**1. Web Api CRUD operation**

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

Employee information has to be updated based on the user input. Use Swagger tool to invoke the action method mapped with Http PUT action verb to update an employee data.

Modify the action method to return Employee data thru ActionResult.

Check if the id value is lesser than or equal to 0. If true, throw BadRequest action result with the message ‘Invalid employee id’

If the value is greater than 0 but not available in the list of employee ids that is there in the hardcoded list of employees, throw BadRequest action result with the same message as stated above.

If the id value is valid, use the JSON data from the input body and update the hardcoded list. Filter the employee list data for the input id and return that as the output.

**Solution:**

* Based on problem 3, I modified the code in that files to add the put and get methods in swagger and make it update the existing user data.
* I updated the [employee.cs](http://employee.cs) and [EmployeeController.cs](http://employeecontroller.cs)

[employee.cs](http://employee.cs)

namespace EmployeeWebApi.Models

{

public class Employee

{

public int Id { get; set; }

public string Name { get; set; }

public string Department { get; set; }

public double Salary { get; set; }

}

}

[EmployeeController.cs](http://employeecontroller.cs)

using Microsoft.AspNetCore.Mvc;

using EmployeeWebApi.Models;

namespace EmployeeWebApi.Controllers

{

[Route("api/[controller]")]

[ApiController]

public class EmployeeController : ControllerBase

{

private static List<Employee> employees = new List<Employee>

{

new Employee { Id = 1, Name = "John", Department = "HR", Salary = 50000 },

new Employee { Id = 2, Name = "Jane", Department = "IT", Salary = 60000 },

new Employee { Id = 3, Name = "Bob", Department = "Finance", Salary = 55000 }

};

[HttpPut("{id}")]

public ActionResult<Employee> UpdateEmployee(int id, [FromBody] Employee updatedEmployee)

{

if (id <= 0)

{

return BadRequest("Invalid employee id");

}

var existingEmployee = employees.FirstOrDefault(e => e.Id == id);

if (existingEmployee == null)

{

return BadRequest("Invalid employee id");

}

existingEmployee.Name = updatedEmployee.Name;

existingEmployee.Department = updatedEmployee.Department;

existingEmployee.Salary = updatedEmployee.Salary;

return Ok(existingEmployee);

}

[HttpGet]

public ActionResult<List<Employee>> GetAllEmployees()

{

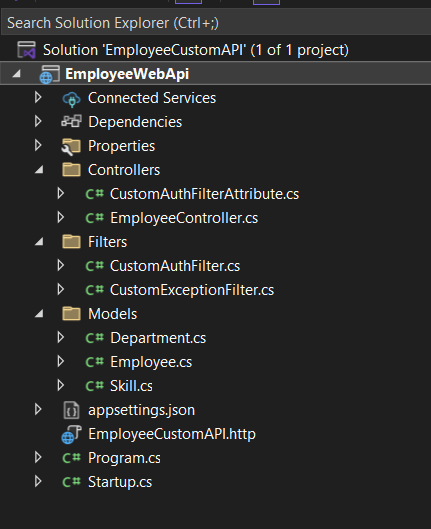
return Ok(employees);

}

}

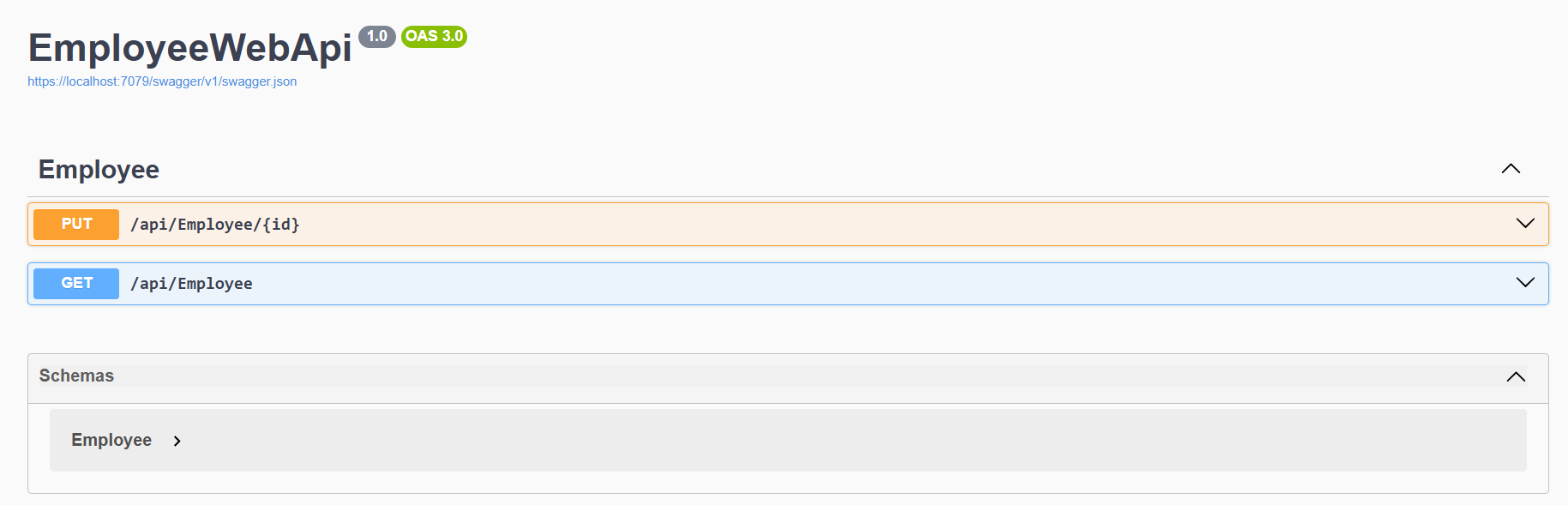
}

**File structure:**



For this solution I used the exercise 3 project files.

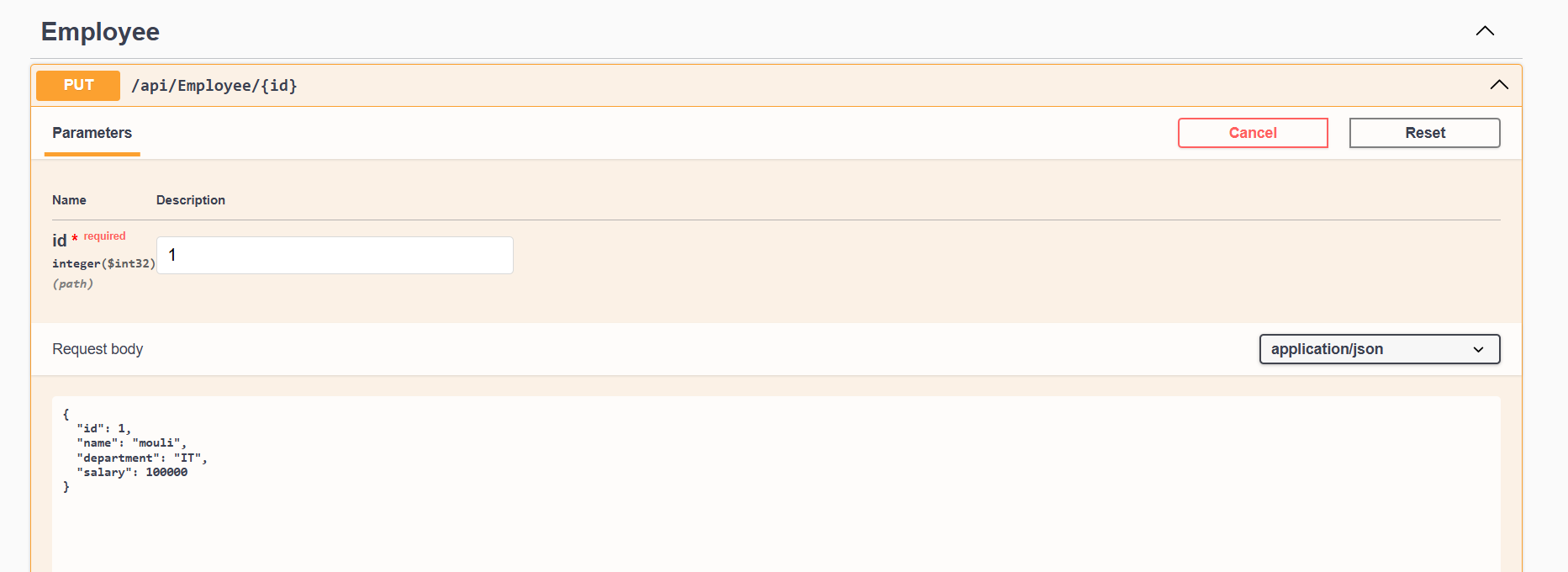
**Result:**

****

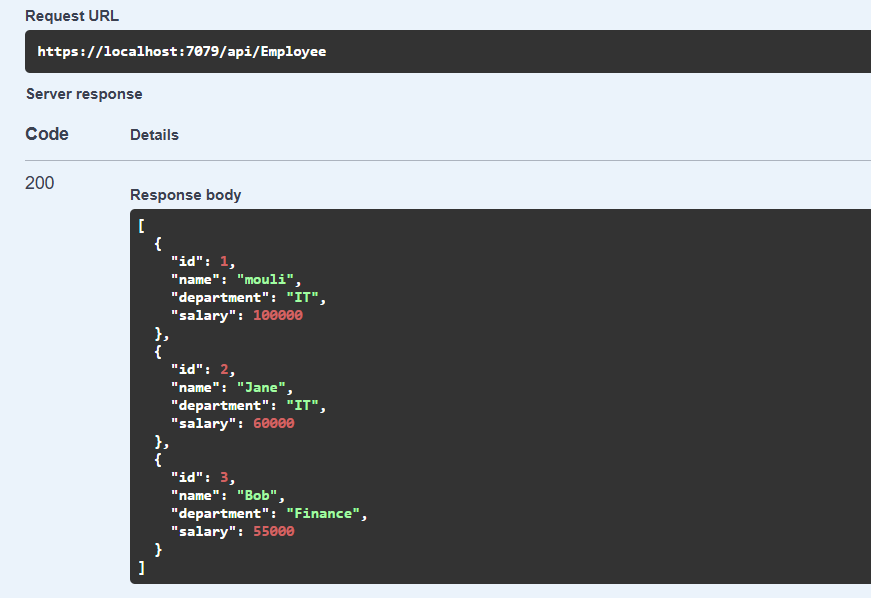
The actual data before i put the updated data:



After updating using put method (updating user 1 data by using application/Json)



And after updating the salary of id 1 the data is updated through PUT



If i entered the employed ID as 5 is out of range or not existing one it shows the 400 error invalid

