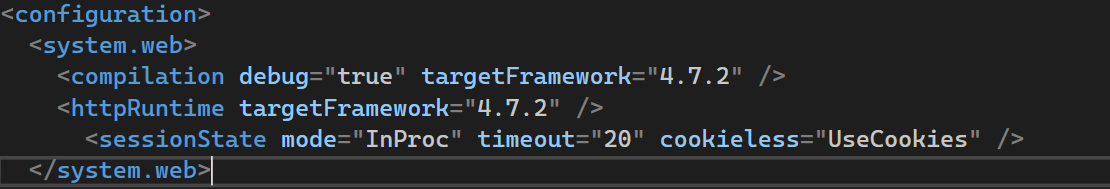
**Practical Number 12: Design Web Applications using Server Side Session Management Techniques**

**Steps to Create a Web Application with Session Management in Visual Studio**

**1. Create an ASP.NET Web Forms Project**

1. Open **Visual Studio**.
2. Click **Create a new project**.
3. Select **ASP.NET Web Application (.NET Framework)** and click **Next**.
4. Name the project (e.g., SessionManagementDemo).
5. Choose **Web Forms** and click **Create**.

**2. Configure Session Management in Web.config**



Modify Web.config to use **InProc (default)**, **StateServer**, or **SQL Server**.

For **InProc Session** (default):

<sessionState mode="InProc" timeout="20" cookieless="UseCookies" />

For **StateServer Session**:

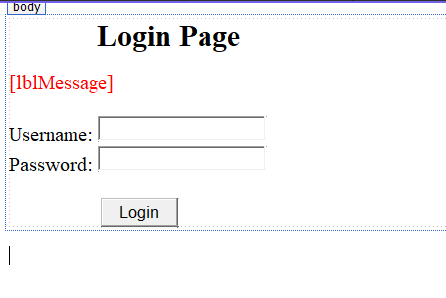
<sessionState mode="StateServer" stateConnectionString="tcpip=127.0.0.1:42424" timeout="20"/>

For **SQL Server Session**:

<sessionState mode="SQLServer" sqlConnectionString="data source=SQLSERVER;Initial Catalog=SessionDB;Integrated Security=True" timeout="20"/>

**3. Implement Login Page (Login.aspx)**

**Frontend: Login.aspx**

****

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Login.aspx.cs" Inherits="SessionManagementDemo.Login" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<title>Login</title>

</head>

<body>

<form id="form1" runat="server">

<div>

<h2>Login Page</h2>

<asp:Label ID="lblMessage" runat="server" ForeColor="Red"></asp:Label>

<br />

Username: <asp:TextBox ID="txtUsername" runat="server"></asp:TextBox>

<br />

Password: <asp:TextBox ID="txtPassword" runat="server" TextMode="Password"></asp:TextBox>

<br />

<asp:Button ID="btnLogin" runat="server" Text="Login" OnClick="btnLogin\_Click" />

</div>

</form>

</body>

</html>

**Backend: Login.aspx.cs**

using System;

using System.Web;

namespace SessionManagementDemo

{

public partial class Login : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] != null)

{

Response.Redirect("Dashboard.aspx");

}

}

protected void btnLogin\_Click(object sender, EventArgs e)

{

string username = txtUsername.Text;

string password = txtPassword.Text;

// Simulating user authentication

if (username == "admin" && password == "1234")

{

Session["Username"] = username;

Response.Redirect("Dashboard.aspx");

}

else

{

lblMessage.Text = "Invalid username or password!";

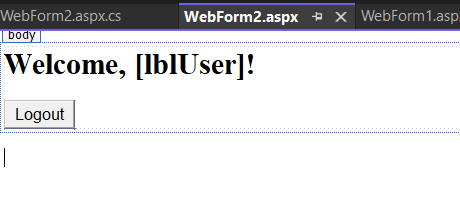
}

}

}

}

**4. Create a Dashboard Page (Dashboard.aspx)**

****

**Frontend: Dashboard.aspx**

<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Dashboard.aspx.cs" Inherits="SessionManagementDemo.Dashboard" %>

<!DOCTYPE html>

<html lang="en">

<head runat="server">

<title>Dashboard</title>

</head>

<body>

<form id="form1" runat="server">

<div>

<h2>Welcome, <asp:Label ID="lblUser" runat="server"></asp:Label>!</h2>

<asp:Button ID="btnLogout" runat="server" Text="Logout" OnClick="btnLogout\_Click" />

</div>

</form>

</body>

</html>

**Backend: Dashboard.aspx.cs**

using System;

using System.Web;

namespace SessionManagementDemo

{

public partial class Dashboard : System.Web.UI.Page

{

protected void Page\_Load(object sender, EventArgs e)

{

if (Session["Username"] == null)

{

Response.Redirect("Login.aspx");

}

else

{

lblUser.Text = Session["Username"].ToString();

}

}

protected void btnLogout\_Click(object sender, EventArgs e)

{

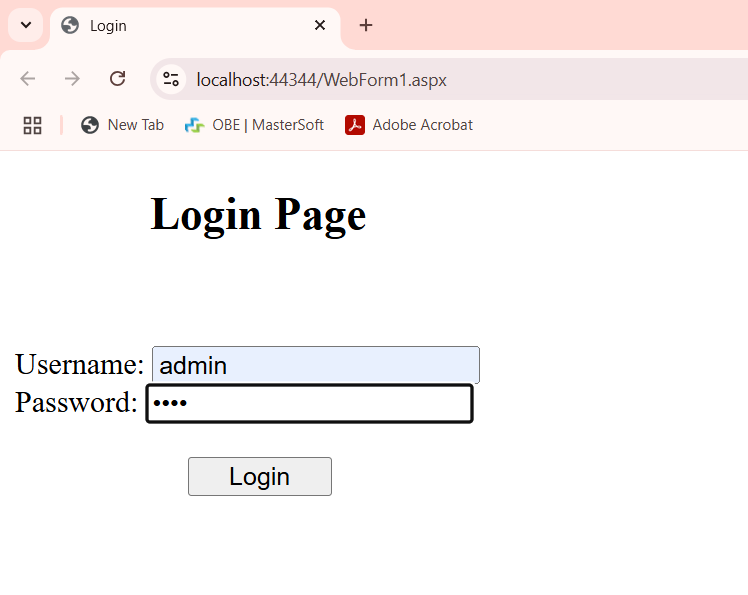
Session.Abandon();

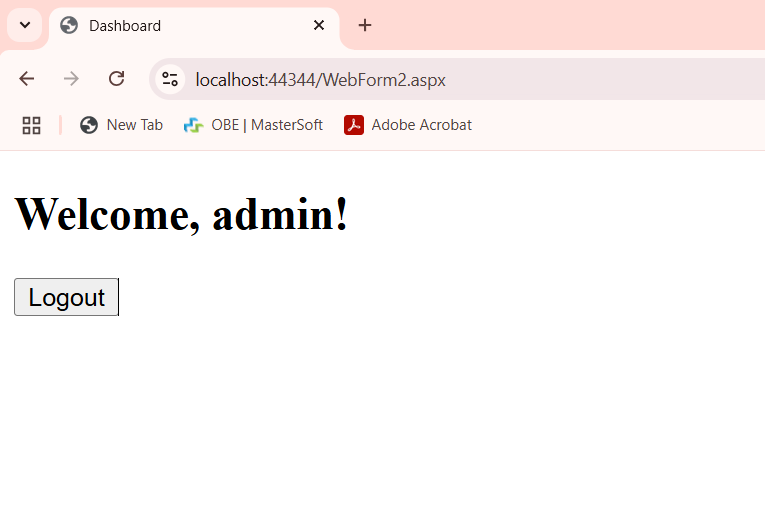
Response.Redirect("Login.aspx");

}

}

}





Difference between **InProc, StateServer, and SQL Server** session modes in **ASP.NET Web.config**:

| **Session Mode** | **Storage Location** | **Performance** | **Scalability** | **Reliability** | **Use Case** |
| --- | --- | --- | --- | --- | --- |
| **InProc (Default)** | Stores session data in **web server memory (RAM)** | **Fastest** (since it's in memory) | **Not scalable** (session lost if app restarts or crashes) | **Not reliable** (session lost if server restarts) | Best for **small applications** with a **single server** |
| **StateServer** | Stores session data in a **separate ASP.NET State Server (Windows Service)** | **Slower than InProc** (requires serialization) | **Scalable** (multiple web servers can use the same state server) | **More reliable** (session survives web server restarts) | Used in **web farms** where multiple servers share session data |
| **SQL Server** | Stores session data in a **SQL Server database** | **Slowest** (due to database access) | **Highly scalable** (multiple servers can share the same DB) | **Most reliable** (session persists even if the web server crashes) | Best for **large applications** needing **high availability** |

### ****When to Use Each Mode?****

* **Use InProc** → When performance is a priority and you are using a **single server**.
* **Use StateServer** → When using **multiple web servers (Web Farm)** and need better session persistence.
* **Use SQL Server** → When building a **large-scale, enterprise application** that requires **high availability and failover support**.