**Security Assignments**

**Assignment 1: Authorize Attribute**

**pe**

**Problem Statement:** Create an MVC application and restrict user to access ‘Contact us’ page but ‘about us’ page should be accessed by any user.

***Hints:*** use Authorize attribute to restrict user and use “AllowAnonymous” attribute to give access to anyone.

***Note:*** The attribute supports a couple of parameters through which developers can restrict the execution of the action method only to certain user names and/or users with a given role. Here’s an example:

|  |
| --- |
| [Authorize(Roles="admin", Users="Bob, Alice")]  public ActionResult Index()  {     ...  } |

If a user is not authenticated, or doesn’t have the required user name and role, then the Authorize attribute prevents access to the method and redirects the user to the login URL.

In the example above, only users Bob and Alice having the role of Admin can have access to the method.

**Assignment 2: Authentication Filter**

**pe**

**Problem Statement:** Create a custom authentication filter to authenticate user to navigate to homepage.

***Step 1:*** Create a MVC project named as “CustomAuthenticationDemo”.

***Step 2:*** Add a folder named as “Filters”.

***Step 3:*** Add a class “CustomAuthenticationFilter” into that folder.

***Step 4:***

* After creating Filter next UserAuthenticationFilter will inherit a class FilterAttribute and IAuthenticationFilter.
* And in this filter we are just going to check Session is IsNullOrEmpty if it is NULL or Empty then you will set Result to HttpUnauthorizedResult ();
* And in OnAuthenticationChallenge method we are going check this result is not null or Result is HttpUnauthorizedResult.
* If result is HttpUnauthorizedResult then we are going to redirect it to Error View.
* Refer the below sample code snippet:

|  |
| --- |
| public class UserAuthenticationFilter : ActionFilterAttribute, IAuthenticationFilter  {  public void OnAuthentication(AuthenticationContext filterContext)  {  //Check Session is Empty Then set as Result is HttpUnauthorizedResult  }  //Runs after the OnAuthentication method  //------------//  //OnAuthenticationChallenge:- if Method gets called when Authentication or Authorization is  //failed and this method is called after  //Execution of Action Method but before rendering of View  //------------//  public void OnAuthenticationChallenge(AuthenticationChallengeContext filterContext)  {  //We are checking Result is null or Result is HttpUnauthorizedResult  // if yes then we are Redirect to Error View  ------------------------------------------  {  filterContext.Result = new ViewResult  {  ViewName = "Error"  };  }  }  } |

***Step 5:*** After creating UserAuthenticationFilter in next step, create a simple login page. For doing that you have to create Model “LoginModel” for login page.

***Step 6:*** Then add a controller named “UserLoginController”.

***Step 7:*** add 2 Action Methods named as ‘Login’ one which handles HttpGet Request and other which handle HttpPost Request.

***Step 8:*** In Post Request we are checking ModelState is valid or not if not then we are return model to View which show error on View.

***Step 9:*** If it is valid then we are going to check Username and Password is valid or not for this demo I have hardcoded values.

***Step 10:*** The Username to "Test" and Password to "Test" and if enter Username and Password is equal then we set Session["UserID"] and redirect to Index View of Home Controller.

Refer the sample code:

|  |
| --- |
| ------------------------------------------------  if (login.UserName == "Rittik" && login.Password == "test1234")  {  Session["UserID"] = Guid.NewGuid();  return RedirectToAction("Index", "Home");  }  else  {  ModelState.AddModelError("", "Invalid Login attempt.");  return View(login);  }  -------------------------- |

***Note:*** Home controller comes by Default when you create Application.

***Step 11:*** After adding UserLoginController in next step we are going to add apply UserAuthentication Filter on HomeController.

|  |
| --- |
| [<Custom Authentication Filter Name>]    public class HomeController : Controller  { |

***Note:*** Now if User is logged with valid credentials into application then only will be able to access Home Controller, because we have applied UserAuthentication Filter on Home Controller.

***Step 12:*** save this application and run.

Next Index Action Method of HomeController.

URL: [http://localhost:####/home/index](https://www.learnmvc.in/aspnet-mvc5-authentication-filters-example.php)

|  |
| --- |
| C:\Users\M1042829\Desktop\15.jpg |

***Note:*** do not have access to this page, so it is redirecting to Error View because we have applied UserAuthentication Filter on it.

***Step 13*** – Have to access Login Action Method of UserLoginController.

URL - [http://localhost:####/UserLogin/login](https://www.learnmvc.in/aspnet-mvc5-authentication-filters-example.php)

|  |
| --- |
|  |

***Step 14 –*** After entering Valid Credentials Authentication will be checked and will be redirected to HomeController as show below.

If you enter Invalid Credentials then you cannot Login and cannot access Home controller.

***KeyFocus Points of this assignment:***

* Functionality of Authorize attribute
* Custom authentication using only ActionFilterAttribute and IAuthentication Filter

***Future Scope:***

* Custom authentication using AuthorizeAttribute class
* Functionality of other MVC filters

--Thanks--