

## **LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : Rekayasa Perangkat Lunak 2

Kelas : 4IA14

Praktikum ke- : 6

Tanggal : 18 November 2024

Materi : Implementasi AOP dan Dependency Injection pada Project Spring dan Hibernate

NPM : 50421651

Nama : Indah Dwi Apriliani

Ketua Asisten : Suryo Aji Widagdo

Paraf Asisten :

Nama Asisten :

Jumlah Lembar : 13 Lembar

**LABORATORIUM TEKNIK INFORMATIKA**

**UNIVERSITAS GUNADARMA**

**2024**

1. Jelaskan Kode dan langkah-langkah program yang telah dibuat!

Jawab :

### Pertemuan6\_50421651.java

```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change
3  */
4
5  package me.indah;
6
7  import me.indah.controller.MahasiswaController;
8  import me.indah.service.MahasiswaService;
9  import me.indah.view.MahasiswaView;
10 import org.springframework.boot.ApplicationArguments;
11 import org.springframework.boot.ApplicationRunner;
12 import org.springframework.boot.SpringApplication;
13 import org.springframework.boot.autoconfigure.SpringBootApplication;
14 import org.springframework.beans.factory.annotation.Autowired;
15 import org.springframework.context.ApplicationContext;
16
17 /**
18 * @author ACER
19 */
20 @SpringBootApplication
21 public class Pertemuan6_50421651 implements ApplicationRunner {
22     @Autowired
23     private MahasiswaService mahasiswaService;
24
25     public static void main(String[] args) {
26         System.setProperty(key: "java.awt.headless", value: "false");
27
28         ApplicationContext context = SpringApplication.run(primarySource: Pertemuan6_50421651
29
30         MahasiswaController controller = context.getBean(requiredType: MahasiswaController.c
31         MahasiswaView mahasiswaView = new MahasiswaView(controller);
32         mahasiswaView.setVisible(b: true);
33     }
34 }
```

Penjelasan :

Kode di atas adalah aplikasi Spring Boot yang mengimplementasikan pola Model-View-Controller (MVC) untuk mengelola data mahasiswa. Aplikasi dimulai dengan kelas utama `Pertemuan6\_50421651`, yang menjalankan aplikasi Spring Boot dan mengimplementasikan interface `ApplicationRunner`. Melalui anotasi `@Autowired`, Spring secara otomatis meng-inject objek `MahasiswaService` untuk menangani logika bisnis. Di dalam metode `main`, aplikasi mengambil instance dari `ApplicationContext` untuk mendapatkan `MahasiswaController`, yang menghubungkan model dan view. Tampilan aplikasi (`MahasiswaView`) kemudian dikaitkan dengan controller dan ditampilkan. Metode `run()` kosong, namun dapat digunakan untuk logika tambahan setelah aplikasi dijalankan. Secara keseluruhan, aplikasi ini adalah contoh dasar penggunaan Spring Boot dengan pola desain MVC.

## Pom.xml

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/
3     <modelVersion>4.0.0</modelVersion>
4     <groupId>me.indah</groupId>
5     <artifactId>Pertemuan6_50421651</artifactId>
6     <version>1.0-SNAPSHOT</version>
7     <packaging>jar</packaging>
8     <properties>
9         <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
10        <maven.compiler.source>20</maven.compiler.source>
11        <maven.compiler.target>20</maven.compiler.target>
12        <exec.mainClass>me.indah.Pertemuan6_50421651</exec.mainClass>
13    </properties>
14
15    <parent>
16        <groupId>org.springframework.boot</groupId>
17        <artifactId>spring-boot-starter-parent</artifactId>
18        <version>3.3.3</version>
19        <relativePath/>
20    </parent>
21
22    <dependencies>
23        <!-- Hibernate + Spring Data JPA -->
24        <dependency>
25            <groupId>org.springframework.boot</groupId>
26            <artifactId>spring-boot-starter-data-jpa</artifactId>
27        </dependency>
28
29        <!-- MySQL Connector -->
30        <dependency>
31            <groupId>mysql</groupId>
32            <artifactId>mysql-connector-java</artifactId>
33            <version>8.0.33</version>
```

### Penjelasan :

File pom.xml ini adalah konfigurasi Maven untuk proyek Spring Boot. Di dalamnya, terdapat pengaturan dasar seperti groupId, artifactId, dan version yang mendefinisikan identitas proyek. Bagian properties mengonfigurasi encoding dan versi Java yang digunakan, yaitu Java 20. Proyek ini menggunakan spring-boot-starter-parent sebagai proyek induk untuk memanfaatkan pengaturan default Spring Boot.

Dependensi yang digunakan termasuk spring-boot-starter-data-jpa untuk integrasi dengan Hibernate dan JPA, mysql-connector-java untuk koneksi ke database MySQL, serta spring-boot-starter-web untuk aplikasi berbasis web. spring-boot-starter-test disertakan untuk pengujian.

Bagian build mengonfigurasi penggunaan spring-boot-maven-plugin untuk memudahkan proses build dan menjalankan aplikasi sebagai JAR yang dapat dieksekusi. Secara keseluruhan, file ini mengonfigurasi proyek Spring Boot dengan dukungan untuk database MySQL, JPA, dan pengujian.

## ModelMahasiswa.java

```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this
4   */
5   package me.indah.model;
6
7   import jakarta.persistence.*;
8
9   /**
10    *
11    * @author ACER
12    */
13
14   @Entity
15   @Table(name = "mahasiswa")
16   public class ModelMahasiswa {
17
18       @Id
19       @GeneratedValue(strategy = GenerationType.IDENTITY)
20       @Column(name = "id")
21       private int id;
22
23       @Column(name = "npm", nullable = false, length = 10)
24       private String npm;
25
26       @Column(name = "nama", nullable = false, length = 55)
27       private String nama;
28
29       @Column(name = "semester")
30       private int semester;
31
32       @Column(name = "ipk")
33       private float ipk;
```

### Penjelasan :

Kode di atas adalah kelas model `ModelMahasiswa` yang menggunakan JPA (Java Persistence API) untuk memetakan entitas data mahasiswa ke dalam tabel `mahasiswa` di database. Kelas ini memiliki atribut seperti `id`, `npm`, `nama`, `semester`, dan `ipk`, yang masing-masing dipetakan ke kolom dalam tabel. Atribut `id` berfungsi sebagai kunci utama dengan strategi auto-increment. Setiap kolom diatur dengan anotasi `@Column` untuk menentukan properti seperti panjang dan apakah kolom tersebut boleh bernilai `null`. Kelas ini juga menyediakan konstruktor dan metode getter/setter untuk mengakses dan memperbarui atribut. Secara keseluruhan, kelas ini mewakili data mahasiswa yang dikelola melalui JPA.

## MahasiswaRepository.java

```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to cha
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this tem
4   */
5   package me.indah.repository;
6
7   import me.indah.model.ModelMahasiswa;
8   import org.springframework.data.jpa.repository.JpaRepository;
9   import org.springframework.stereotype.Repository;
10  /**
11   *
12   * @author ACER
13   */
14
15  @Repository
16  public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Integer> {
17
18  }
19
```

### Penjelasan :

Kode di atas adalah interface `MahasiswaRepository` yang memperluas `JpaRepository` untuk mempermudah operasi CRUD pada entitas `ModelMahasiswa` dengan Spring Data JPA. Anotasi `@Repository` menandakan bahwa interface ini dikelola oleh Spring untuk mengakses data di database.

## MahasiswaService.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit th
4   */
5   package me.indah.service;
6
7   import me.indah.model.ModelMahasiswa;
8   import me.indah.repository.MahasiswaRepository;
9   import java.util.List;
10  import org.springframework.beans.factory.annotation.Autowired;
11  import org.springframework.stereotype.Service;
12
13  /**
14   * @author ACER
15   */
16
17  @Service
18  public class MahasiswaService {
19      @Autowired
20      private MahasiswaRepository repository;
21
22      public void addMhs(ModelMahasiswa mhs) {
23          repository.save(entity:mhs);
24      }
25
26      public ModelMahasiswa getMhs(int id) {
27          return repository.findById(id).orElse(other: null);
28      }
29
30      public void updateMhs(ModelMahasiswa mhs) {
31          repository.save(entity:mhs);
32      }
33  }
```

### Penjelasan :

Kode di atas adalah kelas `MahasiswaService` yang menyediakan layanan untuk mengelola data mahasiswa melalui operasi CRUD menggunakan repository `MahasiswaRepository`. Metode yang tersedia meliputi menambah, mengambil, memperbarui, menghapus, dan mengambil seluruh data mahasiswa. Kelas ini dikelola oleh Spring melalui anotasi `@Service` dan memanfaatkan dependency injection untuk mengakses repository. Secara keseluruhan, kelas ini mengelola logika bisnis terkait data mahasiswa tanpa memerlukan penulisan query SQL manual.

## MahasiswaContoller.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit thi
4   */
5   package me.indah.controller;
6
7   import org.springframework.beans.factory.annotation.Autowired;
8   import org.springframework.web.bind.annotation.*;
9   import me.indah.model.ModelMahasiswa;
10  import me.indah.service.MahasiswaService;
11
12  import java.util.List;
13  import org.springframework.stereotype.Controller;
14
15  /**
16   *
17   * @author ACER
18   */
19  @Controller
20  public class MahasiswaController {
21      @Autowired
22      private MahasiswaService mahasiswaService;
23
24      public String addMahasiswa(@RequestBody ModelMahasiswa mhs) {
25          mahasiswaService.addMhs(mhs);
26          return "Mahasiswa added successfully";
27      }
28
29      public ModelMahasiswa getMahasiswa(@PathVariable int id) {
30          return mahasiswaService.getMhs(id);
31      }
32  }
```

### Penjelasan :

Kode di atas adalah kelas `MahasiswaController` yang berfungsi untuk menangani permintaan HTTP terkait data mahasiswa. Dengan anotasi `@Controller`, kelas ini menghubungkan antarmuka pengguna dengan layanan `MahasiswaService`. Terdapat beberapa metode untuk operasi CRUD: menambah mahasiswa, mengambil data berdasarkan ID, memperbarui, menghapus, dan mengambil seluruh data mahasiswa. Setiap metode memanfaatkan layanan `MahasiswaService` untuk berinteraksi dengan database dan mengembalikan respons berupa pesan konfirmasi atau data mahasiswa.

## MahasiswaView.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to
 * Click nbfs://nbhost/SystemFileSystem/Templates/GuiForms/JFrame.java to edit t
 */
package me.indah.view;

import java.util.List;
import me.indah.controller.MahasiswaController;
import me.indah.model.ModelMahasiswa;
import javax.swing.*;
import me.indah.model.ModelTabelMahasiswa;

/**
 *
 * @author ACER
 */
public class MahasiswaView extends JFrame {

    private MahasiswaController controller;

    public MahasiswaView(MahasiswaController controller) {
        this.controller = controller;
        initComponents();
        loadMahasiswaTable();
    }

    public void loadMahasiswaTable() {
        List<ModelMahasiswa> listMahasiswa = controller.getAllMahasiswa();
        ModelTabelMahasiswa tableModel = new ModelTabelMahasiswa(listM
        dataTable.setModel(tableModel);
    }

    private MahasiswaView() {
        throw new UnsupportedOperationException("Not supported yet.");
    }
}
```

### Penjelasan :


Kelas MahasiswaView adalah antarmuka pengguna (GUI) yang dibangun menggunakan Swing untuk mengelola data mahasiswa. Kelas ini berisi elemen-elemen seperti `JTextField` untuk input data (NPM, Nama, Semester, IPK), `JTable` untuk menampilkan daftar mahasiswa, dan tombol (`JButton`) untuk menyimpan, menghapus, dan me-refresh data.

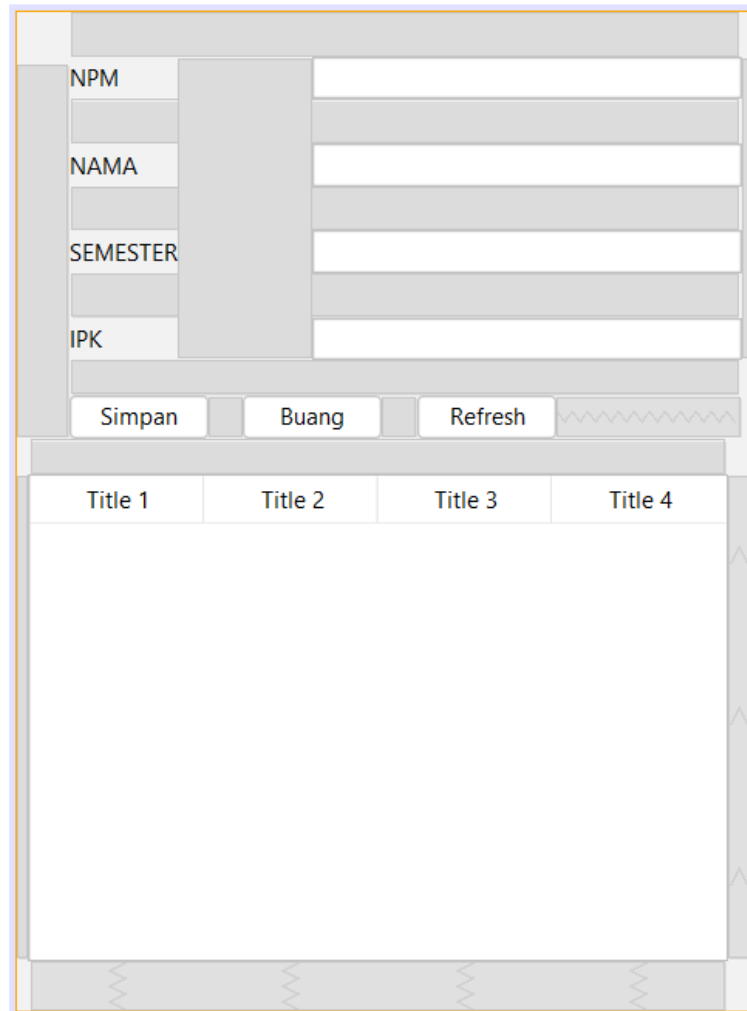
Metode loadMahasiswaTable() menampilkan data mahasiswa dari database ke dalam tabel. Tombol Simpan menyimpan data baru, Buang menghapus data berdasarkan ID, dan Refresh memperbarui tampilan data. Semua operasi ini terhubung dengan MahasiswaController, yang menangani logika bisnis dan database. Kelas ini memungkinkan pengguna untuk melakukan operasi CRUD pada data mahasiswa.



## Design



 The Preview Design button (in the toolbar) enables you to test the design of the form.



NPM	
NAMA	
SEMESTER	
IPK	

Simpan Buang Refresh

Title 1	Title 2	Title 3	Title 4

Penjelasan :

Input Data Mahasiswa:

NPM: Kolom input untuk mengisi Nomor Pokok Mahasiswa.

NAMA: Kolom input untuk mengisi nama mahasiswa.

SEMESTER: Kolom input untuk mengisi semester aktif mahasiswa.

IPK: Kolom input untuk mengisi Indeks Prestasi Kumulatif (IPK).

Tombol Operasi:

Simpan: Digunakan untuk menyimpan data yang telah dimasukkan ke dalam form.

Buang: Digunakan untuk menghapus data yang dipilih.

Refresh: Berfungsi untuk memperbarui tampilan atau membersihkan input form.

## ModelTabelMahasiswa.java

```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit t
4   */
5   package me.indah.model;
6
7   import javax.swing.table.AbstractTableModel;
8   import java.util.List;
9
10  public class ModelTabelMahasiswa extends AbstractTableModel{
11      private List<ModelMahasiswa> mahasiswaList;
12      private String[] columnNames = {"ID", "NPM", "Nama", "Semester", "IPK"};
13
14      public ModelTabelMahasiswa(List<ModelMahasiswa> mahasiswaList) {
15          this.mahasiswaList = mahasiswaList;
16      }
17
18      @Override
19      public int getRowCount() {
20          return mahasiswaList.size();
21      }
22
23      @Override
24      public int getColumnCount() {
25          return columnNames.length;
26      }
27
28      @Override
29      public Object getValueAt(int rowIndex, int columnIndex) {
30          ModelMahasiswa mahasiswa = mahasiswaList.get(rowIndex);
31          switch (columnIndex) {
32              case 0:
33                  return mahasiswa.getId();
34              case 1:
35                  return mahasiswa.getNpm();
36              case 2:
37                  return mahasiswa.getNama();
```

Penjelasan :

- mahasiswaList yang berisi data mahasiswa yang ditampilkan.
- columnNames yang mendefinisikan nama-nama kolom tabel (ID, NPM, Nama, Semester, IPK).
- getRowCount() dan getColumnCount() yang mengembalikan jumlah baris dan kolom tabel.
- getValueAt() yang mengambil nilai data untuk sel tabel berdasarkan baris dan kolom.
- isCellEditable() yang memastikan sel tabel tidak dapat diedit.
- setMahasiswaList() untuk memperbarui data tabel.

## Application.properties

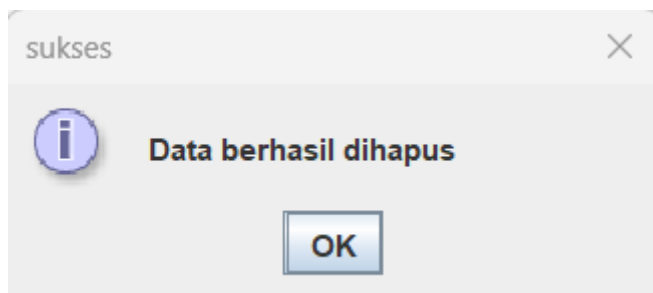
```
1  # Konfigurasi MySQL Hibernate
2  spring.datasource.url=jdbc:mysql://localhost:3306/spring_50421651?useSSL=false&
3  spring.datasource.username=root
4  spring.datasource.password=
5  spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
6
7  # Hibernate settings
8  spring.jpa.hibernate.ddl-auto=update
9  spring.jpa.show-sql=true
10
11  server.port=8082
```

### Penjelasan :

Konfigurasi di atas menghubungkan aplikasi Spring ke database MySQL dengan Hibernate. Beberapa pengaturan penting meliputi URL database, kredensial login (root tanpa kata sandi), dan driver JDBC terbaru. Hibernate diatur untuk memperbarui skema tanpa menghapus data, serta menampilkan query SQL yang dijalankan. Aplikasi dijalankan di port 8082. Dengan konfigurasi ini, aplikasi dapat mengelola skema database dan menjalankan query SQL.

**OUTPUT :**

<b>NPM</b>	<input type="text" value="50421651"/>
<b>NAMA</b>	<input type="text" value="indah"/>
<b>SEMESTER</b>	<input type="text" value="7"/>
<b>IPK</b>	<input type="text" value="3.9"/>
<div><input type="button" value="Simpan"/> <input type="button" value="Buang"/> <input type="button" value="Refresh"/></div>	



**NPM**

50421651

**NAMA**

indah

**SEMESTER**

7

**IPK**

3.9

**Simpan**

**Buang**

**Refresh**

ID	NPM	Nama	Semester	IPK