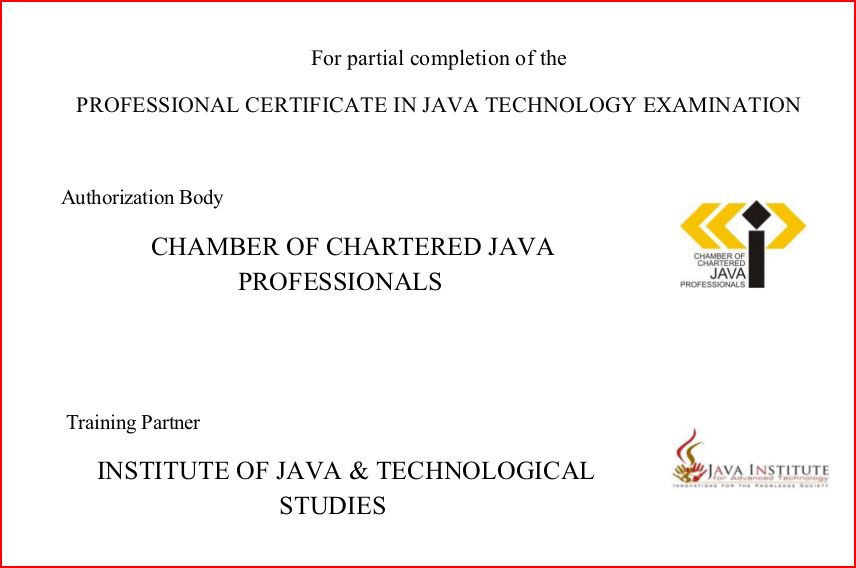


**JSAD Final Project Documentation**

Completed By

Batch: 27

920971768v  
Ishan Uthpal Samarasinghe  
Occult Technologies



Contents

* [Acknowledgement 3](#_Toc360471518)
* [Introduction 4](#_Toc360471519)
* [System Analysis 5](#_Toc360471520)
* [Scope Of Proposed System 6](#_Toc360471521)
* [System Design 7](#_Toc360471522)
  + [Usecase Diagram of Receptionist 7](#_Toc360471523)
  + [Usecase Diagram of Manager 8](#_Toc360471524)
  + [Activity Diagrams Common to Receptionist & Manager 9](#_Toc360471525)
    - [Marking Attendance 9](#_Toc360471526)
    - [Receive & Mark Class Fees 10](#_Toc360471527)
    - [Register Students 11](#_Toc360471528)
    - [View Timetables 12](#_Toc360471529)
  + [Activity Diagrams of Manager 13](#_Toc360471530)
    - [Arrange Timetables 13](#_Toc360471531)
    - [Make Payments 14](#_Toc360471532)
    - [Manage Users 15](#_Toc360471533)
    - [Register Tutors 16](#_Toc360471534)
    - [View All Data Sheets 17](#_Toc360471535)
* [Source Code 18](#_Toc360471536)
* [Database Schema 69](#_Toc360471537)
* [Notes 70](#_Toc360471538)

# Acknowledgement

F

irst and foremost, I would like to thank the lecturer of PCJT course, Mr. Anushka Tennakoon and Mr.Ishara for the valuable guidance and advice. They inspired me highly to work in this Project. Their willingness to motivate, contributed tremendously to my project.

I would also like to thank staff of Java Institute for showing examples that relate to the topic of the project. Furthermore, I would like to thank the authority of Java Institute for providing a good environment and facilities to complete this Project.

Finally, an honorable mention goes to my parents and friends for their Understanding and support in completing this project. Without help of the Particular mentioned above, I would have faced many difficulties while doing this Project.

# Introduction

M

y JavaSE Project is based for the Institute Management of London Bridge Private Institute, Kandy. The purpose of this project is to facilitate their work with a new computerized database, which provides management with details concerning the students and tutors as well as the transaction of students and tutors that took place.

The new computerized database will create a better system where information can be added, modified and updated easily. With this system it would decrease the problems of the prevailing methods. This whole system is designed in Java and the database is designed by MySQL.

# System Analysis

Problems In Existing System

As we know manual systems are quiet tedious, time consuming and less efficient and accurate in comparison to the computerized system. So following are some disadvantages of the old system.

* Time consuming
* Lot of paper work

User Requirements : Functional Requirements

**Input / Output**

* Managing student information.
* Managing tutors’ information.
* Recording all payments to and from institute.

**Processing**

* Calculating the payments and due according to input fees.
* Database backup/restore

**Error handling**

* Reporting invalid login tries.
* Reporting invalid input data errors.

# Scope Of Proposed System

The scope of this system is to provide user efficient working environment. This system provides user friendly interfaces resulting in knowing each and every usability features of the system. It completes the work in a very less time, resulting in less time consumption and high level of efficiency.

The system is developed in such a way that a naïve user can also operate the system easily. The calculations are made very quickly and the records are directly saved into database and database can be maintained for a long period of time. Each record can be retrieved and verified for the future transactions.

Also this system provides high level of security for data leaking as only admin people can access the database and even if a hacker tries to break passwords, it is perfectly impossible to get done because of the BCrypt Technology used in password hashing.

**System Requirements**

Hardware Requirements

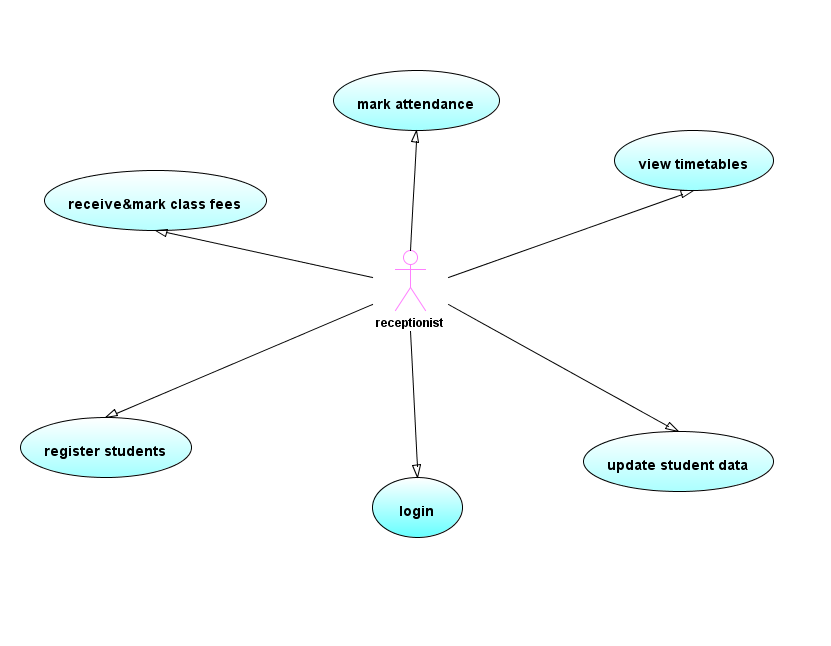
* Processor :- Pentium D and upwards
* RAM :- Minimum 750MB
* Hard disk :- Minimum 10GB

Software Requirements

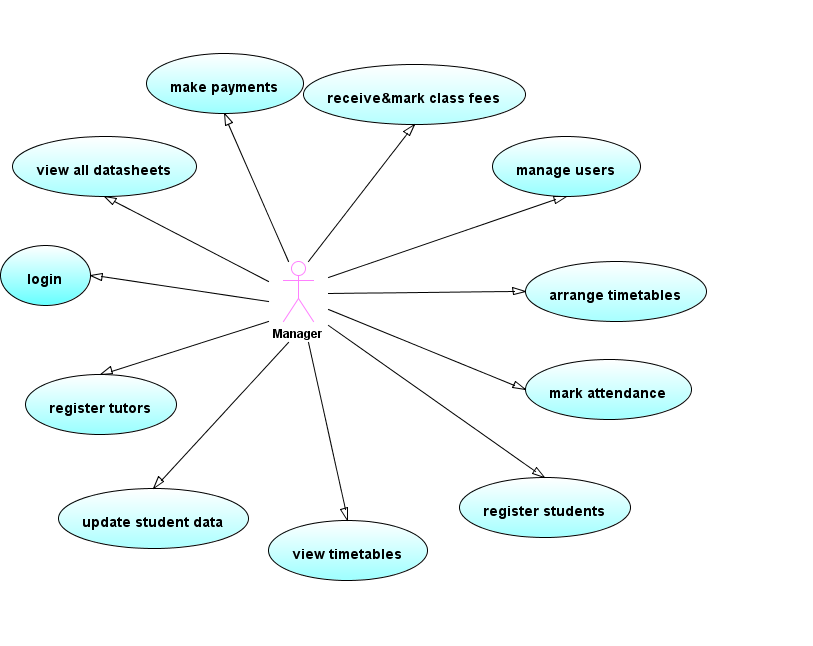
* Operating Systems :- MS Windows / Any Linux Distribution
* Frontend : Java Runtime Environment 1.7+
* Backend : MySQL 5.0+

# System Design­

## Usecase Diagram of Receptionist

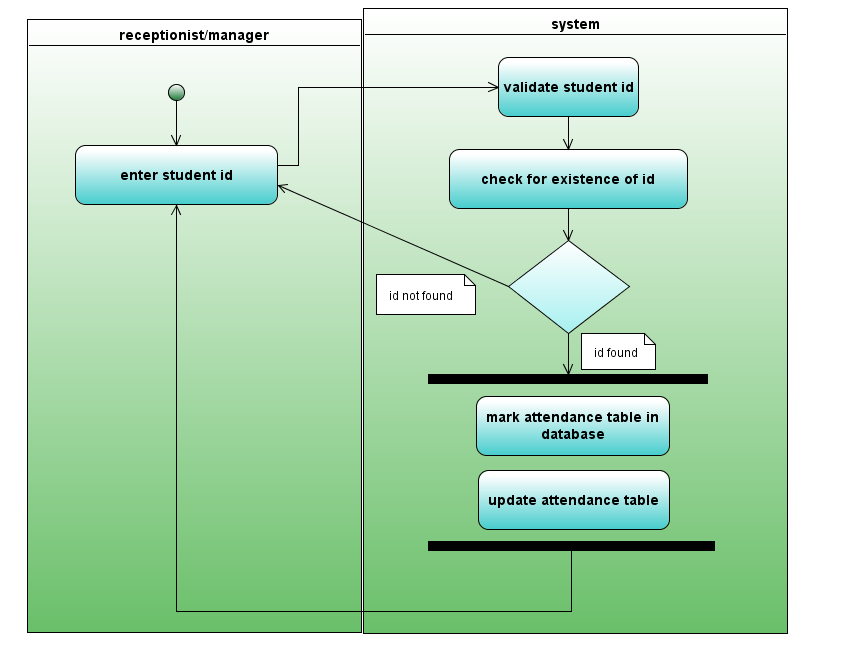
****

## Usecase Diagram of Manager

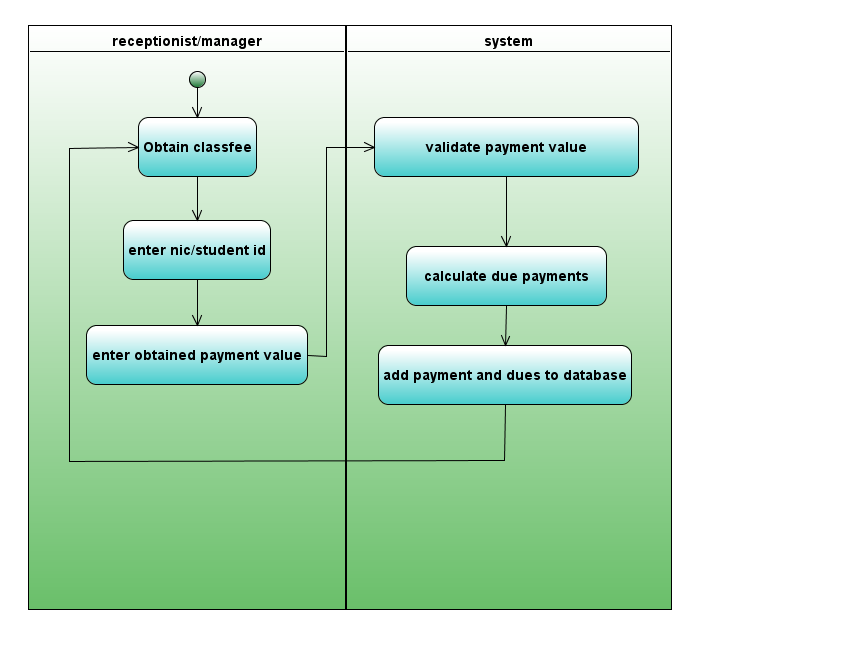
****

## Activity Diagrams Common to Receptionist & Manager

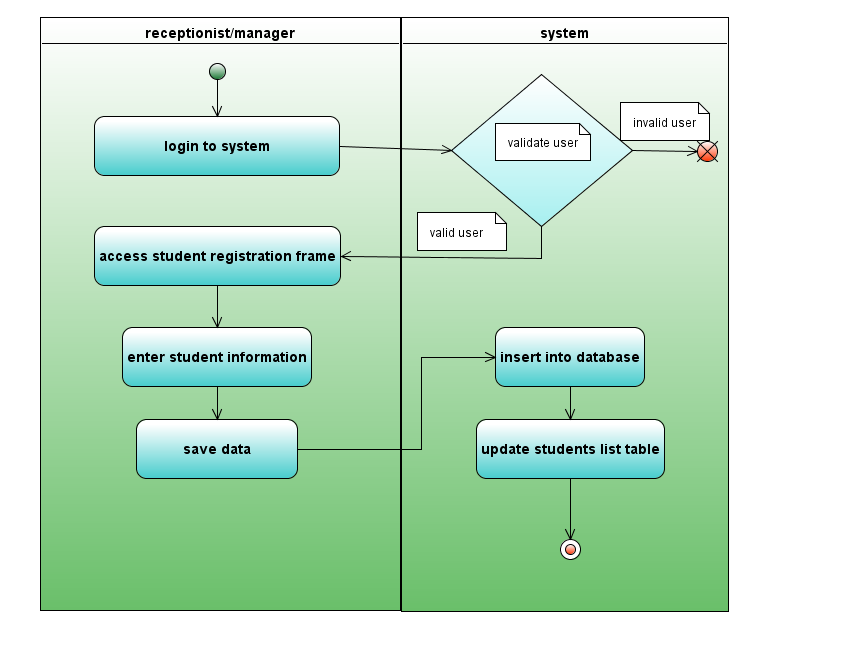
### Marking Attendance

****

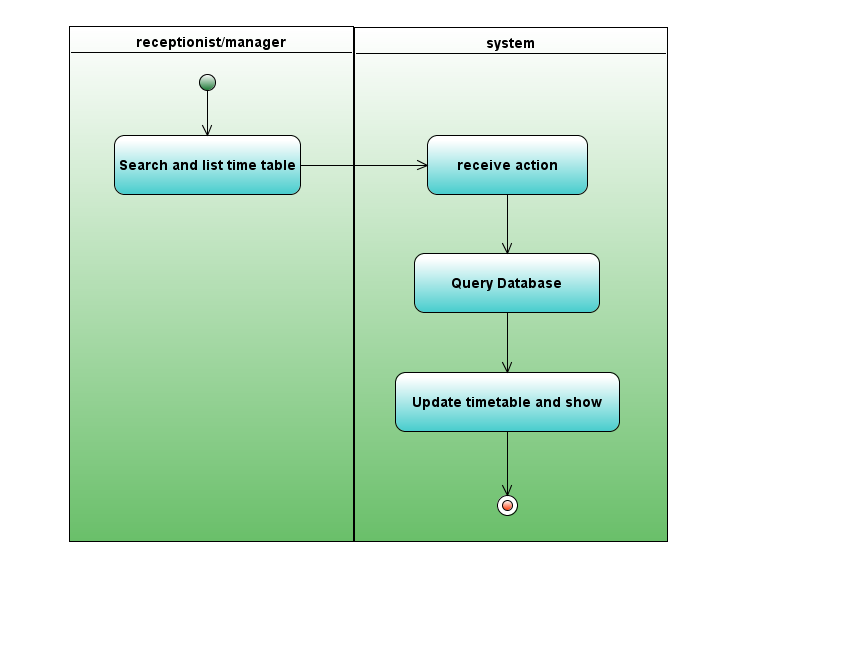
### Receive & Mark Class Fees

****

### Register Students

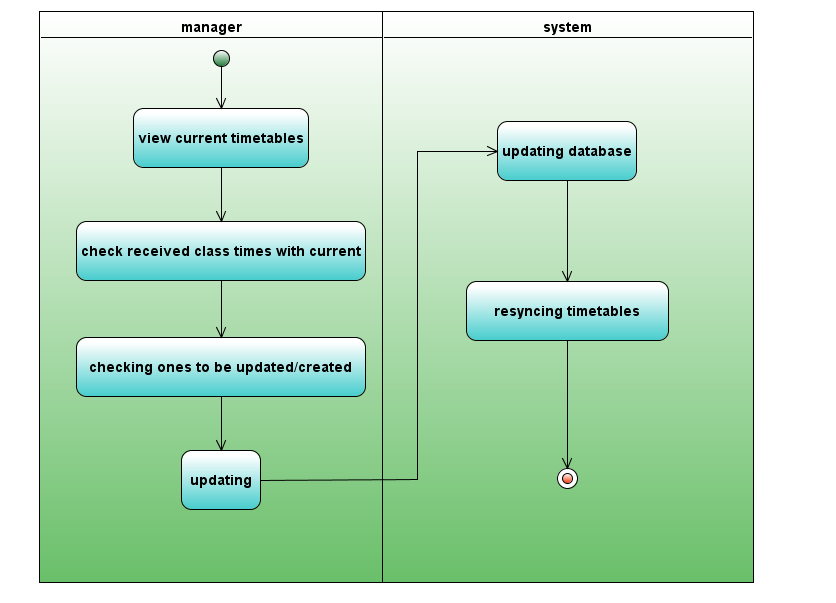
****

### View Timetables

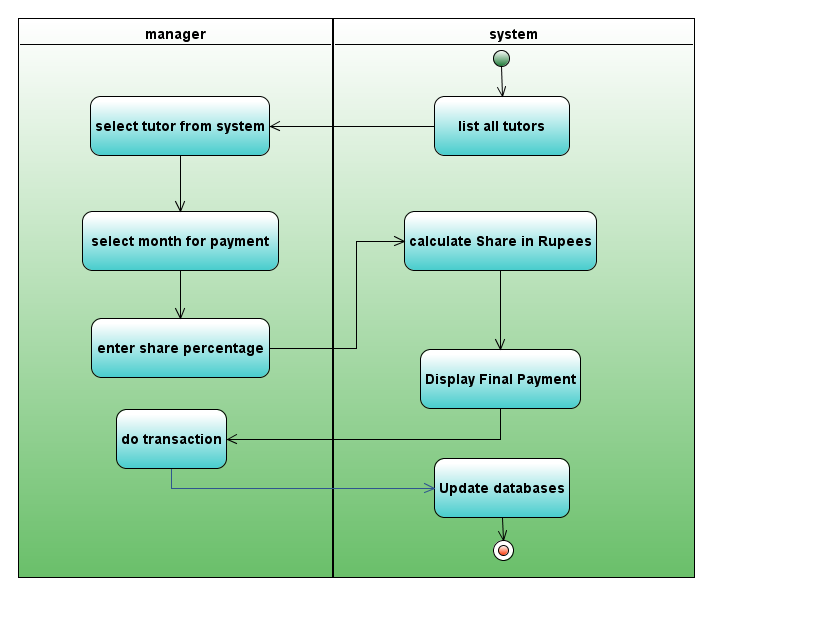
****

## Activity Diagrams of Manager

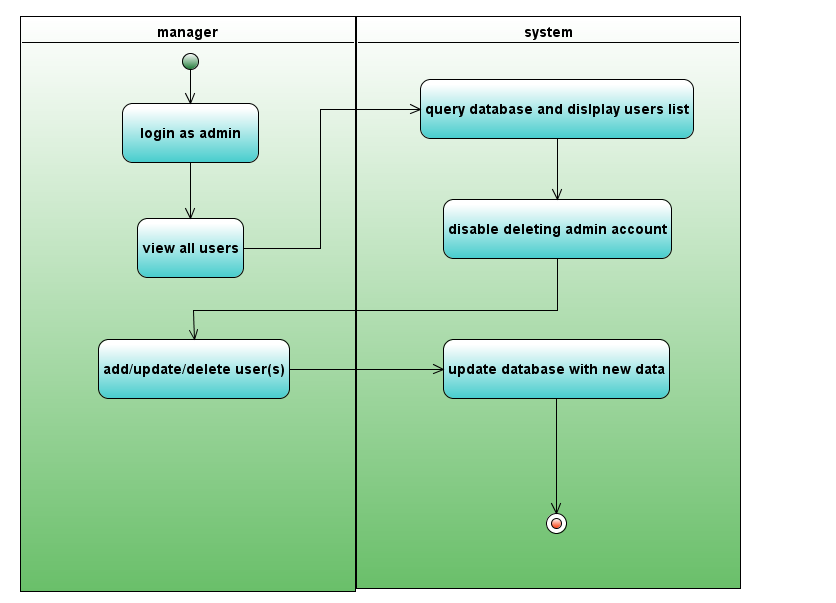
### Arrange Timetables

****

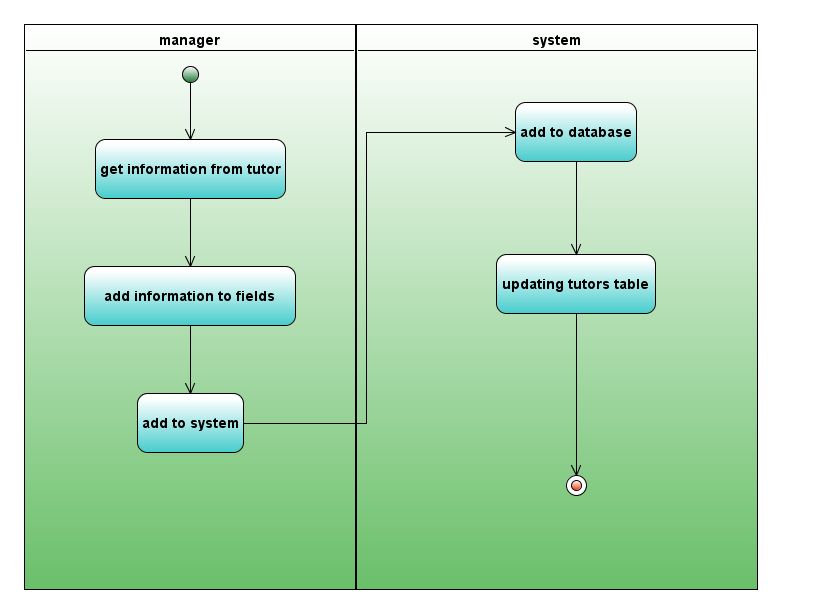
### Make Payments

****

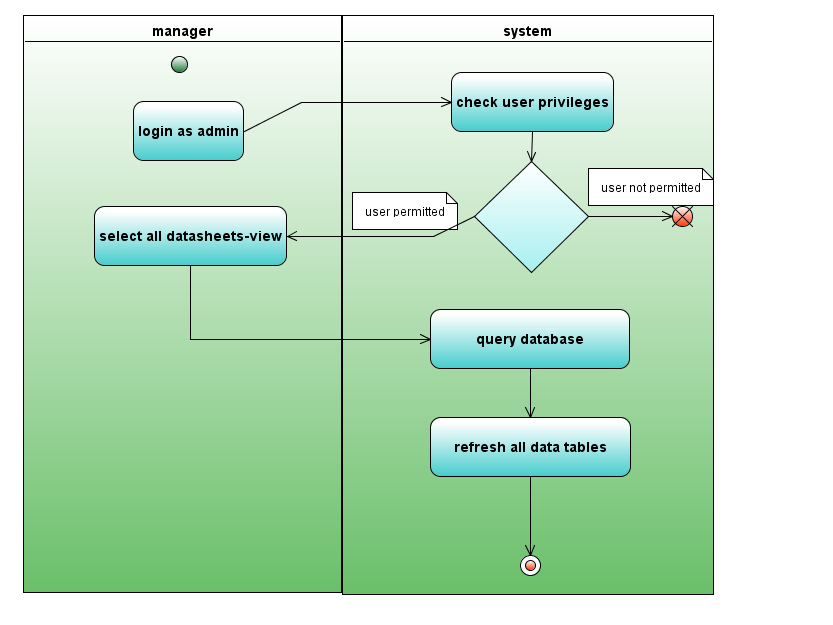
### Manage Users

****

### Register Tutors

****

### View All Data Sheets

****

# ­­­­­­Source Code

****

**Login**

package LOGIN;

import Connection.jdbc;

import jBCrypt.BCrypt;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

import javax.swing.UIManager;

import javax.swing.plaf.nimbus.NimbusLookAndFeel;

import london.newHome;

/\*\*

\* @author joshua

\*/

public final class login extends javax.swing.JFrame {

newHome H = null;

jdbc cdb = new jdbc();

//<editor-fold defaultstate="collapsed" desc="staticint 'loginTries' to count">

static int loginTries = 1;

//</editor-fold>

private String userAgent = null;

public login() {

initComponents();

} private void l\_exitMouseClicked(java.awt.event.MouseEvent evt) {

System.exit(0);

}

private void t\_passKeyPressed(java.awt.event.KeyEvent evt) {

runByEnter(evt);

}

private void t\_userKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 10) {

t\_pass.grabFocus();

} }

private void l\_ch\_passwordMouseClicked(java.awt.event.MouseEvent evt) {

newUser();

}

private void jLabel5MouseClicked(java.awt.event.MouseEvent evt) {

JOptionPane.showMessageDialog(null, "<html><p align=right><b><font color=red>Please contact Administrator/Owner of the institute</b></font>&nbsp;&nbsp;"

+ " <br>to get help with your situation"

+ " <br><b>or Contact Systems Administrator</b></p></html>.", "Help", JOptionPane.OK\_CANCEL\_OPTION);

}

private void newUser() {

//---BUGGED METHOD :: if the first pass field is empty for first time, it redisplays,

// but if its empty for 2nd time, nothing happens

try {

JPasswordField pwd = new JPasswordField(20);

JPasswordField pwd2 = new JPasswordField(20);

setState(ICONIFIED);//----minimize login window

if (t\_user.getText().equals("") | new String(t\_pass.getPassword()).equals("")) {

//---------setting error if no username/password is provided

JOptionPane.showMessageDialog(null, "Enter both of your current username AND password ",

"Error", JOptionPane.OK\_OPTION);

dispose();

new login().setVisible(true);

} else { //----------start new password process :)

setState(ICONIFIED);//----minimize login window

setLocation(100, 100);

int getpwd0 = JOptionPane.showConfirmDialog(null, pwd, "Enter New Password for user: " + t\_user.getText(), JOptionPane.OK\_CANCEL\_OPTION);

if (getpwd0 == JOptionPane.CANCEL\_OPTION) {

//----press cancel and exit

System.exit(0);

} // gave first password field to enter

if (new String(pwd.getPassword()).equals("")) {

//--------if first password field is empty, giving an error message

JOptionPane.showMessageDialog(null, "Empty password : \nyou cannot set an Empty password",

"Error", 2);

setState(NORMAL);

t\_user.setText(null);

// if empty, gave an error message

} else if (!new String(pwd.getPassword()).equals("")) {

int getpwd2 = JOptionPane.showConfirmDialog(null, pwd2, "Confirm Password",

JOptionPane.OK\_CANCEL\_OPTION);

if (getpwd2 == JOptionPane.CANCEL\_OPTION) {

//----press cancel and exit

System.exit(0);

}

if (new String(pwd2.getPassword()).equals("")) {

JOptionPane.showMessageDialog(null, "Empty password : \nyou cannot set an Empty password",

"Error", 2);

int getpwd3 = JOptionPane.showConfirmDialog(null, pwd2, "Confirm Password",

JOptionPane.OK\_CANCEL\_OPTION);

if (getpwd3 == JOptionPane.CANCEL\_OPTION) {

System.exit(0);

} } else //-----------------------------------------------------------------------------

//if both are filled, checking passwords' compatibility

//if (new String(pwd.getPassword()) == new String(pwd2.getPassword())) {

//--------------above 'if' replaced by following null check ternary .equals()

//if (new String(pwd.getPassword()) == null ? new String(pwd2.getPassword()) == null : new String(pwd.getPassword()).equals(new String(pwd2.getPassword()))) {

// update 2013\_06\_19@5.25pm - null check removed coz expression is never null

if (new String(pwd.getPassword()).equals(new String(pwd2.getPassword()))) {

// if matched, updating database

cdb.putdata("UPDATE users SET password='" + new String(pwd2.getPassword()) + "' "

+ "WHERE username='" + t\_user.getText() + "'");

JOptionPane.showMessageDialog(null, "Password updated successfully.\n Please login to Continue");

System.exit(0);

} else {

JOptionPane.showMessageDialog(null, "Passwords Do not Match\n Please try again");

} } } // }

} catch (Exception e) {

System.out.println(e);

} }

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

try { UIManager.setLookAndFeel(new NimbusLookAndFeel());

new login().setVisible(true);

} catch (Exception e) {

System.out.println(e);

System.out.println("at login face");

} } }); }

private void runByEnter(java.awt.event.KeyEvent evt) {

ResultSet r = null;

if (evt.getKeyCode() == 10) {

//------------when enter is pressed, if admin admin is given,go home..

//else get the strings and display a message,..if the error count goes more than 5--system-exit blah

try { r = cdb.getdata("SELECT \* from users WHERE username='" + t\_user.getText() + "' ");

if (r.first()) {

if (BCrypt.checkpw(new String(t\_pass.getPassword()), r.getString("password"))) {

//------------------old written procedure continues...........

if (r.getString("username").equals("admin")) {

H = new newHome();

this.dispose();

H.titleFromLogin(r.getString("username"));

H.setVisible(true);

H.admin();

setUserAgent("ADMIN");

} else {

H = new newHome();

this.dispose();

H.titleFromLogin(r.getString("username"));

H.setVisible(true);

setUserAgent(r.getString("username"));

}

cdb.putdata("insert into login\_sessions(users\_uid,users\_username,status) "

+ " values('" + r.getInt("uid") + "','" + r.getString("username") + "','Success') ");

} else {

JOptionPane.showMessageDialog(null, "<html><font color=red><b>"

+ "Incorrect login attempt(s) detected</html></font></b>", "Warning !", 1, null);

t\_pass.setText(null);

t\_user.setText(null);

loginTries++;

cdb.putdata("insert into login\_sessions(users\_uid,users\_username,status) "

+ " values('" + r.getInt("uid") + "','" + r.getString("username") + "','Error\_Try') ");

//--------login session opened and recorded

if (loginTries == 6) { //--------- int loginTries is (global) (instance) static

System.exit(0);

JOptionPane.showMessageDialog(null, "You have not proven to be a valid user!"

+ " \n The System will restart now");

Runtime sys = Runtime.getRuntime();

sys.exec("shutdown -r -f -t 00");

//sys.exec("$RESTART$");

} } } }

catch (Exception ex) {

System.out.println(ex);

} finally { try { if (r != null) { r.close(); }

} catch (SQLException ex) {

System.out.println(ex);

} } } }

public void setUserAgent(String userAgent) {

this.userAgent = userAgent;

}

public String getUserAgent() {

// method is used in 'in\_compensation' class in setting the user performing the action

return userAgent;

}}

**Logout**

public class logout {

public void log() {

ResultSet r2 = null;

try {

String d = new Date().toString();

int y = new Date().getYear() + 1900;

int m = new Date().getMonth() + 1;

String da = d.split(" ")[2];

String date = y + "-" + m + "-" + da;

String s = " " + d.split(" ")[3];

r2 = jdbc.con().createStatement()

.executeQuery("SELECT \* from login\_sessions order by idlogin\_sessions desc limit 1");

if (r2.first()) {

jdbc.con().createStatement().executeUpdate("update login\_sessions "

+ " set logoutAt='" + date + s + "' "

+ " where idlogin\_sessions='" + r2.getInt("idlogin\_sessions") + "' ");

} } catch (Exception ex) {

System.out.println("" + ex);

} finally {

try {

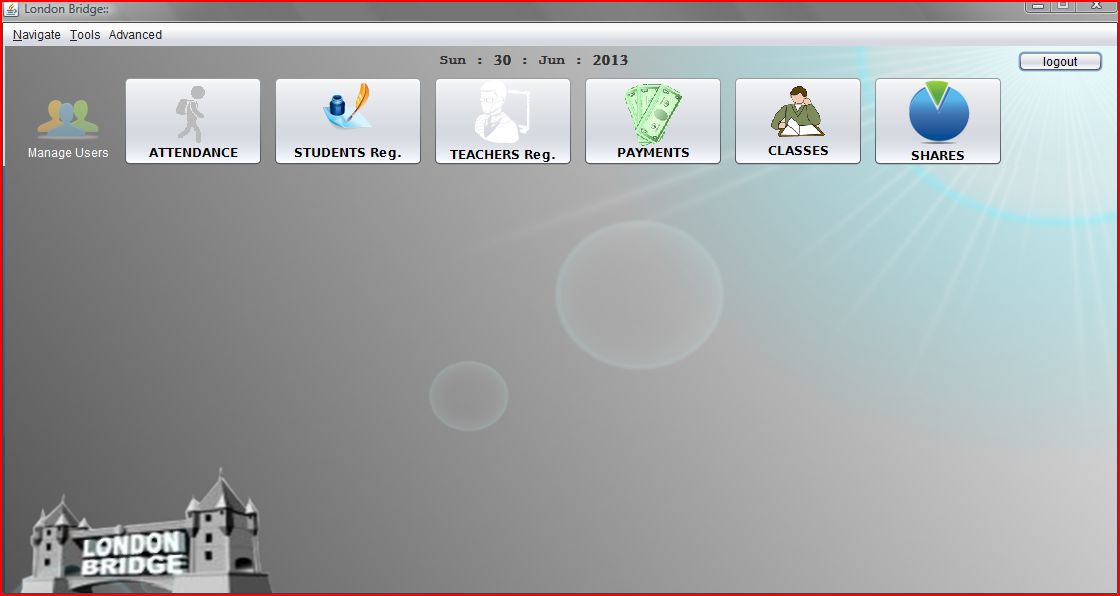
if (r2 != null) {

r2.close();

} } catch (SQLException ex) {

System.out.println(ex);

} } }}

**Home**

package london;

import in\_frames.In\_attendance;

import in\_frames.In\_classes\_times;

import in\_frames.In\_payments;

import in\_frames.In\_stu\_reg;

import in\_frames.In\_tea\_reg;

import in\_frames.in\_compensation;

import java.util.Date;

import javax.swing.JOptionPane;

import javax.swing.JPasswordField;

import javax.swing.UIManager;

import javax.swing.UnsupportedLookAndFeelException;

import javax.swing.plaf.nimbus.NimbusLookAndFeel;

import london.rescumanager.RescueManager;

public class newHome extends javax.swing.JFrame {

themes theme = new themes();

newHome nhome;

public newHome() {

initComponents();

date();

titleFromLogin("");

}

public void killall() {

this.dispose();

System.gc();

new LOGIN.login().setVisible(true);

}

private void date() {

String d = new Date().toString();

String day = d.split(" ")[0];

String y = d.split(" ")[5];

String m = d.split(" ")[1];

String da = d.split(" ")[2];

String date = day + " : " + da + " : " + m + " : " + y;

l\_date.setText(date);

}

private void battendanceActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

attendance();

}

private void bpaymentsActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

payments();

}

private void bteachersActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

tea();

}

private void bcompensationActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

compensation();

}

private void bregisterActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

stu();

}

private void bclassesActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

classes();

}

private void busermanActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

JPasswordField pwd = new JPasswordField(20);

int getpwd0 = JOptionPane.showConfirmDialog(null, pwd, "Enter your password: ",

JOptionPane.OK\_CANCEL\_OPTION);

if (getpwd0 == JOptionPane.OK\_OPTION) {

if (new String(pwd.getPassword()).equals("admin")) {

System.gc();

this.setState(ICONIFIED);

new user\_manage().setVisible(true);

} else {

JOptionPane.showMessageDialog(null, "You are not the owner/administrator \n"

+ "thus you are not allowed for user management ", "Warning", 2);

} } }

private void blogoutActionPerformed(java.awt.event.ActionEvent evt) {

new LOGIN.logout().log();

killall();

}

private void jMenuItem2ActionPerformed(java.awt.event.ActionEvent evt) {

int showConfirmDialog = JOptionPane.showConfirmDialog(null, "<html><font color=red>"

+ "Are you sure about the process?? </font></html>");

if (showConfirmDialog == JOptionPane.OK\_OPTION) {

this.dispose();

System.gc();

new RescueManager().setVisible(true);

} }

private void jMenuItem16ActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

System.exit(0);

}

private void logoutFromMenuActionPerformed(java.awt.event.ActionEvent evt) {

new LOGIN.logout().log();

killall();

}

private void login\_menuActionPerformed(java.awt.event.ActionEvent evt) {

System.gc();

new logins().setVisible(true);

}

In\_stu\_reg s;//--------------student reg object initiallized globally

int i = 0; //------------count and maintain visibility of the internal frame

private void stu() {

s = new In\_stu\_reg();

if (i >= 1) {

s.dispose();

jDesktopPane1.remove(s);

i = 0;

jDesktopPane1.add(s);

s.setVisible(true);

i++;

} else if (i == 0) {

i++;

jDesktopPane1.add(s);

s.setVisible(true);

} }

In\_tea\_reg t; //-----------teacher-reg object initiallized globally

int j = 0; //------------count and maintain visibility of the internal frame

private void tea() {

t = new In\_tea\_reg();

if (j >= 1) {

t.dispose();

jDesktopPane1.remove(t);

j = 0;

jDesktopPane1.add(t);

t.setVisible(true);

j++;

} else if (j == 0) {

jDesktopPane1.add(t);

t.setVisible(true);

j++;

} }

In\_attendance a;

int k = 0;

private void attendance() {

a = new In\_attendance();

if (k >= 1) {

a.dispose();

jDesktopPane1.remove(a);

k = 0;

jDesktopPane1.add(a);

a.setVisible(true);

k++;

} else if (k == 0) {

jDesktopPane1.add(a);

a.setVisible(true);

k++;

} }

In\_classes\_times clst;

int c = 0;

private void classes() {

clst = new In\_classes\_times();

if (c >= 1) {

clst.dispose();

jDesktopPane1.remove(clst);

c = 0;

jDesktopPane1.add(clst);

clst.setVisible(true);

c++;

} else if (c == 0) {

jDesktopPane1.add(clst);

clst.setVisible(true);

c++;

} }

in\_frames.in\_compensation comps;

int co = 0;

private void compensation() {

comps = new in\_compensation();

if (co >= 1) {

comps.dispose();

jDesktopPane1.remove(comps);

co = 0;

jDesktopPane1.add(comps);

comps.setVisible(true);

co++;

} else if (co == 0) {

jDesktopPane1.add(comps);

comps.setVisible(true);

co++;

} }

in\_frames.In\_payments pay;

int pa = 0;

private void payments() {

pay = new In\_payments();

if (pa >= 1) {

pay.dispose();

jDesktopPane1.remove(pay);

pa = 0;

jDesktopPane1.add(pay);

pay.setVisible(true);

pa++;

} else if (pa == 0) {

jDesktopPane1.add(pay);

pay.setVisible(true);

pa++;

} }

public void admin() {//--------------enabling user management

buserman.setEnabled(true);

login\_menu.setEnabled(true);

System.gc();

}

public void refreshui() {

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

new newHome().repaint();

System.gc();

} }); }

public static void main(String args[]) {

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

try {

UIManager.setLookAndFeel(new NimbusLookAndFeel());

} catch (UnsupportedLookAndFeelException ex) {

System.out.println(ex);

}

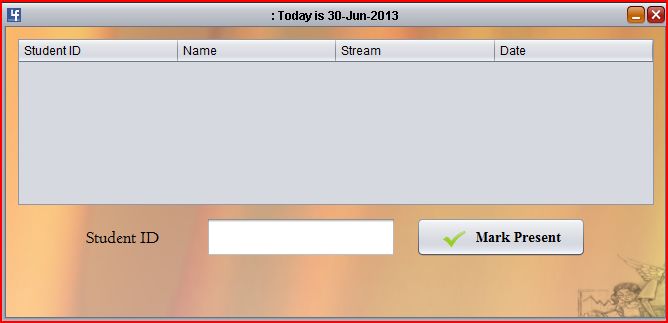
new newHome().setVisible(true);

// new newHome().framefunctions();

} }); }

public final void titleFromLogin(String name) {

System.gc(); this.setTitle(this.getTitle().concat(name)); }}

**Attendance**

package in\_frames;

import Connection.jdbc;

import java.sql.ResultSet;

import java.util.Date;

import java.util.Vector;

import javax.swing.JOptionPane;

import javax.swing.table.DefaultTableModel;

public class In\_attendance extends javax.swing.JInternalFrame {

jdbc cdb = new jdbc();

public In\_attendance() {

initComponents();

loadTable();

t\_atten\_id.grabFocus();

}

private void t\_atten\_idKeyPressed(java.awt.event.KeyEvent evt) {

DefaultTableModel dtm = (DefaultTableModel) table\_atte.getModel();

ResultSet rs2 = null, rs1 = null;

try {

if (evt.getKeyChar() == 10) {//-----marking student with the subject(s) on this date as present

String date = new Date().toString();

String datef = date.replaceAll("[^\\x00-\\x7F]", "");

String y = date.split(" ")[5];

String m = date.split(" ")[1];

String da = date.split(" ")[2];

String dall = m + da + y;

rs2 = cdb.getdata("select \* from attendance");

while (rs2.next()) {

if (rs2.getString("stu\_no").equals(t\_atten\_id.getText().toString())

&& rs2.getString("systemtime").equals(dall.toString())) {

JOptionPane.showMessageDialog(null, "The Student has already Entered the premises \n"

+ " and been marked present");

} else if (!rs2.getString("stu\_no").equals(t\_atten\_id.getText().toString())) {

if (!rs2.getString("systemtime").equals(dall.toString())) {

rs1 = cdb.getdata("SELECT name,stream FROM students2 "

+ "WHERE stu\_no='" + t\_atten\_id.getText() + "'");

while (rs1.next()) {

String name = rs1.getString("name");

String stream = rs1.getString("stream");

Vector v = new Vector();

v.add(t\_atten\_id.getText());

v.add(name);

v.add(stream);

v.add(datef);

dtm.addRow(v); cdb.putdata("INSERT INTO attendance(stu\_no,name,stream,date,systemtime) "

+ "VALUES ('" + t\_atten\_id.getText() + "','" + name + "',"

+ "'" + stream + "','" + datef + "','" + dall + "' )");

t\_atten\_id.setText("");

t\_atten\_id.grabFocus();

loadTable();

} } } }

} else if (evt.getKeyCode() == 32) {

t\_atten\_id.setEditable(false);

} else if (evt.getKeyCode() == 37 | evt.getKeyCode() == 39) {

t\_atten\_id.setEditable(true);

} } catch (Exception e) {

System.out.println("error at student id input field using enter key");

System.out.println(e);

} finally {

try {

if (rs2 != null) {

rs2.close();

}

if (rs1 != null) {

rs1.close();

} } catch (Exception ex) {

System.out.println(ex);

} }}

private void table\_atteKeyPressed(java.awt.event.KeyEvent evt) {

try {

if (evt.getKeyChar() == 127 | evt.getKeyChar() == 8) { //-------------confirm before deleting

JOptionPane.showMessageDialog(null, "YOU ARE NOT ALLOWED \n"

+ " TO CLEAR OR DELETE \n ATTENDANCE RECORDS", "WARNING", 2);

} else {

JOptionPane.showMessageDialog(null, "Invalid operation on table");

} } catch (Exception e) {

System.out.println(e);

} }

private void t\_atten\_idFocusLost(java.awt.event.FocusEvent evt) {

badd.grabFocus();

}

private void baddActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel dtm = (DefaultTableModel) table\_atte.getModel();

ResultSet rs1 = null, rs2 = null;

try {

String date = new Date().toString();

String datef = date.replaceAll("[^\\x00-\\x7F]", "");

String y = date.split(" ")[5];

String m = date.split(" ")[1];

String da = date.split(" ")[2];

String dall = m + da + y;

rs1 = cdb.getdata("SELECT name,stream FROM students2 WHERE stu\_no='" + t\_atten\_id.getText() + "'");

while (rs1.next()) {

String name = rs1.getString("name");

String stream = rs1.getString("stream");

rs2 = cdb.getdata("select \* from attendance");

if (rs2.first()) {

if (rs2.getString("stu\_no").equals(t\_atten\_id.getText().toString())

&& rs2.getString("systemtime").equals(dall.toString())) {

JOptionPane.showMessageDialog(null, "The Student has already Entered the premises \n"

+ " and been marked present");

} else {

Vector v = new Vector();

v.add(t\_atten\_id.getText());

v.add(name);

v.add(stream);

v.add(datef);

dtm.addRow(v);

cdb.putdata("INSERT INTO attendance(stu\_no,name,stream,date,systemtime) "

+ "VALUES ('" + t\_atten\_id.getText() + "','" + name + "',"

+ "'" + stream + "','" + datef + "','" + dall + "' )");

t\_atten\_id.setText("");

t\_atten\_id.grabFocus();

loadTable();

} } } } catch (Exception e) {

System.out.println("at student id input field from button");

System.out.println(e);

} finally {

try {

if (rs2 != null) {

rs2.close();

}

if (rs1 != null) {

rs1.close();

} } catch (Exception ex) {

System.out.println(ex);

} } }

private void t\_atten\_idFocusGained(java.awt.event.FocusEvent evt) {

loadTable();

}

private void loadTable() {

ResultSet rs = null;

String date = new Date().toString();

String y = date.split(" ")[5];

String m = date.split(" ")[1];

String da = date.split(" ")[2];

String dall = m + da + y;

this.setTitle("" + ": Today is " + da + "-" + m + "-" + y);

DefaultTableModel dtm = (DefaultTableModel) table\_atte.getModel();

while (dtm.getRowCount() > 0) {

dtm.removeRow(0);

} try {

rs = cdb.getdata("select \* from attendance where systemtime='" + dall + "'");

while (rs.next()) {

Vector v = new Vector();

v.add(rs.getString("stu\_no"));

v.add(rs.getString("name"));

v.add(rs.getString("stream"));

v.add(rs.getString("date"));

dtm.addRow(v);

} this.repaint();

} catch (Exception e) {

System.out.println(e);

} finally {

try {

if (rs != null) {

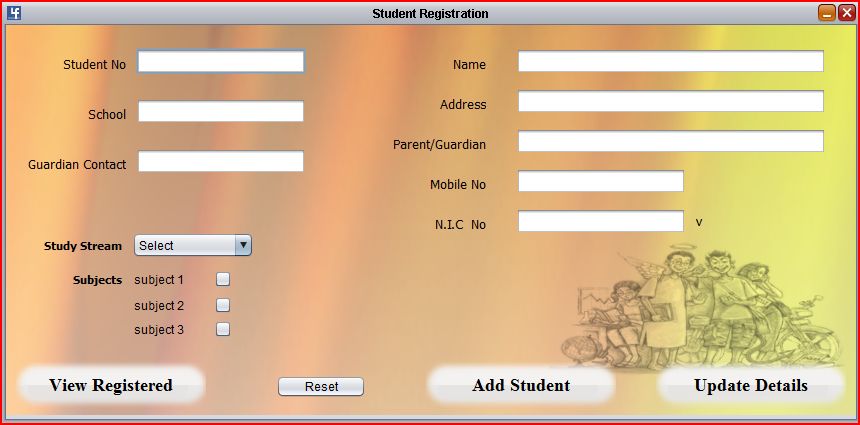
rs.close();

} } catch (Exception ex) {

System.out.println(ex);

} } }

**Student Registration**



package in\_frames;

import Connection.jdbc;

import java.awt.Color;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.JOptionPane;

import london.All\_data;

public class In\_stu\_reg extends javax.swing.JInternalFrame {

jdbc cdb = new jdbc();

public In\_stu\_reg() {

initComponents();

}

private void t\_snoFocusLost(java.awt.event.FocusEvent evt) {

ResultSet r = null;

try {

if ((r = cdb.getdata("select \* from students2 where stu\_no='" + t\_sno.getText() + "'")) == null) {

t\_sname.grabFocus();

} else {

while (r.next()) {

t\_sname.setText(r.getString("name"));

t\_guar\_cont.setText(r.getString("parent\_mobile"));

t\_parent.setText(r.getString("parent\_name"));

t\_saddr.setText(r.getString("address"));

t\_smob.setText(r.getString("mobile"));

t\_snic.setText(r.getString("nic"));

t\_ssch.setText(r.getString("school"));

if (!r.getString("subj1").isEmpty()) {

check\_s4.setText(r.getString("subj1"));

check\_s4.setSelected(true);

} if (!r.getString("subj2").isEmpty()) {

check\_s5.setText(r.getString("subj2"));

check\_s5.setSelected(true);

} if (!r.getString("subj3").isEmpty()) {

check\_s6.setText(r.getString("subj3"));

check\_s6.setSelected(true);

} } } } catch (Exception e) {

System.out.println(e);

} finally {

try {

if (r != null) {

r.close();

}

} catch (SQLException e) {

System.out.println(e);

} } }

private void t\_snicFocusLost(java.awt.event.FocusEvent evt) {

ResultSet r = null;

try {

if ((r = cdb.getdata("select \* from students2 where nic='" + t\_snic.getText() + "'")) == null) { stream\_list.grabFocus();

} else {

while (r.next()) {

t\_sname.setText(r.getString("name"));

t\_guar\_cont.setText(r.getString("parent\_mobile"));

t\_parent.setText(r.getString("parent\_name"));

t\_saddr.setText(r.getString("address"));

t\_smob.setText(r.getString("mobile"));

t\_sno.setText(r.getString("nic"));

t\_ssch.setText(r.getString("school"));

} } } catch (Exception e) {

System.out.println(e);

} finally {

try {

if (r != null) {

r.close();

}

} catch (SQLException e) {

System.out.println(e);

} } }

private void bcreate\_stuActionPerformed(java.awt.event.ActionEvent evt) {

String subj1 = "";

String subj2 = "";

String subj3 = "";

if (check\_s4.isSelected()) {

subj1 = check\_s4.getText();

}

if (check\_s5.isSelected()) {

subj2 = check\_s5.getText();

}

if (check\_s6.isSelected()) {

subj3 = check\_s6.getText();

}

String stream;//-----------setting the stream from the selection list

if (stream\_list.getSelectedIndex() == 0) {

stream = "";

} else {

stream = stream\_list.getSelectedItem().toString();

}

try {

if (t\_sno.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_sno.grabFocus();

t\_sno.setBackground(Color.pink);

} else if (t\_sname.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_sname.grabFocus();

t\_sname.setBackground(Color.pink);

} else if (t\_saddr.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_saddr.grabFocus();

t\_saddr.setBackground(Color.pink);

} else if (t\_snic.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_snic.grabFocus();

t\_snic.setBackground(Color.pink);

} else {

cdb.putdata("INSERT INTO students2(stu\_no,nic,name,address,mobile,school,guardian\_name,"

+ " landline,stream,subj1,subj2,subj3) "

+ "VALUES ('" + t\_sno.getText() + "','" + t\_snic.getText() + "','" + t\_sname.getText() + "',"

+ " '" + t\_saddr.getText() + "','" + t\_smob.getText() + "','" + t\_ssch.getText() + "'," + " '" + t\_parent.getText() + "','" + t\_guar\_cont.getText() + "',"

+ " '" + stream + "','" + subj1 + "','" + subj2 + "','" + subj3 + "')");

JOptionPane.showMessageDialog(null, "Student Details Successfully Entered to Database");

} clean();

} catch (Exception e) {

System.out.println("error in creating student data");

System.out.println(e);

} }

private void stream\_listMouseClicked(java.awt.event.MouseEvent evt) {

String stream;//-----------setting the stream from the selection list

if (stream\_list.getSelectedIndex() == 0) {

stream = "";

} else {

stream = stream\_list.getSelectedItem().toString();

}

switch (stream) {

case "Maths":

//--------setting check box names when stream is selected

check\_s4.setText("Mathematics");

check\_s5.setText("Chemistry");

check\_s6.setText("Physics");

break;

case "Bio":

check\_s4.setText("Biology");

check\_s5.setText("Chemistry");

check\_s6.setText("Physics");

break;

case "Commerce":

check\_s4.setText("Economics");

check\_s5.setText("Business St");

check\_s6.setText("Accounting");

break;

} }

private void stream\_listKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 10) {

String stream;//-----------setting the stream from the selection list

if (stream\_list.getSelectedIndex() == 0) {

stream = "";

} else {

stream = stream\_list.getSelectedItem().toString();

}

switch (stream) {

case "Maths":

//--------setting check box names when stream is selected

check\_s4.setText("Mathematics");

check\_s5.setText("Chemistry");

check\_s6.setText("Physics");

break;

case "Bio":

check\_s4.setText("Biology");

check\_s5.setText("Chemistry");

check\_s6.setText("Physics");

break;

case "Commerce":

check\_s4.setText("Economics");

check\_s5.setText("Business St");

check\_s6.setText("Accounting");

break;

} } }

private void t\_snoKeyPressed(java.awt.event.KeyEvent evt) {

ResultSet rs = null;

if (evt.getKeyCode() == 10) {

try {

if ((rs = cdb.getdata("select \* from students2 where stu\_no='" + t\_sno.getText() + "'")) == null) {

t\_sname.grabFocus();

} else {

while (rs.next()) {

t\_snic.setText(rs.getString("nic"));

t\_sname.setText(rs.getString("name"));

t\_smob.setText(rs.getString("mobile"));

t\_guar\_cont.setText(rs.getString("landline"));

t\_parent.setText(rs.getString("guardian\_name"));

t\_saddr.setText(rs.getString("address"));

t\_ssch.setText(rs.getString("school"));

stream\_list.setSelectedItem(rs.getString("stream"));

if (!rs.getString("subj1").isEmpty()) {

check\_s4.setText(rs.getString("subj1"));

check\_s4.setSelected(true);

}

if (!rs.getString("subj2").isEmpty()) {

check\_s5.setText(rs.getString("subj2"));

check\_s5.setSelected(true);

}

if (!rs.getString("subj3").isEmpty()) {

check\_s6.setText(rs.getString("subj3"));

check\_s6.setSelected(true);

} } } } catch (Exception ex) {

System.out.println(ex);

} finally {

try {

if (rs != null) {

rs.close();

}

} catch (SQLException e) {

System.out.println(e);

} } } }

private void bup\_stuActionPerformed(java.awt.event.ActionEvent evt) {

String subj1 = "";

String subj2 = "";

String subj3 = "";

if (check\_s4.isSelected()) {

subj1 = check\_s4.getText();

} if (check\_s5.isSelected()) {

subj2 = check\_s5.getText();

} if (check\_s6.isSelected()) {

subj3 = check\_s6.getText();

} String stream;//-----------setting the stream from the selection list

if (stream\_list.getSelectedIndex() == 0) {

stream = "";

} else {

stream = stream\_list.getSelectedItem().toString();

} try {

if (t\_sno.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_sno.grabFocus();

t\_sno.setBackground(Color.pink);

} else if (t\_sname.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_sname.grabFocus();

t\_sname.setBackground(Color.pink);

} else if (t\_saddr.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_saddr.grabFocus();

t\_saddr.setBackground(Color.pink);

} else if (t\_snic.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_student, "Required Field Cannot be left blank !!");

t\_snic.grabFocus();

t\_snic.setBackground(Color.pink);

} else {

cdb.putdata("update students2 set "

+ " nic='" + t\_snic.getText() + "',name='" + t\_sname.getText() + "',"

+ " address='" + t\_saddr.getText() + "',mobile='" + t\_smob.getText() + "',"

+ " school='" + t\_ssch.getText() + "',"

+ " guardian\_name='" + t\_parent.getText() + "',landline='" + t\_guar\_cont.getText() + "',"

+ " stream='" + stream + "',subj1='" + subj1 + "',subj2='" + subj2 + "',"

+ " subj3='" + subj3 + "' "

+ " where stu\_no='" + t\_sno.getText() + "'");

JOptionPane.showMessageDialog(null, "Student Details Successfully Updated");

} clean();

} catch (Exception e) {

System.out.println("Error in updating");

System.out.println(e);

} }

private void blist\_stuActionPerformed(java.awt.event.ActionEvent evt) {

london.All\_data all = new All\_data();

all.setVisible(true);

}

private void stream\_listItemStateChanged(java.awt.event.ItemEvent evt) {

if (evt.getStateChange() == java.awt.event.ItemEvent.SELECTED) {

String stream;//-----------setting the stream from the selection list

if (stream\_list.getSelectedIndex() == 0) {

stream = "";

} else {

stream = stream\_list.getSelectedItem().toString();

} //----------------------------------------------------------------------------

if (stream.equals("Maths")) {//--------setting check box names when stream is selected

check\_s4.setText("Mathematics");

check\_s5.setText("Chemistry");

check\_s6.setText("Physics");

} else if (stream.equals("Bio")) {

check\_s4.setText("Biology");

check\_s5.setText("Chemistry");

check\_s6.setText("Physics");

} else if (stream.equals("Commerce")) {

check\_s4.setText("Economics");

check\_s5.setText("Business St");

check\_s6.setText("Accounting");

} } }

private void t\_snicKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() == 37 | evt.getKeyCode() == 39

| evt.getKeyCode() == 144) {

t\_snic.setEditable(true);

} else if (evt.getKeyCode() >= 48 && evt.getKeyCode() <= 57) {

t\_snic.setEditable(true);

} else if (evt.getKeyCode() >= 96 && evt.getKeyCode() <= 105) {

t\_snic.setEditable(true);

} else if (evt.getKeyCode() == 10) {

bcreate\_stu.grabFocus();

} else {

t\_snic.setEditable(false);

JOptionPane.showMessageDialog(null, "There are only numerical National ID values yet", "Error", 2);

t\_snic.setText("");

t\_snic.grabFocus();

}

if (t\_snic.getText().length() <= 8) {

t\_snic.setEditable(true);

} else if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() == 37 | evt.getKeyCode() == 39

| evt.getKeyCode() == 144) {

t\_snic.setEditable(true);

} else { t\_snic.setEditable(false);

} }

private void t\_smobKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() >= 48 && evt.getKeyCode() <= 57

| evt.getKeyCode() == 37

| evt.getKeyCode() == 39 | evt.getKeyCode() == 144

| evt.getKeyCode() >= 96 && evt.getKeyCode() <= 105) {

t\_smob.setEditable(true);

} else if (evt.getKeyCode() == 10) {

t\_smob.setEditable(true);

} else {

t\_smob.setEditable(false);

JOptionPane.showMessageDialog(null, "Enter only the numeric values of the mobile number excluding DASH(-)", "Error", 2);

t\_smob.setText("");

t\_smob.grabFocus();

} }

private void t\_guar\_contKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() >= 46 && evt.getKeyCode() <= 57) {

t\_guar\_cont.setEditable(true);

} else if (evt.getKeyCode() == 10) {

t\_guar\_cont.setEditable(true);

} else {

t\_guar\_cont.setEditable(false);

JOptionPane.showMessageDialog(null, "Enter only the numeric values of the mobile number excluding DASH(-)", "Error", 2);

t\_guar\_cont.setText("");

t\_guar\_cont.grabFocus();

} }

private void bresetActionPerformed(java.awt.event.ActionEvent evt) {

clean();

}

void clean() {

t\_sno.setText("");

t\_sname.setText("");

t\_saddr.setText("");

t\_snic.setText("");

t\_smob.setText("");

t\_parent.setText("");

t\_ssch.setText("");

t\_guar\_cont.setText("");

stream\_list.setSelectedIndex(0);

check\_s4.setText("subject 1");

check\_s5.setText("subject 2");

check\_s6.setText("subject 3");

check\_s4.setSelected(false);

check\_s5.setSelected(false);

check\_s6.setSelected(false);

}

}

**Teachers Registration**

package in\_frames;

import Connection.jdbc;

import java.awt.Color;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.JOptionPane;

import london.All\_data;

public class In\_tea\_reg extends javax.swing.JInternalFrame {

jdbc cdb = new jdbc();

public In\_tea\_reg() {

initComponents(); }

void resetFields() {

t\_tno.setText("");

t\_tnic.setText("");

t\_tname.setText("");

t\_taddr.setText("");

t\_tmob.setText("");

t\_tvehicle.setText("");

combo\_subject.setSelectedIndex(0);

}

private void bcreate\_teacherActionPerformed(java.awt.event.ActionEvent evt) {

if (t\_tno.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_tno.grabFocus();

t\_tno.setBackground(Color.pink);

} else if (t\_tname.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_tname.grabFocus();

t\_tname.setBackground(Color.pink);

} else if (t\_taddr.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_taddr.grabFocus();

t\_taddr.setBackground(Color.pink);

} else if (t\_tnic.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_tnic.grabFocus();

t\_tnic.setBackground(Color.pink);

} else { try {

cdb.putdata("INSERT INTO tutor(tutor\_no,nic,name,address,mobile,subject,vehicle) "

+ "VALUES ('" + t\_tno.getText() + "','" + t\_tnic.getText() + "','" + t\_tname.getText() + "',"

+ "'" + t\_taddr.getText() + "','" + t\_tmob.getText() + "',"

+ "'" + combo\_subject.getSelectedItem() + "',"

+ "'" + t\_tvehicle.getText() + "')");

} catch (Exception e) {

System.out.println(e);

} } resetFields(); }

private void bup\_teacherActionPerformed(java.awt.event.ActionEvent evt) {

if (t\_tno.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_tno.grabFocus();

t\_tno.setBackground(Color.pink);

} else if (t\_tname.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_tname.grabFocus();

t\_tname.setBackground(Color.pink);

} else if (t\_taddr.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_taddr.grabFocus();

t\_taddr.setBackground(Color.pink);

} else if (t\_tnic.getText().equals("")) {

JOptionPane.showMessageDialog(layer\_teachers, "Required Field Cannot be left blank !!"); t\_tnic.grabFocus();

t\_tnic.setBackground(Color.pink);

} else { try { cdb.putdata("update tutor set "

+ " nic='" + t\_tnic.getText() + "',name='" + t\_tname.getText() + "',"

+ " address='" + t\_taddr.getText() + "',mobile='" + t\_tmob.getText() + "',"

+ " subject='" + combo\_subject.getSelectedItem() + "',"

+ " vehicle='" + t\_tvehicle.getText() + "' "

+ " where tutor\_no='" + t\_tno.getText() + "'");

} catch (Exception e) {

System.out.println(e);

} } resetFields(); }

private void t\_tnoFocusLost(java.awt.event.FocusEvent evt) {

ResultSet rs = null;

try { rs = cdb.getdata("select \* from tutor where tutor\_no='" + t\_tno.getText() + "'"); rs.next(); t\_taddr.setText(rs.getString("address"));

t\_tname.setText(rs.getString("name"));

t\_tmob.setText(rs.getString("mobile"));

t\_tvehicle.setText(rs.getString("vehicle"));

t\_tnic.setText(rs.getString("nic"));

combo\_subject.setSelectedItem(rs.getString("subject"));

} catch (Exception e) { System.out.println(e);

} finally {

try {

if (rs != null) { rs.close(); } } catch (SQLException e) {

System.out.println(e); } } }

private void t\_tnoKeyPressed(java.awt.event.KeyEvent evt) {

ResultSet rs = null;

if (evt.getKeyCode() == 10) {

try {

rs = cdb.getdata("select \* from tutor where tutor\_no='" + t\_tno.getText() + "'");

rs.next();

t\_taddr.setText(rs.getString("address"));

t\_tname.setText(rs.getString("name"));

t\_tmob.setText(rs.getString("mobile"));

t\_tvehicle.setText(rs.getString("vehicle"));

t\_tnic.setText(rs.getString("nic"));

combo\_subject.setSelectedItem(rs.getString("subject"));

} catch (Exception e) {

System.out.println(e);

} finally {

try {

if (rs != null) {

rs.close();

}

} catch (SQLException e) {

System.out.println(e);

} } } }

private void blist\_teaActionPerformed(java.awt.event.ActionEvent evt) {

All\_data all = new All\_data();

all.setVisible(true);

}

private void t\_tnicKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() >= 48 && evt.getKeyCode() <= 57

| evt.getKeyCode() == 37

| evt.getKeyCode() == 39 | evt.getKeyCode() == 144

| evt.getKeyCode() >= 96 && evt.getKeyCode() <= 105) {

t\_tnic.setEditable(true);

} else if (evt.getKeyCode() == 10) {

t\_tvehicle.grabFocus();

} else {

t\_tnic.setEditable(false);

JOptionPane.showMessageDialog(null, "There are only numerical National ID values yet", "Error", 2);

}

if (t\_tnic.getText().length() <= 8) {

t\_tnic.setEditable(true);

} else {

t\_tnic.setEditable(false);

} }

private void t\_tmobKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() >= 48 && evt.getKeyCode() <= 57

| evt.getKeyCode() == 37

| evt.getKeyCode() == 39 | evt.getKeyCode() == 144

| evt.getKeyCode() >= 96 && evt.getKeyCode() <= 105) {

t\_tmob.setEditable(true);

} else if (evt.getKeyCode() == 10) {

t\_tmob.setEditable(true);

} else {

t\_tmob.setEditable(false);

JOptionPane.showMessageDialog(null,

"Enter only the numeric values of the mobile number excluding DASH(-)", "Error", 2); t\_tmob.setText(""); t\_tmob.grabFocus(); } }

private void bresetActionPerformed(java.awt.event.ActionEvent evt) {

resetFields();

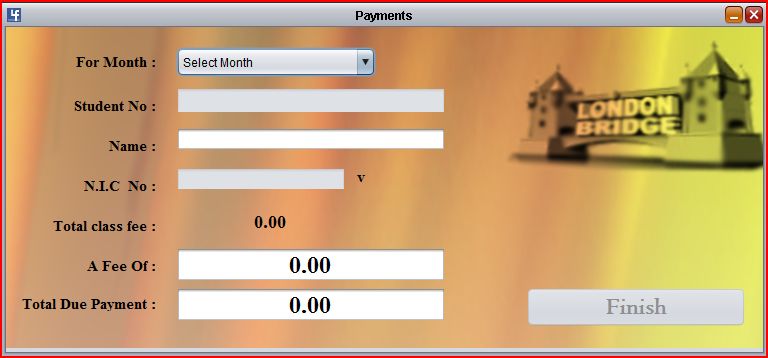
}

private void t\_tnicKeyReleased(java.awt.event.KeyEvent evt) {

if (t\_tnic.getText().length() <= 8) {

t\_tnic.setEditable(true);

} else { t\_tnic.setEditable(false); } }

**Marking Class Fees Received**

package in\_frames;

import Connection.jdbc;

import java.sql.ResultSet;

import java.sql.SQLException;

import javax.swing.JOptionPane;

public class In\_payments extends javax.swing.JInternalFrame {

jdbc cdb = new jdbc();

public In\_payments() {

initComponents();

}

private void t\_sno1FocusLost(java.awt.event.FocusEvent evt) {

ResultSet r = null;

try {

r = cdb.getdata("select \* from students2 where stu\_no='" + t\_sno1.getText() + "'");

if (r.first()) {

t\_sname1.setText(r.getString("name"));

t\_snic1.setText(r.getString("nic"));

} } catch (Exception e) { System.out.println(e);

} finally {

try { if (r != null) { r.close(); }

} catch (SQLException e) {

System.out.println(e);

} } }

private void t\_snic1FocusLost(java.awt.event.FocusEvent evt) {

ResultSet r = null;

if (t\_sno1.getText().isEmpty()) {

JOptionPane.showMessageDialog(layer\_payment, "Enter Student number to continue"); } else { try {

int i = 0;

r = cdb.getdata("select \* from students2 where stu\_no='" + t\_sno1.getText() + "'"); if (r.next()) {

t\_sname1.setText(r.getString("name"));

t\_snic1.setText(r.getString("nic"));

if (!r.getString("subj1").isEmpty()) {

i++;

} if (!r.getString("subj2").isEmpty()) {

// l\_totalfee.setText(sub);

i++; }

if (!r.getString("subj3").isEmpty()) {

// l\_totalfee.setText(sub);

i++; }

l\_totalfee.setText(Integer.toString(i \* 1200));

} } catch (Exception e) {

System.out.println(e);

} finally {

try { if (r != null) {

r.close();

} } catch (SQLException e) { System.out.println(e);

} } } }

private void bfinishActionPerformed(java.awt.event.ActionEvent evt) {

try { Connection.jdbc.con().createStatement()

.executeUpdate("insert into payments(stu\_no,nic,total\_fee,paid,due,month)"

+ "values('" + t\_sno1.getText() + "','" + t\_snic1.getText() + "',"

+ " '" + l\_totalfee.getText() + "','" + t\_fee.getText() + "','" + t\_due.getText() + "',"

+ " '" + combo\_month.getSelectedItem().toString() + "')");

System.out.println("payment success");

JOptionPane.showMessageDialog(null, "Payment Successfull!",

"Success..", JOptionPane.INFORMATION\_MESSAGE);

resetFields();

} catch (Exception e) { System.out.println(e); } System.gc(); }

private void t\_sno1KeyPressed(java.awt.event.KeyEvent evt) {

ResultSet r2 = null, r = null;

if (evt.getKeyChar() == 10) {

if (t\_sno1.getText().isEmpty()) {

JOptionPane.showMessageDialog(layer\_payment, "Enter Student number to continue"); } else {

try { int i = 0;

r = cdb.getdata("select \* from students2 where stu\_no='" + t\_sno1.getText() + "'"); r.next();

t\_sname1.setText(r.getString("name"));

t\_snic1.setText(r.getString("nic"));

if (!r.getString("subj1").isEmpty()) {

i++; } if (!r.getString("subj2").isEmpty()) {

// l\_totalfee.setText(sub);

i++; }

if (!r.getString("subj3").isEmpty()) {

// l\_totalfee.setText(sub);

i++; }

l\_totalfee.setText(Integer.toString(i \* 1200));

r2 = jdbc.con().createStatement().executeQuery("select \* from payments "

+ "where stu\_no='" + t\_sno1.getText() + "' and month='" + combo\_month.getSelectedItem().toString() + "'");

if (r2.first()) {

l\_totalfee.setText(r2.getString("total\_fee").toString());

t\_due.setText(r2.getString("due" + ".00"));

} } catch (Exception e) {

System.out.println("error loading due of month" + e);

} finally {

try { if (r != null) { r.close(); }if(r2!=null){

r2.close(); }

} catch (SQLException e) {

System.out.println(e);

} } } } }

private void t\_feeKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() >= 46 && evt.getKeyCode() <= 57 | evt.getKeyCode() == 8

| evt.getKeyCode() == 127) {

t\_fee.setEditable(true);

} else if (evt.getKeyCode() == 10) {

if (t\_due.getText().isEmpty()) {//------------empty due field makes a new payment for the month,

int total = Integer.valueOf(l\_totalfee.getText()); //class fee

int fee = Integer.valueOf(t\_fee.getText()); // given money

int due = total - fee;

int negDue = fee - total;

if (due < 0) {

t\_due.setText(String.valueOf(negDue));

t\_fee.setText(t\_fee.getText() + ".00");

labelDue.setText("<html><font color=red>To Student</font><html>");

combo\_month.setEnabled(true);

bfinish.setEnabled(true);

bfinish.grabFocus();

} else if (due > 0) {

t\_due.setText(String.valueOf(due) + ".00");

t\_fee.setText(t\_fee.getText() + ".00");

labelDue.setText("<html><font color=red>From Student</font><html>");

combo\_month.setEnabled(true);

bfinish.setEnabled(true);

bfinish.grabFocus();

} } else if (!t\_due.getText().isEmpty()) {

int duedue = Integer.valueOf(t\_due.getText()) - Integer.valueOf(t\_fee.getText());

int negduedue = Integer.valueOf(t\_fee.getText()) - Integer.valueOf(t\_due.getText());

if (duedue < 0) {

t\_due.setText(String.valueOf(negduedue));

t\_fee.setText(t\_fee.getText() + ".00");

labelDue.setText("<html><font color=red>To Student</font><html>");

combo\_month.setEnabled(true);

bfinish.setEnabled(true);

bfinish.grabFocus();

} else if (duedue > 0) {

t\_due.setText(String.valueOf(duedue) + ".00");

t\_fee.setText(t\_fee.getText() + ".00");

labelDue.setText("<html><font color=red>From Student</font><html>");

combo\_month.setEnabled(true);

bfinish.setEnabled(true);

bfinish.grabFocus();

} }

} else { t\_fee.setEditable(false);

JOptionPane.showMessageDialog(layer\_payment, "Enter a numeric cash value !");

t\_fee.setText("");

t\_fee.grabFocus();

} }

private void t\_feeFocusGained(java.awt.event.FocusEvent evt) {

if (t\_due.getText().isEmpty()) {

t\_fee.setText("");

} else {

t\_fee.setText("");

t\_due.setText("");

} }

private void combo\_monthItemStateChanged(java.awt.event.ItemEvent evt) {

if (combo\_month.getSelectedIndex() != 0) {

t\_sno1.setEnabled(true);

t\_snic1.setEnabled(true);

} else if (combo\_month.getSelectedIndex() == 0) {

t\_sno1.setEnabled(false);

t\_snic1.setEnabled(false);

} }

void resetFields() {

t\_sno1.setText("");

t\_snic1.setText("");

t\_sname1.setText("");

t\_due.setText("0.00");

t\_fee.setText("0.00");

l\_totalfee.setText("0");

l\_totalfeecents.setText(".00");

combo\_month.setSelectedIndex(0); }

**Paying up Teachers’ Shares**

package in\_frames;

import Connection.jdbc;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.util.Date;

import javax.swing.JOptionPane;

public class in\_compensation extends javax.swing.JInternalFrame {

public in\_compensation() {

initComponents();

lbl\_useragent.setText("USER : " + new LOGIN.login().getUserAgent());

}

void loadTeachers() {

ResultSet rs1 = null;

combo\_teacher.removeAllItems();

if (!combo\_teacher.isEnabled()) {

JOptionPane.showMessageDialog(null, "Teacher List is not enabled!"

+ " \n Something has gone wrong. \n Please contact System Administrator");

} else if (combo\_teacher.isEnabled()) {

try { rs1 = jdbc.con().createStatement().executeQuery("select \* from tutor");

while (rs1.next()) {

combo\_teacher.addItem(rs1.getString("name"));

} } catch (Exception e) {

System.out.println(e);

} finally {

try {

if (rs1 != null) {

rs1.close();

} } catch (SQLException e) { System.out.println(e); } } } }

private void combo\_monthItemStateChanged(java.awt.event.ItemEvent evt) {

if (evt.getStateChange() == java.awt.event.ItemEvent.SELECTED) {

combo\_batch.setEnabled(true); } }

private void combo\_monthKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 10) {

combo\_batch.grabFocus(); } }

private void combo\_batchItemStateChanged(java.awt.event.ItemEvent evt) {

if (evt.getStateChange() == java.awt.event.ItemEvent.SELECTED) {

combo\_teacher.setEnabled(true); } }

private void combo\_batchKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 10) {

combo\_teacher.grabFocus(); } }

private void combo\_teacherFocusGained(java.awt.event.FocusEvent evt) {

loadTeachers(); }

private void combo\_teacherItemStateChanged(java.awt.event.ItemEvent evt) {

ResultSet rs = null;

if (evt.getStateChange() == java.awt.event.ItemEvent.SELECTED) {

try { rs = jdbc.con().createStatement().executeQuery("select subject from tutor "

+ "where name='" + combo\_teacher.getSelectedItem().toString() + "'");

rs.next(); l\_subject.setText(rs.getString("subject"));

t\_income.setEnabled(true);

} catch (Exception ex) {

System.out.println(ex);

} finally {

try { if (rs != null) { rs.close(); }

} catch (SQLException e) {

System.out.println(e); } } } }

private void combo\_teacherKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 10) {

t\_income.grabFocus(); } }

private void t\_incomeKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() == 127 | evt.getKeyCode() >= 48 && evt.getKeyCode() <= 57

| evt.getKeyCode() == 37

| evt.getKeyCode() == 39 | evt.getKeyCode() == 144

| evt.getKeyCode() >= 96 && evt.getKeyCode() <= 105) {

t\_income.setEditable(true);

} else if (evt.getKeyCode() == 10) {

t\_percentage.grabFocus();

} else if (evt.getKeyCode() == 46 | evt.getKeyCode() == 110) {

t\_income.setEditable(false);

JOptionPane.showMessageDialog(null, "Decimal values aren't allowed !");

t\_income.setText("");

} else { t\_income.setEditable(false);

JOptionPane.showMessageDialog(null, "this input field stands for numbers only");

t\_income.setText(""); } }

private void t\_incomeFocusLost(java.awt.event.FocusEvent evt) {

if (t\_income.getText() == null | "".equals(t\_income.getText())) {

JOptionPane.showMessageDialog(null, "Empty Income field");

t\_income.grabFocus();

} else if (Integer.valueOf(t\_income.getText()) == 0) {

JOptionPane.showMessageDialog(null,

"You cannot pay for something \n that has nothing to be paid for", "Error", 2);

} else { t\_percentage.setEnabled(true); check\_perc\_ok.setEnabled(true); } }

private void t\_percentageFocusLost(java.awt.event.FocusEvent evt) {

if (t\_percentage.getText() == null | "".equals(t\_percentage.getText())) {

JOptionPane.showMessageDialog(null, "Empty share percentage field");

t\_percentage.grabFocus(); } }

private void t\_percentageKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 8 | evt.getKeyCode() >= 48 && evt.getKeyCode() <= 57

| evt.getKeyCode() == 127

| evt.getKeyCode() == 37

| evt.getKeyCode() == 39 | evt.getKeyCode() == 144

| evt.getKeyCode() >= 96 && evt.getKeyCode() <= 105) {

t\_percentage.setEditable(true);

} else if (evt.getKeyCode() == 10) {

check\_perc\_ok.grabFocus();

} else if (evt.getKeyCode() == 46 | evt.getKeyCode() == 110) {

t\_percentage.setEditable(false);

JOptionPane.showMessageDialog(null, "Decimal values aren't allowed !");

t\_percentage.setText("");

} else { t\_percentage.setEditable(false);

JOptionPane.showMessageDialog(null, "this input field stands for numbers only");

t\_percentage.setText(""); } }

private void check\_perc\_okMouseClicked(java.awt.event.MouseEvent evt) {

if (check\_perc\_ok.isSelected()) {

if (Integer.valueOf(t\_percentage.getText()) == 0) {

int conf = JOptionPane.showConfirmDialog(null, "Are you sure that institute \n"

+ "should receive no share of the income?", "Confirm action", JOptionPane.QUESTION\_MESSAGE);

if (conf == JOptionPane.NO\_OPTION) {

t\_percentage.setText("");

t\_percentage.grabFocus();

} else { calculateShare();

check\_final.setEnabled(true); } } else {

calculateShare(); check\_final.setEnabled(true); } } }

private void check\_perc\_okMouseReleased(java.awt.event.MouseEvent evt) {

if (check\_perc\_ok.isSelected()) {

if (Integer.valueOf(t\_percentage.getText()) == 0) {

int conf = JOptionPane.showConfirmDialog(null, "Are you sure that institute \n"

+ "should receive no share of the income?", "Confirm action", JOptionPane.QUESTION\_MESSAGE);

if (conf == JOptionPane.NO\_OPTION) {

t\_percentage.setText("");

t\_percentage.grabFocus();

} else { calculateShare(); check\_final.setEnabled(true); } }

calculateShare(); check\_final.setEnabled(true); } } }

private void check\_perc\_okKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 32) {

if (Integer.valueOf(t\_percentage.getText()) == 0) {

int conf = JOptionPane.showConfirmDialog(null, "Are you sure that institute \n"

+ "should receive no share of the income?", "Confirm action", JOptionPane.QUESTION\_MESSAGE);

if (conf == JOptionPane.NO\_OPTION) {

t\_percentage.setText("");

t\_percentage.grabFocus();

} else { calculateShare(); check\_final.setEnabled(true);

} } else { calculateShare(); check\_final.setEnabled(true); } } }

private void check\_perc\_okItemStateChanged(java.awt.event.ItemEvent evt) {

if (evt.getStateChange() == java.awt.event.ItemEvent.DESELECTED) {

t\_share.setText("");

check\_final.setEnabled(false); } }

private void check\_finalMouseClicked(java.awt.event.MouseEvent evt) {

if (check\_final.isSelected()) {

bfinish.setEnabled(true); } }

private void check\_finalMouseReleased(java.awt.event.MouseEvent evt) {

if (check\_final.isSelected()) {

bfinish.setEnabled(true); } }

private void check\_finalItemStateChanged(java.awt.event.ItemEvent evt) {

if (evt.getStateChange() == java.awt.event.ItemEvent.DESELECTED) {

bfinish.setEnabled(false);

} else if (evt.getStateChange() == java.awt.event.ItemEvent.SELECTED) {

bfinish.setEnabled(true); } }

private void bfinishActionPerformed(java.awt.event.ActionEvent evt) {

finish(); }

private void check\_finalKeyPressed(java.awt.event.KeyEvent evt) {

if (evt.getKeyCode() == 32) {

if (check\_final.isSelected()) {

bfinish.setEnabled(true); } } }

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {

this.dispose();

london.All\_data all = new london.All\_data();

all.setVisible(true); all.selectRecords(); }

private void finish() {

ResultSet rs = null;

String date = new Date().toString();

int income = Integer.valueOf(t\_income.getText());

int per = Integer.valueOf(t\_percentage.getText());

int per\_value = (income \* per) / 100;

try { rs = jdbc.con().createStatement().executeQuery("select tutor\_no from tutor "

+ "where name='" + combo\_teacher.getSelectedItem() + "'");

if (rs.first()) { jdbc.con().createStatement().executeUpdate("insert into compensation(month,batch,tutor\_no,name,"

+ "income,percentage,percentage\_value,for\_teacher,date,useragent)"

+ "values('" + combo\_month.getSelectedItem() + "',"

+ " '" + combo\_batch.getSelectedItem() + "','" + rs.getString("tutor\_no") + "',"

+ " '" + combo\_teacher.getSelectedItem() + "','" + t\_income.getText() + "'," + " '" + t\_percentage.getText() + "'," + " '" + per\_value + "', '" + t\_share.getText() + "','" + date + "',"

+ " '" + new LOGIN.login().getUserAgent() + "') ");

System.out.println("compensations syncing successful");

JOptionPane.showMessageDialog(null, "Payment Completed Successfully");

combo\_batch.setEnabled(false);

t\_income.setEnabled(false);

t\_percentage.setEnabled(false);

t\_share.setEnabled(false);

check\_perc\_ok.setSelected(false);

check\_final.setSelected(false);

check\_perc\_ok.setEnabled(false);

check\_final.setEnabled(false);

bfinish.setEnabled(false);

combo\_batch.setSelectedIndex(0);

combo\_month.setSelectedIndex(0);

combo\_teacher.removeAllItems();

combo\_teacher.addItem("Select Tutor");

combo\_teacher.setEnabled(false);

l\_subject.setText("");

t\_income.setText("");

t\_percentage.setText("");

t\_share.setText("");

System.out.println("all fields resetted"); } } catch (Exception e) {

JOptionPane.showMessageDialog(null, "Payment Not Completed !","Error",JOptionPane.ERROR\_MESSAGE);

System.out.println("error in inserting payments to teachers");

System.out.println(e);

} finally {

try { if (rs != null) {

rs.close(); } } catch (SQLException e) {

System.out.println(e); } } }

private void calculateShare() {

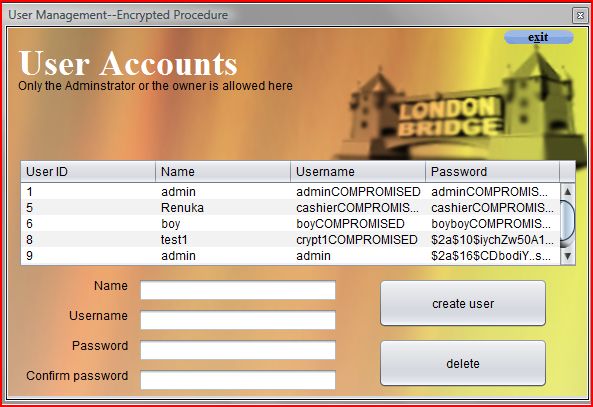
int income = Integer.valueOf(t\_income.getText());

int per = Integer.valueOf(t\_percentage.getText());

int per\_value = (income \* per) / 100;

t\_share.setText(String.valueOf(income - per\_value)); }

**User Management**

package london;

import Connection.jdbc;

import jBCrypt.BCrypt;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.sql.Statement;

import java.util.Vector;

import javax.swing.JOptionPane;

import javax.swing.UIManager;

import javax.swing.UnsupportedLookAndFeelException;

import javax.swing.plaf.nimbus.NimbusLookAndFeel;

import javax.swing.table.DefaultTableModel;

public class user\_manage extends javax.swing.JFrame {

newHome h = null;

public user\_manage() {

initComponents();

loadtable(); }

private void bcreateActionPerformed(java.awt.event.ActionEvent evt) {

loadtable();

String hashed = BCrypt.hashpw(new String(t\_pass2.getPassword()), BCrypt.gensalt());

try { if (new String(t\_pass.getPassword()).equals(new String(t\_pass2.getPassword()))) {

jdbc.con().createStatement().executeUpdate("insert into users(name,username,password)"

+ " values('" + t\_name.getText() + "','" + t\_uname.getText() + "','"

+ hashed + "') "); } else {

JOptionPane.showMessageDialog(null, "Passwords do not match \n Try again !", "Error", 2);

} t\_name.setText(""); t\_uname.setText(""); t\_pass.setText("");

t\_pass2.setText(""); } catch (Exception e) { System.out.println(e); }

this.dispose(); new user\_manage().setVisible(true); }

private void bdeleteActionPerformed(java.awt.event.ActionEvent evt) {

DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

Statement st = null;

int sRow = jTable1.getSelectedRow();

if (sRow == 0) {

JOptionPane.showMessageDialog(null, "You cannot delete the Administrator account", "Error", 2); } else {

try {

st = jdbc.con().createStatement();

st.executeUpdate("delete from users where uid='" + dtm.getValueAt(sRow, 0) + "'"); dtm.removeRow(jTable1.getSelectedRow());

} catch (Exception ex) {

System.out.println("delete error in users table");

System.out.println(ex); } finally {

try { if (st != null) { st.close(); }}catch (SQLException e) {

System.out.println(e); } } } this.dispose();

new user\_manage().setVisible(true); }

private void jTable1KeyPressed(java.awt.event.KeyEvent evt) {

JOptionPane.showMessageDialog(null, "Invalid Function \n Keyboard Operation is not allowed!", "Error", 2); }

private void t\_nameFocusLost(java.awt.event.FocusEvent evt) {

t\_uname.grabFocus(); }

private void t\_unameFocusLost(java.awt.event.FocusEvent evt) {

t\_pass.grabFocus(); }

private void t\_passFocusLost(java.awt.event.FocusEvent evt) {

t\_pass2.grabFocus(); }

private void t\_pass2FocusLost(java.awt.event.FocusEvent evt) {

bcreate.grabFocus(); }

private void jLabel6MouseClicked(java.awt.event.MouseEvent evt) {

this.dispose(); }

final void loadtable() {

ResultSet rs = null;

DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

while (dtm.getRowCount() > 0) {

dtm.removeRow(0); }

try { rs = jdbc.con().createStatement().executeQuery("select \* from users");

while (rs.next()) {

Vector v = new Vector();

v.add(rs.getString("uid"));

v.add(rs.getString("name"));

v.add(rs.getString("username"));

v.add(rs.getString("password"));

dtm.addRow(v); }

JOptionPane.showMessageDialog(null, "Database successfully Synced to table");

} catch (Exception ex) {

System.out.println("Error loading users table");

System.out.println(ex);

} finally { try { if (rs != null) { rs.close(); }

} catch (SQLException e) {

System.out.println(e); } } }

public static void main(String args[]) {

try { for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Windows".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break; } } } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(user\_manage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (InstantiationException ex) { java.util.logging.Logger.getLogger(user\_manage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) { java.util.logging.Logger.getLogger(user\_manage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (javax.swing.UnsupportedLookAndFeelException ex) { java.util.logging.Logger.getLogger(user\_manage.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); }

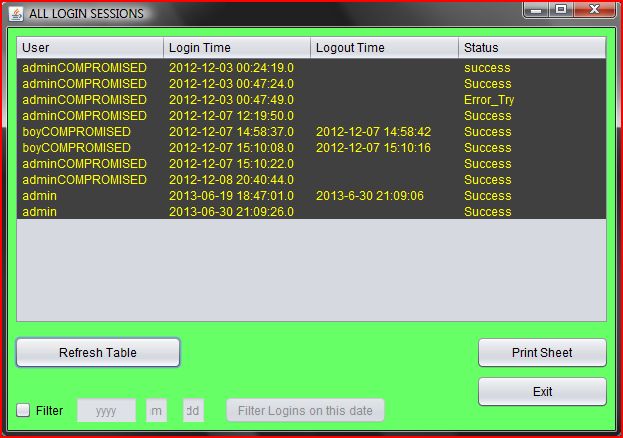
/\* Create and display the form \*/

java.awt.EventQueue.invokeLater(new Runnable() {

public void run() { try {UIManager.setLookAndFeel(new NimbusLookAndFeel());

new user\_manage().setVisible(true);} catch (UnsupportedLookAndFeelException ex) {

System.out.println(ex); } } }); }

**Login Sessions**

package london;

import Connection.jdbc;

import java.awt.Color;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.MessageFormat;

import java.util.Date;

import java.util.Vector;

import javax.swing.JOptionPane;

import javax.swing.JTable;

import javax.swing.table.DefaultTableModel;

public class logins extends javax.swing.JFrame {

public logins() {

initComponents();

refr();

tableColors();

}

private void refr() {

ResultSet r0 = null;

DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

while (dtm.getRowCount() > 0) {

dtm.removeRow(0);

} try { r0 = jdbc.con().createStatement().executeQuery("select \* from login\_sessions");

while (r0.next()) { Vector v0 = new Vector();

v0.add(r0.getString("users\_username"));

v0.add(r0.getString("loginAt"));

v0.add(r0.getString("logoutAt"));

v0.add(r0.getString("status"));

dtm.addRow(v0);

} System.gc();

} catch (Exception e) {

System.out.println("error refreshing logins list");

System.out.println(e);

} finally { try { if (r0 != null) { r0.close(); }

} catch (SQLException e) {

System.out.println(e); } } }

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {

MessageFormat header = new MessageFormat("Login Records' Datasheet.");

MessageFormat footer = new MessageFormat("—Occult Solutions--" + new Date().toString());

try { jTable1.print(JTable.PrintMode.FIT\_WIDTH, header, footer);

} catch (Exception e) {

System.err.format("Cannot Print %s%n", e.getMessage()); }

JOptionPane.showMessageDialog(null, "<html>Remember, You Can only Print the Sheet with your Printer!<br> "

+ "<font color=red><b>Do not Use other File Formats like .pdf .ps or .xps</b></font></html>",

"Remember!", JOptionPane.WARNING\_MESSAGE); }

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {

refr(); System.gc(); }

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {

this.dispose(); }

private void t\_searchYearFocusGained(java.awt.event.FocusEvent evt) {

t\_searchYear.setText(null); }

private void t\_searchMonthFocusGained(java.awt.event.FocusEvent evt) {

t\_searchMonth.setText(null); }

private void t\_searchDayFocusGained(java.awt.event.FocusEvent evt) {

t\_searchDay.setText(null); }

private void bfilterLoginsActionPerformed(java.awt.event.ActionEvent evt) {

ResultSet r0 = null;

DefaultTableModel dtm = (DefaultTableModel) jTable1.getModel();

while (dtm.getRowCount() > 0) {

dtm.removeRow(0); } try {

r0 = jdbc.con().createStatement().executeQuery("select \* from login\_sessions "

+ " where loginAt like "

+ " '" + t\_searchYear.getText().toString() + "-" + t\_searchMonth.getText().toString() + "-"

+ t\_searchDay.getText().toString() + "%' ");

while (r0.next()) {

Vector v0 = new Vector();

v0.add(r0.getString("users\_username"));

v0.add(r0.getString("loginAt"));

v0.add(r0.getString("logoutAt"));

v0.add(r0.getString("status"));

dtm.addRow(v0); } } catch (Exception e) {

System.out.println("" + e); } finally { try { if (r0 != null) {

r0.close(); } } catch (SQLException e) { System.out.println(e); } } }

private void jCheckBox1StateChanged(javax.swing.event.ChangeEvent evt) {

if (jCheckBox1.isSelected()) {

t\_searchDay.setEnabled(true);

t\_searchYear.setEnabled(true);

t\_searchMonth.setEnabled(true);

bfilterLogins.setEnabled(true);

} else { refr();

t\_searchDay.setText("dd");

t\_searchYear.setText("yyyy");

t\_searchMonth.setText("mm");

t\_searchDay.setEnabled(false);

t\_searchYear.setEnabled(false);

t\_searchMonth.setEnabled(false);

bfilterLogins.setEnabled(false); } }

public static void main(String args[]) {

try { for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break; } }

} catch (ClassNotFoundException ex) { java.util.logging.Logger.getLogger(logins.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); } catch (InstantiationException ex) { java.util.logging.Logger.getLogger(logins.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) { java.util.logging.Logger.getLogger(logins.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); } catch (javax.swing.UnsupportedLookAndFeelException ex) { java.util.logging.Logger.getLogger(logins.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); }

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

public void run() {

new logins().setVisible(true); } }); }

private void tableColors() {

jTable1.setBackground(Color.darkGray);

jTable1.setForeground(Color.yellow); }}

**All Data Tables**

package london;

import london.rescumanager.RescueManager;

import Connection.jdbc;

import java.awt.Color;

import java.sql.ResultSet;

import java.sql.SQLException;

import java.text.MessageFormat;

import java.util.Date;

import java.util.Vector;

import javax.swing.JOptionPane;

import javax.swing.JTable;

import javax.swing.table.DefaultTableModel;

public class All\_data extends javax.swing.JFrame {

newHome h;

public All\_data() {

initComponents();

loadStudents();

loadTeachers();

loadComps();

table\_comps.setSelectionBackground(Color.lightGray);

table\_comps.setSelectionForeground(Color.white);

table\_stu.setSelectionBackground(Color.lightGray);

table\_stu.setSelectionForeground(Color.white);

table\_tea1.setSelectionBackground(Color.lightGray);

table\_tea1.setSelectionForeground(Color.white);}

public void selectRecords() {

jTabbedPane1.setSelectedIndex(1); }

private void table\_stuKeyPressed(java.awt.event.KeyEvent evt) {

System.out.println(evt.getKeyCode());

}

private void loadStudents() {

ResultSet rs1 = null;

try { DefaultTableModel dtm1 = (DefaultTableModel) table\_stu.getModel();

while (dtm1.getRowCount() > 0) { dtm1.removeRow(0); }

System.out.println("student list resyncing");

rs1 = jdbc.con().createStatement().executeQuery("select \* from students2");

while (rs1.next()) {

Vector v = new Vector();

v.add(rs1.getString("stu\_no"));

v.add(rs1.getString("name"));

v.add(rs1.getString("address"));

v.add(rs1.getString("mobile"));

v.add(rs1.getString("stream"));

v.add(rs1.getString("guardian\_name"));

v.add(rs1.getString("landline"));

v.add(rs1.getString("school"));

v.add(rs1.getString("nic"));

dtm1.addRow(v); }

System.out.println("sync complete");

} catch (Exception e) {

System.out.println("student table retrieval error");

System.out.println(e);

} finally { try { if (rs1 != null) { rs1.close(); }

} catch (SQLException e) { System.out.println(e); } }}

private void loadTeachers() {

ResultSet rs2 = null;

DefaultTableModel dtm2 = (DefaultTableModel) table\_tea1.getModel();

while (dtm2.getRowCount() > 0) {

dtm2.removeRow(0); }

System.out.println("teacher list resyncing");

try { rs2 = jdbc.con().createStatement().executeQuery("select \* from tutor");

while (rs2.next()) {

Vector v = new Vector();

v.add(rs2.getString("tutor\_no"));

v.add(rs2.getString("subject"));

v.add(rs2.getString("name"));

v.add(rs2.getString("address"));

v.add(rs2.getString("mobile"));

v.add(rs2.getString("nic"));

v.add(rs2.getString("vehicle"));

dtm2.addRow(v); }

System.out.println("sync complete");

} catch (Exception e) {

System.out.println("teacher table retrieval error");

System.out.println(e);

} finally { try { if (rs2 != null) { rs2.close(); } } catch (SQLException e) {

System.out.println(e); } } }

private void loadComps() {

ResultSet rs2 = null;

DefaultTableModel dtm3 = (DefaultTableModel) table\_comps.getModel();

while (dtm3.getRowCount() > 0) {

dtm3.removeRow(0); }

System.out.println("income shares resyncing");

try { rs2 = jdbc.con().createStatement().executeQuery("select \* from compensation");

while (rs2.next()) {

Vector v = new Vector();

v.add(rs2.getString("name"));

v.add(rs2.getString("batch"));

v.add(rs2.getString("income"));

v.add(rs2.getString("percentage"));

v.add(rs2.getString("percentage\_value"));

v.add(rs2.getString("for\_teacher"));

v.add(rs2.getString("date"));

dtm3.addRow(v);

} System.out.println("sync complete");

} catch (Exception e) {

System.out.println(e);

} finally { try { if (rs2 != null) { rs2.close(); } } catch (SQLException e) {

System.out.println(e); } } }

private void brefresh\_tab1ActionPerformed(java.awt.event.ActionEvent evt) {

loadStudents();

tname.setText("");

taddress.setText("");

tmobile.setText("");

tschool.setText("");

stream\_list.setSelectedIndex(0); }

private void brefresh\_tab2ActionPerformed(java.awt.event.ActionEvent evt) {

combo\_batch.setSelectedIndex(0);

combo\_teacher.setSelectedIndex(0);

combo\_month.setSelectedIndex(0);

loadComps(); }

private void bhomeActionPerformed(java.awt.event.ActionEvent evt) {

this.dispose(); }

private void bbackupActionPerformed(java.awt.event.ActionEvent evt) {

h = new newHome();

h.dispose();

h = null;

this.dispose();

System.gc();

new RescueManager().setVisible(true); }

private void combo\_monthItemStateChanged(java.awt.event.ItemEvent evt) {

ResultSet rs2 = null;

if (combo\_month.getSelectedIndex() == 0

&& combo\_batch.getSelectedIndex() == 0

&& combo\_teacher.getSelectedIndex() == 0) {

loadComps();

} else if (combo\_month.getSelectedIndex() != 0) {

DefaultTableModel dtm3 = (DefaultTableModel) table\_comps.getModel();

while (dtm3.getRowCount() > 0) {

dtm3.removeRow(0); }

System.out.println("income shares resyncing");

try { rs2 = jdbc.con().createStatement().executeQuery("select \* from compensation"

+ " where month='" + combo\_month.getSelectedItem().toString() + "'");

while (rs2.next()) { Vector v = new Vector();

v.add(rs2.getString("name"));

v.add(rs2.getString("batch"));

v.add(rs2.getString("income"));

v.add(rs2.getString("percentage"));

v.add(rs2.getString("percentage\_value"));

v.add(rs2.getString("for\_teacher"));

v.add(rs2.getString("date"));

dtm3.addRow(v); }

System.out.println("sync complete");

} catch (Exception e) {

System.out.println(e);

} finally { try { if (rs2 != null) {

rs2.close(); }

} catch (SQLException e) { System.out.println(e); } } } }

private void combo\_batchItemStateChanged(java.awt.event.ItemEvent evt) {

ResultSet rs2 = null;

if (combo\_month.getSelectedIndex() == 0

&& combo\_batch.getSelectedIndex() == 0

&& combo\_teacher.getSelectedIndex() == 0) {

loadComps();

} else if (combo\_batch.getSelectedIndex() != 0) {

DefaultTableModel dtm3 = (DefaultTableModel) table\_comps.getModel();

while (dtm3.getRowCount() > 0) {

dtm3.removeRow(0);

} System.out.println("income shares resyncing");

try { rs2 = jdbc.con().createStatement().executeQuery("select \* from compensation"

+ " where batch='" + combo\_batch.getSelectedItem().toString() + "'");

while (rs2.next()) {

Vector v = new Vector();

v.add(rs2.getString("name"));

v.add(rs2.getString("batch"));

v.add(rs2.getString("income"));

v.add(rs2.getString("percentage"));

v.add(rs2.getString("percentage\_value"));

v.add(rs2.getString("for\_teacher"));

v.add(rs2.getString("date"));

dtm3.addRow(v); }

System.out.println("sync complete");

} catch (Exception e) {

System.out.println(e);

} finally { try {

if (rs2 != null) { rs2.close();} } catch (SQLException e) {

System.out.println(e); } } } }

private void combo\_teacherItemStateChanged(java.awt.event.ItemEvent evt) {

ResultSet rs2 = null;

if (combo\_month.getSelectedIndex() == 0

&& combo\_batch.getSelectedIndex() == 0

&& combo\_teacher.getSelectedIndex() == 0) {

loadComps();

} else if (combo\_teacher.getSelectedIndex() != 0) {

DefaultTableModel dtm3 = (DefaultTableModel) table\_comps.getModel();

while (dtm3.getRowCount() > 0) {

dtm3.removeRow(0);

} System.out.println("income shares resyncing");

try { rs2 = jdbc.con().createStatement().executeQuery("select \* from compensation"

+ " where name='" + combo\_teacher.getSelectedItem().toString() + "'");

while (rs2.next()) {

Vector v = new Vector();

v.add(rs2.getString("name"));

v.add(rs2.getString("batch"));

v.add(rs2.getString("income"));

v.add(rs2.getString("percentage"));

v.add(rs2.getString("percentage\_value"));

v.add(rs2.getString("for\_teacher"));

v.add(rs2.getString("date"));

dtm3.addRow(v); } System.out.println("sync complete");

} catch (Exception e) {

System.out.println(e);

} finally { try { if (rs2 != null) { rs2.close();

} } catch (SQLException e) {System.out.println(e); } } } }

private void bprintActionPerformed(java.awt.event.ActionEvent evt) {

MessageFormat header = new MessageFormat("Records Printing.");

MessageFormat footer = new MessageFormat("--Air Solutions--" + new Date().toString()); try {

table\_comps.print(JTable.PrintMode.FIT\_WIDTH, header, footer);

} catch (Exception e) {

System.err.format("Cannot Print %s%n", e.getMessage()); }

JOptionPane.showMessageDialog(null, "<html>Remember, You Can only Print the Sheet with your Printer!<br> "

+ "<font color=red><b>Do not Use other File Formats like .pdf .ps or .xps</b></font></html>",

"Remember", JOptionPane.WARNING\_MESSAGE); }

private void bprint1ActionPerformed(java.awt.event.ActionEvent evt) {

MessageFormat header = new MessageFormat("Records Printing.");

MessageFormat footer = new MessageFormat("--Air Solutions--" + new Date().toString()); try {

table\_stu.print(JTable.PrintMode.FIT\_WIDTH, header, footer);

} catch (Exception e) {

System.err.format("Cannot Print %s%n", e.getMessage()); }

JOptionPane.showMessageDialog(null, "<html>Remember, You Can only Print the Sheet with your Printer!<br> "

+ "<font color=red><b>Do not Use other File Formats like .pdf .ps or .xps</b></font></html>",

"Remember!", JOptionPane.WARNING\_MESSAGE); }

private void brefresh\_tab3ActionPerformed(java.awt.event.ActionEvent evt) {

loadTeachers(); }

private void bprint2ActionPerformed(java.awt.event.ActionEvent evt) {

MessageFormat header = new MessageFormat("Records Printing.");

MessageFormat footer = new MessageFormat("--Air Solutions--");

try { table\_tea1.print(JTable.PrintMode.FIT\_WIDTH, header, footer);

} catch (Exception e) {

System.err.format("Cannot Print %s%n", e.getMessage()); }

JOptionPane.showMessageDialog(null, "<html>Remember, You Can only Print the Sheet with your Printer!<br> "

+ "<font color=red><b>Do not Use other File Formats like .pdf .ps or .xps</b></font></html>",

"Remember", JOptionPane.WARNING\_MESSAGE); }

private void tnameKeyTyped(java.awt.event.KeyEvent evt) {

ResultSet rs1 = null;

if (!"".equals(tname.getText())) {

DefaultTableModel dtm1 = (DefaultTableModel) table\_stu.getModel();

while (dtm1.getRowCount() > 0) {

dtm1.removeRow(0); }

System.out.println("student list resyncing"); try {

rs1 = jdbc.con().createStatement().executeQuery("select \* from students2 where " + "name like '%" + tname.getText() + "%'");

while (rs1.next()) {

Vector v = new Vector(); v.add(rs1.getString("stu\_no")); v.add(rs1.getString("name")); v.add(rs1.getString("address")); v.add(rs1.getString("mobile")); v.add(rs1.getString("stream")); v.add(rs1.getString("guardian\_name")); v.add(rs1.getString("landline")); v.add(rs1.getString("school")); v.add(rs1.getString("nic"));

dtm1.addRow(v); }

System.out.println("sync complete");

} catch (Exception e) {

System.out.println("student table retrieval error");

System.out.println(e);

} finally { try { if (rs1 != null) { rs1.close(); } } catch (SQLException e) {

System.out.println(e); } } }}

private void tnameFocusGained(java.awt.event.FocusEvent evt) {

tname.setText(""); taddress.setText("");

tmobile.setText(""); tschool.setText(""); tname.grabFocus(); }

private void taddressFocusGained(java.awt.event.FocusEvent evt) {

tname.setText(""); taddress.setText("");

tmobile.setText(""); tschool.setText(""); taddress.grabFocus(); }

private void tmobileFocusGained(java.awt.event.FocusEvent evt) {

tname.setText(""); taddress.setText(""); tmobile.setText("");

tschool.setText(""); tmobile.grabFocus(); }

private void tschoolFocusGained(java.awt.event.FocusEvent evt) {

tname.setText("");

taddress.setText("");

tmobile.setText("");tschool.setText("");tschool.grabFocus(); }

private void taddressKeyTyped(java.awt.event.KeyEvent evt) {

ResultSet rs1 = null;

if (!"".equals(taddress.getText())) {

DefaultTableModel dtm1 = (DefaultTableModel) table\_stu.getModel();

while (dtm1.getRowCount() > 0) {

dtm1.removeRow(0); }

System.out.println("student list resyncing");

try { rs1 = jdbc.con().createStatement().executeQuery("select \* from students2 "

+ "where address like '%" + taddress.getText() + "%'");

while (rs1.next()) {

Vector v = new Vector(); v.add(rs1.getString("stu\_no")); v.add(rs1.getString("name")); v.add(rs1.getString("address")); v.add(rs1.getString("mobile")); v.add(rs1.getString("stream"));

v.add(rs1.getString("guardian\_name")); v.add(rs1.getString("landline")); v.add(rs1.getString("school")); v.add(rs1.getString("nic"));

dtm1.addRow(v); } System.out.println("sync complete");

} catch (Exception e) { System.out.println("student table retrieval error");

System.out.println(e);

} finally { try { if (rs1 != null) { rs1.close(); }} catch (SQLException e) {

System.out.println(e); } } } }

private void tmobileKeyTyped(java.awt.event.KeyEvent evt) {

ResultSet rs1 = null;

if (!"".equals(tmobile.getText())) {

DefaultTableModel dtm1 = (DefaultTableModel) table\_stu.getModel();

while (dtm1.getRowCount() > 0) {

dtm1.removeRow(0); }

System.out.println("student list resyncing");

try { rs1 = jdbc.con().createStatement().executeQuery("select \* from students2 where "

+ "mobile like '%" + tmobile.getText() + "%'");

while (rs1.next()) {

Vector v = new Vector(); v.add(rs1.getString("stu\_no")); v.add(rs1.getString("name")); v.add(rs1.getString("address")); v.add(rs1.getString("mobile"));v.add(rs1.getString("stream")); v.add(rs1.getString("guardian\_name")); v.add(rs1.getString("landline")); v.add(rs1.getString("school")); v.add(rs1.getString("nic")); dtm1.addRow(v); } System.out.println("sync complete");

} catch (Exception e) { System.out.println("student table retrieval error");

System.out.println(e); } finally { try { if (rs1 != null) { rs1.close(); } } catch (SQLException e) {

System.out.println(e); } } } }

private void tschoolKeyTyped(java.awt.event.KeyEvent evt) {

ResultSet rs1 = null;

if (!"".equals(tschool.getText())) {

DefaultTableModel dtm1 = (DefaultTableModel) table\_stu.getModel();

while (dtm1.getRowCount() > 0) {

dtm1.removeRow(0); }

System.out.println("student list resyncing");

try { rs1 = jdbc.con().createStatement().executeQuery("select \* from students2 where "

+ "school like '%" + tschool.getText() + "%'");

while (rs1.next()) {

Vector v = new Vector(); v.add(rs1.getString("stu\_no"));v.add(rs1.getString("name"));

v.add(rs1.getString("address"));v.add(rs1.getString("mobile")); v.add(rs1.getString("stream")); v.add(rs1.getString("guardian\_name"));

v.add(rs1.getString("landline"));v.add(rs1.getString("school")); v.add(rs1.getString("nic")); dtm1.addRow(v); } System.out.println("sync complete");

} catch (Exception e) { System.out.println("student table retrieval error");

System.out.println(e); } finally { try { if (rs1 != null) {

rs1.close();} } catch (SQLException e) {

System.out.println(e); } } } }

private void stream\_listItemStateChanged(java.awt.event.ItemEvent evt) {

ResultSet rs1 = null;

if (stream\_list.getSelectedIndex() == 0) {

loadStudents();

} else if (stream\_list.getSelectedIndex() != 0) {

DefaultTableModel dtm1 = (DefaultTableModel) table\_stu.getModel();

while (dtm1.getRowCount() > 0) {

dtm1.removeRow(0); }

System.out.println("student list resyncing");

try { rs1 = jdbc.con().createStatement().executeQuery("select \* from students2 where "

+ "stream='" + stream\_list.getSelectedItem().toString() + "'");

while (rs1.next()) {

Vector v = new Vector(); v.add(rs1.getString("stu\_no")); v.add(rs1.getString("name"));

v.add(rs1.getString("address")); v.add(rs1.getString("mobile")); v.add(rs1.getString("stream")); v.add(rs1.getString("guardian\_name"));v.add(rs1.getString("landline"));

v.add(rs1.getString("school")); v.add(rs1.getString("nic")); dtm1.addRow(v); }

System.out.println("sync complete");

} catch (Exception e) { System.out.println("student table retrieval error");

System.out.println(e); } finally { try { if (rs1 != null) { rs1.close(); }

} catch (SQLException e) { System.out.println(e); } } } }

public static void main(String args[]) {

try { for (javax.swing.UIManager.LookAndFeelInfo info : javax.swing.UIManager.getInstalledLookAndFeels()) {

if ("Nimbus".equals(info.getName())) {

javax.swing.UIManager.setLookAndFeel(info.getClassName());

break; } }

} catch (ClassNotFoundException ex) { java.util.logging.Logger.getLogger(All\_data.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); } catch (InstantiationException ex) { java.util.logging.Logger.getLogger(All\_data.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);

} catch (IllegalAccessException ex) { java.util.logging.Logger.getLogger(All\_data.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); } catch (javax.swing.UnsupportedLookAndFeelException ex) { java.util.logging.Logger.getLogger(All\_data.class.getName()).log(java.util.logging.Level.SEVERE, null, ex); }

java.awt.EventQueue.invokeLater(new Runnable() {

@Override

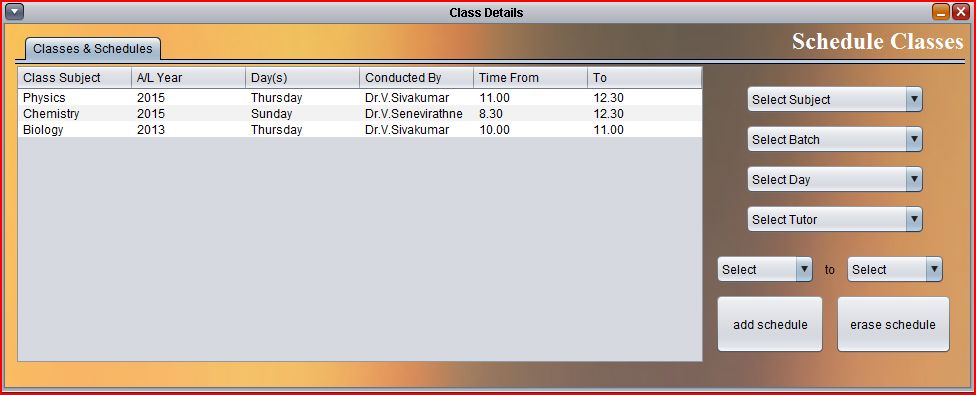
public void run() {

new All\_data().setVisible(true); }

});

}

**Class TimeTable**



package in\_frames;

import ….

public class In\_classes\_times extends javax.swing.JInternalFrame {

public In\_classes\_times() {

initComponents();

loadTable(); }

private void baddActionPerformed(java.awt.event.ActionEvent evt) {

ResultSet rs0 = null;

try { DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();

if (combo\_subject.getSelectedIndex() == 0 | combo\_day.getSelectedIndex() == 0

| combo\_batch.getSelectedIndex() == 0 | combo\_from.getSelectedIndex() == 0

| combo\_to.getSelectedIndex() == 0) {

JOptionPane.showMessageDialog(null, "You Cannot continue with empty fields", "Error", 2);} else { rs0 = jdbc.con().createStatement().executeQuery("select \* from classes");

if (rs0.first()) { if (rs0.getString("subject").equals(combo\_subject.getSelectedItem().toString())

&& rs0.getString("year").equals(combo\_batch.getSelectedItem().toString())

&& rs0.getString("day").equals(combo\_day.getSelectedItem().toString())

&& rs0.getString("teacher").equals(combo\_teacher.getSelectedItem().toString())

&& rs0.getString("fr").equals(combo\_from.getSelectedItem().toString())

&& rs0.getString("onto").equals(combo\_to.getSelectedItem().toString())) {

JOptionPane.showMessageDialog(null, "Duplicate Schedules cannot exist");

resetFields(); } else {

Vector v = new Vector(); v.add(combo\_subject.getSelectedItem()); v.add(combo\_batch.getSelectedItem()); v.add(combo\_day.getSelectedItem()); v.add(combo\_teacher.getSelectedItem());

v.add(combo\_from.getSelectedItem()); v.add(combo\_to.getSelectedItem()); dtm.addRow(v);

System.out.println("visual table updated");

jdbc.con().createStatement().executeUpdate("insert into classes"

+ "(subject,year,day,teacher,fr,onto)"

+ " values('" + combo\_subject.getSelectedItem() + "',"

+ " '" + combo\_batch.getSelectedItem().toString() + "',"

+ " '" + combo\_day.getSelectedItem() + "','" + combo\_teacher.getSelectedItem() + "'," + combo\_from.getSelectedItem().toString() +"',"

+ " '" + combo\_to.getSelectedItem().toString() + "') ");

System.out.println("SQL table updated");

resetFields(); for (int x = 0; x < dtm.getRowCount(); x++) {

dtm.removeRow(0); } loadTable(); } }}

} catch (Exception ex) { System.out.println(ex);} finally { try { if (rs0 != null) {

rs0.close();} } catch (SQLException e) {System.out.println(e);} }}

private void loadTeachers() {

ResultSet rs1 = null; try {

rs1 = jdbc.con().createStatement().executeQuery("select \* from tutor");

while (rs1.next()) { combo\_teacher.addItem(rs1.getString("name"));

} this.repaint(); } catch (Exception e) { System.out.println(e);

} finally { try { if (rs1 != null) { rs1.close(); } } catch (SQLException e) {

System.out.println(e); } } }

private void bremoveActionPerformed(java.awt.event.ActionEvent evt) {

ResultSet rs = null;

DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();

int sr = jTable2.getSelectedRow();

int showConfirmDialog = JOptionPane.showConfirmDialog(null, "Are You Sure To Proceed?", "Confirm Removal", JOptionPane.OK\_CANCEL\_OPTION);

if (showConfirmDialog == JOptionPane.OK\_OPTION) {

try { rs = jdbc.con().createStatement().executeQuery("select id from classes where "

+ " subject='" + dtm.getValueAt(sr, 0) + "' AND year='" + dtm.getValueAt(sr, 1) + "' "

+ " AND day='" + dtm.getValueAt(sr, 2) + "' AND teacher='" + dtm.getValueAt(sr, 3) + "' "

+ " AND fr='" + dtm.getValueAt(sr, 4) + "' AND onto='" + dtm.getValueAt(sr, 5) + "'");

if (rs.first()) { jdbc.con().createStatement().executeUpdate("delete from classes where "

+ "id='" + rs.getString("id") + "'"); System.out.println("SQL row deleted");

} } catch (Exception ex) {

System.out.println("error in deleting class schedule");

System.out.println(ex); } finally {

try { if (rs != null) { rs.close(); } } catch (SQLException e) {System.out.println(e); } }

dtm.removeRow(jTable2.getSelectedRow());

System.out.println("Table row deleted"); }

loadTable(); resetFields(); }

private void combo\_teacherFocusGained(java.awt.event.FocusEvent evt) {

combo\_teacher.removeAllItems();

loadTeachers(); }

void resetFields() {

combo\_day.setSelectedIndex(0);

combo\_subject.setSelectedIndex(0);

combo\_teacher.removeAllItems();

combo\_teacher.addItem("Select Tutor");

combo\_batch.setSelectedIndex(0);

combo\_from.setSelectedIndex(0);

combo\_to.setSelectedIndex(0);

} private void loadTable() {

ResultSet rs = null;

DefaultTableModel dtm = (DefaultTableModel) jTable2.getModel();

while (dtm.getRowCount() > 0) {

dtm.removeRow(0); }

try { rs = jdbc.con().createStatement().executeQuery("select \* from classes");

while (rs.next()) {

Vector v = new Vector();v.add(rs.getString("subject")); v.add(rs.getString("year"));

v.add(rs.getString("day"));v.add(rs.getString("teacher"));v.add(rs.getString("fr"));

v.add(rs.getString("onto")); dtm.addRow(v); }

this.repaint(); } catch (Exception e) { System.out.println(e);} finally {

try { if (rs != null){ rs.close();} } catch (SQLException e) { System.out.println(e); } }}

# Database Schema

**attendance**

id , stu\_no , name , stream , date , systemtime , batch , PRIMARY KEY id,stu\_no

**classes**

id , subject , year , day , teacher , fr , onto , PRIMARY KEY id

**compensation**

idperc , month , batch , tutor\_no , name , income , percentage , percentage\_value , for\_teacher , date , useragent , PRIMARY KEY idperc

**login\_sessions**

idlogin\_sessions , loginAt , logoutAt , users\_uid , users\_username , status , PRIMARY KEY idlogin\_sessions,

**payments**

idpayment , stu\_no , nic , total\_fee , paid , due , month , PRIMARY KEY idpayment,stu\_no,

**students**

id , stu\_no , name , address , landline , mobile , stream , medium , guardian\_name , school , subj1 , subj 2 , subj 3 , reg\_fee , nic , PRIMARY KEY id,stu\_no

**subjects**

idsubjects , name , forGrade , PRIMARY KEY idsubjects

**tutor**

tutor\_no , nic , name , address , mobile , stream , subject , vehicle ,  
 PRIMARY KEY tutor\_no,nic,mobile

**users**

uid , name , username , password , PRIMARY KEY uid,username

# Notes