LAPORAN AKHIR PRAKTIKUM

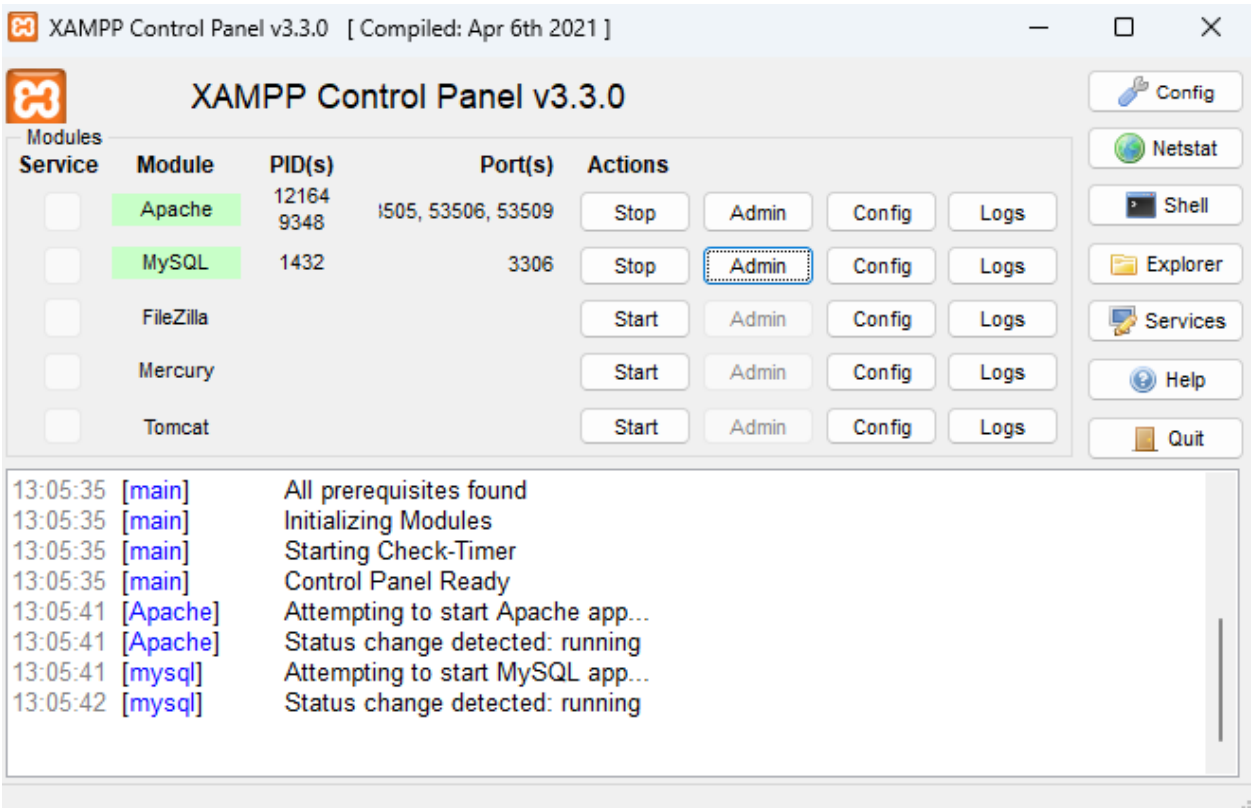
Mata Praktikum	: Rekayasa Perangkat Lunak 2
Kelas	: 4IA06
Praktikum ke-	: 3
Tanggal	: 29 Oktober 2024
Materi	: Konsep Model – View – Controller (MVC)
NPM	: 51421055
Nama	: Muhammad Ridho
Ketua Asisten	: Gilbert Jefferson Faozato Mendrofa
Paraf Asisten	:
Nama Asisten	:
Jumlah Lembar	: 22 Lembar

LABORATORIUM TEKNIK INFORMATIKA
UNIVERSITAS GUNADARMA
2024

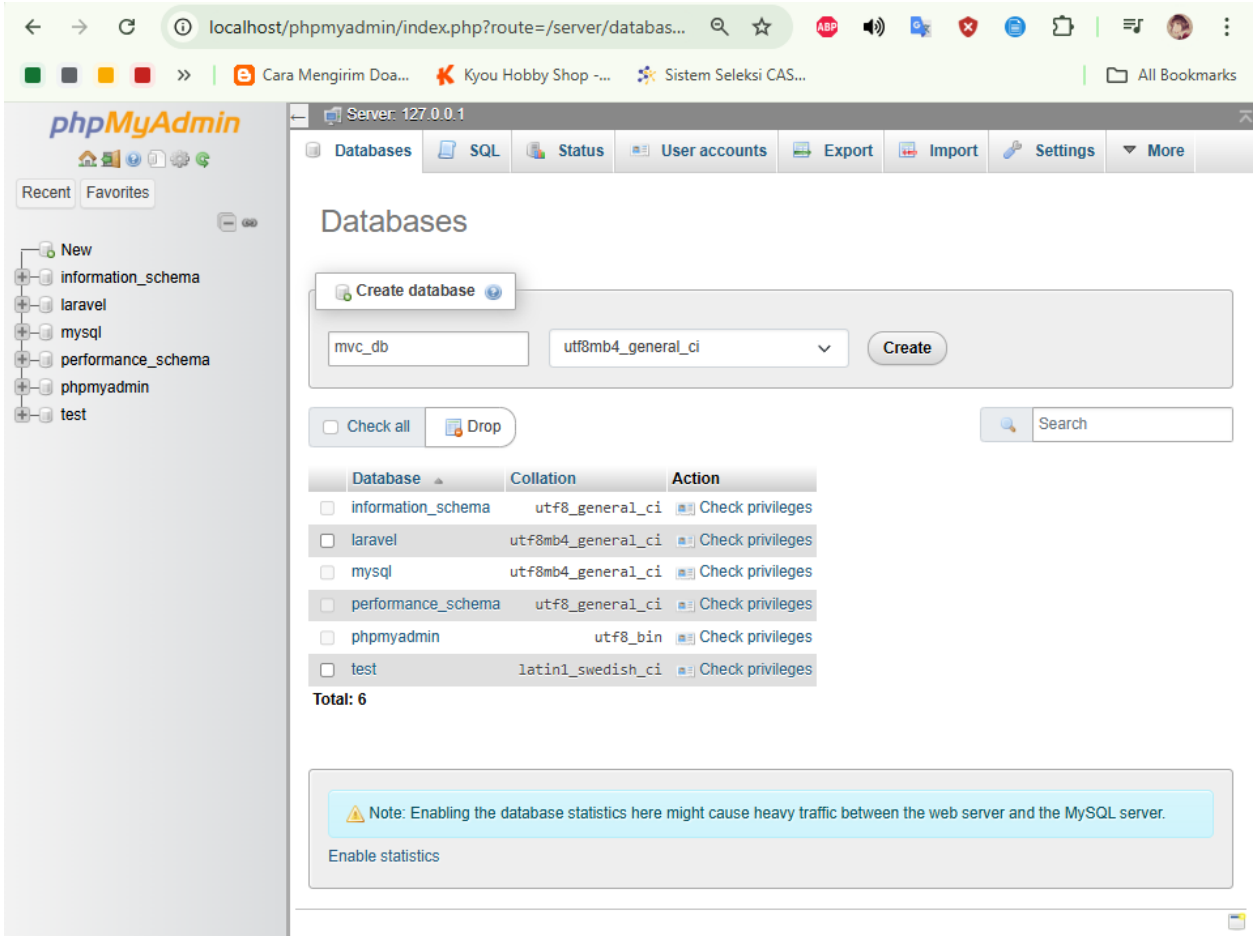
Soal:

1. Jelaskan satu per satu Codingan kalian dari hasil screenshoot activity.

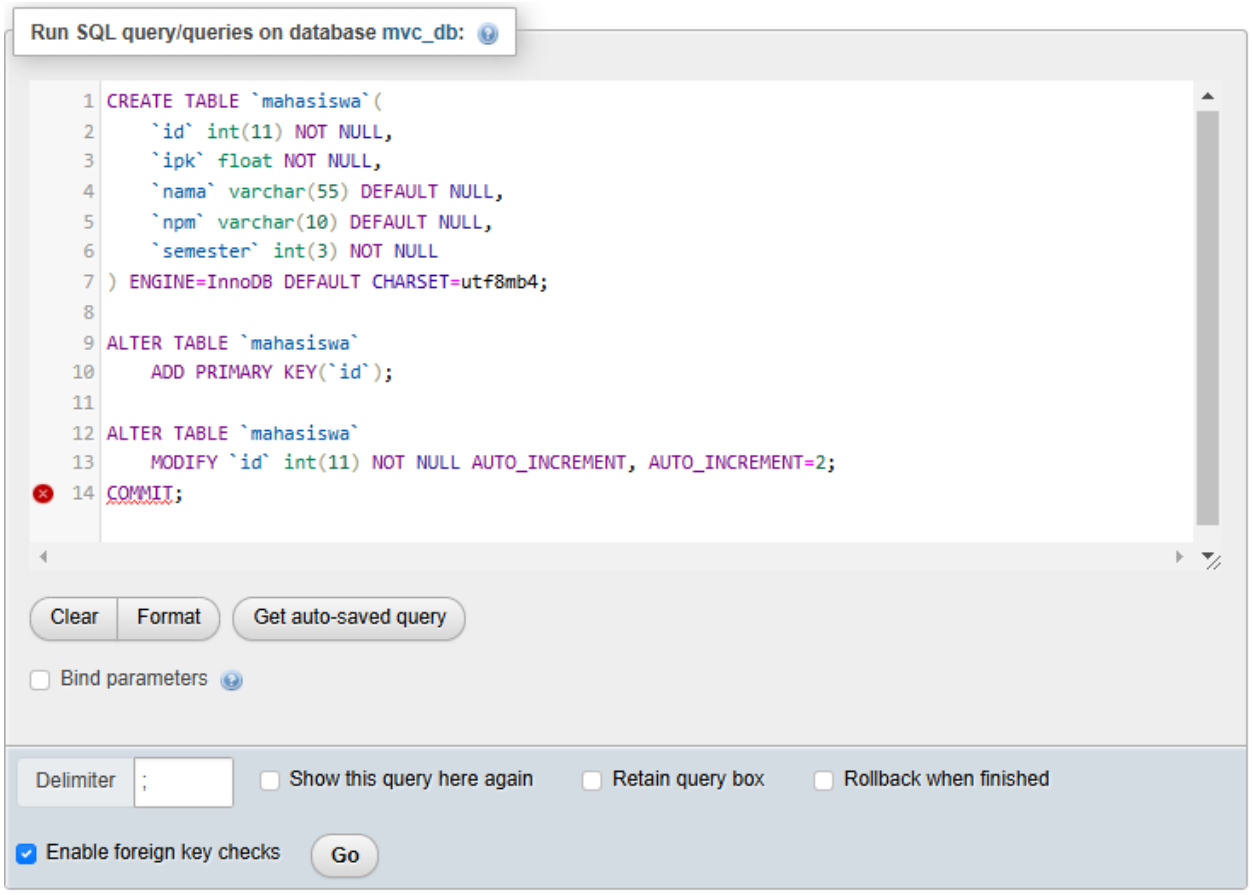
Jawaban:



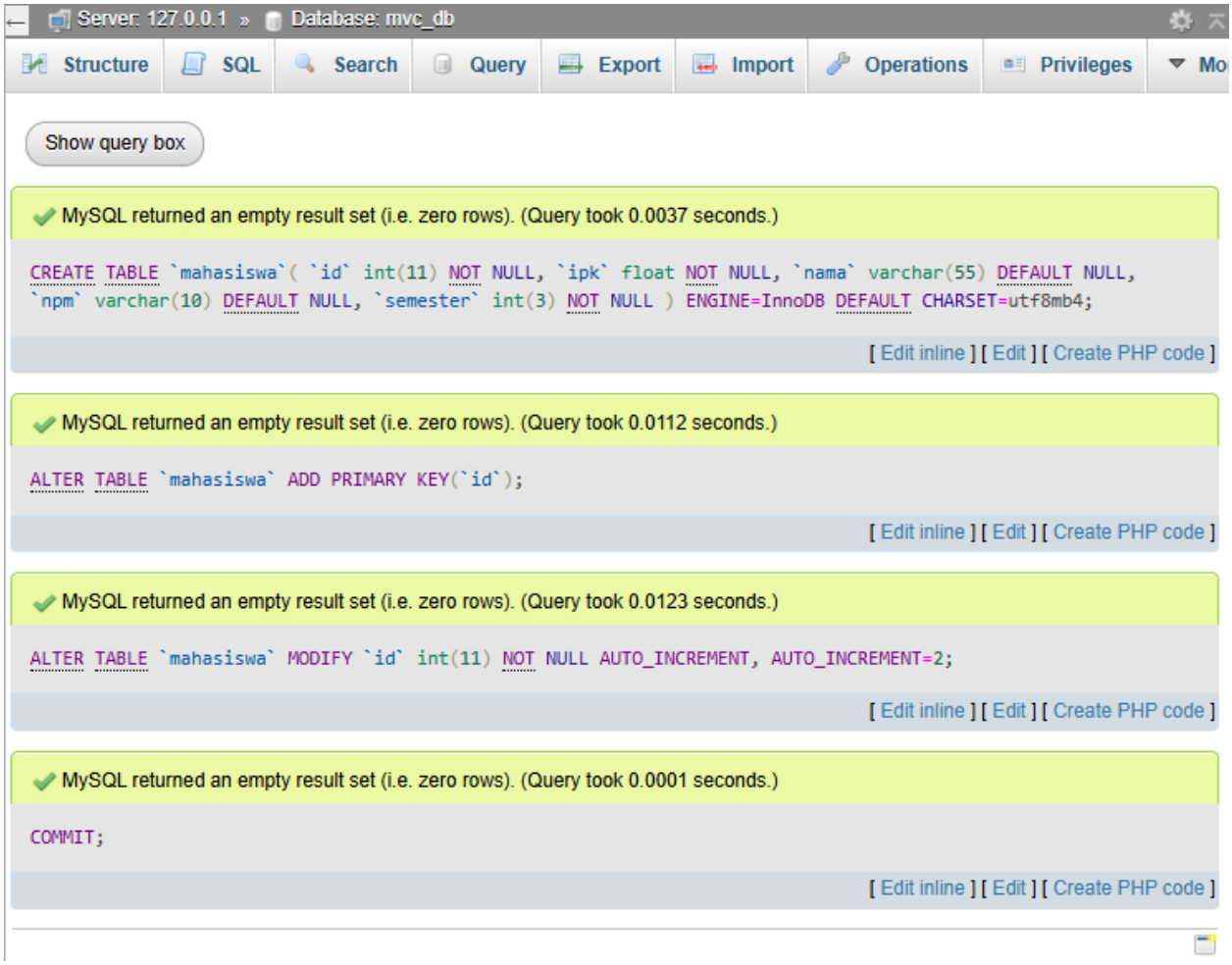
Penjelasan: Pertama kita start apache dan mySQL



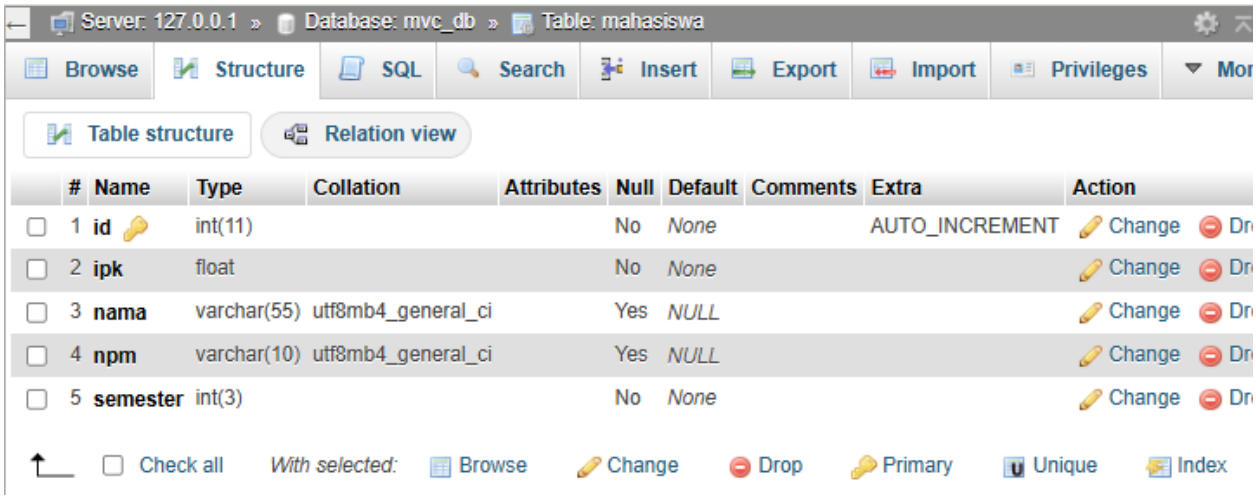
Penjelasan: Lalu kita menuju database dan membuatnya dengan nama MVC_DB

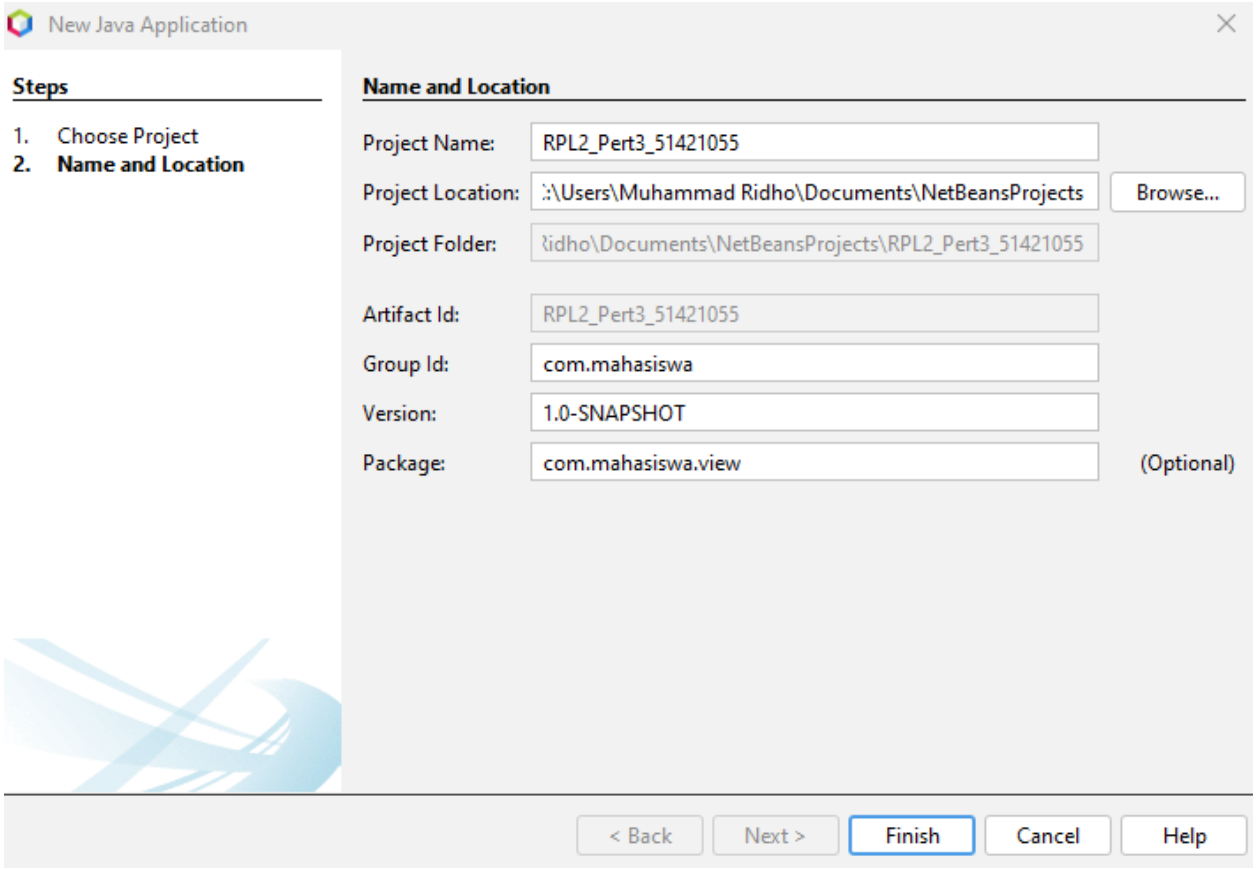


Penjelasan: Kemudian kita masukan query yang telah diberikan.



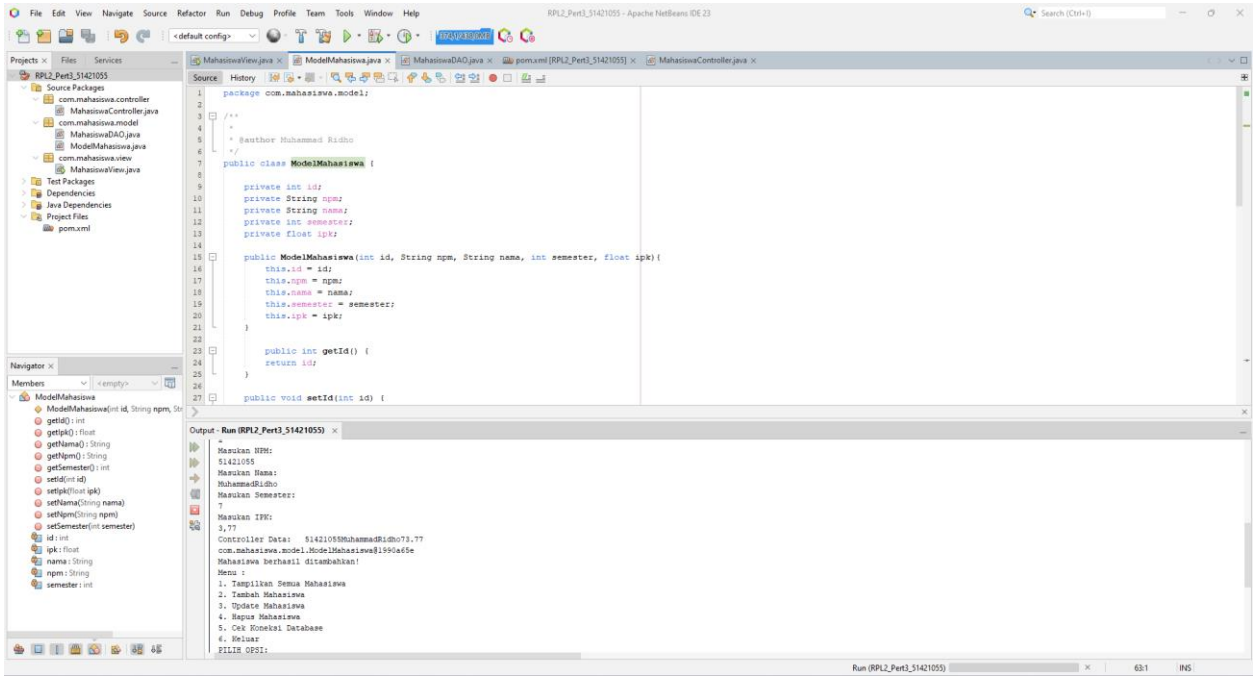
Penjelasan: setelah selesai kita bisa cek distructure gambar dibawah.





Penjelasan: Lalu kita buat projectnya dengan nama Group id, dan Package yang telah dibuat.

ModelMahasiswa.java



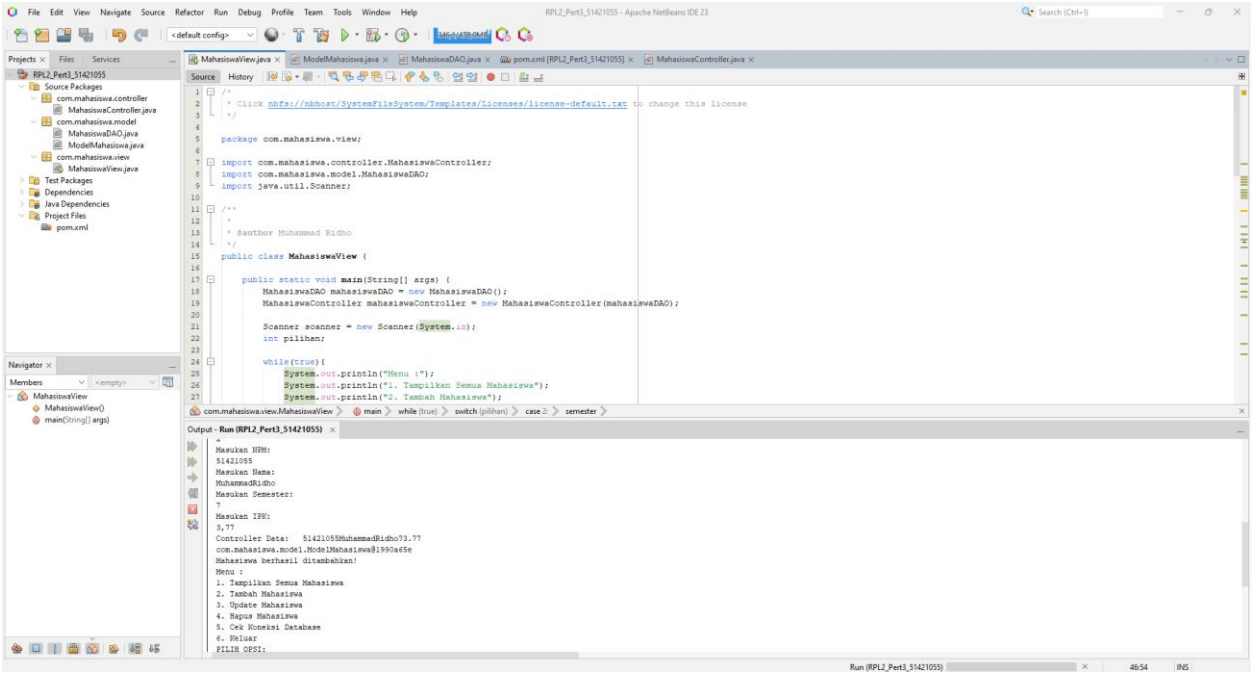
Source Code:

```
package com.mahasiswa.model;
/**
 *
 * @author Muhammad Ridho
 */
public class ModelMahasiswa {

    private int id;
    private String npm;
    private String nama;
    private int semester;
    private float ipk;

    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;
    }

        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;
    }
}
```

Penjelasan: Kelas ModelMahasiswa dalam kode di atas merupakan representasi data (model) dari seorang mahasiswa. Kelas ini memiliki beberapa atribut, yaitu id (int) untuk identitas unik mahasiswa, npm (String) sebagai Nomor Pokok Mahasiswa, nama (String) untuk nama mahasiswa, semester (int) yang menunjukkan semester saat ini, dan ipk (float) untuk menyimpan nilai Indeks Prestasi Kumulatif mahasiswa. Konstruktor ModelMahasiswa(int id, String npm, String nama, int semester, float ipk) digunakan untuk menginisialisasi objek ModelMahasiswa dengan memberikan nilai awal pada setiap atributnya. Kelas ini juga dilengkapi dengan metode getter dan setter untuk setiap atribut, yang memungkinkan pengaksesan (getter) dan pengubahan (setter) nilai atribut secara individual.

MahasiswaView.java



Source Code:

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 */

package com.mahasiswa.view;

import com.mahasiswa.controller.MahasiswaController;
import com.mahasiswa.model.MahasiswaDAO;
import java.util.Scanner;
```

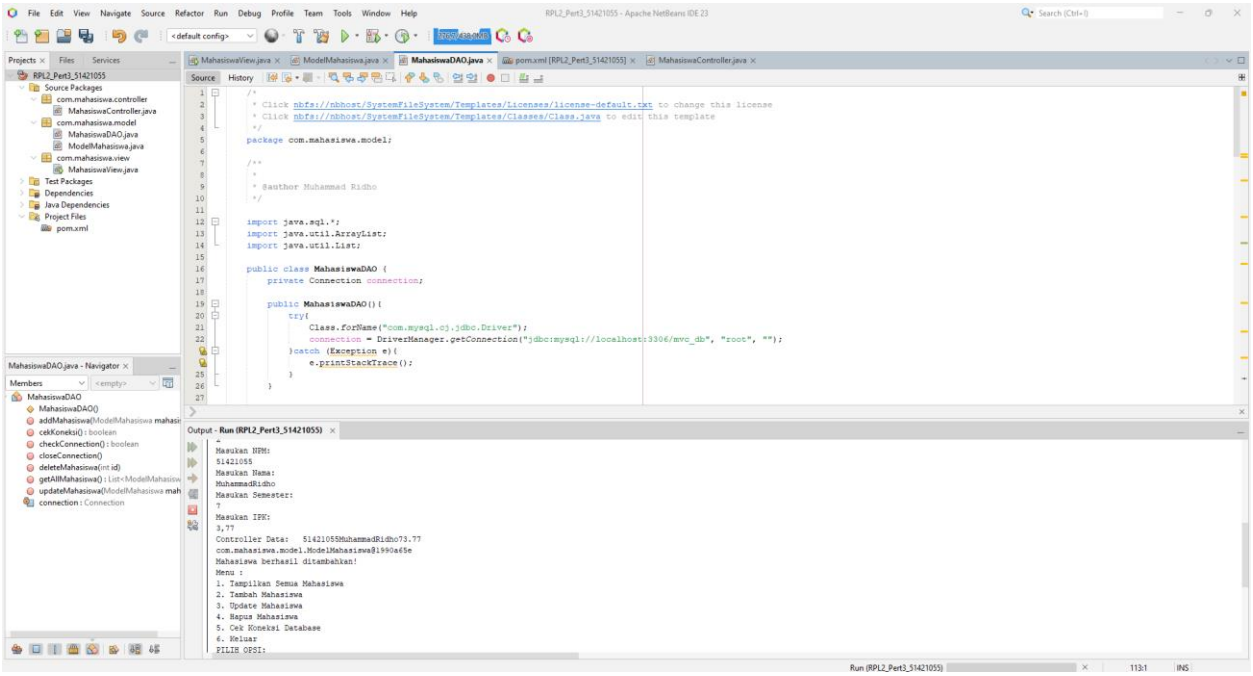
```
/**
 *
 * @author Muhammad Ridho
 */
public class MahasiswaView {

    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("Menu :");
            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.println("PILIH OPSI: ");
            pilihan = scanner.nextInt();
            scanner.nextLine();

            switch(pilihan){
                case 1:
                    mahasiswaController.displayAllMahasiswa();
                    break;
                case 2:
                    System.out.println("Masukan NPM: ");
```

```
            String npm = scanner.next();
            System.out.println("Masukan Nama: ");
            String nama = scanner.next();
            System.out.println("Masukan Semester: ");
            int semester = scanner.nextInt();
            System.out.println("Masukan IPK: ");
            float ipk = scanner.nextFloat();

            mahasiswaController.addMahasiswa(npm, nama, semester, ipk);
            break;
        case 3:
            System.out.println("Masukan ID Mahasiswa: ");
            int id = scanner.nextInt();
            scanner.nextLine();

            System.out.println("Masukan NPM: ");
            String npmBaru = scanner.next();
            System.out.println("Masukan Nama: ");
            String namaBaru = scanner.next();
            System.out.println("Masukan Semester");
            int semesterBaru = scanner.nextInt();
            System.out.println("Masukan IPK: ");
            float ipkBaru = scanner.nextFloat();

            mahasiswaController.updateMahasiswa(id, npmBaru, namaBaru, semesterBaru, ipkBaru);
            break;
        case 4:
            System.out.println("Masukan ID Mahasiswa yg Ingin Dihapus: ");
            int idHapus = scanner.nextInt();
            mahasiswaController.deleteMahasiswa(idHapus);
            break;
```

```
            case 5:
                mahasiswaController.checkDatabaseConnection();
                break;
            case 6:
                mahasiswaController.closeConnection();
                System.out.println("Program selesai");
                return;
            default:
                System.out.println("Input Tidak Valid");
        }
    }
  }
}
```

Penjelasan: Kelas MahasiswaView pada kode di atas adalah kelas utama untuk menampilkan antarmuka teks sederhana yang memungkinkan pengguna berinteraksi dengan aplikasi manajemen data mahasiswa. Kelas ini mengimpor MahasiswaController dari package controller dan MahasiswaDAO dari package model, serta menggunakan Scanner untuk menerima input dari pengguna. Di dalam metode main, objek MahasiswaDAO dan MahasiswaController dibuat untuk mengelola data mahasiswa. Program ini menyediakan menu utama yang memungkinkan pengguna untuk menampilkan seluruh data mahasiswa, menambahkan mahasiswa baru, memperbarui data mahasiswa, menghapus mahasiswa, mengecek koneksi database, dan keluar dari program. Setiap opsi pada menu akan dieksekusi dengan menggunakan struktur switch-case, yang memanggil metode di MahasiswaController sesuai dengan pilihan yang dimasukkan pengguna.

MahasiswaDAO.java



Source Code:

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mahasiswa.model;

/**
 *
 * @author Muhammad Ridho
 */

import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class MahasiswaDAO {
    private Connection connection;
```

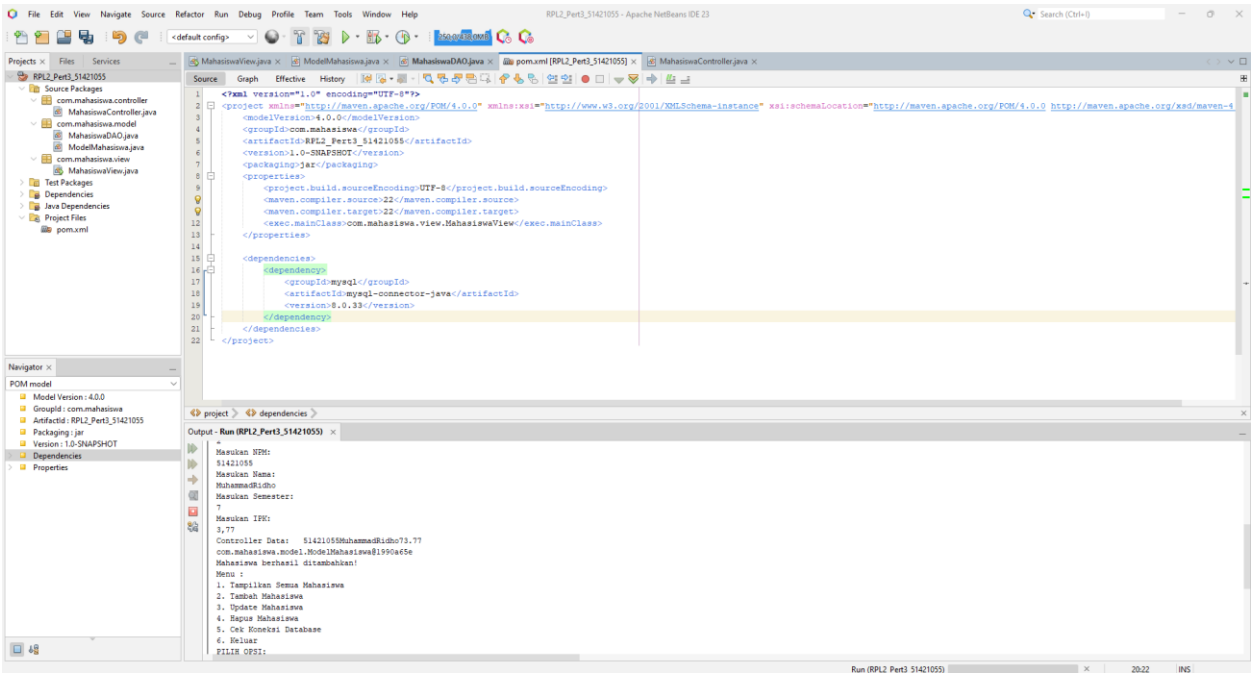
```
    public MahasiswaDAO(){
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/mvc_db", "root",
"");
        }catch (Exception e){
            e.printStackTrace();
        }
    }
    public boolean cekKoneksi () {
        try{
            if(connection != null && connection.isClosed());
                return true;
        }catch (SQLException e) {
            e.printStackTrace();
        }
        return false;
    }
    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();
        }
```

```
}
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;
}
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();
        }
    }
    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 untuk menutup koneksi database
    public void closeConnection() {
        try {
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
    public boolean checkConnection() {
        throw new UnsupportedOperationException("Not supported yet."); // Generated from
nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/GeneratedMethodBody
    }
}
```

Penjelasan: Kelas MahasiswaDAO dalam kode di atas berfungsi sebagai pengelola akses data mahasiswa ke dalam database (data access object). Kelas ini memiliki atribut connection yang digunakan untuk menghubungkan aplikasi dengan database MySQL. Konstruktornya memuat konfigurasi untuk menginisialisasi koneksi ke database. Kelas ini menyediakan beberapa metode untuk pengelolaan data mahasiswa, antara lain addMahasiswa(ModelMahasiswa mahasiswa) untuk menambah data mahasiswa, getAllMahasiswa() untuk mendapatkan seluruh data mahasiswa dari database, updateMahasiswa(ModelMahasiswa mahasiswa) untuk memperbarui data mahasiswa berdasarkan ID, serta deleteMahasiswa(int id) untuk menghapus data mahasiswa dari database. Selain itu, ada metode cekKoneksi() untuk memeriksa koneksi ke database dan closeConnection() untuk menutup koneksi database setelah selesai digunakan.

Pom.xml



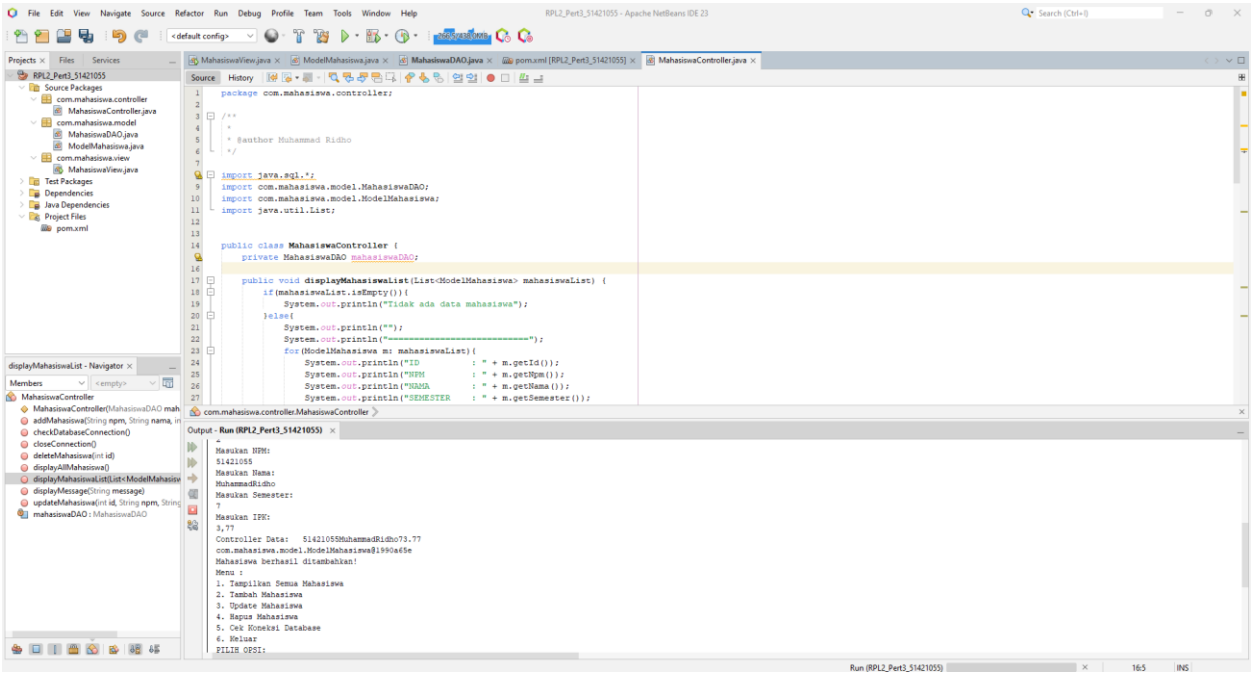
Source Code:

```
<?xml version="1.0" encoding="UTF-8"?>

<project xmlns="http://maven.apache.org/POM/4.0.0"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">

    <modelVersion>4.0.0</modelVersion>

    <groupId>com.mahasiswa</groupId>

    <artifactId>RPL2_Pert3_51421055</artifactId>

    <version>1.0-SNAPSHOT</version>

    <packaging>jar</packaging>
```

```
<properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>22</maven.compiler.source>
        <maven.compiler.target>22</maven.compiler.target>
        <exec.mainClass>com.mahasiswa.view.MahasiswaView</exec.mainClass>
    </properties>

    <dependencies>
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.33</version>
        </dependency>
    </dependencies>
</project>
```

Penjelasan: Kode di atas adalah file konfigurasi Maven (pom.xml) untuk proyek Java. File ini mendefinisikan atribut proyek, seperti groupId (com.mahasiswa), artifactId (RPL2_Pert3_51421055), dan version (versi proyek yaitu 1.0-SNAPSHOT). Bagian <properties> mengatur pengkodean sumber (UTF-8) dan menentukan versi Java yang digunakan (Java 22). Properti exec.mainClass juga disertakan untuk menunjuk kelas utama MahasiswaView yang akan dijalankan saat aplikasi dimulai. Dalam <dependencies>, proyek ini mengimpor pustaka mysql-connector-java versi 8.0.33, yang memungkinkan proyek berkomunikasi dengan database MySQL.

MahasiswaController.java



Source Code:

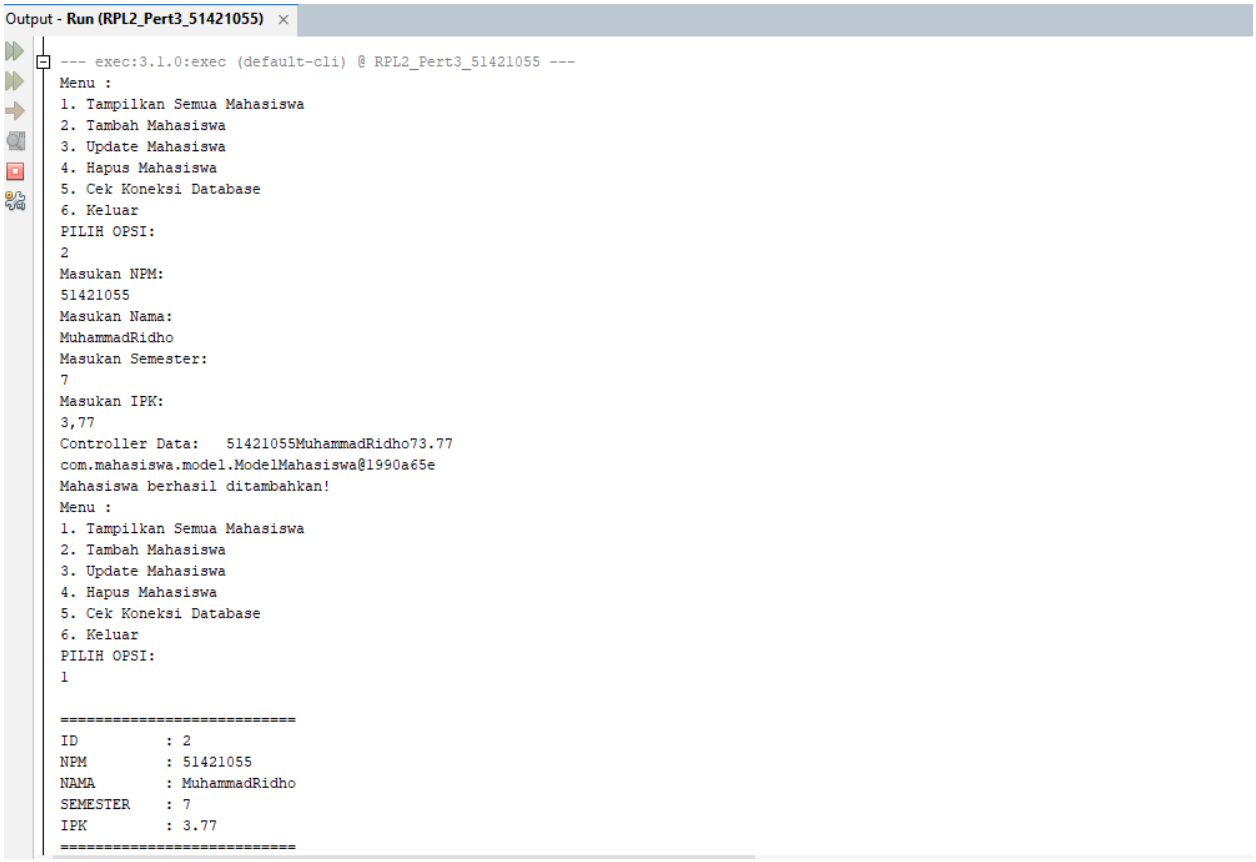
```
package com.mahasiswa.controller;


/**
 *
 * @author Muhammad Ridho
 */

import java.sql.*;
import com.mahasiswa.model.MahasiswaDAO;
import com.mahasiswa.model.ModelMahasiswa;
import java.util.List;


public class MahasiswaController {
    private MahasiswaDAO mahasiswaDAO;

    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());
            System.out.println("============================");
        }
        displayMessage("Mahasiswa berhasi ditampilkan");
    }
}
public void displayMessage(String message){
    System.out.println(message);
}
public MahasiswaController(MahasiswaDAO mahasiswaDAO){
    this.mahasiswaDAO = mahasiswaDAO;
}
public void checkDatabaseConnection(){
    boolean isConnected = mahasiswaDAO.cekKoneksi();
    if (isConnected){
        displayMessage("Koneksi ke db berhasil");
    } else{
        displayMessage("Koneksi DB Gagal");
    }
}
public void displayAllMahasiswa(){
    List<ModelMahasiswa> mahasiswaList = mahasiswaDAO.getAllMahasiswa();
```

```
        displayMahasiswaList(mahasiswaList);
    }
    public void addMahasiswa(String npm, String nama, int semester, float ipk){
        ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm, nama, semester, ipk);
        System.out.println("Controller Data:   " + npm + nama + semester + ipk);
        System.out.println(mahasiswaBaru);
        mahasiswaDAO.addMahasiswa(mahasiswaBaru);
        displayMessage("Mahasiswa berhasil ditambahkan!");
    }
    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 diperbarui!");
    }
    public void deleteMahasiswa(int id){
        mahasiswaDAO.deleteMahasiswa(id);
        displayMessage("Mahasiswa Berhasil Dihapus!");
    }
    public void closeConnection() {
        mahasiswaDAO.closeConnection();
    }
}
```

Penjelasan: Kelas MahasiswaController dalam kode di atas bertanggung jawab sebagai penghubung antara antarmuka pengguna dan akses data, dengan menggunakan MahasiswaDAO untuk melakukan operasi terkait data mahasiswa. Kelas ini memiliki atribut mahasiswaDAO, yang diinisialisasi melalui konstruktor MahasiswaController(MahasiswaDAO mahasiswaDAO). Metode displayMahasiswaList menampilkan daftar mahasiswa beserta detailnya seperti ID, NPM, NAMA, SEMESTER, dan IPK, atau pesan "Tidak ada data mahasiswa" jika kosong. Metode displayMessage menampilkan pesan tertentu di konsol. Selain itu, ada checkDatabaseConnection untuk memeriksa status koneksi ke database, displayAllMahasiswa untuk menampilkan seluruh data mahasiswa, addMahasiswa untuk menambahkan mahasiswa baru, updateMahasiswa untuk memperbarui data mahasiswa berdasarkan ID, dan deleteMahasiswa untuk menghapus data mahasiswa berdasarkan ID. Metode closeConnection digunakan untuk menutup koneksi database.

OUTPUT:


Output - Run (RPL2_Pert3_51421055)
--- exec:3.1.0:exec (default-cli) @ RPL2_Pert3_51421055 ---
Menu :
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI:
2
Masukan NPM:
51421055
Masukan Nama:
MuhammadRidho
Masukan Semester:
7
Masukan IPK:
3,77
Controller Data: 51421055MuhammadRidho73.77
com.mahasiswa.model.ModelMahasiswa@1990a65e
Mahasiswa berhasil ditambahkan!
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 : 51421055
NAMA : MuhammadRidho
SEMESTER : 7
IPK : 3.77
==========