.NET Technologies (01CE0523)

5th Semester

4 Credits

Prof. Ravikumar R N

Dept. of Computer Engineering

Marwadi University

Marwadi Chandarana Group

NAAC A+

Objectives

- Net Technologies are blend of technologies supported by Microsoft .Net Framework that allows user to create various applications.
- Students will be able to work with various technologies provided by Microsoft .NET platform.

Course Outcomes

- To Review the components of .Net Framework
- To practice Web based application
- To create web applications using MVC framework
- To practice basic database application using ADO.net
- To designing, developing, and deploying APIs

Unit 2

ASP.NET Web Application

Contents

- ASP.Net Web Application
- Page life cycle of ASP.NET Application
- Web Controls (Button, TextBox, CheckBox, Image etc.)
- Rich Controls (Calendar, AdRotator)
- Validation Controls
- State management
- Cookie
- Session

ASP.NET

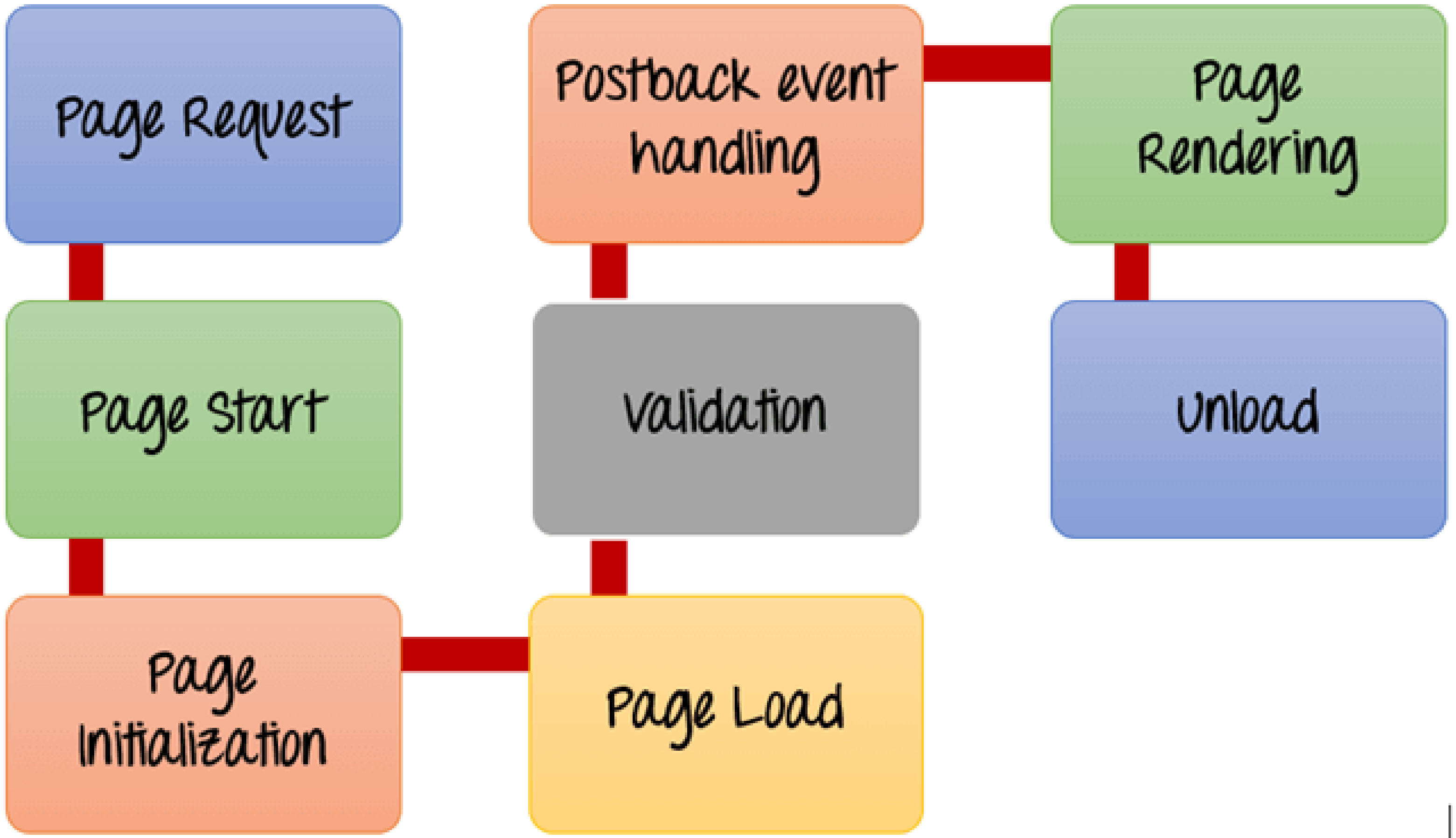
- ASP stands for Active Server Pages.
- ASP is a development framework for building dynamic web pages.
- ASP and ASP.NET are **server side** technologies.
- Both technologies enable computer code to be executed by an Internet server.
- When a browser requests an ASP or ASP.NET file, the ASP engine reads the file, executes any code in the file, and returns the result to the browser.
- ASP.NET was released in 2002 as a successor to Classic ASP.
- ASP.NET pages have the **extension .aspx** and are normally written in C# aka (C sharp).
- Latest version 5.2.9

ASP.NET

- ASP.NET Web Pages is an SPA application model (Single Page Application).
- A Single-Page Application is an app that doesn't need to reload the page during its use and works within a browser.
- Think of the apps you use daily: Facebook, Google Maps, GitHub etc.
- The SPA model is quite similar to PHP and Classic ASP.

ASP.Net Page Lifecycle

- When an ASP.Net page is called, it goes through a particular lifecycle. This is done before the response is sent to the user.



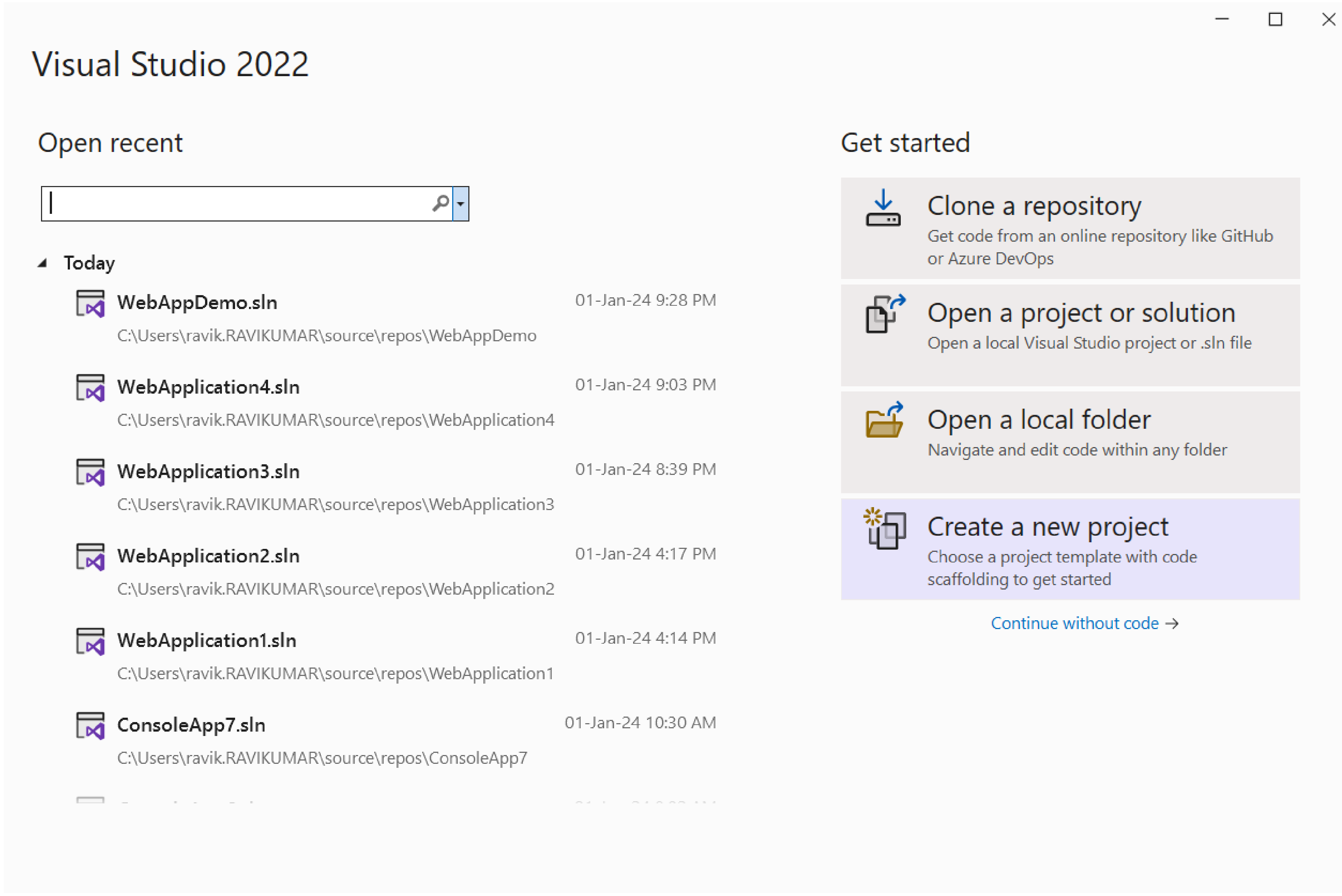
ASP.Net Page Lifecycle

- Page Request– This is when the page is first requested from the server. When the page is requested, the server checks if it is requested for the first time. If so, then it needs to compile the page, parse the response and send it across to the user. If it is not the first time the page is requested, the cache is checked to see if the page output exists. If so, that response is sent to the user.
- Page Start – During this time, 2 objects, known as the Request and Response object are created. The Request object is used to hold all the information which was sent when the page was requested. The Response object is used to hold the information which is sent back to the user.
- Page Initialization – During this time, all the controls on a web page is initialized. So if you have any label, textbox or any other controls on the web form, they are all initialized.
- Page Load – This is when the page is actually loaded with all the default values. So if a textbox is supposed to have a default value, that value is loaded during the page load time.

ASP.Net Page Lifecycle

- Validation – Sometimes there can be some validation set on the form. For example, there can be a validation which says that a list box should have a certain set of values. If the condition is false, then there should be an error in loading the page.
- Postback event handling – This event is triggered if the same page is being loaded again. This happens in response to an earlier event. Sometimes there can be a situation that a user clicks on a submit button on the page. In this case, the same page is displayed again. In such a case, the Postback event handler is called.
- Page Rendering – This happens just before all the response information is sent to the user. All the information on the form is saved, and the result is sent to the user as a complete web page (mirror).
- Unload – Once the page output is sent to the user, there is no need to keep the ASP.net web form objects in memory. So the unloading process involves removing all unwanted objects from memory.

First Web Application



Visual Studio 2022

Open recent

Today

WebAppDemo.sln 01-Jan-24 9:28 PM
C:\Users\ravik.RAVIKUMAR\source\repos\WebAppDemo

WebApplication4.sln 01-Jan-24 9:03 PM
C:\Users\ravik.RAVIKUMAR\source\repos\WebApplication4

WebApplication3.sln 01-Jan-24 8:39 PM
C:\Users\ravik.RAVIKUMAR\source\repos\WebApplication3

WebApplication2.sln 01-Jan-24 4:17 PM
C:\Users\ravik.RAVIKUMAR\source\repos\WebApplication2

WebApplication1.sln 01-Jan-24 4:14 PM
C:\Users\ravik.RAVIKUMAR\source\repos\WebApplication1

ConsoleApp7.sln 01-Jan-24 10:30 AM
C:\Users\ravik.RAVIKUMAR\source\repos\ConsoleApp7

Get started

Clone a repository
Get code from an online repository like GitHub or Azure DevOps

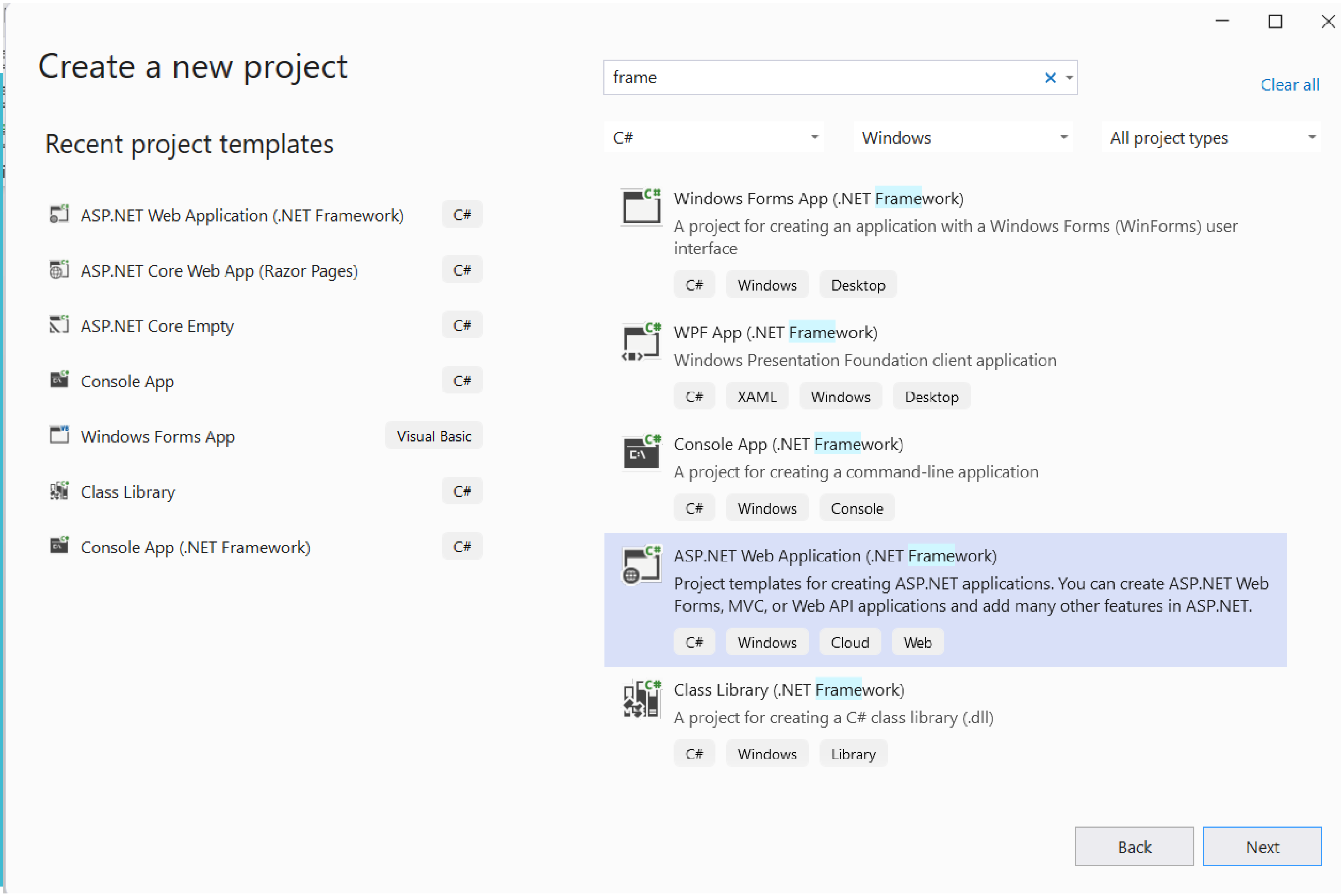
Open a project or solution
Open a local Visual Studio project or .sln file

Open a local folder
Navigate and edit code within any folder

Create a new project
Choose a project template with code scaffolding to get started

Continue without code →

First Web Application



Create a new project

Recent project templates

ASP.NET Web Application (.NET Framework)	C#
ASP.NET Core Web App (Razor Pages)	C#
ASP.NET Core Empty	C#
Console App	C#
Windows Forms App	Visual Basic
Class Library	C#
Console App (.NET Framework)	C#

frame

Clear all

C# | Windows | All project types

Windows Forms App (.NET Framework)
A project for creating an application with a Windows Forms (WinForms) user interface
C# Windows Desktop

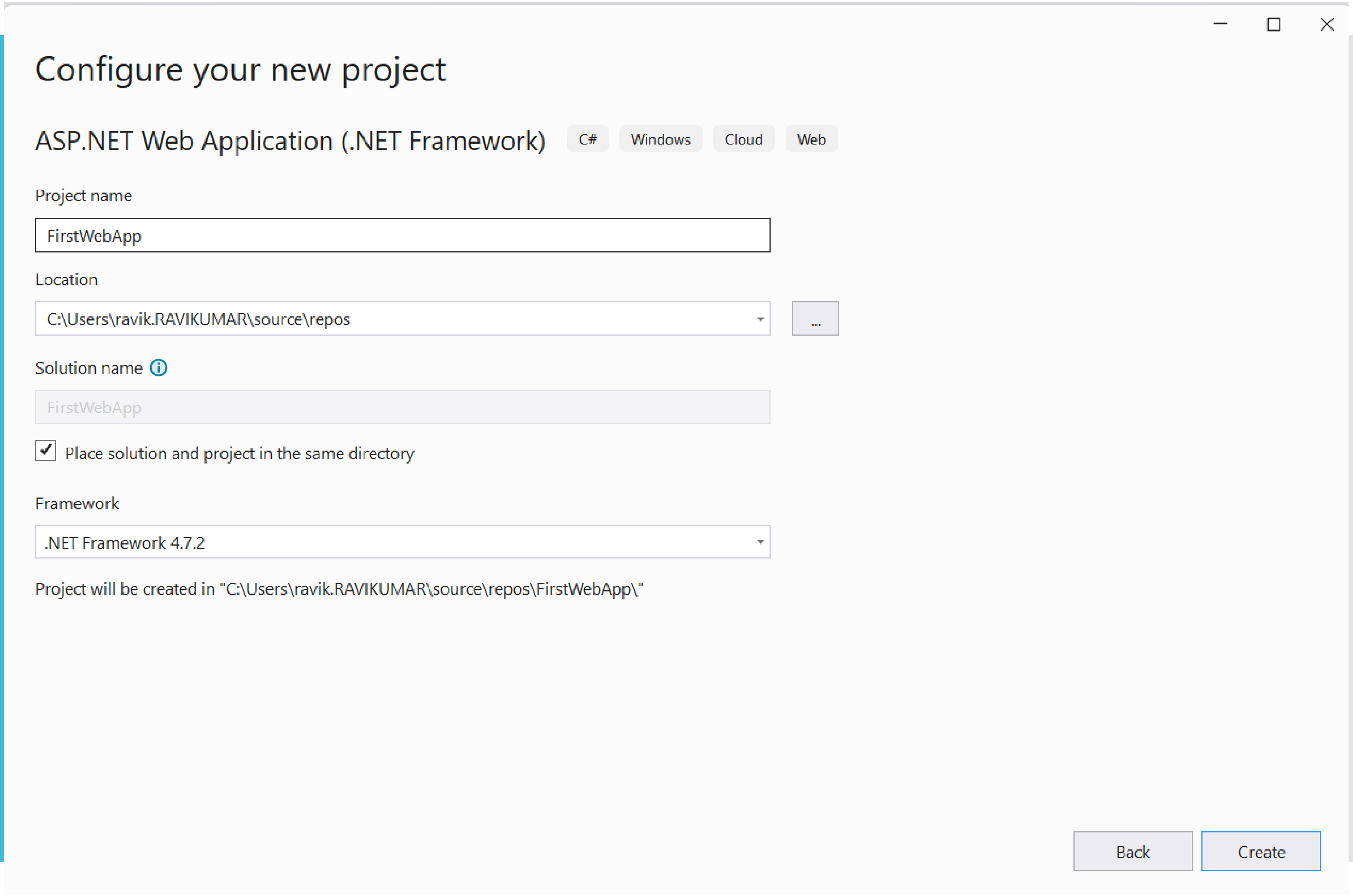
WPF App (.NET Framework)
Windows Presentation Foundation client application
C# XAML Windows Desktop

Console App (.NET Framework)
A project for creating a command-line application
C# Windows Console

ASP.NET Web Application (.NET Framework)
Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.
C# Windows Cloud Web

Class Library (.NET Framework)
A project for creating a C# class library (.dll)
C# Windows Library

First Web Application



ASP.NET Web Application (.NET Framework)

Project name: FirstWebApp

Location: C:\Users\ravik.RAVIKUMAR\source\repos

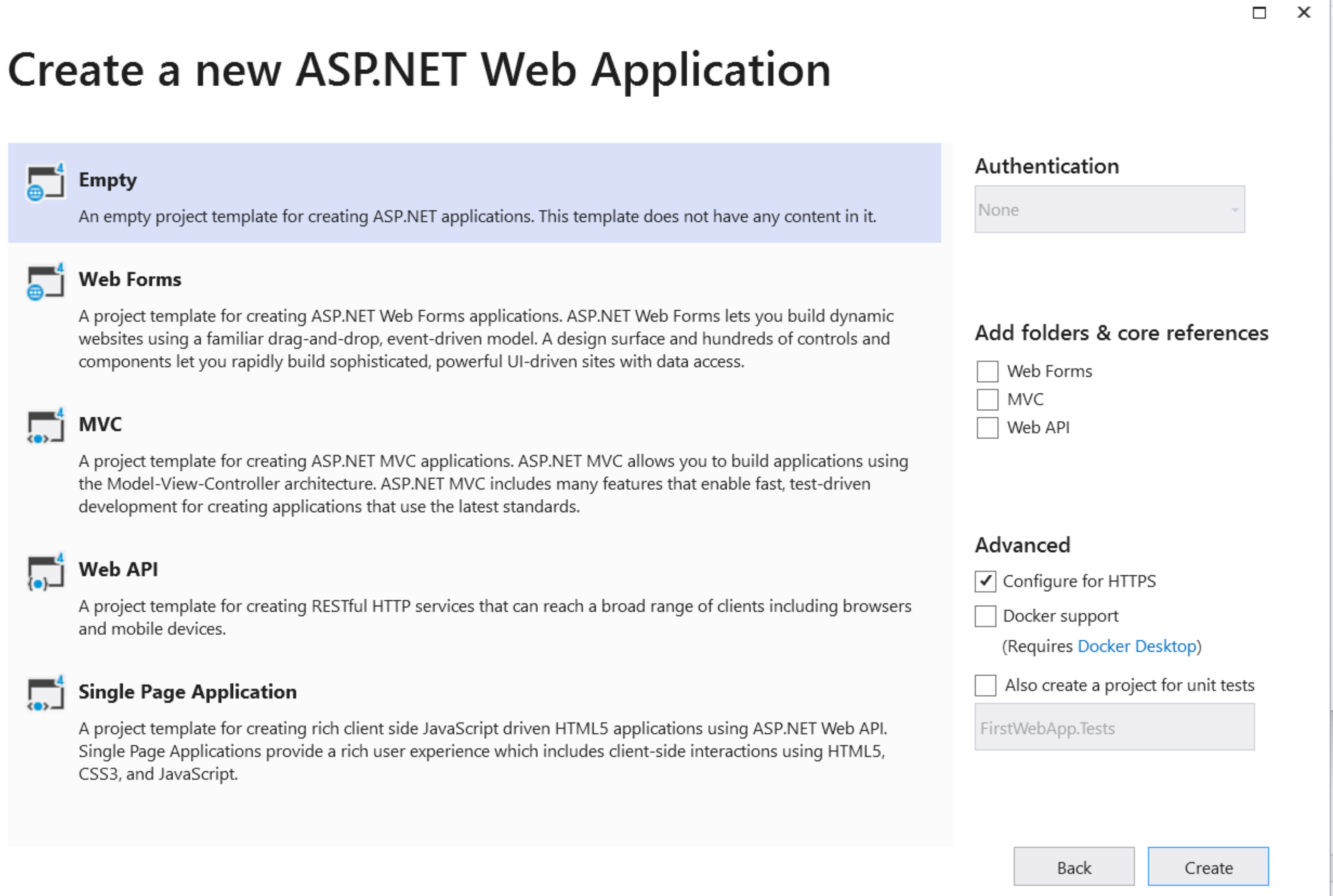
Solution name: FirstWebApp

☑ Place solution and project in the same directory

Framework: .NET Framework 4.7.2

Project will be created in "C:\Users\ravik.RAVIKUMAR\source\repos\FirstWebApp\"

First Web Application



Create a new ASP.NET Web Application

Empty

An empty project template for creating ASP.NET applications. This template does not have any content in it.

Web Forms

A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.

MVC

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.

Web API

A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.

Single Page Application

A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

Authentication

None

Add folders & core references

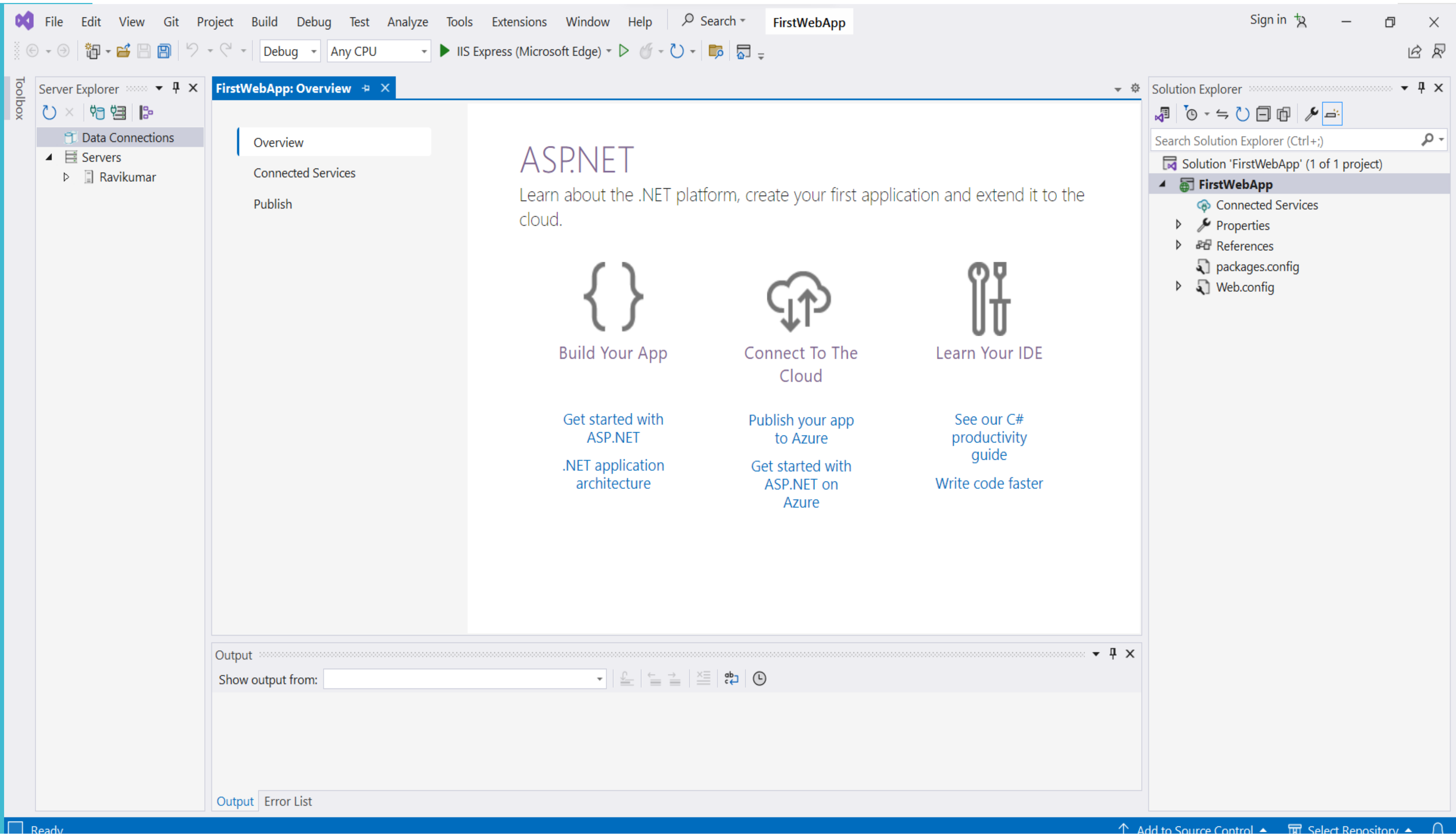
- [ ] Web Forms
- [ ] MVC
- [ ] Web API

Advanced

- [✓] Configure for HTTPS
- [ ] Docker support
  (Requires Docker Desktop)
- [ ] Also create a project for unit tests

FirstWebApp.Tests

First Web Application

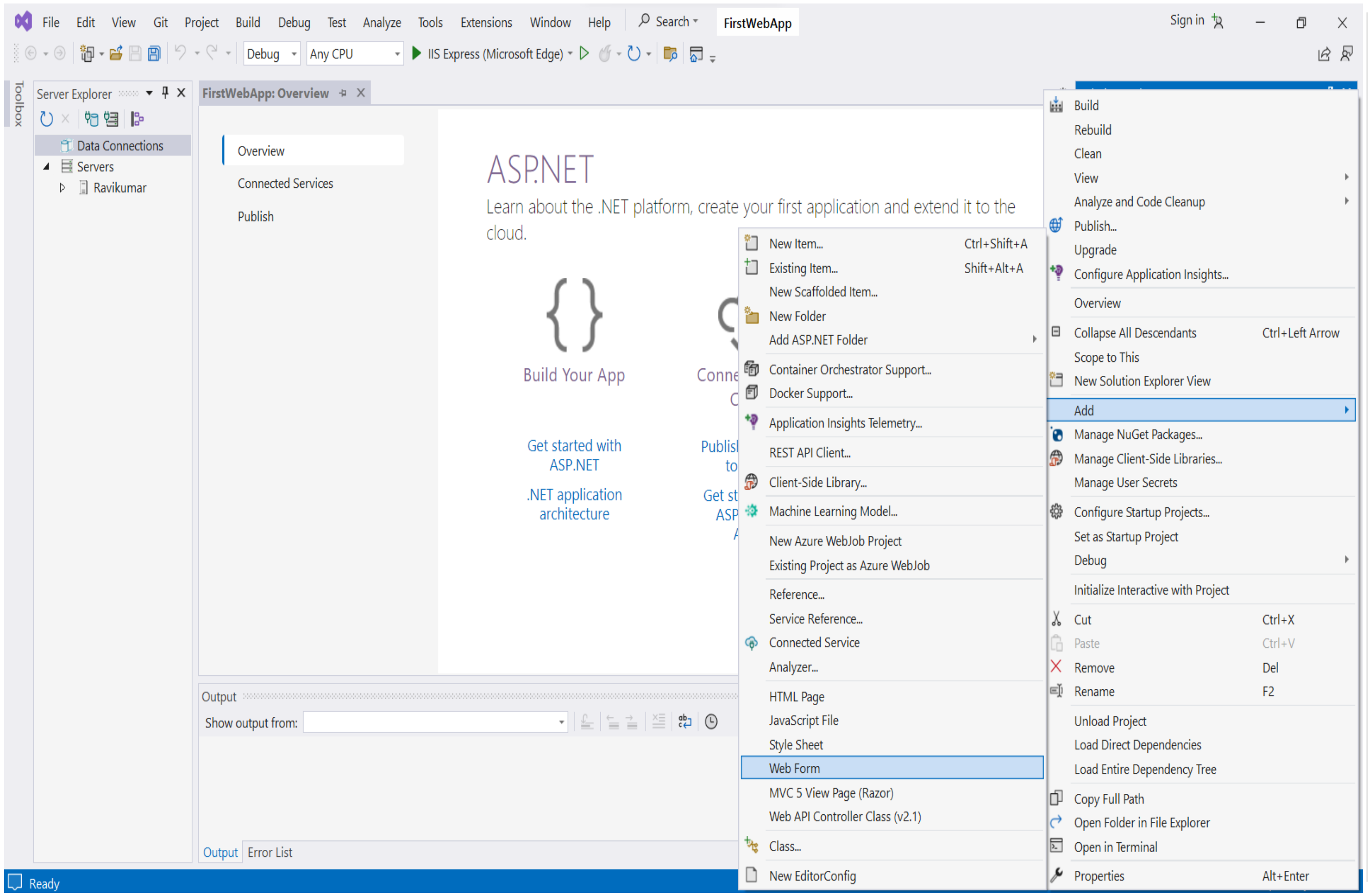


ASP.NET

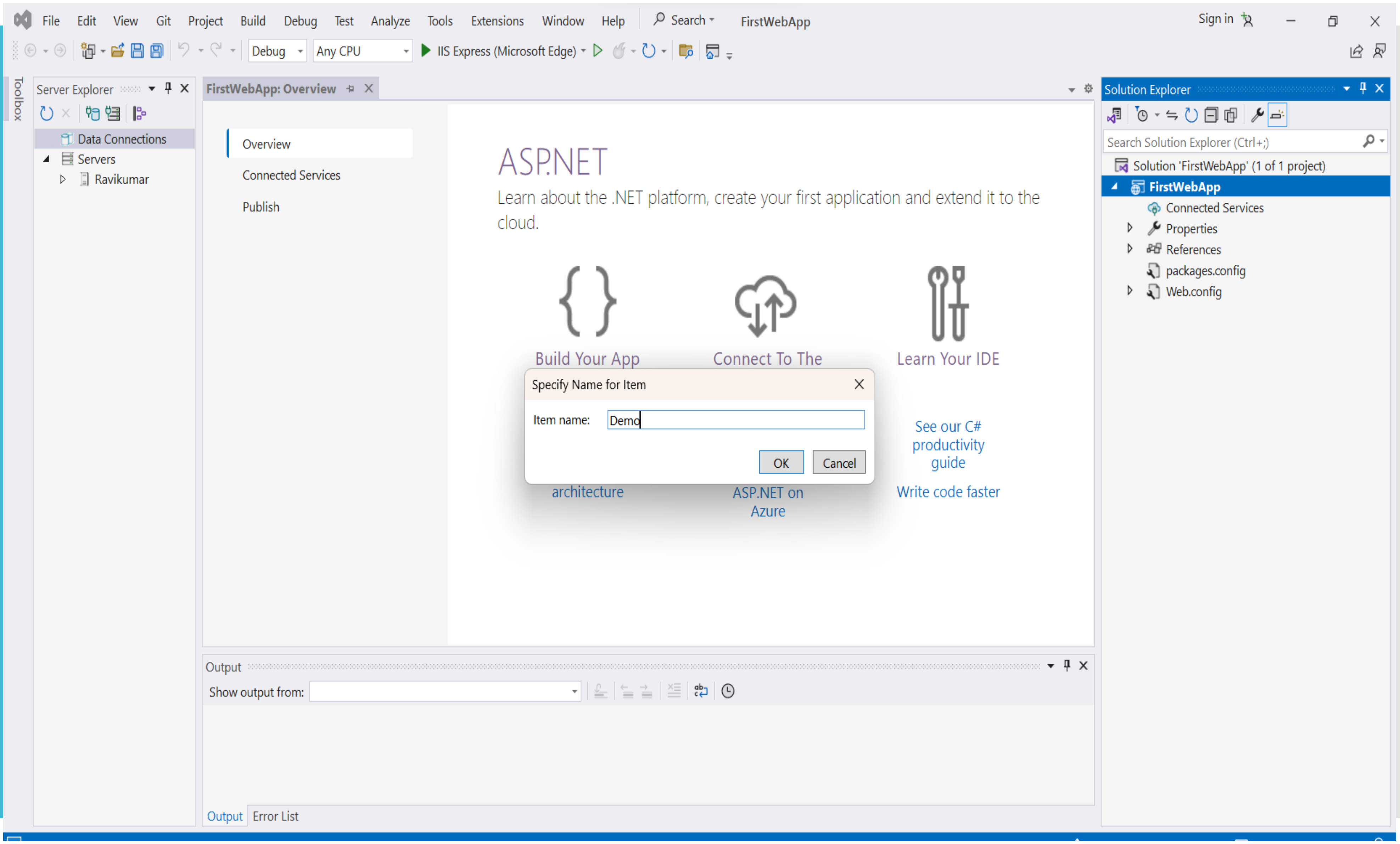
Learn about the .NET platform, create your first application and extend it to the cloud.

Build Your App	Connect To The Cloud	Learn Your IDE
Get started with ASP.NET	Publish your app to Azure	See our C# productivity guide
.NET application architecture	Get started with ASP.NET on Azure	Write code faster

First Web Application



First Web Application

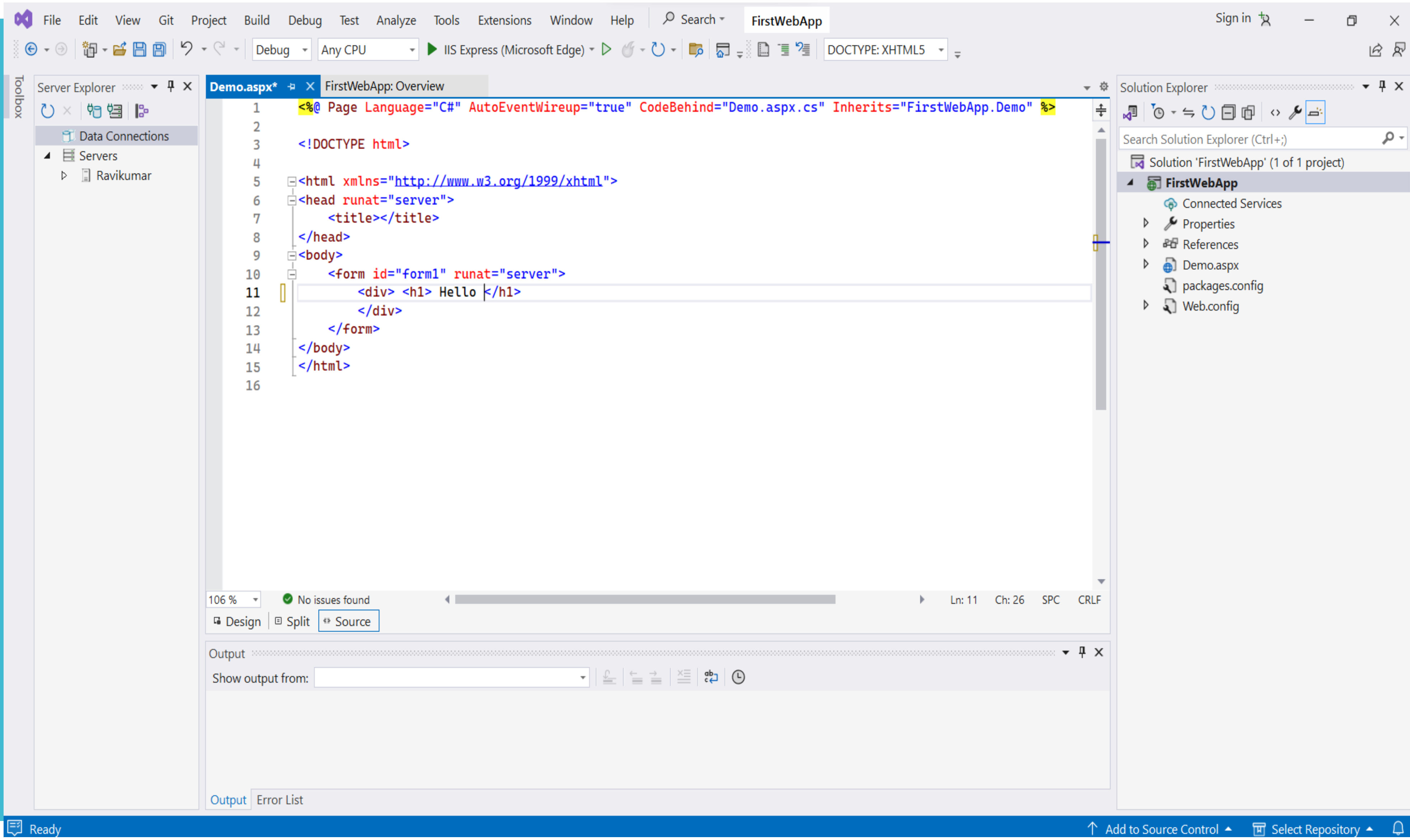


Specify Name for Item

Item name: Demo

OK Cancel

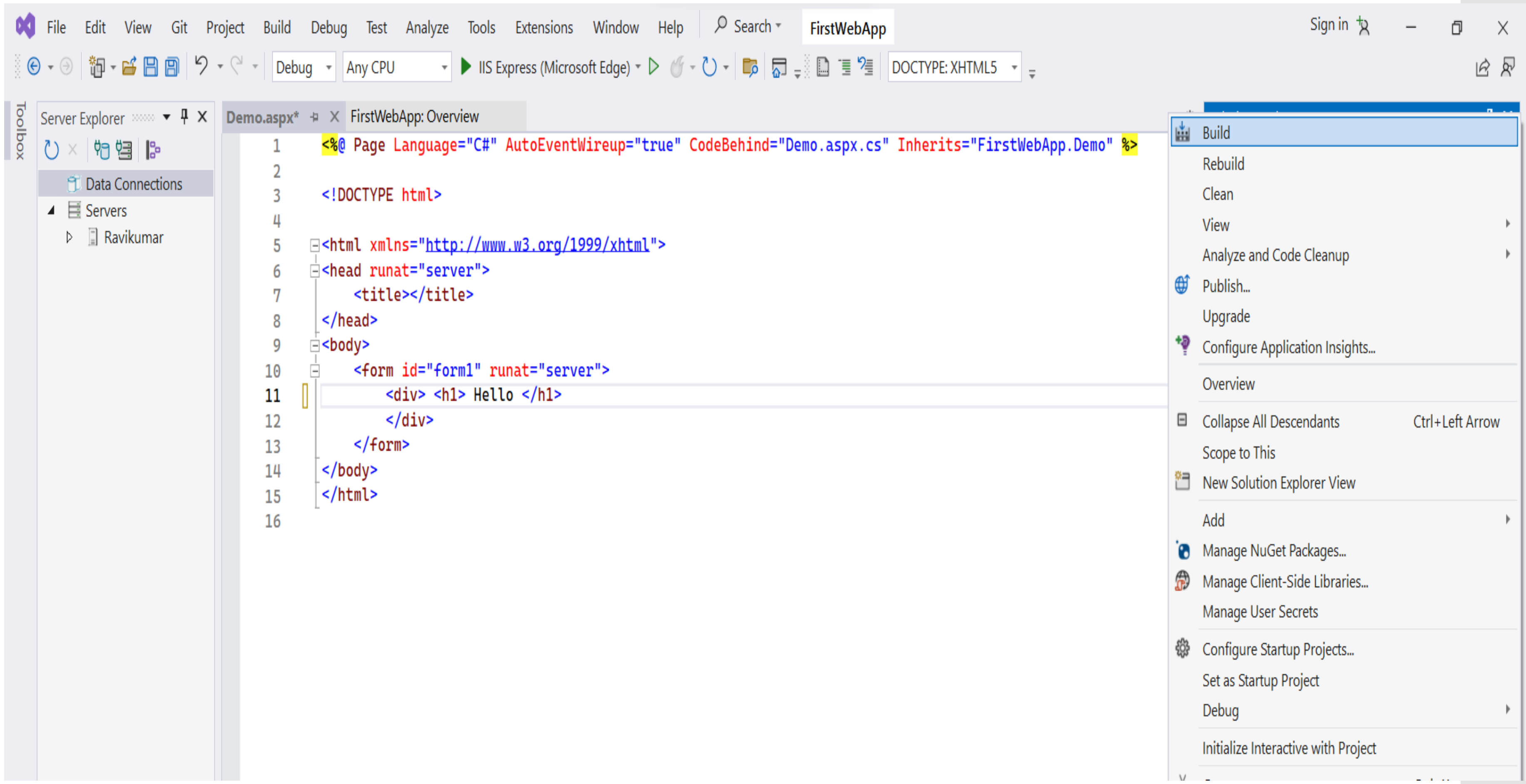
First Web Application



Demo.aspx*

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

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div> <h1> Hello </h1>
        </div>
    </form>
</body>
</html>
```

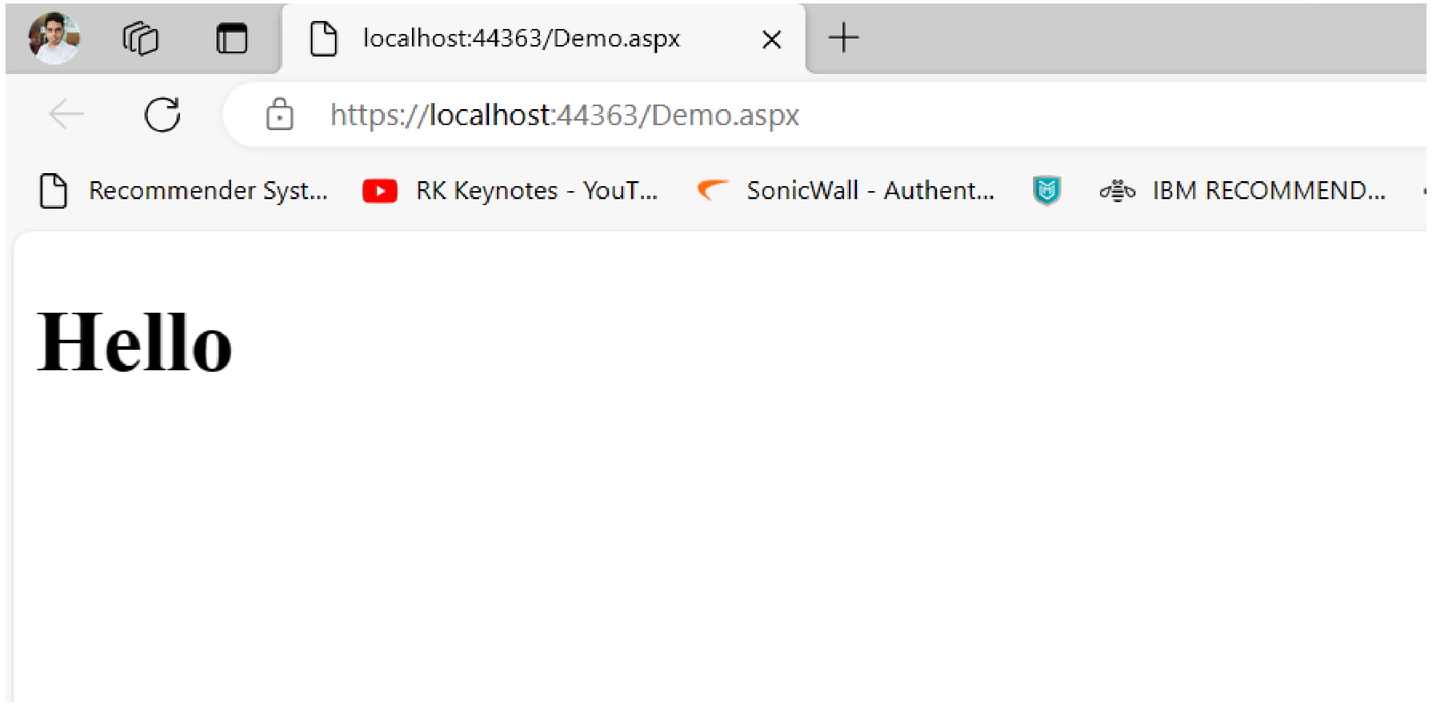
First Web Application



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

<!DOCTYPE html>

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
    <form id="form1" runat="server">
        <div> <h1> Hello </h1>
        </div>
    </form>
</body>
</html>
```

First Web Application

- Set as start page
- Build and Run



Hello

What is IIS?

- Internet Information Services (IIS) is a flexible, general-purpose web server from Microsoft that runs on Windows systems to serve requested HTML pages or files. Latest version 10.0 released in 2018.
- An **IIS web server** accepts **requests** from remote client computers and returns the appropriate **response**.
- It enables web servers to deliver, transfer information across an array of Local Area Networks (LANs) including corporate intranets.
- It has the ability to share information with the user in numerous forms such as text documents, HTML webpages, and images.

Web Controls

Server Controls
Label
TextBox
Button
LinkButton
ImageButton
Hyperlink
DropDownList
ListBox
DataGrid
DataList
CheckBox
CheckBoxList
RadioButton

RadioButtonList
Image
Panel
PlaceHolder
Calendar
AdRotator
Table
XML
Literal

Label

- This control is used to display textual information on the web forms. It is mainly used to create caption for the other controls like: textbox.
- To create label either we can write code or use the drag and drop facility of visual studio.

```
<asp:Label ID="Label1" runat="server"
 Text="Label" > </asp:Label>
```

TextBox

- This is an input control which is used to take user input.

```
<asp:TextBox ID="TextBox1" runat="server">
</asp:TextBox>
```

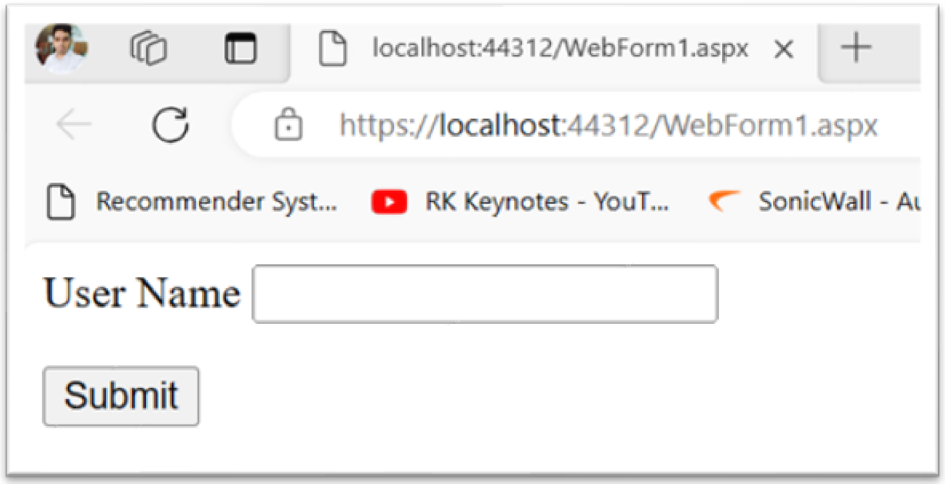
Label, TextBox and Button

```
<body>
    <form id="form1" runat="server">
        <div>
            <asp:Label ID="labelId" runat="server">User 
Name</asp:Label>


<asp:TextBox ID="UserName" runat="server" ToolTip="Enter 
User Name"></asp:TextBox>
        </div>
        <p>
        <asp:Button ID="SubmitButton" runat="server" 
Text="Submit" OnClick="SubmitButton_Click" />

        </p>
        <br />
    </form>
     <asp:Label ID="userInput" 
runat="server"></asp:Label>
</body>
```

Code Behind C#

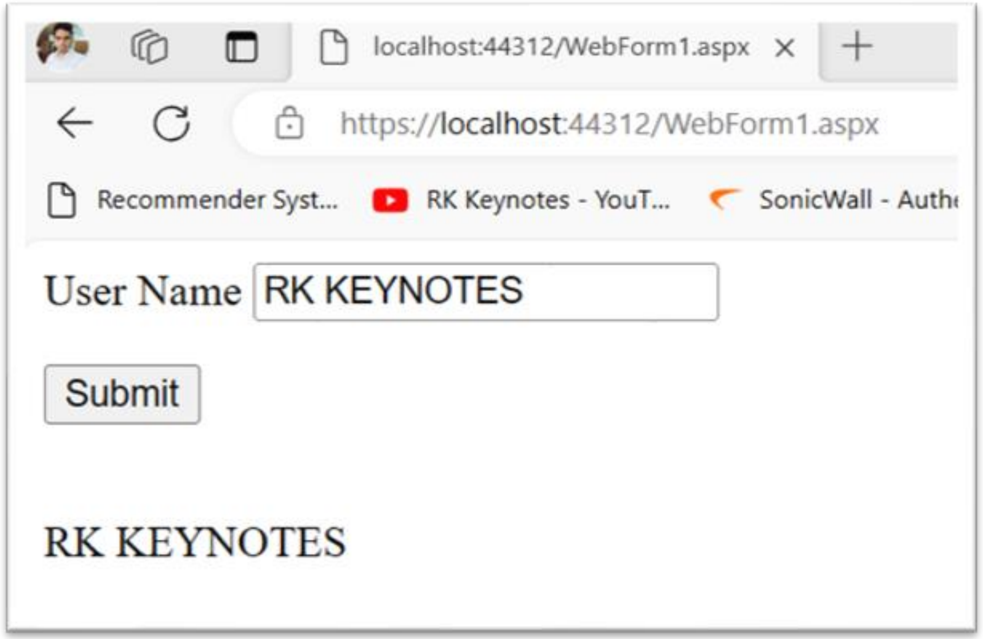
```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

namespace WebApplication6
{
    public partial class WebForm1 : System.Web.UI.Page
    {
        protected void Page_Load(object sender, EventArgs e)
        {

        }

        protected void SubmitButton_Click(object sender, 
EventArgs e)
        {
            userInput.Text = UserName.Text;
        }
    }
}
```

Output


localhost:44312/WebForm1.aspx
https://localhost:44312/WebForm1.aspx
Recommender Syst...
RK Keynotes - YouT...
SonicWall - Au

User Name

Submit


localhost:44312/WebForm1.aspx
https://localhost:44312/WebForm1.aspx
Recommender Syst...
RK Keynotes - YouT...
SonicWall - Authe

User Name RK KEYNOTES

Submit

RK KEYNOTES

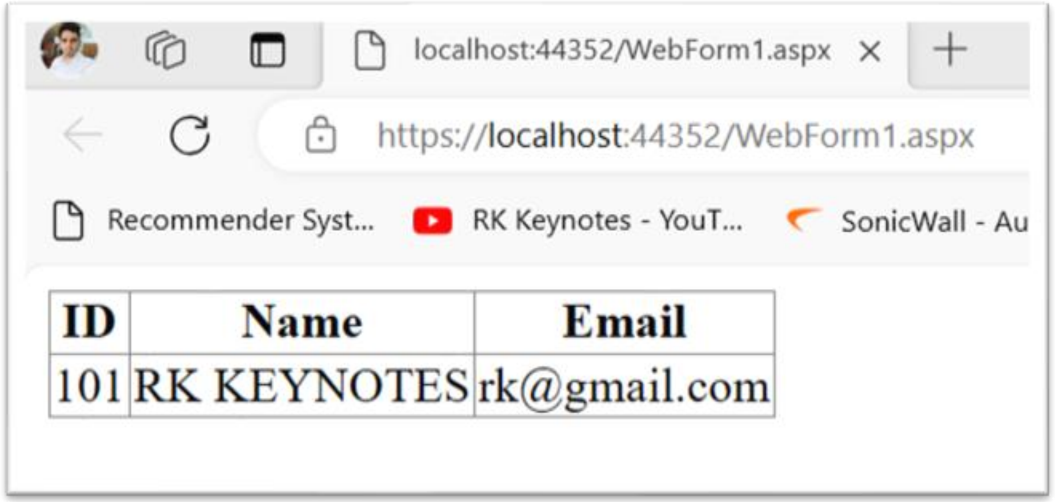
DataGrid

- DataGrid is used to display data in scrollable grid. It requires data source to populate data in the grid.
- It is a server side control and can be dragged from the toolbox to the web form.

```
<body>
    <form id="form1" runat="server">
        <div>
            <asp:GridView ID="GridView1"
runat="server"></asp:GridView>
        </div>
    </form>
</body>
```

DataGrid

```
protected void Page_Load(object sender, EventArgs e)
{
    DataTable table = new DataTable();
    table.Columns.Add("ID");
    table.Columns.Add("Name");
    table.Columns.Add("Email");
    table.Rows.Add("101", "RK KEYNOTES", "rk@gmail.com");
    GridView1.DataSource = table;
    GridView1.DataBind();
}
```

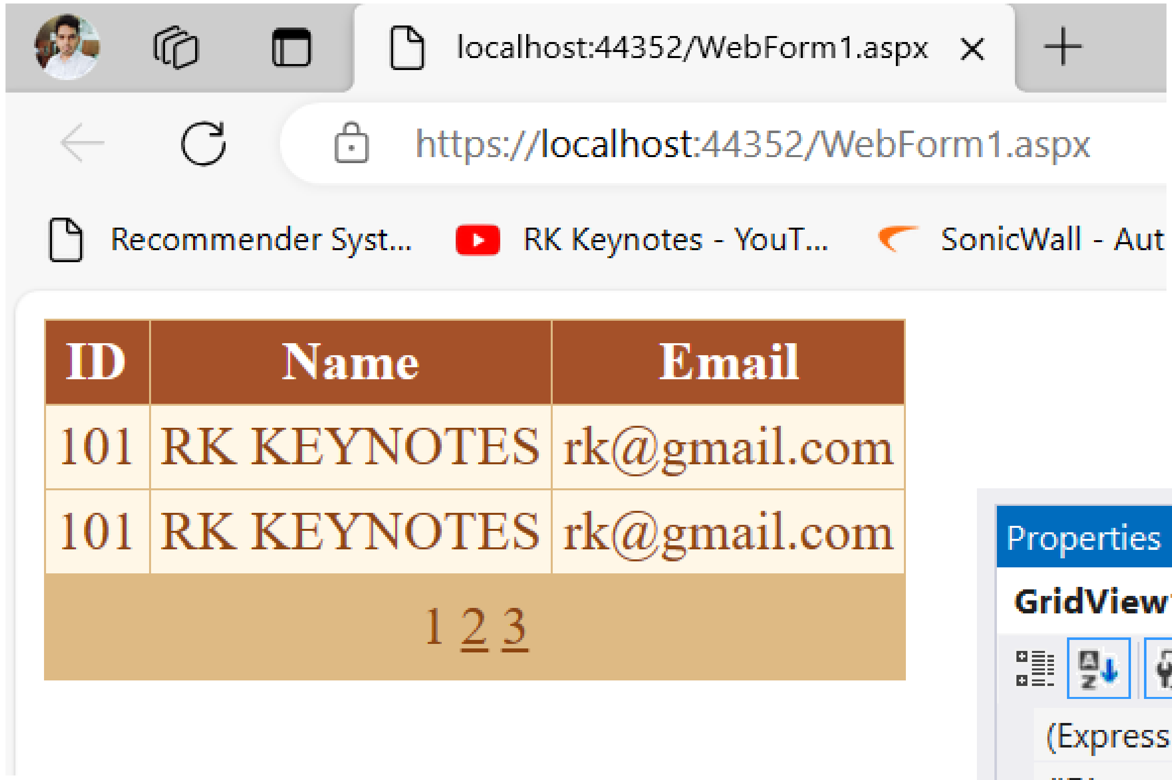

localhost:44352/WebForm1.aspx
https://localhost:44352/WebForm1.aspx
Recommender Syst...
RK Keynotes - YouT...
SonicWall - Au

ID	Name	Email
101	RK KEYNOTES	rk@gmail.com

GridView with Pagination

```
protected void Page_Load(object sender, EventArgs e){
    if (!IsPostBack)
        LoadGridData();
}
private void LoadGridData() {
    DataTable table = new DataTable();
    table.Columns.Add("ID");
    table.Columns.Add("Name");
    table.Columns.Add("Email");
    table.Rows.Add("101", "RK KEYNOTES", "rk@gmail.com"); //add 10 records
    GridView1.DataSource = table;
    GridView1.DataBind();
}
protected void GridView1_PageIndexChanging(object sender,
GridViewPageEventArgs e) {
    GridView1.PageIndex = e.NewPageIndex;
    LoadGridData();
}
```

GridView with Pagination



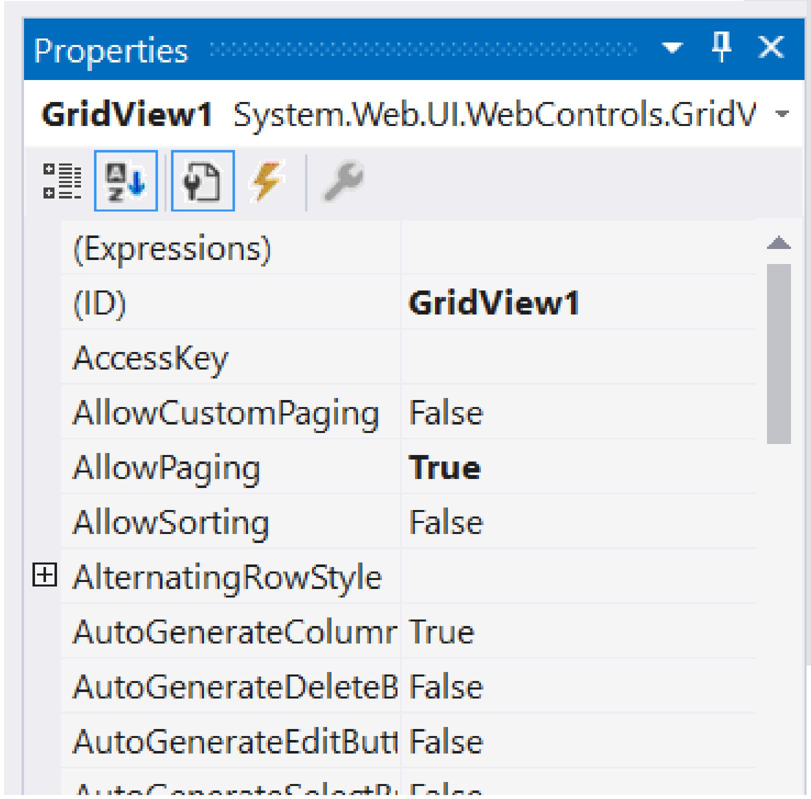
localhost:44352/WebForm1.aspx

https://localhost:44352/WebForm1.aspx

Recommender Syst... RK Keynotes - YouT... SonicWall - Aut

ID	Name	Email
101	RK KEYNOTES	rk@gmail.com
101	RK KEYNOTES	rk@gmail.com

1 2 3



Properties

GridView1 System.Web.UI.WebControls.GridV

(Expressions)	
(ID)	GridView1
AccessKey	
AllowCustomPaging	False
AllowPaging	True
AllowSorting	False
AlternatingRowStyle	
AutoGenerateColumr	True
AutoGenerateDeleteB	False
AutoGenerateEditButt	False

Calendar

- It is used to display selectable date in a calendar. It also shows data associated with specific date. This control displays a calendar through which users can move to any day in any year.
- We can also set Selected Date property that shows specified date in the calendar.

```
<asp:Calendar ID="Calendar1" runat="server" OnSelectionChanged
="Calendar1_SelectionChanged"></asp:Calendar>
```

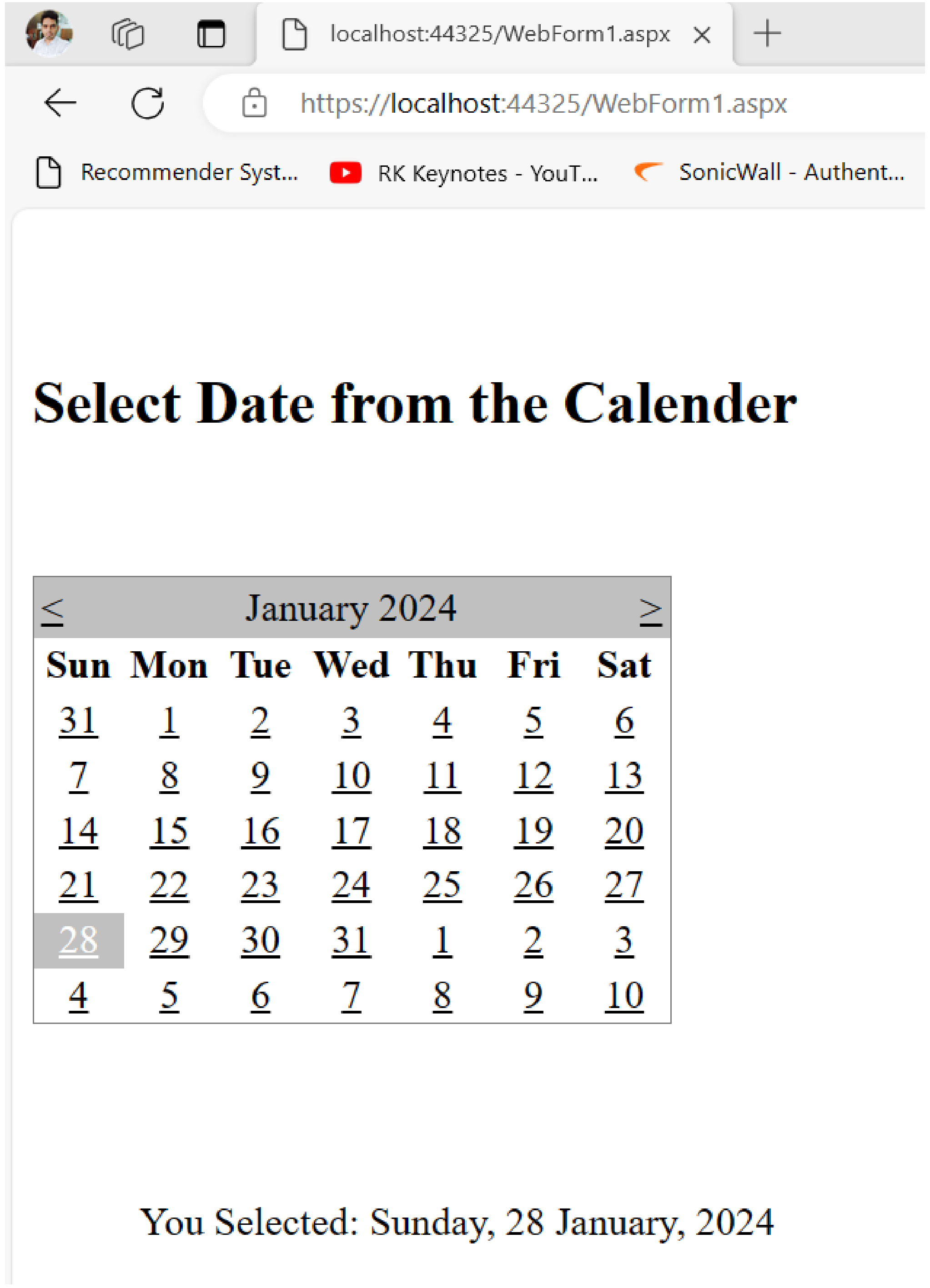
Calendar

```
<body>
    <form id="form1" runat="server">
        <h2>Select Date from the Calender</h2>
        <div>
            <asp:Calendar ID="Calendar1" runat="server"
            OnSelectionChanged="Calendar1_SelectionChanged">
             </asp:Calendar>
        </div>
    </form>
    <p>
        <asp:Label runat="server" ID="ShowDate" ></asp:Label>
    </p>
</body>
```

Calendar

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WebFormsControlls
{
    public partial class WebControls : System.Web.UI.Page
    {
        public void Calendar1_SelectionChanged(object sender, EventArgs e)
        {
            ShowDate.Text = "You Selected: "+Calendar1.SelectedDate.ToString();
        }
    }
}
```

Calendar



localhost:44325/WebForm1.aspx
https://localhost:44325/WebForm1.aspx
Recommender Syst...
RK Keynotes - YouT...
SonicWall - Authent...

Select Date from the Calender

<			January 2024			>
Sun	**Mon**	**Tue**	**Wed**	**Thu**	**Fri**	**Sat**
31	1	2	3	4	5	6
7	8	9	10	11	12	13
14	15	16	17	18	19	20
21	22	23	24	25	26	27
28	29	30	31	1	2	3
4	5	6	7	8	9	10

You Selected: Sunday, 28 January, 2024

File Upload

- It is an input controller which is used to upload file to the server. It creates a browse button on the form that pop up a window to select the file from the local machine.

```
<input name="FileUpload1" id="FileUpload1" type="file">
```

File Upload

```
<body>
    <form id="form1" runat="server">
        <div>
            <p>Browse to Upload File</p>
            <asp:FileUpload ID="FileUpload1" runat="server" />
        </div>
        <p>
<asp:Button ID="Button1" runat="server" Text="Upload File"
OnClick="Button1_Click" />
        </p>
    </form>
    <p>
        <asp:Label runat="server" ID="FileUploadStatus"></asp:Label>
    </p>
</body>
```

File Upload

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;
namespace WebFormsControlls  {
    public partial class WebControls : System.Web.UI.Page  {
        protected System.Web.UI.HtmlControls.HtmlInputFile File1;
        protected System.Web.UI.HtmlControls.HtmlInputButton Submit1;
        protected void Page_Load(object sender, EventArgs e)  {  }
        protected void Button1_Click(object sender, EventArgs e)
        {
            if ((FileUpload1.PostedFile != null) && (FileUpload1.PostedFile.ContentLength > 0))
            {
                string fn = System.IO.Path.GetFileName(FileUpload1.PostedFile.FileName);
                string SaveLocation = Server.MapPath("upload")  + "\\" + fn;
                try
                {
                    FileUpload1.PostedFile.SaveAs(SaveLocation);
                    FileUploadStatus.Text  = "The file has been uploaded.";
                }
                catch (Exception ex)
                {                   FileUploadStatus.Text  = "Error: " + ex.Message;    }            }
            else  {                 FileUploadStatus.Text = "Please select a file to upload.";  }         }  }
}
```

Demo

Download File

- ASP.NET provides implicit object Response and its methods to download file from the server. We can use these methods in our application to add a feature of downloading file from the server to the local machine.

```
<body>
    <form id="form1" runat="server">
<div><p>Click the button to download a file</p>

<asp:Button ID="Button1" runat="server"
Text="Download" OnClick="Button_click" />

<asp:Label ID="Label1" runat="server"
Text="Label"></asp:Label>
        </div>
    </form>
</body>
```

Download File

```
protected void Button_click(object sender, EventArgs e)
{
string filePath =
"C:\\Users\\ravik.RAVIKUMAR\\OneDrive\\Desktop\\OpenAI API.txt";
FileInfo file = new FileInfo(filePath);
if (file.Exists)
{
Response.Clear();
Response.AddHeader("Content-Disposition", "attachment;
filename=" + file.Name);
Response.AddHeader("Content-Length", file.Length.ToString());
Response.ContentType = "text/plain";
Response.Flush();
Response.TransmitFile(file.FullName);
Response.End();
}
else Label1.Text = "Requested file is not available to
download";
        }
```

Demo

Image and ImageButton

The image button is used to display a clickable image,and a control that displays an image and responds to mouse clicks on the image.

```
<asp:ImageButton ID="ImageButton1"
runat="server"
Height="268px"
ImageUrl="~/WhatsApp Image 2024-01-06 at
8.53.06 PM (1).jpeg"
PostBackUrl="~/WebForm2.aspx"
Width="357px" />
```

Image and ImageButton

```
<asp:Image ID="Image1" runat="server"
Height="162px" ImageUrl="~/WhatsApp
Image 2024-01-06 at 8.53.06 PM
(1).jpeg" Width="226px" />
```

DEMO

RadioButton

- It is an input control which is used to takes input from the user. It allows user to select a choice from the group of choices.

-
  ```
  <asp:RadioButton ID="RadioButton1"
  runat="server" Text="Male" />
  ```

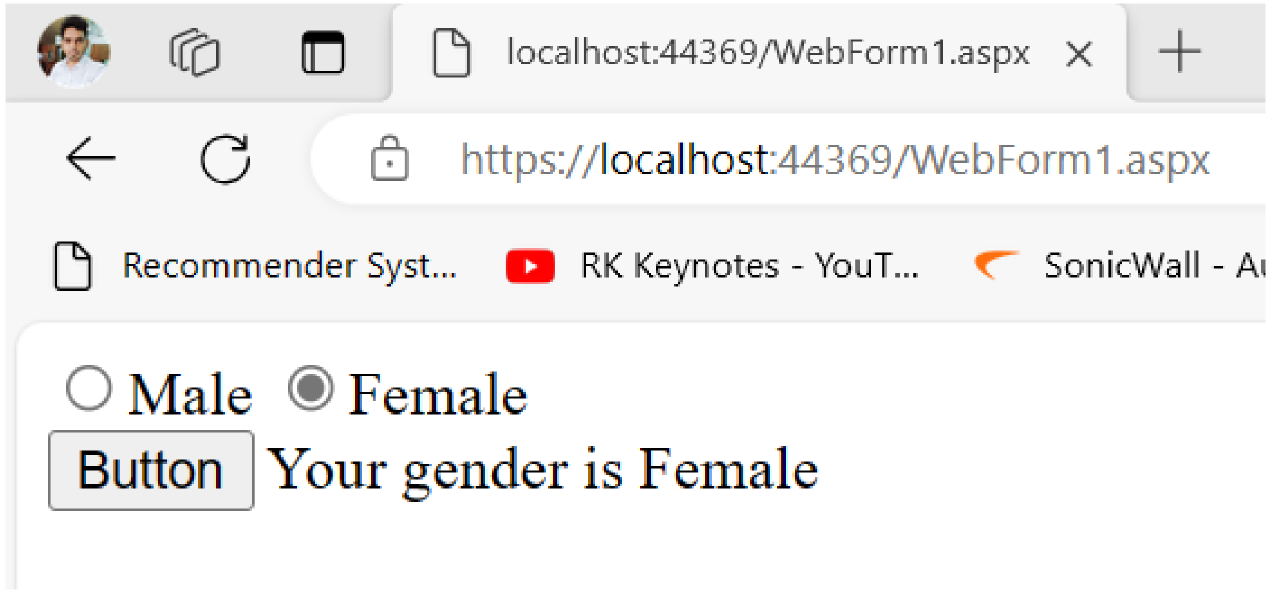
RadioButton

```
<form id="form1" runat="server">
    <div>
        <asp:RadioButton ID="RadioButton1" runat="server"
Text="Male" GroupName="gender" />
        <asp:RadioButton ID="RadioButton2" runat="server"
Text="Female" GroupName="gender" />
    </div>
    <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />
    <asp:Label ID="Label1" runat="server"
Text="Label"></asp:Label>
</form>
```

RadioButton

```
 protected void Button1_Click(object sender, EventArgs e)
 {
     Label1.Text = "";
     if (RadioButton1.Checked)
     {
      Label1.Text = "Your gender is " + RadioButton1.Text;
     }
     else
     Label1.Text = "Your gender is " + RadioButton2.Text;
 }
```

RadioButton


localhost:44369/WebForm1.aspx
https://localhost:44369/WebForm1.aspx
Recommender Syst...
RK Keynotes - YouT...
SonicWall - A

○ Male [✓ Female]

Button Your gender is Female

CheckBox

- It is used to get multiple inputs from the user. It allows user to select choices from the set of choices.
- It takes user input in yes or no format. It is useful when we want multiple choices from the user.
- `< asp:CheckBox ID="CheckBox2" runat="server" Text="J2EE"/>`

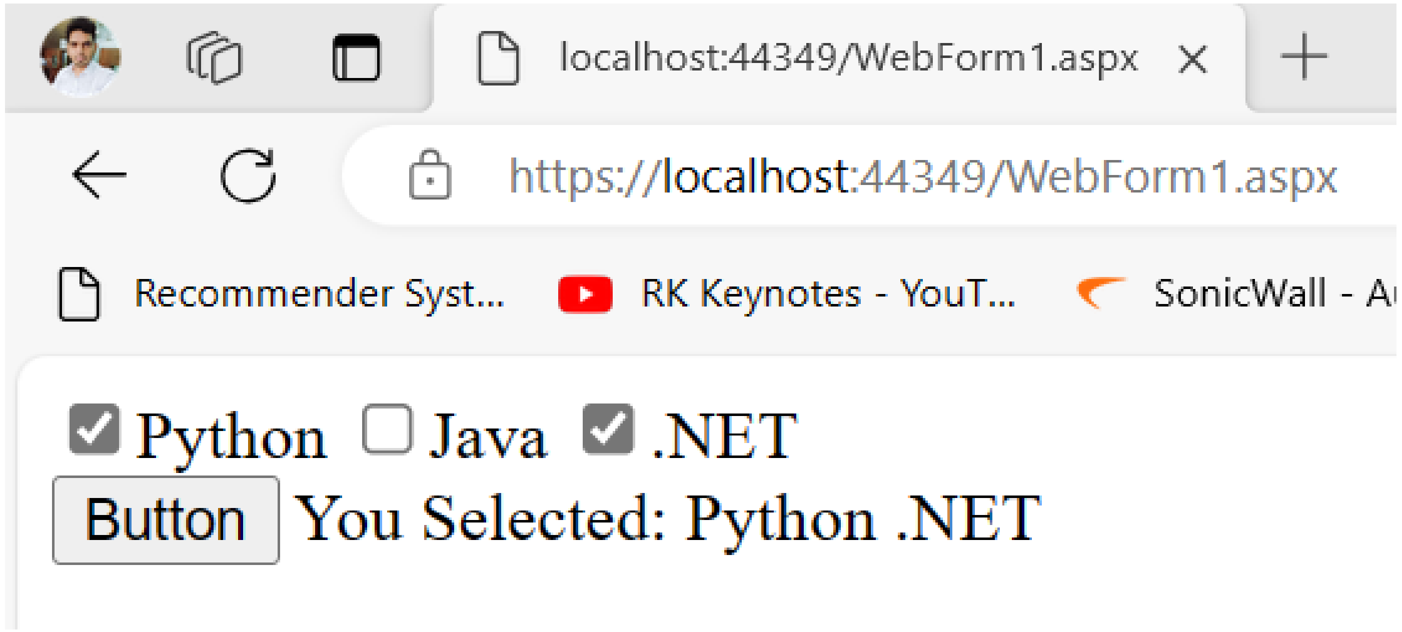
CheckBox

```
 <form id="form1" runat="server">
     <div>
         <asp:CheckBox ID="CheckBox1" runat="server" Text="Python"/>
         <asp:CheckBox ID="CheckBox2" runat="server" Text="Java"/>
         <asp:CheckBox ID="CheckBox3" runat="server" Text=".NET"/>


     </div>
     <asp:Button ID="Button1" runat="server" Text="Button"
OnClick="Button1_Click" />
     You Selected: <asp:Label ID="Label1" runat="server"
Text="Label"></asp:Label>
 </form>
```

CheckBox

```
protected void Page_Load(object sender, EventArgs e)
    {     Label1.Text = "";
    }
    protected void Button1_Click(object sender, EventArgs e) {
        var message = "";
        if (CheckBox1.Checked)
        {             message = CheckBox1.Text + " ";
        }
        if (CheckBox2.Checked)
        {            message += CheckBox2.Text + " ";
        }
        if (CheckBox3.Checked)
        {            message += CheckBox3.Text;
        }
        Label1.Text = message;
    } }
```

CheckBox



localhost:44349/WebForm1.aspx

https://localhost:44349/WebForm1.aspx

Recommender Syst... RK Keynotes - YouT... SonicWall - A

[✓] Python [ ] Java [✓] .NET

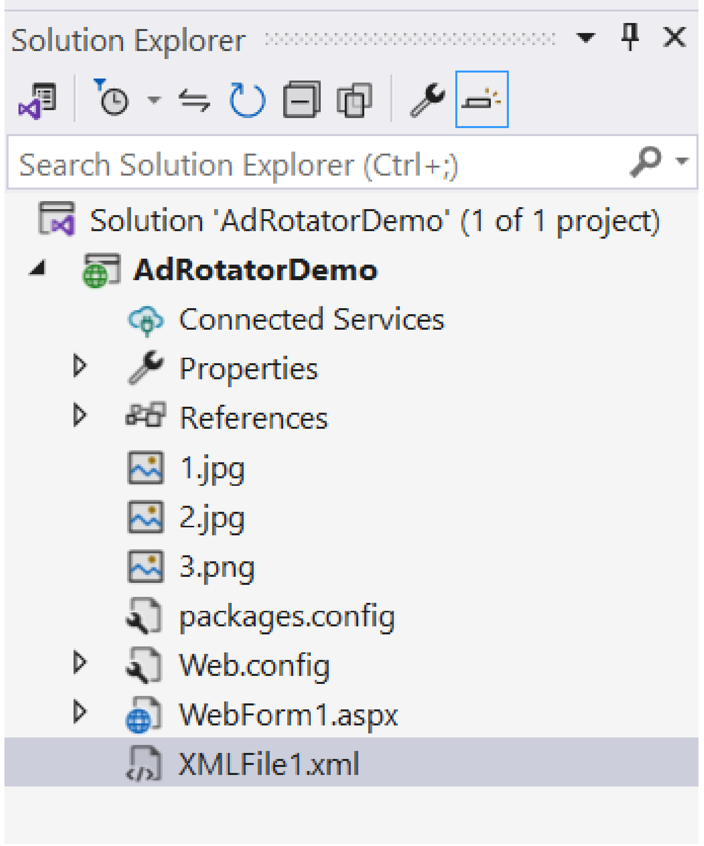
Button You Selected: Python .NET

AdRotator

- The AdRotator is one of the rich web server control of asp.net. AdRotator control is used to display a sequence of advertisement images as per given priority of image.
- AdRotator control displays the sequence of images, which is specified in the external XML file. In a xml file we indicate the images to display with some other attributes, like image impressions, NavigateUrl, ImageUrl, AlternateText.
- In a Adrotator control images will be changed each time while refreshing the web page.

```
<asp:AdRotator ID="AdRotator1" runat="server" />
```

AdRotator


Solution Explorer

Search Solution Explorer (Ctrl+;)

```
Solution 'AdRotatorDemo' (1 of 1 project)
◢ AdRotatorDemo
    Connected Services
  ▷ Properties
  ▷ References
    1.jpg
    2.jpg
    3.png
    packages.config
  ▷ Web.config
  ▷ WebForm1.aspx
    XMLFile1.xml
```

```
<form id="form1" runat="server">
     <div>
         <asp:AdRotator ID="AdRotator1" runat="server"
AdvertisementFile="~/XMLFile1.xml" />
     </div>
 </form>
```

AdRotator

```
<?xml version="1.0" encoding="utf-8" ?>
<Advertisements>
    <Ad>
    <ImageUrl>1.jpg</ImageUrl>
    <NavigateUrl>https://rkkeynotes.blogspot.com</NavigateUrl>
    <AlternateText>RK</AlternateText>
    <Impressions>50</Impressions>
    <Keyword>RK</Keyword>
    </Ad>
</Advertisements>
```

AdRotator

- **ImageUrl:** The URL of the image that will be displayed through AdRotator control.
- **NavigateUrl:** If the user clicks the banner or ad then the new page is opened according to given URL.
- **AlternateText:** It is used for displaying text instead of the picture if picture is not displayed. It is also used as a tooltip.
- **Impressions:** It is a number that sets how frequently an advertisement will appear.
- **Keyword:** It is used to filter ads or identifies a group of advertisement.

Validation Controls

- Validation controls are an essential part of ASP.NET web development because they help ensure the integrity of user input.
- When users enter information into web forms, there is always the potential for intentional or unintentional errors.
- Validation controls provide a way to check the accuracy and validity of user input before the web application processes it.
- Why validation controls are important in ASP.NET?
- Preventing invalid data from being submitted.
- Improving user experience
- Enhancing security
- Business requirements

Validation Controls

- ASP.NET supports two types of validation controls:
- **Client-side validation controls**
- **Server-side validation controls**
- Client-side validation is good, but we must depend on browser and scripting language support. The client-side validation is done in the user's browser using JavaScript and another scripting. You can use client-side validation libraries such as WebUIValidation.js in .NET.
- Server-side validation in ASP.NET is done in the **C# code-behind**, where the value of the user input is read and validated on the server. This process is time-consuming but provides better security and is easier to implement in ASP.NET.

Validation Controls

- ASP.NET provides several types of validation controls that can be used to validate user input in web forms. Some of the common validation controls are:
- **RequiredFieldValidation Control**
- **CompareValidator Control**
- **RangeValidator Control**
- **RegularExpressionValidator Control**
- **CustomValidator Control**
- **ValidationSummary**

Validation Controls

Validation Control	Description
RequiredField Validation	This control ensures that a field is not left empty or blank. It can be used for textboxes, dropdown lists, checkboxes, and other input controls.
Compare Validator	This control compares the value of one input control to another. It can validate passwords, confirm email addresses, and other scenarios where two values must match.
RangeValidator	This control checks if a value falls within a specific range. For example, it can be used to validate a user's age, income, or date of birth.
Regular Expression Validator	This control checks if a value matches a specified regular expression pattern. For example, it can validate email addresses, phone numbers, zip codes, and other input types.
Custom Validator	This control allows developers to define their validation logic. It usually depends on the business rules.
Validation Summary	This control displays a report of all validation errors that occurred on a Web page.

Error

- <appSettings>
- <add key="ValidationSettings:UnobtrusiveValidationMode" value="None" />
- </appSettings>

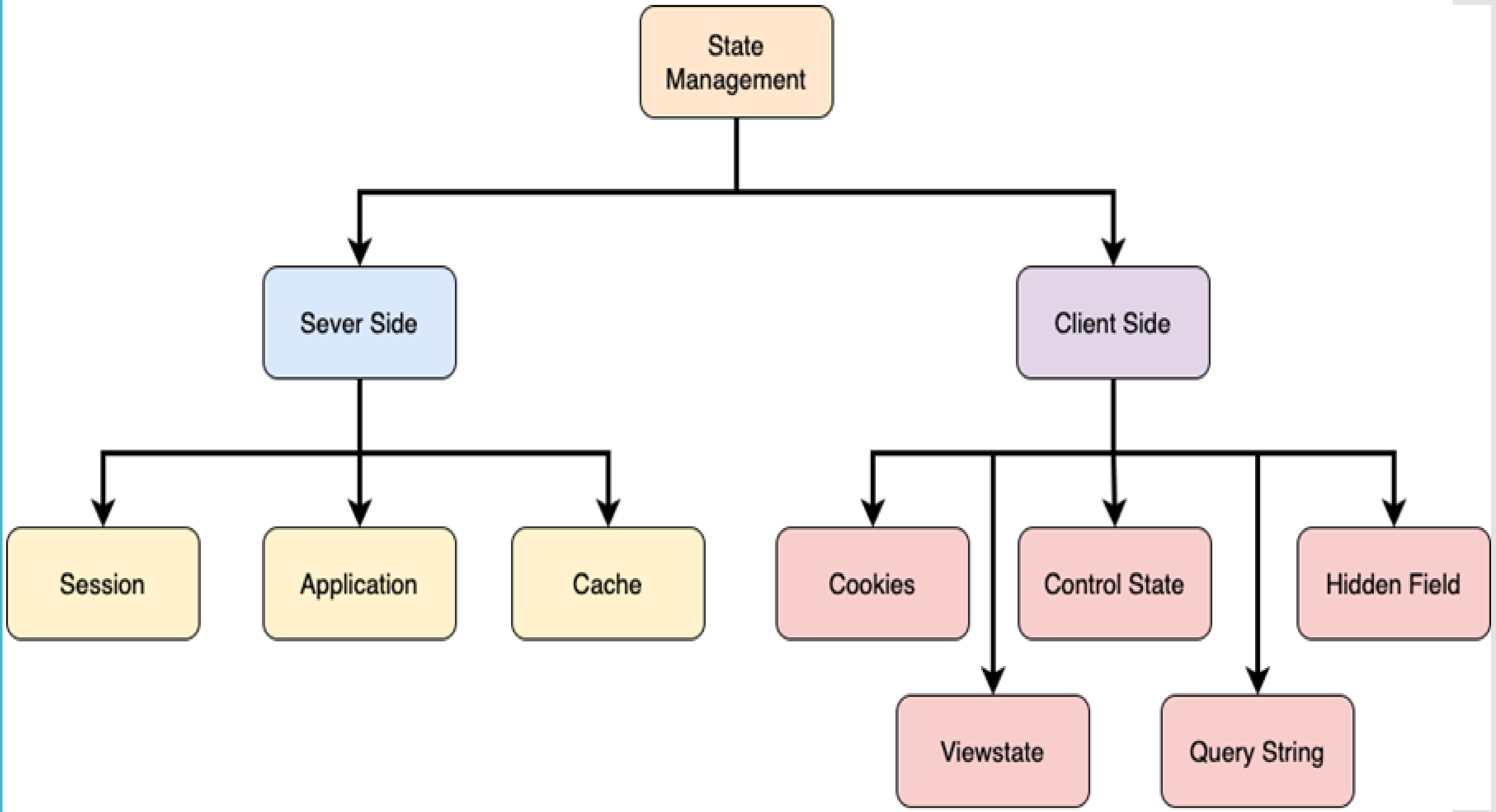
- Unobtrusive Validation means without writing a lot of validation code, you can perform simple client-side validation.

Demo

State Management

- State management is a process by which you maintain the state of an object or variable throughout the lifetime of a page.
- In ASP.NET, there are two types of state management: client-side state management and server-side state management.
- Client-side state management refers to the technique of storing data on the client's browser in the form of either cookies or hidden fields.
- Server-side state management, on the other hand, stores data on the server in the form of either application state or session state.
- Application state is a global storage mechanism that is used to store data that needs to be available to all users of an ASP.NET application.
- Session state, on the other hand, is a per-user storage mechanism that is used to store data that is specific to a user's session.

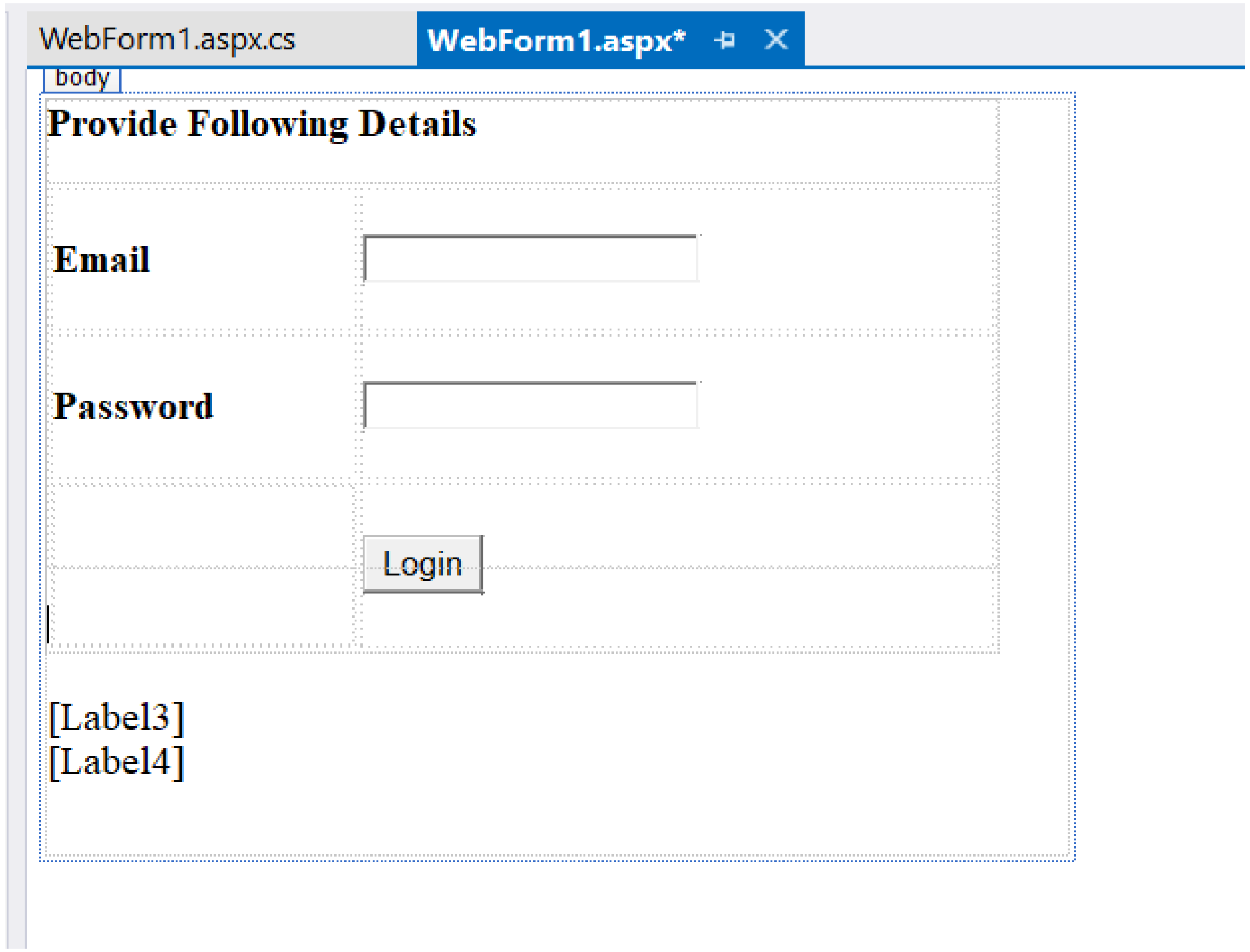
Types of State Management



Sessions

- In ASP.NET session is a state that is used to store and retrieve values of a user.
- It helps to identify requests from the same browser during a time period (session).
- It is used to store value for the particular time session. **By default, ASP.NET session state is enabled for all ASP.NET applications.**
- Each created session is stored in SessionStateItemCollection object.
- We can get current session value by using Session property of Page object.

Form Design



Provide Following Details

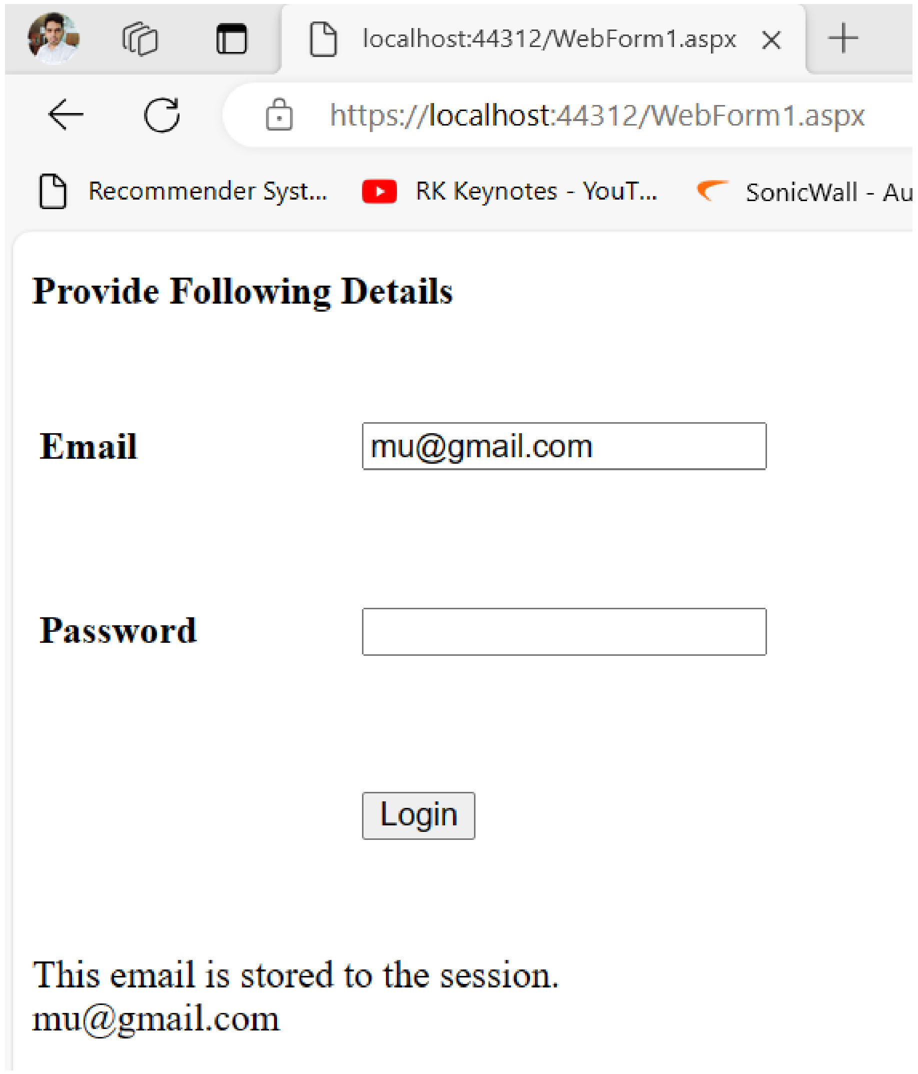
Email	
Password	
	Login

[Label3]
[Label4]

Code Behind

```
protected void login_Click(object sender, EventArgs e)  {
    if (password.Text == "123"){
        // Storing email to Session variable
        Session["email"] = email.Text;     }
    else {
        Label3.Text = "Wrong Password";
        Label4.Text = "";
    }
    // Checking Session variable is not empty
    if ((Session["email"] != null) && (password.Text=="123")){
        // Displaying stored email
        Label3.Text = "This email is stored to the session.";
        Label4.Text = Session["email"].ToString();
    }
    else    {
        Label3.Text = "Wrong Password";
        Label4.Text = "";  }}
```

Output


localhost:44312/WebForm1.aspx
https://localhost:44312/WebForm1.aspx
Recommender Syst...
RK Keynotes - YouT...
SonicWall - Au

Provide Following Details

Email	mu@gmail.com
Password	
	Login

This email is stored to the session.
mu@gmail.com

Cookie

- ASP.NET Cookie is a small bit of text that is used to store user-specific information. This information can be read by the web application whenever user visits the site.
- **When a user requests for a web page, web server sends not just a page, but also a cookie containing the date and time.** This cookie stores in a folder on the user's hard disk.
- When the user requests for the web page again, browser looks on the hard drive for the cookie associated with the web page. Browser stores separate cookie for each different sites user visited.
- Note: The Cookie is limited to small size and can be used to store only 4 KB (4096 Bytes) text.

Type of Cookies

- Persist Cookie - A cookie that doesn't have expired time is called a Persist Cookie
- Non-Persist Cookie - A cookie which has expired time is called a Non-Persist Cookie
- **How to create a cookie?**
- `HttpCookie userInfo = new HttpCookie("userInfo");`

-
  ```
  userInfo["UserName"] = "Annathurai";
  userInfo["UserColor"] = "Black";
  ```
- `Response.Cookies.Add(userInfo);`

How to retrieve from cookie?

- `string User_Name = string.Empty;`
- `string User_Color = string.Empty;`

-
  ```
  User_Name = Request.Cookies["userName"].Value;
  User_Color = Request.Cookies["userColor"].Value;
  ```

- How to clear the cookie information?
- `userInfo.Expires = DateTime.Now.AddHours(1);`

Advantages and Disadvantages

- **Advantages of Cookie**
- It has clear text so the user can read it.
- We can store user preference information on the client machine.
- It is an easy way to maintain.
- Fast accessing.
- **Disadvantages of Cookie**
- If the user clears the cookie information, we can't get it back.
- No security.
- Each request will have cookie information with page.

Form

```
<form id="form1" runat="server">
<div>
<asp:Label ID="Label1" runat="server" Text="Label"></asp:Label>
</div>
</form>
```

Code Behind

```
protected void Page_Load(object sender, EventArgs e)
{
    //-- Creating Cookie --//
    HttpCookie cookie = new HttpCookie("student");

    // Assigning value to the created cookie
    cookie.Value = "Kumar";

    // Adding Cookie to the response instance
    Response.Cookies.Add(cookie);

    //-- Fetching Cookie --//
    var cookievalue = Response.Cookies["student"].Value;
    Label1.Text = cookievalue;
}
```

Demo

Programming Activity

Summary

- ASP.Net Web Application
- Page life cycle of ASP.NET Application
- Web Controls (Button, TextBox, CheckBox, Image etc.)
- Rich Controls (Calendar, AdRotator)
- Validation Controls
- State management
- Cookie
- Session

Up Next

- **ASP.Net MVC:**
- Introduction to ASP.NET MVC
- MVC Architecture Overview
- Controllers
- Razor Views
- LayoutView
- PartialView
- Models
- HTML helpers
- Action Filters
- Model Validation
- URLs and
- Routing

End of Unit 2