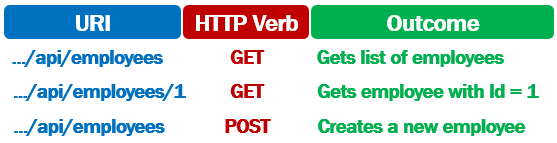
Post in ASP.NET Core REST API

In this video we will discuss, how to create a new item, i.e implement **http post in ASP.NET Core REST API**.

To create a new item, issue an HTTP POST request to the URI /api/employees. Posting to the collection /api/employees makes sense because, to the collection of employees we want to add a new employee.



ASP.NET Core REST API - HTTP POST Example

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

[ApiController]

public class EmployeesController : ControllerBase

{

private readonly IEmployeeRepository employeeRepository;

public EmployeesController(IEmployeeRepository employeeRepository)

{

this.employeeRepository = employeeRepository;

}

[HttpPost]

public async Task<ActionResult<Employee>> CreateEmployee(Employee employee)

{

try

{

if (employee == null)

return BadRequest();

var createdEmployee = await employeeRepository.AddEmployee(employee);

return CreatedAtAction(nameof(GetEmployee),

new { id = createdEmployee.EmployeeId }, createdEmployee);

}

catch (Exception)

{

return StatusCode(StatusCodes.Status500InternalServerError,

"Error creating new employee record");

}

}

}

Copy

Code Explanation

It is the POST request that is used to create a new resource, in our case a new employee. This is the reason CreateEmployee() method is decorated with the HttpPost attribute.

EmployeesController class is decorated with ApiController attribute. This attribute allows the data from the request to be mapped to the employee parameter on CreateEmployee() method.

Either this ApiController attribute is required or the method parameter must be decorated with [FromBody] attribute. Otherwise, model binding will not work as expected and the employee data from the request will not be mapped to the employee parameter on the CreateEmployee method.

[HttpPost]

public async Task<IActionResult> CreateEmployee([FromBody]Employee employee)

{

}

Copy

The injected EmployeeRepository instance adds the new employee to the SQL Server database.

var createdEmployee = await employeeRepository.AddEmployee(employee);

Copy

When a new resource is created the following 3 things usually happen

1. Return the http status code 201 to indicate that the resource is successfully created.
2. Return the newly created resource. In our case, the newly created employee.
3. Add a Location header to the response. The Location header specifies the URI of the newly created employee object.

CreatedAtAction method helps us achieve all the above three things. We are using the nameof operator instead of including the method name (GetEmployee) in a string. This is a good a practice, because if we later change the name of the method and forget to change it here, the compiler will generate an error.

return CreatedAtAction(nameof(GetEmployee),

new { id = createdEmployee.EmployeeId }, createdEmployee);

Copy

In the request body, if we do not include a value for a specific property of the employee. That property will be set to null or default value depending on the datatype.

In the request body, if we include a property that does not exist on the employee object, it will be lost. This is the behaviour we want. If the request includes unnecessary or additional data, simply ignore it.

Complete REST API Controller Code

using EmployeeManagement.Api.Models;

using EmployeeManagement.Models;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using System;

using System.Threading.Tasks;

namespace EmployeeManagement.Api.Controllers

{

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

[ApiController]

public class EmployeesController : ControllerBase

{

private readonly IEmployeeRepository employeeRepository;

public EmployeesController(IEmployeeRepository employeeRepository)

{

this.employeeRepository = employeeRepository;

}

[HttpGet]

public async Task<ActionResult> GetEmployees()

{

try

{

return Ok(await employeeRepository.GetEmployees());

}

catch (Exception)

{

return StatusCode(StatusCodes.Status500InternalServerError,

"Error retrieving data from the database");

}

}

[HttpGet("{id:int}")]

public async Task<ActionResult<Employee>> GetEmployee(int id)

{

try

{

var result = await employeeRepository.GetEmployee(id);

if (result == null)

{

return NotFound();

}

return result;

}

catch (Exception)

{

return StatusCode(StatusCodes.Status500InternalServerError,

"Error retrieving data from the database");

}

}

[HttpPost]

public async Task<ActionResult<Employee>>

CreateEmployee([FromBody]Employee employee)

{

try

{

if (employee == null)

return BadRequest();

var createdEmployee = await employeeRepository.AddEmployee(employee);

return CreatedAtAction(nameof(GetEmployee),

new { id = createdEmployee.EmployeeId }, createdEmployee);

}

catch (Exception)

{

return StatusCode(StatusCodes.Status500InternalServerError,

"Error creating new employee record");

}

}

}

}