PRACTICAL NO. 4

Session Management and AJAX

Aim: To implement various client-side and server-side session management techniques and to demonstrate various AJAX controls in ASP.Net.

4.1 ASP.Net Web Applications managing States:

- State management means preserving state of control, web page, object/data, and user in the application explicitly because all ASP.NET web applications are stateless, i.e., by default, for each page posted to the server, the state of control is lost. Nowadays all web apps demand a high level of state management from control to application level.
   Types of state management:
   There are two types of state management techniques: client side and server side.

Client-Side State Management:

- Hidden Fields:
   A Hidden control is the control which does not render anything on the web page at client browser but can be used to store some information on the web page which can be used on the page. Hidden fields are used to store data at the page level. These fields are not rendered by the browser, rather it's just like a standard control for which you can set its properties. If you use hidden fields, it is best to store only small amounts of frequently changed data on the client.

- Cookies:
- A cookie is a small amount of data which is either stored at client side in text file or in memory of the client browser session. Cookies are always sent with the request to the web server and information can be retrieved from the cookies at the web server. Every time a user visits a website; cookies are retrieved from the user machine and help identify the user.
- Cookies are useful for storing small amounts of frequently changed information on the client. The information is sent with the request to the server.

- Query String:
   A Query string is used to pass the values or information form one page to another page. They are passed along with URL in clear text. Query strings provide a simple but limited way of maintaining some state information. When surfing the internet, you should have seen strange internet addresses such as:
   http://www.localhost.com/Webform2.aspx?name=ABC&&lastName=XYZ

This HTML address uses a QueryString property to pass values between pages.

- View State:
  ViewState is an approach to saving data for the user. Because of the stateless nature of web pages, regular page member variables will not maintain their values across postbacks. ViewState is the mechanism that allows state values to be preserved across page postbacks and by default, EnableViewState property will be set to true. In a nutsheell, ViewState:
- Stores values per control by key name, like a Hashtable.
- Tracks changes to a View State value's initial state.
- Serializes and Deserializes saved data into a hidden form field on the client.
- Automatically restores View State data on postbacks.

Server-Side State Management:
Global.asax file:

- Using this file, you can define event handlers with application-wide or session-wide scope.
- The Global.asax, also known as the ASP.NET application file, is located in the root directory of an ASP.NET application. This file contains code that is executed in response to application-level and session-level events raised by ASP.NET or by HTTP modules.
- The following are example of some events:
- Application_Start() – fired when the first resource is requested from the web server and the web application starts.
- Session_Start() – fired when session starts on each new user requesting a page.
- Application_Error() – fired when an error occurs.
- Session_End() – fired when the session of a user ends.
- Application_End() – fired when the web application ends.

Application State:

Application State is used to store information which is shared among users of the ASP.Net web application. Application state is stored in the memory of the windows process which is processing user requests on the web server. Application state is useful in storing a small amount of often-used data. If application state is used for such data instead of frequent trips to the database, then it increases the response time/performance of the web application.

In classic ASP, an application object is used to store connection strings. It's a great place to store data that changes infrequently.

Session State:

ASP.NET Session state provides a place to store values that will persist across page requests. Values stored in Session are stored on the server and will remain in memory until they are explicitly removed or until the Session expires. It is defined as the “period of time” that a unique user interacts with a Web application. Session state is a collection of objects, tied to a session stored on a server.

Session State – Cookieless Session ID’s:

By default a session uses a cookie in the background. To enable a cookie-less session, we need to change some configuration in the Web.Config file. Follow these steps:

1. Open Web.Config file
2. Add a <sessionState> tag under <system.web> tag
3. Add an attribute "cookieless" in the <sessionState> tag and set its value to "AutoDetect" like below:

```
<sessionState     cookieless="AutoDetect" regenerateExpiredSessionId="true"/>
```

Session State – Cookieless Session ID’s:

```
<sessionState        cookieless="AutoDetect" regenerateExpiredSessionId="true"/>
```

The possible values for "cookieless" attribute are:

- AutoDetect : Session uses browser cookie if cookies are enabled. If cookies are disabled, then the URL is used to store session information.
- UseCookie: Session always use browser cookie. This is default.
- UseDeviceProfile: Session uses browser/device support cookie if browser supports cookies else URL is used.
- UseUri: Session always use URL.

"regenerateExpiredSessionId" is used to ensure that if a cookieless url is expired a new new url is created with a new session. And if the same cookieless url is being used by multiple users an the same time, they all get a new regenerated session url.

Session State – Cookieless Session ID’s:

1. Then Open Mozilla Firefox and Click on (Tools -> Options -> Pricacy)
2. Now on History group box, select (Firefox will: Use custom settings for history) 3. Now uncheck (Accept cookeies from sites)
4. Run the web application.

Storing and retrieving values from Session:

Storing and retrieving values in session are quite similar to that in ViewState. We can interact with session state with the System.Web.SessionState.HttpSessionState class, because this provides the built-in session object in ASP.NET pages.

The following code is used for storing a value to session:

```
Session["UserName"] = txtUser.Text;
```

Now, let's see how we can retrieve values from session:

```
if (Session["UserName"] != null)
        {lblWelcome.Text = "Welcome: " +
Session["UserName"]; }            else { //Do Something else }
```

In ASP.NET, Session Mode determines how session state is stored and managed across user requests. It allows data to persist across multiple requests from the same user.

Types of Session Modes in ASP.NET

1. InProc (In-Process) o Default mode. o Stores session data in memory on the web server. o Fastest but not suitable for web farms (multiple servers). o Data is lost if the application restarts.

   ```
   <sessionState mode="InProc" timeout="30" />
   ```

2. StateServer (Out-of-Process) o Stores session data in a separate Windows service (aspnet_state).
   - o Can be used in web farms. o Slower than InProc but more reliable across restarts.
   - o For this start the ASP.NET State Service from Control Panel -> Administrative tools -> Services.

   ```
   <sessionState mode="StateServer"
   stateConnectionString="tcpip=127.0.0.1:42424" />
   ```

3. SQLServer o Stores session data in a SQL Server database.
   - o Reliable and good for web farms. o Slower than InProc and StateServer.
   - o Requires database setup using aspnet_regsql.

   ```
   <sessionState mode="SQLServer" allowCustomSqlDatabase="true"
   sqlConnectionString="Data Source=(localdb)\MSSQLLocalDB;Initial
   Catalog=ASPState;Integrated Security=True;" cookieless="false" timeout="5" />
   ```

4. Custom o Allows custom storage providers (e.g., Redis, NoSQL). o You must implement a custom session state provider.

```
<sessionState mode="Custom" customProvider="MySessionProvider">
  <providers>  <add name="MySessionProvider"
type="MyNamespace.MyProviderClass" />
  </providers>  </sessionState>
```

Cache:

The Cache object is an instance of the System.Web.Caching.Cache class. Cache is stored on the server side and is more scalable in nature, as ASP.NET removes objects if the memory becomes scarce. This also makes it unreliable in some cases. Cache objects can have expiration polices set on them and is shared across users.

Catching has two types of expiration as in the following:

- Absolute expiration means in caching the cache memory will be deleted after a fixed amount of time, the same as for cookies

```
Cache.Insert("Uname", txtUser.Text, null, DateTime.Now.AddSeconds(20),
TimeSpan.Zero);
```

- Sliding expiration means in caching the cache memory will be deleted after a fixed amount of time if that memory is not in use, if it is used in that specific time then it can be relatively increased. the same as for a session.

```
Cache.Insert("Uname", txtUser.Text, null, DateTime.MaxValue,
TimeSpan.FromSeconds(20));
```

The following Example shows as another way to store and retrieve data in cache

```
Cache[“UserName”]= “Sachin”
lblUserName.Text = Cache[“UserName”].ToString();
```

4.2 Web Applications using ASP.NET Ajax:

- AJAX stands for Asynchronous JavaScript and XML. In other words, Ajax is the combination of various technologies such as a JavaScript, CSS, XHTML, and DOM etc.
- AJAX allows web pages to be updated asynchronously by exchanging small amounts of data with the server behind the scenes. This means that it is possible to update parts of a web page, without reloading the entire page.
- We can also define Ajax is a combination of client-side technologies that provides asynchronous communication between the user interface and the web server so that partial page rendering occurs instead of complete page post back.

- Ajax is platform-independent; in other words, AJAX is a cross-platform technology that can be used on any Operating System since it is based on XML & JavaScript. It also supports open-source implementation of other technologies.

The controls provided by ASP.NET for having AJAX functionality are:

1. Using ScriptManager:

   When we use any Ajax control then there is a requirement to use the ScriptManager to handle the Scripting on the client side; without the ScriptManager Ajax controls are not run. So, it's a requirement to use the ScriptManager.
2. Using UpdatePanel:

   Update panel is one of the most commonly used Ajax controls which is responsible for updating the requested content of the page instead of the entire page which is not requested. The ASP.Net controls are put under the update panel to make the benefit of this control. The ASP.Net controls which are kept under the update panel will be updated when the user clicks on a particular ASP.Net Control which is used in an application.

   We can use multiple update panels on a single web page.
3. Using Timer:

   The Timer is also one of the important controls; by using it we can update the content of the update panel area automatically without clicking the browsers refresh button.

   The Timer control is used along with the Update Panel, so the Contents put under the update panel are automatically updated according to the timing set under the timer_click event.
4. Using Update Progress:

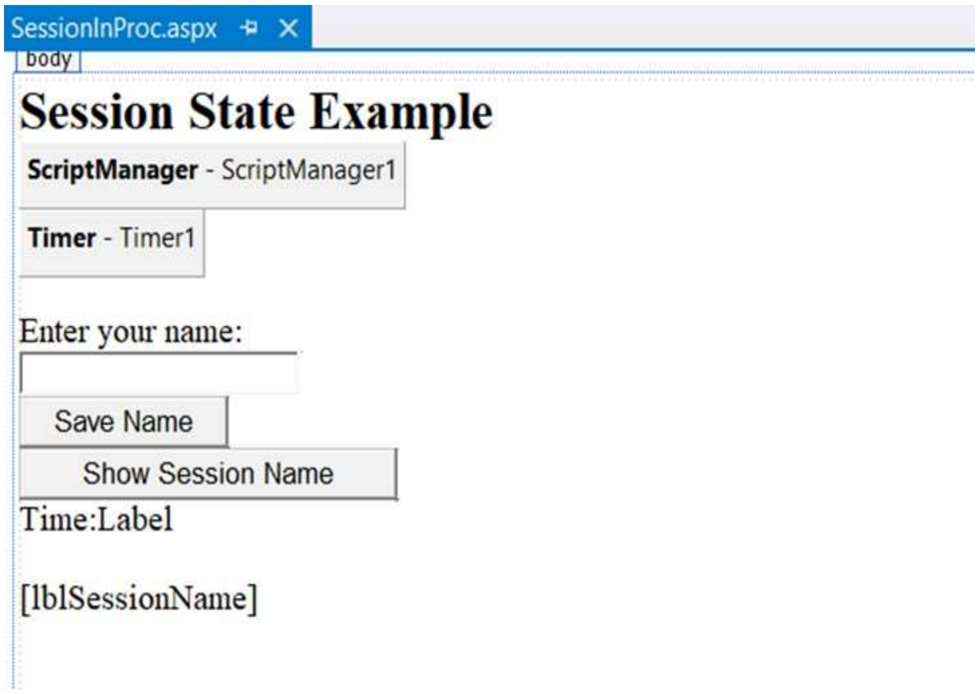
   This control is used to notify the user to wait until the requests are processed on the server. Update progress control is used when the user clicks on any tab or control of an application. At that time the progress bar is shown which is the interval between the times taken by the server to process the client request.
5. Using ScriptManagerProxy :

   When you need to reference a service from your content page and yet the ScriptManager resides on the Master Page use a ScriptManagerProxy. The ScriptManagerProxy works by detecting the main ScriptManager on your page at runtime and hooking itself to that ScriptManager, making sure that any references given to it are also given to the real ScriptManager.
6. Using Pointer:

   This Ajax Control used to specify the style of the mouse pointer such as arrow, thumb, and progress bar and much more. The above is the basic introduction about the Ajax.

Exercise:

1. **Design Web application to maintain the session data using In-proc session state mode.**



Code:

SessionProc.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="SessionInProc.aspx.cs" Inherits="Practical_04.SessionInProc" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
<div>
<h2>Session State Example<asp:ScriptManager ID="ScriptManager1" runat="server">
</asp:ScriptManager>
<asp:Timer ID="Timer1" runat="server" OnTick="Timer1_Tick">
</asp:Timer>
```

```
</h2>

<asp:Label ID="lblMessage" runat="server" Text="Enter your name:" /><br />
<asp:TextBox ID="txtName" runat="server" /><br />
<asp:Button ID="btnSave" runat="server" Text="Save Name" OnClick="btnSave_Click"
/><br />
<asp:Button ID="btnShow" runat="server" Text="Show Session Name"
OnClick="btnShow_Click" /><br />

Time:<asp:Label ID="lblDigitalClock" runat="server" Text="Label"></asp:Label>
<br />
<br />

<asp:Label ID="lblSessionName" runat="server" Text="" />
</div>
<p>
&nbsp;</p>
</form>
</body>
</html><%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="SessionInProc.aspx.cs" Inherits="Practical_04.SessionInProc" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
<div>
<h2>Session State Example<asp:ScriptManager ID="ScriptManager1" runat="server">
</asp:ScriptManager>
<asp:Timer ID="Timer1" runat="server" OnTick="Timer1_Tick">
</asp:Timer>
</h2>
```

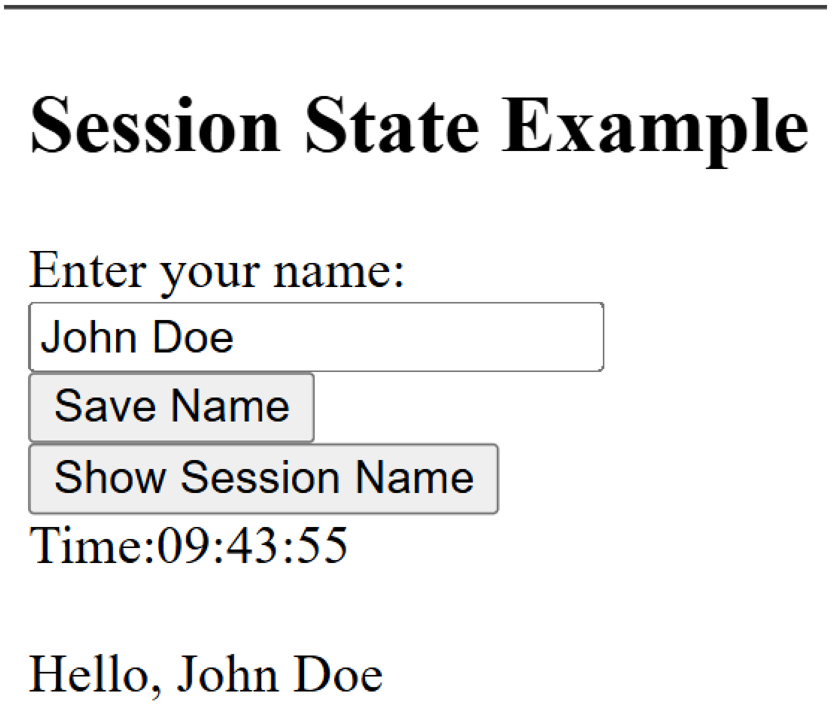
```
<asp:Label ID="lblMessage" runat="server" Text="Enter your name:" /><br />
<asp:TextBox ID="txtName" runat="server" /><br />
<asp:Button ID="btnSave" runat="server" Text="Save Name" OnClick="btnSave_Click"
/><br />
<asp:Button ID="btnShow" runat="server" Text="Show Session Name"
OnClick="btnShow_Click" /><br />

Time:<asp:Label ID="lblDigitalClock" runat="server" Text="Label"></asp:Label>
<br />
<br />

<asp:Label ID="lblSessionName" runat="server" Text="" />
</div>
<p>
&nbsp;</p>
</form>
</body>
</html>
```

SessionInProc.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class SessionInProc : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                lblDigitalClock.Text = DateTime.Now.ToString("HH:mm:ss");
            }
            if (Session["UserName"] != null)
            {
                lblSessionName.Text = "Hello, " + Session["UserName"].ToString();
            }
            else
            {
                lblSessionName.Text = "No session data found.";
            }
        }
        protected void btnSave_Click(object sender, EventArgs e)
        {

            string userName = txtName.Text; Session["UserName"] = userName;

            lblSessionName.Text = "Name saved to session.";
        }

        protected void btnShow_Click(object sender, EventArgs e)
        {

            if (Session["UserName"] != null)
            {
                lblSessionName.Text = "Hello, " + Session["UserName"].ToString();
            }
            else
            {
                lblSessionName.Text = "Session data is not available.";
            }
        }

        protected void Timer1_Tick(object sender, EventArgs e)
        {
            lblDigitalClock.Text = DateTime.Now.ToString("HH:mm:ss");
        }
```

```
    }
} using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class SessionInProc : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                lblDigitalClock.Text = DateTime.Now.ToString("HH:mm:ss");
            }
            if (Session["UserName"] != null)
            {
                lblSessionName.Text = "Hello, " + Session["UserName"].ToString();
            }
            else
            {
                lblSessionName.Text = "No session data found.";
            }
        }
        protected void btnSave_Click(object sender, EventArgs e)
        {

            string userName = txtName.Text; Session["UserName"] = userName;

            lblSessionName.Text = "Name saved to session.";
        }

        protected void btnShow_Click(object sender, EventArgs e)
        {

            if (Session["UserName"] != null)
            {
                lblSessionName.Text = "Hello, " + Session["UserName"].ToString();
            }
            else
            {
                lblSessionName.Text = "Session data is not available.";
            }
        }

        protected void Timer1_Tick(object sender, EventArgs e)
        {
            lblDigitalClock.Text = DateTime.Now.ToString("HH:mm:ss");
        }
    }
}
```

Output:



Session State Example

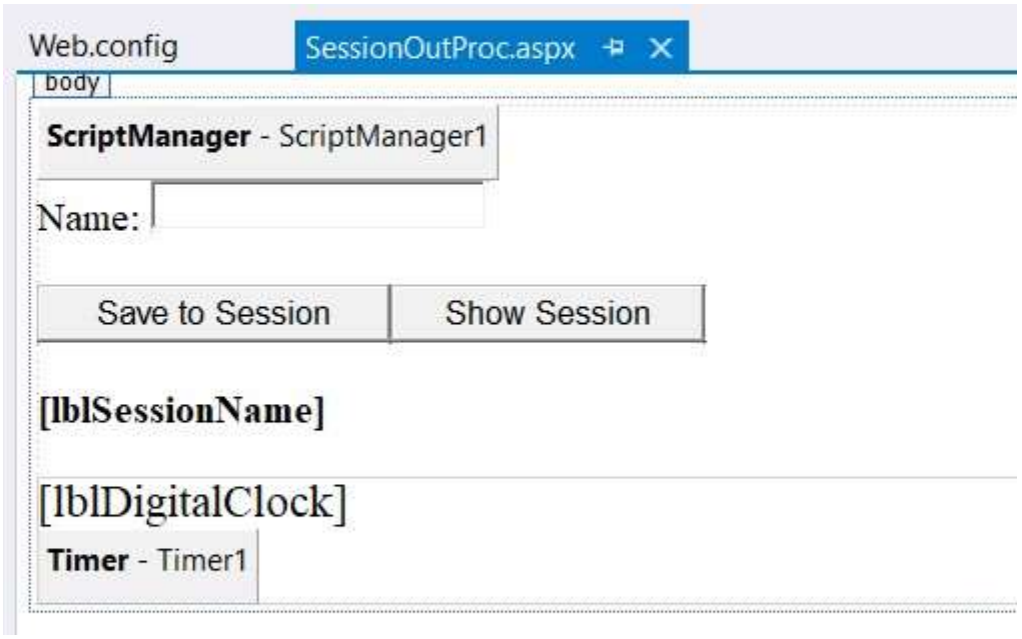
Enter your name:

John Doe

Time:09:43:55

Hello, John Doe

2. **Design Web application to maintain the session data using Out of process session state mode (Use StateServer and SQLServer modes).**



Code:

SessionOutProc.aspx:

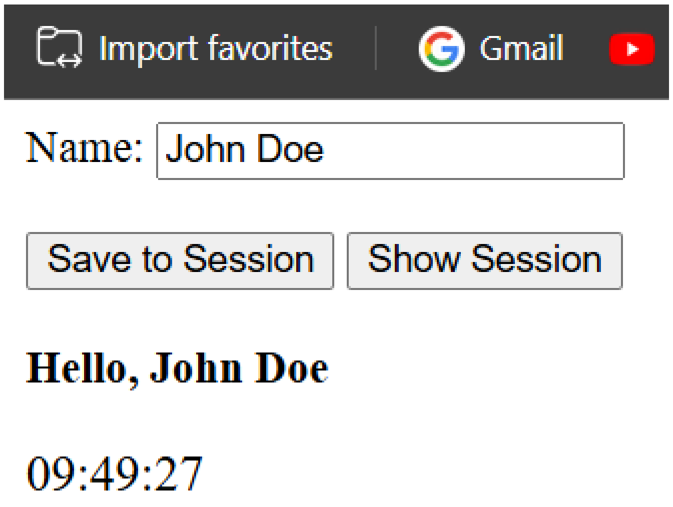
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="SessionOutProc.aspx.cs" Inherits="Practical_04.SessionOutProc" %>

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head>
    <title>Out-of-Proc Session Demo (SQL Server)</title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:ScriptManager ID="ScriptManager1" runat="server" />

        <div>
            Name: <asp:TextBox ID="txtName" runat="server"></asp:TextBox><br /><br />
            <asp:Button ID="btnSave" runat="server" Text="Save to Session"
OnClick="btnSave_Click" />
            <asp:Button ID="btnShow" runat="server" Text="Show Session"
OnClick="btnShow_Click" /><br /><br />
```

```
            <asp:Label ID="lblSessionName" runat="server" Font-
Bold="true"></asp:Label><br /><br />

            <asp:UpdatePanel ID="UpdatePanel1" runat="server">
                <ContentTemplate>
                    <asp:Label ID="lblDigitalClock" runat="server" Font-
Size="Large"></asp:Label><br />
                    <asp:Timer ID="Timer1" runat="server" Interval="1000"
OnTick="Timer1_Tick" />
                </ContentTemplate>
            </asp:UpdatePanel>
        </div>
    </form>
</body>
</html>
```

SessionOutProc.aspx.cs:

```
using System;
using System.Web.UI;

namespace Practical_04
{
    public partial class SessionOutProc : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                lblDigitalClock.Text = DateTime.Now.ToString("HH:mm:ss");
            }

            if (Session["UserName"] != null)
            {
                lblSessionName.Text = "Hello, " + Session["UserName"].ToString();
            }
            else
            {
                lblSessionName.Text = "No session data found.";
            }
        }

        protected void btnSave_Click(object sender, EventArgs e)
        {
            string userName = txtName.Text;
            Session["UserName"] = userName;
            lblSessionName.Text = "Name saved to session.";
        }

        protected void btnShow_Click(object sender, EventArgs e)
        {
            if (Session["UserName"] != null)
            {
                lblSessionName.Text = "Hello, " + Session["UserName"].ToString();
            }
            else
            {
                lblSessionName.Text = "Session data is not available.";
            }
        }

        protected void Timer1_Tick(object sender, EventArgs e)
        {
            lblDigitalClock.Text = DateTime.Now.ToString("HH:mm:ss");
        }
    }
}
```

Output:

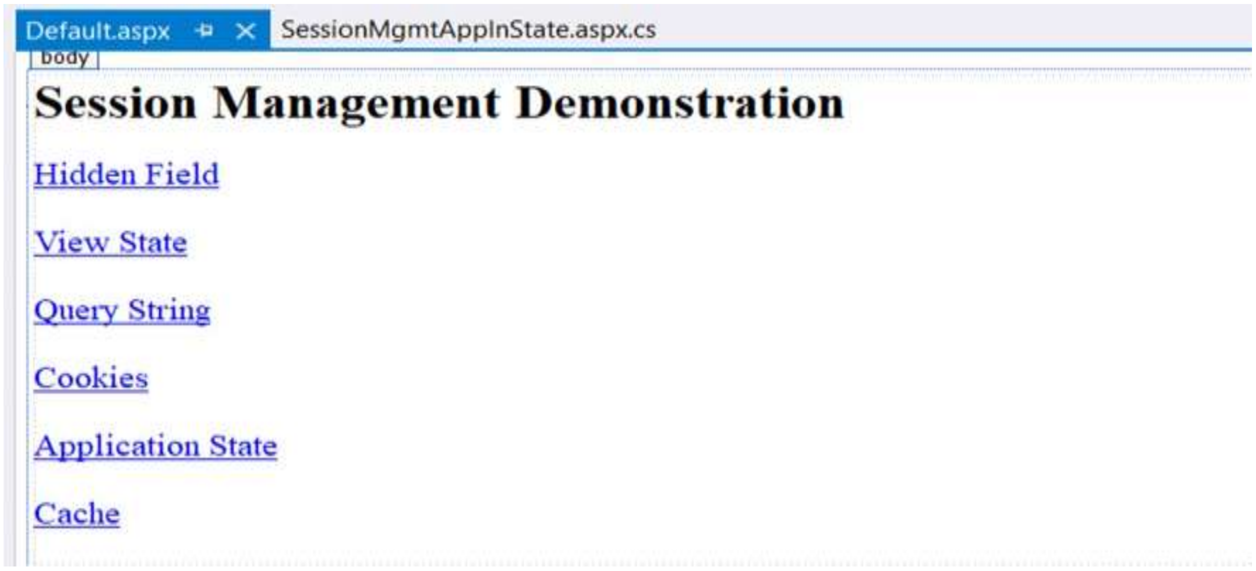


Name: John Doe

Hello, John Doe

09:49:27

3. **Design an ASP.NET web application to display the number of times the current page is visited using the following session management techniques:**
   a. **Hidden Field** **d. Cookies**
   b. **View State** **e. Application State**
   c. **QueryString** **f. Cache**



Session Management Demonstration

Hidden Field

View State

Query String

Cookies

Application State

Cache

Code:

Global.ajax :

using System;

```
using
System.Web;
namespace Practical_04
{
            public class Global : System.Web.HttpApplication
    {
              protected void Application_Start(object sender, EventArgs e)
    {
      Application["SiteVisiterCounter"] = 0;
      Application["OnlineUserCounter"] = 0;
    }

    protected void Session_Start(object sender, EventArgs e)
    {
      Application.Lock();
    Application["SiteVisiterCounter"] =
    Convert.ToInt32(Application["SiteVisiterCounter"]) + 1;
     Application["OnlineUserCounter"] =
    Convert.ToInt32(Application["OnlineUserCounter"]) + 1;
      Application.UnLock();
    }
               protected void Application_BeginRequest(object sender, EventArgs e)


{

}
    protected void Application_AuthenticateRequest(object sender, EventArgs e)          {
    }
              protected void Application_Error(object sender, EventArgs e)


{

}
             protected void Session_End(object sender, EventArgs e)
    {
      Application.Lock();
      int onlineUsers = Convert.ToInt32(Application["OnlineUserCounter"]);
Application["OnlineUserCounter"] = (onlineUsers > 0) ? (onlineUsers - 1) : 0;
Application.UnLock();
    }
```

```
                protected void Application_End(object sender, EventArgs e)
        {
        }
        }
}
```

CacheDemo1.aspx.cs :

```
using System;
using System.Collections.Generic;
using System.Linq; using
System.Web; using
System.Web.UI; using
System.Web.UI.WebControls;
namespace Practical_04
{
    public partial class CacheDemo1 : System.Web.UI.Page
        {
         protected void Page_Load(object sender, EventArgs e)
         {

         }
         protected void btnSetData_Click(object sender, EventArgs e)
         {
             Cache["FirstName"] = txtName.Text;
             Cache["LastName"] = txtLName.Text;

             Response.Redirect("CacheDemo2.aspx");
         }
        }
}
```

SessionMgmt_Cookies.aspx.cs : using

```
System;
using System.Collections.Generic;
using System.Linq; using
System.Web; using
System.Web.UI; using
System.Web.UI.WebControls;
namespace Practical_04
{
    public partial class SessionMgmt_Cookies : System.Web.UI.Page
        {
```

```
        int counter = 0;
        protected void Page_Load(object sender, EventArgs e)
        {
          if(Request.Cookies["counter"]==null)
            {
             counter = 1;

}
else
            {
             counter = int.Parse(Request.Cookies["counter"].Value) + 1;
            }
            Response.Cookies["counter"].Value = counter.ToString();
            Response.Cookies["counter"].Expires = DateTime.Now.AddSeconds(5);
if(Request.Cookies["counter"]!=null)
            {
             lblCounter.Text = Request.Cookies["counter"].Value;
            }
        }
       }
}
```

<u>SessionMgmt_HiddenFields.aspx.cs :</u>

```
using System;
using
System.Collections.Generic;
using System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class SessionMgmt_HiddenFields : System.Web.UI.Page
       {
        protected void Page_Load(object sender, EventArgs e)
        {
             if(!IsPostBack)
            {
             hidFieldCount.Value = 1.ToString();

}
else
```

```
            {
             int visitCount = int.Parse(hidFieldCount.Value);
visitCount++;
             hidFieldCount.Value = visitCount.ToString();
            }
            lblVisitCount.Text = string.Format("Clicked {0} times!", hidFieldCount.Value);
        }
       }
}
```

SessionMgmt_QueryString.aspx.cs :

```
using System;
using
System.Collections.Generic;
using System.Linq; using
System.Web; using
System.Web.UI;
using System.Web.UI.WebControls; namespace
Practical_04
{
    public partial class SessionMgmt_QueryString : System.Web.UI.Page
       {
        int cnt = 0;
        protected void Page_Load(object sender, EventArgs e)
        {
            cnt = Convert.ToInt32(Server.UrlDecode(Request.QueryString["count"]));

            if(cnt !=0)
            {
             cnt = cnt + 1;
            }
else
{
cnt = 1;
            }

            lblCounter.Text = Convert.ToString(cnt);
            lblMessage.Text = "Welcome " + Request.QueryString["name"];
        }
        protected void btnSubmit_Click(object sender, EventArgs e)
        {
            Response.Redirect("SessionMgmt_QueryString.aspx?count=" +
Server.UrlEncode((lblCounter.Text))+ "&&name=sanika");
```

```
         }
        }
}
```

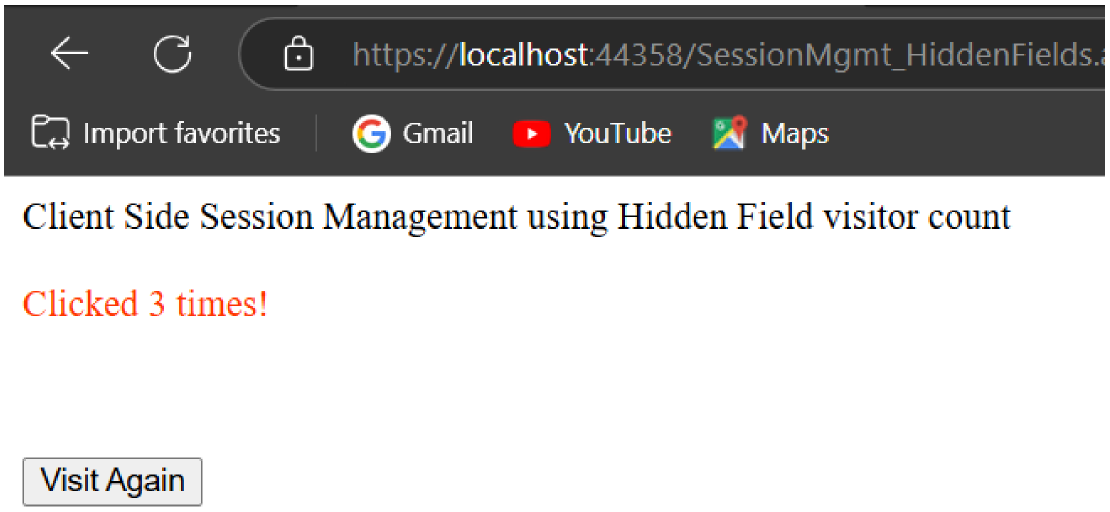
SessionMgmt_ViewState.aspx.cs :

```
using System;
using System.Collections.Generic;
using System.Linq; using
System.Web; using
System.Web.UI; using
System.Web.UI.WebControls;
namespace Practical_04
{
    public partial class SessionMgmt_ViewState : System.Web.UI.Page
       {
        int counter;
        protected void Page_Load(object sender, EventArgs e)
        {
            if(! IsPostBack)
            {
             counter = 1;

}
else
            {
             counter = (int)ViewState["Counter"];
             counter++;
            }
            ViewState["Counter"] = counter;
            lblCounter.Text = "Counter : " + ViewState["Counter"];
        }
        protected void btnVisit_Click(object sender, EventArgs e)
        {

        }
       }
}
```

SessionMgmtApplnState.aspx.cs :

```
using System;
using System.Collections.Generic;
using System.Linq; using
System.Web; using
System.Web.UI; using
System.Web.UI.WebControls;
namespace Practical_04
{
    public partial class SessionMgmtApplnState : System.Web.UI.Page
      {
       protected void Page_Load(object sender, EventArgs e)
       {
                lblVisitorsCount.Text = "No. of times site visited = " +
      Application["SiteVisiterCounter"].ToString();
              lblOnlineVisitorsCount.Text = "No. of users online on the site = " +
      Application["OnlineUserCounter"].ToString();
       }
       protected void Btn_Clear_Click(object sender, EventArgs e)
       {
          Session.Abandon();
       }
      }
}
```

Output :



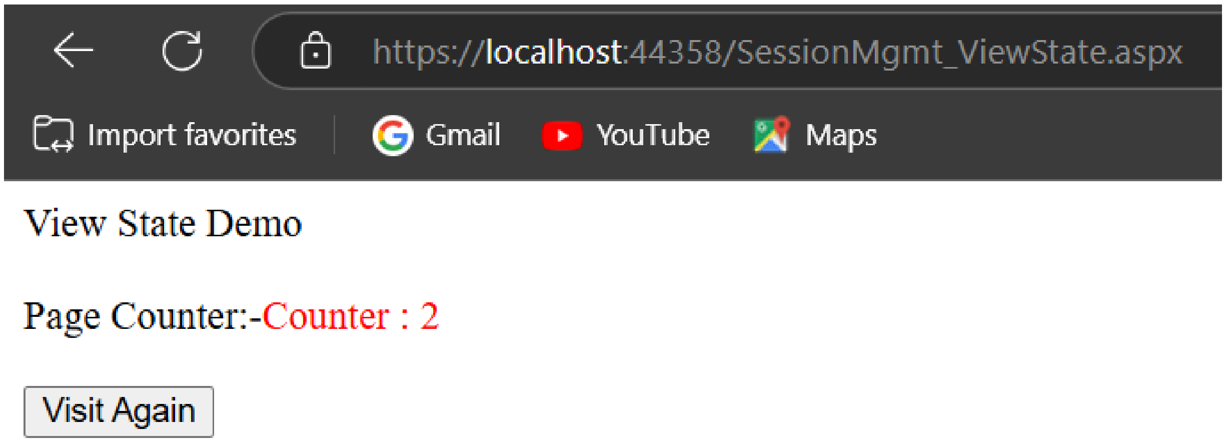
https://localhost:44358/SessionMgmt_HiddenFields.a

Import favorites | Gmail | YouTube | Maps

Client Side Session Management using Hidden Field visitor count

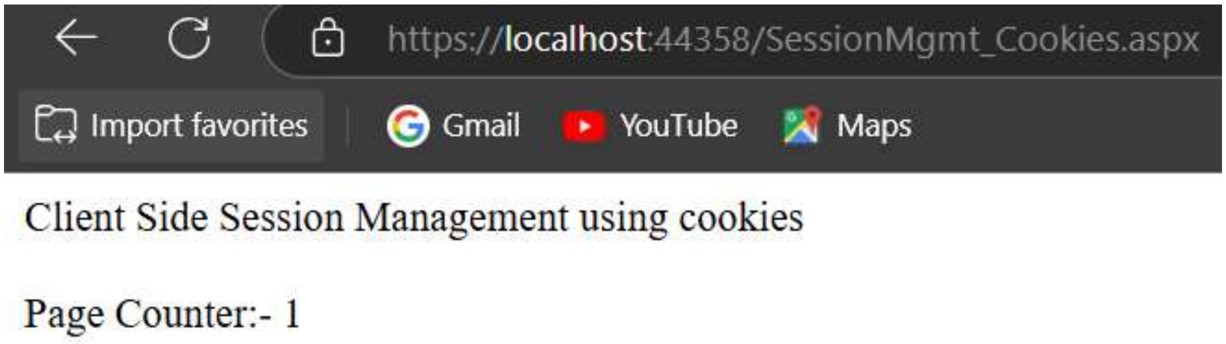
Clicked 3 times!

Visit Again



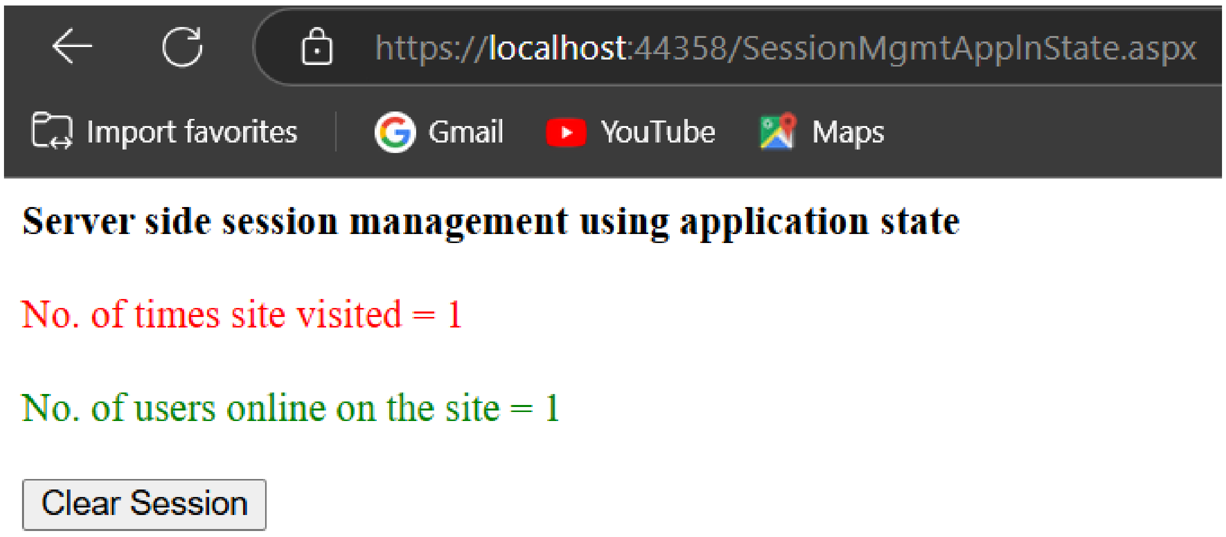
View State Demo

Page Counter:-Counter : 2



Client Side Session Management using cookies

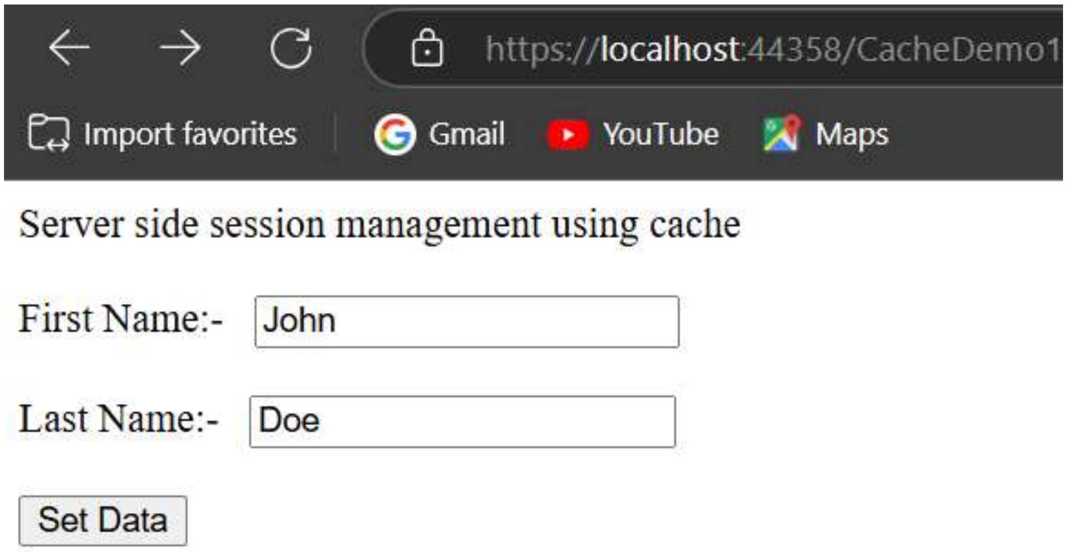
Page Counter:- 1



Server side session management using application state

No. of times site visited = 1

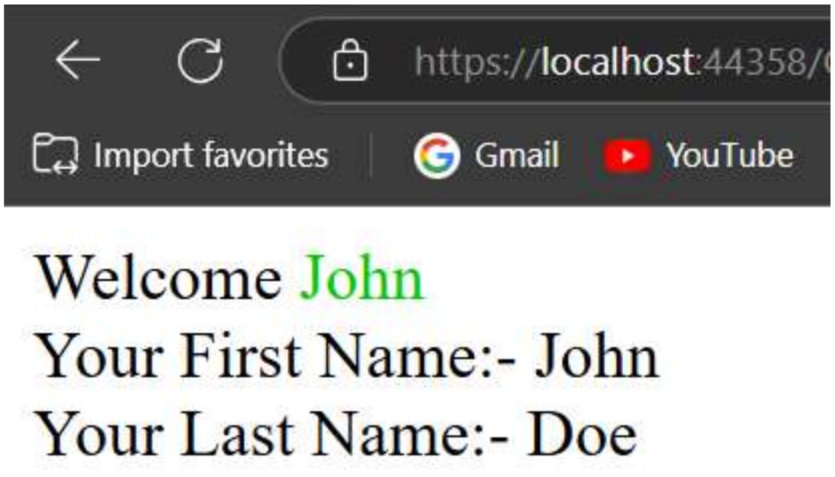
No. of users online on the site = 1



Server side session management using cache

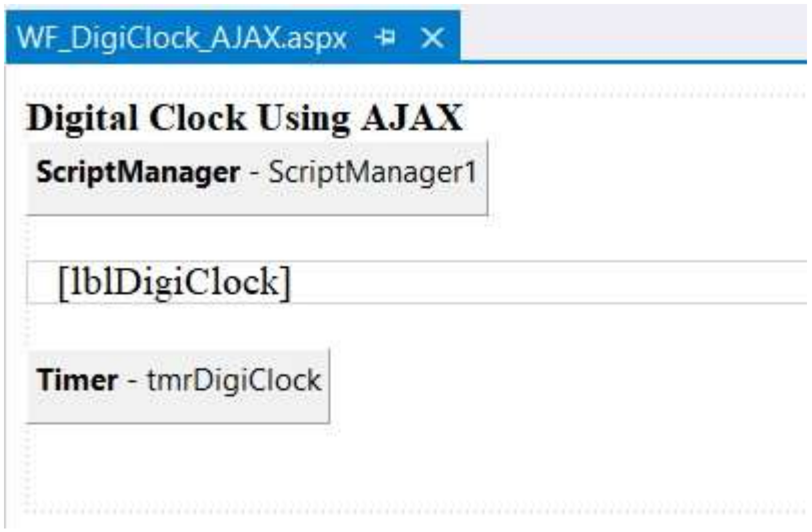
First Name:- John

Last Name:- Doe



Welcome John
Your First Name:- John
Your Last Name:- Doe

4. **Design an ASP.NET web form to display digital clock using AJAX.**



Digital Clock Using AJAX
ScriptManager - ScriptManager1
[lblDigiClock]
Timer - tmrDigiClock

Code:

WF_DigiClock_AJAX.aspx:

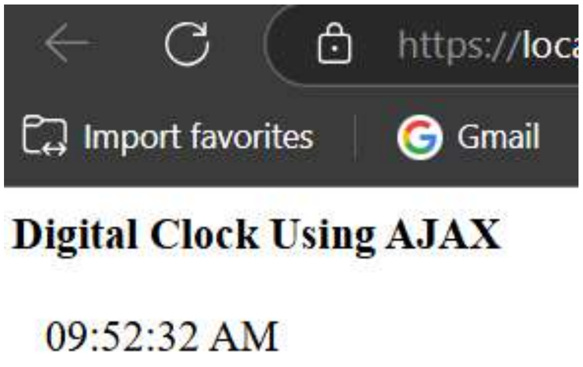
```
 <%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WF_DigiClock_AJAX.aspx.cs" Inherits="Practical_04.WF_AJAX" %>

<!DOCTYPE html>
```

```
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="Label1" runat="server" Font-Bold="True" Text="Digital Clock
Using AJAX"></asp:Label>
            <br />
            <asp:ScriptManager ID="ScriptManager1" runat="server">
            </asp:ScriptManager>
            <br />
            <asp:UpdatePanel ID="UpdatePanel1" runat="server">
                <ContentTemplate>
                    &nbsp;&nbsp;
                    <asp:Label ID="lblDigiClock" runat="server"></asp:Label>
                    &nbsp;
                </ContentTemplate>
                <Triggers>
                    <asp:AsyncPostBackTrigger ControlID="tmrDigiClock" EventName="Tick"
/>
                </Triggers>
            </asp:UpdatePanel>
            <br />
            <asp:Timer ID="tmrDigiClock" runat="server" Interval="1000"
OnTick="tmrDigiClock_Tick">
            </asp:Timer>
            <br />
            <br />
        </div>
    </form>
</body>
</html>
```

WF_DigiClock_AJAX.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class WF_AJAX : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void tmrDigiClock_Tick(object sender, EventArgs e)
        {
            lblDigiClock.Text = DateTime.Now.ToString("hh:mm:ss tt");
        }
    }
}
```

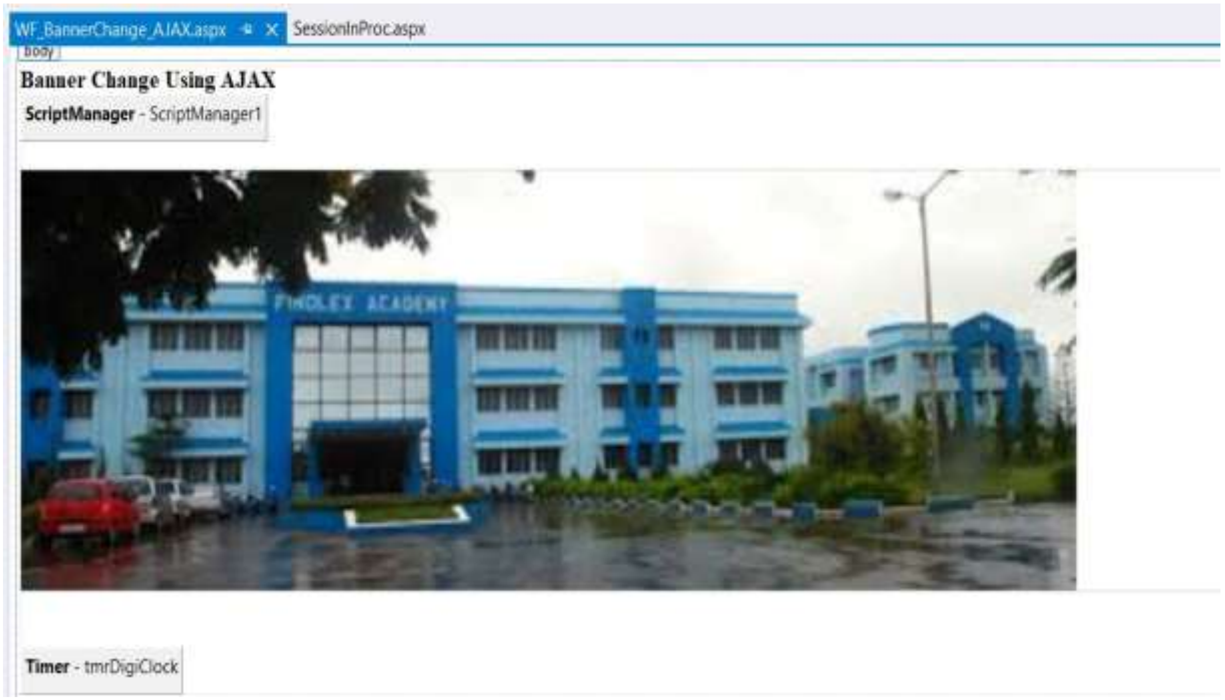
Output:



Digital Clock Using AJAX

09:52:32 AM

5. **Design an ASP.NET web form using AJAX controls to change the banner image of your college website every 10 seconds.**



Code:

WF_BannerChange_AJAX.aspx:

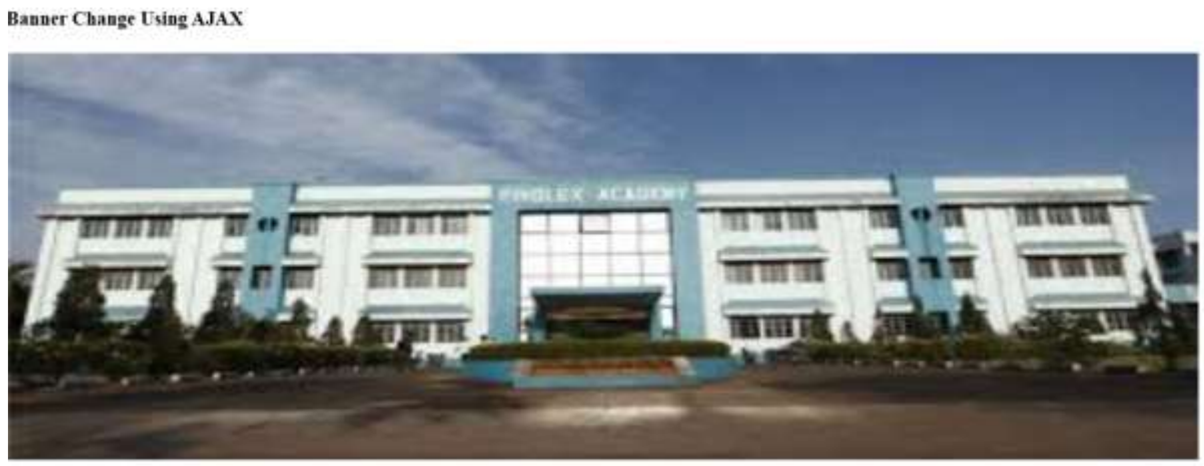
```
 <%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WF_BannerChange_AJAX.aspx.cs"
Inherits="Practical_04.WF_BannerChange" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:Label ID="Label1" runat="server" Font-Bold="True" Text="Banner Change
Using AJAX"></asp:Label>
        <br />
        <asp:ScriptManager ID="ScriptManager1" runat="server">
        </asp:ScriptManager>
        <br />
        <asp:UpdatePanel ID="UpdatePanel1" runat="server">
```

```
            <ContentTemplate>
                <asp:Image ID="Image1" runat="server" Height="336px"
ImageUrl="~/images/banner_1.jpg" Width="986px" />
            </ContentTemplate>
        </asp:UpdatePanel>
        <br />
        <br />
        <asp:Timer ID="tmrDigiClock" runat="server" Interval="1000"
OnTick="tmrDigiClock_Tick">
        </asp:Timer>
    </form>
</body>
</html>
```

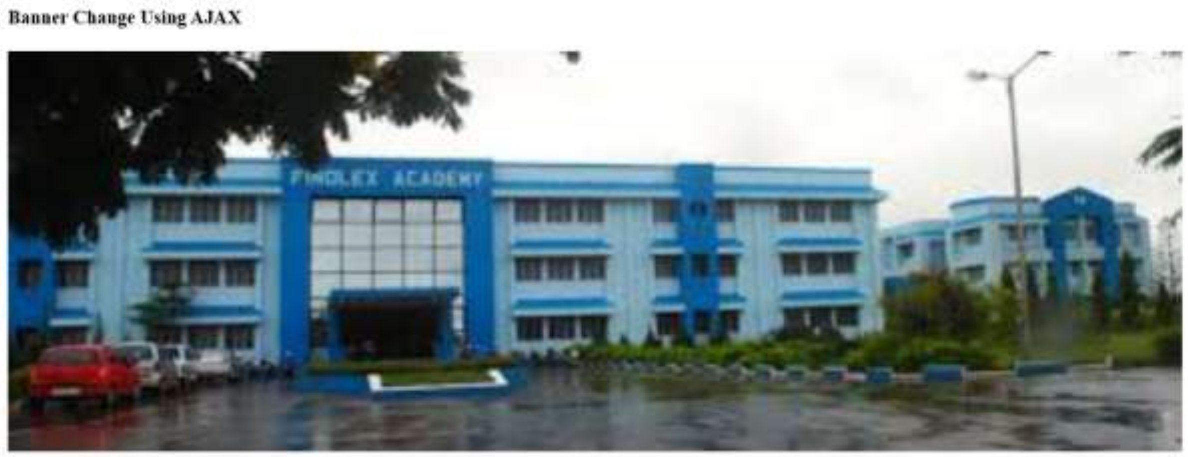
WF_BannerChange_AJAX.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class WF_BannerChange : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void tmrDigiClock_Tick(object sender, EventArgs e)
        {
            Random RandomNumber = new Random();

            int n = RandomNumber.Next(1, 5);
            Image1.ImageUrl = String.Concat("images/banner_", n.ToString(), ".jpg");
        }
    }
}
```

Output:

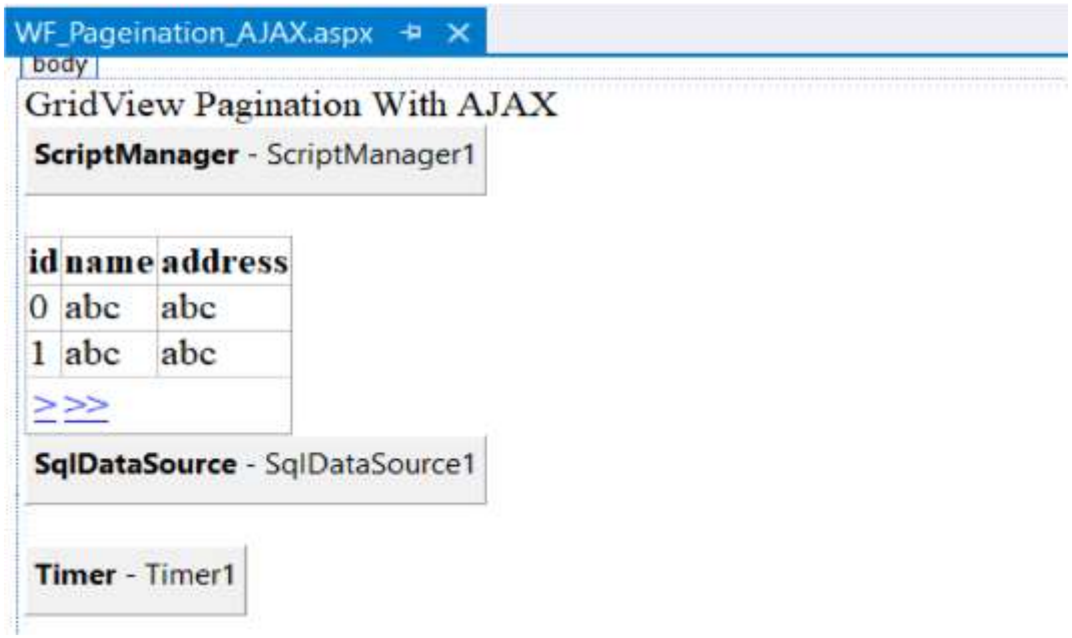


Banner Change Using AJAX



Banner Change Using AJAX

6. **Design an ASP.NET web form to demonstrate UpdatePanel and UpdateProgress AJAX controls.**



GridView Pagination With AJAX

ScriptManager - ScriptManager1

id	name	address
0	abc	abc
1	abc	abc
> >>		

SqlDataSource - SqlDataSource1

Timer - Timer1

Code:

WF_UpdateProgress_AJAX.aspx:

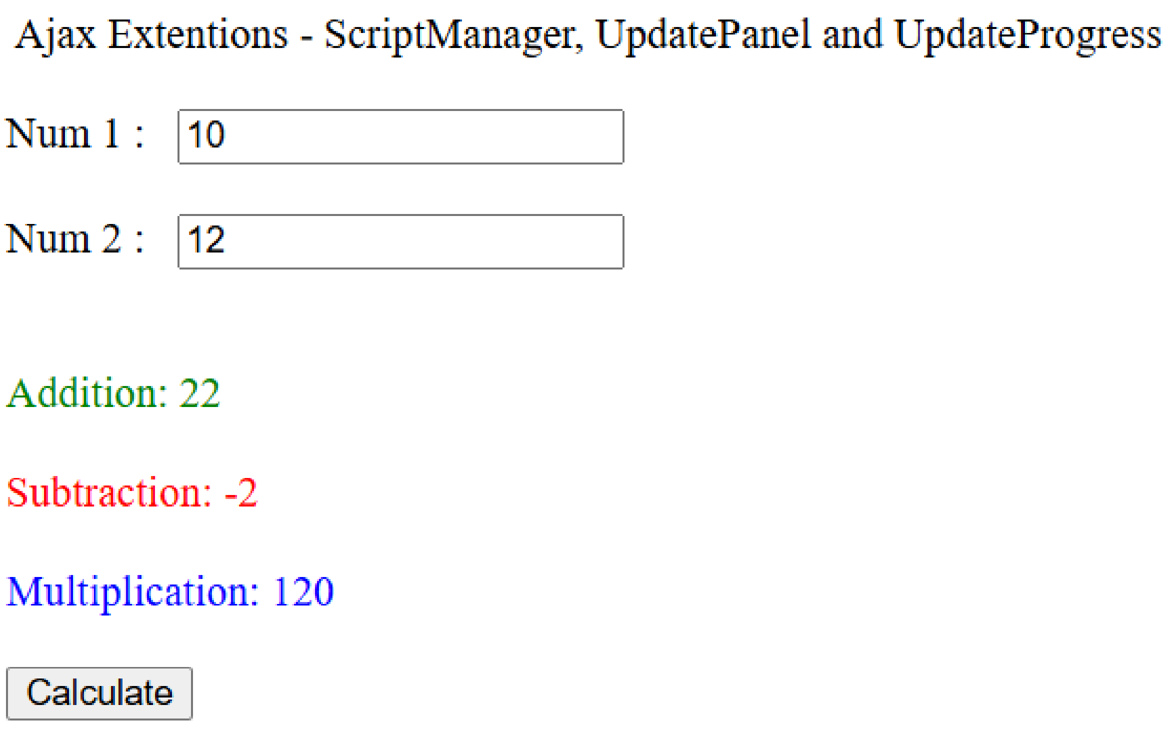
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WF_UpdateProgress_AJAX.aspx.cs"
Inherits="Practical_04.WF_UpdateProgress_AJAX" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <asp:UpdatePanel ID="UpdatePanel1" runat="server">
            <ContentTemplate>
                &nbsp;<asp:Label ID="Label4" runat="server" Text="Ajax Extentions -
ScriptManager, UpdatePanel and UpdateProgress"></asp:Label>
                <br />
                <asp:ScriptManager ID="ScriptManager2" runat="server">
                </asp:ScriptManager>
                <br />
                <asp:Label ID="lbl_num1" runat="server" Text="Num 1 :"></asp:Label>
                &nbsp;
                <asp:TextBox ID="txt_num1" runat="server"></asp:TextBox>
                <br />
                <br />
                <asp:Label ID="Label6" runat="server" Text="Num 2 :"></asp:Label>
                &nbsp;
                <asp:TextBox ID="txt_num2" runat="server"></asp:TextBox>
                <br />
                <br />
                <asp:UpdateProgress ID="UpdateProgress1" runat="server">
                    <ProgressTemplate>
                        Calculating please wait....
                    </ProgressTemplate>
                </asp:UpdateProgress>
                <br />
                <asp:Label ID="Label7" runat="server" ForeColor="Green"
Text="Label"></asp:Label>
                <br />
                <br />
                <asp:Label ID="Label8" runat="server" ForeColor="Red"
Text="Label"></asp:Label>
                <br />
                <br />
                <asp:Label ID="Label9" runat="server" ForeColor="Blue"
Text="Label"></asp:Label>
                <br />
                <br />
                <asp:Button ID="btn_Calc" runat="server" OnClick="btn_Calc_Click"
Text="Calculate" />
            </ContentTemplate>
        </asp:UpdatePanel>
    </form>
</body>
</html>
```

WF_UpdateProgress_AJAX.aspx.cs:

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class WF_UpdateProgress_AJAX : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void btn_Calc_Click(object sender, EventArgs e)
        {
            try
            {
                System.Threading.Thread.Sleep(1000);
                if (int.TryParse(txt_num1.Text, out int num1) && int.TryParse(txt_num2.Text,
out int num2))
                {
                    int sum = num1 + num2;
                    int diff = num1 - num2;
                    int product = num1 * num2;

                    Label7.Text = "Addition: " + sum;
                    Label8.Text = "Subtraction: " + diff;
                    Label9.Text = "Multiplication: " + product;
                }
                else
                {
```

```
                Label7.Text = "Invalid input!";
                Label8.Text = "";
                Label9.Text = "";
            }
        }
        catch(Exception ex)
        {

        }
    }
}
}
```

Output:


Ajax Extentions - ScriptManager, UpdatePanel and UpdateProgress

Num 1 : 10

Num 2 : 12

Addition: 22

Subtraction: -2

Multiplication: 120

7. **Design an ASP.NET web form to show paginated output of customer_info table in GridView.**

Code:

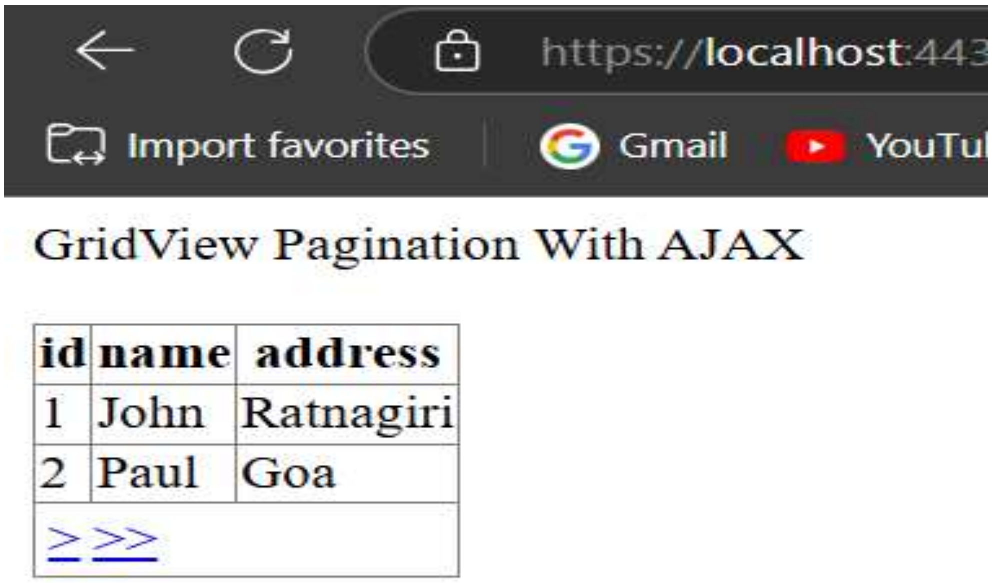
WF_Pageination_AJAX.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="WF_Pageination_AJAX.aspx.cs"
Inherits="Practical_04.WF_Pageinator_AJAX" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="Label1" runat="server" Text="GridView Pagination With
AJAX"></asp:Label>
            <br />
            <asp:ScriptManager ID="ScriptManager1" runat="server">
            </asp:ScriptManager>
            <br />
            <asp:GridView ID="GridView1" runat="server" AllowPaging="True"
AutoGenerateColumns="False" DataKeyNames="id" DataSourceID="SqlDataSource1"
PageSize="2">
                <Columns>
                    <asp:BoundField DataField="id" HeaderText="id" ReadOnly="True"
SortExpression="id" />
                    <asp:BoundField DataField="name" HeaderText="name"
SortExpression="name" />
                    <asp:BoundField DataField="address" HeaderText="address"
SortExpression="address" />
                </Columns>
                <PagerSettings Mode="NextPreviousFirstLast" />
            </asp:GridView>
            <asp:SqlDataSource ID="SqlDataSource1" runat="server" ConnectionString="<%$
ConnectionStrings:StudDBConnectionString %>" SelectCommand="SELECT * FROM
[stud]"></asp:SqlDataSource>
            <br />
            <asp:Timer ID="Timer1" runat="server" OnTick="Timer1_Tick">
            </asp:Timer>
            <br />
        </div>
    </form>
</body>
</html>
```

WF_Pageination_AJAX.aspx.cs:

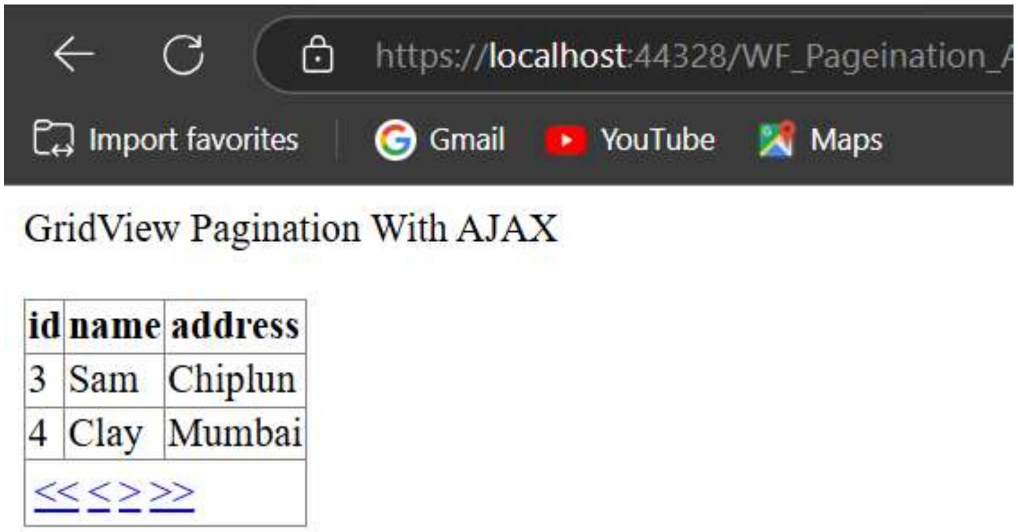
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_04
{
    public partial class WF_Pageinator_AJAX : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void Timer1_Tick(object sender, EventArgs e)
        {
            if (GridView1.PageIndex != GridView1.PageCount - 1)
            {
                GridView1.PageIndex = GridView1.PageIndex - 1;
            }
            else
            {
                GridView1.PageIndex = 0;
            }
        }
    }
}
```

Output:



GridView Pagination With AJAX

id	name	address
1	John	Ratnagiri
2	Paul	Goa
> >>		



GridView Pagination With AJAX

id	name	address
3	Sam	Chiplun
4	Clay	Mumbai
<< < > >>		