

# Informatics College Pokhara



informatics  
college pokhara

**Application Development**

**CS6004NI**

**Course Work 1**

**Submitted By:** Suman Bhandari  
**London Met ID:** Enter ID Here

**Submitted To:** Ishwor Sapkota  
Module Leader

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

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

### C. Programming Style

<b>Clarity of code,Popper Naming convention &amp; comments</b>	Very poor code
<b>System Usability</b>	very poorly developed application

<b>Overall Grade:</b>	<b>C</b>	<b>C</b>
-----------------------	----------	----------

### Overall Comment:

Code should be self explainable with less comments. Need some proper naming of the component 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

# Informatics College Pokhara



## Application Development

### CS6004NP

#### Coursework 1

**Submitted By:**

Student Name: Suman Bhandari  
London Met ID: 17030755  
Group: L1C3  
Date: 10-Jan-2020

**Submitted To:**

Mr. Ishwor Sapkota  
Application Development

## **Abstract**

This is the documentation for the individual coursework of module Application Development. Task assigned in the coursework is to develop a Student Registration system using Visual Studio Platform. The program is developed in C# programming language using WPF(Windows Presentation Foundation). The program will help educational institute to get rid from daily paperwork's and shift to digital platform .

## Table of Contents

Introduction.....	1
Current Scenario.....	1
Proposed System .....	1
User Manual .....	2
Login Screen .....	2
Home page .....	3
Add course.....	4
Enrol Student.....	6
Bulk Import.....	7
View Report .....	9
System Architecture .....	12
Functionality .....	12
Class Diagram .....	13
Individual Diagram.....	14
Login individual diagram .....	14
Home Page Individual diagram.....	14
Add Student Individual Diagram .....	15
View Report individual diagram .....	16
Add Course individual diagram .....	17
Import Excel individual diagram.....	17
Datahandler individual diagram .....	18
Student Info individual diagram .....	18
Flow chart.....	19
Test Cases .....	20
Test case 1 .....	20
Test case 2 .....	20
Test case 3 .....	21
Test case 4 .....	22
Test case 5 .....	22
Test case 6 .....	23
Test case 7 .....	24
Test case 8 .....	24
Sorting Algorithm .....	26
Related Research.....	28
WPF in Action with Visual Studio 2008( (Arlen Feldman, Maxx Daymon, 2008)).....	28

Microsoft .Net( <a href="https://docs.microsoft.com">https://docs.microsoft.com</a> ).....	28
WPF Tutorial( <a href="http://www.wpf-tutorial.com">www.wpf-tutorial.com</a> ).....	28
SearchCIO ( <a href="http://searchcio.techtarget.com">searchcio.techtarget.com</a> ) .....	28
techopedia ( <a href="http://www.techopedia.com">www.techopedia.com</a> ) .....	28
Reflection.....	29
Conclusion .....	30
References.....	31
Appendix.....	32

## Table of Figure

Figure 1:Login screen .....	2
Figure 2:Login screen alert message when username and password is blank	2
Figure 3:Login screen alert message when username and password are incorrect.....	3
Figure 4:Home Page Design.....	3
Figure 5:Form to add course.....	4
Figure 6:Alert message dialog when field ae empty.....	5
Figure 7:Dialog showing successfully course added.....	5
Figure 8:Enroll student or student registration form .....	6
Figure 9:Auto generated studentId.....	6
Figure 10:Message dialog showing sucessfull registration of student.....	7
Figure 11:Bulk Import or import from excel form .....	7
Figure 12:File chooser opened when browse file button is clicked .....	8
Figure 13:Data from Excel shown in table format .....	8
Figure 14:Message Dialog showing successful import of data from excel .....	9
Figure 15:View report page.....	9
Figure 16:Data sorted in ascending order by name .....	10
Figure 17:Data sorted in descending order by registration date .....	10
Figure 18:graphical representation of data .....	11
Figure 19:weekly report in table format.....	11
Figure 20:system architecture for student registration system .....	12
Figure 21:Class diagram for student registration system .....	13
Figure 22:Flowchart for student registration.....	19
Figure 23:test case 1: open file chooser .....	20
Figure 24:test case 2: import excel .....	21
Figure 25:test case 3: add course.....	21
Figure 26:test case 4:email validation.....	22
Figure 27:test case 5: phone no validation .....	23
Figure 28:test case 6:auto generate student id.....	23
Figure 29:test case 7 unique student id .....	24
Figure 30:test case 8: delete student data .....	25



## Table of table

Table 1:login individual diagram.....	14
Table 2:Home page individual diagram.....	14
Table 3:Add student individual diagram .....	15
Table 4:view report page individual diagram.....	16
Table 5:add course individual diagram .....	17
Table 6:import excel individual diagram.....	17
Table 7:datahandler individual diagram .....	18
Table 8 student info individual diagram.....	18
Table 9:test case 1 :open file chooser .....	20
Table 10:test case 2: import excel .....	20
Table 11:test case 3: add course .....	21
Table 12:test case 4:email validation .....	22
Table 13:test case 5: phone no validation.....	22
Table 14:test case 6:auto generate student id .....	23
Table 15:test case 7 unique student id .....	24
Table 16:test case 8: delete student data .....	24

## **Introduction**

Today, the world is heading toward digital system and almost all institute are shifting to digital system. Student Registration System is a desktop-based application designed primarily for educational institute. Unlike the traditional way of deploying the manpower to and different registers to keep record of students and teachers in hardcopy, this system simplifies the working efficiency of the institute in more efficient way. This system will allow any educational institute in a real time environment to increase the scope of business by reducing the labour cost and other paperwork cost.

Here I have developed a GUI based Student Registration system with the sequential aid of C# components, and finally to the detailed description of reports with the aid of different resources and tons of research. The system has functionality of login by employee and adding students and courses filling the registration form present in the system. Also, employee can register student by importing the data from Excel in csv format. Similarly, employee can view daily and weekly report also graphical report.

## **Current Scenario**

Whenever, we go for registration in any educational institute we have to fill printed form and our information is managed in register. In this method more staff, more time and manual effort is required. Moreover, there is difficulties in searching data and chance of unusual modification by other people. Many educational institutes today also keep record of student data and other data in this traditional way which is totally paper-based. In addition to this there are some institute who uses digital system but lacks some important features.

## **Proposed System**

The system I have developed is digitized system that can be used by educational institutes to enrol student, keep record of courses and view graphical as well as daily and weekly report. This system is the best alternative for those institute which are using traditional way of recording data. Also security is also ensured as it has login section. The GUI is present in simple and user friendly manner.

## User Manual

The process of operating the system are described below with screenshot and proper explanation.

### Login Screen

For the security purpose so that other unknown user modifies or delete any data the system consists of login screen. As end user operates the system the initial screen after program open is security screen. User have to provide valid username and password inorder to login successfully. The username and password of the system is both "admin". After successful login user is redirected to home page.

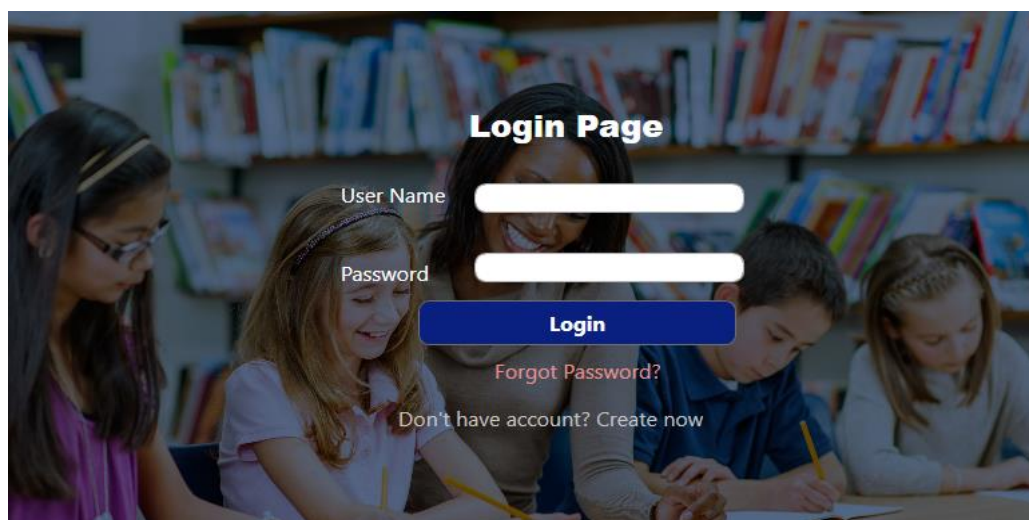


Figure 1:Login screen

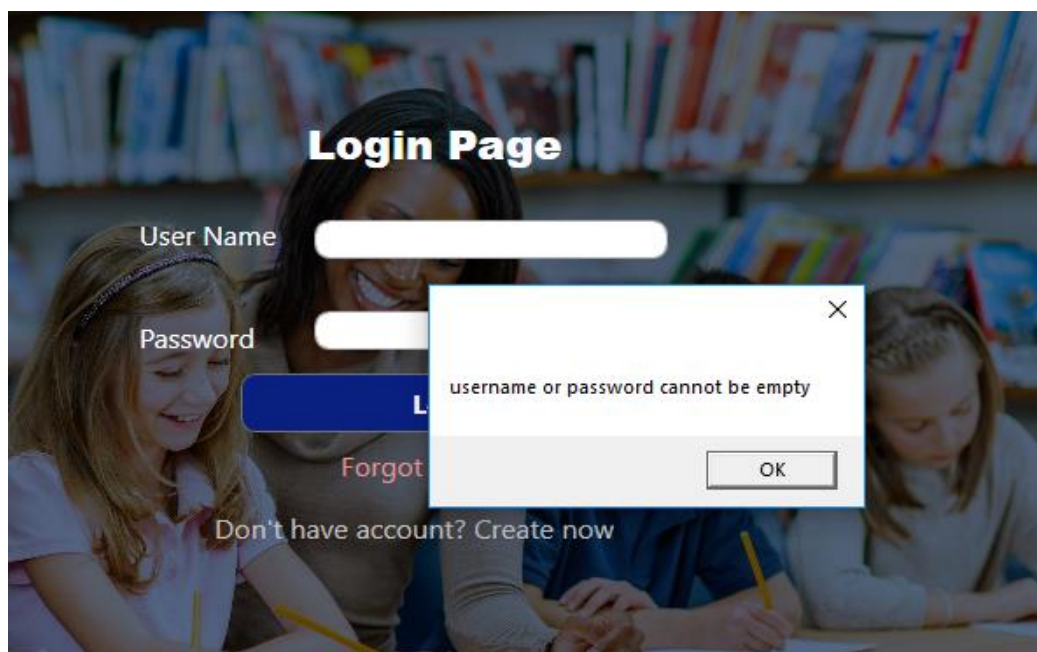


Figure 2:Login screen alert message when username and password is blank

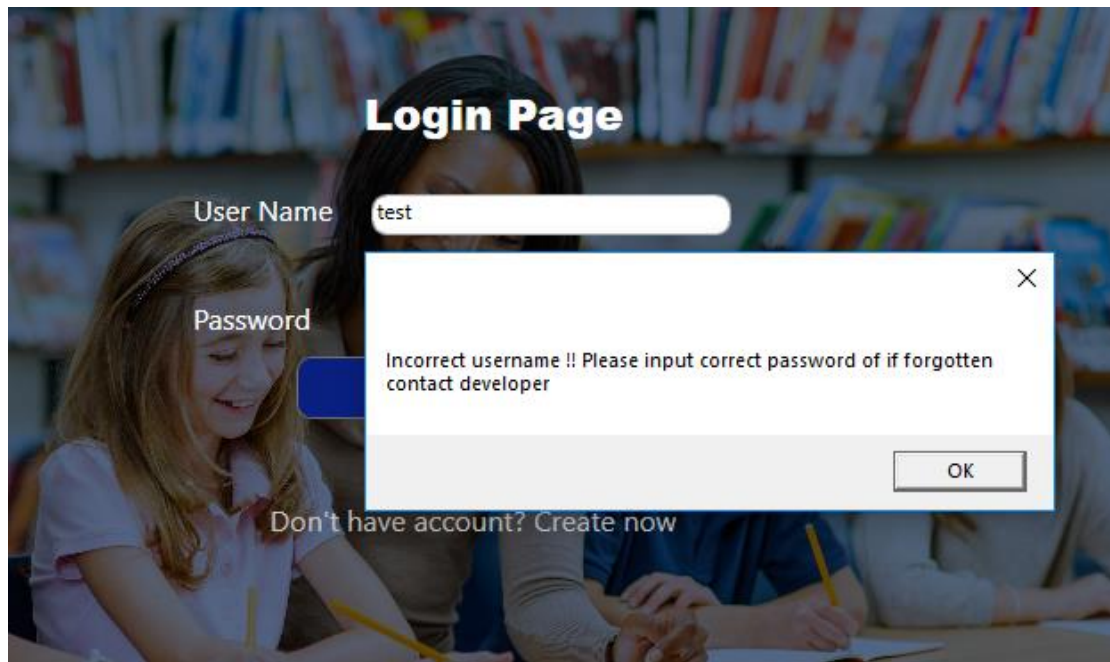


Figure 3:Login screen alert message when username and password are incorrect

## Home page

After successful login home page or main screen of the system appears. Here user can choose either to add course or enrol student or bulk import student data from excel or to view report.

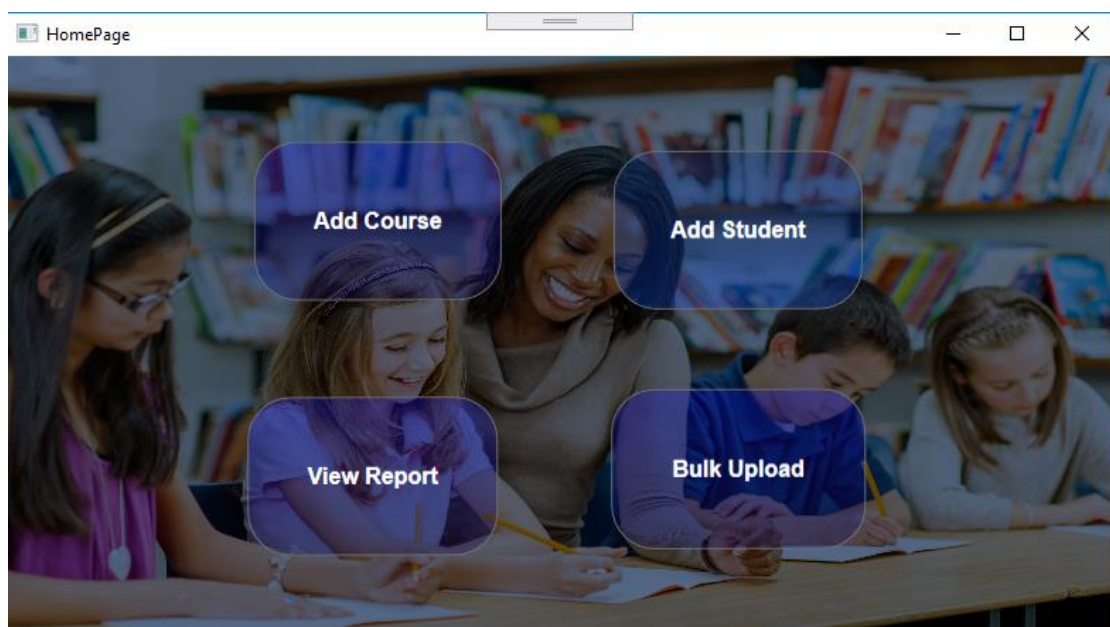


Figure 4:Home Page Design

## Add course

After user click on “Add Course” from home page user is redirected to new window where user have to fill the form inorder to add new course. The form consists of course name , course fee and course credit hours. User cannot leave any field blank and course fee and credit hours have to be input in number format only. The course added from here will be displayed in student registration form.

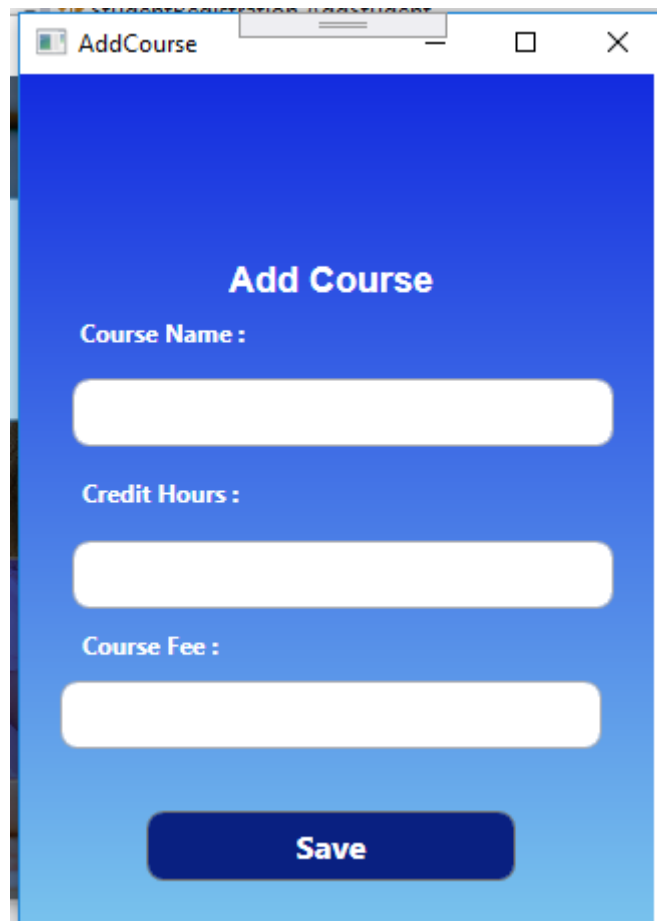
The image shows a web browser window titled "AddCourse". The main content area has a blue background. At the top, the text "Add Course" is displayed in white. Below this, there are three input fields, each with a label above it: "Course Name :", "Credit Hours :", and "Course Fee :". Each label is in white text. The input fields are white with rounded corners. At the bottom of the form, there is a dark blue button with the word "Save" in white text.

Figure 5:Form to add course

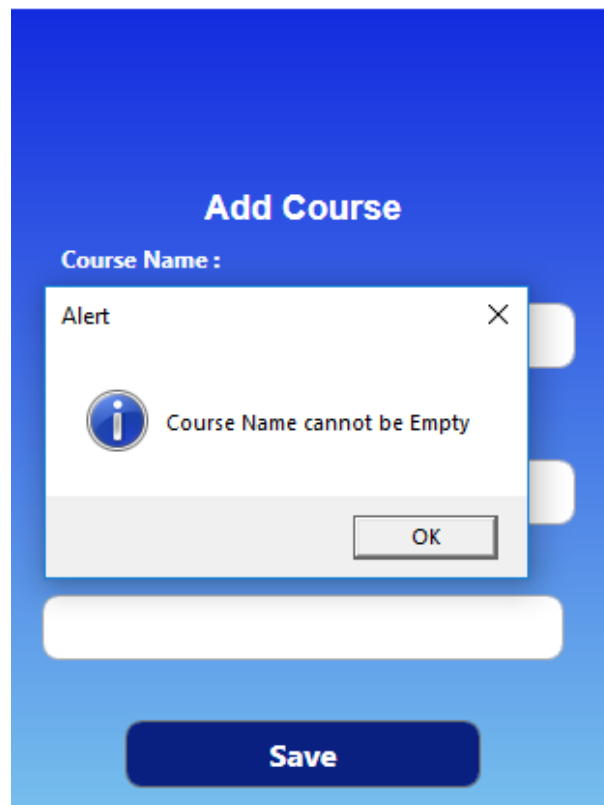


Figure 6:Alert message dialog when field ae empty

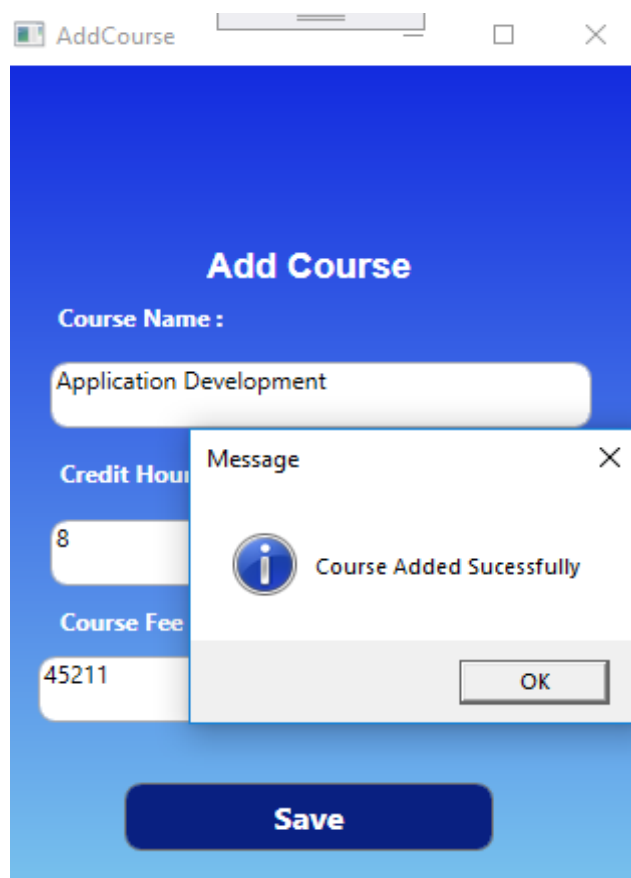


Figure 7:Dialog showing successfully course added

## Enrol Student

After user click on “Enrol Student” from home page user is redirected to student registration form. The form consists of three section. The first section consists of form which employee have to fill to register student. The second section consist of table which display old student information and newly added student information. The third section consists of buttons from where user can modify and delete student’s data. In the first section user have to provide the students information like students name, email, phone number, course enrol, address, parents number etc. Students id is given by system but if user want to change it he/she can do so. User cannot provide same student id that is used previously. User cannot leave any field blank for successful registration of student.

The screenshot shows a web application window titled "AddStudent". The form is divided into two columns of input fields. The first column contains: Student ID (with value 17144161), First Name, Zone (dropdown), Tole, Course Enroll (dropdown), and Email. The second column contains: Last Name, District (dropdown), Gender (radio buttons for Male, Female, Other), Contact No, Guardian No, and Your Registration Id (with value 17144161). To the right of the form are four buttons: Add (green), Clear (yellow), Delete (orange), and Exit (red). Below the form is a table with the following headers: stdId, firstName, lastname, gender, ContactNo, email, CourseEnroll, zone, District, Tole, guardianNo, RegistrationDate. The table body is currently empty.

Figure 8:Enroll student or student registration form

This screenshot is a close-up of the registration form. The "Student ID" field is highlighted with a yellow background and contains the value "17144161". The "Your Registration Id" field at the bottom right is also highlighted in yellow and contains the value "17144161". Other fields like First Name, Zone, Tole, Course Enroll, Email, Last Name, District, Gender, Contact No, and Guardian No are visible but not highlighted.

Figure 9:Auto generated studentId

The 'AddStudent' window contains the following fields and controls:

- Student ID : 17144161
- Last Name : Bhandari
- First Name : Suman
- District : kaski
- Zone : gandaki
- Gender : ☒ Male ☐ Female ☐ Other
- Tole : malepatan
- Contact No : 9827120831
- Course Enroll : Application Developer
- Email : shumanbhandari20@gmail.com
- Guardian No : 9824623563
- Registration Date : 1/9/2020 12:00:00 AM

Buttons: Add (green), Clear (yellow), Delete (orange), Exit (red).

**Message Dialog:**

Enroled Sucessfully

OK

stdId	firstname	lastname	gender	ContactNo	email	CourseEnroll	zone	District	Tole	guardianNo	RegistrationDate
17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Developer	gandaki	kaski	malepatan	9824623563	1/9/2020 12:00:00 AM

Figure 10: Message dialog showing sucessfull registration of student

## Bulk Import

After user click on “Bulk Import” from home page user is redirected to new window. In this page user have to click on browser file button and chose excel file that is in csv format and have students data in pre-defined order. The data from chosen file is displayed in table. If user find any data mistake, he/she have to edit data in excel file and again upload file. At last if all the information is correct user have to click on save button to save students data.

The 'ExcellImport' window features a large text input field at the top, a 'Browse Excel' button, and a 'Save' button at the bottom. Below the input field is a large, empty rectangular area for displaying data.

Figure 11: Bulk Import or import from excel form



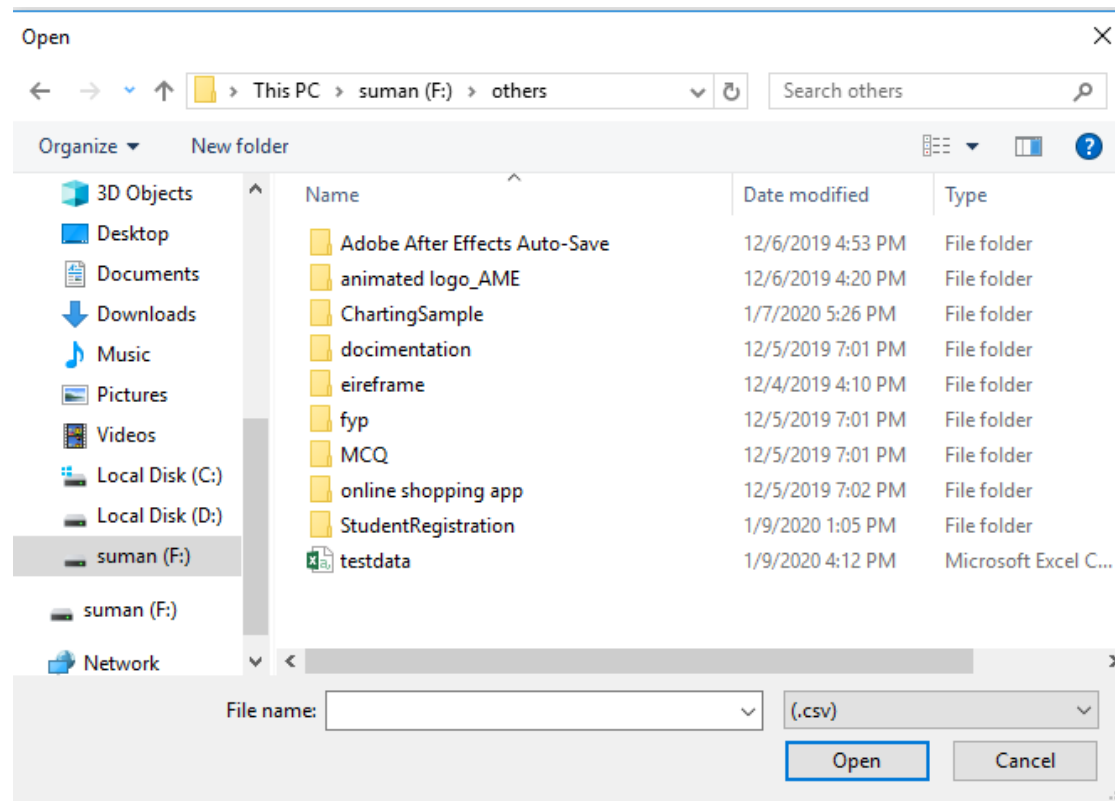


Figure 12:File chooser opened when browse file button is clicked

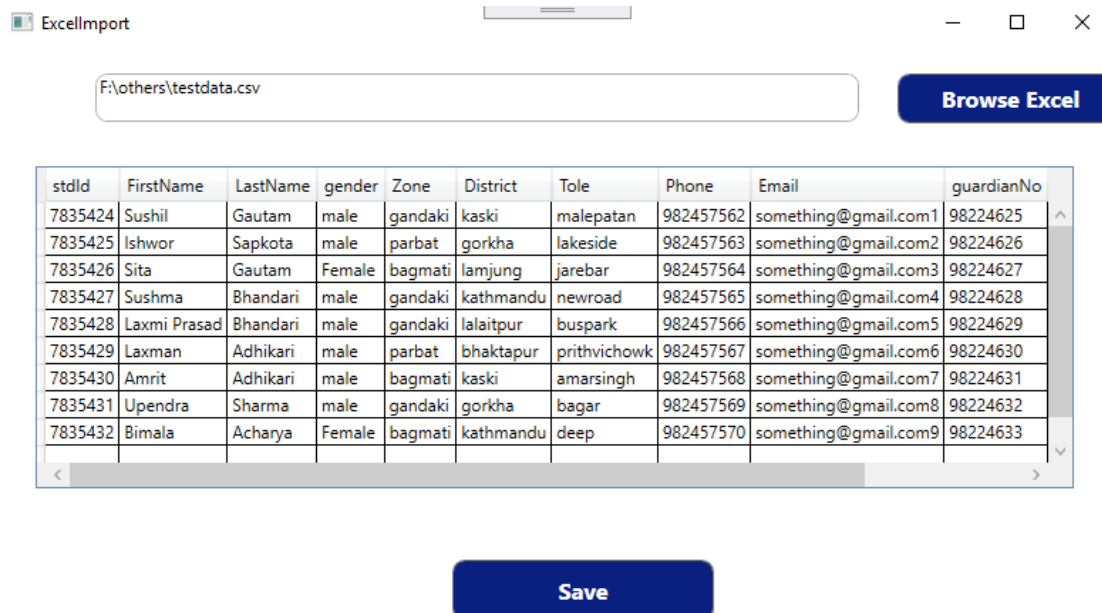


Figure 13:Data from Excel shown in table format

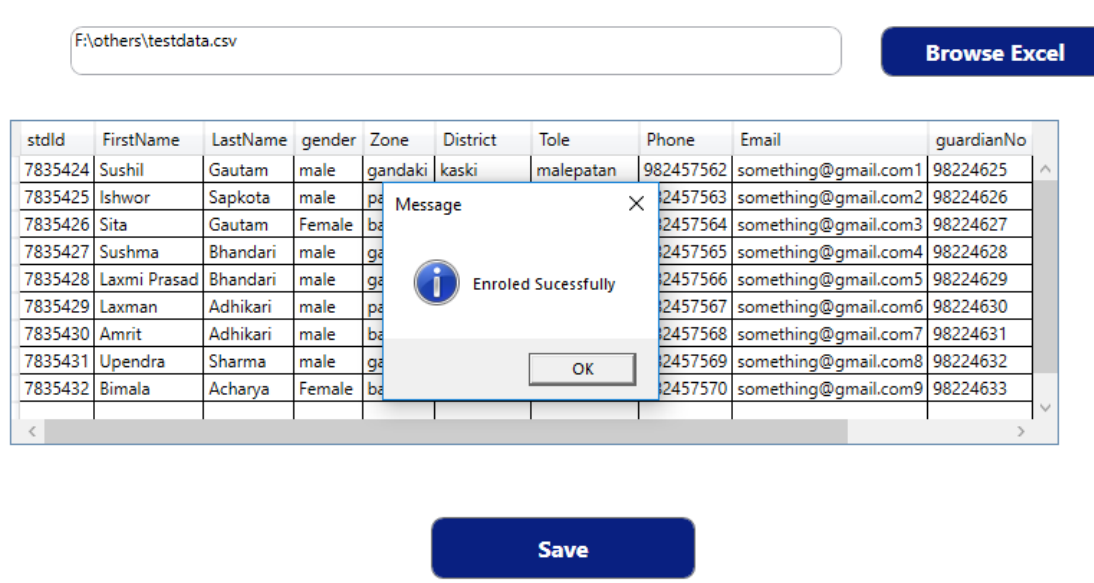


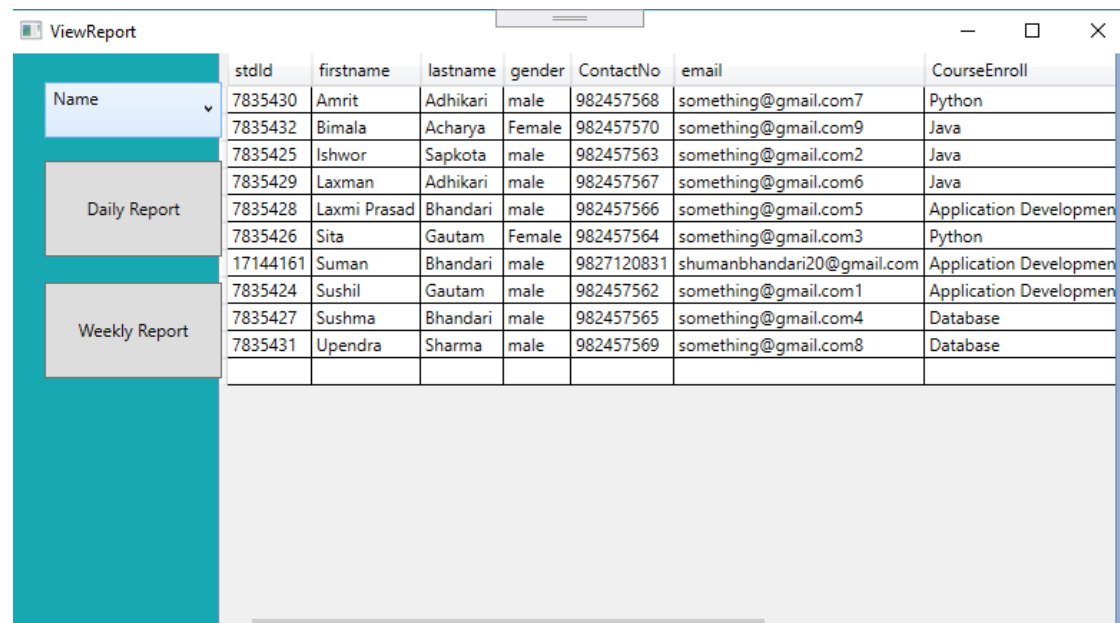
Figure 14: Message Dialog showing successful import of data from excel

## View Report

User is redirected to report section after user click on report button from home page. This page consists of three section. On first section all registered students data is displayed in table and user can order them by students first name and date of registration. On section user can view daily and weekly wise report of student registered on courses. On third section user can view graphical view of student number and course registered till the date.

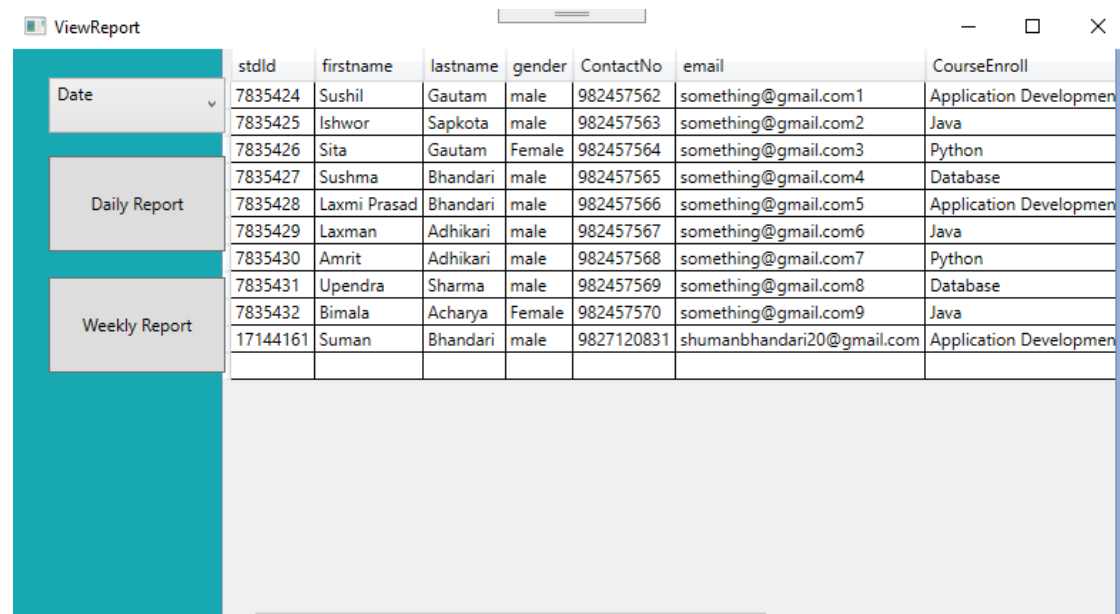
ViewReport		stdId	firstname	lastname	gender	ContactNo	email	CourseEnroll
<div>▼</div> <div>Daily Report</div> <div>Weekly Report</div>		7835424	Sushil	Gautam	male	982457562	something@gmail.com1	Application Developmen
		7835425	Ishwor	Sapkota	male	982457563	something@gmail.com2	Java
		7835426	Sita	Gautam	Female	982457564	something@gmail.com3	Python
		7835427	Sushma	Bhandari	male	982457565	something@gmail.com4	Database
		7835428	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Developmen
		7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java
		7835430	Amrit	Adhikari	male	982457568	something@gmail.com7	Python
		7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database
		7835432	Bimala	Acharya	Female	982457570	something@gmail.com9	Java
		17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Developmen

Figure 15: View report page



stdId	firstname	lastname	gender	ContactNo	email	CourseEnroll
7835430	Amrit	Adhikari	male	982457568	something@gmail.com7	Python
7835432	Bimala	Acharya	Female	982457570	something@gmail.com9	Java
7835425	Ishwor	Sapkota	male	982457563	something@gmail.com2	Java
7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java
7835428	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Development
7835426	Sita	Gautam	Female	982457564	something@gmail.com3	Python
17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Development
7835424	Sushil	Gautam	male	982457562	something@gmail.com1	Application Development
7835427	Sushma	Bhandari	male	982457565	something@gmail.com4	Database
7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database

Figure 16: Data sorted in ascending order by name



stdId	firstname	lastname	gender	ContactNo	email	CourseEnroll
7835424	Sushil	Gautam	male	982457562	something@gmail.com1	Application Development
7835425	Ishwor	Sapkota	male	982457563	something@gmail.com2	Java
7835426	Sita	Gautam	Female	982457564	something@gmail.com3	Python
7835427	Sushma	Bhandari	male	982457565	something@gmail.com4	Database
7835428	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Development
7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java
7835430	Amrit	Adhikari	male	982457568	something@gmail.com7	Python
7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database
7835432	Bimala	Acharya	Female	982457570	something@gmail.com9	Java
17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Development

Figure 17: Data sorted in descending order by registration date

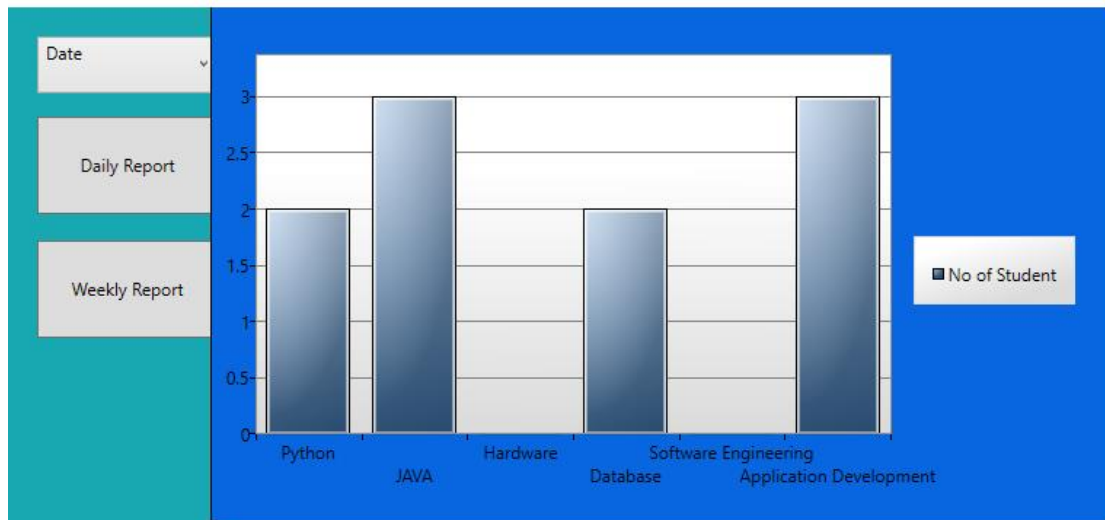


Figure 18:graphical representation of data

A table titled 'No of Student' showing the number of students for various courses. The table has two columns: Course and No of Student. The rows show the following data:

Course	No of Student
JAVA	3
Python	2
Hardware	0
Database	2
Software Engineering	0
Application Development	3

Figure 19:weekly report in table format

## System Architecture

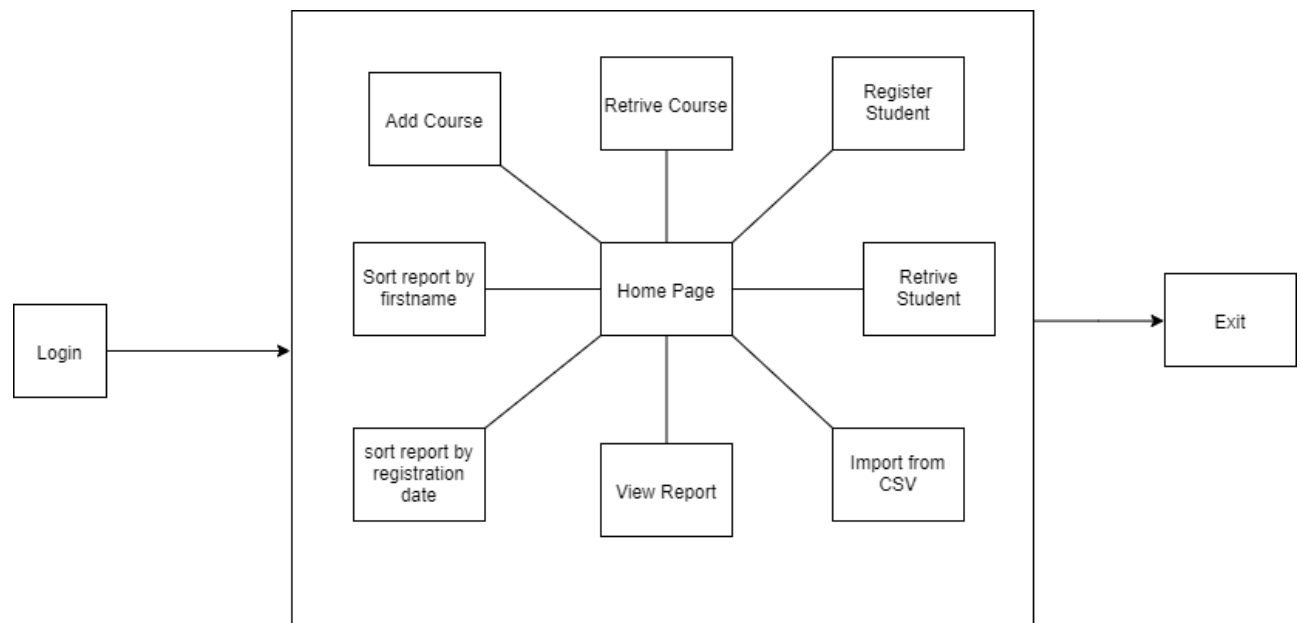


Figure 20:system architecture for student registration system

The figure above is the system architecture of student registration system. The user first login to the system providing valid username and password. After login the user is redirected to main window or home page. from home page user perform other task such as adding new course, enrolling student, viewing report, importing student data from csv etc. After performing the desired task user exit the system.

## Functionality

This is windows based application so to run this program a laptop or computer with windows operating system is required and the computer system must contain 2 GB of RAM in order to run this application. Students data and course data are saved in xml format and user can view them and edit them as well as user can view graphical as well as daily and weekly report.

## Class Diagram

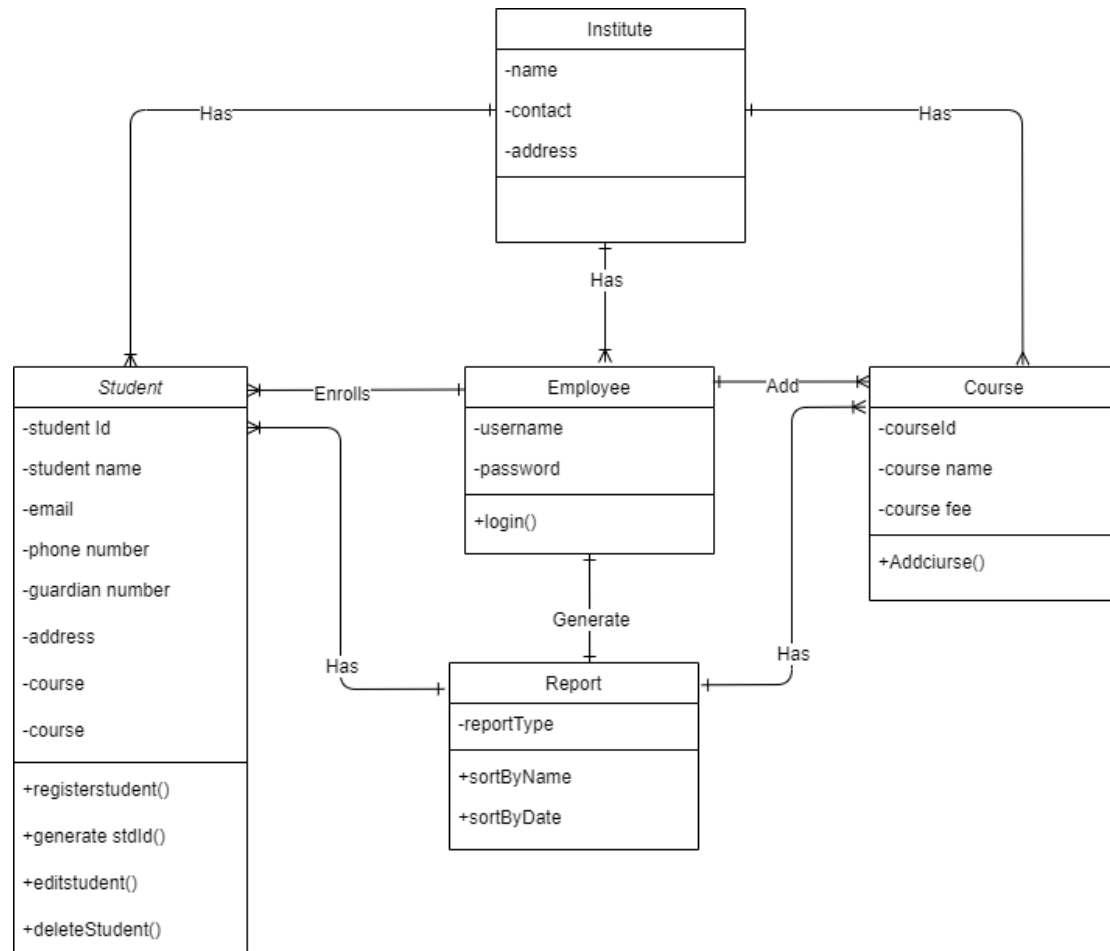


Figure 21: Class diagram for student registration system

## Individual Diagram

### Login individual diagram

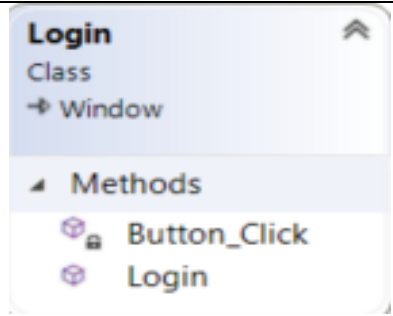
Method	Description	Figure
Button_click	This button redirects to home page if correct username and password is provided.	 <p>The screenshot shows a UML class diagram for the 'Login' class. It is a 'Class' and has a 'Window' reference. Under the 'Methods' section, there are two methods listed: 'Button_Click' and 'Login'.</p>

Table 1:login individual diagram

### Home Page Individual diagram

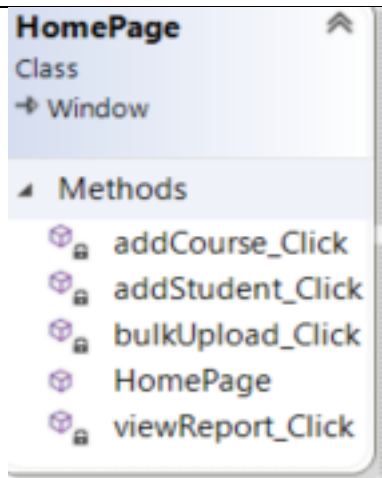
Method	Description	Figure
addCourse_Click	Opens Course adding form	 <p>The screenshot shows a UML class diagram for the 'HomePage' class. It is a 'Class' and has a 'Window' reference. Under the 'Methods' section, there are five methods listed: 'addCourse_Click', 'addStudent_Click', 'bulkUpload_Click', 'HomePage', and 'viewReport_Click'.</p>
addStudent_Click	Opens student registration form	
bulkUpload_Click	Opens page to upload data from excel.	
viewReport_Click	Opens report section	

Table 2:Home page individual diagram

**Add Student Individual Diagram**

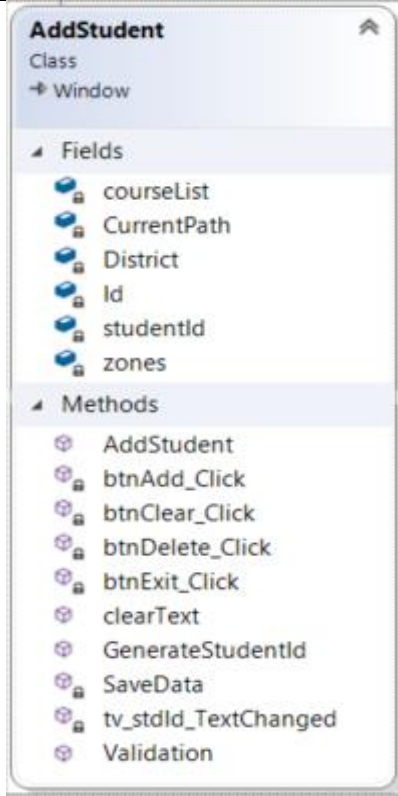
Methods	Description	Figure
btnAdd_click	Save new student data in xml and view in datagrid.	
btnClear_click	Clears the data from textbox and combobox.	
btnDelete_Click	Deletes the selected data.	
btnExit_Click	Close the student registration window.	
cleartext	Clears the data from textbox and other field after data is saved.	
GenerateStudentId	Generates unique student id	
SavesData	Saves student data in xml format	
Validation	Check if textbox is empty or not	

Table 3: Add student individual diagram



**View Report individual diagram**


Method	Description	Figure
Button_Click	Display Daily data in tabular form.	
Button_Click_1	Display weekly data in tabular form.	
Load ColumnChartData	Display column chart of student number and course.	
sortBy_SelectionChanged	Sort data by name and registration date.	

Table 4:view report page individual diagram

### Add Course individual diagram

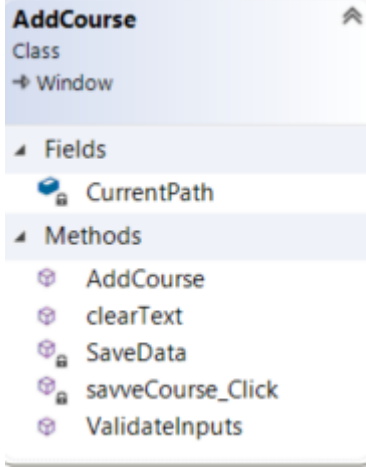
Method	Description	Figure
ClearText	Clears the inputted text from field	 <p>The screenshot shows the 'AddCourse' class in a software development environment. It is a 'Class' with a 'Window' icon. Under 'Fields', there is 'CurrentPath'. Under 'Methods', there are 'AddCourse', 'clearText', 'SaveData', 'saveCourse_Click', and 'ValidateInputs'.</p>
SaveData	Saves course details to xml.	
saveCourse_click	Calls saveData method and save course details.	
ValidateInputs	Checks wheather the fields are empty or not.	

Table 5:add course individual diagram

### Import Excel individual diagram

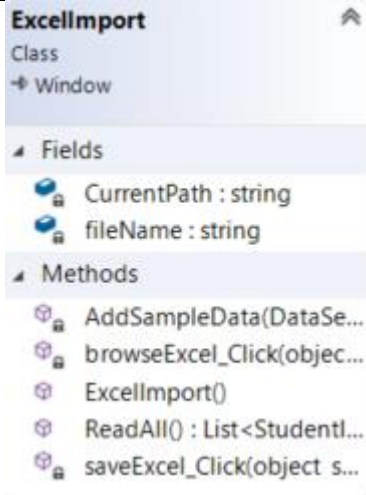
Method	Description	Figure
AddSampleData	It adds the data from excel to dataset.	 <p>The screenshot shows the 'ExcellImport' class in a software development environment. It is a 'Class' with a 'Window' icon. Under 'Fields', there are 'CurrentPath : string' and 'fileName : string'. Under 'Methods', there are 'AddSampleData(DataSe...', 'browseExcel_Click(objec...', 'ExcellImport()', 'ReadAll() : List&lt;Studentl...', and 'saveExcel_Click(object s...'.</p>
browseExcel_Click	It opens file chooser to chose csv file.	
ReadAll	It reads the data from excel.	
SaveExcel	It saves the read data from excel to xml	

Table 6:import excel individual diagram

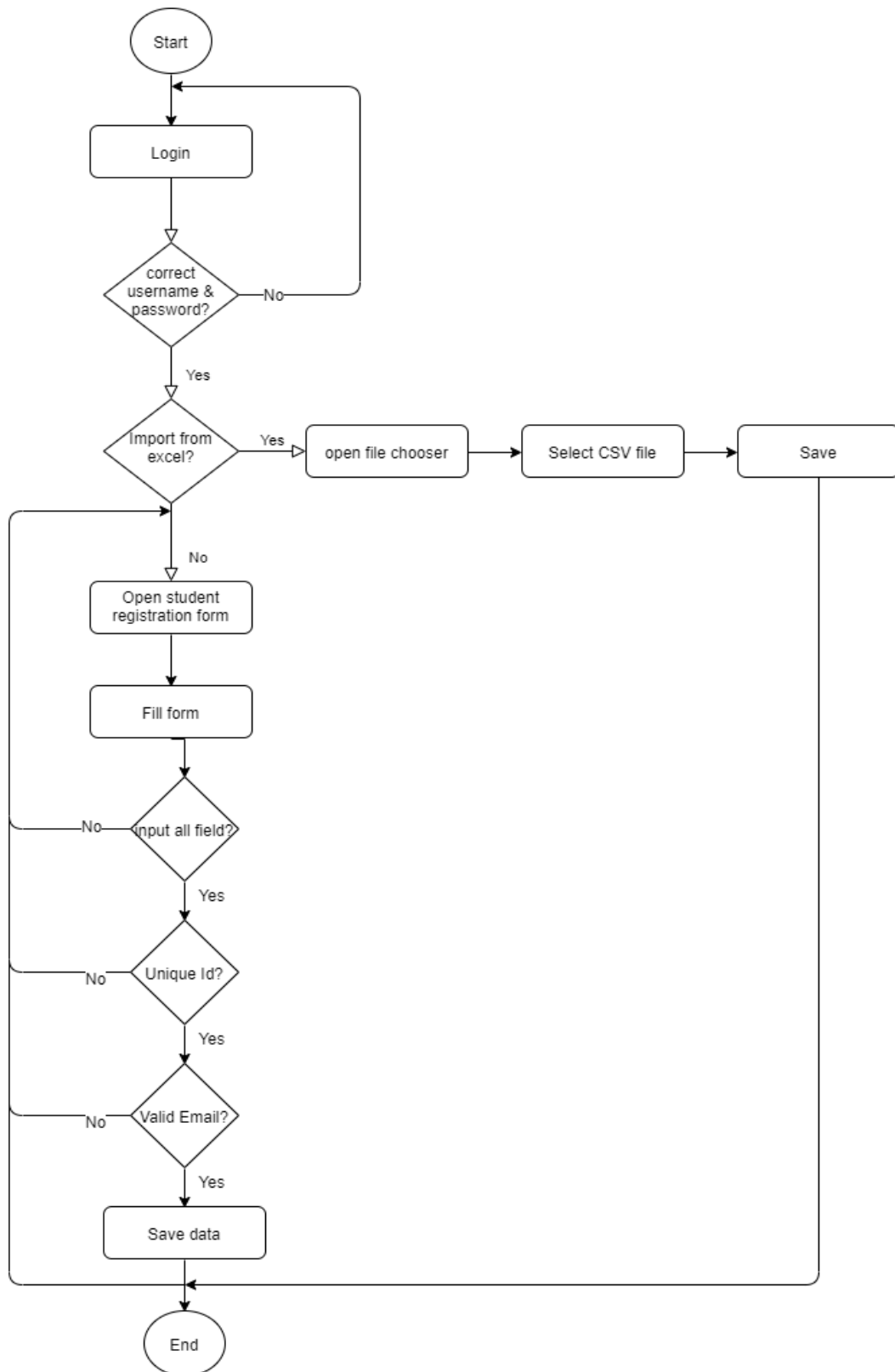
**Datahandler individual diagram**

Method	Description	Figure
CreateCourseTable	It creates xml format for course or creates course table	
CreateDataSet	It crates the dataset from table.	
CreateStudentTable	It creates the student table and also creates xml format for keeping record of stdent data.	

*Table 7: datahandler individual diagram***Student Info individual diagram**

Method	Description	Figure
ConvertToObject	It split the data using certain symbol and stores data in specific variable.	
StudentInfo	It contains getter and setter for Students data read from excel.	

*Table 8 student info individual diagram*

**Flow chart***Figure 22:Flowchart for student registration*

## Test Cases

### Test case 1

Objective	Open File Chooser
Action	Click on Browse Excel
Expected Result	Open file chooser dialog box and only show .csv format
Actual Result	File chooser opened
Conclusion	Test successful

Table 9:test case 1 :open file chooser

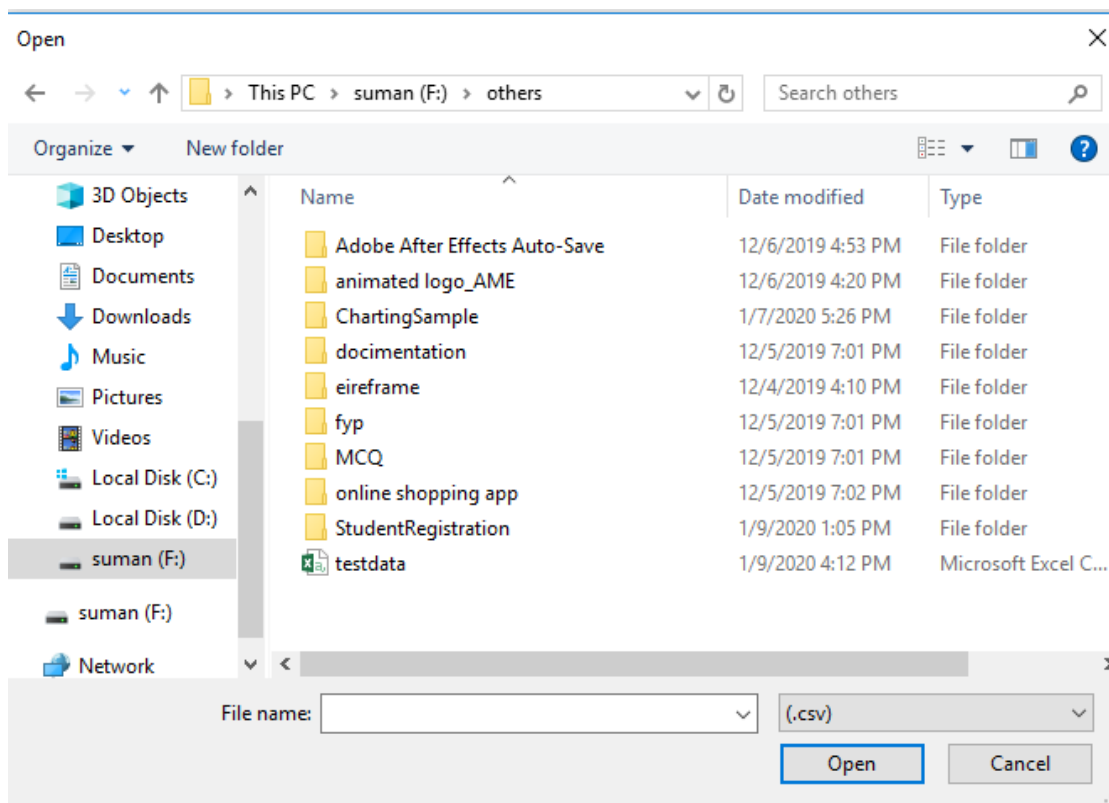


Figure 23:test case 1: open file chooser

### Test case 2

Objective	Import Excel
Action	Select csv file
Expected Result	Data from Csv should be displayed in table
Actual Result	Data displayed in table
Conclusion	Test successful

Table 10:test case 2: import excel

ExcellImport

F:\others\testdata.csv

**Browse Excel**

stdId	FirstName	LastName	gender	Zone	District	Tole	Phone	Email	guardianNo
7835424	Sushil	Gautam	male	gandaki	kaski	malepatan	982457562	something@gmail.com1	98224625
7835425	Ishwor	Sapkota	male	parbat	gorkha	lakeside	982457563	something@gmail.com2	98224626
7835426	Sita	Gautam	Female	bagmati	lamjung	jarebar	982457564	something@gmail.com3	98224627
7835427	Sushma	Bhandari	male	gandaki	kathmandu	newroad	982457565	something@gmail.com4	98224628
7835428	Laxmi Prasad	Bhandari	male	gandaki	lalaitpur	buspark	982457566	something@gmail.com5	98224629
7835429	Laxman	Adhikari	male	parbat	bhaktapur	prithvichowk	982457567	something@gmail.com6	98224630
7835430	Amrit	Adhikari	male	bagmati	kaski	amarsingh	982457568	something@gmail.com7	98224631
7835431	Upendra	Sharma	male	gandaki	gorkha	bagar	982457569	something@gmail.com8	98224632
7835432	Bimala	Acharya	Female	bagmati	kathmandu	deep	982457570	something@gmail.com9	98224633

**Save**

Figure 24:test case 2: import excel

### Test case 3

Objective	Add course
Action	Click on add course.
Expected Result	Course detail should be saved and displayed while enrolling student.
Actual Result	Course added displayed in student registration form
Conclusion	Test successful

Table 11:test case 3: add course

AddCourse

**Add Course**

Course Name :  
Application Development

Credit Hour :  
8

Course Fee :  
45211

**Save**

**Message**

Course Added Sucessfully

**OK**

StudentRegistration.exe

View Project Build Debug Test Analyze Tools Extensions Win

Debug Any CPU Continue

StudentRegistration.exe Lifecycle Events Thread:

AddStudent

Student ID : 21642889

First Name :

Zone :

Tole :

Course Enroll :  
Java

Email :  
Application Development

Figure 25:test case 3: add course

**Test case 4**

Objective	Email Validation
Action	Click on add student button
Expected Result	Wrong email format should be prevented displaying message in dialog box.
Actual Result	Message displayed saying invalid email format
Conclusion	Test successful

*Table 12:test case 4:email validation*

stdId	firstname	lastname	gender	ContactNo	email	CourseE	istrict	Tole	guardianNo	RegistrationDate
7835425	Ishwor	Sapkota	male	982457563	something@gmail.com2	Java				
7835426	Sita	Gautam	Female	982457564	something@gmail.com3	Python				
7835427	Sushma	Bhandari	male	982457565	something@gmail.com4	Database				
7835428	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Development	gandaki	lalaitpur	buspark	98224629
7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java	parbat	bhaktapur	prithvichowk	98224630
7835430	Amrit	Adhikari	male	982457568	something@gmail.com7	Python	bagmati	kaski	amarsingh	98224631
7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database	gandaki	gorkha	bagar	98224632
7835432	Bimala	Acharya	Female	982457570	something@gmail.com9	Java	bagmati	kathmandu	deep	98224633
17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Development	gandaki	kaski	malepatan	9824623563

*Figure 26:test case 4:email validation***Test case 5**

Objective	Contact number should only accept number.
Action	Click Add student button
Expected Result	Contact number should only accept number and display error message if any other character in input,
Actual Result	Error message displayed saying invalid phone number
Conclusion	Test successful

*Table 13:test case 5: phone no validation*

Student ID : 75316388 Last Name : Bhandari

First Name : Suman District : baitadi

Zone : bagmati Gender : ☐ Male ☐ Female ☐ Other

Tole : malepatan Contact No : test

Course Enroll : Application Developer Guardian No : 9804148281

Email : suman@gmail.com Your Registration Id is : 17144161

**Add** **Clear** **Delete** **Exit**

stdId	firstname	lastname	gender	ContactNo	email	Database	gandaki	katrimanou	newroad	guardianNo	RegistrationDate
7835425	Ishwor	Sapkota	male	982457563	something@gmail.com2					98224626	1/9/2020 12:00:00 AM
7835426	Sita	Gautam	Female	982457564	something@gmail.com3					98224627	1/9/2020 12:00:00 AM
7835427	Sushma	Bhandari	male	982457565	something@gmail.com4					98224628	1/9/2020 12:00:00 AM
7835428	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Development	gandaki	lalaitpur	buspark	98224629	1/9/2020 12:00:00 AM
7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java	parbat	bhaktapur	prithvichowk	98224630	1/9/2020 12:00:00 AM
7835430	Amrit	Adhikari	male	982457568	something@gmail.com7	Python	bagmati	kaski	amarsingh	98224631	1/9/2020 12:00:00 AM
7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database	gandaki	gorkha	bagar	98224632	1/9/2020 12:00:00 AM
7835432	Bimala	Acharya	Female	982457570	something@gmail.com9	Java	bagmati	kathmandu	deep	98224633	1/9/2020 12:00:00 AM
17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Development	gandaki	kaski	malepatan	9824623563	1/9/2020 12:00:00 AM

Figure 27:test case 5: phone no validation

## Test case 6

Objective	Auto generate student Id
Action	Click on Add student
Expected Result	Student Id should be auto generated and displayed in textbox.
Actual Result	Student id auto generated and displayed
Conclusion	Test successful

Table 14:test case 6:auto generate student id

Student ID : 17144161 Last Name :

First Name : District :

Zone : Gender : ☐ Male ☐ Female ☐ Other

Tole : Contact No :

Course Enroll : Guardian No :

Email : Your Registration Id is : 17144161

Figure 28:test case 6:auto generate student id



## Test case 7

Objective	Do not allow student registration with same Id.
Action	Student Id changed so that it matches already saved student Id
Expected Result	Error message saying student with this already exist should be displayed.
Actual Result	Error message displayed saying student with this Id already exist.
Conclusion	Test successful

Table 15:test case 7 unique student id

The screenshot shows a web application interface for student registration. The form fields are filled with the following data:

- Student ID: 7835425
- Last Name: Bhandari
- First Name: Suman
- District: baitadi
- Zone: bagmati
- Tole: malepatan
- Course Enroll: Application Developer
- Email: suman@gmail.com

An alert dialog box is displayed in the center of the screen with the message "Student Id already Exist" and an "OK" button. Below the form, there is a table showing a list of registered students. The first row of the table is highlighted in yellow, corresponding to the student ID 7835425.

stdid	firstname	lastname	gender	ContactNo	email	CourseEnroll	zone	District	Tole	guardianNo	RegistrationDate
7835425	Ishwor	Sapkota	male	982457563	something@gmail.com2	Java	parbat	gorkha	lakeside	98224626	1/9/2020 12:00:00 AM
7835426	Sita	Gautam	Female	982457564	something@gmail.com3	Python	bagmati	lamjung	jarebar	98224627	1/9/2020 12:00:00 AM
7835427	Sushma	Bhandari	male	982457565	something@gmail.com4	Database	gandaki	kathmandu	newroad	98224628	1/9/2020 12:00:00 AM
7835428	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Development	gandaki	lalaitpur	buspark	98224629	1/9/2020 12:00:00 AM
7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java	parbat	bhaktapur	prithvichowk	98224630	1/9/2020 12:00:00 AM
7835430	Amrit	Adhikari	male	982457568	something@gmail.com7	Python	bagmati	kaski	amarsingh	98224631	1/9/2020 12:00:00 AM
7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database	gandaki	gorkha	bagar	98224632	1/9/2020 12:00:00 AM
7835432	Bimala	Acharya	Female	982457570	something@gmail.com9	Java	bagmati	kathmandu	deep	98224633	1/9/2020 12:00:00 AM
17144161	Suman	Bhandari	male	9827120831	shumanbhandari20@gmail.com	Application Development	gandaki	kaski	malepatan	9824623563	1/9/2020 12:00:00 AM

Figure 29:test case 7 unique student id

## Test case 8

Objective	Delete student Data
Action	Click on student data and click delete button.
Expected Result	Student data should be permanently deleted from xml file.
Actual Result	Student data deleted
Conclusion	Test successful

Table 16:test case 8: delete student data

Student ID :  Last Name :

First Name :  District :

Zone :  Gender : ☐ Male ☒ Female ☐ Other

Tole :

Course Enroll :

Email :

Add  
Clear  
Delete  
Exit

Student ID :  Last Name :

First Name :  District :

Zone :  Gender : ☐ Male ☒ Female ☐ Other

Tole :  Contact No :

Course Enroll :  Guardian No :

Email :  Your Registration Id is : 49774274

Add  
Clear  
Delete  
Exit

Deleted successfully

Id is : 49774274

Ok

stid	firstname	lastname	gender	ContactNo	email	CourseEnroll	zone	District	Tole	guardianNo	RegistrationDate
7835426	Suman	Bhandari	male	982112081	suman@gmail.com	Application Development	bagmati	kaski	malepatan	982449281	1/9/2020 3:55:04 AM
7835428	Sita	Gautam	female	982457564	something@gmail.com3	Python	bagmati	lanjung	janibar	98244927	1/9/2020 12:00:00 AM
7835427	Sushma	Bhandari	male	982457565	something@gmail.com4	Database	gandaki	kathmandu	newroad	98244928	1/9/2020 12:00:00 AM
7835429	Laxmi Prasad	Bhandari	male	982457566	something@gmail.com5	Application Development	gandaki	lalitpur	busspark	98244929	1/9/2020 12:00:00 AM
7835429	Laxman	Adhikari	male	982457567	something@gmail.com6	Java	parbat	shaktapur	prithivichowk	98244930	1/9/2020 12:00:00 AM
7835430	Jenrit	Adhikari	male	982457568	something@gmail.com7	Python	bagmati	kaski	amanasingh	98244931	1/9/2020 12:00:00 AM
7835431	Upendra	Sharma	male	982457569	something@gmail.com8	Database	gandaki	goriha	bagar	98244932	1/9/2020 12:00:00 AM
7835432	Bimala	Acharya	female	982457570	something@gmail.com9	Java	bagmati	kathmandu	deep	98244933	1/9/2020 12:00:00 AM
17144181	Suman	Bhandari	male	982112081	sumanbhandari20@gmail.com	Application Development	gandaki	kaski	malepatan	98244928	1/9/2020 12:00:00 AM

Figure 30:test case 8: delete student data

## Sorting Algorithm

Bubble sort, sometimes referred to as sinking sort, is a simple sorting algorithm that repeatedly steps through the list, compares adjacent elements and swaps them if they are in the wrong order (.tutorialspoint, 2020). The pass through the list is repeated until the list is sorted. Bubble sort has a worst-case and average complexity of  $O(n^2)$ , where  $n$  is the number of items being sorted. When the list is already sorted (best-case), the complexity of bubble sort is only  $O(n)$  (Astrachan, 2019).

It compares all the 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 (Paul Biggar, David Gregg, 2005).

### Working Mechanism

Below is the example showing how bubble sort algorithm works:

First Pass:

Lets take an unsorted array .

( 5 1 4 2 8 )  $\rightarrow$  ( 1 5 4 2 8 ), Here, algorithm compares the first two elements, and swaps since  $5 > 1$ .

( 1 5 4 2 8 )  $\rightarrow$  ( 1 4 5 2 8 ), Swap since  $5 > 4$

( 1 4 5 2 8 )  $\rightarrow$  ( 1 4 2 5 8 ), Swap since  $5 > 2$

( 1 4 2 5 8 )  $\rightarrow$  ( 1 4 2 5 8 ), Now, since these elements are already in order ( $8 > 5$ ), algorithm does not swap them.

Second Pass:

( 1 4 2 5 8 )  $\rightarrow$  ( 1 4 2 5 8 )

( 1 4 2 5 8 )  $\rightarrow$  ( 1 2 4 5 8 ), Swap since  $4 > 2$

( 1 2 4 5 8 )  $\rightarrow$  ( 1 2 4 5 8 )

( 1 2 4 5 8 )  $\rightarrow$  ( 1 2 4 5 8 )

Now, the array is already sorted, but our algorithm does not know if it is completed. The algorithm needs one whole pass without any swap to know it is sorted.

Third Pass:

( 1 2 4 5 8 )  $\rightarrow$  ( 1 2 4 5 8 )

( 1 2 4 5 8 )  $\rightarrow$  ( 1 2 4 5 8 )

( 1 2 4 5 8 )  $\rightarrow$  ( 1 2 4 5 8 )

( 1 2 4 5 8 )  $\rightarrow$  ( 1 2 4 5 8 )

And finally, we get the sorted array. Same algorithm is used in the development process to sort student data in ascending order by name and by registration date.

## **Related Research**

### **WPF in Action with Visual Studio 2008( (Arlen Feldman, Maxx Daymon, 2008))**

WPF in Action with visual studio is a book written by Arlen Fedman and Maxx Daymon is an awesome book which give great idea on working with WPF. This book helps us to learn about WPF desktop app development in the simplest, funny & Graphical way. This book made my journey of learning desktop app development much easier and fun.

### **Microsoft .Net(<https://docs.microsoft.com>)**

Microsoft.Net is the official website of Microsoft and it provide all the information of visual studio and working with visual studio. I have got so many information from this website when I was in confusion.

### **WPF Tutorial([www.wpf-tutorial.com](http://www.wpf-tutorial.com))**

WPF tutorial, currently consisting of 125 articles, where you'll learn to make your own applications using the WPF UI framework. This is an awesome website which also help in completion of my project in time. Functionalities like importing data from excel was done with the help of this website.

### **SearchCIO ([searchcio.techtarget.com](http://searchcio.techtarget.com))**

SearchCIO.com provides technology management strategies designed exclusively for the enterprise CIO. Our award-winning team of editors and industry luminaries offer strategic advice and technology best practices to help streamline global IT operations. This website was also used during development of my project.

### **techopedia ([www.techopedia.com](http://www.techopedia.com))**

Techopedia is the go-to tech source for professional IT insight and inspiration, plus it tirelessly feed anyone who is proud to be called a "geek" with the informational and entertaining content they need.

## Reflection

Student Registration system is a desktop app developed using visual studio 2019 using C# programming language in WPF design form. The system is user friendly and have simple and clean UI. Logic used in the system reflects the real working environment of educational institute.

Lots of research on different relevant topics was done in order to complete this project in time. Research on topics like visual studio, C# programming, WPF etc. was done before and during development of the project. I also surf different websites, read different books and journals as well watched different tutorial videos. While doing this project I faced many problems but those problems were solved with the help of tutor and friends and solving the problem was so happy moment.

The main features of the system include addition of new course, Registration of student in specific one course, Bulk registration of student by importing from csv and viewing daily and weekly report.

The project not only complete the coursework in time but also taught me many new ideas. I got more working experience in working in visual studio in C# programming language. I learned features like saving data in xml and retrieving data from xml. Also importing data from excel in csv format was learnt. Similarly, working with charts was also learnt. Thus, I had a great experience on working on C# programming language in visual studio in developing a desktop-based application.

## Conclusion

Task assigned in the coursework was so tough and it was completed in time with strong determination and hard labor. I have completed the task by taking help from lecturer, friends and different books, journals and website. The main objective of the task was to develop a student registration system with features of enrolling student in course by filling form and by importing data from excel and generating weekly and daily report.

The tasks assigned in the coursework were not easy at all. It required lots of labor and research. For the successful completion of all the tasks, each task was carried out in steps, in every step deploying the full effort. At first, lots of study and research was done on the relevant topics like C# programming language, visual studio WPF, excel import, XML etc. In the next step UI was designed in WPF window. After that, login system was designed and adding course was done. Then student registration was done by saving data in xml and retrieving course while enrolling student. And then student data was imported from excel in csv format. And lastly, weekly and daily report was generated with bar graph along with sorting feature. Finally, report was written and submission was done.

This project didn't only complete all the tasks assigned in the coursework in time, but also, helped in developing various skills and taught many things which can be really useful in future career as a programmer. While being involved in this project, sound knowledge on XML, Excel data import, C#, visual studio, sorting algorithm and graph generation was acquired. And these learning would surely be a lot helpful in the pursue of development of career as a good programmer. Valuable experience has been gained working on this project. All in all, although the tasks were tough and required nights of hard work and labor, successfully completing those tough tasks was a great fun.

## References

.tutorialspoint, 2020. *.tutorialspoint*. [Online]

Available at:

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

[Accessed 04 01 2020].

Arlen Feldman, Maxx Daymon, 2008. *WPF in Action with Visual Studio 2008*. s.l.: Manning.

Astrachan, O., 2019. Bubble Sort: An Archaeological Algorithmic Analysis. In: s.l.:s.n.

Paul Biggar, David Gregg, 2005. *Sorting in the Presence of Branch Prediction*. s.l.:s.n.



## Appendix

### Login Page

```

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

        private void Button_Click(object sender, RoutedEventArgs e)
        {
            string username = tvUsername.Text;
            string password = tvPassword.Password;
            if (username == "" || password == "")
            {
                MessageBox.Show("username or password cannot be empty");
            }
            else
            {
                if (username == "admin")
                {
                    if (password == "admin")
                    {
                        HomePage home = new HomePage();
                        home.Show();
                        this.Hide();
                    }
                    else
                    {
                        MessageBox.Show("Incorrect password !! Please input correct password of if forgotten contact developer");
                    }
                }
                else
                {
                    MessageBox.Show("Incorrect username !! Please input correct password of if forgotten contact developer");
                }
            }
        }
    }
}

```

**Home page**

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

        private void addStudent_Click(object sender, RoutedEventArgs e)
        {
            AddStudent addstd = new AddStudent();
            addstd.Show();
        }

        private void bulkUpload_Click(object sender, RoutedEventArgs e)
        {
            ExcelImport excelImport = new ExcelImport();
            excelImport.Show();
        }

        private void viewReport_Click(object sender, RoutedEventArgs e)
        {
            ViewReport viewReport = new ViewReport();
            viewReport.Show();
        }

        private void addCourse_Click(object sender, RoutedEventArgs e)
        {
            AddCourse addCourse = new AddCourse();
            addCourse.Show();
        }
    }
}
```

**Add Course**

```

namespace StudentRegistration
{
    /// <summary>
    /// Interaction logic for AddCourse.xaml
    /// </summary>
    public partial class AddCourse : Window
    {
        private int course;
        private string CurrentPath =
System.AppDomain.CurrentDomain.BaseDirectory+ "\\StudentCWData.xml";
        public AddCourse()
        {
            InitializeComponent();
            var dataHandler = new datahandler();
            var dataSet = dataHandler.CreateDataSet();
            if (File.Exists(CurrentPath))
            {
                dataSet.ReadXml(CurrentPath);
            }
            else
            {
                dataSet.WriteXml(CurrentPath);
            }
        }

        private void savveCourse_Click(object sender, RoutedEventArgs e)
        {
            var dataHandler = new datahandler();
            var dataSet = dataHandler.CreateDataSet();
            SaveCourse(dataSet);

            if (ValidateInputs())
            {
                if (File.Exists(CurrentPath))
                {
                    dataSet.ReadXml(CurrentPath);
                    dataSet.WriteXml(CurrentPath);
                    MessageBox.Show("Course Added Sucessfully", "Message",
MessageBoxButton.OK, MessageBoxImage.Information);
                    clearText();
                }
                else
                {
                    dataSet.WriteXml(CurrentPath);
                    MessageBox.Show("Course Added Sucessfully", "Message",
MessageBoxButton.OK, MessageBoxImage.Information);
                    clearText();
                }
            }
        }
        //This method is used to save course data in xml by taking input from
form
        private void SaveCourse(DataSet dataSet)
        {
            var dr = dataSet.Tables["Course"].NewRow();
            dr["CourseName"] = courseName.Text;
            dr["CreditHours"] = courseFee.Text;

```

```

        dr["CourseFee"] = creditHours.Text;
        dataSet.Tables["Course"].Rows.Add(dr);
    }
    //this method is used to clear fields
    public void clearText()
    {
        courseName.Text = "";
        courseFee.Text = "";
        creditHours.Text = "";
    }
    //This method is used to check the user input
    public Boolean ValidateInputs()
    {
        if (courseName.Text.Equals(""))
        {
            MessageBox.Show("Course Name cannot be Empty", "Alert",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            return false;
        }
        else if (courseFee.Text.Equals(""))
        {
            MessageBox.Show("Course Fee Name cannot be Empty", "Alert",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            return false;
        }
        else if (creditHours.Text.Equals(""))
        {
            MessageBox.Show("Credit Hours cannot be Empty", "Alert",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            return false;
        }
        else if (course > 0)
        {
            MessageBox.Show("Course already Exist", "Alert",
                MessageBoxButtons.OK, MessageBoxIcon.Information);
            return false;
        }

        return true;
    }
    //this method is used to check if same course exist or not
    public int courseNameChecker()
    {
        var dataset = new DataSet();
        dataset.ReadXml(CurrentPath);
        DataTable studentTable = dataset.Tables["course"];

        int course = 0;

        for (int i = 0; i < studentTable.Rows.Count; i++)
        {
            string col = studentTable.Rows[i]["CourseName"].ToString();
            if (col == courseName.Text)
            {
                course++;
            }
        }
        return course;
    }

```

```

    }

    private void courseName_TextChanged(object sender, TextChangedEventArgs
e)
    {
        course = courseNameChecker();
    }
}

```

### Register Student

```

namespace StudentRegistration
{
    /// <summary>
    /// Interaction logic for AddStudent.xaml
    /// </summary>
    public partial class AddStudent : Window
    {
        private List<string> courseList = new List<string>();
        private List<string> zones = new List<string>();
        private List<string> District = new List<string>();
        private string studentId, selectedId;
        private int _Id;
        private string _gender;
        private bool _isValidEmail;
        private bool _isContactNumber;
        private bool _isGuardianContactNumber;
        private string CurrentPath =
System.AppDomain.CurrentDomain.BaseDirectory + "\\StudentCWData.xml";

        public AddStudent()
        {
            InitializeComponent();
            studentId= GenerateStudentId();
            tv_stdId.Text = studentId;

            var dataHandler = new datahandler();
            var dataSet = dataHandler.CreateDataSet();
            DataTable studentTable = dataSet.Tables["student"];
            _Id = studentTable.Select("stdId =
"+"+tv_stdId.Text+"").Count<DataRow>();
            stdId.Content = studentId;
            if (File.Exists(CurrentPath))
            {
                dataSet.ReadXml(CurrentPath);
            }
            else
            {
                dataSet.WriteXml(CurrentPath);
            }
            DataGridTest.ItemsSource = new DataView(dataSet.Tables["Student"]);
            DataTable courseTable = dataSet.Tables["Course"];
            foreach (DataRow row in courseTable.Rows)
            {
                var courseName = row["CourseName"].ToString();
                courseList.Add(courseName);
            }
            course.ItemsSource = courseList;

            zones.Add("gandaki");

```

```
zones.Add("bagmati");
zones.Add("bheri");
zones.Add("dhawalagiri");
zones.Add("gandaki");
zones.Add("janakpur");
zones.Add("karnali");
zones.Add("koshi");
zones.Add("lumbini");
zones.Add("mahakali");
zones.Add("mechi");
zones.Add("narayani");
zones.Add("rapti");
zones.Add("sagarmatha");
zones.Add("seti");
cb_zone.ItemsSource = zones;

District.Add("achham");
District.Add("arghakhanchi");
District.Add("baglung");
District.Add("baitadi");
District.Add("bajhang");
District.Add("bajura");
District.Add("banke");
District.Add("bara");
District.Add("bardiya");
District.Add("bhaktapur");
District.Add("bhojpur");
District.Add("chitwan");
District.Add("dadeldhura");
District.Add("dailekh");
District.Add("dang deukhuri");
District.Add("darchula");
District.Add("dhading");
District.Add("dhankuta");
District.Add("dhanusa");
District.Add("dholkha");
District.Add("dolpa");
District.Add("doti");
District.Add("gorkha");
District.Add("gulmi");
District.Add("humla");
District.Add("ilam");
District.Add("jajarkot");
District.Add("jhapa");
District.Add("jumla");
District.Add("kailali");
District.Add("kalikot");
District.Add("kanchanpur");
District.Add("kapilvastu");
District.Add("kaski");
District.Add("kathmandu");
District.Add("kavrepalanchok");
District.Add("khotang");
District.Add("lalitpur");
District.Add("lamjung");
District.Add("mahottari");
District.Add("makwanpur");
District.Add("manang");
District.Add("morang");
District.Add("mugu");
District.Add("mustang");
District.Add("myagdi");
```

```

District.Add("nawalparasi");
District.Add("nuwakot");
District.Add("okhaldhunga");
District.Add("palpa");
District.Add("panchthar");
District.Add("parbat");
District.Add("parsa");
District.Add("pyuthan");
District.Add("ramechhap");
District.Add("rasuwa");
District.Add("rautahat");
District.Add("rolpa");
District.Add("rukum");
District.Add("rupandehi");
District.Add("salyan");
District.Add("sankhuwasabha");
District.Add("saptari");
District.Add("sarlahi");
District.Add("sindhuli");
District.Add("sindhupalchok");
District.Add("siraha");
District.Add("solukhumbu");
District.Add("sunsari");
District.Add("surkhet");
District.Add("syangja");
District.Add("tanahu");
District.Add("taplejung");
District.Add("terhathum");
District.Add("udayapur");
cb_district.ItemsSource = District;
}

private void btnAdd_Click(object sender, RoutedEventArgs e)
{
    var dataHandler = new datahandler();
    var dataSet = dataHandler.CreateDataSet();
    studentId = GenerateStudentId();

    if (Validation())
    {
        SaveStudent(dataSet);
        if (File.Exists(CurrentPath))
        {
            dataSet.ReadXml(CurrentPath);
            dataSet.WriteXml(CurrentPath);
            MessageBox.Show("Enroled Sucessfully", "Message",
                MessageBoxButton.OK, MessageBoxImage.Information);
            clearText();
            tv_stdId.Text = studentId;
        }
        else
        {
            dataSet.WriteXml(CurrentPath);
            MessageBox.Show("Enroled Sucessfully", "Message",
                MessageBoxButton.OK, MessageBoxImage.Information);
            clearText();
        }
    }
    else
    {

```

```

        dataSet.ReadXml(CurrentPath);
    }
}
//this method saves student data to xml
private void SaveStudent(DataSet dataSet)
{
    var dr = dataSet.Tables["Student"].NewRow();
    dr["stdId"] = tv_stdId.Text;
    dr["firstName"] = tv_firstName.Text;
    dr["lastName"] = tv_lastName.Text;
    dr["gender"] = _gender;
    dr["ContactNo"] = tv_contactNo.Text;
    dr["email"] = tv_email.Text;
    dr["CourseEnroll"] = course.Text;
    dr["zone"] = cb_zone.Text;
    dr["District"] = cb_district.Text;
    dr["Tole"] = tv_tole.Text;
    dr["guardianNo"] = tv_guardianNo.Text;
    dr["RegistrationDate"] = DateTime.Now;
    dataSet.Tables["Student"].Rows.Add(dr);
    DataGridTest.ItemsSource = new DataView(dataSet.Tables["Student"]);
}
//this method is used to clear fields
public void clearText()
{
    tv_firstName.Text = "";
    tv_lastName.Text = "";
    tv_email.Text = "";
    tv_contactNo.Text = "";
    tv_email.Clear();
    tv_tole.Clear();
    tv_guardianNo.Clear();
    tv_contactNo.Clear();
    cb_district.SelectedIndex=-1;
    cb_zone.SelectedIndex = -1;
    course.SelectedIndex = -1;
}

//this method is used to validate user input before saving data
public Boolean Validation()
{
    if (tv_firstName.Text.Equals(""))
    {
        MessageBox.Show("First Name cannot be Empty", "Alert",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        return false;
    }
    else if (tv_lastName.Text.Equals(""))
    {
        MessageBox.Show("Last Name cannot be Empty", "Alert",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        return false;
    }
    else if (cb_district.Text.Equals(""))
    {
        MessageBox.Show("District cannot be Empty", "Alert",
        MessageBoxButtons.OK, MessageBoxIcon.Information);
        return false;
    }
    else if (cb_zone.Text.Equals(""))
    {

```



```

        MessageBox.Show("Zone cannot be Empty", "Alert",
        MessageBoxButton.OK, MessageBoxImage.Information);
        return false;
    }
    else if (tv_contactNo.Text.Equals(""))
    {
        MessageBox.Show("Contact Number cannot be
        Empty"+_isContactNumber, "Alert", MessageBoxButton.OK,
        MessageBoxImage.Information);
        return false;
    }
    else if (!_isContactNumber)
    {
        MessageBox.Show("please input valid Contact Number", "Alert",
        MessageBoxButton.OK, MessageBoxImage.Information);
        return false;
    }
    else if (!_isGuardianContactNumber)
    {
        MessageBox.Show("Please input valid Guardian Nuumber", "Alert",
        MessageBoxButton.OK, MessageBoxImage.Information);
        return false;
    }
    else if (tv_email.Text.Equals(""))
    {
        MessageBox.Show("Email cannot be Empty", "Alert",
        MessageBoxButton.OK, MessageBoxImage.Information);
        return false;
    }
    else if (!_isValidEmail)
    {
        MessageBox.Show("Email not valid", "Alert",
        MessageBoxButton.OK, MessageBoxImage.Information);
        return false;
    }
    else if (_Id>0)
    {
        MessageBox.Show("Student Id already Exist", "Alert",
        MessageBoxButton.OK, MessageBoxImage.Information);
        return false;
    }

    return true;
}

//clear the field on clicking clear button
private void btnClear_Click(object sender, RoutedEventArgs e)
{
    clearText();
}

//close the window on clicking exit button
private void btnExit_Click(object sender, RoutedEventArgs e)
{
    this.Close();
}

//delete selccted students data on clicking delete button
private void btnDelete_Click(object sender, RoutedEventArgs e)
{
    if (!selectedId.Equals(""))

```

```

        {
            XDocument doc = XDocument.Load(CurrentPath);
            var removeStudent = doc.Descendants("Student").Where(c =>
c.Element("stdId").Value == selectedId).FirstOrDefault();
            removeStudent.Remove();
            doc.Save(CurrentPath);
            var result=MessageBox.Show("Deleted sucessfully", "Alert",
MessageBoxButton.OK, MessageBoxImage.Information);

            if ( result==MessageBoxResult.OK)
            {
                var dataHandler = new datahandler();
                var dataSet = dataHandler.CreateDataSet();
                dataSet.ReadXml(CurrentPath);
                DataGridViewTest.ItemsSource = new
DataView(dataSet.Tables["Student"]);
            }
        }
        else
        {
            MessageBox.Show("Please select data before deleting", "Alert",
MessageBoxButton.OK, MessageBoxImage.Information);
        }
    }

    //this method is used ti generate unique student id
    public string GenerateStudentId()
    {
        var bytes = new byte[4];
        var randomNo = RandomNumberGenerator.Create();
        randomNo.GetBytes(bytes);
        uint random = BitConverter.ToUInt32(bytes, 0) % 100000000;
        return String.Format("{0:D8}", random);
    }

    private void tv_stdId_TextChanged(object sender, TextChangedEventArgs
e)
    {
        var dataHandler = new datahandler();
        var dataSet = dataHandler.CreateDataSet();
        DataTable studentTable = dataSet.Tables["student"];
        _Id = studentIdChecker();
        if (_Id > 0)
        {
            stdId.Content = "Student Id Already Exit please change student
Id";
        }
        else
        {
            stdId.Content = tv_stdId.Text;
        }
    }

    //this method is used to check if student with the same id exist or not
    public int studentIdChecker()
    {
        var dataset = new DataSet();
        dataset.ReadXml(CurrentPath);
        DataTable studentTable = dataset.Tables["student"];
        int stdId = 0;
        for (int i = 0; i < studentTable.Rows.Count; i++)
        {

```

```

        string col = studentTable.Rows[i]["stdId"].ToString();
        if (col == tv_stdId.Text)
        {
            stdId++;
        }
    }
    return stdId;
}

e) private void tv_email_TextChanged(object sender, TextChangedEventArgs
{
    _isValidEmail = IsValidEmail(tv_email.Text);
}
//the method validates the email input
private bool IsValidEmail(string emailAddress)
{
    const string validEmailPattern =
@"^(?!\\.)(\"([^\\"r\\]|\\\"\\r\\\")*\"|"
    + @"([-a-z0-
9!#$%&'*/+=?^_`{|}~]|(?<!\.)\.)*(?<!\.)"
    + @"@[a-z0-9][\w\.-]*[a-z0-9]\.[a-
z][a-z\.\.]*[a-z]$$";

    return new Regex(validEmailPattern,
RegexOptions.IgnoreCase).IsMatch(emailAddress);
}
//this method is used to validate input is number or not
private bool isInputNumber(string number)
{
    const string validNumberPattern = "[0-9]+";

    return new Regex(validNumberPattern).IsMatch(number);
}

private void tv_contactNo_TextChanged(object sender,
TextChangedEventArgs e)
{
    _isContactNumber = isInputNumber(tv_contactNo.Text);
}

private void DataGridTest_SelectionChanged(object sender,
SelectionChangedEventArgs e)
{
    DataGrid datagrid = sender as DataGrid;
    if
((DataRow)datagrid.ItemContainerGenerator.ContainerFromIndex(datagrid.Se
lectedIndex) != null)
    {
        DataRow row =
(DataGridRow)datagrid.ItemContainerGenerator.ContainerFromIndex(datagrid.Sel
ectedIndex);
        DataGridCell rowColumn =
datagrid.Columns[0].GetCellContent(row).Parent as DataGridCell;
        selectedId = ((TextBlock)rowColumn.Content).Text;
    }
}

private void RadioButton_Checked(object sender, RoutedEventArgs e)

```

```

    {
        _gender = "Male";
    }

    private void RadioButton_Checked_1(object sender, RoutedEventArgs e)
    {
        _gender = "Female";
    }

    private void RadioButton_Checked_2(object sender, RoutedEventArgs e)
    {
        _gender = "Other";
    }

    private void tv_guardianNo_TextChanged(object sender,
    TextChangedEventArgs e)
    {
        _isGuardianContactNumber = isInputNumber(tv_guardianNo.Text);
    }
}

```

#### Import From Excel

```

namespace StudentRegistration
{
    /// <summary>
    /// Interaction logic for ExcelImport.xaml
    /// </summary>
    public partial class ExcelImport : Window
    {
        private string CurrentPath =
        System.AppDomain.CurrentDomain.BaseDirectory + "\\StudentCWData.xml";
        private string fileName;
        public ExcelImport()
        {
            InitializeComponent();
        }
        //this method opens file choser to select csv file
        private void browseExcel_Click(object sender, RoutedEventArgs e)
        {
            OpenFileDialog openfile = new OpenFileDialog();
            openfile.DefaultExt = ".csv";
            openfile.Filter = "(.csv)|*.csv";
            openfile.ShowDialog();
            fileName = openfile.FileName;
            excelFilepath.Text = fileName;
            var student = ReadData();
            datagrid.ItemsSource = student;
        }
        //this method is used to read data from csv file
        public List<StudentInfo> ReadData()
        {
            try
            {
                if (!File.Exists(fileName))
                {
                    throw new FileNotFoundException("Student Info file doesn't
exist");
                }
            }
        }
    }
}

```

```

        catch (Exception ex)
        {
            MessageBox.Show("Sorry! unexpected Error occurred! 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();
            ExcelImport excellImport = new ExcelImport();
            excellImport.Show();
        }
        return students;
    }

    //this method is used to save data from excel to xml
    private void saveExcel_Click(object sender, RoutedEventArgs e)
    {
        var dataHandler = new datahandler();
        var dataSet = dataHandler.CreateDataSet();
        SaveData(dataSet);
        if (File.Exists(CurrentPath))
        {
            dataSet.ReadXml(CurrentPath);
            dataSet.WriteXml(CurrentPath);
            System.Windows.MessageBox.Show("Enroled Sucessfully",
                "Message", MessageBoxButton.OK, MessageBoxImage.Information);
        }
        else
        {
            dataSet.WriteXml(CurrentPath);
            System.Windows.MessageBox.Show("Enroled Sucessfully",
                "Message", MessageBoxButton.OK, MessageBoxImage.Information);
            if (MessageBoxResult.OK.Equals(1))
            {
                this.Close();
            }
        }
    }

    private void SaveData(DataSet dataSet)
    {

```

```

var stdData = ReadData();
foreach (StudentInfo students in stdData)
{
    var dr = dataSet.Tables["student"].NewRow();
    dr["stdId"] = students.stdId;
    dr["firstName"] = students.FirstName;
    dr["lastName"] = students.LastName;
    dr["gender"] = students.gender;
    dr["ContactNo"] = students.Phone;
    dr["email"] = students.Email;
    dr["CourseEnroll"] = students.CourseEnroll;
    dr["zone"] = students.Zone;
    dr["District"] = students.District;
    dr["Tole"] = students.Tole;
    dr["guardianNo"] = students.guardianNo;
    dr["RegistrationDate"] = DateTime.Today;
    dataSet.Tables["Student"].Rows.Add(dr);
}
}

//this class contains getter and setter for student data
public class StudentInfo
{
    public StudentInfo() { }

    public StudentInfo(string studentString)
    {
        this.ConvertToObject(studentString);
    }

    public string stdId { get; set; }
    public string FirstName { get; set; }
    public string LastName { get; set; }
    public string gender { get; set; }
    public string Zone { get; set; }
    public string District { get; set; }
    public string Tole { get; set; }
    public string Phone { get; set; }
    public string Email { get; set; }
    public string guardianNo { get; set; }
    public string CourseEnroll { get; set; }

    public override string ToString()
    {
        return
        $"{this.stdId}:{this.FirstName}:{this.LastName}:{this.gender}:{this.Zone}:{this.
        .District}:{this.Tole}:{this.Phone}:{this.Email}:{this.guardianNo}:{this.Course
        Enroll}";
    }
    //this method extract data from file by splliting data with certain
symbol
    private void ConvertToObject(string studentString)
    {
        var splitedStrings = studentString.Split(',');
        this.stdId = splitedStrings[0];
        this.FirstName = splitedStrings[1];
        this.LastName = splitedStrings[2];
        this.gender = splitedStrings[3];
        this.Zone = splitedStrings[4];
        this.District = splitedStrings[5];
        this.Tole = splitedStrings[6];
    }
}

```

```

        this.Phone = splitedStrings[7];
        this.Email = splitedStrings[8];
        this.guardianNo = splitedStrings[9];
        this.CourseEnroll = splitedStrings[10];
    }
}
}

```

### View Report

```

namespace StudentRegistration
{
    /// <summary>
    /// Interaction logic for ViewReport.xaml
    /// </summary>
    public partial class ViewReport : Window
    {
        string CurrentPath = System.AppDomain.CurrentDomain.BaseDirectory +
        "\\StudentCWData.xml";

        private int javaNum,
        pythonNum, hardwareNum, databaseNum, SENUM, ADNum; //initialisation of daily student
        number in specific course
        private int WjavaNum, WpythonNum, WhardwareNum, WdatabaseNum, WSENUM,
        WADNum; //initialisation of weekly student number in specific course
        private int TjavaNum, TpythonNum, ThardwareNum, TdatabaseNum, TSENUM,
        TADNum; ///initialisation of total student number in specific course

        private void WeeklyReport_Click(object sender, RoutedEventArgs e)
        {
            graphGrid.Visibility = Visibility.Hidden;
            dailyreportgrid.Visibility = Visibility.Visible;
            reportDataGrid.Visibility = Visibility.Hidden;
            javano.Content = WjavaNum;
            pythonNo.Content = WpythonNum;
            hardwareNo.Content = WhardwareNum;
            databaseNo.Content = WdatabaseNum;
            SENO.Content = WSENUM;
            ADNo.Content = WADNum;
        }

        private DateTime todaysDate;
        public ViewReport()
        {
            InitializeComponent();
            var dataHandler = new datahandler();
            var dataSet = dataHandler.CreateDataSet();
            todaysDate = DateTime.UtcNow.Date;
            if (File.Exists(CurrentPath))
            {
                dataSet.ReadXml(CurrentPath);
                reportDataGrid.ItemsSource = new
                DataView(dataSet.Tables["Student"]);
                reportDataGrid.Visibility = Visibility.Visible;
            }
            DataTable studentTable = dataSet.Tables["student"];

```

```

        javaNum = studentTable.Select("CourseEnroll = 'java' AND
RegistrationDate>='" + todaysDate + "'").Count<DataRow>();
        pythonNum = studentTable.Select("CourseEnroll = 'python' AND
RegistrationDate >='" + todaysDate + "'").Count<DataRow>();
        hardwareNum = studentTable.Select("CourseEnroll = 'hardware' AND
RegistrationDate >='" + todaysDate + "'").Count<DataRow>();
        databaseNum = studentTable.Select("CourseEnroll = 'Database' AND
RegistrationDate >='" + todaysDate + "'").Count<DataRow>();
        SEnum = studentTable.Select("CourseEnroll = 'Software Engineering'
AND RegistrationDate >='" + todaysDate + "'").Count<DataRow>();
        ADNum = studentTable.Select("CourseEnroll = 'Application
Development' AND RegistrationDate >='" + todaysDate + "'").Count<DataRow>();

        TjavaNum = studentTable.Select("CourseEnroll = 'java'
").Count<DataRow>();
        TpythonNum = studentTable.Select("CourseEnroll =
'python'").Count<DataRow>();
        ThardwareNum = studentTable.Select("CourseEnroll = 'hardware'
").Count<DataRow>();
        TdatabaseNum = studentTable.Select("CourseEnroll = 'Database'
").Count<DataRow>();
        TSEnum = studentTable.Select("CourseEnroll = 'Software Engineering'
").Count<DataRow>();
        TADNum = studentTable.Select("CourseEnroll = 'Application
Development' ").Count<DataRow>());

        WjavaNum = studentTable.Select("CourseEnroll = 'java' AND
RegistrationDate>='" + todaysDate.AddDays(-7) + "'").Count<DataRow>();
        WpythonNum = studentTable.Select("CourseEnroll = 'python' AND
RegistrationDate>='" + todaysDate.AddDays(-7) + "'").Count<DataRow>();
        WhardwareNum = studentTable.Select("CourseEnroll = 'hardware' AND
RegistrationDate>='" + todaysDate.AddDays(-7) + "'").Count<DataRow>();
        WdatabaseNum = studentTable.Select("CourseEnroll = 'Database' AND
RegistrationDate>='" + todaysDate.AddDays(-7) + "'").Count<DataRow>();
        WSEnum = studentTable.Select("CourseEnroll = 'Software Engineering'
AND RegistrationDate>='" + todaysDate.AddDays(-7) + "'").Count<DataRow>();
        WADNum = studentTable.Select("CourseEnroll = 'Application
Development' AND RegistrationDate>='" + todaysDate.AddDays(-7) +
'").Count<DataRow>();
    }

    private void sortBy_SelectionChanged(object sender,
SelectionChangedEventArgs e)
    {
        dailyreportgrid.Visibility = Visibility.Collapsed;
        reportDataGrid.Visibility = Visibility.Visible;
        graphGrid.Visibility = Visibility.Hidden;
        weeklyReportGrid.Visibility = Visibility.Hidden;

        var sortby = sortBy.SelectedIndex;
        if (sortby == 0)
        {
            reportDataGrid.Items.SortDescriptions.Clear();
            reportDataGrid.Items.SortDescriptions.Add(new
SortDescription("firstname", ListSortDirection.Ascending));
            reportDataGrid.Items.Refresh();
        }
        if (sortby == 1)
        {
            reportDataGrid.Items.SortDescriptions.Clear();

```



```

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

private void Button_Click(object sender, RoutedEventArgs e)
{
    dailyreportgrid.Visibility = Visibility.Visible;
    reportDataGrid.Visibility = Visibility.Hidden;
    graphGrid.Visibility = Visibility.Hidden;
    weeklyReportGrid.Visibility = Visibility.Hidden;

    javano.Content = javaNum;
    pythonNo.Content = pythonNum;
    hardwareNo.Content = hardwareNum;
    databaseNo.Content = databaseNum;
    SENO.Content = SENUM;
    ADNo.Content = ADNum;
}
//this method is used to generate graoh
private void LoadColumnChartData()
{
    ((ColumnSeries)mcChart.Series[0]).ItemsSource =
        new KeyValuePair<string, int>[]{
new KeyValuePair<string, int>("Python", TpythonNum),
new KeyValuePair<string, int>("JAVA", TjavaNum),
new KeyValuePair<string, int>("Hardware", ThardwareNum),
new KeyValuePair<string, int>("Database", TdatabaseNum),
new KeyValuePair<string, int>("Software Engineering", TSEnum),
new KeyValuePair<string, int>("Application Development", TADNum) };
}

private void Button_Click_1(object sender, RoutedEventArgs e)
{
    graphGrid.Visibility=Visibility.Visible;
    dailyreportgrid.Visibility = Visibility.Collapsed;
    reportDataGrid.Visibility = Visibility.Hidden;
    weeklyReportGrid.Visibility = Visibility.Hidden;
    LoadColumnChartData();
}

private void GraphicalReport_Click(object sender, RoutedEventArgs e)
{
    graphGrid.Visibility = Visibility.Visible;
    dailyreportgrid.Visibility = Visibility.Collapsed;
    reportDataGrid.Visibility = Visibility.Hidden;
    weeklyReportGrid.Visibility = Visibility.Hidden;
    LoadColumnChartData();
}
}
}

```