**Name: Deepanshu Rathore**

**Superset ID: 6358199**  
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  
  
Week 4  
**2. WebApi\_Handson**Swagger serves as an interactive documentation framework for Web APIs. After installing the Swashbuckle.AspNetCore package, one typically registers Swagger in the service collection by calling AddSwaggerGen. Within that call it is possible to describe the API with metadata such as title, version, contact information and license. In the middleware pipeline the calls to UseSwagger and UseSwaggerUI expose a JSON endpoint and render a web‑based UI respectively. Applying attributes like ProducesResponseType on controller methods enriches the generated documentation by indicating which HTTP status codes and response payloads each action can return.

Postman is a desktop tool that simulates API clients. It has a sidebar of collections into which individual requests can be saved. For each request the developer chooses the HTTP verb, provides the URL, and may add headers such as Authorization or Content‑Type. When a JSON body is required, the raw body tab allows pasting or typing of the payload. After sending a request the response pane displays the status code, headers and the body. The ability to organize requests into collections and to work in multiple tabs makes Postman ideal for iterating against an API during development and testing.

Within ASP NET controllers the [Route] attribute determines how URLs map to action methods. A friendly route name helps clients and tools understand the purpose of an endpoint. Sometimes different overloads of the same HTTP verb are required. The ActionName attribute allows two methods to share the same HTTP method attribute yet have distinct route names or parameter signatures. Together these attributes make the URL design more explicit and maintainable.  
EmployeeController.cs:  
using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace ApiWithSwagger.Controllers

{

[Route("api/emp")]

[ApiController]

public class EmployeeController : ControllerBase

{

[HttpGet]

public ActionResult<IEnumerable<string>> GetEmployees()

{

var employees = new List<string> { "Alice", "Bob", "Charlie" };

return Ok(employees);

}

}

}  
  
Program.cs:

using Microsoft.OpenApi.Models;

namespace ApiWithSwagger

{

public class Program

{

public static void Main(string[] args)

{

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(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo

{

Title = "Swagger Demo",

Version = "v1",

Description = "TBD",

TermsOfService = new Uri("https://example.com/terms"),

Contact = new OpenApiContact

{

Name = "John Doe",

Email = "john@xyzmail.com",

Url = new Uri("https://www.example.com")

},

License = new OpenApiLicense

{

Name = "License Terms",

Url = new Uri("https://www.example.com")

}

});

});

var app = builder.Build();

app.UseSwagger();

app.UseSwaggerUI(c =>

{

c.SwaggerEndpoint("/swagger/v1/swagger.json", "Swagger Demo");

});

if (app.Environment.IsDevelopment())

{

app.UseSwagger();

app.UseSwaggerUI();

}

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();

}

}

}  
**Output:**A screenshot of a computer

AI-generated content may be incorrect.A black and white rectangular object

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.