

TUGAS PERTEMUAN 3

PRAKTIKUM PBO

Dosen Pengampu : Nurfiah, S.ST, M.Kom

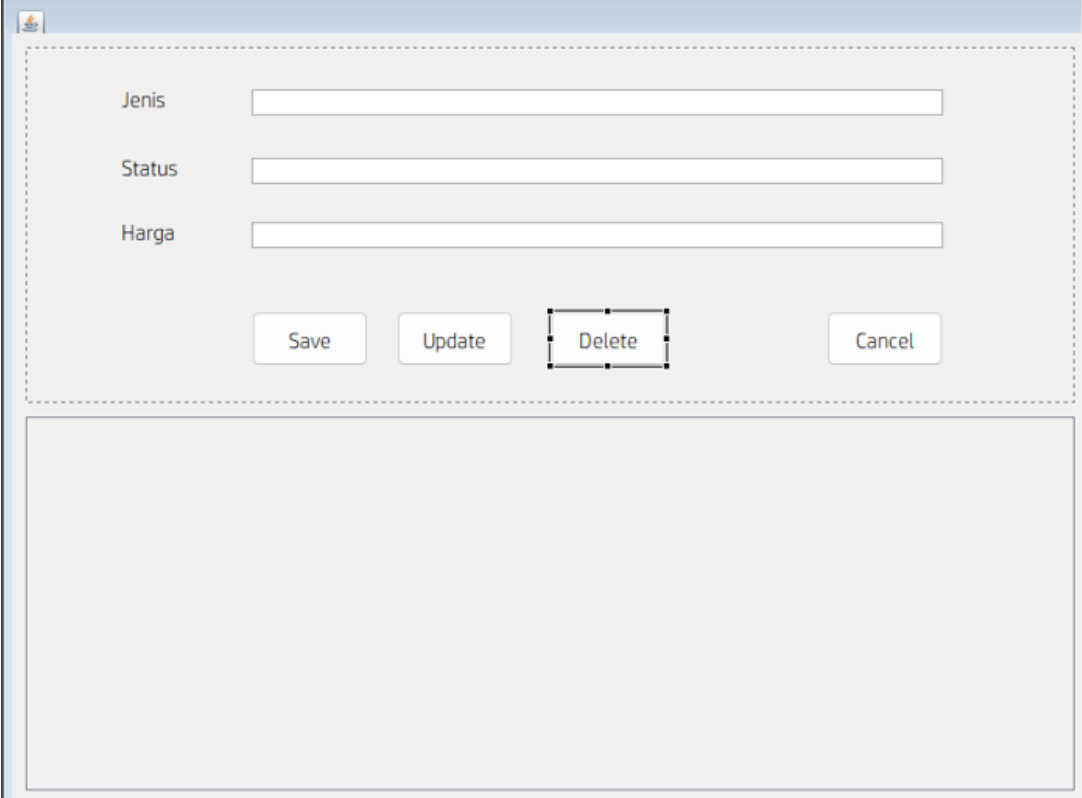


Disusun Oleh :

Yusuf Fatha Mubina Alexander (2211533017)

Departemen Informatika
Fakultas Teknologi Informasi
Universitas Andalas
2024

A. Fungsi CRUD untuk Layanan
1. Membuat UI ServiceFrame



The screenshot shows a Java Swing window titled "ServiceFrame". Inside the window, there is a form with three text input fields labeled "Jenis", "Status", and "Harga". Below these fields are four buttons: "Save", "Update", "Delete", and "Cancel". The "Delete" button is highlighted with a dashed border. The window has a standard Mac OS X title bar with a red, yellow, and green button on the left.

2. Membuat ServiceDAO dan ServiceRepo.

```
package DAO;

import java.util.List;

import model.Service;

public interface ServiceDAO {
    void save(Service service);
    public List<Service> show();
    public void delete (String id);
    public void update ([Service service]);
}
```

```

package DAO;

import java.sql.Connection;

public class ServiceRepo implements ServiceDAO{
    private Connection connection;
    final String insert = "INSERT INTO service (jenis, status, harga) VALUES (?, ?, ?);";
    final String select = "SELECT * FROM service;";
    final String delete = "DELETE FROM service WHERE id = ?;";
    final String update = "UPDATE service SET jenis = ?, status = ?, harga = ? WHERE id = ?;";

    public ServiceRepo() {
        connection = Database.koneksi();
    }

    @Override
    public void save(Service service) {
        PreparedStatement st = null;
        try {
            st = connection.prepareStatement(insert);
            st.setString(1, service.getJenis());
            st.setString(2, service.getStatus());
            st.setInt(3, service.getHarga());
            st.executeUpdate();

        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                st.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

```

    @Override
    public List<Service> show() {
        List<Service> ls = null;
        try {
            ls = new ArrayList<Service>();
            Statement st = connection.createStatement();
            ResultSet rs = st.executeQuery(select);
            while(rs.next()) {
                Service service = new Service();
                service.setId(rs.getString("id"));
                service.setJenis(rs.getString("jenis"));
                service.setStatus(rs.getString("status"));
                service.setHarga(rs.getInt("harga"));
                ls.add(service);
            }
        } catch (SQLException e) {
            Logger.getLogger(ServiceDAO.class.getName()).log(Level.SEVERE, null, e);
        }
        return ls;
    }

    @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();
            }
        }
    }
}

```

```

@Override
public void update(Service service) {
    PreparedStatement st = null;
    try {
        st = connection.prepareStatement(update);
        st.setString(1, service.getJenis());
        st.setString(2, service.getStatus());
        st.setInt(3, service.getHarga());
        st.setString(4, service.getId());
    } catch (SQLException e) {
        e.printStackTrace();
    } finally {
        try {
            st.close();
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}
}

```

Ada sedikit perubahan di bagian getHarga. Karena jenis variable getHarga adalah integer, maka dibutuhkan setInt untuk mengambil nilai dari table service.

3. Membuat TableService

```

public class TableService extends AbstractTableModel{
    List<Service> ls;
    private String[] columnNames = {"ID", "Jenis", "Status", "Harga"};
    public TableService(List<Service> ls) {
        this.ls = ls;
    }

    @Override
    public int getRowCount() {
        // TODO Auto-generated method stub
        return ls.size();
    }

    @Override
    public int getColumnCount() {
        // TODO Auto-generated method stub
        return 4;
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        // TODO Auto-generated method stub
        switch (columnIndex) {
            case 0:
                return ls.get(rowIndex).getId();
            case 1:
                return ls.get(rowIndex).getJenis();
            case 2:
                return ls.get(rowIndex).getStatus();
            case 3:
                return ls.get(rowIndex).getHarga();
            default:
                return null;
        }
    }
}

```

4. Memasukkan fungsi ServiceDAO kedalam tombol ServiceFrame.

- Tombol Save

```
JButton btnSave = new JButton("Save");
btnSave.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        Service service = new Service();
        service.setJenis(txtJenis.getText());
        service.setStatus(txtStatus.getText());
        service.setHarga(Integer.parseInt(txtHarga.getText()));
        srv.save(service);
        reset();
        loadTable();
    }
});
btnSave.setFont(new Font("HP Simplified Jpan Light", Font.PLAIN, 16));
btnSave.setBounds(165, 193, 85, 40);
panel.add(btnSave);
```

- Tombol Update

```
JButton btnUpdate = new JButton("Update");
btnUpdate.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        Service service = new Service();
        service.setJenis(txtJenis.getText());
        service.setStatus(txtStatus.getText());
        service.setHarga(Integer.parseInt(txtHarga.getText()));
        service.setId(id);
        srv.update(service);
        reset();
        loadTable();
    }
});
btnUpdate.setFont(new Font("HP Simplified Jpan Light", Font.PLAIN, 16));
btnUpdate.setBounds(271, 193, 85, 40);
panel.add(btnUpdate);
```

- Tombol Delete

```
JButton btnDelete = new JButton("Delete");
btnDelete.addActionListener(new ActionListener() {
    public void actionPerformed(ActionEvent e) {
        if(id != null) {
            srv.delete(id);
            reset();
            loadTable();
        } else {
            JOptionPane.showMessageDialog(null, "Silahkan pilih data yang akan dihapus");
        }
    }
});
btnDelete.setFont(new Font("HP Simplified Jpan Light", Font.PLAIN, 16));
btnDelete.setBounds(383, 193, 85, 40);
panel.add(btnDelete);
```


- Membuat table dapat diklik dan masuk kedalam JTextField.

```
tableService = new JTable();
tableService.addMouseListener(new MouseAdapter() {
    @Override
    public void mouseClicked(MouseEvent e) {
        id = tableService.getValueAt(tableService.getSelectedRow(), 0).toString();
        txtJenis.setText(tableService.getValueAt(tableService.getSelectedRow(), 1).toString());
        txtStatus.setText(tableService.getValueAt(tableService.getSelectedRow(), 2).toString());
        txtHarga.setText(tableService.getValueAt(tableService.getSelectedRow(), 3).toString());
    }
});
scrollPane.setViewportView(tableService);
```

- Menampilkan table, memasukkan interface, dan reset.

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

public void reset() {
    txtJenis.setText("");
    txtStatus.setText("");
    txtHarga.setText("");
}

ServiceRepo srv = new ServiceRepo();
List<Service> ls;
public String id;

public void loadTable() {
    ls = srv.show();
    TableService ts = new TableService(ls);
    tableService.setModel(ts);
    tableService.getTableHeader().setVisible(true);
}
}
```

B. Fungsi CRUD untuk Pelanggan

1. Membuat UI CostumerFrame

The image shows a Java Swing window titled "CustomerFrame". The window has a light blue title bar and a light gray background. It contains a dashed-line border enclosing a form area. Inside the form area, there are three text input fields labeled "Nama Pelanggan", "Alamat", and "No. HP". Below these fields are four buttons: "Save", "Update", "Delete", and "Cancel". Below the dashed border is a large, empty rectangular area.

2. Membuat TableCostumer

```

1 package table;
2
3 import java.util.List;
4
5
6
7
8
9 public class TableCostumer extends AbstractTableModel {
10     List<Costumer> ls;
11     private String[] columnNames = {"ID", "Nama", "Alamat", "Nohp"};
12     public TableCostumer(List<Costumer> ls) {
13         this.ls = ls;
14     }
15
16     @Override
17     public int getRowCount() {
18         return ls.size();
19     }
20
21     @Override
22     public int getColumnCount() {
23         return 4;
24     }
25
26     @Override
27     public String getColumnName(int column) {
28         return columnNames[column];
29     }
30
31     @Override
32     public Object getValueAt(int rowIndex, int columnIndex) {
33         switch (columnIndex) {
34             case 0:
35                 return ls.get(rowIndex).getId();
36             case 1:
37                 return ls.get(rowIndex).getNama();
38             case 2:
39                 return ls.get(rowIndex).getAlamat();
40             case 3:
41                 return ls.get(rowIndex).getNohp();
42             default:
43                 return null;
44         }
45     }
46 }

```

3. Membuat CostumerDAO dan CostumerRepo


```

package DAO;

import java.util.List;

import model.Costumer;

public interface CostumerDAO {
    void save(Costumer costumer);
    public List<Costumer> show();
    public void delete (String id);
    public void update (Costumer costumer);
}

```

```

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

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

    @Override
    public void save(Costumer costumer) {
        PreparedStatement st = null;
        try {
            st = connection.prepareStatement(insert);
            st.setString(1, costumer.getNama());
            st.setString(2, costumer.getAlamat());
            st.setString(3, costumer.getNohp());
            st.executeUpdate();

        } catch (SQLException e) {
            e.printStackTrace();
        } finally {
            try {
                st.close();
            } catch (SQLException e) {
                e.printStackTrace();
            }
        }
    }
}

```

```

public List<Costumer> show(){
    List<Costumer> ls = null;
    try {
        ls = new ArrayList<Costumer>();
        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.setNohp(rs.getString("nohp"));
            ls.add(costumer);
        }
    }catch(SQLException e) {
        Logger.getLogger(CostumerDAO.class.getName()).log(Level.SEVERE, null, e);
    }
    return ls;
}

@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.getNohp());
        st.setString(4, costumer.getId());
    }catch(SQLException e) {
        e.printStackTrace();
    }finally {
        try {
            st.close();
        }catch(SQLException e){
            e.printStackTrace();
        }
    }
}
}

```

```

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();
        }
    }
}
}

```

4. Membuat Fungsi CRUD di JFrame
 - Tombol Save

```

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.setNohp(txtNohp.getText());
        cst.save(costumer);
        reset();
        loadTable();
    }
});
btnSave.setFont(new Font("HP Simplified Jpan Light", Font.PLAIN, 16));
btnSave.setBounds(165, 193, 85, 40);
panel.add(btnSave);

```

- Tombol Update

```

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.setNohp(txtNohp.getText());
        costumer.setId(id);
        cst.update(costumer);
        reset();
        loadTable();
    }
});
btnUpdate.setFont(new Font("HP Simplified Jpan Light", Font.PLAIN, 16));
btnUpdate.setBounds(271, 193, 85, 40);
panel.add(btnUpdate);

```

- Tombol Delete

```

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");
        }
    }
});
btnDelete.setFont(new Font("HP Simplified Jpan Light", Font.PLAIN, 16));
btnDelete.setBounds(383, 193, 85, 40);
panel.add(btnDelete);

```

- Membuat fungsi klik pada table

```

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());
        txtNohp.setText(tableCostumer.getValueAt(tableCostumer.getSelectedRow(), 3).toString());
    }
});
scrollPane.setViewportView(tableCostumer);

```

- Menampilkan table saat pertama kali run, method reset, dan memasukkan interface.

```

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();
            }
        }
    });
}

public void reset() {
    txtNama.setText("");
    txtAlamat.setText("");
    txtNohp.setText("");
}

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

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

```