Advanced Web Technologies (AWT) Lab (MCAL25)

INDEX

Name of the faculty: Ganesh Bhagwat

Experiment Number	Name of the experiment	Date	CO	Sign
1	Design a Web Application for an Organization with Registration forms and advanced controls.		CO1	
2	Create a website using the master page concept.		CO1	
3	Design a Web Application using advanced controls.		CO1	
4	Webpage Demonstrating Connection-Oriented Architecture (ASP.NET Web Forms with SQL Server Database)		CO2	
5	Webpage Demonstrating Disconnected Architecture (ASP.NET Web Forms with SQL Server Database)		CO2	
6	Create a webpage that demonstrates the use of data bound controls of ASP.NET.		CO2	
7	Design a webpage to demonstrate the working of a simple stored procedure.		CO2	
8	Design a webpage to demonstrate the working of parameterized stored procedure.		CO2	
9	Design a webpage to display the use of LINQ.		CO2	
10	Build websites to demonstrate the working of entity frameworks in dot net.		CO3	
11	Design Web Applications using Client-Side Session Management		C03	
12	Design Web Applications using Server-Side Session Management Techniques		CO3	
13	Build a web page using AJAX Controls.		CO3	
14	Build a web application to create and use web service in ASP.net		CO3	
15	Build a web application to create and WCF service in ASP.net		CO3	
16	Design web application using MVC framework		CO4	

Practical no.: 1
Design a Web Application for an Organization with Registration forms and advanced controls.

Code:

WebForm1.aspx

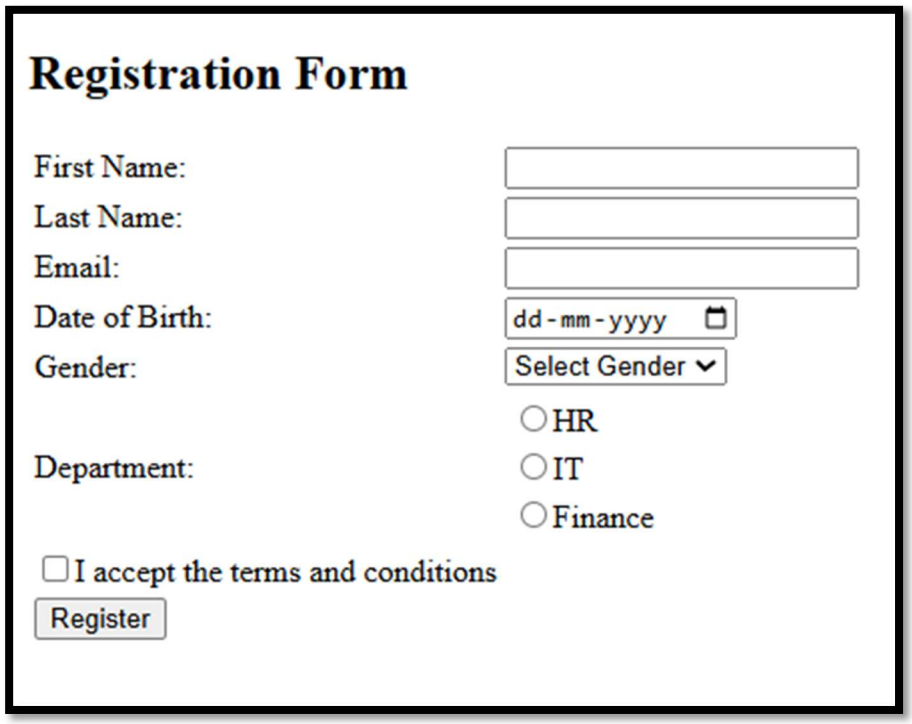
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="WebApplication1.WebForm1" %>

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h2>Registration Form</h2>
      <table>
        <tr>
          <td>
            <label for="txtFirstName">First Name:</label></td>
          <td>
            <asp:TextBox ID="txtFirstName" runat="server" CssClass="form-
control"></asp:TextBox></td>
        </tr>
        <tr>
          <td>
            <label for="txtLastName">Last Name:</label></td>
          <td>
            <asp:TextBox ID="txtLastName" runat="server" CssClass="form-
control"></asp:TextBox></td>
        </tr>
        <tr>
          <td>
            <label for="txtEmail">Email:</label></td>
          <td>
            <asp:TextBox ID="txtEmail" runat="server" CssClass="form-control"></asp:TextBox></td>
        </tr>
        <tr>
          <td>
            <label for="txtDOB">Date of Birth:</label></td>
          <td>
```

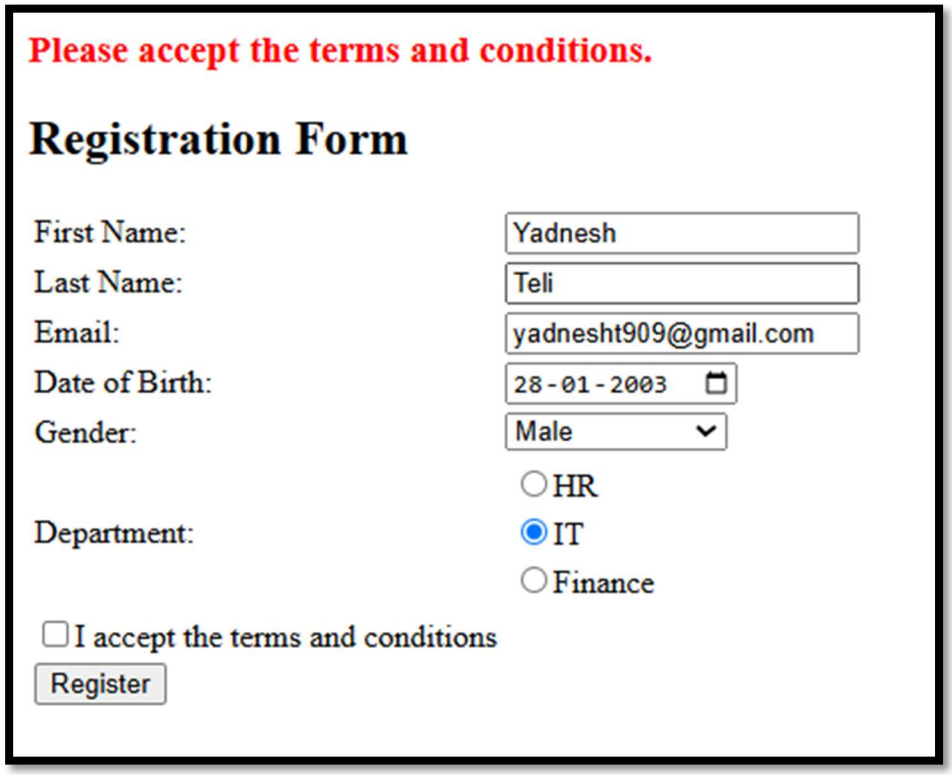
```
            <asp:TextBox ID="txtDOB" runat="server" TextMode="Date" CssClass="form-
control"></asp:TextBox></td>
        </tr>
        <tr>
          <td>
            <label for="ddlGender">Gender:</label></td>
          <td>
            <asp:DropDownList ID="ddlGender" runat="server" CssClass="form-control">
              <asp:ListItem Text="Select Gender" Value=""></asp:ListItem>
              <asp:ListItem Text="Male" Value="Male"></asp:ListItem>
              <asp:ListItem Text="Female" Value="Female"></asp:ListItem>
              <asp:ListItem Text="Other" Value="Other"></asp:ListItem>
            </asp:DropDownList>
          </td>
        </tr>
        <tr>
          <td>
            <label>Department:</label></td>
          <td>
            <asp:RadioButtonList ID="rblDepartment" runat="server">
              <asp:ListItem Text="HR" Value="HR"></asp:ListItem>
              <asp:ListItem Text="IT" Value="IT"></asp:ListItem>
              <asp:ListItem Text="Finance" Value="Finance"></asp:ListItem>
            </asp:RadioButtonList>
          </td>
        </tr>
        <tr>
          <td>
            <asp:CheckBox ID="chkTerms" runat="server" Text="I accept the terms and conditions" />
          </td>
        </tr>
        <tr>
          <td colspan="2">
            <asp:Button ID="btnSubmit" runat="server" Text="Register" OnClick="btnSubmit_Click" />
          </td>
        </tr>
      </table>
    </div>

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

WebForm1.aspx.cs

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

namespace WebApplication1
{
   public partial class WebForm1 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void btnSubmit_Click(object sender, EventArgs e)
      {
         if (chkTerms.Checked)
         {
            string firstName = txtFirstName.Text;
            string lastName = txtLastName.Text;
            string email = txtEmail.Text;
            string dob = txtDOB.Text;
            string gender = ddlGender.SelectedValue;
            string department = rblDepartment.SelectedValue;

            // Display confirmation message
            Response.Write($"<h3>Registration Successful!</h3>");
            Response.Write($"<p>Name: {firstName} {lastName}</p>");
            Response.Write($"<p>Email: {email}</p>");
            Response.Write($"<p>Date of Birth: {dob}</p>");
            Response.Write($"<p>Gender: {gender}</p>");
            Response.Write($"<p>Department: {department}</p>");
         }
         else
         {
            Response.Write("<h3 style='color:red'>Please accept the terms and conditions.</h3>");
         }

      }
   }
}
```

Output:


Registration Form
First Name:
Last Name:
Email:
Date of Birth:
dd-mm-yyyy
Gender:
Select Gender
Department:
HR
IT
Finance
I accept the terms and conditions
Register


Please accept the terms and conditions.
Registration Form
First Name:
Yadnesh
Last Name:
Teli
Email:
yadnesht909@gmail.com
Date of Birth:
28-01-2003
Gender:
Male
Department:
HR
IT
Finance
I accept the terms and conditions
Register

Registration Successful!

Name: Yadnesh Teli

Email: yadnesht909@gmail.com

Date of Birth: 2003-01-28

Gender: Male

Department: IT

Registration Form

First Name: Yadnesh

Last Name: Teli

Email: yadnesht909@gmail.com

Date of Birth: 28-01-2003

Gender: Male

Department: HR / IT / Finance

I accept the terms and conditions

Register

Practical no.: 2
Create a website using the master page concept.

Overview of the Master Page Concept

The **Master Page** allows you to define a consistent layout and design for multiple pages in a web application. Content pages can use the master page to inherit this layout while providing their unique content.

---

1. Steps to Create the Website

Step 1: Create an ASP.NET Web Forms Project

1. Open Visual Studio.
2. Create a new project: **ASP.NET Web Forms Application**.
3. Name the project WebsiteWithMasterPage.

Step 2: Add a Master Page

1. Right-click on the project in **Solution Explorer**.
2. Select **Add** > **New Item** > **Web Forms Master Page**.
3. Name it SiteMaster.master.

Code:

SiteMaster.master

```
<%@ Master Language="C#" AutoEventWireup="true" CodeFile="SiteMaster.master.cs"
Inherits="SiteMaster" %>

<!DOCTYPE html>
<html>
<head>
   <title>My Website</title>
```

```
  <link rel="stylesheet" href="styles.css" />
</head>
<body>
  <div class="header">
    <h1>Welcome to My Website</h1>
    <nav>
      <a href="Home.aspx">Home</a>
      <a href="About.aspx">About</a>
      <a href="Contact.aspx">Contact</a>
    </nav>
  </div>

  <div class="content">
    <asp:ContentPlaceHolder ID="MainContent" runat="server"></asp:ContentPlaceHolder>
  </div>

  <div class="footer">
    <p>&copy; 2025 My Website. All Rights Reserved.</p>
  </div>
</body>
</html>
```

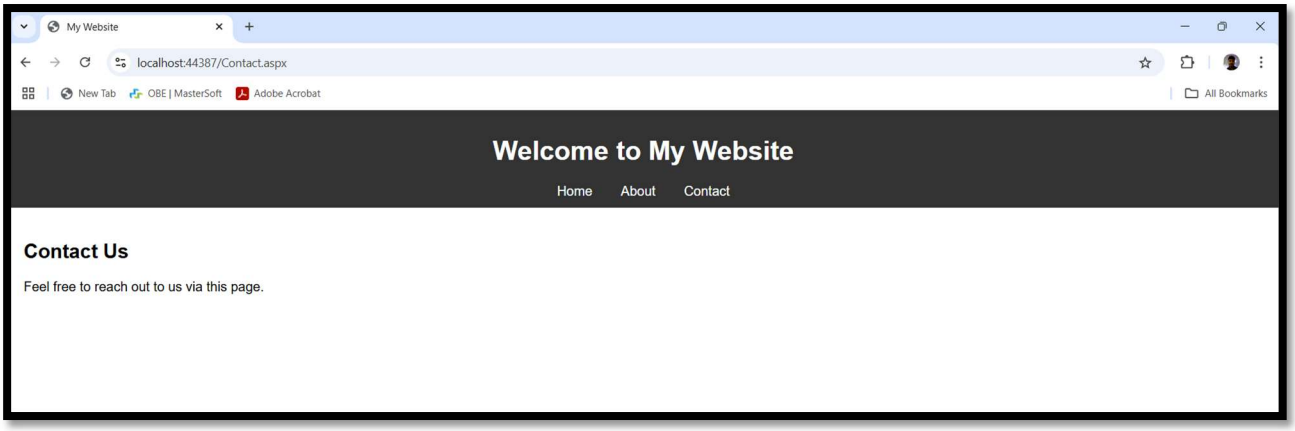
Step 3: Add Content Pages

1. Right-click on the project in **Solution Explorer**.
2. Select **Add** > **New Item** > **Web Form with Master Page**.
3. Name the page Home.aspx and link it to SiteMaster.master.
4. Repeat to create About.aspx and Contact.aspx.

Home.aspx

```
<%@ Page Title="Home" Language="C#" MasterPageFile="~/SiteMaster.master"
AutoEventWireup="true" CodeFile="Home.aspx.cs" Inherits="Home" %>

<asp:Content ID="Content1" ContentPlaceHolderID="MainContent" runat="server">
   <h2>Welcome to the Home Page</h2>
   <p>This is the main content of the Home page.</p>
</asp:Content>
```

Output:



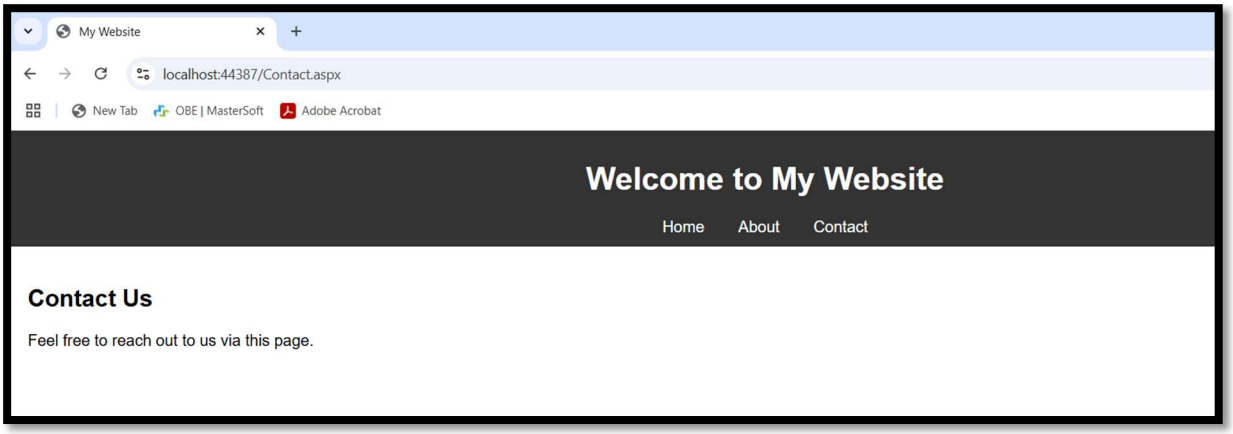
About.aspx

```
<%@ Page Title="About" Language="C#" MasterPageFile="~/SiteMaster.master"
AutoEventWireup="true" CodeFile="About.aspx.cs" Inherits="About" %>

<asp:Content ID="Content1" ContentPlaceHolderID="MainContent" runat="server">
   <h2>About Us</h2>
   <p>Learn more about our organization on this page.</p>
</asp:Content>
```

Contact.aspx

```
<%@ Page Title="Contact" Language="C#" MasterPageFile="~/SiteMaster.master"
AutoEventWireup="true" CodeFile="Contact.aspx.cs" Inherits="Contact" %>

<asp:Content ID="Content1" ContentPlaceHolderID="MainContent" runat="server">
   <h2>Contact Us</h2>
   <p>Feel free to reach out to us via this page.</p>
</asp:Content>
```

Output:



Step 4: Add CSS for Styling

1. Right-click the project in **Solution Explorer**.
2. Add a new folder named Content and a new file styles.css inside it.

styles.css

```
body {
    font-family: Arial, sans-serif;
    margin: 0;
    padding: 0;
}

.header {
    background-color: #333;
    color: white;
    padding: 10px;
    text-align: center;
}

.header nav a {
    color: white;
    text-decoration: none;
    margin: 0 15px;
}

.header nav a:hover {
    text-decoration: underline;
}

.content {
    padding: 20px;
}

.footer {
    background-color: #333;
    color: white;
    text-align: center;
    padding: 10px;
    position: fixed;
```

```
    width: 100%;
    bottom: 0;
}
```

Step 5: Run the Application

- Press **F5** to run the application.
- Navigate between the Home.aspx, About.aspx, and Contact.aspx pages. Each page will have a consistent layout inherited from SiteMaster.master.

Key Advantages of Using Master Pages

1. **Consistency**: Provides a unified layout for all pages.
2. **Maintainability**: Changes to the master page automatically reflect on all linked pages.

Code Reusability: Common elements like navigation bars and footers are defined only once.

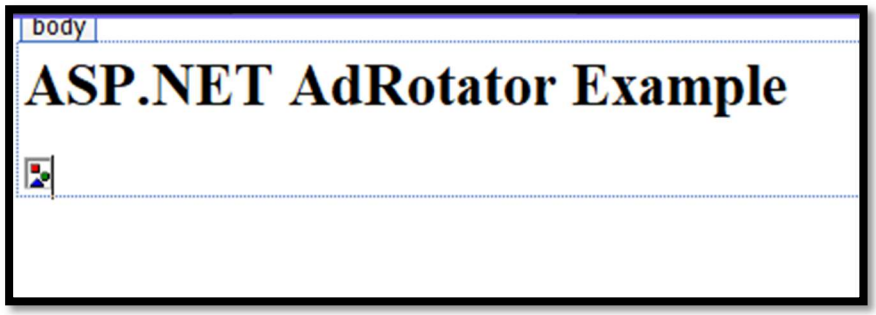
Practical no.: 3
Design a Web Application using advanced controls.

1. AdRotator Control in ASP.NET

The **AdRotator** control displays rotating advertisements based on an XML file.

Steps to Run:

1. Open Visual Studio and create an **ASP.NET Web Application**.
2. Add a new **Web Form** (AdRotator.aspx).
3. Create an XML file (Ads.xml) in the project.
4. Place some ad images in the Images folder.



Code:

AdRotator.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="Practical_Number_3.WebForm1" %>

<!DOCTYPE html>
<html>
<head>
  <title>AdRotator Example</title>
</head>
<body>
  <h2>ASP.NET AdRotator Example</h2>
  <asp:AdRotator ID="AdRotator1" runat="server" AdvertisementFile="~/Ads.xml" />
</body>
</html>
```

Ads.xml (Advertisement File)

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
	<Ad>
		<ImageUrl>Images/geeks.jpg</ImageUrl>
		<NavigateUrl>https://www.geeksforgeeks.org</NavigateUrl>
		<AlternateText> geeksforgeeks </AlternateText>
		<Impressions>50</Impressions>
	</Ad>
	<Ad>
		<ImageUrl>Images/google.jpg</ImageUrl>
		<NavigateUrl>https://www.google.com</NavigateUrl>
		<AlternateText>Google</AlternateText>
		<Impressions>30</Impressions>
	</Ad>
</Advertisements>
```

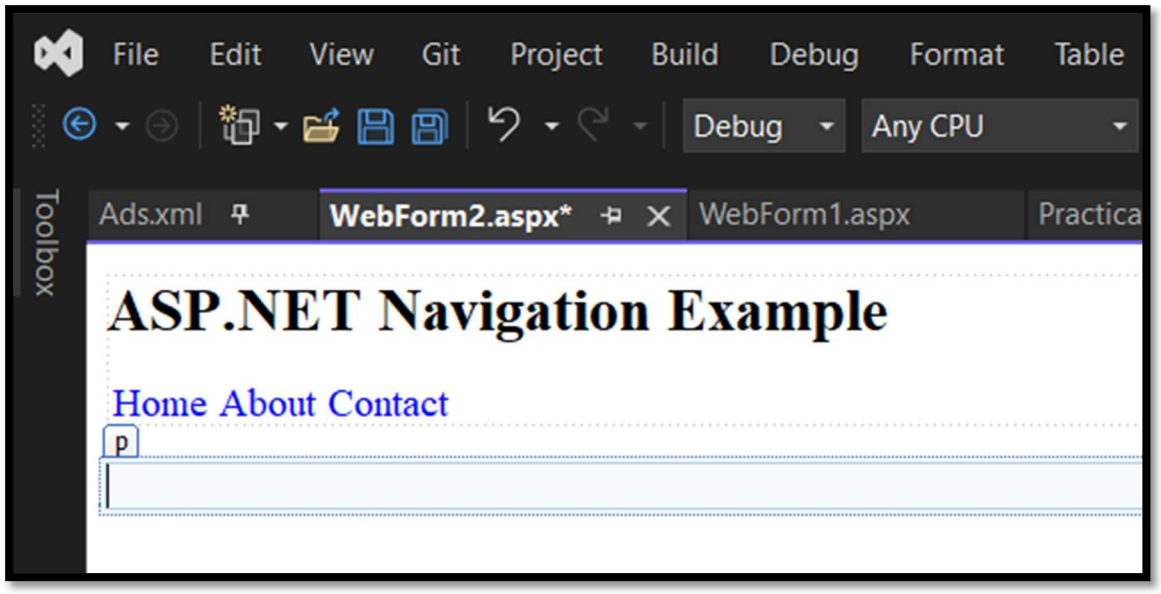
Output:

2. Navigation Control (Menu Navigation)

This example demonstrates how to use a **Menu control** for site navigation.

Steps to Run:

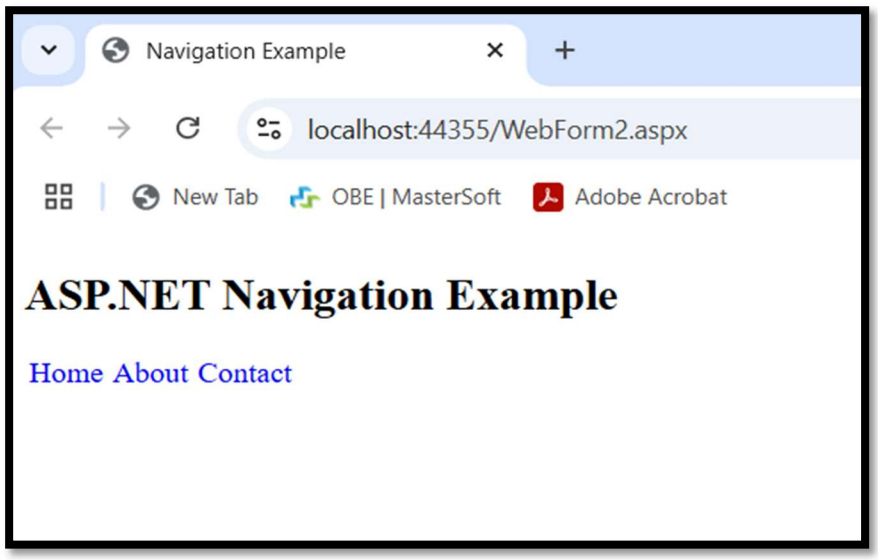
1. Create a new Web Form (Navigation.aspx).
2. Use the Menu control inside an asp:SiteMapDataSource.



Navigation.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="Practical_Number_3.WebForm2" %>

<!DOCTYPE html>
<html>
<head runat="server">
<title>Navigation Example</title>
</head>
<body>
  <form id="form1" runat="server">
  <h2>ASP.NET Navigation Example</h2>
  <asp:Menu ID="Menu1" runat="server" Orientation="Horizontal">
    <Items>
      <asp:MenuItem Text="Home" NavigateUrl="Home.aspx"/>
      <asp:MenuItem Text="About" NavigateUrl="About.aspx"/>
```

```
        <asp:MenuItem Text="Contact" NavigateUrl="Contact.aspx"/>
      </Items>
   </asp:Menu>
   </form>
</body>
</html>
```

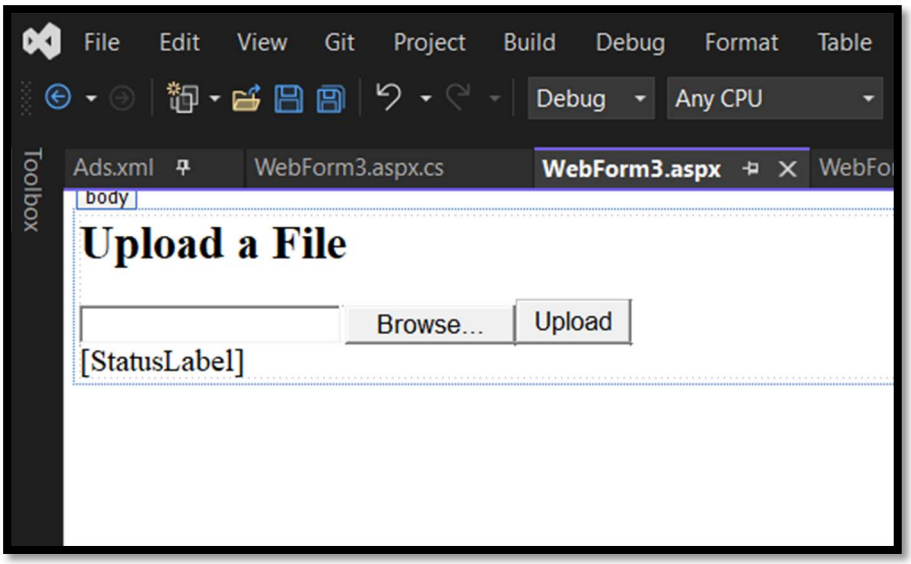


3. File Upload in ASP.NET

This example allows users to upload a file to the server.

Steps to Run:

1. Create a Web Form (FileUpload.aspx).
2. Implement the file upload functionality in C#.



FileUpload.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm3.aspx.cs"
Inherits="Practical_Number_3.WebForm3" %>

<!DOCTYPE html>
<html>
<head runat="server">
  <title>File Upload Example</title>
</head>
<body>
  <form id="form1" runat="server">
  <h2>Upload a File</h2>
  <asp:FileUpload ID="FileUploadControl" runat="server" />
  <asp:Button ID="UploadButton" runat="server" Text="Upload" OnClick="UploadButton_Click"
/>
  <br />
  <asp:Label ID="StatusLabel" runat="server" Text=""></asp:Label>
```

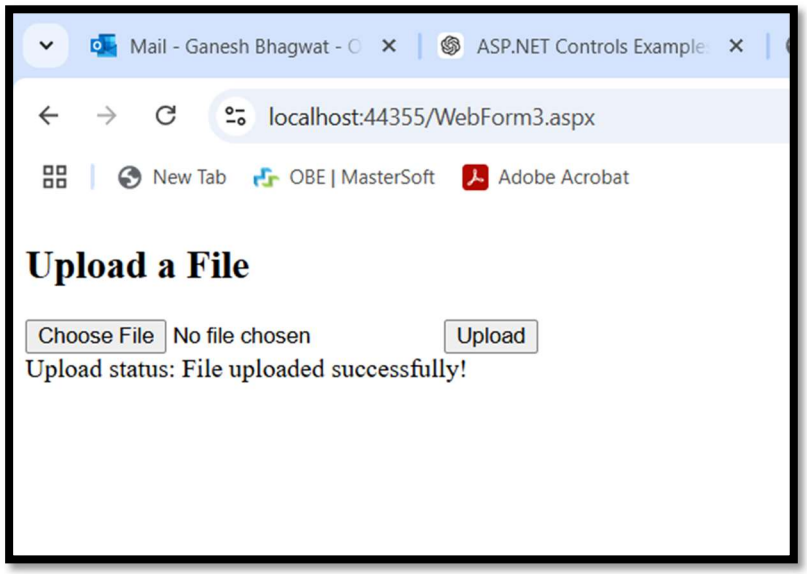
```
    </form>
</body>
</html>
```

FileUpload.aspx.cs

```
using System;
using System.Collections.Generic;
using System.IO;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace Practical_Number_3
{
   public partial class WebForm3 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {

      }

      protected void UploadButton_Click(object sender, EventArgs e)
      {
         if (FileUploadControl.HasFile)
         {
            try
            {
               string filename = Path.GetFileName(FileUploadControl.FileName);
               FileUploadControl.SaveAs(Server.MapPath("~/Uploads/") + filename);
               StatusLabel.Text = "Upload status: File uploaded successfully!";
            }
            catch (Exception ex)
            {
               StatusLabel.Text = "Upload status: Error - " + ex.Message;
            }
         }
         else
         {
            StatusLabel.Text = "Upload status: No file selected.";
         }
```

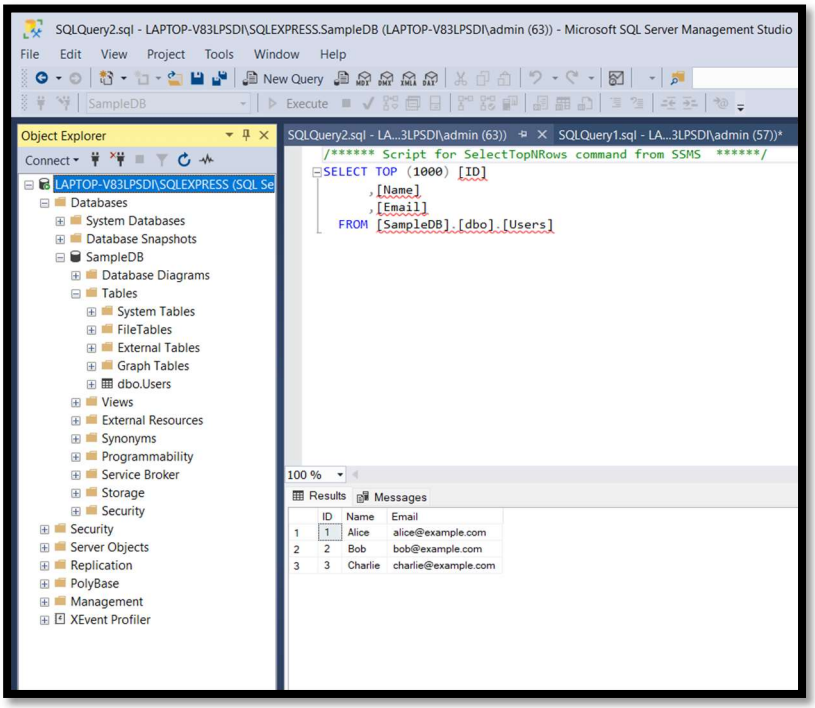
```
    }
  }
}
```

Output:


Mail - Ganesh Bhagwat - C
ASP.NET Controls Example
localhost:44355/WebForm3.aspx
New Tab
OBE | MasterSoft
Adobe Acrobat
Upload a File
Choose File
No file chosen
Upload
Upload status: File uploaded successfully!

Practical no.: 4
Webpage Demonstrating Connection-Oriented Architecture (ASP.NET Web Forms with SQL Server Database)

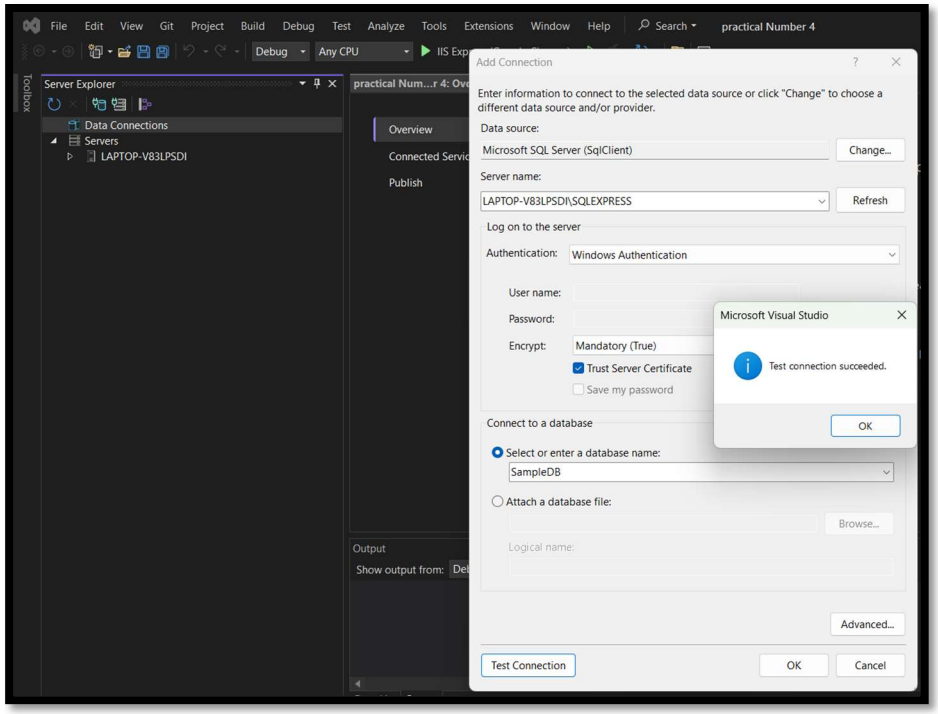
A **connection-oriented architecture** involves establishing a persistent connection between the client and server. This is typically demonstrated using **ADO.NET with SQL Server**, where a connection is opened, data is fetched, and then the connection is closed.

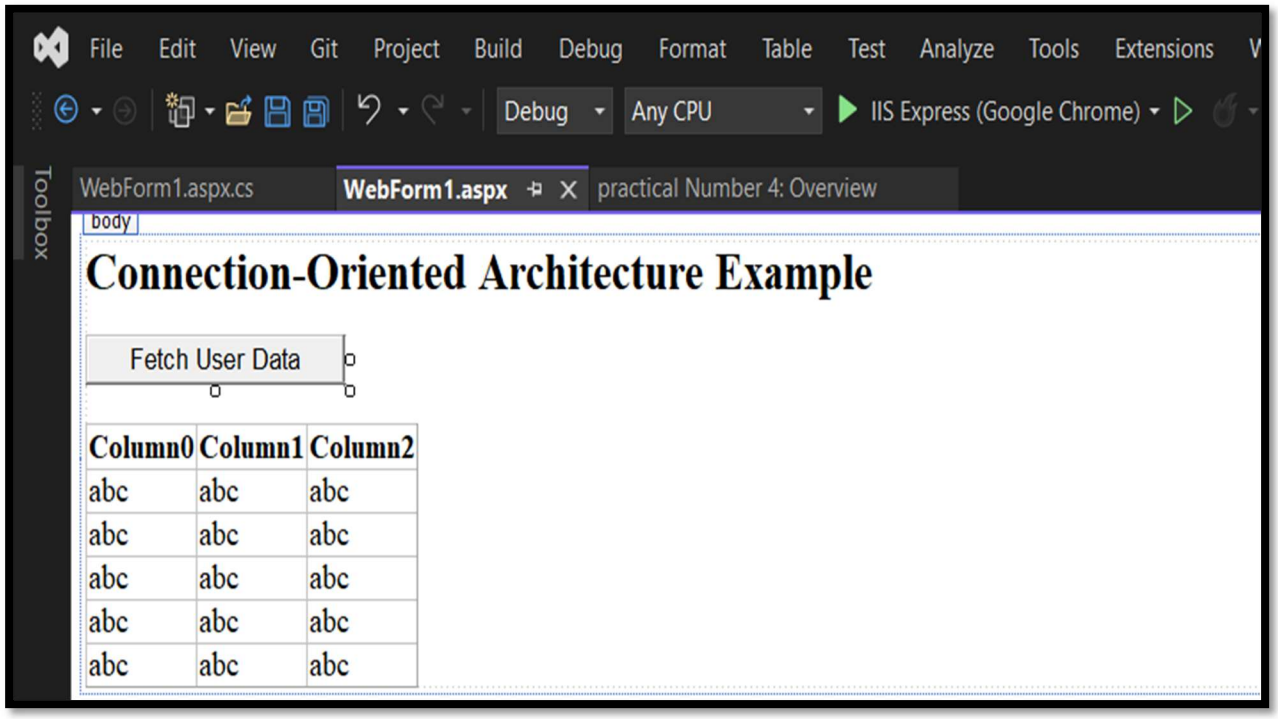


Create SQL Server Table

Run the following SQL script in **SQL Server Management Studio (SSMS)**:

```
CREATE DATABASE SampleDB;
USE SampleDB;

CREATE TABLE Users (
    ID INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(50),
    Email NVARCHAR(100)
);

INSERT INTO Users (Name, Email) VALUES
('Alice', 'alice@example.com'),
('Bob', 'bob@example.com'),
('Charlie', 'charlie@example.com');
```


File Edit View Git Project Build Debug Test Analyze Tools Extensions Window Help
practical Number 4
Debug
Any CPU
Server Explorer
Data Connections
Servers
LAPTOP-V83LPSDI
Add Connection
Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.
Data source:
Microsoft SQL Server (SqlClient)
Change...
Server name:
LAPTOP-V83LPSDI\SQLEXPRESS
Refresh
Log on to the server
Authentication:
Windows Authentication
User name:
Password:
Encrypt:
Mandatory (True)
Trust Server Certificate
Save my password
Microsoft Visual Studio
Test connection succeeded.
OK
Connect to a database
Select or enter a database name:
SampleDB
Attach a database file:
Browse...
Logical name:
Advanced...
Test Connection
OK
Cancel
Overview
Connected Servic
Publish
Output
Show output from:


File Edit View Git Project Build Debug Format Table Test Analyze Tools Extensions
Debug
Any CPU
IIS Express (Google Chrome)
Toolbox
WebForm1.aspx.cs
WebForm1.aspx
practical Number 4: Overview
body
Connection-Oriented Architecture Example
Fetch User Data
Column0 Column1 Column2
abc abc abc
abc abc abc
abc abc abc
abc abc abc
abc abc abc

Code:

WebForm1.aspx

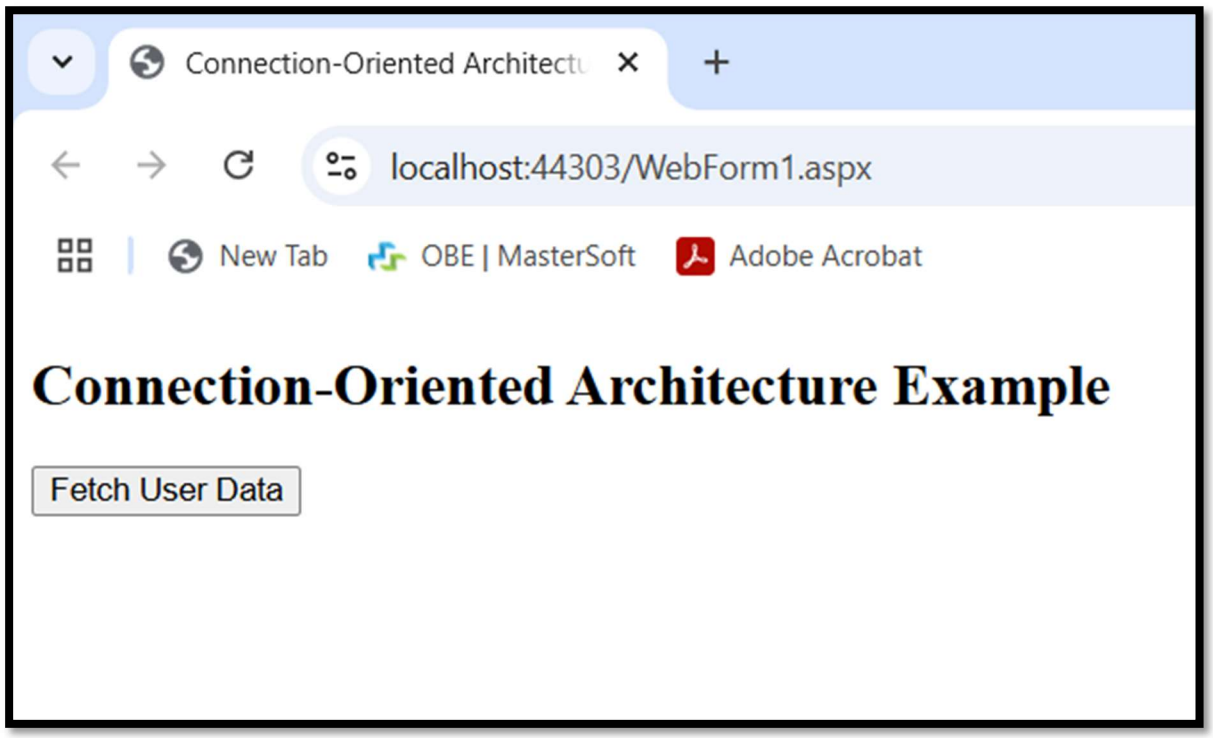
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="practical_Number_4.WebForm1" %>

<!DOCTYPE html>
<html>
<head runat="server">
  <title>Connection-Oriented Architecture</title>
</head>
<body>
  <form id="form1" runat="server">
    <h2>Connection-Oriented Architecture Example</h2>

    <asp:Button ID="FetchDataButton" runat="server" Text="Fetch User Data"
OnClick="FetchDataButton_Click" />
    <br /><br />

    <asp:GridView ID="UsersGridView" runat="server" AutoGenerateColumns="true"
BorderWidth="1" />

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

WebForm1.aspx.cs

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

namespace practical_Number_4
{
  public partial class WebForm1 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {
```

```
    }

    protected void FetchDataButton_Click(object sender, EventArgs e)
    {
        // Define the connection string (Update with your server details)
        string connectionString = "Data Source=LAPTOP-V83LPSDI\\SQLEXPRESS;Initial
Catalog=SampleDB;Integrated Security=True";

        // Create a connection object
        using (SqlConnection conn = new SqlConnection(connectionString))
        {
            try
            {
                conn.Open();  // Open the connection

                // SQL query to fetch data
                string query = "SELECT * FROM Users";
                SqlDataAdapter da = new SqlDataAdapter(query, conn);
                DataTable dt = new DataTable();
                da.Fill(dt);

                // Bind data to GridView
                UsersGridView.DataSource = dt;
                UsersGridView.DataBind();
            }
            catch (Exception ex)
            {
                Response.Write("<script>alert('Error: " + ex.Message + "');</script>");
            }
        } // Connection closes au
    }
  }
}
```

Output:


Connection-Oriented Architecture
localhost:44303/WebForm1.aspx
New Tab
OBE | MasterSoft
Adobe Acrobat
Connection-Oriented Architecture Example
Fetch User Data

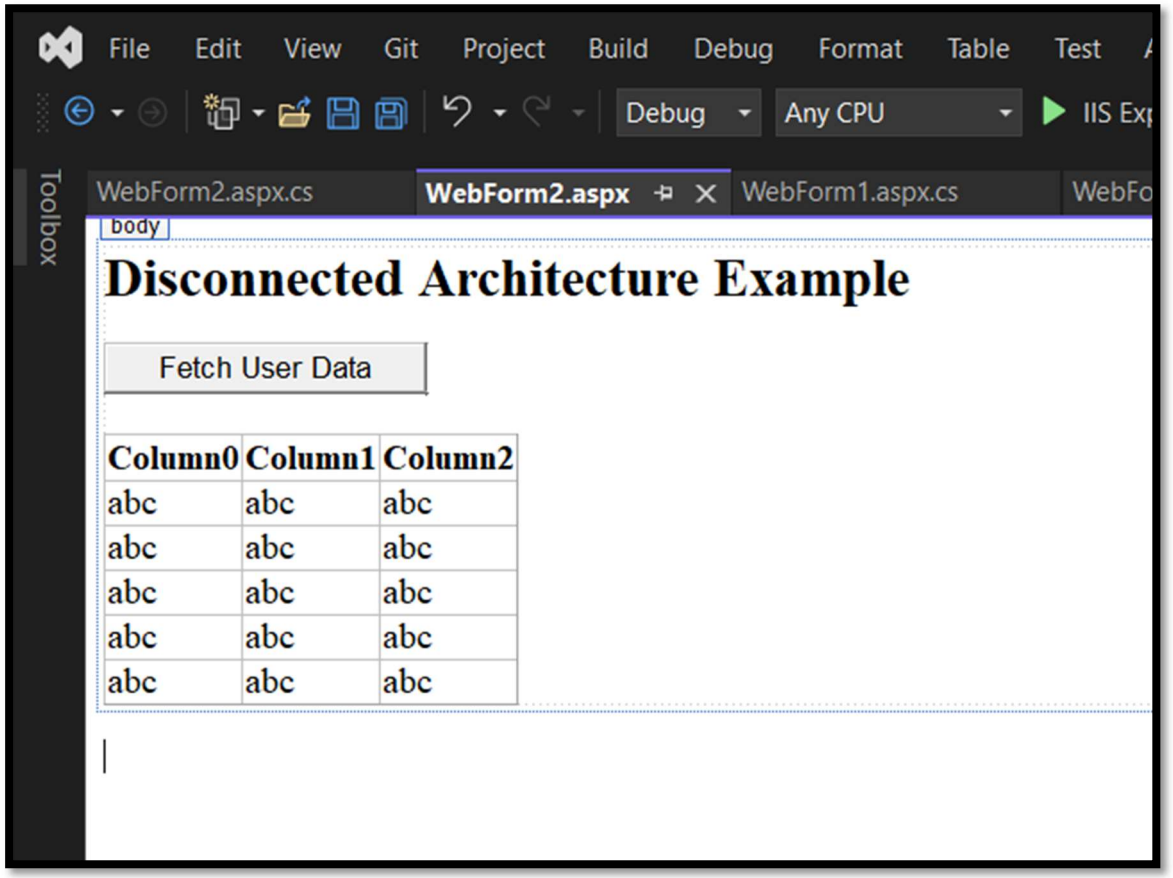

Connection-Oriented Architecture
localhost:44303/WebForm1.aspx
New Tab
OBE | MasterSoft
Adobe Acrobat
Connection-Oriented Architecture Example
Fetch User Data
ID Name Email
1 Alice alice@example.com
2 Bob bob@example.com
3 Charlie charlie@example.com

Practical no.: 5
Webpage Demonstrating Disconnected Architecture (ASP.NET Web Forms with SQL Server Database)

Create SQL Server Table

Run the following SQL script in **SQL Server Management Studio (SSMS)**:

```
CREATE DATABASE SampleDB;
USE SampleDB;

CREATE TABLE Users (
    ID INT IDENTITY(1,1) PRIMARY KEY,
    Name NVARCHAR(50),
    Email NVARCHAR(100)
);

INSERT INTO Users (Name, Email) VALUES
('Alice', 'alice@example.com'),
('Bob', 'bob@example.com'),
('Charlie', 'charlie@example.com');
```



Code:

WebForm2.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="practical_Number_4.WebForm2" %>
<!DOCTYPE html>
<html>
<head runat="server">
  <title>Disconnected Architecture Example</title>
</head>
<body>
  <form id="form1" runat="server">

    <h2>Disconnected Architecture Example</h2>

    <asp:Button ID="FetchDataButton" runat="server" Text="Fetch User Data"
OnClick="FetchDataButton_Click" />
    <br /><br />

    <asp:GridView ID="UsersGridView" runat="server" AutoGenerateColumns="true" BorderWidth="1"
/>

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

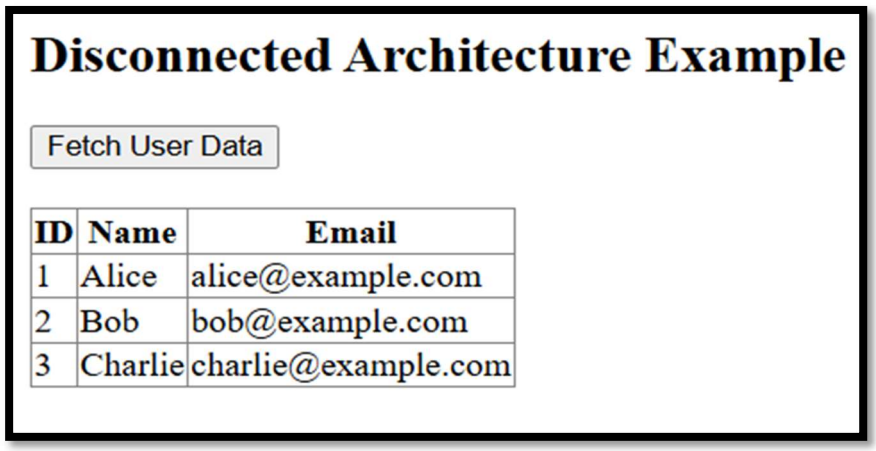
WebForm2.aspx.cs

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

namespace practical_Number_4
{
  public partial class WebForm2 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void FetchDataButton_Click(object sender, EventArgs e)
```

```
    {
        // Define the connection string (Update with your server details)
        string connectionString = "Data Source=LAPTOP-V83LPSDI\\SQLEXPRESS;Initial
Catalog=SampleDB;Integrated Security=True";

        // Create objects for disconnected architecture
        SqlDataAdapter da;
        DataSet ds = new DataSet();

        try
        {
            using (SqlConnection conn = new SqlConnection(connectionString))
            {

                // SQL query to fetch data
                string query = "SELECT * FROM Users";
                da = new SqlDataAdapter(query, conn);

                // Fill dataset with data from the database
                da.Fill(ds, "Users");

            } // Connection is closed after this block

            // Bind data to GridView (data remains in memory)
            UsersGridView.DataSource = ds.Tables["Users"];
            UsersGridView.DataBind();

        }
        catch (Exception ex)
        {
            Response.Write("<script>alert('Error: " + ex.Message + "');</script>");
        }
    }
  }
}
```

Output:

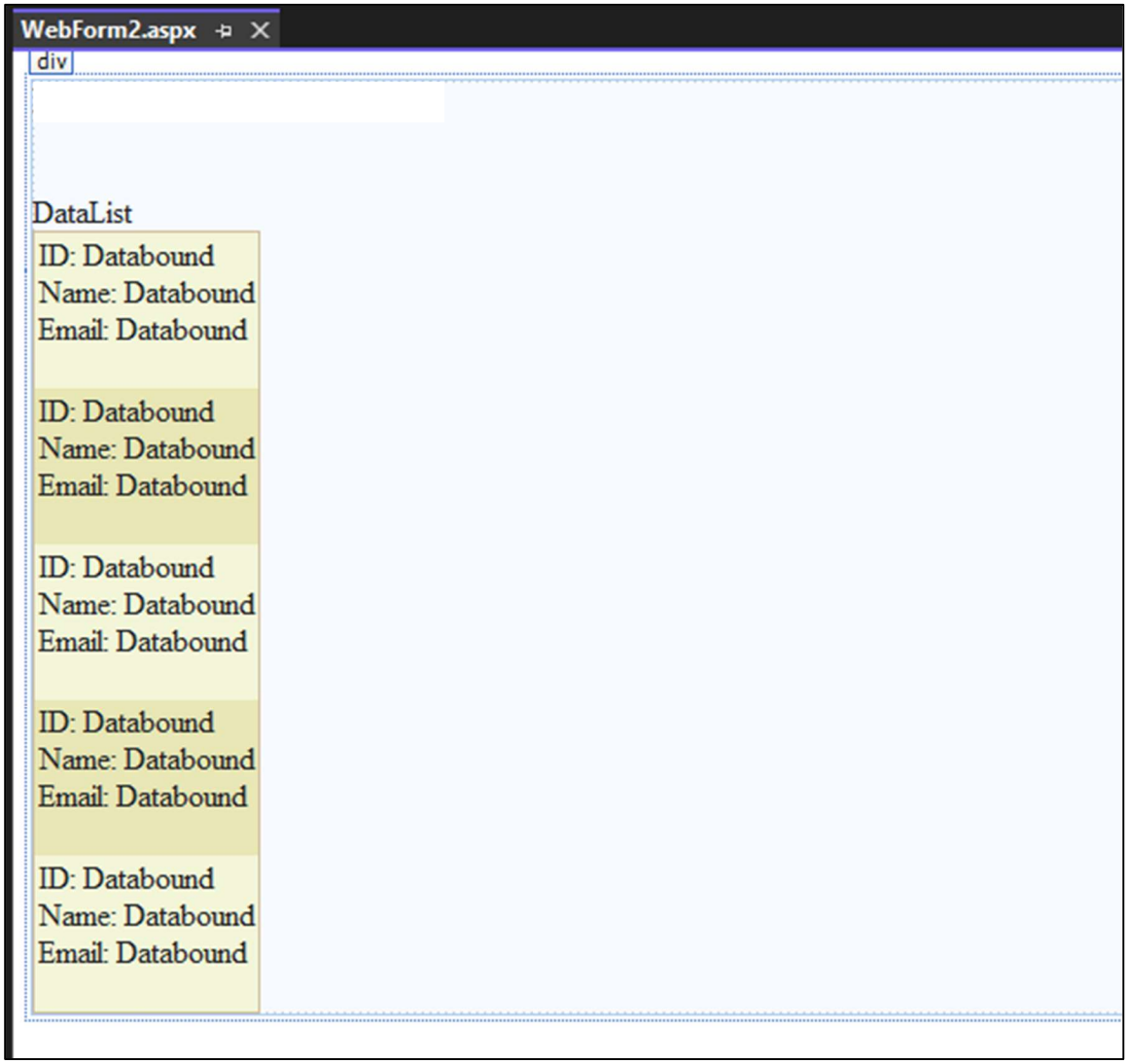

Disconnected Architecture Exa
localhost:44303/WebForm2.aspx
New Tab
OBE | MasterSoft
Adobe Acrobat
Disconnected Architecture Example
Fetch User Data


Disconnected Architecture Example
Fetch User Data
ID Name Email
1 Alice alice@example.com
2 Bob bob@example.com
3 Charlie charlie@example.com

Practical no.: 6

Create a webpage that demonstrates the use of data bound controls of ASP.NET.

Data Controls:

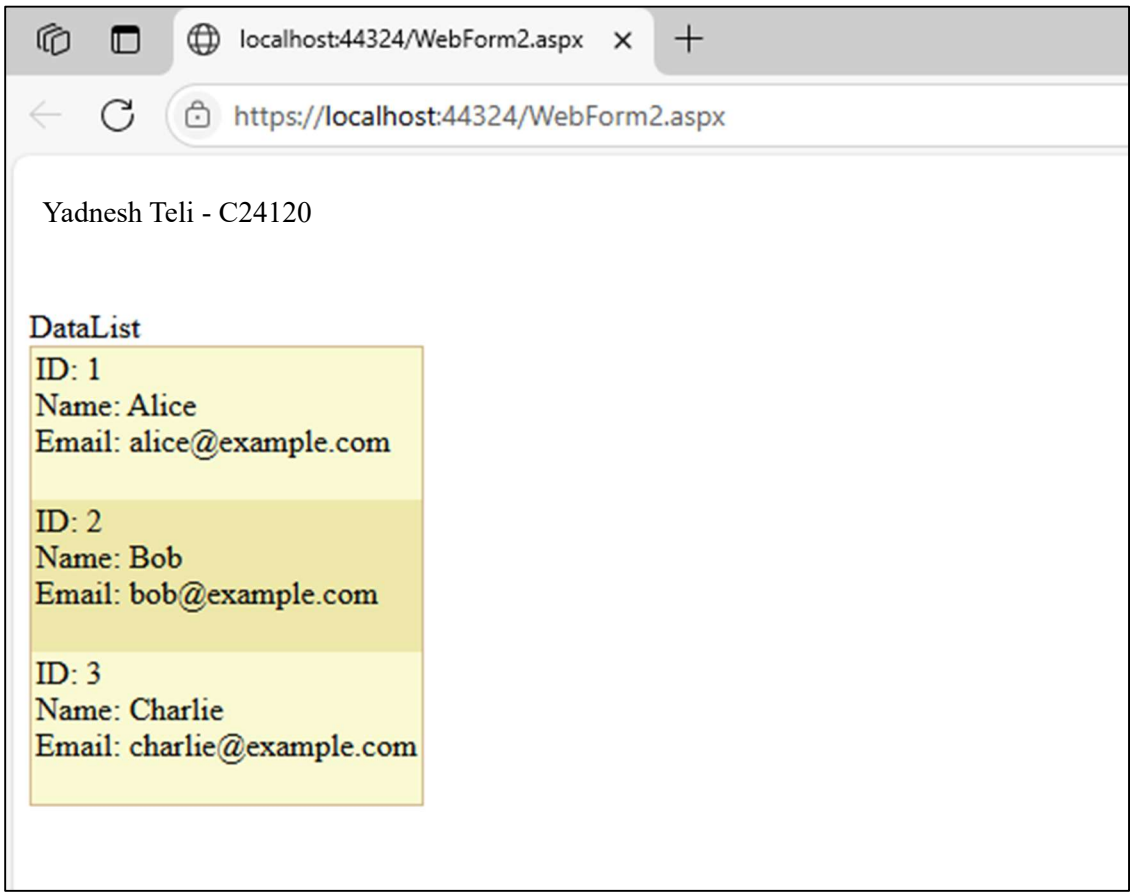


Code:

WebForm2.aspx:

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="PRACTICAL_5.WebForm2" %>
<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <h3> Yadnesh Teli - C24120</h3>
            <br />
            <asp:Label ID="Label2" runat="server" Text="DataList"></asp:Label>
            <br />
            <asp:DataList ID="DataList1" runat="server"
BackColor="LightGoldenrodYellow" BorderColor="Tan" BorderWidth="1px" CellPadding="2"
DataKeyField="ID" DataSourceID="SqlDataSource2" ForeColor="Black">
                <AlternatingItemStyle BackColor="PaleGoldenrod" />
                <FooterStyle BackColor="Tan" />
                <HeaderStyle BackColor="Tan" Font-Bold="True" />
                <ItemTemplate>
                    ID:
                    <asp:Label ID="IDLabel" runat="server" Text='<%# Eval("ID") %>'
/>
                    <br />
                    Name:
                    <asp:Label ID="NameLabel" runat="server" Text='<%# Eval("Name")
%>' />
                    <br />
                    Email:
                    <asp:Label ID="EmailLabel" runat="server" Text='<%#
Eval("Email") %>' />
                    <br />
<br />
                </ItemTemplate>
                <SelectedItemStyle BackColor="DarkSlateBlue" ForeColor="GhostWhite"
/>
            </asp:DataList>

            <asp:SqlDataSource ID="SqlDataSource2" runat="server"
ConnectionString="<%$ ConnectionStrings:SampleDBConnectionString %>"
ProviderName="<%$ ConnectionStrings:SampleDBConnectionString.ProviderName %>"
SelectCommand="SELECT * FROM [Users]"></asp:SqlDataSource>
        </div>
    </form>
</body>
</html>
```

Output:


localhost:44324/WebForm2.aspx
https://localhost:44324/WebForm2.aspx
Yadnesh Teli - C24120
DataList
ID: 1
Name: Alice
Email: alice@example.com
ID: 2
Name: Bob
Email: bob@example.com
ID: 3
Name: Charlie
Email: charlie@example.com

Practical no.: 7
Design a webpage to demonstrate the working of a simple stored procedure.

The webpage will use **ASP.NET Web Forms** and **SQL Server** to retrieve and display user details.

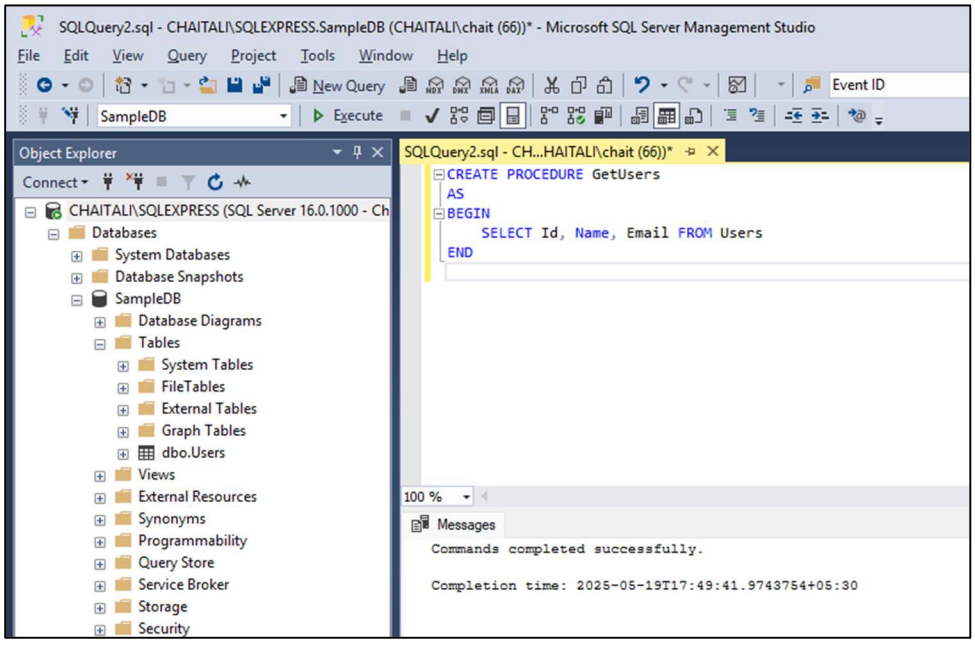
Steps to Implement:

1. **Create a Stored Procedure in SQL Server**
2. **Design an ASP.NET Web Form (ASPX Page)**
3. **Connect to the Database and Execute the Stored Procedure**
4. **Display the Results in a GridView**

Create the Stored Procedure in SQL Server

Run this SQL script in **SQL Server Management Studio (SSMS)**:

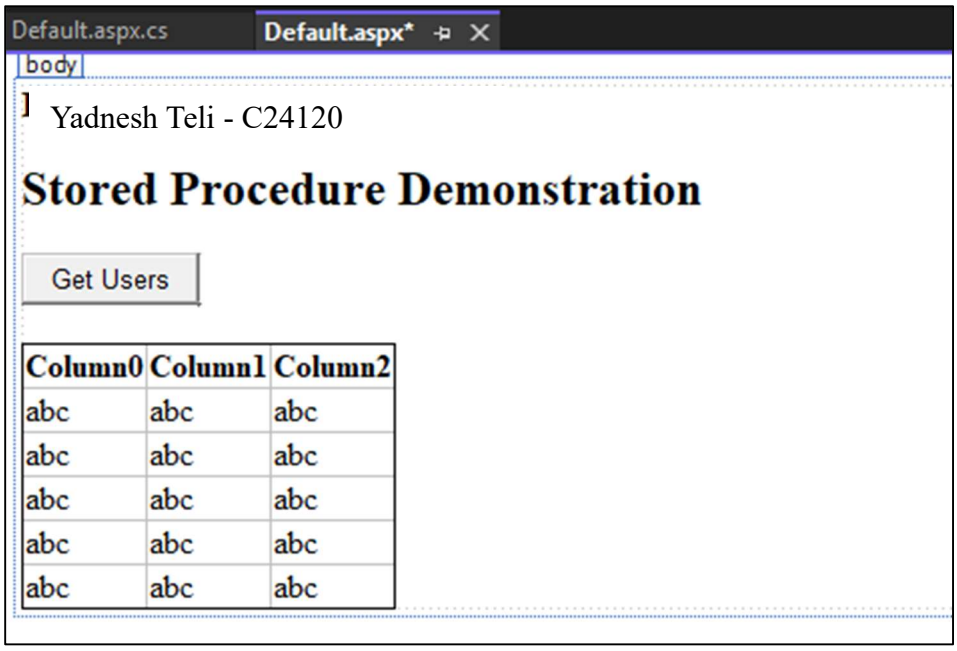
```
CREATE PROCEDURE GetUsers
AS
BEGIN
    SELECT Id, Name, Email FROM Users
END
```



---

2. Create an ASP.NET Web Application in Visual Studio

- Open **Visual Studio**
- Create a new **ASP.NET Web Forms Application**
- Add a **Web Form (Default.aspx)**



Code:

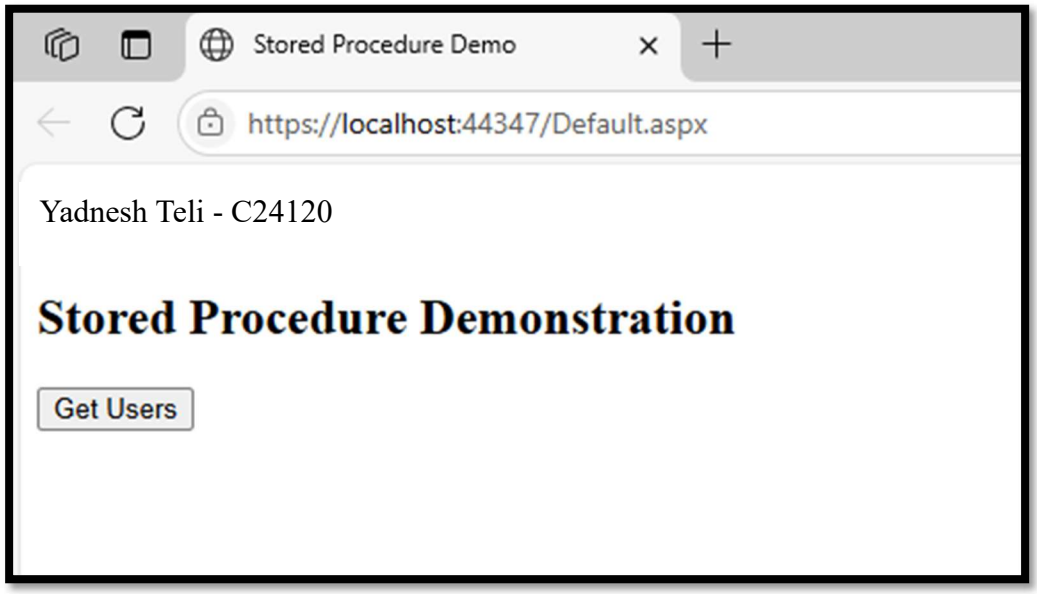
Default.aspx

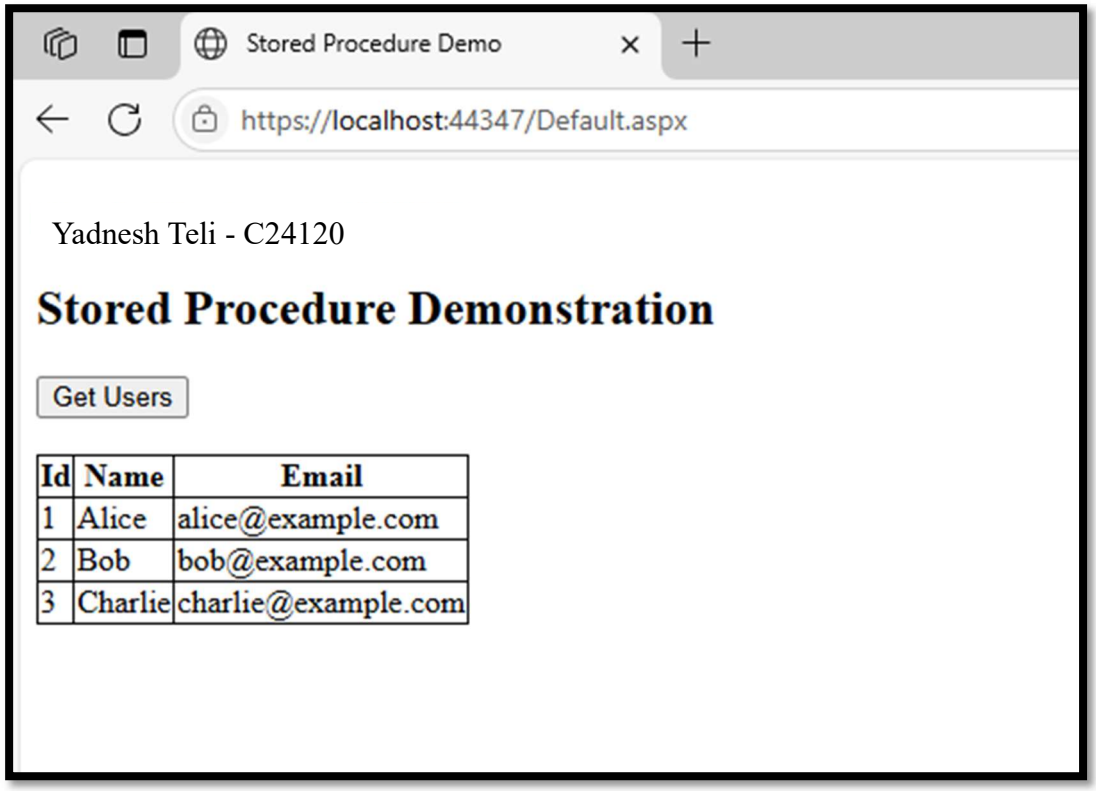
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Default.aspx.cs"
Inherits="PRACTICAL_7.Default" %>
<!DOCTYPE html>
<html>
<head>
   <title>Stored Procedure Demo</title>
</head>
<body>
   <form id="form1" runat="server">
      <h3> Yadnesh Teli - C24120</h3>
      <h2>Stored Procedure Demonstration</h2>

      <asp:Button ID="btnGetUsers" runat="server" Text="Get Users" OnClick="btnGetUsers_Click" />
      <br /><br />

      <asp:GridView ID="GridViewUsers" runat="server" AutoGenerateColumns="true"
BorderColor="Black" BorderWidth="1px" />
   </form>
</body>
</html>
```

Default.aspx.cs

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

namespace PRACTICAL_7
{
       public partial class Default : System.Web.UI.Page
       {
              protected void Page_Load(object sender, EventArgs e)
              {

              }
        protected void btnGetUsers_Click(object sender, EventArgs e)
        {
            // Connection string from Web.config
            string connStr = "Data Source=CHAITALI\\SQLEXPRESS;Initial
Catalog=SampleDB;Integrated Security=True;";

            using (SqlConnection conn = new SqlConnection(connStr))
            {
                using (SqlCommand cmd = new SqlCommand("GetUsers", conn))
                {
                    cmd.CommandType = CommandType.StoredProcedure;
                    conn.Open();

                    SqlDataAdapter da = new SqlDataAdapter(cmd);
                    DataTable dt = new DataTable();
                    da.Fill(dt);

                    GridViewUsers.DataSource = dt;
                    GridViewUsers.DataBind();

                }
            }
        }

    }
}
```

Output:


Stored Procedure Demo
https://localhost:44347/Default.aspx
Yadnesh Teli - C24120
Stored Procedure Demonstration
Get Users


Stored Procedure Demo
https://localhost:44347/Default.aspx
Yadnesh Teli - C24120
Stored Procedure Demonstration
Get Users
Id Name Email
1 Alice alice@example.com
2 Bob bob@example.com
3 Charlie charlie@example.com

Practical no.: 8
Design a webpage to demonstrate the working of parameterized stored procedure.

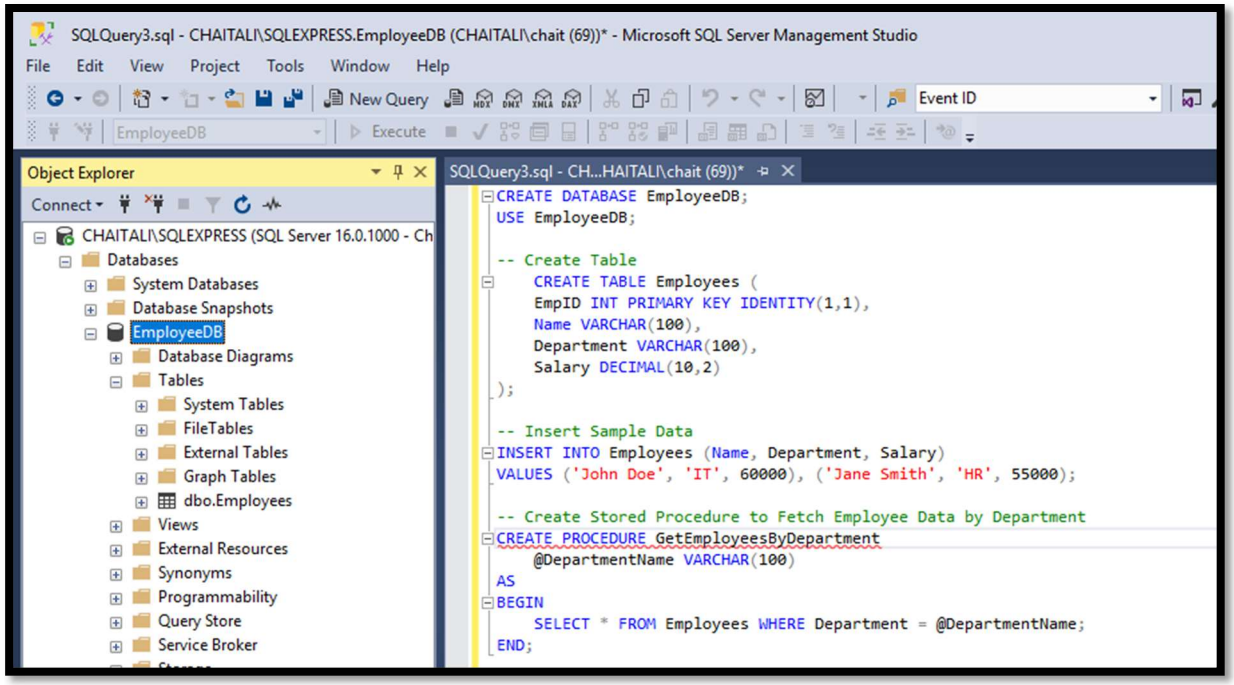
Steps to Implement

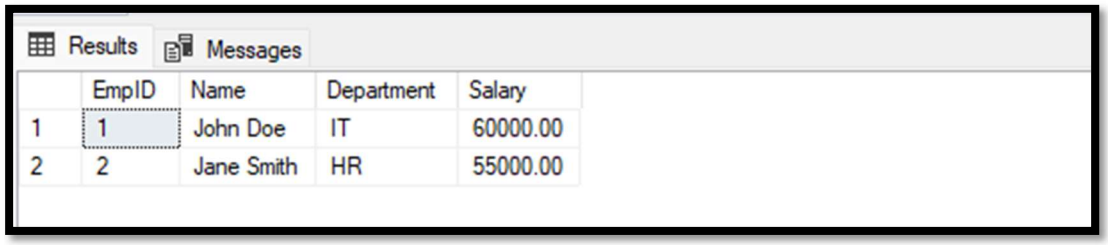
1. **Create a SQL Server Database** with a stored procedure.
2. **Design an ASP.NET Web Form** using Visual Studio.
3. **Use ADO.NET to call the stored procedure** from C#.
4. **Display the output on the webpage.**

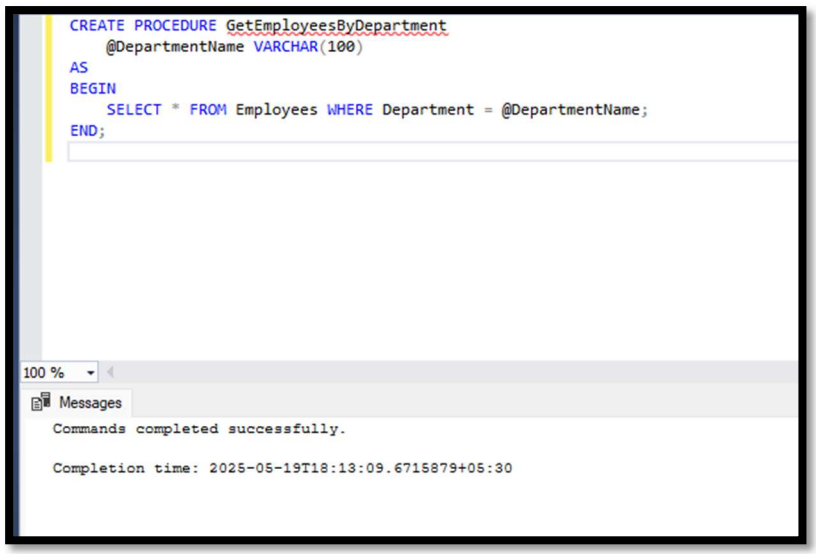
SQL Server: Create a Database & Stored Procedure

Run the following SQL commands in **SQL Server Management Studio (SSMS):**

```
CREATE DATABASE EmployeeDB;
USE EmployeeDB;

-- Create Table
    CREATE TABLE Employees (
    EmpID INT PRIMARY KEY IDENTITY(1,1),
    Name VARCHAR(100),
    Department VARCHAR(100),
    Salary DECIMAL(10,2)
);

-- Insert Sample Data
INSERT INTO Employees (Name, Department, Salary)
VALUES ('John Doe', 'IT', 60000), ('Jane Smith', 'HR', 55000);

-- Create Stored Procedure to Fetch Employee Data by Department
CREATE PROCEDURE GetEmployeesByDepartment
    @DepartmentName VARCHAR(100)
AS
BEGIN
    SELECT * FROM Employees WHERE Department = @DepartmentName;
END;
```

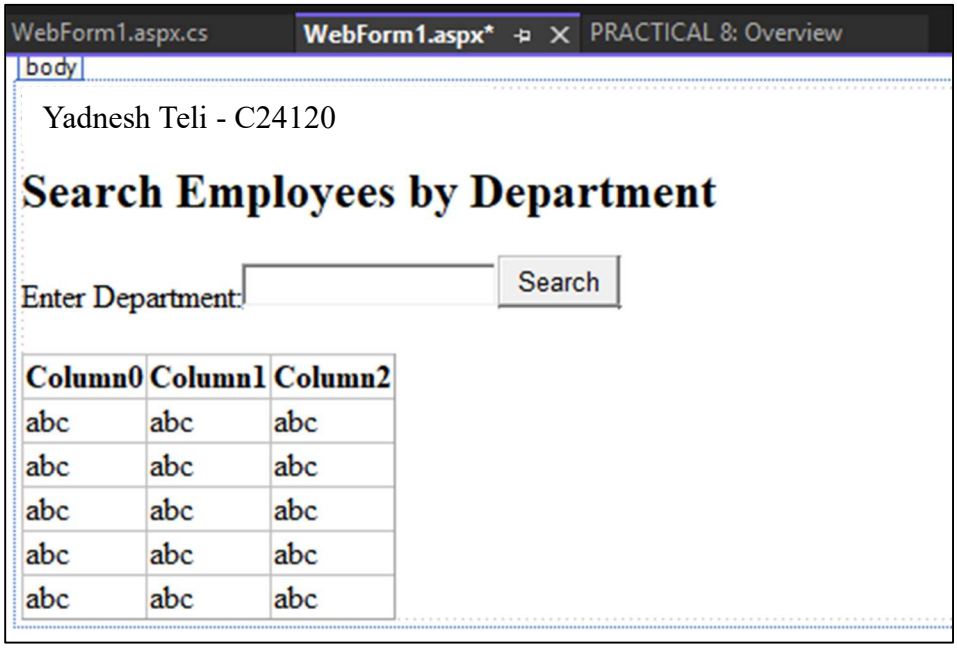

SQLQuery3.sql - CHAITALI\SQLEXPRESS.EmployeeDB (CHAITALI\chait (69))* - Microsoft SQL Server Management Studio
File Edit View Project Tools Window Help
New Query
Event ID
EmployeeDB
Execute
Object Explorer
Connect
CHAITALI\SQLEXPRESS (SQL Server 16.0.1000 - Ch
Databases
System Databases
Database Snapshots
EmployeeDB
Database Diagrams
Tables
System Tables
FileTables
External Tables
Graph Tables
dbo.Employees
Views
External Resources
Synonyms
Programmability
Query Store
Service Broker
SQLQuery3.sql - CH...HAITALI\chait (69))*
CREATE DATABASE EmployeeDB;
USE EmployeeDB;
-- Create Table
CREATE TABLE Employees (
EmpID INT PRIMARY KEY IDENTITY(1,1),
Name VARCHAR(100),
Department VARCHAR(100),
Salary DECIMAL(10,2)
);
-- Insert Sample Data
INSERT INTO Employees (Name, Department, Salary)
VALUES ('John Doe', 'IT', 60000), ('Jane Smith', 'HR', 55000);
-- Create Stored Procedure to Fetch Employee Data by Department
CREATE PROCEDURE GetEmployeesByDepartment
@DepartmentName VARCHAR(100)
AS
BEGIN
SELECT * FROM Employees WHERE Department = @DepartmentName;
END;


Results
Messages
EmpID Name Department Salary
1 1 John Doe IT 60000.00
2 2 Jane Smith HR 55000.00


CREATE PROCEDURE GetEmployeesByDepartment
@DepartmentName VARCHAR(100)
AS
BEGIN
SELECT * FROM Employees WHERE Department = @DepartmentName;
END;
100 %
Messages
Commands completed successfully.
Completion time: 2025-05-19T18:13:09.6715879+05:30

Code:

WebForm1.aspx

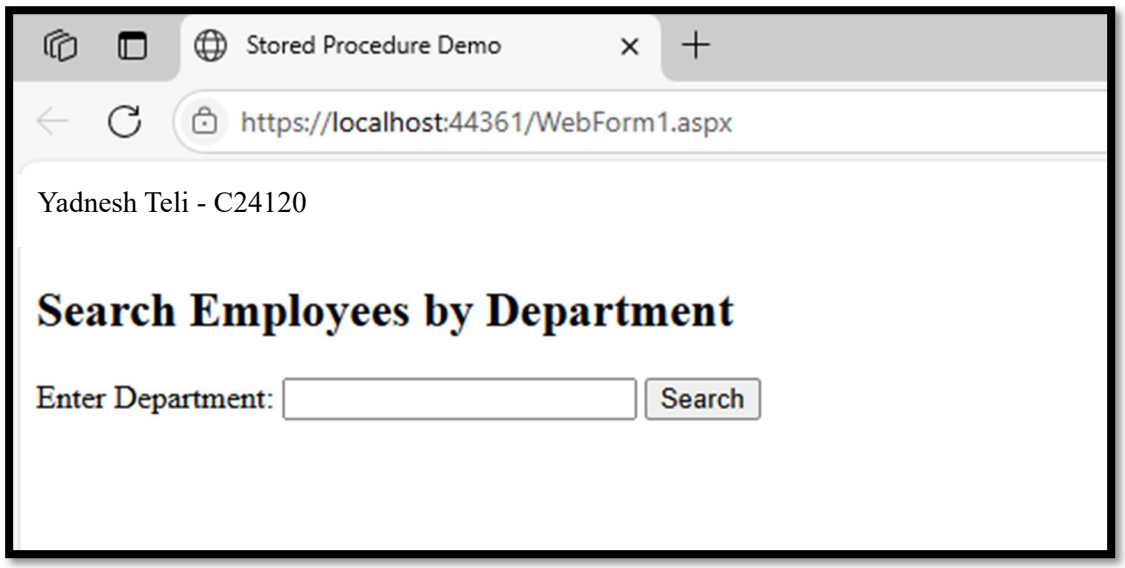


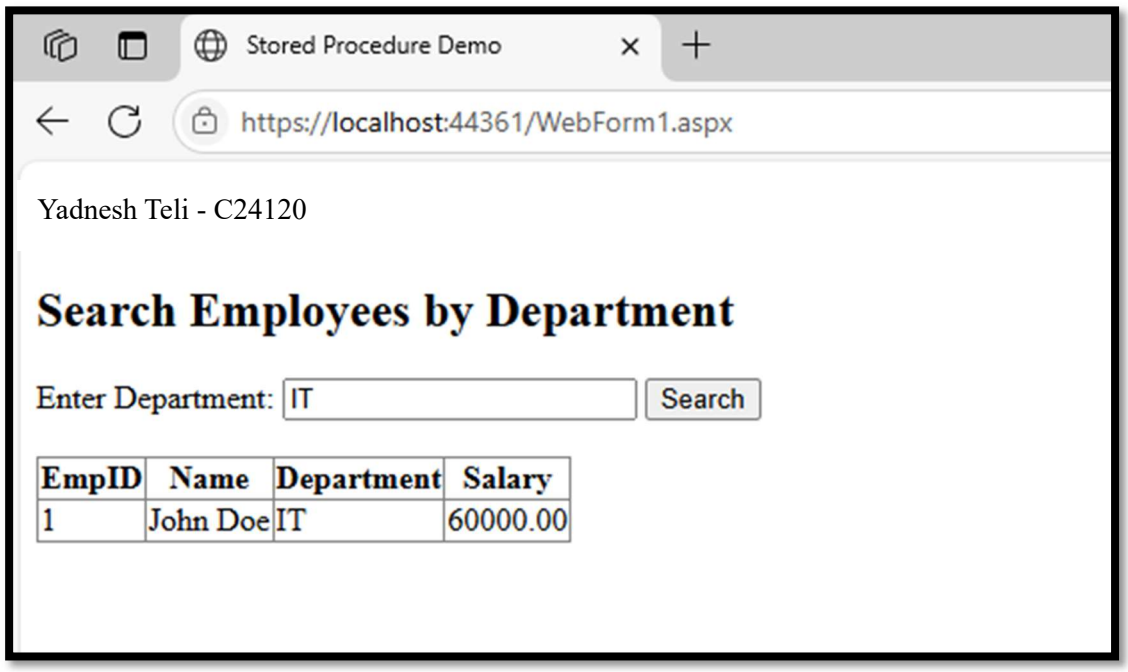

```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="PRACTICAL_8.WebForm1" %>
<!DOCTYPE html>
<html lang="en">
<head runat="server">
  <title>Stored Procedure Demo</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h3> Yadnesh Teli - C24120</h3>
      <h2>Search Employees by Department</h2>
      <asp:Label runat="server" Text="Enter Department:"></asp:Label>
      <asp:TextBox ID="txtDepartment" runat="server"></asp:TextBox>
      <asp:Button ID="btnSearch" runat="server" Text="Search" OnClick="btnSearch_Click"/>

      <br /><br />
      <asp:GridView ID="gvEmployees" runat="server" AutoGenerateColumns="True"></asp:GridView>
    </div>
  </form>
</body>
</html>
```

WebForm1.aspx.cs

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

namespace PRACTICAL_8
{
          public partial class WebForm1 : System.Web.UI.Page
          {
                    protected void Page_Load(object sender, EventArgs e)
                    {

                    }
     protected void btnSearch_Click(object sender, EventArgs e)
     {
        string connStr = "Data Source=YADNESH\\SQLEXPRESS;Initial Catalog=EmployeeDB;Integrated
Security=True;";

        {
           SqlConnection conn = new SqlConnection(connStr);
           SqlCommand cmd = new SqlCommand("GetEmployeesByDepartment", conn);
           cmd.CommandType = CommandType.StoredProcedure;
           cmd.Parameters.AddWithValue("@DepartmentName", txtDepartment.Text);

           SqlDataAdapter da = new SqlDataAdapter(cmd);
           DataTable dt = new DataTable();
           da.Fill(dt);

           gvEmployees.DataSource = dt;
           gvEmployees.DataBind();
        }
     }
  }
}
```

Output:


Stored Procedure Demo
https://localhost:44361/WebForm1.aspx
Yadnesh Teli - C24120
Search Employees by Department
Enter Department:
Search

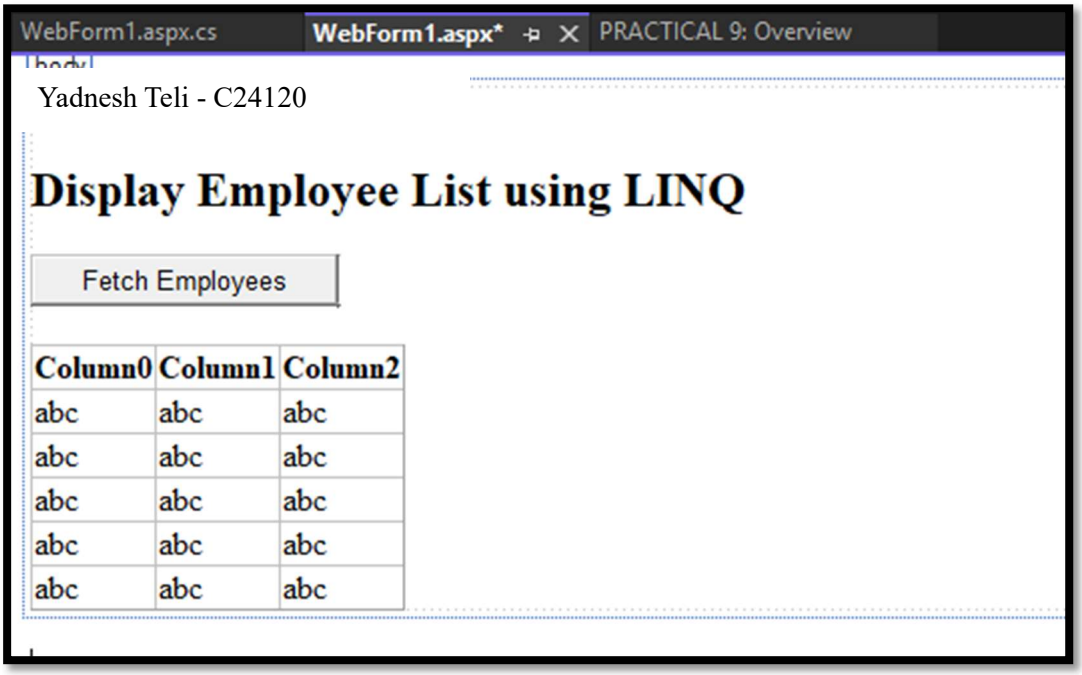

Stored Procedure Demo
https://localhost:44361/WebForm1.aspx
Yadnesh Teli - C24120
Search Employees by Department
Enter Department: IT
Search
EmpID Name Department Salary
1 John Doe IT 60000.00

Practical no.: 9
Design a webpage to display the use of LINQ.

Steps to Implement

1. **Create an ASP.NET Web Forms project** in Visual Studio.
2. **Use LINQ to query a list of employees** (In-Memory Collection).
3. **Bind the LINQ results to a GridView** for display.

Code:
WebForm1.aspx




```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="PRACTICAL_9.WebForm1" %>

<!DOCTYPE html>
<html lang="en">
<head runat="server">
    <title>LINQ Demo</title>
</head>
<body>
    <form id="form1" runat="server">
        <div>
            <h3> Akhila Nare - C24069 </h3>
            <h2>Display Employee List using LINQ</h2>
            <asp:Button ID="btnFetchData" runat="server" Text="Fetch Employees"
OnClick="btnFetchData_Click"/>
```

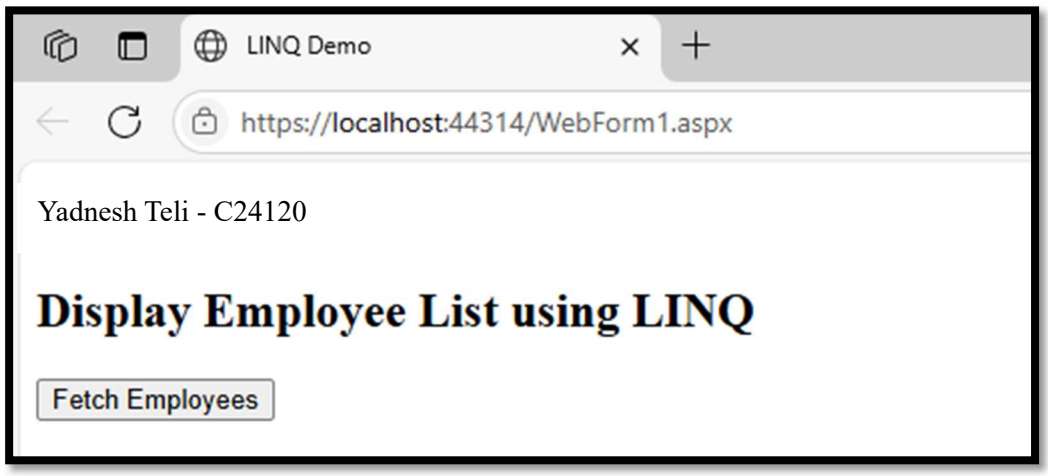
```
            <br /><br />
            <asp:GridView ID="gvEmployees" runat="server"
AutoGenerateColumns="True"></asp:GridView>
        </div>
    </form>
</body>
</html>
```

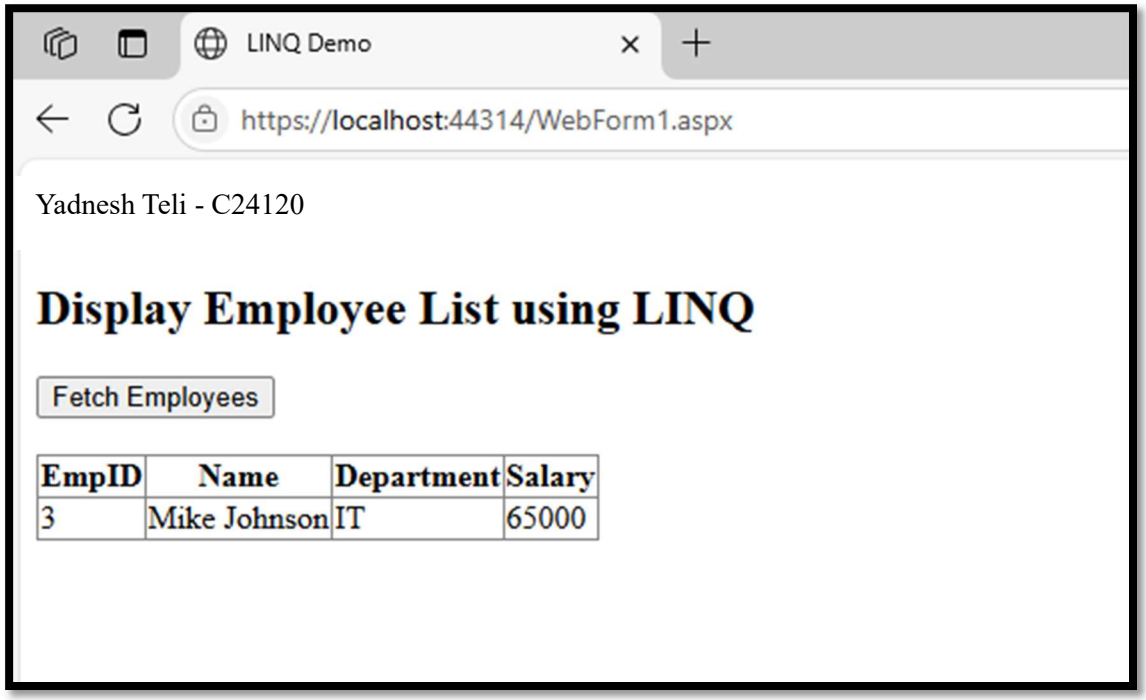
WebForm1.aspx.cs

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

namespace PRACTICAL_9
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }
        public class Employee
        {
            public int EmpID { get; set; }
            public string Name { get; set; }
            public string Department { get; set; }
            public decimal Salary { get; set; }
        }

        // Sample Employee Data (In-Memory Collection)
        private List<Employee> employees = new List<Employee>
        {
            new Employee { EmpID = 1, Name = "John Doe", Department = "IT", Salary =
60000 },
            new Employee { EmpID = 2, Name = "Jane Smith", Department = "HR", Salary
= 55000 },
            new Employee { EmpID = 3, Name = "Mike Johnson", Department = "IT",
Salary = 65000 },
            new Employee { EmpID = 4, Name = "Emily Davis", Department = "Finance",
Salary = 70000 }
        };


        protected void btnFetchData_Click(object sender, EventArgs e)
        {
            // Use LINQ to fetch IT department employees with salary > 60,000
            var result = from emp in employees
                         where emp.Department == "IT" && emp.Salary > 60000
                         select emp;

            // Bind data to GridView
            gvEmployees.DataSource = result.ToList();
            gvEmployees.DataBind();
```

```
        }
    }
}
```

Output:


LINQ Demo
https://localhost:44314/WebForm1.aspx
Yadnesh Teli - C24120
Display Employee List using LINQ
Fetch Employees


LINQ Demo
https://localhost:44314/WebForm1.aspx
Yadnesh Teli - C24120
Display Employee List using LINQ
Fetch Employees
EmpID Name Department Salary
3 Mike Johnson IT 65000

Practical no.: 10
Build websites to demonstrate the working of entity frameworks in dot net.

Steps to Implement

1. **Create a SQL Server Database & Table**
2. **Create an ASP.NET Web Application in Visual Studio**
3. **Install & Configure Entity Framework (EF) ORM**
4. **Use EF to perform CRUD operations**
5. **Display data in GridView & allow users to Add, Edit, Delete records**

Create a Database & Table

Open **SQL Server Management Studio (SSMS)** and execute:

```
CREATE DATABASE EmployeeDB;
USE EmployeeDB;
CREATE TABLE Employees (
   EmpID INT PRIMARY KEY IDENTITY(1,1),
   Name NVARCHAR(100),
   Department NVARCHAR(100),
   Salary DECIMAL(10,2)
);
INSERT INTO Employees (Name, Department, Salary)
VALUES ('John Doe', 'IT', 60000), ('Jane Smith', 'HR', 55000);
```

---

2 Create an ASP.NET Web Application

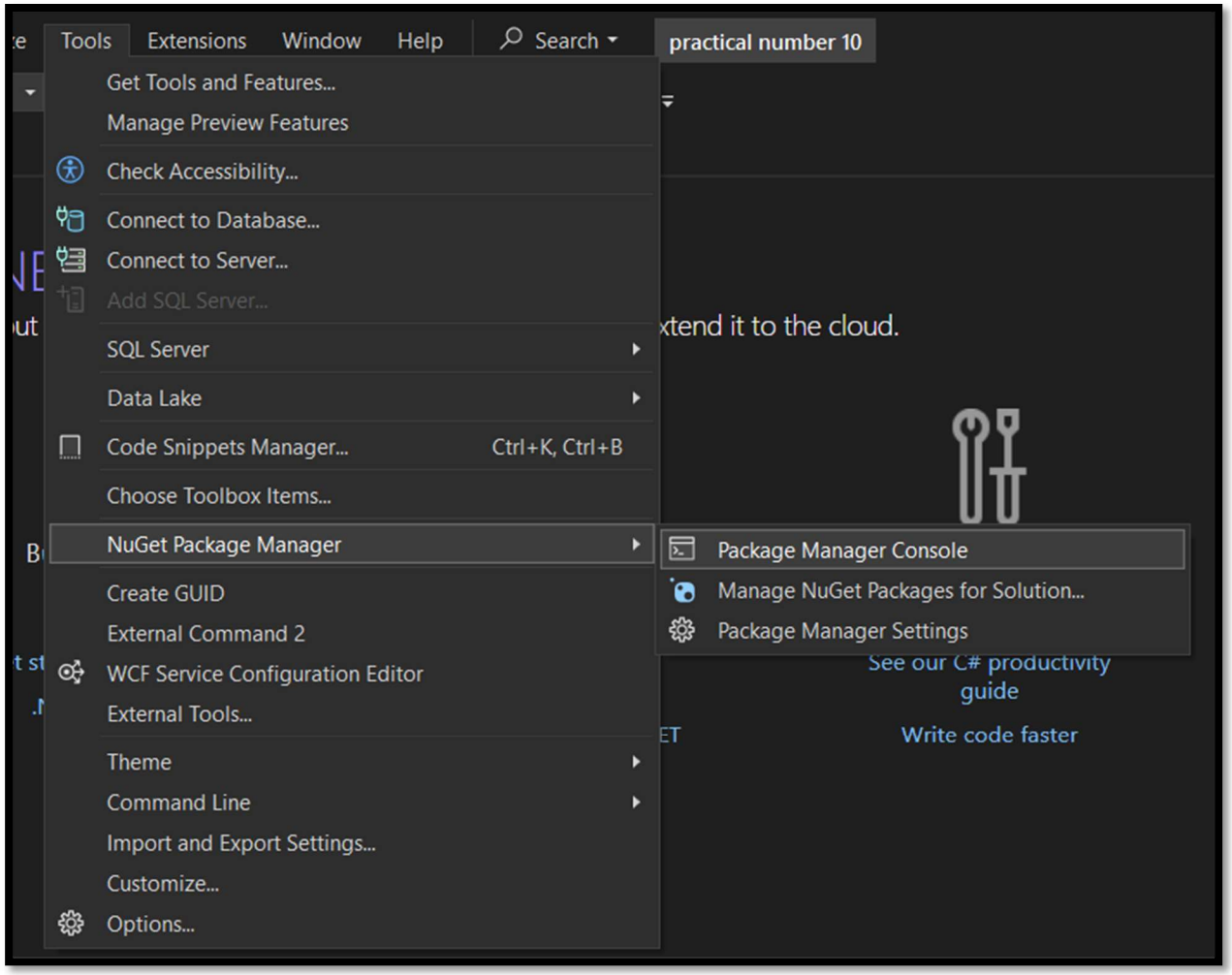
1. Open **Visual Studio** → Create a **New Project** → Choose **ASP.NET Web Application (.NET Framework)**
2. Select **Web Forms** and click **Create**

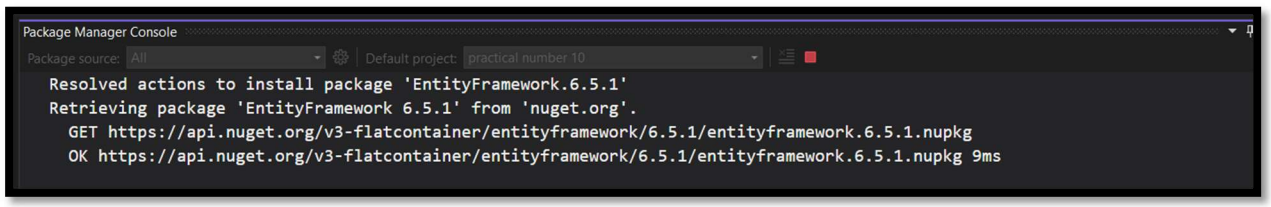
---

3 Install Entity Framework

1. Open **Package Manager Console** (Tools → NuGet Package Manager → Package Manager Console)
2. Run the command:

```
Install-Package EntityFramework
```

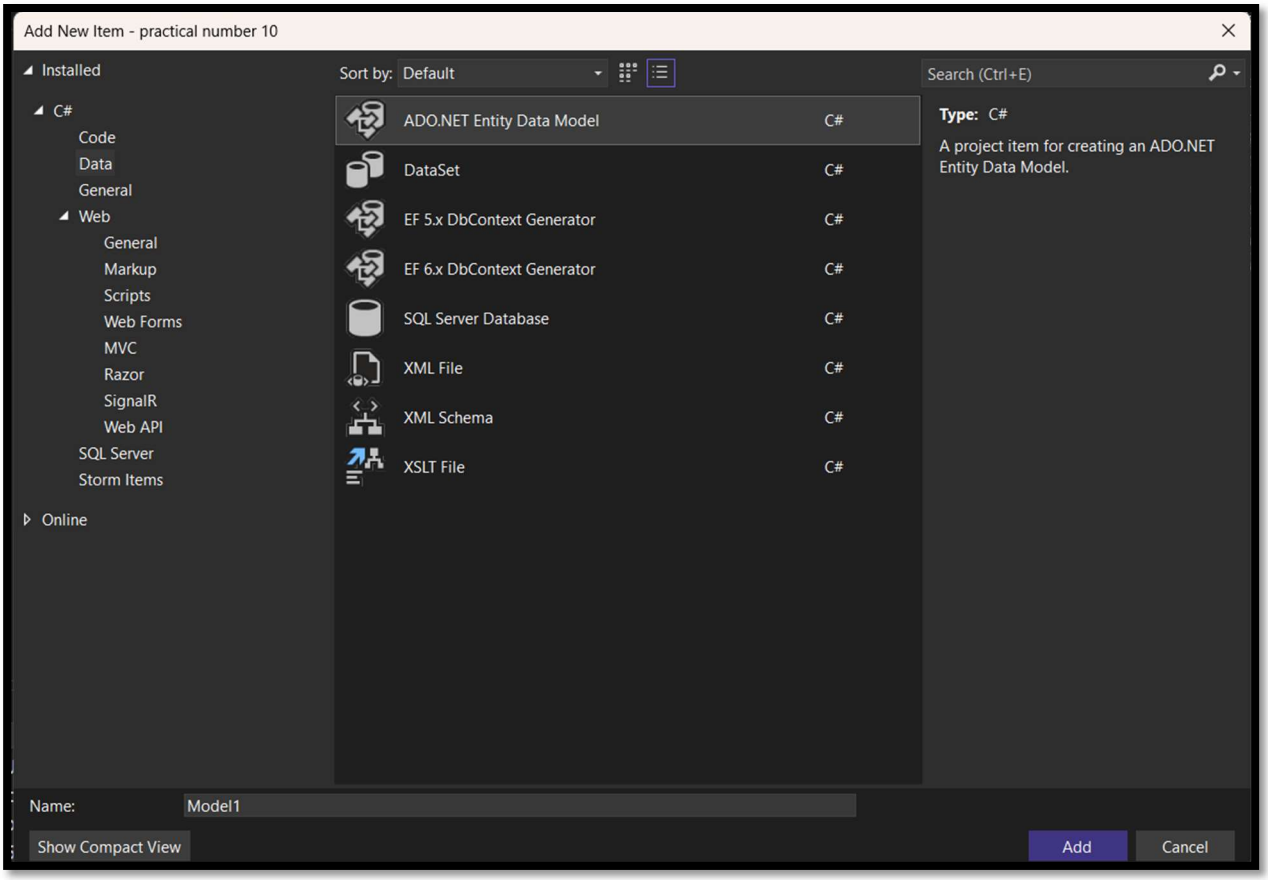


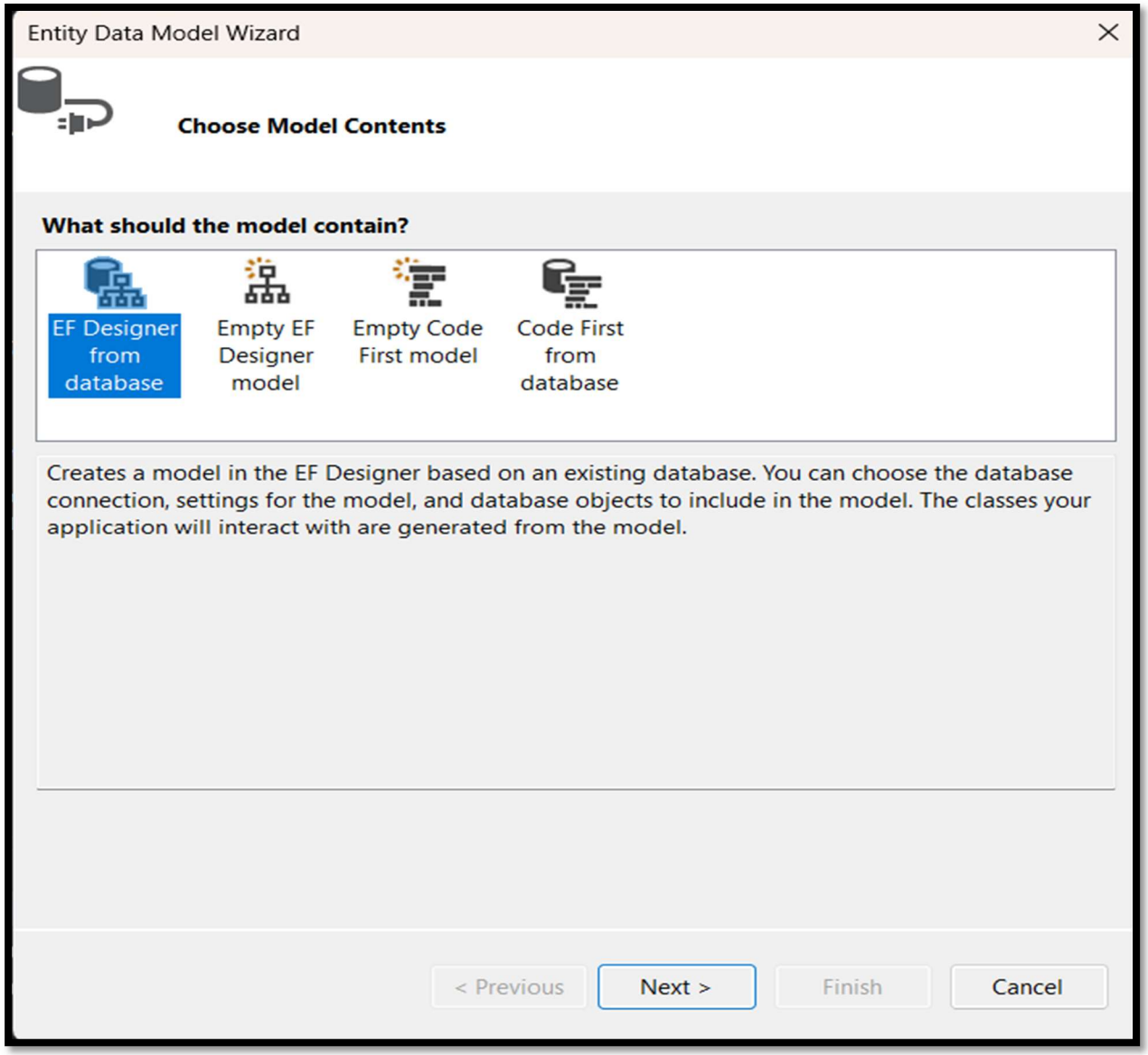


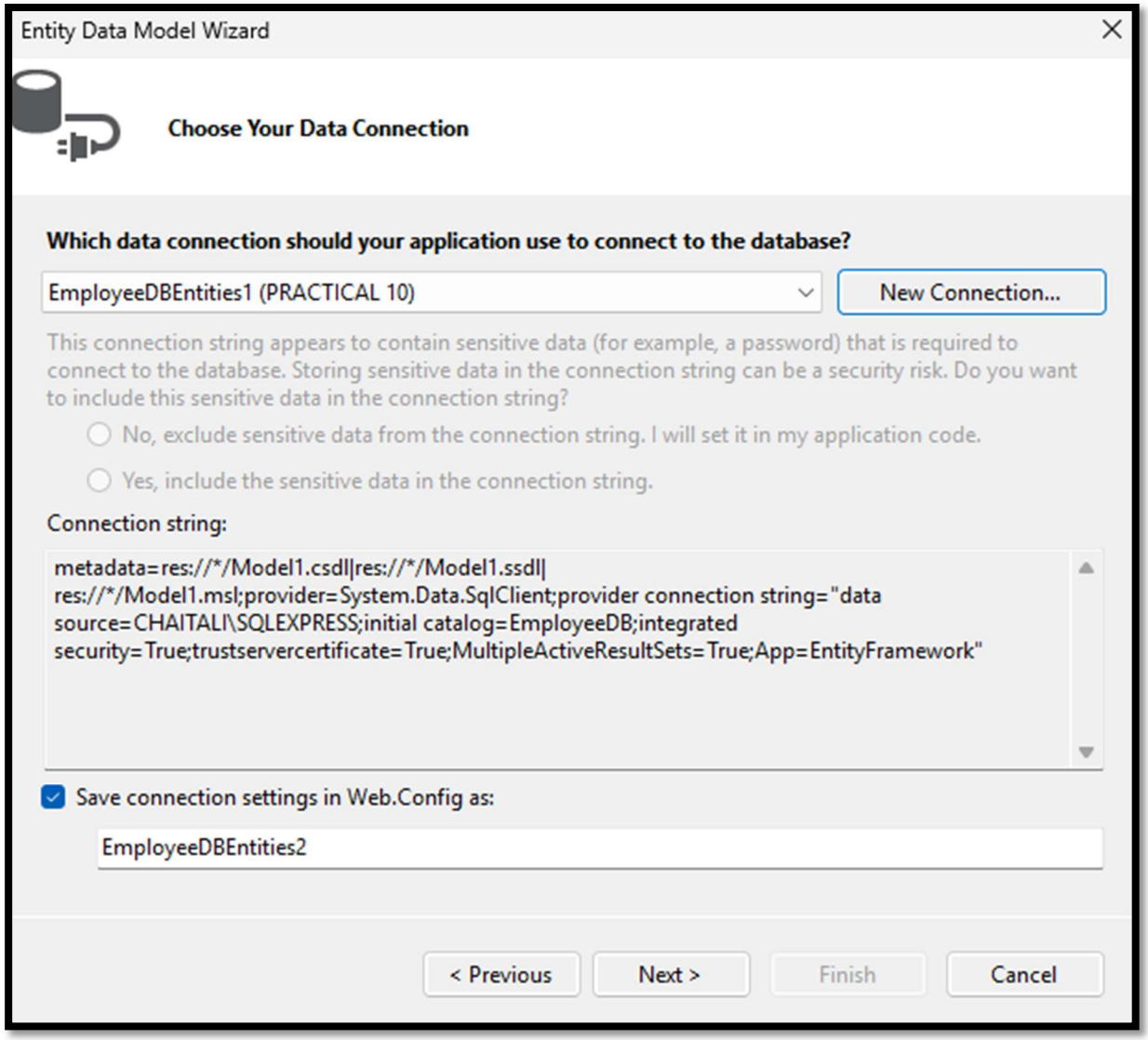
Create Entity Framework Model

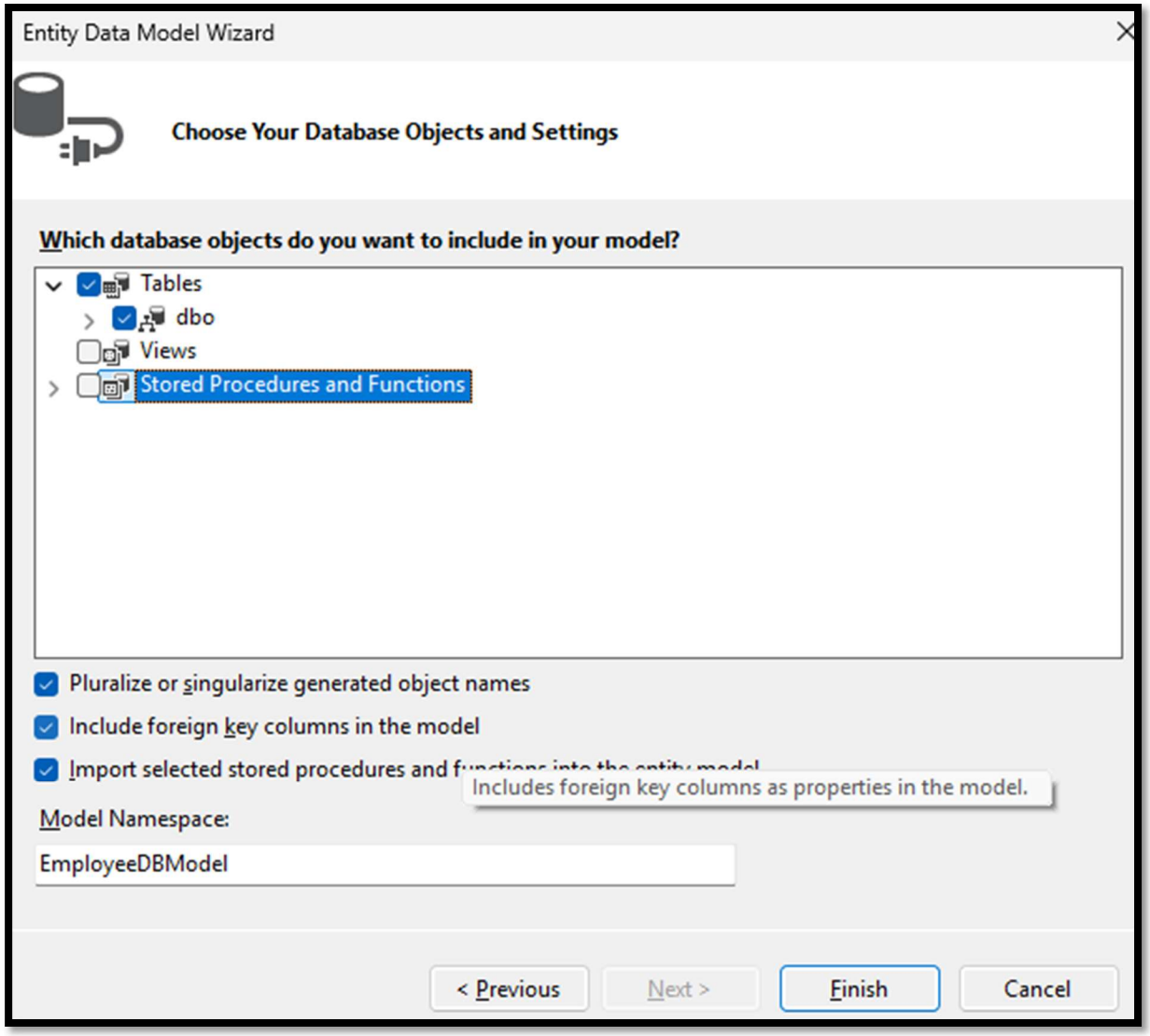
1. Right-click the project → Add → New Item → Select **ADO.NET Entity Data Model**
2. Choose **EF Designer from Database**
3. Select **"EmployeeDB"** as the database
4. Select the **Employees** table → Finish

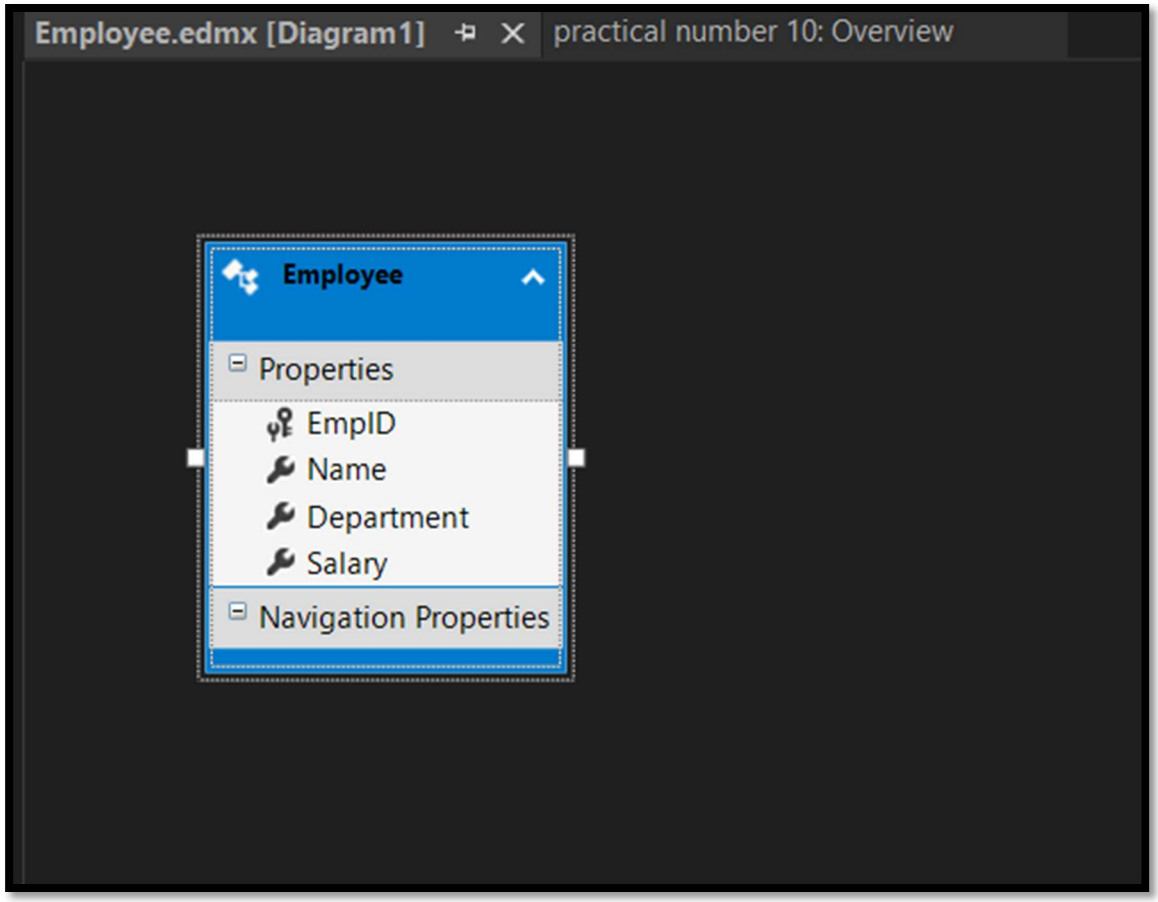
This generates the **Employee.cs model class**.


Add New Item - practical number 10
Installed
C#
Code
Data
General
Web
General
Markup
Scripts
Web Forms
MVC
Razor
SignalR
Web API
SQL Server
Storm Items
Online
Sort by: Default
Search (Ctrl+E)
ADO.NET Entity Data Model C#
DataSet C#
EF 5.x DbContext Generator C#
EF 6.x DbContext Generator C#
SQL Server Database C#
XML File C#
XML Schema C#
XSLT File C#
Type: C#
A project item for creating an ADO.NET Entity Data Model.
Name: Model1
Show Compact View
Add
Cancel


Entity Data Model Wizard
Choose Model Contents
What should the model contain?
EF Designer from database
Empty EF Designer model
Empty Code First model
Code First from database
Creates a model in the EF Designer based on an existing database. You can choose the database connection, settings for the model, and database objects to include in the model. The classes your application will interact with are generated from the model.
< Previous
Next >
Finish
Cancel

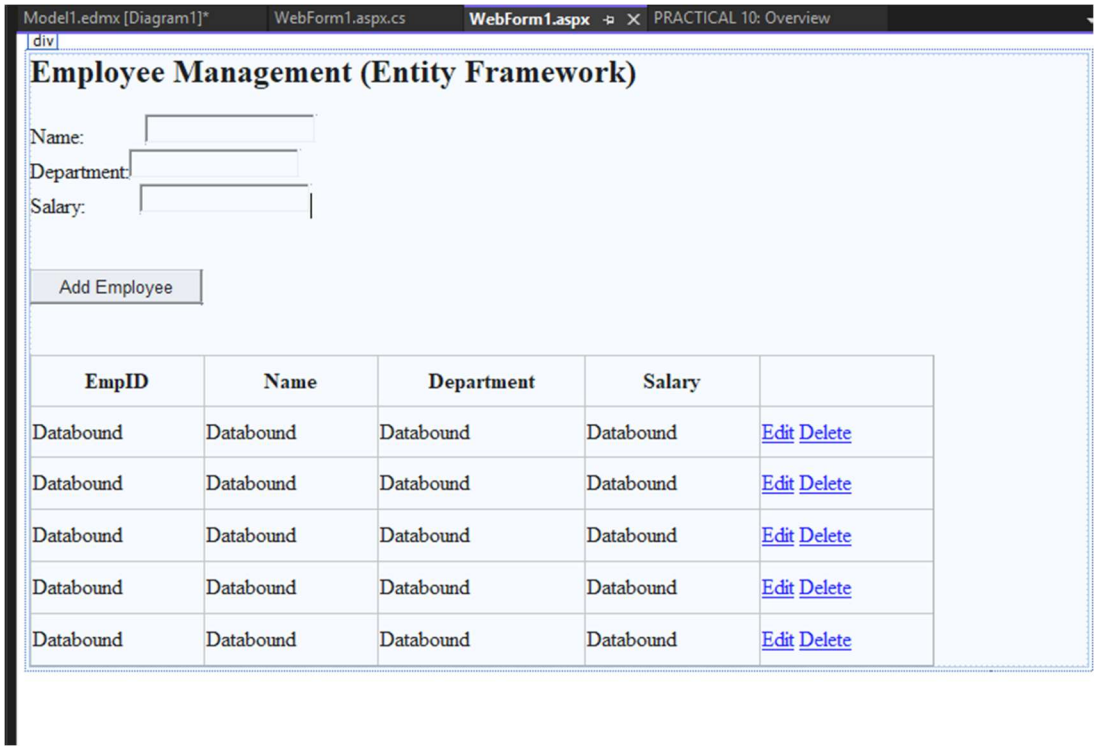

Entity Data Model Wizard
Choose Your Data Connection
Which data connection should your application use to connect to the database?
EmployeeDBEntities1 (PRACTICAL 10)
New Connection...
This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?
No, exclude sensitive data from the connection string. I will set it in my application code.
Yes, include the sensitive data in the connection string.
Connection string:
metadata=res://*/Model1.csdl|res://*/Model1.ssdl|
res://*/Model1.msl;provider=System.Data.SqlClient;provider connection string="data
source=CHAITALI\SQLEXPRESS;initial catalog=EmployeeDB;integrated
security=True;trustservercertificate=True;MultipleActiveResultSets=True;App=EntityFramework"
Save connection settings in Web.Config as:
EmployeeDBEntities2
< Previous
Next >
Finish
Cancel


Entity Data Model Wizard
Choose Your Database Objects and Settings
Which database objects do you want to include in your model?
Tables
dbo
Views
Stored Procedures and Functions
Pluralize or singularize generated object names
Include foreign key columns in the model
Import selected stored procedures and functions into the entity model
Includes foreign key columns as properties in the model.
Model Namespace:
EmployeeDBModel
< Previous
Next >
Finish
Cancel


Employee.edmx [Diagram1]
practical number 10: Overview
Employee
Properties
EmpID
Name
Department
Salary
Navigation Properties

Code:

WebForm1.aspx




```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="PRACTICAL_10.WebForm1" %>
<!DOCTYPE html>
<html lang="en">
<head runat="server">
  <title>Entity Framework CRUD Demo</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h2>Employee Management (Entity Framework)</h2>
      <!-- Add Employee Form -->
      <asp:Label runat="server" Text="Name:"></asp:Label>
      <asp:TextBox ID="txtName" runat="server" style="margin-left: 46px"></asp:TextBox>
      <br />
      <asp:Label runat="server" Text="Department:"></asp:Label>
      <asp:TextBox ID="txtDepartment" runat="server"></asp:TextBox> <br />
      <asp:Label runat="server" Text="Salary:"></asp:Label>
      <asp:TextBox ID="txtSalary" runat="server" style="margin-left: 41px"></asp:TextBox>
      <br />
```

```
        <br />
        <br />
        <asp:Button ID="btnAdd" runat="server" Text="Add Employee" OnClick="btnAdd_Click" />
        <br /><br />
        <!-- Display Employees -->
        <asp:GridView ID="gvEmployees" runat="server" AutoGenerateColumns="False"
DataKeyNames="EmpID"
          OnRowEditing="gvEmployees_RowEditing" OnRowUpdating="gvEmployees_RowUpdating"
          OnRowCancelingEdit="gvEmployees_RowCancelingEdit"
OnRowDeleting="gvEmployees_RowDeleting" Height="233px" Width="677px">
          <Columns>
            <asp:BoundField DataField="EmpID" HeaderText="EmpID" ReadOnly="True" />
            <asp:BoundField DataField="Name" HeaderText="Name" />
            <asp:BoundField DataField="Department" HeaderText="Department" />
            <asp:BoundField DataField="Salary" HeaderText="Salary" />
            <asp:CommandField ShowEditButton="True" ShowDeleteButton="True" />
          </Columns>
        </asp:GridView>
      </div>
    </form>
</body>
</html>
```

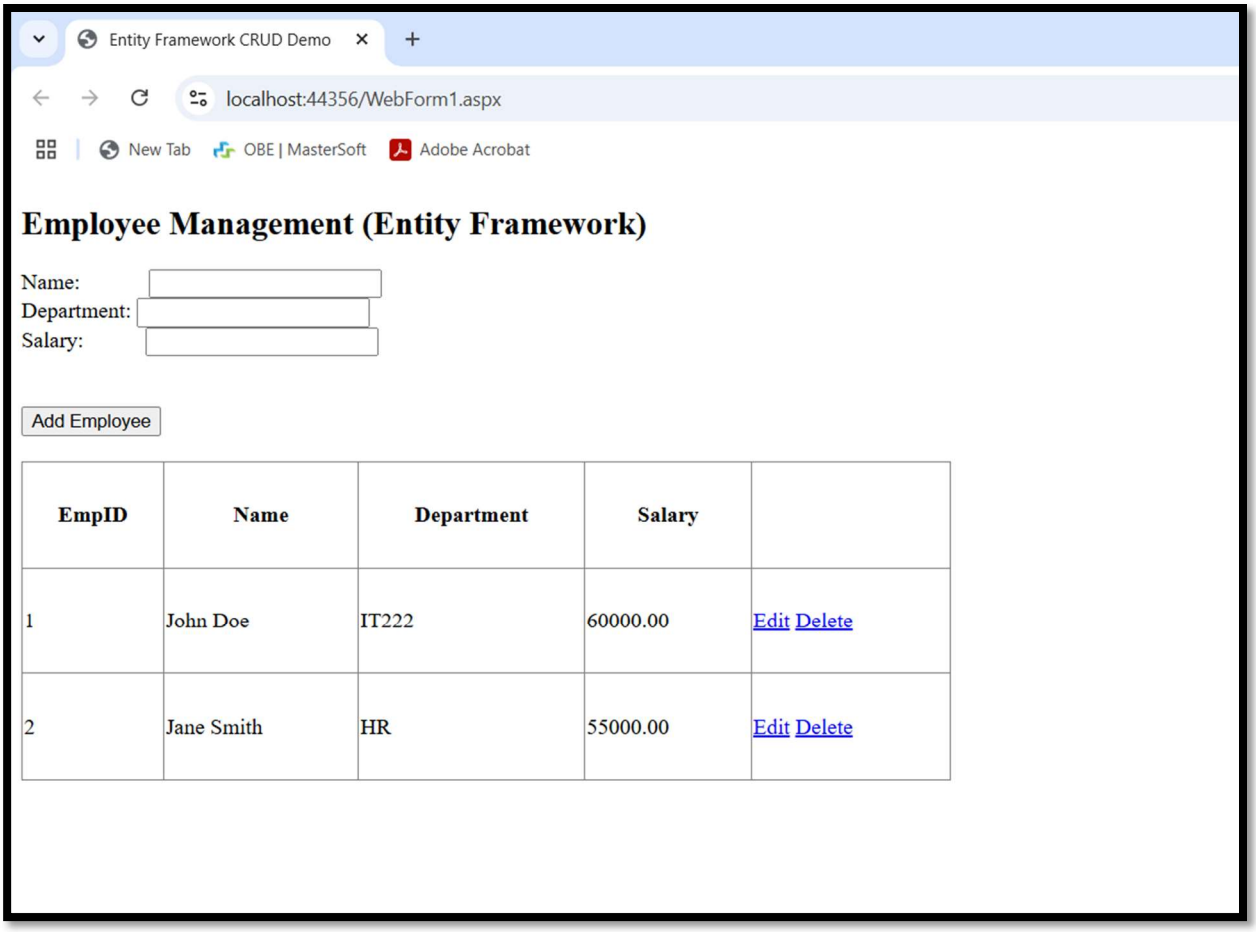
WebForm1.aspx.cs

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

namespace PRACTICAL_10
{
          public partial class WebForm1 : System.Web.UI.Page
          {
                    EmployeeDBEntities1 db = new EmployeeDBEntities1();
      protected void Page_Load(object sender, EventArgs e)
      {
        if (!IsPostBack)
        {
          LoadEmployees();
        }
```

```
}
private void LoadEmployees()
{
   gvEmployees.DataSource = db.Employees.ToList();
   gvEmployees.DataBind();
}
protected void btnAdd_Click(object sender, EventArgs e)
{
   Employee emp = new Employee
   {
      Name = txtName.Text,
      Department = txtDepartment.Text,
      Salary = Convert.ToDecimal(txtSalary.Text)
   };
   db.Employees.Add(emp);
   db.SaveChanges();
   LoadEmployees();
}
protected void gvEmployees_RowEditing(object sender, GridViewEditEventArgs e)
{
   gvEmployees.EditIndex = e.NewEditIndex;
   LoadEmployees();
}
// Update Employee
protected void gvEmployees_RowUpdating(object sender, GridViewUpdateEventArgs e)
{

   int empID = Convert.ToInt32(gvEmployees.DataKeys[e.RowIndex].Value);

   Employee emp = db.Employees.Find(empID);
   TextBox txtName = (TextBox)gvEmployees.Rows[e.RowIndex].Cells[1].Controls[0];
   TextBox txtDepartment = (TextBox)gvEmployees.Rows[e.RowIndex].Cells[2].Controls[0];
   TextBox txtSalary = (TextBox)gvEmployees.Rows[e.RowIndex].Cells[3].Controls[0];
   emp.Name = txtName.Text;
   emp.Department = txtDepartment.Text;
   emp.Salary = Convert.ToDecimal(txtSalary.Text);
   db.SaveChanges();
   gvEmployees.EditIndex = -1;
   LoadEmployees();
}
protected void gvEmployees_RowCancelingEdit(object sender, GridViewCancelEditEventArgs e)
{
   gvEmployees.EditIndex = -1;
```

```
        LoadEmployees();
    }
    // Delete Employee
    protected void gvEmployees_RowDeleting(object sender, GridViewDeleteEventArgs e)
    {
        int empID = Convert.ToInt32(gvEmployees.DataKeys[e.RowIndex].Value);
        Employee emp = db.Employees.Find(empID);
        db.Employees.Remove(emp);
        db.SaveChanges();
        LoadEmployees();
    }
        }
}
```

Output:



Practical no.: 11
Design Web Applications using Client-Side Session Management

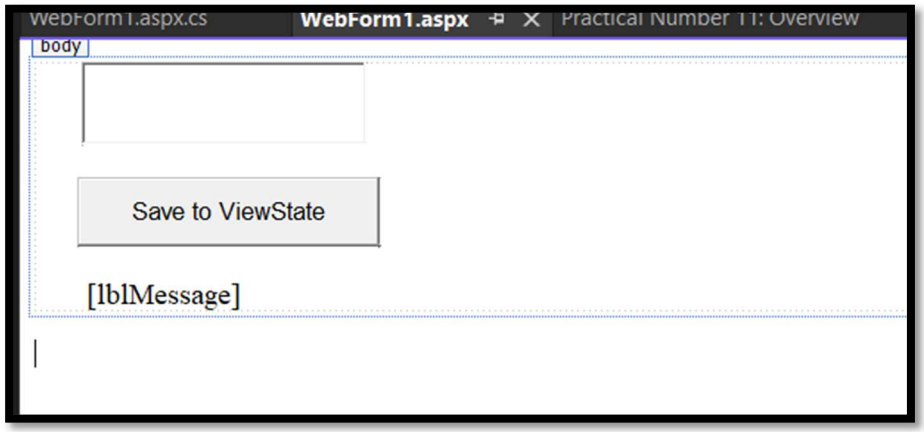
Steps to Implement Client-Side State Management

- **View State**: Maintains data across post backs.
- **Query String**: Passes data in the URL.
- **Cookies**: Stores small data persistently.

Hidden Fields: Stores temporary data in forms.

1. View State (ASP.NET Web Forms Only)

Stores data in a hidden field but is accessible only on the same page.



Code:

WebForm1.aspx

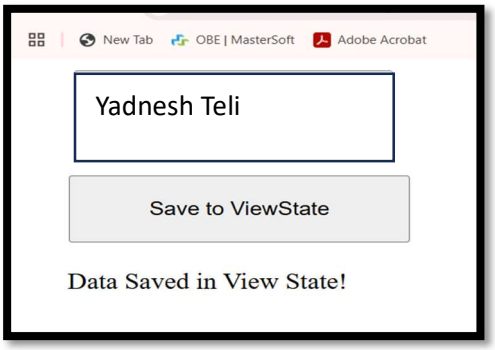
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="Practical_Number_11.WebForm1" %>

<!DOCTYPE html>
<html>
<head>
  <title>View State Example</title>
</head>
<body>
  <form runat="server">
    <asp:TextBox ID="txtName" runat="server" Height="50px" style="margin-left: 32px"
Width="185px"></asp:TextBox>
```

```
    <br />
    <br />
    <asp:Button ID="btnSubmit" runat="server" Text="Save to ViewState" OnClick="btnSubmit_Click"
Height="47px" style="margin-left: 29px" Width="205px" />
    <br />
    <br />
      
    <asp:Label ID="lblMessage" runat="server"></asp:Label>
  </form>
</body>
</html>
```

WebForm1.aspx.cs

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

namespace Practical_Number_11
{
  public partial class WebForm1 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {
      if (ViewState["UserName"] != null)
      {
        lblMessage.Text = "Stored in View State: " + ViewState["UserName"].ToString();
      }

    }
    protected void btnSubmit_Click(object sender, EventArgs e)
    {
      ViewState["UserName"] = txtName.Text;
      lblMessage.Text = "Data Saved in View State!";
    }
  }
}
```

Output:



WebForm2.aspx

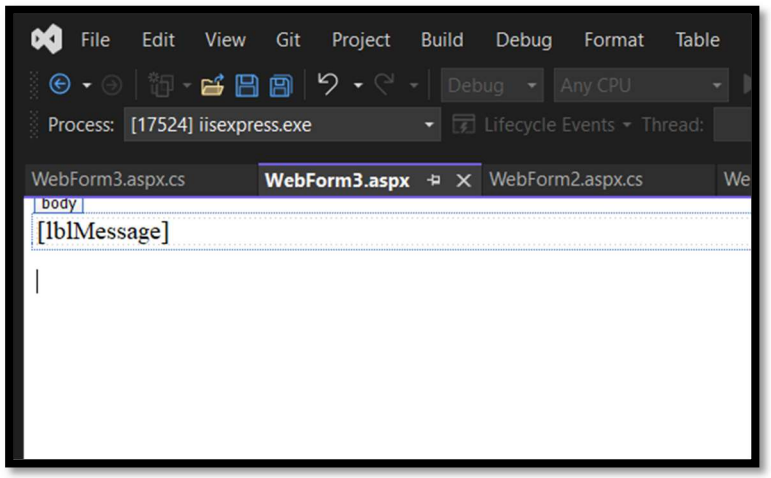
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm2.aspx.cs"
Inherits="Practical_Number_11.WebForm2" %>

<!DOCTYPE html>
<html>
<head>
  <title>Query String Example</title>
</head>
<body>
  <form runat="server">
    <asp:TextBox ID="txtName" runat="server" Height="67px" Width="193px"></asp:TextBox>
    <br />
    <br />
    <br />
    <asp:Button ID="btnSubmit" runat="server" Text="Go to Next Page" OnClick="btnSubmit_Click"
Height="62px" Width="203px" />
  </form>
</body>
</html>
```

WebForm2.aspx.cs

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

namespace Practical_Number_11
{
  public partial class WebForm2 : System.Web.UI.Page
```

```
  {
    protected void Page_Load(object sender, EventArgs e)
    {

    }
    protected void btnSubmit_Click(object sender, EventArgs e)
    {
      Response.Redirect("WebForm3.aspx?name=" + txtName.Text);
    }
  }
}
```

Output:



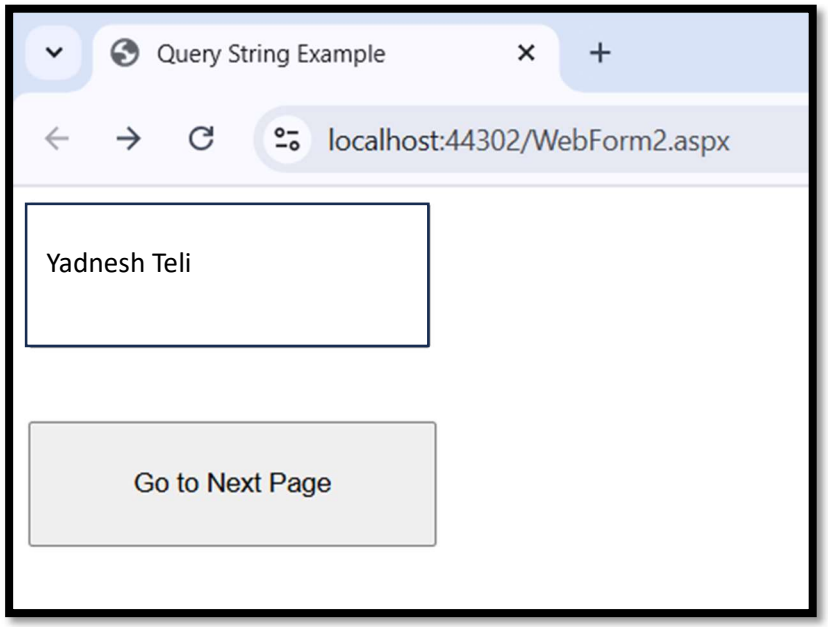
WebForm3.aspx

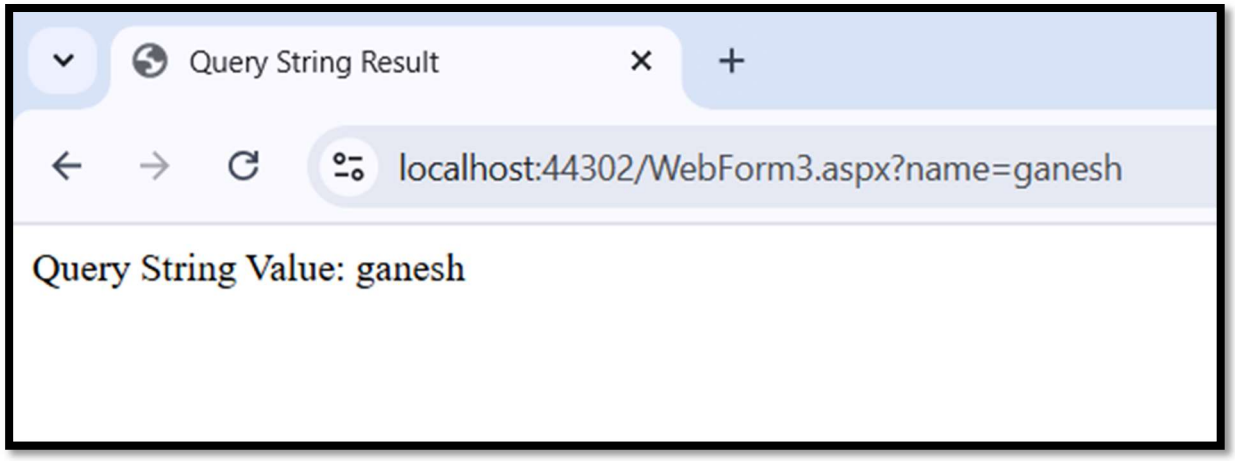
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm3.aspx.cs"
Inherits="Practical_Number_11.WebForm3" %>

<!DOCTYPE html>
<html>
<head>
  <title>Query String Result</title>
</head>
<body>
  <form runat="server">
    <asp:Label ID="lblMessage" runat="server"></asp:Label>
  </form>
</body>
</html>
```

WebForm3.aspx.cs

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

namespace Practical_Number_11
{
   public partial class WebForm3 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (Request.QueryString["name"] != null)
         {
            lblMessage.Text = "Query String Value: " + Request.QueryString["name"];
         }

      }
   }
}
```

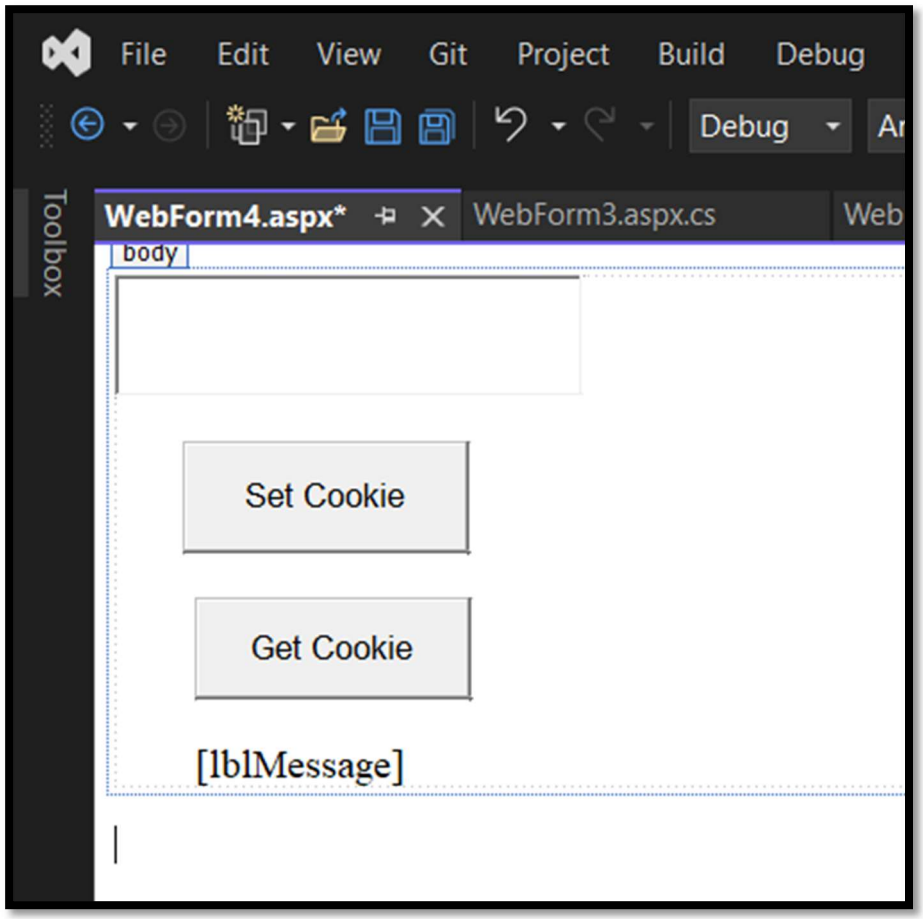
Output:





3 Cookies (Persistent Client-Side Storage)

Stores small data on the client-side, accessible across pages.



WebForm4.aspx

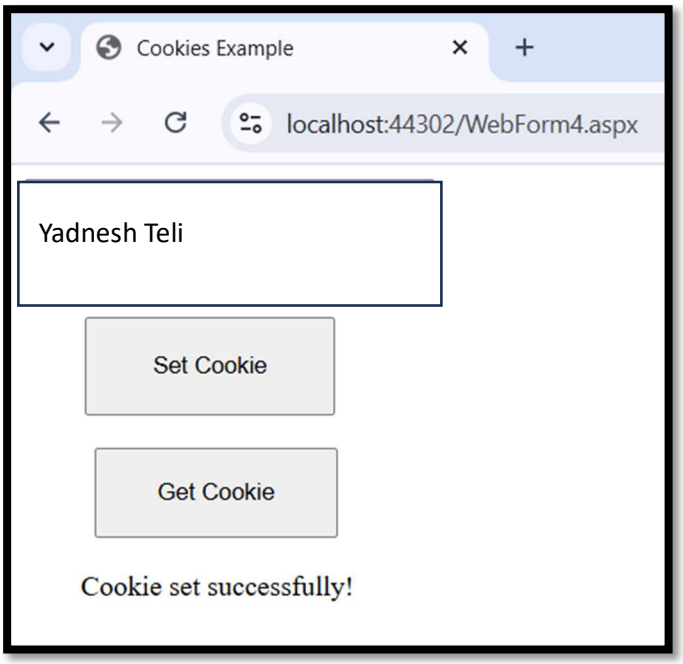
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm4.aspx.cs"
Inherits="Practical_Number_11.WebForm4" %>

<!DOCTYPE html>
<html>
<head>
   <title>Cookies Example</title>
</head>
<body>
   <form runat="server">
      <asp:TextBox ID="txtName" runat="server" Height="55px" Width="226px"></asp:TextBox>
      <br />
      <br />
      <asp:Button ID="btnSetCookie" runat="server" Text="Set Cookie" OnClick="btnSetCookie_Click"
Height="56px" style="margin-left: 34px" Width="143px" />
      <br />
      <br />
      <asp:Button ID="btnGetCookie" runat="server" Text="Get Cookie"
OnClick="btnGetCookie_Click" Height="51px" style="margin-left: 40px" Width="138px" />
      <br />
      <br />
       
      <asp:Label ID="lblMessage" runat="server"></asp:Label>
   </form>
</body>
</html>
```

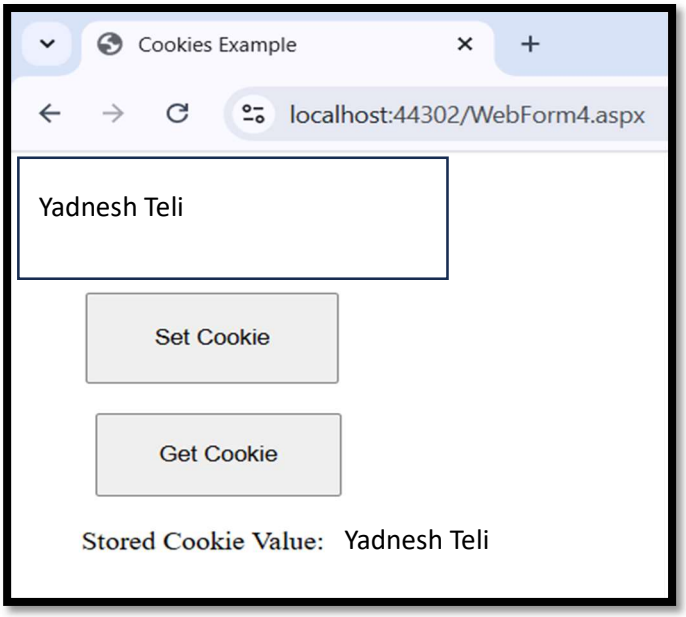
WebForm4.aspx.cs

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

namespace Practical_Number_11
{
   public partial class WebForm4 : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
```

```
    }

    protected void btnSetCookie_Click(object sender, EventArgs e)
    {
        HttpCookie cookie = new HttpCookie("UserName", txtName.Text);
        cookie.Expires = DateTime.Now.AddDays(7); // Cookie valid for 7 days
        Response.Cookies.Add(cookie);
        lblMessage.Text = "Cookie set successfully!";
    }
    protected void btnGetCookie_Click(object sender, EventArgs e)
    {
        HttpCookie cookie = Request.Cookies["UserName"];
        if (cookie != null)
        {
            lblMessage.Text = "Stored Cookie Value: " + cookie.Value;
        }
        else
        {
            lblMessage.Text = "No Cookie Found!";
        }

    }
  }
}
```

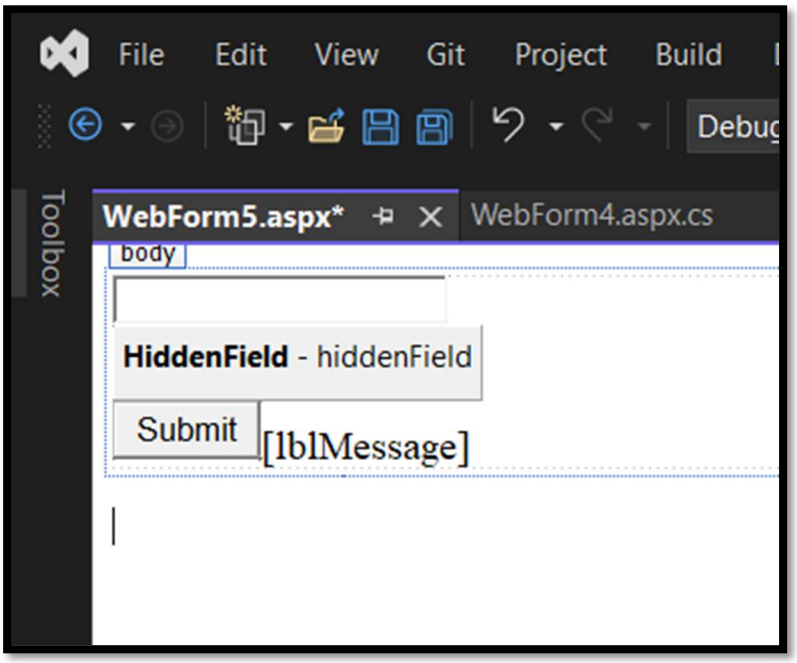
Output:





4 Hidden Fields (Storing Temporary Data)

Stores data in an HTML form but remains invisible to the user.



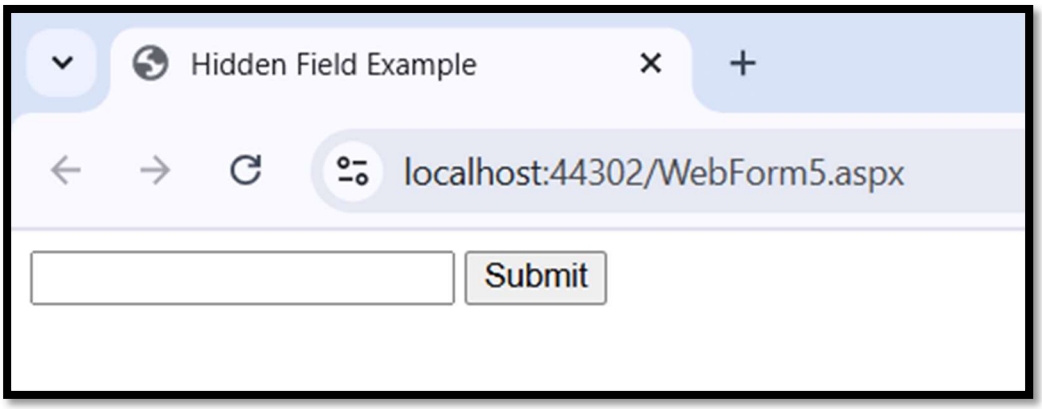
WebForm5.aspx

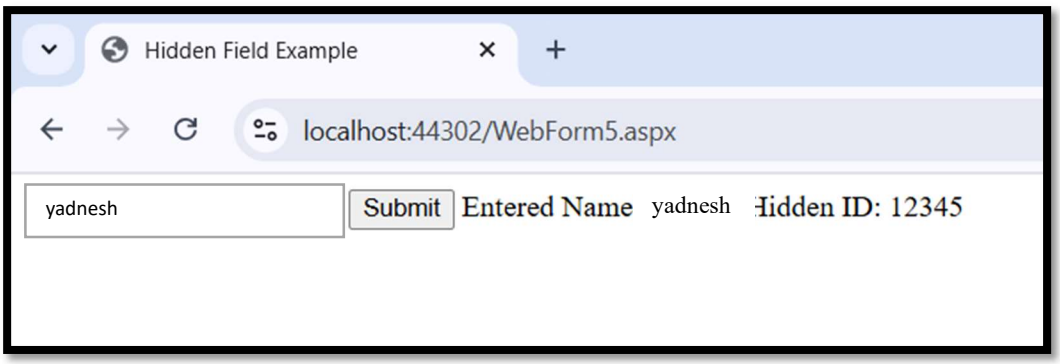
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm5.aspx.cs"
Inherits="Practical_Number_11.WebForm5" %>

<html>
<head>
  <title>Hidden Field Example</title>
</head>
<body>
  <form runat="server">
    <asp:TextBox ID="txtName" runat="server"></asp:TextBox>
    <asp:HiddenField ID="hiddenField" runat="server" Value="12345" />
    <asp:Button ID="btnSubmit" runat="server" Text="Submit" OnClick="btnSubmit_Click" />
    <asp:Label ID="lblMessage" runat="server"></asp:Label>
  </form>
</body>
</html>
```

WebForm5.aspx.cs

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

namespace Practical_Number_11
{
  public partial class WebForm5 : System.Web.UI.Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void btnSubmit_Click(object sender, EventArgs e)
    {
      lblMessage.Text = "Entered Name: " + txtName.Text + ", Hidden ID: " + hiddenField.Value;
    }
  }
}
```

Output:


Hidden Field Example
localhost:44302/WebForm5.aspx
Submit


Hidden Field Example
localhost:44302/WebForm5.aspx
yadnesh
Submit
Entered Name yadnesh
Hidden ID: 12345

Practical no.: 12
Design Web Applications using Server-Side Session Management Techniques

Code:

Web.config

```
<?xml version="1.0" encoding="utf-8"?>
<!--
 For more information on how to configure your ASP.NET application, please visit
 https://go.microsoft.com/fwlink/?LinkId=169433
 -->
<configuration>
 <system.web>
  <compilation debug="true" targetFramework="4.7.2" />
  <httpRuntime targetFramework="4.7.2" />
 <sessionState mode="InProc" timeout="20" cookieless="UseCookies" />
 </system.web>
 <system.codedom>
  <compilers>
   <compiler language="c#;cs;csharp" extension=".cs"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.CSharpCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default
/nowarn:1659;1699;1701" />
    <compiler language="vb;vbs;visualbasic;vbscript" extension=".vb"
type="Microsoft.CodeDom.Providers.DotNetCompilerPlatform.VBCodeProvider,
Microsoft.CodeDom.Providers.DotNetCompilerPlatform, Version=2.0.1.0, Culture=neutral,
PublicKeyToken=31bf3856ad364e35" warningLevel="4" compilerOptions="/langversion:default
/nowarn:41008 /define:_MYTYPE=\&quot;Web\&quot; /optionInfer+" />
  </compilers>
 </system.codedom>
</configuration>
```

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>
```

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!";
        }
    }
  }
}
```

Dashboard.aspx

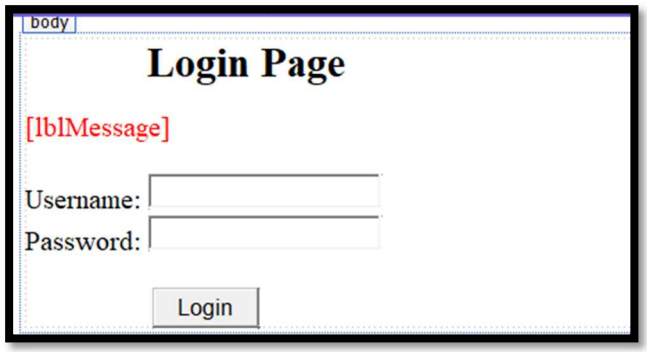
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Dashboard.aspx.cs"
Inherits="SessionManagementDemo.Dashboard" %>
<!DOCTYPE html>
<html lang="en">
<head runat="server">
  <title>Dashboard</title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      <h2>Welcome, <asp:Label ID="lblUser" runat="server"></asp:Label>!</h2>
      <asp:Button ID="btnLogout" runat="server" Text="Logout" OnClick="btnLogout_Click" />
    </div>
  </form>
</body>
</html>
```

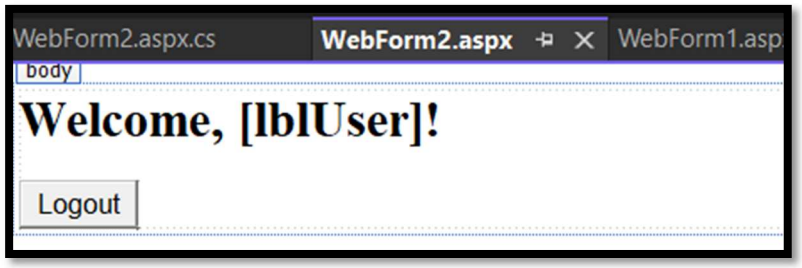
Dashboard.aspx.cs

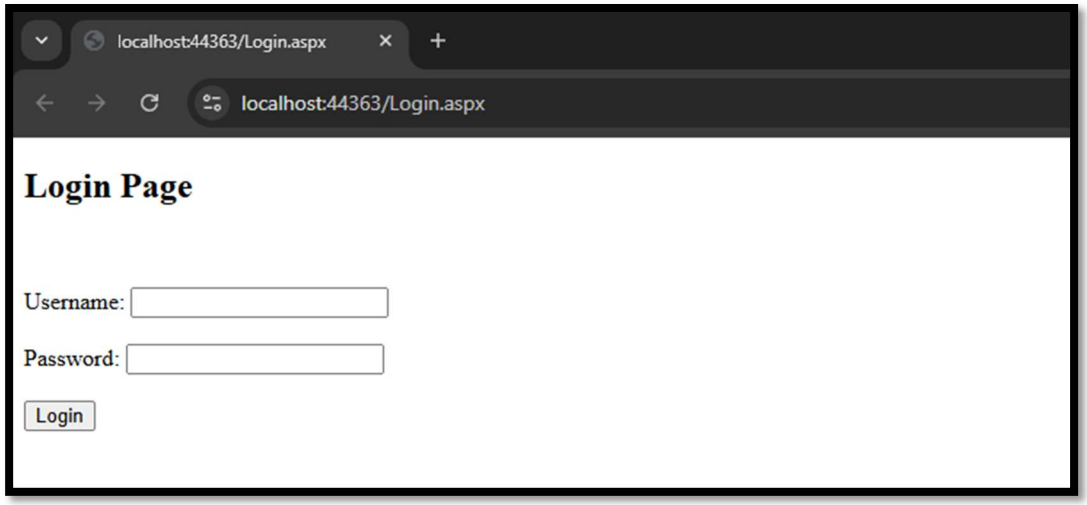
```
using System;
using System.Web;

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");
```

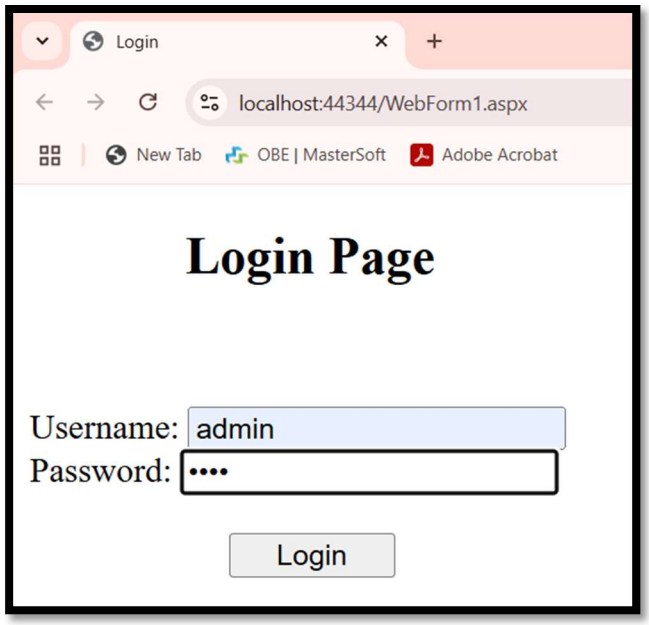
```
        }
        else
        {
            lblUser.Text = Session["Username"].ToString();
        }
    }

    protected void btnLogout_Click(object sender, EventArgs e)
    {
        Session.Abandon();
        Response.Redirect("Login.aspx");
    }
  }
}
```

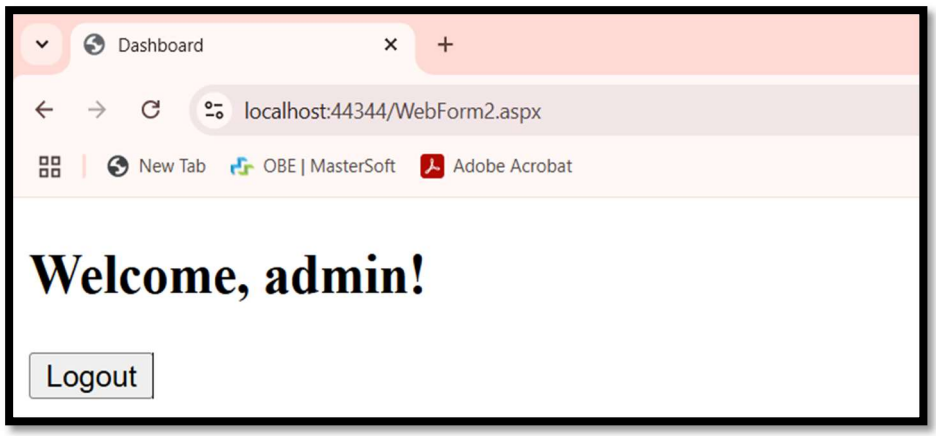
Output:

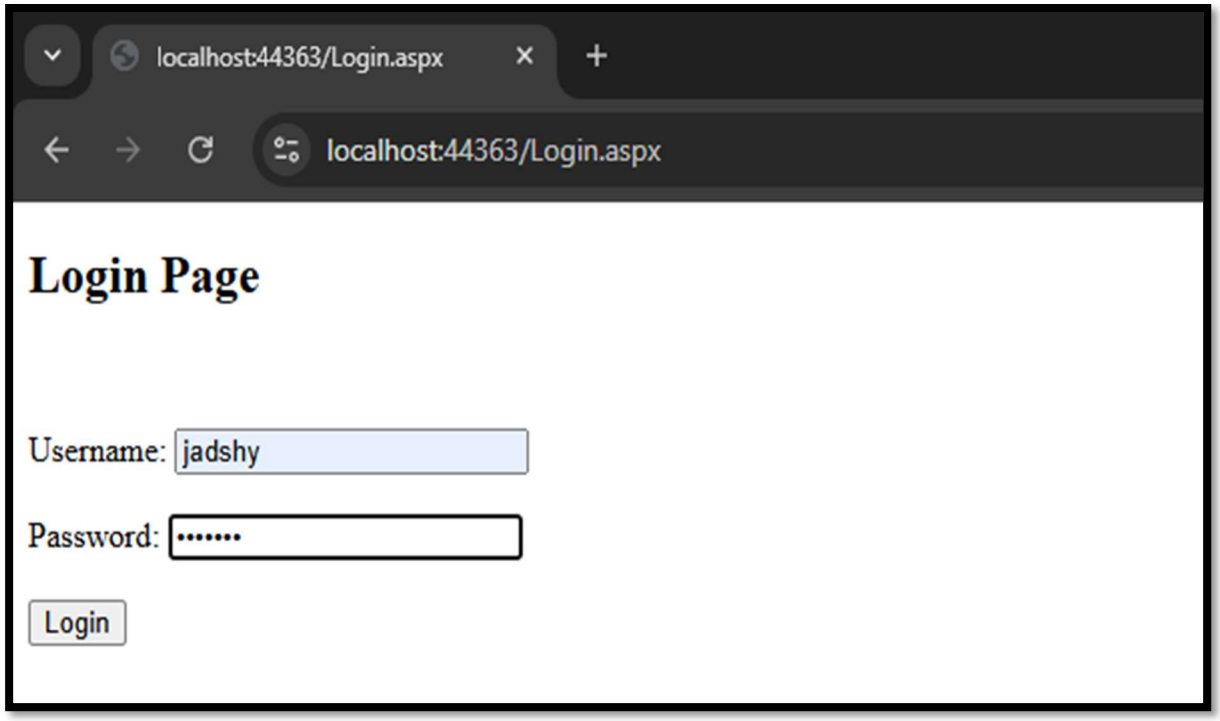


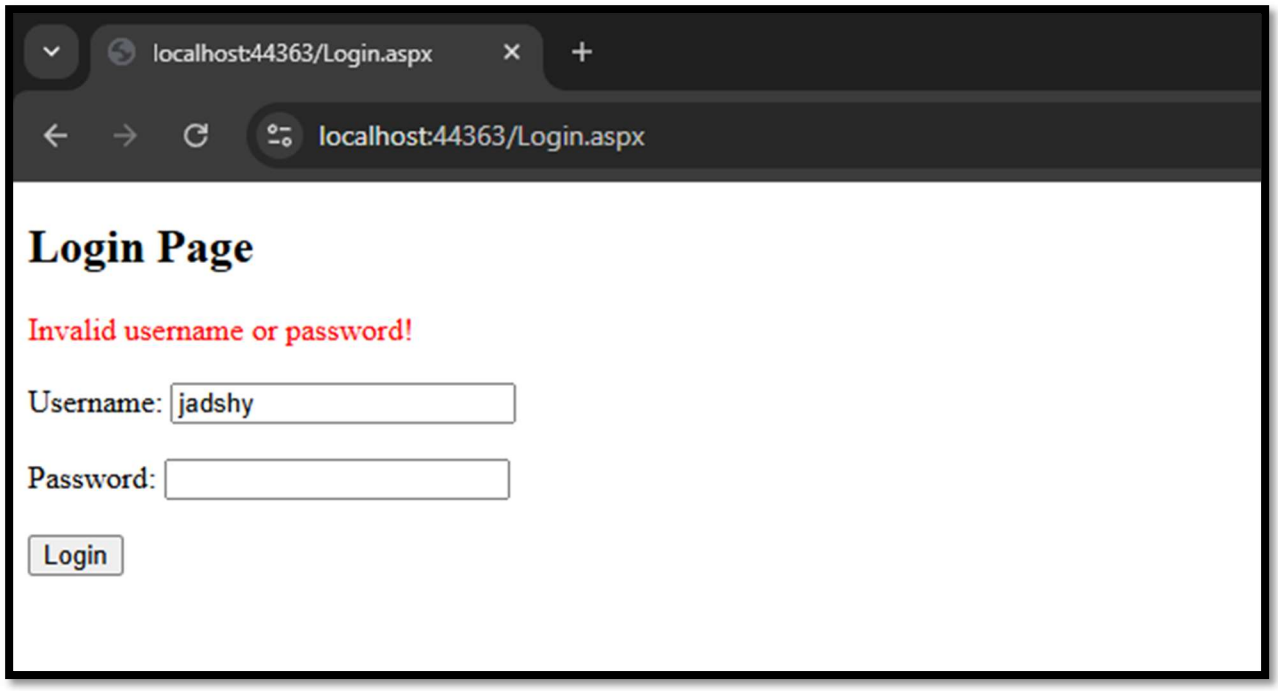



localhost:44363/Login.aspx
localhost:44363/Login.aspx
Login Page
Username:
Password:
Login


Login
localhost:44344/WebForm1.aspx
New Tab
OBE | MasterSoft
Adobe Acrobat
Login Page
Username: admin
Password: ••••
Login


Dashboard
localhost:44344/WebForm2.aspx
New Tab
OBE | MasterSoft
Adobe Acrobat
Welcome, admin!
Logout


localhost:44363/Login.aspx
localhost:44363/Login.aspx
Login Page
Username: jadshy
Password:
Login


localhost:44363/Login.aspx
localhost:44363/Login.aspx
Login Page
Invalid username or password!
Username: jadshy
Password:
Login

Practical no.: 13
Build a web page using AJAX Controls.

Code:

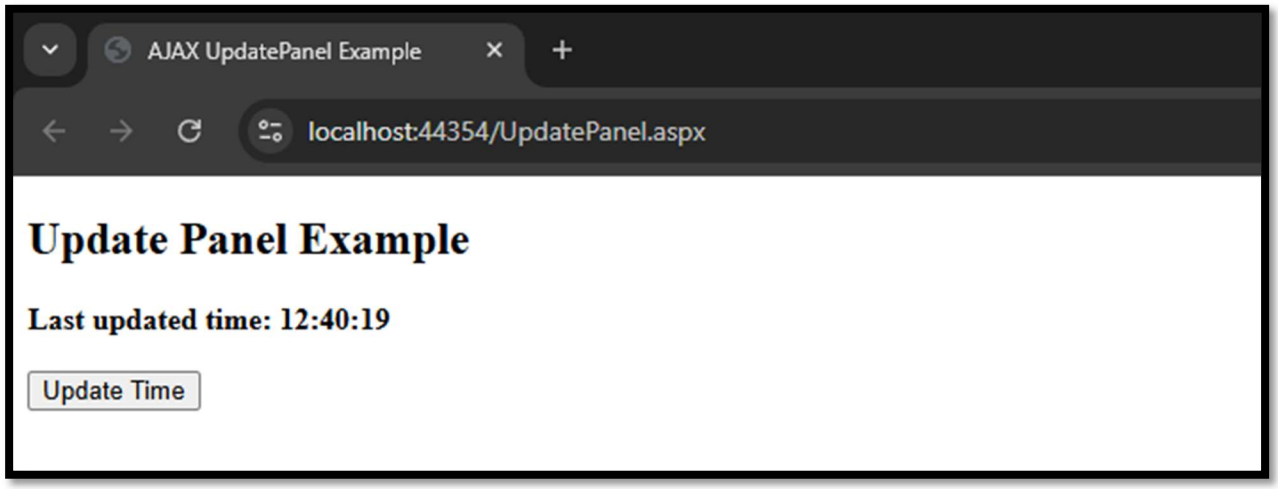
UpdatePanel.aspx

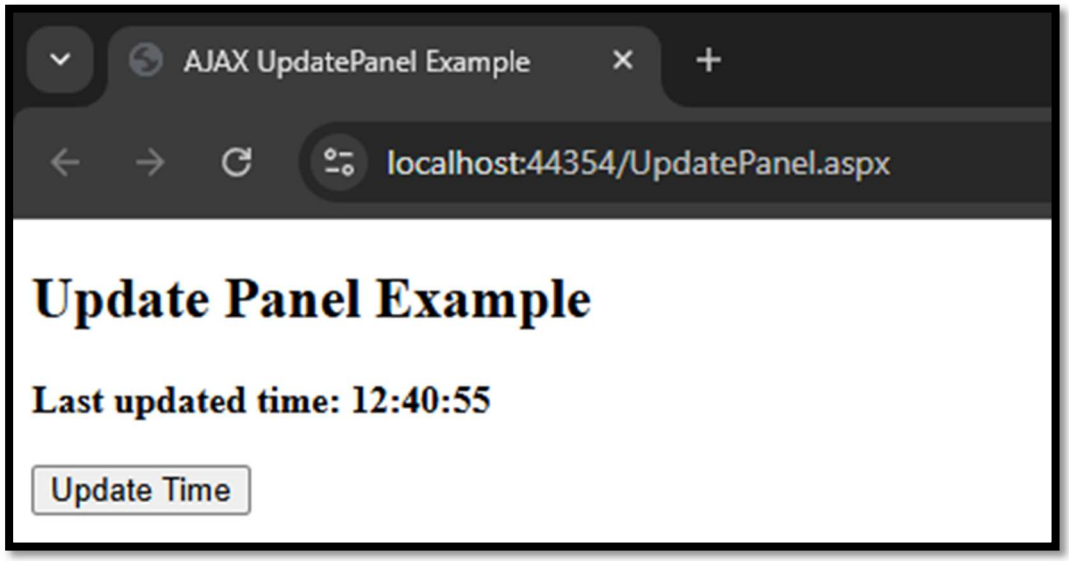
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="UpdatePanel.aspx.cs"
Inherits="Practical13.UpdatePanel" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>AJAX UpdatePanel Example</title>
</head>
<body>
   <form id="form1" runat="server">
      <asp:ScriptManager ID="ScriptManager1" runat="server"></asp:ScriptManager>
      <h2>Update Panel Example</h2>
      <asp:UpdatePanel ID="UpdatePanel1" runat="server">
         <ContentTemplate>
            <asp:Label ID="lblTime" runat="server" Font-Bold="True"></asp:Label>
            <br /><br />
            <asp:Button ID="btnUpdate" runat="server" Text="Update Time" OnClick="btnUpdate_Click" />
         </ContentTemplate>
      </asp:UpdatePanel>
   </form>
</body>
</html>
```

UpdatePanel.aspx.cs

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

namespace Practical13
{
   public partial class UpdatePanel : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            lblTime.Text = "Last updated time: " + DateTime.Now.ToString("HH:mm:ss");
         }
      }
      protected void btnUpdate_Click(object sender, EventArgs e)
      {
         lblTime.Text = "Last updated time: " + DateTime.Now.ToString("HH:mm:ss");
      }
   }
}
```

Output:


AJAX UpdatePanel Example
localhost:44354/UpdatePanel.aspx
Update Panel Example
Last updated time: 12:40:19
Update Time


AJAX UpdatePanel Example
localhost:44354/UpdatePanel.aspx
Update Panel Example
Last updated time: 12:40:55
Update Time

Code:

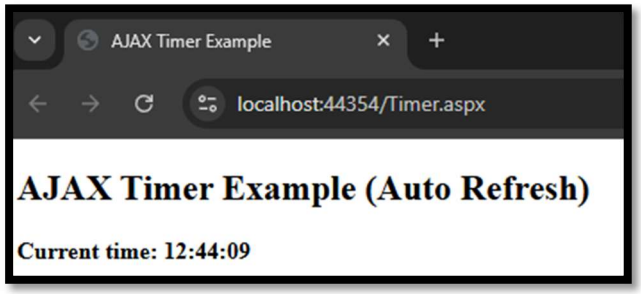
Timer.aspx

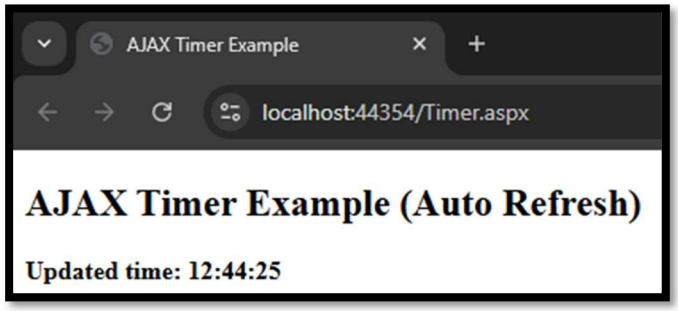
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="Timer.aspx.cs"
Inherits="Practical13.Timer" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title>AJAX Timer Example</title>
</head>
<body>
  <form id="form1" runat="server">
    <asp:ScriptManager ID="ScriptManager1" runat="server"></asp:ScriptManager>
    <h2>AJAX Timer Example (Auto Refresh)</h2>
    <asp:UpdatePanel ID="UpdatePanel1" runat="server">
      <ContentTemplate>
        <asp:Label ID="lblTime" runat="server" Font-Bold="True"></asp:Label>
      </ContentTemplate>
    </asp:UpdatePanel>
    <asp:Timer ID="Timer1" runat="server" Interval="5000" OnTick="Timer1_Tick" />
  </form>
</body>
</html>
```

Timer.aspx.cs

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


namespace Practical13
{
   public partial class Timer : System.Web.UI.Page
   {
      protected void Page_Load(object sender, EventArgs e)
      {
         if (!IsPostBack)
         {
            lblTime.Text = "Current time: " + DateTime.Now.ToString("HH:mm:ss");
         }
      }
      protected void Timer1_Tick(object sender, EventArgs e)
      {
         lblTime.Text = "Updated time: " + DateTime.Now.ToString("HH:mm:ss");
      }
   }
}
```

Output:





Practical no.: 14
Build a web application to create and use web service in ASP.net

Code:

Simple_Interest.asmx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;

namespace SimpleInterest
{
   /// <summary>
   /// Summary description for Simple_Interest
   /// </summary>
   [WebService(Namespace = "http://tempuri.org/")]
   [WebServiceBinding(ConformsTo = WsiProfiles.BasicProfile1_1)]
   [System.ComponentModel.ToolboxItem(false)]
   // To allow this Web Service to be called from script, using ASP.NET AJAX, uncomment the following
line.
   // [System.Web.Script.Services.ScriptService]
   public class Simple_Interest : System.Web.Services.WebService
   {

      [WebMethod]
      public double SI(double p, double n, double r)
      {
         return (p * n * r)/100;
      }
   }
}
```

WebForm1.aspx

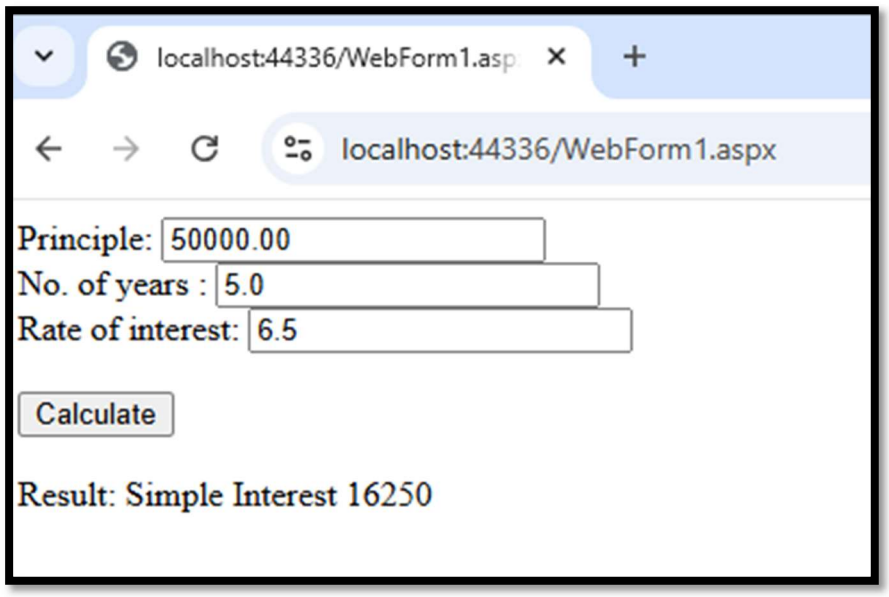
```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="WebForm1.aspx.cs"
Inherits="SimpleInterest.WebForm1" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    Principle:
    <asp:TextBox ID="txtp" runat="server" />
    <br />
    No. of years :
    <asp:TextBox ID="txtn" runat="server" />
    <br />
    Rate of interest:
    <asp:TextBox ID="txtr" runat="server" />
    <br />
    <br />
    <asp:Button ID="btnCal" runat="server" OnClick="btnCal_Click" Text="Calculate" />
    <br />
    <br />
    Result:
    <asp:Label ID="lblResult" runat="server" Text="" />
    <div>
    </div>
  </form>
</body>
</html>
```

WebForm1.aspx.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Services;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace SimpleInterest
{
  public partial class WebForm1 : System.Web.UI.Page
```

```
  {
    Simple_Interest Service1 = new Simple_Interest();
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void btnCal_Click(object sender, EventArgs e)
    {
      double p=Convert.ToDouble(txtp.Text);
      double n = Convert.ToDouble(txtn.Text);
      double r = Convert.ToDouble(txtr.Text);
      double result= Service1.SI(p,n,r);
      lblResult.Text = "Simple Interest " + result.ToString();

    }
  }
}
```

Output:



Practical no.: 15
Build a web application to create and WCF service in ASP.net

Calculator using WCF

Code:

IService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;


// NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name "IService" in both code and config file together.
[ServiceContract]
public interface IService
{
[OperationContract]
string GetData(int value);


[OperationContract]
double add(double a, double b);


    [OperationContract]
    double sub(double a, double b);


    [OperationContract]
    double mul(double a, double b);


    [OperationContract]
    double div(double a, double b);


    [OperationContract]
CompositeType GetDataUsingDataContract(CompositeType composite);
```

```
// TODO: Add your service operations here
}


// Use a data contract as illustrated in the sample below to add composite types to service operations.
[DataContract]
public class CompositeType
{
bool boolValue = true;
string stringValue = "Hello ";


[DataMember]
public bool BoolValue
{
get { return boolValue; }
set { boolValue = value; }
}


[DataMember]
public string StringValue
{
get { return stringValue; }
set { stringValue = value; }
}
}
```

Service.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the class name
"Service" in code, svc and config file together.
public class Service : IService
{
public string GetData(int value)
{
return string.Format("You entered: {0}", value);
```

```
}

public double add(double a, double b)
{
return a + b;
}
   public double sub(double a, double b)
   {
      return a - b;
   }
   public double mul(double a, double b)
   {
      return a * b;
   }
   public double div(double a, double b)
   {
      return a / b;
   }
   public CompositeType GetDataUsingDataContract(CompositeType composite)
{
if (composite == null)
{
throw new ArgumentNullException("composite");
}
if (composite.BoolValue)
{
composite.StringValue += "Suffix";
}
return composite;
}
}
```

Default.aspx

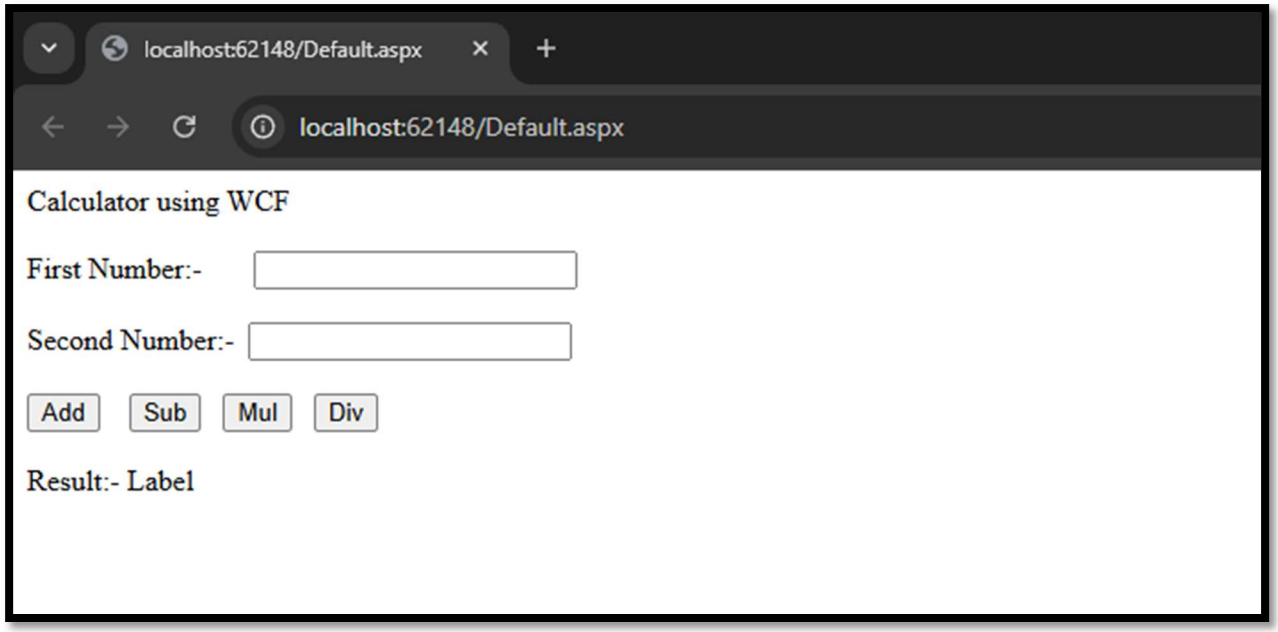
```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    <div>
      Calculator using WCF<br />
      <br />
      First Number:-      
      <asp:TextBox ID="txt1" runat="server"></asp:TextBox>
      <br />
      <br />
      Second Number:-  <asp:TextBox ID="txt2" runat="server"></asp:TextBox>
      <br />
      <br />
      <asp:Button ID="btnAdd" runat="server" OnClick="btnAdd_Click" Text="Add" />
  
      <asp:Button ID="btnSub" runat="server" OnClick="btnSub_Click" Text="Sub" />
 
      <asp:Button ID="btnMul" runat="server" OnClick="btnMul_Click" Text="Mul" />
 
      <asp:Button ID="btnDiv" runat="server" OnClick="btnDiv_Click" Text="Div" />
      <br />
      <br />
      Result:-
      <asp:Label ID="lblres" runat="server" Text="Label"></asp:Label>
      <br />
      <br />
    </div>
  </form>
</body>
</html>
```

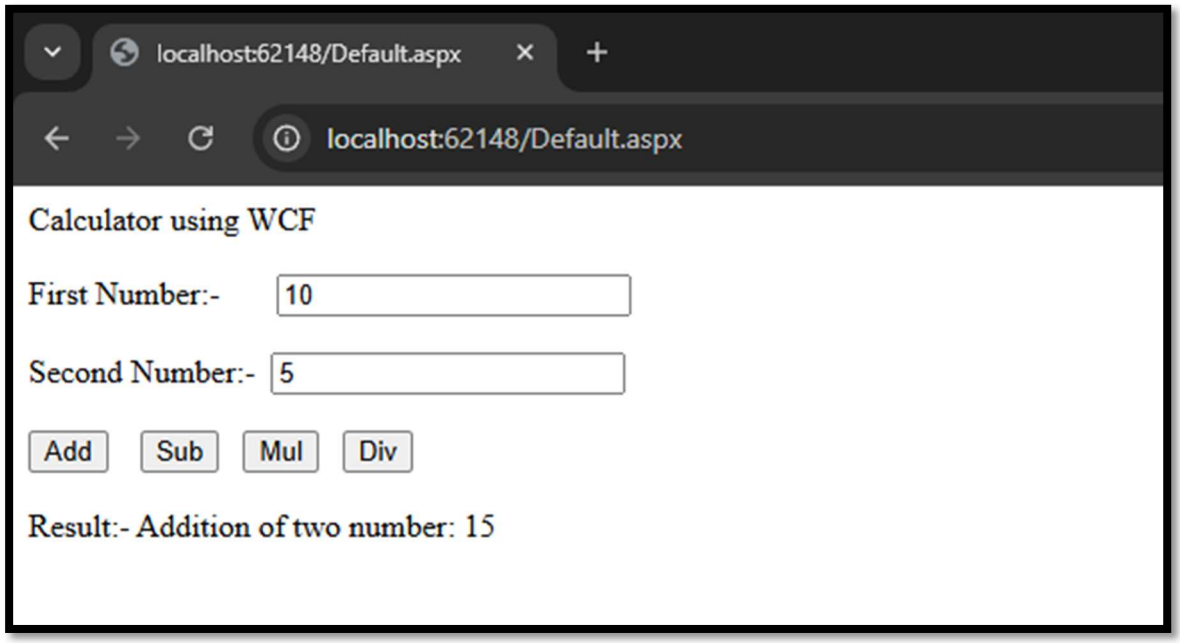
Default.aspx.cs

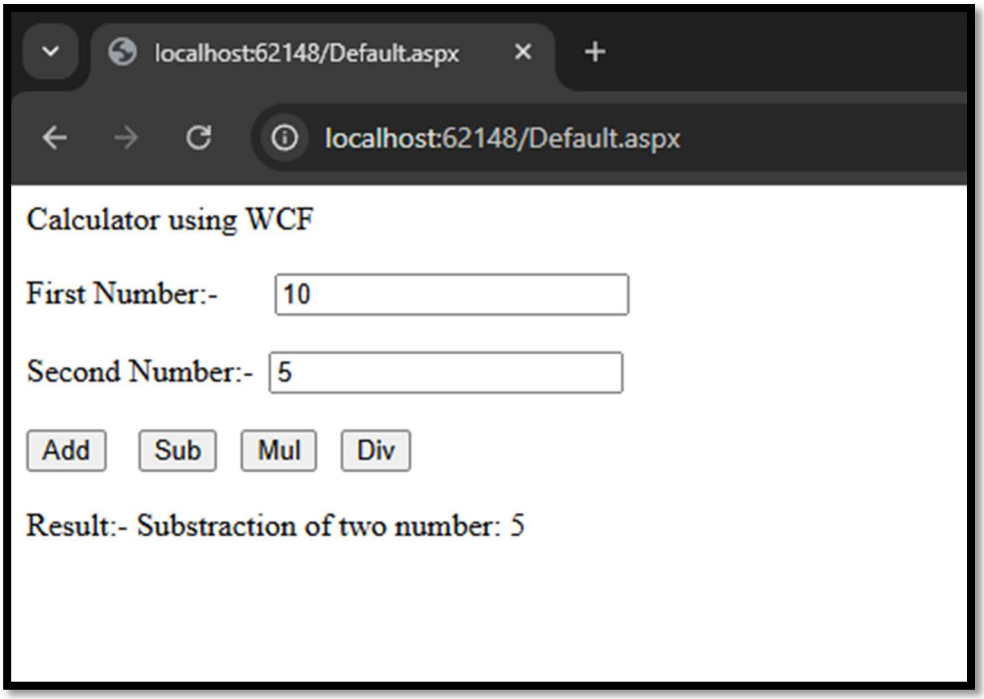
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;


public partial class _Default : System.Web.UI.Page
{
   ServiceReference1.ServiceClient service=new ServiceReference1.ServiceClient();
   protected void Page_Load(object sender, EventArgs e)
   {

   }
   protected void btnAdd_Click(object sender, EventArgs e)
   {
      double a= Convert.ToDouble(txt1.Text);
      double b= Convert.ToDouble(txt2.Text);
      double result= service.add(a, b);
      lblres.Text = "Addition of two number: " + result.ToString();
   }
   protected void btnSub_Click(object sender, EventArgs e)
   {
      double a = Convert.ToDouble(txt1.Text);
      double b = Convert.ToDouble(txt2.Text);
      double result = service.sub(a, b);
      lblres.Text = "Substraction of two number: " + result.ToString();
   }
   protected void btnMul_Click(object sender, EventArgs e)
   {
      double a = Convert.ToDouble(txt1.Text);
      double b = Convert.ToDouble(txt2.Text);
      double result = service.mul(a, b);
      lblres.Text = "Multiplication of two number: " + result.ToString();
   }
   protected void btnDiv_Click(object sender, EventArgs e)
   {
      double a = Convert.ToDouble(txt1.Text);
      double b = Convert.ToDouble(txt2.Text);
      double result = service.div(a, b);
      lblres.Text = "Division of two number: " + result.ToString();
```

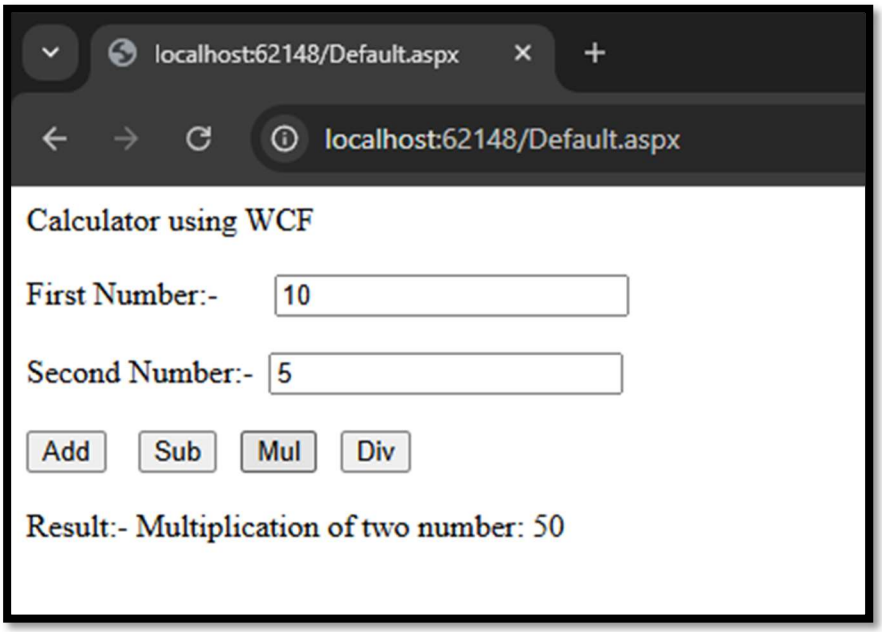
```
    }
}
```

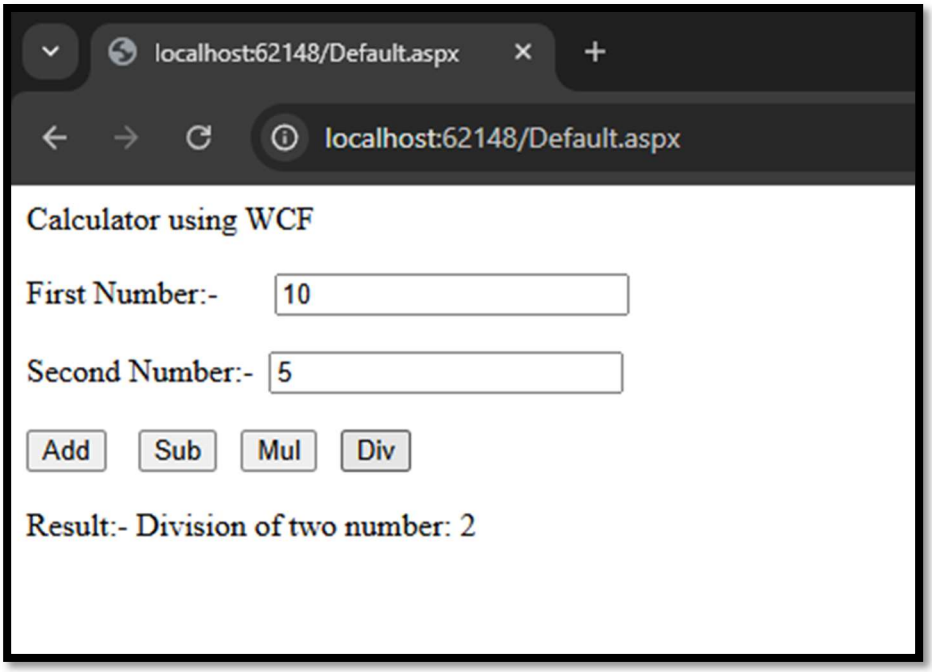
Output:


localhost:62148/Default.aspx
localhost:62148/Default.aspx
Calculator using WCF
First Number:-
Second Number:-
Add
Sub
Mul
Div
Result:- Label


localhost:62148/Default.aspx
localhost:62148/Default.aspx
Calculator using WCF
First Number:- 10
Second Number:- 5
Add
Sub
Mul
Div
Result:- Addition of two number: 15


localhost:62148/Default.aspx
localhost:62148/Default.aspx
Calculator using WCF
First Number:-
10
Second Number:-
5
Add
Sub
Mul
Div
Result:- Substraction of two number: 5


localhost:62148/Default.aspx
localhost:62148/Default.aspx
Calculator using WCF
First Number:-
10
Second Number:-
5
Add
Sub
Mul
Div
Result:- Multiplication of two number: 50


localhost:62148/Default.aspx
localhost:62148/Default.aspx
Calculator using WCF
First Number:-
10
Second Number:-
5
Add
Sub
Mul
Div
Result:- Division of two number: 2

Simple Interest Using WCF

Code:

IService.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;


// NOTE: You can use the "Rename" command on the "Refactor" menu to change the interface name
"IService" in both code and config file together.
[ServiceContract]
public interface IService
{
[OperationContract]
string GetData(int value);


[OperationContract]
double SI(double p, double n, double r);


   [OperationContract]
CompositeType GetDataUsingDataContract(CompositeType composite);
// TODO: Add your service operations here
}


// Use a data contract as illustrated in the sample below to add composite types to service operations.
[DataContract]
public class CompositeType
{
bool boolValue = true;
string stringValue = "Hello ";


[DataMember]
public bool BoolValue
{
get { return boolValue; }
set { boolValue = value; }
```

```
}


[DataMember]
public string StringValue
{
get { return stringValue; }
set { stringValue = value; }
}
}
```

Service.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.Serialization;
using System.ServiceModel;
using System.ServiceModel.Web;
using System.Text;

// NOTE: You can use the "Rename" command on the "Refactor" menu to change the class name
"Service" in code, svc and config file together.
public class Service : IService
{
public string GetData(int value)
{
return string.Format("You entered: {0}", value);
}
public double SI(double p, double n, double r)
{
      return (p * n * r) / 100;
   }

public CompositeType GetDataUsingDataContract(CompositeType composite)
{
if (composite == null)
{
throw new ArgumentNullException("composite");
}
if (composite.BoolValue)
{
composite.StringValue += "Suffix";
}
```

```
return composite;
}
}
```

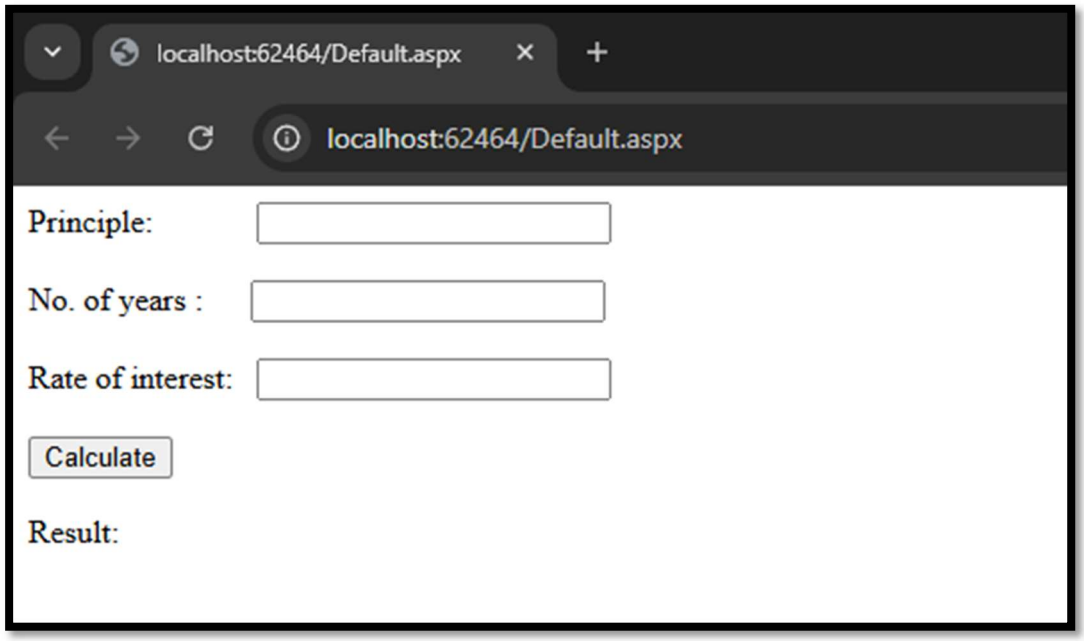
Default.aspx.cs

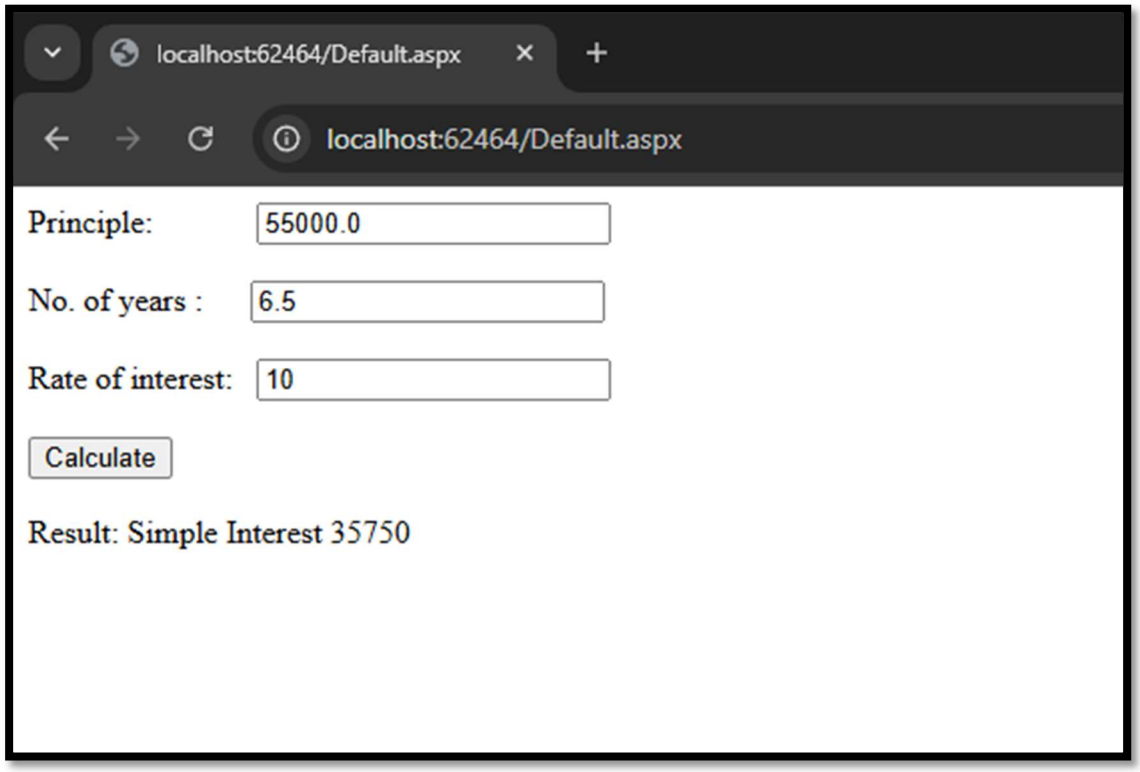
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class _Default : System.Web.UI.Page
{
   ServiceReference1.ServiceClient service = new ServiceReference1.ServiceClient();
   protected void Page_Load(object sender, EventArgs e)
   {

   }

   protected void btnCal_Click(object sender, EventArgs e)
   {
      double p = Convert.ToDouble(txtp.Text);
      double n = Convert.ToDouble(txtn.Text);
      double r = Convert.ToDouble(txtr.Text);
      double result = service.SI(p, n, r);

      lblResult.Text = "Simple Interest " + result.ToString();
   }
}
```

Default.aspx

```
<%@ Page Language="C#" AutoEventWireup="true" CodeFile="Default.aspx.cs" Inherits="_Default" %>
<!DOCTYPE html>
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
  <title></title>
</head>
<body>
  <form id="form1" runat="server">
    Principle:            
    <asp:TextBox ID="txtp" runat="server" />
    <br />
    <br />
    No. of years :     
    <asp:TextBox ID="txtn" runat="server" />
    <br />
    <br />
    Rate of interest:  
    <asp:TextBox ID="txtr" runat="server" />
    <br />
    <br />
    <asp:Button ID="btnCal" runat="server" OnClick="btnCal_Click" Text="Calculate" />
    <br />
    <br />
    Result:
    <asp:Label ID="lblResult" runat="server" Text="" />
    <div>
    </div>
  </form>
</body>
</html>
```

Output:


localhost:62464/Default.aspx
localhost:62464/Default.aspx
Principle:
No. of years :
Rate of interest:
Calculate
Result:


localhost:62464/Default.aspx
localhost:62464/Default.aspx
Principle:
55000.0
No. of years :
6.5
Rate of interest:
10
Calculate
Result: Simple Interest 35750

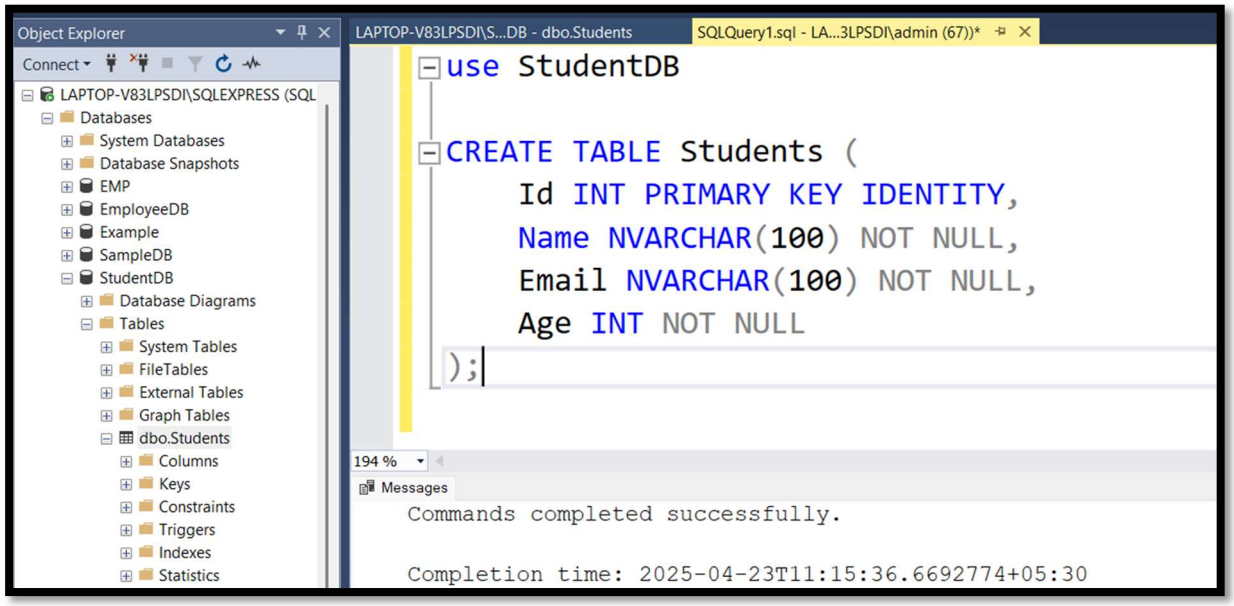
Practical no.: 16
Design web application using MVC framework

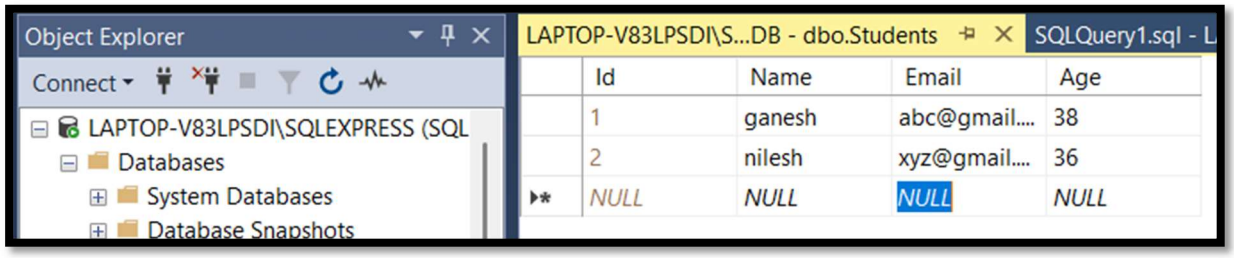
Database Setup

Create a Database in SQL Server

Open SQL Server Management Studio (SSMS) or Visual Studio SQL Server Object Explorer.
Create a Database named `StudentDB`.
Create a Table using the following SQL: Create Database `StudentDB`

```
use StudentDB

CREATE TABLE Students (
    Id INT PRIMARY KEY IDENTITY,
    Name NVARCHAR(100) NOT NULL,
    Email NVARCHAR(100) NOT NULL,
    Age INT NOT NULL
);
```





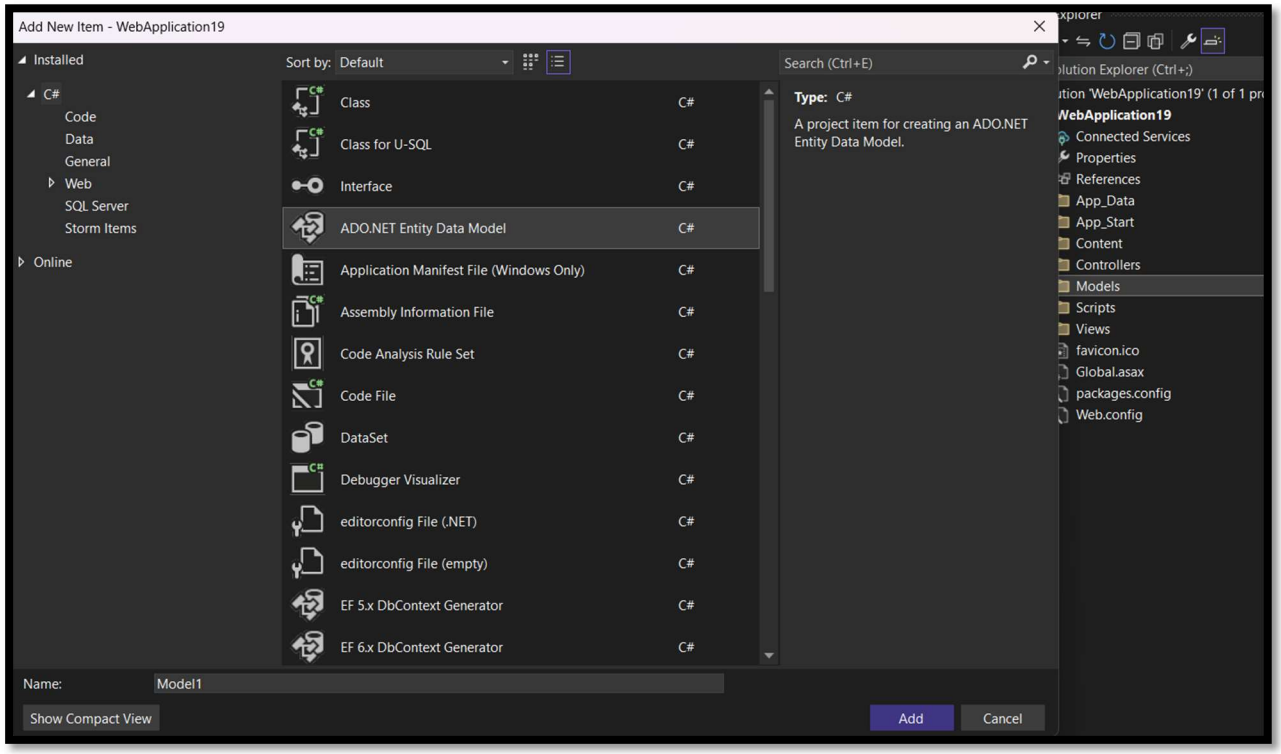
Step 2: Create a New ASP.NET MVC Project

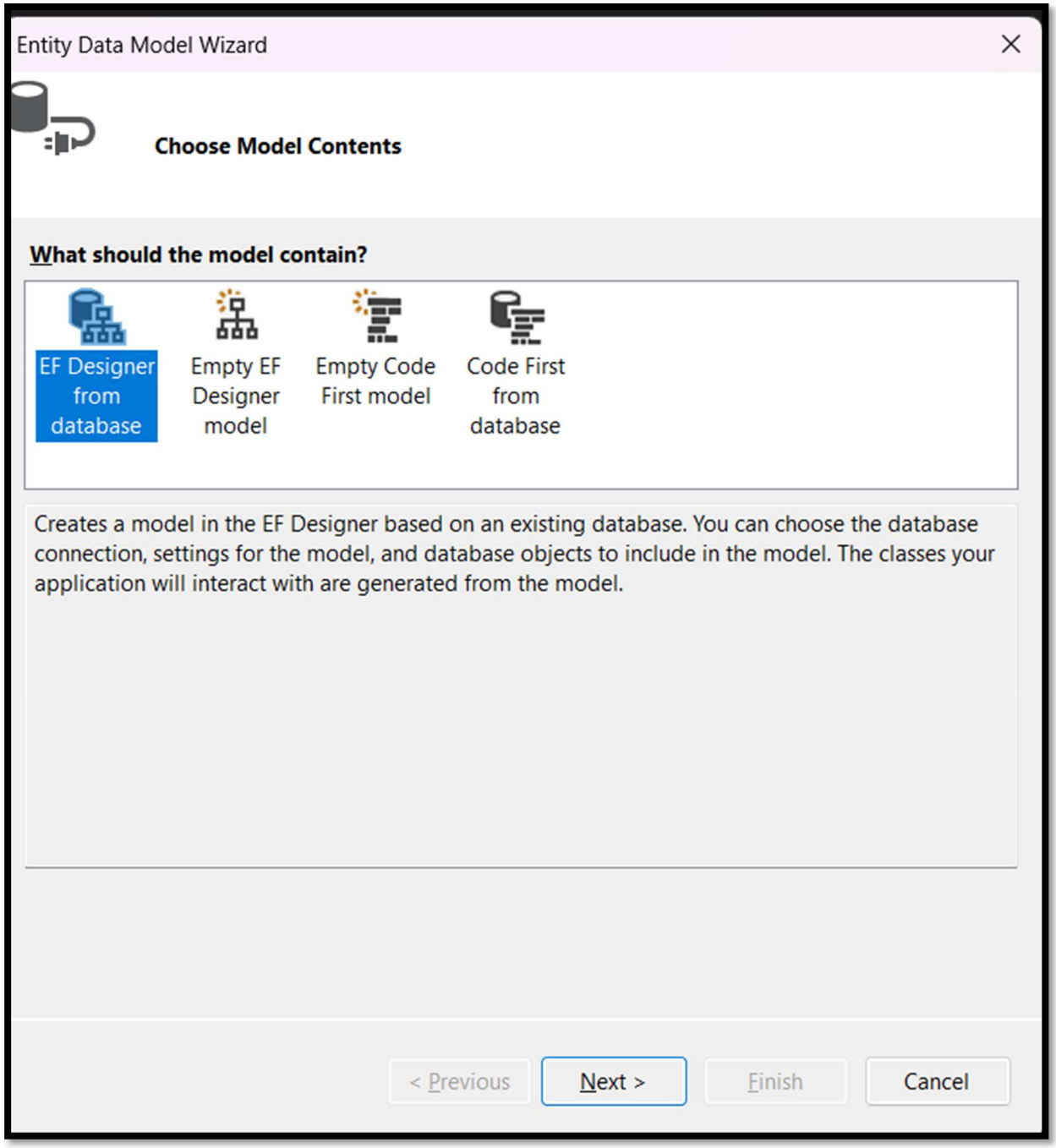
1. Open **Visual Studio**
2. Select **Create a new project**
3. Choose: **ASP.NET Web Application (.NET Framework)**
4. Name: `StudentMVCApp`
5. Choose **MVC** as the template
6. Click **Create**

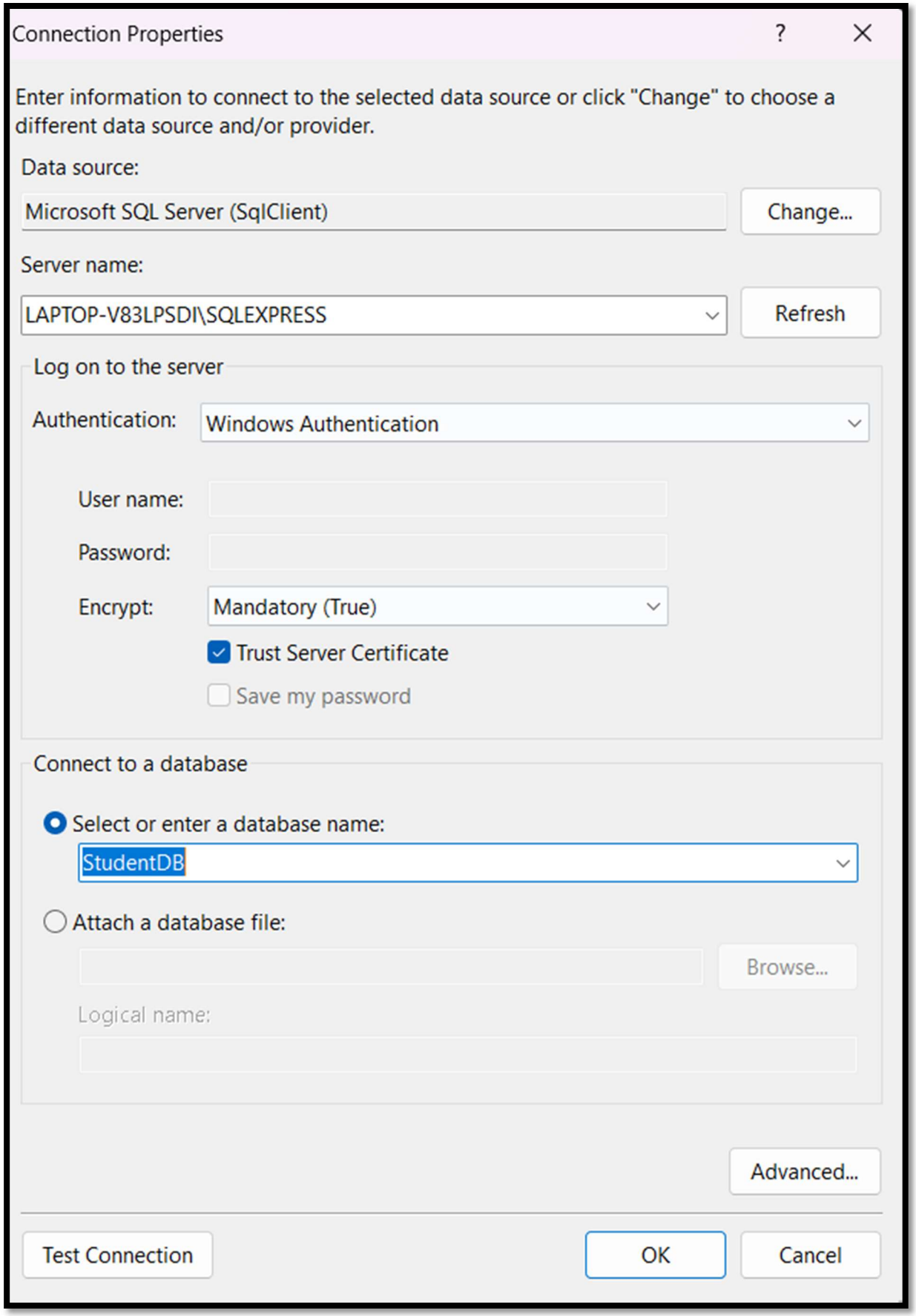
Entity Framework Model Setup

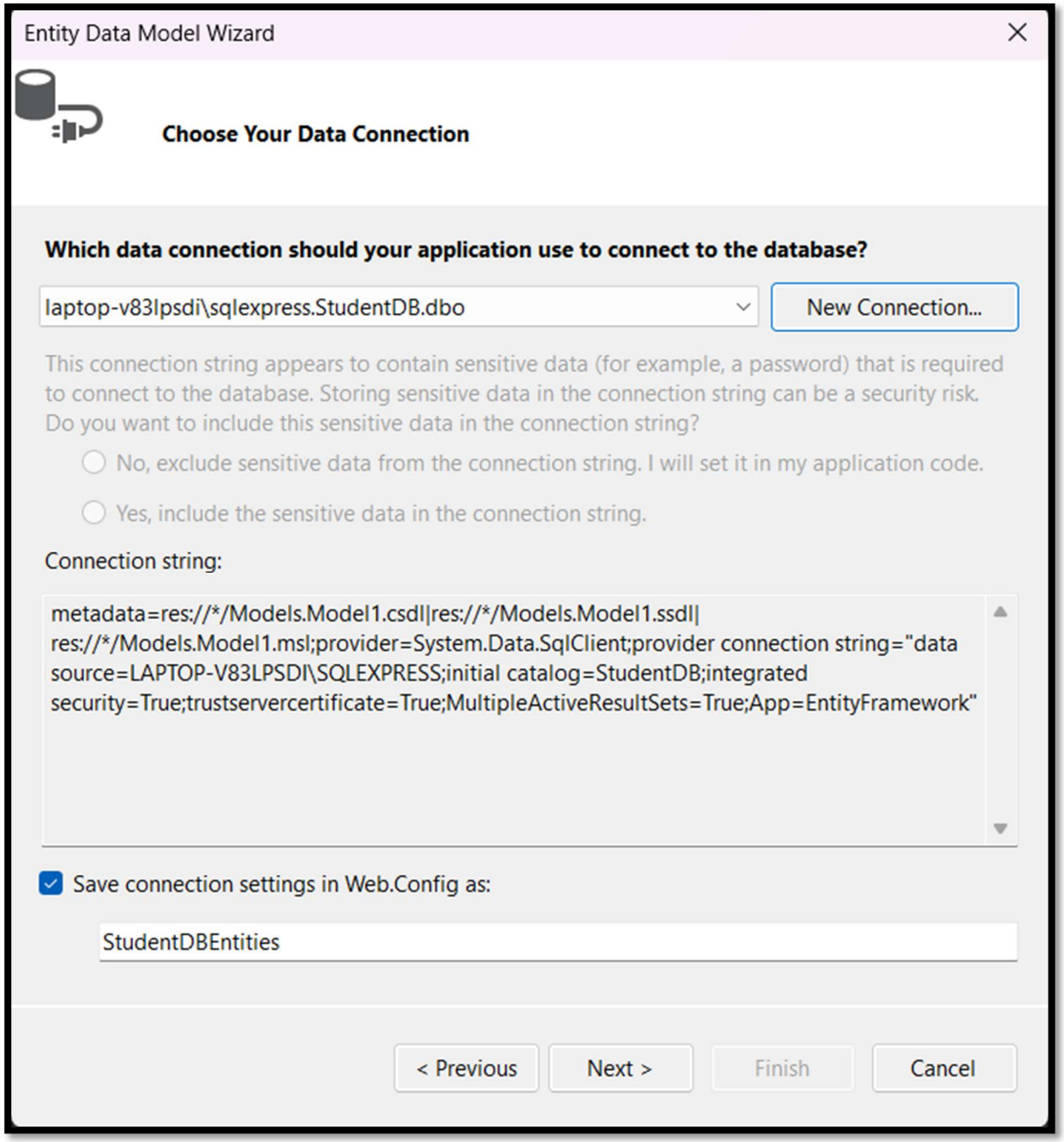
Step 3: Add Entity Framework Model

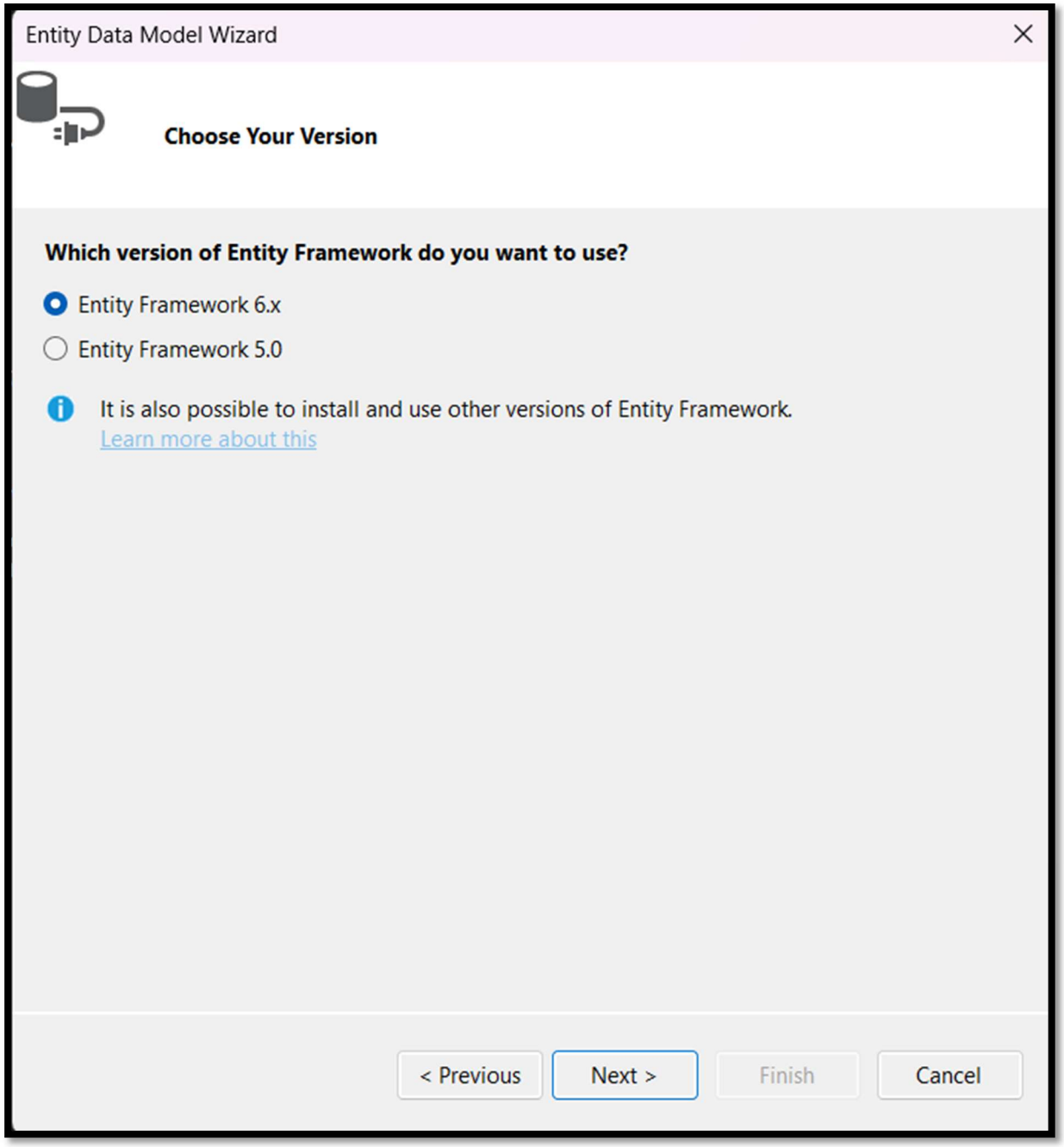
1. Right-click the **Models** folder → Add → New Item
2. Choose **ADO.NET Entity Data Model**
3. Name it: `StudentModel.edmx`
4. Choose: "EF Designer from database"
5. Select your SQL Server database (`StudentDB`)
6. Select the `Students` table
7. Finish to generate model classes

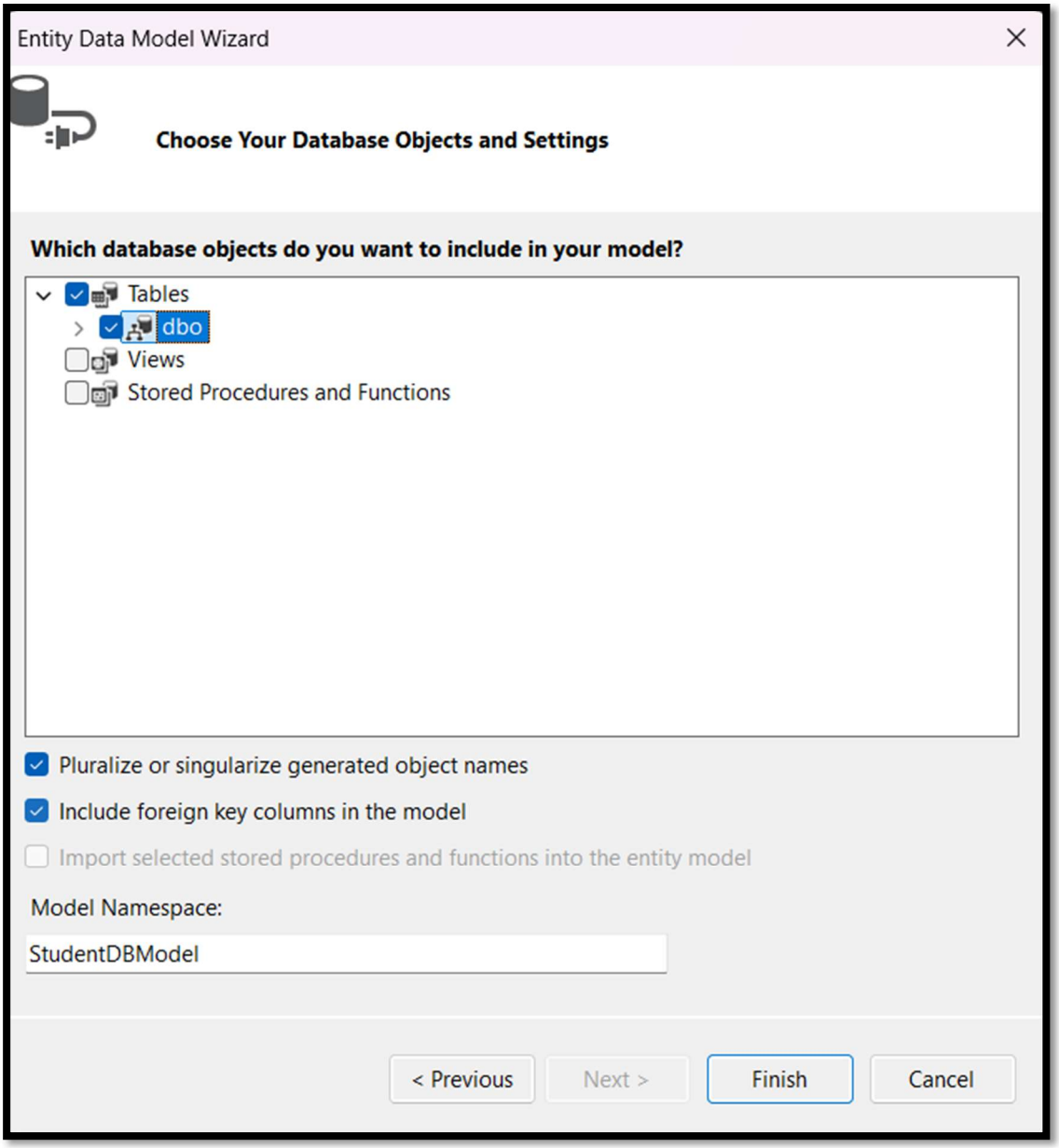


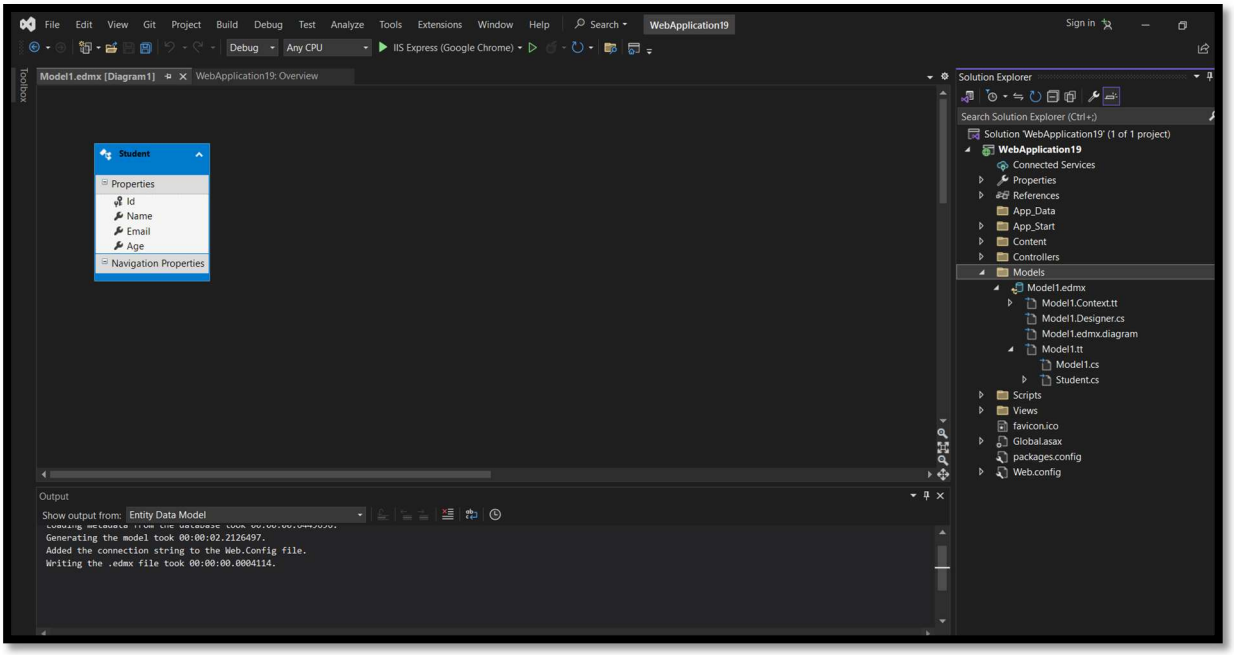

Entity Data Model Wizard
Choose Model Contents
What should the model contain?
EF Designer from database
Empty EF Designer model
Empty Code First model
Code First from database
Creates a model in the EF Designer based on an existing database. You can choose the database connection, settings for the model, and database objects to include in the model. The classes your application will interact with are generated from the model.
< Previous
Next >
Finish
Cancel


Connection Properties
Enter information to connect to the selected data source or click "Change" to choose a different data source and/or provider.
Data source:
Microsoft SQL Server (SqlClient)
Change...
Server name:
LAPTOP-V83LPSDI\SQLEXPRESS
Refresh
Log on to the server
Authentication:
Windows Authentication
User name:
Password:
Encrypt:
Mandatory (True)
Trust Server Certificate
Save my password
Connect to a database
Select or enter a database name:
StudentDB
Attach a database file:
Browse...
Logical name:
Advanced...
Test Connection
OK
Cancel


Entity Data Model Wizard
Choose Your Data Connection
Which data connection should your application use to connect to the database?
laptop-v83lpsdi\sqlexpress.StudentDB.dbo
New Connection...
This connection string appears to contain sensitive data (for example, a password) that is required to connect to the database. Storing sensitive data in the connection string can be a security risk. Do you want to include this sensitive data in the connection string?
No, exclude sensitive data from the connection string. I will set it in my application code.
Yes, include the sensitive data in the connection string.
Connection string:
metadata=res://*/Models.Model1.csdl|res://*/Models.Model1.ssdl|
res://*/Models.Model1.msl;provider=System.Data.SqlClient;provider connection string="data
source=LAPTOP-V83LPSDI\SQLEXPRESS;initial catalog=StudentDB;integrated
security=True;trustservercertificate=True;MultipleActiveResultSets=True;App=EntityFramework"
Save connection settings in Web.Config as:
StudentDBEntities
< Previous
Next >
Finish
Cancel


Entity Data Model Wizard
Choose Your Version
Which version of Entity Framework do you want to use?
Entity Framework 6.x
Entity Framework 5.0
It is also possible to install and use other versions of Entity Framework.
Learn more about this
< Previous
Next >
Finish
Cancel

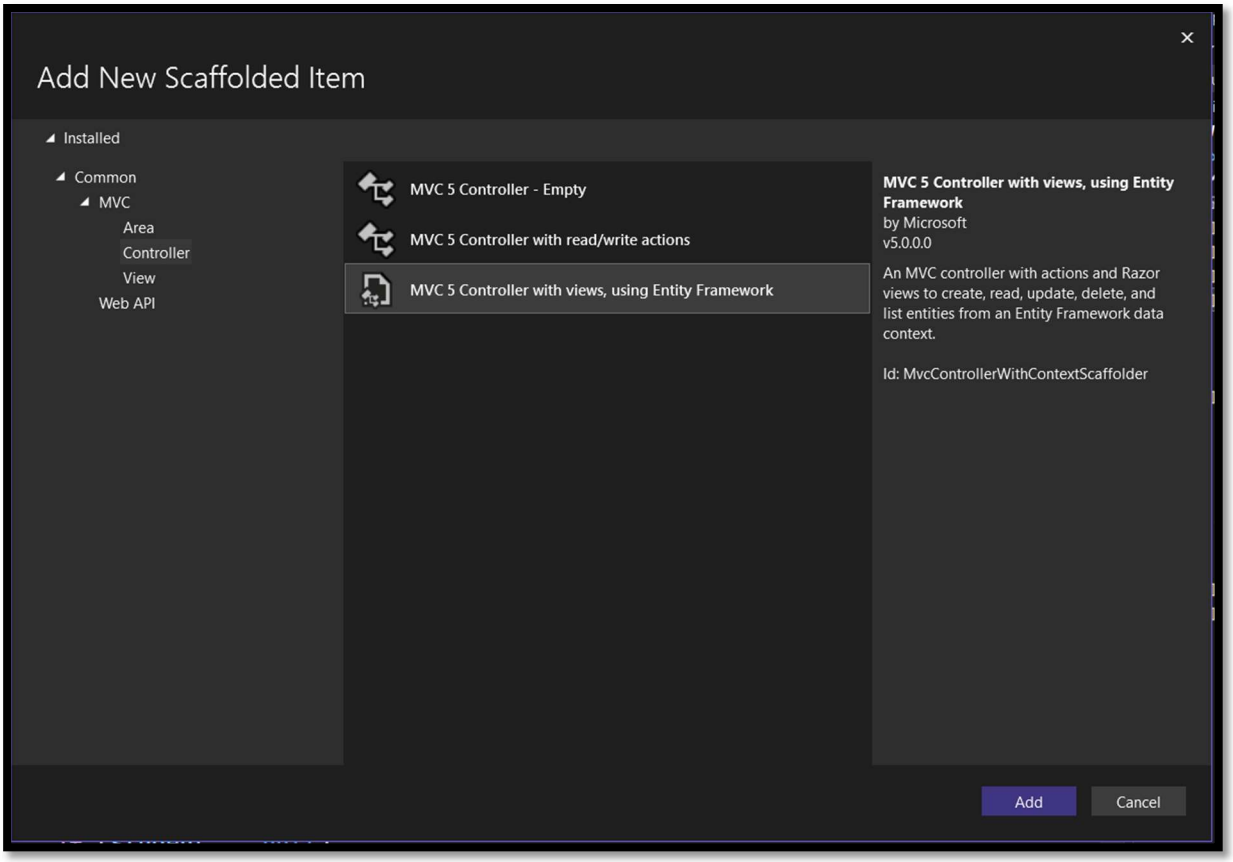

Entity Data Model Wizard
Choose Your Database Objects and Settings
Which database objects do you want to include in your model?
Tables
dbo
Views
Stored Procedures and Functions
Pluralize or singularize generated object names
Include foreign key columns in the model
Import selected stored procedures and functions into the entity model
Model Namespace:
StudentDBModel
< Previous
Next >
Finish
Cancel

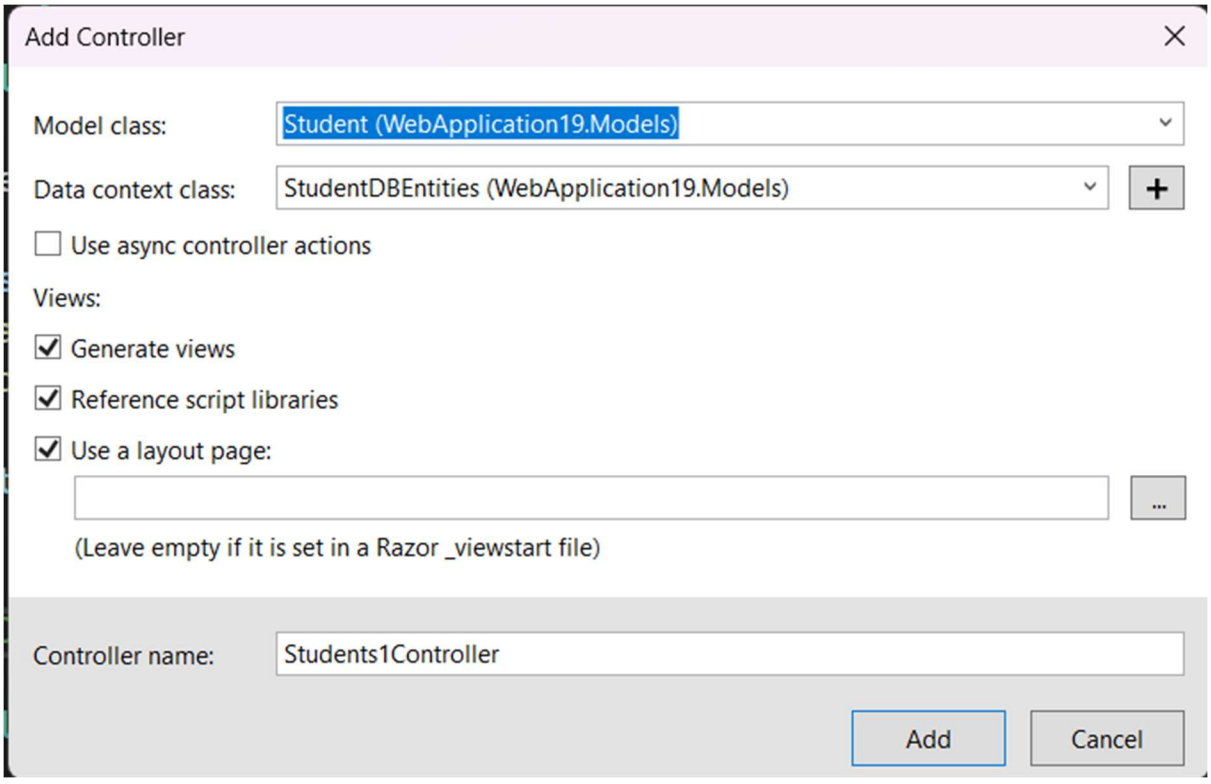


Step 4: Create Controller

1. Right-click **Controllers** → Add → Controller
2. Choose: **MVC 5 Controller with views, using Entity Framework**
3. Model class: `Student`
4. Data context: `StudentDBEntities` (if using .edmx) or `StudentDBContext`

Click **Add**


Add New Scaffolded Item
Installed
Common
MVC
Area
Controller
View
Web API
MVC 5 Controller - Empty
MVC 5 Controller with read/write actions
MVC 5 Controller with views, using Entity Framework
MVC 5 Controller with views, using Entity Framework
by Microsoft
v5.0.0.0
An MVC controller with actions and Razor views to create, read, update, delete, and list entities from an Entity Framework data context.
Id: MvcControllerWithContextScaffolder
Add
Cancel


Add Controller
Model class:
Student (WebApplication19.Models)
Data context class:
StudentDBEntities (WebApplication19.Models)
Use async controller actions
Views:
Generate views
Reference script libraries
Use a layout page:
(Leave empty if it is set in a Razor _viewstart file)
Controller name:
Students1Controller
Add
Cancel

Code:

Create Views

Example: Views/Students/Index.cshtml

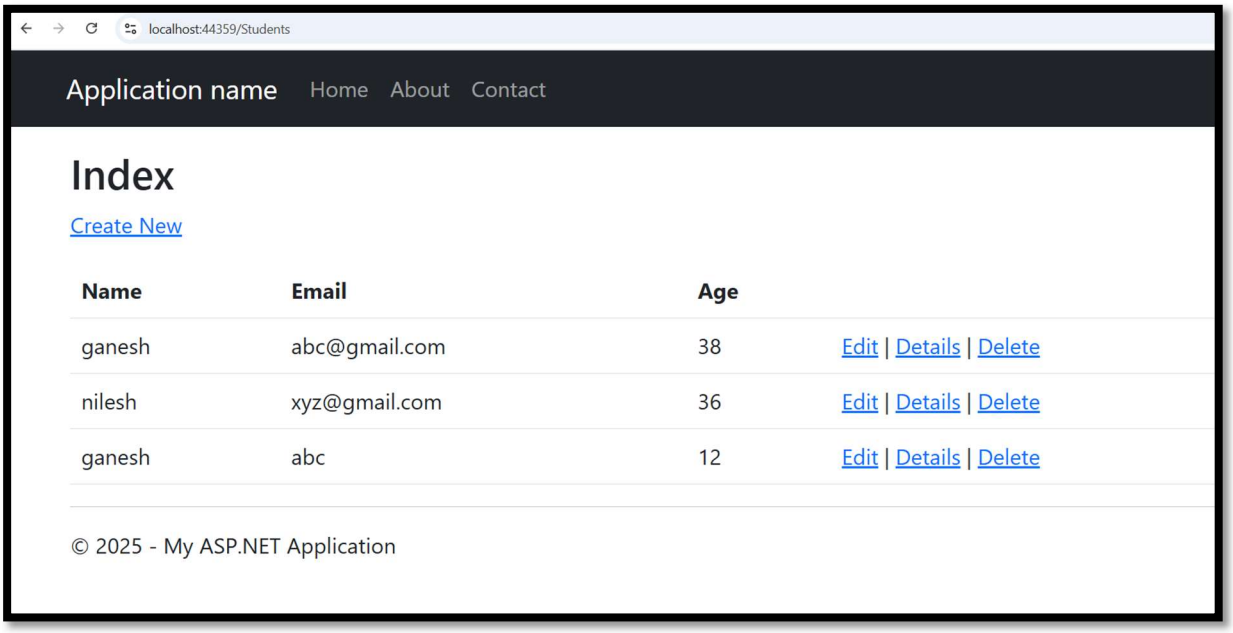
```
@model IEnumerable<StudentMVCApp.Models.Student>
@{
  ViewBag.Title = "Student List";
}
<h2>Student List</h2>
<p>@Html.ActionLink("Create New", "Create")</p>

<table class="table">
  <tr>
    <th>@Html.DisplayNameFor(model => model.Name)</th>
    <th>@Html.DisplayNameFor(model => model.Email)</th>
    <th>@Html.DisplayNameFor(model => model.Age)</th>
    <th>Actions</th>
  </tr>
@foreach (var item in Model) {
  <tr>
    <td>@item.Name</td>
    <td>@item.Email</td>
    <td>@item.Age</td>
    <td>
      @Html.ActionLink("Edit", "Edit", new { id = item.Id }) |
      @Html.ActionLink("Details", "Details", new { id = item.Id }) |
      @Html.ActionLink("Delete", "Delete", new { id = item.Id })
    </td>
  </tr>
}
</table>
```

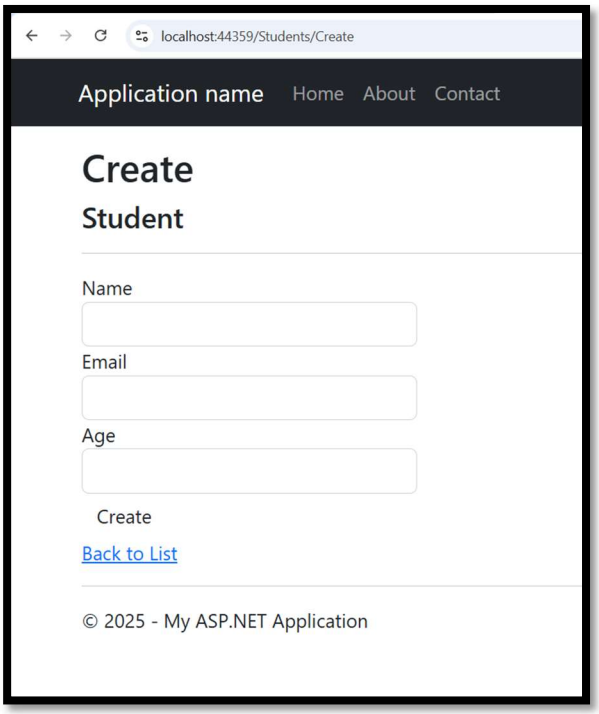
Set Default Route

In `App_Start/RouteConfig.cs`, change default route to:

```
csharp
CopyEdit
defaults: new { controller = "Students", action = "Index", id =
UrlParameter.Optional }
```

Output:


localhost:44359/Students
Application name
Home
About
Contact
Index
Create New
Name
Email
Age
ganesh
abc@gmail.com
38
Edit | Details | Delete
nilesh
xyz@gmail.com
36
Edit | Details | Delete
ganesh
abc
12
Edit | Details | Delete
© 2025 - My ASP.NET Application


localhost:44359/Students/Create
Application name
Home
About
Contact
Create
Student
Name
Email
Age
Create
Back to List
© 2025 - My ASP.NET Application