9. Create the ASP.Net Web Application that accepts Name, Password, age, email id, and userid. All the information entry is mandatory. Password should be reconfirmed; age should be within 21- 30. Email id should be valid. User id should have atleast a capital letter and digit as well as length should be between 7 and 20 characters. Use all Validation Controls.

Step 1: Create an ASP.NET Web Forms Project

1. **Open Visual Studio → Create New Project → ASP.NET Web Application (.NET Framework) → Name it UserRegistrationApp.**
2. **Choose Web Forms template.**

Step 2: Open Default.aspx in Designer

- **Switch to Design view (not Source view) so you can drag controls easily.**

Step 3: Add Controls Using Drag-and-Drop

Field	Control Type	Properties / Notes
Name	TextBox	ID = txtName
	RequiredFieldValidator	ControlToValidate=txtName, ErrorMessage=Name is required
Password	TextBox	ID = txtPassword, TextMode=Password
	RequiredFieldValidator	ControlToValidate=txtPassword, ErrorMessage=Password is required
Confirm Password	TextBox	ID = txtConfirmPassword, TextMode=Password
	RequiredFieldValidator	ControlToValidate=txtConfirmPassword, ErrorMessage=Confirm Password required
	CompareValidator	ControlToValidate=txtConfirmPassword, ControlToCompare=txtPassword, ErrorMessage=Passwords do not match
Age	TextBox	ID = txtAge
	RequiredFieldValidator	ControlToValidate=txtAge, ErrorMessage=Age required
	RangeValidator	ControlToValidate=txtAge, MinimumValue=21, MaximumValue=30, Type=Integer, ErrorMessage=Age must be between 21-30
Email ID	TextBox	ID = txtEmail
	RequiredFieldValidator	ControlToValidate=txtEmail, ErrorMessage=Email required
	RegularExpressionValidator	ControlToValidate=txtEmail, ValidationExpression=\w+([-+.’]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*, ErrorMessage=Invalid Email
User ID	TextBox	ID = txtUserID
	RequiredFieldValidator	ControlToValidate=txtUserID, ErrorMessage=UserID required
	RegularExpressionValidator	ControlToValidate=txtUserID, ValidationExpression=^(?=.*[A-Z])(?=.*\d).{7,20}$, ErrorMessage=UserID must have at least 1 capital letter, 1 digit, 7-20 chars
Submit	Button	Text=Register, ID=btnSubmit

Step 4: Configure Validation Controls

1. Drag the RequiredFieldValidator, CompareValidator, RangeValidator, and RegularExpressionValidator next to the corresponding TextBoxes.
2. Set ControlToValidate to the TextBox you want to validate.
3. Set ErrorMessage to show an appropriate message.
4. For RegularExpressionValidator for email and UserID, copy the following expressions:

- Email Regex:

\w+([-+.’]\w+)*@\w+([-.]\w+)*\.\w+([-.]\w+)*

- UserID Regex:

^(?=.*[A-Z])(?=.*\d).{7,20}$

Step 5: Handle Submit Button Click (Optional)

Double-click the **Register** button to create btnSubmit_Click in Default.aspx.cs:

```
protected void btnSubmit_Click(object sender, EventArgs e)
{
   if (Page.IsValid)
   {
      lblMessage.Text = "Registration Successful!";
      lblMessage.ForeColor = System.Drawing.Color.Green;
   }
}
```

- Add a **Label** (lblMessage) to show success messages.

This setup ensures:

- All fields are mandatory.
- Passwords must match.
- Age is between 21–30.
- Email is valid.
- UserID contains at least one uppercase letter and one digit, and is 7–20 characters long.

7. Sending Mail with SmtpMail : Use a simple web form to demonstrate how to use the SmtpMail class in the .Net Framework.

Step 1: Create ASP.NET Web Application

1. Open Visual Studio → Create New Project → ASP.NET Web Application (.NET Framework)
2. Name it: SendMailApp → Choose Empty template → Check Web Forms

Step 2: Design the Web Form

Open Default.aspx in **Design View** and drag the following controls from Toolbox:

Control	Name	Properties / Notes
Label	lblTo	Text = "To:"
TextBox	txtTo	Width = 300
Label	lblSubject	Text = "Subject:"
TextBox	txtSubject	Width = 300
Label	lblBody	Text = "Body:"
TextBox	txtBody	Width = 300, Height = 100, TextMode = MultiLine
Button	btnSend	Text = "Send Email"
Label	lblMessage	Text = "", ForeColor = Red

Step 3: Add Backend Code in Default.aspx.cs

Double-click the **Send Email** button to create btnSend_Click:

```
using System;
using System.Net;
using System.Net.Mail;

namespace SendMailApp
{
   public partial class Default : System.Web.UI.Page
   {
      protected void btnSend_Click(object sender, EventArgs e)
      {
         try
         {
            // Create MailMessage object
            MailMessage mail = new MailMessage();
            mail.From = new MailAddress("your_email@example.com"); // Your
email
            mail.To.Add(txtTo.Text);
            mail.Subject = txtSubject.Text;
            mail.Body = txtBody.Text;

            // Configure SMTP client
            SmtpClient smtp = new SmtpClient("smtp.example.com"); // Replace
with your SMTP server
            smtp.Port = 587; // Typical SMTP port
            smtp.Credentials = new
NetworkCredential("your_email@example.com", "your_password");
            smtp.EnableSsl = true;

            // Send the email
            smtp.Send(mail);

            lblMessage.ForeColor = System.Drawing.Color.Green;
            lblMessage.Text = "Email sent successfully!";
          }
          catch (Exception ex)
          {
            lblMessage.ForeColor = System.Drawing.Color.Red;
            lblMessage.Text = "Error: " + ex.Message;
          }
      }
   }
}
```

Step 4: Run the Application

1. Enter **recipient email**, **subject**, and **body**.
2. Click **Send Email** → Email will be sent via your configured SMTP server.

11. Create Website Application for Student Management System with a master page which is linked to other web pages in the application.

Step 1: Create Website Project

1.Open **Visual Studio 2022** → **Create New Project** → **ASP.NET Web Application (.NET Framework)**

2.Name it: SMS1 → Select **Empty** template → Check **Web Forms**

Step 2: Add Master Page

1. Right-click project → **Add → New Item → Master Page** → Name it Site1.Master
2. Design the Master Page using **drag-and-drop**:

Master Page Controls

- **Header:**

```
<header><h1>CVR COLLEGE OF ENGINEERING</h1></header>
```

- **Menu:** Drag an **ASP:Menu**

```
<asp:Menu ID="Menu1" runat="server" Orientation="Horizontal">
   <Items>
  <asp:MenuItem NavigateUrl="~/HOME.aspx" Text="HOME" Value="HOME" />
      <asp:MenuItem NavigateUrl="~/ADDSTUDENT.aspx" Text="ADD"
Value="ADD" />
   </Items>
   <StaticMenuItemStyle Font-Bold="True" Font-Size="15pt"
HorizontalPadding="10px" ItemSpacing="5px" VerticalPadding="5px" />
</asp:Menu>
```

- **Image:** Drag **ASP:Image** → set ImageUrl="~/IMAGE/COLLEGE.jpg" and Width="50%"
- **Content Placeholder:** Drag **ASP:ContentPlaceHolder** → ID=ContentPlaceHolder1
- **Footer:** Use a <div> for footer text

```
<div id="footer" style="background-color: #eee; text-align: center; padding:
15px; font-size: 12px;">
   © 2025 Student Management System. All rights reserved.
</div>
```

Step 3: Add Content Pages

1. Right-click project → **Add → Web Form using Master Page**
2. Name it ADDSTUDENT.aspx → Select Site1.Master as Master Page

ADDSTUDENT.aspx Design (Drag-and-Drop)

- Drag **Labels** and **TextBoxes** for student info:

Label	**TextBox**
Student Name:	txtName
Age:	txtAge

- Add **Submit Button** (btnAddStudent) if needed

```
<h3>NEW STUDENT ADDING DETAILS</h3>
```

```
<asp:Label ID="Label1" runat="server" Text="Student Name: " />
<asp:TextBox ID="txtName" runat="server" /><br /><br />

<asp:Label ID="Label2" runat="server" Text="Age: " />
<asp:TextBox ID="txtAge" runat="server" /><br /><br />

<asp:Button ID="btnAddStudent" runat="server" Text="Add Student"
OnClick="btnAddStudent_Click" />
```

Step 4: Backend Code (C#)

In ADDSTUDENT.aspx.cs:

```
using System;
using System.Web.UI;

namespace SMS1
{
  public partial class WebForm1 : Page
  {
    protected void Page_Load(object sender, EventArgs e)
    {
    }

    protected void btnAddStudent_Click(object sender, EventArgs e)
    {
      string name = txtName.Text;
      int age = Convert.ToInt32(txtAge.Text);

      // Code to save student to database can be added here
      lblMessage.Text = $"Student {name} aged {age} added successfully!";
    }
  }
}
```

- Add a **Label** lblMessage to show success messages.

13. Use ADO.NET for storing and manipulating the data. Develop the necessary forms for the better user Interface.

1. Create Database Table

```
CREATE TABLE Students (
  StudentID INT IDENTITY(1,1) PRIMARY KEY,
  StudentName VARCHAR(50),
  Age INT
);
```

2. Add Connection String in Web.config

```
<connectionStrings>
  <add name="ConnString"
     connectionString="Data Source=.\SQLEXPRESS;Initial
Catalog=SMSDB;Integrated Security=True"
     providerName="System.Data.SqlClient" />
</connectionStrings>
```

3. Add Forms (Web Forms with Master Page)

Form	Purpose
AddStudent.aspx	Add new student
ViewStudent.aspx	Display all students
UpdateStudent.aspx	Update student info
DeleteStudent.aspx	Delete a student record

3. Add Controls (Drag-and-Drop)

AddStudent.aspx:

```
<asp:Label Text="Name:" /><asp:TextBox ID="txtName" runat="server" /><br />
<asp:Label Text="Age:" /><asp:TextBox ID="txtAge" runat="server" /><br />
<asp:Button ID="btnAdd" runat="server" Text="Add Student"
OnClick="btnAdd_Click" />
```

```
<asp:Label ID="lblMessage" runat="server" ForeColor="Green" />
```

ViewStudent.aspx:

```
<asp:GridView ID="GridView1" runat="server"
AutoGenerateColumns="True"></asp:GridView>
```

UpdateStudent.aspx / DeleteStudent.aspx:

- Labels, TextBoxes for student ID and details
- Buttons for Load, Update, Delete

5. Backend Code (ADO.NET)
AddStudent.aspx.cs

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

namespace SMS1
{
    public partial class AddStudent : System.Web.UI.Page
    {
        string connStr =
ConfigurationManager.ConnectionStrings["ConnString"].ConnectionString;

        protected void btnAdd_Click(object sender, EventArgs e)
        {
            using (SqlConnection con = new SqlConnection(connStr))
            {
                string query = "INSERT INTO Students (StudentName, Age) VALUES
(@Name, @Age)";
                SqlCommand cmd = new SqlCommand(query, con);
                cmd.Parameters.AddWithValue("@Name", txtName.Text);
                cmd.Parameters.AddWithValue("@Age", txtAge.Text);
                con.Open();
                cmd.ExecuteNonQuery();
                lblMessage.Text = "Student added successfully!";
            }
        }
    }
}
```

ViewStudent.aspx.cs

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Configuration;

namespace SMS1
{
    public partial class ViewStudent : System.Web.UI.Page
    {
        string connStr =
ConfigurationManager.ConnectionStrings["ConnString"].ConnectionString;

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

        private void BindGrid()
        {
            using (SqlConnection con = new SqlConnection(connStr))
            {
                SqlDataAdapter da = new SqlDataAdapter("SELECT * FROM Students",
con);
                DataTable dt = new DataTable();
                da.Fill(dt);
                GridView1.DataSource = dt;
                GridView1.DataBind();
            }
        }
    }
}
```

6. Result

- AddStudent.aspx: Adds new student

- ViewStudent.aspx: Shows all students in a GridView
- UpdateStudent.aspx / DeleteStudent.aspx: Update or remove records
- Uses ADO.NET for database connectivity
- Drag-and-drop controls provide a clean UI

14. Convert the above application to a web application using ASP.NET and SQL Server. Use IIS to deploy the web application developed in ASP.NET.

1. Create ASP.NET Web Application

1. Open Visual Studio → **Create New Project** → **ASP.NET Web Application (.NET Framework)**
2. Name: SMSWebApp → Empty template → Check **Web Forms**

2. Add Master Page

1. Right-click project → Add → **Master Page** → Site1.Master
2. **Drag-and-Drop Controls:**

Header: Label → Text: CVR COLLEGE OF ENGINEERING
Menu: ASP:Menu → Add MenuItems: Home, Add Student, View Students, Update Student, About
ContentPlaceHolder: Drag from Toolbox → ID=ContentPlaceHolder1
Footer: Label or Panel → Text: © 2025 SMS. All rights reserved.

3. Add 5 Content Pages
Right-click project → Add → Web Form using Master Page → select Site1.Master

Page Name	Drag-and-Drop Controls
Home.aspx	Label / Panel for welcome message
AddStudent.aspx	Labels, TextBoxes, Button (btnAdd)
ViewStudent.aspx	GridView → AutoGenerateColumns=True
UpdateStudent.aspx	Labels, TextBoxes, Buttons (Load, Update)
About.aspx	Label / Panel → About info

4. Add Navigation Controls on Master Page
Drag from Toolbox:
Menu Control → Add MenuItems for all pages
HyperLink → One for each page
TreeView → Connect to **SiteMapDataSource**
SiteMapPath → Drag onto Master Page

5. Add ADO.NET Controls for Database
1.Open **Server Explorer** → Connect to SQL Server
2.Drag **Students table** to page or use **GridView** + **SqlDataSource**
3.For forms like Add/Update/Delete:
Drag Labels, TextBoxes, Buttons
Use code-behind to handle Click events using **ADO.NET**

6. Set Connection String

- In **Web.config**, drag in the connection string:

```
<connectionStrings>
  <add name="ConnString" connectionString="Data
Source=.\SQLEXPRESS;Initial Catalog=SMSDB;Integrated Security=True"
providerName="System.Data.SqlClient" />
</connectionStrings>
```

7. Deploy to IIS (Drag-and-Drop Folder)

1. Right-click project → **Publish → Folder** → Choose folder like C:\inetpub\wwwroot\SMSWebApp
2. Open **IIS Manager** → Right-click **Sites** → Add Website

Site name: SMSWebApp
Physical path: Folder above
Port: 8080

3. Select **Application Pool** → .NET v4.0 → Integrated
4. Open browser → http://localhost:8080 → Test all pages