

## **LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : Rekayasa Perangkat Lunak 2  
Kelas : 4IA06  
Praktikum ke- : 3  
Tanggal : 29 Oktober 2024  
Materi : Konsep Model-View-Controller (MVC), Pembuatan Program dengan Konsep MVC, Instalasi & Penerapan, dan Akses Database dengan JDBC  
NPM : 51421517  
Nama : William Devin Septianus Pranggono  
Ketua Asisten :  
Paraf Asisten :  
Nama Asisten : Gilbert Jefferson Faozato Mendrofa  
Jumlah Lembar : 25 lembar

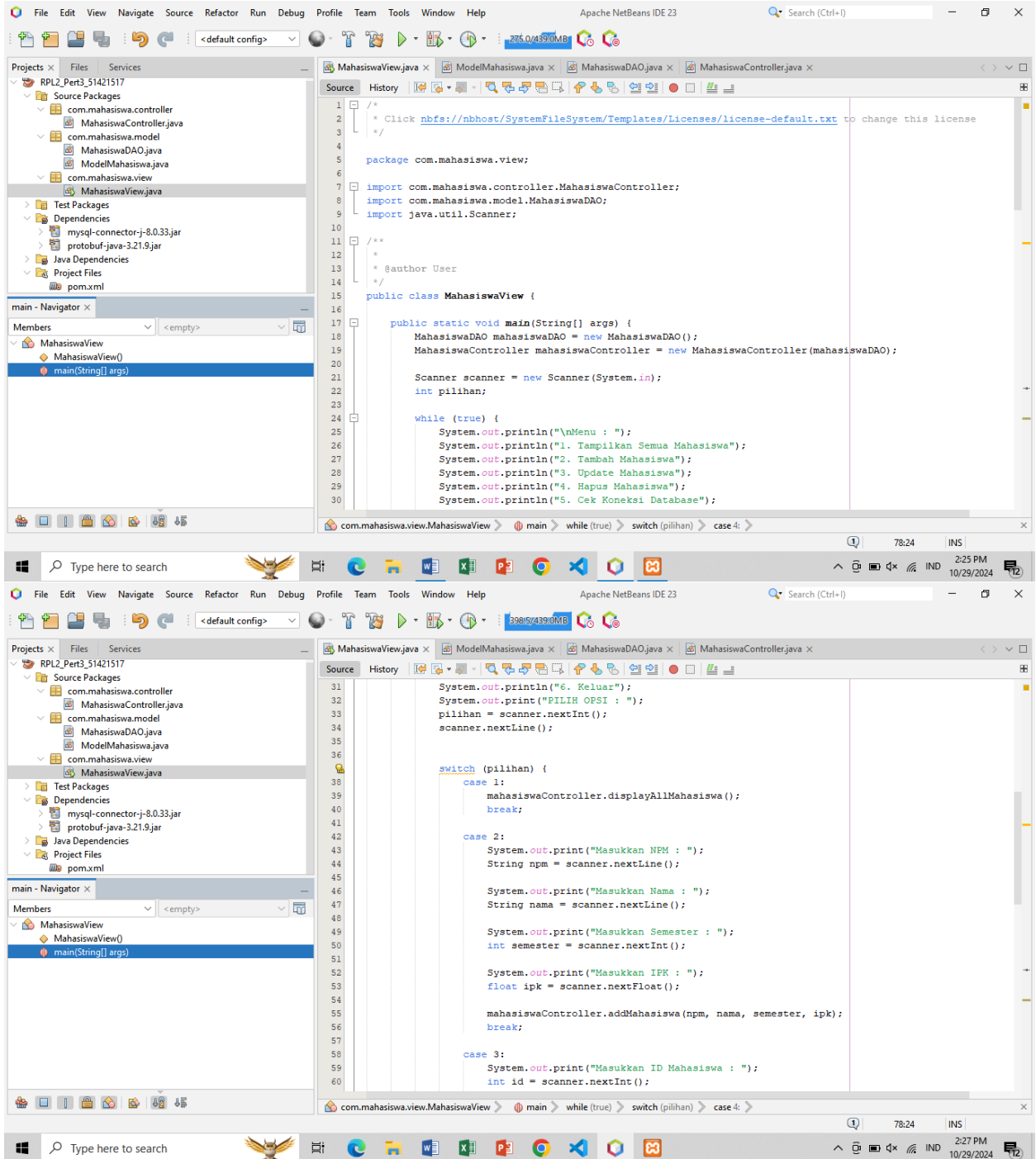
**LABORATORIUM TEKNIK INFORMATIKA**

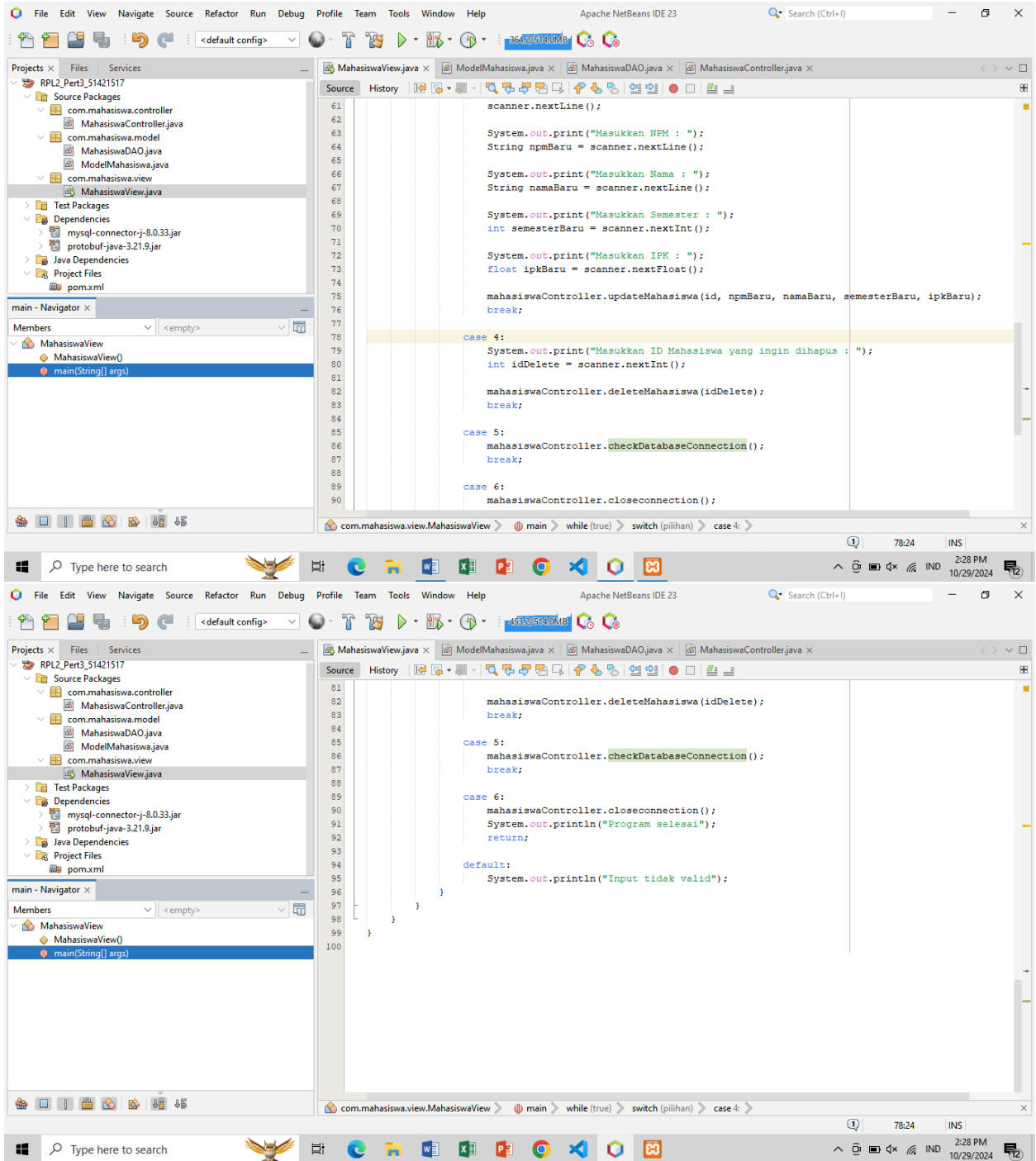
**UNIVERSITAS GUNADARMA**

**2024**

# LISTING PROGRAM

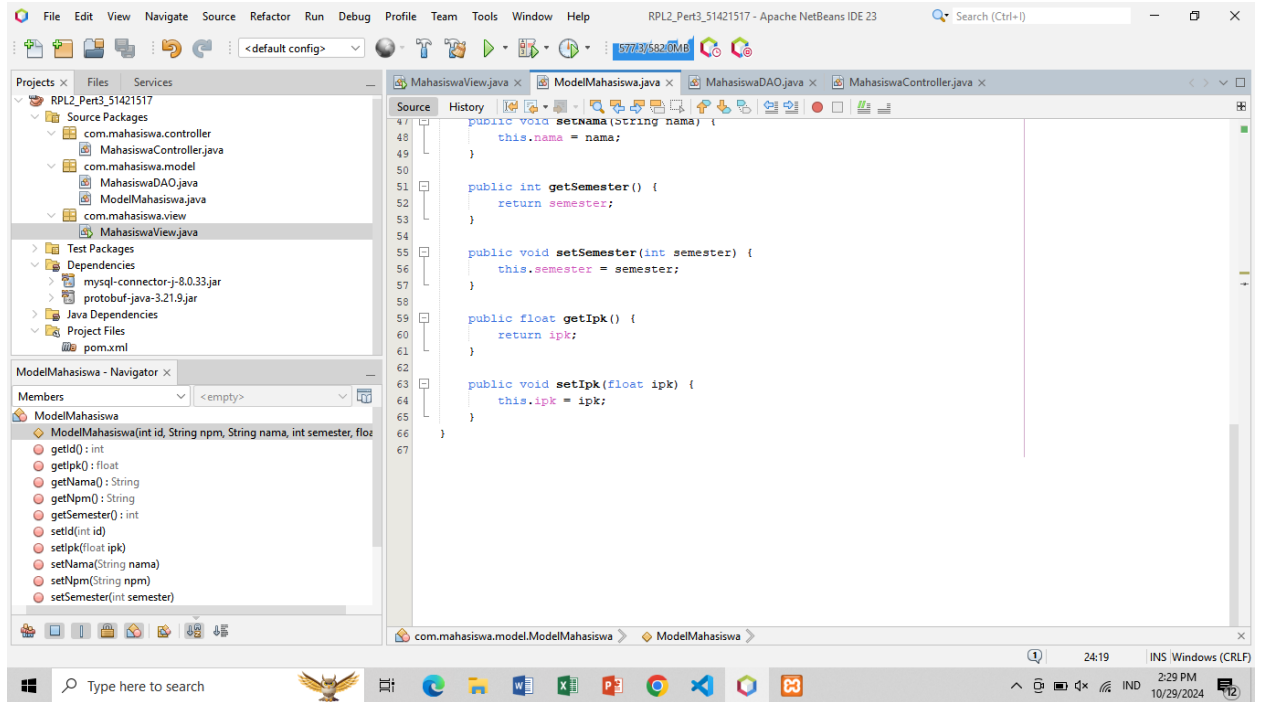
## 1. Mahasiswa.java



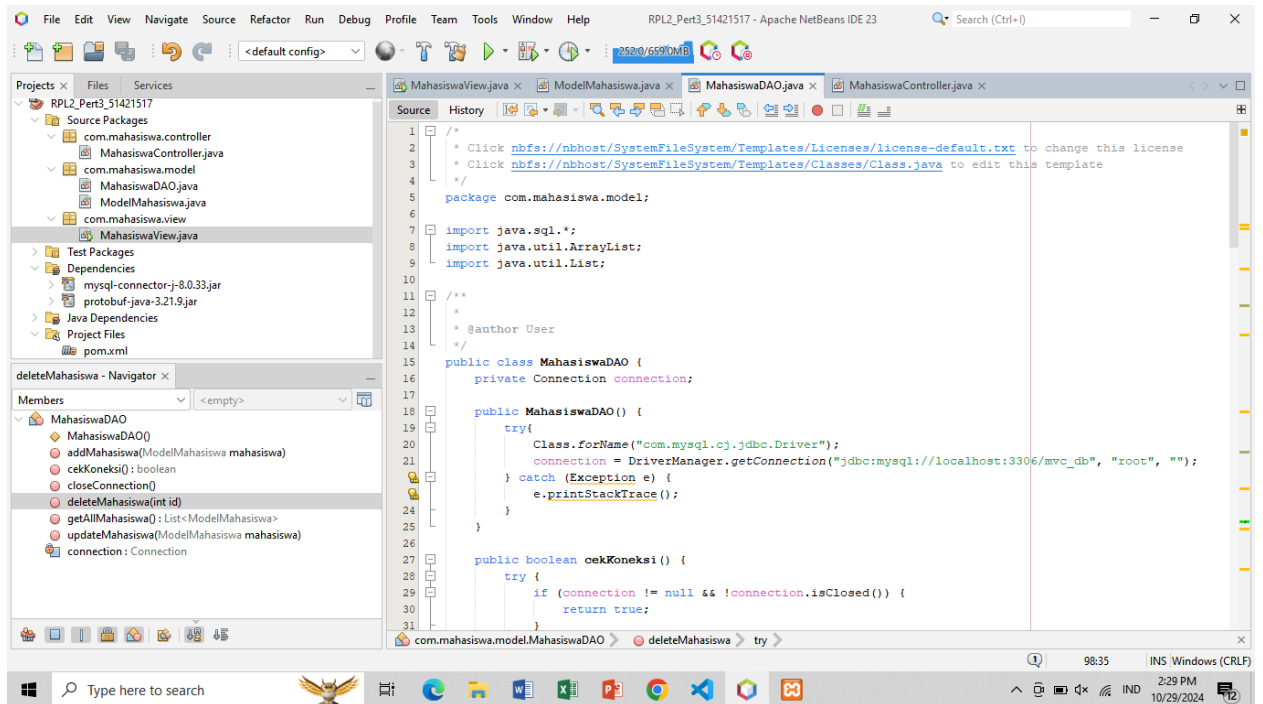


## 2. ModelMahasiswa.java





### 3. MahasiswaDAO.java



File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2\_Pert3\_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)

Projects Files Services

- RPL2\_Pert3\_51421517
  - Source Packages
    - com.mahasiswa.controller
    - com.mahasiswa.model
    - com.mahasiswa.view
  - Test Packages
  - Dependencies
    - mysql-connector-j-8.0.33.jar
    - protobuf-java-3.21.9.jar
  - Java Dependencies
  - Project Files
    - pom.xml

deleteMahasiswa - Navigator

Members

- MahasiswaDAO
  - MahasiswaDAO0
  - addMahasiswa(ModelMahasiswa mahasiswa)
  - cekKoneksi(): boolean
  - closeConnection()
  - deleteMahasiswa(int id)
  - getAllMahasiswa(): List<ModelMahasiswa>
  - updateMahasiswa(ModelMahasiswa mahasiswa)
  - connection: Connection

Source History

```
31 }
32 } catch (SQLException e) {
33     e.printStackTrace();
34 }
35 return false;
36 }
37
38 public void addMahasiswa(ModelMahasiswa mahasiswa) {
39     String sql = "INSERT INTO mahasiswa (npm, nama, semester, ipk)"
40         + "VALUES (?, ?, ?, ?)";
41     try {
42         PreparedStatement pstmt = connection.prepareStatement(sql);
43         pstmt.setString(1, mahasiswa.getNpm());
44         pstmt.setString(2, mahasiswa.getNama());
45         pstmt.setInt(3, mahasiswa.getSemester());
46         pstmt.setFloat(4, mahasiswa.getIpk());
47
48         pstmt.executeUpdate();
49     } catch (SQLException e) {
50         e.printStackTrace();
51     }
52 }
53
54 public List<ModelMahasiswa> getAllMahasiswa() {
55     List<ModelMahasiswa> mahasiswaList = new ArrayList<>();
56     String sql = "SELECT * FROM mahasiswa";
57     try {
58         Statement stmt = connection.createStatement();
59         ResultSet rs = stmt.executeQuery(sql);
60         while (rs.next()) {
61             mahasiswaList.add(new ModelMahasiswa(
62                 rs.getInt("id"),
63                 rs.getString("npm"),
64                 rs.getString("nama"),
65                 rs.getInt("semester"),
66                 rs.getFloat("ipk")
67             ));
68         }
69     } catch (SQLException e) {
70         e.printStackTrace();
71     }
72     return mahasiswaList;
73 }
74
75 public void updateMahasiswa(ModelMahasiswa mahasiswa) {
76     String sql = "UPDATE mahasiswa SET npm = ?, nama = ?,
77         + "semester = ?, ipk = ? WHERE id = ?";
78     try {
79         PreparedStatement pstmt = connection.prepareStatement(sql);
80         pstmt.setString(1, mahasiswa.getNpm());
81         pstmt.setString(2, mahasiswa.getNama());
82         pstmt.setInt(3, mahasiswa.getSemester());
83         pstmt.setFloat(4, mahasiswa.getIpk());
84         pstmt.setInt(5, mahasiswa.getId());
85
86         pstmt.executeUpdate();
87     } catch (SQLException e) {
88         e.printStackTrace();
89     }
90 }
91 }
```

com.mahasiswa.model.MahasiswaDAO > deleteMahasiswa > try >

98:35 INS Windows (CRLF) 2:30 PM 10/29/2024

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2\_Pert3\_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)

Projects Files Services

- RPL2\_Pert3\_51421517
  - Source Packages
    - com.mahasiswa.controller
    - com.mahasiswa.model
    - com.mahasiswa.view
  - Test Packages
  - Dependencies
    - mysql-connector-j-8.0.33.jar
    - protobuf-java-3.21.9.jar
  - Java Dependencies
  - Project Files
    - pom.xml

deleteMahasiswa - Navigator

Members

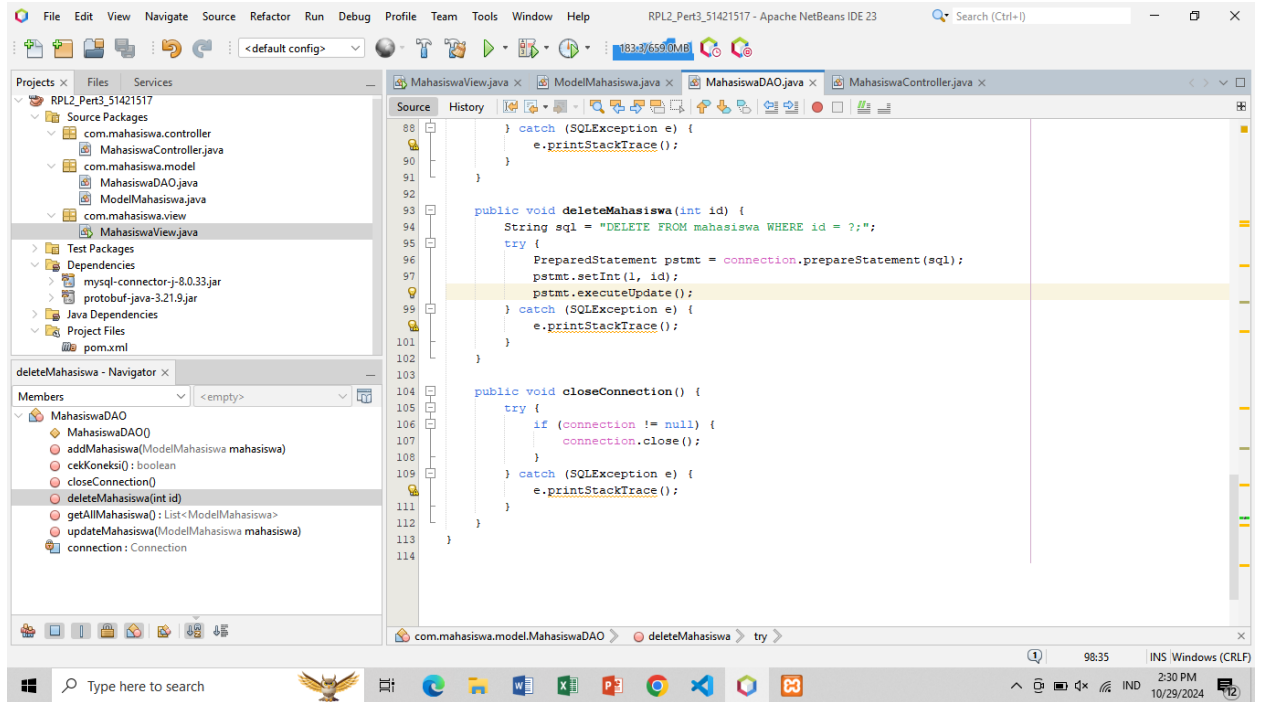
- MahasiswaDAO
  - MahasiswaDAO0
  - addMahasiswa(ModelMahasiswa mahasiswa)
  - cekKoneksi(): boolean
  - closeConnection()
  - deleteMahasiswa(int id)
  - getAllMahasiswa(): List<ModelMahasiswa>
  - updateMahasiswa(ModelMahasiswa mahasiswa)
  - connection: Connection

Source History

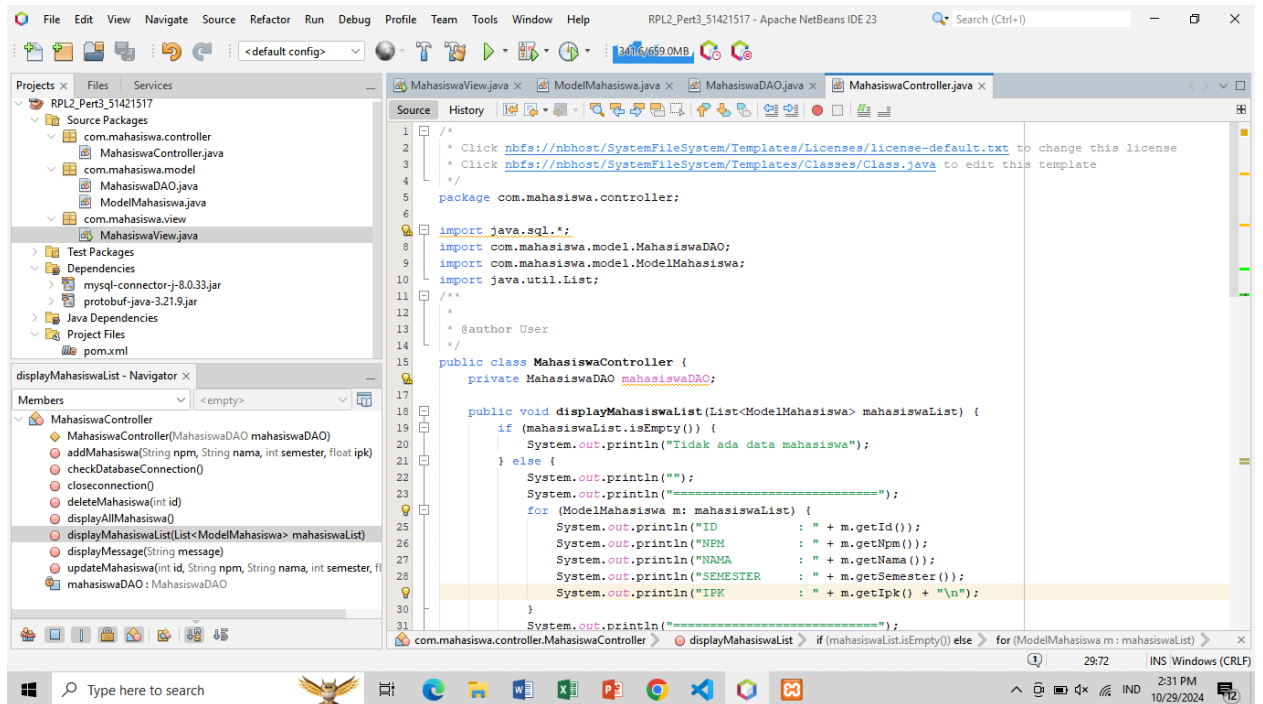
```
61 mahasiswaList.add(new ModelMahasiswa(
62     rs.getInt("id"),
63     rs.getString("npm"),
64     rs.getString("nama"),
65     rs.getInt("semester"),
66     rs.getFloat("ipk")
67 ));
68 }
69 } catch (SQLException e) {
70     e.printStackTrace();
71 }
72 return mahasiswaList;
73 }
74
75 public void updateMahasiswa(ModelMahasiswa mahasiswa) {
76     String sql = "UPDATE mahasiswa SET npm = ?, nama = ?,
77         + "semester = ?, ipk = ? WHERE id = ?";
78     try {
79         PreparedStatement pstmt = connection.prepareStatement(sql);
80         pstmt.setString(1, mahasiswa.getNpm());
81         pstmt.setString(2, mahasiswa.getNama());
82         pstmt.setInt(3, mahasiswa.getSemester());
83         pstmt.setFloat(4, mahasiswa.getIpk());
84         pstmt.setInt(5, mahasiswa.getId());
85
86         pstmt.executeUpdate();
87     } catch (SQLException e) {
88         e.printStackTrace();
89     }
90 }
91 }
```

com.mahasiswa.model.MahasiswaDAO > deleteMahasiswa > try >

98:35 INS Windows (CRLF) 2:30 PM 10/29/2024



#### 4. MahasiswaController.java



File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2\_Pert3\_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)

Projects Files Services

- RPL2\_Pert3\_51421517
  - Source Packages
    - com.mahasiswa.controller
      - MahasiswaController.java
    - com.mahasiswa.model
      - MahasiswaDAO.java
      - ModelMahasiswa.java
    - com.mahasiswa.view
      - MahasiswaView.java
    - Test Packages
    - Dependencies
      - mysql-connector-j-8.0.33.jar
      - protobuf-java-3.21.9.jar
    - Java Dependencies
    - Project Files
      - pom.xml

displayMahasiswaList - Navigator

Members

- MahasiswaController
  - MahasiswaController(MahasiswaDAO mahasiswaDAO)
  - addMahasiswa(String npm, String nama, int semester, float ipk)
  - checkDatabaseConnection()
  - closeConnection()
  - deleteMahasiswa(int id)
  - displayAllMahasiswa()
  - displayMahasiswaList(List<ModelMahasiswa> mahasiswaList)
  - displayMessage(String message)
  - updateMahasiswa(int id, String npm, String nama, int semester, float ipk)
  - mahasiswaDAO : MahasiswaDAO

Source History

```
31 System.out.println("=====");
32 }
33 displayMessage("Mahasiswa berhasil ditampilkan");
34 }
35
36 public void displayMessage(String message) {
37     System.out.println(message);
38 }
39
40 public MahasiswaController(MahasiswaDAO mahasiswaDAO) {
41     this.mahasiswaDAO = mahasiswaDAO;
42 }
43
44 public void checkDatabaseConnection() {
45     boolean isConnected = mahasiswaDAO.cekKoneksi();
46     if (isConnected) {
47         displayMessage("Koneksi ke database berhasil");
48     } else {
49         displayMessage("Koneksi ke database gagal");
50     }
51 }
52
53 public void displayAllMahasiswa() {
54     List<ModelMahasiswa> mahasiswaList = mahasiswaDAO.getAllMahasiswa();
55     displayMahasiswaList(mahasiswaList);
56 }
57
58 public void addMahasiswa(String npm, String nama, int semester, float ipk) {
59     ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm, nama, semester, ipk);
60     mahasiswaDAO.addMahasiswa(mahasiswaBaru);
61     displayMessage("Mahasiswa berhasil ditambahkan");
62 }
```

com.mahasiswa.controller.MahasiswaController > displayMahasiswaList > if (mahasiswaList.isEmpty()) else > for (ModelMahasiswa m : mahasiswaList) >

29.72 INS Windows (CRLF) 2:31 PM 10/29/2024

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2\_Pert3\_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)

Projects Files Services

- RPL2\_Pert3\_51421517
  - Source Packages
    - com.mahasiswa.controller
      - MahasiswaController.java
    - com.mahasiswa.model
      - MahasiswaDAO.java
      - ModelMahasiswa.java
    - com.mahasiswa.view
      - MahasiswaView.java
    - Test Packages
    - Dependencies
      - mysql-connector-j-8.0.33.jar
      - protobuf-java-3.21.9.jar
    - Java Dependencies
    - Project Files
      - pom.xml

displayMahasiswaList - Navigator

Members

- MahasiswaController
  - MahasiswaController(MahasiswaDAO mahasiswaDAO)
  - addMahasiswa(String npm, String nama, int semester, float ipk)
  - checkDatabaseConnection()
  - closeConnection()
  - deleteMahasiswa(int id)
  - displayAllMahasiswa()
  - displayMahasiswaList(List<ModelMahasiswa> mahasiswaList)
  - displayMessage(String message)
  - updateMahasiswa(int id, String npm, String nama, int semester, float ipk)
  - mahasiswaDAO : MahasiswaDAO

Source History

```
56
57
58 public void addMahasiswa(String npm, String nama, int semester, float ipk) {
59     ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm, nama, semester, ipk);
60     mahasiswaDAO.addMahasiswa(mahasiswaBaru);
61     displayMessage("Mahasiswa berhasil ditambahkan");
62 }
63
64 public void updateMahasiswa(int id, String npm, String nama, int semester, float ipk) {
65     ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(id, npm, nama, semester, ipk);
66     mahasiswaDAO.updateMahasiswa(mahasiswaBaru);
67     displayMessage("Mahasiswa berhasil diperbaharui");
68 }
69
70 public void deleteMahasiswa(int id) {
71     mahasiswaDAO.deleteMahasiswa(id);
72     displayMessage("Mahasiswa berhasil dihapus");
73 }
74
75 public void closeConnection() {
76     mahasiswaDAO.closeConnection();
77 }
78
79 }
```

com.mahasiswa.controller.MahasiswaController > displayMahasiswaList > if (mahasiswaList.isEmpty()) else > for (ModelMahasiswa m : mahasiswaList) >

29.72 INS Windows (CRLF) 2:31 PM 10/29/2024



## LOGIKA PROGRAM

### 1. ModelMahasiswa.java

Langkah pertama adalah mendefinisikan atribut dari *class* ModelMahasiswa. Atribut memiliki *identifier private*, sehingga memerlukan *method getter* dan *setter*.

```
public class ModelMahasiswa {  
    private int id;  
    private String npm;  
    private String nama;  
    private int semester;  
    private float ipk;
```

Kemudian, di bawah ini adalah kode untuk *constructor* ketika objek dari *class* diinstansiasi.

```
    public ModelMahasiswa(int id, String npm, String nama, int  
semester, float ipk) {  
        this.id = id;  
        this.npm = npm;  
        this.nama = nama;  
        this.semester = semester;  
        this.ipk = ipk;  
    }  
}
```

Terakhir, berikut adalah *method getter* dan *setter* untuk memodifikasi dan mengakses atribut dari *class*.

```
    public int getId() {  
        return id;  
    }  
  
    public void setId(int id) {  
        this.id = id;
```

```
}

public String getNpm() {
    return npm;
}

public void setNpm(String npm) {
    this.npm = npm;
}

public String getNama() {
    return nama;
}

public void setNama(String nama) {
    this.nama = nama;
}

public int getSemester() {
    return semester;
}

public void setSemester(int semester) {
    this.semester = semester;
}

public float getIpk() {
    return ipk;
}
```

```

    }

    public void setIpk(float ipk) {
        this.ipk = ipk;
    }
}

```

## 2. MahasiswaDAO.java

MahasiswaDAO berperan sebagai *Data Access Object* yang menyediakan akses terhadap *database* dan sebagai perantara antara *controller* dan *database*. Class MahasiswaDAO memiliki atribut *connection*, yaitu objek *Connection* dari package *java.sql.\**.

```

public class MahasiswaDAO {
    private Connection connection;
}

```

Kemudian, *constructor* untuk class MahasiswaDAO didefinisikan tanpa menerima parameter apa pun. Ketika class MahasiswaDAO diinstansiasi, maka akan memuat Driver JDBC yang memungkinkan Java untuk terhubung ke *database*. Kemudian, atribut *connection* adalah objek *Connection* yang akan terhubung dengan *database* sesuai URL yang diberikan.

```

public MahasiswaDAO() {
    try{
        Class.forName("com.mysql.cj.jdbc.Driver");
        connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mvc_db",
"root", "");
    } catch (Exception e) {
        e.printStackTrace();
    }
}

```

Method cekKoneksi akan mengecek apakah *connection* memiliki nilai (berhasil terhubung ke *database*) dan apakah *connection* tidak tertutup, dan akan mengembalikan nilai *true* jika kedua syarat tersebut terpenuhi, dan *false* untuk sebaliknya.

```

public boolean cekKoneksi() {
    try {
        if (connection != null && !connection.isClosed()) {
            return true;
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return false;
}

```

*Method* addMahasiswa menerima objek ModelMahasiswa dan akan menambahkan data ke tabel mahasiswa dengan atribut yang dimiliki objek ModelMahasiswa.

```

public void addMahasiswa(ModelMahasiswa mahasiswa) {
    String sql = "INSERT INTO mahasiswa (npm, nama, semester,
ipk)"
        + "VALUES (?, ?, ?, ?);";
    try {
        PreparedStatement pstmt =
connection.prepareStatement(sql);
        pstmt.setString(1, mahasiswa.getNpm());
        pstmt.setString(2, mahasiswa.getNama());
        pstmt.setInt(3, mahasiswa.getSemester());
        pstmt.setFloat(4, mahasiswa.getIpk());

        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```
}
```

Kemudian, *method* `getAllMahasiswa` akan mengambil seluruh data dari tabel mahasiswa pada *database* dan akan mengembalikan List bertipe data `ModelMahasiswa`.

```
public List<ModelMahasiswa> getAllMahasiswa() {  
    List<ModelMahasiswa> mahasiswaList = new ArrayList<>();  
    String sql = "SELECT * FROM mahasiswa;";  
    try {  
        Statement stmt = connection.createStatement();  
        ResultSet rs = stmt.executeQuery(sql);  
        while (rs.next()){  
            mahasiswaList.add(new ModelMahasiswa(  
                rs.getInt("id"),  
                rs.getString("npm"),  
                rs.getString("nama"),  
                rs.getInt("semester"),  
                rs.getFloat("ipk")  
            ));  
        }  
    } catch (SQLException e) {  
        e.printStackTrace();  
    }  
    return mahasiswaList;  
}
```

*Method* `updateMahasiswa` akan memperbarui *record* yang telah ada dengan data dari parameter mahasiswa bertipe data `ModelMahasiswa`.

```
public void updateMahasiswa(ModelMahasiswa mahasiswa) {
```

```

        String sql = "UPDATE mahasiswa SET npm = ?, nama = ?,"
            + "semester = ?, ipk = ? WHERE id = ?;";

        try {
            PreparedStatement pstmt =
connection.prepareStatement(sql);

            pstmt.setString(1, mahasiswa.getNpm());
            pstmt.setString(2, mahasiswa.getNama());
            pstmt.setInt(3, mahasiswa.getSemester());
            pstmt.setFloat(4, mahasiswa.getIpk());
            pstmt.setInt(5, mahasiswa.getId());

            pstmt.executeUpdate();

        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

*Method* deleteMahasiswa akan menghapus *record* dari tabel mahasiswa berdasarkan parameter *id* yang diberikan.

```

public void deleteMahasiswa(int id) {
    String sql = "DELETE FROM mahasiswa WHERE id = ?;";
    try {
        PreparedStatement pstmt =
connection.prepareStatement(sql);

        pstmt.setInt(1, id);

        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

```

```
}  
}
```

*Method* `closeConnection` akan menutup koneksi dengan *database* apabila variabel *connection* tidak bernilai *null*.

```
public void closeConnection() {  
    try {  
        if (connection != null) {  
            connection.close();  
        }  
    } catch (SQLException e) {  
        e.printStackTrace();  
    }  
}  
}
```

### 3. MahasiswaController.java

*Class* `MahasiswaController` berperan sebagai *Controller*, yang merupakan perantara *Model* dengan *View*. Atribut yang dimiliki adalah objek `MahasiswaDAO`.

```
public class MahasiswaController {  
    private MahasiswaDAO mahasiswaDAO;
```

*Method* `displayMahasiswaList` berfungsi untuk menampilkan seluruh data dari tabel mahasiswa. Data tersebut merupakan parameter `mahasiswaList` bertipe data `List ModelMahasiswa`.

```
public void displayMahasiswaList(List<ModelMahasiswa>  
mahasiswaList) {  
    if (mahasiswaList.isEmpty()) {  
        System.out.println("Tidak ada data mahasiswa");  
    } else {
```

```

        System.out.println("");
        System.out.println("=====");
        for (ModelMahasiswa m: mahasiswaList) {
            System.out.println("ID            : " + m.getId());
            System.out.println("NPM            : " + m.getNpm());
            System.out.println("NAMA            : " + m.getNama());
            System.out.println("SEMESTER        : " +
m.getSemester());
            System.out.println("IPK            : " + m.getIpk() +
"\n");
        }
        System.out.println("=====");
    }
    displayMessage("Mahasiswa berhasil ditampilkan");
}

```

*Method* displayMessage berfungsi untuk menampilkan String message yang merupakan parameter dari *method* ke layar CLI.

```

public void displayMessage(String message) {
    System.out.println(message);
}

```

*Constructor* MahasiswaController menginisiasi atribut mahasiswaDAO dengan parameter bertipe data MahasiswaDAO.

```

public MahasiswaController(MahasiswaDAO mahasiswaDAO) {
    this.mahasiswaDAO = mahasiswaDAO;
}

```



*Method* `checkDatabaseConnection` berfungsi memeriksa status koneksi *database* dengan memanggil *method* `cekKoneksi` milik objek `mahasiswaDAO`, dan akan menampilkan pesan status ke layar CLI.

```
public void checkDatabaseConnection() {  
    boolean isConnected = mahasiswaDAO.cekKoneksi();  
    if (isConnected) {  
        displayMessage("Koneksi ke database berhasil");  
    } else {  
        displayMessage("Koneksi ke database gagal");  
    }  
}
```

*Method* `displayAllMahasiswa` berfungsi untuk menampilkan seluruh data mahasiswa dengan memanggil *method* `getAllMahasiswa` milik objek `mahasiswaDAO` untuk mendapatkan data, dan `displayMahasiswaList` untuk menampilkan data.

```
public void displayAllMahasiswa() {  
    List<ModelMahasiswa> mahasiswaList =  
mahasiswaDAO.getAllMahasiswa();  
    displayMahasiswaList(mahasiswaList);  
}
```

*Method* `addMahasiswa` berfungsi untuk menambahkan *record* ke tabel mahasiswa menggunakan data yang diberikan dari parameter *method*. Penambahan *record* dilakukan dengan memanggil *method* `addMahasiswa` milik objek `mahasiswaDAO`.

```
public void addMahasiswa(String npm, String nama, int semester,  
float ipk) {  
    ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm,  
nama, semester, ipk);  
    mahasiswaDAO.addMahasiswa(mahasiswaBaru);  
    displayMessage("Mahasiswa berhasil ditambahkan");  
}
```

```
}
```

*Method* `updateMahasiswa` berfungsi memperbarui data dengan nilai yang diberikan pada parameter. Pembaruan data dilakukan dengan memanggil *method* `updateMahasiswa` milik objek `mahasiswaDAO`.

```
public void updateMahasiswa(int id, String npm, String nama, int semester, float ipk) {  
    ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(id, npm, nama, semester, ipk);  
    mahasiswaDAO.updateMahasiswa(mahasiswaBaru);  
    displayMessage("Mahasiswa berhasil diperbaharui");  
}
```

*Method* `deleteMahasiswa` berfungsi menghapus *record* dari tabel mahasiswa berdasarkan *id* yang diberikan pada parameter.

```
public void deleteMahasiswa(int id) {  
    mahasiswaDAO.deleteMahasiswa(id);  
    displayMessage("Mahasiswa berhasil dihapus");  
}
```

*Method* `closeConnection` berfungsi menutup koneksi ke *database* dengan memanggil *method* `closeConnection` milik objek `mahasiswaDAO`.

```
public void closeconnection() {  
    mahasiswaDAO.closeConnection();  
}  
}
```

#### 4. MahasiswaView.java

*Class* ini adalah *class* yang memuat *method main* sebagai *method* utama, dan sekaligus berperan sebagai *View*, yang akan menampilkan *prompt* dan data ke layar CLI.

```
public class MahasiswaView {

    public static void main(String[] args) {
        MahasiswaDAO mahasiswaDAO = new MahasiswaDAO();
        MahasiswaController mahasiswaController = new
MahasiswaController(mahasiswaDAO);

        Scanner scanner = new Scanner(System.in);
        int pilihan;

        while (true) {
            System.out.println("\nMenu : ");
            System.out.println("1. Tampilkan Semua Mahasiswa");
            System.out.println("2. Tambah Mahasiswa");
            System.out.println("3. Update Mahasiswa");
            System.out.println("4. Hapus Mahasiswa");
            System.out.println("5. Cek Koneksi Database");
            System.out.println("6. Keluar");
            System.out.print("PILIH OPSI : ");
            pilihan = scanner.nextInt();
            scanner.nextLine();

            switch (pilihan) {
                case 1:
                    mahasiswaController.displayAllMahasiswa();
```

```
        break;

    case 2:
        System.out.print("Masukkan NPM : ");
        String npm = scanner.nextLine();

        System.out.print("Masukkan Nama : ");
        String nama = scanner.nextLine();

        System.out.print("Masukkan Semester : ");
        int semester = scanner.nextInt();

        System.out.print("Masukkan IPK : ");
        float ipk = scanner.nextFloat();

        mahasiswaController.addMahasiswa(npm, nama,
semester, ipk);
        break;

    case 3:
        System.out.print("Masukkan ID Mahasiswa : ");
        int id = scanner.nextInt();
        scanner.nextLine();

        System.out.print("Masukkan NPM : ");
        String npmBaru = scanner.nextLine();

        System.out.print("Masukkan Nama : ");
```

```
        String namaBaru = scanner.nextLine();

        System.out.print("Masukkan Semester : ");
        int semesterBaru = scanner.nextInt();

        System.out.print("Masukkan IPK : ");
        float ipkBaru = scanner.nextFloat();

        mahasiswaController.updateMahasiswa(id, npmBaru,
namaBaru, semesterBaru, ipkBaru);
        break;

    case 4:
        System.out.print("Masukkan ID Mahasiswa yang ingin
dihapus : ");

        int idDelete = scanner.nextInt();

        mahasiswaController.deleteMahasiswa(idDelete);
        break;

    case 5:
        mahasiswaController.checkDatabaseConnection();
        break;

    case 6:
        mahasiswaController.closeconnection();
        System.out.println("Program selesai");
        return;
```

```
default:
```

```
    System.out.println("Input tidak valid");
```

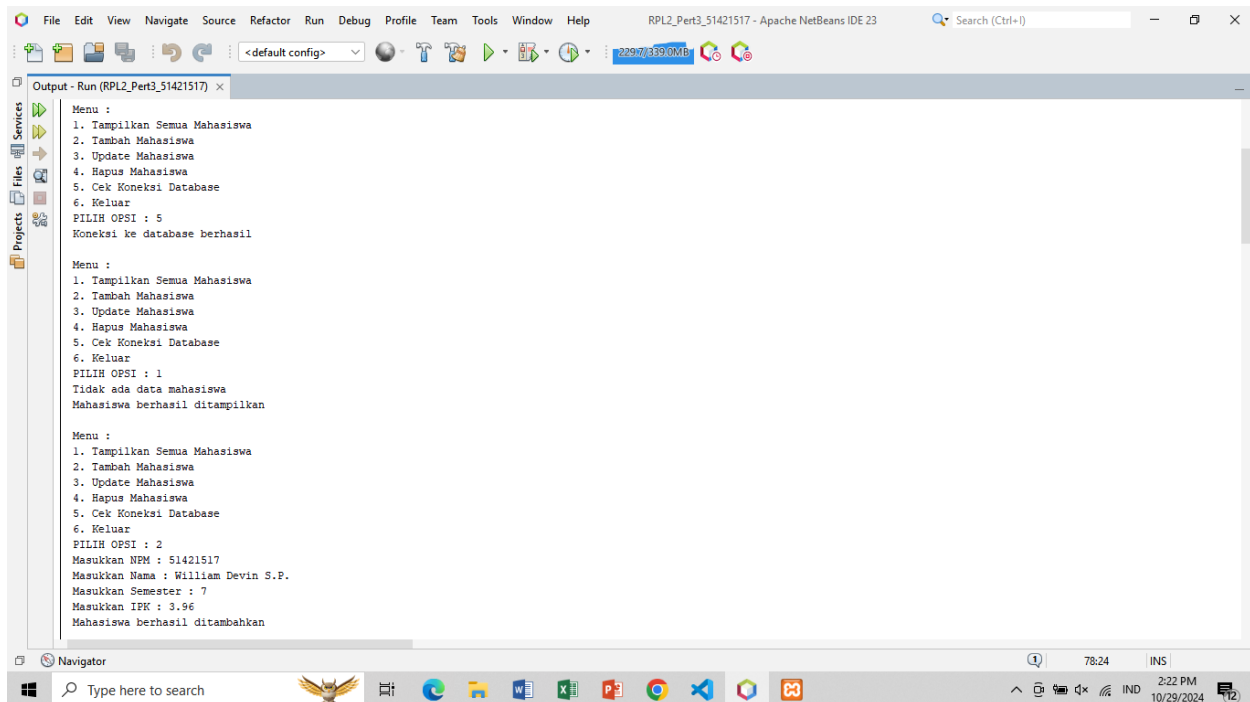
```
    }
```

```
    }
```

```
    }
```

```
}
```

## OUTPUT PROGRAM



The screenshot shows the Apache NetBeans IDE interface. The top menu bar includes File, Edit, View, Navigate, Source, Refactor, Run, Debug, Profile, Team, Tools, Window, and Help. The main window displays the 'Output - Run (RPL2\_Pert3\_51421517)' console. The output text is as follows:

```
Menu :  
1. Tampilkan Semua Mahasiswa  
2. Tambah Mahasiswa  
3. Update Mahasiswa  
4. Hapus Mahasiswa  
5. Cek Koneksi Database  
6. Keluar  
PILIH OPSI : 5  
Koneksi ke database berhasil  
  
Menu :  
1. Tampilkan Semua Mahasiswa  
2. Tambah Mahasiswa  
3. Update Mahasiswa  
4. Hapus Mahasiswa  
5. Cek Koneksi Database  
6. Keluar  
PILIH OPSI : 1  
Tidak ada data mahasiswa  
Mahasiswa berhasil ditampilkan  
  
Menu :  
1. Tampilkan Semua Mahasiswa  
2. Tambah Mahasiswa  
3. Update Mahasiswa  
4. Hapus Mahasiswa  
5. Cek Koneksi Database  
6. Keluar  
PILIH OPSI : 2  
Masukkan NPM : 51421517  
Masukkan Nama : William Devin S.P.  
Masukkan Semester : 7  
Masukkan IPK : 3.96  
Mahasiswa berhasil ditambahkan
```

The bottom status bar shows the time as 7:24 and the location as INS.

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2_Pert3_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)
<default config> 2853/339.0MB
Output - Run (RPL2_Pert3_51421517) x
Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 1

=====
ID      : 2
NPM     : 51421517
NAMA    : William Devin S.P.
SEMESTER : 7
IPK     : 3.96
=====
Mahasiswa berhasil ditampilkan

Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 2
Masukkan NPM : 50422887
Masukkan Nama : Someone
Masukkan Semester : 5
Masukkan IPK : 3.88
Mahasiswa berhasil ditambahkan

Menu :
```

```
File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2_Pert3_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)
<default config> 1984/363.0MB
Output - Run (RPL2_Pert3_51421517) x
Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 1

=====
ID      : 2
NPM     : 51421517
NAMA    : William Devin S.P.
SEMESTER : 7
IPK     : 3.96

ID      : 3
NPM     : 50422887
NAMA    : Someone
SEMESTER : 5
IPK     : 3.88

=====
Mahasiswa berhasil ditampilkan

Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 3
Masukkan ID Mahasiswa : 2
Masukkan NPM : 51421517
```

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2\_Pert3\_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)

Output - Run (RPL2\_Pert3\_51421517) x

```
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 3
Masukkan ID Mahasiswa : 2
Masukkan NPM : 51421517
Masukkan Nama : William Devin S.P.
Masukkan Semester : 7
Masukkan IPK : 3.96
Mahasiswa berhasil diperbaharui

Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 1

=====
ID      : 2
NPM     : 51421517
NAMA    : William Devin S.P.
SEMESTER : 7
IPK     : 3.96

ID      : 3
NPM     : 50422887
NAMA    : Someone
SEMESTER : 5
IPK     : 3.88

=====
Mahasiswa berhasil ditampilkan
```

Navigator 78:24 INS 2:24 PM 10/29/2024

File Edit View Navigate Source Refactor Run Debug Profile Team Tools Window Help RPL2\_Pert3\_51421517 - Apache NetBeans IDE 23 Search (Ctrl+I)

Output - Run (RPL2\_Pert3\_51421517) x

```
Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 4
Masukkan ID Mahasiswa yang ingin dihapus : 3
Mahasiswa berhasil dihapus

Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI : 1

=====
ID      : 2
NPM     : 51421517
NAMA    : William Devin S.P.
SEMESTER : 7
IPK     : 3.96

=====
Mahasiswa berhasil ditampilkan

Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
```

Navigator 78:24 INS 2:24 PM 10/29/2024



