Marking Scheme

Informatics College Pokhara

informatics
college pokhara

Application Development
CS6004NI
Course Work 1

Submitted By: **Niruta Devkota**
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	missing most of the validation
Enrollment Report & weekly report in tabular format	very poorly executed reports and data not shown accurately
Course wise enrollment report & Chart display	Very poorly designed and only contains one report format with in appropriate data
Algorithm used for sorting & proper sorting of data	Default sorting provided by .net is used
B. Documentation	
User Manual for running the application	User Manual is below average. Is textual only.

Application architecture & description of the classes ad methods sued	average work with very limited explanation of the classes and methods used
Flow chart, algoriathms and data sctructures used	average work with very limited explanation and missing diagramatic representation.
Reflective essay	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:	**D+**	**D+**

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: Niruta Devkota

College ID: NP04CP4A170022

University ID: 17030729

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 marks of zero will be awarded.

Table of Contents

List of Figures

1. Introduction

This is an individual coursework for the module “Application Development” which require us to design and implement Student Information System in C# for a company. The system is developed using Visual Studio Platform. The application allows user to input the student personal details including registration date do the system can generate a weekly enrolment report of the student. Student details contain details like Name, address, contact no, email, program enrol and registration date. The application keeps track of student’s details program enrol and registration date.

1.1. Current Scenario

There are numerous Institutions who keep record of their data in old traditional system which is Paper based. In addition to that, there are some institutions with digital system but are lacking the features which are needed for an Institution.

1.2. Proposed System

The proposed system is digitalized system which is specially designed to overcome problem mentioned above. The system ensures security with the presence of login section. Data entry and display has been made easy with user-friendly interface.

2. User Manual

These are the screenshots which will illustrate a user how to operate the system.

When the user operates the system the initial screen will be the login screen. The username and password of the system is “admin”. Only valid username and password can provide access to the system.

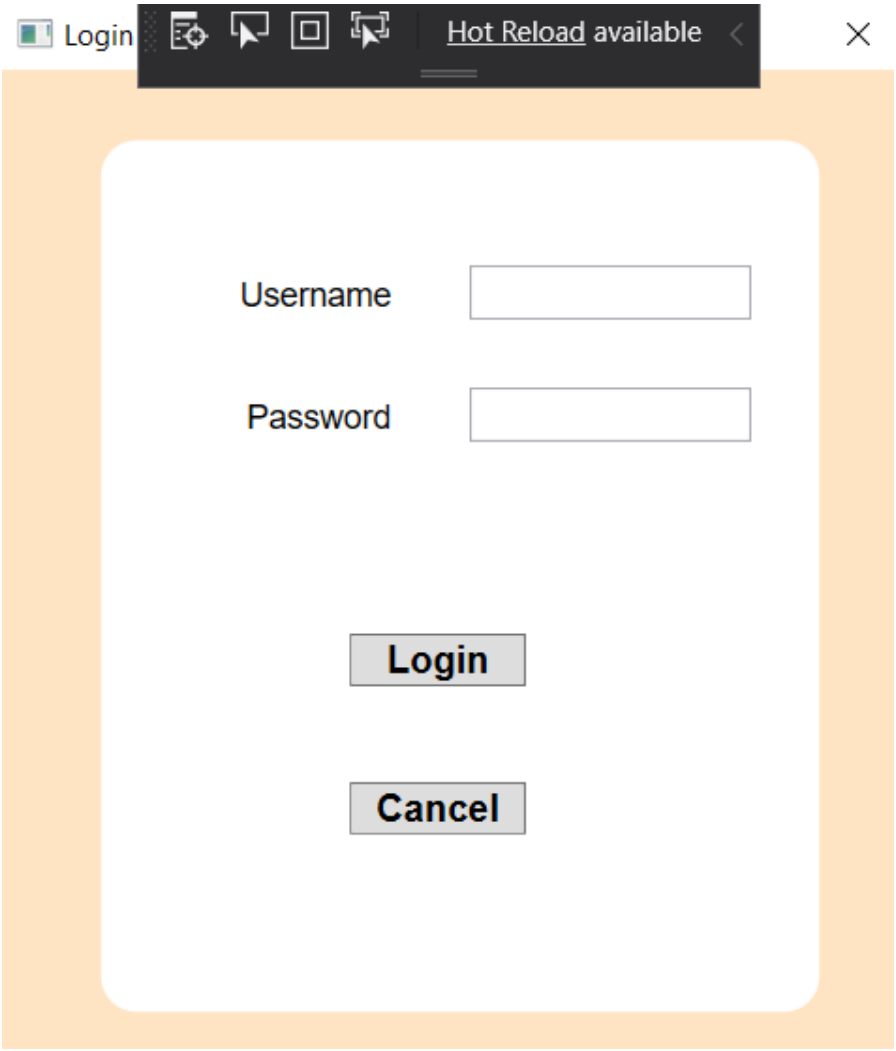

Figure 1: Login Page

After successful login the Student Form Window opens.

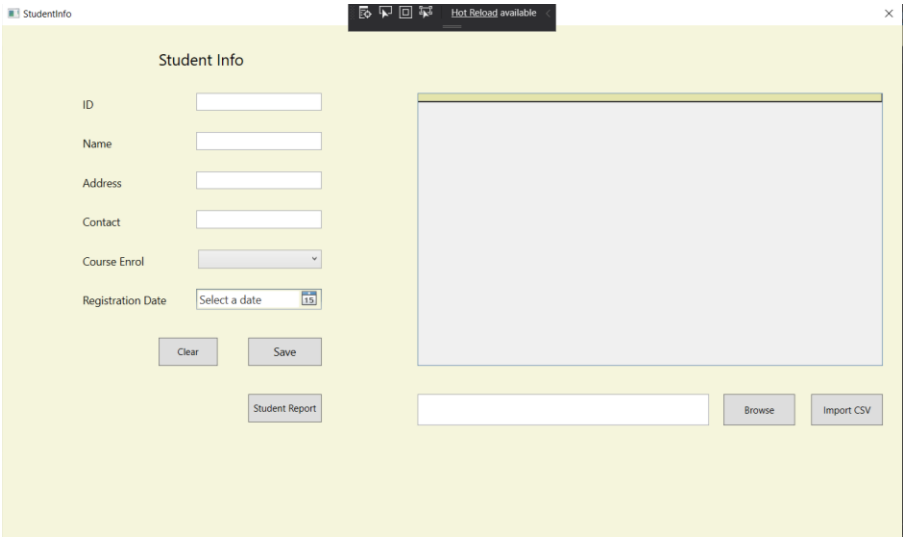

Figure 2: Student Form

In this window user can fill student details and save the information, browse and load csv file and go to student report section.

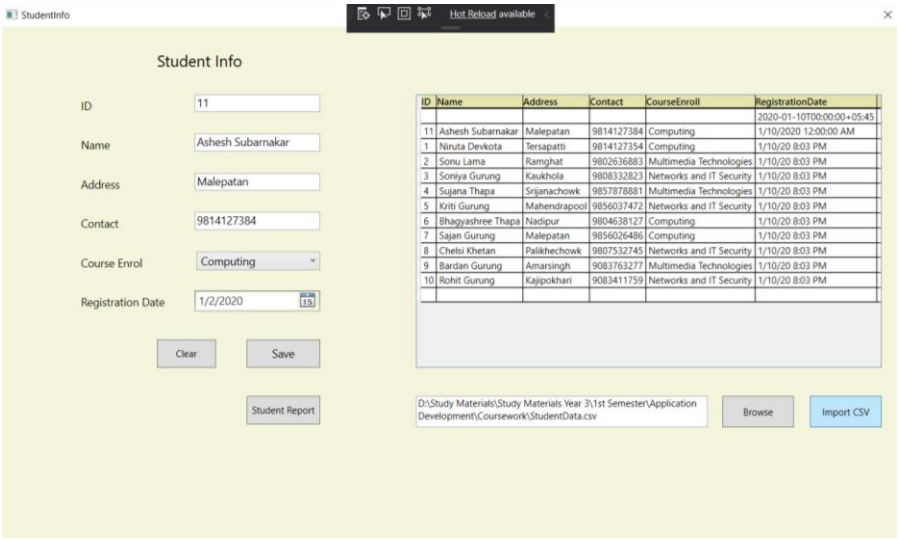

Figure 3: Importing CSV

The Import CSV button imports the manually written student information.

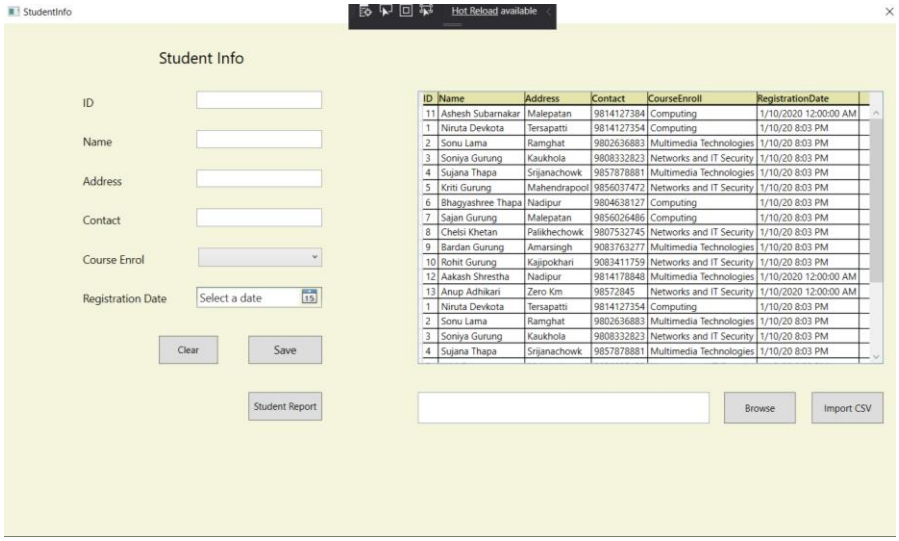

Figure 4: Student Form

The student form window shows the pre-registered student information in the data grid.

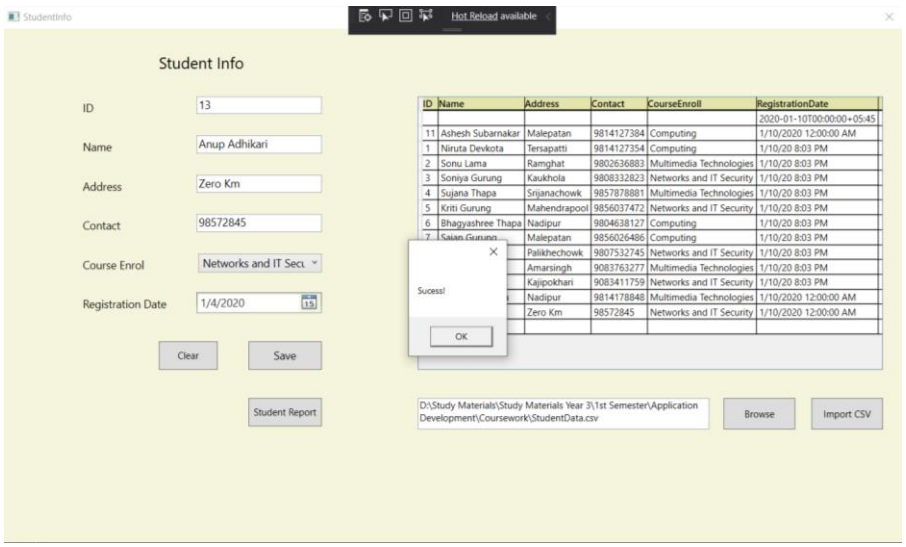

Figure 5: Adding Student

Student information can only be added if all the fields are filled.

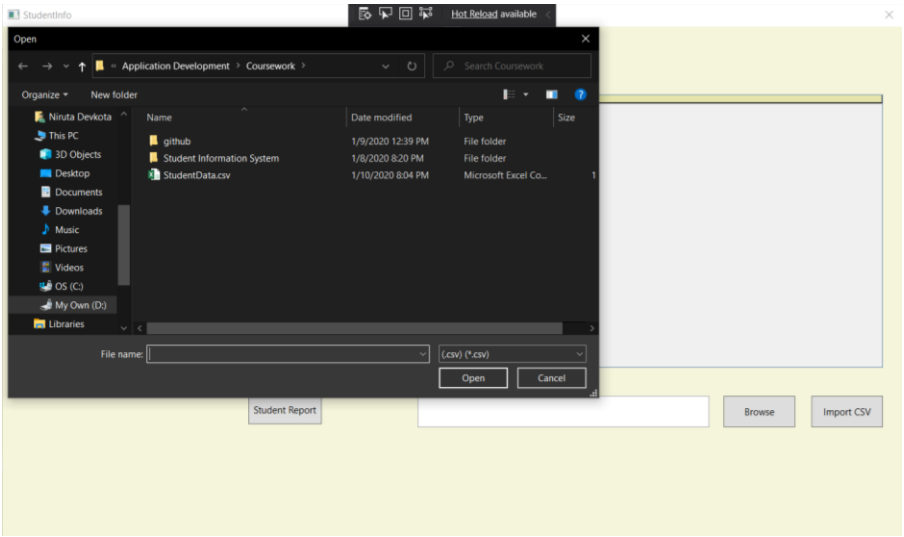

Figure 6: Browsing CSV file

The CSV file can be browsed by clicking the browse button.

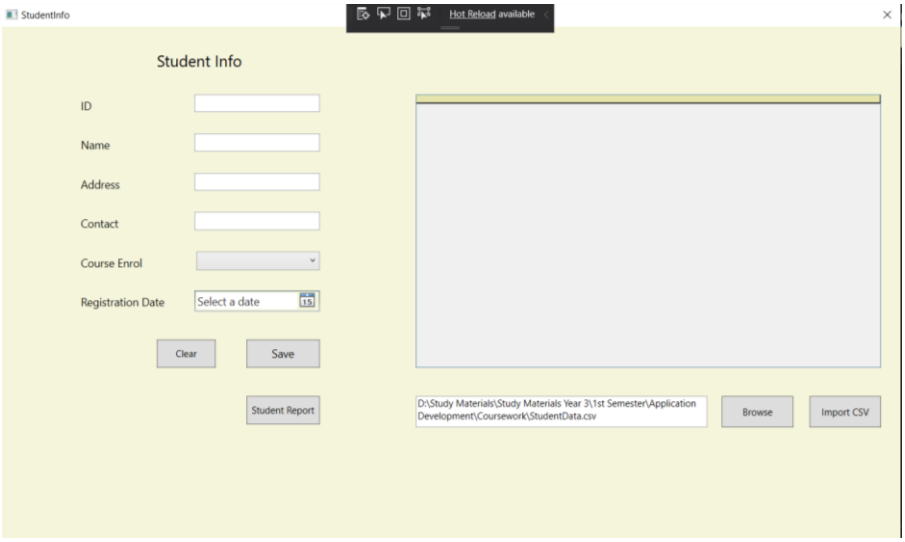

Figure 7: Showing Browsed CSV file location

This figure shows the browsed CSV location in the text filed.

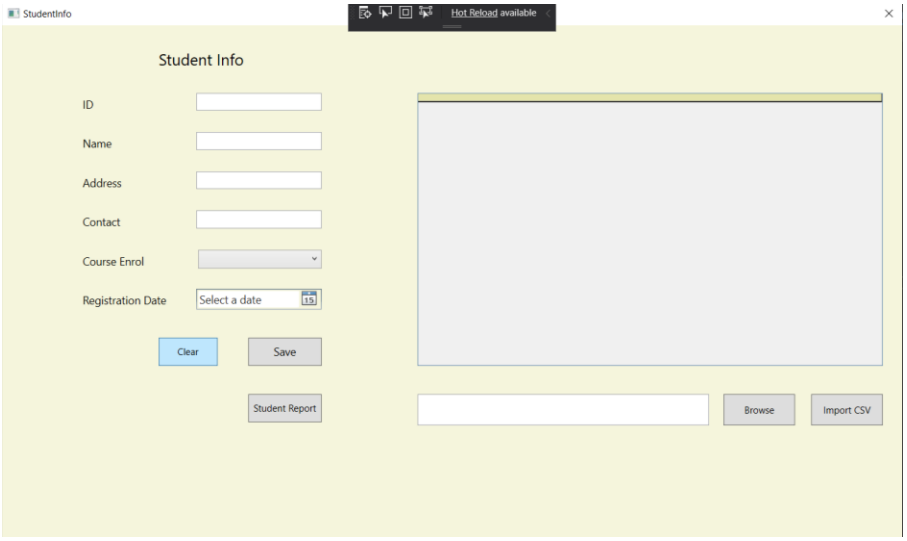

Figure 8: Clear Button implementation

The clear button clears the browsed csv file location and data grid.

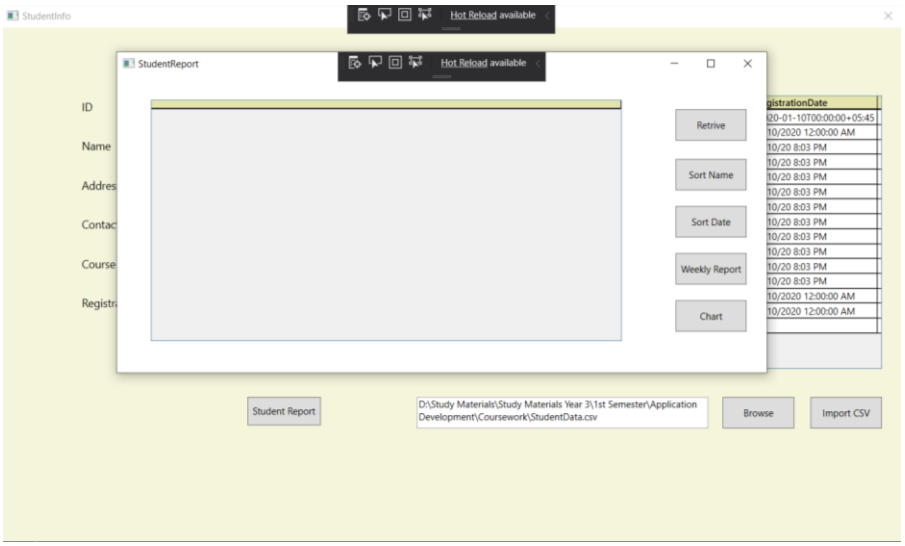

Figure 9: Student Detail Screen

This is the window that is opened after clicking Student Report button. This consists of retrieve button to retrieve student details, sorting buttons, weekly report button and chart button.

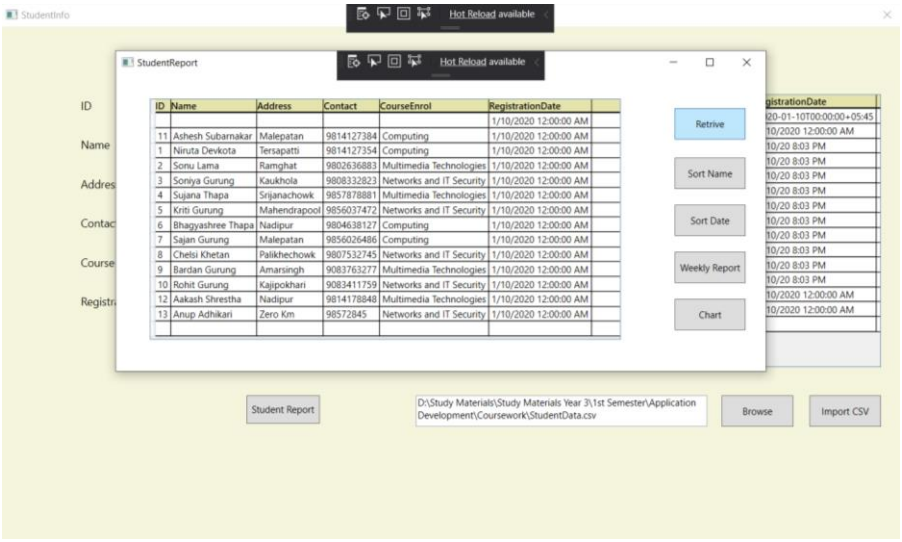

Figure 10: Unsorted data

The figure shows the unsorted retrieved list of students.

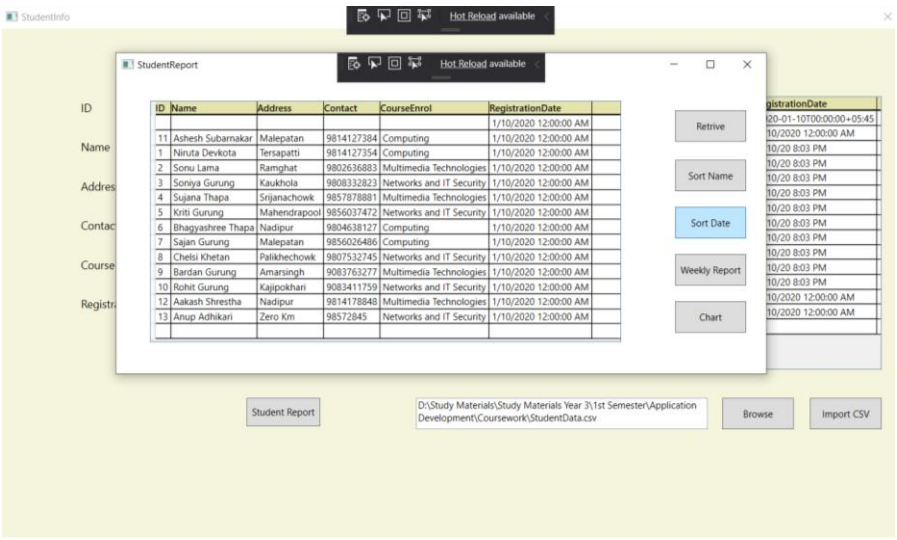

Figure 11: Students Sorted by Date

This is the sorted list of student by date after retrieving the student details.

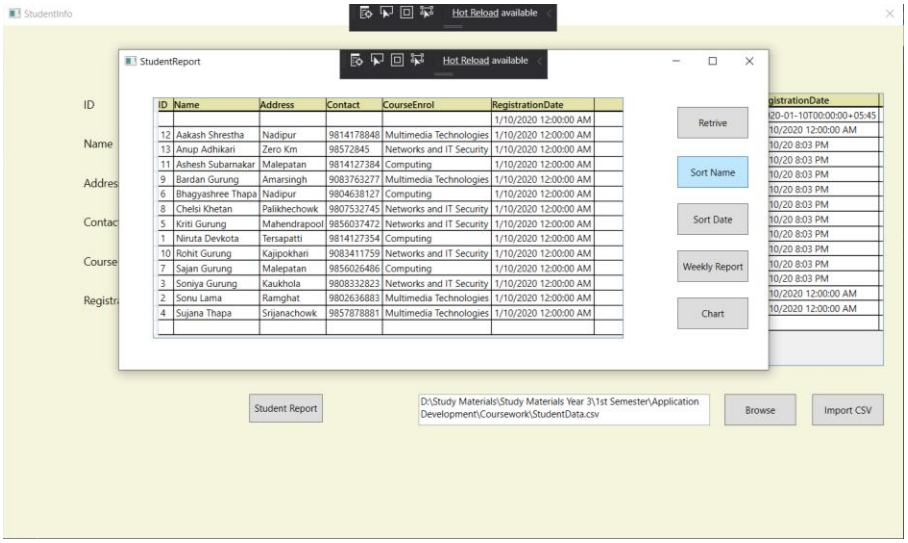

Figure 12: Students Sorted by Name

This is the sorted list of student by name after retrieving the student details.

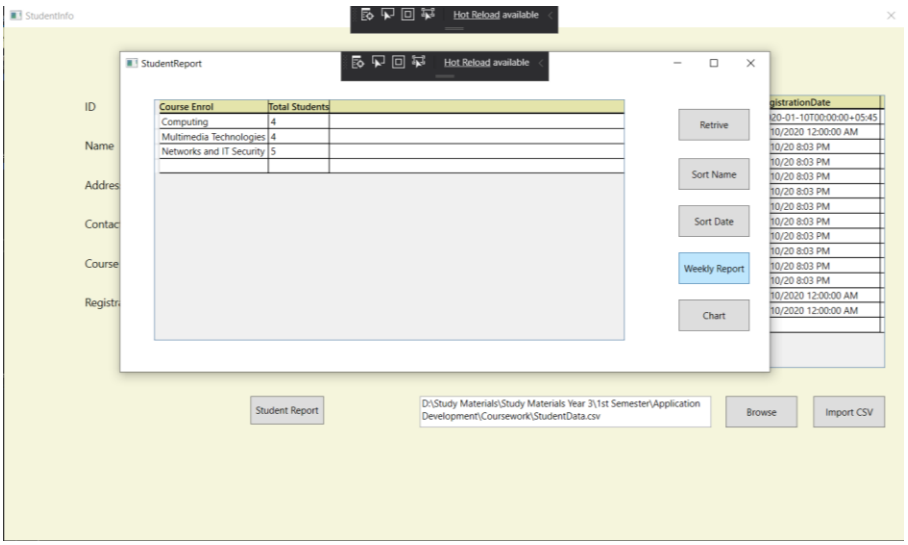

Figure 13: Weekly Report Table

The figure shows the weekly report of the student enrolled in different courses.

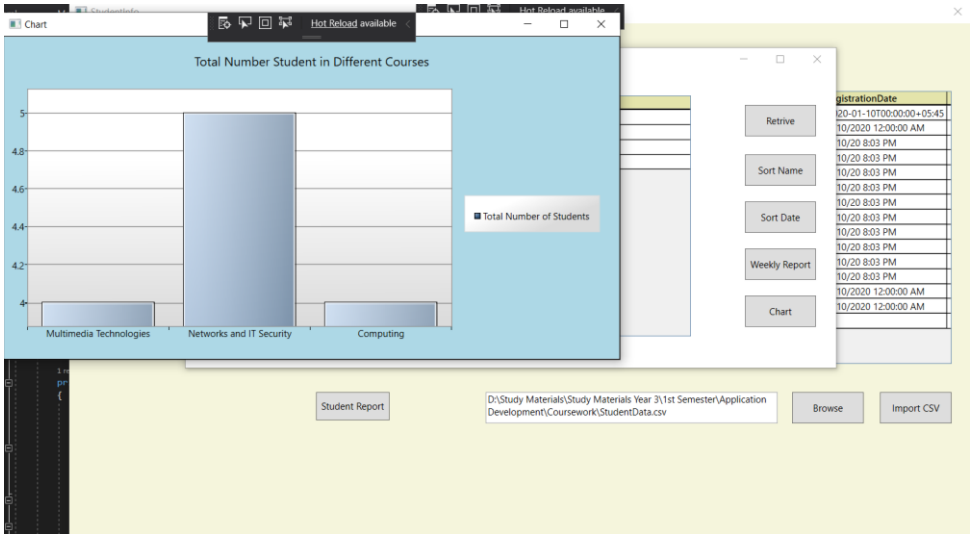

Figure 14: Student Enrollment chart

The figure shows the total number of student enrolled in different courses based on weekly report.

3. System Architecture

Architecture Diagram

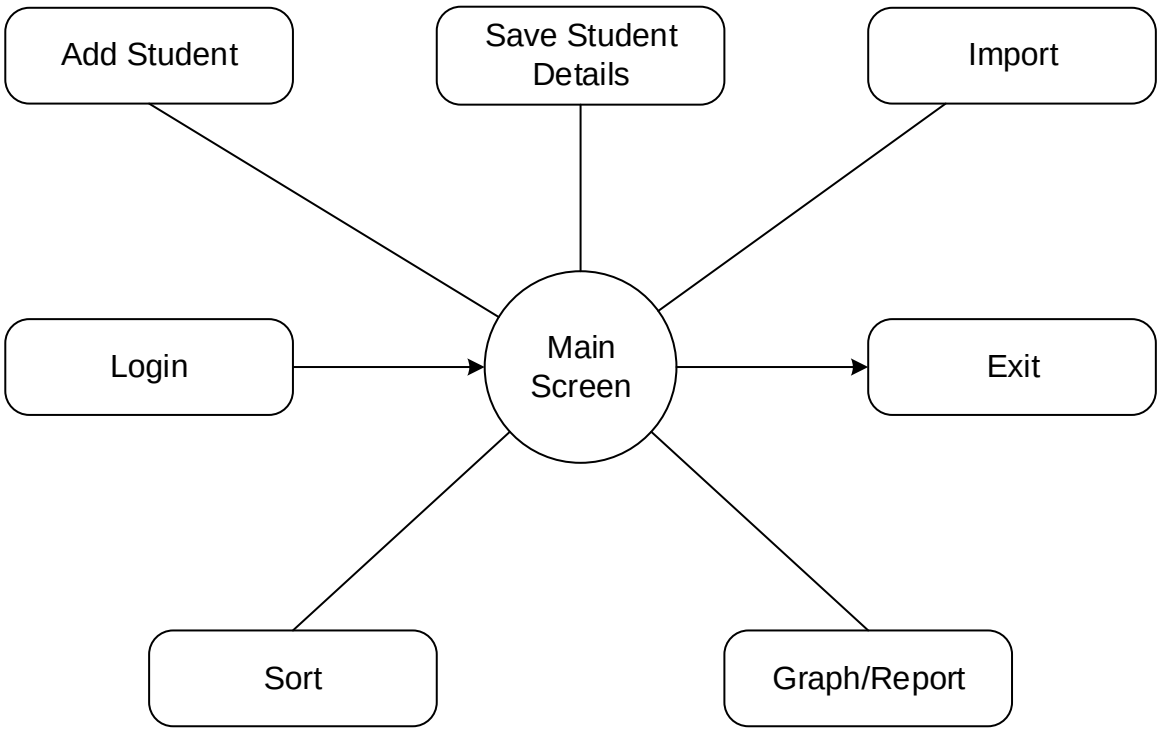

Figure 15: Architecture Diagram

Flow Chart

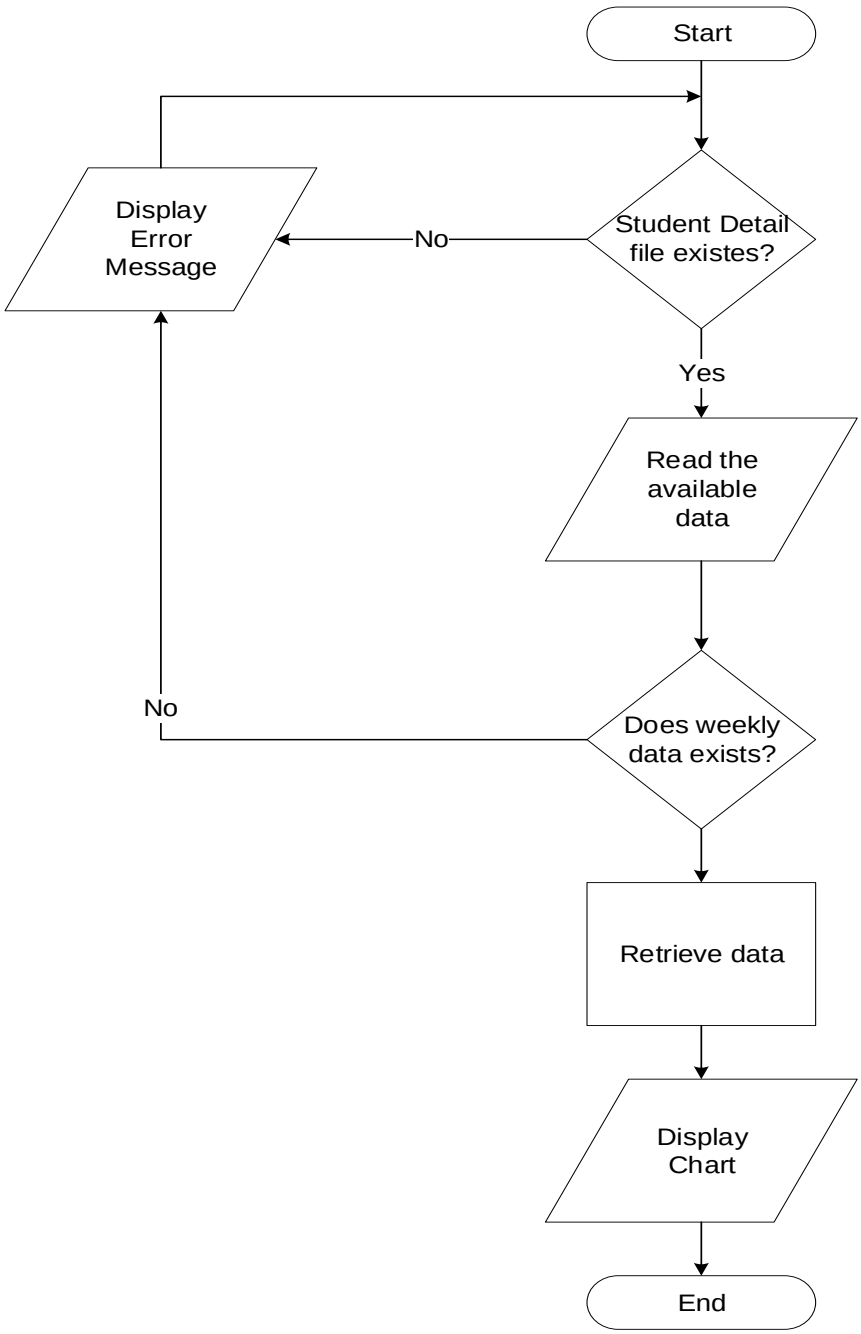

Figure 16: Flowchart for Student Enrollment

Algorithm of Report

Weekly Report for Student Enrolment

Steps:

1. Start
2. Check whether the student data file exists or not.
3. If it doesn’t exists, display error message and restart
4. If exists, read the available data
5. Check whether there is student data or not
6. If data doesn’t exist, display error message and restart
7. If data found, retrieve the data
8. Display the data in the Bar chart
9. Stop

Class Diagram

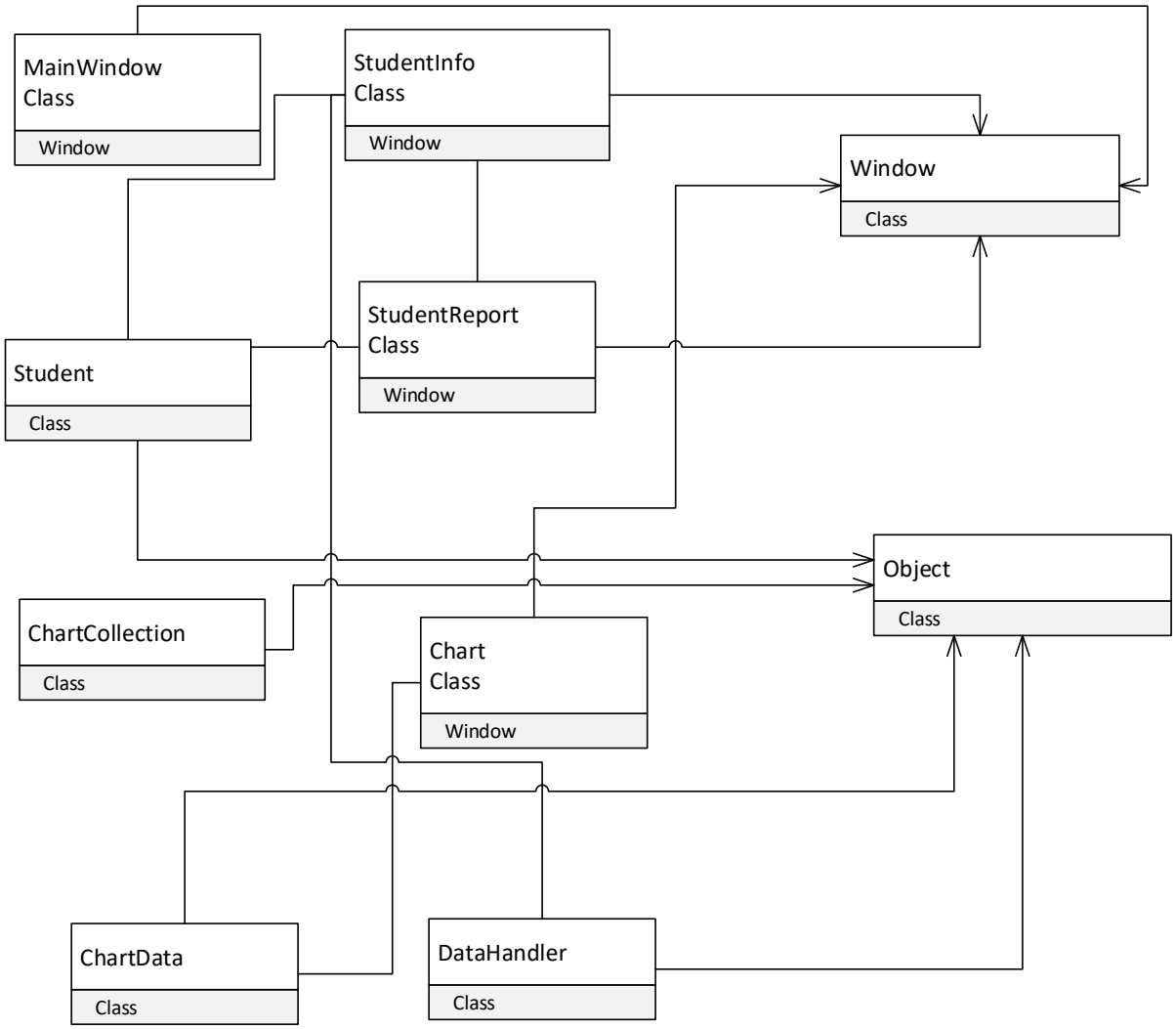

Figure 17: Class Diagram

4. Sorting Algorithm

Bubble sort is a simple sorting algorithm. This sorting algorithm is 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 (tutorialpoint, 2020).

Working

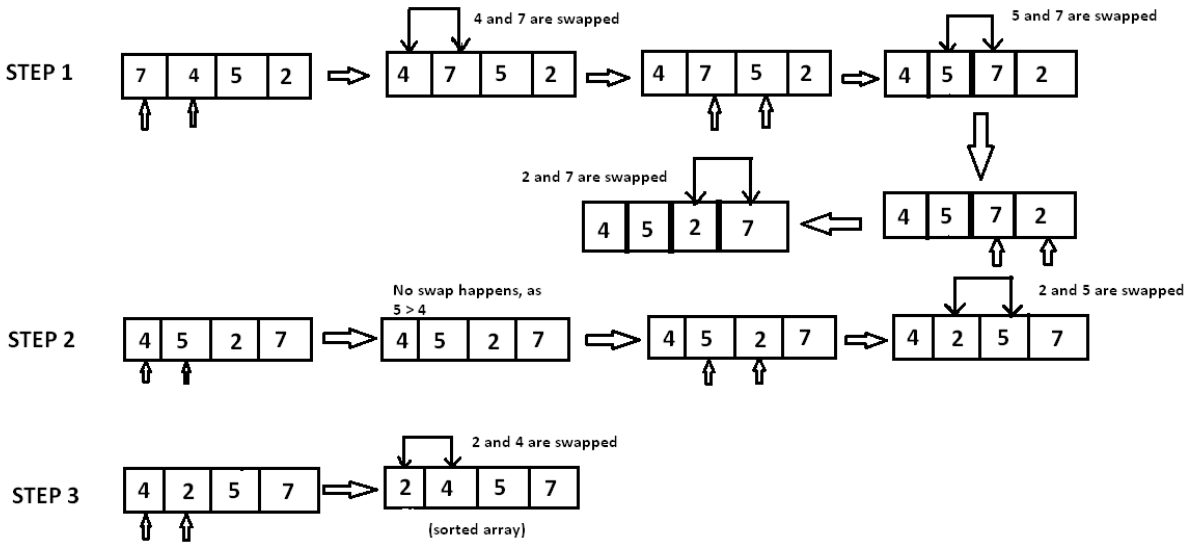

Figure 18: Bubble sort algorithm

In step 1, 7 is compared with 4. Since 7>4, 7 is moved ahead of 4. Since all the other elements are of a lesser value than 7, 7 is moved to the end of the array.

Now the array is A [ ] = {4, 5, 2, 7}.

In step 2, 4 is compared with 5. Since 5>4 and both 4 and 5 are in ascending order, these elements are not swapped. However, when 5 is compared with 2, 5>2 and these elements are in descending order. Therefore, 5 and 2 are swapped.

Now the array is A [ ] = {4, 2, 5, 7}.

In step 3, the element 4 is compared with 2. Since 4>2 and the elements are in descending order, 4 and 2 are swapped.

The sorted array is A [ ] = {2, 4, 5, 7}.

5. Reflection

The developed system is Student Information System. It is developed using Visual Studio 2019 with C# language. The basic logic used in the system reflects the real working environment of the Institution. The GUI designed is highly user friendly so that user with basic system administration can operate the system.

The GUI consists of a login page, student detail form and student report window. After logging in the system, the user can see the student detail form where user can see manually entered data by importing CSV file or user can fill in the form to add more student details for enrolment. There is a button for going to student report page where user can retrieve previous student details, sort the student details by name or registration date, can view weekly report and see enrolment chart.

Though I didn’t have any experience with Visual Studio the coursework was performed in a smooth manner with the guidance from our supervisor and research.

6. Conclusion

The coursework was done with a lot of research and hard work. Since I hadn't used Visual Studio before this coursework it was a challenge. The most difficult part of the coursework was to display the chart but with the help and guidance from my supervisor and lots of research the coursework was completed in time. I learned a lot of new things that will be helpful for my future projects.

7. References

tutorialpoint, 2020. *Data Structure - Bubble Sort Algorithm.* [Online]

Available at:

https://www.tutorialspoint.com/data_structures_algorithms/bubble_sort_algorithm.htm

[Accessed 9 January 2020].

Appendix

Main Window

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

        private void BtnLogin_Click(object sender, RoutedEventArgs e)
        {
            if (txtUsername.Text != "" || txtPassword.Password != "")
            {

                if (txtUsername.Text == "admin" && txtPassword.Password == "admin")
                {
                    MessageBox.Show("Sucess!");
                    this.Hide();
                    StudentInfo studentInfo = new StudentInfo();
                    studentInfo.ShowDialog();


                }
                else ErrorMessage();
            }
        }
        public void ErrorMessage()
        {
            MessageBox.Show("Incorrect username or password", "Login Error");
        }

        private void BtnCancel_Click(object sender, RoutedEventArgs e)
        {
            txtUsername.Text = "";
            txtPassword.Password = "";
        }
    }
}
```

StudentInfo

```
namespace Student_Information_System
{
    /// <summary>
    /// Interaction logic for StudentInfo.xaml
    /// </summary>
    public partial class StudentInfo : Window
    {
        public StudentInfo()
        {
            InitializeComponent();
            //LoadStudentData();
            LoadGrid();

        }


        //csv to DataTable


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

            //var dataSet = new DataSet();
            //dataSet.ReadXmlSchema(@"D:\StudentCWSchema.xml");
            ////dataSet.ReadXml(@"D:\StudentCWData.xml");
            if (txtID.Text == "" || txtName.Text == "" || txtAddress.Text == "" ||
txtContact.Text == "" || cbCourseEnrol.Text == "" || txtDate.Text == "")
            {
                MessageBox.Show("Enter required information in all fields!");
            }
            else
            {
                try
                {
                    if (File.Exists(@"D:\student.xml"))
                    {
                        dataSet.ReadXml(@"D:\student.xml");
                        //dataSet.ReadXmlSchema(@"D:\StudentCWSchema.xml");

                    }
                    else
                    {
                        dataSet = handler.CreateDataSet();
                        //WriteXml(@"D:\student.xml");
                    }
                    AddStudentData(dataSet);
                    dataSet.WriteXml(@"D:\student.xml");
                    //dataSet.WriteXmlSchema(@"D:\StudentCWSchema.xml");
                    LoadGrid();
                    MessageBox.Show("Sucessfully added!");

                }
                catch (Exception)
                {

                }
            }
        }
```

```
public void LoadGrid()
{
    var dataSet = new DataSet();
    if (File.Exists(@"D:\student.xml"))
    {
        dataSet.ReadXml(@"D:\student.xml");
        DataGridXAML.ItemsSource = dataSet.Tables["Student"].DefaultView;
    }
}

private void AddStudentData(DataSet dataSet)
{
    var newRow = dataSet.Tables["Student"].NewRow();
    newRow["ID"] = txtID.Text;
    newRow["Name"] = txtName.Text;
    newRow["Address"] = txtAddress.Text;
    newRow["Contact"] = txtContact.Text;
    newRow["CourseEnroll"] = cbCourseEnrol.Text;
    newRow["RegistrationDate"] = txtDate.DisplayDate;
    dataSet.Tables["Student"].Rows.Add(newRow);
}

private void Btn_Report(object sender, RoutedEventArgs e)
{

    StudentReport studentReport = new StudentReport();
    studentReport.ShowDialog();
    this.Hide();
}

private void Btn_Browse(object sender, RoutedEventArgs e)
{
    OpenFileDialog openfile = new OpenFileDialog();
    openfile.DefaultExt = ".csv";
    openfile.Filter = "(.csv)|*.csv";

    var browsefile = openfile.ShowDialog();

    if (browsefile == true)
    {
        txtFilePath.Text = openfile.FileName;
    }


}

private void Btn_ImportCVS(object sender, RoutedEventArgs e)
{
    var dataSet = new DataSet();
    dataSet.ReadXml(@"D:\student.xml");

    string filePath = txtFilePath.Text;
    //read all std from file code copy

    using (var reader = new StreamReader(filePath))
    {
        reader.ReadLine();
        while (!reader.EndOfStream)
        {
            var line = reader.ReadLine();
            var values = line.Split(',');
            var newRow = dataSet.Tables["Student"].NewRow();
```

```
                    newRow["ID"] = values[0];
                    newRow["Name"] = values[1];
                    newRow["Address"] = values[2];
                    newRow["Contact"] = values[3];
                    newRow["CourseEnroll"] = values[4];
                    newRow["RegistrationDate"] = values[5];
                    dataSet.Tables["Student"].Rows.Add(newRow);

                    dataSet.WriteXml(@"D:\student.xml");
                }
            }
            DataGridXAML.ItemsSource = dataSet.Tables["Student"].DefaultView;
        }

        private void BtnClear(object sender, RoutedEventArgs e)
        {
            DataGridXAML.ItemsSource = null;
            txtFilePath.Text = "";
        }
    }
    }
```

Chart Collection

```
namespace Student_Information_System
{
    class ChartCollection : Collection<ChartData>
    {

        public ChartCollection()
        {

            var dataSet = new DataSet();
            dataSet.ReadXml(@"D:\student.xml");

            DataTable studentReport = dataSet.Tables[0];


            int total_Computing = 0;
            int total_Network = 0;
            int total_Multimedia = 0;

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

            for (int i = 0; i < studentReport.Rows.Count; i++)
            {
                String col = studentReport.Rows[i]["CourseEnroll"].ToString();
                if (col == "Computing")
                {
                    total_Computing++;
                }
                else if (col == "Networks and IT Security")
                {
                    total_Network++;
                }
                else if (col == "Multimedia Technologies")
                {
                    total_Multimedia++;
                }
            }

            Add(new ChartData("Multimedia Technologies", total_Multimedia));
            Add(new ChartData("Networks and IT Security", total_Network));
            Add(new ChartData("Computing", total_Computing));



    }
    }
}
```

ChartData

```
namespace Student_Information_System
{
    class ChartData
    {
        public ChartData(string CourseName, int TotalStudents)    // Constructor
        {
            this.CourseName = CourseName;
            this.TotalStudents = TotalStudents;
        }
        public string CourseName                // Name Property
        {
            get;
            set;
        }
        public long TotalStudents                // Population Property
        {
            get;
            set;
        }
    }
}
```

DataHandler

```
namespace Student_Information_System
{
    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(string));
            dt.Columns.Add("Name", typeof(string));
            dt.Columns.Add("Address", typeof(string));
            dt.Columns.Add("Contact", typeof(string));
            dt.Columns.Add("CourseEnroll", typeof(string));
            dt.Columns.Add("RegistrationDate", typeof(DateTime));
            return dt;
        }
    }

}
```

Student

```
namespace Student_Information_System
{
    class Student
    {
        public string ID { get; set; }

        public string Name { get; set; }

        public string Address { get; set; }

        public string Contact { get; set; }

        public string CourseEnrol { get; set; }

        public DateTime RegistrationDate { get; set; }
    }
}
```

StudentReport

```
namespace Student_Information_System
{
    /// <summary>
    /// Interaction logic for StudentReport.xaml
    /// </summary>
    public partial class StudentReport : Window
    {
        DataTable buffer;
        public StudentReport()
        {

            InitializeComponent();
        }
        private void DataShow()
        {
            string dataXMLFile = @"D:\student.xml";
            System.Data.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("Contact", typeof(String));
            buffer.Columns.Add("CourseEnrol", typeof(String));
            buffer.Columns.Add("RegistrationDate", typeof(DateTime));

            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);
                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());

            }
            DataView dataView = new DataView(buffer);
            DataGrid2.ItemsSource = dataView;
        }

        private void Btn_Date(object sender, RoutedEventArgs e)
        {
            DataView dataView = new DataView(buffer)
            {
                Sort = "RegistrationDate ASC"
            };
            DataGrid2.ItemsSource = dataView;
        }

        private void Btn_SortName(object sender, RoutedEventArgs e)
        {
            DataView dataView = new DataView(buffer)
            {
                Sort = "Name ASC"
            };
            DataGrid2.ItemsSource = dataView;
```

```
        }

        private void Btn_Retrive(object sender, RoutedEventArgs e)
        {
            DataShow();
        }



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

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

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


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

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


            DataGrid2.ItemsSource = dt.DefaultView; // is the name of data grid

        }

        private void BtnChart(object sender, RoutedEventArgs e)
        {
            Chart chart = new Chart();
            chart.ShowDialog();
            this.Hide();

        }
    }
}
```