LAPORAN AKHIR PRAKTIKUM

Mata Praktikum : Rekayasa Perangkat Lunak 2

Kelas : 4IA06

Praktikum ke- : 6

Tanggal : 19 November 2024

Materi : Implementasi Aspect Oriented Programming (AOP) dan Dependency injection pada Project Spring dan Hibernate.

NPM : 51421055

Nama : Muhammad Ridho

Ketua Asisten : Gilbert Jefferson Faozato Mendrofa

Paraf Asisten :

Nama Asisten :

Jumlah Lembar : 24 Lembar

LABORATORIUM TEKNIK INFORMATIKA
UNIVERSITAS GUNADARMA
2024

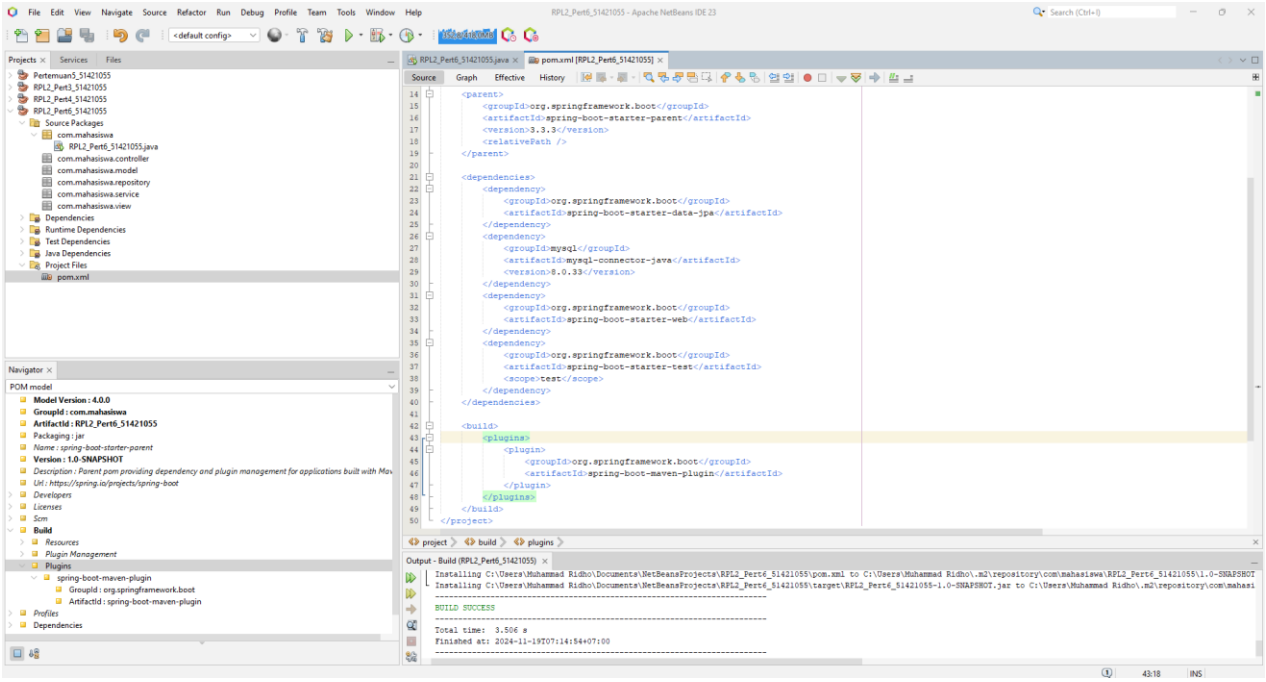
Soal:

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

Jawaban:

1.

Pom.xml



Sourcecode:

<?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_Pert6_51421055</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>

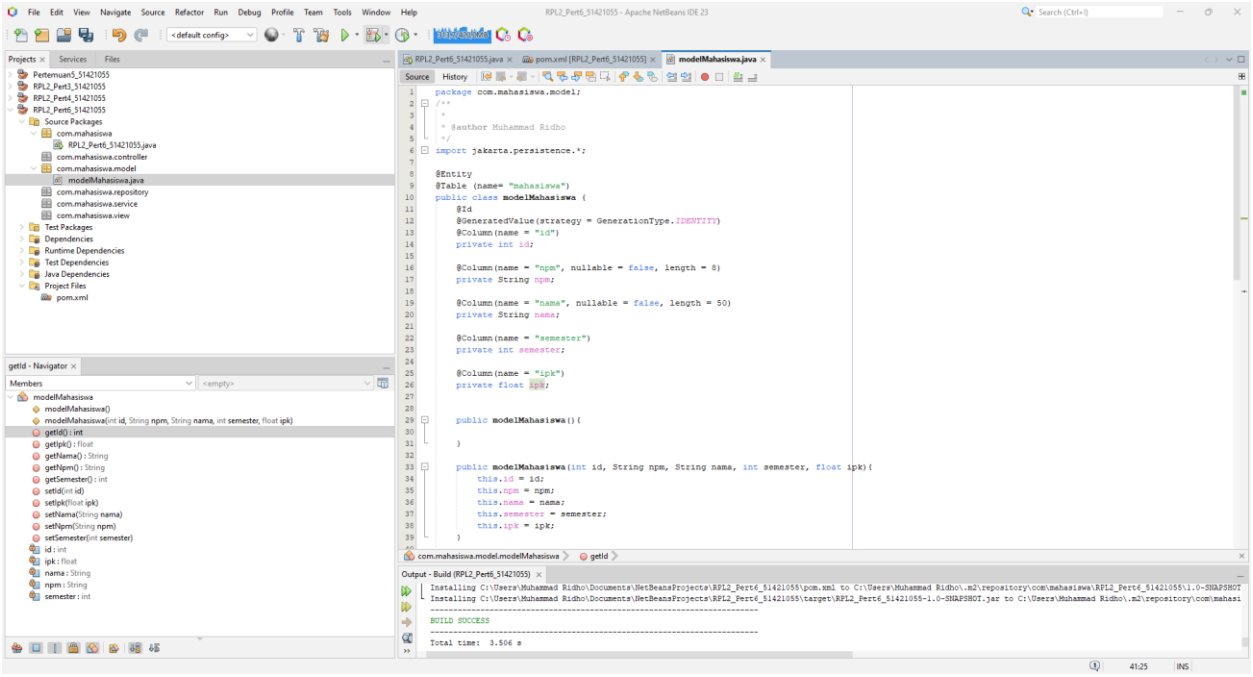
```
    <exec.mainClass>com.mahasiswa.mahasiswaApp</exec.mainClass>
</properties>

<parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.3.3</version>
    <relativePath />
</parent>

<dependencies>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.33</version>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>
```

```
    <build>
        <plugins>
            <plugin>
                <groupId>org.springframework.boot</groupId>
                <artifactId>spring-boot-maven-plugin</artifactId>
            </plugin>
        </plugins>
    </build>
</project>
```

Penjelasan:

Kode di atas adalah file konfigurasi Maven (**pom.xml**) untuk proyek Java berbasis Spring Boot. Proyek ini menggunakan Spring Boot 3.3.3 sebagai framework utama dengan dependensi untuk Spring Data JPA (untuk pengelolaan database), MySQL Connector (untuk koneksi ke database MySQL), dan Spring Web (untuk pengembangan aplikasi berbasis web). Proyek dikemas sebagai file JAR, dengan target JDK 23 melalui properti maven.compiler.release. Class utama aplikasi didefinisikan sebagai com.mahasiswa.mahasiswaApp. Plugin spring-boot-maven-plugin digunakan untuk menjalankan aplikasi dengan perintah Maven seperti mvn spring-boot:run.

ModelMahasiswa.java



Sourcecode:

```
package com.mahasiswa.model;
```

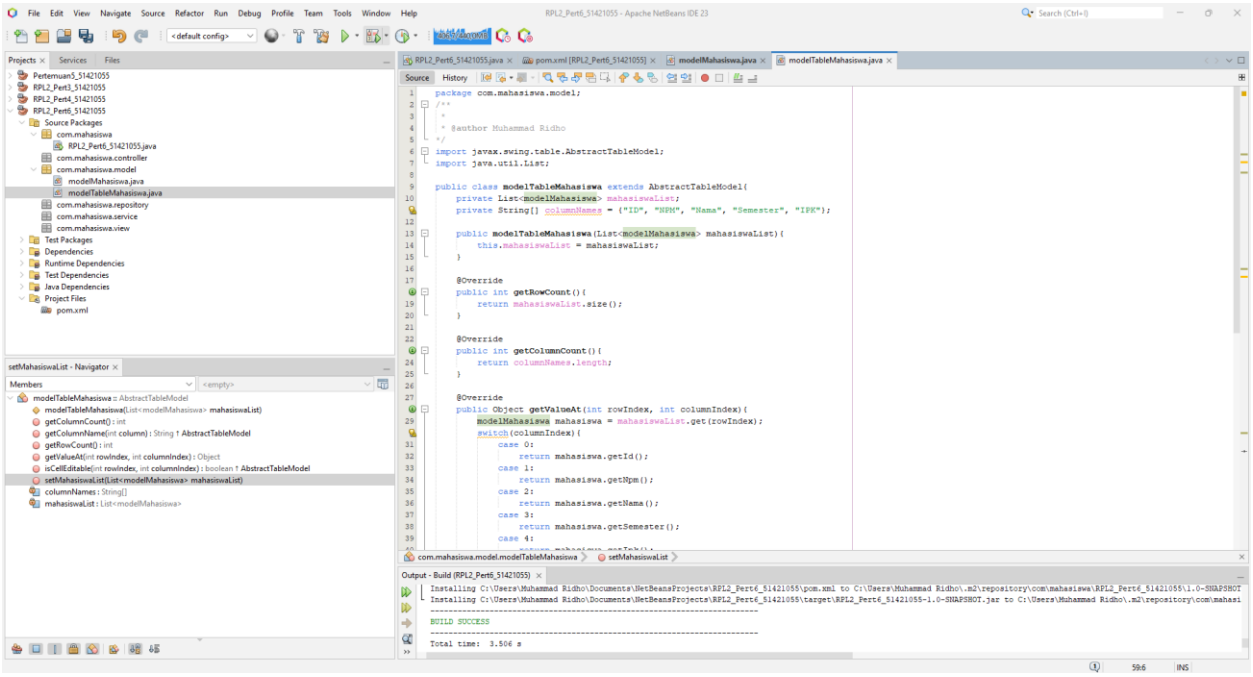
```
/**
 *
 * @author Muhammad Ridho
 */
import jakarta.persistence.*;

@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;
    }

}
```

Penjelasan:
Kode di atas mendefinisikan class **modelMahasiswa** sebagai entitas JPA untuk tabel database bernama **mahasiswa**. Setiap kolom pada tabel direpresentasikan sebagai atribut class: id (primary key dengan auto increment), npm (nomor mahasiswa), nama, semester, dan ipk. Class ini memiliki konstruktor default dan konstruktor parameter untuk inisialisasi data, serta getter dan setter untuk setiap atribut, memungkinkan manipulasi data dengan mudah. Anotasi seperti @Entity, @Table, @Id, dan @Column digunakan untuk memetakan class dan atributnya ke struktur tabel di database.

ModelTableMahasiswa.java



Sourcecode:

```
package com.mahasiswa.model;
/**
 *
 * @author Muhammad Ridho
*/
import javax.swing.table.AbstractTableModel;
import java.util.List;

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
```

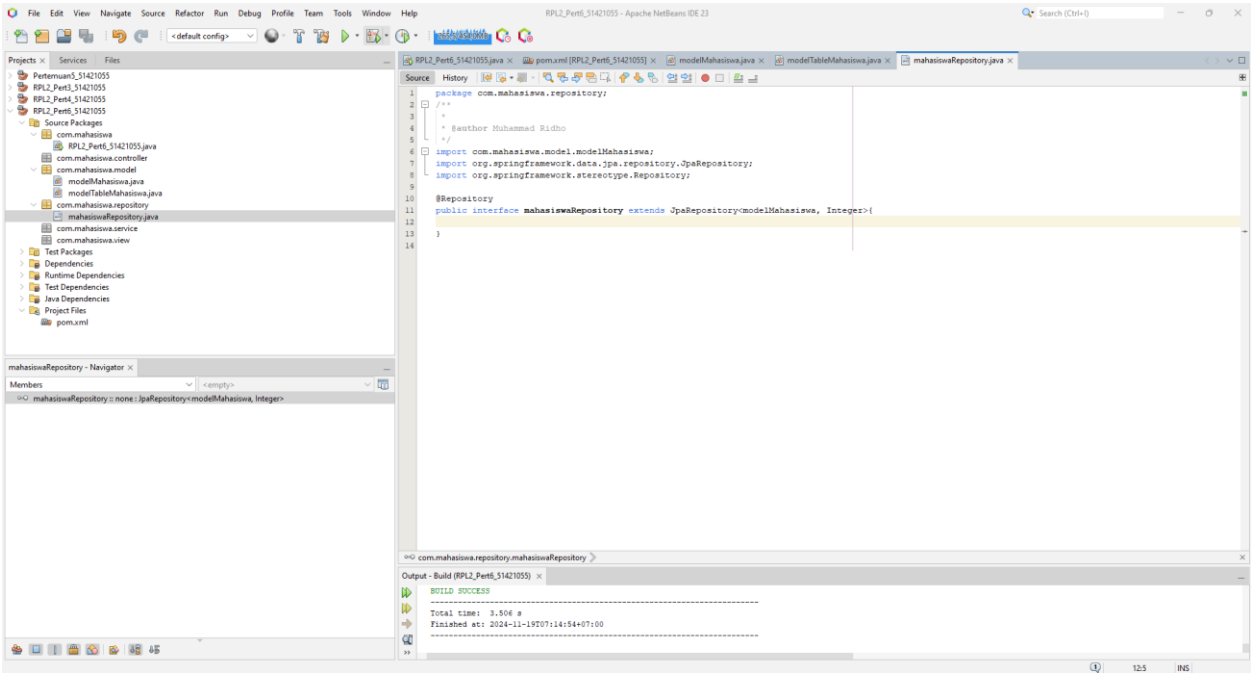
```
public int getRowCount(){
    return mahasiswaList.size();
}

@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();
    }
}
```

Penjelasan:

Kode di atas mendefinisikan class **modelTableMahasiswa** sebagai implementasi tabel berbasis **Swing** dengan mewarisi **AbstractTableModel**. Class ini berfungsi untuk menampilkan daftar objek **modelMahasiswa** dalam bentuk tabel dengan kolom: **ID**, **NPM**, **Nama**, **Semester**, dan **IPK**. Method getRowCount dan getColumnCount menentukan jumlah baris dan kolom tabel, sedangkan getValueAt mengembalikan data pada sel tertentu. Nama kolom diatur menggunakan getColumnName, dan semua sel ditandai tidak dapat diedit melalui isCellEditable. Selain itu, terdapat Method setMahasiswaList untuk memperbarui data tabel dan memberitahu model bahwa data telah berubah menggunakan fireTableDataChanged().

mahasiswaRepository.java

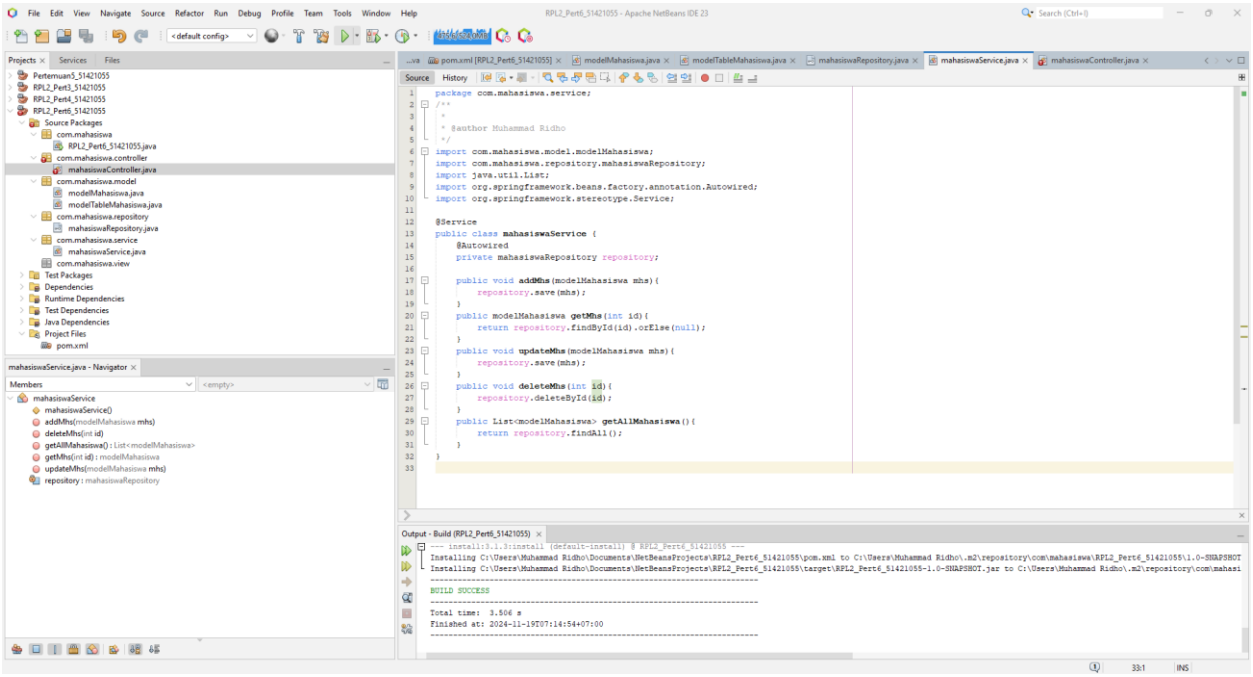


Sourcecode:

```
package com.mahasiswa.repository;

/**
 *
 * @author Muhammad Ridho
 */
import com.mahasiswa.model.modelMahasiswa;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;
@Repository
public interface mahasiswaRepository extends JpaRepository<modelMahasiswa, Integer>{


}
```

Penjelasan:
Kode di atas mendefinisikan antarmuka **mahasiswaRepository** yang merupakan **repository** untuk entitas **modelMahasiswa**, menggunakan **Spring Data JPA**. Antarmuka ini mewarisi **JpaRepository**, yang menyediakan Method bawaan untuk operasi CRUD (Create, Read, Update, Delete) dan manajemen data berbasis database. Anotasi **@Repository** menandai antarmuka ini sebagai komponen Spring untuk interaksi dengan database, dengan primary key bertipe **Integer**.

mahasiswaService.java



Sourcecode:

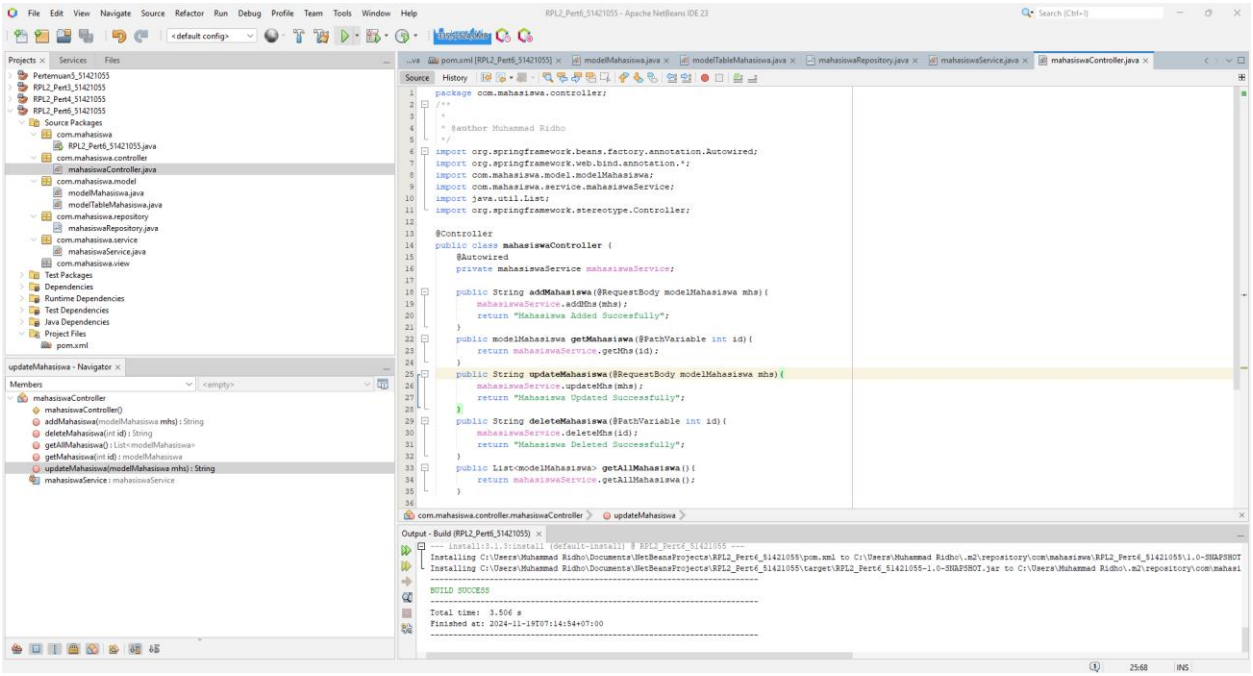
```
package com.mahasiswa.service;

/**
 *
 * @author Muhammad Ridho
 */
import com.mahasiswa.model.modelMahasiswa;
import com.mahasiswa.repository.mahasiswaRepository;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;


@Service
public class mahasiswaService {
    @Autowired
    private mahasiswaRepository repository;


    public void addMhs(modelMahasiswa mhs){
```

```
        repository.save(mhs);
    }
    public modelMahasiswa getMhs(int id){
        return repository.findById(id).orElse(null);
    }
    public void updateMhs(modelMahasiswa mhs){
        repository.save(mhs);
    }
    public void deleteMhs(int id){
        repository.deleteById(id);
    }
    public List<modelMahasiswa> getAllMahasiswa(){
        return repository.findAll();
    }
}
```

Penjelasan:

Kode di atas mendefinisikan class **mahasiswaService** sebagai layanan untuk mengelola operasi terkait entitas **modelMahasiswa**, menggunakan **Spring Framework**. Dengan anotasi **@Service**, class ini dikelola oleh Spring sebagai komponen logika bisnis. Menggunakan dependensi **mahasiswaRepository** yang disuntikkan melalui anotasi **@Autowired**, class ini menyediakan Method untuk: menambahkan data mahasiswa (addMhs), mengambil data berdasarkan ID (getMhs), memperbarui data (updateMhs), menghapus data berdasarkan ID (deleteMhs), dan mendapatkan semua data mahasiswa (getAllMahasiswa). Operasi ini memanfaatkan Method bawaan dari **JpaRepository**.

mahasiswaController.java



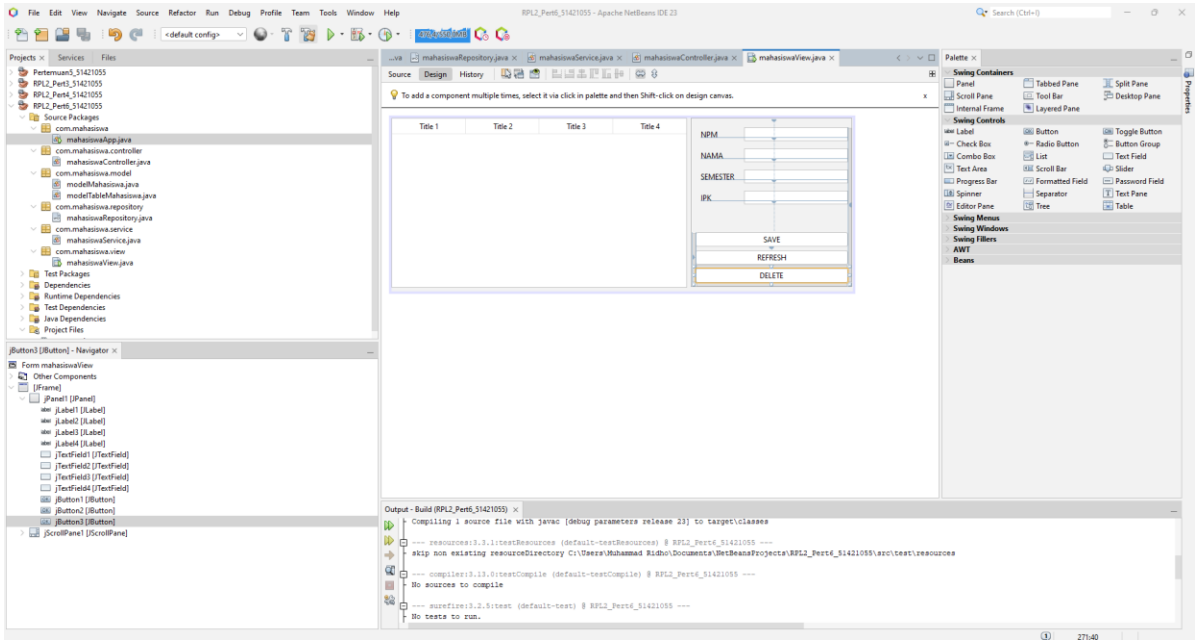
Sourcecode:

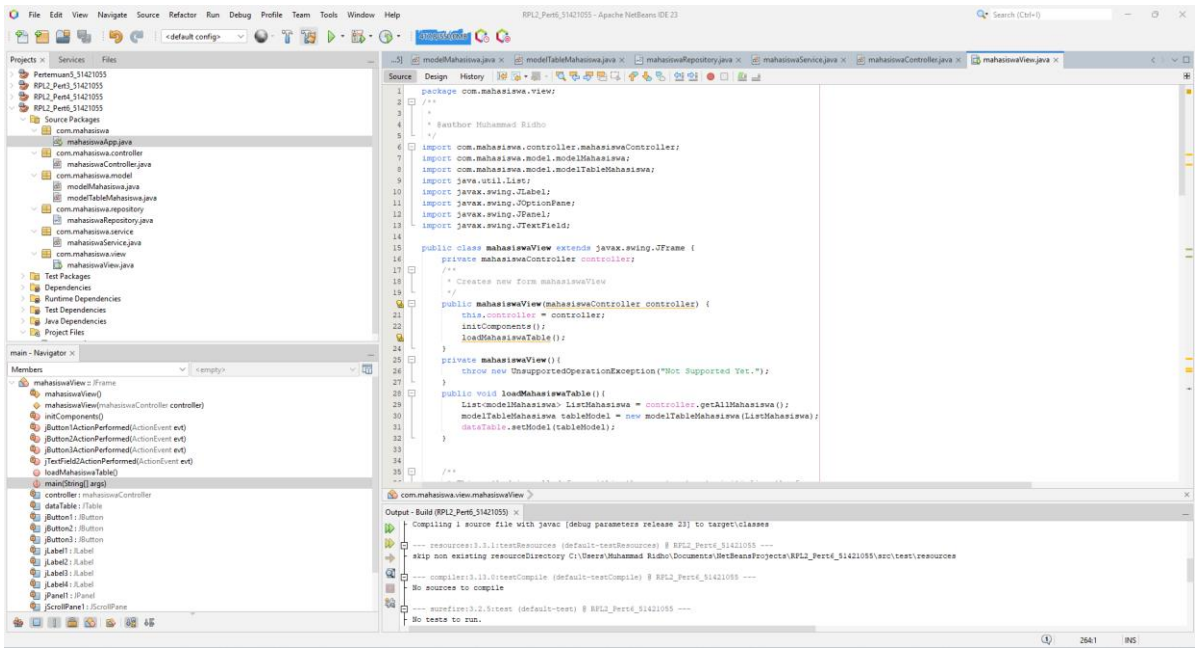
```
package com.mahasiswa.controller;

/**
 *
 * @author Muhammad Ridho
 */
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import com.mahasiswa.model.modelMahasiswa;
import com.mahasiswa.service.mahasiswaService;
import java.util.List;
import org.springframework.stereotype.Controller;


@Controller
public class mahasiswaController {
    @Autowired
    private mahasiswaService mahasiswaService;
```

```
    public String addMahasiswa(@RequestBody modelMahasiswa mhs){
        mahasiswaService.addMhs(mhs);
        return "Mahasiswa Added Succesfully";
    }
    public modelMahasiswa getMahasiswa(@PathVariable int id){
        return mahasiswaService.getMhs(id);
    }
    public String updateMahasiswa(@RequestBody modelMahasiswa mhs){
        mahasiswaService.updateMhs(mhs);
        return "Mahasiswa Updated Successfully";
    }
    public String deleteMahasiswa(@PathVariable int id){
        mahasiswaService.deleteMhs(id);
        return "Mahasiswa Deleted Successfully";
    }
    public List<modelMahasiswa> getAllMahasiswa(){
        return mahasiswaService.getAllMahasiswa();
    }

}
```

Penjelasan:
Kode di atas mendefinisikan class **mahasiswaController** sebagai pengendali (controller) dalam aplikasi berbasis **Spring MVC**. Dengan anotasi **@Controller**, class ini menangani permintaan HTTP yang terkait dengan entitas **modelMahasiswa**. Class ini memanfaatkan layanan **mahasiswaService** yang disuntikkan menggunakan **@Autowired**. Method-Method di dalamnya meliputi: menambahkan data mahasiswa (addMahasiswa), mengambil data mahasiswa berdasarkan ID (getMahasiswa), memperbarui data (updateMahasiswa), menghapus data (deleteMahasiswa), dan mendapatkan semua data mahasiswa (getAllMahasiswa). Setiap Method menggunakan layanan untuk memproses data dan memberikan respons sederhana sebagai hasil operasi.

mahasiswaView.java





Sourcecode:

```
package com.mahasiswa.view;
/**
 *
 * @author Muhammad Ridho
 */
import com.mahasiswa.controller.mahasiswaController;
import com.mahasiswa.model.modelMahasiswa;
import com.mahasiswa.model.modelTableMahasiswa;
```

```
import com.mahasiswa.model.modelTableMahasiswa;
import java.util.List;
import javax.swing.JLabel;
import javax.swing.JOptionPane;
import javax.swing.JPanel;
import javax.swing.JTextField;

public class mahasiswaView extends javax.swing.JFrame {
    private mahasiswaController controller;
    /**
     * Creates new form mahasiswaView
     */
    public mahasiswaView(mahasiswaController controller) {
        this.controller = controller;
        initComponents();
        loadMahasiswaTable();
    }
    private mahasiswaView(){
        throw new UnsupportedOperationException("Not Supported Yet.");
    }
    public void loadMahasiswaTable(){
        List<modelMahasiswa> ListMahasiswa = controller.getAllMahasiswa();
        modelTableMahasiswa tableModel = new modelTableMahasiswa(ListMahasiswa);
        dataTable.setModel(tableModel);
    }
        @SuppressWarnings("unchecked")

    private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
        loadMahasiswaTable();
    }
```

```
    private void jTextField2ActionPerformed(java.awt.event.ActionEvent evt) {

    }

    private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
        String npm = jTextField1.getText();
        String nama = jTextField2.getText();
        int semester = Integer.parseInt(jTextField3.getText());
        float ipk = Float.parseFloat(jTextField4.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.addMahasiswa(mahasiswa);
        loadMahasiswaTable();
    }

    private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
        JTextField idField = new JTextField(10);

        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.deleteMahasiswa(id);
            JOptionPane.showMessageDialog(null, "Data Berhasil dihapus", "Sukses",
JOptionPane.INFORMATION_MESSAGE);
        }catch (NumberFormatException e){
            JOptionPane.showMessageDialog(null,"ID harus berupa angka" , "Error",
JOptionPane.ERROR_MESSAGE);
        }
    }
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    /* Set the Nimbus look and feel */
    //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
    /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
     * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
     */
    try {
        for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
            if ("Nimbus".equals(info.getName())) {
                javax.swing.UIManager.setLookAndFeel(info.getClassName());
                break;
            }
        }
    } catch (ClassNotFoundException ex) {

java.util.logging.Logger.getLogger(mahasiswaView.class.getName()).log(java.util.logging.Level.SEVE
RE, null, ex);
    } catch (InstantiationException ex) {
```

```
java.util.logging.Logger.getLogger(mahasiswaView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (IllegalAccessException ex) {
java.util.logging.Logger.getLogger(mahasiswaView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        } catch (javax.swing.UnsupportedLookAndFeelException ex) {
java.util.logging.Logger.getLogger(mahasiswaView.class.getName()).log(java.util.logging.Level.SEVERE, null, ex);
        }
        //</editor-fold>

        /* Create and display the form */
        java.awt.EventQueue.invokeLater(new Runnable() {
            public void run() {
                new mahasiswaView().setVisible(true);
            }
        });
    }
```

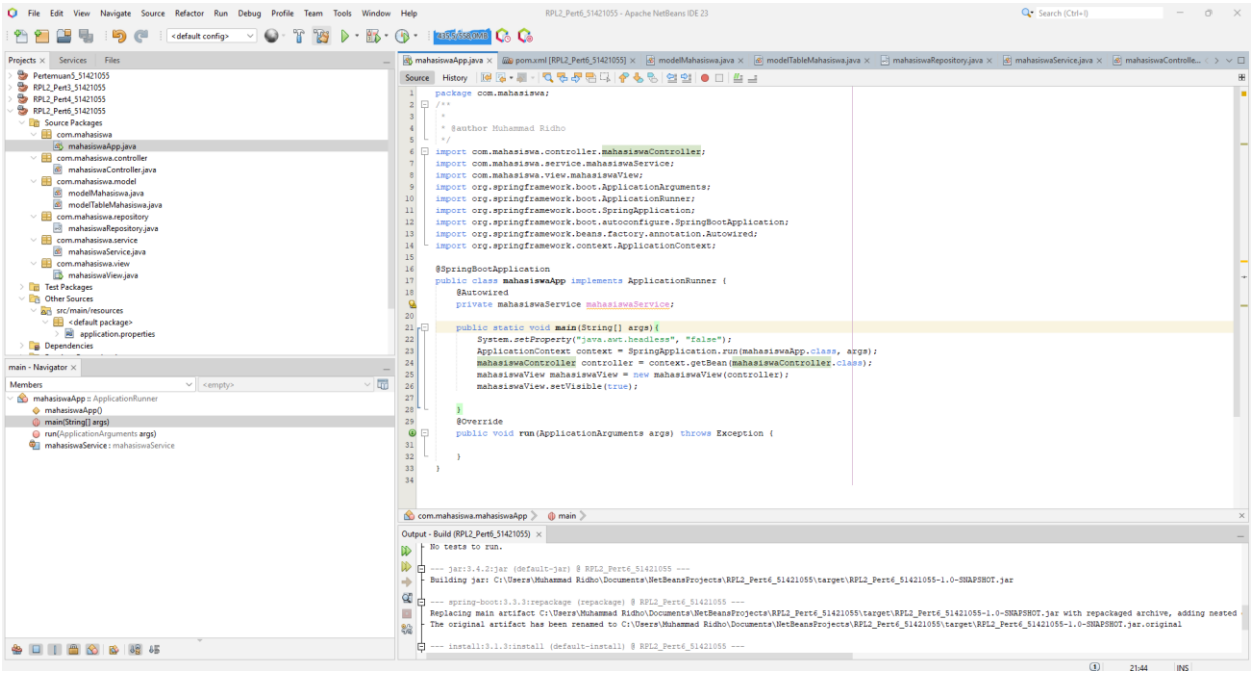
Penjelasan:
Kode di atas mendefinisikan class **mahasiswaView** sebagai antarmuka grafis (GUI) berbasis **Swing** untuk mengelola data mahasiswa. Class ini memanfaatkan **mahasiswaController** untuk berinteraksi dengan data dan melakukan operasi CRUD. Tabel mahasiswa ditampilkan dengan menggunakan model **modelTableMahasiswa**.

Pengguna dapat:

1. **Memuat data** ke tabel dengan Method loadMahasiswaTable.
2. **Menambahkan data** melalui form input yang mengirimkan data ke controller menggunakan tombol aksi jButton1.
3. **Menghapus data** berdasarkan ID yang dimasukkan dalam dialog menggunakan tombol jButton3.

Kode juga mencakup Method main menjalankan aplikasi dengan membuat instans **mahasiswaView**. Implementasi GUI ini memungkinkan pengguna mengelola data mahasiswa dengan mudah dalam aplikasi berbasis desktop.

mahasiswaApp.java



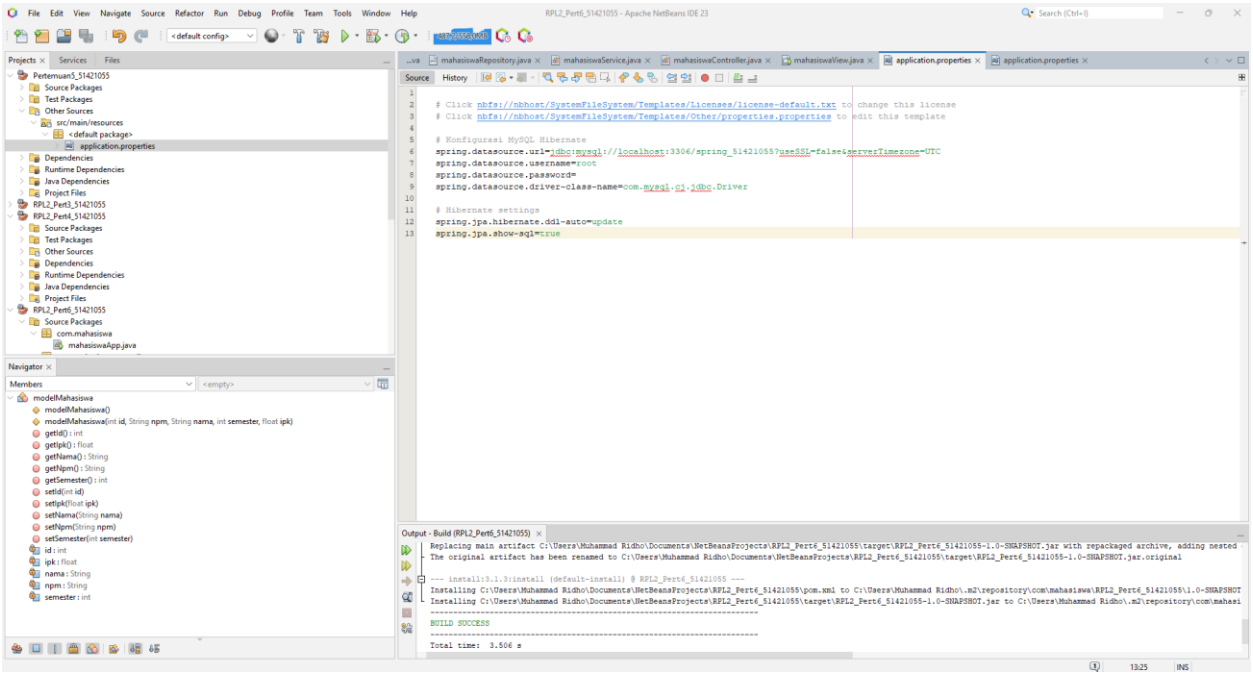
Sourcecode:

```
package com.mahasiswa;

/**

 *

 * @author Muhammad Ridho

 */

import com.mahasiswa.controller.mahasiswaController;

import com.mahasiswa.service.mahasiswaService;

import com.mahasiswa.view.mahasiswaView;

import org.springframework.boot.ApplicationArguments;

import org.springframework.boot.ApplicationRunner;

import org.springframework.boot.SpringApplication;

import org.springframework.boot.autoconfigure.SpringBootApplication;

import org.springframework.beans.factory.annotation.Autowired;
```

```
import org.springframework.context.ApplicationContext;


@SpringBootApplication
public class mahasiswaApp implements ApplicationRunner {
    @Autowired
    private mahasiswaService mahasiswaService;


    public static void main(String[] args){
        System.setProperty("java.awt.headless", "false");
        ApplicationContext context = SpringApplication.run(mahasiswaApp.class, args);
        mahasiswaController controller = context.getBean(mahasiswaController.class);
        mahasiswaView mahasiswaView = new mahasiswaView(controller);
        mahasiswaView.setVisible(true);


    }
    @Override
    public void run(ApplicationArguments args) throws Exception {


    }
}
```

Penjelasan:
Kode di atas mendefinisikan class **mahasiswaApp** sebagai titik masuk utama aplikasi berbasis **Spring Boot** dengan anotasi **@SpringBootApplication**. Method **main** menginisialisasi konteks aplikasi Spring dan mengatur properti **java.awt.headless** ke false untuk memungkinkan penggunaan GUI berbasis **Swing**. Setelah aplikasi dijalankan, objek **mahasiswaController** diperoleh dari konteks Spring dan diteruskan ke **mahasiswaView**, yang kemudian ditampilkan dengan Method setVisible(true). Class ini juga mengimplementasikan **ApplicationRunner**, meskipun Method run belum digunakan. Secara keseluruhan, kode ini menghubungkan logika bisnis, pengendali, dan antarmuka pengguna untuk aplikasi manajemen mahasiswa.

mahasiswa.properties



Sourcecode:

 # Konfigurasi MySQL Hibernate

spring.datasource.url=jdbc:mysql://localhost:3306/pertemuan6_db?useSSL=false&serverTimezone=UTC

spring.datasource.username=root

spring.datasource.password=

spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver

 # Hibernate settings

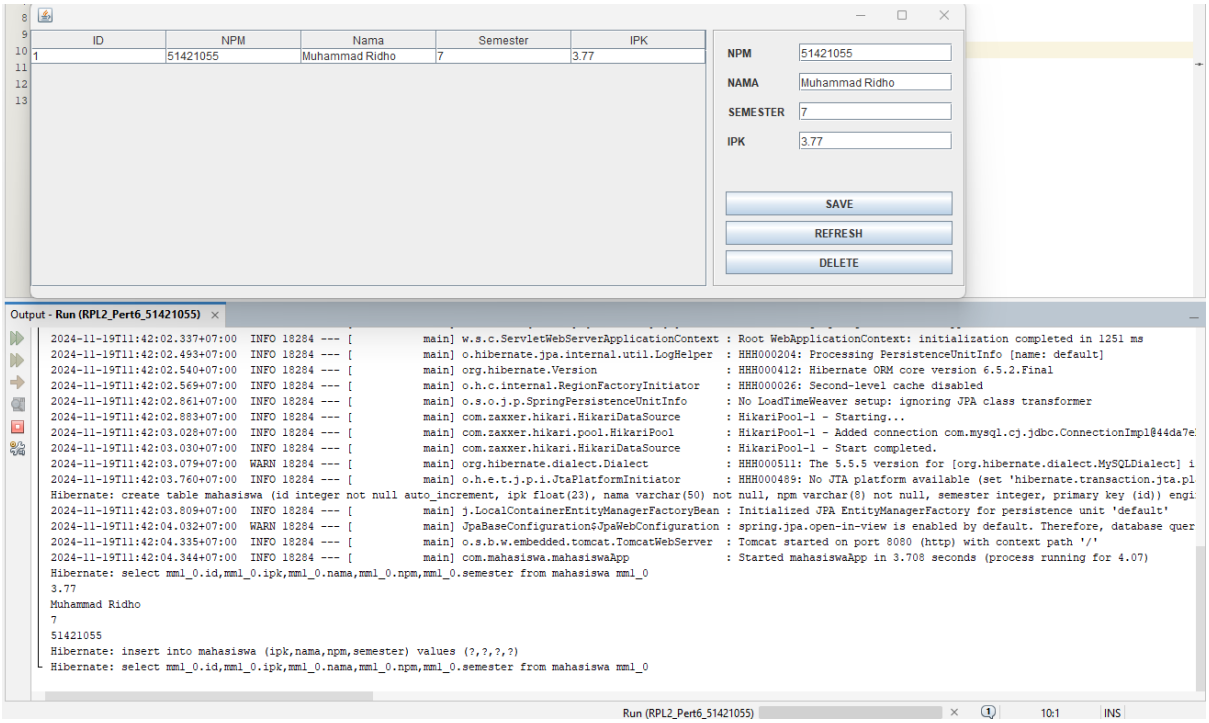
spring.jpa.hibernate.ddl-auto=update

spring.jpa.show-sql=true

Penjelasan:

Kode di atas adalah file konfigurasi untuk aplikasi berbasis **Spring Boot** yang mengatur koneksi ke database **MySQL** dan properti **Hibernate**. Properti **spring.datasource.url** menentukan URL koneksi database, termasuk nama database (pertemuan6_db) dan pengaturan tambahan seperti **useSSL=false** dan **serverTimezone=UTC**. Properti **spring.datasource.username** dan **spring.datasource.password** digunakan untuk kredensial login, sementara **spring.datasource.driver-class-name** menentukan driver JDBC yang digunakan.

Output:


ID
NPM
Nama
Semester
IPK
1
51421055
Muhammad Ridho
7
3.77
NPM
51421055
NAMA
Muhammad Ridho
SEMESTER
7
IPK
3.77
SAVE
REFRESH
DELETE
Output - Run (RPL2_Pert6_51421055)
2024-11-19T11:42:02.337+07:00 INFO 18284 --- [main] w.s.c.ServletWebServerApplicationContext : Root WebApplicationContext: initialization completed in 1251 ms
2024-11-19T11:42:02.493+07:00 INFO 18284 --- [main] o.hibernate.jpa.internal.util.LogHelper : HHH000204: Processing PersistenceUnitInfo [name: default]
2024-11-19T11:42:02.540+07:00 INFO 18284 --- [main] org.hibernate.Version : HHH000412: Hibernate ORM core version 6.5.2.Final
2024-11-19T11:42:02.569+07:00 INFO 18284 --- [main] o.h.c.internal.RegionFactoryInitiator : HHH000026: Second-level cache disabled
2024-11-19T11:42:02.861+07:00 INFO 18284 --- [main] o.s.o.j.p.SpringPersistenceUnitInfo : No LoadTimeWeaver setup: ignoring JPA class transformer
2024-11-19T11:42:02.883+07:00 INFO 18284 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Starting...
2024-11-19T11:42:03.028+07:00 INFO 18284 --- [main] com.zaxxer.hikari.pool.HikariPool : HikariPool-1 - Added connection com.mysql.cj.jdbc.ConnectionImpl@44da7e
2024-11-19T11:42:03.030+07:00 INFO 18284 --- [main] com.zaxxer.hikari.HikariDataSource : HikariPool-1 - Start completed.
2024-11-19T11:42:03.079+07:00 WARN 18284 --- [main] org.hibernate.dialect.Dialect : HHH000511: The 5.5.5 version for [org.hibernate.dialect.MySQLDialect] i
2024-11-19T11:42:03.760+07:00 INFO 18284 --- [main] o.h.e.t.j.p.i.JtaPlatformInitiator : HHH000489: No JTA platform available (set 'hibernate.transaction.jta.pl
Hibernate: create table mahasiswa (id integer not null auto_increment, ipk float(23), nama varchar(50) not null, npm varchar(8) not null, semester integer, primary key (id)) engi
2024-11-19T11:42:03.809+07:00 INFO 18284 --- [main] j.LocalContainerEntityManagerFactoryBean : Initialized JPA EntityManagerFactory for persistence unit 'default'
2024-11-19T11:42:04.032+07:00 WARN 18284 --- [main] JpaBaseConfiguration$JpaWebConfiguration : spring.jpa.open-in-view is enabled by default. Therefore, database quer
2024-11-19T11:42:04.335+07:00 INFO 18284 --- [main] o.s.b.w.embedded.tomcat.TomcatWebServer : Tomcat started on port 8080 (http) with context path '/'
2024-11-19T11:42:04.344+07:00 INFO 18284 --- [main] com.mahasiswa.mahasiswaApp : Started mahasiswaApp in 3.708 seconds (process running for 4.07)
Hibernate: select mml_0.id,mml_0.ipk,mml_0.nama,mml_0.npm,mml_0.semester from mahasiswa mml_0
3.77
Muhammad Ridho
7
51421055
Hibernate: insert into mahasiswa (ipk,nama,npm,semester) values (?,?,?,?)
Hibernate: select mml_0.id,mml_0.ipk,mml_0.nama,mml_0.npm,mml_0.semester from mahasiswa mml_0
Run (RPL2_Pert6_51421055)
10:1
INS