

## **LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : Rekayasa Perangkat Lunak 2  
Kelas : 4IA06  
Praktikum ke- : 4  
Tanggal : 5 November 2024  
Materi : Konsep Dasar Object Relational Mapping (ORM) dan Framework Hibernate  
NPM : 51421517  
Nama : William Devin Septianus Pranggono  
Ketua Asisten :  
Paraf Asisten :  
Nama Asisten : Gilbert Jefferson Faozato Mendrofa  
Jumlah Lembar : 29 lembar

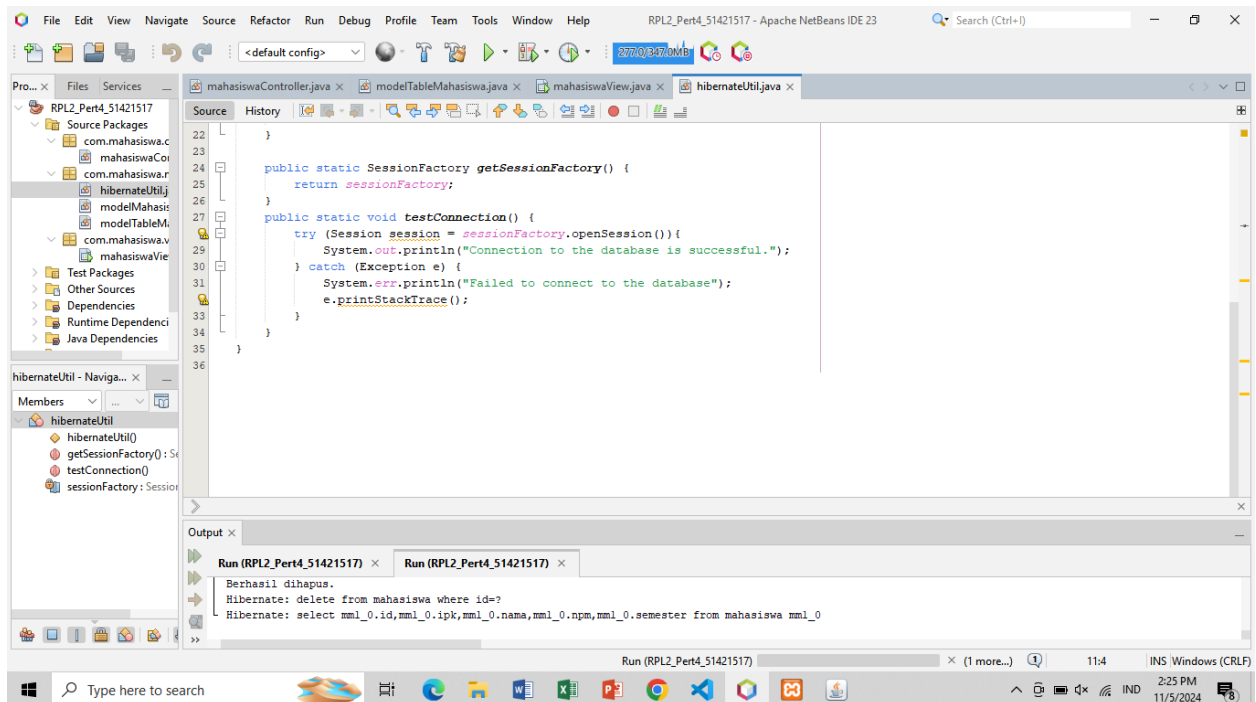
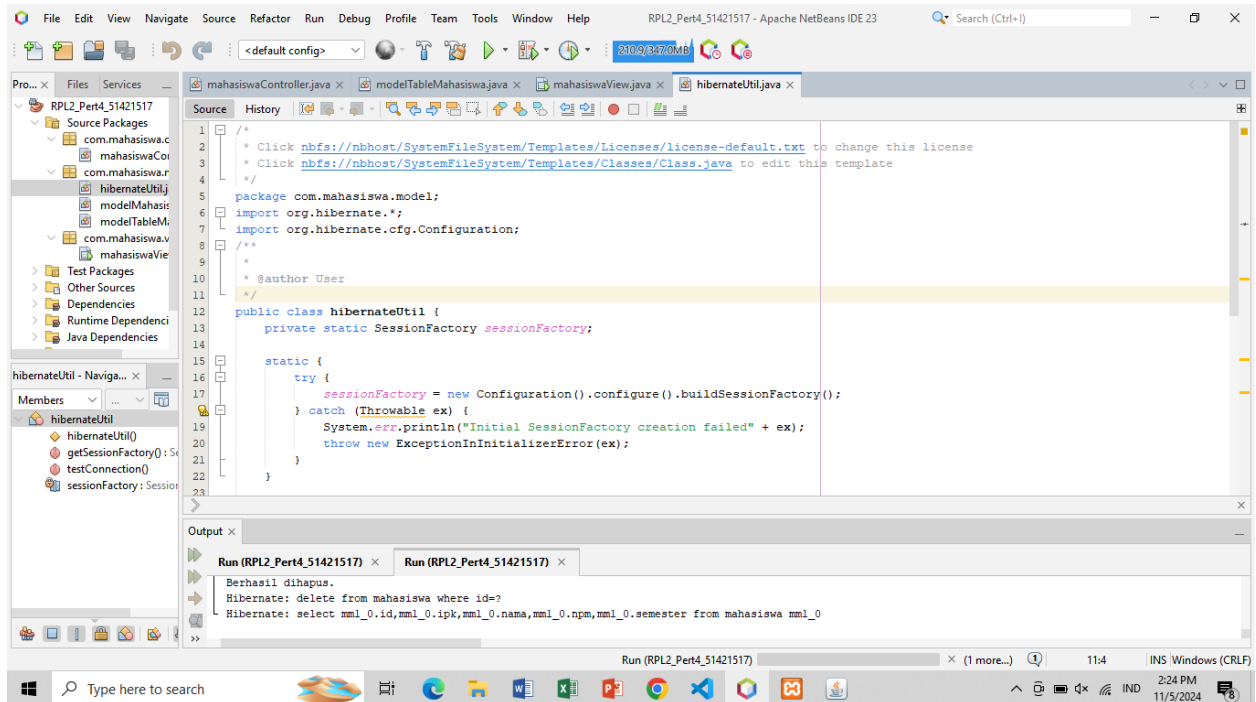
**LABORATORIUM TEKNIK INFORMATIKA**

**UNIVERSITAS GUNADARMA**

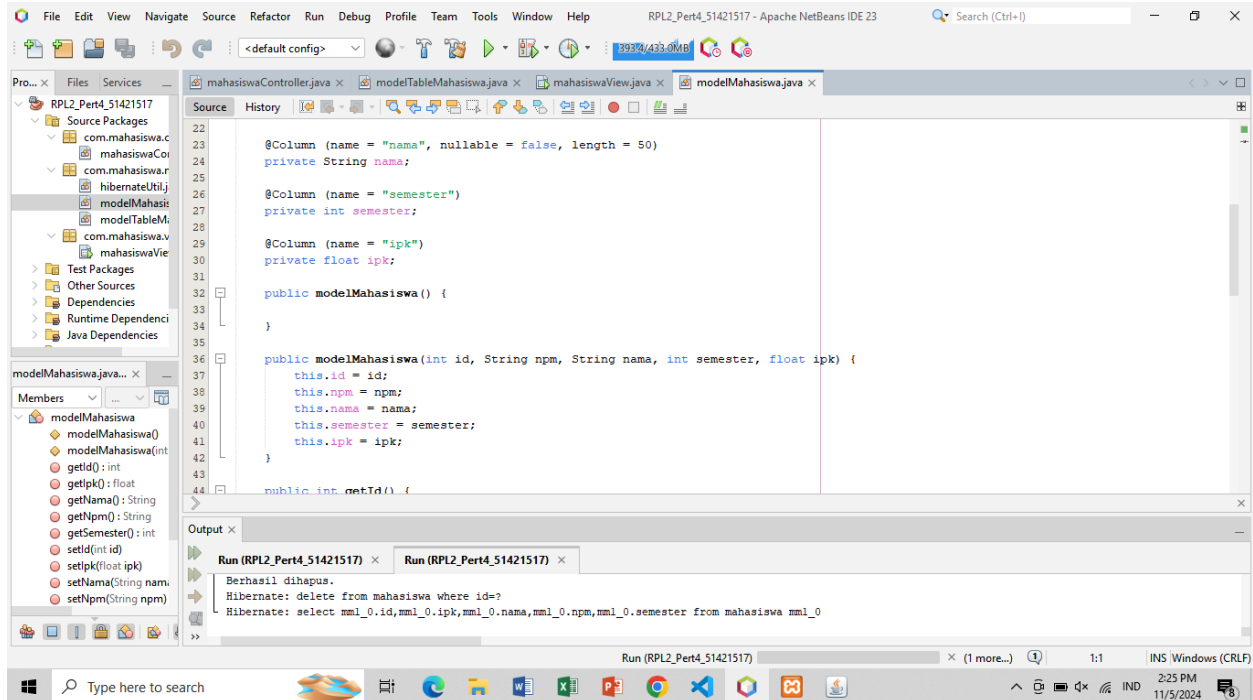
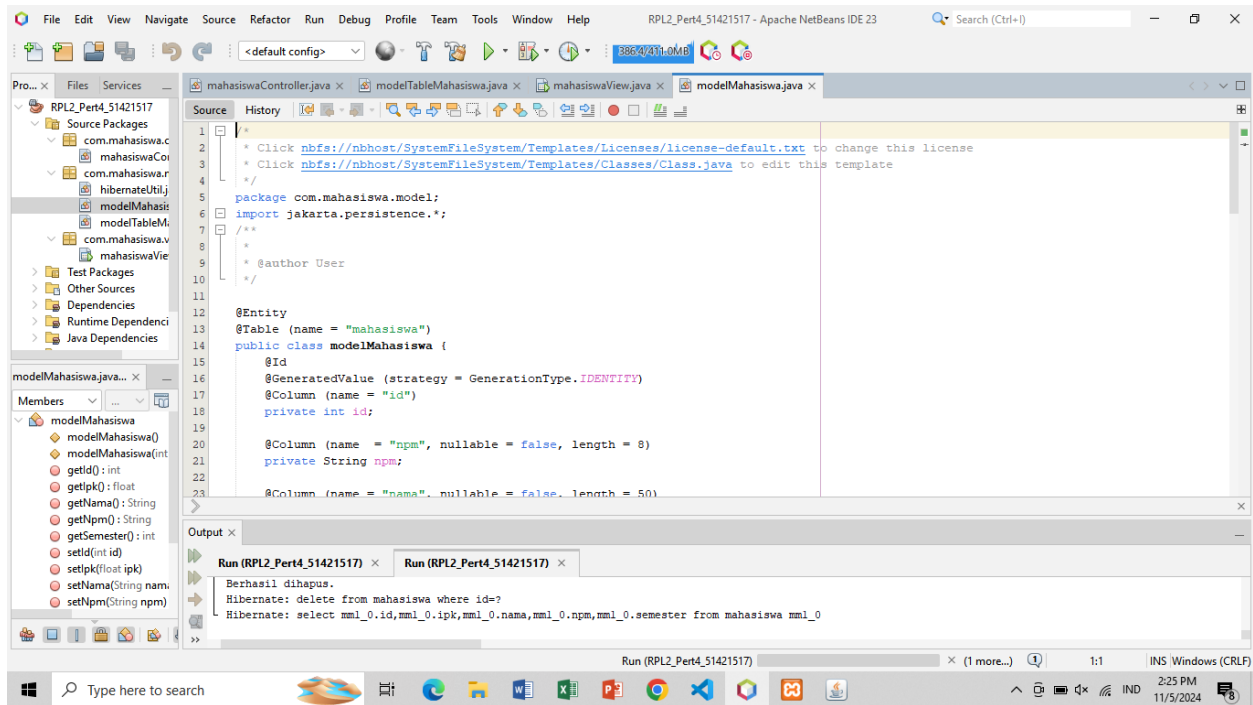
**2024**

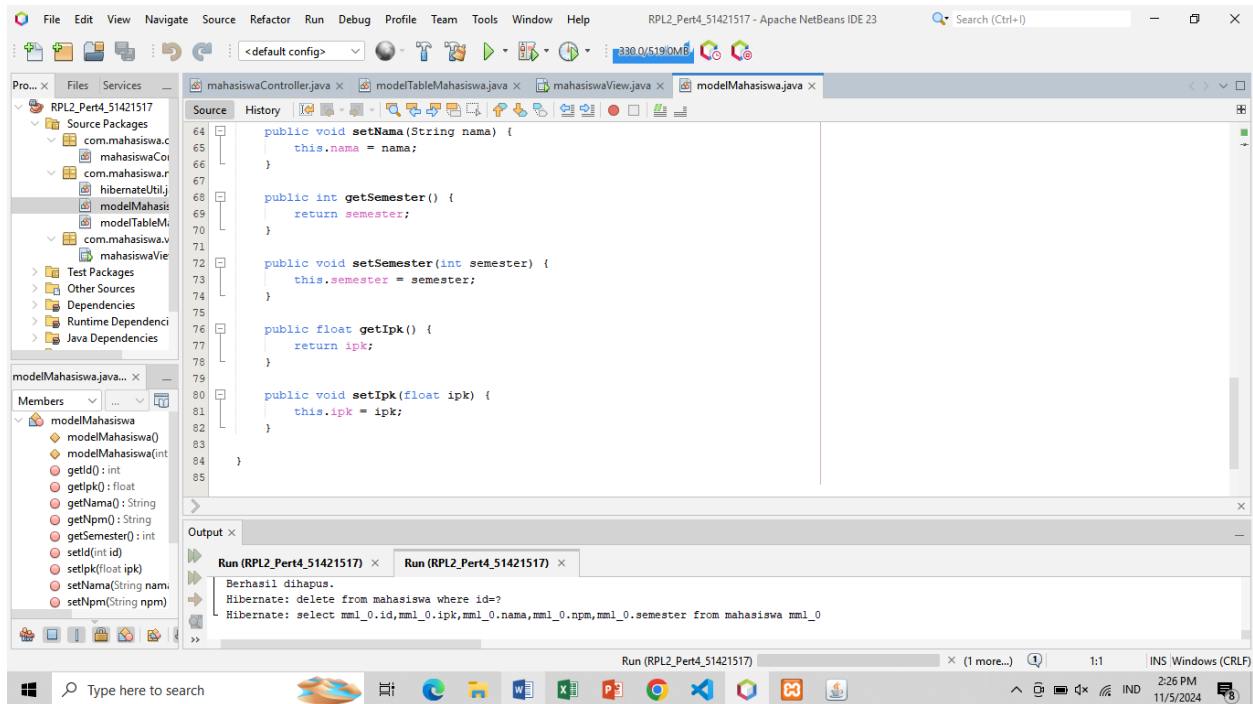
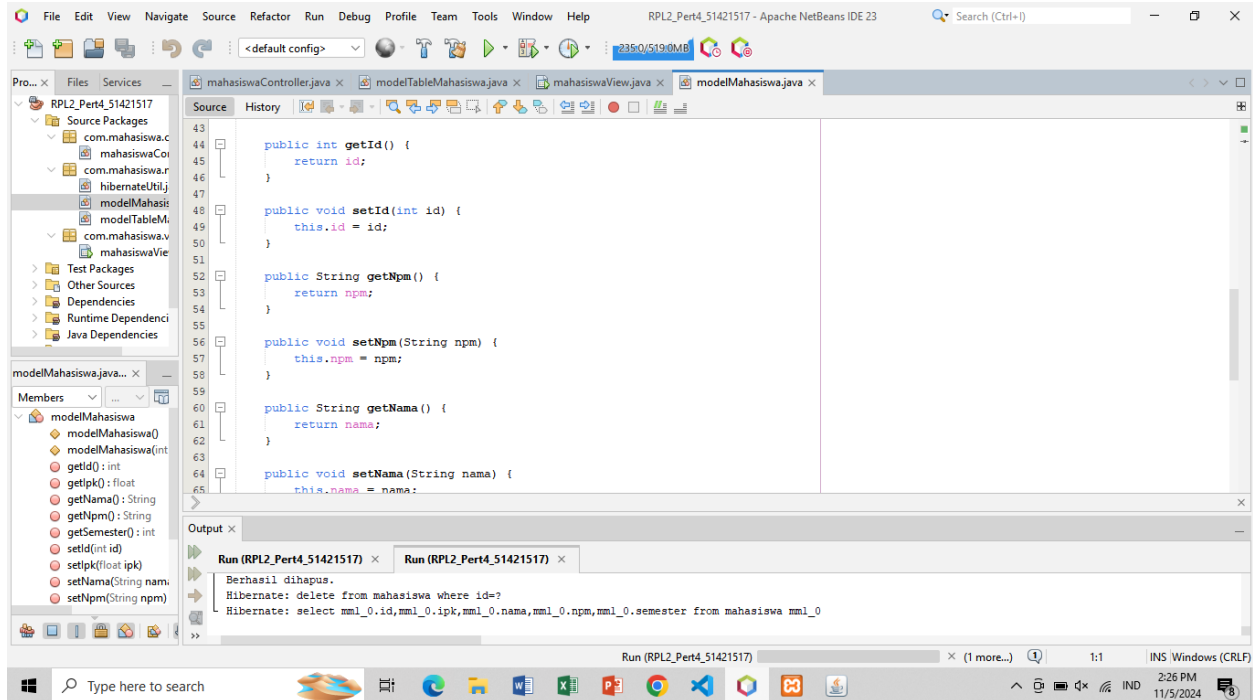
## LISTING PROGRAM

### a. hibernateUtil.java :

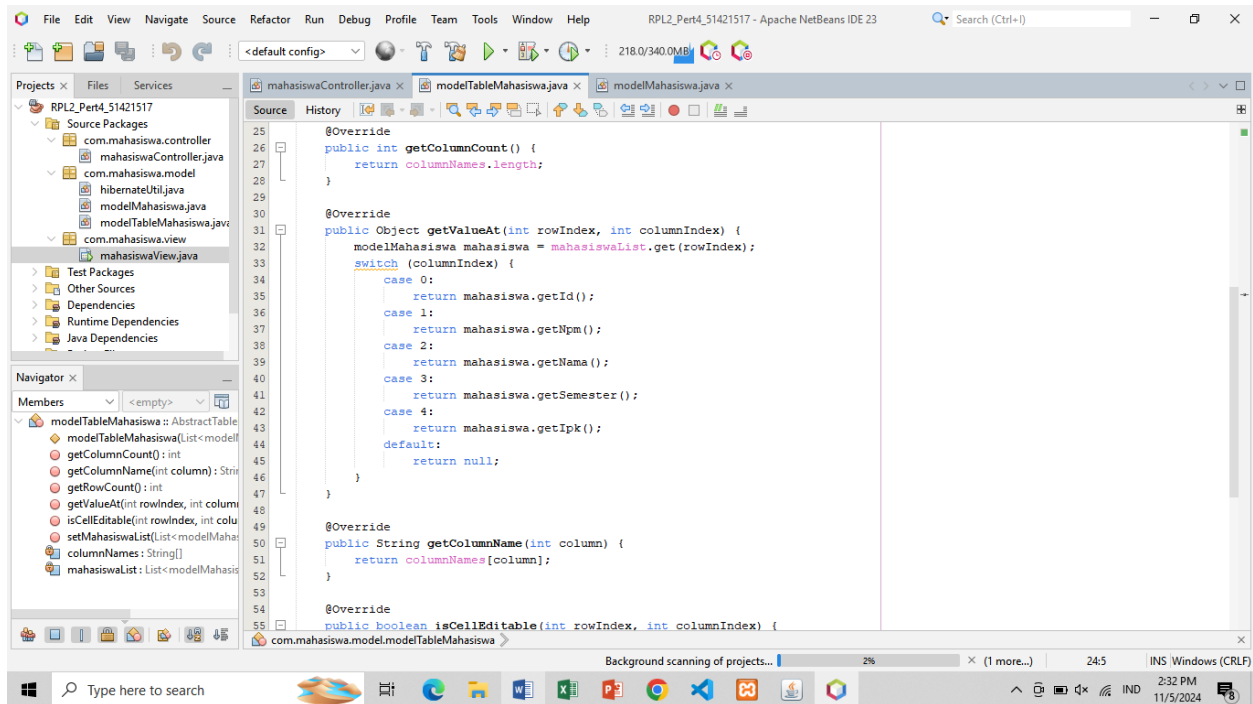
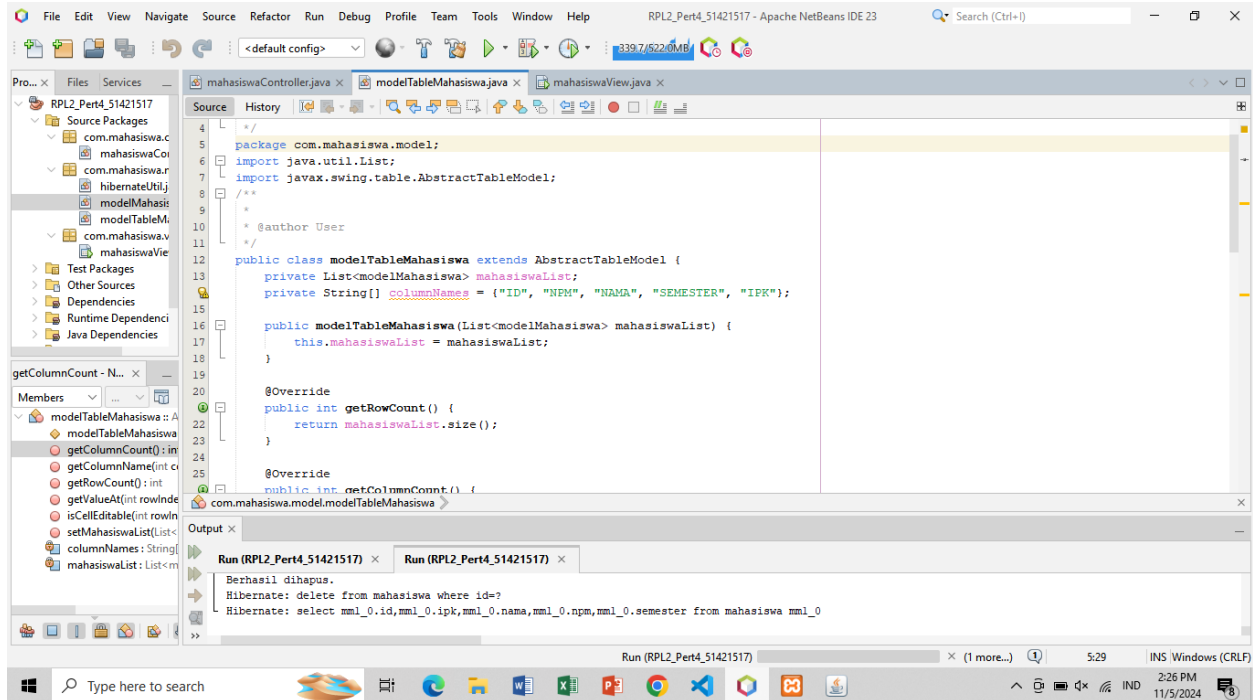


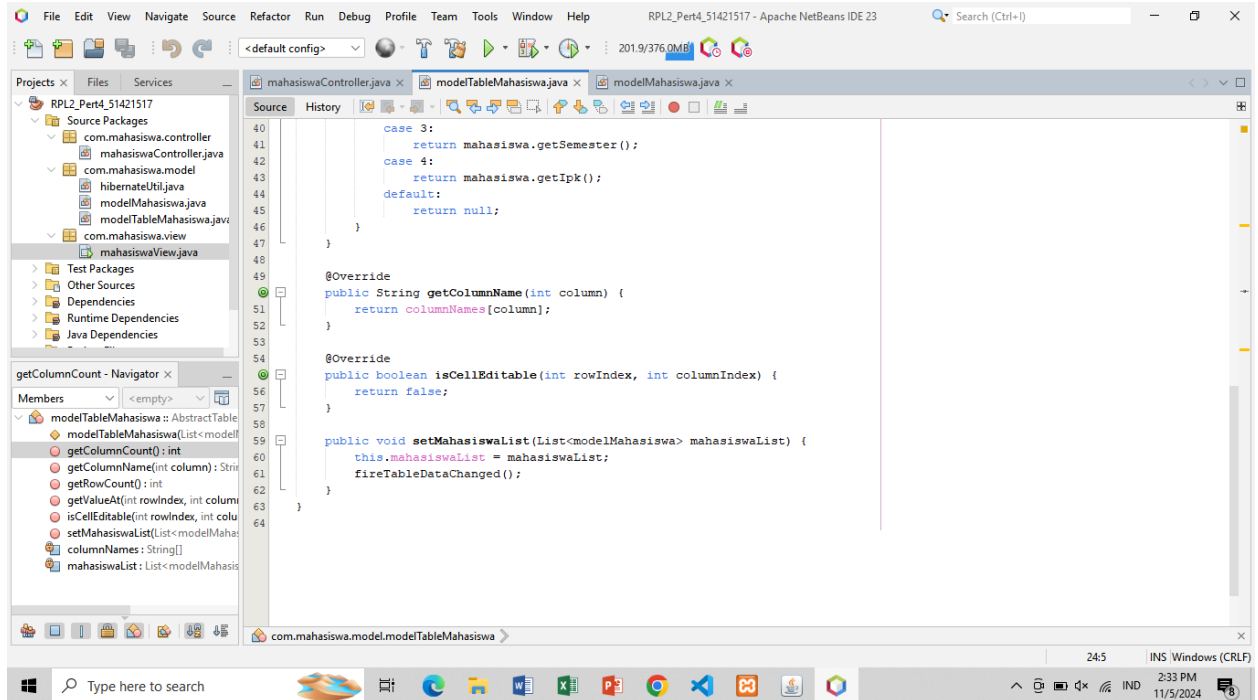
b. modelMahasiswa.java :



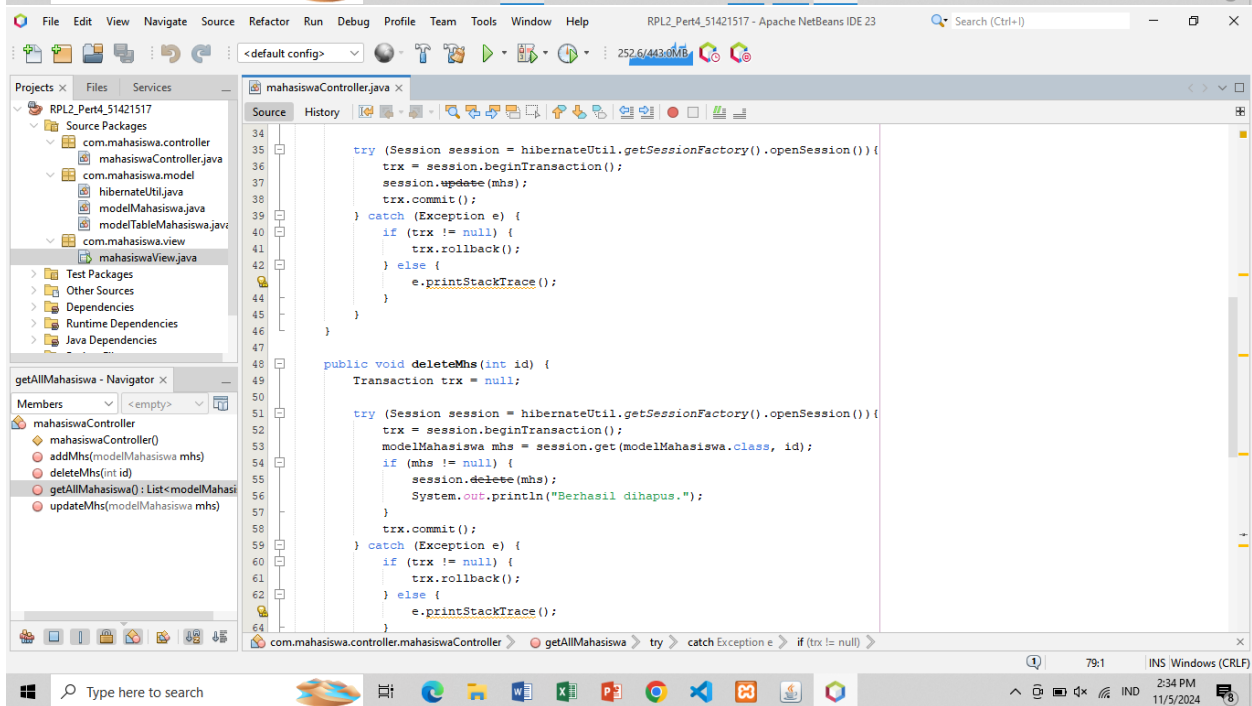
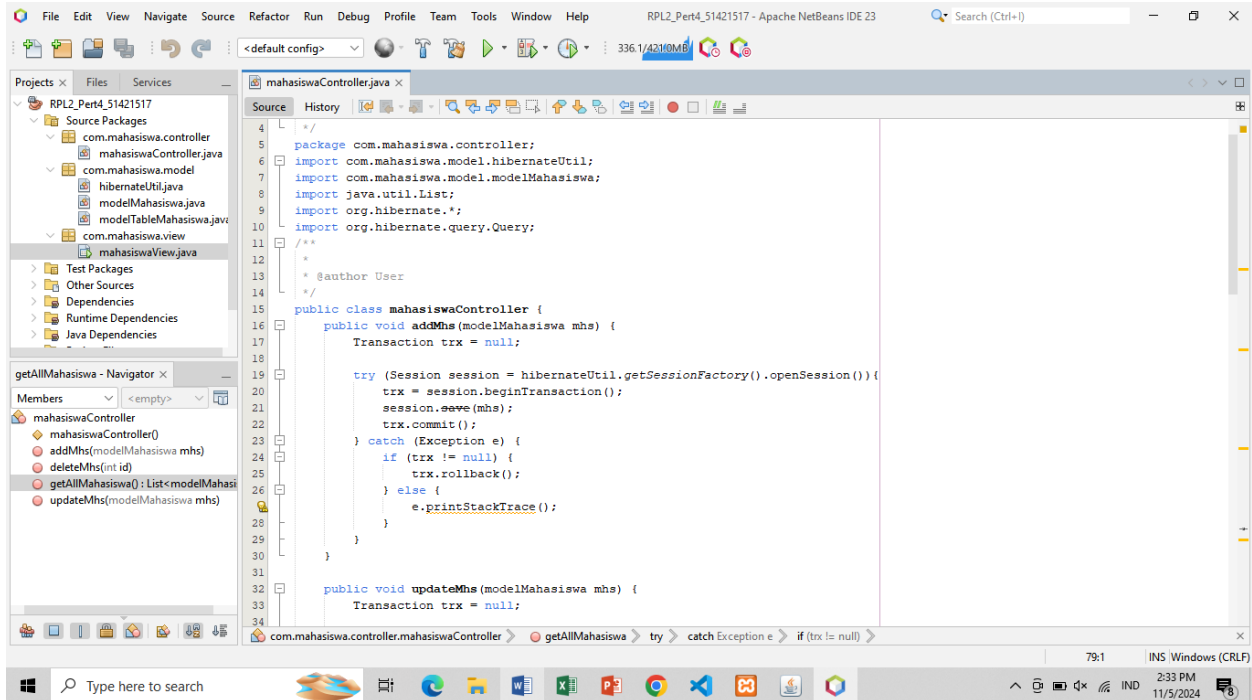


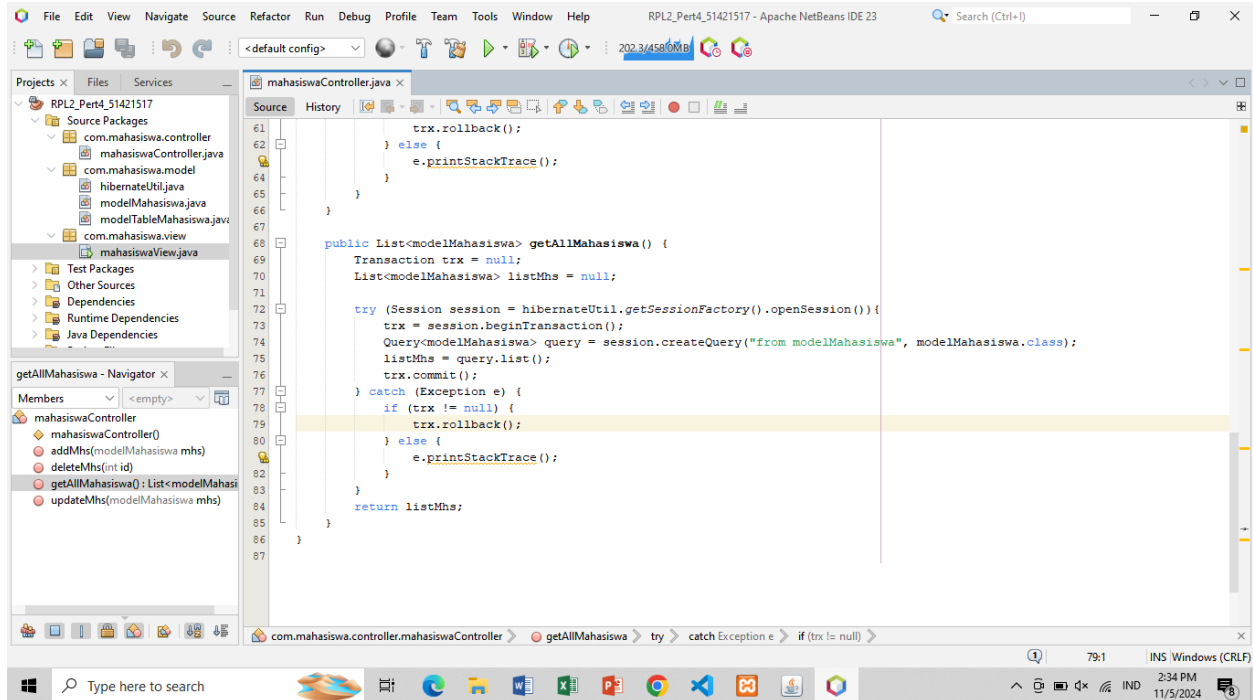
c. modelTableMahasiswa.java :



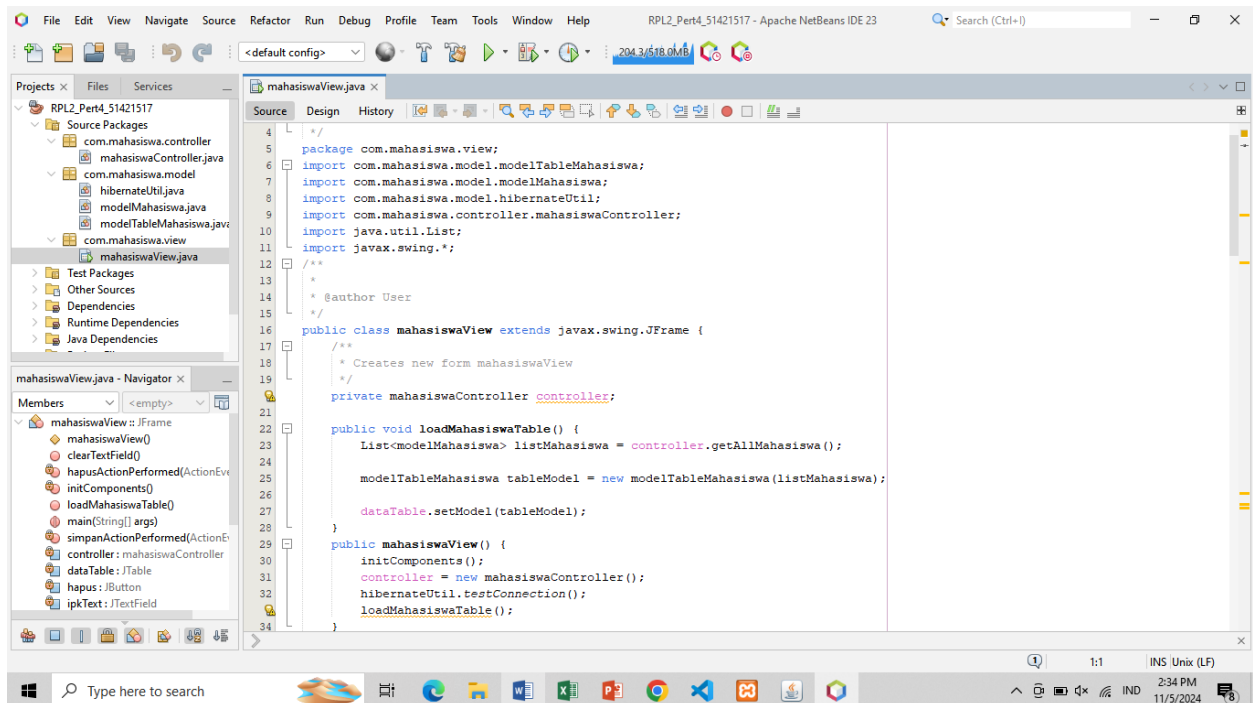


d. mahasiswaController.java :

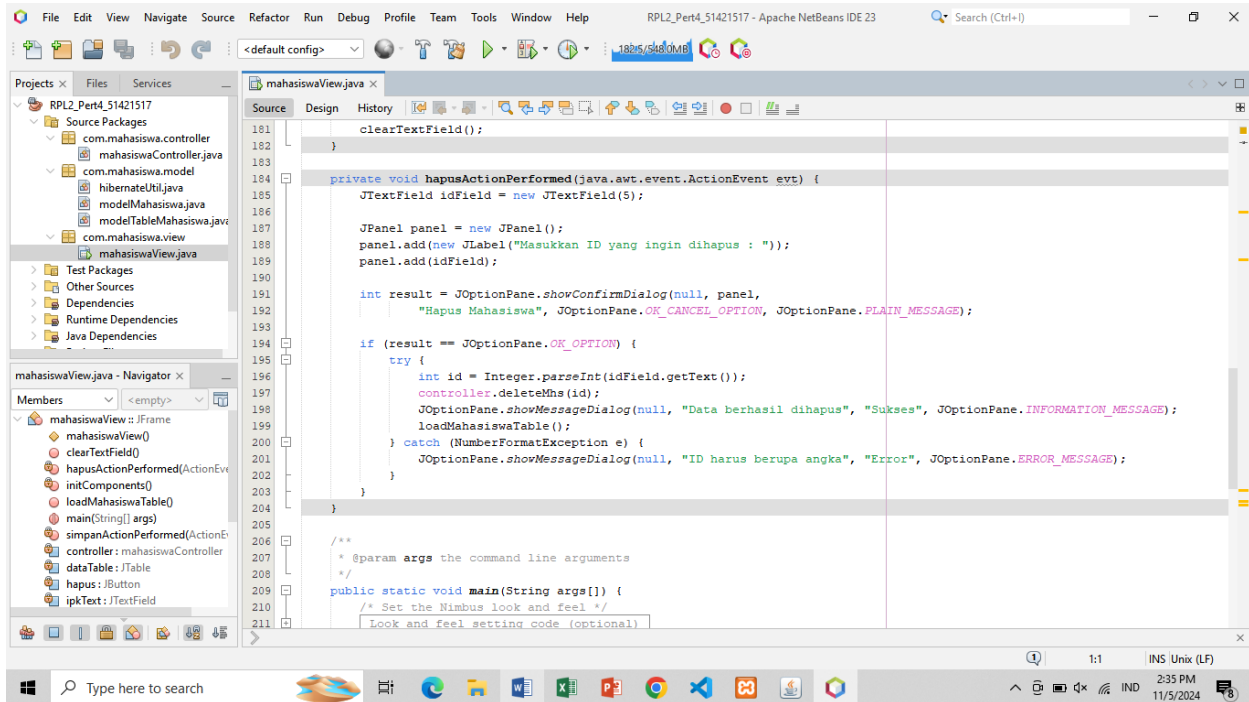
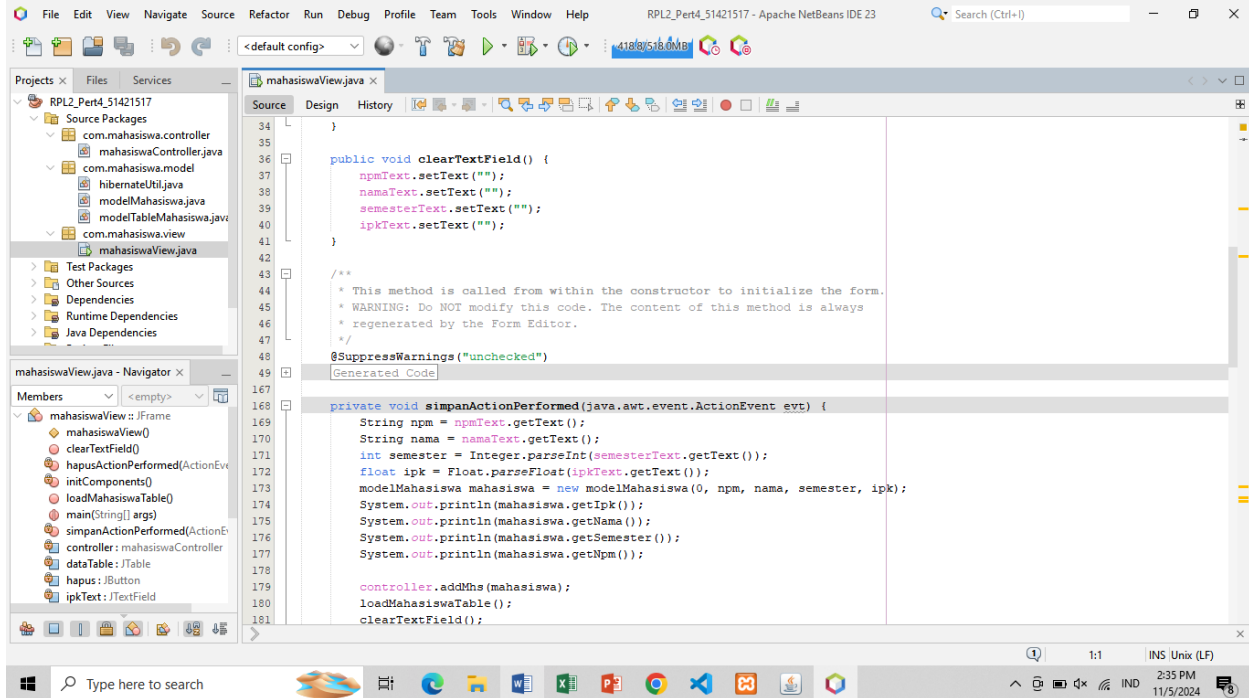


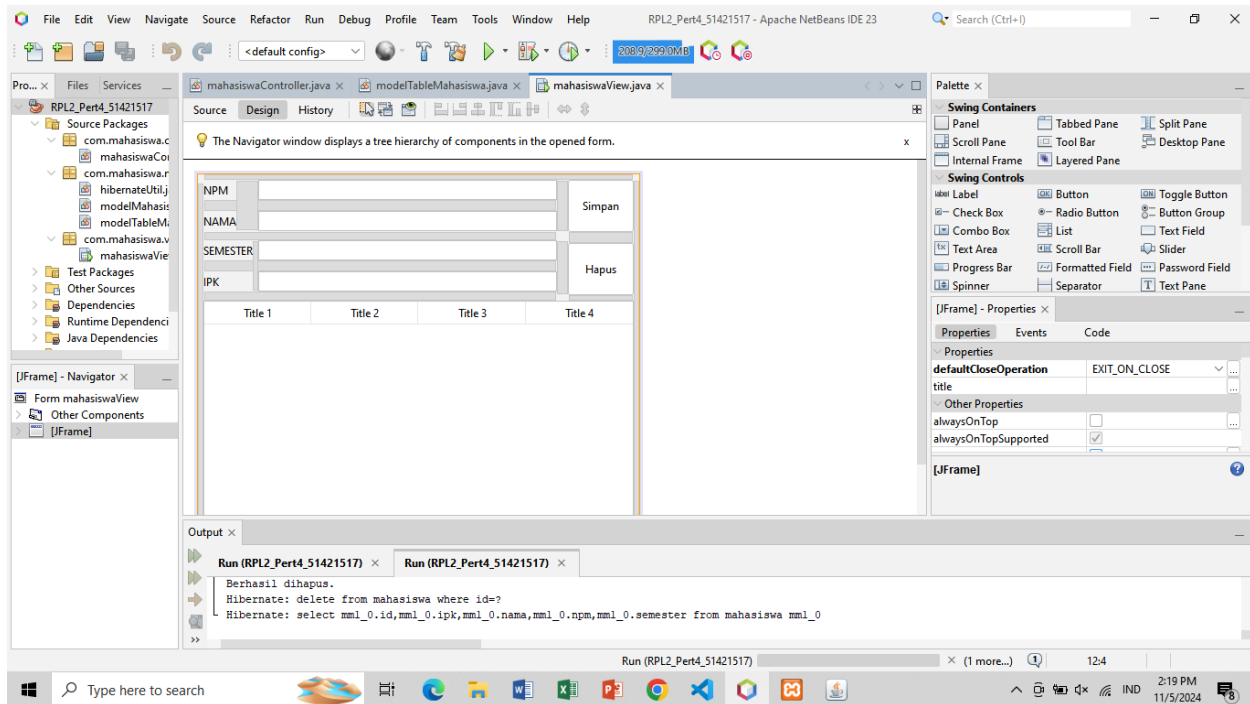
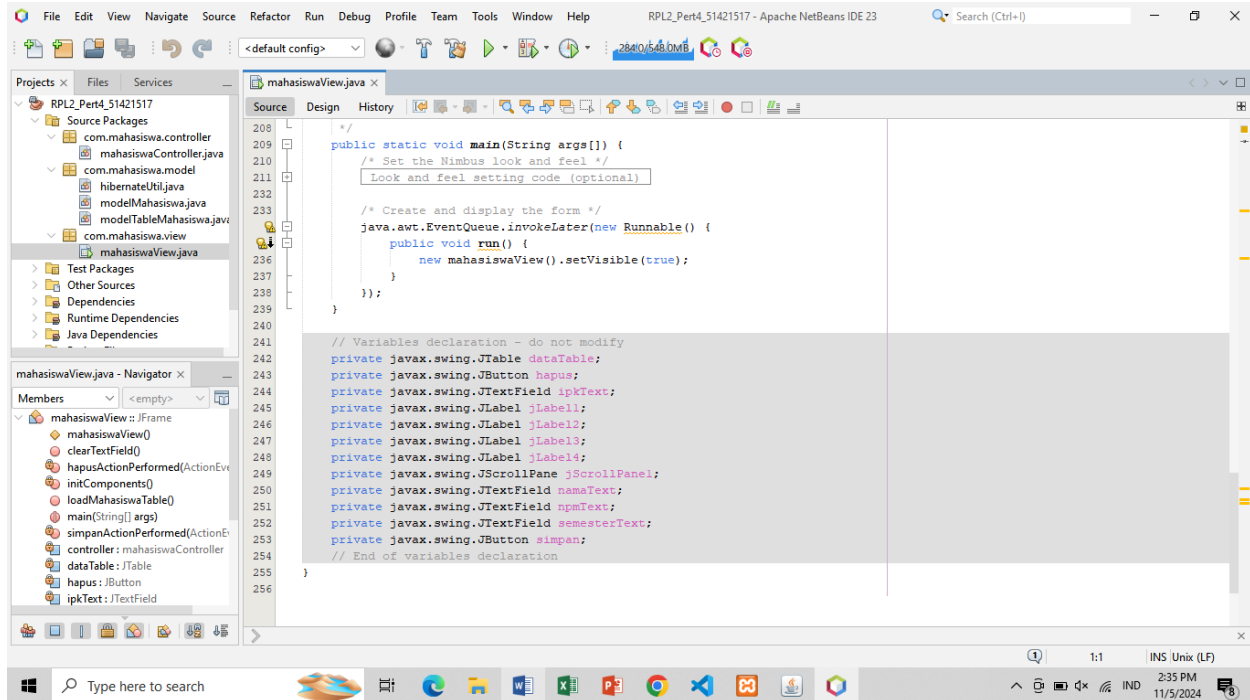


e. mahasiswaView.java :

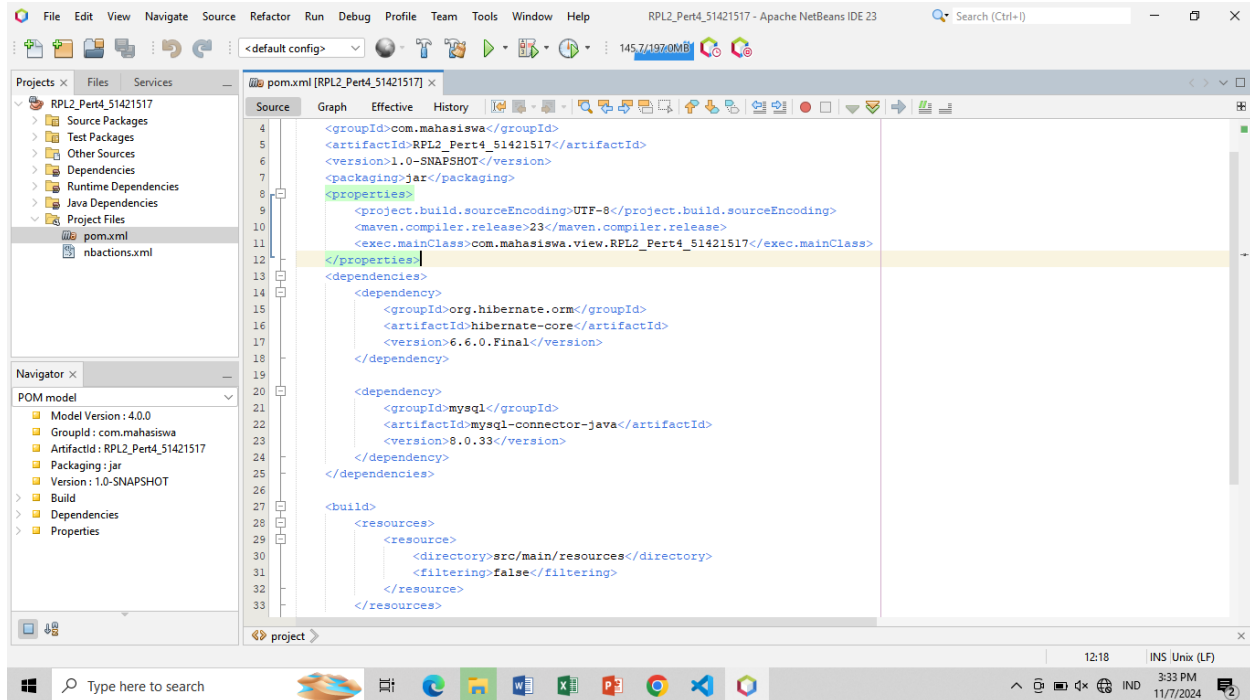




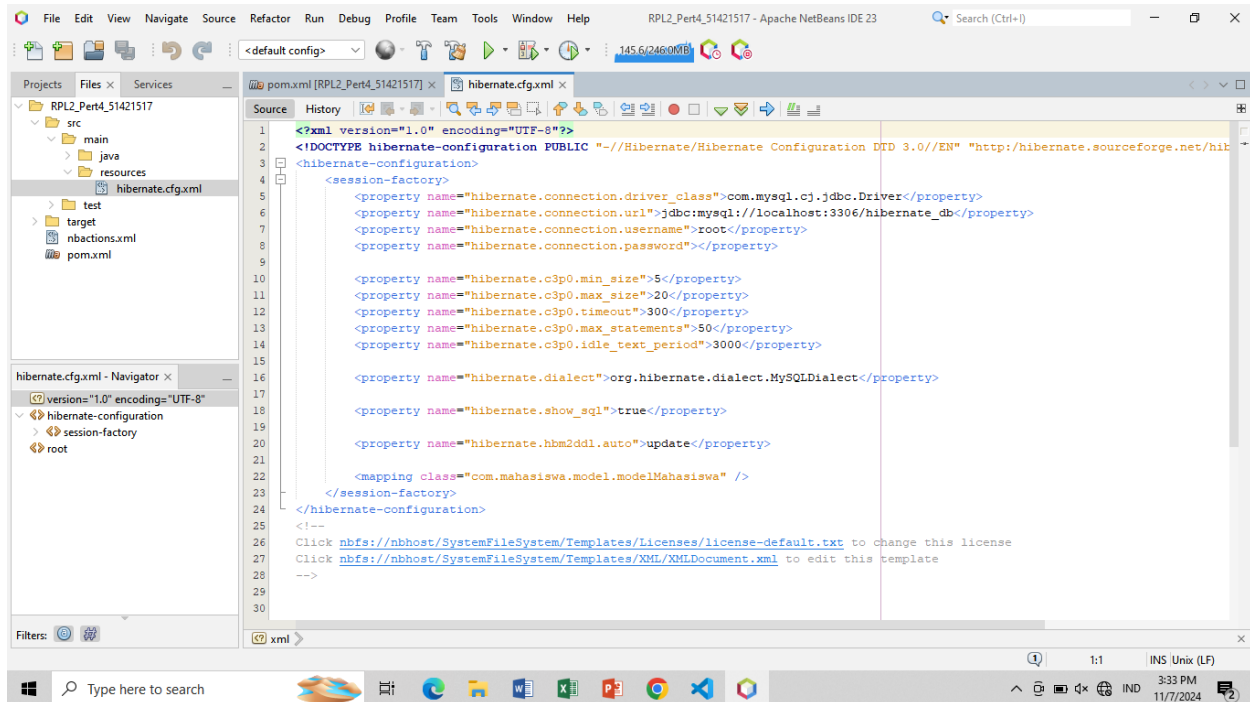




f. pom.xml



## g. hibernate.cfg.xml



## LOGIKA PROGRAM

Setelah membuat database, langkah selanjutnya adalah menuliskan dependencies pada file pom.xml. Dengan file ini, project dapat mengunduh dependencies yang diperlukan agar dapat terhubung ke database. Dependencies yang dibutuhkan adalah mysql connector dan hibernate sebagai ORM, sehingga pengambilan atau manipulasi data dapat dilakukan dengan paradigma pemrograman berbasis objek.

a. Pom.xml

```
<?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_Pert4_51421517</artifactId>

    <version>1.0-SNAPSHOT</version>

    <packaging>jar</packaging>

    <properties>

        <project.build.sourceEncoding>UTF-
8</project.build.sourceEncoding>

        <maven.compiler.release>23</maven.compiler.release>

<exec.mainClass>com.mahasiswa.view.RPL2_Pert4_51421517</exec.mainClass
>

    </properties>

    <dependencies>

        <dependency>

            <groupId>org.hibernate.orm</groupId>

            <artifactId>hibernate-core</artifactId>

            <version>6.6.0.Final</version>

        </dependency>
```

```

        <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>

```

Kemudian, file hibernate.cfg.xml berfungsi untuk mendefinisikan nama database, username, password, dan parameter lain seperti jumlah minimal dan maksimal pool connection dan batas waktu timeout. Melalui file ini, program dapat terhubung ke database sesuai nama database yang diberikan.

```

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

<!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate
Configuration DTD 3.0//EN" "http://hibernate.sourceforge.net/hibernate-
configuration-3.0.dtd">

<hibernate-configuration>
    <session-factory>
        <property
name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</pro
perty>

```

```

        <property
name="hibernate.connection.url">jdbc:mysql://localhost:3306/hibernate_
db</property>

        <property name="hibernate.connection.username">root</property>
        <property name="hibernate.connection.password"></property>

        <property name="hibernate.c3p0.min_size">5</property>
        <property name="hibernate.c3p0.max_size">20</property>
        <property name="hibernate.c3p0.timeout">300</property>
        <property name="hibernate.c3p0.max_statements">50</property>
        <property
name="hibernate.c3p0.idle_text_period">3000</property>

        <property
name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>

        <property name="hibernate.show_sql">true</property>

        <property name="hibernate.hbm2ddl.auto">update</property>

        <mapping class="com.mahasiswa.model.modelMahasiswa" />
    </session-factory>
</hibernate-configuration>

<!--
Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
Click nbfs://nbhost/SystemFileSystem/Templates/XML/XMLDocument.xml to
edit this template
-->

```

<root>

</root>

Class hibernateUtil berfungsi untuk membuka koneksi ke database melalui objek SessionFactory.

b. hibernateUtil.java

```
package com.mahasiswa.model;
import org.hibernate.*;
import org.hibernate.cfg.Configuration;
/**
 *
 * @author User
 */

public class hibernateUtil {
    private static SessionFactory sessionFactory;

    static {
        try {
            sessionFactory = new
Configuration().configure().buildSessionFactory();
        } catch (Throwable ex) {
            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 is
successful.");
    } catch (Exception e) {
        System.err.println("Failed to connect to the database");
        e.printStackTrace();
    }
}
}
}

```

### c. modelMahasiswa.java

Class modelMahasiswa berfungsi untuk mendefinisikan struktur dari objek. Pada class ini, beberapa atribut yang didefinisikan adalah seperti id (yang di auto-increment), npm, nama, semester, dan ipk. Atribut-atribut tersebut akan menjadi field pada database. Untuk kolom npm dan nama, terdapat ketentuan bahwa kedua atribut tersebut tidak boleh memiliki nilai null dan memiliki batas panjang String tertentu.

Karena seluruh atribut objek modelMahasiswa bersifat private, maka diperlukan method getter dan setter untuk mendapatkan dan memodifikasi atribut dari objek.

```

package com.mahasiswa.model;
import jakarta.persistence.*;

```



```
/**
 *
 * @author User
 */

@Entity
@Table (name = "mahasiswa")
public class modelMahasiswa {
    @Id
    @GeneratedValue (strategy = GenerationType.IDENTITY)
    @Column (name = "id")
    private int id;

    @Column (name = "npm", nullable = false, length = 8)
    private String npm;

    @Column (name = "nama", nullable = false, length = 50)
    private String nama;

    @Column (name = "semester")
    private int semester;

    @Column (name = "ipk")
    private float ipk;

    public modelMahasiswa() {

    }
}
```

```
    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;
    }
}

```

#### d. modelTableMahasiswa.java

Class modelTableMahasiswa adalah subclass dari superclass AbstractTableModel, model tabel pada package javax.swing. Class ini berfungsi sebagai blueprint dari model tabel yang akan ditampilkan pada Jtable. Struktur tabel adalah List bertipe data modelMahasiswa. Selain itu, terdapat beberapa method superclass yang di-override oleh class ini untuk menyesuaikan dengan tipe data yang digunakan.

```

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

```

```

import javax.swing.table.AbstractTableModel;

/**
 *
 * @author User
 */
public class modelTableMahasiswa extends AbstractTableModel {
    private List<modelMahasiswa> mahasiswaList;
    private String[] columnNames = {"ID", "NPM", "NAMA", "SEMESTER",
    "IPK"};

    public modelTableMahasiswa(List<modelMahasiswa> mahasiswaList) {
        this.mahasiswaList = mahasiswaList;
    }

    @Override
    public int getRowCount() {
        return mahasiswaList.size();
    }

    @Override
    public int getColumnCount() {
        return columnNames.length;
    }

    @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];
}

```

```

@Override
public boolean isCellEditable(int rowIndex, int columnIndex) {
    return false;
}

```

```

public void setMahasiswaList(List<modelMahasiswa> mahasiswaList) {
    this.mahasiswaList = mahasiswaList;
    fireTableDataChanged();
}

```

```
}  
}
```

e. mahasiswaController.java

Class mahasiswaController berfungsi untuk menyediakan fungsi-fungsi yang dapat melakukan operasi CRUD pada database, seperti menambahkan data mahasiswa, memperbarui, menghapus, dan mengambil data mahasiswa.

```
package com.mahasiswa.controller;  
import com.mahasiswa.model.hibernateUtil;  
import com.mahasiswa.model.modelMahasiswa;  
import java.util.List;  
import org.hibernate.*;  
import org.hibernate.query.Query;  
/**  
 *  
 * @author User  
 */  
public class mahasiswaController {  
    public void addMhs(modelMahasiswa mhs) {  
        Transaction trx = null;  
  
        try (Session session =  
hibernateUtil.getSessionFactory().openSession()){  
            trx = session.beginTransaction();  
            session.save(mhs);  
            trx.commit();  
        } catch (Exception e) {  
            if (trx != null) {
```

```

        trx.rollback();
    } else {
        e.printStackTrace();
    }
}
}

```

```

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();
        } else {
            e.printStackTrace();
        }
    }
}

```

```

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

    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 dihapus.");
        }
        trx.commit();
    } catch (Exception e) {
        if (trx != null) {
            trx.rollback();
        } else {
            e.printStackTrace();
        }
    }
}

public List<modelMahasiswa> getAllMahasiswa() {
    Transaction trx = null;
    List<modelMahasiswa> listMhs = null;

    try (Session session =
hibernateUtil.getSessionFactory().openSession()){
        trx = session.beginTransaction();

        Query<modelMahasiswa> query = session.createQuery("from
modelMahasiswa", modelMahasiswa.class);

        listMhs = query.list();

        trx.commit();
    } catch (Exception e) {

```



```

        if (trx != null) {
            trx.rollback();
        } else {
            e.printStackTrace();
        }
    }
    return listMhs;
}
}

```

#### f. mahasiswaView.java

Class ini berperan sebagai View, yang akan menampilkan program dalam bentuk Java JFrame. Class ini memiliki atribut controller yang merupakan objek mahasiswaController. Beberapa method yang ditambahkan adalah sebagai berikut.

```

public void loadMahasiswaTable() {
    List<modelMahasiswa> listMahasiswa =
    controller.getAllMahasiswa();

    modelTableMahasiswa tableModel = new
    modelTableMahasiswa(listMahasiswa);

    dataTable.setModel(tableModel);
}

```

Method ini berfungsi untuk membaca seluruh data mahasiswa dari tabel mahasiswa. Hal tersebut dapat dilakukan dengan memanggil method getAllMahasiswa dari objek controller. Kemudian, struktur tabel didefinisikan menggunakan List bertipe data modelMahasiswa. Struktur tabel ini dijadikan sebagai argumen yang diberikan ke method setModel milik objek dataTable (Jtable).

```

public mahasiswaView() {
    initComponents();
}

```

```

        controller = new mahasiswaController();
        hibernateUtil.testConnection();
        loadMahasiswaTable();
    }

```

Constructor mahasiswaView akan menginisialisasi komponen widget yang digunakan pada JFrame, menginisiasi controller, menguji koneksi ke database dengan memanggil method testConnection milik class hibernateUtil, dan memuat tabel mahasiswa dengan memanggil method loadMahasiswaTable.

```

public void clearTextField() {
    npmText.setText("");
    namaText.setText("");
    semesterText.setText("");
    ipkText.setText("");
}

```

Method clearTextField berfungsi untuk mengosongkan text field untuk input data mahasiswa.

```

private void simpanActionPerformed(java.awt.event.ActionEvent evt) {
    String npm = npmText.getText();
    String nama = namaText.getText();
    int semester = Integer.parseInt(semesterText.getText());
    float ipk = Float.parseFloat(ipkText.getText());
    modelMahasiswa mahasiswa = new modelMahasiswa(0, npm, nama,
semester, ipk);
    System.out.println(mahasiswa.getIpk());
    System.out.println(mahasiswa.getNama());
    System.out.println(mahasiswa.getSemester());
    System.out.println(mahasiswa.getNpm());
}

```

```

        controller.addMhs(mahasiswa);

        loadMahasiswaTable();

        clearTextField();
    }

```

Method `simpanActionPerformed` akan dieksekusi ketika tombol Simpan diklik. Hal yang dilakukan pertama-tama adalah mendapatkan data input mahasiswa, kemudian membuat objek `modelMahasiswa` baru dengan parameter-parameter tersebut. Data tersebut kemudian ditampilkan pada terminal. Objek `controller` kemudian menambahkan data tersebut ke database dengan memanggil method `addMhs`. Kemudian, tabel dimuat ulang, dan text field dikosongkan kembali.

```

private void hapusActionPerformed(java.awt.event.ActionEvent evt) {
    JTextField idField = new JTextField(5);

    JPanel panel = new JPanel();
    panel.add(new JLabel("Masukkan ID yang ingin dihapus : "));
    panel.add(idField);

    int result = JOptionPane.showConfirmDialog(null, panel,
        "Hapus Mahasiswa", JOptionPane.OK_CANCEL_OPTION,
        JOptionPane.PLAIN_MESSAGE);

    if (result == JOptionPane.OK_OPTION) {
        try {
            int id = Integer.parseInt(idField.getText());
            controller.deleteMhs(id);

            JOptionPane.showMessageDialog(null, "Data berhasil
dihapus", "Sukses", JOptionPane.INFORMATION_MESSAGE);

            loadMahasiswaTable();
        } catch (NumberFormatException e) {

```

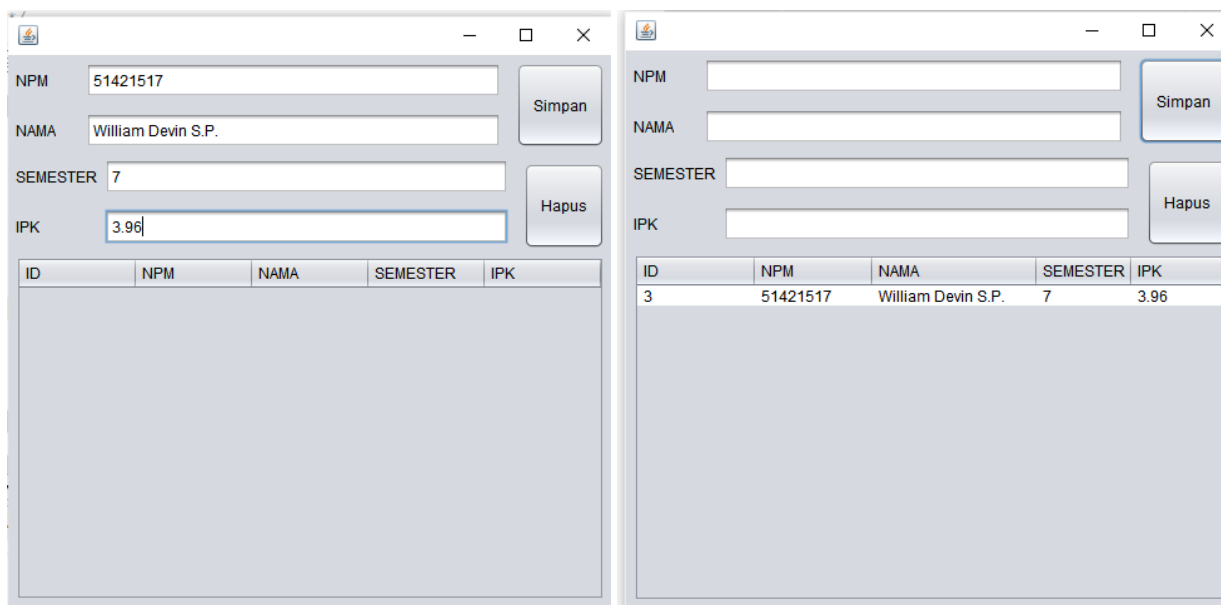
```

        JOptionPane.showMessageDialog(null, "ID harus berupa
angka", "Error", JOptionPane.ERROR_MESSAGE);
    }
}
}

```

Method `hapusActionPerformed` akan dieksekusi ketika tombol Hapus diklik. Ketika tombol tersebut diklik, maka program akan menampilkan Jpanel baru untuk meminta input ID mahasiswa yang ingin dihapus. Jika input berhasil dikonversi ke dalam tipe data integer, maka data dengan ID tersebut akan dihapus dari database dengan memanggil method `deleteMhs` milik objek controller. Setelah berhasil dihapus dan menampilkan dialog konfirmasi, tabel akan dimuat ulang.

## OUTPUT PROGRAM



The image displays two side-by-side screenshots of a Java Swing application window. The window has a title bar with standard Windows controls (minimize, maximize, close). The main area contains a form with four input fields: NPM, NAMA, SEMESTER, and IPK. To the right of the NPM and NAMA fields is a 'Simpan' (Save) button. To the right of the SEMESTER and IPK fields is a 'Hapus' (Delete) button. Below the form is a table with five columns: ID, NPM, NAMA, SEMESTER, and IPK. The table contains one row of data: ID 3, NPM 51421517, NAMA William Devin S.P., SEMESTER 7, and IPK 3.96. The left screenshot shows the 'Simpan' button being clicked, and the right screenshot shows the 'Hapus' button being clicked.

ID	NPM	NAMA	SEMESTER	IPK
3	51421517	William Devin S.P.	7	3.96

Formulir input data mahasiswa:

NPM:

NAMA:

SEMESTER:

IPK:

Simpan

Hapus

ID	NPM	NAMA	SEMESTER	IPK
3	51421517	William Devin S.P.	7	3.96

Hapus Mahasiswa

Masukkan ID yang ingin dihapus :

OK Cancel

Formulir input data mahasiswa:

NPM:

NAMA:

SEMESTER:

IPK:

Simpan

Hapus

ID	NPM	NAMA	SEMESTER	IPK
3	51421517	William Devi...	7	3.96

Sukses

Data berhasil dihapus

OK

Formulir input data mahasiswa:

NPM:

NAMA:

SEMESTER:

IPK:

Simpan

Hapus

ID	NPM	NAMA	SEMESTER	IPK
----	-----	------	----------	-----