JAVA SWING BASED- CYBER SECURITY DATA MANAGEMENT- SQL CONNECTIVITY USING JDBC

A

Report

Submitted in partial fulfilment of the

Requirements for the award of the Degree of

BACHELOR OF ENGINEERING

IN

INFORMATION TECHNOLOGY

BY:

Mahendra Chittupolu <1602-18-737-081>

Department of Information Technology

Vasavi College of Engineering (Autonomous)

(Affiliated to Osmania University)

Ibrahimbagh, Hyderabad-31

2020

BONAFIDE CERTIFICATE

This to Certify that the project report titled "**CYBER SECURITY DATA MANAGEMENT"** project work of Mr.C.H.Mahendra bearing Roll.no:1602-18-737-081 who carried out this project under my supervision in the IV semester for the academic year 2019-2020

Signature
external examine

Signature
internal examine

ABSTRACT

Since most of the world now a days runs through internet and many of us are not able to use it properly and cyber crimes are raising to store the data of complaining process and resolve this project helps to store data and change and delete records data of entries.

After recording we can view them and it stores data starting from victim to culprit. And in middle we can also have records of them employee like to whom the assigned and what is the process and status of the complain and also we can trace the culprit. And what action taken on them.

Through this we can save data efficiently in a better way using RDBMS and also it helps police to look and register complaints

INTRODUCTION:

Requirement Analysis:

List of Tables:

1. User's record
2. Cyber crime
3. Complaints received
4. Department
5. Monitor
6. Action taken
7. Criminal record

List of attributes with their Domain types:

1.User's record:

- Name: Varchar(20)
- Phno: Number(10)
- usid: varchar(15)
- age: number(2)
- hno: varchar(15)
- street: varchar(15)
- Mandal: varchar(15)
- District: varchar(15)

2.Cyber Crime:

- cid: varchar(10)
- location: varchar(25)
- category: varchar(15)

3.Complaints received:

- usid: varchar(15)
- cid: varchar(10)
- received date: Date

4.Department:

- eid: varchar(15)
- ename: varchar(20)
- Designation: varchar(20)
- Branch: varchar(20)

5.Monitor:

- cid: varchar(10)
- eid: varchar(15)
- status: varchar(10)

6.Action taken:

- pid: varchar(15)
- eid: varchar(15)
- date resolved: DATE

7.Criminal record:

- Name: varchar(15)
- Ip addr: varchar(20)
- Pid: varchar(15)

THROUGH THE PROJECT:

This project helps to store data in a efficient way and it can be achieved through various sql commands and we can also store this for any future use and also we can save our data in a many different areas so we cannot lost all the data at once. The user details cannot be lost so it is safer to use it . Users details can be made and also make to visible only to the users and also employee details can be make to visible to only employee, we can make this and can happen. And also by this we can track every ones data in a simple way and also we can make thus available to them and useful .

ARCHITECTURE AND TECHNOLOGY USED:

SOFTWARE USED:

Java Eclipse, Oracle 11g Database, Java SE version 8, SQL LITE.

Java SWING:

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.

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:

```
private void connToDb(){
              try {
             Class.forName("oracle.jdbc.driver.OracleDriver");
             connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1522:xe","Mahendra","sqlma
hi");
             statement = connection.createStatement();

           } catch (SQLException connectException) {
             System.out.println(connectException.getMessage());
             System.out.println(connectException.getSQLState());
             System.out.println(connectException.getErrorCode());
             System.exit(1);
           }
           catch (Exception e)
          {
               System.err.println("Unable to find and load driver");
               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.

DESIGN:

ER DIAGRAM:

DATA DESIGN:

DDL COMMANDS:

```
SQL> CREATE TABLE users_record(
  2  name varchar(20),
  3  phno number(10),
  4  usid varchar(15),
  5  age number(2),
  6  hno varchar(7),
  7  street varchar(15),
  8  mandal varchar(15),
  9  district varchar(15),
 10  primary key(usid));

Table created.
```

```
SQL> CREATE TABLE cyber_crime(
  2  cid varchar(10),
  3  location varchar(25),
  4  category varchar(15),
  5  primary key(cid));

Table created.
```

```
SQL> CREATE TABLE complaints_received(
  2  usid varchar(15),
  3  cid varchar(10),
  4  received_date date,
  5  foreign key(usid) references users_record,
  6  foreign key(cid) references cyber_crime,
  7  primary key(usid,cid));

Table created.
```

```
SQL> CREATE TABLE department(
  2  eid varchar(15),
  3  ename varchar(20),
  4  designation varchar(20),
  5  branch varchar(20),
  6  primary key(eid));

Table created.
```

```
SQL> CREATE TABLE monitors(
  2  cid varchar(10),
  3  eid varchar(15),
  4  status varchar(10),
  5  foreign key(cid) references cyber_crime,
  6  foreign key(eid) references department,
  7  primary key(cid));

Table created.
```

```
SQL> CREATE TABLE criminal_record(
  2  name varchar(15),
  3  ip_addr varchar(20),
  4  pid varchar(15),
  5  primary key(pid));

Table created.
```

```
SQL> CREATE TABLE action_taken(
  2  pid varchar(15),
  3  eid varchar(15),
  4  date_resolved date,
  5  foreign key(eid) references department,
  6  foreign key(pid) references criminal_record,
  7  primary key(pid));

Table created.
```

SQL> select *from tab;

TNAME TABTYPE CLUSTERID

------------------------------ ------- ----------

ACTION_TAKEN TABLE

COMPLAINTS_RECEIVED TABLE

CRIMINAL_RECORD TABLE

CYBER_CRIME TABLE

DEPARTMENT TABLE

MONITORS TABLE

USERS_RECORD TABLE

7 rows selected.

SQL> desc users_record;

Name	Null?	Type
NAME		VARCHAR2(20)
PHNO		NUMBER(10)
USID	NOT NULL	VARCHAR2(15)

AGE	NUMBER(2)
HNO	VARCHAR2(15)
STREET	VARCHAR2(15)
MANDAL	VARCHAR2(15)
DISTRICT	VARCHAR2(15)

SQL> desc cyber_crime;

Name	Null?	Type
CID	NOT NULL	VARCHAR2(10)
LOCATION		VARCHAR2(25)
CATEGORY		VARCHAR2(15)

SQL> desc complaints_received;

Name	Null?	Type
USID	NOT NULL	VARCHAR2(15)
CID	NOT NULL	VARCHAR2(10)
RECEIVED_DATE		DATE

SQL> desc department;

Name	Null?	Type
EID	NOT NULL	VARCHAR2(15)
ENAME		VARCHAR2(20)
DESIGNATION		VARCHAR2(20)
BRANCH		VARCHAR2(20)

SQL> desc monitors;

Name	Null?	Type
CID	NOT NULL	VARCHAR2(10)
EID	NOT NULL	VARCHAR2(15)
STATUS		VARCHAR2(10)

SQL> desc criminal_record;

Name	Null?	Type

NAME		VARCHAR2(15)
IP_ADDR		VARCHAR2(20)
PID	NOT NULL	VARCHAR2(15)

SQL> desc action_taken;

Name	Null?	Type
PID	NOT NULL	VARCHAR2(15)
EID	NOT NULL	VARCHAR2(15)
DATE_RESOLVED		DATE

IMPLEMENTATION:

USER INTERFACE:

```
package view;

import java.awt.CardLayout;
import java.awt.Color;

import complaintsReceived.*;
import java.awt.FlowLayout;
import java.awt.Font;
import java.awt.GridBagLayout;
import complaintsReceived.*;

import users.*;

import java.awt.GridLayout;
import java.awt.TextArea;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.WindowAdapter;
import java.awt.event.WindowEvent;

import javax.swing.BorderFactory;
import javax.swing.ImageIcon;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JMenu;
import javax.swing.JMenuBar;
import javax.swing.JMenuItem;
```

```
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.UIManager;
import javax.swing.UnsupportedLookAndFeelException;
import javax.swing.border.Border;

import actionTaken.delAction;
import actionTaken.insertAction;
import actionTaken.viewAction;
import complaintsReceived.insertComplaints;
import criminalRecords.delCriminalrec;
import criminalRecords.insertCriminalrec;
import criminalRecords.viewCriminalrec;
import cyberCrime.delCyber;
import cyberCrime.insertCyber;
import cyberCrime.viewCyber;
import department.delDept;
import department.insertDept;
import department.viewDept;
import monitors.delMonitors;
import monitors.insertMonitors;
import monitors.viewMonitors;
import users.insertUsers;

public class cyberGUI {
	private JLabel lbl;
	private JFrame f;
	private JMenu mnuusers;
	private JMenu mnuccrime;
	private JMenu mnucomprec;
	private JMenu mnudept;
	private JMenu mnumon;
	private JMenu mnucrec;
	private JMenu mnuactkn;
	private JMenuItem miinusers;
	private JMenuItem midelusers;
	private JMenuItem mivwusers;
	private JMenuItem miinccrime;
	private JMenuItem midelccrime;
	private JMenuItem mivwccrime;
	private JMenuItem miincomprec;
	private JMenuItem midelcomprec;
```

```
        private JMenuItem mivwcomprec;
        private JMenuItem miindept;
        private JMenuItem mideldept;
        private JMenuItem mivwdept;
        private JMenuItem miinmon;
        private JMenuItem midelmon;
        private JMenuItem mivwmon;
        private JMenuItem miincrec;
        private JMenuItem midelcrec;
        private JMenuItem mivwcrec;
        private JMenuItem miinactkn;
        private JMenuItem midelactkn;
        private JMenuItem mivwactkn;
        private JMenuBar mnuBar;
        private JPanel pnl;


public cyberGUI() {
        intializeItems();
        AddItemsToFrame();

}
public void intializeItems()
{
        f = new JFrame();
        lbl = new JLabel("Cyber Security Data Management");
        mnuBar = new JMenuBar();
        mnuusers = new JMenu("User's Records");
        mnuccrime = new JMenu("Cyber Crime");
        miinusers = new JMenuItem("insert user's");
        midelusers = new JMenuItem("delete user's");
        mivwusers = new JMenuItem("view user's");
        miinccrime = new JMenuItem("insert cyber crime");
        midelccrime = new JMenuItem("delete cyber crime");
        mivwccrime = new JMenuItem("view cyber crime");
        mnucomprec = new JMenu("Complaint's Recieved");
        miincomprec = new JMenuItem("insert complaints recieved");
        midelcomprec = new JMenuItem("delete complaints recieved");
        mivwcomprec = new JMenuItem("view complaints recieved");
        mnudept = new JMenu("Department");
        miindept = new JMenuItem("insert department");
        mideldept = new JMenuItem("delete department");
```

```
        mivwdept = new JMenuItem("view department");
        mnumon = new JMenu("Monitor's");
        miinmon = new JMenuItem("insert into monitor's");
        midelmon = new JMenuItem("delete monitor's");
        mivwmon = new JMenuItem("view Monitor's");
        mnucrec = new JMenu("Criminal Record's");
        miincrec = new JMenuItem("insert criminal record");
        midelcrec = new JMenuItem("delete criminal record");
        mivwcrec = new JMenuItem("view criminal records");
        mnuactkn = new JMenu("Action Taken");
        miinactkn = new JMenuItem("insert actiontaken");
        midelactkn = new JMenuItem("delete action taken");
        mivwactkn = new JMenuItem("view action taken");
        pnl = new JPanel(new FlowLayout());

}
public void AddItemsToFrame()
{
        mnuusers.add(miinusers);

        mnuusers.add(midelusers);
        mnuusers.add(mivwusers);
        mnuccrime.add(miinccrime);
        mnuccrime.add(midelccrime);
        mnuccrime.add(mivwccrime);
        mnucomprec.add(miincomprec);
        mnucomprec.add(midelcomprec);
        mnucomprec.add(mivwcomprec);
        mnudept.add(miindept);
        mnudept.add(mideldept);
        mnudept.add(mivwdept);
        mnumon.add(miinmon);
        mnumon.add(midelmon);
        mnumon.add(mivwmon);
        mnucrec.add(miincrec);
        mnucrec.add(midelcrec);
        mnucrec.add(mivwcrec);
        mnuactkn.add(miinactkn);
        mnuactkn.add(midelactkn);
        mnuactkn.add(midelactkn);
        mnuactkn.add(mivwactkn);
        mnuBar.add(mnuusers);
```

```
mnuBar.add(mnucomprec);
mnuBar.add(mnuccrime);
mnuBar.add(mnumon);
mnuBar.add(mnudept);
mnuBar.add(mnuactkn);
mnuBar.add(mnucrec);
Font font = new Font("Monaco", Font.BOLD,25);
Color clr = new Color(200, 100, 150);


lbl.setFont(font);
lbl.setForeground(Color.RED);

miinusers.addActionListener(new ActionListener() {

	@Override
	public void actionPerformed(ActionEvent e) {
		// TODO Auto-generated method stub
		insertUsers inuser = new insertUsers();
		inuser.insusers();
		}

});
midelusers.addActionListener(new ActionListener() {

	@Override
	public void actionPerformed(ActionEvent e) {
		delUsers deluser = new delUsers();
		deluser.delusers();
		// TODO Auto-generated method stub
	}
});
mivwusers.addActionListener(new ActionListener() {

	@Override
	public void actionPerformed(ActionEvent e) {
		// TODO Auto-generated method stub
		viewUsers vwuser = new viewUsers();
		vwuser.vwusers();
	}
});
miincomprec.addActionListener(new ActionListener() {
```

```
		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			insertComplaints incomp = new insertComplaints();
			incomp.incompliants();
		}
});
midelcomprec.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			delComplaints delComp = new delComplaints();
			delComp.delComp();
		}
});
mivwcomprec.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			viewComplaints vwcomp = new viewComplaints();
			vwcomp.vwComp();
			// TODO Auto-generated method stub
		}
});
miinccrime.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			insertCyber inscyber = new insertCyber();
			inscyber.insertcyber();
			// TODO Auto-generated method stub
		}
});
midelccrime.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			delCyber delcyber = new delCyber();
			delcyber.delcyber();
			// TODO Auto-generated method stub
```

```
		}
});
mivwccrime.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			viewCyber vwcyber = new viewCyber();
			vwcyber.viewcyber();
		}
});
miinmon.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			insertMonitors insmon = new insertMonitors();
			insmon.insmoni();
		}
});
midelmon.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			delMonitors delmon = new delMonitors();
			delmon.delmoni();
		}
});
mivwmon.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			viewMonitors vwmon = new  viewMonitors();
			vwmon.vwmoni();
		}
});
miindept.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
```

```
                // TODO Auto-generated method stub
                insertDept insdept = new insertDept();
                insdept.insdept();
        }
});
mideldept.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                delDept deldept = new delDept();
                deldept.deldept();
        }
});
mivwdept.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                viewDept vwDept = new viewDept();
                vwDept.vwdept();
        }
});
miinactkn.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                insertAction insacn = new insertAction();
                insacn.insactn();
        }
});
midelactkn.addActionListener(new ActionListener() {

        @Override
        public void actionPerformed(ActionEvent e) {
                // TODO Auto-generated method stub
                delAction delactn = new delAction();
                delactn.delacn();
        }
});
mivwactkn.addActionListener(new ActionListener() {
```

```
		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			viewAction vwacn = new viewAction();
			vwacn.vwacn();
		}
	});
	miincrec.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			insertCriminalrec inscrim = new insertCriminalrec();
			inscrim.inscrimrec();
		}
	});
	midelcrec.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			delCriminalrec delcrim = new delCriminalrec();
			delcrim.delcrimrec();
		}
	});
	mivwcrec.addActionListener(new ActionListener() {

		@Override
		public void actionPerformed(ActionEvent e) {
			// TODO Auto-generated method stub
			viewCriminalrec vwcrim = new viewCriminalrec();
			vwcrim.vwcrimrec();
		}
	});
	 f.addWindowListener(new WindowAdapter(){
			public void windowClosing(WindowEvent evt)
			{

				int confirmed = JOptionPane.showConfirmDialog(null, "DO YOU WANT
TO EXIT?","!!THIS CLOSES ENTERING DATA!!",JOptionPane.YES_NO_OPTION);
		if(confirmed == JOptionPane.YES_OPTION)
```

```
                {
                    System.exit(0);

                }



                }
            });


        f.add(new JLabel(new ImageIcon("C:\\Users\\mahendra\\Desktop\\lang\\java\\dbms/image.jpg")));

        //f.setBounds(0,0,0,0);

        f.setBackground(clr);
        f.add(pnl);
        f.setJMenuBar(mnuBar);
        f.setTitle("CYBER SECURITY DATA MANAGEMENT");
        f.setLayout(new FlowLayout());
        f.setSize(1400,1400);
        f.setVisible(true);

        try {
            UIManager.setLookAndFeel("com.sun.java.swing.plaf.windows.WindowsLookAndFeel");
        } catch (ClassNotFoundException | InstantiationException | IllegalAccessException
                | UnsupportedLookAndFeelException e1) {
            // TODO Auto-generated catch block
            e1.printStackTrace();
        }

}
}
```

USERS_RECORD:

INSERT_USERS:

```
package users;

import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.event.ActionEvent;
```

```
import java.awt.event.ActionListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;

public class insertUsers {
	Connection connection;
	Statement statement;
	public void connectToDB()
  {
		try
		{
		  connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Mahendra","sqlmahi");
		  statement = connection.createStatement();

		}
		catch (SQLException connectException)
		{
		  System.out.println(connectException.getMessage());
		  System.out.println(connectException.getSQLState());
		  System.out.println(connectException.getErrorCode());
		  System.exit(1);
		}
  }


	public void insusers() {
		  try
			{
				Class.forName("oracle.jdbc.driver.OracleDriver");
			}
			catch (Exception e)
```

```
            {
                    System.err.println("Unable to find and load driver");
                    System.exit(1);
            }
            connectToDB();
    JFrame f = new JFrame();
    JLabel lname;
    JLabel lphno;
    JLabel lusid;
    JLabel lage;
    JLabel lhno;
    JLabel lstreet;
    JLabel lmandal;
    JLabel ldistrict;
    JTextField tfname;
    JTextField tfphno;
    JTextField tfusid;
    JTextField tfage;
    JTextField tfhno;
    JTextField tfstreet;
    JTextField tfmandal;
    JTextField tfdistrict;
    JTextArea tfdesc;
    JButton btnins;


    lname = new JLabel("Name");
    lphno = new JLabel("Ph.no");
    lusid = new JLabel("User id");
    lage = new JLabel("Age");
    lhno = new JLabel("H:no");
    lstreet = new JLabel("Street");
    lmandal = new JLabel("Mandal");
    ldistrict = new JLabel("District");
    tfname = new JTextField(15);
    tfphno = new JTextField(15);
    tfusid = new JTextField(15);
    tfage = new JTextField(15);
    tfhno = new JTextField(15);
    tfstreet = new JTextField(15);
    tfmandal = new JTextField(15);
    tfdistrict = new JTextField(15);
```

```
tfdesc = new JTextArea(10,55);
btnins = new JButton("SUBMIT");
JPanel pnl = new JPanel();
JPanel pnl1 = new JPanel();
JPanel pnl2 = new JPanel();


btnins.addActionListener(new ActionListener() {

@Override
public void actionPerformed(ActionEvent e) {
// TODO Auto-generated method stub
try
{
Statement statement = connection.createStatement();
//String query = "INSERT INTO sailors (userid,NAME,AGE,hno,street,mandal,dist) VALUES (2,'Divya',7,20)";
String query= "INSERT INTO users_record VALUES('"+tfname.getText()+"',"+ tfphno.getText()
+",'"+tfusid.getText()+"','"+tfage.getText()+"','"+tfhno.getText()
+"','"+tfstreet.getText()+"','"+tfmandal.getText()+"','"+tfdistrict.getText()+"')";
int i = statement.executeUpdate(query);
tfdesc.setText(null);
tfdesc.append("\nInserted " + i + " rows successfully");
tfname.setText(null);
tfphno.setText(null);
tfusid.setText(null);
tfage.setText(null);
tfhno.setText(null);
tfstreet.setText(null);
tfdistrict.setText(null);
tfmandal.setText(null);


}
catch (SQLException insertException)
{

tfdesc.append("\nENTER phno and age in number format ONLY!!");
tfdesc.append("\nSQLException: " + insertException.getMessage()
+ "\n");
tfdesc.append("SQLState:     " + insertException.getSQLState() +
"\n");
tfdesc.append("VendorError:  " + insertException.getErrorCode() +
"\n");
```

```
                                        tfname.setText(null);
                                                tfphno.setText(null);
                                                tfusid.setText(null);
                                                tfage.setText(null);
                                                tfhno.setText(null);
                                                tfstreet.setText(null);
                                                tfdistrict.setText(null);
                                                tfmandal.setText(null);


                                    }
                            }
                    });



            pnl1.add(lusid);
            pnl1.add(tfusid);
            pnl1.add(lname);
            pnl1.add(tfname);
            pnl1.add(lphno);
            pnl1.add(tfphno);
            pnl1.add(lage);
            pnl1.add(tfage);
            pnl1.add(lhno);
            pnl1.add(tfhno);
            pnl1.add(lstreet);
            pnl1.add(tfstreet);
            pnl1.add(lmandal);
            pnl1.add(tfmandal);
            pnl1.add(ldistrict);
            pnl1.add(tfdistrict);
            pnl2.add(tfdesc);
            pnl.add(btnins);
            pnl1.setLayout(new FlowLayout());
            pnl2.setLayout(new FlowLayout());

             f.getContentPane().setBackground(Color.DARK_GRAY);

             f.add(pnl1);
             f.add(pnl);
```

```
            f.add(pnl2);

            f.setLayout(new FlowLayout());

            f.setVisible(true);
            f.setSize(1500,600);
            f.setTitle("Insert User's");

        }
}
```

VIEW USERS:

```
package users;

import java.awt.Color;
import java.awt.FlowLayout;
import java.awt.GridLayout;
import java.awt.List;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.awt.FlowLayout;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;

public class viewUsers {
        Connection connection;
        Statement statement;
        ResultSet rs;
        public void connectToDB()
  {
```

```
                try
                {
                 connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Mahendra","sqlmahi");
                 statement = connection.createStatement();


                }
                catch (SQLException connectException)
                {
                 System.out.println(connectException.getMessage());
                 System.out.println(connectException.getSQLState());
                 System.out.println(connectException.getErrorCode());
                 System.exit(1);
                }
  }

        public void vwusers() {
                try
                {
                        Class.forName("oracle.jdbc.driver.OracleDriver");
                }
                catch (Exception e)
                {
                        System.err.println("Unable to find and load driver");
                        System.exit(1);
                }
                connectToDB();
                JFrame f = new JFrame();
                JLabel lname;
                JLabel lphno;
                JLabel lusid;
                JLabel lage;
                JLabel lhno;
                JLabel lstreet;
                JLabel lmandal;
                JLabel ldistrict;
                JTextField tfname;
                JTextField tfphno;
                JTextField tfusid;
                JTextField tfage;
                JTextField tfhno;
```

```
        JTextField tfstreet;
        JTextField tfmandal;
        JTextField tfdistrict;
        JTextArea jtadesc;
        JButton btnup;
        List liusid;


        lname = new JLabel("Name");
        lphno = new JLabel("Ph.no");
        lusid = new JLabel("User id");
        lage = new JLabel("Age");
        lhno = new JLabel("H:no");
        lstreet = new JLabel("Street");
        lmandal = new JLabel("Mandal");
        ldistrict = new JLabel("District");
        tfname = new JTextField(15);
        tfphno = new JTextField(15);
        tfusid = new JTextField(15);
        tfage = new JTextField(15);
        tfhno = new JTextField(15);
        tfstreet = new JTextField(15);
        tfmandal = new JTextField(15);
        tfdistrict = new JTextField(15);
        jtadesc = new JTextArea(10,50);
        liusid = new List(10);
        btnup = new JButton("MODIFY");
    JPanel pnlname = new JPanel();
    JPanel pnlphno = new JPanel();
    JPanel pnlusid = new JPanel();
    JPanel pnlage = new JPanel();
    JPanel pnlhno = new JPanel();
    JPanel pnlstreet = new JPanel();
    JPanel pnlmandal = new JPanel();
    JPanel pnldistrict = new JPanel();
    JPanel pnl = new JPanel();
    JPanel pnl1 = new JPanel();
    JPanel pnl2 = new JPanel();
    jtadesc.setEditable(false);
    try
        {
```

```
                rs = statement.executeQuery("SELECT * FROM USERS_RECORD");
                while (rs.next())
                {
                        liusid.add(rs.getString("USID"));
                }
               }
               catch (SQLException e)
               {
                        jtadesc.append("\nSQLException: " + e.getMessage() + "\n");
                        jtadesc.append("SQLState:     " + e.getSQLState() + "\n");
                        jtadesc.append("VendorError:  " + e.getErrorCode() + "\n");
               }
        liusid.addItemListener(new ItemListener() {

                        @Override
                        public void itemStateChanged(ItemEvent e) {
                                // TODO Auto-generated method stub
                                try {
                                        rs = statement.executeQuery("SELECT *FROM USERS_RECORD
WHERE USID = "+liusid.getSelectedItem());
                                        rs.next();
                                        tfusid.setText(rs.getString("USID"));
                                        tfname.setText(rs.getString("NAME"));
                                        tfage.setText(rs.getString("AGE"));
                                        tfdistrict.setText(rs.getString("DISTRICT"));
                                        tfhno.setText(rs.getString("HNO"));
                                        tfmandal.setText(rs.getString("MANDAL"));
                                        tfphno.setText(rs.getString("PHNO"));
                                        tfstreet.setText(rs.getString("STREET"));
                                } catch (SQLException e1) {
                                        // TODO Auto-generated catch block
                                        jtadesc.append("\nSQLException: " + e1.getMessage() + "\n");
                                        jtadesc.append("SQLState:     " + e1.getSQLState() + "\n");
                                        jtadesc.append("VendorError:  " + e1.getErrorCode() + "\n");
                                }

                        }
                });
        btnup.addActionListener(new ActionListener() {

                        @Override
```

```
public void actionPerformed(ActionEvent e) {
	// TODO Auto-generated method stub
	try {
		Statement statement = connection.createStatement();
		int i = statement.executeUpdate("update users_record SET name ='"+tfname.getText()+"',phno ="+tfphno.getText()+",usid = '"+tfusid.getText()+
			"',age ="+tfage.getText()+", hno ='"+tfhno.getText()+"',street ='"+tfstreet.getText()+"',mandal ='"+tfmandal.getText()+
			"' ,district ='"+tfdistrict.getText()+"' where usid = '"+liusid.getSelectedItem()+"'");
		jtadesc.setText(null);
		jtadesc.append("Updated"+i+"rows sucessfully");
		liusid.removeAll();
		tfname.setText(null);
		tfage.setText(null);
		tfdistrict.setText(null);
		tfhno.setText(null);
		tfmandal.setText(null);
		tfphno.setText(null);
		tfstreet.setText(null);
		tfusid.setText(null);
		 try
			{
			  rs = statement.executeQuery("SELECT * FROM USERS_RECORD");
			  while (rs.next())
			  {
				liusid.add(rs.getString("USID"));
			  }
			}
			catch (SQLException e3)
			{
				jtadesc.append("\nSQLException: " + e3.getMessage() + "\n");
				jtadesc.append("SQLState:     " + e3.getSQLState() + "\n");
				jtadesc.append("VendorError:  " + e3.getErrorCode() + "\n");
			}

	} catch (SQLException e1) {
		// TODO Auto-generated catch block
		jtadesc.append("ENTER PHNO AND AGE IN NUMBER FORMAT ONLY");
```

```
                                    jtadesc.append("\nSQLException: " + e1.getMessage() + "\n");
                                    jtadesc.append("SQLState:     " + e1.getSQLState() + "\n");
                                    jtadesc.append("VendorError:  " + e1.getErrorCode() + "\n");


                                }
                            }
                        });
                pnl1.add(liusid);
                pnlusid.add(lusid);
                pnlusid.add(tfusid);
                pnlusid.setLayout(new FlowLayout());
                pnlname.add(lname);
                pnlname.add(tfname);
                pnlname.setLayout(new FlowLayout());
                pnlage.add(lage);
                pnlage.add(tfage);
                pnlage.setLayout(new FlowLayout());
                pnlhno.add(lhno);
                pnlhno.add(tfhno);
                pnlhno.setLayout(new FlowLayout());
                pnlstreet.add(lstreet);
                pnlstreet.add(tfstreet);
                pnlstreet.setLayout(new FlowLayout());
                pnlmandal.add(lmandal);
                pnlmandal.add(tfmandal);
                pnlmandal.setLayout(new FlowLayout());
                pnldistrict.add(ldistrict);
                pnldistrict.add(tfdistrict);
                pnldistrict.setLayout(new FlowLayout());

                pnlphno.add(lphno);
                pnlphno.add(tfphno);
                pnlphno.setLayout(new FlowLayout());
                pnl.add(pnlusid);
                pnl.add(pnlname);
                pnl.add(pnlphno);
                pnl.add(pnlage);
                pnl.add(pnlhno);
                pnl.add(pnlstreet);
                pnl.add(pnlmandal);
```

```
            pnl.add(pnldistrict);
            pnl2.add(btnup);
            pnl2.add(jtadesc);
            pnl2.setLayout(new FlowLayout());
            pnl1.setLayout(new FlowLayout());
            pnl.setLayout(new FlowLayout());
            f.getContentPane().setBackground(Color.DARK_GRAY);
            f.add(pnl1);
            f.add(pnl);
            f.add(pnl2);

           f.setSize(1500,600);
            f.setLayout(new FlowLayout());
            f.setVisible(true);

        }

}
```

DELETE USERS:

```
package users;

import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
import java.awt.event.ItemEvent;
import java.awt.event.ItemListener;
import java.sql.DriverManager;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import javax.swing.JButton;
import javax.swing.JFrame;
import javax.swing.JLabel;
import javax.swing.JPanel;
import javax.swing.JTextArea;
import javax.swing.JTextField;

import java.awt.*;

import java.sql.*;
```

```
public class delUsers {
	Connection connection;
	Statement statement;
	ResultSet rs;
	public void connectToDB()
  {
		try
		{
		 connection =
DriverManager.getConnection("jdbc:oracle:thin:@localhost:1521:xe","Mahendra","sqlmahi");
		 statement = connection.createStatement();

		}
		catch (SQLException connectException)
		{
		 System.out.println(connectException.getMessage());
		 System.out.println(connectException.getSQLState());
		 System.out.println(connectException.getErrorCode());
		 System.exit(1);
		}
  }

	public void delusers() {
		 try
			{
				Class.forName("oracle.jdbc.driver.OracleDriver");
			}
			catch (Exception e)
			{
				System.err.println("Unable to find and load driver");
				System.exit(1);
			}
			connectToDB();
		JFrame f = new JFrame();
		JLabel lname = new JLabel("Name");
		JLabel lphno = new JLabel("PH.NO");
		JLabel luserid = new JLabel("User I'D");
		JLabel lage = new JLabel("AGE");
		JLabel lhno = new JLabel("H.no");
		JLabel lstreet = new JLabel("Street");
```

```
JLabel lmandal = new JLabel("Mandal");
JLabel ldist = new JLabel("District");
JTextField tfuserid = new JTextField(15);
JTextField tfname = new JTextField(15);
JTextField tfphno = new JTextField(15);
JTextField tfage = new JTextField(15);
JTextField tfhno = new JTextField(15);
JTextField tfstreet = new JTextField(15);
JTextField tfmandal = new JTextField(15);
JTextField tfdist = new JTextField(15);
 List uidlist = new List(10);
JButton btndel = new JButton("DELETE");
JTextArea jtadesc = new JTextArea(15,55);
JPanel pnl1 = new JPanel();
JPanel pnl2 = new JPanel();
jtadesc.setEditable(false);
        try
        {
          rs = statement.executeQuery("SELECT * FROM users_record");
          while (rs.next())
          {
                uidlist.add(rs.getString("USID"));
          }
        }
        catch (SQLException e)
        {
                jtadesc.append("\nSQLException: " + e.getMessage() + "\n");
                jtadesc.append("SQLState:     " + e.getSQLState() + "\n");
                jtadesc.append("VendorError:  " + e.getErrorCode() + "\n");
        }
uidlist.addItemListener(new ItemListener() {

        @Override
        public void itemStateChanged(ItemEvent e) {
                // TODO Auto-generated method stub
                try {
                        rs =statement.executeQuery("select *from users_record");
                        while (rs.next())
                        {
                                if (rs.getString("USID").equals(uidlist.getSelectedItem()))
                                break;
```

```
                                        }
                                        if (!rs.isAfterLast())
                                        {
                                                tfuserid.setText(rs.getString("USID"));
                                                tfname.setText(rs.getString("NAME"));
                                                tfphno.setText(rs.getString("PHNO"));
                                                tfage.setText(rs.getString("AGE"));
                                                tfhno.setText(rs.getString("HNO"));
                                                tfstreet.setText(rs.getString("STREET"));
                                                tfmandal.setText(rs.getString("MANDAL"));
                                                tfdist.setText(rs.getString("DISTRICT"));

                                        }
                                } catch (SQLException e1) {
                                        // TODO Auto-generated catch block
                                        jtadesc.append("\nSQLException: " + e1.getMessage() + "\n");
                                        jtadesc.append("SQLState:     " + e1.getSQLState() + "\n");
                                        jtadesc.append("VendorError:  " + e1.getErrorCode() + "\n");

                                }

                        }
                });
                btndel.addActionListener(new ActionListener() {

                        @Override
                        public void actionPerformed(ActionEvent e) {
                                // TODO Auto-generated method stub
                                try {
                                        Statement statement = connection.createStatement();
                                        int i = statement.executeUpdate("DELETE FROM users_record
WHERE USID = "
                                                        + uidlist.getSelectedItem());
                                        jtadesc.append("\nDeleted"+i+"Rows Sucessfully");
                                        tfuserid.setText(null);
                                        tfstreet.setText(null);
                                        tfphno.setText(null);
                                        tfname.setText(null);
                                        tfmandal.setText(null);
                                        tfhno.setText(null);
                                        tfdist.setText(null);
```

```
                                        tfage.setText(null);
                                        uidlist.removeAll();
                                        try
                                        {
                                         rs = statement.executeQuery("SELECT * FROM users_record");
                                         while (rs.next())
                                         {
                                                uidlist.add(rs.getString("USID"));
                                         }
                                        }
                                        catch (SQLException e3)
                                        {
                                                jtadesc.append("\nSQLException: " + e3.getMessage() +
"\n");
                                                jtadesc.append("SQLState:     " + e3.getSQLState() + "\n");
                                                jtadesc.append("VendorError:  " + e3.getErrorCode() +
"\n");
                                        }


                                } catch (SQLException e5) {
                                        jtadesc.append("\nSQLException: " + e5.getMessage() + "\n");
                                        jtadesc.append("SQLState:     " + e5.getSQLState() + "\n");
                                        jtadesc.append("VendorError:  " + e5.getErrorCode() + "\n");
                                }
                        }
                });

                pnl2.add(uidlist);
                pnl1.add(lname);
                pnl1.add(tfname);
                pnl1.add(lphno);
                pnl1.add(tfphno);
                pnl1.add(luserid);
                pnl1.add(tfuserid);
                pnl1.add(lage);
                pnl1.add(tfage);
                pnl1.add(lhno);
                pnl1.add(tfhno);
                pnl1.add(lstreet);
                pnl1.add(tfstreet);
                pnl1.add(lmandal);
```

```
                pnl1.add(tfmandal);
                pnl1.add(ldist);
                pnl1.add(tfdist);
                JPanel pnl3 = new JPanel();
                pnl3.add(jtadesc);
                JPanel pnl = new JPanel();
                pnl.add(btndel);
                pnl1.setLayout(new FlowLayout());
                pnl.setLayout(new FlowLayout());
                pnl3.setLayout(new FlowLayout());
                f.getContentPane().setBackground(Color.DARK_GRAY);
                f.setSize(1500, 600);
                f.add(pnl2);
                f.add(pnl1);
                f.add(pnl);
                f.add(pnl3);

                f.setLayout(new FlowLayout());
                f.setVisible(true);


        }

}
```

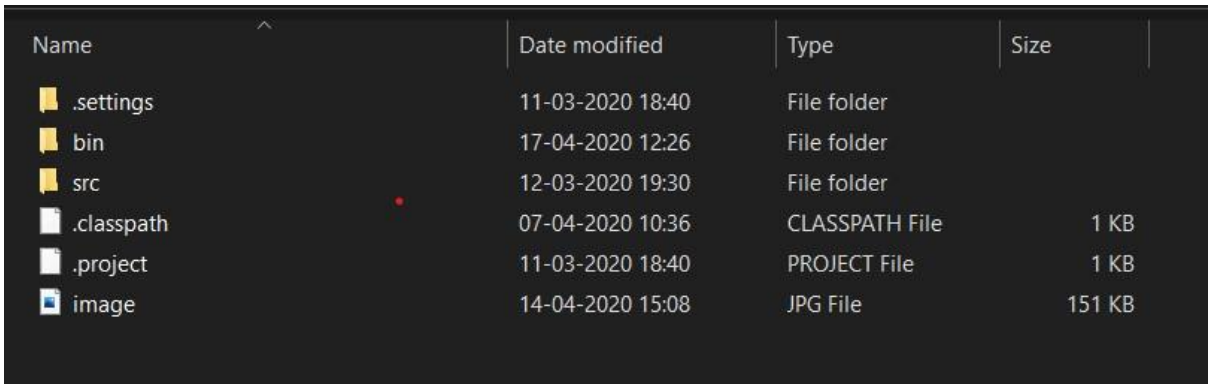
GITHUB LINK:

https://github.com/Mahendra-Chittupolu/DBMS-MINIPROJECT-cybersecurity-

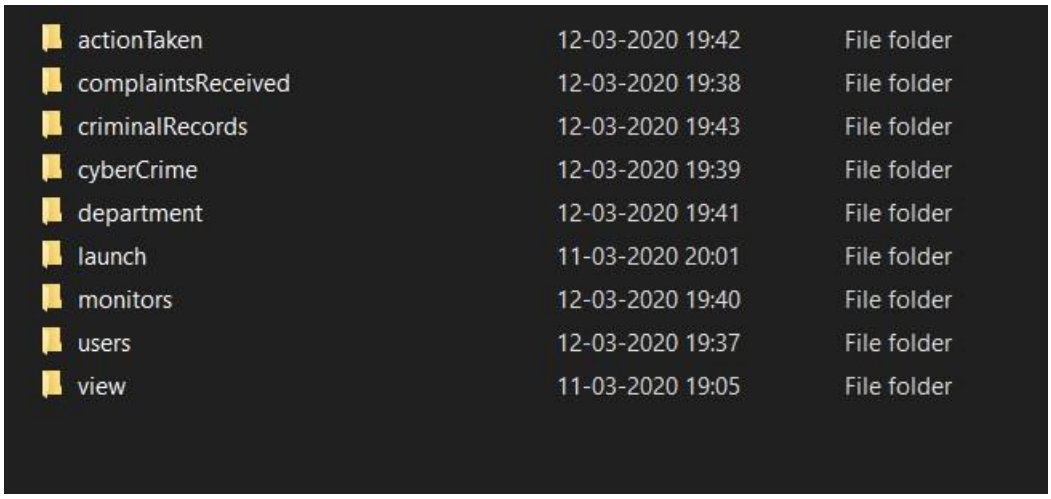
FOLDER STRUCTURE:

This project contains a folder named src in which it has 9 different folders for different purposes each folder has 3 codes such as to make insert, delete, update. By this we can navigate easily to reach code and also we can make many changes we can want easily.

And also those are named perfectly to be understand easily



Name	Date modified	Type	Size
.settings	11-03-2020 18:40	File folder	
bin	17-04-2020 12:26	File folder	
src	12-03-2020 19:30	File folder	
.classpath	07-04-2020 10:36	CLASSPATH File	1 KB
.project	11-03-2020 18:40	PROJECT File	1 KB
image	14-04-2020 15:08	JPG File	151 KB

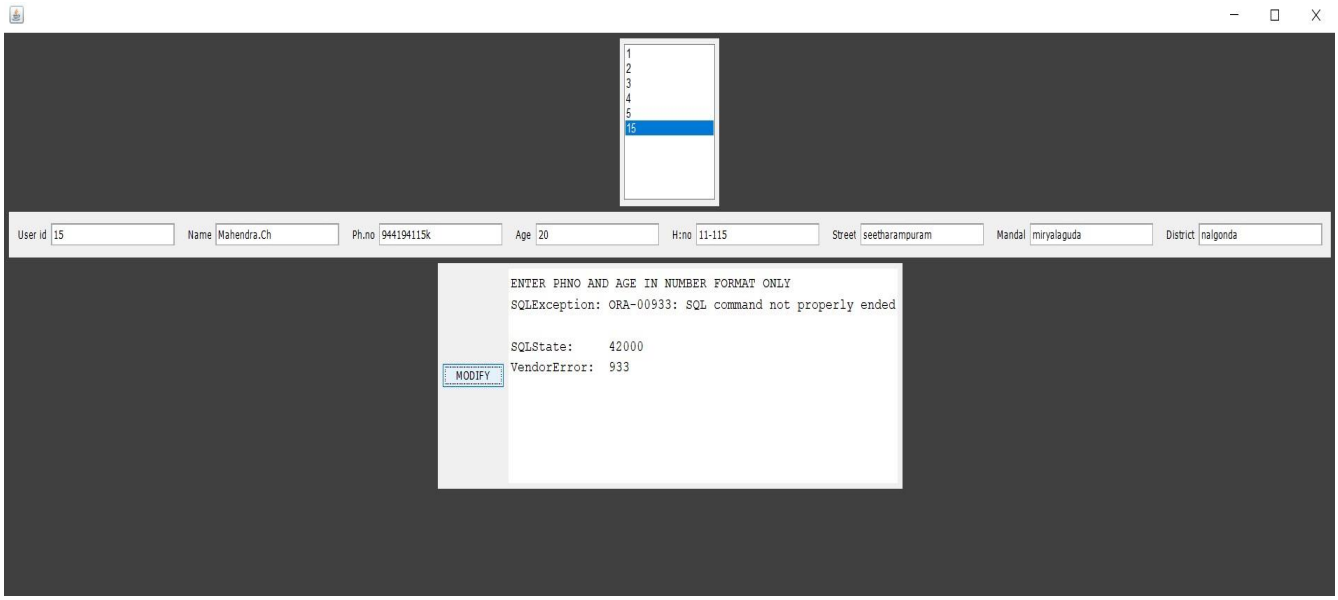


actionTaken	12-03-2020 19:42	File folder
complaintsReceived	12-03-2020 19:38	File folder
criminalRecords	12-03-2020 19:43	File folder
cyberCrime	12-03-2020 19:39	File folder
department	12-03-2020 19:41	File folder
launch	11-03-2020 20:01	File folder
monitors	12-03-2020 19:40	File folder
users	12-03-2020 19:37	File folder
view	11-03-2020 19:05	File folder

Testing:

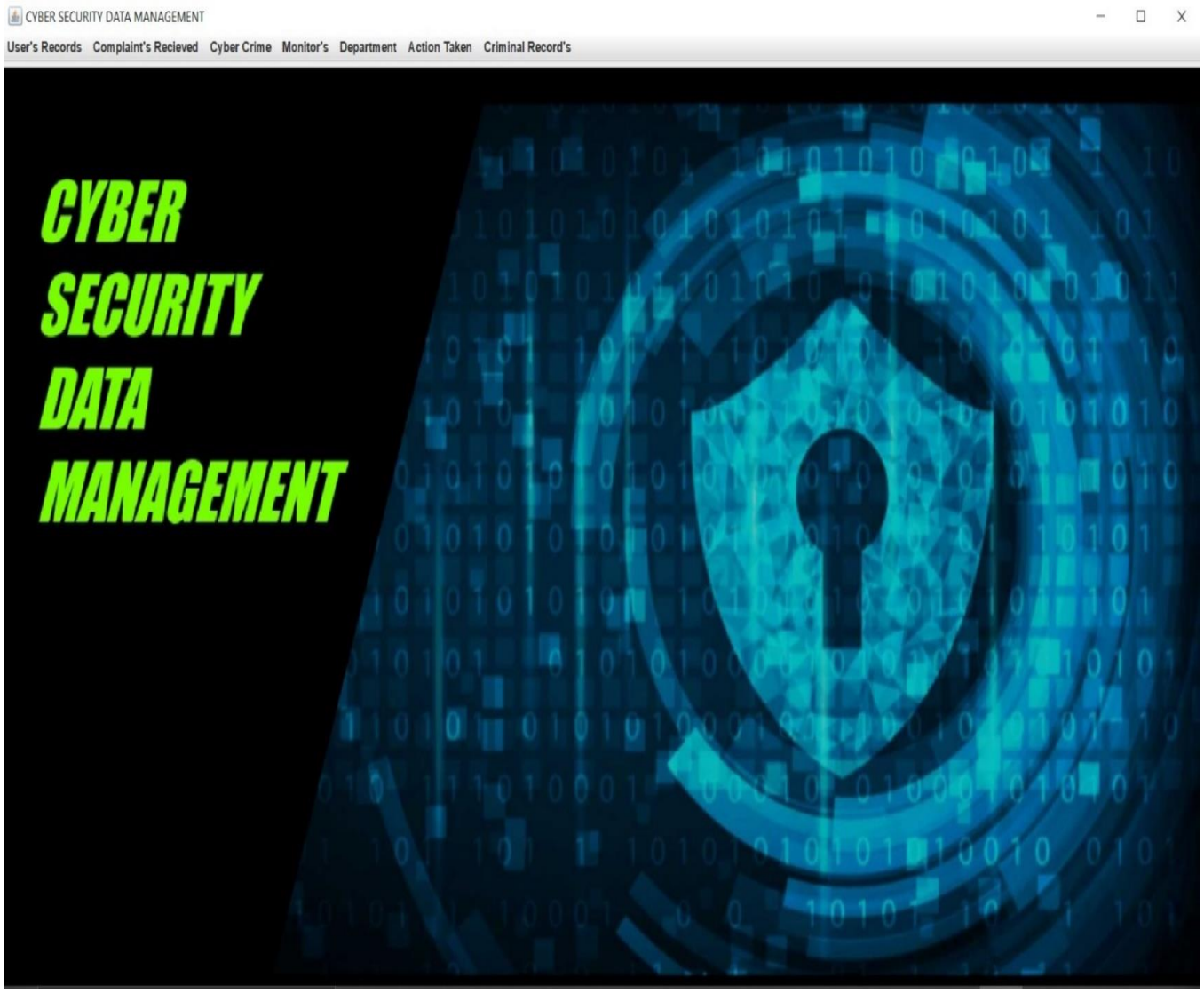
Insert users:

UPDATE USERS:



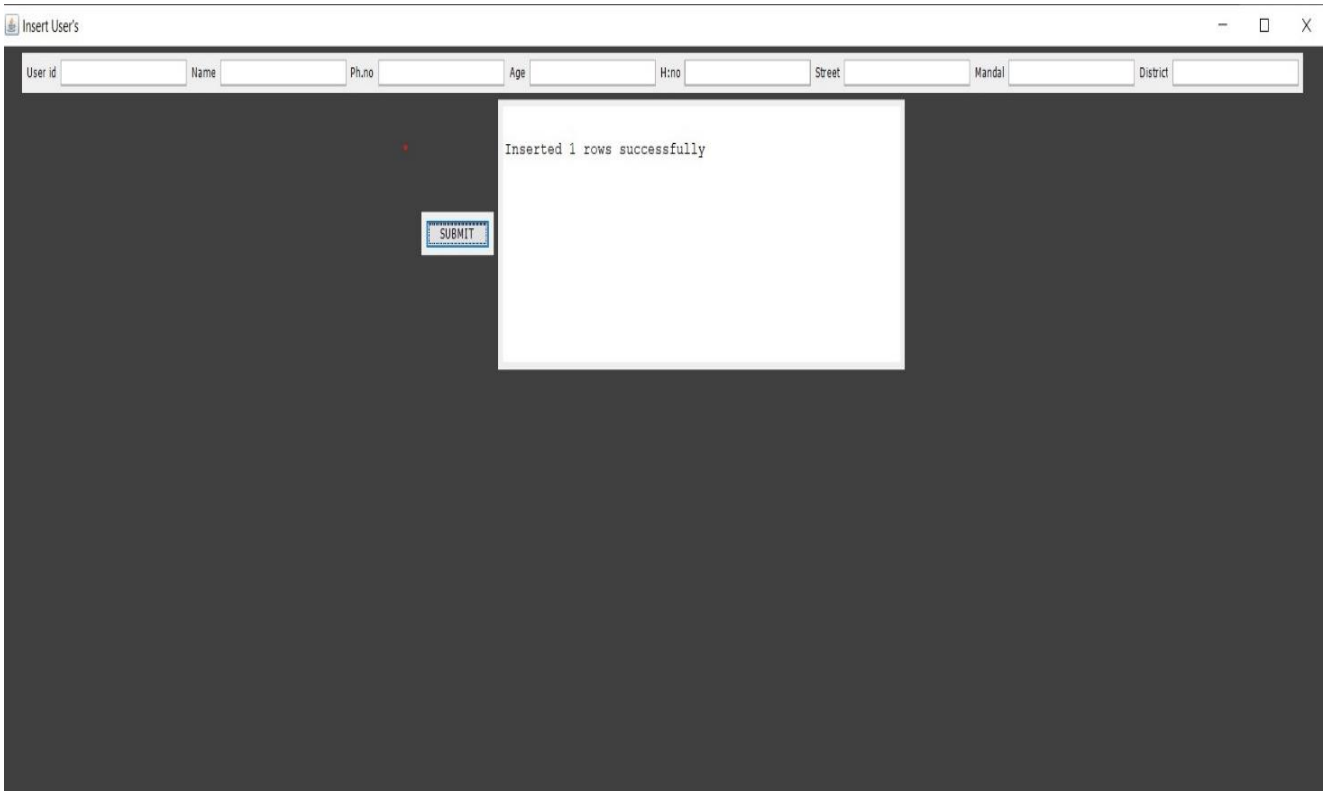
RESULTS:

HOME UI:



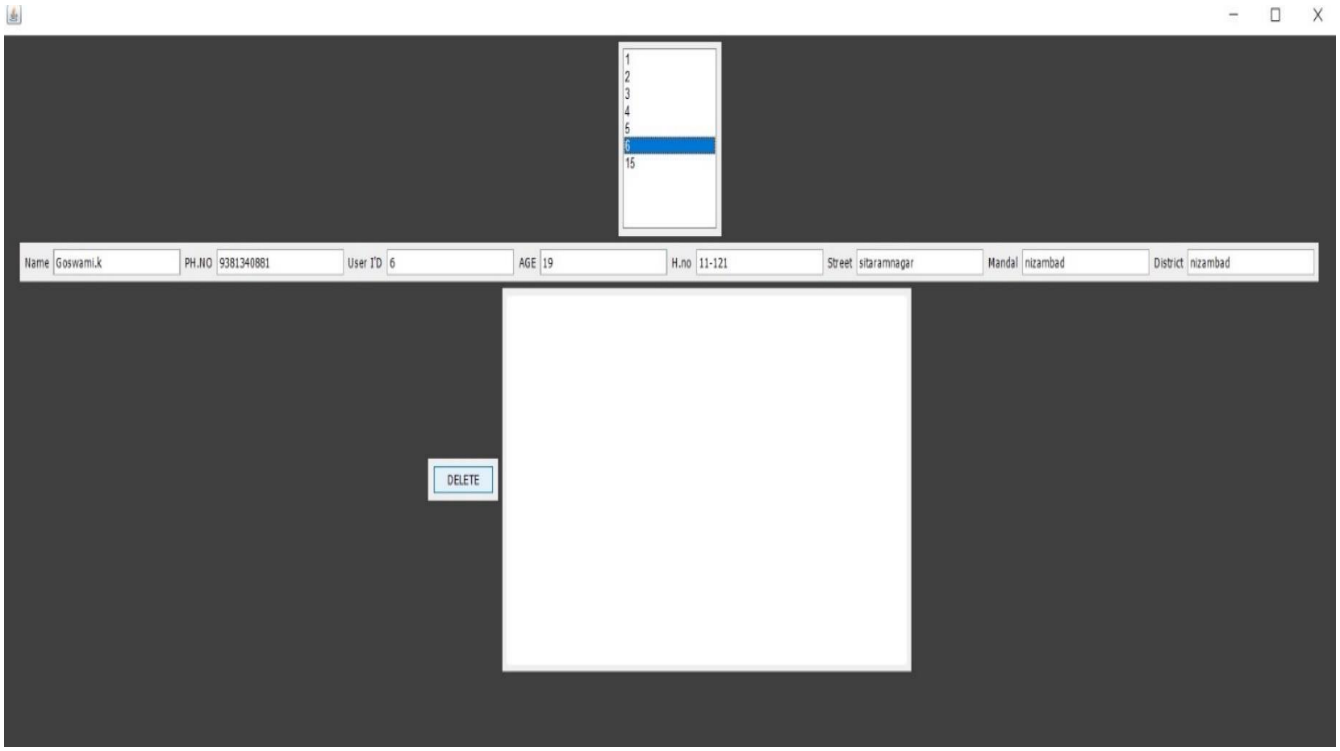
USERS_RECORD:

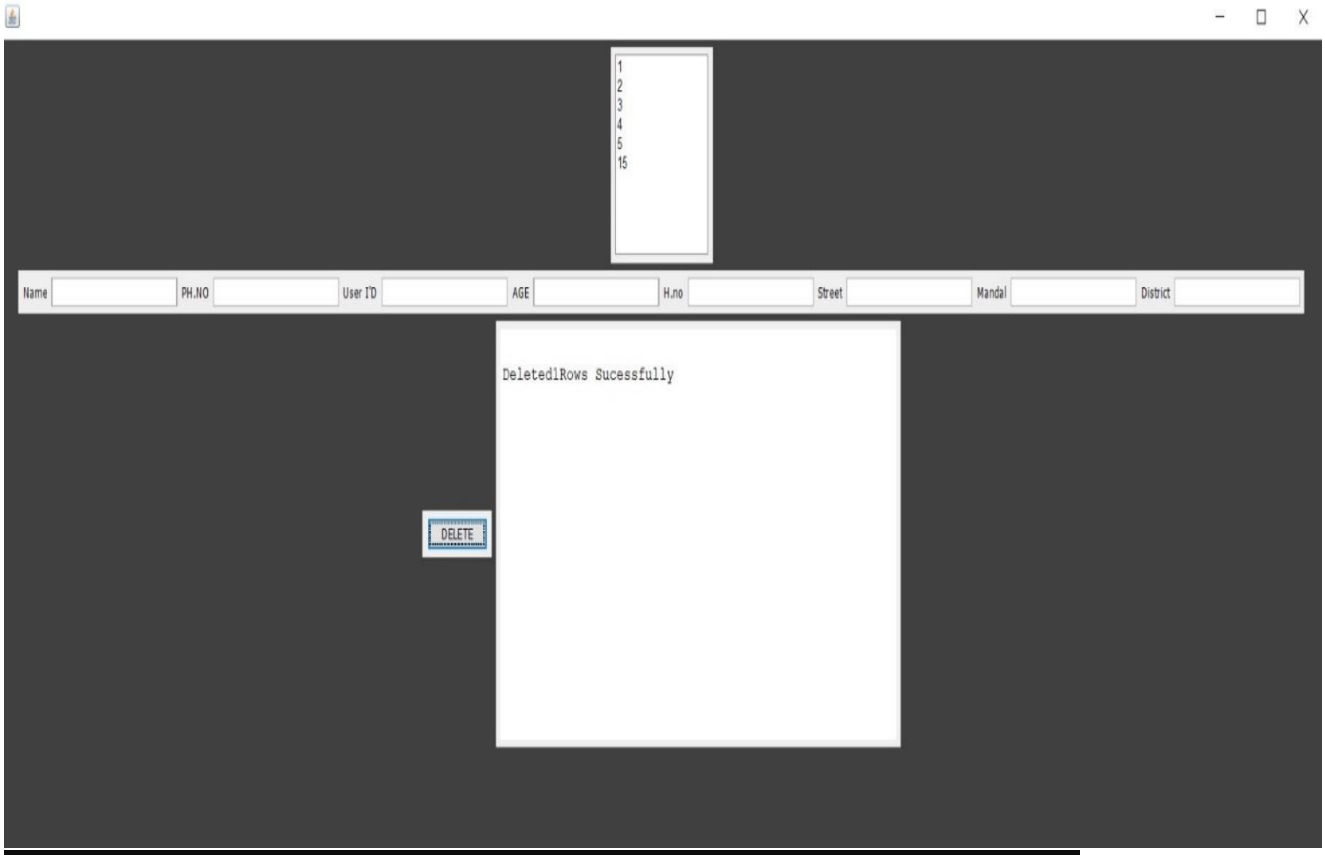
1.INSERT_USERS



```
Connected.
SQL> select *from users_record;

NAME                      PHNO USID                 AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
niteesh              6303520640 1                    19 1-2/a
nandipahad      miryalaguda     nalagonda

khushi               9346498993 2                    18 12-5-505/17
kantinagar      mehadipatnam    hyderabad

mohith               6302560420 3                    19 12-133/6
krishna colony  miryalaguda     nalgonda


NAME                      PHNO USID                 AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
badri                9246897210 4                    19 15-155
kailashnagar    devarakonda     nalagonda

kranthi              9864321102 5                    22 12-33/s
donagalguda     moosapet        rangareddy

Goswami.k            9381340881 6                    19 11-121
sitaramnagar    nizambad        nizambad


NAME                      PHNO USID                 AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
mahendra             8897110606 15                   19 11-115
seetharampuram  miryalaguda     nalgonda


7 rows selected.
```

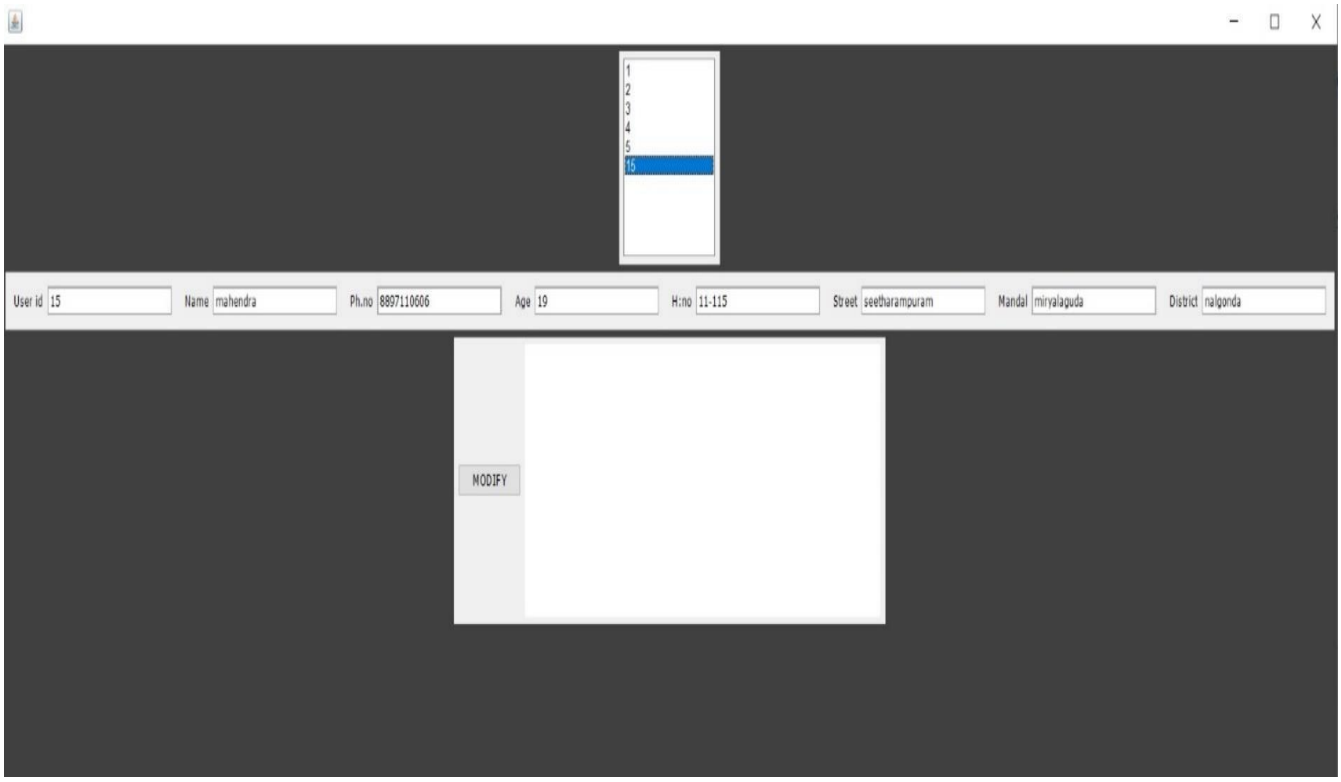
DELETE USERS:

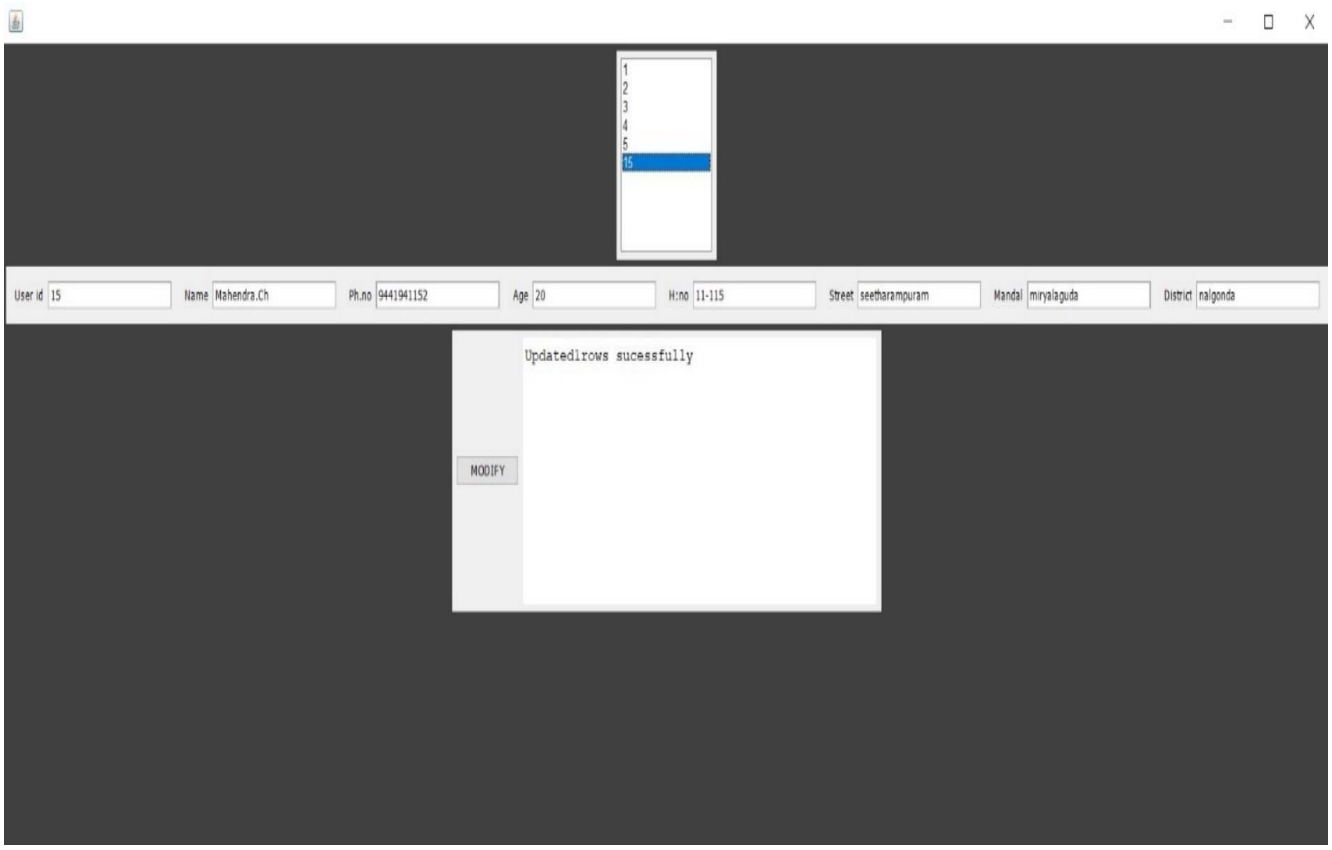





```
SQL> select *from users_record;

NAME                      PHNO USID                 AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
niteesh              6303520640 1                    19 1-2/a
nandipahad      miryalaguda     nalagonda

khushi               9346498993 2                    18 12-5-505/17
kantinagar      mehadipatnam    hyderabad

mohith               6302560420 3                    19 12-133/6
krishna colony  miryalaguda     nalgonda


NAME                      PHNO USID                 AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
badri                9246897210 4                    19 15-155
kailashnagar    devarakonda     nalagonda

kranthi              9864321102 5                    22 12-33/s
donagalguda     moosapet        rangareddy

mahendra             8897110606 15                   19 11-115
seetharampuram  miryalaguda     nalgonda


6 rows selected.
```

MODIFY USERS:

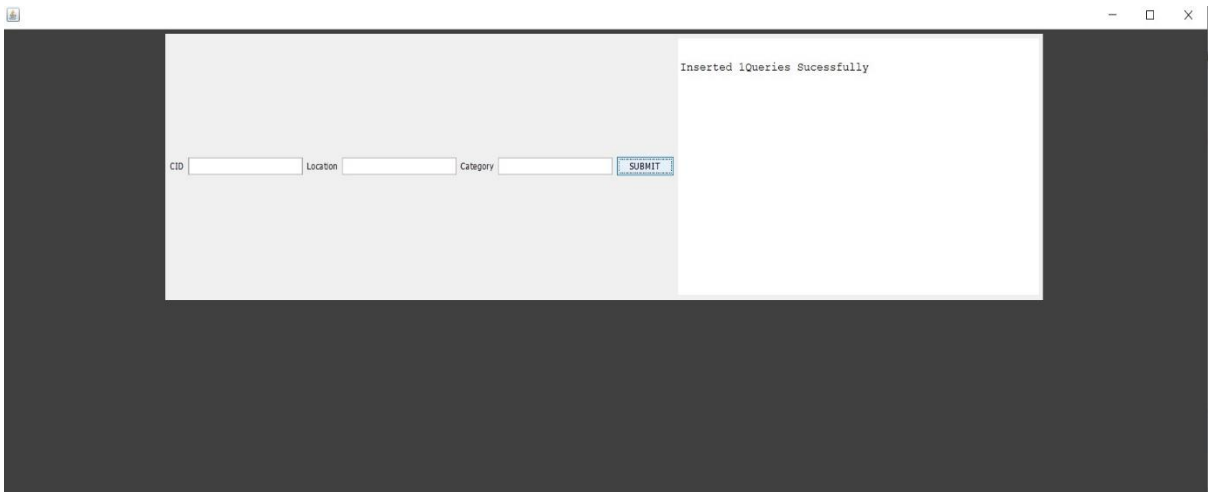




```
SQL> select *from users_record;

NAME                      PHNO USID                    AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
niteesh              6303520640 1                        19 1-2/a
nandipahad      miryalaguda     nalagonda

khushi               9346498993 2                        18 12-5-505/17
kantinagar      mehadipatnam    hyderabad

mohith               6302560420 3                        19 12-133/6
krishna colony  miryalaguda     nalgonda


NAME                      PHNO USID                    AGE HNO
------------------- ---------- --------------- ---------- ---------------
STREET          MANDAL          DISTRICT
--------------- --------------- ---------------
badri                9246897210 4                        19 15-155
kailashnagar    devarakonda     nalagonda

kranthi              9864321102 5                        22 12-33/s
donagalguda     moosapet        rangareddy

Mahendra.Ch          9441941152 15                       20 11-115
seetharampuram  miryalaguda     nalgonda


6 rows selected.
```

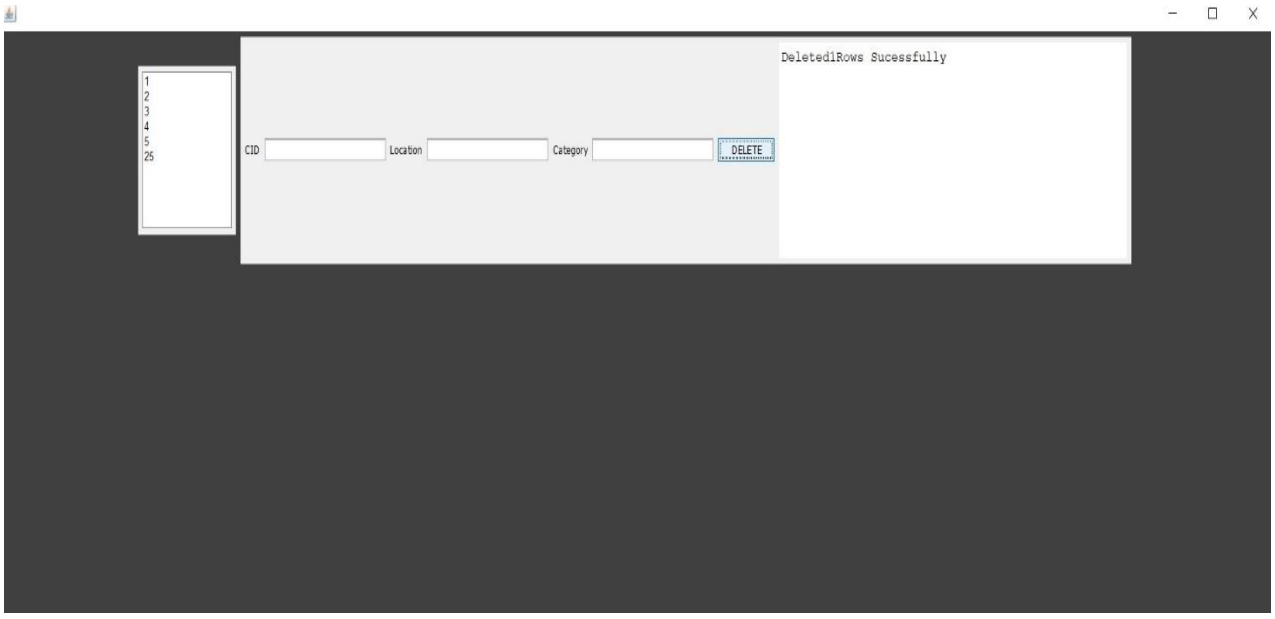
CYBER_CRIME:

INSERT_CYBER:



```
6 rows selected.

SQL> select *from cyber_crime;

CID        LOCATION                CATEGORY
---------- ----------------------- ---------------
1          asif nagar              banking
2          asif nagar              hospitals
3          cyberabad               mobile
4          lbnagar                 mobile
5          secundrabad             mobile_data
15         jb                      h

6 rows selected.
```

DELETE USERS:



```
SQL> select *from cyber_crime;

CID        LOCATION                  CATEGORY
---------- ------------------------- --------------
1          asif nagar                banking
2          asif nagar                hospitals
3          cyberabad                 mobile
4          lbnagar                   mobile
5          secundrabad               mobile_data
25         malakpet                  mobiletheft

6 rows selected.
```

MODIFY CYBERCRIME:


```
SQL> select *from cyber_crime;

CID        LOCATION                  CATEGORY
---------- ------------------------- --------------
1          asif nagar                banking
2          asif nagar                hospitals
3          cyberabad                 mobile
4          lbnagar                   mobile
5          secundrabad               mobile_data
25         shadnagar                 mobiletheft

6 rows selected.
```

EXIT:


CYBER SECURITY DATA MANAGEMENT
User's Records
Complaint's Recieved
Cyber Crime
Monitor's
Department
Action Taken
Criminal Record's
CYBER
SECURITY
DATA
MANAGEMENT
!!THIS CLOSES ENTERING DATA!!
DO YOU WANT TO EXIT?
Yes
No

DISCUSSIONS and FUTURE WORK:

After this mini project “CYBER CRIME DATA MANAGEMENT” I want to do a project based on android like implementing this work in an android app which makes us to use more freely entering data. And also we can sync this to cloud and can keep passcode for security and ease of access data .

I want to work on many ideations projects. Like swayam and some other colleges hackathons .

We can make more use of artificial intelligence and machine learning so I want to make a project based on them which is useful to mankind in many ways . people now a days are facing many issues on crimes related to cyber so I want to make a better place for people to use every app in a better and a easy way to provide secure access to internet in these days

I will work to increase their faith on internet by and try to reduce cyber crimes.

REFERENCES:

- https://docs.oracle.com/javase/7/docs/api/
- https://www.javatpoint.com/dbms-tutorial
- https://en.wikipedia.org/wiki/Computer_security
- DATA SYSTEM CONCEPTS BY: Henry F korth.
- DATA MANAGEMENT SYSTEMS BY: Raghu Ramakrishna.