

Practical 9: CRUD Registration using GridView

StudentRegistry.aspx:

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

<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Student Registration</title>
    <style>
        body { font-family: Arial, sans-serif; margin: 20px; }
        .form-container { background: #f4f4f4; padding: 15px; border-radius: 5px;
margin-bottom: 20px; }
        .form-container input { margin: 5px; }
        .grid-view { width: 100%; border-collapse: collapse; }
        .grid-view th, .grid-view td { border: 1px solid #ddd; padding: 8px; }
        .grid-view th { background-color: #333; color: white; }
        .grid-view tr:nth-child(even) { background-color: #f9f9f9; }
    </style>
</head>
<body>
    <form id="form1" runat="server">

        <div class="form-container">
            <h3>Register New Student</h3>
            <div>
                <asp:Label For="txtNewUsername"
runat="server">Username:</asp:Label>
                <asp:TextBox ID="txtNewUsername" runat="server"></asp:TextBox>
            </div>
            <div>
                <asp:Label For="txtNewPassword"
runat="server">Password:</asp:Label>
                <asp:TextBox ID="txtNewPassword" runat="server"
TextMode="Password"></asp:TextBox>
            </div>
            <asp:Button ID="btnCreate" runat="server" Text="Create Student"
OnClick="btnCreate_Click" />
            <br />
            <asp:Literal ID="ltMessage" runat="server"
EnableViewState="false"></asp:Literal>
        </div>

        <hr />

        <h3>Registered Students</h3>
        <asp:GridView ID="GridView1" runat="server"
            AutoGenerateColumns="False"
            DataKeyNames="Id"
            OnRowEditing="GridView1_RowEditing"
            OnRowCancelingEdit="GridView1_RowCancelingEdit"
            OnRowUpdating="GridView1_RowUpdating"
            OnRowDeleting="GridView1_RowDeleting"
            CssClass="grid-view"
            EmptyDataText="No students found.">
            <Columns>

                <asp:TemplateField HeaderText="Username">
```

```

        <ItemTemplate>
            <asp:Label ID="lblUsername" runat="server" Text='<%#
Eval("Username") %>'></asp:Label>
        </ItemTemplate>
        <EditItemTemplate>
            <asp:TextBox ID="txtEditUsername" runat="server"
Text='<%# Bind("Username") %>'></asp:TextBox>
        </EditItemTemplate>
    </asp:TemplateField>

    <asp:TemplateField HeaderText="Password">
        <ItemTemplate>
            *****
        </ItemTemplate>
        <EditItemTemplate>
            <asp:TextBox ID="txtEditPassword" runat="server"
Text='<%# Bind("Password") %>'></asp:TextBox>
            <%-- We use Bind() here so we can read the original
value, but we also show it, which is bad practice. See security note. --%>
        </EditItemTemplate>
    </asp:TemplateField>

    <asp:CommandField ShowEditButton="True" ShowDeleteButton="True"
ButtonType="Link" />
</Columns>
</asp:GridView>

</form>
</body>
</html>

```

StudentRegistry.aspx.cs:

```

using System;
using System.Configuration;
using System.Data;
using System.Data.SqlClient;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace StudentCRUD
{
    public partial class StudentRegistry : System.Web.UI.Page
    {
        // Get the connection string from Web.config
        private readonly string constr =
ConfigurationManager.ConnectionStrings["DBConnectionString"].ConnectionString;

        protected void Page_Load(object sender, EventArgs e)
        {
            if (!IsPostBack)
            {
                BindGrid();
            }
        }

        //=====
        // (R)EAD Operation
        //=====

        private void BindGrid()

```

```

    {
        using (SqlConnection con = new SqlConnection(constr))
        {
            using (SqlDataAdapter sda = new SqlDataAdapter("SELECT Id,
Username, Password FROM Students", con))
            {
                DataTable dt = new DataTable();
                sda.Fill(dt);
                GridView1.DataSource = dt;
                GridView1.DataBind();
            }
        }
    }

//=====
// (C)REATE Operation
//=====

protected void btnCreate_Click(object sender, EventArgs e)
{
    string username = txtNewUsername.Text;
    string password = txtNewPassword.Text; // *** See security note!

    if (string.IsNullOrEmpty(username) || string.IsNullOrEmpty(password))
    {
        ltMessage.Text = "<span style='color:red;'>Username and Password
are required.</span>";
        return;
    }

    string query = "INSERT INTO Students (Username, Password) VALUES
(@Username, @Password)";
    using (SqlConnection con = new SqlConnection(constr))
    {
        using (SqlCommand cmd = new SqlCommand(query, con))
        {
            cmd.Parameters.AddWithValue("@Username", username);
            cmd.Parameters.AddWithValue("@Password", password); // ***
Storing plain text!
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
        }
    }

    // Clear fields and re-bind the grid
    txtNewUsername.Text = "";
    txtNewPassword.Text = "";
    ltMessage.Text = "<span style='color:green;'>Student registered
successfully!</span>";
    BindGrid();
}

//=====
// (U)PDATE Operations
//=====

// This event fires when the "Edit" link is clicked
protected void GridView1_RowEditing(object sender, GridViewEditEventArgs
e)
{
    GridView1.EditIndex = e.NewEditIndex;
    BindGrid();
}

```

```

}

// This event fires when the "Cancel" link is clicked
protected void GridView1_RowCancelingEdit(object sender,
GridViewCancelEventArgs e)
{
    GridView1.EditIndex = -1; // Exit edit mode
    BindGrid();
}

// This event fires when the "Update" link is clicked
protected void GridView1_RowUpdating(object sender,
GridViewUpdateEventArgs e)
{
    // 1. Get the 'Id' of the row being edited
    int id = Convert.ToInt32(GridView1.DataKeys[e.RowIndex].Value);

    // 2. Find the TextBox controls in the EditItemTemplate
    TextBox txtUsername =
(TextBox)GridView1.Rows[e.RowIndex].FindControl("txtEditUsername");
    TextBox txtPassword =
(TextBox)GridView1.Rows[e.RowIndex].FindControl("txtEditPassword");

    // 3. Get the new values
    string username = txtUsername.Text;
    string password = txtPassword.Text; // *** See security note!

    // 4. Execute the UPDATE query
    string query = "UPDATE Students SET Username = @Username, Password = "
@Password WHERE Id = @Id";
    using (SqlConnection con = new SqlConnection(constr))
    {
        using (SqlCommand cmd = new SqlCommand(query, con))
        {
            cmd.Parameters.AddWithValue("@Username", username);
            cmd.Parameters.AddWithValue("@Password", password);
            cmd.Parameters.AddWithValue("@Id", id);
            con.Open();
            cmd.ExecuteNonQuery();
            con.Close();
        }
    }

    // 5. Exit edit mode and re-bind the grid
    GridView1.EditIndex = -1;
    BindGrid();
}

//=====
// (D)ELETE Operation
//=====

protected void GridView1_RowDeleting(object sender,
GridViewDeleteEventArgs e)
{
    // 1. Get the 'Id' of the row being deleted
    int id = Convert.ToInt32(GridView1.DataKeys[e.RowIndex].Value);

    // 2. Execute the DELETE query
    string query = "DELETE FROM Students WHERE Id = @Id";
    using (SqlConnection con = new SqlConnection(constr))
    {
        using (SqlCommand cmd = new SqlCommand(query, con))
        {

```

```

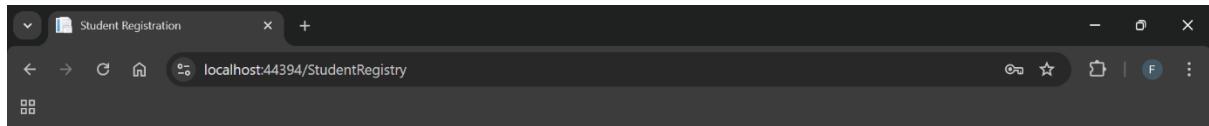
        cmd.Parameters.AddWithValue("@Id", id);
        con.Open();
        cmd.ExecuteNonQuery();
        con.Close();
    }
}

// 3. Re-bind the grid
BindGrid();
}
}

```

Output:

CREATE & READ:



Register New Student

Username:

Password:

Student registered successfully!

Registered Students

Username	Password	
jack	*****	Edit Delete
james	*****	Edit Delete
daniel	*****	Edit Delete



UPDATE:

The screenshot shows a web application for student registration. At the top, there is a header bar with the URL `localhost:44394/StudentRegistry`. Below the header, there is a form titled "Register New Student" with fields for "Username" and "Password", and a "Create Student" button. Below the form is a table titled "Registered Students" with columns "Username" and "Password". The table contains three rows: "jack ward" with password "jack123", "james" with password "*****", and "daniel" with password "*****". Each row has "Update Cancel" and "Edit Delete" links.

Username	Password	
jack ward	jack123	Update Cancel
james	*****	Edit Delete
daniel	*****	Edit Delete

DELETE:

This screenshot is identical to the one above, showing the "Register New Student" form and the "Registered Students" table. The table still lists "jack ward", "james", and "daniel". The "Edit" and "Delete" links are present for each student entry.

Username	Password	
jack ward	jack123	Update Cancel
james	*****	Edit Delete
daniel	*****	Edit Delete

