# Lab 4 - Web API CRUD Operation

## Objectives

• Demonstrate creation of an Action method to perform Create, Update, and Delete operations.  
• Use [FromBody] attribute to read the request body as a model object.  
• Use hardcoded employee data to simulate update/delete.  
• Test using Swagger and Postman.

## 1. Model Class Creation

Inside the Models folder, create the following classes:

* Employee.cs, Department.cs, Skill.cs

public class Department {  
 public int Id { get; set; }  
 public string Name { get; set; }  
}  
  
public class Skill {  
 public int Id { get; set; }  
 public string Name { get; set; }  
}  
  
public class Employee {  
 public int Id { get; set; }  
 public string Name { get; set; }  
 public int Salary { get; set; }  
 public bool Permanent { get; set; }  
 public Department Department { get; set; }  
 public List<Skill> Skills { get; set; }  
 public DateTime DateOfBirth { get; set; }  
}

## 2. Controller Class

Create EmployeeController.cs under Controllers folder and paste the following code:

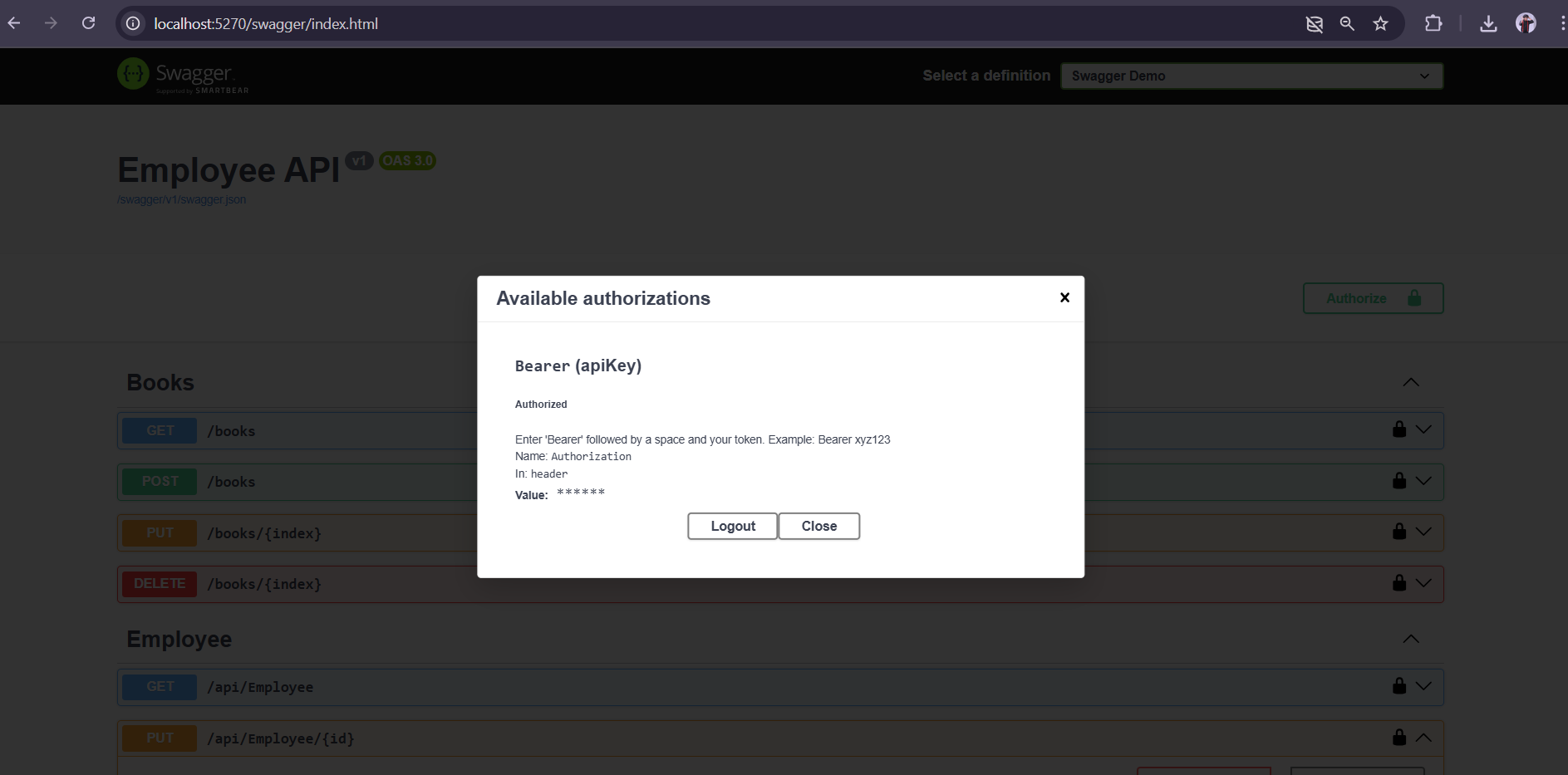
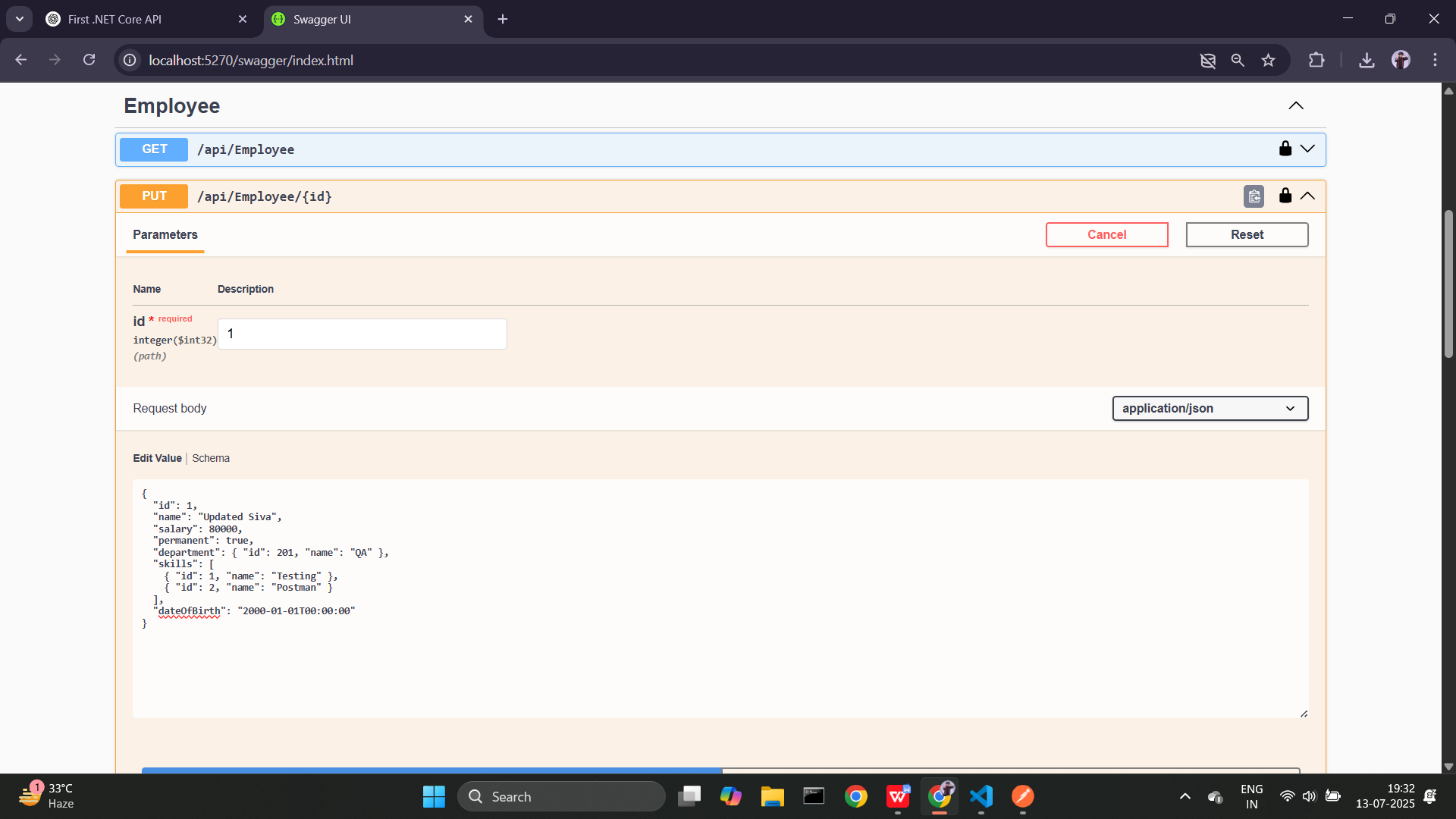
[Route("api/[controller]")]  
[ApiController]  
public class EmployeeController : ControllerBase  
{  
 private static List<Employee> employeeList = new List<Employee>  
 {  
 new Employee  
 {  
 Id = 1,  
 Name = "Siva",  
 Salary = 70000,  
 Permanent = true,  
 Department = new Department { Id = 101, Name = "IT" },  
 Skills = new List<Skill>  
 {  
 new Skill { Id = 1, Name = "C#" },  
 new Skill { Id = 2, Name = "SQL" }  
 },  
 DateOfBirth = new DateTime(2000, 1, 1)  
 }  
 };  
  
 [HttpPut("{id}")]  
 public ActionResult<Employee> UpdateEmployee(int id, [FromBody] Employee emp)  
 {  
 if (id <= 0)  
 return BadRequest("Invalid employee id");  
  
 var existing = employeeList.FirstOrDefault(e => e.Id == id);  
 if (existing == null)  
 return BadRequest("Invalid employee id");  
  
 existing.Name = emp.Name;  
 existing.Salary = emp.Salary;  
 existing.Permanent = emp.Permanent;  
 existing.Department = emp.Department;  
 existing.Skills = emp.Skills;  
 existing.DateOfBirth = emp.DateOfBirth;  
  
 return Ok(existing);  
 }  
}

## 3. Test with Swagger and Postman

Run the app and navigate to Swagger UI to test the PUT method. Use the following sample JSON in body:

{  
 "id": 1,  
 "name": "Updated Siva",  
 "salary": 80000,  
 "permanent": true,  
 "department": { "id": 201, "name": "QA" },  
 "skills": [  
 { "id": 1, "name": "Testing" },  
 { "id": 2, "name": "Postman" }  
 ],  
 "dateOfBirth": "2000-01-01T00:00:00"  
}

## 4. Screenshots

* 
* 
* 