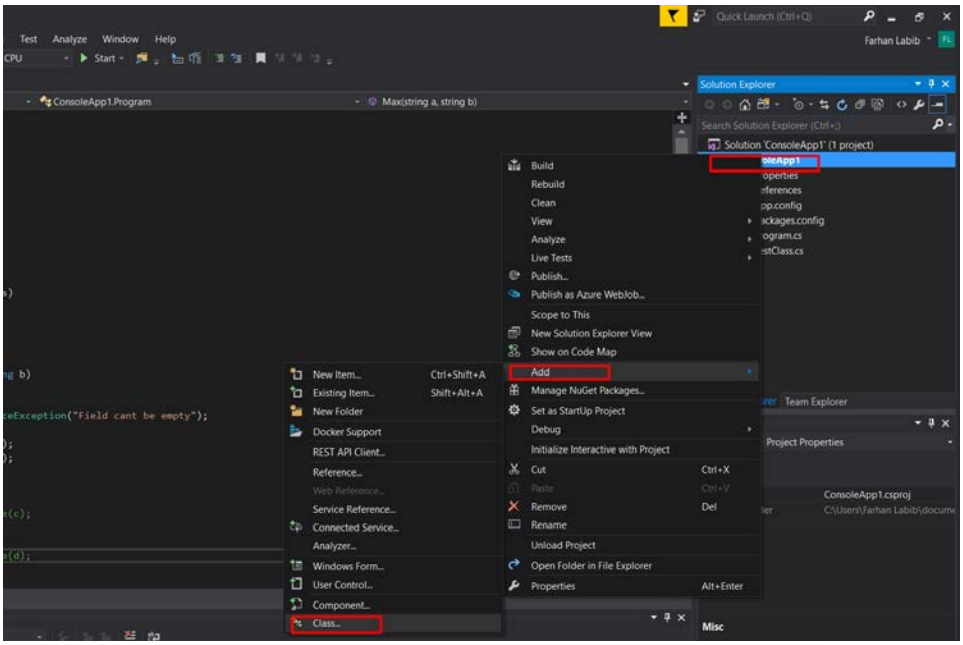
#Step_01:

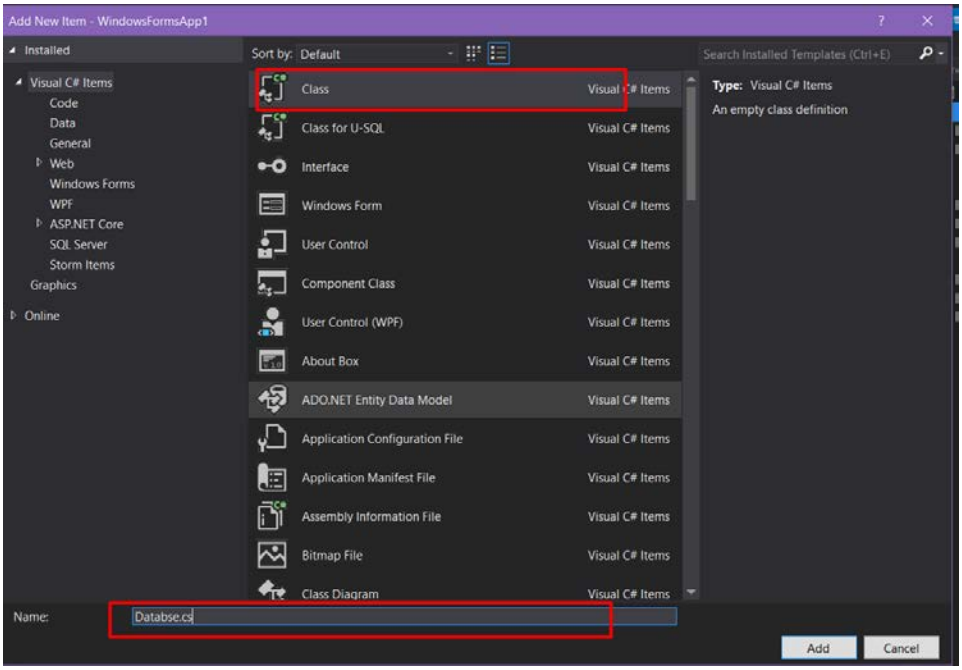
1. First, build a login page.
2. View this tutorial: https://www.youtube.com/watch?v=NX8-LhgFnUU
3. After building the login page, we have to test it.

#Step_02:

Now right click on your project and click on Add>Class



Now add a Class and named it as Database.



#Step_03:

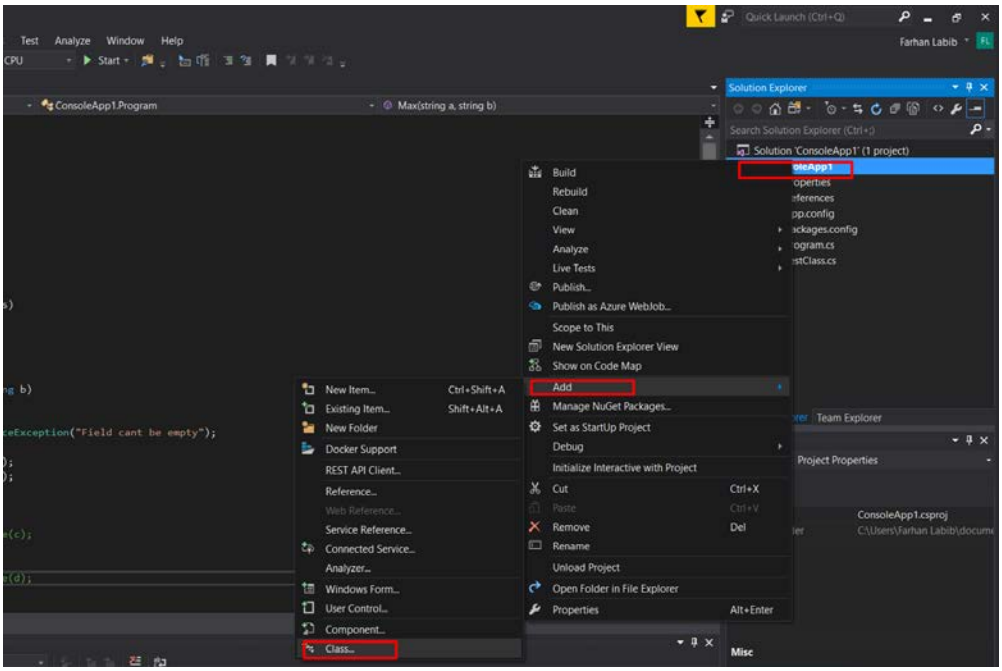
Now in database class, we need to build a database connection and try to use the same logic as we did for making the Login Form.

```
WindowsFormsApp1 | WindowsFormsApp1.Database | Checkdb(string name, string pass)
7   using System.Data;
8   using System.Drawing;
9   using System.Windows.Forms;
10  using System.Data.SqlClient;
11
12  namespace WindowsFormsApp1
13  {
        2 references
14      public class Database
15      {
16          SqlConnection sqlcon = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Farhan Labib\Documents\Visual Studi
17
            4 references | 2/3 passing
18          public bool Checkdb(string name, string pass)
19          {
20              if(name=="" || pass==="")
21              {
22                  throw new NullReferenceException("Field cannt be empty");
23              }
24
25              string query = "Select * from LoginDB Where username = '" + name.Trim() + "' and password = '" + pass.Trim() + "'";
26              SqlDataAdapter sda = new SqlDataAdapter(query, sqlcon);
27              DataTable dtbl = new DataTable();
28              sda.Fill(dtbl);
29              if (dtbl.Rows.Count == 0)
30              {
31                  return false;
32              }
33              else
34              {
35                  return true;
36              }
37
38          }
            1 reference | 1/1 passing
```

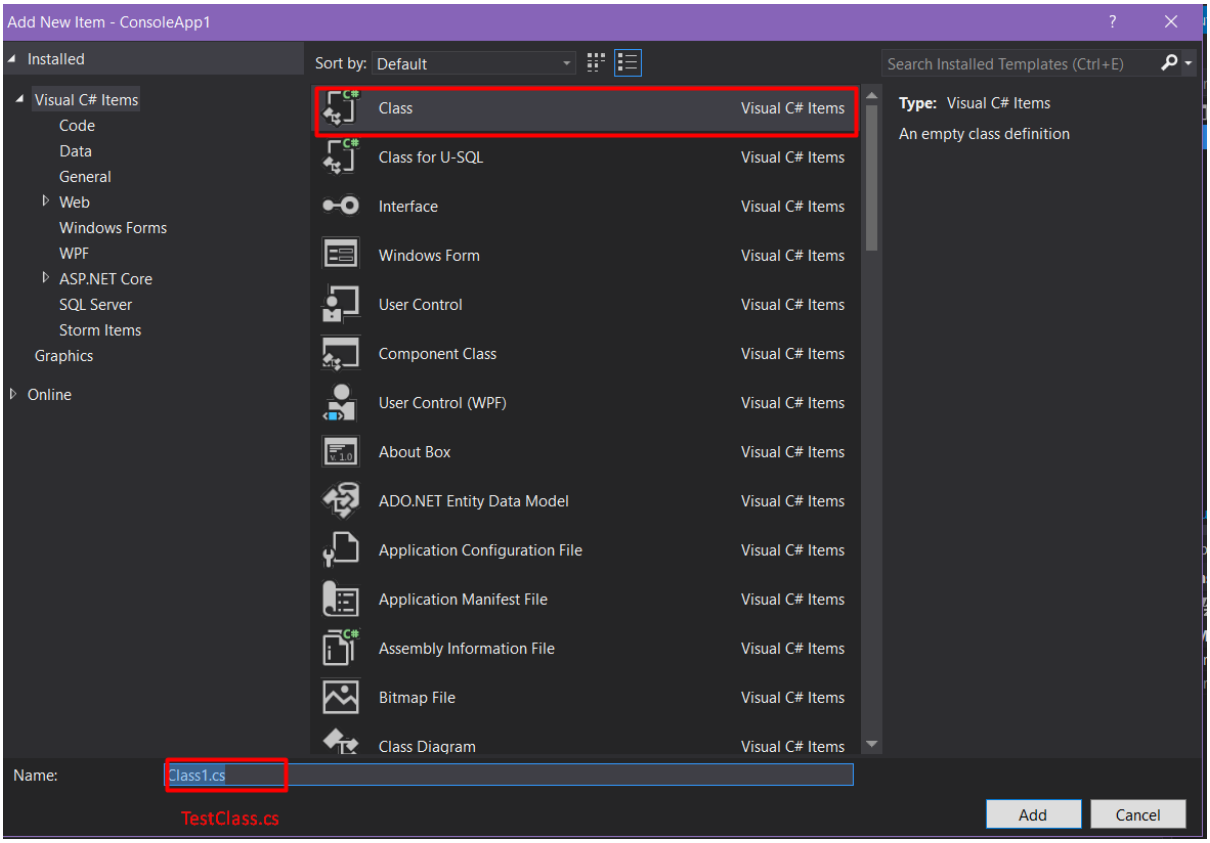
This method returns boolean because if we can log in successfully then it returns true and if not then it returns false.

#Step_04:

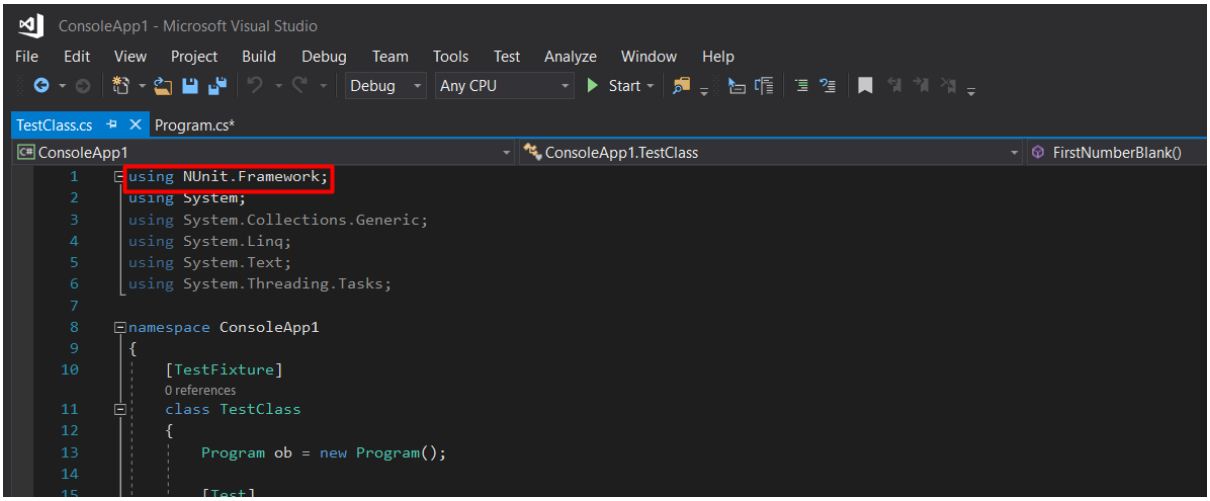
Now right click on your project and click on Add>Class



Create a class and name it to TestClass.cs



Then add a library> using Nunit.Framework;



#Step_05:

Then we have to create an object of our class.

There will four test cases for this Login method.

- Test Case 1: Login Using Valid Information
- Test Case 2: Login Using False Information
- Test Case 3: Username Null
- Test Case 4: Password Null

```
WindowsFormsApp1                    WindowsFormsApp1.TestClass                    inserttest()
1   using System;
2   using System.Collections.Generic;
3   using System.Linq;
4   using System.Text;
5   using System.Threading.Tasks;
6   using NUnit.Framework;
7
8   namespace WindowsFormsApp1
9   {
10      [TestFixture]
        0 references
11      class TestClass
12      {
13          Database ob = new Database();
14
15          [Test]
            | 0 references
16          public void entry()
17          {
18              bool r = ob.Checkdb("Farhan","1234");
19              Assert.IsTrue(r);
20          }
21          [Test]
            | 0 references
22          public void falseentry()
23          {
24              bool r = ob.Checkdb("Farhan", "4234");
25              Assert.IsFalse(r);
26          }
27          [Test]
            | 0 references
28          public void nullentry()
29          {
30              Assert.Throws<NullReferenceException>(()=> ob.Checkdb("", "4234"));
31          }
```

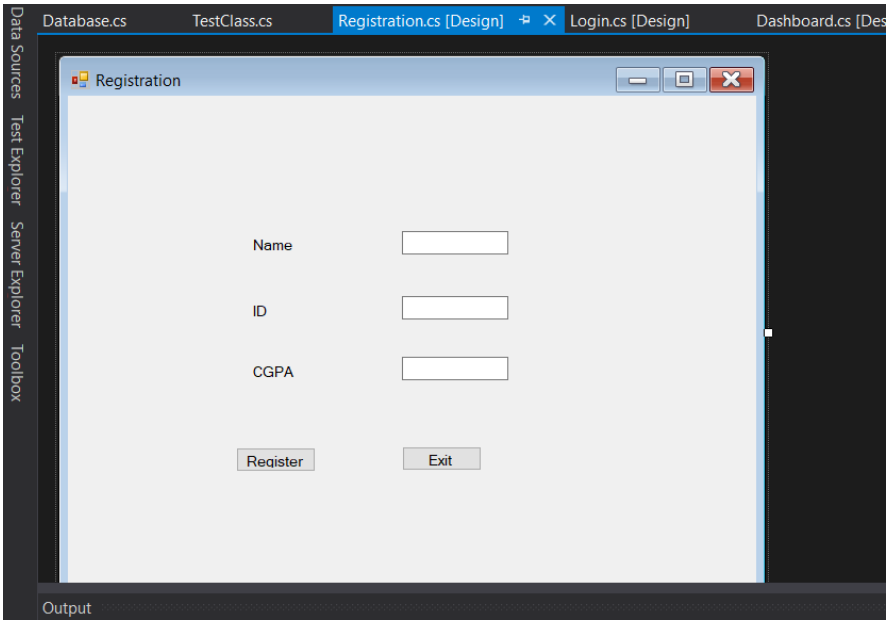
Now test your written code as we showed in the previous pdf.

After a successful run then it shows it passes All the test cases.

```
Debug    Any CPU    Start
Data Sources   Test Explorer   Server Explorer   Toolbox
Test Explorer                 cs [Design]   Login.cs [Design]   Dashboard.cs [Design]   Progra
Search                        WindowsFormsApp1.TestClass
Streaming Video: Configure continuous integrat
Run All | Run... | Playlist : All Tests
Passed Tests (4)
  entry          182 ms
  falseentry       3 ms
  nullentry        9 ms
  nullpassentry  < 1 ms
```

#Step_06:

Now we try to test insert operation. For this first, make the below form using your gained knowledge from the previously viewed tutorial.

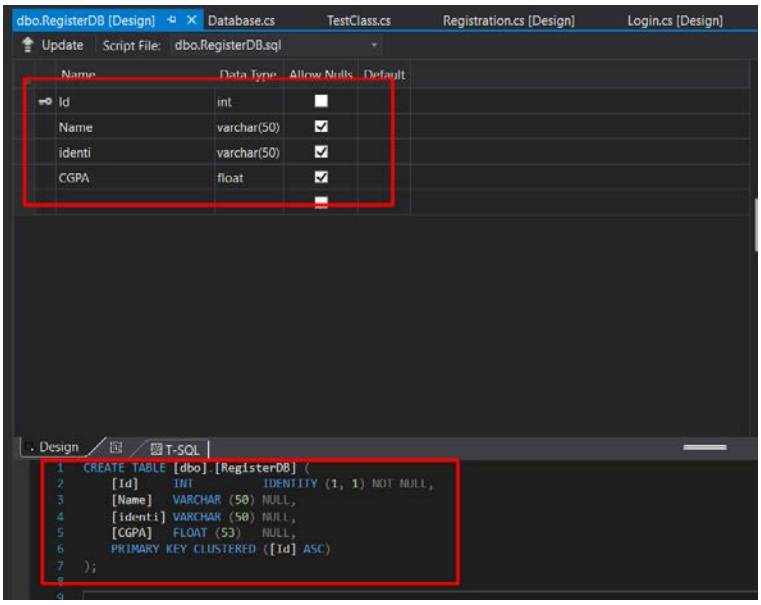


Your button code will be:

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

namespace WindowsFormsApp1
{
    3 references
    public partial class Registration : Form
    {
        SqlConnection sqlcon = new SqlConnection(@"Data Source=(LocalDB)\MSSQLLocalDB;AttachDbFilename=C:\Users\Farhan Labib\Documents\Visual Studio 2017\Proje

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

        1 reference
        private void btnReg_Click(object sender, EventArgs e)
        {
            if(txtName.Text != "" && txtID.Text != "" && txtCG.Text != "")
            {
                sqlcon.Open();
                string query = "insert into RegisterDB values ('" + txtName.Text.Trim() + "', '" + txtID.Text.Trim() + "', '" + txtCG.Text.Trim() + "')";
                SqlDataAdapter sda = new SqlDataAdapter(query, sqlcon);
                sda.SelectCommand.ExecuteNonQuery();
                MessageBox.Show("Registered Successfully");
            }
            else
            {
                MessageBox.Show("Fill The Exery Box Correctly");
            }
        }
    }
}
```

Figure:10

Your table definition should be:



#Step_07:

Now we go to the Database Class and write the code using the same logic as Registration (Figure:10) Form.

```
        }
        4 references | 4/4 passing
        public bool inesertdb(string name, string id, string cgpa)
        {
            if(name == "" || id == "" || cgpa == "")
            {
                throw new NullReferenceException("Field cannt be empty");
            }
            else if( name != "" && id !="" && cgpa!="")
            {
                sqlcon.Open();
                string query = "insert into RegisterDB values ('" + name.Trim() + "', '" + id.Trim() + "', '" + cgpa.Trim() + "')";
                SqlDataAdapter sda = new SqlDataAdapter(query, sqlcon);
                sda.SelectCommand.ExecuteNonQuery();
                return true;

            }
            else
            {
                return false;
            }

        }
```

#Step_08:

Now go to the TestClass and write the test code.

There will four test cases for this Insert method.

- Test Case 1: Registration Using Valid Information
- Test Case 2: Name Null
- Test Case 3: Id Null
- Test Case 4: CGPA Null

```
        }

        [Test]
        0 references
        public void inserttest()
        {
            bool get = ob.inesertdb("bony", "144", "3.5");
            Assert.IsTrue(get);
        }
        0 references
        public void namenullentry()
        {
            Assert.Throws<NullReferenceException>(() => ob.inesertdb("", "144", "3.5"));
        }
        [Test]
        0 references
        public void idnullentry()
        {
            Assert.Throws<NullReferenceException>(() => ob.inesertdb("bony", "", "3.5"));
        }
        [Test]
        0 references
        public void cgpanullentry()
        {
            Assert.Throws<NullReferenceException>(() => ob.inesertdb("bony", "144", ""));
        }


    }
}
```

Now run the test cases.