* **Demonstrate creation of an Action method to perform data create, update & delete operation**
  + **Use FromBody attribute, extract data to custom model class using FromBody attribute, use hardcoded data to update & delete data, Use Swagger and POSTMAN to test**

**Answer-->**

1. Step-by-Step Implementation

### ****Model Class****

// Models/Student.csnamespace DemoWebAPI.Models

{

public class Student

{

public int Id { get; set; }

public string Name { get; set; }

public string Course { get; set; }

}

}

### ****Controller with CRUD Operations****

using Microsoft.AspNetCore.Mvc;using DemoWebAPI.Models;using System.Collections.Generic;using System.Linq;

namespace DemoWebAPI.Controllers

{

[ApiController]

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

public class StudentController : ControllerBase

{

// Simulated in-memory data

private static List<Student> students = new List<Student>

{

new Student { Id = 1, Name = "Alice", Course = "CS" },

new Student { Id = 2, Name = "Bob", Course = "Math" }

};

// CREATE - POST

[HttpPost]

public IActionResult CreateStudent([FromBody] Student student)

{

student.Id = students.Max(s => s.Id) + 1;

students.Add(student);

return CreatedAtAction(nameof(GetStudentById), new { id = student.Id }, student);

}

// READ by ID - GET

[HttpGet("{id}")]

public IActionResult GetStudentById(int id)

{

var student = students.FirstOrDefault(s => s.Id == id);

if (student == null) return NotFound();

return Ok(student);

}

// UPDATE - PUT

[HttpPut("{id}")]

public IActionResult UpdateStudent(int id, [FromBody] Student updatedStudent)

{

var existing = students.FirstOrDefault(s => s.Id == id);

if (existing == null) return NotFound();

existing.Name = updatedStudent.Name;

existing.Course = updatedStudent.Course;

return Ok(existing);

}

// DELETE - DELETE

[HttpDelete("{id}")]

public IActionResult DeleteStudent(int id)

{

var student = students.FirstOrDefault(s => s.Id == id);

if (student == null) return NotFound();

students.Remove(student);

return NoContent(); // 204

}

}

}

## 2. Testing with Swagger

### Steps:

* 1. Run the API → Swagger UI auto-opens (if enabled in Startup.cs)
  2. Use the **POST**, **PUT**, and **DELETE** sections:
  3. Select POST /api/student → Fill JSON body → Execute
  4. Use the Id from POST response to test PUT and DELETE

### Sample JSON Body:

{

"name": "John",

"course": "Physics"

}

## 3. Testing with Postman

### Create Request:

**POST** → URL: http://localhost:{port}/api/student

**Headers**: Content-Type: application/json

**Body**: (raw JSON)

{

"name": "John",

"course": "Physics"

}

**PUT**: Use the Id from response and update the record.

{

"name": "Johnny",

"course": "Astrophysics"

}

**DELETE**: Send DELETE request to /api/student/3

**Output:**

