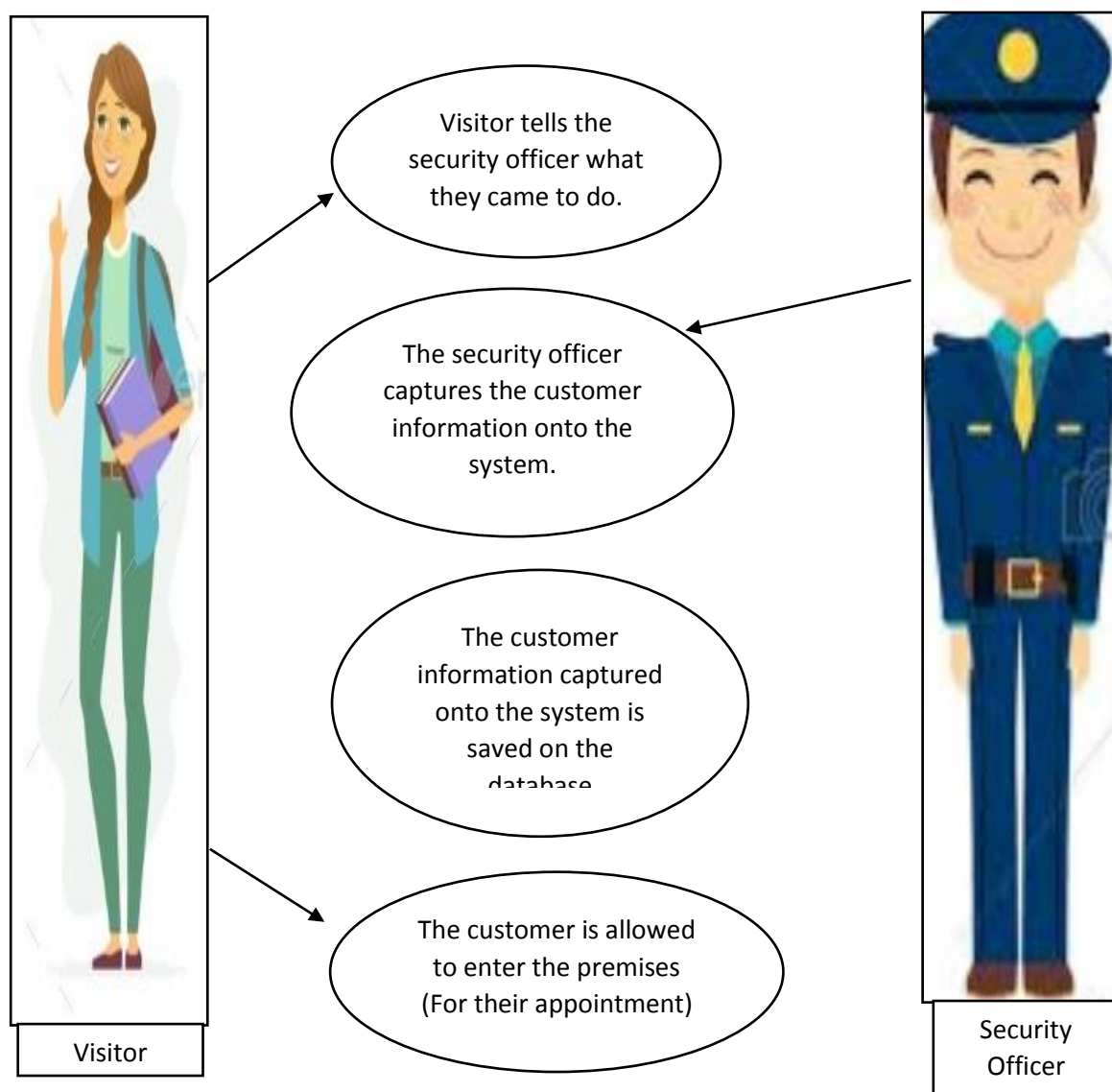


Name : Sandra Susan
Surname : Sibanda
Student Number : 18003804
Support Centre : Randburg
Summative Assessment 1 : Systems Development 2A (HSYD2A61)

QUESTION 1.

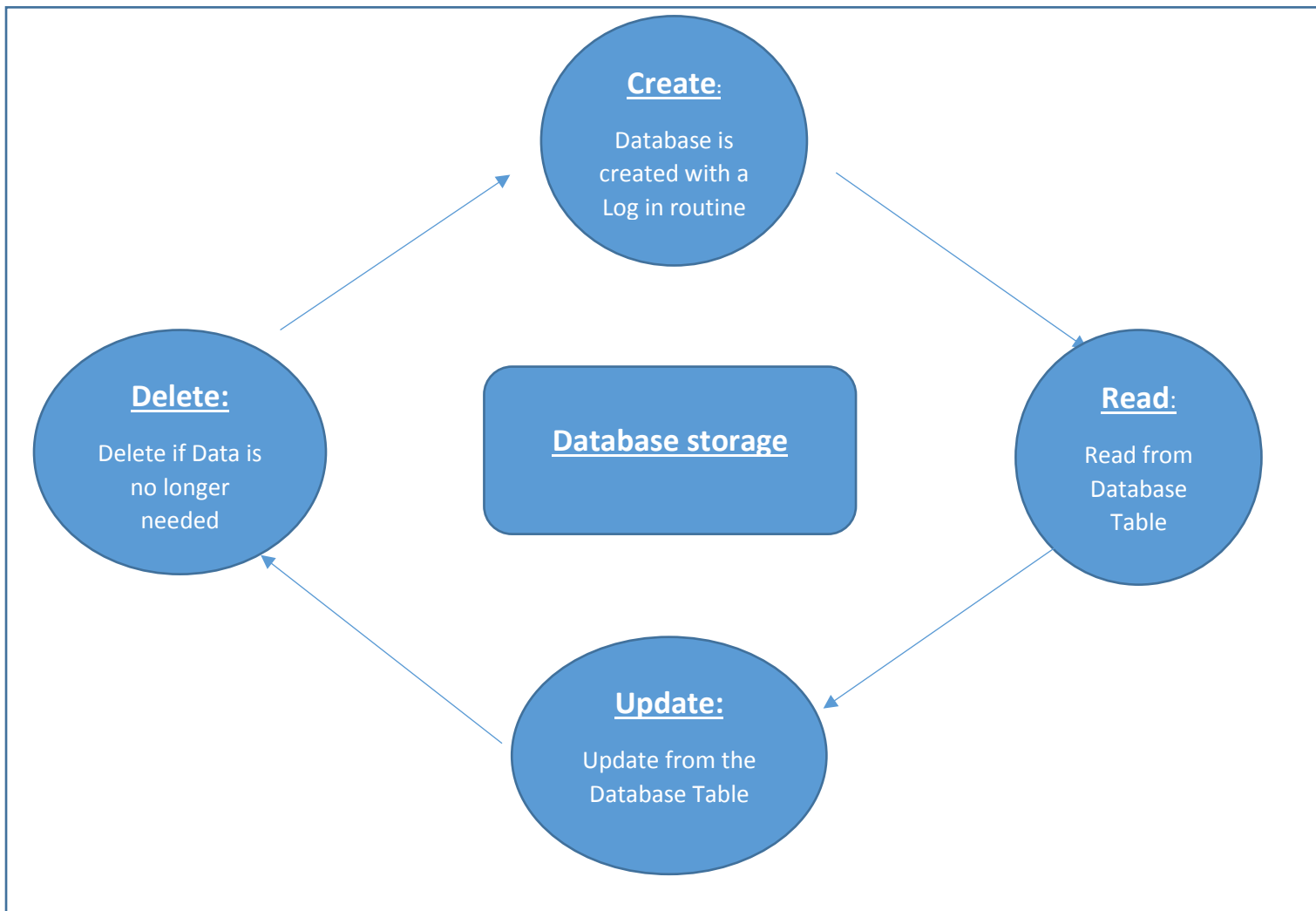
Khusela Securities used to capture visitor's information in a hard copy book. The manager would like this information to be now stored electronically into their database for future references. The CRUD cycle will be used to fulfil this purpose and below is the use case model diagram that shows how the system will work.



The Khusela Security system will use the CRUD cycle shown below.

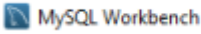
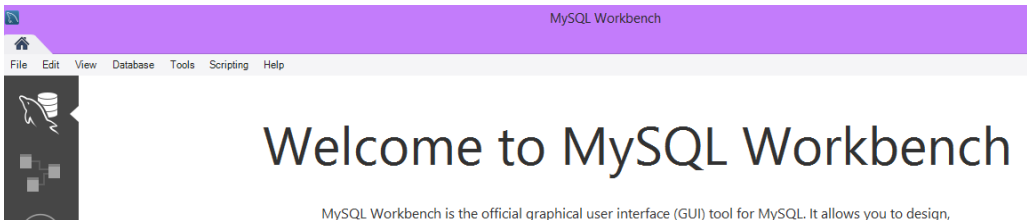
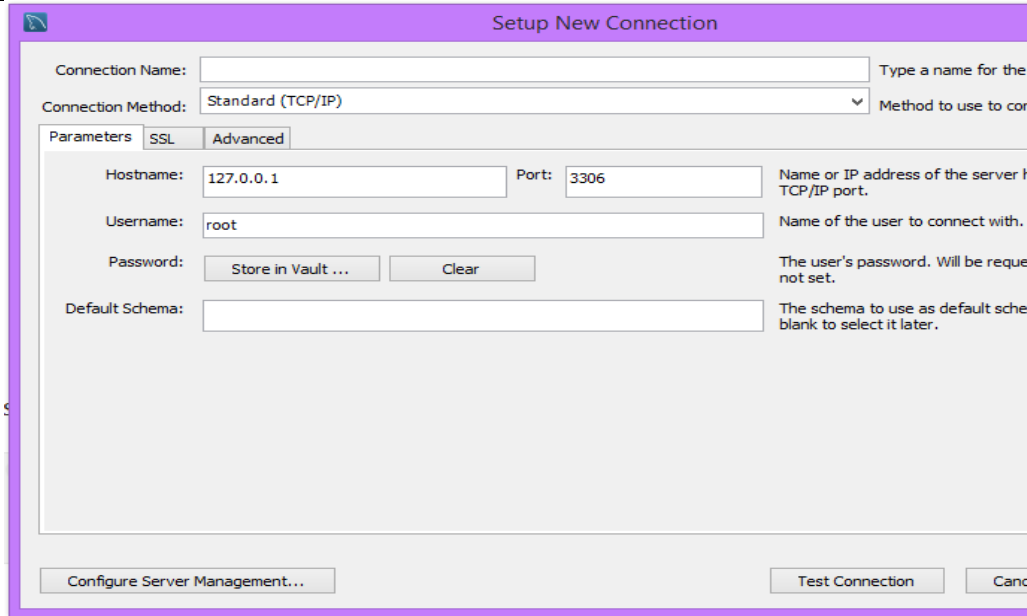
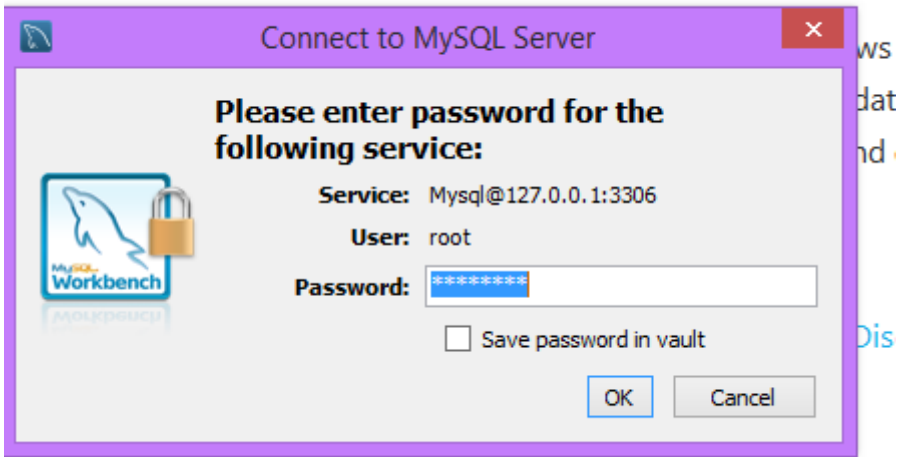
The CRUD (CREATE, READ, UPDATE, DELETE) is a full data cycle where a full cycle is followed. During this process, the data is:

- Always created first.
- Saved to the database.
- Retrieved from the database table.
- Updated from the database table.
- Deleted if data is no longer needed.

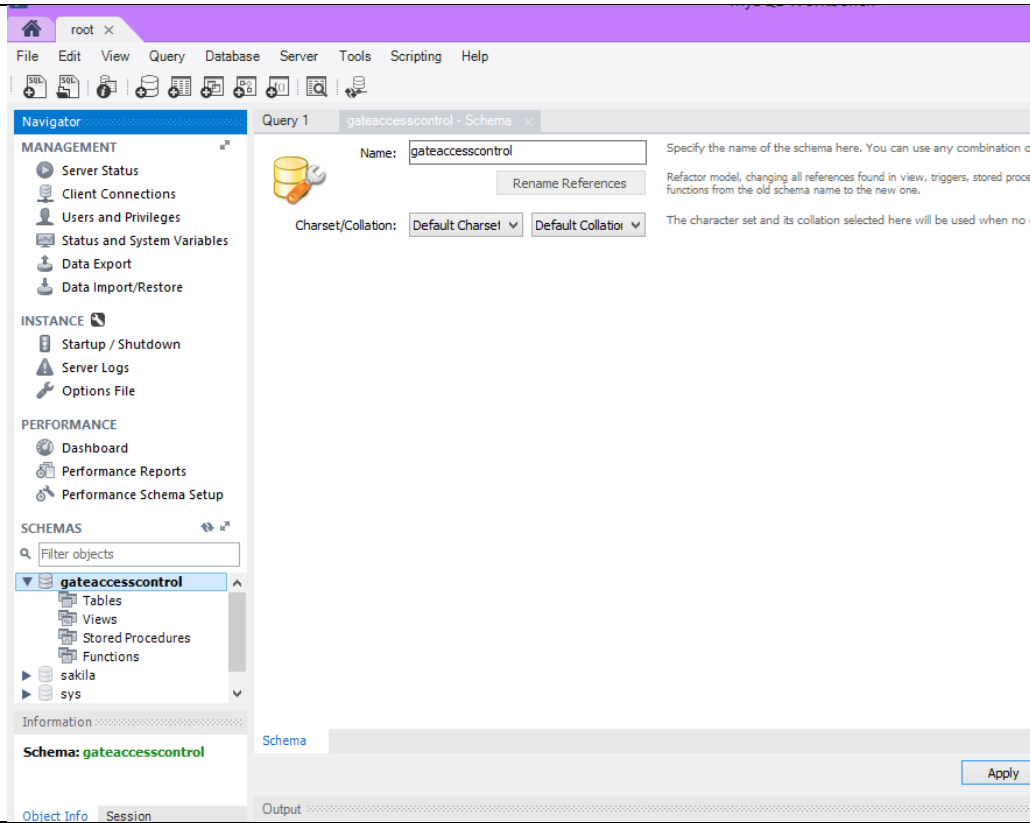


QUESTION 2.

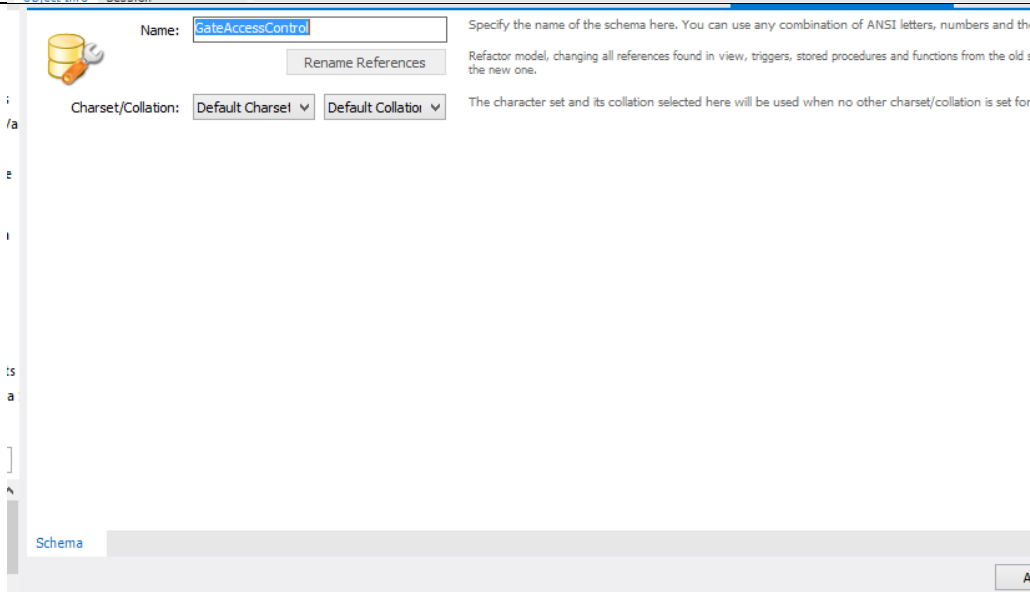
Create a database called Gate Access Control using MySQL.

1. Open the workbench on the local pc.	
2. Right click on Database and select "Connect to Database".	
3. The Connect to Database window will appear. Click ok to accept the root username (I set my username as root).	
4. The connect to MySQL server window will also appear requesting a password. Use "Password" as the password.	

5. Select the “create new schema” indicated in the screenshot. Name the new schema “GateAccessControl”



7. Click on apply to run the query.



8. The review SQL Script window will appear click apply again

Review SQL Script

Apply SQL Script

Review the SQL Script to be Applied on the Database

Online DDL

Algorithm: Default Lock Type: Default

1
2

CREATE SCHEMA `gateaccesscontrol` ;

Back

Apply

9. The query will execute then click Finish

Review SQL Script

Apply SQL Script

Applying SQL script to the database

The following tasks will now be executed. Please monitor the execution. Press Show Logs to see the execution logs.

☒ Execute SQL Statements

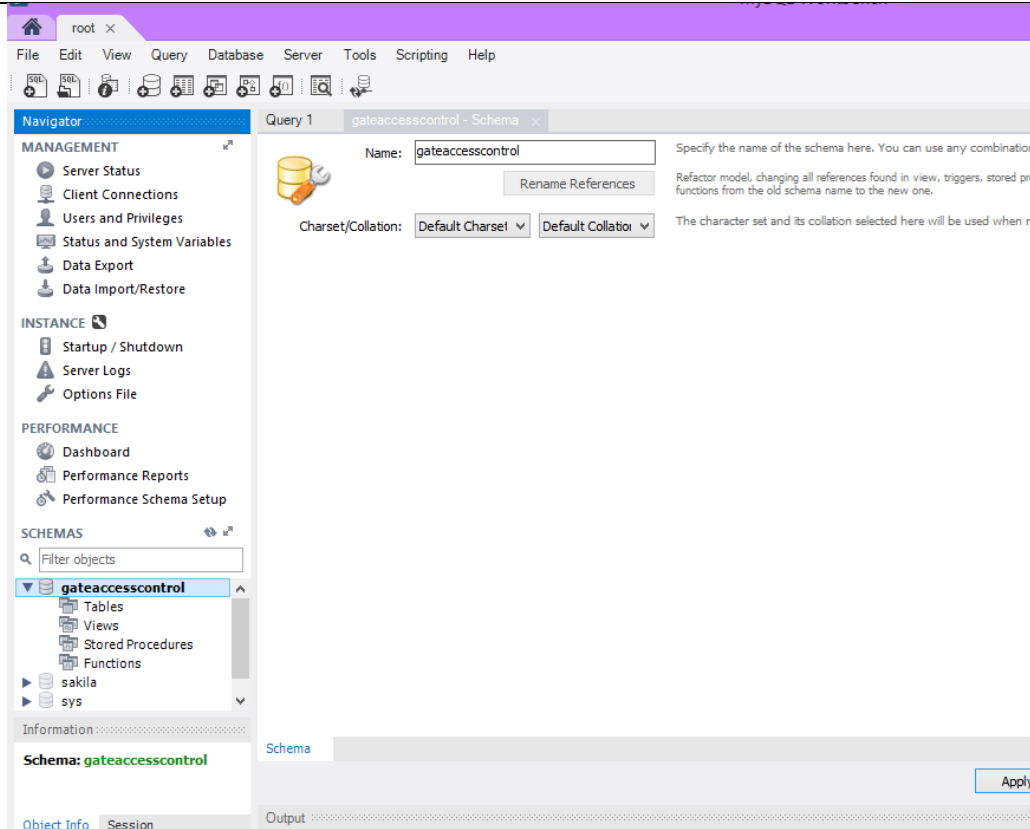
SQL script was successfully applied to the database.

Show Logs

Back

Finish

10. At the bottom left corner under SCHEMAS, select the schema "GateAccessControl" and right click on it to make it a default schema.



11. Select the “create new table” and name the new table as “Visitors” with 9 columns named-

VisitorID (INT, PK, NN,
UQ, AI)
Date(Date
TimeIn (DateTime, Not Null
, UQ)
Name(Varchar(255) NN,
UQ)
From
Company(Varchar(255) Not
Null(NN), Unique (UQ))
Vehicle Registration
Number(Varchar(255 Not
Null(NN), Unique (UQ))
Telephone (INT
Reason For
Visit(Varchar(255) Not
Null(NN), Unique (UQ))
Person To See
(Varchar(255), Not
Null(NN), Unique (UQ))

11. Select the “create new table” and name the new table as “Visitors” with 9 columns named-

VisitorID (INT, PK, NN,
UQ, AI)
Date(Date
TimeIn (DateTime, Not Null
, UQ)
Name(Varchar(255) NN,
UQ)
From
Company(Varchar(255) Not
Null(NN), Unique (UQ))
Vehicle Registration
Number(Varchar(255 Not
Null(NN), Unique (UQ))
Telephone (INT
Reason For
Visit(Varchar(255) Not
Null(NN), Unique (UQ))
Person To See
(Varchar(255), Not
Null(NN), Unique (UQ))

[illegible]

12. The review window will appear then click apply and the query will execute successfully.

Review SQL Script

Apply SQL Script

Review the SQL Script to be Applied on the Database

Online DDL

Algorithm: Default

Lock Type: Default

```
1 CREATE TABLE `gateaccesscontrol`.`visitors` (  
2   `VisitorID` INT NOT NULL AUTO_INCREMENT,  
3   `Date` DATE NOT NULL,  
4   `TimeIn` DATETIME NOT NULL,  
5   `Name` VARCHAR(255) NOT NULL,  
6   `From Company` VARCHAR(255) NOT NULL,  
7   `Vehicle Registration Number` VARCHAR(255) NOT NULL,  
8   `Telephone Number` INT NOT NULL,  
9   `Reason For Visit` VARCHAR(255) NOT NULL,  
10  `Person To See` VARCHAR(255) NOT NULL,  
11  PRIMARY KEY (`VisitorID`),  
12  UNIQUE INDEX `VisitorID_UNIQUE` (`VisitorID` ASC) VISIBLE,  
13  UNIQUE INDEX `Date_UNIQUE` (`Date` ASC) VISIBLE,  
14  UNIQUE INDEX `TimeIn_UNIQUE` (`TimeIn` ASC) VISIBLE,  
15  UNIQUE INDEX `Name_UNIQUE` (`Name` ASC) VISIBLE,  
16  UNIQUE INDEX `From Company_UNIQUE` (`From Company` ASC) VISIBLE,  
17  UNIQUE INDEX `Vehicle Registration Number_UNIQUE` (`Vehicle Registration Number` ASC) VISIBLE,  
18  UNIQUE INDEX `Telephone Number_UNIQUE` (`Telephone Number` ASC) VISIBLE,  
19  UNIQUE INDEX `Reason For Visit_UNIQUE` (`Reason For Visit` ASC) VISIBLE,  
20  UNIQUE INDEX `Person To See_UNIQUE` (`Person To See` ASC) VISIBLE,  
21
```

Back

Apply

13. Click on Finish

Review SQL Script

Apply SQL Script

Applying SQL script to the database

The following tasks will now be executed. Please monitor the execution. Press Show Logs to see the execution logs.

☒ Execute SQL Statements

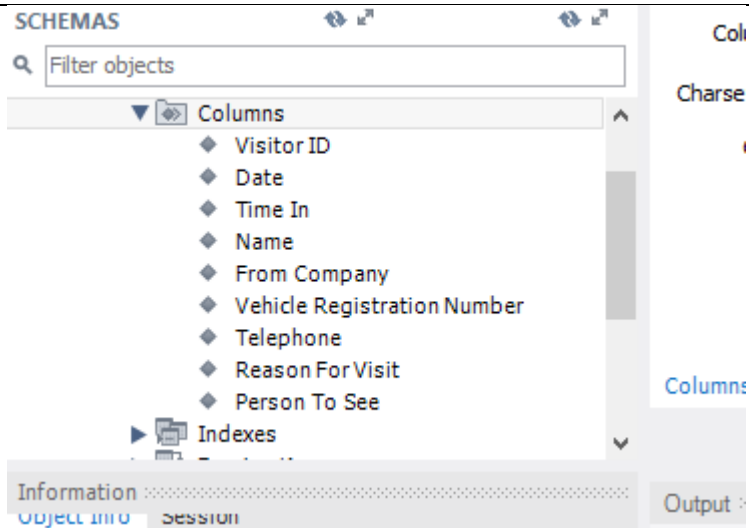
SQL script was successfully applied to the database.

Show Logs

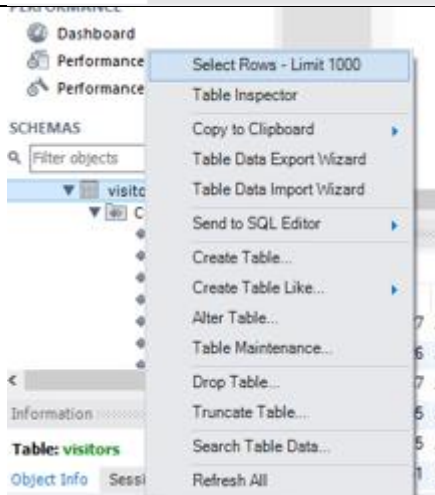
Back

Finish

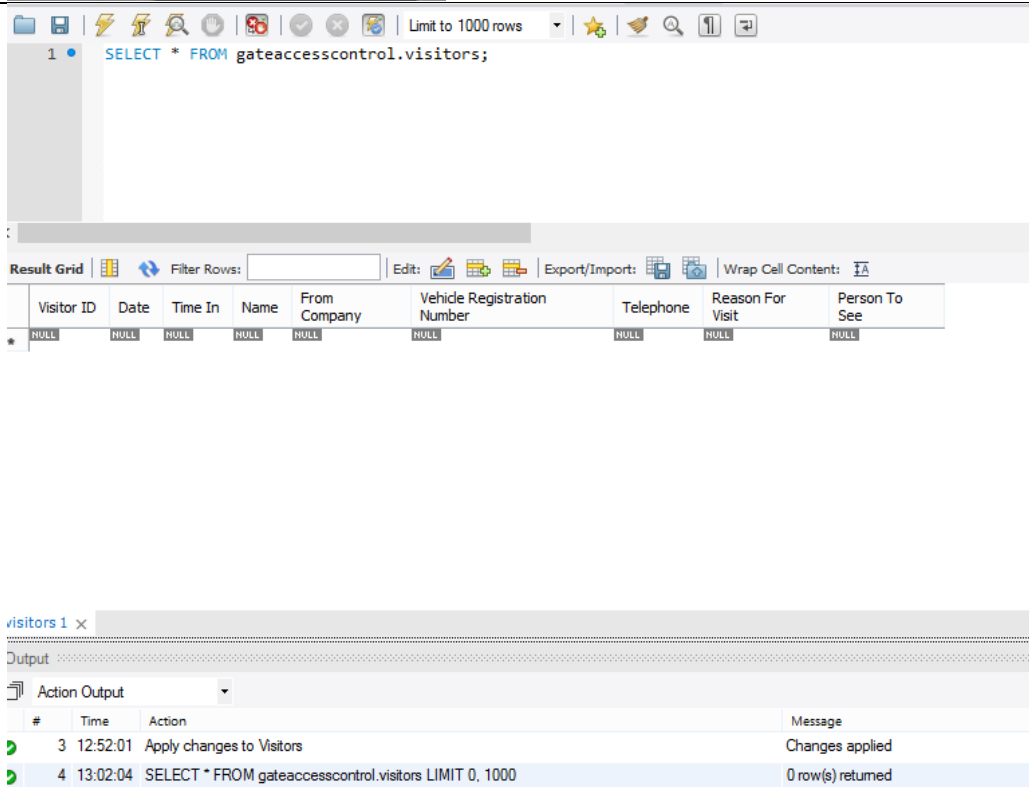
14. On schemas, expand the GateAccessControl Schema



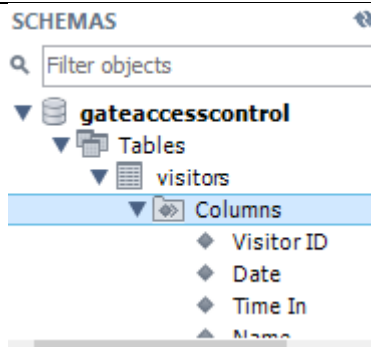
15. Right click on the Visitors and select Rows-Limit 1000.



16. The columns will be shown on the tab.



17. The database is now created with a table called Visitors inside it



We will create another table called Users that will store the information of the manager and the security to allow them to log in to the system that will be created using netbeans ide.

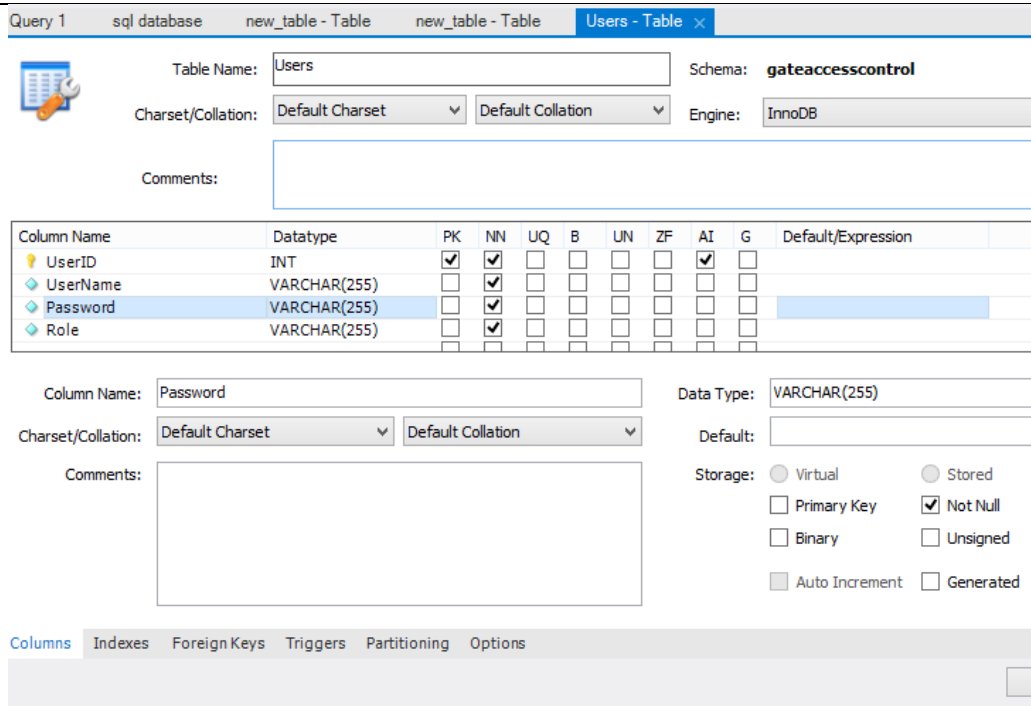
18. With the GateAccessControl schema still open and set as the default schema, create a new table called "Users" with the following column names-
UserID (INT (Primary key(PK),NOT NULL(NN), AUTO INCREMENT(AI))

UserName (Varchar(255),
NOT NULL(NN))

Password
(Varchar(255),Not
Null(NN))

Role (Varchar(255), Not Null(NN))

After inserting the column names click apply



19. click on apply to execute the query

Review SQL Script

Apply SQL Script

Review the SQL Script to be Applied on the Database

Online DDL

Algorithm: Default

Lock Type: Default

```
1 CREATE TABLE `gateaccesscontrol`.`users` (  
2   `UserID` INT NOT NULL AUTO_INCREMENT,  
3   `UserName` VARCHAR(255) NOT NULL,  
4   `Password` VARCHAR(255) NOT NULL,  
5   `Role` VARCHAR(255) NOT NULL,  
6   PRIMARY KEY (`UserID`));  
7
```

Back

Apply

20. The query will execute successfully. Click finish

Review SQL Script

Apply SQL Script

Applying SQL script to the database

The following tasks will now be executed. Please monitor the execution. Press Show Logs to see the execution logs.

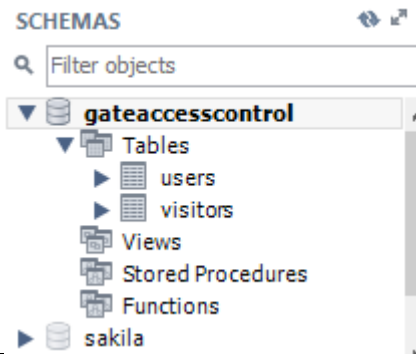
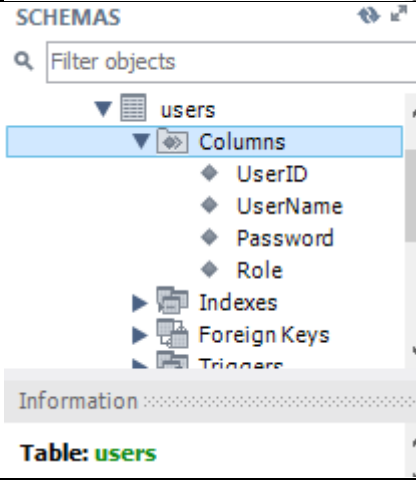
☒ Execute SQL Statements

SQL script was successfully applied to the database.

Show Logs

Back

Finish

<p>Go to Schemas and check the table users has been created.</p>		
<p>22. expand table users to view the column names</p>		
<p>Save the schema and close the work bench.</p>	<p>To be zipped and sent with the project as zip file.</p>	

We are now creating a project called Khusela Security Company system.

1. Open Netbeans IDE and create a project called 'KhuselaSecurityCompanySystem'

New Java Application

Steps

1. Choose Project
2. **Name and Location**

Name and Location

Project Name:

Project Location:

Project Folder:

☒ Use Dedicated Folder for Storing Libraries

Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

☒ Create Main Class

< Back Next > Finish Cancel

2. The project is created

```

-  /*
   * To change this license header, choose License Headers in Project Properties
   * To change this template file, choose Tools | Templates
   * and open the template in the editor.
   */
package khuselasecuritycompanysystem;

-  /**
   *
   * @author mahlobo
   */
public class KhuselaSecurityCompanySystem {

-    /**
     * @param args the command line arguments
     */
-    public static void main(String[] args) {

    }

}
}
```

3. Create a JFrame form inside the project and rename it to "frmLogin"

New JFrame Form

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > Finish Cancel Help

4. Create a JFrame form and name it frmMain

New JFrame Form

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > Finish Cancel Help

5. Add a JInternalFrame form and name it to frmUsers

New JInternalFrame Form

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

Project:

Location:

Package:

Created File:

< Back Next > **Finish** Cancel

6. Add a JInternalFrame and name it frmVisitors

New JInternalFrame Form

Steps

1. Choose File Type
2. **Name and Location**

Name and Location

Class Name:

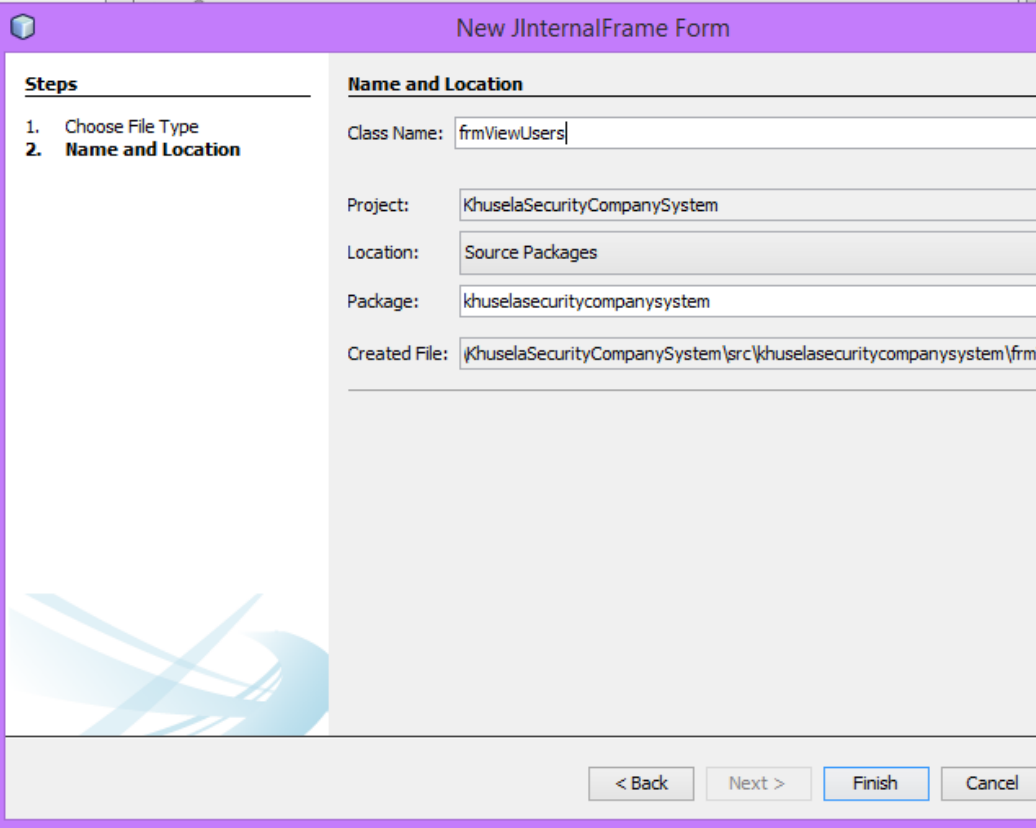
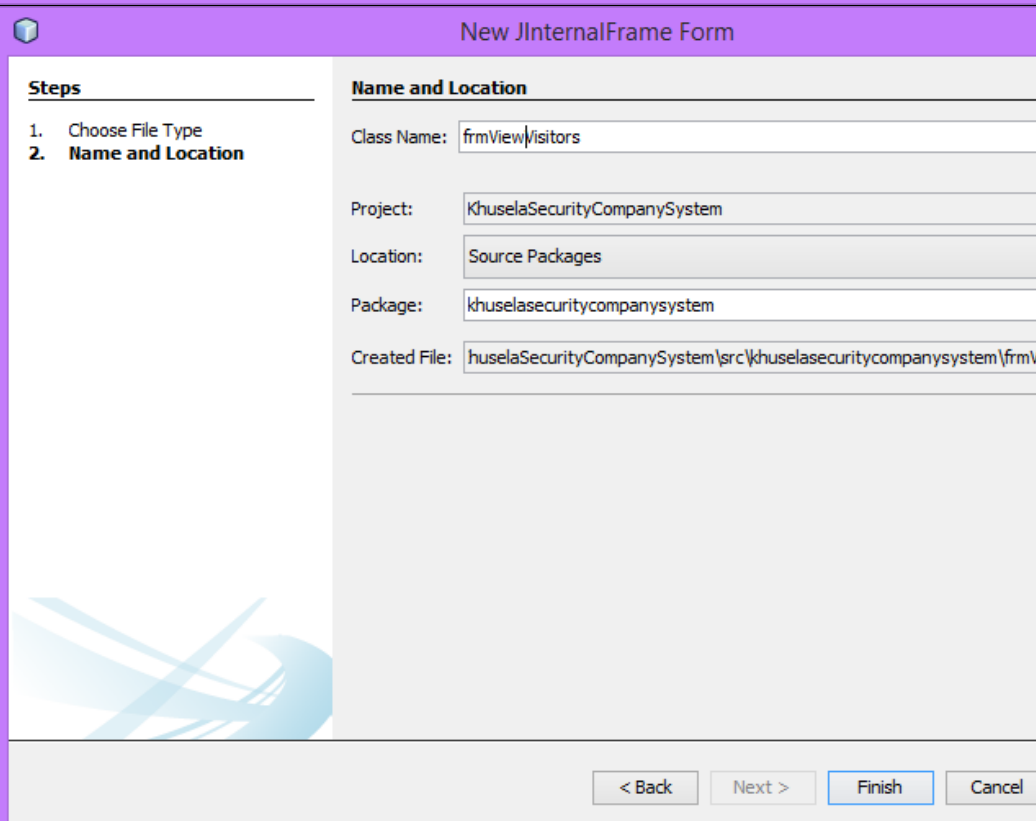


Project:


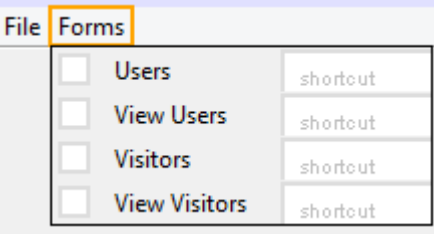
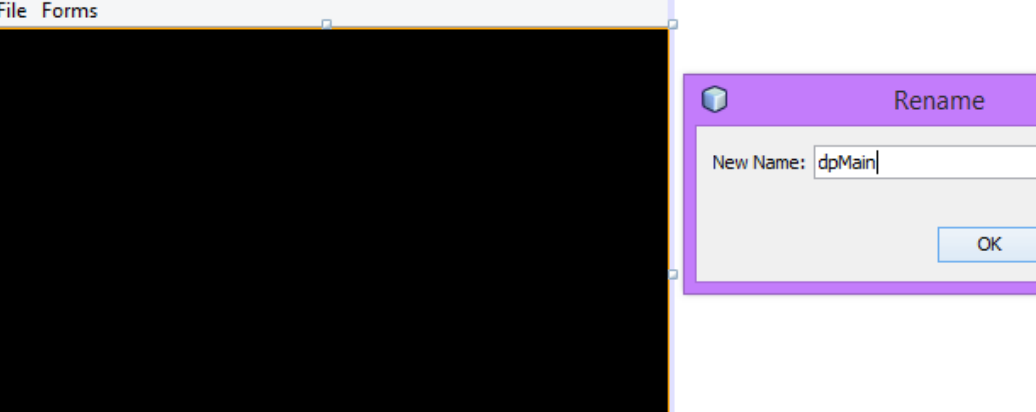
Location:

Package:

Created File:

< Back Next > **Finish** Cancel

<p>7. Add a JInternalFrame and name it frmViewUsers</p>	
<p>8. Add a JInternalFrame and name it frmViewVisitors.</p>	
<p>9. Select the “frmMain” tab</p>	
<p>10. Add a Menubar, menu and name them: File -mnuFile Forms - mnuForms</p>	

<p>11. Add menu items to the File menu and name them: LogOff - mnuLogOff Exit - mnuExit</p>	
<p>12. Add menu items to the Form menu and name them: Users - mnuUsers ViewUsers - mnuViewUsers Visitors- mnuVisitors View Visitors - mnuViewVisitors</p>	
<p>13. Add a desktop pane to the frmMain and change the variable name to dpMain then fill the dpMain frame to the pane of the whole window.</p>	
<p>14. Double click on the mnuLogOff menu item and put the code</p>	<pre>private void mnuLogOffActionPerformed(java.awt.event.ActionEvent e) { frmLogin frmL=new frmLogin(); frmL.setVisible(true); this.setVisible(false); }</pre>
<p>15. Double click on the mnuExit menu item and put the code</p>	<pre>private void mnuExitActionPerformed(java.awt.event.ActionEvent e) { System.exit(0); }</pre>
<p>16. Double click on the mnuUsers menu item and put the code.</p>	<pre>private void mnuUsersActionPerformed(java.awt.event.ActionEvent e) { frmUsers frmU=new frmUsers(); dpMain.add(frmU); frmU.setVisible(true); }</pre>
<p>17. Double click on the mnuViewUsers and put the code.</p>	<pre>private void mnuViewUsersActionPerformed(java.awt.event.ActionEvent e) { frmViewUsers frmVU = new frmViewUsers(); dpMain.add(frmVU); frmVU.setVisible(true); }</pre>
<p>18. Double click on the mnuVisitors and put the code.</p>	<pre>private void mnuVisitorsActionPerformed(java.awt.event.ActionEvent e) { frmVisitors frmV = new frmVisitors(); dpMain.add(frmV); frmV.setVisible(true); }</pre>

19. Double click the mnuViewVisitors and put the code.

```
private void mnuViewVisitorsActionPerformed(java.awt.event.ActionEvent  
    frmViewVisitors frmVV = new frmViewVisitors();  
    dpMain.add(frmVV);  
    frmVV.setVisible(true);  
}
```

20. On frmVisitors, click the design tab and add the following controls and space them accordingly. Name them as per screenshot and add the variable names accordingly.

- Combo Box (cboName)
- Eight Labels
 1. Date (lblDate)
 2. Time In (lblTimeIn)
 3. Name (lblName)
 4. From Company (lblFromCompany)
 5. Vehicle Registration Number (lblVehicleRegistrationNumber)
 6. Telephone Number (lblTelephoneNumber)
 7. Reason For Visit (lblReasonToVisit)
 8. Person To See (lblPersonToSee)
- Eight Text Fields
 1. txtDate
 2. txtTimeIn
 3. txtName
 4. txtFromCompany
 5. txtVehicleRegistrationNumber
 6. txtTelephoneNumber
 7. txtReasonForVisit
 8. txtPersonToSee

The screenshot shows a Java Swing window titled 'frmVisitors'. At the top is a JComboBox. Below it are eight labels and text input fields arranged in a list: 'Date', 'Time In', 'Name', 'From Company', 'Vehicle Registration Number', 'Telephone Number', 'Reason For Visit', and 'Person To See'. Each label is followed by a text input field. At the bottom of the form are five buttons: 'CREATE', 'LOAD', 'EDIT', 'SAVE', and 'DELETE'.

<ul style="list-style-type: none"> Five Buttons <ol style="list-style-type: none"> 1. CREATE (btnCreate) 2. LOAD (btnLoad) 3. EDIT (btnEdit) 4. SAVE (btnSave) 5. DELETE (btnDelete) 	
<p>21. Click the source tab and insert the following code underneath the package name.</p>	<pre>import java.awt.HeadlessException; import java.sql.DriverManager; import java.sql.ResultSet; import java.sql.SQLException; import java.sql.Statement; import java.text.DateFormat; import java.text.SimpleDateFormat; import java.util.Date; import javax.swing.JOptionPane;</pre>
<p>22. Locate the method frmVisitors and declare the variables as shown.</p>	<pre>Boolean boolRecordExists = false; Boolean boolEdit = false; Boolean boolCreate = false; String strDate; String strTimeIn; String strName; String strFromCompany; String strVehicleRegistrationNumber; String strTelephoneNumber; String strReasonForVisit; String strPersonToSee; int intVisitorID;</pre>
<p>23. Underneath the declaration add the method mClearVariables.</p>	<pre>private void mClearVariables () { strDate=""; strTimeIn=""; strName = ""; strFromCompany=""; strVehicleRegistrationNumber=""; strTelephoneNumber=""; strReasonForVisit=""; strPersonToSee=""; intVisitorID = 0; }</pre>

<p>24. Underneath the mClearVariables method add the method mGetTheValuesFromGUI</p>	<pre>private void mGetValuesFromGUI () { strDate=txtDate.getText (); strTimeIn=txtTimeIn.getText (); strName = txtName.getText (); strFromCompany = txtFromCompany.getText (); strVehicleRegistrationNumber = txtVehicleRegistrationNumber.getText (); strTelephoneNumber = txtTelephoneNumber.getText (); strReasonForVisit = txtReasonForVisit.getText (); strPersonToSee = txtPersonToSee.getText (); }</pre>
<p>25. Underneath the mGetTheValuesFromGUI add the mSetValuesInGUI</p>	<pre>private void mSetValuesInGUI () { txtDate.setText (strDate); txtTimeIn.setText (strTimeIn); txtName.setText (strName); txtFromCompany.setText (strFromCompany); txtVehicleRegistrationNumber.setText (strVehicleRegistrationNumber); txtTelephoneNumber.setText (strTelephoneNumber); txtReasonForVisit.setText (strReasonForVisit); txtPersonToSee.setText (strPersonToSee); }</pre>
<p>26. Underneath the mSetValesInGUI add the mClearTextFields method.</p>	<pre>private void mClearTextFields () { txtDate.setText (""); txtTimeIn.setText (""); txtName.setText (""); txtFromCompany.setText (""); txtVehicleRegistrationNumber.setText (""); txtTelephoneNumber.setText (""); txtReasonForVisit.setText (""); txtPersonToSee.setText (""); }</pre>
<p>27. Underneath the mClearTextFields add the mCheckIfItemsExistInTable</p>	<pre>private void mCheckIfItemsExistInTable() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection (strDBConnectionString, strDBUser, strDBPassword); Statement myStatement = conMySQLConnectionString.createStatement(); String strQuery ="SELECT*FROM visitors where Date='"+strDate+"' and Time In='"+strTimeIn+"' and Name='"+strName+"' and FromCompany='"+strFromCompany+"'"; myStatement.execute(strQuery); ResultSet rs = myStatement.getResultSet(); boolRecordExists=rs.next(); } catch(Exception e) { JOptionPane.showMessageDialog(null, e); } finally { try { myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e); } } }</pre>

<p>28. Underneath the mCheckIfItemsExistInTable add the mCreateVisitors</p>	<pre>private void mCreateVisitor() { String URL = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String User = "root"; String Password = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(URL, User, Password); try (Statement myStatement = conMySQLConnectionString.createStatement()) { String sqlInsert = "INSERT INTO visitors"+"(Date,Time In,Name,From Company,Vehicle Registration Number,Telephone Number,Re"; myStatement.executeUpdate(sqlInsert); myStatement.close(); } JOptionPane.showMessageDialog(null,"Complete"); } catch(Exception e) { JOptionPane.showMessageDialog(null, e); } }</pre>
<p>29. Underneath the mCreateVisitors add mLoadName</p>	<pre>private void mLoadName() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement = conMySQLConnectionString.createStatement(); String strQuery ="SELECT Name FROM visitors"; myStatement.executeUpdate(strQuery); ResultSet rs = myStatement.getResultSet(); while(rs.next()) { cboName.addItem(rs.getString(1)); } myStatement.close(); } catch(Exception e) { } }</pre>
<p>30. Underneath the mLoadName add the mUpdateVisitors</p>	<pre>private void mUpdateVisitor() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement = conMySQLConnectionString.createStatement(); String strQuery ="UPDATE visitors SET Date='"+strDate+"',TimeIn='"+strTimeIn+"',Name='"+strName+"',FromCompany='"+strFromCompany+"';"; myStatement.executeUpdate(strQuery); myStatement.close(); JOptionPane.showMessageDialog(null, "Updated"); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e); } }</pre>
<p>31. Underneath the mUpdateVisitors add the mClearComboBox</p>	<pre>private void mClearComboBox() { String[] arrArray = new String[0]; javax.swing.DefaultComboBoxModel model = new javax.swing.DefaultComboBoxModel(arrArray); cboName.setModel(model); }</pre>

32. Underneath the mClearComboBox add the mDeleteVisitors

```
private void mDeleteVisitor()
{
    String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol";
    String strDBUser = "root";
    String strDBPassword = "Password";
    try
    {
        java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword);
        String strQuery = "DELETE FROM visitors WHERE Date='"+strDate+"' AND TimeIn='"+strTimeIn+"' AND Name='"+strName+"' AND From Comp";
        Statement myStatement = conMySQLConnectionString.prepareStatement(strQuery);
        myStatement.execute(strQuery);
    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, e);
    }
    finally
    {
        try
        {
            myStatement.close();
            conMySQLConnectionString.close();
        }
        catch(Exception e)
        {
            JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e);
        }
    }
}
```

33. Underneath the mDeleteVisitors add the mReadVisitorDetails

```
private void mReadVisitorDetails ()
{
    String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol";
    String strDBUser = "root";
    String strDBPassword = "Password";
    try
    {
        java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword);
        Statement myStatement=conMySQLConnectionString.createStatement();
        String strQuery = "SELECT Name,FromCompany,VehicleRegistrationNumber,TelephoneNumber,ReasonForVisit,PersonToSee";
        + "WHERE Name ='"+cboName.getSelectedItem().toString()+"'";
        myStatement.execute(strQuery);
        ResultSet rs = myStatement.getResultSet();
        while(rs.next())
        {
            intVisitorID = rs.getInt(1);
            strName=rs.getString(4);
            strFromCompany=rs.getString(5);
            strVehicleRegistrationNumber=rs.getString(6);
            strTelephoneNumber=rs.getString(7);
            strReasonForVisit=rs.getString(8);
            strPersonToSee=rs.getString(9);
        }
    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, e);}
    finally
    {try
    {
        myStatement.close();
    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, e);}
    finally
    {try
    {
        myStatement.close();
    }
    catch(Exception e)
    {
        JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e);
    }
    }
    }
}
```

<p>34. Underneath the mReadVisitorDetails add mLoadGUIControls</p>	<pre>private void mLoadGUIControls() { txtDate.setEnabled(false); txtTimeIn.setEnabled(false); txtName.setEnabled(false); txtFromCompany.setEnabled(false); txtVehicleRegistrationNumber.setEnabled(false); txtTelephoneNumber.setEnabled(false); txtReasonForVisit.setEnabled(false); txtPersonToSee.setEnabled(false); cboName.setEnabled(true); btnCreate.setEnabled(true); btnLoad.setEnabled(true); btnEdit.setEnabled(true); btnSave.setEnabled(false); btnDelete.setEnabled(true); }</pre>
<p>35. Underneath the mLoadGUIControls add the mEditGUIControls</p>	<pre>private void mEditGUIControls() { txtDate.setEnabled(true); txtTimeIn.setEnabled(true); txtName.setEnabled(true); txtFromCompany.setEnabled(true); txtVehicleRegistrationNumber.setEnabled(true); txtTelephoneNumber.setEnabled(true); txtReasonForVisit.setEnabled(true); txtPersonToSee.setEnabled(true); cboName.setEnabled(false); btnCreate.setEnabled(false); btnLoad.setEnabled(false); btnEdit.setEnabled(false); btnSave.setEnabled(true); btnDelete.setEnabled(true); }</pre>

<p>36. Underneath the mEditGUIControls add mSaveGUIControls</p>	<pre>private void mSaveGUIControls() { txtDate.setEnabled(false); txtTimeIn.setEnabled(false); txtName.setEnabled(false); txtFromCompany.setEnabled(false); txtVehicleRegistrationNumber.setEn txtTelephoneNumber.setEnabled(fals txtReasonForVisit.setEnabled(false txtPersonToSee.setEnabled(false); cboName.setEnabled(true); btnCreate.setEnabled(true); btnLoad.setEnabled(true); btnEdit.setEnabled(true); btnSave.setEnabled(false); btnDelete.setEnabled(false); }</pre>
<p>37. Underneath the mSaveGUIControls add mCreateGUIControls</p>	<pre>private void mCreateGUIControls() { txtDate.setEnabled(true); txtTimeIn.setEnabled(true); txtName.setEnabled(true); txtFromCompany.setEnabled(true); txtVehicleRegistrationNumber.setEnabled(true); txtTelephoneNumber.setEnabled(true); txtReasonForVisit.setEnabled(true); txtPersonToSee.setEnabled(true); cboName.setEnabled(false); btnCreate.setEnabled(false); btnLoad.setEnabled(false); btnEdit.setEnabled(false); btnSave.setEnabled(true); btnDelete.setEnabled(true); }</pre>

<p>38. Underneath the mCreateGUIControls add mDeleteGUIControls</p>	<pre>private void mDeleteGUIControls() { txtDate.setEnabled(false); txtTimeIn.setEnabled(false); txtName.setEnabled(false); txtFromCompany.setEnabled(false); txtVehicleRegistrationNumber.setEnabled(false); txtTelephoneNumber.setEnabled(false); txtReasonForVisit.setEnabled(false); txtPersonToSee.setEnabled(false); cboName.setEnabled(true); btnCreate.setEnabled(true); btnLoad.setEnabled(true); btnEdit.setEnabled(true); btnSave.setEnabled(false); btnDelete.setEnabled(false); }</pre>
<p>39. Add the mDefaultDate and mDefaultTime</p>	<pre>private String mDefaultDate() { Date dt = new Date(); SimpleDateFormat sm = new SimpleDateFormat("yyyy-MM-dd"); String strVDate = sm.format(dt); return strDate = strVDate; } private String mDefaultTime() { Date dtTimeIn = new Date(); strTimeIn = DateFormat.getTimeInstance().format(dtTimeIn).replace("PM", ""); return strTimeIn.replace("PM", ""); }</pre>
<p>40. Select the design tab on the frmVisitors and double click on the CREATE button and add the code.</p>	<pre>private void btnCreateActionPerformed(java.awt.event.ActionEvent evt) { mCreateGUIControls(); txtDate.setText(mDefaultDate()); txtTimeIn.setText(mDefaultTime()); txtName.requestFocusInWindow(); btnDelete.setText("Cancel"); boolCreate = true; }</pre>
<p>41. Select the design tab and double click on the LOAD button and add the code</p>	<pre>private void btnLoadActionPerformed(java.awt.event.ActionEvent evt) { mReadVisitorDetails(); mSetValuesInGUI(); mLoadGUIControls(); }</pre>

<p>42. Select the design tab and double click on the EDIT button and add the code.</p>	<pre>private void btnEditActionPerformed(java.awt.event.ActionEvent evt) { mReadVisitorDetails(); mSetValuesInGUI(); mEditGUIControls(); txtName.requestFocusInWindow(); btnDelete.setText("Cancel"); boolEdit = true; }</pre>
<p>43. Select the design tab and double click on the SAVE button and add the code</p>	<pre>private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) { if(boolCreate == true) { if(txtName.getText().equals("")) { JOptionPane.showMessageDialog(null, "The field cannot be left empty"); txtName.requestFocusInWindow(); } else if(txtFromCompany.getText().equals("")) { JOptionPane.showMessageDialog(null, "The field cannot be left empty"); txtFromCompany.requestFocusInWindow(); } if(txtVehicleRegistrationNumber.getText().equals("")) { JOptionPane.showMessageDialog(null, "The field cannot be left empty"); txtVehicleRegistrationNumber.requestFocusInWindow(); } if(txtReasonForVisit.getText().equals("")) { JOptionPane.showMessageDialog(null, "The field cannot be left empty"); txtReasonForVisit.requestFocusInWindow(); } if(txtPersonToSee.getText().equals("")) { JOptionPane.showMessageDialog(null, "The field cannot be left empty"); txtPersonToSee.requestFocusInWindow(); } if(txtTelephoneNumber.getText().equals("")) { JOptionPane.showMessageDialog(null, "The field cannot be left empty"); } } }</pre>

```

else
{
    mGetValuesFromGUI();
    mSetValuesToUpperCase();
    mCheckIfItemsExistInTable();
    if(boolRecordExists == true)
    {
        boolRecordExists = false;
        JOptionPane.showMessageDialog(null, "Visitor al
    }
    else if(boolRecordExists == false)
    {
        boolCreate = false;
        mCreateVisitor();
        mClearTextFields();
        mClearVariables();
        mClearComboBox();
        mLoadName();
        mLoadGUIControls();
    }
}
else if(boolEdit == true)
{
    boolEdit = false;
    mGetValuesFromGUI();
    mSetValuesToUpperCase();
    mUpdateVisitor();
    mClearTextFields();
    mClearVariables();
    mClearComboBox();
    mLoadName();
    mLoadGUIControls();
}
}

```

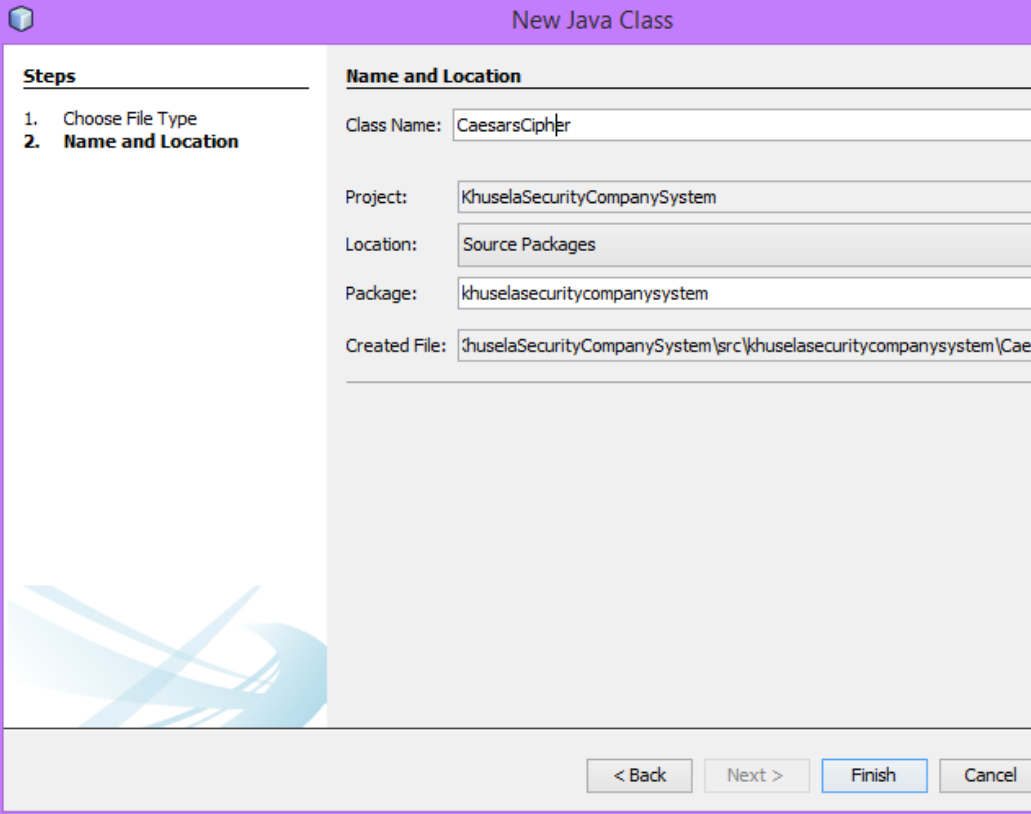
44. Select the design tab and double click on the DELETE button and add the code.

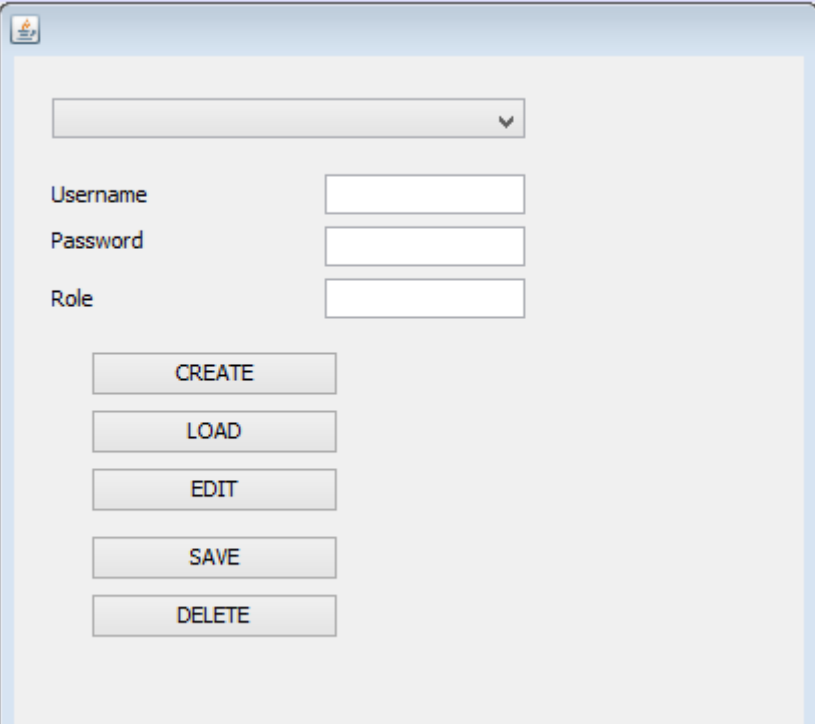
```

private void btnDeleteActionPerformed(java.awt.event.ActionEvent
    if ("Delete".equals(btnDelete.getText()))
    {
        mReadVisitorDetails();
        mDeleteVisitor();
        mClearComboBox();
        mClearVariables();
        mLoadName();
    }
    else if ("Cancel".equals(btnDelete.getText()))
    {
        mClearTextFields();
        mClearVariables();
        mClearComboBox();
        mLoadName();
        mLoadGUIControls();
        btnDelete.setText("Delete");
    }
}

```

45. Select the design tab and double click on the combobox and add the code	<pre>private void cboNameActionPerformed(java.awt.event.ActionEvent e) { mReadVisitorDetails(); mSetValuesInGUI(); }</pre>
46. On the frmVisitors method, add the code	<pre>public frmVisitors() { initComponents(); mLoadName(); mLoadGUIControls(); }</pre>
47. Select the frmViewVisitors tab and insert the code underneath the package name.	<pre>import java.awt.BorderLayout; import java.sql.DriverManager; import java.sql.ResultSet; import java.sql.ResultSetMetaData; import java.sql.Statement; import java.util.Vector; import javax.swing.JOptionPane; import javax.swing.JPanel; import javax.swing.JScrollPane; import javax.swing.JTable;</pre>
48. Underneath the class frmViewVisitors add the code	<pre>private void mLoadViewVisitors() { String URL = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String User = "root"; String Password = "Password"; String strSQLQuery; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(URL, User, Password); Statement stSQLQuery = conMySQLConnectionString.createStatement(); strSQLQuery = "SELECT VisitorID , Date , TimeIn , Name , FromCompany , VehicleRegistrationNumber , TelephoneNumber , P + "PersonToSee FROM Visitors;"; ResultSet rsVisitor = stSQLQuery.executeQuery(strSQLQuery); ResultSetMetaData rsmt = rsVisitor.getMetaData(); int intColumnCount = rsmt.getColumnCount(); Vector vColumn = new Vector(intColumnCount); for(int i = 1; i <= intColumnCount; i++) { vColumn.add(rsmt.getColumnName(i)); } Vector vData = new Vector(); Vector vRow = new Vector(); while(rsVisitor.next()) { vRow = new Vector(intColumnCount); for(int i = 1; i <= intColumnCount; i++) { vRow.add(rsVisitor.getObject(i)); } vData.add(vRow); } } }</pre>

	<pre> JPanel pnlTable = new JPanel(); JTable tblVisitors = new JTable(vData, vColumn); JScrollPane jspUsersPane = new JScrollPane(tblVisitors); tblVisitors.setFillViewportHeight(true); tblVisitors.setVisible(true); tblVisitors.validate(); pnlTable.setLayout(new BorderLayout()); pnlTable.add(jspUsersPane, BorderLayout.CENTER); this.setContentPane(pnlTable); stSQLQuery.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, e); } } </pre>
49. Locate the frmViewVisitors method and add the codes.	<pre>mLoadViewVisitors();</pre>
50. Add a java class name "CaesarsCipher"	
51. Inside the class add the variables code.	<pre> private String strCipherCharacters = "abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ1234567890~`!@#%&*~"; private int intKey = 5; </pre>

<p>52. Underneath the declared variables add the mCleanString code.</p>	<pre>private String mCleanString(String strText) { strText = strText.replace("\n", ""); for(int x = 0; x < strText.length(); x++) { int intPosition = strCipherCharacters.indexOf(strText.charAt(x)); if(intPosition == -1) { strText = strText.replace(strText.charAt(x), ' '); } } return strText; }</pre>
<p>53. Underneath the mCleanString code add the mEncrypt code.</p>	<pre>public String mEncrypt(String strText, int intKey) { String strCleanedText = mCleanString(strText); String strEncrypted = ""; for(int i = 0; i < strCleanedText.length(); i++) { int intCharacterPosition = strCipherCharacters.indexOf(strCleanedText.charAt(i)); if((intCharacterPosition + intKey) < strCipherCharacters.length()) { strEncrypted += strCipherCharacters.charAt(intCharacterPosition + intKey); } else { strEncrypted += strCipherCharacters.charAt((intCharacterPosition + intKey) - strCipherCharacters.length()); } } return strEncrypted; }</pre>
<p>54. Underneath the mEncrypt code add the mDecrypt code.</p>	<pre>public String mDecrypt(String strText, int intKey) { String strCleanedText = mCleanString(strText); String strDecrypted = ""; for(int i = 0; i < strCleanedText.length(); i++) { int intCharacterPosition = strCipherCharacters.indexOf(strCleanedText.charAt(i)); if((intCharacterPosition - intKey) < 0) { strDecrypted += strCipherCharacters.charAt((intCharacterPosition - intKey) + strCipherCharacters.length()); } else { strDecrypted += strCipherCharacters.charAt(intCharacterPosition - intKey); } } return strDecrypted; }</pre>
<p>55. Select the frmUsers tab and click on the design tab and add the following controls:</p> <ul style="list-style-type: none"> 3 Labels <ol style="list-style-type: none"> 1. Username (lblUsername) 2. Password (lblPassword) 3. Role (lblRole) 3 textfields <ol style="list-style-type: none"> 1. txtUsername 2. txtPassword 3. txtRole 5 buttons <ol style="list-style-type: none"> 1. CREATE (btnCreate) 2. LOAD (btnLoad) 3. EDIT (btnEdit) 4. SAVE (btnSave) 5. DELETE (btnDelete) 	

56. Select the source tab and underneath the package name add the code	<pre>import java.sql.DriverManager; import java.sql.ResultSet; import java.sql.Statement; import javax.swing.JOptionPane;</pre>
57. Locate the method "frmUsers" and declare the variables	<pre>CaesarCipher clsCC = new CaesarCipher(); Boolean boolRecordExists = false; Boolean boolEdit = false; Boolean boolCreate = false; String strUserName; String strPassword; String strRole; int intUserID;</pre>
58. Underneath the declared variables add the mClearVariables code	<pre>private void mClearVariables () { strUserName = ""; strPassword = ""; strRole = ""; intUserID = 0; }</pre>
59. Underneath the mClearVariables add the mgetValuesFromGUI	<pre>private void mGetValuesFromGUI () { strUserName = txtUserName.getText (); strPassword = txtPassword.getText (); strRole = txtRole.getText (); }</pre>
60. Underneath the mGetValuesFromGUI add the mSetValuesToUpperCase	<pre>private void mSetValuesToUpperCase () { strUserName = strUserName.toUpperCase (); strPassword = strPassword.toUpperCase (); strRole = strRole.toUpperCase (); }</pre>
61. Underneath the mSetValuesToUpperCase add the mClearTextFields	<pre>private void mClearTextFields () { txtUserName.setText (""); txtPassword.setText (""); txtRole.setText (""); }</pre>
62. Add the mSetValuesInGUI	<pre>private void mSetValuesInGUI () { txtUserName.setText (strUserName); txtPassword.setText (strPassword); txtRole.setText (strRole); }</pre>

<p>63. Underneath the mClearTextFields add the mCheckIfItemsExistInTable</p>	<pre>private void mCheckIfItemsExistInTable() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement; myStatement = conMySQLConnectionString.createStatement(); String strQuery = "SELECT * FROM users WHERE UserName='" + strUsername + "' AND Password='" + strPassword + "' AND Role='" + strRole + "'"; myStatement.execute(strQuery); ResultSet rs = myStatement.getResultSet(); boolean boolRecordExists = rs.next(); myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection String not closed"+e); } }</pre>
<p>64. Underneath the mCheckIfItemsExistInTable add the mCreateUser</p>	<pre>private void mCreateUser() { String URL = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String User = "root"; String Password = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(URL, User, Password); try (Statement myStatement = conMySQLConnectionString.createStatement()) { String sqlInsert = "INSERT INTO users (" + (UserName,Password,Role) + "VALUES ('" + strUsername + "','" + strPassword + "','" + strRole + "')"; myStatement.executeUpdate(sqlInsert); myStatement.close(); } JOptionPane.showMessageDialog(null, "Complete"); } catch(Exception e) { JOptionPane.showMessageDialog(null, ""); } }</pre>
<p>65. Underneath the mCreateUser add the mLoadUsername</p>	<pre>private void mLoadUsername() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement; myStatement = conMySQLConnectionString.createStatement(); String strQuery = "SELECT Username FROM users"; myStatement.execute(strQuery); ResultSet rs = myStatement.getResultSet(); while(rs.next()) { cboUsername.addItem(rs.getString(1)); } myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection String not closed"+e); } }</pre>

<p>66. Underneath the mLoadUser add the mUpdateUser</p>	<pre>private void mUpdateUser() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement; myStatement = conMySQLConnectionString.createStatement(); String strQuery = "UPDATE users SET UserName='"+strUsername+"' Password = '"+ strPassword+"' Role ='"+ strRole +"' WHERE U"; myStatement.executeUpdate(strQuery); myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e); } }</pre>
<p>67. Underneath the mUpdateUser add the mClearComboBox</p>	<pre>private void mClearComboBox() { String[] arrArray = new String[0]; javax.swing.DefaultComboBoxModel model = new javax.swing.DefaultComboBoxModel(arrArray); cboUsername.setModel(model); }</pre>
<p>68. Underneath the mClearComboBox add the mDeleteUser</p>	<pre>private void mDeleteUser() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); String strQuery = "DELETE FROM users WHERE UserName='"+strUsername+"' AND Password = '"+ strPassword+"' AND Role ='"+ strRole +"'"; Statement myStatement; myStatement = conMySQLConnectionString.prepareStatement(strQuery); myStatement.executeUpdate(strQuery); myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e); } }</pre>
<p>69. Underneath the mDeleteUser add the mReadUserDetails</p>	<pre>private void mReadUserDetails() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement; myStatement = conMySQLConnectionString.createStatement(); String strQuery = "SELECT UserID,UserName, Password, Role FROM users WHERE Username='"+cboUsername.getSelectedItem()+"'"; myStatement.executeUpdate(strQuery); ResultSet rs = myStatement.getResultSet(); while(rs.next()) { intUserID = rs.getInt(1); strUsername = rs.getString(2); strPassword = rs.getString(3); strRole = rs.getString(4); } myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection string not closed"+" "+e); } }</pre>

<p>70. Underneath the mReadUserDetails add the mLoadGUIControls</p>	<pre>private void mLoadGUIControls() { txtUsername.setEnabled(false); txtPassword.setEnabled(false); txtRole.setEnabled(false); cboUsername.setEnabled(true); btnCreate.setEnabled(true); btnLoad.setEnabled(true); btnEdit.setEnabled(true); btnSave.setEnabled(false); btnDelete.setEnabled(true); }</pre>
<p>71. Underneath the mLoadGUIControls add the mEditGUIControls</p>	<pre>private void mEditGUIControls() { txtUsername.setEnabled(true); txtPassword.setEnabled(true); txtRole.setEnabled(true); cboUsername.setEnabled(false); btnCreate.setEnabled(false); btnLoad.setEnabled(false); btnEdit.setEnabled(false); btnSave.setEnabled(true); btnDelete.setEnabled(true); }</pre>
<p>72. Underneath the mLoadGUIControls add mSaveGUIControls.</p>	<pre>private void mSaveGUIControls() { txtUsername.setEnabled(false); txtPassword.setEnabled(false); txtRole.setEnabled(false); cboUsername.setEnabled(true); btnCreate.setEnabled(true); btnLoad.setEnabled(true); btnEdit.setEnabled(true); btnSave.setEnabled(false); btnDelete.setEnabled(false); }</pre>
<p>73. Underneath the mSaveGUIControls add mDeleteGUIControls.</p>	<pre>private void mCreateGUIControls() { txtUsername.setEnabled(true); txtPassword.setEnabled(true); txtRole.setEnabled(true); cboUsername.setEnabled(false); btnCreate.setEnabled(false); btnLoad.setEnabled(false); btnEdit.setEnabled(false); btnSave.setEnabled(true); btnDelete.setEnabled(true); }</pre>

<p>74. Underneath the mDeleteControls add the mCreateGUIControls</p>	<pre>private void mCreateGUIControls() { txtUsername.setEnabled(true); txtPassword.setEnabled(true); txtRole.setEnabled(true); cboUsername.setEnabled(false); btnCreate.setEnabled(false); btnLoad.setEnabled(false); btnEdit.setEnabled(false); btnSave.setEnabled(true); btnDelete.setEnabled(true); }</pre>
<p>75. Underneath the mCreateGUIControls add the mEncryptPassword and the mDecryptPassword</p>	<pre>private void mEncryptPassword() { strPassword = clsCC.mEncrypt(strPassword, 5); } private void mDecryptPassword() { strPassword = clsCC.mDecrypt(strPassword, 5); }</pre>
<p>76. Select the design tab and double click on the CREATE button and add the code</p>	<pre>private void btnCreateActionPerformed(java.awt.event.ActionEvent e) { mCreateGUIControls(); txtUsername.requestFocusInWindow(); btnDelete.setText("Cancel"); boolCreate = true; }</pre>
<p>77. Select the design tab and double click on the LOAD button and enter the code</p>	<pre>private void btnLoadActionPerformed(java.awt.event.ActionEvent e) { mReadUserDetails(); mEncryptPassword(); mSetValuesInGUI(); mLoadGUIControls(); }</pre>
<p>78. Select the design tab and double click on the EDIT button and enter the code</p>	<pre>private void btnEditActionPerformed(java.awt.event.ActionEvent e) { mReadUserDetails(); mDecryptPassword(); mSetValuesInGUI(); mEditGUIControls(); txtUsername.requestFocusInWindow(); btnDelete.setText("Cancel"); boolEdit = true; }</pre>

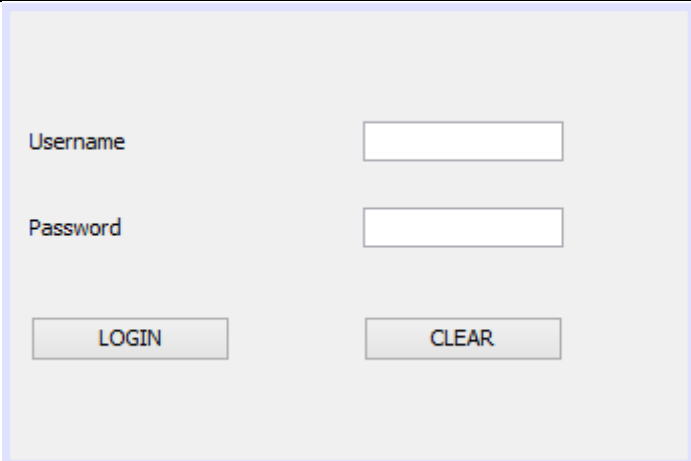
79. Select the design tab and double click on the SAVE button and enter the code

```
private void btnSaveActionPerformed(java.awt.event.ActionEvent evt) {  
    if(boolCreate == true)  
    {  
        if(txtUsername.getText().equals(""))  
        {  
            JOptionPane.showMessageDialog(null, "The field cannot be left empty");  
            txtUsername.requestFocusInWindow();  
        }  
        else if(txtPassword.getText().equals(""))  
        {  
            JOptionPane.showMessageDialog(null, "The field cannot be left empty");  
            txtPassword.requestFocusInWindow();  
        }  
        else if(txtRole.getText().equals(""))  
        {  
            JOptionPane.showMessageDialog(null, "The field cannot be left empty");  
            txtRole.requestFocusInWindow();  
        }  
        else  
        {  
            mGetValuesFromGUI();  
            mSetValuesToUpperCase();  
            mEncryptPassword();  
            mCheckIfItemsExistInTable();  
            if(boolRecordExists == true)  
            {  
                boolRecordExists = false;  
                JOptionPane.showMessageDialog(null, "User already Exists");  
            }  
            else if(boolRecordExists == false)
```

	<pre> else if (boolRecordExists == false) { boolCreate = false; mCreateUser(); mClearTextFields(); mClearVariables(); mClearComboBox(); mLoadUsername(); mLoadGUIControls(); } } else if (boolEdit == true) { boolEdit = false; mGetValuesFromGUI(); mSetValuesToUpperCase(); mEncryptPassword(); mUpdateUser(); mClearTextFields(); mClearVariables(); mClearComboBox(); mLoadUsername(); mLoadGUIControls(); } btnDelete.setText("Delete"); </pre>
<p>80. Select the design tab and double click on the DELETE button and enter the code</p>	<pre> private void btnDeleteActionPerformed(java.awt.event.ActionEvent if ("Delete".equals(btnDelete.getText())) { mReadUserDetails(); mDeleteUser(); mClearComboBox(); mClearVariables(); mLoadUsername(); } else if ("Cancel".equals(btnDelete.getText())) { mClearTextFields(); mClearVariables(); mClearComboBox(); mLoadUsername(); mLoadGUIControls(); btnDelete.setText("Delete"); } } </pre>

81. Locate the frmUsers and add the code	<pre>mLoadUsername(); mLoadGUIControls();</pre>
82. Select the frmViewUsers tab and click on the source tab and add the code under the package name.	<pre>import java.awt.BorderLayout; import java.sql.Connection; import java.sql.DriverManager; import java.sql.ResultSet; import java.sql.ResultSetMetaData; import java.sql.Statement; import java.util.Vector; import javax.swing.JFrame; import javax.swing.JOptionPane; import javax.swing.JPanel; import javax.swing.JScrollPane; import javax.swing.JTable;</pre>
83. Locate the frmViewUser method and add the code	<pre>mLoadUsername(); mLoadGUIControls();</pre>
84. Underneath the frmViewUsers method add the mLoadViewUsers code.	<pre>private void mLoadUsername() { String strDBConnectionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; try { java.sql.Connection conMySQLConnectionString = DriverManager.getConnection(strDBConnectionString, strDBUser, strDBPassword); Statement myStatement; myStatement = conMySQLConnectionString.createStatement(); String strQuery = "SELECT Username FROM users"; myStatement.execute(strQuery); ResultSet rs = myStatement.getResultSet(); while(rs.next()) { cboUsername.addItem(rs.getString(1)); } myStatement.close(); } catch(Exception e) { JOptionPane.showMessageDialog(null, "Connection String not closed"+e); } }</pre>

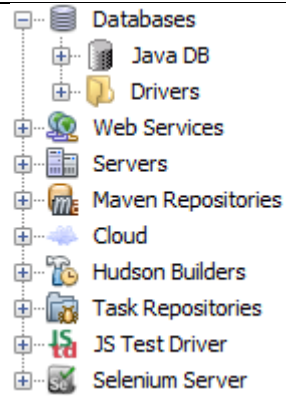
	<pre> vData.add(vRow); } JPanel pnlTable = new JPanel(); JTable tblActor = new JTable(vData, vColumn); JScrollPane jspUsersPane = new JScrollPane(tblActor); tblActor.setFillViewportHeight(true); tblActor.setVisible(true); tblActor.validate(); pnlTable.setLayout(new BorderLayout()); pnlTable.add(jspUsersPane, BorderLayout.CENTER); this.setContentPane(pnlTable); } catch(Exception eX) { JOptionPane.showMessageDialog(null, eX); } finally { try { stSQLQuery.close(); } catch(Exception eX) { JOptionPane.showMessageDialog(null, "Connection string not } } </pre>
85. Scroll up and locate the frmViewUsers method and add the code	<pre> mLoadViewUsers(); </pre>
86. Select the java application called "KhuselaSecurityCompanySystem"	
87. Add the code in the main method.	<pre> public static void main(String[] args) { frmLogin frmL = new frmLogin(); frmL.setLocationRelativeTo(null); frmL.setVisible(true); } } </pre>

<p>88. Select the frmLogin tab and insert the following and change the variable names:</p> <ul style="list-style-type: none"> 2 labels <ol style="list-style-type: none"> Username (lblUsername) Password (lblPassword) 2 text fields <ol style="list-style-type: none"> txtUsername txt Password 2 buttons <ol style="list-style-type: none"> LOGIN (btnLogin) CANCEL (btnCancel) 		
<p>89. Select the source tab and under the package name insert the code.</p>	<pre>import static java.awt.Frame.MAXIMIZED_BOTH; import java.sql.DriverManager; import java.sql.ResultSet; import java.sql.Statement; import javax.swing.JOptionPane;</pre>	
<p>90. Underneath the frmLogin method declare the variables as shown</p>	<pre>String strUsername; String strPassword; String strRole; CaesarsChiper clsCC = new CaesarsChiper();</pre>	
<p>91. Underneath the class instance, add the mFetchUserDetails code</p>	<pre>private void mFetchUserDetails() { String strDBConnctionString = "jdbc:mysql://localhost:3306/gateaccesscontrol"; String strDBUser = "root"; String strDBPassword = "Password"; java.sql.Connection conMySQLConnectionString; Statement stStatement; ResultSet rs; try { conMySQLConnectionString = DriverManager.getConnection(strDBConnctionString, strDBUser, strDBPassword); stStatement = conMySQLConnectionString.createStatement(); String strQuery = "Select ID, Username, Password, Role from Users where Username='" + txtUsername.getText().to stStatement.execute(strQuery); rs = stStatement.getResultSet(); while(rs.next()) { strUsername = rs.getString(2); strPassword = rs.getString(3); strRole = rs.getString(4); stStatement.close(); } } catch (Exception e) { JOptionPane.showMessageDialog(null, e); } }</pre>	

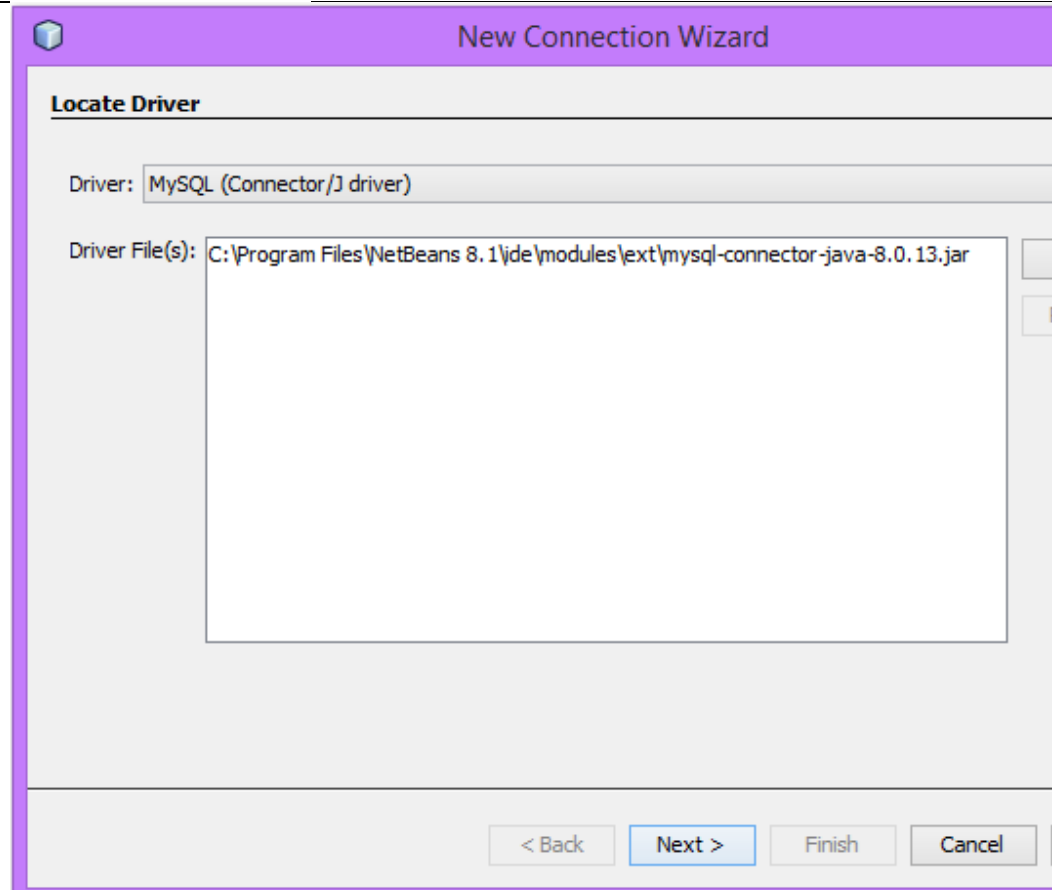
<p>92. Underneath the mFetchUserDetails add the mEncryptPassword code and mDecryptPassword code</p>	<pre>private void mEncryptPassword() { strPassword=clsCC.mEncrypt(txtPassword.getText().toUpperCase(), } private void mDecryptPassword() { strPassword=clsCC.mDecrypt(strPassword.toLowerCase(),5); }</pre>
<p>93. Select the design tab and double click on the CLEAR button and add the code.</p>	<pre>private void btnClearActionPerformed(java.awt.event.ActionEvent System.exit(0); }</pre>
<p>94. Select the design tab and double click on login button and add the code</p>	<pre>private void btnLoginActionPerformed(java.awt.event.ActionEvent evt) { frmMain frmM= new frmMain(); if(txtUsername.getText().equals("Sandra")&&txtPassword.getText().equals("18003804")) { frmM.strRole="MANAGER"; frmM.mSetRoleRights(); frmM.setVisible(true); this.setVisible(false); frmM.setExtendedState(MAXIMIZED_BOTH); } if(txtUsername.getText().equals("SANDRA")&&txtPassword.getText().equals("18003804")) { frmM.strRole="SECURITY"; frmM.mSetRoleRights(); frmM.setVisible(true); this.setVisible(false); frmM.setExtendedState(MAXIMIZED_BOTH); } else { mEncryptPassword(); mFetchUserDetails(); mDecryptPassword(); if(txtUsername.getText().toUpperCase().equals(strUsername)&& txtPassword.getText().toUpperCase().equals(strPassword) { frmM.strRole=strRole; frmM.mSetRoleRights(); frmM.setVisible(true); this.setVisible(false); frmM.setExtendedState(MAXIMIZED_BOTH); } else { mEncryptPassword(); mFetchUserDetails(); mDecryptPassword(); if(txtUsername.getText().toUpperCase().equals(strUsername)&& txtPassword.getText().toUpperCase().equals(strPassword) { frmM.strRole=strRole; frmM.mSetRoleRights(); frmM.setVisible(true); this.setVisible(false); frmM.setExtendedState(MAXIMIZED_BOTH); } else { JOptionPane.showMessageDialog(null,"Check UserName and Password"); } } } }</pre>

<p>95. Select the frmMain tab and select the source tab. Underneath the declared variables (strRole), add the mSetRoleRights code.</p>	<pre> public String strRole; public void mSetRoleRights () { if (null!=strRole) switch (strRole) { case "MANAGER": mnuLogOff.setEnabled(true); mnuExit.setEnabled(true); mnuUsers.setEnabled(true); mnuViewUsers.setEnabled(true); mnuVisitors.setEnabled(true); mnuViewVisitors.setEnabled(true); break; case "SECURITY": mnuLogOff.setEnabled(true); mnuExit.setEnabled(true); mnuUsers.setEnabled(true); mnuViewUsers.setEnabled(true); mnuVisitors.setEnabled(true); mnuViewVisitors.setEnabled(true); break; case "STANDARD": mnuLogOff.setEnabled(true); mnuExit.setEnabled(true); mnuUsers.setEnabled(false); mnuViewUsers.setEnabled(true); mnuVisitors.setEnabled(false); mnuViewVisitors.setEnabled(true); break; default: break; } } </pre>
<p>We are done with the coding for this project. The next step is to create a database connection on the application.</p>	
<p>96. With Netbeans IDE still open, locate the Services tab</p>	

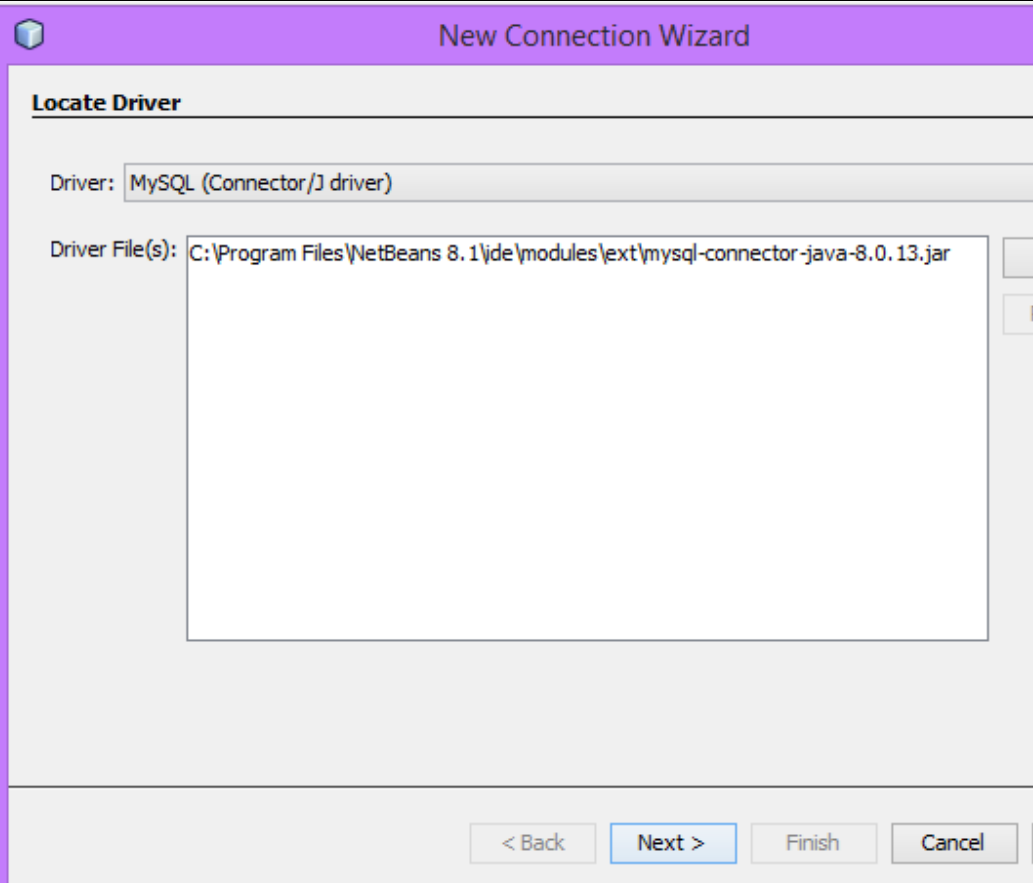
97. Locate the Databases on the services tab and expand the Databases tab.



98. On the Databases tab, right click and click on New Connection

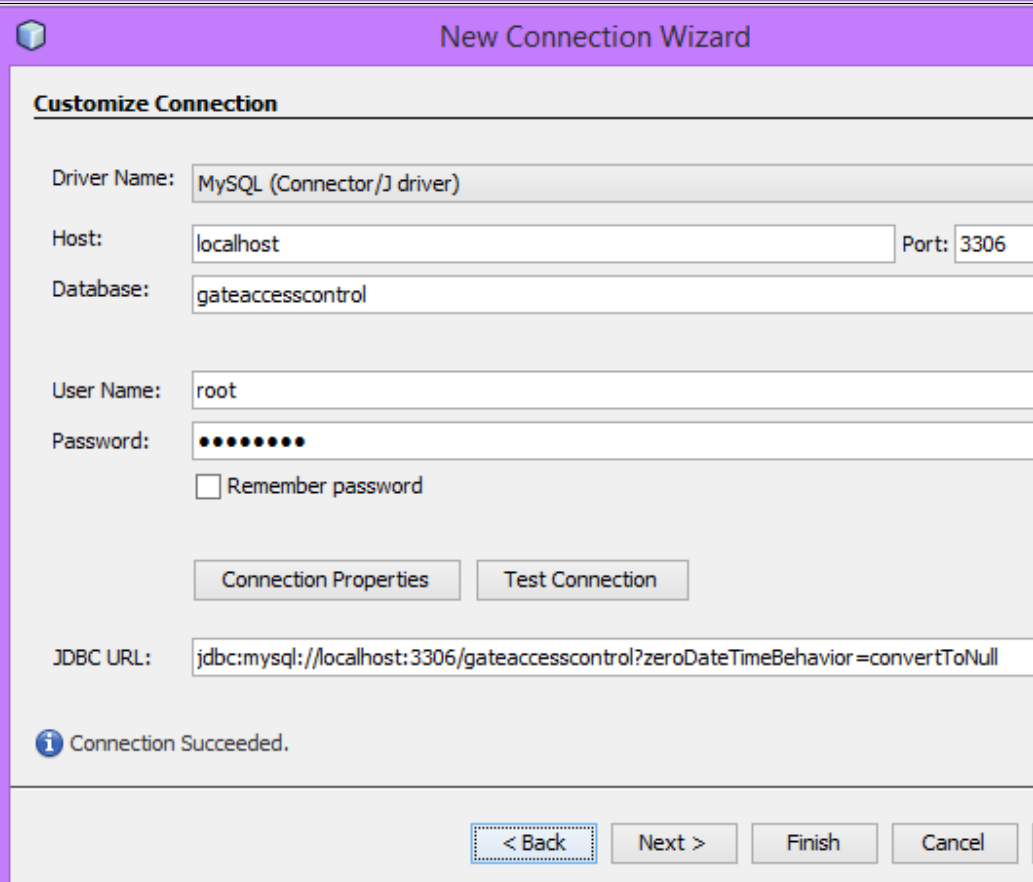


99. The new connection wizard will display. Click on the Driver drop down and select MySQL (Connector/J driver).





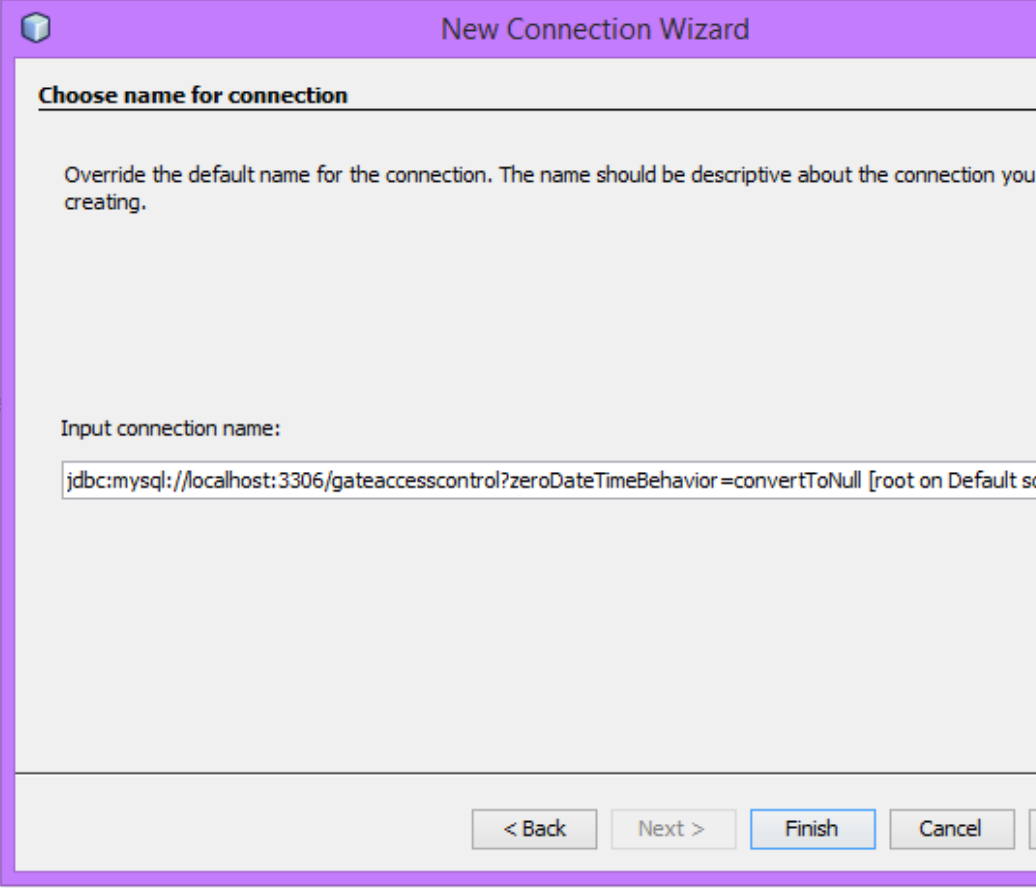
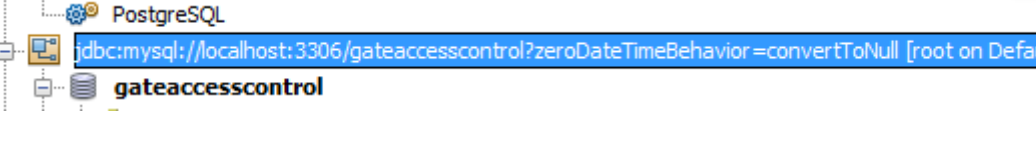
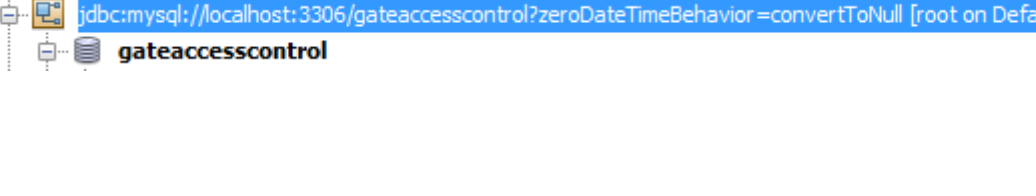
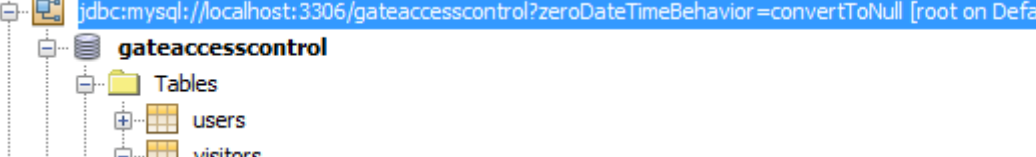
The screenshot shows the 'New Connection Wizard' dialog box, specifically the 'Locate Driver' step. The title bar is purple with a white cube icon and the text 'New Connection Wizard'. The main area has a light gray background. At the top, there's a section header 'Locate Driver' in bold. Below it, there are two text fields: 'Driver:' with the value 'MySQL (Connector/J driver)' and 'Driver File(s):' with the value 'C:\Program Files\NetBeans 8.1\ide\modules\ext\mysql-connector-java-8.0.13.jar'. At the bottom right, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

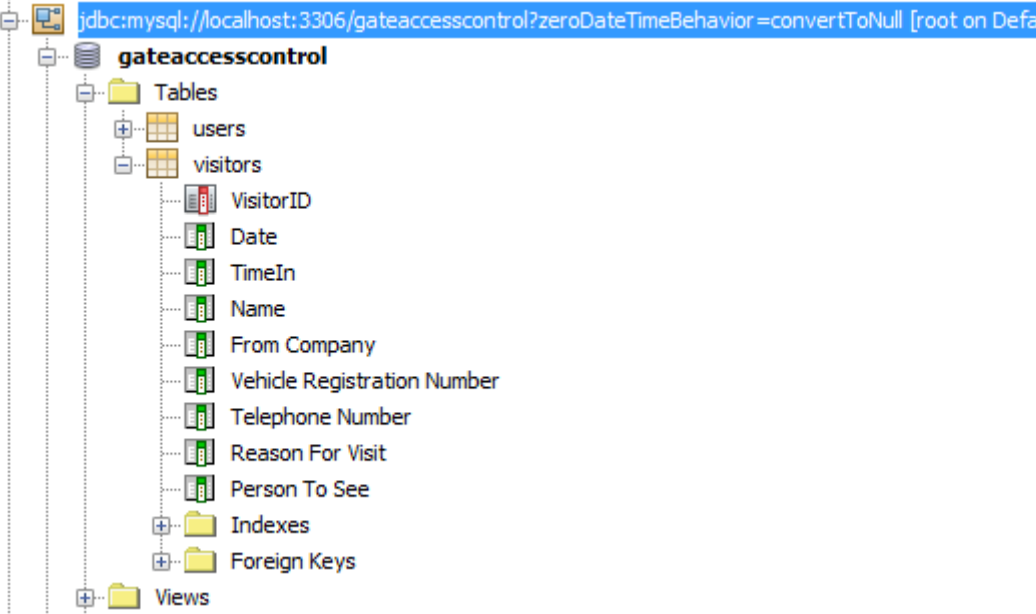
100. Enter gateaccesscontrol as the Database then enter "root" as the user name and "Password" as the password.



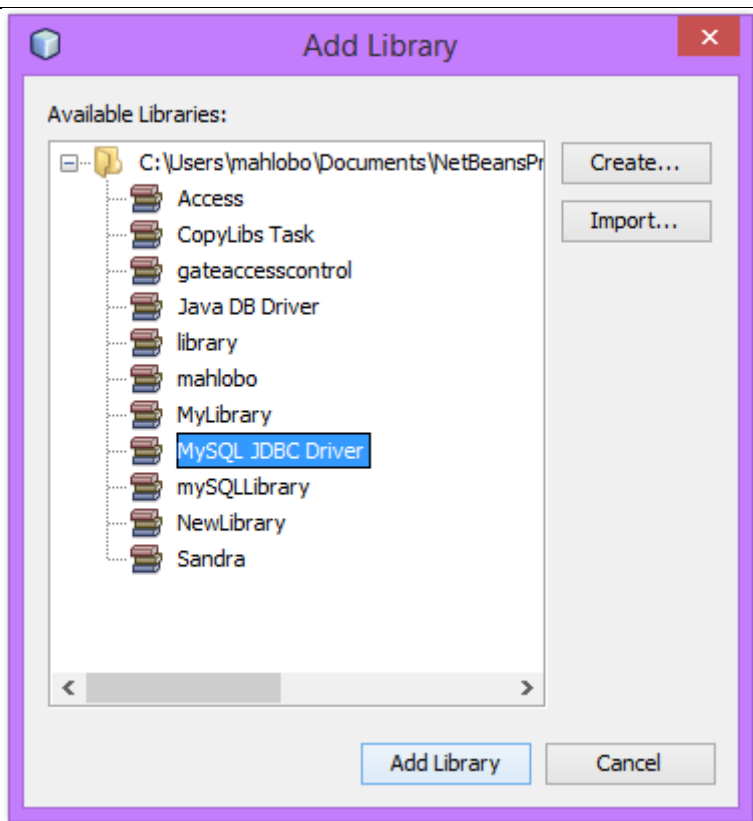
The screenshot shows the 'New Connection Wizard' dialog box, specifically the 'Customize Connection' step. The title bar is purple with a white cube icon and the text 'New Connection Wizard'. The main area has a light gray background. At the top, there's a section header 'Customize Connection' in bold. Below it, there are several text fields: 'Driver Name:' with the value 'MySQL (Connector/J driver)', 'Host:' with the value 'localhost', 'Port:' with the value '3306', 'Database:' with the value 'gateaccesscontrol', 'User Name:' with the value 'root', and 'Password:' with a masked password '••••••••'. There is also a checkbox labeled 'Remember password' which is unchecked. Below these fields are two buttons: 'Connection Properties' and 'Test Connection'. At the bottom, there is a text field for 'JDBC URL:' with the value 'jdbc:mysql://localhost:3306/gateaccesscontrol?zeroDateTimeBehavior=convertToNull'. At the bottom left, there is an information icon and the text 'Connection Succeeded.'. At the bottom right, there are four buttons: '< Back', 'Next >', 'Finish', and 'Cancel'.

101. Click on test the connection. The connection is shown to be successful	 Connection Succeeded.
102. Click on Next and the choose database schema menu will show. Click Next	<div><div></div><div>New Connection Wizard</div></div> <div><div>Choose Database Schema</div><div>For each database connection, the Services window only displays objects from one database schema. Select the schema of the tables to be displayed.</div><div>Select schema: <input type="text" value="<No schema>"/></div><div><div>< Back</div><div>Next ></div><div>Finish</div><div>Cancel</div></div></div>

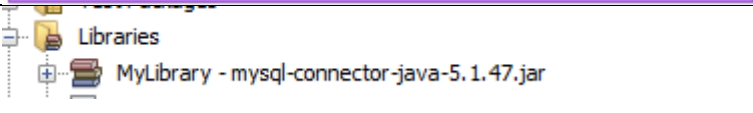
<p>103. The choose name for connection window will show, Click on Finish.</p>	
<p>104. Notice on the left handside on the Services,Databases tab the connection to MySQL is established.</p>	
<p>105. Expand the connected schema that has been connected and "gateaccesscontrol" will be shown as the database schema.</p>	
<p>106. Expand the gateaccesscontrol database and you will notice 2 tables "Visitors" and "Users".</p>	

<p>107. Expand the Visitors table and you will notice the columns inside the table.</p>	
<p>108. The SQL gateaccesscontrol schema is connected to the netbeans .</p>	
<p>We will now add a JDBC driver to the project.</p>	
<p>109. In the Project window, select Libraries menu.</p>	
<p>110. Right click on the Libraries menu and select Add Library.</p>	

111. The Add Library window will display. Locate the MySQL JDBC Driver and click on it then click Add Library



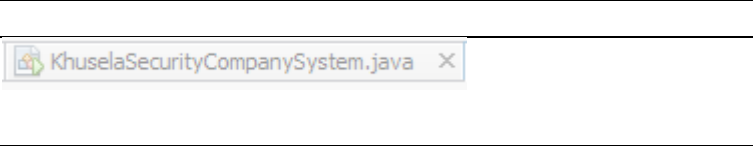
112. Notice on the Libraries the MySQL JDBC Driver is added.



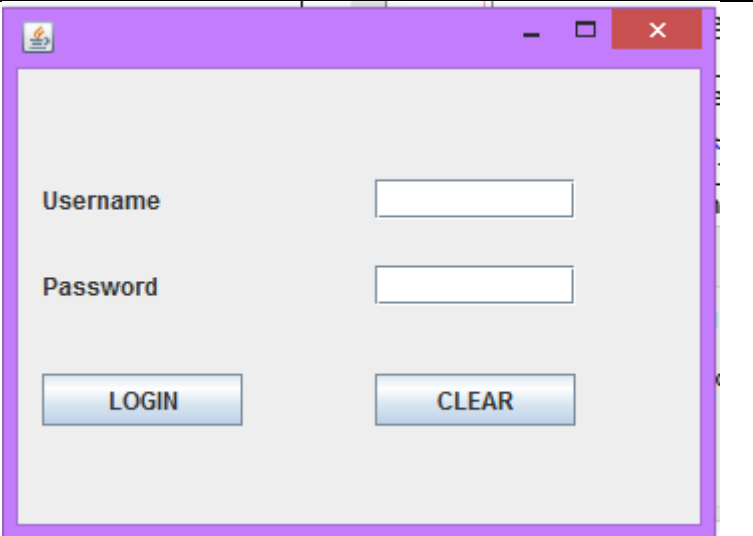
We are then done adding a JDBC Driver into our project.

We will now test the project if it is running well. We will first test the Login screen.

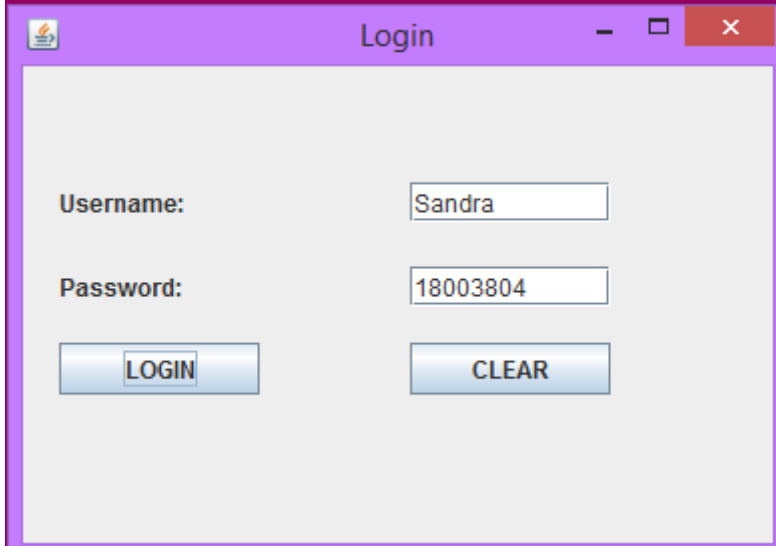
113. Select the "KhuselaSecurityCompanySystem.java" java application.



114. Run the project and the LOGIN SCREEN will display.



115. Enter the following credentials to authenticate.
Username: Sandra
Password: 18003804 then click the LOGIN button.



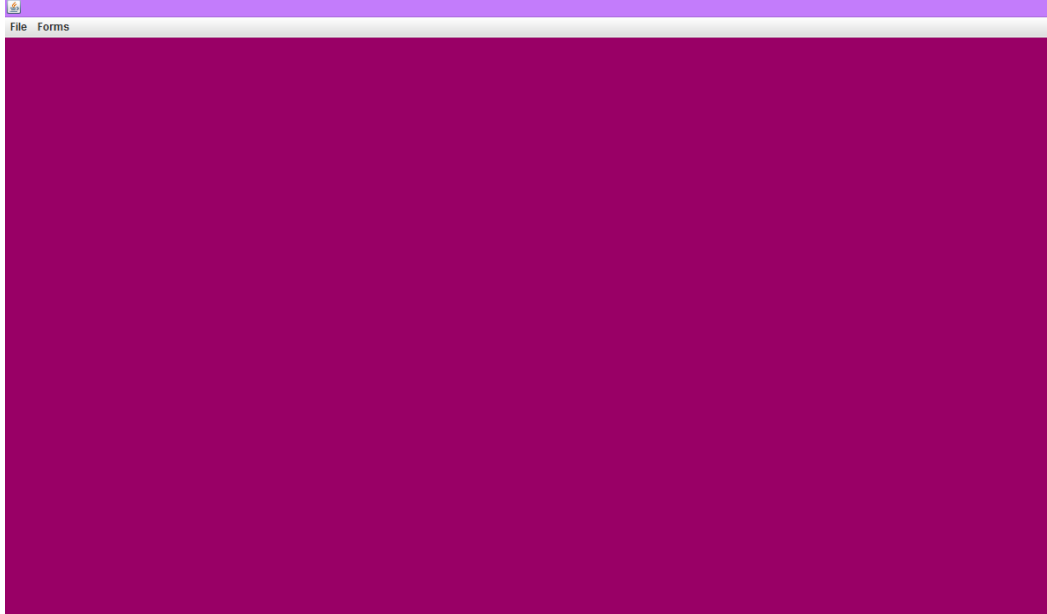
Login

Username: Sandra

Password: 18003804

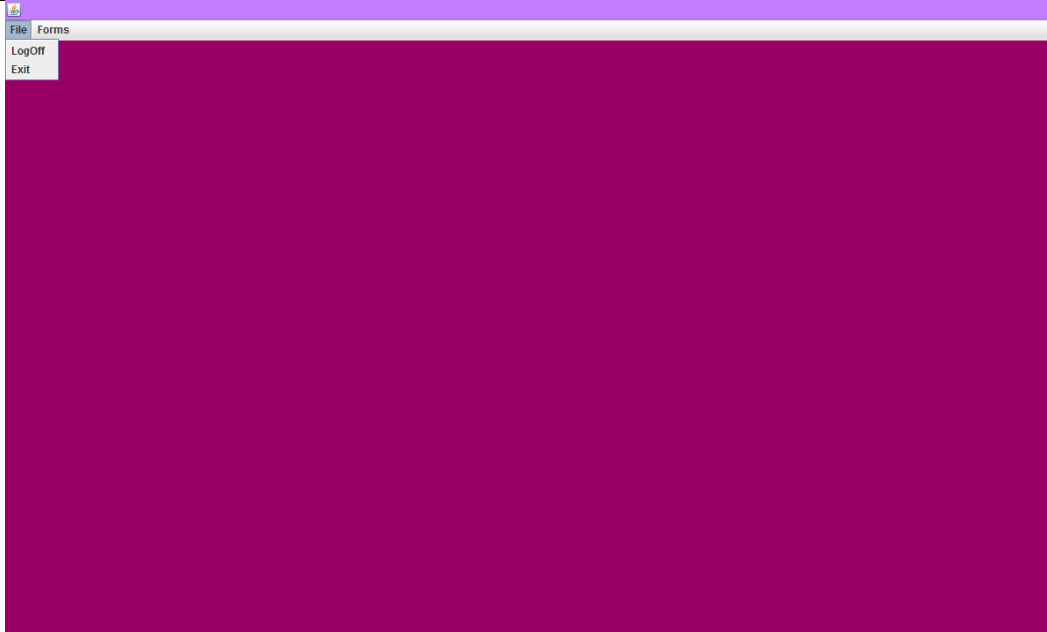
LOGIN CLEAR

116. The main form will display



File Forms

117. Click on the File menu and notice the LogOff and the Exit menu items.



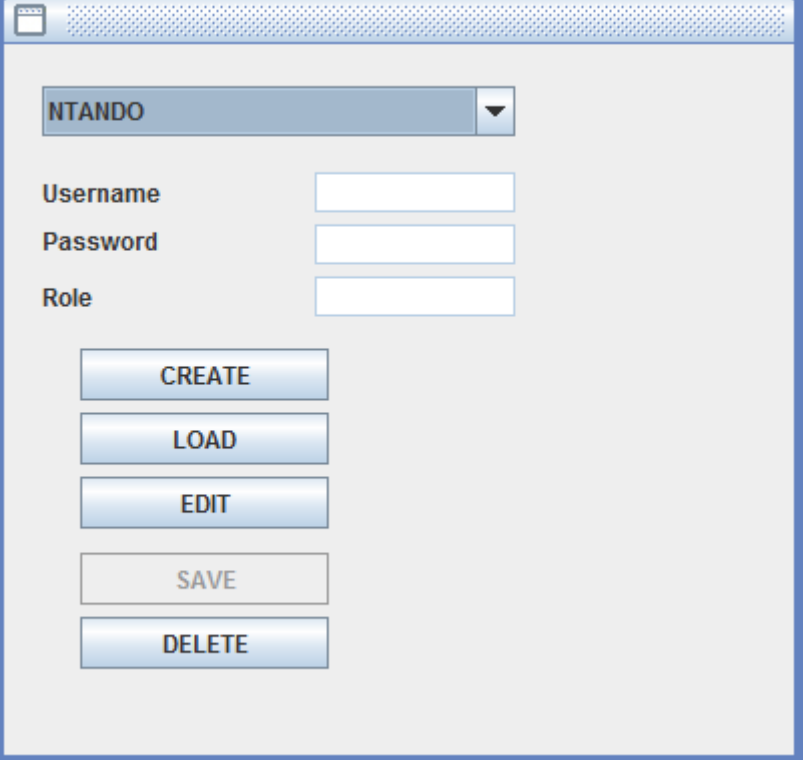
File Forms

LogOff

Exit

118. Click on the Forms menu and notice the:
Users
View Users
Visitors
View Visitors

119. Click on the Users menu and the Users form will be displayed



NTANDO

Username

Password

Role

CREATE

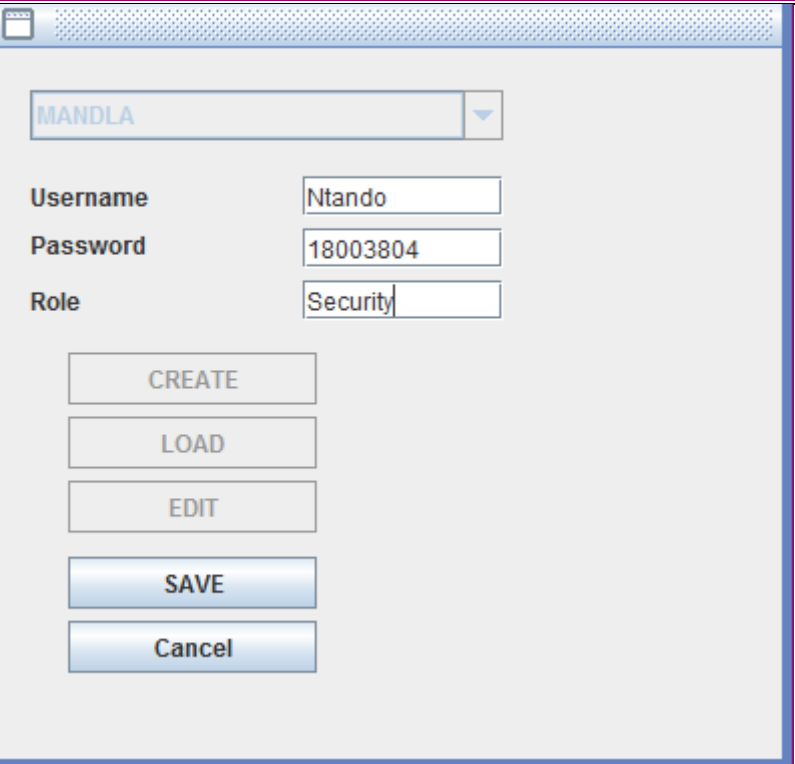
LOAD

EDIT

SAVE

DELETE

120. Click on the Create button to add the following user details.
Username: Ntando
Password: 18003804
Role: Security



MANDLA

Username Ntando

Password 18003804

Role Security

CREATE

LOAD

EDIT

SAVE

Cancel

121. Click on the save button and notice a popup message indicating that the user has been added.the click on ok. (More users are added and to be shown on the view users form)

The screenshot shows a web application window titled "File Forms". Inside, there is a form for adding a new user. At the top, there is a dropdown menu currently showing "MANDLA". Below this are four input fields: "Username" with the value "Ntando", "Password" with the value "18003804", and "Role" with the value "Security". Below the input fields are five buttons: "CREATE", "LOAD", "EDIT", "SAVE", and "Cancel". The "SAVE" button is highlighted in blue. To the right of the main form, a purple "Message" popup is visible, containing an information icon, the text "Complete", and an "OK" button.

122. Click on the Visitors menu and the Visitors form will be displayed.

The screenshot shows the same "File Forms" application window, but now displaying the "Visitors" form. The form has a dropdown menu at the top. Below it are seven input fields, each with a label to its left: "Date", "Time In", "Name", "From Company", "Vehicle Registration Number", "Telephone Number", and "Reason For Visit". The last field, "Person To See", is located below the "Reason For Visit" field. At the bottom of the form are five buttons: "CREATE", "LOAD", "EDIT", "SAVE", and "DELETE". The "CREATE" button is highlighted in blue.

123. Click on the create button to add visitors' details. Notice the date and time are added to the form automatically as this was set on the code. Add the following details:
Name: Mahlobo
From Company: Multi Choice
Vehicle Registration Number: FM86XH GP
Telephone Number: 0732015302
Reason For Visit: Interview
Person To See: Reshoketshwe Masia
- Then click on the Save button to save details.

File Forms

ADD

Date 2019-01-09

Time In 4:03:07

Name Mahlobo

From Company MultiChoice

Vehicle Registration Number FM86XH GP

Telephone Number 073 2105302

Reason For Visit Interview

Person To See Reshoketswe Masia

CREATE

LOAD

EDIT

SAVE

Cancel

124. The message indicating that the Visitor has been added successfully will be shown. (more Visitors are to be added and will be shown on the View Visitors form).

Message

Complete

OK

125. Click on the View Users form to displayed all the added users.

File Forms

UserID	UserName	Password	Role
4	MANDLA	67PMO	SECURITY
5	MASIA	RRFYMFUJQT	MANAGER
6	MONWABISI	6##7RTS2FN	SECURITY
9	NTANDO	6@\$8@\$9	SECURITY

- Click on the View Visitors form to displayed all the added Visitors.

File Forms

VisitorID	Date	TimeIn	Name	From Company	Vehicle Registr...	Telephone Nu...	Reason For V
-----------	------	--------	------	--------------	--------------------	-----------------	--------------

We will now check on the database schema “gateaccesscontrol” if all our details were entered correctly. With the Workbench still open from the previous task.(avoiding to log in over and over again)

126. Click on the Visitors table under the gateaccesscontrol schema. Right click and select refresh to refresh the table.

127. Right click on the columns on Visitors table and click on the Select Rows – limit to 1000

128. Notice the information that was entered on the Visitors form is displayed on the Visitors table on the schema.

usersvisitors

1

SELECT * FROM gateaccesscontrol.visitors;

Result Grid

Filter Rows:

Edit:

Export/Import:

Wrap Cell Content:

	VisitorID	Date	TimeIn	Name	From Company	Vehicle Registration Number	Telephone Number	Reason For Visit	Pe Se
*	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

visitors 1

Output

129. Click on the Users table under the gateaccesscontrol schema. Right click and select refresh the users table.
130. Right click on Columns and click Select Rows – Limit 1000
131. Notice the information that was entered on the Users form is displayed on the Users table on the schema.

The screenshot shows the SQL editor with the query `SELECT * FROM gateaccesscontrol.users;` entered. Below the editor, the Result Grid displays the data returned by the query. The grid has four columns: UserID, UserName, Password, and Role. The data is as follows:

	UserID	UserName	Password	Role
▶	4	MANDLA	67PMO	SECURITY
	5	MASIA	RRFYMFUJQT	MANAGER
	6	MONWABISI	6##7RTS2FN	SECURITY
	9	NTANDO	6@\$8@\$9	SECURITY
	10	SIHLE	7\$6~	STANDARD
*	NULL	NULL	NULL	NULL