# ASP.NET Core Web API

## 1. Program.cs:

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.IdentityModel.Tokens;

using System.Text;

var builder = WebApplication.CreateBuilder(args);

builder.Services.AddControllers();

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(builder.Configuration["Jwt:Key"]!))

        };

        options.Events = new JwtBearerEvents

        {

            OnAuthenticationFailed = context =>

            {

                if (context.Exception is SecurityTokenExpiredException)

                {

                    context.Response.Headers.Append("Token-Expired", "true");

                }

                return Task.CompletedTask;

            }

        };

    });

builder.Services.AddAuthorization();

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var app = builder.Build();

app.UseAuthentication();

app.UseAuthorization();

app.UseSwagger();

app.UseSwaggerUI();

app.MapControllers();

app.Run();

## appsetting.json:

{

  "Jwt": {

    "Key": "ThisIsASecretKeyForJwtToken",

    "Issuer": "MyAuthServer",

    "Audience": "MyApiUsers",

    "DurationInMinutes": 60

  },

  "Logging": {

    "LogLevel": {

      "Default": "Information",

      "Microsoft.AspNetCore": "Warning"

    }

  },

  "AllowedHosts": "\*"

}

### AdminController.cs:

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

[ApiController]

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

public class AdminController : ControllerBase

{

    [HttpGet("dashboard")]

    [Authorize(Roles = "Admin")]

    public IActionResult GetAdminDashboard()

    {

        return Ok("Welcome to the admin dashboard.");

    }

}

### AuthController.cs:

using Microsoft.AspNetCore.Mvc;

using System.IdentityModel.Tokens.Jwt;

using System.Security.Claims;

using System.Text;

using Microsoft.IdentityModel.Tokens;

[ApiController]

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

public class AuthController : ControllerBase

{

    [HttpPost("login")]

    public IActionResult Login([FromBody] LoginModel model)

    {

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

        {

            var claims = new[]

            {

                new Claim(ClaimTypes.Name, model.Username),

                new Claim(ClaimTypes.Role, "Admin")

            };

            var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("ThisIsASecretKeyForJwtToken"));

            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 Ok(new { token = new JwtSecurityTokenHandler().WriteToken(token) });

        }

        return Unauthorized();

    }

}

public class LoginModel

{

    public string Username { get; set; } = string.Empty;

    public string Password { get; set; } = string.Empty;

}

## SecureController.cs:

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

[ApiController]

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

public class SecureController : ControllerBase

{

    [HttpGet("data")]

    [Authorize]

    public IActionResult GetSecureData()

    {

        return Ok("This is protected data.");

    }

}

## Outputs:







