Informatics College Pokhara

Application Development

CS6004NP

Coursework 1

<table>
<tr><td>Submitted By:</td><td></td><td>Submitted To:</td></tr>
<tr><td>Student Name:</td><td>Prashanta Timsina</td><td>Mr. Ishwor Sapkota</td></tr>
<tr><td>London Met ID:</td><td>1731921</td><td>Module Leader</td></tr>
<tr><td>Group:</td><td>L3C2</td><td>Application Development</td></tr>
<tr><td>Date:</td><td>10-Jan-2019</td><td></td></tr>
</table>

Abstract

This is an individual course work of the module "Application Development" for Student Management System. The application is developed using Visual Studio Platform with C# language. The coursework is released in the week 5 and it is supposed to be submitted in the week 11.

With the great contribution of Mr. Ishwor Sapkota (Module Leader) the course work was completed within the time frame.

Table of Contents

Table of Figure

Table of Tables

1. Introduction

The Student Information System (SIS) is a fully computerized system or more precisely a database where all the student related data can be stored, retrieved, monitored and analysed. The data is saved in one central location which can be accessed by multiple persons at the same time provided they have the login credentials, this ensures the safety of the stored information. Registration, demand generation, admission, billing, provision of financial aid to students and many other things can be managed with ease making the whole process of student enrolment quick, error free, systematic and undemanding.

Student information system has gained huge popularity in the educational sector because of its unique characteristics which has helped countless schools and higher educational institutes immensely. Right from maintaining and updating student data with proficiency to analysing and creating reports, the education management software has helped substantially. Here are some valued features of SIS (MasterSoft, n.d.).

Figure 1: Student Management System.

1.1. Current Scenario

If we look at the current scenario there are numerous schools who keep record of their data in old traditional system which are often paper based system. These system are often slower and incontinent to operate and maintain. In addition to that they are often considered unsafe as some with access to the files can change and modify them.

1.2. Proposed System

The proposed system is a digital version of Student Management System which does all the task the tradition system along with some additional benefits of high security, easy to operate & maintain & reliable. Some of the advantages of digital student management system over the traditional system are given below:

- Eco-Friendly: paperwork can be avoided
- Efficient control over student data
- Monitor student performance
- Supervise multiple branches
- Cost-efficient and User-friendly
- Single solution for total College management
- Easy access to forums, attendance, timetable, marks, grades and examination schedule

2. User Manual

This manual provides the guidelines to run the Student Management System application:

2.1. Running the Application

1. To run the Student Management System first open the Application Student Management System folder and run Student Management System.sln
2. Or to run the .exe file Open "StudentManagementSystem" Folder.
3. Open "bin" folder.
4. Open "debug" folder
5. Then, run "StudentManagementSytem.exe" application.

2.2. Login Screen

The figure below is the screenshot of the login screen. After running the application the user is prompted with the login window. Here, the user enters the username and password for authentication purpose.
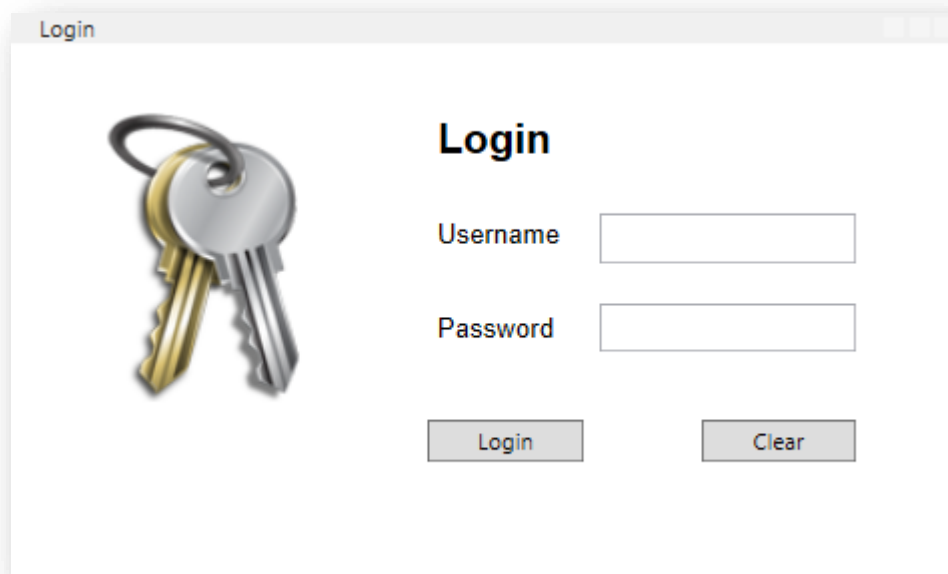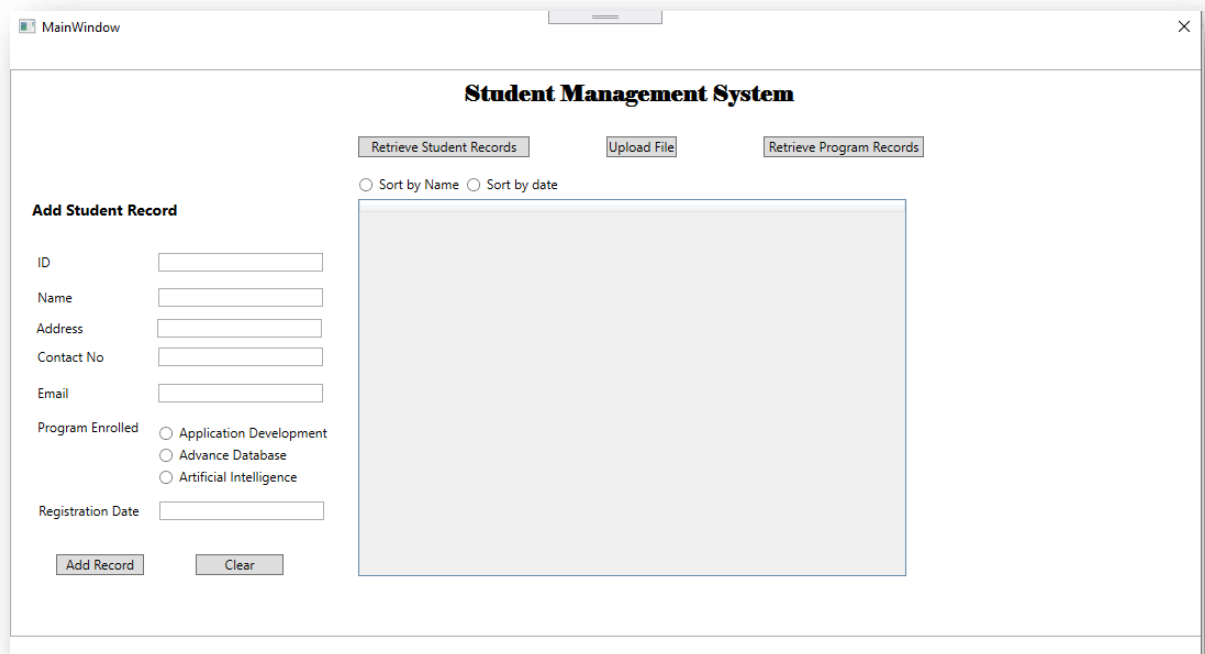

Figure 2: Login Screen.

2.3. Home Page

The figure below is a screenshot of home page of the Student Management System.   After logging in successfully the user is then prompted with the main window which is also the home page. Here, the user can add student record, retrieve student records, upload student record for bulk entry & finally retrieve program record. In addition to that the user can also sort the student records according to name or by registration date. The user can also view a chart representing data of the program record.

Figure 3: Home page of Student Management System.

2.4. Adding Student Record.

The figure below is a screenshot of add student record section which is located in the home page of Student Management System. Here, as the name suggests, the user can add student record into database by filling the respective textbox. In, addition to that the user can also clear all the text boxes for any invalid entry.

Figure 4: Add Student Record section of Student Management System.

2.5. Retrieving Student Record.

The screenshot below illustrate the Retrieve Student Record section of the Student Management System. Here, the user can retrieve the student records stored in the database. The record are displayed in the datagrid as tables. In addition to that, the user can also sort the retrieved data in accordance to the name of the students or the registration data.

Figure 5: Retrieve Student Record section of the Student Management System.

2.6. Retrieving Program Record.

The figure below is a screenshot of Retrieve Program Record section of the Student Management System. Here, the user can retrieve program record based on the number of enrolled student in a certain program. The records are then displayed on the datagrid in table format. Also a pie chart is generated based on the program record which is displayed on the right side of the table.
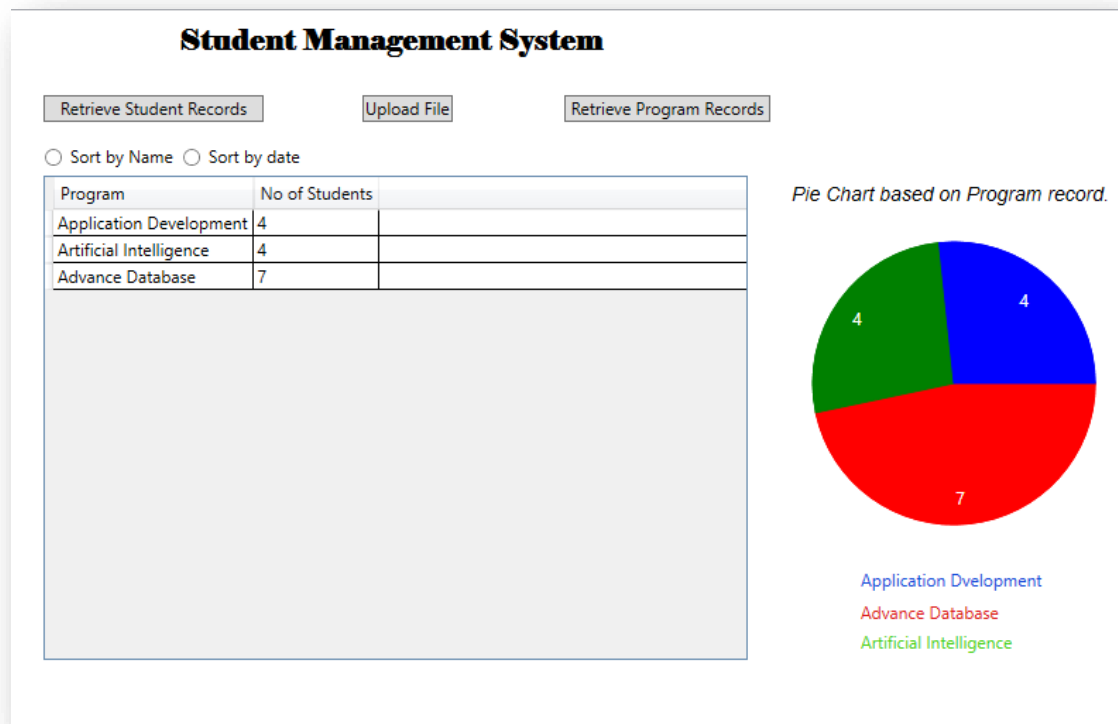
Figure 6: Retrieve Program Record section of Student Management System.

2.7. Uploading CSV file.

The figure below is a screenshot of the upload csv file section of the Student Management System. Here, the user can upload a csv file for bulk data entry. After clicking the button the user is prompted with the file explorer window where the user can select a file to upload. The record is then displayed on the datagrid along with the insertion on the database.
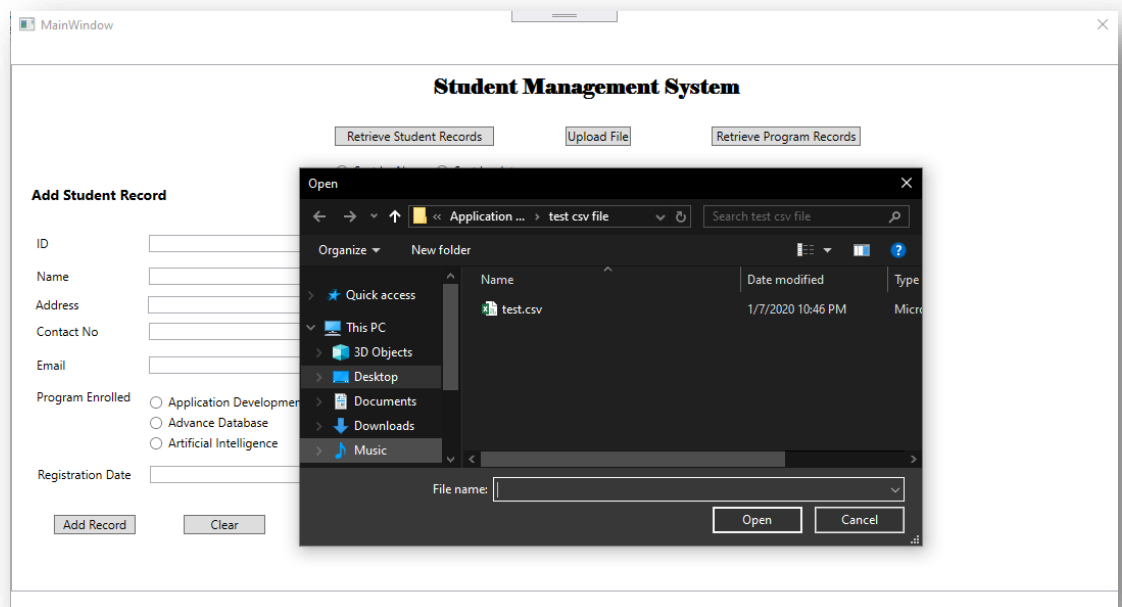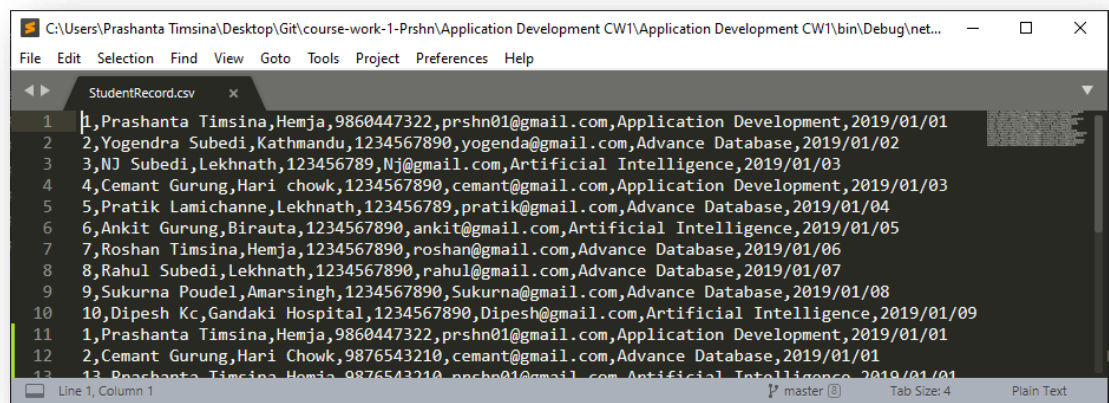
Figure 7: Upload CSV file section of the Student Management System.

2.8. Csv files

The data storing & retrieving of Student Management System is done on the CSV file located inside the project's "bin\debug" folder. A Comma Separated Values (CSV) file is a plain text file that contains a list of data. These files are often used for exchanging data between different applications. For example, databases and contact managers often support CSV files.

These files may sometimes be called Character Separated Values or Comma Delimited files. They mostly use the comma character to separate (or delimit) data, but sometimes use other characters, like semicolons. The idea is that you can export complex data from one

application to a CSV file, and then import the data in that CSV file into another application.

Figure 8: Screenshot of the CSV file for Student Management System.

3. Research

During the initial development stage some research was done on the digital Student Management System. The research were based on the sites on the internet and some journals and articles. Some of the material used as the references in the development cycle of the Student Management System are given below:

- Features Of Student Information Management System (SIMS)

  This is an article based on Digital Student Management System which is published in the site MasterSoft. Here, the author publishes some of the most underlying features of the proposed system. Some more information on the article can be obtained on the link given below.

  https://www.iitms.co.in/products/student-information-system-sis/

- What is Student Management System Software? Advantages vs Disadvantages

  In this article the author which is EduSys.co publishes about the advantage and disadvantages of the student management system. Some more information on the site can be found on the link below.

  https://www.edusys.co/blog/what-is-student-management-systemsoftware

- Student Management System – A Survey

  This is a paper published on the International research journal of computer science on the month of May of 2017. The author of the paper is Dr M Ramakrishna PG student & professor of computer science and engineering. In this paper professor illustrate about the current system, system design, proposed system, dataflow diagram etc. about the student management system. A link is given below referring to the original paper that was published.

  http://irjcs.com/volumes/vol4/iss05/46.MYCSSP10083.pdf

4. System Architecture

Systems Architecture is a generic discipline to handle objects (existing or to be created) called "systems", in a way that supports reasoning about the structural properties of these objects. Systems Architecture is a response to the conceptual and practical difficulties of the description and the design of complex systems.
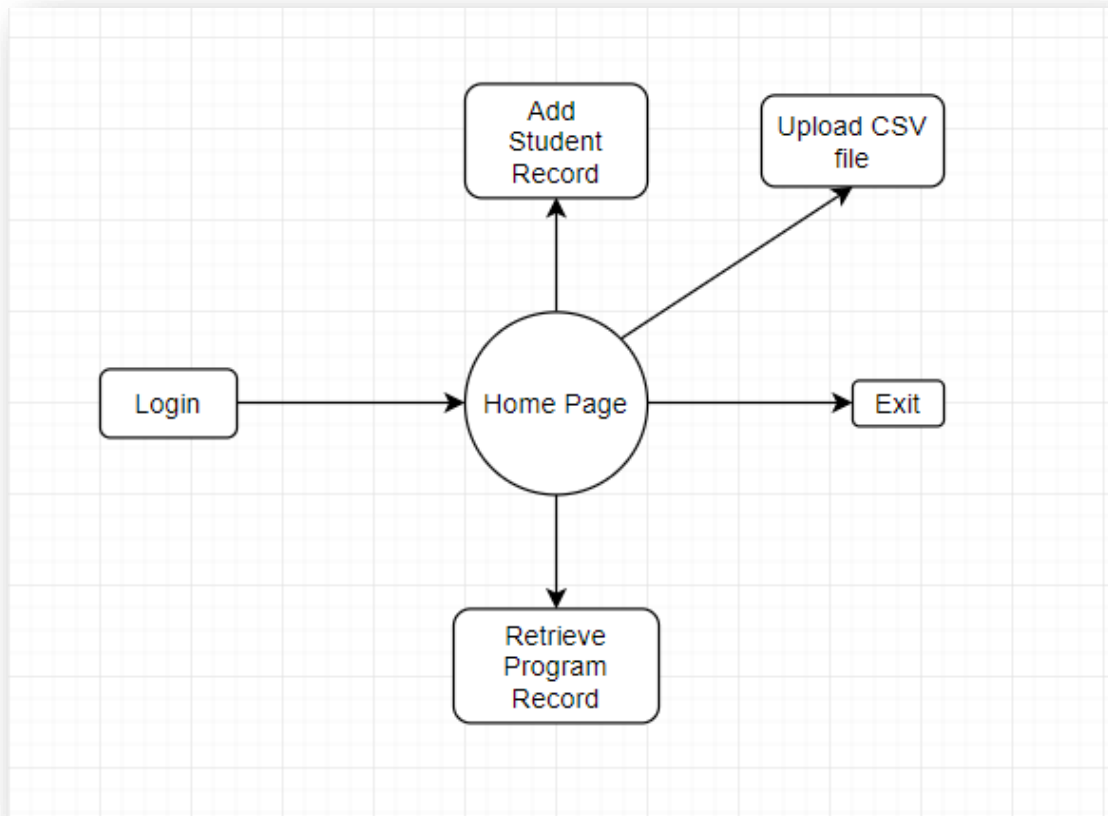
Figure 9: System Architecture Diagram of SMS.

4.1. Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object-oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram. The class diagram for the Student Management System is given below:
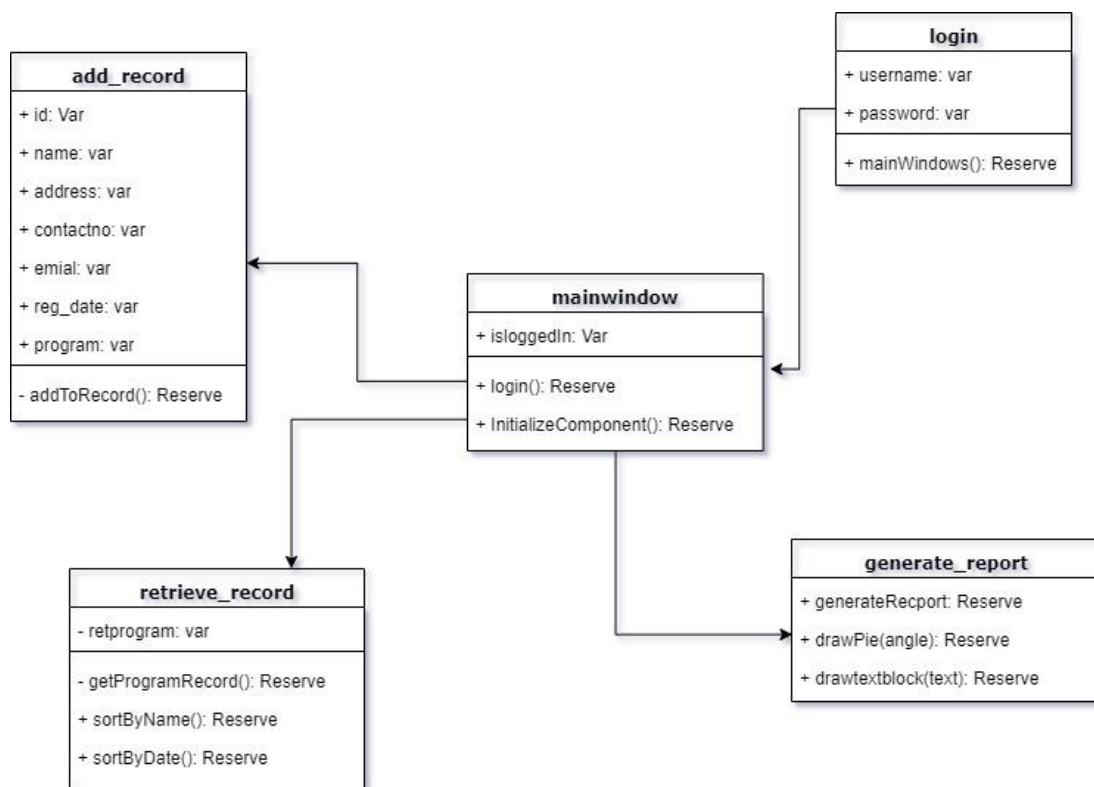
Figure 10: Class Diagram of SMS.

4.2. Class Description

- Login

Methods	Description
mainWindow	The method redirect the program to the main widow after successful authentication.

Table 1: Class description of Login.

- mainWindow

Methods	Description
Login	The method opens the login window for authentication purpose.
InitializeComponent	This is built in method of the program. The method initialize of the wpf components of the method.

Table 2: Class Description of MainWindow.

- add_record

Methods	Description
addToRecord	This method is used to add record of student to the database. The method pulls all the data from the textbox verifies them and add them to the record.

Table 3 Class Description of add_record.:

- retrieve_record

Methods	Description
getStudentRecord	The method generates the student records available in the database.
sortByName	The method sort the student record in accordance to name of the students.
sortByDate	The method sort the student record in accordance to the registration date.

Table 4: Class description of retrieve_record.

- generateRecord

Methods	Description
generateRecord	The method generates the program record from the database.
drawPie	The method draws an arc on the canvas for drawing pie.
drawTextBlock	The method draws texts on the canvas for representation of the pie.

Table 5: Class description of generate_repprt

4.3. Flowcharts

A flow chart is a graphical or symbolic representation of a process. Each step in the process is represented by a different symbol and contains a short description of the process step. The flow chart symbols are linked together with arrows showing the process flow direction. The following flowcharts were used in our project.

- Add to record
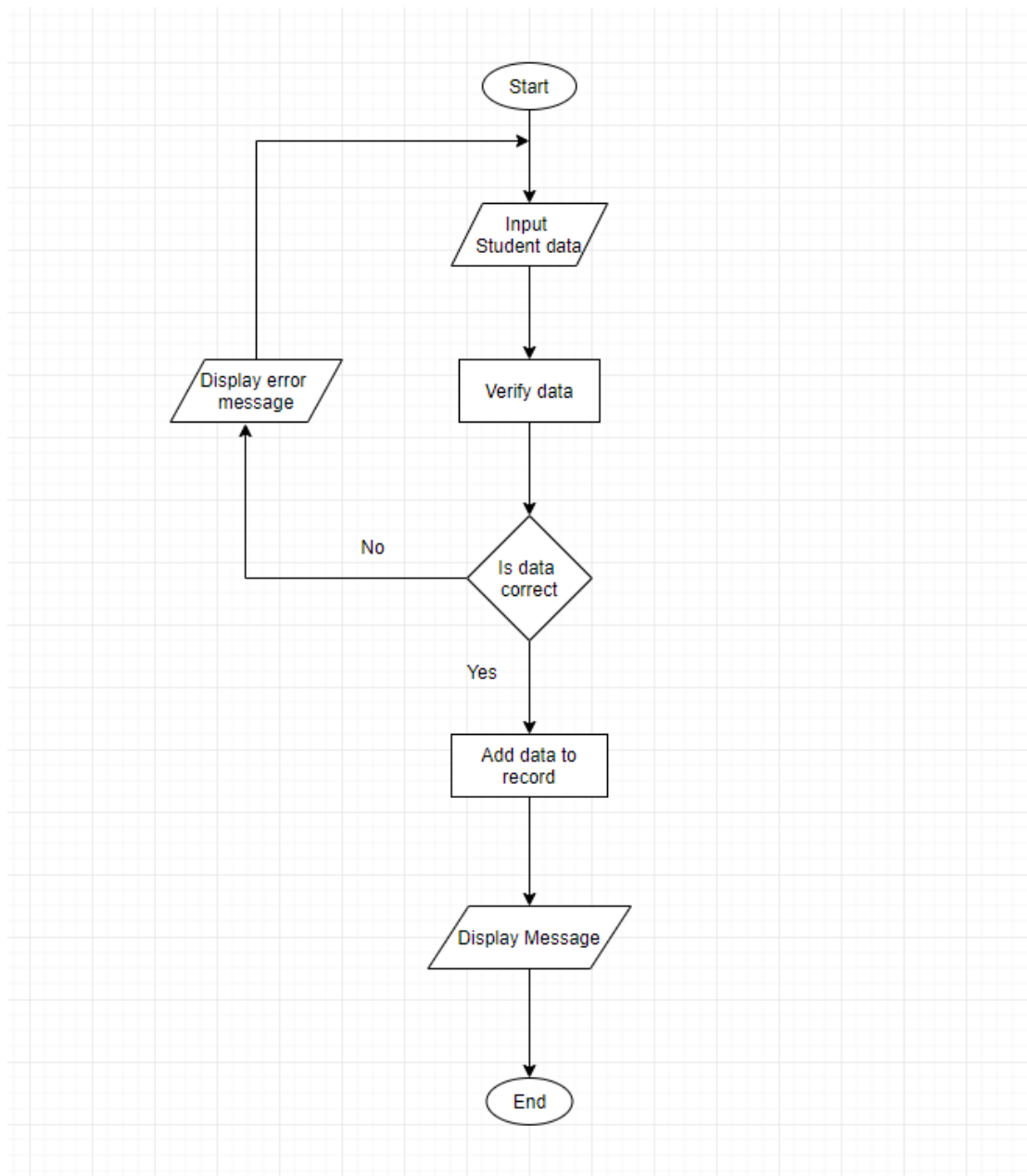
Figure 11: Add to record flowchart.

- Retrieve Student Record
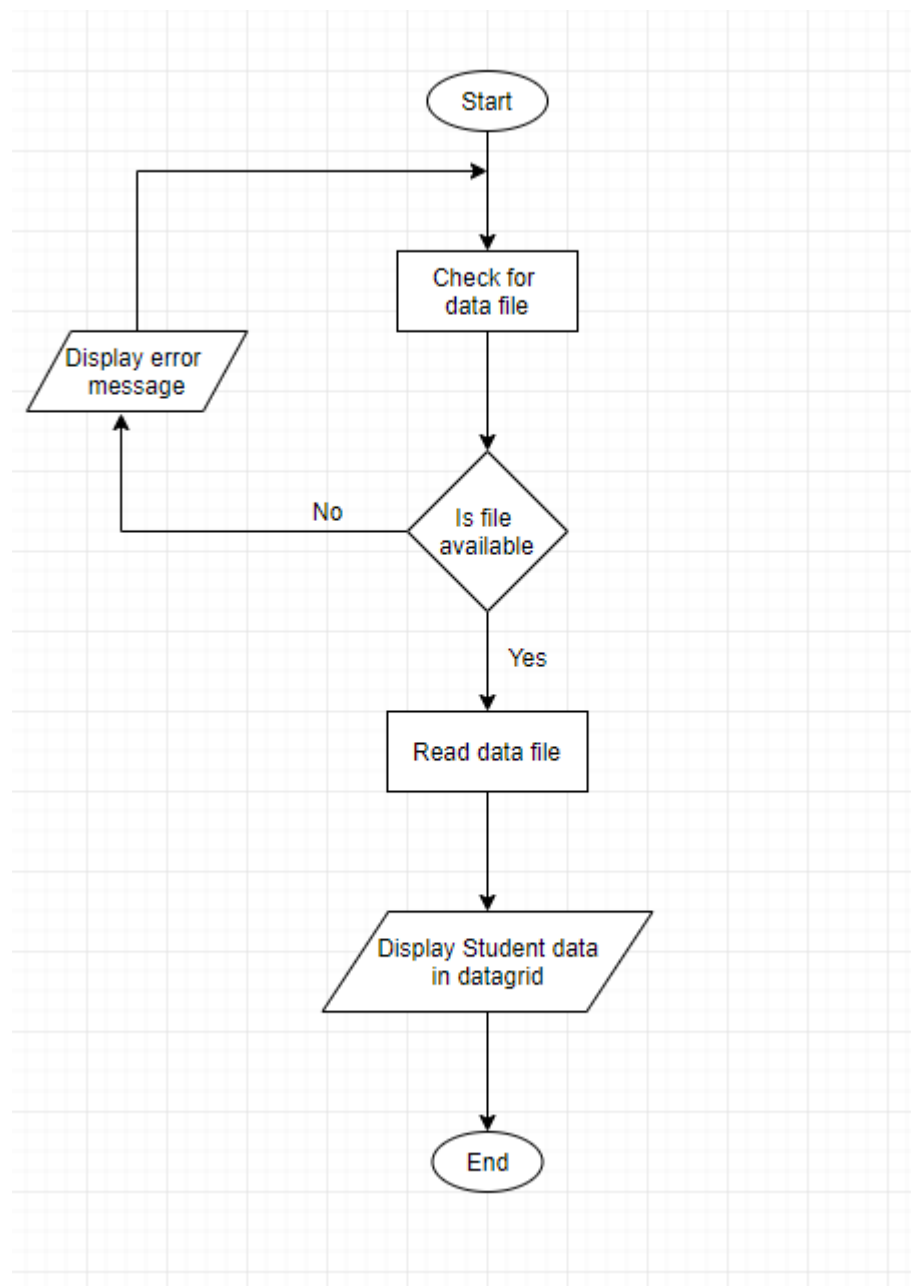
Figure 12: Flowchart of Retrieve Student Record.

- Retrieve Program Record
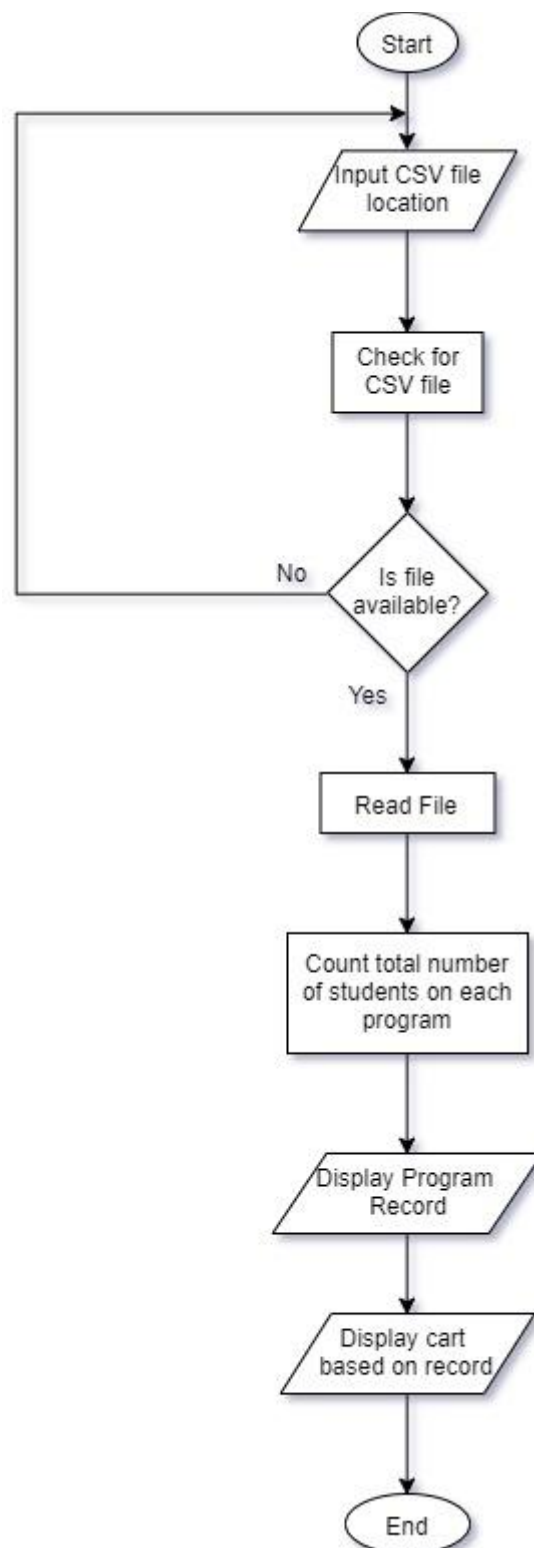
Figure 13: Flowchart for Retrieve Program Record.

4.4. Algorithm of the Report

- Add to Record
    1. Start
    2. Input student data
    3. Verify data
    4. If invalid, go to step 2.
    5. If valid, go to step 6.
    6. Add data to record.
    7. Display an appropriate message.
    8. End.

- Retrieve Student Record
    1. Start
    2. Check for csv file
    3. Is file available
    4. If no, go to step 2
    5. If yes, go to step 6
    6. Display student data
    7. End

- Retrieve Program Record
    1. Start
    2. Input csv file location
    3. Check for csv file
    4. Is file available
    5. No, go to step 2
    6. Yes, go to step 7
    7. Read file
    8. Count total no of students of each program
    9. Display program record
    10. Display chart based on record.
    11. End

5. Sorting Algorithm

The sorting algorithm that is being used in our application is bubble sort algorithm. Here, the sorting algorithm is being used to sort the student record in accordance to name and registration date.

Bubble Sort is a simple algorithm which is used to sort a given set of n elements provided in form of an array with n number of elements. Bubble Sort compares 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.

If we have total n elements, then we need to repeat this process for n-1 times.

It is known as bubble sort, because with every complete iteration the largest element in the given array, bubbles up towards the last place or the highest index, just like a water bubble rises up to the water surface.

Sorting takes place by stepping through all the elements one-by-one and comparing it with the adjacent element and swapping them if required.

Following are the steps involved in bubble sort (for sorting a given array in ascending order):

1. Starting with the first element (index = 0), compare the current element with the next element of the array.
2. If the current element is greater than the next element of the array, swap them.
3. If the current element is less than the next element, move to the next element. Repeat Step 1.

   Let's consider an array with values {5, 1, 6, 2, 4, 3}

   Below, we have a pictorial representation of how bubble sort will sort the given array.
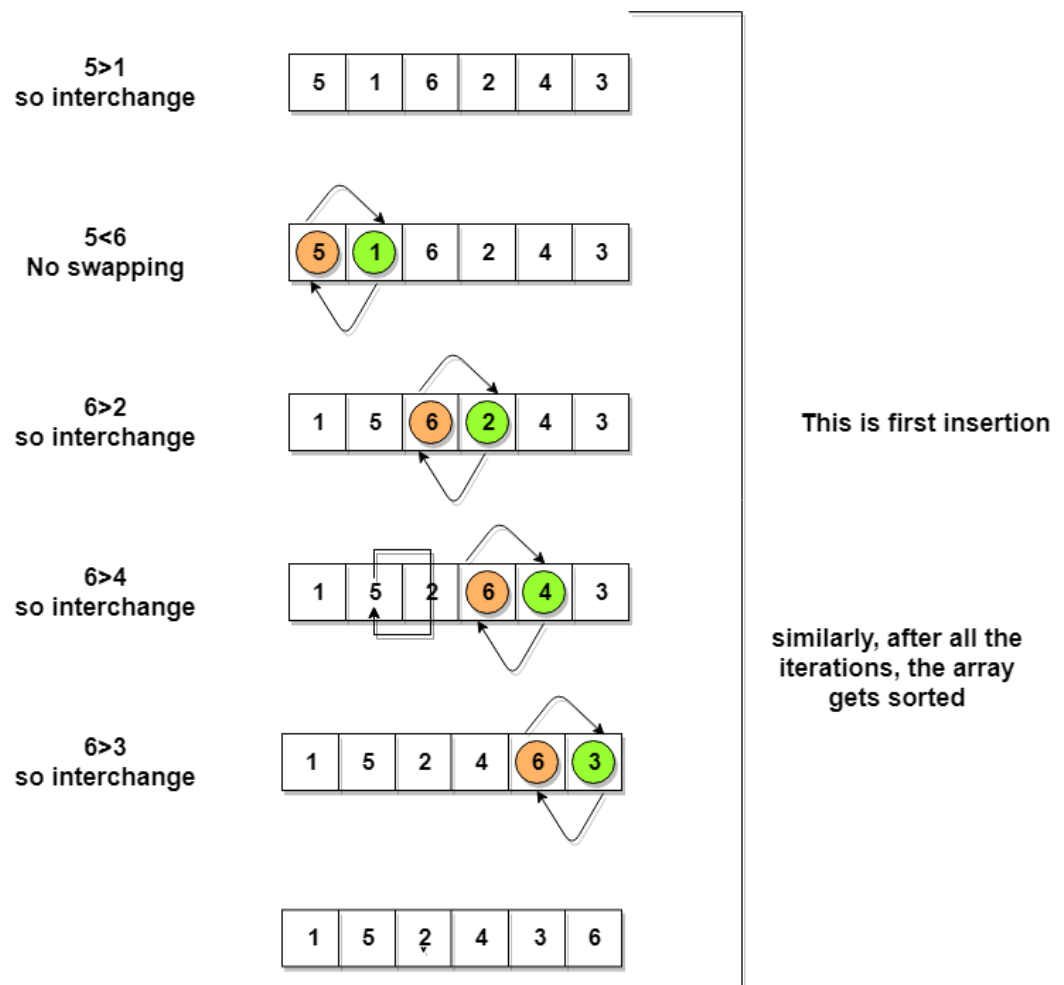
Figure 14: Bubble Sort Algorithm

So as we can see in the representation above, after the first iteration, 6 is placed at the last index, which is the correct position for it. Similarly after the second iteration, 5 will be at the second last index, and so on (Ahlawat, n.d.).

4. Reflection

The proposed Student Management System was developed on the using Visual studio using C# programming language. The IDLE was quite a new experience for me, also the C# programming language. Though, I had some experiences on C and C++ it was somewhat different than what I used to remember. Programming in C# was easy for me. It was quite similar to other programming languages that I had used before. Although I had some minor problems starting up they weren't major issues as I went on. Also, the inputs from the module leader was quite helpful at different times. I also did lots of research on the internet regarding the C# codes, function, errors and libraries. In a nutshell, I can say it was quite a pleasing experience for me. I am hopeful that this experience would be of great use to me in my future days.

5. Conclusion

After the project I was drawn to the conclusion that the proposed system "Student Management System" was a great start for me into the C# programming language. Since we had lots of time and the fact that the project was not that complicated in itself was a huge relief to me. The project was completed with the given time frame. Lots of research was done during the development phase of the project. Also the inputs from the module leader was of a great help. All the coding was done on the visual studio application. Microsoft Visual Studio is an integrated development environment from Microsoft. It is used to develop computer programs, as well as websites, web apps, web services and mobile apps. During the development I had the opportunity to be familiar with coding C# programming language. After the completion of the project, I can say now I have certain level of experience and expertise that I hope someday would be of great use to me.

Bibliography

Ahlawat, A., n.d. *StudyTonight.* [Online]
Available at: https://www.studytonight.com/data-structures/bubble-sort
[Accessed 2020].

MasterSoft, n.d. [Online]
Available at: https://www.iitms.co.in/products/student-information-system-sis/

Appendix

- MainWindow.Xaml

```xml
<Window x:Class="Application_Development_CW1.MainWindow"
    xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
    xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml"
    xmlns:d="http://schemas.microsoft.com/expression/blend/2008"
    xmlns:mc="http://schemas.openxmlformats.org/markup-
compatibility/2006"
    xmlns:local="clr-namespace:Application_Development_CW1"
    mc:Ignorable="d"
    Title="MainWindow" Height="600" Width="1100"
ResizeMode="NoResize">
  <Grid Height="516" Margin="0,0,-43,0">
    <TextBox Name="txtEditor" />
    <Button Name="btnOpenFile" HorizontalAlignment="Left"
Margin="543,61,0,0" VerticalAlignment="Top"
Click="btnOpenFile_Click">Upload File</Button>
    <TextBlock HorizontalAlignment="Center" Margin="0,10,0,0"
Text="Student Management System" TextWrapping="Wrap"
VerticalAlignment="Top" FontSize="20" FontWeight="Bold"
FontFamily="Elephant"/>
    <TextBlock HorizontalAlignment="Left" Margin="20,118,0,0" Text="Add
Student Record" TextWrapping="Wrap" VerticalAlignment="Top"
FontSize="14" FontWeight="Bold"/>
    <Label Content="ID" HorizontalAlignment="Left" Margin="20,163,0,0"
VerticalAlignment="Top"/>
    <Label Content="Name" HorizontalAlignment="Left" Margin="20,195,0,0"
VerticalAlignment="Top"/>
    <Label Content="Address" HorizontalAlignment="Left"
Margin="19,223,0,0" VerticalAlignment="Top"/>
    <Label Content="Contact No" HorizontalAlignment="Left"
Margin="20,249,0,0" VerticalAlignment="Top"/>
    <Label Content="Email" HorizontalAlignment="Left" Margin="20,282,0,0"
VerticalAlignment="Top"/>
    <Label Content="Program Enrolled" HorizontalAlignment="Left"
Margin="20,313,0,0" VerticalAlignment="Top"/>
    <Label Content="Registration Date" HorizontalAlignment="Left"
Margin="21,389,0,0" VerticalAlignment="Top"/>
    <TextBox x:Name="tbox_id" HorizontalAlignment="Left"
Margin="135,167,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150"/>
    <TextBox x:Name="tbox_name" HorizontalAlignment="Left"
Margin="135,199,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150"/>
    <TextBox x:Name="tbox_address" HorizontalAlignment="Left"
Margin="134,227,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150"/>
```

```xml
        <TextBox x:Name="tbox_contact" HorizontalAlignment="Left"
Margin="135,253,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150"/>
        <TextBox x:Name="tbox_email" HorizontalAlignment="Left"
Margin="135,286,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150"/>
        <TextBox x:Name="tbox_regdate" HorizontalAlignment="Left"
Margin="136,393,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Width="150"/>
        <Button x:Name="button_addrec" Content="Add Record"
HorizontalAlignment="Left" Margin="42,441,0,0" VerticalAlignment="Top"
Width="80" Click="button_addrec_Click"/>
        <Button x:Name="button_clear" Content="Clear"
HorizontalAlignment="Left" Margin="169,441,0,0" VerticalAlignment="Top"
Width="80" Click="button_clear_Click"/>
        <DataGrid x:Name="datagrid_1" Margin="317,118,311,55"
IsReadOnly="True"/>
        <Button x:Name="button_retrec" Content="Retrieve Student Records"
HorizontalAlignment="Left" Margin="317,61,0,0" VerticalAlignment="Top"
Width="156" Click="button_retrec_Click"/>
        <Button x:Name="button_progrec" Content="Retrieve Program Records"
HorizontalAlignment="Left" Margin="686,61,0,0" VerticalAlignment="Top"
Width="146" Click="button_progrec_Click"/>
        <Image HorizontalAlignment="Left" Height="76" Margin="20,39,0,0"
VerticalAlignment="Top" Width="196" Source="Resources\icplogo.png"/>
        <RadioButton x:Name="radbutton_sortbyname" Content="Sort by Name"
HorizontalAlignment="Left" Margin="317,98,0,0" VerticalAlignment="Top"/>
        <RadioButton x:Name="radbutton_sortbydate" Content="Sort by date"
HorizontalAlignment="Left" Margin="415,98,0,0" VerticalAlignment="Top"/>
        <Canvas Margin="844,158,47,143" Name="PieCanvas"/>
        <RadioButton x:Name="radbutton_appdev" Content="Application
Development" HorizontalAlignment="Left" Margin="135,324,0,0"
VerticalAlignment="Top" Width="163"/>
        <RadioButton x:Name="radbutton_advdat" Content="Advance
Database" HorizontalAlignment="Left" Margin="135,344,0,0"
VerticalAlignment="Top" Width="150"/>
        <RadioButton x:Name="radbutton_artint" Content="Artificial Intelligence"
HorizontalAlignment="Left" Margin="135,364,0,0" VerticalAlignment="Top"
Width="150"/>
        <TextBlock x:Name="tbox1" HorizontalAlignment="Left"
Margin="896,397,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Foreground="#FF164CDA"/>
        <TextBlock x:Name="tbox2" HorizontalAlignment="Left"
Margin="896,420,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Foreground="#FFDE1F1F"/>
        <TextBlock x:Name="tbox3" HorizontalAlignment="Left"
Margin="896,441,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
Foreground="#FF4DD31B"/>
```

```
    <TextBlock x:Name="tblock_title" HorizontalAlignment="Left"
Margin="847,124,0,0" TextWrapping="Wrap" VerticalAlignment="Top"
FontStyle="Italic" FontFamily="Arial" FontSize="14"/>
  </Grid>
</Window>
```

- Mainwindow.xaml.cs

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

namespace Application_Development_CW1
{
    /// <summary>
    /// Interaction logic for MainWindow.xaml
    /// </summary>
    public partial class MainWindow : Window
    {
        public MainWindow()
        {
            InitializeComponent();

            if (!Login.isLoggedIn)//login windows is called for authentification
purpose.
            {
```

```
        Login login = new Login();

        login.Show();

        this.Hide();

    }



    }


    public  void  clearField()//function  to  clear  textfields,datagrid  and
radiobuttons.
    {

        tbox_id.Text = "";

        tbox_name.Text = "";

        tbox_address.Text = "";

        tbox_contact.Text = "";

        tbox_email.Text = "";

        tbox_regdate.Text = "";

        datagrid_1.ItemsSource = "";

        radbutton_advdat.IsChecked = false;

        radbutton_appdev.IsChecked = false;

        radbutton_artint.IsChecked = false;

        radbutton_sortbydate.IsChecked = false;

        radbutton_sortbyname.IsChecked = false;


    }


    private void button_clear_Click(object sender, RoutedEventArgs e)
    {
        clearField();
    }
    //function  to  add  student  record  on  csv  file  with  several  input
parameters.
    private  void  addRecord(string id, string name, string address, string
contactno, string email, string enrolprogram, string enroldate)
```

```
{
    StringBuilder csvcontent = new StringBuilder();
    csvcontent.AppendLine(id + "," + name + "," + address + "," +
contactno + "," + email + "," + enrolprogram + "," + enroldate);
    string csvfilepath = "StudentRecord.csv";
    File.AppendAllText(csvfilepath, csvcontent.ToString());
}


private void button_addrec_Click(object sender, RoutedEventArgs e)
{

    if (tbox_id.Text.Trim() == "" || tbox_name.Text.Trim() == "" ||
tbox_address.Text.Trim() == "" || tbox_contact.Text.Trim() == "" ||
tbox_email.Text.Trim() == "" || tbox_regdate.Text.Trim() == "")
    {
        MessageBox.Show("Please enter all the details.");
    }
    else
    {
        string program = "";
        if (radbutton_advdat.IsChecked == true) { program = "Advance
Database"; }
        else if (radbutton_appdev.IsChecked == true) { program =
"Application Development"; }
        else if (radbutton_artint.IsChecked == true) { program = "Artificial
Intelligence"; }
        addRecord(tbox_id.Text, tbox_name.Text, tbox_address.Text,
tbox_contact.Text, tbox_email.Text, program, tbox_regdate.Text);
        MessageBox.Show("The record was added succesfully");
        clearField();
    }
}
    private void retrieveStudentRecord()//function to retrieve student
record for displaying in datagrid.
```

```csharp
        {
            string delimiter = ",";
            string tableName = "Student Record";
            String FileName = "StudentRecord.csv";

            DataSet dataset = new DataSet();
            StreamReader sr = new StreamReader(FileName);

            dataset.Tables.Add(tableName);
            dataset.Tables[tableName].Columns.Add("ID");
            dataset.Tables[tableName].Columns.Add("Name");
            dataset.Tables[tableName].Columns.Add("Address");
            dataset.Tables[tableName].Columns.Add("Contact No");
            dataset.Tables[tableName].Columns.Add("Email");
            dataset.Tables[tableName].Columns.Add("Enrolled Program");
            dataset.Tables[tableName].Columns.Add("Registration Date");

            string allData = sr.ReadToEnd();
            string[] rows = allData.Split("\r".ToCharArray());

            foreach (string r in rows)
            {
                string[] items = r.Split(delimiter.ToCharArray());
                dataset.Tables[tableName].Rows.Add(items);

            }
            this.datagrid_1.ItemsSource = dataset.Tables[0].DefaultView;
        }

        private void button_retrec_Click(object sender, RoutedEventArgs e)
        {
            if (radbutton_sortbyname.IsChecked == true)
            {
                retrieveStudentRecord();
```

```csharp
        Sort("Name");
    }
    else if (radbutton_sortbydate.IsChecked == true)
    {
        retrieveStudentRecord();
        Sort("Registration Date");
    }
    else
    {
        retrieveStudentRecord();
    }
}
private void getProgramRecord()//function to retrieve program record
with no of students on each.
{
    var temp = File.ReadAllLines("StudentRecord.csv");
    int advdat = 0, artint = 0, appdev = 0;
    foreach (string line in temp)
    {
        var delimitedLine = line.Split(',');
        if (delimitedLine[5] == "Advance Database")
        {
            advdat++;
        }
        else if (delimitedLine[5] == "Artificial Intelligence")
        {
            artint++;
        }
        else if (delimitedLine[5] == "Application Development")
        {
            appdev++;
        }

    }
```

```
DataTable dt = new DataTable();
dt.Columns.AddRange(new          DataColumn[2]          {new
DataColumn("Program", typeof(String)),
new DataColumn("No of Students",typeof(int))});
dt.Rows.Add("Application Development", appdev);
dt.Rows.Add("Artificial Intelligence", artint);
dt.Rows.Add("Advance Database", advdat);
this.datagrid_1.ItemsSource = dt.DefaultView;
int total = advdat + artint + appdev;
drawPie((advdat * 360 / total), Brushes.Red, advdat);//drawpie
```
function is called with angle as an input along with respective program.
```
drawPie((artint * 360 / total), Brushes.Green, artint);
drawPie((appdev * 360 / total), Brushes.Blue, appdev);
}


Point lastArcPoint = new Point(9999, 9999);
int lastAngle = 0;


private    void    drawPie(int    angle,    SolidColorBrush    color,int
```
prog)//function to draw pie chart with input angle along with color.
```
{
    int curentSliceAngle = angle;
    lastAngle += angle;
    angle = lastAngle;
    PieCanvas.Width = 200;
    PieCanvas.Height = 200;
    Double midPoint = 100.0;


    System.Windows.Shapes.Path          path          =          new
System.Windows.Shapes.Path();
    path.Fill = color;
    path.Stroke = color;
    PathGeometry pathGeometry = new PathGeometry();
    PathFigure pathFigure = new PathFigure();
```

```
        pathFigure.StartPoint = new Point(midPoint, midPoint);
        pathFigure.IsClosed = true;
        Double radius = midPoint;

        LineSegment lineSegment = new LineSegment(new Point(radius +
midPoint, midPoint), true);
        if (lastArcPoint.X != 9999 && lastArcPoint.Y != 9999)
        {
            lineSegment = new LineSegment(new Point(lastArcPoint.X,
lastArcPoint.Y), true);
        }
        pathFigure.Segments.Add(lineSegment);
        ArcSegment arcSegment = new ArcSegment();
        arcSegment.Point = new Point(midPoint + Math.Cos(angle *
Math.PI / 180) * radius, midPoint + Math.Sin(angle * Math.PI / 180) * radius);
        lastArcPoint         =         new         Point(arcSegment.Point.X,
arcSegment.Point.Y);
        arcSegment.Size = new Size(radius, radius);
        arcSegment.SweepDirection = SweepDirection.Clockwise;
        pathFigure.Segments.Add(arcSegment);
        pathGeometry.Figures.Add(pathFigure);
        path.Data = pathGeometry;
        PieCanvas.Children.Add(path);
        Point labelPoint = new Point(midPoint + Math.Cos((angle -
curentSliceAngle / 2) * Math.PI / 180) * radius * 0.8, midPoint +
Math.Sin((angle - curentSliceAngle / 2) * Math.PI / 180) * radius * 0.8);
        drawText(labelPoint.X - 7, labelPoint.Y - 7, prog.ToString());

    }

    private void drawText(double x, double y, string text)//function to draw
text for pie chart.
    {
```

```
TextBlock textBlock = new TextBlock();

textBlock.Text = text;

textBlock.Foreground = Brushes.White;

Canvas.SetLeft(textBlock, x);
Canvas.SetTop(textBlock, y);

PieCanvas.Children.Add(textBlock);

}

private void button_progrec_Click(object sender, RoutedEventArgs e)
{
    getProgramRecord();
    tbox1.Text = "Application Dvelopment";
    tbox2.Text = "Advance Database";
    tbox3.Text = "Artificial Intelligence";
    tblock_title.Text = "Pie Chart based on Program record.";
}

private void Sort(string sortBy)//function to sort the datagrid base on
given parameter.
{
    ICollectionView                dataView                =
CollectionViewSource.GetDefaultView(datagrid_1.ItemsSource);

    dataView.SortDescriptions.Clear();
    SortDescription    sd    =    new    SortDescription(sortBy,
ListSortDirection.Ascending);
    dataView.SortDescriptions.Add(sd);
    dataView.Refresh();
}
```

```csharp
public void readUploadRecord()
{
    OpenFileDialog openFileDialog = new OpenFileDialog();
    if (openFileDialog.ShowDialog() == true)
    {
        string[] raw_text = System.IO.File.ReadAllLines(openFileDialog.FileName);
        string[] data_col = null;
        int x = 0;
        DataTable dataTable = new DataTable();
        foreach (string text_line in raw_text)
        {
            data_col = text_line.Split(",");
            if (x == 0)

                for (int i = 0; i <= data_col.Count() - 1; i++)
                {
                    dataTable.Columns.Add(data_col[i]);
                    //MessageBox.Show(data_col[i]);
                }
            else
            {
                dataTable.Rows.Add(data_col);
                addRecord(data_col[0], data_col[1], data_col[2], data_col[3], data_col[4], data_col[5], data_col[6]);
            }

            x++;

        }
        this.datagrid_1.ItemsSource = dataTable.DefaultView;
        MessageBox.Show("The record was updated successfully.");

    }
```

```
        }
        private void btnOpenFile_Click(object sender, RoutedEventArgs e)
        {
            readUploadRecord();
        }
    }
}
```