

Informatics College Pokhara



informatics
college pokhara

Application Development

CS6004NI

Course Work 1

Submitted By: Amit Gurung
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 | Data entry is proper with appropriate validation and use of proper datatyps and contains CRUD methods |
| Data Validation | missing most of the validation |
| Enrollment Report & weekly report in tabular format | very poorly executed reports and data not shown accurately |
| Course wise enrollment report & Chart display | any one component is missing or inappropriate data is shown |
| Algorithm used for sorting & proper sorting of data | Default sorting provided by .net is used |
| B. Documentation | |
| User Manual for running the application | User Manual is good. Contains all varieties of forms. |

| | |
|--|--|
| Application architecture & description of the classes ad methods sued | average work with very limited explanation of the classes and methods used |
| Flow chart, algorithms and data sctructures used | average work with very limited explanation and missing diagramatic representation. |
| Reflective essay | Average work with un clear learnings, experience or findings. |

C. Programming Style

| | |
|--|---|
| Clarity of code,Popper Naming convention & comments | Code is poorly written and lacks comments |
| System Usability | System can't be used and have issues |

| | | |
|-----------------------|----------|----------|
| Overall Grade: | C | C |
|-----------------------|----------|----------|

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: Amit Gurung

LondonMet ID: 17030692

Group: L3C2

Date: January 10, 2020

Submitted To:

Mr. Ishwor Sapkota

Application Development

Abstract

This report describes about the Student Information System that was developed in WPF (.NET Framework). It is a system that is used to keep track of student that were enrolled in the institute. This system will help institutes with old school record keeping to change into a digital format. This will help the work flow to be much faster and the to keep the data safe from physical harm. All the student's details will be saved in the local hard drive and can be accessed freely. All the student's details can also be arranged according to the need in this system. And also, the admin can keep track of number of students enrolled in every course by looking in the Data Table as well as reviewing the Pie chart.

Table of Contents

| | |
|---------------------------------------|-----------|
| Introduction | 1 |
| Current Scenario | 1 |
| Proposed System | 1 |
| User Manual | 2 |
| Journals Articles | 12 |
| System Architecture | 13 |
| Architecture Diagram | 13 |
| Class Diagram | 13 |
| Individual Class Diagram | 14 |
| Flowcharts for Reports | 17 |
| • Enrol Students | 17 |
| • Import CSV data..... | 18 |
| Algorithms | 19 |
| Quick Sort Algorithm | 19 |
| Reflection | 20 |
| Conclusion | 21 |
| References | 22 |
| Appendix | 23 |
| Login.xaml.cs | 23 |
| MainWindow.xaml.cs..... | 23 |
| StudentDetails.xaml.cs | 26 |
| Chart.xaml.cs | 29 |
| Handler.cs..... | 30 |

List of Figures

| | |
|---|----|
| Figure 1: Login Form..... | 2 |
| Figure 2: Login Form filled..... | 2 |
| Figure 3: Username/Password not matched. | 3 |
| Figure 4: Add Students Details Form | 3 |
| Figure 5: Name field validation..... | 4 |
| Figure 6: Address field validation | 4 |
| Figure 7: Contact field validation | 5 |
| Figure 8: Email field validation | 5 |
| Figure 9: Course not selected validation | 6 |
| Figure 10: Date not selected validation | 6 |
| Figure 11: Add Student Details Form filled..... | 7 |
| Figure 12: Details Saved message..... | 7 |
| Figure 13: Data Table of Student Details | 8 |
| Figure 14: Students Details sorted by Date..... | 8 |
| Figure 15: Students Details sorted by Name..... | 9 |
| Figure 16: Data Table of Weekly Report | 9 |
| Figure 17: Pie chart of Weekly Report..... | 10 |
| Figure 18: View CSV browse window..... | 10 |
| Figure 19: Data Table showing CSV data | 11 |
| Figure 20: Import CSV file..... | 11 |
| Figure 21: Architecture Diagram | 13 |
| Figure 22: Class Diagram..... | 13 |
| Figure 23: Flowchart of Enrol Students | 17 |
| Figure 24: Import CSV flowchart | 18 |

List of Tables

| | |
|--|----|
| <i>Table 1: Login Individual Class Diagram</i> | 14 |
| <i>Table 2: MainWindow Individual Class Diagram</i> | 15 |
| <i>Table 3: StudentList Individual Class Diagram</i> | 15 |
| <i>Table 4: Chart Individual Class Diagram</i> | 16 |
| <i>Table 5: Handler Individual Class Diagram</i> | 16 |

Introduction

The system developed in this coursework is a Student Information System developed in Visual Studio 2019 using C#. It is a desktop application which can be used by any schools or institutes to keep track of their enrolled students. In this application we can add the details of newly enrolled students which can be saved in a “.xml” file. Likewise, the file can be then imported to show the student details according to the registration date or name. In addition to that, the final details of all the students can also be seen in a grid form. Number of students that are enrolled in a specific Program till this date can also be seen in a data grid or a chart. Likewise, bulk data in CSV form can also be imported and appended into the main file of student data.

Current Scenario

There are many schools and institutions in Nepal which till this date use an old fashion way of keeping their data in a written format. While keeping data in written format data can be easily lost or physically damaged. Retrieving data is also a very time-consuming process and arranging all the collected data in certain format is very difficult task. Furthermore, keeping track of collected data is hard. The storage of the physical files is also a problem.

Proposed System

To solve the above problems the same thing can be done in a digital format. The collection of data can be done in a computer and saved in the system or cloud storage for safety. Retrieving data and calculating many aspects of the data can also be done by a simple click of a button. The work will be extremely fast as well as the data can be safe from any physical harm. Keeping track of data and finding a specific data can be done very easily. The data can also be kept in cloud platform for safety from theft and it also takes very less storage space.

User Manual

To help the users to learn to use this program, detailed steps are shown below with screenshots of the process.

1. When User first runs the program, this window will be shown. This is the Login window where correct username and password should be entered to go to next screen.

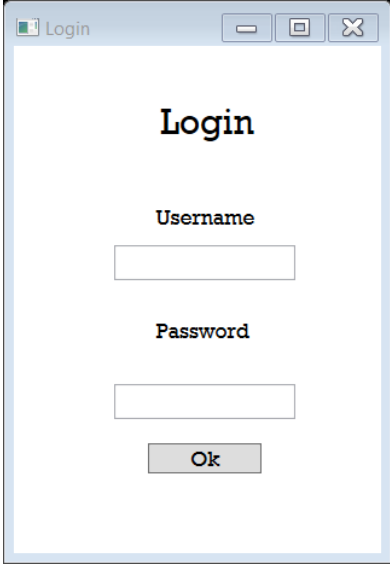
A screenshot of a Windows-style window titled "Login". The window has a standard title bar with minimize, maximize, and close buttons. The main content area is white and contains the word "Login" in a large, bold, black serif font. Below the title, there are two labels: "Username" and "Password", both in a black serif font. Under the "Username" label is a single-line text input field. Under the "Password" label is a single-line text input field. At the bottom center of the window is a button labeled "Ok" in a black serif font.

Figure 1: Login Form

2. Current username and password for the program is "admin."

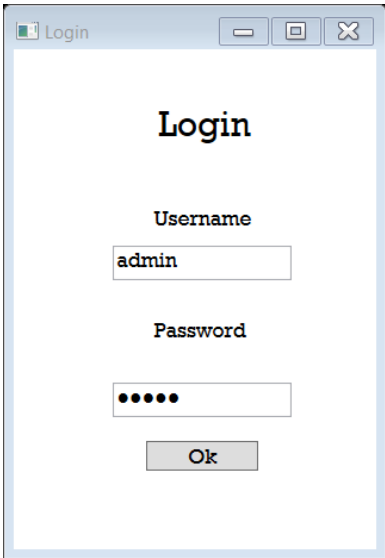
A screenshot of the same "Login" window. The "Username" input field now contains the text "admin". The "Password" input field contains five black dots, indicating a masked password. The "Ok" button remains at the bottom.

Figure 2: Login Form filled

3. If wrong username or password is entered then login will not be successful and the window given below will appear.

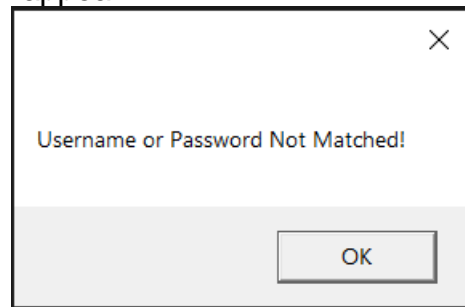


Figure 3: Username/Password not matched.

4. After the login is successful the “Add Student Details” window will open which is the window where new student’s data can be added.

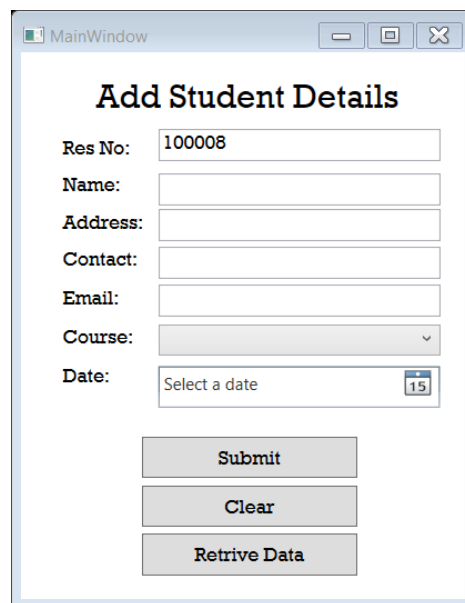
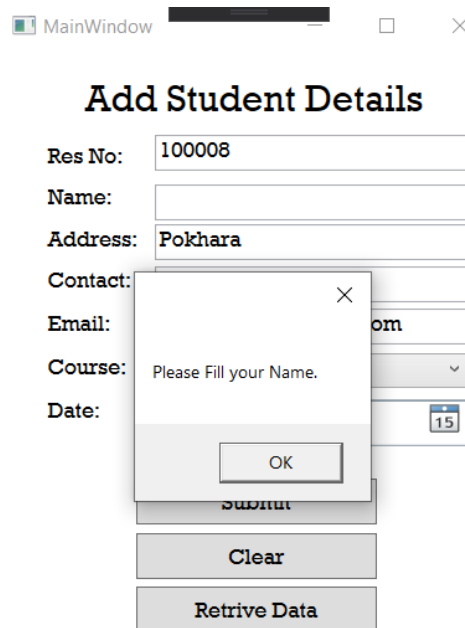


Figure 4: Add Students Details Form

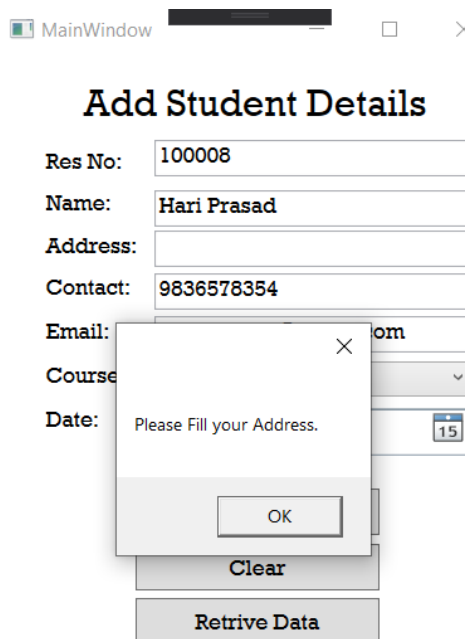
5. Data should be filled in this window if we want to add new student.
- Res No: This is the registration number of new students which is automatically generated.
 - Name: Full Name of student.
 - Address: Address of student.
 - Contact: Phone number of the student.
 - Email: Email address of student.
 - Course: The available courses are shown in drop down menu and the required course should be selected.
 - Date: The date of registration should be selected.

6. If any of the text box are empty or not filled with correct data then a message box will appear as warning and data will not be saved.



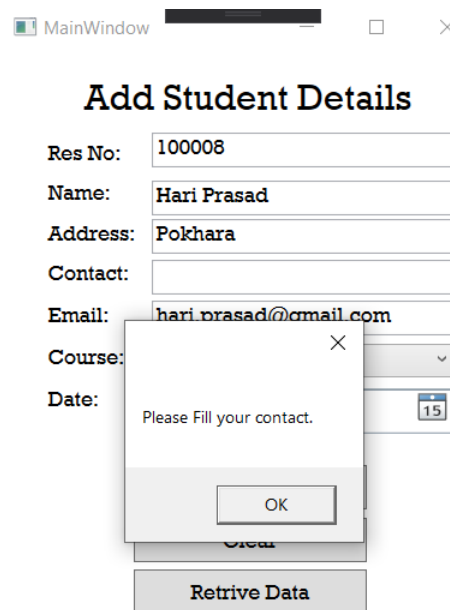
The screenshot shows a Windows application window titled 'MainWindow'. Inside, there is a form titled 'Add Student Details'. The form contains several input fields: 'Res No:' with the value '100008', 'Name:' (empty), 'Address:' with the value 'Pokhara', 'Contact:', 'Email:', 'Course:', and 'Date:' with a calendar icon showing '15'. A modal dialog box is displayed over the form, containing the text 'Please Fill your Name.' and an 'OK' button. Below the form, there are three buttons: 'Submit', 'Clear', and 'Retrive Data'.

Figure 5: Name field validation



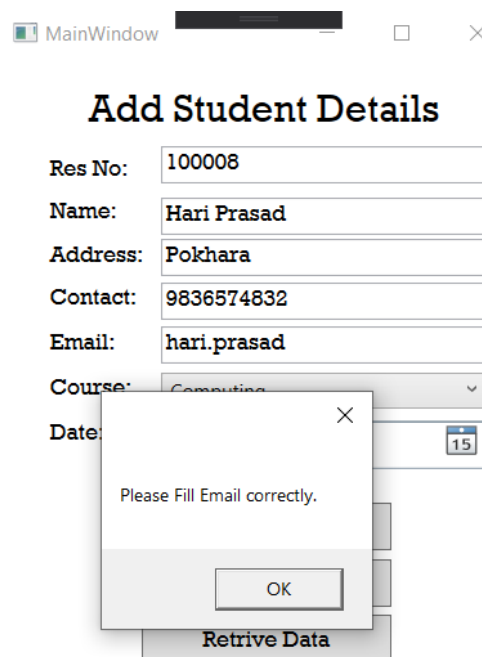
The screenshot shows the same 'Add Student Details' form. In this instance, the 'Name' field is filled with 'Hari Prasad' and the 'Address' field is empty. The modal dialog box now displays the text 'Please Fill your Address.' with an 'OK' button. The 'Res No:' field still contains '100008', and the 'Date' field still shows '15'. The 'Submit', 'Clear', and 'Retrive Data' buttons are still visible at the bottom.

Figure 6: Address field validation



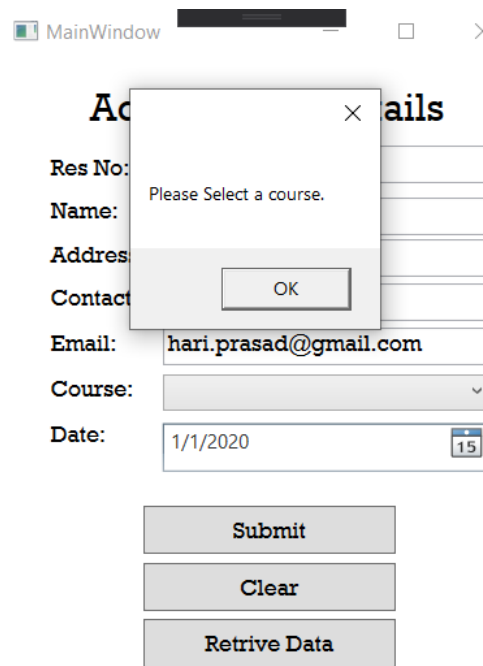
The screenshot shows a window titled 'MainWindow' with a form titled 'Add Student Details'. The form contains the following fields: 'Res No:' with value '100008', 'Name:' with value 'Hari Prasad', 'Address:' with value 'Pokhara', 'Contact:' (empty), 'Email:' with value 'hari.prasad@gmail.com', 'Course:' (dropdown menu), and 'Date:' (calendar icon showing '15'). A modal dialog box is displayed over the 'Contact' field with the message 'Please Fill your contact.' and an 'OK' button. Below the form are two buttons: 'Clear' and 'Retrive Data'.

Figure 7: Contact field validation



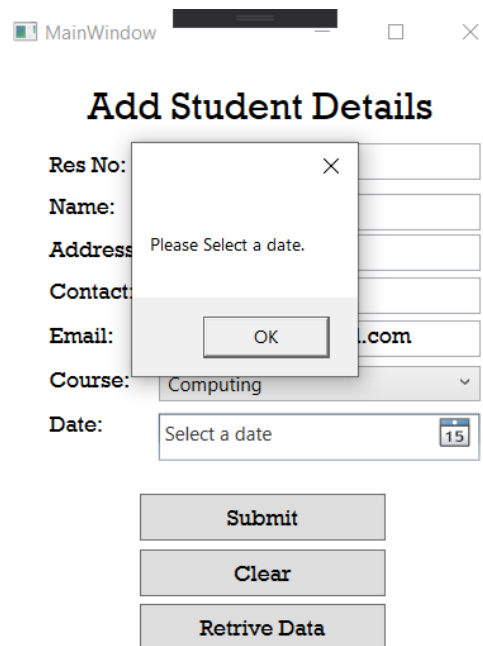
The screenshot shows the same 'Add Student Details' form as Figure 7, but with the 'Contact' field filled with '9836574832'. The 'Email' field now contains 'hari.prasad' (without the domain). A modal dialog box is displayed over the 'Email' field with the message 'Please Fill Email correctly.' and an 'OK' button. The 'Retrive Data' button is visible at the bottom.

Figure 8: Email field validation



The screenshot shows a Windows application window titled 'MainWindow'. Inside, there is a form titled 'Add Student Details'. The form contains the following fields: 'Res No:', 'Name:', 'Address:', 'Contact:', 'Email:' (with the value 'hari.prasad@gmail.com'), 'Course:' (a dropdown menu), and 'Date:' (a date picker showing '1/1/2020'). Below the form are three buttons: 'Submit', 'Clear', and 'Retrive Data'. A modal dialog box is displayed over the form, containing the text 'Please Select a course.' and an 'OK' button. The 'Course:' dropdown menu in the background is currently empty.

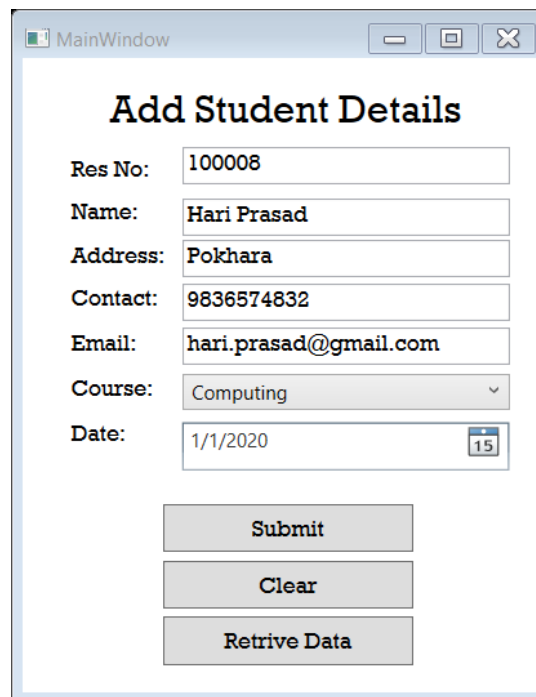
Figure 9: Course not selected validation



The screenshot shows the same 'Add Student Details' form. In this instance, the 'Course:' dropdown menu is set to 'Computing'. The 'Date:' field is a date picker showing '15' of the month. A modal dialog box is displayed over the form, containing the text 'Please Select a date.' and an 'OK' button. The 'Date:' field in the background is currently empty.

Figure 10: Date not selected validation

7. Fill all the text fields with correct data.



MainWindow

Add Student Details

Res No: 100008

Name: Hari Prasad

Address: Pokhara

Contact: 9836574832

Email: hari.prasad@gmail.com

Course: Computing

Date: 1/1/2020

Submit

Clear

Retrive Data

Figure 11: Add Student Details Form filled

8. After the data is filled click the “Submit” button and the data will be saved showing this window.

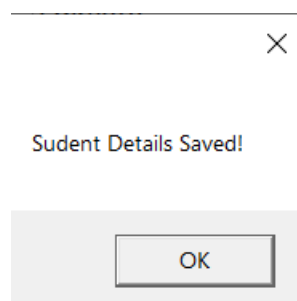
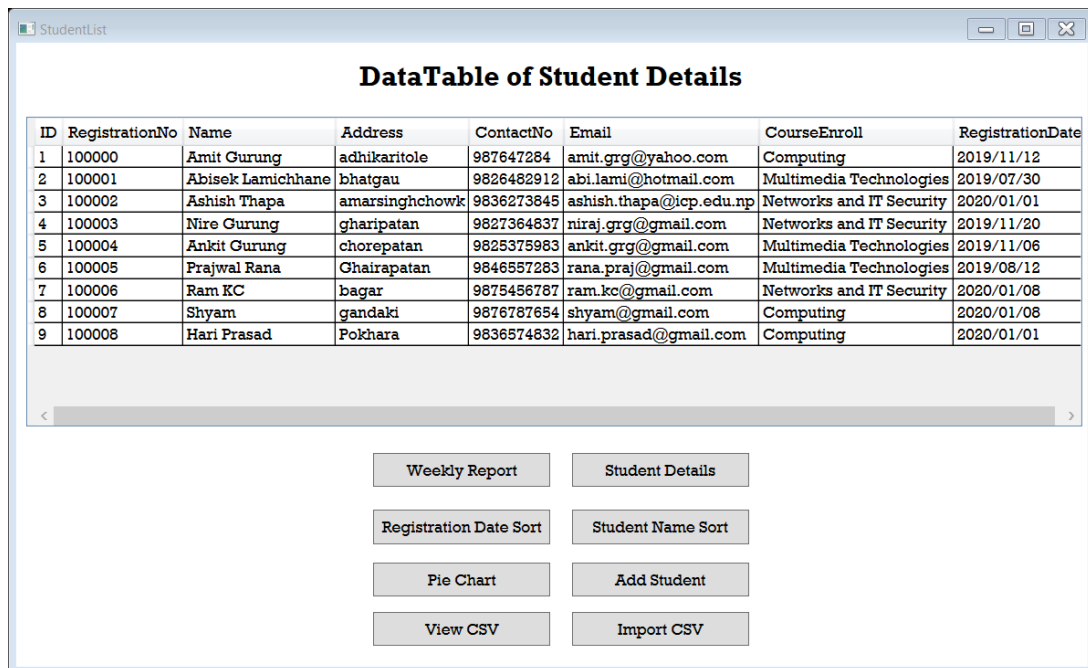


Figure 12: Details Saved message

9. Clicking the “Student Details” button in the “Add Student Details” window will open the window which is given below. This window has the DataGrid which shows the details of all the students currently enrolled.

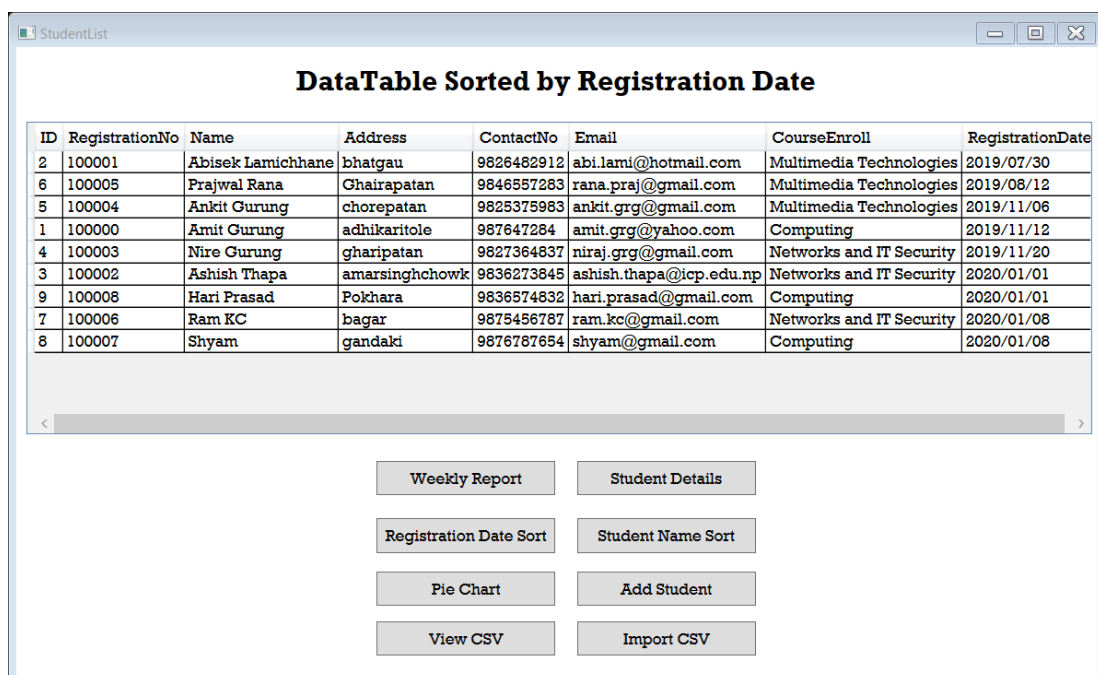


The screenshot shows a window titled "StudentList" with a title bar containing minimize, maximize, and close buttons. The main content area is titled "DataTable of Student Details" and contains a DataGrid with 8 columns: ID, RegistrationNo, Name, Address, ContactNo, Email, CourseEnroll, and RegistrationDate. The DataGrid lists 9 students. Below the DataGrid is a horizontal scrollbar. At the bottom of the window, there are eight buttons arranged in four rows and two columns: Weekly Report, Student Details, Registration Date Sort, Student Name Sort, Pie Chart, Add Student, View CSV, and Import CSV.

| ID | RegistrationNo | Name | Address | ContactNo | Email | CourseEnroll | RegistrationDate |
|----|----------------|-------------------|----------------|------------|-------------------------|--------------------------|------------------|
| 1 | 100000 | Amit Gurung | adhikaritole | 987647284 | amit.grg@yahoo.com | Computing | 2019/11/12 |
| 2 | 100001 | Abisek Lamichhane | bhatgau | 9826482912 | abi.lami@hotmail.com | Multimedia Technologies | 2019/07/30 |
| 3 | 100002 | Ashish Thapa | amarsinghchowk | 9836273845 | ashish.thapa@icp.edu.np | Networks and IT Security | 2020/01/01 |
| 4 | 100003 | Nire Gurung | gharipatan | 9827364837 | niraj.grg@gmail.com | Networks and IT Security | 2019/11/20 |
| 5 | 100004 | Ankit Gurung | chorepatan | 9825375983 | ankit.grg@gmail.com | Multimedia Technologies | 2019/11/06 |
| 6 | 100005 | Prajwal Rana | Ghairapatan | 9846557283 | rana.praj@gmail.com | Multimedia Technologies | 2019/08/12 |
| 7 | 100006 | Ram KC | bagar | 9875456787 | ram.kc@gmail.com | Networks and IT Security | 2020/01/08 |
| 8 | 100007 | Shyam | gandaki | 9876787654 | shyam@gmail.com | Computing | 2020/01/08 |
| 9 | 100008 | Hari Prasad | Pokhara | 9836574832 | hari.prasad@gmail.com | Computing | 2020/01/01 |

Figure 13: Data Table of Student Details

10. Pressing “Registration Date Sort” button will arrange the values in the DataGrid with ascending order with respect to the Registration Date.

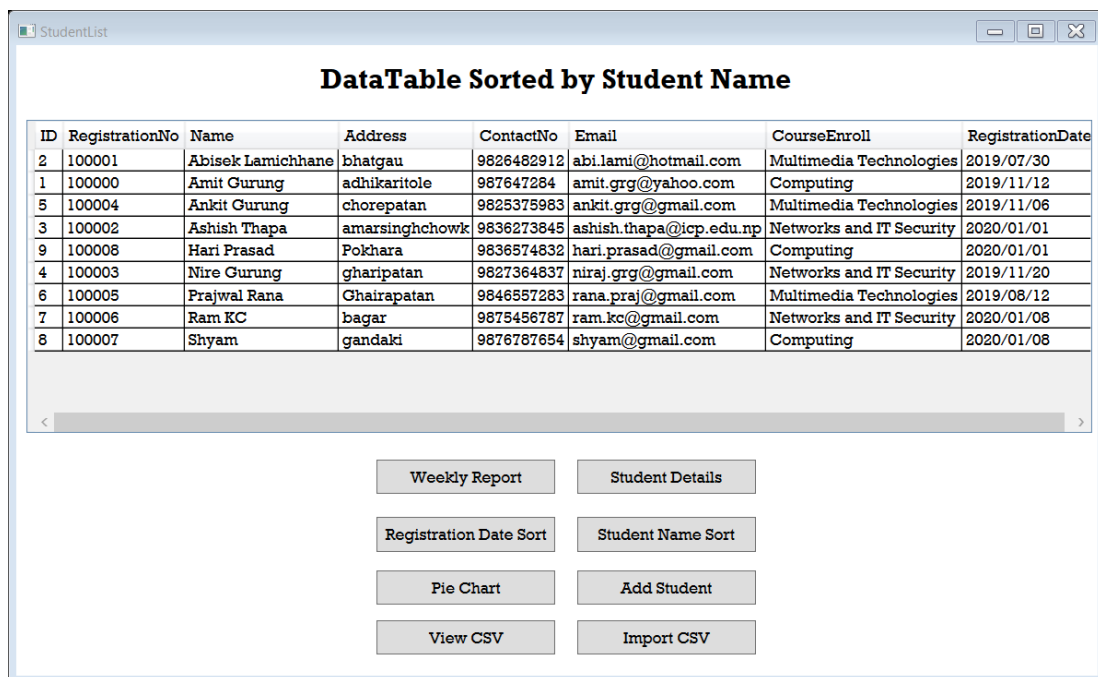


The screenshot shows the same "StudentList" window, but the DataGrid is titled "DataTable Sorted by Registration Date". The data is sorted by RegistrationDate in ascending order. The buttons at the bottom remain the same as in Figure 13.

| ID | RegistrationNo | Name | Address | ContactNo | Email | CourseEnroll | RegistrationDate |
|----|----------------|-------------------|----------------|------------|-------------------------|--------------------------|------------------|
| 2 | 100001 | Abisek Lamichhane | bhatgau | 9826482912 | abi.lami@hotmail.com | Multimedia Technologies | 2019/07/30 |
| 6 | 100005 | Prajwal Rana | Ghairapatan | 9846557283 | rana.praj@gmail.com | Multimedia Technologies | 2019/08/12 |
| 5 | 100004 | Ankit Gurung | chorepatan | 9825375983 | ankit.grg@gmail.com | Multimedia Technologies | 2019/11/06 |
| 1 | 100000 | Amit Gurung | adhikaritole | 987647284 | amit.grg@yahoo.com | Computing | 2019/11/12 |
| 4 | 100003 | Nire Gurung | gharipatan | 9827364837 | niraj.grg@gmail.com | Networks and IT Security | 2019/11/20 |
| 3 | 100002 | Ashish Thapa | amarsinghchowk | 9836273845 | ashish.thapa@icp.edu.np | Networks and IT Security | 2020/01/01 |
| 9 | 100008 | Hari Prasad | Pokhara | 9836574832 | hari.prasad@gmail.com | Computing | 2020/01/01 |
| 7 | 100006 | Ram KC | bagar | 9875456787 | ram.kc@gmail.com | Networks and IT Security | 2020/01/08 |
| 8 | 100007 | Shyam | gandaki | 9876787654 | shyam@gmail.com | Computing | 2020/01/08 |

Figure 14: Students Details sorted by Date

11. Pressing “Student Name Sort” button will arrange the values in the DataGrid with ascending order with respect to the Name of students.

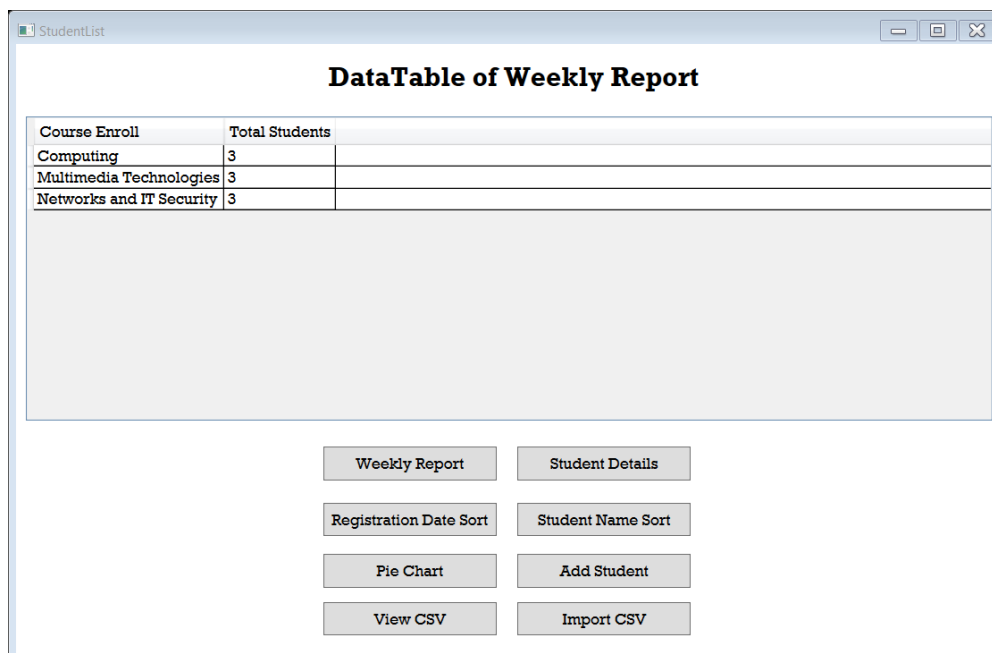


DataTable Sorted by Student Name

| ID | RegistrationNo | Name | Address | ContactNo | Email | CourseEnroll | RegistrationDate |
|----|----------------|-------------------|----------------|------------|-------------------------|--------------------------|------------------|
| 2 | 100001 | Abisek Lamichhane | bhatgau | 9826482912 | abi.lami@hotmail.com | Multimedia Technologies | 2019/07/30 |
| 1 | 100000 | Amit Gurung | adhikaritole | 987647284 | amit.grg@yahoo.com | Computing | 2019/11/12 |
| 5 | 100004 | Ankit Gurung | chorepatan | 9825375983 | ankit.grg@gmail.com | Multimedia Technologies | 2019/11/06 |
| 3 | 100002 | Ashish Thapa | amarsinghchowk | 9836273845 | ashish.thapa@icp.edu.np | Networks and IT Security | 2020/01/01 |
| 9 | 100008 | Hari Prasad | Pokhara | 9836574832 | hari.prasad@gmail.com | Computing | 2020/01/01 |
| 4 | 100003 | Nire Gurung | gharipatan | 9827364837 | niraj.grg@gmail.com | Networks and IT Security | 2019/11/20 |
| 6 | 100005 | Prajwal Rana | Ghairapatan | 9846557283 | rana.praj@gmail.com | Multimedia Technologies | 2019/08/12 |
| 7 | 100006 | Ram KC | bagar | 9875456787 | ram.kc@gmail.com | Networks and IT Security | 2020/01/08 |
| 8 | 100007 | Shyam | gandaki | 9876787654 | shyam@gmail.com | Computing | 2020/01/08 |

Figure 15: Students Details sorted by Name

12. Pressing “Weekly Report” button will show the Courses and the total number of students currently enrolled in that courses.



DataTable of Weekly Report

| Course Enroll | Total Students |
|--------------------------|----------------|
| Computing | 3 |
| Multimedia Technologies | 3 |
| Networks and IT Security | 3 |

Figure 16: Data Table of Weekly Report

13. “Pie chart” button shows the Pie chart of the Weekly Report. In above figure we can see there are 3 students in all the courses so the pie chart shown below is created.

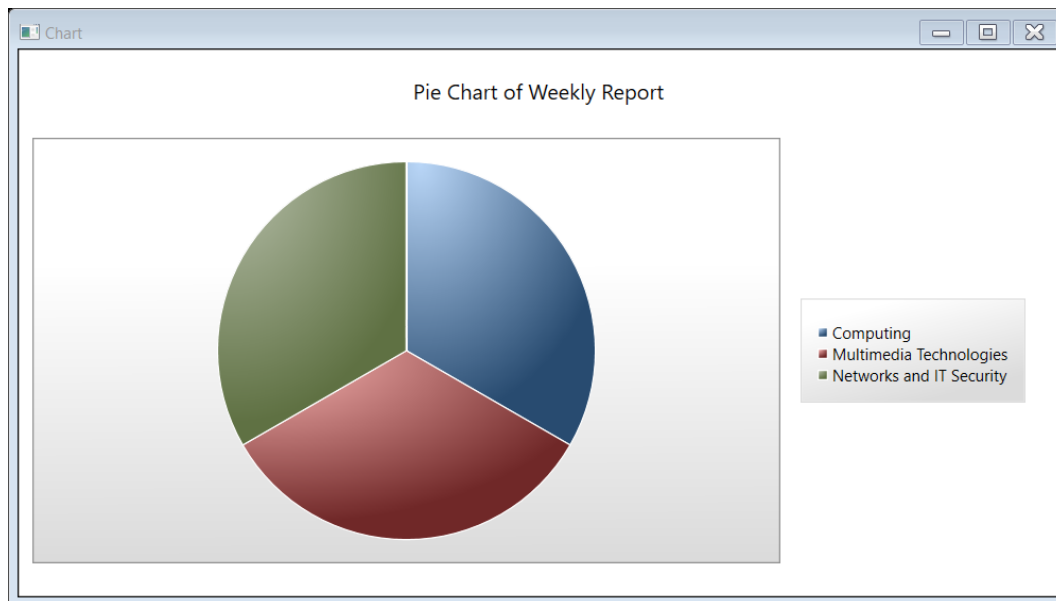


Figure 17: Pie chart of Weekly Report

14. Pressing “View CSV” button will open a browse window where we should select the “.csv” file that we want to view into the DataGrid inside our project.

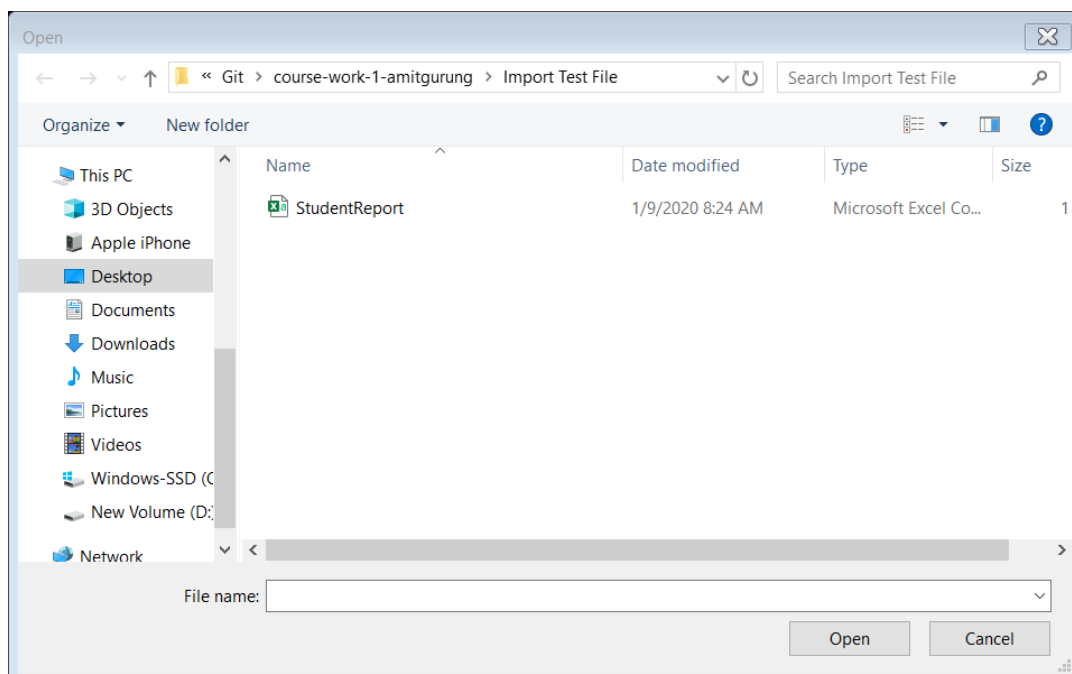
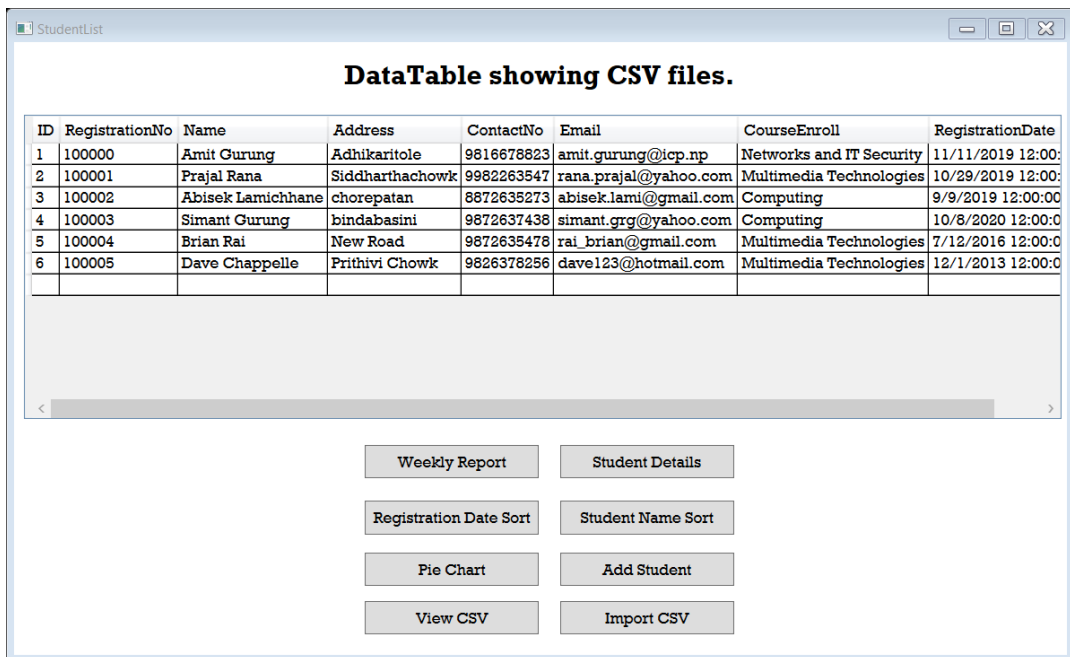


Figure 18: View CSV browse window.

15. Opening the “.csv” file will view the values in the csv file in our DataGrid.

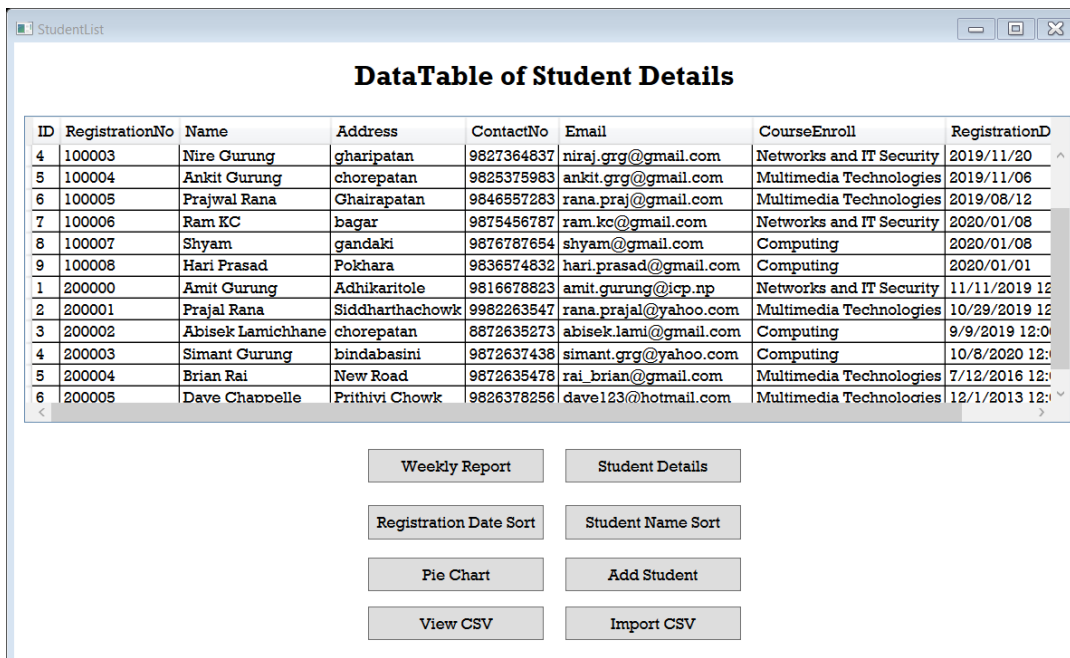


DataTable showing CSV files.

| ID | RegistrationNo | Name | Address | ContactNo | Email | CourseEnroll | RegistrationDate |
|----|----------------|-------------------|-----------------|------------|-----------------------|--------------------------|---------------------|
| 1 | 100000 | Amit Gurung | Adhikaritole | 9816678823 | amit.gurung@icp.np | Networks and IT Security | 11/11/2019 12:00:00 |
| 2 | 100001 | Prajai Rana | Siddharthachowk | 9982263547 | rana.prajai@yahoo.com | Multimedia Technologies | 10/29/2019 12:00:00 |
| 3 | 100002 | Abisek Lamichhane | chorepatan | 8872635273 | abisek.lami@gmail.com | Computing | 9/9/2019 12:00:00 |
| 4 | 100003 | Simant Gurung | bindabasini | 9872637438 | simant.grg@yahoo.com | Computing | 10/8/2020 12:00:00 |
| 5 | 100004 | Brian Rai | New Road | 9872635478 | rai_brian@gmail.com | Multimedia Technologies | 7/12/2016 12:00:00 |
| 6 | 100005 | Dave Chappelle | Prithivi Chowk | 9826378256 | dave123@hotmail.com | Multimedia Technologies | 12/1/2013 12:00:00 |

Figure 19: Data Table showing CSV data

16. If we press the “Import CSV” button then the same browse window will open and the chosen “.csv” file’s data will be added to the current data. You can see the old data with the added data of CSV file in the picture below.



DataTable of Student Details

| ID | RegistrationNo | Name | Address | ContactNo | Email | CourseEnroll | RegistrationDate |
|----|----------------|-------------------|-----------------|------------|-----------------------|--------------------------|---------------------|
| 4 | 100003 | Nire Gurung | gharipatan | 9827364837 | niraj.grg@gmail.com | Networks and IT Security | 2019/11/20 |
| 5 | 100004 | Ankit Gurung | chorepatan | 9825375983 | ankit.grg@gmail.com | Multimedia Technologies | 2019/11/06 |
| 6 | 100005 | Prajwal Rana | Ghairapatan | 9846557283 | rana.praj@gmail.com | Multimedia Technologies | 2019/08/12 |
| 7 | 100006 | Ram KC | bagar | 9875456787 | ram.kc@gmail.com | Networks and IT Security | 2020/01/08 |
| 8 | 100007 | Shyam | gandaki | 9876787654 | shyam@gmail.com | Computing | 2020/01/08 |
| 9 | 100008 | Hari Prasad | Pokhara | 9836574832 | hari.prasad@gmail.com | Computing | 2020/01/01 |
| 1 | 200000 | Amit Gurung | Adhikaritole | 9816678823 | amit.gurung@icp.np | Networks and IT Security | 11/11/2019 12:00:00 |
| 2 | 200001 | Prajai Rana | Siddharthachowk | 9982263547 | rana.prajai@yahoo.com | Multimedia Technologies | 10/29/2019 12:00:00 |
| 3 | 200002 | Abisek Lamichhane | chorepatan | 8872635273 | abisek.lami@gmail.com | Computing | 9/9/2019 12:00:00 |
| 4 | 200003 | Simant Gurung | bindabasini | 9872637438 | simant.grg@yahoo.com | Computing | 10/8/2020 12:00:00 |
| 5 | 200004 | Brian Rai | New Road | 9872635478 | rai_brian@gmail.com | Multimedia Technologies | 7/12/2016 12:00:00 |
| 6 | 200005 | Dave Chappelle | Prithivi Chowk | 9826378256 | dave123@hotmail.com | Multimedia Technologies | 12/1/2013 12:00:00 |

Figure 20: Import CSV file

17. To see the original student details table, press the “Student Details” button.

18. To go back to adding new students’ details press the “Add Student” button.

Journals Articles

1. Implementation on Curriculum Management System based on .NET
(Ma, 2016)

This article gives us the information about creating a course management system in .NET Framework. Managing courses are a important part of a school. This article creates a system in which courses can be added, deleted, updated and viewed. The current system that is to be developed in this course work is of similar aspects. This article helps us understand the inner workings of the system.

2. WEB BASED SCHOOL ADMINISTRATION SYSTEM
(Amingad, et al., 2017)

This article gives us a detailed description on how the writer has created a school administration system. This system manages the records of the students and faculties more easily. The data is kept up-to-date which makes the retrieval of data easily. The core functions of this project and our coursework matches, due to which reviewing this article helped me a lot while developing my software.

3. Centralized School Management System for Government Schools in Ethiopia using Distributed Database
(Amare, et al., 2018)

This is an article which gives us the information about the School Management System that was developed to digitize the government schools in Ethiopia. This program reduces the work for the workers in schools to manage data and also manage the whole school. It is for both schools and the parents of the students. Communication between them can be done easily through this software. The process of making a digitized system for school is similar objectives that we have. This article helps us understand the problems we ma face and how to solve them.

System Architecture

Architecture Diagram

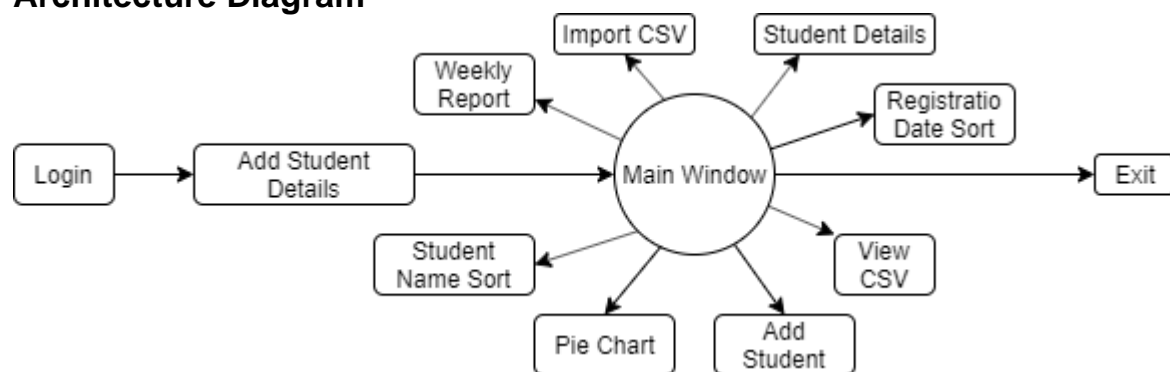


Figure 21: Architecture Diagram

Class Diagram

This is the class diagram of the developed Student Information System. The diagram was developed in Visual Studio 2019.

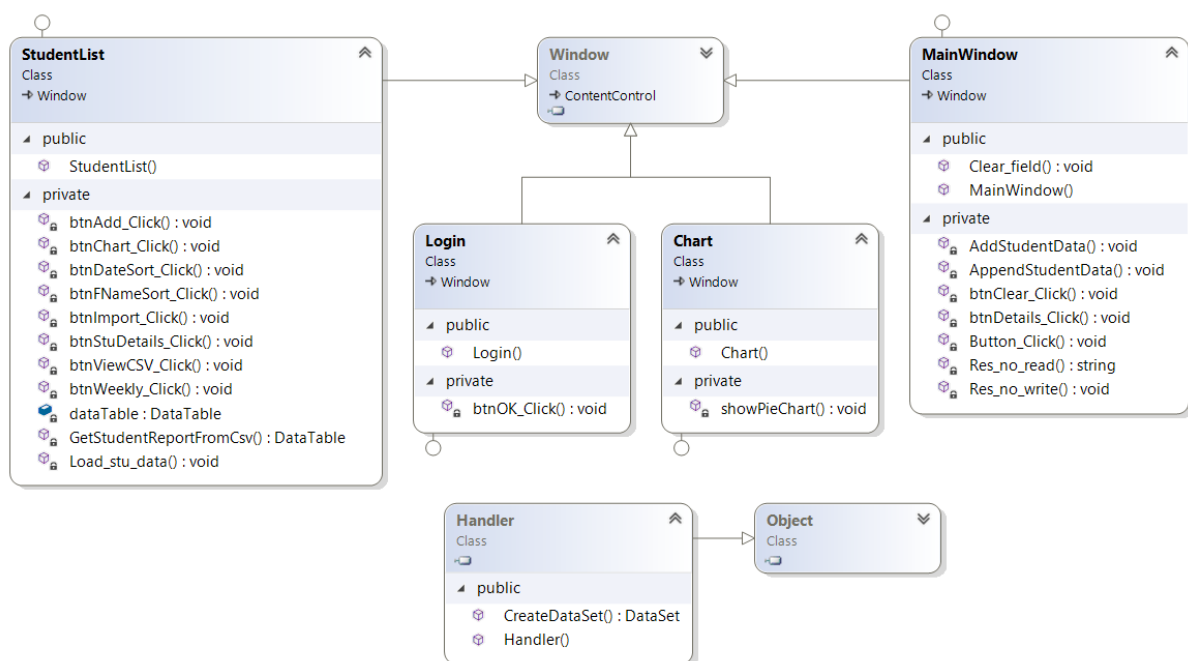


Figure 22: Class Diagram

Individual Class Diagram

1. Login

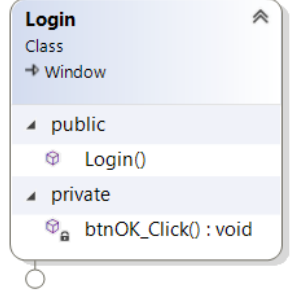
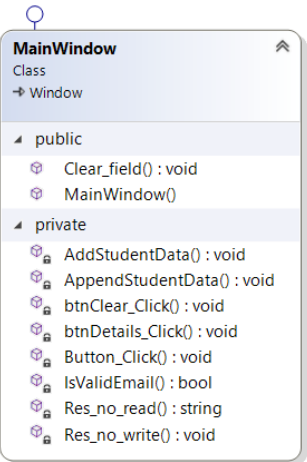
| Methods | Description | Figure |
|---------------|--|--|
| Login() | Constructor method. |  <pre> classDiagram class Login { +public Login() -private btnOK_Click() void } Login -- > Window </pre> |
| btnOK_Click() | Button click method which validates the username and password and logs in the program. | |

Table 1: Login Individual Class Diagram

2. MainWindow

| Methods | Description | Figure |
|---------------------|--|---|
| MainWindow() | Constructor method. |  <pre> classDiagram class MainWindow { +public Clear_field() : void +public MainWindow() -private AddStudentData() : void -private AppendStudentData() : void -private btnClear_Click() : void -private btnDetails_Click() : void -private Button_Click() : void -private IsValidEmail() : bool -private Res_no_read() : string -private Res_no_write() : void } MainWindow -- > Window </pre> |
| AddStudentData() | Add individual student data. | |
| AppendStudentData() | Add all student data in a file. | |
| Clear_Field() | Clear the text fields. | |
| btnCear_Click() | Button click calls the Clear_Field() method. | |
| btnDetails_Click() | Opens StudentList class. | |
| Button_Click() | Validates the data entered in form and Calls AddStudentData() and AppendStudentData() class if all the data are correct. | |
| IsValidEmail() | This method checks if the email address entered is in correct format | |

| | | |
|----------------|---|--|
| Res_no_write() | Stores the current resistration number in a file. | |
| Res_no_read() | Reads the resgistration count stored file. | |

Table 2: MainWindow Individual Class Diagram

3. StudentList

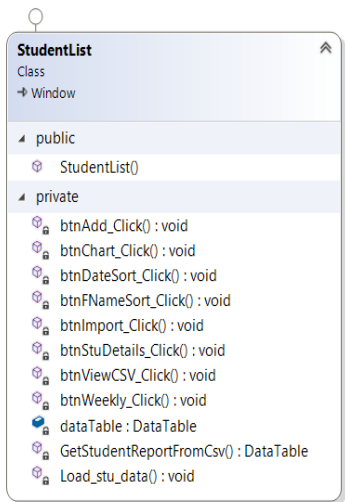
| Methods | Description | Figure |
|-----------------------|---|--|
| StudentList() | Constructor of the class. |  |
| btnAdd_Click() | Opens "MainWindow". | |
| btnDateSort_Click() | Sorts the student data in ascending order according to registration date and shows in datagrid. | |
| btnFNameSort_Click() | Sorts the student data in ascending order according to name and shows in datagrid. | |
| btnStuDetails_Click() | Shows all the student's details in data grid. | |
| btnViewCSV_Click() | Open CSV file and view the data inside it. | |
| btnWeekly_Click() | Shows the number of students enrolled in each course. | |
| btnChart_Click() | Opens "Chart". | |
| Load_Stu_data() | Shows student details in grid. | |
| | | |
| | | |
| | | |

Table 3: StudentList Individual Class Diagram

4. Chart

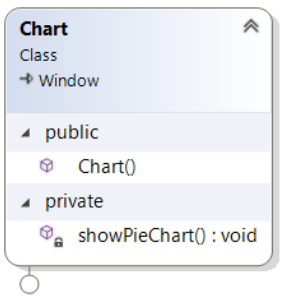
| Methods | Description | Figure |
|----------------|--|--|
| Chart() | Constructor of class. |  <pre> classDiagram class Chart { +Chart() -showPieChart() void } Chart -- > Window </pre> <p>The figure shows a UML class diagram for the Chart class. It is a subclass of Window. The class has a public constructor Chart() and a private method showPieChart() : void.</p> |
| showPieChart() | Shows pie chart of the students enrolled in each course. | |

Table 4: Chart Individual Class Diagram

5. Handler

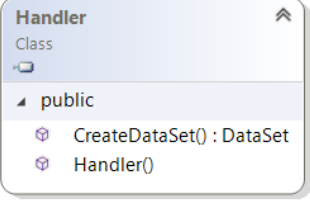
| Methods | Description | Figure |
|-----------------|--|--|
| Handler() | Constructor of the class. |  <pre> classDiagram class Handler { +CreateDataSet() DataSet +Handler() } </pre> <p>The figure shows a UML class diagram for the Handler class. The class has a public method CreateDataSet() : DataSet and a public constructor Handler().</p> |
| CreateDataSet() | Calls the method that creates dataset for individual student details and student report. | |

Table 5: Handler Individual Class Diagram

Flowcharts for Reports

- Enrol Students

This the flowchart of the class MainWindow in which new students fill the form enrol in a course and the details entered are also saved in the system.

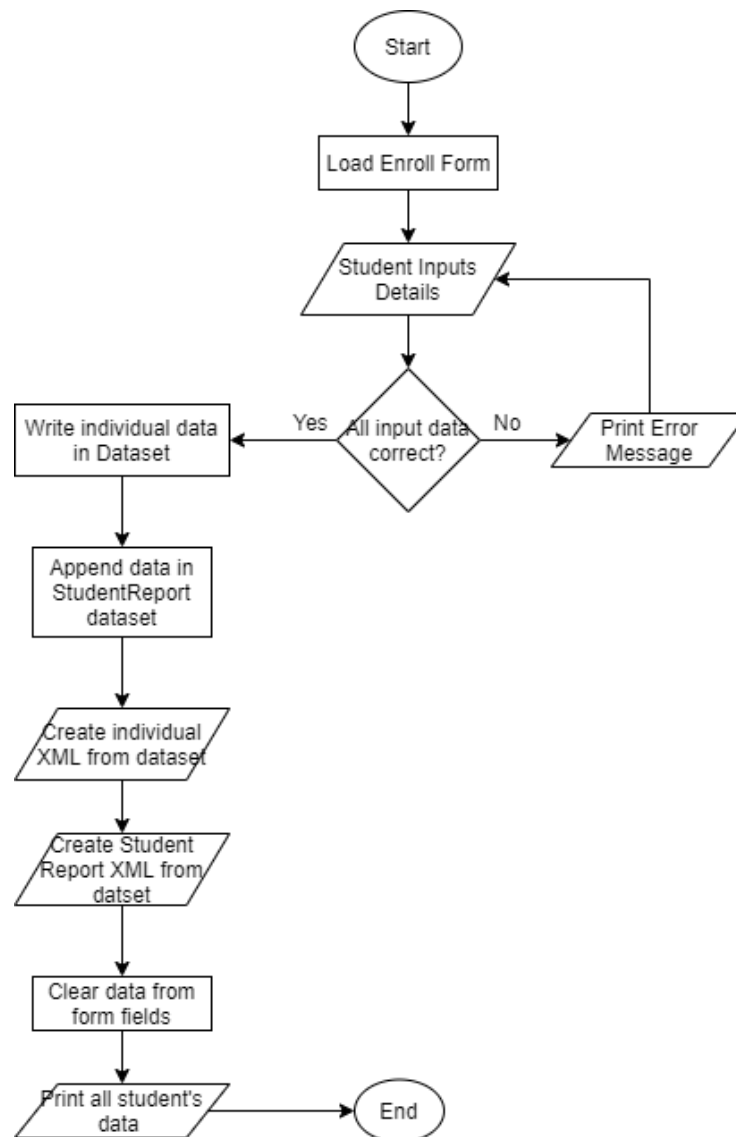


Figure 23: Flowchart of Enrol Students

- Import CSV data

This flowchart is of the method that reads the CSV file from the system and adds that data in the main student details file.

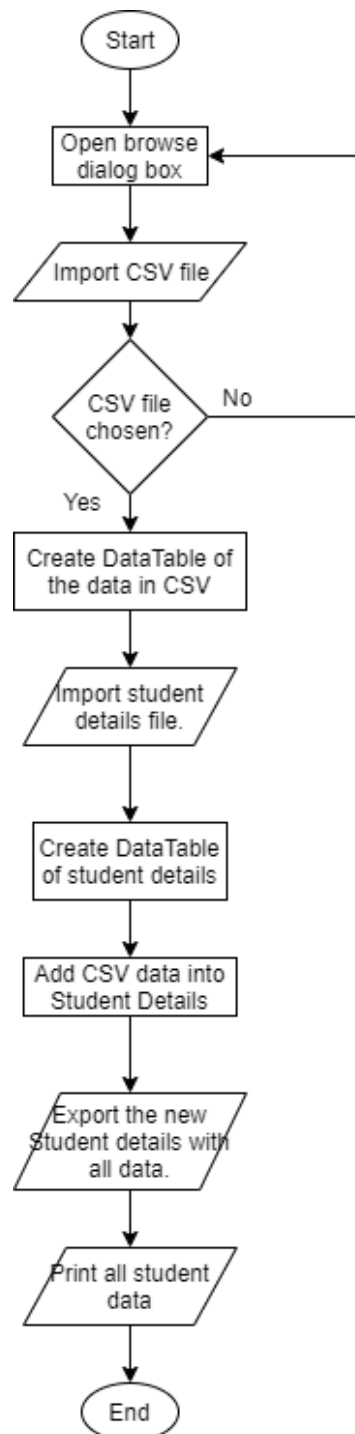


Figure 24: Import CSV flowchart

Algorithms

We sort the data of our Student Details once according to the Registration Date and once according to the Name of the Student. In C# we used the default sorting method to sort our data. The algorithm that is used in the default sort method is Quick Sort Algorithm.

Quick Sort Algorithm

This is an algorithm that uses divide and conquer method to sort the data in an array. The algorithm first takes a pivot point usually the last element. The main array is then partitioned in to sub array where left subarray will contain all the data smaller than the pivot and right subarray will contain all the data greater than pivot. Then the left subarray and right subarray are sorted recursively until final array has all the data sorted (InterviewBit, n.d.).

Let us take an example.

Input = {4 1 9 7 3 6}

Output = {1 3 4 6 7 9}

Step 1: The last element “6” is taken as pivot point and the whole array is portioned around 6.

Output: {4 1 3 6 9 7}

Step 2: Now sort the left subarray (4 1 3). Which is sorted by taking “3” as pivot and the sub array is portioned around it.

Output: {1 3 4 6 9 7}

Step 3: Since the right-side sub array (1 3 4) is now sorted now left side sub array (9 7) is sorted taken “7” as pivot.

Output: {1 3 4 6 7 9}

Step 4: At last all the sub arrays are also sorted so the final output is given.

Final Output: {1 3 4 6 7 9}

Reflection

The system in this coursework was developed using C# language in Visual Studio 2019. WPF (.Net Framework) was used to develop this project. Visual studio is a great program to use to make WPF forms because it provides us with drag and drop functionality while adding the UI elements in the forms. Giving the forms many designing elements and editing them to our own preference was easy as we can do them in the UI rather than writing codes. We can also run our project and see the project from the user's viewpoint and also debug the errors and problems in the program. Many plugins and tools are also available for visual studio on the internet which can be downloaded and used to add functionality like creating charts and creating class diagram in visual studio itself. These things made it easy for us in development process.

C# is an object-oriented language that runs in .NET Framework which was created by Microsoft. This language was used while coding in Visual Studio (w3Schools, n.d.). It is one of the most popular programming languages in the world which is also simple and easy to use. Its coding mechanism is closer to that of other popular languages like C++ and JAVA. Since we learned Java in previous years it was relatively easy to learn to program in this language. Coding in C# in Visual Studio is really easy because visual studio helps us by suggesting the codes to be written as well as auto tabs the codes as well as auto ends the brackets. This made coding in C# very easy to learn and helped us significantly while developing this system.

Conclusion

In the end, the developed application “Student Information System” is a good application that can be used in any institutions to keep track of their enrolled students. The app relatively user friendly and is quite functional. Developing this application for this coursework helped us to learn many aspects of creating WPF applications and helped us learn deeply about C#. This coursework helped us created a foundation in creating desktop applications. It was hard to learn at first. Creating a fully functional user-friendly system in WPF was new for us. The inner workings of the system were difficult to construct at first. But with the help of Module leader and research journals little by little the system came into fruition. And after many hours of coding and debugging the coursework was completed in time and was a success from which many new ideas were learned.

References

Amare, et al., 2018. Centralized School Management System for Government Schools in Ethiopia using Distributed Database. *International Journal of Engineering Trends and Technology(IJETT)* , 60(2), pp. 97-101.

Amingad, V., Prronima, S. & Arpitha, H., 2017. WEB BASED SCHOOL ADMINISTRATION SYSTEM. *International Research Journal of Engineering and Technology(IRJET)*, 04(05), pp. 2443-2445.

InterviewBit, n.d. *InterviewBit*. [Online]
Available at: <https://www.interviewbit.com/tutorial/quicksort-algorithm/>
[Accessed 10 01 2020].

Ma, L., 2016. Implementation on Curriculum Management System based on .NET. *Proceedings of the 2016 3rd International Conference on Management, Education Technology and Sports Science (METSS 2016)*, Volume 25, pp. 393-398.

w3Schools, n.d. *w3schools.com*. [Online]
Available at: https://www.w3schools.com/cs/cs_intro.asp
[Accessed 10 01 2020].

Appendix

Login.xaml.cs

```
using System.Windows;

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

        private void btnOK_Click(object sender, RoutedEventArgs e)
        {
            if(txtUsername.Text == "admin" && txtPassword.Password == "admin")
            {
                MainWindow mainWindow = new MainWindow();
                mainWindow.Show();
                Close();
            }
            else
            {
                MessageBox.Show("Username or Password Not Matched!");
            }
        }
    }
}
```

MainWindow.xaml.cs

```
using DataHandler;
using System.Windows;
using System.Data;
using System.IO;

namespace ApplicationDevelopment
{
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();

            txtResNo.Text = Res_no_read();
        }

        private void AddStudentData(DataSet dataSet)
        {
            var dr_Std = dataSet.Tables["Student"].NewRow();
            dr_Std["Name"] = txtName.Text;
            dr_Std["Address"] = txtAddr.Text;
            dr_Std["ContactNo"] = txtContact.Text;
            dr_Std["Email"] = txtEmail.Text;
        }
    }
}
```

```

        dr_Std["CourseEnroll"] = comboProgram.Text;
        dr_Std["RegistrationDate"] =
Date_Pick.SelectedDate.Value.ToString("yyyy/MM/dd");
        dataSet.Tables["Student"].Rows.Add(dr_Std);
    }

    private void AppendStudentData(DataSet dataSet)
    {
        if (File.Exists("Student Report/StudentReport.xml"))
        {
            dataSet.Tables["StudentReport"].ReadXml("Student
Report/StudentReport.xml");

            var dr_Std = dataSet.Tables["StudentReport"].NewRow();
            dr_Std["RegistrationNo"] = txtResNo.Text;
            dr_Std["Name"] = txtName.Text;
            dr_Std["Address"] = txtAddr.Text;
            dr_Std["ContactNo"] = txtContact.Text;
            dr_Std["Email"] = txtEmail.Text;
            dr_Std["CourseEnroll"] = comboProgram.Text;
            dr_Std["RegistrationDate"] =
Date_Pick.SelectedDate.Value.ToString("yyyy/MM/dd");
            dataSet.Tables["StudentReport"].Rows.Add(dr_Std);

            dataSet.Tables["StudentReport"].WriteXml("Student
Report/StudentReport.xml");
        }
        else
        {
            dataSet.Tables["StudentReport"].WriteXml("Student
Report/StudentReport.xml");
            AppendStudentData(dataSet);
        }
    }

    private void Button_Click(object sender, RoutedEventArgs e)
    {
        if (txtName.Text == "")
        {
            MessageBox.Show("Please Fill your Name.");
        }
        else if (txtAddr.Text == "")
        {
            MessageBox.Show("Please Fill your Address.");
        }
        else if (txtContact.Text == "")
        {
            MessageBox.Show("Please Fill your contact.");
        }
        else if (IsValidEmail(txtEmail.Text) == false)
        {
            MessageBox.Show("Please Fill Email correctly.");
        }
        else if (Date_Pick.SelectedDate == null)
        {
            MessageBox.Show("Please Select a date.");
        }
        else if (comboProgram.SelectedItem == null)
        {
            MessageBox.Show("Please Select a course.");
        }
    }

```



```

        else
        {
            var handler = new Handler();
            var dataSet = handler.CreateDataSet();
            AddStudentData(dataSet);
            AppendStudentData(dataSet);

            dataSet.Tables["Student"].WriteXml("Individual Data/" + txtName.Text +
"-Data-" + txtResNo.Text + ".xml");
            Res_no_write(txtResNo.Text);
            txtResNo.Text = Res_no_read();

            MessageBox.Show("Student Details Saved!");

            Clear_field();
        }
    }

    private void btnClear_Click(object sender, RoutedEventArgs e)
    {
        Clear_field();
    }

    public void Clear_field()
    {
        txtName.Text = "";
        txtAddr.Text = "";
        txtContact.Text = "";
        txtEmail.Text = "";
        comboProgram.Text = "";
        Date_Pick.Text = "";
    }

    private void Res_no_write(string text)
    {
        File.WriteAllText("Res Count/count_res.txt", text);
    }

    private string Res_no_read()
    {
        int i = 100000;
        if (File.Exists("Res Count/count_res.txt"))
        {
            string text = File.ReadAllText("Res Count/count_res.txt");
            i = int.Parse(text.ToString());
            i += 1;
        }
        else
        {
            File.WriteAllText("Res Count/count_res.txt", "100000");
        }

        return i.ToString();
    }

    private void btnDetails_Click(object sender, RoutedEventArgs e)
    {
        StudentList studentList = new StudentList();
        studentList.Show();
        Close();
    }
}

```

```

        bool IsValidEmail(string email)
        {
            try
            {
                var eAdd = new System.Net.Mail.MailAddress(email);
                return eAdd.Address == email;
            }
            catch
            {
                return false;
            }
        }
    }
}

```

StudentDetails.xaml.cs

```

using DataHandler;
using System;
using System.Data;
using System.Data.OleDb;
using System.Globalization;
using System.IO;
using System.Windows;

namespace ApplicationDevelopment
{
    public partial class StudentList : Window
    {
        DataTable dataTable;

        public StudentList()
        {
            InitializeComponent();
            Load_stu_data();
        }

        private void Load_stu_data()
        {
            if (File.Exists("Student Report/StudentReport.xml"))
            {
                var dataset = new DataSet();
                dataset.ReadXml("Student Report/StudentReport.xml");

                dataTable = dataset.Tables["StudentReport"];
                stdGrid.DataContext = dataTable.DefaultView;
            }
        }

        private void btnWeekly_Click(object sender, RoutedEventArgs e)
        {
            int total_Com = 0;
            int total_Mul = 0;
            int total_Net = 0;

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

            for (int i = 0; i < dataTable.Rows.Count; i++)

```

```

    {
        String col = dataTable.Rows[i]["CourseEnroll"].ToString();
        if (col == "Computing")
        {
            total_Com++;
        }
        else if (col == "Multimedia Technologies")
        {
            total_Mul++;
        }
        else if (col == "Networks and IT Security")
        {
            total_Net++;
        }
    }

    dt.Rows.Add("Computing", total_Com);
    dt.Rows.Add("Multimedia Technologies", total_Mul);
    dt.Rows.Add("Networks and IT Security", total_Net);

    stdGrid.DataContext = dt.DefaultView;
    lblWindowName.Content = "DataTable of Weekly Report";
}

private void btnDateSort_Click(object sender, RoutedEventArgs e)
{
    var dataset = new DataSet();
    dataset.ReadXml("Student Report/StudentReport.xml");

    dataTable = dataset.Tables["StudentReport"];
    dataTable.DefaultView.Sort = "RegistrationDate ASC";
    stdGrid.DataContext = dataTable.DefaultView;
    lblWindowName.Content = "DataTable Sorted by Registration Date";
}

private void btnFNameSort_Click(object sender, RoutedEventArgs e)
{
    var dataset = new DataSet();
    dataset.ReadXml("Student Report/StudentReport.xml");

    dataTable = dataset.Tables["StudentReport"];
    dataTable.DefaultView.Sort = "Name ASC";
    stdGrid.DataContext = dataTable.DefaultView;
    lblWindowName.Content = "DataTable Sorted by Student Name";
}

private void btnStuDetails_Click(object sender, RoutedEventArgs e)
{
    if (File.Exists("Student Report/StudentReport.xml"))
    {
        var dataset = new DataSet();
        dataset.ReadXml("Student Report/StudentReport.xml");

        DataTable tableStd = dataset.Tables["StudentReport"];
        stdGrid.DataContext = tableStd.DefaultView;
        lblWindowName.Content = "DataTable of Student Details";
    }
}

private void btnAdd_Click(object sender, RoutedEventArgs e)
{

```

```

        MainWindow mainWindow = new MainWindow();
        mainWindow.Show();
        Close();
    }

    private void btnViewCSV_Click(object sender, RoutedEventArgs e)
    {
        Microsoft.Win32.OpenFileDialog dlg = new Microsoft.Win32.OpenFileDialog();
        dlg.DefaultExt = ".csv";
        Nullable<bool> result = dlg.ShowDialog();

        if (result == true)
        {
            DataTable tableStd = GetStudentReportFromCsv(dlg.FileName, true);
            stdGrid.DataContext = tableStd.DefaultView;
            lblWindowName.Content = "DataTable showing CSV files.";
        }
    }

    static DataTable GetStudentReportFromCsv(string path, bool isFirstRowHeader)
    {
        string header = isFirstRowHeader ? "Yes" : "No";

        string pathAddress = Path.GetDirectoryName(path);
        string fileName = Path.GetFileName(path);

        string sql = @"SELECT * FROM [" + fileName + "]";

        using (OleDbConnection connection = new OleDbConnection(
            @"Provider=Microsoft.Jet.OLEDB.4.0;Data Source=" + pathAddress +
            ";Extended Properties=\"Text;HDR=" + header + "\""))
        using (OleDbCommand command = new OleDbCommand(sql, connection))
        using (OleDbDataAdapter adapter = new OleDbDataAdapter(command))
        {
            DataTable dataTable = new DataTable();
            dataTable.Locale = CultureInfo.CurrentCulture;
            adapter.Fill(dataTable);
            return dataTable;
        }
    }

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

    private void btnImport_Click(object sender, RoutedEventArgs e)
    {
        Microsoft.Win32.OpenFileDialog dlg = new Microsoft.Win32.OpenFileDialog();
        dlg.DefaultExt = ".csv";
        Nullable<bool> result = dlg.ShowDialog();

        if (result == true)
        {
            DataTable tableStd = GetStudentReportFromCsv(dlg.FileName, true);
            DataTable dtCloned = tableStd.Clone();
            dtCloned.Columns["ContactNo"].DataType = typeof(String);
            dtCloned.Columns["RegistrationDate"].DataType = typeof(String);
            foreach (DataRow row in tableStd.Rows)
            {

```

```

        dtCloned.ImportRow(row);
    }
    var handler = new Handler();
    var dataSet = handler.CreateDataSet();
    dataSet.Tables["StudentReport"].ReadXml("Student
Report/StudentReport.xml");
    dataSet.Tables["StudentReport"].Merge(dtCloned);
    dataSet.Tables["StudentReport"].WriteXml("Student
Report/StudentReport.xml");

    {
        var dataset = new DataSet();
        dataset.ReadXml("Student Report/StudentReport.xml");
        DataTable table = dataset.Tables["StudentReport"];
        stdGrid.DataContext = table.DefaultView;
        lblWindowName.Content = "DataTable of Student Details";
    }
}
}
}
}
}
}
}
}
}
}

```

Chart.xaml.cs

```

using System;
using System.Collections.Generic;
using System.Data;
using System.Windows;
using System.Windows.Controls.DataVisualization.Charting;

namespace ApplicationDevelopment
{
    public partial class Chart : Window
    {
        public Chart()
        {
            InitializeComponent();
            showPieChart();
        }

        private void showPieChart()
        {
            var dataset = new DataSet();
            dataset.ReadXml("Student Report/StudentReport.xml");

            DataTable dataTable = dataset.Tables["StudentReport"];

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

            for (int i = 0; i < dataTable.Rows.Count; i++)
            {
                String col = dataTable.Rows[i]["CourseEnroll"].ToString();
                if (col == "Computing")
                {
                    total_Com++;
                }
                else if (col == "Multimedia Technologies")

```

```

        {
            total_Mul++;
        }
        else if (col == "Networks and IT Security")
        {
            total_Net++;
        }
    }

    ((PieSeries)pieChart).ItemsSource = new KeyValuePair<string, int>[]{
        new
        KeyValuePair<string,int>("Computing", total_Com),
        new
        KeyValuePair<string,int>("Multimedia Technologies", total_Mul),
        new KeyValuePair<string,int>("Networks
and IT Security", total_Net) };
    }
}

```

Handler.cs

```

using System.Data;

namespace DataHandler
{
    public class Handler
    {
        public DataSet CreateDataSet()
        {
            var dataSet = new DataSet();
            dataSet.Tables.Add(CreateStudentTable());
            dataSet.Tables.Add(StudentReportTable());

            return dataSet;
        }

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

            dataTable.Columns.Add(dataColumn);

            dataTable.Columns.Add("RegistrationNo", typeof(int));
            dataTable.Columns.Add("Name", typeof(string));
            dataTable.Columns.Add("Address", typeof(string));
            dataTable.Columns.Add("ContactNo", typeof(string));
            dataTable.Columns.Add("Email", typeof(string));
            dataTable.Columns.Add("CourseEnroll", typeof(string));
            dataTable.Columns.Add("RegistrationDate", typeof(string));
            dataTable.PrimaryKey = new DataColumn[] { dataTable.Columns["ID"] };
            return dataTable;
        }

        private DataTable StudentReportTable()
        {
            var dataTable = new DataTable("StudentReport");
            DataColumn dataColumn = new DataColumn("ID", typeof(int));

```

```
dataColumn.AutoIncrement = true;
dataColumn.AutoIncrementSeed = 1;
dataColumn.AutoIncrementStep = 1;

dataTable.Columns.Add(dataColumn);

dataTable.Columns.Add("RegistrationNo", typeof(int));
dataTable.Columns.Add("Name", typeof(string));
dataTable.Columns.Add("Address", typeof(string));
dataTable.Columns.Add("ContactNo", typeof(string));
dataTable.Columns.Add("Email", typeof(string));
dataTable.Columns.Add("CourseEnroll", typeof(string));
dataTable.Columns.Add("RegistrationDate", typeof(string));

return dataTable;
}
}
```