

# **LAPORAN TUGAS PROYEK**

**“Aplikasi Menghitung Nilai Akhir Mahasiswa”**



**Nama : Yeyen Arisanti**

**Stambuk : 13020180033**

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS MUSLIM INDONESIA**

**MAKASSAR**

**2019**

## **BAB 1**

### **PENDAHULUAN**

Suatu pengolahan data mahasiswa dalam sebuah usaha bias mendapatkan kendala, seperti yang dapat dihadapi seperti yang ada di program tersebut. Mahasiswa yang menggunakan system konvensional karena adanya suatu aplikasi untuk membantu perusahaan dalam bidang studi Teknik Informatika, pengeluaran nilai membutuhkan waktu yang cukup lama.

Pada system yang dapat mempercepat dan mempermudah proses perhitungan nilai mahasiswa dapat membuat Aplikasi Menghitung Nilai Akhir Mahasiswa pada bidang studi Teknik Informatika.

## BAB 2

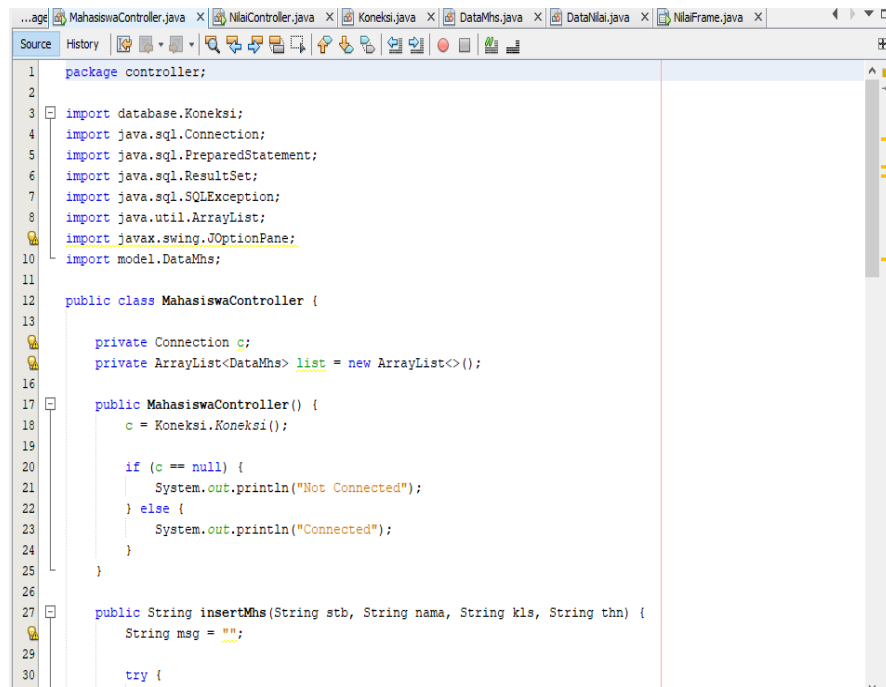
### ANALISIS DAN DESAIN

1. (Prosedur Kerja / Flowchart)
2. Konsep Pemrograman Berorientasi Object

#### a. Polimorphism

Polymorphism merupakan kemampuan suatu method untuk bekerja dengan lebih dari satu tipe argumen. Pada bahasa lain (khususnya C++), konsep ini sering disebut dengan method overloading. Pada dasarnya, Python tidak menangani hal ini secara khusus. Hal ini disebabkan karena Python merupakan suatu bahasa pemrograman yang bersifat dynamic typing yaitu tidak memerlukan deklarasi tipe.

Contohnya mengoverride method update() dengan menambahkan beberapa teks tambahan. Method update() pada class GajiServiceImp.java perilaku yang berbeda dengan class GajiServiceImp



```
1 package controller;
2
3 import database.Koneksi;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.util.ArrayList;
9 import javax.swing.JOptionPane;
10 import model.DataMhs;
11
12 public class MahasiswaController {
13
14     private Connection c;
15     private ArrayList<DataMhs> list = new ArrayList<>();
16
17     public MahasiswaController() {
18         c = Koneksi.Koneksi();
19
20         if (c == null) {
21             System.out.println("Not Connected");
22         } else {
23             System.out.println("Connected");
24         }
25     }
26
27     public String insertMhs(String stb, String nama, String kls, String thn) {
28         String msg = "";
29
30         try {
```

```

try {
    // String sql ="INSERT INTO tb_data (periode, data_aktual) VALUES ('"+periode+"','"+data+"')";
    String sql = "INSERT INTO tb_mhs VALUES (?, ?, ?, ?)";
    // String sql ="INSERT INTO tb_data VALUES (null, '"+periode+"', '"+data+"')";
    PreparedStatement ps = c.prepareStatement(sql);
    ps.setString(1, stb);
    ps.setString(2, nama);
    ps.setString(3, kls);
    ps.setString(4, thn);
    // ps.setInt(2, data);
    int rowsInserted = ps.executeUpdate();
    if (rowsInserted > 0) { //kondisi mengecek data masuk/tidak
        msg = "Insert Success";
    } else {
        msg = "Insert Failed";
    }
}

} catch (SQLException e) {
    msg = "Error " + e.getMessage();
}

return msg;
}

```

```

public ArrayList<DataMhs> getDataMhs() {

    try {

        String sql = "SELECT *FROM tb_mhs";
        PreparedStatement ps = c.prepareStatement(sql);
        ResultSet rs = ps.executeQuery();

        while (rs.next()) {
            list.add(new DataMhs(rs.getString("stambuk"), rs.getString("nama"),
                                rs.getString("kelas"), rs.getString("angkatan")));
        }

    } catch (SQLException e) {
        System.out.println("salah"+e);
    }

    return list;
}

public static void main(String[] args) {
    MahasiswaController m = new MahasiswaController();
}

```

## **b. Inheritance (Pewarisan)**

Inheritance atau Pewarisan/Penurunan adalah konsep pemrograman dimana sebuah class dapat ‘menurunkan’ property dan method yang dimilikinya kepada class lain. Konsep inheritance digunakan untuk memanfaatkan fitur ‘code reuse’ untuk menghindari duplikasi kode program.

Konsep inheritance membuat sebuah struktur atau ‘hierarchy’ class dalam kode program. Class yang akan ‘diturunkan’ bisa disebut sebagai class induk (parent class), super class, atau base class. Sedangkan class yang ‘menerima penurunan’ bisa disebut sebagai class anak (child class), sub class, derived class atau heir class.

## **c. Enkapsulasi / pengkapsulan (Encapsulation)**

Enkapsulasi adalah pembungkus, maksud pembungkus disini adalah untuk menjaga suatu proses program agar tidak dapat diakses secara sembarangan atau diintervensi oleh program lain. Konsep enkapsulasi sangat penting dilakukan untuk menjaga kebutuhan program agar dapat diakses sewaktu-waktu, sekaligus menjaga program tersebut.

Begitulah konsep kerja dari enkapsulasi, dia akan melindungi sebuah program dari akses ataupun intervensi dari program lain yang mempengaruhinya. Hal ini sangat menjaga keutuhan program yang telah dibuat dengan konsep dan rencana yang sudah ditentukan dari awal.

Enkapsulasi menciptakan abstraksi untuk desain kelas. Jika Anda ingin melindungi beberapa anggota dari kelas dasar maka timbullah situasi enkapsulasi. Dalam konsep ini ada tiga kata kunci yang digunakan. Konsep hak akses ini biasa disebut Access Modifier.

## BAB 3

### HASIL IMPLEMENTASI PROGRAM

#### 1. Luaran / Output Program

The screenshot shows a web application window titled "Design Preview [NilaiFrame]". The interface is divided into two main sections. On the left is the "Input Data" section, which includes a dropdown menu labeled "-- Silahkan Pilih Mahasiswa --", five text input fields for "Nilai UTS", "Nilai UAS", "Nilai Project", "Nilai Quiz", and "Nilai Kehadiran", and three buttons labeled "INSERT", "UPDATE", and "DELETE". On the right is a table with the following columns: "Stambuk", "Nama", "UTS", "UAS", "Project", "QUIZ", "Kehadiran", "Nilai Akhir", and "Grade". The table is currently empty.

Stambuk	Nama	UTS	UAS	Project	QUIZ	Kehadiran	Nilai Akhir	Grade
---------	------	-----	-----	---------	------	-----------	-------------	-------

Input Data

Charlotte PSBB Linlin

Nilai UTS

80

Nilai UAS

86

Nilai Project

78

Nilai Quiz

60

Nilai Kehadiran

70

INSERT

Stambuk	Nama Mhs	UTS	UAS	Project	Quiz	Kehadiran	Nilai Akhir	Grade
130201800...	Covid Beck...	90	90	80	70	60	79.5	A

Input Data

Charlotte PSBB Linlin

Nilai UTS

80

Nilai UAS

86

Nilai Project

78

Nilai Quiz

60

Nilai Kehadiran

70

INSERT

Stambuk	Nama Mhs	UTS	UAS	Project	Quiz	Kehadiran	Nilai Akhir	Grade
130201800...	Covid Beck...	90	90	80	70	60	79.5	A

Message

i

Insert Berhasil

OK

Input Data

Charlotte PSBB Linlin

Nilai UTS

70

Nilai UAS

86

Nilai Project

78

Nilai Quiz

60

Nilai Kehadiran

70

INSERT

UPDATE

DELETE

Stambuk	Nama Mhs	UTS	UAS	Project	Quiz	Kehadiran	Nilai Akhir	Grade
130201800...	Covid Beck...	90	90	80	70	60	79.5	A
130201800...	Charlotte P...	86	86	78	60	70	76.800000...	A

Message

i

Update Berhasil

OK

Input Data

Charlotte PSBB Linlin

Nilai UTS

86

Nilai UAS

86

Nilai Project

78

Nilai Quiz

60

Nilai Kehadiran

70

INSERT

UPDATE

DELETE

Stambuk	Nama Mhs	UTS	UAS	Project	Quiz	Kehadiran	Nilai Akhir	Grade
130201800...	Covid Beck...	90	90	80	70	60	79.5	A
130201800...	Charlotte P...	86	86	78	60	70	72.800000...	B

Message

i

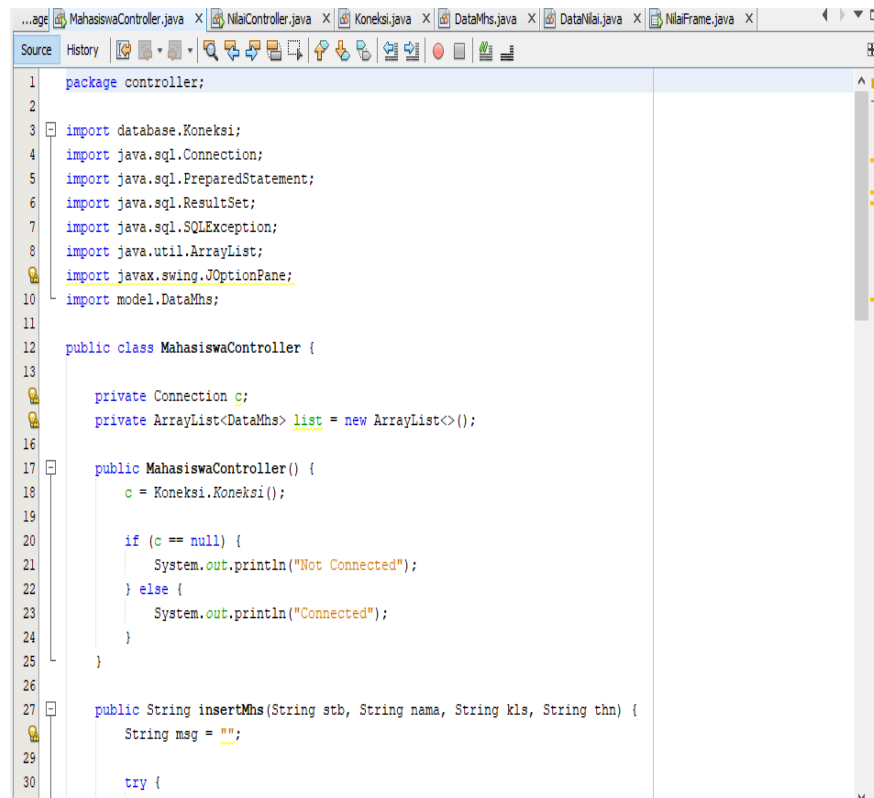
Delete Berhasil

OK



## 2. Penggalan Source Code Program Aplikasi

### a. Class Mahasiswa Controller



The screenshot shows an IDE window with the source code for the `MahasiswaController` class. The code is as follows:

```
1 package controller;
2
3 import database.Koneksi;
4 import java.sql.Connection;
5 import java.sql.PreparedStatement;
6 import java.sql.ResultSet;
7 import java.sql.SQLException;
8 import java.util.ArrayList;
9 import javax.swing.JOptionPane;
10 import model.DataMhs;
11
12 public class MahasiswaController {
13
14     private Connection c;
15     private ArrayList<DataMhs> list = new ArrayList<>();
16
17     public MahasiswaController() {
18         c = Koneksi.Koneksi();
19
20         if (c == null) {
21             System.out.println("Not Connected");
22         } else {
23             System.out.println("Connected");
24         }
25     }
26
27     public String insertMhs(String stb, String nama, String kls, String thn) {
28         String msg = "";
29
30         try {
```

```

try {
    // String sql ="INSERT INTO tb_data (periode, data_aktual) VALUES ('"+periode+"','"+data+"')";
    String sql = "INSERT INTO tb_mhs VALUES (?, ?, ?, ?)";
    // String sql ="INSERT INTO tb_data VALUES (null,'"+periode+"','"+data+"')";
    PreparedStatement ps = c.prepareStatement(sql);
    ps.setString(1, stb);
    ps.setString(2, nama);
    ps.setString(3, kls);
    ps.setString(4, thn);
    // ps.setInt(2, data);
    int rowsInserted = ps.executeUpdate();
    if (rowsInserted > 0) { //kondisi mengecek data masuk/tidak
        msg = "Insert Success";
    } else {
        msg = "Insert Failed";
    }

} catch (SQLException e) {
    msg = "Error " + e.getMessage();
}

return msg;
}

```

```

public ArrayList<DataMhs> getDataMhs() {

    try {

        String sql = "SELECT *FROM tb_mhs";
        PreparedStatement ps = c.prepareStatement(sql);
        ResultSet rs = ps.executeQuery();

        while (rs.next()) {
            list.add(new DataMhs(rs.getString("stambuk"), rs.getString("nama"),
                                rs.getString("kelas"), rs.getString("angkatan")));
        }

    } catch (SQLException e) {
        System.out.println("salah"+e);
    }

    return list;
}

public static void main(String[] args) {
    MahasiswaController m = new MahasiswaController();
}

```

## b. Class Nilai Controller

```
package controller;

import database.Koneksi;
import java.sql.Connection;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.util.ArrayList;
import javax.swing.JOptionPane;
import model.DataMhs;
import model.DataNilai;

public class NilaiController {

    public NilaiController() {
        c = Koneksi.Koneksi();

        if (c == null) {
            System.out.println("Not Connected");
        } else {
            System.out.println("Connected");
        }
    }

    private Connection c;
    private ArrayList<DataNilai> list = new ArrayList<>();

    public ArrayList<DataNilai> getDataNilai() {

        try {

            String sql = "SELECT tb_mhs.stambuk, tb_mhs.nama, nilai.id, nilai.uts, nilai.uas, "
                + "nilai.project, nilai.quiz, nilai.kehadiran FROM tb_mhs, "
                + "nilai WHERE nilai.stambuk = tb_mhs.stambuk";
            PreparedStatement ps = c.prepareStatement(sql);
            ResultSet rs = ps.executeQuery();

            while (rs.next()) {
                list.add(new DataNilai(rs.getString("stambuk"), rs.getString("nama"),
                    rs.getInt("id"), rs.getInt("uas"), rs.getInt("uts"), rs.getInt("project"),
                    rs.getInt("quiz"), rs.getInt("kehadiran")));
            }

        } catch (SQLException e) {
            System.out.println("salah" + e);
        }

        return list;
    }

    public String insertNilai(String stb, String uts, String uas, String project,
        String quiz, String kehadiran) {
        String msg = "";

        try {
            String sql = "INSERT INTO nilai VALUES (null, ?, ?, ?, ?, ?, ?)";

            PreparedStatement ps = c.prepareStatement(sql);
            ps.setString(1, stb);
            ps.setString(2, uts);
        }
    }
}
```

```

        ps.setString(3, uas);
        ps.setString(4, project);
        ps.setString(5, quiz);
        ps.setString(6, kehadiran);

        int rowsInserted = ps.executeUpdate();
        if (rowsInserted > 0) { //kondisi mengecek data masuk/tidak
            msg = "success";
        } else {
            msg = "failed";
        }

    } catch (SQLException e) {
        msg = "Error " + e.getMessage();
    }

    return msg;
}

public String updateNilai(String id, String uts, String uas, String project,
    String quiz, String kehadiran) {
    String msg = "";

    try {
        String sql = "UPDATE nilai SET uts=?, uas=?, project=?, quiz=?, kehadiran=? where id=?";

        PreparedStatement ps = c.prepareStatement(sql);
        ps.setString(1, uts);
        ps.setString(2, uas);
        ps.setString(3, project);

```

```

        ps.setString(3, project);
        ps.setString(4, quiz);
        ps.setString(5, kehadiran);
        ps.setString(6, id);

        int rowsInserted = ps.executeUpdate();
        if (rowsInserted > 0) { //kondisi mengecek data masuk/tidak
            msg = "success";
        } else {
            msg = "failed";
        }

    } catch (SQLException e) {
        msg = "Error " + e.getMessage();
    }

    return msg;
}

public String deleteNilai(String id) {
    String msg = "";
    try {
        String sql = "DELETE FROM nilai where id=?";
        PreparedStatement ps = c.prepareStatement(sql);
        ps.setString(1, id);

        int rowsInserted = ps.executeUpdate();
        if (rowsInserted > 0) {
            msg = "success";
        } else {

```

```

        int rowsInserted = ps.executeUpdate();
        if (rowsInserted > 0) {
            msg = "success";
        } else {
            msg = "failed";
        }

    } catch (SQLException ex) {
        System.out.println(ex);
    }

    return msg;
}

public static void main(String[] args) {
    NilaiController n = new NilaiController();
}
}

```

### c. Class Koneksi

```

package database;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.logging.Level;
import java.util.logging.Logger;

public class Koneksi {

    private static Connection con;

    public static Connection Koneksi() {
        try {
            Class.forName("com.mysql.jdbc.Driver");
            String url = "jdbc:mysql://localhost/mahasiswa?user=root&password=";
            con = DriverManager.getConnection(url);
            con.createStatement();
            System.out.println("database connected");
        } catch (SQLException | ClassNotFoundException ex) {
            Logger.getLogger(Koneksi.class.getName()).log(Level.SEVERE, null, ex);
            System.out.println("database disconnected");
        }

        return con;
    }
}

```

#### d. Class Data Mahasiswa

```
package model;

public class DataMhs {
    private String stb, nama, kls, thn;

    public DataMhs(String stb, String nama, String kls, String thn) {
        this.stb = stb;
        this.nama = nama;
        this.kls = kls;
        this.thn = thn;
    }

    public String getStb() {
        return stb;
    }

    public String getNama() {
        return nama;
    }

    public String getKls() {
        return kls;
    }

    public String getThn() {
        return thn;
    }
}
```

#### e. Class Data Nilai

```
package model;

public class DataNilai {
    private String stb, nama;
    private int id, uts, uas, project, quiz, kehadiran;

    public DataNilai(String stb, String nama, int id, int uts, int uas,
        int project, int quiz, int kehadiran) {
        this.stb = stb;
        this.nama = nama;
        this.id = id;
        this.uts = uts;
        this.uas = uas;
        this.project = project;
        this.quiz = quiz;
        this.kehadiran = kehadiran;
    }

    public int getId() {
        return id;
    }

    public String getStb() {
        return stb;
    }

    public String getNama() {
        return nama;
    }

    public int getUts() {
        return uts;
    }
}
```

```

public int getUas() {
    return uas;
}

public int getProject() {
    return project;
}

public int getQuiz() {
    return quiz;
}

public int getKehadiran() {
    return kehadiran;
}

```

```

}

```

#### f. Class Nilai Frage

```

package view;

import controller.MahasiswaController;
import controller.NilaiController;
import java.util.ArrayList;
import javax.swing.ComboBoxEditor;
import javax.swing.ComboBoxModel;
import javax.swing.DefaultComboBoxModel;
import javax.swing.JOptionPane;
import javax.swing.table.DefaultTableModel;
import model.DataMhs;
import model.DataNilai;

public class NilaiFrame extends javax.swing.JFrame {

    private NilaiController n = new NilaiController();
    private MahasiswaController mhs = new MahasiswaController();

    private String resStb, resNama, resId;
    private String resUts, resUas, resProject, resQuiz, resKehadiran;

    ArrayList<DataMhs> listMhs = mhs.getDataMhs();
    ArrayList<DataNilai> list = new ArrayList<>();

    DefaultTableModel dtmNilai = new DefaultTableModel();

```

```

public NilaiFrame() {
    initComponents();

    initTableNilai();
    initMhs();
    btnUpdate.setVisible(false);
    btnDelete.setVisible(false);
}

private void initTableNilai() {

    list.clear();
    list = n.getDataNilai();

    dtmNilai = new DefaultTableModel();
    dtmNilai.addColumn("Stambuk");
    dtmNilai.addColumn("Nama Mhs");
    dtmNilai.addColumn("UTS");
    dtmNilai.addColumn("UAS");
    dtmNilai.addColumn("Project");
    dtmNilai.addColumn("Quiz");
    dtmNilai.addColumn("Kehadiran");
    dtmNilai.addColumn("Nilai Akhir");
    dtmNilai.addColumn("Grade");

    System.out.println("list Size " + list.size());
    for (DataNilai u : list) {

        System.out.println("nama User" + u.getNama());
    }
    for (DataNilai u : list) {

        System.out.println("nama User" + u.getNama());

        double uas = Double.valueOf(u.getUas()) * 0.25;
        double uts = Double.valueOf(u.getUts()) * 0.2;
        double project = Double.valueOf(u.getProject()) * 0.2;
        double quiz = Double.valueOf(u.getQuiz()) * 0.2;
        double hadir = Double.valueOf(u.getKehadiran()) * 0.15;

        double nilaiAkhir = uas+uts+project+quiz+hadir;
        String grade = "";

        if(nilaiAkhir > 75){
            grade = "A";
        }else if(nilaiAkhir <= 75 && nilaiAkhir > 60){
            grade = "B";
        } else if(nilaiAkhir <= 60 && nilaiAkhir > 50){
            grade = "C";
        } else{
            grade = "D";
        }

        dtmNilai.addRow(new Object[]{u.getStb(), u.getNama(), u.getUts(),
            u.getUts(), u.getProject(), u.getQuiz(), u.getKehadiran(), nilaiAkhir, grade});
    }

    tableNilai.setModel(dtmNilai);
}

```



```

3 private void initMhs() {
    String[] model = new String[listMhs.size()];

    for (int i = 0; i < listMhs.size(); i++) {
        model[i] = listMhs.get(i).getNama();
        resStb = listMhs.get(0).getStb();
    }
    cbMhs.setModel(new DefaultComboBoxModel(model));

- }

3 private void clearField() {
    tfKehadiran.setText("");
    tfProject.setText("");
    tfQuiz.setText("");
    tfUas.setText("");
    tfUts.setText("");

- }

```

```

3 private void cbMhsActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    String nama = cbMhs.getSelectedItem().toString();
    for (DataMhs m : listMhs) {
        if (m.getNama().equals(nama)) {
            resStb = m.getStb();
        }
    }

- }

3 private void btnInsertActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    resUts = tfUts.getText();
    resUas = tfUas.getText();
    resProject = tfProject.getText();
    resQuiz = tfQuiz.getText();
    resKehadiran = tfKehadiran.getText();

    String insert = n.insertNilai(resStb, resUts, resUas, resProject, resQuiz, resKehadiran);
    if (insert.equals("success")) {
        JOptionPane.showMessageDialog(null, "Insert Berhasil");
    } else {
        JOptionPane.showMessageDialog(null, "Insert Gagal");
    }

- }

```

```

        initTableNilai();
        clearField();
    }

    private void tableNilaiMouseClicked(java.awt.event.MouseEvent evt) {
        // TODO add your handling code here:
        btnDelete.setVisible(true);
        btnUpdate.setVisible(true);

        int row = tableNilai.getSelectedRow();
        int column = tableNilai.getSelectedColumn();

        final String valueInCell = (String) tableNilai.getValueAt(row, 0);
        tfUts.setText(valueInCell);

        for (DataNilai d : list) {
            if (d.getStb().equals(valueInCell)) {
                tfUts.setText((String) String.valueOf(tableNilai.getValueAt(row, 2)));
                tfUas.setText((String) String.valueOf(tableNilai.getValueAt(row, 3)));
                tfProject.setText((String) String.valueOf(tableNilai.getValueAt(row, 4)));
                tfQuiz.setText((String) String.valueOf(tableNilai.getValueAt(row, 5)));
                tfKehadiran.setText((String) String.valueOf(tableNilai.getValueAt(row, 6)));
                cbMhs.getModel().setSelectedItem(tableNilai.getValueAt(row, 1));
                resStb = String.valueOf(tableNilai.getValueAt(row, 0));
                resId = String.valueOf(d.getId());
            }
        }
    }
}

```

```

    private void btnDeleteActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        String delete = n.deleteNilai(resId);
        if (delete.equals("success")) {
            JOptionPane.showMessageDialog(null, "Delete Berhasil");
        } else {
            JOptionPane.showMessageDialog(null, "Delete Gagal");
        }
        initTableNilai();
        clearField();
    }

    private void btnUpdateActionPerformed(java.awt.event.ActionEvent evt) {
        // TODO add your handling code here:
        resUts = tfUts.getText();
        resUas = tfUas.getText();
        resProject = tfProject.getText();
        resQuiz = tfQuiz.getText();
        resKehadiran = tfKehadiran.getText();

        String insert = n.updateNilai(resId, resUts, resUas, resProject, resQuiz, resKehadiran);
        if (insert.equals("success")) {
            JOptionPane.showMessageDialog(null, "Update Berhasil");
        } else {
            JOptionPane.showMessageDialog(null, "Update Gagal");
        }
    }
}

```

```

initTableNilai();
clearField();
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    Look and feel setting code (optional)

    /* Create and display the form */
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new NilaiFrame().setVisible(true);
        }
    });
}

```

## g. Database Mahasiswa

### 1. Tabel Nilai

Server: 127.0.0.1 x Database: mahasiswa x Table: nilai

[Browse](#)
[Structure](#)
[SQL](#)
[Search](#)
[Insert](#)
[Export](#)
[Import](#)
[Privileges](#)
[Operations](#)
[Tracking](#)
[Triggers](#)

[Table structure](#)
[Relation view](#)

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/> 1	id	int(11)			No	None		AUTO_INCREMENT	<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
<input type="checkbox"/> 2	stambuk	varchar(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
<input type="checkbox"/> 3	uts	varchar(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
<input type="checkbox"/> 4	uas	varchar(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
<input type="checkbox"/> 5	project	varchar(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
<input type="checkbox"/> 6	quiz	varchar(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>
<input type="checkbox"/> 7	kehadiran	varchar(11)	utf8mb4_general_ci		No	None			<a href="#">Change</a> <a href="#">Drop</a> <a href="#">More</a>

☐ Check all
 With selected: [Browse](#) [Change](#) [Drop](#) [Primary](#) [Unique](#) [Index](#) [Fulltext](#) [Add to central columns](#)

[Remove from central columns](#)

[Print](#)
[Propose table structure](#)
[Track table](#)
[Move columns](#)
[Normalize](#)

[Add](#) 1 column(s) after kehadiran [Go](#)

## 2. Tabel Mahasiswa

Server: 127.0.0.1 » Database: mahasiswa » Table: tb\_mhs

Browse Structure SQL Search Insert Export Import Privileges Operations

Table structure Relation view

#	Name	Type	Collation	Attributes	Null	Default	Comments	Extra	Action
<input type="checkbox"/>	1	stambuk	varchar(11)	utf8mb4_general_ci	No	None			Change  Drop  More
<input type="checkbox"/>	2	nama	varchar(255)	utf8mb4_general_ci	No	None			Change  Drop  More
<input type="checkbox"/>	3	kelas	varchar(2)	utf8mb4_general_ci	No	None			Change  Drop  More
<input type="checkbox"/>	4	angkatan	varchar(4)	utf8mb4_general_ci	No	None			Change  Drop  More

☐ Check all With selected: Browse Change Drop Primary Unique Index Fulltext

Remove from central columns

---

Print Propose table structure Track table Move columns Normalize

Add  column(s)  Go

## **BAB 4 PENUTUP**

### **1. Kesimpulan**

Dengan adanya aplikasi tersebut bisa mengolah nilai mahasiswa dengan mudah dan tidak membutuhkan waktu yang cukup lama.

### **2. Saran**

Sistem aplikasi ini masih bisa dikembangkan lagi agar lebih detail dan saya harap aplikasi ini bisa di aplikasikan.