TUGAS FINAL

PEMROGRAMAN BERORIENTASI OBJEK

Raya Srikandi

E1E20046

JURUSAN TEKNIK INFORMATIKA

FAKULTAS TEKNIK

UNIVERSITAS HALUOOLEO

2022

A. UML *(unified modeling language)*

UML (Unified Modeling Language) adalah metode pemodelan yang digunakan untuk memvisualisasikan desain sistem berorientasi objek atau yang biasa kita sebut OOP.

a. Use case diagram

Use case diagram adalah salah satu dari berbagai jenis diagram Unified Modeling Language (UML) yang menggambarkan interaksi antara sistem dan aktornya. Sebuah use case dapat menggambarkan jenis interaksi antara pengguna sistem dan sistem.

Fungsi :

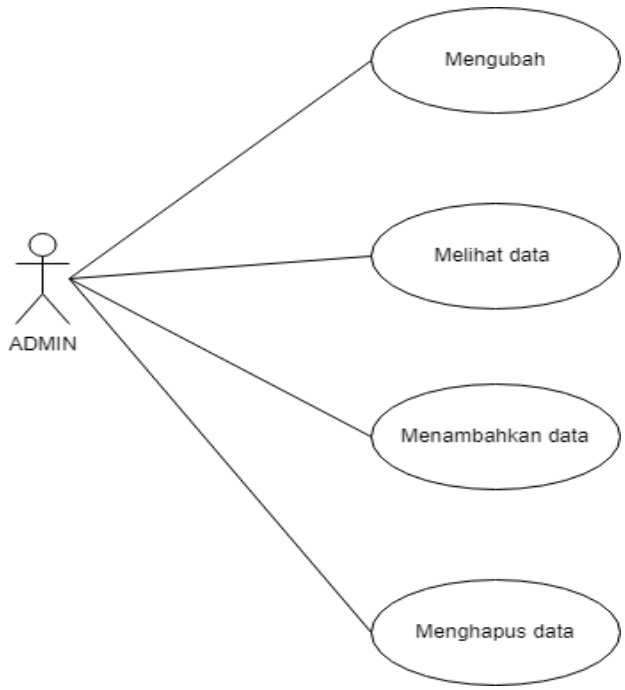
1. Menunjukkan urutan kegiatan proses dalam suatu sistem

Fungsi pertama dapat memperkenalkan tahap awal dari setiap aktivitas proses dalam sistem yang dikembangkan. Ini dapat membantu pengembang dengan mudah mengidentifikasi kebutuhan berdasarkan perangkat lunak dan pengguna.

2. Mendeskripsikan proses bisnis dalam sistem

Dapat menggambarkan urutan proses bisnis secara lebih jelas dan transparan untuk mencegah terjadinya kesalahan pada sistem yang akan dibangun.

Berikut use case pengisisan data mahasiswa:



Gambar 1 Usecase admin

Penjelasan :

Nama: Use case diagram ini menggambarkan interaksi antara sistem dengan aktor "Admin" dalam mengelolah biodata mahasiswa.

- Mengubah data: Action ini menggambarkan proses dimana admin mengubah data mahasiswa yang ada dalam sistem.
- Melihat data: Action ini menggambarkan proses dimana admin dapat melihat data mahasiswa yang ada dalam sistem.
- Menambah data: Action ini menggambarkan proses dimana admin menginputkan data mahasiswa ke sistem
- Menghapus data: Action ini menggambarkan proses dimana admin dapat menghapus data mahasiswa yang ada dalam sistem

Dengan menggunakan use case diagram ini, kita dapat mengetahui interaksi yang terjadi antara sistem dengan admin dalam menambahkan data, mengubah data, menghapus data, dan melihat data yang ada. Diagram ini juga dapat membantu dalam memahami proses bisnis atau sistem yang akan dibuat dengan lebih jelas.

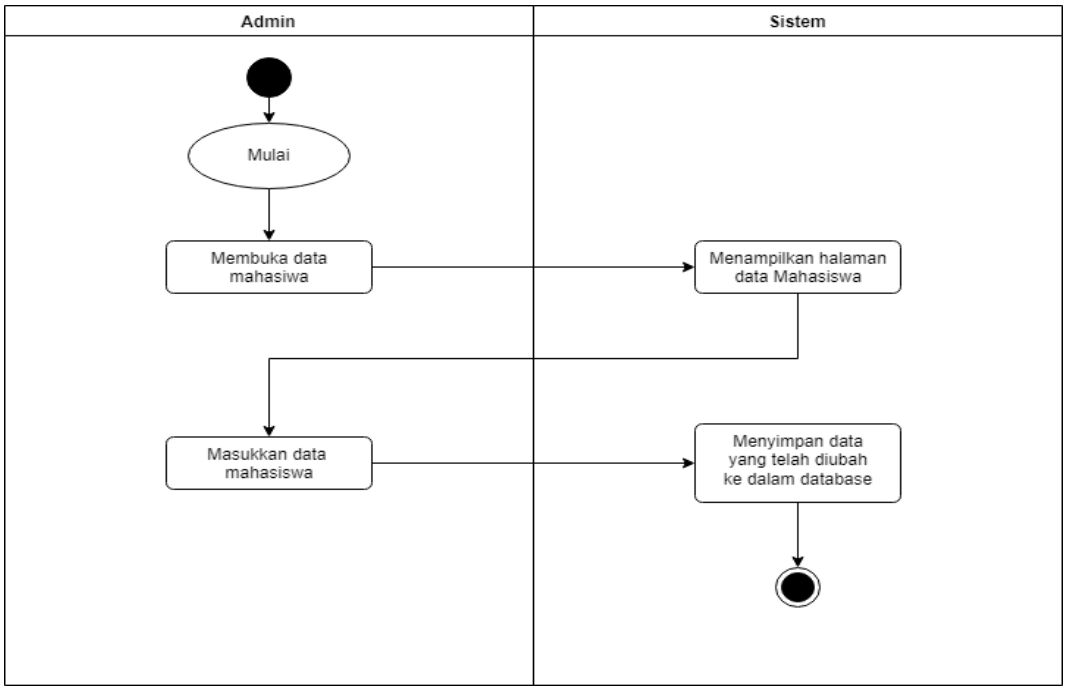
b. Activity Diagram

Activity Diagram adalah bentuk visual dari alur kerja. Diagram pemodelan ini berisi urutan proses dari suatu sistem berupa gambar vertikal. Dalam Activity Diagram, konten alur kerja dapat berupa aktivitas dan tindakan, pilihan, atau pengulangan. Activity Diagram dapat digunakan untuk mengidentifikasi atau mengelompokkan aliran yang terlihat dari sistem tertentu. Dalam Activity Diagram, terdapat panah yang memandu urutan operasi dari awal hingga akhir. Beberapa tujuan dan fungsi dari Activity Diagram adalah:

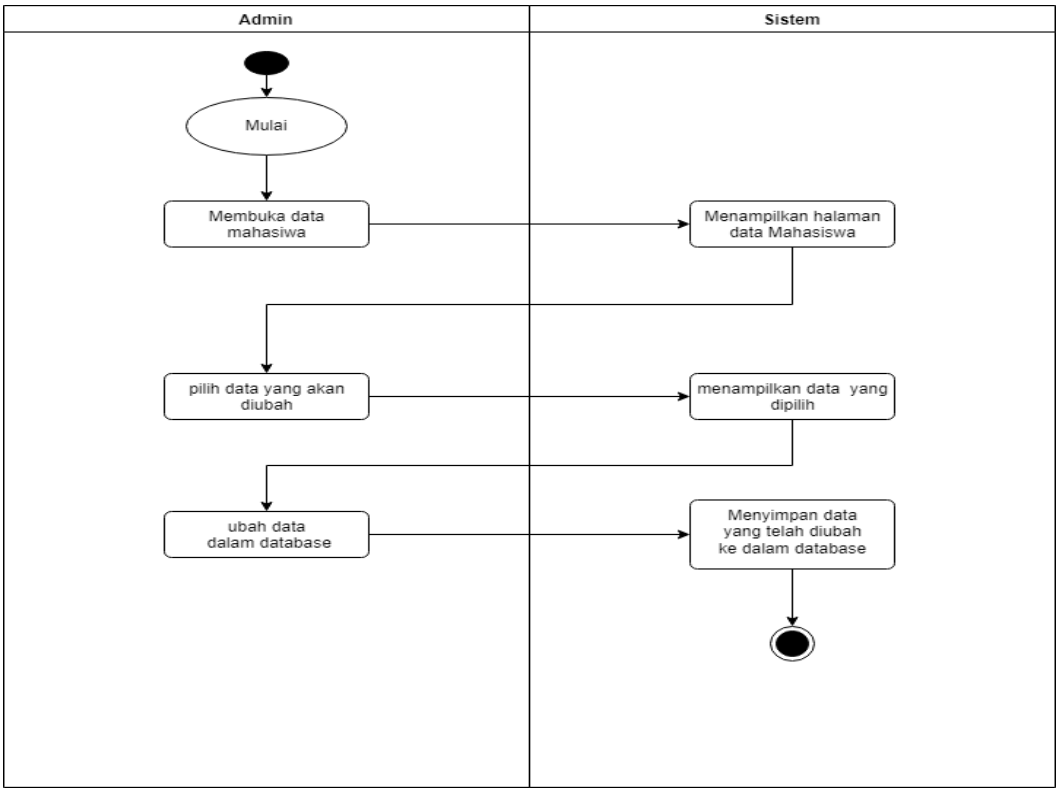
a. Dapat menjelaskan urutan kegiatan dalam suatu proses
b. Bisa digunakan untuk model
c. Terbuat dari satu atau lebih kasus penggunaan
d. Membantu memahami keseluruhan proses
e. Struktur metode desain, mirip dengan flowchart
f. Dapat menemukan aktivitas pengguna berdasarkan kasus penggunaan yang dibuat sebelumnya

g. Menggambarkan aliran paralel, bercabang

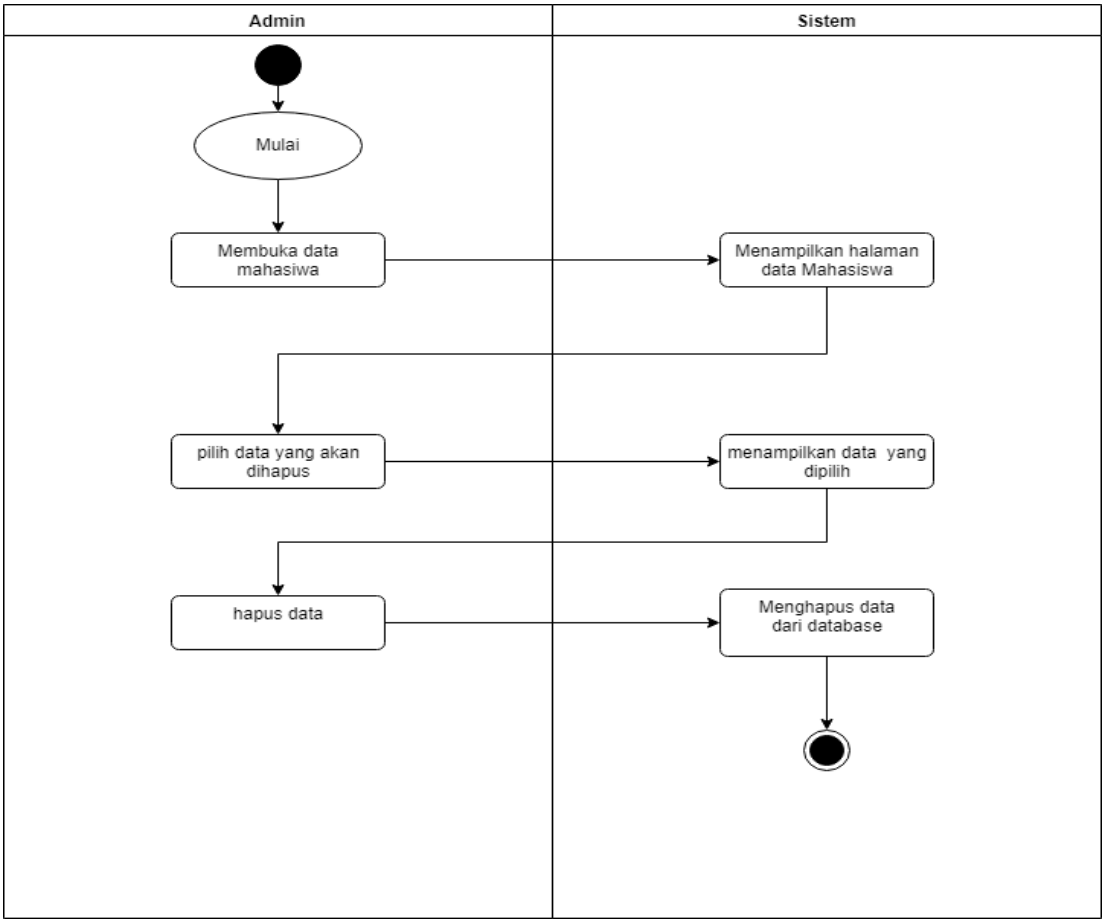
Berikut ini adalah activity diagram pengisisan data mahasiswa :



Gambar 3 Activity diagram untuk Menambah data



Gambar 2 Activity diagram untuk mengubah data



Gambar 4 Activity diagram untuk menghapus data

Penjelasan :

a) Activity Diagram Menambah data

1. Pertama, admin membuka system data mahasiswa
2. Setelah itu, admin diminta untuk menginputkan data mahasiswa dengan menginputkannya pada kolom yang tersedia di tampilan system.
3. Setelah seluruh data terisi, admin dapat menambahkan data dengan mengklik tombol "Tambah" pada sistem.

b) Activity Diagram Mengubah data

1. Pertama, admin membuka system data mahasiswa
2. Setelah itu, admin diminta untuk menginputkan ulang data mahasiswa dengan menginputkannya pada kolom yang tersedia di tampilan system atau bisa mengklik salah satu data mahasiswa yang ada pada tabel.

3. Setelah seluruh data terisi, admin dapat menambahkan data dengan mengklik tombol "Edit" pada sistem.

c) Activity Diagram Menghapus data

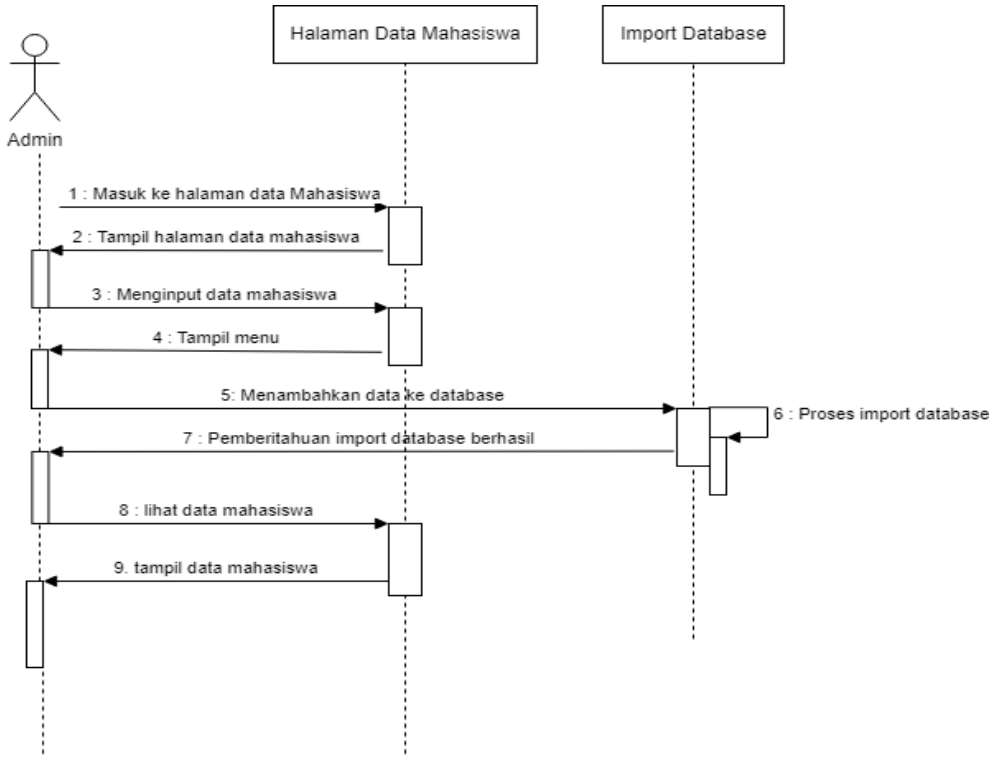
1. Pertama, admin membuka system data mahasiswa
2. Setelah itu, admin diminta untuk menginputkan Nim mahasiswa yang ada di database dengan menginputkannya pada kolom yang tersedia di tampilan system atau bisa mengklik salah satu data mahasiswa yang ada pada tabel.
3. Setelah seluruh data terisi, admin dapat menambahkan data dengan mengklik tombol "hapus" pada sistem.

c. Sequence Diagram

Sequence Diagram adalah diagram interaktif yang merinci bagaimana suatu operasi dilakukan. Sequence diagram atau diagram urutan menggambarkan interaksi antar kelas dalam hal pertukaran pesan dari waktu ke waktu. Sequence diagram juga terkadang disebut event diagram. Keuntungan dari Sequence Diagram adalah sebagai berikut :

1. Sequence diagram adalah salah satu jenis diagram UML yang banyak digunakan sebagai acuan untuk bisnis dan aktivitas lainnya serta manfaat dari sequence diagram.
2. Mewakili kasus penggunaan UML terperinci.
3. Buat model logika dari fungsi, operasi, atau prosedur yang kompleks.
4. Menunjukkan bagaimana objek dan komponen harus berinteraksi untuk menyelesaikan suatu proses.
5. Merencanakan dan memahami fungsi rinci situasi saat ini atau masa depan.

Berikut Sequence Diagram pengisisan data mahasiswa:



Gambar 5 Sequence Diagram data mahasiswa

Penjelasan :

- Sistem menampilkan form pendataan.
- Admin memasukkan menginputkan data mahasiswa seperti Nama, NIM, Jurusan, Alamat, dan nomor telefon ke dalam form.
- Sistem menerima inputan dari pengguna.
- Sistem memvalidasi inputan yang diberikan oleh pengguna.
- Jika inputan valid, maka sistem akan menyimpan data pendaftaran ke database lalu menampilkan pemberitahuan berhasil kepada admin.

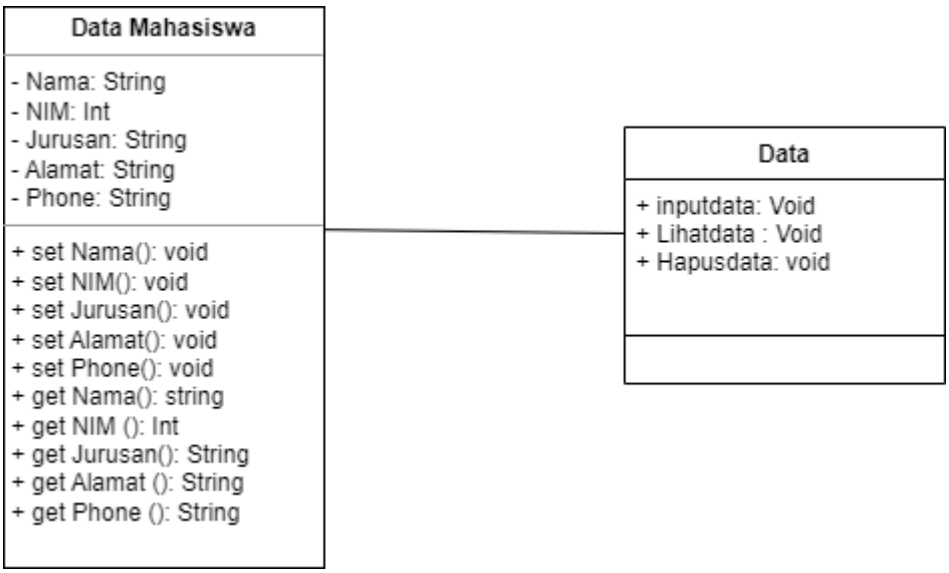
d. Class Diagram

Diagram kelas atau class diagram adalah jenis diagram struktur dalam UML yang secara jelas menggambarkan struktur dan menggambarkan kelas, properti, metode, dan hubungan dari setiap objek. Bersifat statis, dalam arti diagram kelas tidak menjelaskan apa yang terjadi ketika kelas-kelas saling berhubungan, tetapi menjelaskan hubungan mana yang terjadi.

Penggunaan diagram kelas membawa banyak keuntungan bagi proses pengembangan perangkat lunak dan dalam bisnis. Berikut adalah manfaat dari diagram kelas:

- Diagram kelas digunakan untuk menjelaskan model data untuk program, baik model data sederhana maupun kompleks.
- Buat garis besar diagram aplikasi dengan jelas dan lebih baik.
- Membantu kamu untuk menyampaikan kebutuhan dari suatu sistem.

Berikut Class Diagram pengisisan data mahasiswa:



Gambar 6 Class Diagram data mahasiswa

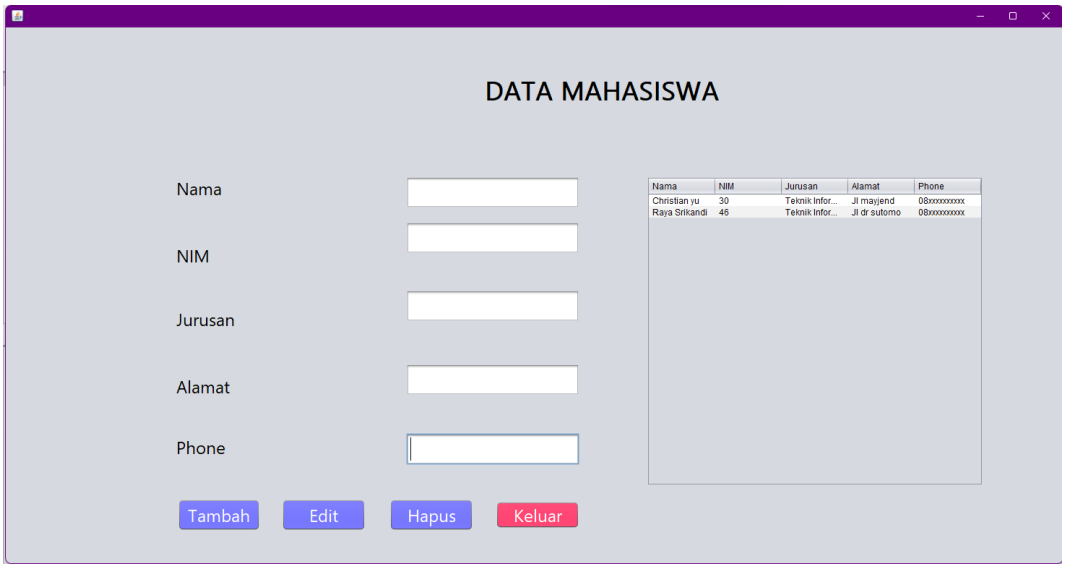
Penjelasan :

- Class data mahasiswa adalah class yang menyimpan informasi tentang data mahasiswa, termasuk Nama, NIM, Jurusan, Alamat, dan Phone.
- Atribut Nama, Jurusan, Alamat, dan Phone pada class Mahasiswa bertipe data string, NIM bertipe data integer dan bertujuan untuk menyimpan informasi data mahasiswa.
- Method setNama(), setNIM(), setJurusan(), setAlamat(), dan setPhone() pada class Mahasiswa bertujuan untuk mengatur informasi data mahasiswa.

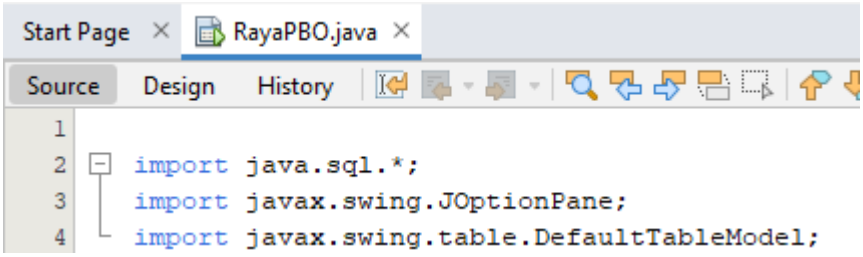
- Method getNama(), getNIM(), getJurusan(), getAlamat(), dan getPhone() pada class Mahasiswa bertujuan untuk mengambil informasi data mahasiswa.
- Class data adalah class yang menyimpan informasi tentang proses tambah, edit, hapus dan liat data mahasiswa.
- Method inputData() pada class data bertujuan untuk memasukkan data mahasiswa ke dalam sistem.
- Method lihatData() pada class Biodata bertujuan untuk menampilkan data mahasiswa yang telah dimasukkan ke dalam sistem.

B. Penjelasan kodingan

Berikut adalah tampilan system inputan data mahasiswa yang telah saya buat.



1. Import



```
public class RayaPBO extends javax.swing.JFrame {

    Connection con;
    Statement st;
    ResultSet rs;
```

Program ini terlihat mengimpor pustaka Java seperti java.sql yang menunjukkan bahwa program ini akan menggunakan fitur-fitur terkait dengan basis data (SQL).

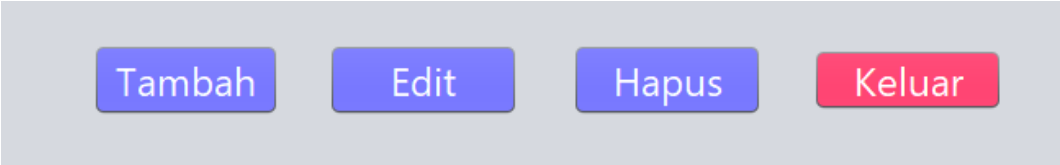
Secara khusus, kode di atas terlihat beberapa kelas yang terkait dengan JDBC (Java Database Connectivity), yaitu Connection, statement, dan resultSet yang merupakan kelas - kelas yang biasa digunakan untuk mengakses dan mengelola data di dalam basis data dari aplikasi Java. Pada kodingan di atas terdapat sebuah class Java yang diberi nama "RayaPBO" yang merupakan turunan dari class "JFrame".

```
    public RayaPBO() {
        initComponents();
        try {
         Class.forName("com.mysql.cj.jdbc.Driver");
         String s = "jdbc:mysql://localhost:3306/db_rayapbo";
         con = DriverManager.getConnection(s,"root","");
         tampilData();


      } catch (Exception e) {
         JOptionPane.showMessageDialog(null, e.getMessage());
      }
    }
```

Class ini juga memiliki konstruktor bernama "RayaPBO" yang akan dieksekusi saat membuat objek kelas ini. Konstruktor memanggil metode "initComponents" yang merupakan metode yang dibuat oleh IDE NetBeans dan digunakan untuk menginisialisasi komponen dalam antarmuka pengguna grafis (GUI) yang dibuat dengan NetBeans. Selain itu, konstruktor memanggil metode "tampilData", yang ditentukan oleh pembuat kelas ini dan fungsinya tidak diketahui secara pasti.

2. Button



a. Button Tambah

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        String Nama=nama.getText();
        int NIM=Integer.parseInt(nim.getText());
        String Jurusan=jurusan.getText();
        String Alamat=alamat.getText();
        String Phone=phone.getText();
        String sql="INSERT INTO `tb_data` (`Nama`, `NIM`, `Jurusan`, `Alamat`, `Phone`) "
                +" VALUES (?,?,?,?,?)";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1, Nama);
        pstmt.setInt(2, NIM);
        pstmt.setString(3, Jurusan);
        pstmt.setString(4, Alamat);
        pstmt.setString(5, Phone);
        pstmt.execute();
        resetTampilanTabel();
        JOptionPane.showMessageDialog(null, "Data Berhasil di ditambahkan");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
}
```

Program ini digunakan untuk menambahkan data ke dalam database. Pada kodingan ini terdapat sebuah percobaan (try) yang akan menjalankan sebuah perintah insert ke dalam tabel tb_data di database. Perintah insert akan menambahkan data baru yang diambil dari beberapa komponen input (Nama, NIM, Jurusan, Alamat, dan Phone). Apabila perintah insert tersebut berhasil dijalankan, maka akan ditampilkan sebuah pesan "Data Berhasil Ditambahkan" dan akan dipanggil fungsi resetTampilanTabel(). Namun jika terjadi error saat mengeksekusi perintah insert, maka akan ditampilkan pesan informasi error akan dicetak ke sistem output (System.out.println(e.getMessage())).

b. Button Edit

```
private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        String Nama=this.nama.getText();
        int NIM=Integer.parseInt(this.nim.getText());
        String Jurusan=this.jurusan.getText();
        String Alamat=this.alamat.getText();
        String Phone=this.phone.getText();
        String sql="UPDATE tb_data SET Nama = ?, Jurusan = ?, Alamat = ?,Phone = ? WHERE tb_data.NIM = ?";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setString(1, Nama);
        pstmt.setString(2, Jurusan);
        pstmt.setString(3, Alamat);
        pstmt.setString(4, Phone);
        pstmt.setInt(5, NIM);
        pstmt.executeUpdate();
        resetTampilanTabel();
        JOptionPane.showMessageDialog(null, "Data Berhasil di ubah");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
}
```

Program ini digunakan untuk Mengubah data yang ada dalam database. Pada kodingan ini terdapat sebuah percobaan (try) yang akan menjalankan sebuah perintah Update ke dalam tabel tb_data di database. Perintah Update akan menambahkan data baru yang telah diubah dan diambil dari beberapa komponen input (Nama, NIM, Jurusan, Alamat, dan Phone). Apabila perintah Update tersebut berhasil dijalankan, maka akan ditampilkan sebuah pesan "Data Berhasil di ubah" dan akan dipanggil fungsi resetTampilanTabel(). Namun jika terjadi error saat mengeksekusi perintah update, maka akan ditampilkan pesan informasi error akan dicetak ke sistem output (System.out.println(e.getMessage())).

c. Button Hapus

```
private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    try {
        int NIM=Integer.parseInt(this.nim.getText());
        String sql="DELETE FROM tb_data WHERE NIM  = ?";
        PreparedStatement pstmt=con.prepareStatement(sql);
        pstmt.setInt(1, NIM);
        pstmt.execute();
        resetTampilanTabel();
        JOptionPane.showMessageDialog(null, "Data Berhasil di dihapus");
    } catch (Exception e) {
        JOptionPane.showMessageDialog(null, e.getMessage());
    }
}
```

Program ini digunakan untuk Menghapus data yang ada dalam database. Pada kodingan ini terdapat sebuah percobaan (try) yang akan menjalankan sebuah perintah Delete ke dalam tabel tb_data di database. Perintah Delete akan menghapus data yang diambil dari komponen input NIM, Apabila perintah Update tersebut berhasil dijalankan, maka akan ditampilkan sebuah pesan "Data Berhasil di hapus" dan akan dipanggil fungsi resetTampilanTabel(). Namun jika terjadi error saat mengeksekusi perintah hapus, maka akan ditampilkan pesan informasi error akan dicetak ke sistem output (System.out.println(e.getMessage())).

d. Button Keluar

```
private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    dispose();
}
```

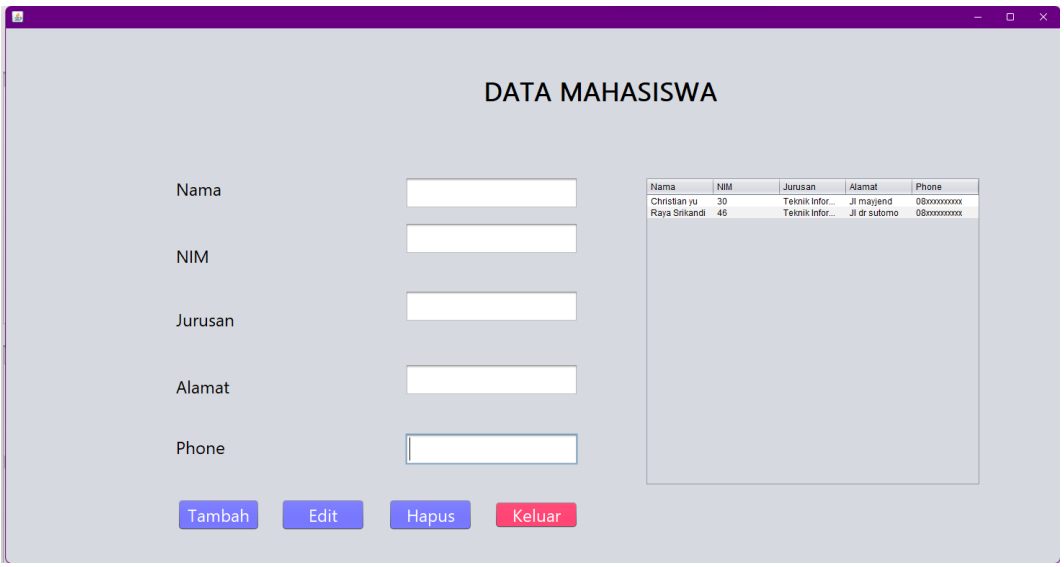
dispose(); menyebabkan halaman pengisian data mahasiswa dibersihkan oleh sistem operasi. Menurut dokumentasi, ini dapat menyebabkan Java VM berakhir jika tidak ada Windows lain yang tersedia.

3. tableMouseClicked

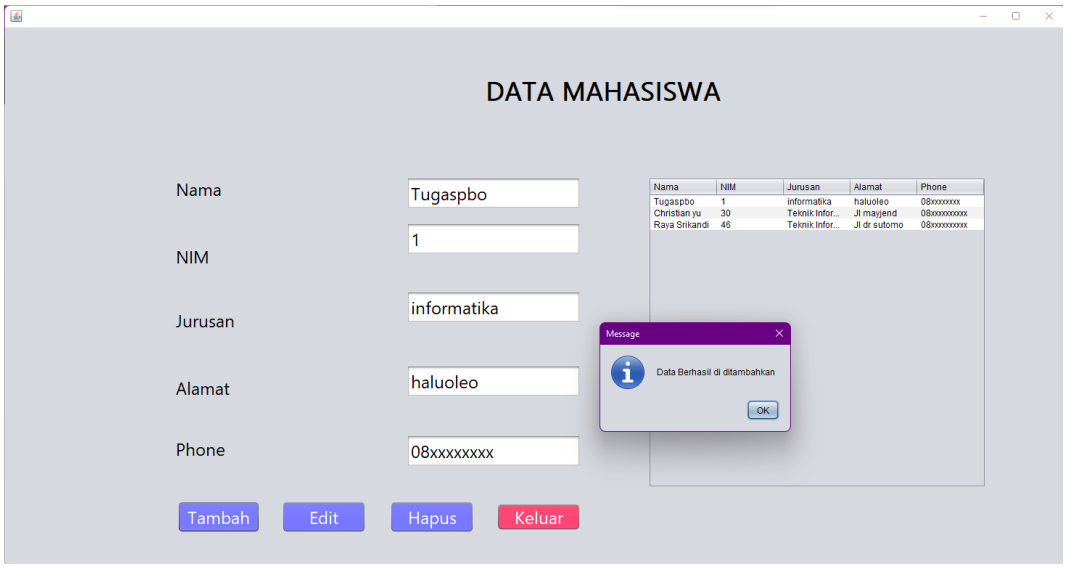
```
private void tableMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
    int i = table.getSelectedRow();
    DefaultTableModel model=(DefaultTableModel)table.getModel();
    if (i== -1){
        return;
    }
    nama.setText (model.getValueAt(i, 0).toString());
    int NIM = Integer.parseInt(model.getValueAt(i, 1).toString());
    jurusan.setText (model.getValueAt(i, 2).toString());
    alamat.setText (model.getValueAt(i, 3).toString());
    phone.setText (model.getValueAt(i, 4).toString());
}
```

C. Hasil Output Program

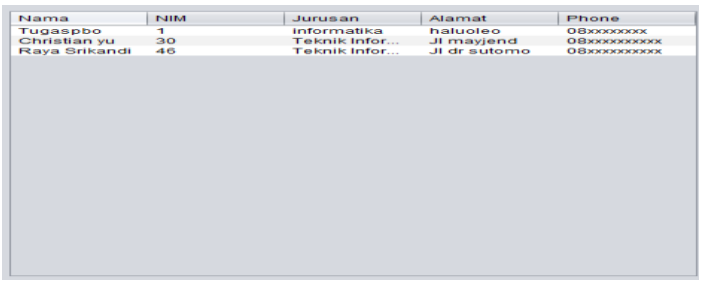
a. Tampilan awal



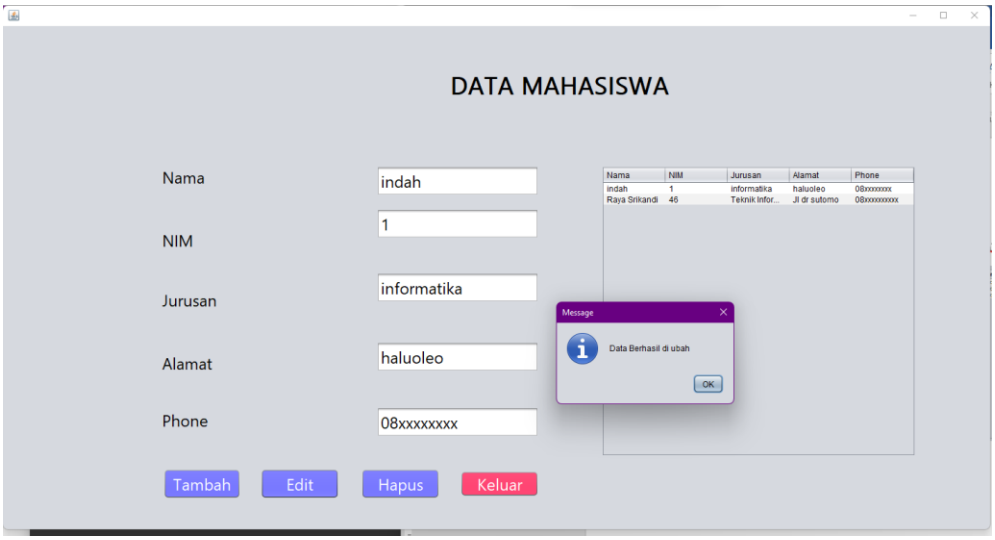
b. Proses Tambah data



c. Proses penyimpanan berhasil dan hasil ditampilkan dalam table



d. Proses Mengubah data



e. Proses Menghapus data