LAPORAN PRAKTIKUM
PEMROGRAMAN II
MODUL 6

GRAPHICAL USER INTERFACE (GUI)

Oleh:

Ryan Muhammad Irfan NIM. 2210817310013

PROGRAM STUDI TEKNOLOGI INFORMASI
FAKULTAS TEKNIK
UNIVERSITAS LAMBUNG MANGKURAT
DESEMBER 2023

LEMBAR PENGESAHAN
LAPORAN PRAKTIKUM PEMROGRAMAN II
MODUL 6

Laporan Praktikum Pemrograman II Modul 6: Graphical User Interface (GUI) ini disusun sebagai syarat lulus mata kuliah Praktikum Pemrograman II. Laporan Praktikum ini dikerjakan oleh:

Nama Praktikan : Ryan Muhammad Irfan
NIM : 2210817310013

Menyetujui, Asisten Praktikum	Mengetahui, Dosen Penanggung Jawab Praktikum
Bachrul Uluum NIM. 2010817210025	Andreyan Rizky Baskara, S.Kom., M.Kom. NIP. 19930703 201903 1 011

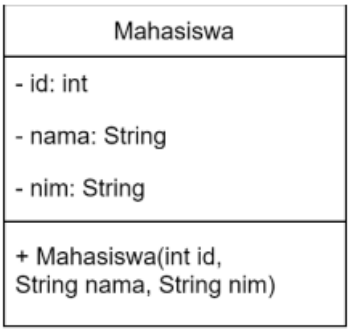
DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

SOAL 1

Diberikan class diagram seperti berikut: (isi program harus sesuai dengan class diagram)



Gambar 1. Soal 1 (Class Diagram)

Implementasikan class diagram diatas menjadi class pada Bahasa pemrograman java. Class mahasiswa harus menerapkan setter dan getter.

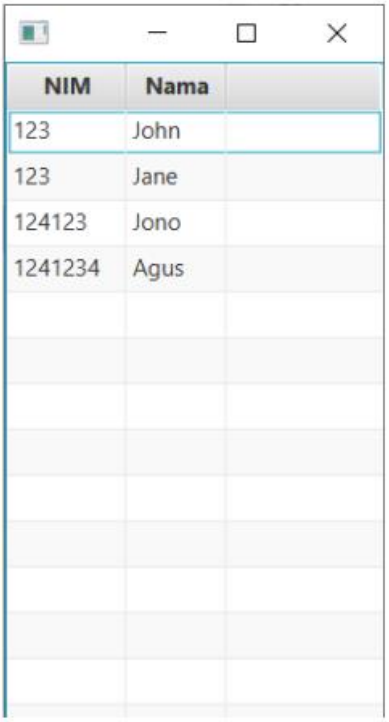
Program harus menampilkan list data dalam bentuk tabel.

Kolom dari tabel adalah :

- NIM
- NAMA

Kemudian buatlah 10 data secara *hardcode* untuk ditampilkan pada tabel

Contoh program dapat dilihat sebagai berikut:



Gambar 2. Soal 1 (Contoh Program)

Simpan coding anda dengan nama folder/proyek: **PRAKTIKUM6**

A. Source Code

Mahasiswa.java

```
1   package com.example.praktikum6;
2
3   public class Mahasiswa {
4       private String nama, nim;
5
6       public Mahasiswa(String nama, String nim) {
7           this.nama = nama;
8           this.nim = nim;
9       }
10
11      public String getNama() {
12          return nama;
13      }
14
15      public void setNama(String nama) {
16          this.nama = nama;
17      }
18
```

```
19      public String getNim() {
20          return nim;
21      }
22
23      public void setNim(String nim) {
24          this.nim = nim;
25      }
26  }
```

Tabel 1. Source Code Soal 1(Mahasiswa.java)

Tabel.java

```
1   package com.example.praktikum6;
2
3   import javafx.application.Application;
4   import javafx.scene.Scene;
5   import javafx.scene.control.TableColumn;
6   import javafx.scene.control.TableView;
7   import javafx.scene.control.cell.PropertyValueFactory;
8   import javafx.scene.layout.VBox;
9   import javafx.stage.Stage;
10
11  public class Tabel extends Application {
12
13      public static void main(String[] args) {
14          launch(args);
15      }
16
17      @Override
18      public void start(Stage primaryStage) {
19          TableView tabel = new TableView();
20          TableColumn<Mahasiswa, String> kolomNIM = new
    TableColumn<>("NIM");
21          kolomNIM.setCellValueFactory(
22                  new PropertyValueFactory<>("nim")
23          );
24          TableColumn<Mahasiswa, String> kolomNama = new
    TableColumn<>("Nama");
25          kolomNama.setCellValueFactory(
26                  new PropertyValueFactory<>("nama")
27          );
28
29          tabel.getColumns().addAll(kolomNIM, kolomNama);
30          tabel.getItems().add(new          Mahasiswa("Ryan
    Muhammad Irfan", "2210817310013"));
31          tabel.getItems().add(new         Mahasiswa("Hafiz
    Pratama Budiman", "2210817310007"));
32          tabel.getItems().add(new          Mahasiswa("Bima
    Sanjaya", "2210817210008"));
```

```
33          tabel.getItems().add(new       Mahasiswa("Akhmad
    Raihan Ridha", "2210817110001"));
34          tabel.getItems().add(new Mahasiswa("Trisna Cahya
    Permadi", "2210817210021"));
35          tabel.getItems().add(new        Mahasiswa("Kevin
    Maleakhi", "2210817210031"));
36          tabel.getItems().add(new  Mahasiswa("Ahmad  Reza
    Alfayiet", "2210817210016"));
37          tabel.getItems().add(new  Mahasiswa("Riyo  Aurora
    Gusion", "2210817310016"));
38          tabel.getItems().add(new    Mahasiswa("Ady    T.
    Adilang", "2210817710001"));
39          tabel.getItems().add(new  Mahasiswa("M.Daffa  Az-
    Zikra", "2210917310011"));
40
41          VBox box = new VBox(tabel);
42          Scene scene = new Scene(box);
43
44          primaryStage.setScene(scene);
45          primaryStage.show();
46      }
47  }
```

Tabel 2. Source Code Soal 1 (Tabel.java)

B. Output Program

Nama	NIM
Ryan Muhammad Irfan	2210817310013
Hafiz Pratama Budiman	2210817310007
Bima Sanjaya	2210817210008
Akhmad Raihan Ridha	2210817110001
Trisna Cahya Permadi	2210817210021
Kevin Maleakhi	2210817210031
Ahmad Reza Alfayiet	2210817210016
Riyo Aurora Gusion	2210817310016
Ady Adilang	2210817710001
M.Daffa Az-Zikra	2210917310011

Gambar 3. Screenshot Hasil Jawaban Soal 1

C. Pembahasan

Mahasiswa.java

Pada baris [1], `package com.example.praktikum6;`
"package com.example.praktikum6" berfungsi untuk mendeklarasikan package dengan nama com.example.praktikum6. Class Mahasiswa berada dalam package com.example.praktikum6.

";" berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3-26], `public class Mahasiswa`
"public" berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

"Mahasiswa" berfungsi untuk membuat class yang dalam baris ini diberi nama Mahasiswa.

"{…}" berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [4], `private String nama, nim;`
"private" berfungsi sebagai penanda bahwa atribut dapat memiliki akses terbatas. Hanya bisa diakses dari dalam class itu sendiri.

“String” merupakan tipe data dari atribut nama, nim.

“nama, nim” merupakan atribut dari sebuah class Mahasiswa.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [6-9], `public Mahasiswa(String nama, String nim){…}`

“public” berfungsi sebagai penanda bahwa constructor dapat diakses dari class lain.

“Mahasiswa(String nama, String nim)” merupakan sebuah constructor dari class Mahasiswa. Constructor ini menerima parameter String naman, String nim.

“{…}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [7], `this.nama = nama;`
“this.nama = nama” berfungsi untuk menginisialisasi atribut “nama” dengan nilai yang diterima oleh parameter “nama”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [8], `this.nim = nim;`
“this.nim = nim” berfungsi untuk menginisialisasi atribut “nim” dengan nilai yang diterima oleh parameter “nim”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [11-13], `public String getNama() {…}`
“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“String” merupakan tipe data yang dikembalikan oleh method getNama ().

“getNama()” merupakan method getter. Digunakan untuk mengambil nilai dari atribut ‘nama’.

“{…}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [12], `return nama;`
“return nama” berfungsi untuk mengembalikan nilai atribut ‘nama’ ketika method getNama() dipanggil.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [15-17], `public void setNama(String nama){…}`
“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari class lain.

“void” berfungsi menyatakan method tersebut tidak mengembalikan nilai.

“setNama()” merupakan method setter. Digunakan untuk mengatur atau mengubah nilai atribut yang private.

“(String nama) merupakan parameter dari method setter. String adalah tipe datanya. Parameter ini digunakan untuk mengatur nilai atribut “Nama”

“{…}” berfungsi untuk memulai dan mengakhiri blok kode

Pada baris [16], `this.nama = nama;`
“this.nama = nama” berfungsi untuk menginisialisasi atribut “nama” dengan nilai yang diterima oleh parameter “nama”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [19-21], `public String getNim() {…}`
“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari luar class.

“String” merupakan tipe data yang dikembalikan oleh method getNim ().

“getNim()” merupakan method getter. Digunakan untuk mengambil nilai dari atribut ‘nim’.

“{…}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [20], `return nim;`
“return nama” berfungsi untuk mengembalikan nilai atribut ‘nim’ ketika method getNim() dipanggil.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [23-25], `public void setNim(String nim){…}`
“public” berfungsi sebagai penanda bahwa method ini dapat diakses dari class lain.

“void” berfungsi menyatakan method tersebut tidak mengembalikan nilai.

“setNim()” merupakan method setter. Digunakan untuk mengatur atau mengubah nilai atribut yang private.

“(String nim) merupakan parameter dari method setter. String adalah tipe datanya. Parameter ini digunakan untuk mengatur nilai atribut “nim”

“{…}” berfungsi untuk memulai dan mengakhiri blok kode

Pada baris [24], `this.nim = nim;`
“this.nim = nim” berfungsi untuk menginisialisasi atribut “nim” dengan nilai yang diterima oleh parameter “nim”.

Tabel.java

Pada baris [1], `package com.example.praktikum6;`
“package com.example.praktikum6” berfungsi untuk mendeklarasikan package dengan nama com.example.praktikum6. Class Tabel berada dalam package com.example.praktikum6.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [3], `import javafx.application.Application;`
Mengimpor class Application yang terdapat di dalam package javafx.application.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [4], `import javafx.scene.Scene;`
Mengimpor class Scene yang terdapat di dalam package javafx.scene.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [5], `import javafx.scene.control.TableColumn;`
Mengimpor class TableColumn yang terdapat di dalam package javafx.scene.control.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [6], `import javafx.scene.control.TableView;`
Mengimpor class TableView yang terdapat di dalam package javafx.scene.control.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [7],
`import javafx.scene.control.cell.PropertyValueFactory;`
Mengimpor class PropertyValueFactory yang terdapat di dalam package javafx.scene.control.cell.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [8], `import javafx.scene.layout.VBox;`
Mengimpor class Vbox yang terdapat di dalam package javafx.scene.layout.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [9], `import javafx.stage.Stage;`
Mengimpor class Stage yang terdapat di dalam package javafx.stage.Stage.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [11-47], `public class Tabel extends Application{…}`
“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“class” berfungsi untuk membuat class yang dalam baris ini diberi nama Tabel.

“extends Application” berfungsi untuk mewariskan method atau atribut dari kelas Application javafx.application.

“{…}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [13-15], `public static void main(String[] args){…}`
“public” berfungsi sebagai penanda bahwa class, objek, method, atau atribut dapat diakses dari class lain.

“static” berfungsi membuat suatu method tanpa perlu melakukan instansiasi terlebih dahulu.

“void” berfungsi untuk tidak mengembalikan nilai apapun.

“main” merupakan nama fungsi yang digunakan oleh java sebagai awal masuk ke program.

“String[] args” berfungsi sebagai parameter yang diperlukan oleh fungsi main. Parameter ini adalah array dari argument perintah yang bisa diteruskan ke program java.

“{…}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [14], `launch(args);`
“launch(args)” berfungsi untuk memulai untuk mengeksekusi aplikasi JavaFX .

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [18-46], `public void start(Stage primaryStage){…}`
Merupakan sebuah method yang diperlukan untuk mengimplementasikan class Application dalam JavaFX. Berfungsi untuk menampilkan (GUI). “Stage primaryStage” merupakan parameter untuk mewakili jendela utama.

“{…}” berfungsi untuk memulai dan mengakhiri blok kode.

Pada baris [19], `TableView tabel = new TableView();`
Berfungsi untuk membuat tabel dalam JavaFX. “tabel” merupakan nama variabel yang digunakan merujuk ke objek yang akan dibuat dengan menggunakan class TableView . “new Tableview” membuat objek baru dari dari kelas Tableview.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [20], `TableColumn<Mahasiswa, String> kolomNIM = new TableColumn<>("NIM");`
Berfungsi untuk membuat kolom pada tabel di JavaFX dengan menerima parameter “<Mahasiswa, String>”. Kolom ini menampilkan data dari objek dengan tipe “Mahasiswa” dan tipe datanya “String”. “kolomNIM” nama variabel untuk objek kolom tersebut . Kolom ini akan diberi nama “NIM”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [21-23],
```
kolomNIM.setCellValueFactory(
        new PropertyValueFactory<>("nim")
);
```
“kolomNIM” merupakan sebuah objek dari TableColumn yang diberi nama “NIM” pada table.

“setCellValueFactory” Merupakan method yang berfungsi untuk mengatur nilai sel untuk kolom yang diberi nama ”NIM”.

“new PropertyValueFactory<>("nim")” berfungsi untuk membuat objek baru “PropertyValueFactory” yang berfungsi menerima parameter “nim”, dari atribut nim kelas mahasiswa yang digunakan untuk mengisi kolom yang diberi nama “NIM”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [25-27], `kolomNama.setCellValueFactory(`
```
                new PropertyValueFactory<>("nama")
        );{…}
```
“kolomNama” merupakan sebuah objek dari TableColumn yang diberi nama “Nama” pada tabel.

“setCellValueFactory” Merupakan method yang berfungsi untuk mengatur nilai sel untuk kolom yang diberi nama ”Nama”.

“new PropertyValueFactory<>("nama")” berfungsi untuk membuat objek baru “PropertyValueFactory” yang berfungsi menerima parameter “nama”, dari atribut nim kelas mahasiswa yang digunakan untuk mengisi kolom yang diberi nama “Nama”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [29], `tabel.getColumns().addAll(kolomNIM, kolomNama);`
Berfungsi untuk menambahkan kolom “kolomNIM dan kolomNama” pada objek “tabel”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [30], `tabel.getItems().add(new Mahasiswa("Ryan Muhammad Irfan", "2210817310013"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: “Ryan Muhammad Irfan”, nim: “2210817310013”.

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [31], `tabel.getItems().add(new Mahasiswa("Hafiz Pratama Budiman", "2210817310007"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Hafiz Pratama Budiman", nim: "2210817310007".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [32], `tabel.getItems().add(new Mahasiswa("Bima Sanjaya", "2210817210008"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Bima Sanjaya", nim: "2210817210008".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [33], `tabel.getItems().add(new Mahasiswa("Akhmad Raihan Ridha", "2210817110001"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Akhmad Raihan Ridha", nim: "2210817110001".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [34], `tabel.getItems().add(new Mahasiswa("Trisna Cahya Permadi", "2210817210021"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Trisna Cahya Permadi", nim: "2210817210021".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [35], `tabel.getItems().add(new Mahasiswa("Kevin Maleakhi", "2210817210031"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Kevin Maleakhi", nim: "2210817210031".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [36], `tabel.getItems().add(new Mahasiswa("Ahmad Reza Alfayiet", "2210817210016"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Ahmad Reza Alfayiet", nim: "2210817210016".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [37], `tabel.getItems().add(new Mahasiswa("Riyo Aurora Gusion", "2210817310016"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Riyo Aurora Gusion", nim: "2210817310016".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [38], `tabel.getItems().add(new Mahasiswa("Ady T. Adilang", "2210817710001"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "Ady T. Adilang", nim: "2210817710001".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [39], `tabel.getItems().add(new Mahasiswa("M.Daffa Az-Zikra", "2210917310011"));`
Berfungsi untuk menambahkan objek “Mahasiswa” baru ke dalam daftar item pada objek tabel dengan menerima parameter yang dibuat dari constructor class Mahasiswa. Disini menerima value dengan parameter nama: "M.Daffa Az-Zikra", nim: "2210917310011".

“;” berfungsi untuk mengakhiri sebuah perintah.

Pada baris [41], `VBox box = new VBox(tabel);`
Berfungsi untuk mengatur tata letak komponen secara vertical didalam JavaFX. Pada baris ini objek "tabel" akan diatur letaknya secara vertical.

";" berfungsi untuk mengakhiri sebuah perintah.

Pada baris [42], `Scene scene = new Scene(box);`
Berfungsi sebagai elemen utama yang akan ditampilkan dalam JavaFX. Dalam hal ini element yang akan ditampilkan adalah objek "box" yang didalamnya terdapat tabel.

";" berfungsi untuk mengakhiri sebuah perintah.

Pada baris [44], `primaryStage.setScene(scene);`
Berfungsi untuk menetapkan tampilan yaitu "scene" yang akan ditampilkan di main window pada javaFX. primaryStage adalah objek yang mewakili main window.

";" berfungsi untuk mengakhiri sebuah perintah.

Pada baris [45], `primaryStage.show();`
Berfungsi untuk meanampilan main window pada javaFX yang telah diatur sebelumnya.

";" berfungsi untuk mengakhiri sebuah perintah.

D. Tautan GIT

https://github.com/ryanmi04/Praktikum-Pemrograman-2-Paralel-1/tree/main/PRAKTIKUM6/src/main/java/com/example/praktikum6