Cookiebased authentication (Assignment 1)

Description

You're tasked with developing a new authentication method for multiple systems at work. You need to use the OpenIDConnect (OIDC) protocol. At the moment the current authentication method is hardcoded into every application in use. It's your job to design a new authentication process that enables a loose coopling between the authentication service and the multiple applications at hand. Ensure that future applications can easily be extended to add authentication capabilities.

Before starting on the task your techlead wants you to

1. Read and understand the OIDC protocol
2. Create a sequence diagram that shows the authentication process.
3. Simulate an Identity Provider (IDP) by setting up a local keycloak server https://www.keycloak.org/getting-started/getting-started-docker

Task

- Create an API that handles the OIDC authentication process from your simulated IDP. You can manually send the required requests or use a library, to do so.
- Create a web application that utilises the OIDC API to authenticate and retrieve the user's claims from the IDP.
- Create a resource API that retrieves a list of names ONLY if the user is authenticated.
- Through your webapplication, retrieve the list of names from the resource API to ensure that you're properly authenticated.

After the Task

- Create a sequence diagram that shows the final authentication process

Stuff to keep in mind

- Setup your code in a repo, and create a new branch before starting so that it's easier to review the code with a PR.
- Both the APIs and webapplication can be written in any desired language and framework.

Potential help

- Identity server 4.0 from duende software or the Microsoft.AspNetCore.Authentication.OpenIdConnect library are typically used commercially to implement OIDC authentication in .NET

Ease the implementation process

If the task is giving you trouble try to follow these steps:

1. Start off by making a webapplication that implements authentication and presents the userclaims on a view.

2. Split the authentication part from the webapplication. The webapplication should still achieve the same result.
3. Create a REST API, and create a resource that returns "I'm here but I'm not authorized!", when called from your webapplication.
4. Add another resource to your REST API that can only be accessed if authorized. It should return "I'm here and I'm authorized!".