REPORT

Step 1: **User Authentication**

```
<asp:Content ID="Content1" ContentPlaceHolderID="MainContent" Runat="Server">
    <h2>Login</h2>
    <asp:TextBox ID="txtUsername" runat="server" placeholder="Username"></asp:TextBox><br
/>
    <asp:TextBox ID="txtPassword" runat="server" TextMode="Password"
placeholder="Password"></asp:TextBox><br />
    <asp:Button ID="btnLogin" runat="server" Text="Login" OnClick="btnLogin_Click" />
    <asp:Label ID="lblMessage" runat="server" ForeColor="Red"></asp:Label>
</asp:Content>
```

CODE behind for handling login (login.aspx.vb):

```
Protected Sub btnLogin_Click(sender As Object, e As EventArgs)
    Dim username As String = txtUsername.Text
    Dim password As String = txtPassword.Text

    ' Simple validation
    If username = "admin" And password = "password123" Then
        ' Set session to remember logged-in user
        Session("Username") = username
        Response.Redirect("Home.aspx")
    Else
        lblMessage.Text = "Invalid username or password."
    End If
End Sub
```

Step 3: **Role-Based Authorization**

Assume you have 2 roles: Admin and HR

```
If username = "admin" And password = "password123" Then
    Session("Username") = username
    Session("Role") = "Admin" ' Assign role
    Response.Redirect("Home.aspx")
ElseIf username = "hruser" And password = "hrpass" Then
    Session("Username") = username
    Session("Role") = "HR" ' Assign role
    Response.Redirect("Home.aspx")
Else
    lblMessage.Text = "Invalid username or password."
End If
```

Restricting page access based on role in Home.aspx

```
If Session("Role") Is Nothing OrElse Session("Role").ToString() <> "Admin" Then
    Response.Redirect("Unauthorized.aspx")
```

End If

Step 3: **SQL Injection Prevention** (Parameterize Queries)

```
Dim conn As New SqlConnection(connectionString)
Dim cmd As New SqlCommand("SELECT * FROM Employees WHERE Username = @Username", conn)

' Use parameters to avoid SQL injection
cmd.Parameters.AddWithValue("@Username", username)

conn.Open()
Dim reader As SqlDataReader = cmd.ExecuteReader()
' Process the reader
conn.Close()
```

Step 4: **Session Management**

```
<configuration>
  <system.web>
    <authentication mode="Forms">
      <forms timeout="20" />
    </authentication>
  </system.web>
</configuration>
```

2. **Logout** button (logout.aspx.vb):

```
vb
CopyEdit
Protected Sub Page_Load(sender As Object, e As EventArgs)
  Session.Abandon()  ' Clear session
  Response.Redirect("login.aspx")
End Sub
```

Step 5: **Password Security**

1. **Hash password before storing**:

```
Imports System.Security.Cryptography
Imports System.Text

Public Function HashPassword(password As String) As String
    Dim sha256 As SHA256 = SHA256.Create()
    Dim bytes As Byte() = sha256.ComputeHash(Encoding.UTF8.GetBytes(password))
    Return BitConverter.ToString(bytes).Replace("-", "").ToLower()
End Function
```

INPUT VALIDATION

```
<asp:RequiredFieldValidator ID="rfvUsername" runat="server" ControlToValidate="txtUsername"
ErrorMessage="Username is required" ForeColor="Red"></asp:RequiredFieldValidator>
<asp:RequiredFieldValidator ID="rfvPassword" runat="server" ControlToValidate="txtPassword"
ErrorMessage="Password is required" ForeColor="Red"></asp:RequiredFieldValidator>
```

RegularExpressionValidator

```
<asp:RegularExpressionValidator ID="revEmail" runat="server" ControlToValidate="txtEmail"
    ValidationExpression="^[^@]+@[^@]+\.[^@]+$" ErrorMessage="Invalid email format"
ForeColor="Red"></asp:RegularExpressionValidator>
```