**Hands-On Exercises: Authentication and Authorization in ASP.NET Core**

**Web API Microservices**

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

**Scenario:**

**You are building a microservice that requires secure login. You need to implement JWT**

**based authentication.**

**Steps:**

**1. Create a new ASP.NET Core Web API project.**

**2. Add a `User` model and a login endpoint.**

**3. Generate a JWT token upon successful login.**

**4. Secure an endpoint using `[Authorize]`.**

**Program.cs**

using Microsoft.AspNetCore.Authentication.JwtBearer;

using Microsoft.IdentityModel.Tokens;

using System.Text;

var builder = WebApplication.CreateBuilder(args);

// Add services

builder.Services.AddControllers();

builder.Services.AddEndpointsApiExplorer();

builder.Services.AddSwaggerGen();

var jwtKey = builder.Configuration["Jwt:Key"] ?? "THIS\_IS\_A\_SUPER\_SECRET\_KEY\_1234567890!"; // Fallback

builder.Services.AddAuthentication(JwtBearerDefaults.AuthenticationScheme)

.AddJwtBearer(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))

};

});

var app = builder.Build();

// Middleware

app.UseSwagger();

app.UseSwaggerUI();

app.UseAuthentication();

app.UseAuthorization();

app.MapControllers();

app.Run();

Models/ UserLogin.cs

namespace JwtAuthDemo.Models;

public class UserLogin

{

public string Username { get; set; }

public string Password { get; set; }

}

Controllers / 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

{

private readonly IConfiguration \_config;

public AuthController(IConfiguration config)

{

\_config = config;

}

[HttpPost("login")]

public IActionResult Login([FromBody] UserLogin user)

{

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

{

var token = GenerateToken(user);

return Ok(new { token });

}

return Unauthorized("Invalid credentials");

}

private string GenerateToken(UserLogin user)

{

var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(\_config["Jwt:Key"]));

var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);

var claims = new[]

{

new Claim(ClaimTypes.Name, user.Username)

};

var token = new JwtSecurityToken(

issuer: \_config["Jwt:Issuer"],

audience: \_config["Jwt:Audience"],

claims: claims,

expires: DateTime.Now.AddMinutes(30),

signingCredentials: credentials);

return new JwtSecurityTokenHandler().WriteToken(token);

}

}

OUTPUT:

