

LAPORAN PRAKTIKUM
PEMOGRAMAN BERORIENTASI OBJEK



OLEH:
TIARA AZIZAH
(2411533001)

DOSEN PENGAMPU:
NURFIAH, S.ST, M.Kom.

FAKULTAS TEKNOLOGI INFORMASI
DEPARTEMEN INFORMATIKA
UNIVERSITAS ANDALAS

2025

A. Pendahuluan

Database adalah kumpulan data yang terorganisasi dan terstruktur yang memiliki hubungan satu sama lain. Data-data ini disimpan secara elektronik dalam sistem komputer untuk memudahkan pengelolaannya.

XAMPP adalah paket software yang bersifat open source. XAMPP digunakan untuk menjalankan halaman web, MySQL, web yang bisa memanajemen basis data sehingga data bisa dimanipulasi.

MySQL merupakan relational database management system (RDBMS) open source. MySQL bisa digunakan untuk memanipulasi data seperti mengelola dan mengambil data dalam bentuk format table.

MySQL Connection/j adalah driver yang digunakan untuk menghubungkan aplikasi berbasis java dengan database MySQL, sehingga bisa berinteraksi seperti menyimpan, mengubah, mengambil dan menghapus data melalui java.

DAO(Data Acces Object) adalah objek yang menyediakan abstrak interface terhadap beberapa method yang berhubungan dengan database, seperti mengambil data(read), menyimpan data(create), menghapus (delete), mengubah (update)

Interface dalam bahasa java berarti mendefinisikan beberapa method abstrak yang harus diimplementasikan oleh class yang akan digunakan.

CRUD(Create, Read, Update, Delete) adalah fungsi dasar yang ada pada sebuah fungsi di database.

B. Tujuan

Pratikum ini bertujuan untuk membantu mahasiswa agar mampu membuat fungsi CRUD data user menggunakan database MySQL melalui Java.

C. Langkah-Langkah Pratikum

Pratikum ini merupakan lanjutan dari modul pratikum ke dua. Setelah membuat CRUD untuk user, modul pratikum ke dua mengarahkan untuk membuat fungsi CRUD di UserFrame.

1. Fungsi CRUD DAO di GUI

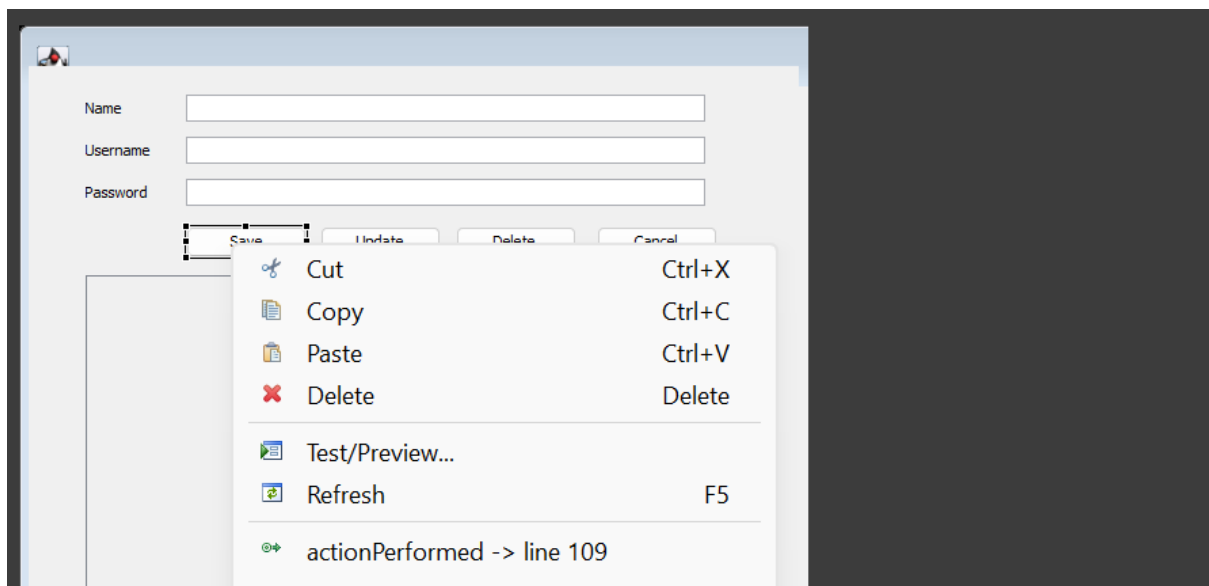
- a. Buka JFrame UserFrame setelah menyelesaikan semua program di UserRepo.
- b. Tambahkan method reset pada JFrame UserFrame untuk mengosongkan JText setelah mengklik button menu.

```
public void reset() {
    txtName.setText("");
    txtUsername.setText("");
    txtPassword.setText("");
}
```

- c. Buatlah list untuk menyimpan variabel yang akan didaftarkan dengan menuliskan kode berikut ini.

```
UserRepo usr = new UserRepo();
List<User> ls;
public String id;
```

- d. Pindah ke bagian desain pada JFrame untuk menambahkan action performed pada buttonsave. Penambahan action performed ini digunakan agar setiap kali menekan button akan terjadi penjalanan program.



```
JButton btnSave = new JButton("Save");
btnSave.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        User user = new User();
        user.setName(txtName.getText());
        user.setUsername(txtUsername.getText());
        user.setPassword(txtPassword.getText());
        usr.save(user);
        reset();
        loadTable();
    }
});
btnSave.setBounds(116, 118, 89, 23);
contentPane.add(btnSave);
```

- e. Tambahkan method loadTable. Method ini berfungsi untuk menampilkan list dalam bentuk table pada interface GUI.

```

public void loadTable() {
    ls = usr.show();
    TableUser tu = new TableUser(ls);
    tableUsers.setModel(tu);
    tableUsers.getTableHeader().setVisible(true);
}

```

- f. Selanjutnya pada class main, panggil method loadTable agar saat menjalankan interface GUI, table akan langsung ditambihkan di bagian JTable.

```

public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                UserFrame frame = new UserFrame();
                frame.loadTable();
                frame.setVisible(true);
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

```

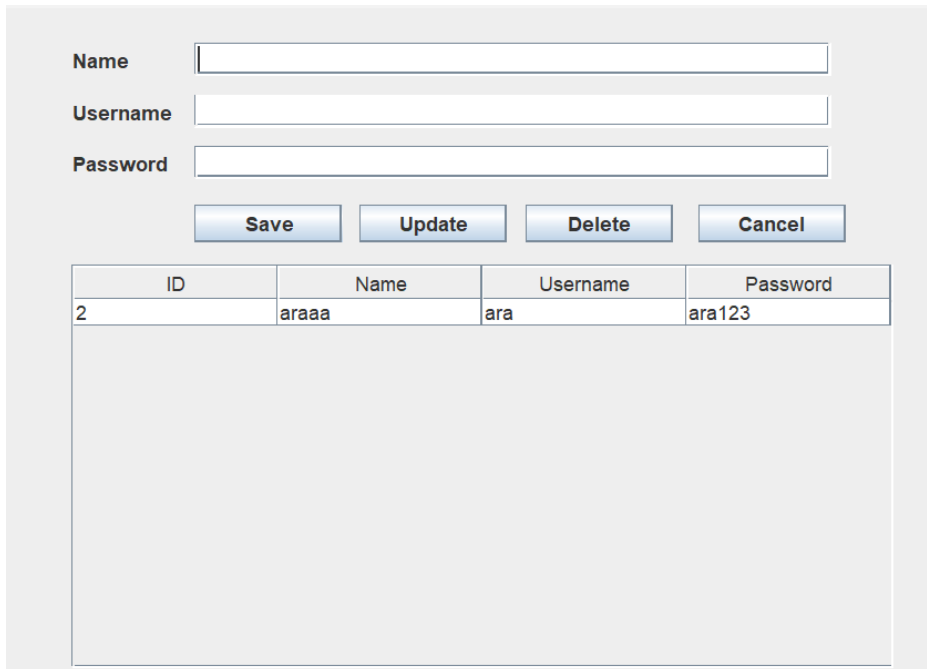
- g. Agar JTable pada interface GUI UserFrame bisa memuat semua list data dalam bentuk table yang bisa dipilih untuk dimodifikasi, tambahkan action mouseClicked yang berisikan program berikut ini. Penambahan JScrollPane pada program JTable berfungsi untuk kemampuan menggeser area JTable jika isi pada JTable terlalu banyak.

```

public void mouseClicked(MouseEvent e) {
    id = tableUsers.getValueAt(tableUsers.getSelectedRow(),0).toString();
    txtName.setText(tableUsers.getValueAt(tableUsers.getSelectedRow(),1).toString());
    txtUsername.setText(tableUsers.getValueAt(tableUsers.getSelectedRow(),2).toString());
    txtPassword.setText(tableUsers.getValueAt(tableUsers.getSelectedRow(),3).toString());
}
});
JScrollPane scrollPane = new JScrollPane(tableUsers);
scrollPane.setBounds(42, 154, 495, 243);
contentPane.add(scrollPane);

```

Tampilan dari program jTable ini adalah sebagai berikut,



The screenshot shows a Java Swing window titled "JTable". It contains a form with three text input fields labeled "Name", "Username", and "Password". Below these fields are four buttons: "Save", "Update", "Delete", and "Cancel". At the bottom of the window is a JTable with the following data:

ID	Name	Username	Password
2	aaaa	ara	ara123

- h. Tahapan berikutnya membuat actionPerformed untuk buttonupdate dan button delete.

Isikan program berikut ini, setelah membuat actionPerformed pada buttonupdate.

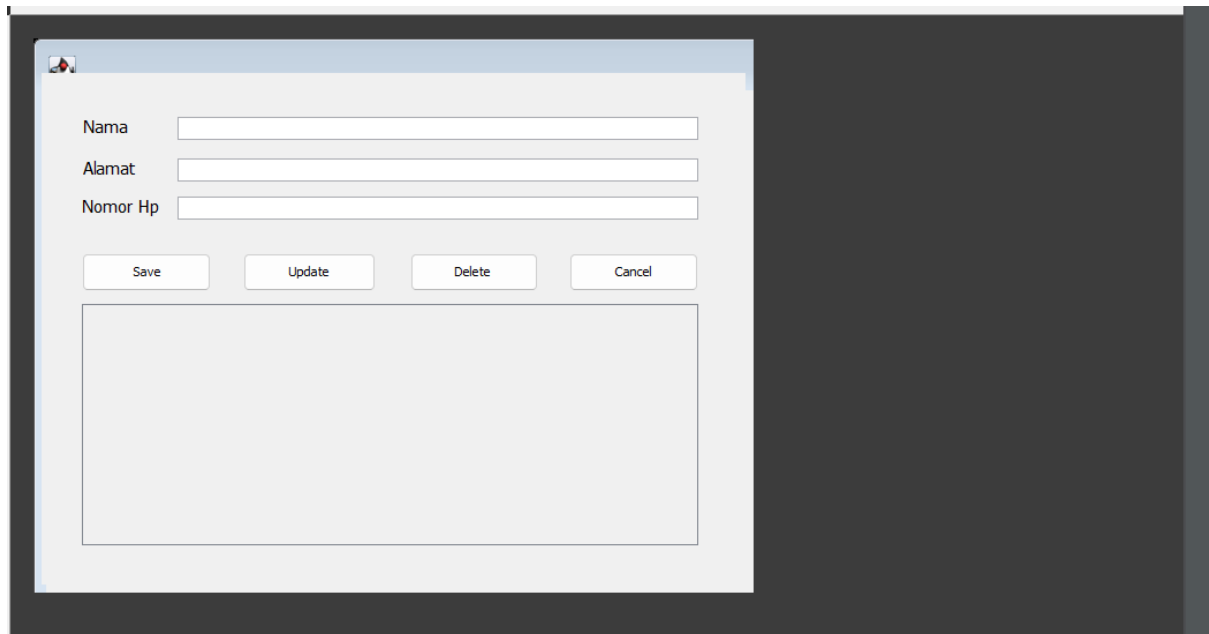
Program ini akan memperbarui table yang dipilih dari jTable.

```
 JButton btnUpdate = new JButton("Update");
 btnUpdate.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
         User user = new User();
         user.setNama(txtName.getText());
         user.setUsername(txtUsername.getText());
         user.setPassword(txtPassword.getText());
         user.setId(id);
         usr.update(user);
         reset();
         loadTable();
     }
 });
```

Penambahan JOptionPane berfungsi untuk memberikan tampilan pesan ketika data yang akan dihapus belum dipilih. Program actionPerformed pada buttonDelete adalah sebagai berikut,

```
 JButton btnDelete = new JButton("Delete");
 btnDelete.addActionListener(new ActionListener() {
     public void actionPerformed(ActionEvent e) {
         if(id != null) {
             usr.delete(id);
             reset();
             loadTable();
         } else {
             JOptionPane.showMessageDialog(null, "Silahkan pilih data yang akan dihapus");
         }
     }
 });
```

2. Selanjutnya untuk tugas pada modul dua adalah pembuatan CRUD pada Costumer dan Service. Untuk langkah-langkah yang digunakan kurang lebih sama dengan pembuatan CRUD di model User.
 - a. Buat frame baru di package ui. Frame baru ini bernama CostumerFrame.
 - b. Desain frame seperti gambar di bawah ini. Elemen yang digunakan adalah JTextField, JButton, dan JTable.



3. Tabel Model
 - a. Langkah selanjutnya membuat table model untuk costumer.
 - b. Kelas ini akan berfungsi sebagai model tabel pelanggan.
 - c. Setelah membuat class baru, isikan program yang bisa menghubungkan data user yang disimpan di List<Costumer> dengan visual tabel di package UI, seperti program di bawah ini. Kode program ini juga akan mengatur visual dan ukuran pada table yang akan ditampilkan.

```

package table;

import java.util.List;
import javax.swing.table.AbstractTableModel;
import model.Customer;

public class TableCustomer extends AbstractTableModel {

    private List<Customer> ls;
    private String[] columnNames = {"id", "nama", "alamat", "nomor_hp"};

    public TableCustomer(List<Customer> ls) {
        this.ls = ls;
    }

    public int getRowCount() {
        return ls.size();
    }

    public int getColumnCount() {
        return columnNames.length;
    }

    public String getColumnName(int column) {
        return columnNames[column];
    }

    public Object getValueAt(int rowIndex, int columnIndex) {
        switch (columnIndex) {
            case 0:
                return ls.get(rowIndex).getId();
            case 1:
                return ls.get(rowIndex).getNama();
            case 2:
                return ls.get(rowIndex).getAlamat();
            case 3:
                return ls.get(rowIndex).getNomor_hp();
            default:
                return null;
        }
    }
}

```

4. Membuat Fungsi DAO

- a. Di package DAO sebelumnya tambahkan class baru bernama CustomerDAO.
- b. Isi class CustomerDAO merupakan interface yang berfungsi sebagai pendefinisian metode yang harus diimplementasikan oleh class yang menggunakannya.

```

package DAO;

import java.util.List;

public interface CustomerDAO {
    void save(Customer customer);
    public List<Customer> show();
    public void delete(String id);
    public void update(Customer customer);
}

```

- c. Selanjutnya tambahkan class baru di dalam package, dengan nama CustomerRepo. Untuk program yang akan diisikan di dalam class ini adalah kode yang akan mengimplementasikan program dari class CustomerDAO. CustomerRepo akan mengelola semua program CRUD yang akan digunakan.

```

public class CostumerRepo implements CostumerDAO{
    private Connection connection;
    final String insert ="INSERT INTO costumer (nama, alamat, nomor_hp) VALUES (?,?,?);";
    final String select ="Select * from costumer;";
    final String delete ="DELETE from costumer where id=?;";
    final String update ="UPDATE costumer SET nama=?, alamat=?, nomor_hp=? WHERE id=?;";

    public CostumerRepo() {
        connection= database.koneksi();
    }
}

```

Program ini adalah variabel konstan yang menyimpan semua perintah SQL. Program ini juga memiliki konstruktor yang digunakan dalam class CostumerRepo untuk memanggil objek database.koneksi dan akan disimpan di variabel connection.

```

@Override
public void save(Costumer costumer) {
    // save
    PreparedStatement st=null;
    try {
        st = connection.prepareStatement(insert);
        st.setString(1, costumer.getNama());
        st.setString(2, costumer.getAlamat());
        st.setString(3, costumer.getNomor_hp());
        st.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            st.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

Method ini akan mengambil objek Costumer sebagai parameter dan menyimpan data yang akan diinputkan dengan menggunakan perintah insert. Blok try-catch-finally digunakan untuk menangani kesalahan dan memastikan resource ditutup.

```

@Override
//select
public List<Costumer> show(){
    List<Costumer> ls=null;
    try {
        ls = new ArrayList<>();
        Statement st = connection.createStatement();
        ResultSet rs =st.executeQuery(select);
        while(rs.next()) {
            Costumer costumer = new Costumer();
            costumer.setId(rs.getString("id"));
            costumer.setNama(rs.getString("nama"));
            costumer.setAlamat(rs.getString("alamat"));
            costumer.setNomor_hp(rs.getString("nomor_hp"));
            ls.add(costumer);
        }
    } catch (SQLException e) {
        Logger.getLogger(UserDAO.class.getName()).log(Level.SEVERE, null, e);
    }
    return ls;
}

```

Method show akan menampung sebuah arraylist dari objek Costumer. Perintah ini akan dijalankan menggunakan select untuk menampilkan semua data dari tabel pelanggan.


```

@Override
public void update(Costumer costumer) {
    PreparedStatement st = null;
    try {
        st = connection.prepareStatement(update);
        st.setString(1, costumer.getNama());
        st.setString(2, costumer.getAlamat());
        st.setString(3, costumer.getNomor_hp());
        st.setString(4, costumer.getId());
        st.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            st.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

Method update akan mengambil objek yang diperbarui dari User. PreparedStatement yang digunakan adalah update, mengatur data baru diinputkan dengan ID sebagai nilai unik untuk menentukan baris nilai yang akan diperbarui.

```

@Override
public void delete(String id) {
    PreparedStatement st = null;
    try {
        st = connection.prepareStatement(delete);
        st.setString(1, id);
        st.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            st.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

Method delete akan menghapus data dengan perintah PreparedStatementnya adalah delete. Id digunakan sebagai nilai unik yang akan menentukan baris data yang akan dihapus.

5. Menambahkan Fungsi CRUD pada GUI

- a. Tambahkan method reset pada JFrame CostumerFrame untuk mengosongkan JText setelah mengklik button menu.

```

public void reset() {
    txtnama.setText("");
    txtalamat.setText("");
    txtnomor_hp.setText("");
}

```

- b. Buatlah list untuk menyimpan variabel yang akan didaftarkan dengan menuliskan kode berikut ini.

```

    CostumerRepo cst = new CostumerRepo();
    List<Costumer> ls;
    public String id;

```

- c. Pindah ke bagian desain pada JFrame untuk menambahkan action performed pada buttonsave. Penambahan action performed ini digunakan agar setiap kali menekan button akan terjadi penjalanan program.

```

JButton btnsave = new JButton("Save");
btnsave.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        Costumer costumer = new Costumer();
        costumer.setNama(txtnama.getText());
        costumer.setAlamat(txtalamat.getText());
        costumer.setNomor_hp(txtnomor_hp.getText());
        cst.save(costumer);
        reset();
    }
});

```

- d. Tambahkan method loadTable. Method ini berfungsi untuk menampilkan list dalam bentuk table pada interface GUI.

```

public void loadTable() {
    ls = cst.show();
    TableCostumer tu = new TableCostumer(ls);
    tableCostumer.setModel(tu);
    tableCostumer.getTableHeader().setVisible(true);
}

```

- e. Selanjutnya pada class main, panggil method loadTable agar saat menjalankan interface GUI, table akan langsung ditambihkan di bagian JTable.

```

public static void main(String[] args) {
    EventQueue.invokeLater(new Runnable() {
        public void run() {
            try {
                CostumerFrame frame = new CostumerFrame();
                frame.setVisible(true);
                frame.loadTable();
            } catch (Exception e) {
                e.printStackTrace();
            }
        }
    });
}

```

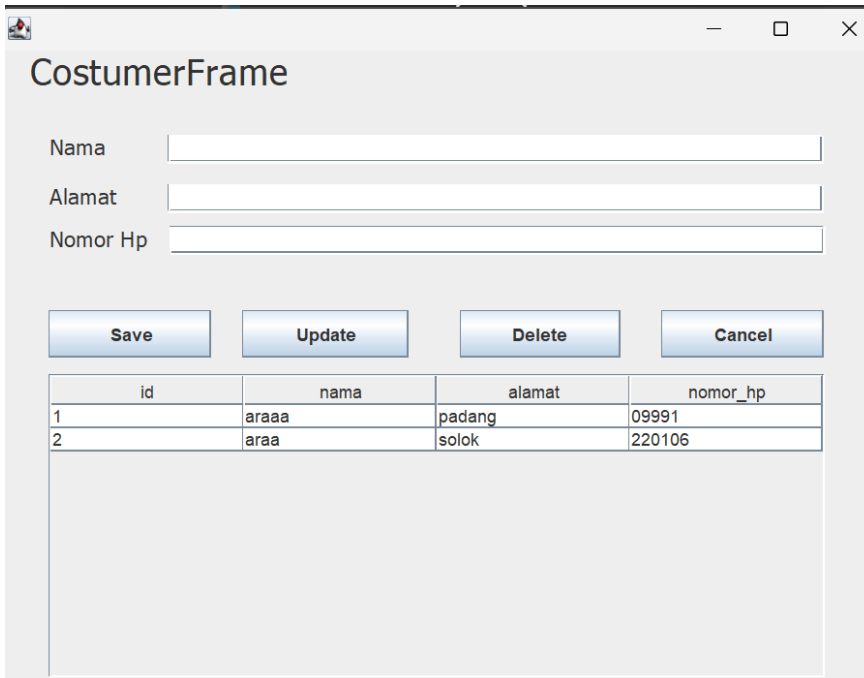
- f. Agar JTable pada interface GUI UserFrame bisa memuat semua list data dalam bentuk table yang bisa dipilih untuk dimodifikasi, tambahkan action mouseClicked yang berisikan program berikut ini. Penambahan JScrollPane pada program JTable berfungsi untuk kemampuan menggeser area JTable jika isi pada JTable terlalu banyak.

```

tableCostumer = new JTable();
tableCostumer.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        id = tableCostumer.getValueAt(tableCostumer.getSelectedRow(), 0).toString();
        txtnama.setText(tableCostumer.getValueAt(tableCostumer.getSelectedRow(),1).toString());
        txtalamat.setText(tableCostumer.getValueAt(tableCostumer.getSelectedRow(),2).toString());
        txtnomor_hp.setText(tableCostumer.getValueAt(tableCostumer.getSelectedRow(),3).toString());
    }
});
JScrollPane scrollPane = new JScrollPane(tableCostumer);
scrollPane.setBounds(36, 223, 535, 209);
contentPane.add(scrollPane);

```

Tampilan dari program JTable ini adalah sebagai berikut,



id	nama	alamat	nomor_hp
1	araaa	padang	09991
2	araa	solok	220106

- g. Tahapan berikutnya membuat actionPerformed untuk buttonupdate dan button delete.

Isikan program berikut ini, setelah membuat actionPerformed pada buttonupdate.

Program ini akan memperbarui table yang dipilih dari JTable.

```

JButton btnUpdate = new JButton("Update");
btnUpdate.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        Costumer costumer = new Costumer();
        costumer.setNama(txtnama.getText());
        costumer.setAlamat(txtalamat.getText());
        costumer.setNomor_hp(txtnomor_hp.getText());
        costumer.setId(id);
        cst.update(costumer);
        reset();
        loadTable();
    }
});
btnUpdate.setBounds(169, 179, 115, 33);

```

Penambahan JOptionPane berfungsi untuk memberikan tampilan pesan ketika data yang akan dihapus belum dipilih. Program actionPerformed pada buttonDelete adalah sebagai berikut,

```

JButton btnDelete = new JButton("Delete");
btnDelete.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(id != null) {
            cst.delete(id);
            reset();
            loadTable();
        } else {
            JOptionPane.showMessageDialog(null, "Silahkan pilih data yang akan dihapus");
        }
    }
});

```

6. Tampilan pada Database

a. Tampilan dari TableUser di Database.

☐ Show all

Number of rows: 25

Filter rows:

Extra options

<div><div>←</div><div>T</div><div>→</div></div>	<div>▼</div>	id	nama	username	password
<div><div><input type="checkbox"/></div><div> Edit</div><div> Copy</div><div> Delete</div></div>	2	aaaa	ara	ara123	







b. Tampilan dari TableCostumer di Database.

☐ Show all

Number of rows: 25

Filter rows:

Extra options

	id	nama	alamat	nomor_hp
<input type="checkbox"/>  Edit  Copy  Delete	1	aaaa	padang	09991
<input type="checkbox"/>  Edit  Copy  Delete	2	araa	solok	220106







c. Tampilan dari TableService di Database.

☐ Show all

Number of rows: 25

Filter rows:

Extra options

	id	jenis	harga	status
<input type="checkbox"/>  Edit  Copy  Delete	6	sweater rajut	Rp50.000	belum dicuci
<input type="checkbox"/>  Edit  Copy  Delete	9	rok rajut	Rp40.000	belum dicuci