Code Academy

Lecturer

Rokas Slaboševičius

ApiKey-based authentication

Data

ApiKey-based authentication

This is another way to authenticate with web services.

This method is best suited for communication between two web services rather than between a human and a service.

To do this, we will create our own attribute and use it on a class or method.

Setting-up

```
[AttributeUsage(AttributeTargets.Method | AttributeTargets.Class)]
public class ApiKeyAuthAttribute : Attribute, IAsyncActionFilter
{
    private const string ApiKeyHeaderName = "ApiKey";
    public async Task OnActionExecutionAsync(ActionExecutingContext context, ActionExecutionDelegate next)
    {
        if(!context.HttpContext.Request.Headers.TryGetValue(ApiKeyHeaderName, out var potentialApiKey))
        {
            context.Result = new UnauthorizedResult();
            return;
        }

        var configuration = context.HttpContext.RequestServices.GetRequiredService<IConfiguration>();
        var apiKey = configuration.GetValue<string>("ApiKey");

        if (!apiKey.Equals(potentialApiKey))
        {
            context.Result = new UnauthorizedResult();
            return;
        }

        await next();
    }
}
```

Usage

```
[ApiKeyAuth]
[HttpGet("ExternalCall")]
public void Get()
{

}
```

Setting-up

```
  "Jwt": {
    "Key": "7e070bdece42bd55243ea423aa9d024ae655d611bc8fab598fb543e5f3545224133bf7befc9112dab34c4535d50c107551b580660c353c83476cd1e4f87ccaed",
    "Issuer": "https://localhost:44338/",
    "Audience": "https://localhost:44338/"
  },
  "ApiKey": "7e070bdece42bd55243ea423aa9d024ae655d611bc8fab598fb543e5f3545224133bf7befc9112dab34c4535d50c107551b580660c353c83476cd1e4f87ccaed"
}
```

ApiKey-based authentication

Task 1

- Create an API call from 'one visual studio to another'. Start two API projects and call one endpoint through the other, create learned attributes and use them to protect the endpoints that you call between services.