

Informatics College Pokhara



informatics
college pokhara

Application Development

CS6004NI

Course Work 1

Submitted By: Suman Bahadur Shrestha
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	missing controls in the interface
Manual data entry or import from csv	appropriate use of data types but missing some properties required or missing CRUD operation
Data Validation	Only basic 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	Very poorly written

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+	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



Application Development
CS6004NP
Coursework 1

Submitted By:

Student's name: Suman Bahadur Shrestha
London Met ID: 17030531
Group: L3C1
Date: 10-Jan-2020

Submitted To:

Mr. Ishwor Sapkota

Abstract

This is the first coursework of this project. It is an individual task of this coursework. This project is all about Student Information System and it is a desktop application. C# programming languages is use to developed this application. This application keep track of the student's details, program enroll and registration date. It also import or export the record of the student's details.

Contents

1.	Introduction	1
1.1.	Current Scenario.....	1
1.2.	Proposed System.....	1
2.	User Manual	2
2.1.	Opening a System.....	2
3.	Journal Articles	14
4.	System Architecture	15
5.	Class Diagram	16
5.1.	Individual Diagram	16
6.	Flowchart	19
7.	Algorithm	20
7.1.	Bubble Sorting	20
8.	Reflection	21
9.	Conclusion.....	21
10.	References	22
11.	Appendix	23

List of Figure

Figure 1: Login Page	2
Figure 2: Pass empty value	3
Figure 3: Not match username and password	4
Figure 4: Login success.....	5
Figure 5: Home Page	6
Figure 6: Student enroll page	7
Figure 7: Generate XML file	7
Figure 8: Show data in data grid.....	8
Figure 9: Student enroll message.....	8
Figure 10: Retrieved data	9
Figure 11: Import csv file	10
Figure 12: Sort by name	11
Figure 13: Sort by date	11
Figure 14: XML file	12
Figure 15: Schema file.....	13
Figure 16: CSV file	13
Figure 17: Pie Chart and table.....	13
Figure 18: System Architecture of Student Information System	15
Figure 19: Class Diagram generated by visual studio	16
Figure 20: Flow Chart in Student Enroll.....	19

List of Table

Table 1: Main Window	16
Table 2: Main Page	17
Table 3: Student Enroll	18
Table 4: Weekly Report	18
Table 5: Data Handler	18

1. Introduction

This is the first coursework of this module. This project is fully base on the C# programming language. This project is about Student Information System and it is desktop application. The features and functions that are required for this application are almost fulfill. It consist the features like adding new student with course enroll and date. Details information are show in data grid and at the same time the added data are save in XML file. There are some other features like importing data from csv file only and save in XML file, a weekly table and chart are shown. Other description are mention below.

1.1. Current Scenario

Almost all the student information system use manual system in which they keep record of the student details in some file (csv,XML,etc.). It is difficult to them to import new data in current data file. Therefore, people are searching for such application, which can new data in a file and can add outer data in current data file. A weekly report is which show a chart of total student.

1.2. Proposed System

This desktop application is made to add new student data and record student details. This application is mainly made to store student data in a file and able to add import data in current file and a weekly chart of students.

2. User Manual

There are screenshot below which will illustrate a user how to operate the system

2.1. Opening a System

As the application open, you will see a login page where you have to enter username and password.



Figure 1: Login Page



Figure 2: Pass empty value

If user press login button without putting value in username and password. A message box is shown.



Figure 3: Not match username and password

If user pass wrong value in username and password, a message box is shown.



Figure 4: Login success

If user enter "admin" in username and "admin" in password. It show a login success message.



If user press exit button it show a message and if user click yes then a program will exit and if user press no then it will stay in program.

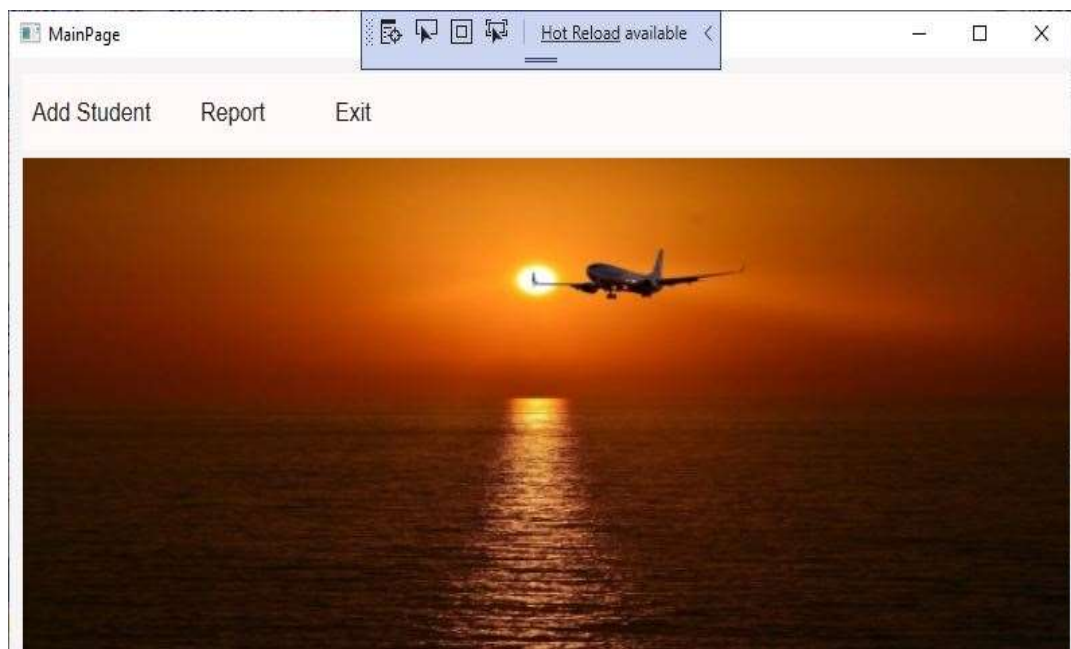
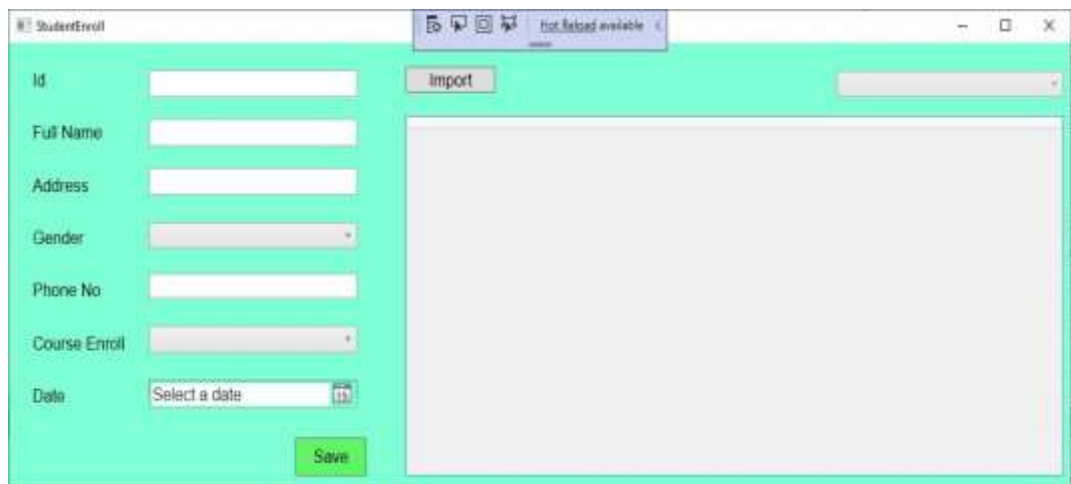


Figure 5: Home Page

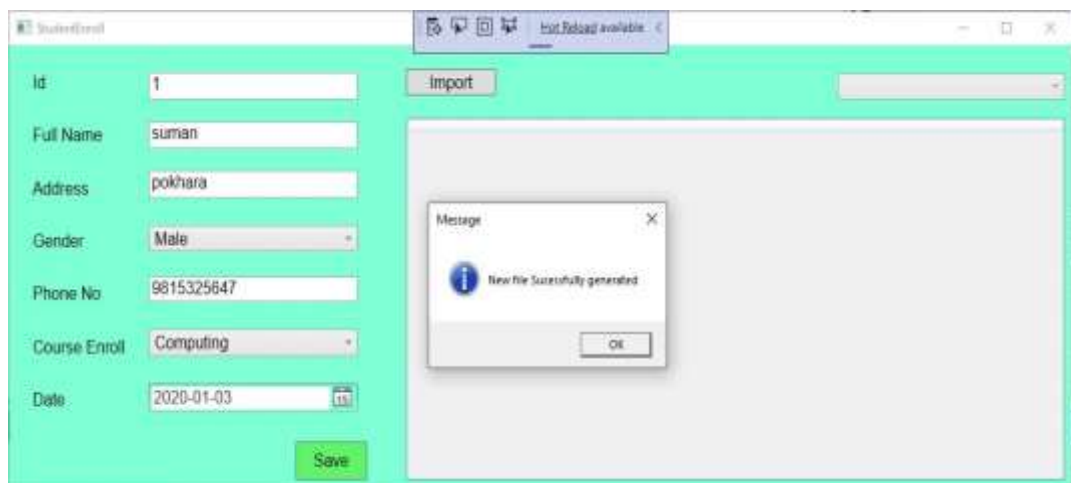
After login success home page will appear.



The screenshot shows a web application window titled "StudentEnroll". On the left, there is a form with the following fields: "Id" (text input), "Full Name" (text input), "Address" (text input), "Gender" (dropdown menu), "Phone No" (text input), "Course Enroll" (dropdown menu), and "Date" (calendar picker). Below these fields is a green "Save" button. On the right, there is an "Import" button and a large empty rectangular area. A status bar at the top right indicates "Not Released available".

Figure 6: Student enroll page

If user click add student then it will open a new student enroll page.



This screenshot shows the same "StudentEnroll" application window, but with data entered in the form: "Id" is 1, "Full Name" is suman, "Address" is pokhara, "Gender" is Male, "Phone No" is 9815325647, "Course Enroll" is Computing, and "Date" is 2020-01-03. A message dialog box is open in the center, displaying an information icon and the text "New file Successfully generated", with an "OK" button at the bottom. The "Save" button remains visible at the bottom of the form.

Figure 7: Generate XML file

Add new data, it will create a new XML file in first time.

The screenshot shows the StudentEnroll application window. On the left is a form with fields for Id, Full Name, Address, Gender, Phone No, Course Enroll, and Date. A green 'Save' button is at the bottom. On the right is a data grid with columns: id, name, address, number, gender, courseenroll, and date. It contains one row of data.

id	name	address	number	gender	courseenroll	date
1	suman	pokhara	9815325647	Male	Computing	2020-01-03 12:00:00 AM

Figure 8: Show data in data grid

Added data are shown in data grid.

This screenshot shows the StudentEnroll application after a new student has been added. The form fields are filled with: Id: 3, Full Name: rjan, Address: tanahun, Gender: Male, Phone No: 9815253655, Course Enroll: Networks and IT Security, and Date: 2020-01-08. The data grid now has two rows. A 'Message' dialog box is displayed in the center, stating 'Student Enrolled Successfully' with an 'OK' button.

id	name	address	number	gender	courseenroll	date
1	suman	pokhara	9815325647	Male	Computing	2020-01-03 12:00:00 AM
2	suraj	kathmandu	9821563247	Male	Multimedia Technologies	2020-01-04 12:00:00 AM

Figure 9: Student enroll message

At next time it show a message of student enroll successful.

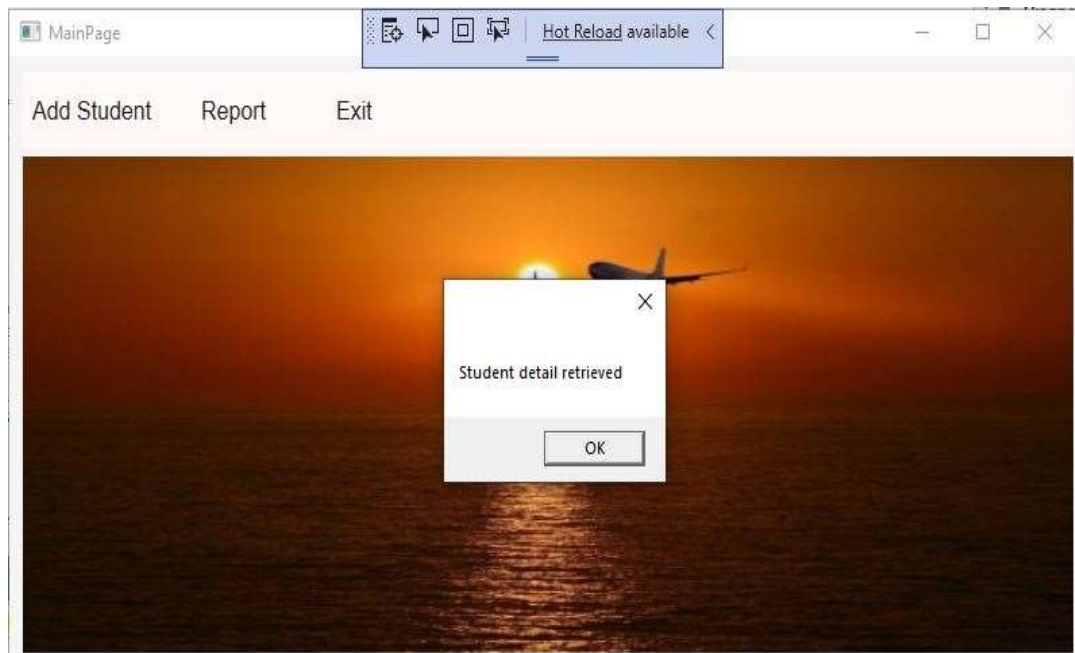


Figure 10: Retrieved data

Next time if user open student enroll it retrieved the student detail automatically.

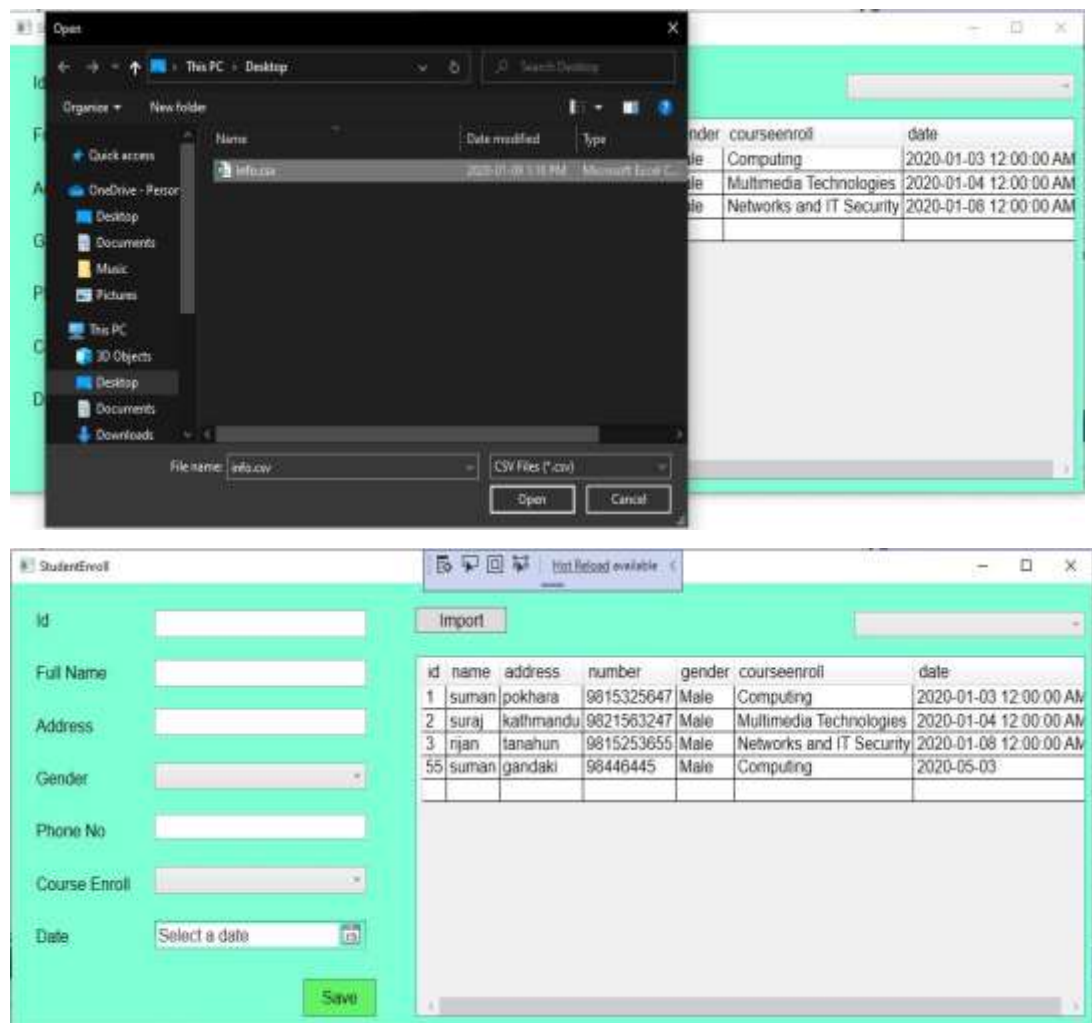
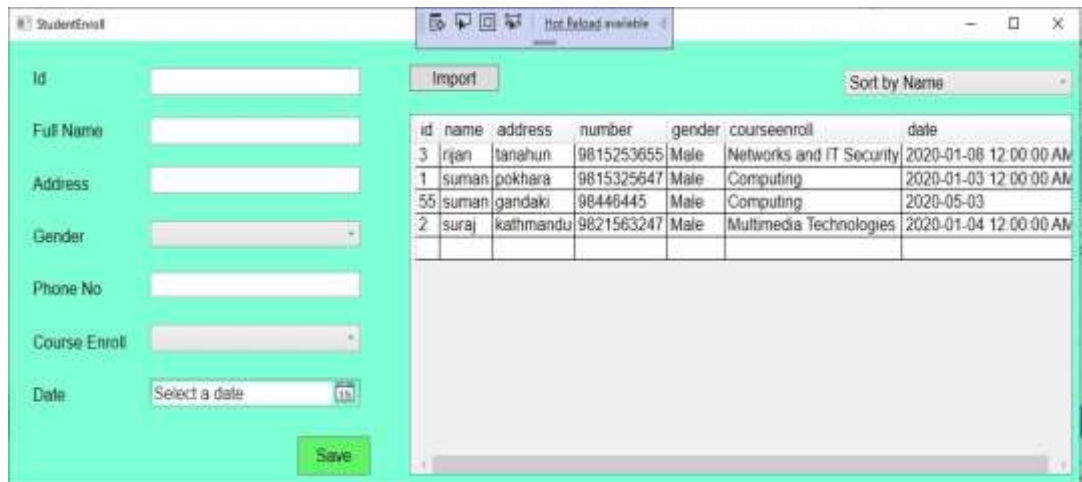


Figure 11: Import csv file

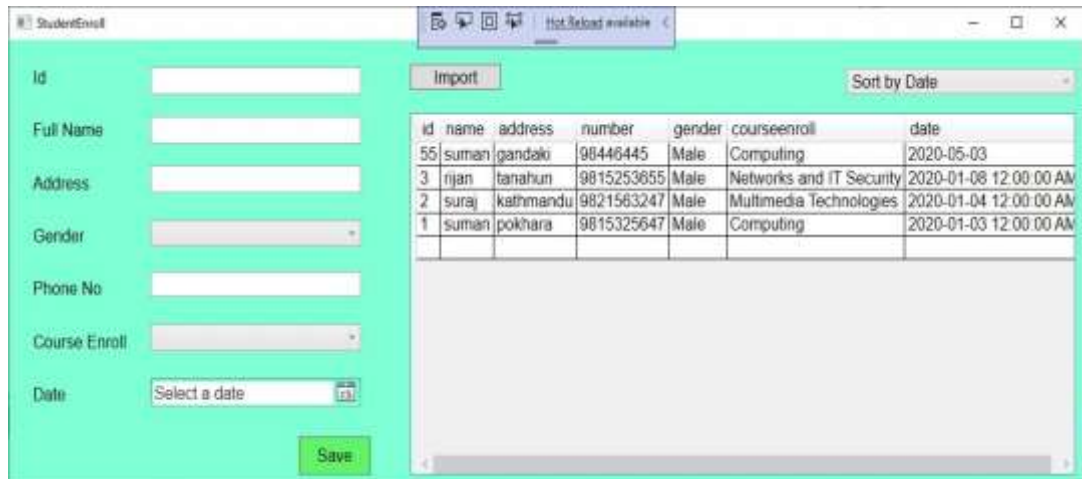
If user click import button, it open a file and user have to select one csv file. It will show in data grid and it automatically save in XML file.



id	name	address	number	gender	courseenroll	date
3	niran	tanahun	9815253655	Male	Networks and IT Security	2020-01-08 12:00:00 AM
1	suman	pokhara	9815325647	Male	Computing	2020-01-03 12:00:00 AM
55	suman	gandaki	98446445	Male	Computing	2020-05-03
2	suraj	kathmandu	9821563247	Male	Multimedia Technologies	2020-01-04 12:00:00 AM

Figure 12: Sort by name

In combo box, if user select sort by name then sort the data grid data by name in ascending order.



id	name	address	number	gender	courseenroll	date
55	suman	gandaki	98446445	Male	Computing	2020-05-03
3	niran	tanahun	9815253655	Male	Networks and IT Security	2020-01-08 12:00:00 AM
2	suraj	kathmandu	9821563247	Male	Multimedia Technologies	2020-01-04 12:00:00 AM
1	suman	pokhara	9815325647	Male	Computing	2020-01-03 12:00:00 AM

Figure 13: Sort by date

In combo box, if user select sort by date then sort the data grid data by date in descending order.



```

StudentCWDData.xml - Notepad
File Edit Format View Help
<?xml version="1.0" standalone="yes"?>
<NewDataSet>
  <Student>
    <id>1</id>
    <name>suman</name>
    <address>pokhara</address>
    <number>9815325647</number>
    <gender>Male</gender>
    <courseenroll>Computing</courseenroll>
    <date>2020-01-03 12:00:00 AM</date>
  </Student>
  <Student>
    <id>2</id>
    <name>suraj</name>
    <address>kathmandu</address>
    <number>9821563247</number>
    <gender>Male</gender>
    <courseenroll>Multimedia Technologies</courseenroll>
    <date>2020-01-04 12:00:00 AM</date>
  </Student>
  <Student>
    <id>3</id>
    <name>rijan</name>
    <address>tanahun</address>
    <number>9815253655</number>
    <gender>Male</gender>
    <courseenroll>Networks and IT Security</courseenroll>
    <date>2020-01-08 12:00:00 AM</date>
  </Student>
  <Student>
    <id>55</id>
    <name>suman</name>
    <address>gandaki</address>
    <number>98446445</number>
    <gender>Male</gender>
    <courseenroll>Computing</courseenroll>
    <date>2020-05-03</date>
  </Student>
</NewDataSet>

```

Figure 14: XML file

List of added data with import data.

```

StudentCsSchema.schema - Notepad
File Edit Format View Help
<?xml version="1.0" standalone="yes"?>
<xs:schema id="NewDataSet" xmlns="" xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns:msdata="urn:schemas-microsoft-com:xml-msdata">
  <xs:element name="NewDataSet" msdata:IsDataSet="true" msdata:UseCurrentLocale="true">
    <xs:complexType>
      <xs:choice minOccurs="0" maxOccurs="unbounded">
        <xs:element name="Student">
          <xs:complexType>
            <xs:sequence>
              <xs:element name="id" type="xs:string" minOccurs="0" />
              <xs:element name="name" type="xs:string" minOccurs="0" />
              <xs:element name="address" type="xs:string" minOccurs="0" />
              <xs:element name="number" type="xs:string" minOccurs="0" />
              <xs:element name="gender" type="xs:string" minOccurs="0" />
              <xs:element name="courseenroll" type="xs:string" minOccurs="0" />
              <xs:element name="date" type="xs:string" minOccurs="0" />
            </xs:sequence>
          </xs:complexType>
        </xs:element>
      </xs:choice>
    </xs:complexType>
  </xs:element>
</xs:schema>

```

Figure 15: Schema file

Schema file is generated.

```

info.csv - Notepad
File Edit Format View Help
id,name,address,number,gender,courseenroll,date
55,suman,gandaki,98446445,Male,Computing,2020-05-03

```

Figure 16: CSV file

Data in csv file.

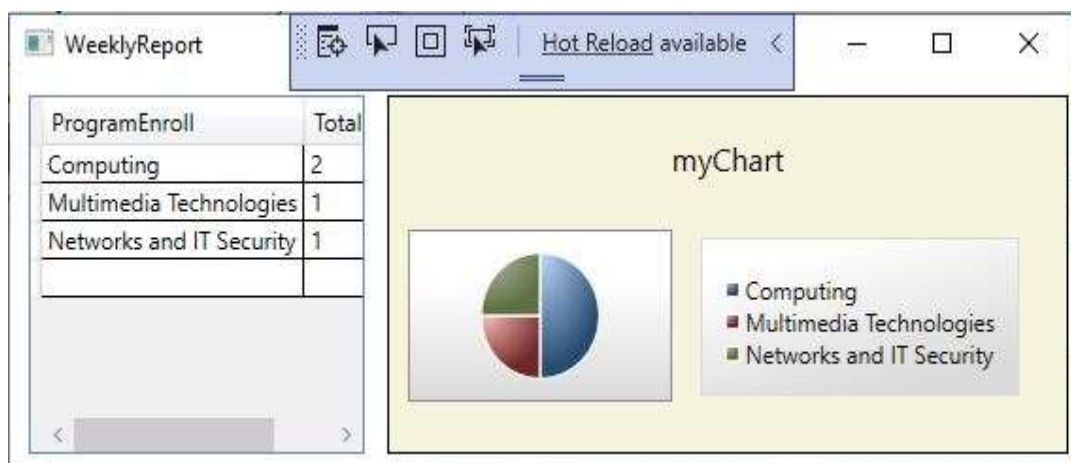


Figure 17: Pie Chart and table

Total number of student are according to program enroll and pie chart according to total student.

3. Journal Articles

Building and Execution of Queries for Educational Process Management System

In this journal, it is saying about the main participants of university's educational process, which require tons of data for his or her daily activities associated with the students, exams, modules, students' assessments etc. Thus, the academic process management system should offer the way of retrieving the requested accessible info simply and quickly. The aim of this work is to design and implement a subsystem for the system known as IAMUniver, which will permit the end users to make queries and execute them to retrieve necessary data. IAMUniver is currently getting used within the faculty of informatics and applied mathematics of YSU on a check surroundings (Arakelyan & Balasanyan , n.d.).

Requirements Analysis for a Student Relationship Management System-- Results from an Empirical Study in Ivy League Universities

The higher education sector encounters increasing variety of scholars with a lot of various attributes, expectations, and demands. In times of sinking budgets and severe competition among universities, student relationship management (SRM) has become a key instrument in attracting paying students and retentive a long-lived relationship, that successively provides financial benefits and enhances the reputation of the university. in this paper, a structured literature review revealed an absence of demand analysis for a student relationship management system (SRMS) from the target cluster perspective. a web survey was conducted with students and alumni from four Ivy League universities. The survey showed that university administration must improve their relationship and communication habits with the target groups. as a result of trendy communication channels like social network, blogs and apps don't seem to be nonetheless wide-spread during this context, SRMS ought to be further enhanced to incorporate them (Lechtchinskaia , et al., 2012).

4. System Architecture

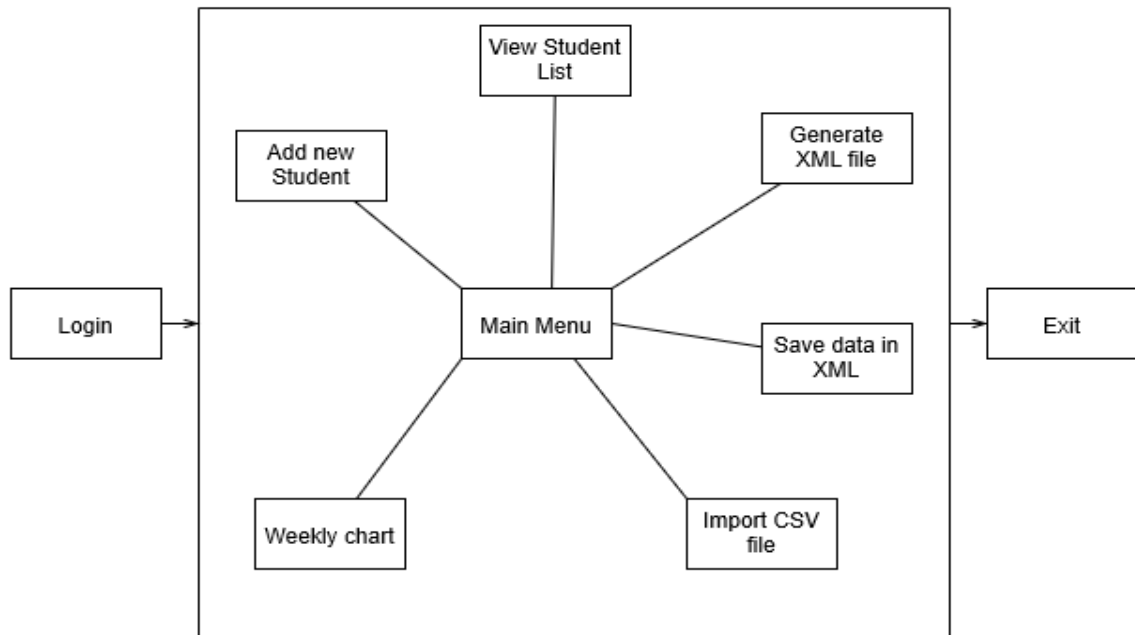


Figure 18: System Architecture of Student Information System

In the above figure, it shows that you should do login first to enter in main system. If username and password match, it directly shows main page with multiple features in menu. Many task can be perform here. After the completion of tasks, you can exit the application.

5. Class Diagram

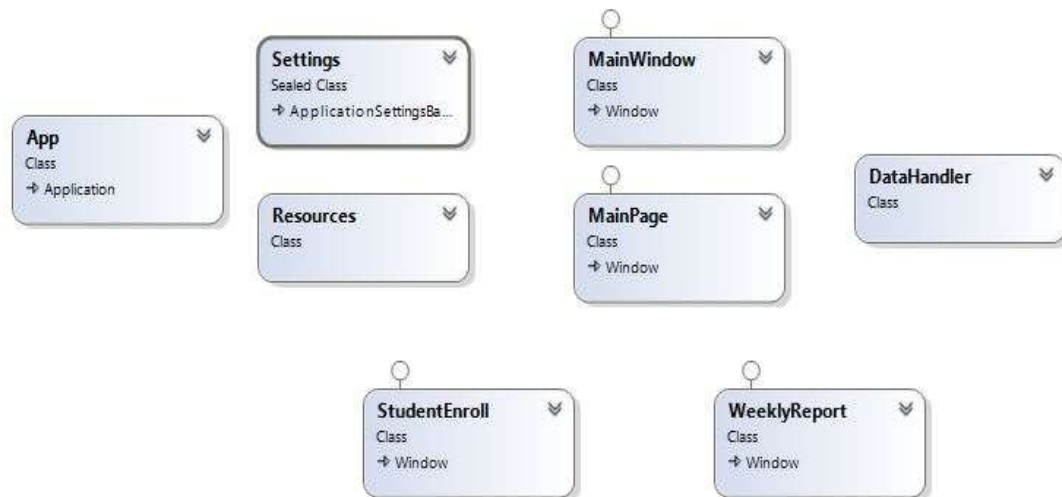


Figure 19: Class Diagram generated by visual studio

5.1. Individual Diagram

- Main Window

Methods	Description	MainWindow
btnsave	Login to main page.	Class → Window Methods btnexit_Click btnsave_Click MainWindow
btnexit	Exit the program.	

Table 1: Main Window

- Main Page

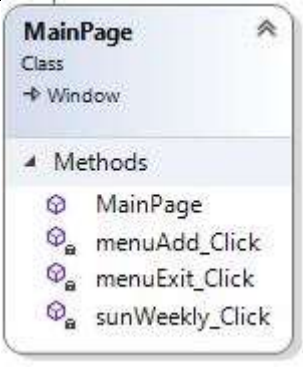
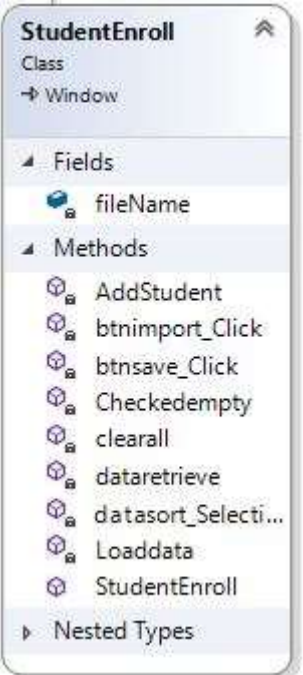
Method	Description	
menuAdd	Open StudentEnroll.	
sunWeekly	Open WeeklyReport.	
menuExit	Exit the program.	

Table 2: Main Page

- Student Enroll

Method	Description	
AddStudent	Pass textbox data in column name.	
btnimport	Import data from csv file and add in datagrid and XML file.	
btnsave	Create XML file and add newly data in XML.	
CheckedEmpty	Check empty value is passed or not.	
clearall	After data add it clear the textbox.	
dataretrieve	Load data when before entering Student Enroll.	

datasort_SelectionChanged	Sort data by name and date.	
Loaddata	If file exists it load data in datagrid after adding new student.	
StudentEnroll	it call a method dataretrieve.	

Table 3: Student Enroll

- Weekly Report


Method	Description	
WeeklyReport	Load total number of student according to Course enroll and chart according to total number of student.	

Table 4: Weekly Report

- Data Handler

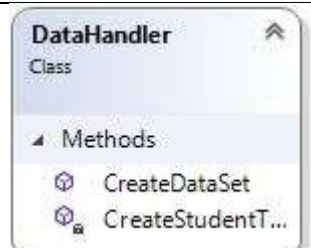
Method	Description	
CreateDataSet	Add data in table and called CreateStudentTable method.	
CreateStudentTable	Create columns in table Student.	

Table 5: Data Handler

6. Flowchart

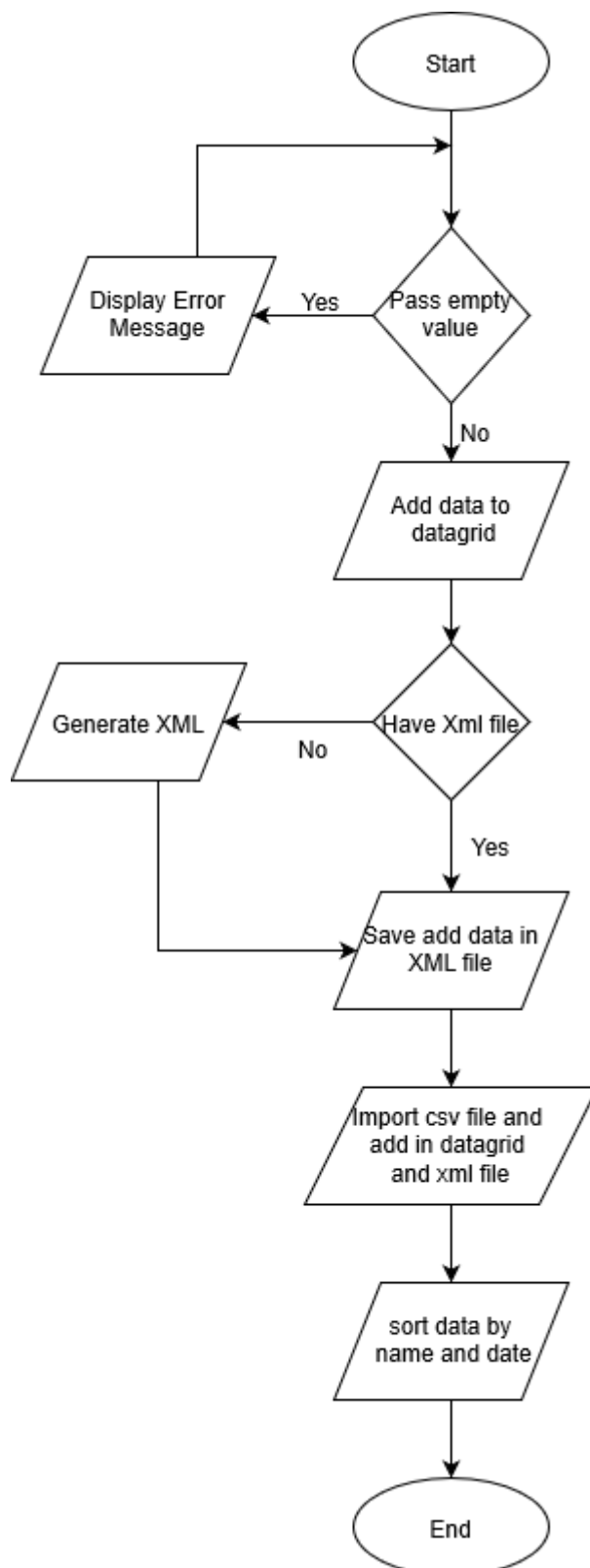


Figure 20: Flow Chart in Student Enroll

7. Algorithm

7.1. Bubble Sorting

Bubble sort is a simple sorting algorithm which is used to sort a given set of element provided in the form of array with number of element. It compare all the elements one by one and sort them in ascending order. Sorting takes place by step by step and compare it with the adjacent element and swapping them if required. (Anon., n.d.)

For example:

We take an unsorted array for our example. Bubble sort takes $O(n^2)$ time so we're keeping it short and precise.

Let us take five array numbers "65847", and sort the array from lowest number to greatest number using bubble sort. In each step, bold letters are being compared.

First pass:

1) (6 5 8 4 7) \rightarrow (5 6 8 4 7), Here, algorithm compares the first two element, and swaps since $6 > 5$.

2) (5 6 8 4 7) \rightarrow (5 6 8 4 7), No swap because they are already in order ($6 < 8$).

3) (5 6 8 4 7) \rightarrow (5 6 4 8 7), Swap since $8 > 4$ (5 6 4 8 7) \square (5 6 4 7 8), Swap since $8 > 7$

Second pass:

1) (5 6 4 7 8) \rightarrow (5 6 4 7 8), No swap because $5 < 6$

2) (5 6 4 7 8) \rightarrow (5 4 6 7 8), Swap since $6 > 4$

3) (5 4 6 7 8) \rightarrow (5 4 6 7 8), No swap because $6 < 7$

4) (5 4 6 7 8) \rightarrow (5 4 6 7 8), No swap because $7 < 8$

Third pass:

1) (5 4 6 7 8) \rightarrow (4 5 6 7 8), Swap since $5 > 4$

2) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $5 < 6$

3) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $6 < 7$

4) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $7 < 8$

Now, the array is sorted, but our algorithm does not know if it is completed because to be sorted one whole round should not contain any swap. Which is in round 4?

Fourth pass:

1) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $4 < 5$

2) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $5 < 6$

3) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $6 < 7$

4) (4 5 6 7 8) \rightarrow (4 5 6 7 8), No swap because $7 < 8$

Finally, the array is sorted and the algorithm can terminate. (Anon., n.d.)

8. Reflection

The development of this application was base on the desktop application. It is developed in Visual Studio 2019 in WPF with C# languages. The logic used in this application reflect real working environment of Student. The GUI designed is user interface.

While doing this project, I did a lot of research and develop this application. While doing this application, research are done in different stage from beginning to end. Through this project, I get to know how to work in WPF. The research are done in website, YouTube, etc. While doing this project, I face many problem and different errors while codding and research help to solve the problem. To solve problem module leader also help in different stage of error. Different mistake help to learn more about the new things.

After this project, I have some experience in Visual Studio. With this coursework, I got different experience. Some features like add student, create XML file, import csv file, sort data and create chart.

9. Conclusion

This is the first coursework of this module. This project (Student Information System) is completely base on desktop application(C# programming language).

Each function for this application is implement and work properly. To develop this application I face many difficulties because it's our first time working in Visual Studio with C# programming language. While doing a lot of research was done on website, journal, book, etc. While doing this project, lots of new thing came to know. Designing phase was easy but coding was difficult. In coding, error and bugs are solve by doing online research. Finally, project was complete with full functional.

10. References

Anon., n.d. *Bubble Sort*. [Online]
Available at: <https://www.geeksforgeeks.org/bubble-sort/>
[Accessed 10 01 2020].

Anon., n.d. *Bubble sort algorithm*. [Online]
Available at: <https://www.studytonight.com/data-structures/bubble-sort>
[Accessed 10 01 2020].

Arakelyan, A. & Balasanyan , A., n.d. Building and Execution of Queries for Educational Process Management System. *Yerevan State University* .

Lechtchinskaia , L., Friedrich, I. & Breitner , . M. H., 2012. Requirements Analysis for a Student Relationship Management System--Results from an Empirical Study in Ivy League Universities. *IEEE*, p. 15.

11. Appendix

Main Window

```

using System;
using System.Windows;

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

        private void btnsave_Click(object sender, RoutedEventArgs e)
        {
            String user, pass;
            user = editusername.Text;
            pass = editpassword.Password;
            if (user.Equals("") || pass.Equals(""))
            {
                MessageBox.Show("You can't pass empty value", "Login Error",
                    MessageBoxButton.OK, MessageBoxImage.Error);
            }
            else if (user != "admin" || pass != "admin")
            {
                MessageBox.Show("Username or password didn't match!", "Login
Error",
                    MessageBoxButton.OK, MessageBoxImage.Error);
                editusername.Clear();
                editpassword.Clear();
            }
            else
            {
                MessageBox.Show("Login Successful", "Login success",
                    MessageBoxButton.OK, MessageBoxImage.Information);
                MainPage hp = new MainPage();
                hp.Show();
                this.Close();
            }
        }

        private void btnexit_Click(object sender, RoutedEventArgs e)
        {
            MessageBoxResult dialogResult = MessageBox.Show("Are you Sure?", "Exit",
                MessageBoxButton.YesNo, MessageBoxImage.Information);
            if (dialogResult == MessageBoxResult.Yes)
            {
                this.Close();
            }
        }
    }
}

```

Main Page

```

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;

namespace StudentInformationSystem
{
    /// <summary>
    /// Interaction logic for MainPage.xaml
    /// </summary>
    public partial class MainPage : Window
    {
        public MainPage()
        {
            InitializeComponent();
        }
        private void menuAdd_Click(object sender, RoutedEventArgs e)
        {
            StudentEnroll studentEnroll = new StudentEnroll();
            studentEnroll.Show();
        }
        private void sunWeekly_Click(object sender, RoutedEventArgs e)
        {
            WeeklyReport weeklyReport = new WeeklyReport();
            weeklyReport.Show();
        }

        private void menuExit_Click(object sender, RoutedEventArgs e)
        {
            MessageBoxResult dialogResult = MessageBox.Show("Are you Sure?", "Exit",
            MessageBoxButton.YesNo, MessageBoxImage.Information);
            if (dialogResult == MessageBoxResult.Yes)
            {
                this.Close();
            }
        }
    }
}

```

Student Enroll

```

using Microsoft.Win32;
using System;
using System.Collections.Generic;
using System.Data;
using System.IO;
using System.Text.RegularExpressions;
using System.Windows;
using System.Windows.Controls;

```

```

namespace StudentInformationSystem
{
    /// <summary>
    /// Interaction logic for StudentEnroll.xml
    /// </summary>
    public partial class StudentEnroll : Window
    {
        private string fileName;
        public StudentEnroll()
        {
            InitializeComponent();
            dataretrieve();
        }

        private void btnsave_Click(object sender, RoutedEventArgs e)
        {
            Checkedempty();
            var handler = new DataHandler();
            var dataSet = new DataSet();
            //AddCSVStudent(dataSet);
            if (File.Exists(@"E:\StudentCWData.xml") &&
            File.Exists(@"E:\StudentCWSchema.schema"))
            {
                dataSet.ReadXml(@"E:\StudentCWData.xml");
                // dataSet.WriteXml(@"E:\StudentCWData.xml");
                dataSet.ReadXmlSchema(@"E:\StudentCWSchema.schema");
                //dataSet.WriteXmlSchema(@"E:\StudentCWSchema.schema");

                MessageBox.Show("student Enroled Sucessfully", "Message",
                MessageBoxButton.OK, MessageBoxImage.Information);
                //clearall();
            }
            else
            {
                dataSet = handler.CreateDataSet();

                // clearall();
            }
            AddStudent(dataSet);
            dataSet.WriteXmlSchema(@"E:\StudentCWSchema.schema");
            dataSet.WriteXml(@"E:\StudentCWData.xml");
            MessageBox.Show("New file Sucessfully generated", "Message",
            MessageBoxButton.OK, MessageBoxImage.Information);
            Loaddata();
            clearall();
        }

        private void Checkedempty()
        {
            var id = autoid.Text;
            var name = editfullname.Text;
            var address = editaddress.Text;
            var phone = editphone.Text;
            var gender = combogender.Text;
            var course = combocourse.Text;
            var date = datepick.Text;
            if (id.Equals("") || name.Equals("") || address.Equals("") ||
            phone.Equals("") || gender.Equals("") || course.Equals("") || date.Equals(""))
            {
                MessageBox.Show("You can't pass empty value", "Login Error",

```



```

        MessageBoxButton.OK, MessageBoxImage.Error);
    }
}

private void AddStudent(DataSet dataSet)
{
    var dr1 = dataSet.Tables["Student"].NewRow();
    dr1["id"] = autoid.Text;
    dr1["name"] = editfullname.Text;
    dr1["address"] = editaddress.Text;
    dr1["number"] = editphone.Text;
    dr1["gender"] = combogender.Text;
    dr1["courseenroll"] = combocourse.Text;
    dr1["date"] = datepick.SelectedDate != null ?
datepick.SelectedDate.Value : DateTime.Today;
    dataSet.Tables["Student"].Rows.Add(dr1);
}

private void clearall()
{
    autoid.Text = "";
    editfullname.Text = "";
    editaddress.Text = "";
    editphone.Text = "";
    combogender.Text = "";
    combocourse.Text = "";
    datepick.Text = "";
}

private void Loaddata()
{
    var dataSet = new DataSet();
    //var dataSet = datahandler.CreateDataSet();
    if (File.Exists(@"E:\StudentCWData.xml"))
    {
        dataSet.ReadXml(@"E:\StudentCWData.xml");
        datagrid.ItemsSource = dataSet.Tables["Student"].DefaultView;
    }
}

private void dataretrieve()
{
    var dataSet = new DataSet();
    //var dataSet = datahandler.CreateDataSet();
    if (File.Exists(@"E:\StudentCWData.xml"))
    {
        dataSet.ReadXml(@"E:\StudentCWData.xml");
        datagrid.ItemsSource = dataSet.Tables["Student"].DefaultView;
        MessageBox.Show("Student detail retrieved");
    }
}

private void datasort_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    var sortdata = datasort.SelectedIndex;
    if (sortdata == 1)
    {
        datagrid.Items.SortDescriptions.Clear();
    }
}

```

```

        datagrid.Items.SortDescriptions.Add(new
System.ComponentModel.SortDescription("date",
System.ComponentModel.ListSortDirection.Descending));
        datagrid.Items.Refresh();
    }
    else
    {
        datagrid.Items.SortDescriptions.Clear();
        datagrid.Items.SortDescriptions.Add(new
System.ComponentModel.SortDescription("name",
System.ComponentModel.ListSortDirection.Ascending));
        datagrid.Items.Refresh();
    }
}

private void btnimport_Click(object sender, RoutedEventArgs e)
{
    try
    {
        var dataSet = new DataSet();
        dataSet.ReadXml(@"E:\StudentCWData.xml");
        OpenFileDialog openfile = new OpenFileDialog();
        openfile.Filter = "CSV Files|*.csv";
        openfile.DefaultExt = ".csv";
        openfile.FilterIndex = 1;
        openfile.Multiselect = false;

        bool? fileselect = openfile.ShowDialog();
        if (fileselect != null || fileselect == true)
        {
            fileName = openfile.FileName;

            using (var reader = new StreamReader(fileName))
            {
                reader.ReadLine();
                while (!reader.EndOfStream)
                {
                    var line = reader.ReadLine();
                    var values = line.Split(',');
                    var dr1 = dataSet.Tables["Student"].NewRow();
                    dr1["id"] = values[0];
                    dr1["name"] = values[1];
                    dr1["address"] = values[2];
                    dr1["number"] = values[3];
                    dr1["gender"] = values[4];
                    dr1["courseenroll"] = values[5];
                    dr1["date"] = values[6];
                    dataSet.Tables["Student"].Rows.Add(dr1);

                    dataSet.WriteXml(@"E:\StudentCWData.xml");
                }
            }
            datagrid.ItemsSource = dataSet.Tables["Student"].DefaultView;
        }
    }
    catch (Exception)
    {
    }
}

```

```
    }
}
```

Weekly Report

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Data;
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 System.Windows.Controls.DataVisualization.Charting;

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

            var dataset = new DataSet();
            dataset.ReadXml(@"E:\StudentCWData.xml");
            DataTable dataTable = dataset.Tables[0];

            int total_computing = 0;
            int total_Multimedia_Technologies = 0;
            int total_Networks_and_IT_Security = 0;

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

            for (int i = 0; i < dataTable.Rows.Count; i++)
            {
                string col = dataTable.Rows[i]["courseenroll"].ToString();
                if (col == "Computing")
                {
                    total_computing++;
                }
                else if (col == "Multimedia Technologies")
                {
                    total_Multimedia_Technologies++;
                }
                else
                {
                    total_Networks_and_IT_Security++;
                }
            }
        }
    }
}
```

```

dt.Rows.Add("Computing", total_computing);
dt.Rows.Add("Multimedia Technologies", total_Multimedia_Technologies);
dt.Rows.Add("Networks and IT Security", total_Networks_and_IT_Security);
weeklygrid.DataContext = dt.DefaultView;

((PieSeries)piechart).ItemsSource = new KeyValuePair<string, int>[]
{
    new KeyValuePair<string, int>("Computing", total_computing),
    new KeyValuePair<string, int>("Multimedia Technologies",
total_Multimedia_Technologies),
    new KeyValuePair<string, int>("Networks and IT Security",
total_Networks_and_IT_Security)
};
    }
}
}

```

Data Handler

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;

namespace StudentInformationSystem
{
    class DataHandler
    {
        public DataSet CreateDataSet()
        {
            var ds = new DataSet();
            ds.Tables.Add(CreateStudentTable());
            return ds;
        }

        private DataTable CreateStudentTable()
        {
            var dt = new DataTable("Student");
            dt.Columns.Add("id", typeof(int));
            dt.Columns.Add("name", typeof(string));
            dt.Columns.Add("address", typeof(string));
            dt.Columns.Add("number", typeof(string));
            dt.Columns.Add("gender", typeof(string));
            dt.Columns.Add("courseenroll", typeof(string));
            dt.Columns.Add("date", typeof(string));
            return dt;
        }
    }
}

```