

## **LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : Rekayasa Perangkat Lunak 2  
Kelas : 4IA11  
Praktikum ke- 4  
Tanggal : 8 November 2024  
Materi : Orm dan Interface pada java  
NPM : 50421786  
Nama : Marsillo Dito Saputra  
Ketua Asisten :  
Paraf Asisten :  
Nama Asisten : Dimas Renaldy Sriwirawan  
Jumlah Lembar : Lembar

**LABORATORIUM TEKNIK INFORMATIKA**

**UNIVERSITAS GUNADARMA**

**2024**

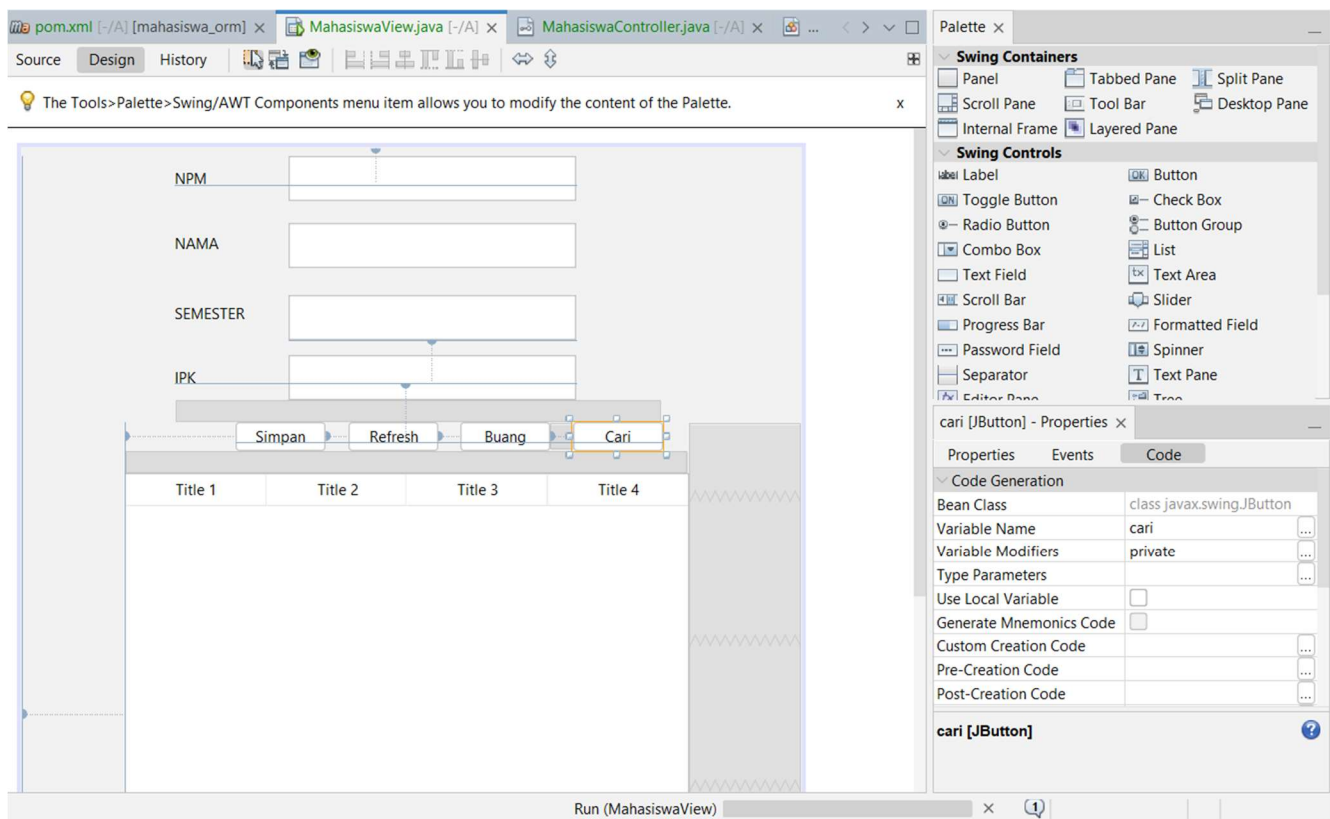
## LISTING PROGRAM

(Jawab Soal Untuk Laporan Akhir)

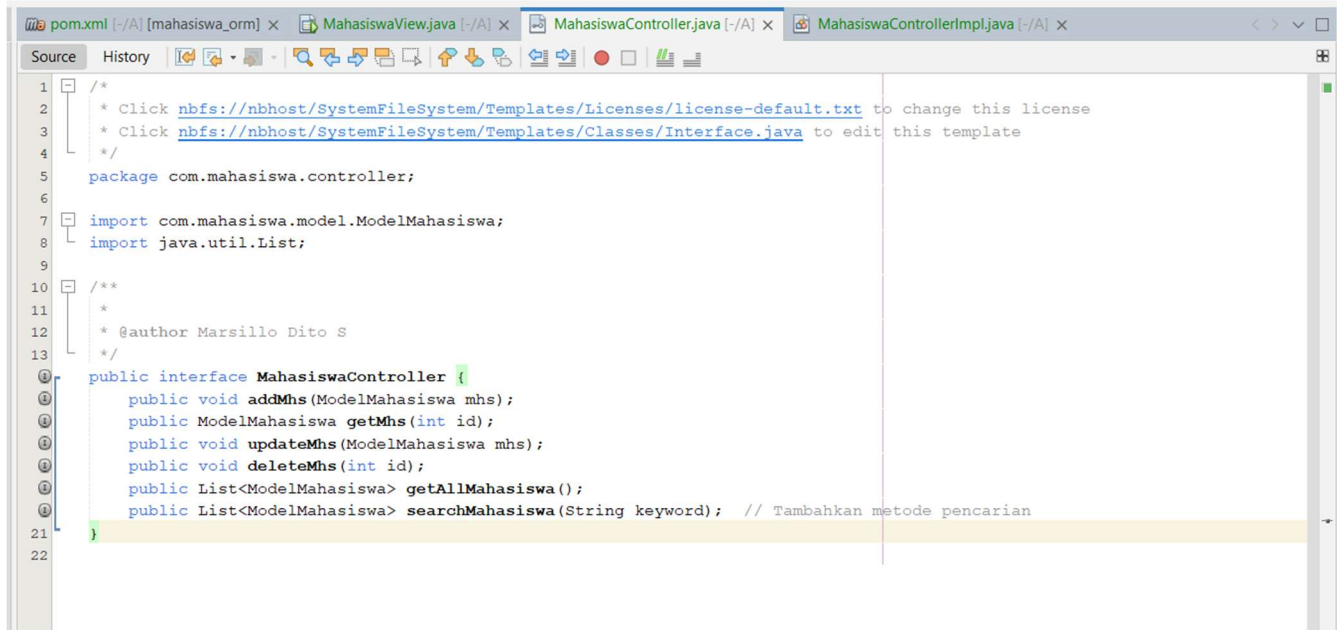
Untuk laporan akhir, buatlah fitur search pada GUI nya  
jelaskan penambahan apa yang dilakukan dan logika nya

**Jawab :**

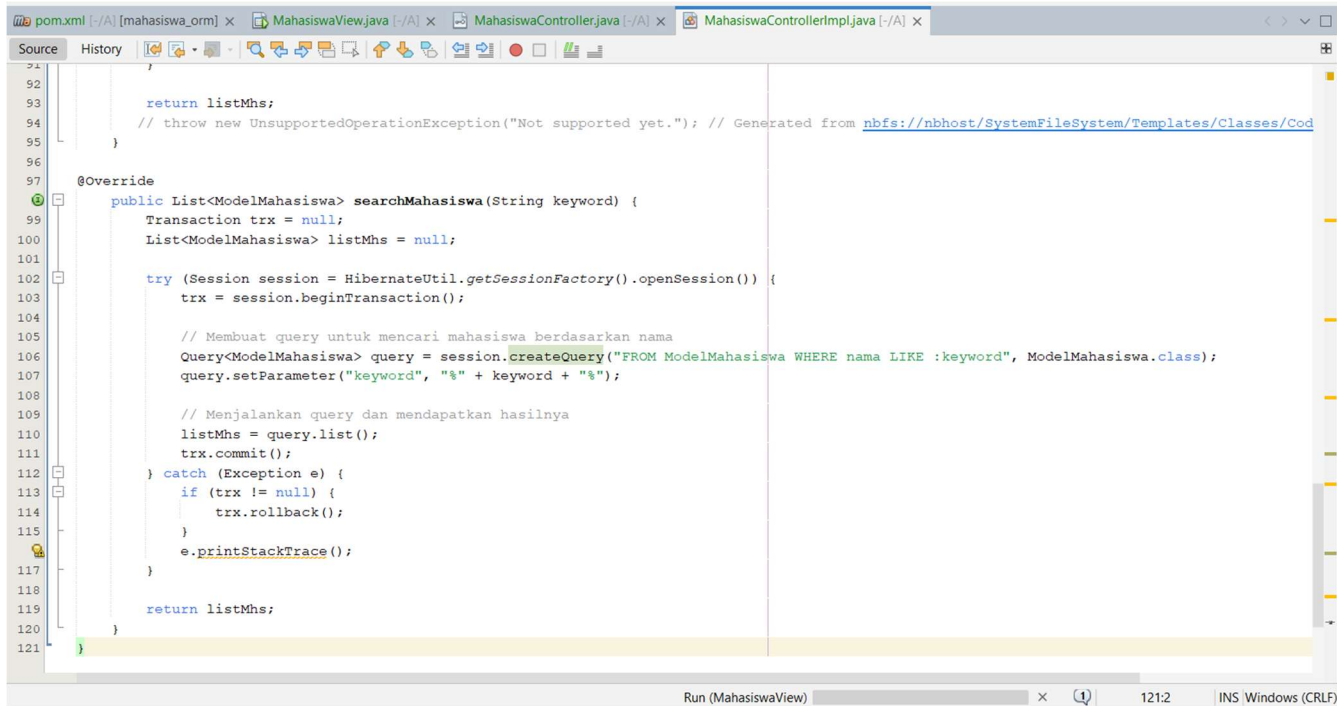
### MENAMBAHKAN BUTTON SEARCH PADA TAMPILANNYA



## Menambahkan searchMahasiswa pada mahasiswacontrollerImpl dan memanggilnya di mahasiswaController



```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Interface.java to edit this template
4   */
5   package com.mahasiswa.controller;
6
7   import com.mahasiswa.model.ModelMahasiswa;
8   import java.util.List;
9
10  /**
11   *
12   * @author Marsillo Dito S
13   */
14  public interface MahasiswaController {
15      public void addMhs(ModelMahasiswa mhs);
16      public ModelMahasiswa getMhs(int id);
17      public void updateMhs(ModelMahasiswa mhs);
18      public void deleteMhs(int id);
19      public List<ModelMahasiswa> getAllMahasiswa();
20      public List<ModelMahasiswa> searchMahasiswa(String keyword); // Tambahkan metode pencarian
21  }
22
```



```
92
93
94     return listMhs;
95     // throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Cod
96 }
97
98 @Override
99 public List<ModelMahasiswa> searchMahasiswa(String keyword) {
100     Transaction trx = null;
101     List<ModelMahasiswa> listMhs = null;
102
103     try (Session session = HibernateUtil.getSessionFactory().openSession()) {
104         trx = session.beginTransaction();
105
106         // Membuat query untuk mencari mahasiswa berdasarkan nama
107         Query<ModelMahasiswa> query = session.createQuery("FROM ModelMahasiswa WHERE nama LIKE :keyword", ModelMahasiswa.class);
108         query.setParameter("keyword", "%" + keyword + "%");
109
110         // Menjalankan query dan mendapatkan hasilnya
111         listMhs = query.list();
112         trx.commit();
113     } catch (Exception e) {
114         if (trx != null) {
115             trx.rollback();
116         }
117         e.printStackTrace();
118     }
119
120     return listMhs;
121 }
```

## MENAMBAHKAN FITUR GUI SEARCH BUTTON KETIKA DI KLIK

```
pom.xml [-/A] [mahasiswa_orm] x MahasiswaView.java [-/A] x MahasiswaController.java [-/A] x MahasiswaControllerImpl.java [-/A] x
Source Design History
249 private void cariActionPerformed(java.awt.event.ActionEvent evt) {
250     // Membuat JFrame untuk dialog box
251     JFrame dialog = new JFrame("Cari Mahasiswa");
252     dialog.setSize(300, 150);
253     dialog.setLocationRelativeTo(null); // Agar dialog muncul di tengah layar
254     dialog.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
255
256     // Membuat panel untuk menampung JTextField dan tombol OK
257     JPanel panel = new JPanel();
258     panel.setLayout(new FlowLayout());
259
260     // Membuat JTextField untuk memasukkan ID mahasiswa
261     JTextField txtIdMahasiswa = new JTextField(20);
262     JLabel lblId = new JLabel("Masukkan ID Mahasiswa: ");
263
264     // Membuat tombol OK
265     JButton btnOk = new JButton("OK");
266
267     // Menambahkan komponen ke dalam panel
268     panel.add(lblId);
269     panel.add(txtIdMahasiswa);
270     panel.add(btnOk);
271
272     // Menambahkan panel ke dialog
273     dialog.add(panel);
274
275     // Menambahkan action listener untuk tombol OK
276     btnOk.addActionListener(new ActionListener() {
277         @Override
278         public void actionPerformed(ActionEvent e) {
279             // Mendapatkan ID mahasiswa yang dimasukkan
280
281             // Mendapatkan ID mahasiswa yang dimasukkan
282             String idMahasiswa = txtIdMahasiswa.getText().trim();
283
284             // Panggil metode untuk mencari data mahasiswa berdasarkan ID
285             cariDataMahasiswa(idMahasiswa);
286
287             // Menutup dialog setelah pencarian selesai
288             dialog.dispose();
289         }
290     });
291
292     // Menampilkan dialog
293     dialog.setVisible(true);
294 }
295
296 private void cariDataMahasiswa(String idMahasiswa) {
297     // Panggil metode untuk mencari data mahasiswa dari database
298     Mahasiswa mahasiswa = cariMahasiswaDariDatabase(idMahasiswa);
299
300     if (mahasiswa != null) {
301         // Jika data mahasiswa ditemukan, tampilkan data mahasiswa di dalam dialog
302         tampilkanDialogMahasiswa(mahasiswa);
303     } else {
304         // Jika data tidak ditemukan, tampilkan pesan error
305         JOptionPane.showMessageDialog(null, "Mahasiswa dengan ID " + idMahasiswa + " tidak ditemukan.");
306     }
307 }
```

```

307 private Mahasiswa cariMahasiswaDariDatabase(String idMahasiswa) {
308     Mahasiswa mahasiswa = null;
309
310     // Koneksi ke database (Gantilah dengan pengaturan database Anda)
311     String url = "jdbc:mysql://localhost:8111/hibernate_db"; // Ganti dengan URL database Anda
312     String username = "root"; // Ganti dengan username database Anda
313     String password = ""; // Ganti dengan password database Anda
314
315     // SQL query untuk mencari mahasiswa berdasarkan ID
316     String query = "SELECT * FROM mahasiswa WHERE id = ?";
317
318     try (Connection conn = DriverManager.getConnection(url, username, password);
319         PreparedStatement stmt = conn.prepareStatement(query)) {
320
321         // Menyusun query
322         stmt.setString(1, idMahasiswa);
323
324         // Eksekusi query
325         ResultSet rs = stmt.executeQuery();
326
327         // Cek apakah ada hasilnya
328         if (rs.next()) {
329             // Ambil data dari ResultSet dan buat objek mahasiswa
330             mahasiswa = new Mahasiswa(
331                 rs.getString("id"),
332                 rs.getString("nama"),
333                 rs.getString("semester"),
334                 rs.getDouble("ipk")
335             );
336         }
337     } catch (SQLException e) {
338         e.printStackTrace();
339     }
340
341     return mahasiswa;
342 }
343
344 private void tampilkanDialogMahasiswa(Mahasiswa mahasiswa) {
345     // Membuat JFrame untuk menampilkan data mahasiswa
346     JFrame dialogData = new JFrame("Data Mahasiswa");
347     dialogData.setSize(300, 200);
348     dialogData.setLocationRelativeTo(null);
349     dialogData.setDefaultCloseOperation(JFrame.DISPOSE_ON_CLOSE);
350
351     // Membuat panel untuk menampung data mahasiswa
352     JPanel panelData = new JPanel();
353     panelData.setLayout(new BoxLayout(panelData, BoxLayout.Y_AXIS)); // Vertikal layout
354
355     // Menampilkan data mahasiswa di dalam label
356     panelData.add(new JLabel("ID: " + mahasiswa.getId()));
357     panelData.add(new JLabel("Nama: " + mahasiswa.getNama()));
358     panelData.add(new JLabel("Semester: " + mahasiswa.getSemester()));
359     panelData.add(new JLabel("IPK: " + mahasiswa.getIpk()));
360
361     // Menambahkan panel ke dialog
362     dialogData.add(panelData);
363
364     // Menampilkan dialog data mahasiswa
365     dialogData.setVisible(true);
366 }

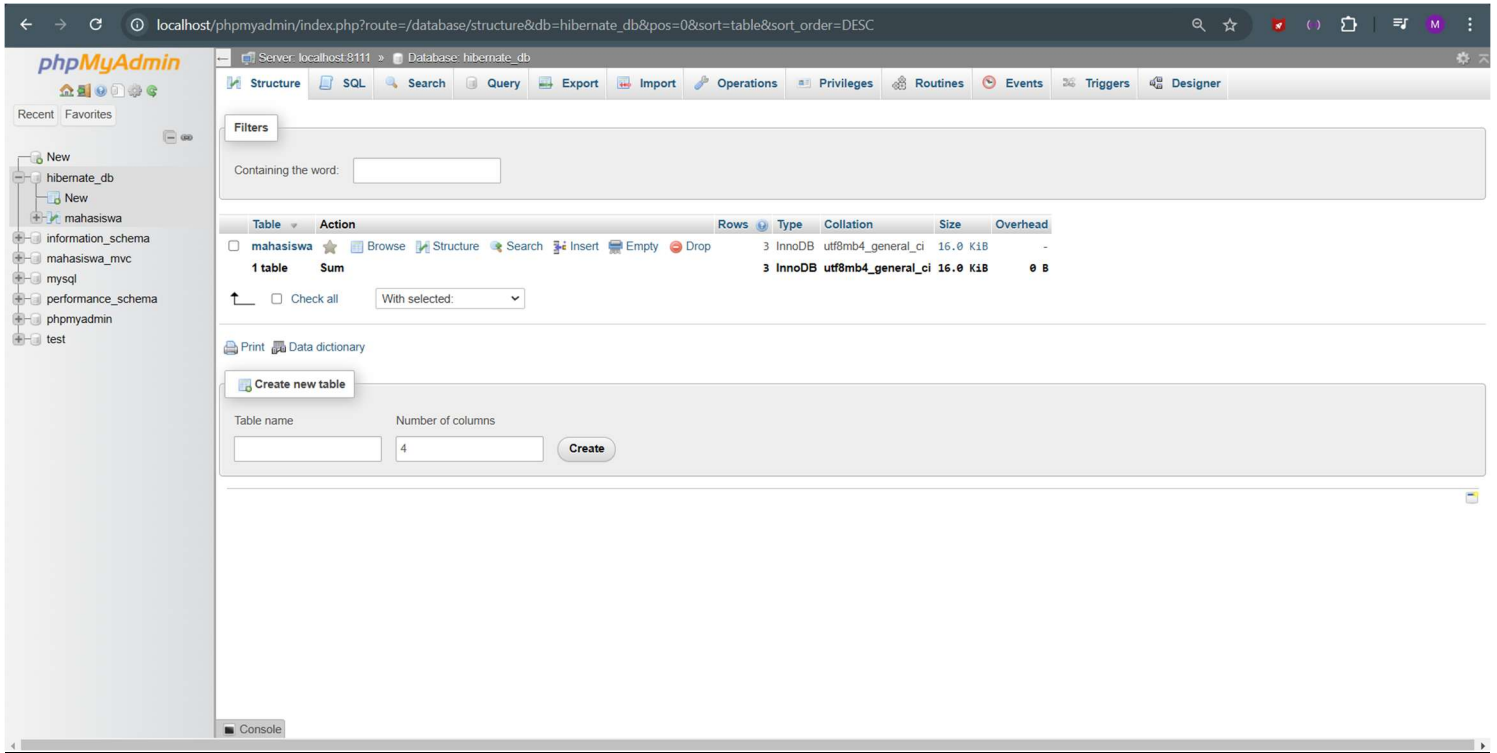
```

```

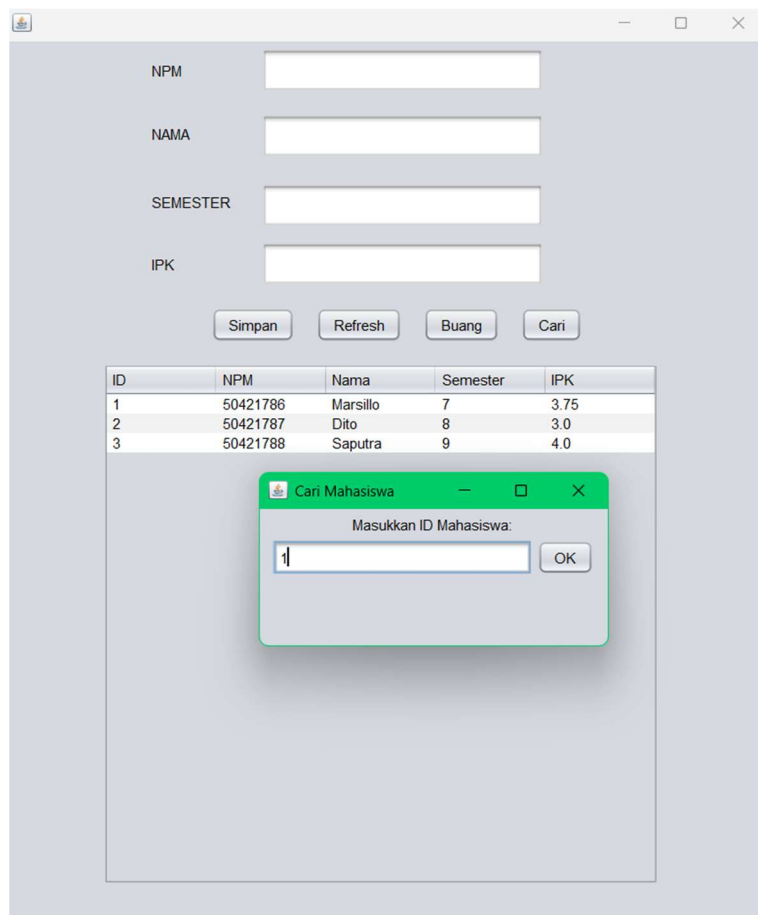
368 // Kelas Mahasiswa untuk menyimpan data mahasiswa
369 class Mahasiswa {
370     private String id;
371     private String nama;
372     private String semester;
373     private double ipk;
374
375     public Mahasiswa(String id, String nama, String semester, double ipk) {
376         this.id = id;
377         this.nama = nama;
378         this.semester = semester;
379         this.ipk = ipk;
380     }
381
382     public String getId() {
383         return id;
384     }
385
386     public String getNama() {
387         return nama;
388     }
389
390     public String getSemester() {
391         return semester;
392     }
393
394     public double getIpk() {
395         return ipk;
396     }
397 }
398
399

```

## DATA PADA MYSQL



## OUTPUT PENCARIAN DATA



NPM

NAMA

SEMESTER

IPK

Data Mahasiswa

ID: 1

Nama: Marsillo

Semester: 7

IPK: 3.75

Simpan

Refresh

Buang

Cari

ID	NPM	Nama	Semester	IPK
1	50421786	Marsillo	7	3.75
2	50421787	Dito	8	3.0
3	50421788	Saputra	9	4.0

NPM

NAMA

SEMESTER

IPK

Data Mahasiswa

ID: 2

Nama: Dito

Semester: 8

IPK: 3.0

Simpan

Refresh

Buang

Cari

ID	NPM	Nama	Semester	IPK
1	50421786	Marsillo	7	3.75
2	50421787	Dito	8	3.0
3	50421788	Saputra	9	4.0

NPM

NAMA

SEMESTER

IPK

Data Mahasiswa

ID: 3

Nama: Saputra

Semester: 9

IPK: 4.0

Simpan

Refresh

Buang

Cari

ID	NPM	Nama	Semester	IPK
1	50421786	Marsillo	7	3.75
2	50421787	Dito	8	3.0
3	50421788	Saputra	9	4.0

## LOGIKA PROGRAM ; Penjelasan Kode **MahasiswaView.java**:

Kode ini adalah bagian dari aplikasi GUI (Graphical User Interface) untuk mengelola data mahasiswa menggunakan Java Swing. Aplikasi ini memiliki berbagai komponen seperti form input untuk data mahasiswa (NPM, Nama, Semester, IPK) dan tombol untuk menyimpan, menghapus, menyegarkan, dan mencari data mahasiswa. Data mahasiswa disimpan dan dikelola melalui database yang terhubung dengan Hibernate.

Penjelasan Fitur Utama:

1. Membuat Koneksi ke Database: Pada konstruktor MahasiswaView(), kode memanggil HibernateUtil.testConnection() untuk menguji koneksi ke database menggunakan Hibernate. Database yang digunakan adalah MySQL atau sistem basis data yang dikonfigurasi.
2. Memuat Data Mahasiswa ke Tabel: Metode loadMahasiswaTable() digunakan untuk mengambil semua data mahasiswa dari controller MahasiswaControllerImpl dan menampilkannya dalam tabel dataTable. Model tabel ini dibuat melalui ModelTableMahasiswa.
3. Form Input Data Mahasiswa: Terdapat empat JTextField untuk memasukkan data mahasiswa: NPM, Nama, Semester, dan IPK. Pengguna bisa memasukkan informasi ini dan mengklik tombol Simpan untuk menambah data mahasiswa ke database.
4. Tombol Simpan: Ketika tombol "Simpan" diklik, data yang dimasukkan akan dikumpulkan dan kemudian disimpan ke dalam database menggunakan metode controller.addMhs(mahasiswa).
5. Tombol Hapus: Tombol "Buang" memungkinkan pengguna untuk menghapus data mahasiswa dengan memasukkan ID mahasiswa yang akan dihapus. Program kemudian mengonfirmasi ID yang dimasukkan dan menghapus entri dari database menggunakan controller.deleteMhs(id).
6. Tombol Refresh: Tombol "Refresh" digunakan untuk memuat ulang data dari database dan menampilkan kembali daftar mahasiswa di tabel.
7. Fungsi Pencarian Mahasiswa: Tombol "Cari" memunculkan dialog pencarian mahasiswa. Pengguna dapat memasukkan ID mahasiswa, kemudian aplikasi mencari data mahasiswa tersebut di database. Jika ditemukan, aplikasi akan menampilkan detail mahasiswa, jika tidak, akan menampilkan pesan error.
8. Dialog Pencarian Mahasiswa: Dialog ini muncul ketika pengguna mengklik tombol "Cari". Pengguna diminta untuk memasukkan ID mahasiswa dan setelah tombol "OK" ditekan, data mahasiswa tersebut dicari menggunakan query SQL yang terhubung dengan database.