**Cookie Authentication in .net core 1**

Please visit the link for more information

<https://docs.microsoft.com/en-us/aspnet/core/security/authentication/cookie?tabs=aspnetcore1x>

Here we are using cookie authentication without core identity

STEP1.

Install the Microsoft.AspNetCore.Authentication.Cookies NuGet package in your project.

Add the following lines to the Configure method in your Startup.cs file before the app.UseMvc()statement:

app.UseCookieAuthentication(new CookieAuthenticationOptions()

{

AuthenticationScheme = "CookieAuthentication",

LoginPath = new PathString("/Account/Login"),

AccessDeniedPath = new PathString("/Account/Forbidden"),

AutomaticAuthenticate = true,

AutomaticChallenge = true

});

In the above path we can set login path we set login view rendering action method.

AccessDeniedPath redirects to a view when authentication challenge happen.

When a user enters his/her credential, it will be verified with data store,if it is a legitimate call then a ClaimsPrincipal object is created and stored in the cookie.

ClaimsPrincipal object contains ClaimasIdentity which is a collection of claims for the legitimate user. Claims contain rolename etc and custom claim like AccName

var claims = new List<Claim>

{

new Claim(ClaimTypes.Name, userVM.LoginName),

new Claim(ClaimTypes.Role, userVM.RoleName),

new Claim("RoleID", userVM.RoleID.ToString()),

new Claim("AccID", userVM.\_LinxAccountObj.ID.ToString()),

new Claim("AccName",userVM.\_LinxAccountObj.Name)

};

var userIdentity = new ClaimsIdentity(claims, "login");

ClaimsPrincipal principal = new ClaimsPrincipal(userIdentity);

//cookie is created using SignInAsync

await HttpContext.Authentication.SignInAsync("CookieAuthentication", principal,new AuthenticationProperties { ExpiresUtc = DateTime.UtcNow.AddMinutes(20) });

STEP2.

After cookie generation, each subsequent http call should be authenticated.

So we use MVC filters.There are several filters. We use authorize filter class.

This filter will be invoked before controller/Action where it decorated

For example

Filter can be applied for entire controller

[Authorize]

public class HomeController : Controller

OR

Filter can be applied for action method only

[HttpGet]

[Authorize(Roles = "god,admin")]

public IActionResult Index()

//We can pass parameters to Authorize filter where it restrict the user having god,admin roles. Otherwise challenge will be invoked redirects to forbidden view

STEP3.

For logout we can use the following code

await HttpContext.Authentication.SignOutAsync("CookieAuthentication");

STEP4.

We can access logged user information from the following object

ClaimsPrincipal

Example:-

In action method

HttpContext.User is a ClaimsPrincipal object

ClaimsPrincipal \_user

Name = \_user.Identity.Name;

RoleID = int.Parse (\_user.Claims.FirstOrDefault(x => x.Type == "RoleID").Value);