**WEEK – 4 Assignments**

**ASP. NET Core 8. 0 Web API**

**Mandatory hands-on :-**

**4. WebApi\_Handson :**

**Objectives:**

· Demonstrate creation of an Action method to perform data create, update & delete operation

o 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.

**1. Web Api CRUD operation :**

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

| **namespace EmployeeApiDemo.Models {  public class Employee  {  public int Id { get; set; }  public string Name { get; set; }  public int Age { get; set; }  public string Position { get; set; }  } }** |
| --- |

[**Program.cs**](http://program.cs) **:**

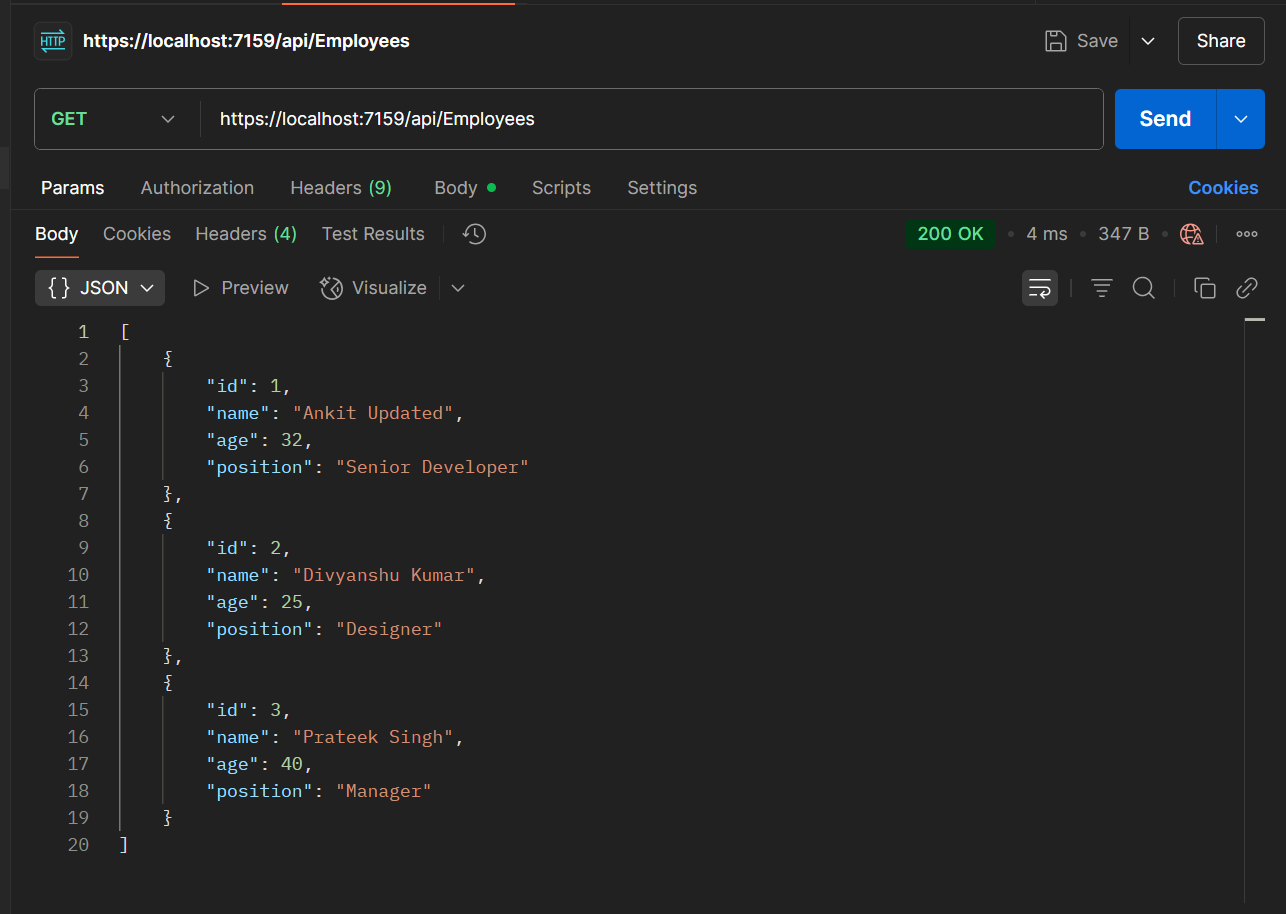
| **var builder = WebApplication.CreateBuilder(args);  // Add services to the container.  builder.Services.AddControllers(); // Learn more about configuring Swagger/OpenAPI at https://aka.ms/aspnetcore/swashbuckle builder.Services.AddEndpointsApiExplorer(); builder.Services.AddSwaggerGen();  var app = builder.Build();  // Configure the HTTP request pipeline. if (app.Environment.IsDevelopment()) {  app.UseSwagger();  app.UseSwaggerUI(); }  app.UseHttpsRedirection();  app.UseAuthorization();  app.MapControllers();  app.Run();** |
| --- |

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

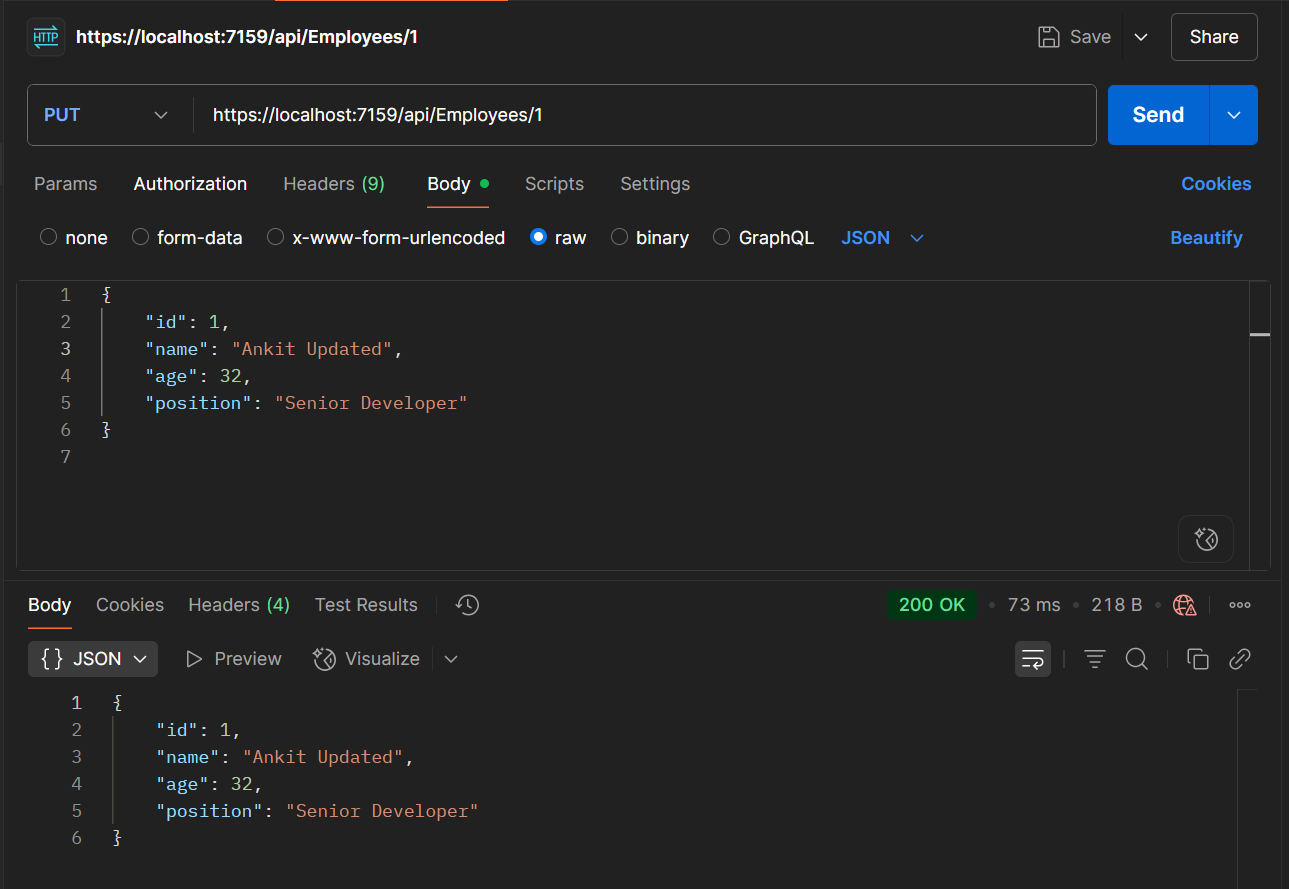
| **using Microsoft.AspNetCore.Mvc; using EmployeeApiDemo.Models; using System.Collections.Generic; using System.Linq;  namespace EmployeeApiDemo.Controllers {  [ApiController]  [Route("api/[controller]")]  public class EmployeesController : ControllerBase  {  private static List<Employee> employees = new List<Employee>  {  new Employee { Id = 1, Name = "Ankit Sahu", Age = 30, Position = "Developer" },  new Employee { Id = 2, Name = "Divyanshu Kumar", Age = 25, Position = "Designer" },  new Employee { Id = 3, Name = "Prateek Singh", Age = 40, Position = "Manager" }  };   // GET: api/Employees  [HttpGet]  public ActionResult<IEnumerable<Employee>> GetAllEmployees()  {  return Ok(employees);  }   // GET: api/Employees/{id}  [HttpGet("{id}")]  public ActionResult<Employee> GetEmployeeById(int id)  {  var employee = employees.FirstOrDefault(e => e.Id == id);  if (employee == null)  {  return NotFound("Employee not found");  }  return Ok(employee);  }   // PUT: api/Employees/{id}  [HttpPut("{id}")]  public ActionResult<Employee> UpdateEmployee(int id, [FromBody] Employee updatedEmployee)  {  if (id <= 0)  {  return BadRequest("Invalid employee id");  }   var employee = employees.FirstOrDefault(e => e.Id == id);  if (employee == null)  {  return BadRequest("Invalid employee id");  }   employee.Name = updatedEmployee.Name;  employee.Age = updatedEmployee.Age;  employee.Position = updatedEmployee.Position;   return Ok(employee);  }  } }** |
| --- |

**OUTPUT (POSTMAN) :**

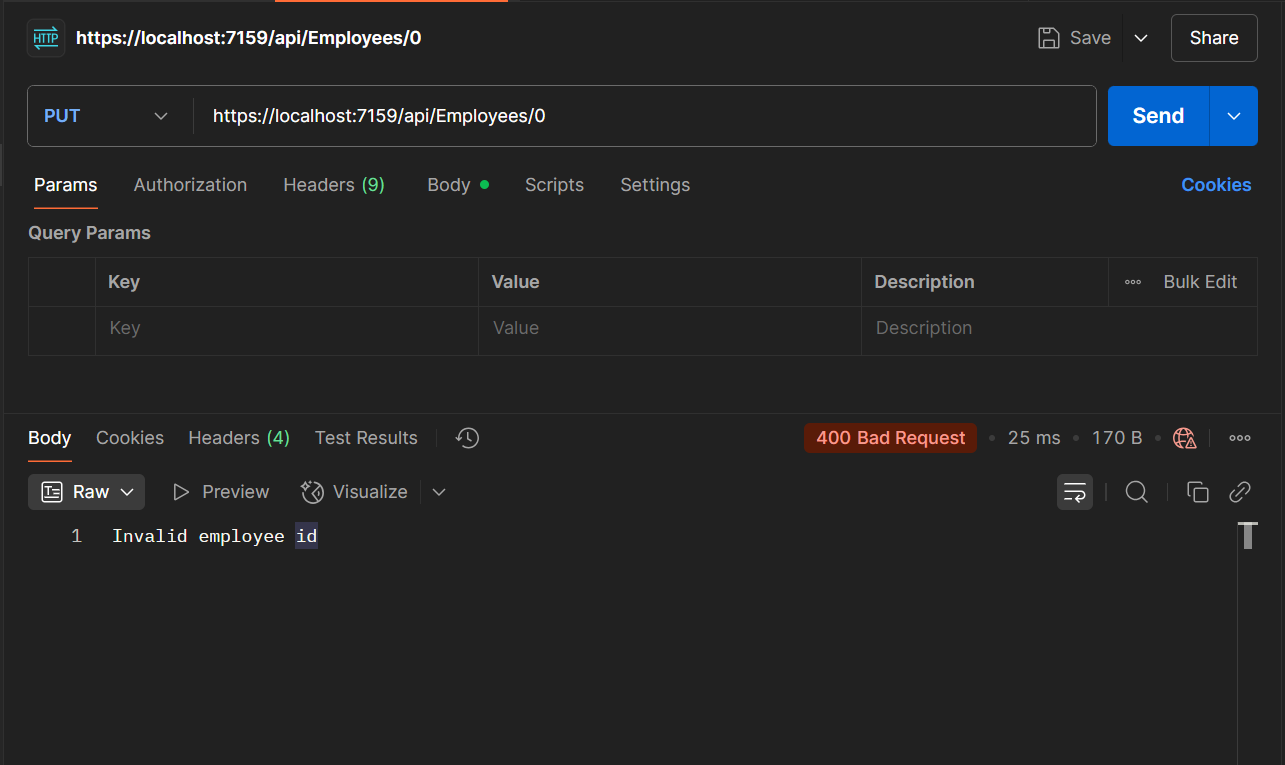
**GET : api/Employees :**

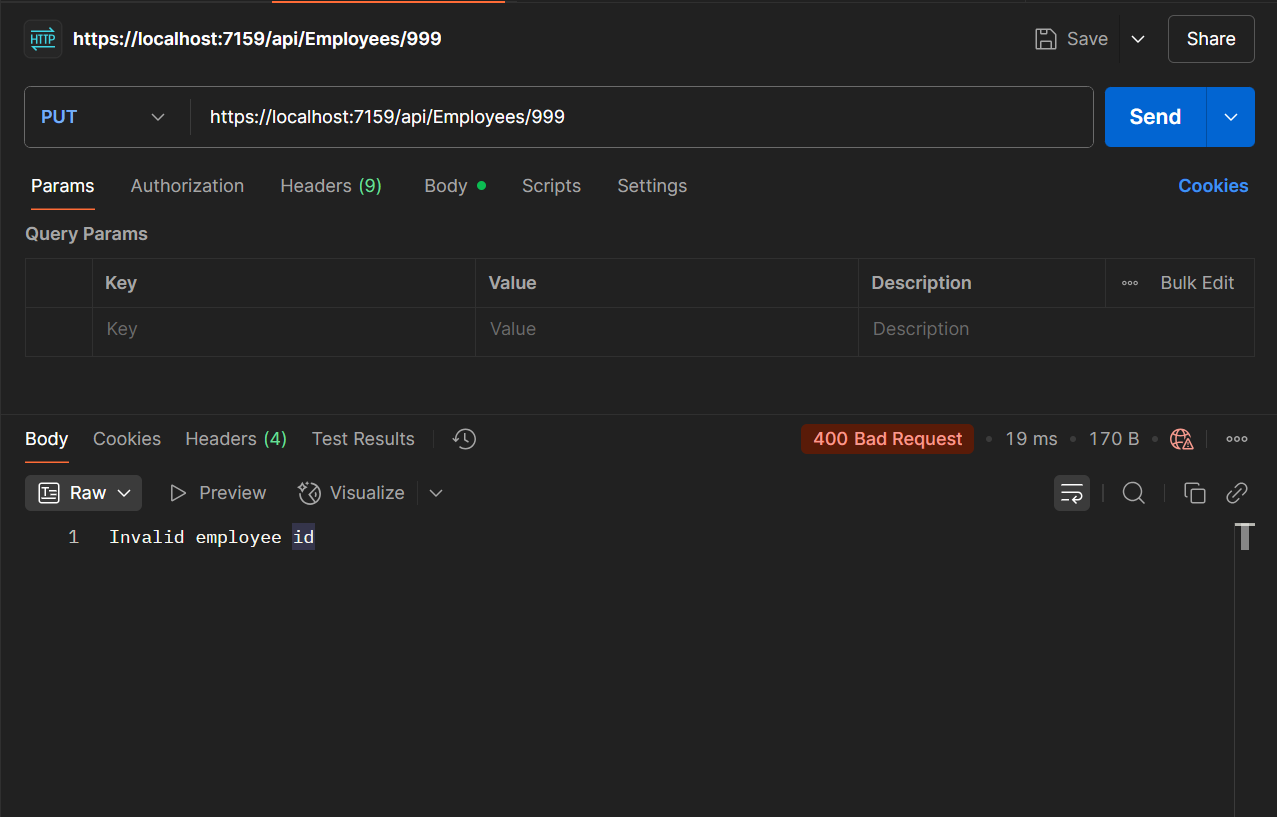
****

**PUT : api/Employees/1 :**

****

**BadRequest :**

****

****