**Q.1 Creating a login page that only checks if the login detail entered by the user is correct or not and gives option to save new username and password.**

**Solution:**

Here we assume database is already created

Then, In .aspx file In the <Form> tag of the design page (.aspx) place two textbox controls and a Button and a label controls and design the log in page as:

<html>

<head runat="server">

<title>Login Form</title>

</head>

<body>

<form id="form1" runat="server">

<div>

<fieldset style="width:280px">

<legend>Login example in asp.net</legend>

<table>

<tr>

<td>User Name: \* </td>

<td>

<asp:TextBox ID="txtUserName" runat="server"></asp:TextBox><br />

<asp:RequiredFieldValidator ID="rfvUserName" runat="server" ErrorMessage="Please enter username" Display="Dynamic" SetFocusOnError="true" ForeColor="Red" ControlToValidate="txtUserName"></asp:RequiredFieldValidator>

</td>

</tr>

<tr>

<td>Password: \*</td>

<td>

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

<asp:RequiredFieldValidator ID="rfvPwd" runat="server" ErrorMessage="Please enter password" Display="Dynamic" SetFocusOnError="true" ForeColor="Red" ControlToValidate="txtPwd"></asp:RequiredFieldValidator>

</td>

</tr>

<tr>

<td>&nbsp;</td>

<td>

<asp:Button ID="btnLogin" runat="server" Text="Login" /></td>

</tr>

<tr>

<td>&nbsp;</td>

<td>

<asp:Label ID="lblStatus" runat="server" Text=""></asp:Label>

</td>

</tr>

</table>

</fieldset>

</div>

</form>

</body>

</html>

In the code behind file ( .aspx.vb) write the code on Login Button’s click event as:

**First include the following namespaces**

Imports System.Data

Imports System.Data.SqlClient

Imports System.Configuration

**Then write the code as:**

Protected Sub btnLogin\_Click(sender As Object, e As System.EventArgs) Handles btnLogin.Click

Dim dt As New DataTable()

Dim adp As New SqlDataAdapter()

Try

Dim con As New SqlConnection(ConfigurationManager.ConnectionStrings("conStr").ConnectionString)

Dim cmd As New SqlCommand("Login\_Check\_Sp", con)

cmd.CommandType = CommandType.StoredProcedure

cmd.Parameters.AddWithValue("@username", txtUserName.Text.Trim())

cmd.Parameters.AddWithValue("@pwd", txtPwd.Text.Trim())

adp.SelectCommand = cmd

adp.Fill(dt)

cmd.Dispose()

If dt.Rows.Count > 0 Then

lblStatus.Text = "Login Successfull"

'Or in show messagebox using ScriptManager.RegisterStartupScript(this, this.GetType(), "Message", "alert('Login Successfull');", true);

'Or write using Response.Write("Login Successfull");

'Or redirect using Response.Redirect("Mypanel.aspx");

Else

lblStatus.Text = "Wrong Username/Password"

Response.Redirect("SignUp.aspx"); /\* contain sign up form for New user where they can register new username and password in SignUp.aspx \*/

'Or show in messagebox usingScriptManager.RegisterStartupScript(this, this.GetType(), "Message", "alert('Wrong Username/Password');", true);

'Or write using Response.Write("Wrong Username/Password");

End Sub

**Here don’t forgot to set the connection string in web.config file here I am getting database connection from web.config file for that reason you need to set the connectionstring in web.config file like this**

<connectionStrings>

<add name="dbconnection" connectionString="Data Source=SureshDasari;Integrated Security=true;Initial Catalog=MySampleDB"/>

</connectionStrings>

**Q.2 Examples of File Handling**

**Answer: You can give any examples that can either Write or Read, open or append file from below examples:**

string path = Server.MapPath("file.txt");

**//Create and write to a file**

if (!File.Exists(path))

{

using (StreamWriter sw = File.CreateText(path))

{

sw.Write(@"Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Mauris rutrum.

Nunc id massa. Pellentesque pretium, tortor eget fermentum tincidunt, eros erat

non sem id ligula venenatis auctor. Donec vulputate. Aenean ut pede a ligula ornare

posuere.");

sw.WriteLine();

sw.WriteLine();

sw.WriteLine("THE END");

}

}

**//Read from the file**

try

{

using (StreamReader sr = new StreamReader(path))

{

String line;

while ((line = sr.ReadLine()) != null)

{

Response.Write(line + "<br />");

}

}

}

catch (Exception ex)

{

Response.Write("<p>The file could not be read:");

Response.Write(ex.Message + "</p>");

}

**//Open the file and append more text**

try

{

using (StreamWriter sw = File.AppendText(path))

{

sw.Write("Opened the file and added this line on ");

sw.WriteLine(DateTime.Now);

}

}

catch (Exception ex)

{

Response.Write("<p>The file could not be opened:");

Response.Write(ex.Message + "</p>");

}

**// Open the file to read from.**

try

{

using (StreamReader sr = File.OpenText(path))

{

String line;

while ((line = sr.ReadLine()) != null)

{

Response.Write(line + "<br />");

}

}

}

catch (Exception ex)

{

Response.Write("<p>The file could not be read:");

Response.Write(ex.Message + "</p>");

}

**//Create new folder/directory**

string newpath = Server.MapPath("NewFolder");

try

{

**// Determine whether the directory exists.**

if (Directory.Exists(newpath))

{

Response.Write("That path exists already.<br />");

return;

}

**// Try to create the directory.**

DirectoryInfo di = Directory.CreateDirectory(newpath);

string created = Directory.GetCreationTime(newpath).ToString();

Response.Write("The directory was created at " + created + "<br />");

**// Delete the directory.**

di.Delete();

Response.Write("The directory was deleted successfully.<br />");

}

catch (Exception ex)

{

Response.Write("The process failed: " + ex.ToString() + "<br />");

}

**//Recreate folder and copy file to it**

try

{

**// Determine whether the directory exists.**

if (Directory.Exists(newpath))

{

Response.Write("That path exists already.<br />");

return;

}

**// Try to create the directory.**

DirectoryInfo di = Directory.CreateDirectory(newpath);

string created = Directory.GetCreationTime(newpath).ToString();

Response.Write("The directory was recreated at " + created + "<br />");

}

catch (Exception ex)

{

Response.Write("The process failed: " + ex.ToString() + "<br />");

}

string copy = newpath + "\\filecopy.txt";

try

{

File.Copy(Server.MapPath("file.txt"), copy);

Response.Write("File successfully copied<br />");

Response.Write("Created: " + File.GetCreationTime(copy) + "<br />");

Response.Write("Last Accessed: " + File.GetLastAccessTime(copy) + "<br />");

Response.Write("Last Written: " + File.GetLastWriteTime(copy) + "<br />");

}

catch (Exception ex)

{

Response.Write("The process failed: " + ex.ToString() + "<br />");

}

[Note: please refer FILE HANDLING IN VB. For more]

**Q.3 Difference between form handling and file handling.**

Answer: Read the following Content and summarize the definition with example given below:

**ASP.NET Web Forms are:**

* Based on Microsoft ASP.NET technology, in which code that runs on the server dynamically generates Web page output to the browser or client device.
* Compatible with any browser or mobile device. An ASP.NET Web page automatically renders the correct browser-compliant HTML for features such as styles, layout, and so on.
* Compatible with any language supported by the .NET common language runtime, such as Microsoft Visual Basic and Microsoft Visual C#.
* Built on the Microsoft .NET Framework. This provides all the benefits of the framework, including a managed environment, type safety, and inheritance.
* Flexible because you can add user-created and third party controls to them.

**ASP.NET Web Forms offer:**

* Separation of HTML and other UI code from application logic.
* A rich suite of server controls for common tasks, including data access.
* Powerful data binding, with great tool support.
* Support for client-side scripting that executes in the browser.
* Support for a variety of other capabilities, including routing, security, performance, internationalization, testing, debugging, error handling and state management.

### Advantages of a Web Forms-Based Web Application

The Web Forms-based framework offers the following advantages:

* It supports an event model that preserves state over HTTP, which benefits line-of-business Web application development. The Web Forms-based application provides dozens of events that are supported in hundreds of server controls.
* It uses a Page Controller pattern that adds functionality to individual pages. For more information, see [Page Controller](http://go.microsoft.com/fwlink/?LinkId=106359) on the MSDN Web site.
* It uses view state or server-based forms, which can make managing state information easier.
* It works well for small teams of Web developers and designers who want to take advantage of the large number of components available for rapid application development.
* In general, it is less complex for application development, because the components (the **Page** class, controls, and so on) are tightly integrated and usually require less code than the MVC model.

# ASP.NET Form Handling

**Process of posting data to the server and retrieving values sent.**

Retrieve Form Values:

To retrieve the values from a **runat server** form:

|  |  |
| --- | --- |
| **Control** | **Retrieve value** |
| <asp:textbox id=”txtName” runat=”server” text=””/> | txtName.text |
| <asp:checkbox id=”cbName” runat=”server” text=”HTML”/> | cbName.text |
| <asp:dropdownlist id=”ddlName” runat=”server”>      <asp:listitem value=”1” text=”First”/>      <asp:listitem value=”2” text=”Second”/>      <asp:listitem value=”3” text=”Third”/> </asp:dropdownlist> | ddlName.SelectedItem.Text  or  ddlName.SelectedItem.Value |

To retrieve form values sent from another page:

|  |  |
| --- | --- |
| **Form Type** | **Retrieve value** |
| <form action=”getvalues.aspx” method=”**post**”>   <input type=”text” name=”txtName”> </form> | Request.Form(“txtName”) |
| <form action=”getvalues.aspx” method=”**get**”>   <input type=”text” name=”txtName”> </form> | Request.QueryString(“txtName”) |