

# Informatics College Pokhara



informatics  
college pokhara

**Application Development**

**CS6004NI**

**Course Work 1**

**Submitted By: Simant 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>	User Interface is complete but not separated and have proper use of controls
<b>Manual data entry or import from csv</b>	appropriate use of data types but missing some properties required or missing CRUD operation
<b>Data Validation</b>	missing most of the validation
<b>Enrollment Report &amp; weekly report in tabular format</b>	Any one of the report is missing or not complete
<b>Course wise enrollment report &amp; Chart display</b>	any one component is missing or inappropriate data is shown
<b>Algorithm used for sorting &amp; proper sorting of data</b>	Sorting not implemented
<b>B. Documentation</b>	
<b>User Manual for running the application</b>	User Manual is below average. Is textual only.

# Marking Scheme

<b>Application architecture &amp; description of the classes ad methods sued</b>	architecture is included and satisfactory descriptoin of class 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>	very poorly developed application

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

## Overall Comment:

Code should be self explainable with less comments. Need some proper naming of the componen 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: Simant Gurung**

**College ID: NP04CP4S180020**

**University ID: 17031922**

## Table of Contents

<b>1. Introduction</b>	<b>1</b>
<b>1.1. Current Scenario</b>	<b>1</b>
<b>1.2 Proposed System</b>	<b>1</b>
<b>2. User Manual</b>	<b>2</b>
<b>3. Journal Articles</b>	<b>10</b>
<b>4. System Architecture</b>	<b>11</b>
<b>Architectural Diagram</b>	<b>11</b>
<b>Class Diagram</b>	<b>11</b>
<b>Individual Diagram</b>	<b>12</b>
<b>Flowchart for Reports</b>	<b>13</b>
<b>5. Sorting Algorithm</b>	<b>15</b>
<b>6. Reflection</b>	<b>16</b>
<b>7. Conclusion</b>	<b>16</b>
<b>References</b>	<b>17</b>
<b>Appendix</b>	<b>18</b>

## List of Figures

<b>Figure 1: Login Window .....</b>	<b>2</b>
<b>Figure 2: Main Window .....</b>	<b>3</b>
<b>Figure 3: Required Fields Missing Error .....</b>	<b>3</b>
<b>Figure 4: Correct form of data entry.....</b>	<b>4</b>
<b>Figure 5: XML File .....</b>	<b>4</b>
<b>Figure 6: Data Retrieval from XML file .....</b>	<b>5</b>
<b>Figure 7: Import File Window .....</b>	<b>5</b>
<b>Figure 8: CSV File .....</b>	<b>6</b>
<b>Figure 9: Importing CSV File.....</b>	<b>6</b>
<b>Figure 10: Data after importing from CSV File .....</b>	<b>7</b>
<b>Figure 11: Data Retrieval from XML File after importing CSV file.....</b>	<b>7</b>
<b>Figure 12: Bar Graph .....</b>	<b>8</b>
<b>Figure 13: Sorting by Name .....</b>	<b>8</b>
<b>Figure 14: Sorting by Date .....</b>	<b>9</b>
<b>Figure 15: Architectural Diagram .....</b>	<b>11</b>
<b>Figure 16: Class Diagram .....</b>	<b>11</b>
<b>Figure 17: Student Enrolment Flowchart.....</b>	<b>13</b>
<b>Figure 18: Import CSV flowchart .....</b>	<b>14</b>

List of Tables

**Table 1: LoginWindow Table..... 12**

**Table 2: Main Window Table ..... 12**

**Table 3: Import File Table..... 13**

## **1. Introduction**

This coursework is about designing and implementing Student Information System. This proposed system helps the company to keep records of student enrolment. It also consists of the features such as importing bulk data into our dataset. The system provides us platform for easier registration of student's record during registration. The name, phone number, address, course selection and registration date of newly enrolled students are stored into a separate file. Also, the system allows us to view the total number of enrolment of students into each courses and display those data in bar graph. Other available features are explained in the following section of this report.

### **1.1. Current Scenario**

In comparison to modern digital world, there are several educational institutes which are still lacking behind in data collection of new and old students. This really causes chaos when we are working with large number of data. It becomes difficult for administration to track the record of students enrolled into different courses.

### **1.2 Proposed System**

This proposed system targets those educational institutes who rely on paper based system for keeping records of student enrolment. Our newly designed system works well in terms of collection of large number of data. It contains of login feature which allow restricted individual to enter into the system and do the work. Furthermore, it consists of easy to use user-interface which helps people working under the institute to learn and adapt quickly into the digitalized system. Moreover, the data collected will always be safe within the system.



## 2. User Manual

I have listed the screenshot of the system below to show how a user can operated this system.

The initial stage a user has to go through before operating the system is login. The username and password of the system is “user”. Only with the correct entry of those fields, a new working window will appear.

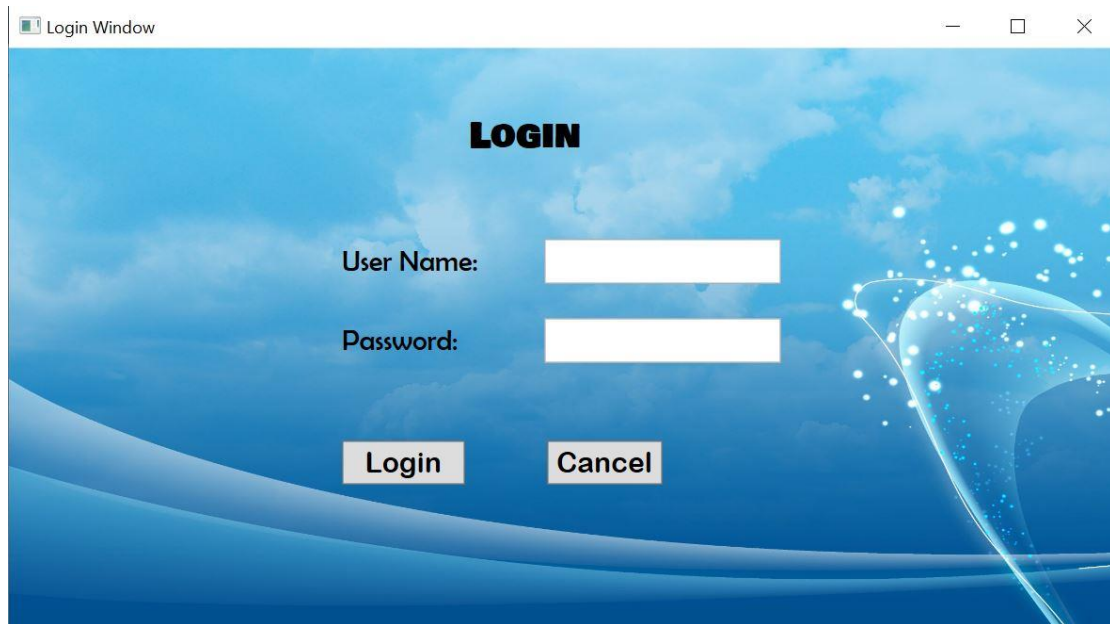


Figure 1: Login Window

After successfully logging into the system, it directly takes us into our Main Window. The Main Window consists of certain labels and textboxes for data entry of new students. There are buttons such as add which saves our entered data into XML file, retrieve to display file from our saved XML and also we have Import CSV button which will take us into next window which is Import File where we can import data from a CSV file into our data grid. On the right bottom corner, we have Show Enrol Report button which will display the total number of students of students enrolled in BIT and BBA course. The data will be generated in two data grid boxes below the buttons. Below data grid, we have two radio buttons for sorting the students according to alphabet and registration date. The registration number increased by 1 after each attempt to saving of data. It is automatically generated.

**Student Information System**

Reg No : 17031910      Contact Num:

Name :       Registration Date :  15

Address :

Course :

☐ SORT BY NAME      ☐ SORT BY DATE

Figure 2: Main Window

At first, a user admin can add a new student by filling up all the text boxes and then press add button on completion. If a user leave any of the text boxes empty, in that case it will not save those data and shows us alert message.

**Student Information System**

Reg No : 17031910      Contact Num:

Name : Ritesh Gurung      Registration Date :  15

Address :

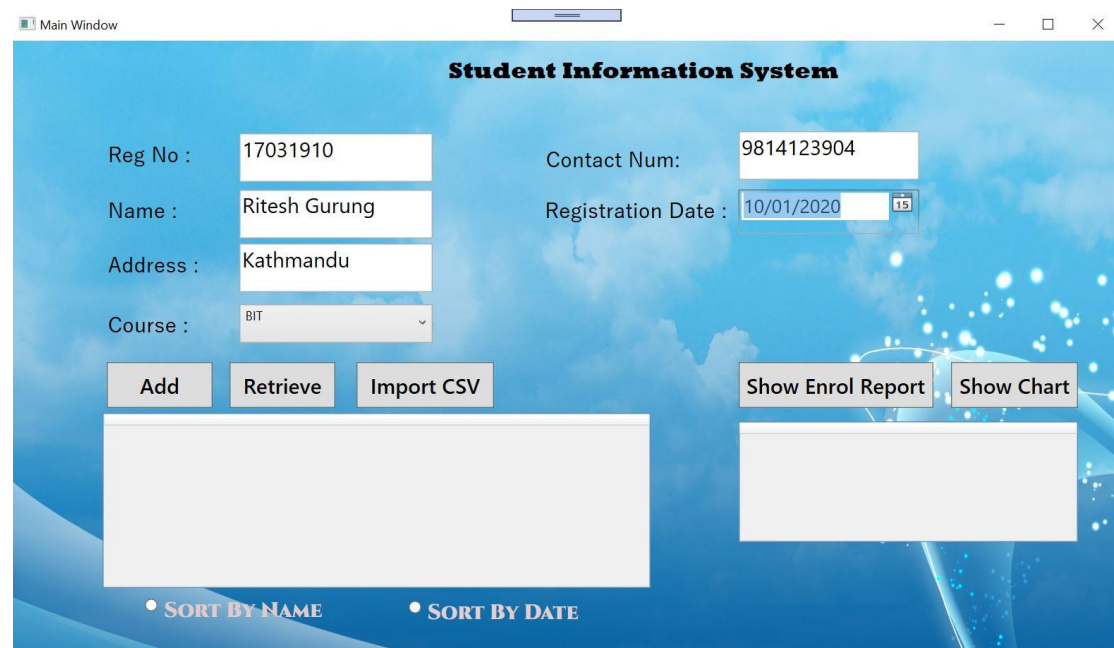
Course :

☐ SORT BY NAME      ☐ SORT BY DATE

Alert  
Required Fields Missing

Figure 3: Required Fields Missing Error

If user adds correct data by filling all the required text boxes like the given figure.

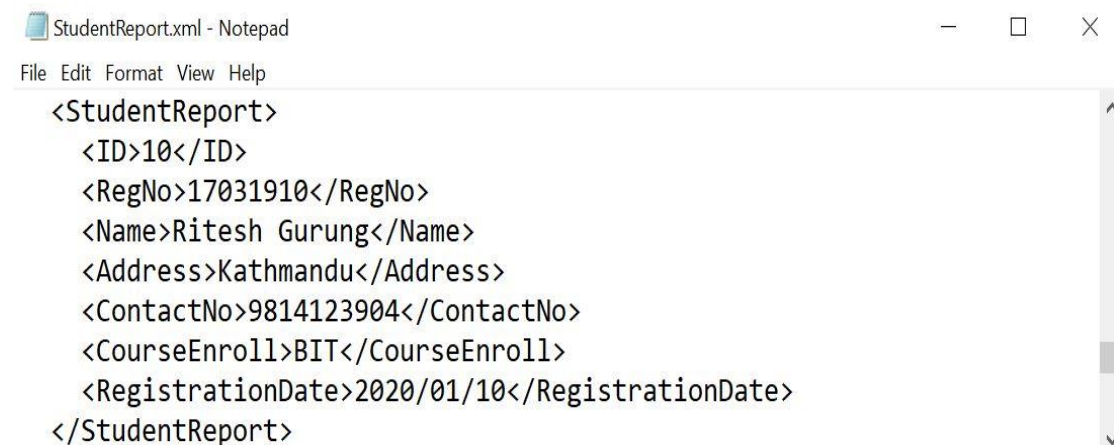


The screenshot shows a window titled "Main Window" with a blue background and the title "Student Information System". The form contains the following fields and buttons:

- Reg No : 17031910
- Contact Num: 9814123904
- Name : Ritesh Gurung
- Registration Date : 10/01/2020 (calendar icon)
- Address : Kathmandu
- Course : BIT (dropdown menu)
- Buttons: Add, Retrieve, Import CSV, Show Enrol Report, Show Chart
- Sorting options: ☒ SORT BY NAME, ☐ SORT BY DATE

Figure 4: Correct form of data entry

Then, our data are successfully saved into our StudentReport.XML File. All the information collected from registration are directly saved into this file.



```
<StudentReport>
  <ID>10</ID>
  <RegNo>17031910</RegNo>
  <Name>Ritesh Gurung</Name>
  <Address>Kathmandu</Address>
  <ContactNo>9814123904</ContactNo>
  <CourseEnroll>BIT</CourseEnroll>
  <RegistrationDate>2020/01/10</RegistrationDate>
</StudentReport>
```

Figure 5: XML File

On pressing Retrieve button we get the newly updated data which is of registration no 17031910.

**Student Information System**

Reg No : 17031911      Contact Num:

Name :       Registration Date : 10/01/2020

Address :

Course : BIT

**Add   Retrieve   Import CSV   Show Enrol Report   Show Chart**

ID	RegNo	Name	Address	ContactNo	CourseEnroll	RegistrationDate
4	17031904	Sonam Gurung	Pokhara	9806645233	BIT	2020/01/08
5	17031905	Roshan Tiwari	Chitwan	9815178342	BBA	2020/01/08
6	17031906	Rahul Pun	Baglung	9814110934	BBA	2020/01/08
7	17031907	Manoj Thapa	Palpa	9856084501	BIT	2020/01/09
8	17031908	Naresh Rai	Manang	9814172993	BBA	2020/01/10
9	17031909	Rakesh Baral	Butwal	9856021844	BBA	2020/01/10
10	17031910	Ritesh Gurung	Kathmandu	9814123904	BIT	2020/01/10

• **SORT BY NAME**      • **SORT BY DATE**

Figure 6: Data Retrieval from XML file

To import new CSV file, we should first click on Import CSV button from Main Window. Import File Window will appear like given below.

**Back   Import CSV**

Figure 7: Import File Window



Before trying to import CSV from our device, we should have a CSV file containing a bulk data of students with column name matching our previously created XML files.



Figure 8: CSV File

After pressing import CSV button in Import File Window a dialog box appears where we need to select our created CSV file.

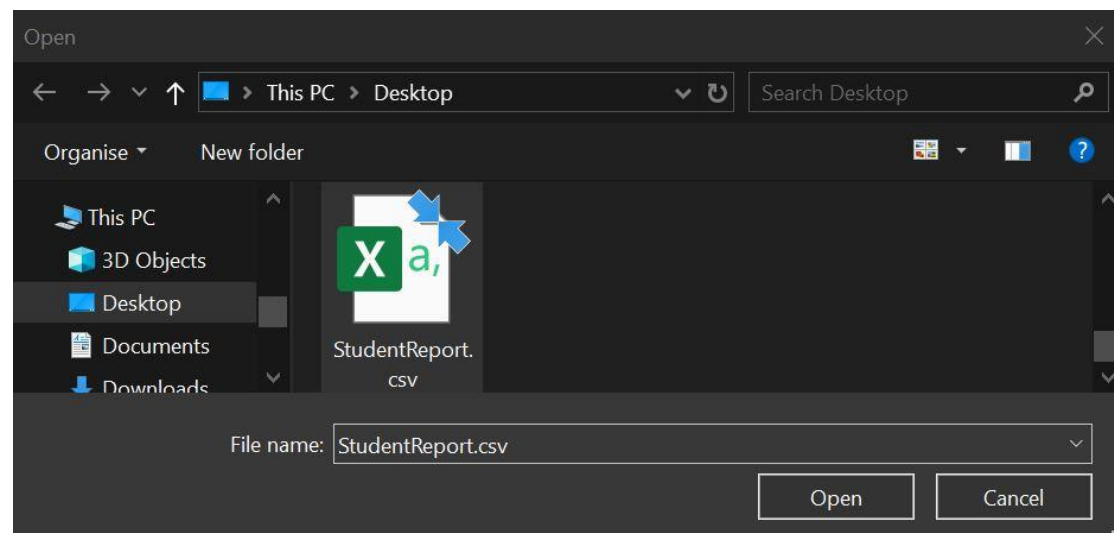


Figure 9: Importing CSV File

After we press open, in the data grid we can see the addition of two new data from CSV file into our display.



Figure 10: Data after importing from CSV File

Those files will be saved to our main StudentReport.XML file as well. To check it, we need to go back to our main window and again press retrieve button. To check the enrolment of student in each course we press Show Enrol Report button. Data will be shown as below in the data grid.

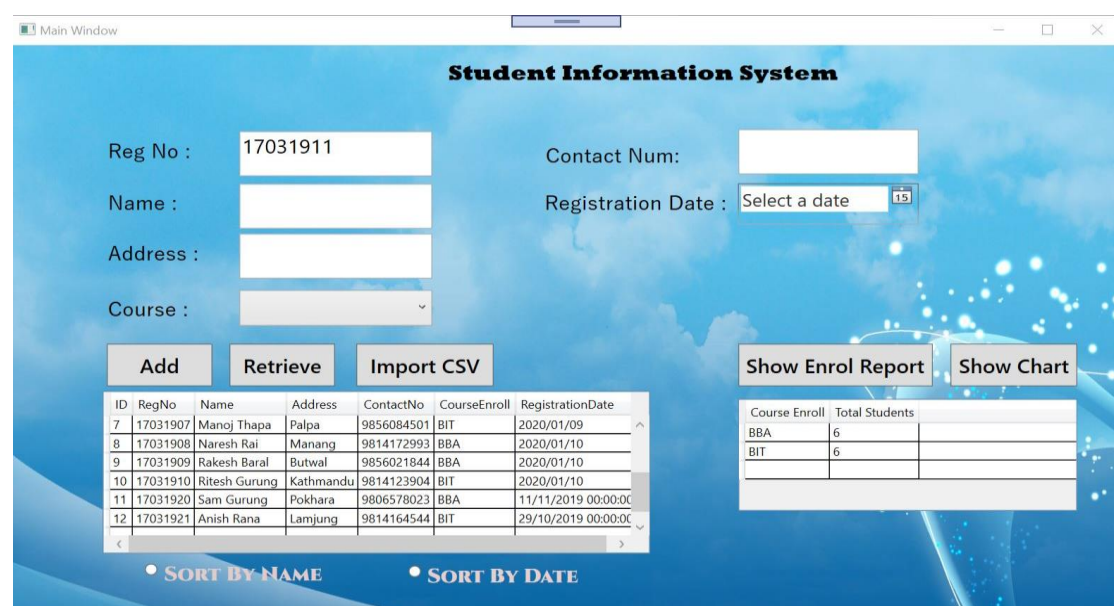


Figure 11: Data Retrieval from XML File after importing CSV file

If we press on Show Chart button, a bar graph will appear showing total number of students enrolled in each courses i.e. BBA and BIT.

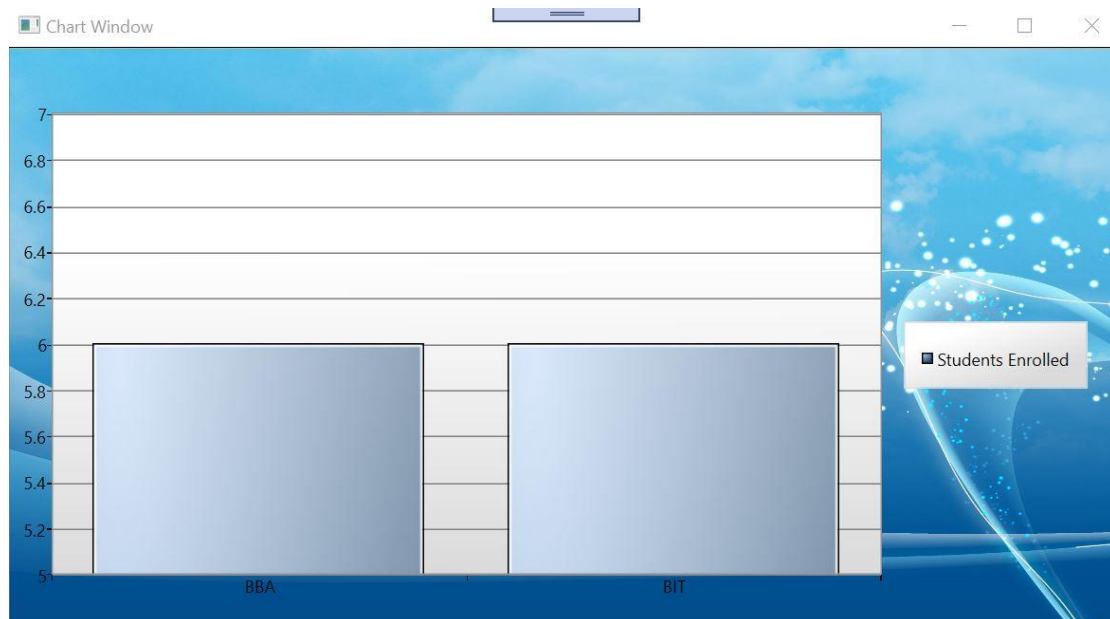


Figure 12: Bar Graph

There is sorting feature available in the system if we press any of two radio buttons present below our data grid. Figure below shows shorting of information according to name in alphabetical order.

**Student Information System**

Reg No : 17031911      Contact Num:

Name :

Address :

Course :

ID	RegNo	Name	Address	ContactNo	CourseEnroll	RegistrationDate
3	17031903	Anil Rai	Pokhara	9806587885	BIT	2020/01/04
12	17031921	Anish Rana	Lamjung	9814164544	BIT	29/10/2019 00:00:00
2	17031902	Anish Thapa	Kathmandu	9867134562	BIT	2020/01/03
7	17031907	Manoj Thapa	Palpa	9856084501	BIT	2020/01/09
8	17031908	Naresh Rai	Manang	9814172993	BBA	2020/01/10
1	17031901	Pawan Tiwari	Lamjung	9816423465	BBA	2020/01/03

☒ SORT BY NAME      ☐ SORT BY DATE

Figure 13: Sorting by Name

The following figure shows the information arranged according to the date of registration.

**Student Information System**

Reg No :  Contact Num:

Name :  Registration Date :

Address :

Course :

ID	RegNo	Name	Address	ContactNo	CourseEnroll	RegistrationDate
11	17031920	Sam Gurung	Pokhara	9806578023	BBA	11/11/2019 00:00:00
1	17031901	Pawan Tiwari	Lamjung	9816423465	BBA	2020/01/03
2	17031902	Anish Thapa	Kathmandu	9867134562	BIT	2020/01/03
3	17031903	Anil Rai	Pokhara	9806587885	BIT	2020/01/04
4	17031904	Sonam Gurung	Pokhara	9806645233	BIT	2020/01/08
5	17031905	Roshan Tiwari	Chitwan	9815178342	BBA	2020/01/08

☒ SORT BY NAME
 ☐ SORT BY DATE

Figure 14: Sorting by Date



### **3. Journal Articles**

#### **1. C# vs Other Programming Languages**

This article has clearly described what is similar, what is different, and the motivation behind the C# preference. The author has made his comparison with other widely popular programming languages like C++ and Java. He has also mentioned that environment of our software is most important for choosing C# (Gul, 2016).

#### **2. Do flexible admission systems affect student enrolment? Evidence from UK universities.**

The authors has revealed the current level of flexible admission systems at UK universities, and explored its impact on student enrolment rates. In order to understand the impact of flexible admission systems in UK, six statistical tests were conducted. They have also found that no robust evidence exists to support claims that universities which apply a higher level of FAS have higher student enrolment (Massoud & Ayoubi, 2018).

#### **3. Online student enrolment system**

In this article the author has described his experiences with the development of online enrolment. He has mentioned that it is a complicated workflow that is based on strict university enrolment rules. He also mentions how in this modern age, universities are still enrolling student via paper based forms and says how his system will help international students without travelling to the universities abroad to fill paper based enrolment (Then, 2006).

## 4. System Architecture

### Architectural Diagram

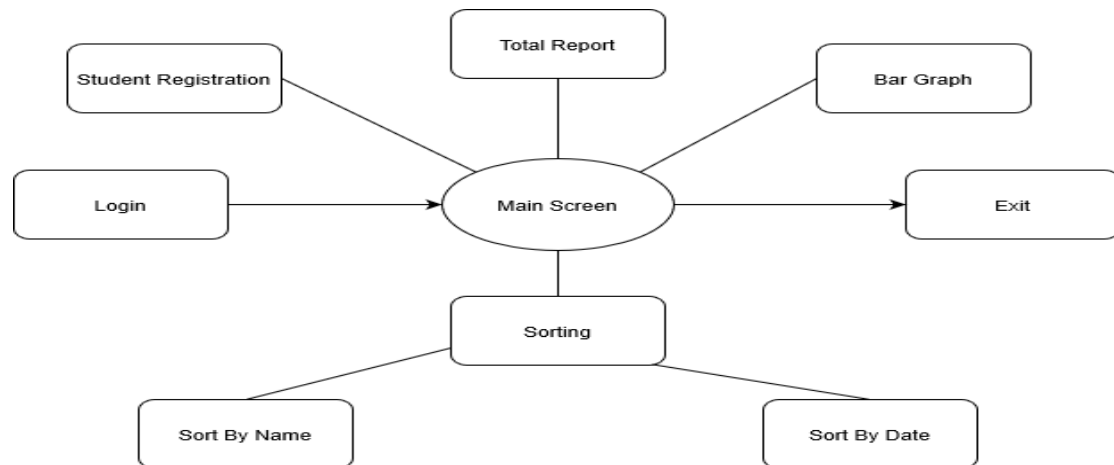


Figure 15: Architectural Diagram

### Class Diagram

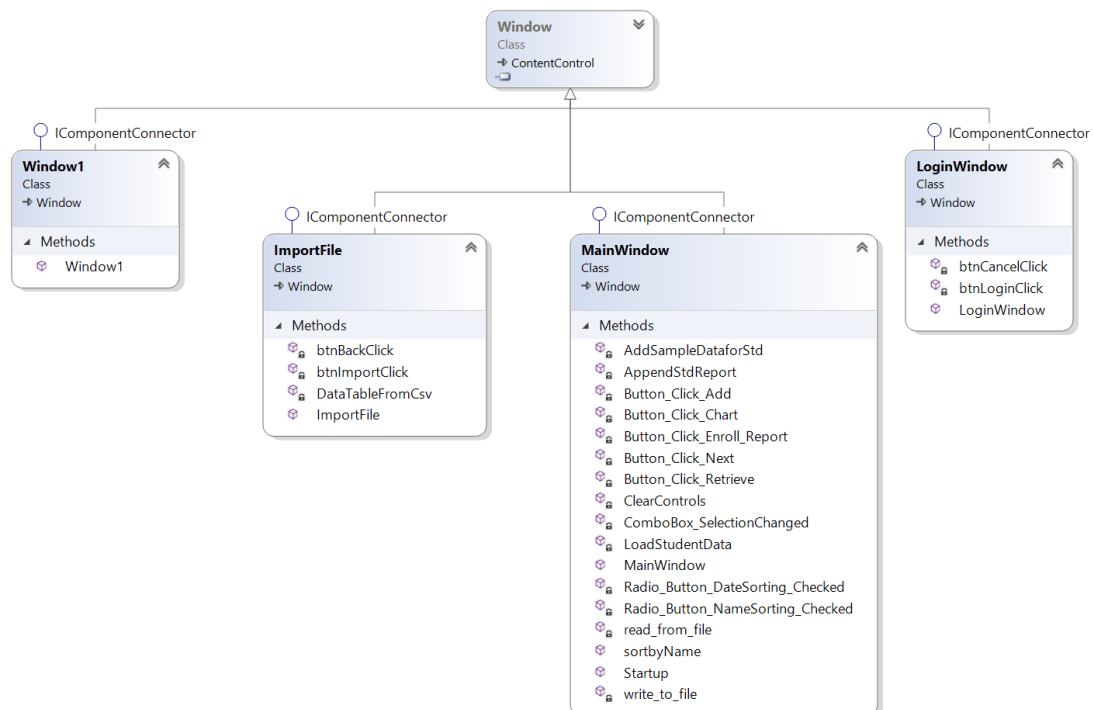


Figure 16: Class Diagram

## Individual Diagram

### 1. LoginWindow

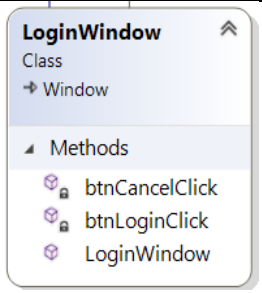
Methods	Description	Diagram
btnLoginClick	Checks username and password and opens Main Window if they are correct.	 <pre> classDiagram     class LoginWindow {         &lt;&lt;window&gt;&gt;         +btnCancelClick()         +btnLoginClick()         +LoginWindow()     } </pre>
btnLoginClick	Closes this window.	

Table 1: LoginWindow Table

### 2. MainWindow

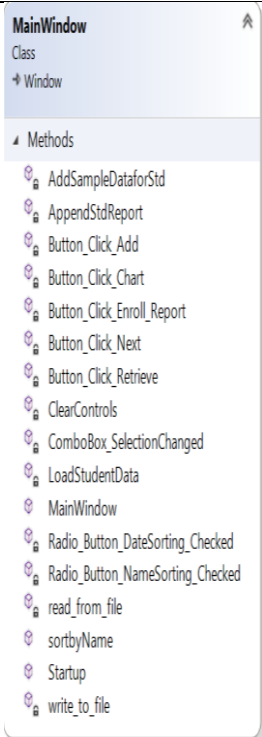
Methods	Description	Diagram
AddSampleDataforStd	Add data into table.	 <pre> classDiagram     class MainWindow {         &lt;&lt;window&gt;&gt;         +AddSampleDataforStd()         +AppendStdReport()         +Button_Click_Add()         +Button_Click_Chart()         +Button_Click_Enroll_Report()         +Button_Click_Next()         +Button_Click_Retrieve()         +ClearControls()         +ComboBox_SelectionChanged()         +LoadStudentData()         +MainWindow()         +Radio_Button_DateSorting_Checked()         +Radio_Button_NameSorting_Checked()         +read_from_file()         +sortByName()         +Startup()         +write_to_file()     } </pre>
AppendStdReport	Appends data into data grid.	
Button_Click_Add	Adds data from text boxes to XML.	
Button_Click_Chart	Displays chart.	
Button_Click_Retrieve	Retrieves data from XML to data grid.	
ClearControls	Clear text boxes.	
LoadStudentData	Load data XML file.	
Radio_Button_DateSorting_Checked	Sorts by date.	
Radio_Button_NameSorting_Checked	Sorts by name.	
read_from_file	Reads data from individual data XML.	
Write_to_file	Writes data to individual XML file.	

Table 2: Main Window Table

### 3. ImportFile

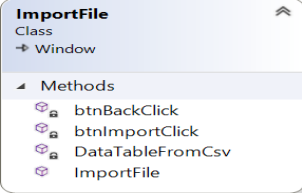
Methods	Description	Diagram
btnBackClick	Takes back to Main Window	 <pre> classDiagram     class ImportFile {         +Window         +btnBackClick         +btnImportClick         +DataTableFromCsv         +ImportFile     } </pre>
btnImportClick	Opens dialog to select file.	
DataTableFromCSV	Function to create data table from CSV file	

Table 3: Import File Table

### Flowchart for Reports

#### 1. Student Enrolment Flowchart

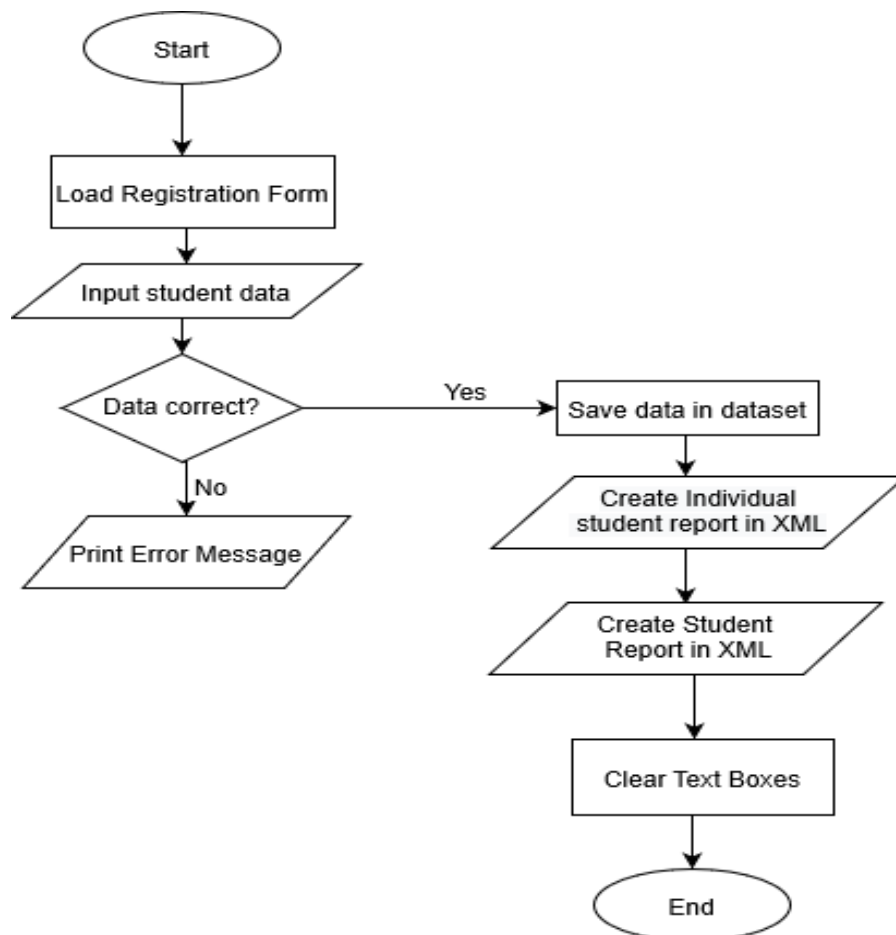
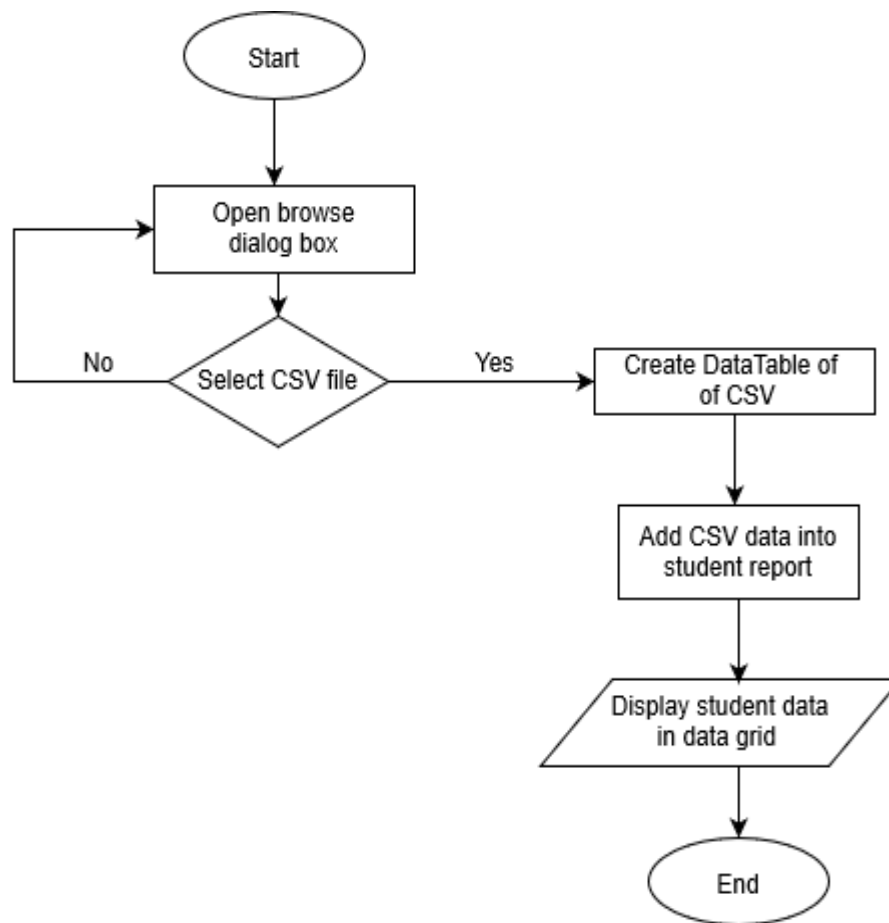


Figure 17: Student Enrolment Flowchart

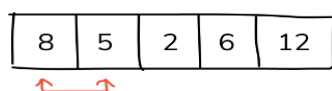
## 2. Import CSV Flowchart

*Figure 18: Import CSV flowchart*

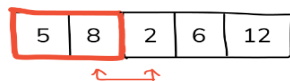
## 5. Sorting Algorithm

Bubble sort algorithm is used for sorting student information in our system. It is a sorting algorithm that interacts through a given array of elements and compares each pair of adjacent elements one after the other. If the first element is greater than second element, then there is swapping between elements and if not, then it moves on to the next pair of elements. The main motive of bubble sort is to sort the given array in ascending order.

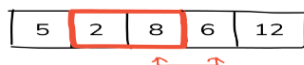
Let us consider an array having 5 elements.



- Start with the 1<sup>st</sup> element and compare it with the adjacent element. Here,  $8 > 5$ , so we swap in this process.



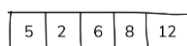
- We compare second and third element.  $8 > 2$ , so again we should swap.



- Then, we should look at the third and fourth element.  $8 > 6$ , so we should swap again.



- We should compare fourth and fifth element.  $8 < 12$ . So, we do not need to swap.



- At last we have found we have found result after first iteration.

In general, it takes  $(n-1)$  iterations in order to sort a given array using this algorithm, where  $n$  is the number of elements in the array. So, in our case, since we have 5 elements, it takes  $(5 - 1)$  which is 4 iterations to completely sort the array (faceprep, 2019).

## 6. Reflection

The developed system is Student Information System made using Visual Studio 2019 with C# language. The actual motive of this project is to make system able to manually send data or use import method to take information from the external CSV file and save it in our data set. The system provides easy to use user interface for convenience of new users. This kind of system helps educational institutes to keep track of student data securely.

I had no experience of using Visual Studio before. Therefore, it was hard to cope with the C# language. Features like drag and drop were similar to what we've been doing in NetBeans. But, implementation of charts was quite difficult in this method as we have to install toolkit and add reference in this system. Coursework was difficult at times during its development phase. The design was an easy and fun to do in this coursework. The objects and connection between tables really takes the development of this system to next level. Though, the knowledge I gained from Application Development classes were really beneficial for creating data tables and working with serialization and deserialization of data. Overall, it was a good learning experience for me as I got opportunity to learn something new.

## 7. Conclusion

The coursework focuses on handling the data and information securely. The system developed could be further taken into next level with more advancement in storing data properly. The project completion would not have been possible without the proper guidelines from our module leader. The classes were relatively more effective for me to learn a new language like C# and do a project with so much difficulty. It was challenging task but of course we need to take them lightly as those hurdles teaches us to be more strong and focused. At last, I would be thankful to our module leader and I suppose that the knowledge we obtained will equally be useful for us while working on future projects like this.

## References

- faceprep. (2019). *faceprep*. Retrieved January 08, 2020, from [www.faceprep.in: https://www.faceprep.in/bubble-sort-in-c/](https://www.faceprep.in/bubble-sort-in-c/)
- Gul, S. A. (2016, November 26). C# vs Other Programming Languages. Retrieved January 1, 2020
- Massoud, H. K., & Ayoubi, R. M. (2018, May 25). Do flexible admission systems affect student enrolment? Evidence from UK universities. Retrieved January 04, 2019
- Then, P. (2006, January). Online studnet enrollment system. Retrieved January 05, 2010



## Appendix

### 1. LoginWindow.xaml.cs

```
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 ApplicationDevelopmentCW
{
    /// <summary>
    /// Interaction logic for LoginWindow.xaml
    /// </summary>
    public partial class LoginWindow : Window
    {
        public LoginWindow()
        {
            InitializeComponent();
        }

        private void btnLoginClick(object sender, RoutedEventArgs e)
        {
            if (txt_username.Text == "user" && txt_password.Password == "user")
            {
                MainWindow showWindow = new MainWindow();
                showWindow.Show();
                this.Close();
            }
            else
            {
                MessageBox.Show("Login Denied", "Alert");
                txt_username.Clear();
                txt_password.Clear();
            }
        }

        private void btnCancelClick(object sender, RoutedEventArgs e)
        {
            this.Close();
        }
    }
}
```

## 2. MainWindow.xaml.cs

```
using DataHandler;
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.Navigation;
using System.Windows.Shapes;

namespace ApplicationDevelopmentCW
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {

        public MainWindow()
        {
            InitializeComponent();
            Startup();

            txtRegNo.Text = read_from_file();
            //LoadStudentData();

        }

        public void Startup()
        {
            var i = 0;
        }

        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["CourseEnroll"] = comboBoxCourse.Text;
            //dr1["RegistrationDate"] = DateTime.Now.ToString("MM/dd/yyyy hh:mm
tt");
        }
    }
}
```

```

        dr1["RegistrationDate"] =
datePick.SelectedDate.Value.ToString("yyyy/MM/dd");
        dataSet.Tables["Student"].Rows.Add(dr1);
    }

    private void AppendStdReport(DataSet dataSet)
    {
        var handler = new Handler();

dataSet.Tables["StudentReport"].ReadXml(@"Files\\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["CourseEnroll"] = comboBoxCourse.Text;
        //dr2["RegistrationDate"] = DateTime.Now.ToString("MM/dd/yyyy hh:mm
tt");
        dr2["RegistrationDate"] =
datePick.SelectedDate.Value.ToString("yyyy/MM/dd");
        dataSet.Tables["StudentReport"].Rows.Add(dr2);

dataSet.Tables["StudentReport"].WriteXml(@"Files\\StudentReport.xml");
    }
    private void Button_Click_Add(object sender, RoutedEventArgs e)
    {
        String reg = txtRegNo.Text;
        String fullname = txtName.Text;
        String address = txtAddress.Text;
        String contact = txtContact.Text;
        String course = comboBoxCourse.Text;
        //String date = datePick.SelectedDate.Value.ToString("yyyy/MM/dd");

        if (reg != "" && fullname != "" && address != "" && contact != ""
&& course != "")
        {
            grdStd.IsReadOnly = true;

            var handler = new Handler();
            var dataSet = handler.CreateDataSet();
            AddSampleDataforStd(dataSet);
            AppendStdReport(dataSet);

            var regno = txtRegNo.Text;
            var name = txtName.Text;

dataSet.Tables["Student"].WriteXml(@"Files\\IndividualStudentRecords\\" + name
+ "CWData" + regno + ".xml");

            write_to_file(txtRegNo.Text);

```

```
txtRegNo.Text = read_from_file();

MessageBox.Show("Student Details saved to file", "Alert");

ClearControls();

}
else
{
    MessageBox.Show("Required Fields Missing", "Alert");
}
}

private void write_to_file(string text)
{

    System.IO.File.WriteAllText(@"Files\count.txt", text);

}

public void sortByName()
{

}

private string read_from_file()
{

    string text = System.IO.File.ReadAllText(@"Files\count.txt");

    int i;

    i = int.Parse(text.ToString());
    i = i + 1;

    return i.ToString();
}

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

private void LoadStudentData()
{

    if (System.IO.File.Exists(@"Files\StudentReport.xml"))
    {
        var handler = new Handler();

        var dataSet = new DataSet();
```

```
        dataSet.ReadXml(@"Files\\StudentReport.xml");

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

    }
    else
        MessageBox.Show("Data Retrieval Unsucessful", "Alert");
}

private void Button_Click_Retrieve(object sender, RoutedEventArgs e)
{
    ClearControls();
    LoadStudentData();
}

private void ComboBox_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
}

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

    var dataSet = new DataSet();

    dataSet.ReadXml(@"Files\\StudentReport.xml");

    DataTable dtStdReport = new DataTable();
    dtStdReport = dataSet.Tables[0];
    dtStdReport.DefaultView.Sort = "Name ASC";
    grdStd.DataContext = dtStdReport.DefaultView;
}

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

    var dataSet = new DataSet();

    dataSet.ReadXml(@"Files\\StudentReport.xml");

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

```

    }

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

        dataSet.ReadXml(@"Files\\StudentReport.xml");

        DataTable dtStdReport = dataSet.Tables[0];

        int total_BIT = 0;
        int total_BBA = 0;

        DataTable dt = new DataTable("newTable");
        dt.Columns.Add("Course Enroll", typeof(String));
        dt.Columns.Add("Total Students", typeof(int));

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

        dt.Rows.Add("BBA", total_BBA);
        dt.Rows.Add("BIT", total_BIT);

        grdReport1.DataContext = dt.DefaultView;
    }

private void Button_Click_Next(object sender, RoutedEventArgs e)
{
    ImportFile showWindow = new ImportFile();
    showWindow.Show();
    this.Close();
}

private void Button_Click_Chart(object sender, RoutedEventArgs e)
{
    Window1 showWindow = new Window1();
    showWindow.Show();
}
}
}

```

### 3. ImportFile.xaml.cs

```

using DataHandler;
using Microsoft.Win32;
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.OleDb;
using System.Globalization;
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 Path = System.IO.Path;

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

        private void btnImportClick(object sender, RoutedEventArgs e)
        {
            OpenFileDialog dialog = new OpenFileDialog();
            dialog.DefaultExt = ".csv";
            Nullable<bool> load = dialog.ShowDialog();

            if (load == true)
            {
                DataTable stdinfo = DataTableFromCsv(dialog.FileName, true);
                DataTable dataTable = stdinfo.Clone();
                dataTable.Columns["RegNo"].DataType = typeof(String);
                dataTable.Columns["ContactNo"].DataType = typeof(String);
                dataTable.Columns["RegistrationDate"].DataType =
typeof(String);
                foreach (DataRow row in stdinfo.Rows)
                {
                    dataTable.ImportRow(row);
                }

                Handler handler = new Handler();
                DataSet dataSet = handler.CreateDataSet();

                dataSet.Tables["StudentReport"].ReadXml(@"Files\\StudentReport.xml");
                dataSet.Tables["StudentReport"].Merge(dataTable);

                dataSet.Tables["StudentReport"].WriteXml(@"Files\\StudentReport.xml");
            }
        }
    }
}

```

```

        var dataset = new DataSet();
        //dataset.ReadXml(@"D:\StudentReport.xml");
        DataTable dataTable1 = dataset.Tables["StudentReport"];
        dataImportGrid.DataContext = dataTable1.DefaultView;
    }
    else
    {
        MessageBox.Show("File not found", "Alert");
    }
}

static DataTable DataTableFromCsv(string path, bool isFirstRowHeader)
{
    string header = isFirstRowHeader ? "Yes" : "No";

    string pathOnly = Path.GetDirectoryName(path);
    string fileName = Path.GetFileName(path);

    string sql = @"SELECT * FROM [" + fileName + "]";

    using (OleDbConnection connection = new OleDbConnection(
        @"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" +
pathOnly +
        ";Extended Properties=\"Text;HDR=" + header + "\""))
    using (OleDbCommand command = new OleDbCommand(sql, connection))
    using (OleDbDataAdapter adapter = new OleDbDataAdapter(command))
    {
        DataTable dataTable = new DataTable();
        dataTable.Locale = CultureInfo.CurrentCulture;
        adapter.Fill(dataTable);
        return dataTable;
    }
}

private void btnBackClick(object sender, RoutedEventArgs e)
{
    MainWindow showWindow = new MainWindow();
    showWindow.Show();
    this.Close();
}
}
}

```



## 4. Window1.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.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;

namespace ApplicationDevelopmentCW
{
    public partial class Window1 : Window
    {
        public Window1()
        {
            InitializeComponent();

            var dataSet = new DataSet();

            dataSet.ReadXml(@"Files\StudentReport.xml");

            DataTable dtStdReport = dataSet.Tables[0];

            int total_BIT = 0;
            int total_BBA = 0;

            DataTable dt = new DataTable("newTable");
            dt.Columns.Add("Course Enroll", typeof(String));
            dt.Columns.Add("Total Students", typeof(int));

            for (int i = 0; i < dtStdReport.Rows.Count; i++)
            {
                String col = dtStdReport.Rows[i]["CourseEnroll"].ToString();
                if (col == "BIT")
                {
                    total_BIT++;
                }
                else if (col == "BBA")
                {
                    total_BBA++;
                }
            }
            dt.Rows.Add("BBA", total_BBA);
            dt.Rows.Add("BIT", total_BIT);
            ((ColumnSeries)chartEnroll).ItemsSource =
new KeyValuePair<string, int>[] {
new KeyValuePair<string, int>("BBA", total_BBA),
new KeyValuePair<string, int>("BIT", total_BIT)};
        }
    }
}

```