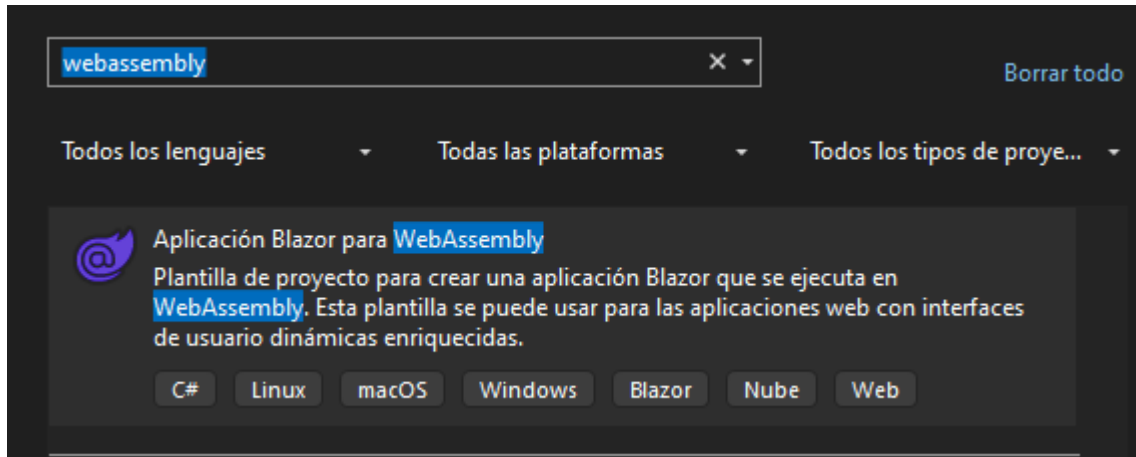


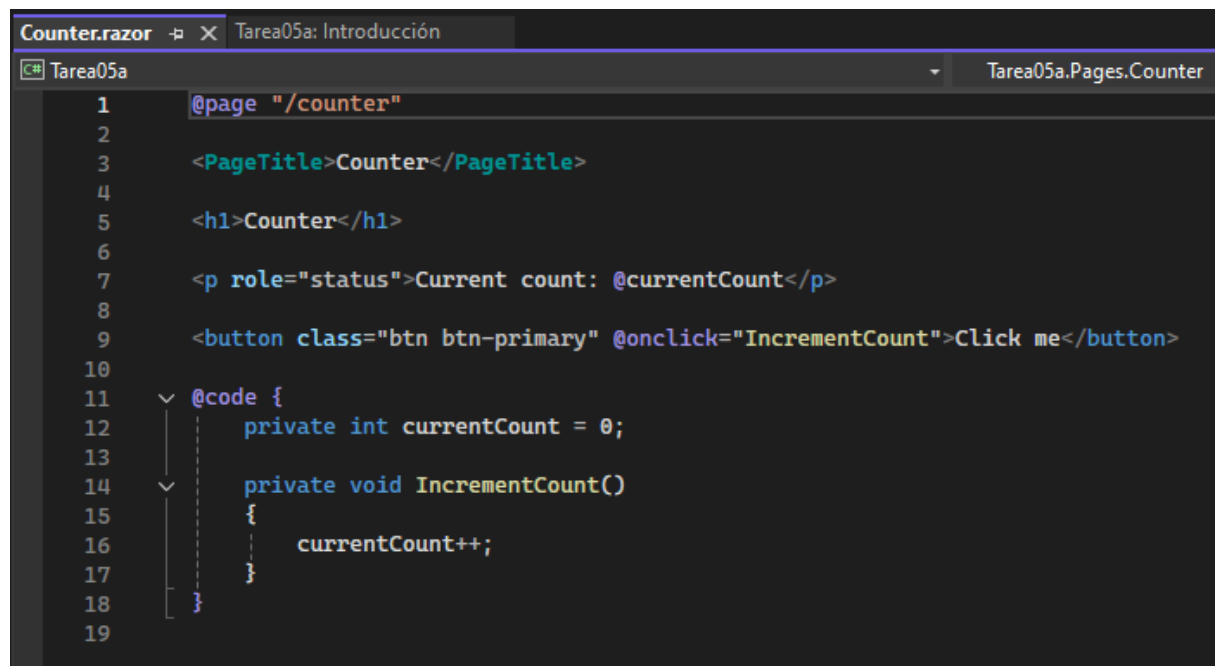
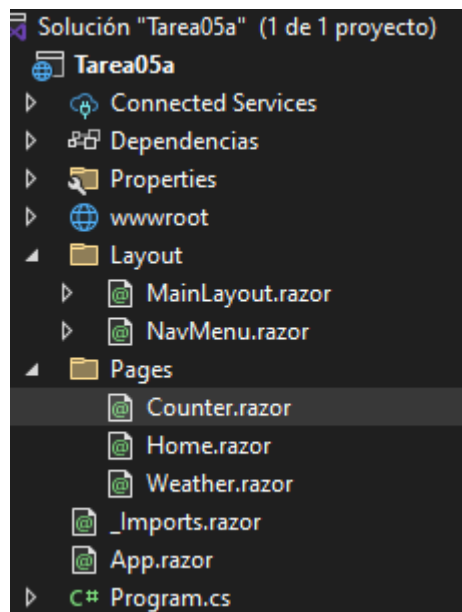
Consigna 1(básico)

Parte A:

1. Usando la plantilla correspondiente crear una aplicación Blazor WebAssembly.



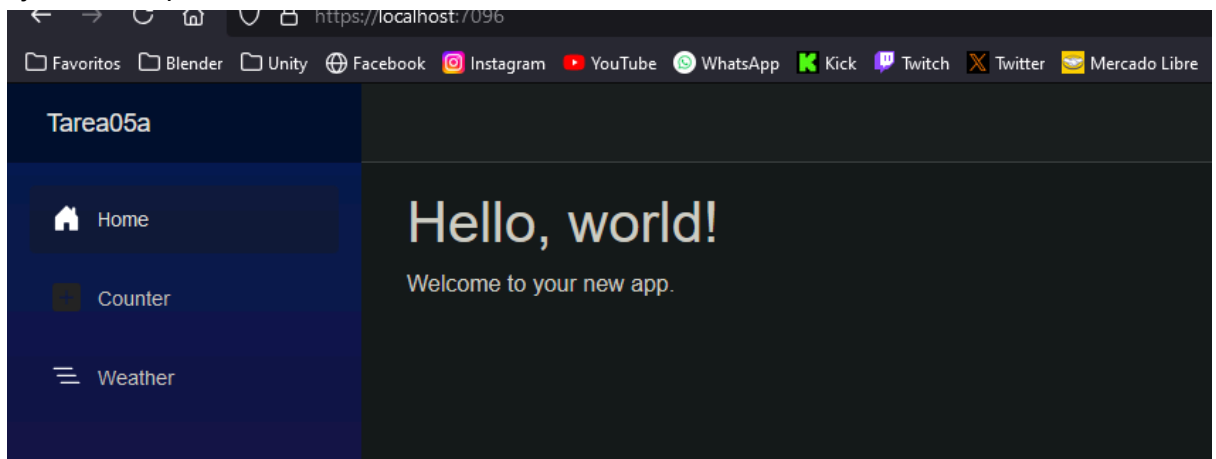
2. Localizar el código de ejemplo generado por el template que se encarga de incrementar el valor del Contador.



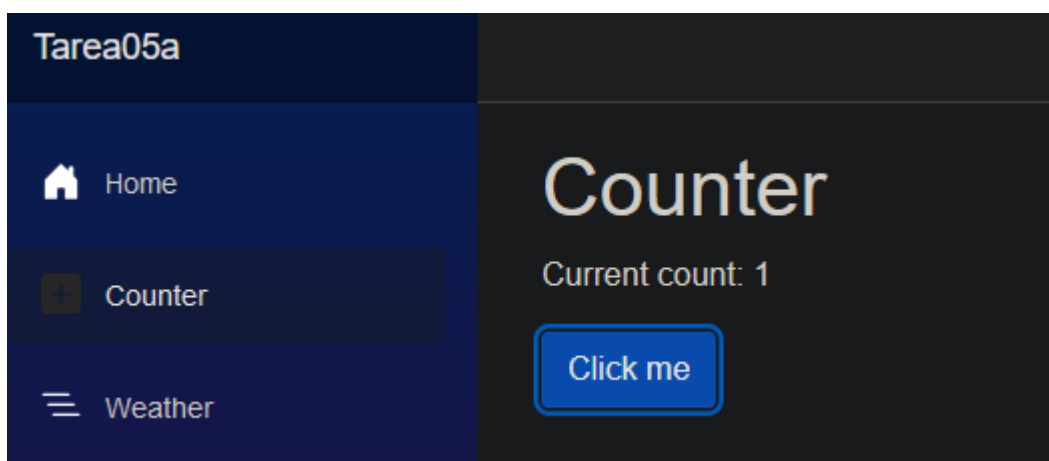
3. Agregar un log (Console.WriteLine) en el método que incrementa dicho contador.

```
1  @page "/counter"
2
3  <PageTitle>Counter</PageTitle>
4
5  <h1>Counter</h1>
6
7  <p role="status">Current count: @currentCount</p>
8
9  <button class="btn btn-primary" @onclick="IncrementCount">Click me</button>
10
11  @code {
12      private int currentCount = 0;
13
14      private void IncrementCount()
15      {
16          currentCount++;
17          Console.WriteLine($"Current count incremented to: {currentCount}");
18      }
19  }
```

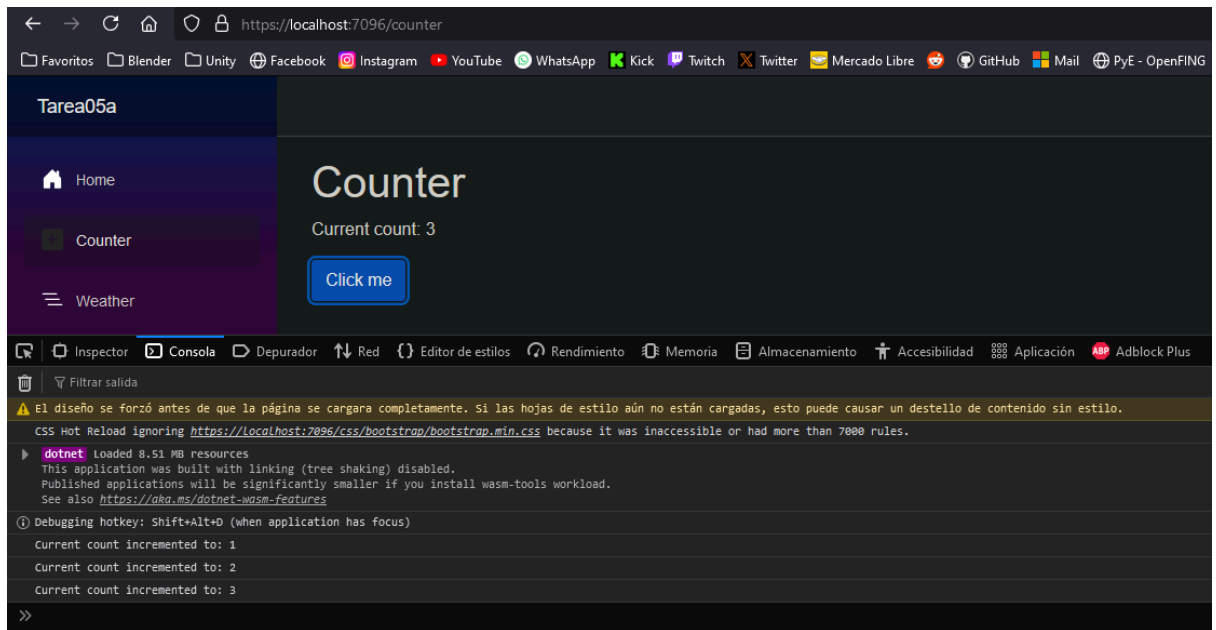
4. Ejecutar la aplicación.



5. ¿Por qué se observa una barra de progreso en la pantalla?
Cuando la aplicación se carga, el navegador necesita descargar todos los archivos necesarios para ejecutar la aplicación en el cliente, durante ese tiempo Blazor WebAssembly muestra una barra de progreso para indicar que la aplicación se está cargando.
6. Incremente el Contador.

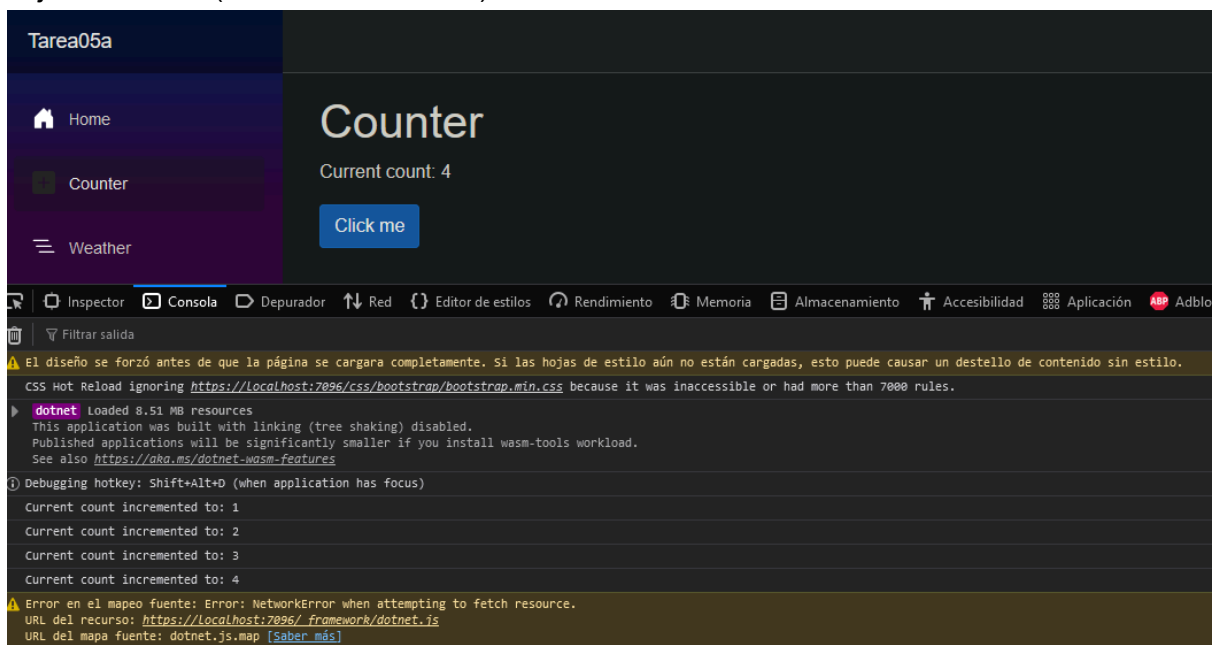


7. El log ingresado en el punto anterior, ¿dónde se despliega (en la consola del browser o en la consola del servidor)? ¿Por qué?



Utilizando **Blazor WebAssembly**, la aplicación se ejecuta completamente en el navegador, lo que significa que `Console.WriteLine` se redirigirá a la consola del browser, ya que toda la lógica se ejecuta en el cliente.

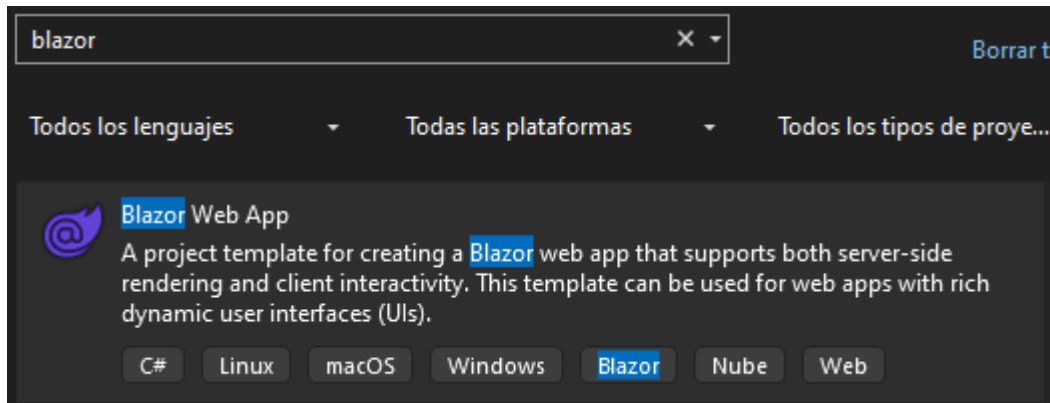
8. Baje el servidor (cerrando la consola) e intente incrementar el contador nuevamente.



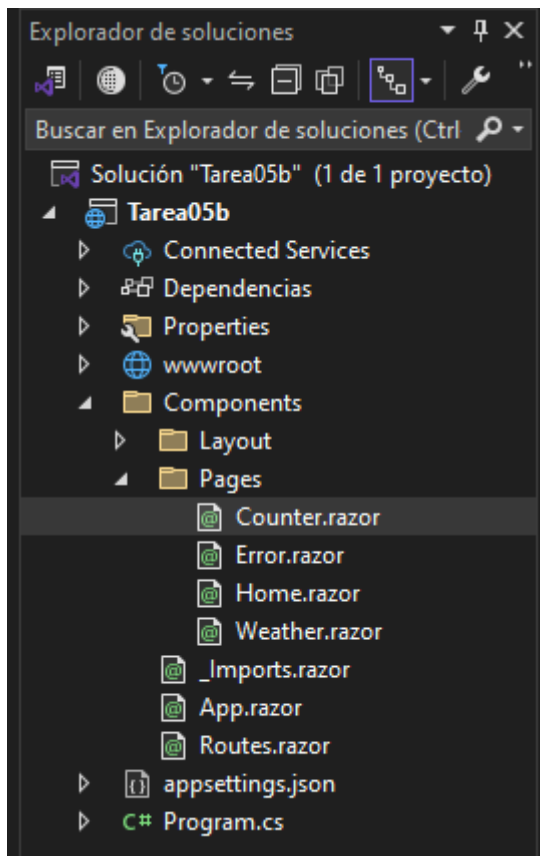
9. ¿Se incrementa el Contador? ¿Por qué?
Sí, porque la aplicación se ejecuta completamente en el navegador, debido a que Blazor descarga todos los archivos necesarios (ensamblado de la aplicación, las dependencias de .NET y otros recursos).

Parte B.

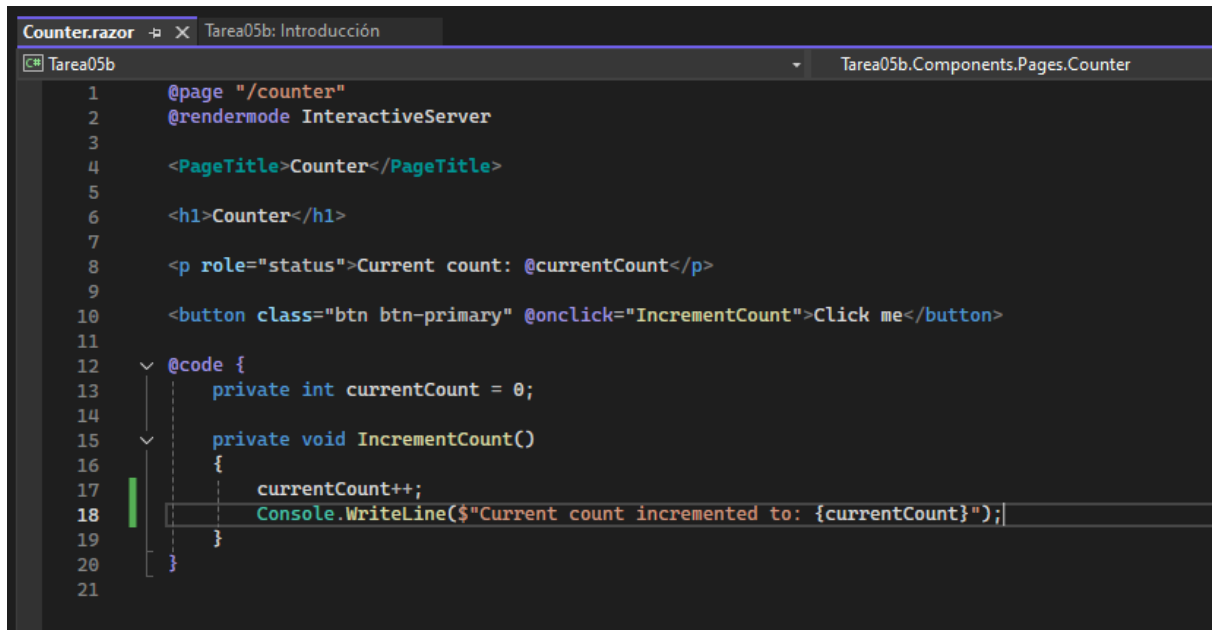
1. Usando la plantilla correspondiente crear una aplicación Blazor Web App.



2. Localizar el código de ejemplo generado por el template que se encarga de incrementar el valor del Contador.

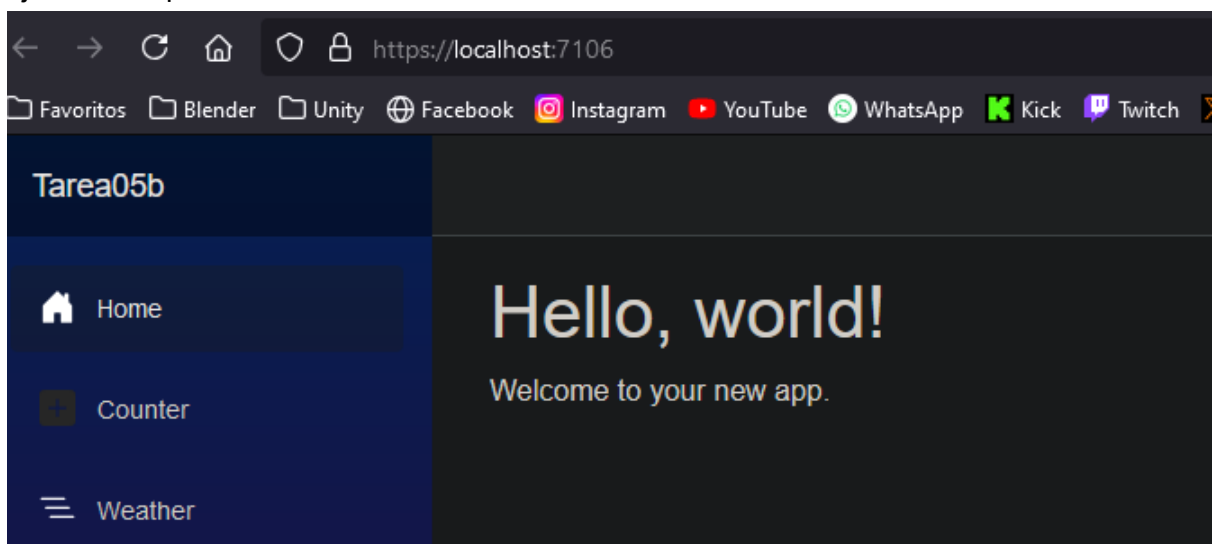


3. Agregar un log (Console.WriteLine) en el método que incrementa dicho contador.



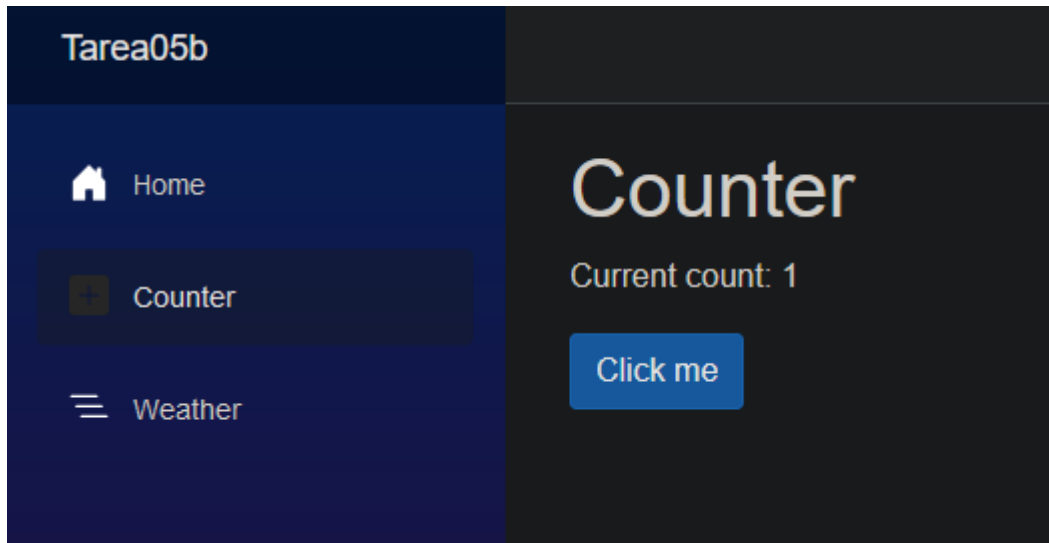
```
1 @page "/counter"
2 @rendermode InteractiveServer
3
4 <PageTitle>Counter</PageTitle>
5
6 <h1>Counter</h1>
7
8 <p role="status">Current count: @currentCount</p>
9
10 <button class="btn btn-primary" @onclick="IncrementCount">Click me</button>
11
12 @code {
13     private int currentCount = 0;
14
15     private void IncrementCount()
16     {
17         currentCount++;
18         Console.WriteLine($"Current count incremented to: {currentCount}");
19     }
20 }
21
```

4. Ejecutar la aplicación.

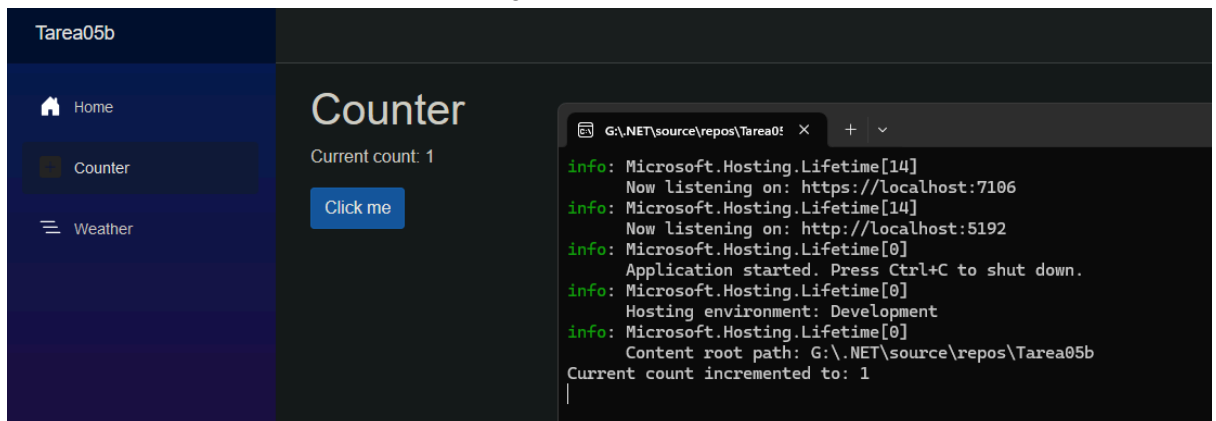


5. ¿Se observa la barra de progreso al iniciar la misma? ¿por qué?
En una **Blazor Web App** no hay barra de progreso porque utiliza **SignalR** para mantener una conexión constante entre el servidor y el cliente.

6. Incremente el Contador.

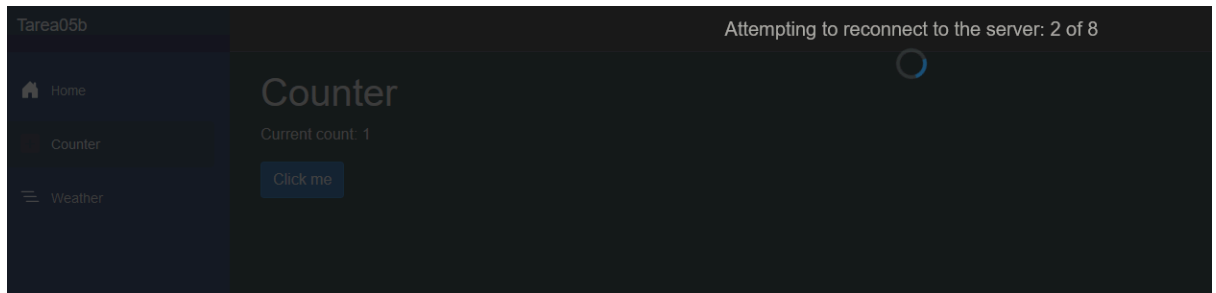


7. El log ingresado en el punto anterior, ¿dónde se despliega (en la consola del browser o en la consola del servidor)? ¿Por qué?



Los logs se despliegan en la consola del servidor porque la lógica de la aplicación se ejecuta completamente en el servidor, cuando se llama a `Console.WriteLine()`, se ejecuta en el entorno del servidor, y los mensajes se envían a la consola del servidor.

8. Baje el servidor (cerrando la consola) e intente incrementar el contador nuevamente.



9. ¿Se incrementa el Contador? ¿Por qué?
No, porque Blazor establece una conexión en tiempo real con el cliente usando **SignalR**. Aunque la interfaz de usuario se renderiza en el navegador, cualquier lógica (como el contador o cualquier proceso que se ejecute al hacer clic en un botón) se procesa en el servidor.
10. Dentro de Counter.razor comente la directiva `@rendermode`. Vuelva a ejecutar la aplicación ¿se incrementa el contador? ¿Por qué?

```
Counter.razor  Tarea05b: Introducción  Tarea05b.Components.Pages
1  @page "/counter"
2  @* @rendermode InteractiveServer *@
3
4  <PageTitle>Counter</PageTitle>
5
6  <h1>Counter</h1>
7
8  <p role="status">Current count: @currentCount</p>
9
10 <button class="btn btn-primary" @onclick="IncrementCount">Click me</button>
11
12 @code {
13     private int currentCount = 0;
14
15     private void IncrementCount()
16     {
17         currentCount++;
18         Console.WriteLine($"Current count incremented to: {currentCount}");
19     }
20 }
21
```

Al comentar `@rendermode`, el componente se renderiza únicamente como HTML estático. No habrá ninguna interacción con el servidor, por lo que el contador no se incrementará cuando se haga clic en el botón.

El evento `@onclick` depende de la conexión entre el servidor y el cliente para procesar la lógica de incremento, pero sin la directiva, esa conexión no se establece, y el clic en el botón no tiene ningún efecto en la funcionalidad del contador.

Repositorio de GitHub:

https://github.com/AndresRomano/DotNET_Pr-cticos/tree/main/Tarea05a