

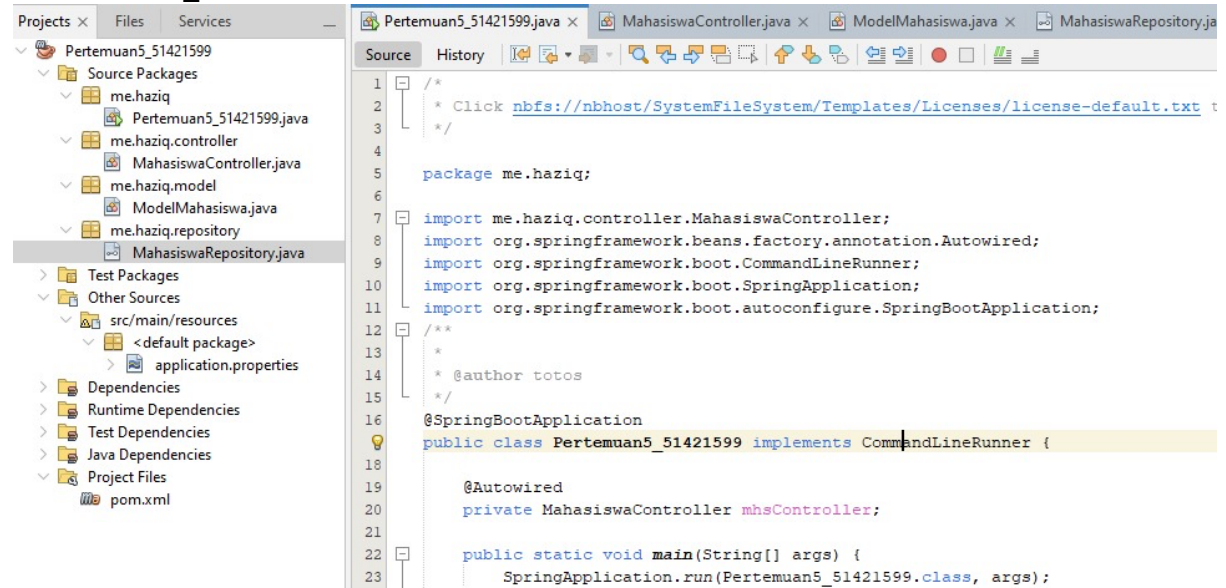
# **LAPORAN AKHIR PRAKTIKUM**

Mata Praktikum : RPL  
Kelas : 4IA14  
Praktikum ke- : 5  
Tanggal :  
Materi :  
NPM : 51421599  
Nama : Haziq Duha Zainul Ihsan  
Ketua Asisten : aji  
Jumlah Lembar :



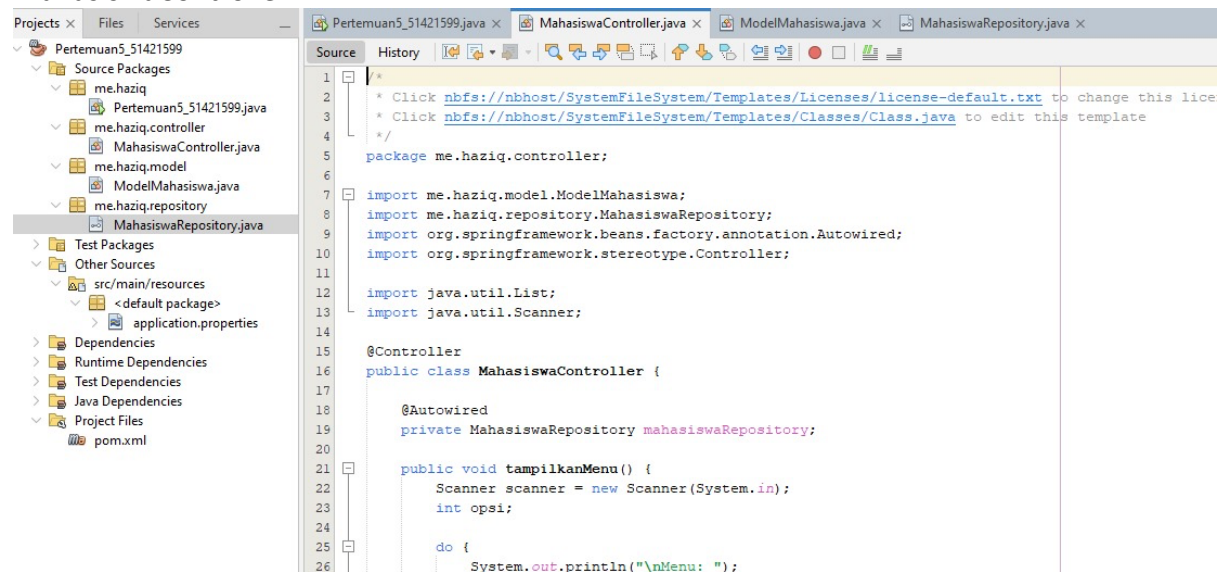
**LABORATORIUM TEKNIK INFORMATIKA  
UNIVERSITAS GUNADARMA  
2023**

## Pertemuan5\_51421599



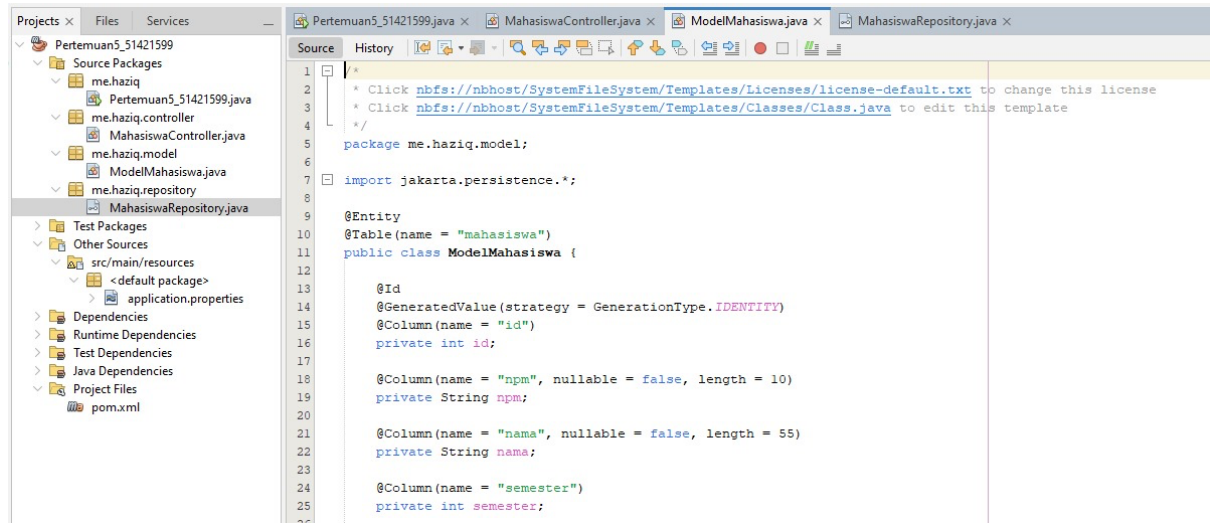
```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  */
4
5  package me.haziq;
6
7  import me.haziq.controller.MahasiswaController;
8  import org.springframework.beans.factory.annotation.Autowired;
9  import org.springframework.boot.CommandLineRunner;
10 import org.springframework.boot.SpringApplication;
11 import org.springframework.boot.autoconfigure.SpringBootApplication;
12
13 /**
14 *
15 * @author tolos
16 */
17 @SpringBootApplication
18 public class Pertemuan5_51421599 implements CommandLineRunner {
19
20     @Autowired
21     private MahasiswaController mhsController;
22
23     public static void main(String[] args) {
24         SpringApplication.run(Pertemuan5_51421599.class, args);
25     }
26 }
```

## MahasiswaController

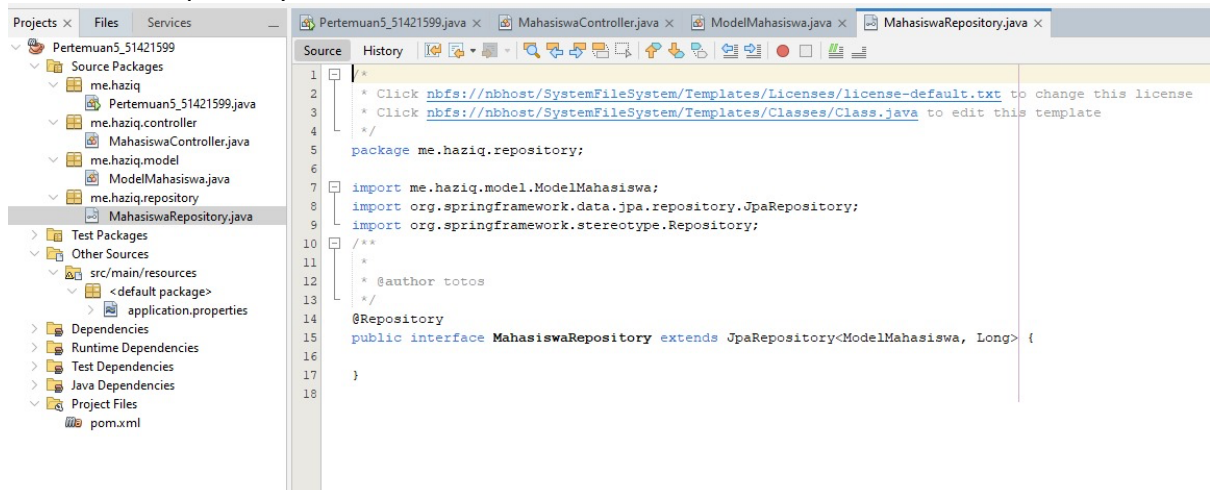


```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5
6  package me.haziq.controller;
7
8  import me.haziq.model.ModelMahasiswa;
9  import me.haziq.repository.MahasiswaRepository;
10 import org.springframework.beans.factory.annotation.Autowired;
11 import org.springframework.stereotype.Controller;
12
13 import java.util.List;
14 import java.util.Scanner;
15
16 @Controller
17 public class MahasiswaController {
18
19     @Autowired
20     private MahasiswaRepository mahasiswaRepository;
21
22     public void tampilkanMenu() {
23         Scanner scanner = new Scanner(System.in);
24         int opsi;
25
26         do {
27             System.out.println("\nMenu: ");
28         } while (opsi < 0 || opsi > 10);
29     }
30 }
```

## ModelMahasiswa



## MahasiswaRepository



## Output

Menu:

1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar

Pilih Opsi:

2

Masukkan NPM:

51421599

Masukkan Nama:

Haziq

Masukkan Semester:

8

Masukkan IPK:

4

Hibernate: insert into mahasiswa (ipk,nama,npm,semester) values (?,?,,?)

Mahasiswa Berhasil ditambahkan.

```

Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih Opsi:
1
Hibernate: select mml_0.id,mml_0.ipk,mml_0.nama,mml_0.npm,mml_0.semester from mahasiswa mml_0
Mahasiswa(id=1, npm='51421599', nama='Haziq', semester=8, ipk='4.0')

```

```

Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih Opsi:
3
Hibernate: select mml_0.id,mml_0.ipk,mml_0.nama,mml_0.npm,mml_0.semester from mahasiswa mml_0
Koneksi ke database berhasil

```

```

Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih Opsi:
4
Keluar dari program

```

## Penjelasan:

### ModelMahasiswa.java

- Baris dengan anotasi `@Entity` dan `@Table(name = "mahasiswa")` menunjukkan bahwa kelas ini adalah entitas JPA yang dipetakan ke tabel mahasiswa di database.
- Atribut `@Id` dan `@GeneratedValue(strategy = GenerationType.IDENTITY)` pada id menunjukkan bahwa kolom ini adalah *primary key* yang nilainya di-generate secara otomatis oleh database.
- Anotasi `@Column` pada atribut lain, seperti npm, nama, semester, dan ipk, mengatur pemetaan setiap atribut ke kolom dalam tabel serta menambahkan aturan seperti panjang maksimum atau larangan nilai null.
- Metode konstruktor mendukung pembuatan objek dengan nilai awal, sementara metode getter dan setter memungkinkan akses dan modifikasi data. Metode `toString` memberikan representasi string untuk menampilkan data mahasiswa secara ringkas.

### MahasiswaController.java

- Atribut `mahasiswaRepository` dideklarasikan dengan anotasi `@Autowired`, memungkinkan Spring Boot untuk secara otomatis menghubungkannya ke implementasi `MahasiswaRepository`.
- Metode `tampilkanMenu` berisi logika menu berbasis konsol yang diimplementasikan menggunakan `Scanner`. Pengguna dapat memilih opsi seperti menampilkan semua data mahasiswa, menambah data baru, atau memeriksa koneksi database.
- Pada opsi 1, metode `tampilkanSemuaMahasiswa` memanggil `findAll()` dari repository untuk mengambil semua data mahasiswa dari database dan mencetaknya ke konsol. Jika data kosong, pesan akan ditampilkan.
- Pada opsi 2, metode `tambahMahasiswa` meminta input pengguna melalui konsol untuk membuat objek `ModelMahasiswa` baru, lalu menyimpannya ke database menggunakan metode `save` dari repository.
- Pada opsi 3, metode `cekKoneksi` memanggil `findAll()` untuk memastikan koneksi ke database berfungsi. Jika terjadi kesalahan, pesan kesalahan ditampilkan.

### **MahasiswaRepository.java**

- Interface ini memperluas JpaRepository dan memberikan implementasi bawaan untuk operasi CRUD, seperti findAll, save, deleteById, dan lainnya, tanpa perlu menulis kode tambahan. Repository ini dipetakan ke model ModelMahasiswa.

### **pert5.java**

- Anotasi @SpringBootApplication menandai kelas ini sebagai titik awal aplikasi Spring Boot.
- Dalam metode main, aplikasi dijalankan menggunakan SpringApplication.run, yang secara otomatis memulai konteks Spring.
- Kelas ini mengimplementasikan CommandLineRunner, sehingga metode run akan dieksekusi setelah aplikasi dimulai. Di dalamnya, metode tampilkanMenu dipanggil untuk menampilkan menu interaktif kepada pengguna.

### **Logika Keseluruhan**

Aplikasi ini berfungsi sebagai pengelola data mahasiswa. Ketika dijalankan, aplikasi memulai menu interaktif yang memungkinkan pengguna melihat data mahasiswa, menambah data baru, atau memeriksa koneksi database. Setiap operasi dikelola oleh controller (MahasiswaController) yang memanfaatkan repository (MahasiswaRepository) untuk berkomunikasi dengan database. Model (ModelMahasiswa) bertindak sebagai representasi data mahasiswa di dalam kode dan database, memastikan data tetap terorganisir dan konsisten.