**ASP.NET Core Workshop**

# **Sessions**

# Requirements

In this workshop, we will create a simple web application using Sessions with the following features.

1. When users access /Home/Login, a login page is displayed.

Graphical user interface, application

Description automatically generated

1. A user can login by providing any **username**, which should not be empty. No other checking is required.
2. For a user who has already logged in and not yet logged out, whenever he/she accesses this **Login page**, redirect him/her to the **Track page**.

Graphical user interface, application

Description automatically generated

1. On the Track page, users can click any of the buttons. The app will display the clicking history.

Graphical user interface, application

Description automatically generated

In the above screenshot, the user with username “john” has clicked button 1, then button 2, and then button 0, 0, 1 and 2.

1. When users click Logout, all the clicking history will be cleared.

# Part I. Prepare Projects

1. In Visual Studio, create a new ASP.NET Core Web App (Model-View-Controller). You can name it anything you want. For example, *SessionWorkshop*.
2. Enable ASP.NET Core Session Middleware

* In **Program.cs**, change

var builder = WebApplication.CreateBuilder(args);

// Add services to the container.

builder.Services.AddControllersWithViews();

builder.Services.AddSession();

…

* In **Program.cs**, also change

app.UseRouting();

app.UseAuthorization();

app.UseSession();

app.MapControllerRoute(

name: "default",

pattern: "{controller=Home}/{action=Index}/{id?}");

1. In the Controllers/**HomeController.cs**, add 2 new methods:

public IActionResult Login (string username)

{

return View();

}

public IActionResult Track(string clickedBtn)

{

return View();

}

1. In the **Views/Home/** folder, add two new Views, **Login** and **Track**, with the following snippets.

**Login.cshtml**

@{

ViewData["Title"] = "Login";

}

<h2>Login</h2>

<form action="Login" method="post">

<div class="form-group row">

<div class="col-sm-2">

<label for="username">Username</label>

</div>

<div class="col-sm-6">

<input name="username" class="form-control" id="username" placeholder="Username" />

</div>

<div class="col-sm-2">

<button type="submit" class="btn btn-primary" value="Submit">Submit</button>

</div>

</div>

</form>

**Track.cshtml**

@{

ViewData["Title"] = "Track";

}

<h2>Track</h2>

<form method="post">

<div class="form-group row">

<label class="col-sm-2">Username</label>

<div class="col-sm-1">@ViewData["username"]</div>

<div class="col-sm-9">

<button type="submit" formaction="Logout">Logout</button>

</div>

</div>

<div class="form-group row">

<label class="col-sm-2">Click me</label>

<div class="col-sm-10">

<button type="submit" formaction="Track?clickedBtn=0">0</button>

<button type="submit" formaction="Track?clickedBtn=1">1</button>

<button type="submit" formaction="Track?clickedBtn=2">2</button>

</div>

</div>

<div class="form-group row">

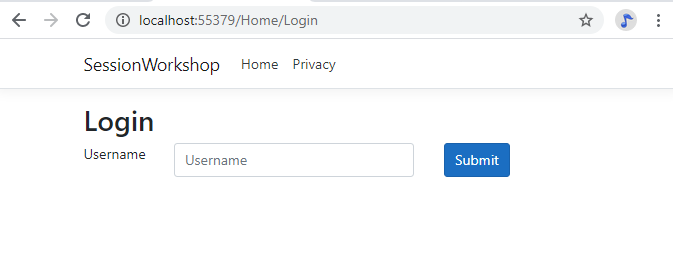
<label class="col-sm-2">Click History</label>

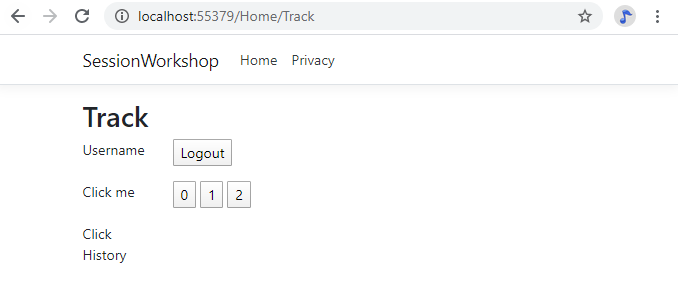
<div class="col-sm-10"></div>

</div>

</form>

1. Run the web application and access the 2 new Controller Actions, make sure you see the following pages, **Login page** and **Track page**.



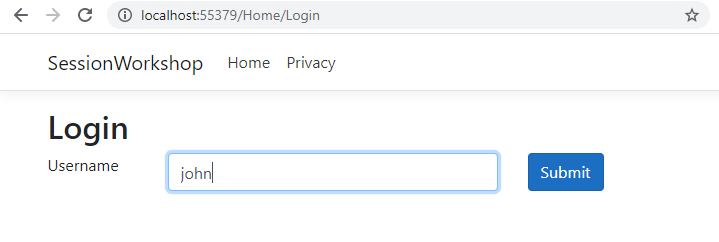


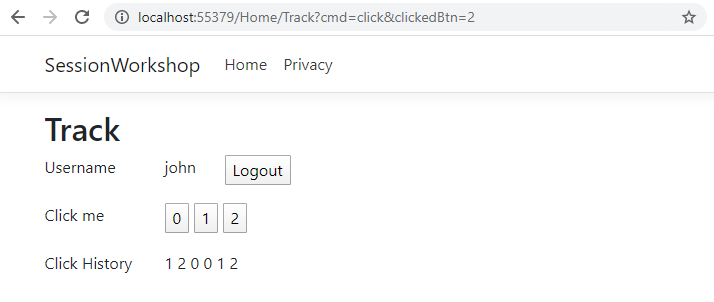
# Part II. Implement Features

Using **ASP.NET Core Session State** and its Session Object to help you in this part.

1. Modify the **HomeController Login()** and its respective **View** (if necessary) with the following features:
   1. Allow a user to login by providing any **username**, which should not be empty. No other checking is required. The web application stores the **username** into **Session Object** and redirect to the **Track page**.
   2. For a user who has already logged in and not yet logged out, whenever he/she accesses this **Login page**, redirect him/her to the **Track page**.
2. Modify the **HomeController Track()** and its respective **View** (if necessary), then add more action methods (if necessary) to implement the following features:
   1. If a user has not logged in, redirect him/her back to the **Login page**.
   2. Display the **username** previously stored in the **Session Object**.
   3. Allow users to click any of the buttons **0, 1, 2** and store that information in the **Session Object**.
   4. Display the **Click History** accordingly.
   5. Allow the user to log out by clearing the session data.

You web application should look like:





In the above screenshot, the user with username “john” has clicked button 1, then button 2, and then button 0…

# Part III. Extend Features (optional)

1. Improve the **Login page** by adding **Password** field.
   1. In order to login successfully, a user needs to provide correct **username** and **password**.
   2. In this step, you can assume that the system only accepts two users:

|  |  |
| --- | --- |
| **Username** | **Password** |
| admin | admin |
| tester | tester |

In order words, you can hard code the username and password for verifying process.

1. In reality, **User accounts** are stored in database. Improve the **Login page** by verifying username and password against a User table in database, instead of hard coding.
2. In the current web application, **Click History** will be cleared and lost after users log out or when the web application **restarts**. Improve the web application to store the **Click History** into a **database table**. Each click is linked to the user so that the user can see the Click History every time he/she logins.
3. **ASP.NET Core Session State** helps you to generate and send Session ID between the web applications and browsers, as well as store Session Data in Session Object. Improve the web application by handling all the tasks without using ASP.NET Core Session State.
   1. Implement generating a Session Id and mapping it to a logged-in user, then persisting it in database.
   2. Implement sending Session Id between the server and browser using Cookie.

With the Click History having been stored in database in the previous step. All your session data now should be persisted in the database.