**Laporan Tugas Proyek**

**Menghitung Nilai Akhir Mahasiswa Program Studi Teknik Informatika**

****

Nisrina Dwi Ambarwati

13020180097

B2

**PROGRAM STUDI TEKNIK INFORMATIKA**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS MUSLIM INDONESIA**

**MAKASSAR**

**2020**

**KATA PENGANTAR**

Puji syukur kami ucapkan atas kehadirat Allah SWT, karena dengan rahmat dan karunia-Nya, kami masih diberi kesempatan untuk menyelesaikan laporan ini tentang konsep *menghitung nilai akhir mahasiswa* sebagai salah satu tugas proyek pemrograman beriontasi objek.

Tidak lupa kami ucapkan terima kasih kepada dosen serta berbagai pihak yang telah memberikan dukungan serta memberikan petunjuk dalam menyelesaikan makalah ini. Kami menyadari bahwa dalam penulisan laporan ini masih banyak kekurangan, oleh sebab itu kami sangat mengharapkan kritik dan saran yang membangun.

Semoga dengan selesainya laporan ini dapat bermanfaat bagi pembaca dan teman-teman.

**Makassar, 20 Juni 2020**

Nisrina Dwi Ambarwati

**DAFTAR ISI**

**Kata Pengantar ii**

**Daftar Isi iii**

**BAB I. PENDAHULUAN 1**

* 1. **Latar Belakang 1**
  2. **Rumusan Masalah 1**
  3. **Batasan Masalah 2**
  4. **Tujuan 2**
  5. **Manfaat 2**

**BAB II. LANDASAN TEORI 3**

**2.1 Pemrograman Beriontasi Objek 3**

**2.2 Percabangan 4**

**2.2.1 Bentuk-bentuk percabangan 4**

**2.3 Perulangan 6**

* 1. **kelas (class) 8**

**2.4.1 Membuat class 8**

* 1. **Objek (Object) 9**

**2.5.1 Membuat object 9**

* 1. **Atribut 9**
     1. **Membuat atribut 9**
     2. **Atribut dapat diakses lewat class 9**
  2. **Konstanta 10**
  3. **Prosedure 10**
  4. **Fungsi 11**

**2.8.1 Bentuk – bentuk fungsi 11**

* 1. **This 13**
  2. **Overloading 13**
  3. **Konstruktor 13**
  4. **Paket 14**

**2.13.1 Macam – macam paket 14**

* 1. **Visibility 15**
  2. **Enkapsulasi 15**

**BAB III. PERANCANGAN SISTEM 17**

* 1. **Sistem Usulan 17**

**BAB IV. IMPLEMENTASI & PEMBAHASAN 18**

**4.1 Implementasi 18**

**BAB V. PENUTUP 24**

**5.1 Kesimpulan 24**

**5.2 Saran 24**

**Daftar Pustaka**

**BAB I**

**PENDAHULUAN**

* 1. **Latar Belakang**

Seiring dengan kemajuan dan perkembangan zaman di era globalisasi ini, manusia dituntut mengikuti perkembangan ilmu pengetahuan dan teknologi. Semakin banyaknya *software*, dan semakin meingkatnya kecanggihan *hardware*, maka perangkat komputer sebagai alat bantu menjadi semakin berperan dalam menyelesaikan suatu pekerjaan.

Hal ini juga termasuk dalam proses penilaian akhir mahasiswa. Untuk memudahkan dosen dalam penghitung nilai akhir mahasiswa pada tiap semester akan merepotkan jika perhitunganya masi manual yang harus menghitung tiap kelas yang di ajar.

* 1. **Rumusan Masalah**

Adapun rumusan masalah pada laporan ini, yaitu :

1. Bagaimana cara ketika memasukan data?
2. Bagaimana cara ketika perhitungan ?
3. Bagaimana tampilan Output yang sudah terhitung ?
4. Bagaimana data yang di masukan itu masuk ke dalam tabel database?
   1. **Batasan Masalah**

Adapun batasan-batasan masalah dalam pembuatan program ini, yaitu:

1. Program ini menggunakan aplikasi pemrograman java.
2. Pada bagian tombol simpan tidak tersimpan ke dalam tabel hanya bias tersimpan dalam tabal database.
   1. **Tujuan**

Adapun tujuan pembuatan laporan ini adalah untuk melengkapi nilai tugas yang telah di berikan sebagai media untuk menambah ilmu pengetahuan.

* 1. **Manfaat**

Dengan adanya laporan ini diharapkan mahasiswa/i dapat lebih memahami dan mengerti tentang materi java GUI, dan lebih memudahkan mahasiswa untuk belanja mata kuliah.

**BAB II**

**LANDASAN TEORI**

* 1. **Pemrograman Beriontasi Objek**

Pemrograman Beroroentasi Objek merupakan sebuah paradigma pemrograman yang berorientasikan kepada objek. Semua atribut, prosedur dan fungsi dibungkus dalam objek. Interaksi tiap data, prosedur dan fungsi dilakukan melalui objek.

Java merupakan strongly typed language, maksudnya setiap membuat variabel, harus dideklarasikan tipe datanya. Ada beberapa tipe data yang terdapat pada java:

* + - * Tipe data angka
      * Tipe data decimal
      * Tipe data karakter
      * Tipe data bolean
      * Tipe data objek

Variabel merupakan sebuah tempat untuk menyimpan data. Di Java setiap pembuatan variabel harus ditentukan tipe datanya. Cara pembuatan variabel adalah dengan menggunakan tipe data dan diikuti nama variabel.

* 1. **Percabangan**

Percabangan adalah suatu pilihan atau opsi dimana terdapat kondisi tertentu yang harus dipenuhi oleh proogram untuk menjalankan suatu perintah, jika kondisi atau syarat itu terpenuhi maka program akan menjalankan perintahnya tetapi jika tidak maka program tidak akan menjalanakan perintahnya atau melewatinya dan melihat kondisi lainnya untuk dijlankan atau berhenti sama sekali.

2.2.1 Bentuk-bentuk percabangan

1. Statement if

Pernyataan if akan menentukan sebuah pernyataan yang akan dieksekusi jika dan hanya jika persyaratan bernilai benar (true).

* + - 1. Bentuk umum statement if

Deklarasi:

if(kondisi){

// isi perintah jika kondisi benar

}

1. Statement if-else

Pernyataan if-else digunakan apabila kita ingin mengeksekusi beberapa pernyataan dengan kondisi true dan pernyataan yang lain dengan kondisi false.

1. Bentuk umum statement if-else

Deklarasi:

if(kondisi){

// isi perintah jika kondisi benar

}else{

// isi perintah jika kondisi salah

}

1. Statement if-else-if

Pernyataan pada bagian kondisi else dari blok if-else dapat menjadi struktur if-else yang lain. Kondisi struktur seperti ini mengijinkan kita untuk membuat seleksi persyaratan yang lebih kompleks dibandingkan dengan pernyataan if-else sebelumnya.

1. Bentuk umum statement if-else-if

Deklarasi:

if(kondisi1){

// isi perintah jika kondisi 1 benar

}else if(kondisi2){

// isi perintah jika kondisi 2 salah

} else if(....){

// ......

}else{

// isi perintah jika semua kondisi salah

}

1. Statement switch case

Cara lain untuk membuat percabangan di dalam bahasa pemrograman Java adalah dengan menggunakan kata kunci switch. Switch mengkonstruksikan cabang untuk beberapa kondisi dari nilai.

1. Bentuk umum statement switch case

Deklarasi:

switch(variabel){

case 1 :

// isi perintah jika variabel = 1

break;

case 2:

case 3:

// isi perintah jika variabel = 2 atau 3

break;

default :

// isi perintah jika variabel sesuai

break;

}

* 1. **Perulangan**

Looping atau dalam bahasa indonesia di sebut dengan perulangan merupakan fungsi pada pemrograman untuk mempermudah penulisan kode program yang tanpa harus menulis program tersebut berulang-ulang.

2.3.1 Bentuk-bentuk perulangan

A. Statement for

perulangan for digunakan ketika angka dari perulangan sebelum kontrol memasuki loop. Proses pengulangan akan terus dilakukan selama kondisi loop bernilai true. Dengan kata lain, proses pengulangan hanya akan dihentikan apabila kondisi telah bernilai false atau sudah tidak terpenuhilagi.  
1. Bentuk umum statement for

Deklarasi:

For (inisialisasi; kondisi; increment) {

// lakukan selama kondisi benar

}

B. Statement while

perulangan while untuk kondisi perulangan dimana banyaknya perulangan tidakdapat dipastikan pada saat penulisan program ([sumber](http://www.duniailkom.com/tutorial-belajar-php-pengertian-dan-cara-penulisan-perulangan-while-dalam-php/))

1. Bentuk umum statement while

Deklarasi :

while( kondisi ){

// lakukan selama kondisi benar

}

C. Statement do-while

pada perulangan*while*dan*do while*hampir sama hanya saja yang membedakannya adalah tempat peletakan kondisi perualngan. melakukan perulangan kalau kondisi (syarat) terpenuhi. Sedangkan *do/while* melakukan perulangan dulu, kemudian memeriksa kondisinya atau sayaratnya. Kalau kondisi terpenuhi, maka *do/while* akan melanjutkan perulangan. Sebaliknya, dia akan berhenti *(break)*.

1. Bentuk umum statement do-while

Deklarasi :

do {

// lakukan selama kondisi benar

} while ( kondisi );

* 1. **kelas (class)**

Kelas merupakan kerangka kode yang berisikan struktur atribut, struktur prosedur dan struktur fungsi yang digunakan untuk membuat objek. Sebuah kelas dapat menghasilkan lebih dari satu objek.

2.4.1 Membuat class

Nama kelas harus sama dengan nama File. Nama kelas tidak boleh mengandung huruf unik (@, #, $, %, &, dll) dan huruf whitespace (spasi, enter, tab, dll)

Bentuk umum.

Deklarasi :

class NamaKelas {

// isi kelas

}

* 1. **Objek (Object)**

Objek merupakan hasil instansiasi dari kelas. Untuk membuat objek, di Java dapat menggunakan perintah new. Atribut, metode dan fungsi yang dimiliki oleh objek, hanya yang dideklarasikan dalam kelas.

* + 1. Membuat objek

Objek merupakan variabel yang memiliki tipe data Kelas Kriteria pembuatan objek sama seperti pembuatan variabel.

Bentuk umum.

Deklarasi :

NamaKelas object = new NamaKelas();

* 1. **Atribut**

Atribut merupakan data atau sifat yang dimiliki oleh sebuah kelas. Atribut dibuat layaknya sebuah variabel.

2.6.1 Membuat atribut

Bentuk umum.

Deklarasi:

class NamaKelas {

TipeData namaAtribut;

}

2.6.2 Atribut dapat diakses lewat kelas.

Untuk mengakses atribut, dapat menggunakan tanda . (titik)

Bentuk umum.

Deklarasi:

NamaKelas objek = new NamaKelas();

objek.namaAtribut = ...;

**2.7 Konstanta**

Sebuah atribut dapat dijadikan sebagai konstanta. Caranya sama dengan membuat variabel konstanta, yaitu dengan menambahkan perintah final.

Bentuk umum.

Deklarasi :

class NamaKelas {

TipeData namaAtribut;

final TipeData namaKonstanta = ...;

}

* 1. **Prosedure**

Prosedur merupakan sebuah kode program yang dapat digunakan untuk menjalankan instruksi program dalam sebuah kelas. Sebuah kelas dapat memiliki lebih dari satu prosedur. Nama prosedur tidak boleh sama dengan prosedur lain.

Bentuk umum.

Deklarasi:

class NamaKelas {

void namaProsedur(){

// isi prosedur

}

}

**2.9** **Fungsi**

Fungsi merupakan kode program yang digunakan untuk menghasilkan sesuatu. Untuk mengembalikan hasil, dapat menggunakan perintah return diikutidengan data yang dihasilkan. Fungsi hanya dapat menghasilkan sebuah nilai. Sebuah kelas dapat memiliki lebih dari satu fungsi. Nama fungsi tidak boleh sama dengan fungsi yang lain.

2.8.1 bentuk – bentuk fungsi

Bentuk umum.

Deklarasi:

class NamaKelas {

TipeData namaFungsi(){

// kode program

return hasil;

}

}

1. Menggunakan fungsi

Bentuk umum.

NamaKelas objek = new NamaKelas();

TipeData hasil = objek.namaFungsi();

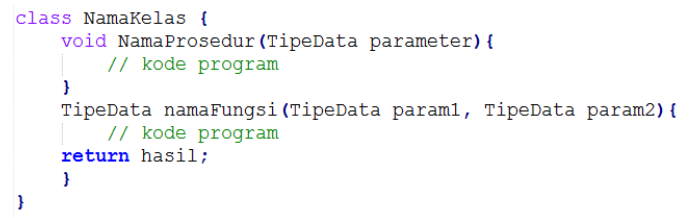
1. Parameter

Parameter merupakan data yang dapat disisipkan ke dalam prosedur

dan fungsi. Jumlah parameter dalam prosedur dan fungsi dapat lebih dari satu.

Bentuk umum.

Deklarasi :



1. Menggunakan fungsi

Bentuk umum.

Deklarasi:

NamaKelas objek = new NamaKelas();

objek.namaProsedur("String");

TipeData hasil = objek.namaFungsi(100, 300);

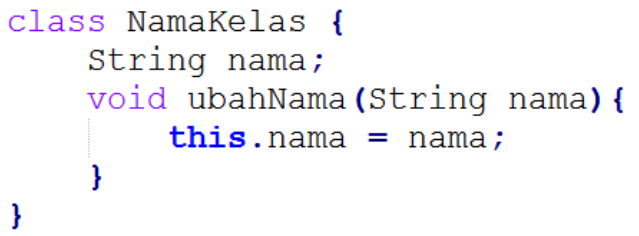
TIpeData variabel = ...;

objek.namaProsedur2(variabel);

* 1. **This**

Kata kunci this digunakan untuk mengakses kelas itu sendiri. Biasanya kata kunci this digunakan ketika mengakses sebuah sebuah atribut kelas, dimana nama atribut tersebut sama dengan nama parameter.

Deklarasi :



* 1. **Overloading**

Overloading merupakan kemampuan java dalam membuat prosedur dan fungsi dengan nama yang sama, namun berbeda jumlah parameter. Parameter dengan jumlah yang sama diperbolehkan dengan ketentuan tipe data parameternya harus berbeda.

* 1. **Konstruktor**

Konstruktor merupakan prosedur yang diakses secara langsung ketika membuat sebuah objek dibuat. Konstruktor dapat memiliki parameter dan juga dapat di overloading. Nama konstruktor harus sama dengan nama kelas.

Bentuk umum.

Deklarasi :

class NamaKelas {

NamaKelas(){

}

// overloading konstruktor

NamaKelas(TipeData parameter){

}

}

* 1. **Paket**

Paket merupakan fitur yang dapat digunakan untuk memanajemen file Kelas. Biasanya kelas-kelas yang sejenis dikumpulkan dalam sebuah paket, misal kelas-kelas data disimpan dalam paket data, kelas-kelas form disimpan dalam paket gui, dan sebagainya.

Membuat paket dilakukan pada kelas, dan dapat menggunakan perintah package diikuti dengan nama paket nya.

Bentuk umum.

Deklarasi:

package namapaket;

class NamaKelas {

}

* + 1. macam-macam paket

1. Paket beritngkat

Sebuah paket dapat dibuat didalam paket lagi dengan menggunakan tanda . (titik).

Bentuk umum.

Deklarasi:

package paket.subpaket;

class NamaKelas {

}

1. Mengakses kelas dari luar paket

Secara default, sebuah kelas tidak dapat menggunakan kelas yang berada dalam luar paket. Untuk menggunakan kelas di luar paket, perlu menggunakan perintah import diikuti dengan nama paket dan kelasnya.

Bentuk umum.

Deklarasi:

import namapaket.subpaket.NamaKelas;

NamaKelas objek = new NamaKelas();

* 1. **Visibility**

Visibility merupakan kemampuan untuk mengontrol sebuah akses ke kelas, atribut, prosedur atau fungsi.

Java mendukung beberapa visibility, yaitu :

• Default

• Public

• Private, dan

• Protected

* 1. **Enkapsulasi (Encapsulation)**

Memastikan atribut objek tidak dapat diubah dan diakses secara langsung. Hanya prosedur dan fungsi tertentu yang diberi izin untuk mengakses dan mengubah atribut objek.

**BAB III**

**PERANCANGAN SISTEM**

* 1. **Sistem Usulan**



**BAB IV**

**IMPLEMENTASI & PEMBAHASAN**

**4.1 Implementasi**

Membuat class konfig untuk menghubungkan form ke database.

package menghitung\_nilai;

import java.sql.Connection;

import java.sql.DriverManager;

import java.sql.SQLException;

public class konfig {

private static Connection MYSQLConfig;

public static Connection configDB()throws SQLException{

try {

String url = "jdbc:mysql://localhost:3306/nilai\_akhir\_mahasiswa";

String user = "root";

String pass = "";

DriverManager.registerDriver(new com.mysql.jdbc.Driver());

MYSQLConfig = DriverManager.getConnection(url,user,pass);

}catch(SQLException e){

System.out.println("Koneksi ke Database Gagal "+ e.getMessage());

}

return MYSQLConfig;

}

}

private void HITUNGActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

double Kehadiran,Proyek,Kuis,UTS,UAS,nilai;

String hasil;

Kehadiran=Integer.parseInt(txtKehadiran.getText());

Proyek=Integer.parseInt(txtProyek.getText());

Kuis=Integer.parseInt(txtKuis.getText());

UTS=Integer.parseInt(txtUTS.getText());

UAS=Integer.parseInt(txtUAS.getText());

nilai=(Kehadiran\*15/100)+(Proyek\*20/100)+(Kuis\*20/100)+(UTS\*20/100)+(UAS\*25/100);

hasil=String.valueOf(nilai);

txtHITUNG.setText(hasil);

if(nilai>=85){

txtNILAI.setText("A");

txtKETERANGAN.setText("LULUS");

}

else if ((nilai<85)&&(nilai>=81)){

txtNILAI.setText("A-");

txtKETERANGAN.setText("LULUS");

}

else if ((nilai<80)&&(nilai>=76)){

txtNILAI.setText("B+");

txtKETERANGAN.setText("LULUS");

}

else if ((nilai<75)&&(nilai>=71)){

txtNILAI.setText("B");

txtKETERANGAN.setText("LULUS");

}

else if ((nilai<70)&&(nilai>=66)){

txtNILAI.setText("B-");

txtKETERANGAN.setText("LULUS");

}

else if ((nilai<65)&&(nilai>=61)){

txtNILAI.setText("C+");

txtKETERANGAN.setText("LULUS");

}

else if ((nilai<60)&&(nilai>=51)){

txtNILAI.setText("C");

txtKETERANGAN.setText("Mengulang");

}

else if ((nilai<50)&&(nilai>=45)){

txtNILAI.setText("D");

txtKETERANGAN.setText("Mengulang");

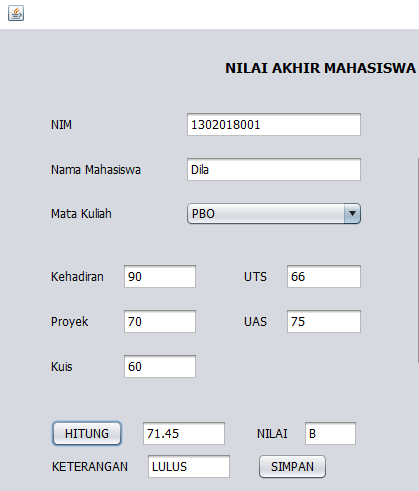
}

else if ((nilai<45)&&(nilai>=0)){

txtNILAI.setText("E");

txtKETERANGAN.setText("Tidak Lulus");

}



ta.setText(

"NIM: "+txtNIM.getText()+"\n"+

"Nama: "+txtNama.getText()+"\n"+

"Mata Kuliah: "+cbMataKuliah.getSelectedItem()+"\n"+

"Absen: "+txtKehadiran.getText()+"\n"+

"Tugas: "+txtProyek.getText()+"\n"+

"Tugas: "+txtKuis.getText()+"\n"+

"UTS: "+txtUTS.getText()+"\n"+

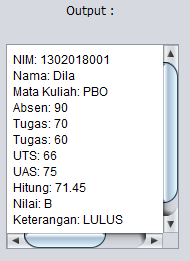
"UAS: "+txtUAS.getText()+"\n"+

"Hitung: "+txtHITUNG.getText()+"\n"+

"NILAI: "+txtNILAI.getText()+"\n"+

"Keterangan: "+txtKETERANGAN.getText()+"\n");

}



private void SIMPANActionPerformed(java.awt.event.ActionEvent evt) {

// TODO add your handling code here:

try{

String sql = "INSERT INTO nilai\_akhir\_mahasiswa VALUES ('"+txtNIM.getText()+"','"+txtNama.getText()+"','"+cbMataKuliah.getSelectedItem()+"','"+txtKehadiran.getText()+"','"+txtProyek.getText()+"','"+txtKuis.getText()+"','"+txtUTS.getText()+"','"+txtUAS.getText()+"')";

java.sql.Connection conn = (Connection)konfig.configDB();

java.sql.PreparedStatement pstm = conn.prepareStatement(sql);

pstm.execute();

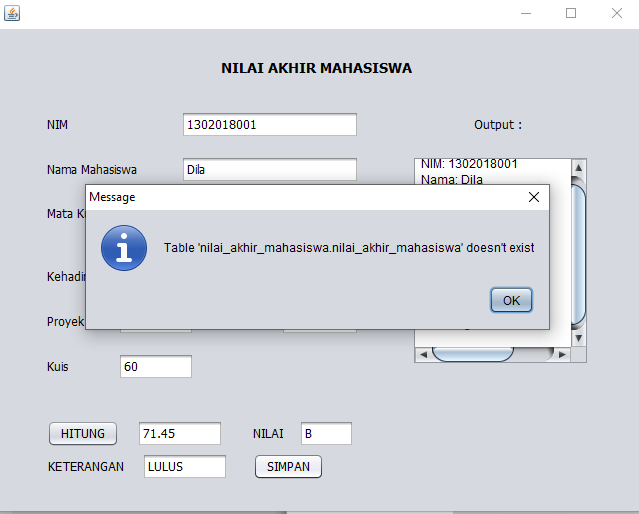
JOptionPane.showMessageDialog(null, "Proses Simpan Data Berhasil..");

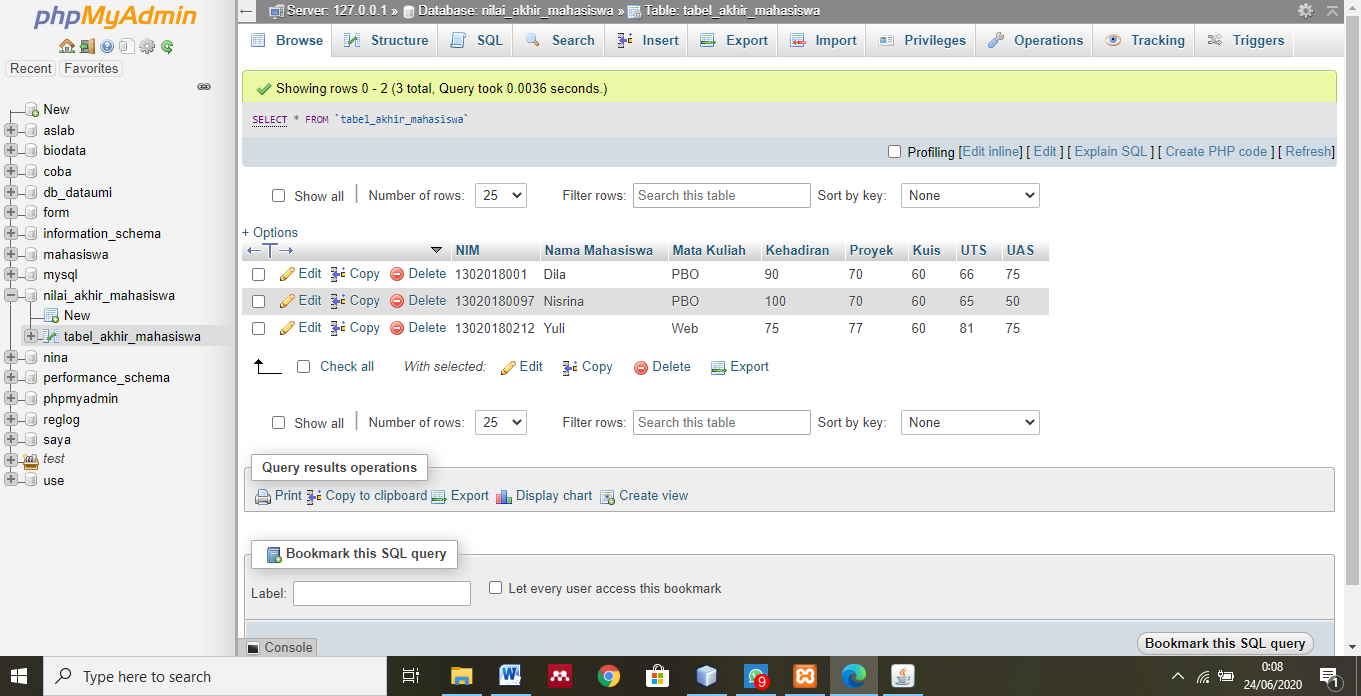
}catch(HeadlessException | SQLException e){

JOptionPane.showMessageDialog(this, e.getMessage());

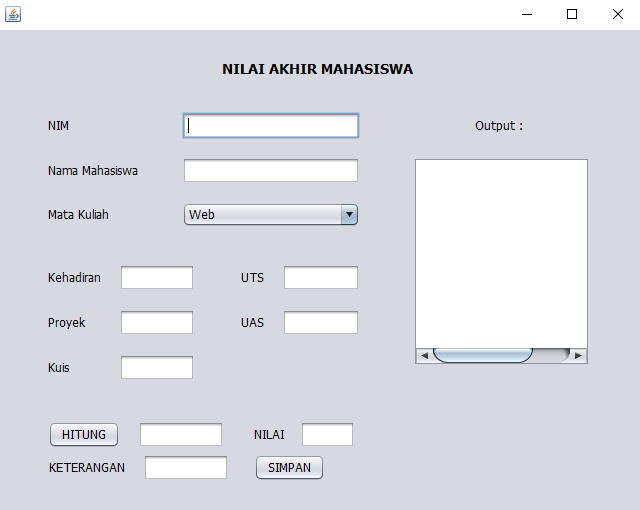
}

}





**Tampilan Luar :**



**BAB V**

**KESIMPULAN & SARAN**

**5.1 Kesimpulan**

Di Era Milenial sekarang teknologi berkembang pesat, dimana teknologi sangat dibutuhkan untuk membantu berbagai pekerjaan agar pekerjaan dapat lebih cepat selesai, lebih efisien dan lebih fleksibel.

Dengan adanya program menghitung nilai akhir mahasiswa, penilaian mata kuliah untuk dosen akan lebih mudah dan waktu yang terpakai lebih efisien.

**5.2 Saran**

Saran saya untuk pemrograman berorientasi onjek (PBO) semoga dengan adanya proyek ini bisa membantu kami kedepannya.

***Daftar Pustaka***

Website :

***Hari Aspriono****,* <https://www.hariaspriyono.com/2018/07/materi-pemrograman-berorientasi-objek.html>, ***Diakses Tanggal 18 Juni 2020***

***Pulti Opok***, <http://www.pultiopok.com/2017/12/flowchart-program-menghitung-nilai.html>, ***Diakses Tanggal 19 Juni 2020***

***Dhanu Tirtha,*** <http://dhanutirtha.blogspot.com/2012/11/algoritma-menghitung-jumlah-nilai.html>, ***Diakses Tanggal 20 Juni 2020***