Marking Scheme

Informatics College Pokhara

informatics
college pokhara

Application Development

CS6004NI

Course Work 1

Submitted By: **Bhawana 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	not properly saved or imported data
Data Validation	No validation at all
Enrollment Report & weekly report in tabular format	very poorly executed reports and data not shown accurately
Course wise enrollment report & Chart display	any one component is missing or inappropriate data is shown
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 average. Includes description for all interfaces

Application architecture & description of the classes ad methods sued	very poorly explained.
Flow chart, algoriathms and data sctructures used	very poorly explained and no 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	unusable system

Overall Grade:	**D**	**D**

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 is implemented with some minor bugs.

CU6004NP Application Development

30% Individual Coursework

2018-19 Autumn

Name: Bhawana Shrestha

College ID: NP04CP4S180004

University ID: 17031939

I confirm that I understand my coursework needs to be submitted online via Google Classroom under the relevant module page before the deadline in order for my assignment to be accepted and marked. I am fully aware that late submissions will be treated as non-submission and a mark of zero will be awarded.

Table of Contents

List of Figure

List of Tables

1. Introduction

The developed system is a Student Information System. The system has been designed to meet all the requirements and have been tested repeatedly to make sure the system is void of an error. The system consists of features like adding student details using a form, importing details from and external file, displaying tabular reports and charts to showing total number of students in each program and sorting the student details according to names and registration date. All the available features have been thoroughly described in the body of the report.

1.1 Current Scenario

There are still a lot of schools that use the traditional way of keeping records i.e. by the use of files and register. This way of storing records is rather unsafe and easy to manipulate. Furthermore, it is also a tedious and time-consuming to find one required record from these piles of records. These data are also hard to analyse manually and is not a very reliable means for future referencing.

1.2 Proposed System

The developed software can be used in any company to keep record of the students enrolled. The main objective of the developed system is to keep track of the students enrolled, generate reports and charts, sort student details by their names and registration dates. The system is lite and easy to use for any user with little computer knowledge and training.

2. User Manual

The detailed information to run the program has been shown below along with proper screenshots:

Login Screen

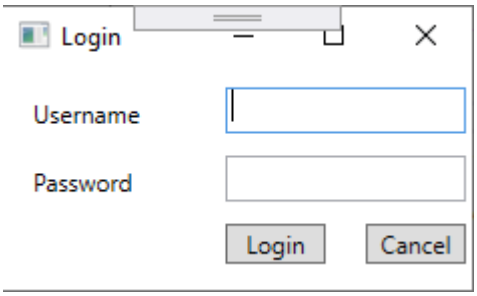


Figure 1: Login screen

- The username and password for this system is admin and admin respectively.
- If wrong username or password is entered it will display a message as shown below.

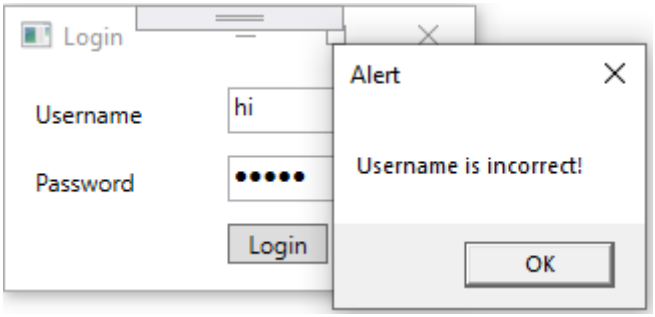


Figure 2: Message displayed when username is incorrect

Figure 3: Message displayed when password is incorrect

Main Page

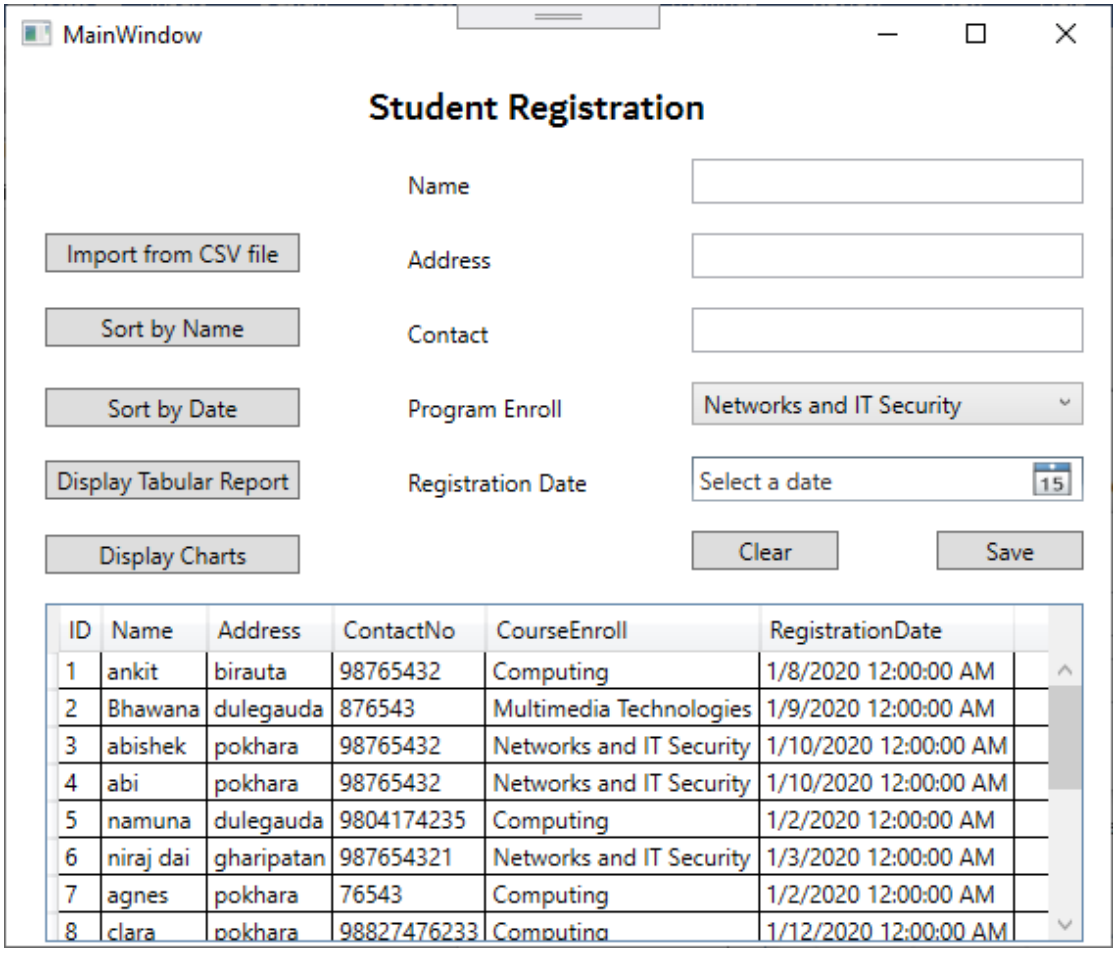


Figure 4: Main Page

After logging into the system, the main window opens which shows the following elements:

- Registration form

  The registration form lets user manually input the student details.

- Datagrid

  The datagrid displays the recorded student details in a tabular form.

- Save

  This button lets the user to save the student details filled using the registration form to and an external xml file.

- Clear

  This button is used to clear the contents of the textboxes.

- Import from CSV file

  This button lets user to import the student details from an external CSV file.

- Sort by Name

  This button is used to sort the names of the students in ascending error (i.e. alphabetically).

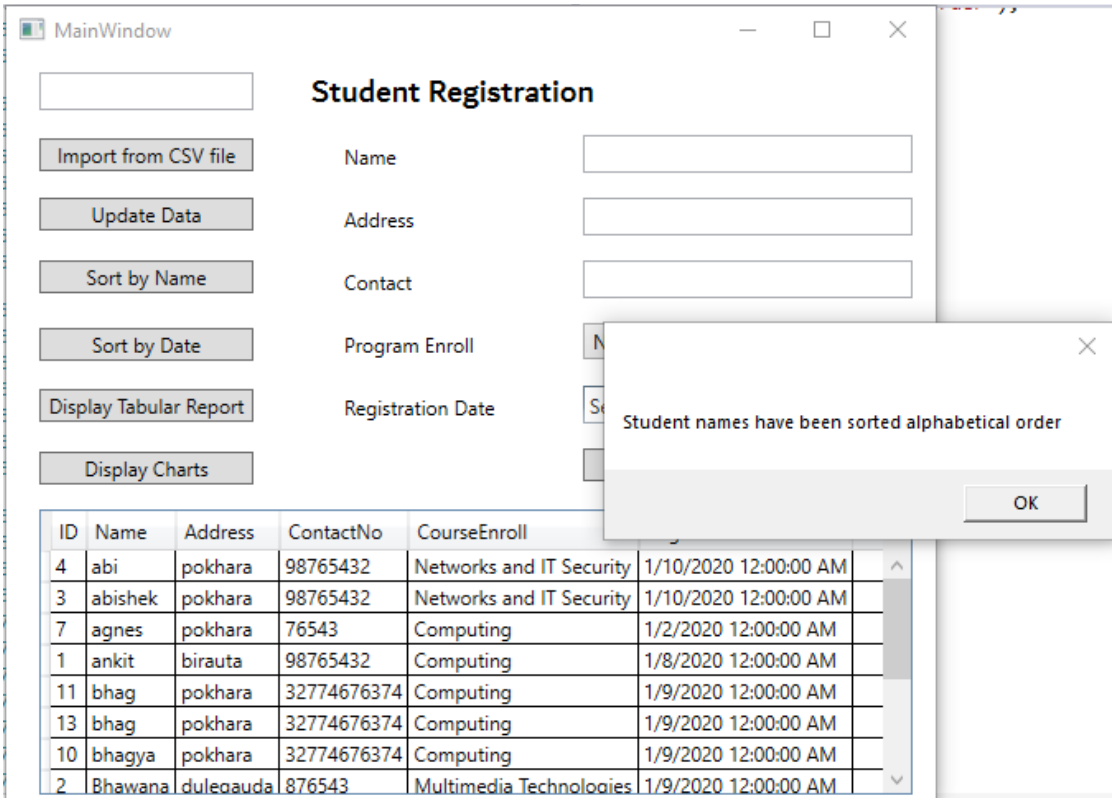


Figure 5: sorting by name

- Sort by Date

  This button is used to sort the student details according to the registration date in ascending error.

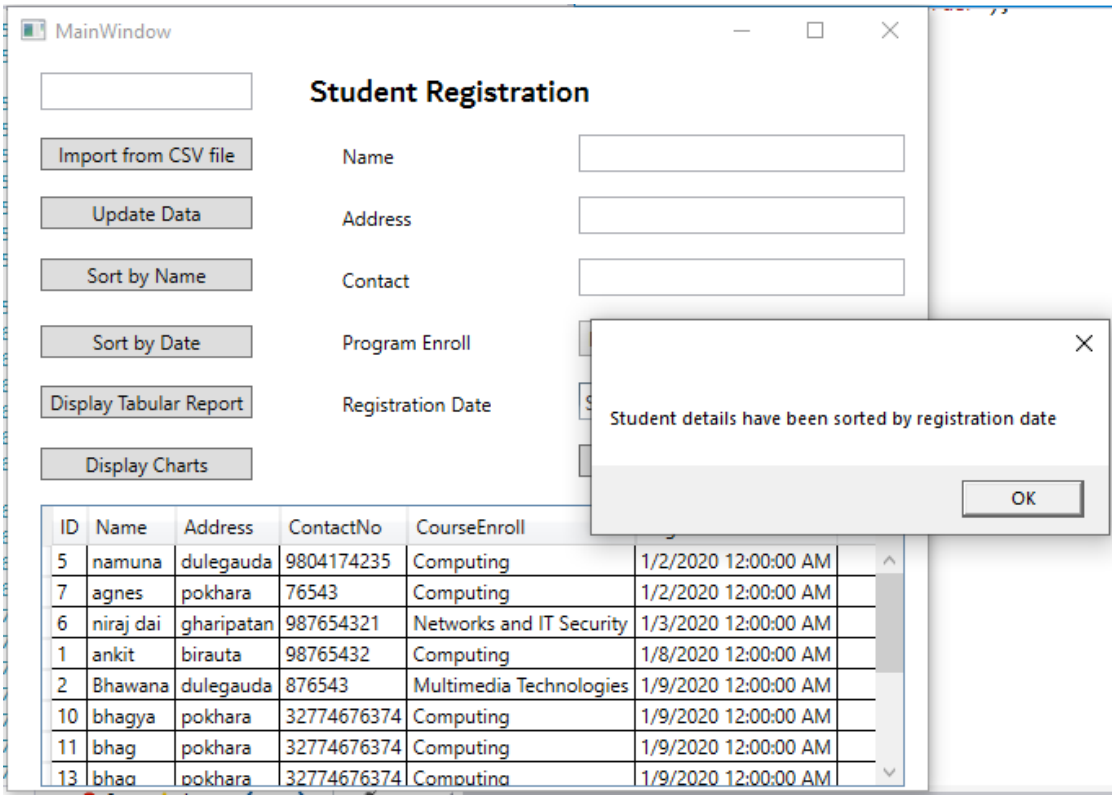


Figure 6: sort by registration date

- Display Tabular Report

  This button lets the user view a tabular report showing total number of students enrolled so far in each program offered by the institution. The courses offered are Computing, Multimedia Technologies, Networks and IT Security.

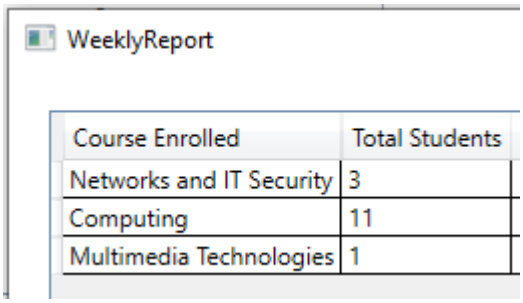


Figure 7: Tabular report

- Display Charts

  This button is used to display the chart (i.e. bar graph) showing total number of students on each program (Computing, Multimedia Technologies, and Networks and IT Security).

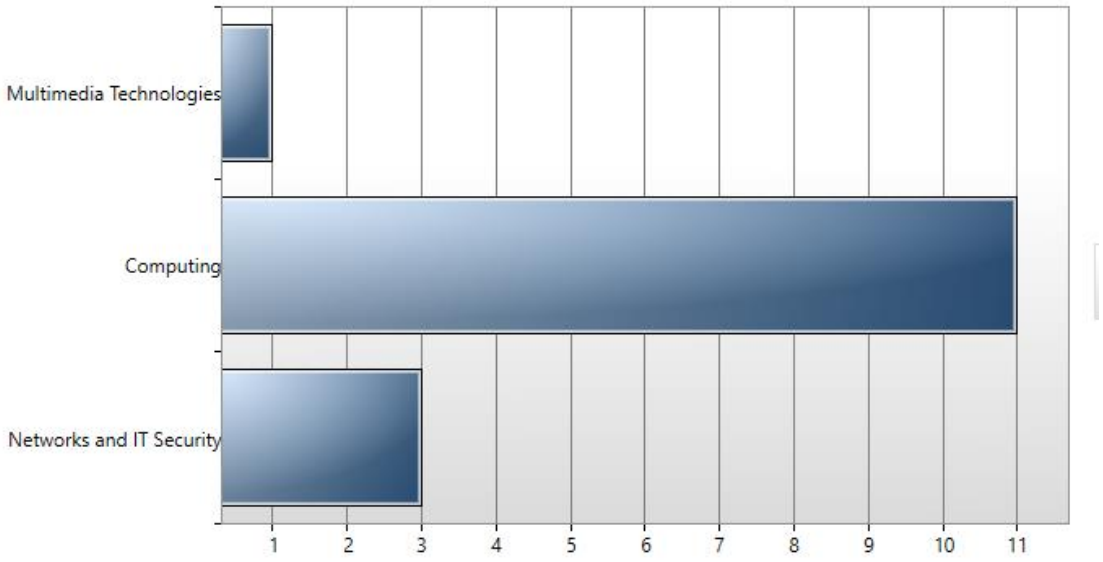


Figure 8: Graphical report

3. System Architecture

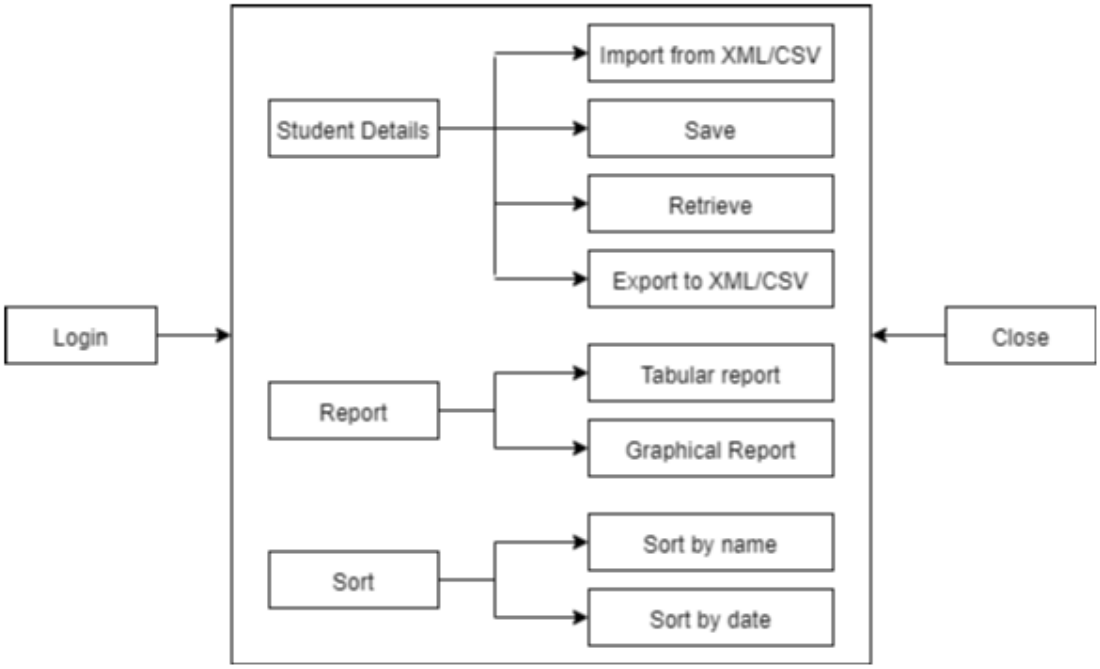


Figure 9: System architecture

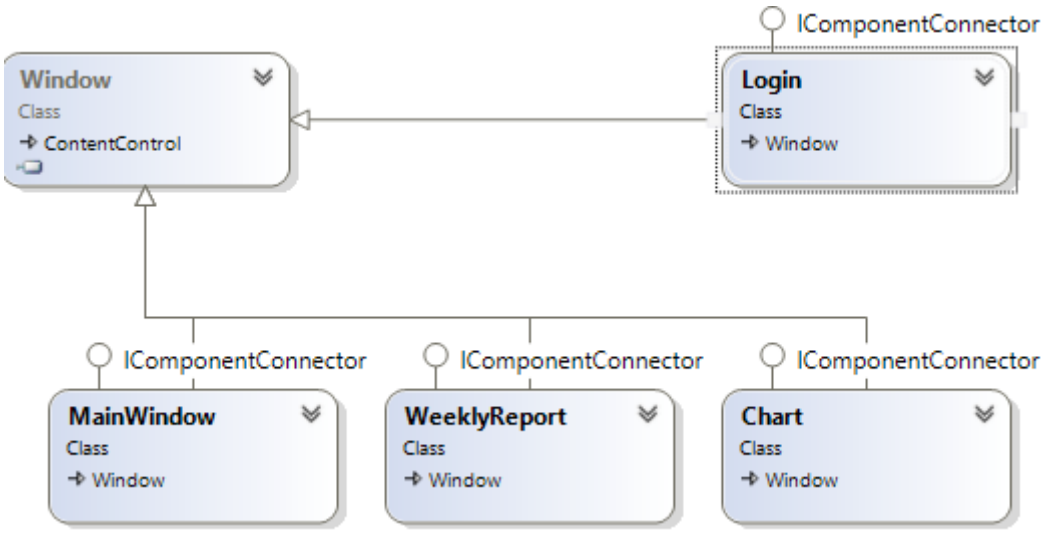


Figure 10: class diagram

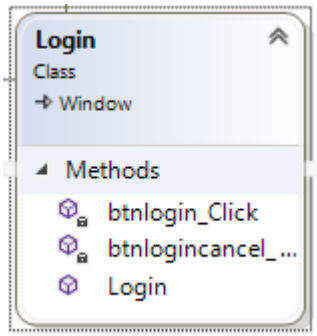


Figure 11: individual diagram of Login class

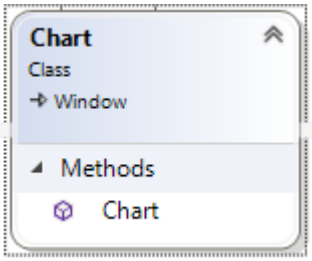


Figure 12: individual diagram of Chart class

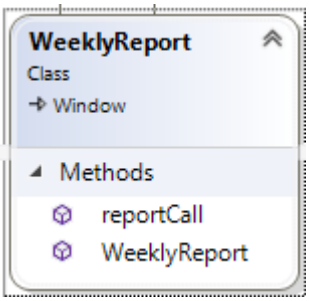


Figure 13: individual diagram of Weekly class

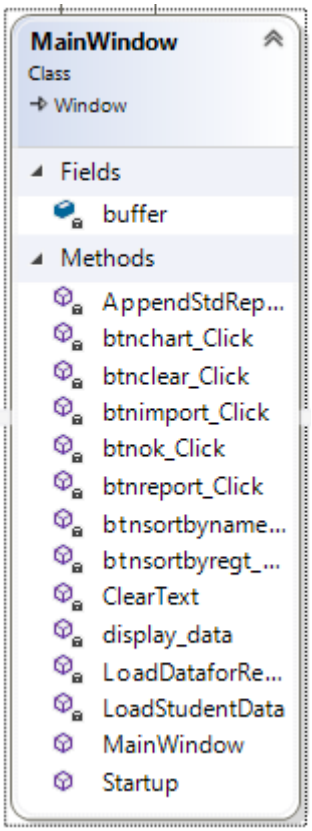


Figure 14: individual diagram of MainWindow class

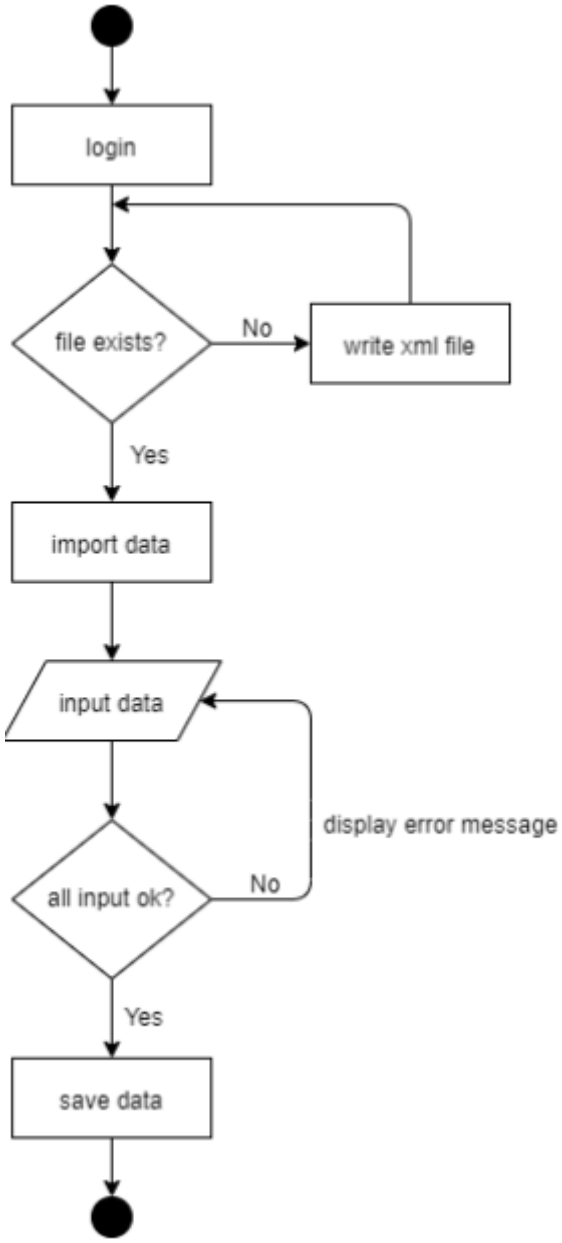


Figure 15: flowchart for student enrolment

4. Bubble Sort Algorithm

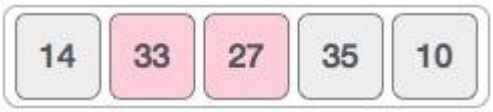
Bubble sort is a simple comparison-based algorithm in which each pair of adjacent elements is compared and the elements are swapped if they are not in order. This algorithm is not suitable for large data sets as its average and worst-case complexity are of $O(n^2)$ where **n** is the number of items.

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

Bubble sort starts with very first two elements, comparing them to check which one is greater.

In this case, value 33 is greater than 14, so it is already in sorted locations. Next, we compare 33 with 27.

We find that 27 is smaller than 33 and these two values must be swapped.



The new array should look like this −

Next we compare 33 and 35. We find that both are in already sorted positions.

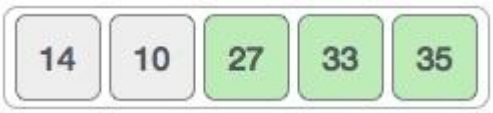
Then we move to the next two values, 35 and 10.

We know then that 10 is smaller 35. Hence they are not sorted.

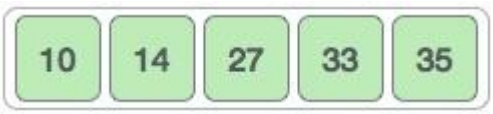
We swap these values. We find that we have reached the end of the array. After one iteration, the array should look like this −

To be precise, we are now showing how an array should look like after each iteration. After the second iteration, it should look like this −

Notice that after each iteration, at least one value moves at the end.



And when there's no swap required, bubble sorts learn that an array is completely sorted (Anon., n.d.).



5. Testing

After the development of the system, testing was done to ensure that there are no bugs or error for the smooth running of the system.

Test Case 1

Objective	To successfully login into the system with correct username and password
Output	Successfully logged into the system

Table 1: testing for login

Figure 16: successful login

Test Case 2

Objective	To check if user can login using incorrect username or password
Output	Unsuccessful login

Table 2: login test with incorrect username and password

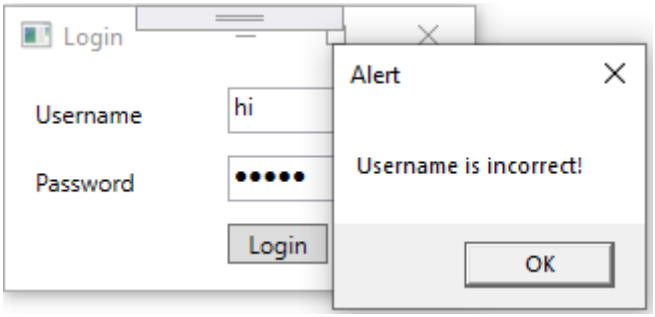


Figure 17:login using incorrect username

Figure 18: login using incorrect password

Test Case 3

Objective	To check if data can be retrieved from external file and displayed in datagrid after successful login
Output	Data successfully retrieved and displayed in datagrid

Table 3: test case for successful retrieval of data

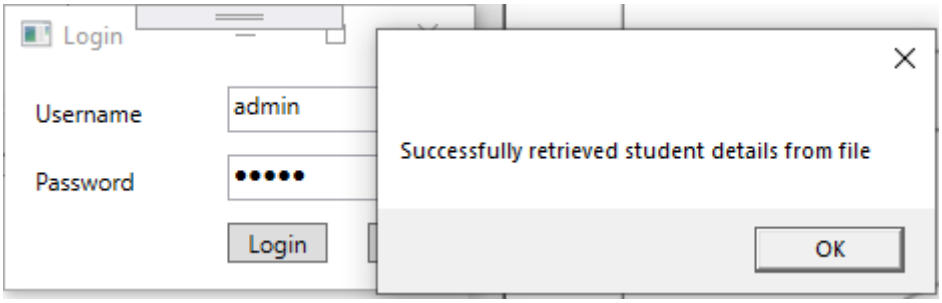


Figure 19: successful retrieval of data

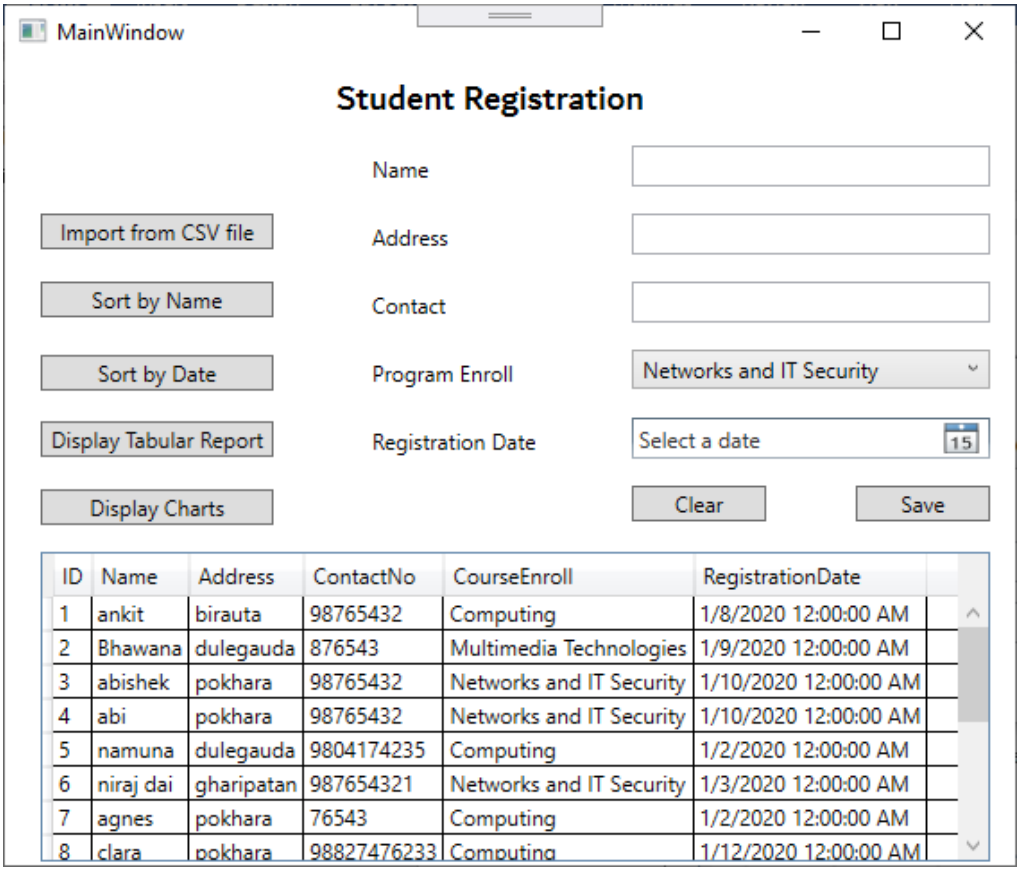


Figure 20: displaying data in datagrid

Test Case 4

Objective	To check if enrolled data report can be displayed in a table
Output	Data successfully displayed using datagrid

Table 4: test case to display tabular report

Course Enrolled	Total Students
Networks and IT Security	3
Computing	3
Multimedia Technologies	1

Figure 21: displaying tabular report

Test Case 5

Objective	To check if enrolled data report can be displayed in a bar graph
Output	Data successfully displayed in bar graph

Table 5: test case to display graphical report

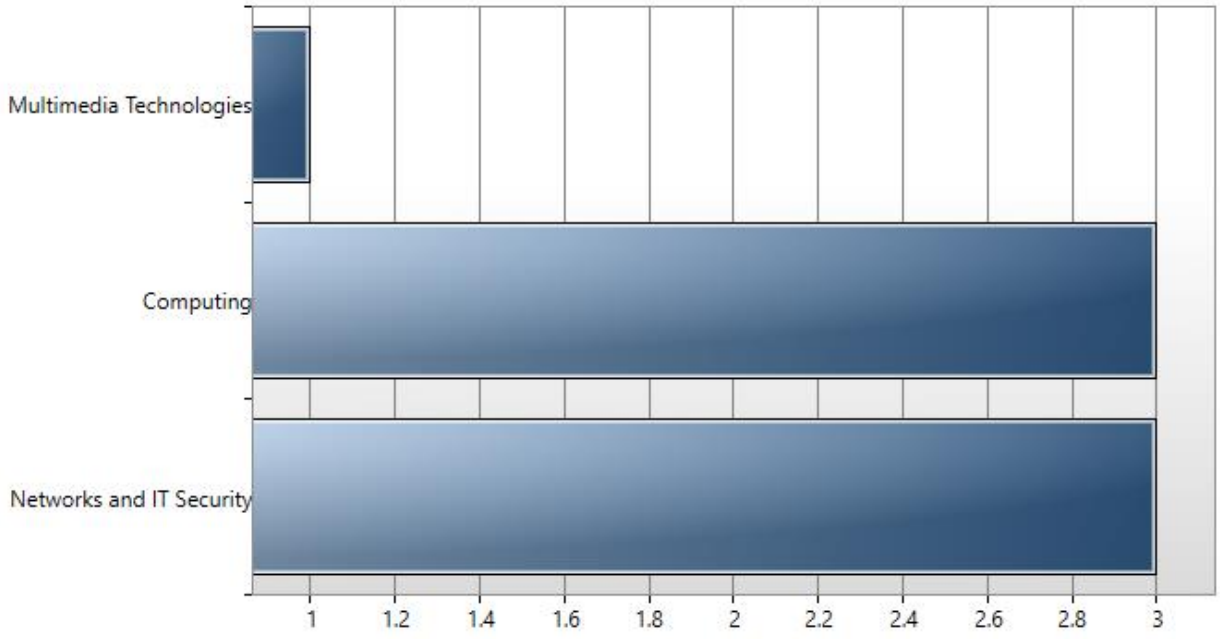


Figure 22: displaying data in a graph

Test Case 6

Objective	To check if data can be manually added to external file and displayed in datagrid
Output	Data successfully added and displayed in datagrid

Table 6: testing if data can be saved using the form

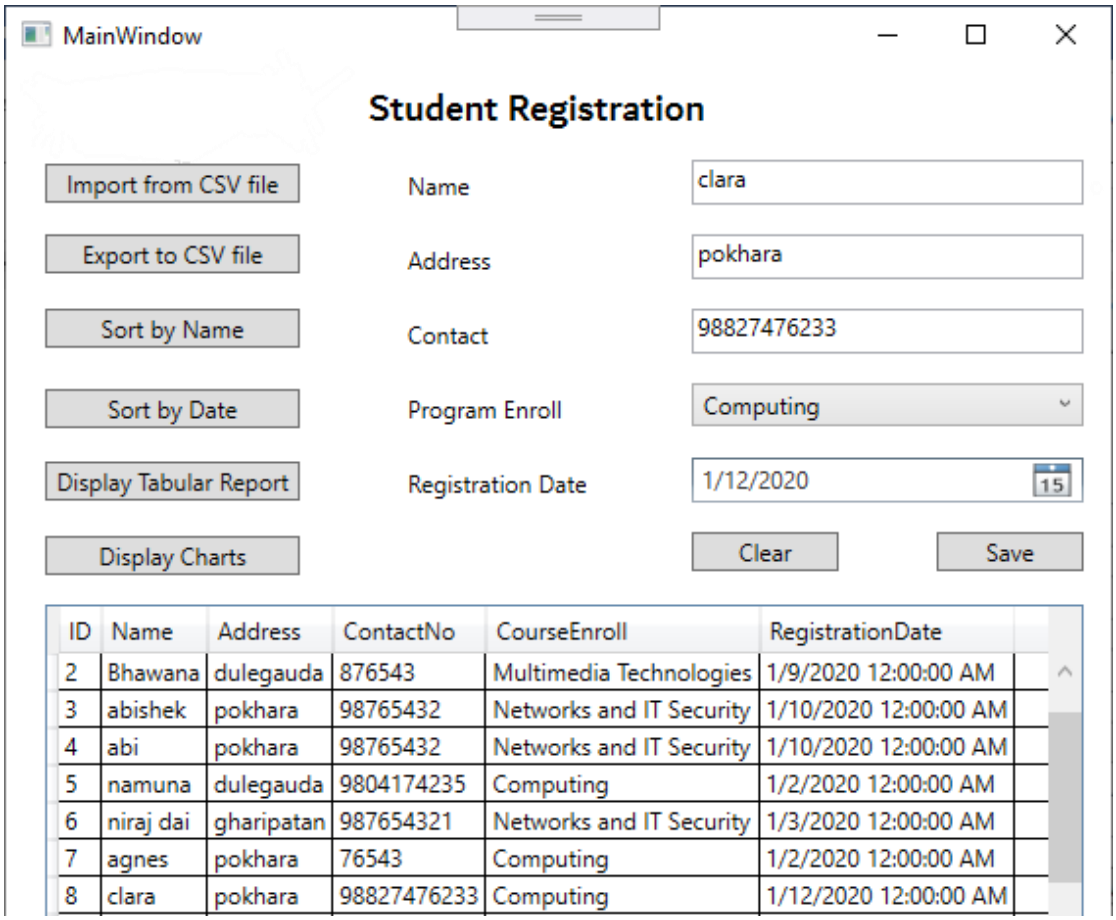


Figure 23: data added using form and displayed in datagrid

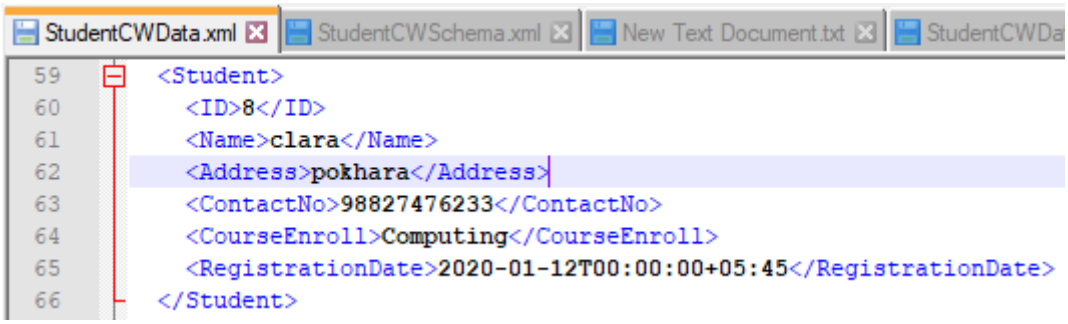


Figure 24: data added to xml file

Test Case 7

Objective	To check if empty values can be passed
Output	Empty values not allowed to passed and a message was displayed

Table 7: test case to pass empty values

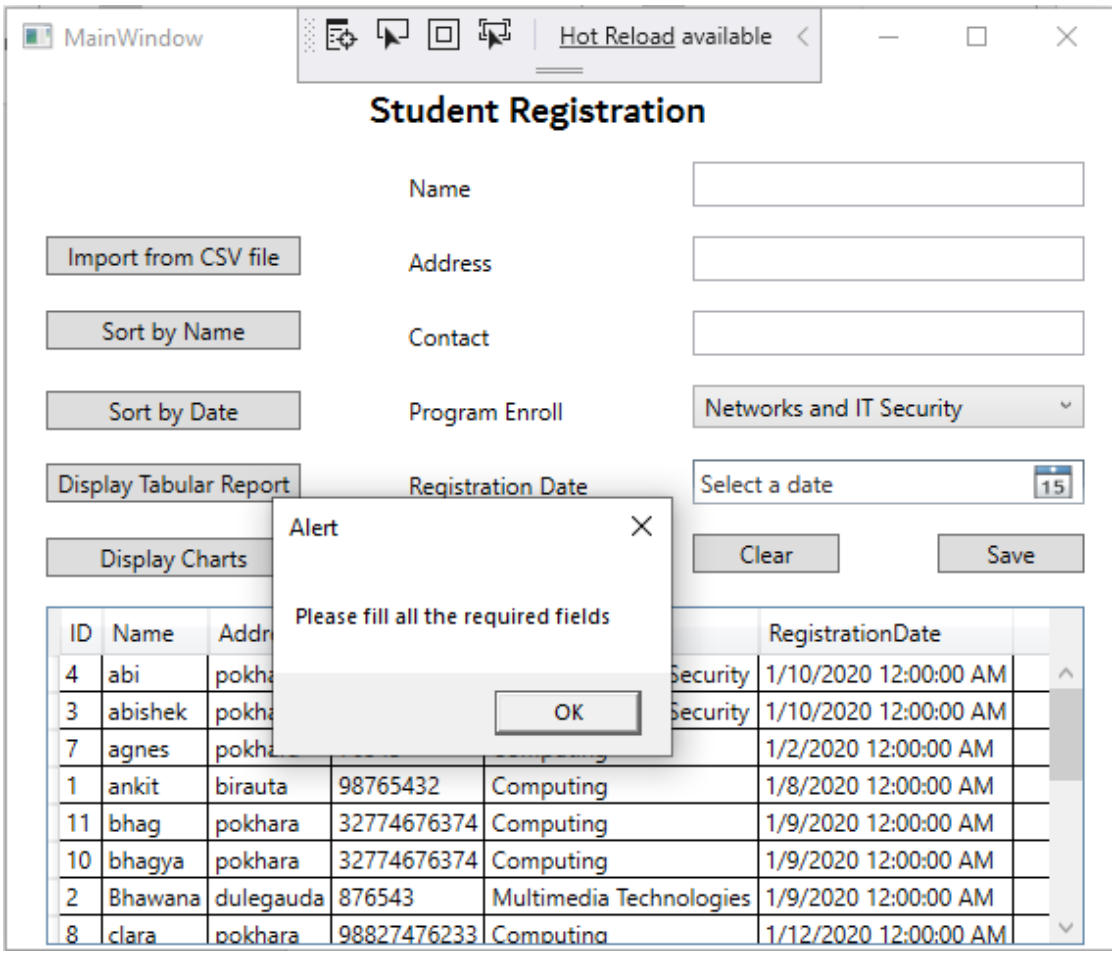


Figure 25: empty values not allowed to pass

Test Case 8

Objective	To check if repetitive ids can be passed
Output	repetitive ids cannot be stored and a message is displayed

Table 8: test case for exception handling in case of repetitive ids

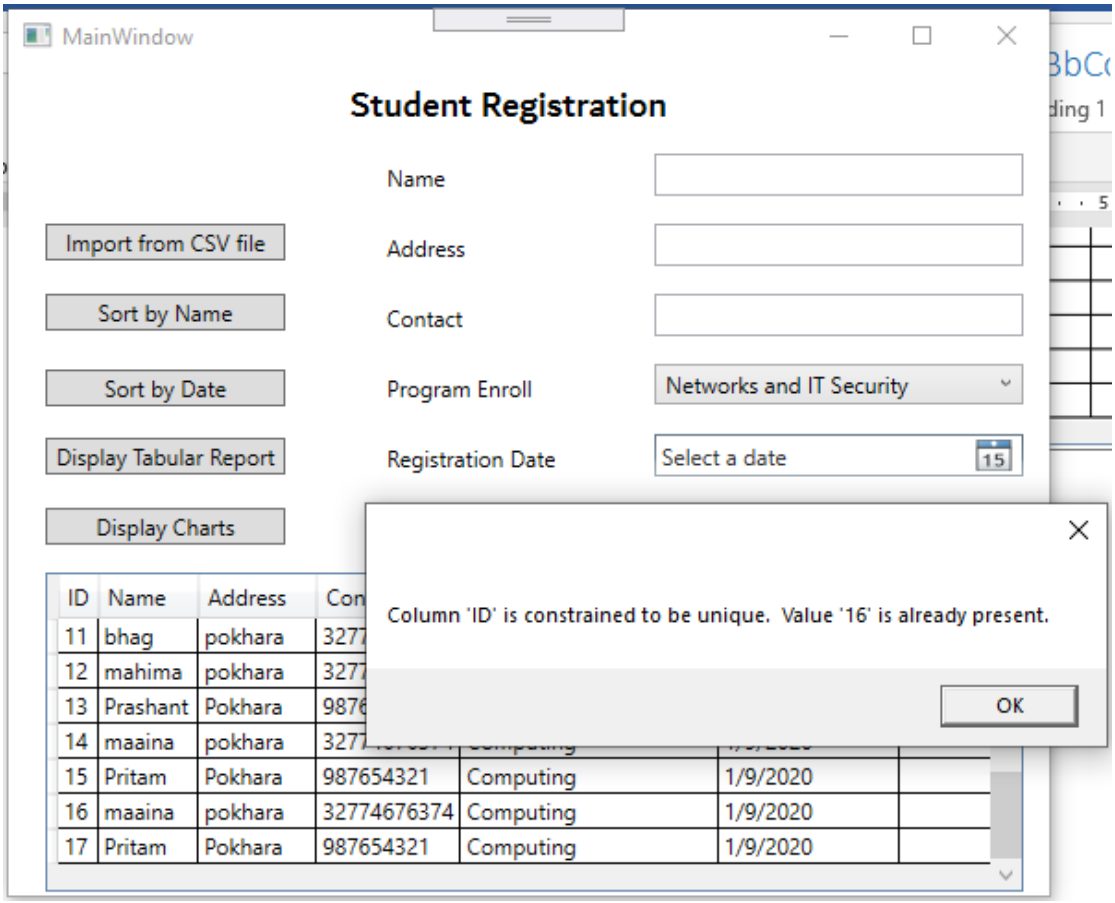


Table 9: exception handling in case of repetitive ids

Test Case 9

Objective	To check if data can be sorted by names in ascending or alphabetical order
Output	Data successfully sorted by names in ascending or alphabetical order

Table 10: test case to sort by name

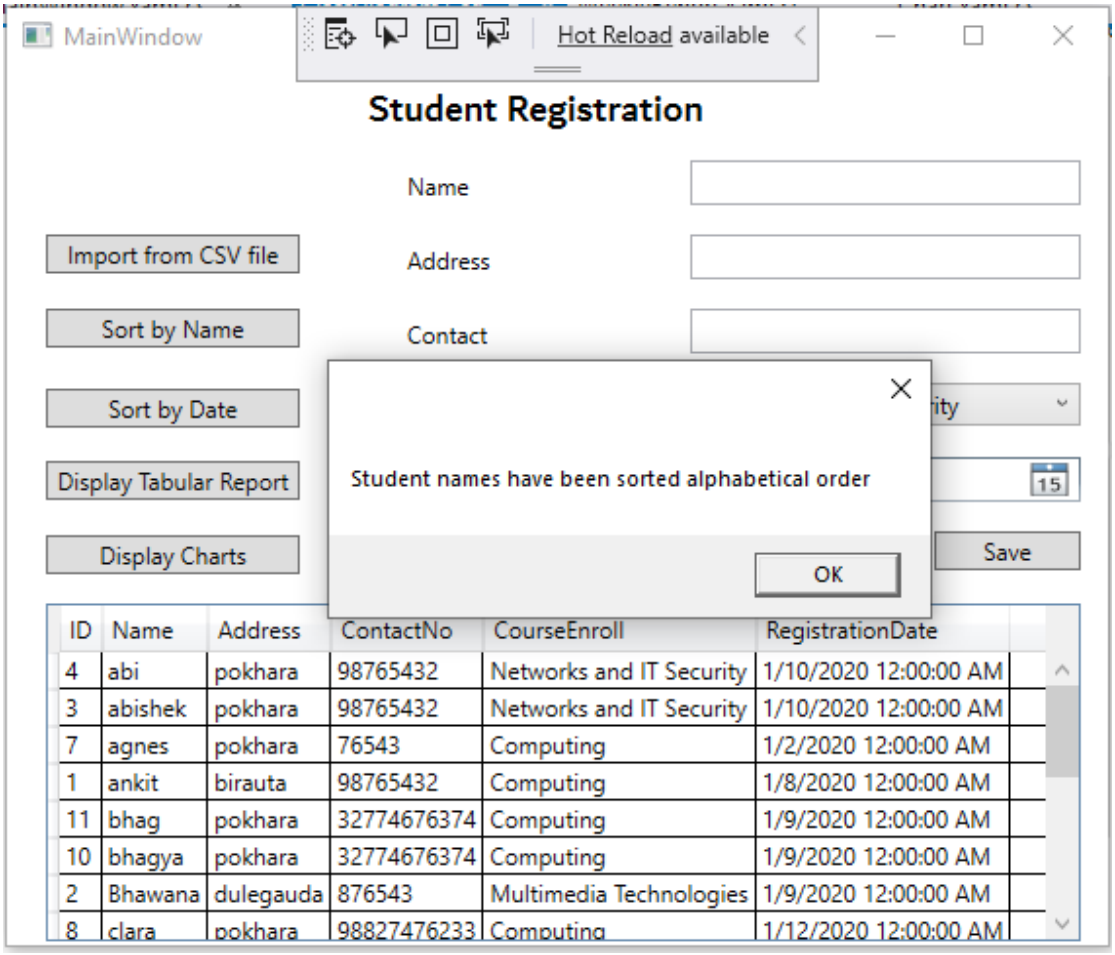


Figure 26: sorting data by name

Test Case 10

Objective	To check if data can be sorted by names in ascending or alphabetical order
Output	Data successfully sorted by names in ascending or alphabetical order

Table 11: test case to sort data by date

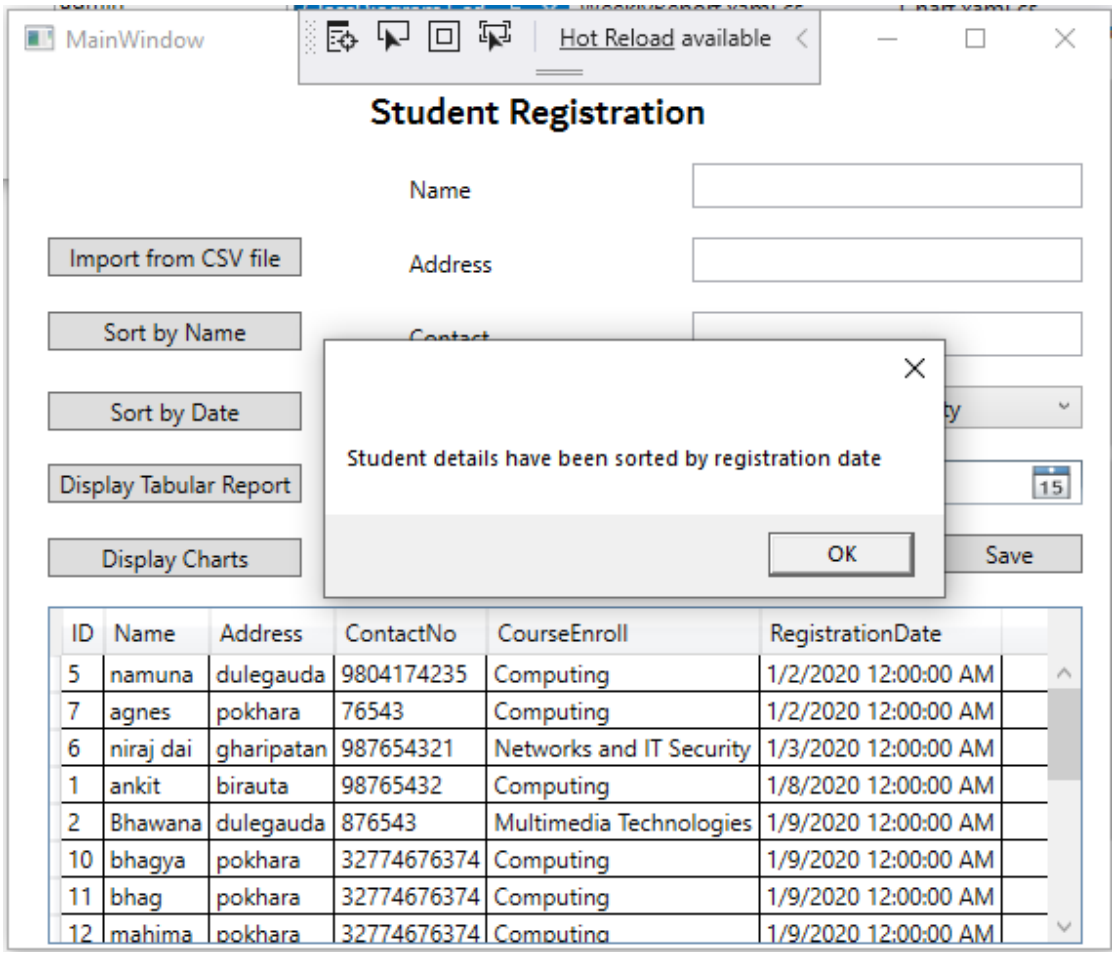


Figure 27: sorting by date

Test Case 11

Objective	To check if data can be sorted by names in ascending or alphabetical order
Output	Data successfully sorted by names in ascending or alphabetical order

Table 12: test case to import data from csv file

ID	Name	Address	ContactNo	CourseEnroll	RegistrationDate
16	maaina	pokhara	32774676374	Computing	1/9/2020 0:00
17	Pritam	Pokhara	987654321	Computing	1/9/2020 0:00

Figure 28: data in csv file

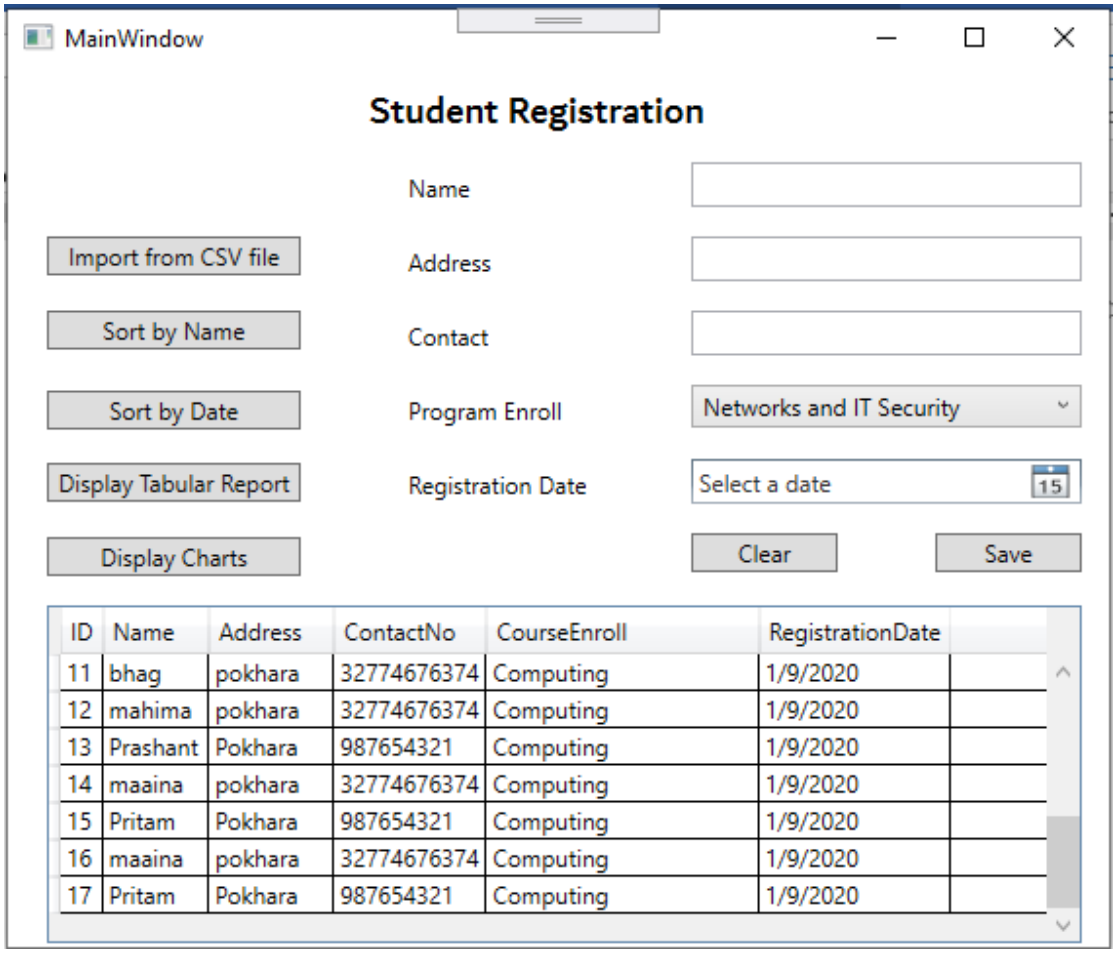


Figure 29: data retrieved in datagrid

6. Reflection and conclusion

Developing a student information system in Microsoft Visual Studio 2019 with C# as primary programming language was an entirely new experience. But having previous knowledge about other programming languages made it a bit easier for us to understand the programming language. Developing the student information system for this project was indeed a difficult task. Serialization, developing graphs, importing and exporting of csv file was a new thing for me.

The main feature that I like about this system was creating reports and charts as it can make data analysing an easier task and people can easily use the data for future references as well. While developing this system, I had to face a lot of errors and exceptions for which I had to go through a lot of articles and constant supervision from the teachers in order to overcome the errors. This system is hands-down a practical approach to the traditional way of record-keeping and can be useful for any company who want to digitalise their way of record-keeping for an easier experience.

7. References

References

Anon., n.d. *tutorialspoint.* [Online]
Available at:
https://www.tutorialspoint.com/data_structures_algorithms/bubble_sort_algorithm.htm
[Accessed 04 01 2020].

8. Appendix

Login class

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

        private void btnlogin_Click(object sender, RoutedEventArgs e)
        {
            if (txtuname.Text != "admin")
            {
                MessageBox.Show("Username is incorrect!", "Alert",
MessageBoxButton.OK);
                txtuname.Clear();

            }
            else if (txtpw.Password != "admin")
            {
                MessageBox.Show("Password is incorrect!", "Alert",
MessageBoxButton.OK);
                txtpw.Clear();
            }
            else
            {
                MessageBox.Show("Logged in Successfully !!!", "Success",
MessageBoxButton.OK);
                MainWindow mainWindow = new MainWindow();
                mainWindow.Show();
            }
        }

        private void btnlogincancel_Click(object sender, RoutedEventArgs e)
        {
            Close();
        }
    }
}
```

MainWindow class

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

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

        public void Startup()
        {
            LoadStudentData();
        }

        //method to add student details to xml file
        private void AppendStdReport(DataSet dataSet)
        {
            dataSet = new DataSet();
            var handler = new Handler();
            dataSet.ReadXmlSchema(@"C:\\StudentCWSchema.xml");
            dataSet.ReadXml(@"C:\StudentCWData.xml");

            var studentTable = dataSet.Tables["Student"];

            var newRow = studentTable.NewRow();
            newRow["Name"] = txtname.Text;
            newRow["Address"] = txtaddress.Text;
            newRow["ContactNo"] = txtcontact.Text;
            newRow["CourseEnroll"] = txtprogram.Text;
            newRow["RegistrationDate"] = txtreg.SelectedDate != null ?
txtreg.SelectedDate.Value : DateTime.Today;
            studentTable.Rows.Add(newRow);
```

```
            if (txtname.Text != "" & txtaddress.Text != "" & txtcontact.Text !=
"" & txtreg.Text != "")
            {
                dataSet.WriteXml(@"C:\StudentCWData.xml");
            }
            else
            {
                MessageBox.Show("Please fill all the required fields", "Alert",
MessageBoxButton.OK);
            }
        }

        //method to retrieve student details from xml file
        private void LoadStudentData()
        {
            if (System.IO.File.Exists(@"C:\StudentCWData.xml"))
            {
                var dataSet = new DataSet();
                dataSet.ReadXmlSchema("C:\\StudentCWSchema.xml");
                dataSet.ReadXml(@"C:\StudentCWData.xml");

                if (dataSet.Tables.Contains("Student"))
                {
                    var dtStdReport = dataSet.Tables["Student"];
                    grid.ItemsSource = dtStdReport.DefaultView;
                }
                MessageBox.Show("Successfully retrieved student details from
file");
            }
            else
                MessageBox.Show("Data Retrieval Unsucessful");
        }

        //method to load data for weekly report
        private void LoadDataforReport()
        {
            {
                var handler = new Handler();
                var dataSet = new DataSet();
                dataSet.ReadXml(@"C:\StudentCWData.Xml");

                for (int i = 0; i <= dataSet.Tables["Student"].Rows.Count - 1;
i++)
                {
                    if
(dataSet.Tables["Student"].Rows[i]["CourseEnroll"].ToString() == "i")
                    {
                        var dr2 = dataSet.Tables["Student"].NewRow();
                        dr2["Name"] = txtname.Text;
                        dr2["Address"] = txtaddress.Text;
                        dr2["ContactNo"] = txtcontact.Text;
                        dr2["CourseEnroll"] = txtprogram.Text;
                        dr2["RegistrationDate"] = txtreg.SelectedDate != null ?
txtreg.SelectedDate.Value : DateTime.Today;
                        dataSet.Tables["Student"].Rows.Add(dr2);


dataSet.Tables["Student"].WriteXml(@"C:\StudentCWData.xml");
                    }
                }
            }
        }
```

```
        private void btnok_Click(object sender, RoutedEventArgs e)
        {
            DataSet dataSet = new DataSet();
            AppendStdReport(dataSet);
            LoadStudentData();
        }

        private void btnclear_Click(object sender, RoutedEventArgs e)
        {
            ClearText();
        }

        //method to clear text from textbox
        private void ClearText()
        {
            txtname.Text = "";
            txtaddress.Text = "";
            txtcontact.Text = "";
            txtreg.Text = "";
        }

        //sort by name
        private void btnsortbyname_Click(object sender, RoutedEventArgs e)
        {
            var dataSet = new DataSet();
            dataSet.ReadXmlSchema("C:\\StudentCWSchema.xml");
            dataSet.ReadXml(@"C:\StudentCWData.xml");

            if (dataSet.Tables.Contains("Student"))
            {
                var dtStdReport = dataSet.Tables["Student"];
                dtStdReport.DefaultView.Sort = "Name ASC";
                grid.ItemsSource = dtStdReport.DefaultView;
            }
            MessageBox.Show("Student names have been sorted alphabetical
order");
        }

        private void btnreport_Click(object sender, RoutedEventArgs e)
        {
            LoadDataforReport();
            WeeklyReport weeklyReport = new WeeklyReport();
            weeklyReport.Show();
        }

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

        //sort by date
        private void btnsortbyregt_Click(object sender, RoutedEventArgs e)
        {
            var dataSet = new DataSet();
            dataSet.ReadXmlSchema("C:\\StudentCWSchema.xml");
            dataSet.ReadXml(@"C:\StudentCWData.xml");

            if (dataSet.Tables.Contains("Student"))
            {
                var dtStdReport = dataSet.Tables["Student"];
```

```
                dtStdReport.DefaultView.Sort = "RegistrationDate ASC";
                grid.ItemsSource = dtStdReport.DefaultView;
            }
            MessageBox.Show("Student details have been sorted by registration
date");
        }

        //import csv file
        private void btnimport_Click(object sender, RoutedEventArgs e)
        {
            try
            {
                var dataSet = new DataSet();
                dataSet.ReadXmlSchema("C:\\StudentCWSchema.xml");
                dataSet.ReadXml(@"C:\StudentCWData.xml");
                Microsoft.Win32.OpenFileDialog openFileDlg = new
Microsoft.Win32.OpenFileDialog();

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

                            dataSet.WriteXml(@"C:\StudentCWData.xml");
                        }
                    }
                    display_data();
                    MessageBox.Show("Student details successfully imported",
"Success!", MessageBoxButton.OK);
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show(ex.Message);
            }
        }

        private void display_data()
        {
            string dataXMLFile = @"C:\StudentCWData.xml";
            DataSet dataset = new DataSet();
            dataset.ReadXml(dataXMLFile);

            buffer = new DataTable("dt");
            buffer.Columns.Add("ID", typeof(String));
            buffer.Columns.Add("Name", typeof(String));
            buffer.Columns.Add("Address", typeof(String));
            buffer.Columns.Add("ContactNo", typeof(String));
```

```
            buffer.Columns.Add("CourseEnroll", typeof(String));
            buffer.Columns.Add("RegistrationDate", typeof(String));

            for (int i = 0; i < dataset.Tables[0].Rows.Count; i++)
            {
                string s = dataset.Tables[0].Rows[i][5].ToString();
                DateTime dtime = DateTime.Parse(s);
                try
                {
                    buffer.Rows.Add(
                        dataset.Tables[0].Rows[i][0].ToString(),
                        dataset.Tables[0].Rows[i][1].ToString(),
                        dataset.Tables[0].Rows[i][2].ToString(),
                        dataset.Tables[0].Rows[i][3].ToString(),
                        dataset.Tables[0].Rows[i][4].ToString(),
                        dtime.ToShortDateString());
                }
                catch (Exception e){
                    MessageBox.Show(e.Message);
                }
            }
            DataView dataView = new DataView(buffer);
            grid.ItemsSource = dataView;
        }
    }
}
```

Chart class

```
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 coursewoek
{
    /// <summary>
    /// Interaction logic for Chart.xaml
    /// </summary>
    public partial class Chart : Window
    {
        public Chart()
        {
            InitializeComponent();
            var dataset = new DataSet();
            dataset.ReadXml(@"C:\StudentCWData.xml");
            DataTable stdReport = dataset.Tables[0];
            int total_Comp = 0;
            int total_MM = 0;
            int total_Net = 0;

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

```
            dt.Columns.Add("Course Enrolled", typeof(String));
            dt.Columns.Add("Total Students", typeof(int));

            for (int i = 0; i < stdReport.Rows.Count; i++)
            {
                String col = stdReport.Rows[i]["CourseEnroll"].ToString();
                if (col == "Networks and IT Security")
                {
                    total_Net++;
                }
                else if (col == "Computing")
                {
                    total_Comp++;
                }
                else if (col == "Multimedia Technologies")
                {
                    total_MM++;
                }
            }

            dt.Rows.Add("Networks and IT Security", total_Net);
            dt.Rows.Add("Computing", total_Comp);
            dt.Rows.Add("Multimedia Technologies", total_MM);

            ((BarSeries)chartGrid).ItemsSource =
        new KeyValuePair<string, int>[]{
        new KeyValuePair<string,int>("Networks and IT Security", total_Net),
        new KeyValuePair<string,int>("Computing", total_Comp),
        new KeyValuePair<string,int>("Multimedia Technologies", total_MM) };
        }
    }
}
```

WeeklyReport class

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

```
            var dataset = new DataSet();
            dataset.ReadXml(@"C:\StudentCWData.xml");
            DataTable stdReport = dataset.Tables[0];
            int total_Comp = 0;
            int total_MM = 0;
            int total_Net = 0;

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

            for (int i = 0; i < stdReport.Rows.Count; i++)
            {
                String col = stdReport.Rows[i]["CourseEnroll"].ToString();
                if (col == "Networks and IT Security")
                {
                    total_Net++;
                }
                else if (col == "Computing")
                {
                    total_Comp++;
                }
                else if (col == "Multimedia Technologies")
                {
                    total_MM++;
                }
            }

            dt.Rows.Add("Networks and IT Security", total_Net);
            dt.Rows.Add("Computing", total_Comp);
            dt.Rows.Add("Multimedia Technologies", total_MM);

            ((BarSeries)chartGrid).ItemsSource =
        new KeyValuePair<string, int>[]{
        new KeyValuePair<string,int>("Networks and IT Security", total_Net),
        new KeyValuePair<string,int>("Computing", total_Comp),
        new KeyValuePair<string,int>("Multimedia Technologies", total_MM) };
        }
    }
}
```

Handler class

```
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("CourseEnroll", typeof(string));
            dt.Columns.Add("RegistrationDate", typeof(DateTime));
            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.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("Name", typeof(string));
            dt.Columns.Add("Address", typeof(string));
            dt.Columns.Add("ContactNo", typeof(string));
            dt.Columns.Add("CourseEnroll", typeof(string));
            dt.Columns.Add("RegistrationDate", typeof(DateTime));
            return dt;
        }
    }
}
```