Name of Student: **Sunny Satish Halkatti**		
Roll Number: **17 (B)**		Lab Assignment Number: **6**
Title of Lab Assignment: Demonstrate usage of data-bound controls, simple stored procedures, and parameterized stored procedures in asp.net		
DOP: 02-05-2023		DOS: 08-05-2023
CO: CO2	PO: PO3, PO5,PO6, PO7, PO11,PO12	Faculty Signature:

Student's Signature

[illegible]

AWTL-Practical-6

Aim: Demonstrate usage of data-bound controls, simple stored procedures, and parameterized stored procedures in asp.net

Theory:

Data-Bound Controls

In ADO.NET, data-bound controls are a set of UI controls designed to work with data from a data source, such as a database. They provide a way to display, manipulate, and update data in a user interface, without writing code to interact with the data source directly.
Some examples of data-bound controls in ADO.NET include
DataGridView: a grid-like control that displays data from a data source in a tabular format. It allows users to sort, filter, and edit the data directly in the grid.
ListBox: a control that displays a list of items from a data source. It allows users to select one or more items from the list.
ComboBox: a control that displays a drop-down list of items from a data source. It allows users to select one item from the list.
TextBox: a control that displays a single value from a data source. It allows users to edit the value directly in the textbox.

Repeater

In ADO.NET, the Repeater control is a data-bound control that is used to display repeated content from a data source in a user interface. It provides a flexible and customizable way to display data from a data source in a repeating format, such as a table, list, or other custom layout.
The Repeater control works by binding to a data source, such as a dataset or data table, and iterating over the data to display the content for each item in the data source. It allows for custom formatting of the data, and can be used to display complex data structures or hierarchical data.
The Repeater control is highly flexible and can be customized in a variety of ways to suit the needs of an application. For example, it allows for custom templates to be defined for the header, footer, and item content, as well as for pagination, sorting, and filtering of the data.

ListView

In ADO.NET, the ListView control is a data-bound control that displays a collection of items in a user interface. It is similar to the ListBox and ComboBox controls, but provides additional features such as multi-column layout, item editing, and custom formatting.

The ListView control works by binding to a data source, such as a dataset or data table, and displaying the data in a set of columns and rows. Each row represents an item in the data source, and each column represents a field or property of the item. The ListView control allows for custom formatting of the data, and can be used to display complex data structures or hierarchical data.
The ListView control provides a variety of features for interacting with the data, such as sorting, filtering, grouping, and editing. It also provides a variety of visual styles and layouts, including an icon view, tile view, and details view.

DataList
In ADO.NET, the DataList control is a data-bound control that is used to display repeated content from a data source in a user interface. It provides a flexible and customizable way to display data from a data source in a repeating format, such as a list, grid or table.
The DataList control works by binding to a data source, such as a dataset or data table, and iterating over the data to display the content for each item in the data source. It allows for custom formatting of the data, and can be used to display complex data structures or hierarchical data.
The DataList control is highly flexible and can be customized in a variety of ways to suit the needs of an application. For example, it allows for custom templates to be defined for the header, footer, and item content, as well as for pagination, sorting, and filtering of the data.

GridView
In ADO.NET, the GridView control is a powerful data-bound control that displays data in a tabular format with columns and rows. It is commonly used for displaying and editing data from a database in a web application.
The GridView control works by binding to a data source, such as a dataset or data table, and displaying the data in a set of columns and rows. Each row represents a record in the data source, and each column represents a field or property of the record. The GridView control allows for custom formatting of the data, and can be used to display complex data structures or hierarchical data.

The GridView control provides a variety of features for interacting with the data, such as sorting, paging, filtering, and editing. It also provides a variety of visual styles and layouts, including alternating row colors, column headers, and footer rows.

FormView

In ADO.NET, the FormView control is a data-bound control that provides a flexible and customizable way to display and edit data from a single record in a user interface. It is commonly used for displaying and editing data from a database in a web application.
The FormView control works by binding to a data source, such as a dataset or data table, and displaying the data for a single record in a custom layout. The layout of the FormView control is highly customizable, allowing for complete control over the appearance and behavior of the user interface.
The FormView control provides a variety of features for interacting with the data, such as editing, inserting, deleting, and paging. It also provides a variety of events that can be used to customize the behavior of the control, such as the ItemCommand event, which is raised when a button is clicked in the FormView control.

Stored Procedure

In database management, a stored procedure is a set of SQL statements that are stored in the database and can be executed repeatedly by calling the procedure name. It is a type of pre-compiled code that is stored in the database and can be called by other programs or scripts.
Stored procedures are typically used to perform a specific task or set of tasks, such as updating a database record, retrieving data from the database, or performing calculations on data. They can be used to simplify complex database operations and improve performance by reducing the amount of data that needs to be transferred between the database and the application.
Stored procedures can also provide security benefits by controlling access to sensitive data and enforcing business rules. They can be executed by authorized users or applications while preventing unauthorized access to the underlying data.

Parameterized stored procedure

A parameterized stored procedure is a type of stored procedure in which input parameters are defined and used in the SQL statements within the procedure. This allows the stored procedure to be executed with different values for the parameters, making it more flexible and reusable.
In a parameterized stored procedure, input parameters are declared and given a data type. The SQL statements within the procedure can then reference these parameters using the

"@" symbol followed by the parameter name. When the stored procedure is executed, values are passed in for the parameters, which are used in place of the parameter references in the SQL statements.

A) Create a web page that demonstrates the use of data-bound controls of ASP.NET.

Code:

DataBound.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="DataBound.aspx.cs"
Inherits="Practical6a.DataBound" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>STUDENT DATA</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h2>STUDENT INFO</h2>
      <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
        CellPadding="6" OnRowCancelingEdit="GridView1_RowCancelingEdit"
        OnRowEditing="GridView1_RowEditing"
OnRowUpdating="GridView1_RowUpdating">
        <Columns>
          <asp:TemplateField>
            <ItemTemplate>
              <asp:Button ID="btn_Edit" runat="server" Text="Edit"
CommandName="Edit" />
            </ItemTemplate>
            <EditItemTemplate>
              <asp:Button ID="btn_Update" runat="server" Text="Update"
                CommandName="Update" />
              <asp:Button ID="btn_Cancel" runat="server" Text="Cancel"
                CommandName="Cancel" />
            </EditItemTemplate>
          </asp:TemplateField>
          <asp:TemplateField HeaderText="ID">
            <ItemTemplate>
```

```
                        <asp:Label ID="lbl_ID" runat="server" Text='<%#Eval("ID")
%>'></asp:Label>
                    </ItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="NAME">
                    <ItemTemplate>
                        <asp:Label ID="lbl_Name" runat="server" Text='<%#Eval("NAME")
%>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="txt_Name" runat="server" Text='<%#Eval("NAME")
%>'></asp:TextBox>
                    </EditItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="COURSE">
                    <ItemTemplate>
                        <asp:Label ID="lbl_Course" runat="server" Text='<%#Eval("DEGREE")
%>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="txt_Course" runat="server" Text='<%#Eval("DEGREE")
%>'></asp:TextBox>
                    </EditItemTemplate>
                </asp:TemplateField>
                <asp:TemplateField HeaderText="SUBJECT">
                    <ItemTemplate>
                        <asp:Label ID="lbl_Subject" runat="server" Text='<%#Eval("SUBJECT")
%>'></asp:Label>
                    </ItemTemplate>
                    <EditItemTemplate>
                        <asp:TextBox ID="txt_Subject" runat="server" Text='<%#Eval("SUBJECT")
%>'></asp:TextBox>
                    </EditItemTemplate>
                </asp:TemplateField>
            </Columns>
            <HeaderStyle BackColor="#FBB917" ForeColor="#000" />
            <RowStyle BackColor="#e7ceb6" />
        </asp:GridView>
    </div>
</form>
```

```
</body>
</html>
```

DataBound.aspx.cs

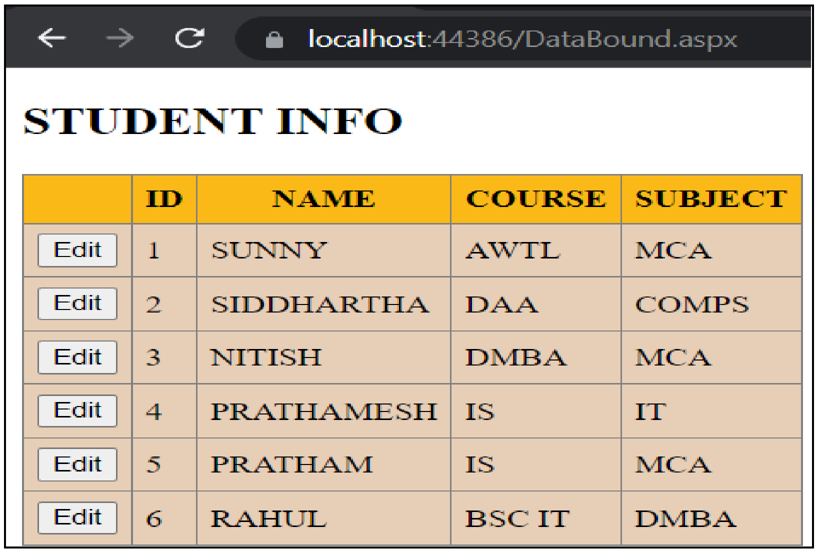
```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
using System.Xml.Linq;
namespace Practical6a
{
   public partial class DataBound : System.Web.UI.Page
   {
      SqlConnection con = null;
      string cs = "Data
Source=LAPTOP-OPRPD79S\\SQLEXPRESS01;Database=MCA;Integrated security=True";
      SqlDataAdapter adapt;
      DataTable dt;
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            ShowData();
         }
      }
      //ShowData method for Displaying Data in Gridview
      protected void ShowData()
      {
         dt = new DataTable();
         con = new SqlConnection(cs);
         con.Open();
         adapt = new SqlDataAdapter("Select ID,NAME,DEGREE,SUBJECT from student",
con);
         adapt.Fill(dt);
         if (dt.Rows.Count > 0)
```

```
        {
          GridView1.DataSource = dt;
          GridView1.DataBind();
        }
        con.Close();
      }
      protected void GridView1_RowEditing(object sender,
     System.Web.UI.WebControls.GridViewEditEventArgs e)
      {
        //NewEditIndex property used to determine the index of the row being edited.
        GridView1.EditIndex = e.NewEditIndex;
        ShowData();
      }
      protected void GridView1_RowUpdating(object
sender,System.Web.UI.WebControls.GridViewUpdateEventArgs e)
      {
        //Finding the controls from Gridview for the row which is going to update
        Label id = GridView1.Rows[e.RowIndex].FindControl("lbl_ID") as Label;
        TextBox name = GridView1.Rows[e.RowIndex].FindControl("txt_NAME") as TextBox;
        TextBox course = GridView1.Rows[e.RowIndex].FindControl("txt_COURSE") as
TextBox;
        TextBox subject = GridView1.Rows[e.RowIndex].FindControl("txt_SUBJECT") as
TextBox;
        con = new SqlConnection(cs);
        con.Open();
        //updating the record
        SqlCommand cmd = new SqlCommand("Update student set NAME = '"+name.Text+"',
DEGREE = '"+course.Text+"', SUBJECT = '"+subject.Text+"' where ID =
"+Convert.ToInt32(id.Text),con);
        cmd.ExecuteNonQuery();
        con.Close();
        //Setting the EditIndex property to -1 to cancel the Edit mode in Gridview
        GridView1.EditIndex = -1;
        //Call ShowData method for displaying updated data
        ShowData();
      }
      protected void GridView1_RowCancelingEdit(object
sender,System.Web.UI.WebControls.GridViewCancelEditEventArgs e)
      {
        //Setting the EditIndex property to -1 to cancel the Edit mode in Gridview
```

```
            GridView1.EditIndex = -1;
            ShowData();
        }
    }}
```

Output:

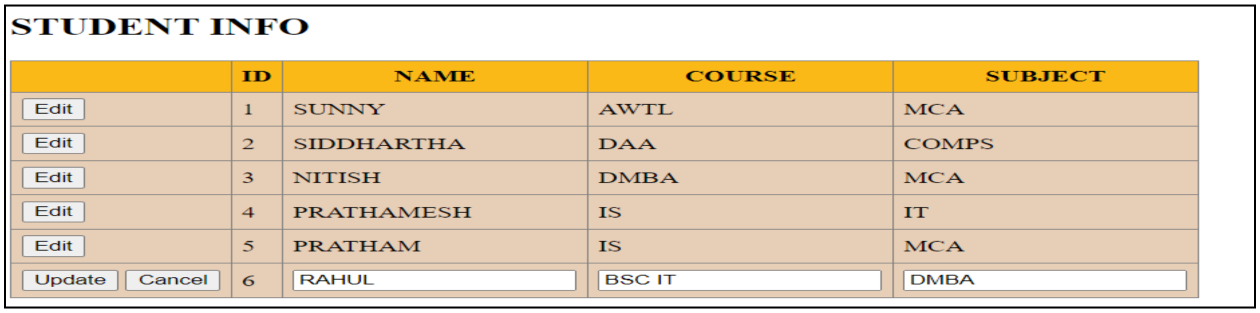
Before Editing:



localhost:44386/DataBound.aspx

STUDENT INFO

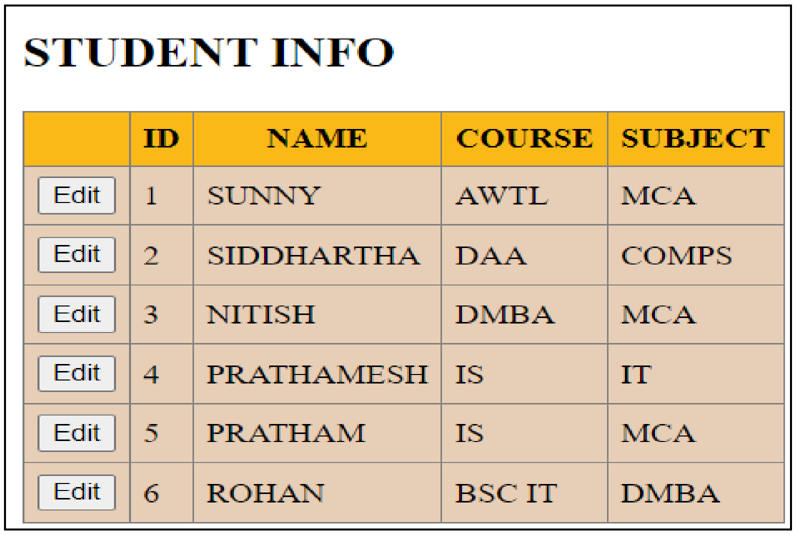
	ID	NAME	COURSE	SUBJECT
Edit	1	SUNNY	AWTL	MCA
Edit	2	SIDDHARTHA	DAA	COMPS
Edit	3	NITISH	DMBA	MCA
Edit	4	PRATHAMESH	IS	IT
Edit	5	PRATHAM	IS	MCA
Edit	6	RAHUL	BSC IT	DMBA



STUDENT INFO

	ID	NAME	COURSE	SUBJECT
Edit	1	SUNNY	AWTL	MCA
Edit	2	SIDDHARTHA	DAA	COMPS
Edit	3	NITISH	DMBA	MCA
Edit	4	PRATHAMESH	IS	IT
Edit	5	PRATHAM	IS	MCA
Update Cancel	6	RAHUL	BSC IT	DMBA

After Editing:



STUDENT INFO

	ID	NAME	COURSE	SUBJECT
Edit	1	SUNNY	AWTL	MCA
Edit	2	SIDDHARTHA	DAA	COMPS
Edit	3	NITISH	DMBA	MCA
Edit	4	PRATHAMESH	IS	IT
Edit	5	PRATHAM	IS	MCA
Edit	6	ROHAN	BSC IT	DMBA

B) Design a web page to demonstrate the working of a simple stored procedure.
Code:
Stored Procedure

```
CREATE OR ALTER PROCEDURE SunnyProcedure
AS
select * from STUDENT
GO;
```

StoredProcedure.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="StoredProcedure.aspx.cs" Inherits="Practical6.StoredProcedure" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <asp:GridView ID="GridView1" runat="server"></asp:GridView>
        </div>
    </form>
```

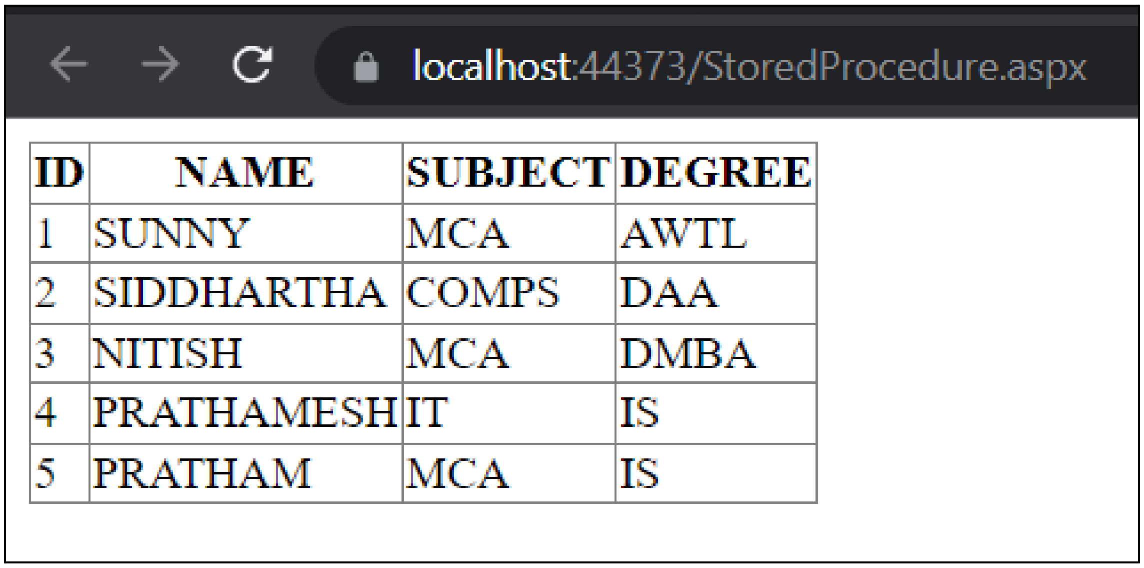
```
</body>
</html>
```

StoredProcedure.aspx.cs

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

namespace Practical6
{
   public partial class StoredProcedure: System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         try
         {
            SqlConnection con = null;
            string connectionstring = "Data
Source=LAPTOP-OPRPD79S\\SQLEXPRESS01;Database=MCA;Integrated
security=True";
            con = new SqlConnection(connectionstring);
            con.Open();
            System.Data.SqlClient.SqlCommand objCmd = new
System.Data.SqlClient.SqlCommand("SunnyProcedure", con);
            objCmd.CommandType = System.Data.CommandType.StoredProcedure;
            GridView1.DataSource = objCmd.ExecuteReader();
            GridView1.DataBind();
            con.Close();
 }
         catch (Exception ex)
         {
            Console.WriteLine(ex.ToString());
         }
      }}}
```

Output:



localhost:44373/StoredProcedure.aspx

ID	NAME	SUBJECT	DEGREE
1	SUNNY	MCA	AWTL
2	SIDDHARTHA	COMPS	DAA
3	NITISH	MCA	DMBA
4	PRATHAMESH	IT	IS
5	PRATHAM	MCA	IS

C} Design a web page to demonstrate the working of parameterized stored procedure.

Code:

parameterized stored procedure.

```
CREATE OR ALTER PROCEDURE SunnyParamProcedure(
@ID INT, @NAME NVARCHAR(50), @DEGREE NVARCHAR(50), @SUBJECT
NVARCHAR(50) )
AS
BEGIN
INSERT INTO student( [ID], [NAME], [DEGREE], [SUBJECT] )
VALUES ( @ID, @NAME, @DEGREE, @SUBJECT )
END
```

ParameterSP.aspx

```
<%@ Page Language="C#" AutoEventWireup="true"
CodeBehind="ParameterSP.aspx.cs" Inherits="Practical6c.ParameterSP" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
   <title>Practical 6c</title>
</head>
<body>
```

```
   <form id="form1" runat="server">
       <asp:Label ID="Label1" runat="server" Text="ID"></asp:Label>
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
       <asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
       <div>
           <asp:Label ID="Label2" runat="server" Text="Name"></asp:Label>
&nbsp;&nbsp;&nbsp;
           <asp:TextBox ID="TextBox2" runat="server"></asp:TextBox>
           <br />
           <asp:Label ID="Label3" runat="server" Text="Degree"></asp:Label>
&nbsp;
           <asp:TextBox ID="TextBox3" runat="server"></asp:TextBox>
           <br />
           <asp:Label ID="Label4" runat="server" Text="Subject"></asp:Label>
&nbsp;
           <asp:TextBox ID="TextBox4" runat="server"></asp:TextBox>
           <br />
&nbsp;&nbsp;&nbsp;&nbsp;&nbsp;
           <asp:Button ID="Button1" runat="server" OnClick="Button1_Click"
Text="submit" />
&nbsp;&nbsp;&nbsp;&nbsp;
           <asp:Label ID="Label5" runat="server" Text="Result"></asp:Label>
           <br />
           <br />
           <asp:GridView ID="GridView1" runat="server" AutoGenerateColumns="False"
DataSourceID="SqlDataSource1">
               <Columns>
                   <asp:BoundField DataField="ID" HeaderText="ID" SortExpression="ID" />
                   <asp:BoundField DataField="NAME" HeaderText="NAME"
SortExpression="NAME" />
                   <asp:BoundField DataField="SUBJECT" HeaderText="SUBJECT"
SortExpression="SUBJECT" />
                   <asp:BoundField DataField="DEGREE" HeaderText="DEGREE"
SortExpression="DEGREE" />
               </Columns>
           </asp:GridView>
```

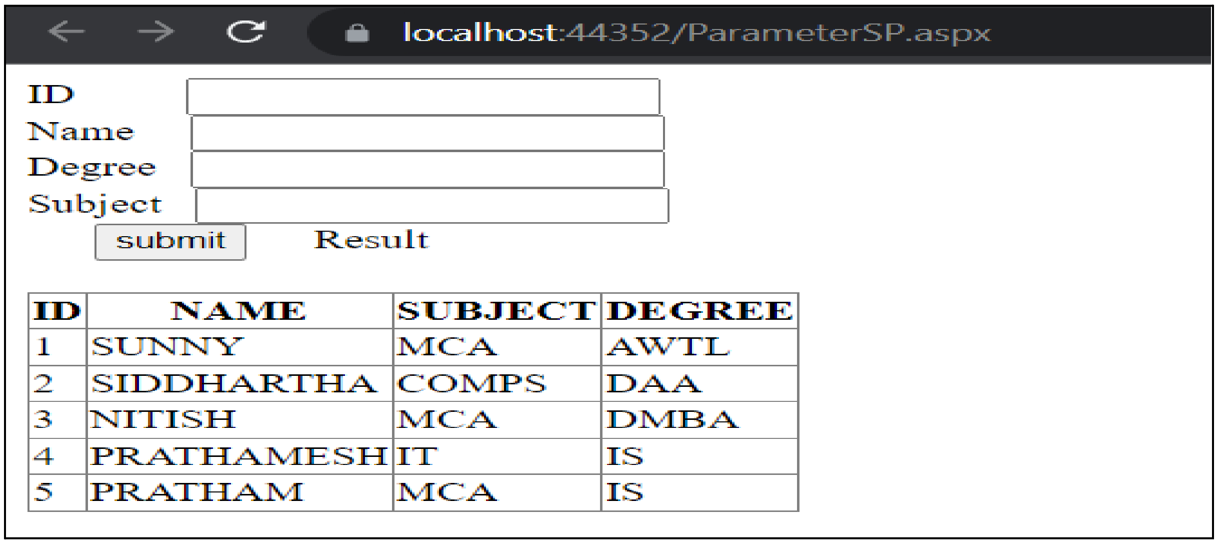
```
            <asp:SqlDataSource ID="SqlDataSource1" runat="server"
ConnectionString="<%$ ConnectionStrings:MCAConnectionString %>"
ProviderName="<%$ ConnectionStrings:MCAConnectionString.ProviderName %>"
SelectCommand="SELECT * FROM [STUDENT]"></asp:SqlDataSource>
        </div>
    </form>
</body>
</html>
```

ParameterSP.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.SqlClient;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace Practical6c
{
    public partial class ParameterSP : System.Web.UI.Page
    {
        SqlConnection con = null;
        string cs = "Data
Source=LAPTOP-OPRPD79S\\SQLEXPRESS01;Database=MCA;Integrated
security=True";
        SqlCommand cmd = new SqlCommand();
        SqlParameter sp1, sp2, sp3, sp4;
        protected void Page_Load(object sender, EventArgs e)
        {
            GridView1.DataBind();
        }
        protected void Button1_Click(object sender, EventArgs e)
        {
            con = new SqlConnection(cs);
            cmd = new SqlCommand("SunnyParamProcedure", con);
            cmd.CommandType = CommandType.StoredProcedure;
            sp1 = new SqlParameter("@ID", SqlDbType.Int);
            sp2 = new SqlParameter("@NAME", SqlDbType.NVarChar);
            sp3 = new SqlParameter("@DEGREE", SqlDbType.NVarChar);
            sp4 = new SqlParameter("@SUBJECT", SqlDbType.NVarChar);
            sp1.Value = Convert.ToInt32(TextBox1.Text);
            sp2.Value = TextBox2.Text;
            sp3.Value = TextBox3.Text;
            sp4.Value = TextBox4.Text;
            cmd.Parameters.Add(sp1);
            cmd.Parameters.Add(sp2);
            cmd.Parameters.Add(sp3);
            cmd.Parameters.Add(sp4);
            con.Open();
            try
            {
               cmd.ExecuteNonQuery();
               Label5.Text = "Successfully inserted!";
               GridView1.DataBind();
            }
            catch (Exception)
            {
               Label5.Text = "Error was occured!"; }}}}
```

Output:

Before Inserting



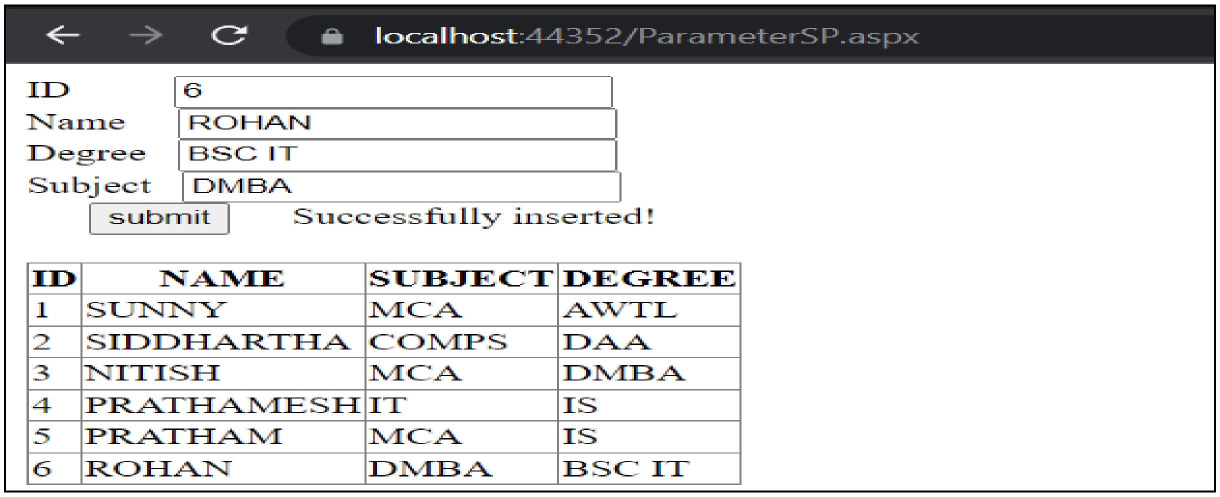
localhost:44352/ParameterSP.aspx

ID	
Name	
Degree	
Subject	

submit Result

ID	NAME	SUBJECT	DEGREE
1	SUNNY	MCA	AWTL
2	SIDDHARTHA	COMPS	DAA
3	NITISH	MCA	DMBA
4	PRATHAMESH	IT	IS
5	PRATHAM	MCA	IS

After Inserting



ID	6
Name	ROHAN
Degree	BSC IT
Subject	DMBA

Successfully inserted!

ID	NAME	SUBJECT	DEGREE
1	SUNNY	MCA	AWTL
2	SIDDHARTHA	COMPS	DAA
3	NITISH	MCA	DMBA
4	PRATHAMESH	IT	IS
5	PRATHAM	MCA	IS
6	ROHAN	DMBA	BSC IT

Conclusions:

We have successfully implemented the usage of data-bound controls, simple stored procedures, and parameterized stored procedures in asp.net in this practical.