Marking Scheme

Informatics College Pokhara

informatics
college pokhara

Application Development
CS6004NI
Course Work 1

Submitted By: **Sushil Gautam**
London Met ID: Enter ID Here

Submitted To: Ishwor Sapkota
Module Leader

Component Grade and Comments	
A. Implementation of Application	
User Interface and proper controls used for designing	User Interface is complete but not separated and have proper use of controls
Manual data entry or import from csv	appropriate use of data types but missing some properties required or missing CRUD operation
Data Validation	missing most of the validation
Enrollment Report & weekly report in tabular format	Any one of the report is missing or not complete
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.

Marking Scheme

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:	**C+**	**C+**

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. Ca present the workflow and explain his task properly.

Application Development

CS6004NA

Coursework 1

Student Name: Sushil Gautam(sushil.gautam.17@icp.edu.np)
Student ID: 17030759
Course: BSc (Hons) Computing
Submitted To: Mr. Ishwor Sapkota

January 10, 2020

Contents

List of Figures

1. Introduction

This report demonstrates the detailed instructions to run the program and the architecture of Students Information System in terms of software classes. Here, I had described about the classes properties and methods that are used in this software. To develop this software, I had developed different classes and among them some classes are from other sources and the functions of this classes are explained detailly in other sections of this report. I had used Sorting algorithm to view the report according to the enrolled students with their name and enrolled date. The main features of this software is allowing user to input the student personal detail including registration date and saving that information in the xml file. Here a user can enroll the students by filling the enroll student UI and all data are saved in the form of xml. User can view total students enrolled by clicking the view report button in the home page. User can also view students' sort by name and date from which user can get benefit to search students easily. To get information about students enrolled this week in different course there is another table in view reports form which provides the information about total students enrolled in different subject in this week only. To view total enrolled students there is one button in the home page called view chart which displays the total number of students with different subject in the graph.

2) User Manual

The following screenshots explains about how to use this student information system and the advantages of using this system.

After running this application Login form is displayed in the initial stage to provide the information security. To enter into the system the user should have to write susheelgatuam321@gmail.com as an email and 1234 as password. This form provides the data security because only a valid user can use the system data.

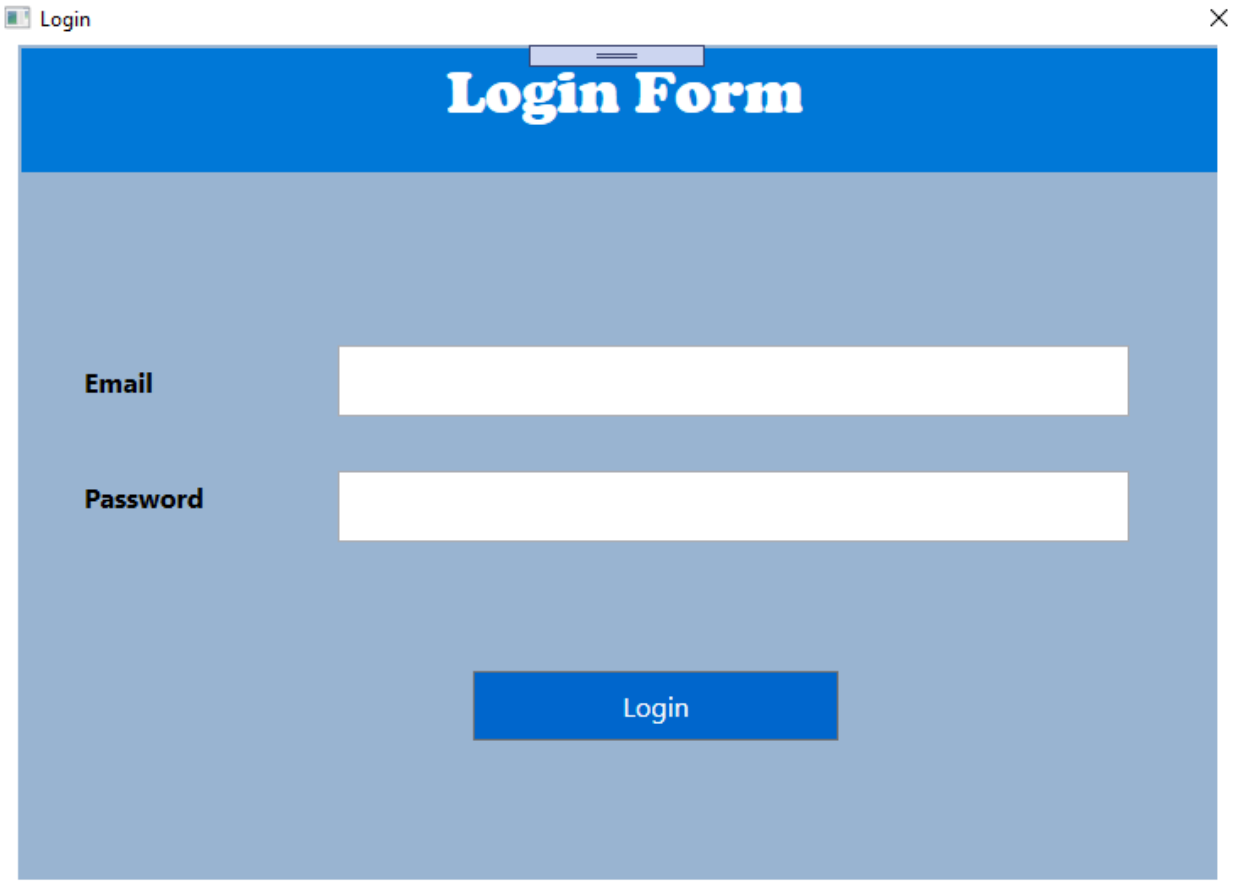


Figure 1: Login Form

If the user inputs incorrect information then the system returns an information message and if the user input correct information then Home page screen appears as same as like figure mentioned below:

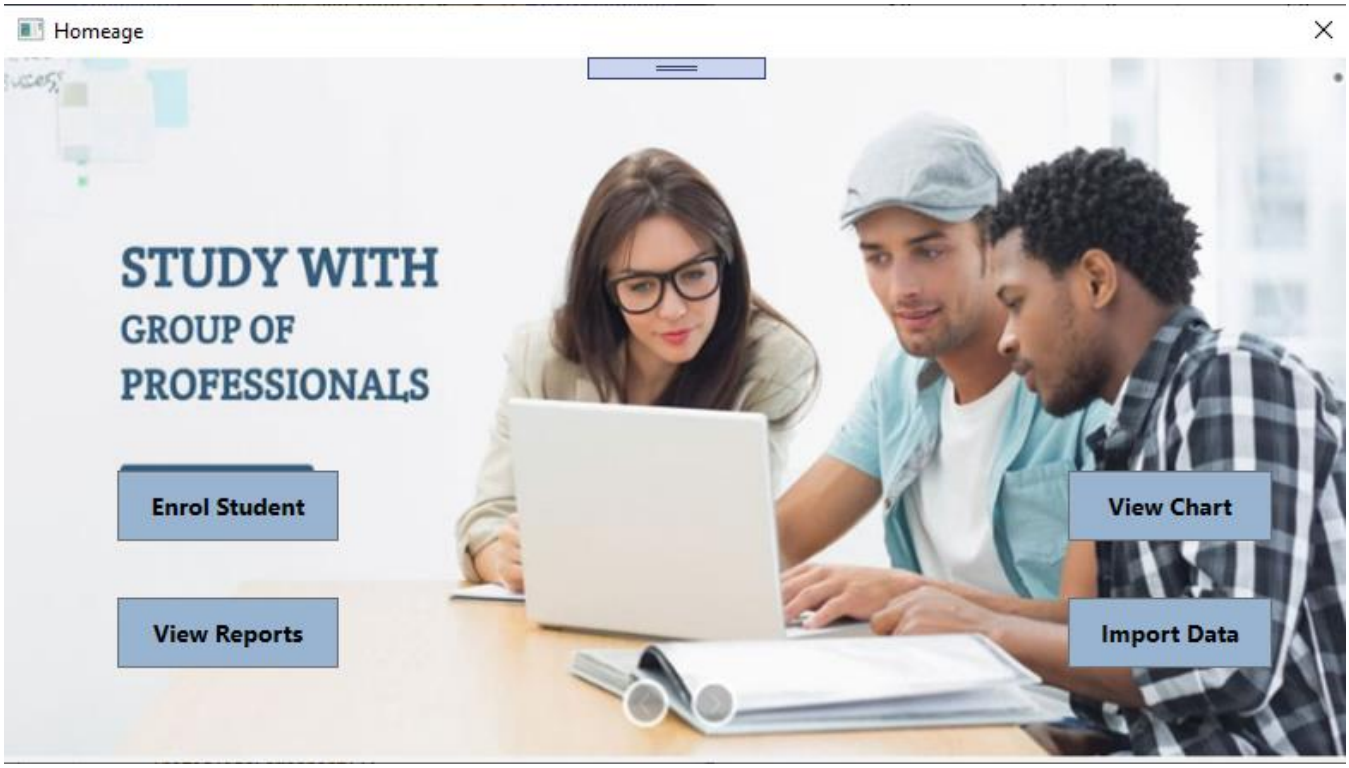


Figure 2: Home Page

This home page has four different buttons Enroll Students, View Chart, View Reports and Import Data. Each button has different functions according to their name.

When a user Clicks Enroll Students button then the new form opens as same as like figure mentioned below:

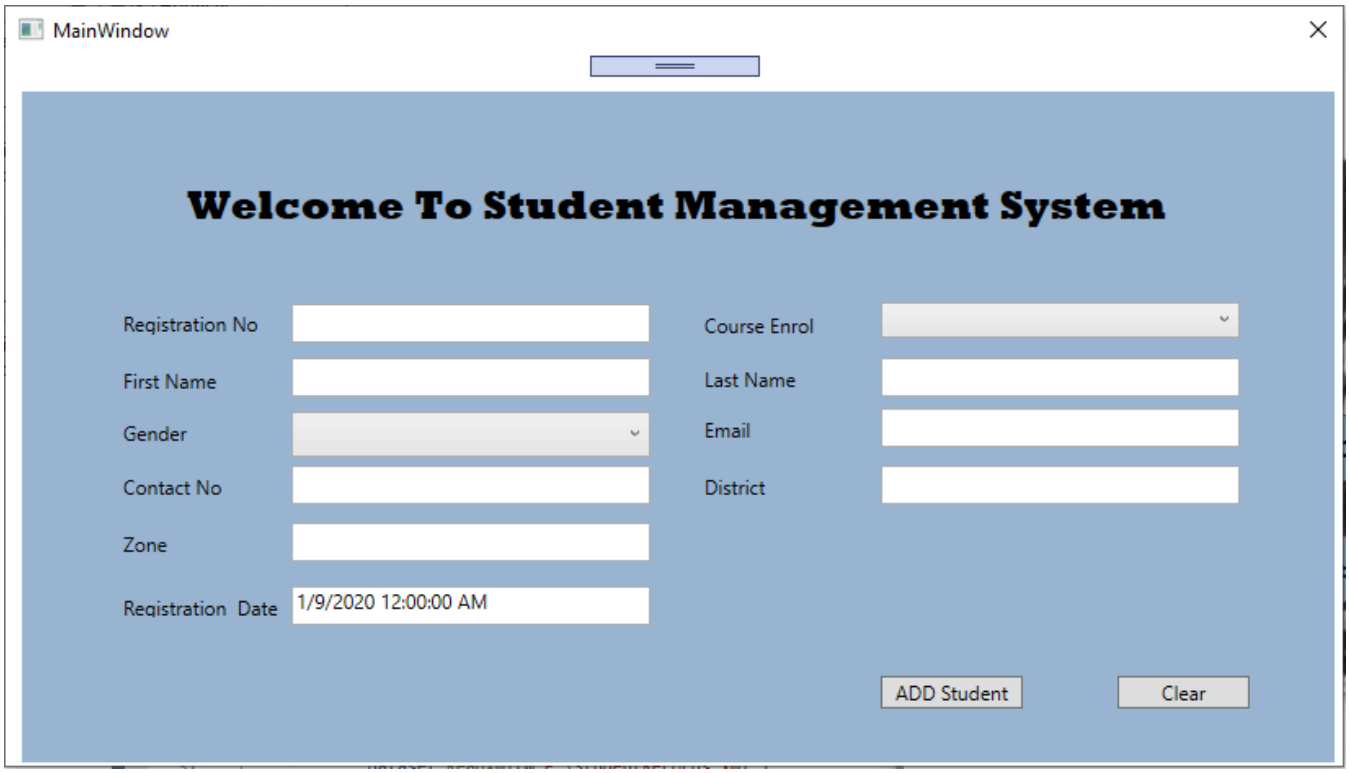


Figure 3: Enroll Students Form

In this form to enroll the student a user can fill the detail information about the student where student registration number and registration date are auto generated by the system. When the user clicks the add student button then all data are stored in the xml file but if the user input incorrect data then the system displays an error message in the dialog box. Here Clear button cleans all the data that are recorded in the text filed of the system.

When a user inputs incorrect data or if a user clicks Add student without inputting the data then the system throws and error message as same as like in the following picture.

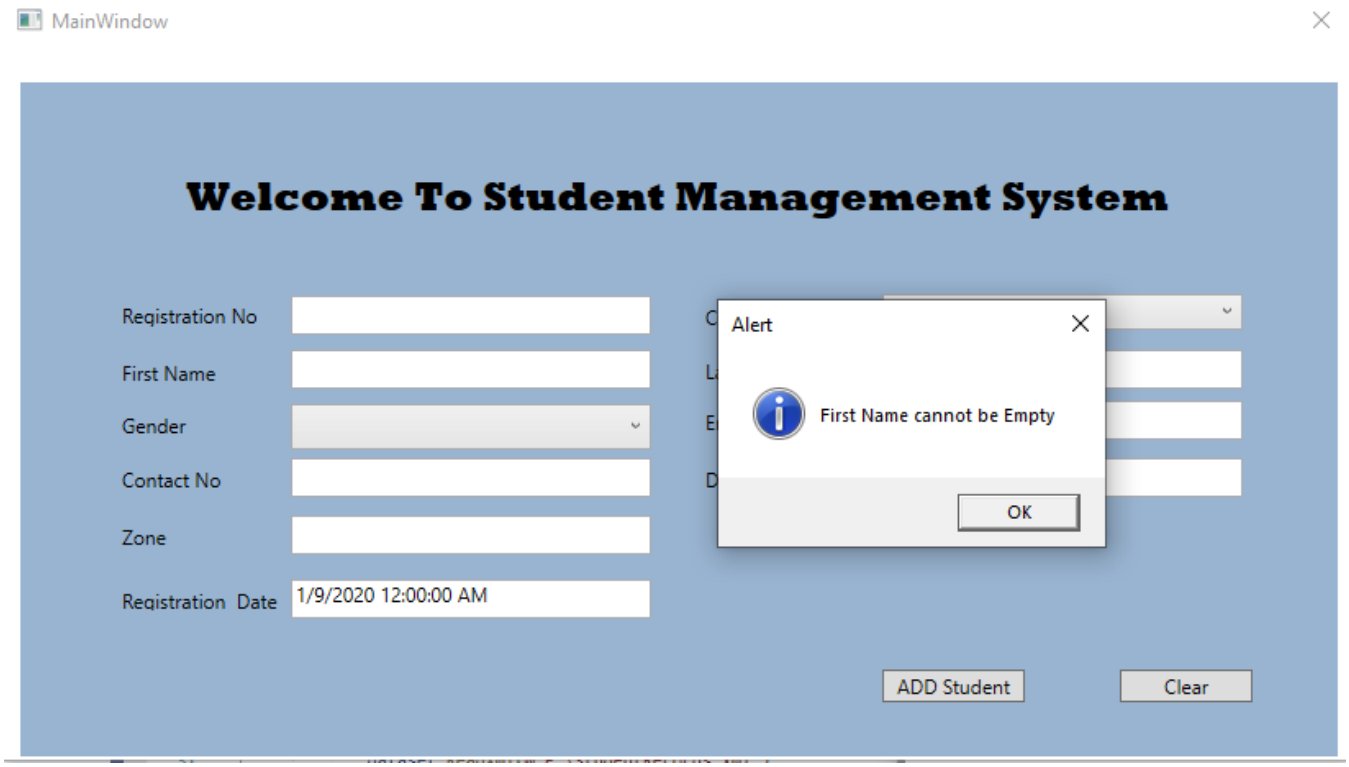


Figure 4: Validating user input

When a user clicks view report button in the home page then a new form is opened where there are two different data grids. One data grid is for displaying all record information of the students and another is for displaying weekly enrolled student detail.

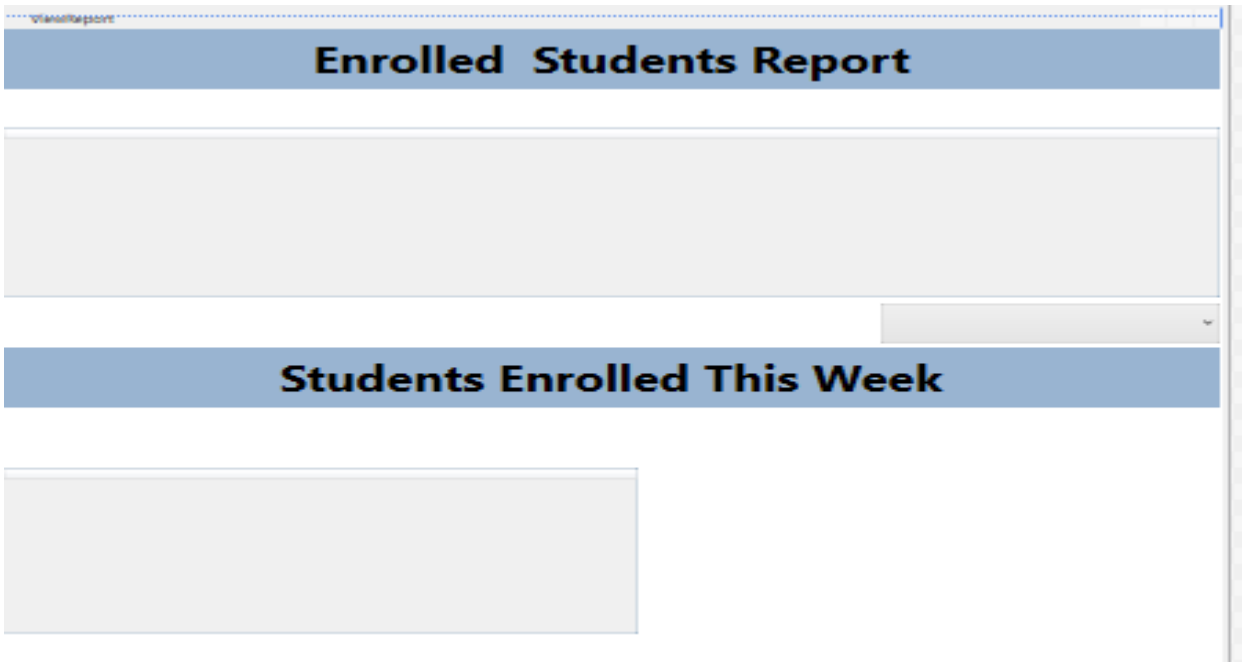


Figure 5: View Report Form

If there are already enrolled students in the system then that data is displayed as same as like in the following picture.

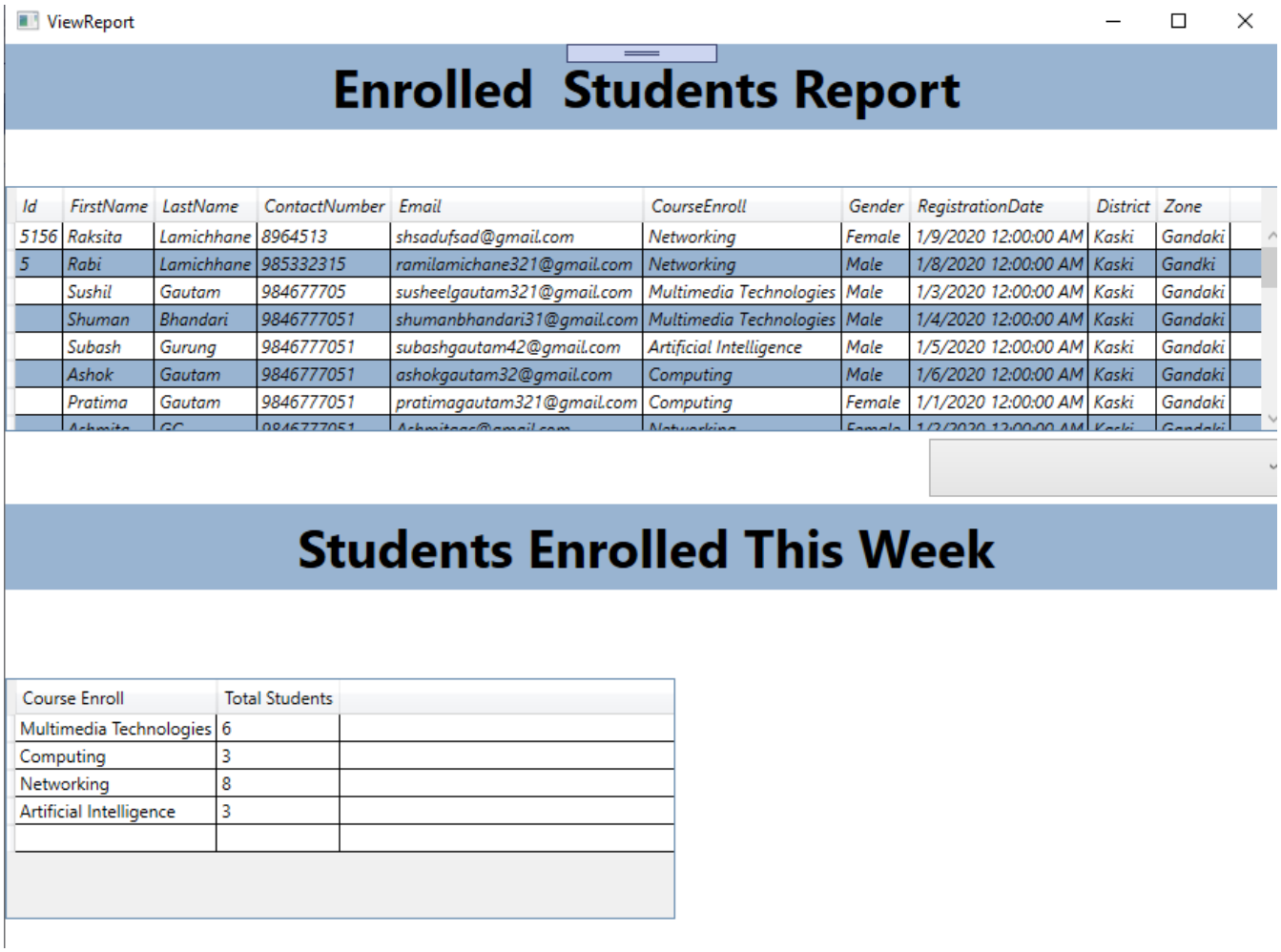


Figure 6: Report with student's data

Here are two different combo box button and each of them has different functions to sort the data according to the student name and enroll date.

When a user clicks sort by name combo box button then all students are shown in the ascending order in the report grid as same as like in the following figure.

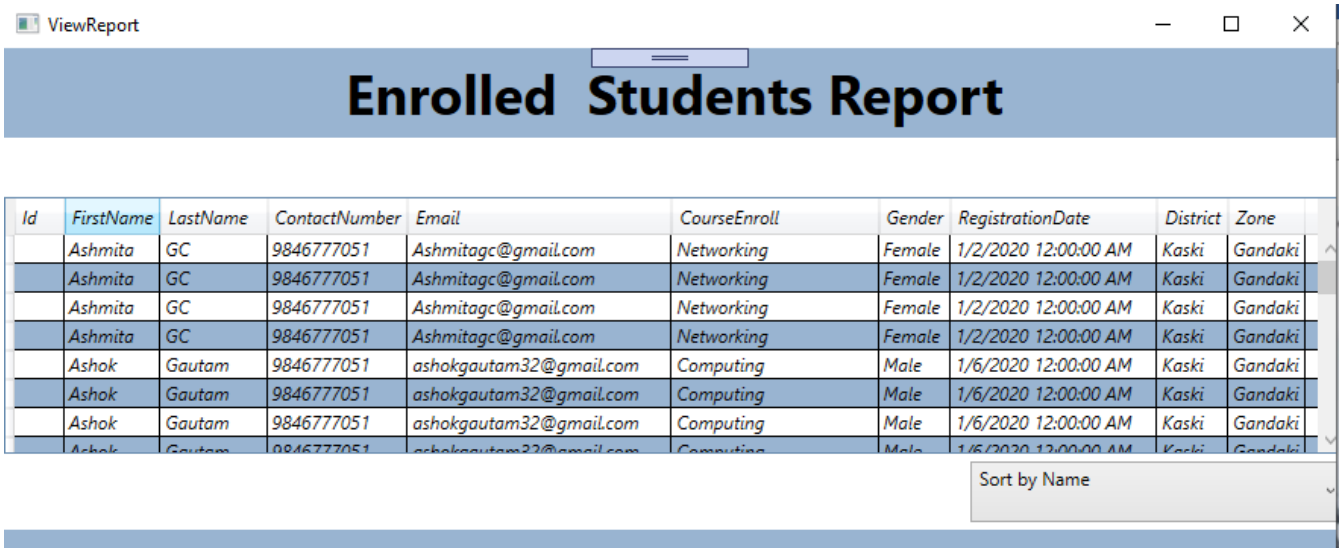


Figure 7: Students sort by name

And If the user clicks sort by date combo box then all students are sorted according to the latest enrolled date as shown as in the following figure:

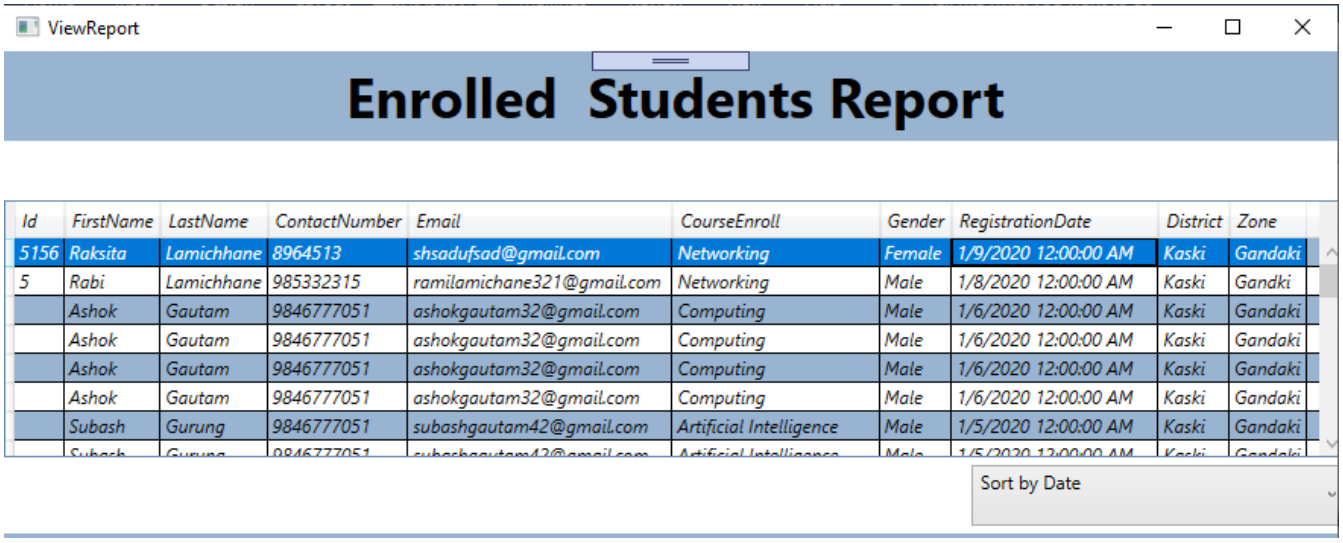


Figure 8: Students Sort by Enrolled Date

To import the csv data from the excel here is a new form called excel data which is opened by clicking the import data from the home page and the following form opens.

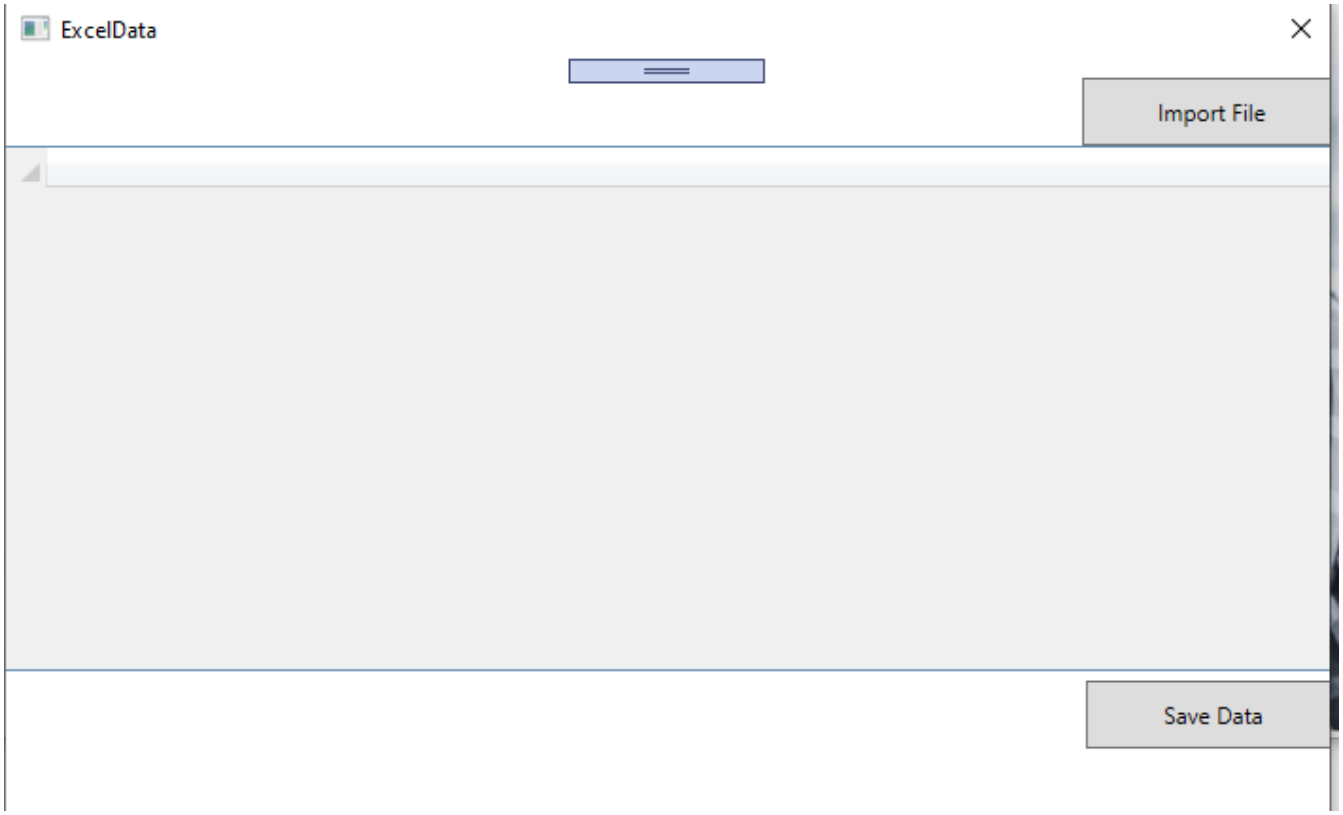
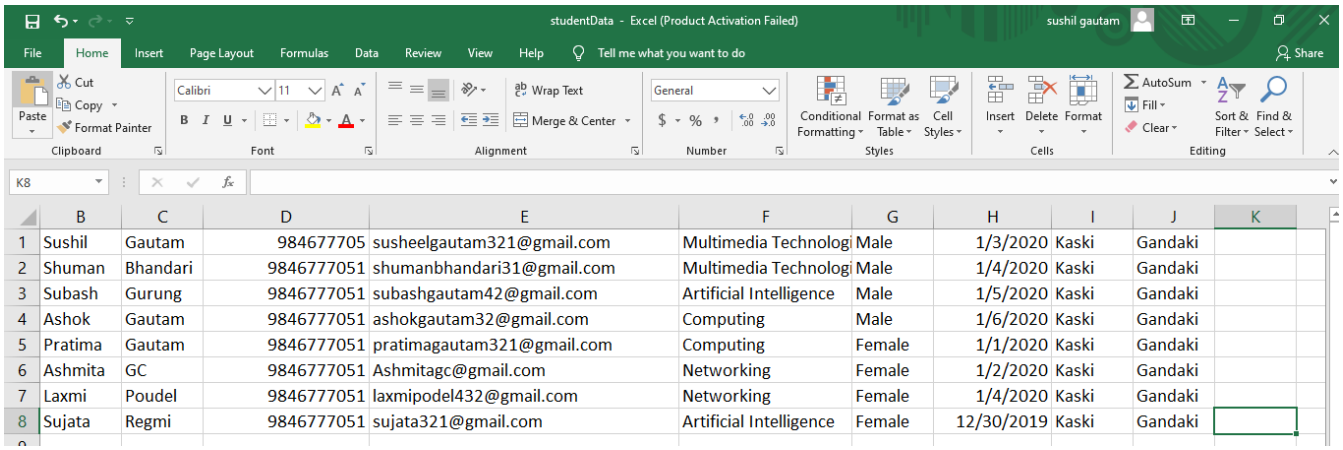


Figure 9: Excel Data form

Here this form has two different buttons where import file opens the file manager to import csv file and save data button saves all csv data to the xml.

To import data from the csv the data should be in the following format in the excel.



	B	C	D	E	F	G	H	I	J	K
1	Sushil	Gautam	984677705	susheelgautam321@gmail.com	Multimedia Technologi	Male	1/3/2020	Kaski	Gandaki	
2	Shuman	Bhandari	9846777051	shumanbhandari31@gmail.com	Multimedia Technologi	Male	1/4/2020	Kaski	Gandaki	
3	Subash	Gurung	9846777051	subashgautam42@gmail.com	Artificial Intelligence	Male	1/5/2020	Kaski	Gandaki	
4	Ashok	Gautam	9846777051	ashokgautam32@gmail.com	Computing	Male	1/6/2020	Kaski	Gandaki	
5	Pratima	Gautam	9846777051	pratimagautam321@gmail.com	Computing	Female	1/1/2020	Kaski	Gandaki	
6	Ashmita	GC	9846777051	Ashmitagc@gmail.com	Networking	Female	1/2/2020	Kaski	Gandaki	
7	Laxmi	Poudel	9846777051	laxmipodel432@gmail.com	Networking	Female	1/4/2020	Kaski	Gandaki	
8	Sujata	Regmi	9846777051	sujata321@gmail.com	Artificial Intelligence	Female	12/30/2019	Kaski	Gandaki	

Figure 10: Source of Excel Data

After inserting all this data, it should have to save in csv format then only this student information system can import this data and save this data to the xml.

After creating the csv file, we can import this data by opening the file manager from the import file button in the excel data page of this system as shown as in the following picture.

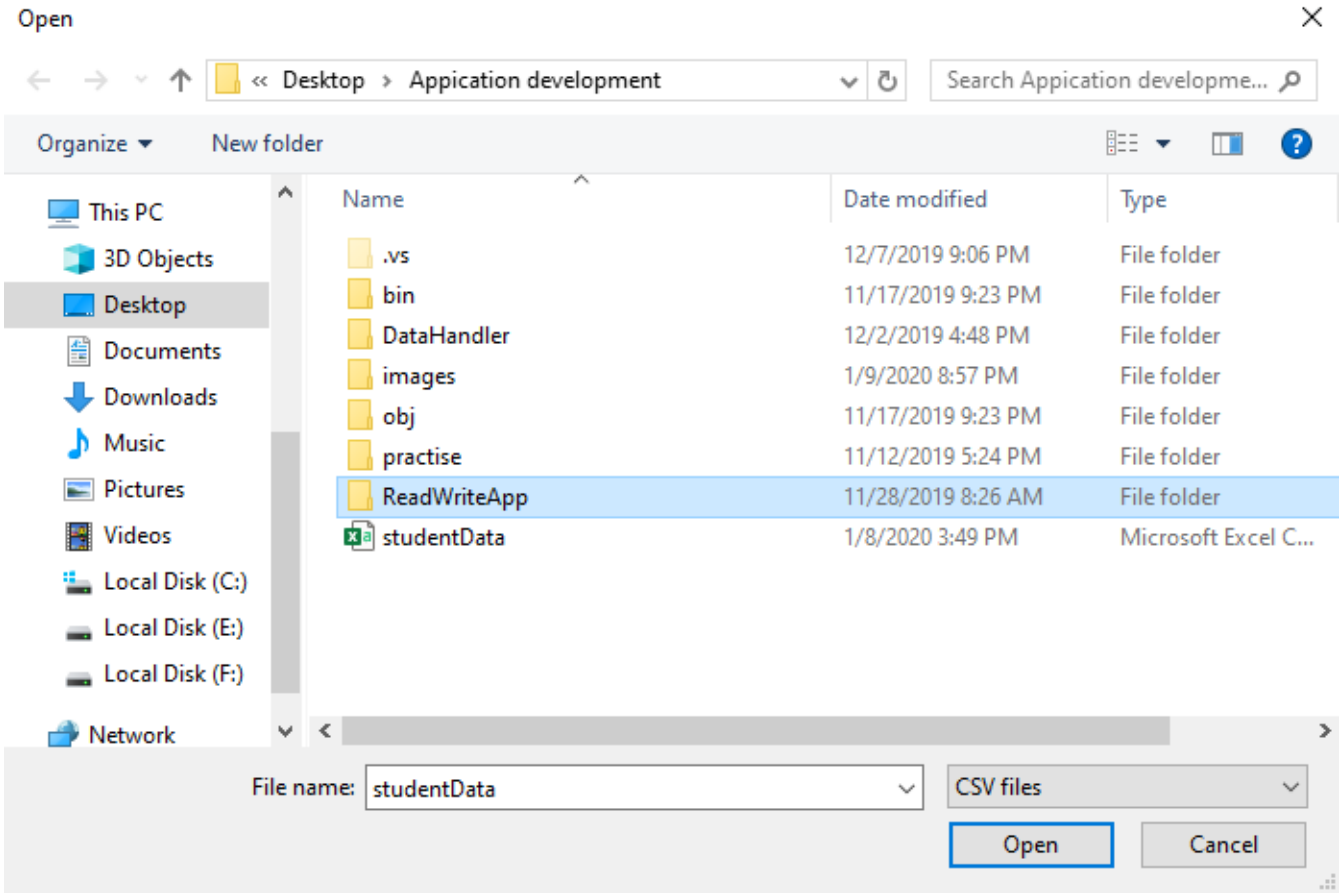
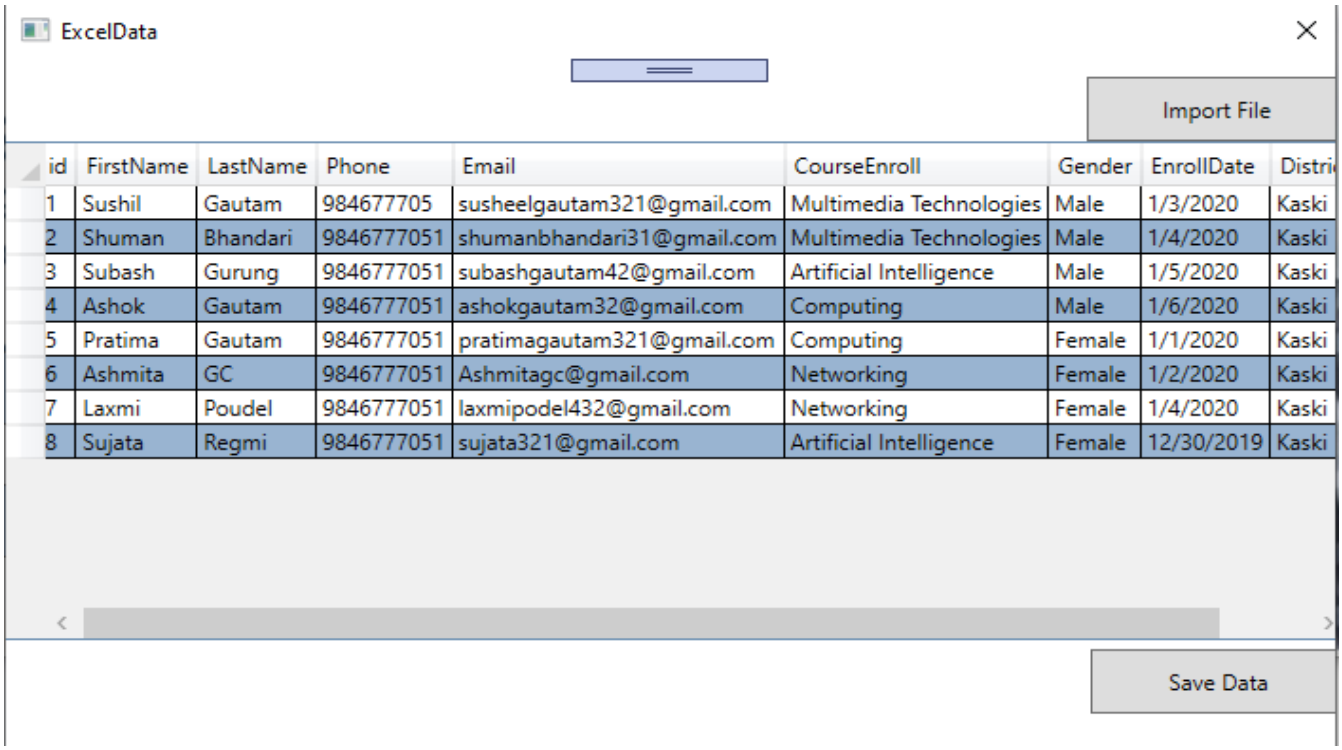


Figure 11: File manager after importing csv file

After successful import, all csv data are shown in the data grid as same as like the following picture:



id	FirstName	LastName	Phone	Email	CourseEnroll	Gender	EnrollDate	Distri
1	Sushil	Gautam	984677705	susheelgautam321@gmail.com	Multimedia Technologies	Male	1/3/2020	Kaski
2	Shuman	Bhandari	9846777051	shumanbhandari31@gmail.com	Multimedia Technologies	Male	1/4/2020	Kaski
3	Subash	Gurung	9846777051	subashgautam42@gmail.com	Artificial Intelligence	Male	1/5/2020	Kaski
4	Ashok	Gautam	9846777051	ashokgautam32@gmail.com	Computing	Male	1/6/2020	Kaski
5	Pratima	Gautam	9846777051	pratimagautam321@gmail.com	Computing	Female	1/1/2020	Kaski
6	Ashmita	GC	9846777051	Ashmitagc@gmail.com	Networking	Female	1/2/2020	Kaski
7	Laxmi	Poudel	9846777051	laxmipodel432@gmail.com	Networking	Female	1/4/2020	Kaski
8	Sujata	Regmi	9846777051	sujata321@gmail.com	Artificial Intelligence	Female	12/30/2019	Kaski

Figure 12: Csv file sown in Data Grid

After getting the student information in the data grid we can save it into the xml by clicking the save data button.

And after adding this data in xml the system provides the success information as same as like in the following picture and all this data can be viewed in the view report form:

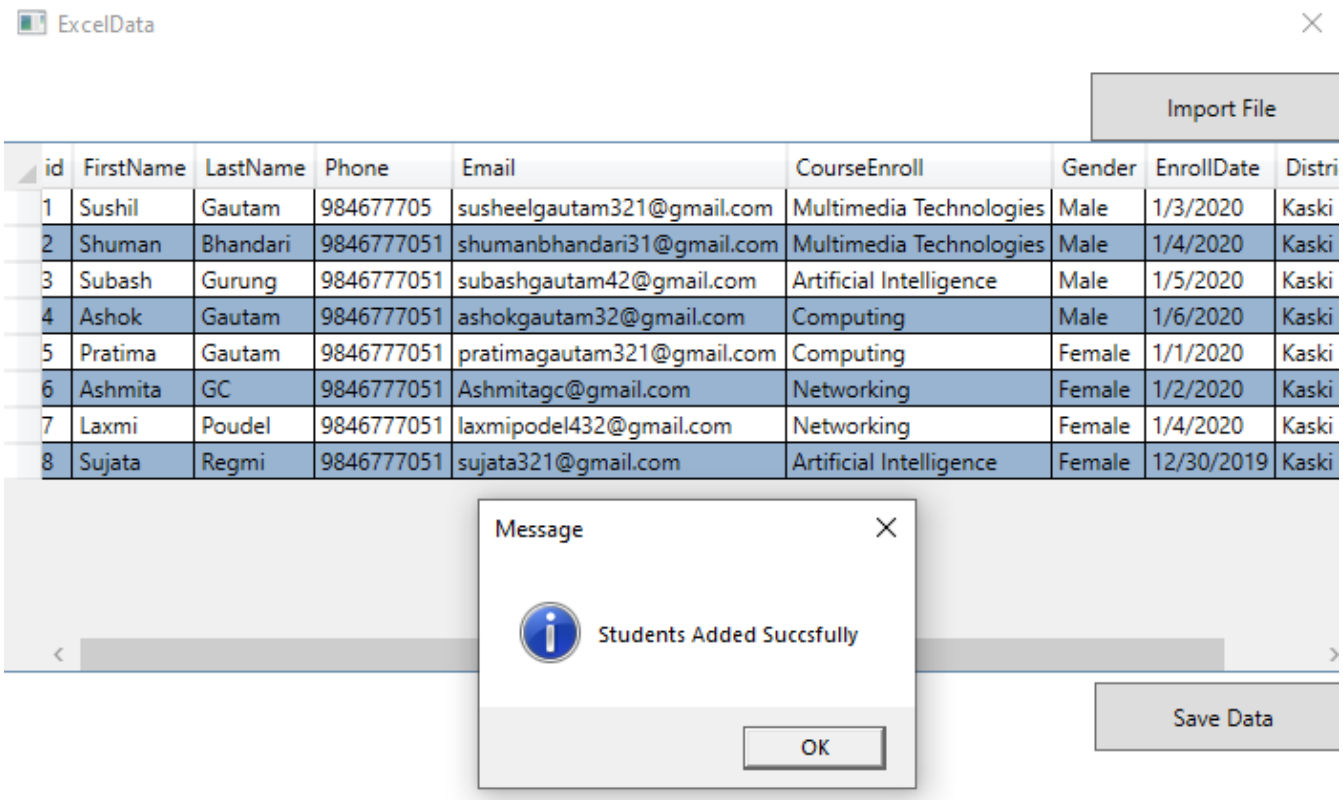


Figure 13: Csv data save to xml

We can look all students enrolled in the system in the graph by clicking the view chart button in the home page and all students according to the course can be viewed in the graph as shown as like in the following picture.

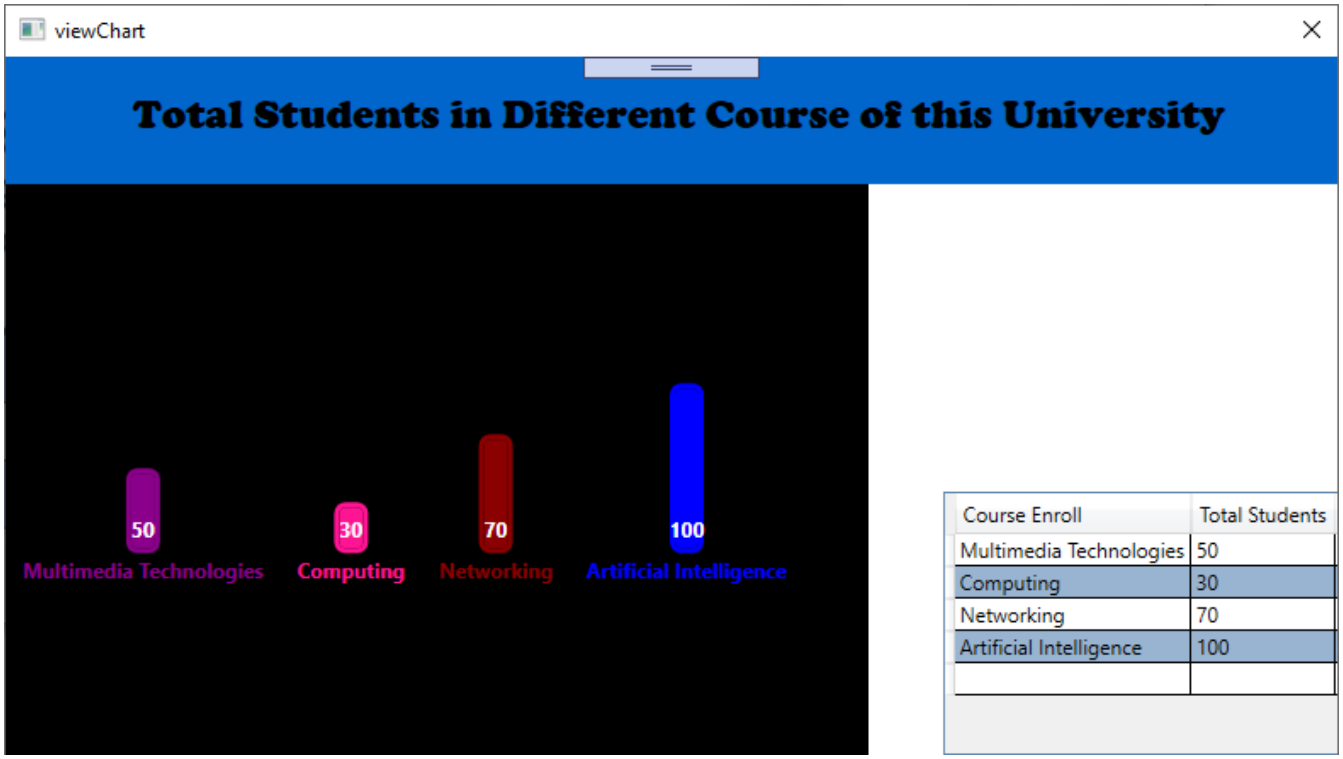


Figure 14: Students information in Graph format

3 Journal Articles

I. To obtain a competitive advantage many companies trained innovation and educational reform mainly focuses on training creative students. For maintenance of student data, student information system provides a simple interface. The conception and organization of accurate information with a students' academic career is critically important in the university. Student management system should deal with every kind of student detail, academic related reports, students enrolled date, completed semesters, final exam result details and many more. At past, every colleges trusted heavily on paper records but paper records are a traditional way of method. If they want to provide information to the students then they should write a paper and attached it in the notice board and every student should have to visit that board to check the information which is so difficult one. So that if there is student management system then we can easily face those problems (S.R.Bharamagoudar, 2013).

II. The goal of Student information system is to create an integrated information technology environment for students, staff and administration. If a system focuses on services and integration for end users the it more effective. To replace the paper-based records the design and implementation of student information is developed. Student information system can deal with every kinds of data from enrollment to graduation including program of study, attendance record, payment fees and examination results. (Dipin Budhrani1,Vivek Mulchandani,Yugchhaya Galphat, 2018)

III. A structure of a class of algorithms can be visible by synthesizing each algorithm in the class from a common high level building up a family tree of algorithms and specification. This article illustrates this technique by a simple example delineation how four common sorting algorithms, quick sort, insertion sort, merge sort, selection sort can be created from a common requirement (K. L. Clark, J. Darlington, 1980).

4. System Architecture

Architecture Diagram

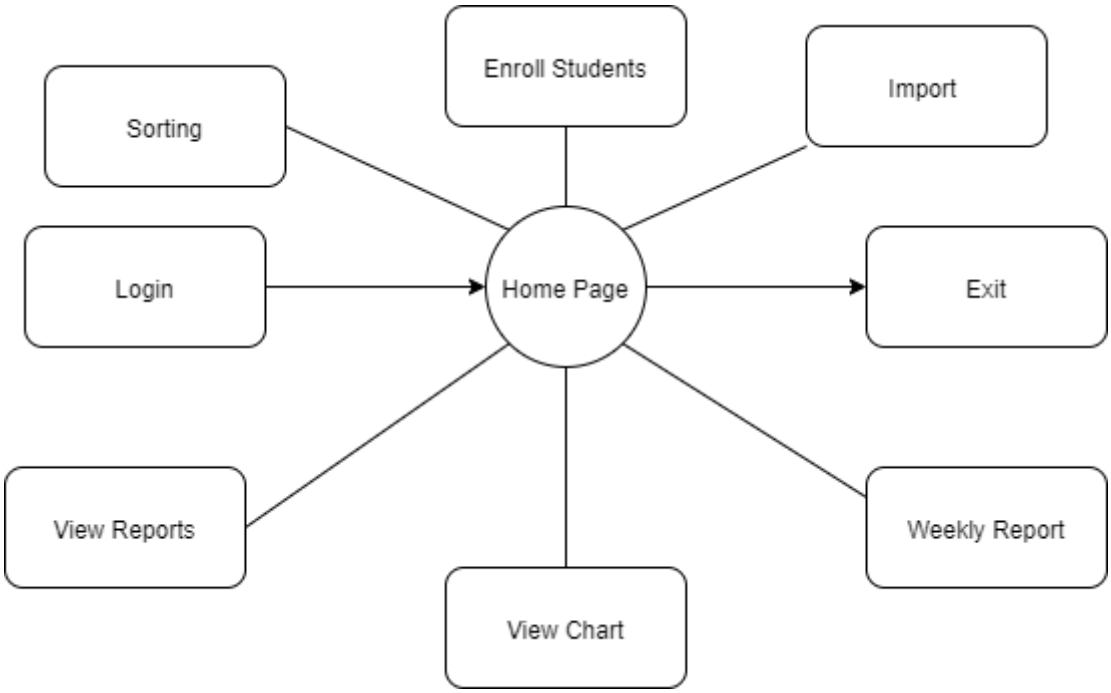


Figure 15: Architecture Diagram

Class Diagram

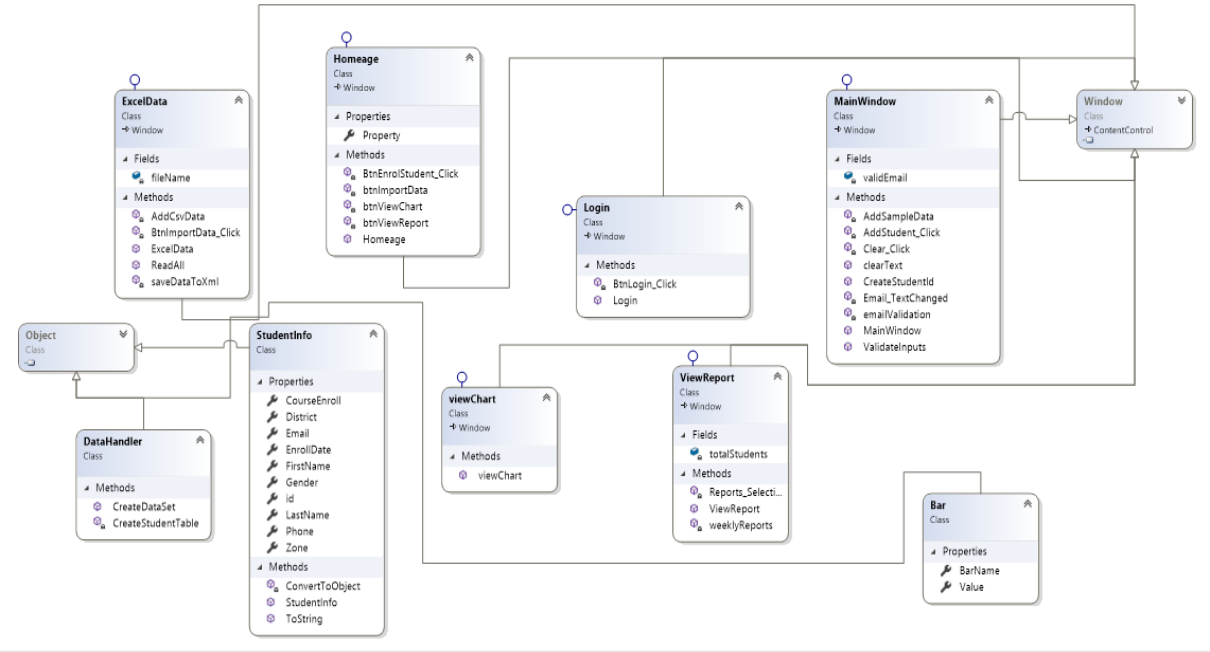
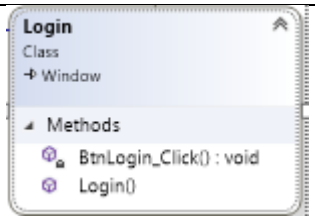


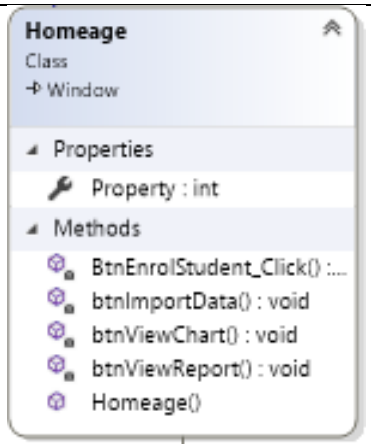
Figure 16: Class Diagram

Individual Diagram

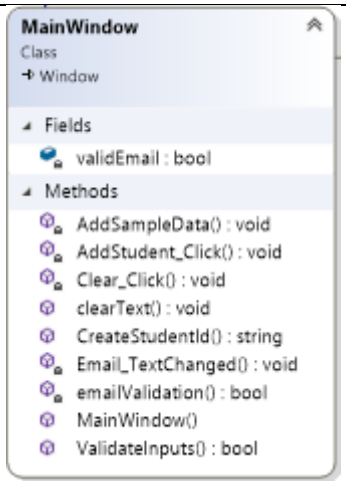
Login to the system.

Methods	Descriptions	Diagrams
btnLogin_click()	This method is used to validate the email and password. If the input is correct then user can enter into the Home page.	

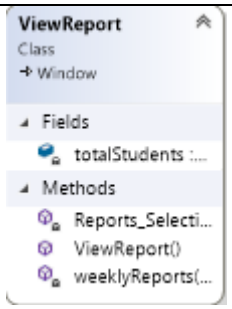
Home page

Methods	Descriptions	Diagrams
btnEnrolStudent_click()	This method is used to go in the enroll student page after clicking the Enroll student button.	
btnImportData()	This method is used to go in the Import csv page after clicking the Import data student button.	
btnViewReport()	This method is used to go in the enroll student page after clicking the Enroll student button.	
HomePage()	This is constructor of this class	

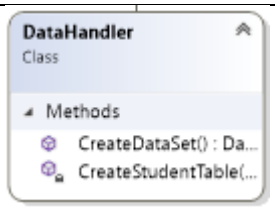
Enroll Students

Methods	Descriptions	Diagrams
AddSampleData()	This method takes all input of the UI and saves that data to the dataset.	
AddStudent_Click()	This method saves all students information in the xml.	
Clear_Click()	This method is used after clicking clear button where it sets all text values to empty string which is done by clear text method.	
Clear_Text()	This method set text values as empty string.	
CreateStudentId()	This method generates and returns string value for student id.	
Email_TextChanged	This method is used to compare email in text email.	
emailValidation()	This method is used to validate the email after inputting the text in text box email.	
ValidateInputs()	This method is used to validate all inputs in GUI and if the input data are in wrong format then it shows a dialog box with certain message.	

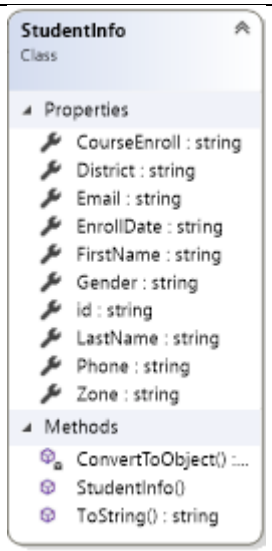
View Reports

Methods	Descriptions	Diagrams
Reports_selctionChanged()	This method is called after clicking the combo box button where it sorts the data grid values by comparing user name and date.	
ViewReport()	This method reads xml file and it shows all information in data grid.	
weeklyReports()	This method reads xml and compare all enrolled students and displays students enrolled this week only.	

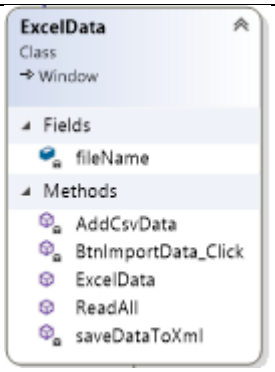
Data Handler Class

Methods	Descriptions	Diagrams
CreateDataSet()	This method saves students table in dataset and return dataset.	
CreateStudentTable()	This method creates new student table and add columns in it.	

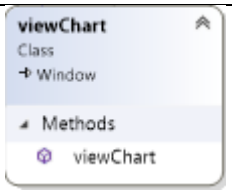
Student Information class

Methods	Descriptions	Diagrams
ConvertToObject()	This method takes one string parameter and split all string data with the help of comma.	
StudentInfo()	This is the constructor of this class and takes one string parameter which uses the methods of converToObject() method.	
ToString()	This is override method which returns all accessor methods of this class to string.	

Excel Data Class

Methods	Descriptions	Diagrams
AddCsvData()	This method takes Data set parameter and read all csv data and add it into the dataset.	
BtnImportData_Click()	This method is called after clicking the import data button which opens the file manager and takes csv file.	
ExcelData()	This is a constructor of this class which initialize all components.	
ReadAll()	This method returns an array list having data type of studentInfo class and reads all line of csv file.	
saveDataToXml()	This method checks an xml file, if there is no file then it generates new file and save all csv file data to xml file and if the file already exists then it adds data to the existing xml file.	

View Chart Class

Methods	Descriptions	Diagrams
ViewChart()	This is a constructor of View Chart class where it creates a list to insert bar in graph. Here it reads xml file in the mentioned location and pulls that data to the data grid and displays a graph according to that data.	

Flow chart for Enroll Students

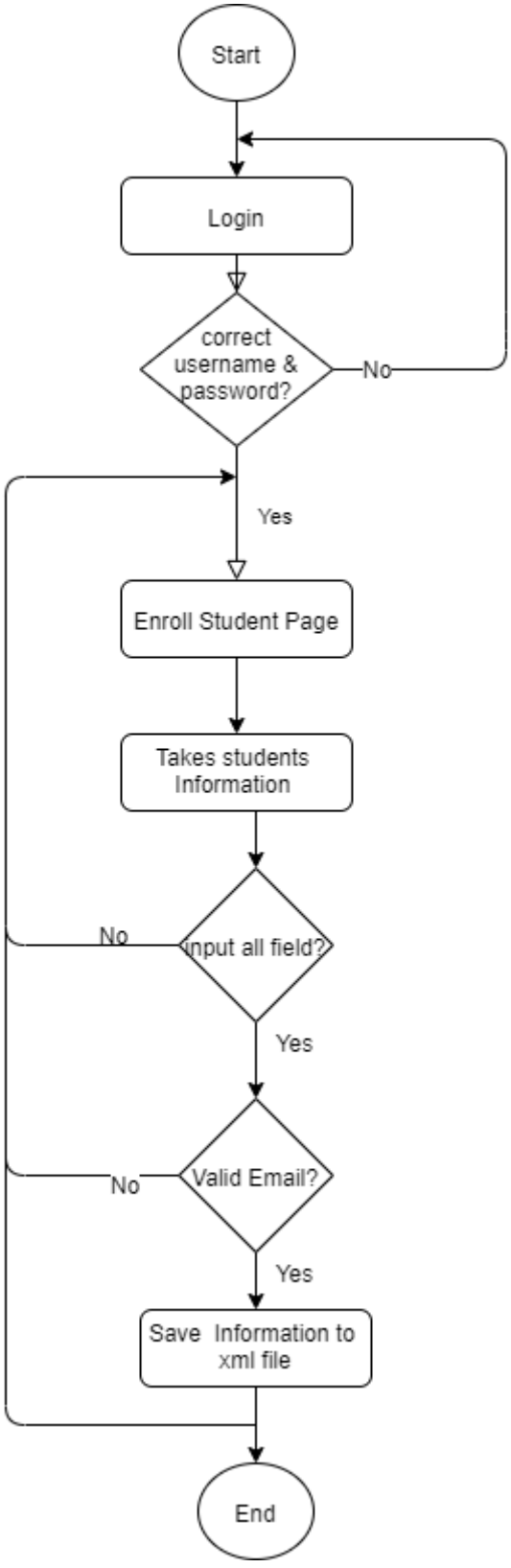


Figure 17: Flow chart for Enroll Students

Algorithms of Student Enrollment

1) Start
2) Open Login form
3) Checks email and password
4) If email and password match moves to Home page
5) If not matched then displays information about email and password.
6) Moves to Enroll student page after clicking enroll student button.
7) Takes student information from the end user.
8) Checks validation.
9) Saves all student information in Data Set.
10) Check whether the studentRecords file exists or not.
11) If not exists, then it creates a new xml file and saves data with the help of Data Set.
12) If exists, then it adds all student information in the same xml file.
13) Stop.

5 Sorting Algorithms

Bubble sort algorithm is used to sort the student's data in this student information system.
It is a sorting algorithm that works by repeatedly stepping through data lists that need to be sorted, comparing each pair of end-to-end items and swapping them if they are in the wrong order. It is repeated until no swaps are required, indicating that the list is sorted. Bubble sort gets its name because smaller elements bubble toward the top of the list.
The following example clearly explains the sorting process for this numbers 40.10,20,30,50

Step 1
It is an array of random numbers in unsorted format.

40	10	20	30	50

Step 2
Now it checks first two index and swap if the numbers are in wrong format

40	10	20	30	50

Step 3
Here 10 is smaller than 40 so that it swaps the value and puts number 10 in first index and 40 in second index.

10	40	20	30	50

Step 4 Now it compares number 40 and 20 where 20 is smaller so that it again swaps the value.

10	20	40	30	50

Step 5 Now it compares number 40 and 30 where 30 is smaller so again it swaps the value.

10	20	30	40	50

Now 40 is smaller than 50 which does not need to swap as a result all array numbers are sorted in ascending order.

10	20	30	40	50

In this way bubble sorting algorithm works.

6 Reflection

This is a student information system which is being developed with the help of Visual Studio 2019 in C# programming language. This system is easy to use because it is designed with highly user interface. This system can be used in colleges to record the information of students and to view total number of students in college.

Here this program has features like enrolling the students in the system where all inputs data are saved in the form of xml and can be viewed if it is needed. It provides the data secure because only a valid user can use this system by entering email and password in the login form. Here After complete login in the system the end user visits the home page where there are four different buttons called Enroll Student, View Chart, Import file and View Reports. Every button has different functions where enroll student button opens the new window where all student's information is entered and save in the xml file. If a user clicks import data button then it opens the window called import data where it has features like importing excel csv data by clicking the button called import. When a user clicks import file button then it opens file manager where we can import csv files only and after selecting csv file all data are save in the dataset and displayed in the table. To save this data in xml there is another button called save data which saves all data set values to xml file and can be easily viewed in report section if needed. If a user clicks a view report button then the new View Report page opens where all students are listed in the table format along with there all information that are recorded during the time of enrolling the students. IF a user wants to view students enrolled this week only then there is another table where students enrolled this week is displayed with the course name. Here we can view the student's information sort by name and date by clicking combo box button. To view information in graph we can click view chart button and see the total enrolled students along with the course.

I had no experience about using visual studio to develop a complete program in C# language but with the help of this course work and with the help of this application development course now I am more familiar with it. I had get knowledge about designing GUI for desktop application, using data grid with data source, importing csv file and creating chart from this coursework which will definitely help for my future career.

7 Conclusion

This course work was given for the Application Development module where it should be developed in visual studio. This course work was to develop student information system in given period. Where I had successfully finished it according to the requirement mentioned in the coursework. During the time of developing this system I had got so many troubles but with the help of my module leader and friends I had completed my work in the give period.

References

Dipin Budhrani1,Vivek Mulchandani,Yugchhaya Galphat. (2018). Student Information Management System. *Department of Computer Engineering, Vivekanand Education Society's Institute of Technology*, 10.

K. L. Clark, J. Darlington. (1980). Algorithm classification through synthesis. *The Computer Journal,*, 61-65.

S.R.Bharamagoudar. (2013). Web Based Student Information Management. *International Journal of Advanced Research in Computer and Communication Engineering*, 2348.

Xiangyang, T., 1993. FAST SORTING METHOD OF SEPARATING SEGMENT [J]. *Journal of Software*, *2*.

Burns, S. and Endress, W., 2014. Using C# and WPF to Create Fast Plots for Telemetry Analysis on Large Data Sets. International Foundation for Telemetering.

Good, N.A., 2005. CSV and Tab-Delimited Files. *Regular Expression Recipes for Windows Developers: A Problem-Solution Approach*, pp.127-154.

Wigley, A., Sutton, M., Wheelwright, S., Burbidge, R. and Mcloud, R., 2002. *Microsoft. net compact framework: Core reference*. Microsoft Press.

Appendix

Login.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 ApplicationDevelopmentCw1
{
    /// <summary>
    /// Interaction logic for Login.xaml
    /// </summary>
    public partial class Login : Window
    {
        public Login()
        {
            InitializeComponent();
        }


        private void BtnLogin_Click(object sender, RoutedEventArgs e)
        {
            var email = txtEmail.Text;
            var password = txtPassword.Text;
            if (email == "susheelgautam321@gmail.com" && password=="1234")
            {
                Homeage homepage = new Homeage();
                homepage.Show();
                this.Close();
            }
            else
            {
                MessageBox.Show("Enter susheelgautam321@gmail.com as Email and 1234 as
password", "Message", MessageBoxButton.OK, MessageBoxImage.Information);
            }
        }
    }
}
```

HomePage.cs

```
using Microsoft.Win32;
using System;
using System.Collections.Generic;
using System.Data;
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.Shapes;

namespace ApplicationDevelopmentCw1
{
    /// <summary>
    /// Interaction logic for Homeage.xaml
    /// </summary>
    public partial class Homeage : Window
    {

        public Homeage()
        {
            InitializeComponent();
        }

        public int Property
        {
            get => default;
            set
            {
            }
        }

        private void BtnEnrolStudent_Click(object sender, RoutedEventArgs e)
        {
            var myForm = new MainWindow();
            myForm.Show();
        }
        private void btnViewReport(object sender, RoutedEventArgs e)
        {
            var viewReport1 = new ViewReport();
            viewReport1.Show();
        }

        private void btnImportData(object sender, RoutedEventArgs e)
        {
            var importData = new ExcelData();
            importData.Show();
        }

        private void btnViewChart(object sender, RoutedEventArgs e)
        {
            var viewChart = new viewChart();
            viewChart.Show();

        }
```

```
    }
}
```

EnrollStudents.cs

```
using System;
using System.Collections.Generic;
using System.Data;
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;
using System.Text.RegularExpressions;
using System.Security.Cryptography;

namespace ApplicationDevelopmentCw1
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        bool validEmail;


        public MainWindow()
        {

            InitializeComponent();
            enrolDate.Text = DateTime.Now.ToString();
            registrationNo.Text = CreateStudentId().ToString();

        }

        private void AddStudent_Click(object sender, RoutedEventArgs e)
        {
            if (ValidateInputs())
            {
                var dataHandler = new DataHandler();
                var dataSet = dataHandler.CreateDataSet();
                AddSampleData(dataSet);

                if (File.Exists(@"E:\StudentRecords.xml") &&
File.Exists(@"E:\StudentRecordsSchema.xml"))
                {
                    dataSet.ReadXmlSchema(@"E:\StudentRecords.xml");
                    dataSet.ReadXml(@"E:\StudentRecords.xml");
                    dataSet.WriteXmlSchema(@"E:\StudentRecords.xml");
                    dataSet.WriteXml(@"E:\StudentRecords.xml");
                    MessageBox.Show("Enrolled Sucessfully", "Message", MessageBoxButton.OK,
MessageBoxImage.Information);
                    clearText();

                }
```

```
                else
                {
                    dataSet.WriteXmlSchema(@"E:\StudentRecordsSchema.xml");
                    dataSet.WriteXml(@"E:\StudentRecords.xml");
                    MessageBox.Show("Enrolled Sucessfully", "Message", MessageBoxButton.OK,
MessageBoxImage.Information);
                    clearText();
                }

            }
        }
        private void AddSampleData(DataSet dataSet)
        {
            var dr = dataSet.Tables["Student"].NewRow();
            dr["Id"] = Int32.Parse(registrationNo.Text);
            dr["FirstName"] = firstName.Text;
            dr["LastName"] = lastName.Text;
            dr["Gender"] = genderCombo.Text;
            dr["ContactNumber"] = contactNo.Text;
            dr["Email"] = email.Text;
            dr["CourseEnroll"] = courseEnrol.Text;
            dr["Zone"] = zone.Text;
            dr["District"] = district.Text;
            dr["RegistrationDate"] = DateTime.Now.ToString();
            dataSet.Tables["Student"].Rows.Add(dr);

        }
        public void clearText()
        {
            registrationNo.Text = "";
            firstName.Text = "";
            lastName.Text = "";
            email.Text = "";
            contactNo.Text = "";
            zone.Text = "";
            district.Text = "";

        }
        public Boolean ValidateInputs()
        {

            if (firstName.Text.Equals(""))
            {
                MessageBox.Show("First Name cannot be Empty", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
            }
            else if (lastName.Text.Equals(""))
            {
                MessageBox.Show("Last Name cannot be Empty", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
            }
            else if (email.Text.Equals(""))
            {
                MessageBox.Show("Email cannot be empty", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
            }
            else if (!validEmail)
            {
                MessageBox.Show("Email not valid", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
```

```
            }
            else if (district.Text.Equals(""))
            {
                MessageBox.Show("District cannot be Empty", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
            }
            else if (zone.Text.Equals(""))
            {
                MessageBox.Show("Zone cannot be Empty", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
            }
            else  if (contactNo.Text.Equals(""))
            {
                MessageBox.Show("Contact Number cannot be Empty", "Alert", MessageBoxButton.OK,
MessageBoxImage.Information);
                return false;
            }
            return true;
        }

        private void Clear_Click(object sender, RoutedEventArgs e)
        {
            clearText();
        }
        public string CreateStudentId()
        {
            var bytes = new byte[4];
            var randomNumber = RandomNumberGenerator.Create();
            randomNumber.GetBytes(bytes);
            uint random = BitConverter.ToUInt32(bytes, 0) % 100000000;
            return String.Format("{0:D8}", random);
        }
        private bool emailValidation(string email)
        {

            return new Regex(@"^[a-zA-Z][\w\.-]*[a-zA-Z0-9]@[a-zA-Z0-9][\w\.-]*[a-zA-Z0-9]\.[a-
zA-Z][a-zA-Z\.]*[a-zA-Z]$", RegexOptions.IgnoreCase).IsMatch(email);
        }

        private void Email_TextChanged(object sender, TextChangedEventArgs e)
        {
            validEmail = emailValidation(email.Text);
        }
    }
}
```

ExcelData.cs

```
using Microsoft.Win32;
using System;
using System.Collections.Generic;
using System.Data;
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.Shapes;

namespace ApplicationDevelopmentCw1
{
    /// <summary>
    /// Interaction logic for ExcelData.xaml
    /// </summary>
    public partial class ExcelData : Window
    {
        private string fileName;
        public ExcelData()
        {
            InitializeComponent();
        }

        private void BtnImportData_Click(object sender, RoutedEventArgs e)
        {
            OpenFileDialog open = new OpenFileDialog();
            //open.ShowDialog();
            // Set filter for file extension and default file extension
            open.DefaultExt = ".csv";
            open.Filter = "CSV files|*.csv";



            // Display OpenFileDialog by calling ShowDialog method
            Nullable<bool> result = open.ShowDialog();


            // Get the selected file name and display in a TextBox
            if (result == true)
            {
                // Open document
               fileName = open.FileName;

                var studenData = ReadAll();
                excelDataGrid.ItemsSource = studenData;
            }
        }
        public List<StudentInfo> ReadAll()
        {


                if (!File.Exists(fileName))
                {
                    throw new FileNotFoundException("Student Info file doesn't exist");
                }
                else
                {

                List<StudentInfo> students = new List<StudentInfo>();
                using (StreamReader streamReader = new StreamReader(fileName))
                {

                    while (!streamReader.EndOfStream)
                    {

                        var studentString = streamReader.ReadLine();
                        var studentInfo = new StudentInfo(studentString);
                        students.Add(studentInfo);
```

```
                }
                streamReader.Close();
            }
            return students;
        }



    }
    private void AddCsvData(DataSet dataSet)
    {
        var students = ReadAll();
        foreach (StudentInfo student in students)
        {
            var dr = dataSet.Tables["Student"].NewRow();
            dr["FirstName"] = student.FirstName;
            dr["LastName"] = student.LastName;
            dr["Gender"] = student.Gender;
            dr["ContactNumber"] = student.Phone;
            dr["Email"] = student.Email;
            dr["CourseEnroll"] = student.CourseEnroll;
            dr["Zone"] = student.Zone;
            dr["District"] = student.District;
            dr["RegistrationDate"] = student.EnrollDate;
            dataSet.Tables["Student"].Rows.Add(dr);
        }
    }

    private void saveDataToXml(object sender, RoutedEventArgs e)
    {
        var data = new DataHandler();
        var dataSet = data.CreateDataSet();
        AddCsvData(dataSet);

        if (File.Exists(@"E:\StudentRecords.xml") &&
File.Exists(@"E:\StudentRecordsSchema.xml"))
        {
            dataSet.ReadXmlSchema(@"E:\StudentRecords.xml");
            dataSet.ReadXml(@"E:\StudentRecords.xml");
            dataSet.WriteXmlSchema(@"E:\StudentRecords.xml");
            dataSet.WriteXml(@"E:\StudentRecords.xml");
            MessageBox.Show("Students Added Succsfully", "Message", MessageBoxButton.OK,
MessageBoxImage.Information);


        }
        else
        {
            dataSet.WriteXmlSchema(@"E:\StudentRecordsSchema.xml");
            dataSet.WriteXml(@"E:\StudentRecords.xml");
            MessageBox.Show("Students Added Succsfully", "Message", MessageBoxButton.OK,
MessageBoxImage.Information);

        }
    }
}
}
```

DataHandler.cs

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

namespace ApplicationDevelopmentCw1
{
    class DataHandler
    {
        public DataSet CreateDataSet()
        {
            var ds = new DataSet();
            ds.Tables.Add(CreateStudentTable());
            return ds;
        }

        private DataTable CreateStudentTable()
        {
            var dt = new DataTable("Student");


            dt.Columns.Add("Id", typeof(int));
            dt.Columns.Add("FirstName", typeof(string));
            dt.Columns.Add("LastName", typeof(string));
            dt.Columns.Add("ContactNumber", typeof(string));
            dt.Columns.Add("Email", typeof(string));
            dt.Columns.Add("CourseEnroll", typeof(string));
            dt.Columns.Add("Gender", typeof(string));
            dt.Columns.Add("RegistrationDate", typeof(DateTime));
            dt.Columns.Add("District", typeof(string));
            dt.Columns.Add("Zone", typeof(string));
            return dt;
        }


    }
}
```

StudentInfo.cs

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

namespace ApplicationDevelopmentCw1
{
    public class StudentInfo
    {


        public StudentInfo(string studentString)
        {
            this.ConvertToObject(studentString);
        }
        public string id { get; set; }
        public string FirstName { get; set; }
        public string LastName { get; set; }
        public string Phone { get; set; }
        public string Email { get; set; }
```

```
        public string CourseEnroll { get; set; }
        public string Gender { get; set; }
        public string EnrollDate { get; set; }
        public string District { get; set; }
        public string Zone { get; set; }



        public override string ToString()
        {
            return
$"{this.id}:{this.FirstName}:{this.LastName}:{this.Phone}:{this.Email}:{this.CourseEnroll}:{thi
s.Gender}:" +
                $"{this.EnrollDate}:{this.District}:{this.Zone}";
        }

        private void ConvertToObject(string studentString)
        {
            var splitedStrings = studentString.Split(',');
            this.id = splitedStrings[0];
            this.FirstName = splitedStrings[1];
            this.LastName = splitedStrings[2];
            this.Phone = splitedStrings[3];
            this.Email = splitedStrings[4];
            this.CourseEnroll = splitedStrings[5];
            this.Gender = splitedStrings[6];
            this.EnrollDate = splitedStrings[7];
            this.District = splitedStrings[8];
            this.Zone = splitedStrings[9];

        }

    }
}
```

ViewChart.cs

```
using System;
using System.Collections.Generic;
using System.Collections.ObjectModel;
using System.ComponentModel;
using System.Data;
using System.Linq;
using System.Reflection;
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 ApplicationDevelopmentCw1
{
    /// <summary>
    /// Interaction logic for viewChart.xaml
    /// </summary>
    public partial class viewChart : Window
    {
        public viewChart()
```

```
        {
            InitializeComponent();
            List<Bar> _bar = new List<Bar>();
            var dataset = new DataSet();
            dataset.ReadXml(@"E:\StudentRecords.xml");
            DataTable stdReports = dataset.Tables[0];
            int total_Computing = 0;
            int total_Networking = 0;
            int multi_media = 0;
            int AI = 0;

            DataTable dt = new DataTable("WeeklyData");
            dt.Columns.Add("Course Enroll", typeof(string));
            dt.Columns.Add("Total Students", typeof(int));
            for (int i = 0; i < stdReports.Rows.Count; i++)
            {
                string col = stdReports.Rows[i]["CourseEnroll"].ToString();

                if (col == "Computing")
                {
                    total_Computing++;
                }
                else if (col == "Multimedia Technologies")
                {
                    multi_media++;

                }
                else if (col == "Networking")
                {
                    total_Networking++;
                }
                else if (col == "Artificial Intelligence")
                {
                    AI++;
                }
            }
            dt.Rows.Add("Multimedia Technologies", multi_media);
            dt.Rows.Add("Computing", total_Computing);
            dt.Rows.Add("Networking", total_Networking);
            dt.Rows.Add("Artificial Intelligence", AI);
            _bar.Add(new Bar() { BarName = "Multimedia Technologies", Value = multi_media });
            _bar.Add(new Bar() { BarName = "Computing", Value = total_Computing });
            _bar.Add(new Bar() { BarName = "Networking", Value = total_Networking });
            _bar.Add(new Bar() { BarName = "Artificial Intelligence", Value = AI });
            this.DataContext = new RecordCollection(_bar);
            gridTotalstd.ItemsSource = dt.DefaultView;
        }

    }
    class Bar
    {

        public string BarName { set; get; }

        public int Value { set; get; }

    }
    class RecordCollection : ObservableCollection<Record>
    {

        public RecordCollection(List<Bar> barvalues)
        {
            Random rand = new Random();
            BrushCollection brushcoll = new BrushCollection();
```

```
            foreach (Bar barval in barvalues)
            {
                int num = rand.Next(brushcoll.Count / 3);
                Add(new Record(barval.Value, brushcoll[num], barval.BarName));
            }
        }

    }
    class BrushCollection : ObservableCollection<Brush>
    {
        public BrushCollection()
        {
            Type _brush = typeof(Brushes);
            PropertyInfo[] props = _brush.GetProperties();
            foreach (PropertyInfo prop in props)
            {
                Brush _color = (Brush)prop.GetValue(null, null);
                if (_color != Brushes.LightSteelBlue && _color != Brushes.White &&
                     _color != Brushes.WhiteSmoke && _color != Brushes.LightCyan &&
                     _color != Brushes.LightYellow && _color != Brushes.Linen)
                    Add(_color);
            }
        }
    }
    class Record : INotifyPropertyChanged
    {
        public Brush Color { set; get; }

        public string Name { set; get; }

        private int _data;
        public int Data
        {
            set
            {
                if (_data != value)
                {
                    _data = value;

                }
            }
            get
            {
                return _data;
            }
        }

        public event PropertyChangedEventHandler PropertyChanged;

        public Record(int value, Brush color, string name)
        {
            Data = value;
            Color = color;
            Name = name;
        }

        protected void PropertyOnChange(string propname)
        {
            if (PropertyChanged != null)
            {
                PropertyChanged(this, new PropertyChangedEventArgs(propname));
            }
        }
```

```
    }
}
```

ViewReport.cs

```
using System;
using System.Collections;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
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.Shapes;

namespace ApplicationDevelopmentCw1
{
    /// <summary>
    /// Interaction logic for ViewReport.xaml
    /// </summary>
    public partial class ViewReport : Window
    {
        List<StudentInfo> totalStudents = new List<StudentInfo>();
        public ViewReport()
        {
            InitializeComponent();
            var dataHandler = new DataHandler();
            var dataSet = dataHandler.CreateDataSet();
            if (File.Exists(@"E:\StudentRecords.xml") &&
File.Exists(@"E:\StudentRecordsSchema.xml"))
            {
                dataSet.ReadXml(@"E:\StudentRecords.xml");
                reportGrid.ItemsSource = new DataView(dataSet.Tables["Student"]);
                weeklyReports();
            }
            else
            {
                MessageBox.Show("There are no students to show", "Message",
MessageBoxButton.OK, MessageBoxImage.Information);

            }


        }

        private void Reports_SelectionChanged(object sender, SelectionChangedEventArgs e)
        {
            var sortName = reports.SelectedIndex;
            if (sortName==1)
            {
                reportGrid.Items.SortDescriptions.Clear();
                reportGrid.Items.SortDescriptions.Add(new SortDescription("FirstName",
ListSortDirection.Ascending));
                reportGrid.Items.Refresh();
            }
            else
            {
```

```
                reportGrid.Items.SortDescriptions.Clear();
                reportGrid.Items.SortDescriptions.Add(new SortDescription("RegistrationDate",
ListSortDirection.Descending));
                reportGrid.Items.Refresh();
            }

        }
        private void weeklyReports()
        {

            var dataset = new DataSet();
            dataset.ReadXml(@"E:\StudentRecords.xml");
            DataTable stdReports = dataset.Tables[0];
            int total_Computing = 0;
            int total_Networking = 0;
            int multi_media = 0;
            int AI = 0;

            DataTable dt = new DataTable("WeeklyData");
            dt.Columns.Add("Course Enroll",typeof(string));
            dt.Columns.Add("Total Students",typeof(int));
            //dt.Columns.Add("Date", typeof(DateTime));
            for (int i = 0; i < stdReports.Rows.Count; i++)
            {
                string col = stdReports.Rows[i]["CourseEnroll"].ToString();
                string date = stdReports.Rows[i]["RegistrationDate"].ToString();
                DateTime myDate = DateTime.Parse(date);
                double thisWeek = (DateTime.Today - myDate).TotalDays;
                if (col=="Computing" && thisWeek<=7)
                {
                    total_Computing++;
                }
                else if(col== "Multimedia Technologies" && thisWeek <= 7)
                {
                    multi_media++;

                }
                else if (col == "Networking" && thisWeek <= 7)
                {
                    total_Networking++;
                }
                else if (col == "Artificial Intelligence" && thisWeek <= 7)
                {
                    AI++;
                }
            }
            dt.Rows.Add("Multimedia Technologies", multi_media);
            dt.Rows.Add("Computing", total_Computing);
            dt.Rows.Add("Networking", total_Networking);
            dt.Rows.Add("Artificial Intelligence", AI);
            weeklyReport.ItemsSource = dt.DefaultView;
        }
    }
}
```