Assignment 5

# Objective

This assignment demonstrates how to implement Cross-Origin Resource Sharing (CORS) and secure a Web API using JWT (JSON Web Token) authentication in ASP.NET Core.  
Key topics covered include:  
- What is CORS and how to enable it  
- JWT token generation and validation  
- Role-based authorization using [Authorize] and claims  
- Using Swagger and Postman to test protected API endpoints

# Implementation Steps

## Step 1: Enableb CORS

In Program.cs, add the following CORS policy:

builder.Services.AddCors(options =>  
{  
 options.AddPolicy("AllowLocalhost3000",  
 builder => builder  
 .WithOrigins("http://localhost:3000")  
 .AllowAnyHeader()  
 .AllowAnyMethod());  
});  
...  
app.UseCors("AllowLocalhost3000");

## Step 2: Configured JWT Authentication

In Program.cs, configure JWT authentication:

string securityKey = "this\_is\_a\_super\_secure\_256bit\_key123!";  
var symmetricSecurityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(securityKey));  
  
builder.Services.AddAuthentication(x =>  
{  
 x.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;  
 x.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;  
 x.DefaultSignInScheme = JwtBearerDefaults.AuthenticationScheme;  
})  
.AddJwtBearer(JwtBearerDefaults.AuthenticationScheme, x =>  
{  
 x.TokenValidationParameters = new TokenValidationParameters  
 {  
 ValidateIssuer = true,  
 ValidateAudience = true,  
 ValidateLifetime = true,  
 ValidateIssuerSigningKey = true,  
 ValidIssuer = "mySystem",  
 ValidAudience = "myUsers",  
 IssuerSigningKey = symmetricSecurityKey  
 };  
});  
...  
app.UseAuthentication();

## Step 3: Created AuthController

Created a controller that generates a JWT token:

[ApiController]  
[Route("api/[controller]")]  
[AllowAnonymous]  
public class AuthController : ControllerBase  
{  
 [HttpGet("token")]  
 public IActionResult GetToken()  
 {  
 var token = GenerateJSONWebToken(1, "Admin");  
 return Ok(new { token });  
 }  
  
 private string GenerateJSONWebToken(int userId, string userRole)  
 {  
 var securityKey = new SymmetricSecurityKey(Encoding.UTF8.GetBytes("this\_is\_a\_super\_secure\_256bit\_key123!"));  
 var credentials = new SigningCredentials(securityKey, SecurityAlgorithms.HmacSha256);  
  
 var claims = new List<Claim>  
 {  
 new Claim(ClaimTypes.Role, userRole),  
 new Claim("UserId", userId.ToString())  
 };  
  
 var token = new JwtSecurityToken(  
 issuer: "mySystem",  
 audience: "myUsers",  
 claims: claims,  
 expires: DateTime.Now.AddMinutes(10),  
 signingCredentials: credentials);  
  
 return new JwtSecurityTokenHandler().WriteToken(token);  
 }  
}

## Step 4: Securing EmployeeController

Modified EmployeeController.cs to require authorization:

[ApiController]  
[Route("api/[controller]")]  
[Authorize(Roles = "Admin,POC")]  
public class EmployeeController : ControllerBase  
{  
 // existing code remains the same...  
}

## Step 5: Enable Authorize Button in Swagger

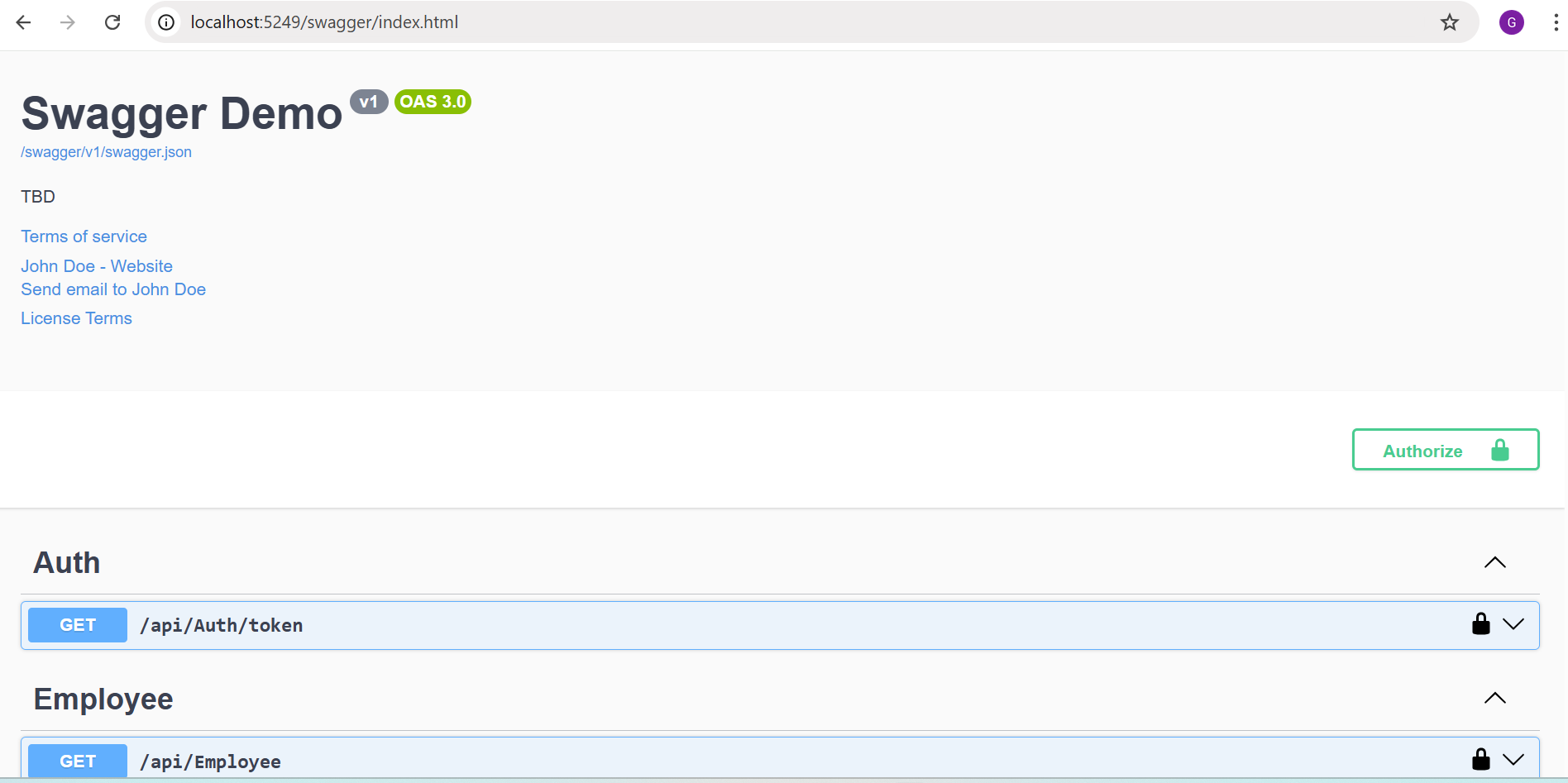
Add the following configuration to Swagger in Program.cs:

options.AddSecurityDefinition("Bearer", new OpenApiSecurityScheme  
{  
 Name = "Authorization",  
 Type = SecuritySchemeType.ApiKey,  
 Scheme = "Bearer",  
 BearerFormat = "JWT",  
 In = ParameterLocation.Header,  
 Description = "Enter 'Bearer' followed by your JWT token."  
});  
  
options.AddSecurityRequirement(new OpenApiSecurityRequirement  
{  
 {  
 new OpenApiSecurityScheme  
 {  
 Reference = new OpenApiReference  
 {  
 Type = ReferenceType.SecurityScheme,  
 Id = "Bearer"  
 }  
 },  
 Array.Empty<string>()  
 }  
});

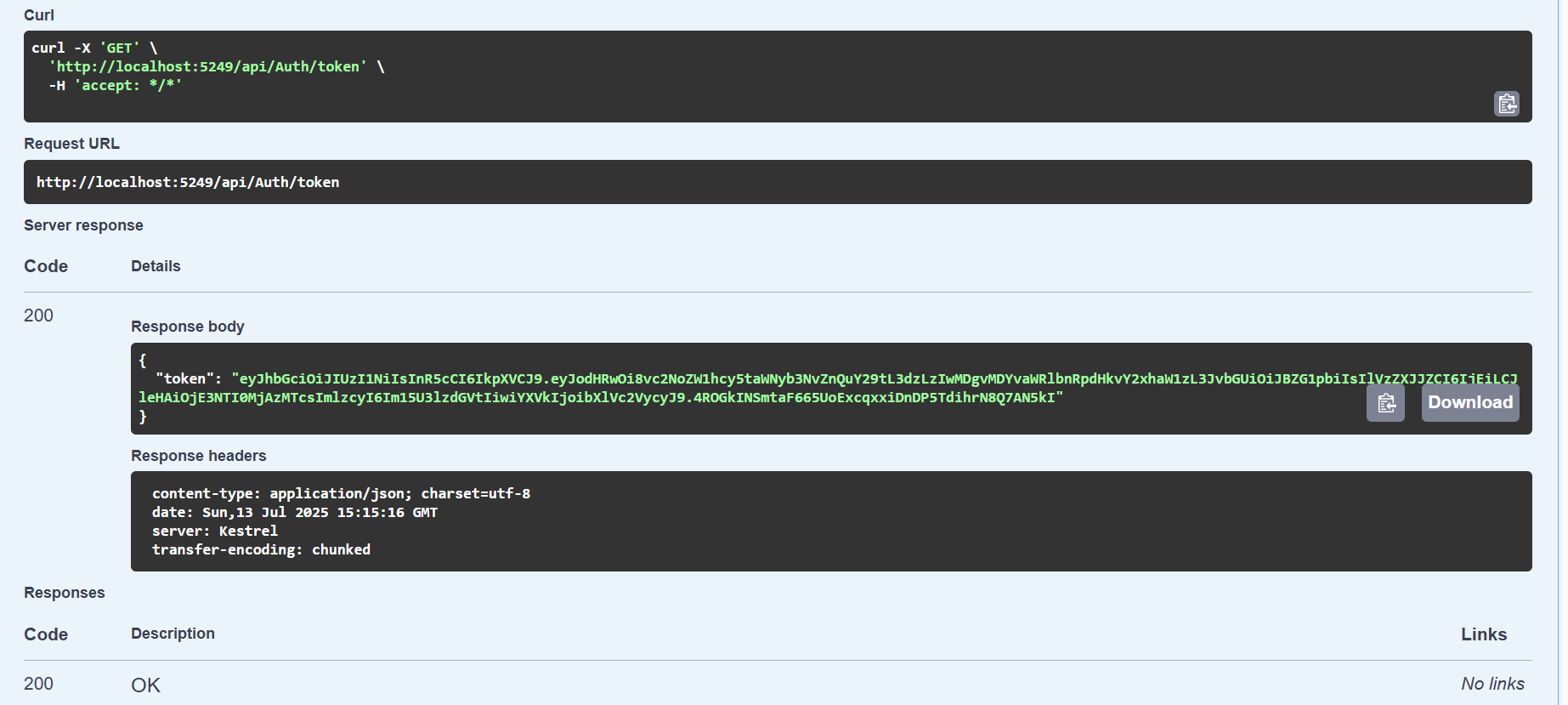
## Step 6: Testing

- Used GET /api/Auth/token to generate a JWT.  
- Used the Authorize button in Swagger or Authorization header in Postman.  
- Access protected endpoints:  
 - Valid token → 200 OK  
 - Expired token → 401 Unauthorized  
 - Missing or invalid token → 401 Unauthorized  
 - Role mismatch → 401 Unauthorized

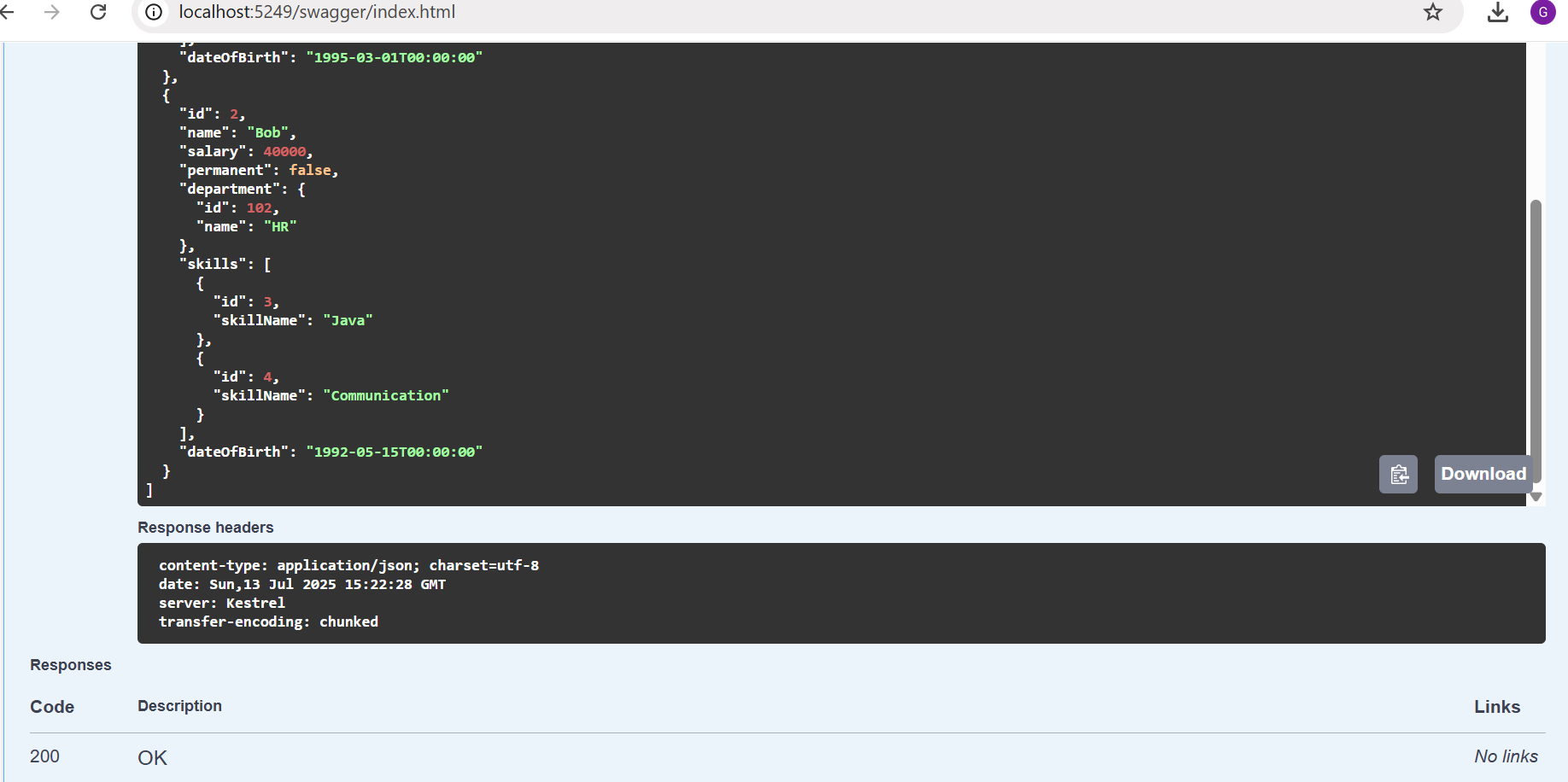
# Screenshots

  
Swagger UI showing /api/Auth/token request and response:

Swagger Authorize popup with pasted token:



GET /api/Employee result with valid token (200 OK):



Postman test for authorized :

