Faculty of Information Technology
University of Moratuwa

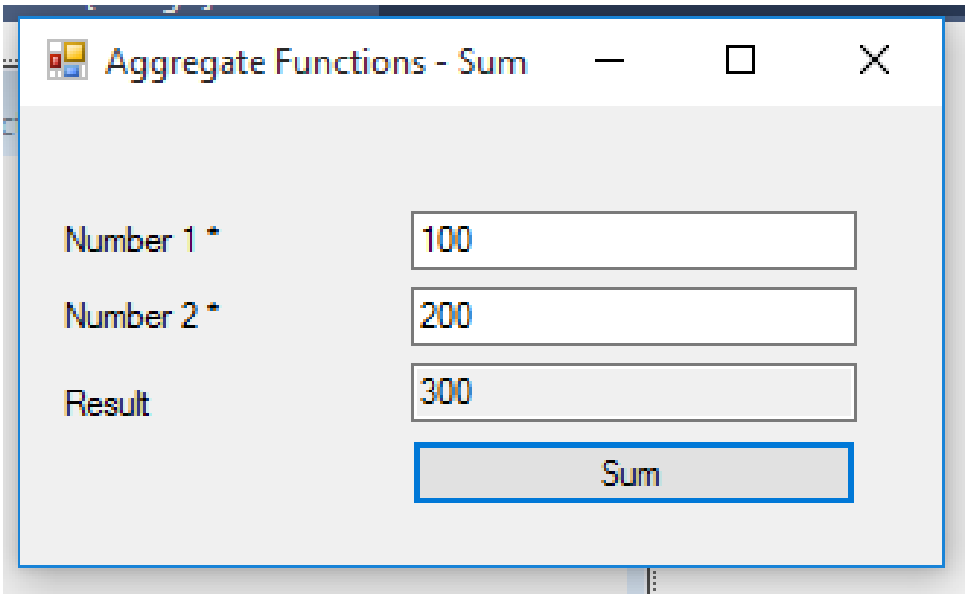
Degree of Bachelor of Information Technology (BIT)External

ITE1112: Visual Application Programming – Activity 1

Learning Objectives: Familier with fundamentals in visual application development

Question 1: Create the simple application to implement an Aggregation Function-Sum of the two numbers and display result.

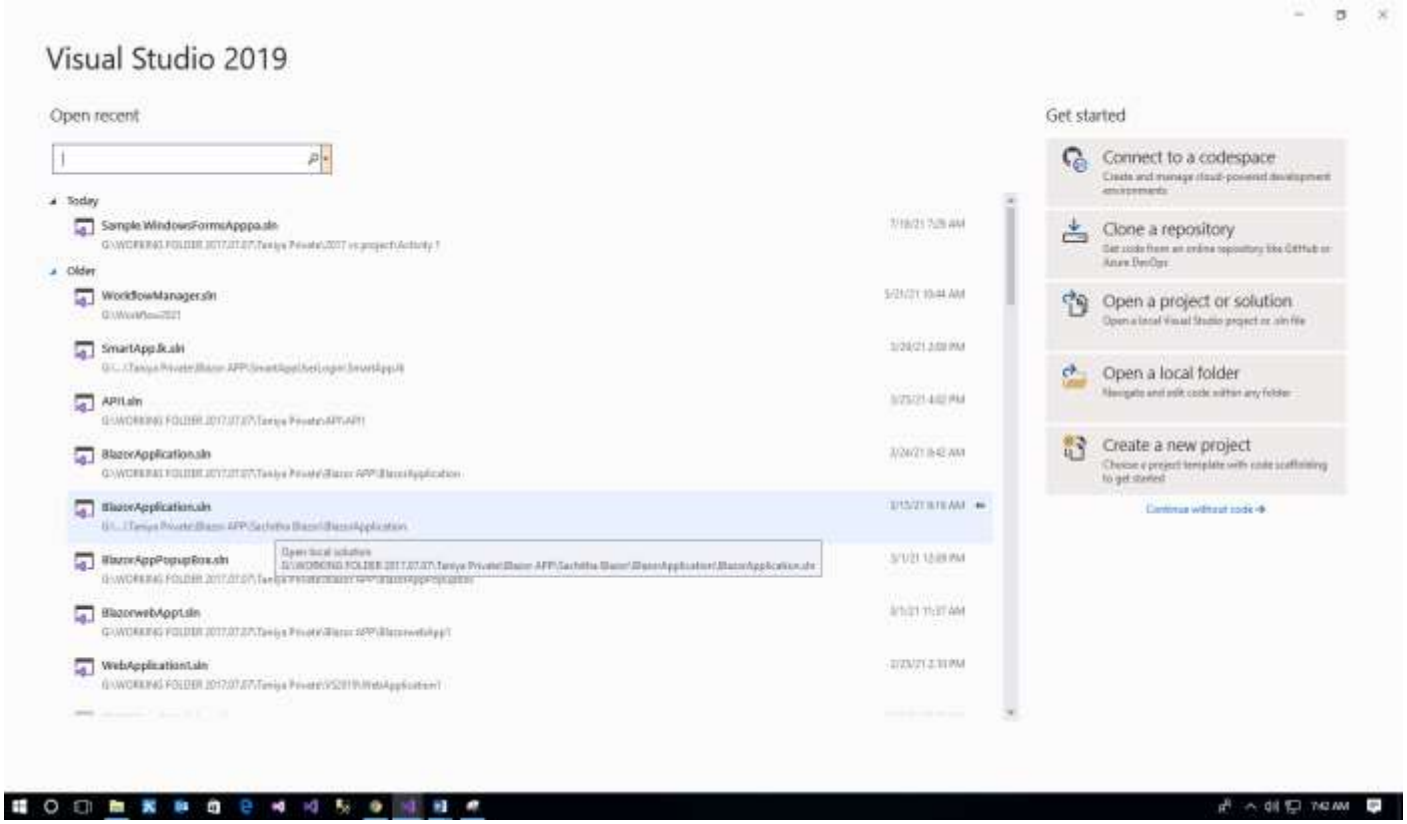
Your final output should be as below.



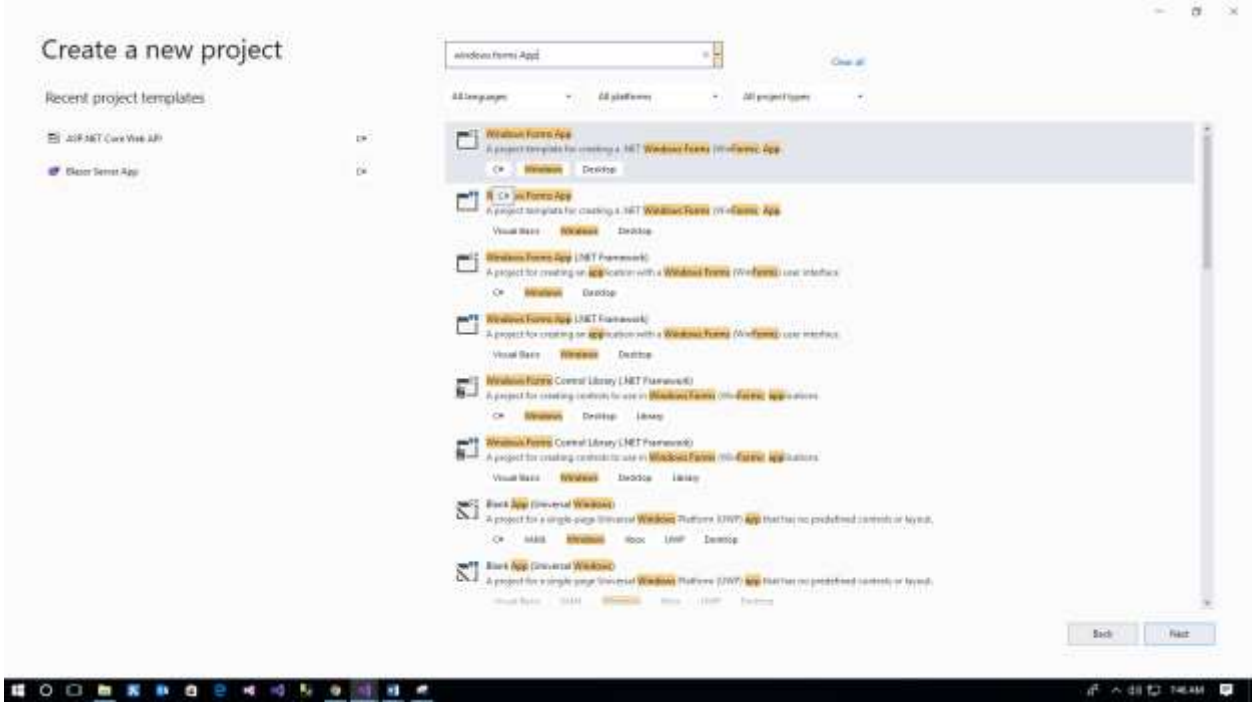
Answer

Steps for developing the above application:

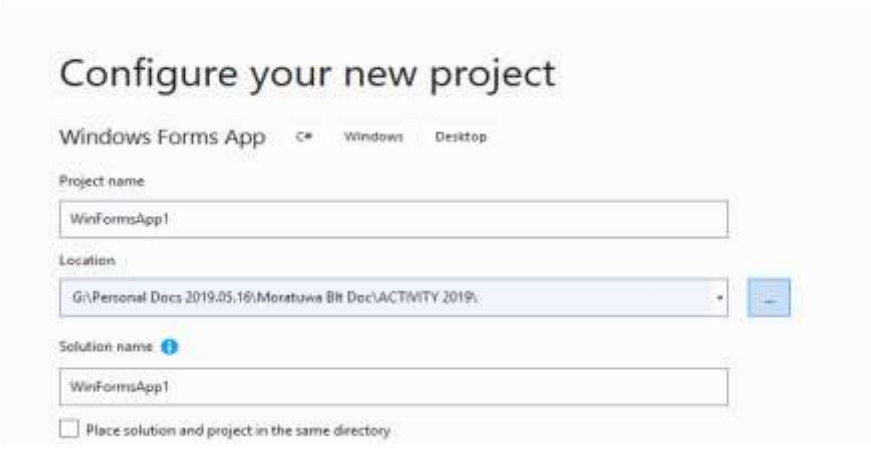
1. On the start page of the visual studio 2019 , you can create new project as below.

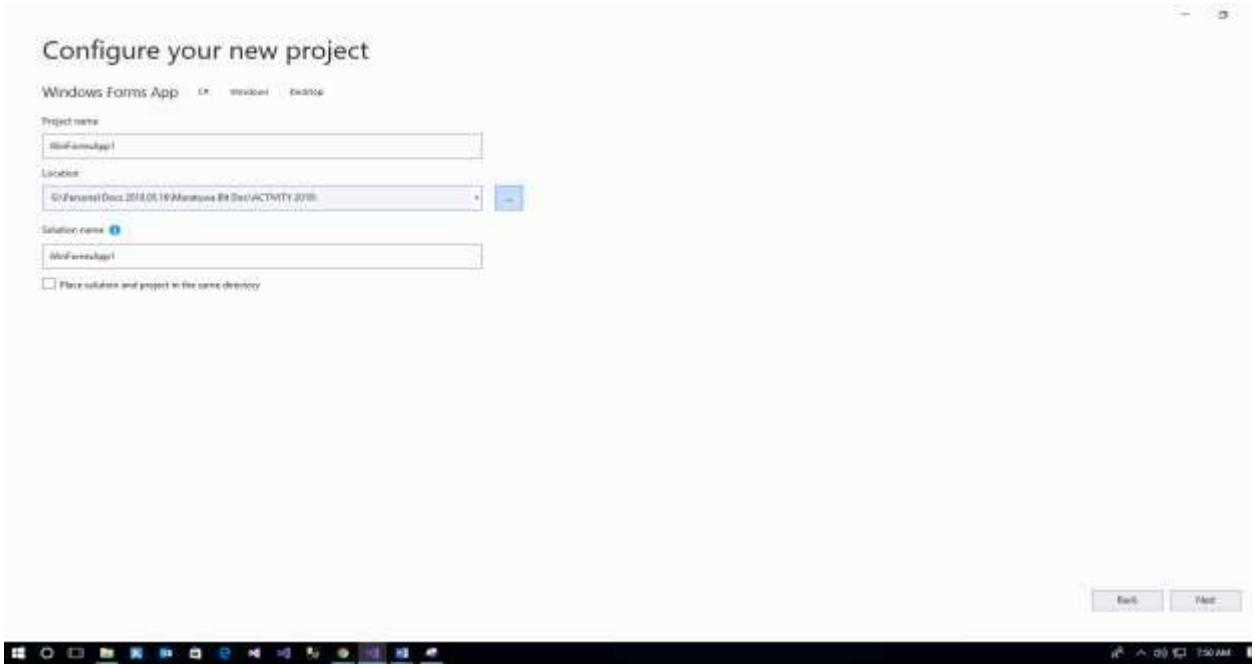


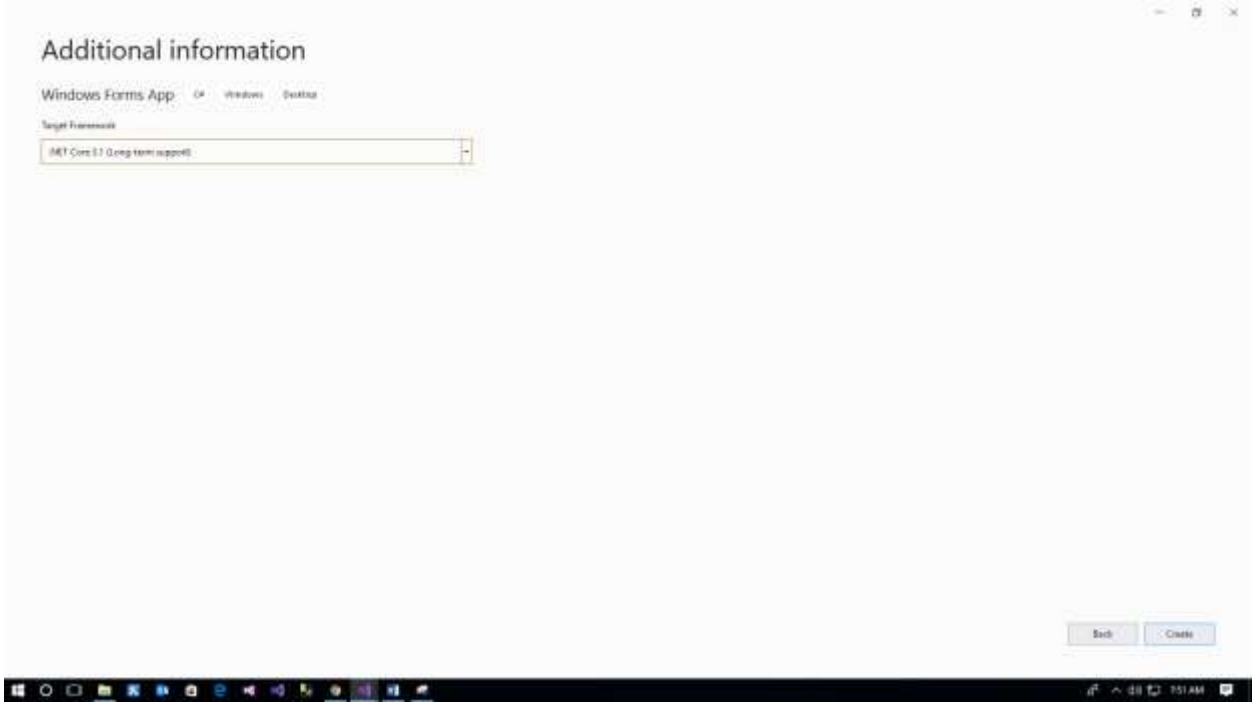
2. In the search box type **Windows form App.** Then, select **Windows form App**



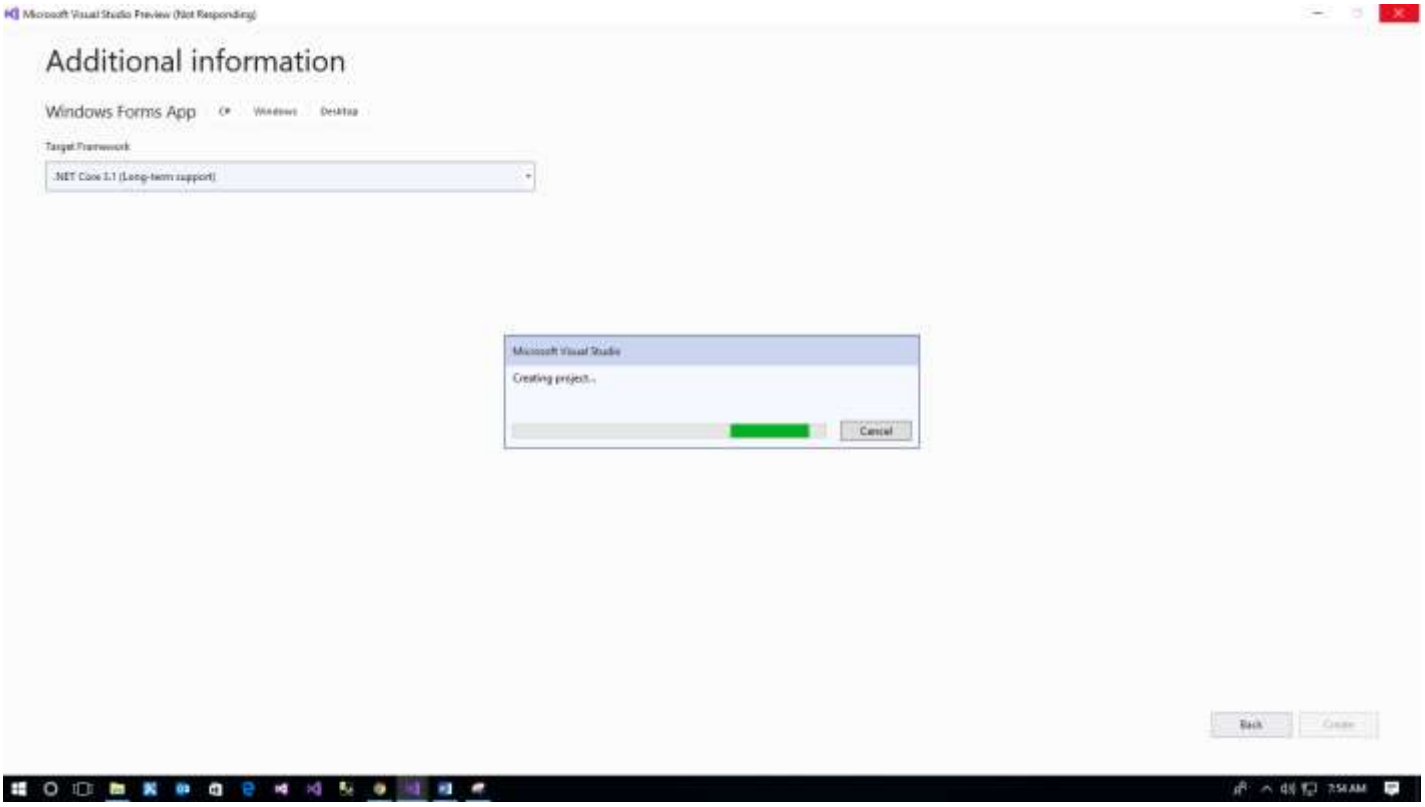
3. Config your project as follow
    - Name the application as you need. Ex: WinformApp1



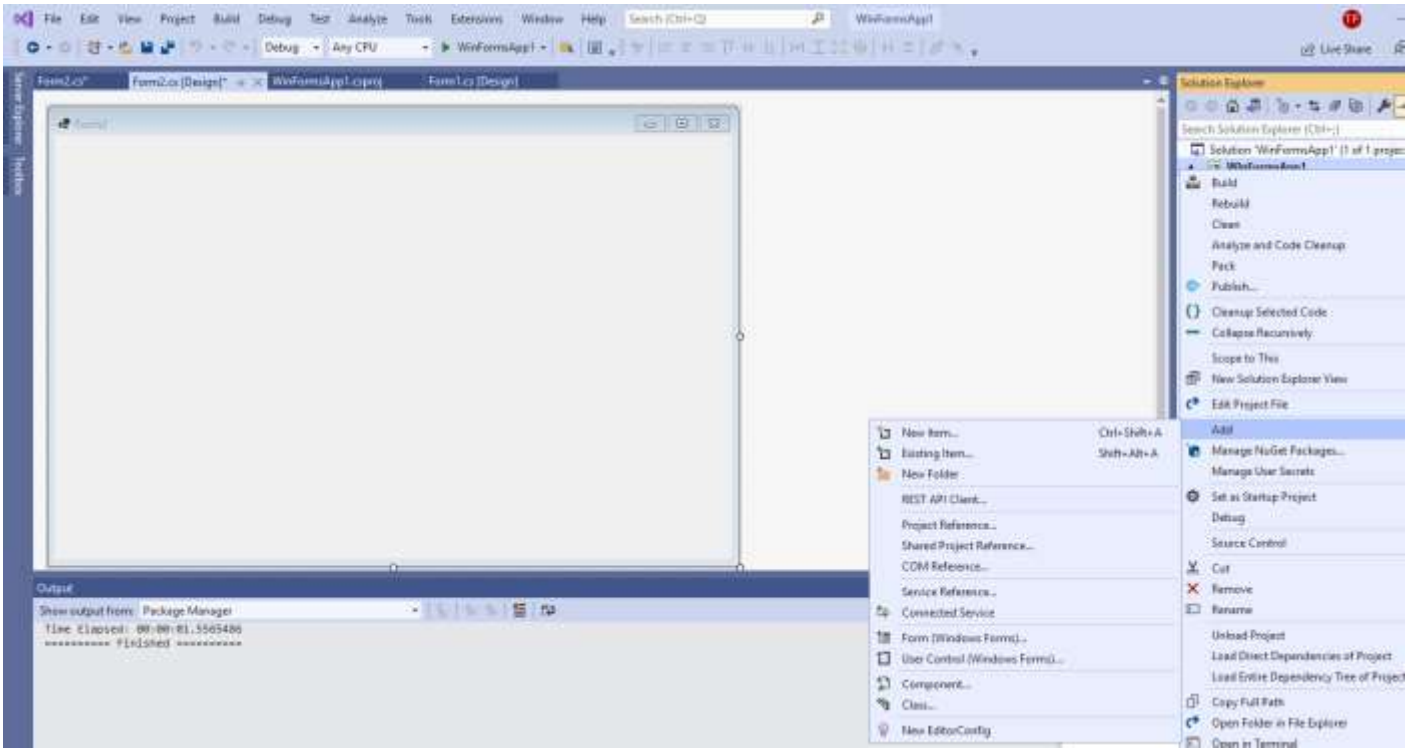




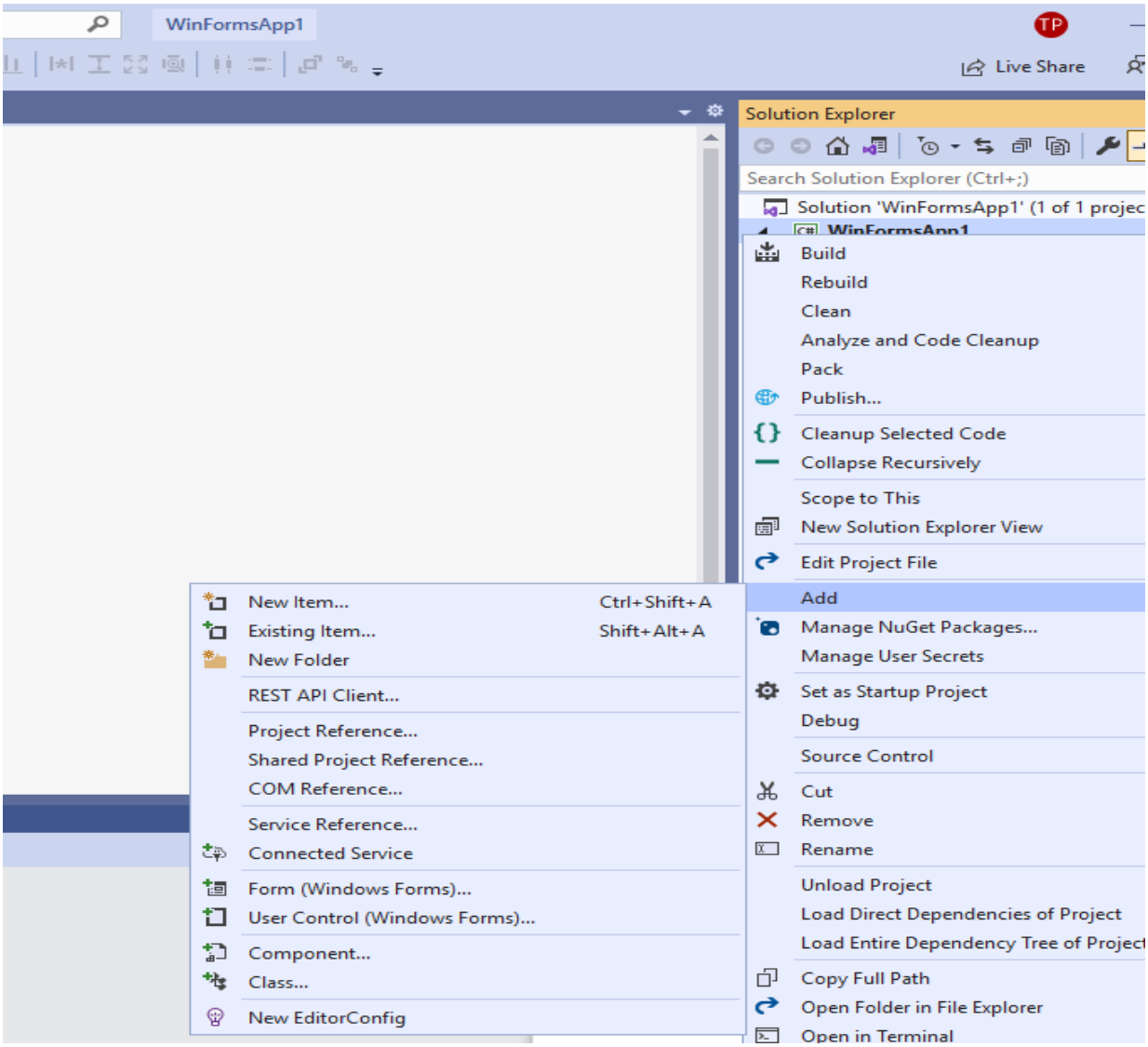
Then Project will be created as follow



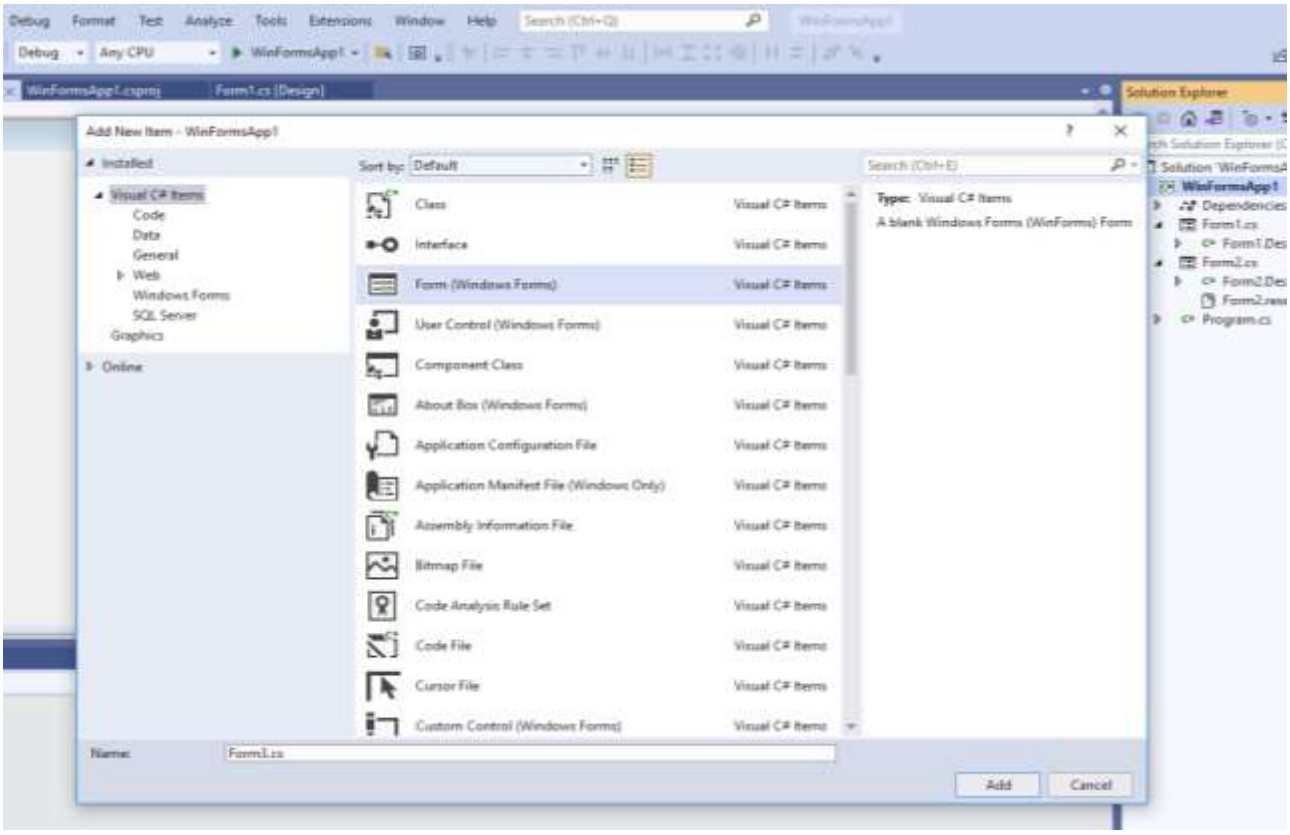
4. Click on the project and you can Add new forms for the project as follow



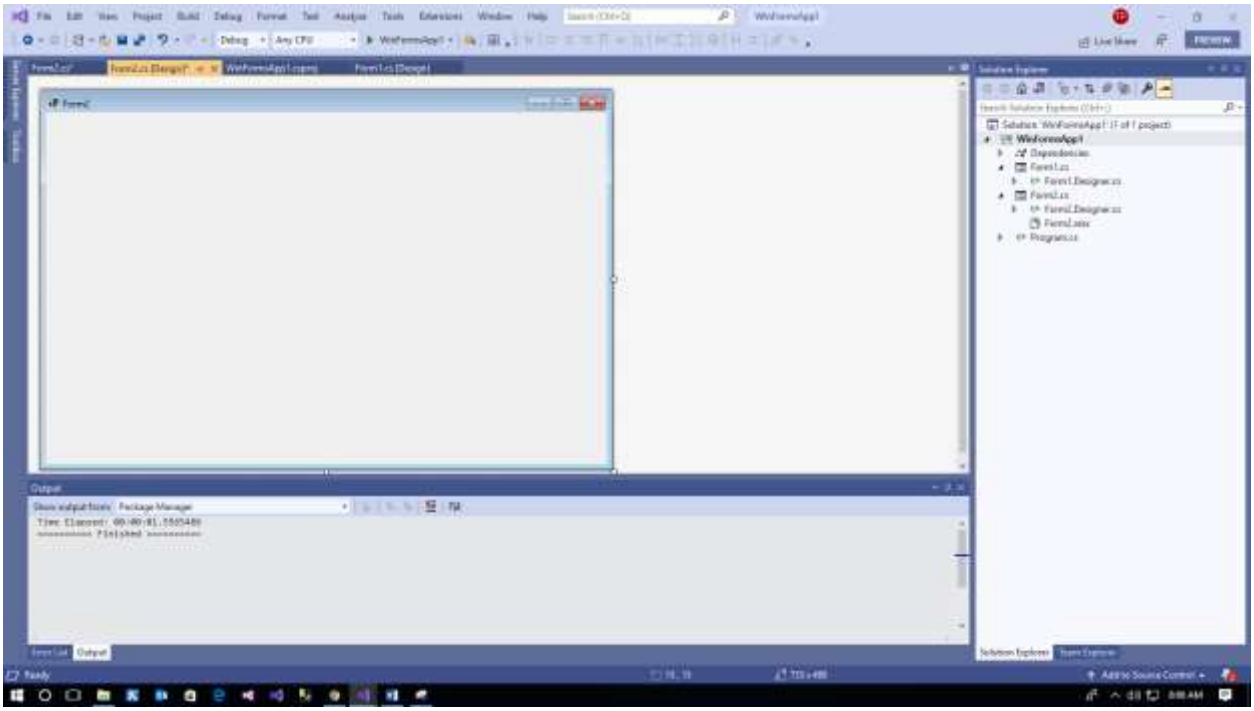
Then select new Item



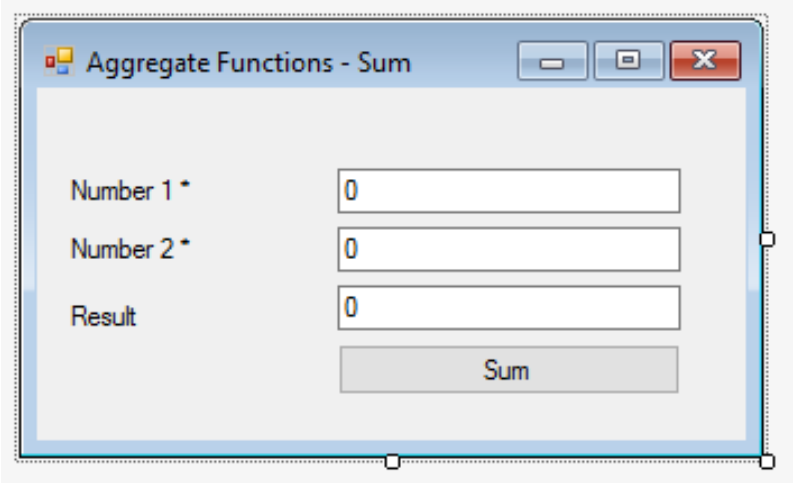
Next Select Form (Windows form)


Debug
Format
Test
Analyze
Tools
Extensions
Window
Help
Debug
Any CPU
WinFormsApp1
WinFormsApp1.csproj
Form1.cs [Design]
Solution Explorer
Add New Item - WinFormsApp1
Installed
Visual C# Items
Code
Data
General
Web
Windows Forms
SQL Server
Graphics
Online
Sort by: Default
Class
Interface
Form (Windows Forms)
User Control (Windows Forms)
Component Class
About Box (Windows Forms)
Application Configuration File
Application Manifest File (Windows Only)
Assembly Information File
Bitmap File
Code Analysis Rule Set
Code File
Cursor File
Custom Control (Windows Forms)
Visual C# Items
Type: Visual C# Items
A blank Windows Forms (WinForms) Form
Name:
Form1.cs
Add
Cancel
Solution 'WinFormsA
WinFormsApp1
Dependencies
Form1.cs
Form1.Des
Form2.cs
Form2.Des
Form2.resx
Program.cs

5. Next you ready to design the on the windows form with Tool box.

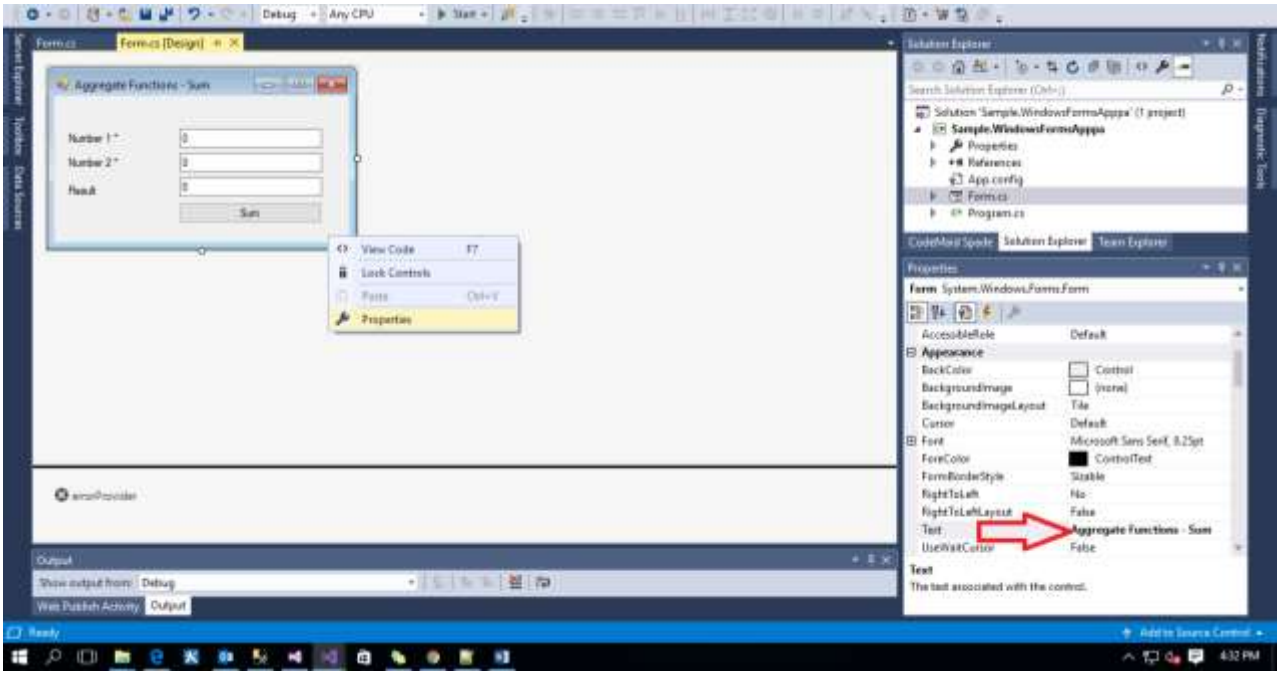


As below you can design your form by dragging tools on the forms form the tool box: (Hope now you have some idea how to design windows application with practice in previous lab sheet.)



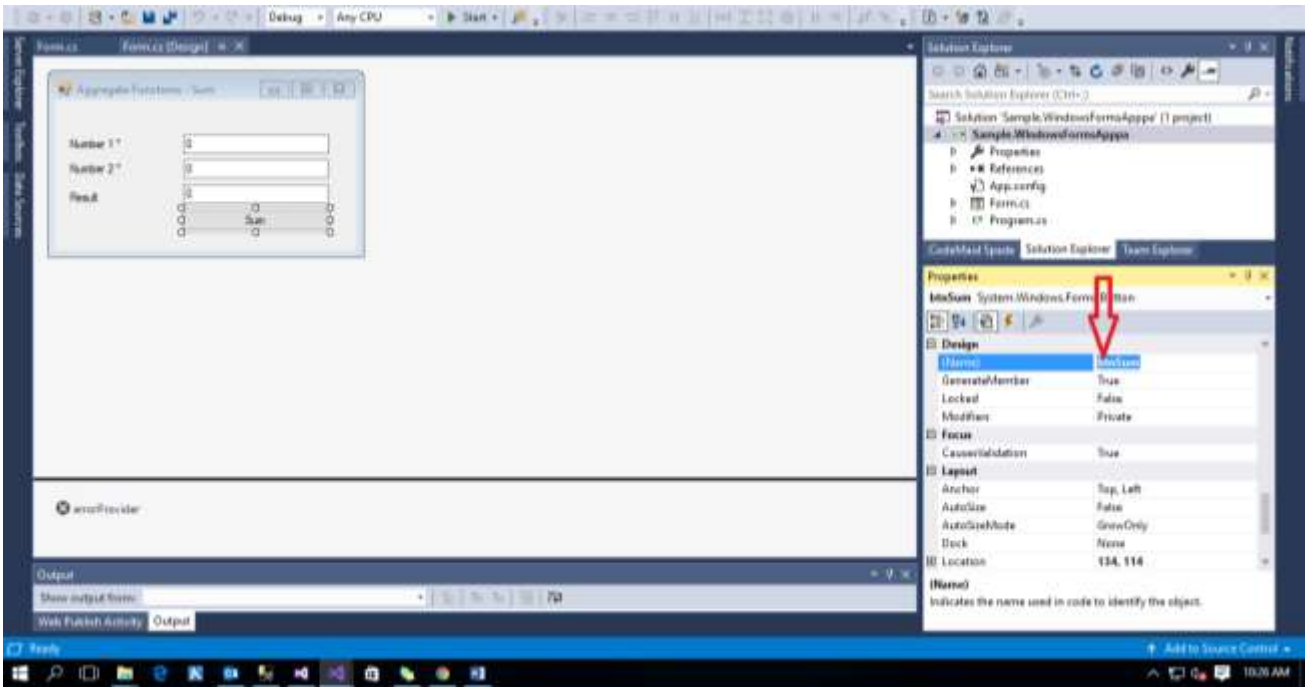
6. Next right click on the form and select properties.

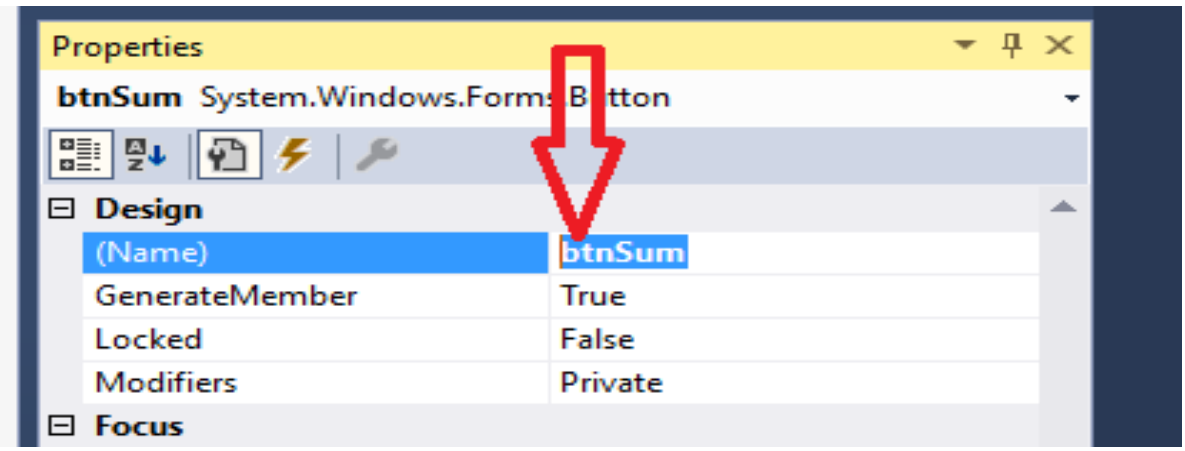
   You can give the name for the form under the **Text** space as below.



7. Next give the name for all controls (Buttons, Textbox, labels) same way. As a good practice you can follow format as below.

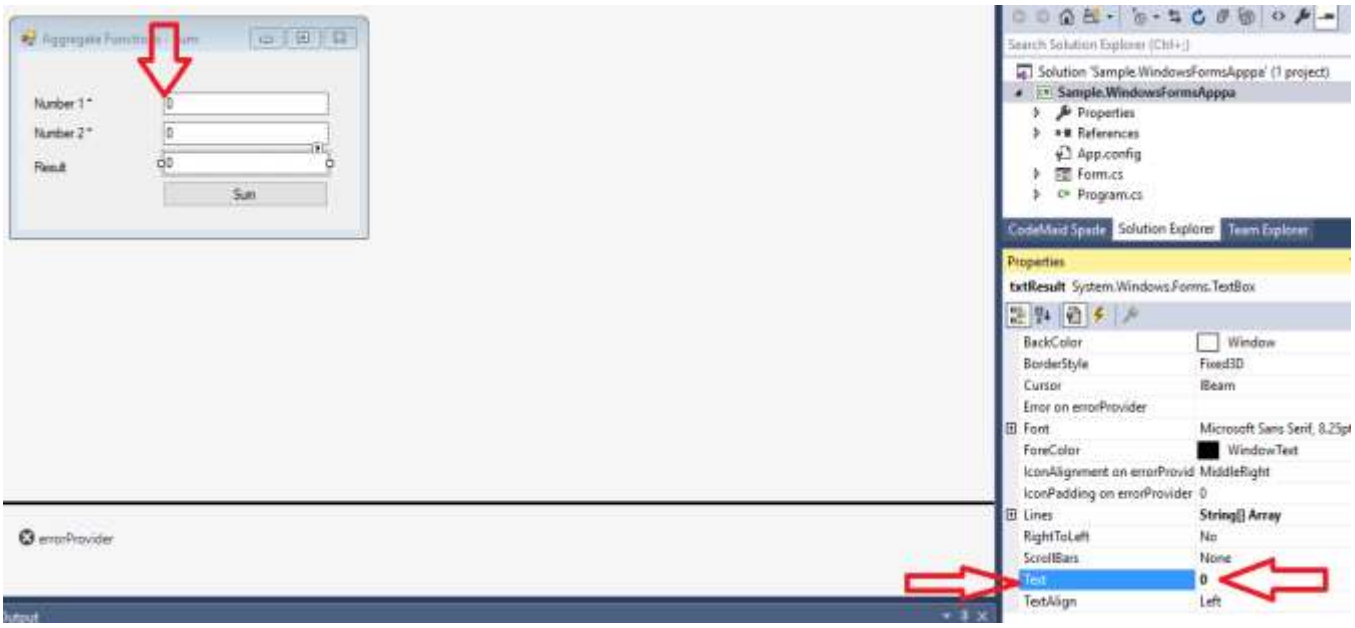
   Example: For the Button name: **btnSum** or **buttonSum**
   (Then You can identify easily button when you need to use it)



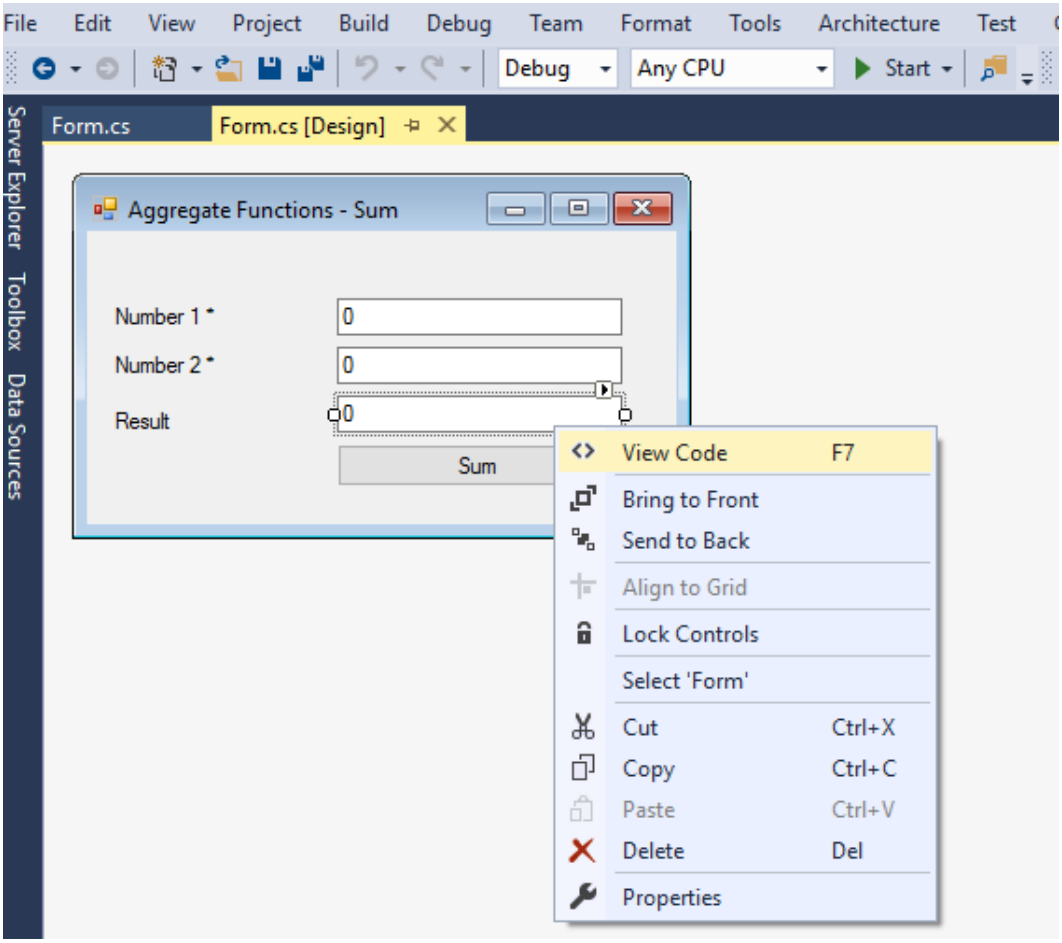


8. Next you can add some features using property window as below.

    - For the example for the number field you can add 0 (zero) in text space when application loading.
    - Like that you find and use other features of the property window.


Number 1 *
Number 2 *
Result
Sum
errorProvider
Search Solution Explorer (Ctrl+;)
Solution 'Sample.WindowsFormsAppps' (1 project)
Sample.WindowsFormsAppps
Properties
References
App.config
Form.cs
Program.cs
CodeMaid Spade
Solution Explorer
Team Explorer
Properties
txtResult System.Windows.Forms.TextBox
BackColor
Window
BorderStyle
Fixed3D
Cursor
IBeam
Error on errorProvider
Font
Microsoft Sans Serif, 8.25pt
ForeColor
WindowText
IconAlignment on errorProvid
MiddleRight
IconPadding on errorProvider
0
Lines
String[] Array
RightToLeft
No
ScrollBars
None
Text
0
TextAlign
Left

9. After you compete the naming for all controls next right click on the form and select the **view Code.**



10. Now you can view form.cs. In form.cs you can initiate the variable as below.

- Declare variable and initialize the variable following below coding.
- Use access modifiers as privet (variable cannot access out of the form.cs).
- Do read only for the **txtResut** (then textbox cannot edit)

```
Form.cs   Form.cs [Design]
Sample.WindowsFormsApppa                    Sample.WindowsFormsApppa.Form
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace Sample.WindowsFormsApppa
{
    3 references
    public partial class Form : System.Windows.Forms.Form
    {
        //Declare Variables
        private int number1;
        private int number2;
        private int result;

        private int intValidation;
        1 reference
        public Form()
        {
            InitializeComponent();

            //Initialise Variables On form constructors.
            number1 = 0;
            number2 = 0;

            //Change attribute
            txtResult.ReadOnly = true;
        }

100 %
```

11. You can write validating event as below.

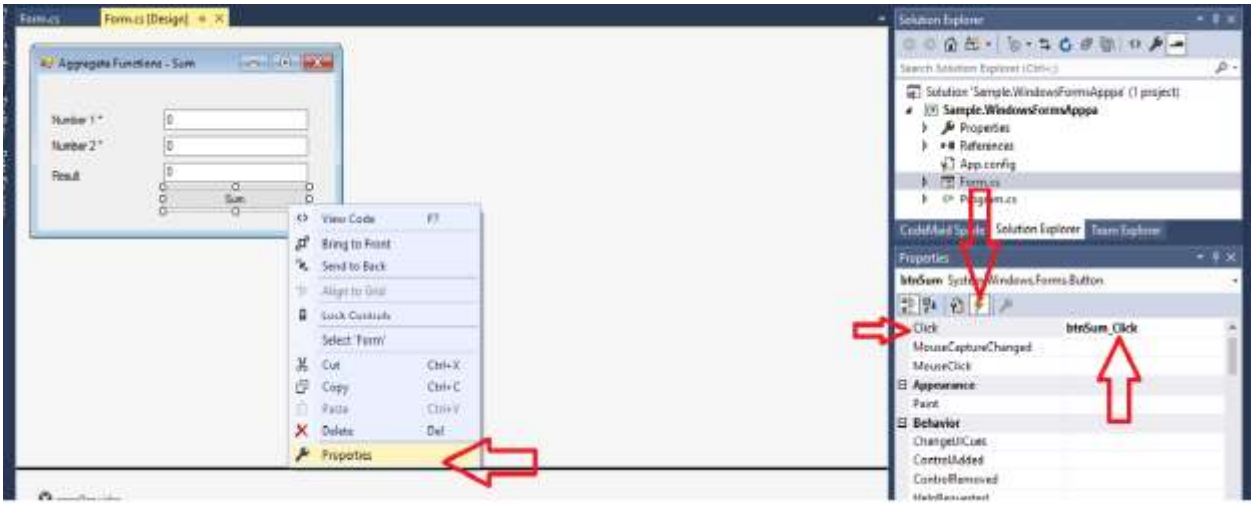
- Right click on the **text box** and select the properties
- Next select the event in the property window.
- After that select the **click** event.

- Next select the validating event under the focus menu in the property event.
- Next you can validate the text field as below.
- Here you can use if statement for the validating.

```
private void txtNumber1_Validating(object sender, CancelEventArgs e)
{
    //Clear errorProvider
    errorProvider.SetError(txtNumber1, "");

    if (!int.TryParse(txtNumber1.Text, out intValidation))
    {
        errorProvider.SetError(txtNumber1, "Please fill the required field");
    }
}

1 reference
private void txtNumber2_Validating(object sender, CancelEventArgs e)
{
    errorProvider.SetError(txtNumber2, "");
    if (!int.TryParse(txtNumber2.Text, out intValidation))
    {
        errorProvider.SetError(txtNumber2, "Please fill the required field");
    }
}

1 reference
private void txtResult_Validating(object sender, CancelEventArgs e)
{
    errorProvider.SetError(txtResult, "");
    if (!int.TryParse(txtResult.Text, out intValidation))
    {
        errorProvider.SetError(txtResult, "Please fill the required field");
    }
```

12. Next you can write sum button click event as below.

- Right click on the **sum button** and select the properties
- Next select the event in the property window.
- After that select the **click** event.

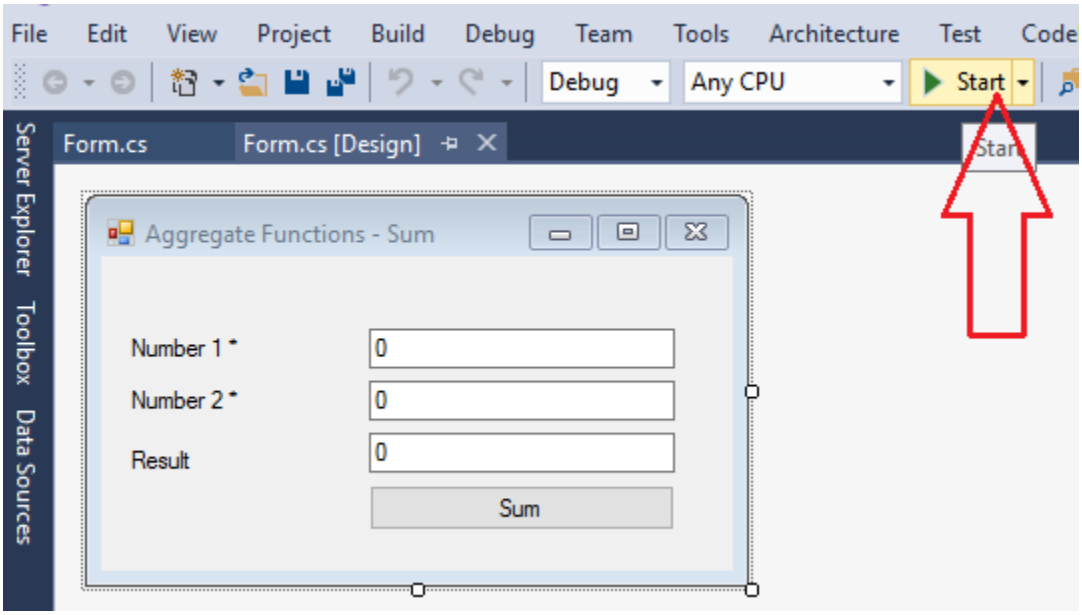


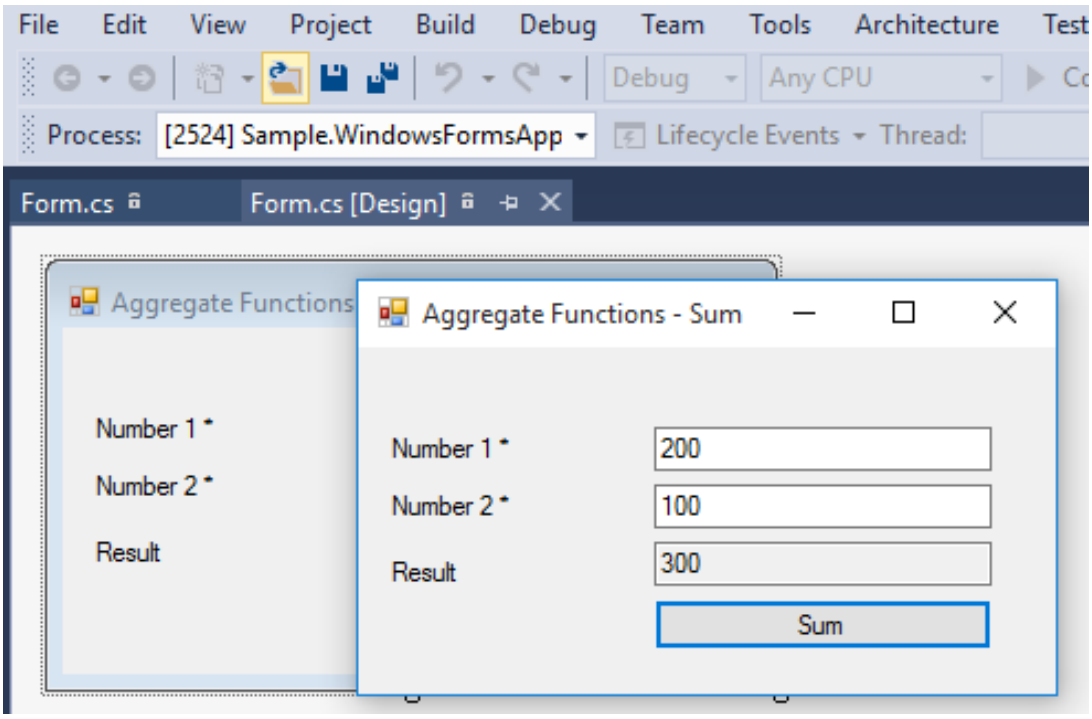
Then you **btnsum_click** event in form.cs. Next you can write the event as below.

- As best practice use **try catch** block.
- A try catch block is placed around code that could throw an exception.

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

    try
    {
        if(ValidateChildren(ValidationConstraints.Enabled))
        {
            number1 = int.Parse(txtNumber1.Text);
            number2 = int.Parse(txtNumber2.Text);

            result = (number1 + number2);

            txtResult.Text = result.ToString();
        }
    }
    catch(Exception ex)
    {
        MessageBox.Show(ex.Message);
    }
}
```

13. Finally, you can run the application





- End of the activity, you have got some ideas about how to develop simple application with functions, event with C# with visual studio.

- Please follow the above Steps further and rewrite different event using C# in visual studio.