MiddleWare

➡ A function that sit between an incoming request and the final response handler in an application

➡ Middleware functions have access to the request, response and the next function (next) in the request-response cycle

➡ They are commonly used as :

- Logging requests
- Authentication & authorization
- Parsing request bodies (JSON, URL-encoded data)
- Error handling
- Modifying requests and responses

MiddleWare Process

1. Request Arrives
   - A client sends a request to the server (e.g., a GET or POST request)

2. Middleware Execution Begins
   - Middleware processes the request, It authenticates, validates, or modifies it

3. next() Invoked
   - The *next()* function passes control. It moves to the next middleware or handler

4. Final Handler
   - The request reaches its destination, The handler creates & sends a response

Middleware Cautions

➡ Always Call next()

- If you forget to call next(), the request will hang indefinitely
- If next() is called multiple times, it may trigger unexpected behavior

➡ Order of Middleware Matters

- Middleware is executed in the order it is defined
- Incorrect ordering can cause unintended behavior

➡ Be Careful with Global Middleware

- Applying middleware globally affects all routes, which might not be intended
- Apply middleware only where needed for better efficiency

Middleware Cautions - Cont..

➡ Avoid Overuse of Middleware

- Too many middleware layers slow down requests
- Optimize by combining functionalities where possible

➡ Avoid Blocking the Event Loop

- Middleware should be non-blocking to ensure smooth performance
- Avoid synchronous operations like heavy computations inside middleware

➡ Be Careful with Third-Party Middleware

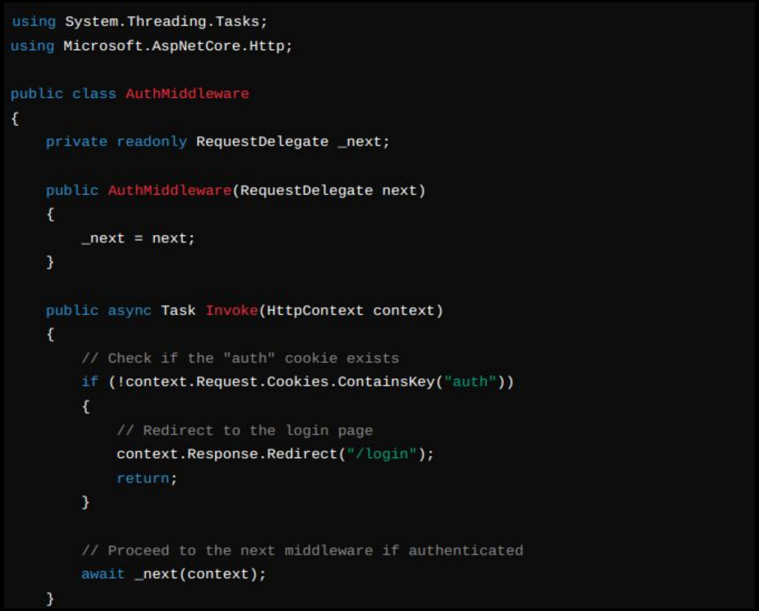
- Always review third-party middleware before using it
- Avoid outdated, unmaintained, or insecure packages

MiddleWare Types

Middleware Type	Purpose
Built-in Middleware	Predefined middleware like UseRouting(), UseAuthentication()
Custom Middleware	Custom classes for processing requests (UseMiddleware<T>())
Inline Middleware	Middleware written directly in Program.cs
Terminal Middleware	Ends request processing (app.Run())
Conditional Middleware	Applies middleware based on conditions (UseWhen())

Middleware Example - Authentication MW

When there is no cookie named “auth” exists, which means user is not authorized yet, hence following middleware redirect user to /login



```
using System.Threading.Tasks;
using Microsoft.AspNetCore.Http;

public class AuthMiddleware
{
    private readonly RequestDelegate _next;

    public AuthMiddleware(RequestDelegate next)
    {
        _next = next;
    }

    public async Task Invoke(HttpContext context)
    {
        // Check if the "auth" cookie exists
        if (!context.Request.Cookies.ContainsKey("auth"))
        {
            // Redirect to the login page
            context.Response.Redirect("/login");
            return;
        }

        // Proceed to the next middleware if authenticated
        await _next(context);
    }
```

Middleware Example - Built-In MW - UseStaticFiles()

A middleware which serves CSS, JS, images, and other static files

```
var builder = WebApplication.CreateBuilder(args);
var app = builder.Build();

app.UseStaticFiles(); // Serves files from wwwroot/

app.MapGet("/", () => "Static files enabled!");
app.Run();
```

Middleware Example - Terminal MW

No next() executed, following middleware always stops execution after returning a response

```
var builder = WebApplication.CreateBuilder(args);
var app = builder.Build();

app.Use(async (context, next) =>
{
    await context.Response.WriteAsync("This is a Terminal Middleware!");
    // No next() call, so the pipeline stops here.
});

app.MapGet("/", () => "This will never be reached!");
```

Resources

https://learn.microsoft.com/en-us/aspnet/core/fundamentals/middleware

https://theonetechnologies.com/blog/post/middleware-in-net-core-application

https://medium.com/@shubhadeepchat/net-core-middleware-explained-8c21bf646700