

# Informatics College Pokhara



## Application Development

### CS6004NP

#### Coursework 1

**Submitted By:**

Student Name: Laxmi Poudel  
London Met ID: 17031920  
Group: L3C2  
Date: 10-Jan-2020

**Submitted To:**

Mr. Ishwor Sapkota

## Table of Contents

<b>Introduction .....</b>	<b>1</b>
<b>User Manual.....</b>	<b>2</b>
<b>Journals Article .....</b>	<b>12</b>
<b>System Architecture .....</b>	<b>15</b>
Architecture Diagram.....	15
Class Diagram .....	15
Flow Chart.....	18
Algorithm .....	19
<b>Bubble Sort.....</b>	<b>20</b>
<b>Conclusion.....</b>	<b>22</b>
<b>References.....</b>	<b>23</b>
<b>Appendix .....</b>	<b>24</b>

## List of Figure

Figure 1: Login Page.....	2
Figure 2: Show error message while user input incorrect user name or password .....	2
Figure 3: Success information after input valid username and password .....	3
Figure 4: Student Information System Layout .....	3
Figure 5: Layout for open csv file .....	4
Figure 6: Message shown while successfully inserted the data .....	4
Figure 7: Added data after press ok button from pop up message box.....	5
Figure 8: File automatically build after save button click .....	5
Figure 9: Individual xml file created of student .....	6
Figure 10: Before sorted by Name .....	6
Figure 11: After Sorted by name .....	7
Figure 12: Before sorted by date.....	7
Figure 13: After Sorted by Date .....	8
Figure 14: After click the Enroll button .....	9
Figure 15: CSV file import.....	9
Figure 16: After clicking the import button.....	10
Figure 17: Shows data in grid after Importing file from the excel file.....	10
Figure 18: Total Number of enroll student.....	10
Figure 19: Pie Chart of total no of people enroll in BIT and BBA .....	11
Figure 20: Architecture Diagram .....	15
Figure 21: Class Diagram .....	16
Figure 22: Individual Class Diagram .....	17
Figure 23: Flow Chart .....	18
Figure 24: Bubble sort.....	21

## Introduction

Windows Presentation Foundation (WPF) is a development framework used to create a desktop application. It is a part of the .NET framework. The WPF has a resolution-independent and vector-based rendering engine which is helpful to deal with modern graphics hardware. The latest version of WPF is 4.6. In this framework, UI of the application is designed in XAML language and Application logic is written in C# programming language (Geeksforgeeks, 2020).

This project is also based on WPF. In this task we need to create Student Information framework. The application enable the client to enter the understudy individual detail including enlistment date with the goal that a framework can create a week after week enrolment report of the understudy. Framework incorporate detail like Name, address, contact no, program select, enrolment date. The application is to monitor the understudy's subtleties, program select and enlistment date.

### Current scenario

Most of the education institution use manual system writing the student information in paper and every few of the education institution use computerized system. By using this manual system many data are lost and input wrong information. Later on there will be difficult in generate report like as weekly or monthly. By using systematic system there is less chance of loss of data so company owner are excited to use computerized system.

## User Manual

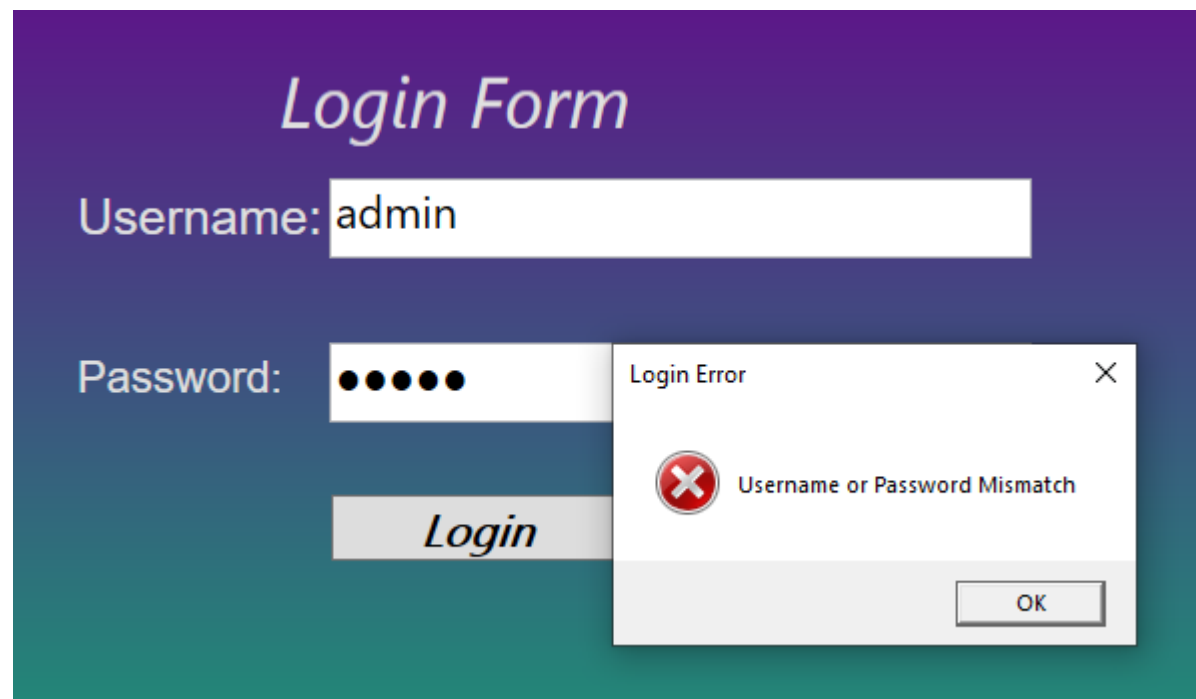
User manual is commonly made to give assistance to people using this system.

### Login Form



The screenshot shows a window titled "Login Page" with a purple-to-green gradient background. The text "Login Form" is displayed in a large, white, italicized font at the top center. Below it, there are two white input fields. The first is labeled "Username:" and the second is labeled "Password:". Below the password field is a grey button with the word "Login" in white, italicized font. The window has standard Windows window controls (minimize, maximize, close) in the top right corner.

Figure 1: Login Page



The screenshot shows the same "Login Page" window as in Figure 1, but with an error message displayed. The "Username:" field now contains the text "admin". The "Password:" field is filled with six black dots. A grey button with the word "Login" in white, italicized font is visible below the password field. An error dialog box is overlaid on the right side of the login form. The dialog box is titled "Login Error" and has a red circle with a white 'X' icon. The text inside the dialog box reads "Username or Password Mismatch". There is an "OK" button at the bottom right of the dialog box.

Figure 2: Show error message while user input incorrect user name or password

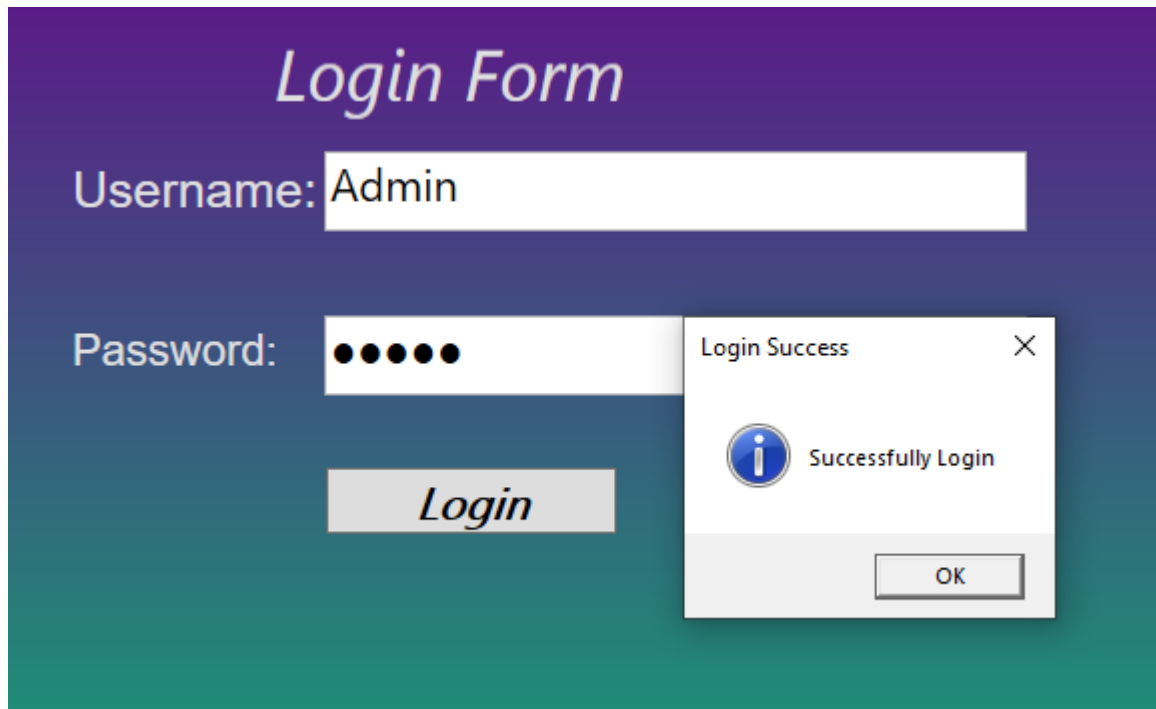


Figure 3: Success information after input valid username and password

## 2. Student Info System Layout

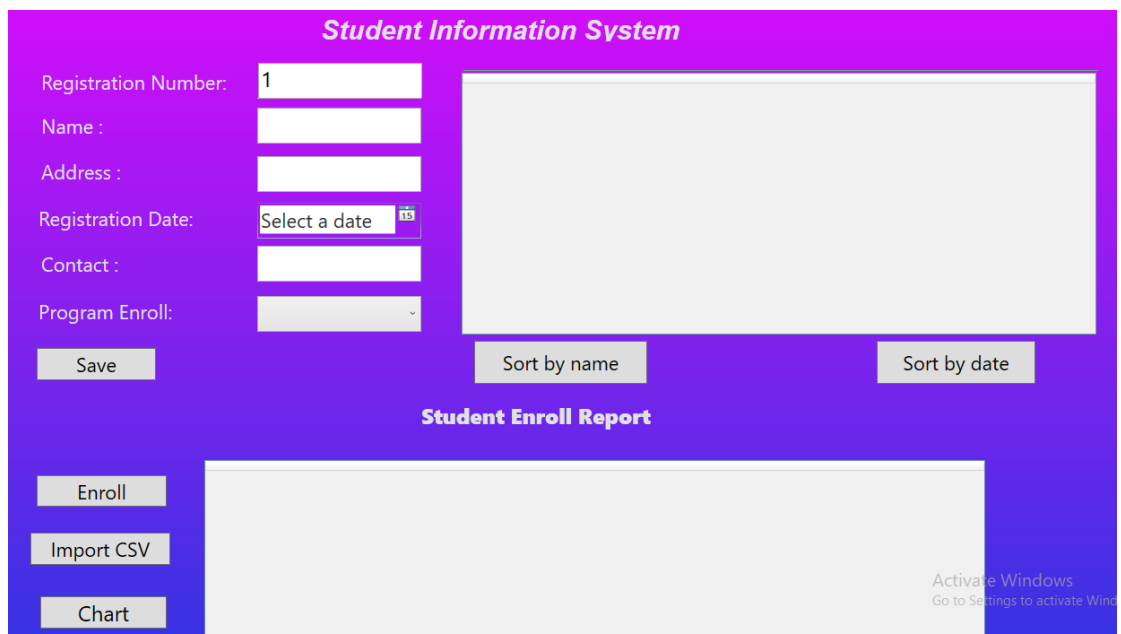


Figure 4: Student Information System Layout

## 3. Layout of open CSV file

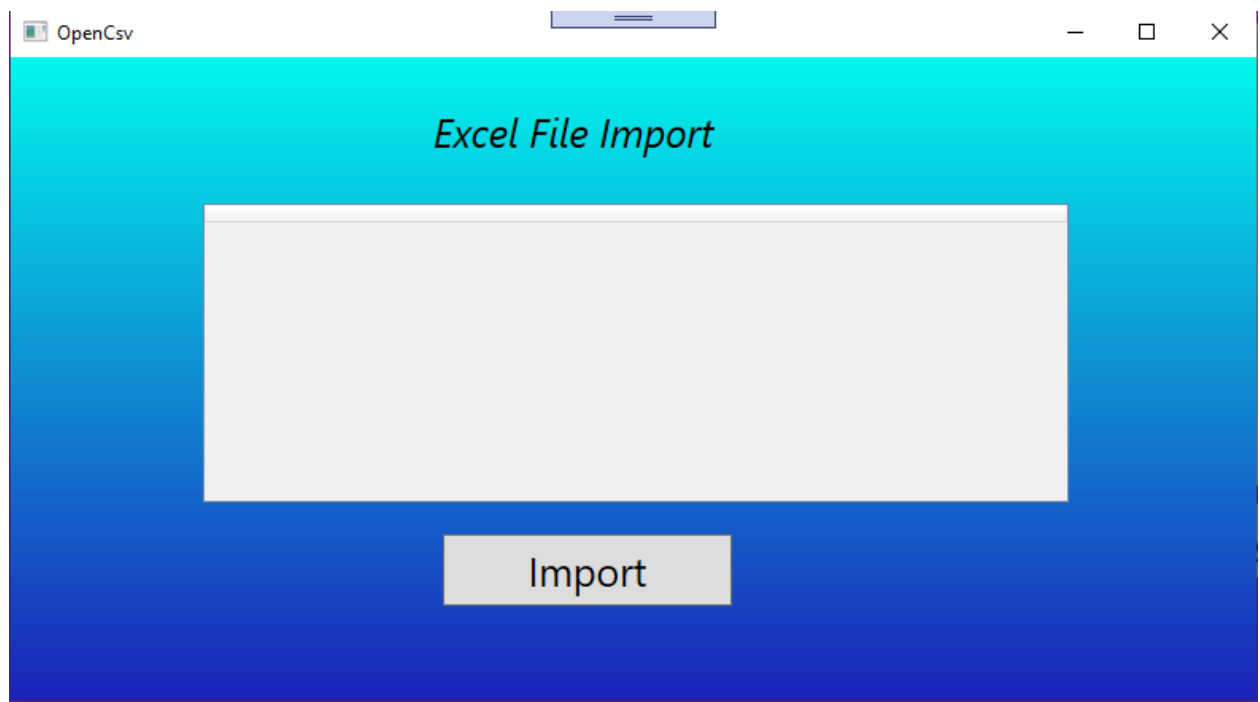


Figure 5: Layout for open csv file


## Inserting Student Info

**Student Information System**

Registration Number:

Name :

Address :

Registration Date:  

Contact :

Program Enroll:

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate	
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020	
2	2	Riza Khatri	Bp Chowk	9832212212	BIT	1/17/2020	

Successful


 Successfully Inserted Data

Figure 6: Message shown while successfully inserted the data

**Student Information System**

Registration Number:

Name :

Address :

Registration Date:

Contact :

Program Enroll:

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020
2	2	Riza Khatri	Bp Chowk	9832212212	BIT	1/17/2020
3	3	Mahima BK	Dulegauda	98032321	BIT	1/24/2020

Figure 7: Added data after press ok button from pop up message box

After saved button clicked following file is create in selected location

This PC > Local Disk (C:) > Appxml

Name	Date modified	Type
count.txt	1/10/2020 2:30 PM	Text Document
StudentCWSchema1.xml	1/10/2020 2:30 PM	XML Document
StudentReport.xml	1/10/2020 3:11 PM	XML Document

Figure 8: File automatically build after save button click

When save button is clicked after input the student information individual xml file also generated.



Local Disk (E:) > Appxml

Search Appxn

Name	Date modified	Type
Ashim GCCWData8.xml	1/9/2020 11:15 PM	XML Document
Asmit TamangCWDData7.xml	1/9/2020 11:15 PM	XML Document
Asmita GcCWDData1.xml	1/10/2020 2:27 PM	XML Document
Asmita GCCWData5.xml	1/9/2020 11:09 PM	XML Document
asmitaCWDData2.xml	1/9/2020 10:15 AM	XML Document
asmittfCWDData4.xml	1/9/2020 1:36 PM	XML Document
CWDData6.xml	1/9/2020 4:15 PM	XML Document
Divya BhattariCWDData4.xml	1/10/2020 2:29 PM	XML Document
Divya BhattraicWDData3.xml	1/9/2020 2:47 PM	XML Document
Divya BhattraicWDData6.xml	1/9/2020 11:09 PM	XML Document
Laxmi PaudelCWDData2.xml	1/9/2020 2:47 PM	XML Document
Laxmi PoudelCWDData1.xml	1/9/2020 9:57 PM	XML Document
Laxmi PoudelCWDData2.xml	1/10/2020 2:28 PM	XML Document
Mahima BKCWDData3.xml	1/9/2020 10:09 PM	XML Document
Mahima BKCWDData5.xml	1/9/2020 2:52 PM	XML Document
Mahima BKCWDData7.xml	1/10/2020 2:30 PM	XML Document
Riza KhaticWDData2.xml	1/9/2020 10:03 PM	XML Document
Riza KhaticWDData3.xml	1/10/2020 2:29 PM	XML Document
Riza KhaticWDData4.xml	1/9/2020 2:48 PM	XML Document
Sneha BKCWDData3.xml	1/9/2020 12:37 PM	XML Document
Sneha GrqCWDData4.xml	1/9/2020 11:08 PM	XML Document

Figure 9: Individual xml file created of student

Sorted by Name

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate	
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020	
2	2	Riza Khati	Bp Chowk	9832212212	BIT	1/17/2020	
3	3	Mahima BK	Dulegauda	98032321	BIT	1/24/2020	
4	4	Sneha Grg	Milan Tole	9837325	BIT	1/21/2020	
5	5	Asmita GC	Parshyang	983727222	BIT	1/31/2020	
6	6	Divya Bhattra	Birauta	98323223	BIT	1/15/2020	
7	7	Asmit Tamang	Arghaun Chowk	98843634	BBA	1/11/2020	
8	8	Ashim GC	Baneshwor	983723	BBA	1/5/2020	

Sort by name

Sort by date

Figure 10: Before sorted by Name

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate	
8	8	Ashim GC	Baneshwor	983723	BBA	1/5/2020	
7	7	Asmit Tamang	Arghaun Chowk	98843634	BBA	1/11/2020	
5	5	Asmita GC	Parshyang	983727222	BIT	1/31/2020	
6	6	Divya Bhattra	Birauta	98323223	BIT	1/15/2020	
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020	
3	3	Mahima BK	Dulegauda	98032321	BIT	1/24/2020	
2	2	Riza Khati	Bp Chowk	9832212212	BIT	1/17/2020	
4	4	Sneha Grg	Milan Tole	9837325	BIT	1/21/2020	

Sort by name

Sort by date

Figure 11: After Sorted by name

Sorted by Date

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate	
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020	
2	2	Riza Khati	Bp Chowk	9832212212	BIT	1/17/2020	
3	3	Mahima BK	Dulegauda	98032321	BIT	1/24/2020	
4	4	Sneha Grg	Milan Tole	9837325	BIT	1/21/2020	
5	5	Asmita GC	Parshyang	983727222	BIT	1/31/2020	
6	6	Divya Bhattra	Birauta	98323223	BIT	1/15/2020	
7	7	Asmit Tamang	Arghaun Chowk	98843634	BBA	1/11/2020	
8	8	Ashim GC	Baneshwor	983723	BBA	1/5/2020	

Sort by name

Sort by date

Figure 12: Before sorted by date

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate	
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020	
7	7	Asmit Tamang	Arghaun Chowk	98843634	BBA	1/11/2020	
6	6	Divya Bhattra	Birauta	98323223	BIT	1/15/2020	
2	2	Riza Khati	Bp Chowk	9832212212	BIT	1/17/2020	
4	4	Sneha Grg	Milan Tole	9837325	BIT	1/21/2020	
3	3	Mahima BK	Dulegauda	98032321	BIT	1/24/2020	
5	5	Asmita GC	Parshyang	983727222	BIT	1/31/2020	
8	8	Ashim GC	Baneshwor	983723	BBA	1/5/2020	

Sort by name

Sort by date

Figure 13: After Sorted by Date

Enrol Student

ID	RegNo	Name	Address	ContactNo	ProgramEnroll	RegistrationDate	
1	1	Laxmi Poudel	Lekhnath	9805801205	BIT	1/1/2020	
2	2	Riza Khati	Bp Chowk	9832212212	BIT	1/17/2020	
3	3	Mahima BK	Dulegauda	98032321	BIT	1/24/2020	
4	4	Sneha Grg	Milan Tole	9837325	BIT	1/21/2020	
5	5	Asmita GC	Parshyang	983727222	BIT	1/31/2020	
6	6	Divya Bhattra	Birauta	98323223	BIT	1/15/2020	
7	7	Asmit Tamang	Arghaun Chowk	98843634	BBA	1/11/2020	
8	8	Ashim GC	Baneshwor	983723	BBA	1/5/2020	

Sort by name

Sort by date

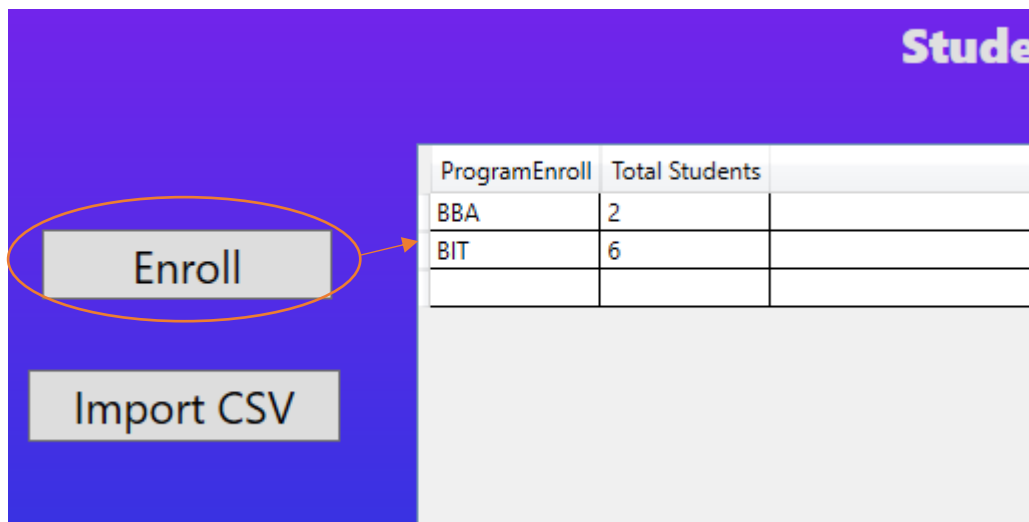


Figure 14: After click the Enroll button

#### CSV File Import Layout

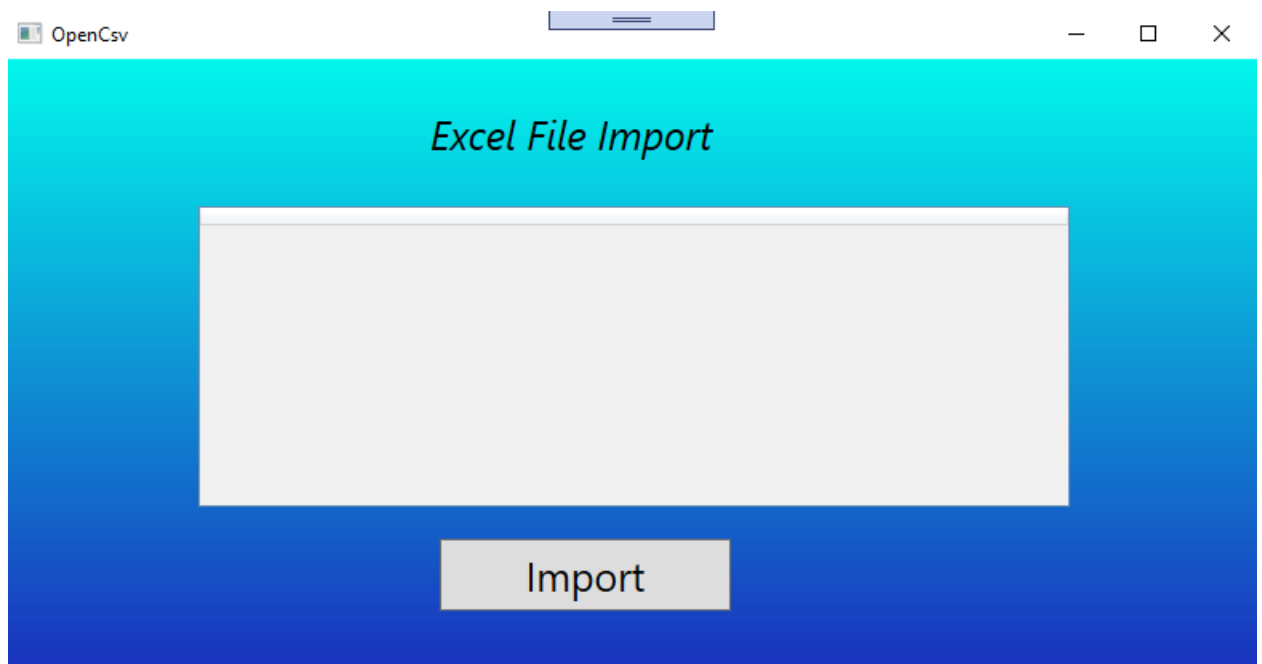


Figure 15: CSV file import

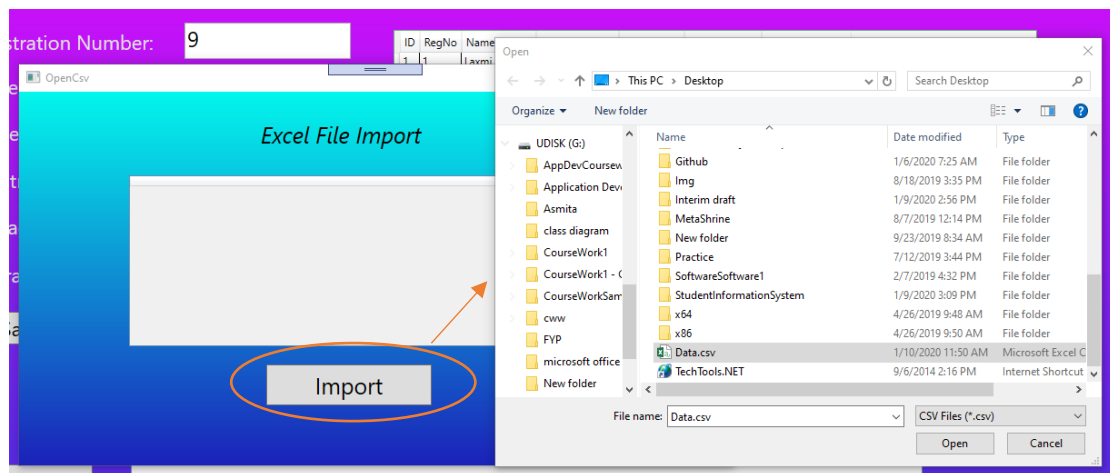


Figure 16: After clicking the import button



Figure 17: Shows data in grid after Importing file from the excel file

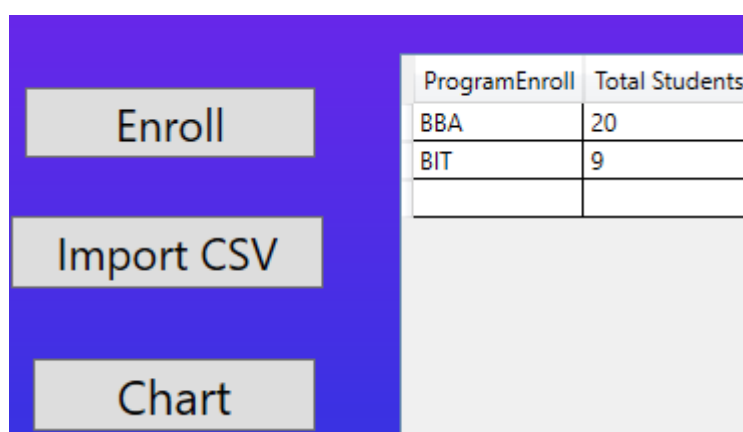


Figure 18: Total Number of enroll student

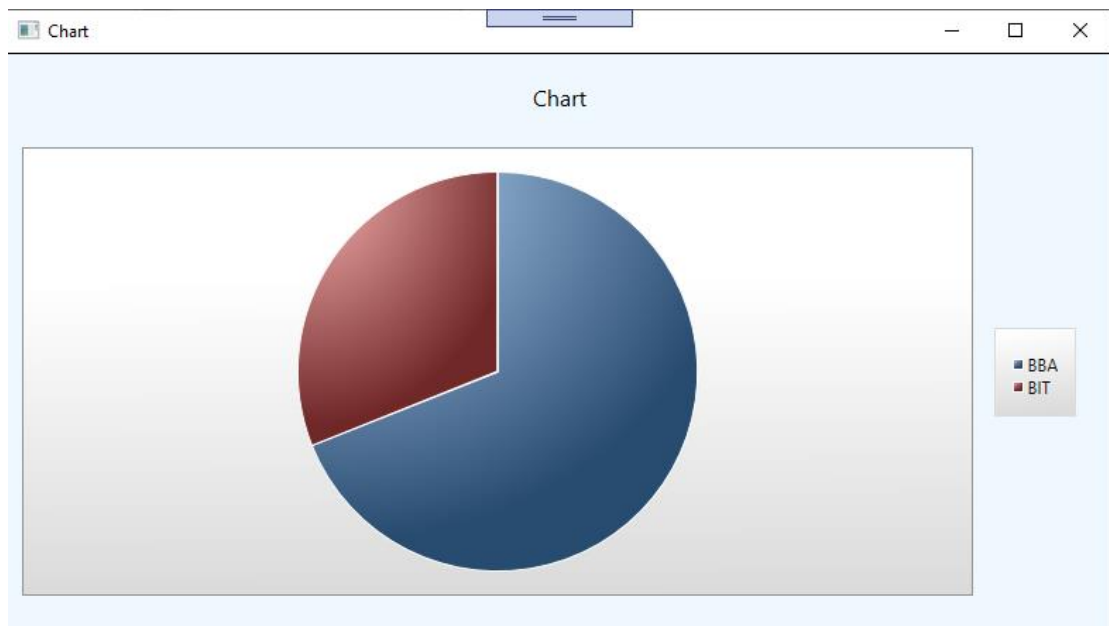


Figure 19: Pie Chart of total no of people enroll in BIT and BBA

## Journals Article

- Our Online understudy enlistment framework empowers understudies to select into their subjects preceding the initiation of their semesters. This enlistment framework not just enables worldwide understudies to enlist through web without going to the grounds yet in addition fuses the business rules. These business rules spread a wide scope of guidelines and approach, for example, subject pre-imperative, understudy's instalment status, course facilitator's choice and the correspondence of understudies' position to the proposed selecting subjects. Other than business controls, the framework additionally consolidates different notice components like Short Messaging Service and Email. XML is utilized to store the business rules and subsequently permit the trans-portability of the framework interface to more extensive scope of gadgets, for example, Personal Device Assistant. The interface auto-distinguishes the client's gadget either PC/PC or a lot littler screen gadget, for example, PDA. To put it plainly, the enlistment framework upheld motor runs dependent on the business rules and front-end motor rushes to give high fulfillment client experience. With the business and UI, the framework can run the work process of understudy enlistment from the online enlistment structure to endorsement work process cycle running parallel with the notice capacity. (H.H., 2006)
- This investigation intended to improve the effectiveness of the current Student Information System of Kalinga State University Rizal grounds. To achieve this target, an appraisal of the current framework was done through observation and meeting techniques from the Acting Registrar, Campus Secretary, Faculty Members and students. Results uncover that the current understudy data framework met the five necessities: re-usability, maintainability, security, helpfulness and functionality and assessment on the framework intrigue of a quality programming just to a "moderate degree". Also, unwavering quality of the current framework was given a "low degree" rating. In view of the evaluation results, the current understudy data framework was refined to incorporate proposals, for example, the inclusion of online inquiry get to, online accessibility of understudy data, and a job based security in the system. During the attempt out of the created understudy data framework, it was surveyed with "high

degree" of viability as far as all framework segments with the exception of its ease of use and productivity which is evaluated as "moderate degree". The created Student Information System gave more noteworthy fulfillment to the clients contrasted and the existing system for an effective questioning of understudy data records, keeping the understudy records in a more secured manner, and it gives progressively solid data records of understudies in Kalinga State University Rizal grounds (Chim, 2016).

- Another kind of understudy data the executive's framework is intended to actualize understudy data distinguishing proof and the executives dependent on unique mark recognizable proof. So as to guarantee the security of information transmission, this paper proposes an information encryption technique dependent on an improved AES calculation. Another - box is cunningly planned, which can altogether diminish the encryption time by improving Byte Sub, Shift Row, and Mix Column in the round change of the conventional AES calculation with the procedure of look-into table. Exploratory outcomes show that the proposed calculation can altogether improve the encryption time contrasted and the conventional AES calculation (Anon., 2017).
- Student Information Management System (SIMS) provides a simple interface for maintenance of student information. It can be used by educational institutes or colleges to maintain the records of students easily. The creation and management of accurate, up-to-date information regarding a students' academic career is critically important in the university as well as colleges. Student information system deals with all kind of student details, academic related reports, college details, course details, curriculum, batch details, placement details and other resource related details too. It tracks all the details of a student from the day one to the end of the course which can be used for all reporting purpose, tracking of attendance, progress in the course, completed semesters, years, coming semester year curriculum details, exam details, project or any other assignment details, final exam result and all these will be available through a secure, online interface embedded in the college's website. It will also have faculty details, batch execution details, students' details in all aspects, the various academic notifications to the staff and students updated by the college administration. It also facilitate us explore all the activities happening in the college, Different reports and Queries can be generated based



on vast options related to students, batch, course, faculty, exams, semesters, certification and even for the entire college (S.R.Bharamagoudar, 2013).

## System Architecture

Set of shows, rules, and measures utilized in a PC framework's specialized structure, in addition to client prerequisites and determinations, that the framework's producer (or a framework integrator) follows in planning (or incorporating) the framework's different parts, (for example, hardware, programming and systems) (Businessdictionary, 2020).

### Architecture Diagram

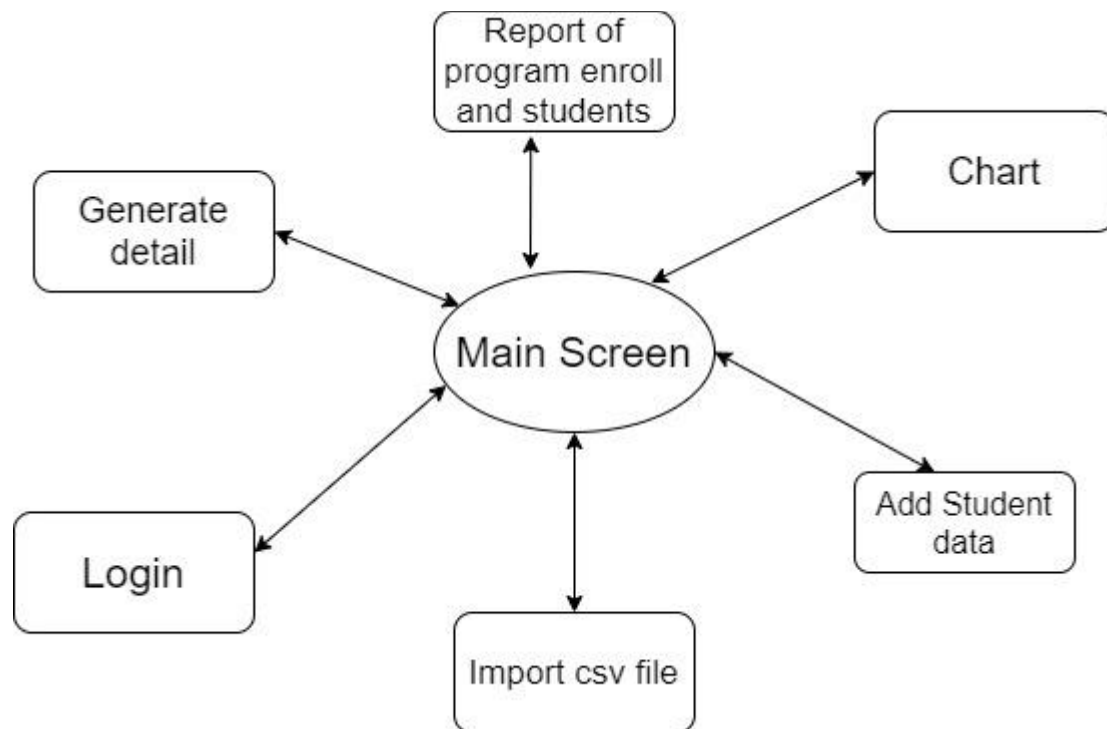


Figure 20: Architecture Diagram

The above diagram shows the architecture diagram of the project developed by me. Here in this project functionality like report of program enrollment by faculty, chart, and add student information to the system, login page for the admin who can only access the information of the student. Import csv file to the system.

### Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application.

Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modeling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages.

Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram (Tutorials point, 2020).

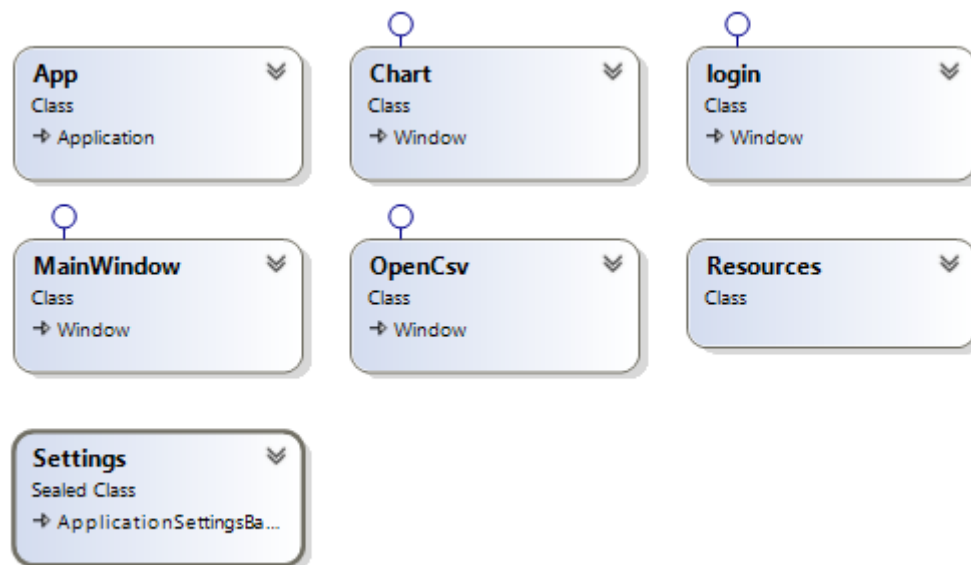


Figure 21: Class Diagram

Individual Diagram

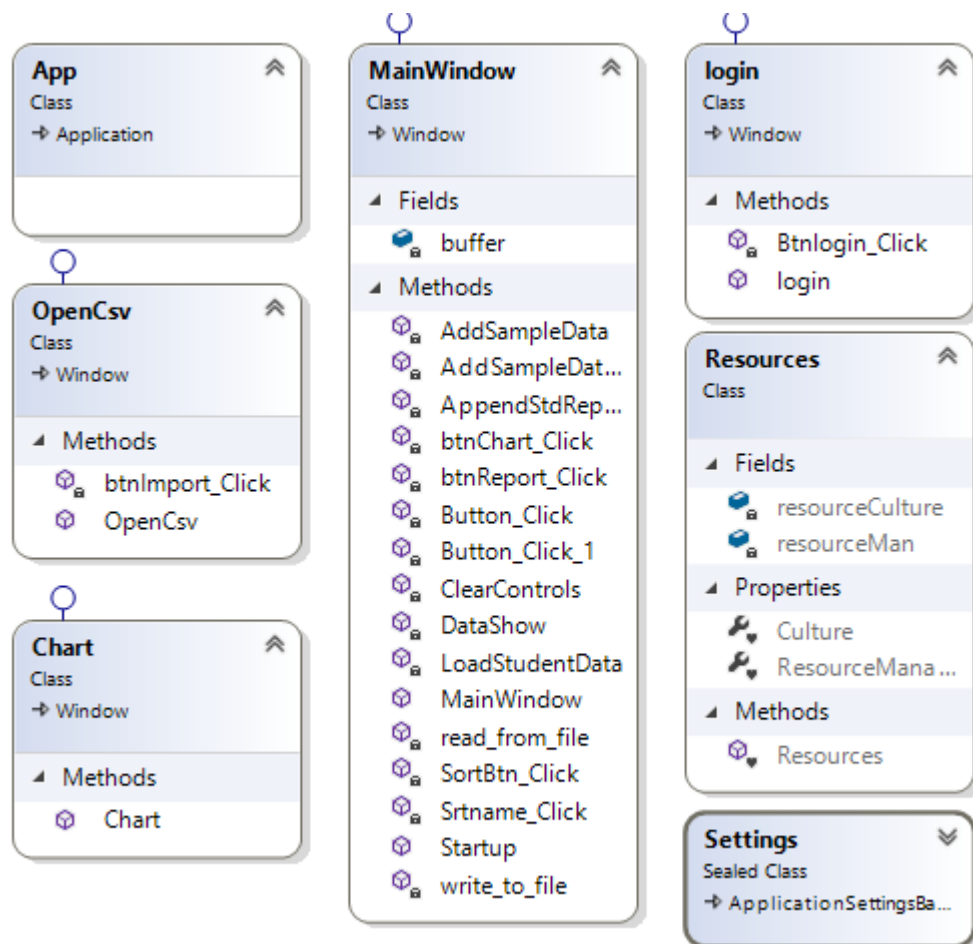


Figure 22: Individual Class Diagram

## Flow Chart

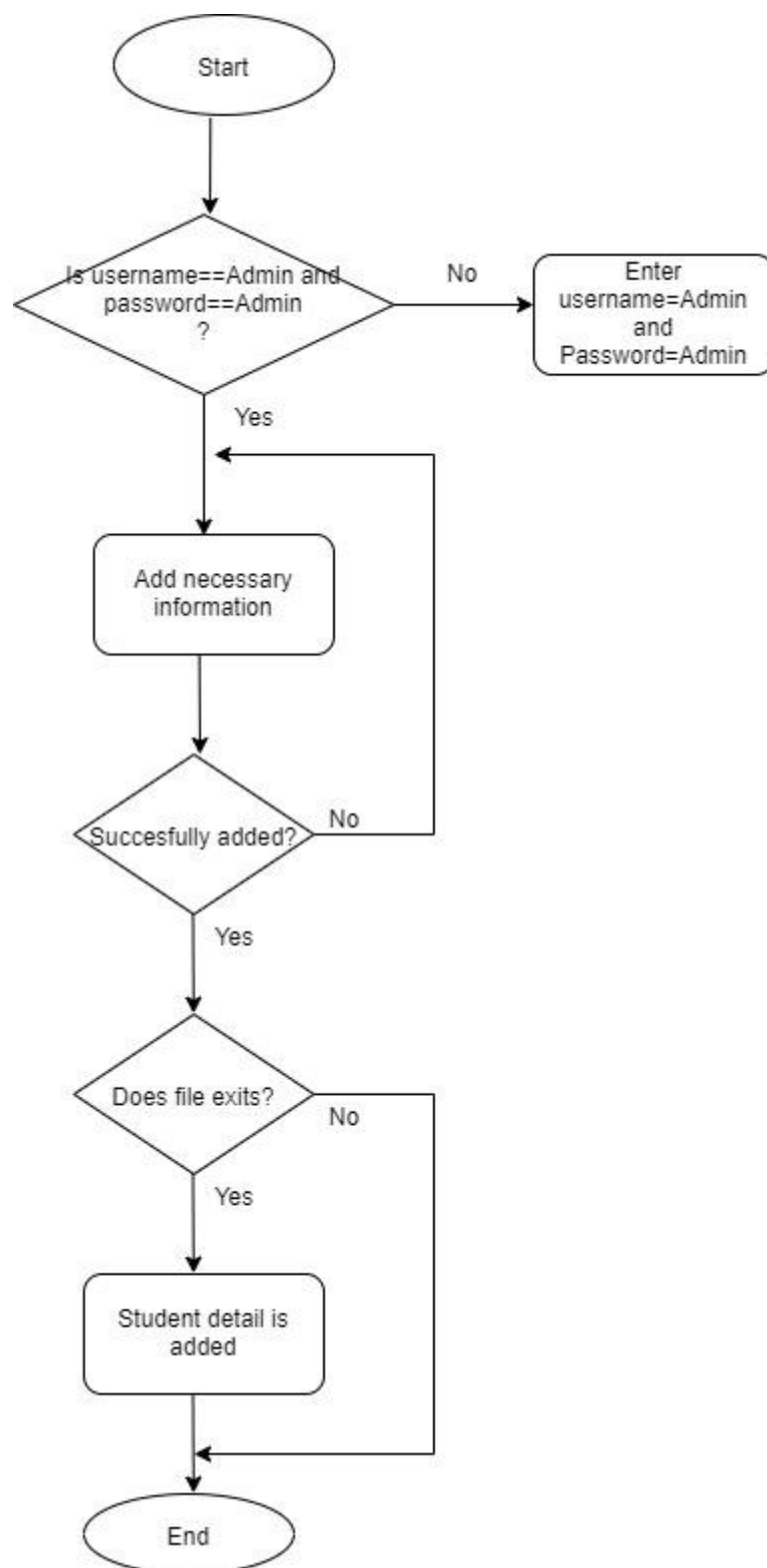


Figure 23: Flow Chart

### Algorithm

Step 1: Start

Step 2: check whether the user input valid username or password.

Step 3: If user does not input valid username or password then show error message.

Step 4: If user input valid username and password the go to step 5.

Step 5: Show Student Information add layout and add the necessary information.

Step 6: check whether information is successfully added or not.

Step 7: If not go to Step 5.

Step 8: If data added successfully check file exists or not.

Step 9: if not go to step 11.

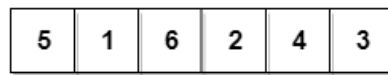
Step 10: File exists then add student detail.

Step 11: End

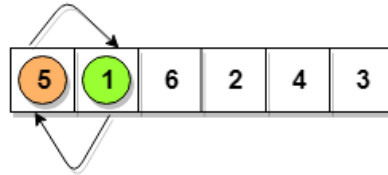
## Bubble Sort

Bubble Sort is a simple algorithm which is used to sort a given set of  $n$  elements provided in form of an array with  $n$  number of elements. Bubble Sort compares the entire element one by one and sort them based on their values. If the given array has to be sorted in ascending order, then bubble sort will start by comparing the first element of the array with the second element, if the first element is greater than the second element, it will swap both the elements, and then move on to compare the second and the third element, and so on. If we have total  $n$  elements, then we need to repeat this process for  $n-1$  times. It is known as bubble sort, because with every complete iteration the largest element in the given array, bubbles up towards the last place or the highest index, just like a water bubble rises up to the water surface. Sorting takes place by stepping through all the elements one-by-one and comparing it with the adjacent element and swapping them if required (studynight, 2020).

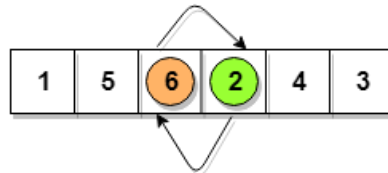
$5 > 1$   
so interchange



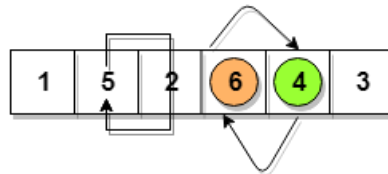
$5 < 6$   
No swapping



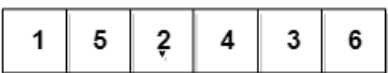
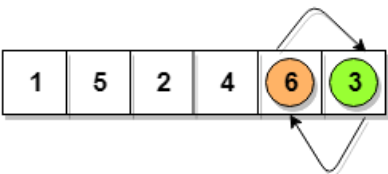
$6 > 2$   
so interchange



$6 > 4$   
so interchange



$6 > 3$   
so interchange



This is first insertion

similarly, after all the  
iterations, the array  
gets sorted

Figure 24: Bubble sort

Source: <https://www.studytonight.com/data-structures/images/basic-bubble-sort.png>



## **Conclusion**

This is the report of the application development (CS6004NA). To complete this coursework lot of researches was done. I have visited many website and journals. While making this project many errors bugs are seen and at the same time that problem was solved by the help of friends, module leader.

This project consists of login page main window where admin put all the information of the student and save. The saved files are saving to the xml file and schemas also develop and CSV file also import.

## References

A Student Information Management System Based on Fingerprint Identification and Data Security Transmission. (2017). *Journal of Electrical and Computer Engineering*.

*Businessdictionary*. (2020, jan 03). Retrieved from <http://www.businessdictionary.com/definition/system-architecture.html>

Chim, P. C. (2016). Student Information System for Kalinga State University. *International Journal of Management and Commerce Innovations*, Vol. 4(1), 330-335.

*Geeksforgeeks*. (2020, jan 2). Retrieved from <https://www.geeksforgeeks.org/what-is-wpf/>

H.H., P. (2006). Online Student Enrollment System. *researchgate*, 393-396.

S.R.Bharamagoudar, G. R. (2013). Web Based Student Information Management. *International Journal of Advanced Research in Computer and Communication Engineering*, Vol. 2(Issue 6), 2342-2348.

*studynight*. (2020, jan 05). Retrieved from <https://www.studytonight.com/data-structures/bubble-sort>

*Tutorials point*. (2020, Jan 05). Retrieved from [https://www.tutorialspoint.com/uml/uml\\_class\\_diagram.htm](https://www.tutorialspoint.com/uml/uml_class_diagram.htm)

## Appendix

### Login Page

```
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;

using AppDevCoursewrk;

namespace AppDevCoursewrk
{
    /// <summary>
    /// Interaction logic for login.xaml
    /// </summary>

    public partial class login : Window
    {
```

```
public login()
{
    InitializeComponent();
}

private void Btnlogin_Click(object sender, RoutedEventArgs e)
{
    var user = txtuser.Text;
    var pass = txtpass.Password;
    if (user.Equals("") || pass.Equals(""))
    {
        MessageBox.Show("Empty value", "Login Error",
        MessageBoxButton.OK, MessageBoxImage.Error);
    }
    else if (user != "Admin" || pass != "Admin")
    {
        MessageBox.Show("Username or Password Mismatch", "Login
        Error", MessageBoxButton.OK, MessageBoxImage.Error);
        txtuser.Clear();
        txtpass.Clear();
    }
    else
```

```
{  
    MessageBox.Show("Successfully Login", "Login Success",  
        MessageBoxButton.OK, MessageBoxImage.Information);  
    this.Hide();  
    MainWindow mainWindow = new MainWindow();  
    mainWindow.Show();  
}  
}  
}  
}
```

Main Window

```
using System;  
using System.Collections.Generic;  
using System.Data;  
using System.IO;  
using System.Windows;  
using CourseWorkOne;  
using DataHandler;  
using Microsoft.Win32;  
  
namespace AppDevCoursewrk  
{  
    /// <summary>  
    /// Interaction logic for MainWindow.xaml
```

```
/// </summary>

public partial class MainWindow : Window
{
    private DataTable buffer;

    public MainWindow()
    {
        InitializeComponent();

        Startup();

        txtRegNo.Text = read_from_file();

        LoadStudentData();

    }

    public void Startup()
    {
        //var handler = new Handler();

        //var dataSet = handler.CreateDataSet();

        //AddSampleData(dataSet);

        //dataSet.WriteXmlSchema(@"D:\StudentCWSchema.xml");
    }
}
```

```
//dataSet.WriteXml(@"D:\StudentCWData.xml");

//var dataSet = new DataSet();

//dataSet.ReadXmlSchema(@"D:\StudentCWSchema.xml");

//dataSet.ReadXml(@"D:\StudentCWData.xml");

//var i = 0;
}

private void AddSampleData(DataSet dataSet)
{

    var dr = dataSet.Tables["Course"].NewRow();
    dr["Name"] = "BBA";
    dr["DisplayText"] = "BBA Hons";
    dataSet.Tables["Course"].Rows.Add(dr);

    dr = dataSet.Tables["Course"].NewRow();
    dr["Name"] = "Network & Communication";
    dr["DisplayText"] = "BCA Network";
    dataSet.Tables["Course"].Rows.Add(dr);

    dr = dataSet.Tables["Course"].NewRow();
    dr["Name"] = "Programming & Application Development";
```

```
dr["DisplayText"] = "BSc CSIT Application Development";  
dataSet.Tables["Course"].Rows.Add(dr);  
  
dr = dataSet.Tables["Student"].NewRow();  
dr["Name"] = "Laxmi Poudel";  
dr["Address"] = "Budi Bazar";  
dr["ContactNo"] = "98938328";  
dr["CourseEnroll"] = 1;  
dr["RegistrationDate"] = DateTime.Today.AddDays(-2);  
dataSet.Tables["Student"].Rows.Add(dr);  
  
dr = dataSet.Tables["Student"].NewRow();  
dr["Name"] = "Riza Khati";  
dr["Address"] = "BP chowk";  
dr["ContactNo"] = "838732873";  
dr["CourseEnroll"] = 2;  
dr["RegistrationDate"] = DateTime.Today.AddDays(-1);  
dataSet.Tables["Student"].Rows.Add(dr);  
  
}  
  
private void AddSampleDataforStd(DataSet dataSet)  
{
```



```
var dr = dataSet.Tables["Course"].NewRow();

dr["Name"] = "BBA";

dr["DisplayText"] = "BBA Hons";

dataSet.Tables["Course"].Rows.Add(dr);


var dr1 = dataSet.Tables["Student"].NewRow();

dr1["Name"] = txtName.Text;

dr1["Address"] = txtAddress.Text;

dr1["ContactNo"] = txtContact.Text;

dr1["ProgramEnroll"] = combo.Text;

dr1["RegistrationDate"] = DateTime.Today.AddDays(-2);

dataSet.Tables["Student"].Rows.Add(dr1);


}


private void AppendStdReport(DataSet dataSet)
{
    if (File.Exists(@"C:\Appxml\StudentReport.xml"))
    {
        var handler = new Handler();

        dataSet.Tables["StudentReport"].ReadXml(@"C:\Appxml\StudentReport.xml")
;
    }
}
```

```
var dr2 = dataSet.Tables["StudentReport"].NewRow();

dr2["RegNo"] = txtRegNo.Text;

dr2["Name"] = txtName.Text;

dr2["Address"] = txtAddress.Text;

dr2["ContactNo"] = txtContact.Text;

dr2["ProgramEnroll"] = combo.Text;

dr2["RegistrationDate"] = txtdate.Text;

dataSet.Tables["StudentReport"].Rows.Add(dr2);


dataSet.Tables["StudentReport"].WriteXml(@"C:\Appxml\StudentReport.xml")
;


    }

    else

    {


dataSet.Tables["StudentReport"].WriteXml(@"C:\Appxml\StudentReport.xml")
;


        AppendStdReport(dataSet);

    }

}

private void Button_Click_1(object sender, RoutedEventArgs e)
```

```
{  
    var handler = new Handler();  
    var dataSet = handler.CreateDataSet();  
    AddSampleDataforStd(dataSet);  
    MessageBox.Show("Successfully Inserted Data", "Successful",  
        MessageBoxButtons.OK, MessageBoxIcon.Information);  
    AppendStdReport(dataSet);  
  
    var regno = txtRegNo.Text;  
    var name = txtName.Text;  
    dataSet.WriteXmlSchema(@"C:\Appxml\StudentCWSchema1.xml");  
    dataSet.Tables["Student"].WriteXml(@"E:\APPXML\" + name +  
        "CWData" + regno + ".xml");  
  
    dataSet.Tables[2].WriteXml(@"C:\Appxml\StudentReport.xml");  
  
    write_to_file(txtRegNo.Text);  
  
    txtRegNo.Text = read_from_file();  
  
    ClearControls();  
    LoadStudentData();  
}  
  
private void write_to_file(string text)
```

```
{

    File.WriteAllText(@"C:\Appxml\count.txt", text);

}

private string read_from_file()
{
    /*
        string      text      =      System.IO.File.ReadAllText(@"C:\Appxml
storage\count.txt");

        int i;

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

        i = i + 1;

        return i.ToString();*/

    int i = 1;

    if (File.Exists(@"C:\Appxml\count.txt"))
    {
        string text = File.ReadAllText(@"C:\Appxml\count.txt");

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

        i = i + 1;
    }
    else
    {
```

```
        //File.WriteAllText(@"E:\APPXML\count.txt", "text");

    }

    return i.ToString();

}

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

private void LoadStudentData()
{
    if (System.IO.File.Exists(@"C:\Appxml\StudentReport.xml"))
    {
        var handler = new Handler();

        var dataSet = new DataSet();

        dataSet.ReadXml(@"C:\Appxml\StudentReport.xml");
    }
}
```

```
        DataTable dtStdReport = new DataTable();

        dtStdReport = dataSet.Tables[0];

        grdStd.DataContext = dtStdReport.DefaultView;
    }

}

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

    dataSet.ReadXml(@"C:\Appxml\StudentReport.xml");

    DataTable dtStdReport = dataSet.Tables[0];

    int total_BIT = 0;

    int total_BBA = 0;

    DataTable dt = new DataTable("newTable");

    dt.Columns.Add("ProgramEnroll", typeof(string));

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

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

dt.Rows.Add("BBA", total_BBA);
dt.Rows.Add("BIT", total_BIT);
grdreport.DataContext = dt.DefaultView;

}

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

```
dataSet.ReadXml(@"C:\Appxml\StudentReport.xml");

DataTable DataTable = dataSet.Tables["StudentReport"];

DataTable.DefaultView.Sort = "Name Asc";

grdStd.DataContext = DataTable.DefaultView;

}

private void SortBtn_Click(object sender, RoutedEventArgs e)

{

    var dataSet = new DataSet();

    dataSet.ReadXml(@"C:\Appxml\StudentReport.xml");

    DataTable DataTable = dataSet.Tables["StudentReport"];

    DataTable.DefaultView.Sort = "RegistrationDate Asc";

    grdStd.DataContext = DataTable.DefaultView;

}

private void DataShow()

{

    string dataXMLFile = @"C:\Appxml\StudentReport.xml";

    System.Data.DataSet dataset = new DataSet();

    dataset.ReadXml(dataXMLFile);

    buffer = new DataTable("dt");

    buffer.Columns.Add("RegNo", typeof(string));

    buffer.Columns.Add("Name", typeof(string));

    buffer.Columns.Add("Address", typeof(string));
```



```
buffer.Columns.Add("ContactNo", typeof(string));

buffer.Columns.Add("ProgramEnroll", typeof(string));

buffer.Columns.Add("RegistrationDate", typeof(string));


for (int i = 0; i < dataset.Tables[0].Rows.Count; i++)
{
    string s = dataset.Tables[0].Rows[i][5].ToString();

    DateTime dtime = DateTime.Parse(s);

    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);

grdreport.ItemsSource = dataView;
}


private void btnReport_Click(object sender, RoutedEventArgs e)
{
    var import = new OpenCsv();
```

```
import.Show();

}

private void btnChart_Click(object sender, RoutedEventArgs e)
{
    var chart = new Chart();

    chart.Show();
}
}

}
```

### Import CSV file page

```
using Microsoft.Win32;
using System;
using System.Collections.Generic;
using System.Data;
using System.Data.OleDb;
using System.Globalization;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows;
using System.Windows.Controls;
using System.Windows.Data;
using System.Windows.Documents;
using System.Windows.Input;
using System.Windows.Media;
using System.Windows.Media.Imaging;
using System.Windows.Shapes;
```

```

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

        private void btnImport_Click(object sender, RoutedEventArgs e)
        {
            var dataSet = new DataSet();
            dataSet.ReadXml(@"C:\Appxml\StudentReport.xml");
            OpenFileDialog openFileDialog = new OpenFileDialog();
            openFileDialog.Filter = "CSV Files|*.csv";
            openFileDialog.DefaultExt = ".csv";

            bool? fileselect = openFileDialog.ShowDialog();
            if (fileselect == true)
            {
                string filePath = openFileDialog.FileName;
                //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["StudentReport"].NewRow();
                        newRow["ID"] = values[0];
                        newRow["RegNo"] = values[1];
                        newRow["Name"] = values[2];
                        newRow["Address"] = values[3];
                        newRow["ContactNo"] = values[4];
                        newRow["ProgramEnroll"] = values[5];
                        newRow["RegistrationDate"] = values[6];
                        dataSet.Tables["StudentReport"].Rows.Add(newRow);
                    }

                    dataSet.WriteXml(@"C:\Appxml\StudentReport.xml");
                    dataGrid1.ItemsSource =
dataSet.Tables["StudentReport"].DefaultView;

                }

            }
        }
    }
}

```

## Chart

```

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

            var dataSet = new DataSet();
            dataSet.ReadXml(@"C:\Appxml\StudentReport.xml");
            DataTable dtStdReport = dataSet.Tables[0];

            int total_BIT = 0;
            int total_BBA = 0;

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

            for (int i = 0; i < dtStdReport.Rows.Count; i++)
            {
                string col = dtStdReport.Rows[i]["ProgramEnroll"].ToString();
                if (col == "BIT")
                {
                    total_BIT++;
                }
                else if (col == "BBA")
                {
                    total_BBA++;
                }
            }
            dt.Rows.Add("BBA", total_BBA);
            dt.Rows.Add("BIT", total_BIT);

            ((PieSeries)PieChart).ItemsSource = new KeyValuePair<string, int>[]
            {
                new KeyValuePair<string, int>("BBA", total_BBA),
            }
        }
    }
}

```

```
        new KeyValuePair<string, int>("BIT", total_BIT++),  
    };  
}  
}
```