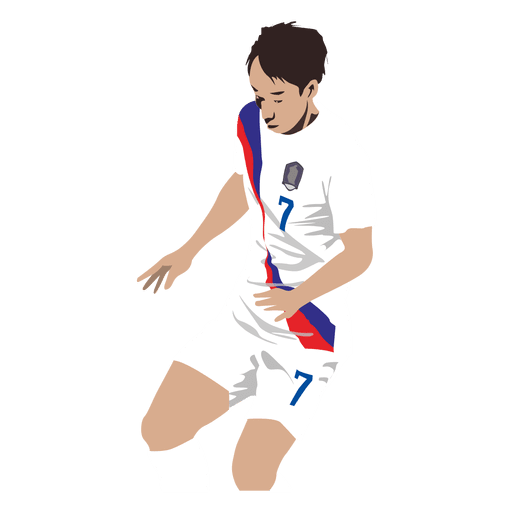
**Soccer Analyst**

**Probability project**

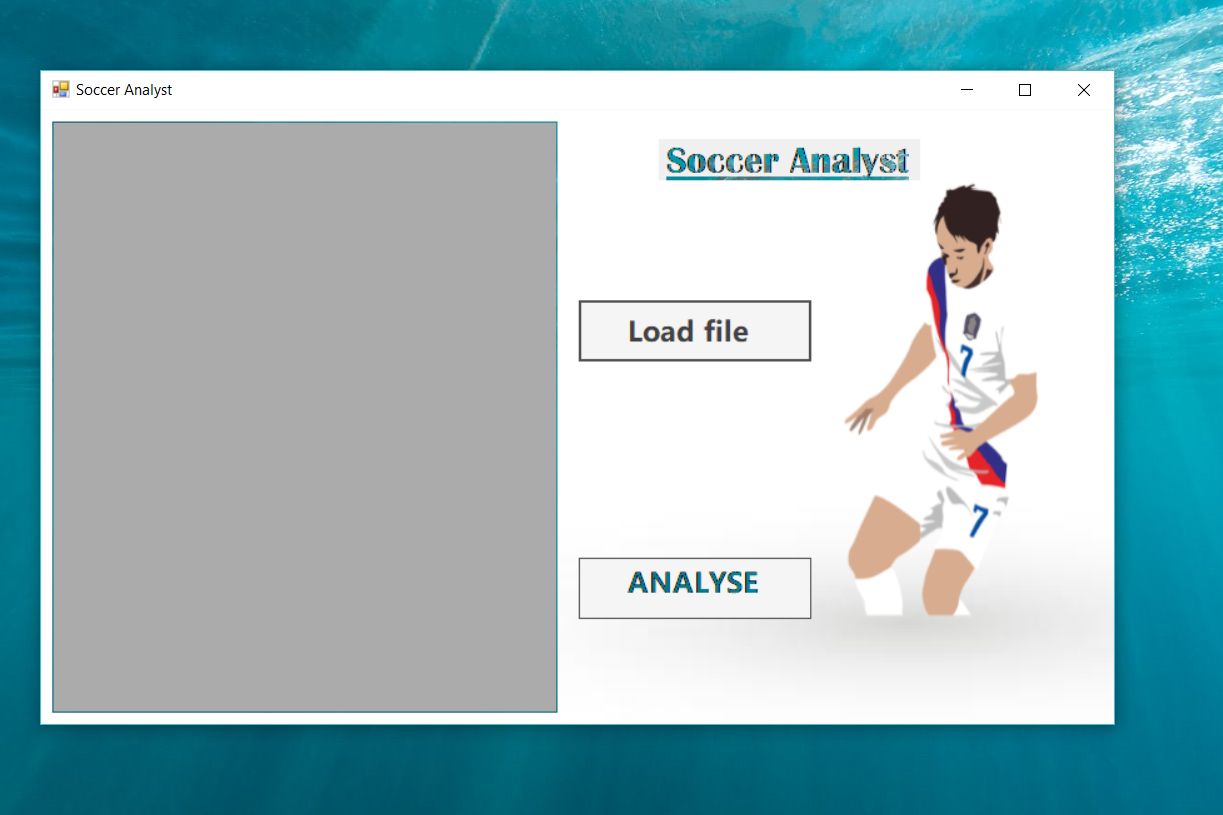


**Ahmed Raza 17f-8060**

**Abdullah Bilal 17f-8349**

**M. Umer 17f-8224**

**Soccer Analyst**



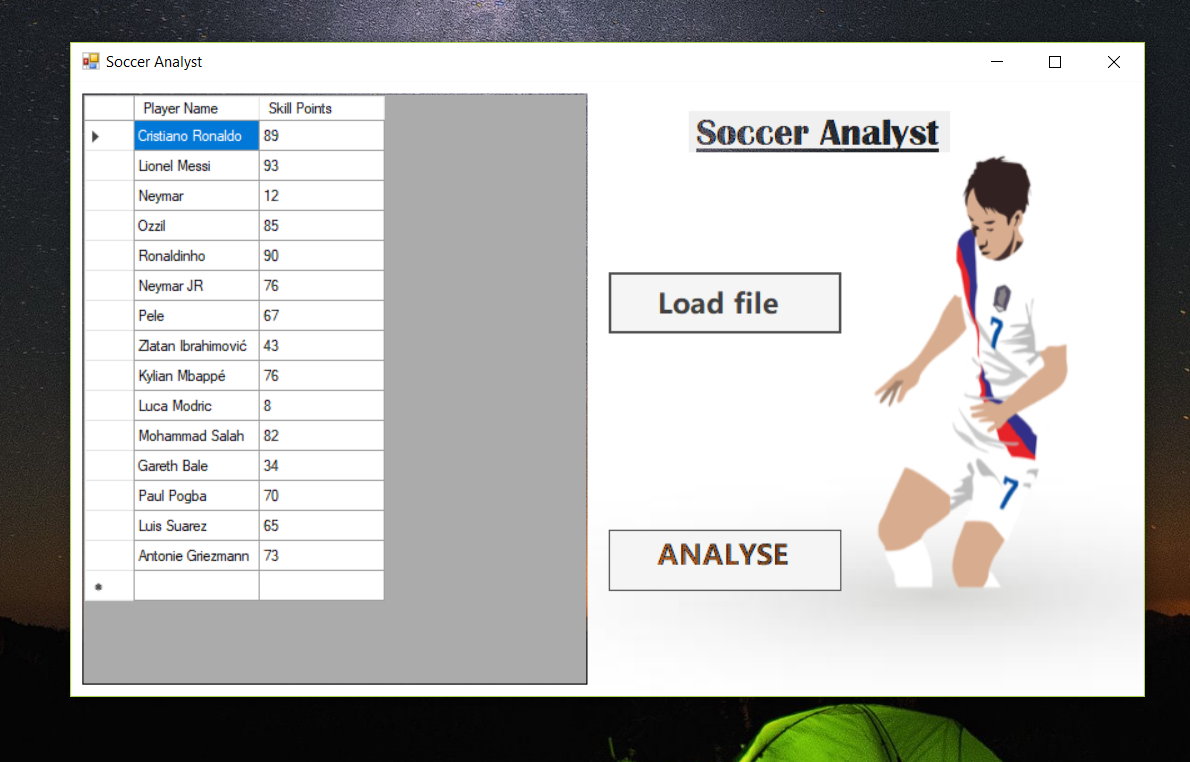
**Description of Soccer Analyst :**

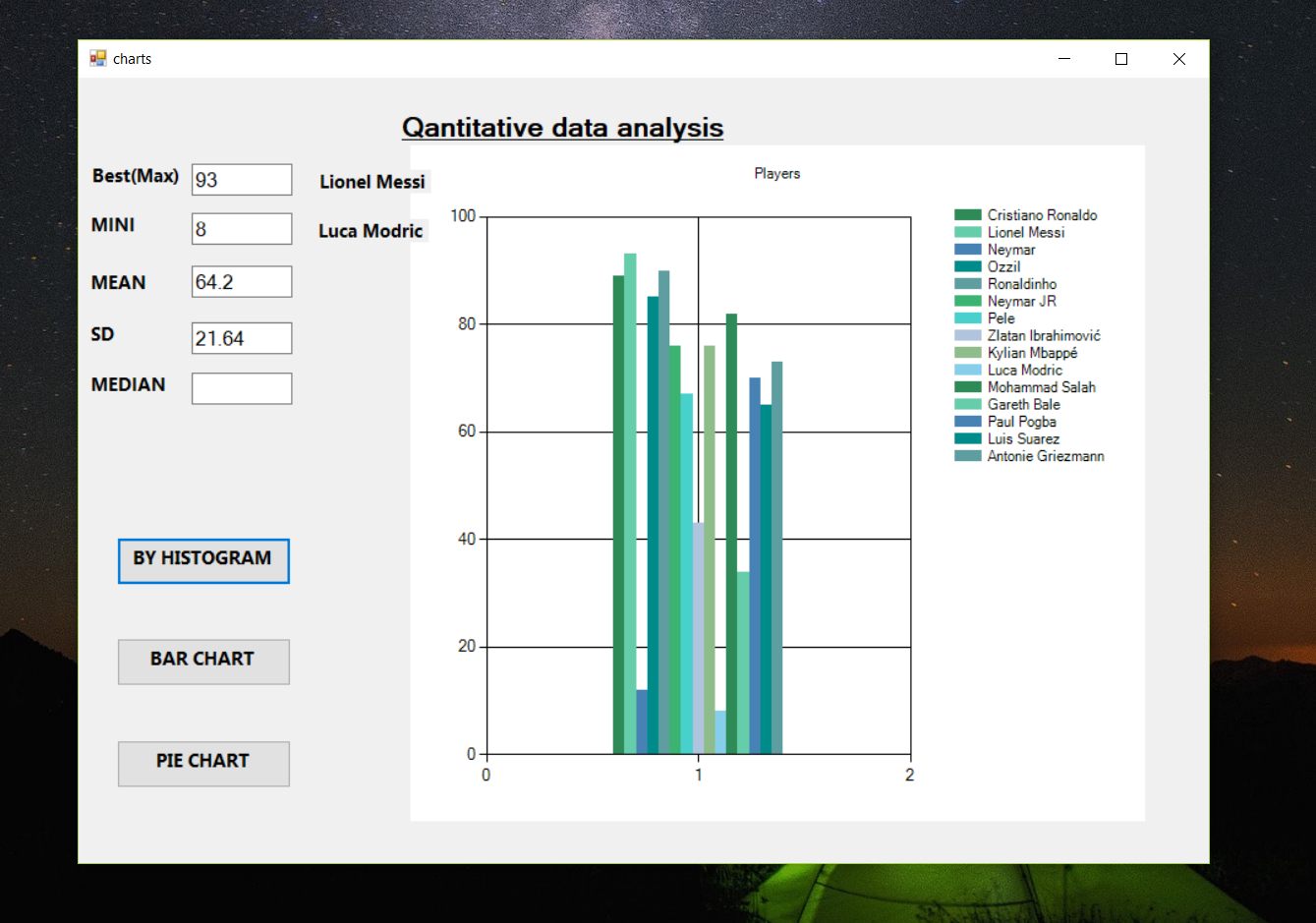
Our program runs on the Visual Studio. We make it in the c# window form. In this program, user is able to choose his/her desired file but it should only contain 2 columns(1 of the name and other of qualitative/quantitative data).User choose the file from the computer and program load the data in the **DATA GRID VIEW**.

Next thing, when user click on the button of **ANALYSE DATA**, the program automatically detects if the data is qualitative or quantitative. If data is quantitative, the page shown to the user will be different than if the user will give qualitative data.

Next thing, it takes **screenshot of the graphs** and **save it** to the Project folder. User can access and use them.

**Screen Shots:**





**Using for a file :**

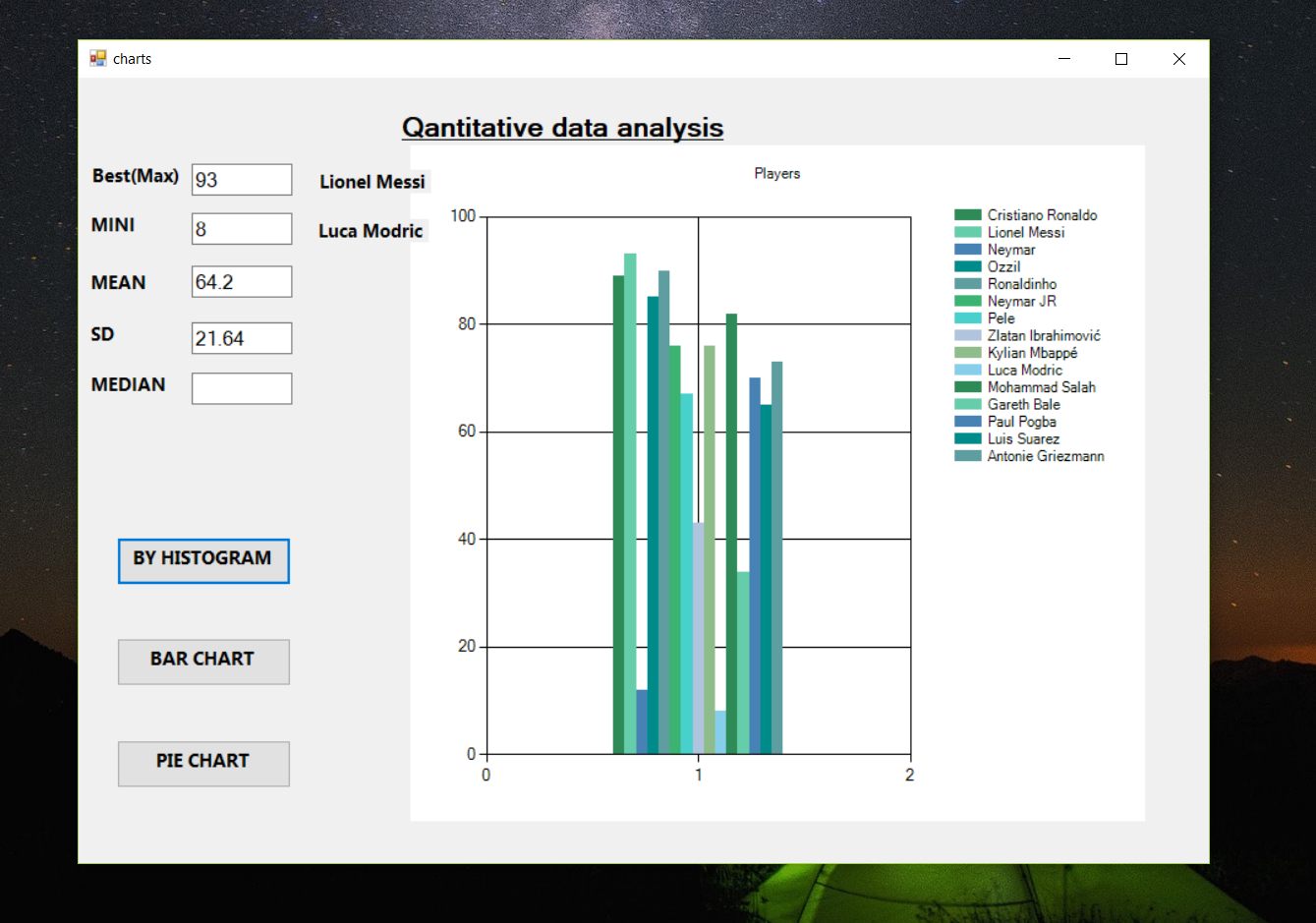
We are using a file ***“Foot Players Data“*** in which there are **2 sheets**.

**Sheet1** have **Quantitative values** for the football players.***Row-1*** have ***Name of the players*** and ***Row2*** have ***Scoring points*** of them out 100.Each player has should have different names so that program act properly.

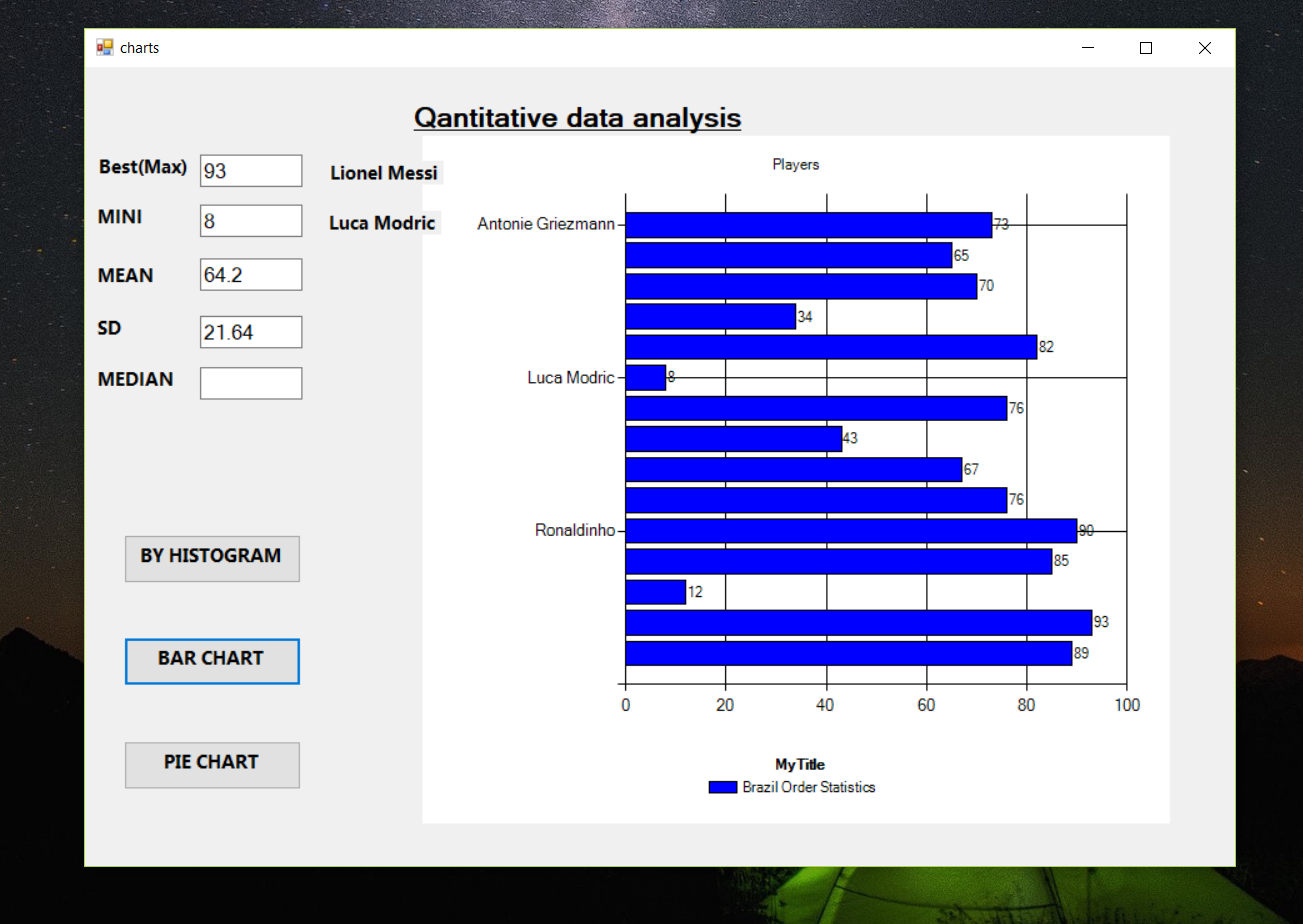
**For Qualitative data(Sheet1):**

For the qualitative data, it will show the following form:

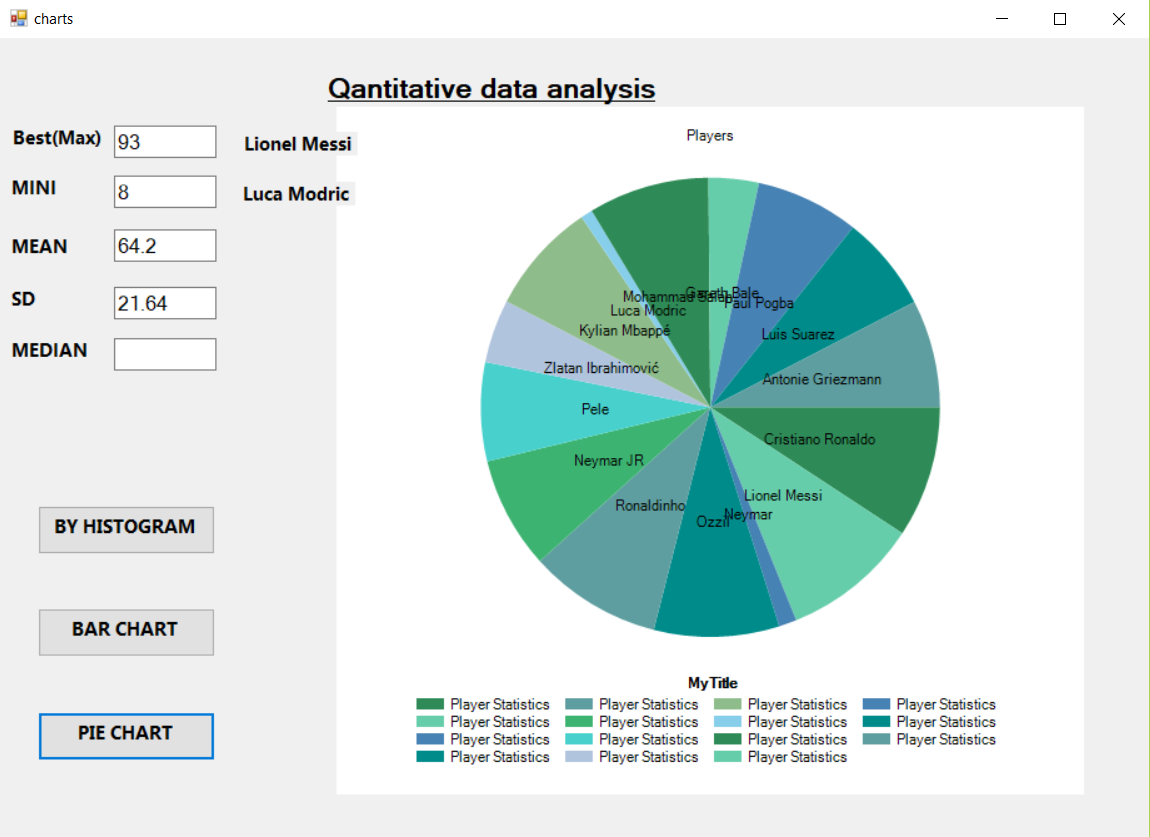
* **Max**
* **Mini**
* **Mean**
* **Standard Deviation**
* **Median**
* **Histogram**



* **Bar Chart**

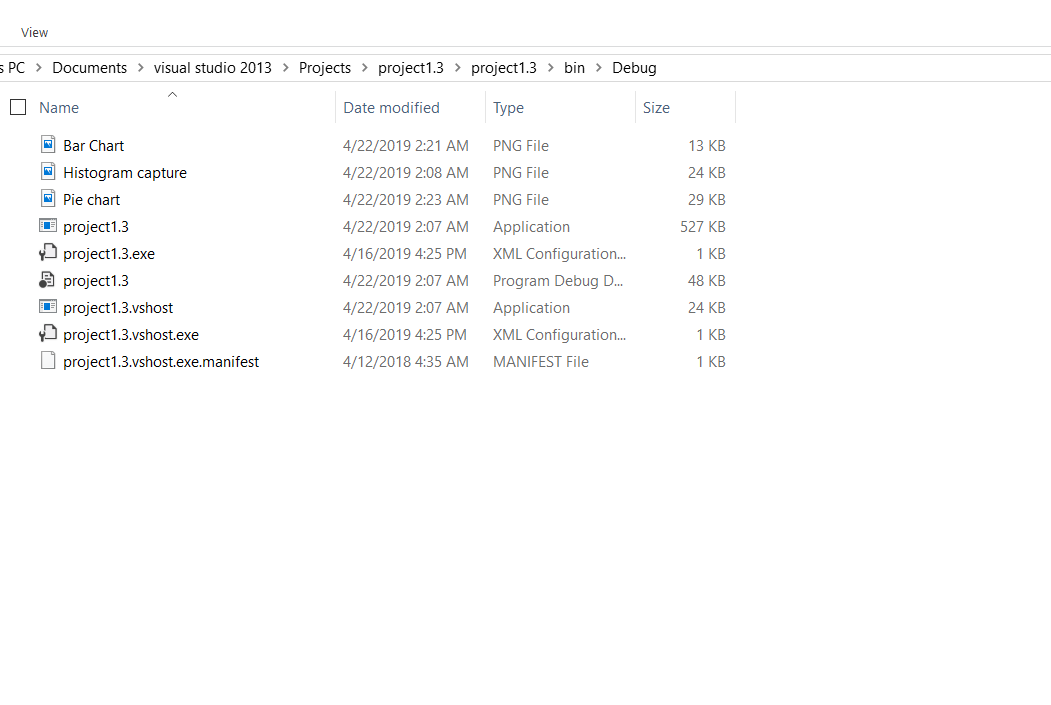


* **Pie Chart**



**Saving it in a Folder :**

The program automatically take **screenshot of the graphs** and **save it** to the Project folder. User can access and use them when they want. The screenshot will be in the following address **“Project Folder ->bin->Debug”.** Screenshot will have **PNG format**.



**Screenshot will be save in this way**

For the **Qualitative data (sheet2),** it will give user the other form and this form evaluates all the qualitative calculations.

**CODE:**

**Language :** C# (Windows Form)

**Form1.cs :**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

using Excel = Microsoft.Office.Interop.Excel;

namespace project1.\_3

{

public partial class Form1 : Form

{

List<Class1> lst1;

public Form1()

{

InitializeComponent();

lst1=new List<Class1>();

}

// CREATE EXCEL OBJECTS.

Excel.Application xlApp = new Excel.Application();

Excel.Workbook xlWorkBook;

Excel.Worksheet xlWorkSheet;

OpenFileDialog OpenFileDialog1 = new OpenFileDialog();

string sFileName;

int iRow, iCol = 2;

private void button1\_Click(object sender, EventArgs e)

{

OpenFileDialog1.Title = "Excel File to Edit";

OpenFileDialog1.FileName = "";

OpenFileDialog1.Filter = "XLS files (\*.xls, \*.xlt)|\*.xls;\*.xlt|XLSX files (\*.xlsx, \*.xlsm, \*.xltx, \*.xltm)|\*.xlsx;\*.xlsm;\*.xltx;\*.xltm|ODS files (\*.ods, \*.ots)|\*.ods;\*.ots|CSV files (\*.csv, \*.tsv)|\*.csv;\*.tsv|HTML files (\*.html, \*.htm)|\*.html;\*.htm";

if (OpenFileDialog1.ShowDialog() == DialogResult.OK)

{

sFileName = OpenFileDialog1.FileName;

if (sFileName.Trim() != "")

{

readExcel(sFileName);

}

}

}

private void readExcel(string sFile)

{

xlApp = new Excel.Application();

xlWorkBook = xlApp.Workbooks.Open(sFile);

xlWorkSheet = xlWorkBook.Worksheets["Sheet1"];

// FIRST, CREATE THE DataGridView COLUMN HEADERS.

int iCol;

for (iCol = 1; iCol <= xlWorkSheet.Columns.Count; iCol++)

{

if (xlWorkSheet.Cells[1, iCol].value == null)

{

break; // BREAK LOOP.

}

else

{

DataGridViewColumn col = new DataGridViewTextBoxColumn();

col.HeaderText = xlWorkSheet.Cells[1, iCol].value;

int colIndex = dataGridView1.Columns.Add(col); // ADD A NEW COLUMN.

}

}

// ADD ROWS TO THE GRID.

for (iRow = 2; iRow <= xlWorkSheet.Rows.Count; iRow++) // START FROM THE SECOND ROW.

{

if (xlWorkSheet.Cells[ iRow, 1].value == null)

{

break; // BREAK LOOP.

}

else

{

// CREATE A STRING ARRAY USING THE VALUES IN EACH ROW OF THE SHEET.

string[] row = new string[] { xlWorkSheet.Cells[iRow, 1].value,

xlWorkSheet.Cells[iRow, 2].value.ToString()};

//xlWorkSheet.Cells[iRow, 3].value, xlWorkSheet.Cells[iRow, 4].value, xlWorkSheet.Cells[iRow, 5].value.ToString() , xlWorkSheet.Cells[iRow, 6].value.ToString()};

Class1 obj = new Class1();

obj.name = xlWorkSheet.Cells[iRow, 1].value;

obj.goal = Int32.Parse(xlWorkSheet.Cells[iRow, 2].value.ToString());

// obj.goal=xlWorkSheet.Cells[iRow, 5].value.ToString() ;

lst1.Add(obj);

// ADD A NEW ROW TO THE GRID USING THE ARRAY DATA.

dataGridView1.Rows.Add(row);

}

}

xlWorkBook.Close();

xlApp.Quit();

System.Runtime.InteropServices.Marshal.ReleaseComObject(xlApp);

System.Runtime.InteropServices.Marshal.ReleaseComObject(xlWorkBook);

System.Runtime.InteropServices.Marshal.ReleaseComObject(xlWorkSheet);

}

private void dataGridView1\_CellContentClick(object sender, DataGridViewCellEventArgs e)

{

}

private void button2\_Click(object sender, EventArgs e)

{

charts chartvala = new charts(lst1);

this.Hide();

chartvala.Show();

}

private void Form1\_Load(object sender, EventArgs e)

{

}

private void button3\_Click(object sender, EventArgs e)

{

this.Hide();

}

}

}

**Chart1.cs**

using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using System.Windows.Forms.DataVisualization.Charting;

using project1.\_3.Properties;

using System.Threading;

namespace project1.\_3

{

public partial class charts : Form

{

string sFile;

List<Class1> listobj = new List<Class1>();

////for usage of panelr

//List<Panel> listPanel = new List<Panel>();

//int click;

public charts()

{

InitializeComponent();

}

public void check\_data()

{

bool check = false;

int[] y = { listobj.ElementAt(0).goal, listobj.ElementAt(1).goal, listobj.ElementAt(2).goal, listobj.ElementAt(3).goal, listobj.ElementAt(4).goal, listobj.ElementAt(5).goal, listobj.ElementAt(6).goal, listobj.ElementAt(7).goal, listobj.ElementAt(8).goal, listobj.ElementAt(9).goal, listobj.ElementAt(10).goal, listobj.ElementAt(11).goal, listobj.ElementAt(12).goal, listobj.ElementAt(13).goal, listobj.ElementAt(14).goal };

for(int i=0;i<15;i++)

{

if(y[i]<0 || y[i]>100)

{

MessageBox.Show("There's invalid data in the fucking 2nd row ");

panel1.Hide();

return;

}

if(y[i]>1)

{

check = true;

break;

}

}

if (check == false)

panel1.Show();

else

panel1.Hide();

}

public charts(List<Class1> list1)

{

InitializeComponent();

listobj = list1;

}

private void charts\_Load(object sender, EventArgs e)

{

check\_data();

string[] x = { listobj.ElementAt(0).name, listobj.ElementAt(1).name, listobj.ElementAt(2).name, listobj.ElementAt(3).name, listobj.ElementAt(4).name, listobj.ElementAt(5).name, listobj.ElementAt(6).name, listobj.ElementAt(7).name, listobj.ElementAt(8).name, listobj.ElementAt(9).name, listobj.ElementAt(10).name, listobj.ElementAt(11).name, listobj.ElementAt(12).name, listobj.ElementAt(13).name, listobj.ElementAt(14).name };

int[] y = { listobj.ElementAt(0).goal, listobj.ElementAt(1).goal, listobj.ElementAt(2).goal, listobj.ElementAt(3).goal, listobj.ElementAt(4).goal, listobj.ElementAt(5).goal, listobj.ElementAt(6).goal, listobj.ElementAt(7).goal, listobj.ElementAt(8).goal, listobj.ElementAt(9).goal, listobj.ElementAt(10).goal, listobj.ElementAt(11).goal, listobj.ElementAt(12).goal, listobj.ElementAt(13).goal, listobj.ElementAt(14).goal };

textBox1.Text = y.Max().ToString();

int max = y.Max();

int index = 0;

for(int i=0;i<=14;i++)

{

if(y[i]==max)

{

index = i;

break;

}

else

{

}

}

label11.Text = x[index];

//for average

textBox4.Text = y.Min().ToString();

int mini = y.Min();

for (int i = 0; i <= 14; i++)

{

if (y[i] == mini)

{

index = i;

break;

}

else

{

}

}

label8.Text = x[index];

textBox3.Text = y.Average().ToString();

//calculating standard devition

int sum=0, sumSquares=0;

// Calculate the total for the sum

for (int i = 0; i < 15; i++)

sum += y[i];

int squareSums = sum \* sum;

for (int i = 0; i < 15; i++)

sumSquares += (y[i] \* y[i]);

// Now we can calculate the standard deviation

double numerator = (14 \* sumSquares - squareSums);

double denominator = (14 \* (14 - 1));

double stdDev = Math.Sqrt(numerator / denominator);

// Display the values(standard deviationand stuff)

//txtC.Text = values.Count.ToString();

//txtSum.Text = sum.ToString("F");

// txtSumSquares.Text = sumSquares.ToString("F");

// txtSquareSums.Text = squareSums.ToString("F");

textBox2.Text = stdDev.ToString("F");

}

private void chart1\_Click(object sender, EventArgs e)

{

// fillDataGridView();

}

private void label1\_Click(object sender, EventArgs e)

{

}

private void button1\_Click(object sender, EventArgs e)

{

//if (click < listPanel.Count - 1)

// listPanel[++click].BringToFront();

this.chart1.Series.Clear();

Series series = new Series();

series.ChartArea = "chart1";

// Data arrays.

string[] x = { listobj.ElementAt(0).name, listobj.ElementAt(1).name, listobj.ElementAt(2).name, listobj.ElementAt(3).name, listobj.ElementAt(4).name, listobj.ElementAt(5).name, listobj.ElementAt(6).name, listobj.ElementAt(7).name, listobj.ElementAt(8).name, listobj.ElementAt(9).name, listobj.ElementAt(10).name, listobj.ElementAt(11).name, listobj.ElementAt(12).name, listobj.ElementAt(13).name, listobj.ElementAt(14).name };

int[] y = { listobj.ElementAt(0).goal, listobj.ElementAt(1).goal, listobj.ElementAt(2).goal, listobj.ElementAt(3).goal, listobj.ElementAt(4).goal, listobj.ElementAt(5).goal, listobj.ElementAt(6).goal, listobj.ElementAt(7).goal, listobj.ElementAt(8).goal, listobj.ElementAt(9).goal, listobj.ElementAt(10).goal, listobj.ElementAt(11).goal, listobj.ElementAt(12).goal, listobj.ElementAt(13).goal, listobj.ElementAt(14).goal };

// // Set palette.

this.chart1.Palette = ChartColorPalette.SeaGreen;

// // Set title.

this.chart1.Titles.Add("Players");

// // Add series.

for (int i = 0; i < x.Length; i++)

{

// // Add series.

Series seri = this.chart1.Series.Add(x[i]);

// // Add point.

seri.Points.Add(y[i]);

}

//saving a scrrenshot

this.chart1.SaveImage("Histogram capture.png", ChartImageFormat.Png);

}

private void button4\_Click(object sender, EventArgs e)

{

//PictureBox pic=new PictureBox();

}

private void button2\_Click(object sender, EventArgs e)

{

//if (click < listPanel.Count - 1)

// listPanel[--click].BringToFront();

chart1.ResetAutoValues();

chart1.Series.Clear();

chart1.Legends.Clear();

//Add a new Legend(if needed) and do some formating

chart1.Legends.Add("MyLegend");

chart1.Legends[0].LegendStyle = LegendStyle.Table;

chart1.Legends[0].Docking = Docking.Bottom;

chart1.Legends[0].Alignment = StringAlignment.Center;

chart1.Legends[0].Title = "MyTitle";

chart1.Legends[0].BorderColor = Color.Transparent;

Series standardSeries = new Series();

standardSeries.Name = "MySeriesName";

standardSeries.ChartType = SeriesChartType.RangeColumn;

standardSeries.BorderWidth = 1;

standardSeries.BorderDashStyle = ChartDashStyle.Solid;

standardSeries.BorderColor = Color.Black;

standardSeries.Color = Color.Blue;

chart1.Series.Add(standardSeries);

string[] x = { listobj.ElementAt(0).name, listobj.ElementAt(1).name, listobj.ElementAt(2).name, listobj.ElementAt(3).name, listobj.ElementAt(4).name, listobj.ElementAt(5).name, listobj.ElementAt(6).name, listobj.ElementAt(7).name, listobj.ElementAt(8).name, listobj.ElementAt(9).name, listobj.ElementAt(10).name, listobj.ElementAt(11).name, listobj.ElementAt(12).name, listobj.ElementAt(13).name, listobj.ElementAt(14).name };

int[] y = { listobj.ElementAt(0).goal, listobj.ElementAt(1).goal, listobj.ElementAt(2).goal, listobj.ElementAt(3).goal, listobj.ElementAt(4).goal, listobj.ElementAt(5).goal, listobj.ElementAt(6).goal, listobj.ElementAt(7).goal, listobj.ElementAt(8).goal, listobj.ElementAt(9).goal, listobj.ElementAt(10).goal, listobj.ElementAt(11).goal, listobj.ElementAt(12).goal, listobj.ElementAt(13).goal, listobj.ElementAt(14).goal };

chart1.Series["MySeriesName"].LegendText = "Brazil Order Statistics";

chart1.Series["MySeriesName"].ChartType = SeriesChartType.Bar;

chart1.Series["MySeriesName"].IsValueShownAsLabel = true;

//chart1.Series[seriesname].Points.DataBindY(y);

chart1.Series["MySeriesName"].Points.DataBindXY(x, y);

//saving a screenshot

this.chart1.SaveImage("Bar Chart.png", ChartImageFormat.Png);

}

private void button3\_Click(object sender, EventArgs e)

{

chart1.ResetAutoValues();

chart1.Series.Clear();

chart1.Legends.Clear();

//Add a new Legend(if needed) and do some formating

chart1.Legends.Add("MyLegend");

chart1.Legends[0].LegendStyle = LegendStyle.Table;

chart1.Legends[0].Docking = Docking.Bottom;

chart1.Legends[0].Alignment = StringAlignment.Center;

chart1.Legends[0].Title = "MyTitle";

chart1.Legends[0].BorderColor = Color.Transparent;

//Add a new chart- series

string seriesname = "MySeriesName";

chart1.Series.Add(seriesname);

//set the chart - type to "Pie"

chart1.Series[seriesname].ChartType = SeriesChartType.Pie;

string[] x = { listobj.ElementAt(0).name, listobj.ElementAt(1).name, listobj.ElementAt(2).name, listobj.ElementAt(3).name, listobj.ElementAt(4).name, listobj.ElementAt(5).name, listobj.ElementAt(6).name, listobj.ElementAt(7).name, listobj.ElementAt(8).name, listobj.ElementAt(9).name, listobj.ElementAt(10).name, listobj.ElementAt(11).name, listobj.ElementAt(12).name, listobj.ElementAt(13).name, listobj.ElementAt(14).name };

int[] y = { listobj.ElementAt(0).goal, listobj.ElementAt(1).goal, listobj.ElementAt(2).goal, listobj.ElementAt(3).goal, listobj.ElementAt(4).goal, listobj.ElementAt(5).goal, listobj.ElementAt(6).goal, listobj.ElementAt(7).goal, listobj.ElementAt(8).goal, listobj.ElementAt(9).goal, listobj.ElementAt(10).goal, listobj.ElementAt(11).goal, listobj.ElementAt(12).goal, listobj.ElementAt(13).goal, listobj.ElementAt(14).goal };

chart1.Series[seriesname].LegendText = "Player Statistics";

chart1.Series[seriesname].Points.DataBindXY(x, y);

//saving scrrenshot

this.chart1.SaveImage("Pie chart.png", ChartImageFormat.Png);

}

private void label7\_Click(object sender, EventArgs e)

{

}

private void label10\_Click(object sender, EventArgs e)

{

}

private void label9\_Click(object sender, EventArgs e)

{

}

private void label8\_Click(object sender, EventArgs e)

{

}

private void label11\_Click(object sender, EventArgs e)

{

}

private void panel1\_Paint(object sender, PaintEventArgs e)

{

}

private void label7\_Click\_1(object sender, EventArgs e)

{

}

}

}

**Class1.cs :**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Threading.Tasks;

namespace project1.\_3

{

public class Class1

{

public string name=null;

public int goal = 0;

public Class1()

{

}

}

}

**CD HERE**