system requires staff’s authentication in which the staff will be given a username and a password as defined by the system administrator. Registered staffs only have the rights granted by the system administrator to scan and upload supporting documents of the students.

Furthermore, this system has been developed to provide a management in records. The focus of this particular system concerns electronic record keeping concepts that need to be understood in the larger context of record information management, electronic record management, and other aspects of information technology. In addition, this system is design for managing and maintaining the records.

asd

Purpose and Description

The propose system is responsible to improve the consistency of the document process electronic based system and minimize the time consumption in managing the documents.

This project is design to digitize the important documents that has been submitted by the students in the Registrar's Office. It provides the ease of storing and retrieving the documents that was stored before in the cabinet. Only the administrator can add a user that can access in the system and provides the users the username and password set by the administrator. In addition, it is develop to provide management in records.

Objectives of the study

The purpose of the study is to develop a convenient electronic based archiving management system. Specifically, it aims to:

- Provide security for the records.
- Reduce manpower and lack of storage space.
- Standardize the life cycle management of student records.
- Provide easy recovery of data in case of any natural disaster.
- Provide ease of searching lacking credentials of the students.
- Provide easy access of information.

Scope and Limitation

The Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College is intended for maintaining the records of the students where the system can store the credentials of the students in the registrar's office. The system can also provide backup in case of loss of records. Registrar's office staff can also check the records that have been submitted by the students. It can also generate reports such as printing the student credential monitoring sheet, institutional student credentials report, request letter for Form 137 and student dismissal form or honorable dismissal. Moreover, the system can generate log reports made by the users. Transactions not mentioned above are beyond the scope of this study.

CHAPTER 2

REVIEW OF RELATED LITERATURE AND STUDIES

Related literature

The information society and the paperless office are by now two timeworn metaphors of contemporary society, where information is considered as the main asset and vehicle for economic, cultural, political achievements and educational purposes, enhanced by the use of information technology (Svard, P. 2014).

Most schools and district have such a space, crammed with shelves, filing cabinets and boxes, all storing legacy district and student records that can't legally be disposed of due to state records retention law. Fortunately, school districts are adopting online student information systems, which means that much of the student record data they now gather is in digital format, reducing some of the need for archival paper storage Kipp Bently (2018).

According to George (2012), the record management ensures that institutional records of historical, fiscal, and legal value are identified and preserved and that non-essential records are discarded in a timely manner according to established guideline and identified legislation.

Scott Kimura (2015) on the other hand, concluded that managing student's academic records – both before and after they graduate – is critical function of an educational institution. Thus it is important, to be taking steps and plans to enhance the student record management process. One step that one should be taking in is investing digital student records. Here are the reasons why: 1) Nothing is ever lost. 2) Student record privacy is enhanced and 3) even centuries old student records can be digitalized.

In addition, Hwang et al (2013) concluded that most institutes consider document standardization to be very important in which 76% of institutes (19 out of 250) answered that an archiving management system was necessary. Moreover, due to the capabilities of EDMS in creating, saving, distributing, archiving, and accessing the documents by searching according to desired criteria. Moreover, Yaldir and Polat (2016) stated that applying an EDMS for university is very important.

Related Studies

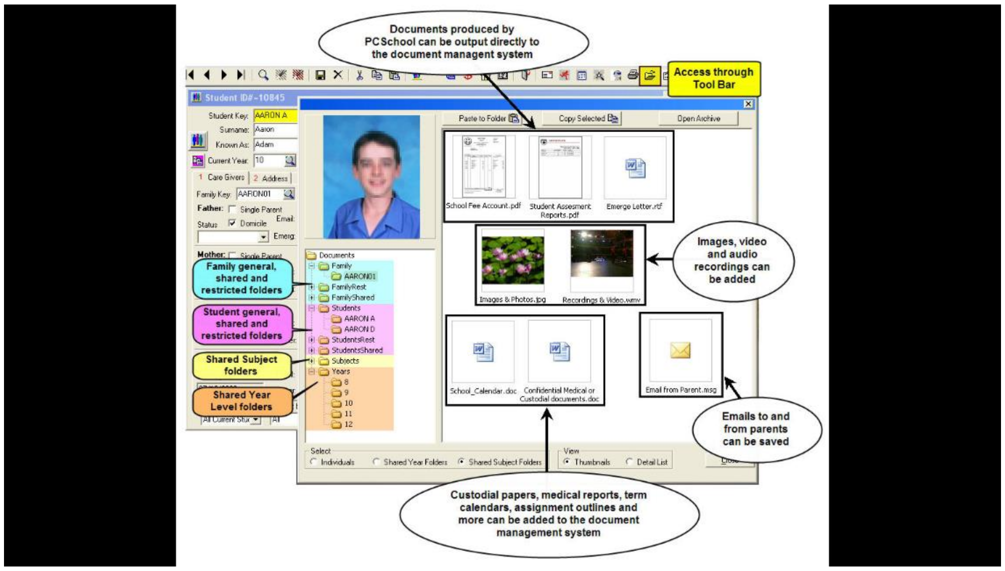
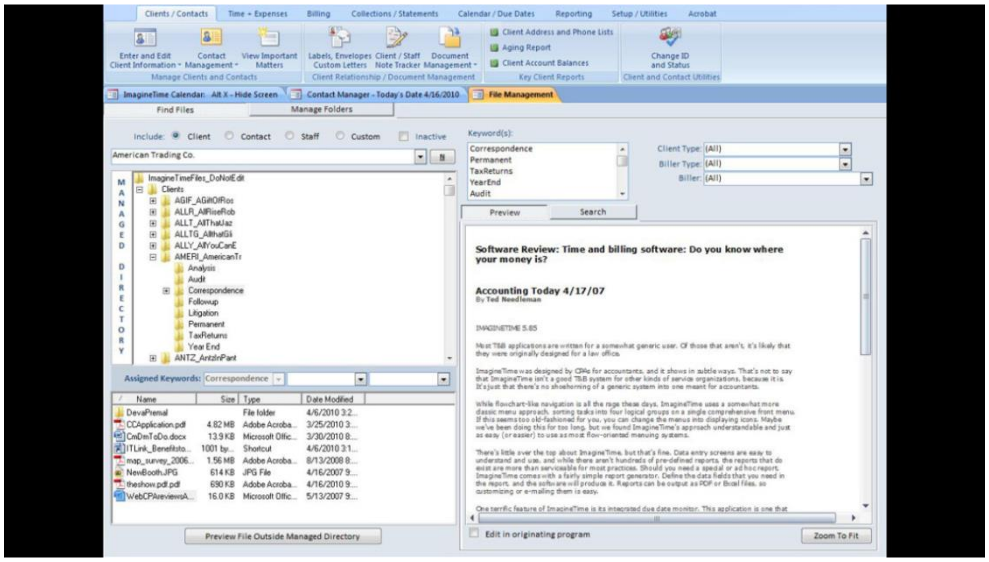

Documents produced by PCSchool can be output directly to the document managent system
Access through Tool Bar
Student ID#-10845
Student Key: AARON A
Surname: Aaron
Known As: Adam
Current Year: 10
Family Key: AARON01
Paste to Folder
Copy Selected
Open Archive
School Fee Account.pdf
Student Assesment Reports.pdf
Emerge Letter.rtf
Images & Photos.jpg
Recordings & Video.wmv
Images, video and audio recordings can be added
Family general, shared and restricted folders
Student general, shared and restricted folders
Shared Subject folders
Shared Year Level folders
School_Calendar.doc
Confidential Medical or Custodial documents.doc
Email from Parent.msg
Emails to and from parents can be saved
Select: Individuals, Shared Year Folders, Shared Subject Folders
View: Thumbnails, Detail List
Custodial papers, medical reports, term calendars, assignment outlines and more can be added to the document management system

Figure 1. PCSchool Document Management System

PCSchool Document Management System enables schools to digitally store and access student, family, staff and creditor related documents.



Name	Size	Type	Date Modified
DevaPremal		File folder	4/6/2010 3:2...
CCApplication.pdf	4.82 MB	Adobe Acroba...	3/25/2010 3:...
CmDmToDo.docx	13.9 KB	Microsoft Offic...	3/30/2010 8:...
ITLink_Benefitsto...	1001 by...	Shortcut	4/6/2010 3:1...
map_survey_2006...	1.56 MB	Adobe Acroba...	8/13/2008 8:...
NewBooth.JPG	614 KB	JPG File	4/16/2007 9:...
theshow.pdf.pdf	690 KB	Adobe Acroba...	4/16/2010 9:...
WebCPAreviewsA...	16.0 KB	Microsoft Offic...	5/13/2007 9:...

Software Review: Time and billing software: Do you know where your money is?

Accounting Today 4/17/07
By Ted Needleman

IMAGINETIME 5.85

Most T&B applications are written for a somewhat generic user. Of those that aren't, it's likely that they were originally designed for a law office.

ImagineTime was designed by CPAs for accountants, and it shows in subtle ways. That's not to say that ImagineTime isn't a good T&B system for other kinds of service organizations, because it is. It's just that there's no shoehorning of a generic system into one meant for accountants.

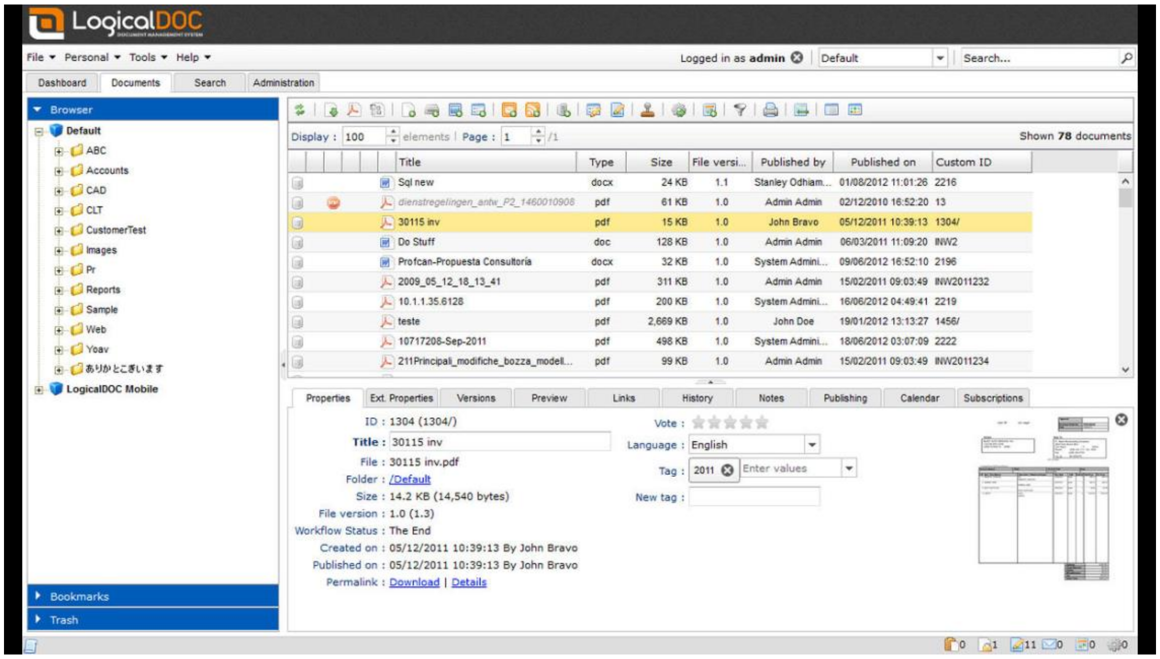
While flowchart-like navigation is all the rage these days, ImagineTime uses a somewhat more classic menu approach, sorting tasks into four logical groups on a single comprehensive front menu. If this seems too old-fashioned for you, you can change the menus into displaying icons. Maybe we've been doing this for too long, but we found ImagineTime's approach understandable and just as easy (or easier) to use as most flow-oriented menuing systems.

There's little over the top about ImagineTime, but that's fine. Data entry screens are easy to understand and use, and while there aren't hundreds of pre-defined reports, the reports that do exist are more than serviceable for most practices. Should you need a special or ad hoc report, ImagineTime comes with a fairly simple report generator. Define the data fields that you need in the report, and the software will produce it. Reports can be output as PDF or Excel files, so customizing or e-mailing them is easy.

One terrific feature of ImagineTime is its integrated due date monitor. This application is one that

Figure 2. Document Management Module

The Imagine Time Document Management System is designed to make most file management tasks quick and painless. During initial setup, Imagine Time will automate the folder structure creation process, and will allow one to move and copy ones' existing files into the manage folder structure.



Title	Type	Size	File versi...	Published by	Published on	Custom ID
Sql new	docx	24 KB	1.1	Stanley Odhiam...	01/08/2012 11:01:26	2216
dienstregelingen_antw_P2_1460010908	pdf	61 KB	1.0	Admin Admin	02/12/2010 16:52:20	13
30115 inv	pdf	15 KB	1.0	John Bravo	05/12/2011 10:39:13	1304/
Do Stuff	doc	128 KB	1.0	Admin Admin	06/03/2011 11:09:20	INW2
Profcan-Propuesta Consultoría	docx	32 KB	1.0	System Admini...	09/06/2012 16:52:10	2196
2009_05_12_18_13_41	pdf	311 KB	1.0	Admin Admin	15/02/2011 09:03:49	INW2011232
10.1.1.35.6128	pdf	200 KB	1.0	System Admini...	16/06/2012 04:49:41	2219
teste	pdf	2,669 KB	1.0	John Doe	19/01/2012 13:13:27	1456/
10717208-Sep-2011	pdf	498 KB	1.0	System Admini...	18/06/2012 03:07:09	2222
211Principali_modifiche_bozza_modell...	pdf	99 KB	1.0	Admin Admin	15/02/2011 09:03:49	INW2011234

ID : 1304 (1304/)
Title : 30115 inv
File : 30115 inv.pdf
Folder : /Default
Size : 14.2 KB (14,540 bytes)
File version : 1.0 (1.3)
Workflow Status : The End
Created on : 05/12/2011 10:39:13 By John Bravo
Published on : 05/12/2011 10:39:13 By John Bravo
Permalink : Download | Details

Vote :
Language : English
Tag : 2011
New tag :

Figure 3. LogicalDOC

LogicalDOC is a proprietary document management system that is designed to handle and share documents within an organization.

CHAPTER 3

DESIGN AND METHODOLOGY

This chapter presents the design and methodology of the proposed study, Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College. Its discussion is focused on the software specification and shows the flow of the system that relates to the process of the proposed system.

Methodology

This study involved the concerns of the staff of office registrar, as interviewed by the researchers. The data gathered helped the researcher to come up with a process and create a system entitled "Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College" in which it would help the registrar's office in maintaining the records of the students. This system as well can help the registrar's office in monitoring the lacking credentials the students need to submit to the office.

In order for the study to be realized, the researchers prepared a standardized questionnaire for the respondents to answer wherein modified questions are indicated. The respondents stated that they have various difficulties in managing the documents. It still depends on paper-based record keeping that eventually is prone to damage and as a consequence takes much time in retrieving such documents. Further, it was also revealed that the papers from these past decades are now almost damage and hard to retrieve. The respondent suggested that the paper printed documents will be digitized by scanning the paper and then store it into the system. It was found out that the respondents agreed to the current propose study on digitization of student credentials of the registrar's office. Thus, the digitization is a primary solution to the existing problem of the college.

In this study, the methods that the researchers use are the unified modeling language and the data flow diagram. Unified modeling language is a method that is intended to provide a standard way to visualize the design of the system. Data Flow Diagram is use to graphically represent the flow of data through an information system, modeling its process. A data flow diagram (DFD) illustrates how data is process by a system in terms of inputs and outputs. As its name indicates, its focus is on the flow of information, where data comes from, where it goes and how it gets stored. This method is used by researchers to envision the interactions between the user and the system and also to pinpoint the flow of the data in the system.

Further, the researchers also use the waterfall model as one of the method of the study. It goes with a series of steps. First is the requirement analysis to fully understand the problem. Next step is designing a technical solution to the problems carried out by the system requirements, including scenarios, layouts and data models. After the designing, will be the implementation where the researchers identify the error of the system. When the identified errors are solved, it will be verified and can create the accuracy of the system. And the last step will be the maintenance of the system. The figure below will illustrate the series of steps the researchers used.

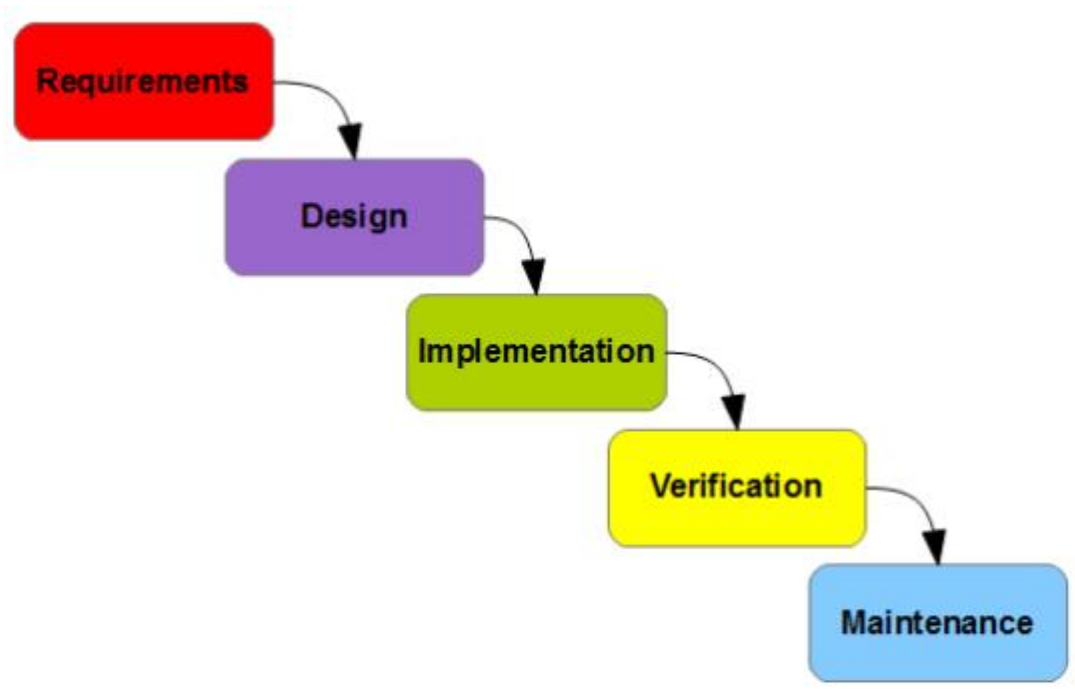
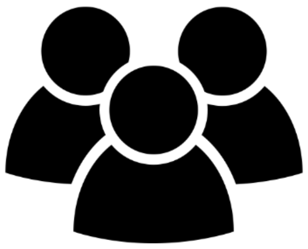


Figure 4. Waterfall Model

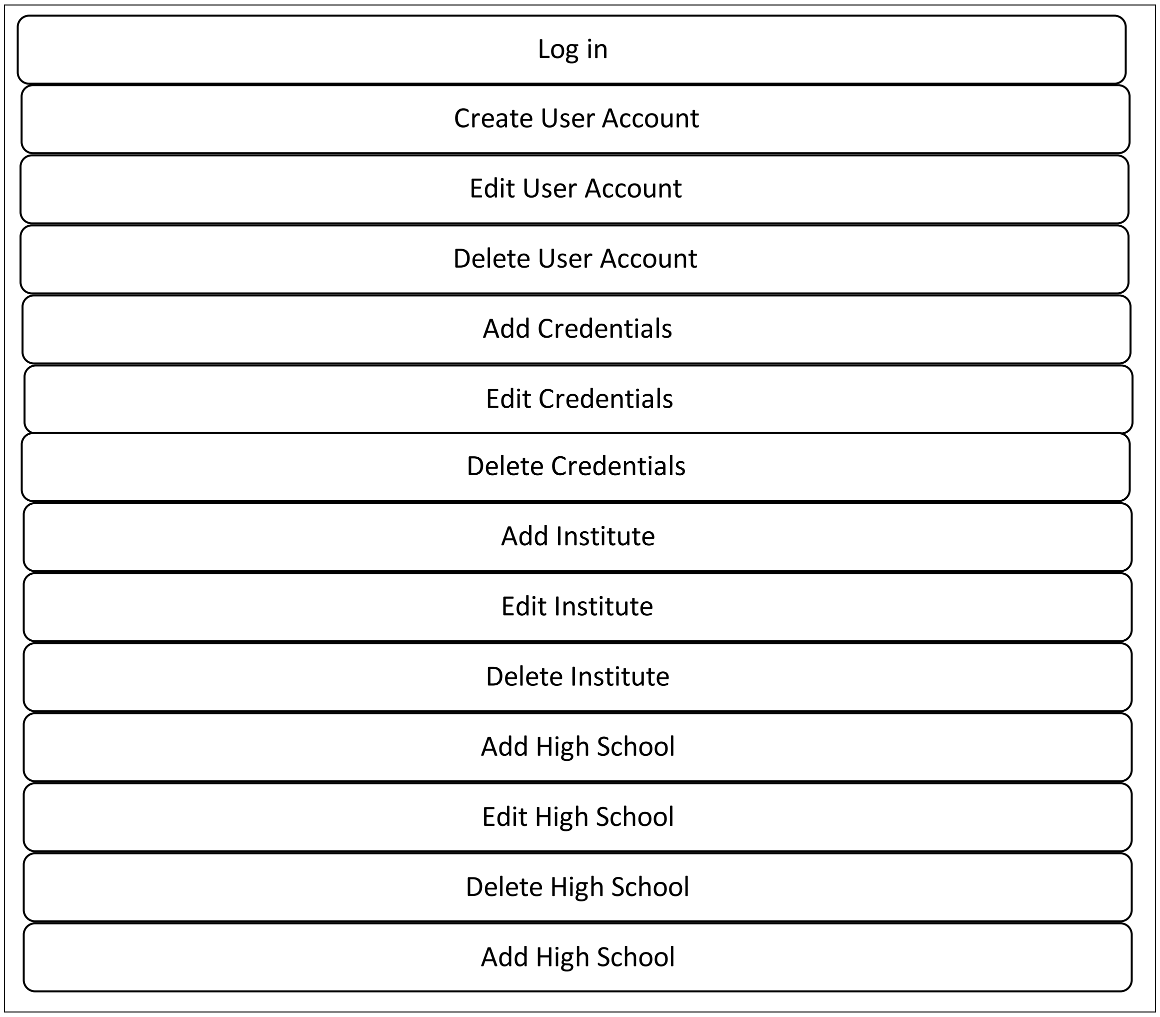
Use Case



- ❖ Administrator
- ❖ Staff

Actors

Figure 5. Actors



Log in

Create User Account

Edit User Account

Delete User Account

Add Credentials

Edit Credentials

Delete Credentials

Add Institute

Edit Institute

Delete Institute

Add High School

Edit High School

Delete High School

Add High School

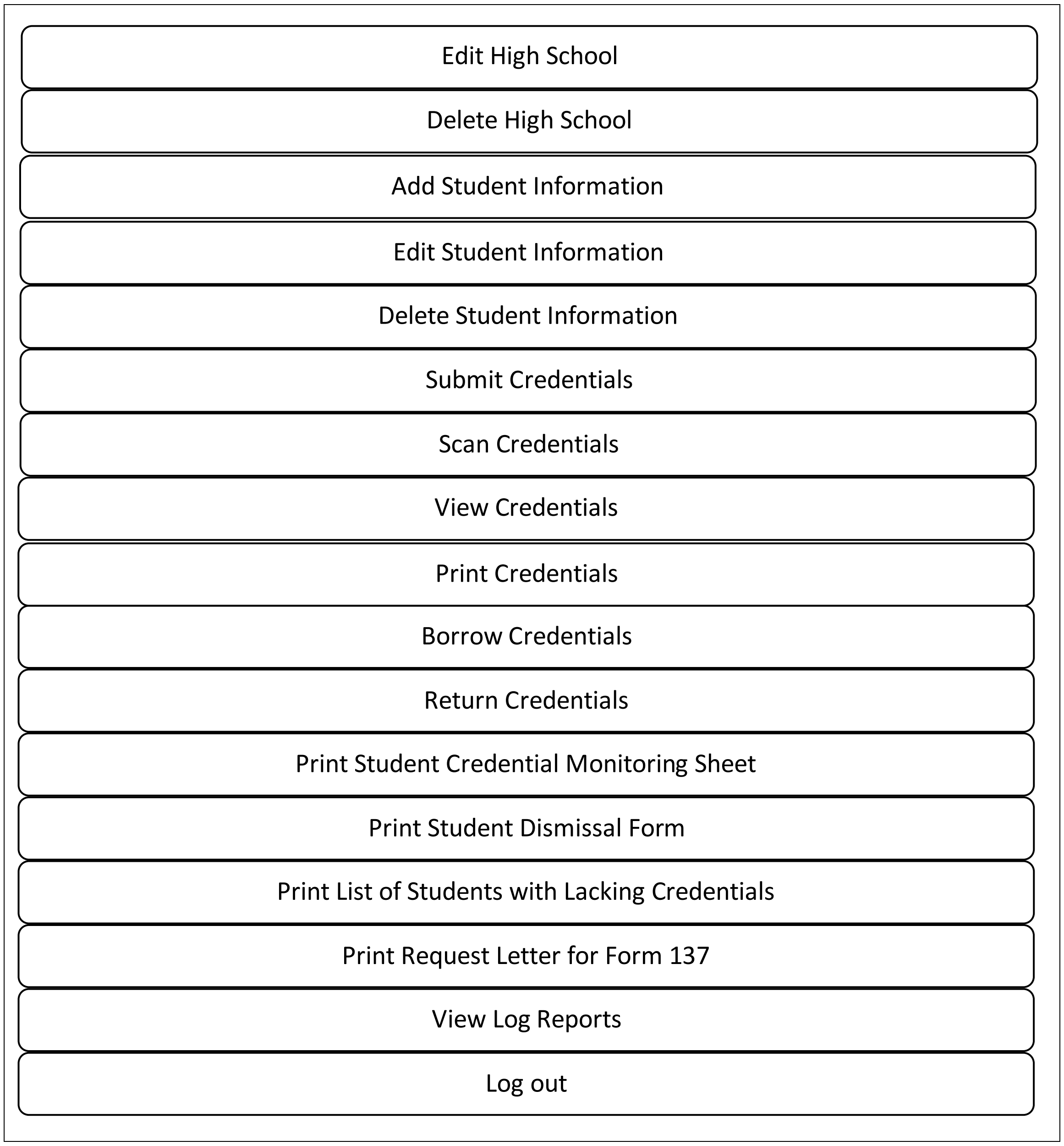


Figure 6. Use Case

USE CASE DIAGRAM

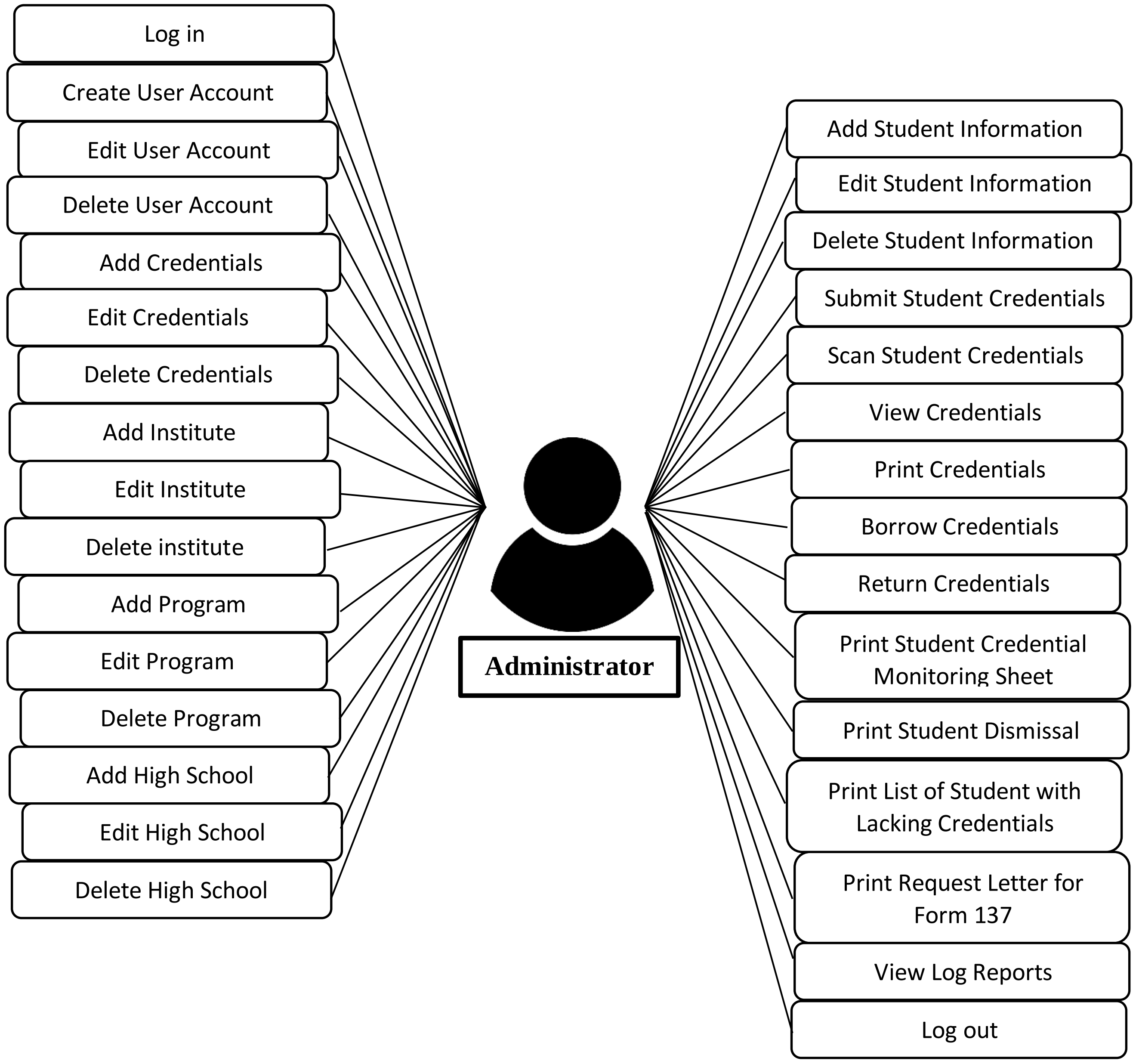


Figure 7. Use Case Diagram (Administrator)

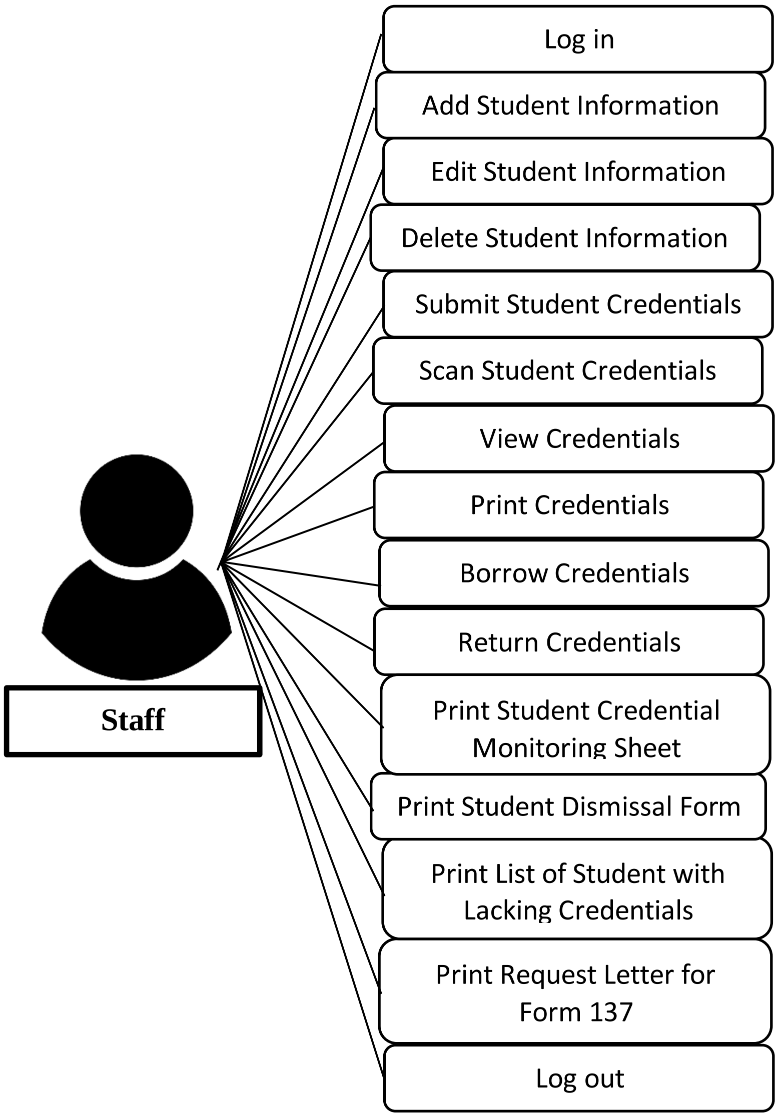

Log in
Add Student Information
Edit Student Information
Delete Student Information
Submit Student Credentials
Scan Student Credentials
View Credentials
Print Credentials
Borrow Credentials
Return Credentials
Print Student Credential Monitoring Sheet
Print Student Dismissal Form
Print List of Student with Lacking Credentials
Print Request Letter for Form 137
Log out
Staff

Figure 8. Use Case Diagram (Staff)

Use Case Scenario

Table 1

Log in

Use Case Name	Log in
Actor	Administrator and Staff
Description	The Use Case is used by the Actor to access the system.
Normal Flow	1) The use case begins when the actor inputs the username and password. 2) The Actor will click Login. 3) The system will show the main form. 4) Use Case instance will terminate.
Precondition	The actor must have an existing account.
Post-condition	If the use case was successful, the actor is now log into the system. If not, the system state is unchanged.
Assumption	The Actor is on the system.

Table 2

Create User Account

Use Case Name	Create User Account
Actor	Administrator
Description	The Use Case is used by the Actor to add new individual account to access the system.
Normal Flow	1) The use case begins when the actor clicks the Manage Account button. 2) The actor will then input the user information and then click the save button. 3) The system will show a message box. 4) The actor will click OK button. 5) The system will save the inputted information in the database and will display the updated list of the user accounts. 6) The use case instance will terminate.
Precondition	The account to be added must be available and the actor must login the system.
Post-condition	The registered account can be used to access the system.
Assumption	The Actor is on the system.

Table 3

Edit User Account

Use Case Name	Edit User Account
Actor	Administrator
Description	The Use Case is used by the Actor to edit the user information in the system.
Normal Flow	1) The use case begins when the actor clicks the Manage Account button. 2) The actor will search or select a user account from the list and then double click the selected account. 3) The actor will input the updates on the user information and then click the Save button. 4) The system will show a message box. 5) The actor will click the OK button. 6) The system will save the edited information in the database and will display the updated user information in the list. 7) The use case instance terminates.
Precondition	The account to be updated must exists in the system.
Post-condition	The updated accounts must be used when accessing the system.
Assumption	The Actor is on the system.

Table 4

Delete User Account

Use Case Name	Delete User Account
Actor	Administrator
Description	The Use Case is used by the Actor to delete user account from the system.
Normal Flow	1) The use case begins when the actor clicks the Manage Account button. 2) The actor will search or select a user account and then double click the selected account. 3) The actor will click the delete button. 4) The system will display a confirmation box. 5) The actor will click Yes button. 6) The system will show a message box that the user is successfully deleted 7) The actor will click OK button. 8) The system will display the updated list of the user account. 9) Use case instance terminates.
Precondition	The account to be deleted must be available in the system.
Post-condition	The account will be removed from the list of accounts.
Assumption	The Actor is on the system.

Table 5

Add Credentials

Use Case Name	Add Credentials
Actor	Administrator
Description	The Use Case is used by the Actor to add new credential to the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Credential Setup. 2) The actor will then input the Credential's Name. 3) The actor will click the Save button. 4) The system will show a message box 5) The actor will click OK button. 6) The system will save the newly added credential and will display the updated list of credentials. 7) Use case instance will terminates.
Precondition	The credentials must be a requirements needed in the Registrar's Office and must be available.
Post-condition	The actor will update the students about the new requirement.
Assumption	The Actor is on the system.

Table 6

Edit Credentials

Use Case Name	Edit Credentials
Actor	Administrator
Description	The Use Case is used by the Actor to edit credential to the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Credential Setup. 2) The actor will then select a credential from the list and then Double Click the selected credential. 3) The actor will edit the credential and then click Save button. 4) The system will show a message box. 5) The actor will click OK button. 6) The system will display the updated credential and the list of credentials. 7) Use case instance terminates.
Precondition	The account to be updated must exists in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 7

Delete Credentials

Use Case Name	Delete Credentials
Actor	Administrator
Description	The Use Case is used by the Actor to delete credentials from the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then Credential selects Setup. 2) The actor will then select a credential from the list and then Double Click the selected credential. 3) The actor will click the Delete button. 4) The system will ask the actor for confirmation. 5) The actor will click Yes button. 6) The system will show a message box. 7) The actor will click OK button. 8) The system will display the updated credential in the list of the credentials. 9) Use case instance terminates.
Precondition	The credential to be deleted must be available in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 8

Add Institute

Use Case Name	Add Institute
Actor	Administrator
Description	The Use Case is used by the Actor to add new institute to the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Manage Institute. 2) The actor will then input the Institute code and name. 3) The actor will click the Save button. 4) The system will show a message box. 5) The actor will click OK. 6) The system will display the updated list of institute. 7) Use case instance terminates.
Precondition	The actor must first validate the adding of new institute and the institute to be added does not exists in the system.
Post-condition	Update other users about the new institute added in the system.
Assumption	The Actor is on the system.

Table 9

Edit Institute

Use Case Name	Edit Institute
Actor	Administrator
Description	The Use Case is used by the Actor to edit institute to the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Manage Institute. 2) The actor will then select an institute from the list and then Double Click the selected institute. 3) The actor will edit the institute code or name and then click Save button. 4) The system will show a message box. 5) The actor will click OK. 6) The system will display the updated list of institute. 7) Use case instance terminates.
Precondition	The institute to be updated must exists in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 10

Delete Institute

Use Case Name	Delete Institute
Actor	Administrator
Description	The Use Case is used by the Actor to delete institute from the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Manage Institute. 2) The actor will then select an institute to be removed from the list and then Double Click the selected institute. 3) The actor will click the delete button. 4) The system will ask the actor for confirmation 5) The actor will click Yes. 6) The system will show a message box. 7) The actor will click OK. 8) The system will display the updated institute and the list of the institute. 9) Use case instance terminates.
Precondition	The institute to be deleted must be available in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 11

Add Program

Use Case Name	Add Program
Actor	Administrator
Description	The Use Case is used by the Actor to add new program to the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Manage Program. 2) The actor will then input the program code and name. 3) The actor will click the save button. 4) The system will show a message box. 5) The actor will click OK button. 6) The system will display the updated list of institute. 7) Use case instance terminates.
Precondition	The actor must first validate the adding of new program and the program to be added does not exists in the system.
Post-condition	Update other users about the new program added in the system.
Assumption	The Actor is on the system.

Table 12

Edit Program

Use Case Name	Edit Program
Actor	Administrator
Description	The use case is use to edit program in the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Manage Program. 2) The actor will then select a program from the list and then Double Click the selected program. 3) The actor will edit the program 4) The actor will click Save button. 5) The system will show a message box. 6) The actor will click OK. 7) The system will display the updated program and the list of program. 8) Use case instance terminates.
Precondition	The program to be updated must be available in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 13

Delete Program

Use Case Name	Delete Program
Actor	Administrator
Description	The Use Case is used by the Actor to delete program from the system.
Normal Flow\	1) The use case begins when the actor clicks the Administration menu and then selects Manage Program. 2) The actor will then select a program to be removed from the list and then Double Click the selected program. 3) The actor will click the delete button. 4) The system will ask the actor for confirmation 5) The actor will click Yes. 6) The system will show a message box. 7) The actor will click OK. 8) The system will display the updated program and the list of the program. 9) Use case instance terminates.
Precondition	The program to be deleted must be available in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 14

Add High School

Use Case Name	Add High School
Actor	Administrator
Description	The use case is use to add student high school background in the system for Form 137 purposes.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then Manage School. 2) The actor will then input the school code and name. 3) The actor will click the save button. 4) The system will show a message box. 5) The actor will click OK button. 6) The system will display the updated school and the list of school. 7) Use case instance terminates.
Precondition	The actor must first validate the adding of new school and the school to be added does not exists in the system.
Post-condition	Update other users about the new school added in the system.
Assumption	The Actor is on the system.

Table 15

Edit High School

Use Case Name	Edit High School
Actor	Administrator
Description	The use case is use to edit student high school background in the system for Form 137 purposes.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then selects Manage School. 2) The actor will then select a school from the list and then Double Click the selected school. 3) The actor will edit the school information. 4) The actor will click Save button. 5) The system will show a message box. 6) The actor will click OK button. 7) The system will display the updated school and the list of school. 8) Use case instance terminate.
Precondition	The institute to be updated must exists in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 16

Delete High School

Use Case Name	Delete High School
Actor	Administrator
Description	The Use Case is used by the Actor to delete program from the system.
Normal Flow	1) The use case begins when the actor clicks the Administration menu and then Manage School. 2) The actor will then select a school to be removed from the list and then Double Click the selected school. 3) The actor will click the delete button. 4) The system will ask the actor for confirmation 5) The actor will click Yes. 6) The system will show a message box. 7) The actor will click OK. 8) The system will display the updated school and the list of the school. 9) Use case instance terminates.
Precondition	The School to be deleted must be available in the system.
Post-condition	Update other users of the system about the changes made by the actor.
Assumption	The Actor is on the system.

Table 17

Add Student Information

Use Case Name	Add student Information
Actor	Administrator and Staff
Description	The Use Case is used to add student in the system.
Normal Flow	1) The use case begins when the actor clicks the Student Entry button. 2) The actor will then input the student information and high school background. 3) The actor will click the Save button. 4) The system will show a message. 5) The actor will click OK button. 6) The system will display the new student and the list of student. 7) Use case instance terminates.
Precondition	The student information must be available.
Post-condition	The student record will be available in the system.
Assumption	The Actor is on the system.

Table 18

Edit Student Information

Use Case Name	Edit student Information
Actor	Administrator and Staff
Description	The Use Case is used to edit student information in the system.
Normal Flow	1) The use case begins when the actor clicks the Student Entry button. 2) The actor will search student and then double click student in the list. 3) The actor will edit the student information. 4) The actor will click Save button. 5) The system will show a message box. 6) The actor will click OK. 7) The system will display the updated student and the list of student. 8) Use case instance terminates.
Precondition	The student information must exist in the system.
Post-condition	The actor can edit student information in the system.
Assumption	The Actor is on the system.

Table 19

Delete Student Information

Use Case Name	Delete student Information
Actor	Administrator and Staff
Description	The use case is use to delete student information in the system.
Normal Flow	1) The use case begins when the actor clicks the Student Entry button. 2) The actor will then search student I.D. and then double click student in the list. 3) The actor will click the delete button. 4) The system will ask for confirmation from the actor and then click Yes. 5) The system will show a message box. 6) The actor will click OK. 7) The system will display the updated list of the student. 8) Use case instance terminate.
Precondition	The student information must in the system.
Post-condition	Actor can delete student in the system.
Assumption	The Actor is on the system.

Table 20

Submit Student Credentials

Use Case Name	Submit Student Credentials
Actor	Administrator and Staff
Description	The Use Case is used to submit credentials of the students in the system.
Normal Flow	1) The use case begins when the actor clicks the Submit Credentials button. 2) The actor will search student using Student I.D. 4) The actor will select credentials. 5) The actor will click save checklist. 6) The system will show a message box. 7) The actor will click OK. 8) Use case instance terminates.
Precondition	The credentials to be submitted must be available.
Post-condition	The Actor can Scan Credentials and save in system.
Assumption	The Actor is on the system.

Table 22

Scan Student Credentials

Use Case Name	Scan Student Credentials
Actor	Administrator and Staff
Description	The Use Case is used to scan and store submitted credentials of the students in the system database.
Normal Flow	1) The use case begins when the actor clicks the Submit Credentials button. 2) The actor will search student using Student I.D. 4) The actor will select credentials to submit. 5) The actor will click SCAN>> button. 6) The system will show a dialog for scanning. 7) The actor will click Scan Document. 8) The system will display the scan document. 9) The actor will click Save Document. 10) Use case instance terminates.
Precondition	The credential to be submitted must be available in the system.
Post-condition	The credential status will be changed into COMPLETE.
Assumption	The Actor is on the system.

Table 23

View Credentials

Use Case Name	View Credentials
Actor	Administrator and Staff
Description	The Use Case is used to view the scanned credentials of the students in the system.
Normal Flow	1) The use case begins when the actor clicks the Submit Credentials button. 2) The actor will search student using Student I.D. 4) The actor will select credentials. 5) The actor will click View Credential button 6) The system will show the scanned document. 7) Use case instance terminates.
Precondition	The credential to be viewed must be available.
Post-condition	The actor can view credentials in the system.
Assumption	The Actor is on the system.

Table 24

Print Credentials

Use Case Name	Print Credentials
Actor	Administrator and Staff
Description	The Use Case is used to print the scanned credentials of the student.
Normal Flow	1) The use case begins when the actor clicks the Submit Credentials button. 2) The actor will search student using Student I.D. 3) The actor will select credentials to print. 4) The actor will click View Credential button 6) The system will show the scanned document. 7) The actor will click Print Report button. 8) Use case instance terminates.
Precondition	The credential to be print must be available in the system.
Post-condition	The actor can print credentials in the system.
Assumption	The Actor is on the system.

Table 25

Borrow Credentials

Use Case Name	Borrow Credentials
Actor	Administrator and Staff
Description	The Use Case is used to manage the student credentials who have been barrowed by students.
Normal Flow	1) The use case begins when the actor clicks the Borrow Credentials button. 2) The actor will search for the student I.D. 3) The actor will select student credential that has already submitted. 4) The actor will input reason for borrowing the credential. 5) The actor will click borrow button. 6) The system will show. 7) The actor will click OK button. 8) Use case instance terminate.
Precondition	The credential to be barrowed must be available.
Post-condition	Transaction made in the system will be display in the Borrow Credential List.
Assumption	The Actor is on the system.

Table 26

Return Credentials

Use Case Name	Return Credentials
Actor	Administrator and Staff
Description	The Use Case is used to manage the student credentials who have been returned by students.
Normal Flow	1) The use case begins when the actor clicks the Borrow Credentials button. 2) The actor will search for the student I.D. 3) The actor will select student borrowed credential from the Borrow Credential list. 4) The user will click Return button. 5) The system will show message box. 6) The actor will click OK. 7) Use case instance terminates.
Precondition	The student must have record in the system that he/she borrowed an credentials.
Post-condition	Transaction made in the system will be registered the Borrow Credential List.
Assumption	The Actor is on the system.

Table 27

Print Student Credential Monitoring Sheet

Use Case Name	Print Student Credential Monitoring Sheet
Actor	Administrator and Staff
Description	The Use Case is used to print individual report of the student about the lacking credentials they need to comply.
Normal Flow	1) The use case begins when the actor clicks the Student Records button. 2) The actor will then search for the student I.D. or last name. 3) The actor will click Print Lacking Credential. 4) The system will then generate the form. 5) The user will click Print Report button. 6) Use case instance terminates
Precondition	The student must have existing record in the system.
Post-condition	Provide the printed report to the student.
Assumption	The Actor is on the system.

Table 28

Print Student Dismissal Form

Use Case Name	Print Student Dismissal Form
Actor	Administrator and Staff
Description	The Use Case is used to print Dismissal Form for the student who will transfer other school.
Normal Flow	1) The use case begins when the actor clicks the Student Records button. 2) The actor will then search for the student I.D. or last name. 3) The actor will input the OR number. 4) The actor will click Generate Dismissal Form. 5) The system will then generate the form. 6) The actor will click Print Report icon. 7) Use case instance terminates.
Precondition	The student must be available in the system.
Post-condition	The actor can print student dismissal form in the system.
Assumption	The Actor is on the system.

Table 29

Print List of Students with Lacking Credentials

Use Case Name	Print List of Students with Lacking Credentials
Actor	Administrator and Staff
Description	The Use Case is used to print reports of the list of student in every institution with lacking credentials.
Normal Flow	1) The use case begins when the actor clicks the List of Lacking Credential menu. 2) The actor will select an institute. 3) The actor will click Search button. 4) The system will generate the form. 5) The actor will click Print Report button. 6) Use case instance terminates.
Precondition	The student must have existing record in the system.
Post-condition	The actor can print the list of students with Lacking Credentials in the system.
Assumption	The Actor is on the system.

Table 30

Print Request Letter for Form 137

Use Case Name	Print Request Letter for Form 137
Actor	Administrator and Staff
Description	The Use Case is used to print request letter that will be send in the students respected high schools for acquiring the Form 137.
Normal Flow	1) The use case begins when the actor clicks the Request Form 137 menu. 2) The actor will search school name. 3) The actor will click Search button. 4) The system will then generate the form. 5) The actor will click Print Report button. 6) Use case instance terminates.
Precondition	The student must have an existing record in the system.
Post-condition	The actor can print letter for Form 137 in the system.
Assumption	The Actor is on the system.

Table 31

View Log Reports

Use Case Name	View Log Reports
Actor	Administrator
Description	The Use Case is used to view the log reports.
Normal Flow	1) The use case begins when the actor clicks the Login History button. 2) The actor will input action either description of the user and then set date range. 3) The actor will click Search. 4) The system will display the list of transactions. 5) Use case instance terminates.
Precondition	There should be a transactions made by the users.
Post-condition	The actor can view the log reports in the system.
Assumption	The Actor is on the system.

Table 32

Backup

Use Case Name	Backup
Actor	Administrator
Description	The Use Case is used to backup system data.
Normal Flow	1) The use case begins when the actor clicks Administration menu and then selects Backup. 2) The system will the show a file dialog. 3) The actor will select directory. 4) The actor will click Save button. 5) Use case instance terminate.
Precondition	The actor is required to backup the system daily.
Post-condition	The actor can backup the system anytime.
Assumption	The Actor is on the system.

Table 33

Log out

Use Case Name	Log out
Actor	Administrator and Staff
Description	The Use Case is used when user wants to log out in the system.
Normal Flow	1) The use case begins when the user clicks the Logout menu. 2) The system will ask for confirmation from the actor to log out. 3) The Actor will click Yes button. 4) Use Case instance terminate.
Precondition	The users must be log in in the system
Post-condition	
Assumption	The Actor is out of the system.

DATA FLOW DIAGRAM

Narrative flow (Existing): The paper-based system where the student must submit the required credentials, will be submitted to the registrar's office staffs.

Data flow diagram (Existing)

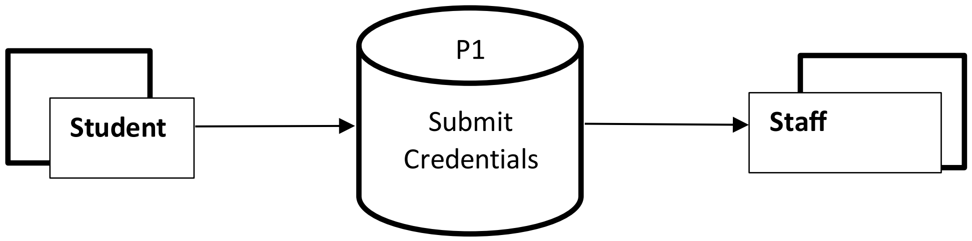


Figure 9. Submission of Credentials

Narrative flow (Existing): After the students submit their documents, the staff will arrange the documents and then store it in a cabinet. When retrieving the documents the staff will look for it in the cabinet.

Data flow diagram (Existing)

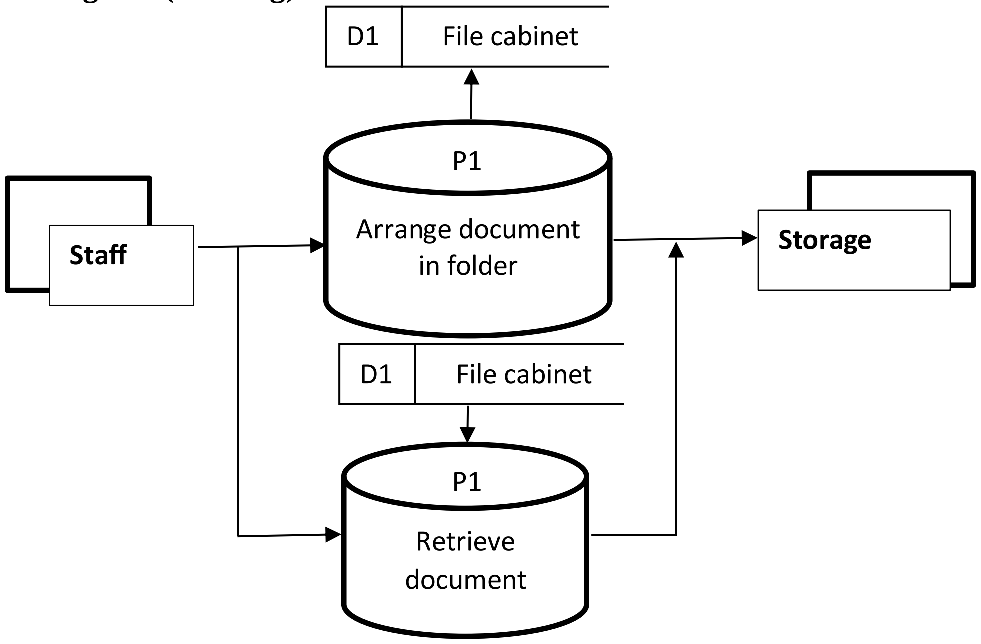


Figure 10. Storing and retrieving of documents

Proposed System

Log in

Narrative flow : For the users to able to access the system, they will first log in into the system. They will first input their username and password. And then click log in into the system.

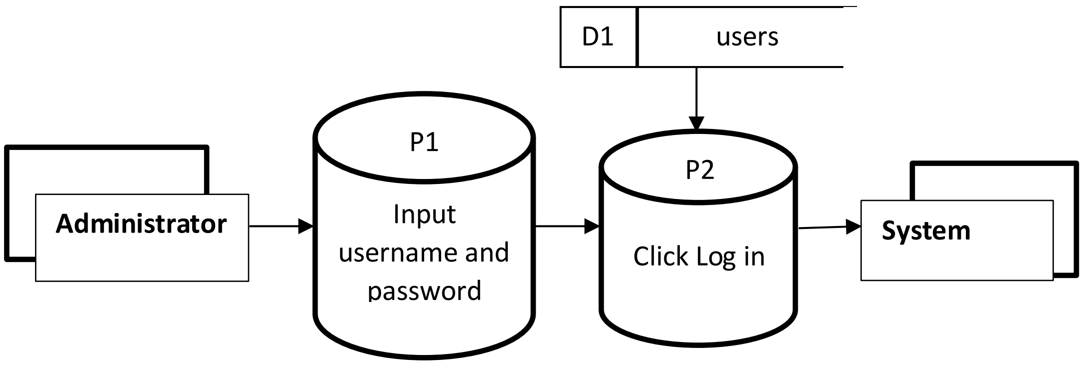


Figure 11. Log in (Proposed)

Manage User Account

Narrative Flow: The administrator will log-in into the system, and then manage user account. In addition, the administrator will add a user in the system and input user's personal information together with the username and password, and then click save. The information will be save in users table. If there is an existing account, the administrator can either edit or delete the user account by searching first the username of the user. Further, if the user wants to clear the information all at once, the user can click the new/refresh button.

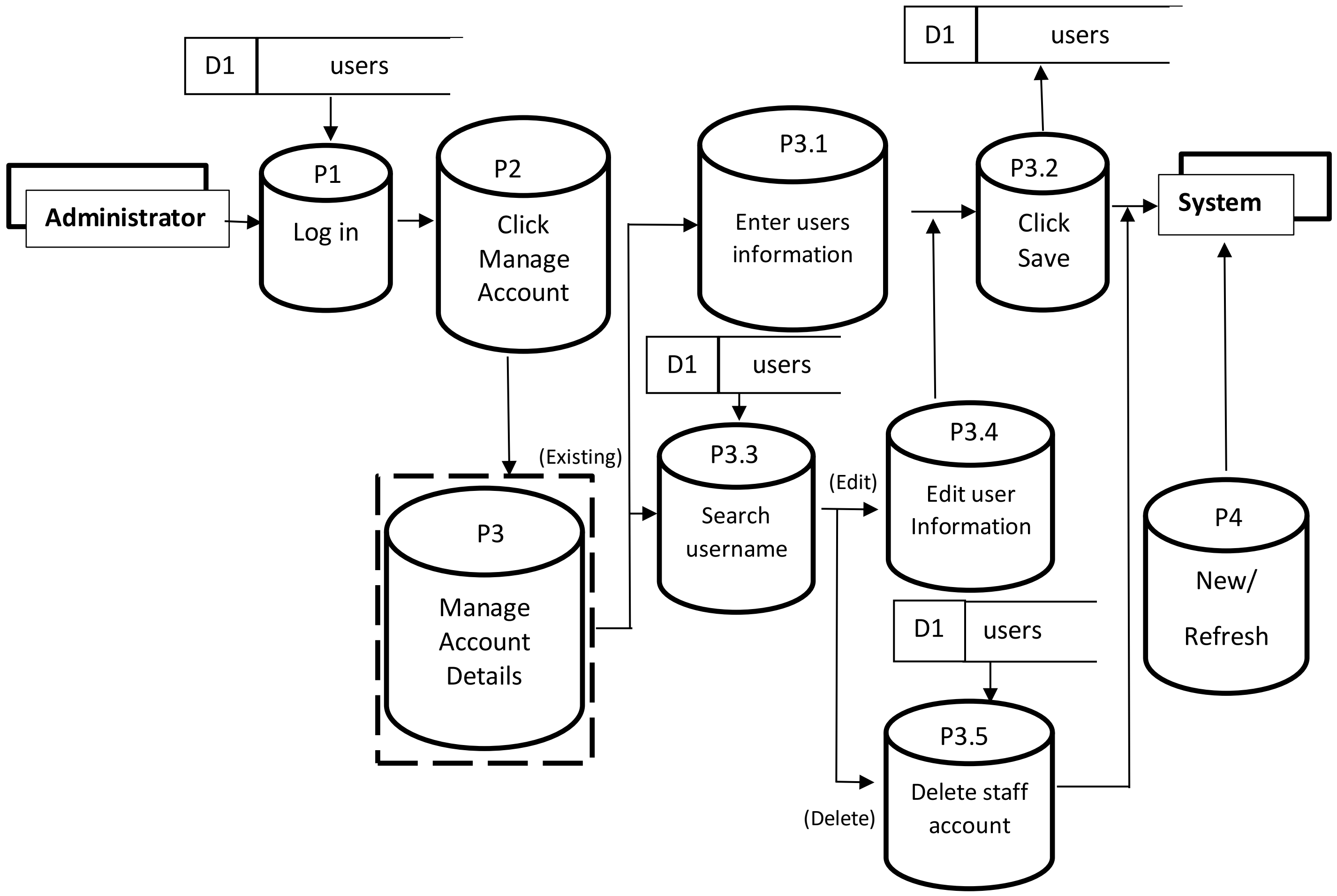


FIGURE 12. Manage User Account (Proposed)

Credential Set Up

Narrative Flow: The administrator will log-in into the system, and then click administration and then select credential setup. The user can input the credentials name and then save. Moreover, if there is existing credentials, the user can either edit or delete credentials in the system.

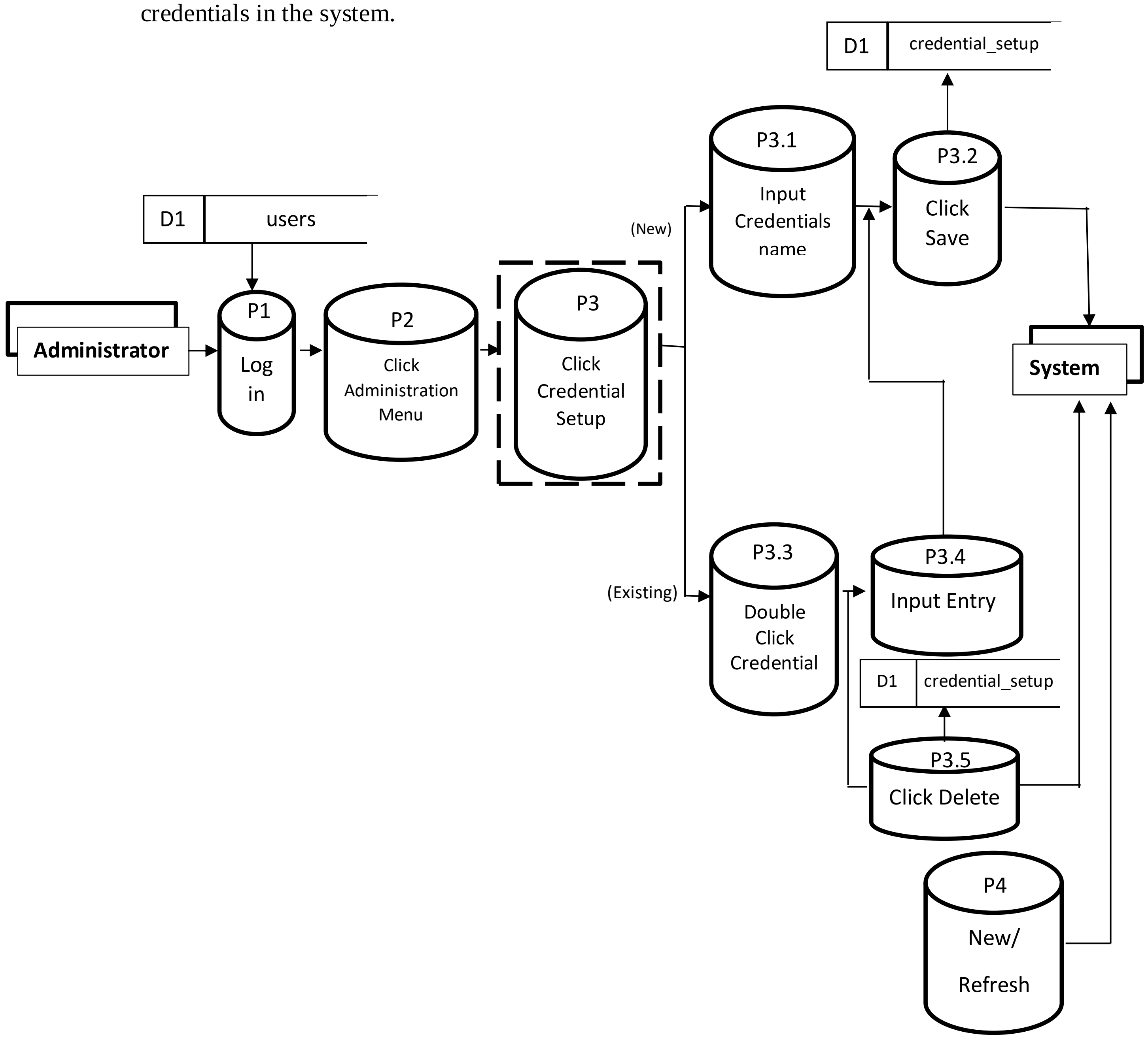


FIGURE 13. Manage Credentials (Proposed)

Manage Institute

Narrative Flow: The administrator will log-in into the system, and then go to administration menu and then select manage institute. After that, the user can input the institute code and name and then save. Moreover, if there is an existing record, the user can either edit or delete institute in the system. The administrator will select and double click the institute and then click edit or delete. The changes will be save and stored in the institute table.

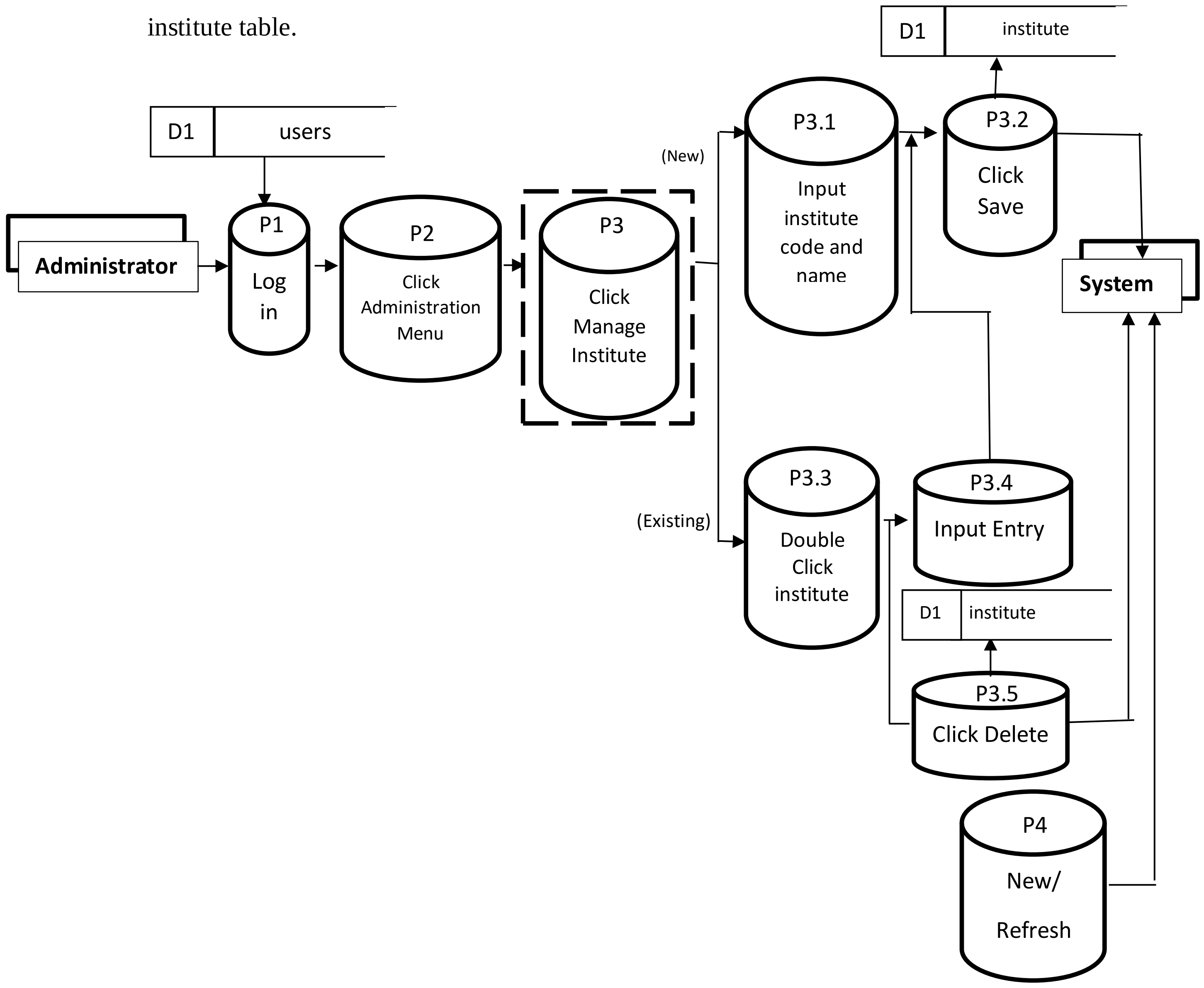


FIGURE 14. Manage Institute (Proposed)

Manage Program

Narrative Flow: The administrator will log-in into the system, and then go to administration menu and then click manage program. After that, the user can input the program code, name, and major. Then the user will select institute where that program belongs then click save. Moreover, if there is an existing record, the user can either edit or delete program in the system. The administrator will select and double click the institute and then click edit or delete. The changes will be save and stored in the program table.

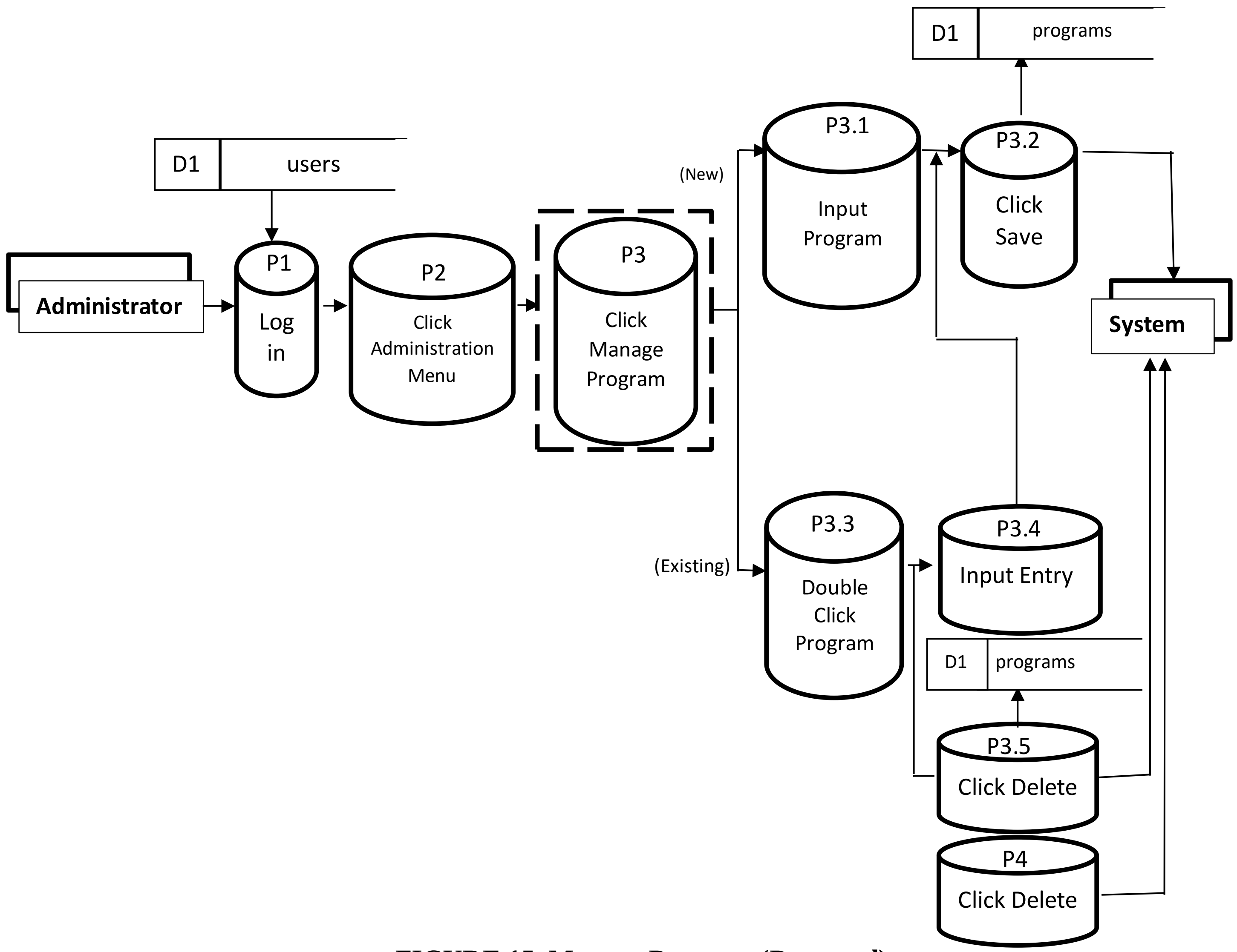


FIGURE 15. Manage Program (Proposed)

Manage High School

Narrative Flow: The administrator will log-in into the system, and then go to administration menu and then click manage school. After that, the user can input the student high school background information and then click save. Moreover, if there is an existing record, the user can either edit or delete the school information in the system. The administrator will select and double click the school and then click edit or delete. The changes will be save and stored in the hs_address table.

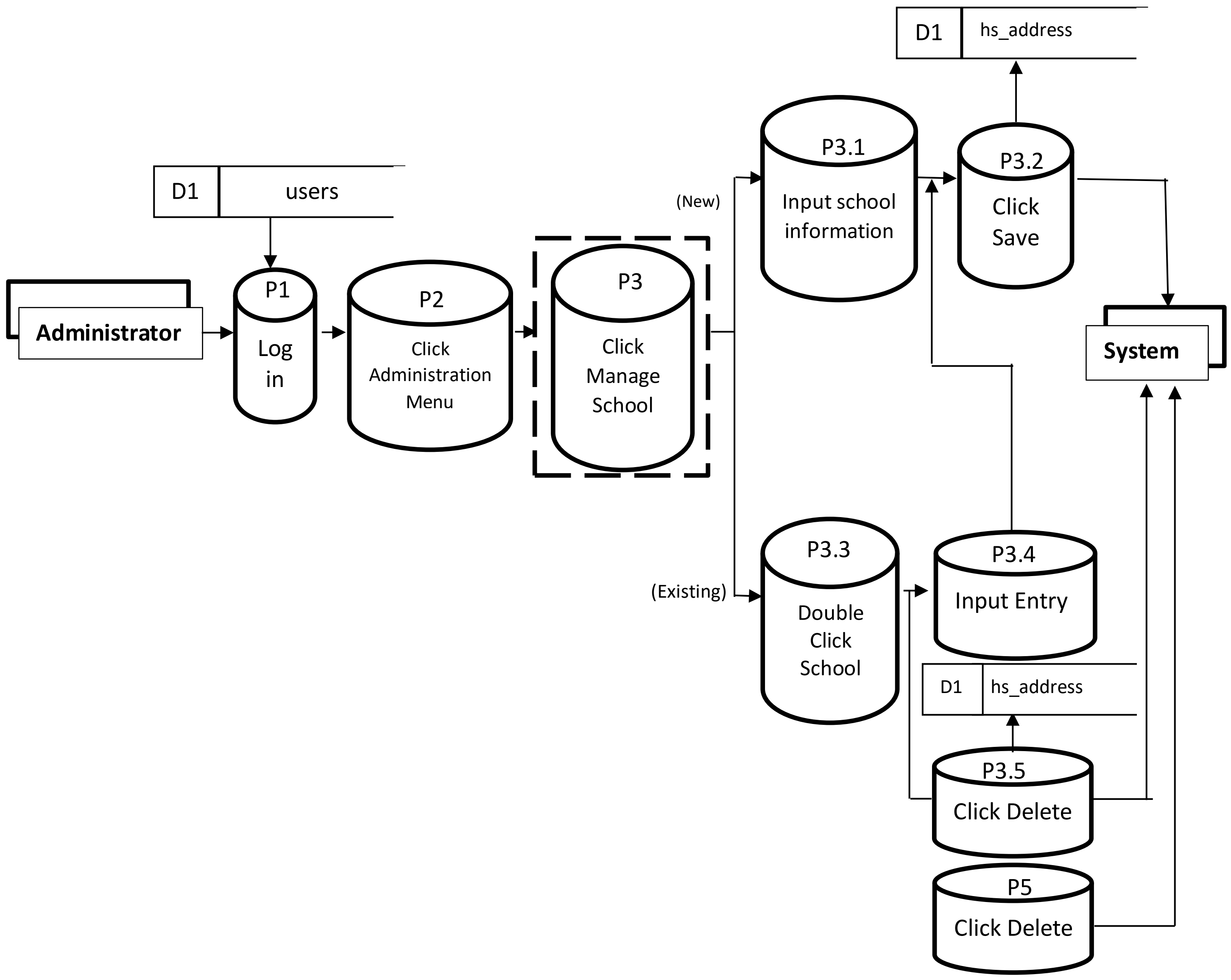


FIGURE 16. Manage High School (Proposed)

Manage Student Information

Narrative Flow: The administrator will log-in into the system, and then go to the student entry. After that, the user can input the student information then click save. Moreover, if there is existing information of the student the user can either edit or delete the student information in the system. The administrator will select and double click the student and then click edit or delete. The changes will be save and stored in the student table.

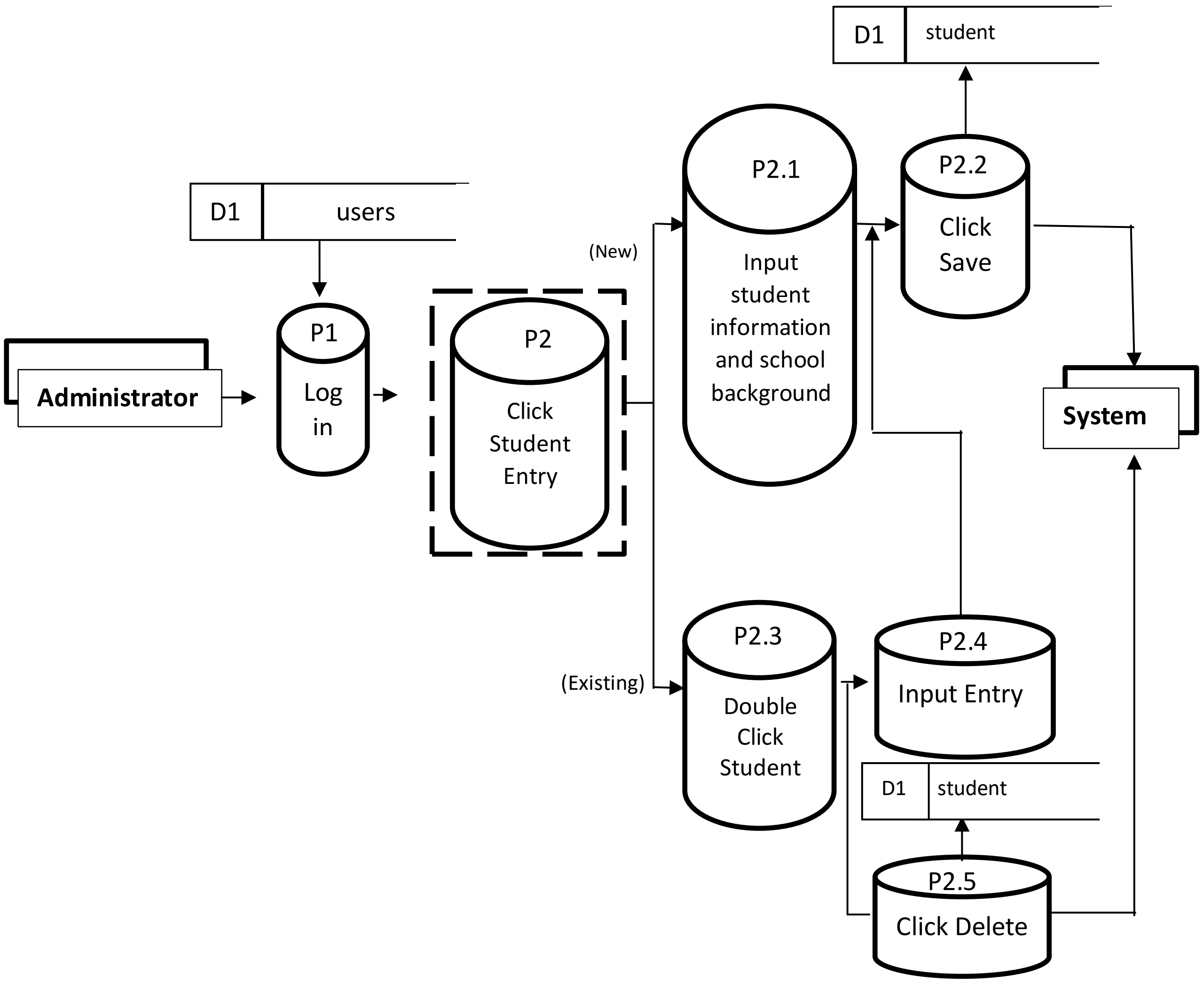


FIGURE 17. Manage Student Information (Proposed)

Scan Student Credentials

Narrative Flow: The user will log-in into the system, and then go to Submit credentials menu. After that, the user search student and then select credential to submit. The user will then click SCAN>> and then clicks scan document. After that, the user will click save document. The scanned credentials will be stored in the credential table.

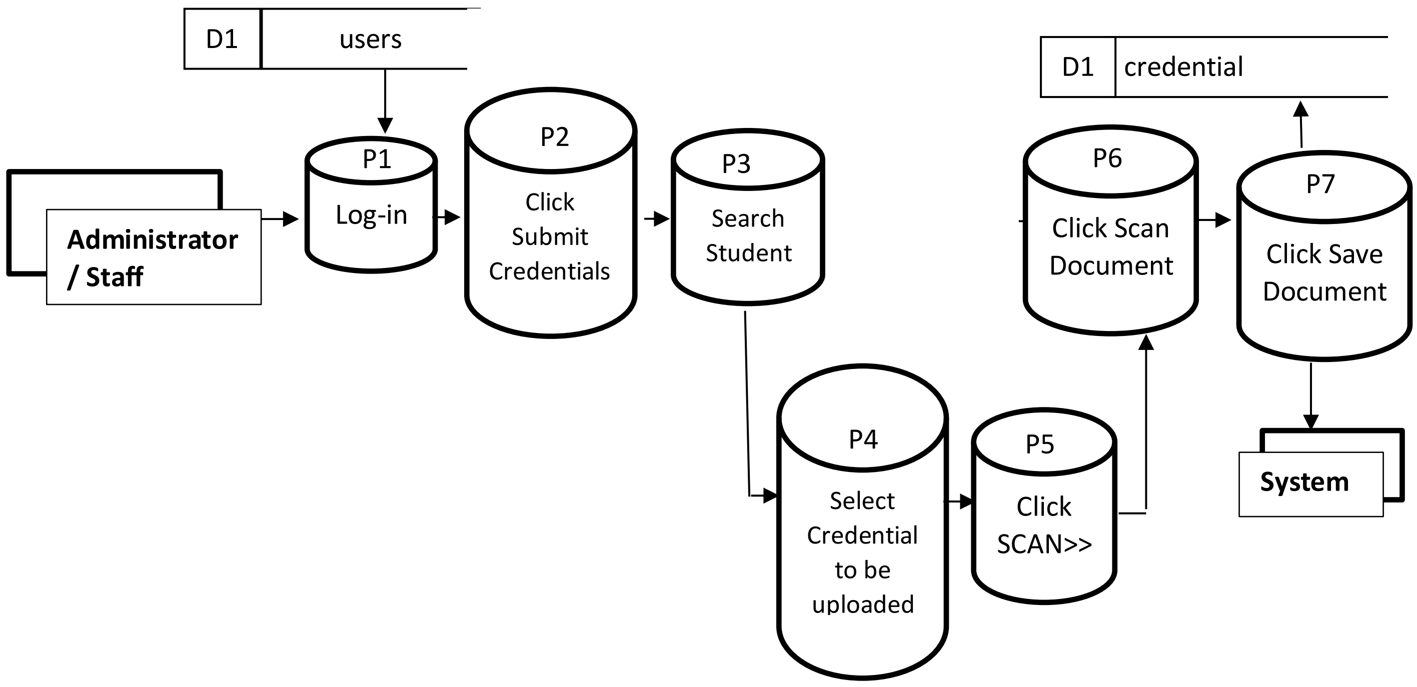


FIGURE 18. Manage Student Credentials (Proposed)

Manage Borrowed Credentials

Narrative Flow: The user must first log in into the system and then Borrow credentials where the user will search for the student. The user will then select credential and input reason in text box and then click borrow. Moreover, if the credential was already borrowed select borrowed credential in the list and then click return.

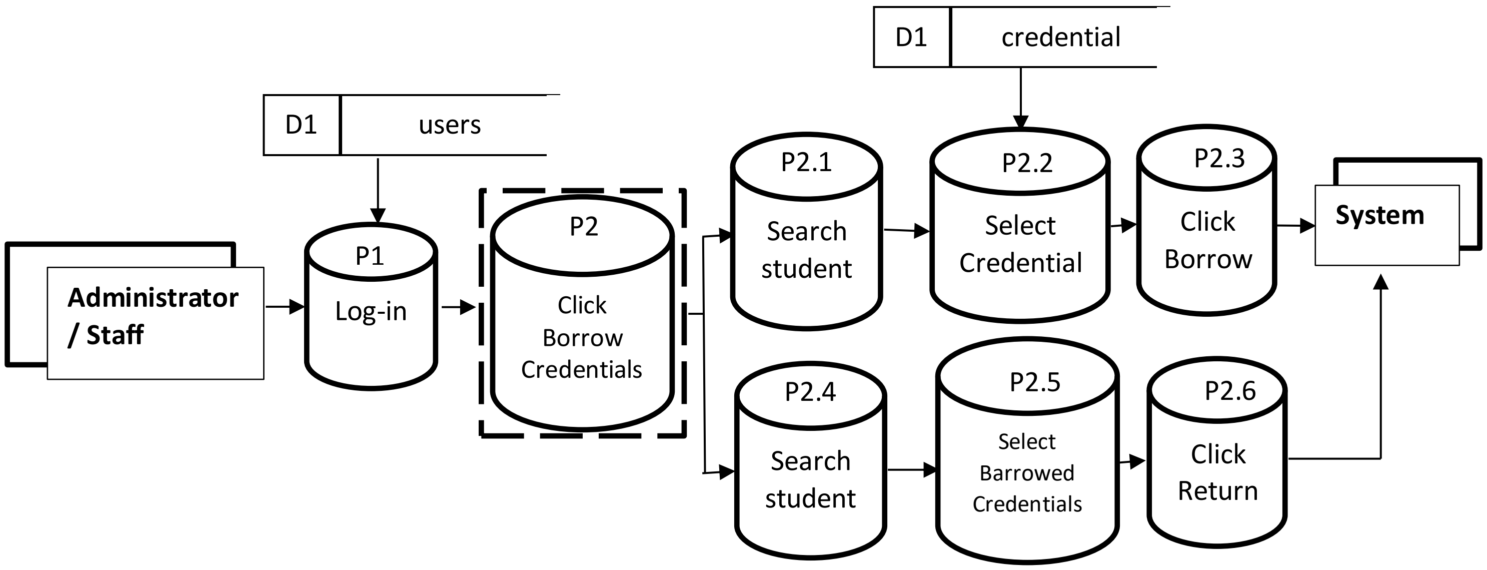


FIGURE 19. Manage Borrowed Credentials (Proposed)

<u>Print Student Credentials Monitoring Sheet</u>

Narrative Flow: The user must first log in into the system and then click Student Records to view record details. The user will then search for the student, then click the Print Lacking Credentials, and then click print the report.

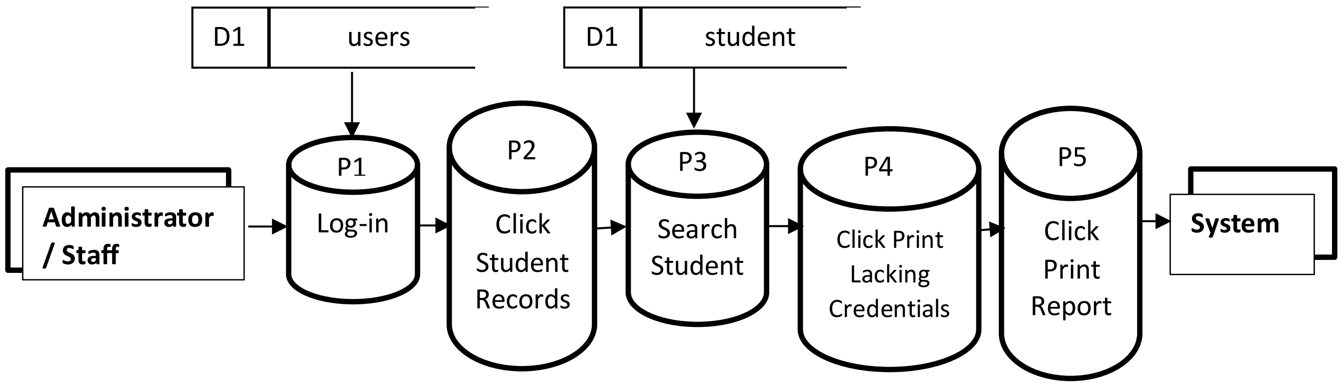


FIGURE 20. Print Student Credentials Monitoring Sheet (Proposed)

Print Student Dismissal Form

Narrative Flow: The user must first log in into the system and then view the Student Records. The user will then search for the student and then click print the form.

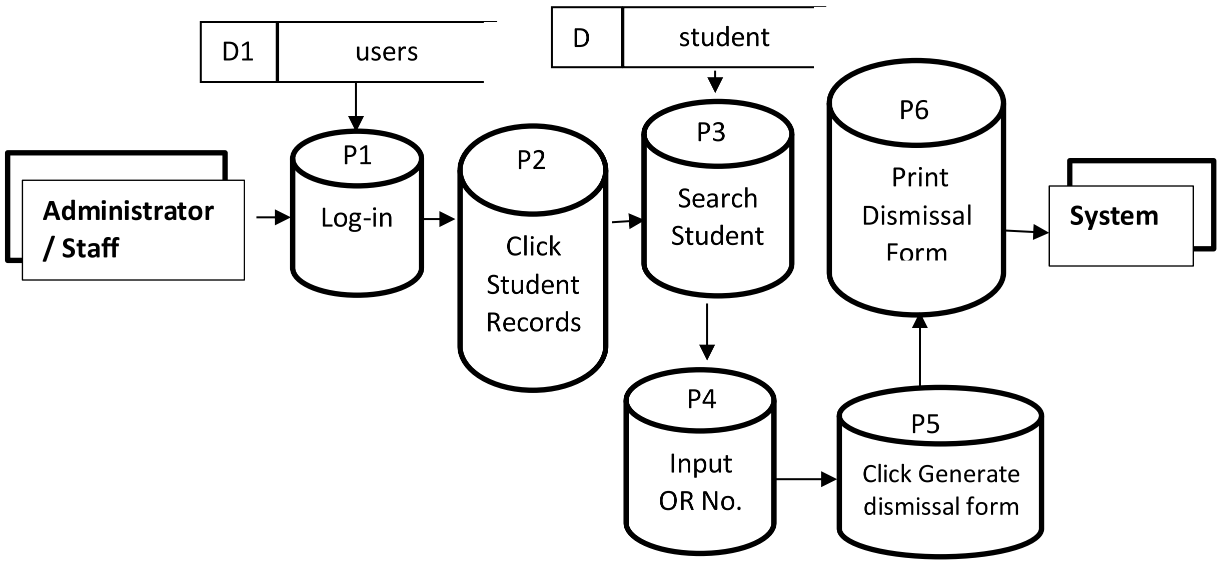


FIGURE 21. Print Student Dismissal Form (Proposed)

Print List of Students with Lacking Credentials

Narrative Flow: The user must first log in into the system and then go to List of Lacking Credentials. After that, the user will then search for the institute and then click print the report.

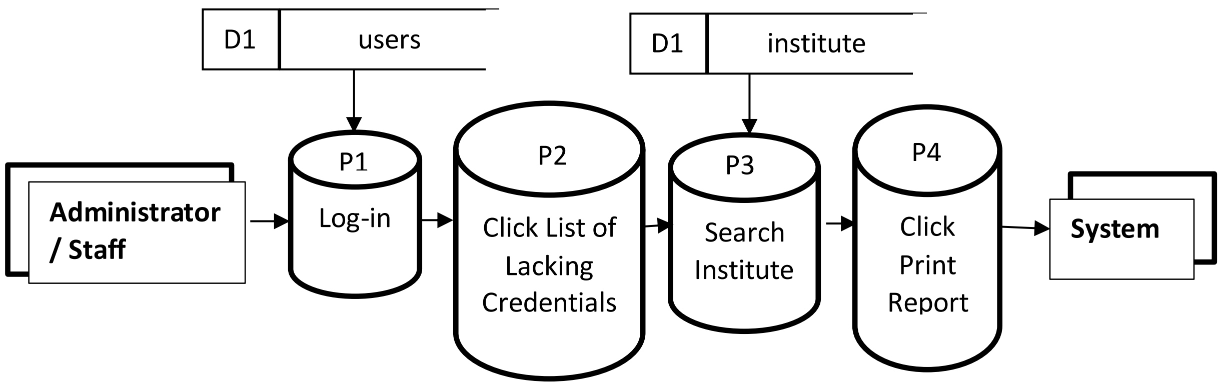


FIGURE 22. Print List of Students with Lacking Credentials (Proposed)

Print Request Letter for Form 137

Narrative Flow: The user must first log in into the system and then go to Request Form 137. After that, the user will then search for school and then click print report.

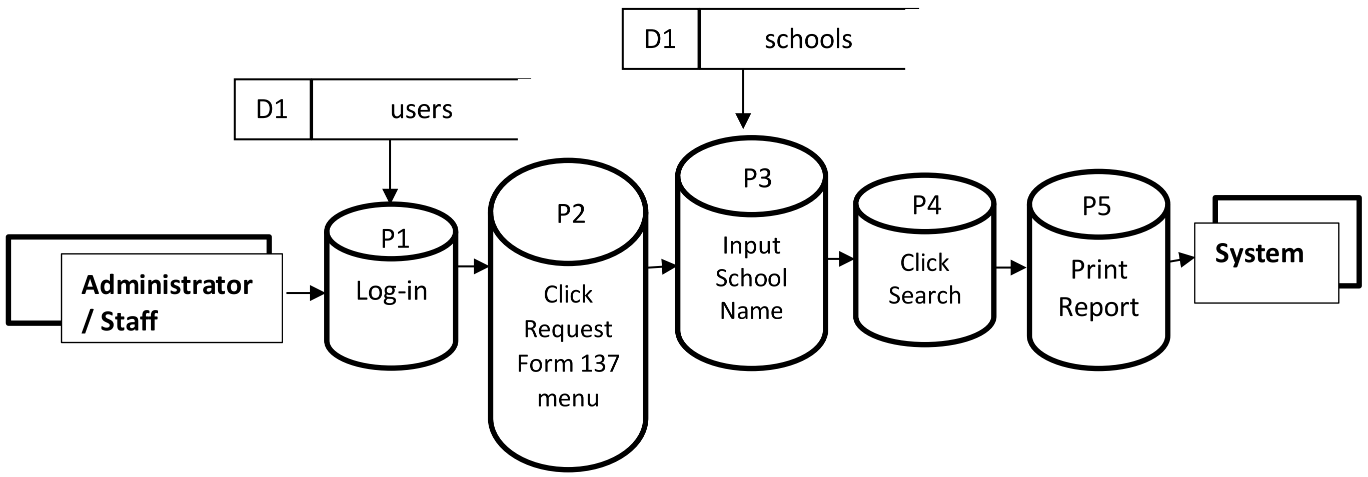


FIGURE 23. Print Request Letter for Form 137 (Proposed)

Log Reports

Narrative Flow: The administrator must first log in into the system and then view the log in history reports of the users.

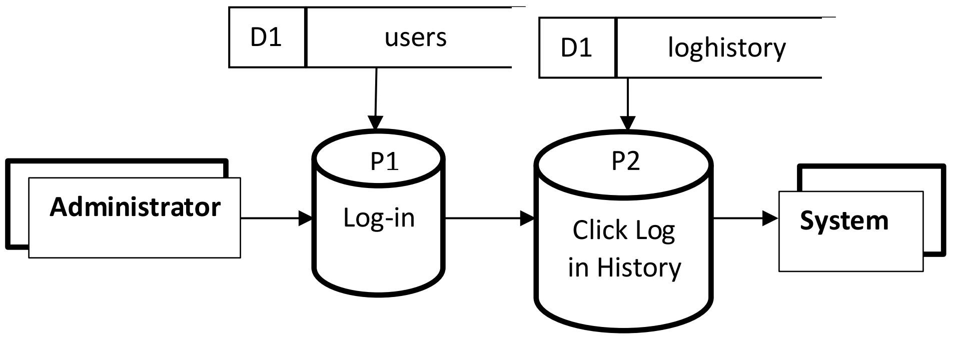
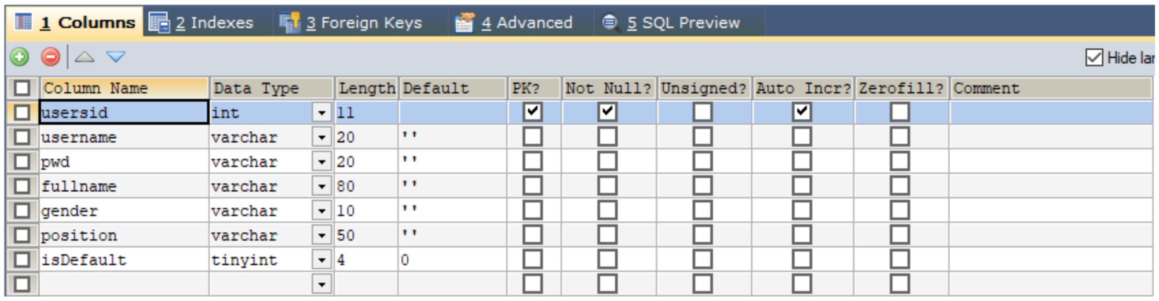


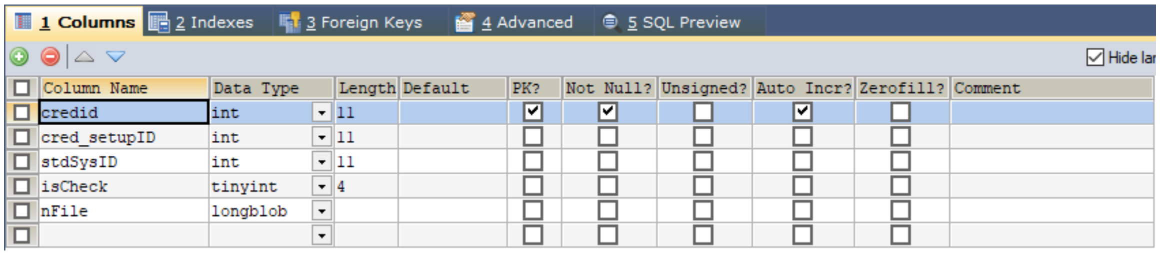
FIGURE 24. Log Reports (Proposed)

Database Structure



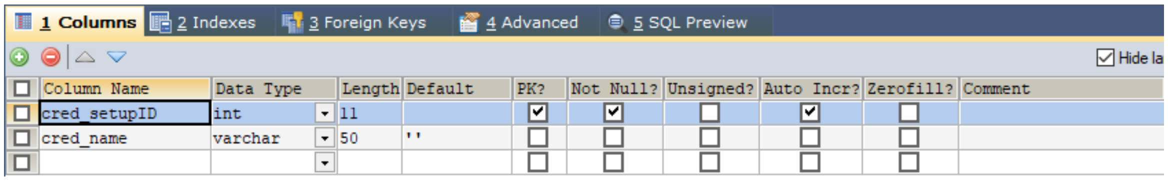
☐	Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
☐	usersid	int	11		☑	☑	☐	☑	☐	
☐	username	varchar	20	''	☐	☐	☐	☐	☐	
☐	pwd	varchar	20	''	☐	☐	☐	☐	☐	
☐	fullname	varchar	80	''	☐	☐	☐	☐	☐	
☐	gender	varchar	10	''	☐	☐	☐	☐	☐	
☐	position	varchar	50	''	☐	☐	☐	☐	☐	
☐	isDefault	tinyint	4	0	☐	☐	☐	☐	☐	
☐					☐	☐	☐	☐	☐	

Figure 25. users



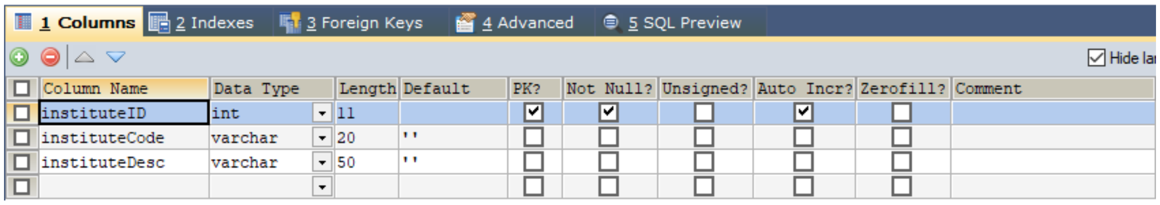
☐	Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
☐	credid	int	11		☑	☑	☐	☑	☐	
☐	cred_setupID	int	11		☐	☐	☐	☐	☐	
☐	stdSysID	int	11		☐	☐	☐	☐	☐	
☐	isCheck	tinyint	4		☐	☐	☐	☐	☐	
☐	nFile	longblob			☐	☐	☐	☐	☐	
☐					☐	☐	☐	☐	☐	

Figure 26. credential



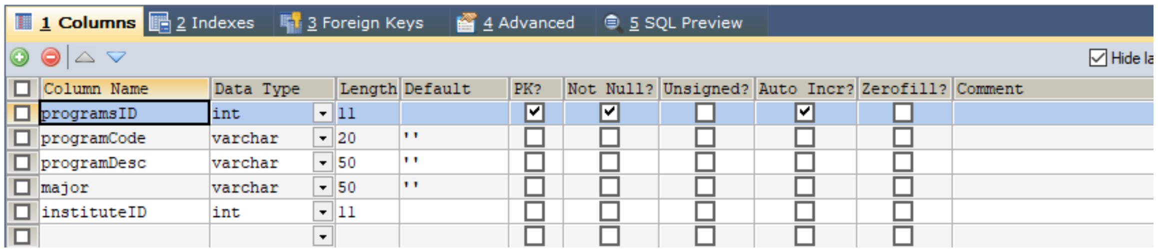
☐	Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
☐	cred_setupID	int	11		☑	☑	☐	☑	☐	
☐	cred_name	varchar	50	''	☐	☐	☐	☐	☐	
☐					☐	☐	☐	☐	☐	

Figure 27. credential_setup



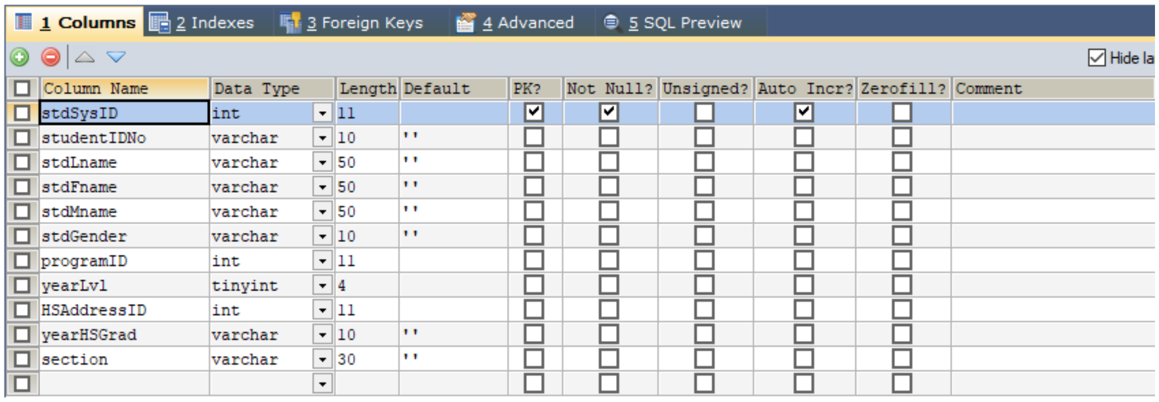
☐	Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
☐	instituteID	int	11		☑	☑	☐	☑	☐	
☐	instituteCode	varchar	20	''	☐	☐	☐	☐	☐	
☐	instituteDesc	varchar	50	''	☐	☐	☐	☐	☐	
☐					☐	☐	☐	☐	☐	

Figure 28. institute



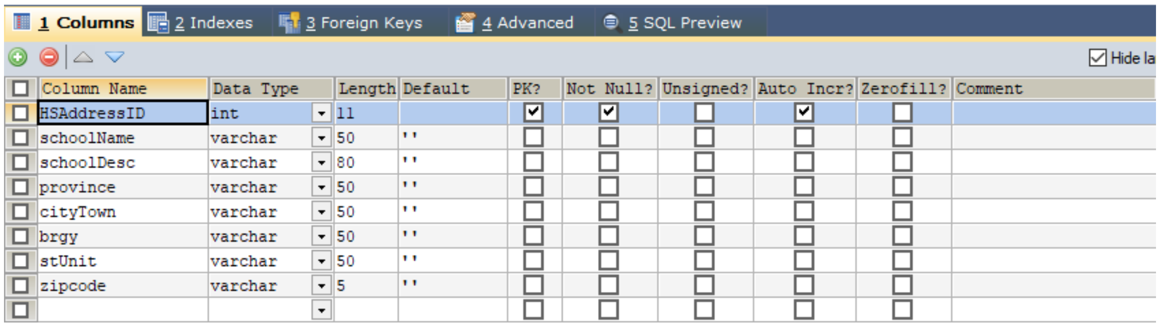
Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
programsID	int	11		☑	☑	☐	☑	☐	
programCode	varchar	20	''	☐	☐	☐	☐	☐	
programDesc	varchar	50	''	☐	☐	☐	☐	☐	
major	varchar	50	''	☐	☐	☐	☐	☐	
instituteID	int	11		☐	☐	☐	☐	☐	
				☐	☐	☐	☐	☐	

Figure 29. programs



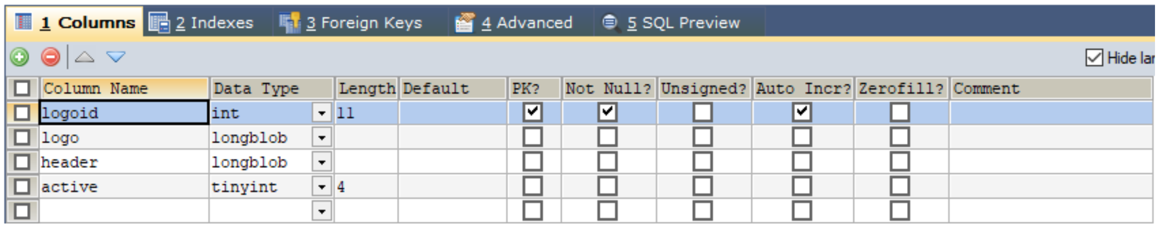
Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
stdSysID	int	11		☑	☑	☐	☑	☐	
studentIDNo	varchar	10	''	☐	☐	☐	☐	☐	
stdLname	varchar	50	''	☐	☐	☐	☐	☐	
stdFname	varchar	50	''	☐	☐	☐	☐	☐	
stdMname	varchar	50	''	☐	☐	☐	☐	☐	
stdGender	varchar	10	''	☐	☐	☐	☐	☐	
programID	int	11		☐	☐	☐	☐	☐	
yearLvl	tinyint	4		☐	☐	☐	☐	☐	
HSAddressID	int	11		☐	☐	☐	☐	☐	
yearHSGrad	varchar	10	''	☐	☐	☐	☐	☐	
section	varchar	30	''	☐	☐	☐	☐	☐	
				☐	☐	☐	☐	☐	

Figure 30. student



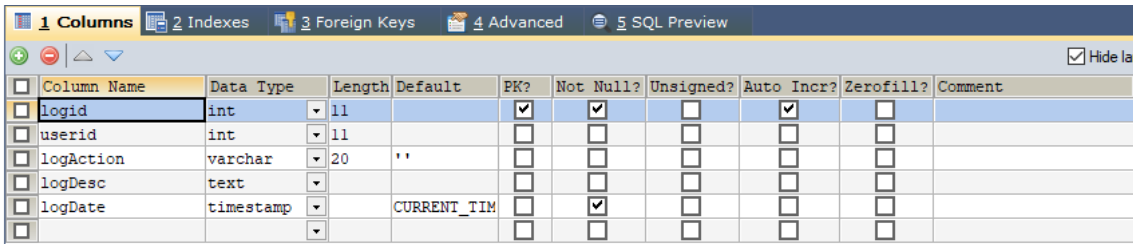
Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
HSAddressID	int	11		☑	☑	☐	☑	☐	
schoolName	varchar	50	''	☐	☐	☐	☐	☐	
schoolDesc	varchar	80	''	☐	☐	☐	☐	☐	
province	varchar	50	''	☐	☐	☐	☐	☐	
cityTown	varchar	50	''	☐	☐	☐	☐	☐	
brgy	varchar	50	''	☐	☐	☐	☐	☐	
stUnit	varchar	50	''	☐	☐	☐	☐	☐	
zipcode	varchar	5	''	☐	☐	☐	☐	☐	
				☐	☐	☐	☐	☐	

Figure 31. hs_address



Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
logoid	int	11		✓	✓		✓		
logo	longblob								
header	longblob								
active	tinyint	4							

Figure 32. logo



Column Name	Data Type	Length	Default	PK?	Not Null?	Unsigned?	Auto Incr?	Zerofill?	Comment
logid	int	11		✓	✓		✓		
userid	int	11							
logAction	varchar	20	''						
logDesc	text								
logDate	timestamp		CURRENT_TIM		✓				

Figure 33. loghistory

Entity Relationship Diagram

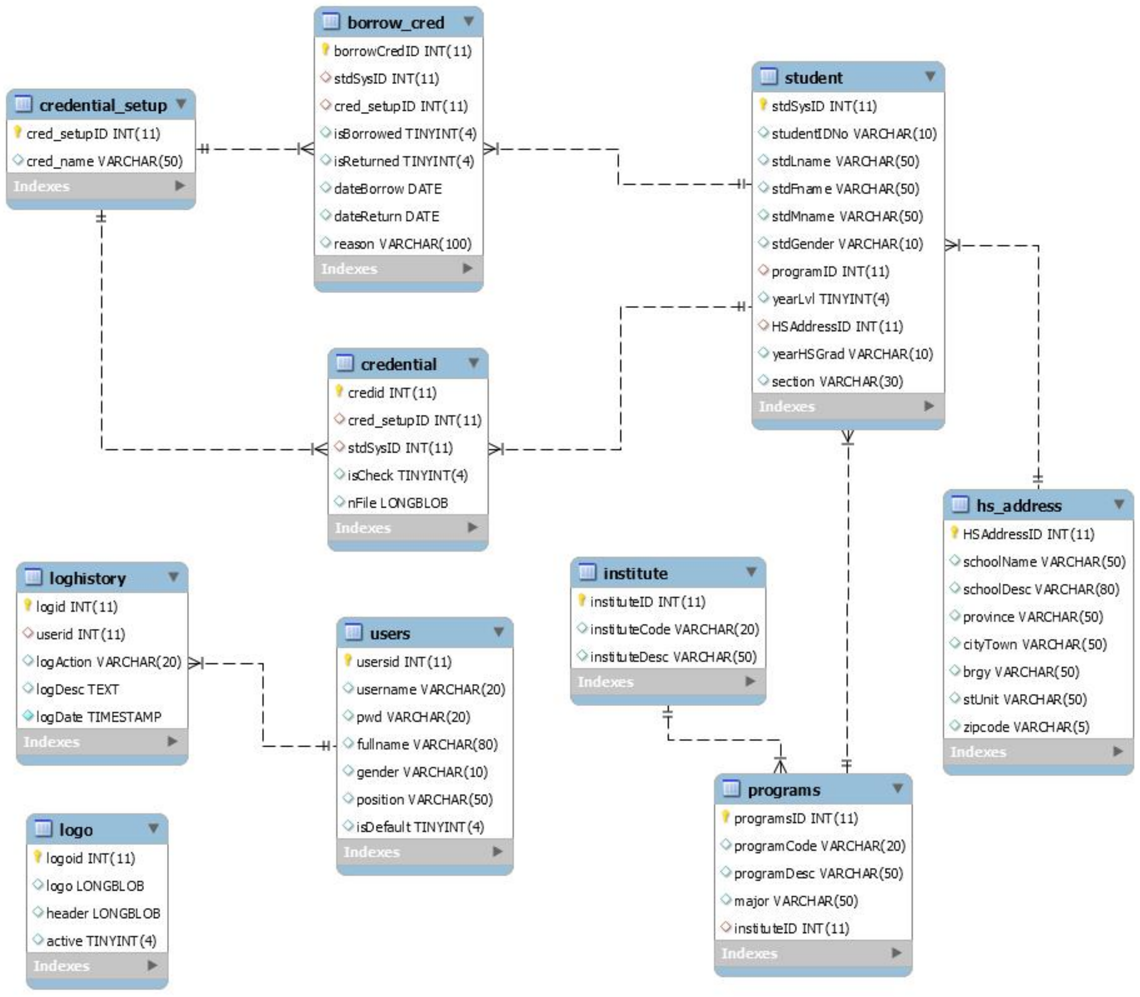


FIGURE 34. Entity Relationship Diagram

Hierarchical Input Process Output (HIPO)

Administrator

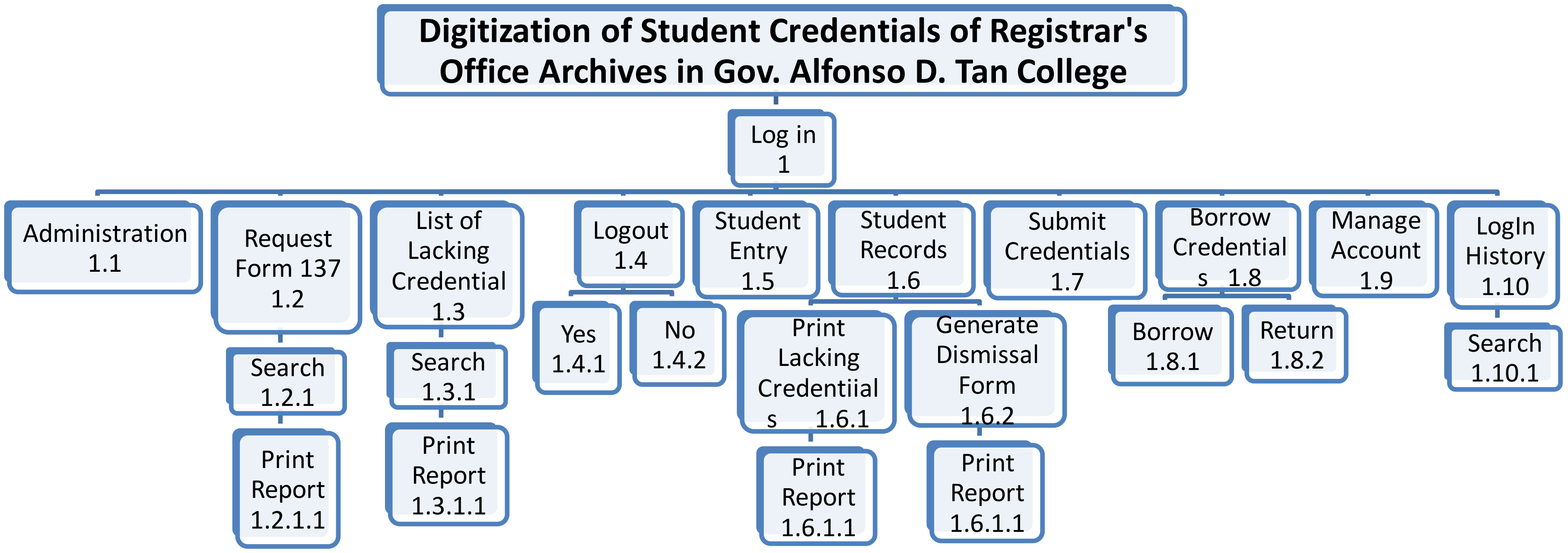


FIGURE 35. HIPO (Administrator)

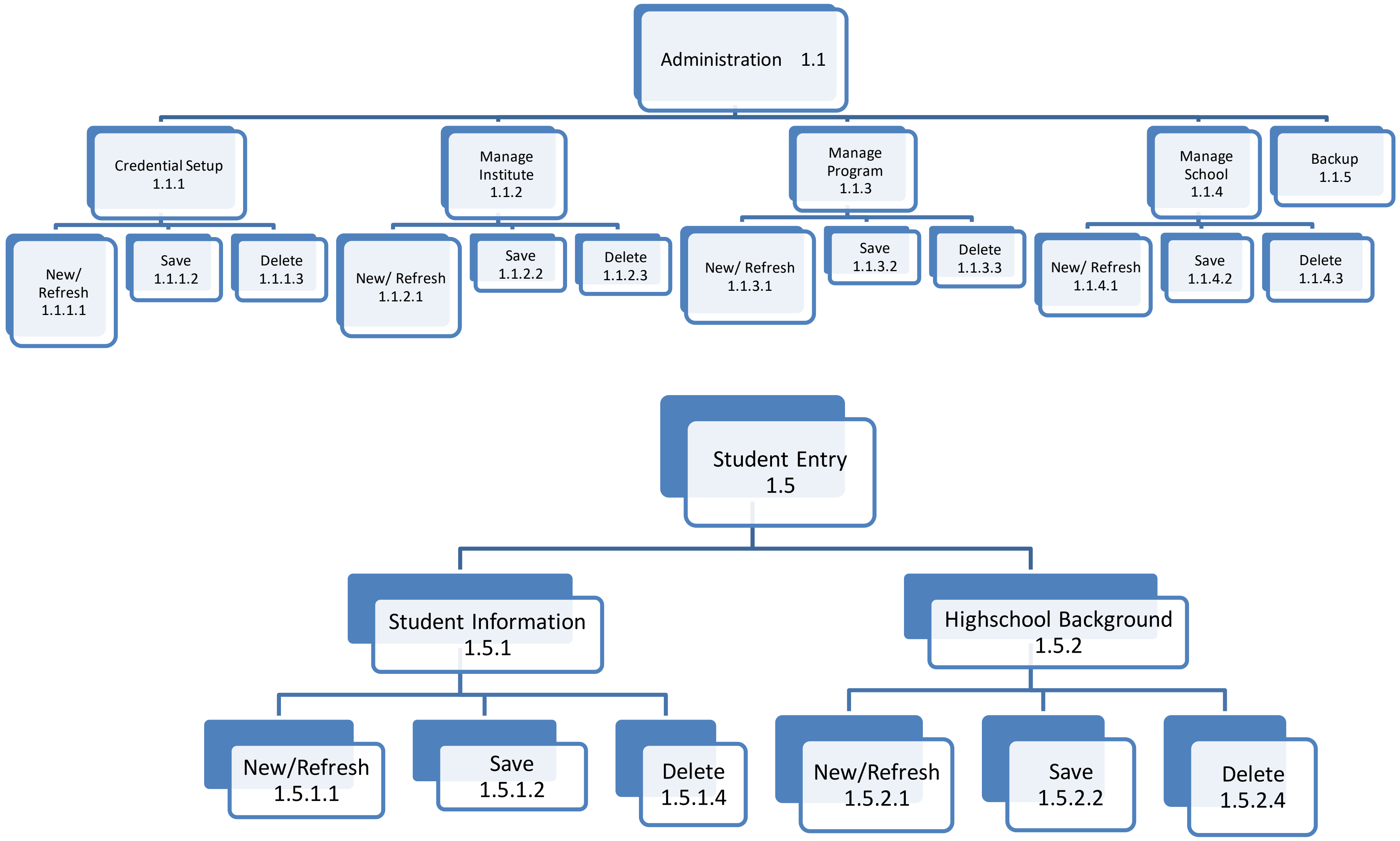
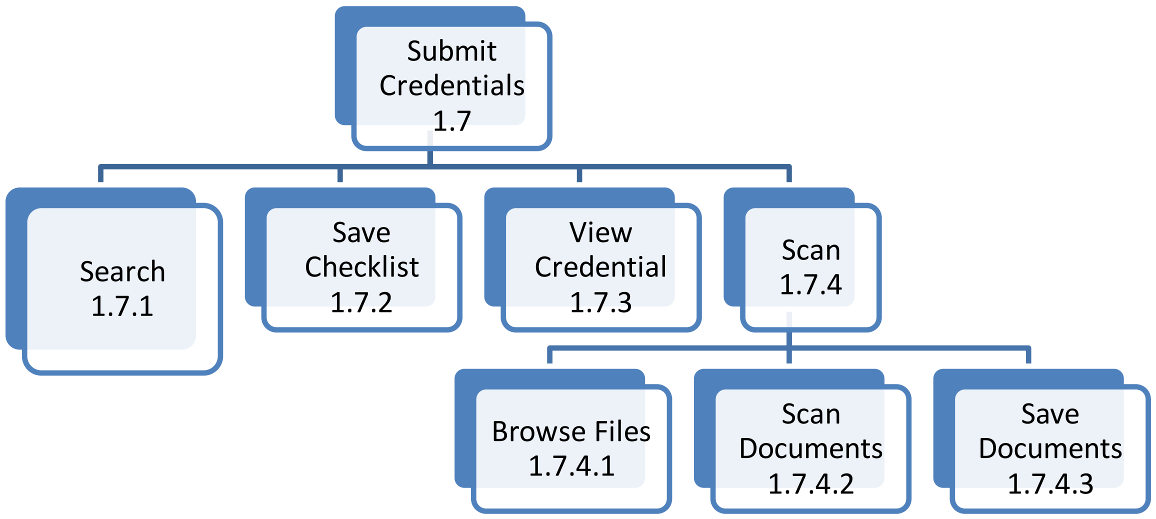


FIGURE 36. HIPO Administration



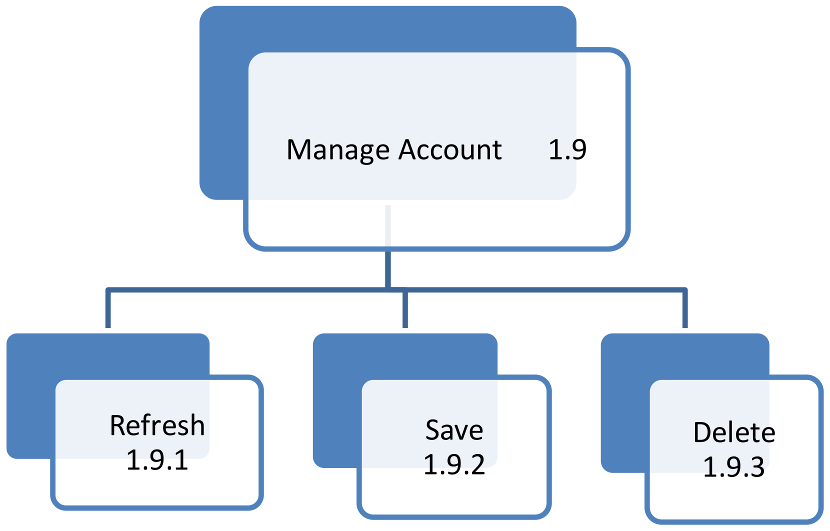


FIGURE 37. HIPO (Administrator)

Staff

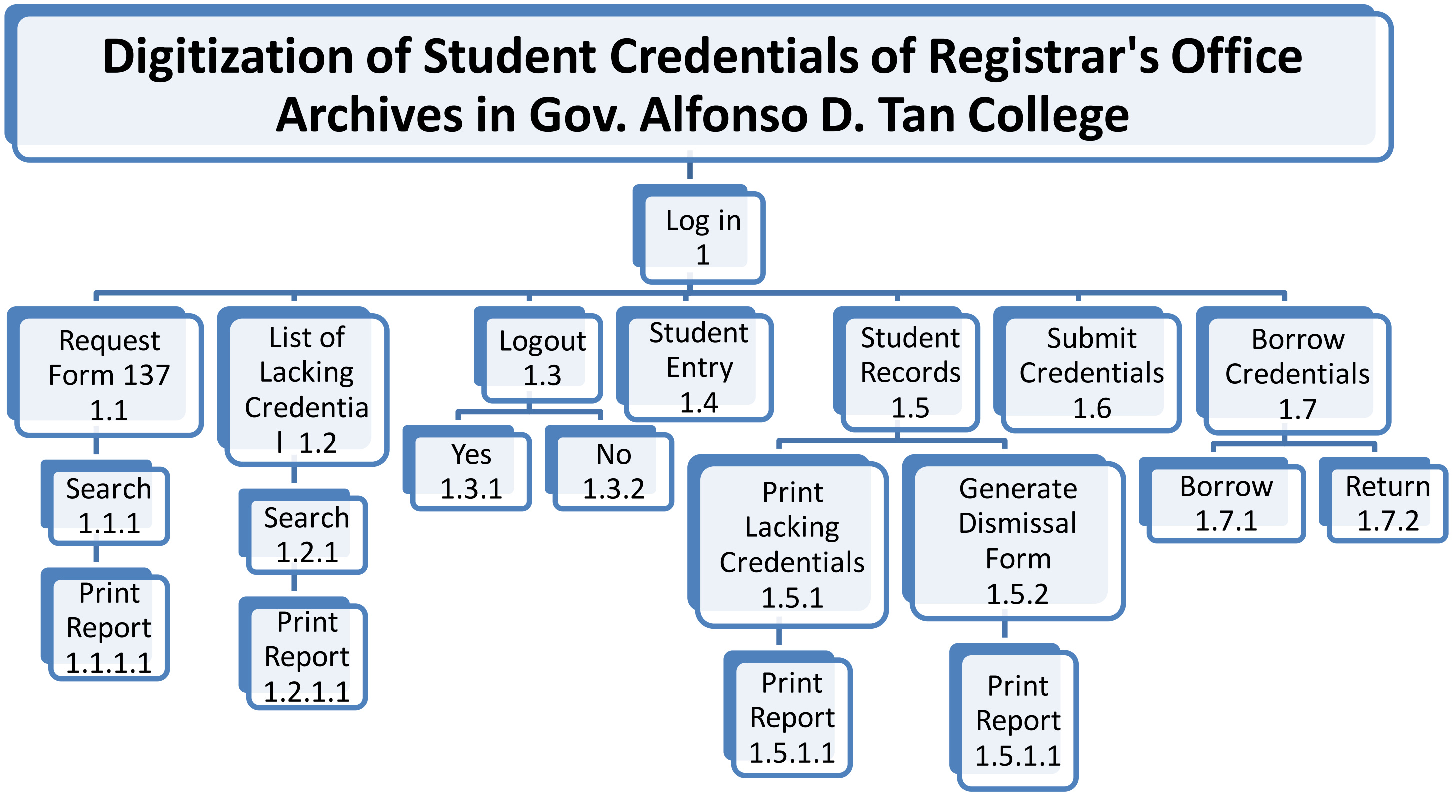


FIGURE 38. HIPO (Staff)

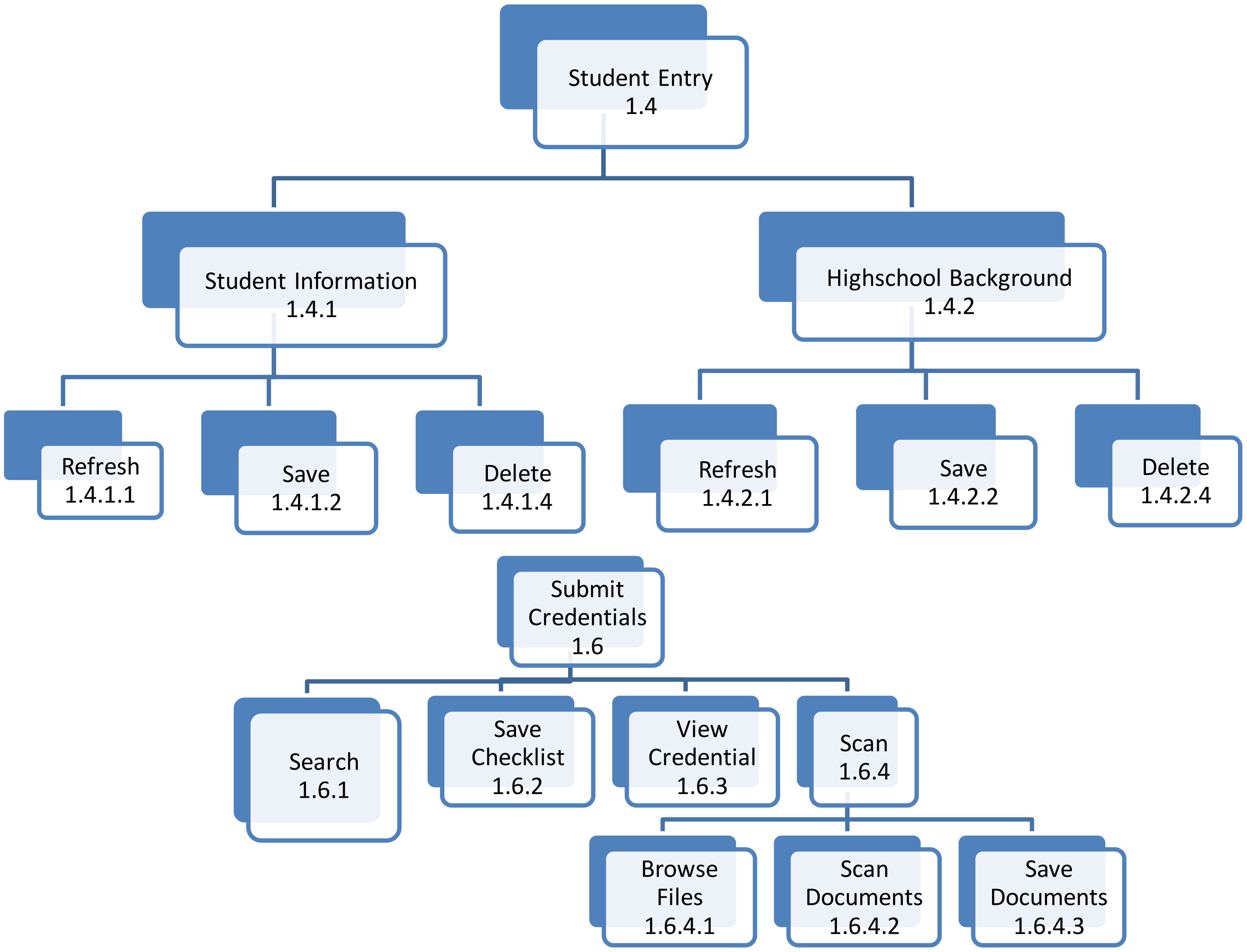


FIGURE 39. Hierarchical Input Process Output (Staff)

Technical Background

The Digitization of Student Credentials of Registrar’s Office Archives in Gov. Alfonso D. Tan College provides ease in managing supporting documents of students of the college. This system allows the registrar’s office staff to scan and store supporting documents of the students into the system.

In order to realize our study, we purchase a computer with a minimum specification of core i3 processor and 4 GB RAM with 2pcs 2TB hard disk for memory storage and backup with a minimum OS - Windows 7. To support the system, end-user must install the following software: .NET Framework 4.5.1, SAP Crystal Report Runtime. We also purchase a multi-printer machine that can scan paper documents to be stored in the system. Further, we use C# as our programming languages and MySQL as database.

CHAPTER 4

RESULTS AND DISCUSSION

Results

Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College is a system that is design to store and retrieve records of the students, by digitizing the paper documents and storing it in an organized manner. It provides user-friendly interface that is easier for the staff of the Registrar's Office to access. Aside from storing documents, the system can also generate reports. Based on the gathered data, it was found out that the respondents encountered problems in accessing further retrieving the documents, especially those that were stored decades ago. It was pointed out that accessing the said documents is time consuming and is prone to damage since the documents are now so fragile. Moreover, the staff of the Registrar's Office involved in the process agreed that digitizing documents of the office would be very essential.

Discussion

As development over the past few years have shown, we are in a time of unprecedented technological growth with the use of technology, everything progresses specially in terms of managing records or documents in any organization. We have come up with a study: “Digitization of Student Credentials of Registrar’s Office Archives in Gov. Alfonso D. Tan College” in order to provide security of the important documents of the college. Further, it makes easier for the employees in managing such documents. It likewise reduces proper record handling and time-consuming process. To put the study into reality, the researchers conducted an interview with the head of the Registrar’s Office of Gov. Alfonso D. Tan College. An interview was conducted and a standardized questionnaire were distributed to the said respondents. The researchers also assured that they could raise honestly their concerns towards the proposed system.

CHAPTER 5

CONCLUSION AND RECOMMENDATIONS

Conclusion

Based on the results of the study, the researchers concluded that digitizing the documents of Registrar's Office of Gov. Alfonso D. Tan College is very beneficial. It is essential in order to resolve the problem of the office concerning accessing and retrieving the old documents that is now fragile and prone to damage. With this system, managing the student credentials is now easier, faster, and more effective. Since the documents will be stored in the database, it will be always secured and available for students and anyone concerned. Indeed, this will greatly help not only the Registrar's Office but also the entire community of Gov. Alfonso D. Tan College.

Recommendations

Based on the results of the study, the proponents recommended the following.

1.) Gov. Alfonso D. Tan College should consider the implementation of the Digitization of Student Credentials of Registrar's Office Archives to help them in their record keeping.

2.) Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College should have SMS notification.

3.) Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College must be maintained regularly.

4.) Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College should have reminder for unreturned credentials after its due date.

Appendix A

Digitization of Student Credentials of Registrar's Office Archives in Gov. Alfonso D. Tan College User's Manual

1. Getting Started (Log in)

To use the system the user must first log in into the system by inputting their username and password.

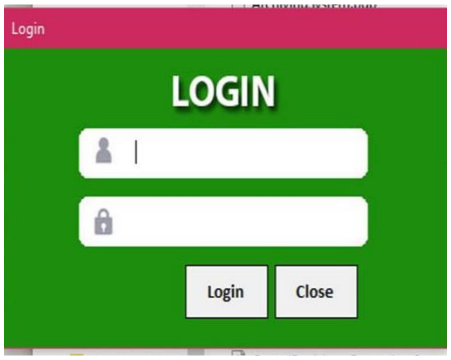
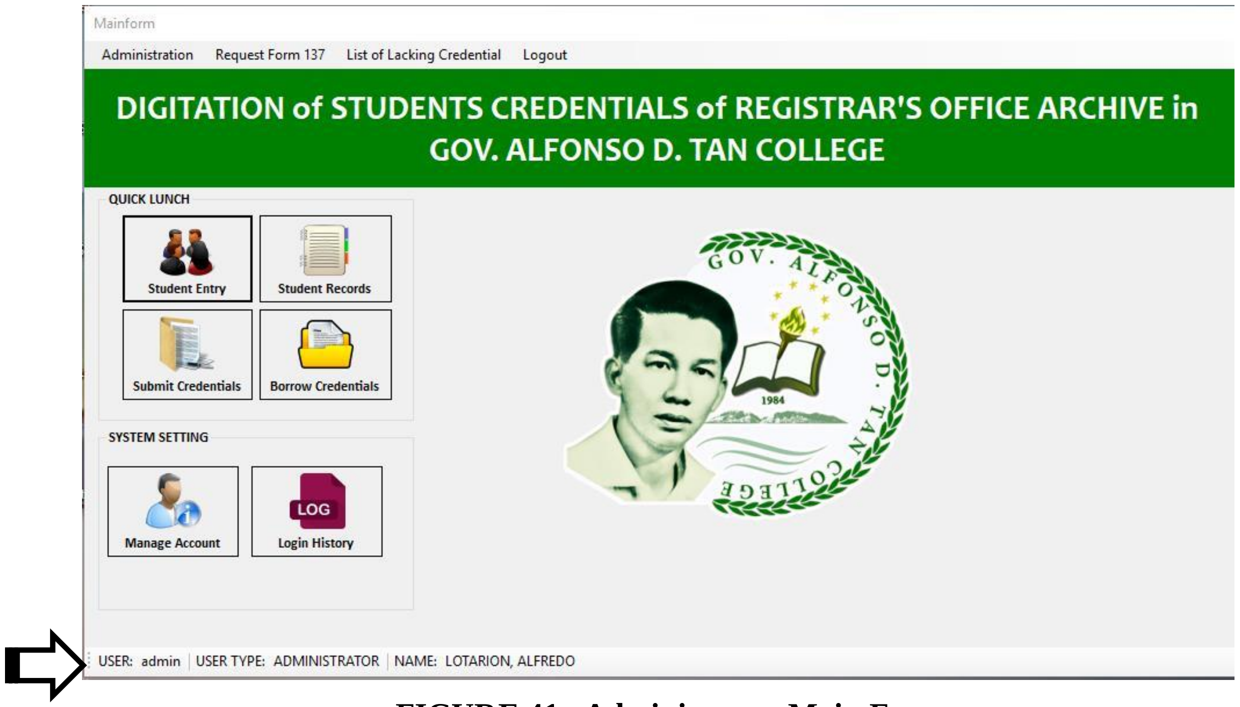


FIGURE 40. Log in

- The system will then show the main form.

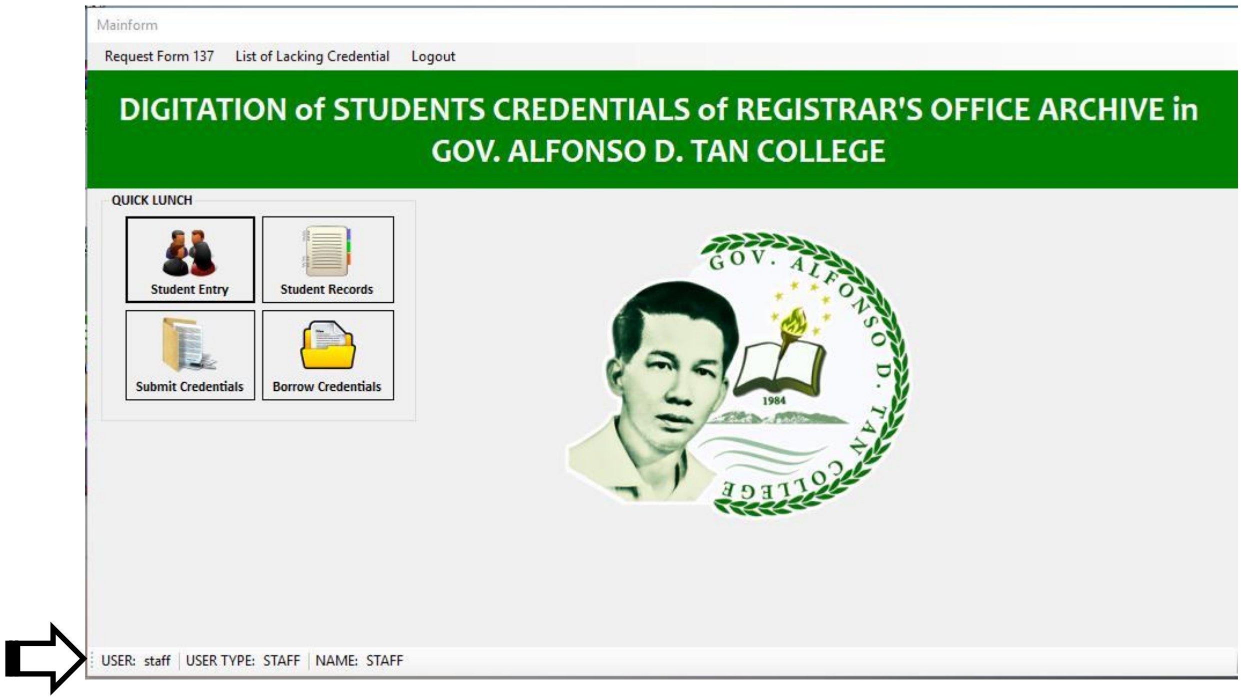
Administrator



USER: admin | USER TYPE: ADMINISTRATOR | NAME: LOTARION, ALFREDO

FIGURE 41. Administrator Main Form

Staff



DIGITATION of STUDENTS CREDENTIALS of REGISTRAR'S OFFICE ARCHIVE in GOV. ALFONSO D. TAN COLLEGE

USER: staff | USER TYPE: STAFF | NAME: STAFF

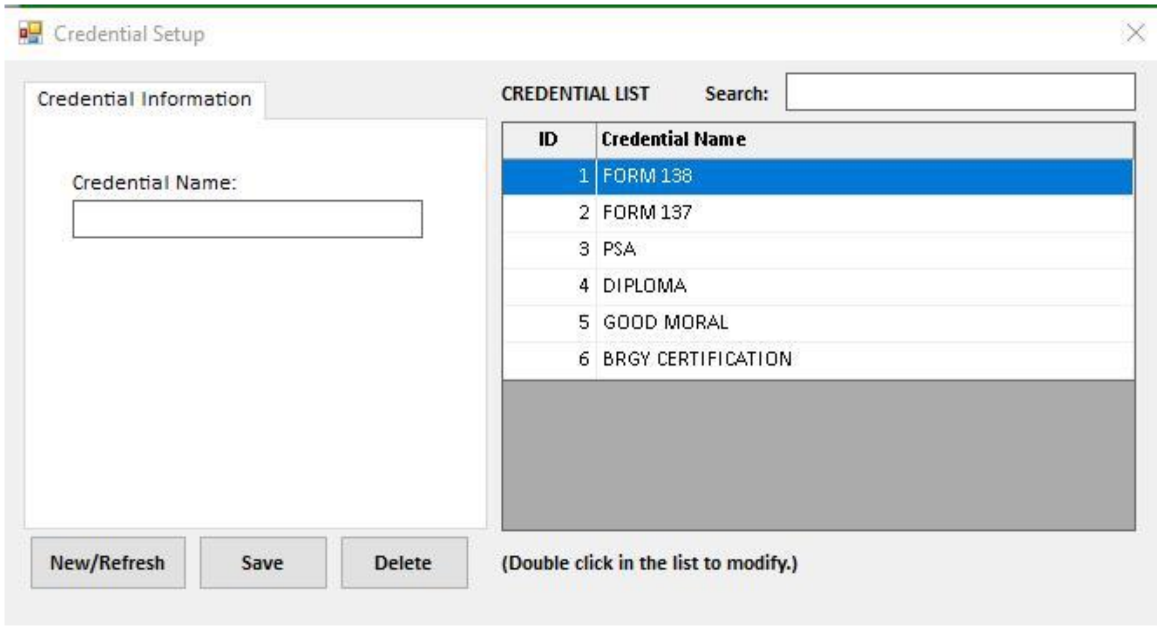
FIGURE 42. Staff Main Form

- The arrow shows the indication of the user's information who logged in.

2. Administration Menu (Administrator Only)

   a) Credential Setup

      - New – Input the credentials name then click save.
      - Edit – Search and Double click credential in the list, then modify credential name. Then click save.
      - Delete - Search and Double click credential in the list, then click delete.
      - Further, if the Admin wants to create new just click new/refresh.



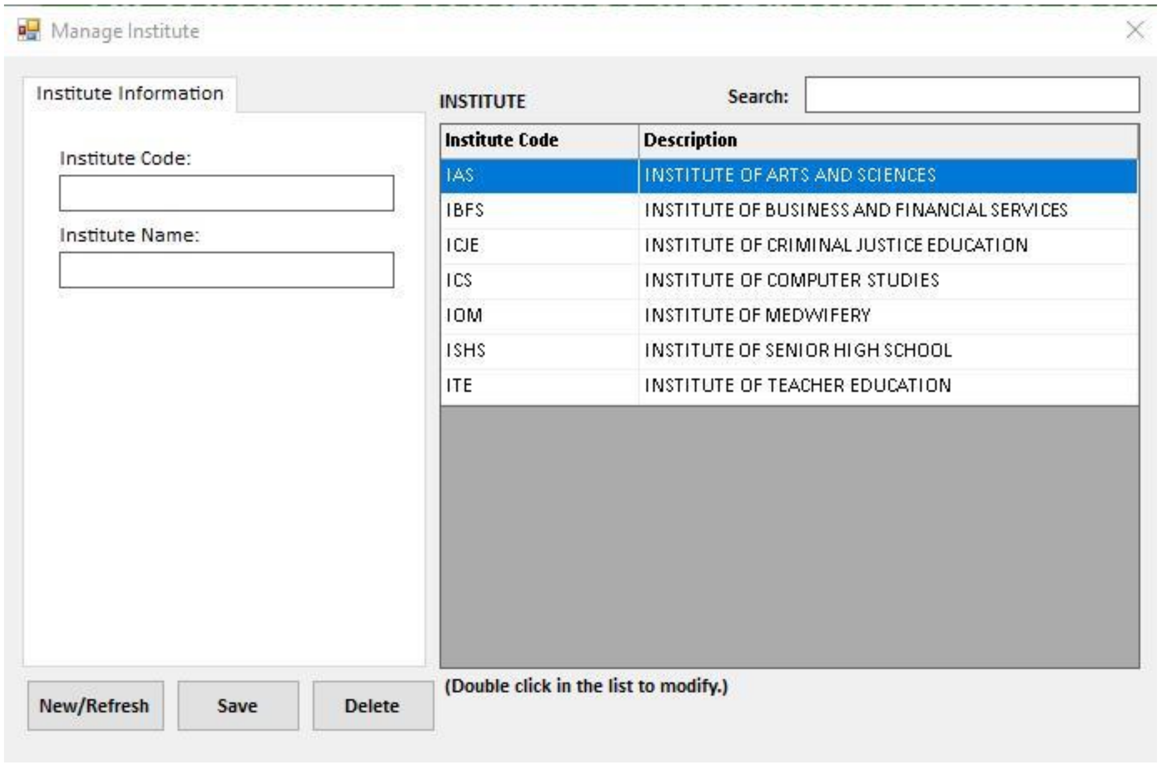
ID	Credential Name
1	FORM 138
2	FORM 137
3	PSA
4	DIPLOMA
5	GOOD MORAL
6	BRGY CERTIFICATION

(Double click in the list to modify.)

FIGURE 43. Credential Setup

b) Manage Institute

- New – Input Institute code and name then click save.
- Edit – Search and Double click institute in the institute list, then modify institute information. Then click save.
- Delete - Search and Double click institute in the institute list, then click delete.
- Further, if the Admin wants to create new just click new/refresh.

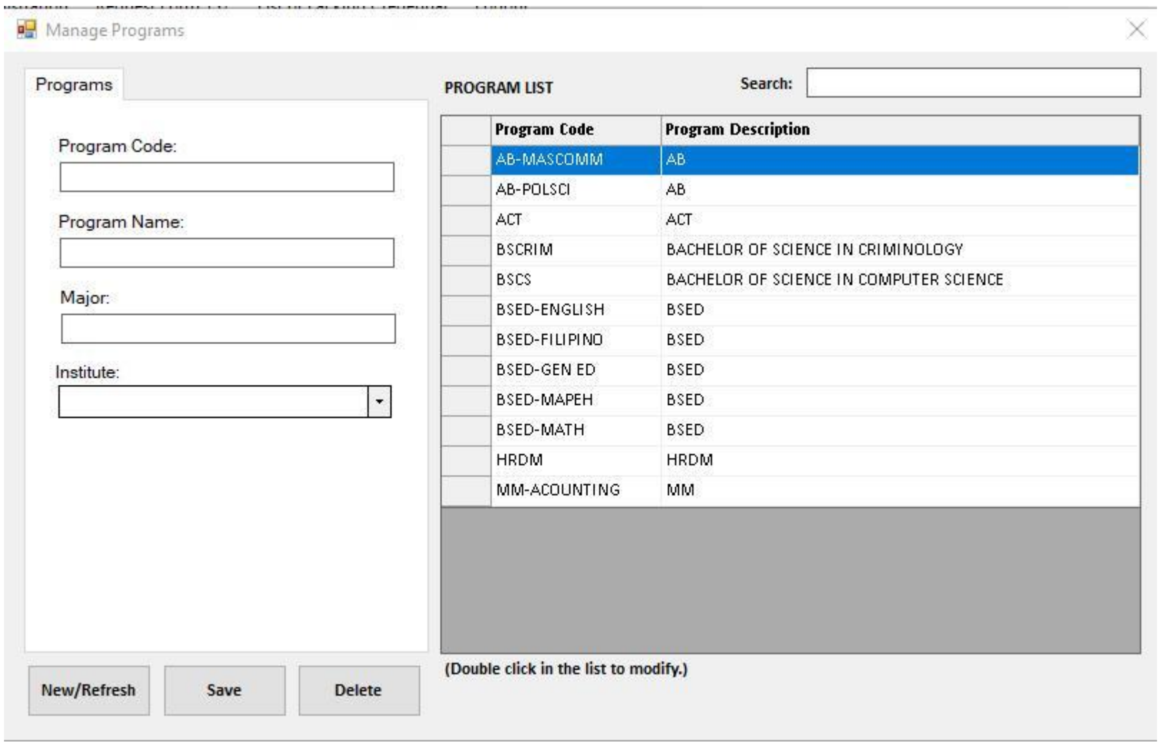


Institute Code	Description
IAS	INSTITUTE OF ARTS AND SCIENCES
IBFS	INSTITUTE OF BUSINESS AND FINANCIAL SERVICES
ICJE	INSTITUTE OF CRIMINAL JUSTICE EDUCATION
ICS	INSTITUTE OF COMPUTER STUDIES
IOM	INSTITUTE OF MEDWIFERY
ISHS	INSTITUTE OF SENIOR HIGH SCHOOL
ITE	INSTITUTE OF TEACHER EDUCATION

FIGURE 44. Manage Institute Form

c) Manage Program

- New – Input program code, name and major, and then select institute where the program belongs. Then click save.
- Edit – Search and Double click program in the list, then modify program information. Then click save.
- Delete - Search and Double click program in the list, then click delete.
- Further, if the Admin wants to create new just click new/refresh.

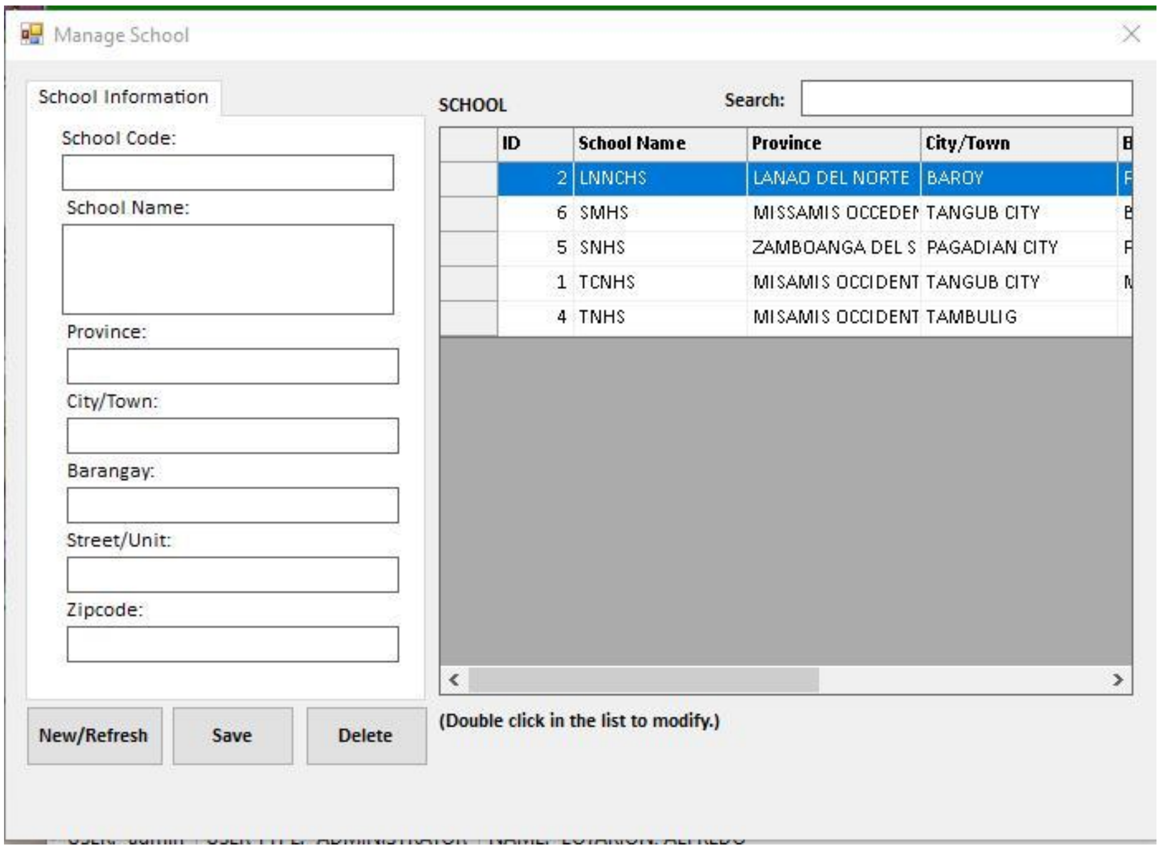


Program Code	Program Description
AB-MASCOMM	AB
AB-POLSCI	AB
ACT	ACT
BSCRIM	BACHELOR OF SCIENCE IN CRIMINOLOGY
BSCS	BACHELOR OF SCIENCE IN COMPUTER SCIENCE
BSED-ENGLISH	BSED
BSED-FILIPINO	BSED
BSED-GEN ED	BSED
BSED-MAPEH	BSED
BSED-MATH	BSED
HRDM	HRDM
MM-ACOUNTING	MM

FIGURE 45. Manage Program Form

d) Manage School

- New – Input school information then click save.
- Edit – Search and Double click school in the list, then modify school information. Then click save.
- Delete - Search and Double click school in the list, then click delete.
- Further, if the Admin wants to create new just click new/refresh.



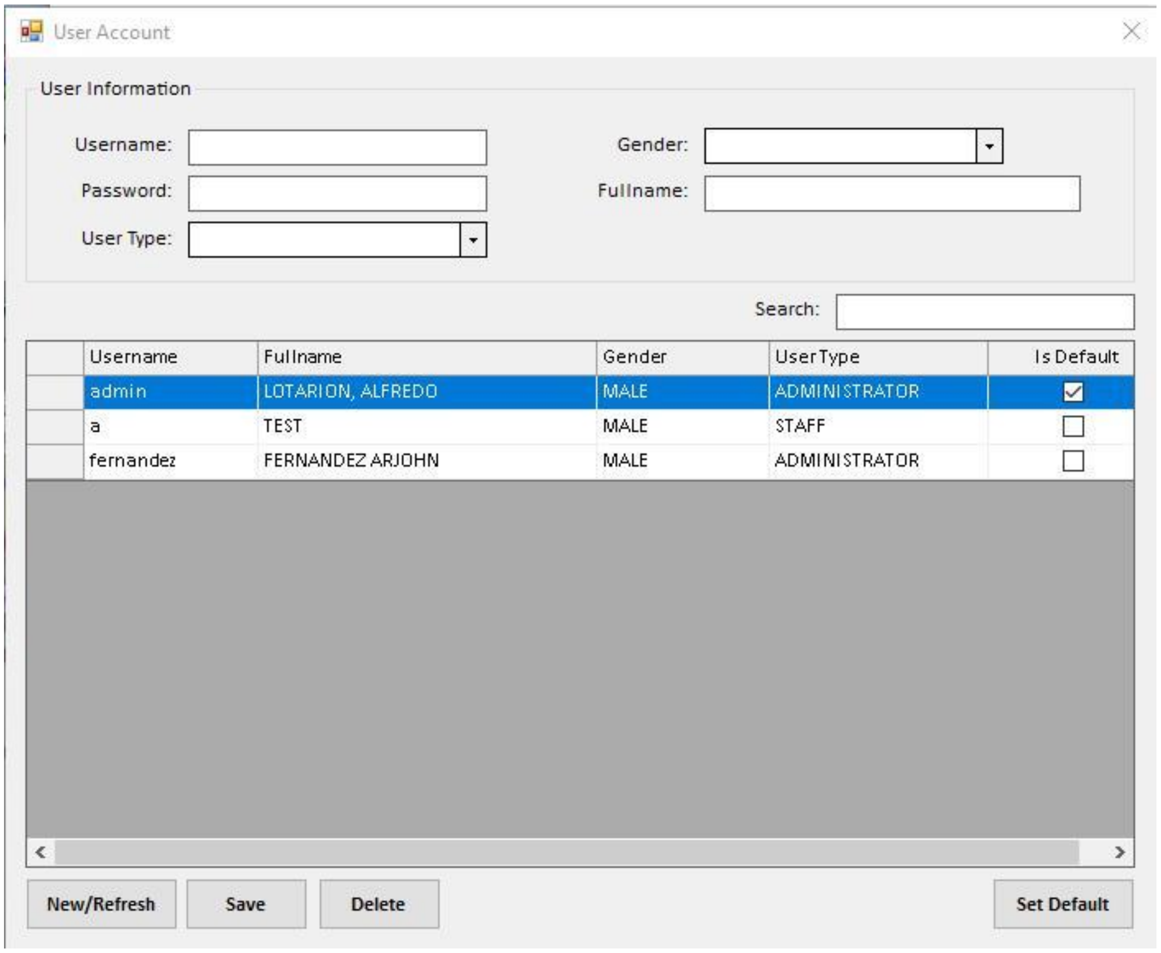
ID	School Name	Province	City/Town	B
2	LNNCHS	LANAO DEL NORTE	BAROY	F
6	SMHS	MISSAMIS OCCEDEI	TANGUB CITY	B
5	SNHS	ZAMBOANGA DEL S	PAGADIAN CITY	F
1	TCNHS	MISAMIS OCCIDENT	TANGUB CITY	M
4	TNHS	MISAMIS OCCIDENT	TAMBULIG	

(Double click in the list to modify.)

FIGURE 46. Manage School Form

3. Manage Account (Administrator Only)

- New – Input user information then click save.
- Edit – Search and Double click user in the list, then modify user information. Then click save.
- Delete - Search and Double click user in the list, then click delete.
- Further, if the Admin wants to create new just click new/refresh.

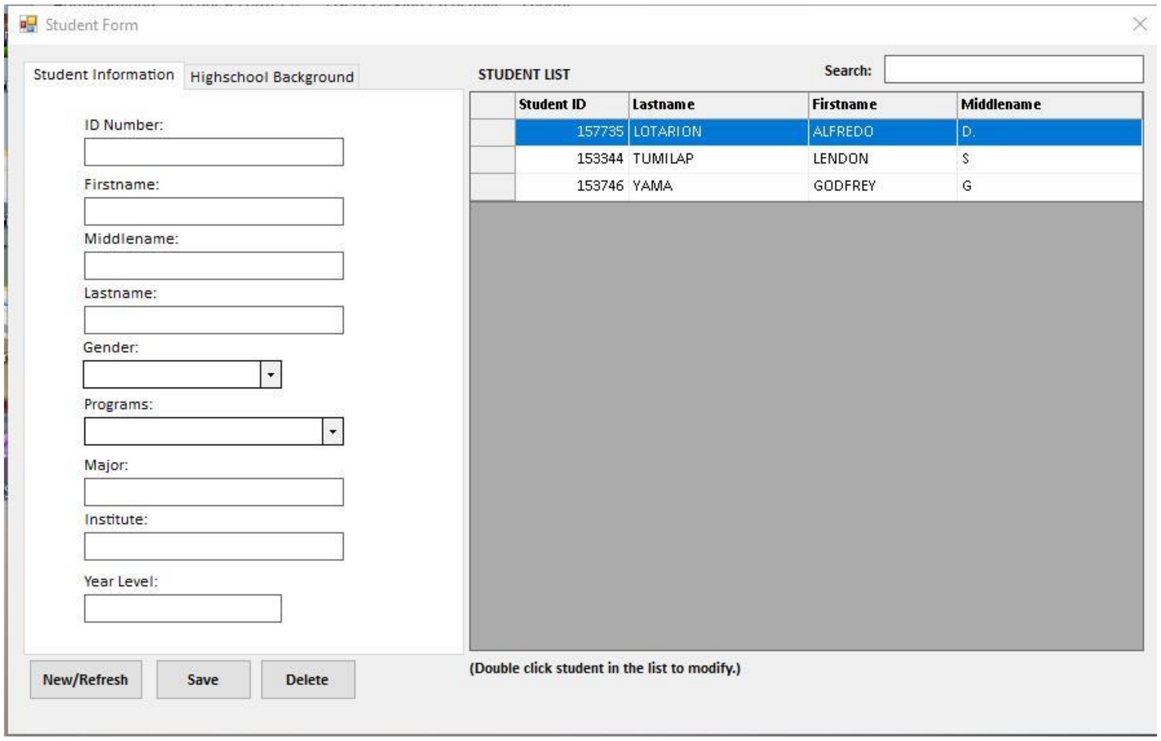


Username	Fullname	Gender	UserType	Is Default
admin	LOTARION, ALFREDO	MALE	ADMINISTRATOR	☑
a	TEST	MALE	STAFF	☐
fernandez	FERNANDEZ ARJOHN	MALE	ADMINISTRATOR	☐

FIGURE 47. Manage Account Form

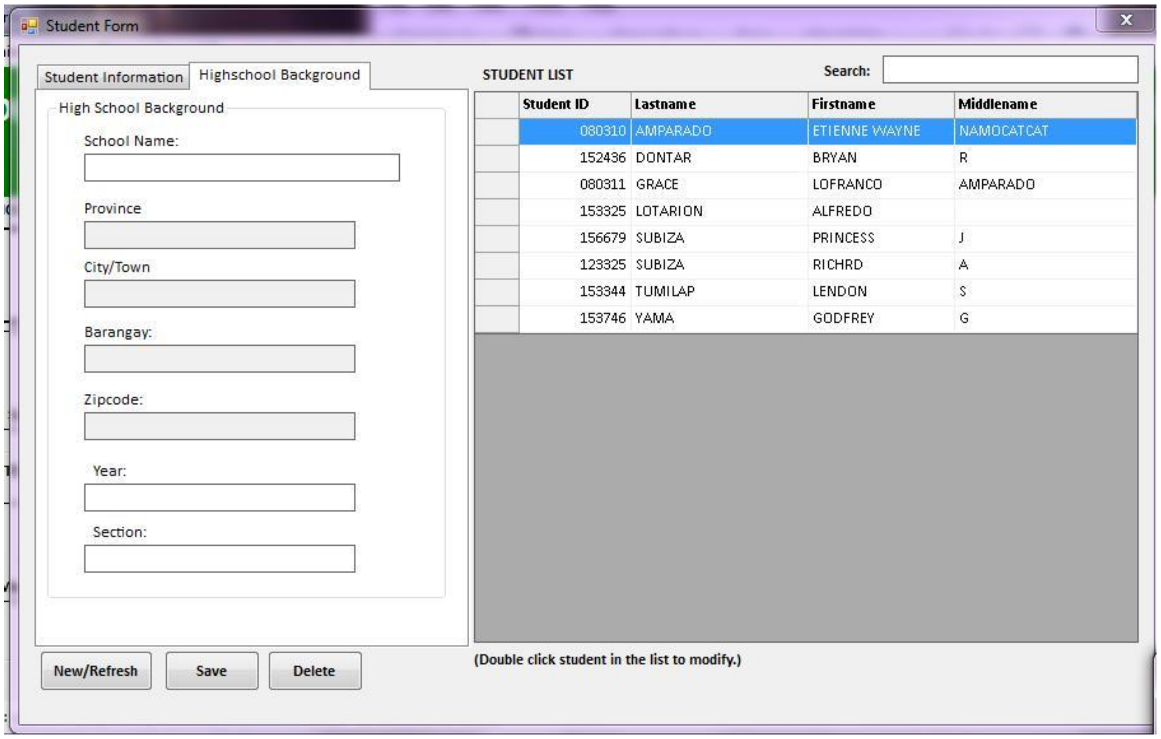
4. Student Entry

- New – Input student information and high school background then click save.
- Edit – Search and Double click student in the list, then modify student information. Then click save.
- Delete - Search and Double click student in the list, then click delete.
- Further, if the user want to create new just click new/refresh.



Student ID	Lastname	Firstname	Middlename
157735	LOTARION	ALFREDO	D.
153344	TUMILAP	LENDON	S
153746	YAMA	GODFREY	G

FIGURE 48. Student Information Form



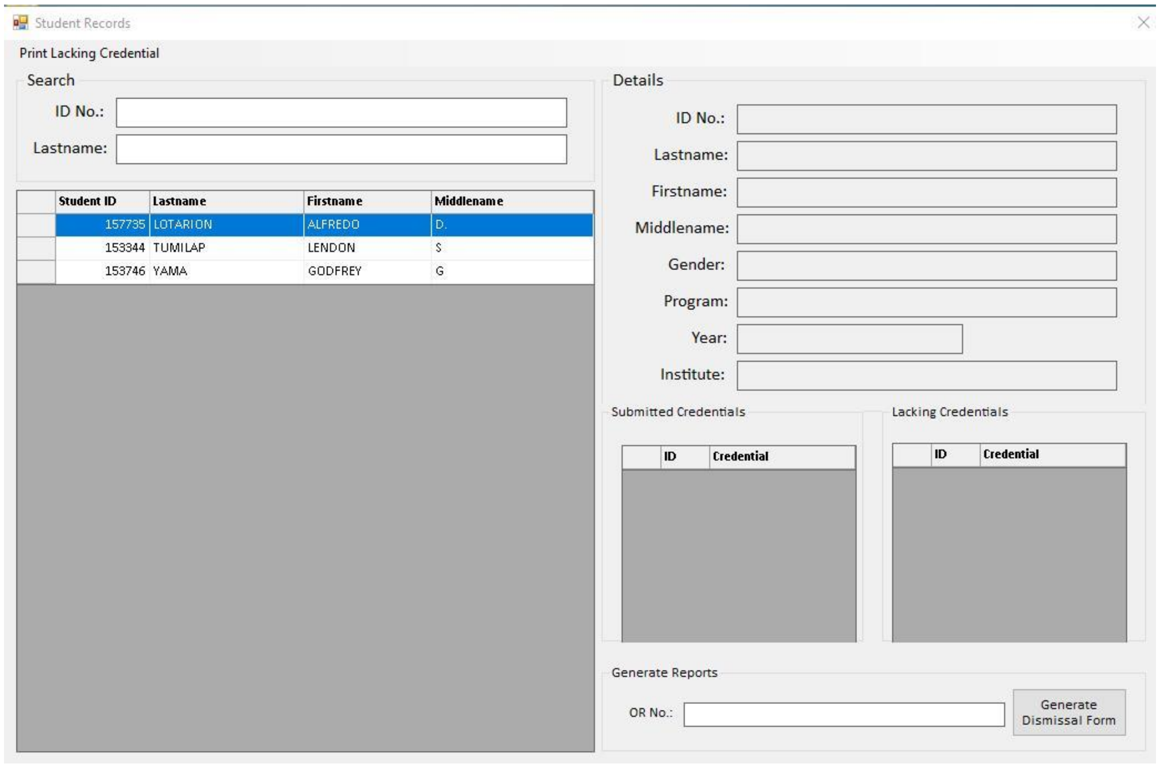
Student ID	Lastname	Firstname	Middlename
080310	AMPARADO	ETIENNE WAYNE	NAMOCATCAT
152436	DONTAR	BRYAN	R
080311	GRACE	LOFRANCO	AMPARADO
153325	LOTARION	ALFREDO	
156679	SUBIZA	PRINCESS	J
123325	SUBIZA	RICHRD	A
153344	TUMILAP	LENDON	S
153746	YAMA	GODFREY	G

(Double click student in the list to modify.)

FIGURE 49. Student High School Background Form

5. Student Records

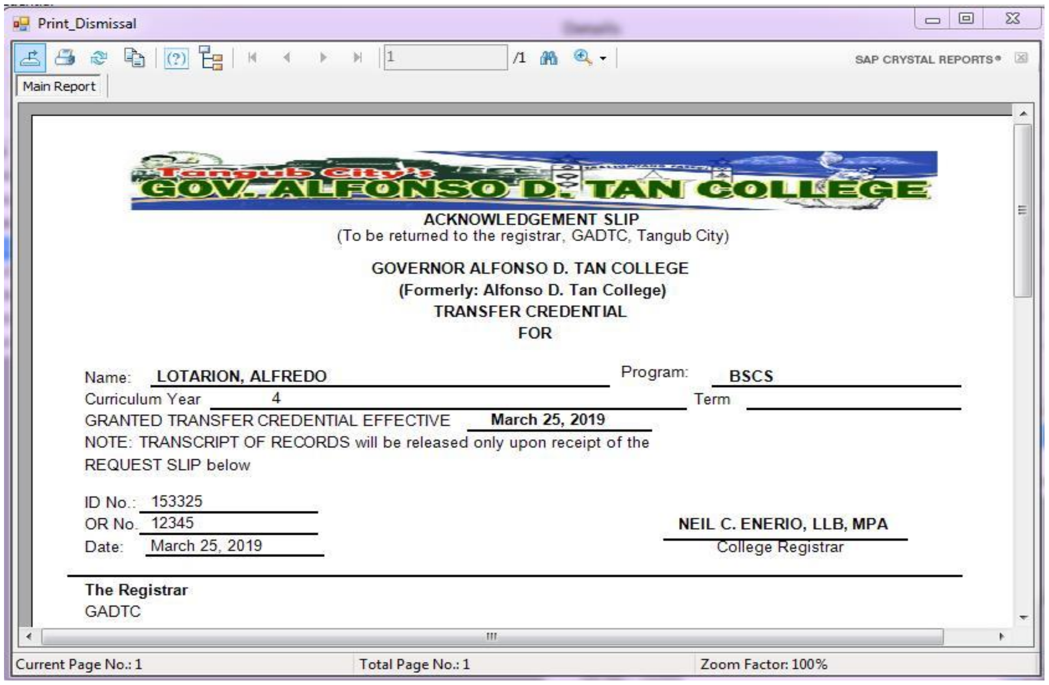
a) Search student I.D. or Last Name then press enter.



Student ID	Lastname	Firstname	Middlename
157735	LOTARION	ALFREDO	D.
153344	TUMILAP	LENDON	S
153746	YAMA	GODFREY	G

FIGURE 50. Student Records Form

b) Click "Generate Dismissal Form" to generate the form and then print.



ACKNOWLEDGEMENT SLIP
(To be returned to the registrar, GADTC, Tangub City)

GOVERNOR ALFONSO D. TAN COLLEGE
(Formerly: Alfonso D. Tan College)
TRANSFER CREDENTIAL
FOR

Name: LOTARION, ALFREDO Program: BSCS
Curriculum Year 4 Term
GRANTED TRANSFER CREDENTIAL EFFECTIVE March 25, 2019
NOTE: TRANSCRIPT OF RECORDS will be released only upon receipt of the REQUEST SLIP below

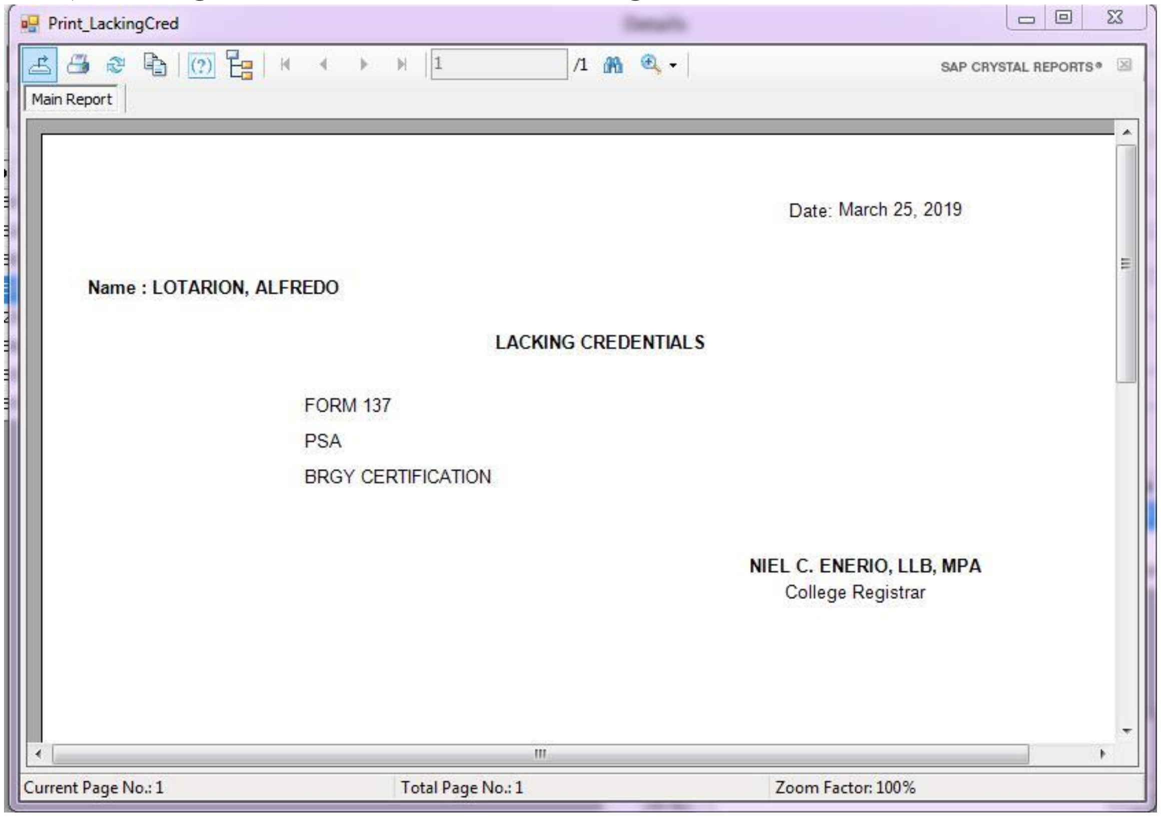
ID No.: 153325
OR No. 12345
Date: March 25, 2019

NEIL C. ENERIO, LLB, MPA
College Registrar

The Registrar
GADTC

FIGURE 51. Print Dismissal Form

c) Printing of Student Credential Monitoring Sheet



Date: March 25, 2019

Name : LOTARION, ALFREDO

LACKING CREDENTIALS

FORM 137
PSA
BRGY CERTIFICATION

NIEL C. ENERIO, LLB, MPA
College Registrar

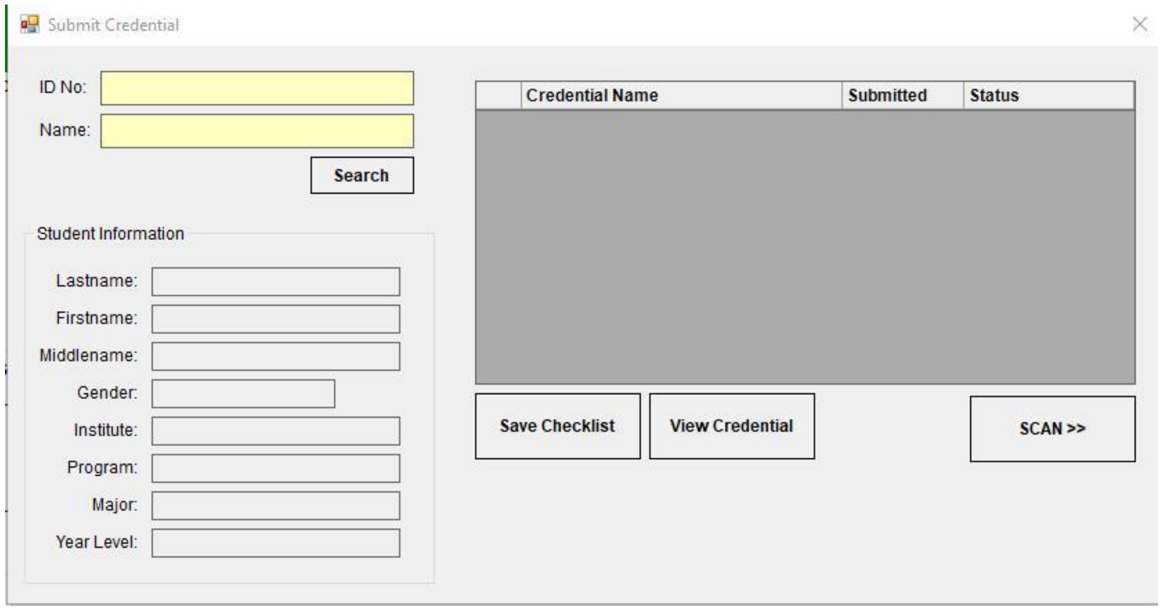
Current Page No.: 1 Total Page No.: 1 Zoom Factor: 100%

FIGURE 52. Print Student Credential Monitoring Sheet

6. Submit Student Credentials

   a) Save CheckList

      - Search for the student I.D. and name.
      - The user can check the credential that was submitted by the students but it would still not mark completed if the said document is not scanned. It is only an indicator that the student already submitted it in the office but not in the system. The user can check as many as he/she want. After checking the box click Save CheckList.

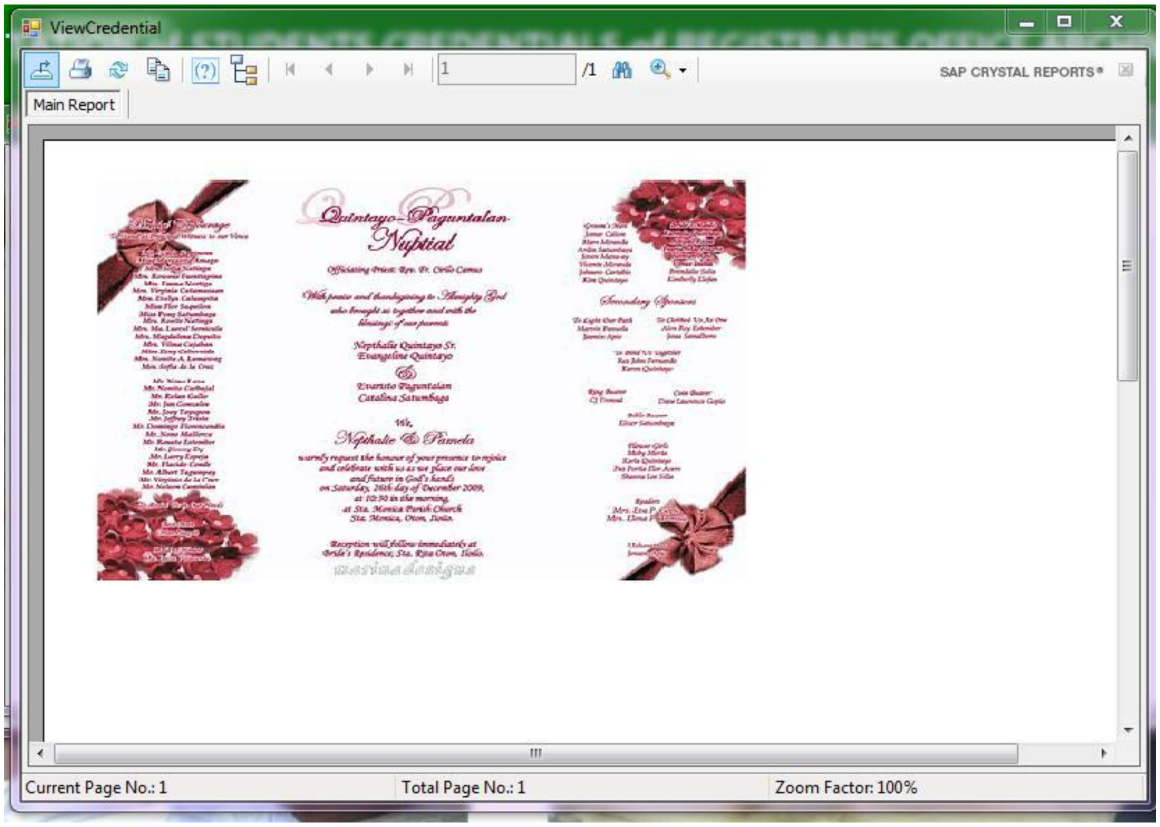


	Credential Name	Submitted	Status

FIGURE 53. Submit Credential Form

b) View Credential

- Search for the student I.D. and name.
- The user will select a credential and then click view Credential.



Quintayo–Paguntalan Nuptial

Officiating Priest: Rev. Fr. Cirilo Camus

With praise and thanksgiving to Almighty God
who brought us together and with the
blessings of our parents

Nepthalie Quintayo Sr.
Evangeline Quintayo
&
Evaristo Paguntalan
Catalina Satumbaga

We,
Nepthalie & Pamela
warmly request the honour of your presence to rejoice
and celebrate with us as we place our love
and future in God's hands
on Saturday, 26th day of December 2009,
at 10:30 in the morning
at Sta. Monica Parish Church
Sta. Monica, Oton, Iloilo.

Reception will follow immediately at
Bride's Residence, Sta. Rita Oton, Iloilo.

Current Page No.: 1

Total Page No.: 1

Zoom Factor: 100%

FIGURE 54. View Credential Form

c) Scan Student Credentials

- Search for the student I.D. and name.
- The user will select a credential then click scan.
- If there is a scanned document, browse the file.
- Then click Save Document.

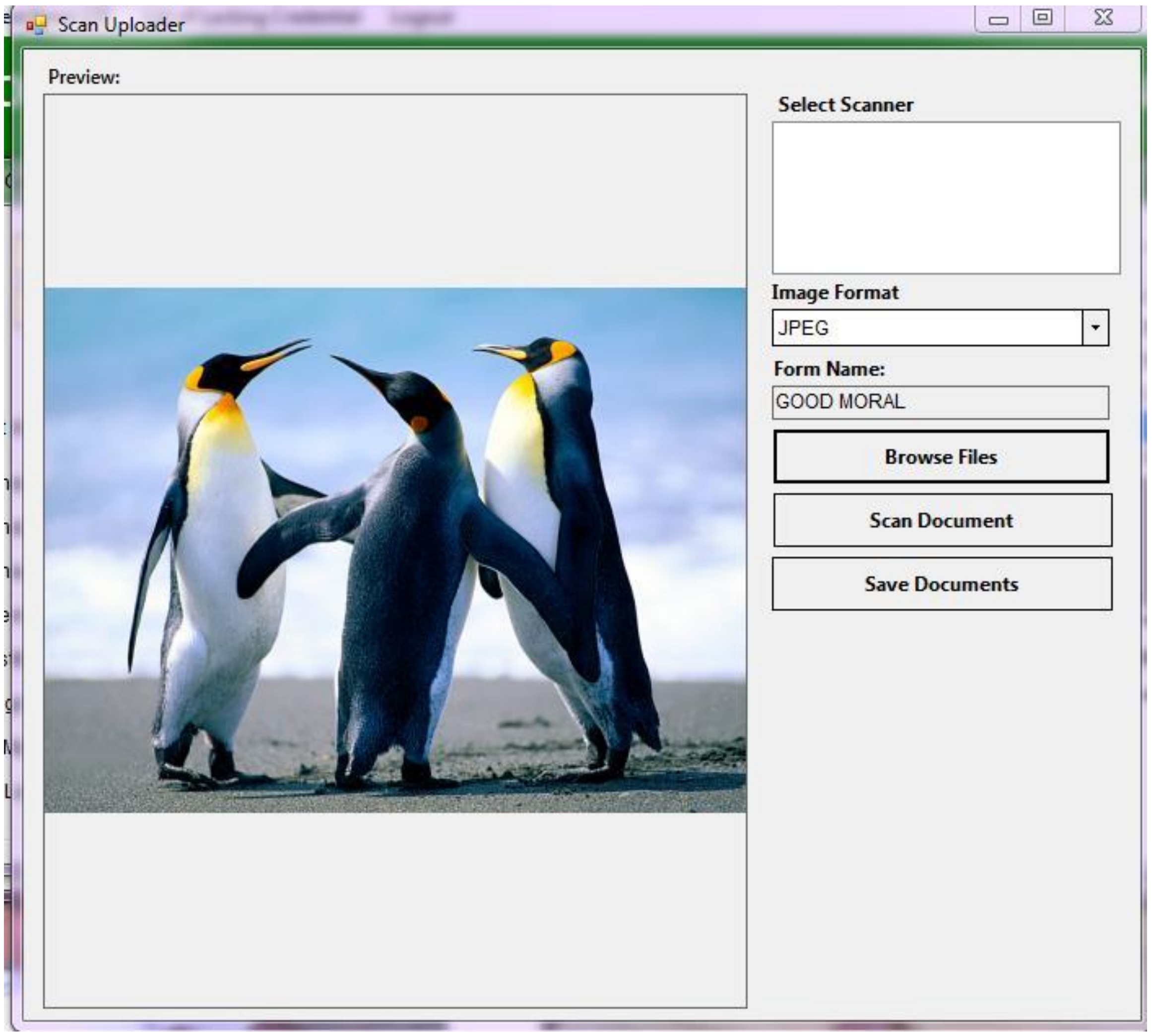


Image Format: JPEG

Form Name: GOOD MORAL

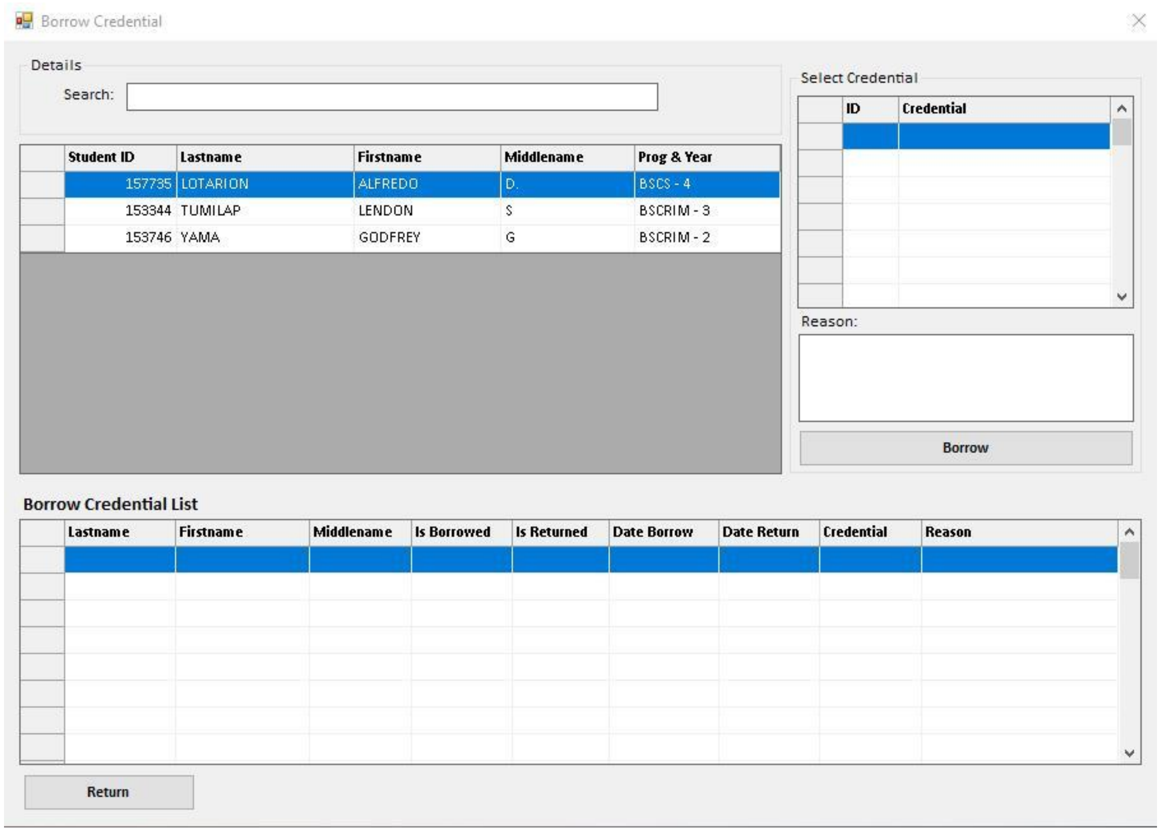
FIGURE 55. Scan Document

7. Manage Borrowed Credentials

   a) Borrow - Search student and then select credential to borrow. The credentials must already have record in the system that was already submitted by the students.

      - Note: There must be a reason. If none, the system will not record the credential to be borrow.

   b) Return - Search student and then select borrowed credential in the Borrow Credential List. Then Click Return.



Student ID	Lastname	Firstname	Middlename	Prog & Year
157735	LOTARION	ALFREDO	D.	BSCS - 4
153344	TUMILAP	LENDON	S	BSCRIM - 3
153746	YAMA	GODFREY	G	BSCRIM - 2

ID	Credential

Lastname	Firstname	Middlename	Is Borrowed	Is Returned	Date Borrow	Date Return	Credential	Reason

FIGURE 58. Borrow Credential Form

8. Manage Students with Lacking Credentials

   a) Select an Institute.

   b) Click Search to generate the List.

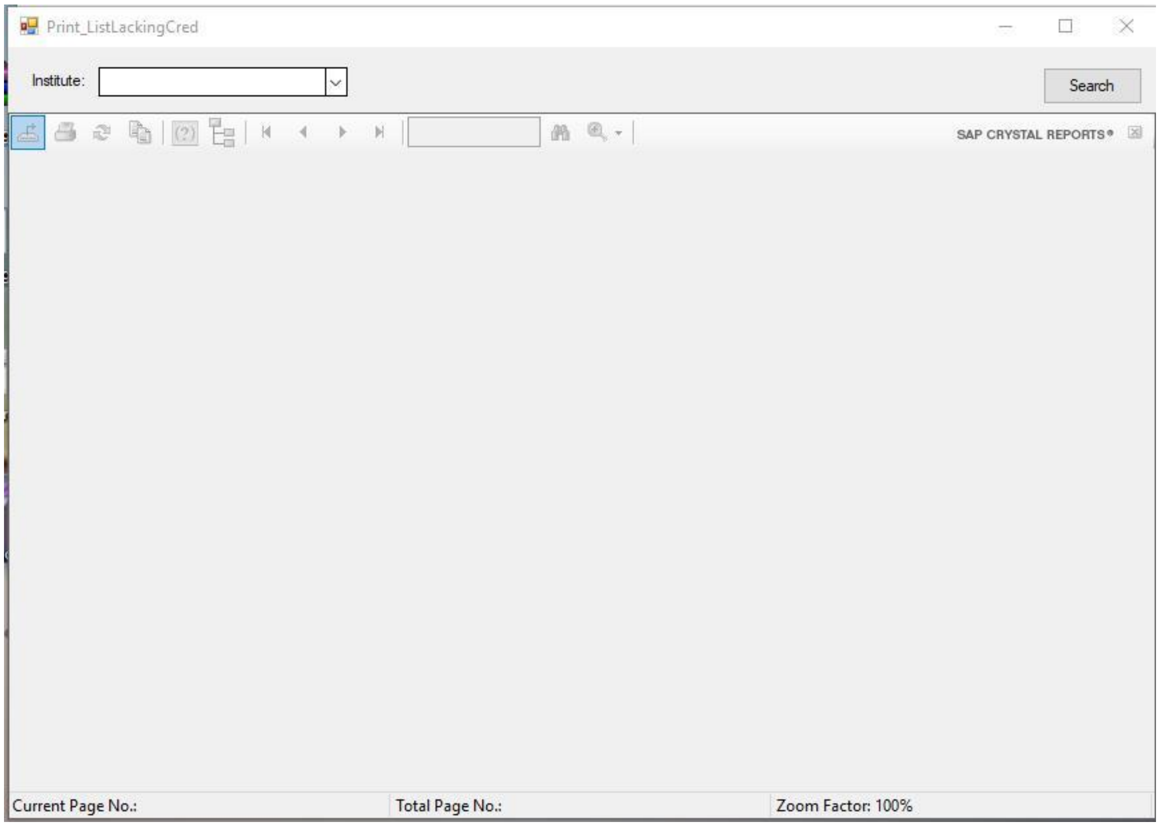
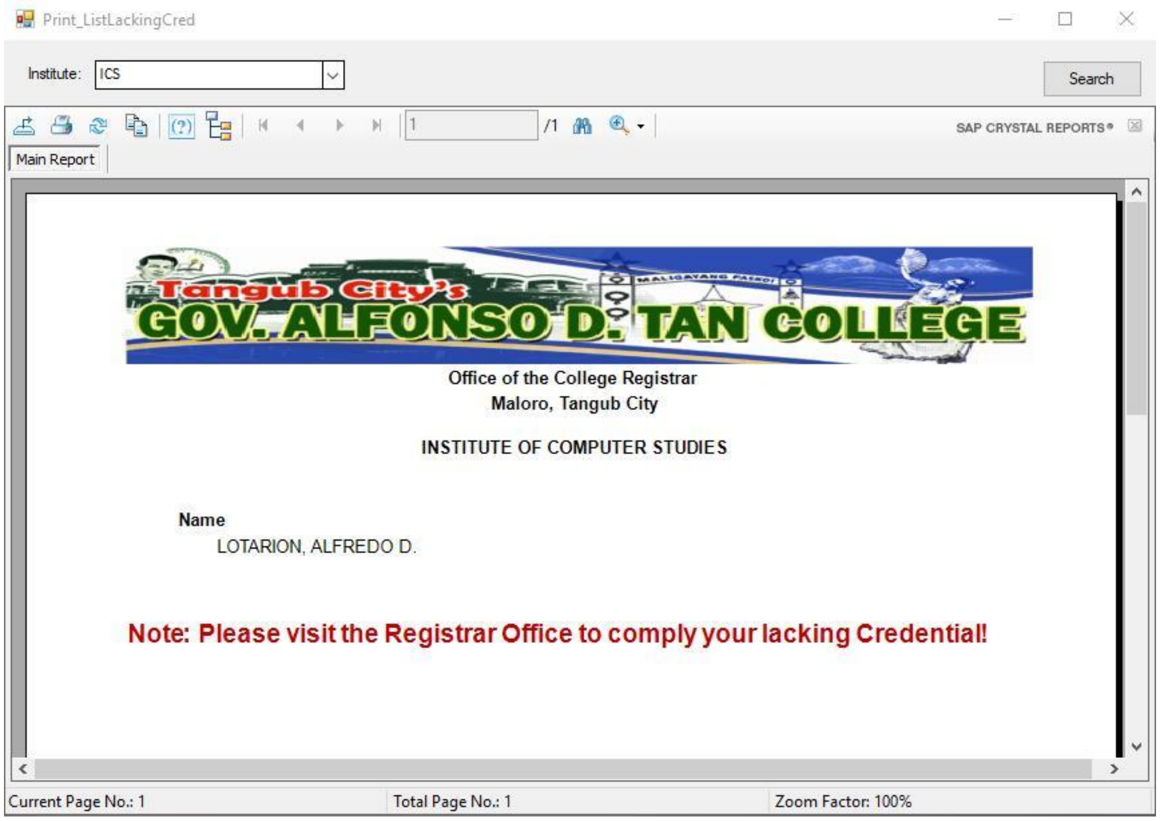


FIGURE 56. Print_ListLackingCred Form

c) Print List



Tangub City's GOV. ALFONSO D. TAN COLLEGE

Office of the College Registrar
Maloro, Tangub City

INSTITUTE OF COMPUTER STUDIES

Name
LOTARION, ALFREDO D.

Note: Please visit the Registrar Office to comply your lacking Credential!

Current Page No.: 1 | Total Page No.: 1 | Zoom Factor: 100%

FIGURE 57. Print List

9. Manage Request Letter for Form 137

    a) The user will search or select school name.

    b) Then Click Search to generate the Letter.

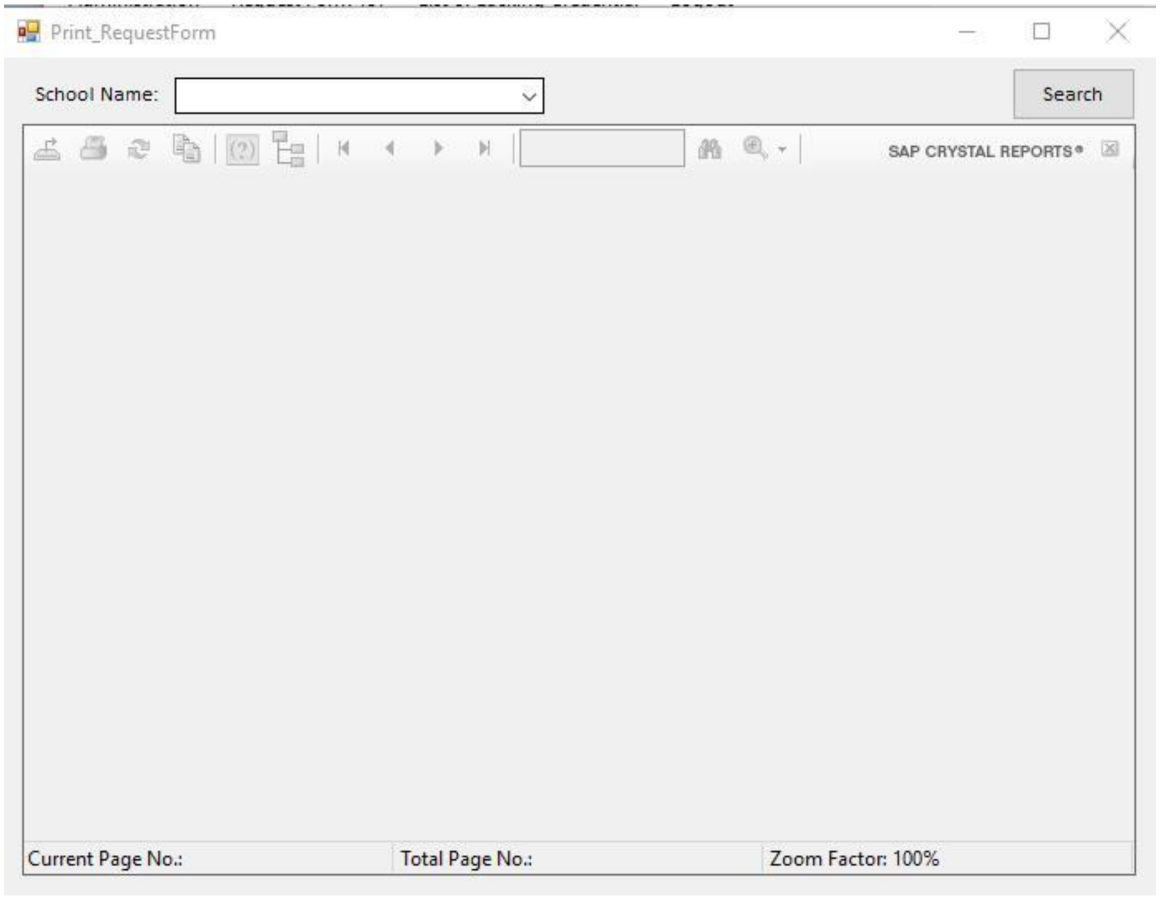
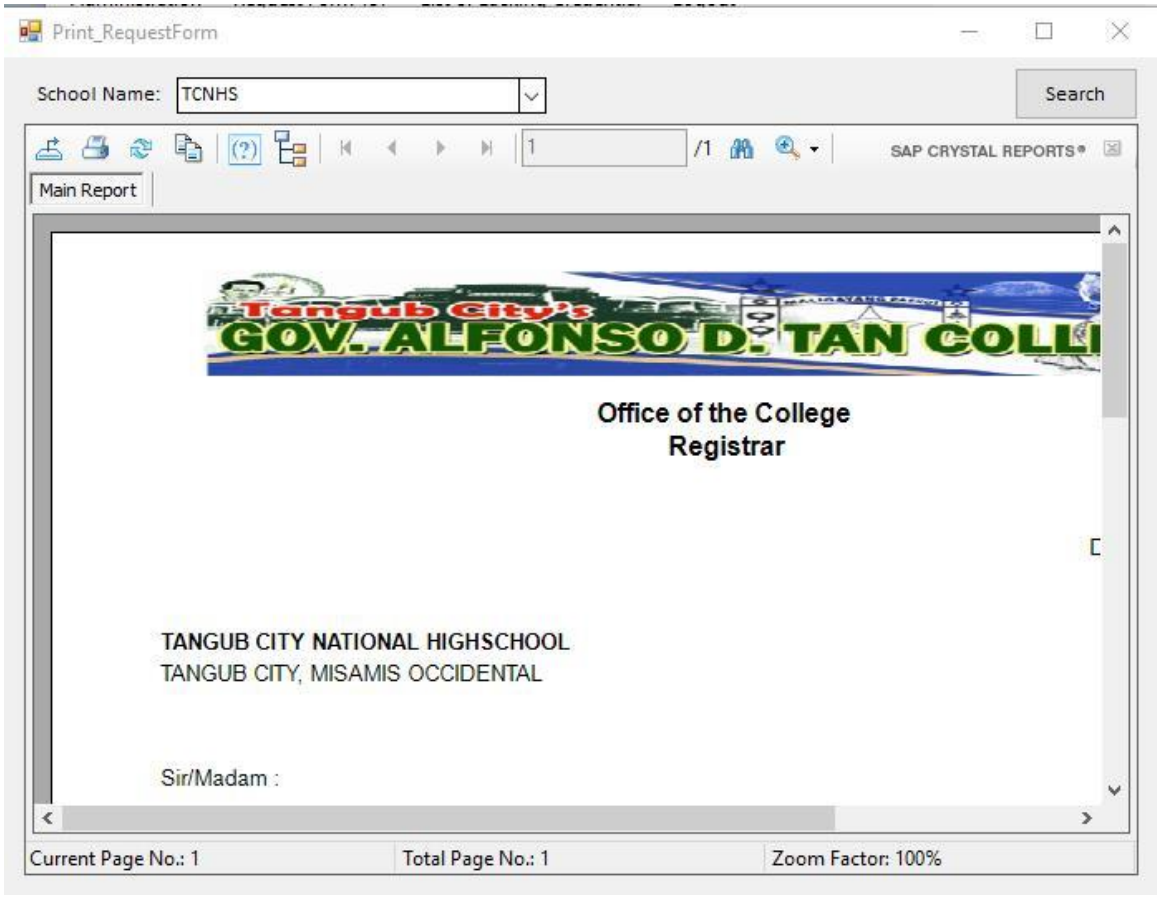


FIGURE 59. Print_RequestForm

c) Print Letter



Office of the College
Registrar

TANGUB CITY NATIONAL HIGHSCHOOL
TANGUB CITY, MISAMIS OCCIDENTAL

Sir/Madam :

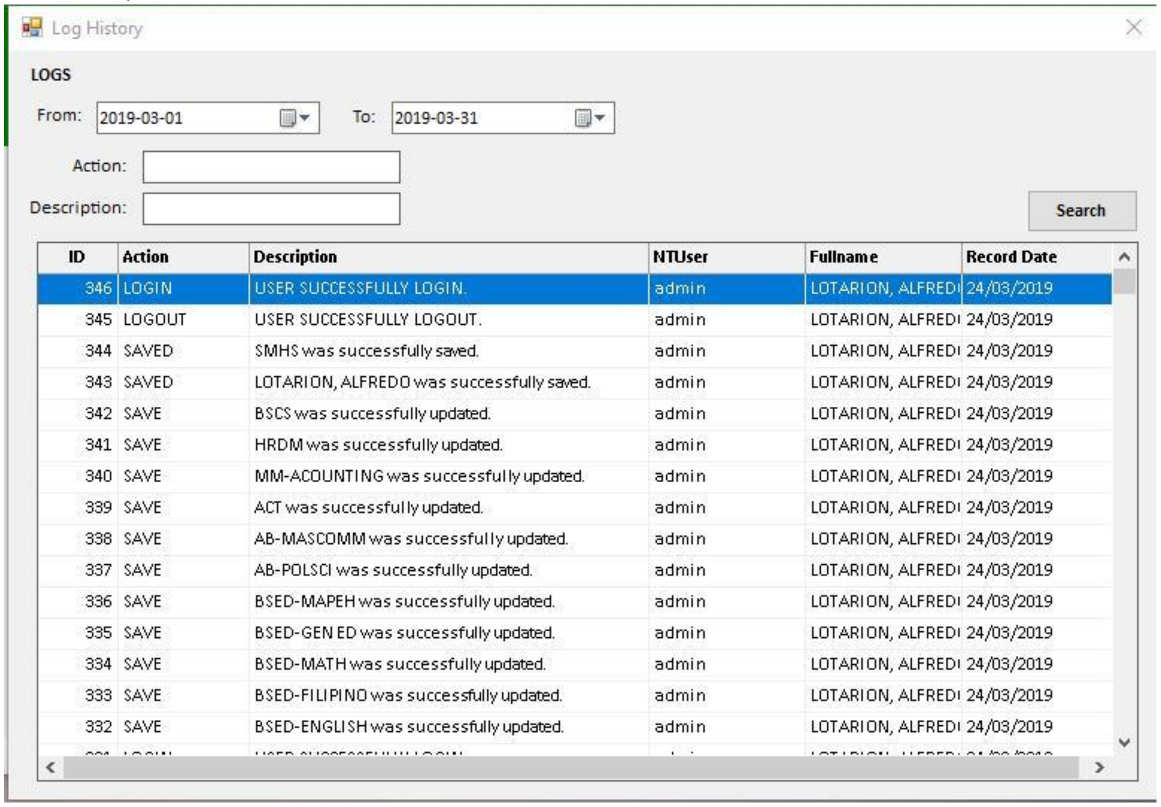
Current Page No.: 1 | Total Page No.: 1 | Zoom Factor: 100%

FIGURE 60. Print Letter

10. View Log Reports (Administrator Only)

a) The user will Input the action and description together with the date.

b) Then click Search.



From: 2019-03-01 To: 2019-03-31

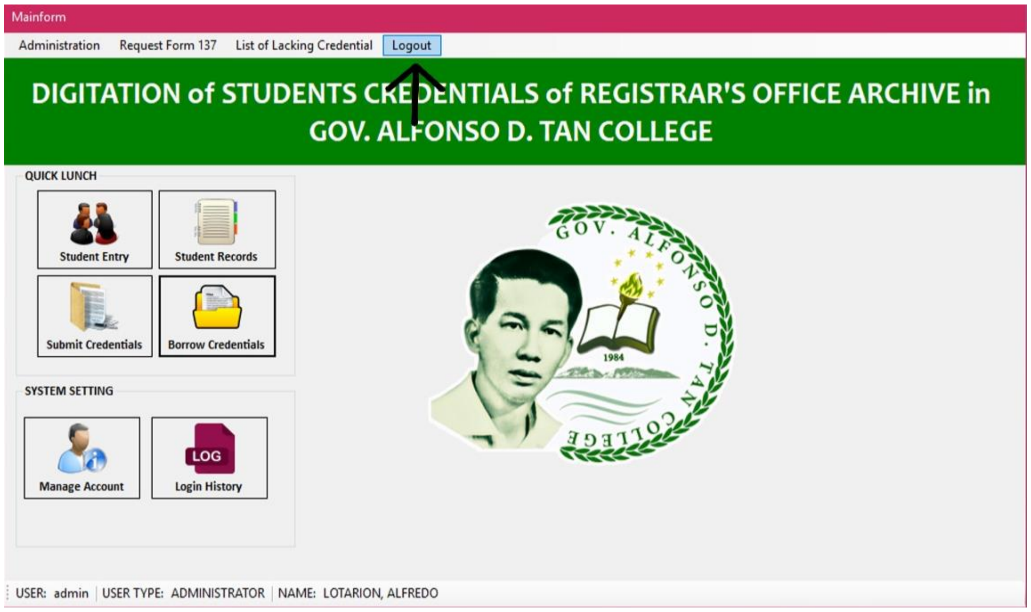
Action:

Description:

ID	Action	Description	NTUser	Fullname	Record Date
346	LOGIN	USER SUCCESSFULLY LOGIN.	admin	LOTARION, ALFRED	24/03/2019
345	LOGOUT	USER SUCCESSFULLY LOGOUT.	admin	LOTARION, ALFRED	24/03/2019
344	SAVED	SMHS was successfully saved.	admin	LOTARION, ALFRED	24/03/2019
343	SAVED	LOTARION, ALFREDO was successfully saved.	admin	LOTARION, ALFRED	24/03/2019
342	SAVE	BSCS was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
341	SAVE	HRDM was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
340	SAVE	MM-ACOUNTING was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
339	SAVE	ACT was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
338	SAVE	AB-MASCOMM was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
337	SAVE	AB-POLSCI was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
336	SAVE	BSED-MAPEH was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
335	SAVE	BSED-GEN ED was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
334	SAVE	BSED-MATH was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
333	SAVE	BSED-FILIPINO was successfully updated.	admin	LOTARION, ALFRED	24/03/2019
332	SAVE	BSED-ENGLISH was successfully updated.	admin	LOTARION, ALFRED	24/03/2019

FIGURE 61. View Log Reports (Administrator Only)

11. Log out



USER: admin | USER TYPE: ADMINISTRATOR | NAME: LOTARION, ALFREDO

FIGURE 61. Log out

Appendix B

GUI (Graphical User Interface)

Log in

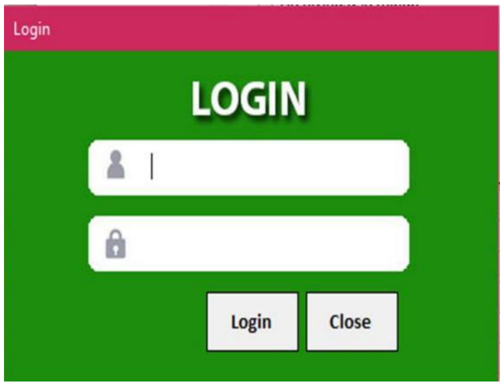
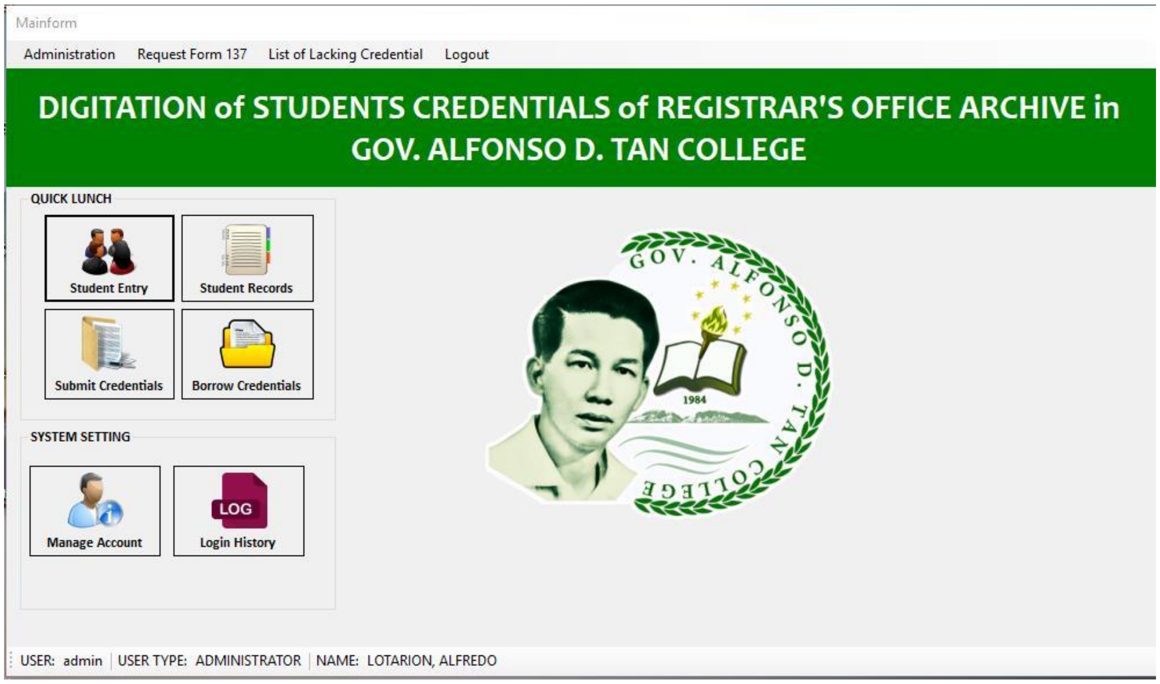


FIGURE 63. Log in (GUI)

Home screen (Admin)

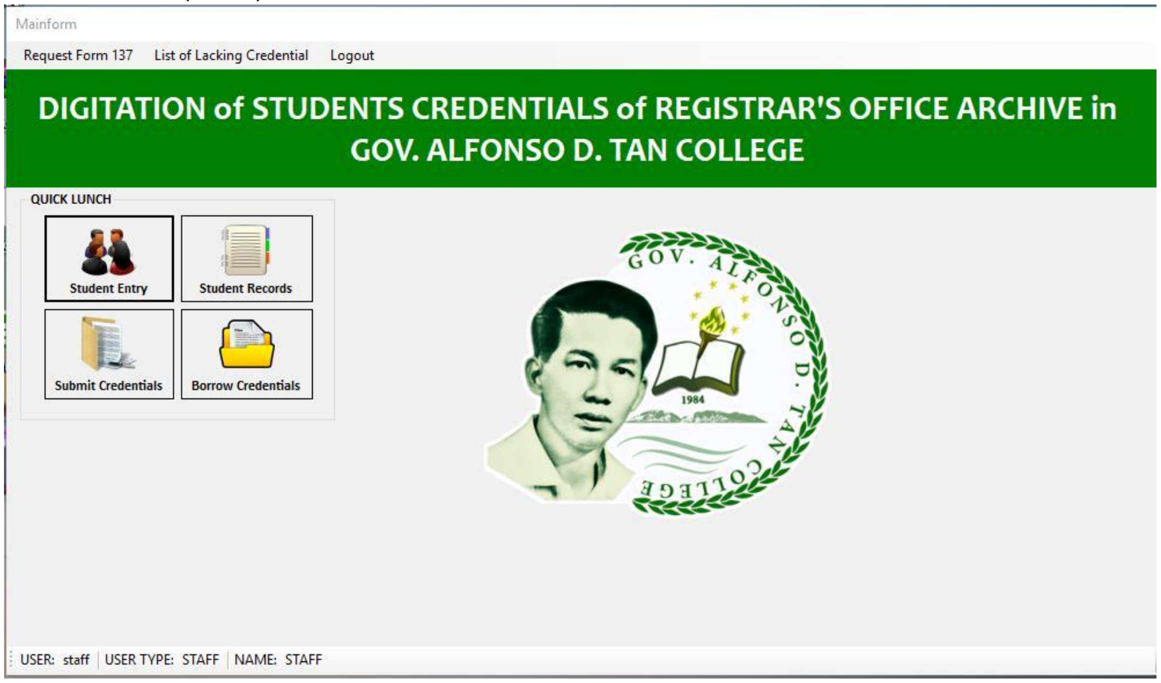


DIGITATION of STUDENTS CREDENTIALS of REGISTRAR'S OFFICE ARCHIVE in GOV. ALFONSO D. TAN COLLEGE

USER: admin | USER TYPE: ADMINISTRATOR | NAME: LOTARION, ALFREDO

FIGURE 64. Home Screen (Admin)

Home Screen (Staff)

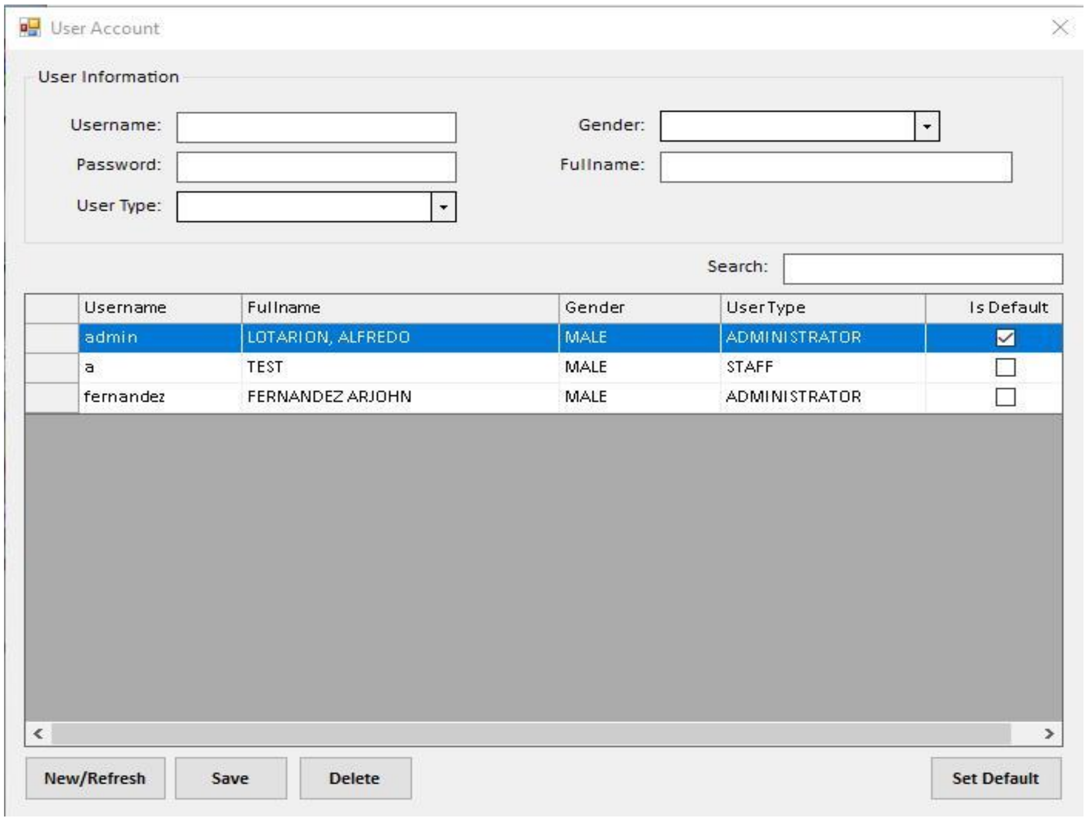

Mainform

DIGITATION of STUDENTS CREDENTIALS of REGISTRAR'S OFFICE ARCHIVE in GOV. ALFONSO D. TAN COLLEGE

USER: staff | USER TYPE: STAFF | NAME: STAFF

FIGURE 65. Home Screen (Staff)

Manage Account (Admin)


User Account

User Information

Username:

Password:

User Type:

Gender:

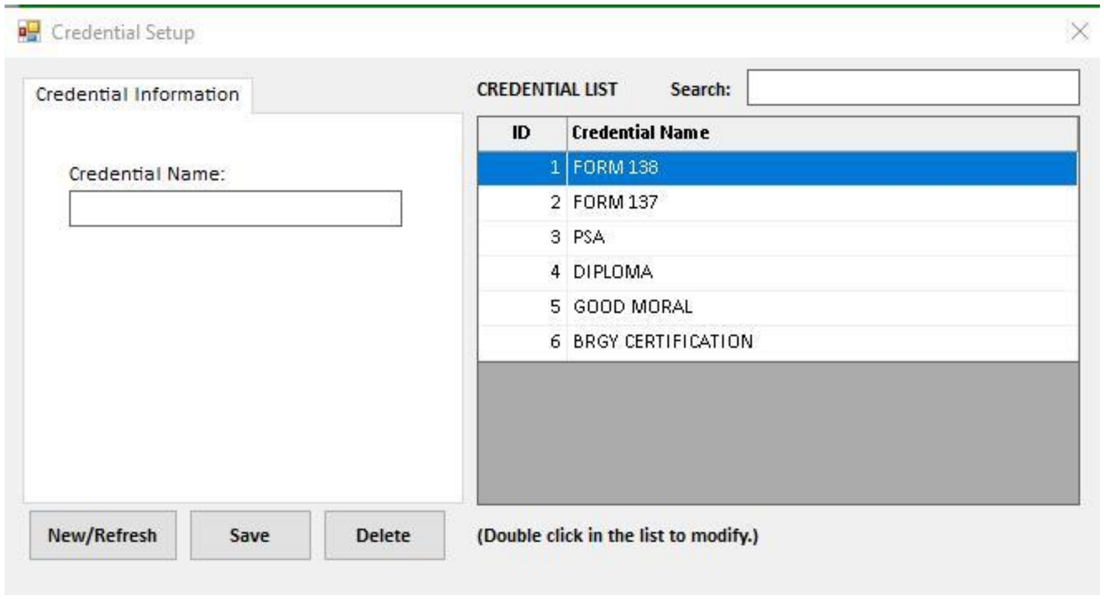
Fullname:

Search:

	Username	Fullname	Gender	UserType	Is Default
	admin	LOTARION, ALFREDO	MALE	ADMINISTRATOR	☑
	a	TEST	MALE	STAFF	☐
	fernandez	FERNANDEZ ARJOHN	MALE	ADMINISTRATOR	☐

FIGURE 66. Manage Account (Admin)

Credentials Setup (Admin)



Credential Name:

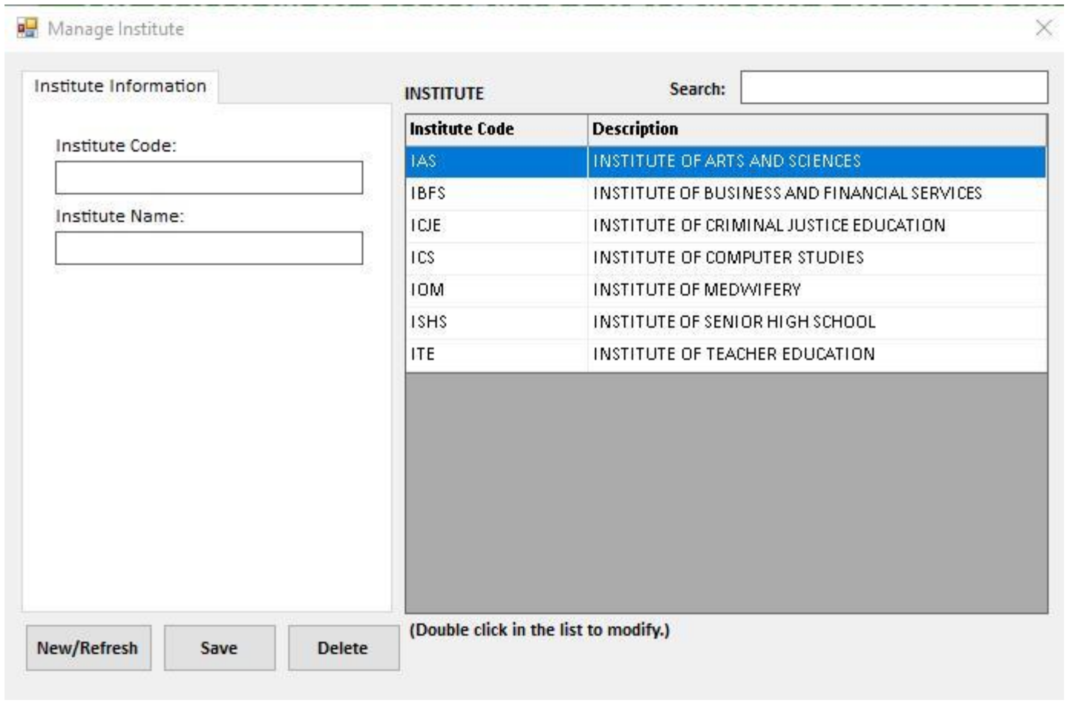
CREDENTIAL LIST Search:

ID	Credential Name
1	FORM 138
2	FORM 137
3	PSA
4	DIPLOMA
5	GOOD MORAL
6	BRGY CERTIFICATION

(Double click in the list to modify.)

FIGURE 67. Credential Setup (Admin)

Manage Institute (Admin)



Institute Code:

Institute Name:

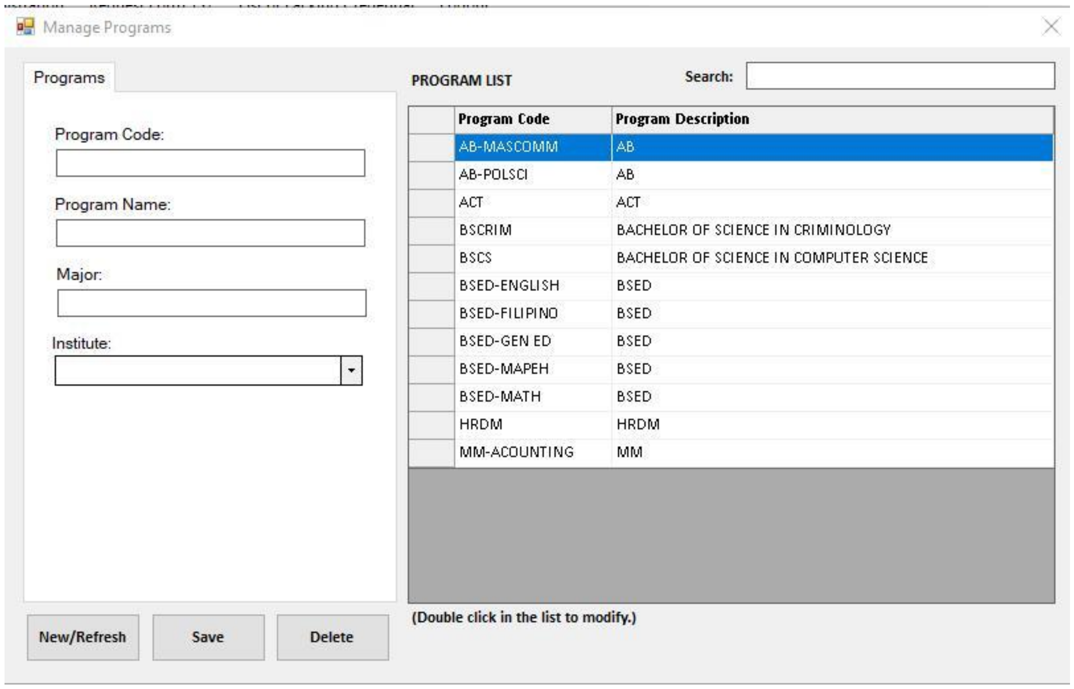
INSTITUTE Search:

Institute Code	Description
IAS	INSTITUTE OF ARTS AND SCIENCES
IBFS	INSTITUTE OF BUSINESS AND FINANCIAL SERVICES
ICJE	INSTITUTE OF CRIMINAL JUSTICE EDUCATION
ICS	INSTITUTE OF COMPUTER STUDIES
IOM	INSTITUTE OF MEDWIFERY
ISHS	INSTITUTE OF SENIOR HIGH SCHOOL
ITE	INSTITUTE OF TEACHER EDUCATION

(Double click in the list to modify.)

FIGURE 68. Manage Institute (Admin)

Manage Programs (Admin)



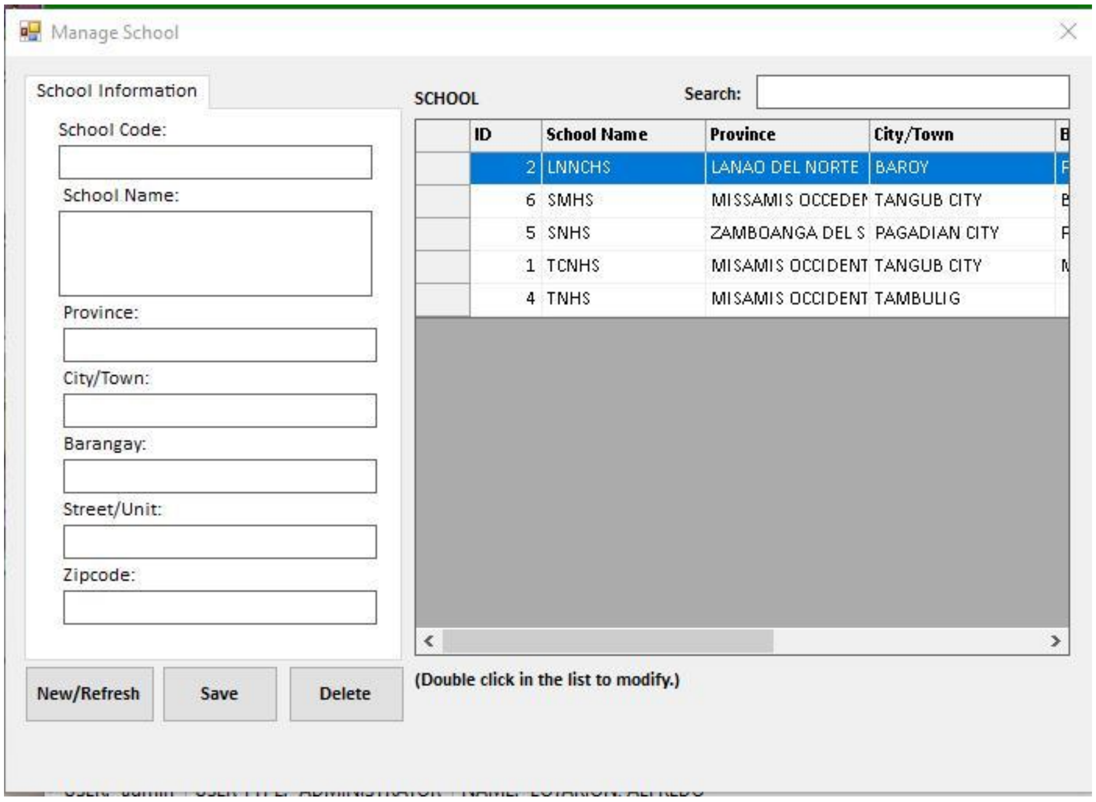
PROGRAM LIST Search:

	Program Code	Program Description
	AB-MASCOMM	AB
	AB-POLSCI	AB
	ACT	ACT
	BSCRIM	BACHELOR OF SCIENCE IN CRIMINOLOGY
	BSCS	BACHELOR OF SCIENCE IN COMPUTER SCIENCE
	BSED-ENGLISH	BSED
	BSED-FILIPINO	BSED
	BSED-GEN ED	BSED
	BSED-MAPEH	BSED
	BSED-MATH	BSED
	HRDM	HRDM
	MM-ACOUNTING	MM

(Double click in the list to modify.)

FIGURE 69. Manage Programs (Admin)

Manage High School Background (Admin)



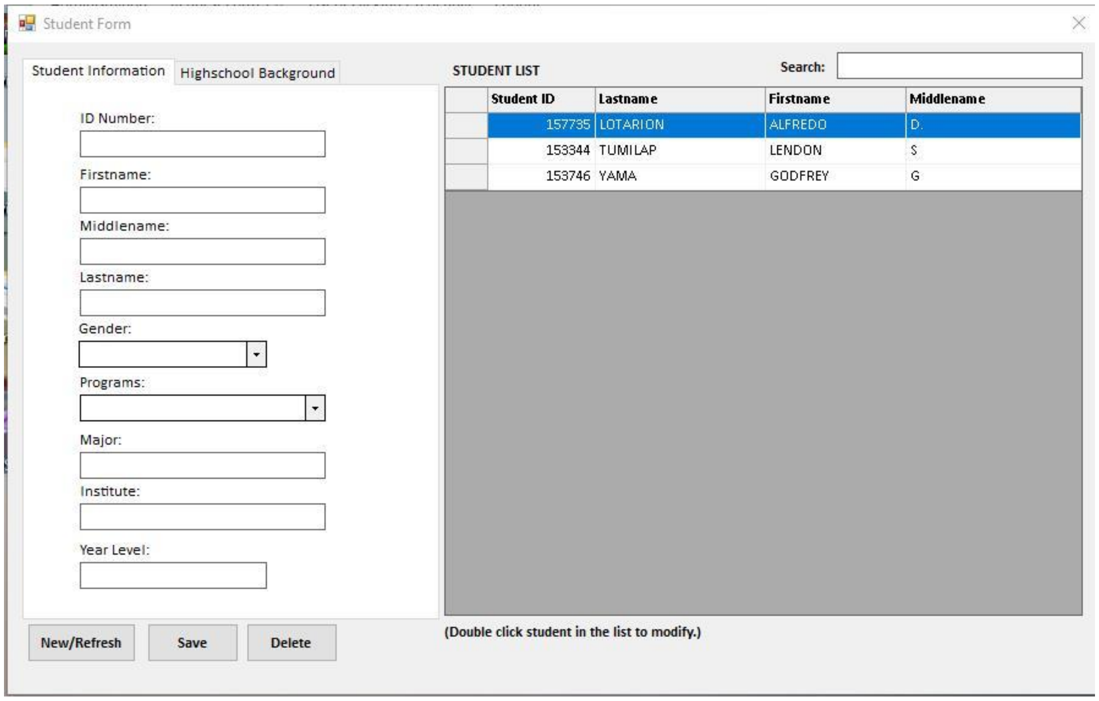
SCHOOL Search:

	ID	School Name	Province	City/Town	B
	2	LNNCHS	LANAO DEL NORTE	BAROY	F
	6	SMHS	MISSAMIS OCCEDEI	TANGUB CITY	B
	5	SNHS	ZAMBOANGA DEL S	PAGADIAN CITY	F
	1	TCNHS	MISAMIS OCCIDENT	TANGUB CITY	M
	4	TNHS	MISAMIS OCCIDENT	TAMBULIG	

(Double click in the list to modify.)

FIGURE 70. Manage High School Background (Admin)

Student Entry



Student Form

Student Information | Highschool Background

ID Number:

Firstname:

Middlename:

Lastname:

Gender:

Programs:

Major:

Institute:

Year Level:

STUDENT LIST

Search:

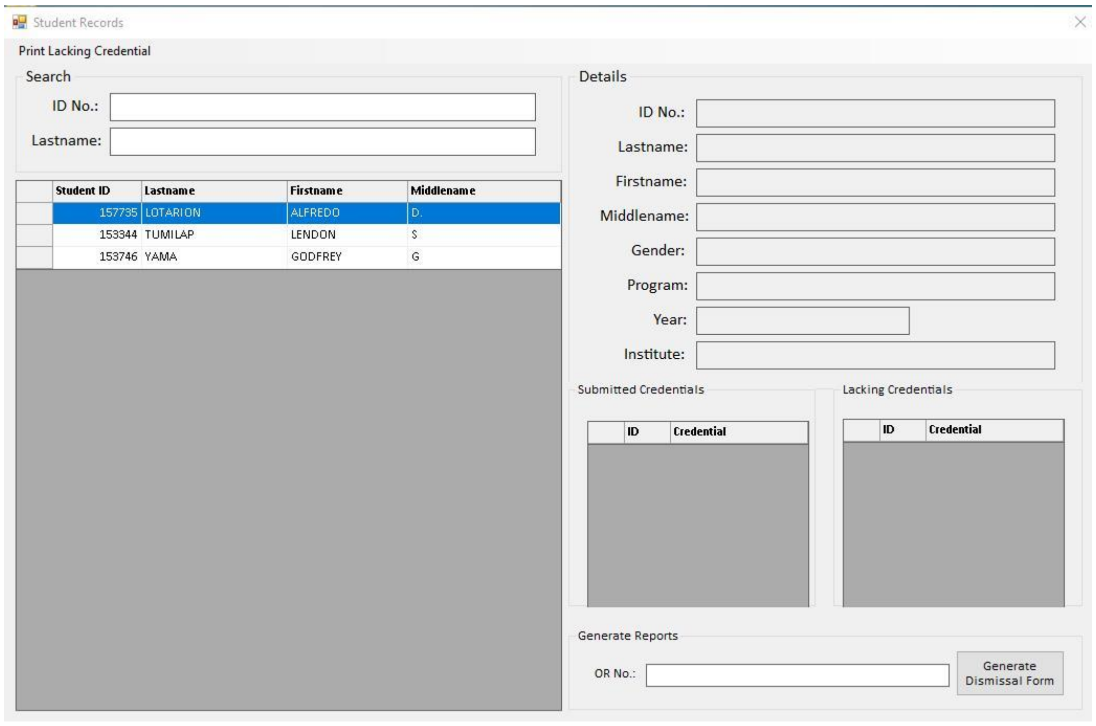
Student ID	Lastname	Firstname	Middlename
157735	LOTARION	ALFREDO	D.
153344	TUMILAP	LENDON	S
153746	YAMA	GODFREY	G

(Double click student in the list to modify.)

New/Refresh | Save | Delete

FIGURE 71. Student Entry

Student Records



Student Records

Print Lacking Credential

Search

ID No.:

Lastname:

Student ID	Lastname	Firstname	Middlename
157735	LOTARION	ALFREDO	D.
153344	TUMILAP	LENDON	S
153746	YAMA	GODFREY	G

Details

ID No.:

Lastname:

Firstname:

Middlename:

Gender:

Program:

Year:

Institute:

Submitted Credentials

ID	Credential

Lacking Credentials

ID	Credential

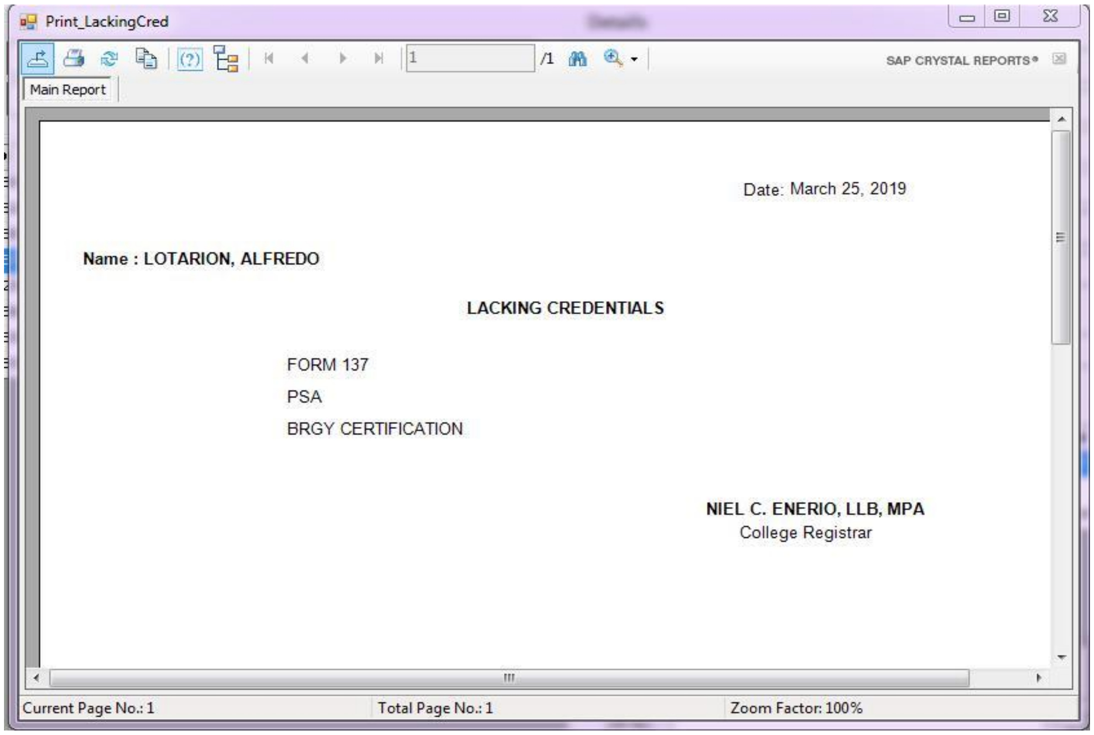
Generate Reports

OR No.:

Generate Dismissal Form

FIGURE 72. Student Records

Print Student Credential Monitoring Sheet



Date: March 25, 2019

Name : LOTARION, ALFREDO

LACKING CREDENTIALS

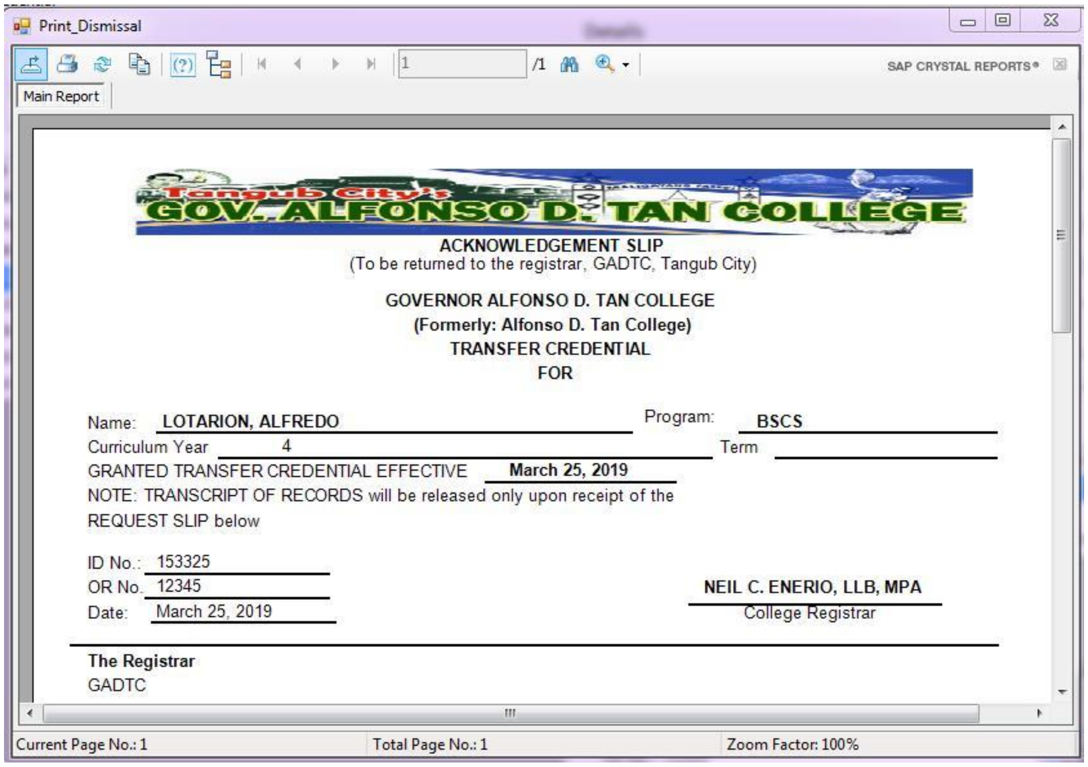
FORM 137
PSA
BRGY CERTIFICATION

NIEL C. ENERIO, LLB, MPA
College Registrar

Current Page No.: 1 Total Page No.: 1 Zoom Factor: 100%

FIGURE 73. Print Student Credential Monitoring Sheet

Print Honourable Dismissal



ACKNOWLEDGEMENT SLIP
(To be returned to the registrar, GADTC, Tangub City)

GOVERNOR ALFONSO D. TAN COLLEGE
(Formerly: Alfonso D. Tan College)
TRANSFER CREDENTIAL
FOR

Name: LOTARION, ALFREDO Program: BSCS
Curriculum Year 4 Term
GRANTED TRANSFER CREDENTIAL EFFECTIVE March 25, 2019
NOTE: TRANSCRIPT OF RECORDS will be released only upon receipt of the REQUEST SLIP below

ID No.: 153325
OR No. 12345
Date: March 25, 2019

NEIL C. ENERIO, LLB, MPA
College Registrar

The Registrar
GADTC

Current Page No.: 1 Total Page No.: 1 Zoom Factor: 100%

FIGURE 74. Manage Student Dismissal

Submission of Student Credentials

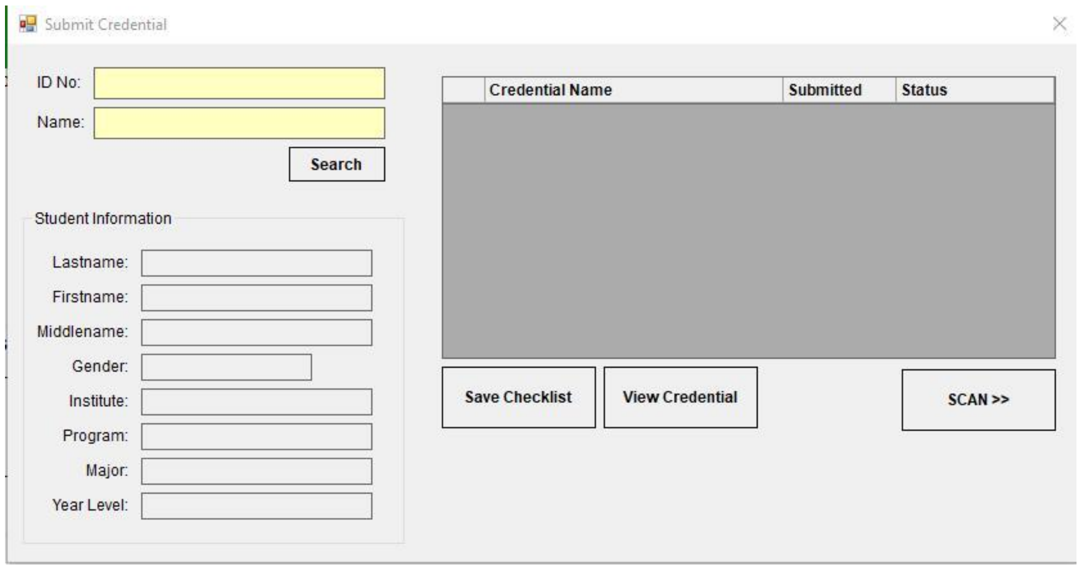


FIGURE 75. Submission of Student Credentials

Scan and Saving of Credentials

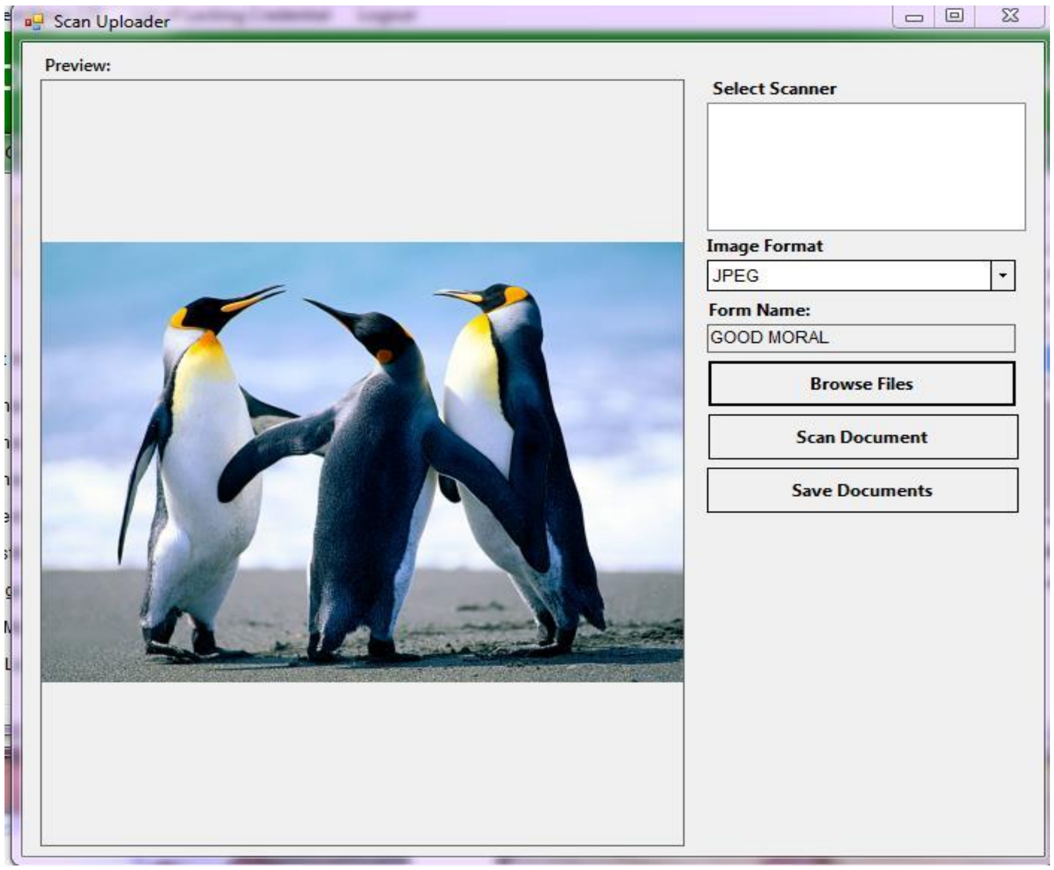
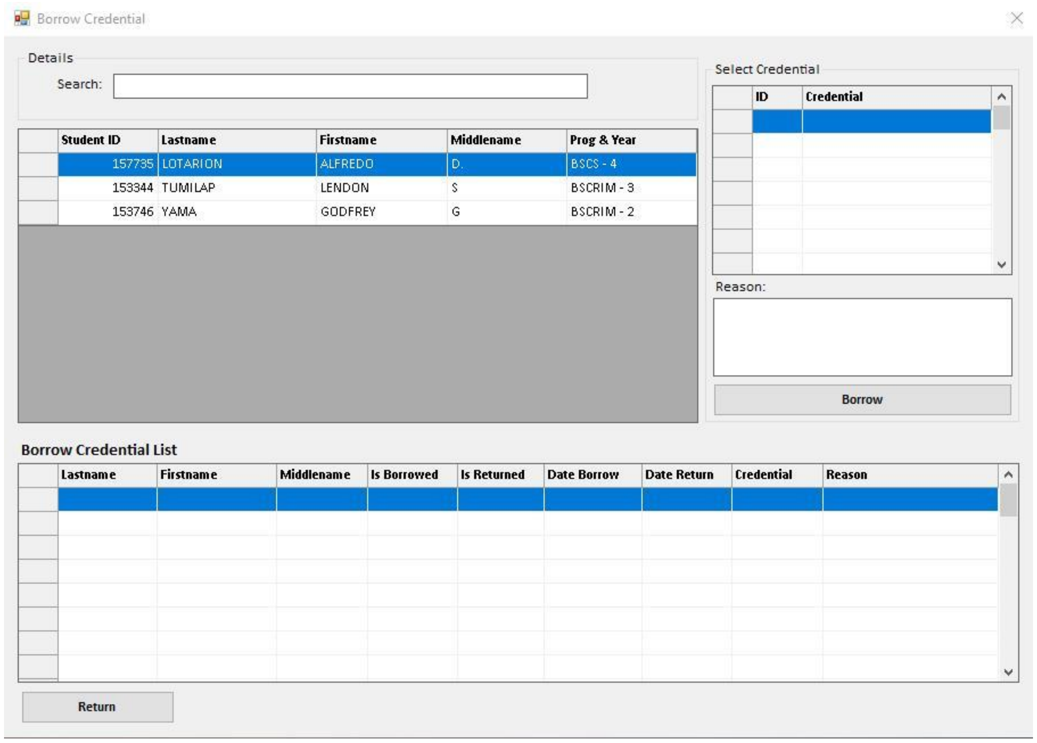


Image Format: JPEG

Form Name: GOOD MORAL

FIGURE 76. Scan and Saving of Student Credentials

Barrowing/Returning of Student Credentials



Borrow Credential

Details

Search:

Student ID	Lastname	Firstname	Middlename	Prog & Year
157735	LOTARION	ALFREDO	D.	BSCS - 4
153344	TUMILAP	LENDON	S	BSCRIM - 3
153746	YAMA	GODFREY	G	BSCRIM - 2

Select Credential

ID	Credential

Reason:

Borrow

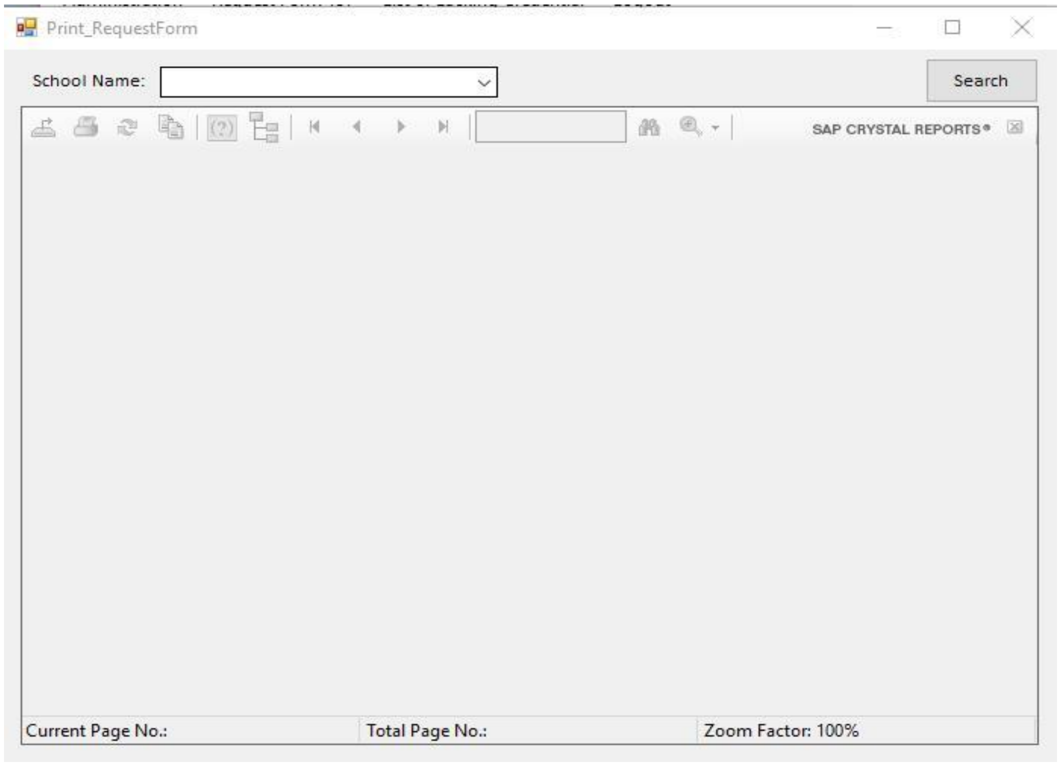
Borrow Credential List

Lastname	Firstname	Middlename	Is Borrowed	Is Returned	Date Borrow	Date Return	Credential	Reason

Return

FIGURE 77. Barrowing/Returning of Student Credentials

Manage Request letter for Form 137



Print_RequestForm

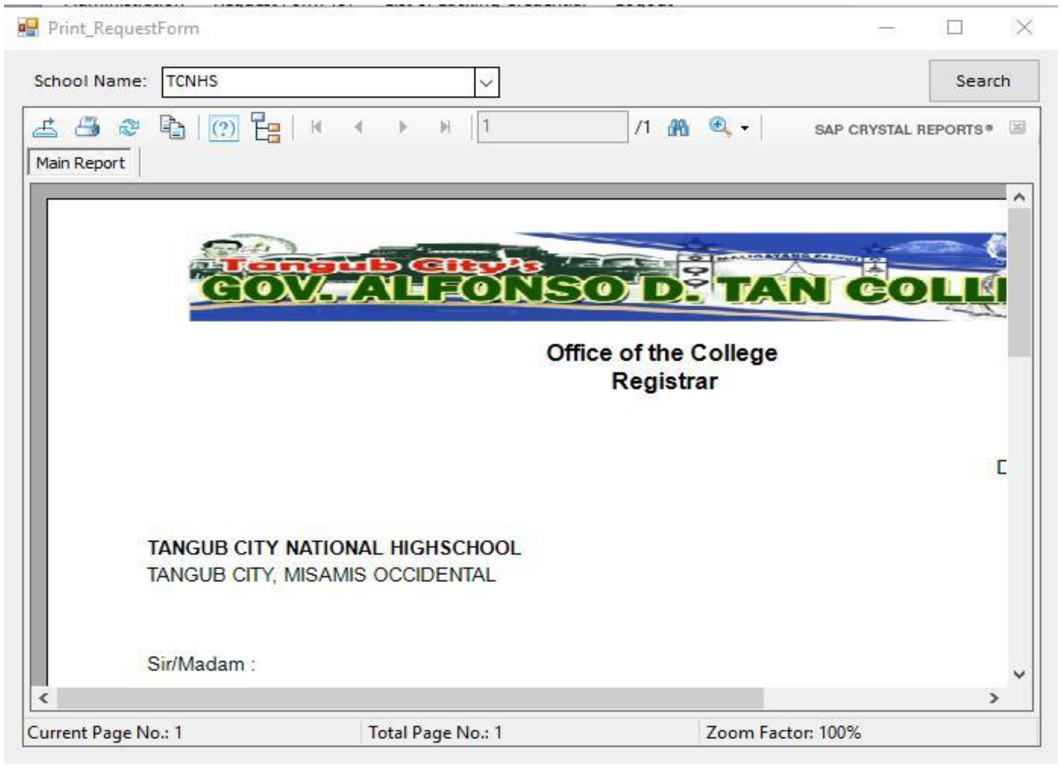
School Name:

Search

Current Page No.: Total Page No.: Zoom Factor: 100%

FIGURE 78. Manage Request Letter for Form 137

Print Request Letter



School Name: TCNHS

Tangub City's GOV. ALFONSO D. TAN COLL

Office of the College Registrar

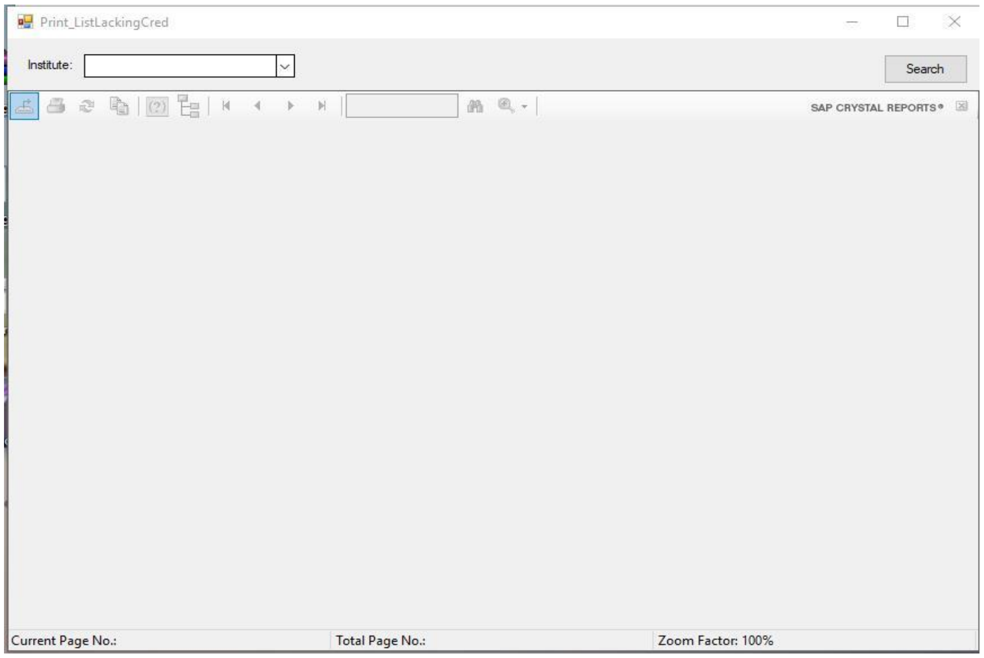
TANGUB CITY NATIONAL HIGHSCHOOL
TANGUB CITY, MISAMIS OCCIDENTAL

Sir/Madam :

Current Page No.: 1 Total Page No.: 1 Zoom Factor: 100%

FIGURE 79. Print Request Letter

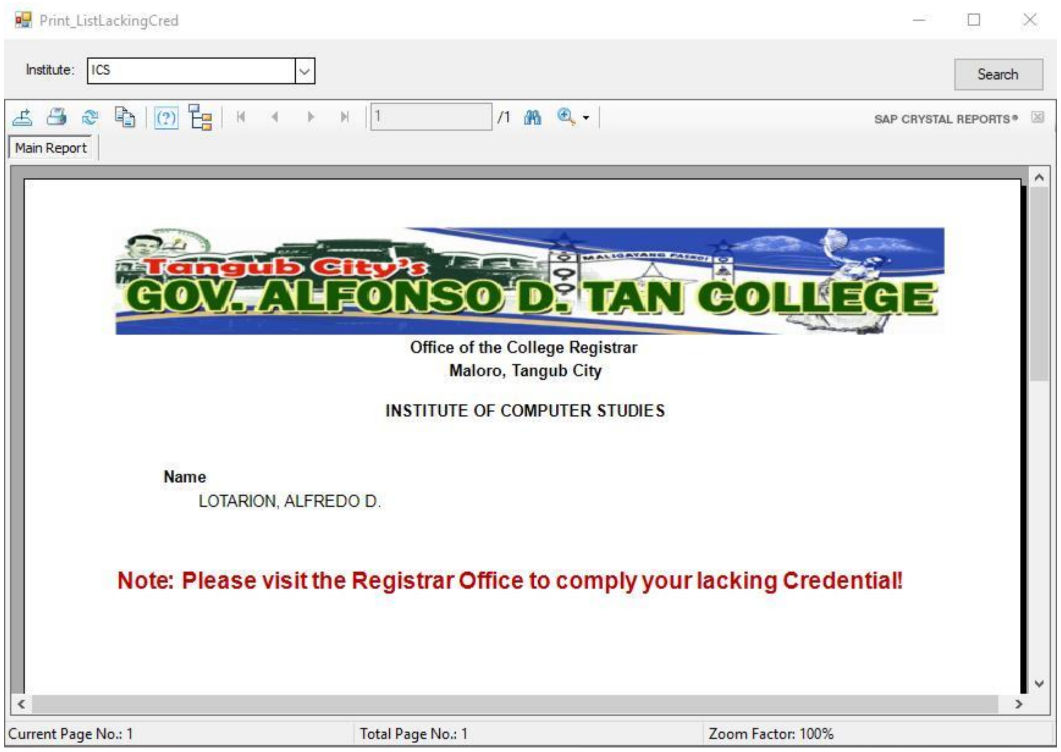
Manage List of Students with Lacking Credentials



Current Page No.: Total Page No.: Zoom Factor: 100%

FIGURE 80. Manage List of Students with Lacking Credentials

Print List



Office of the College Registrar
Maloro, Tangub City

INSTITUTE OF COMPUTER STUDIES

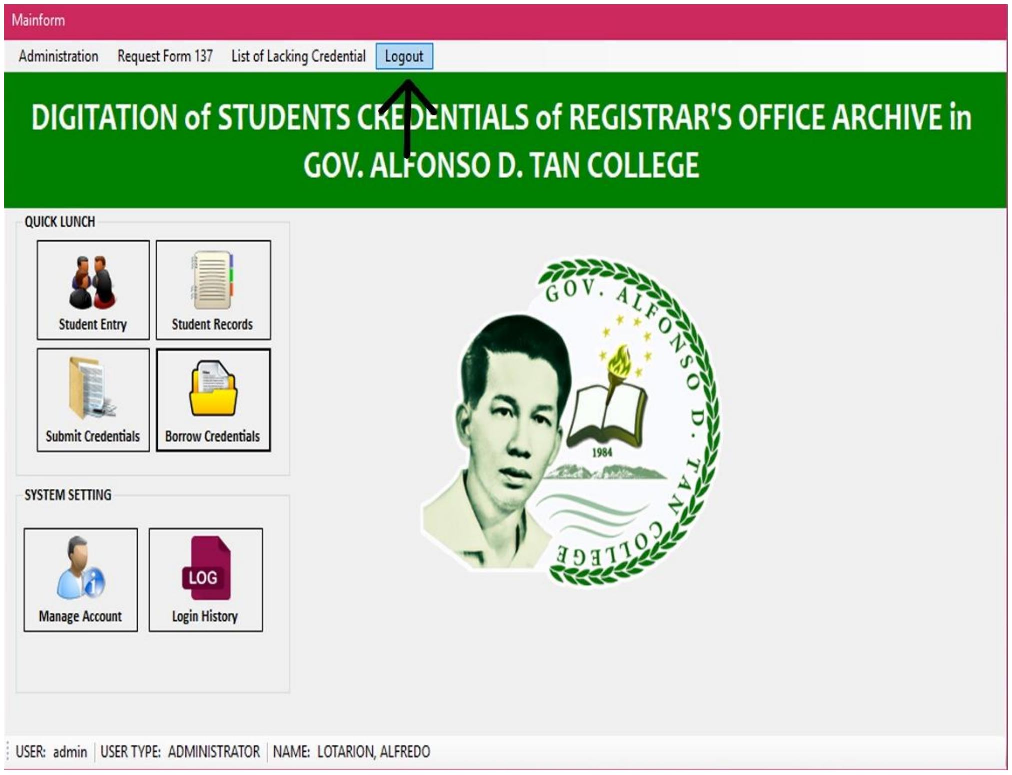
Name
LOTARION, ALFREDO D.

Note: Please visit the Registrar Office to comply your lacking Credential!

Current Page No.: 1 | Total Page No.: 1 | Zoom Factor: 100%

FIGURE 81. Print List

Log out



USER: admin | USER TYPE: ADMINISTRATOR | NAME: LOTARION, ALFREDO

FIGURE 82. Log out

Appendix C

Source Code

Borrow credentials

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
using System.Windows.Forms;

namespace ArchivingSystem
{
    public partial class BorrowCredential : Form
    {


        MySqlConnection con;
        MySqlCommand cmd;
        string query;



        public BorrowCredential()
        {
            InitializeComponent();
        }


        public void LoadData()
        {

            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from vw_studentinfo where studentIDNo like ?key
or stdLname like ?key or stdFname like ?key order by stdLname asc";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?key", txtSearchID.Text + "%");
                //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flxStdList.AutoGenerateColumns = false;
                this.flxStdList.DataSource = dt;
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }


        public void LoadBorrowList()
        {

            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from vw_borrow_cred where studentIDNo=?stdno";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?stdno",
Convert.ToInt32(flxStdList[flxStdList.RowSel,"studentIDNo"]));
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flxBorrow.AutoGenerateColumns = false;
                this.flxBorrow.DataSource = dt;

            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        void LoadCredential()
        {

            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from vw_credential where studentIDNo=?stdno and
isCheck = 1" +
                    "";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?stdno",
flxStdList[flxStdList.RowSel, "studentIDNo"]);
                //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flxCred.AutoGenerateColumns = false;
                this.flxCred.DataSource = dt;

            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }


        private void BorrowCredential_Load(object sender, EventArgs e)
        {
            Helper.setAutoComplete(txtSearchID, "select distinct(studentIDNo) from
student order by studentIDNo asc", "studentIDNo");

            LoadData();
        }

        private void txtSearchID_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }

        private void txtSearchLname_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }

        private void flxStdList_DoubleClick(object sender, EventArgs e)
        {
            LoadCredential();

            if (flxCred.Rows.Count > 1)
            {
                LoadBorrowList();
                //flxBorrow.Rows.Count = flxBorrow.Rows.Fixed;
            }

        }

        private void btnBorrow_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtReason.Text))
            {
                Box.warnBox("Please indicate reason.");
                return;
            }
            if(flxCred.Rows.Count > 1)
            {

                con = Connection.con();
                con.Open();

                query = "select * from borrow_cred where stdSysID=?sysID and 
cred_setupID=?credsetupID and isBorrowed = 1 and isReturned = 0";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?sysID", 
Convert.ToInt32(flxStdList[flxStdList.RowSel, "stdSysID"]));
                cmd.Parameters.AddWithValue("?credsetupID", 
Convert.ToInt32(flxCred[flxCred.RowSel, "cred_setupID"]));
                MySqlDataReader dr;
                dr = cmd.ExecuteReader();
                bool flag = dr.Read();
                dr.Close();
                cmd.Dispose();

                if (flag)
                {
                    Box.warnBox("Credential is not returned yet. Transaction 
cannot proceed.");
                    con.Close();
                    con.Dispose();
                    return;
                }

                query = "INSERT INTO borrow_cred SET stdSysID=?sysID, 
cred_setupID=?credsetupID, isBorrowed=1, isReturned=0, dateBorrow=curdate(), 
reason=?reas";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?sysID", 
Convert.ToInt32(flxStdList[flxStdList.RowSel, "stdSysID"]));
                cmd.Parameters.AddWithValue("?credsetupID", 
Convert.ToInt32(flxCred[flxCred.RowSel, "cred_setupID"]));
                cmd.Parameters.AddWithValue("?reas", txtReason.Text);
                cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Box.infoBox("Credential Borrowed.");
                LoadBorrowList();
            }
            else
            {
                Box.warnBox("No data detected.");
            }
        }

        private void flxCred_DoubleClick(object sender, EventArgs e)
        {

        }

        private void btnReturn_Click(object sender, EventArgs e)
        {
            if (flxCred.Rows.Count > 1)
            {
                con = Connection.con();
                con.Open();
                query = "UPDATE borrow_cred SET stdSysID=?sysID, 
cred_setupID=?credsetupID, isReturned=1, dateReturn=curdate() where 
borrowCredID=?bCredID";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?sysID", 
Convert.ToInt32(flxStdList[flxStdList.RowSel, "stdSysID"]));
                cmd.Parameters.AddWithValue("?credsetupID", 
Convert.ToInt32(flxCred[flxCred.RowSel, "cred_setupID"]));
                cmd.Parameters.AddWithValue("?bCredID", 
Convert.ToInt32(flxBorrow[flxBorrow.RowSel, "borrowCredID"]));
                cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Box.infoBox("Credential Returned.");
                LoadBorrowList();
            }
        }
    }
}
```

Box.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ArchivingSystem
{
    class Box
    {

        public static void infoBox(string msg)
        {
            MessageBox.Show(msg, "INFORMATION!", MessageBoxButtons.OK, 
MessageBoxIcon.Information);
        }
        public static void warnBox(string msg)
        {
            MessageBox.Show(msg, "WARNING!", MessageBoxButtons.OK, 
MessageBoxIcon.Warning);
        }
        public static void errBox(string msg)
        {
            MessageBox.Show(msg, "ERROR!", MessageBoxButtons.OK, 
MessageBoxIcon.Error);
        }
        public static Boolean questionBox(string msg, string title)
        {
            bool flag;
            DialogResult dg = MessageBox.Show(msg, title, MessageBoxButtons.YesNo,
MessageBoxIcon.Question);
            if (dg == DialogResult.Yes)
            {
                flag = true;
            }
            else
            {
                flag = false;
            }
            return flag;
        }

        public static Boolean warningQuestionBox(string msg, string title)
        {
            bool flag;
            DialogResult dg = MessageBox.Show(msg, title, MessageBoxButtons.YesNo,
MessageBoxIcon.Warning);
            if (dg == DialogResult.Yes)
            {
                flag = true;
            }
            else
            {
                flag = false;
            }
            return flag;
        }


        public static void successSave()
        {
            MessageBox.Show("Data successfully saved!", "SAVED!",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }

        public static void successUpdate()
        {
            MessageBox.Show("Data successfully update!", "SAVED!",
MessageBoxButtons.OK, MessageBoxIcon.Information);
        }

        public static void failedSave()
        {
            MessageBox.Show("An error occured. Please contact system
administrator.", "ERROR!", MessageBoxButtons.OK, MessageBoxIcon.Error);
        }



    }
}
```

Connections.cs

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;


namespace ArchivingSystem
{
    class Connection
    {

        public static MySqlConnection con()
        {
            string[] lines = System.IO.File.ReadAllLines(Application.StartupPath +
"/config.txt");
            return new MySqlConnection(lines[0] + ";" + lines[1] +
";user=root;password=''");

            //catch (Exception er)
            //{
            //    Box.infoBox("Connecting to database error. Please check your
connection to server or you may contact system administrator.");
            //    // throw;
            //    return null;
            //}

        }

        public static MySqlConnection conRestore()
        {
            string[] lines = System.IO.File.ReadAllLines(Application.StartupPath +
"/config.txt");
            return new MySqlConnection(lines[0] + ";" + lines[1] +
";user=root;password=''; charset=utf8;convertzerodatetime=true;");

            //catch (Exception er)
            //{
            //    Box.infoBox("Connecting to database error. Please check your
connection to server or you may contact system administrator.");
            //    // throw;
            //    return null;
            //}

        }


    }
}
```

Crendential.setup

```
       using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class CredentialSetup : Form
    {


        MySqlConnection con;
        MySqlCommand cmd;
        string query;

        int id;

        public CredentialSetup()
        {
            InitializeComponent();
        }

        public void LoadData()
        {

            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from credential_setup where cred_name like 
?key";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flx.AutoGenerateColumns = false;
                this.flx.DataSource = dt;
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }
        private void CredentialSetup_Load(object sender, EventArgs e)
        {

            Helper.setAutoComplete(txtsearch, "select distinct(cred_name) from
credential_setup order by cred_setupID asc", "cred_name");

            LoadData();
        }

        private void flx_DoubleClick(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (flx[flx.RowSel, "cred_setupID"] != null)
                {
                    con = Connection.con();
                    con.Open();
                    query = "select * from credential_setup where
cred_setupID=?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?id",
Convert.ToInt32(flx[flx.RowSel, "cred_setupID"]));
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    while (dr.Read())
                    {
                        id = Convert.ToInt32(dr["cred_setupID"]);
                        txtCred.Text = Convert.ToString(dr["cred_name"]);
                       // txtfname.Text = Convert.ToString(dr["stdFname"]);

                    }
                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();
                }

            }
        }

        private void btnRefresh_Click(object sender, EventArgs e)
        {
            LoadData();
            clear();
        }

        void clear()
        {
            id = 0;
            txtCred.Text = "";
            txtsearch.Text = "";
            txtCred.Focus();
        }

        private void txtsearch_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (Box.questionBox("Are you sure you want to delete this row?",
"DELETE?"))
                {
                    int id = Convert.ToInt32(flx[flx.RowSel, 1]);
                    Helper.deleteByInt("credential_setup", "cred_setupID", id);
                    Box.infoBox("Deleted successfully.");
                    LoadData();
                }
            }
            else
            {
                Box.warnBox("Please select credential in the list.");
            }
        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtCred.Text))
            {
                Box.warnBox("Please input credential.");
                return;
            }

            if (String.IsNullOrEmpty(txtCred.Text))
            {
                Box.warnBox("Please input institute description.");
                return;
            }

            if (Helper.isExistString("credential_setup","cred_name",
txtCred.Text))
            {
                Box.warnBox("Credential name already in the database.");
                return;
            }

            processSave();
            LoadData();
            clear();
        }

        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();
                query = @"UPDATE credential_setup SET cred_name=?cname
                        WHERE cred_setupID = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?cname", txtCred.Text);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Credential successfully updated.");
                    Helper.RecordLogs("UPDATE", txtCred.Text + " was updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }
            }
            else
            {
                con = Connection.con();
                con.Open();
                query = @"INSERT INTO credential_setup SET cred_name=?cname";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?cname", txtCred.Text);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Credential successfully saved.");
                    Helper.RecordLogs("SAVE", txtCred.Text + " was saved.");
                    // Helper.RecordLogs("SAVED", txtlname.Text + ", " +
txtfname.Text + " was successfully saved.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }

            }
        }


    }
}
```

Helper.cs

```
using System;

using System.Collections.Generic;

using System.Linq;
using System.Text;

using System.Threading.Tasks;

using System.Windows.Forms;

using C1.C1Excel;

using C1.Win.C1FlexGrid;

using MySql.Data.MySqlClient;



namespace ArchivingSystem

{

    class Helper

    {







        //public static IEnumerable<Control> GetControlHierarchy(Control root)

        //{

        //    var queue = new Queue<Control>();


        //    queue.Enqueue(root);


        //    do

        //    {

        //        var control = queue.Dequeue();


        //        yield return control;
        //      foreach (var child in control.Controls.OfType<Control>())
        //          queue.Enqueue(child);

        //   } while (queue.Count > 0);

        //}



        //static List<string> getAccessControlsList(string className)
        //{
        //   MySqlConnection con;
        //   MySqlCommand cmd;
        //   string query;

        //   var retList = new List<string>();

        //   con = Connection.con();
        //   con.Open();
        //   query = "select * from vw_access where positionID=?posid and
className=?classname";
        //   cmd = new MySqlCommand(query, con);
        //   cmd.Parameters.AddWithValue("?posid", Properties.Settings.Default.positionID);
        //   cmd.Parameters.AddWithValue("?classname", className);
        //   MySqlDataReader dr;
        //   dr = cmd.ExecuteReader();
        //   while (dr.Read())
        //   {
        //       retList.Add(Convert.ToString(dr["controlName"]));
        //    }
        //    dr.Close();
        //    con.Close();
        //    con.Dispose();

        //    return retList;
        //}



        //public static void LoadCmb(string query, ComboBox cmb)
        //{
        //    MySqlConnection con;
        //    MySqlCommand cmd;
        //    // string query;
        //    cmb.Items.Clear();
        //    con = Connection.con();
        //    con.Open();
        //    cmd = new MySqlCommand(query, con);
        //    MySqlDataReader dr;
        //    dr = cmd.ExecuteReader();
        //    while (dr.Read())
        //    {
        //        cmb.Items.Add(Convert.ToString(dr[0]));
        //    }
        //    dr.Close();
        //    cmd.Dispose();
        //    con.Close();
        //    con.Dispose();
        //}
```

```
public static void LoadCmbPerCon(MySqlConnection con, string query, ComboBox cmb)
{
   //MySqlConnection con;
   MySqlCommand cmd;
   // string query;
   cmb.Items.Clear();
   con = Connection.con();
   con.Open();
   cmd = new MySqlCommand(query, con);
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      cmb.Items.Add(Convert.ToString(dr[0]));
   }
   dr.Close();
   cmd.Dispose();
   // con.Close();
   // con.Dispose();
}



public static bool deleteByInt(string tblename, string colname, int id)
{
   //bool flag = false;
   MySqlConnection con;
   MySqlCommand cmd;
```

```
            // string query;


            con = Connection.con();

            con.Open();

            cmd = new MySqlCommand("DELETE FROM " + tblename + " WHERE " + colname + " =
?id", con);

            cmd.Parameters.AddWithValue("?id", id);

            int i = cmd.ExecuteNonQuery();

            cmd.Dispose();

            con.Close();

            con.Dispose();

            if (i > 0)

            {

                return true;

            }

            else

            {

                return false;

            }


        }


        public static bool deleteByString(string tblename, string colname, string strID)

        {

            //bool flag = false;

            MySqlConnection con;

            MySqlCommand cmd;

            // string query;
            con = Connection.con();

            con.Open();

            cmd = new MySqlCommand("DELETE FROM " + tblename + " WHERE " + colname + " =
?id", con);

            cmd.Parameters.AddWithValue("?id", strID);

            int i = cmd.ExecuteNonQuery();

            cmd.Dispose();

            con.Close();

            con.Dispose();

            if (i > 0)

            {

                return true;

            }

            else

            {

                return false;

            }


        }


        public static bool isExistString(string tblname, string colname, string key)

        {

            MySqlConnection con;

            MySqlCommand cmd;

            con = Connection.con();

            con.Open();

            cmd = new MySqlCommand("SELECT * FROM " + tblname + " WHERE " + colname + " =
?key", con);

            cmd.Parameters.AddWithValue("?key", key);

            MySqlDataReader dr;
            dr = cmd.ExecuteReader();
            bool flag = dr.Read();
            dr.Close();
            cmd.Dispose();
            con.Close();
            return flag;
        }

        public static bool isExistInt(string tblname, string colname, int key)
        {
            MySqlConnection con;
            MySqlCommand cmd;
            con = Connection.con();
            con.Open();
            cmd = new MySqlCommand("SELECT * FROM " + tblname + " WHERE " + colname + " =
?key", con);
            cmd.Parameters.AddWithValue("?key", key);
            MySqlDataReader dr;
            dr = cmd.ExecuteReader();
            bool flag = dr.Read();
            dr.Close();
            cmd.Dispose();
            con.Close();
            return flag;
        }

        public static void txtInteger(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar))
            {
                e.Handled = true;
            }
        }

        public static void txtDecimal(object sender, KeyPressEventArgs e)
        {
            if (!char.IsControl(e.KeyChar) && !char.IsDigit(e.KeyChar) && (e.KeyChar != '.'))
            {
                e.Handled = true;
            }

            // If you want, you can allow decimal (float) numbers
            if ((e.KeyChar == '.') && ((sender as TextBox).Text.IndexOf('.') > -1))
            {
                e.Handled = true;
            }
        }

        public static int returnID(string tblname, string colname, string key, string colid)
        {
            int flag = 0;
            MySqlConnection con;
            MySqlCommand cmd;
            con = Connection.con();
            con.Open();
            cmd = new MySqlCommand("SELECT * FROM " + tblname + " WHERE " + colname + " =
?key", con);
```

```
            cmd.Parameters.AddWithValue("?key", key);

            MySqlDataReader dr;

            dr = cmd.ExecuteReader();

            while (dr.Read())

            {

                flag = Convert.ToInt32(dr[colid]);

            }


            dr.Close();

            cmd.Dispose();

            con.Close();

            return flag;

        }


        public static int returnIDperConnection(MySqlConnection con, string tblname, string
colname, string key, string colid)

        {

            int flag = 0;

            //MySqlConnection con;

            MySqlCommand cmd;

            //con = Connection.con();

            //con.Open();

            cmd = new MySqlCommand("SELECT * FROM " + tblname + " WHERE " + colname + " =
?key", con);

            cmd.Parameters.AddWithValue("?key", key);

            MySqlDataReader dr;

            dr = cmd.ExecuteReader();

            while (dr.Read())

            {

                flag = Convert.ToInt32(dr[colid]);
```

```
    }

    dr.Close();
    cmd.Dispose();
    // con.Close();
    return flag;
  }

  public static void setAutoComplete(TextBox textbox, string query, string colname)
  {
    textbox.AutoCompleteMode = AutoCompleteMode.SuggestAppend;
    textbox.AutoCompleteSource = AutoCompleteSource.CustomSource;
    AutoCompleteStringCollection coll = new AutoCompleteStringCollection();
    MySqlConnection conn = null;
    MySqlCommand cmd;
    MySqlDataReader dr;
    conn = Connection.con();
    conn.Open();
    //string query = "SELECT " + colname + " FROM " + tablename + " GROUP BY " + groupby +
"";
    cmd = new MySqlCommand(query, conn);
    dr = cmd.ExecuteReader();
    while (dr.Read())
    {
      coll.Add(dr[colname].ToString().Trim());
    }
    conn.Close();
    dr.Close();
    textbox.AutoCompleteCustomSource = coll;
}

public static int returnLastInsertID(MySqlConnection con)
{
    MySqlCommand cmd = new MySqlCommand("select last_insert_id()", con);
    int id = Convert.ToInt32(cmd.ExecuteScalar());
    return id;
}



public static void FirstDayOfMonth_AddMethod(DateTime value, DateTimePicker dt)
{
    dt.Value = value.Date.AddDays(1 - value.Day);
}
public static void LastDayOfMonth_AddMethod(DateTime value, DateTimePicker dt)
{
    dt.Value = (value.Date.AddDays(1 - value.Day)).AddMonths(1).AddDays(-1);
}

public static void NumRow(C1FlexGrid grid)
{
    for (int rowno = 1; rowno <= grid.Rows.Count - 1; rowno++)
    {
        grid[rowno, 0] = rowno.ToString();
    }
}
```

```
public static void LoadSheet(C1FlexGrid _flx, XLSheet xsheet, int noCol, string _2ndcol)
{
  int row;
  int col;

  XLCell cell;

  for (row = 1; row < xsheet.Rows.Count; row++)
  {
    _flx.Rows.Add();
    //cell = xsheet.GetCell(row, 1);
    //_flx[row, 1] = Convert.ToString(cell.Value).ToUpper();

    //cell = xsheet.GetCell(row, 2);
    //_flx[row, 1] = _2ndcol.ToUpper();

    for (col = 0; col < noCol; col++)
    {
      if (row <= xsheet.Rows.Count - 1)
      {

        cell = xsheet.GetCell(row, col);
        //_flx[row, col + 1] = cell.Value.ToString().ToUpper();
        if (!String.IsNullOrEmpty(cell.Value.ToString()))
        {
          _flx[row, col + 1] = Convert.ToString(cell.Value).ToUpper();
          _flx[row, col + 2] = _2ndcol;
        }
```

```
            }
        }
        Application.DoEvents();
    }

    Helper.NumRow(_flx);
    //Application.DoEvents();
}



public static void RecordLogs(string action, string des)
{
    MySqlConnection con = null;
    MySqlCommand cmd;
    string query;
    con = Connection.con();
    con.Open();
    query = "INSERT INTO loghistory SET userid=?userid, logAction=?action, logDesc=?desc";
    cmd = new MySqlCommand(query, con);
    //cmd.Parameters.AddWithValue("?id", id);
    cmd.Parameters.AddWithValue("?userid", Properties.Settings.Default.userid);
    cmd.Parameters.AddWithValue("?action", action);
    cmd.Parameters.AddWithValue("?desc", des);
    cmd.ExecuteNonQuery();
    cmd.Dispose();
    con.Close();
    con.Dispose();

}
```

```
public static int getIDFlexGrid(C1FlexGrid flx)
{
    return Convert.ToInt32(flx[flx.RowSel, 1]);
}



public static void getCurrentAY(ComboBox cmb)
{
    MySqlConnection con = null;
    MySqlCommand cmd;
    string query;
    con = Connection.con();
    con.Open();
    query = "SELECT * FROM ay WHERE active = 1";
    cmd = new MySqlCommand(query, con);
    //cmd.Parameters.AddWithValue("?id", id);
    MySqlDataReader dr;
    dr = cmd.ExecuteReader();
    while (dr.Read())
    {
        cmb.Text = Convert.ToString(dr["ay"]);
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
    con.Dispose();
```

```
}

public static void getCurrentSemester(ComboBox cmb)
{
   MySqlConnection con = null;
   MySqlCommand cmd;
   string query;
   con = Connection.con();
   con.Open();
   query = "SELECT * FROM semester WHERE active = 1";
   cmd = new MySqlCommand(query, con);
   //cmd.Parameters.AddWithValue("?id", id);
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      cmb.Text = Convert.ToString(dr["semester"]);
   }
   dr.Close();
   cmd.Dispose();
   con.Close();
   con.Dispose();

}


public static string getStringCurrentAY(MySqlConnection con)
{
   //  MySqlConnection con = null;
   MySqlCommand cmd;
   string query;
   string ay = "";
   //con = Connection.con();
   // con.Open();
   query = "SELECT * FROM ay WHERE active = 1";
   cmd = new MySqlCommand(query, con);
   //cmd.Parameters.AddWithValue("?id", id);
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      ay = Convert.ToString(dr["ay"]);
   }
   dr.Close();
   cmd.Dispose();
   // con.Close();
   // con.Dispose();
   return ay;
}

public static void cmbLoadAY(ComboBox cmb)
{
   MySqlConnection con = null;
   MySqlCommand cmd;
   string query;
   con = Connection.con();
   con.Open();
   query = "SELECT DISTINCT(ay) FROM ay";
   cmd = new MySqlCommand(query, con);
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      cmb.Items.Add(Convert.ToString(dr["ay"]));
   }
   dr.Close();
   cmd.Dispose();
   con.Close();
   con.Dispose();
}




public static void cmbLoadGroup(ComboBox cmb)
{
   MySqlConnection con = null;
   MySqlCommand cmd;
   string query;
   con = Connection.con();
   con.Open();
   query = "SELECT DISTINCT(groupCode) FROM ngroup order by groupCode asc";
   cmd = new MySqlCommand(query, con);
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      cmb.Items.Add(Convert.ToString(dr["groupCode"]));
   }
   dr.Close();
   cmd.Dispose();
   con.Close();
   con.Dispose();
}

public static void cmbLoadProgram(ComboBox cmb)
{
   MySqlConnection con = null;
   MySqlCommand cmd;
   string query;
   con = Connection.con();
   con.Open();
   query = "SELECT DISTINCT(programCode) FROM programs order by programCode asc";
   cmd = new MySqlCommand(query, con);
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      cmb.Items.Add(Convert.ToString(dr["programCode"]));
   }
   dr.Close();
   cmd.Dispose();
   con.Close();
   con.Dispose();
}
```

```
    public static void cmbLoadInstitute(ComboBox cmb)
    {
        MySqlConnection con = null;
        MySqlCommand cmd;
        string query;
        con = Connection.con();
        con.Open();
        query = "SELECT DISTINCT(instituteCode) as instituteCode FROM institute order by
instituteCode asc";
        cmd = new MySqlCommand(query, con);
        MySqlDataReader dr;
        dr = cmd.ExecuteReader();
        while (dr.Read())
        {
            cmb.Items.Add(Convert.ToString(dr["instituteCode"]));
        }
        dr.Close();
        cmd.Dispose();
        con.Close();
        con.Dispose();
    }

    public static void cmbLoadProvince(ComboBox cmb)
    {
        MySqlConnection con = null;
        MySqlCommand cmd;
        string query;
        con = Connection.con();
```

```
            con.Open();

            query = "SELECT DISTINCT(province) as province FROM province order by province asc";

            cmd = new MySqlCommand(query, con);

            MySqlDataReader dr;

            dr = cmd.ExecuteReader();

            while (dr.Read())

            {

                cmb.Items.Add(Convert.ToString(dr["province"]));

            }

            dr.Close();

            cmd.Dispose();

            con.Close();

            con.Dispose();

        }



    }

}
```

<u>LogHistory</u>

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class LogHistory : Form
    {
        MySqlConnection con;
        MySqlCommand cmd;
        string query;


        public LogHistory()
        {
            InitializeComponent();
        }

        void loadData()
        {
            con = Connection.con();
            con.Open();
            query = "select * from (select * from vw_logs where logDate between
?from and ?to) as aa where logAction like ?laction and logDesc like ?desc order by
logDate desc";
            cmd = new MySqlCommand(query, con);
            cmd.Parameters.AddWithValue("?laction", txtAction.Text + "%");
            cmd.Parameters.AddWithValue("?desc", "%" + txtDesc.Text + "%");
            cmd.Parameters.AddWithValue("?from", dtFrom.Value);
            cmd.Parameters.AddWithValue("?to", dtTo.Value);
            DataTable dt = new DataTable();
            MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
            adptr.Fill(dt);
            cmd.Dispose();
            con.Close();
            con.Dispose();

            flx.AutoGenerateColumns = false;
            flx.DataSource = dt;
        }

        private void LogHistory_Load(object sender, EventArgs e)
        {
            Helper.FirstDayOfMonth_AddMethod(DateTime.Now, dtFrom);
            Helper.LastDayOfMonth_AddMethod(DateTime.Now, dtTo);
            loadData();

        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            loadData();
        }
    }
}
```

Login.cs

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class Login : Form
    {

        MySqlConnection con;
        MySqlCommand cmd;
        string query;


        public Login()
        {
            InitializeComponent();
        }

        private void btnClose_Click(object sender, EventArgs e)
        {
            //this.Close();
            Application.Exit();
        }


        bool islogin()
        {
            bool flag = false;

            try
            {
                con = Connection.con();
                con.Open();
                //query = "SELECT username, pwd FROM user_acc WHERE username
=?uname and aes_decrypt(password,'passw0rd') = ?pwd";
                query = "SELECT * FROM users WHERE username =?uname and pwd =
?pwd";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?uname", this.txtuser.Text);
                cmd.Parameters.AddWithValue("?pwd", this.txtpwd.Text);
                MySqlDataReader dr;
                dr = cmd.ExecuteReader();
                while (dr.Read())
                {
                    Properties.Settings.Default.userid =
Convert.ToInt32(dr["usersid"]);
                    Properties.Settings.Default.username =
Convert.ToString(dr["username"]);
                    Properties.Settings.Default.fullname =
Convert.ToString(dr["fullname"]);
                    Properties.Settings.Default.position =
Convert.ToString(dr["position"]);
                    Properties.Settings.Default.Save();
                    flag = true;
                }
                dr.Close();
                cmd.Dispose();
                con.Close();
            }
            catch (Exception er)
            {
                Box.errBox(er.Message + "\nPlease contact system administrator.");

            }

            return flag;
        }

        private void btnLogin_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtuser.Text))
            {
                Box.warnBox("Please input username.");
                return;
            }
            if (String.IsNullOrEmpty(txtpwd.Text))
            {
                Box.warnBox("Please input password.");
                return;
            }

            if (islogin())
            {
                Helper.RecordLogs("LOGIN", "USER SUCCESSFULLY LOGIN.");
                Mainform frm = new Mainform(this);
               txtuser.Text  ="";
                txtpwd.Text = "";
                frm.Show();
                this.Hide();
            }
            else
            {
                Box.errBox("Username and Password error.");
                txtuser.Text = "";
                txtpwd.Text = "";
                txtuser.Focus();
            }
        }
    }
}
```

MainForm

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Diagnostics;
using System.Drawing;
using System.IO;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class Mainform : Form
    {
        Login _login;
        MySqlConnection con;
     //   MySqlCommand cmd;
        string query;


        public Mainform(Login _login)
        {
            InitializeComponent();
            this._login = _login;
        }

        private void button2_Click(object sender, EventArgs e)
        {
            SubmitCredential frm = new SubmitCredential();
            frm.ShowDialog();
        }

        private void btnStudent_Click(object sender, EventArgs e)
        {
            StudentForm frm = new StudentForm();
            frm.ShowDialog();
        }

        private void button5_Click(object sender, EventArgs e)
        {
            UserAccount frm = new UserAccount();
            frm.ShowDialog();
        }

        private void logoutToolStripMenuItem_Click(object sender, EventArgs e)
        {
            if(Box.questionBox("Are you sure you want to logout?", "LOGOUT"))
            {
                Helper.RecordLogs("LOGOUT", "USER SUCCESSFULLY LOGOUT.");
                _login.Show();
                this.Hide();
            }
        }

        private void Mainform_Load(object sender, EventArgs e)
        {

            if(Properties.Settings.Default.position.ToLower() != "administrator")
            {
                groupBox2.Visible = false;
                settingToolStripMenuItem.Visible = false;
            }
            lblUser.Text = Properties.Settings.Default.username;
            lblUserType.Text = Properties.Settings.Default.position;
            lblName.Text = Properties.Settings.Default.fullname;
        }

        private void button1_Click(object sender, EventArgs e)
        {
            StudentRecords frm = new StudentRecords();
            frm.ShowDialog();
        }

        private void borrowCredentialToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            BorrowCredential frm = new BorrowCredential();
            frm.ShowDialog();
        }

        private void manageInstituteToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            ManageInstitute frm = new ManageInstitute();
            frm.ShowDialog();
        }

        private void credentialSetupToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            CredentialSetup frm = new CredentialSetup();
            frm.ShowDialog();
        }

        private void button4_Click(object sender, EventArgs e)
        {
            LogHistory frm = new LogHistory();
            frm.ShowDialog();
        }

        private void button3_Click(object sender, EventArgs e)
        {
            BorrowCredential frm = new BorrowCredential();
            frm.ShowDialog();
        }

        private void manageSchoolToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            ManageSchool frm = new ManageSchool();
            frm.ShowDialog();
        }

        private void reportToolStripMenuItem_Click(object sender, EventArgs e)
        {
            Print_RequestForm frm = new Print_RequestForm();
            frm.ShowDialog();
        }
        private void listOfLackingCredentialToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            Print_ListLackingCred frm = new Print_ListLackingCred();
            frm.ShowDialog();
        }

        private void manageAccountToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            button5_Click(sender, e);
        }

        private void backupToolStripMenuItem_Click(object sender, EventArgs e)
        {
            //Box.infoBox(Application.StartupPath);
            //openDir();
           // openDir();
            backupCopyFiles();
          //

        }

      //  string filename = "c:\\";
        void backupCopyFiles()
        {

            DialogResult result = saveFileDialog1.ShowDialog();

            if (result == DialogResult.OK)
            {
                string filename = saveFileDialog1.FileName;

                Process proc = new Process();
                proc.StartInfo.UseShellExecute = true;
                proc.StartInfo.FileName =
Path.Combine(Environment.SystemDirectory, "xcopy.exe");
                string source = Application.StartupPath;
                proc.StartInfo.Arguments = source + " "+ filename +" /E /I";
                proc.Start();
                //  Box.infoBox(filename.ToString());
                Box.infoBox("Backup successfully.");
            }


        }

        void openDir()
        {
            using (MySqlCommand cmd = new MySqlCommand())
            {
                using (MySqlBackup mb = new MySqlBackup(cmd))
                {
                    con = Connection.conRestore();
                    cmd.Connection = con;
                    con.Open();
                    mb.ExportToFile(Application.StartupPath);
                    con.Close();
                    // Box.infoBox("Backup successfully.");
                }
            }

        }

        private void restoreToolStripMenuItem_Click(object sender, EventArgs e)
        {

            try
            {
                DialogResult result = openFileDialog1.ShowDialog();

                if (result == DialogResult.OK)
                {
                    string filename = openFileDialog1.FileName;
                    using (MySqlCommand cmd = new MySqlCommand())
                    {

                        // Box.infoBox(filename.ToString());
                        using (MySqlBackup mb = new MySqlBackup(cmd))
                        {
                            con = Connection.conRestore();
                            cmd.Connection = con;
                            con.Open();
                            mb.ImportFromFile(filename + "/");
                            con.Close();

                            Box.infoBox("Restore successfully.");
                        }
                    }
                }
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
               // throw;
            }

        }

        private void manageProgramToolStripMenuItem_Click(object sender, EventArgs
e)
        {
            ManagePrograms frm = new ManagePrograms();
            frm.ShowDialog();
        }

        private void manageProvinceToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            ManageProvince frm = new ManageProvince();
            frm.ShowDialog();
        }

        private void manageCityTownToolStripMenuItem_Click(object sender,
EventArgs e)
        {
            ManageCityTown frm = new ManageCityTown();
            frm.ShowDialog();
        }


    }
}
```

ManageCityTown

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;


namespace ArchivingSystem
{
    public partial class ManageCityTown : Form
    {

        MySqlCommand cmd;
        MySqlConnection con;
        string query;

        int id = 0;

        public ManageCityTown()
        {
            InitializeComponent();
        }

        public void LoadData()
        {
            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from vw_citytown where province like ?key or
citytown like ?key";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");
               // cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();
                this.flx.AutoGenerateColumns = false;
                this.flx.DataSource = dt;
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        private void ManageCityTown_Load(object sender, EventArgs e)
        {
            Helper.cmbLoadProvince(cmbProv);
            LoadData();
        }

        private void btnRefresh_Click(object sender, EventArgs e)
        {
            LoadData();
            clear();
        }

        void clear()
        {
            cmbProv.SelectedIndex = -1;

            txtCity.Text = "";
            txtzip.Text = "";
            id = 0;
        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(cmbProv.Text))
            {
                Box.warnBox("Please select province name.");
                return;
            }

            if (String.IsNullOrEmpty(txtCity.Text))
            {
                Box.warnBox("Please input city.");
                return;
            }

            if (id == 0)
            {
                if (isExist())
                {
                    Box.warnBox("Province and City already in the database.");
                    return;
                }
            }


            processSave();
            LoadData();
```

```
            clear();
        }

        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();

                int provid = Helper.returnIDperConnection(con, "province", 
"province", cmbProv.Text, "provinceID");
                query = @"UPDATE city_town SET provinceID=?provID, cityTown=?city, 
zipcode=?zip
                        WHERE cityTownID = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?provID", provid);
                cmd.Parameters.AddWithValue("?city", txtCity.Text);
                cmd.Parameters.AddWithValue("?zip", txtzip.Text);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("City/Town successfully updated.");
                    // Helper.RecordLogs("UPDATE", txtlname.Text + ", " + 
txtfname.Text + " was successfully updated.");
                    Helper.RecordLogs("UPDATE", txtCity.Text + " was successfully 
updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }
            }
            else
            {
                con = Connection.con();
                con.Open();

                int provid = Helper.returnIDperConnection(con, "province", 
"province", cmbProv.Text, "provinceID");

                query = @"INSERT INTO city_town SET provinceID=?provID, 
cityTown=?city, zipcode=?zip";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?provID", provid);
                cmd.Parameters.AddWithValue("?city", txtCity.Text);
                cmd.Parameters.AddWithValue("?zip", txtzip.Text);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Province successfully saved.");
                    Helper.RecordLogs("SAVED", txtCity.Text + " was successfully
saved.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }

            }

        }




        bool isExist()
        {
            bool flag = false; ;
            con = Connection.con();
            con.Open();
            query = "select * from vw_citytown where province=?prov and
cityTown=?city";
            cmd = new MySqlCommand(query, con);
            cmd.Parameters.AddWithValue("?prov", cmbProv.Text);
            cmd.Parameters.AddWithValue("?city", txtCity.Text);
            MySqlDataReader dr;
            dr = cmd.ExecuteReader();
            while (dr.Read())
            {
                flag = true;
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            con.Dispose();
            return flag;

        }

        private void flx_DoubleClick(object sender, EventArgs e)
        {

            if (flx.Rows.Count > 1)
            {
                if (flx[flx.RowSel, "cityTownID"] != null)
                {
                    con = Connection.con();
                    con.Open();
                    query = "select * from vw_citytown where cityTownID=?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?id",
Convert.ToInt32(flx[flx.RowSel, "cityTownID"]));
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    while (dr.Read())
                    {
                        id = Convert.ToInt32(dr["cityTownID"]);
                        cmbProv.Text = Convert.ToString(dr["province"]);
                        txtCity.Text = Convert.ToString(dr["cityTown"]);
                        txtzip.Text = Convert.ToString(dr["zipcode"]);

                    }

                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();
                }
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (Box.questionBox("Are you sure you want to delete this row?",
"DELETE?"))
                {
                    int id = Convert.ToInt32(flx[flx.RowSel, 1]);
                    Helper.deleteByInt("city_town", "cityTownID", id);
                    Box.infoBox("Deleted successfully.");
                    LoadData();
                }
            }
            else
            {
                Box.warnBox("Please select City/Town in the list.");
            }
        }
    }
}
```

ManageInstitute

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class ManageInstitute : Form
    {
        MySqlConnection con;
        MySqlCommand cmd;
        string query;

        int id;

        public ManageInstitute()
        {
            InitializeComponent();
        }


        public void LoadData()
        {


            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from institute where instituteCode like ?key or
instituteDesc like ?key order by instituteCode asc";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");
                //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flx.AutoGenerateColumns = false;
                this.flx.DataSource = dt;
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }


        private void flxStdList_DoubleClick(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (flx[flx.RowSel, "instituteID"] != null)
                {
                    con = Connection.con();
                    con.Open();
                    query = "select * from institute where instituteID=?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?id",
Convert.ToInt32(flx[flx.RowSel, "instituteID"]));
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    while (dr.Read())
                    {
                        id = Convert.ToInt32(dr["instituteID"]);
                        txtInstituteCode.Text =
Convert.ToString(dr["instituteCode"]);
                        txtInstitute.Text = Convert.ToString(dr["instituteDesc"]);

                    }
                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();
                }

            }
        }

        private void txtsearch_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (Box.questionBox("Are you sure you want to delete this row?",
"DELETE?"))
                {
                    int id = Convert.ToInt32(flx[flx.RowSel, 1]);
                    Helper.deleteByInt("institute", "instituteID", id);
                    Box.infoBox("Deleted successfully.");
                    LoadData();
                }
            }
            else
            {
                Box.warnBox("Please select institute in the list.");
            }
        }

        void clear()
        {
            id = 0;
            txtInstituteCode.Text = "";
            txtInstitute.Text = "";
            txtInstituteCode.Focus();
        }



        private void btnRefresh_Click(object sender, EventArgs e)
        {
            LoadData();
        }
        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();
                query = @"UPDATE institute SET instituteCode=?insCode, 
instituteDesc=?ins 
                        WHERE instituteID = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?insCode", txtInstituteCode.Text);
                cmd.Parameters.AddWithValue("?ins", txtInstitute.Text);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Institute successfully updated.");
                    Helper.RecordLogs("UPDATE", txtInstituteCode.Text + " was 
successfully updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }
            }
            else
            {
                con = Connection.con();
                con.Open();
                query = @"INSERT institute SET instituteCode=?insCode, 
instituteDesc=?ins";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?insCode", txtInstituteCode.Text);
                cmd.Parameters.AddWithValue("?ins", txtInstitute.Text);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Insitute successfully saved.");
                    Helper.RecordLogs("SAVE", txtInstituteCode.Text + " was 
successfully updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }

            }
        }
        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtInstituteCode.Text))
            {
                Box.warnBox("Please input institute code.");
                return;
            }
            if (String.IsNullOrEmpty(txtInstitute.Text))
            {
                Box.warnBox("Please input institute description.");
                return;
            }

            if(Helper.isExistString("institute","instituteCode",
txtInstituteCode.Text))
            {
                Box.warnBox("Institute Code already exist.");
                return;
            }
            processSave();
            LoadData();
            clear();
        }

        private void ManageInstitute_Load(object sender, EventArgs e)
        {
            Helper.setAutoComplete(txtsearch, "select distinct(instituteCode) from
institute order by instituteCode asc", "instituteCode");

            LoadData();
        }
    }
}
```

ManageProgram

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class ManagePrograms : Form
    {

        MySqlConnection con;
        MySqlCommand cmd;
        string query;
        int id;


        public ManagePrograms()
        {
            InitializeComponent();
        }


        public void LoadData()
        {
            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from programs where programCode like ?key or
programDesc like ?key order by programCode asc";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");
                //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flx.AutoGenerateColumns = false;
                this.flx.DataSource = dt;
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        private void ManagePrograms_Load(object sender, EventArgs e)
        {
            try
            {
                Helper.setAutoComplete(txtsearch, "select distinct(programCode)
from programs order by programCode asc", "programCode");
                Helper.cmbLoadInstitute(cmbInstitute);

                LoadData();

              //  Helper.setAutoComplete(txtProgramCode, "select * from programs
order by programs asc", "programCode");
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
              //  throw;
            }

        }
        private void btnRefresh_Click(object sender, EventArgs e)
        {
            LoadData();
            clear();
        }

        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();

                int insid = Helper.returnIDperConnection(con, "institute", 
"instituteCode", cmbInstitute.Text, "instituteID");

                query = @"UPDATE programs SET programCode=?pcode, 
programDesc=?pdesc, major=?major, instituteID=?insID
                        WHERE programsID = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?pcode", txtProgramCode.Text);
                cmd.Parameters.AddWithValue("?pdesc", txtProgramDesc.Text);
                cmd.Parameters.AddWithValue("?major", txtMajor.Text);
                cmd.Parameters.AddWithValue("?insID", insid);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Program successfully updated.");
                    Helper.RecordLogs("UPDATE", txtProgramCode.Text + " was 
successfully updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }
            }
            else
            {
                con = Connection.con();
                con.Open();

                int insid = Helper.returnIDperConnection(con, "institute", 
"instituteCode", cmbInstitute.Text, "instituteID");

                query = @"INSERT INTO programs SET programCode=?pcode, 
programDesc=?pdesc, major=?major, instituteID=?insID";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?pcode", txtProgramCode.Text);
                cmd.Parameters.AddWithValue("?pdesc", txtProgramDesc.Text);
                cmd.Parameters.AddWithValue("?major", txtMajor.Text);
                cmd.Parameters.AddWithValue("?insID", insid);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Program successfully saved.");
                    Helper.RecordLogs("SAVE", txtProgramCode.Text + " was 
successfully updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }

            }
        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtProgramCode.Text))
            {
                Box.warnBox("Please input program code.");
                return;
            }
            if (String.IsNullOrEmpty(txtProgramDesc.Text))
            {
                Box.warnBox("Please input program description.");
                return;
            }

            if (String.IsNullOrEmpty(cmbInstitute.Text))
            {
                Box.warnBox("Please select institute.");
                return;
            }

            processSave();
            LoadData();
            clear();
        }


        void clear()
        {
            id = 0;
            txtProgramCode.Text = "";
            txtProgramDesc.Text = "";
            txtMajor.Text = "";
            cmbInstitute.SelectedIndex = -1;
            txtProgramCode.Focus();
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (Box.questionBox("Are you sure you want to delete this row?",
"DELETE?"))
                {
                    int id = Convert.ToInt32(flx[flx.RowSel, 1]);
                    Helper.deleteByInt("programs", "programsID", id);
                    Box.infoBox("Deleted successfully.");
                    LoadData();
                }
            }
            else
            {
                Box.warnBox("Please select programs in the list.");
            }
        }

        private void flx_DoubleClick(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (flx[flx.RowSel, "programsID"] != null)
                {
                    con = Connection.con();
                    con.Open();
                    query = "select * from vw_programs where programsID=?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?id",
Convert.ToInt32(flx[flx.RowSel, "programsID"]));
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    while (dr.Read())
                    {
                        id = Convert.ToInt32(dr["programsID"]);
                        txtProgramCode.Text = Convert.ToString(dr["programCode"]);
                        txtProgramDesc.Text = Convert.ToString(dr["programDesc"]);
                        txtMajor.Text = Convert.ToString(dr["major"]);
                        cmbInstitute.Text = Convert.ToString(dr["instituteCode"]);

                    }
                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();
                }

            }
        }

        private void txtsearch_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }
    }
}
```

ManageProvince

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class ManageProvince : Form
    {

        MySqlCommand cmd;
        MySqlConnection con;
        string query;

        int id = 0;


        public ManageProvince()
        {
            InitializeComponent();
        }

        public void LoadData()
        {


            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from province where province like ?prov";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?prov", txtsearch.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flx.AutoGenerateColumns = false;
                this.flx.DataSource = dt;
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        private void ManageProvince_Load(object sender, EventArgs e)
        {
            //   Helper.setAutoComplete(txtsearch, "select distinct(schoolName)
from hs_address order by schoolName asc", "schoolName");

            LoadData();
        }

        private void flx_DoubleClick(object sender, EventArgs e)
        {

            if (flx.Rows.Count > 1)
            {
                if (flx[flx.RowSel, "provinceID"] != null)
                {
                    con = Connection.con();
                    con.Open();
                    query = "select * from province where provinceID=?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?id",
Convert.ToInt32(flx[flx.RowSel, "provinceID"]));
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    while (dr.Read())
                    {
                        id = Convert.ToInt32(dr["provinceID"]);
                        txtProvince.Text = Convert.ToString(dr["province"]);

                    }

                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();
                }
            }
        }

        void clear()
        {
            txtProvince.Text = "";
            id = 0;
        }

        private void btnRefresh_Click(object sender, EventArgs e)
        {
            LoadData();
            clear();
        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtProvince.Text))
            {
                Box.warnBox("Please input school name.");
                return;
            }

            if (id == 0)
            {
                if (Helper.isExistString("province", "province",
txtProvince.Text))
                {
                    Box.warnBox("Province already in the database.");
                    return;
                }
            }


            processSave();
            LoadData();
            clear();
        }

        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();
                query = @"UPDATE province SET province=?prov
                        WHERE provinceID = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?prov", txtProvince.Text);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Province successfully updated.");
                    // Helper.RecordLogs("UPDATE", txtlname.Text + ", " +
txtfname.Text + " was successfully updated.");
                    Helper.RecordLogs("UPDATE", txtProvince.Text + " was
successfully saved.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }
            }
            else
            {
                con = Connection.con();
                con.Open();
                query = @"INSERT INTO province SET province=?prov";

                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?prov", txtProvince.Text);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Province successfully saved.");
                    Helper.RecordLogs("SAVED", txtProvince.Text + " was
successfully saved.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }

            }

        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            if (flx.Rows.Count > 1)
            {
                if (Box.questionBox("Are you sure you want to delete this row?",
"DELETE?"))
                {
                    int id = Convert.ToInt32(flx[flx.RowSel, 1]);
                    Helper.deleteByInt("province", "provinceID", id);
                    Box.infoBox("Deleted successfully.");
                    LoadData();
                }
            }
            else
            {
                Box.warnBox("Please select province in the list.");
            }
        }

        private void txtsearch_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }
    }
}
```

ManageSchool

```
using System;

using System.Collections.Generic;

using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
  public partial class ManageSchool : Form
  {
    MySqlCommand cmd;
    MySqlConnection con;
    string query;

    int id = 0;

    public ManageSchool()
    {
      InitializeComponent();
    }

    public void LoadData()
    {



      try
      {
            con = Connection.con();

            con.Open();

            query = "select * from hs_address where schoolName like ?key or province like ?key or
cityTown like ?key or brgy like ?key order by schoolName asc";

            cmd = new MySqlCommand(query, con);

            cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");

            DataTable dt = new DataTable();

            MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);

            adptr.Fill(dt);

            adptr.Dispose();

            cmd.Dispose();

            con.Close();


            this.flx.AutoGenerateColumns = false;

            this.flx.DataSource = dt;

        }

        catch (Exception er)

        {

            Box.errBox(er.Message);

            //throw;

        }

    }


    private void ManageSchool_Load(object sender, EventArgs e)

    {

        Helper.setAutoComplete(txtsearch, "select distinct(schoolName) from hs_address order
by schoolName asc", "schoolName");

        Helper.cmbLoadProvince(cmbProvince);


        LoadData();
    }


    void loadCmbCity()
    {
        con = Connection.con();
        con.Open();
        query = "select * from vw_citytown where province = ?prov order by cityTown asc";
        cmd = new MySqlCommand(query, con);
        cmd.Parameters.AddWithValue("?prov", cmbProvince.Text.Trim());
        cmbCity.Items.Clear();
        MySqlDataReader dr;
        dr = cmd.ExecuteReader();
        while(dr.Read()){
            cmbCity.Items.Add(Convert.ToString(dr["cityTown"]));
        }
        dr.Close();
        con.Close();
        con.Dispose();
    }



    void loadZip()
    {
        con = Connection.con();
        con.Open();
        query = "select * from vw_citytown where province = ?prov and cityTown=?cityTown
order by cityTown asc";
        cmd = new MySqlCommand(query, con);
        cmd.Parameters.AddWithValue("?prov", cmbProvince.Text.Trim());
        cmd.Parameters.AddWithValue("?cityTown", cmbCity.Text.Trim());
       // cmbCity.Items.Clear();
        MySqlDataReader dr;
        dr = cmd.ExecuteReader();
        while (dr.Read())
        {
            txtZipcode.Text = Convert.ToString(dr["zipcode"]);
        }
        dr.Close();
        con.Close();
        con.Dispose();
    }

    private void flx_DoubleClick(object sender, EventArgs e)
    {
        if (flx.Rows.Count > 1)
        {
            if (flx[flx.RowSel, "HSAddressID"] != null)
            {
                con = Connection.con();
                con.Open();
                query = "select * from hs_address where HSAddressID=?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?id", Convert.ToInt32(flx[flx.RowSel,
"HSAddressID"]));
                MySqlDataReader dr;
                dr = cmd.ExecuteReader();
                while (dr.Read())
                {
```

```
                id = Convert.ToInt32(dr["HSAddressID"]);
                txtSchoolName.Text = Convert.ToString(dr["schoolName"]);
                txtDesc.Text = Convert.ToString(dr["schoolDesc"]);
                cmbProvince.Text = Convert.ToString(dr["province"]);
                cmbCity.Text = Convert.ToString(dr["cityTown"]);
                txtBarangay.Text = Convert.ToString(dr["brgy"]);
                txtUnitSt.Text = Convert.ToString(dr["stUnit"]);
                txtZipcode.Text = Convert.ToString(dr["zipcode"]);

            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            con.Dispose();
        }

    }
}


void clear()
{
    txtSchoolName.Text = "";
    txtDesc.Text = "";
    cmbProvince.SelectedIndex = -1;
    cmbCity.SelectedIndex = -1;
    txtBarangay.Text = "";
    txtUnitSt.Text = "";
    txtZipcode.Text = "";
    id = 0;
}

private void btnRefresh_Click(object sender, EventArgs e)
{
    LoadData();
    clear();
}

private void btnSave_Click(object sender, EventArgs e)
{
    if (String.IsNullOrEmpty(txtSchoolName.Text))
    {
        Box.warnBox("Please input school name.");
        return;
    }

    if(id == 0)
    {
        if (Helper.isExistString("hs_address", "schoolName", txtSchoolName.Text))
        {
            Box.warnBox("School name already in the database.");
            return;
        }
    }



    if (String.IsNullOrEmpty(cmbProvince.Text))
    {
            Box.warnBox("Please input province.");
            return;
        }
        if (String.IsNullOrEmpty(cmbCity.Text))
        {
            Box.warnBox("Please input City/Town.");
            return;
        }

        processSave();
        LoadData();
        clear();

    }



    void processSave()
    {
        if (id > 0)
        {
            con = Connection.con();
            con.Open();
            query = @"UPDATE hs_address SET schoolName=?scholname,  schoolDesc=?desc,
province=?prov, cityTown=?city,
                brgy=?brgy, stUnit=?sunit, zipcode=?zip
                WHERE HSAddressID = ?id";
            cmd = new MySqlCommand(query, con);
            cmd.Parameters.AddWithValue("?scholname", txtSchoolName.Text);
            cmd.Parameters.AddWithValue("?desc", txtDesc.Text);
            cmd.Parameters.AddWithValue("?prov", cmbProvince.Text);
            cmd.Parameters.AddWithValue("?city", cmbCity.Text);
            cmd.Parameters.AddWithValue("?brgy", txtBarangay.Text);
            cmd.Parameters.AddWithValue("?sunit", txtUnitSt.Text);
            cmd.Parameters.AddWithValue("?zip", txtZipcode.Text);
            cmd.Parameters.AddWithValue("?id", id);
            int n = cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
            con.Dispose();


            if (n > 0)
            {
                Box.infoBox("School successfully updated.");
                // Helper.RecordLogs("UPDATE", txtlname.Text + ", " + txtfname.Text + " was
successfully updated.");
                Helper.RecordLogs("UPDATE", txtSchoolName.Text + " was successfully saved.");
            }
            else
            {
                Box.errBox("An error occured. Please contact administrator.");
            }
        }
        else
        {
            con = Connection.con();
            con.Open();
            query = @"INSERT INTO hs_address SET schoolName=?scholname,  schoolDesc=?desc,
province=?prov, cityTown=?city,
                  brgy=?brgy, stUnit=?sunit, zipcode=?zip";
            cmd = new MySqlCommand(query, con);
            cmd.Parameters.AddWithValue("?scholname", txtSchoolName.Text);
            cmd.Parameters.AddWithValue("?desc", txtDesc.Text);
            cmd.Parameters.AddWithValue("?prov", cmbProvince.Text);
            cmd.Parameters.AddWithValue("?city", cmbCity.Text);
            cmd.Parameters.AddWithValue("?brgy", txtBarangay.Text);
            cmd.Parameters.AddWithValue("?sunit", txtUnitSt.Text);
            cmd.Parameters.AddWithValue("?zip", txtZipcode.Text);
            int n = cmd.ExecuteNonQuery();
            cmd.Dispose();
            con.Close();
            con.Dispose();

            if (n > 0)
            {
                Box.infoBox("School successfully saved.");
                Helper.RecordLogs("SAVED", txtSchoolName.Text + " was successfully saved.");
            }
            else
            {
                Box.errBox("An error occured. Please contact administrator.");
            }

        }
    }


    private void btnDelete_Click(object sender, EventArgs e)
    {
        if (flx.Rows.Count > 1)
        {
            if (Box.questionBox("Are you sure you want to delete this row?", "DELETE?"))
            {
                int id = Convert.ToInt32(flx[flx.RowSel, 1]);
                Helper.deleteByInt("hs_address", "HSAddressID", id);
                Box.infoBox("Deleted successfully.");
                LoadData();
            }
        }
        else
        {
            Box.warnBox("Please select School in the list.");
        }
    }

    private void txtsearch_KeyDown(object sender, KeyEventArgs e)
    {
        if(e.KeyCode == Keys.Enter)
        {
            LoadData();
        }
    }

    private void cmbProvince_SelectedIndexChanged(object sender, EventArgs e)
    {
        loadCmbCity();
    }
```

```
    private void cmbCity_SelectedIndexChanged(object sender, EventArgs e)

    {

      loadZip();

    }

  }

}
```

PrintDismissal

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class Print_Dismissal : Form
    {


        MySqlConnection con;
        MySqlCommand cmd;
        string query;


        public string stdIDNo;
        public string orNo;

        public Print_Dismissal()
        {
            InitializeComponent();
        }


        void LoadReport()
        {

            try
            {
                //int gid = Helper.returnGroupID(cmbGroup);

                con = Connection.con();
                con.Open();
                query = "proc_dismissal";
                cmd = new MySqlCommand(query, con);
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("?vstdIDNo", stdIDNo);
                cmd.Parameters.AddWithValue("?vORNo", orNo);

                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Report_Dismissal rpt = new Report_Dismissal();
                rpt.SetDataSource(dt);
                crystalReportViewer1.ReportSource = rpt;
                crystalReportViewer1.Refresh();



            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //   throw;
            }
        }

        private void Print_Dismissal_Load(object sender, EventArgs e)
        {
            LoadReport();
        }
    }
}
```

PrintLackingCredentials

```
       using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class Print_LackingCred : Form
    {

        MySqlConnection con;
        MySqlCommand cmd;
        string query;


        public string stdIDNo;


        public Print_LackingCred()
        {
            InitializeComponent();
        }


        void LoadReport()
        {

            try
            {
                //int gid = Helper.returnGroupID(cmbGroup);

                con = Connection.con();
                con.Open();
                query = "proc_lackingcred";
                cmd = new MySqlCommand(query, con);
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("?vstdIDNo", stdIDNo);

                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Report_LackingCred rpt = new Report_LackingCred();
                rpt.SetDataSource(dt);
                crystalReportViewer1.ReportSource = rpt;
                crystalReportViewer1.Refresh();



            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //   throw;
            }
        }

        private void Print_LackingCred_Load(object sender, EventArgs e)
        {
            LoadReport();
        }
    }
}
```

PrintListLackingCredentials

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class Print_ListLackingCred : Form
    {
        MySqlCommand cmd;
        MySqlConnection con;
        string query;


        public Print_ListLackingCred()
        {
            InitializeComponent();
        }


        void LoadReport()
        {

            try
            {
                //int gid = Helper.returnGroupID(cmbGroup);

                con = Connection.con();
                con.Open();
                query = "proc_stdlist_lackingcred";
                cmd = new MySqlCommand(query, con);
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("?vstdIDNo", "");
                cmd.Parameters.AddWithValue("?vInstitute", cmbInstitute.Text);
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Report_ListStdLackingCred rpt = new Report_ListStdLackingCred();
                rpt.SetDataSource(dt);
                crystalReportViewer1.ReportSource = rpt;
                crystalReportViewer1.Refresh();
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //   throw;
            }
        }


        private void Print_ListLackingCred_Load(object sender, EventArgs e)
        {
            con = Connection.con();
            con.Open();
            Helper.LoadCmbPerCon(con, "select distinct(instituteCode) from
institute order by instituteCode", cmbInstitute);
            con.Close();
            con.Dispose();


        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            LoadReport();
        }
    }
}
```

<u>PrintReqeustForm</u>

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;


namespace ArchivingSystem
{
    public partial class Print_RequestForm : Form
    {
        MySqlConnection con;
        MySqlCommand cmd;
        string query;


        public Print_RequestForm()
        {
            InitializeComponent();
        }

        void LoadReport()
        {
            con = Connection.con();
            con.Open();
            query = "proc_requestform";
            cmd = new MySqlCommand(query, con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("?vschoolName", cmbSchool.Text);
            DataTable dt = new DataTable();
            MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
            adptr.Fill(dt);
            cmd.Dispose();
            adptr.Dispose();
            con.Close();
            con.Dispose();

            Report_RequestForm rpt = new Report_RequestForm();
            rpt.SetDataSource(dt);
            crystalReportViewer1.ReportSource = rpt;
            crystalReportViewer1.Refresh();

        }

        private void Print_RequestForm_Load(object sender, EventArgs e)
        {
            con = Connection.con();
            con.Open();
            Helper.LoadCmbPerCon(con, "select distinct(schoolName) from hs_address
order by schoolName asc", cmbSchool);
            con.Close();
            con.Dispose();


        }

        private void btnSearch_Click(object sender, EventArgs e)
        {
            LoadReport();
        }
    }
}
```

Program

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Threading.Tasks;
using System.Windows.Forms;

namespace ArchivingSystem
```

```
{
    static class Program
    {
        /// <summary>
        /// The main entry point for the application.
        /// </summary>
        [STAThread]
        static void Main()
        {
            Application.EnableVisualStyles();
            Application.SetCompatibleTextRenderingDefault(false);
            Application.Run(new Login());
        }
    }
}
```

Scanner

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Runtime.InteropServices;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using WIA;

namespace ArchivingSystem
{
    class Scanner
    {

        private readonly DeviceInfo _deviceInfo;
        private int resolution = 150;
        private int width_pixel = 1250;
        private int height_pixel = 1700;
        private int color_mode = 1;

        public Scanner(DeviceInfo deviceInfo)
        {
            this._deviceInfo = deviceInfo;
        }

        /// <summary>
        /// Scan a image with PNG Format
        /// </summary>
        /// <returns></returns>
        public ImageFile ScanPNG()
        {
            // Connect to the device and instruct it to scan
            // Connect to the device
            var device = this._deviceInfo.Connect();

            // Select the scanner
            CommonDialogClass dlg = new CommonDialogClass();
            var item = device.Items[1];

            try
            {
                AdjustScannerSettings(item, resolution, 0, 0, width_pixel,
height_pixel, 0, 0, color_mode);

                object scanResult = dlg.ShowTransfer(item,
WIA.FormatID.wiaFormatPNG, true);

                if (scanResult != null)
                {
                    var imageFile = (ImageFile)scanResult;

                    // Return the imageFile
                    return imageFile;
                }
            }
            catch (COMException e)
            {
                // Display the exception in the console.
                Console.WriteLine(e.ToString());

                uint errorCode = (uint)e.ErrorCode;

                // Catch 2 of the most common exceptions
                if (errorCode == 0x80210006)
                {
                    MessageBox.Show("The scanner is busy or isn't ready");
                }
                else if (errorCode == 0x80210064)
                {
                    MessageBox.Show("The scanning process has been cancelled.");
                }
                else
                {
                    MessageBox.Show("A non catched error occurred, check the
console", "Error", MessageBoxButtons.OK);
                }
            }

            return new ImageFile();
        }

        /// <summary>
        /// Scan a image with JPEG Format
        /// </summary>
        /// <returns></returns>
        public ImageFile ScanJPEG()
        {
            // Connect to the device and instruct it to scan
            // Connect to the device
            var device = this._deviceInfo.Connect();

            // Select the scanner
            CommonDialogClass dlg = new CommonDialogClass();
            var item = device.Items[1];

            try
            {
                AdjustScannerSettings(item, resolution, 0, 0, width_pixel,
height_pixel, 0, 0, color_mode);

                object scanResult = dlg.ShowTransfer(item,
WIA.FormatID.wiaFormatJPEG, true);

                if (scanResult != null)
                {
                    var imageFile = (ImageFile)scanResult;

                    // Return the imageFile
                    return imageFile;
                }
            }
            catch (COMException e)
            {
                // Display the exception in the console.
                Console.WriteLine(e.ToString());

                uint errorCode = (uint)e.ErrorCode;

                // Catch 2 of the most common exceptions
                if (errorCode == 0x80210006)
                {
                    MessageBox.Show("The scanner is busy or isn't ready");
                }
                else if (errorCode == 0x80210064)
                {
                    MessageBox.Show("The scanning process has been cancelled.");
                }
                else
                {
                    MessageBox.Show("A non catched error occurred, check the
console", "Error", MessageBoxButtons.OK);
                }
            }

            return new ImageFile();
        }

        /// <summary>
        /// Scan a image with TIFF Format
        /// </summary>
        /// <returns></returns>
        public ImageFile ScanTIFF()
        {
            // Connect to the device and instruct it to scan
            // Connect to the device
            var device = this._deviceInfo.Connect();

            // Select the scanner
            CommonDialogClass dlg = new CommonDialogClass();

            var item = device.Items[1];
            try
            {
                AdjustScannerSettings(item, resolution, 0, 0, width_pixel,
height_pixel, 0, 0, color_mode);

                object scanResult = dlg.ShowTransfer(item,
WIA.FormatID.wiaFormatTIFF, true);

                if (scanResult != null)
                {
                    var imageFile = (ImageFile)scanResult;

                    // Return the imageFile
                    return imageFile;
                }
            }
            catch (COMException e)
            {
                // Display the exception in the console.
                Console.WriteLine(e.ToString());

                uint errorCode = (uint)e.ErrorCode;

                // Catch 2 of the most common exceptions
                if (errorCode == 0x80210006)
                {
                    MessageBox.Show("The scanner is busy or isn't ready");
                }
                else if (errorCode == 0x80210064)
                {
                    MessageBox.Show("The scanning process has been cancelled.");
                }
                else
                {
                    MessageBox.Show("A non catched error occurred, check the
console", "Error", MessageBoxButtons.OK);
                }
            }

            return new ImageFile();
        }

        /// <summary>
        /// Adjusts the settings of the scanner with the providen parameters.
        /// </summary>
        /// <param name="scannnerItem">Expects a </param>
        /// <param name="scanResolutionDPI">Provide the DPI resolution that should
be used e.g 150</param>
        /// <param name="scanStartLeftPixel"></param>
        /// <param name="scanStartTopPixel"></param>
        /// <param name="scanWidthPixels"></param>
        /// <param name="scanHeightPixels"></param>
        /// <param name="brightnessPercents"></param>
        /// <param name="contrastPercents">Modify the contrast percent</param>
        /// <param name="colorMode">Set the color mode</param>
        private static void AdjustScannerSettings(IItem scannnerItem, int
scanResolutionDPI, int scanStartLeftPixel, int scanStartTopPixel, int
```

```
scanWidthPixels, int scanHeightPixels, int brightnessPercents, int
contrastPercents, int colorMode)
        {
            const string WIA_SCAN_COLOR_MODE = "6146";
            const string WIA_HORIZONTAL_SCAN_RESOLUTION_DPI = "6147";
            const string WIA_VERTICAL_SCAN_RESOLUTION_DPI = "6148";
            const string WIA_HORIZONTAL_SCAN_START_PIXEL = "6149";
            const string WIA_VERTICAL_SCAN_START_PIXEL = "6150";
            const string WIA_HORIZONTAL_SCAN_SIZE_PIXELS = "6151";
            const string WIA_VERTICAL_SCAN_SIZE_PIXELS = "6152";
            const string WIA_SCAN_BRIGHTNESS_PERCENTS = "6154";
            const string WIA_SCAN_CONTRAST_PERCENTS = "6155";
            SetWIAProperty(scannnerItem.Properties,
WIA_HORIZONTAL_SCAN_RESOLUTION_DPI, scanResolutionDPI);
            SetWIAProperty(scannnerItem.Properties,
WIA_VERTICAL_SCAN_RESOLUTION_DPI, scanResolutionDPI);
            SetWIAProperty(scannnerItem.Properties,
WIA_HORIZONTAL_SCAN_START_PIXEL, scanStartLeftPixel);
            SetWIAProperty(scannnerItem.Properties, WIA_VERTICAL_SCAN_START_PIXEL,
scanStartTopPixel);
            SetWIAProperty(scannnerItem.Properties,
WIA_HORIZONTAL_SCAN_SIZE_PIXELS, scanWidthPixels);
            SetWIAProperty(scannnerItem.Properties, WIA_VERTICAL_SCAN_SIZE_PIXELS,
scanHeightPixels);
            SetWIAProperty(scannnerItem.Properties, WIA_SCAN_BRIGHTNESS_PERCENTS,
brightnessPercents);
            SetWIAProperty(scannnerItem.Properties, WIA_SCAN_CONTRAST_PERCENTS,
contrastPercents);
            SetWIAProperty(scannnerItem.Properties, WIA_SCAN_COLOR_MODE,
colorMode);
        }

        /// <summary>
        ///
        /// </summary>
        /// <param name="properties"></param>
        /// <param name="propName"></param>
        /// <param name="propValue"></param>
        private static void SetWIAProperty(IProperties properties, object
propName, object propValue)
        {
            Property prop = properties.get_Item(ref propName);
            prop.set_Value(ref propValue);
        }

        /// <summary>
        /// Declare the ToString method
        /// </summary>
        /// <returns></returns>
        public override string ToString()
        {
            return (string)this._deviceInfo.Properties["Name"].get_Value();
        }

    }
}
```

ScanUploader

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
using WIA;
using System.IO;
using System.Drawing.Imaging;

namespace ArchivingSystem
{
    public partial class ScanUploader : Form
    {
        MySqlConnection con;
        MySqlCommand cmd;
        string query;
        public int credsetupid, studentid, credid;
        public bool isUpdate;


        SubmitCredential _frm;


        OpenFileDialog open;


        Scanner device = null;

        public string formName;


        public ScanUploader(SubmitCredential _frm)
        {
            InitializeComponent();
            this._frm = _frm;
            open = new OpenFileDialog();
        }

        private void ListScanners()
        {
            // Clear the ListBox.
            listBox1.Items.Clear();

            // Create a DeviceManager instance
            var deviceManager = new DeviceManager();

            // Loop through the list of devices and add the name to the listbox
            for (int i = 1; i <= deviceManager.DeviceInfos.Count; i++)
            {
                // Add the device only if it's a scanner
                if (deviceManager.DeviceInfos[i].Type !=
WiaDeviceType.ScannerDeviceType)
                {
                    continue;
                }

                // Add the Scanner device to the listbox (the entire DeviceInfos
object)
                // Important: we store an object of type scanner (which ToString
method returns the name of the scanner)
                listBox1.Items.Add(
                    new Scanner(deviceManager.DeviceInfos[i])
                );
            }
        }


        private void TriggerScan()
        {
            //Console.WriteLine("Image succesfully scanned");
            Box.infoBox("Image successfully scanned.");
            Helper.RecordLogs("SCANNED", "NEW FILE SUCCESSFULLY SCANNED!");
        }

        public void StartScanning()
        {
            //  Scanner device = null;

            ImageFile image = new ImageFile();
            string imageExtension = "";

            this.Invoke(new MethodInvoker(delegate ()
            {
                switch (cmbImageFormat.SelectedIndex)
                {
                    case 0:
                        image = device.ScanPNG();
                        imageExtension = ".png";
                        break;
                    case 1:
                        image = device.ScanJPEG();
                        imageExtension = ".jpeg";
                        break;
                    case 2:
                        image = device.ScanTIFF();
                        imageExtension = ".tiff";
                        break;
                }
            }));


            // Save the image
            // var path = Path.Combine("c:\\", "test" + imageExtension);

            if (b != null)
            {
                b.Dispose();
            }

            string path = "c:\\test" + imageExtension;

            if (File.Exists(path))
            {

                // Box.infoBox("File exist. Continue to overwrite the file.");
                File.Delete(path);
            }

            image.SaveFile(path);
            //using(Bitmap b = new Bitmap(path))
            //{
            //    pictureBox1.Image = b;
            //}
            b = new Bitmap(path);
            pictureBox1.Image = b;

        }

        Bitmap b;

        private void ScanUploader_Load(object sender, EventArgs e)
        {
          //  Box.infoBox(credsetupid.ToString());
            txtFormName.Text = formName;
            txtFormName.ReadOnly = true;

            ListScanners();

            //txtoutput.Text = Path.GetTempPath();

            cmbImageFormat.SelectedIndex = 1;

        }

        private void btnScanDoc_Click(object sender, EventArgs e)
        {

            this.Invoke(new MethodInvoker(delegate ()
            {
                device = listBox1.SelectedItem as Scanner;
            }));

            if (device == null)
            {
                MessageBox.Show("You need to select first an scanner device from
the list",
                                "Warning",
                                MessageBoxButtons.OK, MessageBoxIcon.Warning);
                return;
            }

            Task.Factory.StartNew(StartScanning).ContinueWith(result =>
TriggerScan());

        }
        private void btnSaveDoc_Click(object sender, EventArgs e)
        {
            saveDoc(sender, e);

            this.Close();
        }


        void saveDoc (object sender, EventArgs e)
        {
            try
            {
                int i = 0;


                if (isUpdate == true)
                {
                    bool isNullOrEmptyPictureBox = pictureBox1 == null ||
pictureBox1.Image == null;
                    if (isNullOrEmptyPictureBox)
                    {
                        Box.warnBox("Please provide document.");
                        btnBrowse_Click(sender, e);
                        return;
                        // cmd.Parameters.AddWithValue("?vImage", null);
                    }


                    con = Connection.con();
                    con.Open();



                    query = @"UPDATE credential SET cred_setupID=?credsetupID,
stdSysID=?stdid, isCheck=1, nFile=?nfile WHERE credid=?credid";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?credsetupID", credsetupid);
                    cmd.Parameters.AddWithValue("?stdid", studentid);

                    Image image = pictureBox1.Image;
                    MemoryStream memoryStream = new MemoryStream();
                    image.Save(memoryStream, ImageFormat.Jpeg);
                    byte[] ImageData = memoryStream.ToArray();
                    cmd.Parameters.AddWithValue("?nfile", ImageData);

                    cmd.Parameters.AddWithValue("?credid", credid);
                    i = cmd.ExecuteNonQuery();

                    cmd.Dispose();
                    con.Close();
                    con.Dispose();




                    if (i > 0)
                    {
                        Box.infoBox("File successfully updated.");
                        Helper.RecordLogs("UPDATE", "ID " +credid + " was 
updated!");
                    }



                }
                else
                {
                    //Box.infoBox("insert");
                    con = Connection.con();
                    con.Open();

                    query = @"INSERT INTO credential (cred_setupID, stdSysID, 
isCheck, nFile) VALUES
                        (?credsetupID, ?stdid, ?ischeck, ?nFile)";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?credsetupID", credsetupid);
                    cmd.Parameters.AddWithValue("?stdid", studentid);
                    cmd.Parameters.AddWithValue("?ischeck", 1);


                    bool isNullOrEmptyPictureBox = pictureBox1 == null || 
pictureBox1.Image == null;
                    if (isNullOrEmptyPictureBox)
                    {
                        cmd.Parameters.AddWithValue("?nFile", null);
                    }
                    else
                    {
                        Image image = pictureBox1.Image;
                        MemoryStream memoryStream = new MemoryStream();
                        image.Save(memoryStream, ImageFormat.Jpeg);
                        byte[] ImageData = memoryStream.ToArray();
                        cmd.Parameters.AddWithValue("?nFile", ImageData);
                    }
                    i = cmd.ExecuteNonQuery();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();

                    if (i > 0)
                    {
                        Box.infoBox("File successfully saved.");
                        Helper.RecordLogs("SAVE", "Student ID : " + studentid + ", 
new file was uploaded.");
                    }

                }



                _frm.LoadCredRequirement();


            } //end try catch
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        private void btnBrowse_Click(object sender, EventArgs e)
        {
            try
            {
                open.InitialDirectory = "C:\\Users\\...\\Pictures";
                open.Filter = "JPG Files (*.jpg)|*.jpg";
                open.Title = "Select Picture";
                open.FilterIndex = 1;
                //open.ShowDialog();

                if (open.ShowDialog() == DialogResult.OK)
                {
                    //  imgpath = open.FileName.ToString();
                    //  pictureBox1.ImageLocation = imgpath;
                    // Box.infoBox(open.SafeFileName.ToString());
                    pictureBox1.ImageLocation = open.FileName.ToString();
                }
                else
                {
                    // Box.warnBox("User cancel upload");
                }

            }
            catch (Exception err)
            {
                MessageBox.Show(err.Message, "ERROR!", MessageBoxButtons.OK,
MessageBoxIcon.Error);
            }
        }
    }
}
```

StudentForm

```
using System;

using System.Collections.Generic;

using System.ComponentModel;

using System.Data;

using System.Drawing;

using System.Linq;

using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class StudentForm : Form
    {



        MySqlConnection con;
        MySqlCommand cmd;
        string query;

        int id;

        public StudentForm()
        {
            InitializeComponent();
            txtStdIDNo.KeyPress += Helper.txtInteger;

        }



        public void LoadData()
        {
            con = Connection.con();
            con.Open();
            query = "select * from vw_studentinfo where studentIDNo like ?key or stdLname like
?key or stdFname like ?key order by stdLname asc";
            cmd = new MySqlCommand(query, con);
           cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");
            //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
            DataTable dt = new DataTable();
            MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
            adptr.Fill(dt);
            adptr.Dispose();
            cmd.Dispose();
            con.Close();

            this.flxStdList.AutoGenerateColumns = false;
            this.flxStdList.DataSource = dt;

            try
            {

            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        private void StudentForm_Load(object sender, EventArgs e)
        {
            Helper.cmbLoadProgram(cmbPrograms);
            LoadData();


            Helper.setAutoComplete(txtschoolname, "select distinct(schoolName) from hs_address
order by schoolName asc", "schoolName");


        }


        private void flxStdList_DoubleClick(object sender, EventArgs e)
        {
            if (flxStdList.Rows.Count > 1)
            {
                if (flxStdList[flxStdList.RowSel, "stdSysID"] != null)
                {
                    con = Connection.con();
                    con.Open();
                    query = "select * from vw_studentinfo where stdSysID=?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?id", Convert.ToInt32(flxStdList[flxStdList.RowSel,
"stdSysID"]));
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    while (dr.Read())
                    {
                        id = Convert.ToInt32(dr["stdSysID"]);
                        txtStdIDNo.Text = Convert.ToString(dr["studentIDNo"]);
                        txtfname.Text = Convert.ToString(dr["stdFname"]);
                        txtmname.Text = Convert.ToString(dr["stdMname"]);
                        txtlname.Text = Convert.ToString(dr["stdLname"]);
                        cmbGender.Text = Convert.ToString(dr["stdGender"]);
                        cmbPrograms.Text = Convert.ToString(dr["programCode"]);
                        txtMajor.Text = Convert.ToString(dr["major"]);
                        txtInstitute.Text = Convert.ToString(dr["instituteCode"]);
                        txtyrlvl.Text = Convert.ToString(dr["yearLvl"]);

                        // Box.infoBox(Convert.ToString(dr["schoolName"]));

                        txtschoolname.Text = Convert.ToString(dr["schoolName"]);
                        txtprov.Text = Convert.ToString(dr["province"]);
                        txtcity.Text = Convert.ToString(dr["cityTown"]);
                        txtbrgy.Text = Convert.ToString(dr["brgy"]);
                        txtzip.Text = Convert.ToString(dr["zipcode"]);
                        txtYear.Text= Convert.ToString(dr["yearHSGrad"]);
                        txtsection.Text = Convert.ToString(dr["section"]);
                    }
                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();
                }

            }
        }

        private void cmbPrograms_SelectedIndexChanged(object sender, EventArgs e)
        {
            con = Connection.con();
            con.Open();
            query = "select * from vw_programs where programCode=?pcode";
            cmd = new MySqlCommand(query, con);
            cmd.Parameters.AddWithValue("?pcode", cmbPrograms.Text);
            MySqlDataReader dr;
            dr = cmd.ExecuteReader();
            while (dr.Read())
            {
                txtMajor.Text = Convert.ToString(dr["major"]);
                txtInstitute.Text = Convert.ToString(dr["instituteCode"]);
            }
            dr.Close();
            cmd.Dispose();
            con.Close();
            con.Dispose();
          //  Box.infoBox("test");
        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtStdIDNo.Text))
            {
                Box.warnBox("Please input ID No.");
                return;
            }

            if (String.IsNullOrEmpty(txtfname.Text))
            {
                Box.warnBox("Please input Firstname.");
                return;
            }
            if (String.IsNullOrEmpty(txtlname.Text))
```

```
{
   Box.warnBox("Please input Lastname.");
   return;
}

if (String.IsNullOrEmpty(cmbGender.Text))
{
   Box.warnBox("Please select gender.");
   return;
}

if (String.IsNullOrEmpty(cmbPrograms.Text))
{
   Box.warnBox("Please select Programs.");
   return;
}

if (String.IsNullOrEmpty(txtyrlvl.Text))
{
   Box.warnBox("Please input year level.");
   return;
}

if (String.IsNullOrEmpty(txtschoolname.Text))
{
   Box.warnBox("Please input school.");
   return;
}
```

```
            processSave();
            LoadData();
            clear();
        }

        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();

                int progid = Helper.returnIDperConnection(con, "programs", "programCode",
cmbPrograms.Text, "programsID");
                int addressid = Helper.returnIDperConnection(con, "hs_address", "schoolName",
txtschoolname.Text, "HSAddressID");

                query = @"UPDATE student SET studentIDNo=?stdid,  stdLname=?lname,
stdFname=?fname,
                    stdMname=?mname, stdGender=?gender, programID=?progid, yearLvl=?yrlvl,
HSAddressID=?hid, yearHSGrad=?yrhs, section=?section
                    WHERE stdSysID = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?stdid", txtStdIDNo.Text);
                cmd.Parameters.AddWithValue("?lname", txtlname.Text);
                cmd.Parameters.AddWithValue("?fname", txtfname.Text);
                cmd.Parameters.AddWithValue("?mname", txtmname.Text);
                cmd.Parameters.AddWithValue("?gender", cmbGender.Text);
                cmd.Parameters.AddWithValue("?progid", progid);
                cmd.Parameters.AddWithValue("?yrlvl", txtyrlvl.Text);
                cmd.Parameters.AddWithValue("?hid", addressid);
                cmd.Parameters.AddWithValue("?yrhs", txtYear.Text);
                cmd.Parameters.AddWithValue("?section", txtsection.Text);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Student successfully updated.");
                    Helper.RecordLogs("UPDATE", txtlname.Text + ", " + txtfname.Text + " was
successfully updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }
            }
            else
            {

                con = Connection.con();
                con.Open();

                int progid = Helper.returnIDperConnection(con, "programs", "programCode",
cmbPrograms.Text, "programsID");
                int addressid = Helper.returnIDperConnection(con, "hs_address", "schoolName",
txtschoolname.Text, "HSAddressID");
```

```
        query = @"INSERT INTO student SET studentIDNo=?stdid,  stdLname=?lname,
stdFname=?fname,
            stdMname=?mname, stdGender=?gender, programID=?progid, yearLvl=?yrlvl,
HSAddressID=?hid, yearHSGrad=?yrhs, section=?section";
        cmd = new MySqlCommand(query, con);
        cmd.Parameters.AddWithValue("?stdid", txtStdIDNo.Text);
        cmd.Parameters.AddWithValue("?lname", txtlname.Text);
        cmd.Parameters.AddWithValue("?fname", txtfname.Text);
        cmd.Parameters.AddWithValue("?mname", txtmname.Text);
        cmd.Parameters.AddWithValue("?gender", cmbGender.Text);
        cmd.Parameters.AddWithValue("?progid", progid);
        cmd.Parameters.AddWithValue("?yrlvl", txtyrlvl.Text);

        cmd.Parameters.AddWithValue("?hid", addressid);
        cmd.Parameters.AddWithValue("?yrhs", txtYear.Text);
        cmd.Parameters.AddWithValue("?section", txtsection.Text);

        int n = cmd.ExecuteNonQuery();
        cmd.Dispose();
        con.Close();
        con.Dispose();

        if (n > 0)
        {
          Box.infoBox("Student successfully saved.");
          Helper.RecordLogs("SAVED", txtlname.Text + ", " + txtfname.Text + " was successfully
saved.");
        }
        else
        {
            Box.errBox("An error occured. Please contact administrator.");
        }

    }
}

void clear()
{
    txtStdIDNo.Text = "";
    txtfname.Text = "";
    txtmname.Text = "";
    txtlname.Text = "";
    cmbGender.SelectedIndex = -1;
    cmbPrograms.SelectedIndex = -1;
    txtMajor.Text = "";
    txtInstitute.Text = "";
    txtyrlvl.Text = "";

    txtschoolname.Text = "";
    txtprov.Text = "";
    txtcity.Text = "";
    txtbrgy.Text = "";
    txtzip.Text = "";
    txtYear.Text = "";
    txtsection.Text = "";

    txtStdIDNo.Focus();
    id = 0;
```

```
}

private void btnRefresh_Click(object sender, EventArgs e)
{
   LoadData();
   clear();
}

private void txtschoolname_TextChanged(object sender, EventArgs e)
{
   getSchoolAddress();
}

void getSchoolAddress()
{
   con = Connection.con();
   con.Open();
   query = "select * from hs_address where schoolName=?sname";
   cmd = new MySqlCommand(query, con);
   cmd.Parameters.AddWithValue("?sname", txtschoolname.Text.Trim());
   MySqlDataReader dr;
   dr = cmd.ExecuteReader();
   while (dr.Read())
   {
      txtprov.Text = Convert.ToString(dr["province"]);
      txtcity.Text = Convert.ToString(dr["cityTown"]);
      txtbrgy.Text = Convert.ToString(dr["brgy"]);
      txtzip.Text = Convert.ToString(dr["zipcode"]);
```

```
    }
    dr.Close();
    cmd.Dispose();
    con.Close();
    con.Dispose();
}

private void btnDelete_Click(object sender, EventArgs e)
{
    if(flxStdList.Rows.Count > 1)
    {
        if (Box.questionBox("Are you sure you want to delete this row?", "DELETE?"))
        {
            int id = Convert.ToInt32(flxStdList[flxStdList.RowSel, 1]);
            Helper.deleteByInt("student", "stdSysID", id);
            Box.infoBox("Deleted successfully.");
            LoadData();

            string studidno = Convert.ToString(flxStdList[flxStdList.RowSel, "studentIDNo"]);
            Helper.RecordLogs("DELETE", studidno + " was deleted.");
        }
    }
    else
    {
        Box.warnBox("Please select student in the list.");
    }
}

private void txtsearch_KeyDown(object sender, KeyEventArgs e)
```

```
    {
        if(e.KeyCode == Keys.Enter)
        {
            LoadData();
        }
    }
  }
}
```

<u>StudentRecords</u>

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using MySql.Data.MySqlClient;
using System.Windows.Forms;

namespace ArchivingSystem
{
    public partial class StudentRecords : Form
    {
        MySqlConnection con;
        MySqlCommand cmd;
        string query;



        public StudentRecords()
        {
            InitializeComponent();
        }


        void LoadStudentTable()
        {
            try
            {
                con = Connection.con();
                con.Open();
                query = "select * from vw_studentinfo WHERE stdLname like ?lname
order by stdLname asc";
                cmd = new MySqlCommand(query, con);
```

```
               // cmd.Parameters.AddWithValue("?stdno", txtsearchID.Text + "%");
                cmd.Parameters.AddWithValue("?lname", txtsearchLname.Text + "%");
                //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flxStdList.AutoGenerateColumns = false;
                this.flxStdList.DataSource = dt;

                //if (dt.Rows.Count > 0)
                //{
                //    txtID.Text = Convert.ToString(dt.Rows[0]["studentIDNo"]);
                //    txtlname.Text = Convert.ToString(dt.Rows[0]["stdLname"]);
                //    txtfname.Text = Convert.ToString(dt.Rows[0]["stdFname"]);
                //    txtmname.Text = Convert.ToString(dt.Rows[0]["stdMname"]);
                //    txtgender.Text = Convert.ToString(dt.Rows[0]["stdGender"]);
                //    txtprogram.Text =
Convert.ToString(dt.Rows[0]["programCode"]);
                //    txtyear.Text = Convert.ToString(dt.Rows[0]["yearLvl"]);
                //    txtInstitute.Text =
Convert.ToString(dt.Rows[0]["instituteCode"]);
                //}
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        public void LoadData(string stdidno)
        {
            try
            {

                con = Connection.con();
                con.Open();
                query = "select * from vw_studentinfo WHERE studentIDno = ?stdno";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?stdno", stdidno);
                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                //this.flxStdList.AutoGenerateColumns = false;
               // this.flxStdList.DataSource = dt;

                if(dt.Rows.Count > 0)
                {
                    txtID.Text = Convert.ToString(dt.Rows[0]["studentIDNo"]);
                    txtlname.Text = Convert.ToString(dt.Rows[0]["stdLname"]);
                    txtfname.Text = Convert.ToString(dt.Rows[0]["stdFname"]);
                    txtmname.Text = Convert.ToString(dt.Rows[0]["stdMname"]);
                    txtgender.Text = Convert.ToString(dt.Rows[0]["stdGender"]);
                    txtprogram.Text = Convert.ToString(dt.Rows[0]["programCode"]);
                    txtyear.Text = Convert.ToString(dt.Rows[0]["yearLvl"]);
                    txtInstitute.Text =
Convert.ToString(dt.Rows[0]["instituteCode"]);
                }
                //flxAY

                query = "proc_lackingcred";
                cmd = new MySqlCommand(query, con);
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("?vstdIDNo", txtID.Text);

                //cmd.Parameters.AddWithValue("?desc", txtDesc.Text + "%");
                DataTable dtLacking = new DataTable();
                adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dtLacking);
                adptr.Dispose();
                cmd.Dispose();

                query = "proc_submittedcred";
                cmd = new MySqlCommand(query, con);
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("?vstdIDNo", txtID.Text);
                DataTable dtSubmitted = new DataTable();
                adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dtSubmitted);
                adptr.Dispose();
                cmd.Dispose();


                flxLacking.AutoGenerateColumns = false;
                flxLacking.DataSource = dtLacking;

                flxSubmitted.AutoGenerateColumns = false;
                flxSubmitted.DataSource = dtSubmitted;

            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        private void txtsearchID_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.KeyCode == Keys.Enter)
            {
                LoadData(txtsearchID.Text);
            }
        }

        private void StudentRecords_Load(object sender, EventArgs e)
        {
            //LoadData();
            Helper.setAutoComplete(txtsearchID, "select distinct(studentIDNo) from
student order by studentIDNo asc", "studentIDNo");
            Helper.setAutoComplete(txtsearchLname, "select distinct(stdLname) from
student order by stdLname asc", "stdLname");

            LoadStudentTable();
        }

        private void txtsearchLname_KeyDown(object sender, KeyEventArgs e)
        {
            if (e.KeyCode == Keys.Enter)
            {
                // LoadData();
                LoadStudentTable();
            }
        }

        private void flxStdList_DoubleClick(object sender, EventArgs e)
        {
            LoadData(Convert.ToString(flxStdList[flxStdList.RowSel,
"studentIDNo"]));
        }

        private void printToolStripMenuItem_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtID.Text))
            {
                Box.warnBox("Please select student.");
                return;
            }

            Print_LackingCred frm = new Print_LackingCred();
            frm.stdIDNo = txtID.Text;
            frm.ShowDialog();
        }

        private void button1_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtORNo.Text))
            {
                Box.warnBox("Please input OR no.");
                return;
            }
            if (String.IsNullOrEmpty(txtID.Text))
            {
                Box.warnBox("Please input ID no.");
                return;
            }

            Print_Dismissal frm = new Print_Dismissal();
            frm.stdIDNo = txtID.Text;
            frm.orNo = txtORNo.Text;
            frm.ShowDialog();
        }
    }
}
```

StudentCredentials

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class SubmitCredential : Form
    {

        MySqlConnection con;
        MySqlCommand cmd;
        string query;


        int stdid;

        public SubmitCredential()
        {
            InitializeComponent();
        }

        private void SubmitCredential_Load(object sender, EventArgs e)
        {
            Helper.setAutoComplete(txtsearchID, "select studentIDNo from student 
order by studentIDNo asc", "studentIDNo");
            Helper.setAutoComplete(txtsearchName, "select fullname from 
vw_studentinfo order by fullname asc", "fullname");
        }

        public void LoadCredRequirement()
        {
            flxCred.Rows.Count = flxCred.Rows.Fixed;
            con = Connection.con();
            con.Open();
            query = "proc_credfile";
            cmd = new MySqlCommand(query, con);
            cmd.CommandType = CommandType.StoredProcedure;
            cmd.Parameters.AddWithValue("?vstdIDNo", txtsearchID.Text);
            DataTable dt = new DataTable();
            MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
            adptr.Fill(dt);
            adptr.Dispose();
            cmd.Dispose();
            con.Close();
            con.Dispose();
           // Box.infoBox(dt.Rows.Count.ToString());
            //  flxCred.AutoGenerateColumns = false;
```

```
                // flxCred.DataSource = dt;
                flxCred.AllowAddNew = true;
               if(dt.Rows.Count > 0)
                {
                    for (int row = 0; row <= dt.Rows.Count - 1; row++)
                    {
                        flxCred.Rows.Add();
                        flxCred[row + 1, "cred_setupID"] =
Convert.ToInt32(dt.Rows[row]["cred_setupID"]);
                        flxCred[row + 1, "cred_name"] =
Convert.ToString(dt.Rows[row]["cred_name"]);
                        flxCred[row + 1, "credid"] =
Convert.ToString(dt.Rows[row]["credid"]);
                        flxCred[row + 1, "isCheck"] =
Convert.ToBoolean(dt.Rows[row]["isCheck"]);
                        flxCred[row + 1, "nStatus"] =
Convert.ToString(dt.Rows[row]["nStatus"]);
                        stdid = Convert.ToInt32(dt.Rows[row]["stdSysID"]);


                    }

                    txtsearchName.Text = Convert.ToString(dt.Rows[0]["fullname"]);
                    txtlname.Text = Convert.ToString(dt.Rows[0]["stdLname"]);
                    txtfname.Text = Convert.ToString(dt.Rows[0]["stdFname"]);
                    txtmname.Text = Convert.ToString(dt.Rows[0]["stdMname"]);
                    txtgender.Text = Convert.ToString(dt.Rows[0]["stdGender"]);
                    txtinstitute.Text = Convert.ToString(dt.Rows[0]["instituteCode"]);
                    txtprogram.Text = Convert.ToString(dt.Rows[0]["programCode"]);
                    txtmajor.Text = Convert.ToString(dt.Rows[0]["major"]);
                    txtyearlvl.Text = Convert.ToString(dt.Rows[0]["yearLvl"]);
                }
                flxCred.AllowAddNew = false;
            }

            private void btnSearch_Click(object sender, EventArgs e)
            {
                if (String.IsNullOrEmpty(txtsearchID.Text))
                {
                    Box.warnBox("Please input student id no.");
                    return;
                }

                try
                {
                    LoadCredRequirement();
                }
                catch (Exception er)
                {
                    Box.errBox(er.Message);
                   // throw;
                }


            }

            private void txtsearchID_KeyDown(object sender, KeyEventArgs e)
            {
```

```
            if(e.KeyCode == Keys.Enter)
            {
                btnSearch_Click(sender, e);
            }
        }

        private void btnScan_Click(object sender, EventArgs e)
        {
            if(flxCred.Rows.Count > 1)
            {
                string stat = Convert.ToString(flxCred[flxCred.RowSel,
"nStatus"]);

                if (stat.ToLower() == "complete")
                {
                    ScanUploader frm = new ScanUploader(this);

                    int cid = Convert.ToInt32(flxCred[flxCred.RowSel, "credid"]);
                    frm.isUpdate = true;
                    frm.credid = cid;
                    frm.formName = Convert.ToString(flxCred[flxCred.RowSel,
"cred_name"]);
                    frm.studentid = stdid;
                    frm.credsetupid = Convert.ToInt32(flxCred[flxCred.RowSel,
"cred_setupID"]);
                    frm.ShowDialog();
                }
                else
                {
                    ScanUploader frm = new ScanUploader(this);

                    int cid = Convert.ToInt32(flxCred[flxCred.RowSel, "credid"]);

                    con = Connection.con();
                    con.Open();
                    query = "select * from credential where credid=?credid";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?credid", cid);
                    MySqlDataReader dr;
                    dr = cmd.ExecuteReader();
                    bool flag = dr.Read();
                    dr.Close();
                    cmd.Dispose();
                    con.Close();
                    con.Dispose();

                    if (flag)
                    {
                        frm.isUpdate = true;
                        frm.credid = cid;
                    }
                    else
                    {
                        frm.isUpdate = false;
                        frm.credid = 0;
                    }
                    frm.formName = Convert.ToString(flxCred[flxCred.RowSel, 
"cred_name"]);
                    frm.studentid = stdid;
                    frm.credsetupid = 
Convert.ToInt32(flxCred[flxCred.RowSel,"cred_setupID"]);
                    frm.ShowDialog();
                }
            }
            else
            {
                Box.warnBox("Please search student first.");
                return;
            }


        }

        private void btnSave_Click(object sender, EventArgs e)
        {

            con = Connection.con();
            con.Open();


            for (int row = 1; row <= flxCred.Rows.Count-1; row++)
            {

                int credid = Convert.ToInt32(flxCred[row, "credid"]);
                int setupid = Convert.ToInt32(flxCred[row, "cred_setupID"]);

                query = "select * from credential where credid=?credid";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?credid", credid);
                MySqlDataReader dr;
                dr = cmd.ExecuteReader();
                bool flag = dr.Read();
                dr.Close();
                cmd.Dispose();

                if (flag)
                {
                    int check = Convert.ToInt32(flxCred[row, "isCheck"]);
                    query = "UPDATE credential SET isCheck=?check WHERE credid = 
?id";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?check", check);
                    cmd.Parameters.AddWithValue("?id", credid);
                    cmd.ExecuteNonQuery();
                    cmd.Dispose();
                }
                else
                {

                    int check = Convert.ToInt32(flxCred[row, "isCheck"]);
                    query = "INSERT INTO credential SET cred_setupID=?setupid, 
stdSysID=?sysstdid, isCheck=?check";
                    cmd = new MySqlCommand(query, con);
                    cmd.Parameters.AddWithValue("?setupid", setupid);
                    cmd.Parameters.AddWithValue("?sysstdid", stdid);
                    cmd.Parameters.AddWithValue("?check", check);
                    cmd.ExecuteNonQuery();
                    cmd.Dispose();
                }
            }

            con.Close();
            con.Dispose();

            Box.infoBox("Your Checklist successfully save.");
            //Helper.RecordLogs("CHECKLIST SAVED", "Checklist was successfully
saved.");
            LoadCredRequirement();
        }

        private void btnCredential_Click(object sender, EventArgs e)
        {
            if(flxCred.Rows.Count > 1)
            {
                string nstatus = Convert.ToString(flxCred[flxCred.RowSel,
"nStatus"]);
                if(nstatus.ToLower() == "incomplete")
                {
                    Box.warnBox("No file found.");
                    return;
                }

                ViewCredential frm = new ViewCredential();
                frm.credid = Convert.ToInt32(flxCred[flxCred.RowSel, "credid"]);
                frm.ShowDialog();
            }
        }


    }
}
```

UserAccount

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class UserAccount : Form
    {


        MySqlConnection con;
        MySqlCommand cmd;
        string query;

        int id = 0;

        public UserAccount()
        {
            InitializeComponent();
        }

        public void LoadData()
        {
            try
            {

                con = Connection.con();
                con.Open();
                query = "select * from users where username like ?key or fullname 
like ?key";
                cmd = new MySqlCommand(query, con);

                cmd.Parameters.AddWithValue("?key", txtsearch.Text + "%");

                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();

                this.flxAccount.AutoGenerateColumns = false;
                this.flxAccount.DataSource = dt;
                //flxAY
            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //throw;
            }
        }

        void clear()
        {
            txtuser.Text = "";
            txtpwd.Text = "";
            id = 0;
            cmbGender.SelectedIndex = -1;
            cmbUserType.SelectedIndex = -1;
            txtfullname.Text = "";
            txtuser.Focus();
        }

        private void UserAccount_Load(object sender, EventArgs e)
        {
            Helper.setAutoComplete(txtsearch, "select distinct(fullname) from 
users order by fullname asc", "fullname");
            LoadData();

        }


        void getData()
        {
            con = Connection.con();
            con.Open();
            query = "select * from users where usersid=?id";
            cmd = new MySqlCommand(query, con);
            cmd.Parameters.AddWithValue("?id", 
Convert.ToInt32(flxAccount[flxAccount.RowSel,"usersid"]));
            DataTable dt = new DataTable();
            MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
            adptr.Fill(dt);
            adptr.Dispose();
            cmd.Dispose();
            con.Close();
            con.Dispose();

            if (dt.Rows.Count > 0)
            {
                txtuser.Text = Convert.ToString(dt.Rows[0]["username"]);
                txtfullname.Text = Convert.ToString(dt.Rows[0]["fullname"]);
                cmbGender.Text = Convert.ToString(dt.Rows[0]["gender"]);
                cmbUserType.Text = Convert.ToString(dt.Rows[0]["position"]);
                id = Convert.ToInt32(dt.Rows[0]["usersid"]);
            }
        }


        private void btnRefresh_Click(object sender, EventArgs e)
        {
            LoadData();
            clear();
        }

        private void flxAccount_DoubleClick(object sender, EventArgs e)
        {
            if(flxAccount.Rows.Count > 1)
            {
                getData();
            }
        }

        private void btnDelete_Click(object sender, EventArgs e)
        {
            if(flxAccount.Rows.Count > 1)
            {
                byte isDefault = Convert.ToByte(flxAccount[flxAccount.RowSel, 
"isDefault"]);
               // string usertype = Convert.ToString(flxAccount[flxAccount.RowSel, 
"position"]);
                if(isDefault > 0)
                {
                    Box.warnBox("Account cannot be deleted.");
                    return;
                }

                if (Box.questionBox("Are you sure you want to delete this row?", 
"DELETE?"))
                {
                    int id = Convert.ToInt32(flxAccount[flxAccount.RowSel, 1]);
                    Helper.deleteByInt("users", "usersid", id);
                    Box.infoBox("Deleted successfully.");
                    LoadData();
                }
            }

        }

        private void btnSave_Click(object sender, EventArgs e)
        {
            if (String.IsNullOrEmpty(txtuser.Text))
            {
                Box.warnBox("Please input username.");
                return;
            }
            if (String.IsNullOrEmpty(txtpwd.Text))
            {
                Box.warnBox("Please input password.");
                return;
            }
            if (String.IsNullOrEmpty(cmbUserType.Text))
            {
                Box.warnBox("Please select user type.");
                return;
            }
            if (String.IsNullOrEmpty(cmbUserType.Text))
            {
                Box.warnBox("Please select user type.");
                return;
            }


            processSave();
            clear();
            LoadData();


        }



        void processSave()
        {
            if (id > 0)
            {
                con = Connection.con();
                con.Open();
                query = @"UPDATE users SET username=?uname,  pwd=?pwd, 
fullname=?fullname, gender=?gender, position=?pos
```

```
                        WHERE usersid = ?id";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?uname", txtuser.Text);
                cmd.Parameters.AddWithValue("?pwd", txtpwd.Text);
                cmd.Parameters.AddWithValue("?fullname", txtfullname.Text);
                cmd.Parameters.AddWithValue("?gender", cmbGender.Text);
                cmd.Parameters.AddWithValue("?pos", cmbUserType.Text);
                cmd.Parameters.AddWithValue("?id", id);
                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                if (n > 0)
                {
                    Box.infoBox("Account successfully updated.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }


            }
            else
            {
                con = Connection.con();
                con.Open();
                query = @"INSERT INTO users SET username=?uname,  pwd=?pwd,
fullname=?fullname, gender=?gender, position=?pos";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?uname", txtuser.Text);
                cmd.Parameters.AddWithValue("?pwd", txtpwd.Text);
                cmd.Parameters.AddWithValue("?fullname", txtfullname.Text);
                cmd.Parameters.AddWithValue("?gender", cmbGender.Text);
                cmd.Parameters.AddWithValue("?pos", cmbUserType.Text);

                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();

               // id = Helper.returnLastInsertID(con);
                con.Close();
                con.Dispose();

                // Box.infoBox("Institute successfully updated.");

                if (n > 0)
                {
                    Box.infoBox("Account successfully saved!.");
                }
                else
                {
                    Box.errBox("An error occured. Please contact administrator.");
                }

            }
        }
        private void txtsearch_KeyDown(object sender, KeyEventArgs e)
        {
            if(e.KeyCode == Keys.Enter)
            {
                LoadData();
            }
        }

        private void btnDefault_Click(object sender, EventArgs e)
        {
            if(flxAccount.Rows.Count > 1)
            {
                //  byte isDefault = Convert.ToByte(flxAccount[flxAccount.RowSel,
"isDefault"]);
                int userid = Convert.ToInt32(flxAccount[flxAccount.RowSel,
"usersid"]);
                con = Connection.con();
                con.Open();
                query = "UPDATE users SET isDefault=0; UPDATE users SET
isDefault=1 WHERE usersid=?userid";
                cmd = new MySqlCommand(query, con);
                cmd.Parameters.AddWithValue("?userid", userid);

                int n = cmd.ExecuteNonQuery();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Box.infoBox("Account set to default.");
                LoadData();

            }
            else
            {
                Box.warnBox("No data selected.");

            }

        }
    }
}
```

ViewCredentials

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;
```

```
namespace ArchivingSystem
{
    public partial class ViewCredential : Form
    {
        MySqlConnection con;
        MySqlCommand cmd;
        string query;

        public int credid;


        public ViewCredential()
        {
            InitializeComponent();
        }

        void LoadReport()
        {

            try
            {
                //int gid = Helper.returnGroupID(cmbGroup);

                con = Connection.con();
                con.Open();
                query = "proc_viewcred";
                cmd = new MySqlCommand(query, con);
                cmd.CommandType = CommandType.StoredProcedure;
                cmd.Parameters.AddWithValue("?vcredid", credid);

                DataTable dt = new DataTable();
                MySqlDataAdapter adptr = new MySqlDataAdapter(cmd);
                adptr.Fill(dt);
                adptr.Dispose();
                cmd.Dispose();
                con.Close();
                con.Dispose();

                Report_ViewCred rpt = new Report_ViewCred();
                rpt.SetDataSource(dt);
                crystalReportViewer1.ReportSource = rpt;
                crystalReportViewer1.Refresh();



            }
            catch (Exception er)
            {
                Box.errBox(er.Message);
                //   throw;
            }
        }

        private void ViewCredential_Load(object sender, EventArgs e)
        {
            LoadReport();
        }
    }
}
```

Backup

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.Drawing;
using System.Linq;
using System.Text;
using System.Threading.Tasks;
using System.Windows.Forms;
using MySql.Data.MySqlClient;

namespace ArchivingSystem
{
    public partial class BackupRestore : Form
    {
        MySqlConnection con;

        public BackupRestore()
        {
            InitializeComponent();
        }

        private void BackupRestore_Load(object sender, EventArgs e)
        {
            txtOutput.Text = Properties.Settings.Default.outputDir;
        }

        private void btnOutputDir_Click(object sender, EventArgs e)
        {
            saveFileDialog1.InitialDirectory = @"C:\";
            saveFileDialog1.Title = "Browse Directory";
            DialogResult result = saveFileDialog1.ShowDialog();

            if (result == DialogResult.OK)
            {
                string filename = saveFileDialog1.FileName;
                using (MySqlCommand cmd = new MySqlCommand())
                {

                    Box.infoBox(filename.ToString());
                    using (MySqlBackup mb = new MySqlBackup(cmd))
                    {
                        con = Connection.con();
                        cmd.Connection = con;
                        con.Open();
                        mb.ExportToFile(filename);
                        con.Close();
                        Box.infoBox("Backup successfully.");
                    }
                }
            }
        }


    }
}
```

BIBLIOGRAPHY

Svard (2014) - Information and records management systems and the impact of information culture on the management of public information - *http://sh.diva-portal.org/smash/record.jsf?pid=diva2%3A1121603&dswid=-3681*

Kipp Bently 2018 - How Can Schools Successfully Digitize Student Records? - *http://www.govtech.com/education/k-12/How-Can-Schools-Successfully-Digitize-Student-Records.html*

Prince George (2012) - Records Management - *https://www.unbc.ca/records-management/why-records-management-important*

Scott Kimura (2015) - 4 Advantages of Digital Student Records - *https://blog.mesltd.ca/4-advantages-of-digital-student-records*

Hwang et al (2013) and Yaldir and Polat (2016) - Design and Implementation of an Electronic Document Management System - *http://dergipark.gov.tr/download/article-file/358089*

CURRICULUM VITAE

Name	:	Princess J. Subiza
Age	:	20
Date of Birth	:	March 01, 1999
Place of Birth	:	Pagadian City
Address	:	Pob. Sominot Zamboanga Del Sur
Sex	:	Female
Civil Status	:	Single
Religion	:	Roman Catholic
Father's Name	:	Eduardo R. Subiza
Mother's Name	:	Estrella J. Subiza

Educational Background:

Tertiary	:	Bachelor of Science in Computer Science
	:	Gov. Alfonso D. Tan College
		Maloro, Tangub City
		2015 – 2019
Secondary	:	Sominot National High School
	:	Poblacion, Sominot, Zamboanga Del Sur
		2011 - 2015
Elementary	:	Sominot Central Elementary School
	:	Poblacion, Sominot, Zamboanga Del Sur
		2005 - 2011

Seminar/Workshop Attended

2017	:	Seminar –Workshop on Cybercrime Prevention Act
		November 15, 2017
		GADTC
2018	:	Seminar –Workshop on Advanced Website Development
		July 4, 2018
		GADTC

2019	:	Seminar on Digital Signature January 4, 2019 Function Hall 2
2019	:	Seminar on Computer Hardware and Troubleshooting January 4, 2019 Function Hall 2

Organizations/Affiliations

2016 – 2017	:	Secretary Institute of Computer Studies(ICS) Gov. Alfonso D. Tan College Maloro, Tangub City
2016 – 2017	:	Secretary JPEG.Com Gov. Alfonso D. Tan College Maloro, Tangub City
2017 – 2018	:	PRO JPEG.Com Gov. Alfonso D. Tan College Maloro, Tangub City
2018 – 2019	:	Socio Cultural Officer Institute of Computer Studies(ICS) Gov. Alfonso D. Tan College Maloro, Tangub City

Name	:	Alfredo D. Lotarion Jr.
Age	:	23
Date of Birth	:	January 22, 1996
Place of Birth	:	Pob. Sominot Zambozanga Del Sur
Address	:	Pob. Sominot Zambozanga Del Sur
Sex	:	Male
Civil Status	:	Single
Religion	:	Roman Catholic
Father's Name	:	Alfredo L. Lotarion Sr.
Mother's Name	:	Leonila M. Lotarion

Educational Background:

Tertiary	:	Bachelor of Science in Computer Science
	:	Gov. Alfonso D. Tan College
		Maloro, Tangub City
		2015 – 2019
Secondary	:	Sominot National High School
	:	Poblacion, Sominot, Zamboanga Del Sur
		2010 - 2014
Elementary	:	Sominot Central Elementary School
	:	Poblacion, Sominot, Zamboanga Del Sur
		2004 - 2010

Seminar/Workshop Attended

2017	:	Seminar –Workshop on Cybercrime Prevention Act
		November 15, 2017
		GADTC
2018	:	Seminar –Workshop on Advanced Website Development
		July 4, 2018
		GADTC

2019	:	Seminar on Digital Signature January 4, 2019 Function Hall 2
2019	:	Seminar on Computer Hardware and Troubleshooting January 4, 2019 Function Hall 2

Organizations/Affiliations

2016 – 2017	:	Socio Cultural Officer Institute of Computer Studies(ICS) Gov. Alfonso D. Tan College Maloro, Tangub City
2017 – 2018	:	Sports Development Officer Institute of Computer Studies(ICS) Gov. Alfonso D. Tan College Maloro, Tangub City
2017 – 2018	:	Social Officer JPEG.Com Gov. Alfonso D. Tan College Maloro, Tangub City
2018 – 2019	:	Sports Development Officer Institute of Computer Studies(ICS) Gov. Alfonso D. Tan College Maloro, Tangub City

Name	:	Richard A. Lanterna
Age	:	28
Date of Birth	:	September 19, 1990
Place of Birth	:	Tambulig, Zamboanga Del Sur
Address	:	Tambulig, Zamboanga Del Sur
Sex	:	Male
Civil Status	:	Single
Religion	:	Roman Catholic
Father's Name	:	Danilo Lanterna
Mother's Name	:	Fausta Anora Lanterna

Educational Background:

Tertiary	:	Bachelor of Science in Computer Science
	:	Gov. Alfonso D. Tan College
		Maloro, Tangub City
		2015 – 2019
Secondary	:	Tambulig National High School
	:	Tambulig, Zamboanga del Sur
		2011 - 2015
Elementary	:	Tambulig Elementary School
	:	Tambulig, Zamboanga del Sur
		2005 - 2011

Seminar/Workshop Attended

2017	:	Seminar –Workshop on Cybercrime Prevention Act
		November 15, 2017
		GADTC
2018	:	Seminar –Workshop on Advanced Website Development
		July 4, 2018
		GADTC

2019	:	Seminar on Digital Signature January 4, 2019 Function Hall 2
2019	:	Seminar on Computer Hardware and Troubleshooting January 4, 2019 Function Hall 2

Organizations/Affiliations

2015 – 2019	:	Member JPEG.Com Gov. Alfonso D. Tan College Maloro, Tangub City
2015 – 2019	:	Member Yes Organization Gov. Alfonso D. Tan College Maloro, Tangub City

Name	:	Arjhon M. Fernandez
Age	:	21
Date of Birth	:	November 02, 1997
Place of Birth	:	Simasay, Tangub City, Misamis Occidental
Address	:	Simasay, Tangub City, Misamis Occidental
Sex	:	Male
Civil Status	:	Single
Religion	:	Roman Catholic
Father's Name	:	Agapeto R. Fernandez
Mother's Name	:	Alfreda M. Fernandez

Educational Background:

Tertiary	:	Bachelor of Science in Computer Science
	:	Gov. Alfonso D. Tan College
		Maloro, Tangub City
		2015 – 2019
Secondary	:	Simasay National High School
	:	Simasay, Tangub City
		2011 - 2015
Elementary	:	Guemabot Elementary School
	:	Guemabot, Tangub City
		2005 - 2011

Seminar/Workshop Attended

2017	:	Seminar –Workshop on Cybercrime Prevention Act
		November 15, 2017
		GADTC
2018	:	Seminar –Workshop on Advanced Website Development
		July 4, 2018

		GADTC
2019	:	Seminar on Digital Signature January 4, 2019 Function Hall 2
2019	:	Seminar on Computer Hardware and Troubleshooting January 4, 2019 Function Hall 2

Organizations/Affiliations

2015 – 2019	:	Member JPEG.Com Gov. Alfonso D. Tan College Maloro, Tangub City
2015 – 2019	:	Member Yes Organization Gov. Alfonso D. Tan College Maloro, Tangub City