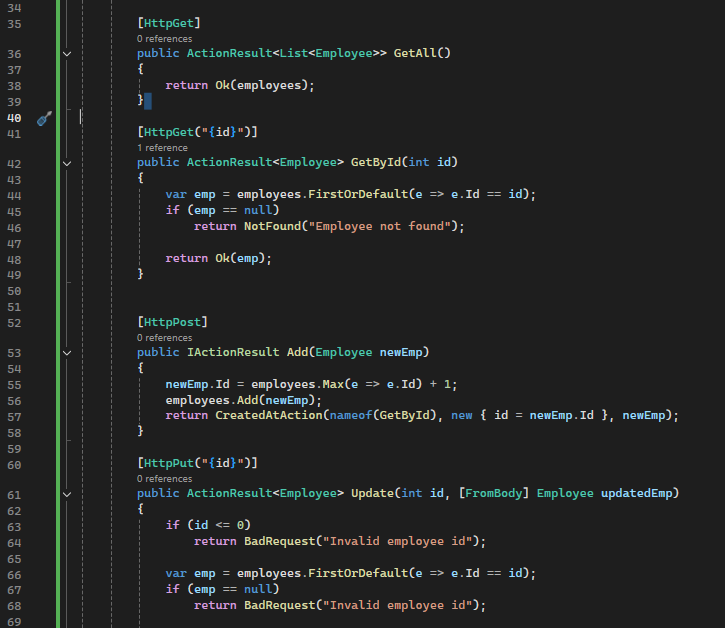
**4.WebApi - Hands\_on**

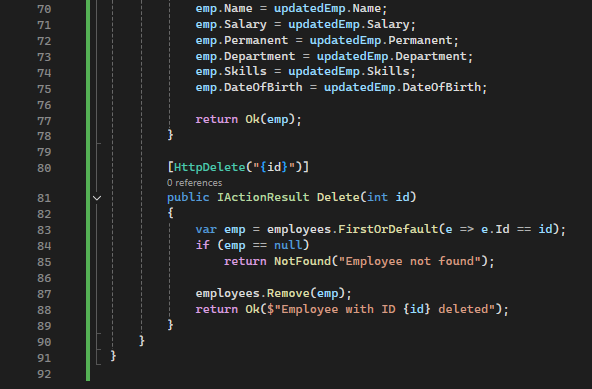
**TASK 1: Web Api CRUD operation**

Update Employee data as per the input thru Web API PUT action method call

**EmployeeController.cs**

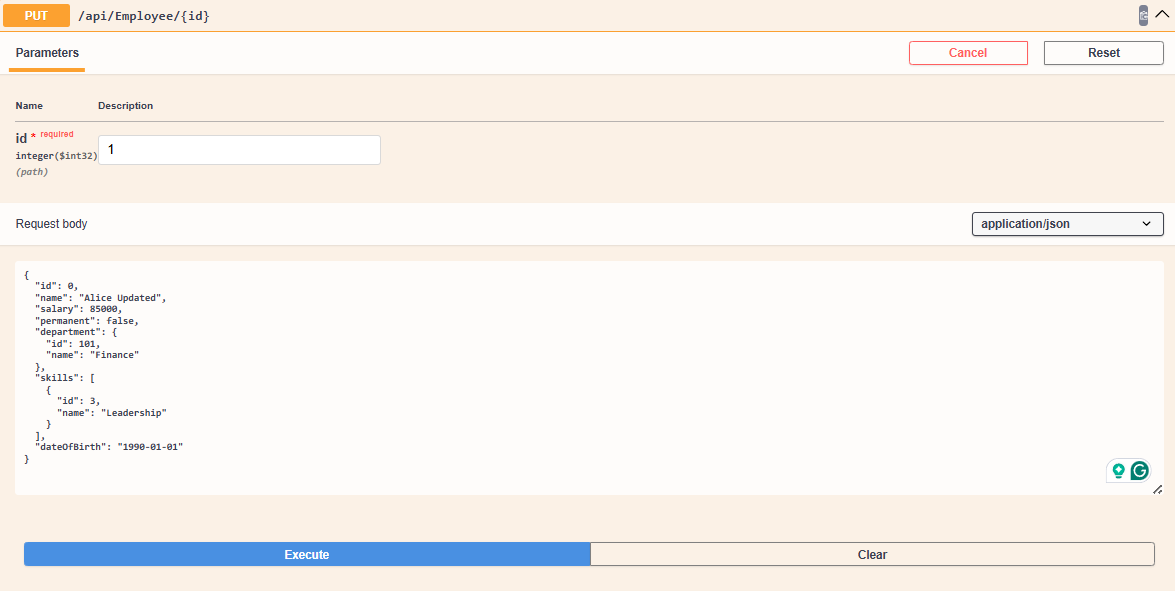
****

****

****

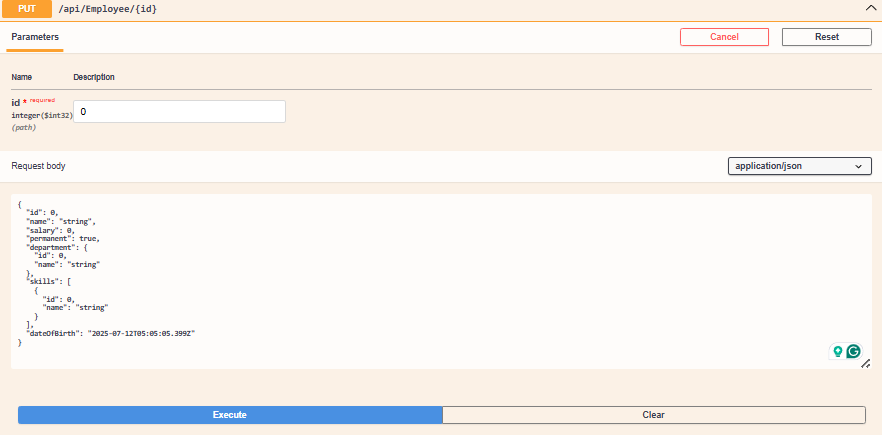
**OUTPUT:**

**1). Output for valid id (id=1)**

****

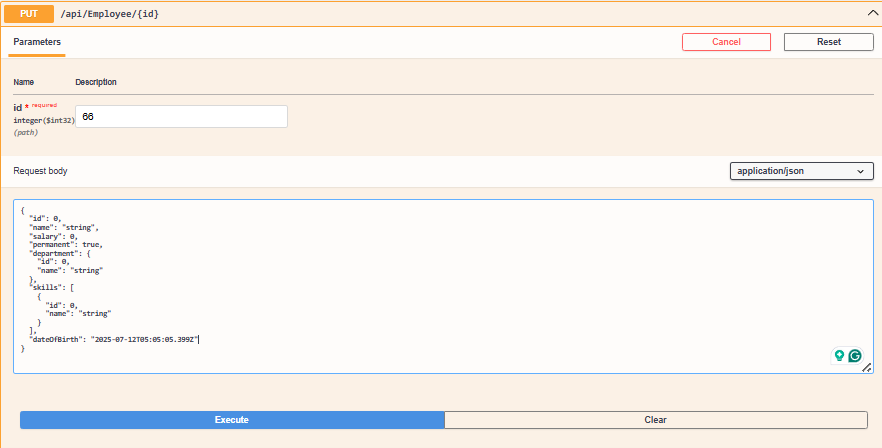
****

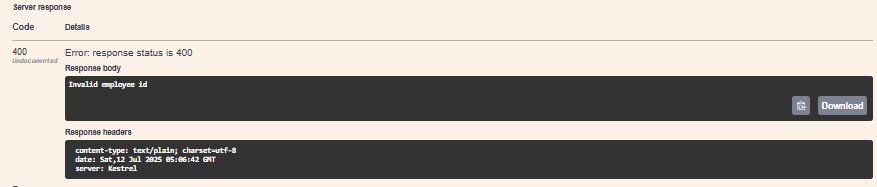
**2). OUTPUT FOR INVALID ID <= 0**

****



3). **OUTPUT FOR ID not found in list → throw BadRequest**

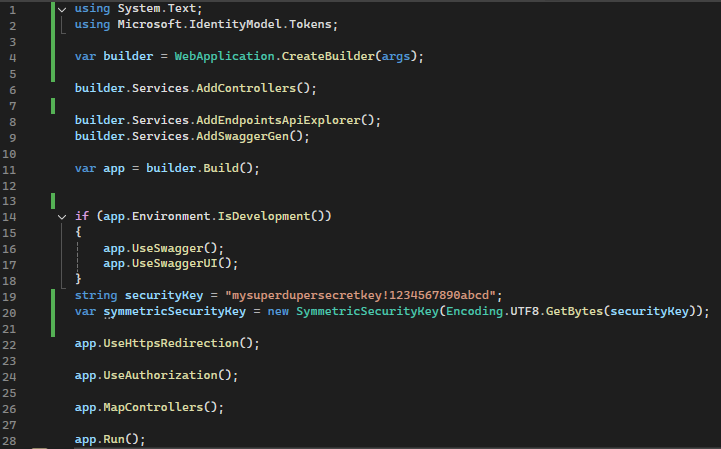




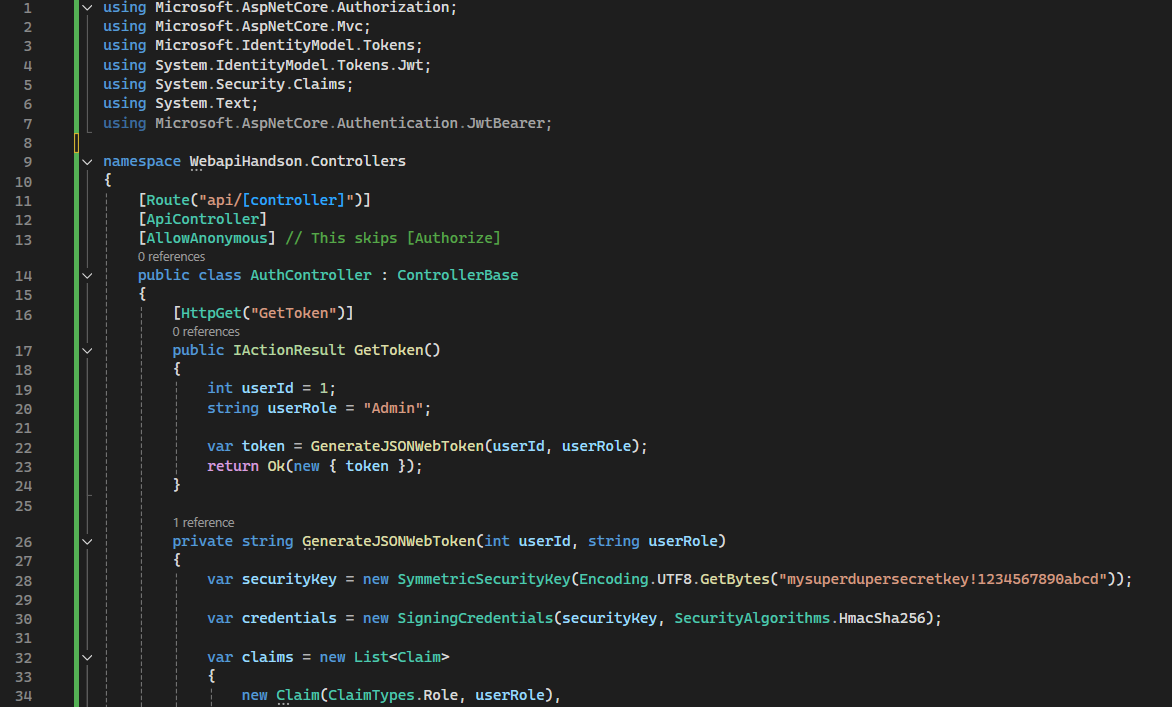
**5.WebApi - Hands\_on**

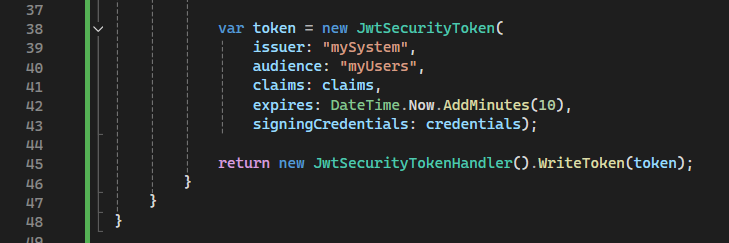
**TASK 1:** **JsonWebToken**

**Program.cs**

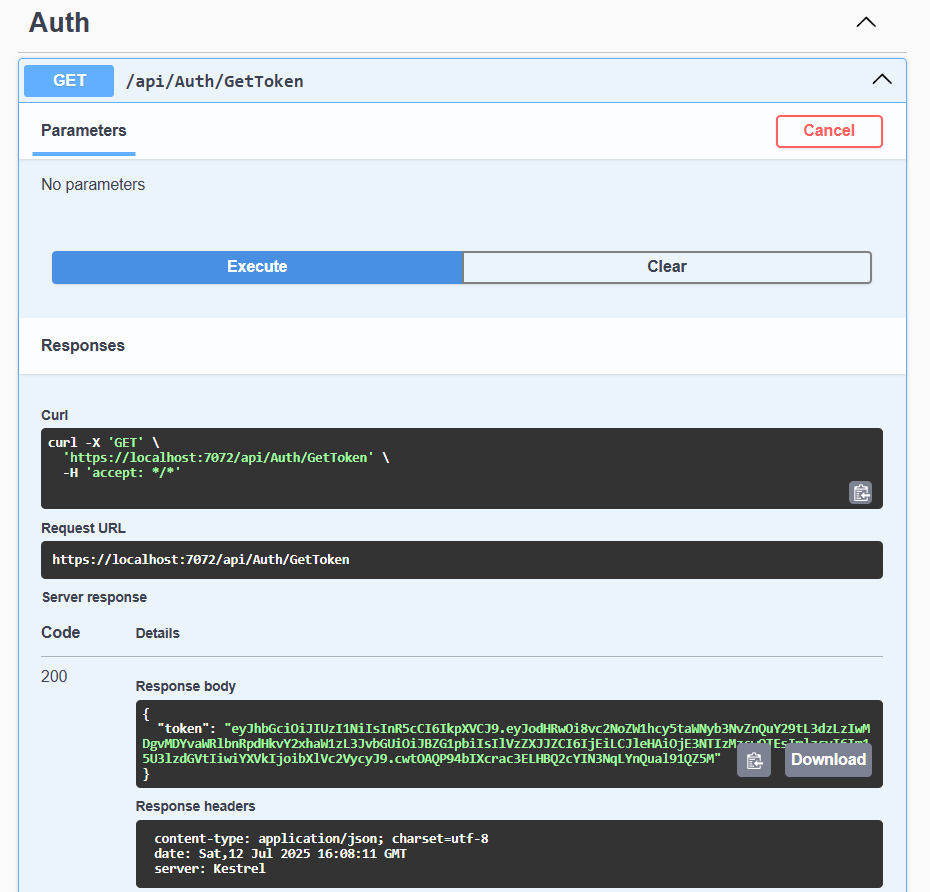
****

**AuthController.cs**

****



**OUTPUT:**

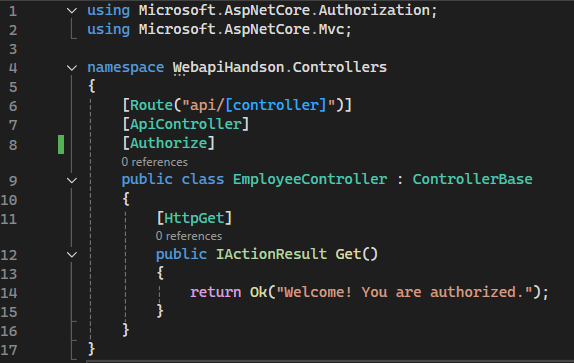


**TASK 2: Use the JWT generated thru the AuthController to be used in POSTMAN request.**

STEPS:

STEP 1:.Add [Authorize] to Your EmployeeController

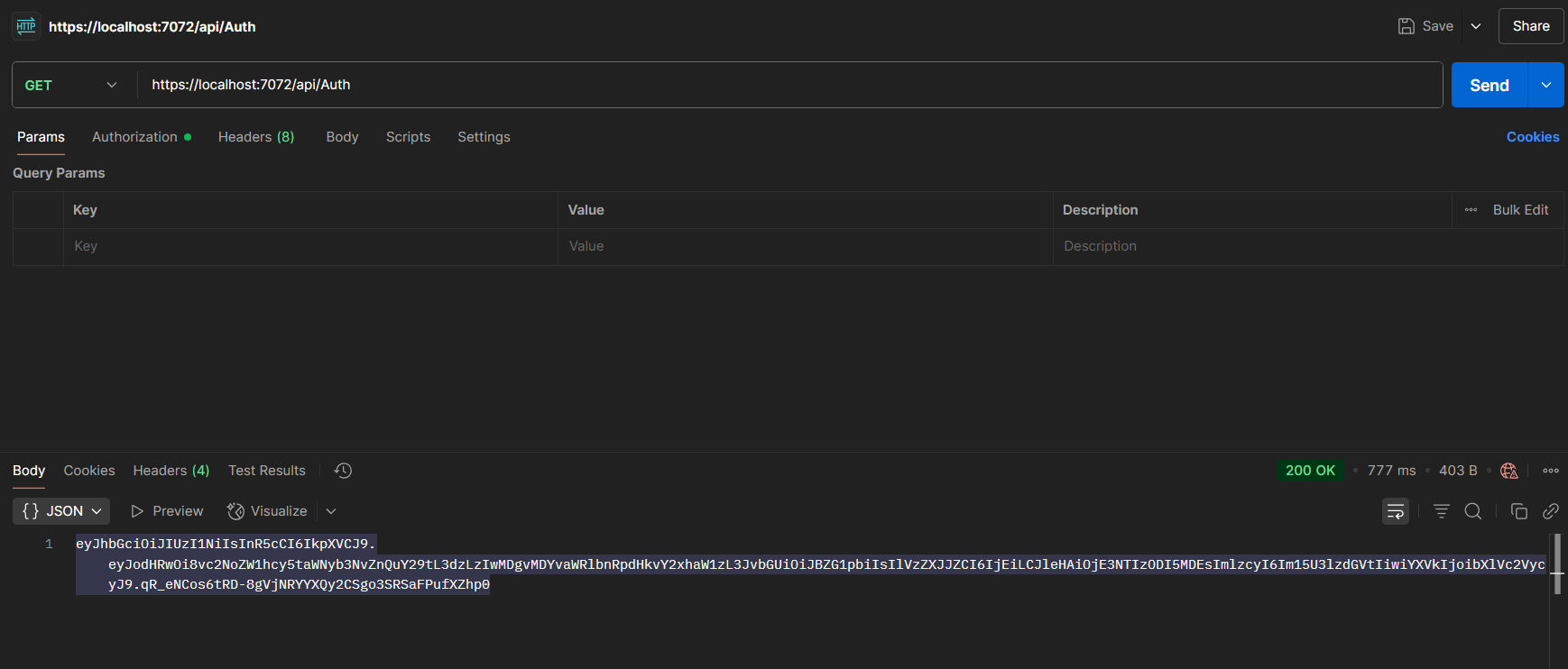
EmployeeController.cs

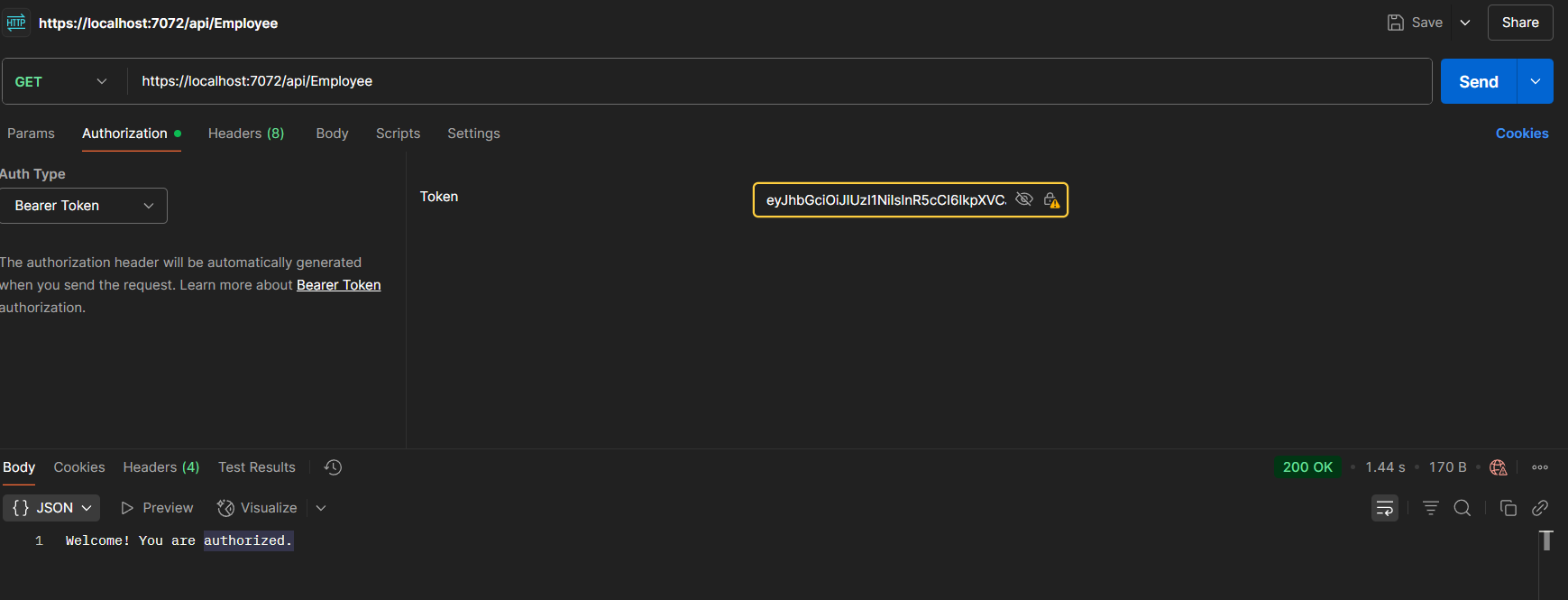


STEP 2: Generate a JWT Token Using AuthController

STEP 3. Use That Token in a GET Request to /api/Employee in Postman

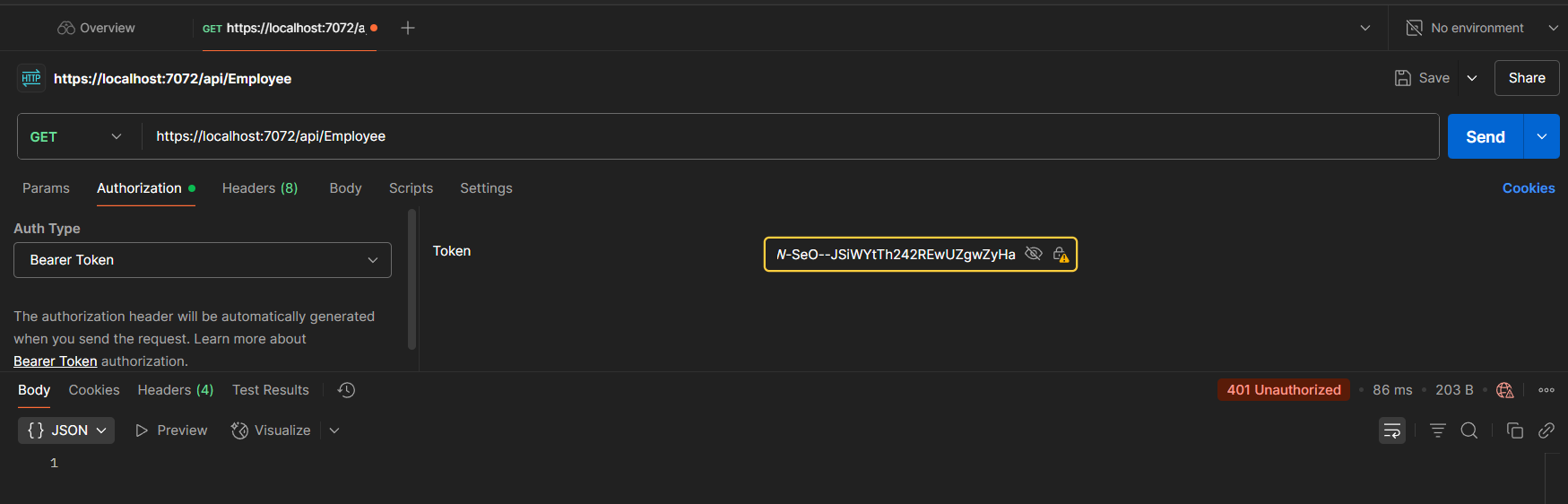
**OUTPUT FOR STEP 3:**





STEP 4.: Modify the Token in Postman and Re-Test

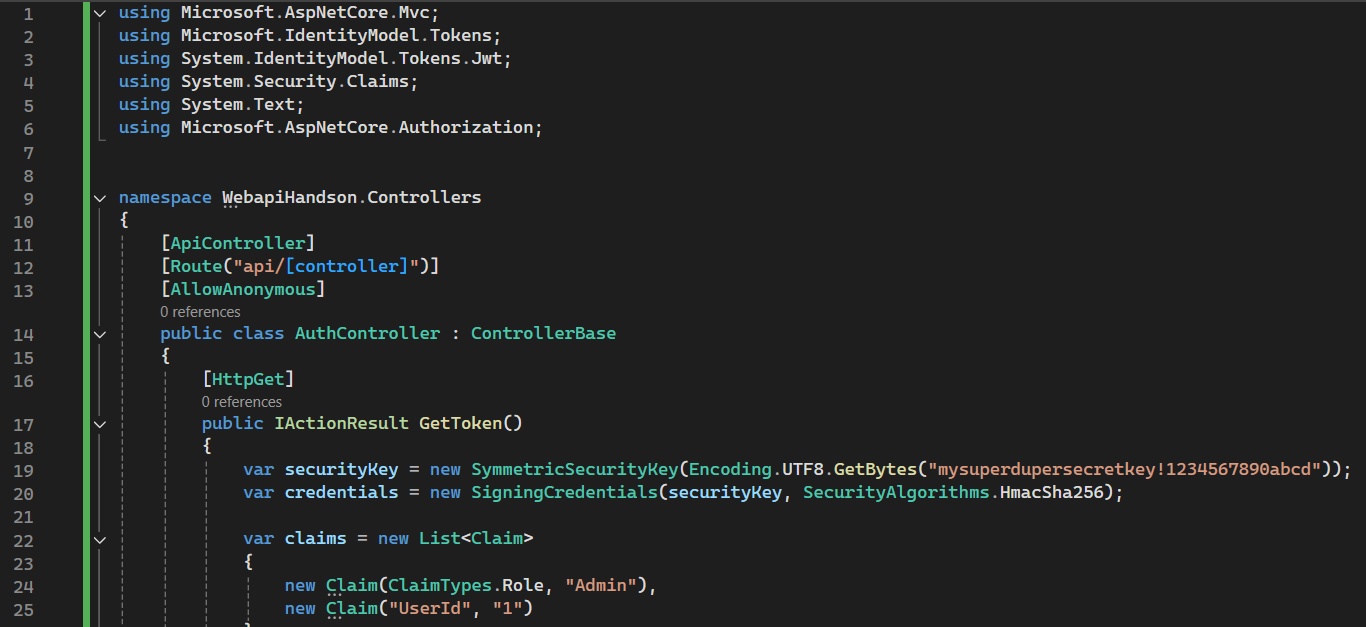
**OUTPUT FOR STEP 4:**

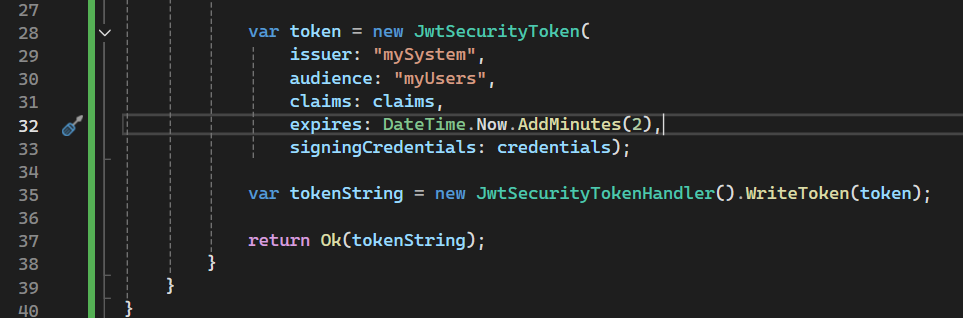


**TASK 3: Check for JWT expiration**

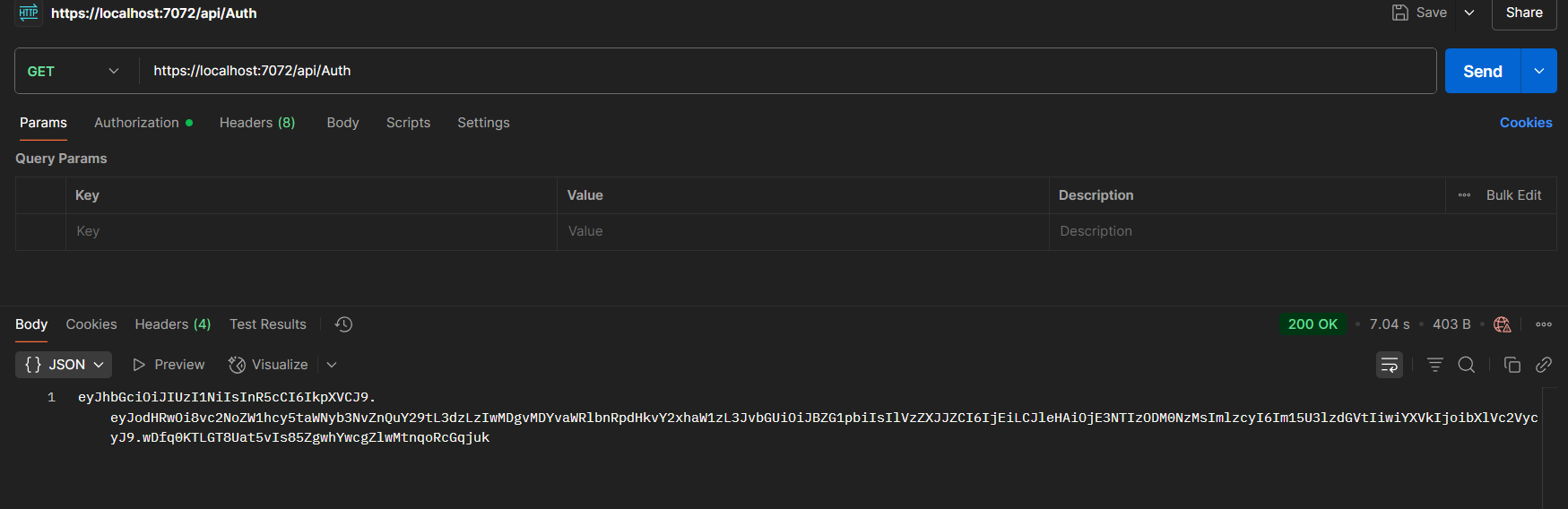
**STEP 1:** Update the expiration time from 10 minutes to **2 minutes**

**AuthController.cs**

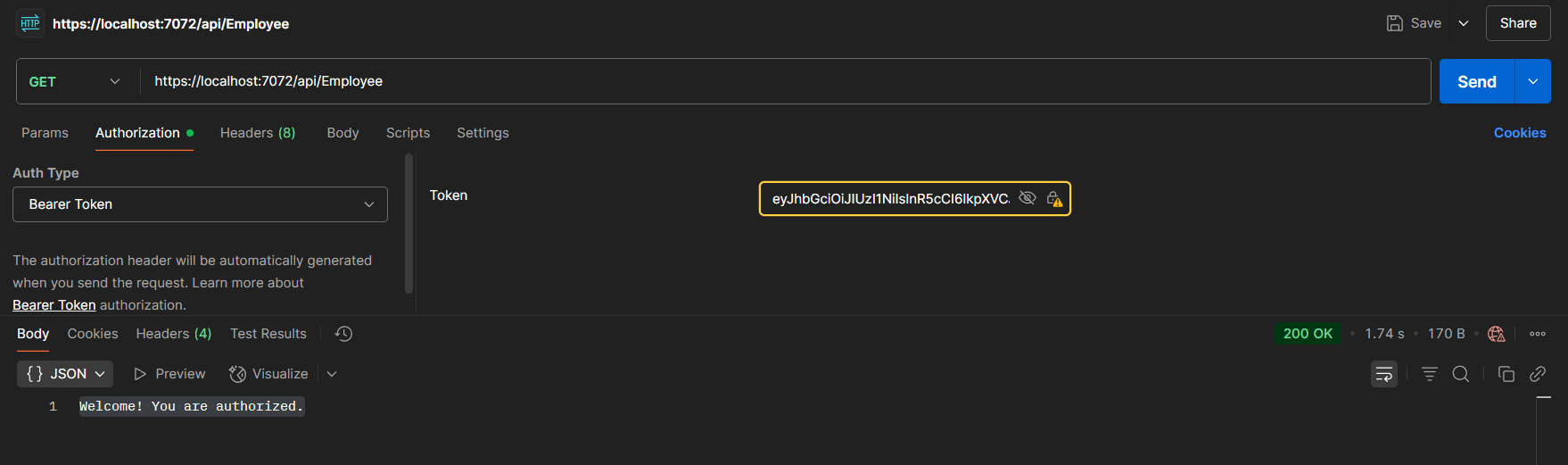




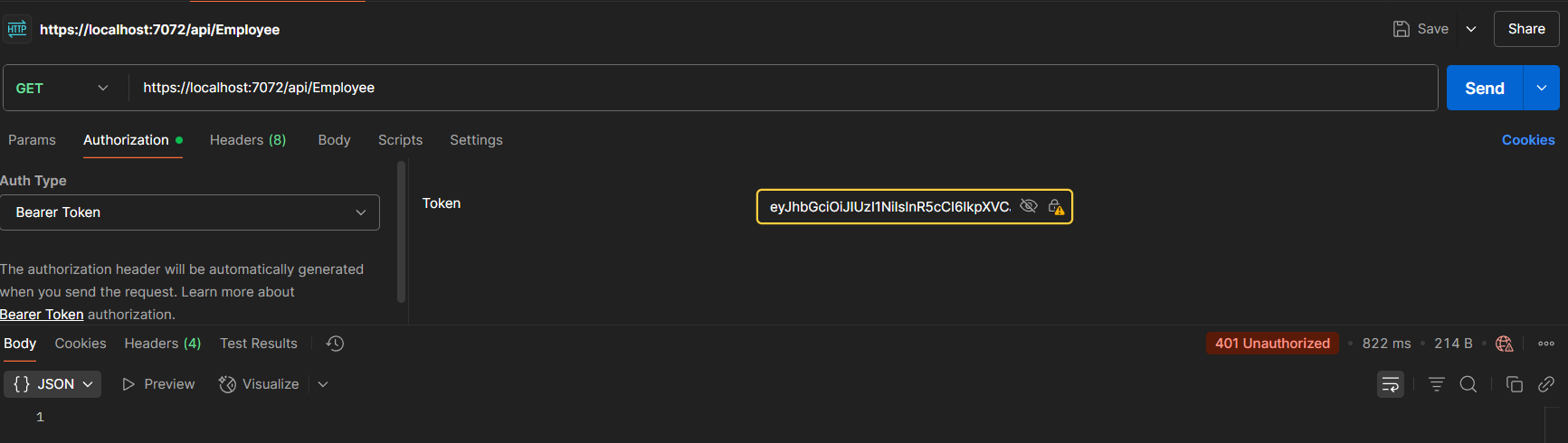
STEP 2: Run the API and Generate a JWT Token



STEP 3 : Test the Token **Before Expiry**

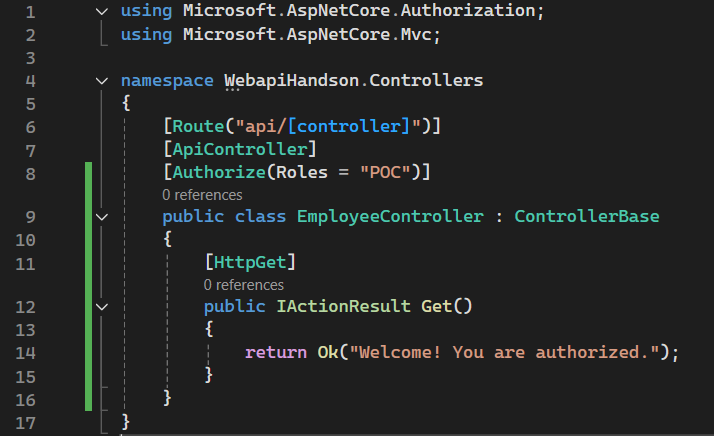


STEP 4: Use the Same Token After Expiry [After 2 mins]

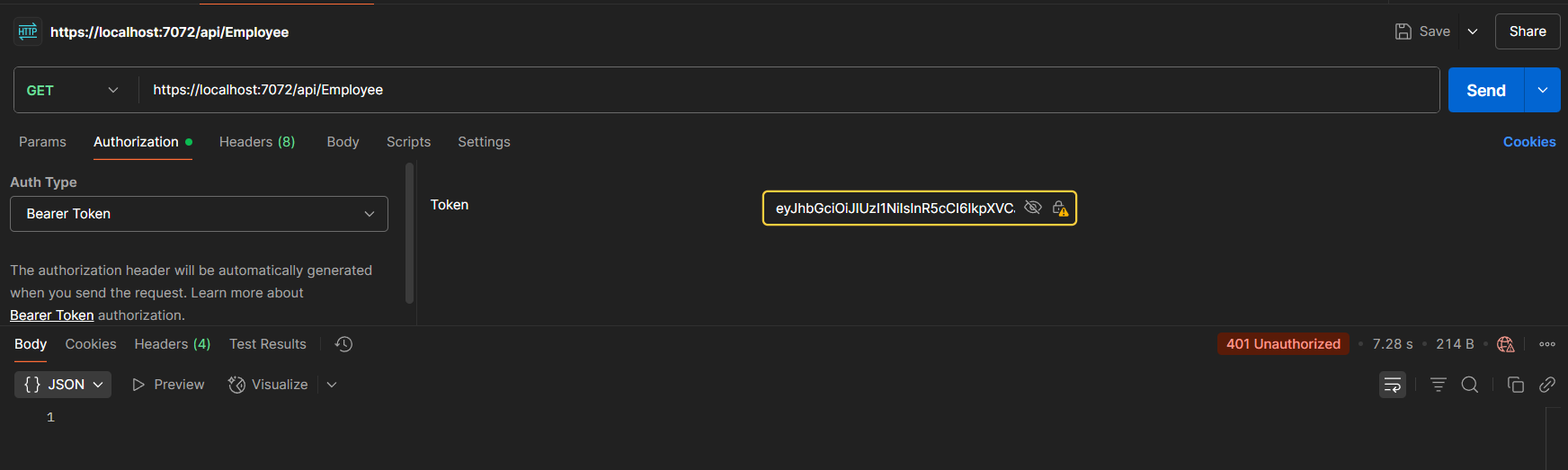


**TASK 4: Add the roles to be authorized in the Authorize attribute.**

**STEP 1: Restrict to Role = POC** in EmployeeController.cs



STEP 2: Use the Existing Admin Token in Postman



STEP 3: Update EmployeeController to Allow Multiple Roles: Admin or POC and use GET again in POSTMAN

