

ABC Education Institute Student Management System



Name- D.L.Chandeepea Janith Peiris

Registration Number – NUG/A-009260

Batch No – 324

Acknowledgement

I would like to express my special thanks of gratitude to my teacher Miss.Ravini who gave me this golden opportunity to do this project of “ABC education institute student management system”. I came to know about so many new things from this project and I am really thankful to them. Secondly, I would also like to thank my parents and friends who helped me a lot in finalizing this project within the limited time frame.

Content

Acknowledgement.....	2
1. Introduction.....	4
a) Company Overview	
b) Existing system with drawbacks	
c) Propose system	
2. Diagrams.....	5
a) ER Diagram	
b) Use Case Diagram	
3. Software Development Model.....	7
4. Functional and Non-Functional Requirements.....	9
5. Interface Design and Coding.....	10
6. Test Plan.....	35
7. Software and Hardware Requirements.....	36
8. Future Improvement	37
9. Conclusion.....	38

1.0 Introduction

Student Management System is a software which is helpful for students as well as the administration authorities. This project is developed mainly to administrate the student records rather easily and comfortably than the existing system. This software of student management system can be used in various educational institutes across the globe and simplifies working of institutes.

a. Company overview

Institute name: ABC Educational Institute

Owner: Mr. J.H. Perera

Location: Nugegoda, SriLanka

b. Existing system with drawbacks

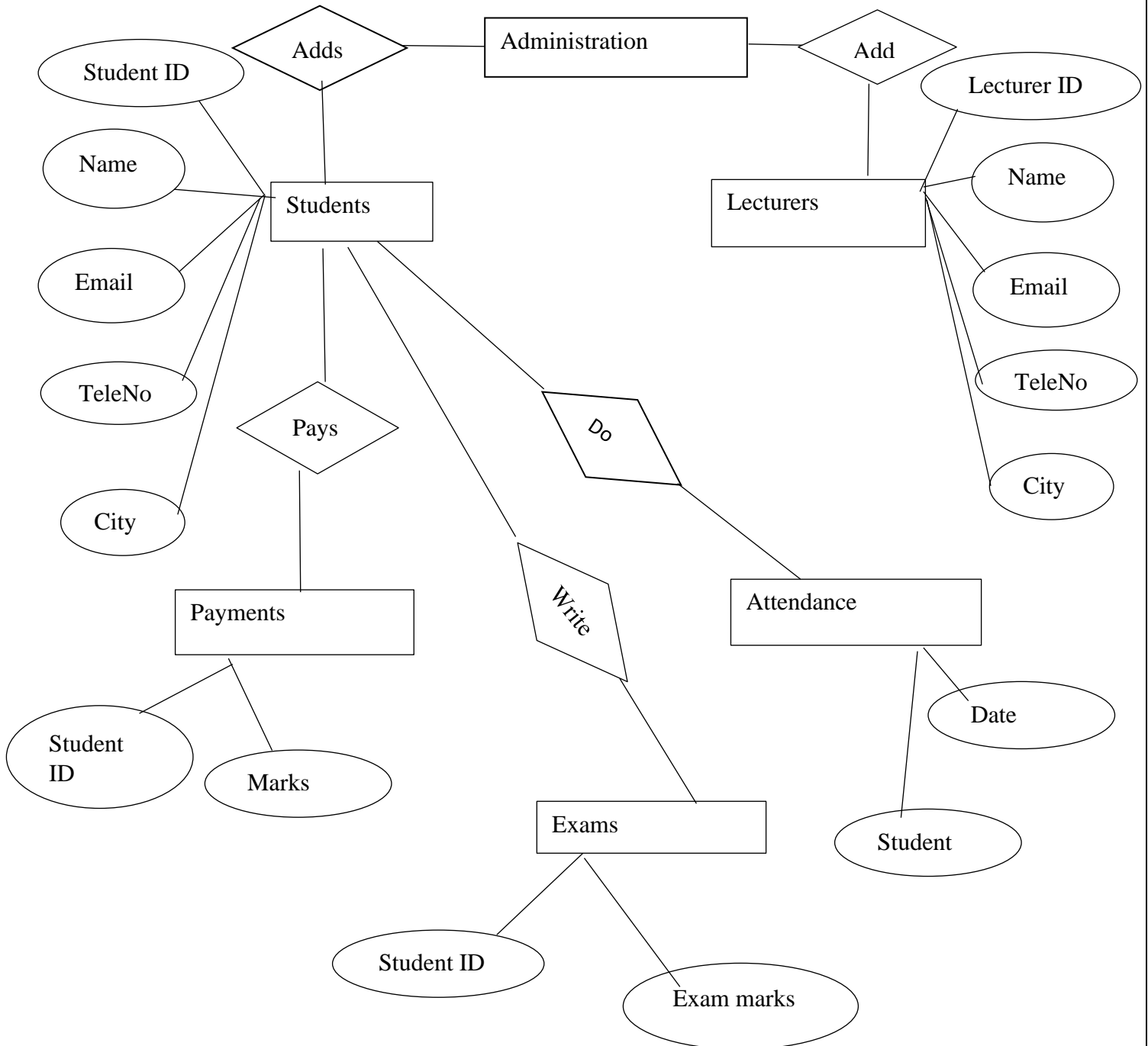
In the current system they need to keep a number of records related to the student, which is very annoying. And the administration has to enter the details of the student, enter the marks of the student, mark the attendance of students all by themselves manually. So, these kinds of things are very time consuming and costly for the institute. So computerized system will be much easier and more efficient for the institute.

c. Proposed System

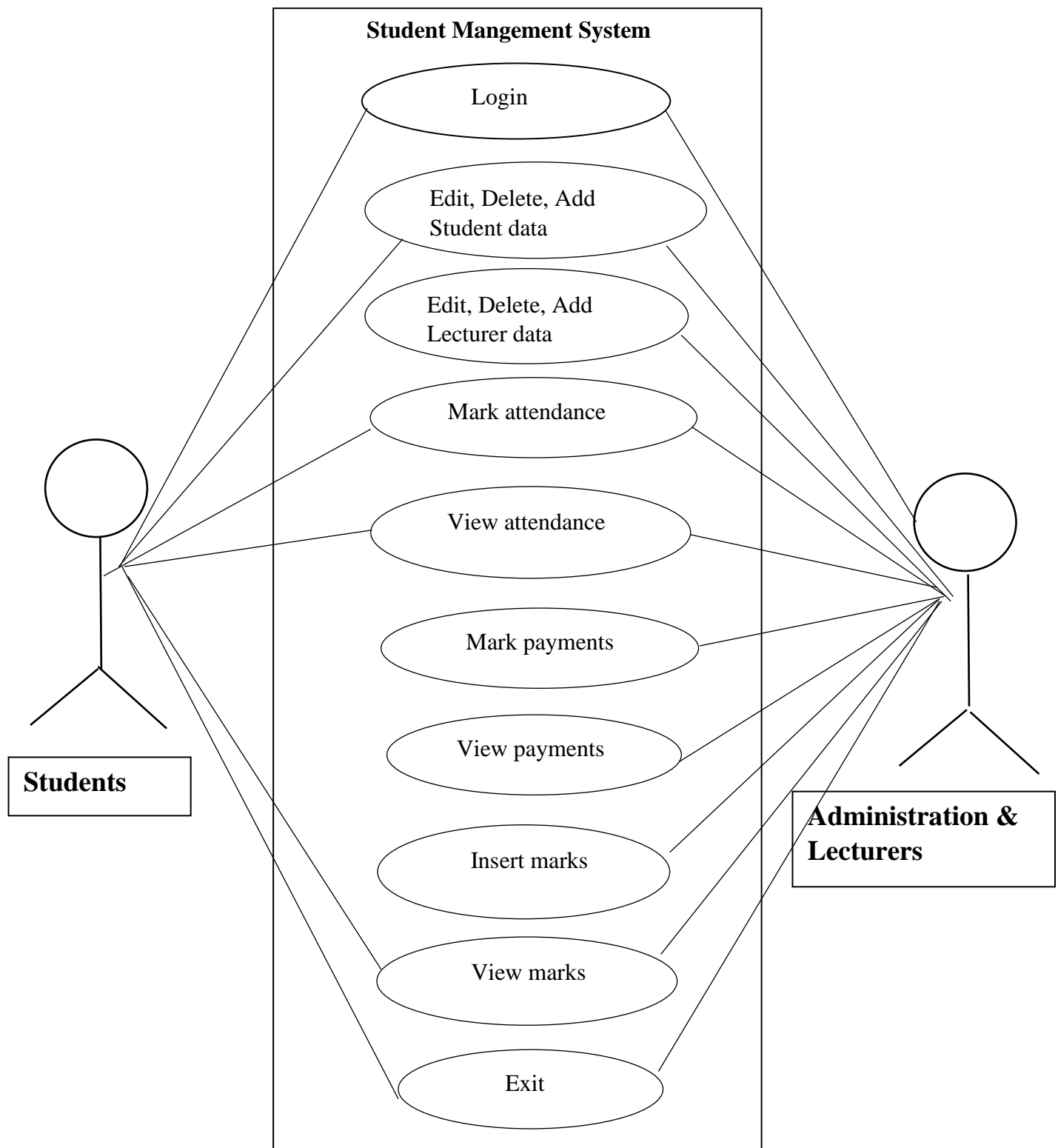
In the proposed system we have the provision for adding the details of the students by themselves. Another advantage of the system is that it is very easy to edit the details of the students and delete a student when it found unnecessary. The marks of the students are added in the database and students can view marks whenever they want. Marking attendance of students can be done by students themselves. Student payments are instantly marked when they done the payments. So like that our new system can be very effective to this institute.

Diagrams

a) ER Diagram

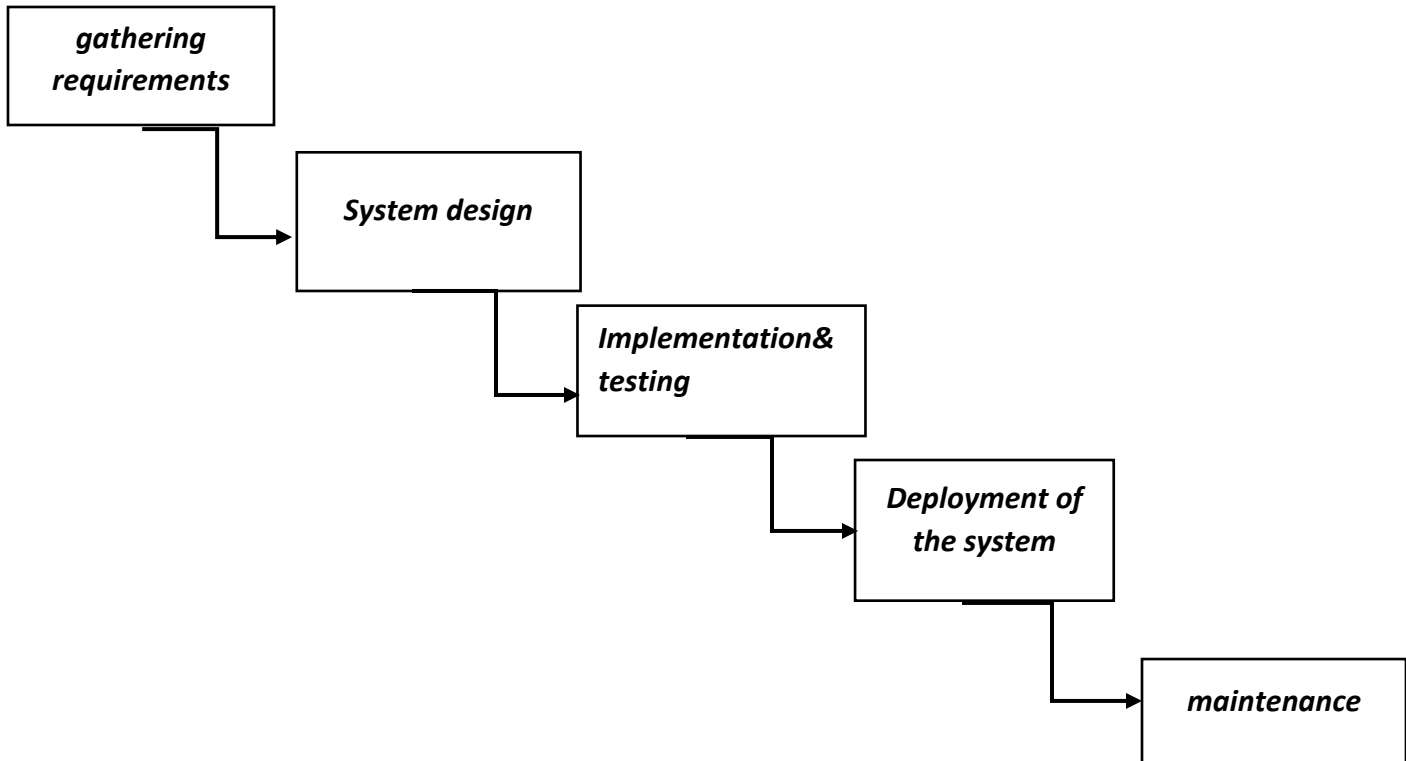


b) Use Case Diagram



3.0 Software development model

- We used waterfall model in order to develop the software.



A. Gathering Requirements

As to begin the development of the software we have to gather requirements. we gathered data and information from several methods. Analyzing documents of the institute, Interview students and administration, surveys and questionnaires, prototyping are the methods we used to gather information. So, we could get thorough understanding what kind of a system that they are expecting us to build.

B. System design

After gathering the information about the system, the next step is to design the system. In order to design the system, we are using the "VISUAL STUDIO.NET" software as the user interface "C#" as the programming language and a local database create inside visual studio.

C. implementation & testing

Implementation is the stage where the theoretical design is turned into a working system. Successful completion of the implementation phrase should comprise system installation and training of end users on the system. So, after installing system to the ABC institute's computer network, we have to train the administration and working staff how to work with this new implemented system.

Testing is the process of evaluating a system or its components with the intent to find whether it satisfies the specified requirements or not. So, from this we can find out what are the errors, missing requirements etc.

D. Deployment of the system

In this phrase, the software is deployed into a live environment (client's server) in order to test its performance. Once the software is deployed, it becomes available to end users which are administration of the institute. This is the stage the institute's administration and working staff start using the new system.

E. Maintenance

After the deployment phrase the next step is to provide support and maintenance for the software, making sure it runs smoothly. If the client and end-users come across errors, defects, bugs etc. during use we have to modify the system in order to correct faults and improve performance of the system.

4.0 Functional and Non-Functional Requirements

Functional Requirements

1. Student Information management
2. Lecturer Information management
3. Student Exam marks management
4. Student attendance management
5. Student payment management

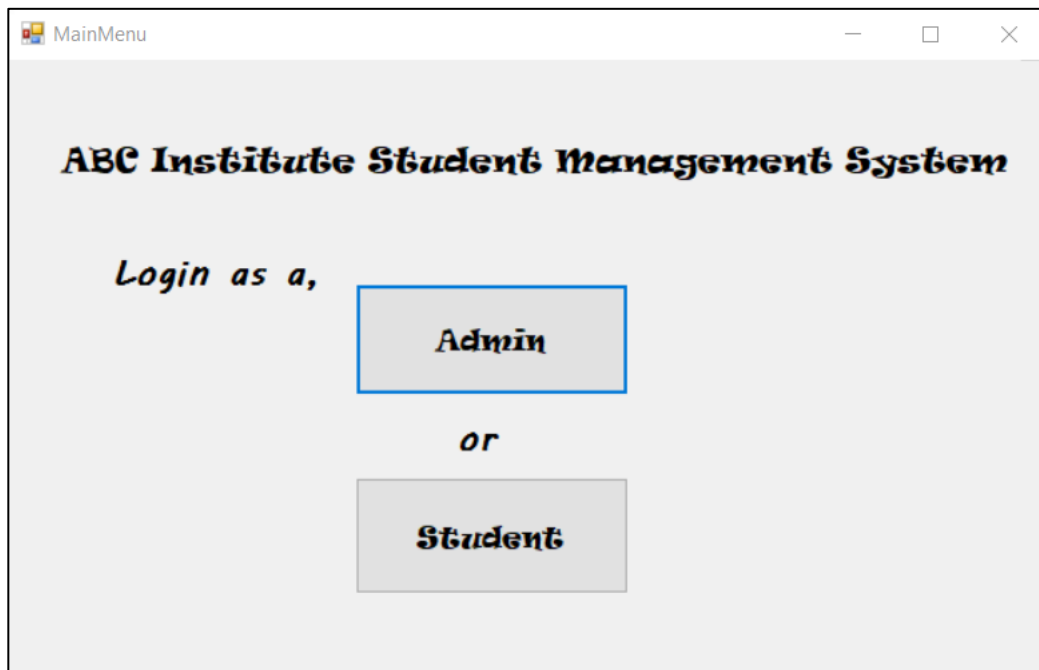
Non-Functional Requirements

1. User friendly environment
2. Maintainability
3. Accessibility
4. Durability

Interface Designing & Coding

When the user open the application, you will get “Main Menu” form.

Main Menu will view as below.



Here the user can login as,

- 1) Admin
- or
- 2) Student.

Main menu coding

```
1  using System;
2  using System.Collections.Generic;
3  using System.ComponentModel;
4  using System.Data;
5  using System.Drawing;
6  using System.Linq;
7  using System.Text;
8  using System.Threading.Tasks;
9  using System.Windows.Forms;
10
11 namespace Student_Management_System_for_ABC_Institute
12 {
13     9 references
14     public partial class MainMenu : Form
15     {
16         4 references
17         public MainMenu()
18         {
19             InitializeComponent();
20         }
21
22         1 reference
23         private void button1_Click(object sender, EventArgs e)
24         {
25             this.Hide();
26             AdminLogin obj1 = new AdminLogin();
27             obj1.Show();
28         }
29
30         1 reference
31         private void button2_Click(object sender, EventArgs e)
32         {
33             this.Hide();
34             StudentPage obj1 = new StudentPage();
35             obj1.Show();
36         }
37     }
38 }
```

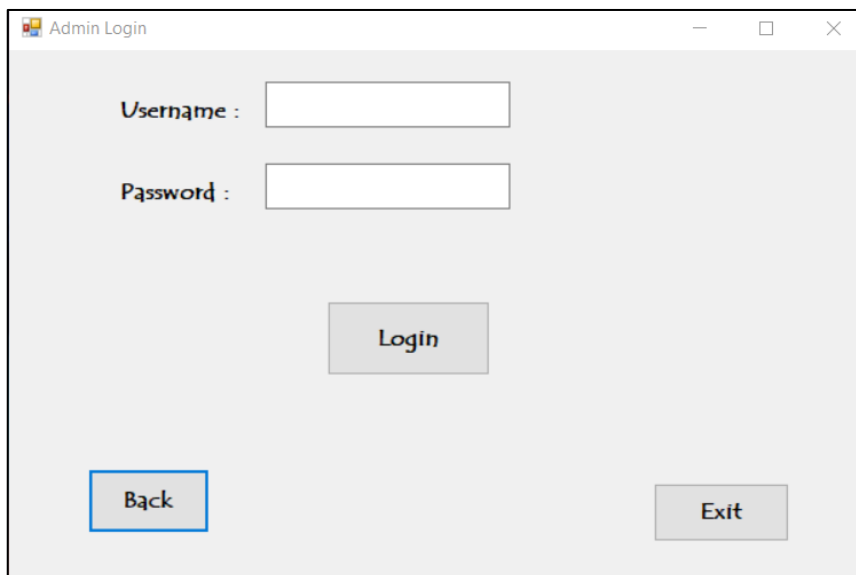
1. Admin Button

After clicking Admin button, user automatically go to Admin login form.

Admin Login Page

By clicking the admin button, user can access to the admin Login Page.

Admin Login Page will view as below.

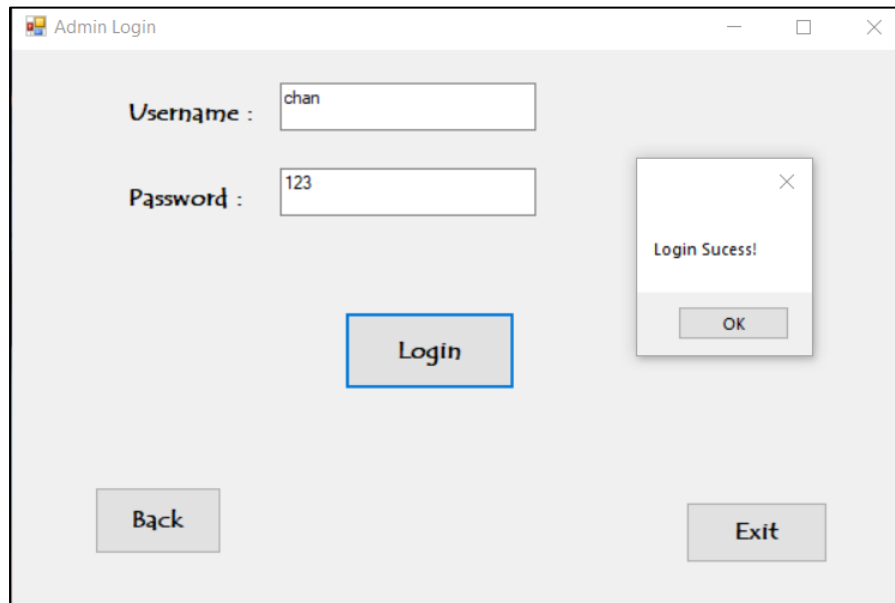


You can see two textboxes in Admin Login Page. As to Login to Admin Page user will have to enter correct username and password.

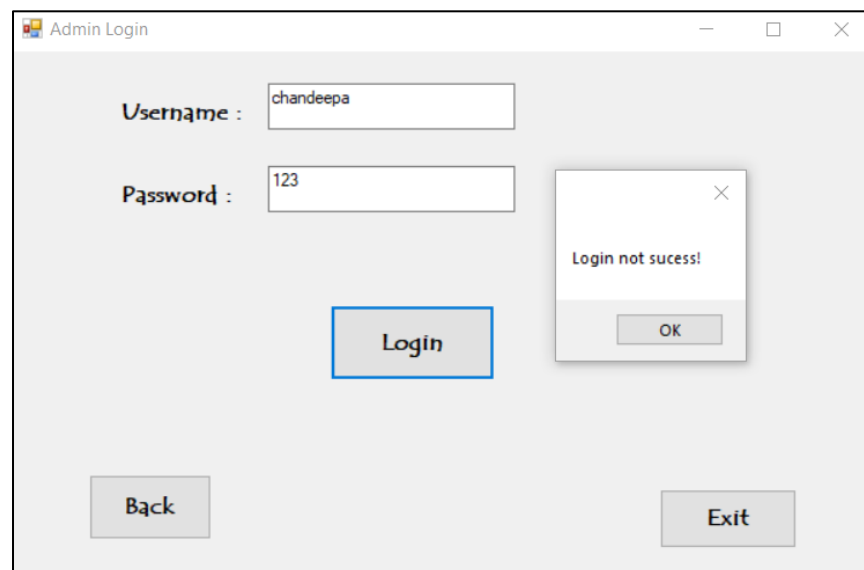
Username= “chan”

Password= “123”

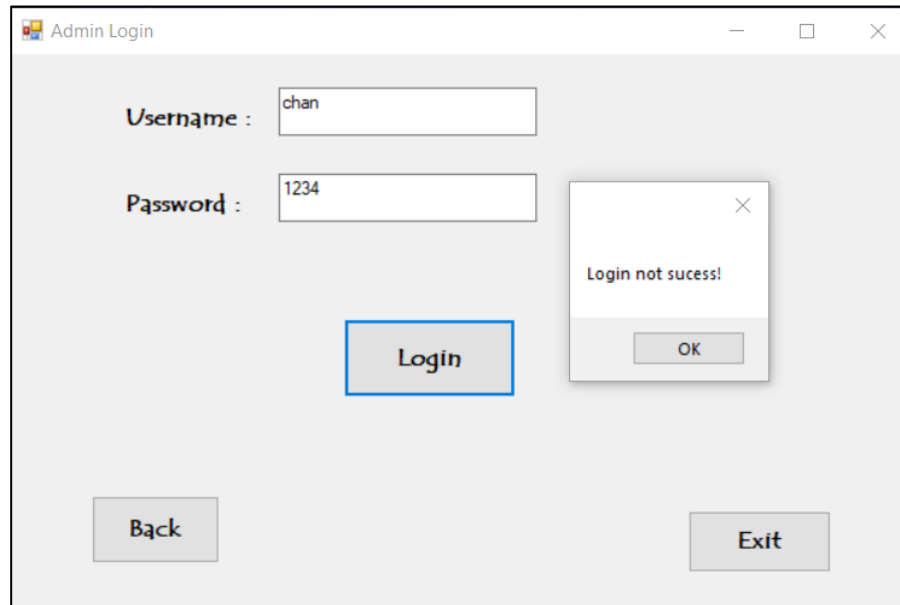
- If the user enter correct username & password user can log into the Admin Page.
Then after clicking Login button, you can get a message box that shows the login is successful. And now user can access to the Admin Page.



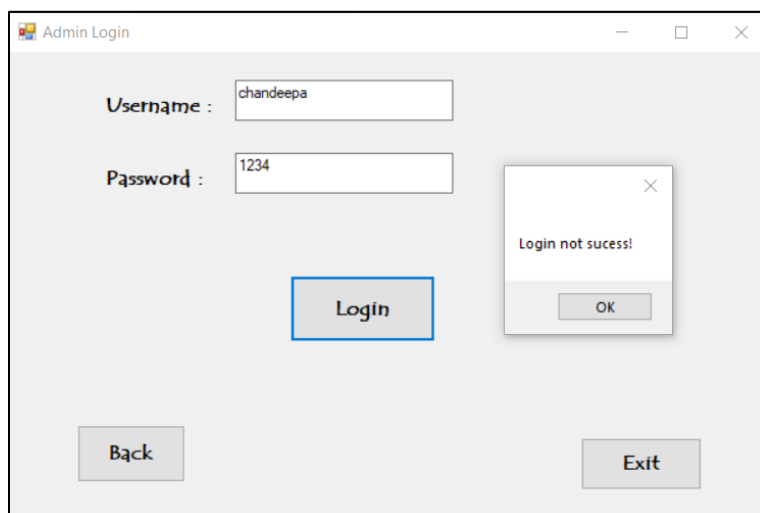
- If the entered username is incorrect user cannot log into the Admin Page. When you click Login button you will get a message box that show the login is not successful.



- If the entered Password is incorrect user cannot log into the Admin Page. When you click Login button you will get a message box that shows login is not successful.



- If the entered Username and Password both are incorrect user cannot log into the Admin Page. When you click Login button you will get a message box that shows login is not successful.



So, in order to have the access to the Admin page the user must enter the correct username & password properly. Otherwise, the user cannot access to the Admin page.

Login button coding

```
1 reference
private void button2_Click(object sender, EventArgs e)
{
    string username = txtusername.Text;
    string password = txtpassword.Text;

    if (username == "chan" && password == "123")
    {
        MessageBox.Show("Login Sucess!");
        this.Hide();
        AdminPage obj1 = new AdminPage();
        obj1.Show();
    }
    else
    {
        MessageBox.Show("Login not sucess!");
    }
}
```

Back button coding

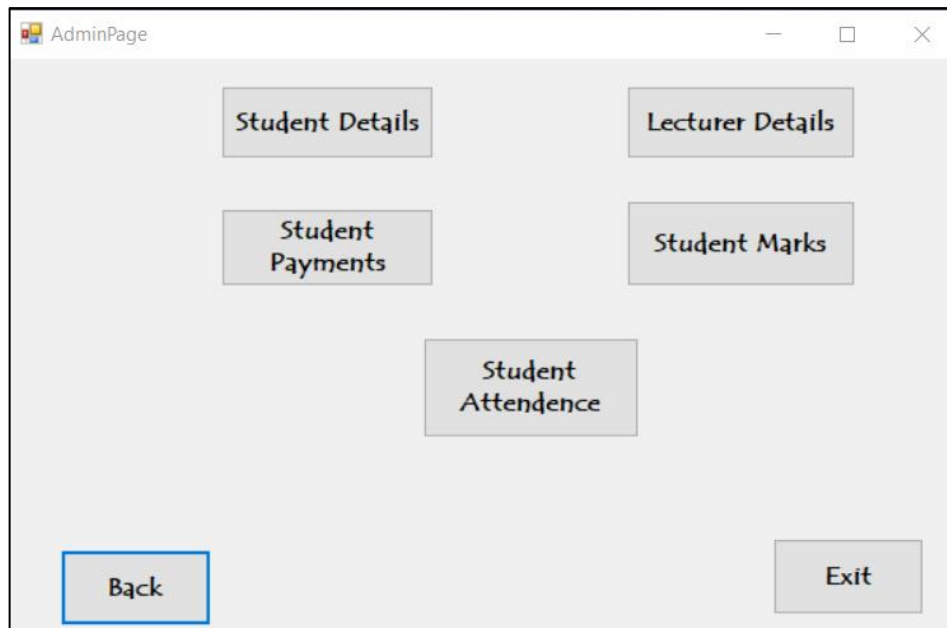
```
19
20 1 reference
21 private void button1_Click(object sender, EventArgs e)
22 {
23     MainMenu b = new MainMenu();
24     this.Hide();
25     b.Show();
26 }
```

Exit button coding

```
5
6
7 1 reference
8 private void button3_Click(object sender, EventArgs e)
9 {
10     Application.Exit();
11 }
12 }
```

After entering correct username & password user can access into the Admin Page.

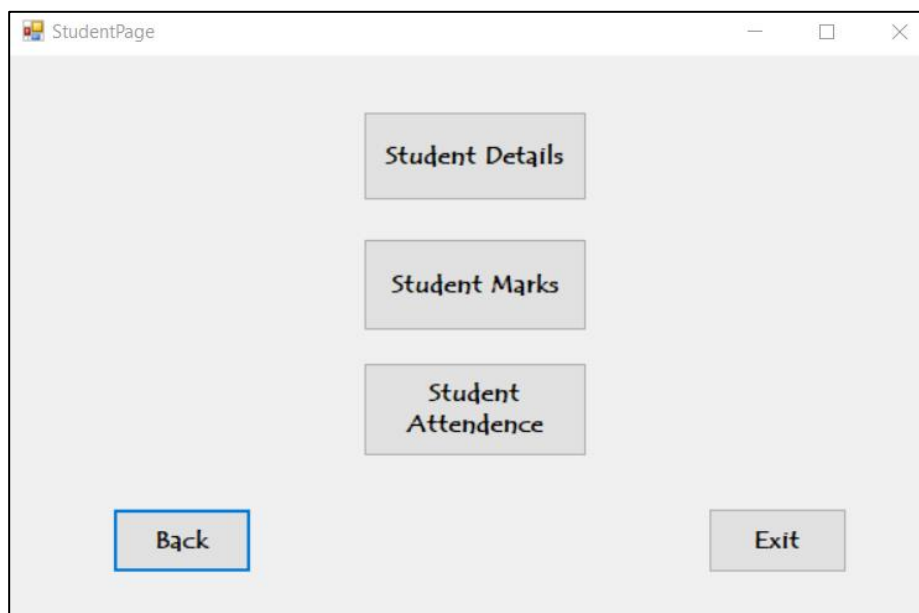
Admin Page will view as below.



2. Student Button

After clicking the student button in Main Menu user can access into the Student Page.

Student page will view as below.



Before discussing about Admin page and Student page, let's look at the database used in this system

In this student management system, we used one database which consists of 5 tables.

Those tables are,

- 1) Student Details table
- 2) Student Marks table
- 3) Student Payments table
- 4) Student Attendance table
- 5) Lecturer Details table

Student Details table designing & coding

The screenshot displays the SQL Server Enterprise Designer interface for the 'StudentDetails' table. The top pane shows the table design with columns: StudentID, Name, Email, TeleNo, and City. The bottom pane shows the T-SQL code for creating the table.

Name	Data Type	Allow Nulls	Default
StudentID	varchar(50)	<input type="checkbox"/>	
Name	varchar(50)	<input type="checkbox"/>	
Email	varchar(50)	<input checked="" type="checkbox"/>	
TeleNo	varchar(50)	<input type="checkbox"/>	
City	varchar(50)	<input type="checkbox"/>	

Keys (1)
<unnamed> (Primary Key, Clustered: StudentID)

Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

```
CREATE TABLE [dbo].[StudentDetails] (  
    [StudentID] VARCHAR (50) NOT NULL,  
    [Name] VARCHAR (50) NOT NULL,  
    [Email] VARCHAR (50) NULL,  
    [TeleNo] VARCHAR (50) NOT NULL,  
    [City] VARCHAR (50) NOT NULL,  
    PRIMARY KEY CLUSTERED ([StudentID] ASC)  
);
```

Student Marks table designing & coding

Update Script File: dbo.StudentMarks.sql

Name	Data Type	Allow Nulls	Default
StudentID	varchar(50)	<input type="checkbox"/>	
Exam 1	int	<input checked="" type="checkbox"/>	
Exam 2	int	<input checked="" type="checkbox"/>	
Exam 3	int	<input checked="" type="checkbox"/>	
Exam 4	int	<input checked="" type="checkbox"/>	
Exam 5	int	<input checked="" type="checkbox"/>	
		<input type="checkbox"/>	

Keys (1)
 <unnamed> (Primary Key, Clustered: StudentID)

Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

Design T-SQL

```
CREATE TABLE [dbo].[StudentMarks] (
    [StudentID] VARCHAR (50) NOT NULL,
    [Exam 1] INT NULL,
    [Exam 2] INT NULL,
    [Exam 3] INT NULL,
    [Exam 4] INT NULL,
    [Exam 5] INT NULL,
    PRIMARY KEY CLUSTERED ([StudentID] ASC)
);
```

Student Payments table designing & coding

Update Script File: dbo.StudentPayments.sql

Name	Data Type	Allow Nulls	Default
StudentID	varchar(50)	<input type="checkbox"/>	
Jan	bit	<input checked="" type="checkbox"/>	
Feb	bit	<input checked="" type="checkbox"/>	
March	bit	<input checked="" type="checkbox"/>	
April	bit	<input checked="" type="checkbox"/>	
May	bit	<input checked="" type="checkbox"/>	
June	bit	<input checked="" type="checkbox"/>	
July	bit	<input checked="" type="checkbox"/>	
August	bit	<input checked="" type="checkbox"/>	
September	bit	<input checked="" type="checkbox"/>	
October	bit	<input checked="" type="checkbox"/>	
November	bit	<input checked="" type="checkbox"/>	
December	bit	<input checked="" type="checkbox"/>	

Keys (1)
 <unnamed> (Primary Key, Clustered: StudentID)

Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

Design T-SQL

```
CREATE TABLE [dbo].[StudentPayments] (
    [StudentID] VARCHAR (50) NOT NULL,
    [Jan] BIT NULL,
    [Feb] BIT NULL,
    [March] BIT NULL,
    [April] BIT NULL,
    [May] BIT NULL,
    [June] BIT NULL,
    [July] BIT NULL,
    [August] BIT NULL,
    [September] BIT NULL,
    [October] BIT NULL,
    [November] BIT NULL,
    [December] BIT NULL,
    PRIMARY KEY CLUSTERED ([StudentID] ASC)
);
```

Student Attendance table designing & coding

Update Script File: dbo.StudentAttendance.sql

Name	Data Type	Allow Nulls	Default
StudentID	varchar(50)	<input type="checkbox"/>	
Day 1	bit	<input checked="" type="checkbox"/>	
Day 2	bit	<input checked="" type="checkbox"/>	
Day 3	bit	<input checked="" type="checkbox"/>	
Day 4	bit	<input checked="" type="checkbox"/>	
Day 5	bit	<input checked="" type="checkbox"/>	
Day 6	bit	<input checked="" type="checkbox"/>	
Day 7	bit	<input checked="" type="checkbox"/>	
Day 8	bit	<input checked="" type="checkbox"/>	
Day 9	bit	<input checked="" type="checkbox"/>	
Day 10	bit	<input checked="" type="checkbox"/>	
Day 11	bit	<input checked="" type="checkbox"/>	
Day 12	bit	<input checked="" type="checkbox"/>	
Day 13	bit	<input checked="" type="checkbox"/>	
Day 14	bit	<input checked="" type="checkbox"/>	
Day 15	bit	<input checked="" type="checkbox"/>	

Keys (1)
 <unnamed> (Primary Key, Clustered: StudentID)

Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

Design T-SQL

```
CREATE TABLE [dbo].[StudentAttendance] (
  [StudentID] VARCHAR (50) NOT NULL,
  [Day 1] BIT NULL,
  [Day 2] BIT NULL,
  [Day 3] BIT NULL,
  [Day 4] BIT NULL,
  [Day 5] BIT NULL,
  [Day 6] BIT NULL,
  [Day 7] BIT NULL,
  [Day 8] BIT NULL,
  [Day 9] BIT NULL,
  [Day 10] BIT NULL,
  [Day 11] BIT NULL,
  [Day 12] BIT NULL,
  [Day 13] BIT NULL,
  [Day 14] BIT NULL,
  [Day 15] BIT NULL
);
```

Lecturer Details table designing & coding

Update Script File: dbo.LecturerDetails.sql

Name	Data Type	Allow Nulls	Default
LecturerID	varchar(50)	<input type="checkbox"/>	
Name	varchar(50)	<input type="checkbox"/>	
Email	varchar(50)	<input checked="" type="checkbox"/>	
TeleNo	int	<input type="checkbox"/>	
City	varchar(50)	<input type="checkbox"/>	

Keys (1)
 <unnamed> (Primary Key, Clustered: LecturerID)

Check Constraints (0)
Indexes (0)
Foreign Keys (0)
Triggers (0)

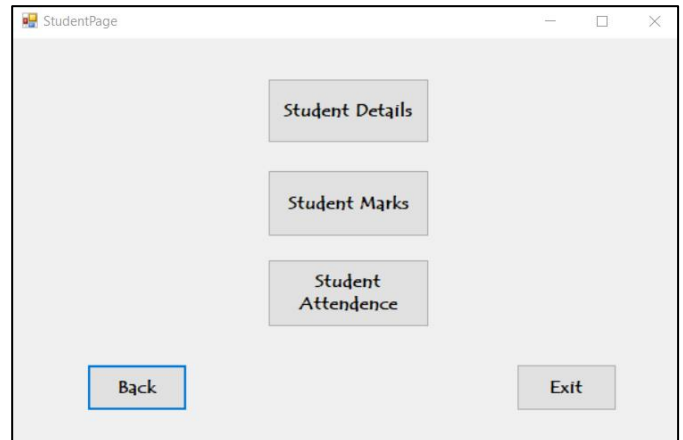
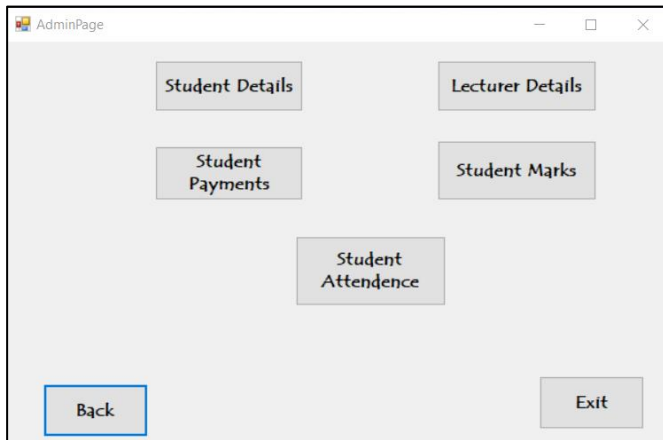
Design T-SQL

```
CREATE TABLE [dbo].[LecturerDetails] (
  [LecturerID] VARCHAR (50) NOT NULL,
  [Name] VARCHAR (50) NOT NULL,
  [Email] VARCHAR (50) NULL,
  [TeleNo] INT NOT NULL,
  [City] VARCHAR (50) NOT NULL,
  PRIMARY KEY CLUSTERED ([LecturerID] ASC)
);
```

So now let's discuss about Admin Page and Student Page.

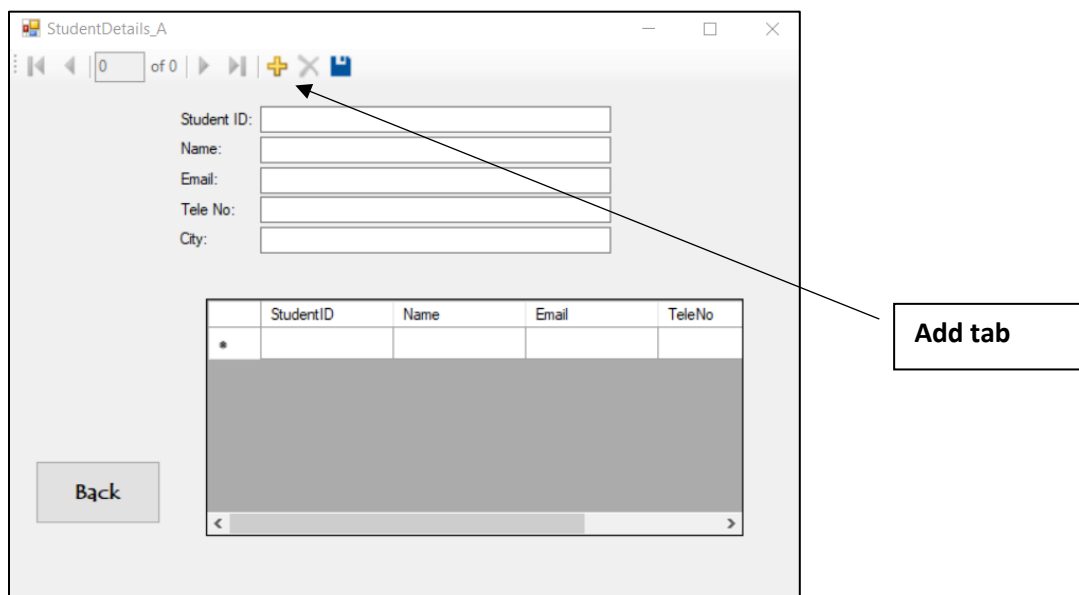
Admin Page

Student Page



Student Details Button (Admin page)

When you click student details button you will get a form shown as below.



In order to enter student details, click “**Add Tab**”. Then the form shown as below.

The screenshot shows the 'StudentDetails_A' window. At the top, there is a toolbar with three icons: a plus sign (Add Tab), a red X (Delete tab), and a blue floppy disk (Save tab). Arrows point from these icons to labels: 'Delete tab' for the plus icon, 'Click here to select the row' for the red X icon, and 'Save tab' for the floppy disk icon. Below the toolbar are five text input fields: 'Student ID:', 'Name:', 'Email:', 'Tele No:', and 'City:'. Below these fields is a table with four columns: 'StudentID', 'Name', 'Email', and 'TeleNo'. The first row of the table is highlighted in blue. A 'Back' button is located at the bottom left of the form.

StudentID	Name	Email	TeleNo

Then enter Student details in each text boxes given and click “**Save tab**” to save Details in the database.

The screenshot shows the 'StudentDetails_A' window with the same layout as the previous one. The text input fields are now filled with the following details: 'Student ID: 1S', 'Name: chandeepa', 'Email: chan@gmail.com', 'Tele No: 709388838', and 'City: colombo'. The table below has the first row filled with these details: '1S', 'chandeepa', 'chan@gmail.com', and '709388838'. The 'Back' button remains at the bottom left.

StudentID	Name	Email	TeleNo
1S	chandeepa	chan@gmail.com	709388838

After entering one student's details, to enter another student's details click **"Add tab"** and enter details in the same way. And click **"Save tab"** to save that data in database.

This is how the student details form look like after entering set of data.

Student ID: 1S
 Name: chandeepea
 Email: chan@gmail.com
 Tele No: 709388838
 City: colombo

StudentID	Name	Email	TeleNo
1S	chandeepea	chan@gmail.com	709388838
2S	kane	kane@gmail.com	70456778
3S	Will	will@gmail.com	70977376
4S	Wall	w1@gmail.com	79873774
5S	Clarke	cl1@gmail.com	70345561

Back

If the user wants to delete entered data, select that specific row and click **"Delete tab"**. After deleting that row data click **"Save tab"** to update the database.

Student ID: 1S
 Name: chandeepea
 Email: chan@gmail.com
 Tele No: 709388838
 City: colombo

StudentID	Name	Email	TeleNo
1S	chandeepea	chan@gmail.com	709388838
2S	kane	kane@gmail.com	70456778
3S	Will	will@gmail.com	70977376
4S	Wall	w1@gmail.com	79873774
5S	Clarke	cl1@gmail.com	70345561

Back

Before deleting 5th row data

Student ID: 4S
 Name: Wall
 Email: w1@gmail.com
 Tele No: 798737744
 City: Colombo

StudentID	Name	Email	TeleNo
1S	chandeepea	chan@gmail.com	709388838
2S	kane	kane@gmail.com	704567788
3S	Will	will@gmail.com	709773763
4S	Wall	w1@gmail.com	798737744

Back

After deleting 5th row data

Student Details Button (Student page)

When you click Student Details button in student page, user will get a same interface form that got from clicking Student Details button in Admin page. This button is also working the same way that the Student Details button in Admin page works.

StudentDetails_S

1 of 4

Student ID: 1S

Name: chandeepa

Email: chan@gmail.com

Tele No: 709388838

City: colombo

StudentID	Name	Email	TeleNo
1S	chandeepa	chan@gmail.com	709388838
2S	kane	kane@gmail.com	704567788
3S	Will	will@gmail.com	709773763
4S	Wall	w1@gmail.com	798737744
*			

Back

When admin inputs student data in student details form Students also can view this data. Students also can input, edit& delete data from through this form.

Student Marks Button (Admin page)

When a user click student marks button in admin page user can see a form shown as below.

StudentMarks_A

0 of 0

Student ID:

Exam 1:

Exam 2:

Exam 3:

Exam 4:

Exam 5:

StudentID	Exam 1	Exam 2	Exam 3
*			

Back

Add tab

In order to enter student marks, click “**Add Tab**”.

Then the form shown as below.

The screenshot shows the 'StudentMarks_A' application window. At the top, there is a toolbar with navigation icons and three action buttons: a yellow plus sign (Add Tab), a red X (Delete Tab), and a blue floppy disk (Save Tab). Below the toolbar, there is a 'Student ID:' label followed by an empty text box. To the right, there are five vertically stacked text boxes labeled 'Exam 1:', 'Exam 2:', 'Exam 3:', 'Exam 4:', and 'Exam 5:'. Below these is a table with four columns: 'StudentID', 'Exam 1', 'Exam 2', and 'Exam 3'. The first row of the table is highlighted in blue. A 'Back' button is located at the bottom left. Three callout boxes with arrows point to the 'Save tab' button, the 'Delete tab' button, and the first row of the table.

Save tab

Delete tab

Click here to select the row

Back

Then enter the correct Student ID and Student Marks in each text boxes given and click “**Save tab**” to save Details in the database

The screenshot shows the 'StudentMarks_A' application window after data entry. The 'Student ID' text box now contains '1S'. The five exam text boxes contain the values: '43', '77', '56', '67', and '90'. The table below has the first row filled with these values: '1S' in the 'StudentID' column, '43' in 'Exam 1', '77' in 'Exam 2', and '56' in 'Exam 3'. The 'Back' button remains at the bottom left.

Back

After entering one student's marks, to enter another student's marks click **"Add tab"** and enter details in the same way. And click **"Save tab"** to save that data in database.

This is how the student details form look like after entering set of data

StudentMarks_A

1 of 5

Student ID: 1S

Exam 1: 43
Exam 2: 77
Exam 3: 56
Exam 4: 67
Exam 5: 90

StudentID	Exam 1	Exam 2	Exam 3
1S	43	77	56
2S	68	89	90
3S	45	34	36
4S	56	77	54
5S	45	56	67

Back

If the user wants to delete entered data, select that specific row and click **"Delete tab"**. After deleting that row data click **"Save tab"** to update the database.

StudentMarks_A

1 of 5

Student ID: 1S

Exam 1: 43
Exam 2: 77
Exam 3: 56
Exam 4: 67
Exam 5: 90

StudentID	Exam 1	Exam 2	Exam 3
1S	43	77	56
2S	68	89	90
3S	45	34	36
4S	56	77	54
5S	45	56	67

Back

Before deleting 5th row data

StudentMarks_A

4 of 4

Student ID: 4S

Exam 1: 56
Exam 2: 77
Exam 3: 54
Exam 4: 35
Exam 5: 44

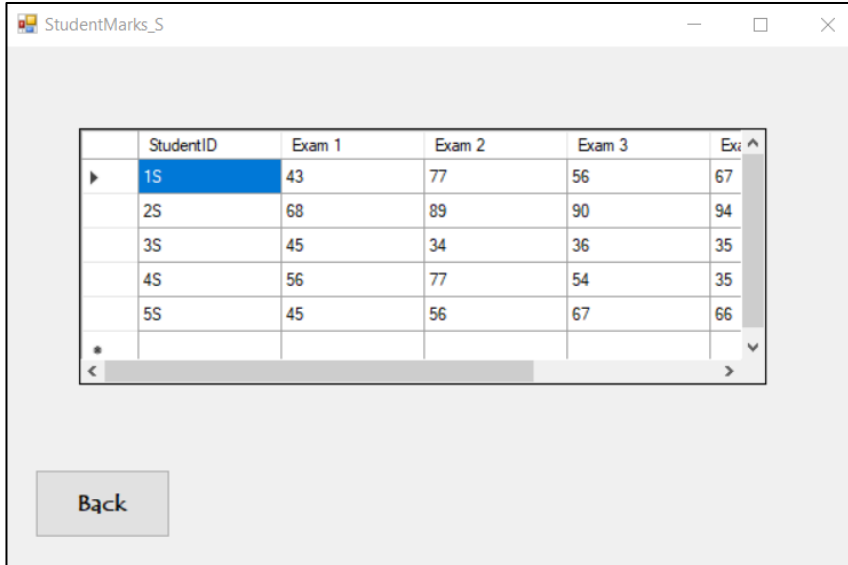
StudentID	Exam 1	Exam 2	Exam 3
1S	43	77	56
2S	68	89	90
3S	45	34	36
4S	56	77	54

Back

After deleting 5th row data

Student Marks Button (Student page)

When a user click student marks button in student page user can see a form shown as below.



The screenshot shows a window titled "StudentMarks_S". Inside, there is a table with the following data:

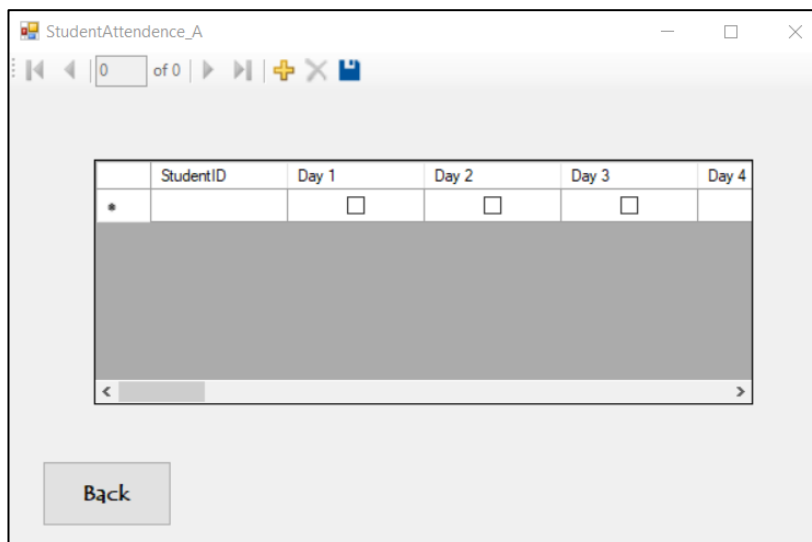
	StudentID	Exam 1	Exam 2	Exam 3	Exi ^
▶	1S	43	77	56	67
	2S	68	89	90	94
	3S	45	34	36	35
	4S	56	77	54	35
	5S	45	56	67	66
*					

Below the table is a "Back" button.

Here students cannot change marks. That can be only done by administration through admin page. So, students can only read their marks.

Student Attendance Button (Admin page)

When a user click student attendance button in admin page user can see a form shown as below.



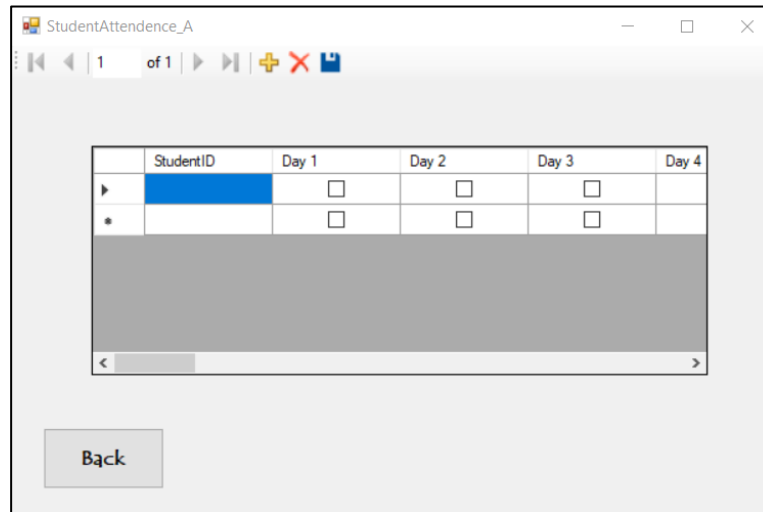
The screenshot shows a window titled "StudentAttendance_A". At the top, there is a navigation bar with "0 of 0" and icons for navigation and actions. Below this is a table with the following data:

	StudentID	Day 1	Day 2	Day 3	Day 4
*		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Below the table is a "Back" button.

In order to enter student marks, click **“Add Tab”**.

Then the form shown as below.

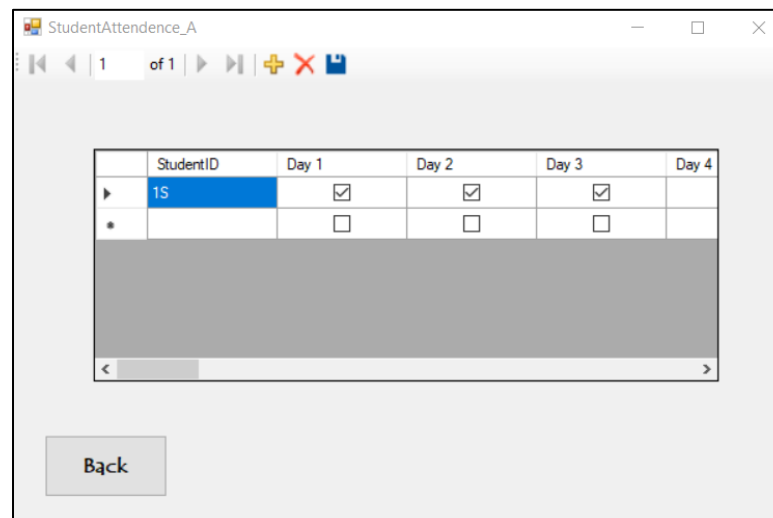


The screenshot shows a window titled "StudentAttendance_A". Inside, there is a table with the following structure:

	StudentID	Day 1	Day 2	Day 3	Day 4
▶		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
*		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Below the table is a greyed-out area and a "Back" button.

Then mark Student attendance in each check boxes given and click **“Save tab”** to save Details in the database.



The screenshot shows the same window "StudentAttendance_A". The table now has the following data:

	StudentID	Day 1	Day 2	Day 3	Day 4
▶	1S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
*		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

The "Back" button is still present at the bottom.

After entering one student’s attendance, to enter another student’s attendance click **“Add tab”** and enter details in the same way. And click **“Save tab”** to save that data in database.

This is how the student details form look like after entering set of data.

StudentID	Day 1	Day 2	Day 3	Day 4
1S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2S	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
3S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5S	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Back

If the user wants to delete entered data, select that specific row and click **“Delete tab”**. After deleting that row data click **“Save tab”** to update the database.

StudentID	Day 1	Day 2	Day 3	Day 4
1S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2S	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
3S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
5S	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Back

Before deleting 5th row data

After deleting 5th row data

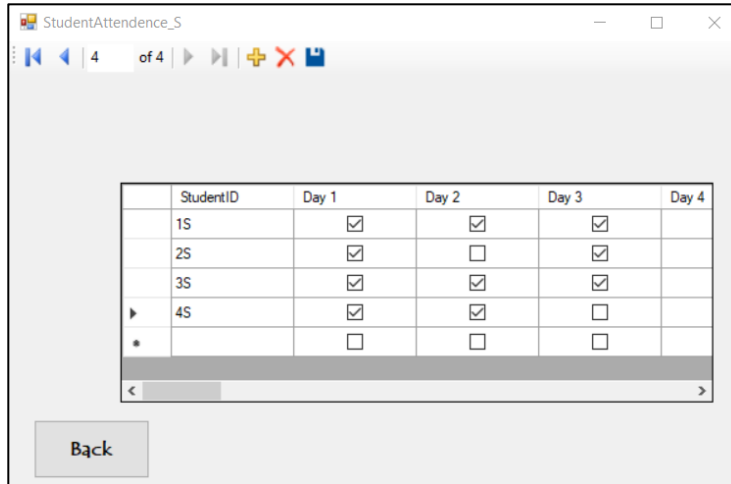
StudentID	Day 1	Day 2	Day 3	Day 4
1S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2S	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
3S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Back

Student Attendance Button (Student page)

When you click Student Attendance button in student page, user will get the same interface form that got from clicking Student Attendance button in Admin page. This button is also working same way Student Attendance button in Admin page works.

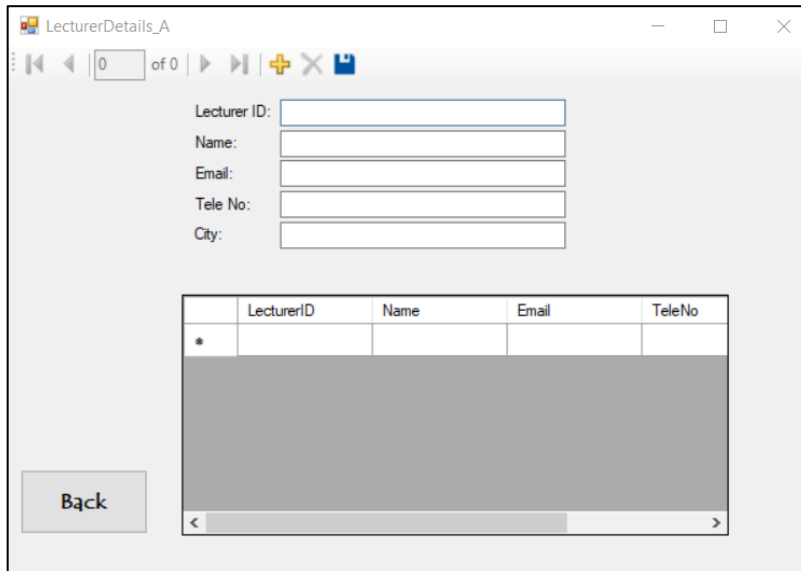
This is how Student Attendance form view after adjustments made by the administration.



StudentID	Day 1	Day 2	Day 3	Day 4
1S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
2S	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
3S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
4S	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	
*	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

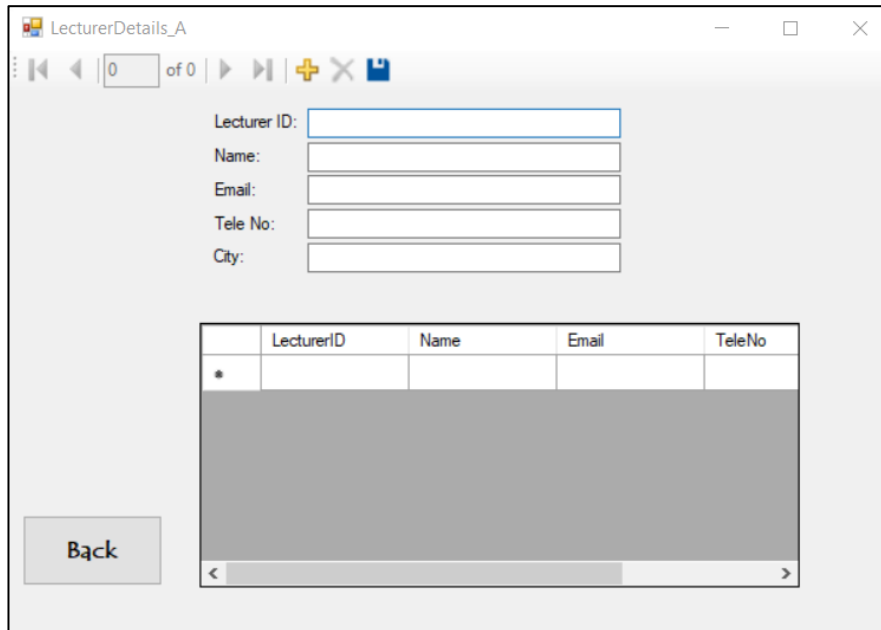
Lecturer Details Button (Admin page)

When a user click Lecturer details button in student page user can see a form shown as below.



LecturerID	Name	Email	TeleNo
*			

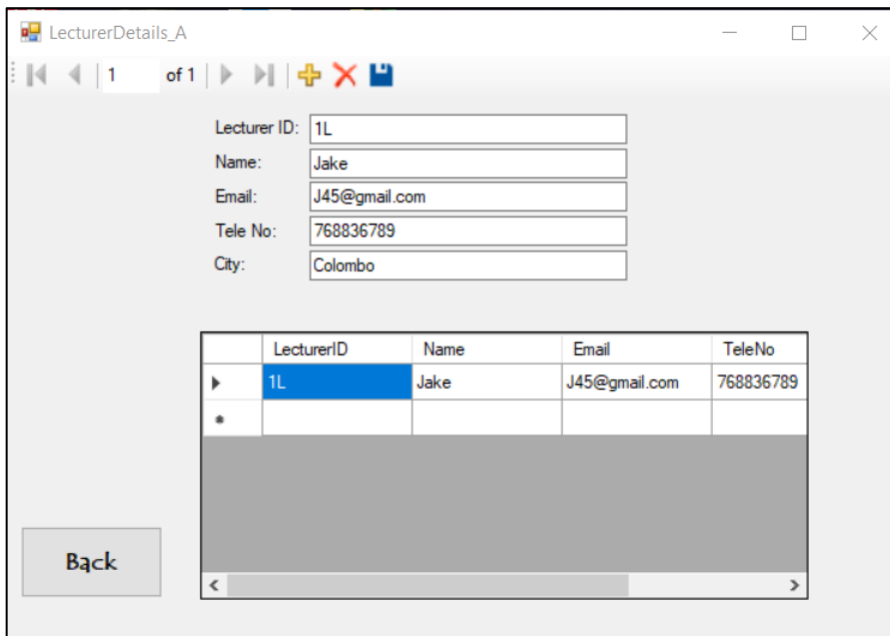
In order to enter lecturer details, click “**Add Tab**”. Then the form shown as below.



The screenshot shows a window titled "LecturerDetails_A" with a toolbar at the top containing navigation icons and a status bar showing "0 of 0". Below the toolbar are five text input fields labeled "Lecturer ID:", "Name:", "Email:", "Tele No:", and "City:". Below these fields is a table with four columns: "LecturerID", "Name", "Email", and "TeleNo". The table has one row with a "*" in the first column and empty cells for the others. A "Back" button is located at the bottom left of the form.

	LecturerID	Name	Email	TeleNo
*				

Then enter Lecturer details in each text boxes given and click “**Save tab**” to save Details in the database.



The screenshot shows the same window "LecturerDetails_A" but now the input fields are filled with data: "Lecturer ID:" is "1L", "Name:" is "Jake", "Email:" is "J45@gmail.com", "Tele No:" is "768836789", and "City:" is "Colombo". The table below now has one row with the entered data: "1L", "Jake", "J45@gmail.com", and "768836789". The "Back" button remains at the bottom left.

	LecturerID	Name	Email	TeleNo
▶	1L	Jake	J45@gmail.com	768836789
*				

After entering one Lecturer's details, to enter another lecturer's details click **"Add tab"** and enter details in the same way. And click **"Save tab"** to save that data in database.

This is how the lecturer details form look like after entering set of data.

The screenshot shows a window titled 'LecturerDetails_A' with a toolbar at the top indicating '1 of 3' records. Below the toolbar are input fields for Lecturer ID, Name, Email, Tele No, and City. The table below contains three rows of data:

	LecturerID	Name	Email	TeleNo
▶	1L	Jake	J45@gmail.com	768836789
	2L	Smith	Smith@gmail.com	754333322
	3L	John	john@gmail.com	701234567

A 'Back' button is located at the bottom left of the form.

If the user wants to delete entered data, select that specific row and click **"Delete tab"**. After deleting that row data click **"Save tab"** to update the database.

This screenshot is identical to the previous one, showing the 'LecturerDetails_A' form with three rows of data. The first row (1L, Jake) is highlighted with a blue background, indicating it is selected.

Before deleting 3rd row data

The screenshot shows the 'LecturerDetails_A' form after the third row has been deleted. The toolbar now indicates '2 of 2' records. The input fields are now populated with the data from the second row (2L, Smith):

Lecturer ID: 2L
 Name: Smith
 Email: Smith@gmail.com
 Tele No: 754333322
 City: Colombo

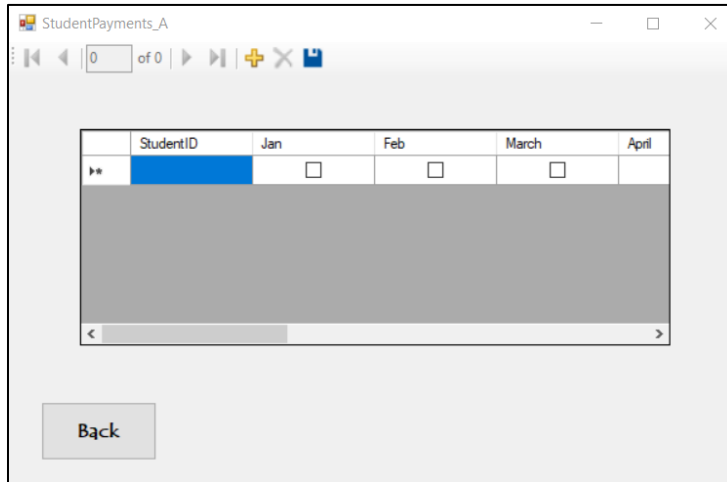
	LecturerID	Name	Email	TeleNo
	1L	Jake	J45@gmail.com	768836789
▶	2L	Smith	Smith@gmail.com	754333322

A 'Back' button is located at the bottom left of the form.

After deleting 3rd row data

Students Payments Button (Admin page)

When a user click student attendance button in admin page user can see a form shown as below.



The screenshot shows a web application window titled "StudentPayments_A". It features a table with the following structure:

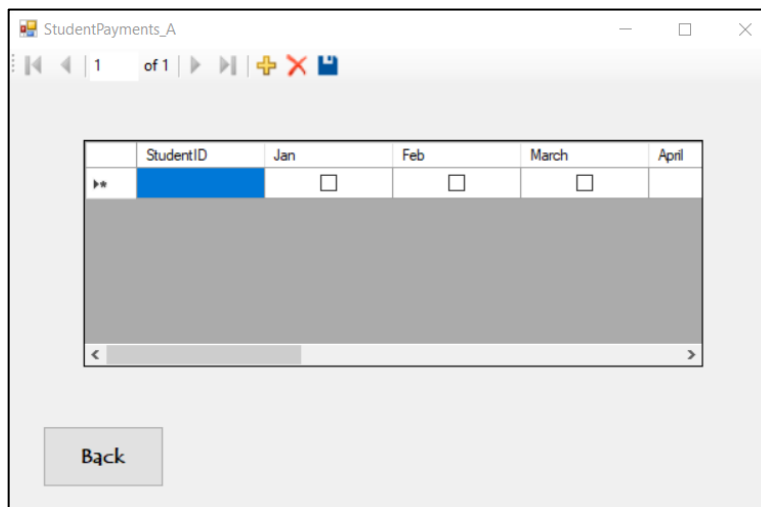
	StudentID	Jan	Feb	March	April
»		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Below the table is a large grey rectangular area, likely for additional data entry or a list of students. At the bottom left, there is a "Back" button. The top of the window includes navigation controls and a status bar showing "0 of 0".

In order to enter student payments, click “**Add Tab**”.

the form shown as below.

Then



The screenshot shows the same web application window "StudentPayments_A" after adding a new tab. The table now contains one row of data:

	StudentID	Jan	Feb	March	April
»		<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

The status bar at the top now shows "1 of 1". The "Back" button remains at the bottom left.

After entering one student’s payments, to enter another student’s payments click “**Add tab**” and enter details in the same way. And click “**Save tab**” to save that data in database.

This is how the student payments form look like after entering set of data.

StudentPayments_A

1 of 5

StudentID	Jan	Feb	March	April
S1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
S2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S4	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S5	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Back

If the user wants to delete entered data, select that specific row and click **“Delete tab”**. After deleting that row data click **“Save tab”** to update the database.

StudentPayments_A

5 of 5

Student ID: S5

Jan: ☒ Feb: ☒ March: ☐ April: ☐ May: ☒ June: ☐ July: ☐ August: ☐ September: ☐ October: ☐ November: ☐ December: ☐

StudentID	Jan	Feb	March	April
S1	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S4	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
S5	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	

Back

Before deleting 5th row data

After deleting 5th row data

StudentPayments_A

4 of 4

StudentID	Jan	Feb	March	April
S1	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input checked="" type="checkbox"/>	
S2	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S3	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	
S4	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	

Back

Back button coding in admin page buttons

```
1 reference
private void button1_Click(object sender, EventArgs e)
{
    AdminPage a = new AdminPage();
    this.Hide();
    a.Show();
}
```

Back button coding in student page buttons

```
1 reference
private void button1_Click(object sender, EventArgs e)
{
    StudentPage c = new StudentPage();
    this.Hide();
    c.Show();
}
```

6.Test Plan

a. Testing Procedure

According to time frame in the project proposal testing procedure has given 30 days.

Time	Frist Week	Second Week	Third Week	Fourth Week
Project Tasks				
Project Planning	3 days			
Project Design	3 days	3 days		
Project Coding		3 days	4 days	
Project Testing			3 days	
Project Reporting				2 days

FUNCTIONS	TIME
PLANNING	3 days
DESIGNING	6 days
CODING	7 days
TESTING	3 days
REPORTING	2 days

a. Tested Items

All the functions (Login, Insert, Update, Delete, Exit) have been tested and make sure that they are working properly and did not have any kind of errors

6. Software and Hardware Requirements

Software

1. Visual Studio 2017

- Here I used this software to build this Student Management System.
- We used c# as the programming language.
- We used a local database created inside visual studio.



2. Operating System

As the operating system we used Microsoft Windows 10.



Hardware

- Hard drive-512GB
- Processor-Intel core i7(10th gen)
- Ram-4GB
- Laptop-Dell
- Printer-Canon 1505

7.Future Improvements

Our Student Management System Provides important information for administration, students and lectures at ABC institute. This system is a very fast, effective and efficient than the previous manual student management system that they had.

We are also hoping to add new features to our system in the future too.

- Adding a student login system to each and every student
- Lecturer and Student performance evaluation method
- Student feedback receiving method

These are the some of future improvements we are hoping to do for this system.

7.Conclusion

Our project is only a humble venture to satisfy the needs in an institution. Several user-friendly coding also adopted. This package shall prove to be a powerful package in satisfying all the requirements of the organization. This new system will help students , administration and lecturers to improve their efficiency and effectiveness.