**Objectives:**

* Demonstrate Swagger installation to WebAPI and WebAPI listing on browser
  + Nuget package to download Swashbuckle.AspNetCore, Usage of ProducesResponseType to Web API method, AddSwaggerGen, UseSwaggerUI
* Demonstrate the usage of Postman tool to hit WebAPI methods
  + Structure in Postman tool, Headers with Authorization, Body as JSON, Option to choose the type of request, Request collection and how to add a new request in the collection, Tabs in the center pane that corresponds to the request
* Demonstrate the usage of Route and Explain Name attribute in Http requests
  + Importance of user friendly name to action method, Explain the usage of ActionName to have more than 1 method with the same Action verbDemonstrate creation of a simple WebAPI - With Read, Write actions

1. **Web Api using .Net core with Swagger**

Create a .Net core web application with API template. (Use existing application if created). Install Swashbuckle.AspNetCore Nuget package. Post this do the following steps in Startup.cs

* In ConfigureServices method, add the code provided below.

services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new Info

{

Title = "Swagger Demo",

Version = "v1",

Description = "TBD",

TermsOfService = "None",

Contact = new Contact() { Name = "John Doe", Email = "john@xyzmail.com", Url = "www.example.com" },

License = new License() { Name = "License Terms", Url = "www.example.com" }

});

});

* In Configure method, add the code provided below.

app.UseSwagger();

app.UseSwaggerUI(c =>

{

// specifying the Swagger JSON endpoint.

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

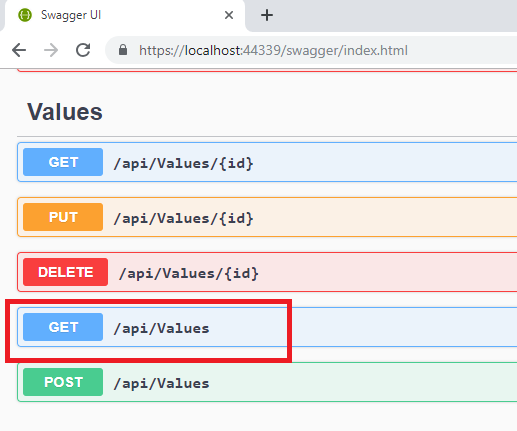
});

Execute the application which will load the default ‘Values’ controller(Settings as per launchSettings.json) GET action method. Change the url to <https://localhost:[port> number]/swagger

Notice the Title, Version, Contact detail provided shown on the top of the page

Notice the Values controller HttpVerb action methods getting listed.

Click the ‘GET’ action verb method(Without the parameter).



It opens a panel which has ‘Try it out’ button. Click that and Click ‘Execute’ button.

1. Use POSTMAN tool, to point to the local Web API that was created with Employee controller. Test the GET action method using POSTMAN.

Verify the output if the List of employees are listed in the ‘Body’ part of the GET window on POSTMAN tool.

Verify the Status on the right side of the output pane on POSTMAN tool.

1. Modify the Controller name in the Route attribute of the Employee controller to ‘Emp’ and check its access thru POSTMAN

Program:

using Microsoft.OpenApi.Models;

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllers();

// Add Swagger generator service

builder.Services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo

{

Title = "Swagger Demo",

Version = "v1",

Description = "Demo project to show Swagger with WebAPI",

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/license")

}

});

});

var app = builder.Build();

// Configure middleware pipeline

// Enable Swagger middleware

app.UseSwagger();

// Enable Swagger UI middleware

app.UseSwaggerUI(c =>

{

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

// Optional: serve Swagger UI at app root

// c.RoutePrefix = "";

});

app.UseHttpsRedirection();

app.UseAuthorization();

app.MapControllers();

app.Run();

OUTPUT:

