ACTIVITY PERTEMUAN 3

NAMA : **Johan**

NPM : **50421686**

KELAS : **4IA28**

MATERI : **KONSEP MODEL, VIEW, CONTROLLER (MVC)**

MATA PRAKTIKUM : **Rekayasa Perangkat Lunak 2**

Source Code:

MahasiswaController.java

MahasiswaController.java

Source History

```java
package com.mahasiswa.controller;

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

/**
 *
 * @author Johan
 */
public class MahasiswaController {
private MahasiswaDAO mahasiswaDAO;

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

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



    public void checkDatabaseConnection(){
        boolean isConnected = mahasiswaDAO.checkConnection();
        if (isConnected){
            displayMessage("Koneksi ke db berhasil");
        } else{
            displayMessage("Koneksi DB Gagal");
        }
    }

    // READ ALL (Menampilkan semua mahasiswa)
    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!");
    }
```

```java
    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();
    }
}
```

Code diatas adalah controller untuk mengelola data mahasiswa. Controller ini terhubung ke MahasiswaDAO untuk berinteraksi dan bisa melakukan CRUD (Create, Read, Update dan Delete) pada data mahasiswa.

MahasiswaDAO.java

MahasiswaDAO.java ×

Source History

```java
package com.mahasiswa.model;

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

/**
 *
 * @author Johan
 */
public class MahasiswaDAO {
    private Connection connection;

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

    public boolean checkConnection(){
        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;
     }
```

```java
    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();
        }
    }
```

```java
    public void closeConnection(){
        try{
            if (connection != null){
                connection.close();
            }

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

Code MahasiswaDAO.java ini berfungsi sebagai Data Access Object (DAO) yang menangani interaksi dengan database untuk entitas mahasiswa. Jadi, code ini menjadi penghubung antara MahasiswaController dan database, memastikan data dapat diakses dan dimodifikasi sesuai kebutuhan aplikasi.

ModelMahasiswa.java ×

Source History

```java
package com.mahasiswa.model;

/**
 *
 * @author Lenovo
 */
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;
    }



}
```

Code ModelMahasiswa.java ini mendefinisikan entitas mahasiswa sebagai objek data, yang digunakan oleh MahasiswaDAO dan MahasiswaController dalam pengelolaan data di aplikasi.

MahasiswaView.java

MahasiswaView.java ×

Source History

```java
package com.mahasiswa.view;

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

/**
 *
 * @author Johan
 */
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.print("PILIH OPSI: ");
            pilihan = scanner.nextInt();
            scanner.nextLine();
```

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

                case 2:
                    // tambah mhs
                    System.out.println("Masukkan NPM: ");
                    String npm = scanner.next();
                    System.out.println("Masukkan Nama: ");
                    String nama = scanner.next();
                    System.out.println("Masukkan Semester: ");
                    int semester = scanner.nextInt();
                    System.out.println("Masukkan IPK: ");
                    float ipk = scanner.nextFloat();
                    System.out.println(npm + nama + semester + ipk);

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

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

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

                    mahasiswaController.updateMahasiswa(id, npmBaru, namaBaru, semesterBaru, ipkBaru);
                    break;
                case 4:
                    System.out.print("Masukkan ID Mahasiswa: ");
                    int idHapus = scanner.nextInt();
                    mahasiswaController.deleteMahasiswa(idHapus);
                case 5:
                    mahasiswaController.checkDatabaseConnection();
                    break;
                case 6:
                    // Keluar
                    mahasiswaController.closeConnection();
                    System.out.println("Program selesai.");
                    return;
                default:
                    System.out.println("Input Tidak valid");
            }
        }
    }
}
```

Code MahasiswaView.java ini adalah kelas utama yang berperan sebagai antarmuka tampilan (view) dan menyediakan menu interaktif untuk pengguna, memungkinkan untuk melakukan operasi CRUD pada data mahasiswa.

Pom.xml

pom.xml [Johan_MVC]

Source Graph Effective History

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.mycompany</groupId>
    <artifactId>Johan_MVC</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>21</maven.compiler.source>
        <maven.compiler.target>21</maven.compiler.target>
        <exec.mainClass>com.mycompany.johan_mvc.Johan_MVC</exec.mainClass>
    </properties>
    <dependencies>
        <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.33</version>
    </dependency>
    </dependencies>


</project>
```

Project files yaitu pom.xml ini berfungsi untuk mengonfigurasi proyek Maven yang sedang dikerjakan dengan struktur MVC. Struktur pom xml ini terdiri dari informasi dasar proyek, properties, dan juga dependencies.

Output:

- Menambahkan 3 data mahasiswa
  1. Data data mahasiswa ke-1

```
Output ×
Run (MahasiswaView) ×   Run (MahasiswaView) ×
--- exec:3.1.0:exec (default-cli) @ Johan_MVC ---
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:
50421686
Masukkan Nama:
Johan
Masukkan Semester:
7
Masukkan IPK:
3.93
50421686Johan73.93
Controller Data:   50421686Johan73.93
com.mahasiswa.model.ModelMahasiswa@782663d3
Mahasiswa berhasil ditambahkan!
```

2. Data data mahasiswa ke-2

```
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:
50421687
Masukkan Nama:
Karina
Masukkan Semester:
1
Masukkan IPK:
4.0
50421687Karina14.0
Controller Data:   50421687Karina14.0
com.mahasiswa.model.ModelMahasiswa@24b1d79b
Mahasiswa berhasil ditambahkan!
```

3. Data data mahasiswa ke-3

```
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:
50421688
Masukkan Nama:
NingNing
Masukkan Semester:
3
Masukkan IPK:
3.99
50421688NingNing33.99
Controller Data:   50421688NingNing33.99
com.mahasiswa.model.ModelMahasiswa@68ceda24
Mahasiswa berhasil ditambahkan!
```

4. Maka, tampilan data mahasiswa setelah menamhahkan 3 data adalah sebagai berikut:

```
Output ×
Run (MahasiswaView) ×   Run (MahasiswaView) ×

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

============================
ID          : 5
NPM         : 50421686
NAMA        : Johan
SEMESTER    : 7
IPK         : 3.93
============================
ID          : 6
NPM         : 50421687
NAMA        : Karina
SEMESTER    : 1
IPK         : 4.0
============================
ID          : 7
NPM         : 50421688
NAMA        : NingNing
SEMESTER    : 3
IPK         : 3.99
============================
```

- Mengupdate data mahasiswa
  1. Data mahasiswa yang ingin di update adalah data dengan ID no. 7, yaitu NingNing dengan data mahasiswa baru.

```
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: 7
Masukkan NPM:
50421688
Masukkan Nama:
Winter
Masukkan Semester:
4
Masukkan IPK:
3.89
Mahasiswa berhasil diperbarui!
```

  2. Maka, tampilan tabel data mahasiswa setelah diupdate adalah sebagai berikut:

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

============================
ID          : 5
NPM         : 50421686
NAMA        : Johan
SEMESTER    : 7
IPK         : 3.93
============================
ID          : 6
NPM         : 50421687
NAMA        : Karina
SEMESTER    : 1
IPK         : 4.0
============================
ID          : 7
NPM         : 50421688
NAMA        : Winter
SEMESTER    : 4
IPK         : 3.89
============================
```

- Menghapus data mahasiswa

1. Data mahasiwa yang ingin dihapus adalah data dengan ID no. 5 yaitu Johan.

```
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: 5
Mahasiswa Berhasil Dihapus!
Koneksi ke db berhasil
```

2. Maka, tampilan tabel mahasiswa setelah ID no.5 dihapus adalah sebagai berikut:

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

============================
ID          : 6
NPM         : 50421687
NAMA        : Karina
SEMESTER    : 1
IPK         : 4.0
============================
ID          : 7
NPM         : 50421688
NAMA        : Winter
SEMESTER    : 4
IPK         : 3.89
============================
```

ACTIVITY PERTEMUAN 4

NAMA : **Johan**

NPM : **50421686**

KELAS : **4IA28**

MATERI : **ORM**

MATA PRAKTIKUM : **Rekayasa Perangkat Lunak 2**

Source Code

Pom.xml

```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.mycompany</groupId>
    <artifactId>MahasiwaORM</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.source>21</maven.compiler.source>
        <maven.compiler.target>21</maven.compiler.target>
        <exec.mainClass>com.mycompany.mahasiwaorm.MahasiwaORM</exec.mainClass>
    </properties>
    <dependencies>
        <dependency>
            <groupId>org.hibernate.orm</groupId>
            <artifactId>hibernate-core</artifactId>
            <version>6.6.0.Final</version>
        </dependency>

        <!-- MySQL Connector -->
        <dependency>
            <groupId>mysql</groupId>
            <artifactId>mysql-connector-java</artifactId>
            <version>8.0.33</version>
        </dependency>
    </dependencies>
    <build>
    <resources>
        <resource>
            <directory>src/main/resources</directory>
            <filtering>false</filtering>
        </resource>
    </resources>
    </build>
</project>
```

MahasiswaController.java

```java
package com.mahasiswa.controller;

import com.mahasiswa.model.ModelMahasiswa;
import java.util.List;

/**
 *
 * @author Johan
 */
public interface MahasiswaController {
    public void addMhs(ModelMahasiswa mhs);
    public ModelMahasiswa getMhs(int id);
    public void updateMhs(ModelMahasiswa mhs);
    public void deleteMhs(int id );
    public List<ModelMahasiswa> getAllMahasiswa();
}

```

MahasiswaControllerImpl.java

```java
package com.mahasiswa.controller;

import com.mahasiswa.model.ModelMahasiswa;
import java.util.List;
import com.mahasiswa.model.HibernateUtil;
import org.hibernate.query.Query;
import org.hibernate.Session;
import org.hibernate.Transaction;


/**
 *
 * @author Johan
 */
public class MahasiswaControllerImpl implements MahasiswaController {

@Override
    public void addMhs(ModelMahasiswa mhs){
        Transaction trx = null;
```

```java
        try (Session session = HibernateUtil.getSessionFactory().openSession()){
            trx = session.beginTransaction();
            session.save(mhs);
            trx.commit();
        }catch (Exception e){
            if (trx != null){
                trx.rollback();
            }
            e.printStackTrace();
        }
    }


    @Override
    public void updateMhs(ModelMahasiswa mhs) {
        Transaction trx = null;

        try (Session session = HibernateUtil.getSessionFactory().openSession()){
            trx = session.beginTransaction();
            session.update(mhs);
            trx.commit();
        } catch (Exception e){
            if (trx != null){
                trx.rollback();
            }
            e.printStackTrace();
        }

    }

    @Override
    public void deleteMhs(int id) {
        Transaction trx = null;
```

```java
        try (Session session = HibernateUtil.getSessionFactory().openSession()){
            trx = session.beginTransaction();
            ModelMahasiswa mhs = session.get(ModelMahasiswa.class, id);
            if(mhs != null){
                session.delete(mhs);
                System.out.println("Berhasil hapus");
            }
            trx.commit();
        } catch (Exception e){
            if (trx != null){
                trx.rollback();
            }
            e.printStackTrace();
        }

    }

    @Override
    public List<ModelMahasiswa> getAllMahasiswa() {
        Transaction trx = null;
        List<ModelMahasiswa> listMhs = null;

        try (Session session = HibernateUtil.getSessionFactory().openSession()){
            trx = session.beginTransaction();
            // Using HQL (Hibernate Query Language) to fetch all records
            Query<ModelMahasiswa> query = session.createQuery("from ModelMahasiswa", ModelMahasiswa.class);
            listMhs = query.list(); // Fetch all results
```

```java
            trx.commit(); // Commit transaction
        } catch (Exception e) {
            if (trx != null) {
                trx.rollback(); // Rollback transaction in case of error
            }
            e.printStackTrace();
        }

        // Return the fetched list
        return listMhs;
    }

    @Override
    public ModelMahasiswa getMhs(int id) {
        throw new UnsupportedOperationException("Not supported yet.");
    }

}
```

HibernateUtil.java

```java
package com.mahasiswa.model;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

/**
 *
 * @author Johan
 */
public class HibernateUtil {
    private static SessionFactory sessionFactory;

    static {
        try {
            // Create the SessionFactory from hibernate.cfg.xml
            sessionFactory = new Configuration().configure().buildSessionFactory();
        } catch (Throwable ex) {
            // Make sure you log the exception, as it might be swallowed
            System.err.println("Initial SessionFactory creation failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }
    public static void testConnection() {
        try (Session session = sessionFactory.openSession()) {
            System.out.println("Connection to the database was successful!");
        } catch (Exception e) {
            System.err.println("Failed to connect to the database.");
            e.printStackTrace();
        }
    }
}
```

```java
package com.mahasiswa.model;
import jakarta.persistence.*;
/**
 *
 * @author Johan
 */
@Entity
@Table(name = "mahasiswa")
public class ModelMahasiswa {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private int id;

    @Column(name="npm",nullable = false, length = 8)
    private String npm;
    @Column(name="nama",nullable = false, length = 8)
    private String nama;
    @Column(name="semester")
    private int semester;
    @Column(name="ipk")
    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 ModelMahasiswa(){

    }
    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;
    }
```

ModelTabelMahasiswa.java

```java
package com.mahasiswa.model;

import java.util.List;
import javax.swing.table.AbstractTableModel;

/**
 *
 * @author Johan
 */
public class ModelTabelMahasiswa extends AbstractTableModel {
    private List<ModelMahasiswa> mahasiswaList;
    private String[] columnNames = {"ID", "NPM", "Nama", "Semester", "IPK"};

    public ModelTabelMahasiswa(List<ModelMahasiswa> mahasiswaList) {
        this.mahasiswaList = mahasiswaList;
    }

    @Override
    public int getRowCount() {
        return mahasiswaList.size(); // Jumlah baris sesuai dengan jumlah data mahasiswa
    }

    @Override
    public int getColumnCount() {
        return columnNames.length; // Jumlah kolom sesuai dengan jumlah elemen dalam columnNames
    }
```

```java
    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        ModelMahasiswa mahasiswa = mahasiswaList.get(rowIndex);
        switch (columnIndex) {
            case 0:
                return mahasiswa.getId();
            case 1:
                return mahasiswa.getNpm();
            case 2:
                return mahasiswa.getNama();
            case 3:
                return mahasiswa.getSemester();
            case 4:
                return mahasiswa.getIpk();
            default:
                return null;
        }
    }

    @Override
    public String getColumnName(int column) {
        return columnNames[column]; // Mengatur nama kolom
    }

    @Override
    public boolean isCellEditable(int rowIndex, int columnIndex) {
        return false; // Semua sel tidak dapat diedit
    }

    // Method untuk menambahkan atau memodifikasi data, jika dibutuhkan
    public void setMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
        this.mahasiswaList = mahasiswaList;
        fireTableDataChanged(); // Memberitahu JTable bahwa data telah berubah
    }

}
```

Desain MahasiswaView.java