Lab 3

Program 1:

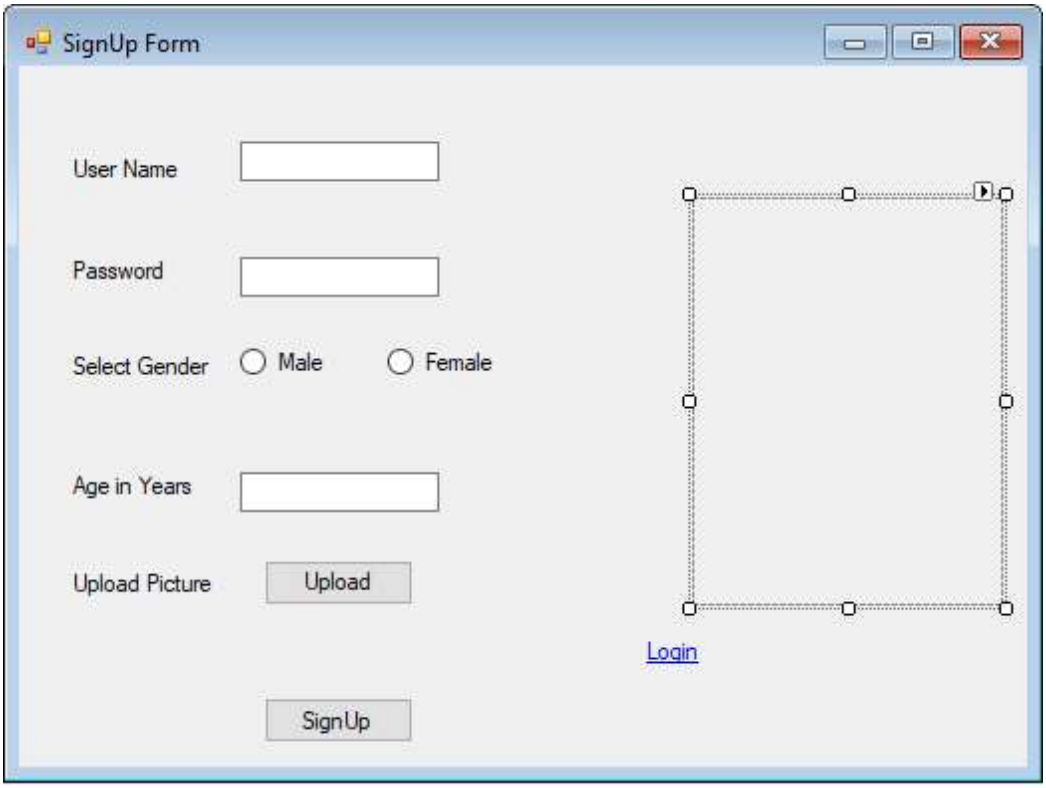
Make Login and Signup page, when user enter his/her user name and password in login form his/her picture should be displayed.

Solution:

Signup:

Make Signup form such that it can take Username, Password, Gender, Age and Picture.

For Gender you will take radio Button from toolbox, for picture you can take picture box.



Global vatiables:

```
string fPath;
System.Drawing.Image fs;  // Return type is Image
```

Upload Button(Event)

```
private void btnPicture_Click(object sender, EventArgs e)
        {
            OpenFileDialog fileDialog = new OpenFileDialog();     // open file Dialogue
            fileDialog.InitialDirectory = "C:\\Users\\Muhammad Usman Ghani\\Pictures";
            fileDialog.Title = "Choose image file";
            fileDialog.Filter = "JPeg Image|*.jpg|Bitmap Image|*.bmp|Gif Image|*.gif";
            if (fileDialog.ShowDialog() == DialogResult.OK)
```

```
            {

                fPath = fileDialog.FileName;


                fs = Image.FromFile(fPath);
               //this.btnPicture.Image.Save(fs, System.Drawing.Imaging.ImageFormat.Jpeg);
                pictureBox1.Image = fs;



            }
        }
```

Signup Button:

```
String extension = System.IO.Path.GetExtension(fPath);
            String fileName = txtuserName.Text;
            fileName = fileName + extension;
            String path = System.IO.Directory.GetCurrentDirectory();

            String pathWithName = System.IO.Path.Combine(path, fileName);

            fs.Save(pathWithName);

            Student std = new Student();
            std.UserName = txtuserName.Text;
            std.PassWord = txtpassWord.Text;
            std.SelectAge = txtAge.Text;
            std.ImagePath = pathWithName;

            if (rdMale.Checked)
            {
                std.SelectGender = "Male";
            }
            else
            {
                if (rdFemale.Checked)

{
                    std.SelectGender = "Female";
                }
            }

            datastore.data.Add(std);
            MessageBox.Show("SignUp Successfully");

        }
```

Login(Linkbutton) // For Navigation

```
string fPath;
        System.Drawing.Image fs;
```

Student Class(student.cs):

```
class Student
    {
        private string userName;
        private string passWord;
        private string selectGender;
        private string selectAge;
```

```
        private string fileContent;
        private System.Drawing.Image fss;
        private string imagePath;
                                                    //Encapsulation
public string UserName
        {
            get
            {
                return userName;
            }

            set
            {
                userName = value;
            }
        }

        public string PassWord
        {
            get
            {
                return passWord;
            }

            set
            {
                passWord = value;
            }
        }

public string SelectGender
        {
            get
            {
                return selectGender;
            }

            set
            {
                selectGender = value;
            }
        }

        public string SelectAge
        {
            get
            {
                return selectAge;
            }

            Set

{
                selectAge = value;
            }
        }

        public string FileContent
        {
            get
            {
                return fileContent;
            }

            set
```

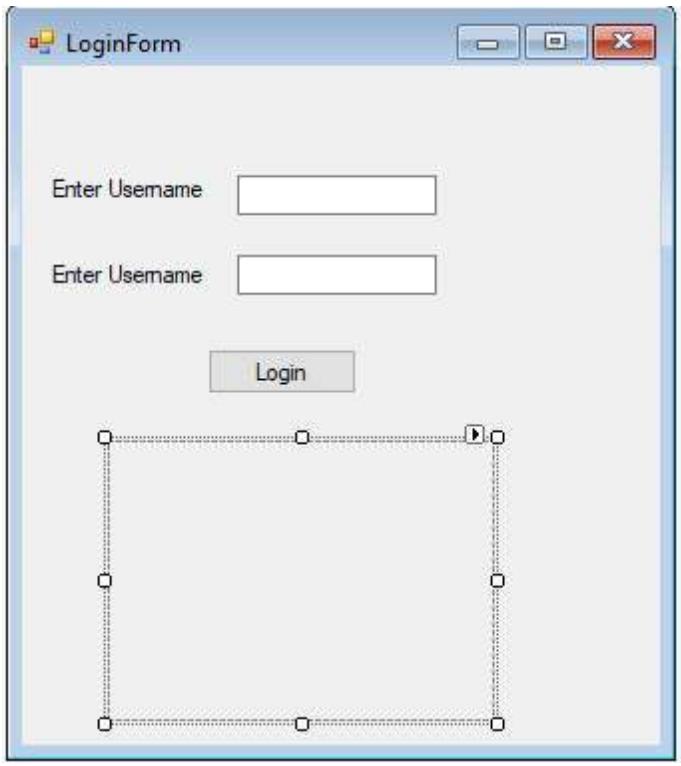
```
            {
                fileContent = value;
            }
        }
```

.....

Class to store data in arraylist (datastore.cs)

```
{
    class datastore
    {
      public static List<Student> data=new List<Student>();  //stereotype array list


    }
}
```

Login Form:




```
private void btnLogin_Click(object sender, EventArgs e)
        {
            Student std1 = datastore.data.Find(obj => obj.UserName == txtNameL.Text && obj.PassWord ==
txtpassL.Text); // predicate
            pbPicture.Load(std1.ImagePath);

        }
```

Home Work:

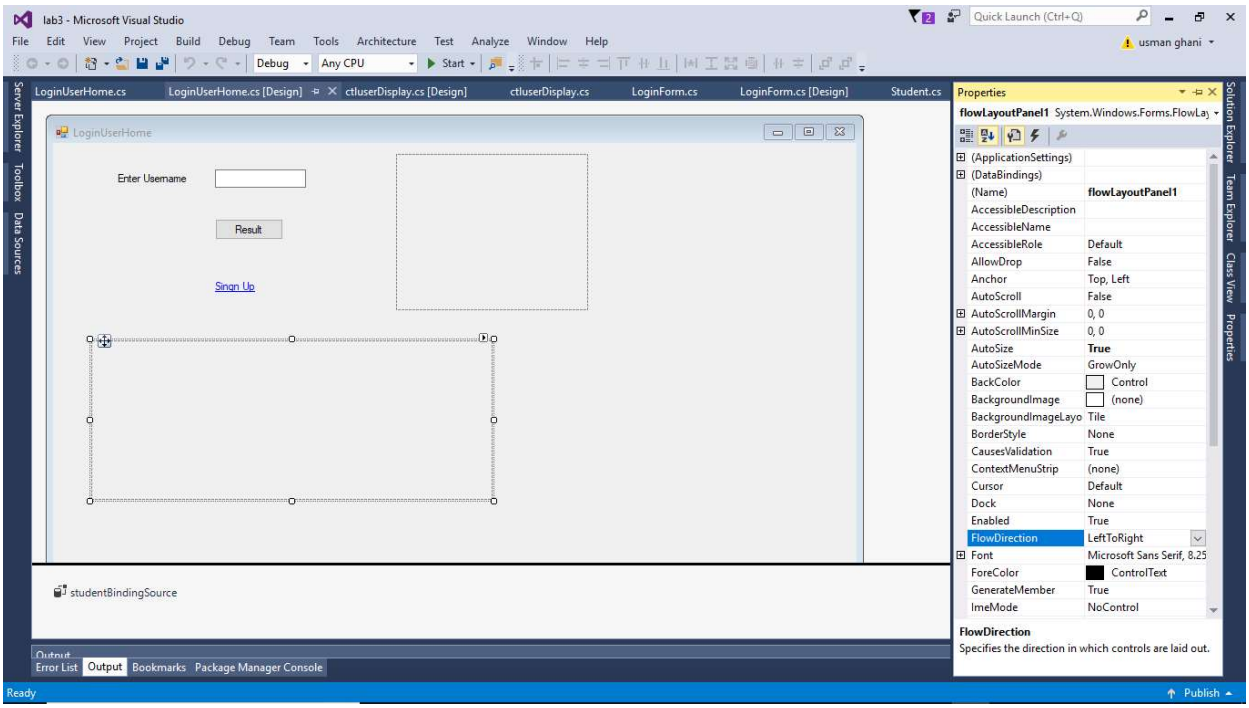
Extend this program and make 3rd form (LoginUserhomepage); on this form user can enter his her name, login event must show his picture in right corner and other user users picture in Button.

Hint: you can use grid to carry multiple users.

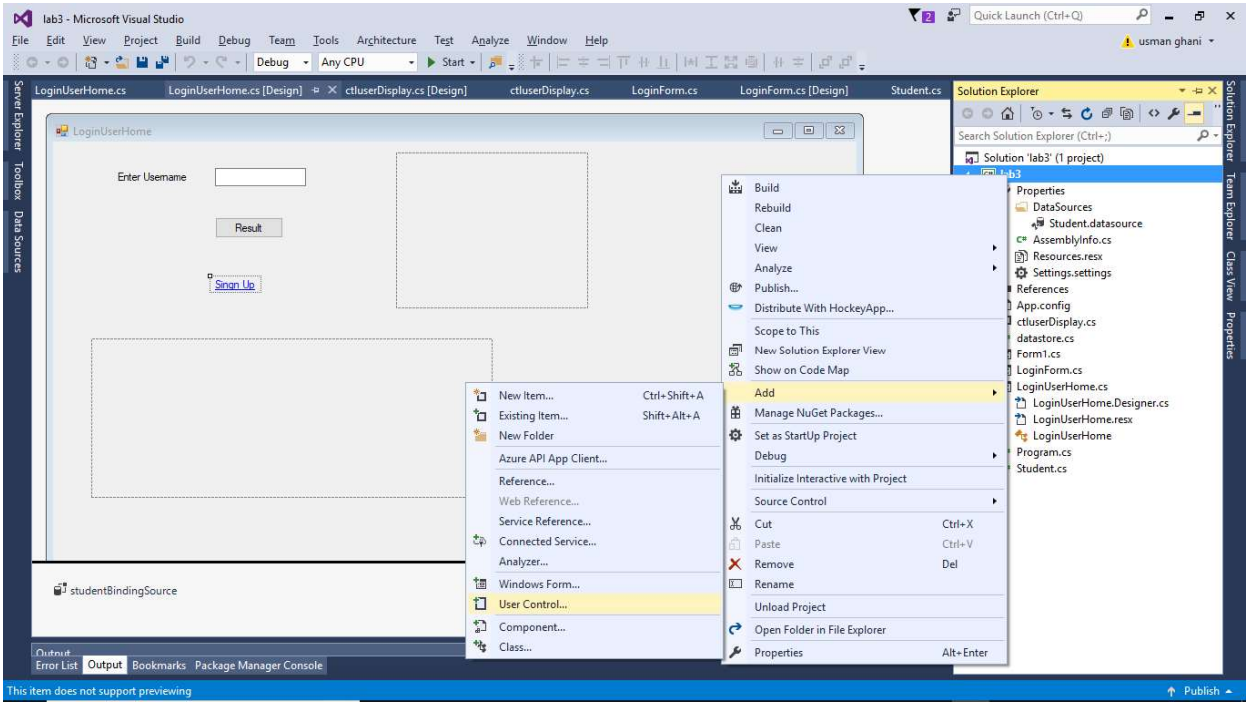
Task: Extend Lab: 3 and add new form (Login Homepage) that shows user's picture in PictureBox and all pictures in flowLayoutPanel.

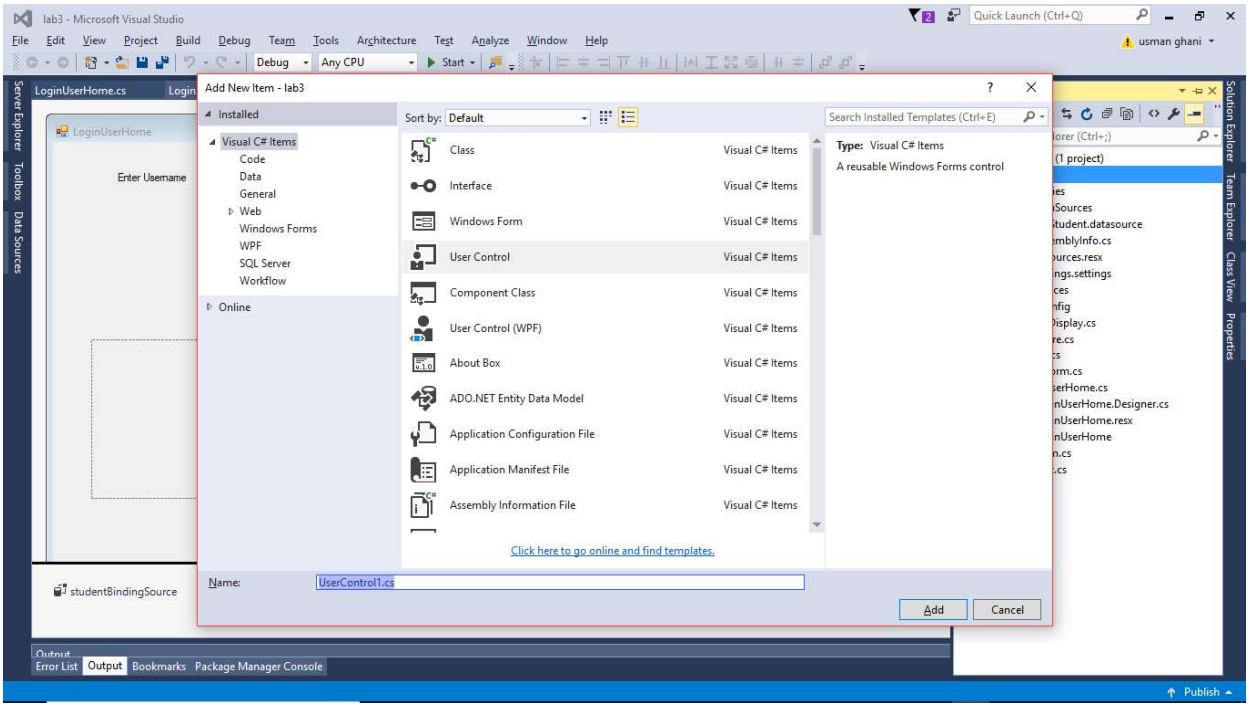
Step 1:

Take Picturebox, flowlayoutcontrol, button, label ,Textbox ,label and linkLabel from tool box

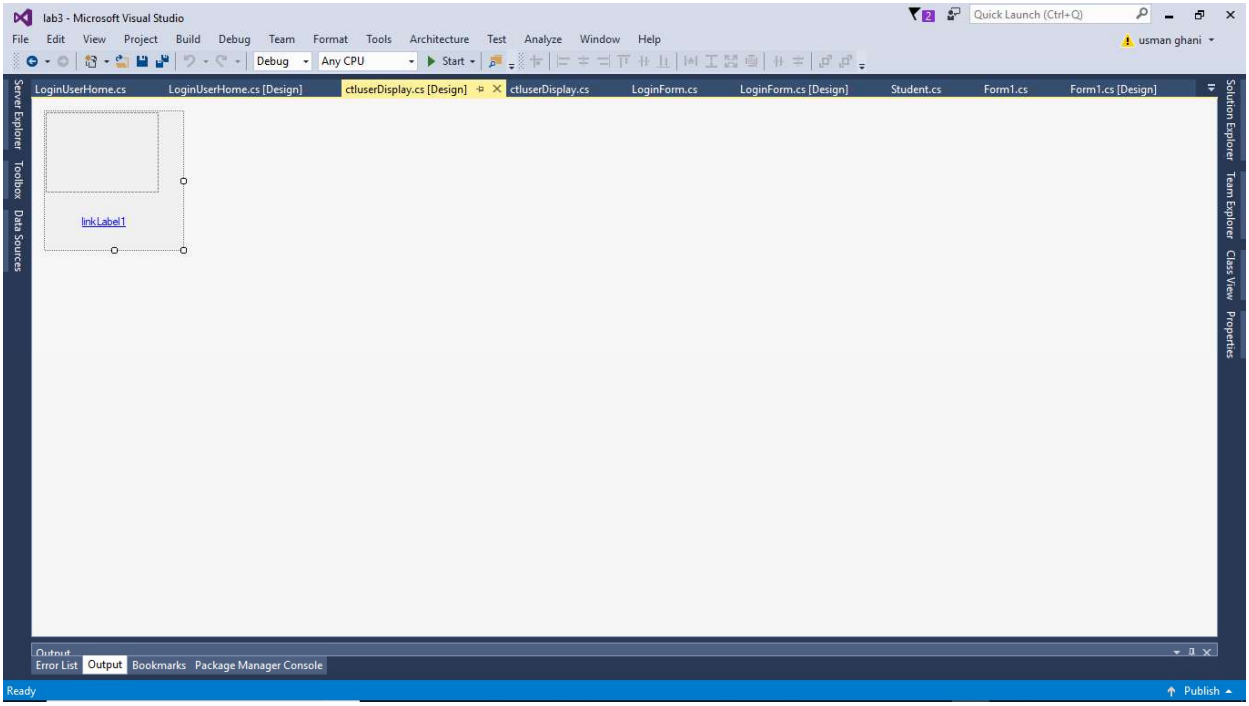


Step 2: Right click on project and add user control:





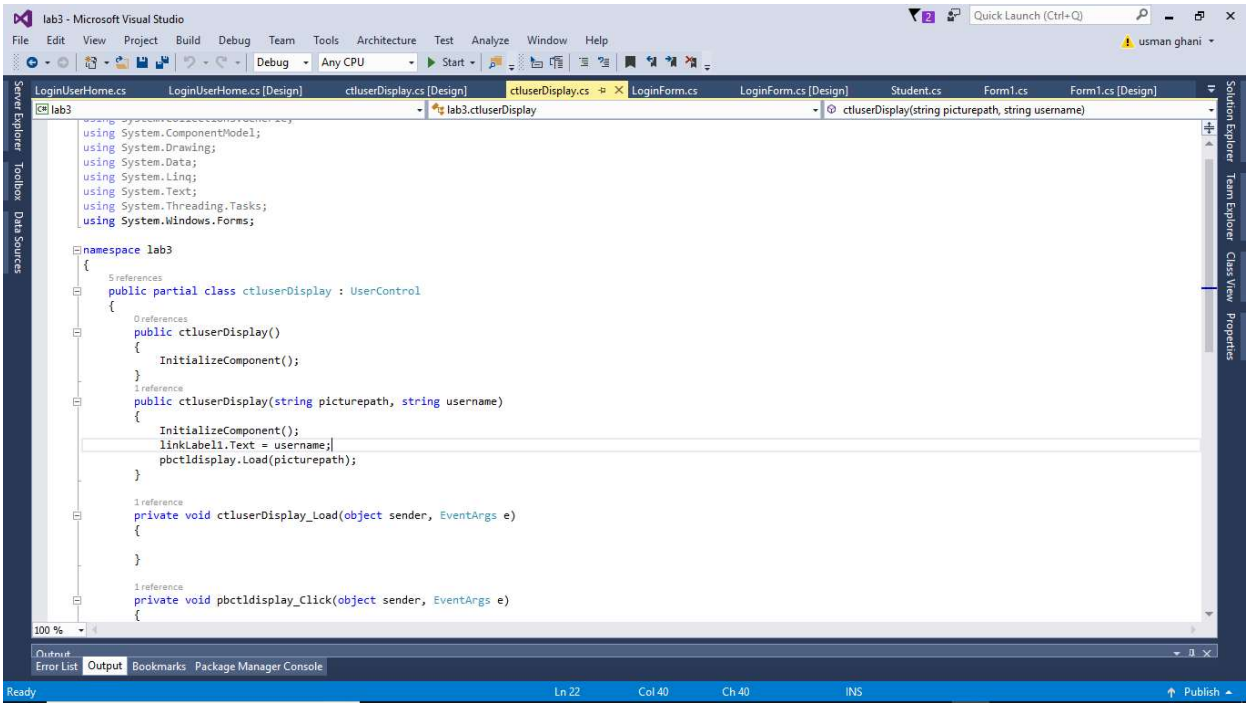
Step 3: Add picturebox and linklabel in controller so that it contain users picture and name.



Step 4:

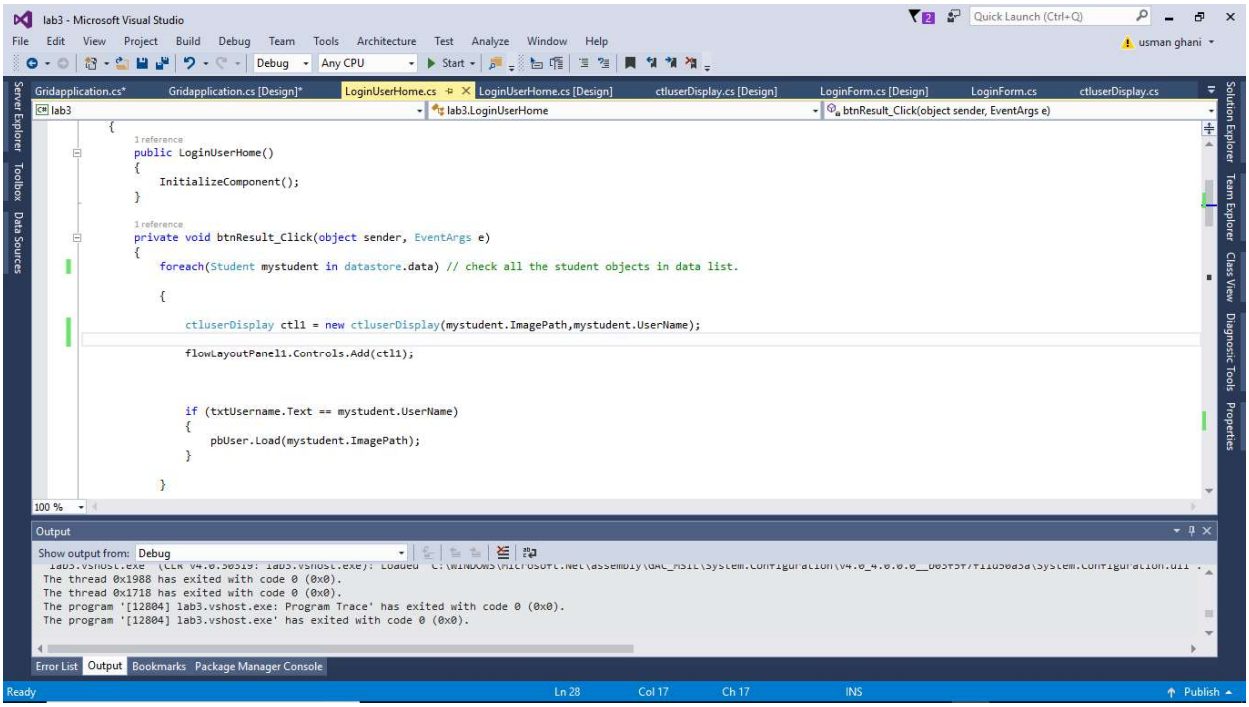
Write constructer for usercontrol(clusterDisplay)

Each time we make new instance of user controller,It takes picturepath and username from student object, and load picture in picture, assign username to linklabel text.

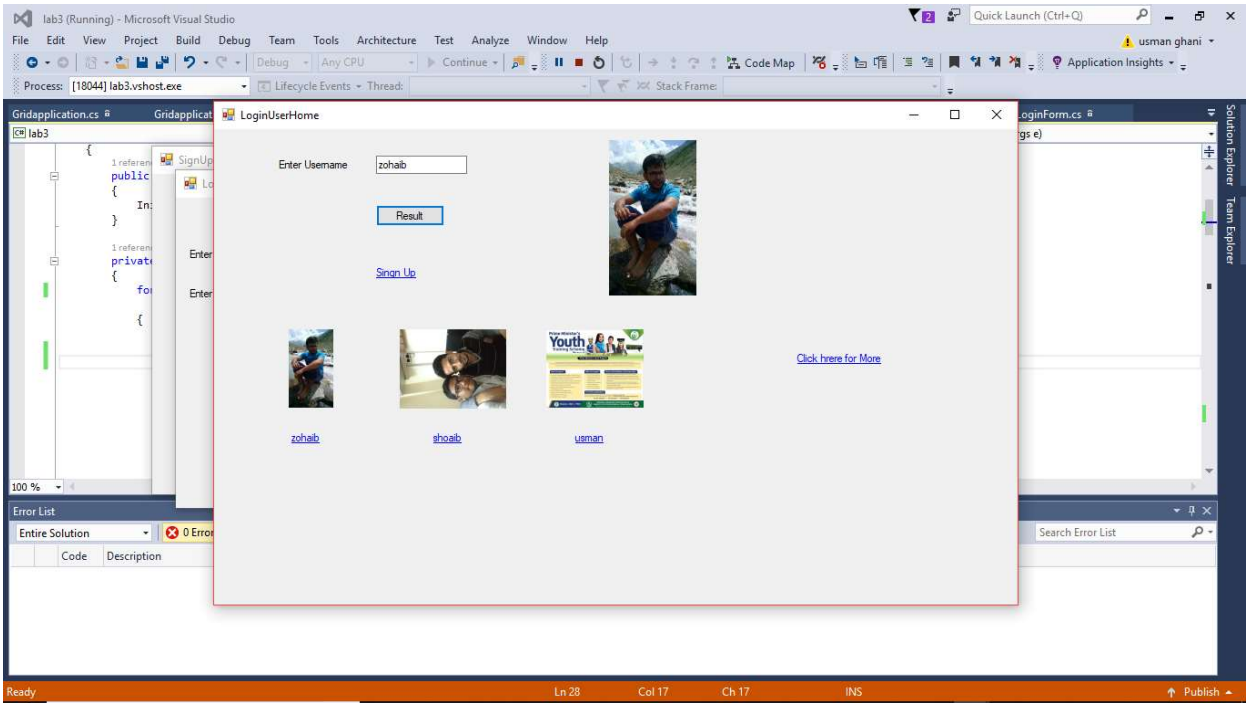


Step 5:

Double click on Result button and add following code to display picture in layout and picturebox



Output:



Windows Communication Foundation (WCF) is a framework for building service-oriented applications. Using WCF, you can send data as asynchronous messages from one service endpoint to another. A service endpoint can be part of a continuously available service hosted by IIS, or it can be a service hosted in an application. An endpoint can be a client of a service that requests data from a service endpoint. The messages can be as simple as a single character or word sent as XML, or as complex as a stream of binary data. A few sample scenarios include:

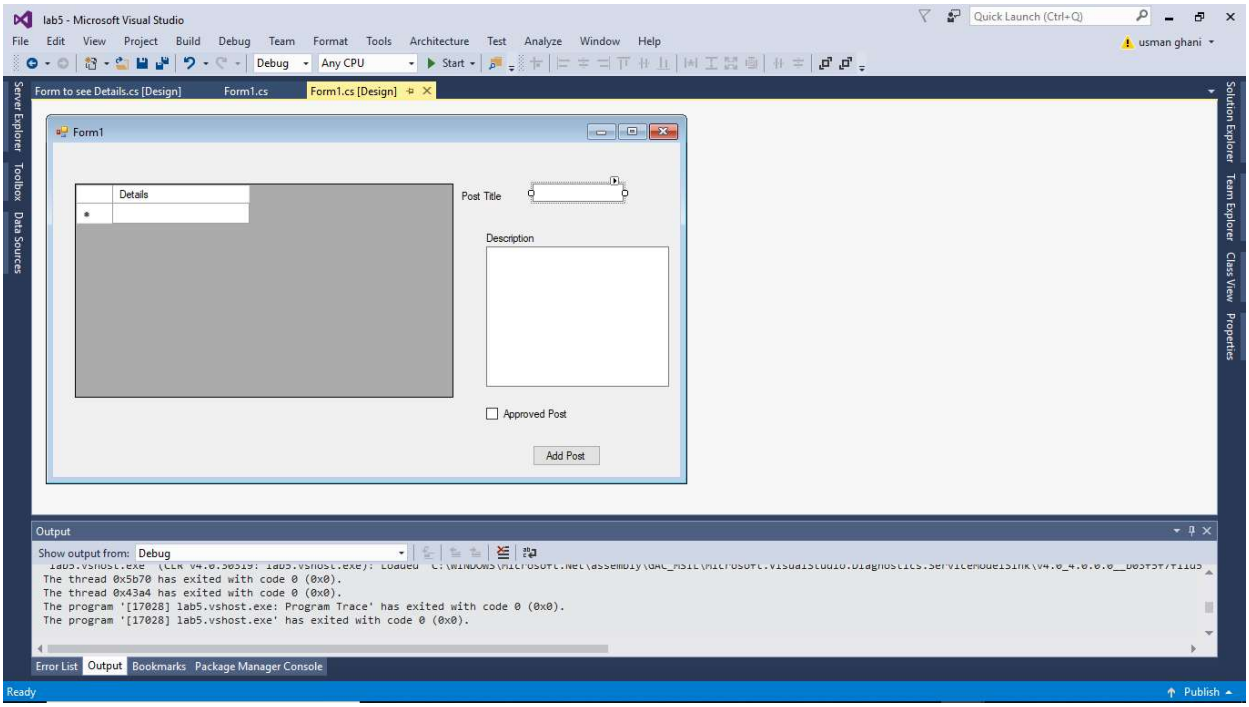
- A secure service to process business transactions.
- A service that supplies current data to others, such as a traffic report or other monitoring service.
- A chat service that allows two people to communicate or exchange data in real time.
- A dashboard application that polls one or more services for data and presents it in a logical presentation.
- Exposing a workflow implemented using Windows Workflow Foundation as a WCF service.
- A Silverlight application to poll a service for the latest data feeds.

Program 1: User can be enter “Post title”, “Description” ,”check Post is approved or not” and show Data in a DataGrid view.

Step 1:

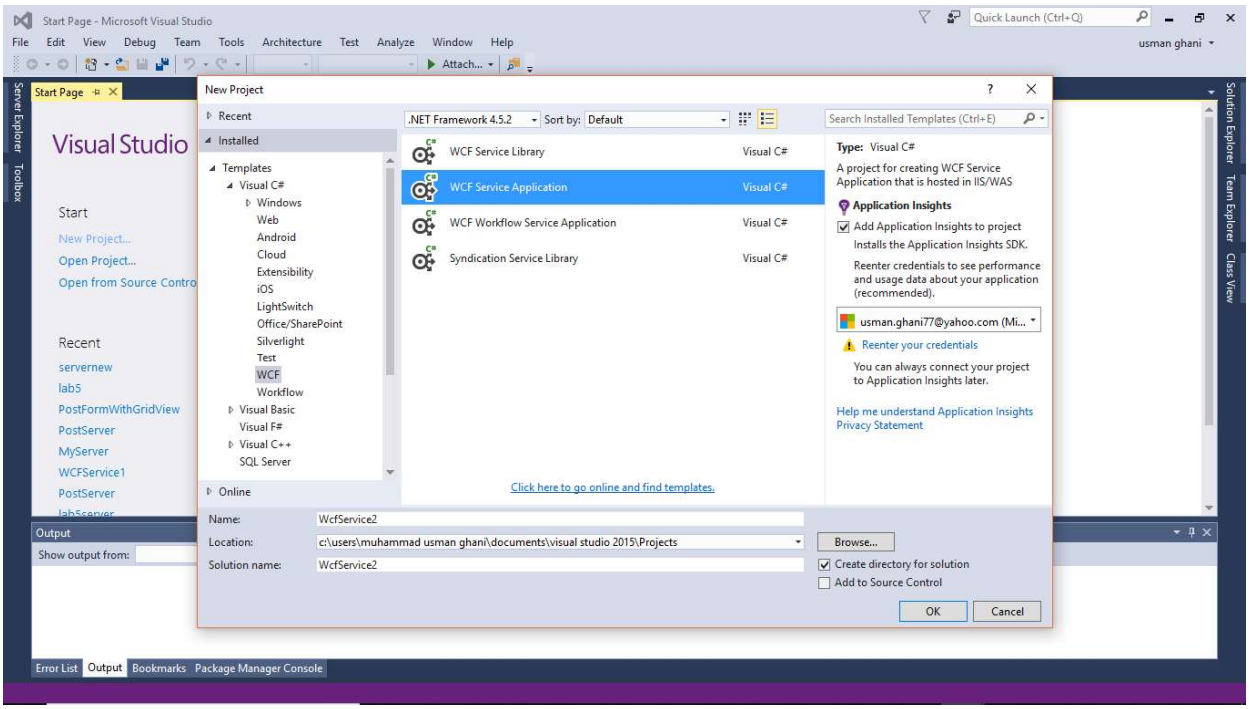
Make User interface:

1. Textbox, labels, button, Grid view,selectionbox



Step 2:

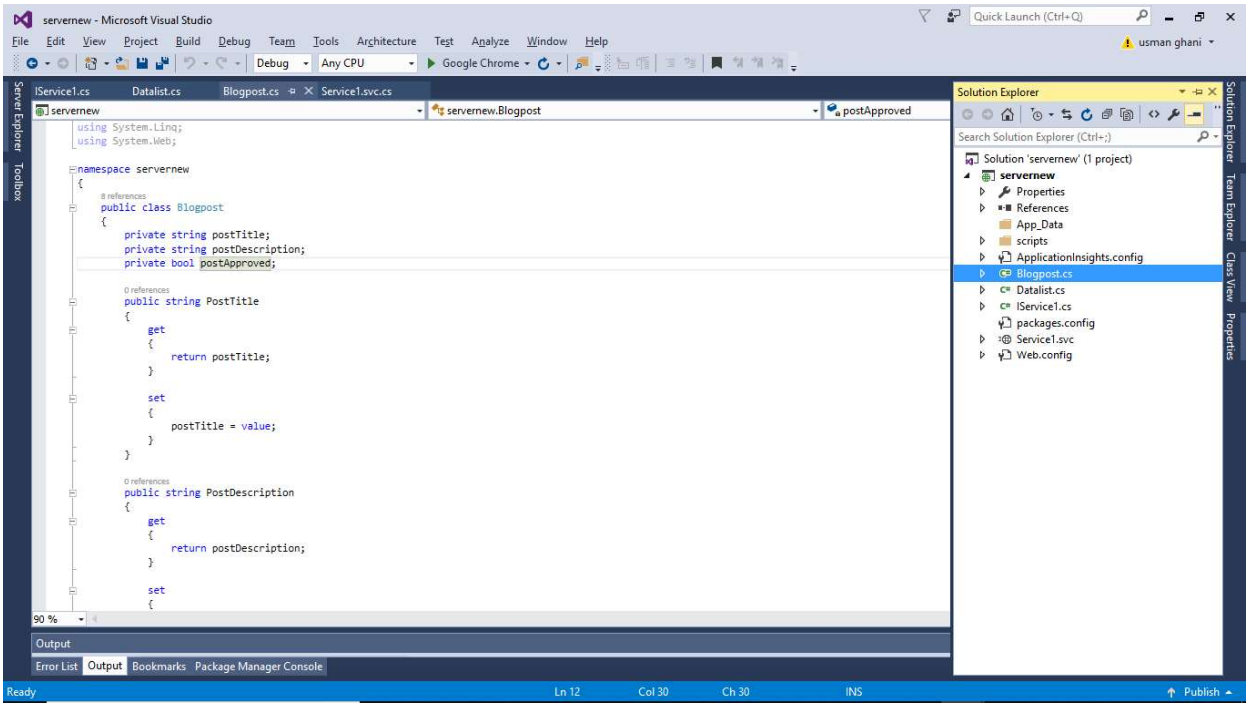
Create new instance of visual studio and Create WCF.



Step 2:

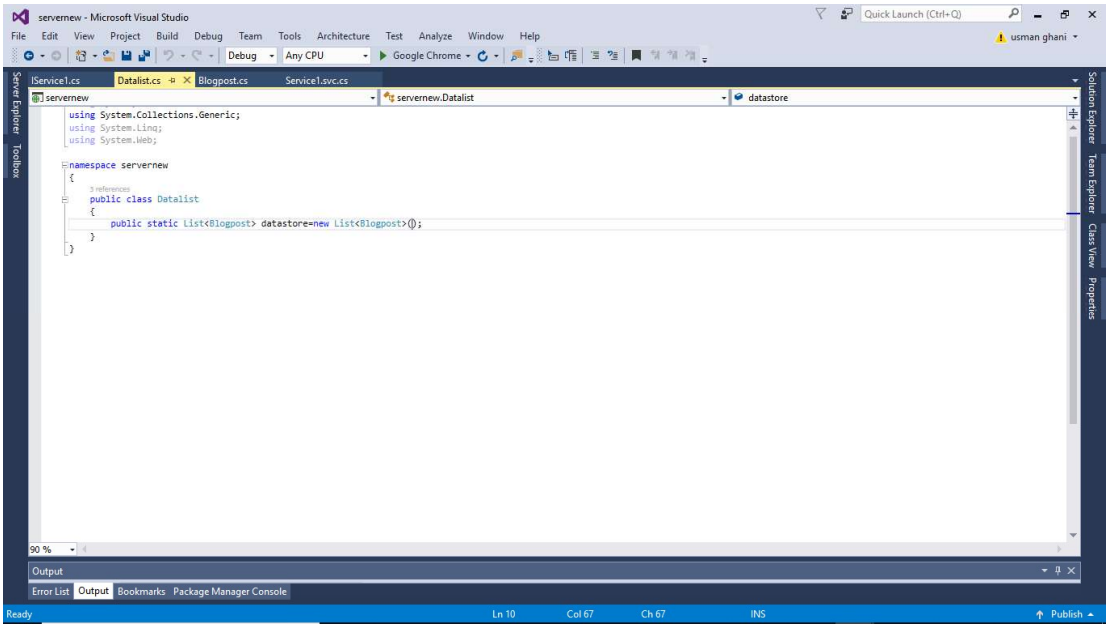
Add class

Define variables and perform Encapsulation



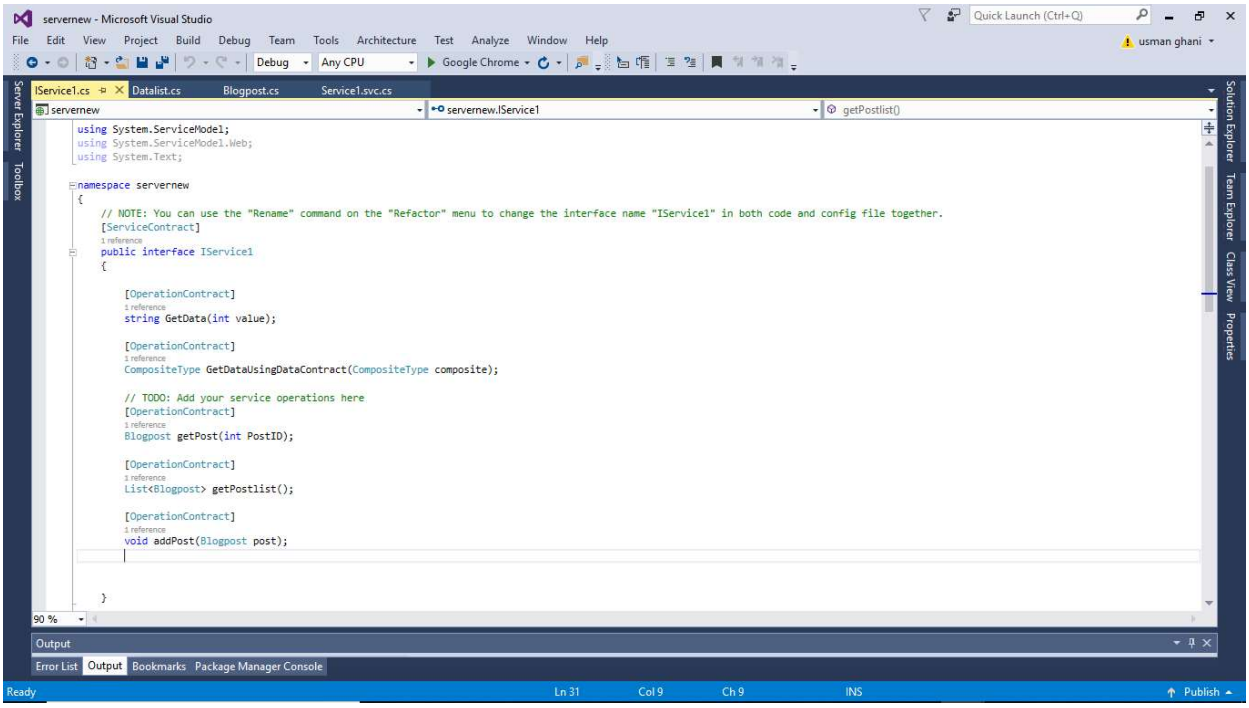
Step 3:

Create list that will store objects of above class



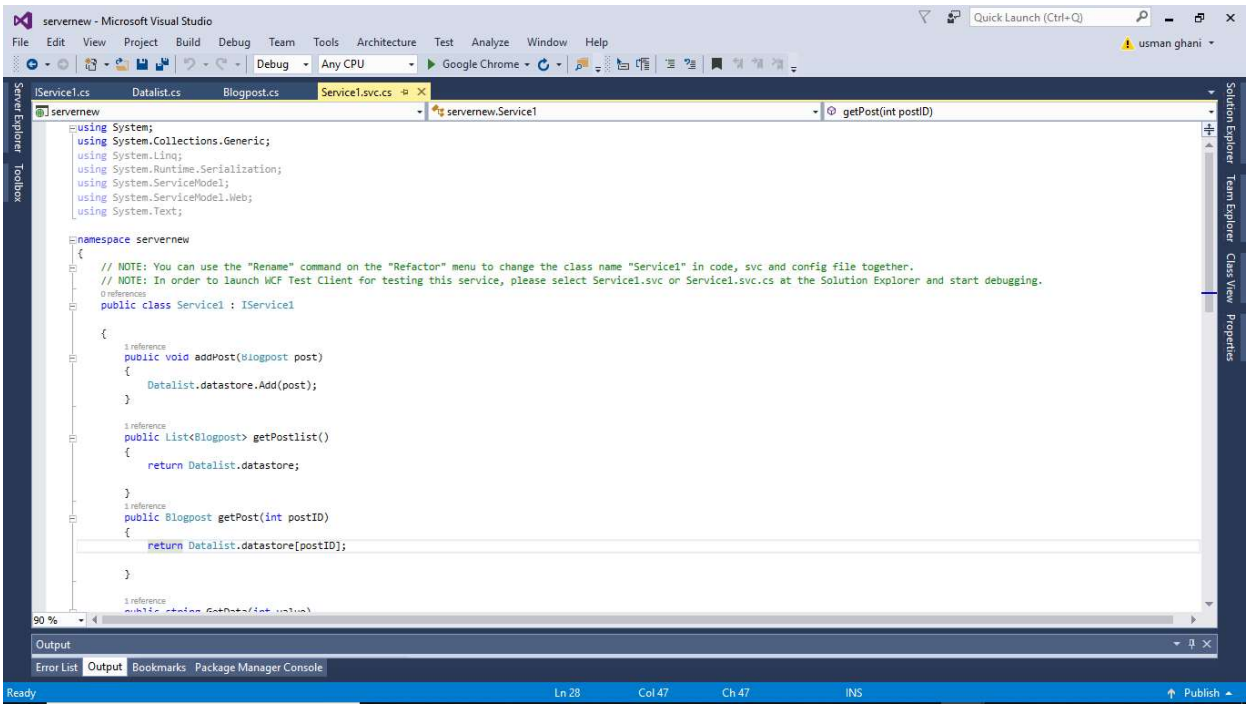
Step 4:

Function Prototype in interface class(Iservice1.cs)



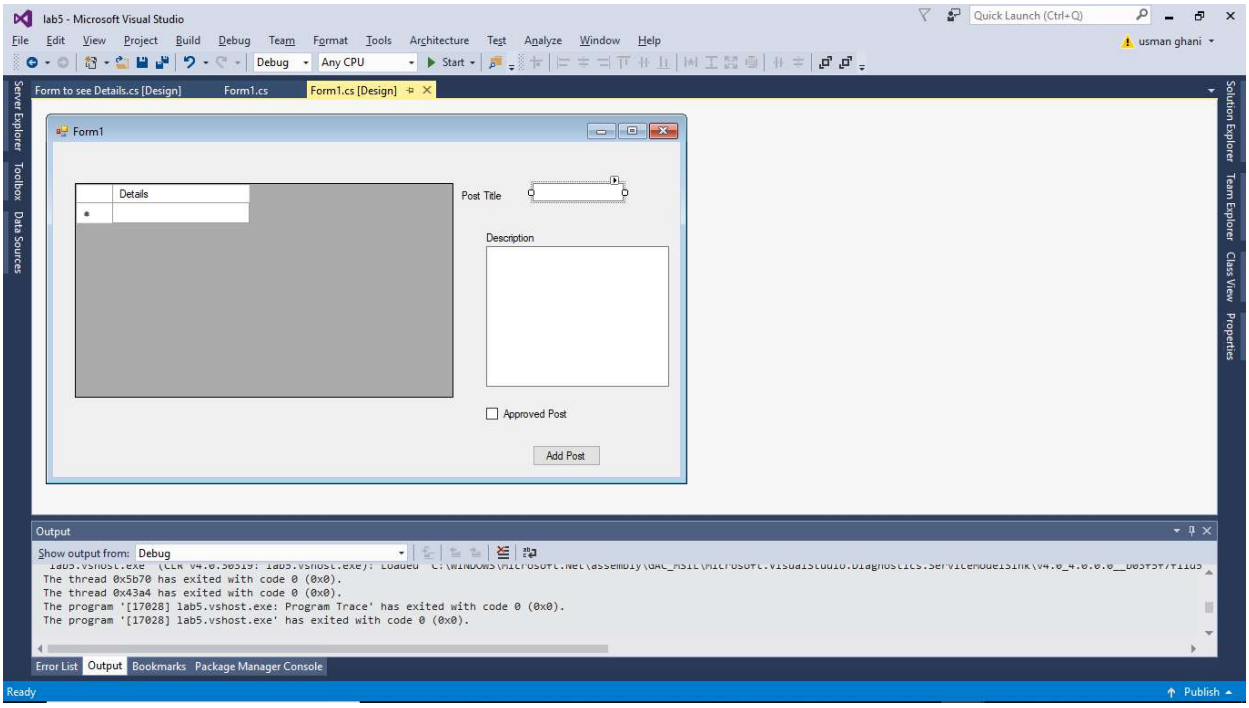
Step 5:

Function Definitions in Service1.scv.css



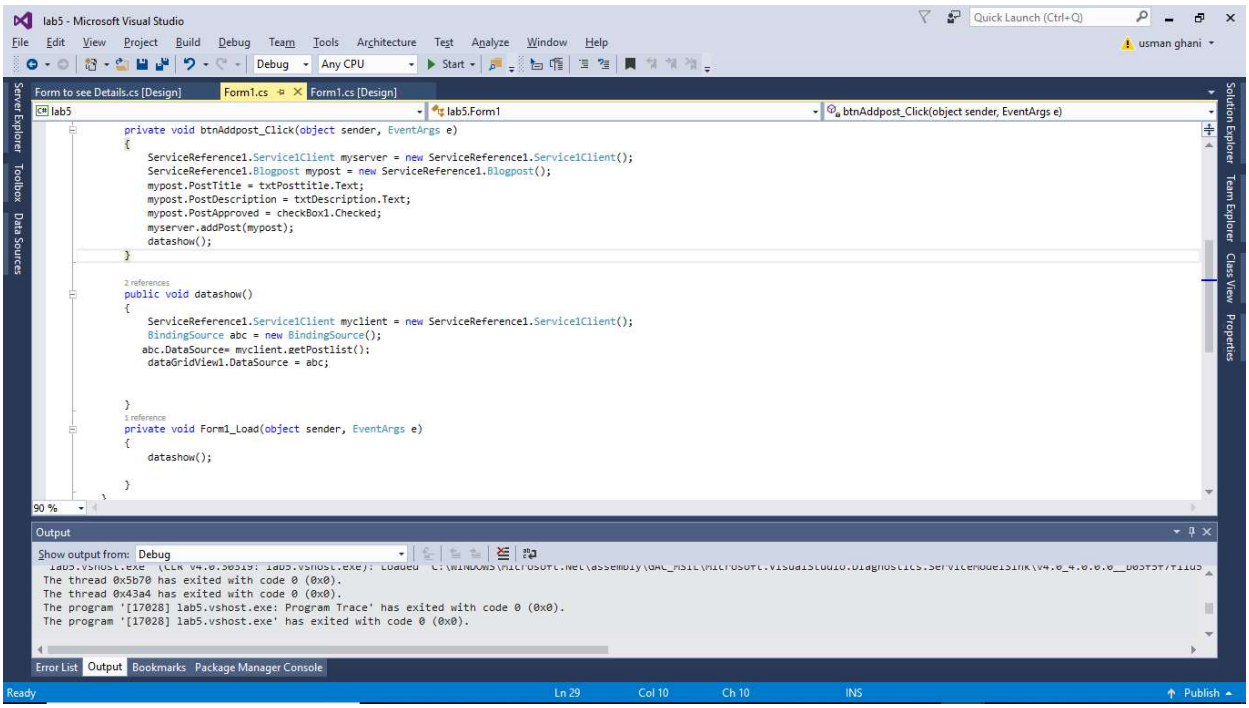
Step 6:

Getting data From client side:

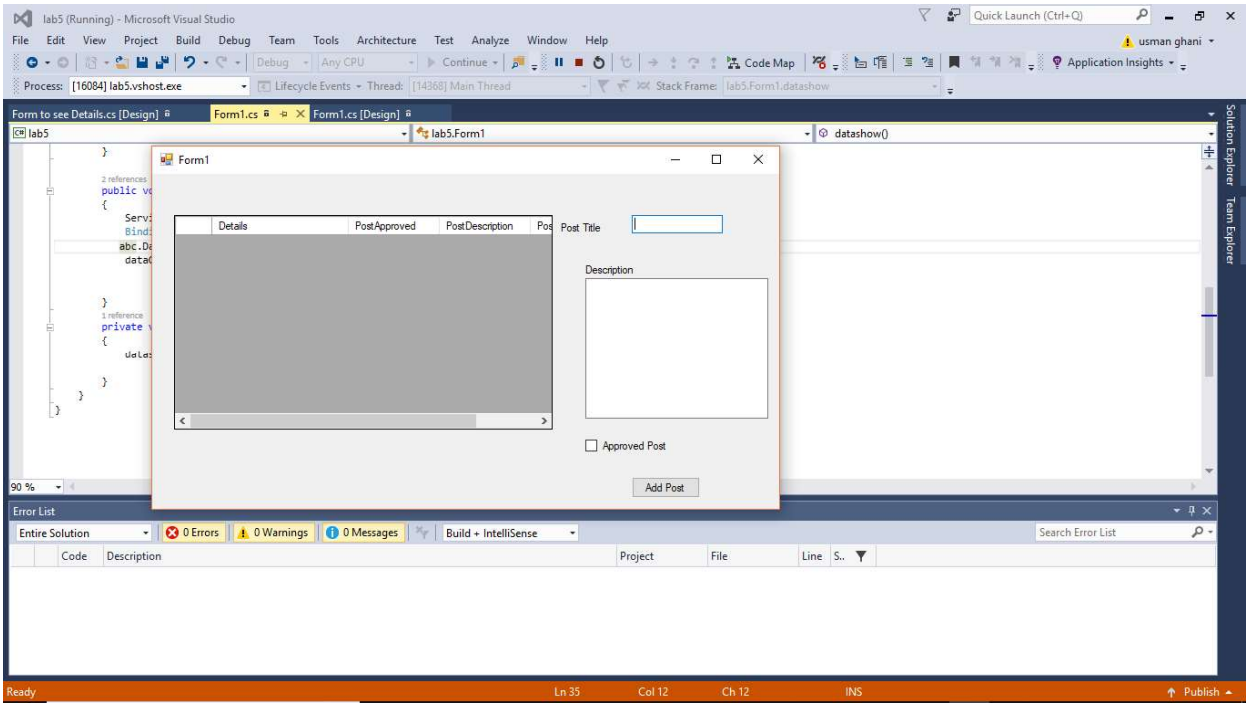


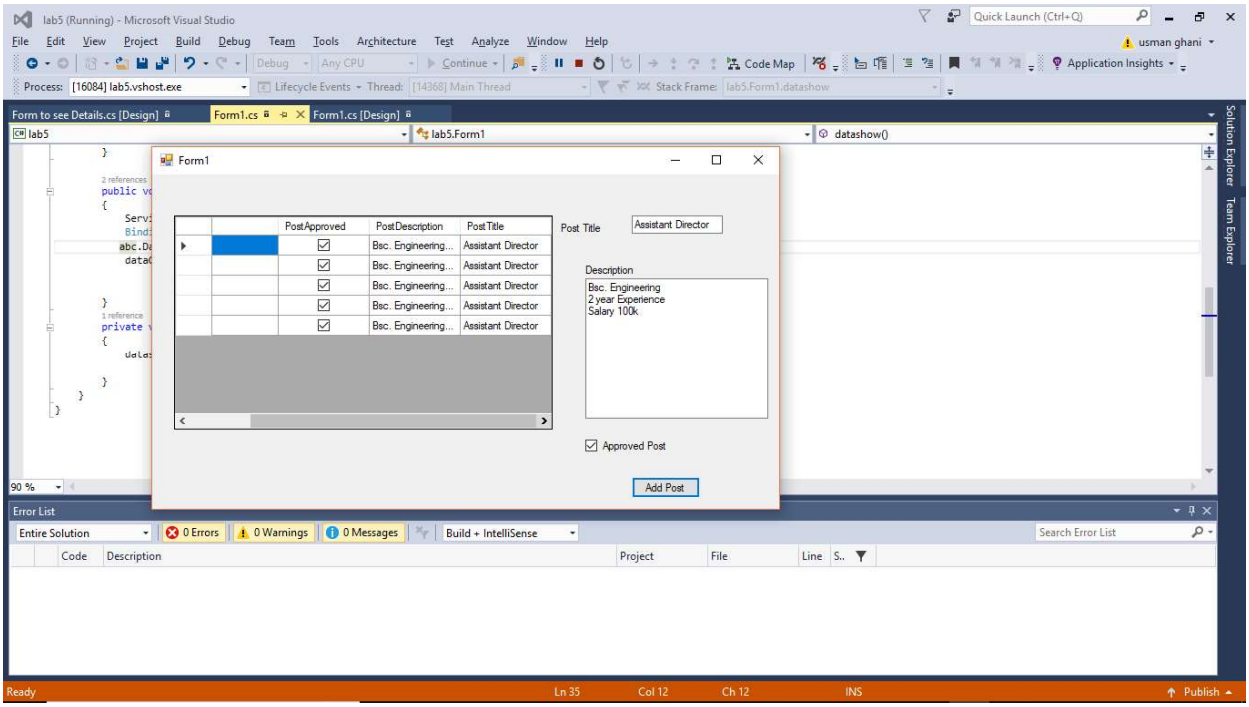
Step 7:

Double Click on “Add post” and add following code



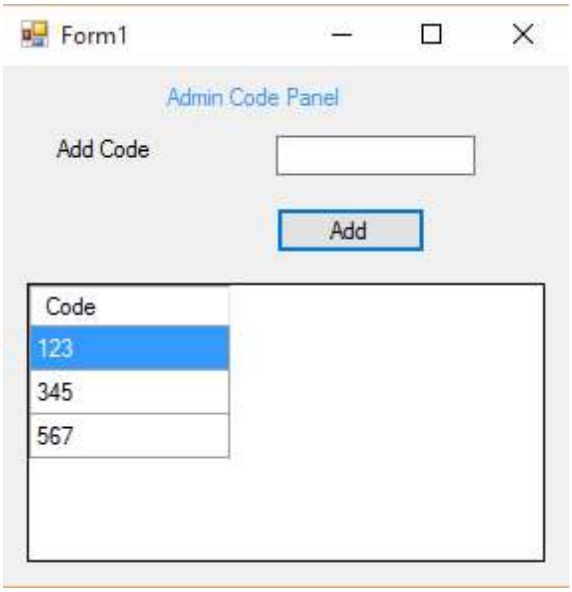
Output:


lab5 (Running) - Microsoft Visual Studio
Form1
Details
PostApproved
PostDescription
Post Title
Description
Approved Post
Add Post
Error List
Ready


lab5 (Running) - Microsoft Visual Studio
Form1
PostApproved
PostDescription
PostTitle
Post Title
Assistant Director
Description
Bsc. Engineering
2 year Experience
Salary 100k
Approved Post
Add Post
Error List
Ready

Develop two panel in C# as one for admin and one for user, admin enter codes using first form and codes entered in field will be shown in grid box. Admin will also enter campus, department and program using second form. User login using codes saved by admin in the first form after successful login campus, department and program entered by admin will be shown to user which he can chose to register himself/ herself. The options selected by user will be shown in grid box. (also validate form against empty as well as wrong input and show message if user successfully registered).

Form 1:

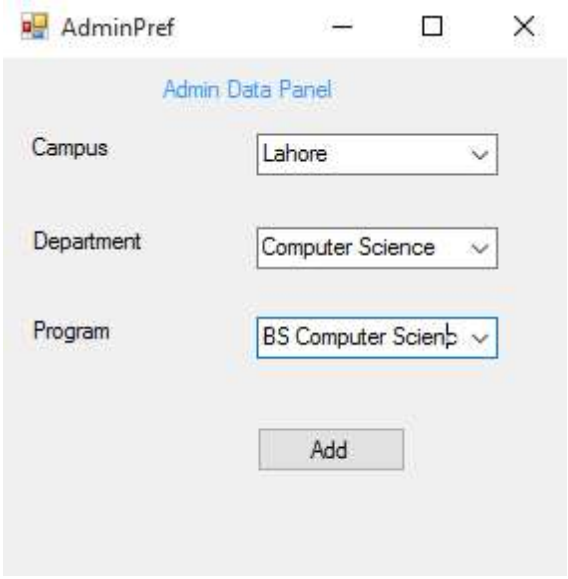


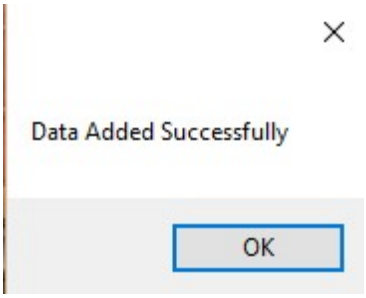
Code:

```
private void button1_Click(object sender, EventArgs e)
    {
      DTO dto = new DTO();
      dto.Code = textcode.Text;

      if (textcode.Text != "")
      {
        DL.addcodes(dto);

        codeslayout.DataSource = null;
        codeslayout.DataSource = DL.codes;

        codeslayout.Columns[1].Visible = false;
        codeslayout.Columns[2].Visible = false;
        codeslayout.Columns[3].Visible = false;
        codeslayout.Columns[4].Visible = false;
        codeslayout.Columns[5].Visible = false;
        codeslayout.Columns[6].Visible = false;
        codeslayout.Columns[7].Visible = false;
        textcode.Text = "";

      }
      else
      {
        MessageBox.Show("Please Fill All Values");
      }
```

}

Form 2:

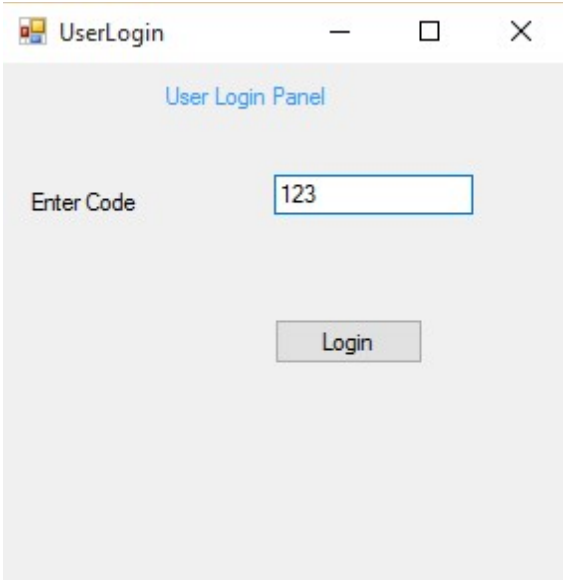




Code:

```
private void button1_Click(object sender, EventArgs e)
    {
      DTO dto = new DTO();
      dto.Campus = textcampus.Text;
      dto.Department = textdepartment.Text;
      dto.Program = textprogram.Text;


      if (textcampus.Text != "" && textdepartment.Text != "" && textprogram.Text != "")
      {
        DL.addadminpref(dto);
        MessageBox.Show("Data Added Successfully");
        textcampus.Text = "";
        textdepartment.Text = "";
        textprogram.Text = "";
      }
      else
      {
        MessageBox.Show("Please Fill All Values");
      }
    }
```

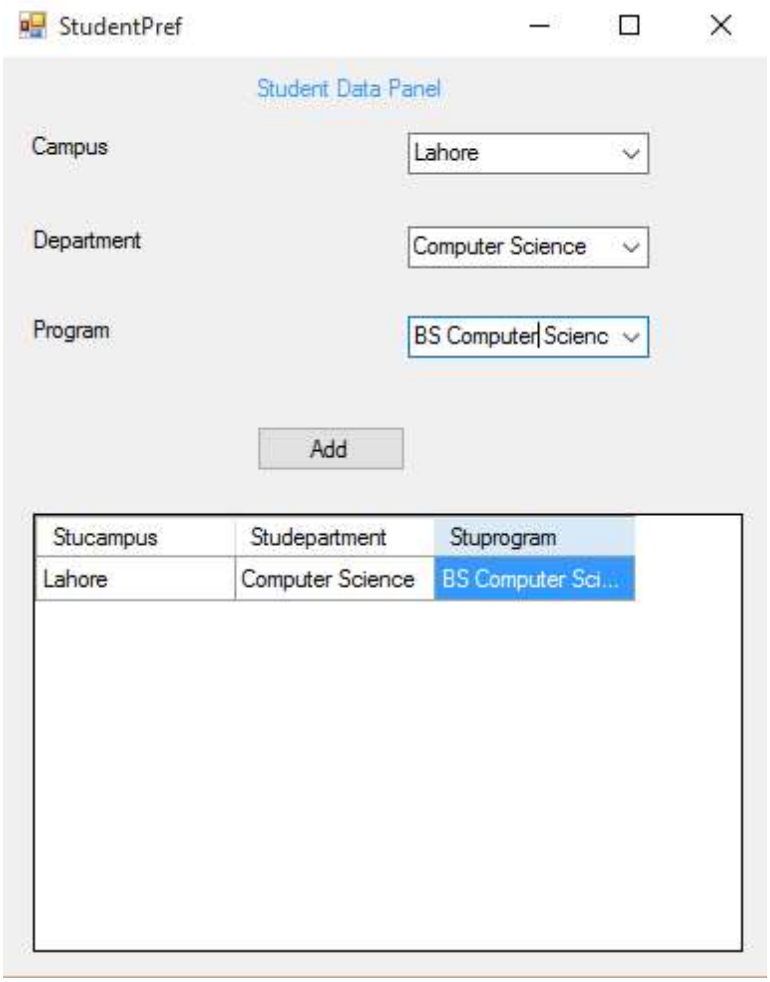
Form 3:



Code:

```
private void button1_Click(object sender, EventArgs e)
    {
      DTO dto = new DTO();
      dto.Usercode = textusercode.Text;
      dto = DL.confirmuser(dto);
      if (dto != null)
      {
        StudentPref stdpref = new StudentPref(dto.Campus,dto.Department,dto.Program);

        stdpref.Show();


      }
      else
        MessageBox.Show("Invalid user");
    }
```

Form 4:



Code:

```
public StudentPref(String dept,String camp,String prog)
    {

        InitializeComponent();
        studtextcampus.Text = camp;
        studtextdepartment.Text = dept;
        studtextprogram.Text = prog;
    }

    private void button1_Click(object sender, EventArgs e)
    {
        DTO dto = new DTO();
        dto.Stucampus = studtextcampus.Text;
        dto.Studepartment = studtextdepartment.Text;
        dto.Stuprogram = studtextprogram.Text;
        if(studtextcampus.Text != "" && studtextdepartment.Text != "" && studtextprogram.Text != "")
        {
            DL.addstudentpref(dto);
            StudentGrid.DataSource = null;
            StudentGrid.DataSource = DL.studentpref;

            StudentGrid.Columns[0].Visible = false;
            StudentGrid.Columns[1].Visible = false;
```

```
            StudentGrid.Columns[2].Visible = false;
            StudentGrid.Columns[3].Visible = false;
            StudentGrid.Columns[4].Visible = false;

            studtextcampus.Text = "";
            studtextdepartment.Text = "";
            studtextprogram.Text = "";
        }
        else
        {
            MessageBox.Show("Please Fill All Values");
        }


    }
  }
```

Other two classes are as follows:

1. **DTO:**

```
public class DTO{
    public DTO()
    {
    }

    private String code;
    private String campus;
    private String department;
    private String program;
    private String usercode;
    private String stucampus;
    private String studepartment;
    private String stuprogram;

    public string Code
    {
        get
        {
            return code;
        }

        set
        {
            code = value;
        }
    }

    public string Campus
    {
        get
        {
            return campus;
        }
```

```
        set
        {
            campus = value;
        }
    }

    public string Department
    {
        get
        {
            return department;
        }

        set
        {
            department = value;
        }
    }

    public string Program
    {
        get
        {
            return program;
        }

        set
        {
            program = value;
        }
    }

    public string Usercode
    {
        get
        {
            return usercode;
        }

        set
        {
            usercode = value;
        }
    }

    public string Stucampus
    {
        get
        {
            return stucampus;
        }
```

```
        set
        {
            stucampus = value;
        }
    }

    public string Studepartment
    {
        get
        {
            return studepartment;
        }

        set
        {
            studepartment = value;
        }
    }

    public string Stuprogram
    {
        get
        {
            return stuprogram;
        }

        set
        {
            stuprogram = value;
        }
    }
}
```

2. DL:

```
public class DL
{
    public DL()
    {
    }

    public static ArrayList adminpref = new ArrayList();
    public static ArrayList codes= new ArrayList();
    public static ArrayList studentpref = new ArrayList();

    public static void addcodes(DTO dto)
    {
        codes.Add(dto);
    }

    public static void addadminpref(DTO dto)
    {
        adminpref.Add(dto);
    }
```

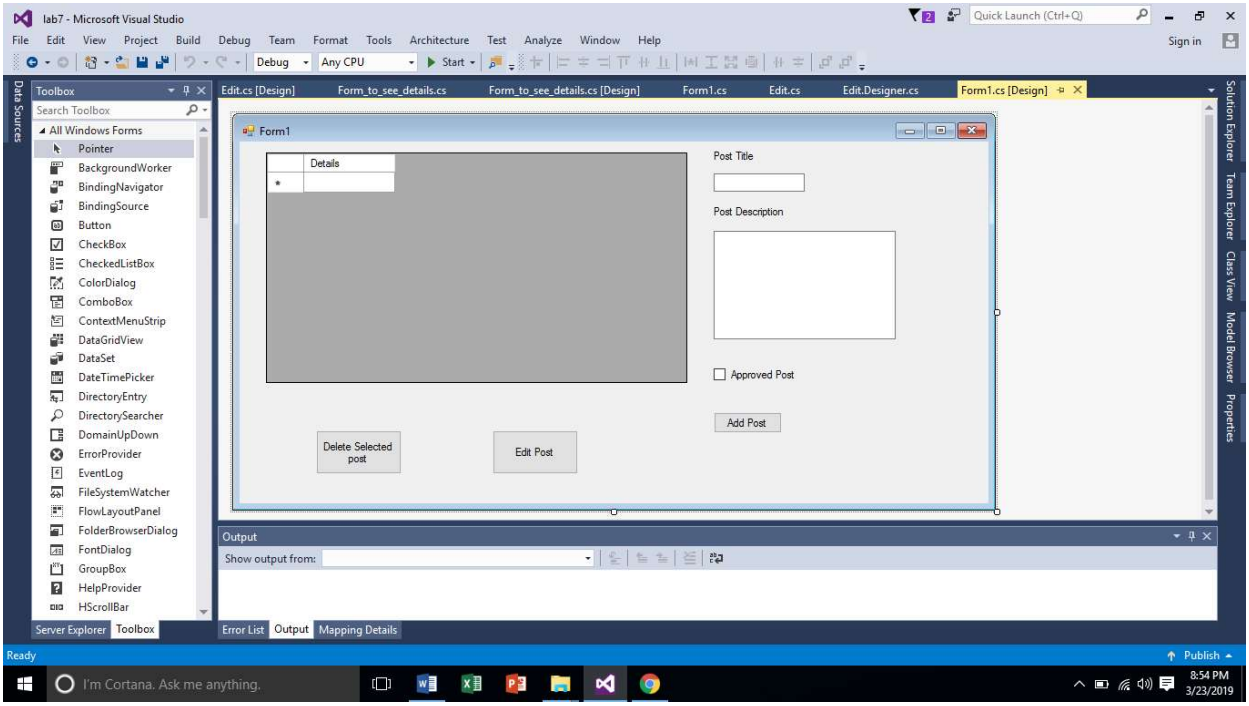
```
    public static void addstudentpref(DTO dto)
    {
        studentpref.Add(dto);
    }
    public static DTO confirmuser(DTO dto)
    {
        foreach (DTO d in codes)
            if (d.Code.Equals(dto.Usercode))
                return d;
        return null;
    }
}
```

Delete:

Method 1:

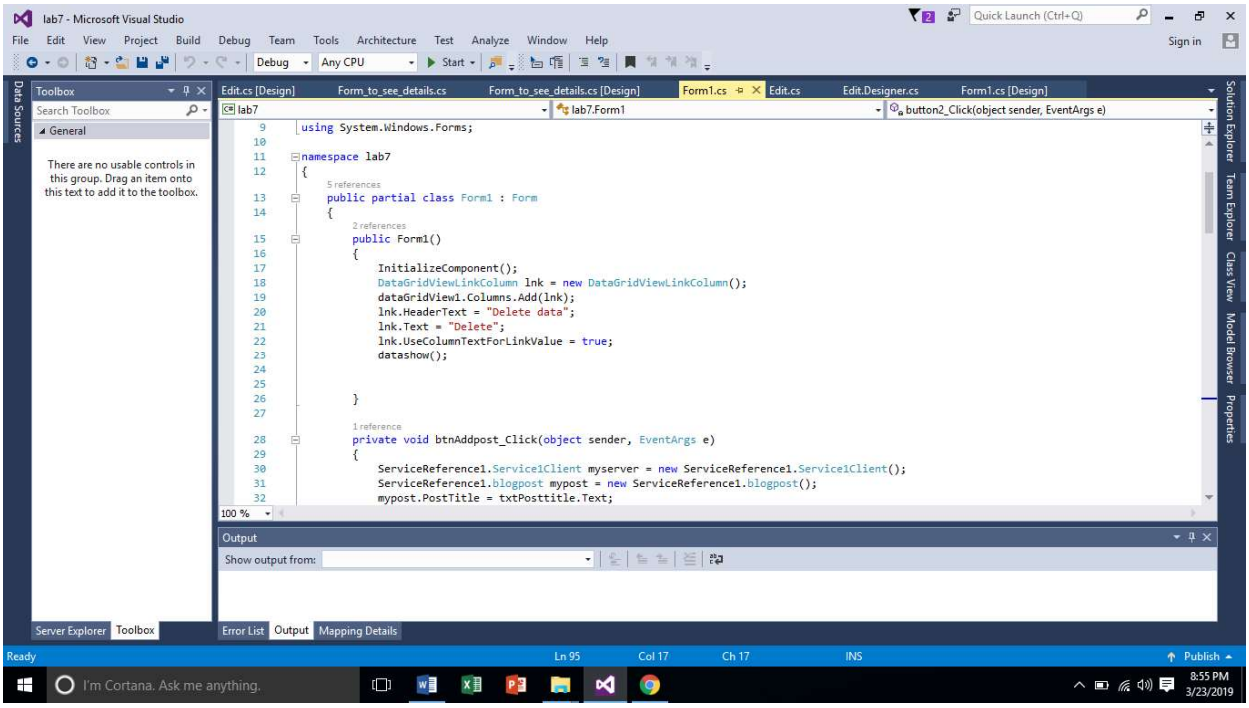
Extend Lab 5 and add Colum in DataGridview manually

Or You can add Column by Clicking on datagridview

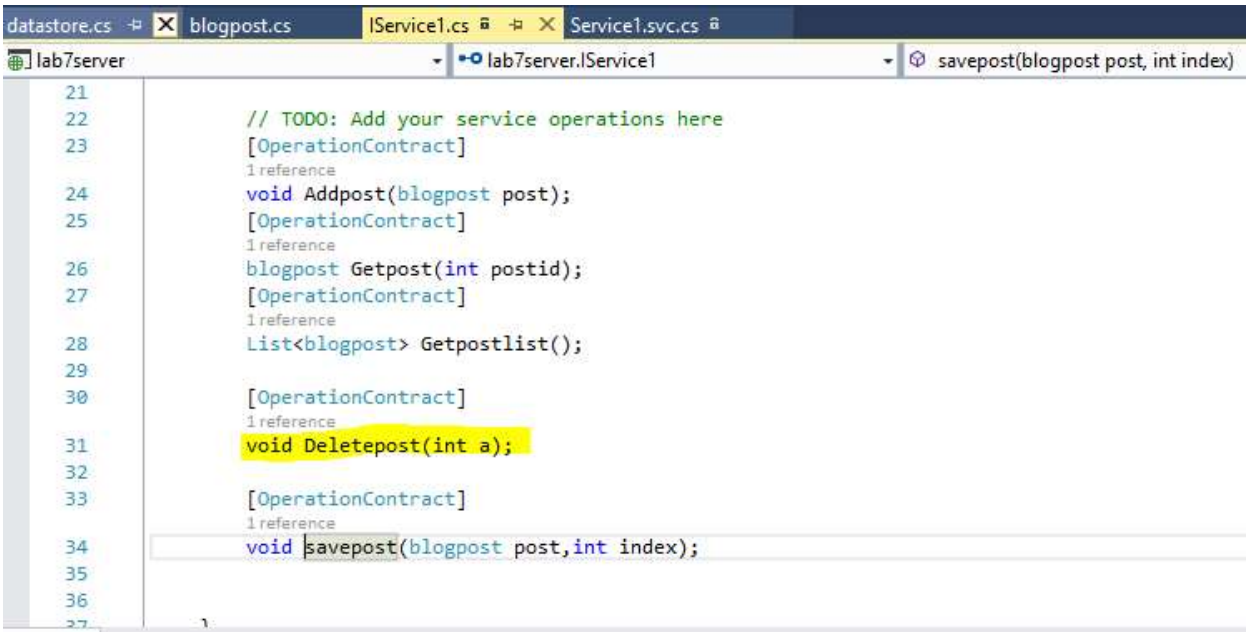


Step 1:

Make Column(For Delete) in Constructor of Form1(Mentioned above)

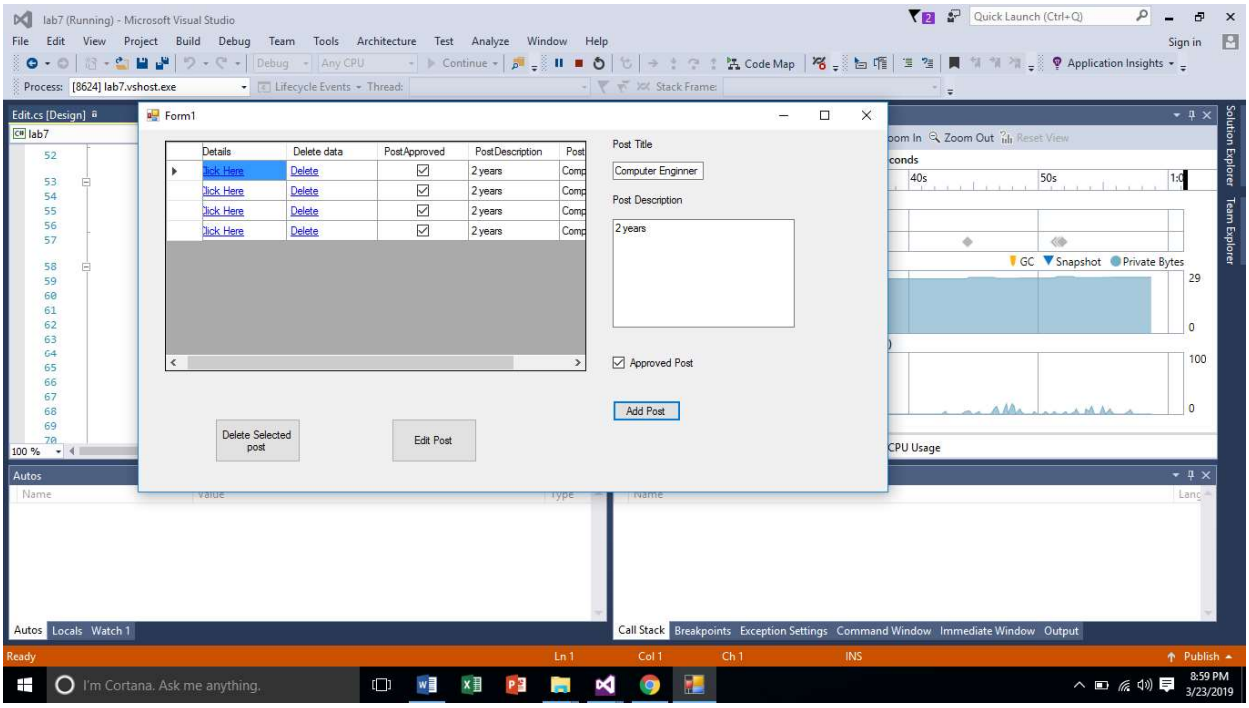


"Deletepost" function prototype(on server side):



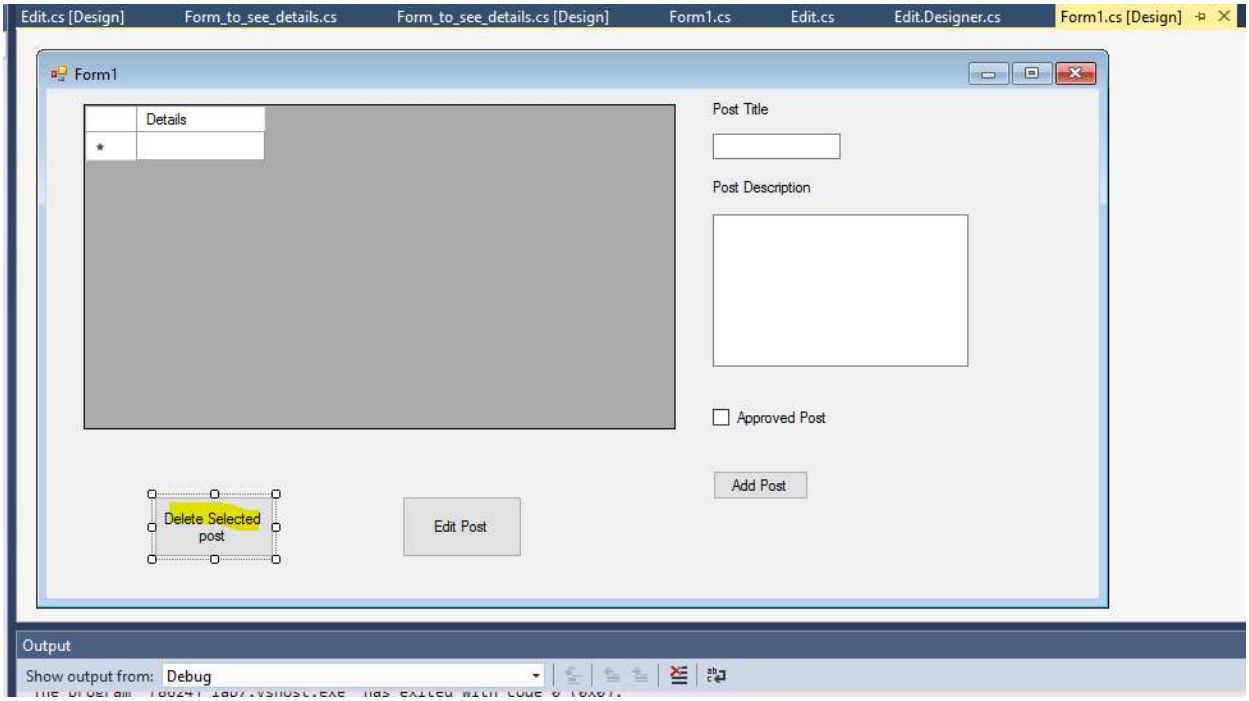
Function Definition:

```
datastore.cs   blogpost.cs   IService1.cs   Service1.svc.cs
lab7server   lab7server.Service1   IService1.savepost(blogpost post,
29          }
30
            1 reference
31          public string GetData(int value)
32          {
33              return string.Format("You entered: {0}", value);
34          }
35
            1 reference
36          public void Deletepost(int a)
37          {
38              datastore.data.RemoveAt(a);
39          }
40
            0 references
41          void savepost(blogpost post, int index)
42          {
43              datastore.data.Insert(index, post);
44
45          }
46
```

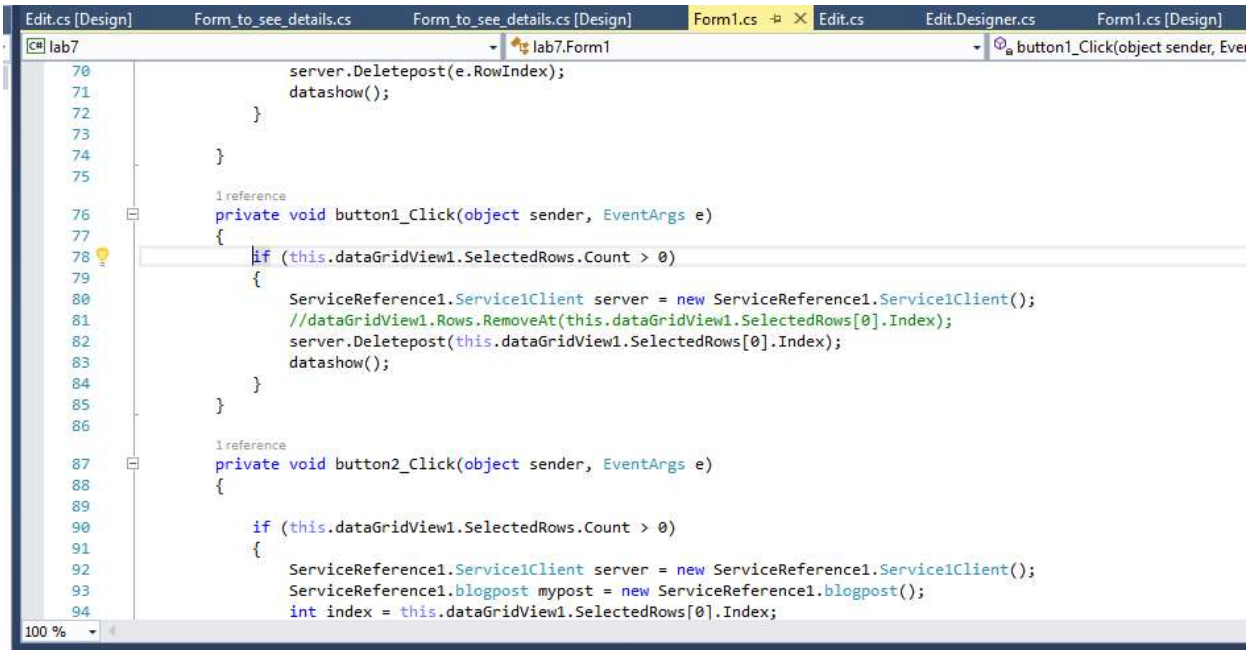


2nd Method:

Make button "Delete Selected Post"



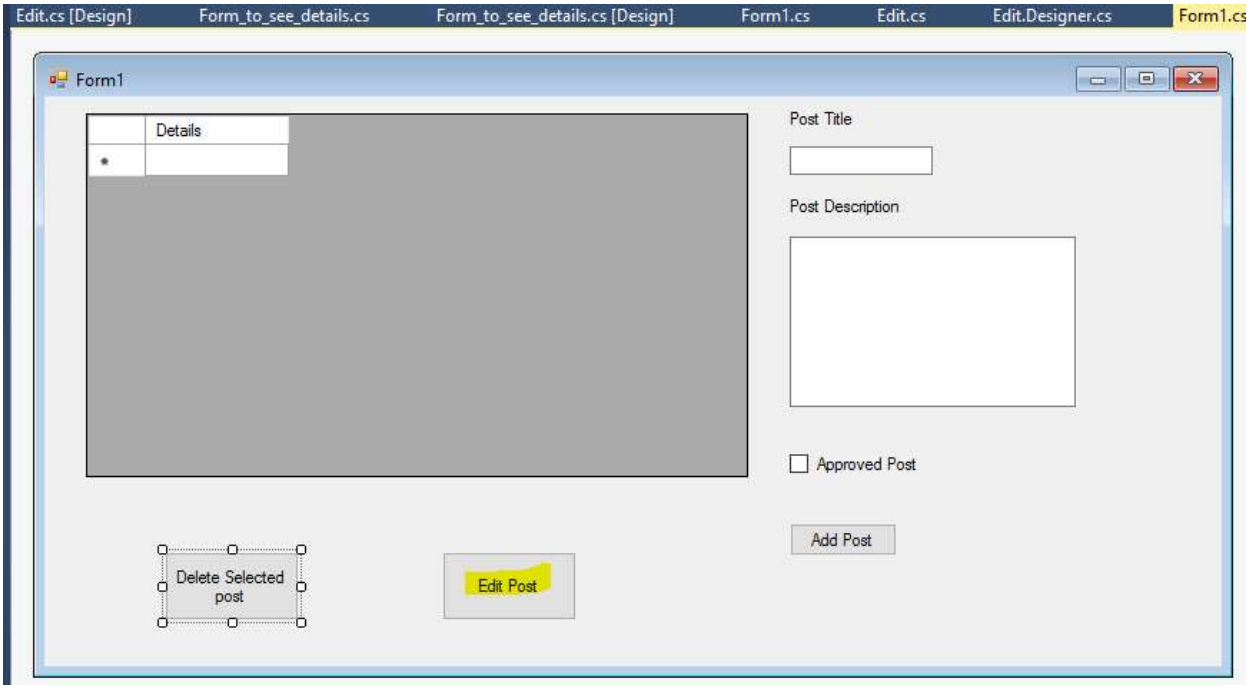
Code behind Delete Button:



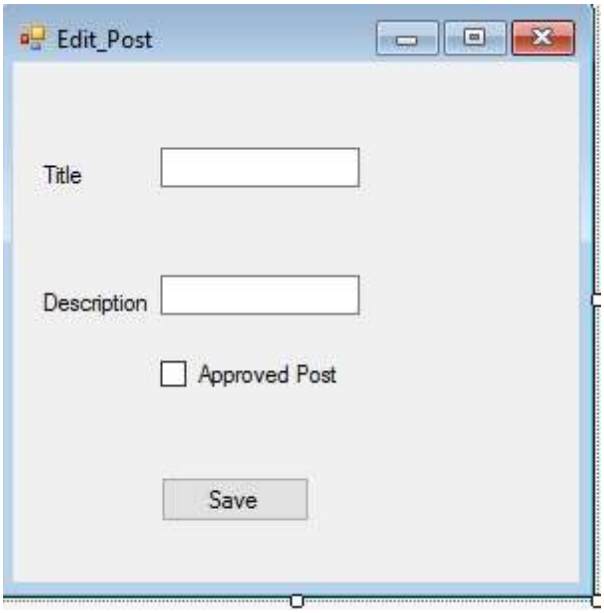
First Select row you want to delete and then press delete button.

Task 2:

Edit Post (add button to edit post)

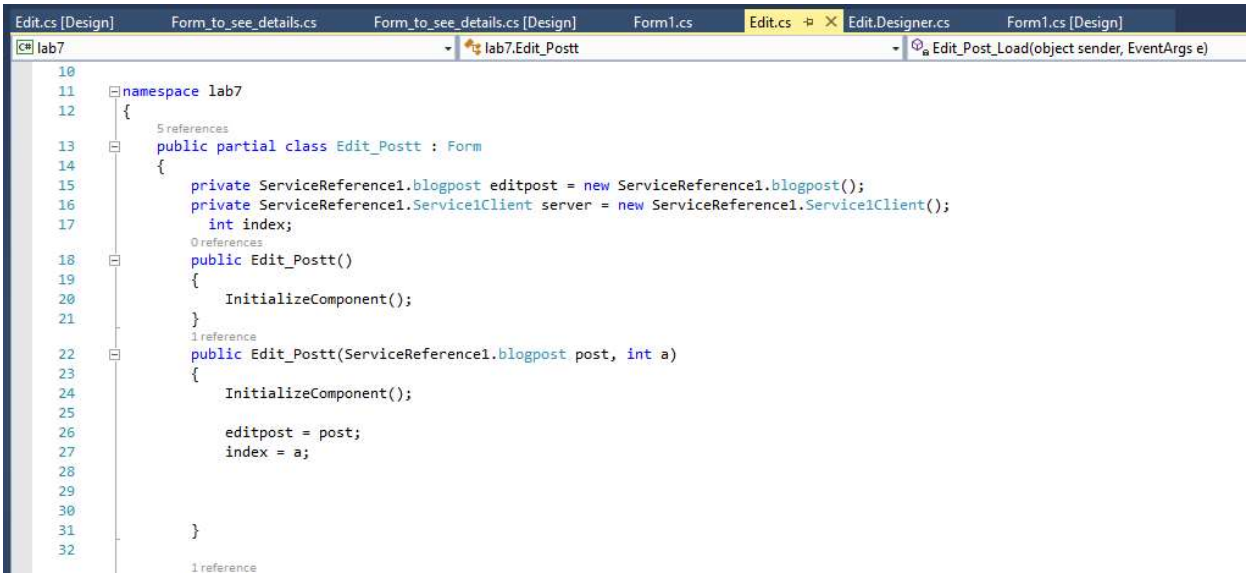


Add 3rd form to edit post (it show the data of selected post you want to edit)



Make new constructor that contain two variables one for post and 2nd for post index

Two global variable that will store edit post and row index so that it store in list on same index.




```
10
11  namespace lab7
12  {
        5 references
13      public partial class Edit_Postt : Form
14      {
15          private ServiceReference1.blogpost editpost = new ServiceReference1.blogpost();
16          private ServiceReference1.Service1Client server = new ServiceReference1.Service1Client();
17            int index;
              0 references
18          public Edit_Postt()
19          {
20              InitializeComponent();
21          }
            1 reference
22          public Edit_Postt(ServiceReference1.blogpost post, int a)
23          {
24              InitializeComponent();
25
26              editpost = post;
27              index = a;
28
29
30
31          }
32
                1 reference
```

Code behind edit post:

```
        1 reference
        private void button2_Click(object sender, EventArgs e)
        {
            if (this.dataGridView1.SelectedRows.Count > 0)
            {
                ServiceReference1.Service1Client server = new ServiceReference1.Service1Client();
                ServiceReference1.blogpost mypost = new ServiceReference1.blogpost();
                int index = this.dataGridView1.SelectedRows[0].Index;
                mypost = server.Getpost(index);
                Edit_Postt editpost = new Edit_Postt(mypost ,index);
               // this.Close();
                editpost.Show();
            }
        }
    }
```

Code behind save button on edit form:

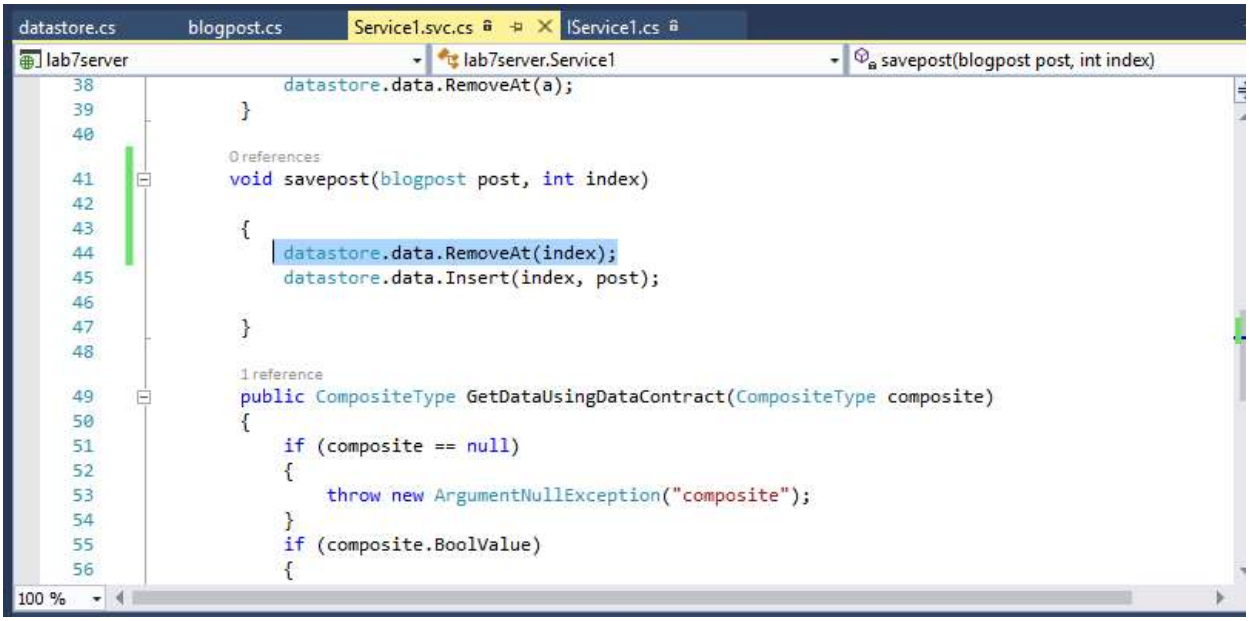
```
lab7                                    lab7.Edit_Postt
31          }
32
            1 reference
33          private void Edit_Post_Load(object sender, EventArgs e)
34          {
35              textBox1.Text = editpost.PostTitle;
36              textBox2.Text = editpost.PostDescription;
37              checkBox1.Checked = editpost.PostApproved;
38          }
39
            1 reference
40          private void button1_Click(object sender, EventArgs e)
41          {
42              editpost.PostTitle = textBox1.Text;
43              editpost.PostDescription = textBox2.Text;
44              editpost.PostApproved = checkBox1.Checked;
45              server.Deletepost(index);
46              server.savepost(editpost,index);
47              this.Close();
48              Form1 firstform = new Form1();
49              firstform.Show();
50          }
51      }
52  }
53
```

"Savepost" function on server side:

Prototype:

```
datastore.cs   blogpost.cs   Service1.svc.cs   IService1.cs
lab7server                  lab7server.IService1        savepost(blogpost post, int index)
            1 reference
28          List<blogpost> Getpostlist();
29
30          [OperationContract]
            1 reference
31          void Deletepost(int a);
32
33          [OperationContract]
            1 reference
34          void savepost(blogpost post,int index);
35
36
37      }
38
39
40      // Use a data contract as illustrated in the sample below to add composite types to service c
41      [DataContract]
            4 references
42      public class CompositeType
43      {
44          bool boolValue = true;
```

Server side save function:

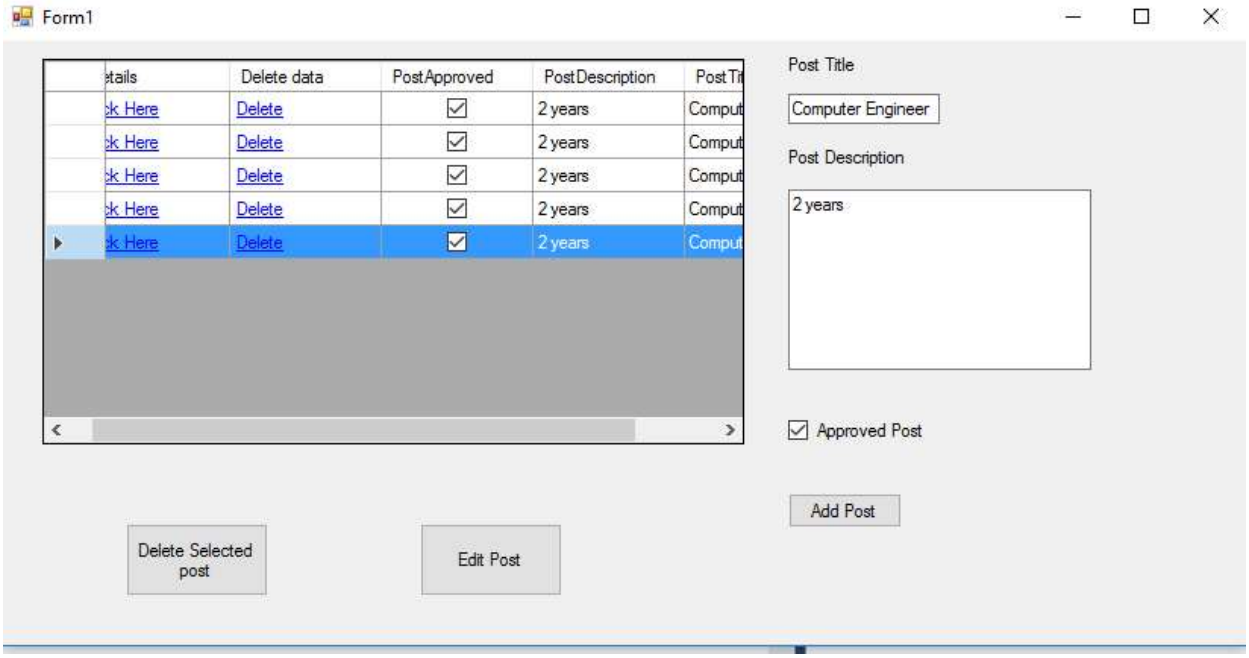


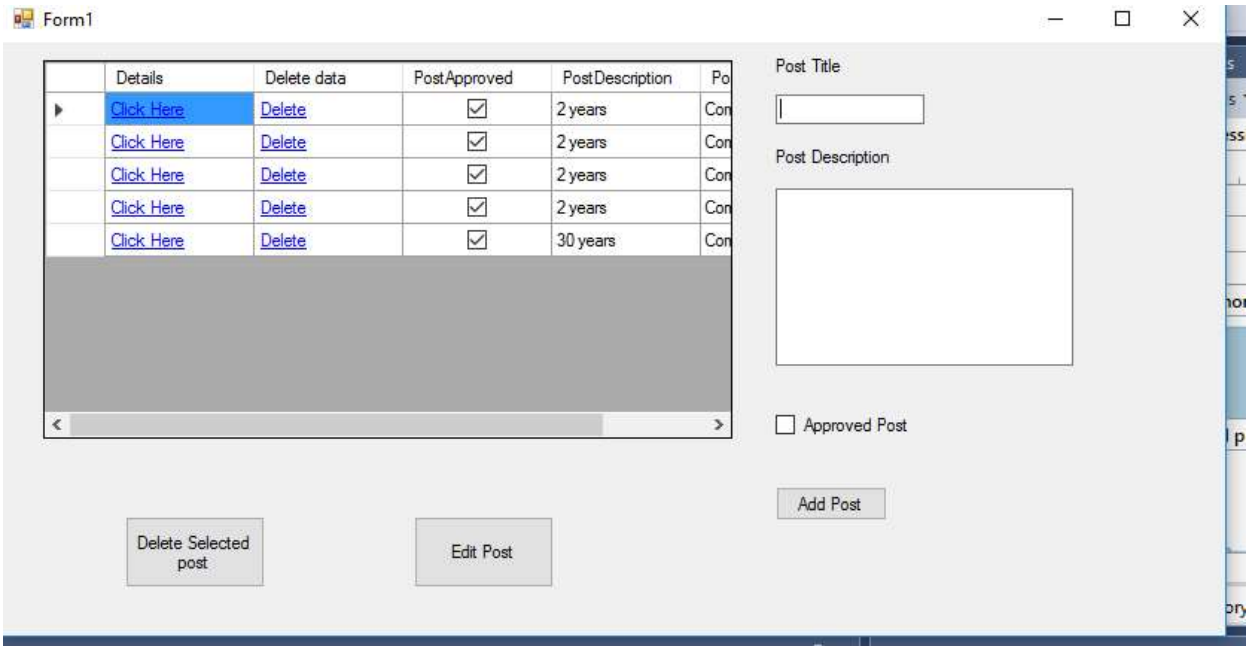
Prototype:

Datastore is class name ,data is list name and insert is a method that insert value in particular index.

Output:

Select post:


Form1
tails
Delete data
PostApproved
PostDescription
PostTi
ck Here
Delete
2 years
Comput
Post Title
Computer Engineer
Post Description
2 years
Approved Post
Add Post
Delete Selected post
Edit Post


Form1
Details
Delete data
PostApproved
PostDescription
Po
Click Here
Delete
2 years
30 years
Post Title
Post Description
Approved Post
Add Post
Delete Selected post
Edit Post