WEEK 6

Controllers/AdminController.cs

using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

namespace JwtAuthDemo.Controllers

{

    [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 Microsoft.IdentityModel.Tokens;

using System.IdentityModel.Tokens.Jwt;

using System.Security.Claims;

using System.Text;

using JwtAuthDemo.Models;

namespace JwtAuthDemo.Controllers

{

    [ApiController]

    [Route("api/[controller]")]

    public class AuthController : ControllerBase

    {

        [HttpPost("login")]

        public IActionResult Login([FromBody] LoginModel model)

        {

            if (IsValidUser(model) && model.Username != null)

            {

                var token = GenerateJwtToken(model.Username);

                return Ok(new { Token = token });

            }

            return Unauthorized("Invalid credentials");

        }

        private bool IsValidUser(LoginModel model)

        {

            // For demo: hardcoded user validation

            return model.Username == "admin" && model.Password == "password";

        }

        private string GenerateJwtToken(string username)

        {

            var claims = new[]

            {

                new Claim(ClaimTypes.Name, username),

                new Claim(ClaimTypes.Role, "Admin") // Q3: Add role

            };

            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 new JwtSecurityTokenHandler().WriteToken(token);

        }

    }

}

DataController.cs  
using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

namespace JwtAuthDemo.Controllers

{

    [Authorize] // This protects the endpoint

    [ApiController]

    [Route("api/[controller]")]

    public class DataController : ControllerBase

    {

        [HttpGet("secure-data")]

        public IActionResult GetSecureData()

        {

            return Ok("🎉 You have accessed protected data!");

        }

        [Authorize(Roles = "Admin")]

        [HttpGet("admin-data")]

        public IActionResult GetAdminData()

        {

            return Ok("🔐 You have accessed admin-only data!");

        }

    }

}

SecureController.cs  
using Microsoft.AspNetCore.Authorization;

using Microsoft.AspNetCore.Mvc;

namespace JwtAuthDemo.Controllers

{

    [ApiController]

    [Route("api/[controller]")]

    public class SecureController : ControllerBase

    {

        [HttpGet("data")]

        [Authorize]

        public IActionResult GetSecureData()

        {

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

        }

    }

}

Model/LoginModel.cs  
namespace JwtAuthDemo.Models

{

    public class LoginModel

    {

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

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

    }

}

