

**Java Institute for Advanced Technology**

UNIT NAME : MANULA KAVISHKA

UNIT ID : H7E1 04

NAME : MANULA KAVISHKA

SCN NO

NIC : 200114902210

BRANCH : GAMPAHA/ZEUS



Com.hms.common package

package com.hms.common;

public class MD5 {

public static String getMd5(String input)

{

try {

// Static getInstance method is called with hashing MD5

MessageDigest md = MessageDigest.getInstance("MD5");

// digest() method is called to calculate message digest

// of an input digest() return array of byte

byte[] messageDigest = md.digest(input.getBytes());

// Convert byte array into signum representation

BigInteger no = new BigInteger(1, messageDigest);

// Convert message digest into hex value

String hashtext = no.toString(16);

while (hashtext.length() < 32) {

hashtext = "0" + hashtext;

}

return hashtext;

}

// For specifying wrong message digest algorithms

catch (NoSuchAlgorithmException e) {

throw new RuntimeException(e);

}

}

}

public class SystemConfigData {

private static String activeUser;

private static String userType;

public static String getActiveUser() {

return activeUser;

}

public static void setActiveUser(String activeUser) {

SystemConfigData.activeUser = activeUser;

}

public static String getUserType() {

return userType;

}

public static void setUserType(String userType) {

SystemConfigData.userType = userType;

}

}

public class SystemVariableList {

public static final String REPORT\_BASE\_URL = "C:\\Users\\User\\Desktop\\HMS Final Project\\REPORT\\";

}

public class timeDate {

private static Date date;

private static SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd-HH:mm:ss");

public void timeAndDate(JLabel x) {

Timer timer = new Timer(1000, new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

date = new Date();

JLabel jLabel1 = x;

jLabel1.setText(sdf.format(date));

}

});

timer.start();

}

}

Com.hms.db package

public class db {

private static Connection connection;

private static void init\_db() throws ClassNotFoundException,SQLException {

Class.forName("com.mysql.cj.jdbc.Driver");

connection = DriverManager.getConnection("jdbc:mysql://127.0.0.1:3306/hms", "root", "1234");

}

public static void push(String sql) throws Exception{

if (connection == null) {

init\_db();

}

connection.createStatement().executeUpdate(sql);

}

public static ResultSet search(String sql) throws Exception{

if (connection==null){

init\_db();

}

return connection.createStatement().executeQuery(sql);

}

public static Connection getDBConnection() throws Exception{

if (connection == null) {

init\_db();

}

return connection;

}

}

package com.hms.dto;

public class treatmentQtyDto {

private double qty;

private static String rid;

private static String nic;

private static String PARID;

public static String getPARID() {

return PARID;

}

public static void setPARID(String PARID) {

treatmentQtyDto.PARID = PARID;

}

public static String getNic() {

return nic;

}

public static void setNic(String nic) {

treatmentQtyDto.nic = nic;

}

public static String getRid() {

return rid;

}

public static void setRid(String arid) {

rid = arid;

}

public double getQty() {

return qty;

}

public void setQty(double qty) {

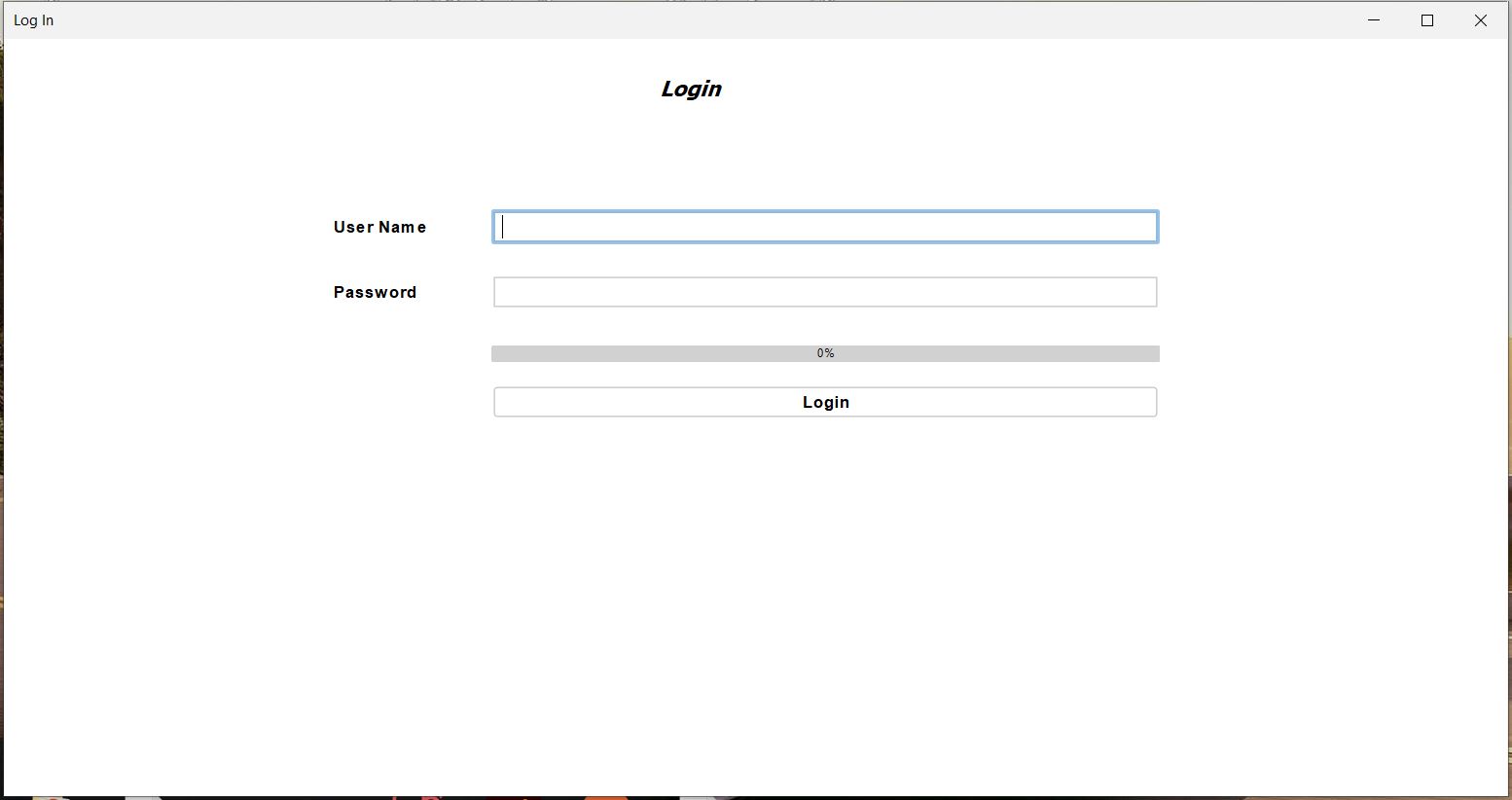
this.qty = qty;

}

}

package com.hms.gui;

Login



Main method

public static void main(String args[]) {

/\* Set the Nimbus look and feel \*/

FlatIntelliJLaf.setup();

/\* Create and display the form \*/

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

public void run() {

new login().setVisible(true);

}

});

}

Constractors

public login() {

initComponents();

setLocationRelativeTo(null);

}

other classes

JFrame frame;

private void loading() {

frame = this;

Thread thread = new Thread(new Runnable() {

@Override

public void run() {

for (int i = 0; i < 101; i++){

jProgressBar1.setValue(i);

try {

Thread.sleep(20);

} catch (Exception e) {

}

if (i == 5) {

File file = new File("C:/system.cfg");

if (!file.exists()) {

JOptionPane.showMessageDialog(frame, "System.cfg File not exists..!!");

break;

}

}

if (i == 100) {

Main homeView = new Main();

homeView.setVisible(true);

dispose();

}

}

}

});

thread.start();

}

Login button

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

try {

String username = txt\_username.getText();

String password = new String(txt\_password.getPassword());

Connection connection = db.getDBConnection();

String sql = "select \* from system\_user where username=? and password=? AND status\_id = '1'";

PreparedStatement ps = connection.prepareStatement(sql);

ps.setString(1, username);

String encPw = MD5.getMd5(password);

ps.setString(2, encPw);

ResultSet rs = ps.executeQuery();

if (rs.next()){

SystemConfigData.setActiveUser(username);

SystemConfigData.setUserType(rs.getString("staff\_type\_id"));

loading();

} else {

JOptionPane.showMessageDialog(this, "Invalid Username Or Password","warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Username Text field

private void txt\_usernameActionPerformed(java.awt.event.ActionEvent evt) {

txt\_password.grabFocus();

}

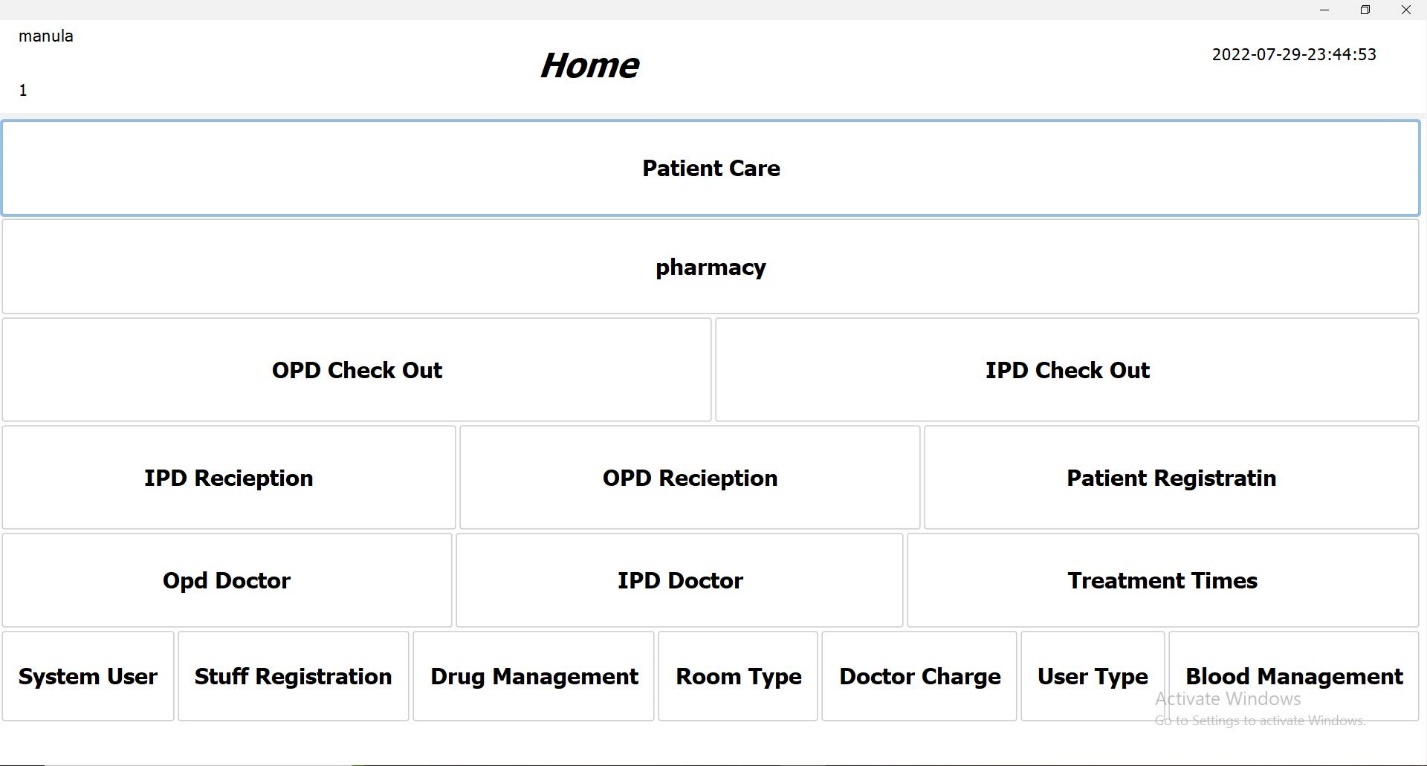
Password Text field

private void txt\_passwordActionPerformed(java.awt.event.ActionEvent evt) {

jButton1.grabFocus();

}

Home



Contractors And Variables

private static Date date;

private static SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd-HH:mm:ss");

public Main() {

initComponents();

setLocationRelativeTo(null);

timeAndDate();

jLabel1.setText(SystemConfigData.getActiveUser());

jLabel2.setText(SystemConfigData.getUserType());

if (SystemConfigData.getUserType().equals("1")) {

btn\_pc.setVisible(true);

btn\_ph.setVisible(true);

btn\_opd\_co.setVisible(true);

btn\_ipd\_co.setVisible(true);

btn\_pr.setVisible(true);

btn\_ipd\_r.setVisible(true);

btn\_opd\_r.setVisible(true);

btn\_opd\_d.setVisible(true);

btn\_ipd\_d.setVisible(true);

btn\_tt.setVisible(true);

btn\_su.setVisible(true);

btn\_sr.setVisible(true);

btn\_dm.setVisible(true);

btn\_rt.setVisible(true);

btn\_dc.setVisible(true);

btn\_ut.setVisible(true);

btn\_bm.setVisible(true);

patientCount();

checkout();

pharmacy();

patientCare();

} else if (SystemConfigData.getUserType().equals("2")) {

btn\_opd\_d.setVisible(true);

btn\_ipd\_d.setVisible(true);

btn\_tt.setVisible(true);

btn\_pc.setVisible(false);

btn\_ph.setVisible(false);

btn\_opd\_co.setVisible(false);

btn\_ipd\_co.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

patientCount();

} else if (SystemConfigData.getUserType().equals("3")) {

btn\_pc.setVisible(false);

btn\_ph.setVisible(false);

btn\_opd\_co.setVisible(false);

btn\_ipd\_co.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

btn\_pr.setVisible(true);

btn\_ipd\_r.setVisible(true);

btn\_opd\_r.setVisible(true);

} else if (SystemConfigData.getUserType().equals("4")) {

btn\_pc.setVisible(false);

btn\_opd\_co.setVisible(false);

btn\_ipd\_co.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

btn\_ph.setVisible(true);

pharmacy();

} else if (SystemConfigData.getUserType().equals("5")) {

btn\_pc.setVisible(false);

btn\_ph.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

btn\_opd\_co.setVisible(true);

btn\_ipd\_co.setVisible(true);

checkout();

} else if (SystemConfigData.getUserType().equals("6")) {

btn\_pc.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

btn\_ph.setVisible(true);

btn\_opd\_co.setVisible(true);

btn\_ipd\_co.setVisible(true);

checkout();

pharmacy();

} else if (SystemConfigData.getUserType().equals("7")) {

btn\_pc.setVisible(false);

btn\_ph.setVisible(false);

btn\_opd\_co.setVisible(false);

btn\_ipd\_co.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(true);

btn\_sr.setVisible(true);

btn\_dm.setVisible(true);

btn\_rt.setVisible(true);

btn\_dc.setVisible(true);

btn\_ut.setVisible(true);

btn\_bm.setVisible(true);

patientCount();

checkout();

pharmacy();

patientCare();

} else if (SystemConfigData.getUserType().equals("8")) {

btn\_pc.setVisible(true);

btn\_ph.setVisible(false);

btn\_opd\_co.setVisible(false);

btn\_ipd\_co.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

patientCare();

}

else {

btn\_pc.setVisible(false);

btn\_ph.setVisible(false);

btn\_opd\_co.setVisible(false);

btn\_ipd\_co.setVisible(false);

btn\_pr.setVisible(false);

btn\_ipd\_r.setVisible(false);

btn\_opd\_r.setVisible(false);

btn\_opd\_d.setVisible(false);

btn\_ipd\_d.setVisible(false);

btn\_tt.setVisible(false);

btn\_su.setVisible(false);

btn\_sr.setVisible(false);

btn\_dm.setVisible(false);

btn\_rt.setVisible(false);

btn\_dc.setVisible(false);

btn\_ut.setVisible(false);

btn\_bm.setVisible(false);

}

}

Other classes

private void timeAndDate() {

Timer timer = new Timer(1000, new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

date = new Date();

jLabel4.setText(sdf.format(date));

}

});

timer.start();

}

private void patientCount() {

try {

String sql = "SELECT COUNT(\*) AS c FROM opd\_reception WHERE stat = '3'";

ResultSet rs = db.search(sql);

rs.next();

jLabel6.setText(rs.getString("c"));

String sql2 = "SELECT COUNT(\*) AS c FROM ipd\_p\_a\_registration WHERE status\_id in ('3', '2' )";

ResultSet rs2 = db.search(sql2);

rs2.next();

jLabel7.setText(rs2.getString("c"));

} catch (Exception ex) {

ex.printStackTrace();

}

}

private void checkout() {

try {

String searchSql = "SELECT COUNT(\*) AS c FROM opd\_reception WHERE stat = '1'";

ResultSet rs = db.search(searchSql);

rs.next();

jLabel14.setText(rs.getString("c"));

String searchSql2 = "SELECT COUNT(\*) AS c FROM ipd\_p\_a\_registration WHERE status\_id = '1'";

ResultSet rs2 = db.search(searchSql2);

rs2.next();

jLabel15.setText(rs2.getString("c"));

} catch (Exception e) {

e.printStackTrace();

}

}

private void patientCare() {

try {

String searchSql = "SELECT COUNT(\*) AS c FROM ipd\_p\_a\_registration WHERE status\_id = '2'";

ResultSet rs = db.search(searchSql);

rs.next();

jLabel13.setText(rs.getString("c"));

} catch (Exception e) {

e.printStackTrace();

}

}

private void pharmacy() {

try {

String sql = "SELECT COUNT(\*) AS c FROM opd\_reception WHERE stat = '2'";

ResultSet rs = db.search(sql);

rs.next();

jLabel9.setText(rs.getString("c"));

} catch (Exception e) {

e.printStackTrace();

}

}

Patient Care Button

private void btn\_pcActionPerformed(java.awt.event.ActionEvent evt) {

patientCare pc = new patientCare();

pc.setVisible(true);

dispose();

}

Pharmacy Button

private void btn\_phActionPerformed(java.awt.event.ActionEvent evt) {

Pharmacy p = new Pharmacy();

p.setVisible(true);

dispose();

}

OPD Check Out Button

private void btn\_opd\_coActionPerformed(java.awt.event.ActionEvent evt) {

CheckOut co = new CheckOut();

co.setVisible(true);

dispose();

}

IPD Check Out Button

private void btn\_ipd\_coActionPerformed(java.awt.event.ActionEvent evt) {

IPD\_checkOut ic = new IPD\_checkOut();

ic.setVisible(true);

dispose();

}

IPD Reception Button

private void btn\_ipd\_rActionPerformed(java.awt.event.ActionEvent evt) {

PatientAddmittedRegistration par = new PatientAddmittedRegistration();

par.setVisible(true);

dispose();

}

OPD Reception Button

private void btn\_opd\_rActionPerformed(java.awt.event.ActionEvent evt) {

OPDReceiption opd = new OPDReceiption();

opd.setVisible(true);

dispose();

}

Patient Registration Button

private void btn\_prActionPerformed(java.awt.event.ActionEvent evt) {

PatientRegistration patient = new PatientRegistration();

patient.setVisible(true);

dispose();

}

Opd Doctor Button

private void btn\_opd\_dActionPerformed(java.awt.event.ActionEvent evt) {

OPD\_Doctor od = new OPD\_Doctor();

od.setVisible(true);

dispose();

}

IPD Doctor Button

private void btn\_ipd\_dActionPerformed(java.awt.event.ActionEvent evt) {

IPD\_Doctor id = new IPD\_Doctor();

id.setVisible(true);

dispose();

}

Treatment Times Button

private void btn\_ttActionPerformed(java.awt.event.ActionEvent evt) {

TreatmentTimes tt = new TreatmentTimes();

tt.setVisible(true);

dispose();

}

System User Button

private void btn\_suActionPerformed(java.awt.event.ActionEvent evt) {

SystemUserManagement su = new SystemUserManagement();

su.setVisible(true);

dispose();

}

Stuff Registration Button

private void btn\_srActionPerformed(java.awt.event.ActionEvent evt) {

stuffMangement sm = new stuffMangement();

sm.setVisible(true);

dispose();

}

Drug Management Button

private void btn\_dmActionPerformed(java.awt.event.ActionEvent evt) {

DrugManagement dm = new DrugManagement();

dm.setVisible(true);

dispose();

}

Room Type Button

private void btn\_rtActionPerformed(java.awt.event.ActionEvent evt) {

RoomType rt = new RoomType();

rt.setVisible(true);

dispose();

}

Doctor Charge Button

private void btn\_dcActionPerformed(java.awt.event.ActionEvent evt) {

DoctorCharge dc = new DoctorCharge();

dc.setVisible(true);

dispose();

}

User Type Button

private void btn\_utActionPerformed(java.awt.event.ActionEvent evt) {

UserType ut = new UserType();

ut.setVisible(true);

dispose();

}

Blood Management Button

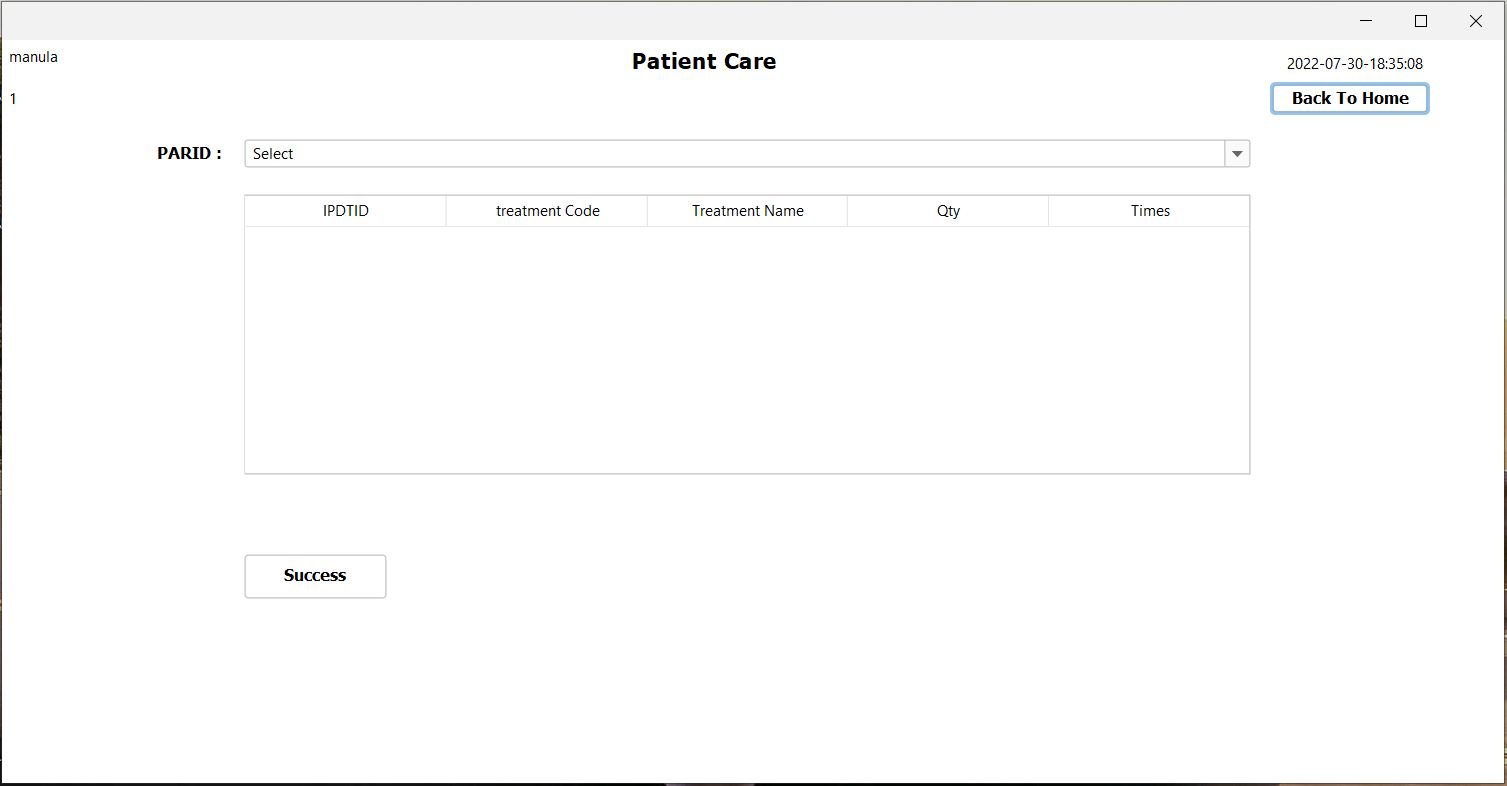
private void btn\_bmActionPerformed(java.awt.event.ActionEvent evt) {

bloodRegistration br = new bloodRegistration();

br.setVisible(true);

}

Patient Care



Constructors and other variables

public patientCare() {

initComponents();

setLocationRelativeTo(null);

loadPARID();

AutoCompleteDecorator.decorate(jComboBox1);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methers

private void loadPARID() {

try {

String searchSql = "SELECT \* FROM ipd\_p\_a\_registration WHERE status\_id = '2'";

ResultSet rs = db.search(searchSql);

Vector v = new Vector();

v.add("Select");

while (rs.next()){

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

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

PARID Combo box

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String PARID = jComboBox1.getSelectedItem().toString();

if (!PARID.equals("Select")) {

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

dtm.setRowCount(0);

String sql = "SELECT \* FROM ipd\_treatment WHERE PARID = '"+PARID+"' && status\_id = '1'";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

String sqlTreatment = "SELECT name FROM drug WHERE drug\_id = '"+rs.getString("treatment")+"'";

ResultSet rsTreatment = db.search(sqlTreatment);

if (rsTreatment.next())

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

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

String sqlTime = "SELECT time FROM treatment\_times WHERE code = '"+rs.getString("times")+"'";

ResultSet rsTime = db.search(sqlTime);

if (rsTime.next())

v.add(rsTime.getString("time"));

dtm.addRow(v);

}

} else {

JOptionPane.showMessageDialog(this, "Please Select RID!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Success button

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

try {

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

int selectRow = jTable1.getSelectedRow();

String data = (String) dtm.getValueAt(selectRow, 0);

String sql = "UPDATE ipd\_treatment SET status\_id = '2' WHERE IPDTID = '"+data+"'";

db.push(sql);

loadPARID();

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

dtm.removeRow(selectRow);

} catch (ArrayIndexOutOfBoundsException e) {

JOptionPane.showMessageDialog(this, "please Select Row","warning",JOptionPane.WARNING\_MESSAGE);

} catch (Exception e) {

e.printStackTrace();

}

}

Back to Home button

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

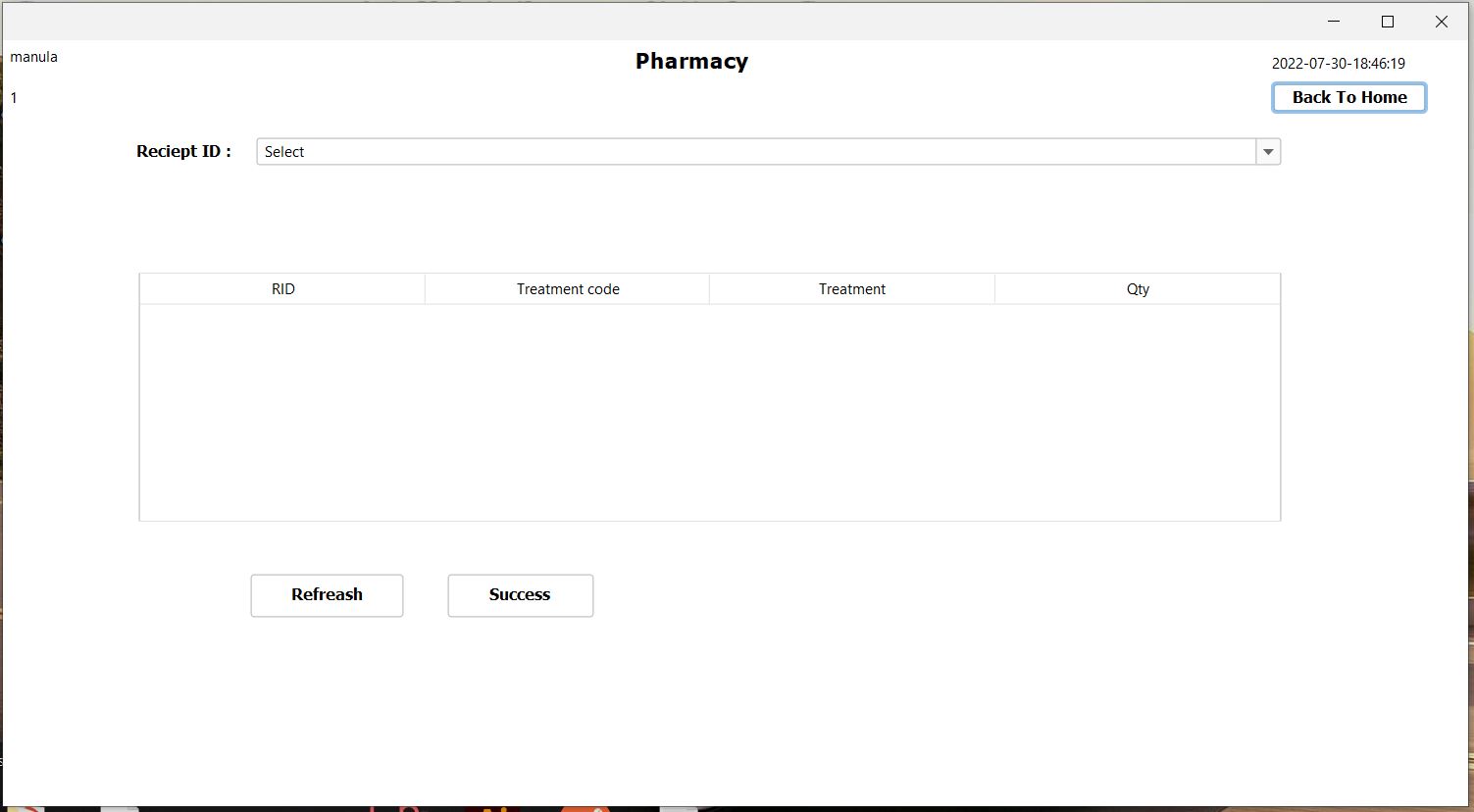
Main main = new Main();

main.setVisible(true);

dispose();

}

Pharmacy



Constructors and other variables

public Pharmacy() {

initComponents();

AutoCompleteDecorator.decorate(combo\_rid);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

loadRid();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

other Methods

private void loadRid() {

try {

String sql = "SELECT \* FROM opd\_reception WHERE stat = '2'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("RID");

v.add(data);

}

combo\_rid.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

Reception Id combo box

private void combo\_ridActionPerformed(java.awt.event.ActionEvent evt) {

try {

String rid = combo\_rid.getSelectedItem().toString();

if (!rid.equals("Select")) {

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

String sql = "SELECT \* FROM treatment WHERE Receipt\_id = '"+rid+"' && status\_id = '1'";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

v.add(rs.getString("Receipt\_id"));

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

String sqlTreatment = "SELECT name FROM drug WHERE drug\_id = '"+rs.getString("treatment")+"'";

ResultSet rsTreatment = db.search(sqlTreatment);

if (rsTreatment.next())

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

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

dtm.addRow(v);

}

} else {

JOptionPane.showMessageDialog(this, "Please Select RID!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Refresh button

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

Pharmacy p = new Pharmacy();

p.setVisible(true);

dispose();

}

Success Button

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

try {

if (!combo\_rid.getSelectedItem().toString().equals("Select")) {

String sql = "UPDATE treatment SET status\_id = '2' WHERE Receipt\_id = '"+combo\_rid.getSelectedItem().toString()+"'";

db.push(sql);

String sql2 = "UPDATE opd\_reception SET stat = '1' WHERE RID = '"+combo\_rid.getSelectedItem().toString()+"'";

db.push(sql2);

loadRid();

JOptionPane.showMessageDialog(this, "success","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Please Select RID","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Back to Home button

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

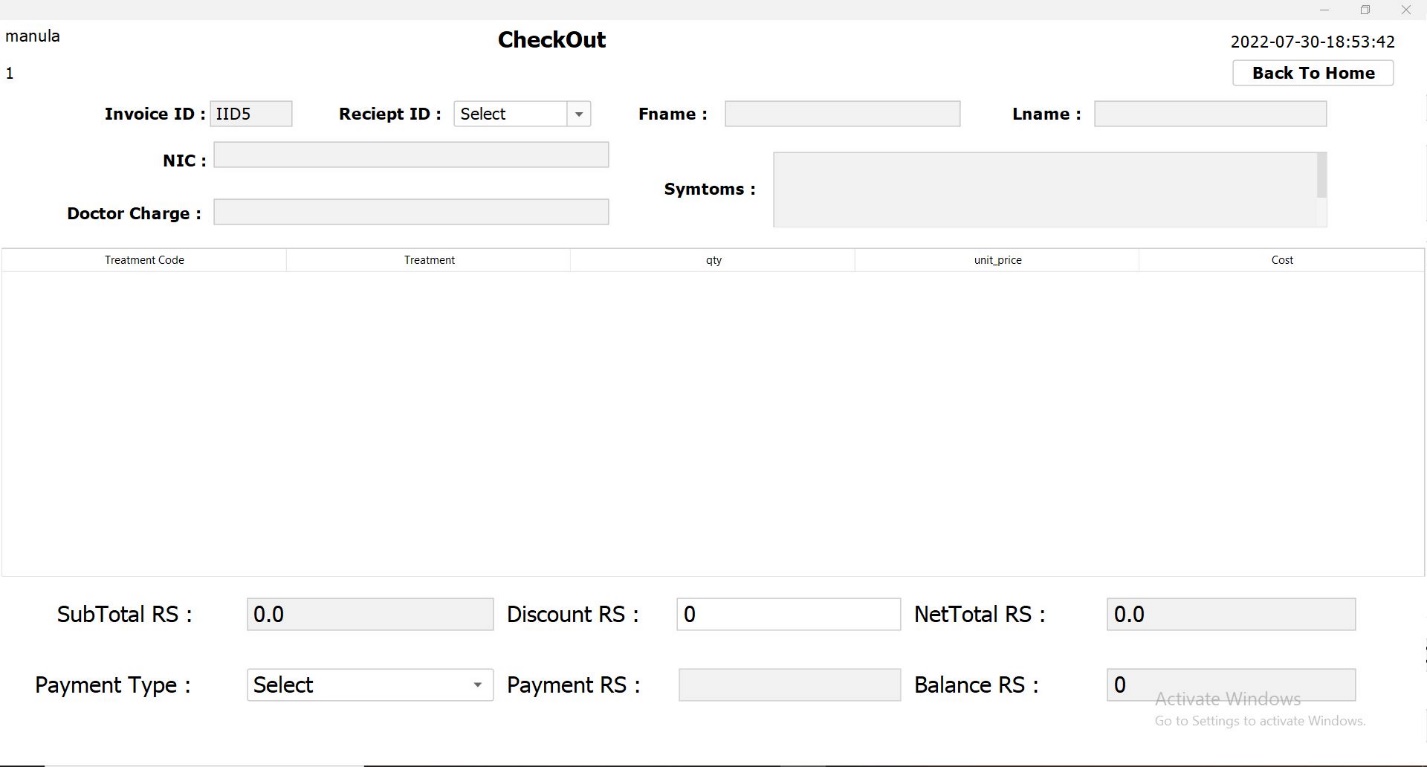
Main main = new Main();

main.setVisible(true);

dispose();

}

OPD Check Out



Constructs and variables

public CheckOut() {

initComponents();

setLocationRelativeTo(null);

AutoCompleteDecorator.decorate(jComboBox1);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

loadRID();

genarateIid();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

loadPaymentType();

}

Other methods

private void loadRID() {

try {

String searchSql = "SELECT \* FROM opd\_reception WHERE stat = '1'";

ResultSet rs = db.search(searchSql);

Vector v = new Vector();

v.add("Select");

while (rs.next()){

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

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void genarateIid() {

try {

String sql = "SELECT COUNT(\*) AS countIID FROM invoice";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("countIID");

String IID = "IID" + (++count);

txt\_iid.setText(IID);

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void createNewInvoice() {

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

dtm.setRowCount(0);

}

private void printInvoice(String iid) {

try {

// String path = "C:\\Users\\User\\JaspersoftWorkspace\\MyReports\\invoice\_main1.jasper";

String path= SystemVariableList.REPORT\_BASE\_URL+ "invoice\_main1.jasper";

String path2 = "C:\\Users\\User\\JaspersoftWorkspace\\MyReports\\invoice\_main1.jrxml";

InputStream inputStream = new FileInputStream(path);

// JasperReport jr = JasperCompileManager.compileReport(path2);

HashMap params = new HashMap();

params.put("iid", iid);

params.put("subReport", SystemVariableList.REPORT\_BASE\_URL+ "invoice\_sub1.jasper");

JasperPrint fillReport = JasperFillManager.fillReport(inputStream, params, db.getDBConnection());

JasperViewer.viewReport(fillReport,false);

} catch (Exception e) {

e.printStackTrace();

}

}

public void loadPaymentType() {

try {

String query = "SELECT `name` FROM `payment\_type`";

ResultSet rs = db.search(query);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

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

}

jComboBox2.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

Reciept ID Combo box

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String rid = jComboBox1.getSelectedItem().toString();

//patient Details

if (!rid.equals("Select")) {

String patientSql = "SELECT \* FROM patient WHERE nic = (SELECT patient\_id FROM opd\_reception WHERE RID = '"+rid+"')";

ResultSet patientRs = db.search(patientSql);

if (patientRs.next()) {

txt\_nic.setText(patientRs.getString("nic"));

txt\_fname.setText(patientRs.getString("fname"));

txt\_lname.setText(patientRs.getString("lname"));

}

String symtomSql = "SELECT symptoms FROM opd\_doctor WHERE Reciept\_id = '"+rid+"'";

ResultSet symtomRs = db.search(symtomSql);

if (symtomRs.next()) {

txt\_symtoms.setText(symtomRs.getString("symptoms"));

}

String dChargeSql = "SELECT charge FROM doctor\_charge WHERE code = (SELECT doctor\_charge FROM opd\_reception WHERE RID = '"+rid+"')";

ResultSet dChargRs = db.search(dChargeSql);

if (dChargRs.next()) {

txt\_dcharge.setText(dChargRs.getString("charge"));

}

}

//tatble implement

if (!rid.equals("Select")) {

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

String sql = "SELECT \* FROM treatment WHERE Receipt\_id = '"+rid+"' && status\_id = '2'";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

String sqlTreatment = "SELECT name FROM drug WHERE drug\_id = '"+rs.getString("treatment")+"'";

ResultSet rsTreatment = db.search(sqlTreatment);

if (rsTreatment.next()){

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

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

String sqlCost = "SELECT cost FROM drug WHERE drug\_id = '"+rs.getString("treatment")+"'";

ResultSet rsCost = db.search(sqlCost);

if (rsCost.next()){

double cost = rsCost.getDouble("cost");

double qty = rs.getDouble("qty");

double totalcost = cost \* qty;

v.add(rsCost.getString("cost"));

v.add(totalcost);

}

}

dtm.addRow(v);

}

}

//payment

if (!rid.equals("Select")) {

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

double subt = 0 ;

double dcharge = Double.parseDouble(txt\_dcharge.getText());

for (int row = 0; row < dtm.getRowCount(); row ++) {

subt += Double.parseDouble(dtm.getValueAt(row, 4).toString()) ;

}

double total = subt + dcharge;

txt\_subt.setText(""+ total);

// System.out.println(subt + dcharge);

txt\_dis.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Please Select RID!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Discount RS : Text field

private void txt\_disActionPerformed(java.awt.event.ActionEvent evt) {

if (txt\_subt.getText() !=null && !txt\_dis.getText().trim().equals("")){

double discount = Double.parseDouble(txt\_dis.getText());

double total = Double.parseDouble(txt\_subt.getText());

if (total > 500) {

txt\_net.setText(""+ (total - discount));

} else {

JOptionPane.showMessageDialog(this, "This Invoice RID not eligible for Discount","warning",JOptionPane.WARNING\_MESSAGE);

txt\_dis.setText("0");

txt\_net.setText(""+ (total));

}

jComboBox2.grabFocus();

}

}

private void txt\_disKeyTyped(java.awt.event.KeyEvent evt) {

String price = txt\_dis.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Payment Type : Combo box

private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {

// TODO add your handling code here:

String text = jComboBox2.getSelectedItem().toString();

if (text.equals("Select")) {

txt\_pay.setEnabled(false);

txt\_pay.setText("");

txt\_bala.setForeground(Color.BLACK);

} else {

txt\_pay.setEnabled(true);

}

}

Payment RS : Text filed

private void txt\_payActionPerformed(java.awt.event.ActionEvent evt) {

if (txt\_pay.getText() !=null && !txt\_pay.getText().trim().equals("")){

double net = Double.parseDouble(txt\_net.getText());

double pay = Double.parseDouble(txt\_pay.getText());

if (pay >= net) {

txt\_bala.setText(""+(pay-net));

txt\_bala.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Payment Not Enough","warning",JOptionPane.WARNING\_MESSAGE);

txt\_pay.setText("");

txt\_pay.grabFocus();

}

}

}

private void txt\_payKeyTyped(java.awt.event.KeyEvent evt) {

String price = txt\_pay.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Balance RS : Text filed

private void txt\_balaActionPerformed(java.awt.event.ActionEvent evt) {

try {

String iid = txt\_iid.getText();

String rid = jComboBox1.getSelectedItem().toString();

String nic = txt\_nic.getText();

String username = jLabel8.getText();

String total = txt\_subt.getText();

String discount = txt\_dis.getText();

String nettotal = txt\_net.getText();

String payment = txt\_pay.getText();

String paymentType = jComboBox2.getSelectedItem().toString();

String searchsql = "SELECT invoice\_id FROM invoice WHERE invoice\_id = '"+iid+"'";

ResultSet rs = db.search(searchsql);

if (!rs.next()) {

ResultSet rs2 = db.search("SELECT \* FROM `payment\_type` WHERE `name` = '" + paymentType + "'");

rs2.next();

String paymentTypeId = rs2.getString("id");

String invoice\_sql = "INSERT INTO invoice VALUES ('"+iid+"','"+rid+"'"

+ ",'"+nic+"','"+username+"',CURRENT\_TIMESTAMP(),'"+total+"'"

+ ",'"+discount+"','"+nettotal+"','"+payment+"','"+paymentTypeId+"')";

db.push(invoice\_sql);

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

for (int row = 0; row< dtm.getRowCount(); row++) {

String treatmentCode = dtm.getValueAt(row, 0).toString();

String qty = dtm.getValueAt(row, 2).toString();

String uprice = dtm.getValueAt(row, 3).toString();

String cost = dtm.getValueAt(row, 4).toString();

String invoiceDrugSql = "INSERT INTO invoice\_drug (invoice\_id,drug\_id,qty,unitt\_price,total) VALUES ('"+iid+"'"

+ ",'"+treatmentCode+"','"+qty+"','"+uprice+"','"+cost+"')";

db.push(invoiceDrugSql);

String RidSql = "UPDATE opd\_reception SET stat = '0' WHERE RID = '"+rid+"'";

db.push(RidSql);

}

} else {

JOptionPane.showMessageDialog(this, "Already have a invoice id","warning",JOptionPane.WARNING\_MESSAGE);

}

printInvoice(iid);

createNewInvoice();

loadRID();

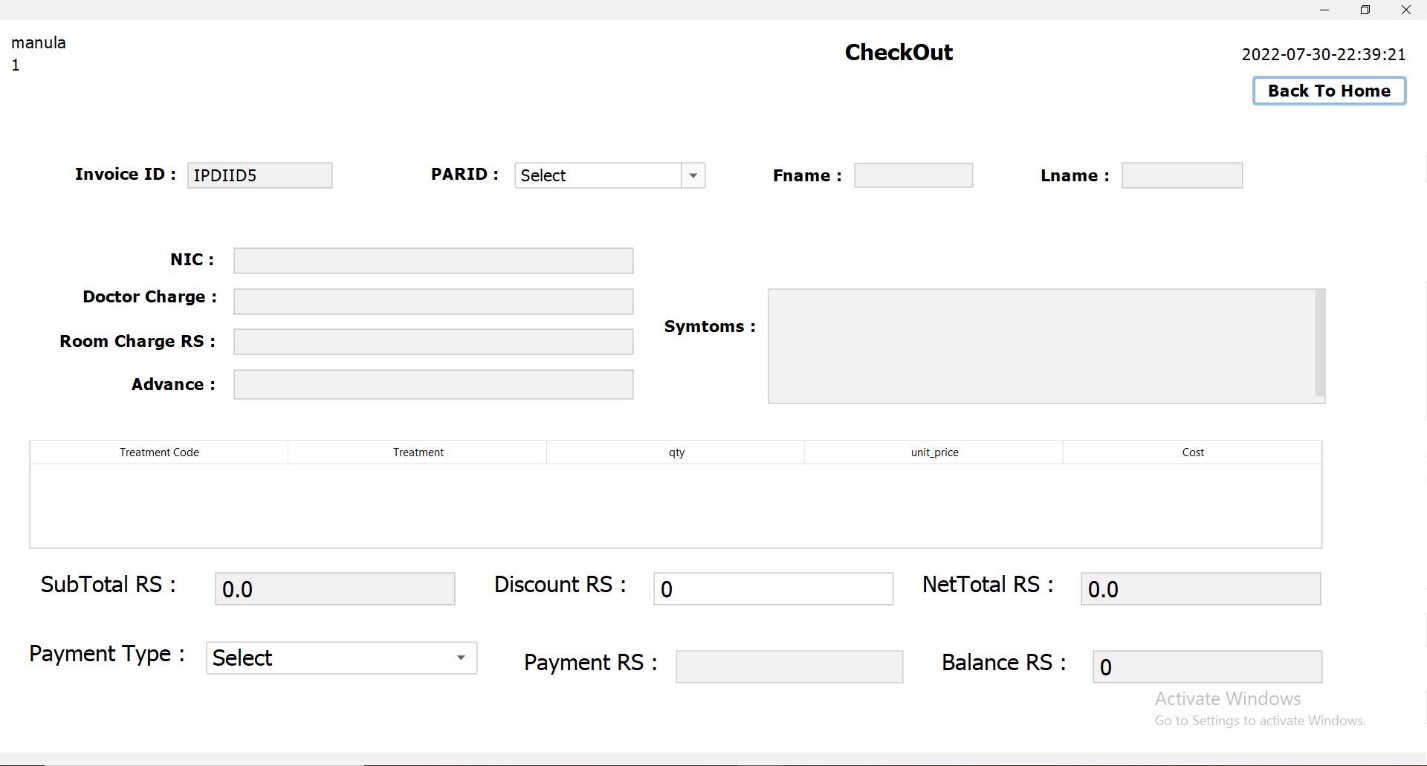
} catch (Exception e) {

e.printStackTrace();

}

}

IPD Check out



Constructs and variables

public IPD\_checkOut() {

initComponents();

setLocationRelativeTo(null);

AutoCompleteDecorator.decorate(jComboBox1);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

loadPARID();

genarateIid();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

loadPaymentType();

}

Other methods

private void genarateIid() {

try {

String sql = "SELECT COUNT(\*) AS countIID FROM ipd\_invoice";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("countIID");

String IID = "IPDIID" + (++count);

txt\_iid.setText(IID);

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadPARID() {

try {

String searchSql = "SELECT \* FROM ipd\_p\_a\_registration WHERE status\_id = '1'";

ResultSet rs = db.search(searchSql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

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

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void printInvoice(String iid) {

try {

String path = SystemVariableList.REPORT\_BASE\_URL + "ipd\_invoice\_main.jasper";

InputStream inputStream = new FileInputStream(path);

Map<String, Object> params = new HashMap<>();

params.put("iid", iid);

params.put("subReport", SystemVariableList.REPORT\_BASE\_URL + "ipd\_invoice\_sub.jasper");

JasperPrint fillReport = JasperFillManager.fillReport(inputStream, params, db.getDBConnection());

JasperViewer.viewReport(fillReport, false);

} catch (Exception e) {

e.printStackTrace();

}

}

private void createNewInvoice() {

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

dtm.setRowCount(0);

}

public void loadPaymentType() {

try {

String query = "SELECT `name` FROM `payment\_type`";

ResultSet rs = db.search(query);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

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

}

jComboBox2.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

PARID : combo box

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String parid = jComboBox1.getSelectedItem().toString();

//patient Details

if (!parid.equals("Select")) {

String patientSql = "SELECT \* FROM patient WHERE nic = (SELECT NIC FROM ipd\_p\_a\_registration WHERE PARID = '" + parid + "')";

ResultSet patientRs = db.search(patientSql);

if (patientRs.next()) {

txt\_nic.setText(patientRs.getString("nic"));

txt\_fname.setText(patientRs.getString("fname"));

txt\_lname.setText(patientRs.getString("lname"));

}

String symtomSql = "SELECT symptoms FROM ipd\_doctor WHERE PARID = '" + parid + "'";

ResultSet symtomRs = db.search(symtomSql);

if (symtomRs.next()) {

txt\_symtoms.setText(symtomRs.getString("symptoms"));

}

String dChargeSql = "SELECT charge FROM doctor\_charge WHERE code = (SELECT dcharge FROM ipd\_p\_a\_registration WHERE PARID = '" + parid + "')";

ResultSet dChargRs = db.search(dChargeSql);

if (dChargRs.next()) {

ResultSet addateRs = db.search("SELECT `date` FROM ipd\_p\_a\_registration WHERE `PARID` ='" + parid + "'");

addateRs.next();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

Date dis = new Date();

Date d1 = addateRs.getDate("date");

String a = sdf.format(d1);

LocalDate date1 = LocalDate.parse(a);

LocalDate date3 = LocalDate.parse(sdf.format(dis));

long noofdays = DAYS.between(date1, date3);

String charge = dChargRs.getString("charge");

if (noofdays == 0) {

Double icharge = Double.parseDouble(charge) \* 1;

txt\_dcharge.setText(icharge.toString());

} else {

Double icharge = Double.parseDouble(charge) \* noofdays;

txt\_dcharge.setText(icharge.toString());

}

}

String rChargeSql = "SELECT dayprice FROM room WHERE room\_no = (SELECT room\_type FROM ipd\_p\_a\_registration WHERE PARID = '" + parid + "')";

ResultSet rChargRs = db.search(rChargeSql);

if (rChargRs.next()) {

ResultSet addateRs = db.search("SELECT `date` FROM ipd\_p\_a\_registration WHERE `PARID` ='" + parid + "'");

addateRs.next();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

Date dis = new Date();

Date d1 = addateRs.getDate("date");

String a = sdf.format(d1);

LocalDate date1 = LocalDate.parse(a);

LocalDate date3 = LocalDate.parse(sdf.format(dis));

long noofdays = DAYS.between(date1, date3);

String charge = rChargRs.getString("dayprice");

if (noofdays == 0) {

Double icharge = Double.parseDouble(charge) \* 1;

txt\_Rcharge.setText(icharge.toString());

} else {

Double icharge = Double.parseDouble(charge) \* noofdays;

txt\_Rcharge.setText(icharge.toString());

}

}

String advanceSql = "SELECT Advance FROM ipd\_p\_a\_registration WHERE PARID = '" + parid + "'";

ResultSet advanceRs = db.search(advanceSql);

if (advanceRs.next()) {

txt\_advance.setText(advanceRs.getString("Advance"));

}

}

//tatble implement

if (!parid.equals("Select")) {

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

String sql = "SELECT \* FROM ipd\_treatment WHERE PARID = '" + parid + "' && status\_id = '2'";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

String sqlTreatment = "SELECT name FROM drug WHERE drug\_id = '" + rs.getString("treatment") + "'";

ResultSet rsTreatment = db.search(sqlTreatment);

if (rsTreatment.next()) {

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

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

String sqlCost = "SELECT cost FROM drug WHERE drug\_id = '" + rs.getString("treatment") + "'";

ResultSet rsCost = db.search(sqlCost);

if (rsCost.next()) {

double cost = rsCost.getDouble("cost");

double qty = rs.getDouble("qty");

double totalcost = cost \* qty;

v.add(rsCost.getString("cost"));

v.add(totalcost);

}

}

dtm.addRow(v);

}

}

//payment

if (!parid.equals("Select")) {

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

double subt = 0;

double dcharge = Double.parseDouble(txt\_dcharge.getText());

for (int row = 0; row < dtm.getRowCount(); row++) {

subt += Double.parseDouble(dtm.getValueAt(row, 4).toString());

}

double rcharge = Double.parseDouble(txt\_Rcharge.getText());

double advance = Double.parseDouble(txt\_advance.getText());

double total = subt + dcharge + rcharge - advance;

txt\_subt.setText("" + total);

txt\_dis.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Please Select PARID!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Discount RS : text field

private void txt\_disActionPerformed(java.awt.event.ActionEvent evt) {

if (txt\_subt.getText() != null && !txt\_dis.getText().trim().equals("")) {

double discount = Double.parseDouble(txt\_dis.getText());

double total = Double.parseDouble(txt\_subt.getText());

if (total > 500) {

txt\_net.setText("" + (total - discount));

} else {

JOptionPane.showMessageDialog(this, "This Invoice PARID not eligible for Discount", "warning", JOptionPane.WARNING\_MESSAGE);

txt\_dis.setText("0");

txt\_net.setText("" + (total));

}

jComboBox2.grabFocus();

}

}

private void txt\_disKeyTyped(java.awt.event.KeyEvent evt) {

String price = txt\_dis.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Payment Type : combo cox

private void jComboBox2ItemStateChanged(java.awt.event.ItemEvent evt) {

String text = jComboBox2.getSelectedItem().toString();

if (text.equals("Select")) {

txt\_pay.setEnabled(false);

txt\_pay.setText("");

txt\_bala.setForeground(Color.BLACK);

} else {

txt\_pay.setEnabled(true);

}

}

Payment RS : text field

private void txt\_payActionPerformed(java.awt.event.ActionEvent evt) {

if (txt\_pay.getText() != null && !txt\_pay.getText().trim().equals("")) {

double net = Double.parseDouble(txt\_net.getText());

double pay = Double.parseDouble(txt\_pay.getText());

if (pay >= net) {

txt\_bala.setText("" + (pay - net));

txt\_bala.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Payment Not Enough", "warning", JOptionPane.WARNING\_MESSAGE);

txt\_pay.setText("");

txt\_pay.grabFocus();

}

}

}

private void txt\_payKeyTyped(java.awt.event.KeyEvent evt) {

String price = txt\_pay.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Balance RS : text field

private void txt\_balaActionPerformed(java.awt.event.ActionEvent evt) {

try {

String iid = txt\_iid.getText();

String parid = jComboBox1.getSelectedItem().toString();

String nic = txt\_nic.getText();

String username = jLabel8.getText();

String total = txt\_subt.getText();

String discount = txt\_dis.getText();

String nettotal = txt\_net.getText();

String payment = txt\_pay.getText();

String rcharge = txt\_Rcharge.getText();

String paymentType = jComboBox2.getSelectedItem().toString();

String sqlSearch = "SELECT date FROM ipd\_p\_a\_registration WHERE PARID = '" + parid + "'";

ResultSet rs = db.search(sqlSearch);

if (rs.next()) {

ResultSet rs2 = db.search("SELECT \* FROM `payment\_type` WHERE `name` = '" + paymentType + "'");

rs2.next();

String paymentTypeId = rs2.getString("id");

String invoice\_sql = "INSERT INTO ipd\_invoice VALUES ('" + iid + "','" + parid + "'"

+ ",'" + nic + "','" + username + "',CURRENT\_TIMESTAMP(),'" + rcharge + "','" + total + "'"

+ ",'" + discount + "','" + nettotal + "','" + payment + "','" + paymentTypeId + "')";

db.push(invoice\_sql);

}

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

for (int row = 0; row < dtm.getRowCount(); row++) {

String treatmentCode = dtm.getValueAt(row, 0).toString();

String qty = dtm.getValueAt(row, 2).toString();

String uprice = dtm.getValueAt(row, 3).toString();

String cost = dtm.getValueAt(row, 4).toString();

String invoiceDrugSql = "INSERT INTO ipd\_invoice\_drug (invoice\_id,drug\_id,qty,unitt\_price,total) VALUES ('" + iid + "'"

+ ",'" + treatmentCode + "','" + qty + "','" + uprice + "','" + cost + "')";

db.push(invoiceDrugSql);

String RidSql = "UPDATE ipd\_p\_a\_registration SET status\_id = '0' WHERE PARID = '" + parid + "'";

db.push(RidSql);

}

printInvoice(iid);

createNewInvoice();

loadPARID();

} catch (Exception e) {

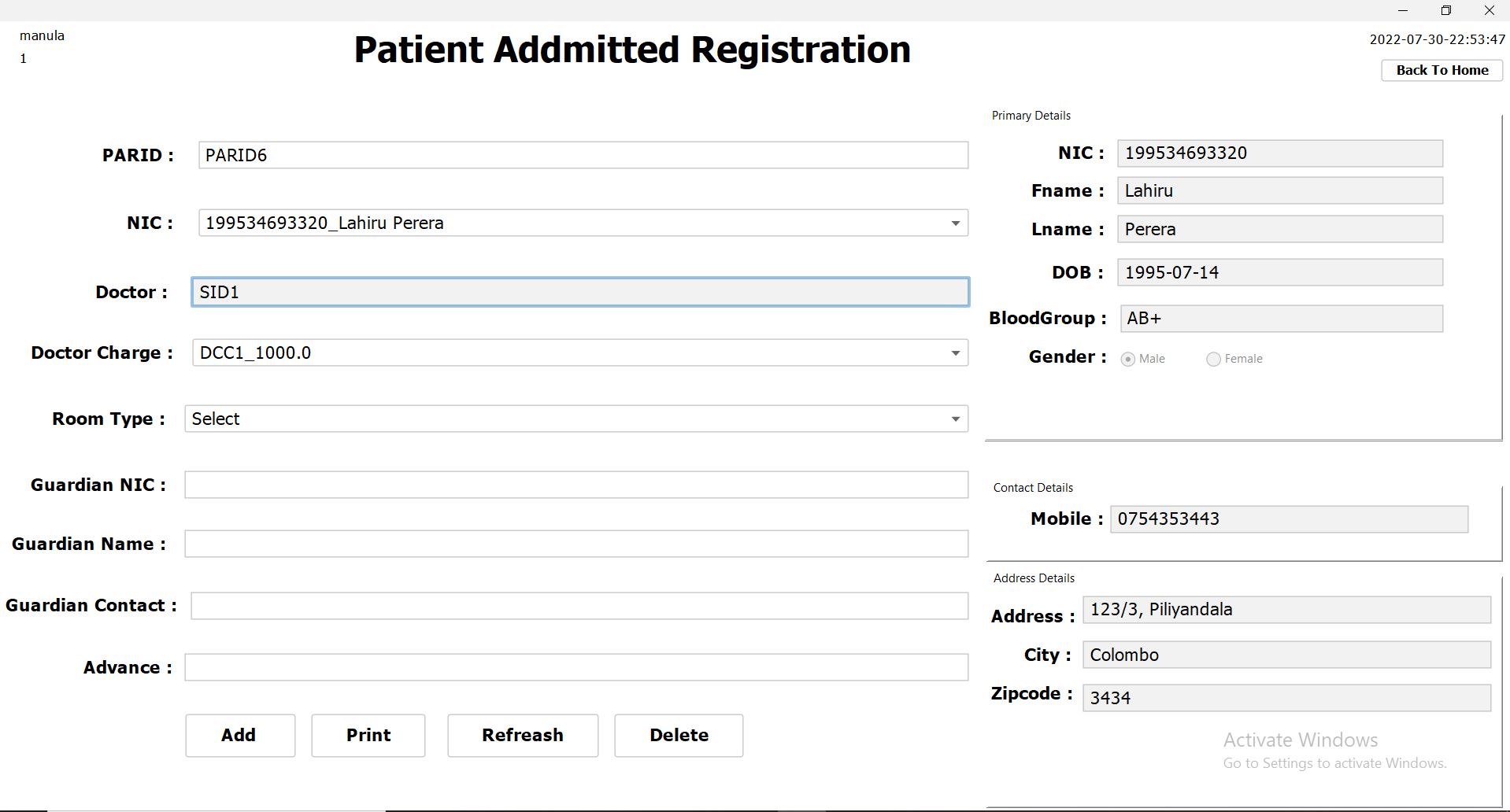
e.printStackTrace();

JOptionPane.showMessageDialog(this, "Already Exit", "warning", JOptionPane.WARNING\_MESSAGE);

}

}

IPD Reception



Constructs and variables

public PatientAddmittedRegistration() {

initComponents();

setLocationRelativeTo(null);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

loadNIC();

loadRoom();

loadDcharge();

genaratePARID();

}

Other methods

private void genaratePARID() {

try {

String sql = "SELECT COUNT(\*) AS countPARID FROM ipd\_p\_a\_registration";

ResultSet rs = db.search(sql);

if (rs.next()){

int count = rs.getInt("countPARID");

String PARID = "PARID" + (++count);

txt\_parid.setText(PARID);

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadNIC() {

try {

String sql = "SELECT \* FROM patient WHERE nic IN (SELECT patient\_nic FROM ipd\_doc\_patient WHERE status\_id = '1') AND status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String nic = rs.getString("nic") + "\_" + rs.getString("fname") + " " + rs.getString("lname");

v.add(nic);

}

combo\_nic.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

}

}

private void loadRoom() {

try {

String sql = "SELECT \* FROM room WHERE status\_id = '1' AND qty != '0'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("room\_no") + "\_"+ " room\_type : " + rs.getString("room\_type") + " dayprice : " + rs.getString("dayprice") + " qty : " + rs.getString("qty");

v.add(data);

}

combo\_room.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadDcharge() {

try {

String sql = "SELECT \* FROM doctor\_charge WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

while (rs.next()) {

String data = rs.getString("code") + "\_" + rs.getString("charge");

v.add(data);

}

combo\_dcharge.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

NIC : combo box

private void combo\_nicActionPerformed(java.awt.event.ActionEvent evt) {

try {

if (!combo\_nic.getSelectedItem().toString().equals("Select")) {

String nic = combo\_nic.getSelectedItem().toString().split("\_")[0];

String sql = "SELECT \* FROM ipd\_doc\_patient WHERE patient\_nic = '"+nic+"' AND status\_id = '1'";

ResultSet rs = db.search(sql);

if (rs.next()){

String sql2 = "SELECT sid FROM stuff WHERE status\_id = '1' AND sid = (SELECT sid FROM system\_user WHERE username = '"+rs.getString("doc")+"')";

ResultSet rs2 = db.search(sql2);

if (rs2.next()){

txt\_doc.setText(rs2.getString("sid"));

}

}

String sql2 = "SELECT \* FROM patient WHERE nic = '"+nic+"'";

ResultSet rs2 = db.search(sql2);

if (rs2.next()) {

txt\_nic.setText(rs2.getString("nic"));

txt\_fname.setText(rs2.getString("fname"));

txt\_lname.setText(rs2.getString("lname"));

txt\_dob.setText(rs2.getString("dob"));

String searchSql = "SELECT name FROM blood\_type WHERE id = '"+rs2.getString("blood-type\_id")+"'";

ResultSet rsblood = db.search(searchSql);

if (rsblood.next())

txt\_bg.setText(rsblood.getString("name"));

txt\_mobile.setText(rs2.getString("mobile"));

ResultSet addressRs = db.search("SELECT \* FROM address WHERE `id`='"+rs2.getString("address\_id")+"' ");

addressRs.next();

txt\_address.setText(addressRs.getString("addresss\_line1")+ ", " + addressRs.getString("address\_line2"));

ResultSet cityRs = db.search("SELECT \* FROM city WHERE id='"+addressRs.getString("city\_id")+"'");

cityRs.next();

txt\_city.setText(cityRs.getString("name"));

txt\_zipcode.setText(addressRs.getString("postal\_code"));

String gender = rs2.getString("gender\_id");

if (gender.equals("1")) {

rdo\_M.setSelected(true);

} else {

rdo\_F.setSelected(true);

}

}

} else {

JOptionPane.showMessageDialog(this, "please Select NIC!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Advance : text field

private void txt\_advanceKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String price = txt\_advance.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Add button

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

try {

String PARID = txt\_parid.getText();

String nic = combo\_nic.getSelectedItem().toString().split("\_")[0];

String doc\_username = txt\_doc.getText();

String room = combo\_room.getSelectedItem().toString().split("\_")[0];

String qty = combo\_room.getSelectedItem().toString().split("qty : ")[1];

String gnic = txt\_gnic.getText();

String gname =txt\_name.getText();

String gcontact = txt\_contact.getText();

String advance = txt\_advance.getText();

String username = jLabel8.getText();

String dcharge = combo\_dcharge.getSelectedItem().toString().split("\_")[0];

String searchSql = "SELECT PARID FROM ipd\_p\_a\_registration WHERE PARID = '"+PARID+"'";

ResultSet rs = db.search(searchSql);

if (!PARID.equals("") && !nic.equals("Select") && !doc\_username.equals("")

&& !gnic.equals("") && !gname.equals("") && !gcontact.equals("")

&& !advance.equals("") && !room.equals("Select")){

if (!rs.next()) {

ResultSet guardianRs= db.search("SELECT \* FROM guardian WHERE nic='"+gnic+"' AND name='"+gname+"' AND contact\_no='"+gcontact+"'");

if (!guardianRs.next()) {

db.push("INSERT INTO guardian (nic,name,contact\_no) VALUES('"+gnic+"','"+gname+"','"+gcontact+"')");

}

ResultSet guardianIdRs= db.search("SELECT \* FROM guardian WHERE nic='"+gnic+"' AND name='"+gname+"' AND contact\_no='"+gcontact+"'");

guardianIdRs.next();

String gid = guardianIdRs.getString("id");

db.push("update room set qty = qty-1 where room\_no = '"+room+"'");

String sql = "INSERT INTO ipd\_p\_a\_registration VALUES ('"+PARID+"',"

+ "'"+nic+"','"+doc\_username+"','"+dcharge+"','"+room+"','"+advance+"','"+username+"',"

+ " CURRENT\_TIMESTAMP(), '3','"+gid+"')";

db.push(sql);

String sql1 = "UPDATE ipd\_doc\_patient SET status\_id='2' WHERE patient\_nic='"+nic+"'";

db.push(sql1);

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "PARID already Defined","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "please fill All the fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Print button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String PARID = txt\_parid.getText();

String fname = txt\_fname.getText();

String lname = txt\_lname.getText();

String dob = txt\_dob.getText();

String mobile = txt\_mobile.getText();

String address = txt\_address.getText();

String city = txt\_city.getText();

String zipcode = txt\_zipcode.getText();

String sql = "SELECT PARID FROM ipd\_p\_a\_registration WHERE PARID = '"+PARID+"'";

ResultSet rs = db.search(sql);

if (rs.next()) {

String path = SystemVariableList.REPORT\_BASE\_URL+"ipd\_patient\_admintted\_registration\_Reciept.jasper";

InputStream inputStream = new FileInputStream(path);

Map<String, Object> params = new HashMap<>();

params.put("PARID", PARID);

params.put("fname", fname);

params.put("lname", lname);

params.put("dob", dob);

params.put("mobile", mobile);

params.put("address", address);

params.put("city", city);

params.put("zipcode", zipcode);

JasperPrint fillReport = JasperFillManager.fillReport(inputStream, params, db.getDBConnection());

JasperViewer.viewReport(fillReport,false);

genaratePARID();

combo\_nic.setSelectedItem("Select");

combo\_room.setSelectedItem("Select");

combo\_dcharge.setSelectedIndex(0);

txt\_gnic.setText("");

txt\_name.setText("");

txt\_contact.setText("");

txt\_advance.setText("");

txt\_nic.setText("");

txt\_fname.setText("");

txt\_lname.setText("");

txt\_dob.setText("");

txt\_bg.setText("");

rdo\_M.setSelected(true);

txt\_mobile.setText("");

txt\_address.setText("");

txt\_zipcode.setText("");

} else {

JOptionPane.showMessageDialog(this, "PARID Does Not Exist!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Refreash button

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

PatientAddmittedRegistration p = new PatientAddmittedRegistration();

p.setVisible(true);

dispose();

}

Delete button

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

try {

String nic = combo\_nic.getSelectedItem().toString().split("\_")[0];

String searchSql = "SELECT \* FROM ipd\_doc\_patient WHERE patient\_nic = '"+nic+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

String sql = "UPDATE ipd\_doc\_patient SET status\_id='2' WHERE patient\_nic='"+nic+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Not Defind","warning",JOptionPane.WARNING\_MESSAGE);

}

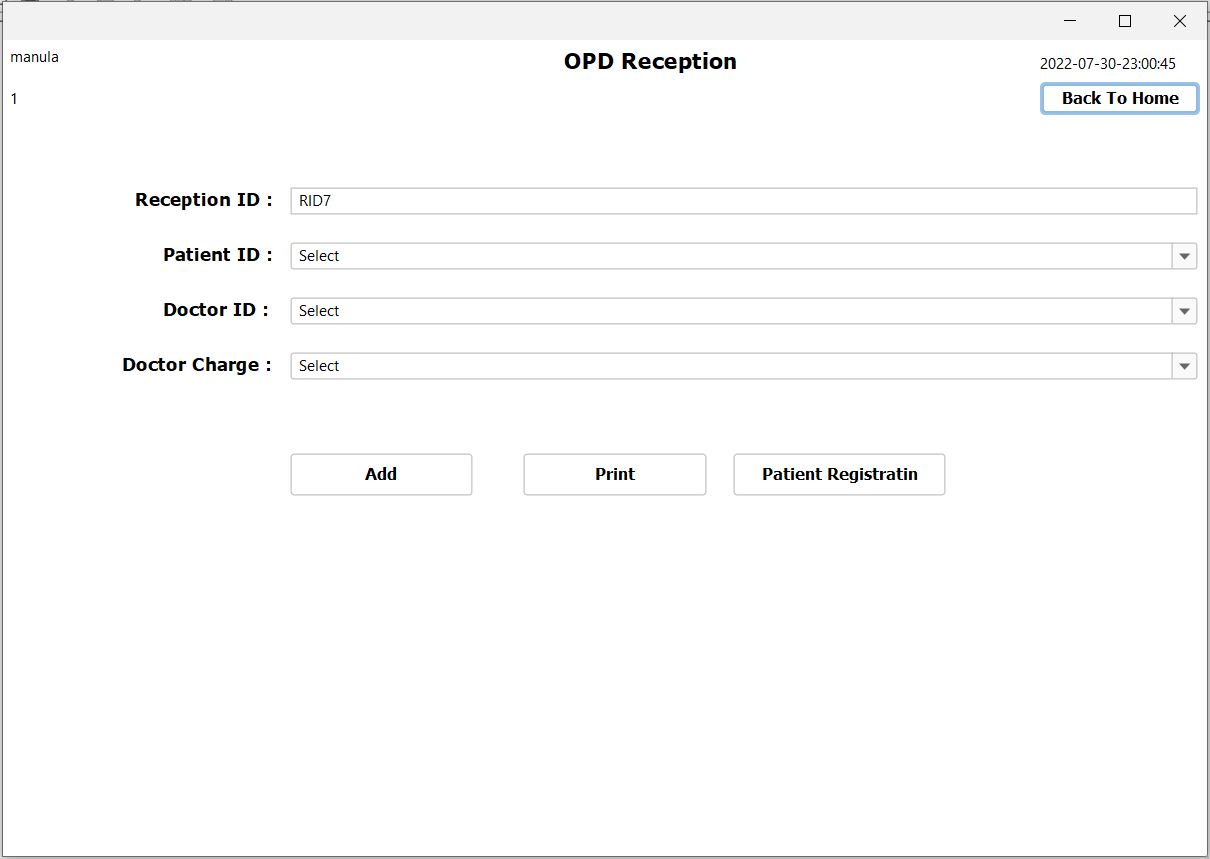
} catch (Exception e) {

e.printStackTrace();

}

}

OPD Reception



Constructs and variables

public OPDReceiption() {

initComponents();

setLocationRelativeTo(null);

AutoCompleteDecorator.decorate(combo\_pid);

AutoCompleteDecorator.decorate(combo\_did);

AutoCompleteDecorator.decorate(combo\_dCharge1);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

genarateRID();

loadPID();

loadDID();

loadDCharge();

labal\_un.setText(SystemConfigData.getActiveUser());

labal\_ut.setText(SystemConfigData.getUserType());

}

Other methods

private void genarateRID() {

try {

String sql = "SELECT COUNT(\*) AS countRID FROM opd\_reception";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("countRID");

String RID = "RID" + (++count);

txt\_rid.setText(RID);

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadPID() {

try {

String sql = "SELECT \* FROM patient WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()){

String data = rs.getString("nic") + "\_" + rs.getString("fname")+ " " + rs.getString("lname");

v.add(data);

}

combo\_pid.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadDID() {

try {

String sql = "SELECT \* FROM stuff WHERE status\_id = '1' AND sid IN (SELECT sid FROM system\_user WHERE staff\_type\_id = '2')";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("sid") + "\_" + "DR. " + rs.getString("fname") + " " + rs.getString("lname");

v.add(data);

}

combo\_did.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadDCharge() {

try {

String sql = "SELECT \* FROM doctor\_charge WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("code") + "\_" + rs.getString("charge");

v.add(data);

}

combo\_dCharge1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

Add button

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

try {

String rid = txt\_rid.getText();

String pid = combo\_pid.getSelectedItem().toString().split("\_")[0];

String did = combo\_did.getSelectedItem().toString().split("\_")[0];

String dcharge = combo\_dCharge1.getSelectedItem().toString().split("\_")[0];

String uname =labal\_un.getText();

String date = jLabel14.getText();

String searchSql = "SELECT RID FROM opd\_reception WHERE RID = '"+rid+"'";

ResultSet rs = db.search(searchSql);

if (!rid.equals("") && !pid.equals("Select") && !did.equals("Select") && !dcharge.equals("Select") && !uname.equals("")){

if (!rs.next()) {

String sql = "INSERT INTO opd\_reception VALUES ('"+rid+"','"+pid+"','"+did+"','"+dcharge+"','"+date+"','"+uname+"','3')";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "RID allready Defined","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "please fill All the fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Print button

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

try {

String rid = txt\_rid.getText();

String sql = "SELECT RID FROM opd\_reception WHERE RID = '"+rid+"'";

ResultSet rs = db.search(sql);

if (rs.next()) {

String path = SystemVariableList.REPORT\_BASE\_URL+"printRecept.jasper";

// InputStream inputStream = new FileInputStream(path);

HashMap params = new HashMap();

params.put("rid", rid);

JasperPrint fillReport = JasperFillManager.fillReport(path, params, db.getDBConnection());

JasperViewer.viewReport(fillReport,false);

genarateRID();

} else {

JOptionPane.showMessageDialog(this, "RID Does Not Exist!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Patient Registratin button

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

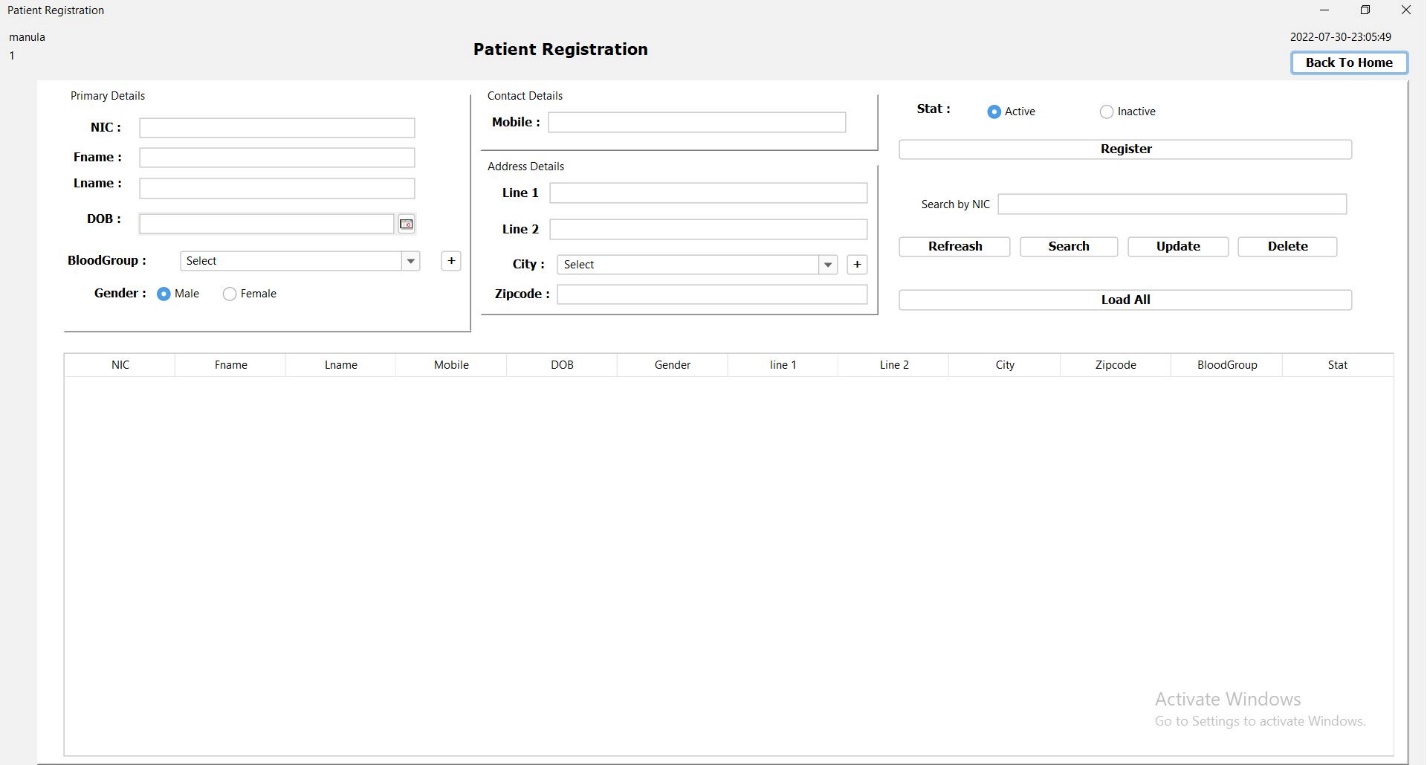
PatientRegistration patient = new PatientRegistration();

patient.setVisible(true);

dispose();

}

Patient Registration



Constructs and variables

public PatientRegistration() {

initComponents();

setLocationRelativeTo(null);

loadBoodGroup();

AutoCompleteDecorator.decorate(combo\_bgroup);

AutoCompleteDecorator.decorate(jComboBox1);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

loadCity();

}

Other methods

public void loadBoodGroup() {

try {

String sql = "SELECT \* FROM blood\_type WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("id") + "\_" + rs.getString("name");

v.add(data);

}

combo\_bgroup.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

public void loadCity() {

try {

String sql = "SELECT \* FROM city";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("id") + "\_" + rs.getString("name");

v.add(data);

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void clearField() {

txt\_fname.setText(null);

txt\_lname.setText(null);

rdo\_m.setSelected(true);

txt\_mobile.setText(null);

txt\_address.setText(null);

txt\_address1.setText(null);

txt\_zipcode.setText(null);

combo\_bgroup.setSelectedIndex(0);

jComboBox1.setSelectedIndex(0);

rdo\_a.setSelected(true);

txt\_dob.setDate(null);

}

}

private void txt\_nicActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_fname.grabFocus();

}

private void txt\_fnameActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_lname.grabFocus();

}

private void txt\_lnameActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_dob.grabFocus();

}

private void combo\_bgroupActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

rdo\_m.grabFocus();

}

private void rdo\_fActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_mobile.grabFocus();

}

private void rdo\_mActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_mobile.grabFocus();

}

private void txt\_mobileActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_address1.grabFocus();

}

private void txt\_address1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_address.grabFocus();

}

private void txt\_addressActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

jComboBox1.grabFocus();

}

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_zipcode.grabFocus();

}

private void txt\_zipcodeActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

rdo\_a.grabFocus();

}

DOB : button

private void txt\_dobPropertyChange(java.beans.PropertyChangeEvent evt) {

// TODO add your handling code here:

try {

Date date = txt\_dob.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd-HH:mm:ss");

String numberdate = sdf.format(date);

String s = jLabel14.getText();

if (s.compareTo(numberdate) > 0) {

// System.out.println(s.compareTo(numberdate) > 0);

} else {

JOptionPane.showMessageDialog(this, "Please choose correct date", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (NullPointerException n) {

} catch (Exception e) {

e.printStackTrace();

}

}

BloodGroup : + button

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

// TODO add your handling code here:

bloodRegistration b = new bloodRegistration(this);

b.setVisible(true);

}

Mobile : text field

private void txt\_mobileKeyTyped(java.awt.event.KeyEvent evt) {

if (!Character.isDigit(evt.getKeyChar())) {

evt.consume();

}

String mobile = txt\_mobile.getText();

String text = mobile + evt.getKeyChar();

if (text.length() == 1) {

if (!text.equals("0")) {

evt.consume();

}

} else if (text.length() == 2) {

if (!text.equals("07")) {

evt.consume();

}

} else if (text.length() == 3) {

if (!Pattern.compile("07[1245678]").matcher((text)).matches()) {

evt.consume();

}

} else if (text.length() <= 10) {

if (!Pattern.compile("07[1245678][0-9]+").matcher((text)).matches()) {

evt.consume();

}

} else {

evt.consume();

}

}

City : + button

private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

CityRegistration c = new CityRegistration(this, true, 3);

c.setVisible(true);

}

Register button

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

try {

String nic = txt\_nic.getText();

String fname = txt\_fname.getText();

String lname = txt\_lname.getText();

String gender = (rdo\_m.isSelected()) ? "1" : "2";

String mobile = txt\_mobile.getText();

String address = txt\_address.getText();

String address1 = txt\_address1.getText();

String city = jComboBox1.getSelectedItem().toString().split("\_")[0];

String zipcode = txt\_zipcode.getText();

int stat = (rdo\_a.isSelected()) ? 1 : 2;

String bloodGroup = combo\_bgroup.getSelectedItem().toString().split("\_")[0];

Date date = txt\_dob.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

if (!nic.equals("") && !fname.equals("") && !lname.equals("")

&& !numberdate.equals("") && !address.equals("")

&& !mobile.equals("") && !city.equals("") && !zipcode.equals("")

&& !bloodGroup.equals("Select") && !city.equals("Select") && !address1.equals("")) {

String searchSql = "SELECT \* FROM patient WHERE nic = '" + nic + "'";

ResultSet rs = db.search(searchSql);

if (!rs.next()) {

ResultSet addressRs = db.search("SELECT \* FROM address WHERE addresss\_line1 ='" + address1 + "' AND address\_line2 ='" + address + "' AND postal\_code='" + zipcode + "' AND city\_id ='" + city + "'");

if (!addressRs.next()) {

db.push("INSERT INTO address (addresss\_line1,address\_line2,postal\_code,city\_id) VALUES ('" + address1 + "','" + address + "','" + zipcode + "','" + city + "')");

}

ResultSet addressIdRs = db.search("SELECT \* FROM address WHERE addresss\_line1 ='" + address1 + "' AND address\_line2 ='" + address + "' AND postal\_code='" + zipcode + "' AND city\_id ='" + city + "'");

addressIdRs.next();

String addrressId = addressIdRs.getString("id");

String sql = "INSERT INTO patient VALUES ('" + nic + "','" + mobile + "','" + fname + "',"

+ "'" + lname + "','" + numberdate + "','" + addrressId + "'"

+ ",'" + gender + "','" + bloodGroup + "','" + stat + "')";

db.push(sql);

clearField();

JOptionPane.showMessageDialog(this, "Success!", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "NIC already defined!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All the fields", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (NullPointerException n) {

JOptionPane.showMessageDialog(this, "Please Fill DOB", "warning", JOptionPane.WARNING\_MESSAGE);

clearField();

} catch (Exception e) {

e.printStackTrace();

}

}

Refreash button

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

PatientRegistration pr = new PatientRegistration();

pr.setVisible(true);

dispose();

}

Search button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String sbNIC = txt\_search\_by\_nic.getText();

String sql = "SELECT \* FROM patient WHERE nic ='" + sbNIC + "'";

ResultSet rs = db.search(sql);

if (!sbNIC.equals("")) {

if (rs.next()) {

txt\_nic.setText(rs.getString("nic"));

txt\_fname.setText(rs.getString("fname"));

txt\_lname.setText(rs.getString("lname"));

txt\_mobile.setText(rs.getString("mobile"));

Date dob = rs.getDate("dob");

txt\_dob.setDate(dob);

String gender = rs.getString("gender\_id");

if (gender.equals("1")) {

rdo\_m.setSelected(true);

} else {

rdo\_f.setSelected(true);

}

ResultSet addressRs = db.search("SELECT \* FROM address WHERE `id`='" + rs.getString("address\_id") + "' ");

addressRs.next();

txt\_address.setText(addressRs.getString("address\_line2"));

txt\_address1.setText(addressRs.getString("addresss\_line1"));

ResultSet cityRs = db.search("SELECT \* FROM city WHERE id='" + addressRs.getString("city\_id") + "'");

cityRs.next();

jComboBox1.setSelectedItem(cityRs.getString("id") + "\_" + cityRs.getString("name"));

txt\_zipcode.setText(addressRs.getString("postal\_code"));

String bloodG = rs.getString("blood-type\_id");

String searchBGSql = "SELECT \* FROM blood\_type WHERE id = '" + bloodG + "' && status\_id = '1'";

ResultSet BGRs = db.search(searchBGSql);

if (BGRs.next()) {

String data = BGRs.getString("id") + "\_" + BGRs.getString("name");

combo\_bgroup.setSelectedItem(data);

}

String status = rs.getString("status\_id");

if (status.equals("1")) {

rdo\_a.setSelected(true);

} else {

rdo\_I.setSelected(true);

}

} else {

JOptionPane.showMessageDialog(this, "Wrong NIC. Please Try Again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Search By NIC Field!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Update button

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String nic = txt\_nic.getText();

String fname = txt\_fname.getText();

String lname = txt\_lname.getText();

String sbNIC = txt\_search\_by\_nic.getText();

String gender = (rdo\_m.isSelected()) ? "1" : "2";

String mobile = txt\_mobile.getText();

String address = txt\_address.getText();

String address1 = txt\_address1.getText();

String city = jComboBox1.getSelectedItem().toString().split("\_")[0];

String zipcode = txt\_zipcode.getText();

String bloodGroup = combo\_bgroup.getSelectedItem().toString().split("\_")[0];

int stat = (rdo\_a.isSelected()) ? 1 : 2;

Date date = txt\_dob.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

if (!sbNIC.equals("") && !nic.equals("") && !fname.equals("") && !lname.equals("")

&& !numberdate.equals("") && !address.equals("")

&& !mobile.equals("") && !city.equals("") && !zipcode.equals("")

&& !bloodGroup.equals("Select") && !city.equals("Select") && !address1.equals("")) {

String searchSql = "SELECT \* FROM patient WHERE nic = '" + sbNIC + "'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

ResultSet addressRs = db.search("SELECT \* FROM address WHERE id IN (SELECT address\_id FROM patient WHERE nic = '" + sbNIC + "')");

if (addressRs.next()) {

db.push("UPDATE `address` SET addresss\_line1='" + address1 + "', `address\_line2`='" + address + "', `postal\_code`='" + zipcode + "', `city\_id`='" + city + "' WHERE id = '" + rs.getString("address\_id") + "' ");

}

String sql = "UPDATE patient SET fname='" + fname + "',"

+ "lname='" + lname + "',mobile='" + mobile + "',dob='" + numberdate + "',"

+ "gender\_id='" + gender + "',"

+ "`blood-type\_id`='" + bloodGroup + "',status\_id='" + stat + "' WHERE nic = '" + sbNIC + "'";

db.push(sql);

clearField();

JOptionPane.showMessageDialog(this, "Success!", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "NIC not defined!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All the fields", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (NullPointerException n) {

JOptionPane.showMessageDialog(this, "Please Fill DOB", "warning", JOptionPane.WARNING\_MESSAGE);

clearField();

} catch (Exception e) {

e.printStackTrace();

}

}

Delete button

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String sbSID = txt\_search\_by\_nic.getText();

String searchSql = "SELECT \* FROM patient WHERE nic = '" + sbSID + "'";

ResultSet rs = db.search(searchSql);

if (!sbSID.equals("")) {

if (rs.next()) {

String sql = "UPDATE patient SET status\_id = '2' WHERE nic = '" + sbSID + "'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Wrong NIC. Please Try Again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Search By NIC Field!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

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

String sql = "SELECT \* FROM patient";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

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

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

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

ResultSet genderRs = db.search("SELECT \* FROM gender WHERE id = '" + rs.getString("gender\_id") + "'");

genderRs.next();

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

ResultSet addressRs = db.search("SELECT \* FROM address WHERE id IN (SELECT address\_id FROM patient WHERE nic = '" + rs.getString("nic") + "')");

addressRs.next();

v.add(addressRs.getString("addresss\_line1"));

v.add(addressRs.getString("address\_line2"));

ResultSet cityRs = db.search("SELECT \* FROM city WHERE id='" + addressRs.getString("city\_id") + "'");

cityRs.next();

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

v.add(addressRs.getString("postal\_code"));

String bg = rs.getString("blood-type\_id");

String bgSql = "SELECT \* FROM blood\_type WHERE id = '" + bg + "'";

ResultSet BgRs = db.search(bgSql);

if (BgRs.next()) {

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

}

String stat = rs.getString("status\_id");

if (stat.equals("1")) {

v.add("Active");

} else {

v.add("Inactive");

}

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String nic = dtm.getValueAt(selectRow, 0).toString();

String fname = dtm.getValueAt(selectRow, 1).toString();

String lname = dtm.getValueAt(selectRow, 2).toString();

String mobile = dtm.getValueAt(selectRow, 3).toString();

String dbo = dtm.getValueAt(selectRow, 4).toString();

String gender = dtm.getValueAt(selectRow, 5).toString();

String address1 = dtm.getValueAt(selectRow, 6).toString();

String address = dtm.getValueAt(selectRow, 7).toString();

String city = dtm.getValueAt(selectRow, 8).toString();

String zipcode = dtm.getValueAt(selectRow, 9).toString();

String bloodGroup = dtm.getValueAt(selectRow, 10).toString();

String stat = dtm.getValueAt(selectRow, 11).toString();

txt\_nic.setText(nic);

txt\_fname.setText(fname);

txt\_lname.setText(lname);

txt\_mobile.setText(mobile);

try {

Date date = new SimpleDateFormat("yyyy-MM-dd").parse(dbo);

txt\_dob.setDate(date);

} catch (ParseException ex) {

ex.printStackTrace();

}

if (gender.equals("Male")) {

rdo\_m.setSelected(true);

} else {

rdo\_f.setSelected(true);

}

txt\_address.setText(address);

txt\_address1.setText(address1);

try {

ResultSet cityRs = db.search("SELECT \* FROM city WHERE name = '" + city + "'");

cityRs.next();

jComboBox1.setSelectedItem(cityRs.getString("id") + "\_" + cityRs.getString("name"));

} catch (Exception e) {

e.printStackTrace();

}

txt\_zipcode.setText(zipcode);

try {

String sql = "SELECT \* FROM blood\_type WHERE name = '" + bloodGroup + "'";

ResultSet rs = db.search(sql);

if (rs.next()) {

String data = rs.getString("id") + "\_" + rs.getString("name");

combo\_bgroup.setSelectedItem(data);

}

} catch (Exception e) {

e.printStackTrace();

}

if (stat.equals("Active")) {

rdo\_a.setSelected(true);

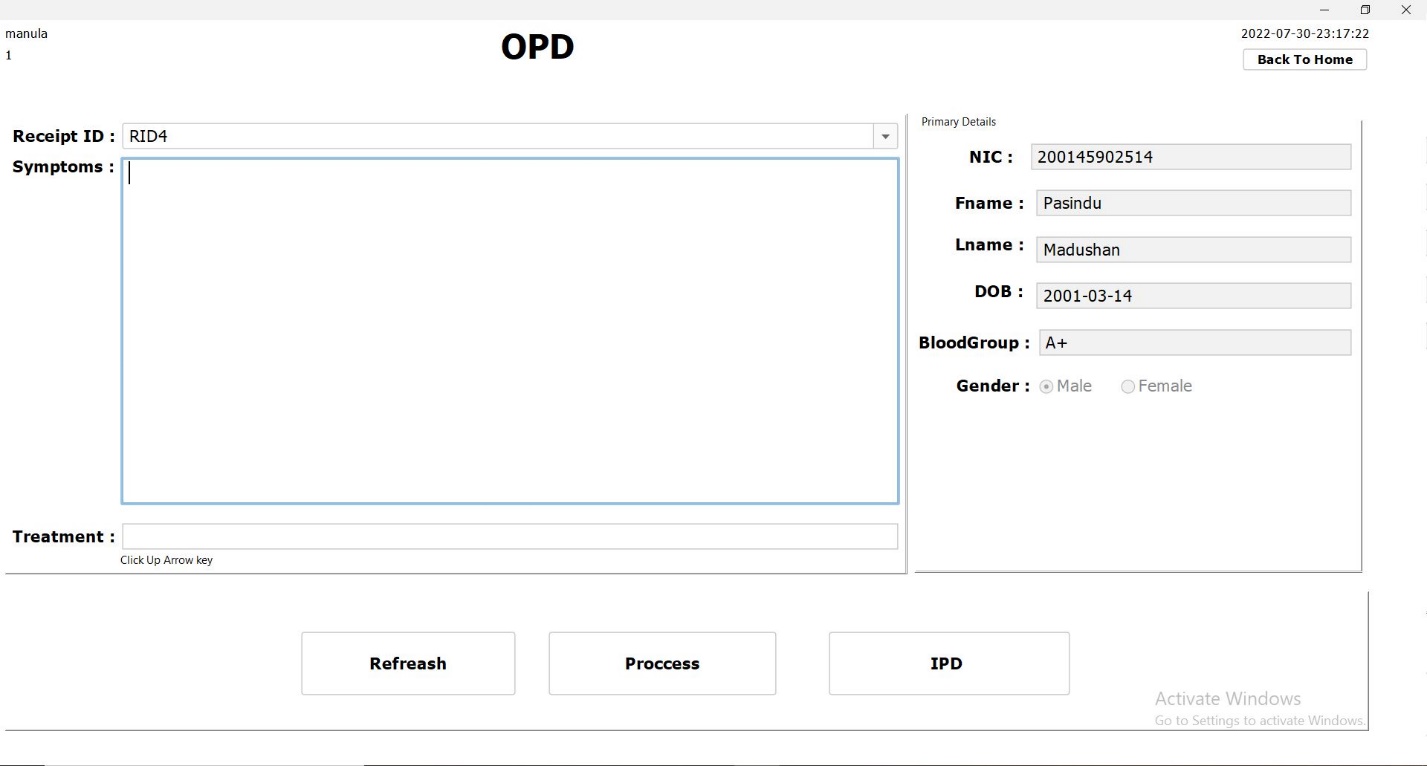
} else {

rdo\_I.setSelected(true);

}

}

OPD Doctor



Constructs and variables

public OPD\_Doctor() {

initComponents();

setLocationRelativeTo(null);

AutoCompleteDecorator.decorate(combo\_rid);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

loadRID();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methods

private void loadRID() {

try {

String sql = "SELECT \* FROM opd\_reception WHERE stat = '3'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("RID");

v.add(data);

}

combo\_rid.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void clearFields() {

txt\_bg.setText(null);

txt\_dob.setText(null);

txt\_fname.setText(null);

txt\_lname.setText(null);

txt\_nic.setText(null);

txt\_symptoms.setText(null);

txt\_treatment.setText(null);

combo\_rid.setSelectedIndex(0);

rdo\_M.setSelected(true);

}

Receipt ID : combo box

private void combo\_ridActionPerformed(java.awt.event.ActionEvent evt) {

try {

String rid = combo\_rid.getSelectedItem().toString();

String sql = "SELECT \* FROM patient WHERE nic = (SELECT patient\_id FROM opd\_reception WHERE RID = '"+rid+"')";

ResultSet rs = db.search(sql);

if (rs.next()) {

txt\_nic.setText(rs.getString("nic"));

txt\_fname.setText(rs.getString("fname"));

txt\_lname.setText(rs.getString("lname"));

txt\_dob.setText(rs.getString("dob"));

String searchSql = "SELECT name FROM blood\_type WHERE id = '"+rs.getString("blood-type\_id")+"'";

ResultSet rsblood = db.search(searchSql);

if (rsblood.next())

txt\_bg.setText(rsblood.getString("name"));

String gender = rs.getString("gender\_id");

if (gender.equals("1")) {

rdo\_M.setSelected(true);

} else {

rdo\_F.setSelected(true);

}

}else {

clearFields();

}

} catch (Exception e) {

e.printStackTrace();

}

}

Treatment : text field

private void txt\_treatmentKeyReleased(java.awt.event.KeyEvent evt) {

if(evt.getKeyCode() == KeyEvent.VK\_UP) {

if (!combo\_rid.getSelectedItem().toString().equals("Select")){

treatmentQtyDto.setRid(combo\_rid.getSelectedItem().toString());

treatment t = new treatment();

t.setVisible(true);

txt\_treatment.setText(treatmentQtyDto.getRid());

} else {

JOptionPane.showMessageDialog(this, "Please Select RID","warning",JOptionPane.WARNING\_MESSAGE);

combo\_rid.grabFocus();

}

}

}

Refreash button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

OPD\_Doctor od = new OPD\_Doctor();

od.setVisible(true);

dispose();

}

Process button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String rid = combo\_rid.getSelectedItem().toString();

String symptoms = txt\_symptoms.getText();

String treatment = txt\_treatment.getText();

String searchSql = "SELECT \* FROM opd\_doctor WHERE Reciept\_id = '"+rid+"'";

ResultSet rs = db.search(searchSql);

if (!rid.equals("Select") && !symptoms.trim().equals("") &&

symptoms != null && treatment != null && !treatment.trim().equals("")) {

if (!rs.next()) {

String DOCsql = "INSERT INTO opd\_doctor VALUES ('"+rid+"','"+symptoms+"','"+treatment+"','"+jLabel8.getText()+"','"+jLabel14.getText()+"')";

db.push(DOCsql);

String ridStatUpdateSql = "UPDATE opd\_reception SET stat = '2' WHERE RID = '"+rid+"'";

db.push(ridStatUpdateSql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

loadRID();

clearFields();

} else {

JOptionPane.showMessageDialog(this, "Please Select RID!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All The Fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

IPD button

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

try {

String rid = combo\_rid.getSelectedItem().toString();

String sql = "SELECT patient\_id FROM opd\_reception WHERE RID = '"+rid+"'";

ResultSet rs = db.search(sql);

if (rs.next()) {

if (!rid.equals("Select")){

treatmentQtyDto.setNic(rs.getString("patient\_id"));

IpdDocPatient idp = new IpdDocPatient();

idp.setVisible(true);

} else {

JOptionPane.showMessageDialog(this, "Please Select RID","warning",JOptionPane.WARNING\_MESSAGE);

combo\_rid.grabFocus();

}

} else {

JOptionPane.showMessageDialog(this, "Please Select RID","warning",JOptionPane.WARNING\_MESSAGE);

}

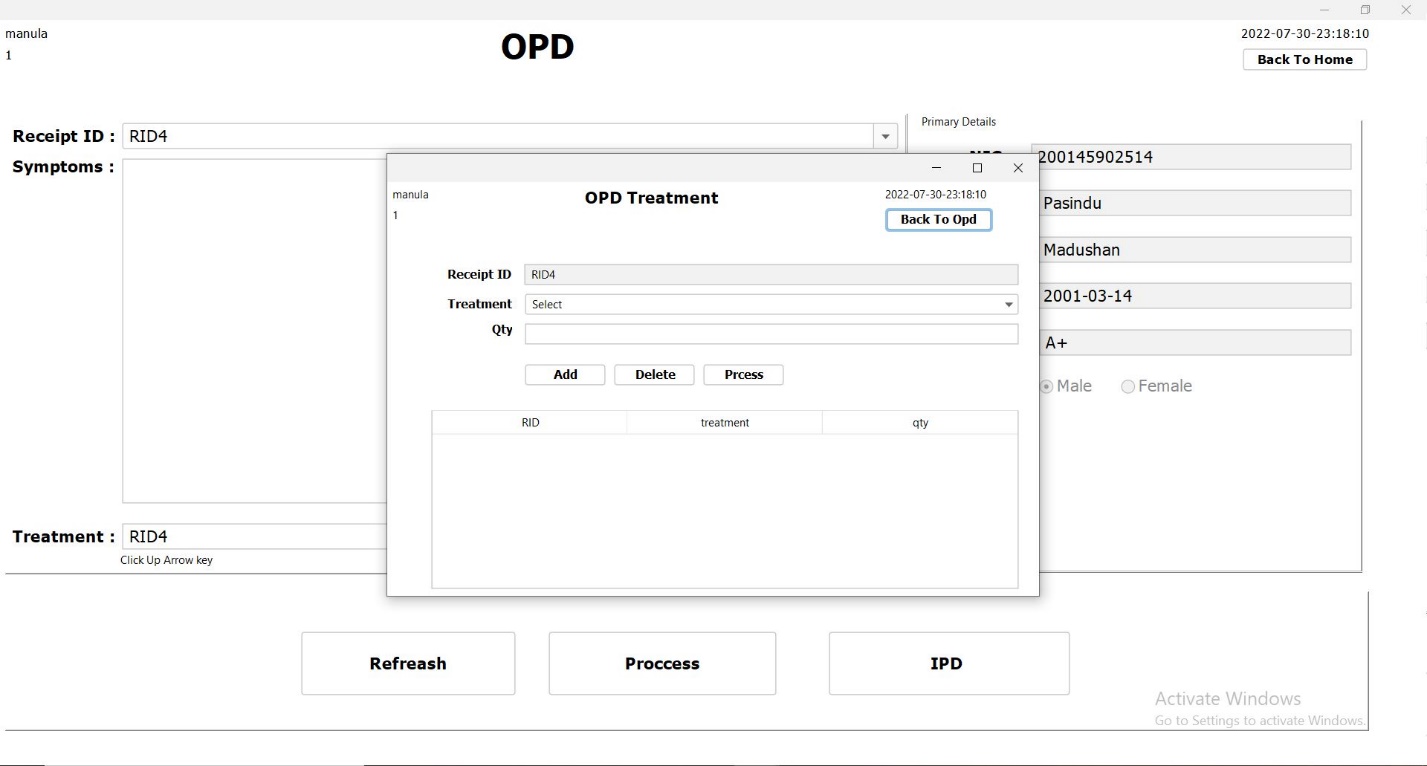
} catch (Exception e) {

e.printStackTrace();

}

}

OPD Treatment



Constructs and variables

public treatment() {

initComponents();

setLocationRelativeTo(null);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

loadTreatment();

txt\_rid.setText(treatmentQtyDto.getRid());

labal\_un.setText(SystemConfigData.getActiveUser());

labal\_ut.setText(SystemConfigData.getUserType());

}

Other methods

private void loadTreatment() {

try {

String sql = "SELECT \* FROM drug WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("drug\_id") + "\_" + rs.getString("name") + " QTY " + rs.getString("qty") + " ";

v.add(data);

}

combo\_treatment.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private treatmentQtyDto checkRowRepeating(String text) {

treatmentQtyDto tqd = null;

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

for (int row = 0; row < jTable1.getRowCount(); row++){

if (text.trim().equals(jTable1.getValueAt(row, 0).toString())){

tqd = new treatmentQtyDto();

double qty = Double.parseDouble(jTable1.getValueAt(row , 2).toString());

tqd.setQty(qty);

dtm.removeRow(row);

break;

}

}

return tqd;

}

Qty text field

private void txt\_qtyActionPerformed(java.awt.event.ActionEvent evt) {

if (!txt\_qty.getText().trim().equals("") && txt\_qty.getText()!= null && !combo\_treatment.getSelectedItem().toString().equals("Select")) {

double qty = Double.parseDouble(txt\_qty.getText());

double ablQty = Double.parseDouble(combo\_treatment.getSelectedItem().toString().split(" ")[2]);

if (qty<ablQty) {

btn\_add.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Qty Not Available","warning",JOptionPane.WARNING\_MESSAGE);

txt\_qty.setText(null);

txt\_qty.grabFocus();

}

} else {

JOptionPane.showMessageDialog(this, "please fill all the Field","warning",JOptionPane.WARNING\_MESSAGE);

combo\_treatment.grabFocus();

}

}

private void txt\_qtyKeyTyped(java.awt.event.KeyEvent evt) {

if (!Character.isDigit(evt.getKeyChar())) {

evt.consume();

}

}

Add button

private void btn\_addActionPerformed(java.awt.event.ActionEvent evt) {

if (txt\_qty.getText() != null && !txt\_qty.getText().trim().equals("") && !combo\_treatment.getSelectedItem().equals("Select")) {

double qty = Double.parseDouble(txt\_qty.getText());

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

Vector v = new Vector();

v.add(txt\_rid.getText());

v.add(combo\_treatment.getSelectedItem().toString().split("\_")[0]);

treatmentQtyDto checkRowRepeating = checkRowRepeating(combo\_treatment.getSelectedItem().toString().split("\_")[0]);

if (checkRowRepeating !=null) {

qty += checkRowRepeating.getQty();

}

v.add(qty);

dtm.addRow(v);

} else {

JOptionPane.showMessageDialog(this, "please fill all the Field","warning",JOptionPane.WARNING\_MESSAGE);

combo\_treatment.grabFocus();

}

}

Delete button

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

try {

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

int selectRow = jTable1.getSelectedRow();

dtm.removeRow(selectRow);

} catch (ArrayIndexOutOfBoundsException e) {

JOptionPane.showMessageDialog(this, "please Select Row","warning",JOptionPane.WARNING\_MESSAGE);

}

}

Process button

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

try {

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

for (int row=0; row < dtm.getRowCount(); row++) {

String rid = jTable1.getValueAt(row, 0).toString();

String treatment = jTable1.getValueAt(row, 1).toString();

String qty = jTable1.getValueAt(row, 2).toString();

String searchSql = "SELECT COUNT(\*) AS countTid FROM treatment";

ResultSet rs = db.search(searchSql);

if (rs.next()){

int count = rs.getInt("countTid") + 1;

String treatmentSql = "INSERT INTO treatment(TID, Receipt\_id, treatment, qty, status\_id) VALUES ('"+count+"', '"+rid+"', '"+treatment+"', '"+qty+"', '1')";

db.push(treatmentSql);

}

String drugStockSql = "UPDATE drug SET qty = qty - '"+qty+"' WHERE drug\_id = '"+treatment+"'";

db.push(drugStockSql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

dispose();

}

} catch (Exception e) {

e.printStackTrace();

}

}

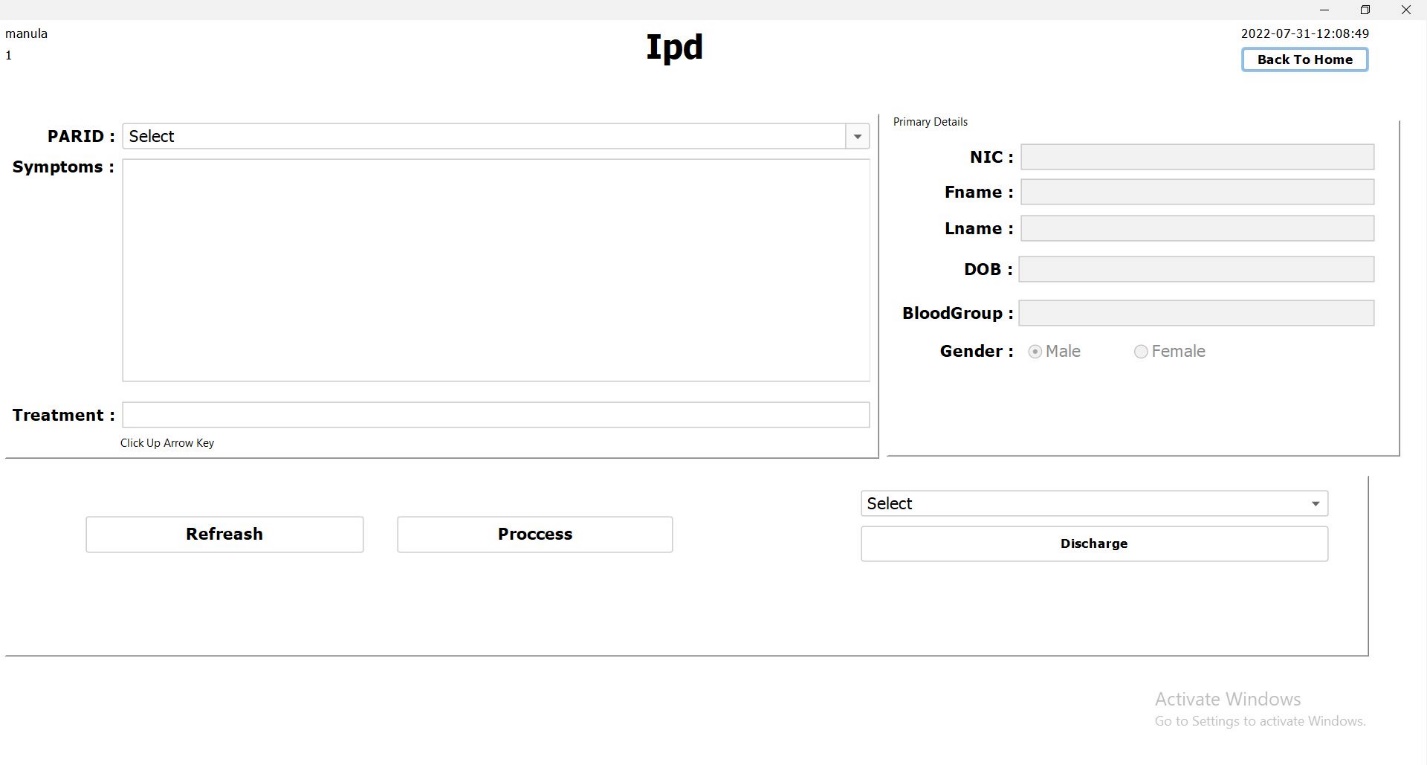
Back to opd button

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

dispose();

}

IPD Doctor



Constructs and variables

public IPD\_Doctor() {

initComponents();

setLocationRelativeTo(null);

loadPARID();

loadDischarge();

AutoCompleteDecorator.decorate(combo\_rid);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methods

private void loadPARID() {

try {

String sql = "SELECT \* FROM ipd\_p\_a\_registration WHERE status\_id in ('3', '2' )";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("PARID");

v.add(data);

}

combo\_rid.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void clearFields() {

txt\_bg.setText(null);

txt\_dob.setText(null);

txt\_fname.setText(null);

txt\_lname.setText(null);

txt\_nic.setText(null);

txt\_symptoms.setText(null);

txt\_treatment.setText(null);

combo\_rid.setSelectedIndex(0);

rdo\_M.setSelected(true);

}

private void loadDischarge() {

try {

String sql = "SELECT \* FROM ipd\_p\_a\_registration WHERE status\_id = '2'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("PARID");

v.add(data);

}

combo\_discharge.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

PARID : combo box

private void combo\_ridActionPerformed(java.awt.event.ActionEvent evt) {

try {

String PARID = combo\_rid.getSelectedItem().toString();

String sql = "SELECT \* FROM patient WHERE nic = (SELECT NIC FROM ipd\_p\_a\_registration WHERE PARID = '"+PARID+"')";

ResultSet rs = db.search(sql);

if (rs.next()) {

txt\_nic.setText(rs.getString("nic"));

txt\_fname.setText(rs.getString("fname"));

txt\_lname.setText(rs.getString("lname"));

txt\_dob.setText(rs.getString("dob"));

String searchSql = "SELECT name FROM blood\_type WHERE id = '"+rs.getString("blood-type\_id")+"'";

ResultSet rsblood = db.search(searchSql);

if (rsblood.next())

txt\_bg.setText(rsblood.getString("name"));

String gender = rs.getString("gender\_id");

if (gender.equals("1")) {

rdo\_M.setSelected(true);

} else {

rdo\_F.setSelected(true);

}

txt\_symptoms.grabFocus();

}else{

clearFields();

}

} catch (Exception e) {

e.printStackTrace();

}

}

Treatment : text field

private void txt\_treatmentKeyReleased(java.awt.event.KeyEvent evt) {

if(evt.getKeyCode() == KeyEvent.VK\_UP) {

if (!combo\_rid.getSelectedItem().toString().equals("Select")){

treatmentQtyDto.setPARID(combo\_rid.getSelectedItem().toString());

IPDTreatment t = new IPDTreatment();

t.setVisible(true);

txt\_treatment.setText(treatmentQtyDto.getPARID());

} else {

JOptionPane.showMessageDialog(this, "Please Select PARID","warning", JOptionPane.WARNING\_MESSAGE);

combo\_rid.grabFocus();

}

}

}

Refreash button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

IPD\_Doctor od = new IPD\_Doctor();

od.setVisible(true);

dispose();

}

Process button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String parid = combo\_rid.getSelectedItem().toString();

String symptoms = txt\_symptoms.getText();

String treatment = txt\_treatment.getText();

String searchSql = "SELECT \* FROM ipd\_doctor WHERE PARID = '"+parid+"'";

ResultSet rs = db.search(searchSql);

if (!parid.equals("Select") && !symptoms.trim().equals("") &&

symptoms != null && treatment != null && !treatment.trim().equals("")) {

rs.next();

String DOCsql = "INSERT INTO ipd\_doctor (`PARID`,`symptoms`,`treatment`,`doctor`,`date`) VALUES ('"+parid+"','"+symptoms+"','"+treatment+"','"+jLabel8.getText()+"','"+jLabel14.getText()+"')";

db.push(DOCsql);

String ridStatUpdateSql = "UPDATE ipd\_p\_a\_registration SET status\_id = '2' WHERE PARID = '"+parid+"'";

db.push(ridStatUpdateSql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

clearFields();

loadDischarge();

} else {

JOptionPane.showMessageDialog(this, "Please Fill All The Fields!","warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Discharge button

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

try {

String parid = combo\_discharge.getSelectedItem().toString();

String searchSql = "SELECT \* FROM ipd\_p\_a\_registration WHERE PARID = '"+parid+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

String sql = "UPDATE ipd\_p\_a\_registration SET status\_id='1' WHERE PARID='"+parid+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Not Defind","warning", JOptionPane.WARNING\_MESSAGE);

}

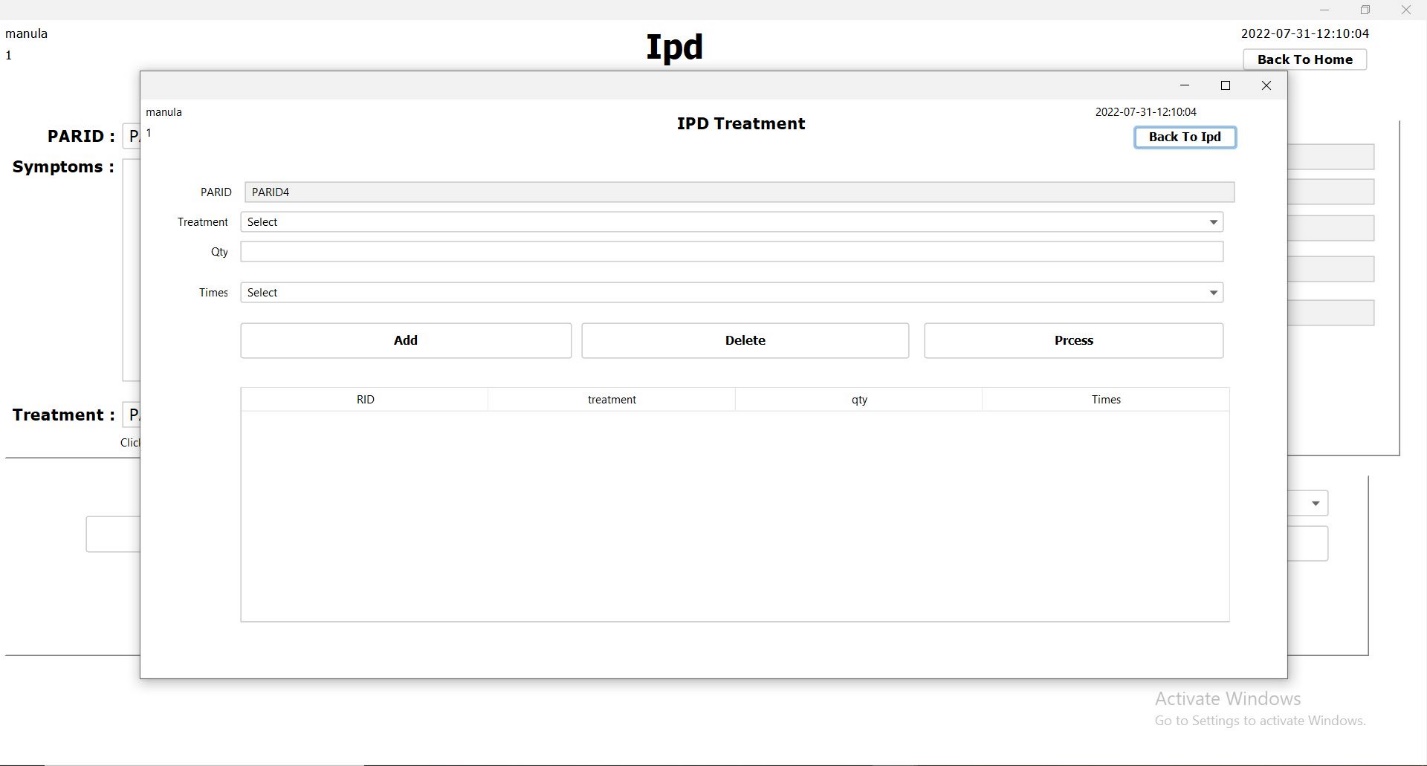
} catch (Exception e) {

e.printStackTrace();

}

}

IPD Treatment



Constructs and variables

public IPDTreatment() {

initComponents();

setLocationRelativeTo(null);

loadTimes();

loadTreatment();

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

txt\_rid.setText(treatmentQtyDto.getPARID());

labal\_un.setText(SystemConfigData.getActiveUser());

labal\_ut.setText(SystemConfigData.getUserType());

}

Other methods

private void loadTimes() {

try {

String searchSql = "SELECT \* FROM treatment\_times";

ResultSet rs = db.search(searchSql);

Vector v = new Vector();

v.add("Select");

while (rs.next()){

String data = rs.getString("code") + "\_" + rs.getString("time");

v.add(data);

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadTreatment() {

try {

String sql = "SELECT \* FROM drug WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("drug\_id") + "\_" + rs.getString("name") + "\_QTY " + rs.getString("qty") + " ";

v.add(data);

}

combo\_treatment.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private treatmentQtyDto checkRowRepeating(String text) {

treatmentQtyDto tqd = null;

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

for (int row = 0; row < jTable1.getRowCount(); row++){

if (text.trim().equals(jTable1.getValueAt(row, 0).toString())){

tqd = new treatmentQtyDto();

double qty = Double.parseDouble(jTable1.getValueAt(row , 2).toString());

tqd.setQty(qty);

dtm.removeRow(row);

break;

}

}

return tqd;

}

Qty text field

private void txt\_qtyActionPerformed(java.awt.event.ActionEvent evt) {

if (!txt\_qty.getText().trim().equals("") && txt\_qty.getText()!= null && !combo\_treatment.getSelectedItem().toString().equals("Select")) {

double qty = Double.parseDouble(txt\_qty.getText());

double ablQty = Double.parseDouble(combo\_treatment.getSelectedItem().toString().split(" ")[2]);

if (qty<ablQty) {

btn\_add.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Qty Not Available","warning", JOptionPane.WARNING\_MESSAGE);

txt\_qty.setText(null);

txt\_qty.grabFocus();

}

} else {

JOptionPane.showMessageDialog(this, "please fill all the Field","warning", JOptionPane.WARNING\_MESSAGE);

combo\_treatment.grabFocus();

}

}

private void txt\_qtyKeyTyped(java.awt.event.KeyEvent evt) {

String qty1 = txt\_qty.getText();

String text = qty1 + evt.getKeyChar();

if (!Pattern.compile("[1-9][0-9]\*").matcher(text).matches()) {

evt.consume();

}

if (!Character.isDigit(evt.getKeyChar())) {

evt.consume();

}

if (!txt\_qty.getText().trim().equals("") && !combo\_treatment.getSelectedItem().toString().equals("Select")) {

double qty = Double.parseDouble(txt\_qty.getText());

double ablQty = Double.parseDouble(combo\_treatment.getSelectedItem().toString().split("\_QTY")[1]);

// System.out.println(ablQty);

if (qty<ablQty) {

// btn\_add.grabFocus();

} else {

JOptionPane.showMessageDialog(this, "Qty Not Available","warning", JOptionPane.WARNING\_MESSAGE);

txt\_qty.setText(null);

txt\_qty.grabFocus();

}

}

// else {

// JOptionPane.showMessageDialog(this, "please fill all the Field","warning", JOptionPane.WARNING\_MESSAGE);

// combo\_treatment.grabFocus();

// }

}

Add button

private void btn\_addActionPerformed(java.awt.event.ActionEvent evt) {

if (txt\_qty.getText() != null && !txt\_qty.getText().trim().equals("") && !combo\_treatment.getSelectedItem().equals("Select") && !jComboBox1.getSelectedItem().equals("Select")) {

double qty = Double.parseDouble(txt\_qty.getText());

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

Vector v = new Vector();

v.add(txt\_rid.getText());

v.add(combo\_treatment.getSelectedItem().toString().split("\_")[0]);

treatmentQtyDto checkRowRepeating = checkRowRepeating(combo\_treatment.getSelectedItem().toString().split("\_")[0]);

if (checkRowRepeating !=null) {

qty += checkRowRepeating.getQty();

}

v.add(qty);

v.add(jComboBox1.getSelectedItem().toString().split("\_")[0]);

dtm.addRow(v);

} else {

JOptionPane.showMessageDialog(this, "please fill all the Field","warning", JOptionPane.WARNING\_MESSAGE);

combo\_treatment.grabFocus();

}

}

Delete button

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

try {

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

int selectRow = jTable1.getSelectedRow();

dtm.removeRow(selectRow);

} catch (ArrayIndexOutOfBoundsException e) {

JOptionPane.showMessageDialog(this, "please Select Row","warning", JOptionPane.WARNING\_MESSAGE);

}

}

Process button

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

try {

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

for (int row=0; row < dtm.getRowCount(); row++) {

String PARID = jTable1.getValueAt(row, 0).toString();

String treatment = jTable1.getValueAt(row, 1).toString();

String qty = jTable1.getValueAt(row, 2).toString();

String time = jTable1.getValueAt(row, 3).toString();

String searchSql = "SELECT COUNT(\*) AS countTid FROM ipd\_treatment";

ResultSet rs = db.search(searchSql);

if (rs.next()){

int count = rs.getInt("countTid") + 1;

String treatmentSql = "INSERT INTO ipd\_treatment (IPDTID, PARID, treatment, qty, times, status\_id) VALUES ('"+count+"', '"+PARID+"', '"+treatment+"', '"+qty+"', '"+time+"', '1')";

db.push(treatmentSql);

}

String drugStockSql = "UPDATE drug SET qty = qty - '"+qty+"' WHERE drug\_id = '"+treatment+"'";

db.push(drugStockSql);

}

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

dispose();

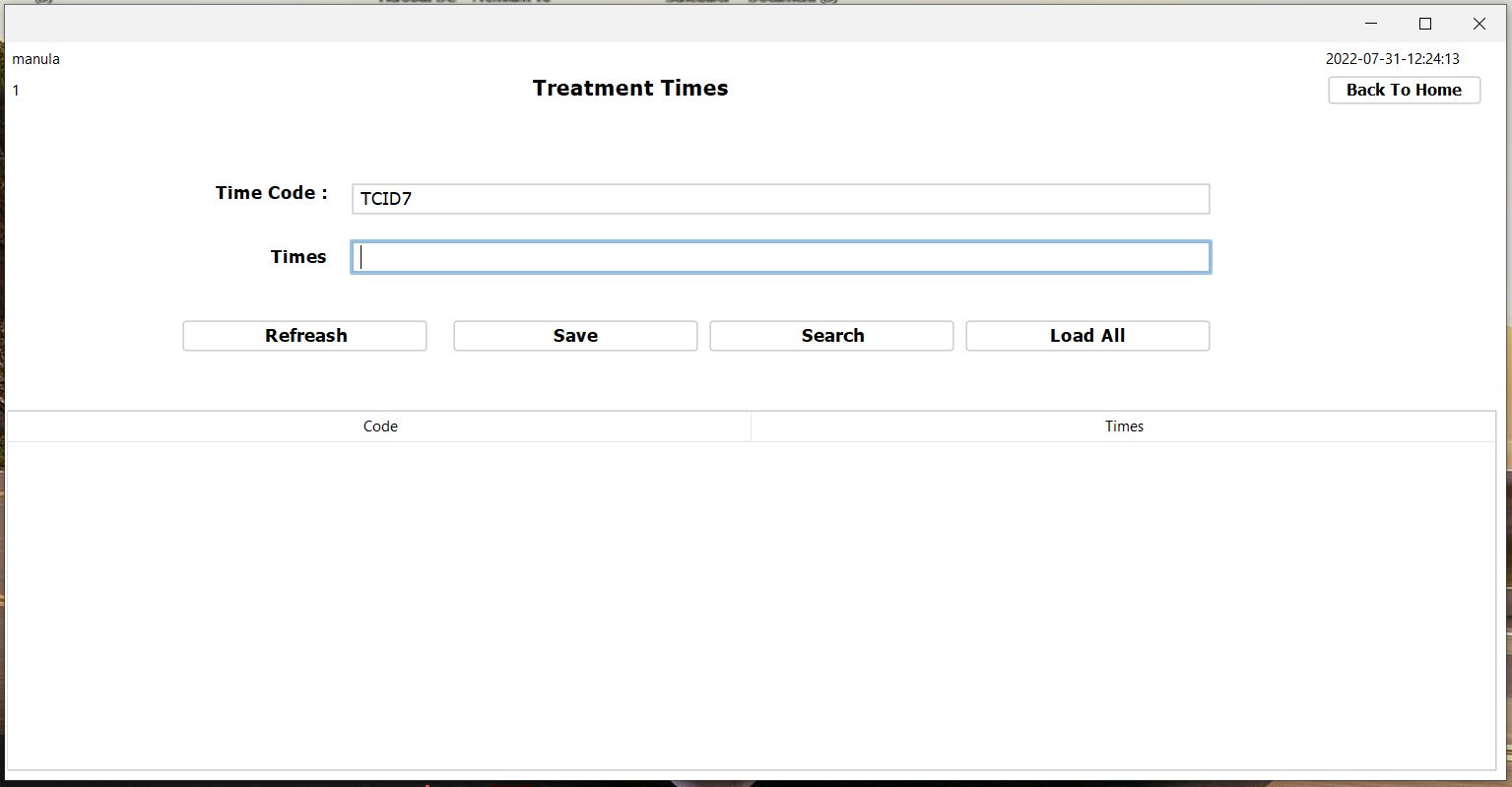
} catch (Exception e) {

e.printStackTrace();

}

}

Treatment Times



Constructors and other variables

public TreatmentTimes() {

initComponents();

setLocationRelativeTo(null);

genarateTimeCode();

timeDate td = new timeDate();

td.timeAndDate(jLabel1);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methods

private void genarateTimeCode() {

try {

String sql = "SELECT COUNT(\*) AS code\_count FROM treatment\_times";

ResultSet rs = db.search(sql);

if (rs.next()){

int count = rs.getInt("code\_count");

String utid = "TCID" + (++count);

txt\_code.setText(utid);

}

txt\_time.grabFocus();

} catch (Exception e) {

e.printStackTrace();

}

}

Refreash button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

TreatmentTimes ut = new TreatmentTimes();

ut.setVisible(true);

dispose();

}

Save button

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

try {

String code = txt\_code.getText();

String description = txt\_time.getText();

String searchSql = "SELECT \* FROM treatment\_times WHERE code = '"+code+"'";

ResultSet rs = db.search(searchSql);

if (!code.equals("") && !description.equals("")) {

if (!rs.next()){

String sql = "INSERT INTO treatment\_times values ('"+code+"','"+description+"')" ;

db.push(sql);

txt\_time.setText(null);

genarateTimeCode();

JOptionPane.showMessageDialog(this, "Saved","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Time Code is already have. please check again!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please fill all the fields","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search button

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

try {

String code = txt\_code.getText();

String searchSql = "SELECT \* FROM treatment\_times WHERE code = '"+code+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()){

txt\_time.setText(rs.getString("time"));

} else {

JOptionPane.showMessageDialog(this, "Time Code is wrong . please check again!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

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

try {

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

dtm.setRowCount(0);

String sql = "SELECT \* FROM treatment\_times";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()){

Vector v = new Vector();

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

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

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String code = dtm.getValueAt(selectRow, 0).toString();

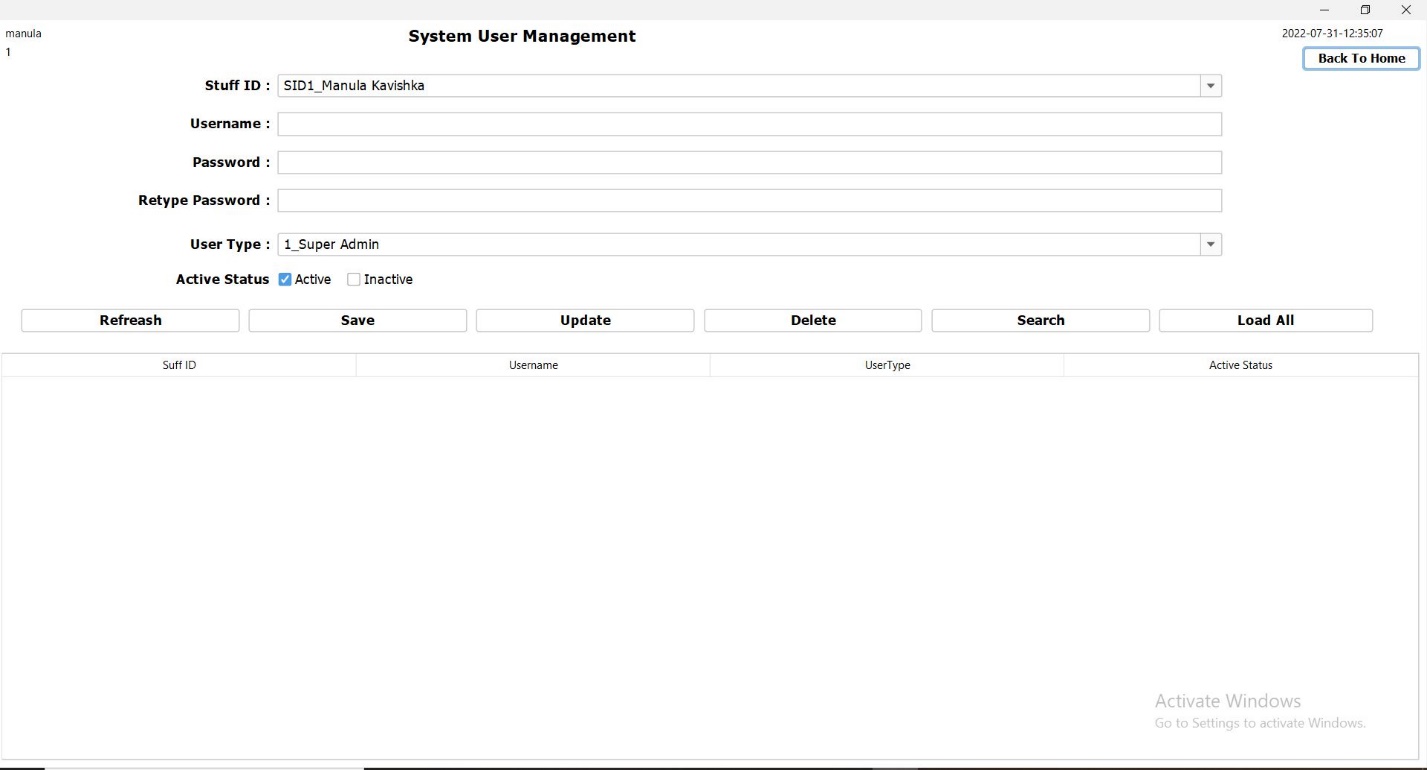
String description = dtm.getValueAt(selectRow, 1).toString();

txt\_code.setText(code);

txt\_time.setText(description);

}

System User



Constructors and other variables

public SystemUserManagement() {

initComponents();

setLocationRelativeTo(null);

AutoCompleteDecorator.decorate(jComboBox1);

AutoCompleteDecorator.decorate(jComboBox2);

loadStuff();

loadUserType();

JLabel x = jLabel1;

timeDate td = new timeDate();

td.timeAndDate(x);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methods

private void loadStuff() {

try {

String sql = "SELECT \* FROM stuff";

ResultSet rs = db.search(sql);

Vector v = new Vector();

while (rs.next()) {

String data = rs.getString("sid") + "\_" + rs.getString("fname") + " " + rs.getString("lname");

v.add(data);

}

jComboBox2.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadUserType() {

try {

String sql = "SELECT \* FROM staff\_type";

ResultSet rs = db.search(sql);

Vector v = new Vector();

while (rs.next()){

String data = rs.getString("id") + "\_" + rs.getString("name");

v.add(data);

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void clearField() {

txt\_username.setText(null);

txt\_repassword.setText(null);

txt\_password.setText(null);

jComboBox1.setSelectedIndex(0);

jComboBox2.setSelectedIndex(0);

acheckbox.setSelected(true);

}

private void jComboBox2ActionPerformed(java.awt.event.ActionEvent evt) {

txt\_username.grabFocus();

}

private void txt\_usernameActionPerformed(java.awt.event.ActionEvent evt) {

txt\_password.grabFocus();

}

private void txt\_passwordActionPerformed(java.awt.event.ActionEvent evt) {

txt\_repassword.grabFocus();

}

private void txt\_repasswordActionPerformed(java.awt.event.ActionEvent evt) {

jComboBox1.grabFocus();

}

private void jComboBox1ActionPerformed(java.awt.event.ActionEvent evt) {

acheckbox.grabFocus();

}

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

jButton1.grabFocus();

}

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

jButton1.grabFocus();

}

Refreash button

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

SystemUserManagement sum = new SystemUserManagement();

sum.setVisible(true);

dispose();

}

Save button

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

try {

String sname = jComboBox2.getSelectedItem().toString().split("\_")[0];

String uname = txt\_username.getText();

String ps = new String (txt\_password.getPassword());

String reps = new String (txt\_repassword.getPassword());

String utype = jComboBox1.getSelectedItem().toString().split("\_")[0];

int activeStatus = (acheckbox.isSelected()) ? 1 : 2;

if (!uname.equals("") && !ps.equals("") && !reps.equals("") ){

String searchSql = "SELECT \* FROM system\_user WHERE username = '"+uname+"'";

ResultSet rs = db.search(searchSql);

if (!rs.next()){

if (ps.equals(reps)) {

String encPs = MD5.getMd5(ps);

String sql = "INSERT INTO system\_user VALUES ('"+uname+"','"+sname+"',"

+ "'"+encPs+"','"+utype+"','"+activeStatus+"')";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

clearField();

} else {

JOptionPane.showMessageDialog(this, "Password Not Maching!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "User name already exist!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please fill all the fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Update button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String sname = jComboBox2.getSelectedItem().toString().split("\_")[0];

String uname = txt\_username.getText();

String ps = new String (txt\_password.getPassword());

String reps = new String (txt\_repassword.getPassword());

String utype = jComboBox1.getSelectedItem().toString().split("\_")[0];

int activeStatus = (acheckbox.isSelected()) ? 1 : 2;

if (!uname.equals("") && !ps.equals("") && !reps.equals("")){

String searchSql = "SELECT \* FROM system\_user WHERE username = '"+uname+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()){

if (ps.equals(reps)) {

String encPs = MD5.getMd5(ps);

String sql = "UPDATE system\_user SET username = '"+uname+"' ,"

+ " sid ='"+sname+"', password = '"+encPs+"',"

+ " staff\_type\_id = '"+utype+"' , status\_id = '"+activeStatus+"'"

+ " WHERE username = '"+uname+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

clearField();

} else {

JOptionPane.showMessageDialog(this, "Password Not Maching!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "User name not exist!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please fill all the fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Delete button

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

try {

String uname = txt\_username.getText();

if (!uname.equals("")){

String sql = "UPDATE system\_user SET status\_id= '2' WHERE username = '"+uname+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "please type username","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search butt

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

try {

String uname = txt\_username.getText();

String sql = "SELECT \* FROM system\_user WHERE username = '"+uname+"'";

ResultSet rs = db.search(sql);

if (rs.next()) {

String utype = rs.getString("staff\_type\_id");

String userSql = "SELECT \* FROM staff\_type WHERE id = '"+utype+"'";

ResultSet userRs = db.search(userSql);

if (userRs.next()) {

String user = userRs.getString("id") + "\_" + userRs.getString("name");

jComboBox1.setSelectedItem(user);

}

String activestatus = rs.getString("status\_id");

if (activestatus.equals("1") ) {

acheckbox.setSelected(true);

}

else {

icheckbox.setSelected(true);

}

} else {

JOptionPane.showMessageDialog(this, "username Not matching","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

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

try {

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

dtm.setRowCount(0);

String sql = "SELECT \* FROM system\_user";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()){

Vector v = new Vector();

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

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

String sql1 = "SELECT \* FROM staff\_type WHERE id = '"+rs.getString("staff\_type\_id")+"'";

ResultSet userRs = db.search(sql1);

if (userRs.next()) {

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

}

String active = rs.getString("status\_id");

if (active.equals("1")) {

String statusA = "Active";

v.add(statusA);

} else {

String statusA = "Inactive";

v.add(statusA);

}

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

try {

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

int selectRow = jTable1.getSelectedRow();

String sname = dtm.getValueAt(selectRow, 0).toString();

String uname = dtm.getValueAt(selectRow, 1).toString();

String utype = dtm.getValueAt(selectRow, 2).toString();

String active = dtm.getValueAt(selectRow, 3).toString();

txt\_username.setText(uname);

String userSql = "SELECT \* FROM staff\_type WHERE name = '"+utype+"'";

ResultSet userRs = db.search(userSql);

if (userRs.next()) {

String user = userRs.getString("id") + "\_" + userRs.getString("name");

jComboBox1.setSelectedItem(user);

}

String stuffsql = "SELECT \* FROM stuff WHERE sid = '"+sname+"'";

ResultSet stuffRs = db.search(stuffsql);

if (stuffRs.next()){

String stuffName = stuffRs.getString("sid") + "\_" + stuffRs.getString("fname") + " " + stuffRs.getString("lname");

jComboBox2.setSelectedItem(stuffName);

}

if (active.equals("Active")) {

acheckbox.setSelected(true);

} else {

icheckbox.setSelected(true);

}

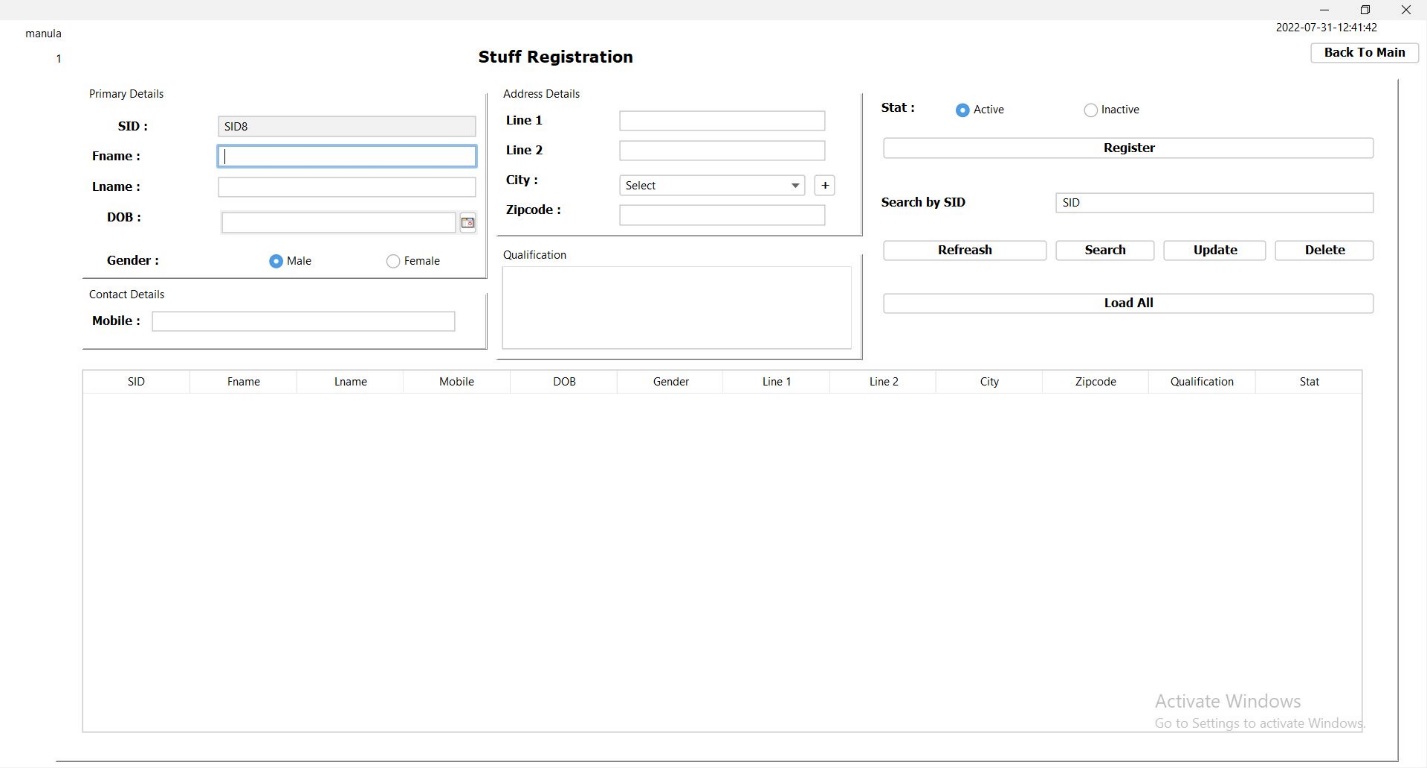
} catch (Exception e) {

e.printStackTrace();

}

}

Stuff registration



Constructors and other variables

public stuffMangement() {

initComponents();

setLocationRelativeTo(null);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

genarateSID();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

loadCity();

}

Other methods

private void genarateSID() {

try {

String sql = "SELECT COUNT(\*) AS stuff\_count FROM stuff";

ResultSet rs = db.search(sql);

if (rs.next()){

int count = rs.getInt("stuff\_count");

String sid = "SID" + (++count);

txt\_sid.setText(sid);

}

txt\_fname.grabFocus();

} catch (Exception e) {

e.printStackTrace();

}

}

private void clearField() {

txt\_fname.setText(null);

txt\_lname.setText(null);

rdo\_m.setSelected(true);

txt\_mobile.setText(null);

txt\_address1.setText(null);

txt\_address.setText(null);

jComboBox1.setSelectedIndex(0);

txt\_zipcode.setText(null);

txt\_qualification.setText(null);

rdo\_a.setSelected(true);

time\_date.setCalendar(null);

genarateSID();

}

public void loadCity() {

try {

String sql = "SELECT \* FROM city";

ResultSet rs = db.search(sql);

Vector v = new Vector();

v.add("Select");

while (rs.next()) {

String data = rs.getString("id") + "\_" + rs.getString("name");

v.add(data);

}

jComboBox1.setModel(new DefaultComboBoxModel(v));

} catch (Exception e) {

e.printStackTrace();

}

}

private void txt\_fnameActionPerformed(java.awt.event.ActionEvent evt) {

txt\_lname.grabFocus();

}

private void txt\_lnameActionPerformed(java.awt.event.ActionEvent evt) {

time\_date.grabFocus();

}

private void rdo\_mActionPerformed(java.awt.event.ActionEvent evt) {

txt\_mobile.grabFocus();

}

private void rdo\_fActionPerformed(java.awt.event.ActionEvent evt) {

txt\_mobile.grabFocus();

}

private void txt\_mobileActionPerformed(java.awt.event.ActionEvent evt) {

txt\_address1.grabFocus();

}

private void txt\_address1ActionPerformed(java.awt.event.ActionEvent evt) {

txt\_address.grabFocus();

}

private void txt\_zipcodeActionPerformed(java.awt.event.ActionEvent evt) {

txt\_qualification.grabFocus();

}

private void txt\_addressActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

jComboBox1.grabFocus();

}

DOB : date chooser

private void time\_datePropertyChange(java.beans.PropertyChangeEvent evt) {

// TODO add your handling code here:

try {

Date date = time\_date.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd-HH:mm:ss");

String numberdate = sdf.format(date);

String s = jLabel14.getText();

if (s.compareTo(numberdate) > 0) {

// System.out.println(s.compareTo(numberdate) > 0);

} else {

JOptionPane.showMessageDialog(this, "Please choose correct date", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (NullPointerException n) {

} catch (Exception e) {

e.printStackTrace();

}

}

Mobile : text field

private void txt\_mobileKeyTyped(java.awt.event.KeyEvent evt) {

if (!Character.isDigit(evt.getKeyChar())){

evt.consume();

}

if (!Character.isDigit(evt.getKeyChar())){

evt.consume();

}

String mobile = txt\_mobile.getText();

String text = mobile + evt.getKeyChar();

if (text.length() == 1) {

if (!text.equals("0")) {

evt.consume();

}

} else if (text.length() == 2) {

if (!text.equals("07")) {

evt.consume();

}

} else if (text.length() == 3) {

if (!Pattern.compile("07[1245678]").matcher((text)).matches()) {

evt.consume();

}

} else if (text.length() <= 10) {

if (!Pattern.compile("07[1245678][0-9]+").matcher((text)).matches()) {

evt.consume();

}

} else {

evt.consume();

}

}

City : + button

private void jButton9ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

CityRegistration c = new CityRegistration(this, true,2);

c.setVisible(true);

}

Register button

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

try {

String sid = txt\_sid.getText();

String fname = txt\_fname.getText();

String lname = txt\_lname.getText();

String gender = (rdo\_m.isSelected()) ? "1" : "2";

String mobile = txt\_mobile.getText();

String address = txt\_address.getText();

String address1 = txt\_address1.getText();

String city = jComboBox1.getSelectedItem().toString().split("\_")[0];

String zipcode = txt\_zipcode.getText();

String qualification = txt\_qualification.getText();

int stat = (rdo\_a.isSelected()) ? 1 : 2;

Date date = time\_date.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

if (!sid.equals("") && !fname.equals("") && !lname.equals("") &&

!numberdate.equals("") && !address1.equals("") && !address.equals("") &&

!mobile.equals("") && !zipcode.equals("") && !city.equals("Select") ) {

String searchSql = "SELECT \* FROM stuff WHERE sid = '"+sid+"'";

ResultSet rs = db.search(searchSql);

if (!rs.next()) {

ResultSet addressRs = db.search("SELECT \* FROM address WHERE addresss\_line1 ='"+address1+"' AND address\_line2 ='"+address+"' AND postal\_code='"+zipcode+"' AND city\_id ='"+city+"'");

if (!addressRs.next()) {

db.push("INSERT INTO address (addresss\_line1,address\_line2,postal\_code,city\_id) VALUES ('"+address1+"','"+address+"','"+zipcode+"','"+city+"')");

}

ResultSet addressIdRs = db.search("SELECT \* FROM address WHERE addresss\_line1 ='"+address1+"' AND address\_line2 ='"+address+"' AND postal\_code='"+zipcode+"' AND city\_id ='"+city+"'");

addressIdRs.next();

String addrressId = addressIdRs.getString("id");

String sql = "INSERT INTO stuff VALUES ('"+sid+"','"+fname+"',"

+ "'"+lname+"','"+mobile+"','"+numberdate+"',"

+ "'"+gender+"','"+addrressId+"',"

+ "'"+qualification+"','"+stat+"')";

db.push(sql);

clearField();

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "SID already defined!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All the fields","warning",JOptionPane.WARNING\_MESSAGE);

}

}catch (NullPointerException n) {

JOptionPane.showMessageDialog(this, "Please Fill DOB","warning",JOptionPane.WARNING\_MESSAGE);

clearField();

}

catch (Exception e) {

e.printStackTrace();

}

}

Refreash button

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

stuffMangement sm = new stuffMangement();

sm.setVisible(true);

dispose();

}

Search button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String sbSID = txt\_searchbySID.getText();

String sql = "SELECT \* FROM stuff WHERE sid ='"+sbSID+"'";

ResultSet rs = db.search(sql);

if (!sbSID.equals("")){

if (rs.next()) {

txt\_sid.setText(rs.getString("sid"));

txt\_fname.setText(rs.getString("fname"));

txt\_lname.setText(rs.getString("lname"));

txt\_mobile.setText(rs.getString("mobile"));

Date dob = rs.getDate("dob");

time\_date.setDate(dob);

String gender = rs.getString("gender\_id");

if (gender.equals("1")){

rdo\_m.setSelected(true);

} else {

rdo\_f.setSelected(true);

}

ResultSet addressRs = db.search("SELECT \* FROM address WHERE `id`='"+rs.getString("address\_id")+"' ");

addressRs.next();

txt\_address.setText(addressRs.getString("address\_line2"));

txt\_address1.setText(addressRs.getString("addresss\_line1"));

ResultSet cityRs = db.search("SELECT \* FROM city WHERE id='"+addressRs.getString("city\_id")+"'");

cityRs.next();

jComboBox1.setSelectedItem(cityRs.getString("name"));

txt\_zipcode.setText(addressRs.getString("postal\_code"));

txt\_qualification.setText(rs.getString("qualification"));

String status = rs.getString("status\_id");

if (status.equals("1")) {

rdo\_a.setSelected(true);

} else {

rdo\_i.setSelected(true);

}

} else {

JOptionPane.showMessageDialog(this, "Wrong SID. Please Try Again!","warning",JOptionPane.INFORMATION\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Search By SID Field!","warning",JOptionPane.INFORMATION\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Update button

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String sid = txt\_sid.getText();

String fname = txt\_fname.getText();

String lname = txt\_lname.getText();

String sbNIC = txt\_searchbySID.getText();

String gender = (rdo\_m.isSelected()) ? "1" : "2";

String mobile = txt\_mobile.getText();

String address = txt\_address.getText();

String address1 = txt\_address1.getText();

String city = jComboBox1.getSelectedItem().toString().split("\_")[0];

String zipcode = txt\_zipcode.getText();

String qualification = txt\_qualification.getText();

int stat = (rdo\_a.isSelected()) ? 1 : 2;

Date date = time\_date.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

if (!sbNIC.equals("") && !sid.equals("") && !fname.equals("") && !lname.equals("") &&

!numberdate.equals("") && !address1.equals("") && !address.equals("") &&

!mobile.equals("") && !city.equals("") && !zipcode.equals("") ) {

String searchSql = "SELECT \* FROM stuff WHERE sid = '"+sbNIC+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

ResultSet addressRs = db.search("SELECT \* FROM address WHERE id IN (SELECT address\_id FROM patient WHERE nic = '"+sbNIC+"')");

if (addressRs.next()) {

db.push("UPDATE `address` SET addresss\_line1='"+address1+"', `address\_line2`='"+address+"', `postal\_code`='"+zipcode+"', `city\_id`='"+city+"' WHERE id = '"+rs.getString("address\_id")+"' ");

}

String sql = "UPDATE stuff SET fname='"+fname+"',"

+ "lname='"+lname+"',mobile='"+mobile+"',dob='"+numberdate+"',"

+ "gender\_id='"+gender+"',"

+ "qualification='"+qualification+"',status\_id='"+stat+"' WHERE sid = '"+sbNIC+"'";

db.push(sql);

clearField();

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "SID not defined!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All the fields","warning",JOptionPane.WARNING\_MESSAGE);

}

}catch (NullPointerException n) {

JOptionPane.showMessageDialog(this, "Please Fill DOB","warning",JOptionPane.WARNING\_MESSAGE);

clearField();

}

catch (Exception e) {

e.printStackTrace();

}

}

Delete button

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String sbSID = txt\_searchbySID.getText();

String searchSql = "SELECT \* FROM stuff WHERE sid = '"+sbSID+"'";

ResultSet rs = db.search(searchSql);

if (!sbSID.equals("")) {

if (rs.next()) {

String sql = "UPDATE stuff SET status\_id = '2' WHERE sid = '"+sbSID+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Wrong SID. Please Try Again!","warning",JOptionPane.INFORMATION\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Search By SID Field!","warning",JOptionPane.INFORMATION\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

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

dtm.setRowCount(0);

String sql = "SELECT \* FROM stuff";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

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

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

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

ResultSet genderRs = db.search("SELECT \* FROM gender WHERE id = '"+rs.getString("gender\_id")+"'");

genderRs.next();

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

ResultSet addressRs = db.search("SELECT \* FROM address WHERE id IN (SELECT address\_id FROM stuff WHERE sid = '"+rs.getString("sid")+"')");

addressRs.next();

v.add(addressRs.getString("addresss\_line1"));

v.add(addressRs.getString("address\_line2"));

ResultSet cityRs = db.search("SELECT \* FROM city WHERE id='"+addressRs.getString("city\_id")+"'");

cityRs.next();

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

v.add(addressRs.getString("postal\_code"));

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

String stat = rs.getString("status\_id");

if (stat.equals("1")) {

v.add("Active");

} else {

v.add("Inactive");

}

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String sid = dtm.getValueAt(selectRow, 0).toString();

String fname = dtm.getValueAt(selectRow, 1).toString();

String lname = dtm.getValueAt(selectRow, 2).toString();

String mobile = dtm.getValueAt(selectRow, 3).toString();

String dbo = dtm.getValueAt(selectRow, 4).toString();

String gender = dtm.getValueAt(selectRow, 5).toString();

String address1 = dtm.getValueAt(selectRow, 6).toString();

String address = dtm.getValueAt(selectRow, 7).toString();

String city = dtm.getValueAt(selectRow, 8).toString();

String zipcode = dtm.getValueAt(selectRow, 9).toString();

String qualification = dtm.getValueAt(selectRow, 10).toString();

String stat = dtm.getValueAt(selectRow, 11).toString();

txt\_sid.setText(sid);

txt\_fname.setText(fname);

txt\_lname.setText(lname);

txt\_mobile.setText(mobile);

try {

Date date = new SimpleDateFormat("yyyy-MM-dd").parse(dbo);

time\_date.setDate(date);

} catch (ParseException ex) {

ex.printStackTrace();

}

if (gender.equals("Male")){

rdo\_m.setSelected(true);

} else {

rdo\_f.setSelected(true);

}

txt\_address1.setText(address1);

txt\_address.setText(address);

try {

ResultSet cityRs = db.search("SELECT \* FROM city WHERE name = '"+city+"'");

cityRs.next();

jComboBox1.setSelectedItem(cityRs.getString("id") + "\_" + cityRs.getString("name"));

} catch (Exception e) {

e.printStackTrace();

}

txt\_zipcode.setText(zipcode);

txt\_qualification.setText(qualification);

if (stat.equals("Active")) {

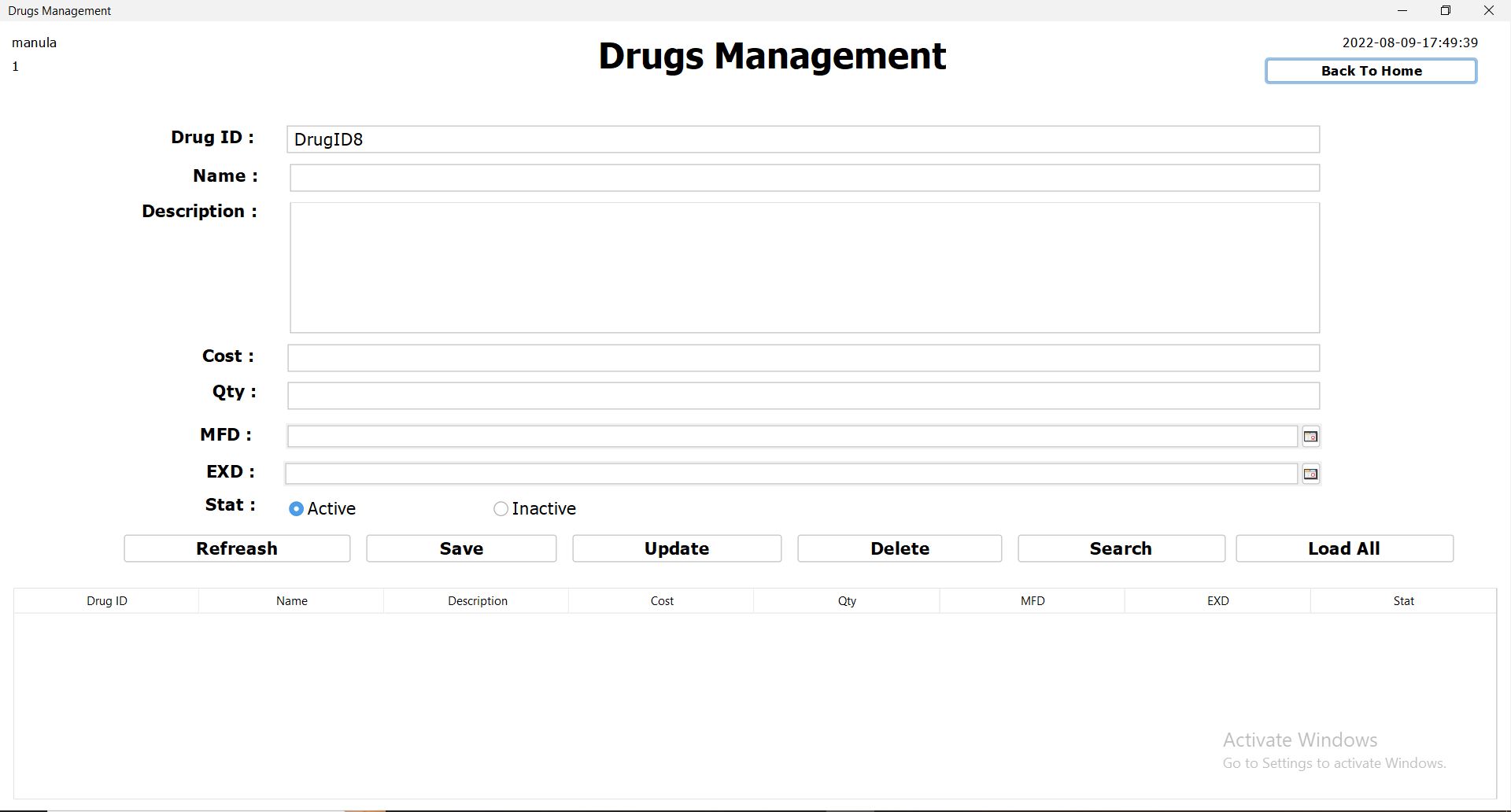
rdo\_a.setSelected(true);

} else {

rdo\_i.setSelected(true);

}

}

Drugs Management

Constructors and other variables

public DrugManagement() {

initComponents();

setLocationRelativeTo(null);

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

genarateDrugID();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methods

private void genarateDrugID() {

try {

String sql = "SELECT COUNT(\*) AS countDrugID FROM drug";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("countDrugID");

String DrugID = "DrugID" + (++count);

txt\_rug\_id.setText(DrugID);

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void clearFields() {

txt\_name.setText("");

txt\_name.setText("");

txt\_description.setText("");

txt\_cost.setText("");

txt\_qty.setText("");

rdo\_a.setSelected(true);

jDateChooser1.setDate(null);

jDateChooser3.setDate(null);

genarateDrugID();

}

private void txt\_rug\_idActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_name.grabFocus();

}

private void txt\_costActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_qty.grabFocus();

}

Cost : text field

private void txt\_costKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String price = txt\_cost.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Qty : text field

private void txt\_qtyKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String qty = txt\_qty.getText();

String text = qty + evt.getKeyChar();

if (!Pattern.compile("[1-9][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

MFD Date Chooser

private void jDateChooser1PropertyChange(java.beans.PropertyChangeEvent evt) {

// TODO add your handling code here:

try {

Date date = jDateChooser1.getDate();

Date date2 = jDateChooser3.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

String numberdate2 = sdf.format(date2);

if (numberdate2.compareTo(numberdate) > 0) {

} else {

jDateChooser1.setDate(null);

jDateChooser3.setDate(null);

JOptionPane.showMessageDialog(this, "Please choose correct date", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (NullPointerException n) {

} catch (Exception e) {

e.printStackTrace();

}

}

EXD Date Chooser

private void jDateChooser3PropertyChange(java.beans.PropertyChangeEvent evt) {

// TODO add your handling code here:

try {

Date date = jDateChooser1.getDate();

Date date2 = jDateChooser3.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

String numberdate2 = sdf.format(date2);

if (numberdate2.compareTo(numberdate) > 0) {

// System.out.println(numberdate2.compareTo(numberdate) < 0);

} else {

jDateChooser1.setDate(null);

jDateChooser3.setDate(null);

JOptionPane.showMessageDialog(this, "Please choose correct date", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (NullPointerException n) {

} catch (Exception e) {

e.printStackTrace();

}

}

Refreash button

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

DrugManagement dm = new DrugManagement();

dm.setVisible(true);

dispose();

}

Save button

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

try {

String drugid = txt\_rug\_id.getText();

String name = txt\_name.getText();

String description = txt\_description.getText();

String cost = txt\_cost.getText();

String qty = txt\_qty.getText();

int stat = (rdo\_a.isSelected()) ? 1: 2;

Date date = jDateChooser1.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

Date date2 = jDateChooser3.getDate();

String numberdate2 = sdf.format(date2);

String searchSql = "SELECT \* FROM drug WHERE drug\_id = '"+drugid+"'";

ResultSet rs = db.search(searchSql);

if (!rs.next()) {

if (!name.equals("") && !description.equals("") && !cost.equals("") && !qty.equals("") && !numberdate.equals("") && !numberdate2.equals("")){

String sql = "INSERT INTO drug VALUES ('"+drugid+"','"+name+"','"+description+"','"+cost+"','"+qty+"','"+numberdate+"','"+numberdate2+"','"+stat+"')";

db.push(sql);

clearFields();

JOptionPane.showMessageDialog(this, "Success","success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "please Fill All The Fields","warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "DrugID Alreasdy Defind","warning", JOptionPane.WARNING\_MESSAGE);

}

}catch (NullPointerException n) {

JOptionPane.showMessageDialog(this, "Please Fill MFD Or EXD", "warning", JOptionPane.WARNING\_MESSAGE);

}

catch (Exception e) {

e.printStackTrace();

}

}

Update button

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

try {

String drugid = txt\_rug\_id.getText();

String name = txt\_name.getText();

String description = txt\_description.getText();

String cost = txt\_cost.getText();

String qty = txt\_qty.getText();

int stat = (rdo\_a.isSelected()) ? 1: 2;

Date date = jDateChooser1.getDate();

SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd");

String numberdate = sdf.format(date);

Date date2 = jDateChooser3.getDate();

String numberdate2 = sdf.format(date2);

String searchSql = "SELECT \* FROM drug WHERE drug\_id = '"+drugid+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

if (!name.equals("") && !description.equals("") && !cost.equals("") && !qty.equals("") && !numberdate.equals("") && !numberdate2.equals("")){

String sql = "UPDATE drug SET name='"+name+"',description='"+description+"',cost='"+cost+"',qty='"+qty+"',status\_id='"+stat+"', mfd='"+numberdate+"',exd='"+numberdate2+"' WHERE drug\_id='"+drugid+"'";

db.push(sql);

clearFields();

JOptionPane.showMessageDialog(this, "Success","success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "please Fill All The Fields","warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "DrugID Not Defind","warning", JOptionPane.WARNING\_MESSAGE);

}

}catch (NullPointerException n) {

JOptionPane.showMessageDialog(this, "Please Fill MFD Or EXD", "warning", JOptionPane.WARNING\_MESSAGE);

} catch (Exception e) {

e.printStackTrace();

}

}

Delete button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String drugid = txt\_rug\_id.getText();

String searchSql = "SELECT \* FROM drug WHERE drug\_id = '"+drugid+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

String sql = "UPDATE drug SET status\_id='2' WHERE drug\_id='"+drugid+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success","success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "DrugID Not Defind","warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search button

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

try {

String drugid = txt\_rug\_id.getText();

String searchSql = "SELECT \* FROM drug WHERE drug\_id = '"+drugid+"'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

txt\_name.setText(rs.getString("name"));

txt\_description.setText(rs.getString("description"));

txt\_cost.setText(rs.getString("cost"));

txt\_qty.setText(rs.getString("qty"));

jDateChooser1.setDate(rs.getDate("mfd"));

jDateChooser3.setDate(rs.getDate("exd"));

String stat = rs.getString("status\_id");

if (stat.equals("1")) {

rdo\_a.setSelected(true);

} else {

rdo\_i.setSelected(true);

}

} else {

JOptionPane.showMessageDialog(this, "DrugID Not Defind","warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

try {

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

String sql = "SELECT \* FROM drug";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

v.add(rs.getString("drug\_id"));

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

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

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

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

v.add(rs.getDate("mfd"));

v.add(rs.getDate("exd"));

String stat = rs.getString("status\_id");

if (stat.equals("1")) {

v.add("Active");

} else {

v.add("Inactive");

}

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String did = dtm.getValueAt(selectRow, 0).toString();

String name = dtm.getValueAt(selectRow, 1).toString();

String description = dtm.getValueAt(selectRow, 2).toString();

String cost = dtm.getValueAt(selectRow, 3).toString();

String qty = dtm.getValueAt(selectRow, 4).toString();

String stat = dtm.getValueAt(selectRow, 7).toString();

txt\_rug\_id.setText(did);

txt\_name.setText(name);

txt\_description.setText(description);

txt\_cost.setText(cost);

txt\_qty.setText(qty);

try {

String mfd = dtm.getValueAt(selectRow, 5).toString();

String exd = dtm.getValueAt(selectRow, 6).toString();

Date date1=new SimpleDateFormat("yyyy-MM-dd").parse(mfd);

Date date2=new SimpleDateFormat("yyyy-MM-dd").parse(exd);

jDateChooser1.setDate(date1);

jDateChooser3.setDate(date2);

}catch (NullPointerException n) {

}

catch (ParseException ex) {

}

if (stat.equals("Active")){

rdo\_a.setSelected(true);

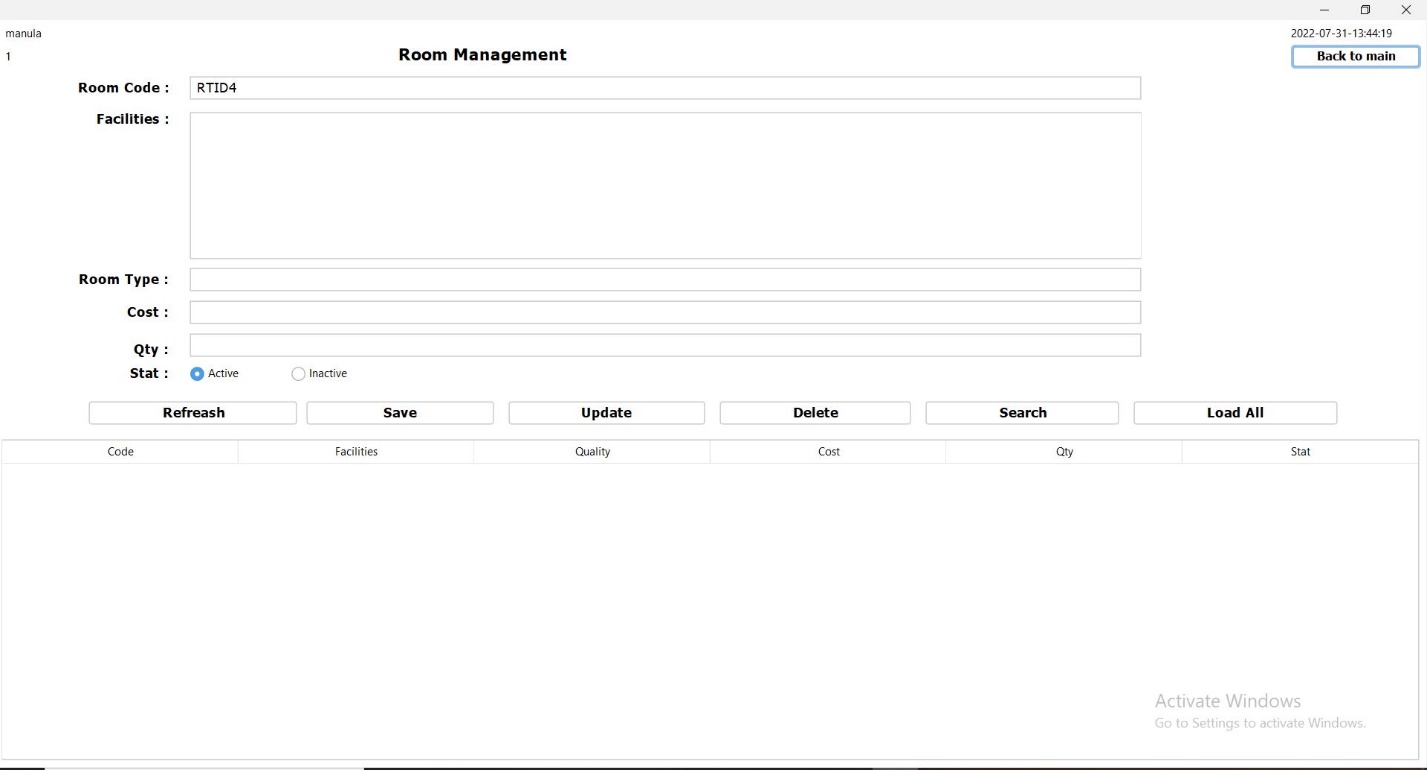
} else {

rdo\_i.setSelected(true);

}

}

Room Management



Constructors and other variables

public RoomType() {

initComponents();

setLocationRelativeTo(null);

genarateRoomID();

timeDate td = new timeDate();

td.timeAndDate(jLabel1);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

Other methods

private void genarateRoomID() {

try {

String sql = "SELECT COUNT(\*) AS roomCount FROM room";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("roomCount");

String rtid = "RTID" + (++count);

txt\_code.setText(rtid);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Cost text field

private void txt\_costKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String price = txt\_cost.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Qty text field

private void txt\_qtyKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String qty = txt\_qty.getText();

String text = qty + evt.getKeyChar();

if (!Pattern.compile("[1-9][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Refreash button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

RoomType rt = new RoomType();

rt.setVisible(true);

dispose();

}

Save button

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

try {

String code = txt\_code.getText();

String facilities = txt\_facilities.getText();

String roomType = txt\_type.getText();

String cost = txt\_cost.getText();

String qty = txt\_qty.getText();

int status = (rdo\_a.isSelected()) ? 1 : 2;

if (!code.equals("") && !facilities.equals("") && !roomType.equals("")

&& !cost.equals("") && !qty.equals("")) {

String Searchsql = "SELECT \* FROM room WHERE room\_no = '"+code+"'";

ResultSet rs = db.search(Searchsql);

if (!rs.next()) {

String sql = "INSERT INTO room VALUES('"+code+"','"+facilities+"','"+roomType+"','"+cost+"','"+qty+"','"+status+"')";

String sql2 = String.format("INSERT INTO room VALUES(%s, %s, %s, %s, %s, %s)", code, facilities, roomType, cost, qty, status);

db.push(sql);

genarateRoomID();

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Room Code Allready Defind!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All The Fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Update button

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

try {

String code = txt\_code.getText();

String facilities = txt\_facilities.getText();

String roomType = txt\_type.getText();

String cost = txt\_cost.getText();

String qty = txt\_qty.getText();

int status = (rdo\_a.isSelected()) ? 1 : 2;

if (!code.equals("") && !facilities.equals("") && !roomType.equals("")

&& !cost.equals("") && !qty.equals("")) {

String Searchsql = "SELECT \* FROM room WHERE room\_no = '"+code+"'";

ResultSet rs = db.search(Searchsql);

if (rs.next()) {

String sql = "UPDATE `room` SET `facilitie`='"+facilities+"',`room\_type`='"+roomType+"',`dayprice`='"+cost+"',`qty`='"+qty+"',`status\_id`='"+status+"' WHERE `room\_no` = '"+code+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Room Code Not Defind!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All The Fields!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Delete button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String code = txt\_code.getText();

if (!code.equals("")) {

String Searchsql = "SELECT \* FROM room WHERE room\_no = '"+code+"'";

ResultSet rs = db.search(Searchsql);

if (rs.next()) {

String sql = "UPDATE `room` SET `status\_id`='2' WHERE `room\_no` = '"+code+"'";

db.push(sql);

JOptionPane.showMessageDialog(this, "Success!","success",JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Room Code Not Defind!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill The Code Field!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search button

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

try {

String code = txt\_code.getText();

String facilities = txt\_facilities.getText();

String roomType = txt\_type.getText();

String cost = txt\_cost.getText();

String qty = txt\_qty.getText();

int status = (rdo\_a.isSelected()) ? 1 : 2;

if (!code.equals("")) {

String Searchsql = "SELECT \* FROM room WHERE room\_no = '"+code+"'";

ResultSet rs = db.search(Searchsql);

if (rs.next()) {

txt\_facilities.setText(rs.getString("facilitie"));

txt\_type.setText(rs.getString("room\_type"));

txt\_cost.setText(rs.getString("dayprice"));

txt\_qty.setText(rs.getString("qty"));

String stat = rs.getString("status\_id");

if (stat.equals("1")) {

rdo\_a.setSelected(true);

} else {

rdo\_i.setSelected(true);

}

} else {

JOptionPane.showMessageDialog(this, "Room Code Not Defind!","warning",JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill The Code Field!","warning",JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

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

try {

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

String sql = "SELECT \* FROM room";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

v.add(rs.getString("room\_type"));

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

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

if (rs.getString("status\_id").equals("1")){

v.add("Active");

} else {

v.add("Inactive");

}

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String code = dtm.getValueAt(selectRow, 0).toString();

String faci = dtm.getValueAt(selectRow, 1).toString();

String type = dtm.getValueAt(selectRow, 2).toString();

String cost = dtm.getValueAt(selectRow, 3).toString();

String qty = dtm.getValueAt(selectRow, 4).toString();

String stat = dtm.getValueAt(selectRow, 5).toString();

txt\_code.setText(code);

txt\_facilities.setText(faci);

txt\_type.setText(type);

txt\_cost.setText(cost);

txt\_qty.setText(qty);

if (stat.equals("Active")){

rdo\_a.setSelected(true);

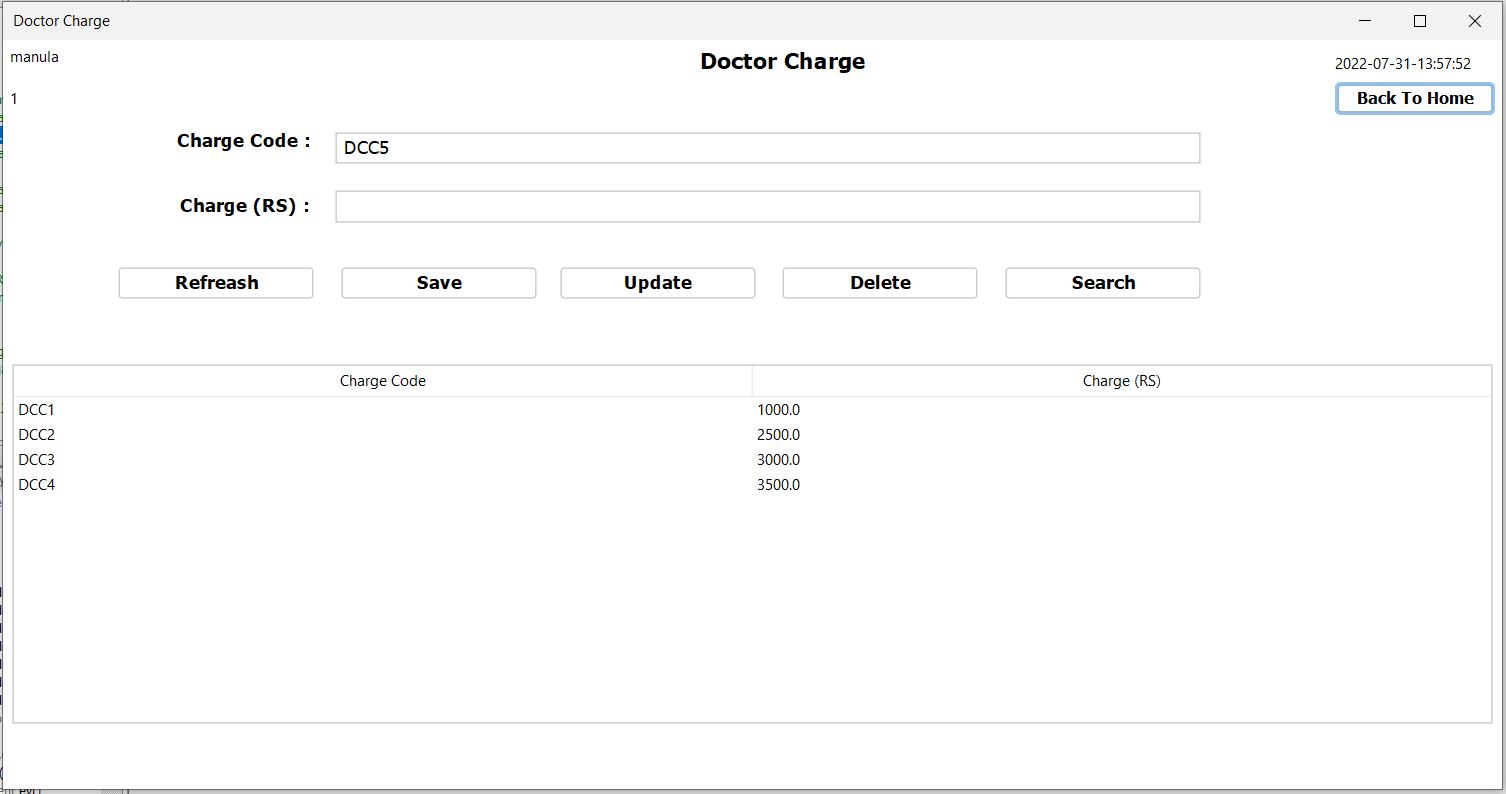
} else {

rdo\_i.setSelected(true);

}

}

Doctor charge



Constructors and other variables

public DoctorCharge() {

initComponents();

setLocationRelativeTo(null);

genarateCode ();

timeDate td = new timeDate();

td.timeAndDate(jLabel14);

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

loadAll();

}

Other methods

private void genarateCode() {

try {

String sql = "SELECT COUNT(\*) AS codeCount FROM doctor\_charge ";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("codeCount");

String dcc = "DCC" + (++count);

txt\_code.setText(dcc);

}

} catch (Exception e) {

e.printStackTrace();

}

}

private void loadAll() {

try {

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

String sql = "SELECT \* FROM doctor\_charge WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()){

Vector v = new Vector();

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

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

dtm.addRow(v);

}

} catch (Exception e) {

}

}

private void txt\_codeActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

txt\_charge.grabFocus();

}

Charge (RS) : text field

private void txt\_chargeKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String price = txt\_charge.getText();

String text = price + evt.getKeyChar();

if (!Pattern.compile("(0|0[.]0[.][0-9]\*)|[1-9]|[1-9][0-9]\*|[1-9][0-9]\*[.]|[1-9][0-9]\*[.][0-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Refreash button

private void jButton7ActionPerformed(java.awt.event.ActionEvent evt) {

DoctorCharge dc = new DoctorCharge();

dc.setVisible(true);

dispose();

}

Save button

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

try {

String code = txt\_code.getText();

String charge = txt\_charge.getText();

String searchSql = "SELECT \* FROM doctor\_charge WHERE code = '"+code+"'";

if (code.equals("")) {

JOptionPane.showMessageDialog(this, "Please Fill Code Fields","warning", JOptionPane.WARNING\_MESSAGE);

} else if (charge.equals("")) {

JOptionPane.showMessageDialog(this, "Please Fill Charge Fields","warning", JOptionPane.WARNING\_MESSAGE);

}

else {

ResultSet rs = db.search(searchSql);

if (!rs.next()){

String sql = "INSERT INTO doctor\_charge (`code`,`charge`) VALUES('"+code+"','"+charge+"')";

db.push(sql);

genarateCode();

txt\_charge.setText("");

loadAll();

JOptionPane.showMessageDialog(this, "Success!", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

genarateCode();

JOptionPane.showMessageDialog(this, "Code is already Defined","warning", JOptionPane.WARNING\_MESSAGE);

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

Update button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String code = txt\_code.getText();

String charge = txt\_charge.getText();

String searchSql = "SELECT \* FROM doctor\_charge WHERE code = '"+code+"'";

if (code.equals("")) {

JOptionPane.showMessageDialog(this, "Please Fill Code Fields","warning", JOptionPane.WARNING\_MESSAGE);

} else if (charge.equals("")) {

JOptionPane.showMessageDialog(this, "Please Fill Charge Fields","warning", JOptionPane.WARNING\_MESSAGE);

}

else {

ResultSet rs = db.search(searchSql);

if (rs.next()){

String sql = "UPDATE doctor\_charge SET charge = '"+charge+"' WHERE code = '"+code+"'";

db.push(sql);

loadAll();

JOptionPane.showMessageDialog(this, "Success!", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Code is Not Defined","warning", JOptionPane.WARNING\_MESSAGE);

}

}

} catch (Exception e) {

e.printStackTrace();

}

}

Delete button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String code = txt\_code.getText();

String searchSql = "SELECT \* FROM doctor\_charge WHERE code = '"+code+"'";

if (!code.equals("")) {

ResultSet rs = db.search(searchSql);

if (rs.next()){

String sql = "UPDATE doctor\_charge SET status\_id = '2' WHERE code = '"+code+"'";

db.push(sql);

loadAll();

JOptionPane.showMessageDialog(this, "Success!", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Code is Not Defined","warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Code Field","warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search button

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

try {

String code = txt\_code.getText();

String searchSql = "SELECT \* FROM doctor\_charge WHERE code = '"+code+"' && status\_id = '1'";

if (!code.equals("")) {

ResultSet rs = db.search(searchSql);

if (rs.next()){

txt\_code.setText(rs.getString("code"));

txt\_charge.setText(rs.getString("charge"));

} else {

JOptionPane.showMessageDialog(this, "Code is Not Defined Or Deleted");

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Code Field","warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String code = dtm.getValueAt(selectRow, 0).toString();

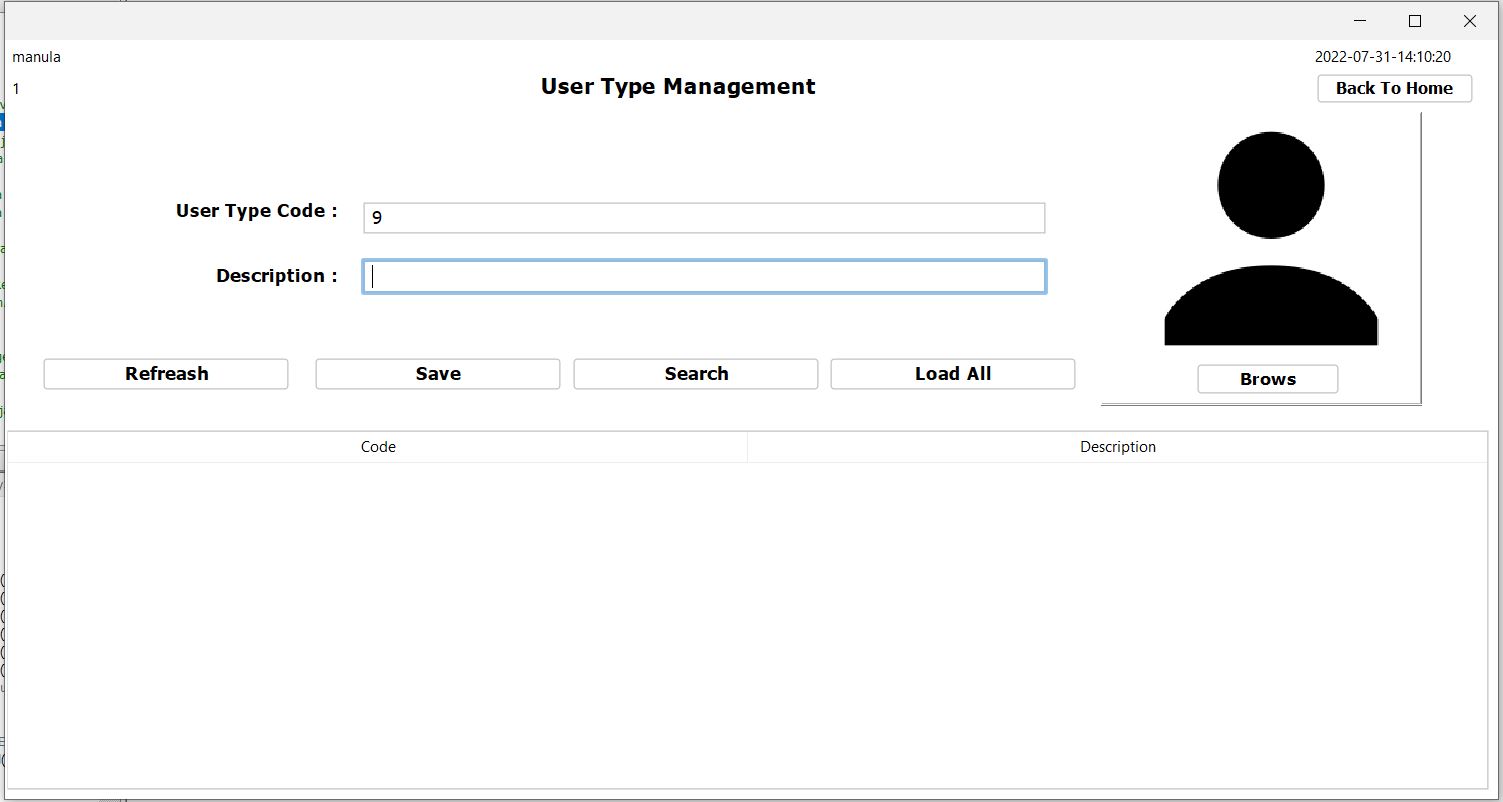
String charge = dtm.getValueAt(selectRow, 1).toString();

txt\_code.setText(code);

txt\_charge.setText(charge);

}

User Type Management



Constructors and other variables

String replacepath = null;

private static Date date;

public UserType() {

initComponents();

setLocationRelativeTo(null);

timeAndDate();

genarateUserCode();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

// loadDefImg();

}

Other methods

private static SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd-HH:mm:ss");

private void timeAndDate() {

Timer timer = new Timer(1000, new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

date = new Date();

jLabel1.setText(sdf.format(date));

}

});

timer.start();

}

private void genarateUserCode() {

try {

String sql = "SELECT COUNT(\*) AS code\_count FROM staff\_type";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("code\_count");

String utid = "" + (++count);

txt\_code.setText(utid);

}

txt\_description.grabFocus();

} catch (Exception e) {

e.printStackTrace();

}

}

static String filename;

private void loadDefImg() {

try {

JFileChooser jfc = new JFileChooser();

jfc.showOpenDialog(null);

File f = jfc.getSelectedFile();

ibl\_image.setIcon(new ImageIcon(f.toString()));

filename = f.getAbsolutePath();

// jLabel2.setText(filename);

String newPath = "....../com/hms/img";

InputStream pathStrem = UserType.class.getResourceAsStream(newPath);

File directory = new File(newPath);

if (!directory.exists()) {

directory.mkdirs();

}

File sourceFile = null;

File destinationFile = null;

String extension = filename.substring(filename.lastIndexOf(".") + 1);

sourceFile = new File(filename);

destinationFile = new File(newPath + "/newImage." + extension);

Files.copy(sourceFile.toPath(), destinationFile.toPath());

System.out.println(newPath);

} catch (Exception e) {

e.printStackTrace();

}

}

User Type Code : text field

private void txt\_codeKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String qty = txt\_code.getText();

String text = qty + evt.getKeyChar();

if (!Pattern.compile("[1-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Refreash button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

UserType ut = new UserType();

ut.setVisible(true);

dispose();

}

Save button

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

try {

String code = txt\_code.getText();

String description = txt\_description.getText();

String searchSql = "SELECT \* FROM staff\_type WHERE id = '" + code + "'";

ResultSet rs = db.search(searchSql);

if (!code.equals("") && !description.equals("")) {

if (replacepath != null && !replacepath.trim().equals("")) {

if (!rs.next()) {

String sql = "INSERT INTO staff\_type values ('" + code + "','" + description + "')";

db.push(sql);

txt\_description.setText(null);

db.push("insert into stufftypeimg (`id`,`path`) values('"+code+"','" + replacepath + "')");

JOptionPane.showMessageDialog(this, "Saved", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "User type code is already have. please check again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Browse a image");

}

} else {

JOptionPane.showMessageDialog(this, "Please fill all the fields", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search button

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

try {

String code = txt\_code.getText();

String searchSql = "SELECT \* FROM staff\_type WHERE id = '" + code + "'";

ResultSet rs = db.search(searchSql);

if (rs.next()) {

ResultSet rsImg = db.search("select \* from stufftypeimg where id = '" + rs.getString("id") + "'");

rsImg.next();

txt\_description.setText(rs.getString("name"));

// ibl\_image.setIcon(null);

replacepath = rsImg.getString("path");

File file = new File(replacepath);

Image img = ImageIO.read(file);

img = img.getScaledInstance(ibl\_image.getWidth(), ibl\_image.getHeight(), Image.SCALE\_SMOOTH);

ibl\_image.setIcon(new ImageIcon(img));

} else {

JOptionPane.showMessageDialog(this, "User Type Code is wrong . please check again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

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

try {

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

String sql = "SELECT \* FROM staff\_type";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Brows button

private void jButton6ActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try {

String newPath = "C:\\Users\\User\\Desktop\\HMS Final Project\\HMS\\Image";

//InputStream pathStrem = UserType.class.getResourceAsStream(newPath);

File directory = new File(newPath);

JFileChooser jf = new JFileChooser(directory);

int showOpenDialog = jf.showOpenDialog(this);

if (showOpenDialog == JFileChooser.APPROVE\_OPTION) {

File selectedFile = jf.getSelectedFile();

String absolutePath = selectedFile.getAbsolutePath();

replacepath = absolutePath.replace("\\", "/");

File file = new File(replacepath);

Image img = ImageIO.read(file);

img = img.getScaledInstance(ibl\_image.getWidth(), ibl\_image.getHeight(), Image.SCALE\_SMOOTH);

ibl\_image.setIcon(new ImageIcon(img));

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String code = dtm.getValueAt(selectRow, 0).toString();

String description = dtm.getValueAt(selectRow, 1).toString();

txt\_code.setText(code);

txt\_description.setText(description);

try {

ResultSet rsImg = db.search("select \* from stufftypeimg where id = '" + code + "'");

rsImg.next();

replacepath = rsImg.getString("path");

File file = new File(replacepath);

Image img = ImageIO.read(file);

img = img.getScaledInstance(ibl\_image.getWidth(), ibl\_image.getHeight(), Image.SCALE\_SMOOTH);

ibl\_image.setIcon(new ImageIcon(img));

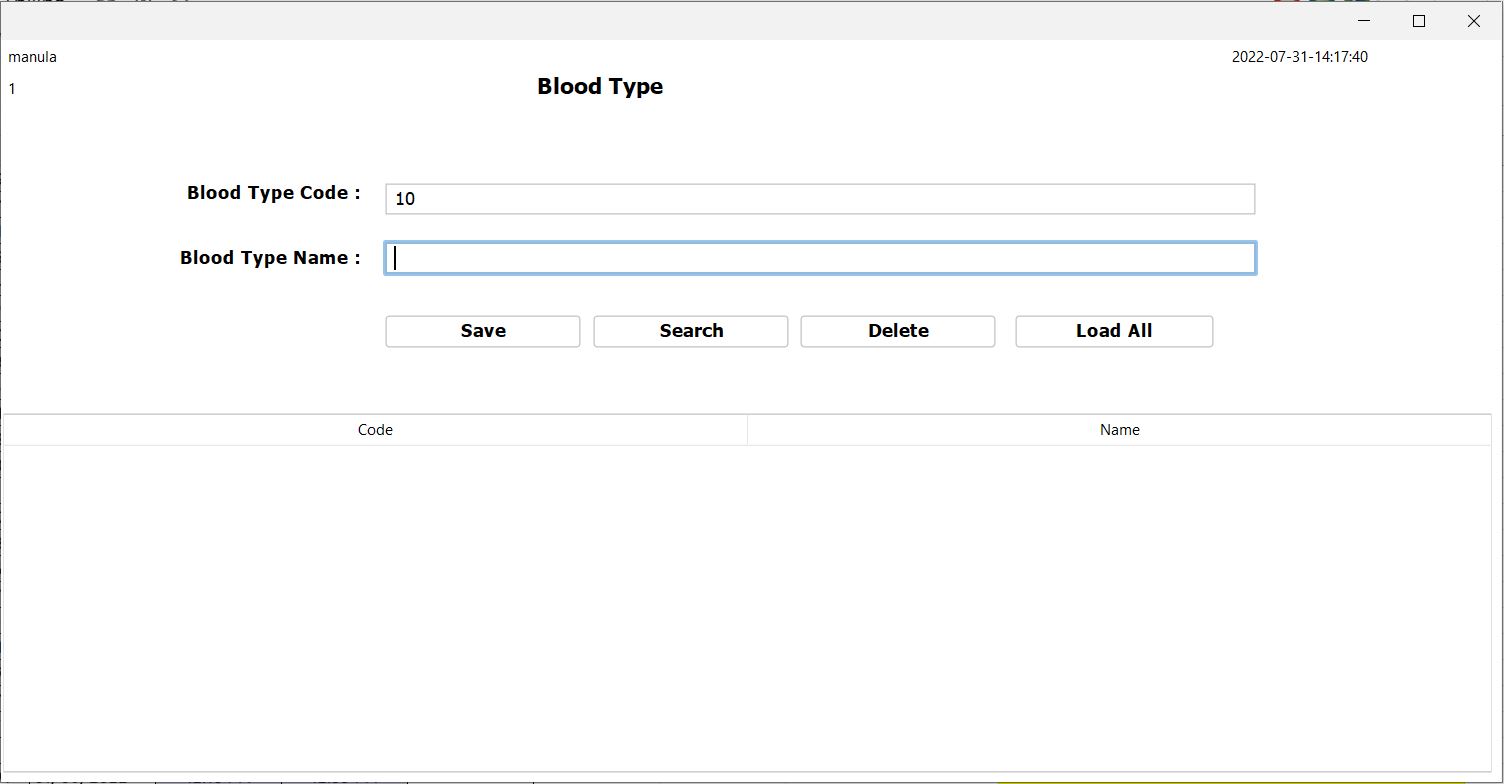
} catch (Exception e) {

e.printStackTrace();

}

}

Blood management



Constructors and other variables

public bloodRegistration() {

initComponents();

setLocationRelativeTo(null);

generateBloodCode();

timeAndDate();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

}

PatientRegistration p;

public bloodRegistration(PatientRegistration p) {

initComponents();

setLocationRelativeTo(null);

generateBloodCode();

timeAndDate();

jLabel8.setText(SystemConfigData.getActiveUser());

jLabel7.setText(SystemConfigData.getUserType());

this.p = p;

}

Other methods

private void generateBloodCode() {

try {

String sql = "SELECT COUNT(\*) AS blood\_Count FROM blood\_type";

ResultSet rs = db.search(sql);

if (rs.next()) {

int count = rs.getInt("blood\_Count");

String bid = "" + (++count);

txt\_code.setText(bid);

}

txt\_name.grabFocus();

} catch (Exception e) {

e.printStackTrace();

}

}

private static Date date;

private static SimpleDateFormat sdf = new SimpleDateFormat("yyyy-MM-dd-HH:mm:ss");

private void timeAndDate() {

Timer timer = new Timer(1000, new ActionListener() {

@Override

public void actionPerformed(ActionEvent e) {

date = new Date();

jLabel1.setText(sdf.format(date));

}

});

timer.start();

}

Blood Type Code : text field

private void txt\_codeKeyTyped(java.awt.event.KeyEvent evt) {

// TODO add your handling code here:

String qty = txt\_code.getText();

String text = qty + evt.getKeyChar();

if (!Pattern.compile("[1-9]\*").matcher(text).matches()) {

evt.consume();

}

}

Save button

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

try {

String code = txt\_code.getText();

String type = txt\_name.getText();

String searchSql = "SELECT \* FROM blood\_type WHERE id = '" + code + "'";

ResultSet rs = db.search(searchSql);

if (!code.equals("") && !type.equals("")) {

if (!rs.next()) {

String sql = "INSERT INTO blood\_type values('" + code + "','" + type + "','1')";

db.push(sql);

txt\_name.setText(null);

try {

p.loadBoodGroup();

} catch (NullPointerException e) {

}

generateBloodCode();

JOptionPane.showMessageDialog(this, "Saved", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Blood Type Code Is AllReady Have. please Check Again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill All The Fields", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Search button

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

try {

String code = txt\_code.getText();

String sql = "SELECT \* FROM blood\_type WHERE id = '" + code + "'";

ResultSet rs = db.search(sql);

if (rs.next()) {

txt\_name.setText(rs.getString("name"));

} else {

JOptionPane.showMessageDialog(this, "Blood Type Code Is Wrong. please Check Again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Delete button

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {

try {

String code = txt\_code.getText();

String searchSql = "SELECT \* FROM blood\_type WHERE id = '" + code + "' && status\_id='1'";

ResultSet rs = db.search(searchSql);

if (!code.equals("")) {

if (rs.next()) {

String sql = "UPDATE blood\_type SET status\_id = '2' WHERE id = '" + code + "'";

db.push(sql);

txt\_code.setText(null);

txt\_name.setText(null);

try {

p.loadBoodGroup();

} catch (NullPointerException e) {

}

JOptionPane.showMessageDialog(this, "Deleted", "success", JOptionPane.INFORMATION\_MESSAGE);

} else {

JOptionPane.showMessageDialog(this, "Blood Type Code Is Not Exist. please Check Again!", "warning", JOptionPane.WARNING\_MESSAGE);

}

} else {

JOptionPane.showMessageDialog(this, "Please Fill Blood Type Code Fields", "warning", JOptionPane.WARNING\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Load All button

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

try {

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

String sql = "SELECT \* FROM blood\_type WHERE status\_id = '1'";

ResultSet rs = db.search(sql);

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Table

private void jTable1MouseClicked(java.awt.event.MouseEvent evt) {

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

int selectRow = jTable1.getSelectedRow();

String code = dtm.getValueAt(selectRow, 0).toString();

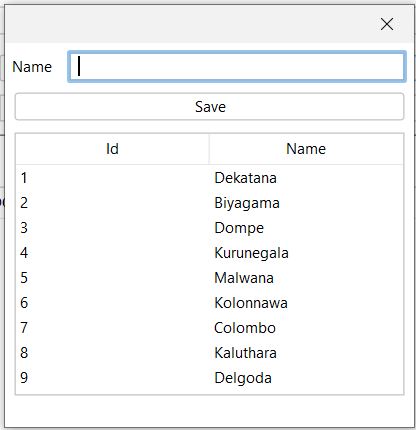
String name = dtm.getValueAt(selectRow, 1).toString();

txt\_code.setText(code);

txt\_name.setText(name);

}

City registration



Constructors and variables

public CityRegistration(java.awt.Frame parent, boolean modal,int x) {

super(parent, modal);

initComponents();

loadCity();

// try {

if(x==3){

this.x = (PatientRegistration) parent;

}

else if (x==2){

this.y = (stuffMangement) parent;

}

// } catch (Exception e) {

// }

}

PatientRegistration x;

stuffMangement y;

public CityRegistration(java.awt.Frame parent, boolean modal) {

super(parent, modal);

initComponents();

loadCity();

// try {

// } catch (Exception e) {

// }

}

Other methods

public void loadCity () {

try {

ResultSet rs = db.search("SELECT \* FROM `city`");

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

dtm.setRowCount(0);

while (rs.next()) {

Vector v = new Vector();

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

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

dtm.addRow(v);

}

} catch (Exception e) {

e.printStackTrace();

}

}

Save button

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

String name = jTextField1.getText();

if (name.isEmpty()){

JOptionPane.showMessageDialog(this, "Please enter city name", "Warning", JOptionPane.WARNING\_MESSAGE);

} else{

try {

ResultSet rs = db.search("SELECT \* FROM `city` WHERE `name` = '"+name+"'");

if (rs.next()) {

JOptionPane.showMessageDialog(this, "City already exists", "Warning", JOptionPane.WARNING\_MESSAGE);

} else {

db.push("INSERT INTO `city` (`name`) VALUES ('"+name+"')");

loadCity();

jTextField1.setText("");

jTextField1.grabFocus();

if(x != null){

x.loadCity();

}

else if (y != null) {

y.loadCity();

}

//

JOptionPane.showMessageDialog(this, "New city added", "Success", JOptionPane.INFORMATION\_MESSAGE);

}

} catch (Exception e) {

e.printStackTrace();

}

}

}