

Informatics College Pokhara



informatics
college pokhara

Application Development

CS6004NI

Course Work 1

Submitted By: Ashish Bhandari
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 some validation
Enrollment Report & weekly report in tabular format	Any one of the report is missing or not complete
Course wise enrollment report & Chart display	any one component is missing or inappropriate data is shown
Algorithm used for sorting & proper sorting of data	Data Sorted using name or registration date only.
B. Documentation	
User Manual for running the application	User Manual is average. Includes description for all interfaces

Marking Scheme

Application architecture & description of the classes and methods used	architecture is included and satisfactory description of class and methods used.
Flow chart, algorithms and data structures used	average work with very limited explanation and missing diagrammatic representation.
Reflective essay	Average work with unclear learnings, experience or findings.

C. Programming Style

Clarity of code, Proper Naming convention & comments	very poorly written code and no comments at all
System Usability	System can't be used and have issues

Overall Grade:	B+	B+
-----------------------	-----------	-----------

Overall Comment:

Code should be self-explainable with less comments. Need some proper naming of the components 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 .
Good Job

Informatics College Pokhara



Application Development

Coursework 1

Submitted By:

Student Name: Ashish Bhandari

Student's Id: 17031918

Submission date: 10th Jan, 2020

Submitted To:

Mr. Ishwor Sapkota

Group: L3C1

Table of Index

ABSTRACT	1
KEYWORD LIST	1
ACKNOWLEDGEMENTS	1
1. INTRODUCTION	2
2. Detail Instruction to run program.....	3
3. Detailed description of the classes, properties and methods.....	15
4. Class Diagram	19
5. System Architecture.....	20
6. Flow Chart	21
7. Algorithms of Reports	25
8. Data structure used in program	28
9. Reflection of own experience of using C# and Visual Studio.....	29
10. Test Case	30
11. CONCLUSION	36
12. References	37
13. Appendix	37

Table of Figure

Figure 1: Start up window	3
Figure 2: Login window.....	4
Figure 3: Forget password window	4
Figure 4: Home window	5
Figure 5: Password change window	6
Figure 6: Register student Window.....	7
Figure 7: Student register window	8
Figure 8:Student Data Confirmation Window.....	9
Figure 9: Register From Existing File Window	10
Figure 10: View Student Data Window	11
Figure 11: Weekly Student Registration Report.....	12
Figure 12: Graphical Chart	13
Figure 13: View All Student Data	14
Figure 14:Class Diagram	19
Figure 15: System Architecture	20
Figure 16: Flow chart of program startup.....	21
Figure 17: Home window Flowchart	22
Figure 18:Flow chart of Bulk Register From file	23
Figure 19: Flow chart of individual student register	24
Figure 20:Flow chart of Bulk Register From file	24
Figure 21: Empty password test	30
Figure 22:Wrong Username and Password test	31
Figure 23: Empty username and password test	31
Figure 24: Entering number as name test.....	32
Figure 25: ID as string test.....	33
Figure 26: Contact number as string test.....	33
Figure 27: Email validation test.....	34
Figure 28: Empty input test.....	35
Figure 29: Short Contact number test.....	35

ABSTRACT

This project addresses the study and development of an WPF student registration management system using c# language. Windows Presentation Foundation(WPF) is a development framework used to create a desktop application. It is a part of the .NET framework. This project enable users, make an computerized registration system to manage student data.

Thus, this project studies some issues on how WPF application is build. This project will provide Decision Support System to manage student registration data. This app will assist in future development that would support a fully integrated system.

KEYWORD LIST

- C#
- Student management
- Student registration management
- WPF application
- XML

ACKNOWLEDGEMENTS

I am grateful to Informatics college Pokhara for providing me this opportunity to prove my academic capabilities in something more professional and practical. This honors project carried out in crunch time has taught me utterly necessary lessons as to how a WPF application is developed and how student management system project should be managed and handled. I am very thank full to my module leader Mr. Ishwor Sapkota for enduring my steep mistakes and shortcomings.

1. INTRODUCTION

As per the requirement of module's coursework, we have developed a student management system for an educational institution. The system is used to keep records of students. Student Management System is a software that is helpful for students as well as the school authorities.

In the current system all the activities are done manually. It is very time consuming and costly. Currently most of school, colleges, etc. uses manual system which is old fashioned. For changes, a digitalized system is required. After the success of system people are looking for it because of its easy record keeping and generating student report and also it help in analyze their business. Our Student Management System deals with the various activities related to the students.

1.1 Current Scenario

There are numerous Museum who keep record of their data in old traditional system which is Paper-Based System. In addition to that, there are some school and colleges with digital system but are well lacking the features which are needed for a school/college.

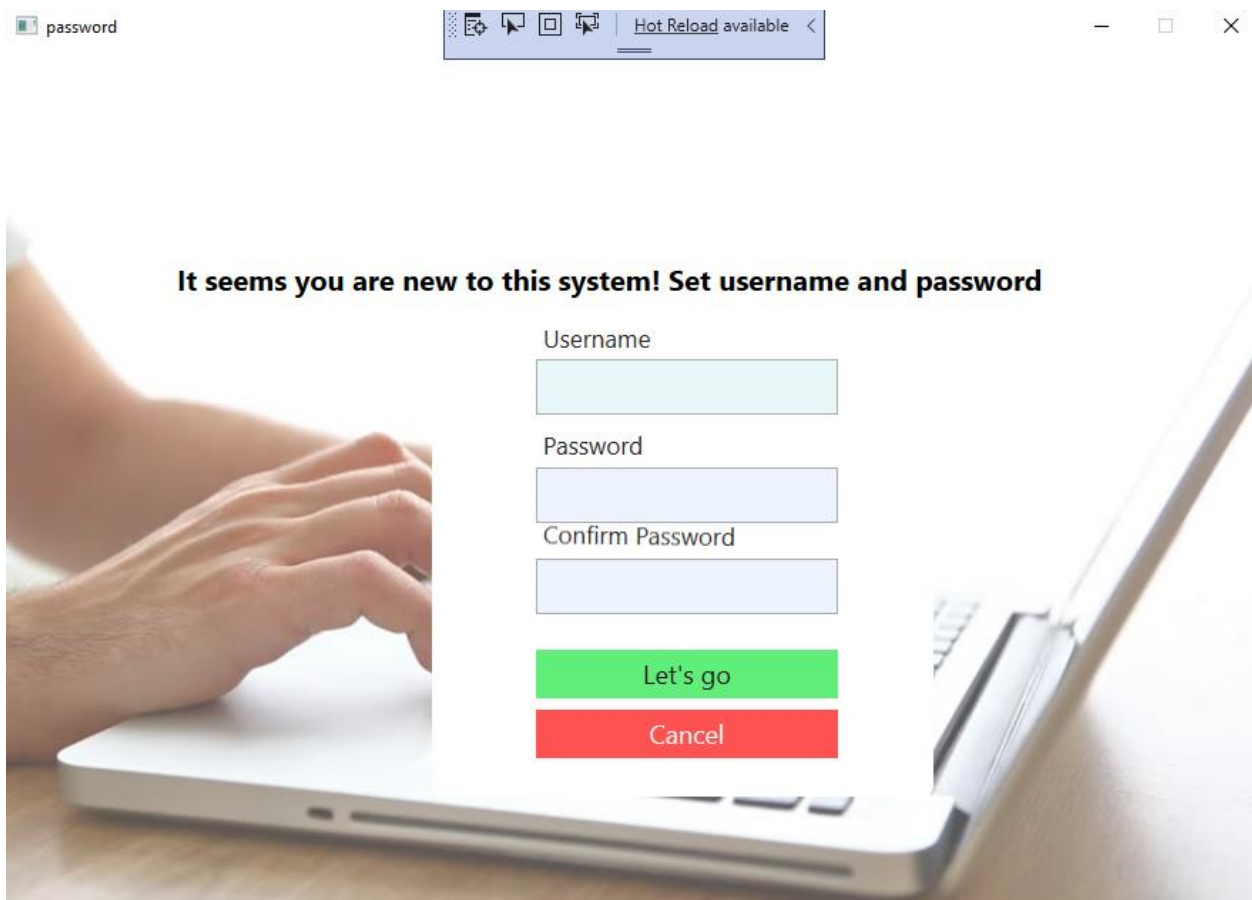
1.2 Proposed System

The proposed system is digitized system which is specially designed to overcome problem mentioned above. The system ensures security with the presence of student registration management section. Entry of data and display of data have been made easy with the presence of easy user-interface

2. Detail Instruction to run program

1) Step 1: setup username and password

As user start the program, the very first step in this program is to login to system. The program detects whether the user is new or not. The program displays username and password setup form for new user, once the user set username and password they need not need to set username and password again. To setup new password, system provides password confirmation field to enhance security. User can quit the program by clicking exit button. We can see the username and password setup form below in figure 1:



The image shows a web browser window titled 'password'. The browser's address bar shows 'Hot Reload available' and navigation icons. The main content area displays a form with the heading 'It seems you are new to this system! Set username and password'. The form includes three input fields: 'Username', 'Password', and 'Confirm Password'. Below these fields are two buttons: a green 'Let's go' button and a red 'Cancel' button. The background of the image shows a person's hands typing on a laptop keyboard.

Figure 1: Start up window

2) Step 2: login to the system

After username and password setup, user needs to login to the system by entering username and password they created in setup window. If user forget username and password user can reset it by clicking 'forget password' button. The 'forget password' button opens new password setup window where user change username and password. User can quit the process by simply clicking exit button. The login window is shown below in figure 2 and password change window is shown in figure 3:

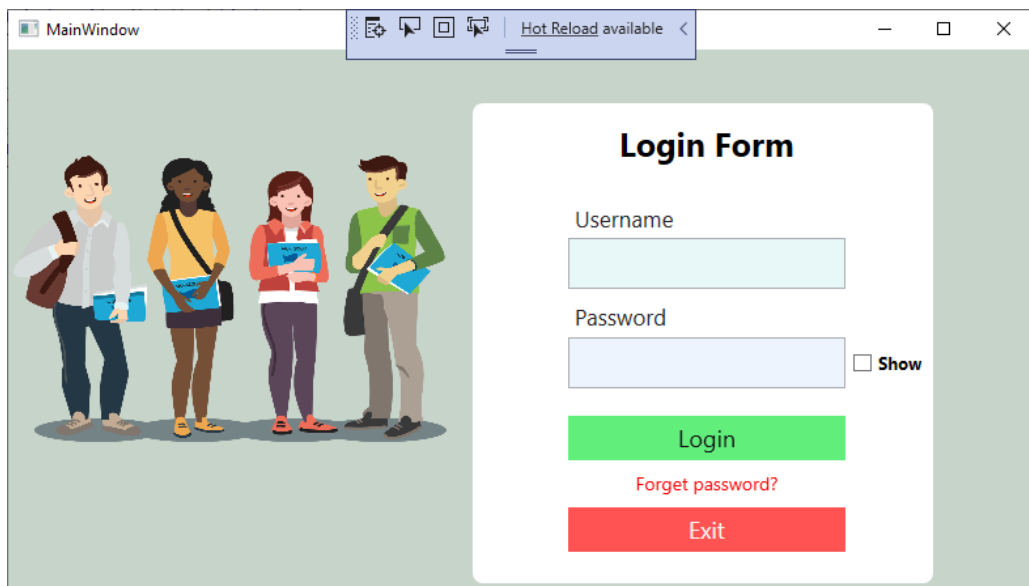


Figure 2: Login window

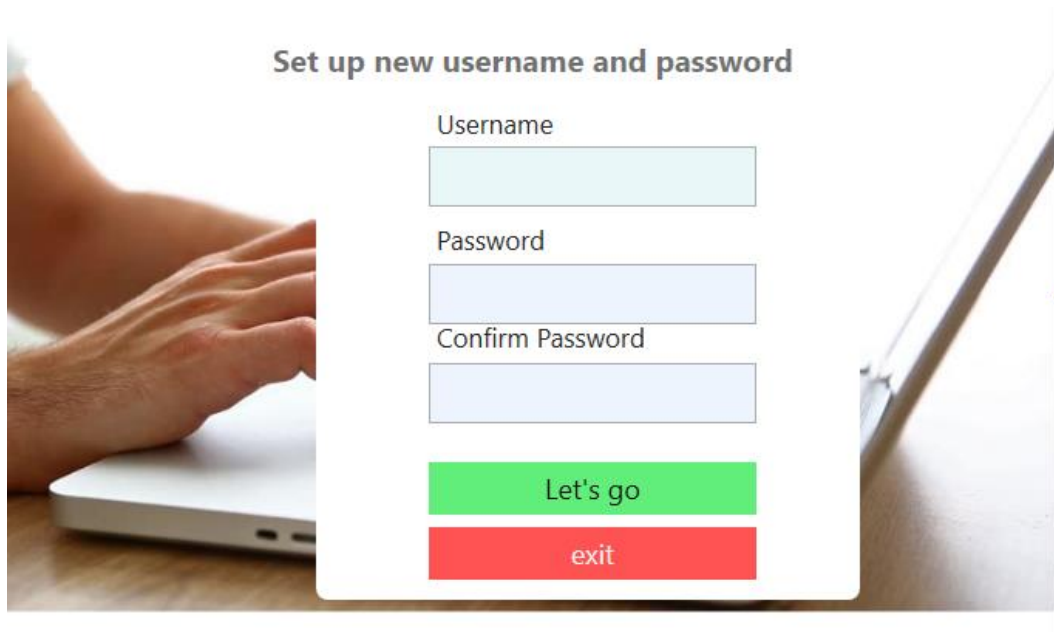


Figure 3: Forget password window

3) Step 3: Home Window

After login to the system program opens a new window which is home window of program which contains 5 buttons i.e. 'change password' button, 'register new student' button, 'view student report button', 'change password' button, 'logout' button and 'exit' button. Program home window is shown below in figure 4:



Figure 4: Home window

Let's discuss from the first, the if user wish to change their password they can click 'change password' button. 'change password' button opens new window to change password. Every time user change password user needs to login again to system. The password change window is shown below in figure 5:

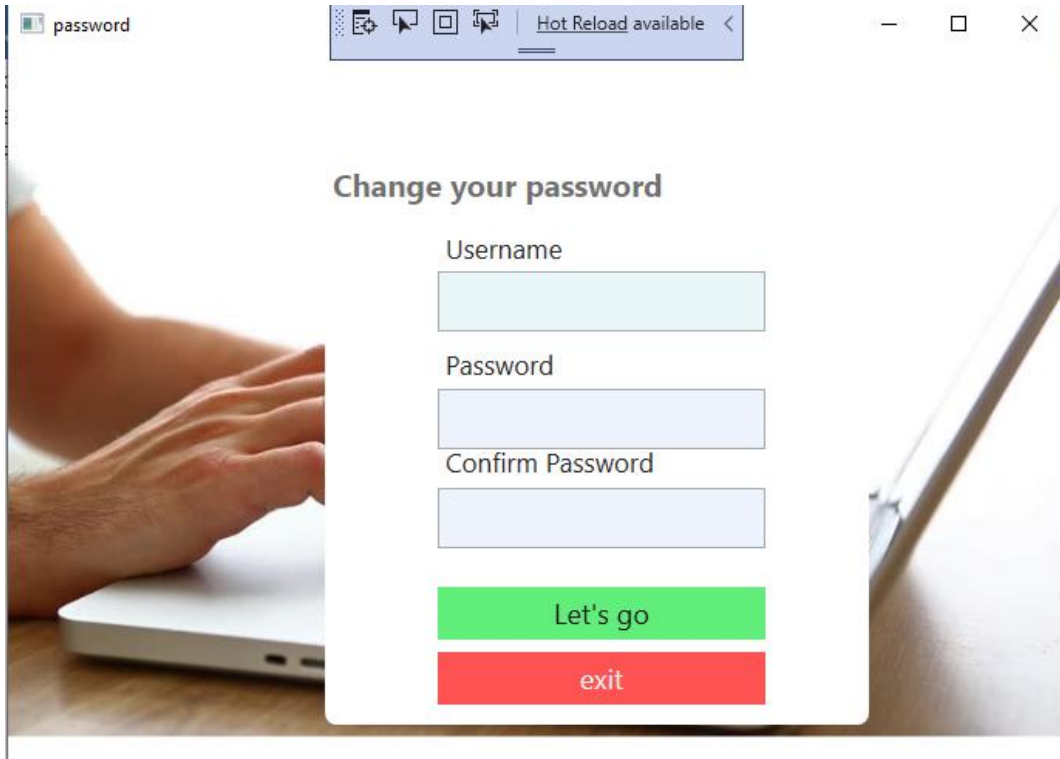


Figure 5: Password change window

4) Step 4: Register student

To register new student, user have to click 'Register Student' button. The button opens new window where two more option on register are provided. i.e. 'Register individual student' button and 'Register from existing file' button. If user wish to register individual student user can click 'Register Individual' and if user wish to register student data from existing file they can click 'Register From Existing File' button. The new window opens by clicking 'Register Student' button is shown below in figure 6:

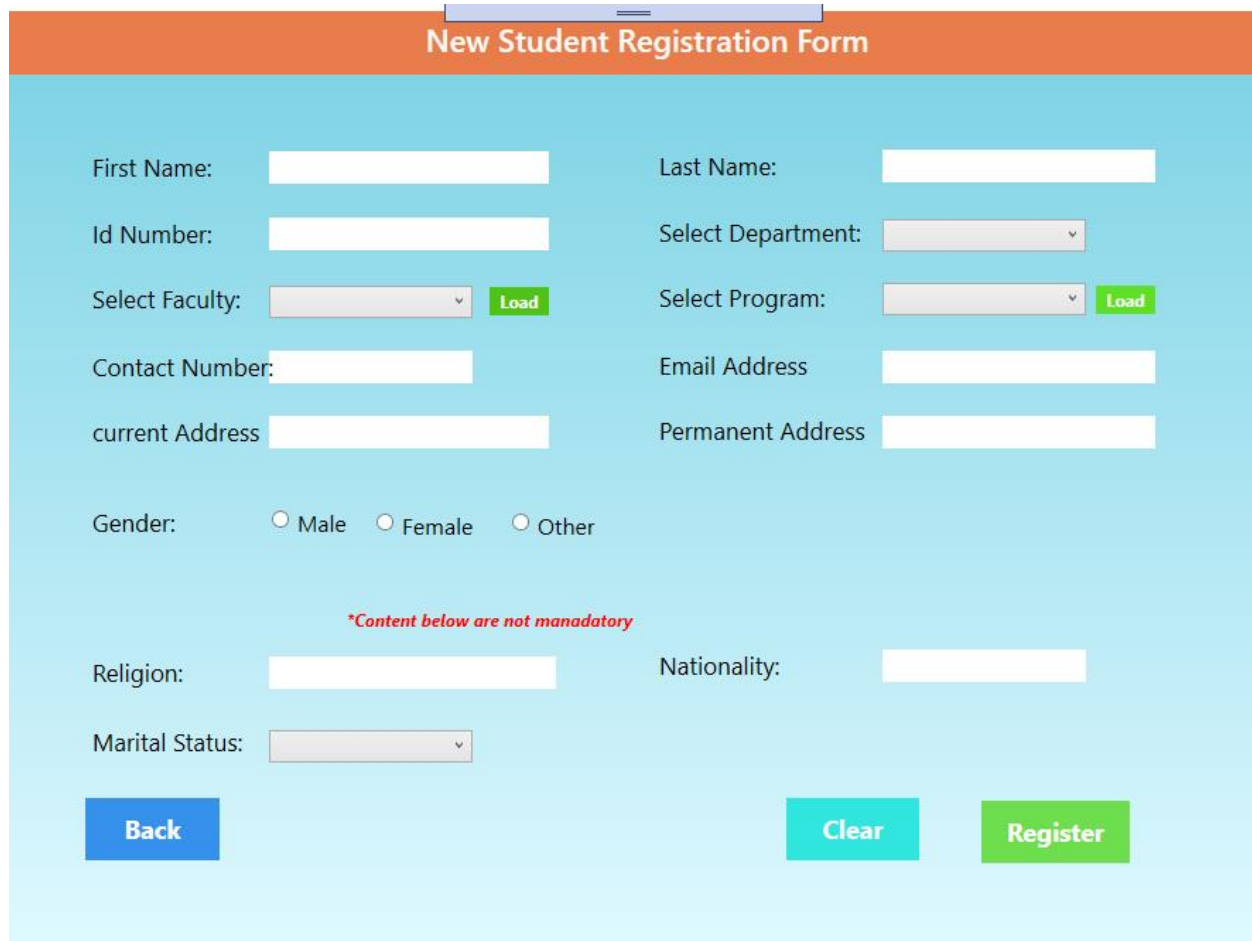


Figure 6: Register student Window

5) Step 5: Register Individual Student

After clicking 'Register Individual Student' button system opens a new window, which contains all the necessary input fields to register student. System have all necessary validation like email format validation, integer validation e.t.c. user is unable to register until all these validation get succeed. Integer filed like student phone number, student id accepts only integer value. Email id of student should be entered properly with valid email format otherwise registration process can't be done. In order to choose program, user first needs to choose department. In every department change, faculty and program are automatically changed. User can't choose faculty before choosing department and can't choose program until choosing faculty. After choosing department user needs to click load button near faculty choose field which load the relative faculty according to selected department. After choosing faculty user needs to choose program, to see available program user needs to click load button near program choose field which load the relative program according to selected faculty. The program contains further input fields like

nationality, marital status and religion which are not mandatory if user don't want to enter, they can simply leave these fields blank. After all successful validation and mandatory input done user can click register button to go further process. And user can clear the the input fields by clicking 'Clear' button. The new student register window is shown below in figure 7:



The image shows a web form titled "New Student Registration Form" with an orange header. The form is set against a light blue background and contains the following fields and controls:

- First Name:** Text input field
- Last Name:** Text input field
- Id Number:** Text input field
- Select Department:** Dropdown menu
- Select Faculty:** Dropdown menu with a green **Load** button next to it
- Select Program:** Dropdown menu with a green **Load** button next to it
- Contact Number:** Text input field
- Email Address:** Text input field
- current Address:** Text input field
- Permanent Address:** Text input field
- Gender:** Radio buttons for **Male**, **Female**, and **Other**
- *Content below are not mandatory** (Red italicized text)
- Religion:** Text input field
- Nationality:** Text input field
- Marital Status:** Dropdown menu
- Buttons:** A blue **Back** button, a cyan **Clear** button, and a green **Register** button.

Figure 7: Student register window

After Clicking 'Register' button system opens 'registration confirmation' window, which is provided to avoid mistakes and to ensure that the user entered data is actually correct. If user finds entered data mistake, user can abort process by clicking 'Cancel' button. If all data are correct and user wish to register data, user can enter 'Register' button which

save student data in xml file. The student data confirmation window is shown below in figure 9:

Register_confirm

Hot Reload available

Student Detail Confirmation Form

Name:	Ashish Bhandari
Id Number:	17031918
Department:	Management
Faculty:	BBA
Program:	BBA course 1
Contact Number:	986913342
Email Address:	ashish@gmail.com
current Address:	Ratnachowk-07, Pokhara
Permanent Address:	Kalika-28, Pokhara
Gender:	Male
Religion:	Hindu
Nationality:	Nepali
Marital Status:	

Cancel **Register**

Figure 8: Student Data Confirmation Window

6) Step 6: Register From Existing File

To register student from existing file, user have to click “Register From Existing File” button. After clicking ‘Register From Existing File’ button system opens a new window, which contains file browse button, data preview panel, back button and register button. First user have to click on ‘choose’ button to browse the csv file. System accepts only csv file. After selecting csv file, system displays all it’s data to provide data convince to user. If user wish to register all the student data, user can click ‘Register All’ button. And user can abort process by clicking ‘cancel’ button. The ‘Register from existing file’ window is shown below in figure 9:

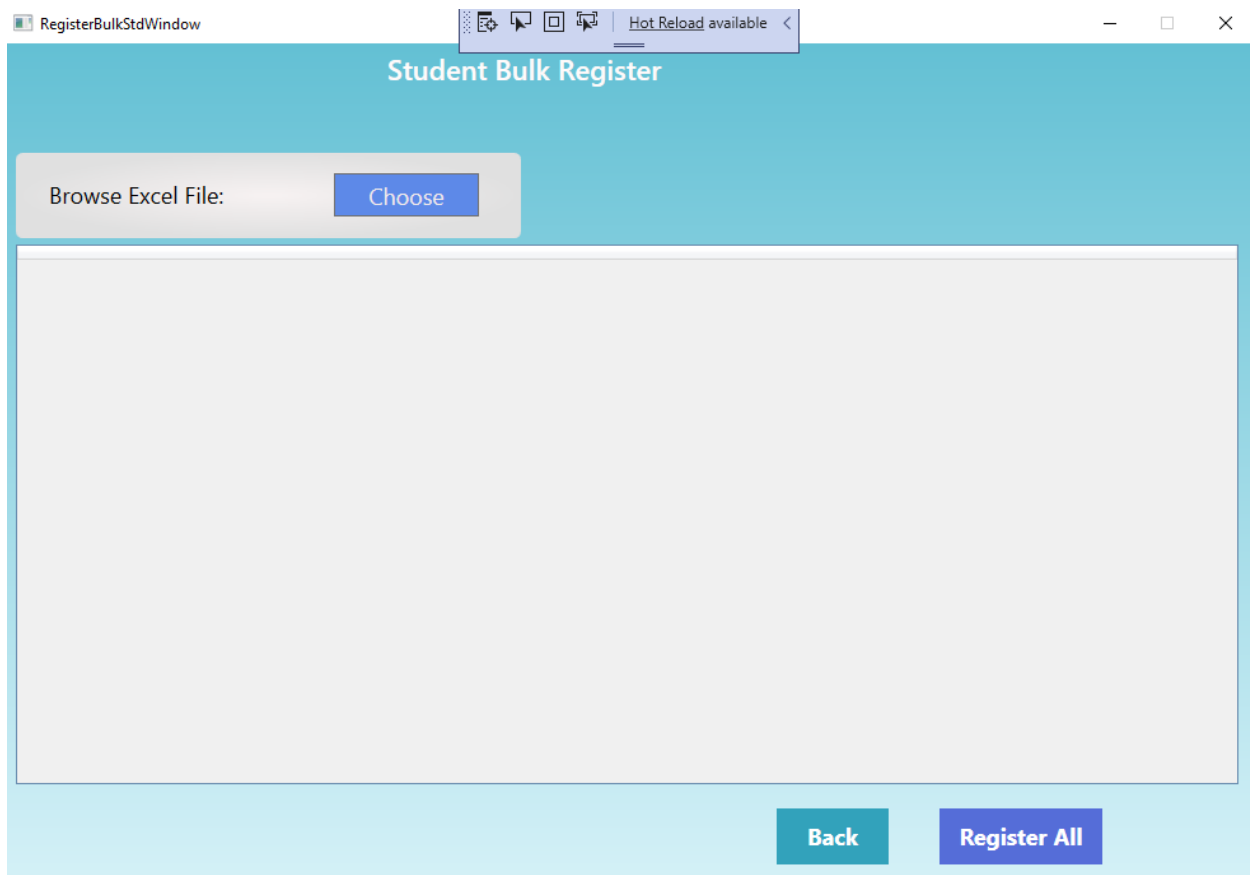


Figure 9: Register From Existing File Window

7) Step 7: View Report

To view registered student report, user have to click 'View Student Report' button. The button opens new window where three buttons are provided. That are 'View Weekly Student Registration Report' button, 'View Chart' button and 'View All Registered Student' button. If user wish to view weekly student registration report, user can click are 'View Weekly Student Registration Report' button if user wish to view graphical chart to view of weekly student registration data, user can click 'View Chart' button and if user wish to view all registered student data, user can click 'View All Registered Student' button. The 'View Report' window is shown below in figure 10:

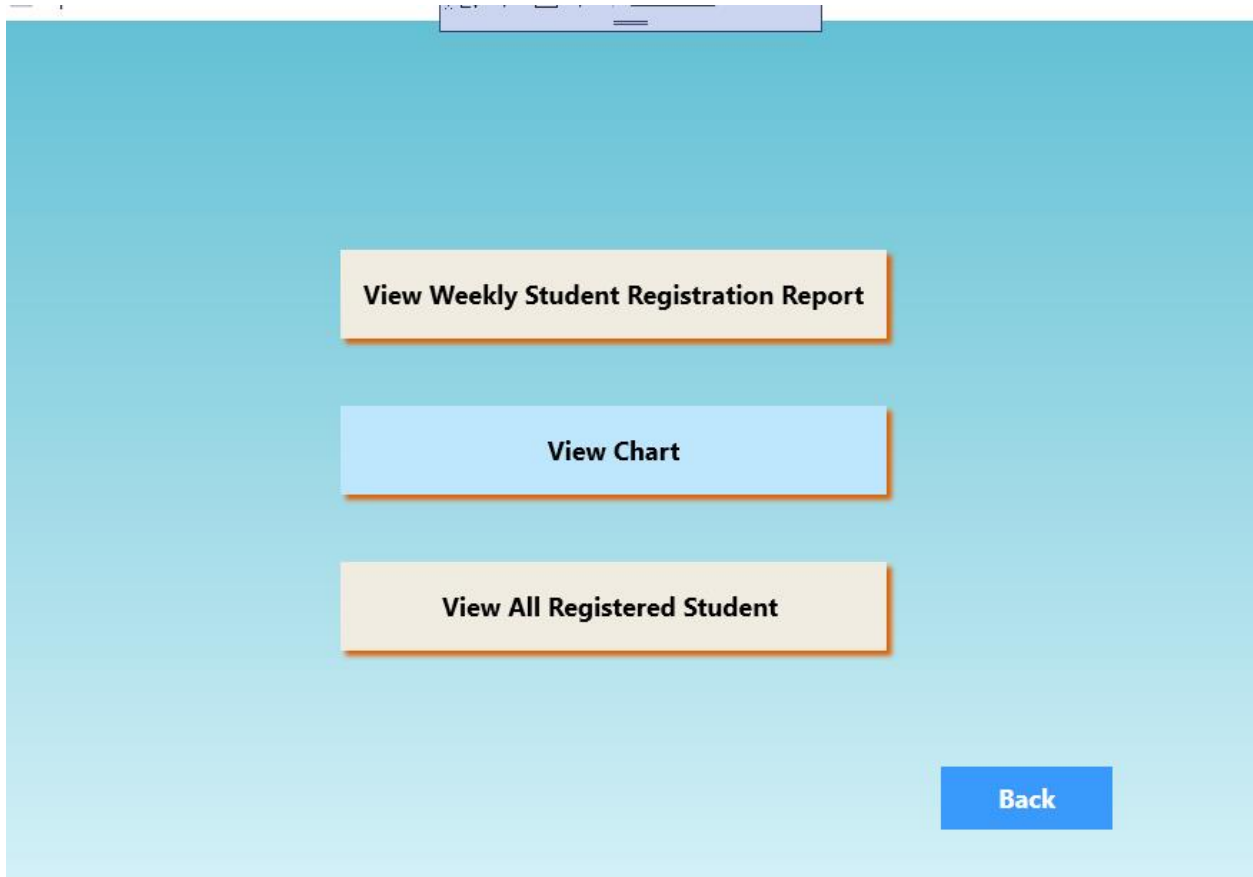


Figure 10: View Student Data Window

8) Step 7: View Weekly Student Registration Report

To view weekly student registration report, user have to click 'View Weekly Student Registration Report' button. The button opens new window where weekly student registered data are displayed. The 'View Weekly Student Registration Report' window is shown below in figure 11:

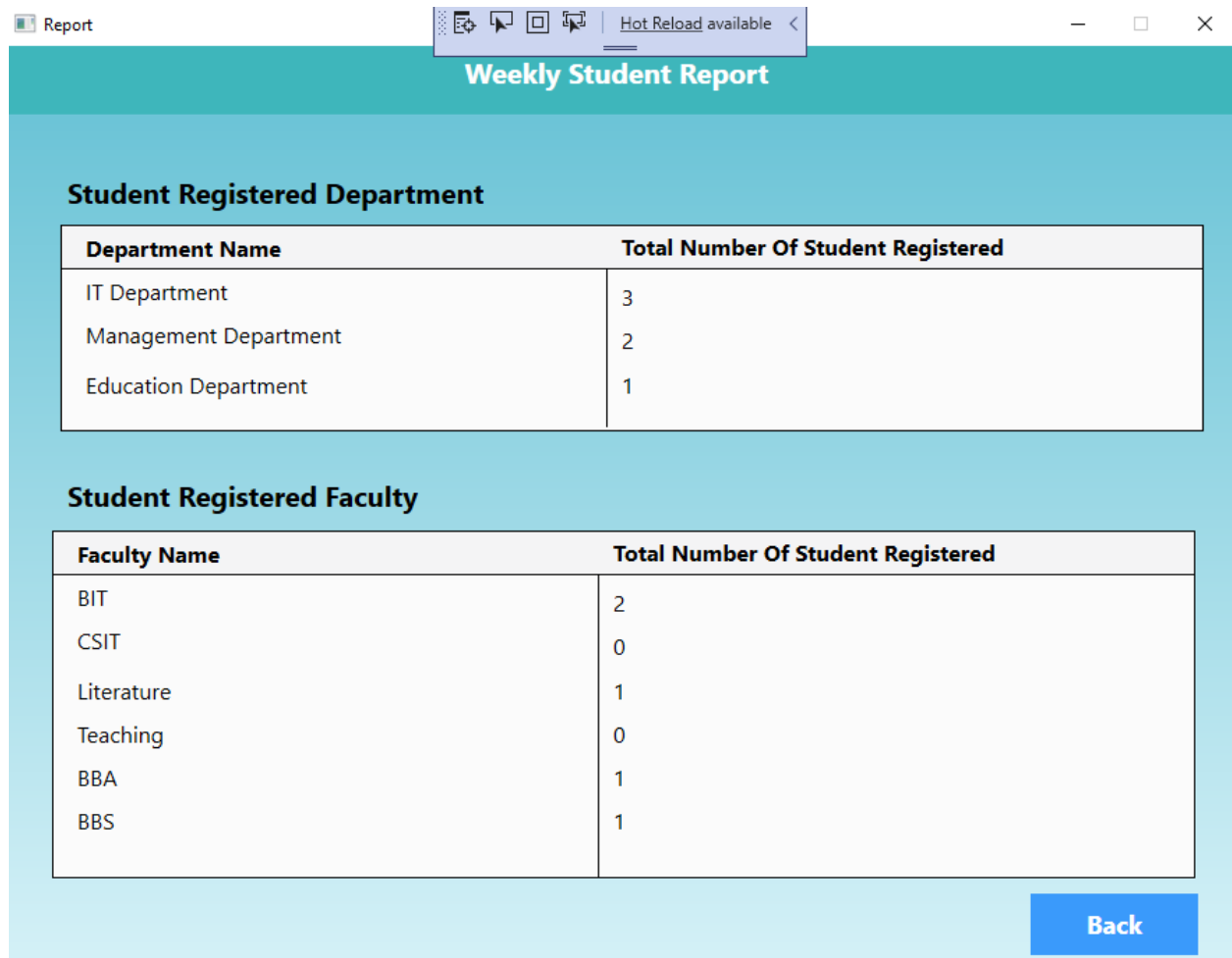


Figure 11: Weekly Student Registration Report

9) Step 9: View Chart

To view graphical student registration chart, user have to click 'View Chart' button. The button opens new window where weekly student registered data are displayed in graphical bar chart. The 'View Chart' window is shown below in figure 12:

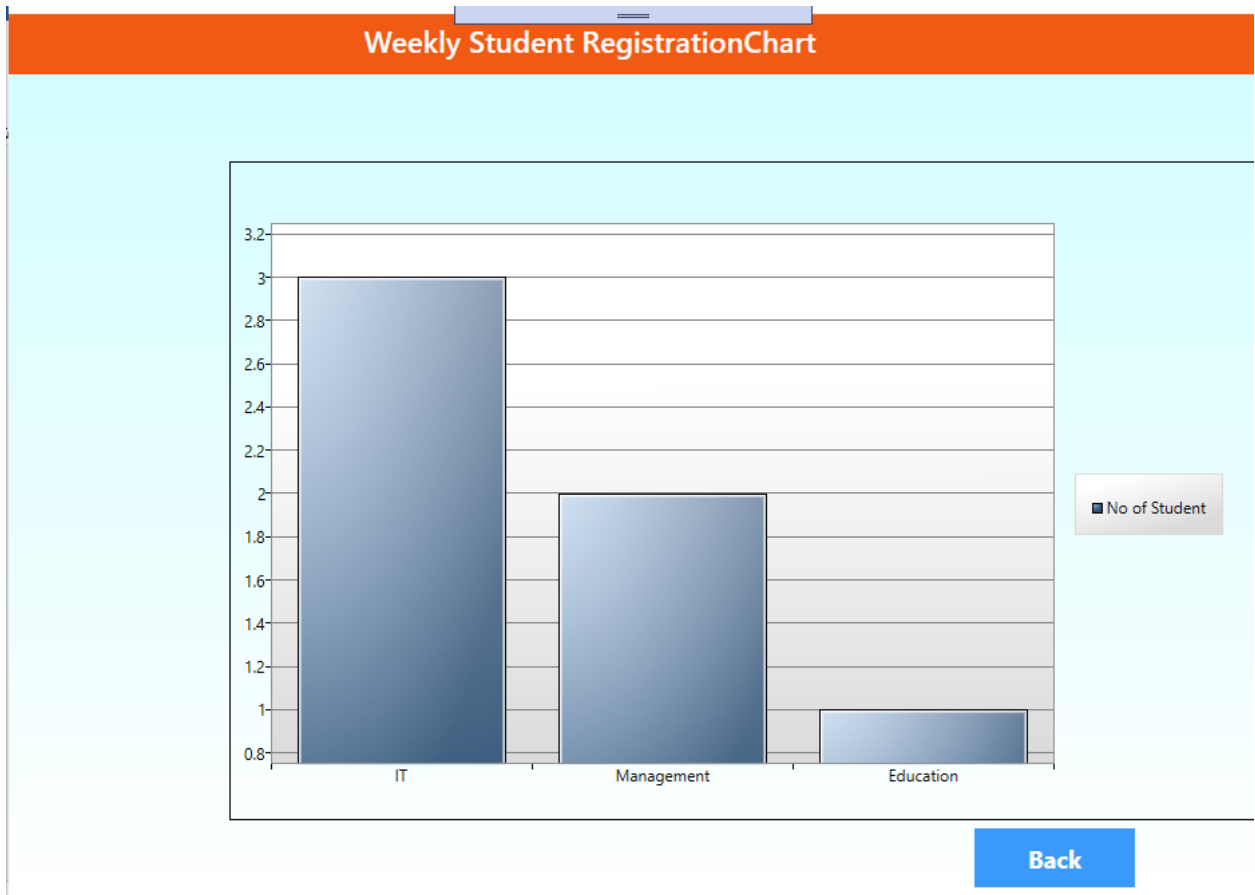


Figure 12: Graphical Chart

10) Step 10: View All Registered Student Data

To view all registered student data, user have to click 'View All Registered Student' button. The button opens new window where all registered student data are displayed in a tabular form. User can view data in ascending and descending order and can sort data by 'Name' and 'Registration Date'. The 'View All Registered Student' window is shown below in figure 12:

The screenshot shows a web application window titled "Student Report Form". It features a table with 12 columns: Name, StudentID, Department, Faculty, Program, Phone, Email, Gender, CurrentAddress, PermanentAddress, Religion, and Nationality. The table contains 7 rows of student data. Above the table, there are controls for "Edit Mode" (ON/OFF) and "Sort by" (Student Name, Ascending/Descending). A "Back" button is located at the bottom right.

Name	StudentID	Department	Faculty	Program	Phone	Email	Gender	CurrentAddress	PermanentAddress	Religion	Nationality	Marital Status
Aliz Tiwari	17031910	IT	BSCIT	Networking	9823651	aliz@gmail.com	Male	Pokhara-31	Pokhara-31	Hindu	Nepali	Unmarried
Aliza Tiwari	17031914	Management	BBS	Networking	9823851	aliza@gmail.com	Female	Pokhara-32	Pokhara-32	Hindu	Nepali	Unmarried
Ashish Bhandari	17031918	IT	BIT	Networking	9823451	ashish@gmail.com	Male	Pokhara-28	Pokhara-28	Hindu	Nepali	Unmarried
Kopila Bishyal	17031913	education	BIT	Networking	9823458	kops@gmail.com	Female	Butwal-04	Butwal-04	Hindu	Nepali	Unmarried
saimon Paudel	17031956	IT	BIT	Networking	9823458	saimon@gmail.com	Male	Pokhara-34	Pokhara-34	Hindu	Nepali	Unmarried
Suman Bhandari	17031991	Management	BBA	Networking	9823464	suman@gmail.com	Male	Pokhara-29	Pokhara-29	Hindu	Nepali	Unmarried
yaman Thapa	17031917	IT	LITERATURE	Networking	9823651	yaman@gmail.com	Male	Pokhara-30	Pokhara-30	Hindu	Nepali	Unmarried

Figure 13: View All Student Data

3. Detailed description of the classes, properties and methods

3.1. Login class (LoginWindow.xaml.cs)

Methods	Description
Login_button	It is login button click listener, it is used to validate username password and redirect user to main window.
Exit_button	It is Exit button click listener, it is used to terminate the program.
Check_file	This method is used to find whether user is new or not.
Forget_pw	It is forget password button click listener, it is used to open new window to setup new username and password.

3.2. Password class (password.xaml.cs)

Methods	Description
register_button	It is register button click listener, it is used to store new username password of user.
Exit_button	It is Exit button click listener, it is used to terminate the Current Window.

3.3. Main window class (MainWindow.xaml.cs)

Methods	Description
register_student	It is 'register student' button click listener, it is used to open student registration window.
View_report	It is 'view report' button click listener, it is used to open view report window.
Logout_click	It is 'Log Out' button click listener, it is used to redirect user to login window.
Change_pw_click	It is 'Change Password' button click listener, it is used to redirect user to password setup window.
Exit_button	It is Exit button click listener, it is used to terminate the program.

3.4. Register Individual student window class (Register_student.xaml.cs)

Methods	Description
register_student	It is 'register student' button click listener, it is used to open student registration confirmation window.
departmentBox_SelectionChanged	It is used to handle change in department
dropdown_loader	It is used to load relative course and faculty according to department choose
validator	It is used to validate user input
passValue	It is used to pass user input data to another window
clear_btn	It is used to reset all the input fields
Exit_button	It is Exit button click listener, it is used to terminate the Current Window.

3.5. Register Confirmation window class (regConfirm.xaml.cs)

Methods	Description
register_button	It is 'register' button click listener, it is used to write student data in xml format.
cancel_button	It is 'cancel' button click listener, it is used to terminate the Current Window.

3.6. Register Bulk Data window class (registerBulkStd.xaml.cs)

Methods	Description
Save	It is used to write all csv data in xml file
fileChooseBtn_Click	It is used to browse csv file and display its data in datagrid
AddData	It is used to store all csv data in data structure
cancelBtn_Click	It is used to terminate current window

3.7. Weekly Student Report Window class (WeeklyReportWindow.xaml.cs)

Methods	Description
LoadData	It is used to load xml data and display it in datagrid
cancelBtn_Click	It is used to terminate current window

3.8. All Registered Student Report Window class (AllStudentWindow.xaml.cs)

Methods	Description
LoadData	It is used to load xml data and display it in datagrid
sortData_SelectionChanged	It is used to sort data by name and registration date
ascending_Checked	It is used to display data in ascending order
descending_Checked	It is used to display data in descending order
cancelBtn_Click	It is used to terminate current window

3.9. Graphical chart Window class (AllStudentWindow.xaml.cs)

Methods	Description
LoadData	It is used to load xml data and passing value to chart
cancelBtn_Click	It is used to terminate current window

4. Class Diagram

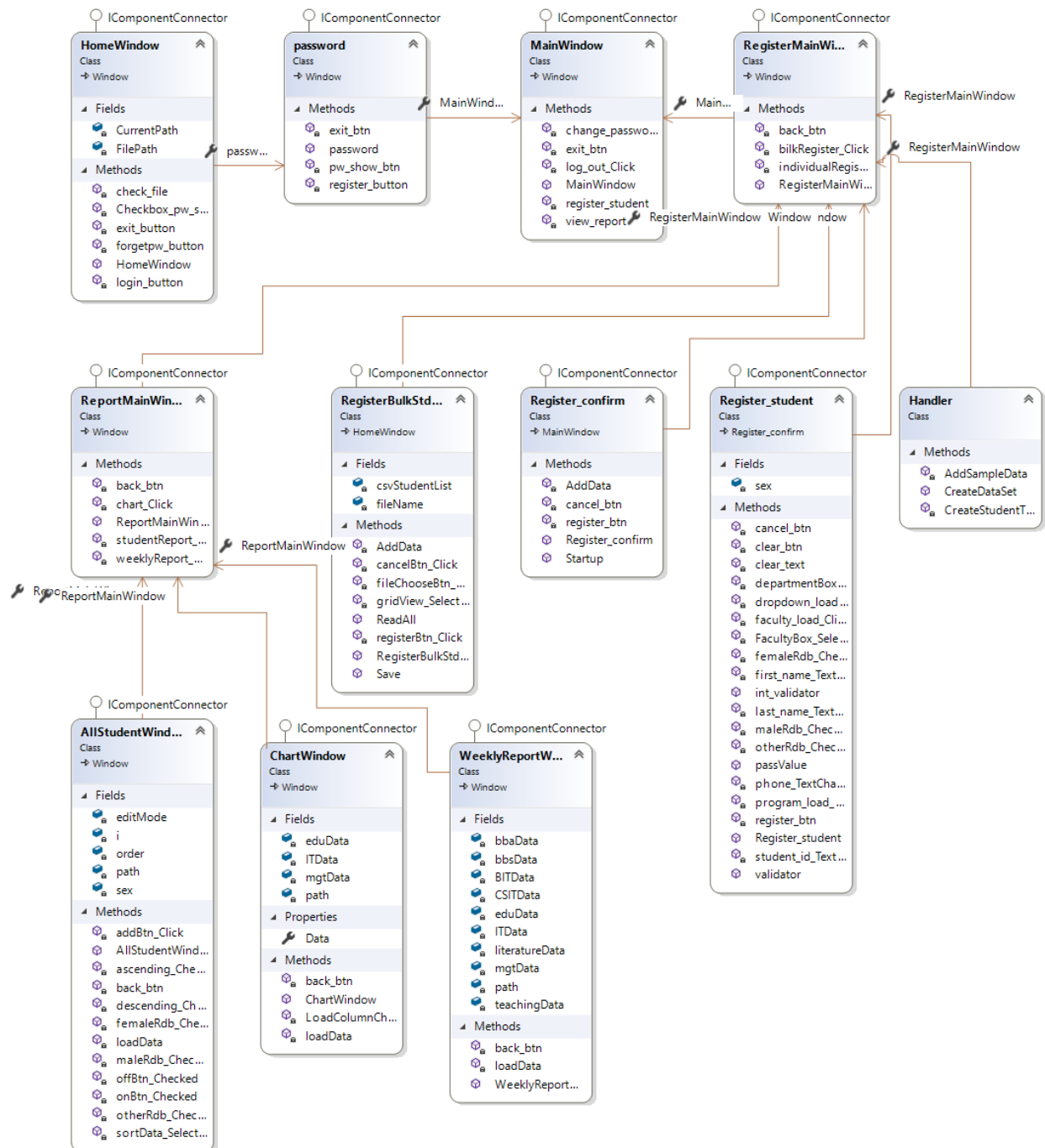


Figure 14:Class Diagram

5. System Architecture

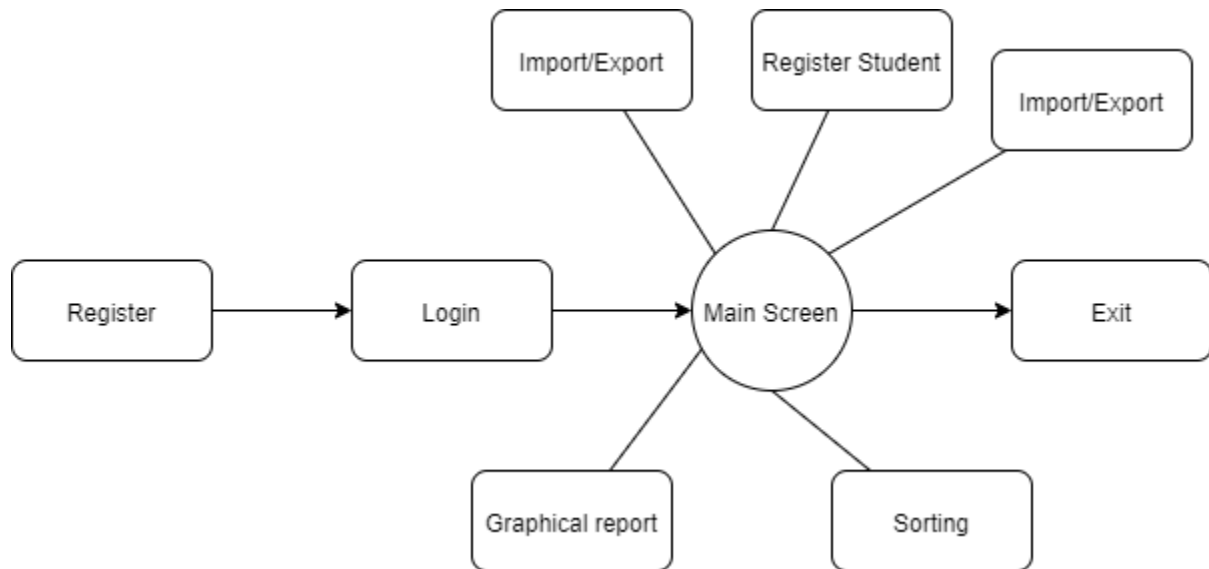


Figure 15: System Architecture

6. Flow Chart

6.1. Flow chart of Program setup

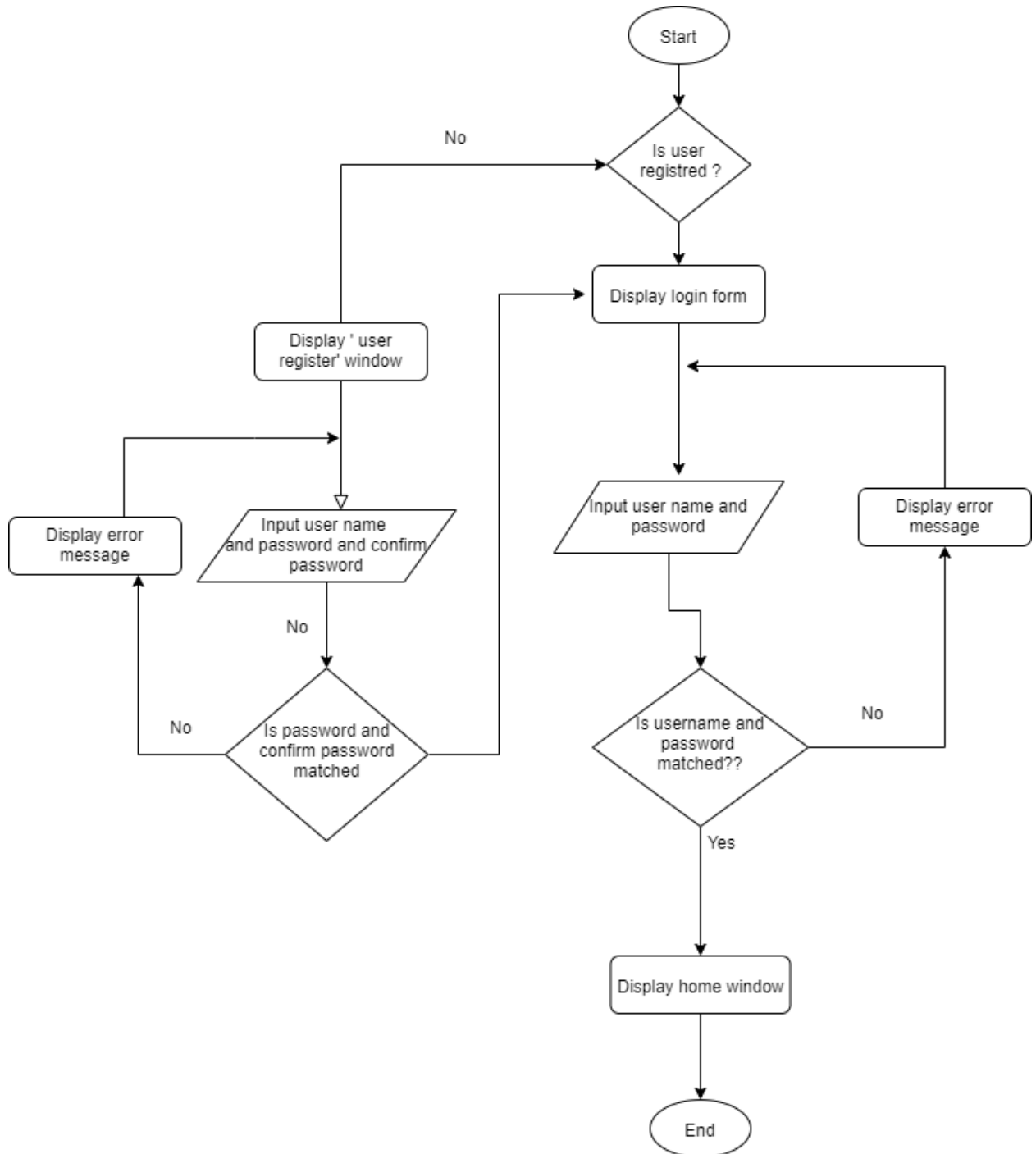
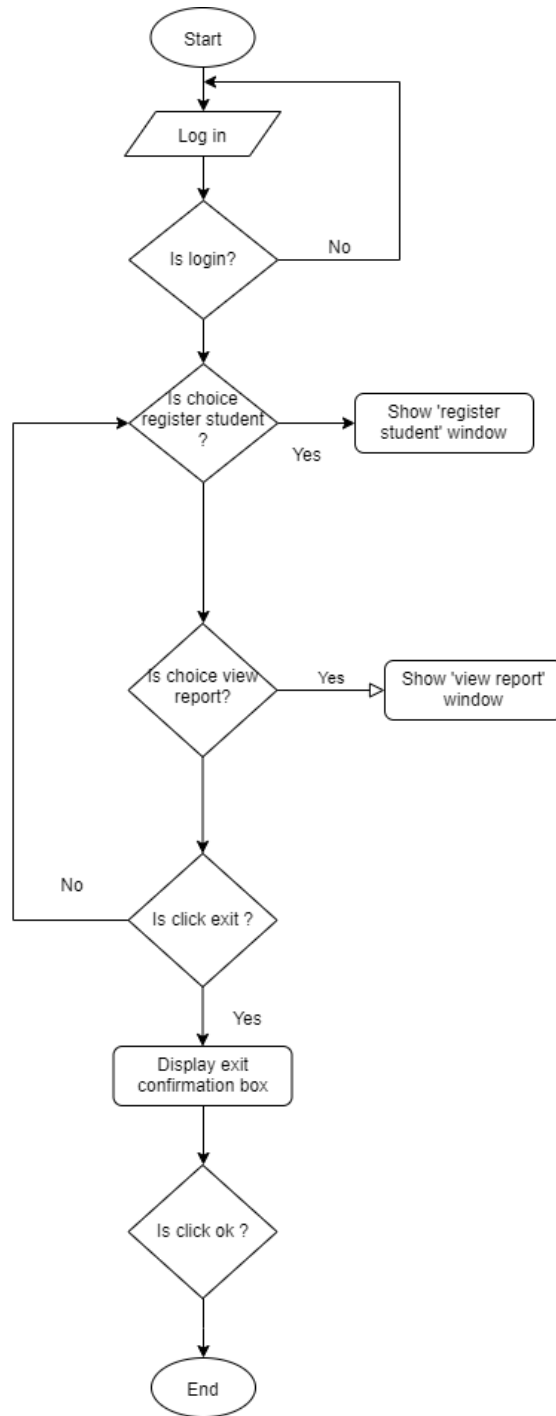


Figure 16: Flow chart of program startup

6.2. Flow chart of home window*Figure 17: Home window Flowchart*

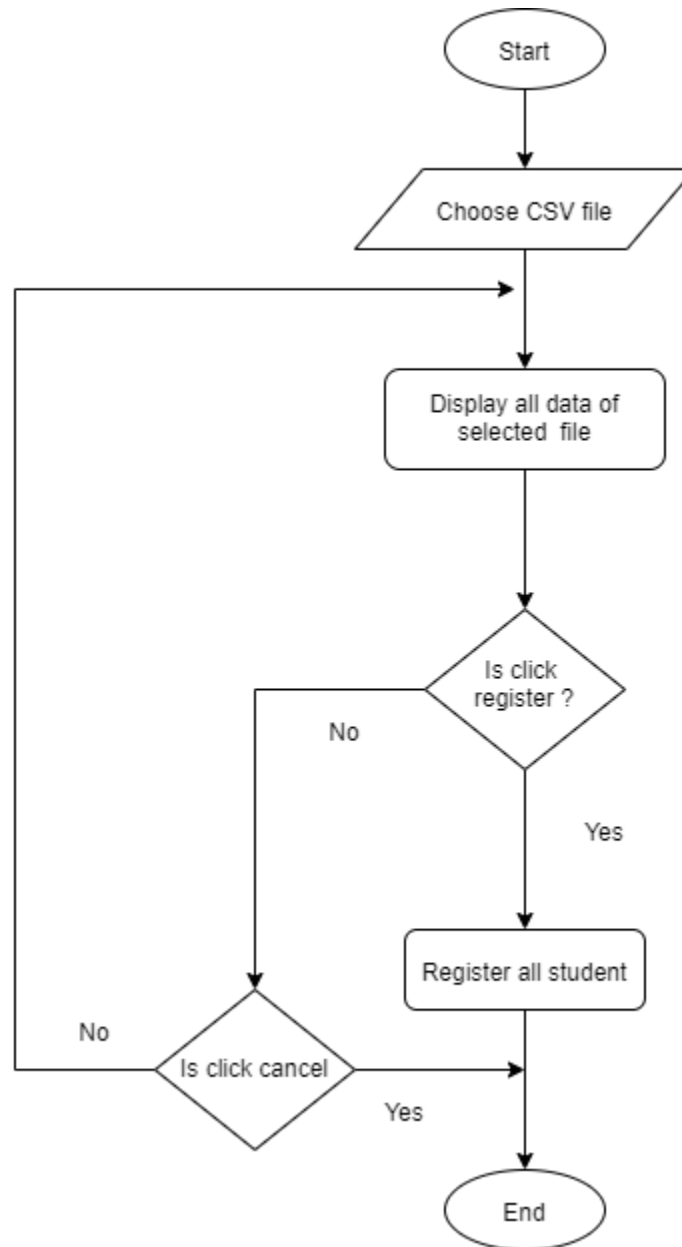
6.3. Flow chart of Bulk Register from file

Figure 18:Flow chart of Bulk Register From file

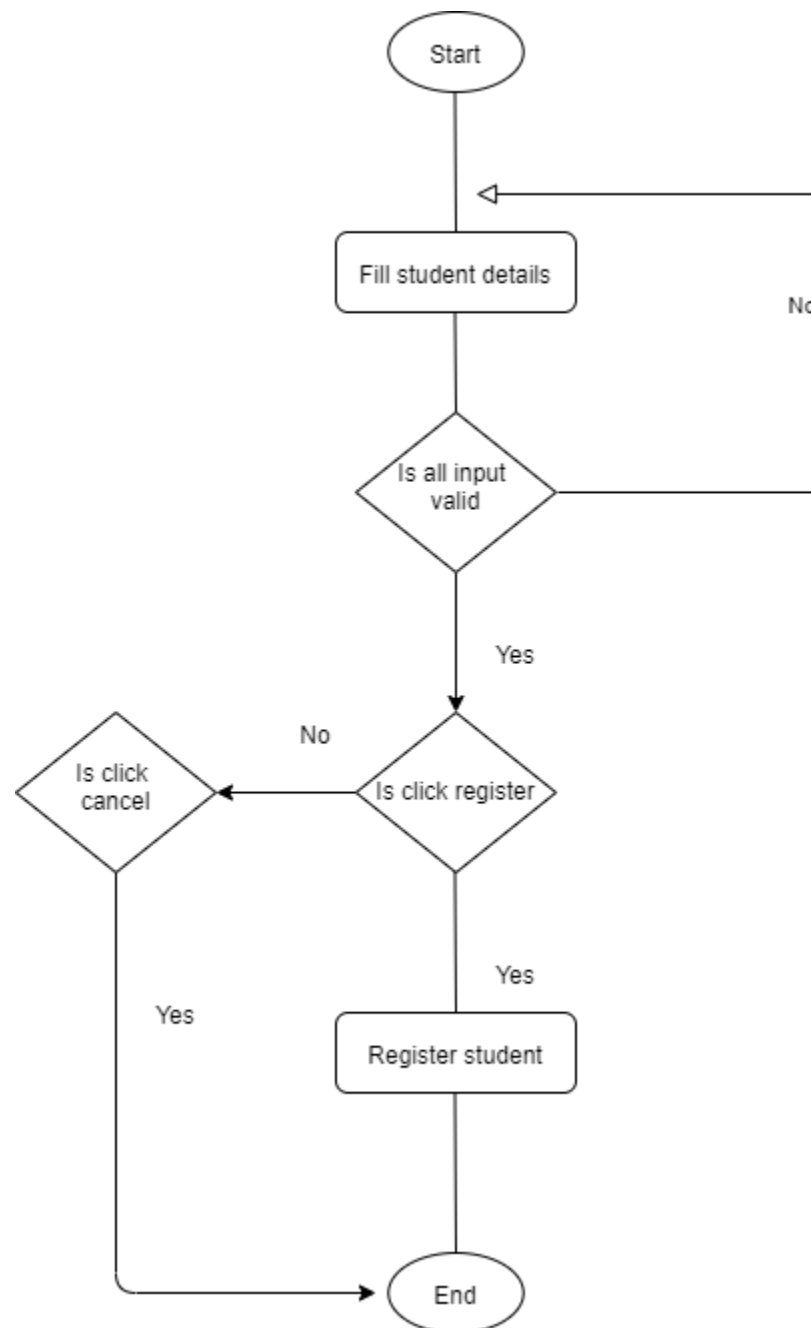
6.4. Flow chart of Individual Student Register

Figure 19: Flow chart of individual student register

7. Algorithms of Reports

7.1. Individual Student Enrolment Algorithm Steps:

1. Start
2. Check whether the user input data is valid or not
3. If not valid, display error message and avoid registration
4. If valid, show confirmation window and ask to register confirm or not
5. If user enter confirm register, write student data in xml file
6. Stop

7.2. Bulk Student Enrolment Algorithm Steps:

1. Start
2. Browse csv file
3. If browse file process aborted, display error message and close current window and open home window
4. If file browsed successfully, show confirmation window and ask to register confirm or not
5. If user enter confirm register, write student data in xml file
6. Stop

7.3. Sorting algorithm:

In this program bubble sort algorithm is use to sort student name and registration date. The detail workflow of bubble sort algorithm is described below.

1) Bubble Sort Algorithm:

Bubble sort is a simple sorting algorithm. This sorting algorithm is comparison-based algorithm in which each pair of adjacent elements is compared and the elements are swapped if they are not in order (tutorialspoint, 2020).

Working of bubble sort algorithm:

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



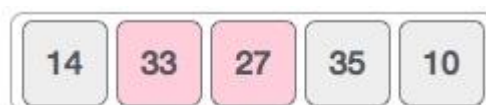
Bubble sort starts with very first two elements, comparing them to check which one is greater.



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



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



The new array should look like this –



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



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



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



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



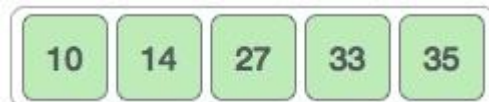
To be precise, we are now showing how an array should look like after each iteration. After the second iteration, it should look like this –



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



And when there's no swap required, bubble sorts learns that an array is completely sorted.



8. Data structure used in program

ring development of this project, many data types like list, array and algorithm like bubble sort algorithm is used.

1) List

When it comes to the list data structure, we should mention that it defines a sequential set of elements to which you can add new elements and remove or change existing ones (medium, 2020).

In program list is used in login window, where user name and password from text file stored in list. The reason for using list in program is mention below:

- The need for this data structure is most obvious when we have the urge to store items without knowing how much space we'll need in advance.
- Also, when we often have the need to add or get elements from a data structure, the array list would suit well because those operations are fast and efficient.

2) Array

An array is a collection of items stored at contiguous memory locations. The idea is to store multiple items of the same type together. This makes it easier to calculate the position of each element by simply adding an offset to a base value, i.e., the memory location of the first element of the array (geeksforgeeks, 2020).

In this program array is used reading xml value, where program read data from xml file and store them in an array.

Here, are some reasons for using arrays:

- Arrays are best for storing multiple values in a single variable
- Arrays are better at processing many values easily and quickly
- Sorting and searching the values is easier in arrays

9. Reflection of own experience of using C# and Visual Studio

The development of this project is based in c# programming language and visual studio 2019. While working with c#, I have a lot of fun ones at that. Since it's similar to other C-type languages like C, C++, and Java, being fluent in C# will make learning the others a breeze. As far as programming languages go, C# is fairly simple to digest. It's a high-level language, and that means it is somewhat similar to English. It's also designed with ease of use as a priority, and it abstracts away most of the complex tasks like memory management and exception handling, enabling coders to learn it without frying their brains.

Among other things, C# is a language that strikes a very good balance between beginner's learning curve, language abstraction and expressiveness, code brevity/verbosity, code maintainability, refactorability and reusability, compiling time, run-time efficiency and predictability etc. making it suitable for many types of work and scenarios. C# language and relevant libraries are very well documented.

Let's talk about visual studio 2019, Visual Studio IDE is a full-featured development platform for multiple operating systems as well as the web and the cloud. It allows users to smoothly navigate the interface so they can write their code speedily and accurately.

With Visual Studio IDE, developers also have access to a host of debugging tools. These assist them in profiling bugs and diagnosing them simply. This way, they can deploy their applications confidently, knowing that they have done away with anything that could cause performance errors.

Although, it has many good points but I find some issue in visual studio 2019. It needs really high processor and high ram. When we run visual studio below 4gb ram, visual studio can not operate well, it consumes a lot of time to load in such low processor/ram laptops.

Thus, only one issue I experienced using visual studio is, it consumes a lot of ram. I think the solution to overcome this problem is, developing two version of 'visual studio'. One will be developed as light version for low processor computer, so that all the user who have light processor/ram computer can also use visual studio efficiently.

10. Test Case

10.1. Login validation test

- I. User name is entered and password is leaved empty, screen shot of output of the application is shown below:

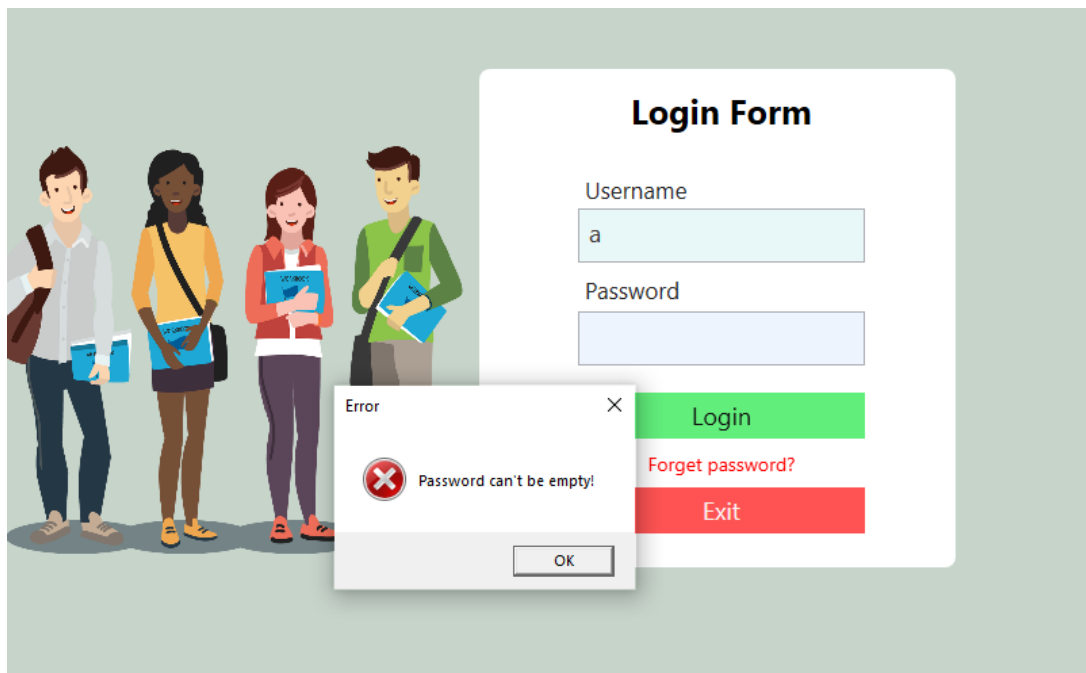


Figure 21: Empty password test

- II. wrong username and password is entered to see how system handle the situation, screenshot of output of the application is shown below:

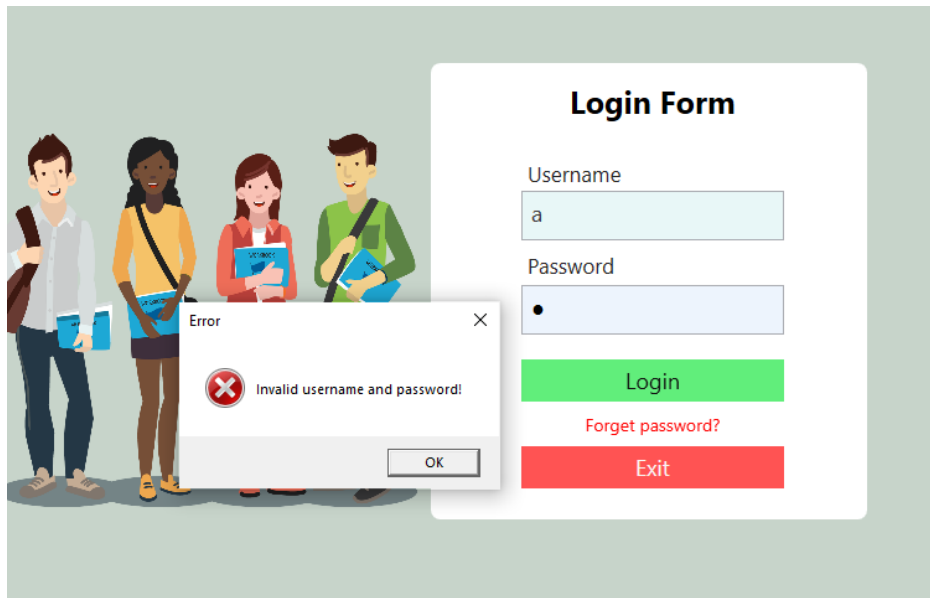


Figure 22: Wrong Username and Password test

- III. username and password is leaved empty to see how system handle the situation, screenshot of output of the application is shown below:

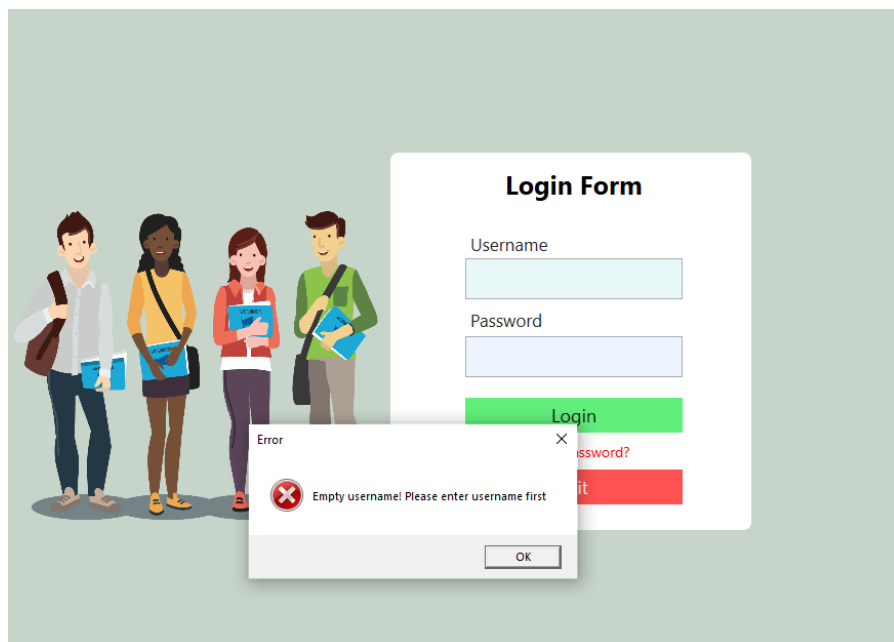
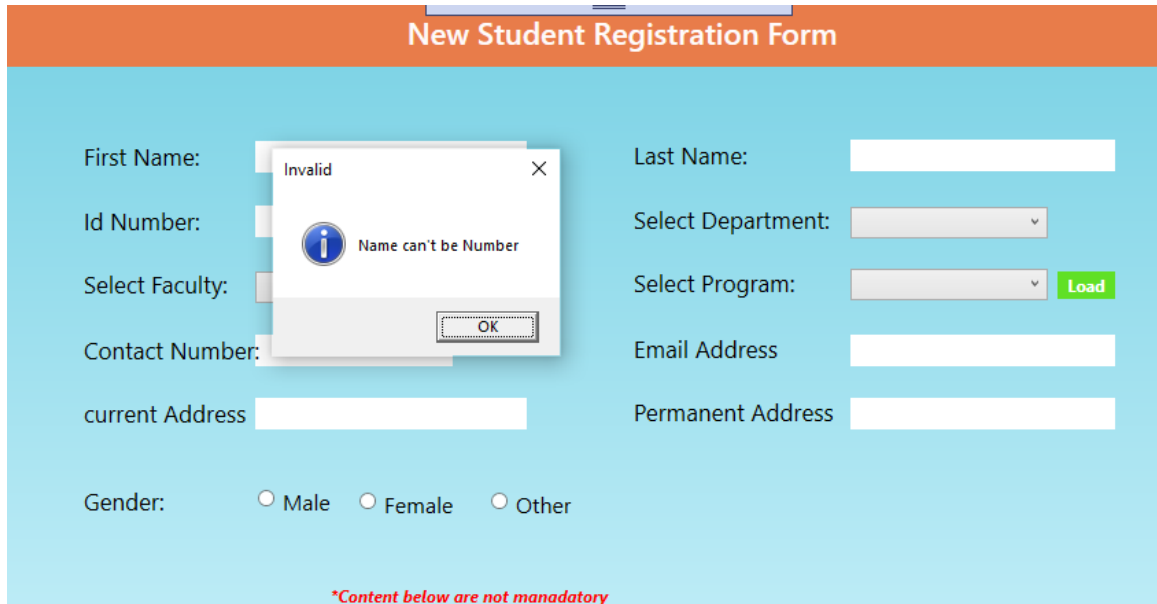


Figure 23: Empty username and password test

10.2. Student Registration validation test

- I. Here in name input field, integer value is entered to see how system handle the situation. screenshot of output of the application is shown below:



The screenshot shows a web form titled "New Student Registration Form" with an orange header. The form has a light blue background and contains the following fields and controls:

- First Name:
- Last Name:
- Id Number:
- Select Department:
- Select Faculty:
- Select Program:
- Contact Number:
- Email Address:
- current Address:
- Permanent Address:
- Gender: ☐ Male ☐ Female ☐ Other

A modal dialog box is displayed over the form, titled "Invalid" with a close button (X). It contains an information icon (i) and the message "Name can't be Number". There is an "OK" button at the bottom of the dialog.

*Content below are not manadatory

Figure 24: Entering number as name test

- II. Here in name input field, integer value is entered to see how system handle the situation. screenshot of output of the application is shown below:

The screenshot shows a web form titled "New Student Registration Form". The form has several input fields: "First Name" (filled with "Ashish"), "Last Name" (filled with "Bhandari"), "Id Number" (empty), "Select Faculty" (dropdown), "Contact Number" (empty), "current Address" (empty), "Gender" (radio buttons for Male, Female, Other), "Select Department" (dropdown), "Select Program" (dropdown), "Email Address" (empty), and "Permanent Address" (empty). There are "Load" buttons next to the "Select Faculty" and "Select Program" dropdowns. An error message dialog box is displayed over the "Id Number" field, titled "Invalid" with a close button (X). The message says "ID can't be String" with an information icon (i) and an "OK" button. At the bottom of the form, there is a red text note: "*Content below are not mandatory".

Figure 25: ID as string test

- III. Here, in contact number field, string value is entered to see how system handle the situation. screenshot of output of the application is shown below:

The screenshot shows the same "New Student Registration Form" as Figure 25. In this instance, the "Contact Number" field is the focus. An error message dialog box is displayed over the "Contact Number" field, titled "Invalid" with a close button (X). The message says "Number can't be String" with an information icon (i) and an "OK" button. The "Id Number" field is now empty. The "Load" buttons and the red text note at the bottom are still present.

Figure 26: Contact number as string test

- IV. Here, in email input field, wrong format email is entered to see how system handle the situation. screenshot of output of the application is shown below:

The screenshot shows a registration form on a light blue background. The form contains the following fields and controls:

- First Name:
- Last Name:
- Id Number:
- Select Department:
- Select Faculty:
- Select Program:
- Contact Number:
- Email Address:
- current Address:
- Gender: ☒ Male ☐ Female
- Religion:
- Nationality:
- Marital Status:

A modal dialog box is displayed in the center of the form with the title "incorrect format". It contains a yellow warning icon and the text "E-mail address format is not correct." with an "OK" button.

Below the Gender field, there is a red text label: **Content below are not manadatory*.

At the bottom of the form, there are three buttons: "Back" (blue), "Clear" (cyan), and "Register" (green).

Figure 27: Email validation test

- V. Here, all the required field are filled correctly and last name filed is leaved blank to see how system handle the situation. screenshot of output of the application is shown below:

The screenshot shows a web form titled "New Student Registration Form". The form has several input fields: First Name (Ashish), Last Name (empty), Id Number (17031918), Select Department (Management), Select Faculty (BBA), Select Program (Click Load button), Contact Number (986921389), Email Address (ashish@gmail.com), current Address (kalika), Address (kalka), Gender (Male selected), Religion, and Nationality. A red asterisk note below the Gender field states: "*Content below are not mandatory". An "Invalid" dialog box is displayed in the center, showing a warning icon and the message "Lastname is required". The dialog box has an "OK" button.

Figure 28: Empty input test

- VI. Here, in contact number field, only two integer value is entered to see how system handle the situation. screenshot of output of the application is shown below:

The screenshot shows the same web form as Figure 28, but with the Contact Number field containing the value "22". The Last Name field is now filled with "Bhandari". The "Invalid" dialog box is displayed in the center, showing a warning icon and the message "Contact number can't be too short". The dialog box has an "OK" button.

Figure 29: Short Contact number test

11. CONCLUSION

The initial coursework for the module CS6004NA Application Development was to build up student information system for an organization. It required a long time to build up the task in Visual Studio 2019 utilizing C# programming dialect. The framework has login screen to add security to the task. After login, the framework shows an main page where there are four clickable button along with logout and exit button after clicking any one button among the four the clickable button the clicked button goes to its respective page where the data are further inserted and send into XML and the data is saved into XML and the data is further retrieve to show the report and chart of the students. Aside from various shape components, class outline for every one of the structures and classes were utilized.

12. References

geeksforgeeks. (2020, 01 05). Retrieved from geeksforgeeks:

<https://www.geeksforgeeks.org/array-data-structure/>

medium. (2020, 01 1). Retrieved from medium: [https://medium.com/better-](https://medium.com/better-programming/data-structures-whats-a-list-ca04b0ba9fa2)

[programming/data-structures-whats-a-list-ca04b0ba9fa2](https://medium.com/better-programming/data-structures-whats-a-list-ca04b0ba9fa2)

tutorialspoint. (2020, 01 10). Retrieved from tutorialspoint:

[https://www.tutorialspoint.com/data_structures_algorithms/bubble_sort_algorithm](https://www.tutorialspoint.com/data_structures_algorithms/bubble_sort_algorithm.htm)
.htm

13. Appendix

Homewindow.xaml.cs

```
using System;  
using System.Collections.Generic;  
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 Student_info
{
    public partial class HomeWindow : Window
    {
        private string CurrentPath = System.AppDomain.CurrentDomain.BaseDirectory;
        private string FilePath = @"d:\Student_Management\password.dat";
        public HomeWindow()
        {
            InitializeComponent();

            //if folder exist system ignore this line
            string folderName = @"d:\Student_Management";

            System.IO.Directory.CreateDirectory(folderName);

            //check whether password file exists or not

            bool checkFile = check_file(FilePath);
            if (checkFile == false)
            {
                string msg = "It seems you are new to this system! Set username
and password";
                this.Hide();

                password pw = new password(msg);
                pw.Show();
            }
        }

        private void exit_button(object sender, RoutedEventArgs e)
        {
            MessageBoxResult messageBoxResult =
            System.Windows.MessageBox.Show("Are you sure wanna exit ?", "Exit Confirmation",
            System.Windows.MessageBoxButton.YesNo, MessageBoxImage.Warning);

            if (messageBoxResult == MessageBoxResult.Yes)
            {
                this.Close();
            }
            else
            {
                //
            }
        }
    }
}

```

```
    }  
}  
  
private void login_button(object sender, RoutedEventArgs e)  
{  
    string user_name = username_box.Text;  
    string password = password_box.Password;  
  
    string user;  
    string pw;  
  
    List<string> lines = File.ReadAllLines(FilePath).ToList();  
  
    foreach (var line in lines)  
    {  
        string[] entries = line.Split(',');  
        user = entries[0];  
        pw = entries[1];  
  
        if (user_name == "")  
        {  
            MessageBox.Show("Empty username! Please enter username  
first", "Error", System.Windows.MessageBoxButton.OK, MessageBoxImage.Error);  
        }  
        else if (password == "")  
        {  
            MessageBox.Show("Password can't be empty!", "Error",  
System.Windows.MessageBoxButton.OK, MessageBoxImage.Error);  
        }  
        else if (password == pw && user_name == user)  
        {  
            // this.Hide();  
            MainWindow mainWindow = new MainWindow();  
            mainWindow.Show();  
            this.Close();  
        }  
        else  
        {  
            MessageBox.Show("Invalid username and password!", "Error",  
System.Windows.MessageBoxButton.OK, MessageBoxImage.Error);  
        }  
    }  
}  
  
private bool check_file(string path)  
{  
    if (!File.Exists(path))  
    {  
        return false;  
    }  
    else  
    {  
        return true;  
    }  
}
```

```

        }
    }

    private void Checkbox_pw_show(object sender, RoutedEventArgs e)
    {

    }

    private void forgetpw_button(object sender, RoutedEventArgs e)
    {
        string msg = "                Set up new username and
password";

        password pw = new password(msg);
        pw.Show();
    }
}

```

Password.xaml.cs

```

using System;
using System.Collections.Generic;
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 Student_info
{
    /// <summary>
    /// Interaction logic for password.xaml
    /// </summary>
    public partial class password : Window
    {
        public password(string msg)
        {
            InitializeComponent();
            header_box.Content = msg;
            header_box.IsEnabled = false;
        }

        private void register_button(object sender, RoutedEventArgs e)
        {
            string pw1 = password_box.Password;
            string pw2 = password_box_Copy.Password;
            if (pw1 == pw2)
            {
                string path = @"d:\Student_Management\password.dat";

                using (StreamWriter sw = File.CreateText(path))
                {
                    sw.Close();
                    //write text
                    string username = username_box.Text;

                    using (StreamWriter streamWriter = new StreamWriter(path, true))
                    {
                        streamWriter.WriteLine($"{username },{pw1}");
                        streamWriter.Close();
                    }
                }

                this.Hide();
                Homewindow hmWindow = new Homewindow();
                hmWindow.Show();
            }
            else
            {
                MessageBox.Show("Password does not match!");
            }
        }

        private void exit_btn(object sender, RoutedEventArgs e)
        {

```

```

        MessageBoxResult messageBoxResult = System.Windows.MessageBox.Show("Are you
sure wanna cancel process ?", "Exit Confirmation", System.Windows.MessageBoxButton.YesNo,
MessageBoxImage.Warning);

        if (messageBoxResult == MessageBoxResult.Yes)
        {
            this.Close();
        }
        else
        {
            //
        }
    }

    private void pw_show_btn(object sender, RoutedEventArgs e)
    {
        password_box.PasswordChar = default(char);
    }
}

```

MainWindow.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.Navigation;
using System.Windows.Shapes;
using System.IO;

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

        public MainWindow()
        {
            InitializeComponent();
        }
        private void exit_btn(object sender, RoutedEventArgs e)
        {

```



```

        MessageBoxResult messageBoxResult = System.Windows.MessageBox.Show("Are you
sure wanna exit ?", "Exit Confirmation",
System.Windows.MessageBoxButton.YesNo, MessageBoxImage.Warning);

        if (messageBoxResult == MessageBoxResult.Yes)
        {
            this.Close();
            MainWindow mainWindow = new MainWindow();
        }
        else
        {
            //
        }

    }

    private void register_student(object sender, RoutedEventArgs e)
    {
        this.Hide();

        RegisterMainWindow rmWindow = new RegisterMainWindow();
        rmWindow.Show();
    }

    private void view_report(object sender, RoutedEventArgs e)
    {
        this.Hide();
        ReportMainWindow reportMainWindow = new ReportMainWindow();
        reportMainWindow.Show();
    }

    private void log_out_Click(object sender, RoutedEventArgs e)
    {
        MessageBoxResult messageBoxResult = System.Windows.MessageBox.Show("Are you
sure wanna logout ?", "Logout Confirmation",
System.Windows.MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (messageBoxResult == MessageBoxResult.Yes)
        {
            this.Hide();
            HomeWindow homeWindow = new HomeWindow();
            homeWindow.Show();
        }
        else
        {
            //
        }

    }

    private void change_password_Click(object sender, RoutedEventArgs e)
    {
        string msg = "Change your password";
        password pw = new password(msg);
        pw.Show();
    }

```

```

    }

}
}
//////////
///

```

RegisterStudent.xaml.cs

```

        using System;
using System.Collections.Generic;
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.Shapes;

namespace Student_info
{
    /// <summary>
    /// Interaction logic for Register_student.xaml
    /// </summary>
    public partial class Register_student : Window
    {
        string sex = "";
        public Register_student()
        {
            InitializeComponent();
        }

        private void cancel_btn(object sender, RoutedEventArgs e)
        {
            MessageBoxResult messageBoxResult = System.Windows.MessageBox.Show("Are you
sure wanna cancel process?", "Confirmation", System.Windows.MessageBoxButton.YesNo,
MessageBoxImage.Question);

            if (messageBoxResult == MessageBoxResult.Yes)
            {
                this.Close();
                RegisterMainWindow RmWindow = new RegisterMainWindow();
                RmWindow.Show();
            }
        }
    }
}

```

```

        else
        {
            //
        }
    }

    private void clear_btn(object sender, RoutedEventArgs e)
    {
        MessageBoxResult messageBoxResult = System.Windows.MessageBox.Show("Are you
sure wanna clear all fields?", "Reset Confirmation",
System.Windows.MessageBoxButton.YesNo, MessageBoxImage.Question);

        if (messageBoxResult == MessageBoxResult.Yes)
        {
            clear_text();
        }
        else
        {
            //
        }
    }

    private void departmentBox_SelectionChanged(object sender,
SelectionChangedEventArgs e)
    {
        facultyBox.Items.Clear();
        facultyBox.Items.Add("Click Load button");
        facultyBox.SelectedItem = "Click Load button";

        programBox.Items.Clear();
        programBox.Items.Add("Click Load button");
        programBox.SelectedItem = "Click Load button";
    }

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

    private void faculty_load_Click(object sender, RoutedEventArgs e)
    {
        facultyBox.Items.Clear();

        if (departmentBox.Text == "")
        {
            facultyBox.Items.Add("Select department first");
            facultyBox.SelectedItem = ("Select department first");
        }
    }

```

```
        else if (departmentBox.Text == "IT")
        {
            facultyBox.Items.Add("BIT");
            facultyBox.Items.Add("CSIT");
        }

        else if (departmentBox.Text == "Education")
        {
            facultyBox.Items.Add("Literature");
            facultyBox.Items.Add("Teaching");
        }

        else if (departmentBox.Text == "Management")
        {
            facultyBox.Items.Add("BBA");
            facultyBox.Items.Add("BBS");
        }
    }

    private void program_load_Click(object sender, RoutedEventArgs e)
    {
        dropdown_loader();
    }

    //validating student phone
    private void phone_TextChanged(object sender, TextChangedEventArgs e)
    {
        bool valid = int_validator(phone.Text);

        if (valid == false)
        {
            System.Windows.MessageBox.Show("Number can't be String", "Invalid",
            System.Windows.MessageBoxButton.OK, MessageBoxImage.Information);
            phone.Clear();
        }
    }

    //validating student id
    private void student_id_TextChanged(object sender, TextChangedEventArgs e)
    {
        bool valid = int_validator(student_id.Text);

        if (valid == false)
        {
            student_id.Clear();
            MessageBox.Show("ID can't be
            String", "Invalid", System.Windows.MessageBoxButton.OK, MessageBoxImage.Information);
        }
    }
```

```

    }

    // register new student
    private void register_btn(object sender, RoutedEventArgs e)
    {
        bool valid = validator();

        //validating email format
        Regex mRegexExpression;

        if (mail_id.Text.Trim() != string.Empty)
        {
            mRegexExpression = new Regex(@"^([a-zA-Z0-9_\-\.\.])*\.[a-zA-Z0-9_\-\.\.]{3}([a-zA-Z0-9_\-\.\.]{2,}|(25[0-5]|2[0-4][0-9]|1[0-9][0-9]|1[0-9][0-9]|1[0-9][0-9]|1[0-9][0-9]|1[0-9][0-9]|1[0-9][0-9])$");

            if (!mRegexExpression.IsMatch(mail_id.Text.Trim()))
            {
                System.Windows.MessageBox.Show("E-mail address format is not correct.", "incorrect format", System.Windows.MessageBoxButton.OK,
                MessageBoxImage.Warning);
                mail_id.Focus();
            }
            //register student only if email format is correct
            else
            {
                if (valid == true)
                {
                    passValue();
                }
            }
        }
    }

    //getting gender

    private void maleRdb_Checked(object sender, RoutedEventArgs e)
    {
        sex = "Male";
    }

    private void femaleRdb_Checked(object sender, RoutedEventArgs e)
    {
        sex = "Female";
    }

    private void otherRdb_Checked(object sender, RoutedEventArgs e)

```

```

    {
        sex = "Other";
    }
    //clearing inputed text in all text fields
    private void clear_text()
    {
        first_name.Clear();
        last_name.Clear();
        student_id.Clear();
        phone.Clear();
        mail_id.Clear();
        tmp_address.Clear();
        permanent_address.Clear();
        religion.Clear();
        nationality.Clear();

        departmentBox.Items.DeferRefresh();
    }

    //loading courses on dropdown list
    private void dropdown_loader()
    {
        programBox.Items.Clear();

        if (facultyBox.Text == "")
        {
            programBox.Items.Add("Select faculty first");
            programBox.SelectedItem = ("Select faculty first");
        }
        else if (facultyBox.Text == "Select department first" || facultyBox.Text ==
"Click Load button")
        {
            programBox.Items.Add("Select faculty first");
            programBox.SelectedItem = ("Select faculty first");
        }

        else if (facultyBox.Text == "BIT")
        {
            programBox.Items.Add("Computing");
            programBox.Items.Add("Multimedia Technologies ");
            programBox.Items.Add("Networks and IT Security ");
        }
        else if (facultyBox.Text == "CSIT")
        {
            programBox.Items.Add("csit course 1");
            programBox.Items.Add("csit course 2");
            programBox.Items.Add("csit course 3");
        }

        else if (facultyBox.Text == "Literature")
        {
            programBox.Items.Add("Literature course 1");
            programBox.Items.Add("Literature course 2");
            programBox.Items.Add("Literature course 3");
        }
    }

```

```
        else if (facultyBox.Text == "Teaching")
        {
            programBox.Items.Add("Teaching course 1");
            programBox.Items.Add("Teaching course 2");
            programBox.Items.Add("Teaching course 3");
        }

        else if (facultyBox.Text == "BBA")
        {
            programBox.Items.Add("BBA course 1");
            programBox.Items.Add("BBA course 2");
            programBox.Items.Add("BBA course 3");
        }

        else if (facultyBox.Text == "BBS")
        {
            programBox.Items.Add("BBS course 1");
            programBox.Items.Add("BBS course 2");
            programBox.Items.Add("BBS course 3");
        }
    }

    //validating all the empty required text box
    public bool validator()
    {
        //empty input validation
        if (first_name.Text == "")
        {
            MessageBox.Show("Firstname is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
            return false;
        }
        else if (last_name.Text == "")
        {
            MessageBox.Show("Lastname is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
            return false;
        }
        else if (student_id.Text == "")
        {
            MessageBox.Show("student_id is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
            return false;
        }
        else if (phone.Text == "")
        {
            MessageBox.Show("Phone is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
            return false;
        }
        else if (phone.Text.Length < 5)
        {
```

```
        MessageBox.Show("Contact number can't be too short", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
        return false;
    }

    else if (mail_id.Text == "")
    {
        MessageBox.Show("Mail-id is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
        return false;
    }

    else if (sex == "")
    {
        MessageBox.Show("Gender is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
        return false;
    }

    else if (tmp_address.Text == "" || permanent_address.Text == "")
    {
        MessageBox.Show("Address is required", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
        return false;
    }

    else if (departmentBox.Text == "")
    {
        MessageBox.Show("Select one Department ");
        return false;
    }

    else if (facultyBox.Text == "" || facultyBox.Text == "Click Load button" ||
facultyBox.Text == "Select department first")
    {
        MessageBox.Show("Select one Faculty", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
        return false;
    }

    else if (programBox.Text == "" || programBox.Text == "Click Load button" ||
programBox.Text == "Select faculty first")
    {
        MessageBox.Show("Select one Program", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Warning);
        return false;
    }

    else
    {
        return true;
    }

}

//validating whether a number is integer or not
```



```

public bool int_validator(string num)
{
    bool isnumber = Int32.TryParse(num, out int j);

    if (isnumber == false)
    {
        return false;
    }
    else
    {
        return true;
    }
}

public void passValue()
{
    string Fname = first_name.Text;
    string Lname = last_name.Text;
    int Id = Convert.ToInt32(student_id.Text);
    string Department = departmentBox.Text;
    string Faculty = facultyBox.Text;
    string Program = programBox.Text;
    int Phone = Convert.ToInt32(phone.Text);
    string Email = mail_id.Text;
    string Gender = sex;
    string currentAddr = tmp_address.Text;
    string permanentAddr = permanent_address.Text;
    string Religion = religion.Text;
    string Nationality = nationality.Text;
    string MarritalStatus = marritalStatus.Text;

    this.Hide();
    Register_confirm register_Confirm = new Register_confirm(Fname, Lname, Id,
Department, Faculty, Program, Phone, Email, Gender, currentAddr, permanentAddr, Religion,
Nationality, MarritalStatus);
    register_Confirm.Show();
}

private void first_name_TextChanged(object sender, TextChangedEventArgs e)
{
    bool valid = int_validator(first_name.Text);

    if (valid == true)
    {
        first_name.Clear();
        MessageBox.Show("Name can't be Number", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Information);
    }
}

private void last_name_TextChanged(object sender, TextChangedEventArgs e)
{

```

```
        bool valid = int_validator(last_name.Text);

        if (valid == true)
        {
            last_name.Clear();
            MessageBox.Show("Name can't be Number", "Invalid",
System.Windows.MessageBoxButton.OK, MessageBoxImage.Information);
        }
    }
}
```

RegisterBulk.xaml.cs

```
using FileHelpers;
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 Student_info
{
    /// <summary>
    /// Interaction logic for RegisterBulkStdWindow.xaml
    /// </summary>
    public partial class RegisterBulkStdWindow : Window
    {
        private string fileName;
        private List<StudentInfo> csvStudentList;
        public RegisterBulkStdWindow()
        {
            InitializeComponent();
        }

        public void Save(StudentInfo studentInfo)
        {
            if (!File.Exists(fileName))
            {
                var file = File.Create(fileName);
                file.Close();
            }

            using (StreamWriter streamWriter = new StreamWriter(fileName, true))
            {
                streamWriter.WriteLine(studentInfo.ToString());
                streamWriter.Close();
            }
        }

        public List<StudentInfo> ReadAll()
        {
            try {
                if (!File.Exists(fileName))
                {
                    throw new FileNotFoundException("Student Info file doesn't exist");
                }
            }
            catch (Exception ex)
            {
                MessageBox.Show("Sorry! unexpected Error occured! try again", "Error",
                MessageBoxButton.OK, MessageBoxImage.Error);
            }
            List<StudentInfo> students = new List<StudentInfo>();
            try

```

```

    {
        using (StreamReader streamReader = new StreamReader(fileName))
        {
            streamReader.ReadLine();

            while (streamReader.Peek() != -1)
            {
                var studentString = streamReader.ReadLine();
                var studentInfo = new StudentInfo(studentString);
                students.Add(studentInfo);
            }
            streamReader.Close();
        }
    }
    catch (Exception ex)
    {
        MessageBox.Show("Sorry! unexpected Error occurred! try again", "Error",
        MessageBoxButton.OK, MessageBoxImage.Error);
        this.Close();
        RegisterMainWindow registerMainWindow = new RegisterMainWindow();
        registerMainWindow.Show();
    }
    return students;
}

private void fileChooseBtn_Click(object sender, RoutedEventArgs e)
{
    OpenFileDialog theDialog = new OpenFileDialog();
    theDialog.Title = "Open csv File";
    theDialog.Filter = "CSV files|*.csv";
    theDialog.InitialDirectory = @"C:\\";
    theDialog.ShowDialog();

    fileName = theDialog.FileName;

    csvStudentList = ReadAll();
    gridView.ItemsSource = csvStudentList;
}

private void registerBtn_Click(object sender, RoutedEventArgs e)
{
    var handler = new Handler();
    var dataSet = handler.CreateDataSet();
    dataSet = new DataSet();
    if (File.Exists(@"D:\Student_Management\StudentRegistrationData.xml"))
    {
        dataSet.ReadXml(@"D:\Student_Management\StudentRegistrationData.xml");
        AddData(dataSet);
        dataSet.WriteXml(@"D:\Student_Management\StudentRegistrationData.xml");
    }
    else
    {

```

```

        handler = new Handler();
        dataSet = handler.CreateDataSet();
        AddData(dataSet);

dataSet.WriteXmlSchema(@"D:\Student_Management\StudentRegistrationSchema.xml");
dataSet.WriteXml(@"D:\Student_Management\StudentRegistrationData.xml");

    }
    MessageBox.Show("Successfully Registered");
    this.Close();
    RegisterMainWindow registerMainWindow = new RegisterMainWindow();
    registerMainWindow.Show();
}
private void AddData(DataSet dataSet)
{
    var studentData = ReadAll();
    foreach (StudentInfo stdData in studentData)
    {
        var dr = dataSet.Tables["Student"].NewRow();
        dr["Name"] = stdData.Name;
        dr["StudentID"] = Convert.ToInt32(stdData.StudentID);
        dr["Department"] = stdData.Department;
        dr["Faculty"] = stdData.Faculty;
        dr["Program"] = stdData.Program;
        dr["Phone"] = Convert.ToInt32(stdData.Phone);
        dr["Email"] = stdData.Email;
        dr["Gender"] = stdData.Gender;
        dr["CurrentAddress"] = stdData.CurrentAddress;
        dr["PermanentAddress"] = stdData.PermanentAddress;
        dr["Religion"] = stdData.Religion;
        dr["Nationality"] = stdData.Nationality;
        dr["MaritalStatus"] = stdData.MarritalStatus;
        dr["RegistrationDate"] = stdData.RegistrationDate;
        dataSet.Tables["Student"].Rows.Add(dr);
    }
}

private void cancelBtn_Click(object sender, RoutedEventArgs e)
{
    MessageBoxResult messageBoxResult = System.Windows.MessageBox.Show("Are you
sure wanna cancel ?", "Cancel Confirmation", System.Windows.MessageBoxButton.YesNo,
MessageBoxImage.Question);

    if (messageBoxResult == MessageBoxResult.Yes)
    {
        this.Hide();

        RegisterMainWindow RmWindow = new RegisterMainWindow();
        RmWindow.Show();
    }
    else
    {
        //

```

```

    }

    }

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

public class StudentInfo
{
    public StudentInfo(string studentString)
    {
        this.ConvertToObject(studentString);
    }
    public string Name { get; set; }
    public string StudentID { get; set; }
    public string Department { get; set; }
    public string Faculty { get; set; }
    public string Program { get; set; }
    public string Phone { get; set; }
    public string Email { get; set; }
    public string Gender { get; set; }
    public string CurrentAddress { get; set; }
    public string PermanentAddress { get; set; }
    public string Religion { get; set; }
    public string Nationility { get; set; }
    public string MarritalStatus { get; set; }
    public string RegistrationDate { get; set; }

    public override string ToString()
    {
        return
        $"{this.Name}:{this.StudentID}:{this.Department}:{this.Faculty}:{this.Program}:{this.Phon
e}:{this.Email}:{this.Gender}:{this.CurrentAddress}:{this.PermanentAddress}:{this.Religio
n}:{this.Nationility}:{this.RegistrationDate}";
    }

    private void ConvertToObject(string studentString)
    {
        var splitedStrings = studentString.Split(',');
        this.Name = splitedStrings[0];
        this.StudentID = splitedStrings[1];
        this.Department = splitedStrings[2];
        this.Faculty = splitedStrings[3];
        this.Program = splitedStrings[4];
        this.Phone = splitedStrings[5];
        this.Email = splitedStrings[6];
        this.Gender = splitedStrings[7];
        this.CurrentAddress = splitedStrings[8];
        this.PermanentAddress = splitedStrings[9];
        this.Religion= splitedStrings[10];
    }
}

```

```
        this.Nationility = splitedStrings[11];  
        this.MarritalStatus = splitedStrings[12];  
        this.RegistrationDate = splitedStrings[13];  
    }  
}  
}
```

Handler.cs

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

```
namespace Student_info
{
    public class Handler
    {
        public DataSet CreateDataSet()
        {
            var ds = new DataSet();
            ds.Tables.Add(CreateStudentTable());
            return ds;
        }

        private DataTable CreateStudentTable()
        {
            var dt = new DataTable("Student");
            //DataColumn dataColumn = new DataColumn("SN", typeof(int));
            //dataColumn.AutoIncrement = true;
            //dataColumn.AutoIncrementSeed = 1;
            //dataColumn.AutoIncrementStep = 1;

            //dt.Columns.Add(dataColumn);
            dt.Columns.Add("Name", typeof(string));
            dt.Columns.Add("StudentID", typeof(int));
            dt.Columns.Add("Department", typeof(string));
            dt.Columns.Add("Faculty", typeof(string));
            dt.Columns.Add("Program", typeof(string));
            dt.Columns.Add("Phone", typeof(int));
            dt.Columns.Add("Email", typeof(string));
            dt.Columns.Add("Gender", typeof(string));
            dt.Columns.Add("CurrentAddress", typeof(string));
            dt.Columns.Add("PermanentAddress", typeof(string));
            dt.Columns.Add("Religion", typeof(string));
            dt.Columns.Add("Nationality", typeof(string));
            dt.Columns.Add("MaritalStatus", typeof(string));
            dt.Columns.Add("RegistrationDate", typeof(string));
            return dt;
        }

        private void AddSampleData(DataSet dataSet)
        {
            var dr = dataSet.Tables["Student"].NewRow();
            dr["Name"] = "Ashish Bhandari";
            dr["StudentID"] = 17031918;
            dr["Depaertment"] = "IT";
            dr["Faculty"] = "BIT";
            dr["Program"] = "Networking";
            dr["Phone"] = 986911390;
            dr["Email"] = "ashish@gmail.com";
            dr["Gender"] = "Male";
            dr["CurrentAddress"] = "Ratnachowk-07";
            dr["PermanentAddress"] = "Kalika-28";
            dr["Reiligion"] = "Hindu";
            dr["Nationality"] = "Nepali";
            dr["MarritalStatus"] = "Unmarried";
            dr["RegistrationDate"] = DateTime.Today;
            dataSet.Tables["Student"].Rows.Add(dr);
        }
    }
}
```



```
    }  
  }  
}
```

ViewModel.cs

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

```

namespace Student_info
{
    class ViewModel
    {
        string path = @"D:\Student_Management\StudentRegistrationData.xml";
        int ITData;
        int mgtData;
        int eduData;

        int ITData2;
        int mgtData2;
        int eduData2;
        public List<Sales> Data { get; set; }
        public ViewModel()
        {

            loadData();
            Data = new List<Sales>()
            {
                new Sales { Week=$"{DateTime.Now.AddDays(-
7).ToString("dd/MM/yyyy")}", ITA= ITData ,EducationB=eduData, ManagementC= mgtData},
                new Sales { Week=$"{DateTime.Now.AddDays(-
14).ToString("dd/MM/yyyy")}", ITA= ITData2 ,EducationB=eduData2, ManagementC= mgtData2},
            };
        }
        private void loadData()
        {
            var handler = new Handler();

            var dataSet = handler.CreateDataSet();

            if (System.IO.File.Exists(path))
            {

                dataSet.ReadXml(path);

                DataTable stdReportTbl = dataSet.Tables["Student"];
                DataTable dv = stdReportTbl.Select("").CopyToDataTable();
                //filtering date of one week

                //counting total number of student registered in a week
                ITData = stdReportTbl.Select("Department = 'IT' AND RegistrationDate>="
+ DateTime.Today.AddDays(-7) + "").Count<DataRow>();
                mgtData = stdReportTbl.Select("Department = 'Management' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
                eduData = stdReportTbl.Select("Department = 'Education' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();

                ITData2 = stdReportTbl.Select("Department = 'IT' AND RegistrationDate>="
+ DateTime.Today.AddDays(-14) + "").Count<DataRow>();
                mgtData2 = stdReportTbl.Select("Department = 'Management' AND
RegistrationDate>=" + DateTime.Today.AddDays(-14) + "").Count<DataRow>();
                eduData2 = stdReportTbl.Select("Department = 'Education' AND
RegistrationDate>=" + DateTime.Today.AddDays(-14) + "").Count<DataRow>();
            }
        }
    }
}

```

```
        }  
    }  
}
```

chartWindow.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 Student_info
{
    /// <summary>
    /// Interaction logic for ChartWindow.xaml
    /// </summary>
    public partial class ChartWindow : Window
    {
        string path = @"D:\Student_Management\StudentRegistrationData.xml";
        int ITData;
        int mgtData;
        int eduData;

        public List<Sales> Data { get; set; }
        public ChartWindow()
        {
            InitializeComponent();

            LoadColumnChartData();
        }
        private void LoadColumnChartData()
        {
            loadData();

            ((ColumnSeries)mcChart.Series[0]).ItemsSource =
                new KeyValuePair<string, int>[] {
                    new KeyValuePair<string, int>("IT", ITData),
                    new KeyValuePair<string, int>("Management", mgtData),
                    new KeyValuePair<string, int>("Education", eduData)
                };
        }

        private void loadData()
        {
            var handler = new Handler();

            var dataSet = handler.CreateDataSet();

            try
            {
                if (System.IO.File.Exists(path))
                {
                    dataSet.ReadXml(path);

                    DataTable stdReportTbl = dataSet.Tables["Student"];
                }
            }
        }
    }
}

```

```

        DataTable dv = stdReportTbl.Select("").CopyToDataTable();
        //filtering date of one week
        //filtering date of one week
        //counting total number of student registered in a week
        //counting total number of student registered in a week
        ITData = stdReportTbl.Select("Department = 'IT' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
        mgtData = stdReportTbl.Select("Department = 'Management' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
        eduData = stdReportTbl.Select("Department = 'Education' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();

    }
    else
    {
        System.Windows.MessageBox.Show("Sorry! XML file is missing", "File
Not Found", MessageBoxButton.OK, MessageBoxImage.Error);
    }
}
catch (Exception ex)
{
    System.Windows.MessageBox.Show("Sorry! Error occured", "Error",
MessageBoxButton.OK, MessageBoxImage.Error);
    this.Close();
    ReportMainWindow rpw = new ReportMainWindow();
    rpw.Show();
}
}

private void back_btn(object sender, RoutedEventArgs e)
{
    this.Hide();
    ReportMainWindow rmw = new ReportMainWindow();
    rmw.Show();
}
}
}

```

ViewWeeklyReport.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 Student_info
{
    /// <summary>
    /// Interaction logic for WeeklyReportWindow.xaml
    /// </summary>
    public partial class WeeklyReportWindow : Window
    {
        string path = @"D:\Student_Management\StudentRegistrationData.xml";
        int ITData;
        int mgtData;
        int eduData;
        int BITData;
        int CSITData;
        int literatureData;
        int teachingData;
        int bbaData;
        int bbsData;
        public WeeklyReportWindow()
        {
            InitializeComponent();
            loadData();
        }
        private void loadData()
        {
            var handler = new Handler();

            var dataSet = handler.CreateDataSet();

            try
            {
                if (System.IO.File.Exists(path))
                {
                    dataSet.ReadXml(path);

                    DataTable stdReportTbl = dataSet.Tables["Student"];
                    DataTable dv = stdReportTbl.Select("").CopyToDataTable();
                    //filtering date of one week

                    //counting total number of student registered in a week
                    ITData = stdReportTbl.Select("Department = 'IT' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
                    mgtData = stdReportTbl.Select("Department = 'Management' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
                    eduData = stdReportTbl.Select("Department = 'Education' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
                    //faculty
                    BITData = stdReportTbl.Select("Faculty = 'BIT' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();

```

```

        CSITData = stdReportTbl.Select("Faculty = 'CSIT' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
        literatureData = stdReportTbl.Select("Faculty = 'Literature' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
        teachingData = stdReportTbl.Select("Faculty = 'Teaching' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
        bbaData = stdReportTbl.Select("Faculty = 'BBA' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();
        bbsData = stdReportTbl.Select("Faculty = 'BBS' AND
RegistrationDate>=" + DateTime.Today.AddDays(-7) + "").Count<DataRow>();

        //assigning value to label
        itDepartLbl.Content = ITData;
        mgtDepartLbl.Content = mgtData;
        eduDepartLbl.Content = eduData;

        //faculty
        bitLbl.Content = BITData;
        CSITLbl.Content = CSITData;
        literatureLbl.Content = literatureData;
        teachingLbl.Content = teachingData;
        bbaLbl.Content = bbaData;
        bbsLbl.Content = bbsData;

    }
    else
    {
        System.Windows.MessageBox.Show("Sorry! XML file is missing", "File
Not Found", MessageBoxButton.OK, MessageBoxImage.Error);
    }
} catch (Exception ex)
{
    System.Windows.MessageBox.Show("Sorry! Error occured", "Error",
MessageBoxButton.OK, MessageBoxImage.Error);
    this.Close();
    ReportMainWindow rpw = new ReportMainWindow();
    rpw.Show();
}

}

private void back_btn(object sender, RoutedEventArgs e)
{
    this.Close();
    ReportMainWindow rpw = new ReportMainWindow();
    rpw.Show();
}
}
}

```