Task 4: Web API CRUD Operation – Update Using PUT

Objectives

- Create an API action method that updates employee data using the `PUT` HTTP verb.
- Use `[FromBody]` to read JSON data sent in the request body.
- Validate the request input and return appropriate HTTP status codes.
- Demonstrate the operation using Swagger and Postman.
- Use hardcoded in-memory data for update operations.

Implementation

Step 1: Update Method in EmployeeController

The following `PUT` action method validates the input ID and updates an existing employee in the list:

```
[HttpPut("{id}")]
public ActionResult<Employee> UpdateEmployee(int id, [FromBody] Employee updated)
{
    if (id <= 0)
        return BadRequest("Invalid employee id");

    var emp = employeeList.FirstOrDefault(e => e.Id == id);
    if (emp == null)
        return BadRequest("Invalid employee id");

    emp.Name = updated.Name;
    emp.Salary = updated.Salary;
    emp.Permanent = updated.Permanent;
    emp.Department = updated.Department;
    emp.Skills = updated.Skills;
    emp.DateOfBirth = updated.DateOfBirth;

    return Ok(emp);
}
```

Explanation

Use of `[FromBody]`:

- `[FromBody]` ensures the input employee data is read from the **HTTP request body** in JSON format and deserialized into the `Employee` object.

Validations in Method:

- If `id <= 0`: Returns `400 Bad Request` with message "Invalid employee id".
- If employee with given ID is **not found** in the hardcoded list: Also returns `400 Bad Request`.
- If found: Updates the matching employee and returns `200 OK` with the updated employee object.

Testing the API

Using Swagger:

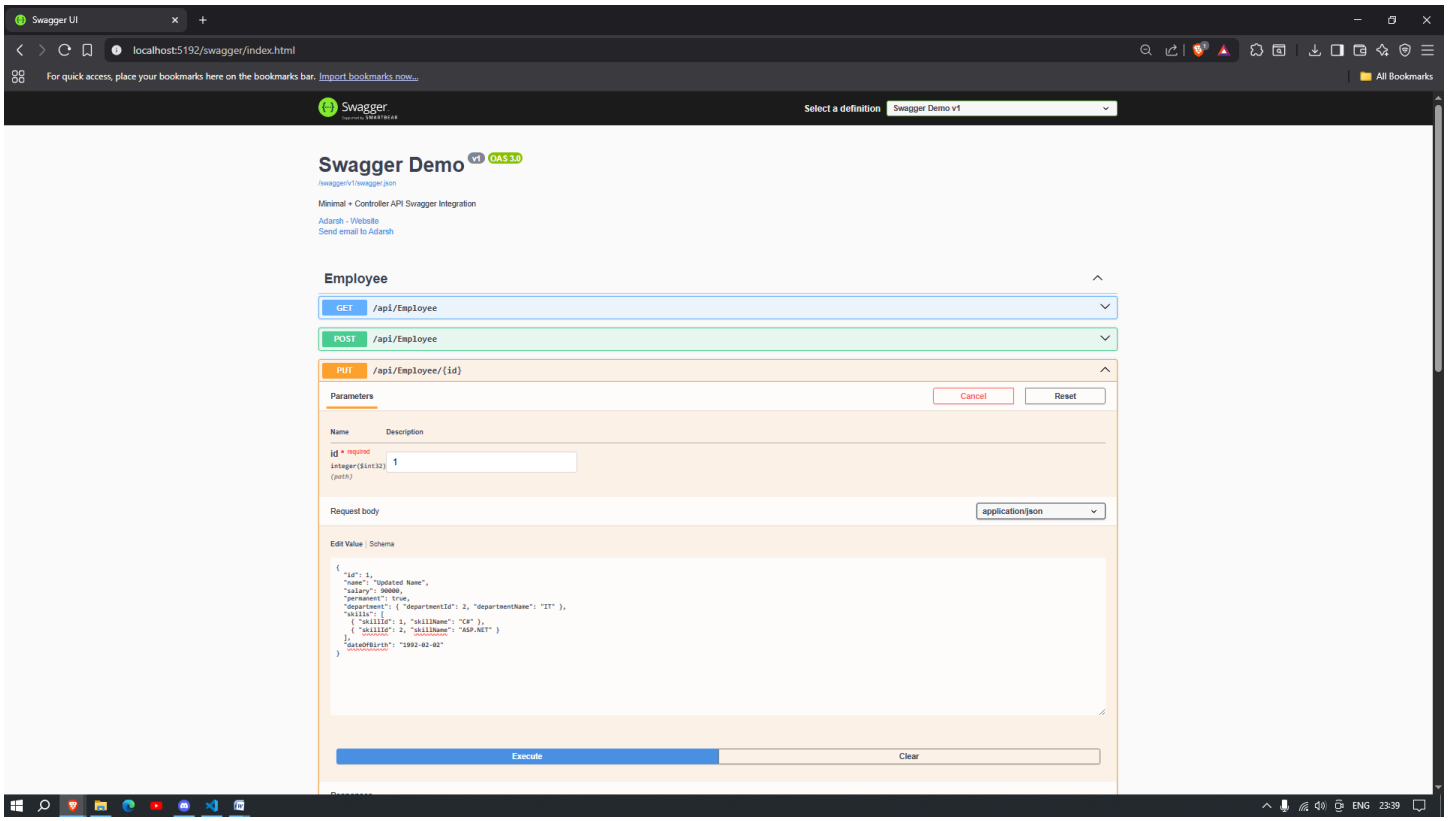
1. Open Swagger UI (`http://localhost:<port>/swagger`)
2. Expand the `PUT /api/Employee/{id}` endpoint
3. Click **Try it out**
4. Enter:
    - A valid employee ID (e.g., `1`)
    - JSON body with updated employee data
5. Click **Execute**
6. Swagger displays:
    - Status Code: `200 OK`
    - Response body: Updated employee data

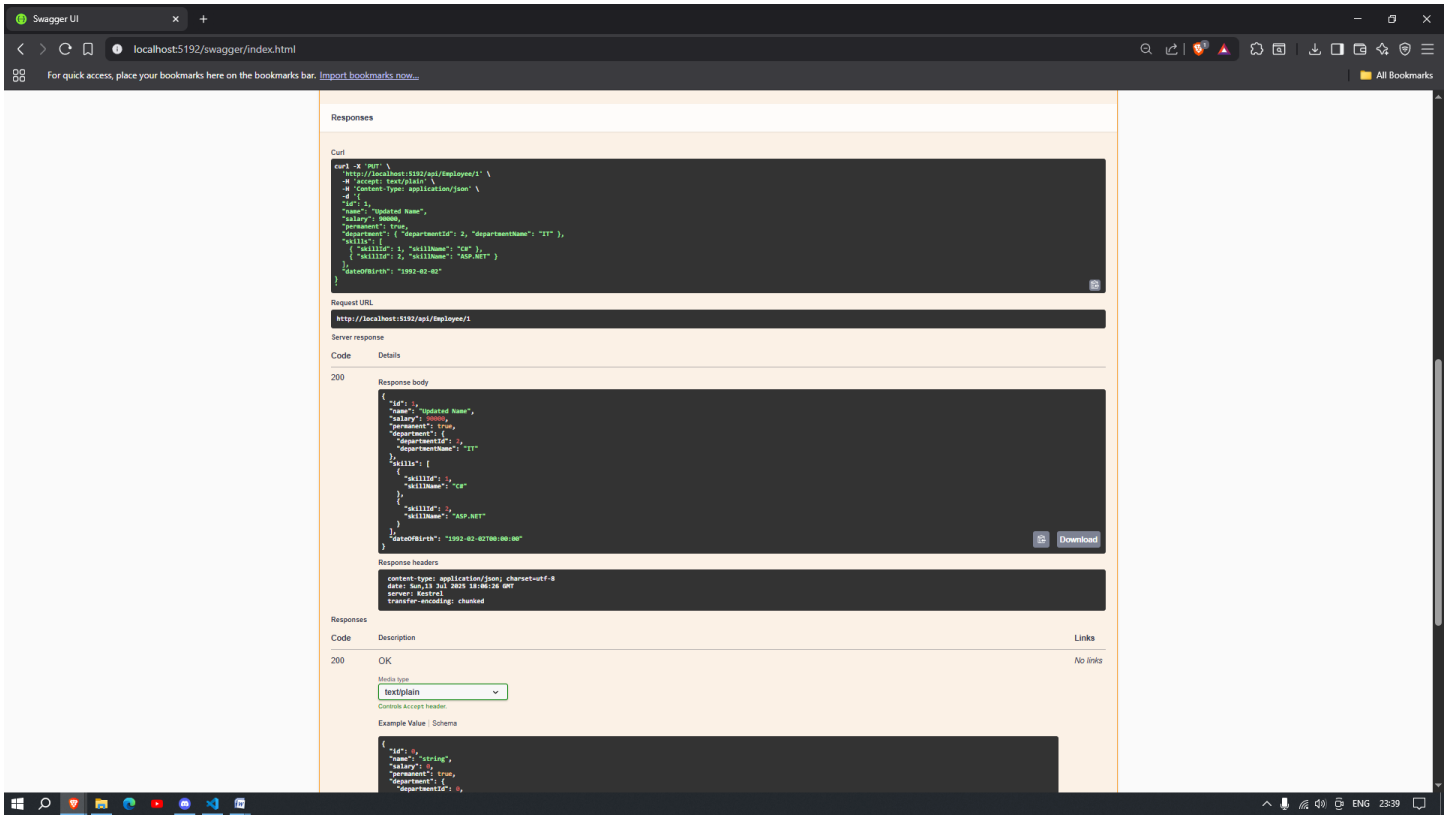
Using Postman:

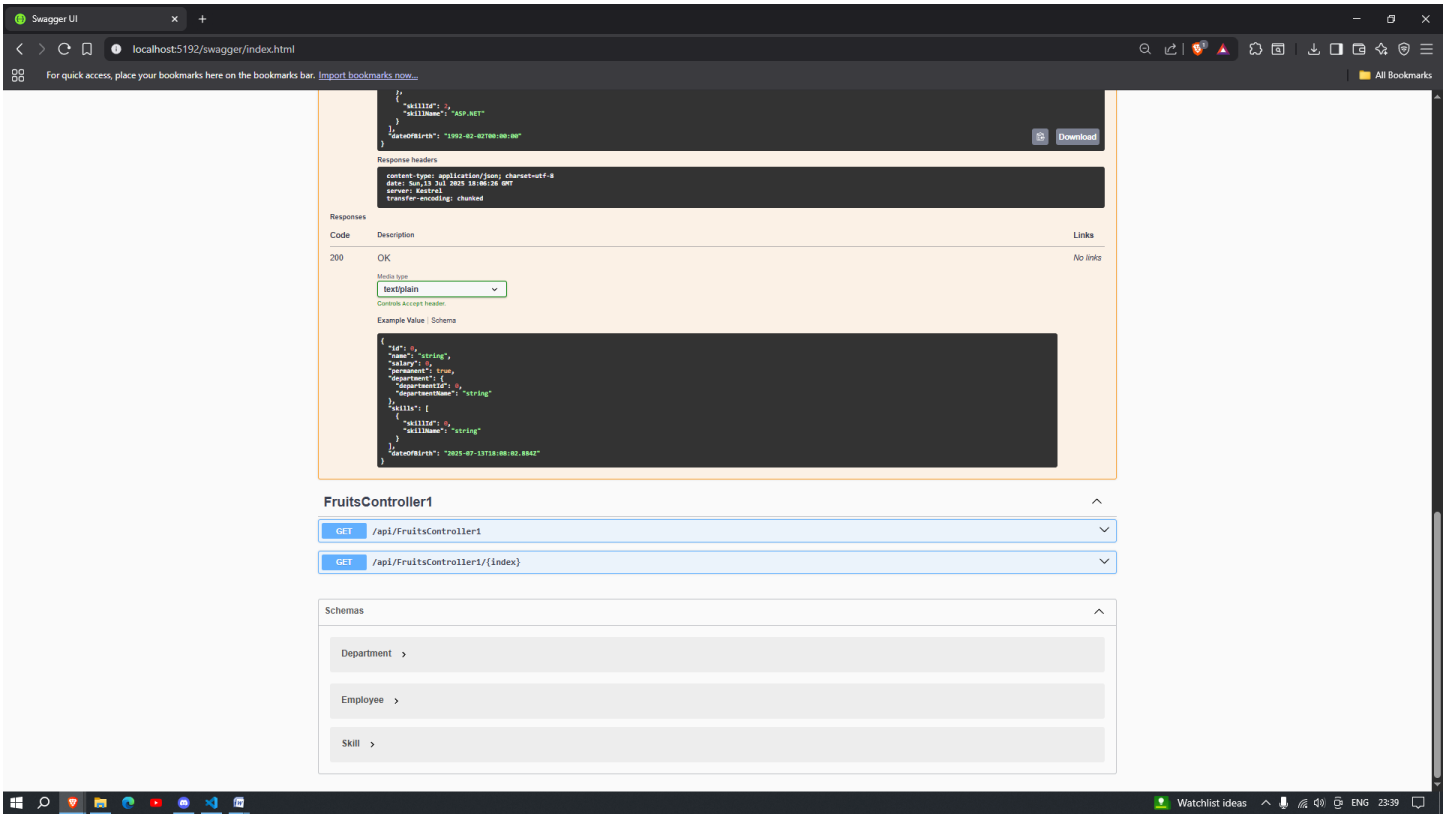
1. Method: `PUT`
2. URL: `http://localhost:<port>/api/Employee/1`
3. Headers:
    - Content-Type: `application/json`
    - Authorization: `Bearer <your_jwt_token>`
4. Body: (raw → JSON)

```
{
  "id": 1,
  "name": "Adarsh Updated",
  "salary": 60000,
  "permanent": true,
  "department": {
    "departmentId": 2,
    "departmentName": "Engineering"
  },
  "skills": [
    { "skillId": 1, "skillName": "C#" },
    { "skillId": 2, "skillName": "SQL" }
  ],
  "dateOfBirth": "1995-01-01"
}
```

Screenshots


Swagger UI
localhost:5192/swagger/index.html
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
All Bookmarks
Swagger
Select a definition
Swagger Demo v1
Swagger Demo
v1
OAS 3.0
/swagger/v1/swagger.json
Minimal + Controller API Swagger Integration
Adarsh - Website
Send email to Adarsh
Employee
GET /api/Employee
POST /api/Employee
PUT /api/Employee/{id}
Parameters
Cancel
Reset
Name
Description
id * required
integer($int32)
(path)
1
Request body
application/json
Edit Value | Schema
{
"id": 1,
"name": "Updated Name",
"salary": 90000,
"permanent": true,
"department": { "departmentId": 2, "departmentName": "IT" },
"skills": [
{ "skillId": 1, "skillName": "C#" },
{ "skillId": 2, "skillName": "ASP.NET" }
],
"dateOfBirth": "1992-02-02"
}
Execute
Clear
ENG
23:39


Swagger UI
localhost:5192/swagger/index.html
For quick access, place your bookmarks here on the bookmarks bar. Import bookmarks now...
All Bookmarks
Responses
Curl
curl -X 'PUT' \
'http://localhost:5192/api/Employee/1' \
-H 'accept: text/plain' \
-H 'Content-Type: application/json' \
-d '{
"id": 1,
"name": "Updated Name",
"salary": 90000,
"permanent": true,
"department": { "departmentId": 2, "departmentName": "IT" },
"skills": [
{ "skillId": 1, "skillName": "C#" },
{ "skillId": 2, "skillName": "ASP.NET" }
],
"dateOfBirth": "1992-02-02"
}
Request URL
http://localhost:5192/api/Employee/1
Server response
Code
Details
200
Response body
{
"id": 1,
"name": "Updated Name",
"salary": 90000,
"permanent": true,
"department": {
"departmentId": 2,
"departmentName": "IT"
},
"skills": [
{
"skillId": 1,
"skillName": "C#"
},
{
"skillId": 2,
"skillName": "ASP.NET"
}
],
"dateOfBirth": "1992-02-02T00:00:00"
}
Download
Response headers
content-type: application/json; charset=utf-8
date: Sun,13 Jul 2025 18:06:26 GMT
server: Kestrel
transfer-encoding: chunked
Responses
Code
Description
Links
200
OK
No links
Media type
text/plain
Controls Accept header.
Example Value | Schema
{
"id": 0,
"name": "string",
"salary": 0,
"permanent": true,
"department": {
"departmentId": 0,
ENG
23:39



Conclusion

This task demonstrates the use of the `PUT` HTTP verb to update data in Web API using in-memory storage. It highlights input validation, use of `[FromBody]` for reading JSON data, and standard RESTful practices for error handling and response codes.