Object Oriented Development With Java

Individual Assignment

Student Name	:	SHAMEE MRIMMOAEE AHMED
TP Number	:	TP040804
Module Code	:	CT038-3-2-OODJ
Module Name	:	OBJECT- ORIENTED DEVELOPMENT WITH JAVA
Lecturer's Name	:	Mr. TANVEER KHALEEL SHAIKH
Intake Code	:	UC2F1704SE
Hand-In Date	:	1st JUNE 2017
Hand-Out Date	:	07th AUGUST 2017

Table of Contents

1.0 Introduction

This project is about the Attendance Management System which requires Java Object Oriented Programming language and includes text file where all the data will store. This software is making for the XYZ university to take their daily attendance of the student. There are three types of users one is Admin, who control most of the function of this software and lecture can take the attendance as well as modify attendance and view the intake wise student that assigned to the lecturer. The third user is the student, they have the less function to use in this system. Based on the user this software is Graphical User Interface (GUI) so it is easier to use and understand. Java programming language is choosing for coding this software because it can easily move from one computer to another and have the ability to run the same program on many different systems. For developing this software NetBeans platform is required because it is easier to develop a GUI application in the NetBeans IDE where it is easier to drag and drop the button, text field and the other that necessary for making GUI instead of writing code.

2.0 Objectives

Attendance Management System software is developed for the XYZ University to maintain their student daily attendance. This software will be handled by three users. Different users have their own username and password to login into the system. The function of each user and the type of user who will use this system is provided below are:

Admin:

1. Admin will provide the username and password for the new user.
2. Admin can view the all the record of each student and lecturer.
3. Admin can modify the record of each student and lecturer if necessary.
4. Admin can delete the record of each student and lecture if the record is not necessary.
5. Admin can search for the specific user or lecturer by using their ID/TP Number or Name.
6. Admin can view the absent reason that sends by the student and based on reason admin can modify the student attendance by absent with reason.

Lecturer:

1. The lecturer can mark the daily attendance of the student based on the intake wise code, module name, date, start time and end time.
2. The lecturer can also modify the attendance but only with the present, absent or late.
3. The lecturer can view the record of the student that assigned to them.

Student:

1. A student can view their attendance of each module.
2. A student can send the reason for absent to the admin so their attendance record absent will change to absent with reason.

3.0 System Modeling

3.1 Class Diagram

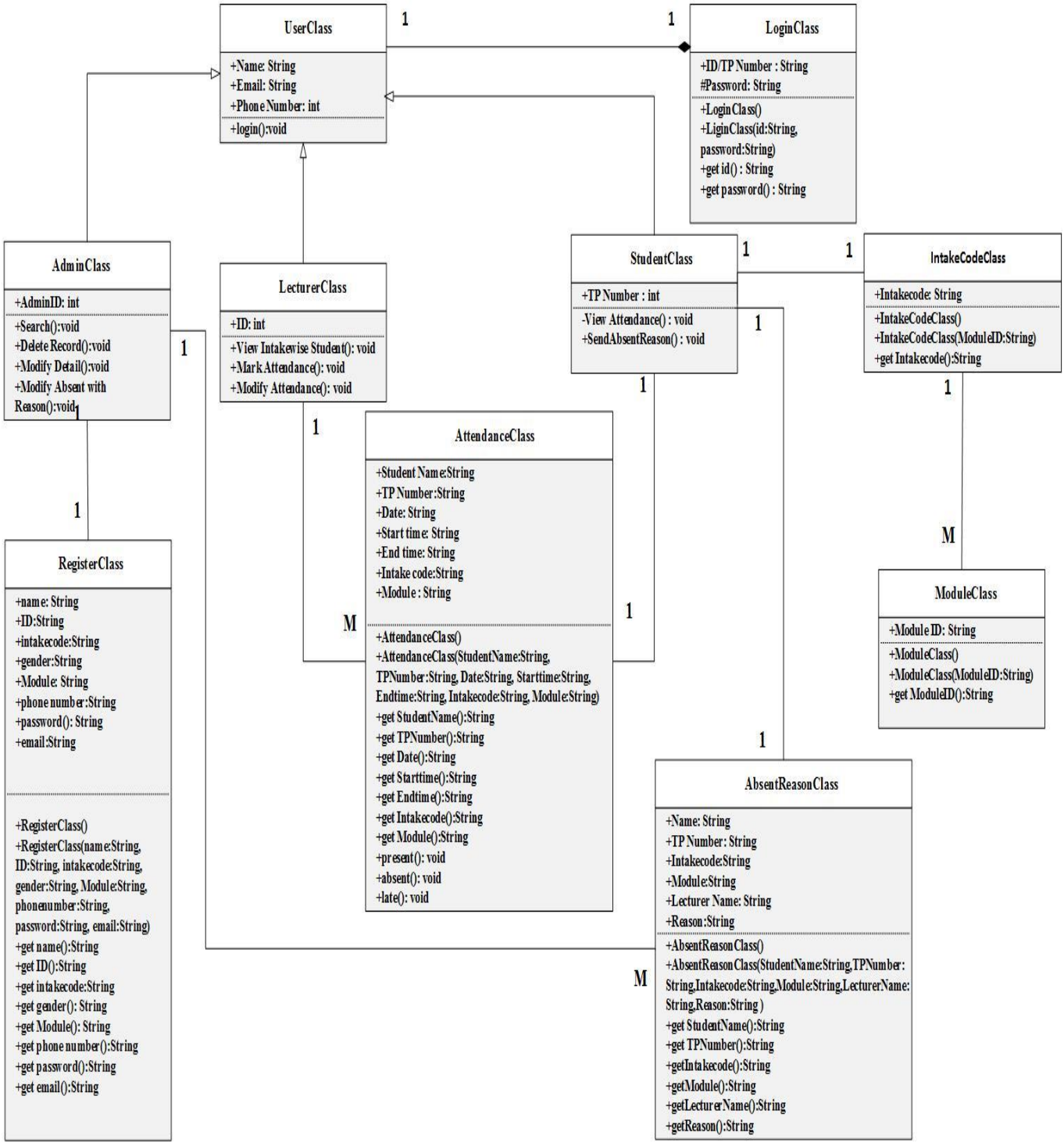

Figure 1: Class Diagram

3.2 Use case diagram

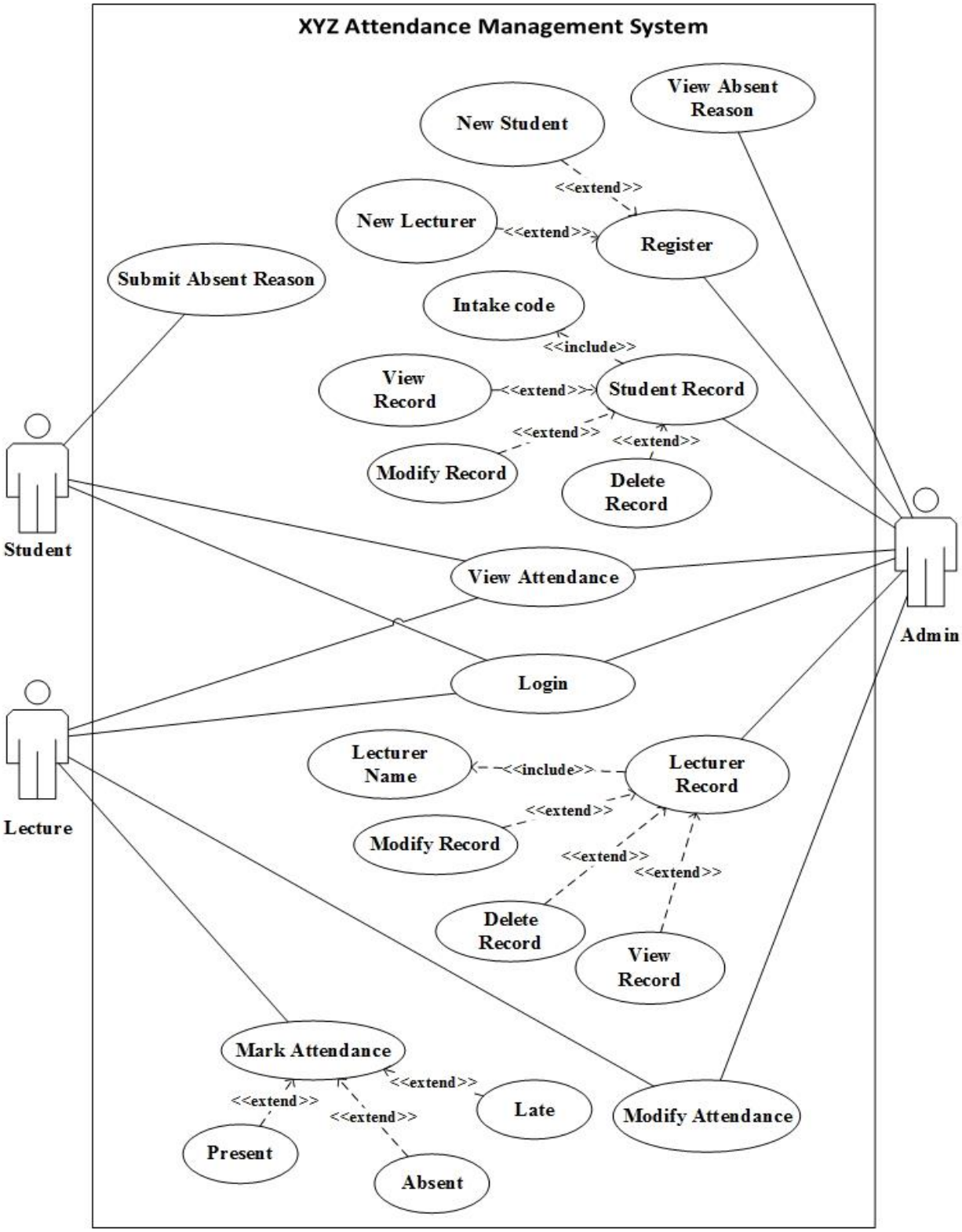

Figure 2: Use Case Diagram

3.2.1 Use case description

1.

USE-CASE DESCRIPTION: SUBMIT ABSENT REASON	
NAME	Submit Absent Reason
ID	N/A
DESCRIPTION	Student can submit reason for absent to change the attendance
ACTOR(S)	Student
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	N/A
TRIGGERS	N/A or Student
PRECONDITIONS	Unable to submit absent reason
POST-CONDITIONS	Able to submit absent reason

2.

USE-CASE DESCRIPTION: VIEW ATTENDANCE	
NAME	View Attendance
ID	N/A
DESCRIPTION	A student can view their individual attendance. Lecturer and Admin can view student attendance by Intake wise code
ACTOR(S)	Student, Lecturer, Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	N/A
TRIGGERS	N/A or Student or Lecturer or Admin
PRECONDITIONS	Unable to view attendance
POST-CONDITIONS	Able to view attendance

3.

USE-CASE DESCRIPTION: LOGIN	
NAME	Login
ID	N/A
DESCRIPTION	Access to enter into the system
ACTOR(S)	Student, Lecturer, Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	Every time use it to login into the system
TRIGGERS	Admin, Lecturer, Student
PRECONDITIONS	Unable to log into the system
POST-CONDITIONS	Able to log into the system

4

USE-CASE DESCRIPTION: MODIFY ATTENDANCE	
NAME	Modify Attendance
ID	N/A
DESCRIPTION	The lecturer can modify attendance by giving present, absent and late as well as Admin can also modify by giving absent with reason only.
ACTOR(S)	Lecturer, Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	N/A
TRIGGERS	N/A or Lecturer or Admin
PRECONDITIONS	Unable to modify attendance
POST-CONDITIONS	Able to modify attendance

5

USE-CASE DESCRIPTION: MARK ATTENDANCE	
NAME	Mark attendance
ID	N/A
DESCRIPTION	Lecturer can mark student attendance
ACTOR(S)	Lecturer
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	Everyday use to take student attendance
TRIGGERS	N/A or Lecturer
PRECONDITIONS	Unable to mark attendance
POST-CONDITIONS	Able to mark attendance

6

USE-CASE DESCRIPTION: REGISTER	
NAME	Register
ID	N/A
DESCRIPTION	Admin can register new student and lecturer
ACTOR(S)	Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	Depend on when new lecturer and student join into the university
TRIGGERS	N/A or Admin
PRECONDITIONS	Unable to register new student and lecturer
POST-CONDITIONS	Able to register new student and lecturer

7

USE-CASE DESCRIPTION: STUDENT RECORD	
NAME	Student record
ID	N/A
DESCRIPTION	Admin can view, modify and delete student's record
ACTOR(S)	Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	N/A
TRIGGERS	N/A or Admin
PRECONDITIONS	Unable to view, modify and delete student's record
POST-CONDITIONS	Able to view, modify and delete student's record

8

USE-CASE DESCRIPTION: LECTURER RECORD	
NAME	Lecturer record
ID	N/A
DESCRIPTION	Admin can view, modify and delete teacher's detail
ACTOR(S)	Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	N/A
TRIGGERS	N/A or Admin
PRECONDITIONS	Unable to view, modify and delete lecturer's record
POST-CONDITIONS	Able to view, modify and delete lecturer's record

9

USE-CASE DESCRIPTION: VIEW ABSENT REASON	
NAME	View absent reason
ID	N/A
DESCRIPTION	Admin can view reason for student's absent
ACTOR(S)	Admin
ORGANIZATIONAL BENEFITS	N/A
FREQUENCY OF USE	N/A
TRIGGERS	N/A or Admin
PRECONDITIONS	Unable to view reason for absent
POST-CONDITIONS	Able to view reason for absent

3.4 Activity Diagram

3.4.1 Admin and System

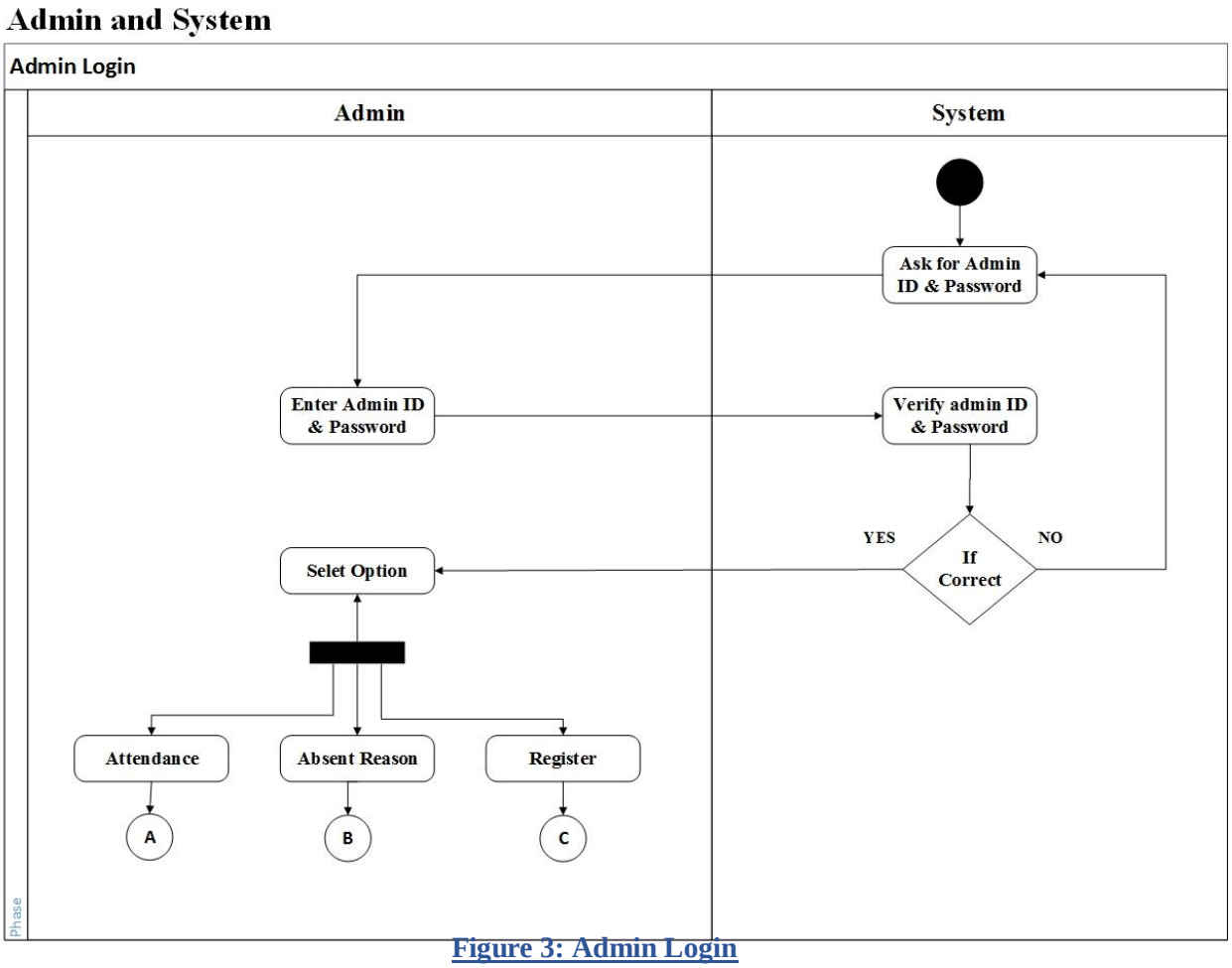

Figure 3: Admin Login

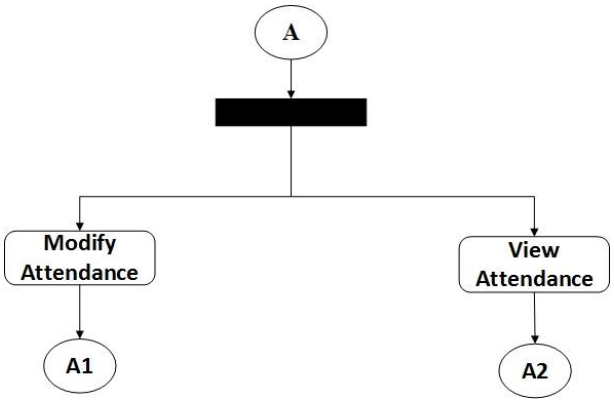

Figure 4: Attendance

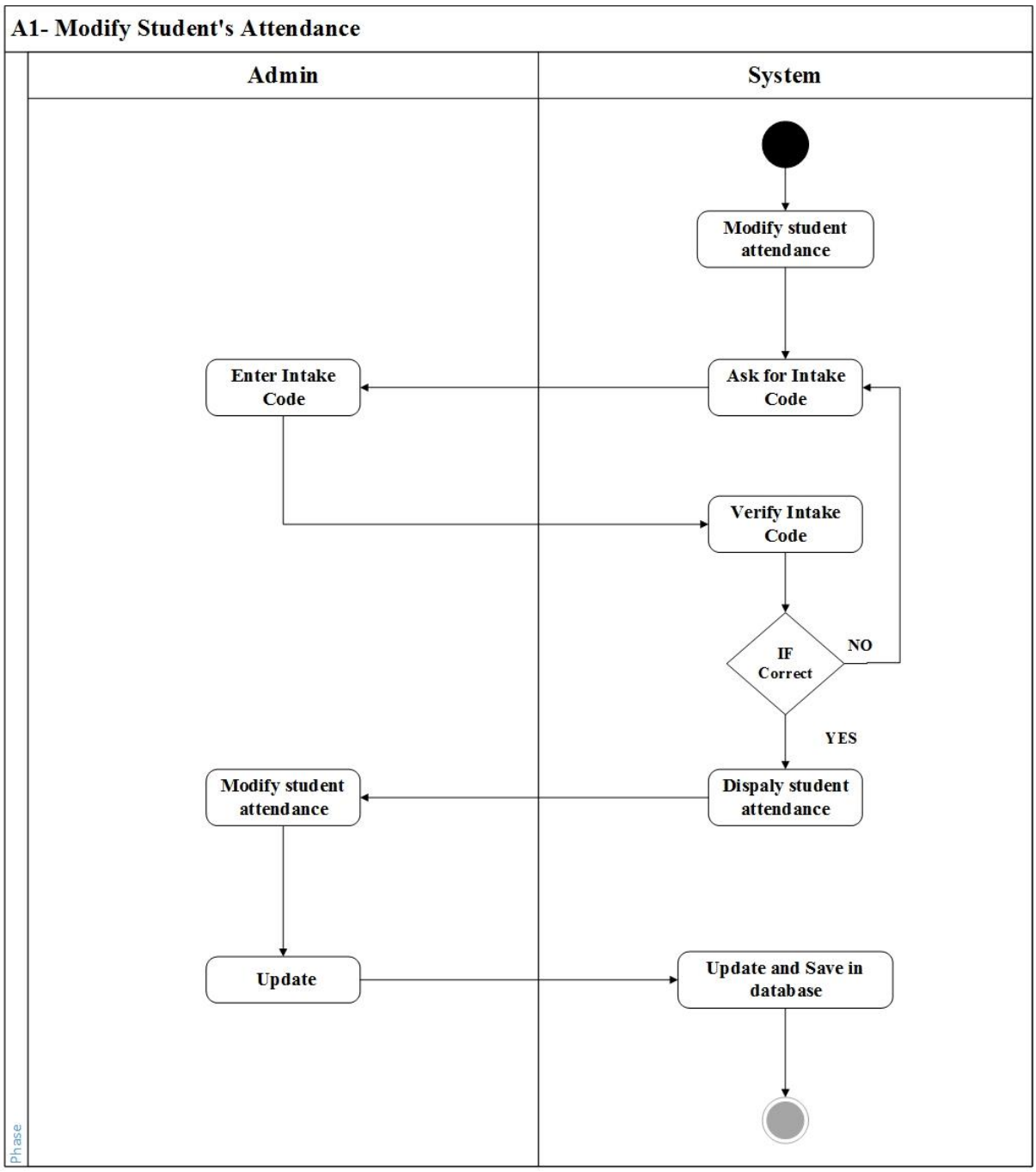

Figure 5: Modify Student's Attendance

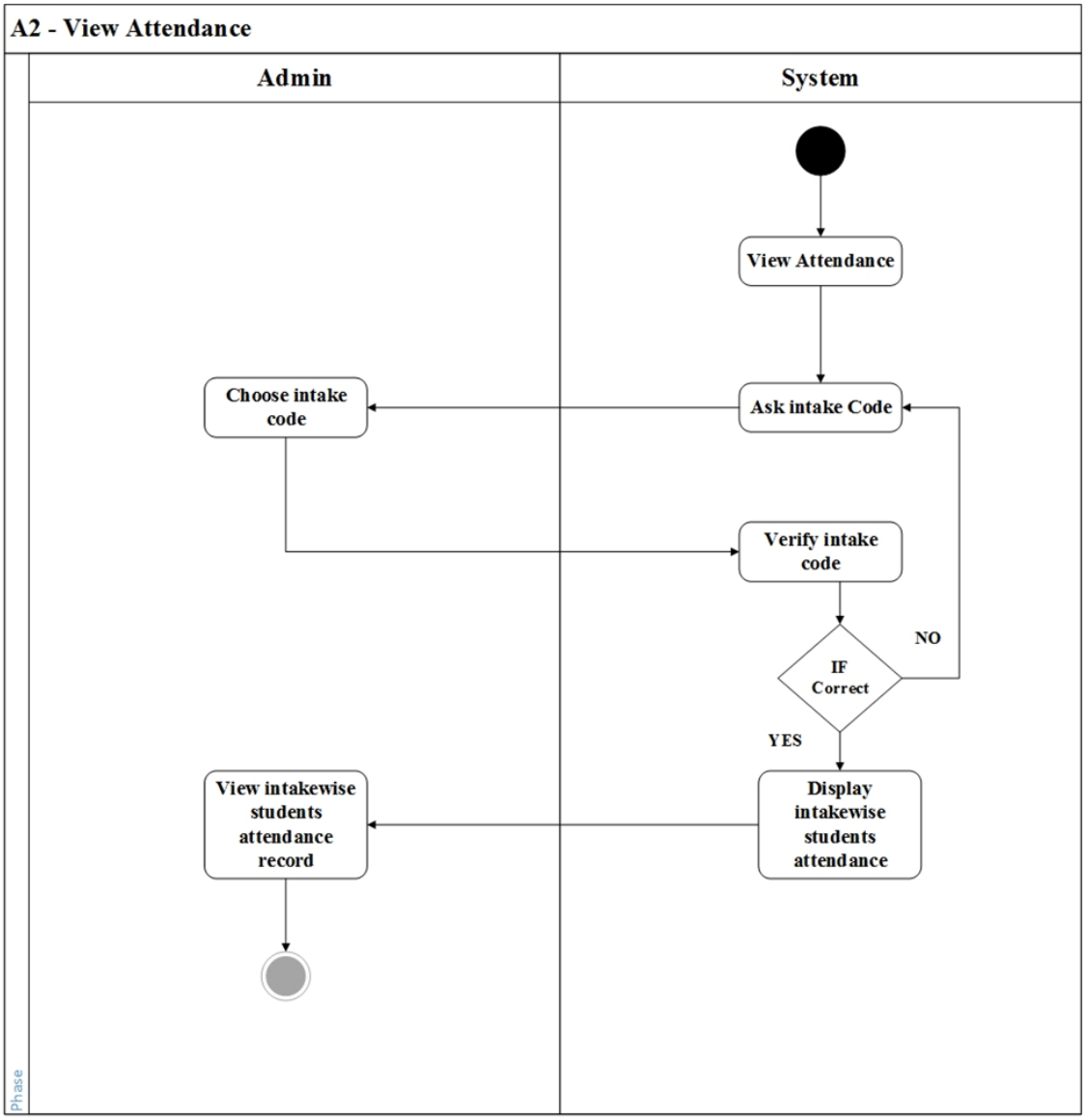

Figure 6: View Attendance

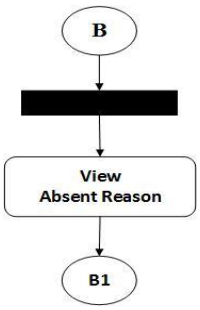

Figure 7: View Absent Reason

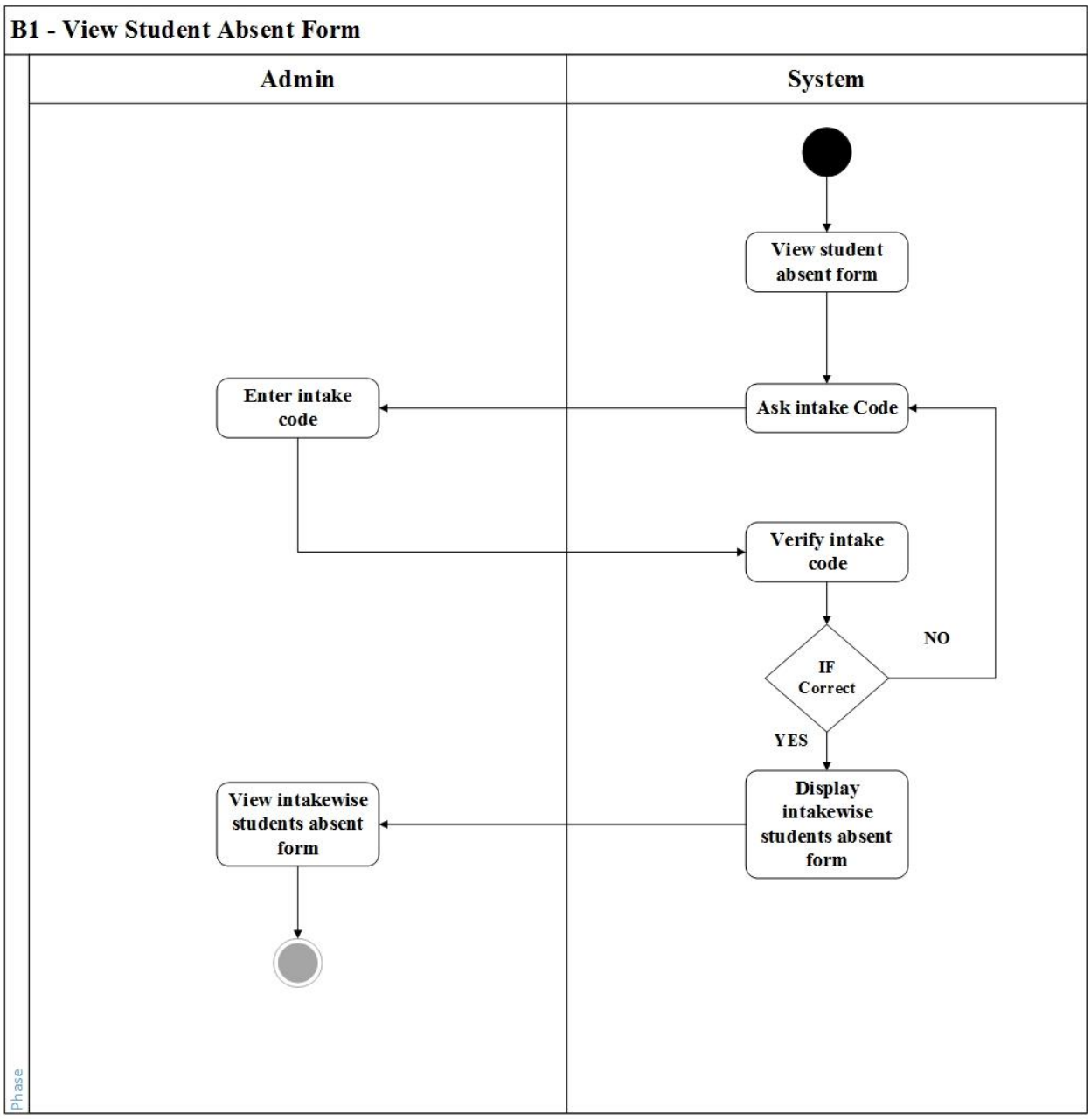

Figure 8: View Student Absent Form

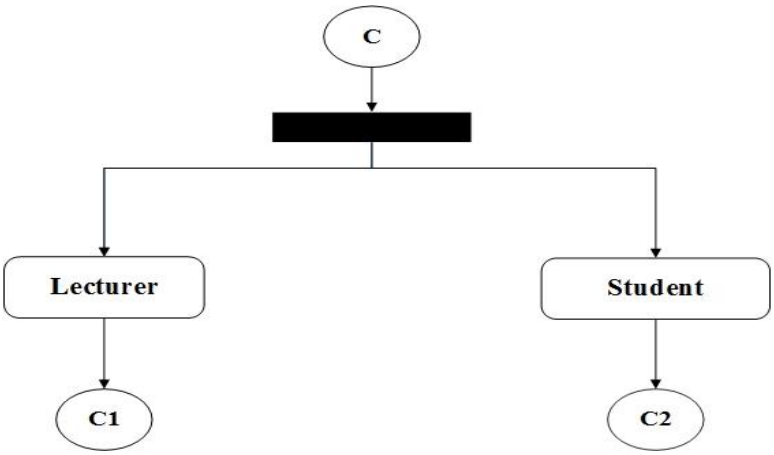

Figure 9: Register and Record

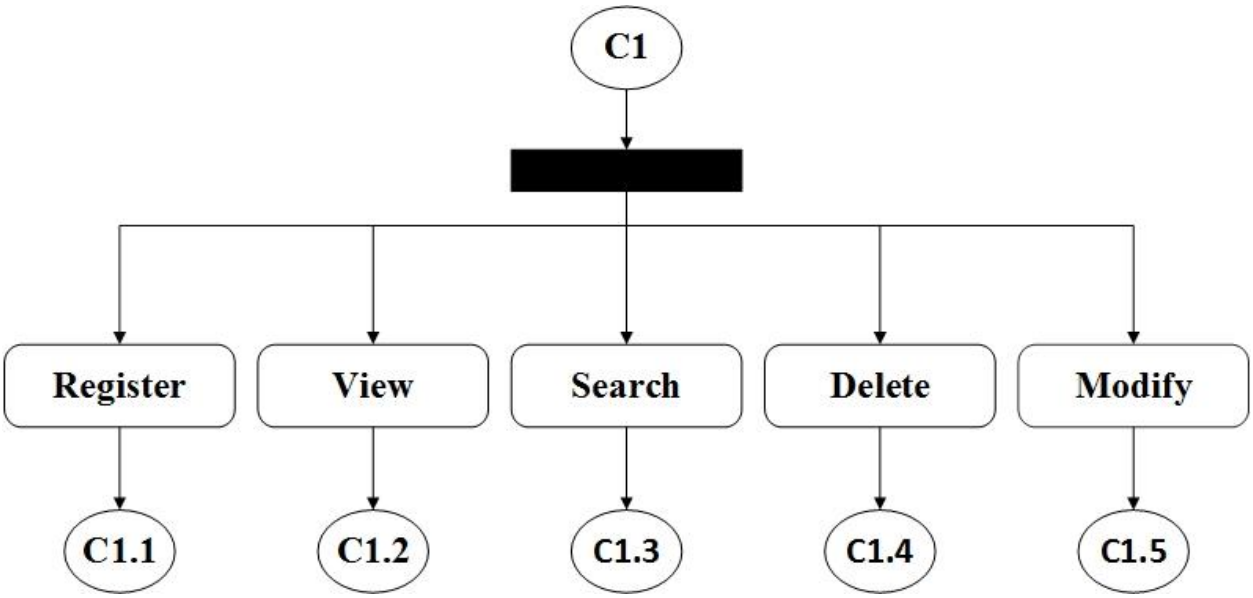

Figure 10: Lecturer

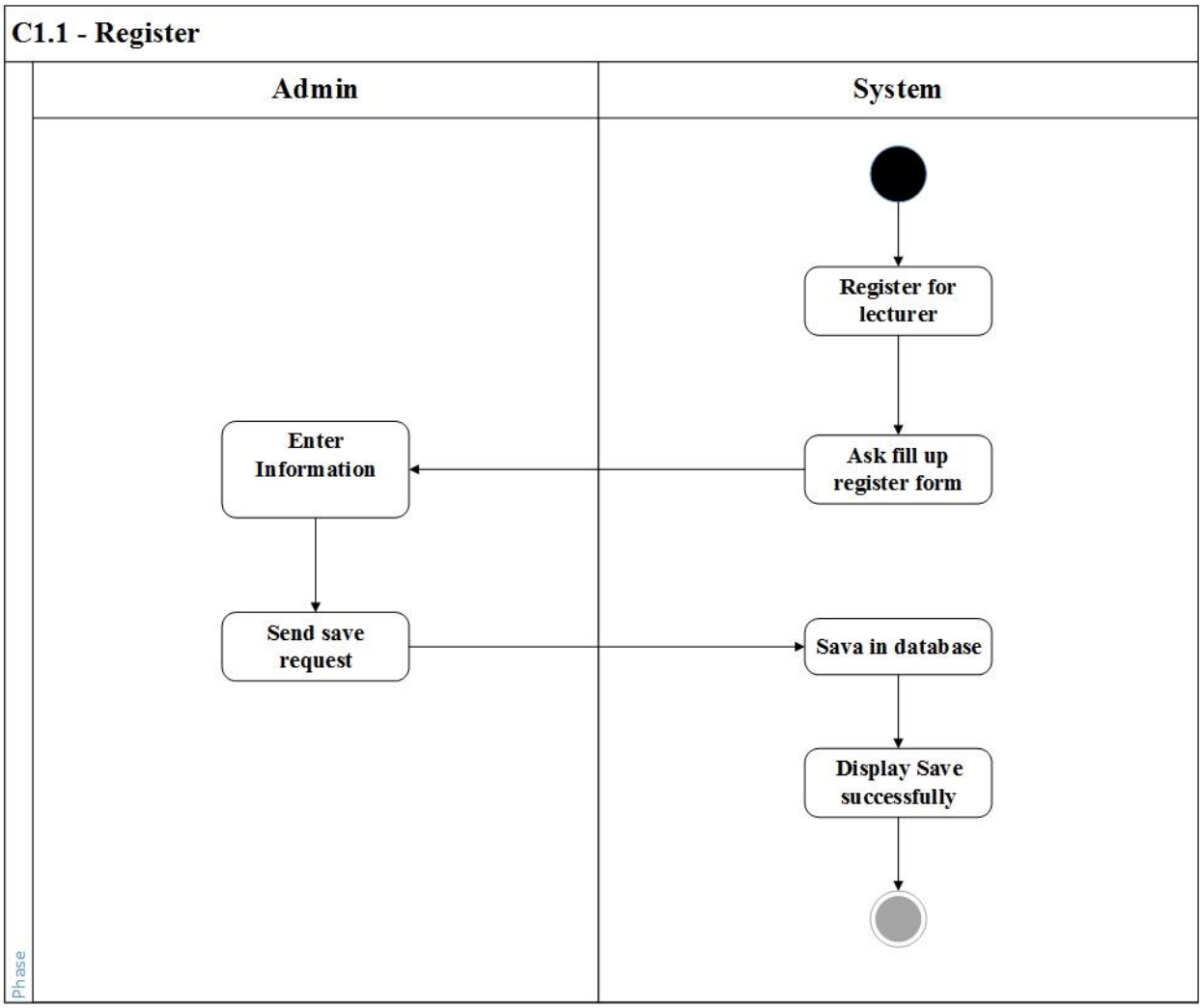

Figure 11: Register

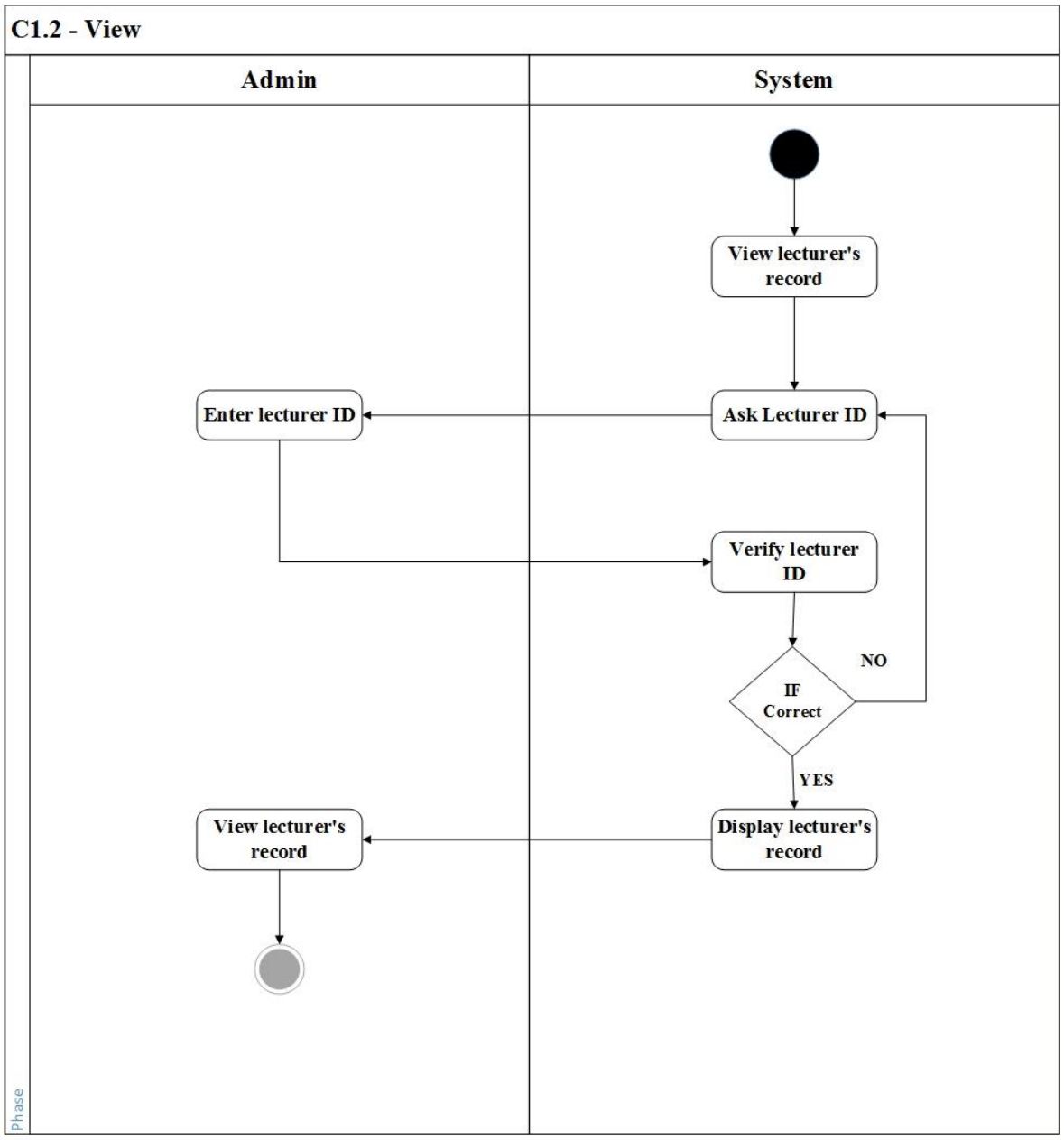

Figure 12: View

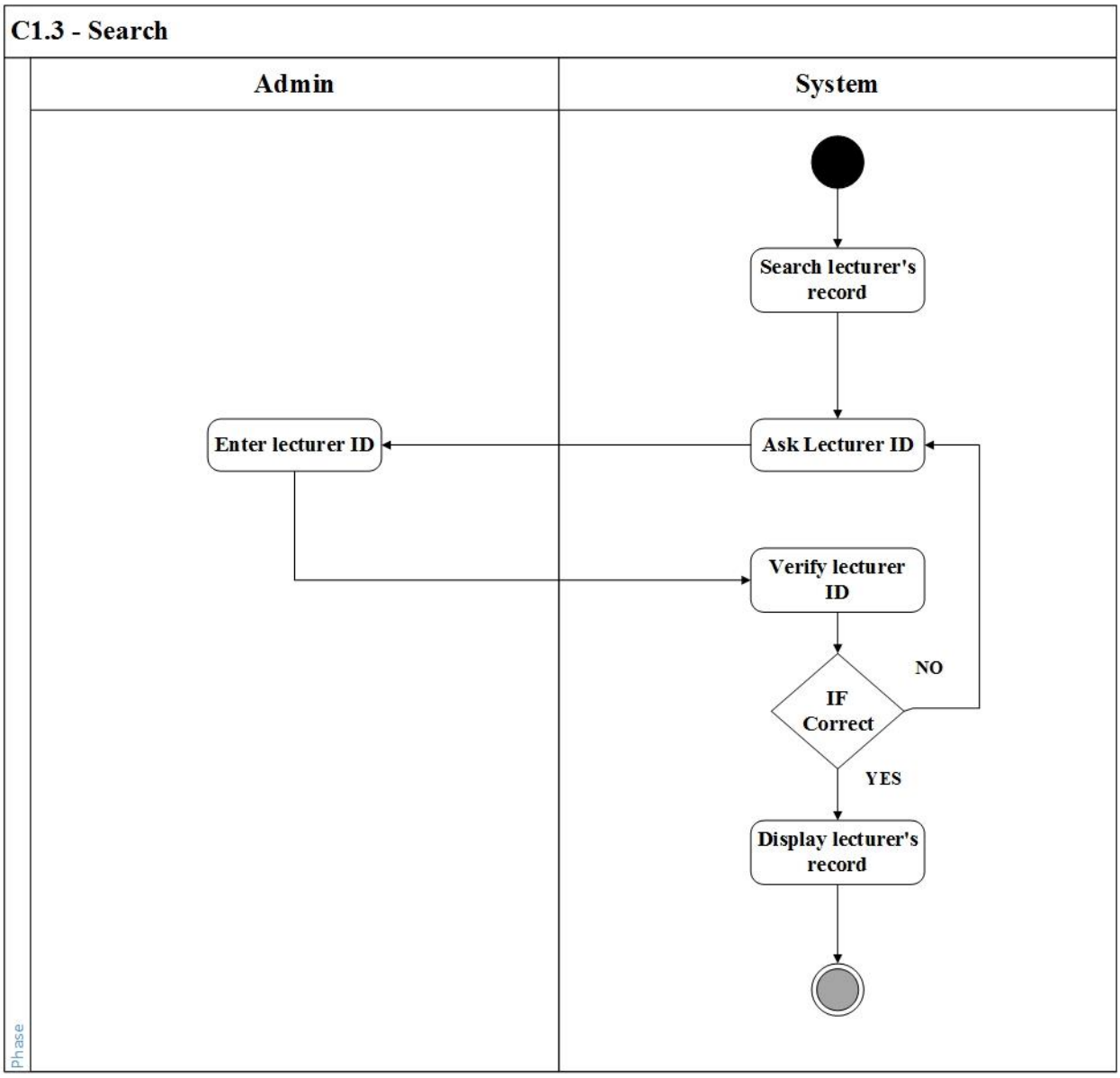

Figure 13: Search

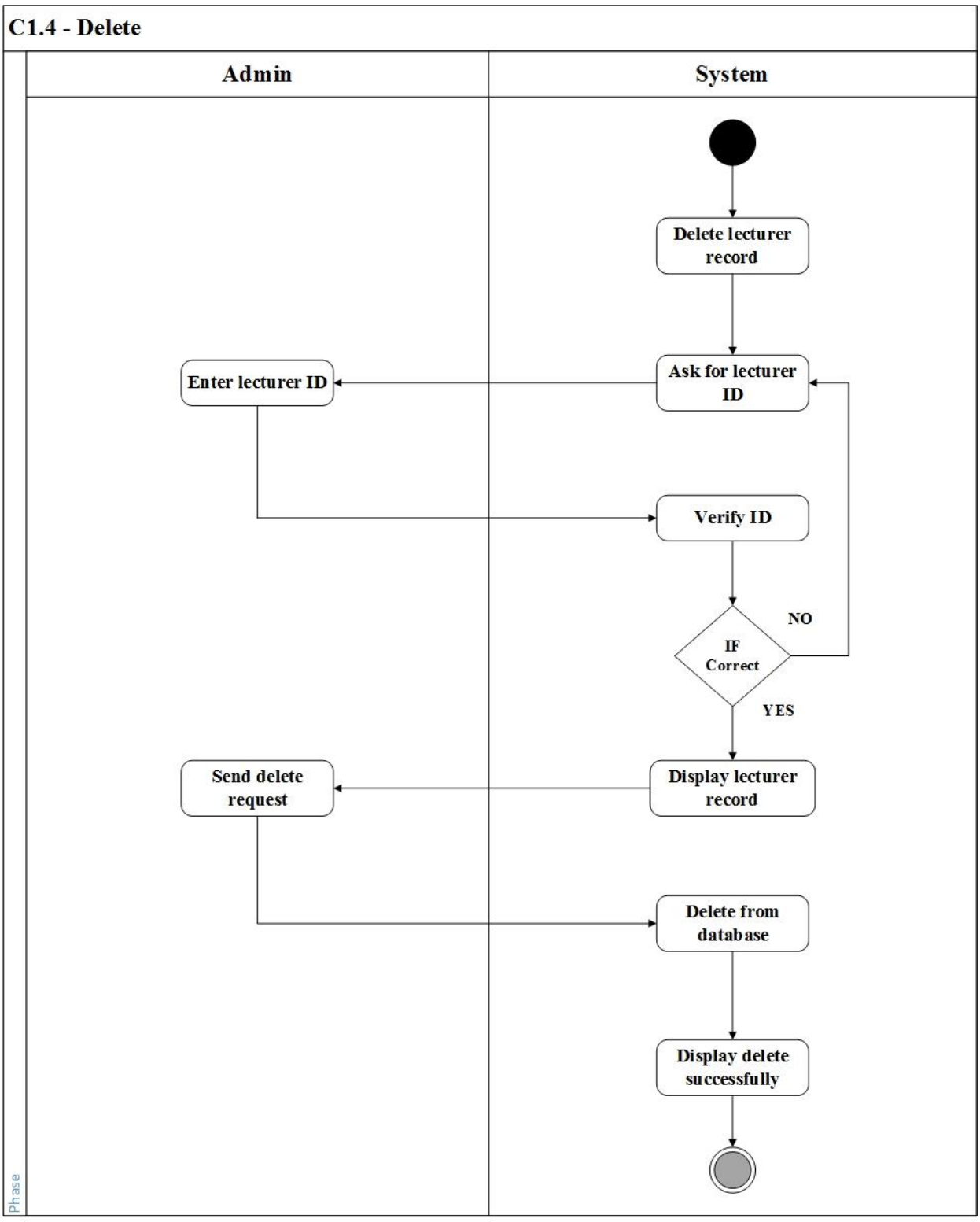

Figure 14: Delete

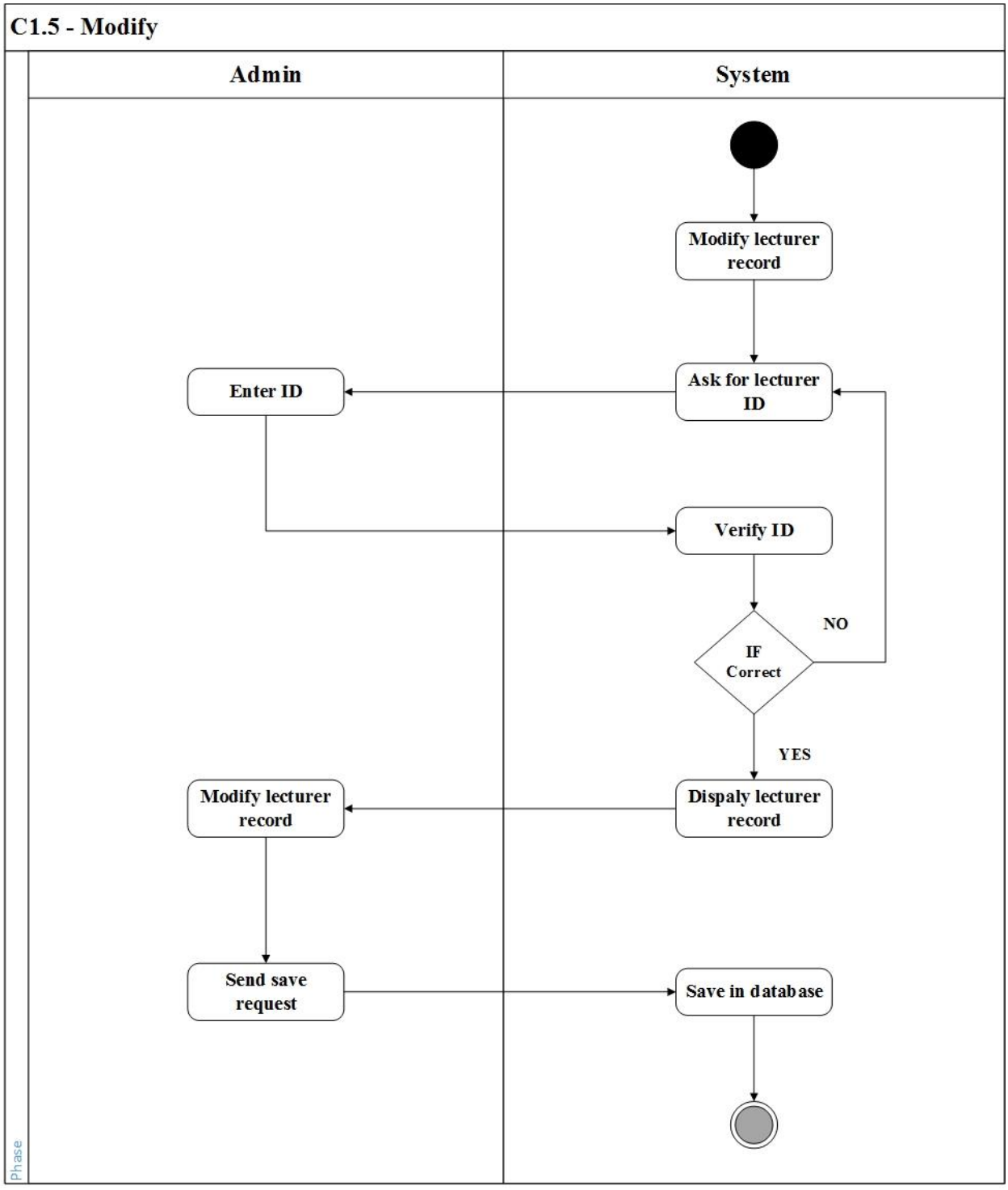

Figure 15: Modify

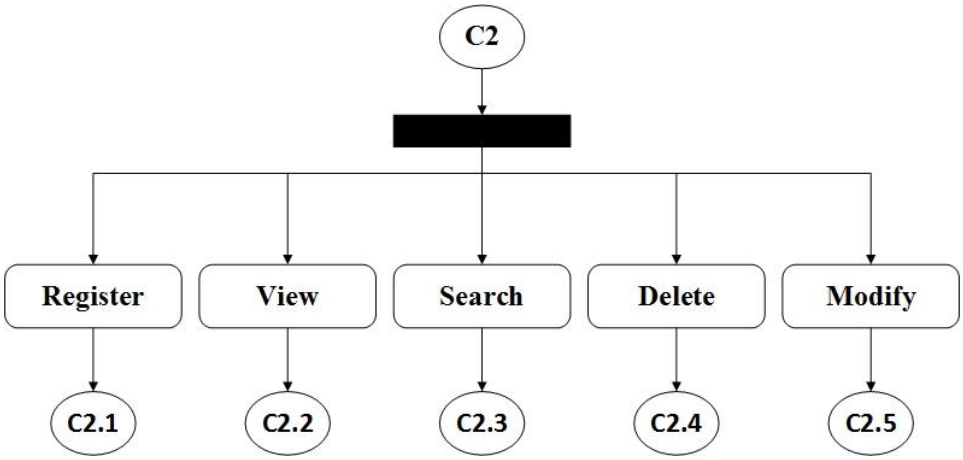

Figure 16: Student

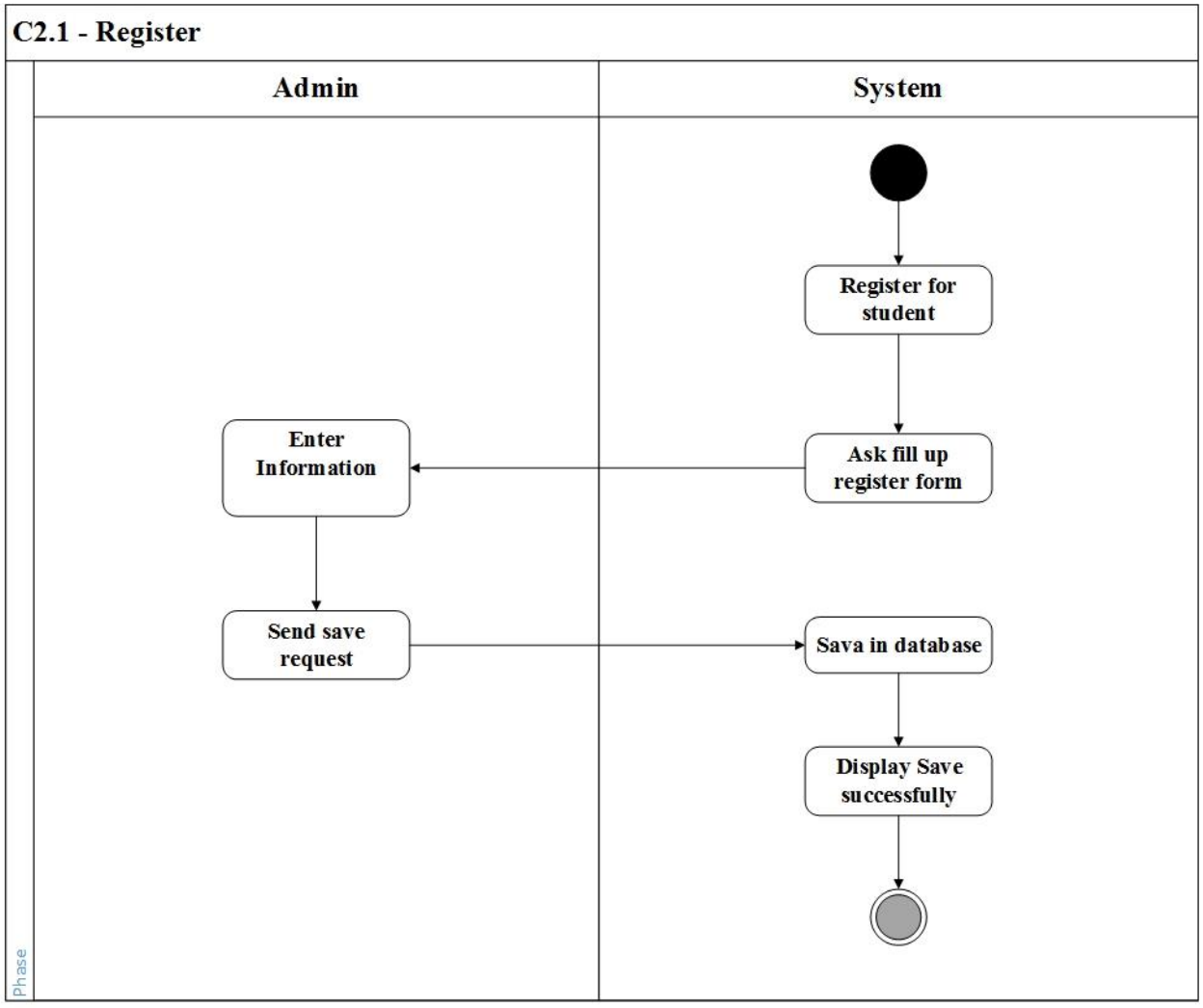

Figure 17: Register

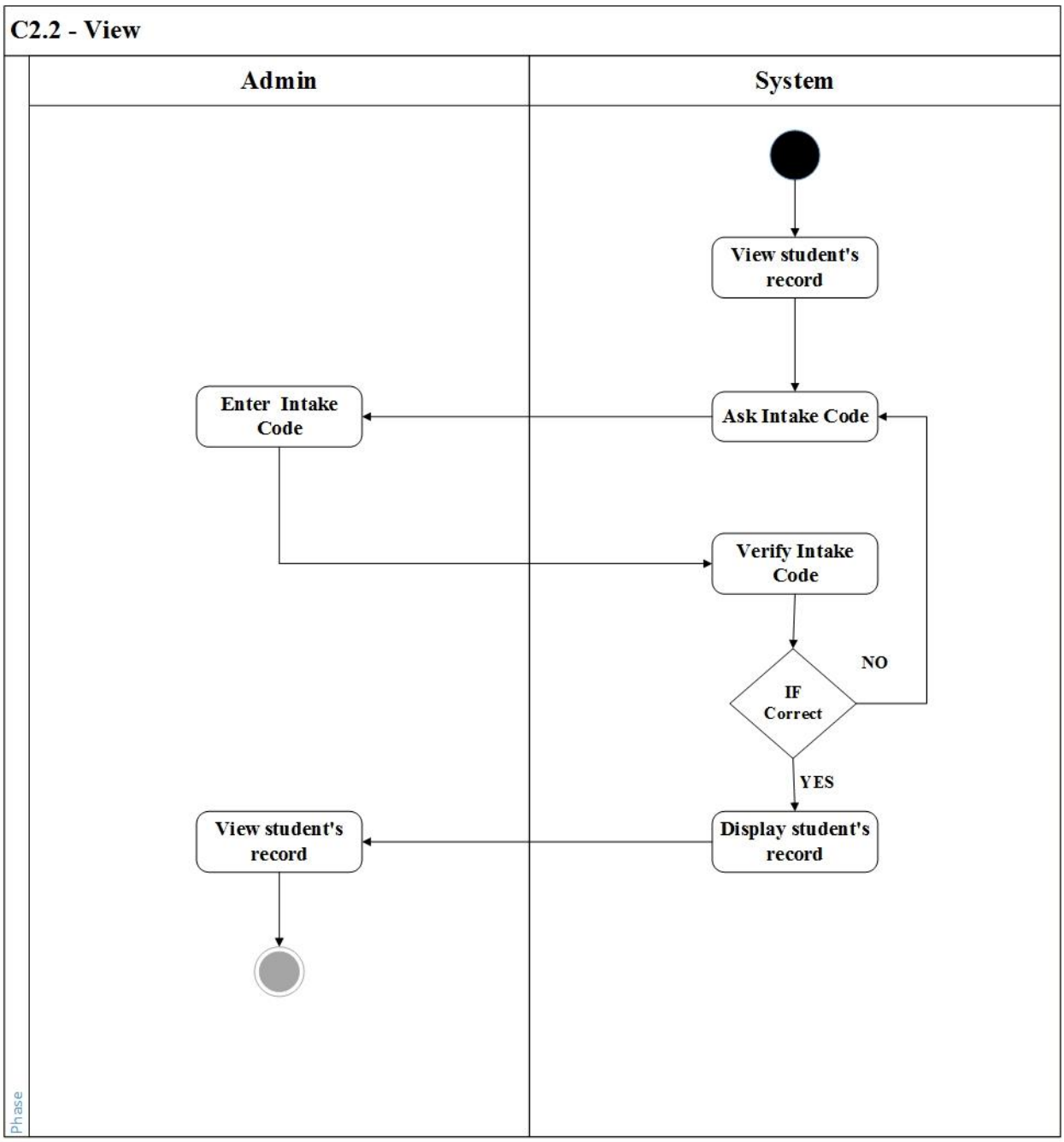

Figure 18: View

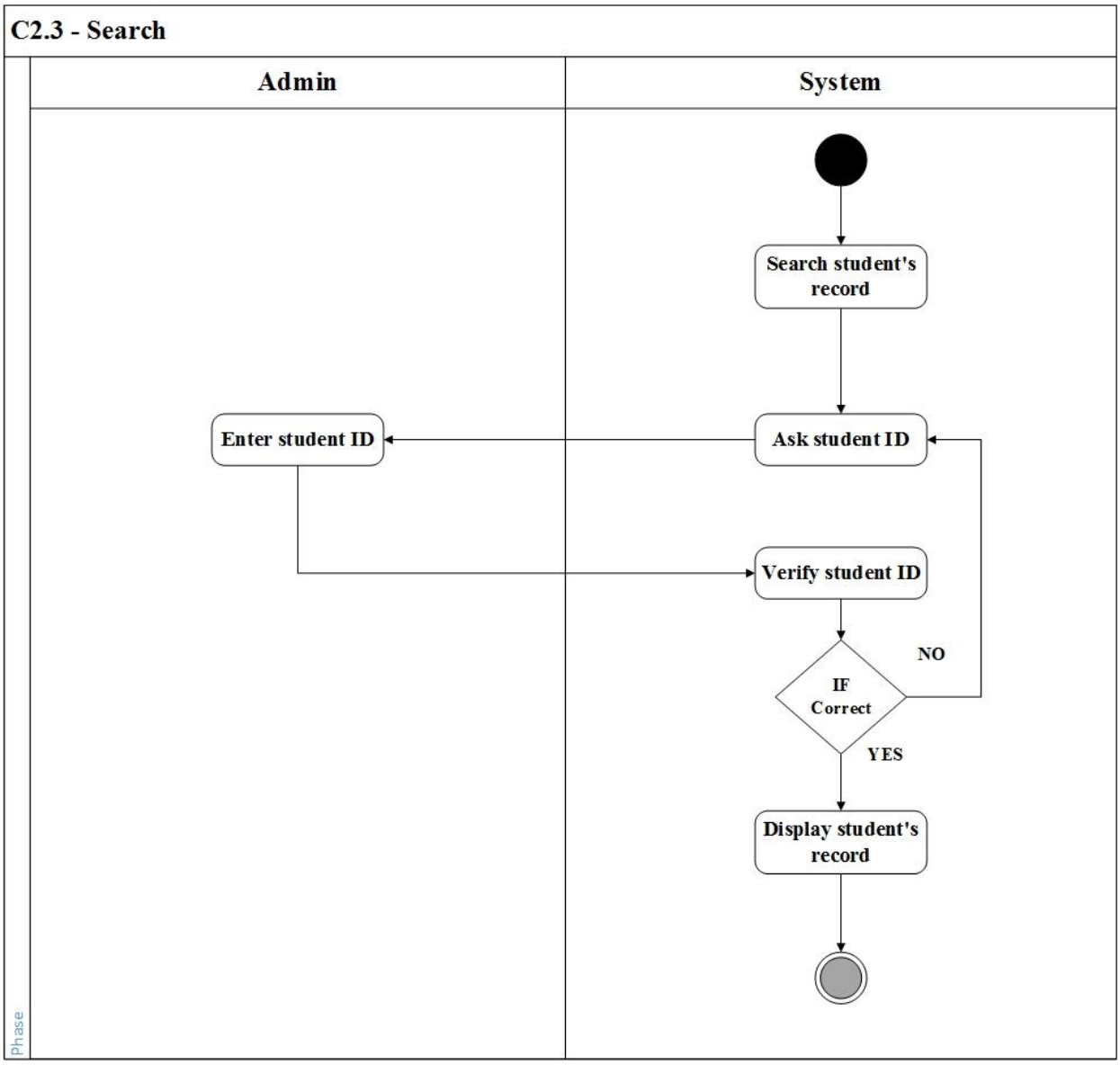

Figure 19: Search

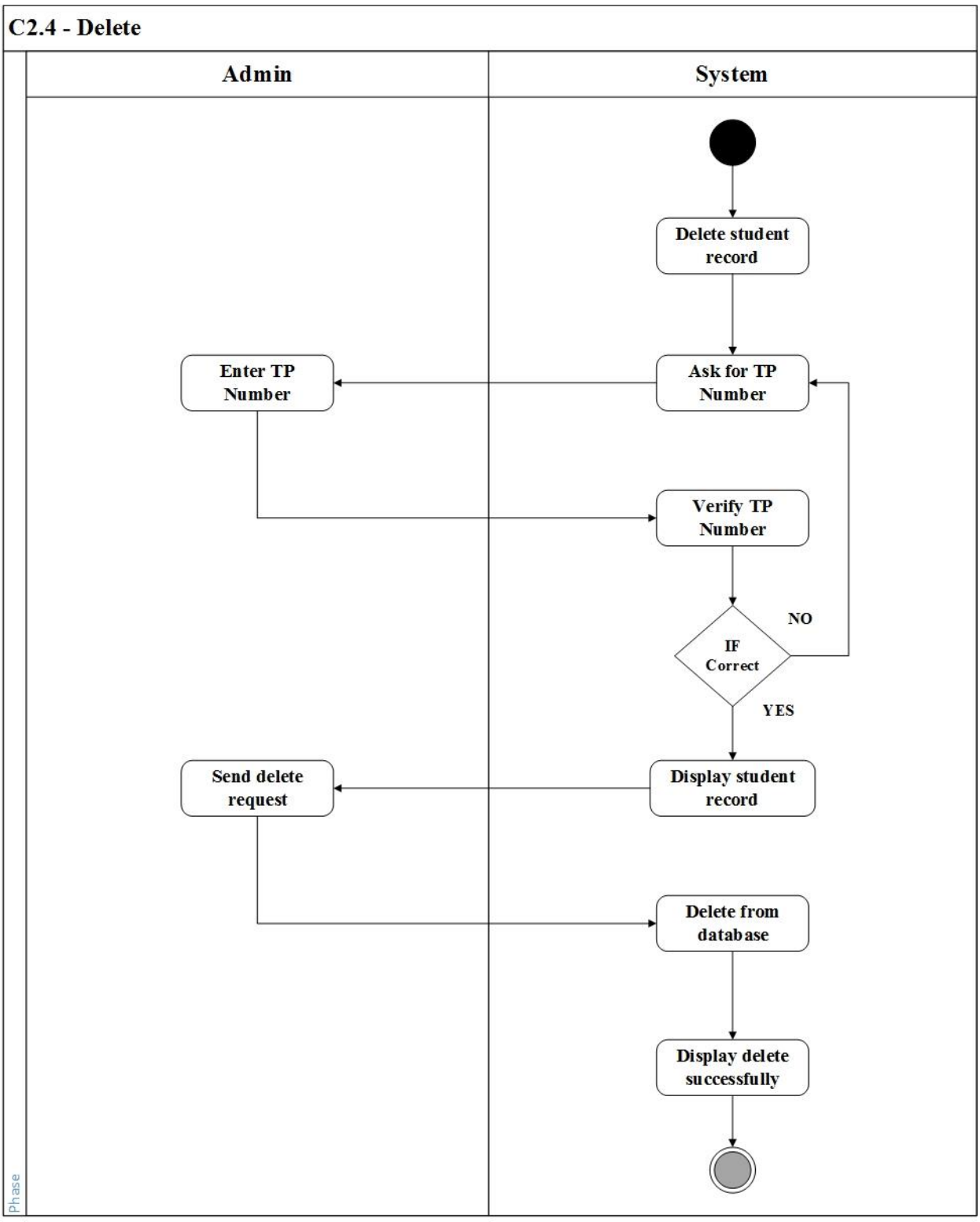

Figure 20: Delete

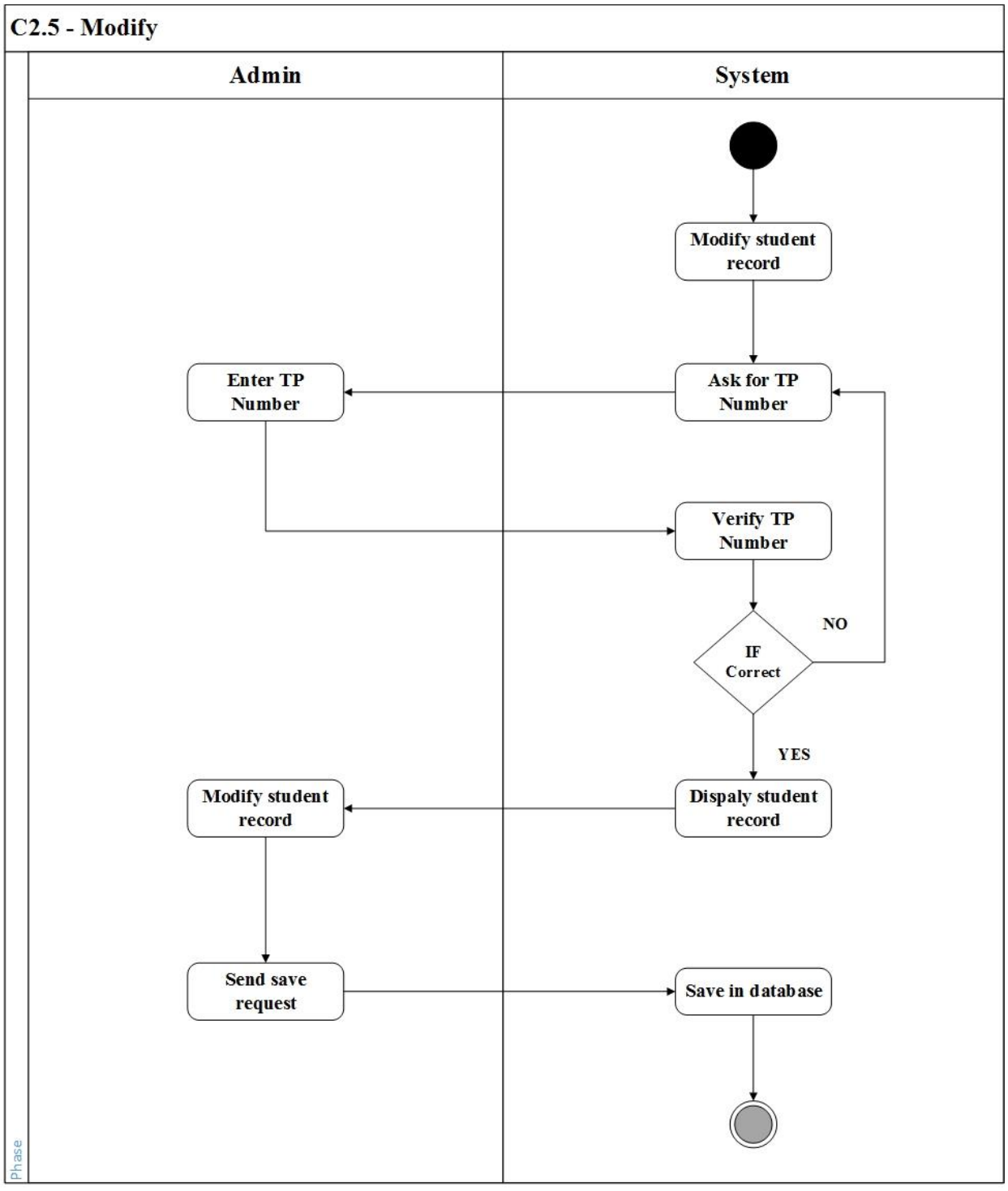

Figure 21: Modify

3.4.2 Lecturer and System

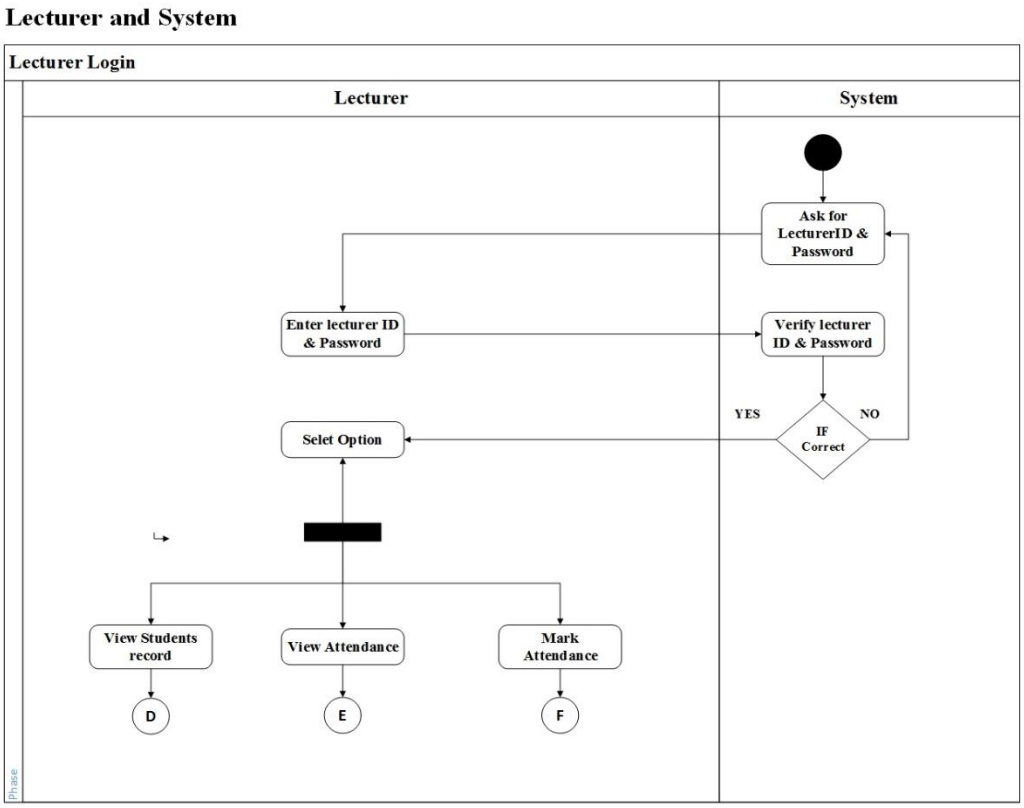

Figure 22: Lecturer Login

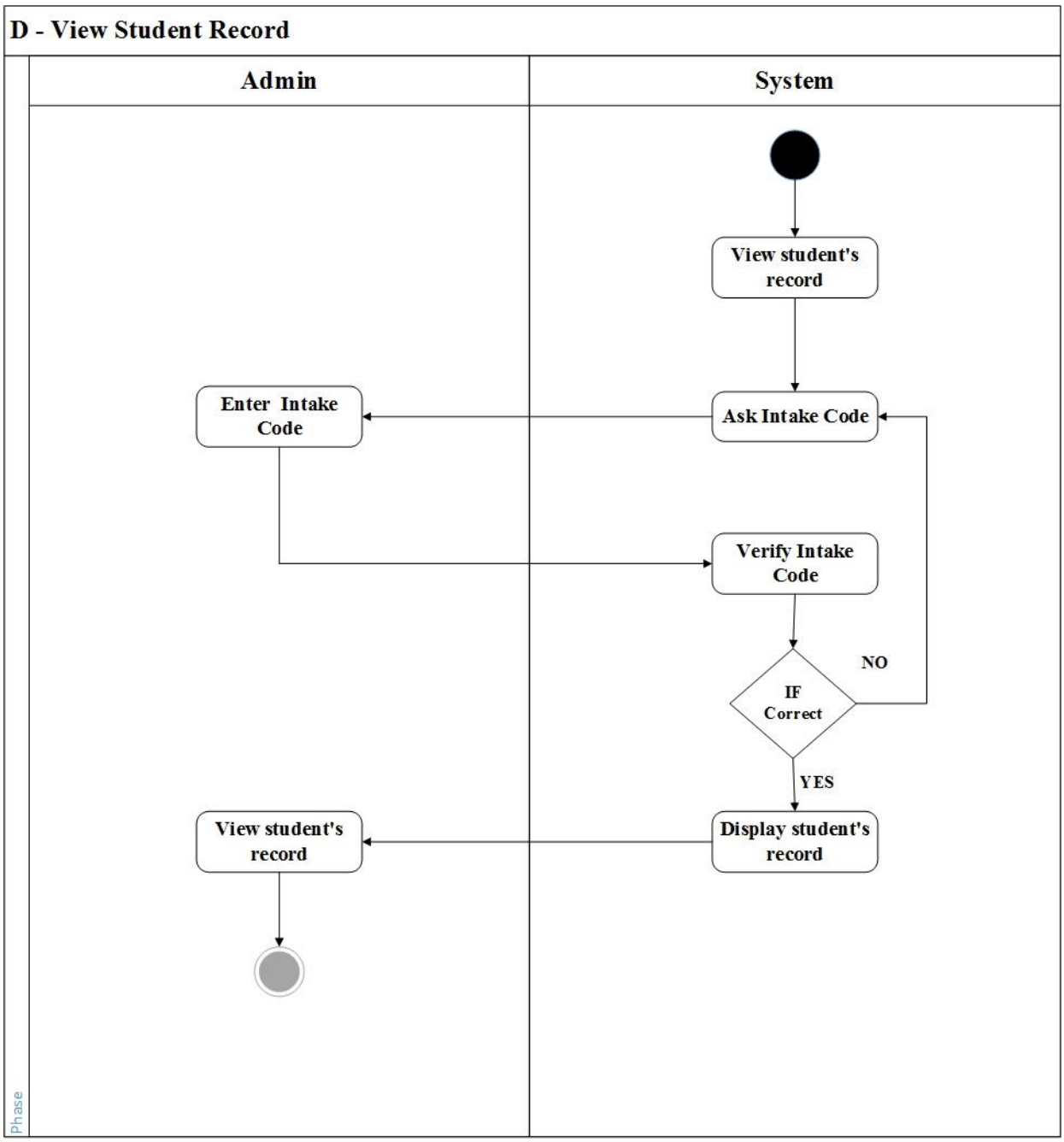

Figure 23: View Student Record

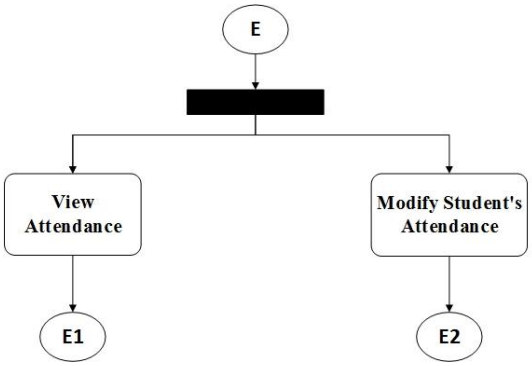

Figure 24: Attendance

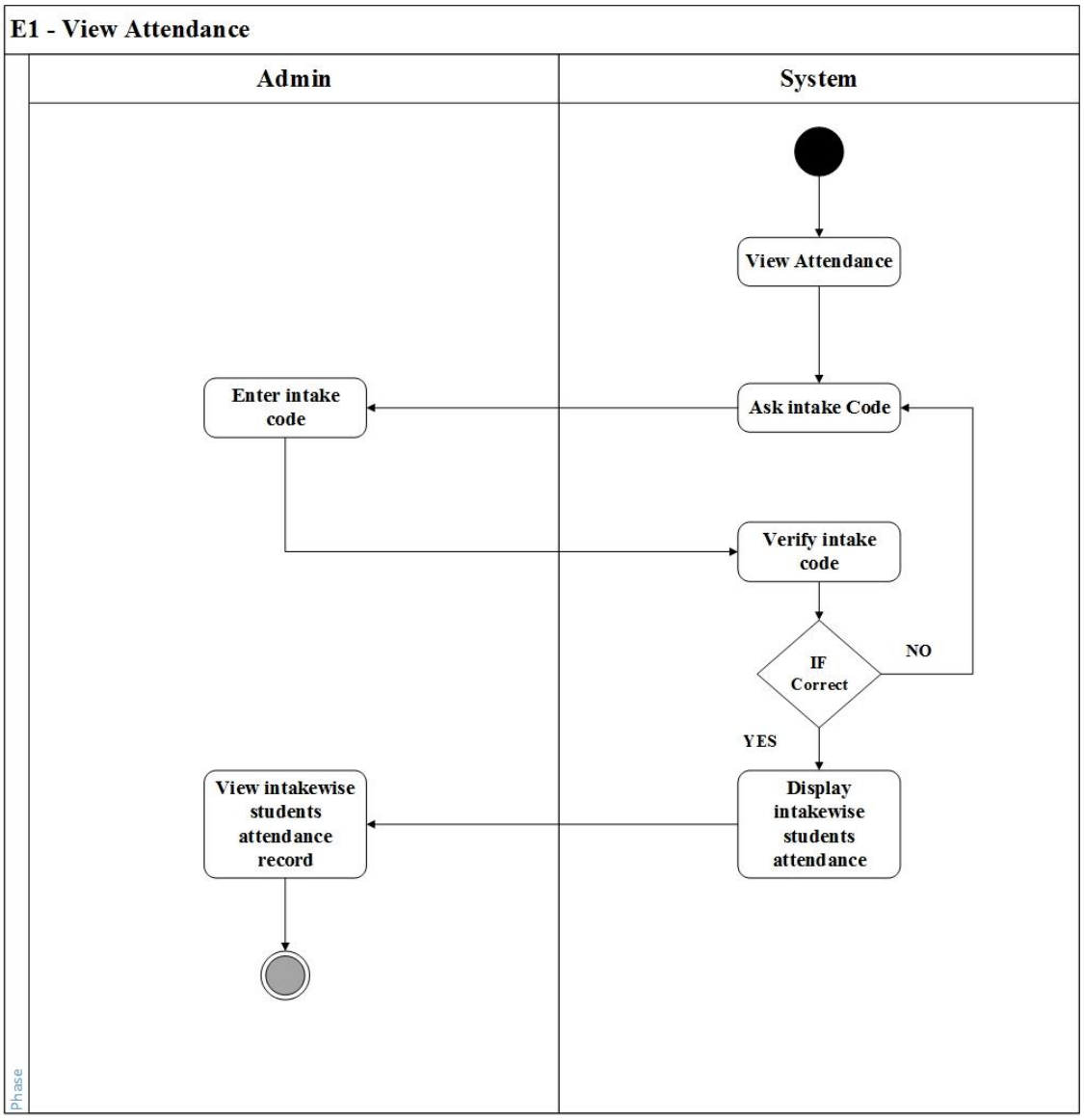

Figure 25: View Attendance

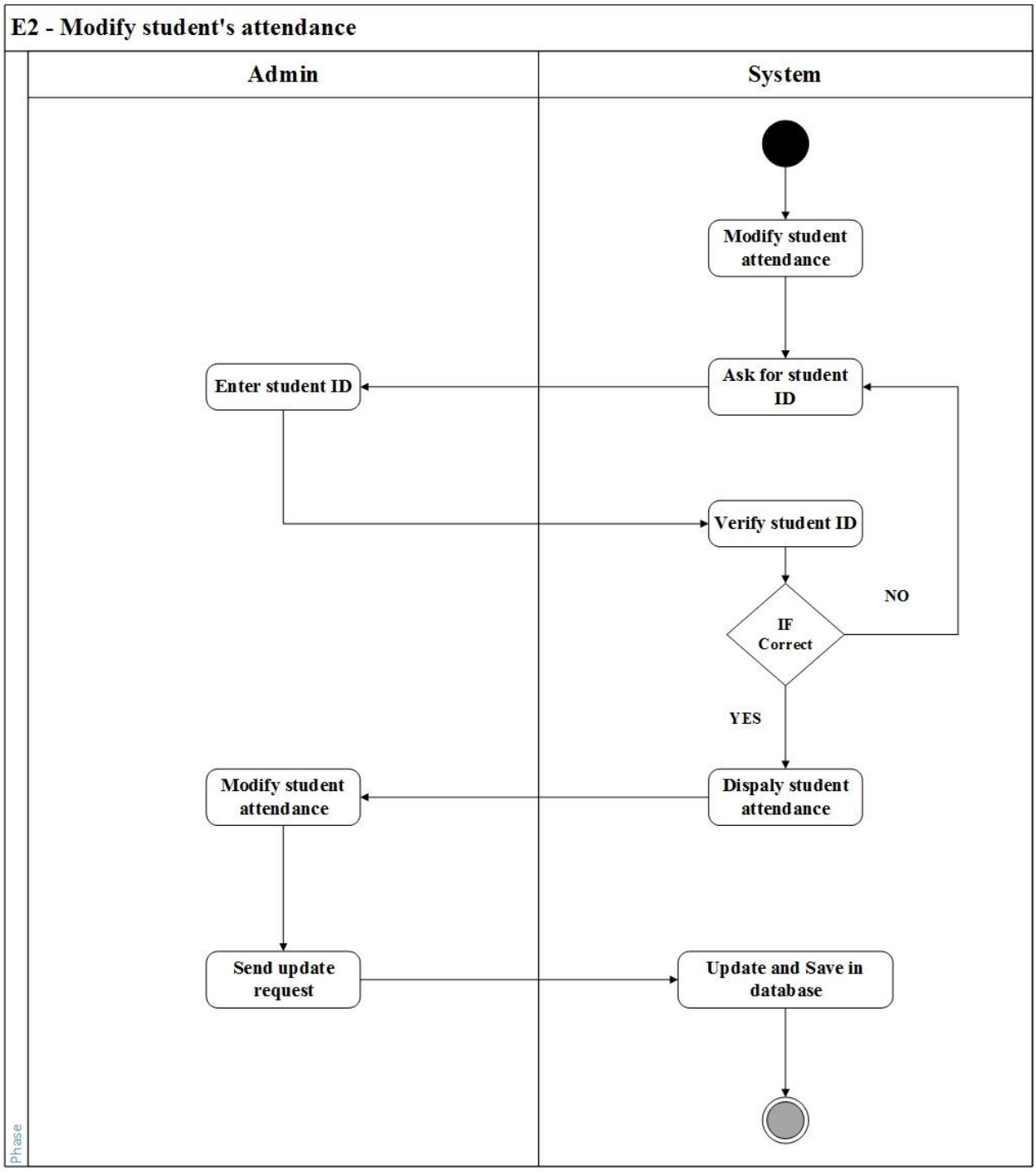

Figure 26: Modify Student Attendance

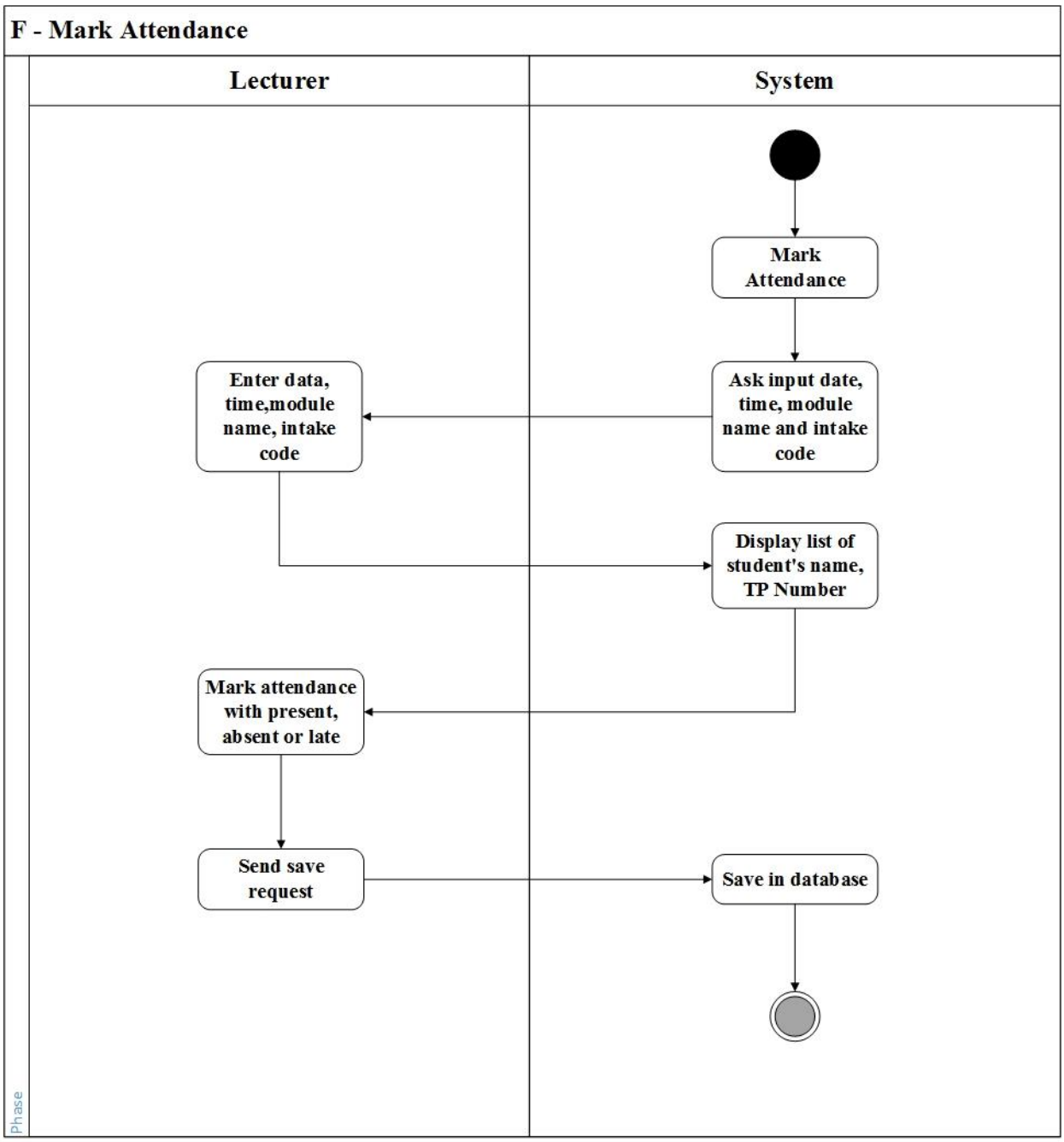

Figure 27: Mark Attendance

3.4.3 Student and System

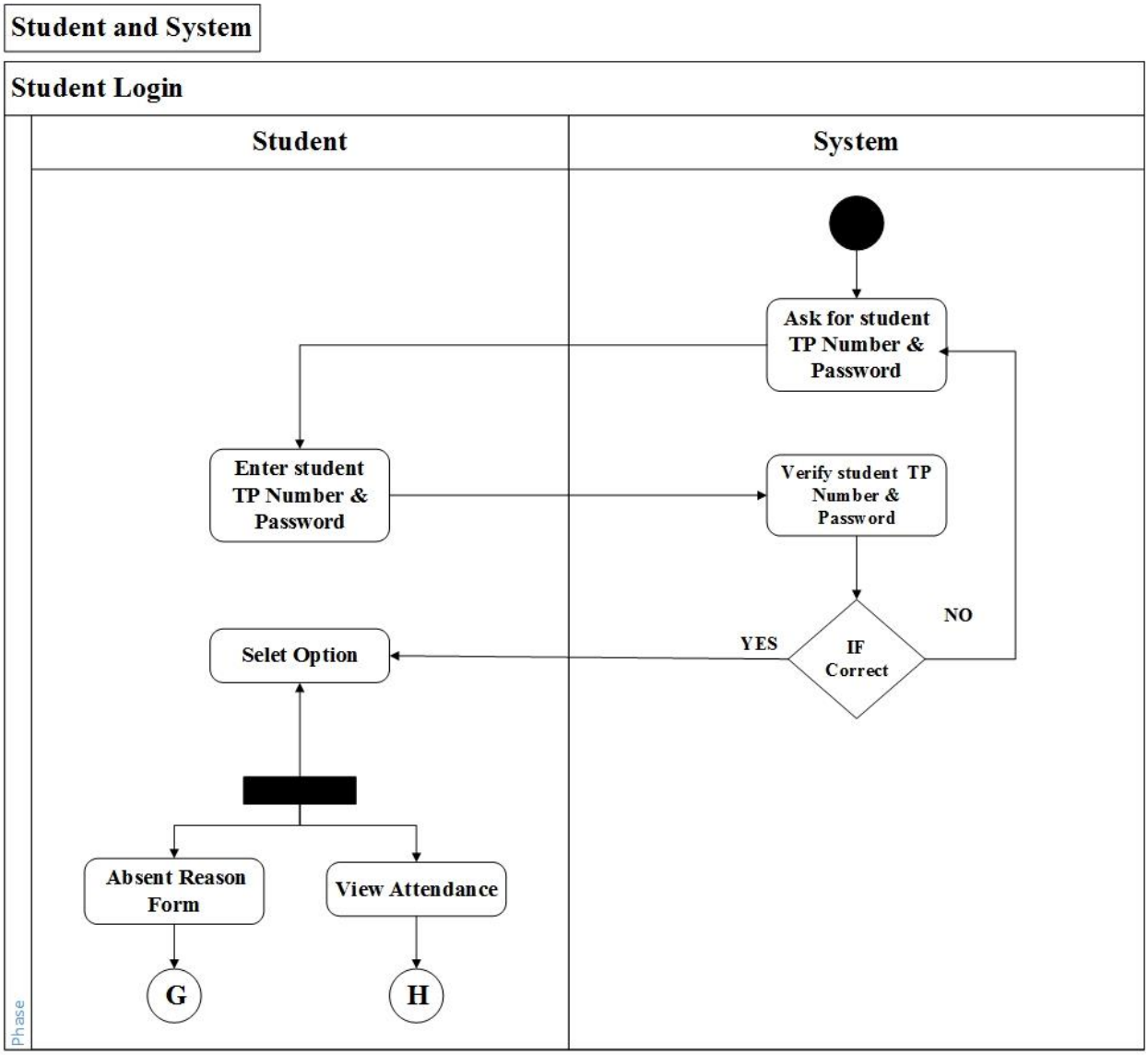

Figure 28: Student Login

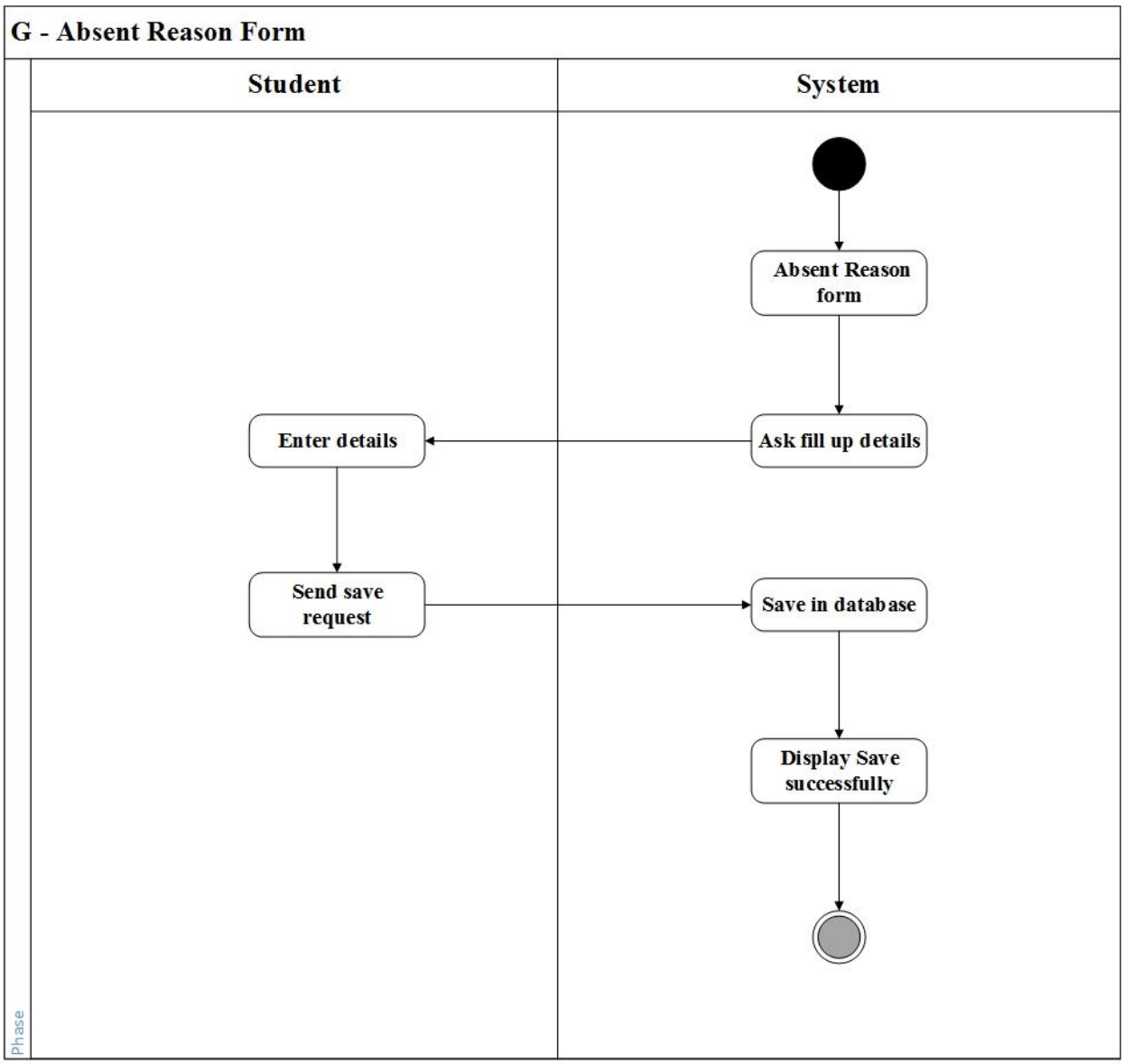

Figure 29: Absent Reason Form

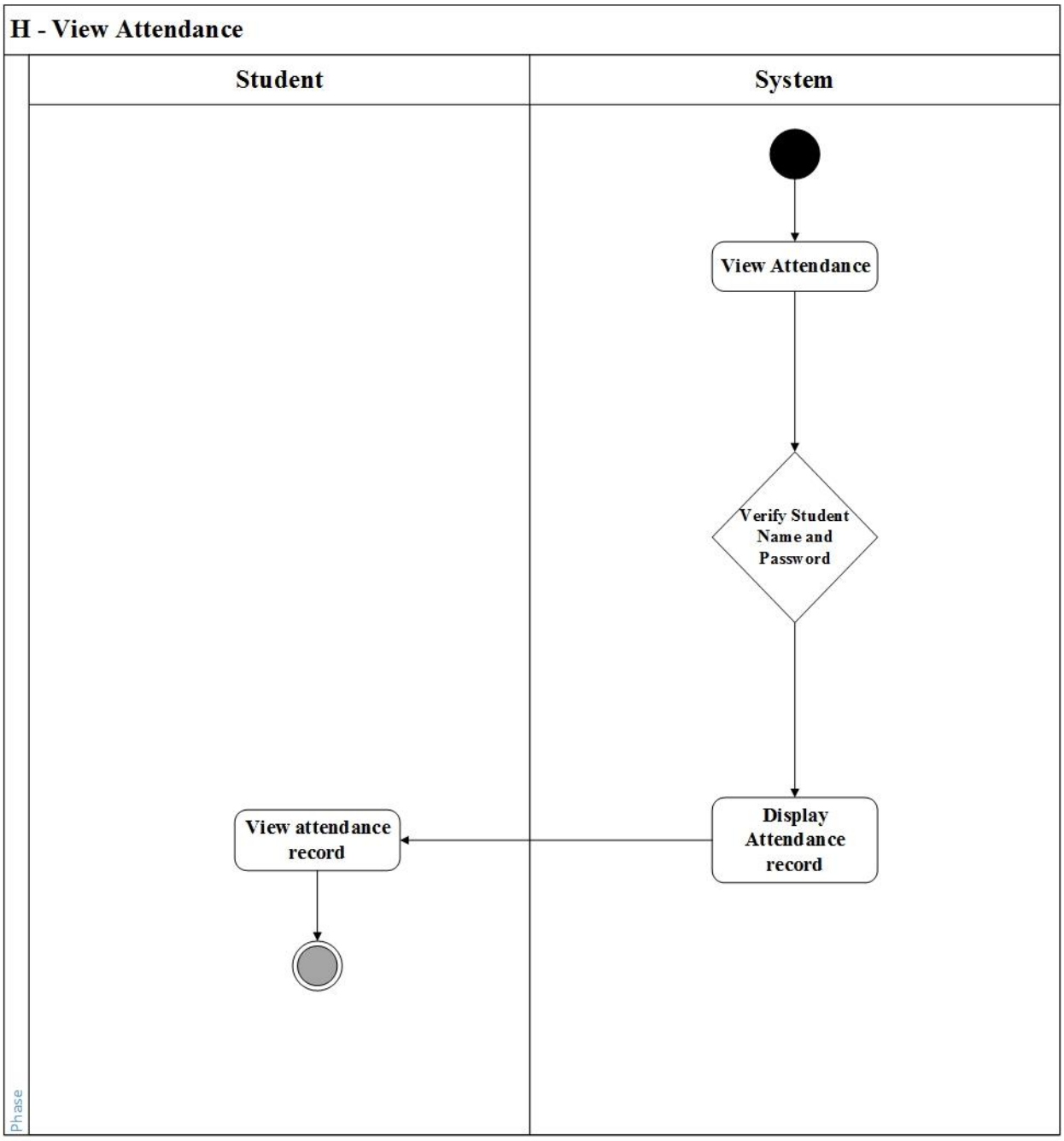

Figure 30: View Attendance

4.0 Source Code

4.1 Classes

```
package xyz.attendance.management.system;

import java.io.BufferedReader;
import java.io.FileReader;
import java.util.ArrayList;
import javax.swing.JOptionPane;

public class AdminClass {
    public int AdminID;


    public void Search()
    {

    }


    public void DeleteRecord()
    {

    }

    public void ModifyDetail()
    {

    }

    public void ModifyAbsentwithReason()
    {
```

Figure 31: Class

The class can be defined by a template that describes the behavior of the object that supports its type. The class can contain three type of variable that is a local variable that is defined inside the method, instance variable that is variables within the class but outside the methods and the class variable that are variable also within a class but outside the methods that are declared with the static keyword. Besides this class have a number of methods to access the value of a different kind of method. In the above example DeleteRecord() and ModifyDetail() are the method.

4.2 Objects

```
RegisterClass nn = new  RegisterClass(FullName,TPNumber,IntakeCode,Password,MobileNumber,Email,Gender,ModuleName);
```

Figure 32: Object

The object is an instance of a class that has states and behaviors. Example human is an object and it has a name that is a behavior. The object is stored in the fields and the behaviors are shown by using methods. Class provides the blue print for the object so on this basis the object is created from the class. In the example above new word is used to generate a new object.

4.3 Variables

```
String intakefile = cd + ".txt";
try {
    BufferedReader input = new BufferedReader(new FileReader(intakefile));
    String line = input.readLine();
    System.out.println(line);
    while (line != null) {
        String[] records = line.split(";");
        String StudentName = records[0];
        String TPNumber = records[1];
```

Figure 33: Variables

Variable is considered as a memory location. Each variable must be declared with a data type. Variables is used to store data such as a number or a string. For example, in the above: String StudentName = record [0], here Student Name is a variable that store the data that is present on the records [0].

4.4 Method

```
String intakefile = cd + ".txt";
try {
    BufferedReader input = new BufferedReader(new FileReader(intakefile));
    String line = input.readLine();
    System.out.println(line);
    while (line != null) {
        String[] records = line.split(";");
        String StudentName = records[0];
        String TPNumber = records[1];
```

Figure 34: Method

A method is a statement that has a specific name and can be executed by calling it by the other place of the program. For example, System.out.prinln(line) in this code line is a method that is calling from the other place in the code.

4.5 Inheritance

```
public class Admin extends User{

    public boolean loginpass(String name, String pass)
    {
    boolean R=false;
    try
    {
```

Figure 35: Inheritance

Inheritance allows one class to inherit the feature of another class that is field or method. The class that feature is inherited to another class are called super class like in this example User is the super class and the class that is Admin, that inherit the feature of User are called sub class.

4.6 Constructor

```
public class Registration extends Admin{
    public String Name;
    public String ID;
    public String Email;
    public String intakecode;
    public String modulename;
    public String loginpassword;

    Registration(){}

    Registration(String name,String id,String email,String intakecode, String modulename, String loginpassword)
    {
        this.Name = name;
        this.ID = id;
        this.Email = email;
        this.intakecode = intakecode;
        this.modulename = modulename;
        this.loginpassword = loginpassword;
    }
```

Figure 36: Constructor

The constructor is similar like the method that also called when an instance of an object is created but constructor do not have the return type as well as constructor name is same as class name and also constructor is not thinking as a member of a class. For example, in the above registration is a constructor because the class name and the constructor name is same.

4.7 Looping structure

```
while ((line = reader.nextLine()) != null) {
    //splite each line to check if the search name is present.
    lineArr = line.split(";");
    if (lineArr[0].equals(searchFullname)) {

        //if the search name is present, add the old data
        //plus the new data in to the temp array
        tempArray.add(
                lineArr[0] + ";"
                + lineArr[1] + ";"
                + lineArr[2] + ";"
                + lineArr[3] + ";"
                + lineArr[4] + ";"
                + newPassword  + ";"
                + newMobileNumber  + ";"
                + newEmail);
```

Figure 37: Looping Structure

While loop first check the condition is true or not, if the condition is true then it goes inside the loop otherwise it goes outside the loop. For example, in the above, while loop will check if the text file is null or something has that can be read so if there is some word to read then the loop will go inside the body otherwise if the text file is empty then it will show error.

4.8 get and set method

```
public int getid()
{
    return Id;
}

public String getpassword()
{
    return Password;
}

public void setid(int Id)
{
    this.Id = Id;
}
```

Figure 38: get and set method

A get method is used to obtain a variable value from a class and it will always return value where the as set method is used to store the variable and it will not return value. In the above example, getting is obtained a variable and set store the variable.

4.9 Exceptional handler

```
String intakefile = cd + ".txt";
try {
    BufferedReader input = new BufferedReader(new FileReader(intakefile));
    String line = input.readLine();
    System.out.println(line);
    while (line != null) {
        String[] records = line.split(";");

        String StudentName = records[0];
        String TPNumber = records[1];
        String Attendance= records[2];
        String ModuleName = records[3];
        String Date = records[4];

        model.addRow(new Object[]{StudentName, TPNumber,Attendance,ModuleName,Date});
        line = input.readLine();
    }
    input.close();
} catch (IOException e) {
    JOptionPane.showMessageDialog(null, "Error");
}
```

Figure 39: Exceptional Handler

Try catch block is uses to validating or error handling purpose. So in the Student record, I use this try and catch block to handle the error if the file is not saved to the text file then it will give a message of "Error".

4.10 File Handling

```
try {
    try (FileReader fr = new FileReader("Lecturer.txt")) {
        Scanner reader = new Scanner(fr);
        String line;
        String[] lineArr;

        while ((line = reader.nextLine()) != null) {
            //splite each line to check if the search name is present.
            lineArr = line.split(";");
            if (lineArr[0].equals(searchFullname)) {

                //if the search name is present, add the old data
                //plus the new data in to the temp array
                tempArray.add(
                        lineArr[0] + ";"
                        + lineArr[1] + ";"
                        + lineArr[2] + ";"
                        + lineArr[3] + ";"
                        + lineArr[4] + ";"
                        + newPassword  + ";"
                        + newMobileNumber  + ";"
                        + newEmail);
```

Figure 40: File System

"Java.io.BufferedReader" class reads text from a character-input stream, buffering characters so as to provide for the efficient reading of characters, arrays, and lines and the "Java.io.FileReader" class is a convenience class for reading character files. "Java.io.BufferedWriter" class writes text to a character-output stream, buffering characters so as to provide for the efficient writing of single characters, arrays, and strings and the "Java.io.FileWriter" class is a convenience class for writing character files.

4.11 Package

```
package xyz.attendance.management.system;

public class XYZAttendanceManagementSystem {


    public static void main(String[] args) {
        WelcomePage l = new WelcomePage();
        l.setVisible(true);

    }

}
```

Figure 41: Package

This package is the XYZ Attendance Management System. It contains all the classes and the image.

5.0 Additional Features

5.1 Jtables used:

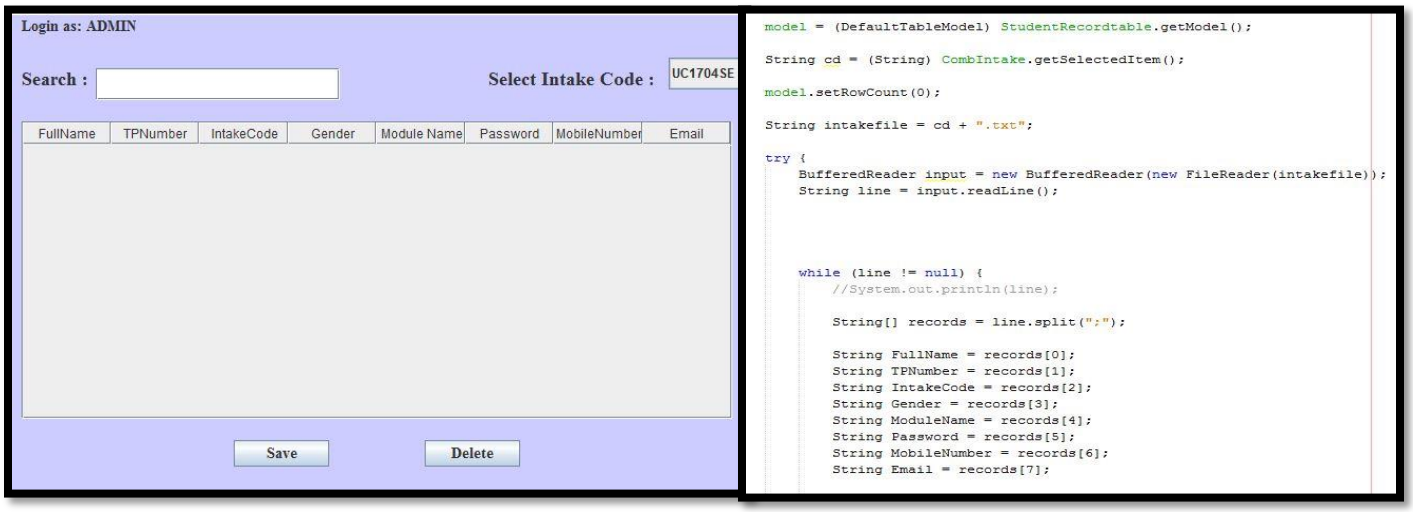

Figure 42: Jtable

Jtables is used to show the student and lecturer record as well as it is used to take attendance, register, modify and delete new student and lecturer. Above the diagram provide the design of jtable and the code used to record the data in the table. In jtable columns are defined as object arrays and each data is loaded in the respective columns of the jtable. This makes easier for the user to see instantly for example if admin register a new student, after register the data of the student will come to the jtable automatically. Since they can confirm it is recorded in the text file.

5.2 Protected Password:

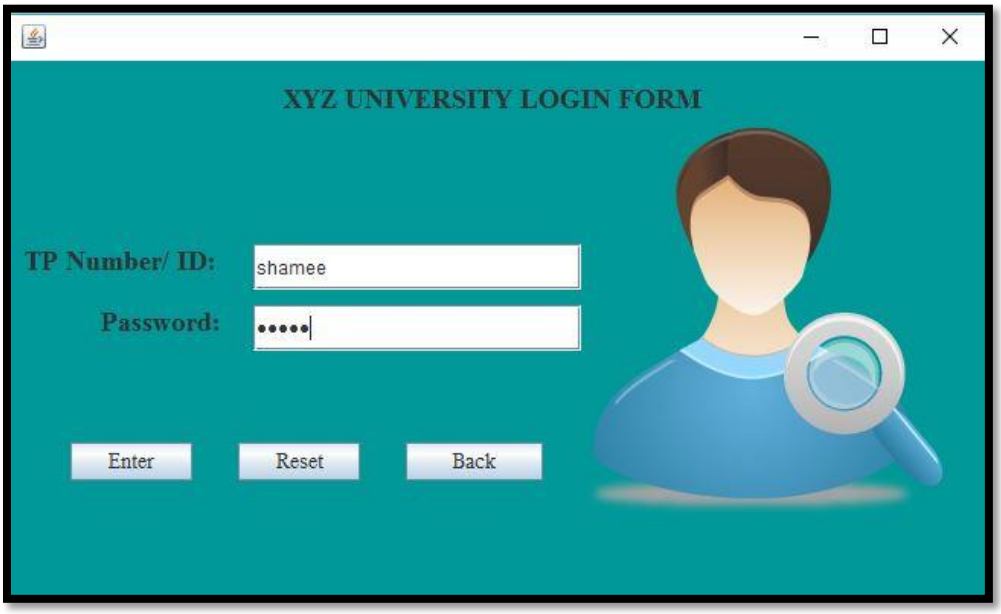

Figure 43: Protected Password

This password text field is giving the safety about their password so other people cannot see what type of number or string the user is typing to access into the system. Since it makes the account safe.

6.0Sample Outputs

6.1 Welcome Page

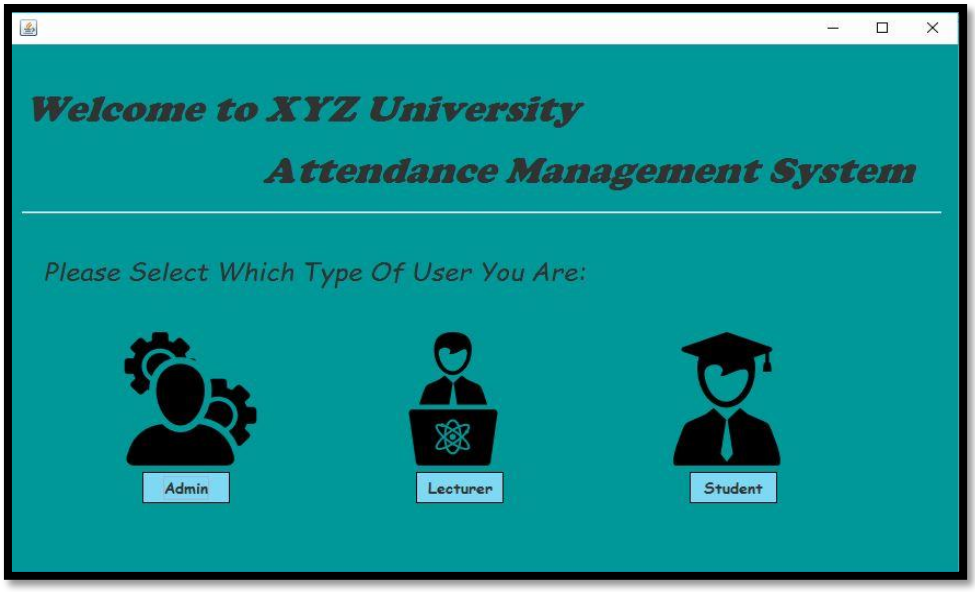

Figure 43: Welcome Page

This is the welcome page of XYZ University, the here user can choose what type of user they are to go the login page.

6.2 Login

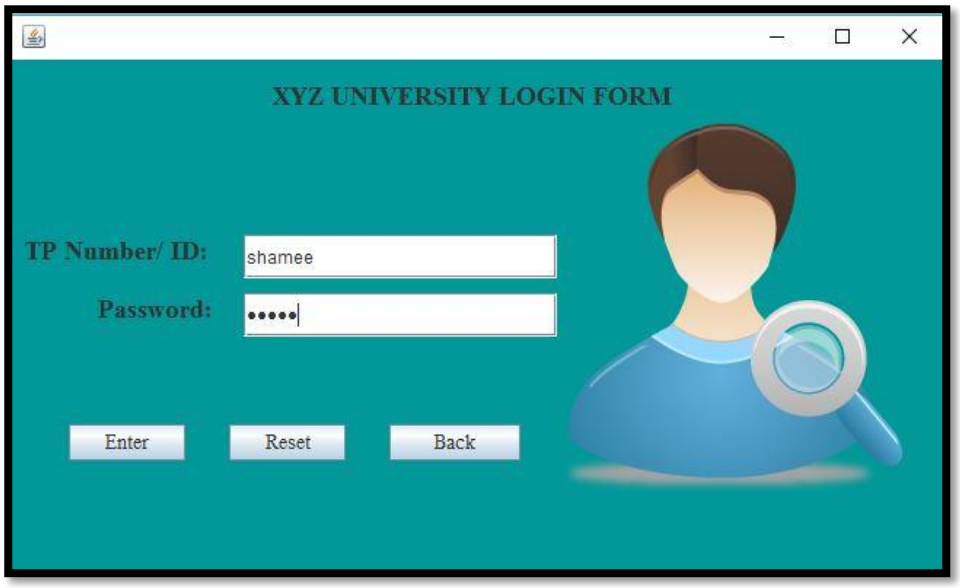

Figure 44: Login Page

In this login, form user needs to write their username and password that provided by admin to access the system. They can press the Reset button to clear the text field and can go back to the welcome page by pressing the Back button.

6.3 Admin main menu

Figure 44: Admin Menu

After login into the system, the admin will see this page as an admin menu. Here admin can choose which type of function he wants to use. There is three four button on page one is for the attendance, another one is to view the student absent reason and the last one is to register new student or lecturer, as well as admin, can view the record. At last, there is another button which is Logout button that is for logout from the account.

6.4 Attendance

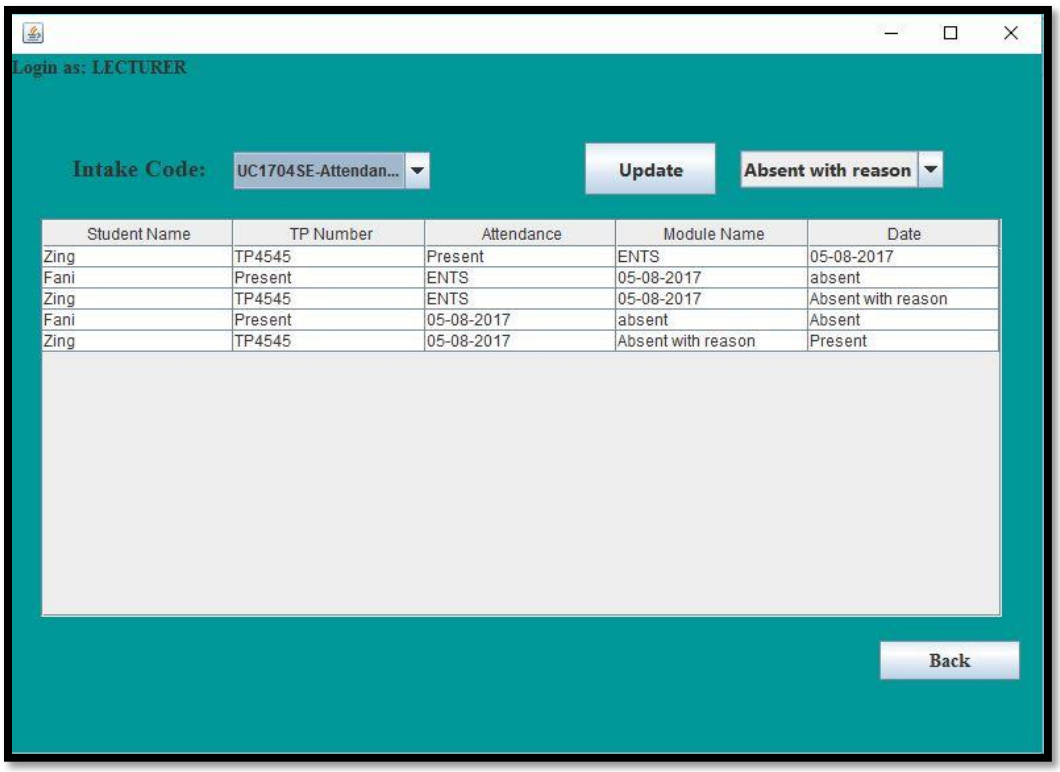

Figure 45: Attendance

After pressing the attendance button, attendance form will come here admin can view the intake wise student attendance by selecting the intake code as well as they can update the attendance of the student by giving absent with reason. Admin can use the back button to go back to the admin again.

6.5 Absent Reason

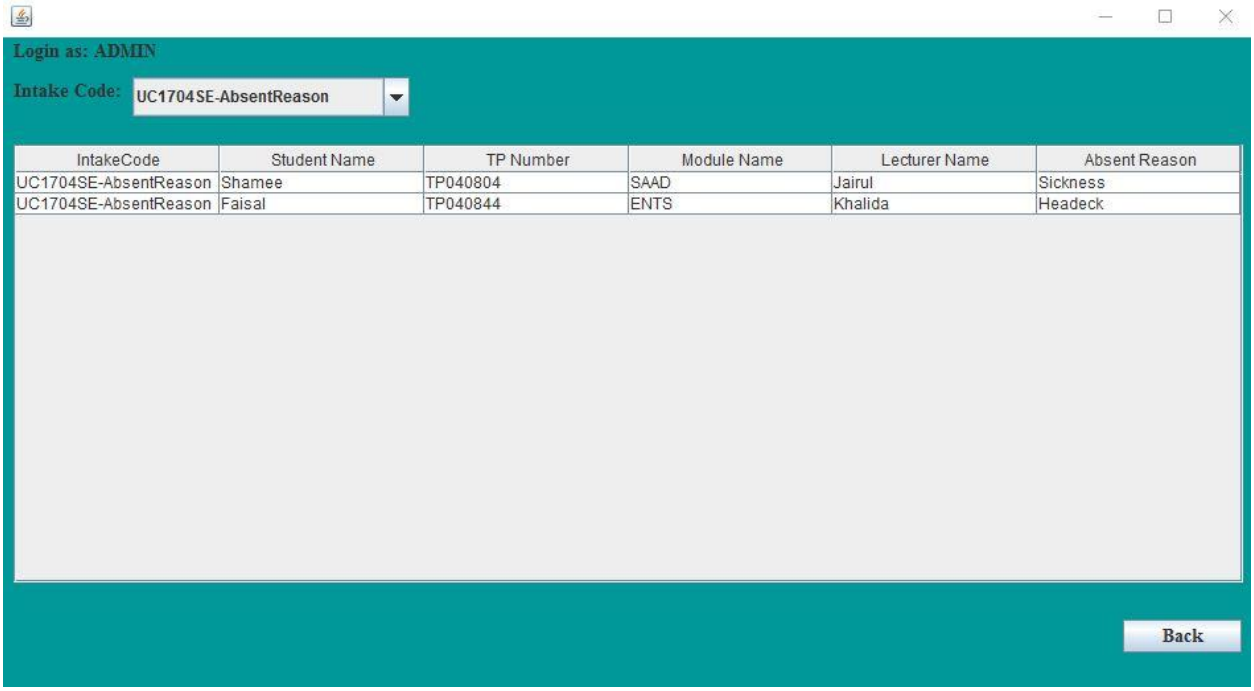

Figure 46: Absent Reason

After pressing the absent reason button, this absent reason page will come. Here admin can see the reason of the student absent by selecting the intake code. In the jtable, admin can see the Intake code, student name, TP Number, Module Name, Lecturer Name and Absent reason. Admin can choose Back button to go back to the Admin Menu page again.

6.6 Register

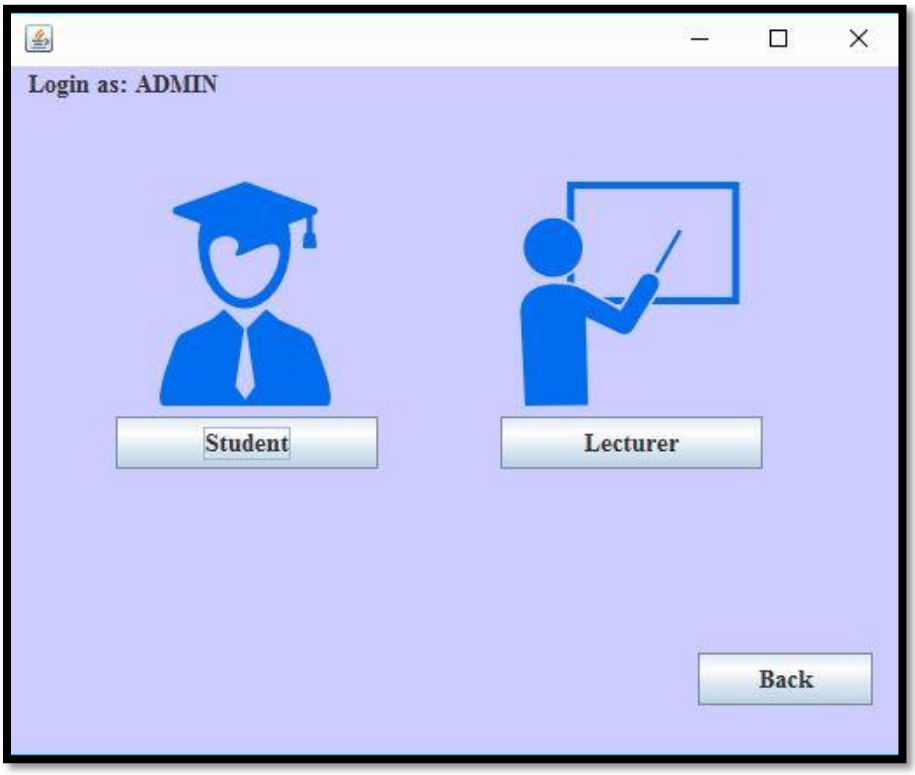

Figure 47: Register

This page is coming after choose the register button, here admin can choose which type of user he wants to select for register or view the record as well as delete and modify the record then also there is back button to go back to the Admin Menu.

6.7 Student

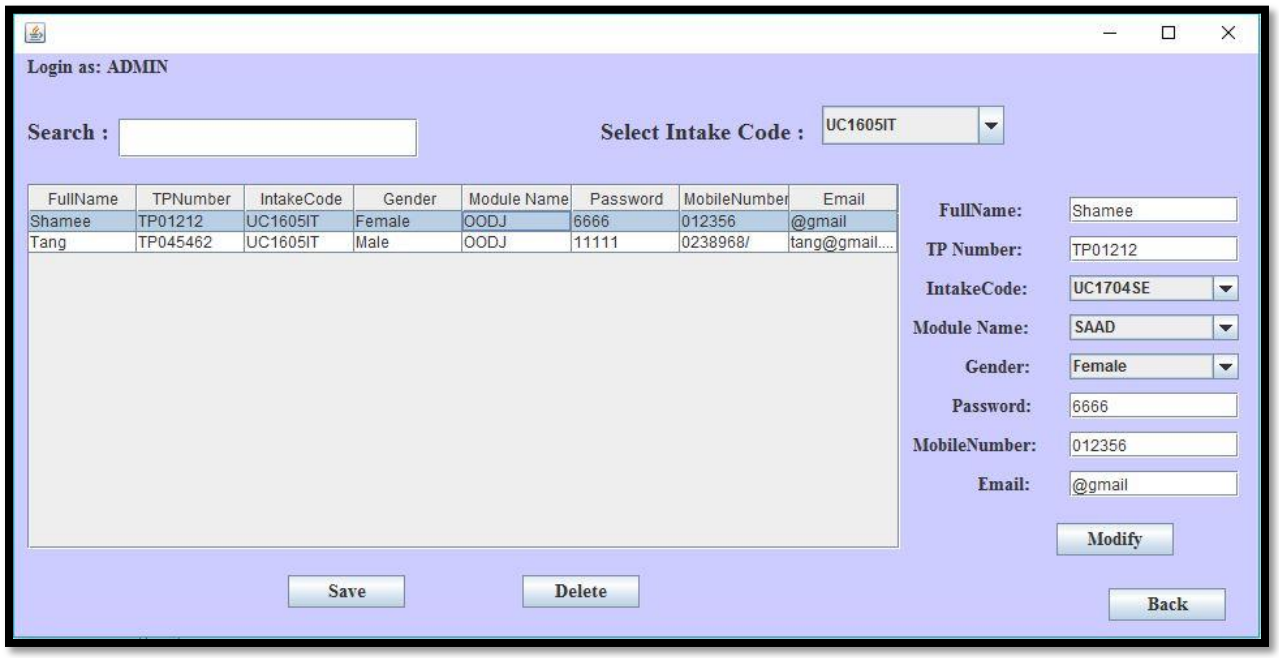

Figure 48: Student

This page will come after pressing the Student button, here admin can view the student detail by using Intake code. Admin can register new student by pressing the save button and modify the student record by pressing the Modify button as well as admin can delete by using the delete button. There is a search option as well to search specific student by their name.

6.8 Lecturer

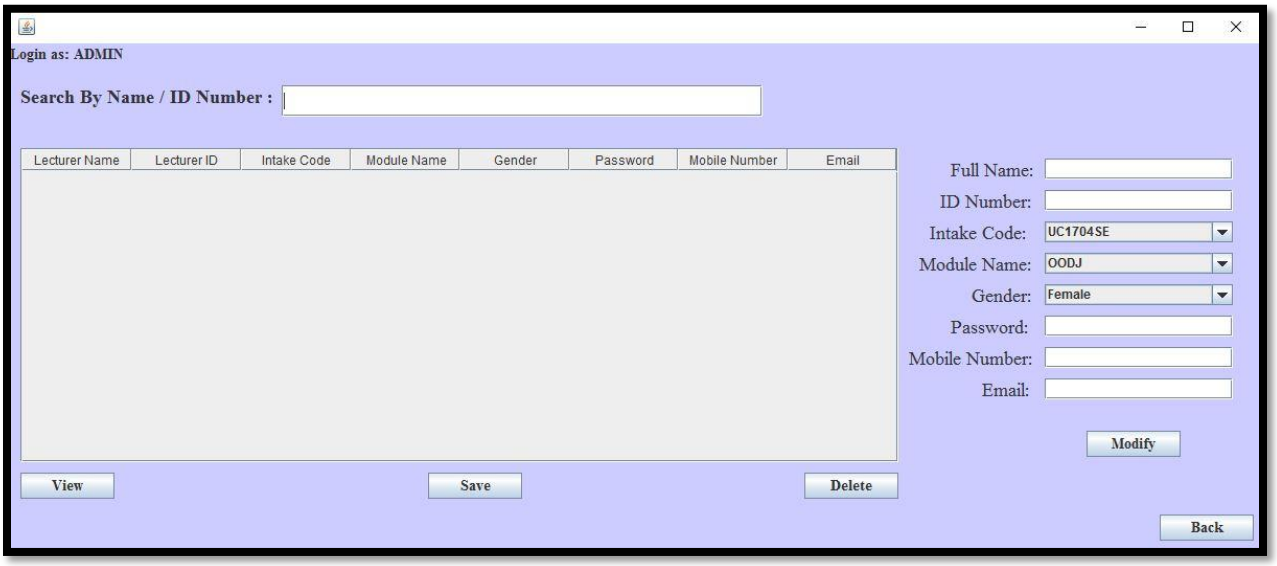

Figure 49: Lecturer

This page will come after pressing the Lecturer button, here admin can view the lecturer detail by using view button. Admin can register new lecturer by pressing the save button and modify the lecturer record by pressing the Modify button as well as admin can delete by using the delete button. There is a search option as well to search specific lecturer by their name or ID.

6.9 Lecturer menu

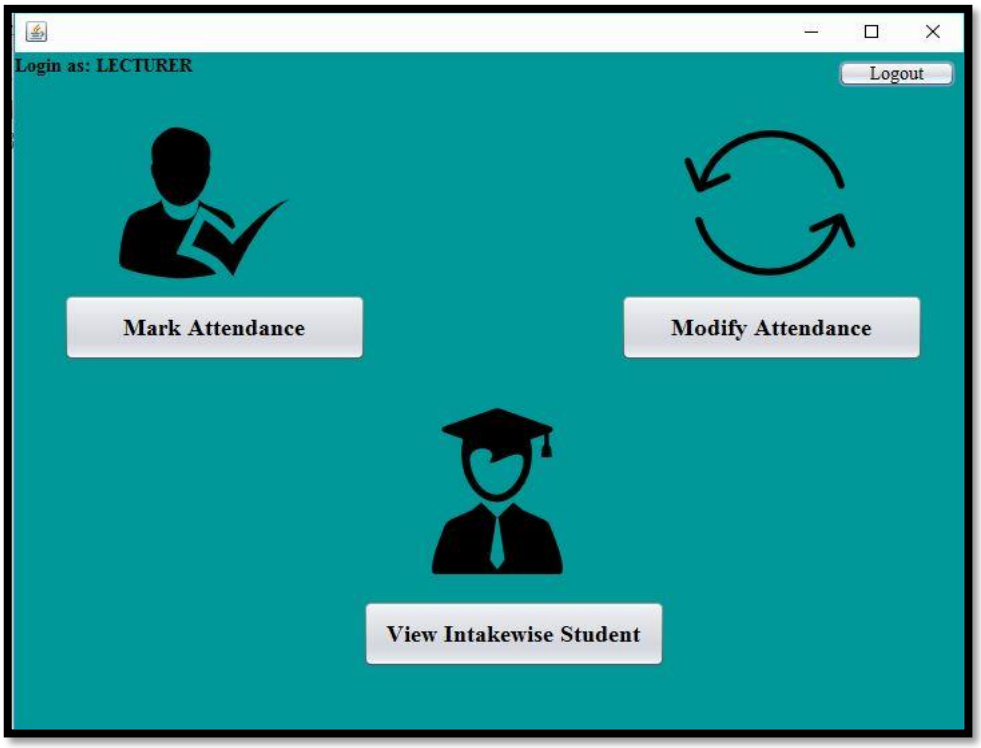

Figure 50: Lecturer Menu

After login into the system, the admin will see this page as an admin menu. Here admin can choose which type of function he wants to use. There is three button on page one is for the attendance, another one is to view the student absent reason and the last one is to register new student or lecturer, as well as admin, can view the record. At last, there is another button which is Logout button that is for logout from the account.

6.10 Mark Attendance

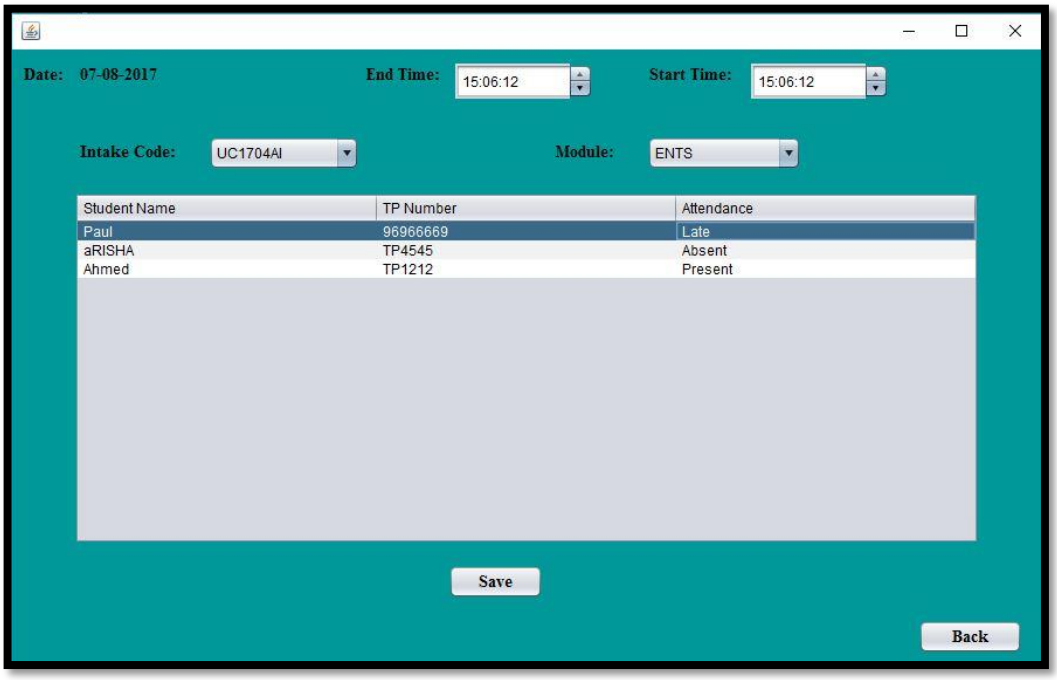

Figure 51: Mark Attendance

After selecting the Mark Attendance button lecturer can view this page. This form is taking the attendance, here lecturer has to select intake code and module name as well as date and time. All the attendance record of the student will show in the jtable and lecturer can take attendance by using jtable then save the attendance record into the system.

6.11 Modify Attendance

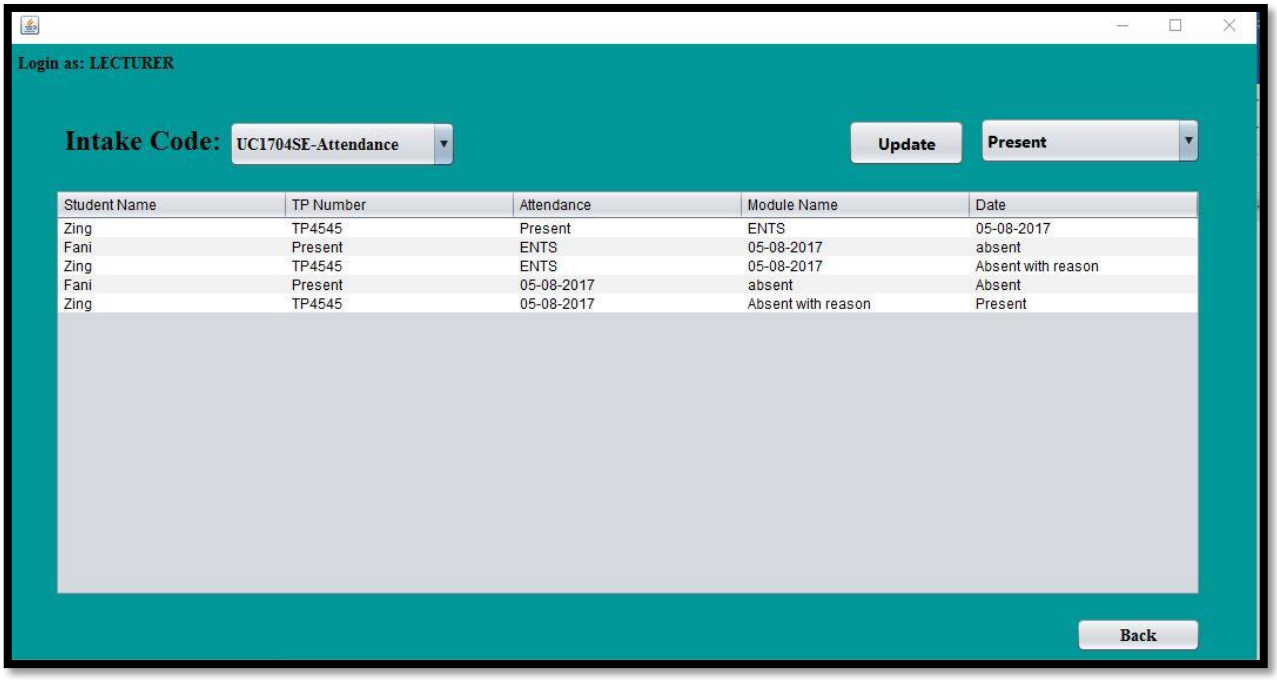

Figure 52: Modify Attendance

After pressing the Modify Attendance button, this attendance form will come here lecturer can view the intake wise student attendance by selecting the intake code as well as they can update the attendance of the student by giving a present, absent or late. The lecturer can use the back button to go back to the Lecturer Menu again.

6.12 View Intake wise student

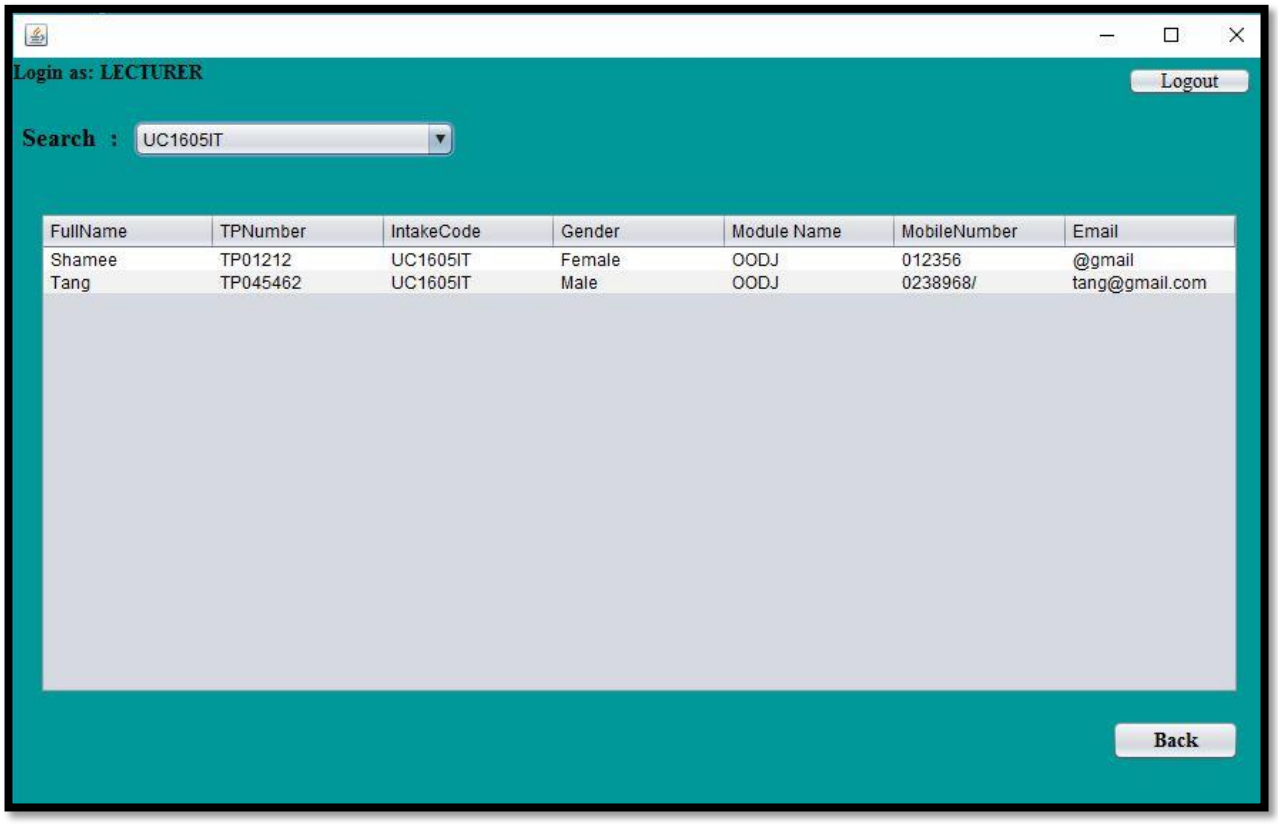

Figure 53: View Intake Wise Student

After selecting View Intake Wise Button this form will come, here lecturer can view the student record base on their intake and they can log out from this page and go to the login page.

6.13 Student menu

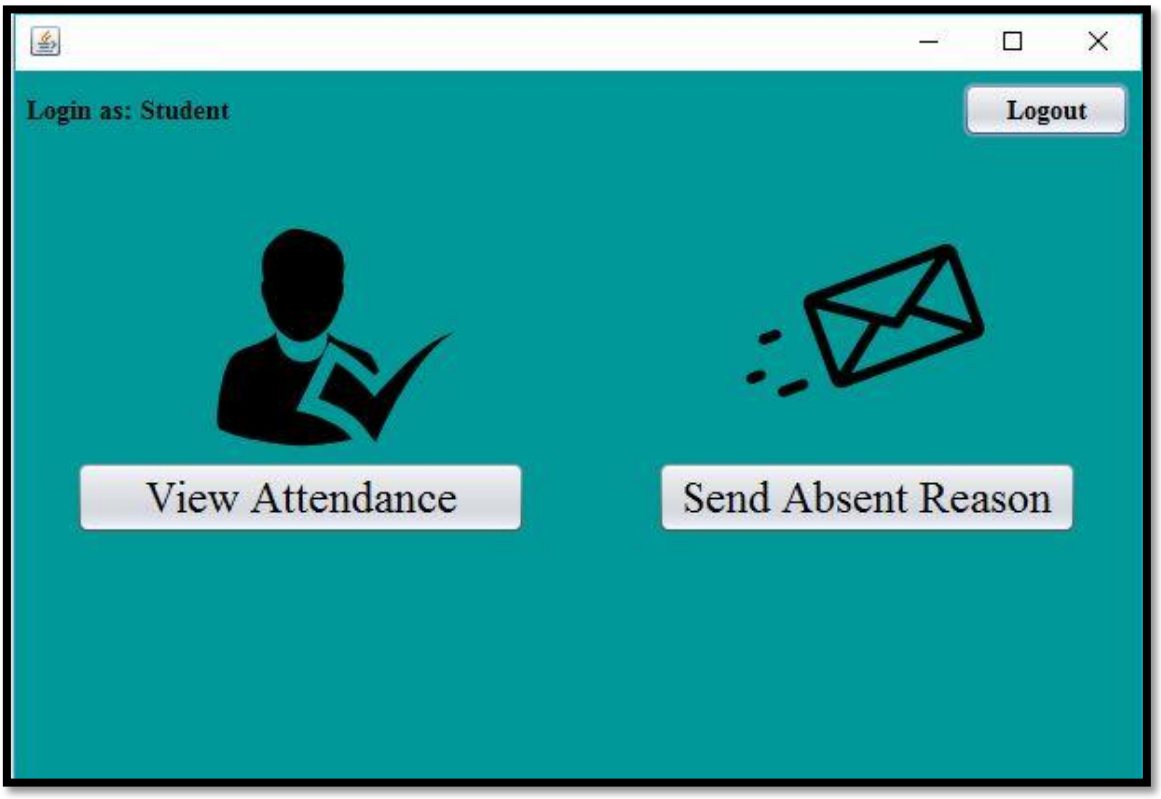

Figure 54: Student Menu

After login into the system student will get this page. Here student can only use two functions. One is to view attendance and the other one is to they can send the reason for the absent to the admin. Moreover, they use the log out button to go back to the login page.

6.14 View Attendance

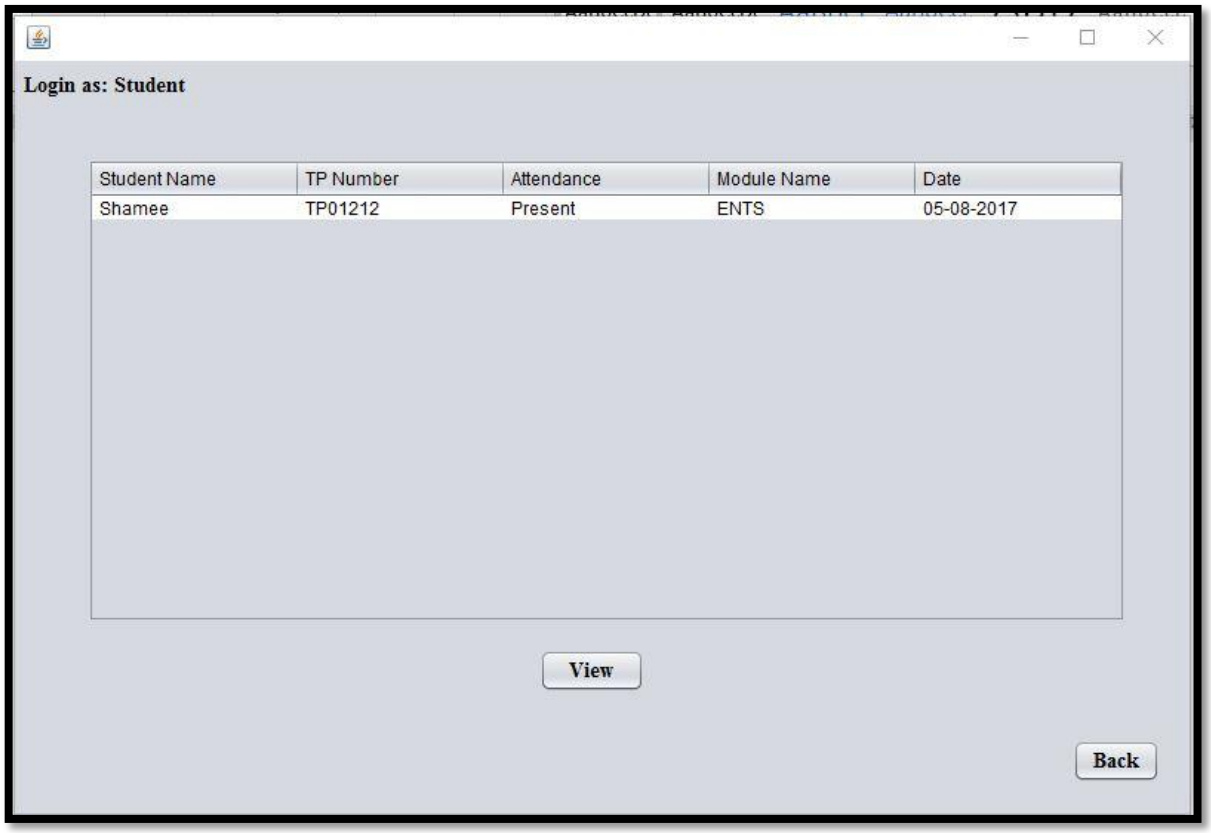

Figure 55: Student

After selecting view attendance, the student can view their own attendance only by pressing View button.

6.15 Send Absent Reason

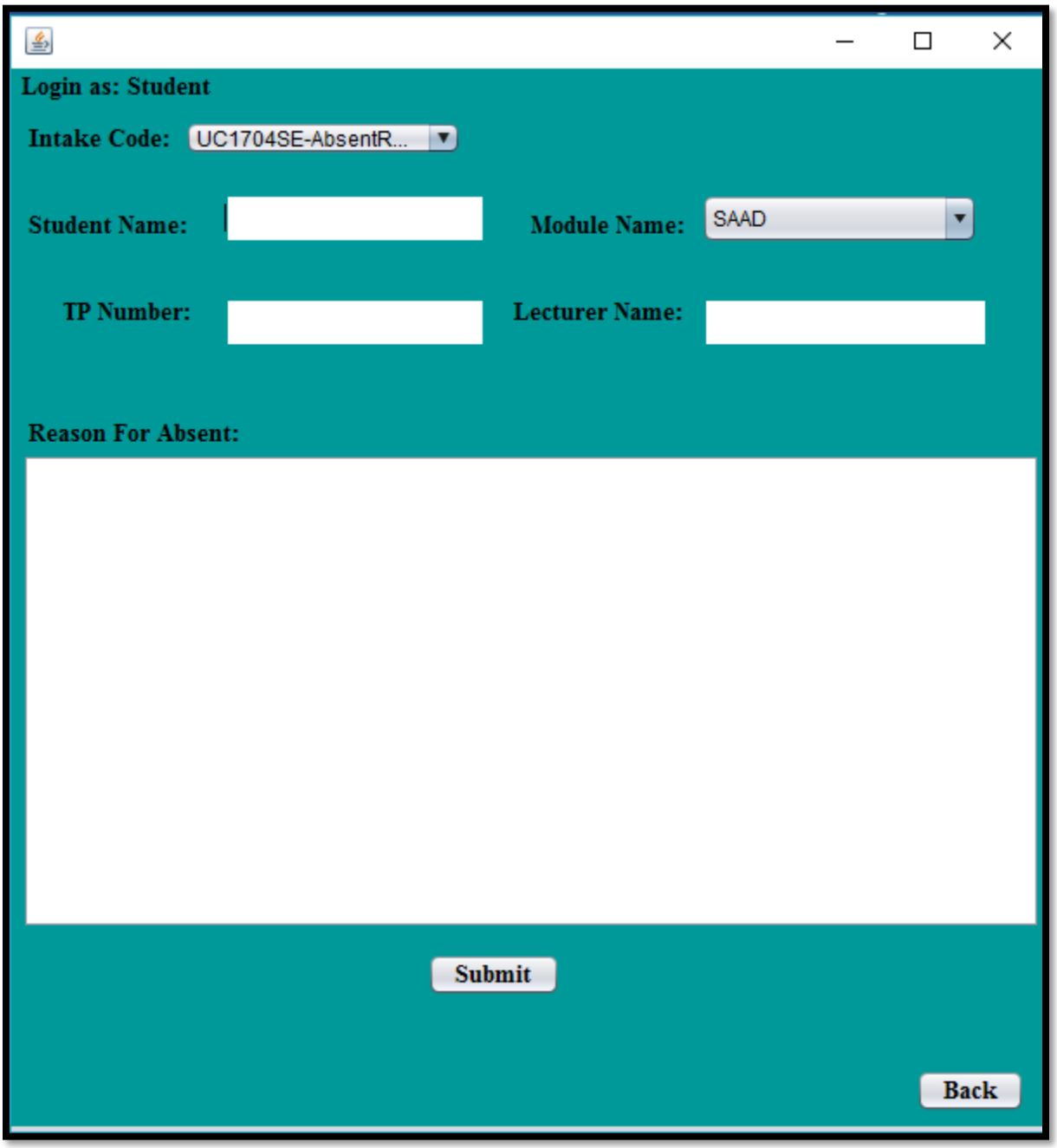

Figure 56: Send Absent Reason

After selecting the absent reason button student will get this form, here they have to select their intake code, module name, student name, TP Number, and lecturer name to submit the reason for their absence.

7.0 Assumptions

This Attendance Management System should be able to take daily attendance of the student and record it to the database that is text file so in future all the record can be seen anytime. This software will reduce the paper base work for the XYZ University due to all the student and lecturer record will be saved into the system and admin can view their record by using their name or ID/TP Number instead of searching in the file where all the paper based record are collected and also admin can able to modify, delete record if necessary. Student do not need to come to the admin office and talk to the admin about their absent reason because student can able to send the absent reason to the admin anytime so it will save the time for both student and admin as well as they do not need to go to the lecturer for checking their attendance because by using this system they can able to view the own daily attendance for each module. The lecturer can able to take the student attendance within less time instead of taking all the attendance on the paper and also if any attendance record is absent instead of the present by mistake than a lecturer can easily modify the attendance by using this system and the record will be updated to the database.

8.0 Conclusion

As a concluded from the above discussion that a secure, reliable, faster and an efficient system has been developed that replacing the manual work in the XYZ University. This system is highly flexible and reliable to use as well as it will give the accurate detail of the record that save into the system.

Since it will save time, the amount of work and reduces paper based work that will reduce the environmental impact. Moreover, this system is provided the expected result for that it is developed but still there is some function that needs to be improved

9.0 References

Inheritance (2017). *Inheritance in Java - GeeksforGeeks*. [online] GeeksforGeeks. Available at: http://www.geeksforgeeks.org/inheritance-in-java/ [Accessed 6 Aug. 2017].

java (2017). *Looping statement in Java*. [online] Sitesbay.com. Available at: http://www.sitesbay.com/java/java-looping-statement [Accessed 1 Aug. 2017].

class (2017). *Java Object and Classes*. [online] www.tutorialspoint.com. Available at: https://www.tutorialspoint.com/java/java_object_classes.htm [Accessed 2 Aug. 2017].

variables (2017). *Variables and Data Types in Java - Javatpoint*. [online] www.javatpoint.com. Available at: https://www.javatpoint.com/variable-datatype [Accessed 4 Aug. 2017].

www.javatpoint.com. (2017). *Variables and Data Types in Java - Javatpoint*. [online] Available at: https://www.javatpoint.com/variable-datatype [Accessed 2 Aug. 2017].

Java (2017). *Definition of a Java Variable (With an Example)*. [online] ThoughtCo. Available at: https://www.thoughtco.com/variable-2034325 [Accessed 7 Aug. 2017].

Method (2017). *Methods and Method Declaration in Java - dummies*. [online] dummies. Available at: http://www.dummies.com/programming/java/methods-and-method-declaration-in-java/ [Accessed 4 Aug. 2017].

Java Language (2017). *Most Significant Advantages of Java Language*. [online] Streetdirectory.com. Available at: http://www.streetdirectory.com/travel_guide/114362/programming/most_significant_advantages_of_java_language.html [Accessed 2 Aug. 2017].

C-sharpcorner.com. (2017). *Advantages of NetBeans IDE*. [online] Available at: http://www.c-sharpcorner.com/UploadFile/fd0172/how-to-create-celsius-to-fahrenheit-converter-app-and-showin/ [Accessed 3 Aug. 2017].