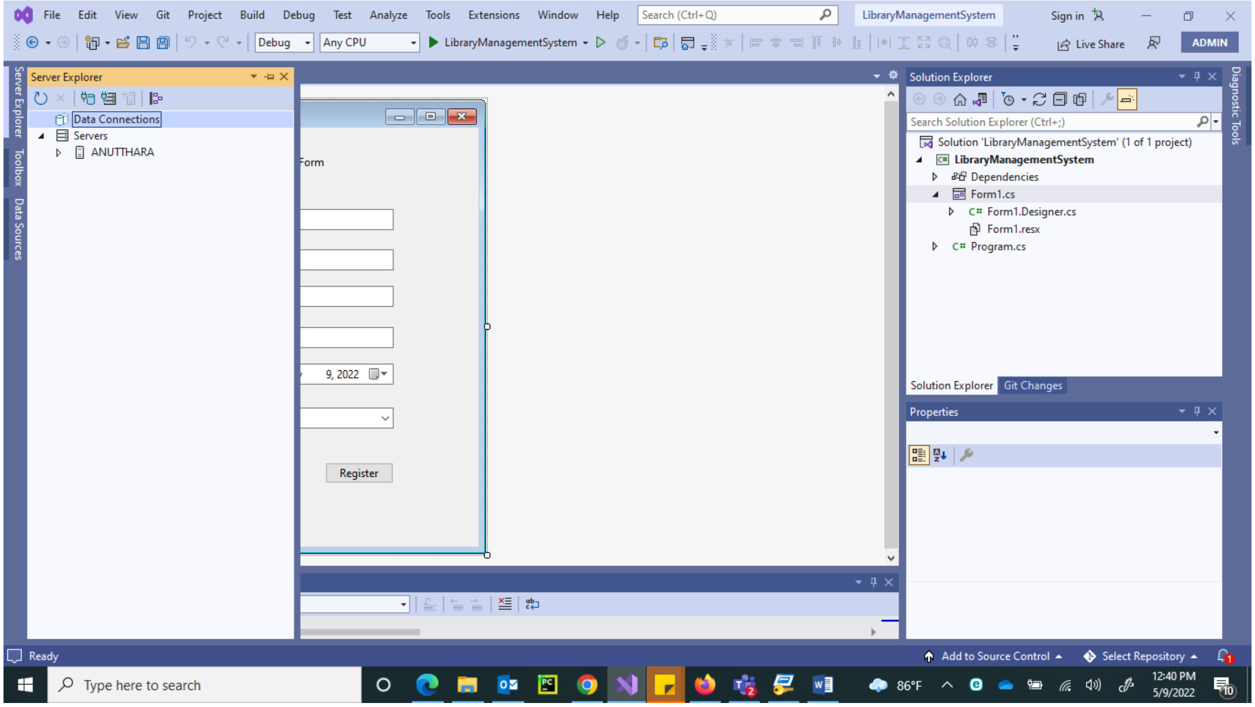
Developing GUI Application with C# .Net and MSSql

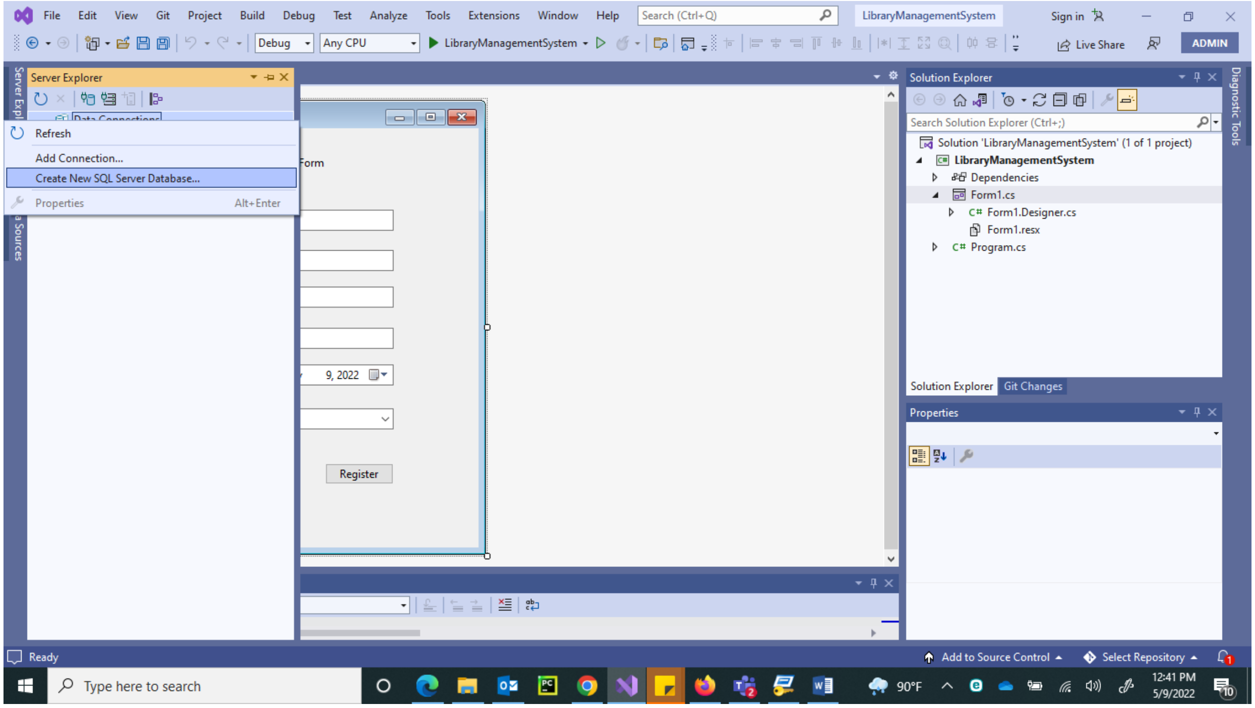
- Attend for the lecturer's demonstration session to start the application development.

Adding a database to the project

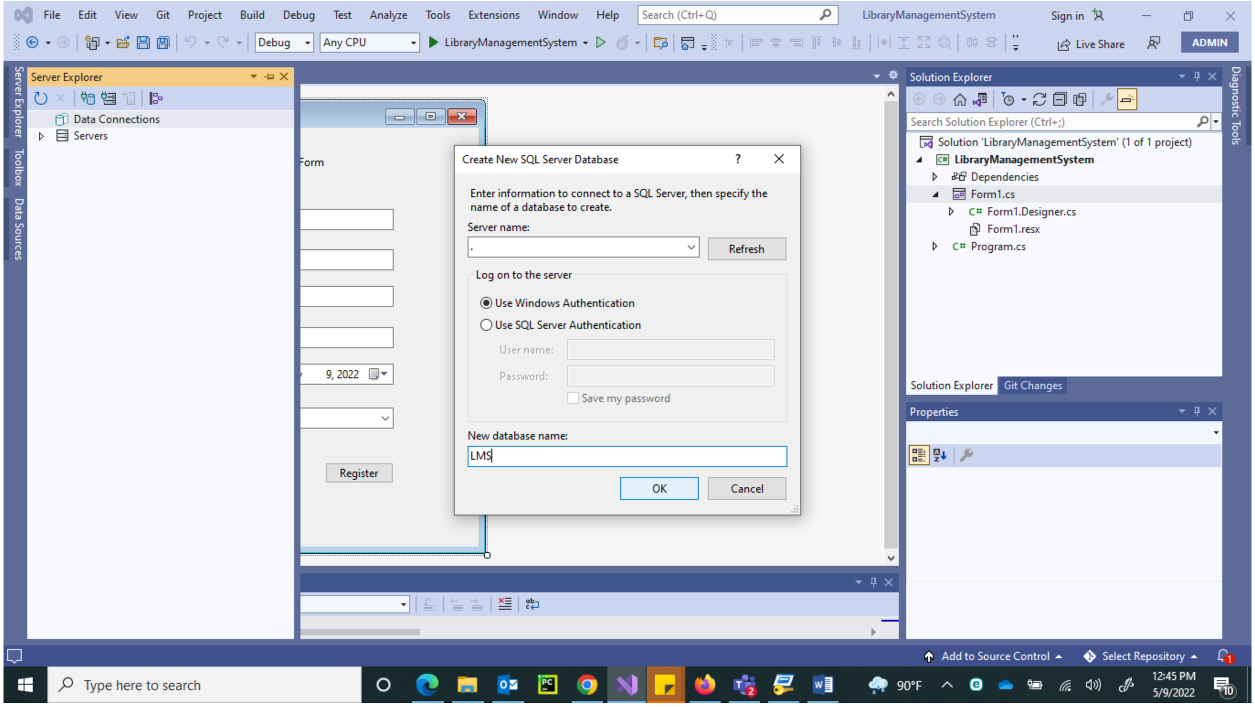
- Click on the Server Explorer



- Write click on the Data Connections and select Create New SQL Server Database



- Provide the server name (“.” Stands for the local host) and the database name that you want to create and clicks on Ok


Create New SQL Server Database

Enter information to connect to a SQL Server, then specify the name of a database to create.

Server name: .

Log on to the server

● Use Windows Authentication

○ Use SQL Server Authentication

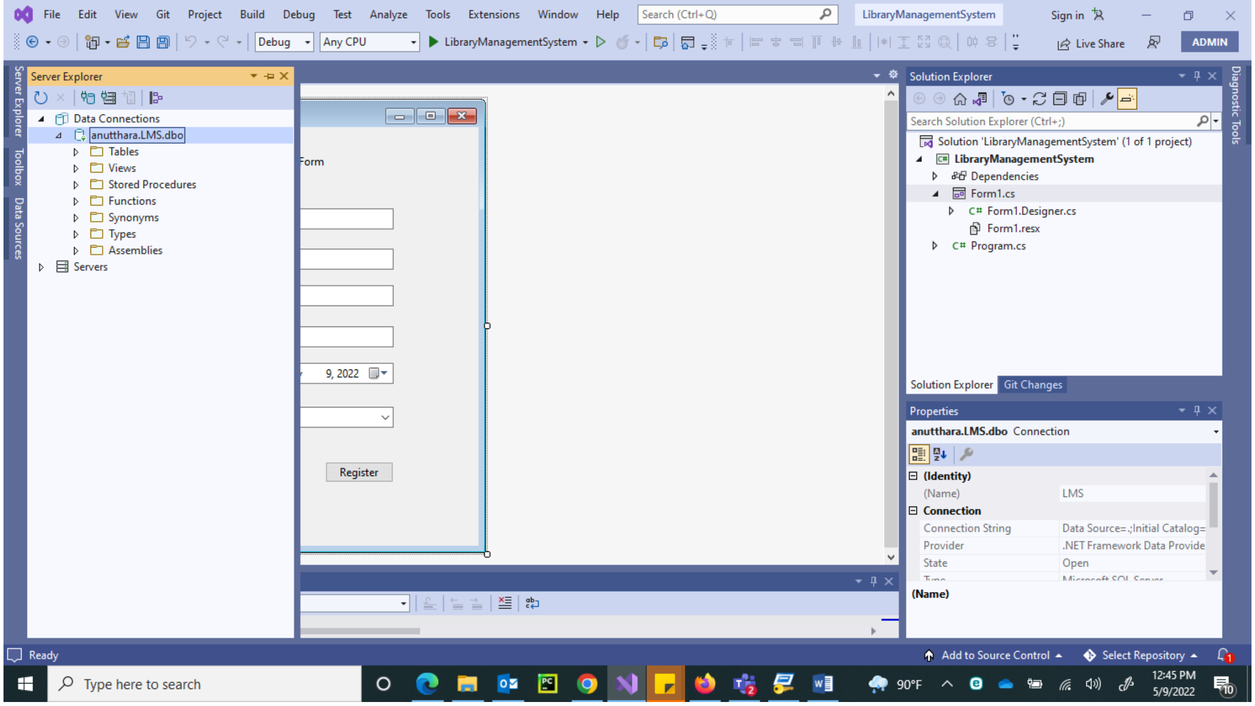
User name:

Password:

□ Save my password

New database name: LMS

- Expand the database and write click on the table to add new table



Properties

anutthara.LMS.dbo Connection

(Identity)

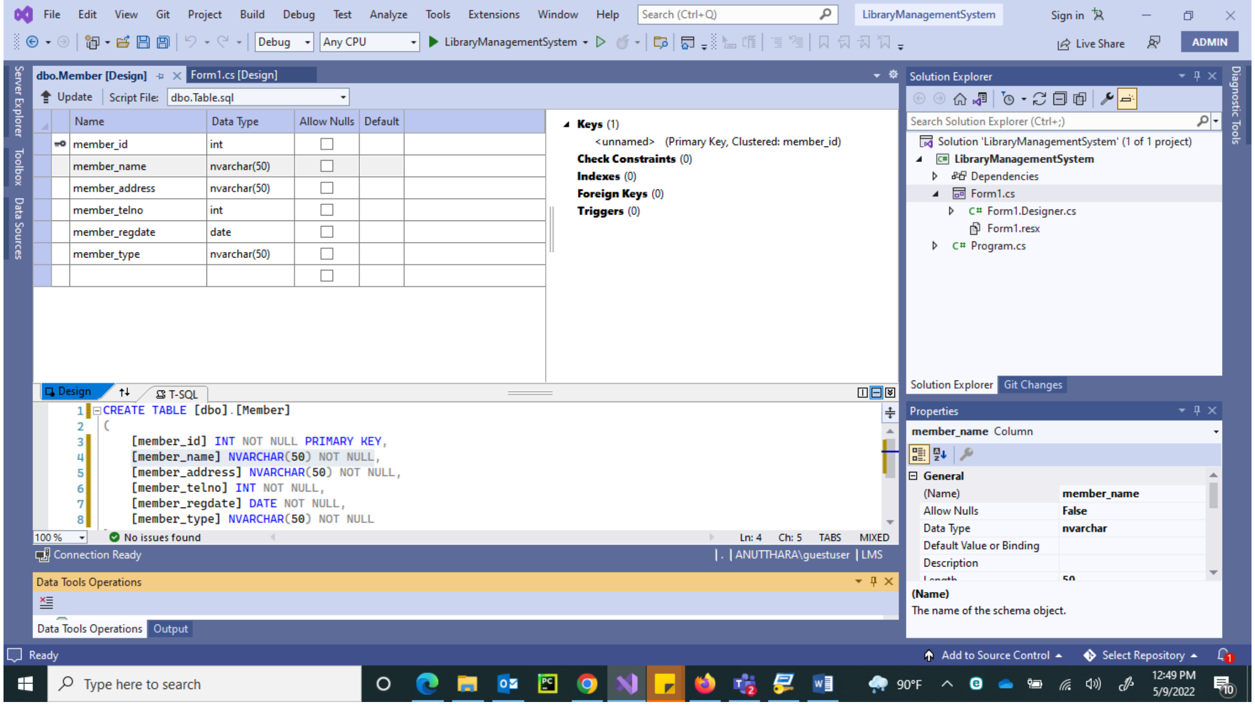
(Name)	LMS

Connection

Connection String	Data Source=.;Initial Catalog=
Provider	.NET Framework Data Provide
State	Open

(Name)

- Create the table Member according to the given requirements



Script File: dbo.Table.sql

	Name	Data Type	Allow Nulls	Default
🔑	member_id	int	□	
	member_name	nvarchar(50)	□	
	member_address	nvarchar(50)	□	
	member_telno	int	□	
	member_regdate	date	□	
	member_type	nvarchar(50)	□	
			□	

Keys (1)
<unnamed> (Primary Key, Clustered: member_id)
Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[Member]
(
    [member_id] INT NOT NULL PRIMARY KEY,
    [member_name] NVARCHAR(50) NOT NULL,
    [member_address] NVARCHAR(50) NOT NULL,
    [member_telno] INT NOT NULL,
    [member_regdate] DATE NOT NULL,
    [member_type] NVARCHAR(50) NOT NULL
```

100 % No issues found Ln: 4 Ch: 5 TABS MIXED

Connection Ready | . | ANUTTHARA\guestuser | LMS

Properties

member_name Column

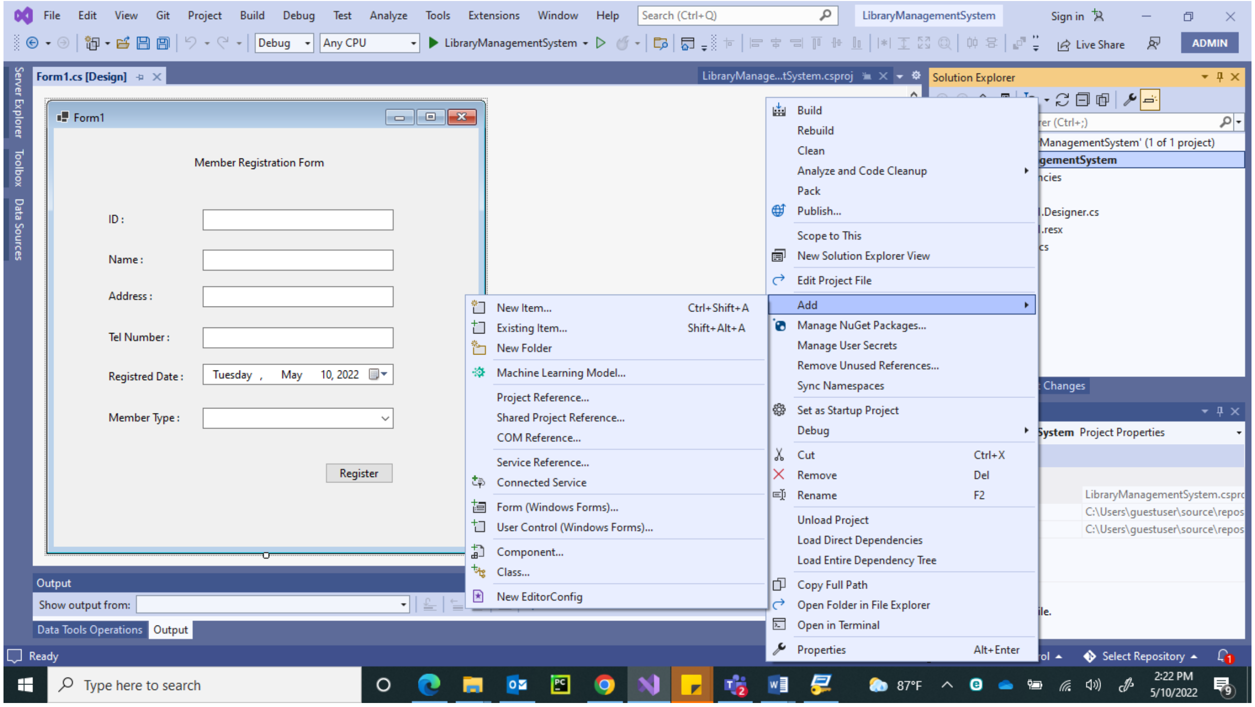
General

(Name)	member_name
Allow Nulls	False
Data Type	nvarchar
Default Value or Binding	
Description	
Length	50

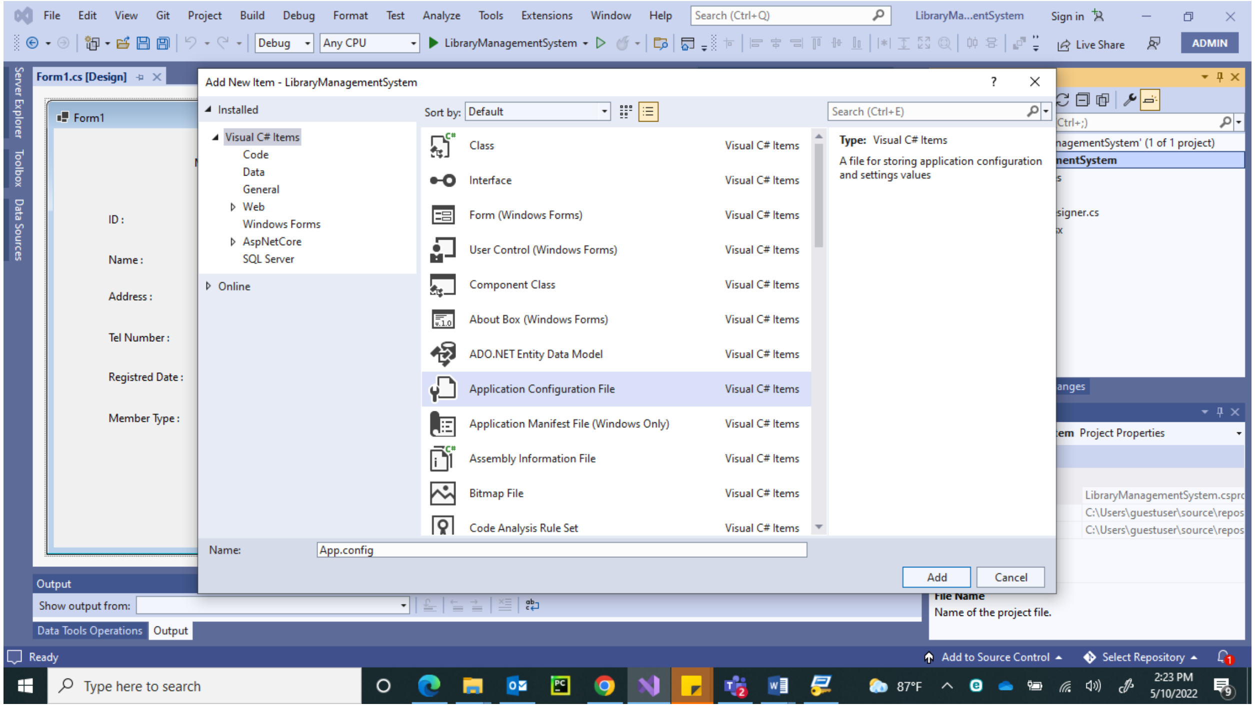
(Name)
The name of the schema object.

Adding a configuration file with a connection string

- Right click on the project and select “Add” and then select “New Item”



- From the given list , select Application Configuration File and click on Add with a proper name



Add New Item - LibraryManagementSystem

Class	Visual C# Items
Interface	Visual C# Items
Form (Windows Forms)	Visual C# Items
User Control (Windows Forms)	Visual C# Items
Component Class	Visual C# Items
About Box (Windows Forms)	Visual C# Items
ADO.NET Entity Data Model	Visual C# Items
Application Configuration File	Visual C# Items
Application Manifest File (Windows Only)	Visual C# Items
Assembly Information File	Visual C# Items
Bitmap File	Visual C# Items
Code Analysis Rule Set	Visual C# Items

Type: Visual C# Items

A file for storing application configuration and settings values

Name: App.config

Complete the App.config file with the following code. Take the connection string by moving to the connection string property in the database

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
	<connectionStrings>
		<add name="ConString" connectionString="Data Source=.;Initial 
Catalog=LMS;Integrated Security=True;
                    Pooling=False"/>
	</connectionStrings>
</configuration>
```

Data access using .NET Framework

The following table outlines the four core objects that make up a .Net framework data provider.

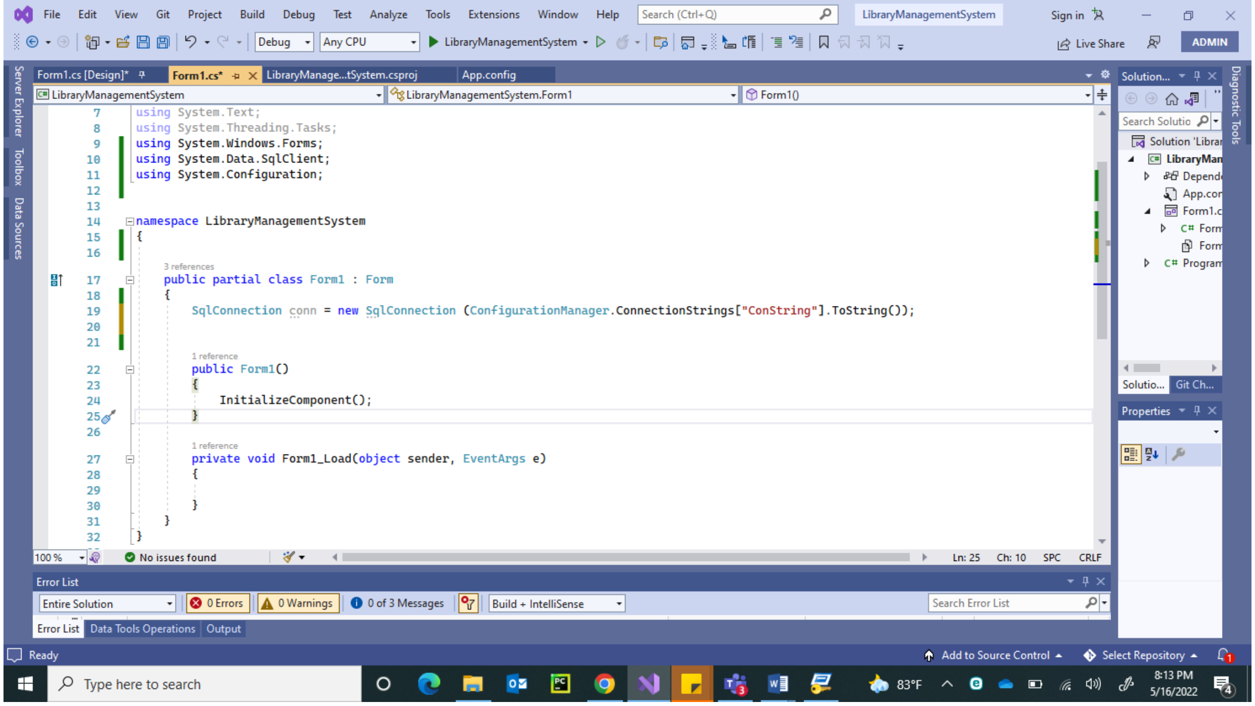
Object	Description
Connection	Establishes a connection to a specific data source
Command	Executes a command against a data source. Exposes **Parameters** and can execute within the scope of a **Transaction** from a **Connection**
DataReader	Reads a forward-only, read-only stream of data from adata source
DataAdapter	Populates a **DataSet** and resolves updates with the data source

Add Member Details to the system

- Add the name space on top of the form design

```
using System.Configuration;

using System.Data.SqlClient;
```

- Define the SQL connection on top of the Member Registration Form



```
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using System.Data.SqlClient;
using System.Configuration;


namespace LibraryManagementSystem
{

    3 references
    public partial class Form1 : Form
    {
        SqlConnection conn = new SqlConnection (ConfigurationManager.ConnectionStrings["ConString"].ToString());


        1 reference
        public Form1()
        {
            InitializeComponent();
        }

        1 reference
        private void Form1_Load(object sender, EventArgs e)
        {

        }
    }
}
```

Note : If you are getting an error in SqlConnection line, please refer the following link

https://www.mytecbits.com/microsoft/dot-net/error-sqlconnection-could-not-be-found

- Double click on the “Register” button and add the following code

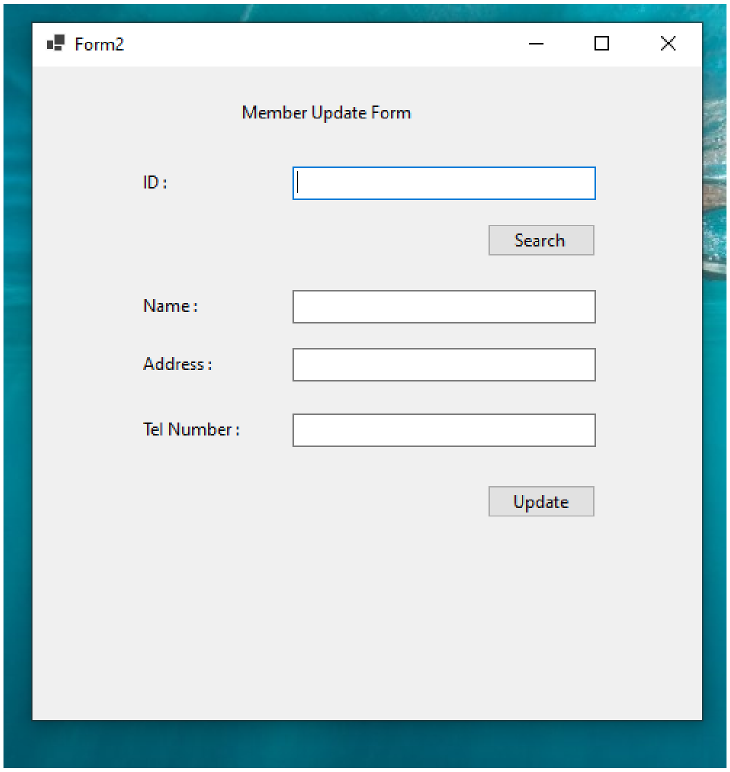
```
try
            {
                string regdate = dtpmemregdate.Value.ToString("yyyy-MM-dd"); 
//assigning the date from the date time picker
                string memtype = cmbmemtype.SelectedText; //assigning the member 
type
```

```
                string sql = "INSERT INTO Member
(member_id,member_name,member_address,member_telno,member_regdate,member_type)
VALUES(" + txtmemid.Text + ",'" + txtmemname.Text + "','" +txtmemaddress.Text + "',"
+ txtmemtnum.Text + ",'" + regdate + "','" + memtype+"')";
                SqlCommand cmd = new SqlCommand(sql, conn);
                conn.Open();
                cmd.ExecuteNonQuery();
                MessageBox.Show("Member Registration Successful ", "Library
Management System", MessageBoxButtons.OK, MessageBoxIcon.Information);

            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Library Management System",
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                conn.Close();
            }
```

Note : change the necessary variables according to your variable names

Update Member Details in the system

- Create the Member Update Form



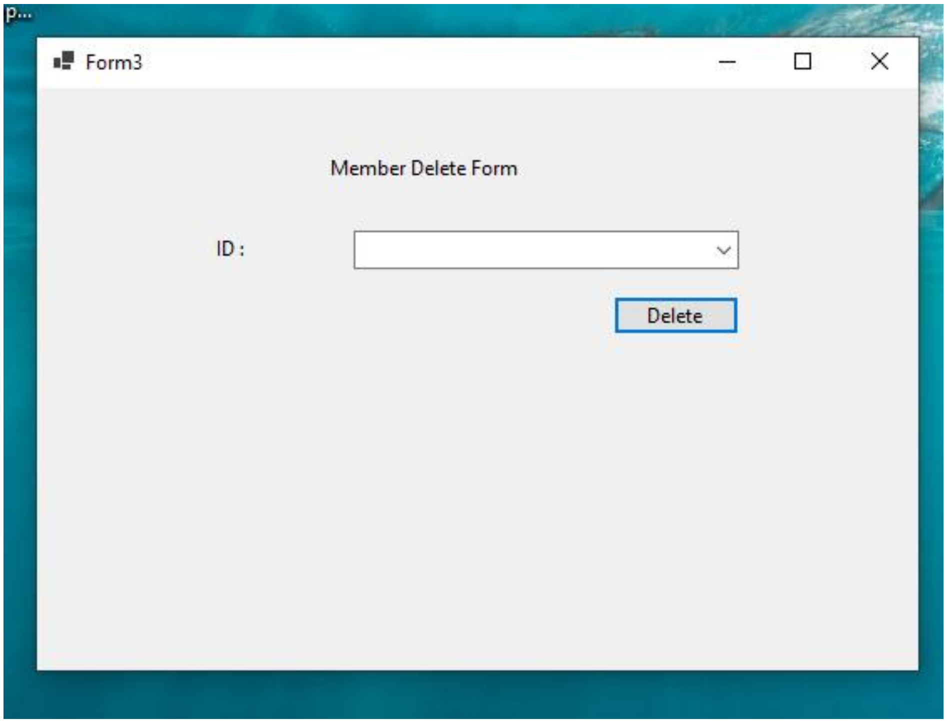
- Double click on the search button and add the following code

```
try
            {
                string sql = "SELECT * FROM Member where member_id=" + 
Convert.ToInt32(txtmemid.Text) + "";
                SqlCommand cmd = new SqlCommand(sql, conn);
                conn.Open();
                SqlDataReader dr = cmd.ExecuteReader();
                if (dr.Read())
                {
                    txtmemname.Text = dr["member_name"].ToString();
                    txtmemaddress.Text = dr["member_address"].ToString();
                    txtmemtnum.Text = dr["member_telno"].ToString();

                }
                else
                {
                    MessageBox.Show("Invalid ID", "Library Management System", 
MessageBoxButtons.OK, MessageBoxIcon.Exclamation);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Library Management System", 
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                conn.Close();
            }
```

- Double click on the update button and add the following code

```
try
            {
                string sql = "UPDATE Member SET member_name='" + txtmemname.Text + 
"', member_address='" + txtmemaddress.Text + "' , member_telno='" + txtmemtnum.Text 
+ "'";
                SqlCommand cmd = new SqlCommand(sql, conn);
                conn.Open();
                cmd.ExecuteNonQuery();

                MessageBox.Show("Record Successfully updated", "Library Management 
System", MessageBoxButtons.OK, MessageBoxIcon.Information);
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Library Management System", 
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                conn.Close();
                }
```

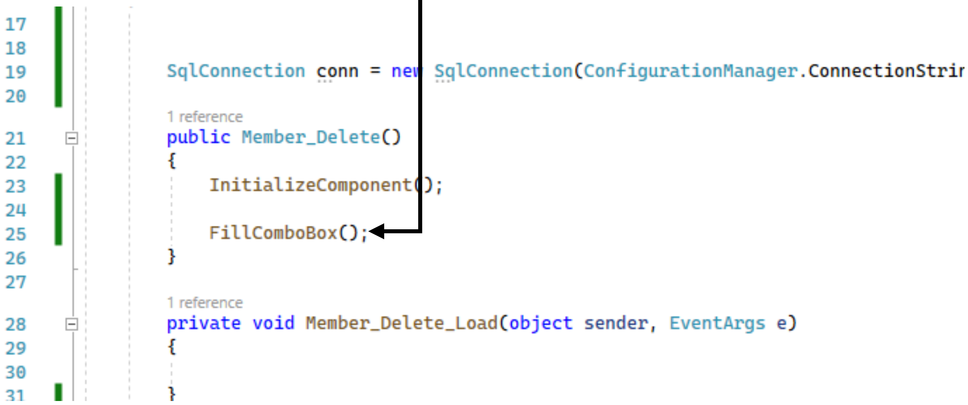
Delete Member Details in the system



- Write the following method in the delete form class

```
  protected void FillComboBox()
        {

            SqlDataAdapter da = new SqlDataAdapter("SELECT member_id FROM Member", 
conn);
            DataTable dt = new DataTable();
            da.Fill(dt);
            cmbmemid.DataSource = dt;
            cmbmemid.DisplayMember = "member_id";
            cmbmemid.ValueMember = "member_id";

        }
```

- And call the FillComboBox() method in the forms' constructor



```
SqlConnection conn = new SqlConnection(ConfigurationManager.ConnectionStrin

1 reference
public Member_Delete()
{
    InitializeComponent();

    FillComboBox();
}

1 reference
private void Member_Delete_Load(object sender, EventArgs e)
{

}
```

- Double click on the Delete button add the following code

```
try
            {
                DialogResult r = MessageBox.Show("Do you need to delete this 
record", "Student Registration System", MessageBoxButtons.YesNo, 
MessageBoxIcon.Information);



                if (r == DialogResult.Yes)
                {
                    string sql = "DELETE FROM Member WHERE member_id=" + 
Convert.ToInt32(cmbmemid.SelectedValue.ToString());
                    SqlCommand cmd = new SqlCommand(sql, conn);
                    conn.Open();
                    cmd.ExecuteNonQuery();



                    MessageBox.Show("Record Successfully deleted", "Library 
Management System", MessageBoxButtons.OK, MessageBoxIcon.Information);

                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message, "Library Management System", 
MessageBoxButtons.OK, MessageBoxIcon.Error);
            }
            finally
            {
                conn.Close();
            }
```