ASSIGNMENT II

NAME: SHOBIKA R

REG NO: 73772214206

ROLL NO: 43

COURSE NAME: **C# and .Net framework**

COURSE CODE: 60 IT L04

---

1. **Design and implement a Student Registration form using C# and Windows Forms. The form should allow users to input and save student details into a database.**

AIM:

To design and implement a Student Registration Form using C# and Windows Forms, allowing users to input student details and save them to a database.

PROCEDURE:

1. **Create a New Windows Forms App in Visual Studio**:
   - Open Visual Studio.
   - Create a new project and select **Windows Forms App (.NET Framework)**.
   - Name your project and click **Create**.
2. **Add the SQLite Database**:
   - You'll need to install System.Data.SQLite. In the **Package Manager Console**, run: **Install-Package System.Data.SQLite**
3. **Create the Form UI**:
   - Drag and drop labels, text boxes, and buttons onto the form to capture details such as:
     - Student ID
     - First Name
     - Last Name
     - Age
     - Email
     - Phone
     - Address
     - Submit button

4. **Code the Form**:
    - Set up the connection to an SQLite database.
    - Handle the Submit button click to insert data into the database.

PROGRAM:

MainForm.cs

This form handles the registration and database insertion.

```
using System;
using System.Data.SQLite;
using System.Windows.Forms;

namespace StudentRegistrationApp
{
   public partial class MainForm : Form
   {
      private SQLiteConnection connection;

      public MainForm()
      {
         InitializeComponent();
         InitializeDatabase();
      }
      // Initialize SQLite Database
      private void InitializeDatabase()
      {
         string connectionString = "Data Source=StudentDB.sqlite;Version=3;";
         connection = new SQLiteConnection(connectionString);
         // Create database file if it doesn't exist
         SQLiteConnection.CreateFile("StudentDB.sqlite");
         // Open connection
         connection.Open();
         // Create table if it doesn't exist
         string createTableQuery = @"CREATE TABLE IF NOT EXISTS Students (
                                    ID INTEGER PRIMARY KEY AUTOINCREMENT,
                                    FirstName TEXT NOT NULL,
                                    LastName TEXT NOT NULL,
                                    Age INTEGER,
                                    Email TEXT,
                                    Phone TEXT,
                                    Address TEXT
                                 )";
```

```
        SQLiteCommand createTable = new SQLiteCommand(createTableQuery,
connection);
        createTable.ExecuteNonQuery();
    }

    private void btnSubmit_Click(object sender, EventArgs e)
    {
        // Collect data from input fields
        string firstName = txtFirstName.Text;
        string lastName = txtLastName.Text;
        int age = int.Parse(txtAge.Text);
        string email = txtEmail.Text;
        string phone = txtPhone.Text;
        string address = txtAddress.Text;

        // Insert data into database
        string insertQuery = "INSERT INTO Students (FirstName, LastName, Age,
Email, Phone, Address) " +
                             "VALUES (@FirstName, @LastName, @Age, @Email,
@Phone, @Address)";

        using (SQLiteCommand cmd = new SQLiteCommand(insertQuery,
connection))
        {
            cmd.Parameters.AddWithValue("@FirstName", firstName);
            cmd.Parameters.AddWithValue("@LastName", lastName);
            cmd.Parameters.AddWithValue("@Age", age);
            cmd.Parameters.AddWithValue("@Email", email);
            cmd.Parameters.AddWithValue("@Phone", phone);
            cmd.Parameters.AddWithValue("@Address", address);

            try
            {
                cmd.ExecuteNonQuery();
                MessageBox.Show("Student registered successfully!");
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error: " + ex.Message);
            }
        }
        // Clear form fields
        ClearFields();
    }
```

```
        private void ClearFields()
        {
            txtFirstName.Clear();
            txtLastName.Clear();
            txtAge.Clear();
            txtEmail.Clear();
            txtPhone.Clear();
            txtAddress.Clear();
        }
    }
}
```

MainForm.Designer.cs

This file contains the designer code, where you create the UI elements (this can also be done through Visual Studio's drag-and-drop editor).

```
partial class MainForm

{

    private System.ComponentModel.IContainer components = null;

    private System.Windows.Forms.TextBox txtFirstName;

    private System.Windows.Forms.TextBox txtLastName;

    private System.Windows.Forms.TextBox txtAge;

    private System.Windows.Forms.TextBox txtEmail;

    private System.Windows.Forms.TextBox txtPhone;

    private System.Windows.Forms.TextBox txtAddress;

    private System.Windows.Forms.Button btnSubmit;



    private void InitializeComponent()

    {

        this.txtFirstName = new System.Windows.Forms.TextBox();

        this.txtLastName = new System.Windows.Forms.TextBox();
```

```
this.txtAge = new System.Windows.Forms.TextBox();
this.txtEmail = new System.Windows.Forms.TextBox();
this.txtPhone = new System.Windows.Forms.TextBox();
this.txtAddress = new System.Windows.Forms.TextBox();
this.btnSubmit = new System.Windows.Forms.Button();

this.SuspendLayout();

// FirstName TextBox
this.txtFirstName.Location = new System.Drawing.Point(130, 30);
this.txtFirstName.Name = "txtFirstName";
this.txtFirstName.Size = new System.Drawing.Size(150, 20);

// LastName TextBox
this.txtLastName.Location = new System.Drawing.Point(130, 70);
this.txtLastName.Name = "txtLastName";
this.txtLastName.Size = new System.Drawing.Size(150, 20);

// Age TextBox
this.txtAge.Location = new System.Drawing.Point(130, 110);
this.txtAge.Name = "txtAge";
this.txtAge.Size = new System.Drawing.Size(50, 20);

// Email TextBox
```

```
this.txtEmail.Location = new System.Drawing.Point(130, 150);
this.txtEmail.Name = "txtEmail";
this.txtEmail.Size = new System.Drawing.Size(200, 20);

// Phone TextBox
this.txtPhone.Location = new System.Drawing.Point(130, 190);
this.txtPhone.Name = "txtPhone";
this.txtPhone.Size = new System.Drawing.Size(150, 20);
// Address TextBox
this.txtAddress.Location = new System.Drawing.Point(130, 230);
this.txtAddress.Name = "txtAddress";
this.txtAddress.Size = new System.Drawing.Size(250, 20);
// Submit Button
this.btnSubmit.Location = new System.Drawing.Point(130, 280);
this.btnSubmit.Name = "btnSubmit";
this.btnSubmit.Size = new System.Drawing.Size(100, 30);
this.btnSubmit.Text = "Submit";
this.btnSubmit.Click += new System.EventHandler(this.btnSubmit_Click);
// MainForm
this.ClientSize = new System.Drawing.Size(500, 350);
this.Controls.Add(this.txtFirstName);
this.Controls.Add(this.txtLastName);
this.Controls.Add(this.txtAge);
this.Controls.Add(this.txtEmail);
```

```
        this.Controls.Add(this.txtPhone);
        this.Controls.Add(this.txtAddress);
        this.Controls.Add(this.btnSubmit);
        this.Text = "Student Registration Form";
        this.ResumeLayout(false);
        this.PerformLayout();
    }
}
```

INPUT:

- **Name**: John Doe
- **Age**: 21
- **Gender**: Male
- **Email**: johndoe@example.com
- **Contact**: 1234567890

OUTPUT:

Form Output (on Successful Submission)

1. **User Input Form**
   The form would look like this in Windows Forms:

2. **Confirmation Message** After clicking the **Save** button, the form displays this message:

Message Box:

"Student registered successfully."

Database Output:

Query: SELECT * FROM Students;

SQL Table Output:

Student Id	Name	Age	Gender	Email	Contact
1	John Doe	21	Gender	johndoe@example.com	1234567890

2. DesignandimplementaStudentFeePaymentSystemusingC#andWindowsForms.The application should allow students to enter their details, pay their fees, and generate a bill with a unique bill number.

AIM:

To design and implement a Student Fee Payment System using C# and Windows Forms, allowing students to enter their details, pay their fees, and generate a bill with a unique bill number.

PROCEDURE:

- **Create a Windows Forms Application**: Open Visual Studio, create a new Windows Forms Application project for the Student Fee Payment System.
- **Design the Form**: Add controls for student details (Student ID, Name, Course), fee amount, and payment method. Include buttons for **Pay** and **Generate Bill**.
- **Setup Database**: Create a SQL Server database called StudentFeesDB with tables for Students and Payments to store student details and payment records.
- **Database Connection**: Use ADO.NET to establish a connection to the StudentFeesDB database.
- **Test the Application**: Run the application to ensure data is correctly entered, saved, and a bill is generated with a unique bill number.

PROGRAM:

MainForm.cs

```
using System;
using System.Data.SQLite;
using System.Windows.Forms;

namespace StudentFeePaymentApp
{
   public partial class MainForm : Form
   {
      private SQLiteConnection connection;
      private int billNumber;
      public MainForm()
      {
         InitializeComponent();
         InitializeDatabase();
```

```
            GenerateBillNumber();
        }

        // Initialize SQLite Database
        private void InitializeDatabase()
        {
            string connectionString = "Data Source=StudentFeesDB.sqlite;Version=3;";
            connection = new SQLiteConnection(connectionString);

            // Create database file if it doesn't exist
            SQLiteConnection.CreateFile("StudentFeesDB.sqlite");

            // Open connection
            connection.Open();

            // Create table if it doesn't exist
            string createTableQuery = @"CREATE TABLE IF NOT EXISTS Fees (
                                ID INTEGER PRIMARY KEY AUTOINCREMENT,
                                StudentID TEXT NOT NULL,
                                FirstName TEXT NOT NULL,
                                LastName TEXT NOT NULL,
                                AmountPaid REAL,
                                BillNumber INTEGER
                            )";
            SQLiteCommand createTable = new SQLiteCommand(createTableQuery,
connection);
            createTable.ExecuteNonQuery();
        }

        // Generate unique bill number
        private void GenerateBillNumber()
        {
            Random random = new Random();
            billNumber = random.Next(1000, 9999);  // Generates a random 4-digit bill
number
            lblBillNumber.Text = "Bill Number: " + billNumber.ToString();
        }

        // Handle Pay Fees button click
        private void btnPayFees_Click(object sender, EventArgs e)
        {
            string studentID = txtStudentID.Text;
            string firstName = txtFirstName.Text;
```

```
        string lastName = txtLastName.Text;
        decimal amountPaid = decimal.Parse(txtAmountPaid.Text);

        // Insert payment data into database
        string insertQuery = "INSERT INTO Fees (StudentID, FirstName, LastName,
AmountPaid, BillNumber) " +
                             "VALUES (@StudentID, @FirstName, @LastName,
@AmountPaid, @BillNumber)";

        using (SQLiteCommand cmd = new SQLiteCommand(insertQuery,
connection))
        {
            cmd.Parameters.AddWithValue("@StudentID", studentID);
            cmd.Parameters.AddWithValue("@FirstName", firstName);
            cmd.Parameters.AddWithValue("@LastName", lastName);
            cmd.Parameters.AddWithValue("@AmountPaid", amountPaid);
            cmd.Parameters.AddWithValue("@BillNumber", billNumber);

            try
            {
                cmd.ExecuteNonQuery();
                MessageBox.Show("Payment recorded successfully!");
                GenerateBillNumber();  // Generate a new bill number after each
payment
                ClearFields();
            }
            catch (Exception ex)
            {
                MessageBox.Show("Error: " + ex.Message);
            }
        }
    }

    private void ClearFields()
    {
        txtStudentID.Clear();
        txtFirstName.Clear();
        txtLastName.Clear();
        txtAmountPaid.Clear();
    }
  }
}
```

MainForm.Designer.cs

In this file, we design the form layout, either through Visual Studio's drag-and-drop interface or by adding code for each UI element.

```
partial class MainForm
{
    private System.ComponentModel.IContainer components = null;
    private System.Windows.Forms.TextBox txtStudentID;
    private System.Windows.Forms.TextBox txtFirstName;
    private System.Windows.Forms.TextBox txtLastName;
    private System.Windows.Forms.TextBox txtAmountPaid;
    private System.Windows.Forms.Label lblBillNumber;
    private System.Windows.Forms.Button btnPayFees;

    private void InitializeComponent()
    {
        this.txtStudentID = new System.Windows.Forms.TextBox();
        this.txtFirstName = new System.Windows.Forms.TextBox();
        this.txtLastName = new System.Windows.Forms.TextBox();
        this.txtAmountPaid = new System.Windows.Forms.TextBox();
        this.lblBillNumber = new System.Windows.Forms.Label();
        this.btnPayFees = new System.Windows.Forms.Button();

        this.SuspendLayout();
```

```
// StudentID TextBox
this.txtStudentID.Location = new System.Drawing.Point(150, 30);
this.txtStudentID.Name = "txtStudentID";
this.txtStudentID.Size = new System.Drawing.Size(150, 20);

// FirstName TextBox
this.txtFirstName.Location = new System.Drawing.Point(150, 70);
this.txtFirstName.Name = "txtFirstName";
this.txtFirstName.Size = new System.Drawing.Size(150, 20);

// LastName TextBox
this.txtLastName.Location = new System.Drawing.Point(150, 110);
this.txtLastName.Name = "txtLastName";
this.txtLastName.Size = new System.Drawing.Size(150, 20);

// AmountPaid TextBox
this.txtAmountPaid.Location = new System.Drawing.Point(150, 150);
this.txtAmountPaid.Name = "txtAmountPaid";
this.txtAmountPaid.Size = new System.Drawing.Size(100, 20);

// BillNumber Label
this.lblBillNumber.AutoSize = true;
this.lblBillNumber.Location = new System.Drawing.Point(150, 190);
```

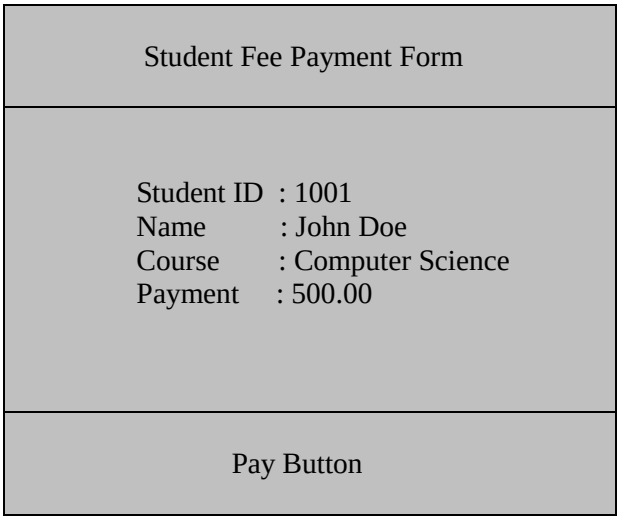
```
        this.lblBillNumber.Name = "lblBillNumber";
        this.lblBillNumber.Size = new System.Drawing.Size(75, 13);
        this.lblBillNumber.Text = "Bill Number: ";

        // PayFees Button
        this.btnPayFees.Location = new System.Drawing.Point(150, 230);
        this.btnPayFees.Name = "btnPayFees";
        this.btnPayFees.Size = new System.Drawing.Size(100, 30);
        this.btnPayFees.Text = "Pay Fees";
        this.btnPayFees.Click += new System.EventHandler(this.btnPayFees_Click);
        // MainForm
        this.ClientSize = new System.Drawing.Size(400, 300);
        this.Controls.Add(this.txtStudentID);
        this.Controls.Add(this.txtFirstName);
        this.Controls.Add(this.txtLastName);
        this.Controls.Add(this.txtAmountPaid);
        this.Controls.Add(this.lblBillNumber);
        this.Controls.Add(this.btnPayFees);
        this.Text = "Student Fee Payment System";
        this.ResumeLayout(false);
        this.PerformLayout();
    }
}
```

INPUT:

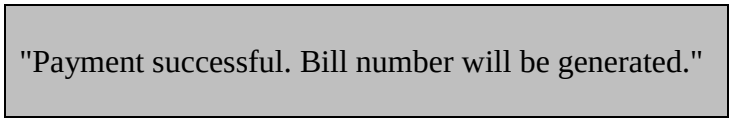
- **Student ID**: 1001
- **Name**: John Doe
- **Course**: Computer Science
- **Fee Amount**: 500.00

OUTPUT:

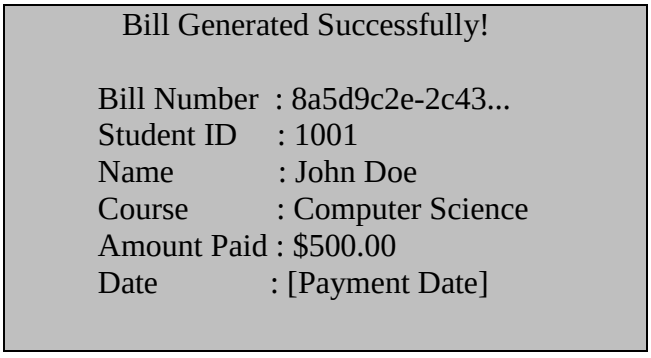
1.Form Layout:



2.Confirmation Message:
Message Box:



3.Bill Display:
Message Box:



3. DesignandimplementaWebServiceusingC#andASP.NETtoexposefunctionality for client applicationsto consume. Theweb serviceshould providea specificset ofoperations, such as retrieving data or performing a calculation.

AIM:

To design and implement a Web Service using C# and ASP.NET that exposes specific functionality for client applications to consume, such as retrieving data or performing calculations.

PROCEDURE:

- ✓ **Create an ASP.NET Web Service Project**: Open Visual Studio, create a new ASP.NET Web Application project, and select the Web API template.
- ✓ **Define the Web Service Operations**: Decide on a set of operations. For example, we'll create an operation that retrieves student data and calculates the average grade.
- ✓ **Implement Web Service Methods**:
- ✓ Define a method to retrieve a list of students.
- ✓ Define a method to calculate the average grade of a student.
- ✓ **Create a Data Model**: Define models for Student and Grade.
- ✓ **Test the Web Service**: Use tools like Postman or Swagger to test the Web API endpoints and verify that data is returned correctly.

PROGRAM:

In the Models folder, create a new class named Student.cs.

```
namespace StudentInfoService.Models
{
   public class Student
   {
      public int Id { get; set; }
      public string FirstName { get; set; }
      public string LastName { get; set; }
      public int Age { get; set; }
      public List<int> Marks { get; set; }
   }
}
```

In the Controllers folder, add a new **API Controller** called StudentController.cs.

```
using Microsoft.AspNetCore.Mvc;
using StudentInfoService.Models;
using System.Collections.Generic;
using System.Linq;

namespace StudentInfoService.Controllers
{
   [Route("api/[controller]")]
   [ApiController]
   public class StudentController : ControllerBase
   {
      private static List<Student> students = new List<Student>
      {
         new Student { Id = 1, FirstName = "John", LastName = "Doe", Age = 20,
Marks = new List<int> { 80, 85, 90 } },
         new Student { Id = 2, FirstName = "Jane", LastName = "Smith", Age = 22,
Marks = new List<int> { 75, 80, 82 } }
      };

      // GET api/student/{id}
      [HttpGet("{id}")]
      public ActionResult<Student> GetStudentById(int id)
      {
         var student = students.FirstOrDefault(s => s.Id == id);
         if (student == null)
         {
            return NotFound("Student not found.");
         }
         return Ok(student);
      }

      // POST api/student/calculateaverage
      [HttpPost("calculateaverage")]
      public ActionResult<double> CalculateAverageMarks([FromBody] List<int>
marks)
      {
         if (marks == null || !marks.Any())
         {
            return BadRequest("Marks list cannot be empty.");
         }
         double average = marks.Average();
```

```
            return Ok(average);
        }
    }
}
```

INPUT:

1. **Get All Students**:
    - URL: http://localhost:5000/api/student
    - Method: GET
2. **Get Average Grade for a Student**:
    - URL: http://localhost:5000/api/student/1/average
    - Method: GET

OUTPUT:


```
{
    "id": 1,
    "firstName": "John",
    "lastName": "Doe",
    "age": 20,
    "marks": [80, 85, 90]
}
```

4.Our college is organizing an Alumni Meet on May 5, 2024. The alumni cell is in the process of creating a database to store a list of registered alumni who will attend the event. You are tasked with designing a registration form and implementing it using ADO.NET.

Requirements:

1. **Design the Registration Form**:
    - Create a Windows Forms application that includes the following controls:
        - **TextBox** for entering the **Alumni Name**
        - **TextBox** for entering the **Email**
        - **TextBox** for entering the **Phone Number**
        - **ComboBox** for selecting the **Department** (e.g., Computer Science, Business, Arts)
        - **Button** to **Register** alumni
        - **Button** to **Display** registered alumni
        - **DataGridView** control to display the list of registered alumni from the selected department
2. **Implement Functionality Using ADO.NET**:
    - **Register Button**:
        - When the **Register** button is clicked, validate the input fields.
        - If the inputs are valid, insert the entered details into the database using ADO.NET. Handle any database exceptions that may occur.
    - **Display Button**:
        - When the **Display** button is clicked, retrieve all registered alumni for the selected department from the ComboBox.
        - Display the results in the **DataGridView** control.

AIM:

To design a Windows Forms application for alumni registration for the Alumni Meet and implement functionality to store and display registered alumni details using ADO.NET.

PROCEDURE:

1. **Create the Database**: Set up an SQL Server database named AlumniDB with an Alumni table containing columns for AlumniID, Name, Email, PhoneNumber, and Department.
2. **Design the Windows Form**:
    - Add text boxes for Alumni Name, Email, and Phone Number.
    - Add a ComboBox for selecting Department.
    - Add a **Register** button to save data, a **Display** button to show data, and a DataGridView to display registered alumni.
3. **Implement Register and Display Functionality Using ADO.NET**:
    - **Register Button**: Validate input fields and insert data into the Alumni table using an ADO.NET SqlCommand.
    - **Display Button**: Retrieve and display alumni data for the selected department in the DataGridView using an ADO.NET SqlDataAdapter.
4. **Test the Application**: Run the form to ensure alumni data is saved correctly in the database and displays in the DataGridView.

PROGRAM:

Database Setup:

```
-- Create the AlumniDB database
CREATE DATABASE AlumniDB;

USE AlumniDB;

-- Create the Alumni table
CREATE TABLE Alumni (
    AlumniID INT PRIMARY KEY IDENTITY,
    Name NVARCHAR(50),
    Email NVARCHAR(50),
    PhoneNumber NVARCHAR(15),
    Department NVARCHAR(50)
);
```

Code

```
using System;
using System.Data;
using System.Data.SqlClient;
using System.Windows.Forms;
namespace AlumniRegistrationApp
{
    public partial class AlumniForm : Form
    {
        private string connectionString = "Data Source=YourServerName;Initial
Catalog=AlumniDB;Integrated Security=True";

        public AlumniForm()
        {
            InitializeComponent();
            LoadDepartments();
        }
        private void LoadDepartments()
        {
            comboBoxDepartment.Items.AddRange(new string[] { "Computer Science",
"Business", "Arts" });
        }


        private void btnRegister_Click(object sender, EventArgs e)
        {
            string name = txtName.Text;
            string email = txtEmail.Text;
            string phoneNumber = txtPhoneNumber.Text;
            string department = comboBoxDepartment.SelectedItem?.ToString();
```

```
        if (string.IsNullOrWhiteSpace(name) || string.IsNullOrWhiteSpace(email) ||
            string.IsNullOrWhiteSpace(phoneNumber) ||
string.IsNullOrWhiteSpace(department))
        {
            MessageBox.Show("All fields are required.");
            return;
        }

        using (SqlConnection conn = new SqlConnection(connectionString))
        {
            try
            {
                conn.Open();
                string query = "INSERT INTO Alumni (Name, Email, PhoneNumber,
Department) VALUES (@Name, @Email, @PhoneNumber, @Department)";
                using (SqlCommand cmd = new SqlCommand(query, conn))
                {
                    cmd.Parameters.AddWithValue("@Name", name);
                    cmd.Parameters.AddWithValue("@Email", email);
                    cmd.Parameters.AddWithValue("@PhoneNumber", phoneNumber);
                    cmd.Parameters.AddWithValue("@Department", department);
                    cmd.ExecuteNonQuery();
                }
                MessageBox.Show("Alumni registered successfully.");
            }
            catch (Exception ex)
            {
                MessageBox.Show($"Error: {ex.Message}");
            }
        }
    }

    private void btnDisplay_Click(object sender, EventArgs e)
    {
        string department = comboBoxDepartment.SelectedItem?.ToString();
        if (string.IsNullOrWhiteSpace(department))
        {
            MessageBox.Show("Please select a department.");
            return;
        }
        using (SqlConnection conn = new SqlConnection(connectionString))
        {
            try
            {
                conn.Open();


              string query = "SELECT AlumniID, Name, Email, PhoneNumber,
Department FROM Alumni WHERE Department = @Department";
                using (SqlCommand cmd = new SqlCommand(query, conn))
                {

                    cmd.Parameters.AddWithValue("@Department", department);
```

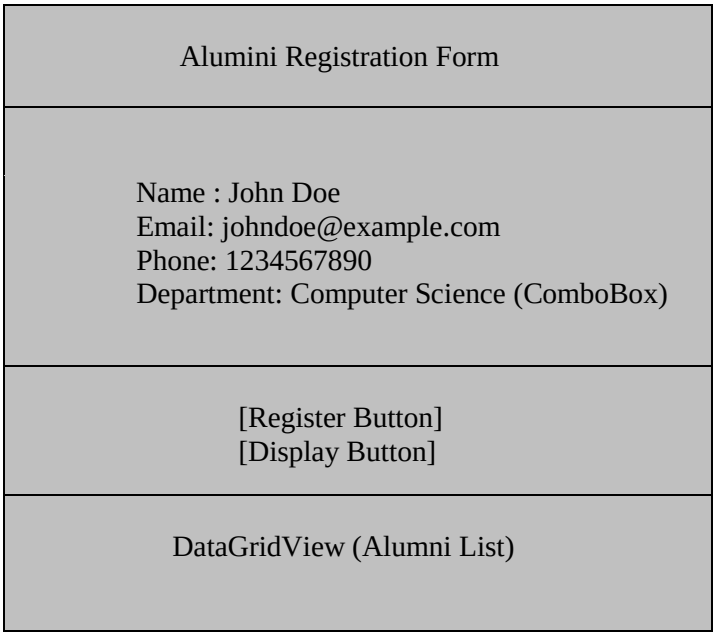
```
                using (SqlDataAdapter adapter = new SqlDataAdapter(cmd))
                {
                    DataTable dt = new DataTable();
                    adapter.Fill(dt);
                    dataGridViewAlumni.DataSource = dt;
                }
            }
        }
        catch (Exception ex)
        {
            MessageBox.Show($"Error: {ex.Message}");
        }
    }
  }
}
```

INPUT:

- **Alumni Name**: John Doe
- **Email**: johndoe@example.com
- **Phone Number**: 1234567890
- **Department**: Computer Science

DISPLAY:

Form Layout:



OUTPUT:

After Registering Alumni:

- Message Box:

"Alumni registered successfully."

Displaying Registered Alumni for Selected Department:

- On clicking **Display** with "Computer Science" selected, the DataGridView displays all registered alumni in the Computer Science department:

DataGridView (Alumni List):

Alumini Id	Name	Email	Phone	Dept
1	John Doe	johndoe@example.com	1234567890	CS