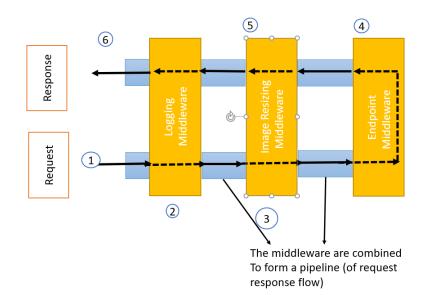
In ASP.NET Core, middleware are C# classes that can handle an HTTP request or response

Each middleware component chooses

- Whether to pass on the incoming HTTP request to another middleware in the pipeline (with or without any modification)
- Pass on the outgoing HTTP response to another middleware (or the ASP.NET Core web server) in the pipeline (with or without any modification)
- The EndpointMiddleware is perhaps the most important piece of middleware.
- This middleware class normally generates all HTML pages and API responses.
- Like the image-resizing middleware, it typically receives a request, generates a response, and then sends it back to the user, as shown in figure 1.

A Middleware Pipeline (Fig 1)

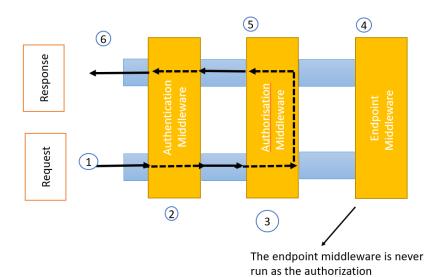


- ASP.NET Core Web Server passes the HTTP request to the middleware pipeline
- The logging middleware notes the arrival time and passes the request to the next middleware
- For request of an image of a specific size, the image resize middleware would handle it(and return the response), otherwise it is passed onto the next middleware
- If the request reaches the endpoint middleware, it will handle it and generate a response
- The response passes back through each middleware that ran previously
- The response is <u>returned back</u> to ASP.NET Core Web Server

Common Use Cases of Middleware:

- Irrespective of the resource requested(from the pipeline) or even the specific request path, there are common concerns for our application as follows:
 - Associating a request with the relevant user
 - Setting the language for the current request
 - Logging each request
 - Adding security headers to the response

Common Use Cases (Fig 2)



- ASP.NET Core Web Server passes the HTTP request to the middleware pipeline
- The authentication middleware associates a user with the request
- The authorization middleware uses the user associated with the request to see if the request can execute
- 4. If the user is not allowed, the authorization middleware will short circuit the pipeline
- The response passes back through each middleware that ran previously
- The response is returned to ASP.NET Core Web Server
- From the previous fig (Fig 2) and its discussion, we have an important point to consider, which is that the pipeline is two-directional

middleware handled the request

- The request passes through the pipeline in one direction till it is handled by a middleware, in which case a response is generated, and the flow is reversed through the pipeline
- It passes through each piece of the middleware a second time, reaching the first piece of middleware (This means the middleware that was next to a middleware when the request was flowing through them, becomes the previous middleware while the response travels back)
- Finally, the first(last in the reverse direction) middleware will pass the response back to the web server