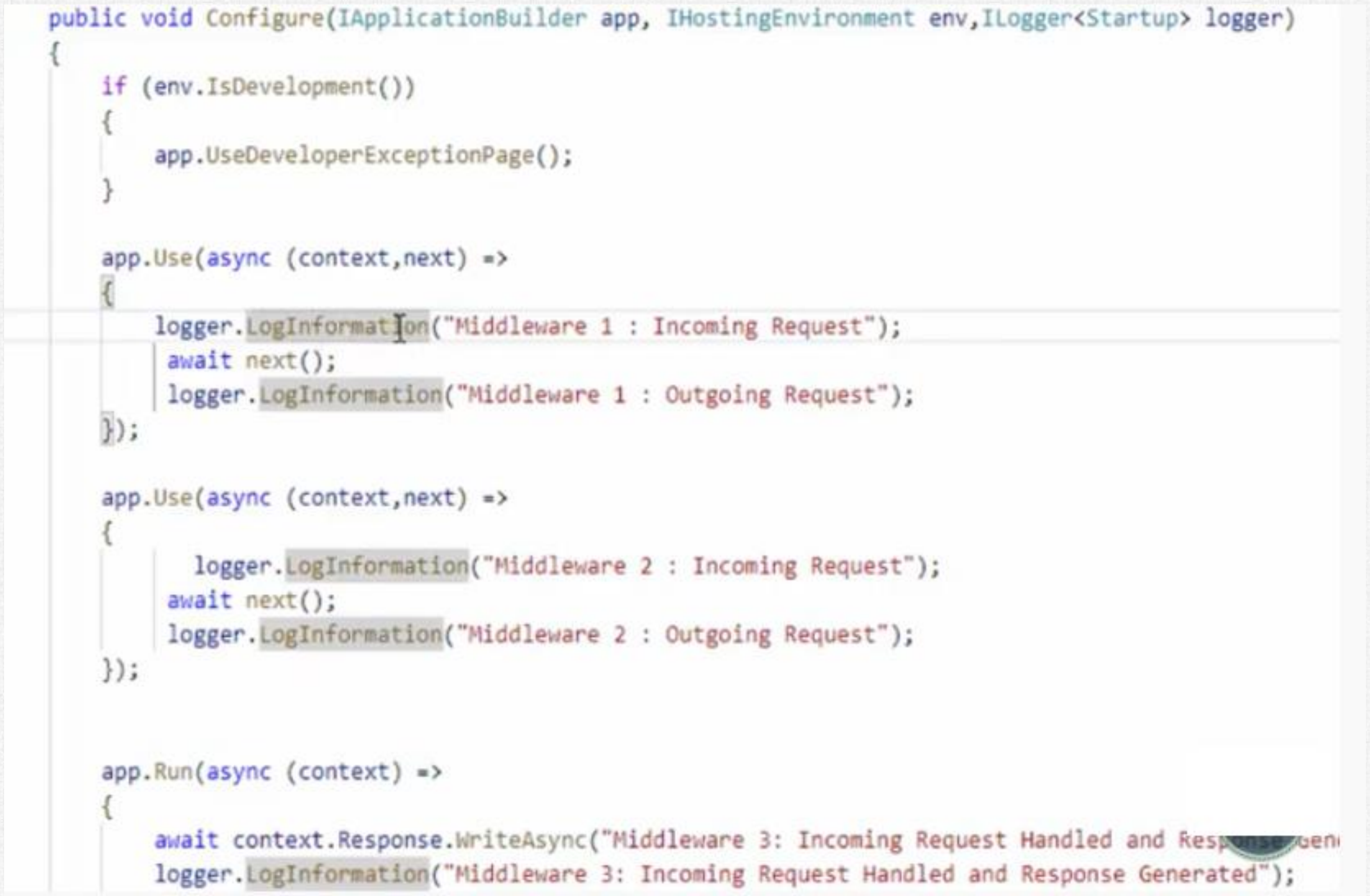
Request Processing Pipeline ASP.NET Core || Execution Order in Request Processing Pipeline.

Understanding the Request Processing Pipeline:

- In order to understand the Request Processing Pipeline in ASP.NET Core, concept, let us modify the Configure() method of the Startup class
- Here we will be registering three middleware components into the request processing pipeline.
- We will register the first two components using the **Use()** extension method and the last one is registered using the **Run()** extension method.

3 middle ware component



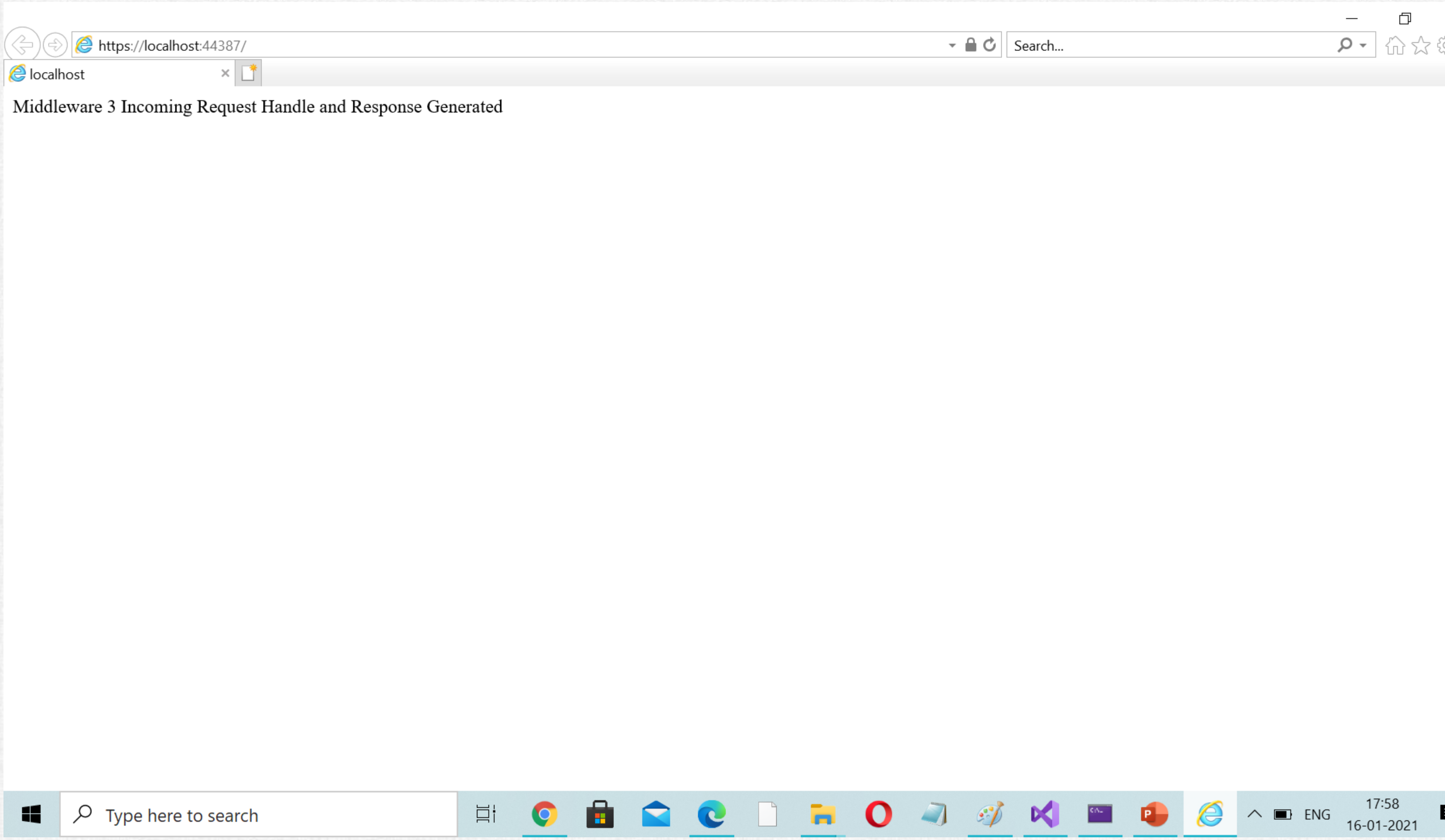
```
public void Configure(IApplicationBuilder app, IHostingEnvironment env,ILogger<Startup> logger)
{
    if (env.IsDevelopment())
    {
        app.UseDeveloperExceptionPage();
    }

    app.Use(async (context,next) =>
    {
        logger.LogInformation("Middleware 1 : Incoming Request");
         await next();
         logger.LogInformation("Middleware 1 : Outgoing Request");
    });

    app.Use(async (context,next) =>
    {
          logger.LogInformation("Middleware 2 : Incoming Request");
         await next();
         logger.LogInformation("Middleware 2 : Outgoing Request");
    });


    app.Run(async (context) =>
    {
        await context.Response.WriteAsync("Middleware 3: Incoming Request Handled and Resp[illegible]Gen
        logger.LogInformation("Middleware 3: Incoming Request Handled and Response Generated");
```

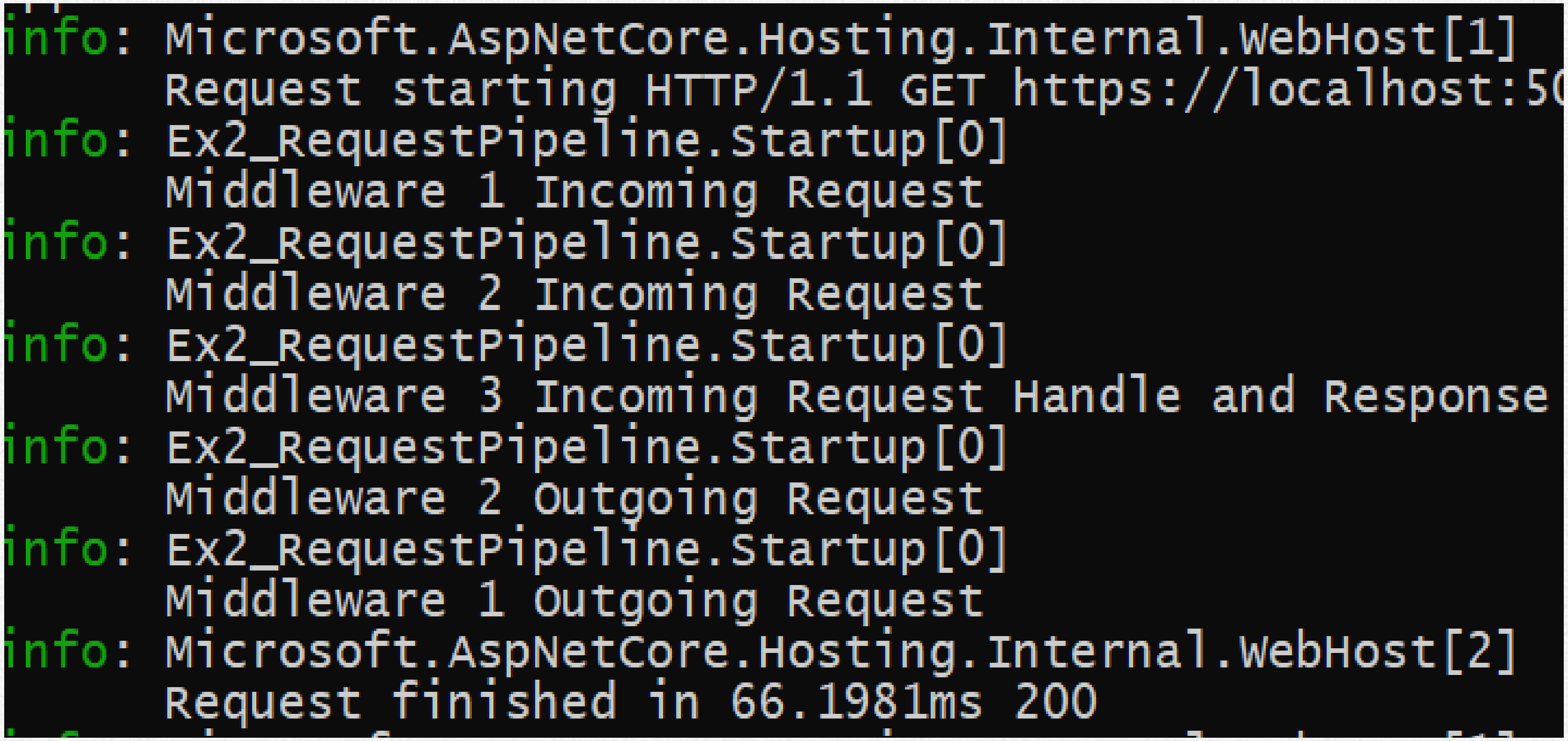
Output in browser



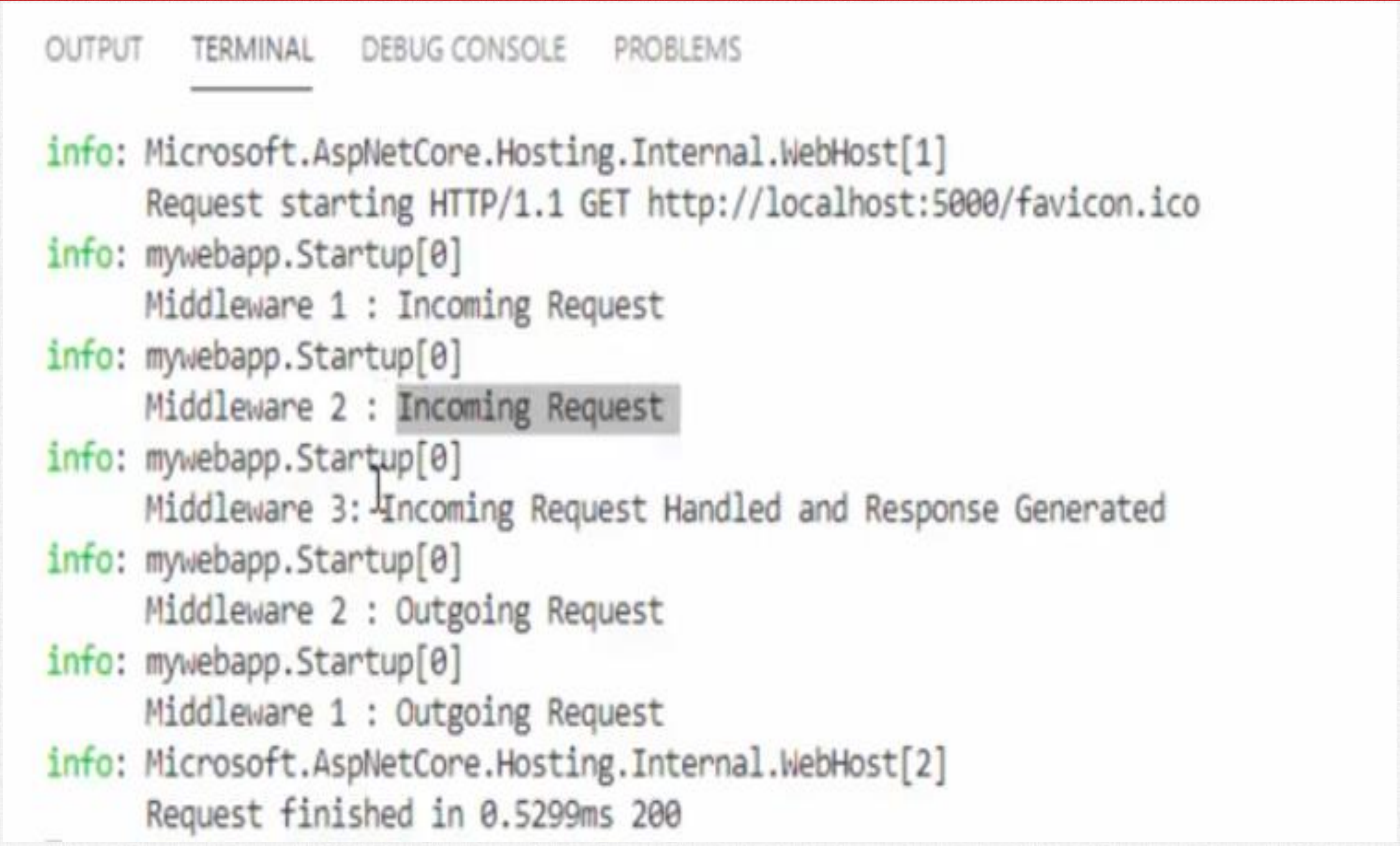
```
Middleware 3 Incoming Request Handle and Response Generated
```

Command prompt output

- Middleware 1 Incoming Request
- Middleware 2 Incoming Request
- Middleware 3 Incoming Request Handle and Response Generated
- Middleware 2 Outgoing Request
- Middleware 1 Outgoing Request



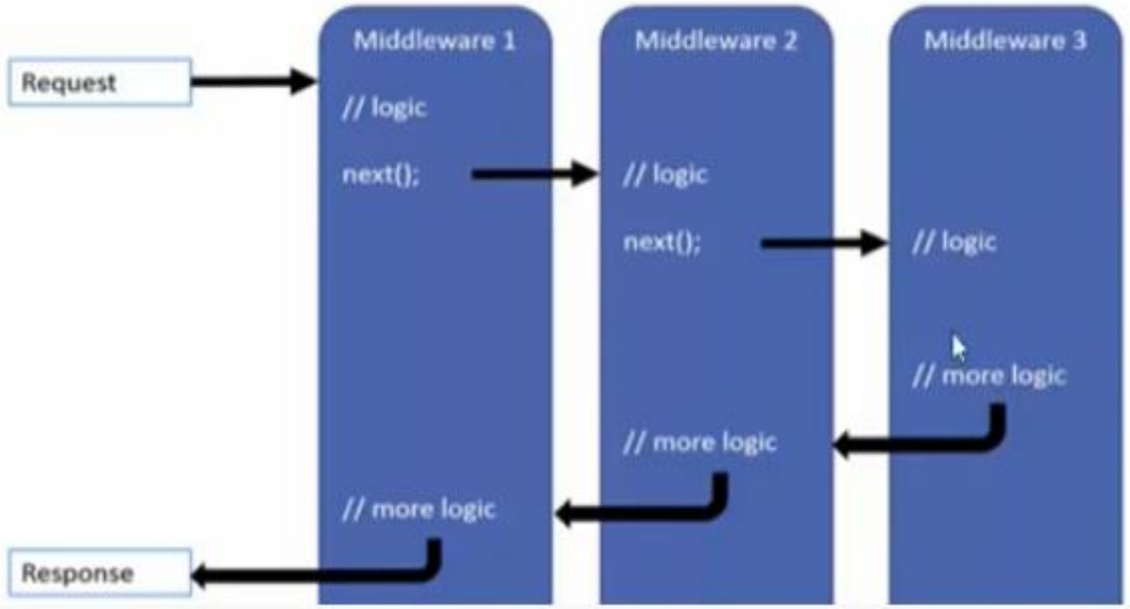
```
info: Microsoft.AspNetCore.Hosting.Internal.WebHost[1]
      Request starting HTTP/1.1 GET https://localhost:5
info: Ex2_RequestPipeline.Startup[0]
      Middleware 1 Incoming Request
info: Ex2_RequestPipeline.Startup[0]
      Middleware 2 Incoming Request
info: Ex2_RequestPipeline.Startup[0]
      Middleware 3 Incoming Request Handle and Response
info: Ex2_RequestPipeline.Startup[0]
      Middleware 2 Outgoing Request
info: Ex2_RequestPipeline.Startup[0]
      Middleware 1 Outgoing Request
info: Microsoft.AspNetCore.Hosting.Internal.WebHost[2]
      Request finished in 66.1981ms 200
```

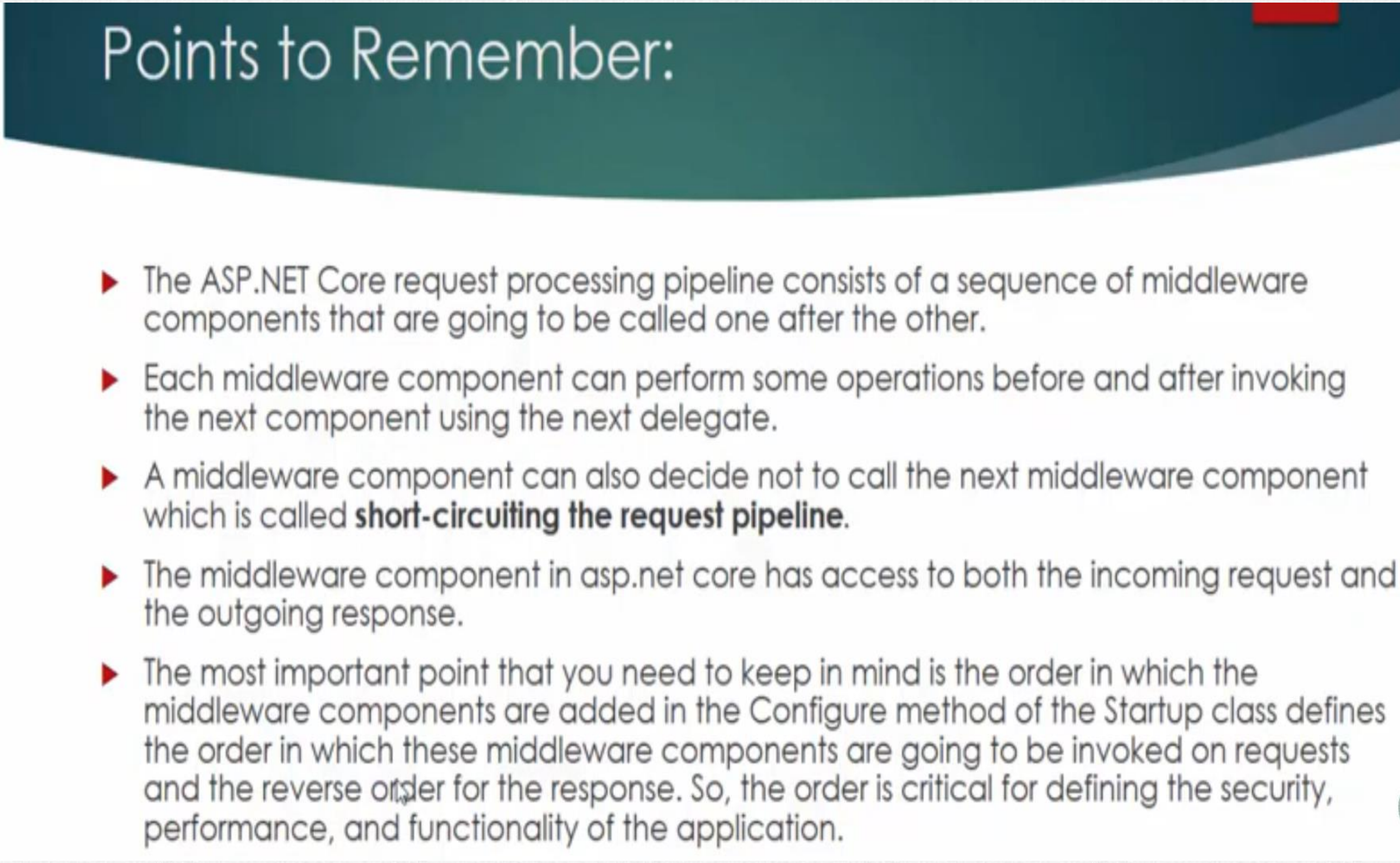


```
info: Microsoft.AspNetCore.Hosting.Internal.WebHost[1]
      Request starting HTTP/1.1 GET http://localhost:5000/favicon.ico
info: mywebapp.Startup[0]
      Middleware 1 : Incoming Request
info: mywebapp.Startup[0]
      Middleware 2 : Incoming Request
info: mywebapp.Startup[0]
      Middleware 3: Incoming Request Handled and Response Generated
info: mywebapp.Startup[0]
      Middleware 2 : Outgoing Request
info: mywebapp.Startup[0]
      Middleware 1 : Outgoing Request
info: Microsoft.AspNetCore.Hosting.Internal.WebHost[2]
      Request finished in 0.5299ms 200
```

Understanding the ASP.NET Core Request Processing Pipeline Execution Order

- In order to understand this, let us compare the above output with the following diagram to understand the ASP.NET Core Request Processing Pipeline





Points to Remember:

- The ASP.NET Core request processing pipeline consists of a sequence of middleware components that are going to be called one after the other.
- Each middleware component can perform some operations before and after invoking the next component using the next delegate.
- A middleware component can also decide not to call the next middleware component which is called **short-circuiting the request pipeline**.
- The middleware component in asp.net core has access to both the incoming request and the outgoing response.
- The most important point that you need to keep in mind is the order in which the middleware components are added in the Configure method of the Startup class defines the order in which these middleware components are going to be invoked on requests and the reverse order for the response. So, the order is critical for defining the security, performance, and functionality of the application.

Thanks