Elliot One • 2nd
AI-Powered SaaS Builder | Founder @ XANT & Monoversity | Entrepreneur
16h •

Follow

Middleware is the heart of ASP .NET Core.

It controls how requests flow into your app and how responses are returned.

One of the most common mistakes?

Misordering middleware.

This can break authentication, introduce security vulnerabilities, and slow down performance.

How Middleware Works in ASP .NET Core:

↳ Middleware runs exactly in the order it's registered.

↳ Each piece of middleware can:

- Inspect or modify the request.
- Pass it along to the next component.
- Short-circuit the pipeline if needed.
- Requests flow top-down.
- Responses flow bottom-up.

Incorrect ordering can prevent middleware from running or make it run too late.

✅ Recommended Middleware Order:

1) **Exception Handling** & **Security**

• app.UseExceptionHandler("/error"); ⇢ Catches unhandled errors.

• app.UseHsts(); ⇢ Enforces HTTPS in production.

• app.UseHttpsRedirection(); ⇢ Redirects HTTP to HTTPS.

2) **Static Files**

- app.UseStaticFiles(); ⇢ Serves CSS, JS, images, etc.

3) **Routing**

- app.UseRouting(); ⇢ Determines the request path.

4) **CORS** (**Cross-Origin Resource Sharing**)

- app.UseCors(); ⇢ Must come before authentication.

5) **Authentication** & **Authorization**

- app.UseAuthentication(); ⇢ Identifies the user.

- app.UseAuthorization(); ⇢ Checks user permissions.

6) **Custom Middleware**

- app.UseMiddleware<CustomMiddleware>(); ⇢ For custom logic.

7) **Endpoint Mapping**

- app.UseEndpoints(endpoints => { endpoints.MapControllers(); });

Putting middleware in the wrong place can lead to hard-to-debug issues.

Ever struggled with middleware ordering?

♻️ Reshare this with your network to spread knowledge!

🔔 Follow me [Elliot One] for modern software engineering content.

#dotnet #csharp #coding #programming

Tauseef Fayyaz and 115 others 7 comments 22 reposts