Assignment 2

1. **Database Connection**

```
package Database;
import java.sql.*;
public class DBConnection {
   Connection conn;
   public Connection dbConnect(){
        try{
        Class.forName("com.mysql.jdbc.Driver");
      }catch(ClassNotFoundException cnfe){
        System.out.println(cnfe.getMessage());
      }
try{
        conn = DriverManager.getConnection("jdbc:mysql://localhost/company abc","root","");
      }catch(SQLException sqle){
        System.out.println(sqle.getMessage());
      }
      if(conn != null){
        System.out.println("Connection Successful");
      }else{
        System.out.println("Could not connect to database");
      }
      return conn;
   }
}
```

2. **Insert function**

```
private void insertBtnActionPerformed(java.awt.event.ActionEvent evt) {
      String username=Username.getText().toString();
```

```
    String password=Password.getText().toString();
    String role=Role.getText().toString();

    DBOperation dbo=new DBOperation();
    String result=dbo.insert(username, password, role);
    JOptionPane.showMessageDialog(this, result, "Operation Result" , WIDTH);
  }
```

3. **Select function**

```
private void loginDtmActionPerformed(java.awt.event.ActionEvent evt) {
    String newUsername = username.getText().toString();
    String newPassword = password.getText().toString();

    if(newUsername.contentEquals(user)){
      if(newPassword.contentEquals(pass)){
        JOptionPane.showMessageDialog(this, "Login Successful", user, WIDTH);

        Home home = new Home();
        home.setVisible(true);

        this.setVisible(false);

      }else{
        JOptionPane.showMessageDialog(this, "Password Incorrect", user, WIDTH);

      }
    }else{
      JOptionPane.showMessageDialog(this, "User not found", user, WIDTH);
    }
  }
```

4. Update function

```
  private void updateBtnActionPerformed(java.awt.event.ActionEvent evt) {

  }

try {

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

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

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

               break;

            }

        }

    } catch (ClassNotFoundException ex) {

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

    } catch (InstantiationException ex) {

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

    } catch (IllegalAccessException ex) {

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

    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

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

    }

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

        public void run() {

            new Update().setVisible(true);

        }

    });

  }
```

5. **Delete function**

```
private void deleteBtnActionPerformed(java.awt.event.ActionEvent evt) {

  }

  public static void main(String args[]) {

    try {

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

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

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

          break;

        }

      }

    } catch (ClassNotFoundException ex) {

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

    } catch (InstantiationException ex) {

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

    } catch (IllegalAccessException ex) {

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

    } catch (javax.swing.UnsupportedLookAndFeelException ex) {

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

    }


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

      public void run() {

        new Delete().setVisible(true);

      }
```

```
    });
}
```

6. Database

```
DROP DATABASE IF EXISTS hospital;
 CREATE DATABASE IF NOT EXISTS hospital;
 use hospital;

CREATE TABLE `users` (
  `username` int(22) NOT NULL,
  `password` varchar(22) NOT NULL,
  `role` varchar(22) NOT NULL
) ;
```