LABSHEET 1B

Module

Enterprise Solutions Development

Module Code

IT53003FP

Duration

6 hours

Title: Implementing Web API for CRUD operations using Entity Framework Core

Objective(s):

In this lab, students will learn**:**
Part A: Implement **GET** method
Part B: Implement **POST** method
Part C: Implement **PUT** method
Part D: Implement **DELETE** method

Tools, Equipment and Materials:

1 Hardware: Personal Computer with Internet access
2 Operating System: Windows, Mac, Linux
3 Editor: Visual Studio 2022
4 Dot Net Core SDK 6: This is the software development KIT and this KIT is helpful for the development and running of the application in the system.
5 Database: SQL Server 2019
6 SSMS (SQL Server Management Studio): To interact with SQL Server database

Instructions:

Understanding CRUD Operations.
In general, when we talk about a database table, there are four operations (known as CRUD Operations) that we perform on it. They are as follows

1. **C- Create a Row,**
2. **R- Read a Row,**
3. **U- Update a Row, and**
4. **D- Delete a Row**

From the context of an ASP.NET Web API resource, the above four actions (**Read, Create, Update and Delete**) are corresponded to **GET, POST, PUT and DELETE** methods respectively. Let us understand some concepts which are required to understand the HTTP verbs.

Request Verbs (GET, POST, AND DELETE):
- These describe what should be done with the resource.

Request Header:
- When a client sends a request to the server, the request contains a header and a body. The request method contains additional information, such as – what type of response is required. For example, the response should be in XML, JSON, or some other format.

LABSHEET 1B

Module
Enterprise Solutions Development

Module Code
IT53003FP

Duration
6 hours

Request Body:
- The request body contains the data that we want to send to the server. For example, a post request contains the data for a new item that we want to create. The data format may be in XML or in JSON

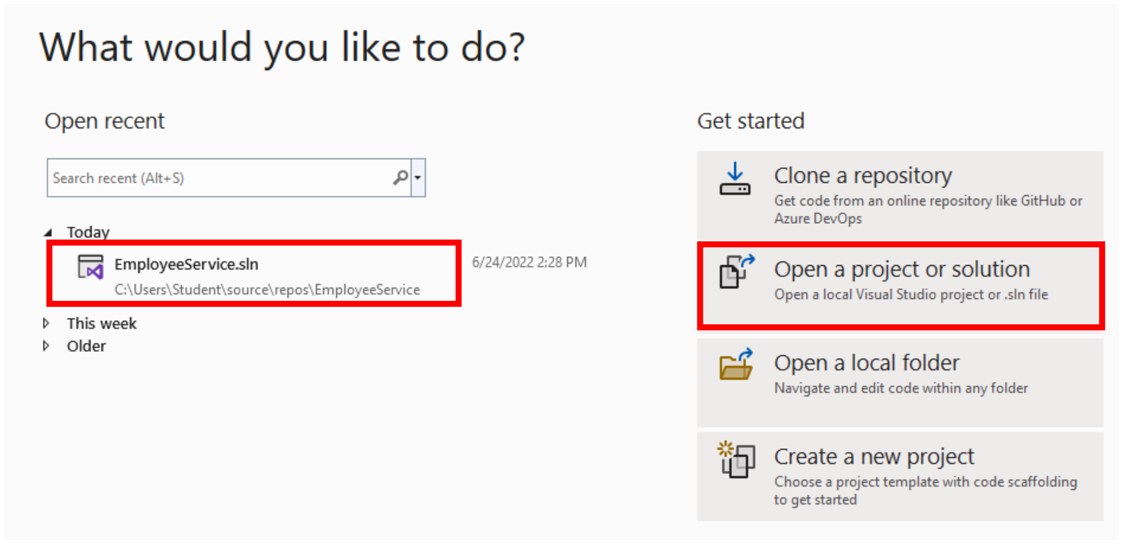
Response Body:
- The response body contains the data of the response from the server. For example, if the request is for a specific employee, the response body includes employee details in XML, JSON, and so on.

Response Status Codes:
- The Response Status codes are nothing but the HTTP status codes which specifically gives the status of the response to the client. Some of the common status codes are 404 not found, 200 OK, 204 No content, 500 Internal Server Error and so on.

Part A: Implement GET method

1. This lab is a continuation of the previous lab (Lab 1A) where we created the infrastructure for our EmployeeService Web API project.

2. Open your EmployeeService Web API project from Visual Studio 2022, either from 'Open recent' or 'Open a project or solution' as shown below.


What would you like to do?
Open recent
Search recent (Alt+S)
Today
EmployeeService.sln
C:\Users\Student\source\repos\EmployeeService
6/24/2022 2:28 PM
This week
Older
Get started
Clone a repository
Get code from an online repository like GitHub or Azure DevOps
Open a project or solution
Open a local Visual Studio project or .sln file
Open a local folder
Navigate and edit code within any folder
Create a new project
Choose a project template with code scaffolding to get started

In this section we will implement more Get action methods in our Web API controller class that will handle HTTP GET requests.

3. As per Web API naming convention, action method that starts with a word "Get" will handle HTTP GET request. We can either name it only Get or with any suffix. Open EmployeesControllers.cs and look at our first Get action method that will

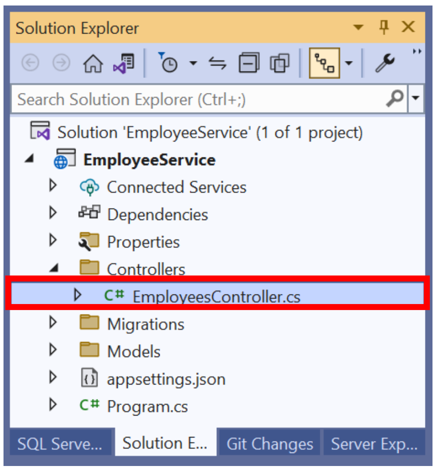
LABSHEET 1B

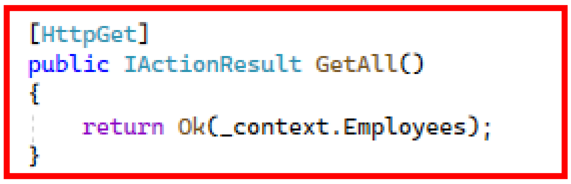
Module
Enterprise Solutions Development

Module Code
IT53003FP

Duration
6 hours

return all the employees records from the DB. Following an appropriate naming methodology increases readability and anybody can understand the purpose of a method easily.

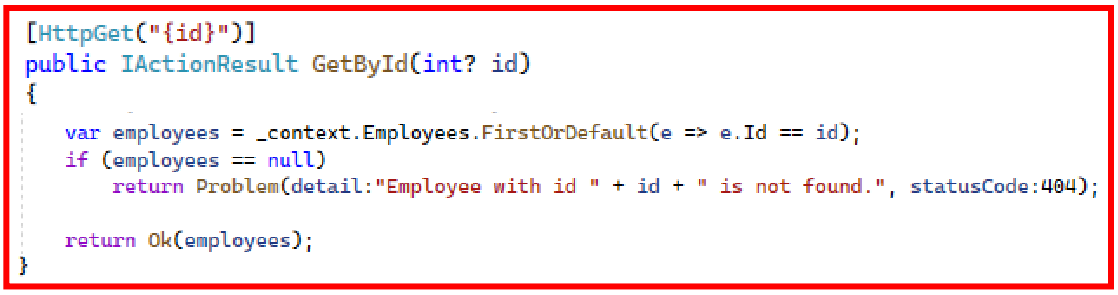




```
[HttpGet]
public IActionResult GetAll()
{
    return Ok(_context.Employees);
}
```

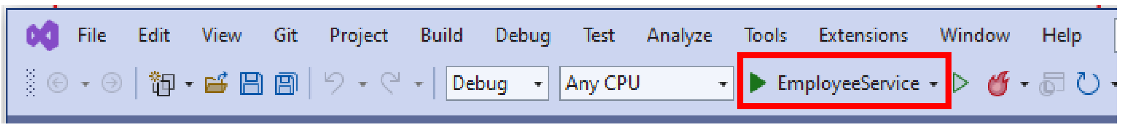
As you can see in the above example, GetAll() method returns all the employee records using EF. If no employee exists in the DB, then it will return **404 NotFound** response, otherwise it will return **200 OK** response with employees data. The NotFound() and Ok() methods defined in the ApiController returns 404 and 200 response respectively.

4. Next, below 'GetAll()', add a new **GetById()** method with an 'id' parameter as shown below in the EmployeesController.cs. Then it will return the specific employee with the id that we passed in the method.



```
[HttpGet("{id}")]
public IActionResult GetById(int? id)
{
    var employees = _context.Employees.FirstOrDefault(e => e.Id == id);
    if (employees == null)
        return Problem(detail:"Employee with id " + id + " is not found.", statusCode:404);

    return Ok(employees);
}
```

5. Click on the EmployeeService green button as shown in the below image to run the EmployeeService Web API application.



LABSHEET 1B

Module

Enterprise Solutions Development

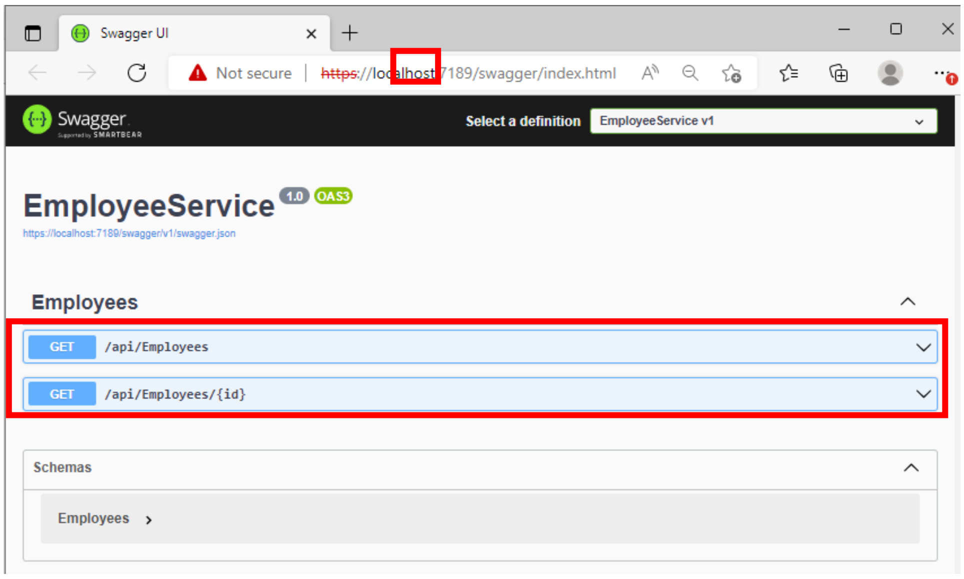
Module Code

IT53003FP

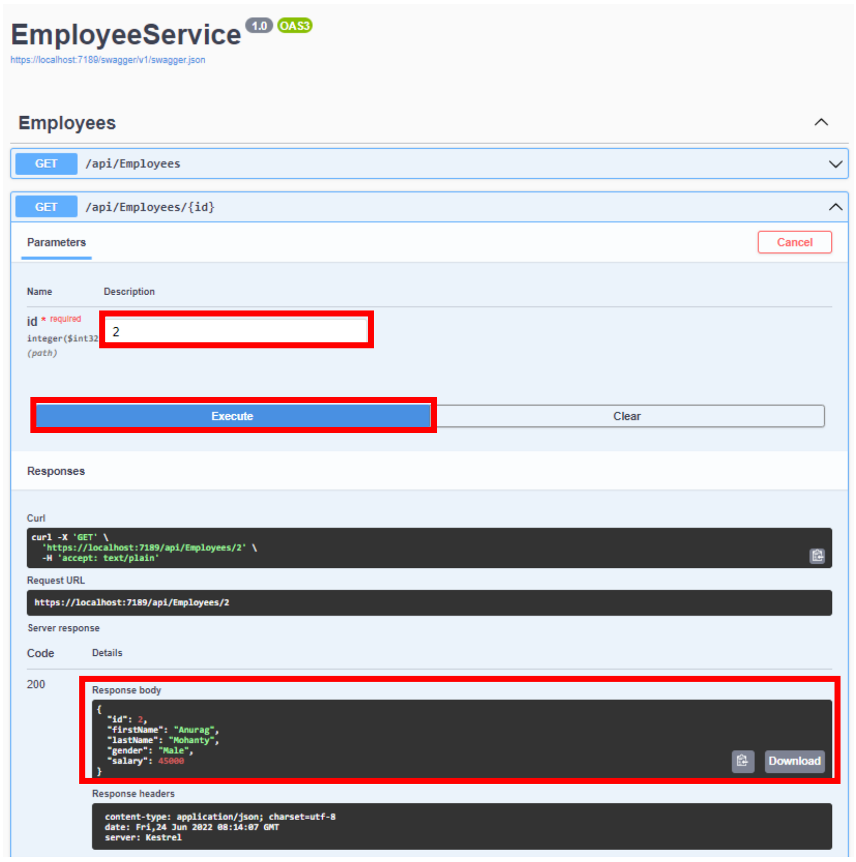
Duration

6 hours

6. The application is now running in the following swagger page with both GET actions in the browser with Port number (7189).



7. You may now click on the 'Try it out' and 'Execute' the 2nd GET action by enter **'2'** in the id textbox to display details of the employee whose Id=2.



```
curl -X 'GET' \
  'https://localhost:7189/api/Employees/2' \
  -H 'accept: text/plain'
```

```
https://localhost:7189/api/Employees/2
```

```
{
  "id": 2,
  "firstName": "Anurag",
  "lastName": "Mohanty",
  "gender": "Male",
  "salary": 45000
}
```

```
content-type: application/json; charset=utf-8
date: Fri,24 Jun 2022 08:14:07 GMT
server: Kestrel
```

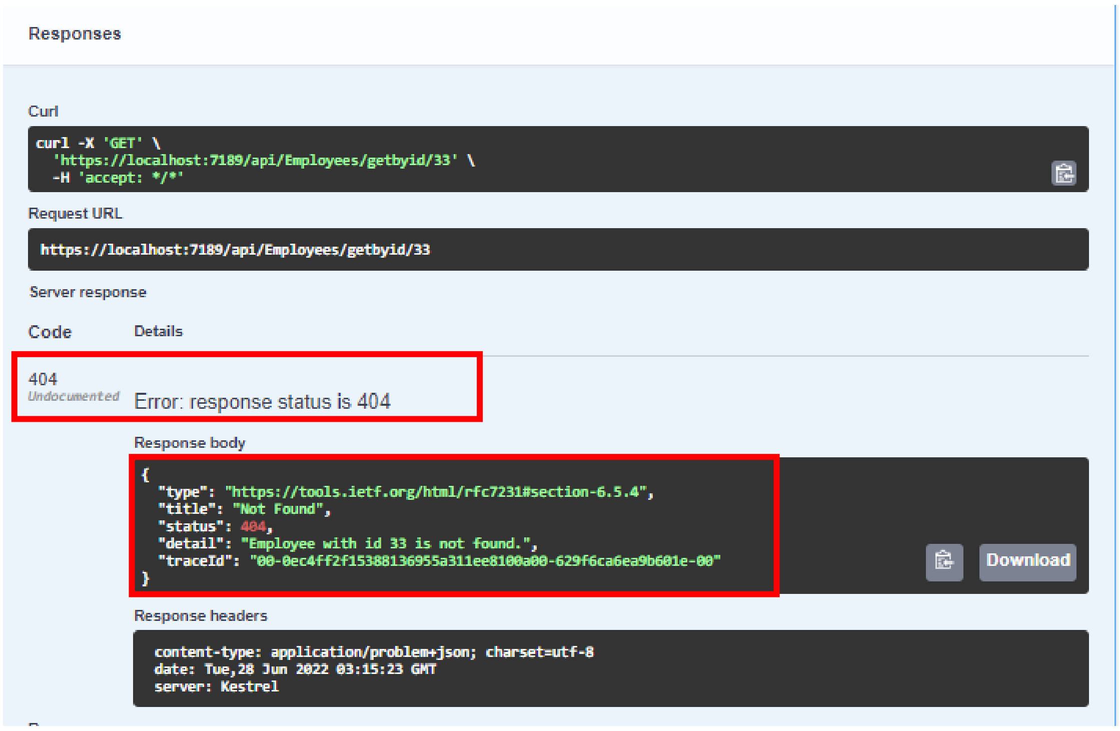
LABSHEET 1B

Module
Enterprise Solutions Development

Module Code
IT53003FP

Duration
6 hours

8. Let's see what will happen when we issue a request for an employee whose id = '**33**' that does not exist with one example.



Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7189/api/Employees/getbyid/33' \
  -H 'accept: */*'
```

Request URL

```
https://localhost:7189/api/Employees/getbyid/33
```

Server response

Code	Details
404 *Undocumented*	Error: response status is 404

Response body

```
{
  "type": "https://tools.ietf.org/html/rfc7231#section-6.5.4",
  "title": "Not Found",
  "status": 404,
  "detail": "Employee with id 33 is not found.",
  "traceId": "00-0ec4ff2f15388136955a311ee8100a00-629f6ca6ea9b601e-00"
}
```

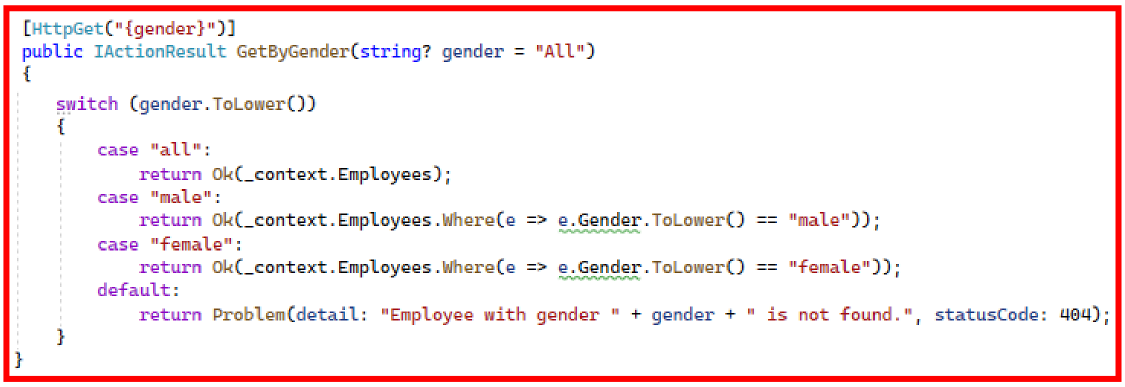
Response headers

```
content-type: application/problem+json; charset=utf-8
date: Tue,28 Jun 2022 03:15:23 GMT
server: Kestrel
```

At this point, when we issue a request for an employee with **id = 33** which does not exist we get a **404** status code along with the title "**Not Found**" and meaningful message such as **"Employee with id 33 is not found**." as shown above.

9. Now, let's add another new **GetByGender()** action method into the controller.

   Suppose you want to implement another GET method to get either a male, female, or even all the employees regardless of their gender, then you may create another Get method as shown below.



```
[HttpGet("{gender}")]
public IActionResult GetByGender(string? gender = "All")
{
    switch (gender.ToLower())
    {
        case "all":
            return Ok(_context.Employees);
        case "male":
            return Ok(_context.Employees.Where(e => e.Gender.ToLower() == "male"));
        case "female":
            return Ok(_context.Employees.Where(e => e.Gender.ToLower() == "female"));
        default:
            return Problem(detail: "Employee with gender " + gender + " is not found.", statusCode: 404);
    }
}
```

10. Click on the EmployeeService green button to run the EmployeeService Web API application in swagger page with all GET actions in the browser.

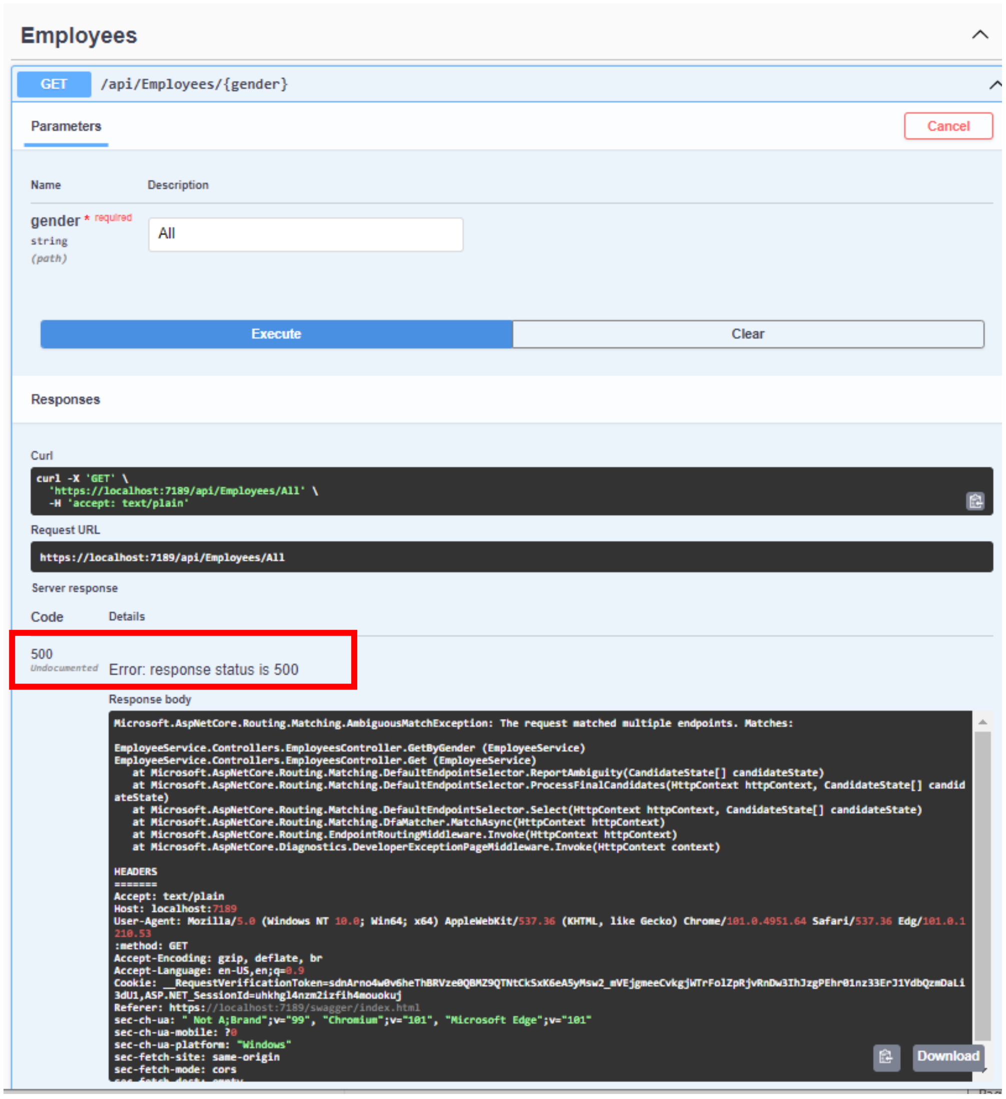
LABSHEET 1B

11. You may now click on the ‘Try it out’ and ‘Execute’ the GET action by enter **‘All’** (set as default) in the gender textbox.

    The web API project will compile without an error but when you execute HTTP GET request then it will respond with the following error:

```
Microsoft.AspNetCore.Routing.Matching.AmbiguousMatchException: The request matched multiple endpoints. Matches:

EmployeeService.Controllers.EmployeesController.GetByGender (EmployeeService)
EmployeeService.Controllers.EmployeesController.Get (EmployeeService)
```

    This is because you cannot have multiple action methods with same number of parameters with same type. Due to both action methods (GetById & GetByGender) above have the same number of parameters, Web API does not understand which method to execute for the HTTP GET request (http://localhost:7189/api/Employees/1 or http://localhost:7189/api/Employees/All).



```
curl -X 'GET' \
  'https://localhost:7189/api/Employees/All' \
  -H 'accept: text/plain'
```

```
https://localhost:7189/api/Employees/All
```

```
Microsoft.AspNetCore.Routing.Matching.AmbiguousMatchException: The request matched multiple endpoints. Matches:

EmployeeService.Controllers.EmployeesController.GetByGender (EmployeeService)
EmployeeService.Controllers.EmployeesController.Get (EmployeeService)
   at Microsoft.AspNetCore.Routing.Matching.DefaultEndpointSelector.ReportAmbiguity(CandidateState[] candidateState)
   at Microsoft.AspNetCore.Routing.Matching.DefaultEndpointSelector.ProcessFinalCandidates(HttpContext httpContext, CandidateState[] candid
ateState)
   at Microsoft.AspNetCore.Routing.Matching.DefaultEndpointSelector.Select(HttpContext httpContext, CandidateState[] candidateState)
   at Microsoft.AspNetCore.Routing.Matching.DfaMatcher.MatchAsync(HttpContext httpContext)
   at Microsoft.AspNetCore.Routing.EndpointRoutingMiddleware.Invoke(HttpContext httpContext)
   at Microsoft.AspNetCore.Diagnostics.DeveloperExceptionPageMiddleware.Invoke(HttpContext context)

HEADERS
=======
Accept: text/plain
Host: localhost:7189
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/101.0.4951.64 Safari/537.36 Edg/101.0.1
210.53
:method: GET
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: __RequestVerificationToken=sdnArno4w0v6heThBRVze0QBMZ9QTNtCkSxK6eA5yMsw2_mVEjgmeeCvkgjWTrFolZpRjvRnDw3IhJzgPEhr01nz33ErJ1YdbQzmDaLi
3dU1,ASP.NET_SessionId=uhkhgl4nzm2izfih4mouokuj
Referer: https://localhost:7189/swagger/index.html
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="101", "Microsoft Edge";v="101"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
sec-fetch-site: same-origin
sec-fetch-mode: cors
```

LABSHEET 1B

Module

Enterprise Solutions Development

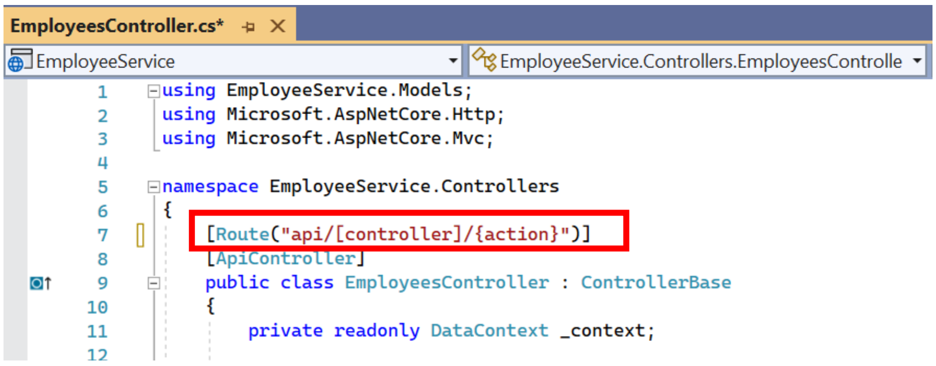
Module Code

IT53003FP

Duration

6 hours

12. To solve the issue to implement multiple GET methods with different parameters and types, in ‘EmployeesController.cs’, add an **‘{action}**’ placeholder in the route template as shown below.



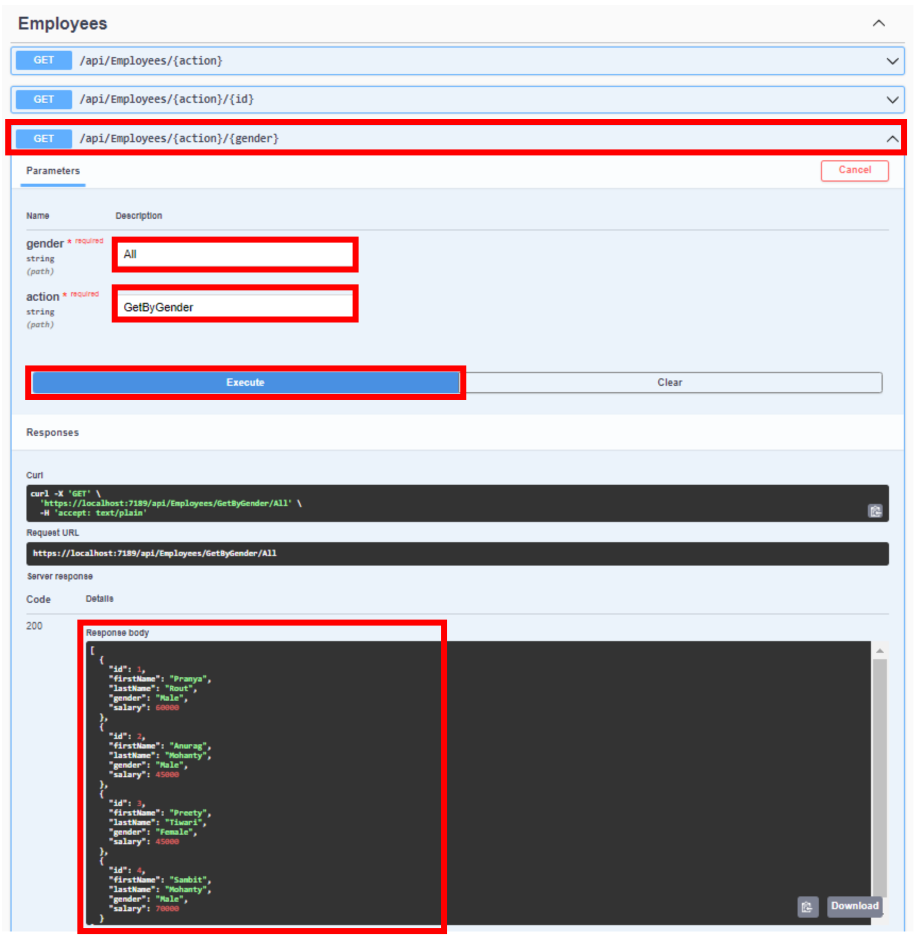
EmployeesController.cs*

EmployeeService | EmployeeService.Controllers.EmployeesControlle

```
using EmployeeService.Models;
using Microsoft.AspNetCore.Http;
using Microsoft.AspNetCore.Mvc;

namespace EmployeeService.Controllers
{
    [Route("api/[controller]/{action}")]
    [ApiController]
    public class EmployeesController : ControllerBase
    {
        private readonly DataContext _context;
```

13. Click on the EmployeeService green button to run the EmployeeService Web API application in swagger page with all GET actions in the browser.

14. You may now click on the ‘Try it out’ to ‘execute’ the GET action by enter **‘All’** in the gender textbox and ‘**GetByGender**’ as action name.



Employees

GET /api/Employees/{action}

GET /api/Employees/{action}/{id}

GET /api/Employees/{action}/{gender}

Parameters | Cancel

Name	Description
gender * required string (path)	All
action * required string (path)	GetByGender

Execute | Clear

Responses

Curl

```
curl -X 'GET' \
  'https://localhost:7189/api/Employees/GetByGender/All' \
  -H 'accept: text/plain'
```

Request URL

```
https://localhost:7189/api/Employees/GetByGender/All
```

Server response

Code: 200

Details: Response body

```
[
  {
    "id": 1,
    "firstName": "Pranya",
    "lastName": "Rout",
    "gender": "Male",
    "salary": 60000
  },
  {
    "id": 2,
    "firstName": "Anurag",
    "lastName": "Mohanty",
    "gender": "Male",
    "salary": 45000
  },
  {
    "id": 3,
    "firstName": "Preety",
    "lastName": "Tiwari",
    "gender": "Female",
    "salary": 45000
  },
  {
    "id": 4,
    "firstName": "Sambit",
    "lastName": "Mohanty",
    "gender": "Male",
    "salary": 70000
  }
```

LABSHEET 1B

Module

Enterprise Solutions Development

Module Code

IT53003FP

Duration

6 hours

Part B: Implement POST method

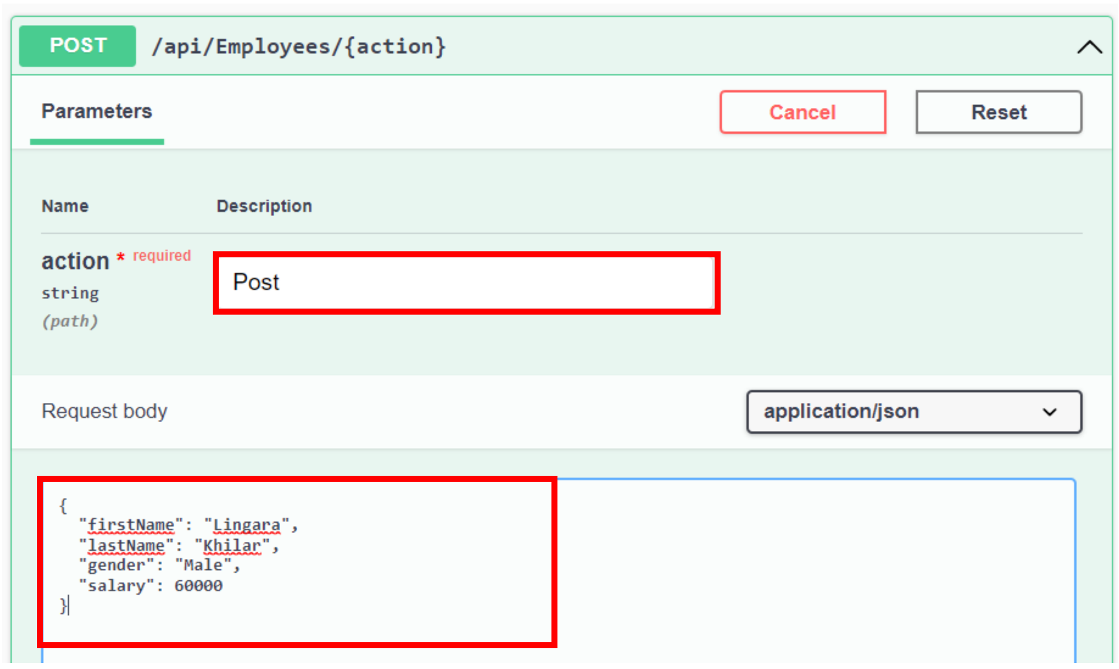
The Post Method in the Web API application allows us to create a new item. Here, we want to add a new Employee to the Employees table. First, Include the following **Post()** method within the EmployeesController. Notice that the employee object is being passed as a parameter to the Post method.
The Employees parameter is decorated with the employee attribute to tell the Web API to get the employee data from the request body.

15. Continue in EmployeesController.cs, add a new **Post()** method as shown below:

```
[HttpPost]
public IActionResult Post(Employee  employee)
{
    _context.Employees.Add(employee);
    _context.SaveChanges();

    return CreatedAtAction("GetAll", new { id = employee.Id }, employee);
}
```

16. Click on the EmployeeService green button to run the EmployeeService Web API application in swagger page with all actions in the browser.

17. You may now click on the ‘Try it out’ and ‘Execute’ the POST action by enter ‘**Post’** as action name, then modify the Request body to insert the new employee record into the database.



```
{
  "firstName": "Lingara",
  "lastName": "Khilar",
  "gender": "Male",
  "salary": 60000
}
```

LABSHEET 1B

Module

Enterprise Solutions Development

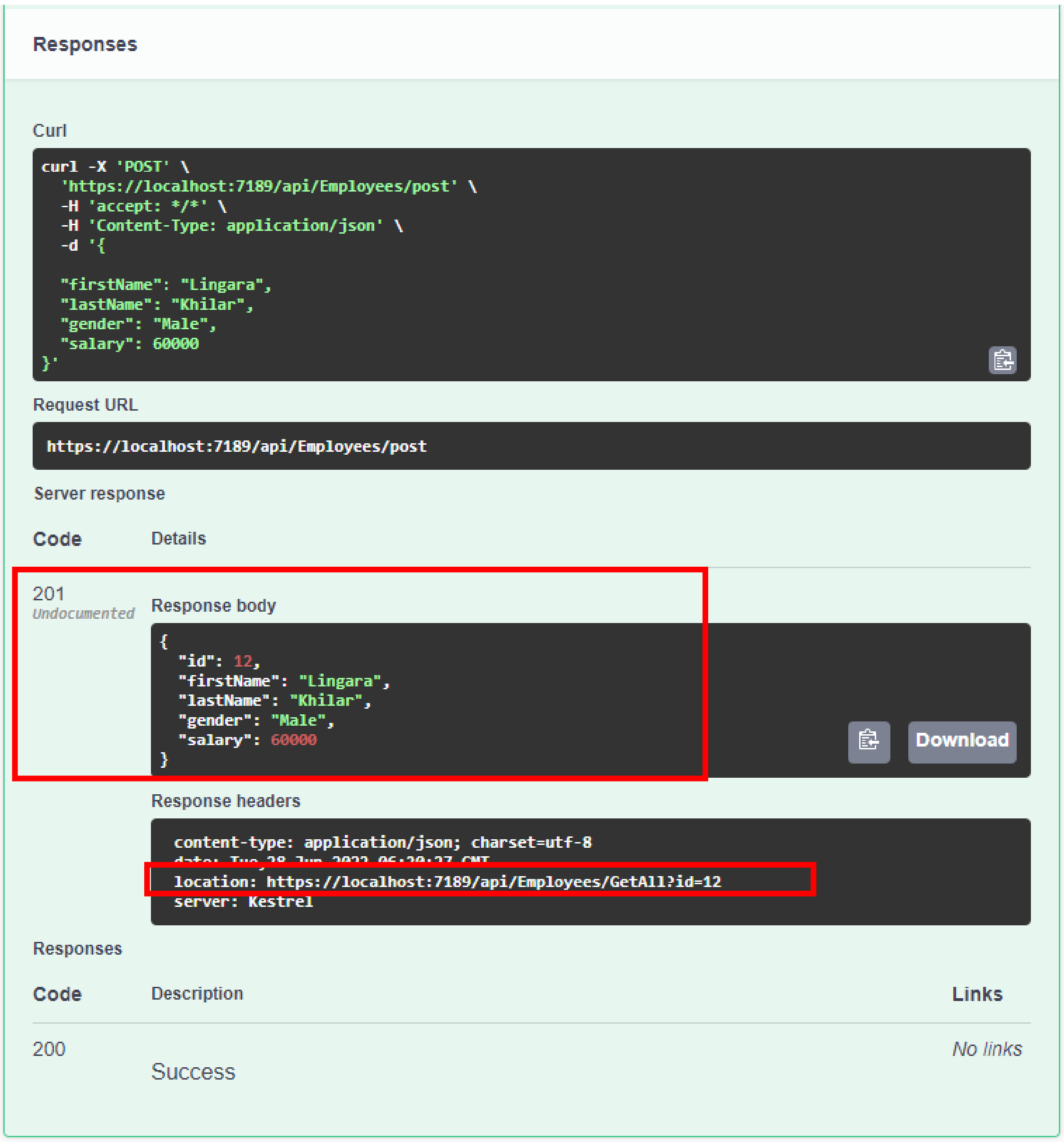
Module Code

IT53003FP

Duration

6 hours

18. CreatedAtAction method allows us to set Location URI of the newly created resource by *specifying the name of an action (e.g. GetAll)* where we can retrieve our resource.



Responses

Curl

```
curl -X 'POST' \
  'https://localhost:7189/api/Employees/post' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{

  "firstName": "Lingara",
  "lastName": "Khilar",
  "gender": "Male",
  "salary": 60000
}'
```

Request URL

```
https://localhost:7189/api/Employees/post
```

Server response

Code	Details
201 *Undocumented*	Response body

```
{
  "id": 12,
  "firstName": "Lingara",
  "lastName": "Khilar",
  "gender": "Male",
  "salary": 60000
}
```

Download

Response headers

```
 content-type: application/json; charset=utf-8
 date: Tue,28 Jun 2022 06:20:27 GMT
 location: https://localhost:7189/api/Employees/GetAll?id=12
 server: Kestrel
```

Responses

Code	Description	Links
200	Success	*No links*

Points to Remember while working with Web API **Post()** Method:

1. If a method return type is void in Web API Service then by default Web API Service returns the status code **204 No Content**.
2. When a new item is created, we should be returning status code **201 Item Created**.
3. With the **201** status code, we should also include the location i.e. URI of the newly created item.

Module

Enterprise Solutions Development

Module Code

IT53003FP

Duration

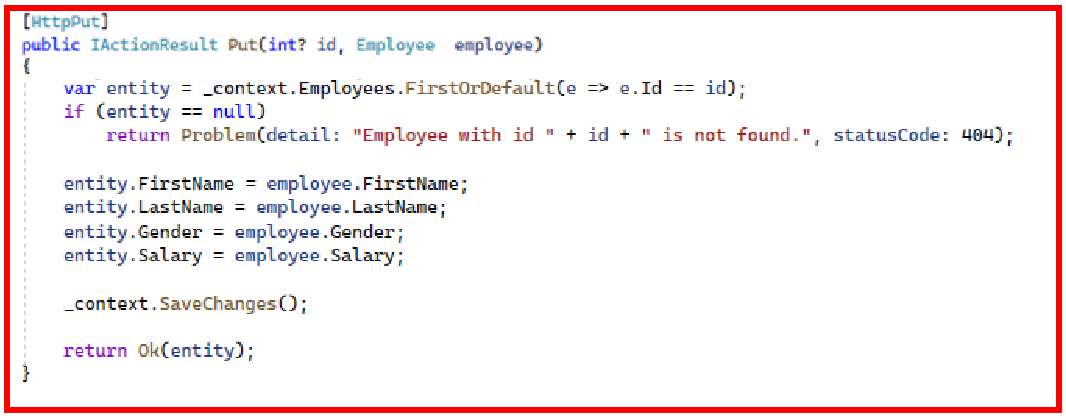
6 hours

LABSHEET 1B

Part C: **Implement PUT method**

19. The PUT method in Web API allows us to update an item. Here, we want to update the employee by Id. Include the following **Put()** method in EmployeesController.

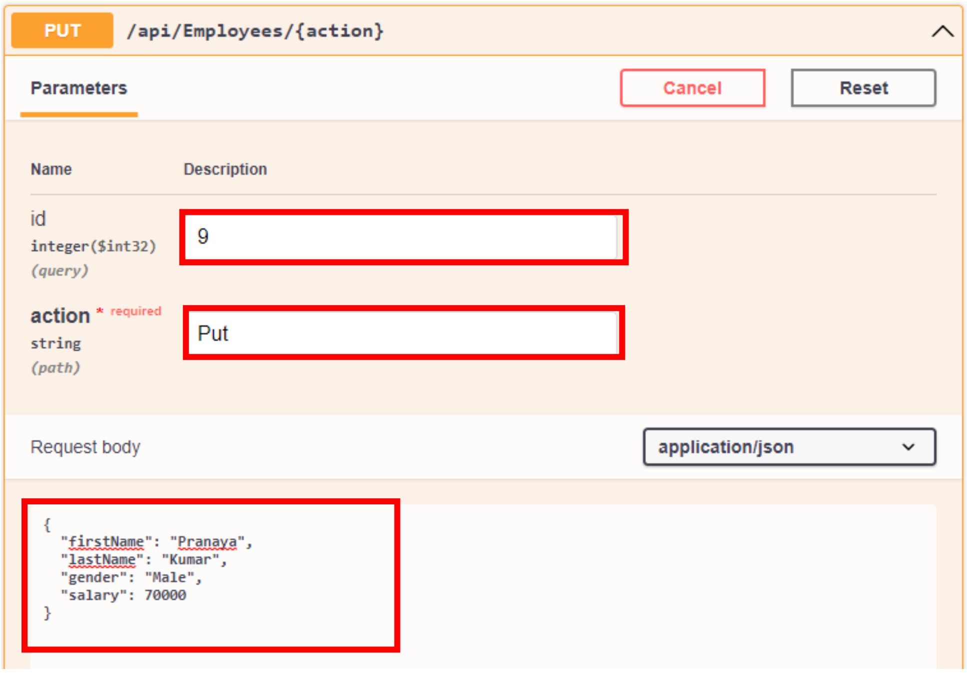
    Notice the id of the employee that we want to update and the Employee object with which we want to update are being passed as parameters to the Put method. The Employees parameter is decorated with the employee attribute. This tells Web API to get employee data from the request body.



```
[HttpPut]
public IActionResult Put(int? id, Employee  employee)
{
    var entity = _context.Employees.FirstOrDefault(e => e.Id == id);
    if (entity == null)
        return Problem(detail: "Employee with id " + id + " is not found.", statusCode: 404);

    entity.FirstName = employee.FirstName;
    entity.LastName = employee.LastName;
    entity.Gender = employee.Gender;
    entity.Salary = employee.Salary;

    _context.SaveChanges();

    return Ok(entity);
}
```

20. Click on the EmployeeService green button to run the EmployeeService Web API application in swagger page with all actions in the browser.

21. You may now click on the ‘Try it out’ and ‘Execute’ the PUT action by enter **‘9’** as id and **‘Put’** as action name, then modify the Request body to update the existing employee record into the database.



id integer($int32) (query): 9

action * required string (path): Put

```
{
  "firstName": "Pranaya",
  "lastName": "Kumar",
  "gender": "Male",
  "salary": 70000
}
```

LABSHEET 1B

Module

Enterprise Solutions Development

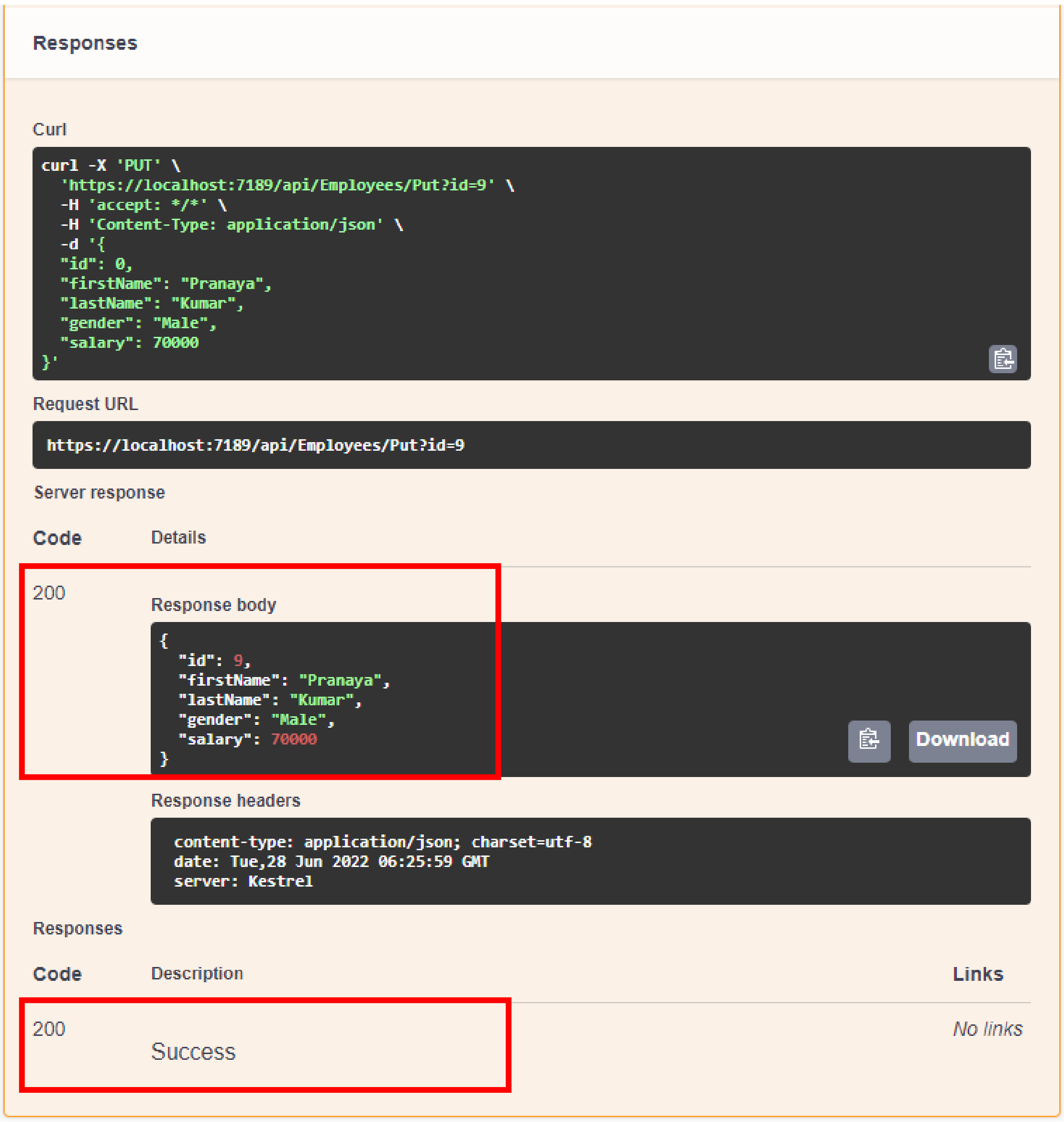
Module Code

IT53003FP

Duration

6 hours

22. When the update is successful and return status code **200 OK** indicating that the update is successful.



Responses

Curl

```
curl -X 'PUT' \
  'https://localhost:7189/api/Employees/Put?id=9' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 0,
  "firstName": "Pranaya",
  "lastName": "Kumar",
  "gender": "Male",
  "salary": 70000
}'
```

Request URL

```
https://localhost:7189/api/Employees/Put?id=9
```

Server response

Code: 200

Response body

```
{
  "id": 9,
  "firstName": "Pranaya",
  "lastName": "Kumar",
  "gender": "Male",
  "salary": 70000
}
```

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,28 Jun 2022 06:25:59 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	Success	*No links*

23. Notice in the response header we have status code **200 OK**. Also, when we try to update an employee whose id (e.g. 90) does not exist, we get the status code **404 Not Found** instead of **500 Internal Server Error** as shown below.

LABSHEET 1B

Module

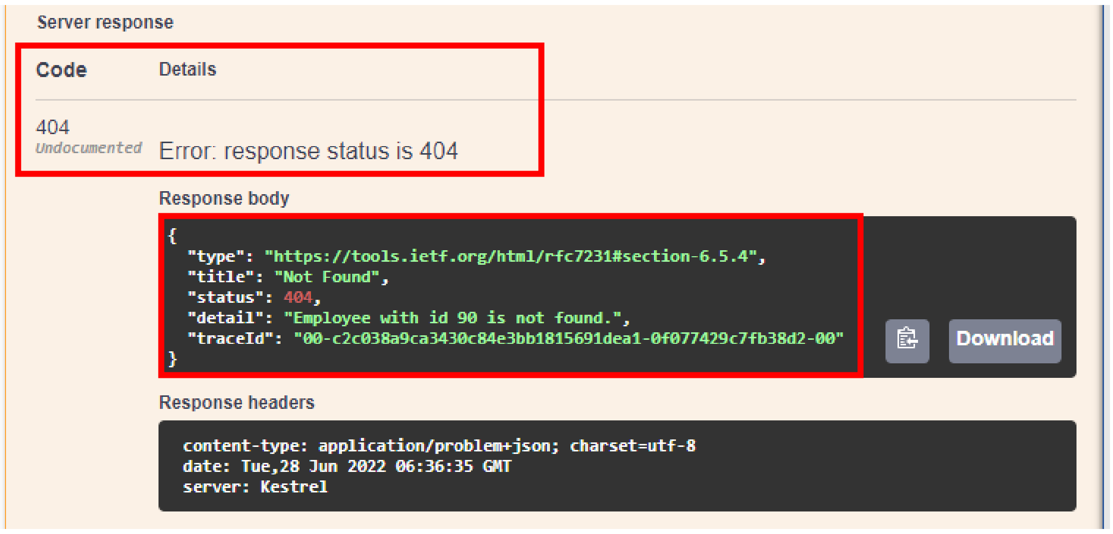
Enterprise Solutions Development

Module Code

IT53003FP

Duration

6 hours



```
{
  "type": "https://tools.ietf.org/html/rfc7231#section-6.5.4",
  "title": "Not Found",
  "status": 404,
  "detail": "Employee with id 90 is not found.",
  "traceId": "00-c2c038a9ca3430c84e3bb1815691dea1-0f077429c7fb38d2-00"
}
```

```
content-type: application/problem+json; charset=utf-8
date: Tue,28 Jun 2022 06:36:35 GMT
server: Kestrel
```

Part D: **Implement DELETE method**

24. The Delete Method in Web API allows us to delete an item. We want to delete a specified employee from the Employees database table. To achieve this Include the following **Delete()** method in EmployeesController.

```
[HttpDelete]
public IActionResult Delete(int? id, Employee employee)
{
    var entity = _context.Employees.FirstOrDefault(e => e.Id == id);
    if (entity == null)
        return Problem(detail: "Employee with id " + id + " is not found.", statusCode: 404);

    _context.Employees.Remove(entity);
    _context.SaveChanges();

    return Ok(entity);
}
```

25. Click on the EmployeeService green button to run the EmployeeService Web API application in swagger page with all actions in the browser.

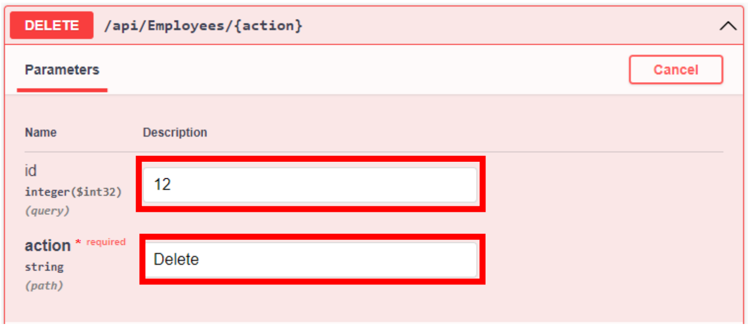
26. You may now click on the ‘Try it out’ and ‘Execute’ the DELETE action by enter **‘12’** as id and **‘Delete’** as action name, then modify the Request body to update the existing employee record into the database.

LABSHEET 1B

Module
Enterprise Solutions Development

Module Code
IT53003FP

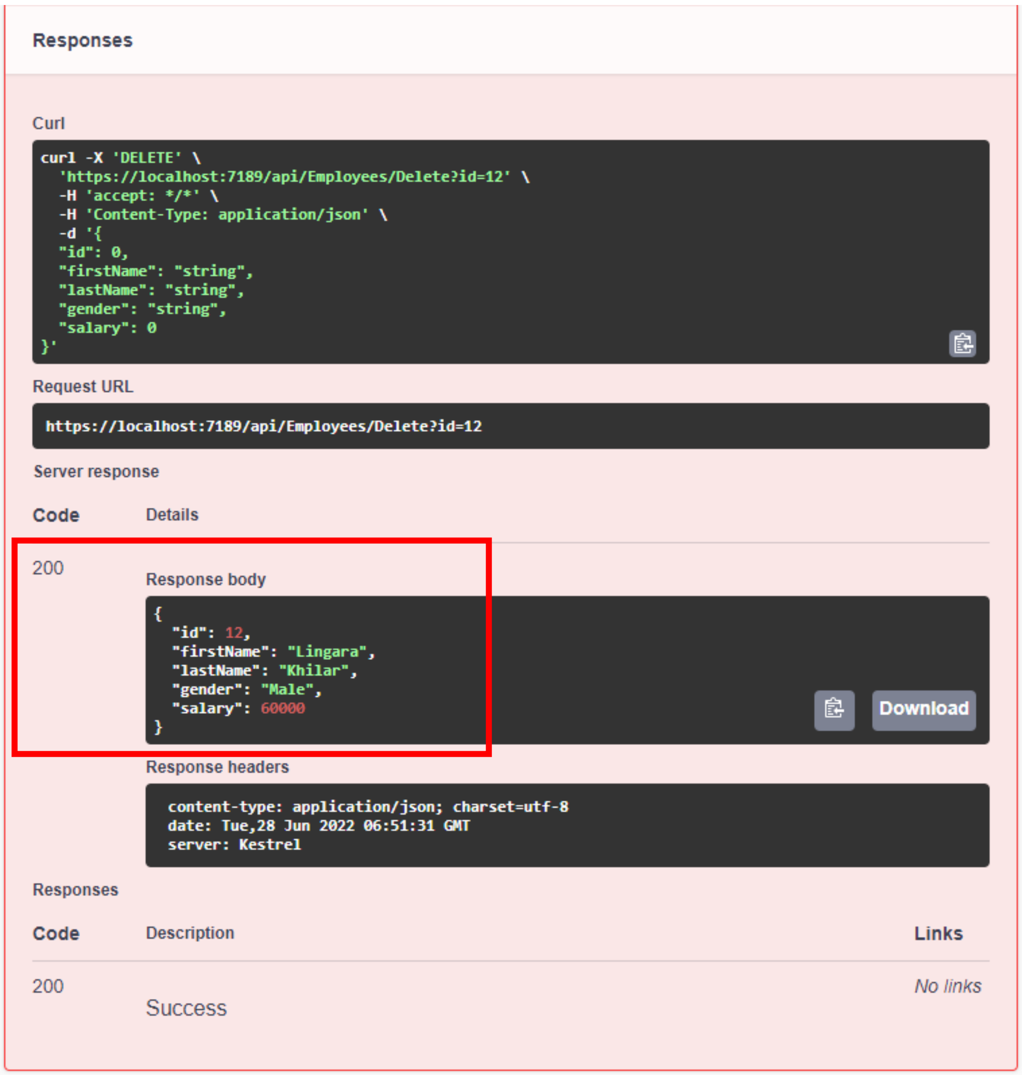
Duration
6 hours


DELETE /api/Employees/{action}

Parameters | Cancel

Name	Description
id integer($int32) (query)	12
action * required string (path)	Delete

27. When the deletion is successful and return status code ‘**200 OK’** indicating that the delete is successful.


Responses

Curl

```
curl -X 'DELETE' \
  'https://localhost:7189/api/Employees/Delete?id=12' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "id": 0,
  "firstName": "string",
  "lastName": "string",
  "gender": "string",
  "salary": 0
}'
```

Request URL

```
https://localhost:7189/api/Employees/Delete?id=12
```

Server response

Code | Details

200

Response body

```
{
  "id": 12,
  "firstName": "Lingara",
  "lastName": "Khilar",
  "gender": "Male",
  "salary": 60000
}
```

Download

Response headers

```
content-type: application/json; charset=utf-8
date: Tue,28 Jun 2022 06:51:31 GMT
server: Kestrel
```

Responses

Code	Description	Links
200	Success	*No links*

- End -