Lab 3 – Razor and Request/Session

# Description

This lab is designed to give you a basic understanding of the Razor Rendering Engine and an overview of MVC’s routing feature.

# Estimated Time

This lab will take an estimated 4 hours to complete

# Deliverable

Push your code to github and submit the link to Brightspace.

Deploy your website to Windows Azure and submit the link to Brightspace.

See “Brightspace -> Course Content -> Extra Materials -> Microsoft Azure Web Application” for information about deploying Azure Web Apps.

# Notes

* Be sure Visual Studio is up to date.
* Follow along closely to the instructions!
* When using the <input> tag be sure to set the type to collect the proper values
* Use the example code ‘<https://github.com/aarad-ac/IntroductionToASP.NETMVCCore>’ for help and tips.
* Demo can be found at: <http://afrasialab3.azurewebsites.net/>

# Step 1: Create a new MVC Core project called ‘Lab3’

1. Open Visual Studio 2019
2. Click: Create a new project
3. Select: C# -> Windows -> Web
4. Select ‘ASP.NET Core Web Application’
5. Name the application ‘Lab3’, save the project in your desired location and click the button ‘OK’
6. Select the ‘Empty’ ASP.NET Core Template
7. Uncheck configure for HTTPS
8. Click the ‘OK’ button.

# Step 2: Configure your new Web Application

1. Modify Startup.cs
   1. Add the following lines to the method ‘ConfigureServices(IServiceCollection services)’

services.AddControllersWithViews();

* 1. Replace the following lines to the Configure method

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

app.UseRouting();

app.UseEndpoints(endpoints =>

{

endpoints.MapGet("/", async context =>

{

await context.Response.WriteAsync("Hello World!");

});

});

with:

if (env.IsDevelopment())

{

app.UseDeveloperExceptionPage();

}

else

{

app.UseExceptionHandler("/Home/Error");

}

app.UseStaticFiles();

app.UseRouting();

app.UseAuthorization();

app.UseEndpoints(endpoints =>

{

endpoints.MapControllerRoute(

name: "default",

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

});

}

# Step 3: Create the ‘Controllers’ and ‘Views’

1. Create a folder in your project called ‘Controllers’
2. Create a new Controller in this folder called ‘Home’ – Note that your class should be named ‘HomeController.cs’
   1. NOTE: if you use ‘MVC Controller Class’ template from visual studio it will create all the basic code you need to have a valid controller. To do so:
      1. Right click on the ‘Controllers’ folder
      2. Click ‘Add’
      3. Click ‘Controller...’
      4. Select the ‘MVC Controller - Empty’, and click ‘Add’
      5. Select ‘Controller Class – Empty’, and click ‘Add’
3. Create a folder in your project called ‘Views’
4. At the root of this new ‘Views’ folder, create a file called ‘\_ViewImports.cshtml’. You can use the following steps:
   1. Right click on the ‘Views’ folder
   2. Click ‘Add’
   3. Click ‘View...’
   4. Select the ‘Razor View - Empty’, and click ‘Add’
   5. Select ‘Razor View Imports’, and click ‘Add’
5. Add the following lines of code to the file ‘\_ViewImports.cshtml’

@using Lab3

@addTagHelper "\*, Microsoft.AspNetCore.Mvc.TagHelpers"

1. Follow similar steps and add ‘\_ViewStart.cshtml’

# Step 4: You are on your own (Part 1)

1. Create your own StyleSheet.css, or reuse what you find in my sample code
2. Create a \_Layout.cshtml file
3. Add a header and footer
4. In header, put:
   1. Your name and your welcome message
   2. Hyperlinks to your other pages
5. In footer, use HTML tags/style of your choice to explain steps you took to add css file and \_Layout. Should include the following:
   1. Short description of what layout does
   2. Why you put the files in those directories. E.g. CSS in directory x, and why. Layout in directory Y and why. Etc.
   3. You should not include the content of css or how you defined your css

# Step 5: You are on your own (Part 2)

1. Create a View in your home controller called ‘Razor.cshtml’
2. Get the number of bottles from user and store it in session or ViewData
   1. Make sure you limit the user to the range of 50 to 100 bottles (it does not matter f includes 50 and 100 or not)
3. Create a second view called ‘Count.cshtml’
4. Use Razor to draw the lyrics to the children’s song ’10 Green Bottles…’ (see: <https://en.wikipedia.org/wiki/Ten_Green_Bottles>)
   1. Use the value user entered for the bottles in Razor view
5. Let the user play again
6. Create appropriate actions in HomeController

# Step 6: You are on your own (Part 3)

1. Create a model for Person, Person.cs, under the ‘models’ folder
   1. You can do that by right clicking on Models and adding a new class
2. Create a view called ‘CreatePerson.cshtml’
3. Collect the following information from the user
   1. First Name
   2. Last Name
   3. Age
   4. Email Address
   5. Date of Birth
      1. In your model, use the attribute [DataType(DataType.Date)]
   6. Password (should be masked as you type, and it should not be shown to the user on the DisplayPerson.cshtml – see below)
   7. Description of person
4. Once collected, direct the user to the ‘DisplayPerson.cshtml’. You should also build a model for Person.
5. Display the results collected on this new view.
6. Create appropriate actions in HomeController

# Step 7: You are on your own (Part 4)

1. Create a View under Home named “Error”
   1. Put a custom message there
2. Create an action in HomeController that navigates to the Error view