Add Custom Middleware in ASP.NET Core Application

You will learn how to create and add your own custom middleware into the request pipeline of ASP.NET Core application.

The custom middleware component is like any other .NET class with Invoke() method. However, in order to execute next middleware in a sequence, it should have RequestDelegate type parameter in the constructor.

Visual Studio includes template for creating standard middleware class. For this, right click on the project or folder where you want to create middleware class and select Add -> New Item. This will open Add New Item popup. Search for word "middleware" in the top right search box as shown below.

Select Middleware Class item and give it a name and click on Add button. This will add a new class for the middleware with extension method as shown below.

// You may need to install the Microsoft.AspNetCore.Http.Abstractions package into your project

public class MyMiddleware

{

private readonly RequestDelegate \_next;

public MyMiddleware(RequestDelegate next)

{

\_next = next;

}

public Task Invoke(HttpContext httpContext)

{

return \_next(httpContext);

}

}

// Extension method used to add the middleware to the HTTP request pipeline.

public static class MyMiddlewareExtensions

{

public static IApplicationBuilder UseMyMiddleware(this IApplicationBuilder builder)

{

return builder.UseMiddleware<MyMiddleware>();

}

}

As you can see above, the Invoke() method is not asynchronous. So, change it to asynchronous and write your custom logic before calling next();

public class MyMiddleware

{

private readonly RequestDelegate \_next;

private readonly ILogger \_logger;

public MyMiddleware(RequestDelegate next, ILoggerFactory logFactory)

{

\_next = next;

\_logger = logFactory.CreateLogger("MyMiddleware");

}

public async Task Invoke(HttpContext httpContext)

{

\_logger.LogInformation("MyMiddleware executing..");

await \_next(httpContext); // calling next middleware

}

}

// Extension method used to add the middleware to the HTTP request pipeline.

public static class MyMiddlewareExtensions

{

public static IApplicationBuilder UseMyMiddleware(this IApplicationBuilder builder)

{

return builder.UseMiddleware<MyMiddleware>();

}

}

Add Custom Middleware

Now, we need to add our custom middleware in the request pipeline by using Use extension method as shown below.

Example: Add Middleware into Request Pipeline

 Copy

public void Configure(IApplicationBuilder app, IHostingEnvironment env)

{

app.UseMyMiddleware();

app.Run(async (context) =>

{

await context.Response.WriteAsync("Hello World!");

});

}

We can add middleware using app.UseMiddleware<MyMiddleware>() method of IApplicationBuilder also.

Thus, we can add custom middleware in the ASP.NET Core application.