

JWT Tokens



1. Install the Required NuGet Packages

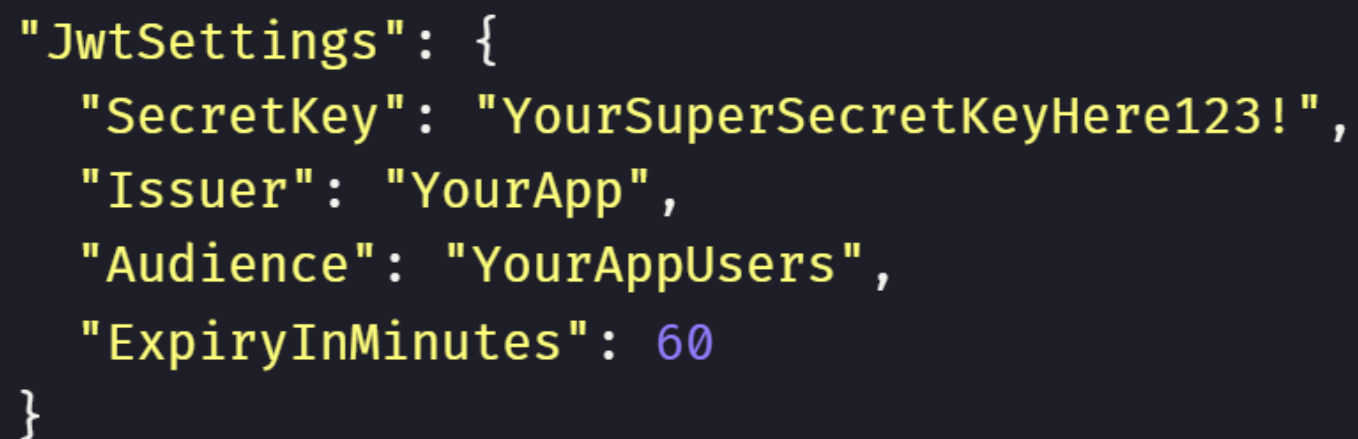
Make sure your project has these packages:



```
dotnet add package Microsoft.AspNetCore.Authentication.JwtBearer  
dotnet add package System.IdentityModel.Tokens.Jwt
```



2. Add JWT Settings to appsettings.json



```
"JwtSettings": {  
  "SecretKey": "YourSuperSecretKeyHere123!",  
  "Issuer": "YourApp",  
  "Audience": "YourAppUsers",  
  "ExpiryInMinutes": 60  
}
```



3. Configure Authentication in Program.cs or Startup.cs

```
var jwtSettings = builder.Configuration.GetSection("JwtSettings");
var secretKey = Encoding.UTF8.GetBytes(jwtSettings["SecretKey"]);

builder.Services.AddAuthentication(options =>
{
    options.DefaultAuthenticateScheme = JwtBearerDefaults.AuthenticationScheme;
    options.DefaultChallengeScheme = JwtBearerDefaults.AuthenticationScheme;
})
.AddJwtBearer(options =>
{
    options.TokenValidationParameters = new TokenValidationParameters
    {
        ValidateIssuer = true,
        ValidateAudience = true,
        ValidateLifetime = true,
        ValidateIssuerSigningKey = true,
        ValidIssuer = jwtSettings["Issuer"],
        ValidAudience = jwtSettings["Audience"],
        IssuerSigningKey = new SymmetricSecurityKey(secretKey)
    };
});
```



4. Create a Token Generator Helper

```
public class JwtTokenGenerator
{
    private readonly IConfiguration _config;

    public JwtTokenGenerator(IConfiguration config)
    {
        _config = config;
    }

    public string GenerateToken(string userId, string email)
    {
        var claims = new[]
        {
            new Claim(JwtRegisteredClaimNames.Sub, userId),
            new Claim(JwtRegisteredClaimNames.Email, email),
            new Claim(JwtRegisteredClaimNames.Jti, Guid.NewGuid().ToString())
        };

        var key = new SymmetricSecurityKey(Encoding.UTF8.GetBytes(_config["JwtSettings:SecretKey"]));
        var creds = new SigningCredentials(key, SecurityAlgorithms.HmacSha256);
        var expires = DateTime.UtcNow.AddMinutes(
            Convert.ToDouble(_config["JwtSettings:ExpiryInMinutes"]));

        var token = new JwtSecurityToken(
            issuer: _config["JwtSettings:Issuer"],
            audience: _config["JwtSettings:Audience"],
            claims: claims,
            expires: expires,
            signingCredentials: creds
        );

        return new JwtSecurityTokenHandler().WriteToken(token);
    }
}
```



5. Use It in Your Login Endpoint

```
[HttpPost("login")]
public IActionResult Login([FromBody] LoginDto dto)
{
    // Example validation (replace with your DB/user check)
    if (dto.Email == "test@demo.com" && dto.Password == "123456")
    {
        var token = _jwtTokenGenerator.GenerateToken("1", dto.Email);
        return Ok(new { Token = token });
    }

    return Unauthorized();
}
```



6. Protect Your API Endpoints with [Authorize]



```
[Authorize]
[HttpGet("secure-data")]
public IActionResult GetSecureData()
{
    return Ok("You are authorized to see this!");
}
```