WebApi Handson 2 Question and Answers

1. Demonstrate Swagger Installation to Web API and Web API Listing on Browser

- NuGet Package Installation
  - Install Swashbuckle.AspNetCore via NuGet Package Manager.
  - This package enables Swagger support in ASP.NET Core Web API.

- Register Swagger in Program.cs

  In builder.Services, add:

  ```
  builder.Services.AddSwaggerGen();
  ```

  In the middleware pipeline, add:

  ```
  app.UseSwagger();
  app.UseSwaggerUI();
  ```

- Web API Listing on Browser
  - Run the application and open https://localhost:[port]/swagger.
  - Swagger UI will list all controllers and API methods available.
- Usage of [ProducesResponseType]

  This attribute helps Swagger document the expected HTTP status codes.

  Example:

  ```
  [ProducesResponseType(200)]

  [ProducesResponseType(400)]

  public IActionResult GetEmployee() { ... }
  ```

2. Demonstrate the Usage of Postman Tool to Hit Web API Methods

- Creating a New Request
  - Open Postman and click New → HTTP Request.
  - Set the method (GET, POST, etc.) and paste the API URL
- Setting Headers

  Under the Headers tab, add:

  Key: Authorization

  Value: Bearer <your_token>

- Sending JSON in Body
  - Under the Body tab, select raw → choose JSON.
  - Paste your JSON payload for POST/PUT requests.

- Choosing Request Type
  - Use the dropdown next to the URL to choose GET, POST, PUT, or DELETE.

- Request Collections
  - Create a new collection to organize API requests.
  - Right-click the collection → Add Request → Save and reuse requests easily.

- Understanding Tabs
  - The center pane in Postman shows the request builder, response status, body, and headers organized in tabs like Body, Headers, Tests, and Console.

3. Demonstrate the Usage of Route and Explain Name Attribute in HTTP Requests

- Route Attribute
  - Used to define custom routing for API endpoints.
    [Route("api/employees")]
- Name Attribute in Routing
  - Provides a user-friendly or reference name for the route.
    [HttpGet("{id}", Name = "GetEmployeeById")]
- Importance of Name Attribute
  - Useful when generating URLs dynamically with CreatedAtAction or Url.Link.
- Using ActionName for Overloading Methods
  - If you want to use multiple methods with the same HTTP verb, use [ActionName] to differentiate them:

```
[HttpGet]

[ActionName("GetAll")]

public IActionResult GetAll()

{ ... }

[HttpGet("{id}")]
[ActionName("GetById")]
public IActionResult
GetById(int id) { ... }
```