

# Informatics College Pokhara



informatics  
college pokhara

**Application Development**

**CS6004NI**

**Course Work 1**

**Submitted By: Rika Gurung**  
**London Met ID:** Enter ID Here

**Submitted To:** Ishwor Sapkota  
Module Leader

Component Grade and Comments	
<b>A. Implementation of Application</b>	
<b>User Interface and proper controls used for designing</b>	missing controls in the interface
<b>Manual data entry or import from csv</b>	not properly saved or imported data
<b>Data Validation</b>	Only basic validation
<b>Enrollment Report &amp; weekly report in tabular format</b>	very poorly executed reports and data not shown accurately
<b>Course wise enrollment report &amp; Chart display</b>	Very poorly designed and only contains one report format with in appropriate data
<b>Algorithm used for sorting &amp; proper sorting of data</b>	Default sorting provided by .net is used
<b>B. Documentation</b>	
<b>User Manual for running the application</b>	User Manual is below average. Is textual only.

<b>Application architecture &amp; description of the classes ad methods sued</b>	average work with very limited explanation of the classes and methods used
<b>Flow chart, algorithms and data sctructures used</b>	average work with very limited explanation and missing diagramatic representation.
<b>Reflective essay</b>	Very poorly written

### C. Programming Style

<b>Clarity of code,Popper Naming convention &amp; comments</b>	very poorly written code and no comments at all
<b>System Usability</b>	unusable system

<b>Overall Grade:</b>	<b>F+</b>
-----------------------	-----------

### Overall Comment:

Code should be self explainable with less comments. Need some proper naming of the component and require to add comments on required area.
In overall the code is working and all the functionality seems working and system can be used

# Informatics College Pokhara



## Application Development

### CS6004NP

#### Coursework 1

**Submitted By:**

Student Name: Rika Gurung  
Student ID: 17030740  
Group: L3C2  
Date: 10-JAN-2020

**Submitted To:**

Mr Ishwor Sapkota  
Module Leader  
Application Development

## Contents

<b>1.Introduction .....</b>	<b>1</b>
<b>2. User Manual.....</b>	<b>2</b>
<b>3. Journals.....</b>	<b>11</b>
<b>4. System Architecture.....</b>	<b>12</b>
<b>5. Class Diagram.....</b>	<b>13</b>
<b>6. FlowChart.....</b>	<b>14</b>
<b>7. Sorting Algorithm.....</b>	<b>15</b>
<b>8. Conclusion.....</b>	<b>18</b>
<b>9. Bibliography .....</b>	<b>19</b>
<b>10. Appendix .....</b>	<b>20</b>

## Table of Figure

Figure 1First login .....	2
Figure 2 Login by entering username and password .....	3
Figure 3 Login Successful.....	3
Figure 4 Main Page.....	4
Figure 5 Adding detail .....	5
Figure 6 Save detail .....	5
Figure 7 Importing file .....	6
Figure 8 Import file .....	6
Figure 9 click on sort name .....	7
Figure 10 Sort name .....	7
Figure 11 CClick on date.....	8
Figure 12 sort registration date .....	8
Figure 13 Weekly report.....	9
Figure 14 Showing weekly report.....	9
Figure 15 Chart show.....	10
Figure 16 System Architecture.....	12
Figure 17 Class Diagram .....	13
Figure 18 Flowchart .....	14
Figure 19 Bubble sort.....	16
Figure 20 Sorting .....	17

## **1.Introduction**

The project is about keeping student record or information system .The system can manage and track record of the student's details, program enrol and registration date. Moreover, the developed system is capable to track the record of weekly students report. The system will present the chart for weekly student report.

### **Current Scenario**

Most of the schools and colleges are still using traditional method of record keeping which are unsafe and risky to maintain it for long period of time. But somehow , few of them have used digital system to maintain the record of the students. In this 21st century , use of digital system is lacking behind.

### **Proposed System**

The given project is all making digitalizes system and to tackle all the problems of using traditional system. This system is easy to use and is totally safe .

## 2. User Manual

Following are the proper screenshots which will help user to run the system.

### Login Screen

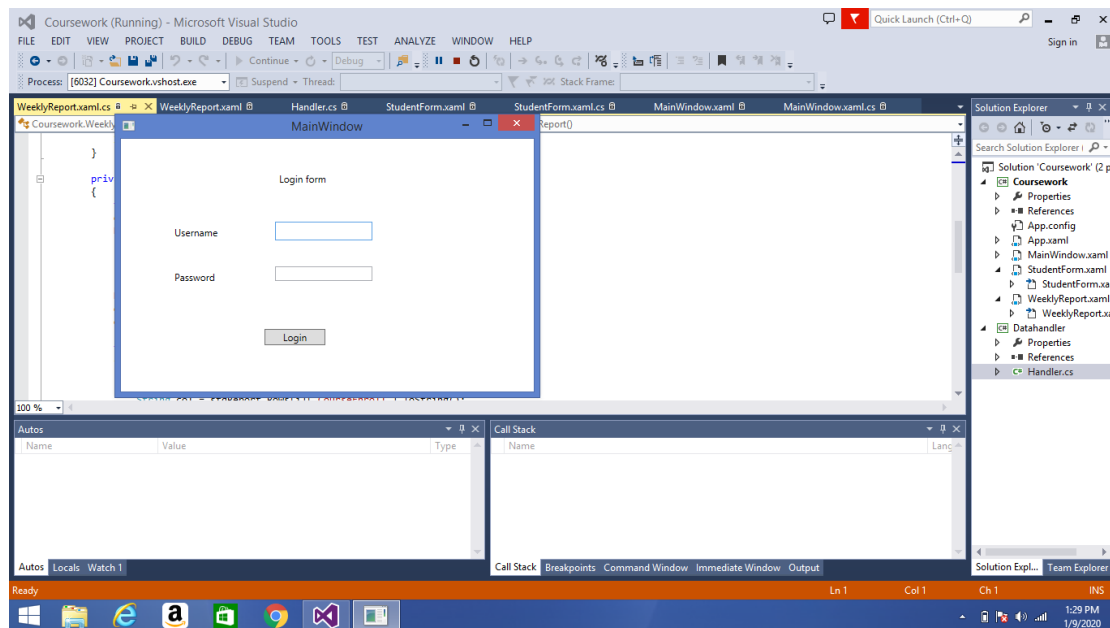


Figure 1First login

This is the first login screen of the developed system.



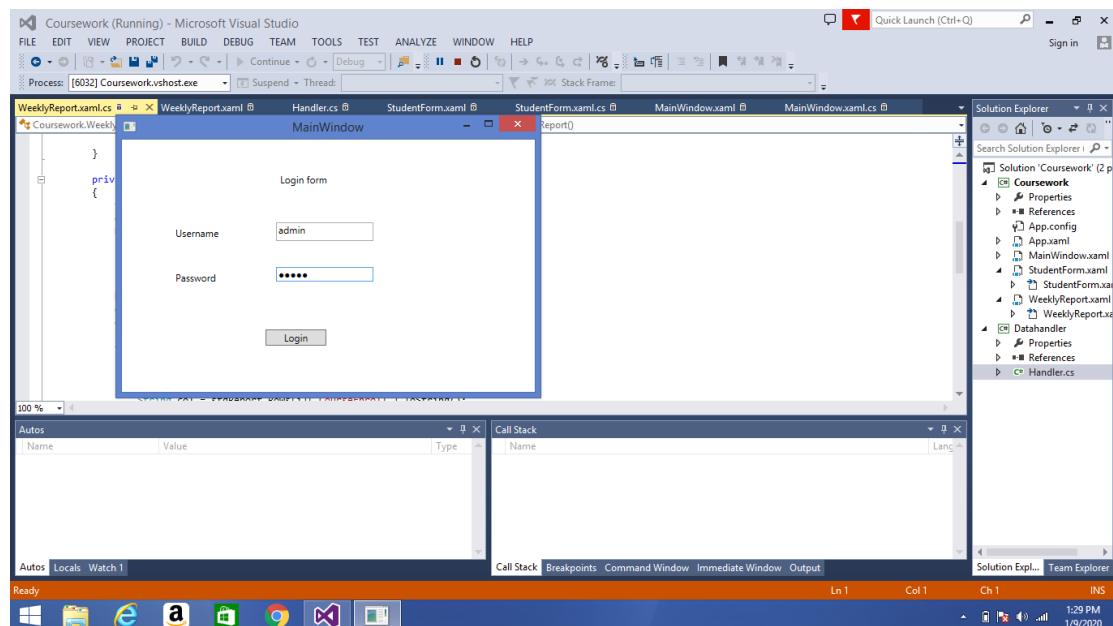


Figure 2 Login by entering username and password

The username and password for this system is admin and admin respectively..

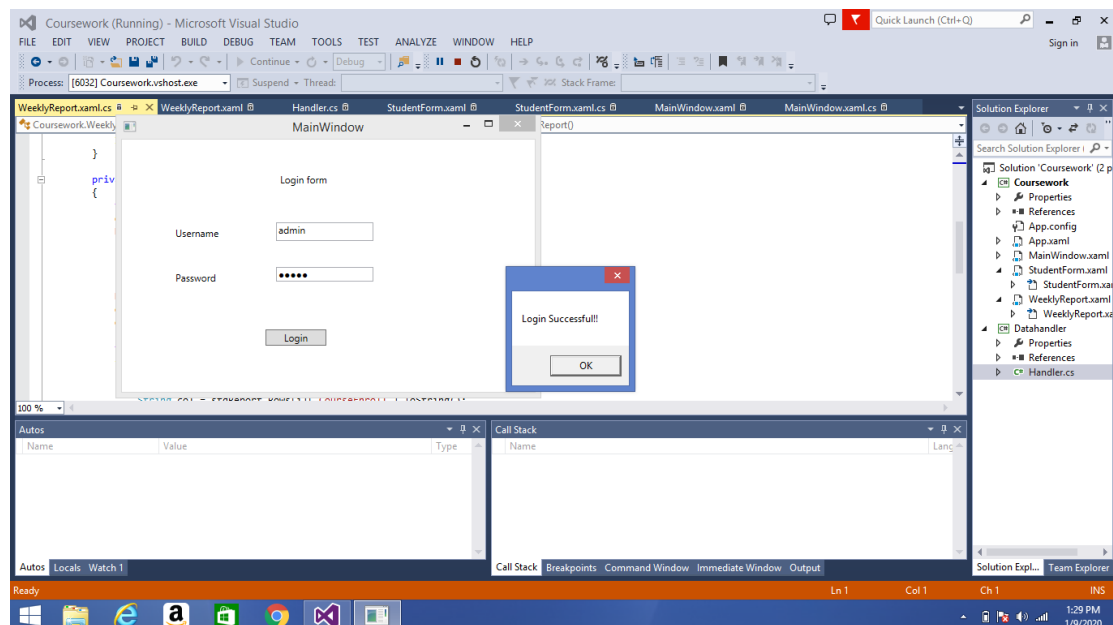


Figure 3 Login Successful

If username and password is correct then it will display login successful message.

## Main Page

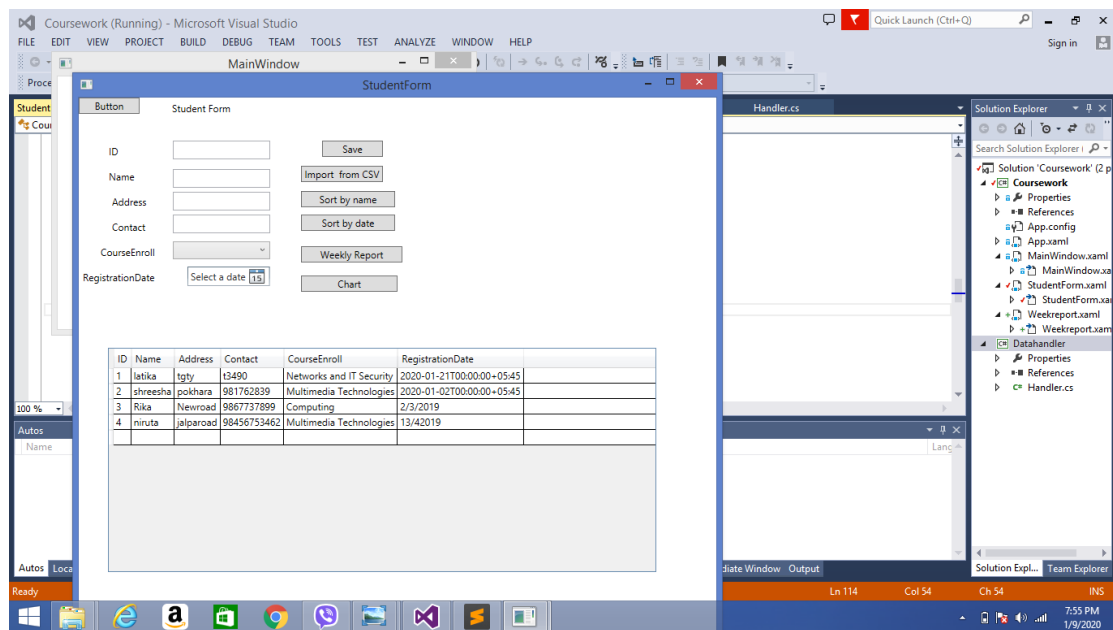


Figure 4 Main Page

After log in to the system ,user will see this home page.

## Save Button

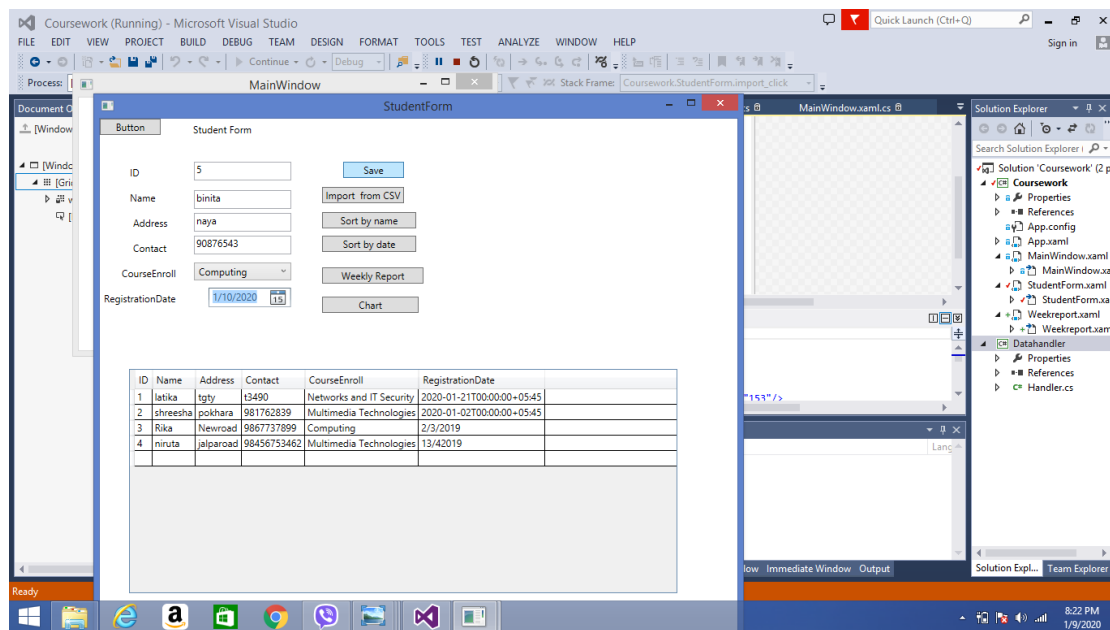


Figure 5 Adding detail

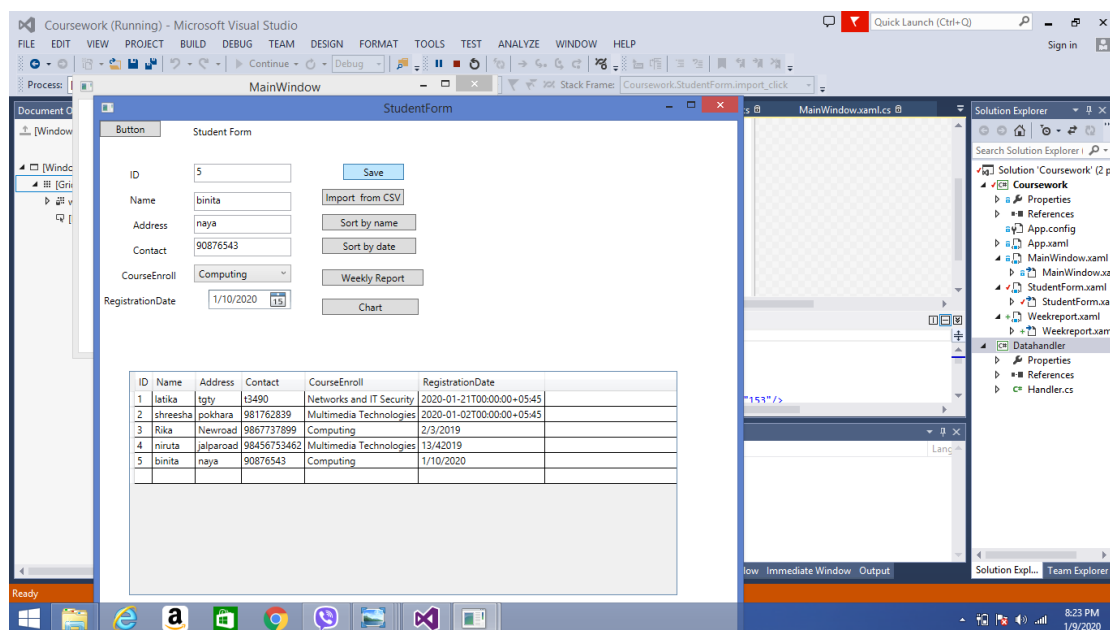


Figure 6 Save detail

The save button saves the detail of student i.e. name, contact, address, courses and date and show in grid.

## Import from CSV button

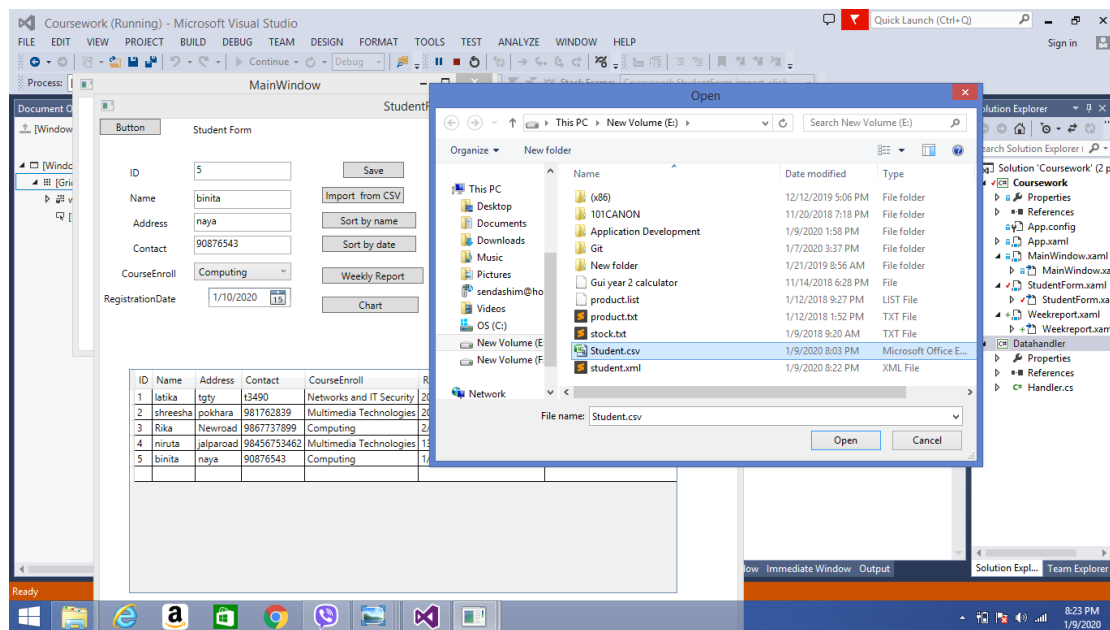


Figure 7 Importing file

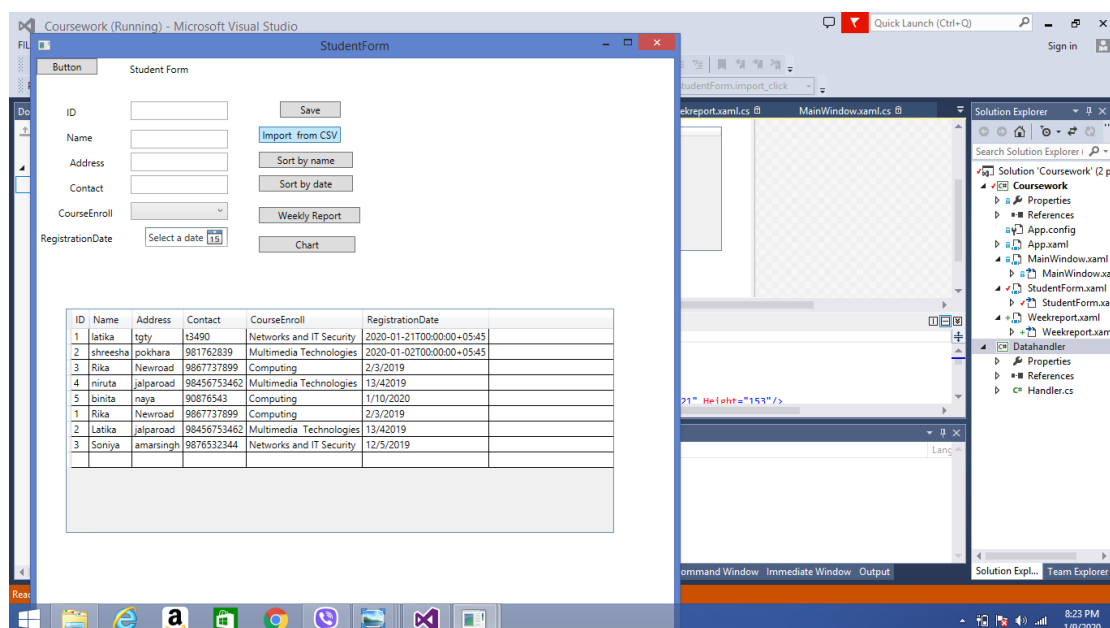


Figure 8 Import file

The import from CSV button lets user to import the CSV file from external file.

## Sort by name button

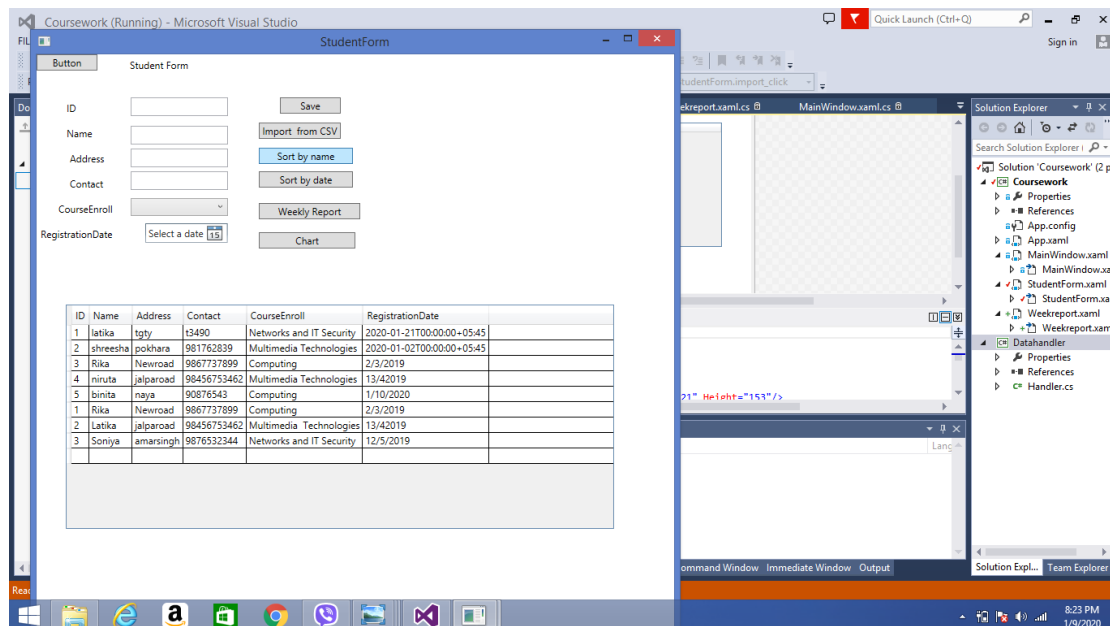


Figure 9 click on sort name

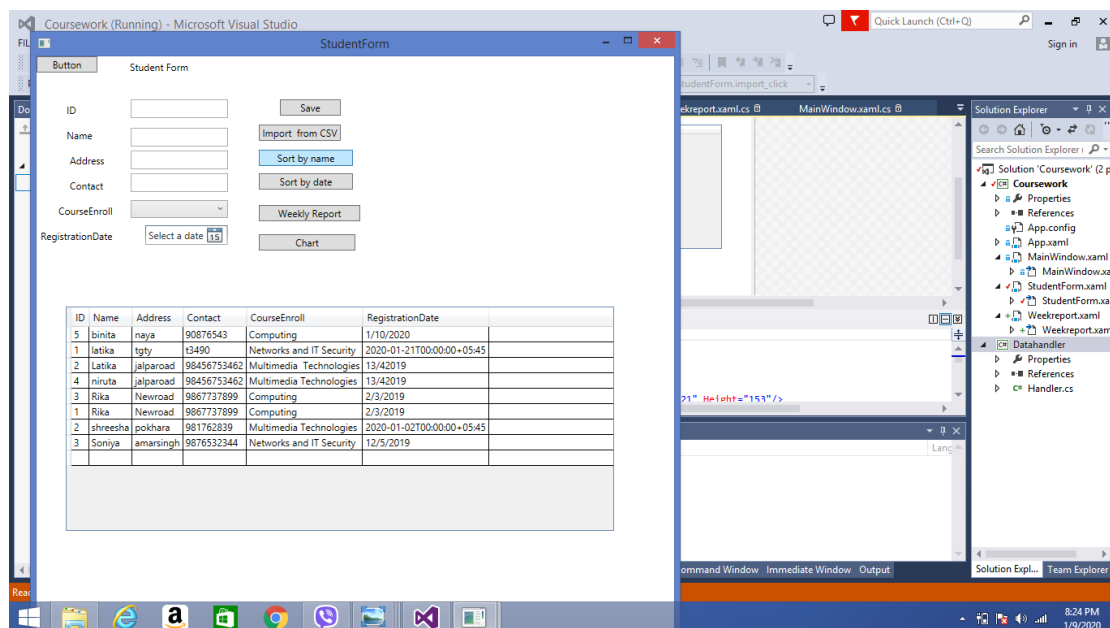


Figure 10 Sort name

The sort name button lets user to sort the student's name in ascending order based on name.

## Sort by registration date

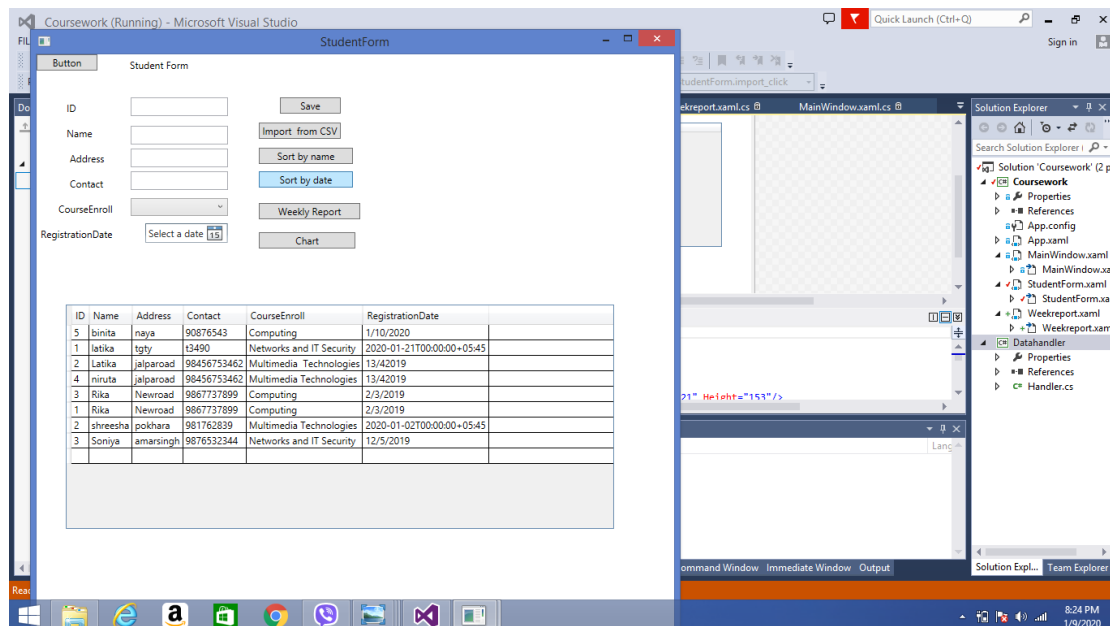


Figure 11 Click on date

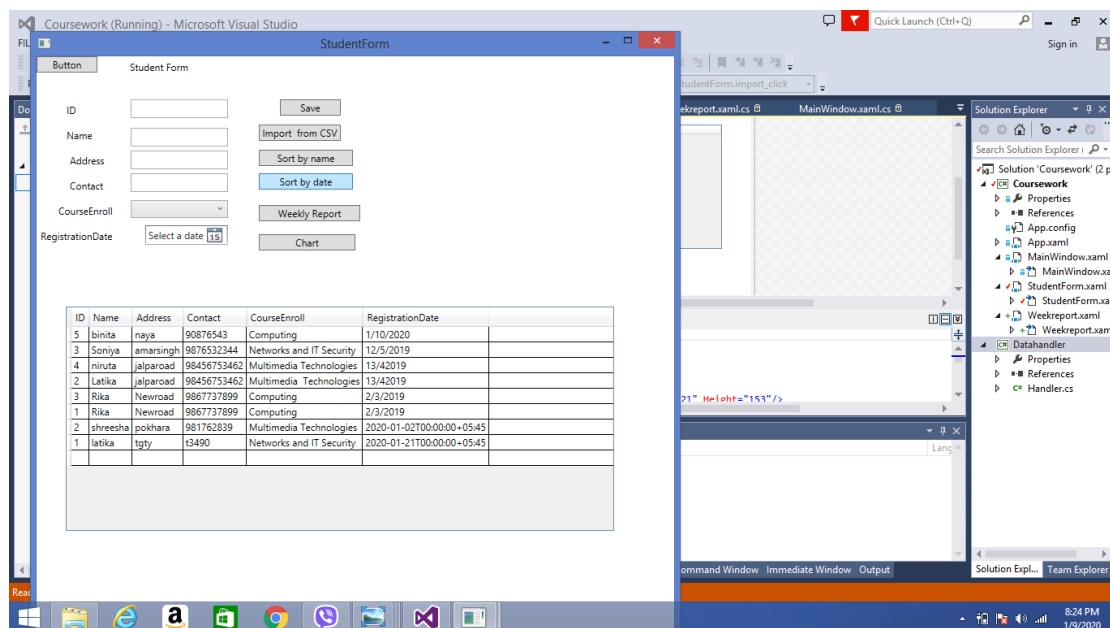


Figure 12 sort registration date

The sort name button lets user to sort the student's date in ascending order based on registration date.

## Weekly report button

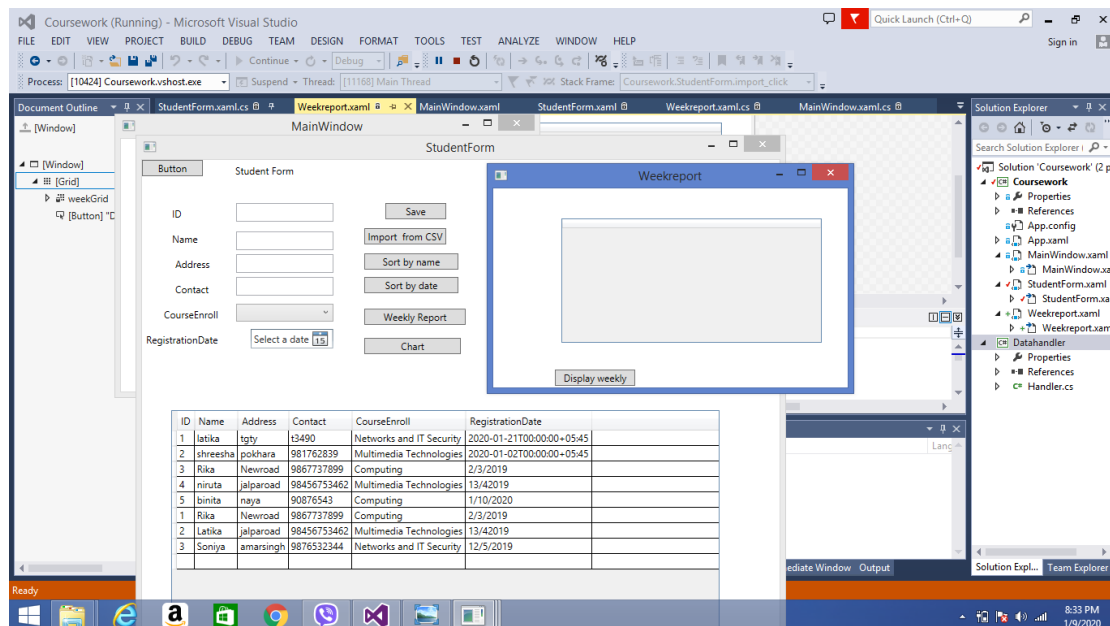


Figure 13 Weekly report

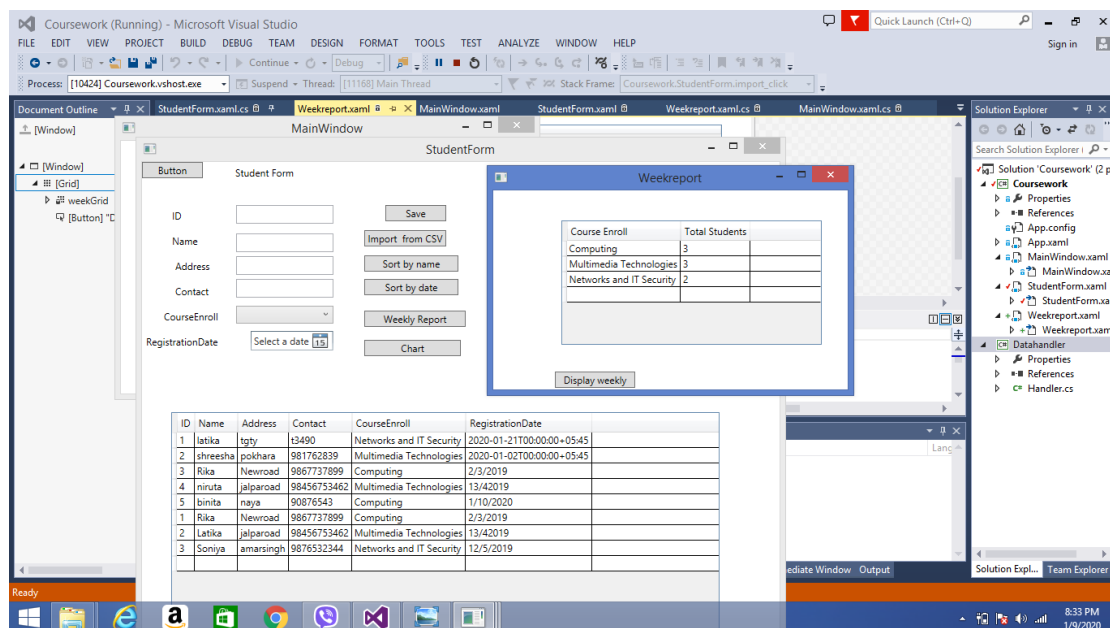


Figure 14 Showing weekly report

When weekly report button is clicked, it will display weekly Report.

## Chart button

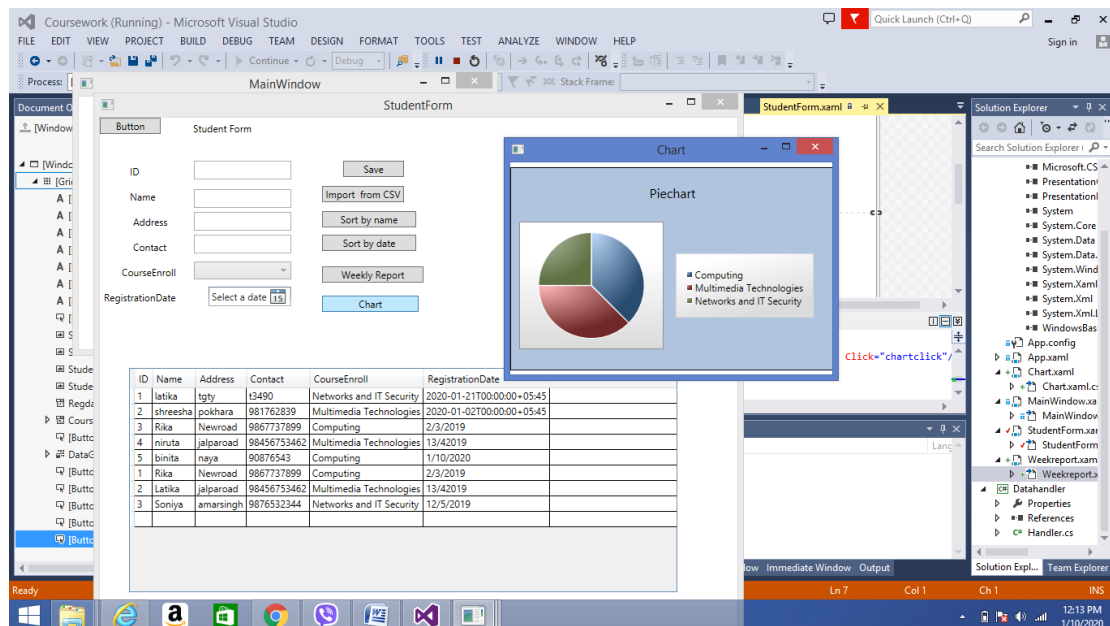


Figure 15 Chart show

When chart button is clicked on, it displays a chart. This chart shows the total course that student has enrolled which is based on weekly report.



### 3. Journals

I. With advanced technology, it seems, comes more and more paperwork. Ironically, this is exactly the opposite of the "paperless society" predicted for years. Fortunately for educators, there are some products available right now that help ease paperwork-intensive tasks. Keeping detailed records of each and every student's grades, class schedules and personal information - including medical, parental and disciplinary information - has long been the bane of educators everywhere. Now add the monumental task of keeping all such records updated district-wide. And every year seems to bring even more regulations and laws that, in turn bring even more paperwork. To keep track of this plethora of data, software firms have responded with a wide range of Student Information System (SIS) programs, gradebook software and miscellaneous administration (ARTICLE, n.d.)

II A new type of student information management system is designed to implement student information identification and management based on fingerprint identification. In order to ensure the security of data transmission, this paper proposes a data encryption method based on an improved AES algorithm. A new -box is cleverly designed, which can significantly reduce the encryption time by improving ByteSub, ShiftRow, and MixColumn in the round transformation of the traditional AES algorithm with the process of look-up table. Experimental results show that the proposed algorithm can significantly improve the encryption time compared with the traditional AES algorithm

(hindawi, n.d.)

## 4. System Architecture

Systems Architecture is a response to the conceptual and practical difficulties of the description and the design of complex systems. The following depicts about architecture diagram of developed system. At first, user needs to login to the system for which the user needs to input the correct credentials. After logging into the system with correct email and password, the system will display the main form which is the main panel of the developed system. Using the save button on the main form, the user can record the student name, student address, contact, courses and date. Using the import button, user can import file. And also user can sort the student name and date by using sorting name and date button. Moreover, the user can generate the report of the student in bargraph.

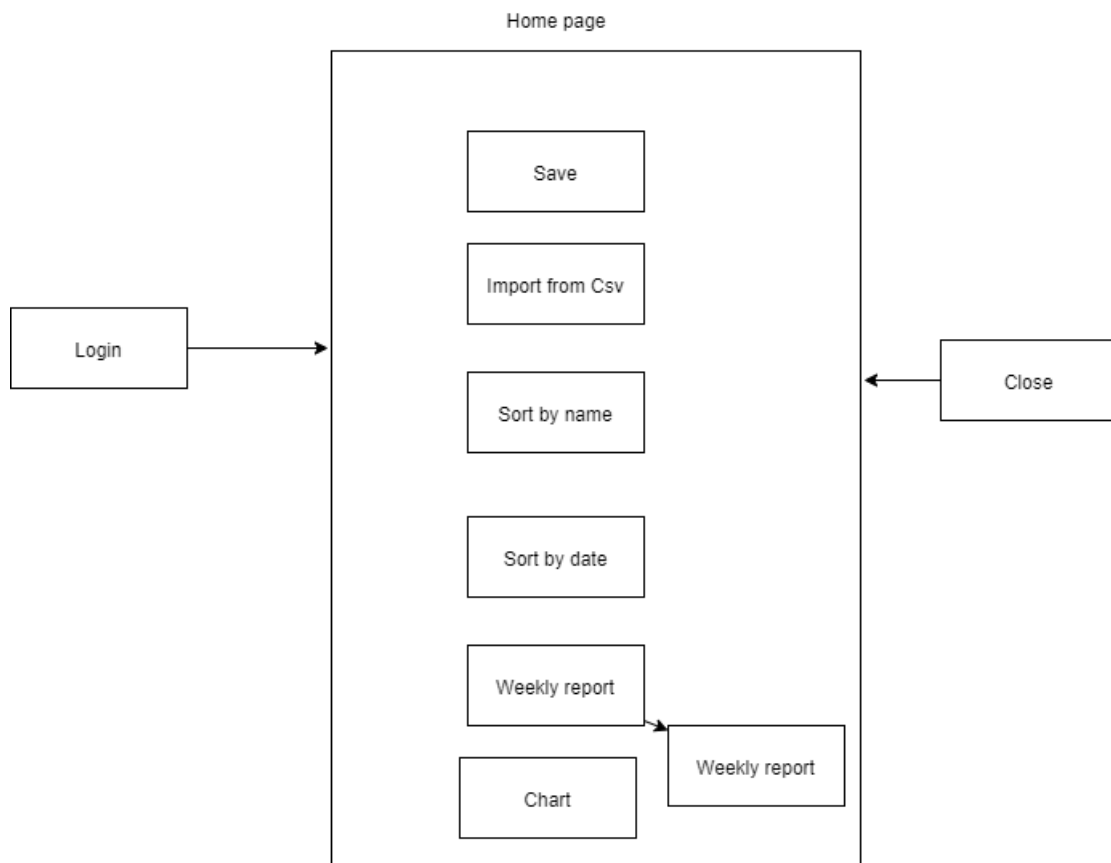


Figure 16 System Architecture

## 5. Class Diagram

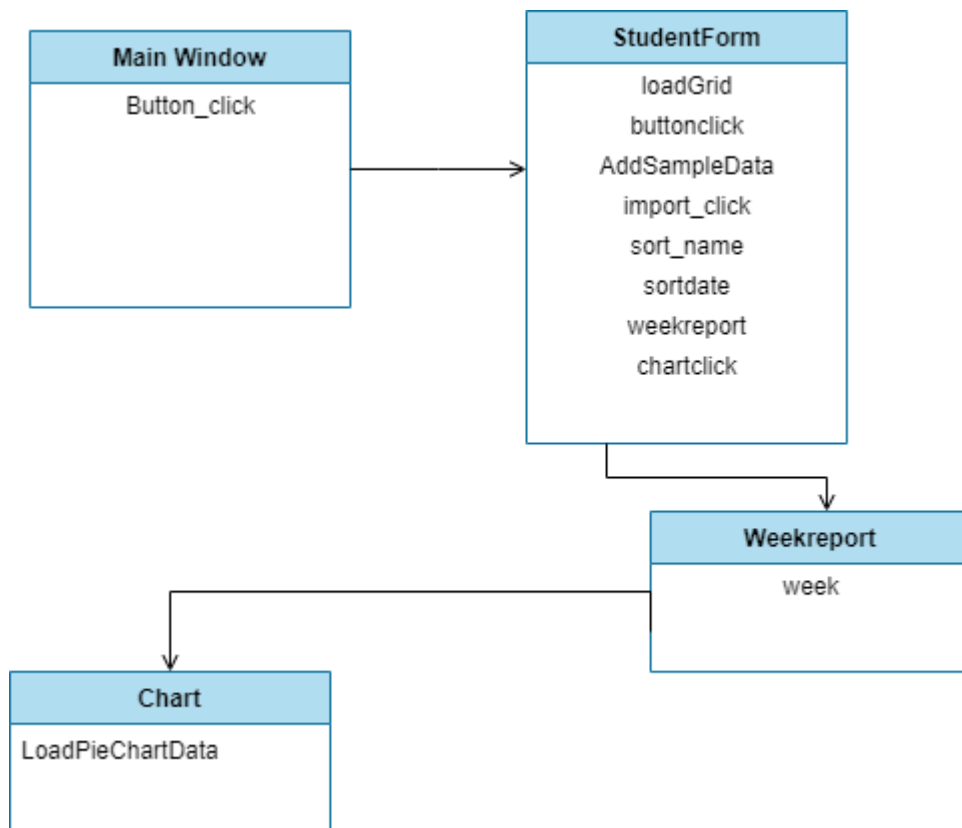


Figure 17 Class Diagram

## 6. FlowChart

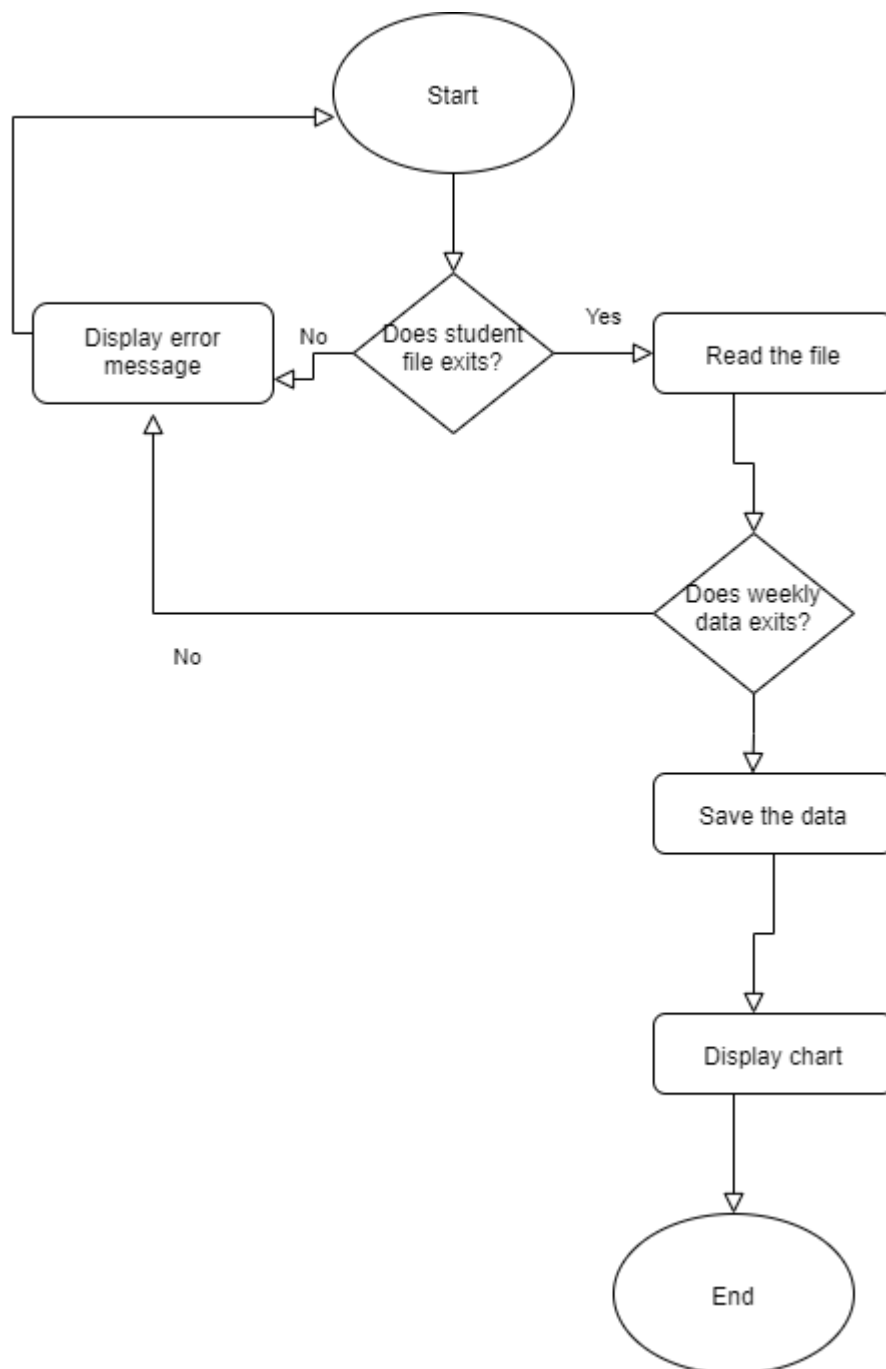


Figure 18 Flowchart

## 7. Sorting Algorithm

Bubble Sort is a simple algorithm which is used to sort a given set of  $n$  elements provided in form of an array with  $n$  number of elements. Bubble Sort compares all the element one by one and sort them based on their values. If the given array has to be sorted in ascending order, then bubble sort will start by comparing the first element of the array with the second element, if the first element is greater than the second element, it will swap both the elements, and then move on to compare the second and the third element, and so on. If we have total  $n$  elements, then we need to repeat this process for  $n-1$  times.

It is known as bubble sort, because with every complete iteration the largest element in the given array, bubbles up towards the last place or the highest index, just like a water bubble rises up to the water surface. Sorting takes place by stepping through all the elements one-by-one and comparing it with the adjacent element and swapping them if required.

### Implementing Bubble Sort Algorithm

Following are the steps involved in bubble sort (for sorting a given array in ascending order):

1. Starting with the first element (index = 0), compare the current element with the next element of the array.
2. If the current element is greater than the next element of the array, swap them.
3. If the current element is less than the next element, move to the next element. Repeat Step 1.

Let's consider an array with values {5, 1, 6, 2, 4, 3}

Below, we have a pictorial representation of how bubble sort will sort the given array.

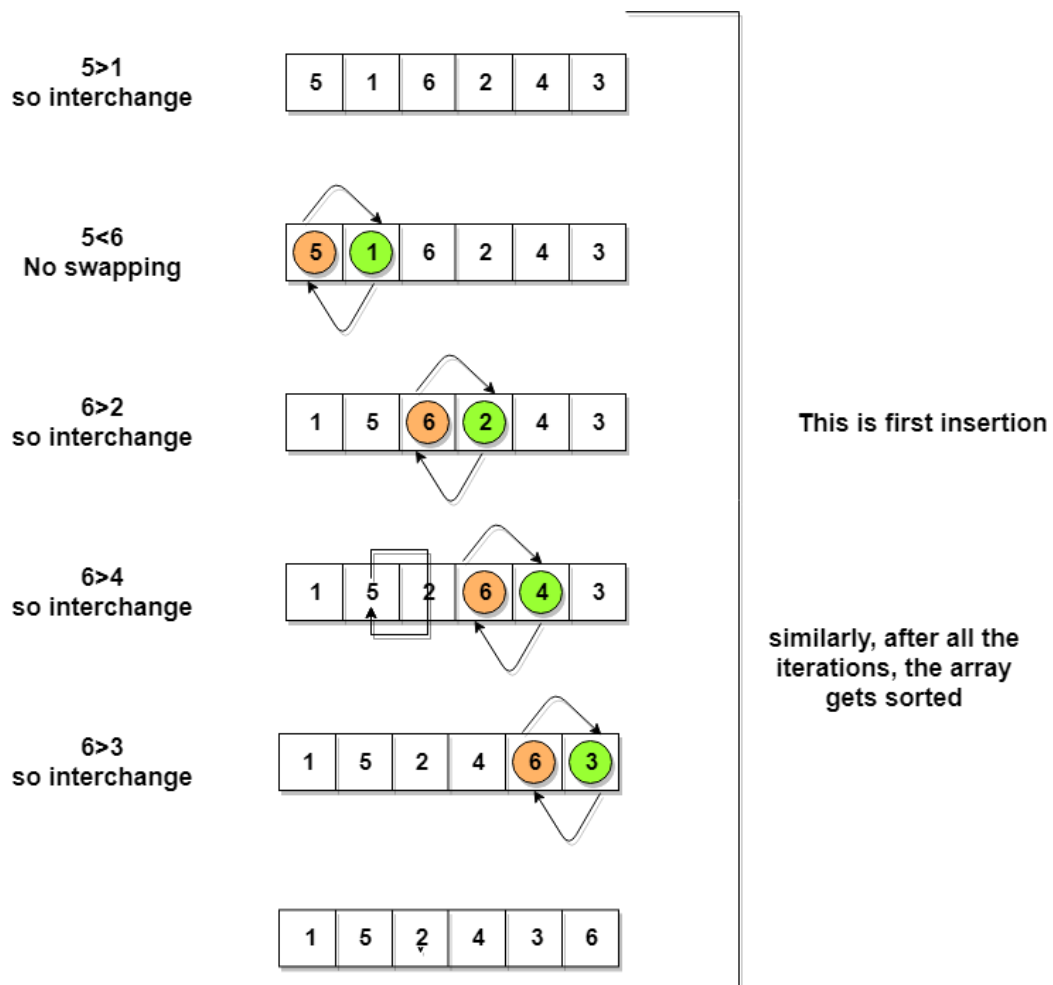


Figure 19 Bubble sort

### Bubble sort algorithm

So as we can see in the representation above, after the first iteration, 6 is placed at the last index, which is the correct position for it.

Similarly after the second iteration, 5 will be at the second last index, and so on.

### Optimized Bubble Sort Algorithm

To optimize our bubble sort algorithm, we can introduce a flag to monitor whether elements are getting swapped inside the inner for loop.

Hence, in the inner for loop, we check whether swapping of elements is taking place or not, everytime.

If for a particular iteration, no swapping took place, it means the array has been sorted and we can jump out of the for loop, instead of executing all the iterations.

Let's consider an array with values {11, 17, 18, 26, 23}

Below, we have a pictorial representation of how the optimized bubble sort will sort the given array



Figure 20 Sorting

### Optimized Bubble sort algorithm

As we can see, in the first iteration, swapping took place, hence we updated our flag value to 1, as a result, the execution enters the for loop again. But in the second iteration, no swapping will occur, hence the value of flag will remain 0, and execution will break out of loop.

The main advantage of Bubble Sort is the simplicity of the algorithm.

The space complexity for Bubble Sort is  $O(1)$ , because only a single additional memory space is required i.e. for temp variable. Also, the best case time complexity will be  $O(n)$ , it is when the list is already sorted (StudyTonight, 2020)

## **8. Conclusion**

This coursework is based on making desktop application using WPF .net framework. This assignment has been done on visual studio platform using C#. Programming. The system has login to secure the task. After log in to system user can interact to the system. A lot of researches are done to complete this coursework. I would like to thank our module leader Mr. Ishwor Sapkota for his guidance and help for these coursework.



## 9. Bibliography

ARTICLE, A. J., n.d. [Online]

Available at: <https://www.questia.com/library/journal/1G1-18213264/student-information-systems-are-integrating-more-functions>

hindawi, n.d. [Online]

Available at: <https://www.hindawi.com/journals/jece/2017/9598581/>  
[Accessed 2017].

StudyTonight, 2020. [Online]

Available at: <https://www.studytonight.com/data-structures/bubble-sort>  
[Accessed 2020].

## 10. Appendix

```
<Window x:Class="Coursework.MainWindow"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="MainWindow" Height="350" Width="525">
    <Grid>
        <Label Content="Login form" HorizontalAlignment="Left"
Margin="190,36,0,0" VerticalAlignment="Top"/>
        <Label Content="Password" HorizontalAlignment="Left"
Margin="61,157,0,0" VerticalAlignment="Top"/>
        <Label Content="Username" HorizontalAlignment="Left"
Margin="61,102,0,0" VerticalAlignment="Top"/>
        <TextBox x:Name="Username" HorizontalAlignment="Left" Height="23"
Margin="190,102,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120"/>
        <Button x:Name="Login" Content="Login" HorizontalAlignment="Left"
Margin="177,234,0,0" VerticalAlignment="Top" Width="75" Click="Button_Click"/>
        <PasswordBox x:Name="Password" HorizontalAlignment="Left"
Margin="190,157,0,0" VerticalAlignment="Top" Width="120"/>

    </Grid>
</Window>
```

### MainWindow

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Navigation;
using System.Windows.Shapes;

namespace Coursework
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();
        }

        private void Button_Click(object sender, RoutedEventArgs e)
        {
            if(Password.Password!=""&& Username.Text!="")
            {
                if(Password.Password=="admin" && Username.Text == "admin")
                {

```

```

MessageBox.Show("Login Successful!!");
StudentForm studentform = new StudentForm();
studentform.ShowDialog();

```

```

    }
}
}
}
}

```

### Chart.xaml

```

<Window x:Class="Coursework.Chart"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        xmlns:DV="clr-
namespace:System.Windows.Controls.DataVisualization;assembly=System.Windows.Con
trols.DataVisualization.Toolkit"
        xmlns:DVC="clr-
namespace:System.Windows.Controls.DataVisualization.Charting;assembly=System.Wi
ndows.Controls.DataVisualization.Toolkit"
        Title="Chart" Height="300" Width="300">
    <Grid>
        <DVC:Chart Margin="0" Title="Piechart"
Width="400" Height="250"
Background="LightSteelBlue">

            <DVC:PieSeries x:Name="Piedisplay"
IndependentValueBinding="{Binding Path=Key}" DependentValueBinding="{Binding
Path=Value}">

                </DVC:PieSeries>

            </DVC:Chart>

        </Grid>
    </Window>

```

### Chart.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace Coursework
{
    /// <summary>
    /// Interaction logic for Chart.xaml
    /// </summary>
    public partial class Chart : Window
    {
        public Chart()
        {
            InitializeComponent();
            LoadPieChartData();
        }
        private void LoadPieChartData()
        {
            var dataset = new DataSet(); // declaring new data set
            dataset.ReadXml(@"E:\student.xml"); // reading main report
            DataTable stdReport = dataset.Tables[0];
            int total_Com = 0; // assigning initial values of Course to
            int total_Mul = 0;
            int total_Net = 0;

            DataTable dt = new DataTable("tbl");
            dt.Columns.Add("Course Enroll", typeof(String)); // creating two columns
            dt.Columns.Add("Total Students", typeof(int));

            for (int i = 0; i < stdReport.Rows.Count; i++)
            {
                String col = stdReport.Rows[i]["CourseEnroll"].ToString();
                if (col == "Computing")
                {
                    total_Com++; // incrementing values of each course based on user
input
                }
                else if (col == "Multimedia Technologies")
                {
                    total_Mul++;
                }
                else if (col == "Networks and IT Security")
                {
                    total_Net++;
                }
            }

            dt.Rows.Add("Computing", total_Com); // final assign

```

```
dt.Rows.Add("Multimedia Technologies", total_Mul);
dt.Rows.Add("Networks and IT Security", total_Net);
```

```
((System.Windows.Controls.DataVisualization.Charting.PieSeries)Piedisplay).Item
sSource =
    new KeyValuePair<string,int>[] {
        new KeyValuePair<string,int>("Computing", total_Com),
        new KeyValuePair<string,int>("Multimedia Technologies", total_Mul),
        new KeyValuePair<string,int>("Networks and IT Security", total_Net)};
    }
    }
}
```

### StudentForm.xaml

```
<Window x:Class="Coursework.StudentForm"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="StudentForm" Height="748.756" Width="802.48">
    <Grid>
        <Label Content="Student Form" HorizontalAlignment="Left"
Margin="110,0,0,0" VerticalAlignment="Top"/>
        <Label Content="ID" HorizontalAlignment="Left" Margin="32,53,0,0"
VerticalAlignment="Top"/>
        <Label Content="Name" HorizontalAlignment="Left" Margin="32,84,0,0"
VerticalAlignment="Top"/>
        <Label Content="Address" HorizontalAlignment="Left" Margin="36,115,0,0"
VerticalAlignment="Top"/>
        <Label Content="Contact" HorizontalAlignment="Left" Margin="36,146,0,0"
VerticalAlignment="Top"/>
        <Label Content="RegistrationDate" HorizontalAlignment="Left"
Margin="0,208,0,0" VerticalAlignment="Top"/>
        <Label Content="CourseEnroll" HorizontalAlignment="Left"
Margin="22,177,0,0" VerticalAlignment="Top"
RenderTransformOrigin="0.784,1.224"/>
        <Button Content="Save" HorizontalAlignment="Left" Margin="300,53,0,0"
VerticalAlignment="Top" Width="75" Click="buttonclick"/>
        <TextBox x:Name="Studentid" HorizontalAlignment="Left" Height="23"
Margin="116,53,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120" />
        <TextBox x:Name="Studentname" HorizontalAlignment="Left" Height="23"
Margin="116,88,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120" />
        <TextBox x:Name="StudentAddress" HorizontalAlignment="Left" Height="23"
Margin="116,116,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120" />
        <TextBox x:Name="Studentcontact" HorizontalAlignment="Left" Height="23"
Margin="116,144,0,0" TextWrapping="Wrap" VerticalAlignment="Top" Width="120" />
        <DatePicker x:Name="Regdate" HorizontalAlignment="Left"
Margin="134,208,0,0" VerticalAlignment="Top"/>
        <ComboBox x:Name="Courseenroll" HorizontalAlignment="Left"
Margin="116,177,0,0" VerticalAlignment="Top" Width="120">
            <ComboBoxItem Content="Networks and IT Security"/>
            <ComboBoxItem Content="Multimedias Technologies"/>
            <ComboBoxItem Content="Computing"/>
        </ComboBox>
        <Button Content="Import from CSV" HorizontalAlignment="Left"
Margin="274,84,0,0" VerticalAlignment="Top" Width="101" Click="import_click"/>
        <DataGrid x:Name="DataGridXAML" HorizontalAlignment="Left"
Margin="36,308,0,0" VerticalAlignment="Top" Height="276" Width="676">
```

```

        </DataGrid>
        <Button Content="Sort by name" HorizontalAlignment="Left"
Margin="274,115,0,0" VerticalAlignment="Top" Width="116" Click="sort_name"/>
        <Button Content="Button" HorizontalAlignment="Left"
VerticalAlignment="Top" Width="75"/>
        <Button Content="Sort by date" HorizontalAlignment="Left"
Margin="274,144,0,0" VerticalAlignment="Top" Width="116" Click="sortdate"/>
        <Button Content="Weekly Report" HorizontalAlignment="Left"
Margin="274,183,0,0" VerticalAlignment="Top" Width="125"
RenderTransformOrigin="-0.056,0.919" Click="weekreport">

        </Button>
        <Button Content="Chart" HorizontalAlignment="Left" Margin="274,219,0,0"
VerticalAlignment="Top" Width="119" Click="chartclick"/>

    </Grid>
</Window>

```

### StudentForm.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
using DataHandler;
using Microsoft.Win32;

namespace Coursework
{
    /// <summary>
    /// Interaction logic for StudentForm.xaml
    /// </summary>
    public partial class StudentForm : Window
    {
        DataTable dataTable;
        public StudentForm()
        {
            InitializeComponent();
            loadGrid();
        }
        public void loadGrid()
        {
            if (File.Exists(@"E:\student.xml"))
            {

```

```

        var dataSet = new DataSet();
        dataSet.ReadXml(@"E:\student.xml");
        dataTable = dataSet.Tables["Student"];
        DataGridXAML.ItemsSource =
dataSet.Tables["Student"].DefaultView;
    }
}

public class Student
{
    public string ID { get; set; }
    public string Name { get; set; }
    public string Address { get; set; }

    public string Contact { get; set; }
    public string CourseEnroll { get; set; }
    public string RegistrationDate { get; set; }
}

private void buttonclick(object sender, RoutedEventArgs e)
{
    var handler = new Handler();
    var dataSet = new DataSet();

    if (File.Exists(@"E:\student.xml"))
    {
        dataSet.ReadXml(@"E:\student.xml");
    }
    else
    {
        dataSet = handler.CreateDataSet();
    }
    AddSampleData(dataSet);
    dataSet.WriteXml(@"E:\student.xml");
    loadGrid();
}

private void AddSampleData(DataSet dataSet)
{
    var dr1 = dataSet.Tables["Student"].NewRow();
    dr1["ID"] = Studentid.Text;
    dr1["Name"] = Studentname.Text;
    dr1["Address"] = StudentAddress.Text;
    dr1["Contact"] = Studentcontact.Text;
    dr1["CourseEnroll"] = Courseenroll.Text;
    dr1["RegistrationDate"] = Regdate.Text;
    dataSet.Tables["Student"].Rows.Add(dr1);
}

```

```

private void import_click(object sender, RoutedEventArgs e)
{
    var dataSet = new DataSet();
    dataSet.ReadXml(@"E:\student.xml");
    OpenFileDialog openFileDialog = new OpenFileDialog();
    if (openFileDialog.ShowDialog() == true)
    {
        string filePath = openFileDialog.FileName;
        //read all std from file code copy

        using (var reader = new StreamReader(filePath))
        {
            reader.ReadLine();
            while (!reader.EndOfStream)
            {
                var line = reader.ReadLine();
                var values = line.Split(',');
                var newRow = dataSet.Tables["Student"].NewRow();
                newRow["ID"] = values[0];
                newRow["Name"] = values[1];
                newRow["Address"] = values[2];
                newRow["Contact"] = values[3];
                newRow["CourseEnroll"] = values[4];
                newRow["RegistrationDate"] = values[5];
                dataSet.Tables["Student"].Rows.Add(newRow);

                dataSet.WriteXml(@"E:\student.xml");
            }
        }
    }
}

private void sort_name(object sender, RoutedEventArgs e)
{
    dataTable.DefaultView.Sort = "Name ASC";
    DataGridXAML.DataContext = dataTable.DefaultView;
}

private void sortdate(object sender, RoutedEventArgs e)
{
    dataTable.DefaultView.Sort = "RegistrationDate ASC";
    DataGridXAML.DataContext = dataTable.DefaultView;
}

private void weekreport(object sender, RoutedEventArgs e)
{
    Weekreport weekreport = new Weekreport();
    weekreport.ShowDialog();
}

private void chartclick(object sender, RoutedEventArgs e)
{
    Chart chart = new Chart();
    chart.Show();
}
}
}

```



## Weekreport.xaml

```
<Window x:Class="Coursework.Weekreport"
        xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
        xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
        Title="Weekreport" Height="283.566" Width="453.358">
    <Grid>
        <DataGrid x:Name="weekGrid" Margin="51,37,0,0" VerticalAlignment="Top"
Width="321" Height="153"/>
        <Button Content="Display weekly" HorizontalAlignment="Left"
Margin="76,223,0,0" VerticalAlignment="Top" Width="99" Click="week"/>
    </Grid>
```

## Weekreport.xaml.cs

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace Coursework
{
    /// <summary>
    /// Interaction logic for Weekreport.xaml
    /// </summary>
    public partial class Weekreport : Window
    {
        public Weekreport()
        {
            InitializeComponent();
        }

        private void week(object sender, RoutedEventArgs e)
        {
            var dataset = new DataSet(); // declaring new data set
            dataset.ReadXml(@"E:\student.xml"); // reading main report
            DataTable stdReport = dataset.Tables[0];
            int total_Com = 0; // assigning initial values of Course to
            int total_Mul = 0;
            int total_Net = 0;

            DataTable dt = new DataTable("tbl");
```

```

columns
    dt.Columns.Add("Course Enroll", typeof(String)); // creating two
    dt.Columns.Add("Total Students", typeof(int));

    for (int i = 0; i < stdReport.Rows.Count; i++)
    {

        String col = stdReport.Rows[i]["CourseEnroll"].ToString();
        if (col == "Computing")
        {
            total_Com++; // incrementing values of each course based
on user input
        }
        else if (col == "Multimedia Technologies")
        {
            total_Mul++;
        }
        else if (col == "Networks and IT Security")
        {
            total_Net++;
        }
    }

    dt.Rows.Add("Computing", total_Com); // final assign
    dt.Rows.Add("Multimedia Technologies", total_Mul);
    dt.Rows.Add("Networks and IT Security", total_Net);

    weekGrid.ItemsSource = dt.DefaultView; // is the name of data grid
    }
}

```

**Handler.cs**

```

using System;
using System.Data;

namespace DataHandler
{
    public class Handler
    {
        public DataSet CreateDataSet()
        {
            var ds = new DataSet();
            ds.Tables.Add(CreateCourseTable());
            ds.Tables.Add(CreateStudentTable());
            ds.Tables.Add(CreateStudentReportTable());

            ForeignKeyConstraint courseWorkFK = new
            ForeignKeyConstraint("courseWorkFK",
            ds.Tables["Course"].Columns["ID"],
            ds.Tables["Student"].Columns["CourseEnroll"]);
            courseWorkFK.DeleteRule = Rule.None;
            ds.Tables["Student"].Constraints.Add(courseWorkFK);
            return ds;
        }

        private DataTable CreateStudentTable()
        {
            var dt = new DataTable("Student");
            DataColumn dataColumn = new DataColumn("ID", typeof(int));
            dataColumn.AutoIncrement = true;
            dataColumn.AutoIncrementSeed = 1;
            dataColumn.AutoIncrementStep = 1;

            dt.Columns.Add(dataColumn);

            dt.Columns.Add("Name", typeof(string));
            dt.Columns.Add("Address", typeof(string));
            dt.Columns.Add("ContactNo", typeof(string));
            dt.Columns.Add("CourseEnroll", typeof(int));
            dt.Columns.Add("RegistrationDate", typeof(DateTime));
            //dt.Columns.Add("PermanentAddress", typeof(string));
            //dt.Columns.Add("ParentsName", typeof(string));
            //dt.Columns.Add("ParentsContact", typeof(string));
            //dt.Columns.Add("", typeof(string));
            //dt.Columns.Add("Address", typeof(string));
            //dt.Columns.Add("Address", typeof(string));
            //dt.Columns.Add("Address", typeof(string));

            dt.PrimaryKey = new DataColumn[] { dt.Columns["ID"] };
            return dt;
        }

        private DataTable CreateCourseTable()
        {
            var dt = new DataTable("Course");
            DataColumn dataColumn = new DataColumn("ID", typeof(int));
            dataColumn.AutoIncrement = true;
            dataColumn.AutoIncrementSeed = 1;
            dataColumn.AutoIncrementStep = 1;
            dt.Columns.Add(dataColumn);
        }
    }
}

```

```
        dt.Columns.Add("Name", typeof(string));
        dt.Columns.Add("DisplayText", typeof(string));
        // dt.Columns.Add("CourseDuration", typeof(string));

        dt.PrimaryKey = new DataColumn[] { dt.Columns["ID"] };
        return dt;
    }

    private DataTable CreateStudentReportTable()
    {
        var dt = new DataTable("StudentReport");
        DataColumn dataColumn = new DataColumn("ID", typeof(int));
        dataColumn.AutoIncrement = true;
        dataColumn.AutoIncrementSeed = 1;
        dataColumn.AutoIncrementStep = 1;

        dt.Columns.Add(dataColumn);

        dt.Columns.Add("RegNo", typeof(string));
        dt.Columns.Add("Name", typeof(string));
        dt.Columns.Add("Address", typeof(string));
        dt.Columns.Add("ContactNo", typeof(string));
        dt.Columns.Add("CourseEnroll", typeof(int));
        dt.Columns.Add("RegistrationDate", typeof(DateTime));

        //dt.PrimaryKey = new DataColumn[] { dt.Columns["ID"] };
        return dt;
    }
}
```