

# Informatics College Pokhara



informatics  
college pokhara

**Application Development**

**CS6004NI**

**Course Work 1**

**Submitted By: Nirakar Sigdel**  
**London Met ID:** Enter ID Here

**Submitted To:** Ishwor Sapkota  
Module Leader

Component Grade and Comments	
<b>A. Implementation of Application</b>	
<b>User Interface and proper controls used for designing</b>	missing controls in the interface
<b>Manual data entry or import from csv</b>	appropriate use of data types but missing some properties required or missing CRUD operation
<b>Data Validation</b>	Only basic validation
<b>Enrollment Report &amp; weekly report in tabular format</b>	very poorly executed reports and data not shown accurately
<b>Course wise enrollment report &amp; Chart display</b>	Very poorly designed and only contains one report format with in appropriate data
<b>Algorithm used for sorting &amp; proper sorting of data</b>	Default sorting provided by .net is used
<b>B. Documentation</b>	
<b>User Manual for running the application</b>	User Manual is below average. Is textual only.

# Marking Scheme

<b>Application architecture &amp; description of the classes ad methods sued</b>	average work with very limited explanation of the classes and methods used
<b>Flow chart, algorithms and data sctructures used</b>	average work with very limited explanation and missing diagramatic representation.
<b>Reflective essay</b>	Very poorly written

## C. Programming Style

<b>Clarity of code,Popper Naming convention &amp; comments</b>	very poorly written code and no comments at all
<b>System Usability</b>	very poorly developed application

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

## Overall Comment:

Code should be self explainable with less comments. Need some proper naming of the componen and require to add comments on required area. Missing some feature

OK to go for.

## **CS6004NA – Application Development**

### **Assessment Weightage & Type**

**30% Individual Coursework**

### **Year and Semester**

**2019-20 Autumn**

**Name: Nirakar Sigdel**

**College ID: NP04CP4S180009**

**University ID: 17031944**

## **Abstract**

This coursework is about developing and implementing a C# desktop application that helps to manage the student's record. This application will help to keep track of each student detail with date duration of enrollment. With the help of this application, Staff can also check students' weekly report. The chart for the report of the student will also be generated. Thus, with the help of user-friendly GUI, staff can easily implement this application.

## Table of Contents

<b>1. Introduction .....</b>	<b>1</b>
1.1. Current Scenario.....	1
1.2. Proposed System .....	1
<b>2. User Manual .....</b>	<b>2</b>
<b>3. Journal Articles .....</b>	<b>9</b>
<b>4. Classes, Properties and Methods .....</b>	<b>12</b>
<b>5. Sorting Algorithm .....</b>	<b>17</b>
<b>6. Reflection .....</b>	<b>20</b>
<b>7. Problem Experienced.....</b>	<b>21</b>
<b>8. Solution .....</b>	<b>22</b>
<b>9. Conclusion.....</b>	<b>23</b>
<b>10. Bibliography.....</b>	<b>24</b>

## List of Figures

Figure 1: Login Screen.....	2
Figure 2: Error Login .....	3
Figure 3: MainWindow .....	3
Figure 4: Saving record of Student .....	4
Figure 5: Importing to DataGridView.....	4
Figure 6: Exception Handling of data entry .....	5
Figure 7: Display Report Window.....	5
Figure 8: Retrieved data.....	6
Figure 9: Sorting by name.....	6
Figure 10: Sorting by date.....	7
Figure 11: Weekly Report .....	7
Figure 12: Generated Pie-chart.....	8
Figure 13: Flowchart of Weekly Report.....	15

## List of Table

Table 1: Method description.....	14
----------------------------------	----

## **1. Introduction**

This coursework is about developing and implementing a C# desktop application that helps school to manage the student's record. This application helps the school staff to input the student's basic detail manually such as Name, Email, Contact and system automatically import that detail to a CSV file. The enrollment date and course chosen by student is also managed by the system. The system generates daily and weekly report and sort them by the Enrollment date and Name.

### **1.1. Current Scenario**

Large number of schools has their own software to keep the record of students but some of them are still keeping their record in old traditional system which is Paper-Based System. Despite many of schools having the software, they are not updated and lack the features needed for Schools.

### **1.2. Proposed System**

The proposed system is digitized system which is specially designed to overcome the issues mentioned. The system ensures security with presence of login section. With the use of Graphical User Interface, entry of data and display of data has been made easier.



## 2. User Manual

The following screenshots show the proper instruction how to operate the system. As the end user operates the system the initial screen will be the security screen. The username and password of the system is “nirakar”. Valid credentials can only provide access to the system.

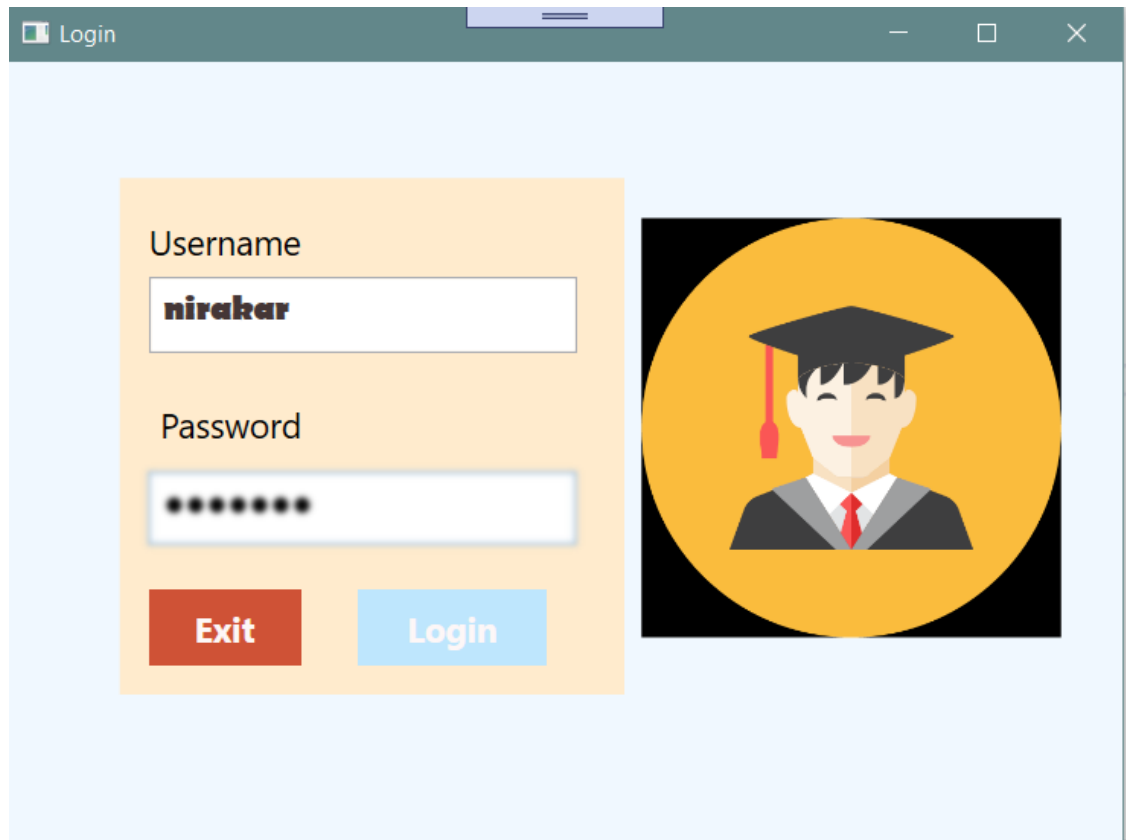


Figure 1: Login Screen

If the credentials didn't match then the error is displayed as in the screenshot below:

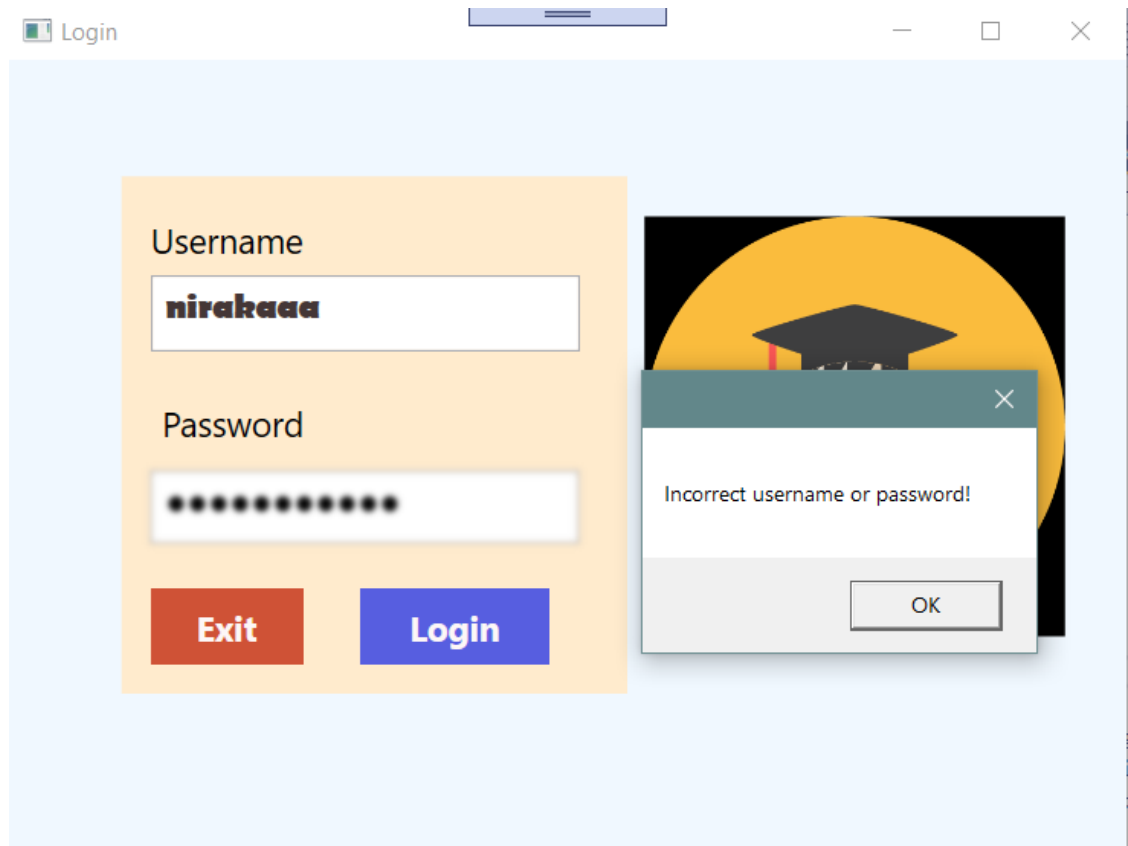


Figure 2: Error Login

After the successful login, the Main Window gets opened.

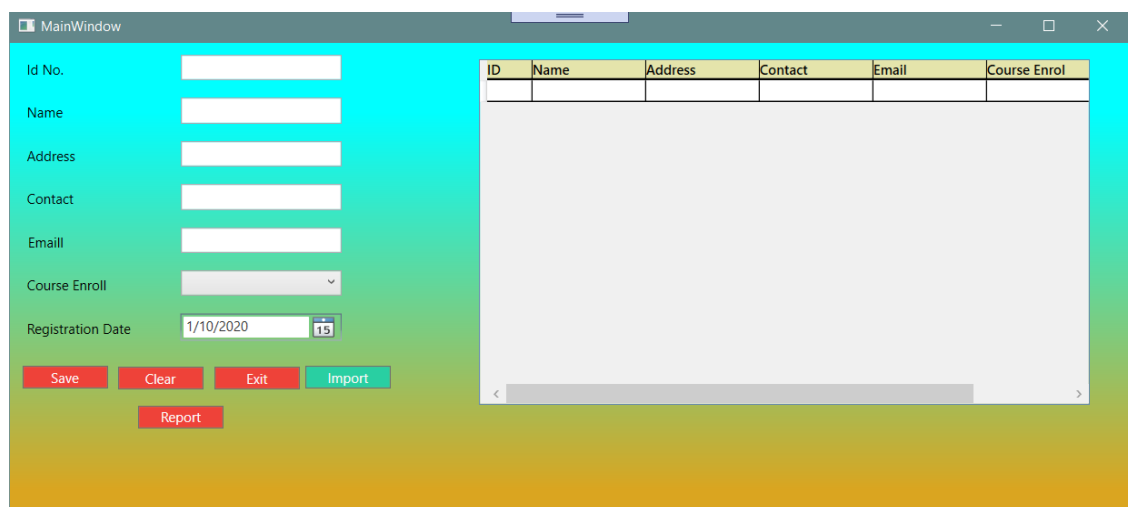


Figure 3: MainWindow

Entering student record and then saving it.

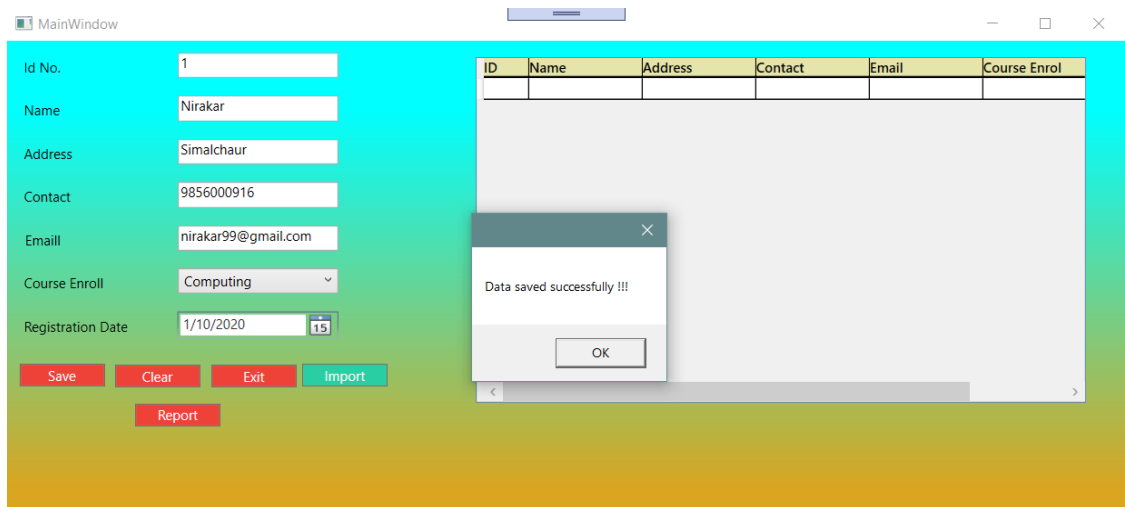


Figure 4: Saving record of Student

Importing the data to data grid.

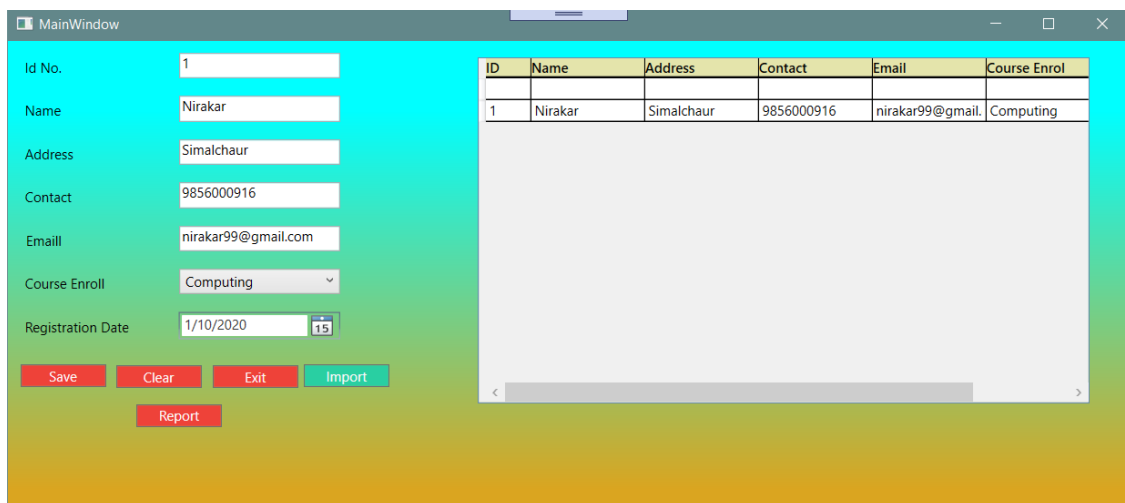


Figure 5: Importing to DataGrid

Handling exception while empty or wrong input is given in the form.

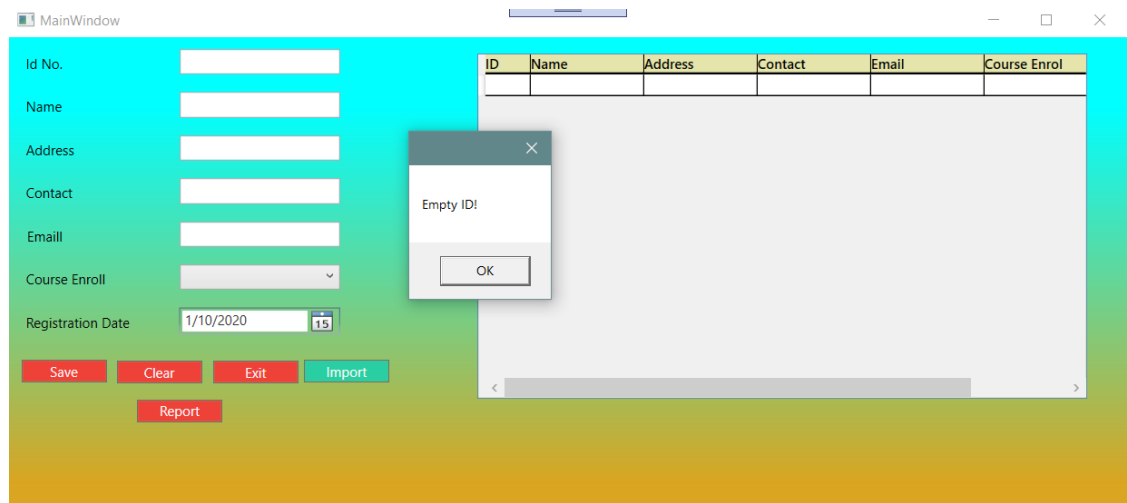


Figure 6: Exception Handling of data entry

In the Report button present in the Main Window, Display Report gets opened which has five more buttons.

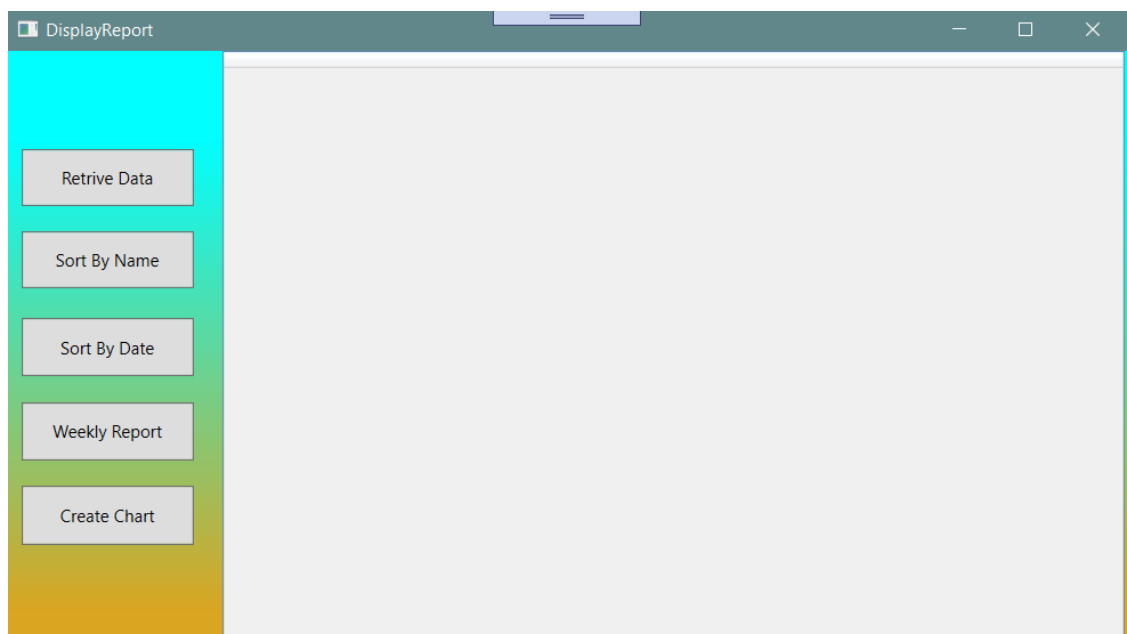


Figure 7: Display Report Window

Retrieve Data button retrieves the data from the data grid of Main Window.

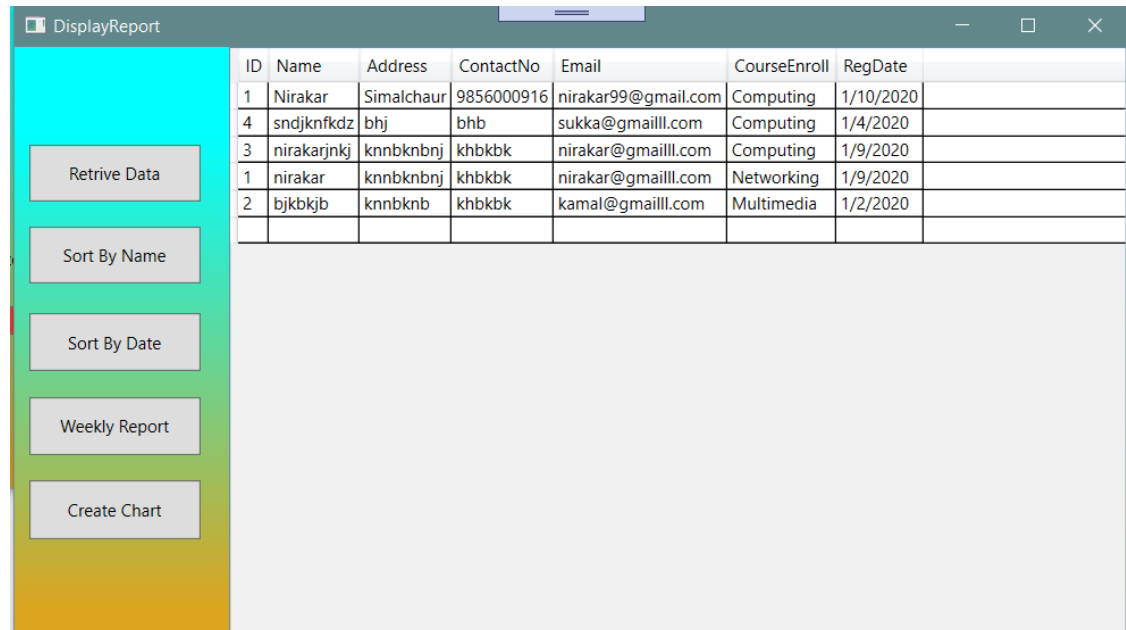


Figure 8: Retrieved data

Sorting the data according to name by clicking the Sort by Name button is done below:

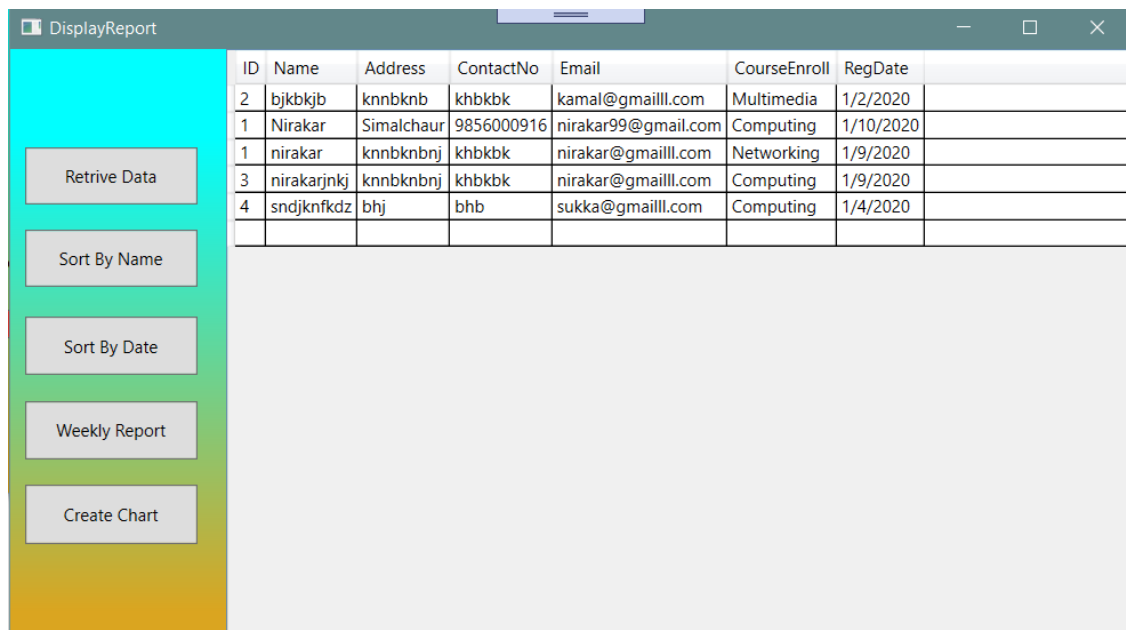
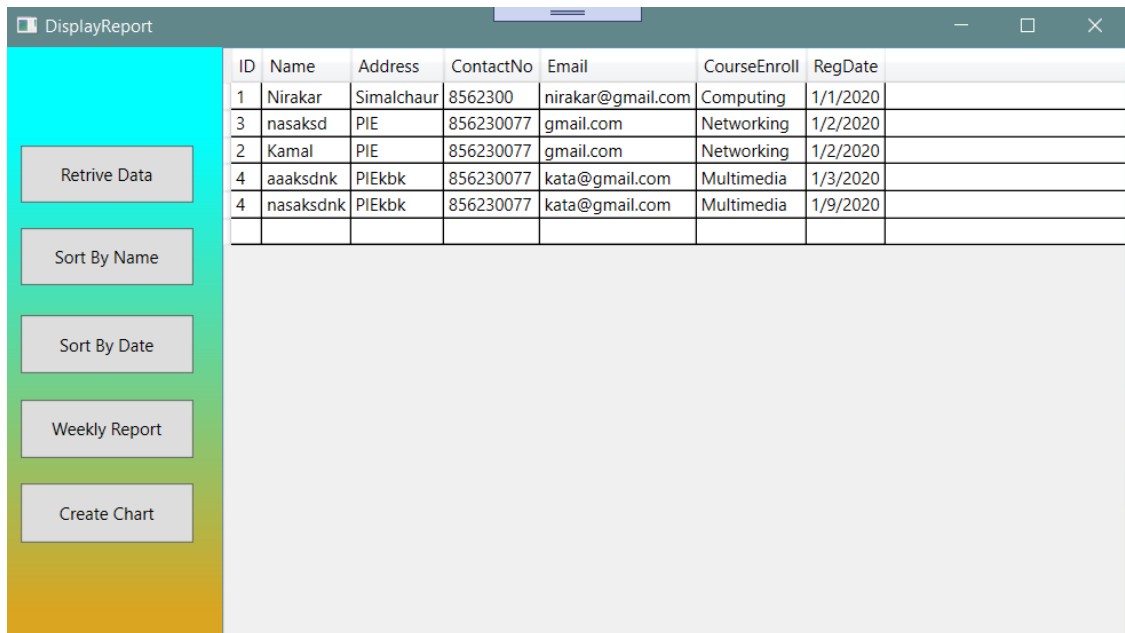


Figure 9: Sorting by name

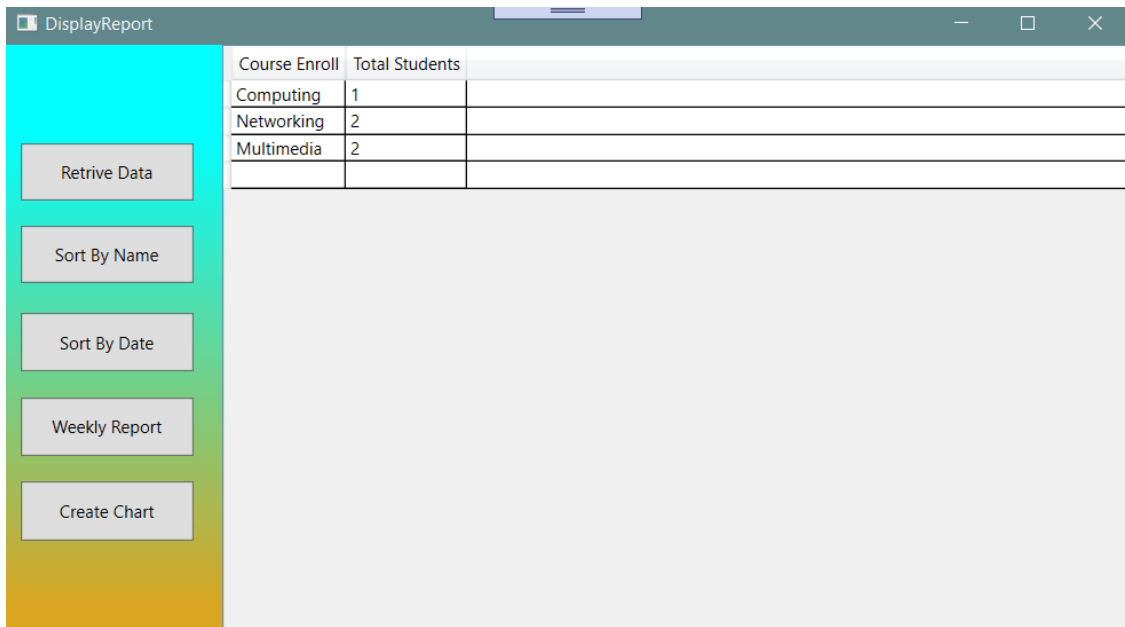
Sorting the data according to name by clicking the Sort by date:



ID	Name	Address	ContactNo	Email	CourseEnroll	RegDate
1	Nirakar	Simalchaur	8562300	nirakar@gmail.com	Computing	1/1/2020
3	nasaksd	PIE	856230077	gmail.com	Networking	1/2/2020
2	Kamal	PIE	856230077	gmail.com	Networking	1/2/2020
4	aaaksdnk	PIEkbk	856230077	kata@gmail.com	Multimedia	1/3/2020
4	nasaksdnk	PIEkbk	856230077	kata@gmail.com	Multimedia	1/9/2020

Figure 10: Sorting by date

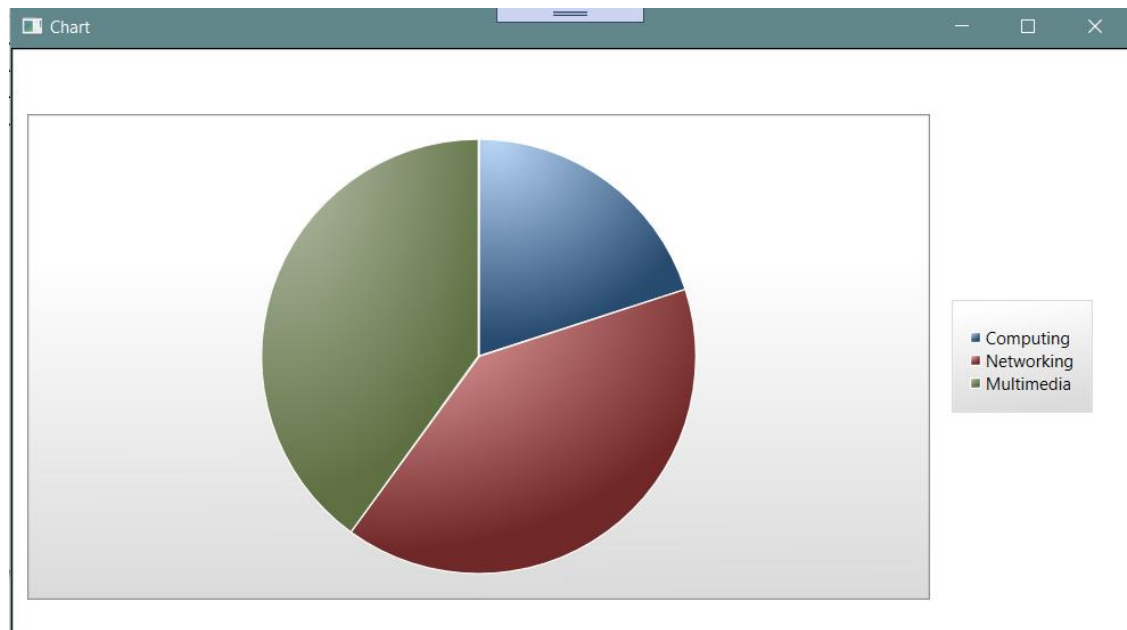
After clicking the Weekly Report button, we can see the total number of students enrolled in each Course. It is shown in the following screenshot.



Course Enroll	Total Students
Computing	1
Networking	2
Multimedia	2

Figure 11: Weekly Report

Create chart button creates a pie-chart creating new window as shown in the figure below:



*Figure 12: Generated Pie-chart*

### 3. Journal Articles

- i. The management and provision of information about the educational process is an essential part of effective management of the educational process in the institutes of higher education. In this paper the requirements of a reliable student management system are analyzed, formed a use-case model of student information management system, designed and implemented the architecture of the application. Regarding the implementation process, modern approaches were used to develop and deploy a reliable online application in cloud computing environments specifically (Alameri, 2017).
- ii. Despite the benefits of school management information systems (SMIS), the concept of data-driven school culture failed to materialize for many educational institutions. Challenges posed by the quality of data in the big data era have prevented many schools from realizing the real potential of the SMIS. The paper analyses the uses, features, and inhibiting factors of SMIS. The paper proposes a five-phase conceptual model that assist administrators with making timely, quality decisions. The paper enriches the theoretical landscape of SMIS usage in the era of big data and lays a foundation for the future by establishing an educational decision-making model (Forrester, 2019).
- iii. Most of the Academic institutions face difficulty in managing records of students, attendance, accounts, admissions, etc., and track the information of their interest as they still rely on paperwork and manual processes. A web-based school management system will reduce the manual work by deploying centralized software incorporated with various loosely coupled services which interact with each other to address above mentioned issues and improves the communication between management and student/guardian



through notifications via email, SMS and push messages. As it is a server-side enterprise application it is designed to support desktop browsers, mobile browsers and native mobile applications. The use of micro-service architecture and ReST (Representational State Transfer) architecture makes it easy for designing and developing loosely coupled web services, (K, 2019).

- iv. System usability is one of the key elements that should be focused on, especially during the design and test phases of a system, because it provides feedback to system administrators in order to improve the system. In the literature, System Usability Scale (SUS) is widely used as the gold standard method to evaluate system usability. Today, machine learning, which is one of the subfields of artificial intelligence, also provide new perspectives on the evaluation of system usability. In this study, it is aimed to predict usability of a Student Information System (SIS) by using machine learning techniques. In the study method, the Cross-Industry Standard Process for Data Mining (CRISP-DM) steps have been followed. Analysis are performed on two different datasets namely “sus0” and “sus1”. “sus0” dataset is consisted of demographic characteristics (age, gender, department) of 324 students using a SIS of a foundation university in Turkey, also their responses to the Turkish version of the SUS (SUS-TR). “sus 1” includes only responses to the SUS-TR. C4.5 Decision Tree Algorithm, Naive Bayes Classifier and k-Nearest Neighbor Algorithm are used to create models and performance of the models are evaluated. In the analysis with 80% to 20% hold-out method, the best performance was obtained on the “sus0” data set with k-Nearest Neighbor Algorithm (accuracy= 0.698, F-measure = 0.796 for k = 20) (Kartal, 2019).
- v. The mission of the Student Information Management system is to create an integrated information technology environment for students, HOD, faculty, staff and administration. Our goal is to

focus on services and integration for end users. It is a web-based self-service environment for students, prospective students, and employees; administrative transaction processing environment for yearly admissions; an informative environment for all levels of faculty and staff to do reporting, data extraction and information analysis. It is mainly useful and used for educational establishments to manage student data which also facilitates all individual associated information for easier navigation on daily basis. It provides capabilities for entering student test and other assessment scores, building student schedules, tracking student attendance and managing many other student-related data needs in a college. Our easy-to-use, integrated college administration application would be used to reduce time spent on administrative tasks, as to concentrate on other skillful practical activities other than book worming. It can accept, process and generate reports at any given point of time accurately (Budhrani, 2018).

## 4. Classes, Properties and Methods

### Classes and Properties

This application consists of the following main classes:

**Login.xaml:** This is the main page of the desktop application. After running the application Login.xaml class is called and the login form is displayed where the user enters the username and password in the respective text-box. The user has to login to enter the main page of desktop application and if the user leaves any text-boxes empty or wrong username or password then the pop-up message will be displayed.

**MainWindow.xaml:** After logging into the system, MainWindow.xaml class is called. In this class the tabs controller is used for changing the tabs i.e. Save, Import, Report. In the Save Button, the student's details manually entered by user is saved in XML file and the Import Button shows the details of students in data grid view. The Report button opens the new window DisplayReport.xaml. The Clear button clears the form and Exit button exits the window.

**DisplayReport.xaml:** After clicking on the Report button of MainWindow, this class gets opened where five buttons i.e. Retrieve Data, Sort by Name, Sort by Date, Weekly Report and Create chart; retrieves data, Sort by Name and Date, generates the weekly report and creates the chart respectively.

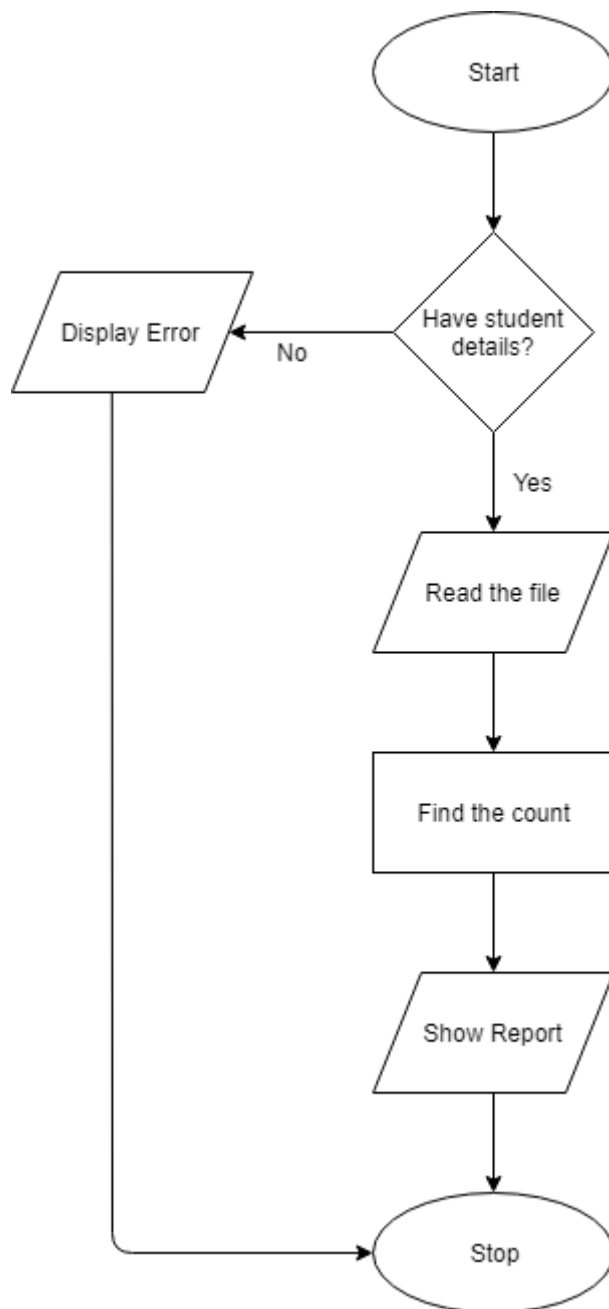
## Methods

This application consists of following methods:

Methods	Description
<code>private void Button_Click(object sender, RoutedEventArgs e)</code>	When Button_Click is clicked, it will login into the application; only if the admin and password matches.
<code>private void Exit_Click(object sender, RoutedEventArgs e)</code>	When Exit_Click is clicked, it will close the application.
<code>private void btnSave_Click(object sender, RoutedEventArgs e)</code>	When btnSave_Click is clicked, it saves the data in XML file.
<code>private void AddSampleData(DataSet dataSet)</code>	It adds the sample data.
<code>private void btnImport_Click(object sender, RoutedEventArgs e)</code>	It imports the data into data grid.
<code>private void btnClear_Click(object sender, RoutedEventArgs e)</code>	It clears the uncleared data in the student record form.
<code>private void btnExit_Click(object sender, RoutedEventArgs e)</code>	It exits the window.
<code>private void btnReport_Click(object sender, RoutedEventArgs e)</code>	It opens another window naming DisplayReport
<code>private void buttonRetrive_Click(object sender, RoutedEventArgs e)</code>	It retrieves the data and shows in the data grid of DisplayReport
<code>private void buttonSName_Click(object sender, RoutedEventArgs e)</code>	It sorts the data according to Name in ascending order.

<code>private void buttonSD_Click(object sender, RoutedEventArgs e)</code>	It sorts the data according to Name in ascending order.
<code>private void buttonWeekly_Click(object sender, RoutedEventArgs e)</code>	It shows the total number of students added in respective course in a week.
<code>private void buttonChart_Click(object sender, RoutedEventArgs e)</code>	It leads to the new window to show the pie-chart.

*Table 1: Method description*

**Flowchart for Report***Figure 13: Flowchart of Weekly Report*

**Algorithm of Report**Steps

1. Start
2. Check if student details is there or not.
3. If No, display error message and exit.
4. If Yes, Read the file.
5. Find the count.
6. Display Count.
7. Stop

## 5. Sorting Algorithm

Bubble sort is a simple sorting algorithm which works on the comparison-based algorithm. In this algorithm, the adjacent elements are compared and the elements are swapped if they are not in order. This algorithm is usually 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.

### Working Methods:

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



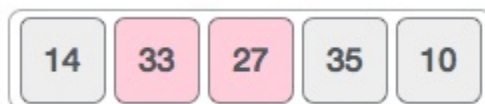
Bubble sort starts with very first two elements, comparing them to check which one is greater.



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.



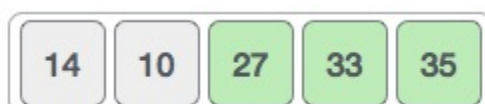
We swap these values. We find that we have reached the end of the array. After one iteration, the array should look like this –



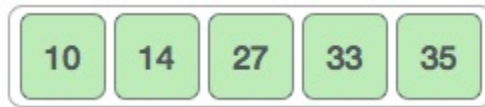
To be precise, we are now showing how an array should look like after each iteration. After the second iteration, it should look like this –



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.



So, with the comparison between the elements, the elements are finally sorted. (TutorialsPoint, n.d.)

## 6. Reflection

This coursework is about developing desktop application in C# for Student Information. I got knowledge about C# in my module, Application Development. It was my first-time using C# language for developing a desktop application. As it was my first-time using C#, I had many confusions but with the help of module leader, I was able to develop desktop application using C#. C# is an object oriented and general-purpose programming language created and developed by Microsoft. As C# is a high-level language, it is similar to Java and C++. Since, it has resemblance with Java, several methods were easy to develop.

The desktop application is developed in Microsoft Visual Studio which is an Integrated Development Environment (IDE). As I had used Microsoft Visual Studio previously, I felt less issues regarding working on it but still had some problems which got solved during tutorial classes and I was able to develop the application. While working for this project I found Microsoft Visual Studio much easier to implement an application as it automatically analyzes code to point out errors and offer suggestions.

## **7. Problem Experienced**

While developing this desktop application, many complications were faced. With the help of tutorial classes, it became easy to start the project. There was a problem working with toolbox and solution explorer for developing GUI for this application. Many errors were faced while importing CSV file. Generating report and showing the pie chart was a challenging part of this application.

## **8. Solution**

The problem faced while developing this project was solved with the help of various websites and video tutorials. Login and generating CSV file was done with the help of example, which was done in classroom. The lecture and tutorial classes helped me in solving errors and to work effectively in Visual Studio Platform. I performed many researches, to develop pie-chart. Thus, in this way, I completed this coursework successfully.

## **9. Conclusion**

As per the scenario, a desktop application for Student Information was developed with the help of C# programming language. Those codes were validated and is successfully working though several problems occurred while developing this application. Through this coursework, it helped me understand C# which will help me in future. This project also helped me more about the desktop application. As this project is successfully developed and tested and has user friendly GUI, this project can be easily used in school in order to save student details more efficiently.

## 10. Bibliography

Alameri, I., 2017. *Development of Student Information Management System based on Cloud Computing Platform*. [Online]

Available at:

[https://www.researchgate.net/publication/319881143\\_Development\\_of\\_Student\\_Information\\_Management\\_System\\_based\\_on\\_Cloud\\_Computing\\_Platform](https://www.researchgate.net/publication/319881143_Development_of_Student_Information_Management_System_based_on_Cloud_Computing_Platform)

[Accessed 10 01 2020].

Budhrani, D., 2018. *Student Information Management System*. [Online]

Available at: <https://www.ijedr.org/papers/IJEDR1801002.pdf>

[Accessed 10 01 2020].

Forrester, V., 2019. *School Management Information Systems*. [Online]

Available at:

[https://www.academia.edu/38700223/SCHOOL\\_MANAGEMENT\\_INFORMATION\\_SYSTEMS\\_CHALLENGES\\_TO\\_EDUCATIONAL\\_DECISION-MAKING\\_IN\\_THE\\_BIG\\_DATA\\_ERA](https://www.academia.edu/38700223/SCHOOL_MANAGEMENT_INFORMATION_SYSTEMS_CHALLENGES_TO_EDUCATIONAL_DECISION-MAKING_IN_THE_BIG_DATA_ERA)

[Accessed 10 01 2020].

Kartal, E., 2019. *Predicting Usability of a Student Information System by Using Machine Learning Techniques*. [Online]

Available at:

[https://www.academia.edu/39894115/Predicting\\_Usability\\_of\\_a\\_Student\\_Information\\_System\\_by\\_Using\\_Machine\\_Learning\\_Techniques](https://www.academia.edu/39894115/Predicting_Usability_of_a_Student_Information_System_by_Using_Machine_Learning_Techniques)

[Accessed 10 01 2020].

K, S., 2019. *Student Management System*. [Online]

Available at: <https://www.ijert.org/student-management-system>

[Accessed 10 01 2020].

TutorialsPoint, n.d. *Bubble Sort Algorithm*. [Online]

Available at:

[https://www.tutorialspoint.com/data\\_structures\\_algorithms/bubble\\_sort\\_algorithm.htm](https://www.tutorialspoint.com/data_structures_algorithms/bubble_sort_algorithm.htm)

[Accessed 10 01 2020].

## Appendix

### Login.xaml.cs

```

        using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

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

        private void Button_Click(object sender, RoutedEventArgs e)
        {
            string username = textBox1.Text;
            string password = textBox2.Password;

            if (username == "")
            {
                MessageBox.Show("Username cannot be empty!");
            }
            else if (password == "")
            {
                MessageBox.Show("Password cannot be empty!");
            }
            else if (password == "nirakar" && username == "nirakar")
            {
                this.Hide();
                MainWindow mainWindow = new MainWindow();
                mainWindow.Show();
            }

            else
            {
                MessageBox.Show("Incorrect username or password!");
            }
        }

        private void Exit_Click(object sender, RoutedEventArgs e)
        {
            MessageBox.Show("Window is being exited.");
            this.Close();
        }
    }
}

```



**MainWindow.xaml.cs**

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

namespace ApplicationDevelopmentCW
{
    public partial class MainWindow : Window
    {
        public object Utility { get; }

        public MainWindow()
        {
            InitializeComponent();

            Student student = new Student();

            DataGridXAML.Items.Add(student);
        }
        public class Student
        {
            public string ID
            {
                get;
                set;
            }

            public string Name { get; set; }

            public string Address { get; set; }

            public string Contact { get; set; }

            public string CourseEnroll { get; set; }

            public string RegDate { get; set; }

            public string Email { get; internal set; }
        }
        private void btnSave_Click(object sender, RoutedEventArgs e)
        {

```

```

//empty input validation
if (textId.Text == "")
{
    MessageBox.Show("Empty ID!");
}
else if (textName.Text == "")
{
    MessageBox.Show("Name is required");
}

else if (textAddress.Text == "")
{
    MessageBox.Show("Empty Address!");
}
else if (textContact.Text == "")
{
    MessageBox.Show("Empty Contact Number!");
}
else if (textEmail.Text == "")
{
    MessageBox.Show("Empty email id!");
}

else if (textCourse.Text == "")
{
    MessageBox.Show("Invalid Course!");
}

else
{
    var handler = new DataHandler();
    var dataSet = handler.CreateDataSet();
    AddSampleData(dataSet);
    MessageBox.Show("Data saved successfully !!!");
    if (File.Exists(@"D:\student.xml"))
    {
        dataSet.ReadXml(@"D:\student.xml");
        dataSet.WriteXml(@"D:\student.xml");
    }
    else
    {
        dataSet.WriteXml(@"D:\student.xml");
    }
}

private void AddSampleData(DataSet dataSet)
{
    var dr1 = dataSet.Tables["Student"].NewRow();
    dr1["ID"] = textId.Text;
    dr1["Name"] = textName.Text;
    dr1["Address"] = textAddress.Text;
}

```

```
dr1["Contact"] = textContact.Text;
dr1["Email"] = textEmail.Text;
dr1["CourseEnroll"] = textCourse.Text;
string text = textDate.Text;
dr1["RegDate"] = text;
dataSet.Tables["Student"].Rows.Add(dr1);
}

private void btnImport_Click(object sender, RoutedEventArgs e)
{
    if (textId.Text == "")
    {
        MessageBox.Show("Empty ID!");
    }
    else if (textName.Text == "")
    {
        MessageBox.Show("Name is required");
    }

    else if (textAddress.Text == "")
    {
        MessageBox.Show("Empty Address!");
    }
    else if (textContact.Text == "")
    {
        MessageBox.Show("Empty Contact Number!");
    }
    else if (textEmail.Text == "")
    {
        MessageBox.Show("Empty email id!");
    }

    else if (textCourse.Text == "")
    {
        MessageBox.Show("Invalid Course!");
    }

    else
    {
        Student dataStudent = new Student();
        dataStudent.ID = textId.Text;
        dataStudent.Name = textName.Text;
        dataStudent.Address = textAddress.Text;
        dataStudent.Contact = textContact.Text;
        dataStudent.Email = textEmail.Text;
        dataStudent.CourseEnroll = textCourse.Text;
        dataStudent.RegDate = textDate.Text;

        DataGridXAML.Items.Add(dataStudent);

        //var dataSet = new DataSet();
        //dataSet.ReadXml(@"D:\student.xml");
        //DataTable dtStdReport = dataSet.Tables[0];
    }
}
```

```

        //int total_Com = 0;
        //int total_Net = 0;
        //int total_Mul = 0;

        //DataTable dt = new DataTable("newTable");
        //dt.Columns.Add("CourseEnroll", typeof(string));
        //dt.Columns.Add("Total Students", typeof(int));

        //for (int i = 0; i < dtStdReport.Rows.Count; i++)
        //{
            string col =
dtStdReport.Rows[i]["CourseEnroll"].ToString();
            if (col == "Computing")
            {
                total_Com++;
            }
            else if (col == "Networking")
            {
                total_Net++;
            }
            else if (col == "Multimedia")
            {
                total_Mul++;
            }

        }

        //dt.Rows.Add("Networking", total_Net);
        //dt.Rows.Add("Computing", total_Com);
        //dt.Rows.Add("Multimedia", total_Mul);
        //grdreport.DataContext = dt.DefaultView;
    }
}

private void btnClear_Click(object sender, RoutedEventArgs e)
{
    textId.Clear();
    textName.Clear();
    textAddress.Clear();
    textContact.Clear();
    textCourse.SelectedIndex = -1;
    textEmail.Clear();
}

private void btnExit_Click(object sender, RoutedEventArgs e)
{
    MessageBox.Show("Window is being exited.");
    this.Close();
}

private void btnReport_Click(object sender, RoutedEventArgs e)
{
    DisplayReport displayReport = new DisplayReport();
    displayReport.Show();
}

}

}

```

**DisplayReport.xaml.cs**

```
using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace ApplicationDevelopmentCW
{
    /// <summary>
    /// Interaction logic for DisplayReport.xaml
    /// </summary>
    public partial class DisplayReport : Window
    {
        DataTable buffer;

        public DisplayReport()
        {
            InitializeComponent();
        }

        private void show_data()
        {
            String dataXML = @"D:\student.xml";
            DataSet dataset = new DataSet();
            dataset.ReadXml(dataXML);

            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("Email", typeof(String));
            buffer.Columns.Add("CourseEnroll", typeof(String));
            buffer.Columns.Add("RegDate", typeof(String));

            for (int i = 0; i < dataset.Tables[0].Rows.Count; i++)
            {
                string s = dataset.Tables[0].Rows[i][6].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(),
                    dataset.Tables[0].Rows[i][5].ToString(),
                    dtime.ToShortDateString());
            }
            DataView dataV = new DataView(buffer);
            DataGridReport.ItemsSource = dataV;
        }
    }
}
```

```

    }

    private void buttonRetrive_Click(object sender, RoutedEventArgs e)
    {
        show_data();
        //var dataSet = new DataSet();
        //dataSet.ReadXml(@"D:\student.xml");
        //DataTable dtStdReport = dataSet.Tables[0];

        //int total_Com = 0;
        //int total_Net = 0;
        //int total_Mul = 0;

        //DataTable dt = new DataTable("newTable");
        //dt.Columns.Add("CourseEnroll", typeof(string));
        //dt.Columns.Add("Total Students", typeof(int));

        //for (int i = 0; i < dtStdReport.Rows.Count; i++)
        //{
            //    string col = dtStdReport.Rows[i]["CourseEnroll"].ToString();
            //    if (col == "Computing")
            //    {
            //        total_Com++;
            //    }
            //    else if (col == "Networking")
            //    {
            //        total_Net++;
            //    }
            //    else if (col == "Multimedia")
            //    {
            //        total_Mul++;
            //    }
            //}
        //dt.Rows.Add("Networking", total_Net);
        //dt.Rows.Add("Computing", total_Com);
        //dt.Rows.Add("Multimedia", total_Mul);
        //grdreport.DataContext = dt.DefaultView;
    }

    private void buttonSName_Click(object sender, RoutedEventArgs e)
    {
        DataView dataV = new DataView(buffer);
        dataV.Sort = "Name ASC";
        DataGridReport.ItemsSource = dataV;
    }

    private void buttonSD_Click(object sender, RoutedEventArgs e)
    {
        DataView dataV = new DataView(buffer);
        dataV.Sort = "RegDate ASC";
        DataGridReport.ItemsSource = dataV;
    }

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

```

```

private void buttonWeekly_Click_1(object sender, RoutedEventArgs e)
{
    var dataSet = new DataSet();
    dataSet.ReadXml(@"D:\student.xml");
    DataTable dtStudentReport = dataSet.Tables[0];

    int total_Computing = 0;
    int total_Networking = 0;
    int total_Multimedia = 0;

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

    for (int i = 0; i < dtStudentReport.Rows.Count; i++)
    {
        String col =
dtStudentReport.Rows[i]["CourseEnroll"].ToString();
        if (col == "Computing")
        {
            total_Computing++;
        }
        else if (col == "Networking")
        {
            total_Networking++;
        }
        else if (col == "Multimedia")
        {
            total_Multimedia++;
        }
    }
    dt.Rows.Add("Computing", total_Computing);
    dt.Rows.Add("Networking", total_Networking);
    dt.Rows.Add("Multimedia", total_Multimedia);

    DataGridReport.DataContext = dt.DefaultView;
}

private void buttonChart_Click_1(object sender, RoutedEventArgs e)
{
    Chart chart = new Chart();
    chart.Show();
}
}
}

```

**Chart.xaml.cs**

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Controls.DataVisualization.Charting;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;

namespace ApplicationDevelopmentCW
{
    /// <summary>
    /// Interaction logic for Chart.xaml
    /// </summary>
    public partial class Chart : Window
    {
        public Chart()
        {
            InitializeComponent();
            var dataSet = new DataSet();
            dataSet.ReadXml(@"D:\student.xml");
            DataTable dtStudentReport = dataSet.Tables[0];

            int total_Computing = 0;
            int total_Networking = 0;
            int total_Multimedia = 0;

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

            for (int i = 0; i < dtStudentReport.Rows.Count; i++)
            {
                String col = dtStudentReport.Rows[i]["CourseEnroll"].ToString();
                if (col == "Computing")
                {
                    total_Computing++;
                }
                else if (col == "Networking")
                {
                    total_Networking++;
                }
                else if (col == "Multimedia")
                {
                    total_Multimedia++;
                }
            }
            dt.Rows.Add("Computing", total_Computing);
            dt.Rows.Add("Networking", total_Networking);
            dt.Rows.Add("Multimedia", total_Multimedia);
        }
    }
}

```



```
((PieSeries)chartGrid).ItemsSource =  
new KeyValuePair<string, int>[]{  
new KeyValuePair<string, int>("Computing", total_Computing),  
new KeyValuePair<string, int>("Networking", total_Networking),  
new KeyValuePair<string, int>("Multimedia", total_Multimedia)};  
    }  
}
```