**JAVA SWING BASED – Insurance Management System - SQL**

**CONNECTIVITY USING JDBC**

*A*

*Report*

*Submitted in partial fulfillment of the*

*Requirements for the award of the Degree of*

**BACHELOR OF ENGINEERING IN**

**INFORMATION TECHNOLOGY**

**By**

**K. Sai Prasad <1602-19-737-095>**

**Under the guidance of Ms B. Leelavathy**



**Department of Information Technology**

**Vasavi College of Engineering (Autonomous)**

**(Affiliated to Osmania University) Ibrahimbagh, Hyderabad-31**

**2020-2021**

# BONAFIDE CERTIFICATE

This is to certify that this project report titled

***‘Insurance Management System’***

is a project work of  **K Sai Prasad** bearing

roll no. 1602-19-737-095 who carried out this

project under my supervision in the IV semester for the academic year 2020- 2021

|  |  |
| --- | --- |
| Signature | Signature |
| External Examiner | Internal Examiner |

**ABSTRACT**

The insurance management project deals the adding new insurance schemes and managing the clients for the insurance. The project has complete access for the crud operations that are to create, read, update and delete the database entries. At first you need to add a branch and the staff members for the branch then secondly add a user to the database now you can add an insurance scheme and finally make the payments for the client to the added insurance.

**REQUIREMENT ANALYSIS:**

**List of Tables:**

* Branch
* Employee
* Client
* Insurance
* Payment

**List of attributes with their domain types:**

**Branch:**

* branch\_code:varchar2(50)
* branch\_name:varchar2(50)
* branch\_address:varchar2(150)

**Employee:**

* employee\_id: number(5)
* employee \_name: varchar2(50)
* employee \_address: varchar2(150)
* employee \_contact: varchar2(15)

**Client:**

* client\_id: number(5)
* client\_name: varchar2(50)
* client\_contact: varchar2(15)
* client\_address: varchar2(150)

**Insurance:**

* insurance\_num: number(5)
* insurance\_type: varchar2(50)
* insurance\_amount: number(5)
* pay\_per\_month: number(5)
* start\_date: DATE
* end\_date: DATE

**Payment:**

* payment\_id
* client\_id: number(5)
* insurance\_num:number(5)
* amount:number(5)
* date\_of\_payment:DATE

**AIM AND PRIORITY OF THE PROJECT**

To create a Java GUI-based desktop application that connects people looking for insurance with employees looking for clients . It takes values like client name, employee name, branch name, etc through forms which are then updated in the database using JDBC connectivity.

**ARCHITECTURE AND TECHNOLOGY**

**Software used**: Java Eclipse, Oracle 11g Database, Java SE version 13, SQL\*Plus. **Java SWING**: Java SWING is a GUI widget toolkit for Java. It is part of Oracle's Java Foundation Classes (JFC) - an API for providing a graphical user interface (GUI) for Java programs.

Swing was developed to provide a more sophisticated set of GUI components than the earlier AWT. Swing provides a look and feel that emulates the look and feel of several platforms, and also supports a pluggable look and feel that allows applications to have a look and feel unrelated to the underlying platform. It has more powerful and flexible components than AWT. In addition to familiar components such as buttons, check boxes and labels, Swing provides several advanced components such as tabbed panel, scroll panes, trees, tables, and lists.

**SQL**: Structure Query Language(SQL) is a database query language used for storing and managing data in **Relational** DBMS. SQL was the first commercial language introduced for E.F Codd's Relational model of database. Today almost all RDBMS (MySql, Oracle, Infomix, Sybase, MS Access) use **SQL** as the standard database query language. SQL is used to perform all types of data operations in RDBMS.

**DESIGN**

**Entity relationship diagram**

**Diagram

Description automatically generated**

**DATABASE DESIGN**

**MAPPING CARDINALITY AND PARTICIPATION CONSTRAINTS**

branch-employee:

A branch have many employees but an employee can be in only one branch so, ONE TO MANY relationship

Employee-Insurance:

One employee can provide many insurances , so

ONE TO MANY relationship

Client-insurance :

A client can take many insurances and one insurance contains many clients so, MANY TO MANY relationship

Client-payment:

A client can have many payments and one payment is associated with one client so, ONE TO MANY

**DDL COMMANDS:**

**1.**create table branch(

branch\_code varchar2(5),

branch\_name varchar2(10) NOT NULL,

branch\_address varchar2(5) NOT NULL,

CONSTRAINT pk\_branch\_code PRIMARY KEY (branch\_code));

**2**.create table employee(

employee\_id number(5),

employee\_name varchar2(10) NOT NULL,

employee\_contact varchar2(10) NOT NULL,

employee\_address varchar2(5) NOT NULL,

branch\_code varchar2(5) NOT NULL,

CONSTRAINT fk\_employee\_bc FOREIGN KEY (branch\_code) REFERENCES branch(branch\_code) ON DELETE CASCADE,

CONSTRAINT pk\_emp\_id PRIMARY KEY (employee\_id)) ;

**3**. create table client(

client\_id number(5),

client\_name varchar2(10) NOT NULL,

client\_contact varchar2(10) NOT NULL,

client\_address varchar2(10) NOT NULL,

CONSTRAINT pk\_cl\_id PRIMARY KEY (client\_id));

**4**. create table insurance(

insurance\_num number(5),

insurance\_type varchar2(10) NOT NULL,

client\_id number(5) NOT NULL,

insurance\_amount number(7) NOT NULL,

pay\_per\_month number(5) NOT NULL,

start\_date DATE NOT NULL,

end\_date DATE NOT NULL,

branch\_code varchar2(10) NOT NULL,

employee\_id number(5) NOT NULL,

CONSTRAINT fk\_insurance\_ci FOREIGN KEY (client\_id) REFERENCES client(client\_id) ON DELETE CASCADE,

CONSTRAINT fk\_insurance\_bc FOREIGN KEY (branch\_code) REFERENCES branch(branch\_code) ON DELETE CASCADE,

CONSTRAINT fk\_insurance\_ei FOREIGN KEY (employee\_id) REFERENCES employee(employee\_id) ON DELETE CASCADE,

CONSTRAINT pk\_ insurance\_num PRIMARY KEY (insurance\_num));

**5**. create table payment(

Payment\_id(5) NOT NULL,

client\_id number(5) NOT NULL,

insurnace\_num number(5) NOT NULL,

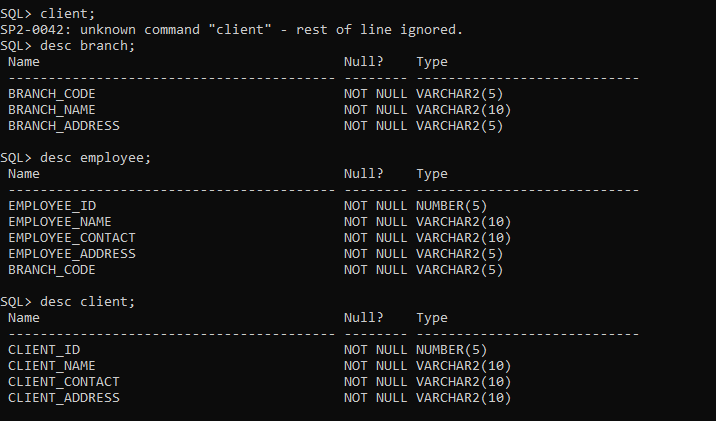
amount number(5) NOT NULL,

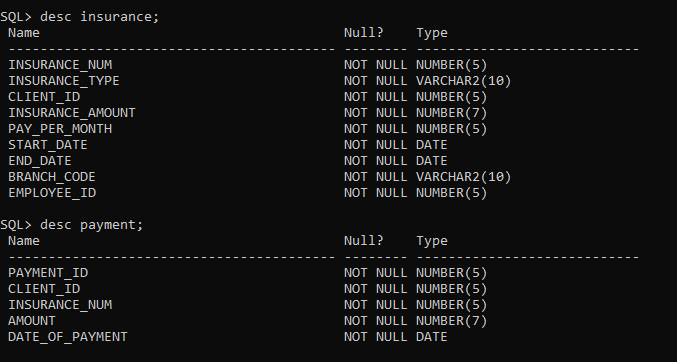
date\_of\_payment DATE NOT NULL,

CONSTRAINT fk\_payment\_ci FOREIGN KEY (client\_id) REFERENCES client(client\_id) ON DELETE CASCADE,

CONSTRAINT fk\_payment\_in FOREIGN KEY (insurance\_num) REFERENCES insurance(insurance\_num) ON DELETE CASCADE,

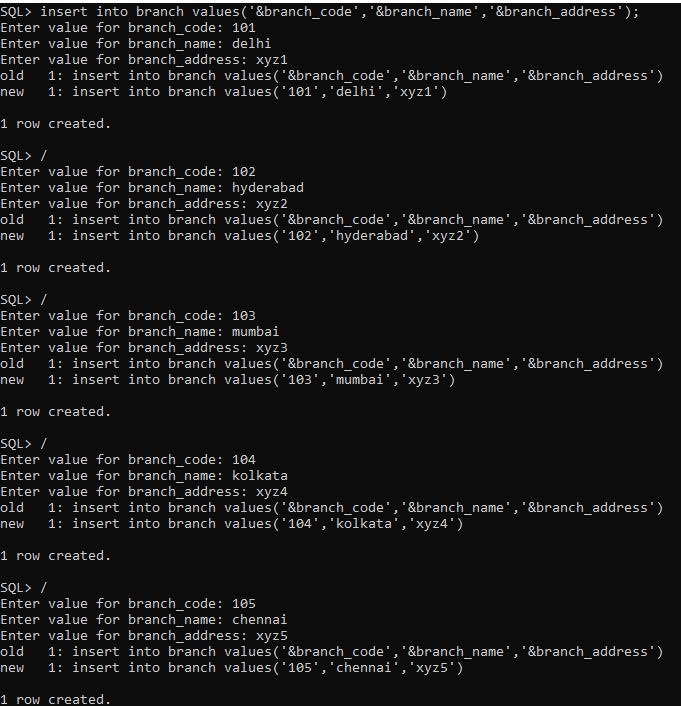
CONSTRAINT pk\_payment\_pk PRIMARY KEY (payment\_id));

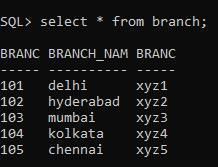




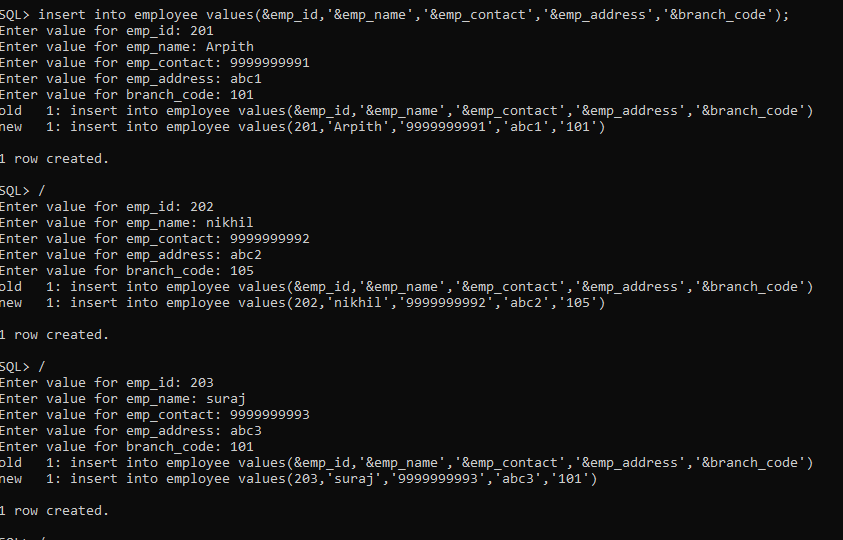
**DML COMMANDS**

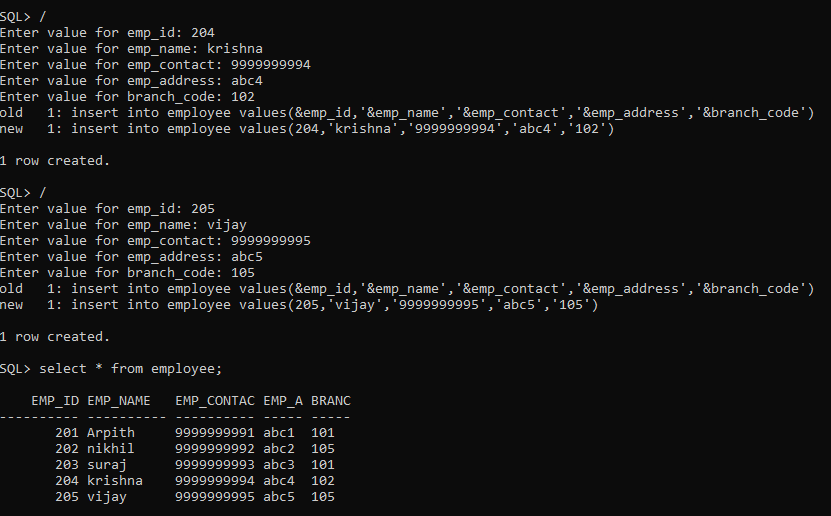
**1.**insert into branch values('&branch\_code','&branch\_name','&branch\_address');

****

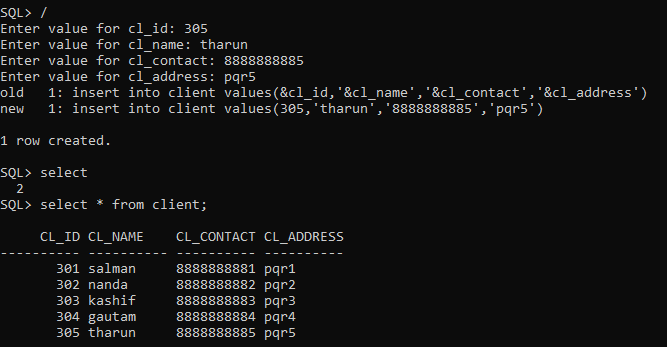
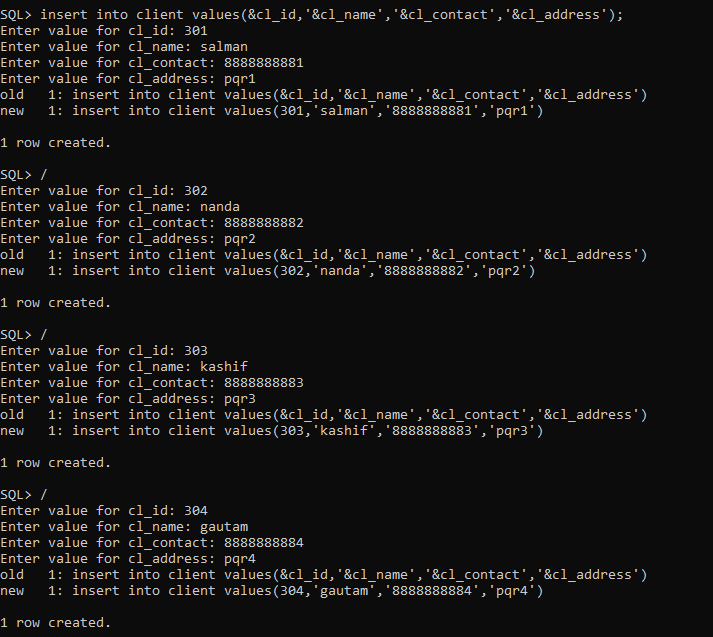
****

**2.**insert into employee values(&emp\_id,'&emp\_name','&emp\_contact','&emp\_address','&branch\_code');

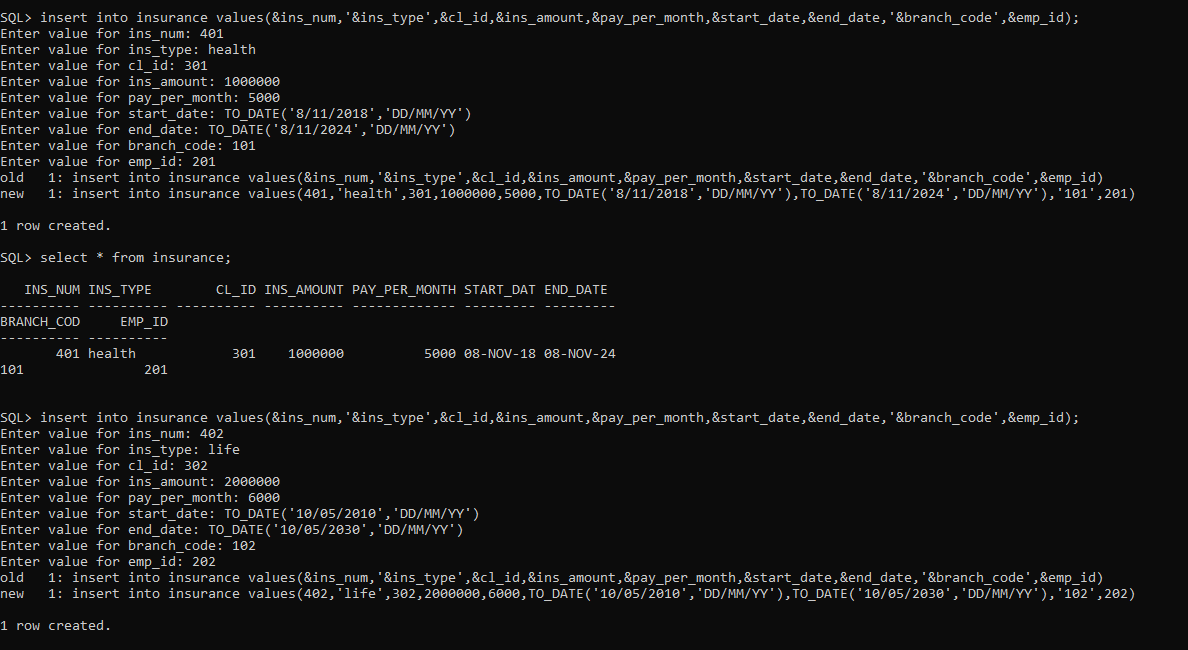
****

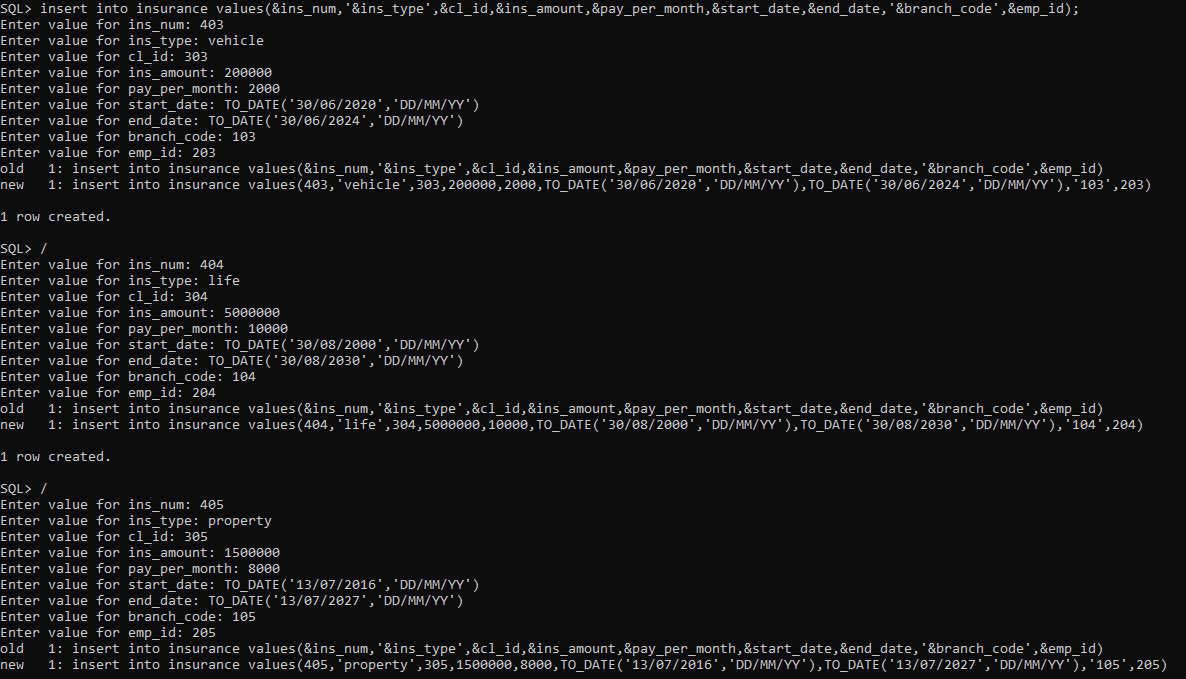
****

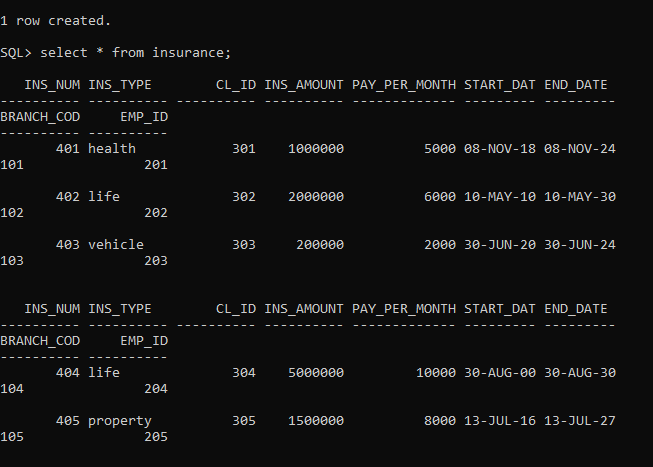
**3.** insert into client values(&cl\_id,'&cl\_name','&cl\_contact','&cl\_address');

****

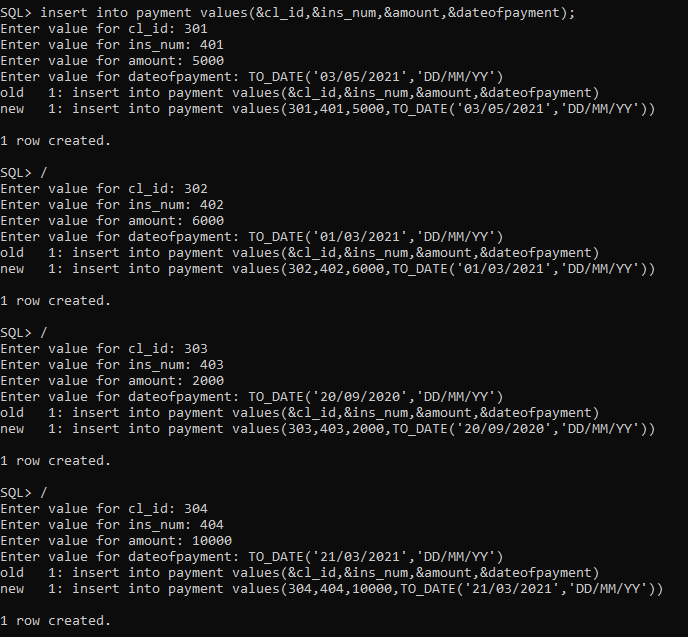
**4**. Insert into insurance values (&ins\_num,'&ins\_type',&cl\_id,&ins\_amount,&pay\_per\_month,&start\_date,&end\_date,'&branch\_code',&emp\_id);

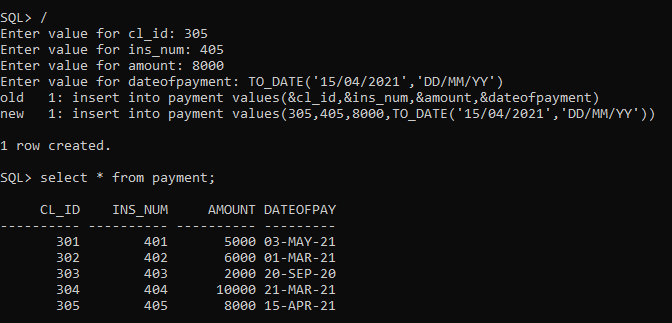
****

****

****

**5.**insert into payment values(&cl\_id,&ins\_num,&amount,&dateofpayment);

****

****

**IMPLEMENTATION**

**JAVA-SQL Connectivity using JDBC:**

**Java Database Connectivity (JDBC)** is an application programming interface (API) for the programming language Java, which defines how a client may access a database. It is a Java-based data access technology used for Java database connectivity. It is part of the Java Standard Edition platform, from Oracle Corporation. It provides methods to query and update data in a database and is oriented towards relational databases.

The connection to the database can be performed using Java programming (JDBC API) as:

**public** **void** connectDatabase()

{

**try**

{

Connection con=DriverManager.*getConnection*(

"jdbc:oracle:thin:@localhost:1521:xe","it19737095","vasavi");

stmt=con.createStatement();

stmt.executeUpdate("commit");

}

**catch** (SQLException connectException)

{

System.***out***.println(connectException.getMessage());

System.***out***.println(connectException.getSQLState());

System.***out***.println(connectException.getErrorCode());

System.*exit*(1);

}

}

Thus, the connection from Java to Oracle database is performed and therefore, can be used for

updating tables in the database directly.

**Branch table:**

//K.SAI PRASAD 1602-19-737-095 INSURANCE MANAGEMENT SYSTEM

**package** IMS;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** javax.swing.table.DefaultTableModel;

**import** java.sql.\*;

**import** java.util.Properties;

**public** **class** branch

{

**private** JPanel pn,pn1,pn2,pn3;

**private** JFrame jframe;

**private** JButton JB\_insert,JB\_modify,JB\_view,JB\_delete;

**private** JLabel JL\_branch\_code,JL\_branch\_name,JL\_branch\_address;

**private** JTextField JTF\_branch\_code,JTF\_branch\_name,JTF\_branch\_address;

Connection con;

ResultSet rs;

Statement stmt;

**private** JMenuItem insert1,update1,view1,delete1;

**private** List branchList;

**public** branch(JPanel pn,JFrame jframe,JMenuItem insert1,JMenuItem update1,JMenuItem view1,JMenuItem delete1)

{

**try**

{

Class.*forName*("oracle.jdbc.driver.OracleDriver");

}

**catch** (Exception e)

{

System.***err***.println("Unable to find and load driver");

System.*exit*(1);

}

connectDatabase();

**this**.jframe=jframe;

**this**.insert1=insert1;

**this**.update1=update1;

**this**.view1=view1;

**this**.delete1=delete1;

JL\_branch\_code=**new** JLabel("Branch\_Code:");

JTF\_branch\_code=**new** JTextField(10);

JL\_branch\_name=**new** JLabel("Branch Name:");

JTF\_branch\_name=**new** JTextField(10);

JL\_branch\_address=**new** JLabel("Branch Address:");

JTF\_branch\_address=**new** JTextField(10);

**this**.pn=pn;

}

**public** **void** connectDatabase()

{

**try**

{

Connection con=DriverManager.*getConnection*(

"jdbc:oracle:thin:@localhost:1521:xe","it19737095","vasavi");

stmt=con.createStatement();

stmt.executeUpdate("commit");

}

**catch** (SQLException connectException)

{

System.***out***.println(connectException.getMessage());

System.***out***.println(connectException.getSQLState());

System.***out***.println(connectException.getErrorCode());

System.*exit*(1);

}

}

**private** **void** displaySQLErrors(SQLException e)

{

JOptionPane.*showMessageDialog*(pn1,"\nSQLException: " + e.getMessage() + "\n"+"SQLState: " + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

}

**public** **void** loadbranch()

{

**try**

{

branchList=**new** List();

branchList.removeAll();

rs=stmt.executeQuery("select \* from branch");

**while**(rs.next())

{

branchList.add(rs.getString("branch\_code"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** buildGUI()

{

insert1.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_insert=**new** JButton("Submit");

JTF\_branch\_code.setText(**null**);

JTF\_branch\_name.setText(**null**);

JTF\_branch\_address.setText(**null**);

loadbranch();

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn1.add(JL\_branch\_name);

pn1.add(JTF\_branch\_name);

pn1.add(JL\_branch\_address);

pn1.add(JTF\_branch\_address);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_insert);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

branchList=**new** List(10);

loadbranch();

pn2.add(branchList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_insert.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

String query= "INSERT INTO branch VALUES(" + JTF\_branch\_code.getText() + ","

+ "'" +JTF\_branch\_name.getText() +"'," + "'"+JTF\_branch\_address.getText() +"')";

**int** i = stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nInserted "+i+" rows successfully");

loadbranch();

System.***out***.println("Done");

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

update1.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_modify=**new** JButton("Modify");

JTF\_branch\_code.setText(**null**);

JTF\_branch\_name.setText(**null**);

JTF\_branch\_address.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn1.add(JL\_branch\_name);

pn1.add(JTF\_branch\_name);

pn1.add(JL\_branch\_address);

pn1.add(JTF\_branch\_address);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_modify);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

branchList=**new** List(10);

loadbranch();

pn2.add(branchList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

branchList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from branch");

**while** (rs.next())

{

**if** (rs.getString("branch\_code").equals(branchList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_branch\_code.setText(rs.getString("branch\_code"));

JTF\_branch\_name.setText(rs.getString("branch\_name"));

JTF\_branch\_address.setText(rs.getString("branch\_address"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_modify.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to update:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

String pack=JOptionPane.*showInputDialog*(pn,"Enter New Branch Name:");

JTF\_branch\_name.setText(pack);

String query="update branch set branch\_name='"+pack+"' where branch\_code="+JTF\_branch\_code.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nUpdated "+i+" rows succesfully");

loadbranch();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

delete1.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_delete=**new** JButton("Delete");

JTF\_branch\_code.setText(**null**);

JTF\_branch\_name.setText(**null**);

JTF\_branch\_address.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn1.add(JL\_branch\_name);

pn1.add(JTF\_branch\_name);

pn1.add(JL\_branch\_address);

pn1.add(JTF\_branch\_address);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_delete);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

branchList=**new** List(10);

loadbranch();

pn2.add(branchList);

pn2.setBounds(200,350,300,200);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

branchList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from branch");

**while** (rs.next())

{

**if** (rs.getString("branch\_code").equals(branchList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_branch\_code.setText(rs.getString("branch\_code"));

JTF\_branch\_name.setText(rs.getString("branch\_name"));

JTF\_branch\_address.setText(rs.getString("branch\_address"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_delete.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to Delete:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

//String query="DELETE FROM branch WHERE branch\_code="+branchList.getSelectedItem();

String query="DELETE FROM branch WHERE branch\_code="+JTF\_branch\_code.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nDeleted "+i+" rows succesfully");

loadbranch();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

view1.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

JLabel view=**new** JLabel("Branch View");

JB\_view=**new** JButton("View");

Font myFont = **new** Font("Serif",Font.***BOLD***,50);

view.setFont((myFont));

pn1=**new** JPanel();

pn2=**new** JPanel();

pn1.add(view);

pn2.add(JB\_view);

pn.add(pn1);

pn.add(pn2);

pn.setBounds(500,800,300,300);

pn.setLayout(**new** FlowLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_view.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JFrame jf=**new** JFrame("Branch Details");

JTable jt;

DefaultTableModel model = **new** DefaultTableModel();

jt = **new** JTable(model);

model.addColumn("branch\_code");

model.addColumn("branch name");

model.addColumn("branch address");

**try**

{

rs=stmt.executeQuery("select \* from branch");

**while**(rs.next())

{

model.addRow(**new** Object[]{rs.getInt(1),

rs.getString(2),rs.getString(3)});

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

jt.setEnabled(**false**);

jt.setBounds(30, 40, 300, 300);

JScrollPane jsp = **new** JScrollPane(jt);

jf.add(jsp);

jf.setSize(800, 400);

jf.setVisible(**true**);

}

});

}

});

}

}

**Employee table:**

//K.SAI PRASAD 1602-19-737-095 INSURANCE MANAGEMENT SYSTEM

**package** IMS;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** javax.swing.table.DefaultTableModel;

**import** java.sql.\*;

**import** java.util.Properties;

**public** **class** employee

{

**private** JPanel pn,pn1,pn2,pn3;

**private** JFrame jframe;

**private** JButton JB\_insert,JB\_modify,JB\_view,JB\_delete;

**private** JLabel JL\_employee\_id,JL\_employee\_name,JL\_employee\_contact,JL\_employee\_address,JL\_branch\_code;

**private** JTextField JTF\_employee\_id,JTF\_employee\_name,JTF\_employee\_contact,JTF\_employee\_address,JTF\_branch\_code;

Connection con;

ResultSet rs;

Statement stmt;

**private** JMenuItem insert2,update2,view2,delete2;

**private** List employeeList;

**private** Choice branchCode;

**public** employee(JPanel pn,JFrame jframe,JMenuItem insert2,JMenuItem update2,JMenuItem view2,JMenuItem delete2)

{

**try**

{

Class.*forName*("oracle.jdbc.driver.OracleDriver");

}

**catch** (Exception e)

{

System.***err***.println("Unable to find and load driver");

System.*exit*(1);

}

connectDatabase();

**this**.jframe=jframe;

**this**.insert2=insert2;

**this**.update2=update2;

**this**.view2=view2;

**this**.delete2=delete2;

JL\_employee\_id=**new** JLabel("employee\_id:");

JTF\_employee\_id=**new** JTextField(10);

JL\_employee\_name=**new** JLabel("employee Name:");

JTF\_employee\_name=**new** JTextField(10);

JL\_employee\_contact=**new** JLabel("employee Contact:");

JTF\_employee\_contact=**new** JTextField(10);

JL\_employee\_address=**new** JLabel("employee Address:");

JTF\_employee\_address=**new** JTextField(10);

JL\_branch\_code=**new** JLabel("Branch Code:");

branchCode=**new** Choice();

JTF\_branch\_code=**new** JTextField(10);

**this**.pn=pn;

}

**public** **void** connectDatabase()

{

**try**

{

Connection con=DriverManager.*getConnection*(

"jdbc:oracle:thin:@localhost:1521:xe","it19737095","vasavi");

stmt=con.createStatement();

stmt.executeUpdate("commit");

}

**catch** (SQLException connectException)

{

System.***out***.println(connectException.getMessage());

System.***out***.println(connectException.getSQLState());

System.***out***.println(connectException.getErrorCode());

System.*exit*(1);

}

}

**private** **void** displaySQLErrors(SQLException e)

{

JOptionPane.*showMessageDialog*(pn1,"\nSQLException: " + e.getMessage() + "\n"+"SQLState: " + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

}

**public** **void** loadbranch()

{

**try**

{

//branchCode=new Choice();

branchCode.removeAll();

rs=stmt.executeQuery("select \* from branch");

**while**(rs.next())

{

branchCode.add(rs.getString("branch\_code"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** loademployee()

{

**try**

{

employeeList=**new** List();

employeeList.removeAll();

rs=stmt.executeQuery("select \* from employee");

**while**(rs.next())

{

employeeList.add(rs.getString("employee\_id"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** buildGUI()

{

insert2.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_insert=**new** JButton("Submit");

loadbranch();

JTF\_employee\_id.setText(**null**);

JTF\_employee\_name.setText(**null**);

JTF\_employee\_contact.setText(**null**);

JTF\_employee\_address.setText(**null**);

//JTF\_branch\_code.setText(null);

loademployee();

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_employee\_id);

pn1.add(JTF\_employee\_id);

pn1.add(JL\_employee\_name);

pn1.add(JTF\_employee\_name);

pn1.add(JL\_employee\_contact);

pn1.add(JTF\_employee\_contact);

pn1.add(JL\_employee\_address);

pn1.add(JTF\_employee\_address);

pn1.add(JL\_branch\_code);

pn1.add(branchCode);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_insert);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

employeeList=**new** List(10);

loademployee();

pn2.add(employeeList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_insert.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

String query= "INSERT INTO employee VALUES(" + JTF\_employee\_id.getText() + ","

+ "'" +JTF\_employee\_name.getText() +"'," +"'"+JTF\_employee\_contact.getText() +"'," + "'"+JTF\_employee\_address.getText() +"',"+"'"+branchCode.getSelectedItem()+"')";

**int** i = stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nInserted "+i+" rows successfully");

loademployee();

System.***out***.println("Done");

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

update2.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_modify=**new** JButton("Modify");

JTF\_employee\_id.setText(**null**);

JTF\_employee\_name.setText(**null**);

JTF\_employee\_contact.setText(**null**);

JTF\_employee\_address.setText(**null**);

JTF\_branch\_code.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_employee\_id);

pn1.add(JTF\_employee\_id);

pn1.add(JL\_employee\_name);

pn1.add(JTF\_employee\_name);

pn1.add(JL\_employee\_contact);

pn1.add(JTF\_employee\_contact);

pn1.add(JL\_employee\_address);

pn1.add(JTF\_employee\_address);

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_modify);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

employeeList=**new** List(10);

loademployee();

pn2.add(employeeList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

employeeList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from employee");

**while** (rs.next())

{

**if** (rs.getString("employee\_id").equals(employeeList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_employee\_id.setText(rs.getString("employee\_id"));

JTF\_employee\_name.setText(rs.getString("employee\_name"));

JTF\_employee\_contact.setText(rs.getString("employee\_contact"));

JTF\_employee\_address.setText(rs.getString("employee\_address"));

JTF\_branch\_code.setText(rs.getString("branch\_code"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_modify.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to update:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

String pack=JOptionPane.*showInputDialog*(pn,"Enter New employee Name:");

JTF\_employee\_name.setText(pack);

String query="update employee set employee\_name='"+pack+"' where employee\_id="+JTF\_employee\_id.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nUpdated "+i+" rows succesfully");

loademployee();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

delete2.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_delete=**new** JButton("Delete");

JTF\_employee\_id.setText(**null**);

JTF\_employee\_name.setText(**null**);

JTF\_employee\_contact.setText(**null**);

JTF\_employee\_address.setText(**null**);

JTF\_branch\_code.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_employee\_id);

pn1.add(JTF\_employee\_id);

pn1.add(JL\_employee\_name);

pn1.add(JTF\_employee\_name);

pn1.add(JL\_employee\_contact);

pn1.add(JTF\_employee\_contact);

pn1.add(JL\_employee\_address);

pn1.add(JTF\_employee\_address);

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_delete);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

employeeList=**new** List(10);

loademployee();

pn2.add(employeeList);

pn2.setBounds(200,350,300,200);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

employeeList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from employee");

**while** (rs.next())

{

**if** (rs.getString("employee\_id").equals(employeeList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_employee\_id.setText(rs.getString("employee\_id"));

JTF\_employee\_name.setText(rs.getString("employee\_name"));

JTF\_employee\_contact.setText(rs.getString("employee\_contact"));

JTF\_employee\_address.setText(rs.getString("employee\_address"));

JTF\_branch\_code.setText(rs.getString("branch\_code"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_delete.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to Delete:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

//String query="DELETE FROM employee WHERE employee\_id="+employeeList.getSelectedItem();

String query="DELETE FROM employee WHERE employee\_id="+JTF\_employee\_id.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nDeleted "+i+" rows succesfully");

loademployee();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

view2.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

JLabel view=**new** JLabel("employee View");

JB\_view=**new** JButton("View");

Font myFont = **new** Font("Serif",Font.***BOLD***,50);

view.setFont((myFont));

pn1=**new** JPanel();

pn2=**new** JPanel();

pn1.add(view);

pn2.add(JB\_view);

pn.add(pn1);

pn.add(pn2);

pn.setBounds(500,800,300,300);

pn.setLayout(**new** FlowLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_view.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JFrame jf=**new** JFrame("employee Details");

JTable jt;

DefaultTableModel model = **new** DefaultTableModel();

jt = **new** JTable(model);

model.addColumn("employee\_id");

model.addColumn("employee name");

model.addColumn("employee contact");

model.addColumn("employee address");

model.addColumn("branch Code");

**try**

{

rs=stmt.executeQuery("select \* from employee");

**while**(rs.next())

{

model.addRow(**new** Object[]{rs.getString("employee\_id"),

rs.getString("employee\_name"),rs.getString("employee\_contact"),rs.getString("employee\_address")

,rs.getString("branch\_code")});

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

jt.setEnabled(**false**);

jt.setBounds(30, 40, 300, 300);

JScrollPane jsp = **new** JScrollPane(jt);

jf.add(jsp);

jf.setSize(800, 400);

jf.setVisible(**true**);

}

});

}

});

}

}

**Client table:**

//K.SAI PRASAD 1602-19-737-095 INSURANCE MANAGEMENT SYSTEM

**package** IMS;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** javax.swing.table.DefaultTableModel;

**import** java.sql.\*;

**import** java.util.Properties;

**public** **class** client

{

**private** JPanel pn,pn1,pn2,pn3;

**private** JFrame jframe;

**private** JButton JB\_insert,JB\_modify,JB\_view,JB\_delete;

**private** JLabel JL\_client\_id,JL\_client\_name,JL\_client\_contact,JL\_client\_address;

**private** JTextField JTF\_client\_id,JTF\_client\_name,JTF\_client\_contact,JTF\_client\_address;

Connection con;

ResultSet rs;

Statement stmt;

**private** JMenuItem insert3,update3,view3,delete3;

**private** List clientList;

**public** client(JPanel pn,JFrame jframe,JMenuItem insert3,JMenuItem update3,JMenuItem view3,JMenuItem delete3)

{

**try**

{

Class.*forName*("oracle.jdbc.driver.OracleDriver");

}

**catch** (Exception e)

{

System.***err***.println("Unable to find and load driver");

System.*exit*(1);

}

connectDatabase();

**this**.jframe=jframe;

**this**.insert3=insert3;

**this**.update3=update3;

**this**.view3=view3;

**this**.delete3=delete3;

JL\_client\_id=**new** JLabel("client\_id:");

JTF\_client\_id=**new** JTextField(10);

JL\_client\_name=**new** JLabel("client Name:");

JTF\_client\_name=**new** JTextField(10);

JL\_client\_contact=**new** JLabel("Client Contact:");

JTF\_client\_contact=**new** JTextField(10);

JL\_client\_address=**new** JLabel("client Address:");

JTF\_client\_address=**new** JTextField(10);

**this**.pn=pn;

}

**public** **void** connectDatabase()

{

**try**

{

Connection con=DriverManager.*getConnection*(

"jdbc:oracle:thin:@localhost:1521:xe","it19737095","vasavi");

stmt=con.createStatement();

stmt.executeUpdate("commit");

}

**catch** (SQLException connectException)

{

System.***out***.println(connectException.getMessage());

System.***out***.println(connectException.getSQLState());

System.***out***.println(connectException.getErrorCode());

System.*exit*(1);

}

}

**private** **void** displaySQLErrors(SQLException e)

{

JOptionPane.*showMessageDialog*(pn1,"\nSQLException: " + e.getMessage() + "\n"+"SQLState: " + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

}

**public** **void** loadclient()

{

**try**

{

clientList=**new** List();

clientList.removeAll();

rs=stmt.executeQuery("select \* from client");

**while**(rs.next())

{

clientList.add(rs.getString("client\_id"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** buildGUI()

{

insert3.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_insert=**new** JButton("Submit");

JTF\_client\_id.setText(**null**);

JTF\_client\_name.setText(**null**);

JTF\_client\_contact.setText(**null**);

JTF\_client\_address.setText(**null**);

loadclient();

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_client\_name);

pn1.add(JTF\_client\_name);

pn1.add(JL\_client\_contact);

pn1.add(JTF\_client\_contact);

pn1.add(JL\_client\_address);

pn1.add(JTF\_client\_address);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_insert);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

clientList=**new** List(10);

loadclient();

pn2.add(clientList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_insert.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

String query= "INSERT INTO client VALUES(" + JTF\_client\_id.getText() + ","

+ "'" +JTF\_client\_name.getText() +"'," +"'"+JTF\_client\_contact.getText() +"'," + "'"+JTF\_client\_address.getText() +"')";

**int** i = stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nInserted "+i+" rows successfully");

loadclient();

System.***out***.println("Done");

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

update3.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_modify=**new** JButton("Modify");

JTF\_client\_id.setText(**null**);

JTF\_client\_name.setText(**null**);

JTF\_client\_contact.setText(**null**);

JTF\_client\_address.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_client\_name);

pn1.add(JTF\_client\_name);

pn1.add(JL\_client\_contact);

pn1.add(JTF\_client\_contact);

pn1.add(JL\_client\_address);

pn1.add(JTF\_client\_address);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_modify);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

clientList=**new** List(10);

loadclient();

pn2.add(clientList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

clientList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from client");

**while** (rs.next())

{

**if** (rs.getString("client\_id").equals(clientList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_client\_id.setText(rs.getString("client\_id"));

JTF\_client\_name.setText(rs.getString("client\_name"));

JTF\_client\_contact.setText(rs.getString("client\_contact"));

JTF\_client\_address.setText(rs.getString("client\_address"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_modify.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to update:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

String pack=JOptionPane.*showInputDialog*(pn,"Enter New client Name:");

JTF\_client\_name.setText(pack);

String query="update client set client\_name='"+pack+"' where client\_id="+JTF\_client\_id.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nUpdated "+i+" rows succesfully");

loadclient();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

delete3.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_delete=**new** JButton("Delete");

JTF\_client\_id.setText(**null**);

JTF\_client\_name.setText(**null**);

JTF\_client\_contact.setText(**null**);

JTF\_client\_address.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_client\_name);

pn1.add(JTF\_client\_name);

pn1.add(JL\_client\_contact);

pn1.add(JTF\_client\_contact);

pn1.add(JL\_client\_address);

pn1.add(JTF\_client\_address);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_delete);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

clientList=**new** List(10);

loadclient();

pn2.add(clientList);

pn2.setBounds(200,350,300,200);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

clientList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from client");

**while** (rs.next())

{

**if** (rs.getString("client\_id").equals(clientList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_client\_id.setText(rs.getString("client\_id"));

JTF\_client\_name.setText(rs.getString("client\_name"));

JTF\_client\_contact.setText(rs.getString("client\_contact"));

JTF\_client\_address.setText(rs.getString("client\_address"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_delete.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to Delete:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

//String query="DELETE FROM client WHERE client\_id="+clientList.getSelectedItem();

String query="DELETE FROM client WHERE client\_id="+JTF\_client\_id.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nDeleted "+i+" rows succesfully");

loadclient();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

view3.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

JLabel view=**new** JLabel("client View");

JB\_view=**new** JButton("View");

Font myFont = **new** Font("Serif",Font.***BOLD***,50);

view.setFont((myFont));

pn1=**new** JPanel();

pn2=**new** JPanel();

pn1.add(view);

pn2.add(JB\_view);

pn.add(pn1);

pn.add(pn2);

pn.setBounds(500,800,300,300);

pn.setLayout(**new** FlowLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_view.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JFrame jf=**new** JFrame("client Details");

JTable jt;

DefaultTableModel model = **new** DefaultTableModel();

jt = **new** JTable(model);

model.addColumn("client\_id");

model.addColumn("client name");

model.addColumn("client contact");

model.addColumn("client address");

**try**

{

rs=stmt.executeQuery("select \* from client");

**while**(rs.next())

{

model.addRow(**new** Object[]{rs.getInt(1),

rs.getString(2),rs.getString(3),rs.getString(4)});

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

jt.setEnabled(**false**);

jt.setBounds(30, 40, 300, 300);

JScrollPane jsp = **new** JScrollPane(jt);

jf.add(jsp);

jf.setSize(800, 400);

jf.setVisible(**true**);

}

});

}

});

}

}

**Insurance table:**

//K.SAI PRASAD 1602-19-737-095 INSURANCE MANAGEMENT SYSTEM

**package** IMS;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** javax.swing.table.DefaultTableModel;

**import** java.sql.\*;

**import** java.util.Properties;

**public** **class** insurance

{

**private** JPanel pn,pn1,pn2,pn3;

**private** JFrame jframe;

**private** JButton JB\_insert,JB\_modify,JB\_view,JB\_delete;

**private** JLabel JL\_insurance\_num,JL\_insurance\_type,JL\_client\_id,JL\_insurance\_amount,JL\_pay\_per\_month,JL\_start\_date,JL\_end\_date,JL\_branch\_code,JL\_employee\_id;

**private** JTextField JTF\_insurance\_num,JTF\_insurance\_type,JTF\_client\_id,JTF\_insurance\_amount,JTF\_pay\_per\_month,JTF\_start\_date,JTF\_end\_date,JTF\_branch\_code,JTF\_employee\_id;

Connection con;

ResultSet rs;

Statement stmt;

**private** JMenuItem insert4,update4,view4,delete4;

**private** List insuranceList;

**private** Choice clientId,branchCode,employeeId;

**public** insurance(JPanel pn,JFrame jframe,JMenuItem insert4,JMenuItem update4,JMenuItem view4,JMenuItem delete4)

{

**try**

{

Class.*forName*("oracle.jdbc.driver.OracleDriver");

}

**catch** (Exception e)

{

System.***err***.println("Unable to find and load driver");

System.*exit*(1);

}

connectDatabase();

**this**.jframe=jframe;

**this**.insert4=insert4;

**this**.update4=update4;

**this**.view4=view4;

**this**.delete4=delete4;

JL\_insurance\_num=**new** JLabel("insurance\_num:");

JTF\_insurance\_num=**new** JTextField(10);

JL\_insurance\_type=**new** JLabel("insurance Name:");

JTF\_insurance\_type=**new** JTextField(10);

JL\_client\_id=**new** JLabel("Client Id:");

clientId=**new** Choice();

JTF\_client\_id=**new** JTextField(10);

JL\_insurance\_amount=**new** JLabel("insurance Amount:");

JTF\_insurance\_amount=**new** JTextField(10);

JL\_pay\_per\_month=**new** JLabel("Pay Per Month:");

JTF\_pay\_per\_month=**new** JTextField(10);

JL\_start\_date=**new** JLabel("Start Date:");

JTF\_start\_date=**new** JTextField(10);

JL\_end\_date=**new** JLabel("End Date:");

JTF\_end\_date=**new** JTextField(10);

JL\_branch\_code=**new** JLabel("Branch Code:");

branchCode=**new** Choice();

JTF\_branch\_code=**new** JTextField(10);

JL\_employee\_id=**new** JLabel("Employee Id:");

employeeId=**new** Choice();

JTF\_employee\_id=**new** JTextField(10);

**this**.pn=pn;

}

**public** **void** connectDatabase()

{

**try**

{

Connection con=DriverManager.*getConnection*(

"jdbc:oracle:thin:@localhost:1521:xe","it19737095","vasavi");

stmt=con.createStatement();

stmt.executeUpdate("commit");

}

**catch** (SQLException connectException)

{

System.***out***.println(connectException.getMessage());

System.***out***.println(connectException.getSQLState());

System.***out***.println(connectException.getErrorCode());

System.*exit*(1);

}

}

**private** **void** displaySQLErrors(SQLException e)

{

JOptionPane.*showMessageDialog*(pn1,"\nSQLException: " + e.getMessage() + "\n"+"SQLState: " + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

}

**public** **void** loadclient()

{

**try**

{

//clientId=new Choice();

clientId.removeAll();

rs=stmt.executeQuery("select \* from client");

**while**(rs.next())

{

clientId.add(rs.getString("client\_id"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** loadbranch()

{

**try**

{

//branchCode=new Choice();

branchCode.removeAll();

rs=stmt.executeQuery("select \* from branch");

**while**(rs.next())

{

branchCode.add(rs.getString("branch\_code"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** loademployee()

{

**try**

{

//employeeId=new Choice();

employeeId.removeAll();

rs=stmt.executeQuery("select \* from employee");

**while**(rs.next())

{

employeeId.add(rs.getString("employee\_id"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** loadinsurance()

{

**try**

{

insuranceList=**new** List();

insuranceList.removeAll();

rs=stmt.executeQuery("select \* from insurance");

**while**(rs.next())

{

insuranceList.add(rs.getString("insurance\_num"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** buildGUI()

{

insert4.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_insert=**new** JButton("Submit");

loadclient();

loadbranch();

loademployee();

loadinsurance();

JTF\_insurance\_num.setText(**null**);

JTF\_insurance\_type.setText(**null**);

//JTF\_client\_id.setText(null);

JTF\_insurance\_amount.setText(**null**);

JTF\_pay\_per\_month.setText(**null**);

JTF\_start\_date.setText(**null**);

JTF\_end\_date.setText(**null**);

//JTF\_branch\_code.setText(null);

//JTF\_employee\_id.setText(null);

loadinsurance();

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_insurance\_num);

pn1.add(JTF\_insurance\_num);

pn1.add(JL\_insurance\_type);

pn1.add(JTF\_insurance\_type);

pn1.add(JL\_client\_id);

pn1.add(clientId);

pn1.add(JL\_insurance\_amount);

pn1.add(JTF\_insurance\_amount);

pn1.add(JL\_pay\_per\_month);

pn1.add(JTF\_pay\_per\_month);

pn1.add(JL\_start\_date);

pn1.add(JTF\_start\_date);

pn1.add(JL\_end\_date);

pn1.add(JTF\_end\_date);

pn1.add(JL\_branch\_code);

pn1.add(branchCode);

pn1.add(JL\_employee\_id);

pn1.add(employeeId);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_insert);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

insuranceList=**new** List(10);

loadinsurance();

pn2.add(insuranceList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_insert.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

String query= "INSERT INTO insurance VALUES(" + JTF\_insurance\_num.getText() + ","

+ "'" +JTF\_insurance\_type.getText() +"',"+"'"+clientId.getSelectedItem()+"',"+JTF\_insurance\_amount.getText()+","+JTF\_pay\_per\_month.getText()+","+JTF\_start\_date.getText()+","+JTF\_end\_date.getText()+","+"'"+branchCode.getSelectedItem()+"',"+"'"+employeeId.getSelectedItem()+"')";

**int** i = stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nInserted "+i+" rows successfully");

loadinsurance();

System.***out***.println("Done");

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

update4.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_modify=**new** JButton("Modify");

JTF\_insurance\_num.setText(**null**);

JTF\_insurance\_type.setText(**null**);

JTF\_client\_id.setText(**null**);

JTF\_insurance\_amount.setText(**null**);

JTF\_pay\_per\_month.setText(**null**);

JTF\_start\_date.setText(**null**);

JTF\_end\_date.setText(**null**);

JTF\_branch\_code.setText(**null**);

JTF\_employee\_id.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_insurance\_num);

pn1.add(JTF\_insurance\_num);

pn1.add(JL\_insurance\_type);

pn1.add(JTF\_insurance\_type);

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_insurance\_amount);

pn1.add(JTF\_insurance\_amount);

pn1.add(JL\_pay\_per\_month);

pn1.add(JTF\_pay\_per\_month);

pn1.add(JL\_start\_date);

pn1.add(JTF\_start\_date);

pn1.add(JL\_end\_date);

pn1.add(JTF\_end\_date);

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn1.add(JL\_employee\_id);

pn1.add(JTF\_employee\_id);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_modify);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

insuranceList=**new** List(10);

loadinsurance();

pn2.add(insuranceList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

insuranceList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from insurance");

**while** (rs.next())

{

**if** (rs.getString("insurance\_num").equals(insuranceList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_insurance\_num.setText(rs.getString("insurance\_num"));

JTF\_insurance\_type.setText(rs.getString("insurance\_type"));

JTF\_client\_id.setText(rs.getString("client\_id"));

JTF\_insurance\_amount.setText(rs.getString("insurance\_amount"));

JTF\_pay\_per\_month.setText(rs.getString("pay\_per\_month"));

JTF\_start\_date.setText(rs.getString("start\_date"));

JTF\_end\_date.setText(rs.getString("end\_date"));

JTF\_branch\_code.setText(rs.getString("branch\_code"));

JTF\_employee\_id.setText(rs.getString("employee\_id"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_modify.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to update:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

String pack=JOptionPane.*showInputDialog*(pn,"Enter New insurance Type:");

JTF\_insurance\_type.setText(pack);

String query="update insurance set insurance\_type='"+pack+"' where insurance\_num="+JTF\_insurance\_num.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nUpdated "+i+" rows succesfully");

loadinsurance();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

delete4.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_delete=**new** JButton("Delete");

JTF\_insurance\_num.setText(**null**);

JTF\_insurance\_type.setText(**null**);

JTF\_client\_id.setText(**null**);

JTF\_insurance\_amount.setText(**null**);

JTF\_pay\_per\_month.setText(**null**);

JTF\_start\_date.setText(**null**);

JTF\_end\_date.setText(**null**);

JTF\_branch\_code.setText(**null**);

JTF\_employee\_id.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_insurance\_num);

pn1.add(JTF\_insurance\_num);

pn1.add(JL\_insurance\_type);

pn1.add(JTF\_insurance\_type);

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_insurance\_amount);

pn1.add(JTF\_insurance\_amount);

pn1.add(JL\_pay\_per\_month);

pn1.add(JTF\_pay\_per\_month);

pn1.add(JL\_start\_date);

pn1.add(JTF\_start\_date);

pn1.add(JL\_end\_date);

pn1.add(JTF\_end\_date);

pn1.add(JL\_branch\_code);

pn1.add(JTF\_branch\_code);

pn1.add(JL\_employee\_id);

pn1.add(JTF\_employee\_id);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_delete);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

insuranceList=**new** List(10);

loadinsurance();

pn2.add(insuranceList);

pn2.setBounds(200,350,300,200);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

insuranceList.addItemListener(**new** ItemListener() {

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from insurance");

**while** (rs.next())

{

**if** (rs.getString("insurance\_num").equals(insuranceList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_insurance\_num.setText(rs.getString("insurance\_num"));

JTF\_insurance\_type.setText(rs.getString("insurance\_type"));

JTF\_client\_id.setText(rs.getString("client\_id"));

JTF\_insurance\_amount.setText(rs.getString("insurance\_amount"));

JTF\_pay\_per\_month.setText(rs.getString("pay\_per\_month"));

JTF\_start\_date.setText(rs.getString("start\_date"));

JTF\_end\_date.setText(rs.getString("end\_date"));

JTF\_branch\_code.setText(rs.getString("branch\_code"));

JTF\_employee\_id.setText(rs.getString("employee\_id"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_delete.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to Delete:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

//String query="DELETE FROM insurance WHERE insurance\_num="+insuranceList.getSelectedItem();

String query="DELETE FROM insurance WHERE insurance\_num="+JTF\_insurance\_num.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nDeleted "+i+" rows succesfully");

loadinsurance();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

view4.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

JLabel view=**new** JLabel("insurance View");

JB\_view=**new** JButton("View");

Font myFont = **new** Font("Serif",Font.***BOLD***,50);

view.setFont((myFont));

pn1=**new** JPanel();

pn2=**new** JPanel();

pn1.add(view);

pn2.add(JB\_view);

pn.add(pn1);

pn.add(pn2);

pn.setBounds(500,800,300,300);

pn.setLayout(**new** FlowLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_view.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JFrame jf=**new** JFrame("insurance Details");

JTable jt;

DefaultTableModel model = **new** DefaultTableModel();

jt = **new** JTable(model);

model.addColumn("insurance\_num");

model.addColumn("insurance type");

model.addColumn("Client Id");

model.addColumn("insurance amount");

model.addColumn("Pay per month");

model.addColumn("Start day");

model.addColumn("End day");

model.addColumn("branch Code");

model.addColumn("Employee Id");

**try**

{

rs=stmt.executeQuery("select \* from insurance");

**while**(rs.next())

{

model.addRow(**new** Object[]{rs.getString("insurance\_num"),

rs.getString("insurance\_type"),rs.getString("client\_id"),rs.getString("insurance\_amount"),rs.getString("pay\_per\_month")

,rs.getString("start\_date"),rs.getString("end\_date"),rs.getString("branch\_code"),rs.getString("employee\_id")});

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

jt.setEnabled(**false**);

jt.setBounds(30, 40, 300, 300);

JScrollPane jsp = **new** JScrollPane(jt);

jf.add(jsp);

jf.setSize(800, 400);

jf.setVisible(**true**);

}

});

}

});

}

}

**Payment table:**

//K.SAI PRASAD 1602-19-737-095 INSURANCE MANAGEMENT SYSTEM

**package** IMS;

**import** java.awt.\*;

**import** java.awt.event.\*;

**import** javax.swing.\*;

**import** javax.swing.table.DefaultTableModel;

**import** java.sql.\*;

**import** java.util.Properties;

**public** **class** payment

{

**private** JPanel pn,pn1,pn2,pn3;

**private** JFrame jframe;

**private** JButton JB\_insert,JB\_modify,JB\_view,JB\_delete;

**private** JLabel JL\_payment\_id,JL\_client\_id,JL\_insurance\_num,JL\_amount,JL\_date\_of\_payment;

**private** JTextField JTF\_payment\_id,JTF\_client\_id,JTF\_insurance\_num,JTF\_amount,JTF\_date\_of\_payment;

Connection con;

ResultSet rs;

Statement stmt;

**private** JMenuItem insert5,update5,view5,delete5;

**private** List paymentList;

**private** Choice clientId,insuranceNum;

**public** payment(JPanel pn,JFrame jframe,JMenuItem insert5,JMenuItem update5,JMenuItem view5,JMenuItem delete5)

{

**try**

{

Class.*forName*("oracle.jdbc.driver.OracleDriver");

}

**catch** (Exception e)

{

System.***err***.println("Unable to find and load driver");

System.*exit*(1);

}

connectDatabase();

**this**.jframe=jframe;

**this**.insert5=insert5;

**this**.update5=update5;

**this**.view5=view5;

**this**.delete5=delete5;

JL\_payment\_id=**new** JLabel("Payment\_Id:");

JTF\_payment\_id=**new** JTextField(10);

JL\_client\_id=**new** JLabel("Client Id:");

JTF\_client\_id=**new** JTextField(10);

JL\_insurance\_num=**new** JLabel("Insurance Num:");

JTF\_insurance\_num=**new** JTextField(10);

JL\_amount=**new** JLabel("amount:");

JTF\_amount=**new** JTextField(10);

JL\_date\_of\_payment=**new** JLabel("date\_of\_payment:");

JTF\_date\_of\_payment=**new** JTextField(10);

**this**.pn=pn;

}

**public** **void** connectDatabase()

{

**try**

{

Connection con=DriverManager.*getConnection*(

"jdbc:oracle:thin:@localhost:1521:xe","it19737095","vasavi");

stmt=con.createStatement();

stmt.executeUpdate("commit");

}

**catch** (SQLException connectException)

{

System.***out***.println(connectException.getMessage());

System.***out***.println(connectException.getSQLState());

System.***out***.println(connectException.getErrorCode());

System.*exit*(1);

}

}

**private** **void** displaySQLErrors(SQLException e)

{

JOptionPane.*showMessageDialog*(pn1,"\nSQLException: " + e.getMessage() + "\n"+"SQLState: " + e.getSQLState() + "\n"+"VendorError: " + e.getErrorCode() + "\n");

}

**public** **void** loadclient()

{

**try**

{

clientId=**new** Choice();

clientId.removeAll();

rs=stmt.executeQuery("select \* from client");

**while**(rs.next())

{

clientId.add(rs.getString("client\_id"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** loadinsurance()

{

**try**

{

insuranceNum=**new** Choice();

insuranceNum.removeAll();

rs=stmt.executeQuery("select \* from insurance");

**while**(rs.next())

{

insuranceNum.add(rs.getString("insurance\_num"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** loadpayment()

{

**try**

{

paymentList=**new** List();

paymentList.removeAll();

rs=stmt.executeQuery("select \* from payment");

**while**(rs.next())

{

paymentList.add(rs.getString("payment\_id"));

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

**public** **void** buildGUI()

{

insert5.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_insert=**new** JButton("Submit");

loadclient();

loadinsurance();

loadpayment();

JTF\_payment\_id.setText(**null**);

//JTF\_client\_id.setText(null);

//JTF\_insurance\_num.setText(null);

JTF\_amount.setText(**null**);

JTF\_date\_of\_payment.setText(**null**);

loadpayment();

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_payment\_id);

pn1.add(JTF\_payment\_id);

pn1.add(JL\_client\_id);

pn1.add(clientId);

pn1.add(JL\_insurance\_num);

pn1.add(insuranceNum);

pn1.add(JL\_amount);

pn1.add(JTF\_amount);

pn1.add(JL\_date\_of\_payment);

pn1.add(JTF\_date\_of\_payment);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_insert);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

paymentList=**new** List(10);

loadpayment();

pn2.add(paymentList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_insert.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

String query= "INSERT INTO payment VALUES(" + JTF\_payment\_id.getText() + ","

+"'"+clientId.getSelectedItem()+"',"+"'"+insuranceNum.getSelectedItem()+"',"+JTF\_amount.getText() +"," +JTF\_date\_of\_payment.getText() +")";

**int** i = stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nInserted "+i+" rows successfully");

loadpayment();

System.***out***.println("Done");

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

update5.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_modify=**new** JButton("Modify");

JTF\_payment\_id.setText(**null**);

JTF\_client\_id.setText(**null**);

JTF\_insurance\_num.setText(**null**);

JTF\_amount.setText(**null**);

JTF\_date\_of\_payment.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_payment\_id);

pn1.add(JTF\_payment\_id);

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_insurance\_num);

pn1.add(JTF\_insurance\_num);

pn1.add(JL\_amount);

pn1.add(JTF\_amount);

pn1.add(JL\_date\_of\_payment);

pn1.add(JTF\_date\_of\_payment);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_modify);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

paymentList=**new** List(10);

loadpayment();

pn2.add(paymentList);

pn2.setBounds(200,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

paymentList.addItemListener(**new** ItemListener()

{

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from payment");

**while** (rs.next())

{

**if** (rs.getString("payment\_Id").equals(paymentList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_payment\_id.setText(rs.getString("payment\_id"));

JTF\_client\_id.setText(rs.getString("client\_id"));

JTF\_insurance\_num.setText(rs.getString("insurance\_num"));

JTF\_amount.setText(rs.getString("amount"));

JTF\_date\_of\_payment.setText(rs.getString("date\_of\_payment"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_modify.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure do you want to update:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

String pack=JOptionPane.*showInputDialog*(pn,"Enter New amount:");

JTF\_amount.setText(pack);

String query="update payment set amount='"+pack+"' where payment\_id="+JTF\_payment\_id.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nUpdated "+i+" rows succesfully");

loadpayment();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

delete5.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

JB\_delete=**new** JButton("Delete");

JTF\_payment\_id.setText(**null**);

JTF\_client\_id.setText(**null**);

JTF\_insurance\_num.setText(**null**);

JTF\_amount.setText(**null**);

JTF\_date\_of\_payment.setText(**null**);

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

pn1=**new** JPanel();

pn1.setLayout(**new** GridLayout(10,10));

pn1.add(JL\_payment\_id);

pn1.add(JTF\_payment\_id);

pn1.add(JL\_client\_id);

pn1.add(JTF\_client\_id);

pn1.add(JL\_insurance\_num);

pn1.add(JTF\_insurance\_num);

pn1.add(JL\_amount);

pn1.add(JTF\_amount);

pn1.add(JL\_date\_of\_payment);

pn1.add(JTF\_date\_of\_payment);

pn3=**new** JPanel(**new** FlowLayout());

pn3.add(JB\_delete);

pn1.setBounds(115,80,300,250);

pn3.setBounds(200,350,75,35);

pn2=**new** JPanel(**new** FlowLayout());

paymentList=**new** List(10);

loadpayment();

pn2.add(paymentList);

pn2.setBounds(250,350,300,180);

pn.add(pn1);

pn.add(pn3);

pn.add(pn2);

pn.setLayout(**new** BorderLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

paymentList.addItemListener(**new** ItemListener()

{

**public** **void** itemStateChanged(ItemEvent ievt)

{

**try**

{

rs=stmt.executeQuery("select \* from payment");

**while**(rs.next())

{

**if** (rs.getString("payment\_id").equals(paymentList.getSelectedItem()))

**break**;

}

**if** (!rs.isAfterLast())

{

JTF\_payment\_id.setText(rs.getString("payment\_id"));

JTF\_client\_id.setText(rs.getString("client\_id"));

JTF\_insurance\_num.setText(rs.getString("insurance\_num"));

JTF\_amount.setText(rs.getString("amount"));

JTF\_date\_of\_payment.setText(rs.getString("date\_of\_payment"));

}

}

**catch** (SQLException selectException)

{

displaySQLErrors(selectException);

}

}

});

JB\_delete.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

**try**

{

**int** a=JOptionPane.*showConfirmDialog*(pn,"Are you sure want to Delete:");

**if**(a==JOptionPane.***YES\_OPTION***)

{

//String query="DELETE FROM employee WHERE employee\_id="+employeeList.getSelectedItem();

String query="DELETE FROM payment WHERE payment\_id="+JTF\_payment\_id.getText();

**int** i=stmt.executeUpdate(query);

JOptionPane.*showMessageDialog*(pn,"\nDeleted "+i+" rows succesfully");

loadpayment();

}

}

**catch**(SQLException e)

{

displaySQLErrors(e);

}

}

});

}

});

view5.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent aevt)

{

pn.removeAll();

jframe.invalidate();

jframe.validate();

jframe.repaint();

JLabel view=**new** JLabel("payment View");

JB\_view=**new** JButton("View");

Font myFont = **new** Font("Serif",Font.***BOLD***,50);

view.setFont((myFont));

pn1=**new** JPanel();

pn2=**new** JPanel();

pn1.add(view);

pn2.add(JB\_view);

pn.add(pn1);

pn.add(pn2);

pn.setLayout(**new** FlowLayout());

jframe.add(pn);

jframe.setSize(800,800);

jframe.validate();

JB\_view.addActionListener(**new** ActionListener() {

@Override

**public** **void** actionPerformed(ActionEvent e)

{

JFrame jf=**new** JFrame("payment Details");

JTable jt;

DefaultTableModel model = **new** DefaultTableModel();

jt= **new** JTable(model);

model.addColumn("Payment Id");

model.addColumn("Client Id");

model.addColumn("insurance num");

model.addColumn("amount");

model.addColumn("date\_of\_payment");

**try**

{

rs=stmt.executeQuery("select \* from payment");

**while**(rs.next())

{

model.addRow(**new** Object[]{rs.getString("payment\_id"),rs.getString("client\_id"),rs.getString("insurance\_num"),

rs.getString("amount"),rs.getString("date\_of\_payment")});

}

}

**catch**(SQLException exp)

{

displaySQLErrors(exp);

}

jt.setEnabled(**false**);

jt.setBounds(30, 40, 180, 150);

JScrollPane sp = **new** JScrollPane(jt);

jf.add(sp);

jf.setSize(800, 400);

jf.setVisible(**true**);

}

});

}

});

}

}

**Main**

**package** IMS;

**import** java.awt.\*;

**import** javax.swing.\*;

**public** **class** IMSUI **extends** JFrame

{

**private** JPanel Jpn0,Jpn1;

**private** JMenuBar mbar;

**private** JMenu branch,employee,client,insurance,payment;

**private** JMenuItem insert1,update1,view1,delete1;

**private** JMenuItem insert2,update2,view2,delete2;

**private** JMenuItem insert3,update3,view3,delete3;

**private** JMenuItem insert4,update4,view4,delete4;

**private** JMenuItem insert5,update5,view5,delete5;

/\*private JMenuItem insert6,update6,view6,delete6;

private JMenuItem s1,s2,s3,s4,s5;\*/

**private** JLabel labelName;

**public** **void** defineComponents()

{

Jpn0=**new** JPanel();

Jpn1=**new** JPanel();

mbar=**new** JMenuBar();

//statesAvailable=new JMenu("List Of States Available");//d\_1

branch=**new** JMenu("Branch");

employee=**new** JMenu("Employee");

client=**new** JMenu("Client");

insurance=**new** JMenu("Insurance");

payment=**new** JMenu("Payment");

labelName=**new** JLabel("Insurance Management System");

}

**public** **void** addComponents()

{

add(Jpn0);

Jpn1.add(labelName);

Jpn1.setAlignmentY(***CENTER\_ALIGNMENT***);

Jpn1.setBounds(500,500,800,100);

Jpn0.add(Jpn1);

setJMenuBar(mbar);

mbar.add(branch);

mbar.add(employee);

mbar.add(client);

mbar.add(insurance);

mbar.add(payment);

/\*statesAvailable.add(s1=new JMenuItem("Goa"));

statesAvailable.add(s2=new JMenuItem("Kerala"));

statesAvailable.add(s3=new JMenuItem("Telangana"));

statesAvailable.add(s4=new JMenuItem("Maharashtra"));

statesAvailable.add(s5=new JMenuItem("Uttar Pradesh"));\*/

branch.add(insert1=**new** JMenuItem("Insert"));

branch.add(update1=**new** JMenuItem("Update"));

branch.add(view1=**new** JMenuItem("View"));

branch.add(delete1=**new** JMenuItem("Delete"));

employee.add(insert2=**new** JMenuItem("Insert"));

employee.add(update2=**new** JMenuItem("Update"));

employee.add(view2=**new** JMenuItem("View"));

employee.add(delete2=**new** JMenuItem("Delete"));

client.add(insert3=**new** JMenuItem("Insert"));

client.add(update3=**new** JMenuItem("Update"));

client.add(view3=**new** JMenuItem("View"));

client.add(delete3=**new** JMenuItem("Delete"));

insurance.add(insert4=**new** JMenuItem("Insert"));

insurance.add(update4=**new** JMenuItem("Update"));

insurance.add(view4=**new** JMenuItem("View"));

insurance.add(delete4=**new** JMenuItem("Delete"));

payment.add(insert5=**new** JMenuItem("Insert"));

payment.add(update5=**new** JMenuItem("Update"));

payment.add(view5=**new** JMenuItem("View"));

payment.add(delete5=**new** JMenuItem("Delete"));

}

**public** **void** registerComponents()

{

branch obj\_branch=**new** branch(Jpn0,IMSUI.**this**,insert1,update1,view1,delete1);

obj\_branch.buildGUI();

employee obj\_employee=**new** employee(Jpn0,IMSUI.**this**,insert2,update2,view2,delete2);

obj\_employee.buildGUI();

client obj\_client=**new** client(Jpn0,IMSUI.**this**,insert3,update3,view3,delete3);

obj\_client.buildGUI();

insurance obj\_insurance=**new** insurance(Jpn0,IMSUI.**this**,insert4,update4,view4,delete4);

obj\_insurance.buildGUI();

payment obj\_payment=**new** payment(Jpn0,IMSUI.**this**,insert5,update5,view5,delete5);

obj\_payment.buildGUI();

}

**public** IMSUI()

{

defineComponents();

addComponents();

registerComponents();

setSize(400,500);

setBackground(Color.***GRAY***);

setVisible(**true**);

setTitle("INSURANCE MANAGEMENT SYSTEM");

setDefaultCloseOperation(JFrame.***EXIT\_ON\_CLOSE***);

}

}

**package** IMS;

**public** **class** Main

{

**public** **static** **void** main(String []args)

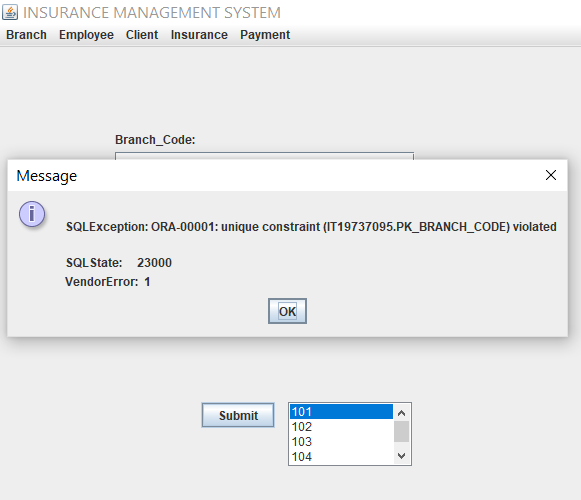
{

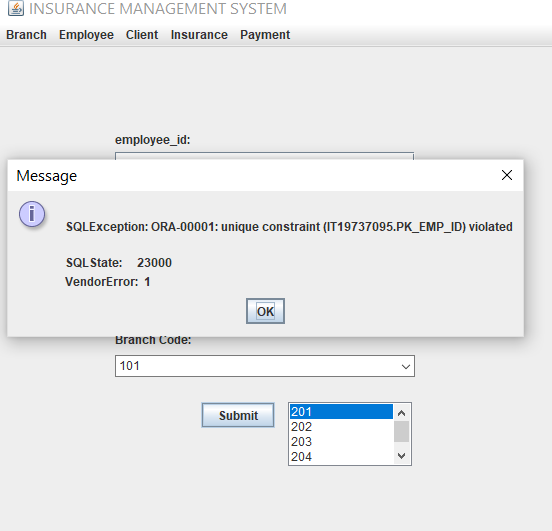
**new** IMSUI();

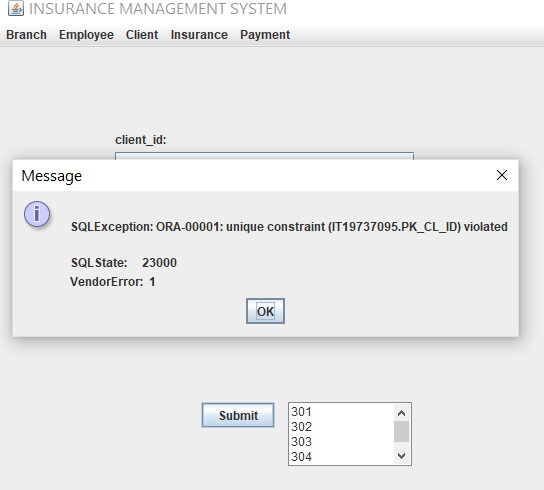
}

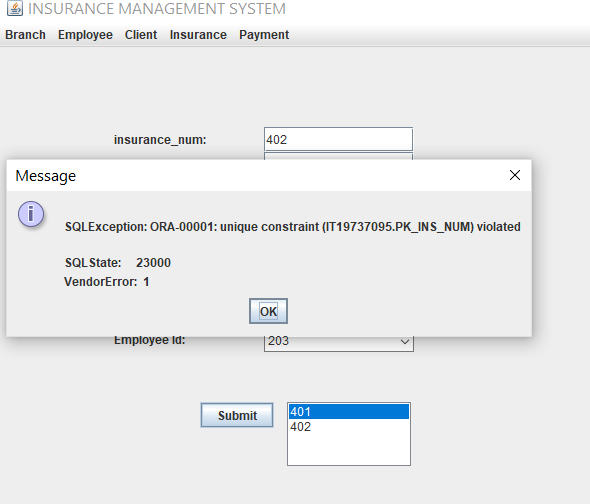
}

**TESTING**



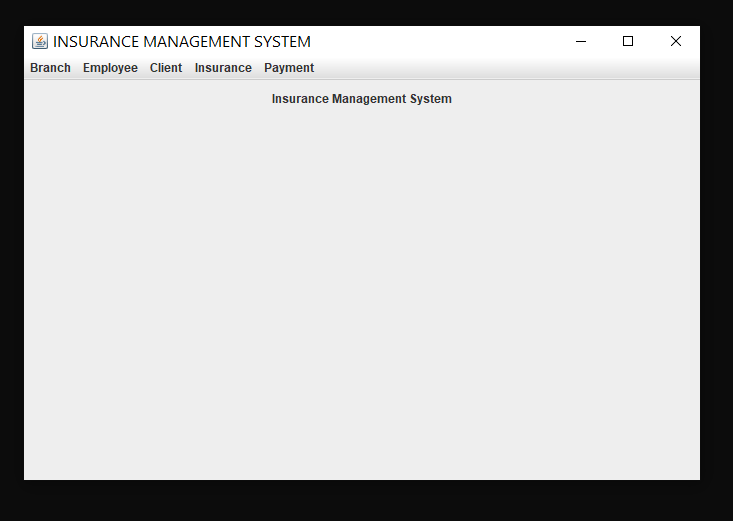






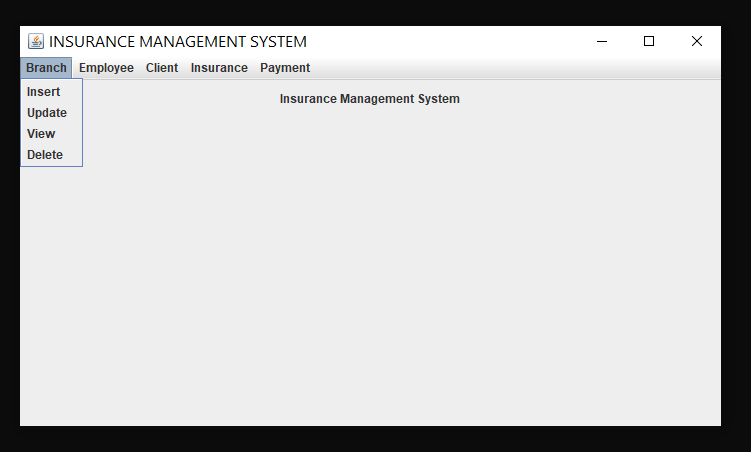
**RESULTS**

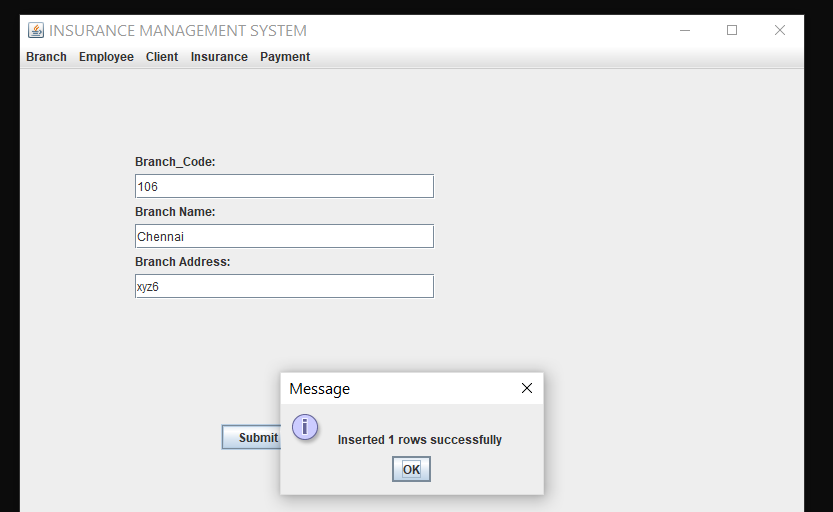
**MAIN:**

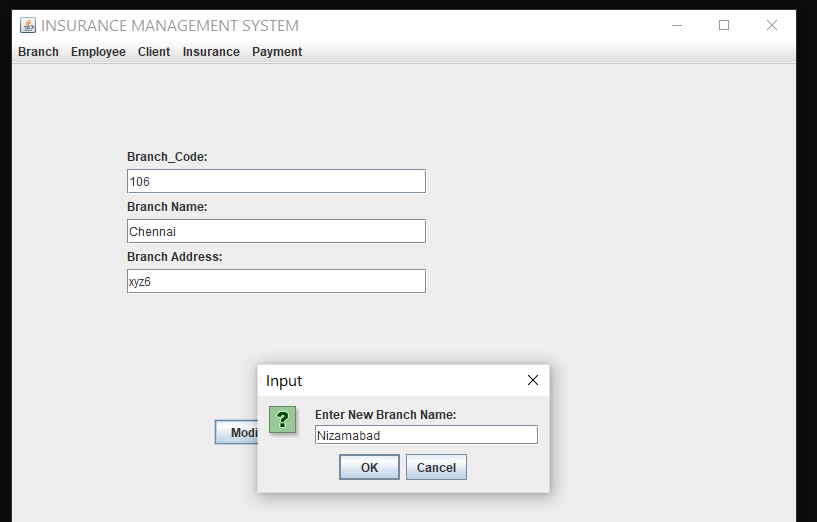


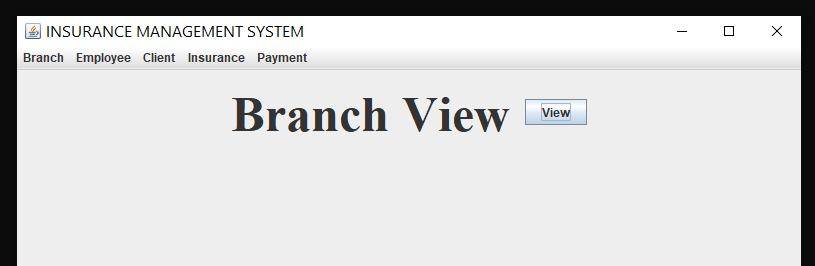
**BRANCH TABLE:**

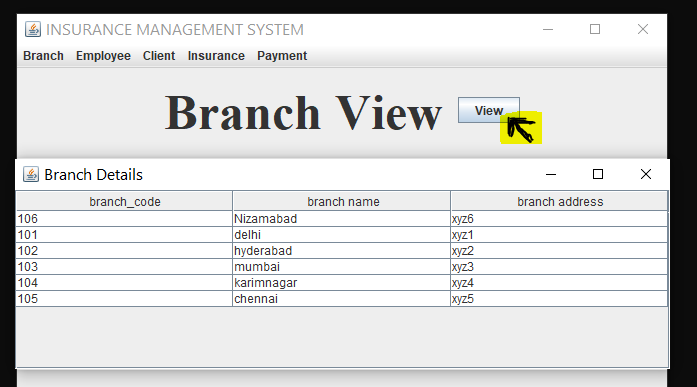
The branch\_code is an unique value and it can only have a numeric value and not anything else, similarly the branch\_name can only be a string. In case of any unexpected or incorrect input it throws an exception.

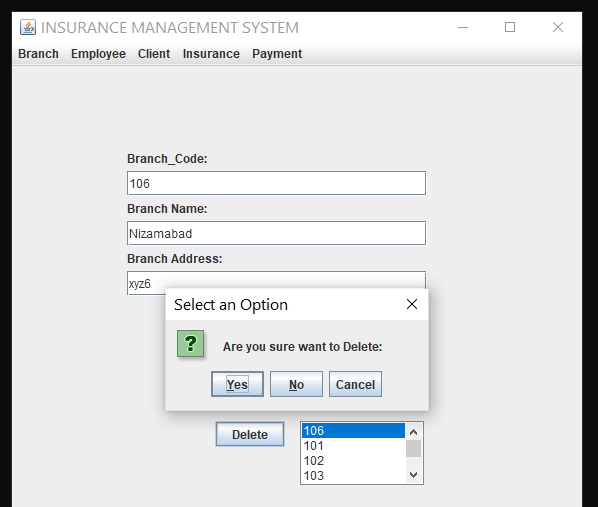


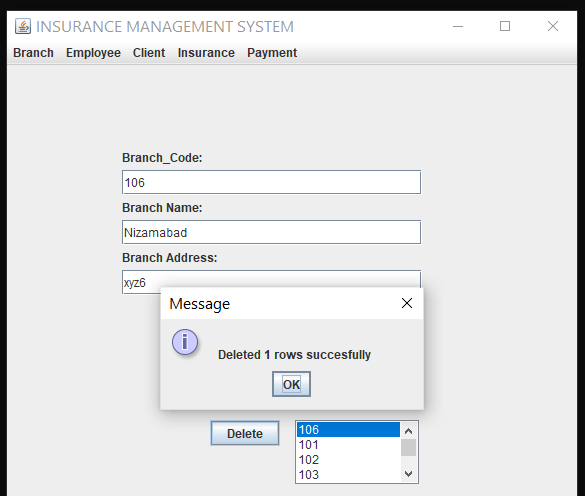






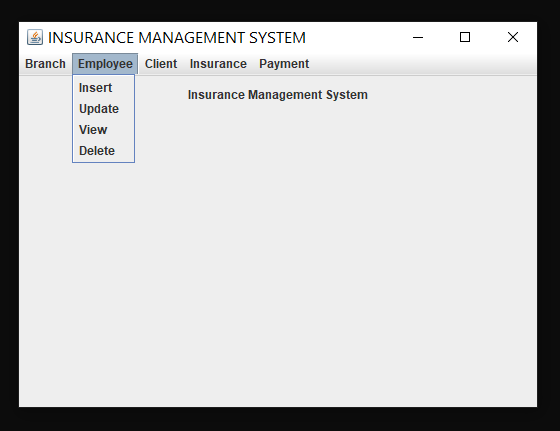


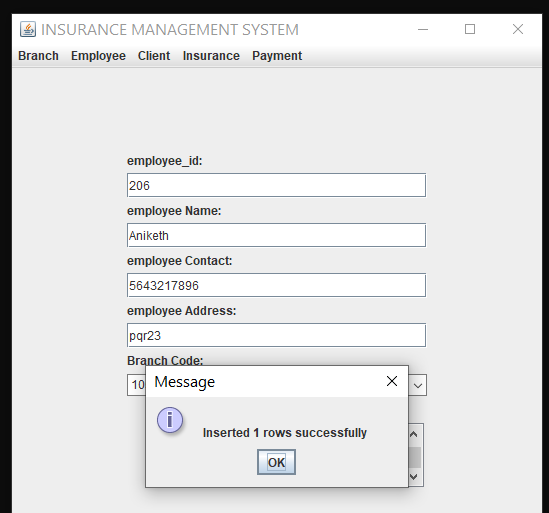
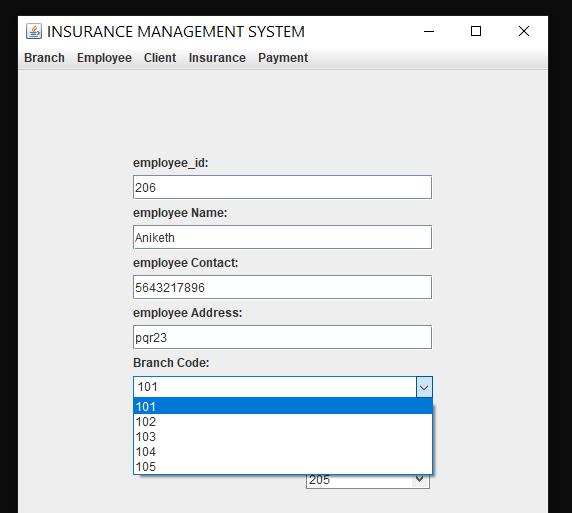


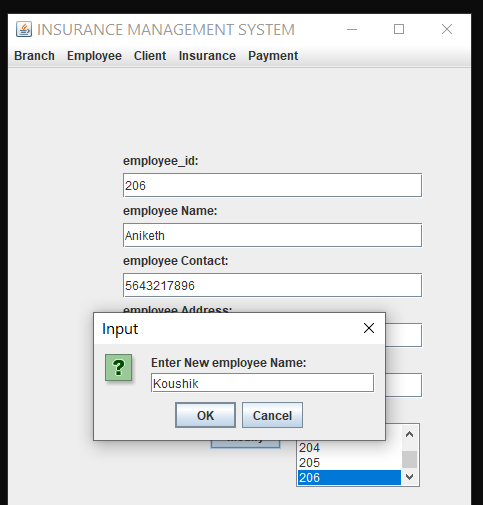
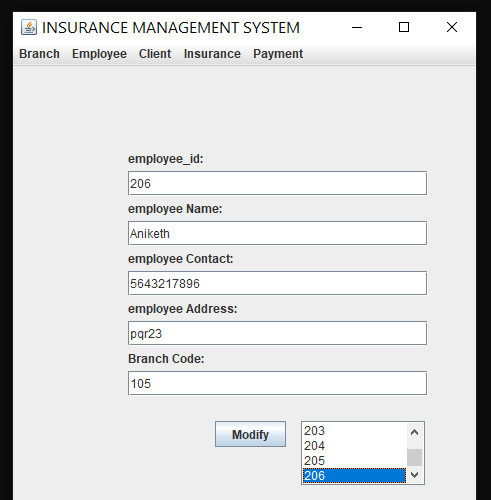


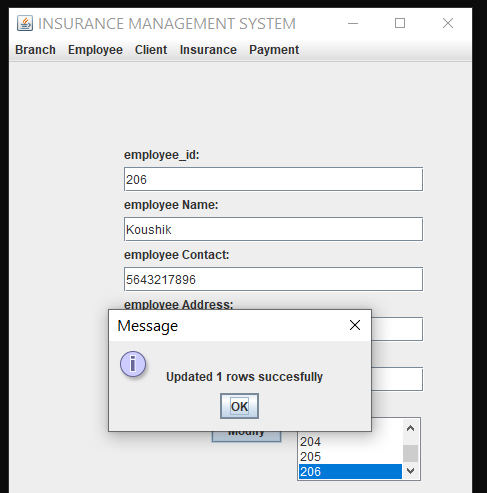
**EMPLOYEE TABLE:**

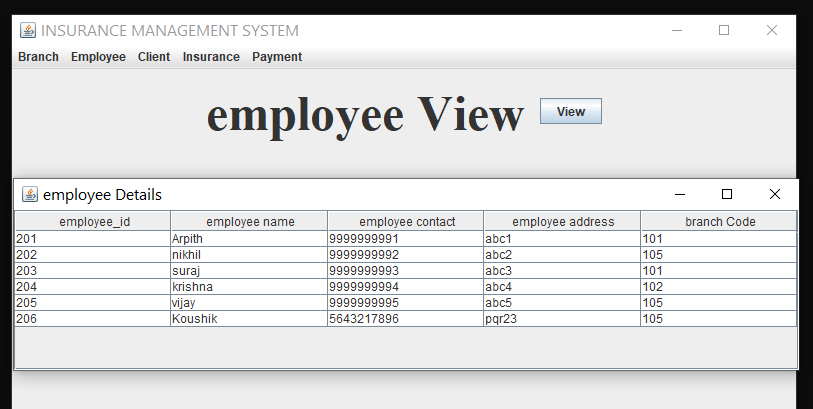
Employee\_Id is the primary and unique key. branch\_code is the foreign key.

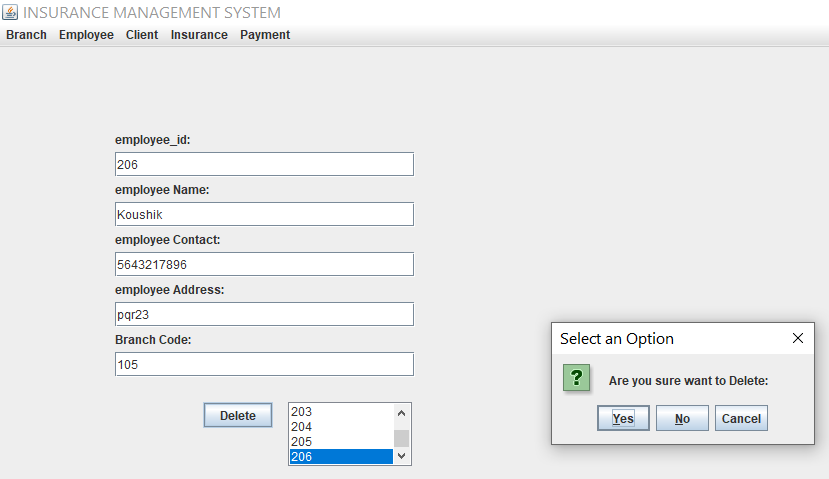






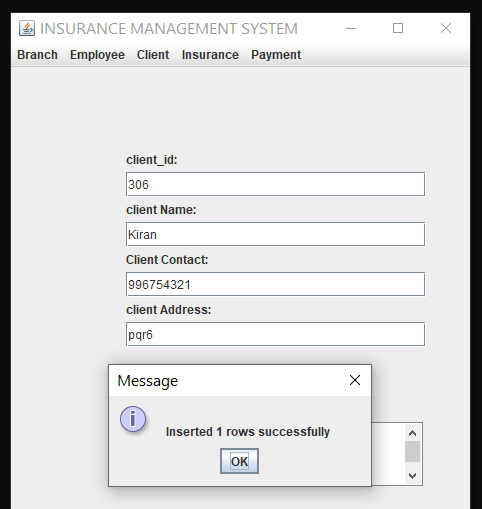
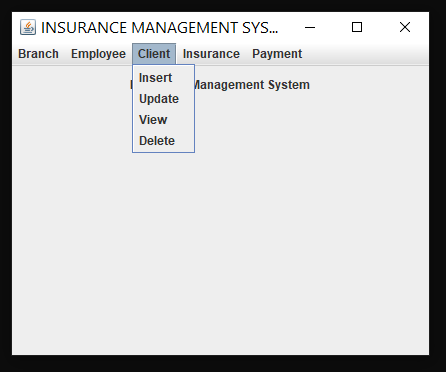


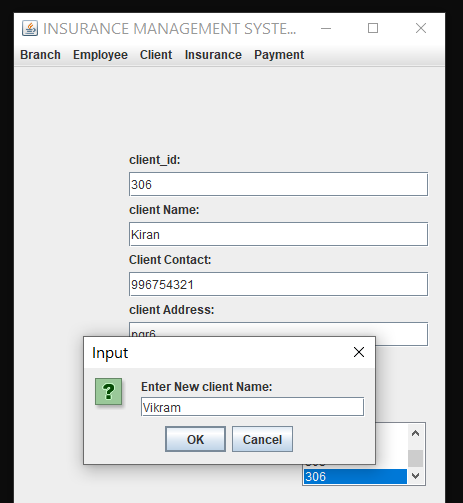
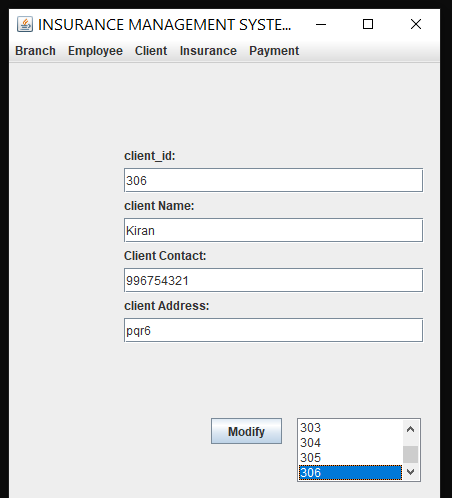


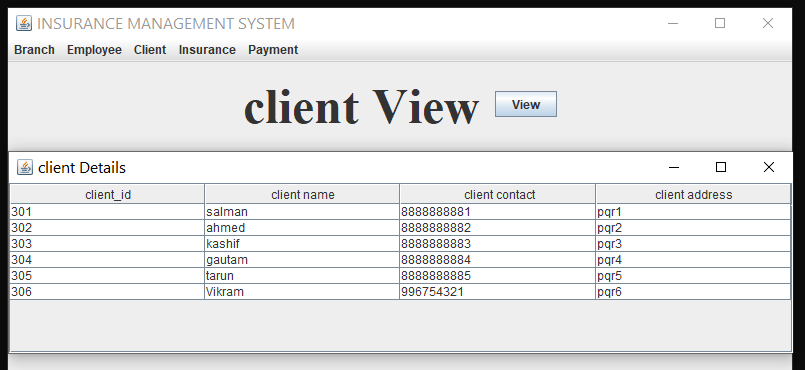


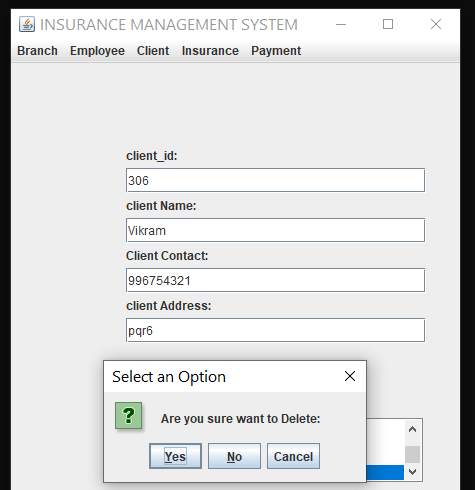
**CLIENT TABLE:**

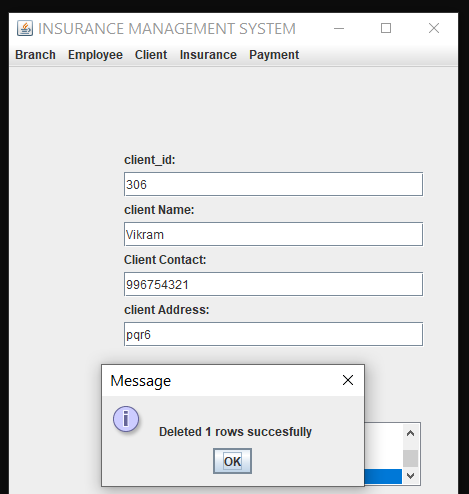
The client\_id is an unique value and it can only have a numeric value and not anything else, similarly the client\_name can only be a string. In case of any unexpected or incorrect input it throws an exception.





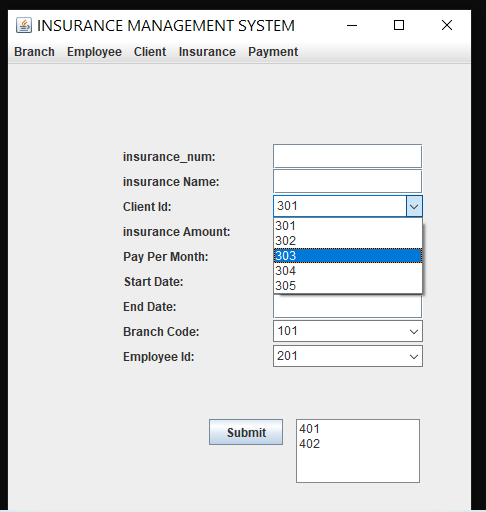


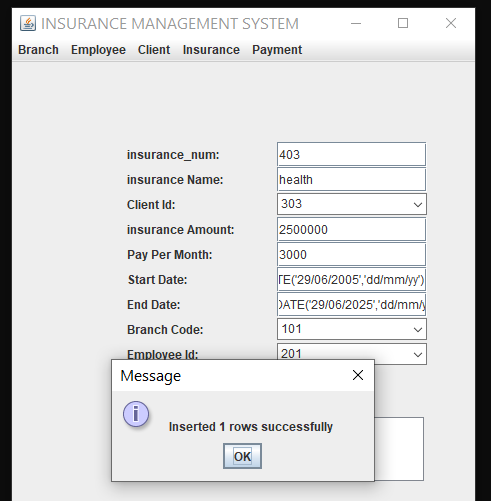


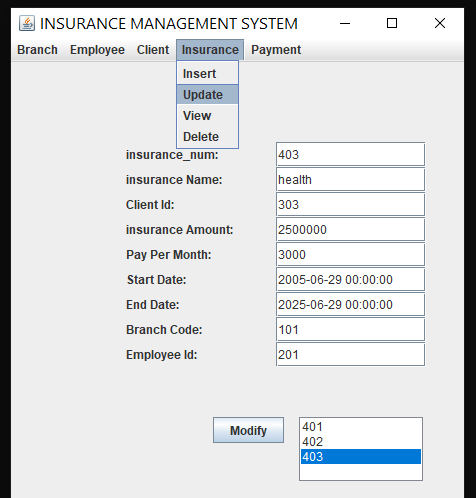


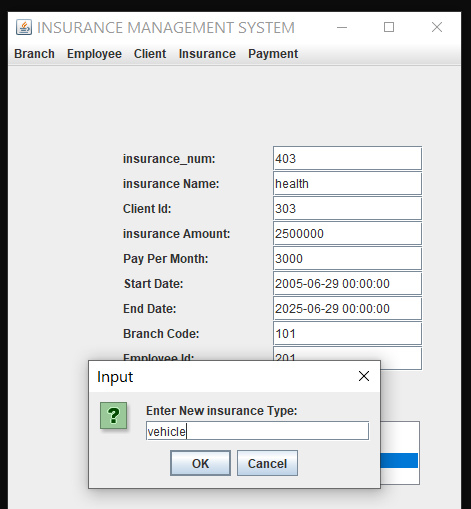
**INSURANCE TABLE:**

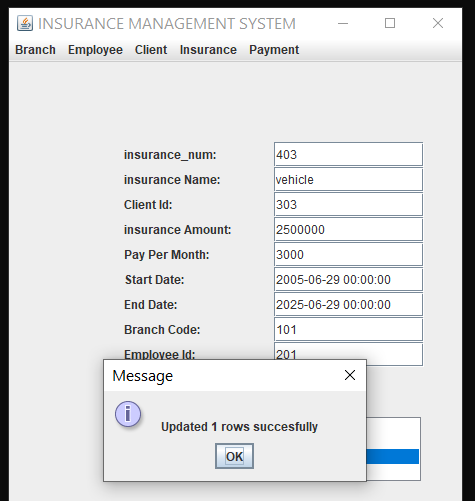
In insurance table insurance\_num is primary key and we have drop down for all foreign keys.

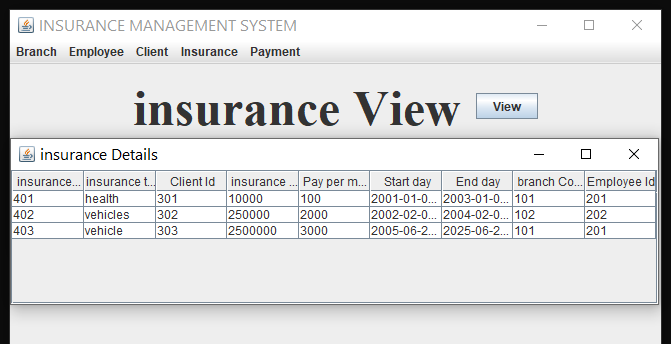


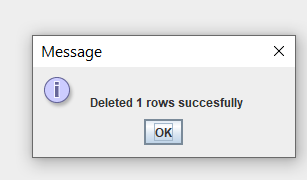
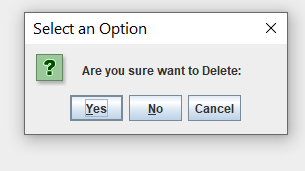
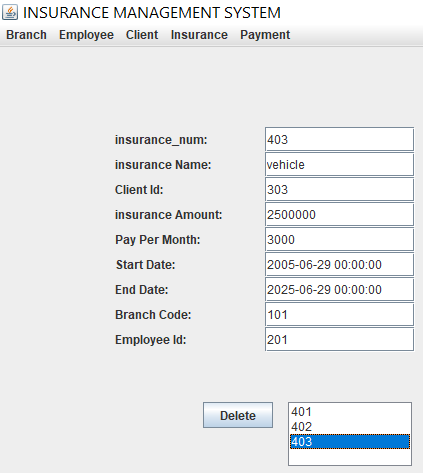






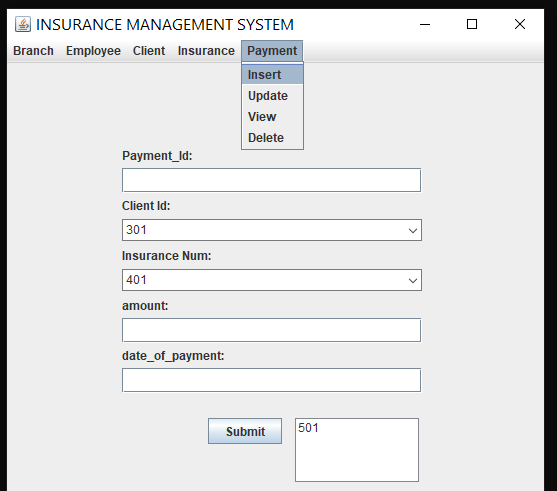


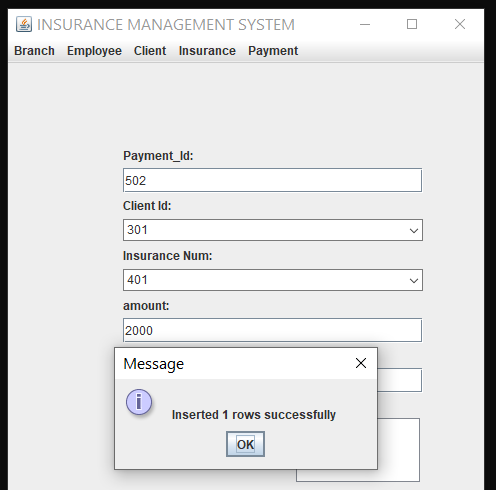


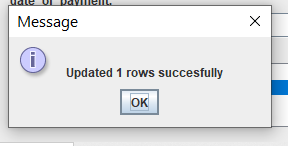
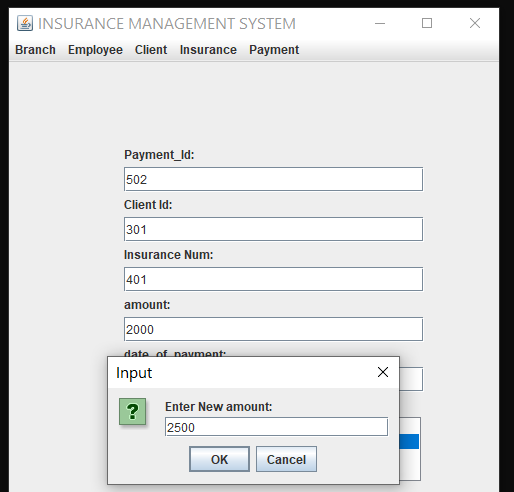


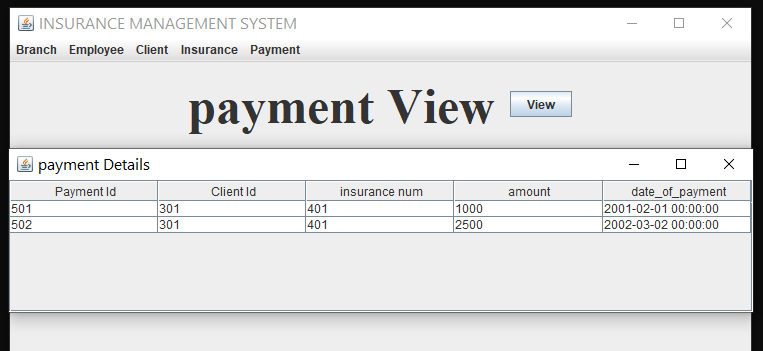
**PAYMENT TABLE:**

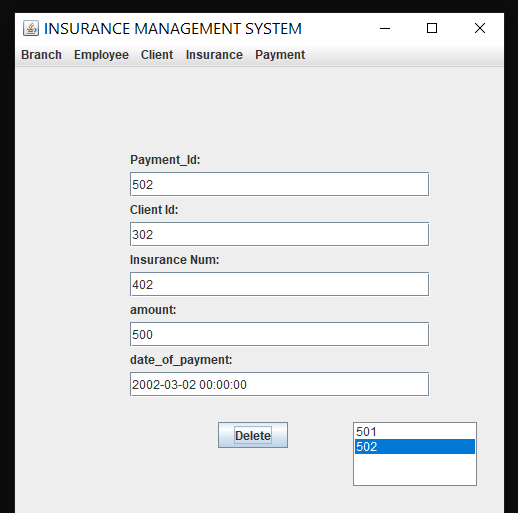
Here in payment table payment id is primary key and foreign keys are client\_id , insurance\_num and they have drop downs froms which we can select.

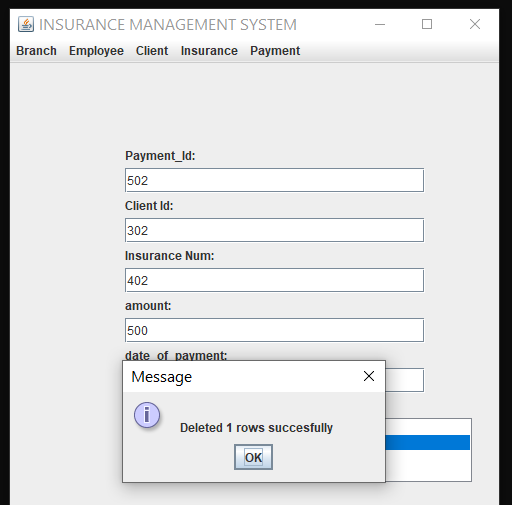












**GITHUB LINK**

https://github.com/saiprasadkoyalkar/INSURANCE-MANAGEMENT-SYSTEM/

**DISCUSSION AND FUTURE WORK**

This project contains the basic interaction of applying for insurance by clients and getting insurance with the help of employees. It has a very basic user interface. Future scope would be to make the UI more appealing by using graphics. One more feature would be to allow client-users to upload their financial status , business ,official documents required so that the employees can take a look while they give insurance to them. We can also think of including a feedback system to allow the users to leave their valuable feedback after using this app. Making this feedback to be publicly viewable, would attract many more users to use this app.

**REFERENCES**

* [Overview (Java Platform SE 7 ) (oracle.com)](https://docs.oracle.com/javase/7/docs/api/)
* [Java Swing Tutorial - javatpoint](https://www.javatpoint.com/java-swing)
* [Stack Overflow - Where Developers Learn, Share, & Build Careers](https://stackoverflow.com/)