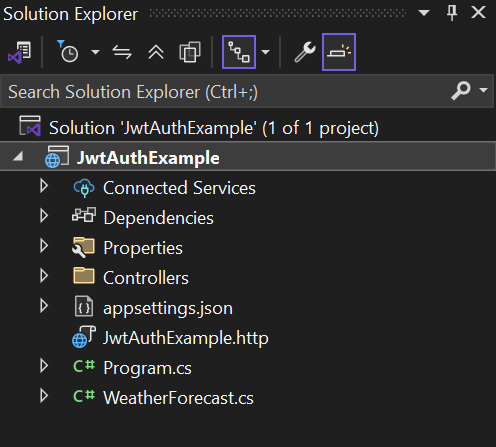
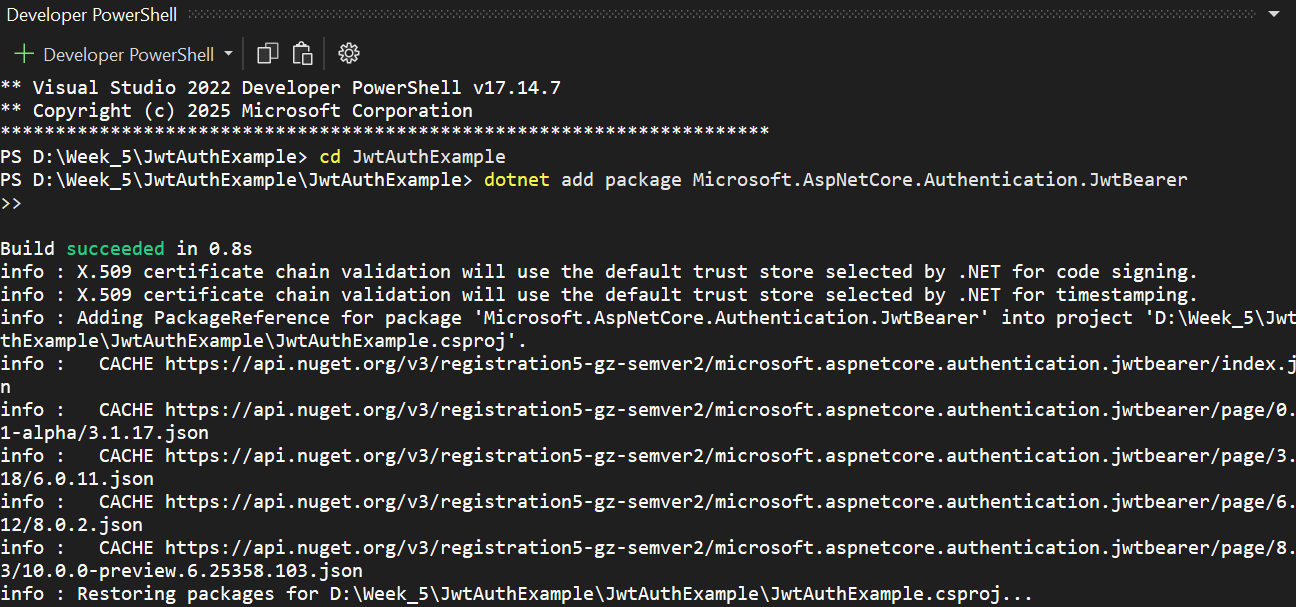
Name:- Question 1: Implement JWT Authentication in ASP.NET Core Web API

Createing a ASP.NET **named as** JwtAuthExample



Installing JWT Package



Open appsettings.json and update the code

{

"Logging": {

"LogLevel": {

"Default": "Information",

"Microsoft.AspNetCore": "Warning"

}

},

"AllowedHosts": "\*",

"Jwt": {

"Key": "ThisIsASuperSecretJwtKeyThatIsLongEnough@123",

"Issuer": "MyAuthServer",

"Audience": "MyApiUsers",

"DurationInMinutes": 60

}

}

Updated code : Program.cs

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.IdentityModel.Tokens;

using Microsoft.OpenApi.Models;

using System.Text;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

var jwtKey = builder.Configuration["Jwt:Key"];

if (string.IsNullOrEmpty(jwtKey))

throw new Exception("JWT key is missing in appsettings.json.");

int byteLength = Encoding.UTF8.GetByteCount(jwtKey);

Console.WriteLine("🔐 JWT Key Byte Length: " + byteLength);

if (byteLength < 32)

throw new Exception("JWT key must be at least 256 bits (32 ASCII characters) long.");

builder.Services.AddAuthentication("Bearer")

.AddJwtBearer("Bearer", options =>

{

options.TokenValidationParameters = new TokenValidationParameters

{

ValidateIssuer = true,

ValidateAudience = true,

ValidateLifetime = true,

ValidateIssuerSigningKey = true,

ValidIssuer = builder.Configuration["Jwt:Issuer"],

ValidAudience = builder.Configuration["Jwt:Audience"],

IssuerSigningKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(jwtKey!))

};

});

builder.Services.AddAuthorization();

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen(c =>

{

c.SwaggerDoc("v1", new OpenApiInfo { Title = "JWT API", Version = "v1" });

c.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme

{

In = ParameterLocation.Header,

Description = "Enter 'Bearer {your JWT token}'",

Name = "Authorization",

Type = SecuritySchemeType.ApiKey

});

c.AddSecurityRequirement(new OpenApiSecurityRequirement

{

{

new OpenApiSecurityScheme

{

Reference = new OpenApiReference

{

Type = ReferenceType.SecurityScheme,

Id = "Bearer"

}

},

Array.Empty<string>()

}

});

});

var app = builder.Build();

app.UseSwagger();

app.UseSwaggerUI(c =>

{

c.SwaggerEndpoint("/swagger/v1/swagger.json", "JWT API v1");

});

app.UseHttpsRedirection();

app.UseAuthentication();

app.UseAuthorization();

app.MapControllers();

app.Run();

Creating a AuthController.cs in Controllers with add controller with Api with Empty Option

using Microsoft.AspNetCore.Mvc;

using Microsoft.IdentityModel.Tokens;

using System.IdentityModel.Tokens.Jwt;

using System.Security.Claims;

using System.Text;

namespace JwtAuthExample.Controllers

{

[ApiController]

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

public class AuthController : ControllerBase

{

[HttpPost("login")]

public IActionResult Login([FromBody] LoginModel model)

{

if (model.Username == "admin" && model.Password == "password")

{

var token = GenerateJwtToken(model.Username);

return Ok(new { token });

}

return Unauthorized();

}

private string GenerateJwtToken(string username)

{

var claims = new[]

{

new Claim(ClaimTypes.Name, username)

};

var key = new SymmetricSecurityKey(

Encoding.UTF8.GetBytes("ThisIsASuperSecretJwtKeyThatIsLongEnough@123"));

var creds = new SigningCredentials(key, SecurityAlgorithms.HmacSha256);

var token = new JwtSecurityToken(

issuer: "MyAuthServer",

audience: "MyApiUsers",

claims: claims,

expires: DateTime.Now.AddMinutes(60),

signingCredentials: creds);

return new JwtSecurityTokenHandler().WriteToken(token);

}

}

public class LoginModel

{

public string? Username { get; set; }

public string? Password { get; set; }

}

}

In Controllers folder create a clas TestController.cs

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

namespace JwtAuthExample.Controllers

{

[ApiController]

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

public class TestController : ControllerBase

{

[HttpGet("public")]

public IActionResult Public() => Ok("Anyone can access this");

[Authorize]

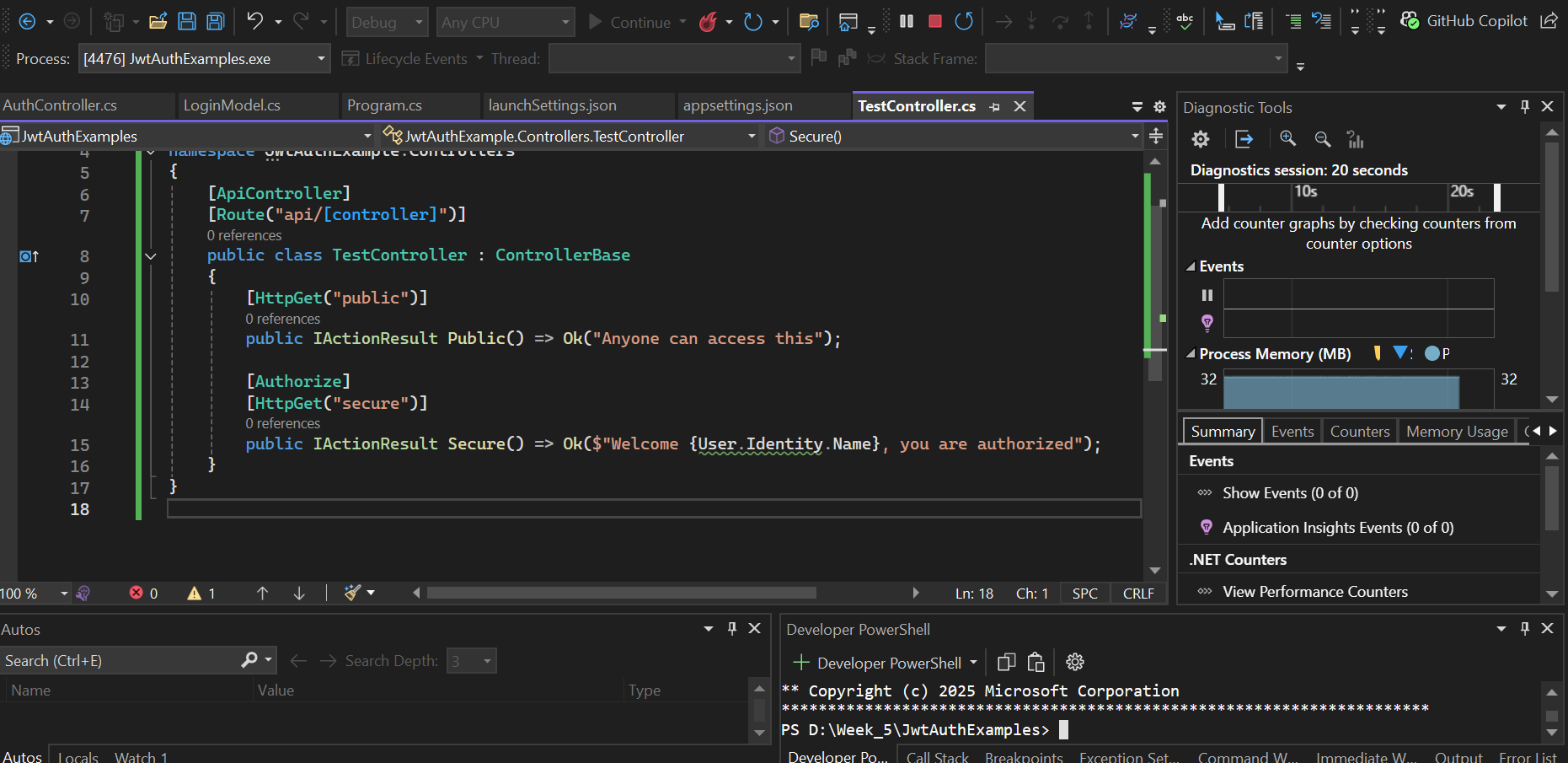
[HttpGet("secure")]

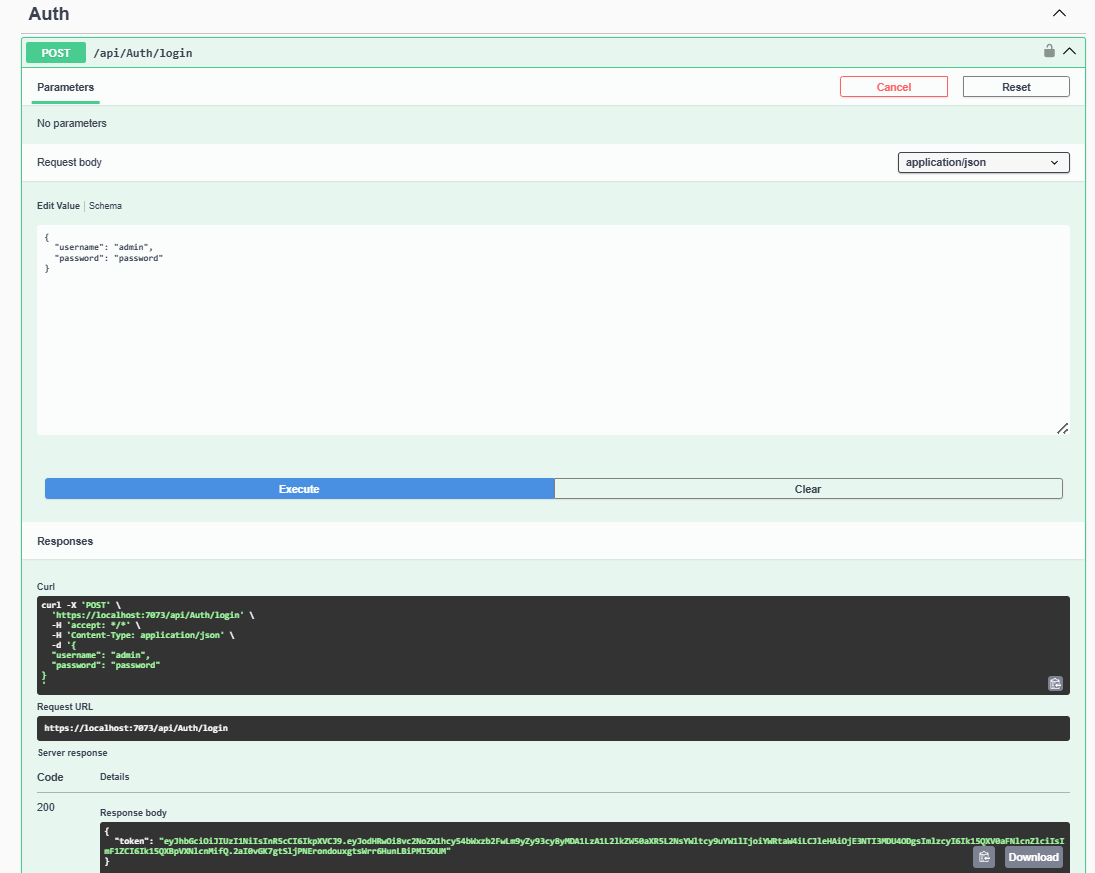
public IActionResult Secure() => Ok($"Welcome {User.Identity.Name}, you are authorized");

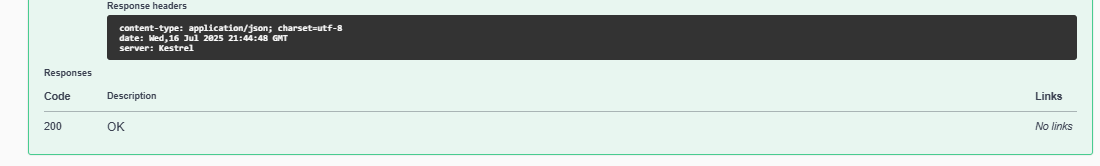
}

}

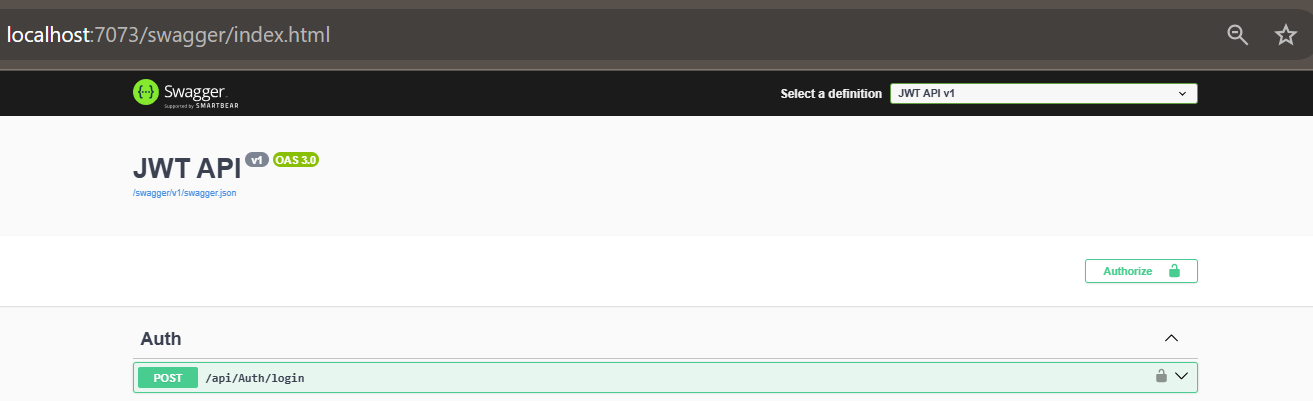
Now Build the Code to check output

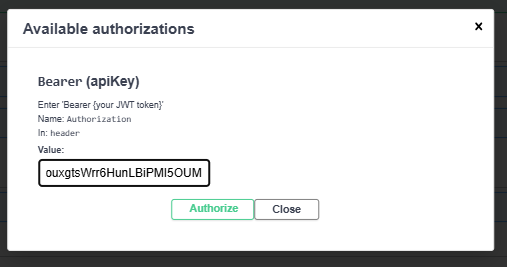
******

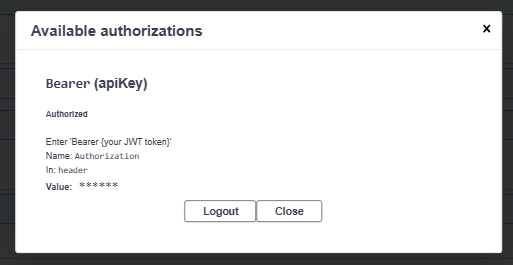


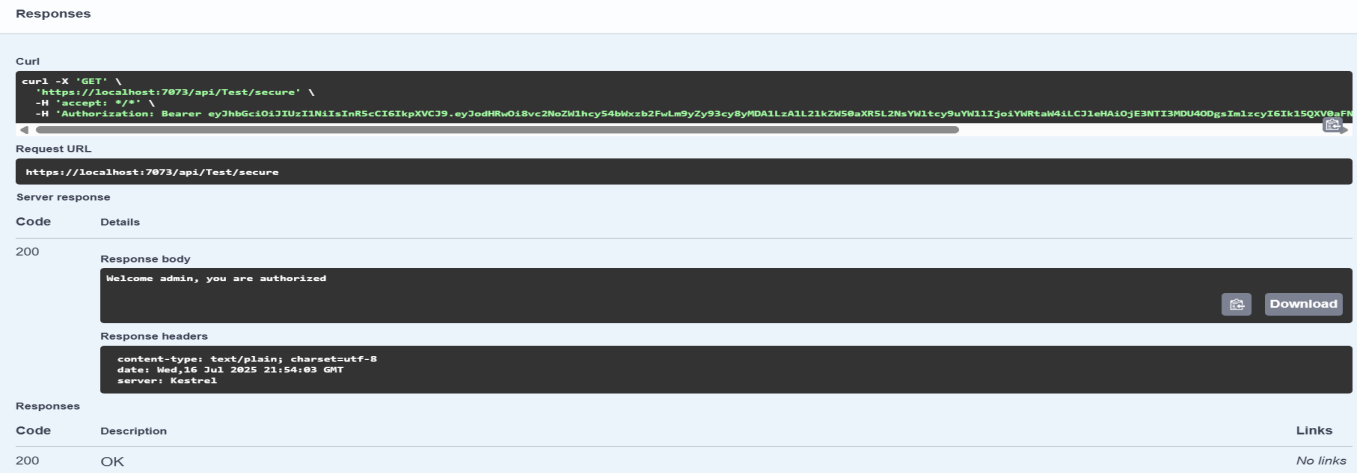


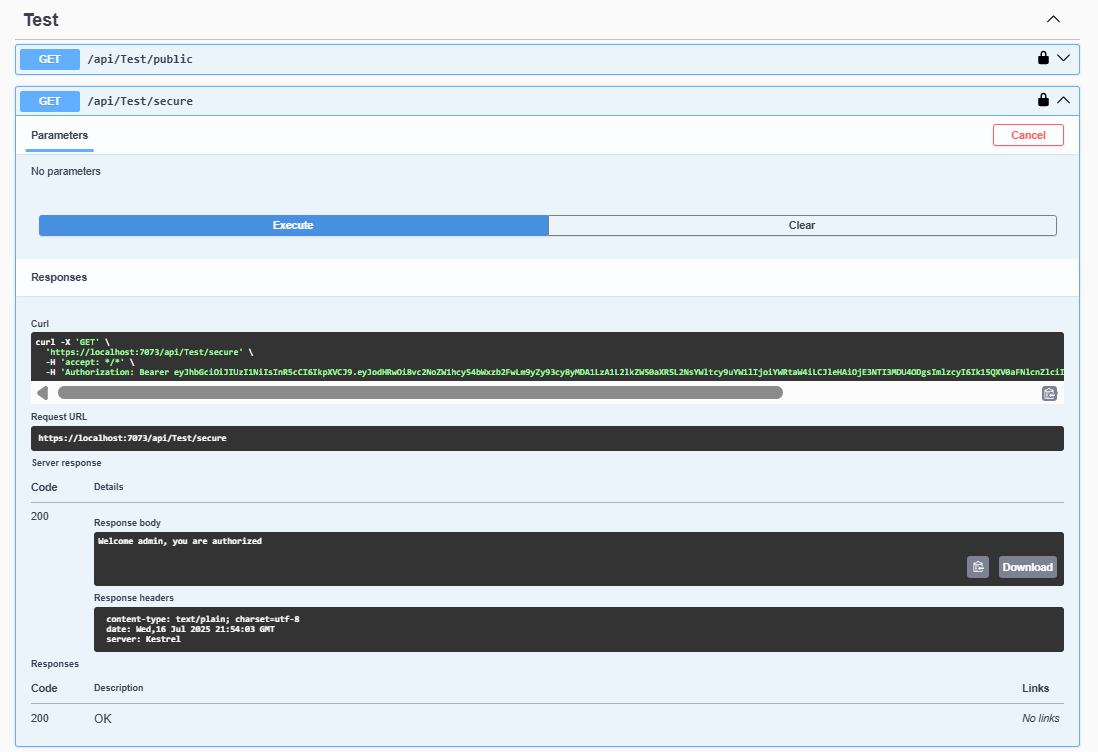
Authorizing Using the Token











Now Checking output Test the Login Endpoint to Get JWT Token in Postman also

For LOGIN (Token Request) use method Post with URL {

"username": "admin",

"password": "password"

}

