

Informatics College Pokhara



informatics
college pokhara

Application Development

CS6004NI

Course Work 1

Submitted By: Kapil Raj Gurung
London Met ID: Enter ID Here

Submitted To: Ishwor Sapkota
Module Leader

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

Application architecture & description of the classes ad methods sued	average work with very limited explanation of the classes and methods used
Flow chart, algoriathms and data sctructures used	average work with very limited explanation and missing diagramatic representation.
Reflective essay	Average work with un clear learnings, experience or findings.

C. Programming Style

Clarity of code,Popper Naming convention & comments	very poorly written code and no comments at all
System Usability	very poorly developed application

Overall Grade:	E+
-----------------------	-----------

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



Module Code & Module Title

CS6004NP Application Development

Assessment Weightage & Type

30% Individual Coursework

Year and Semester

2019-20 Autumn

Name: Kapil Raj Gurung

College ID: NP04CP4A170013

University ID: 17030714

Table of Contents

1. Introduction	1
1.1 Current Scenario.....	1
1.2 Proposed System	2
2. System Overview.....	2
2.1 User Manual	2
2.2 Architecture Diagram	10
2.3 Functionality.....	10
2.4 Class Diagram	11
2.5 Flowchart Diagram.....	16
2.6 Algorithms of Reports	17
3.Sorting Algorithm	18
4. Reflection	19
5. Conclusion	20
6. References	21
7. Appendix	22

Abstract

In today's generation the technology has been on the top where all the work is done in digitalized system. This coursework is also done in Visual Studio Platform.

This project is about Student Information system as the topic given in our coursework. This system is made to reduce the paper work system into the modern system. This system helps to reduce time and difficulties seen on the old system. This system enters the student information, entry details, keeps report of daily and weekly data.

This is an individual course work for the module "Application Development" for Student Information System which is developed using Visual Studio Platform using C# language. The coursework is released in the week 5 and it is supposed to be submitted in the 11th week.

List of Figures

Figure 1: Login Screen	2
Figure 2: Incorrect Username and Password.....	3
Figure 3: Main Page	4
Figure 4: Students details Added Button 1	5
Figure 5: Students details Added Button 2.....	5
Figure 6: Retrieve	6
Figure 7: Sort by name	6
Figure 8: Sort by date	7
Figure 9: Import from CSV 1	7
Figure 10: Import from CSV 2	8
Figure 11: Enrolled Students	8
Figure 12: Total Students Chart	9
Figure 13: Architecture Diagram of the system	10
Figure 14: Class Diagram of Student Information System	11
Figure 15: Flowchart.....	16

List of Tables

Table 1: Class Diagram of Login 12

Table 2: Class Diagram of MainWindow 13

Table 3: Class Diagram of Chart 14

Table 4: Class Diagram of App 14

Table 5: Class Diagram of Resources..... 15

Table 6: Class Diagram of Settings..... 15

1. Introduction

The designed system is student Information System. The system is majorly designed developed and test under various circumstances. The features and functions that are required by schools are almost fulfilled by the developed system. It comprises of features like inputting details like ID number, name, address, contact no, course enrol, registration date of students etc. Furthermore, there is a feature to view weekly tabular report showing total number of students enrolled so far in each program offered by the institution and also shown in chart.

All this information should be secured through the process that shouldn't be affected by any factors. I have developed this application for recording the student's information in easier way. Users can add or view the student's details and also watch the weekly chart of the visitor. The main key functional features are: Login, save student's details, retrieve enrolled student information etc. After this successful system, user will be able to record the information in easier way and data loss is almost low.

1.1 Current Scenario

Currently we are in the old-fashioned time where till now the paper works are done in every system. It is very difficult to do such system on the paper works which consumes time, effort etc. As, we have seen in most of the schools we have to fill up the form while enrolling so, later on the form can be damaged, lost, tear etc. To reduce such problems new system for this school is done which is digitalized system. The new digitalized system is easy to use and keep record, which is helpful in daily life, also safe to store record for long period of time. This new digitalized system is made to replace the old fashioned paper work.

1.2 Proposed System

The proposed system is digitized system which is specially designed to overcome problem mentioned above. The system ensures security with the presence of login section. Entry of data and display of data have been made easy with the presence of easy user-interface.

2. System Overview

2.1 User Manual

1.First of all, start the program then the below screen of the login page will open which ask for the Username and Password. If the user enters the correct username and password the user will get access for next steps. If the user enter incorrect username and password the user can't login to the program.

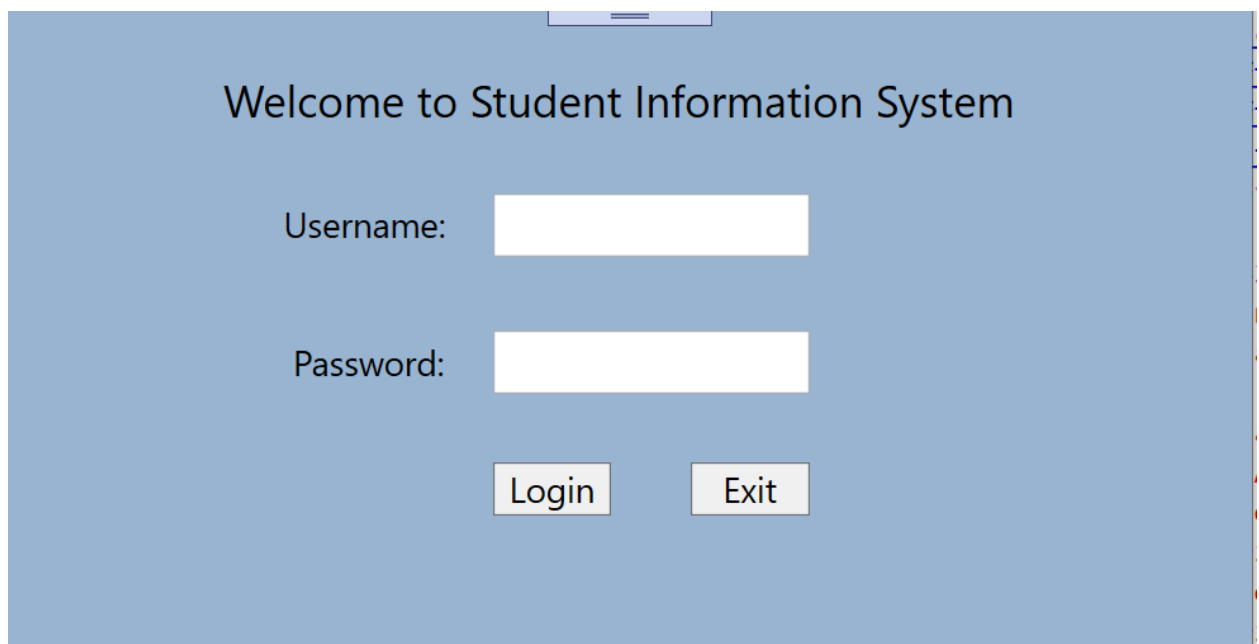
The image shows a login screen with a blue background. At the top, the text "Welcome to Student Information System" is displayed in a large, black, sans-serif font. Below this, there are two input fields. The first is labeled "Username:" and the second is labeled "Password:". Both labels are in a black, sans-serif font. The input fields are white with a thin black border. Below the input fields, there are two buttons: "Login" and "Exit". Both buttons are white with a thin black border and a slight shadow. The "Login" button is on the left and the "Exit" button is on the right.

Figure 1: Login Screen

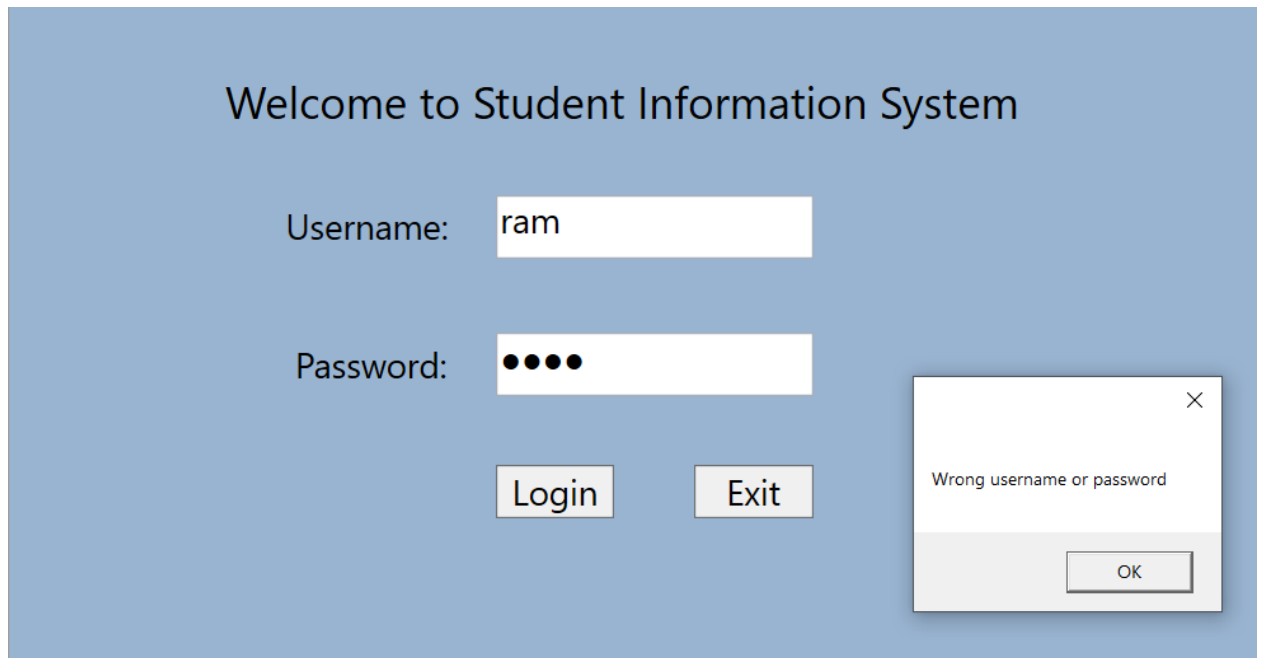


Figure 2: Incorrect Username and Password

2. After entering the correct Username and Password, the main page is shown.

The screenshot displays the main interface of a student management application. On the left side, there is a form for adding new student records with the following fields: 'Reg Number' (containing '11'), 'Name', 'Address', 'Contact', 'Program Enroll' (a dropdown menu), and 'Registration Date' (a date picker set to '15'). Below these fields is an 'Add' button. The central part of the page contains two large, empty rectangular boxes, likely intended for displaying student lists or charts. To the right of the top box are buttons for 'Retrieve', 'Sort by name', and 'sort by date'. Below the bottom box are buttons for 'Enrolled Students' and 'Show Total Student Chart'. On the far right, there is a 'Next' button.

Figure 3: Main Page

This is the main page of the system as all the records of students are recorded according to the forms given in the task. The system has different menus in the main page which have Add, Enrolled students, retrieve students' details, sort and show total students chart etc. Each of the menu has its own functions.

3. This system has “add” menu strip which has its own functions to add students in the table. After adding the student’s details message is popped to show students details have been added.

Reg Number: 2

Name : Sita

Address : Mahendrapool

Contact : 980523697

Program Enroll: BBA

Registration Date: 1/2/2020

Add

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
1	1	Ram	Pokhara	980412586	BIT	1/1/2020

Retrieve Sort by name sort by date

Student details added

OK

Figure 4: Students details Added Button 1

4. After clicking on the “ok” button in the student’s details added popped up message the student’s data has been added in the table which is highlighted in the figure.

Reg Number: 3

Name :

Address :

Contact :

Program Enroll: BBA

Registration Date: 1/2/2020

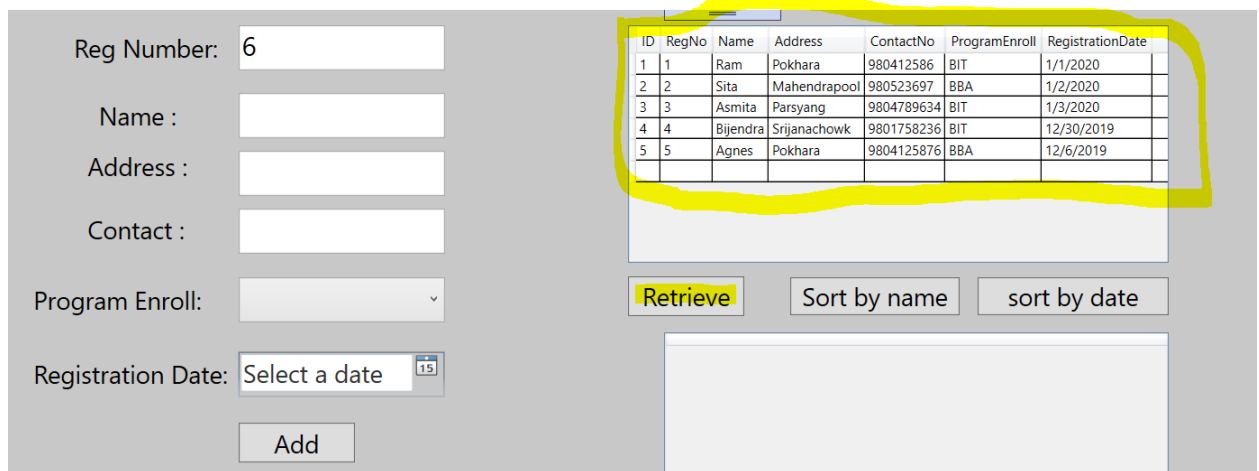
Add

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
1	1	Ram	Pokhara	980412586	BIT	1/1/2020
2	2	Sita	Mahendrapool	980523697	BBA	1/2/2020

Retrieve Sort by name sort by date

Figure 5: Students details Added Button 2

5. After clicking on the “retrieve” button it will show all the details of added students up to the latest date so far.



Reg Number: 6

Name :

Address :

Contact :

Program Enroll:

Registration Date: Select a date 15

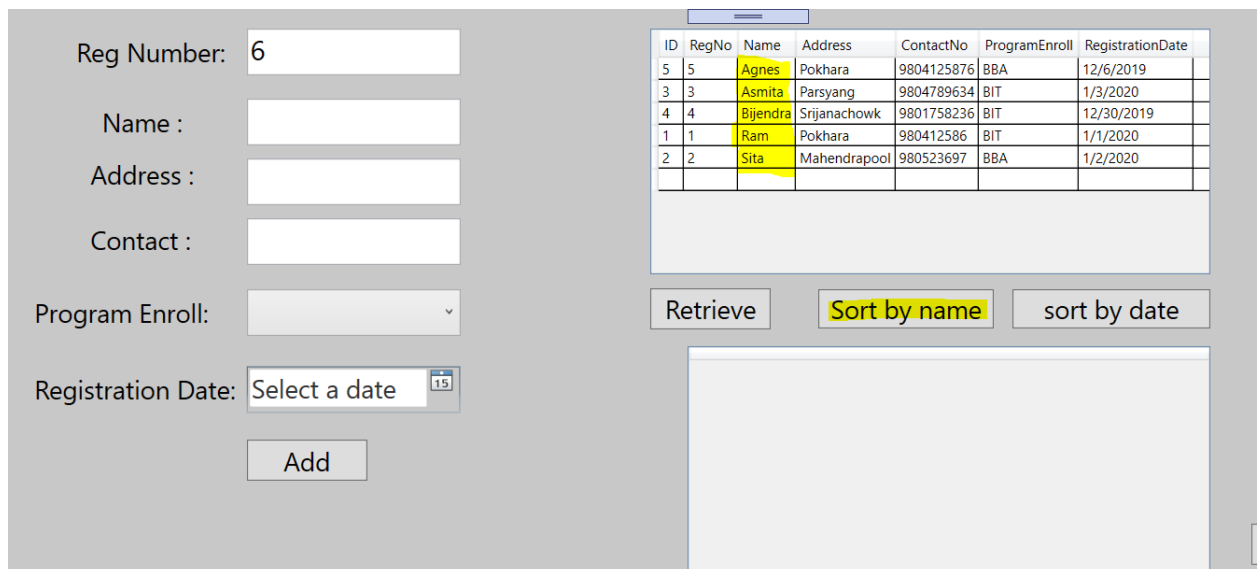
Add

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
1	1	Ram	Pokhara	980412586	BIT	1/1/2020
2	2	Sita	Mahendrapool	980523697	BBA	1/2/2020
3	3	Asmita	Parsyang	9804789634	BIT	1/3/2020
4	4	Bijendra	Srijanachowk	9801758236	BIT	12/30/2019
5	5	Agnes	Pokhara	9804125876	BBA	12/6/2019

Retrieve Sort by name sort by date

Figure 6: Retrieve

6. After clicking on the “sort by name” button it will help to display the name section of the table in an ordered way.



Reg Number: 6

Name :

Address :

Contact :

Program Enroll:

Registration Date: Select a date 15

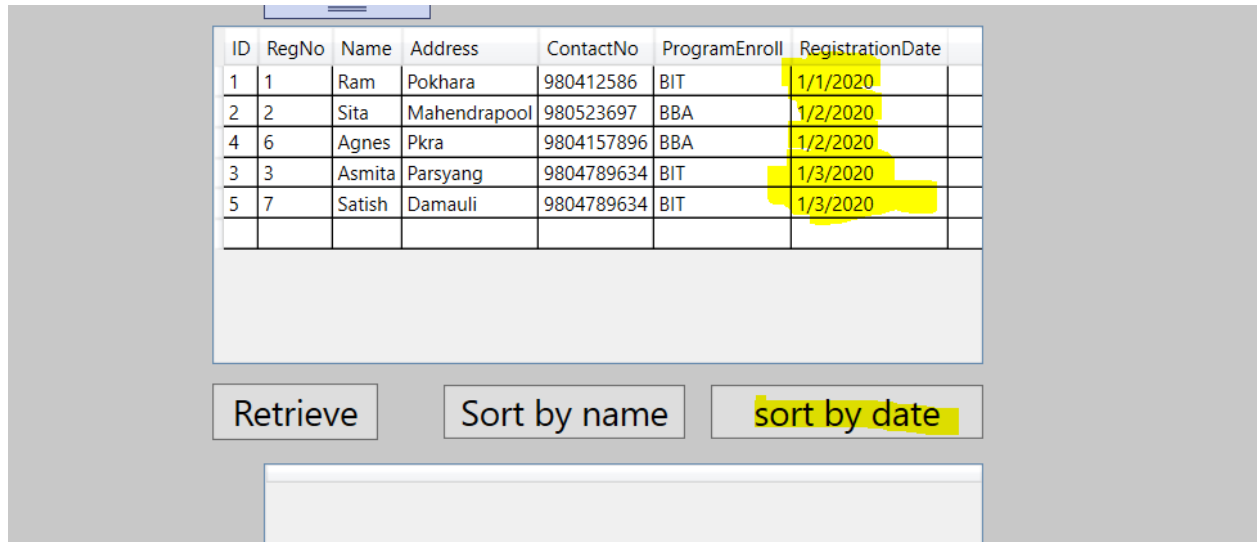
Add

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
5	5	Agnes	Pokhara	9804125876	BBA	12/6/2019
3	3	Asmita	Parsyang	9804789634	BIT	1/3/2020
4	4	Bijendra	Srijanachowk	9801758236	BIT	12/30/2019
1	1	Ram	Pokhara	980412586	BIT	1/1/2020
2	2	Sita	Mahendrapool	980523697	BBA	1/2/2020

Retrieve Sort by name sort by date

Figure 7: Sort by name

7. "Sort by date" function helps to show the students data according to date registered orderly.



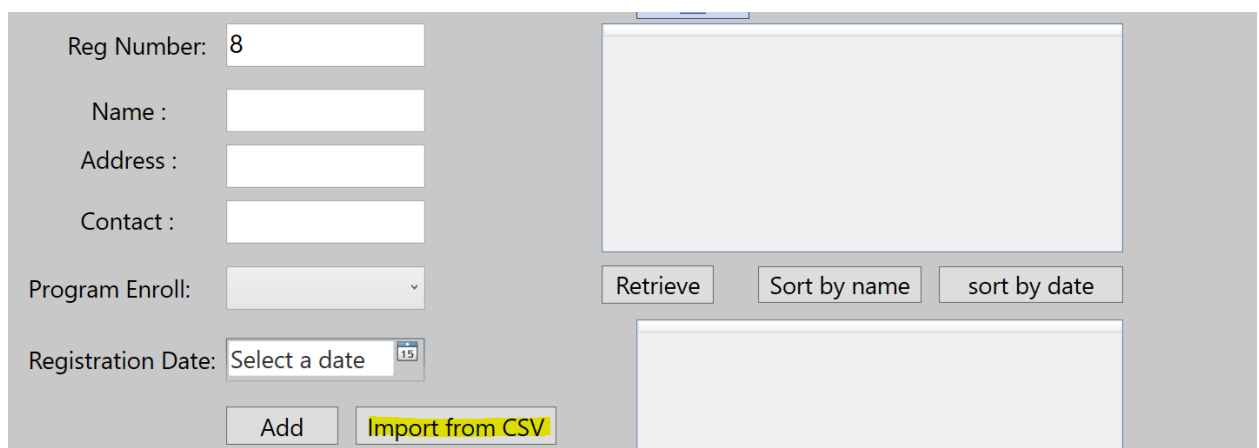
The screenshot shows a web application interface. At the top, there is a table with 7 columns: ID, RegNo, Name, Address, ContactNo, ProgramEnroll, and RegistrationDate. The table contains 5 rows of student data. The 'RegistrationDate' column is highlighted in yellow. Below the table, there are three buttons: 'Retrieve', 'Sort by name', and 'sort by date'. The 'sort by date' button is highlighted in yellow.

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
1	1	Ram	Pokhara	980412586	BIT	1/1/2020
2	2	Sita	Mahendrapool	980523697	BBA	1/2/2020
4	6	Agnes	Pkra	9804157896	BBA	1/2/2020
3	3	Asmita	Parsyang	9804789634	BIT	1/3/2020
5	7	Satish	Damauli	9804789634	BIT	1/3/2020

Retrieve Sort by name sort by date

Figure 8: Sort by date

8." Import from CSV" button helps to import the saved data from CSV file and display in the grid table.



The screenshot shows a web application interface for adding a new student. On the left, there are input fields for 'Reg Number' (value: 8), 'Name', 'Address', 'Contact', 'Program Enroll' (dropdown), and 'Registration Date' (calendar picker). Below these fields are 'Add' and 'Import from CSV' buttons. The 'Import from CSV' button is highlighted in yellow. On the right, there is a large empty grid table. Below the grid, there are three buttons: 'Retrieve', 'Sort by name', and 'sort by date'.

Reg Number: 8

Name :

Address :

Contact :

Program Enroll:

Registration Date: Select a date 15

Add Import from CSV

Retrieve Sort by name sort by date

Figure 9: Import from CSV 1

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
3	3	Asmita	Parsyang	9804789634	BIT	1/3/2020
4	6	Agnes	Pkra	9804157896	BBA	1/2/2020
5	7	Satish	Damauli	9804789634	BIT	1/3/2020
1	1	Ram	Pokhara	980412586	BIT	1/1/2020
2	2	Sita	Mahendrapool	980523697	BBA	1/2/2020
3	3	Asmita	Parsyang	9804789634	BIT	1/3/2020
4	6	Agnes	Pkra	9804157896	BBA	1/2/2020
5	7	Satish	Damauli	9804789634	BIT	1/3/2020

Figure 10: Import from CSV 2

9. “Enrolled students” button helps to display the total number of students enrolled so far weekly in each program offered by the institution.

ProgramEnroll	Total Students
BBA	4
BIT	6

Figure 11: Enrolled Students

10." Show Total Student Chart" helps to show the total number of students enrolled so far weekly in a chart.

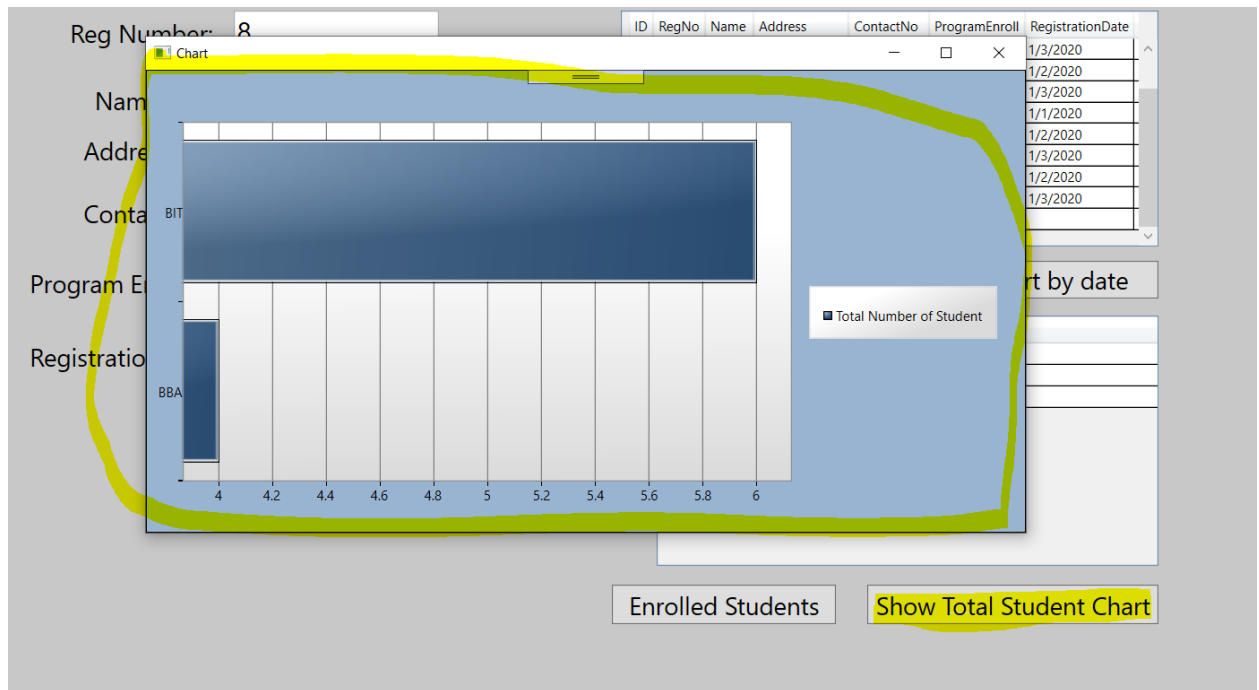


Figure 12: Total Students Chart

2.2 Architecture Diagram

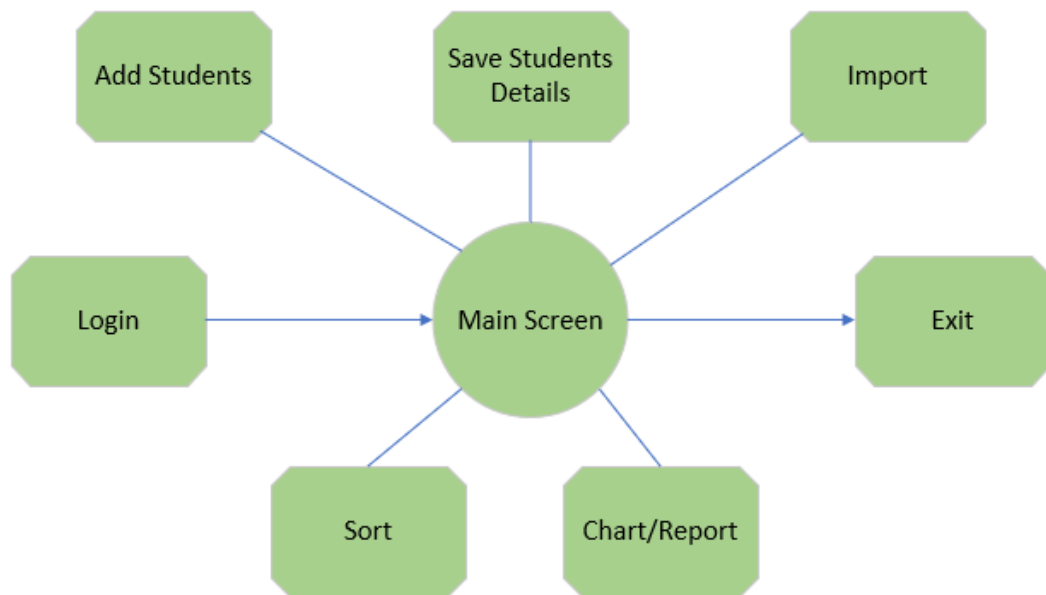


Figure 13: Architecture Diagram of the system

2.3 Functionality

This system can only be used by the head or staffs of the school. This is a desktop application, so a laptop or computer with minimum 2GB RAM, core i3 is required to run the system. The system records the details of the students and that data is saved safely into the system and can be exported to a laptop by exporting and can also import the data.

2.4 Class Diagram

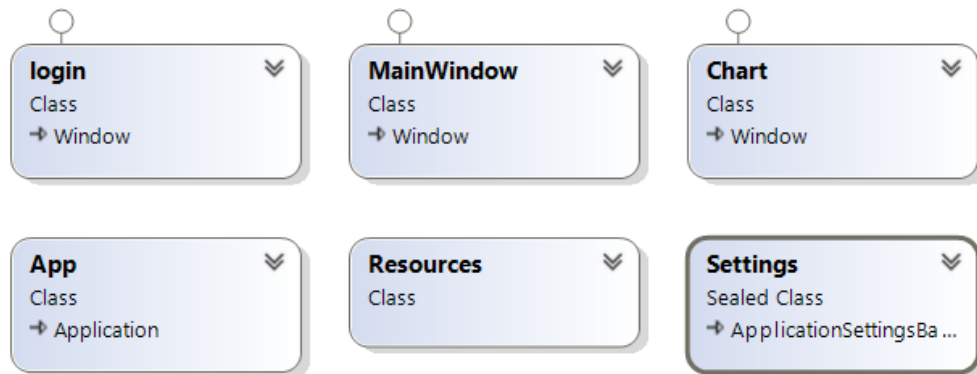


Figure 14: Class Diagram of Student Information System

1. Login

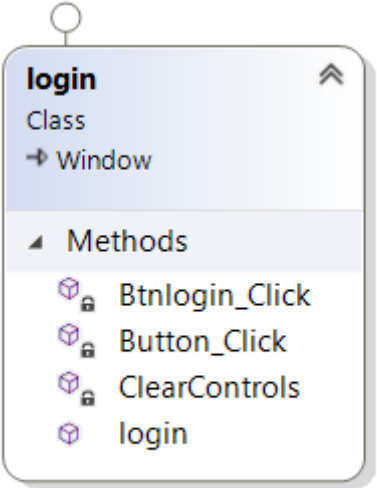
Methods	Description	 <pre> classDiagram class login { <<class>> +login() +Button_Click() +ClearControls() +Btnlogin_Click() } login -- > Window </pre>
Btnlogin_Click	The login button displays the main page of the system.	
Button_Click	This exit button closes the system.	

Table 1: Class Diagram of Login

2.MainWindow

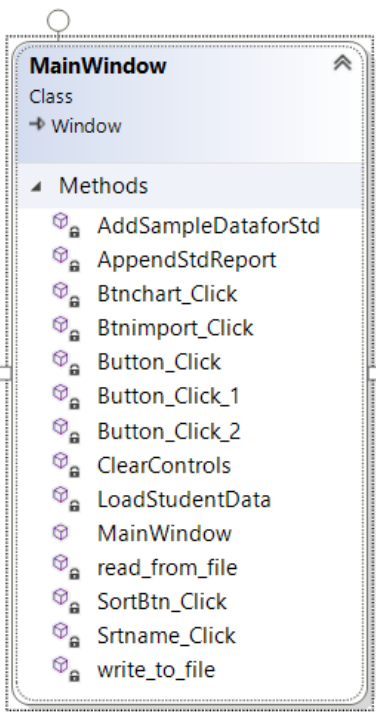
Methods	Description	 <pre> classDiagram class MainWindow { +AddSampleDataforStd +AppendStdReport +Btnchart_Click +Btnimport_Click +Button_Click +Button_Click_1 +Button_Click_2 +ClearControls +LoadStudentData +MainWindow +read_from_file +SortBtn_Click +Srtname_Click +write_to_file } </pre>
Btnchart_Click	It displays the weekly chart of the system.	
Btnimport_Click	It helps to import the csv data and display the data in table.	
Button_Click	It displays the weekly total number of students enrolled in each program.	
Button_Click_1	It helps to add the data in the table as well in the xml file.	
Button_Click_2	It helps to retrieve the saved student data.	
SortBtn_Click	It helps to show the student data sorted by date.	
Sortname_Click	It helps to show the student data sorted by name.	

Table 2: Class Diagram of MainWindow

3. Chart

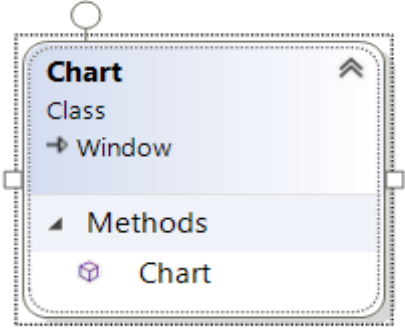
Methods	Description	
Chart	This method helps to show the total data of students enrolled weekly in particular program in a chart diagram.	 <pre> classDiagram class Chart { +Window +Methods } </pre> <p>The diagram shows a class named 'Chart' with a 'Window' attribute and a 'Methods' section containing a 'Chart' method.</p>

Table 3: Class Diagram of Chart

4. App

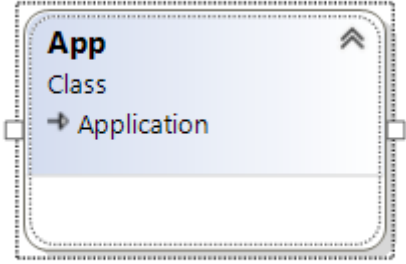
Methods	Description	
App	This method helps to use the system.	 <pre> classDiagram class App { +Application } </pre> <p>The diagram shows a class named 'App' with an 'Application' attribute.</p>

Table 4: Class Diagram of App

5. Resources

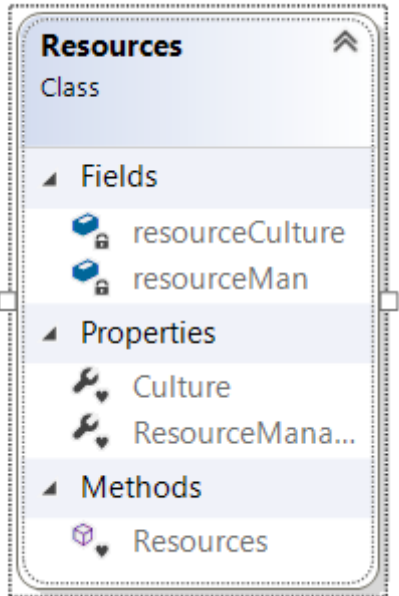
Methods	Description	
Resources	This method helps to use the available resources needed for the app development purpose.	

Table 5: Class Diagram of Resources

6. Settings

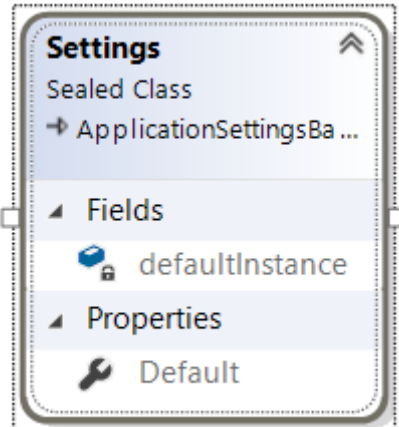
Methods	Description	
Settings	This method helps to solve problems when needed.	

Table 6: Class Diagram of Settings

2.5 Flowchart Diagram

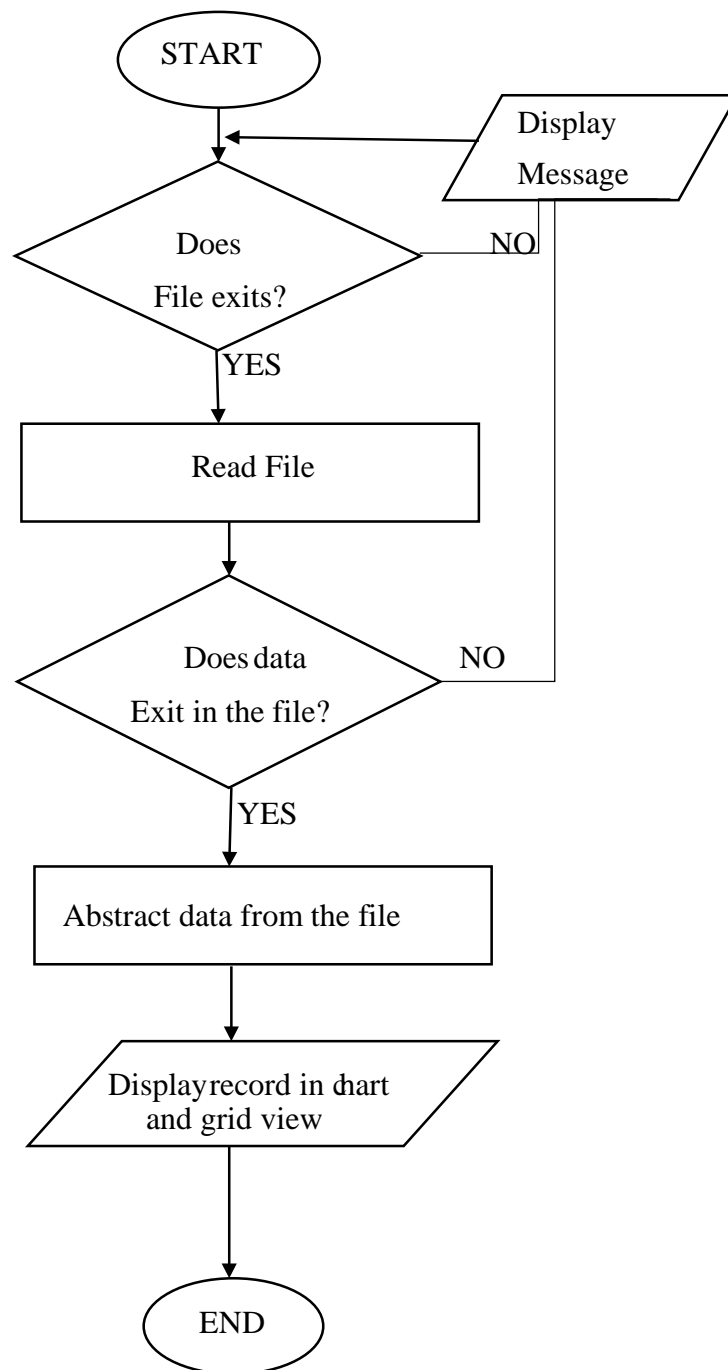


Figure 15: Flowchart

2.6 Algorithms of Reports

Steps:

1. Start
2. Check whether the file exists or not.
3. If it doesn't exist, display error message and restart
4. If exists, read the available data
5. If data found, retrieve the data
6. Display the data in the chart and grid view
7. Stop

3.Sorting Algorithm

The sorting Algorithm used in the Student Information System is bubble sorting algorithm.

Bubble Sort is the simplest sorting algorithm that works by repeatedly swapping the adjacent elements if they are in wrong order.

Example:**First Pass:**

(**5** 1 4 2 8) → (**1** **5** 4 2 8), Here, algorithm compares the first two elements, and swaps since $5 > 1$.

(1 **5** 4 2 8) → (1 **4** **5** 2 8), Swap since $5 > 4$

(1 4 **5** 2 8) → (1 4 **2** **5** 8), Swap since $5 > 2$

(1 4 2 **5** 8) → (1 4 2 **5** 8), Now, since these elements are already in order ($8 > 5$), algorithm does not swap them.

Second Pass:

(**1** 4 2 5 8) → (**1** 4 2 5 8)

(1 **4** 2 5 8) → (1 **2** **4** 5 8), Swap since $4 > 2$

(1 2 **4** 5 8) → (1 2 **4** 5 8)

(1 2 4 **5** 8) → (1 2 4 **5** 8)

Now, the array is already sorted, but our algorithm does not know if it is completed. The algorithm needs one **whole** pass without **any** swap to know it is sorted.

Third Pass:

(**1** 2 4 5 8) → (**1** 2 4 5 8)

(1 **2** 4 5 8) → (1 **2** 4 5 8)

(1 2 **4** 5 8) → (1 2 **4** 5 8)

(1 2 4 **5** 8) → (1 2 4 **5** 8)

4. Reflection

As using the Visual Studio 2019 it has been a new experience, the development of “Student Information” system. The system is made to use in the real working for the schools.

It has been a great experience while working with the new technology. In the system serialization, deserialization was also new thing for me and import, export data in .csv and .xml file in this system. I find some difficulties in showing data of weekly chart, connecting user interface with coding, has more complications on codes and error handling.

This project has made me familiar with visual studio and dealing with codes. It took longer while spending the time with this project. There can be more features to make perfect the system and in future it can much better application.

5. Conclusion

The name of my project is “Student Information” system. This system is designed and developed in the Visual Studio 2019. In past days all the works are done in the paper which causes difficulties, problems and waste of time. To reduce these problems the new system is made for recovery of the problems.

It took a long time to build up the task in Visual Studio Enterprise 2019 utilizing C# programming dialect. The framework has login screen to add security to the task. After login, the framework shows a primary screen where every one of the functionalities are found. Aside from various shape components, class outline for every one of the structures and classes were utilized.

In this project I find some difficulties in showing data of weekly chart, connecting user interface with coding, has more complications on codes and error handling. This project also has made me familiar with visual studio and dealing with codes. With the help of my friends and teacher I have done the project and solved the problems seen on the system. I have also taken the help from the Google and online website study. It has been a great experience while working in this project.

6. References

<https://www.geeksforgeeks.org/bubble-sort/>

7. Appendix

1. Login

```
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.Shapes;
using AppDevCoursewrk;

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

        private void Btnlogin_Click(object sender, RoutedEventArgs e)
        {
            string user = txtuser.Text;
            string pass = txtpass.Password;

            if (user == "kapil" && pass == "admin")
            {
                //this.Hide();
                MessageBox.Show("Login Successful!");
                MainWindow mainWindow = new MainWindow();
                mainWindow.Show();
            }
        }
    }
}
```

```
    } else if(user!="kapil" && pass != "admin"){  
        MessageBox.Show("Wrong username or password");  
        ClearControls();  
    }  
}  
  
private void ClearControls() {  
    txtuser.Text = "";  
    txtpass.Password="";  
}  
  
private void Button_Click(object sender, RoutedEventArgs e)  
{  
    this.Close();  
}  
}  
}
```

2.MainWindow

```
using System;  
using System.Data;  
using System.IO;  
using System.Windows;  
using DataHandler;  
using Microsoft.Win32;  
  
namespace AppDevCoursewrk  
{  
    /// <summary>  
    /// Interaction logic for MainWindow.xaml  
    /// </summary>  
    public partial class MainWindow : Window  
    {  
  
        public MainWindow()  
        {  
            InitializeComponent();  
  
            txtRegNo.Text = read_from_file();  
  
            //LoadStudentData();  
        }  
    }  
}
```

```
}
```

```
private void AddSampleDataforStd(DataSet dataSet)
{

    var dr = dataSet.Tables["Course"].NewRow();
    dr["Name"] = "BBA";
    dr["DisplayText"] = "BBA Hons";
    dataSet.Tables["Course"].Rows.Add(dr);

    var dr1 = dataSet.Tables["Student"].NewRow();
    dr1["Name"] = txtName.Text;
    dr1["Address"] = txtAddress.Text;
    dr1["ContactNo"] = txtContact.Text;
    dr1["ProgramEnroll"] = combo.Text;
    dr1["RegistrationDate"] = DateTime.Today.AddDays(-2);
    dataSet.Tables["Student"].Rows.Add(dr1);

}
```

```
private void AppendStdReport(DataSet dataSet)
{
    if (File.Exists(@"F:\XML storage\StudentReport.xml"))
    {
        var handler = new Handler();

        dataSet.Tables["StudentReport"].ReadXml(@"F:\XML storage\StudentReport.xml");

        var dr2 = dataSet.Tables["StudentReport"].NewRow();
        dr2["RegNo"] = txtRegNo.Text;
        dr2["Name"] = txtName.Text;
        dr2["Address"] = txtAddress.Text;
        dr2["ContactNo"] = txtContact.Text;
        dr2["ProgramEnroll"] = combo.Text;
        dr2["RegistrationDate"] = txtdate.Text;
        dataSet.Tables["StudentReport"].Rows.Add(dr2);
    }
}
```



```
        dataSet.Tables["StudentReport"].WriteXml(@"F:\XML storage\StudentReport.xml");
    }
    else
    {
        dataSet.Tables["StudentReport"].WriteXml(@"F:\XML storage\StudentReport.xml");
        AppendStdReport(dataSet);
    }
}

private void Button_Click_1(object sender, RoutedEventArgs e)
{
    var handler = new Handler();
    var dataSet = handler.CreateDataSet();
    AddSampleDataforStd(dataSet);
    AppendStdReport(dataSet);

    var regno = txtRegNo.Text;
    var name = txtName.Text;
    dataSet.Tables["Student"].WriteXml(@"F:\XML storage\" + name + "CWData" + regno +
    ".xml");

    dataSet.Tables[2].WriteXml(@"F:\XML storage\StudentReport.xml");

    write_to_file(txtRegNo.Text);
    MessageBox.Show("Student details added");

    txtRegNo.Text = read_from_file();

    ClearControls();
    LoadStudentData();
}

private void write_to_file(string text)
{
    File.WriteAllText(@"F:\XML storage\count.txt", text);

}

private string read_from_file()
{

```

```
int i = 1;
if (File.Exists(@"F:\XML storage\count.txt"))
{
    string text = File.ReadAllText(@"F:\XML storage\count.txt");
    i = int.Parse(text.ToString());
    i = i + 1;
}
else
{
    File.WriteAllText(@"F:\XML storage\count.txt", "text");
}
return i.ToString();
}

private void ClearControls()
{
    txtName.Text = "";
    txtAddress.Text = "";
    txtContact.Text = "";
}

private void LoadStudentData()
{
    if (System.IO.File.Exists(@"F:\XML storage\StudentReport.xml"))
    {
        var handler = new Handler();

        var dataSet = new DataSet();

        dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");

        DataTable dtStdReport = new DataTable();
        dtStdReport = dataSet.Tables[0];
        grdStd.DataContext = dtStdReport.DefaultView;
    }
}
```

```
private void Button_Click(object sender, RoutedEventArgs e)
{
    var dataSet = new DataSet();
    dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");
    DataTable dtStdReport = dataSet.Tables[0];

    int total_BIT = 0;
    int total_BBA = 0;

    DataTable dt = new DataTable("newTable");
    dt.Columns.Add("ProgramEnroll", typeof(string));
    dt.Columns.Add("Total Students", typeof(int));

    for (int i = 0; i < dtStdReport.Rows.Count; i++)
    {
        string col = dtStdReport.Rows[i]["ProgramEnroll"].ToString();
        if (col == "BIT")
        {
            total_BIT++;
        }
        else if (col == "BBA")
        {
            total_BBA++;
        }
    }
    dt.Rows.Add("BBA", total_BBA);
    dt.Rows.Add("BIT", total_BIT);
    grdreport.DataContext = dt.DefaultView;
}

private void Srtnam_Click(object sender, RoutedEventArgs e)
{
    var dataSet = new DataSet();
    dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");
    DataTable DataTable = dataSet.Tables["StudentReport"];
    DataTable.DefaultView.Sort = "Name Asc";
}
```

```
grdStd.DataContext = DataTable.DefaultView;
}

private void SortBtn_Click(object sender, RoutedEventArgs e)
{
    var dataSet = new DataSet();
    dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");
    DataTable DataTable = dataSet.Tables["StudentReport"];
    DataTable.DefaultView.Sort = "RegistrationDate Asc";
    grdStd.DataContext = DataTable.DefaultView;
}

private void Button_Click_2(object sender, RoutedEventArgs e)
{
    if (System.IO.File.Exists(@"F:\XML storage\StudentReport.xml"))
    {
        var handler = new Handler();

        var dataSet = new DataSet();

        dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");

        DataTable dtStdReport = new DataTable();
        dtStdReport = dataSet.Tables[0];
        grdStd.DataContext = dtStdReport.DefaultView;
    }
    else
        MessageBox.Show("Data Retrieval Unsuccessful", "Alert");
}

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

private void Btnimport_Click(object sender, RoutedEventArgs e)
{
    {
        var dataSet = new DataSet();
        dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");
    }
}
```

```

OpenFileDialog openFileDialog = new OpenFileDialog();
openFileDialog.Filter = "CSV Files*.csv";
openFileDialog.DefaultExt = ".csv";

bool? fileselect = openFileDialog.ShowDialog();
if (fileselect == 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["StudentReport"].NewRow();
            newRow["ID"] = values[0];
            newRow["RegNo"] = values[1];
            newRow["Name"] = values[2];
            newRow["Address"] = values[3];
            newRow["ContactNo"] = values[4];
            newRow["ProgramEnroll"] = values[5];
            newRow["RegistrationDate"] = values[6];
            dataSet.Tables["StudentReport"].Rows.Add(newRow);
        }

        dataSet.WriteXml(@"F:\XML storage\StudentReport.xml");
        grdStd.ItemsSource = dataSet.Tables["StudentReport"].DefaultView;
    }
}
}
}
}
}
}

```

3.Handler

```
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());

            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("ProgramEnroll", typeof(string));
            dt.Columns.Add("RegistrationDate", typeof(string));
            //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("ProgramEnroll", typeof(string));
    dt.Columns.Add("RegistrationDate", typeof(string));

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

```
}  
}
```

4.Chart

```
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.Controls.DataVisualization.Charting;  
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;  
  
namespace AppDevCoursewrk  
{  
    /// <summary>  
    /// Interaction logic for Chart.xaml  
    /// </summary>  
    public partial class Chart : Window  
    {  
        public Chart()  
        {  
            InitializeComponent();  
            var dataSet = new DataSet();  
            dataSet.ReadXml(@"F:\XML storage\StudentReport.xml");  
            DataTable dtStdReport = dataSet.Tables[0];  
  
            int total_BIT = 0;  
            int total_BBA = 0;  
  
            DataTable dt = new DataTable("newTable");  
            dt.Columns.Add("ProgramEnroll", typeof(string));  
            dt.Columns.Add("Total Students", typeof(int));
```



```
for (int i = 0; i < dtStdReport.Rows.Count; i++)
{
    string col = dtStdReport.Rows[i]["ProgramEnroll"].ToString();
    if (col == "BIT")
    {
        total_BIT++;
    }
    else if (col == "BBA")
    {
        total_BBA++;
    }
}

dt.Rows.Add("BBA", total_BBA);
dt.Rows.Add("BIT", total_BIT);
//grdreport.DataContext = dt.DefaultView;

((BarSeries)totalStdChart.Series[0]).ItemsSource = new KeyValuePair<string, int>[] {
    new KeyValuePair<string, int>("BBA", total_BBA),
    new KeyValuePair<string, int>("BIT", total_BIT)
};
}
}
}
```