

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 : 51421517
Nama : William Devin Septianus Pranggono
Ketua Asisten :
Paraf Asisten :
Nama Asisten : Gilbert Jefferson Faozato Mendrofa
Jumlah Lembar : 15 lembar

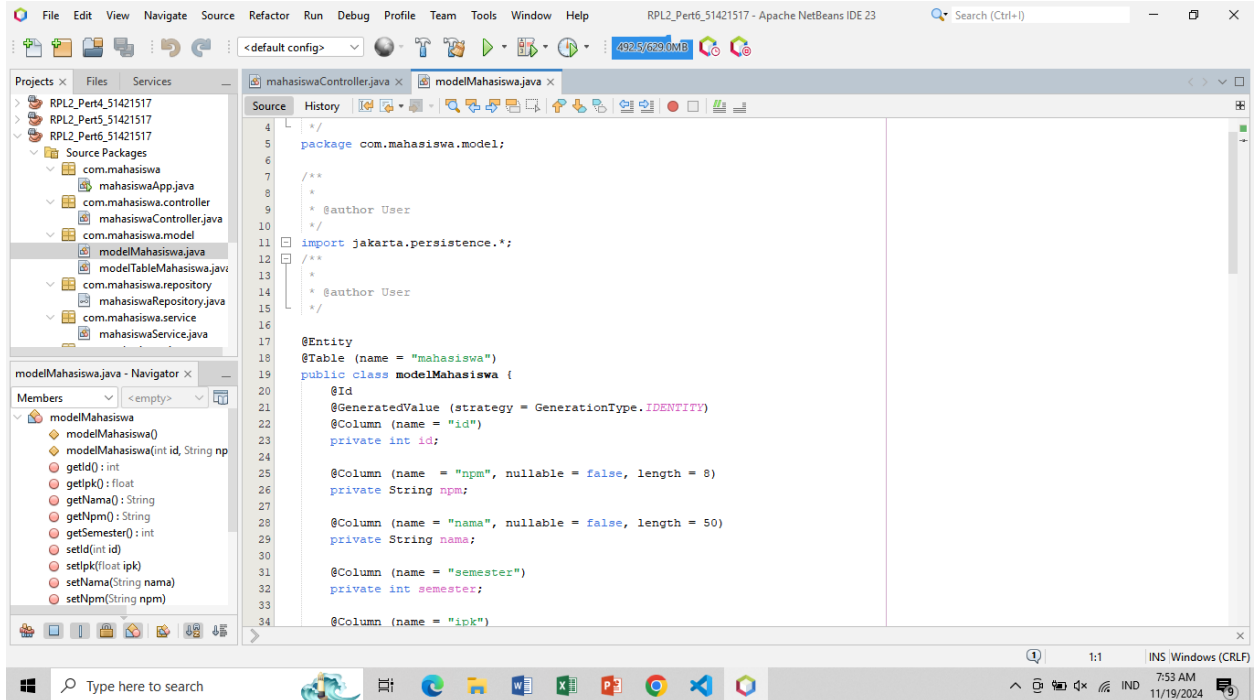
LABORATORIUM TEKNIK INFORMATIKA

UNIVERSITAS GUNADARMA

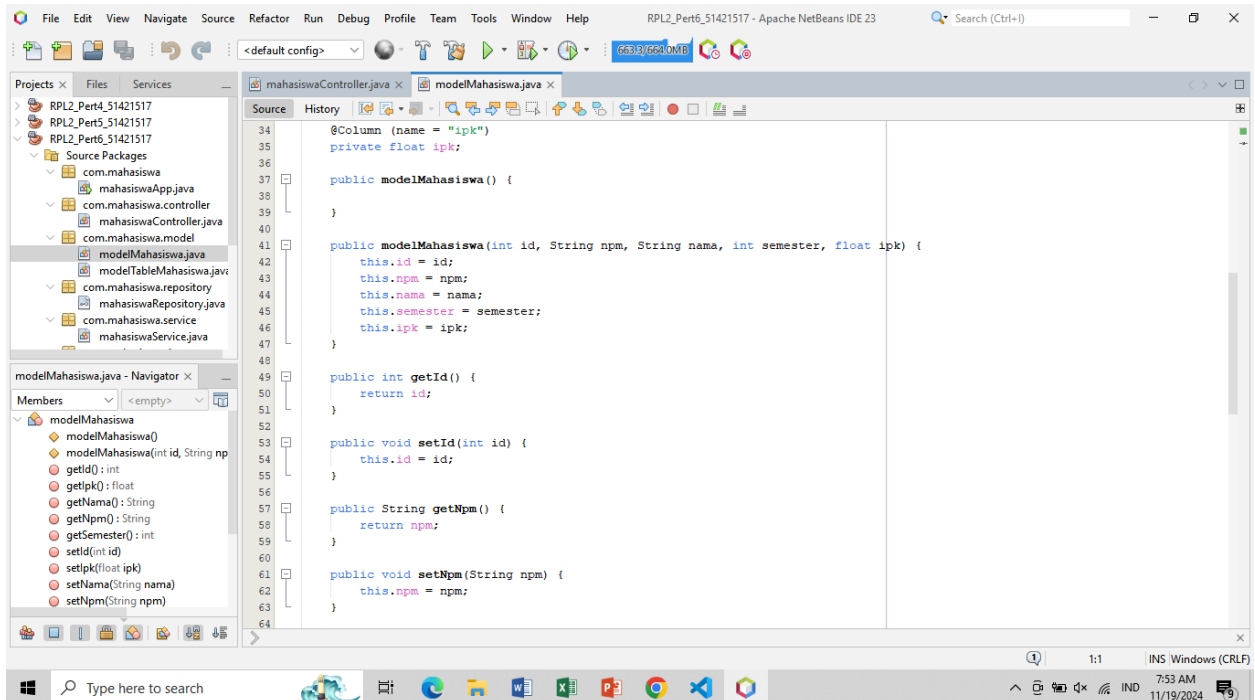
2024

LISTING DAN LOGIKA PROGRAM

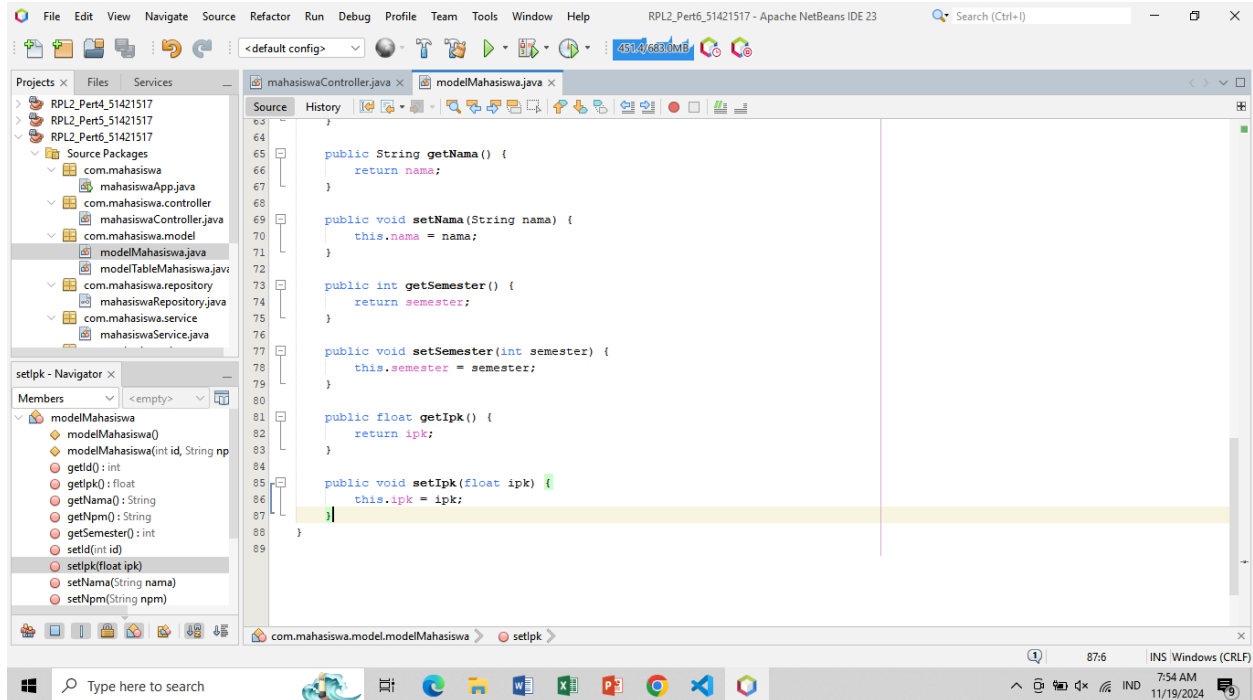
a. modelMahasiswa.java



```
4  */
5  package com.mahasiswa.model;
6
7  /**
8   * @author User
9   */
10
11  import jakarta.persistence.*;
12
13  /**
14   * @author User
15   */
16
17  @Entity
18  @Table(name = "mahasiswa")
19  public class modelMahasiswa {
20
21      @Id
22      @GeneratedValue(strategy = GenerationType.IDENTITY)
23      @Column(name = "id")
24      private int id;
25
26      @Column(name = "npm", nullable = false, length = 8)
27      private String npm;
28
29      @Column(name = "nama", nullable = false, length = 50)
30      private String nama;
31
32      @Column(name = "semester")
33      private int semester;
34
35      @Column(name = "ipk")
```

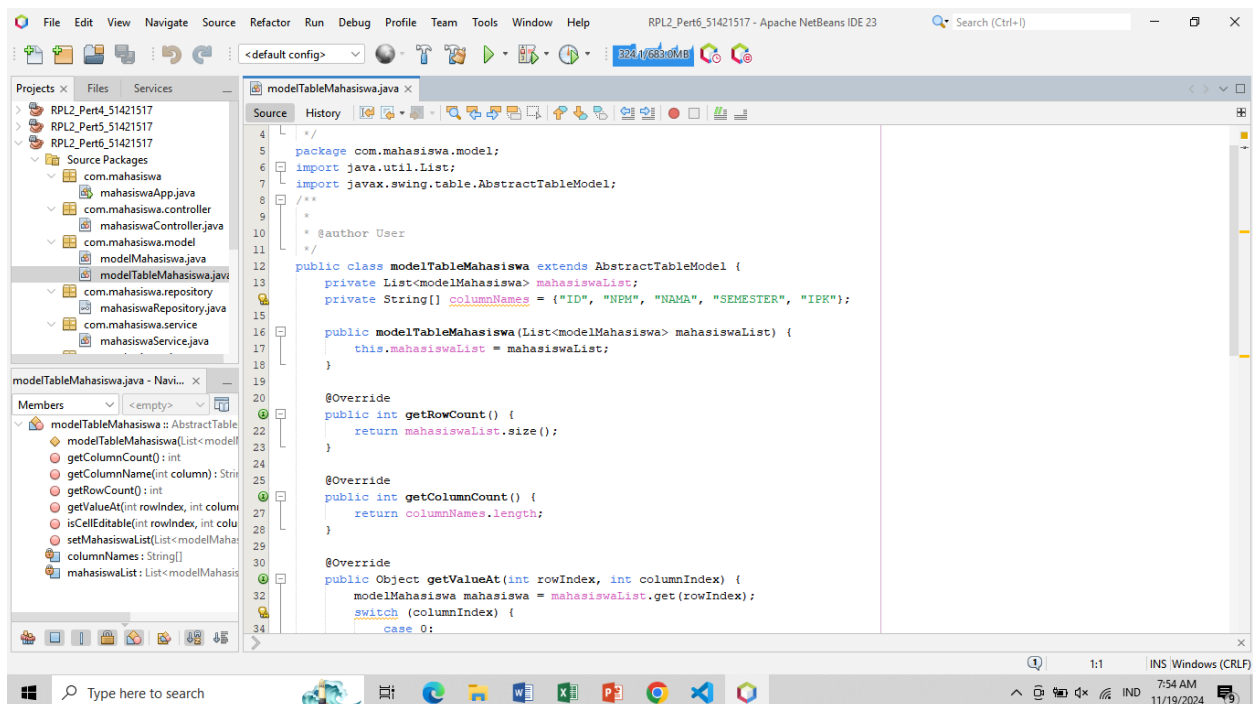


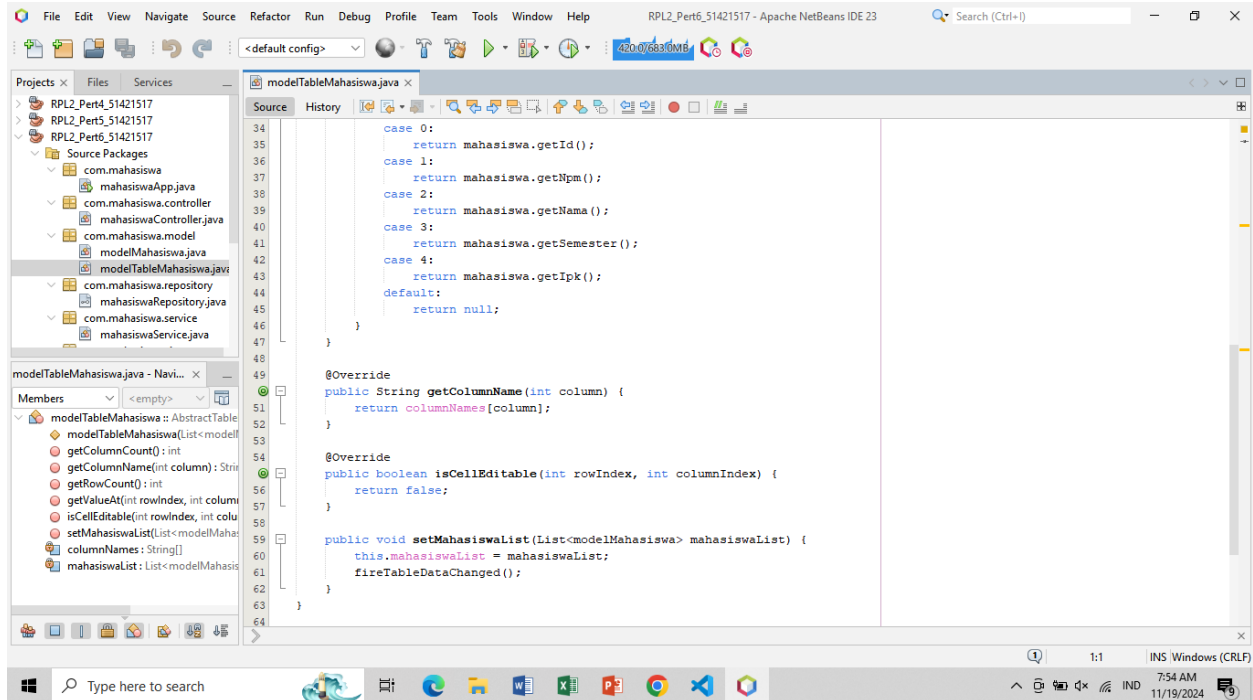
```
34  @Column(name = "ipk")
35  private float ipk;
36
37  public modelMahasiswa() {
38
39  }
40
41  public modelMahasiswa(int id, String npm, String nama, int semester, float ipk) {
42      this.id = id;
43      this.npm = npm;
44      this.nama = nama;
45      this.semester = semester;
46      this.ipk = ipk;
47  }
48
49  public int getId() {
50      return id;
51  }
52
53  public void setId(int id) {
54      this.id = id;
55  }
56
57  public String getNpm() {
58      return npm;
59  }
60
61  public void setNpm(String npm) {
62      this.npm = npm;
63  }
64  }
```



Class modelMahasiswa mendefinisikan struktur dari objek/tabel mahasiswa pada database, yaitu atribut serta tipe data yang dipakai. Karena atribut-atribut tersebut bersifat private, maka diperlukan method getter dan setter untuk membaca dan memanipulasi atribut.

b. 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, beserta method `setMahasiswaList` untuk memodifikasi atribut `mahasiswaList` milik objek.

c. `mahasiswaController.java`

```
1 package com.mahasiswa.controller;
2 import org.springframework.beans.factory.annotation.Autowired;
3 import org.springframework.web.bind.annotation.*;
4 import com.mahasiswa.model.modelMahasiswa;
5 import com.mahasiswa.service.mahasiswaService;
6 import java.util.List;
7 import org.springframework.stereotype.Controller;
8
9 /**
10  * @author User
11  */
12 @Controller
13 public class mahasiswaController {
14     @Autowired
15     private mahasiswaService mahasiswaService;
16
17     public String addMahasiswa(@RequestBody modelMahasiswa mhs) {
18         mahasiswaService.addMhs(mhs);
19         return "Mahasiswa berhasil ditambahkan";
20     }
21
22     public modelMahasiswa getMahasiswa(@PathVariable int id) {
23         return mahasiswaService.getMhs(id);
24     }
25
26     public String updateMahasiswa(@RequestBody modelMahasiswa mhs) {
27         mahasiswaService.updateMhs(mhs);
28         return "Mahasiswa berhasil diperbarui";
29     }
30
31     public String deleteMahasiswa(@PathVariable int id) {
```

```
32     }
33
34     public List<modelMahasiswa> getAllMahasiswa() {
35         return mahasiswaService.getAllMahasiswa();
36     }
37
38 }
39
40
41
42
43
44
```

Class Mahasiswa Controller dianotasi dengan `@Controller`, yang berarti merupakan class yang berperan sebagai Controller pada Spring MVC. Objek mahasiswaService akan di-inject secara otomatis dengan bean mahasiswaService.

Method `addMahasiswa` menerima objek `modelMahasiswa` dan akan menambahkan objek tersebut ke tabel dengan memanggil method `addMhs` milik objek `mahasiswaService`.

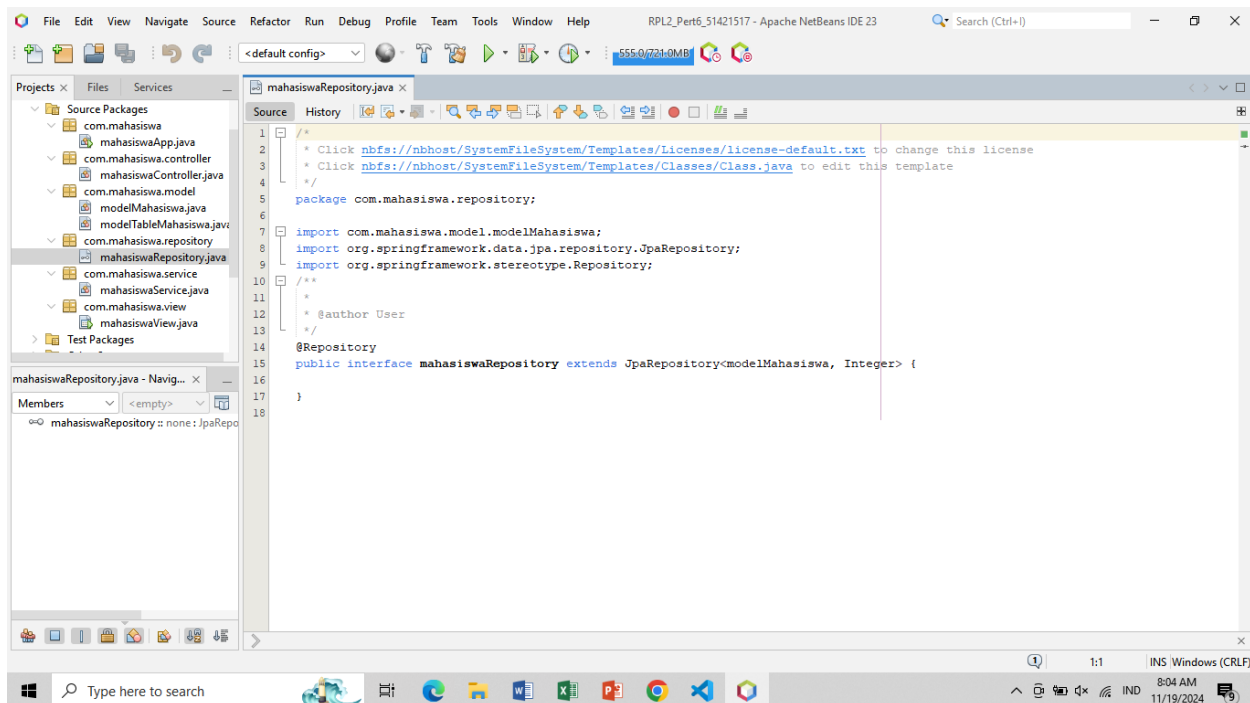
Method `getMahasiswa` menerima parameter `id` berupa integer dan akan mendapatkan record sesuai `id` yang diberikan dari tabel dengan memanggil method `getMhs` milik objek `mahasiswaService`.

Method `updateMahasiswa` menerima objek `modelMahasiswa` dan akan mengupdate data pada tabel dengan memanggil method `updateMhs` milik objek `mahasiswaService`.

Method `deleteMahasiswa` menerima parameter `id` berupa integer dan akan menghapus record sesuai `id` yang diberikan pada tabel dengan memanggil method `deleteMhs` milik objek `mahasiswaService`.

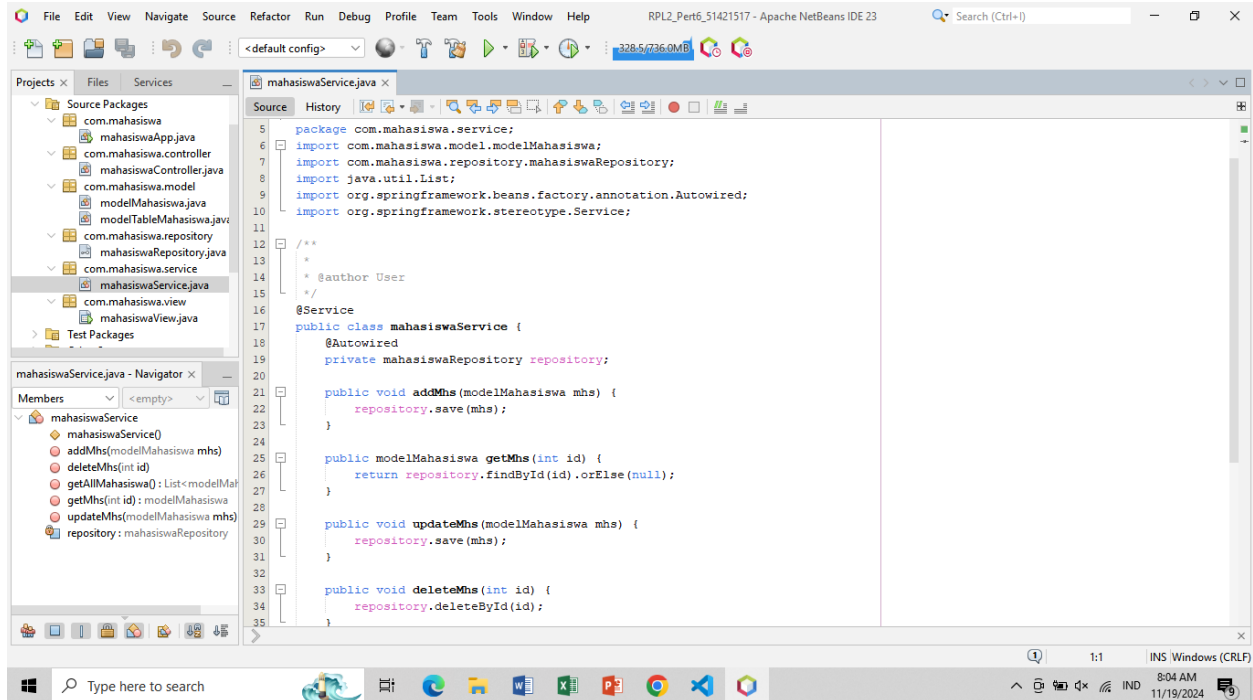
Method `getAllMahasiswa` akan membaca seluruh data pada tabel dengan memanggil method `getAllMahasiswa` milik objek `mahasiswaService`.

d. mahasiswaRepository.java



Class `mahasiswaRepository` berperan sebagai Repository atau penyimpanan data pada program, dengan tipe data yang disimpan adalah `modelMahasiswa`.

e. mahasiswaService.java



```
package com.mahasiswa.service;
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;

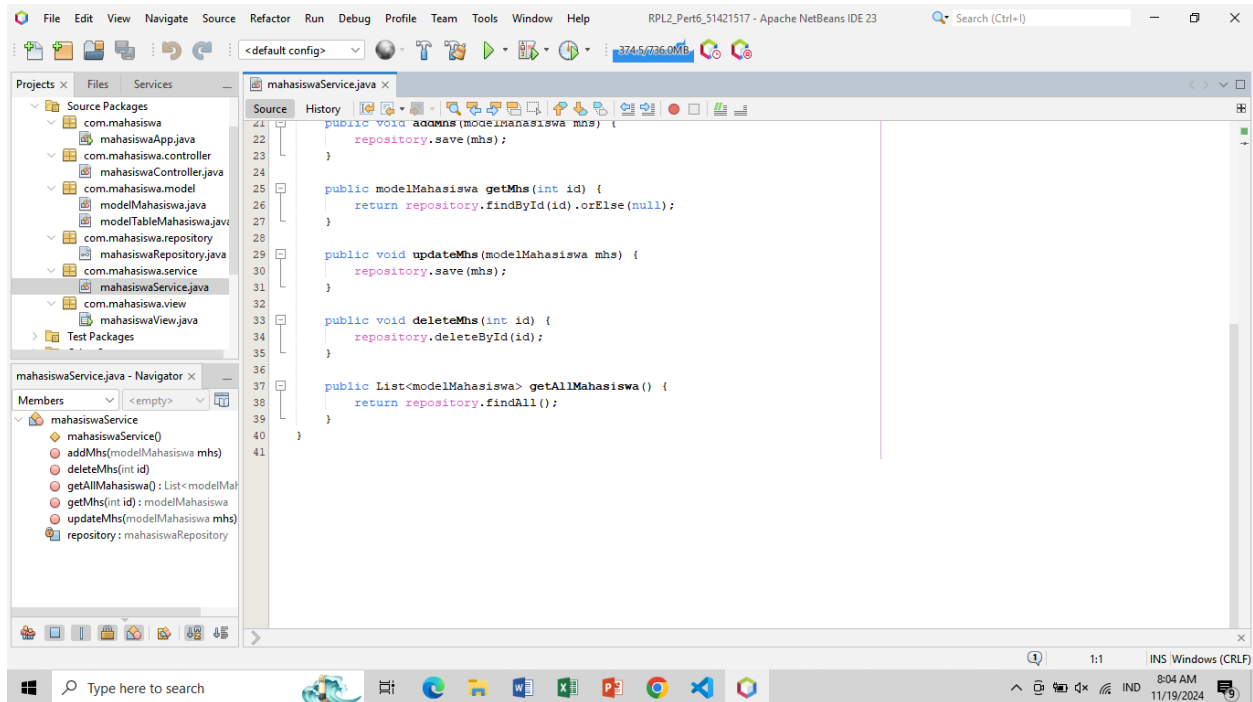
/**
 * @author User
 */
@Service
public class mahasiswaService {
    @Autowired
    private mahasiswaRepository repository;

    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 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();
}
}
```

Class mahasiswaService menyediakan service untuk menambahkan dan memanipulasi data pada database melalui objek mahasiswaRepository yang di-inject.

Method addMhs menerima parameter objek modelMahasiswa dan akan menyimpan data tersebut pada tabel melalui objek mahasiswaRepository.

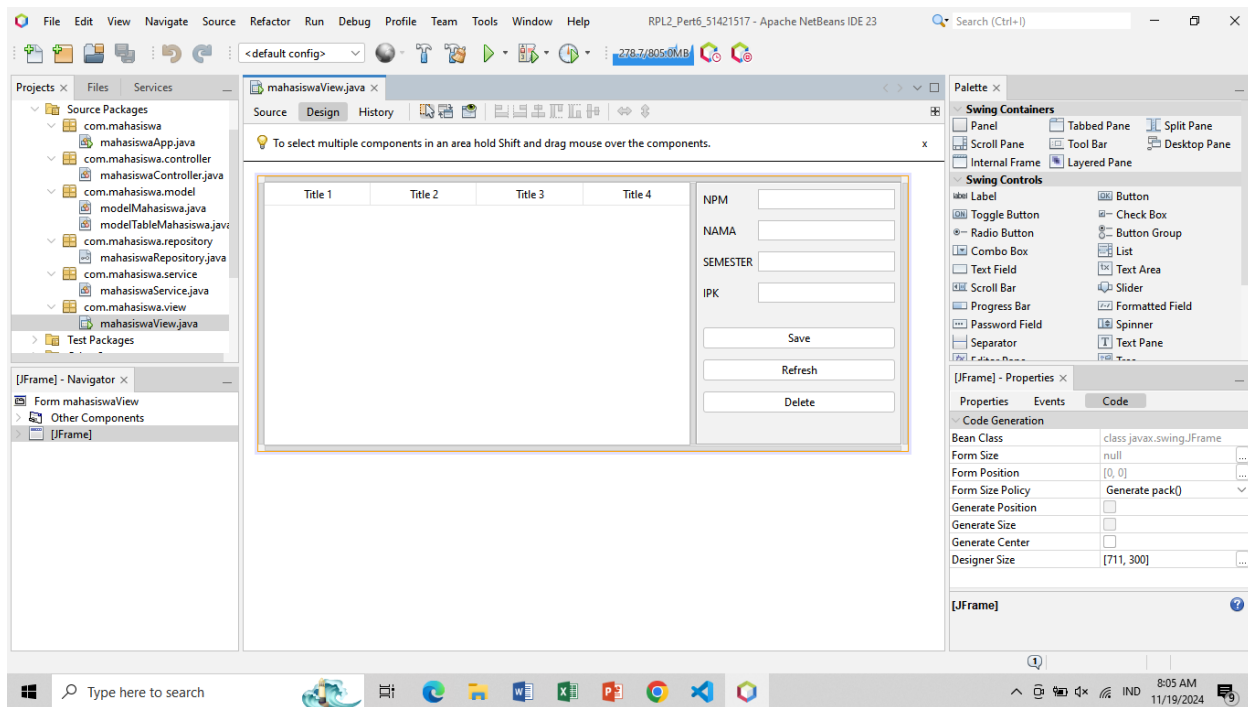
Method `getMhs` menerima parameter `id` berupa integer dan akan mencari data dengan id tersebut, dan jika data dengan id tersebut tidak ditemukan, maka akan mengembalikan `null`.

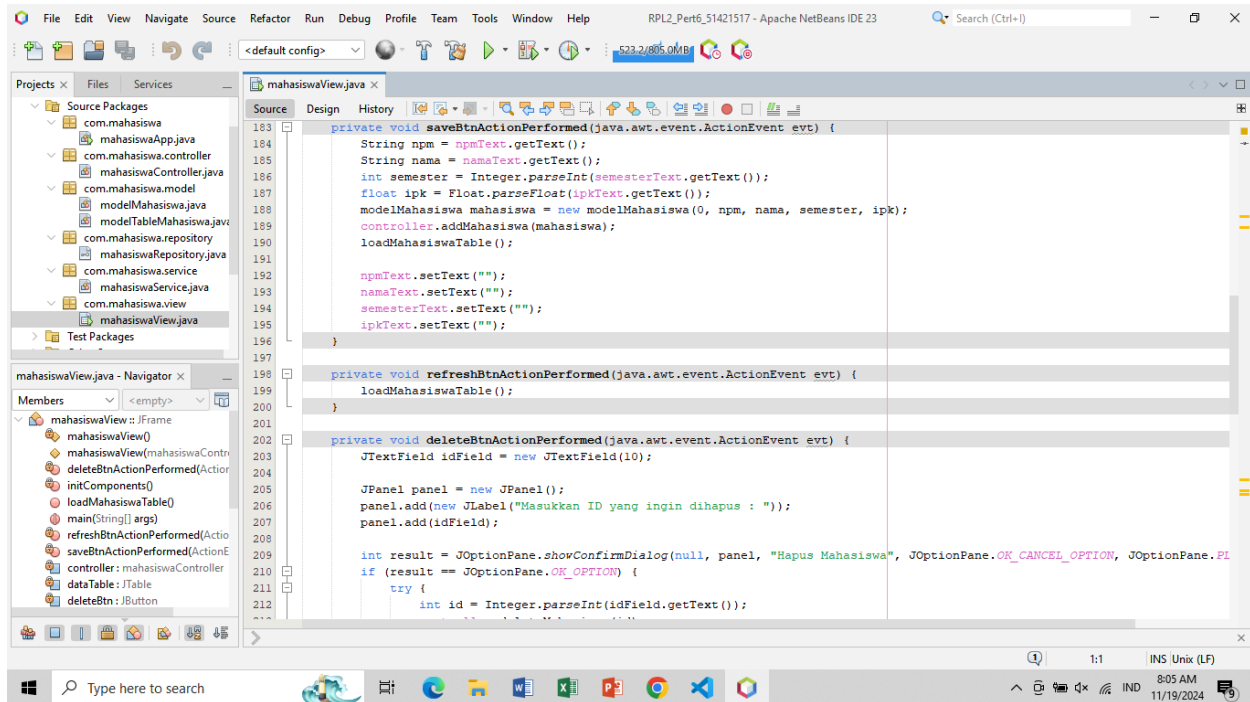
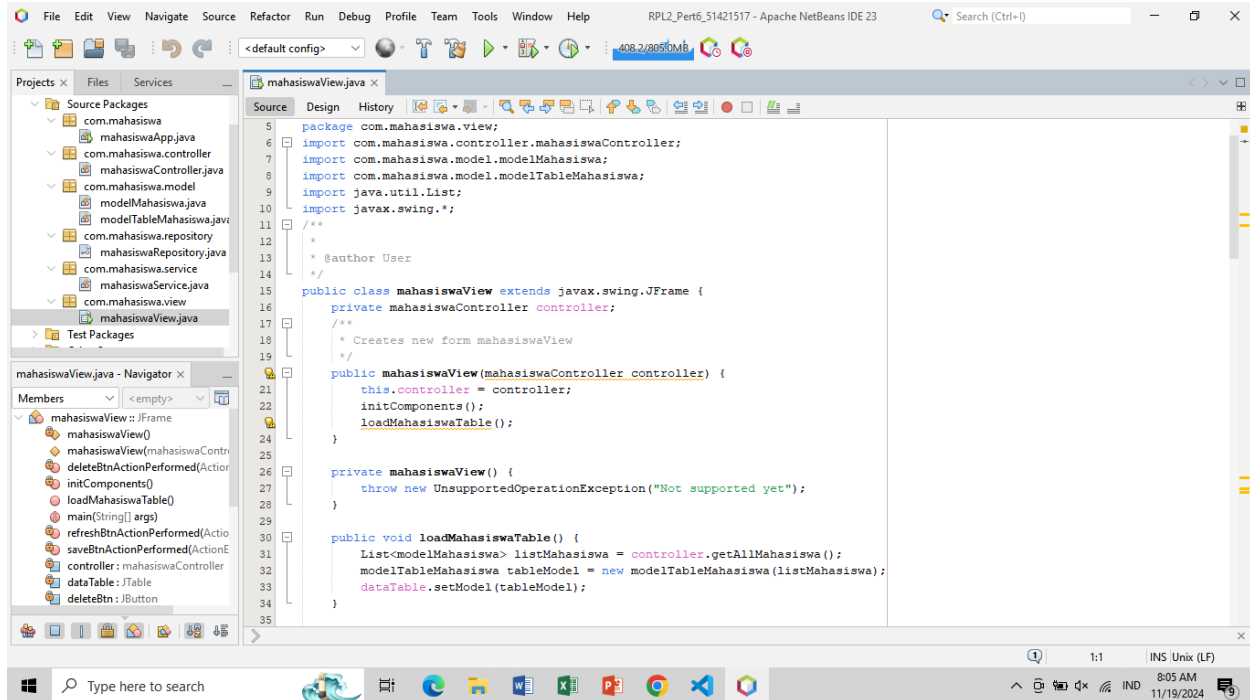
Method `updateMhs` menerima parameter objek `modelMahasiswa` dan akan mengupdate data tersebut pada tabel melalui objek `mahasiswaRepository`.

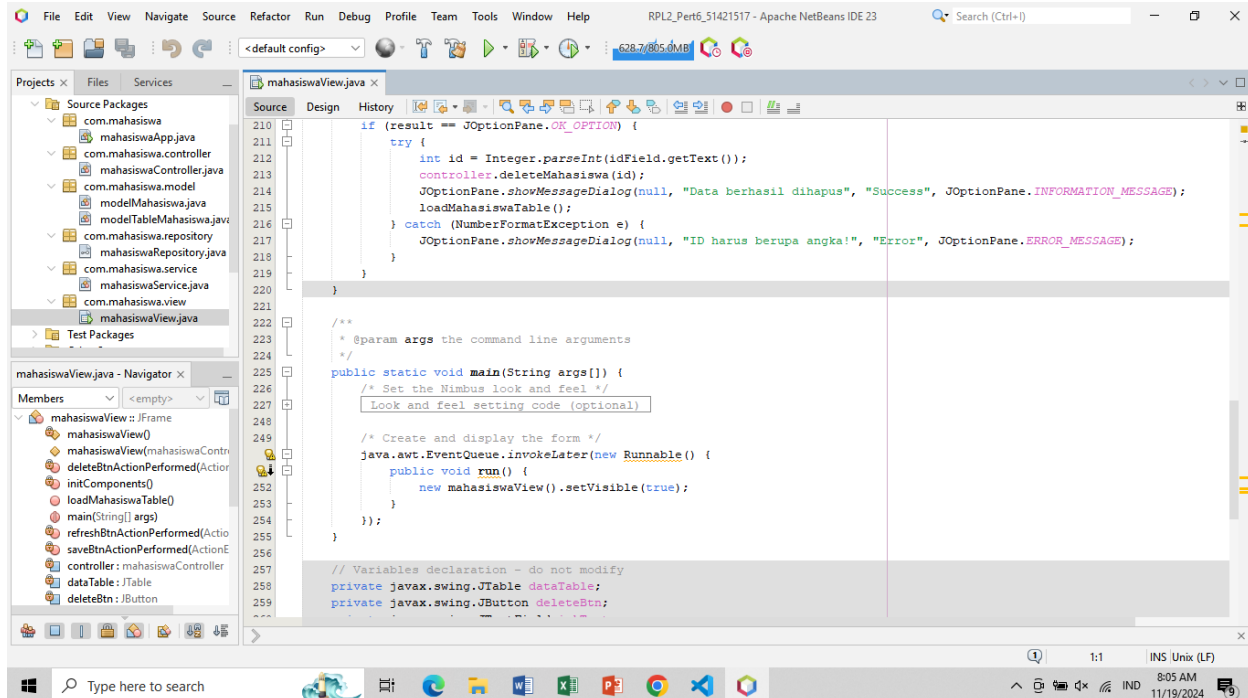
Method `deleteMhs` menerima parameter `id` berupa integer dan akan menghapus data dengan id tersebut.

Method `getAllMahasiswa` akan mengembalikan seluruh data pada tabel dengan memanggil method `findAll` milik objek `repository`.

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.

Constructor mahasiswaView akan menginisialisasi komponen widget yang digunakan pada JFrame, menginisiasi controller, dan memuat tabel mahasiswa dengan memanggil method loadMahasiswaTable.

Method loadMahasiswaTable 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).

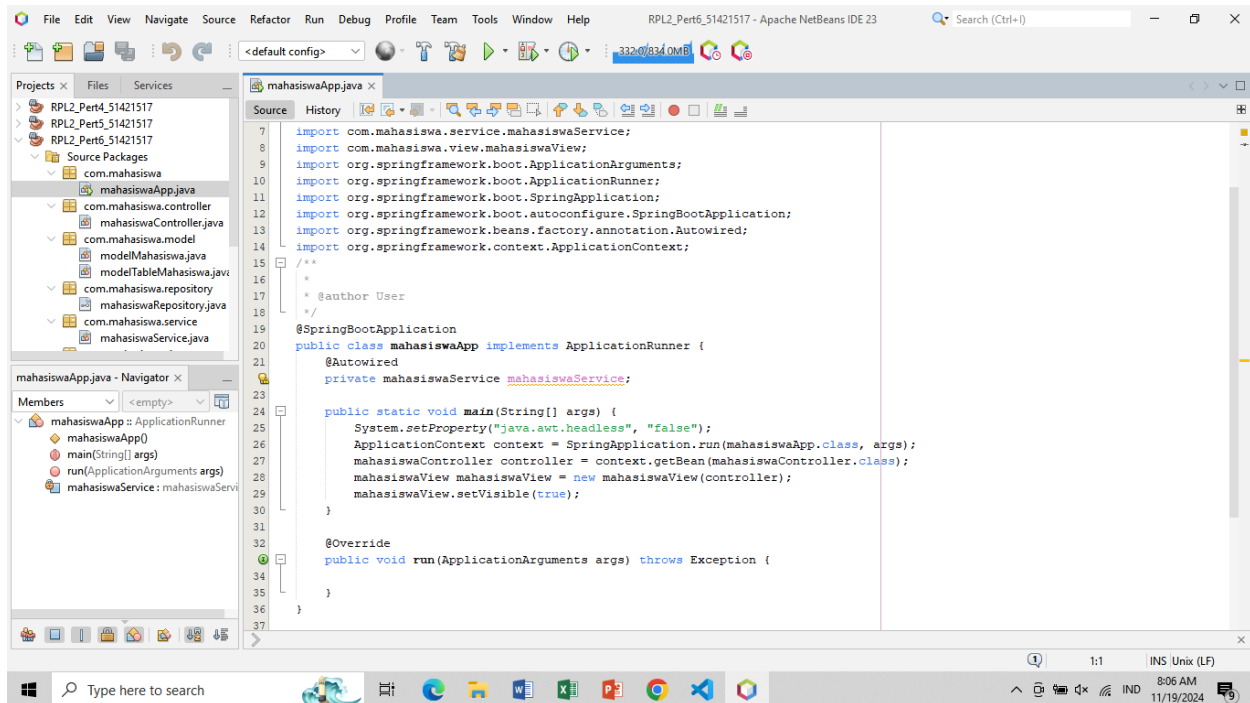
Method saveBtnActionPerformed akan dieksekusi ketika tombol Simpan diklik. Hal yang dilakukan pertama-tama adalah mendapatkan data input mahasiswa, kemudian membuat objek modelMahasiswa baru dengan parameter-parameter tersebut. Objek controller kemudian menambahkan data tersebut ke database dengan memanggil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

Method refreshBtnActionPerformed berfungsi untuk memuat ulang tabel dengan memanggil fungsi loadMahasiswaTable.

Method deleteBtnActionPerformed akan dieksekusi ketika tombol Delete 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

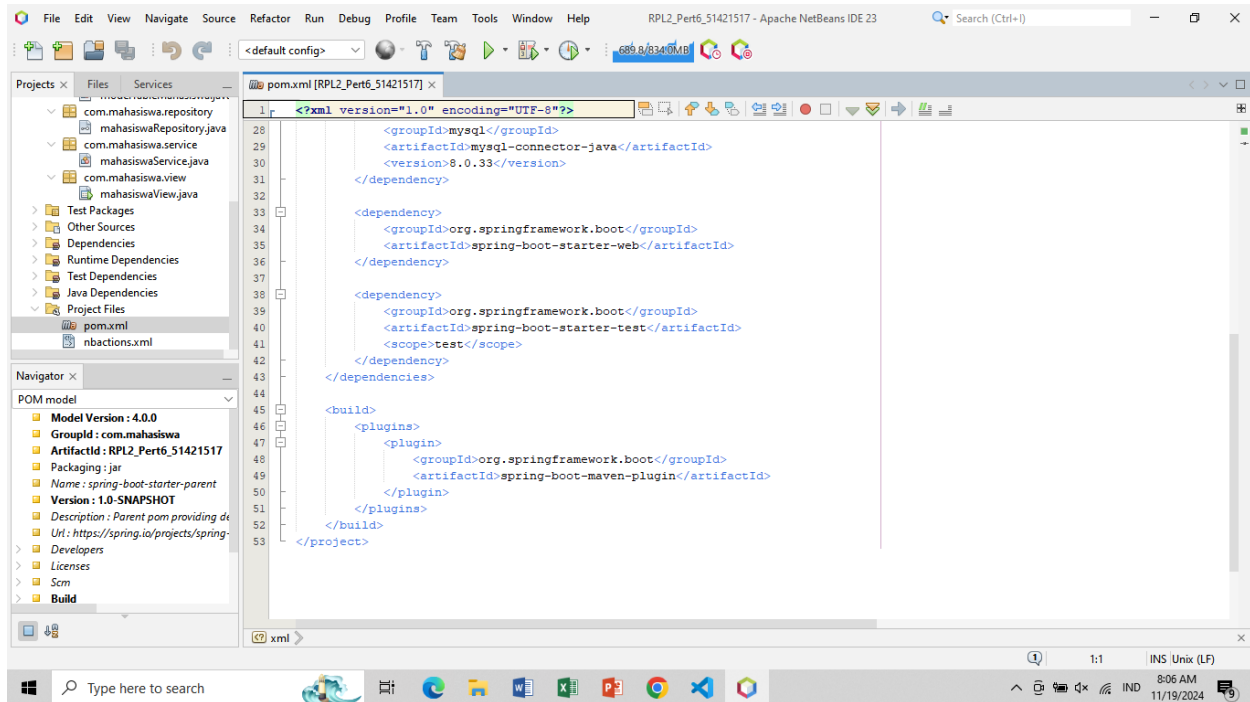
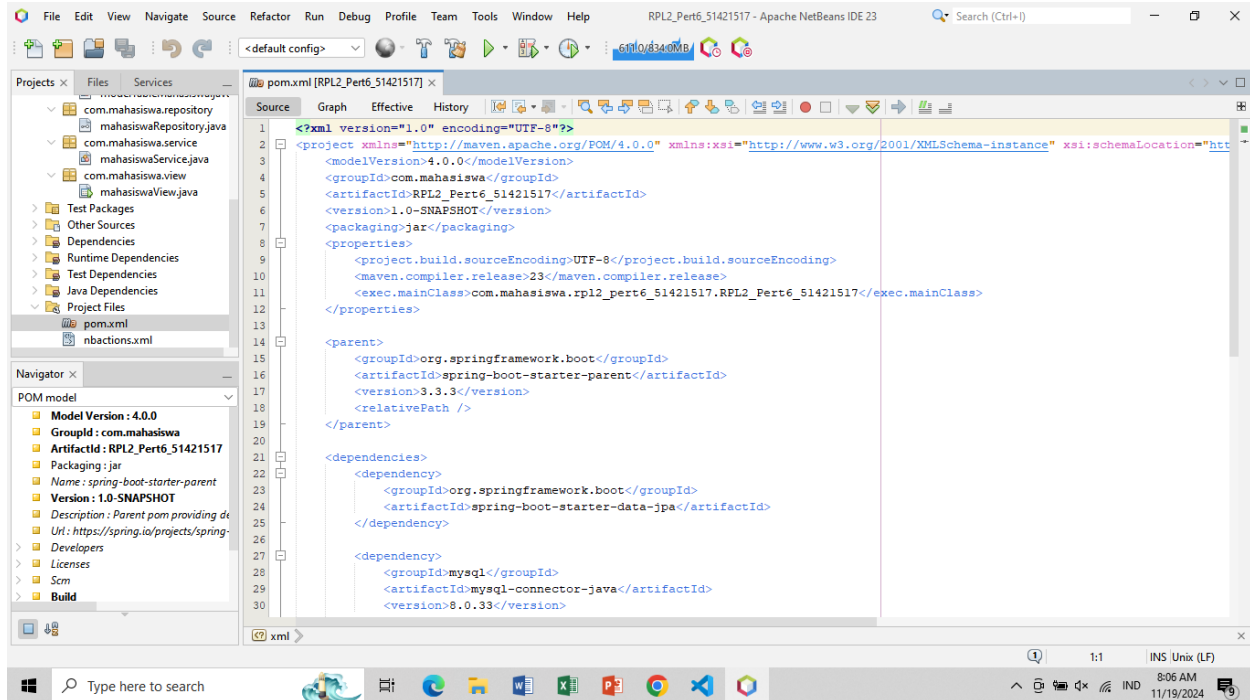
deleteMahasiswa milik objek controller. Setelah berhasil dihapus dan menampilkan dialog konfirmasi, tabel akan dimuat ulang.

g. mahasiswaApp.java



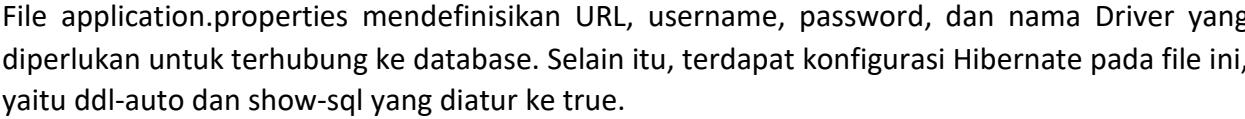
Class tersebut adalah class main yang akan dipanggil ketika project dijalankan. Class tersebut mengimplementasi interface ApplicationRunner dari Spring. Objek dari MahasiswaController dan mahasiswaView akan diinisialisasi, kemudian program akan dijalankan.

h. pom.xml



File pom.xml berisi dependencies yang digunakan pada program.

i. application.properties



Tampilan awal

Tampilan memasukkan record

A screenshot of a software window titled 'Hapus Mahasiswa' (Delete Student). The window contains a table with columns: ID, NPM, NAMA, SEMESTER, and IPK. The table is currently empty. To the right of the table are four input fields labeled NPM, NAMA, SEMESTER, and IPK, each containing a value: 51421517, William Devin S.P., 7, and 3.96 respectively. Below these fields are three buttons: Save, Refresh, and Delete.

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

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

Save
 Refresh
 Delete

Tampilan ketika record berhasil ditambahkan

A screenshot of the same software window. The table now contains one row with the following data: ID: 1, NPM: 51421517, NAMA: William Devin S.P., SEMESTER: 7, IPK: 3.96. The input fields on the right are now empty. The Save, Refresh, and Delete buttons are still present.

ID	NPM	NAMA	SEMESTER	IPK
1	51421517	William Devin S.P.	7	3.96

NPM:
 NAMA:
 SEMESTER:
 IPK:

Save
 Refresh
 Delete

Tampilan dialog box setelah klik tombol Delete

A screenshot of the software window with the same data as the previous image. A dialog box titled 'Hapus Mahasiswa' is displayed in the foreground. It contains a text field labeled 'Masukkan ID yang ingin dihapus : ' and two buttons: OK and Cancel.

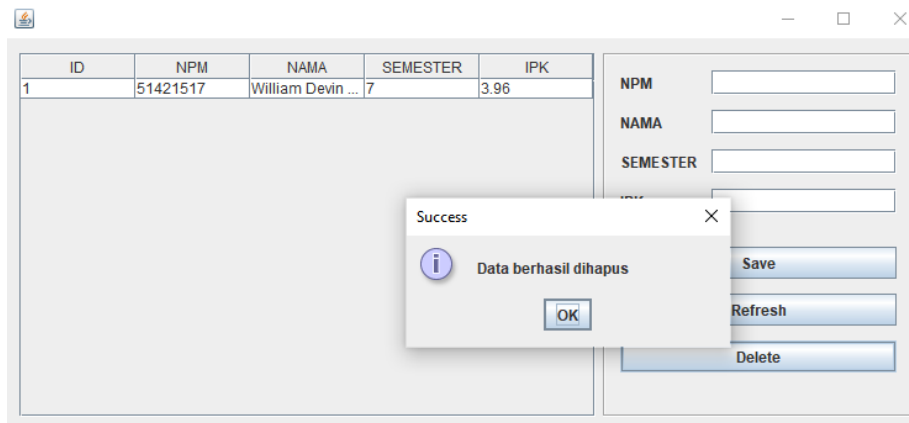
ID	NPM	NAMA	SEMESTER	IPK
1	51421517	William Devin ...	7	3.96

NPM:
 NAMA:
 SEMESTER:
 IPK:

Save
 Refresh
 Delete

Hapus Mahasiswa
 Masukkan ID yang ingin dihapus :
 OK Cancel

Tampilan dialog box setelah menghapus record



Tampilan setelah record dihapus

