Authentication and authorization state data in code in blazor

In our previous video we discussed, how to use [Authorize] attribute to authorize access to routable components (i.e components with @page directive). We did this in the component html.

In this video we will discuss, how to obtain authentication and authorization state data in code in blazor.

Cascading AuthenticationState parameter

* Cascading AuthenticationState parameter (Task<AuthenticationState>) provides authentication and authorization state data.
* If the user is not authenticated, the request is redirected to the login page.
* The return url is also passed as the query string parameter to the login page.
* Upon successful login, the user will be redirected to the page he was trying to access.

public class EditEmployeeBase : ComponentBase

{

[CascadingParameter]

private Task<AuthenticationState> authenticationStateTask { get; set; }

[Inject]

public NavigationManager NavigationManager { get; set; }

protected async override Task OnInitializedAsync()

{

var authenticationState = await authenticationStateTask;

if (!authenticationState.User.Identity.IsAuthenticated)

{

string returnUrl = WebUtility.UrlEncode($"/editEmployee/{Id}");

NavigationManager.NavigateTo($"/identity/account/login?returnUrl={returnUrl}");

}

// rest of the code

}

}

Copy

Check if authenticated user is in a specific role

if (authenticationState.User.IsInRole("Administrator"))

{

// Execute Admin logic

}

Copy

Check if authenticated user satisfies a specific policy

Task<AuthenticationState> can be combined with IAuthorizationService, to check if a specific aothorization policy is satisfied.

public class EditEmployeeBase : ComponentBase

{

[CascadingParameter]

private Task<AuthenticationState> authenticationStateTask { get; set; }

[Inject]

private IAuthorizationService AuthorizationService { get; set; }

protected async override Task OnInitializedAsync()

{

var user = (await authenticationStateTask).User;

if ((await AuthorizationService.AuthorizeAsync(user, "admin-policy"))

.Succeeded)

{

// Execute code specific to admin-policy

}

}

}