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

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

Code:

public class LoginModel  
{  
 public string Username { get; set; }  
 public string Password { get; set; }  
}

public class Program  
{  
 public static void Main(string[] args)  
 {  
 var builder = WebApplication.CreateBuilder(args);  
   
 // Add services to the container.  
   
 builder.Services.AddControllers();  
 // Learn more about configuring Swagger/OpenAPI at  
https://aka.ms/aspnetcore/swashbuckle  
 builder.Services.AddEndpointsApiExplorer();  
 builder.Services.AddSwaggerGen();  
 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["Jw  
:Key"]))  
 };  
 });  
 builder.Services.AddAuthorization();  
   
 var app = builder.Build();  
   
 // Configure the HTTP request pipeline.  
 if (app.Environment.IsDevelopment())  
 {  
 app.UseSwagger();  
 app.UseSwaggerUI();  
 }  
   
 app.UseHttpsRedirection();  
   
 app.UseAuthentication();  
 app.UseAuthorization();  
   
   
 app.MapControllers();  
   
 app.Run();  
 }  
}

[ApiController]  
[Route("api/[controller]")]  
public class AuthController : ControllerBase  
{  
 [HttpPost("login")]  
 public IActionResult Login([FromBody] LoginModel model)  
 {  
 if (IsValidUser(model))  
 {  
 var token = GenerateJwtToken(model.Username);  
 return Ok(new { Token = token });  
 }  
   
 return Unauthorized();  
 }  
   
 private bool IsValidUser(LoginModel model)  
 {  
 return model.Username == "admin" && model.Password ==  
"password";  
 }  
   
 private string GenerateJwtToken(string username)  
 {  
 var claims = new[]  
 {  
 new Claim(ClaimTypes.Name, username)  
 };  
   
 var key = new SymmetricSecurityKey(  
   
Encoding.UTF8.GetBytes("ThisIsASuperSecretKeyThatIsAtLeast32CharsLong  
));  
 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);  
 }  
}

Output:



