using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

namespace WebApplication8.Controllers

{

public class Home1Controller : Controller

{

public IActionResult Index()

{

//read cookie from Request object

string userName = Request.Cookies["UserName"];

return View("Index", userName);

}

[HttpPost]

public IActionResult Index(IFormCollection form)

{

string userName = form["userName"].ToString();

//set the key value in Cookie

CookieOptions option = new CookieOptions();

option.Expires = DateTime.Now.AddMinutes(10);

Response.Cookies.Append("UserName", userName, option);

return RedirectToAction(nameof(Index));

}

public IActionResult RemoveCookie()

{

//Delete the cookie

Response.Cookies.Delete("UserName");

return View("Index");

}

}

}

@model string

@{

ViewData["Title"] = "Home Page";

}

@if (!string.IsNullOrWhiteSpace(Model))

{

@:<div>Welcome back, @Model</div>

@Html.ActionLink("Forget Me", "RemoveCookie")

}

else

{

@:

<form asp-action="Index">

<span>Hey, seems like it's your first time here!</span><br />

<label>Please provide your name:</label>

@Html.TextBox("userName")

<div class="form-group">

<input type="submit" value="Update" class="btn btn-primary" />

</div>

</form>

}

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Logging;

using WebApplication8.Models;

namespace WebApplication8.Controllers

{

public class HomeController : Controller

{

private readonly ILogger<HomeController> \_logger;

public HomeController(ILogger<HomeController> logger)

{

\_logger = logger;

if (users == null)

{

users = new List<User>()

{

new User() { UserName="ajay", Password="ajay"},

new User() { UserName="jay", Password="jay"}

};

}

}

List<User> users = null;

public IActionResult Index()

{

string name = this.HttpContext.Session.GetString("user");

if (name == null)

{

TempData["user"] = "Invalid ";

return Redirect("~/Home/Login");

}

return View();

}

public IActionResult Login()

{

return View();

}

[HttpPost]

public IActionResult Login(User user)

{

var obj = users.FirstOrDefault(x => x.UserName == user.UserName &&

x.Password == user.Password);

if (obj != null)

{

this.HttpContext.Session.SetString("user", obj.UserName);

return Redirect("~/");

}

else

ViewBag.msg = "Invalid user";

return View();

}

public IActionResult Privacy()

{

return View();

}

[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]

public IActionResult Error()

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

}

}

Extension Method

using Microsoft.AspNetCore.Http;

using Newtonsoft.Json;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication8.Models

{

public static class SessionExtensions

{

public static void Set<T>(this ISession session,

string key, T value)

{

session.SetString(key, JsonConvert.SerializeObject(value));

}

public static T Get<T>(this ISession session, string key)

{ var calue = session.GetString(key);

return calue == null ? default(T) :

JsonConvert.DeserializeObject<T>(calue);

}

}

}

Test.cs

using Microsoft.AspNetCore.Http;

using System;

using System.Collections.Generic;

using System.Linq;

using System.Threading.Tasks;

namespace WebApplication8.Models

{

public class Test

{

public string value = "";

private readonly ISession \_session;

public Test(IHttpContextAccessor httpContextAccessor)

{

\_session = httpContextAccessor.HttpContext.Session;

}

public Test()

{

}

public string GetId()

{

if (\_session != null)

{

return \_session.GetString("user").ToString();

}

else

return null;

}

public void SetId(string name)

{

if (\_session != null)

{

\_session.SetString("user", name);

}

}

}

}

Home

using System;

using System.Collections.Generic;

using System.Diagnostics;

using System.Linq;

using System.Threading.Tasks;

using Microsoft.AspNetCore.Http;

using Microsoft.AspNetCore.Mvc;

using Microsoft.Extensions.Logging;

using WebApplication8.Models;

namespace WebApplication8.Controllers

{

public class HomeController : Controller

{

private readonly IServiceProvider \_serviceProvider;

private readonly ILogger<HomeController> \_logger;

public HomeController(ILogger<HomeController> logger, IServiceProvider serviceProvider)

{

\_serviceProvider = serviceProvider;

\_logger = logger;

if (users == null)

{

users = new List<User>()

{

new User() { UserName="ajay", Password="ajay"},

new User() { UserName="jay", Password="jay"}

};

}

}

List<User> users = null;

public IActionResult Index()

{

string name = this.HttpContext.Session.GetString("user");

if (name == null)

{

return Redirect("~/Home/Login");

}

else

{

var t2 = (Test)\_serviceProvider.GetService(typeof(Test));

//TempData["user"] = "Invalid ";

var t1 = t2.GetId();

return View();

//return Redirect("~/Home/Login");

}

}

public IActionResult Login()

{

return View();

}

[HttpPost]

public IActionResult Login(User user)

{

var obj = users.FirstOrDefault(x => x.UserName == user.UserName &&

x.Password == user.Password);

if (obj != null)

{

var t2 = (Test)\_serviceProvider.GetService(typeof(Test));

t2.SetId(obj.UserName);

return Redirect("~/");

}

else

ViewBag.msg = "Invalid user";

return View();

}

public IActionResult Privacy()

{

return View();

}

[ResponseCache(Duration = 0, Location = ResponseCacheLocation.None, NoStore = true)]

public IActionResult Error()

{

return View(new ErrorViewModel { RequestId = Activity.Current?.Id ?? HttpContext.TraceIdentifier });

}

}

}