Nama : Muhammad Izzuddin Almansyur
NPM : 51422084
Kelas : 4IA07
Praktikum : Rekayasa Perangkat Lunak 2, ACT 5

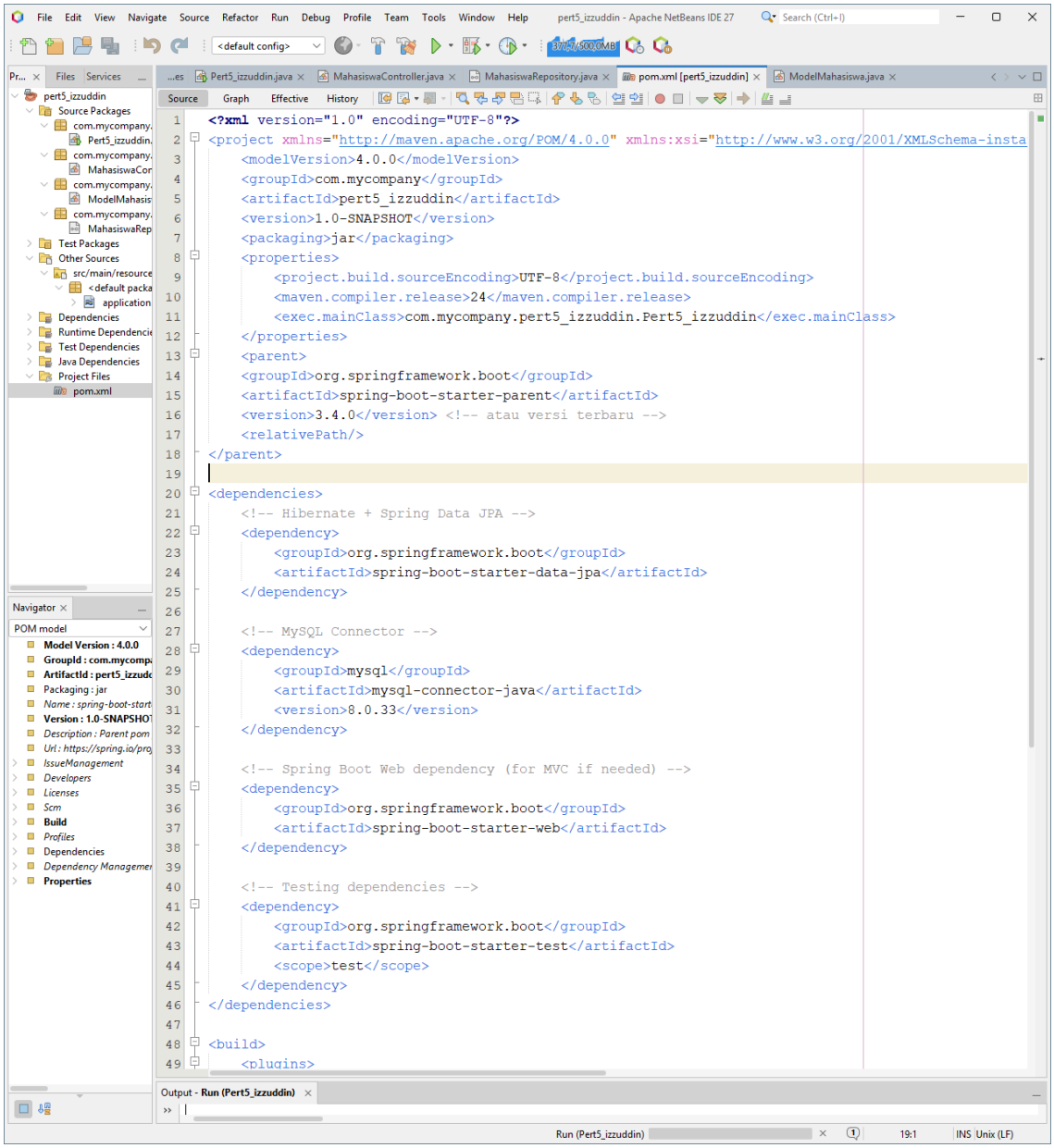
1. Dependency Injection (DI) adalah konsep penting dalam Spring Framework. Jelaskan apa itu Dependency Injection mengapa DI sangat penting dalam pengembangan aplikasi berbasis Spring Boot

   Dependency Injection adalah sebuah *design pattern* di mana *dependency* (objek yang dibutuhkan oleh suatu kelas) tidak dibuat langsung di dalam kelas tersebut, melainkan diberikan (*injected*) dari luar oleh sebuah *container* atau *framework*. Dalam Spring, *container* ini disebut **IoC Container (Inversion of Control Container)**.

   Mengapa Dependency Injection sangat penting dalam pengembangan aplikasi berbasis Spring Boot? Karena Dependency Injection memberikan fleksibilitas dan modularitas yang tinggi. Dengan DI, komponen dalam aplikasi tidak perlu membuat objek dependensi sendiri, melainkan menerima objek tersebut dari luar melalui IoC Container. Hal ini menciptakan **loose coupling**, yaitu mengurangi ketergantungan langsung antar kelas sehingga setiap komponen lebih mudah diganti, diperluas, atau diuji tanpa memengaruhi bagian lain. Selain itu, DI meningkatkan **testability** karena memungkinkan penggunaan *mock objects* dalam pengujian, sehingga developer dapat menguji logika bisnis tanpa harus bergantung pada implementasi nyata seperti database atau layanan eksternal. Dari sisi **maintainability**, aplikasi menjadi lebih mudah dirawat karena perubahan pada satu modul tidak menimbulkan efek domino pada modul lain. Spring Boot juga mempermudah pengelolaan konfigurasi dengan menyediakan anotasi seperti **`@Autowired`**, **`@Component`**, **`@Service`**, dan **`@Repository`** yang secara otomatis mengatur dependency, sehingga developer tidak perlu menulis kode boilerplate untuk membuat dan menghubungkan objek. Pada akhirnya, DI mendukung **scalability dan flexibility** aplikasi, karena struktur yang modular memungkinkan aplikasi berkembang dengan cepat dan aman seiring bertambahnya kebutuhan bisnis.

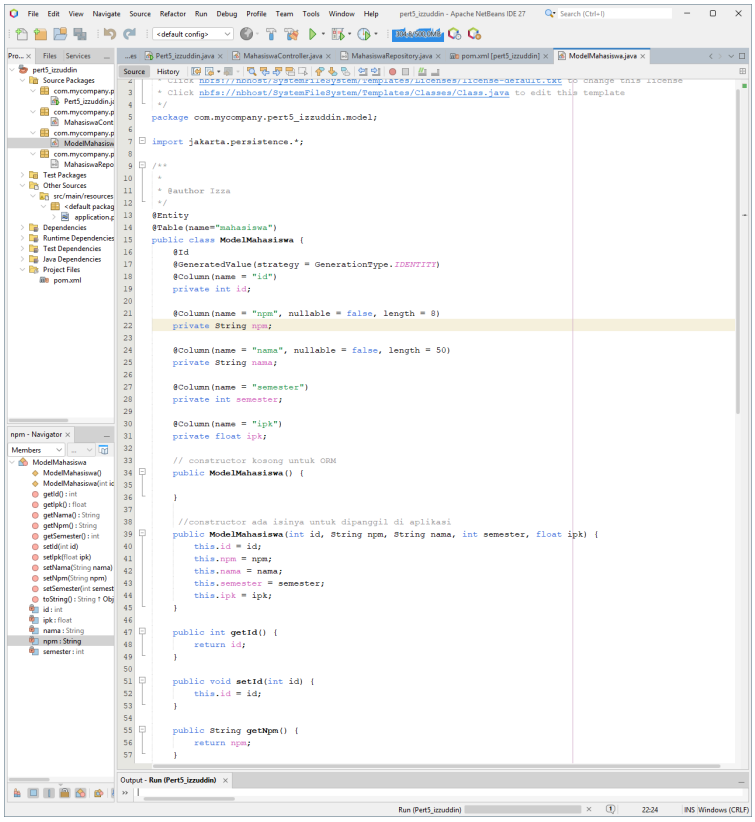
2. Screenshoot kode dan output program!

Pom.xml

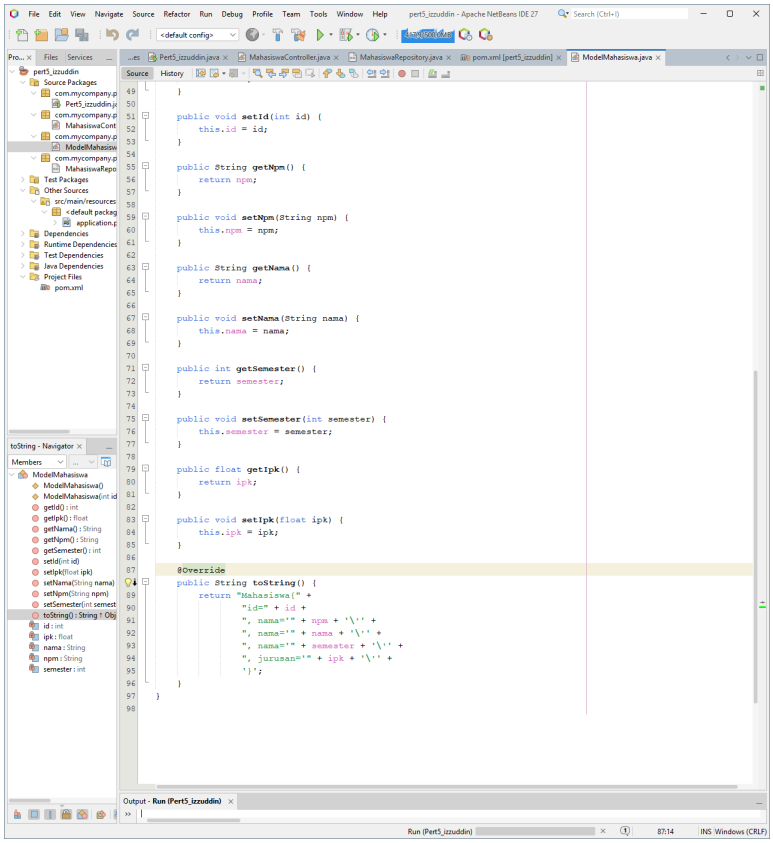



```xml
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001/XMLSchema-insta
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.mycompany</groupId>
    <artifactId>pert5_izzuddin</artifactId>
    <version>1.0-SNAPSHOT</version>
    <packaging>jar</packaging>
    <properties>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
        <maven.compiler.release>24</maven.compiler.release>
        <exec.mainClass>com.mycompany.pert5_izzuddin.Pert5_izzuddin</exec.mainClass>
    </properties>
    <parent>
    <groupId>org.springframework.boot</groupId>
    <artifactId>spring-boot-starter-parent</artifactId>
    <version>3.4.0</version> <!-- atau versi terbaru -->
    <relativePath/>
</parent>

<dependencies>
    <!-- Hibernate + Spring Data JPA -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-data-jpa</artifactId>
    </dependency>

    <!-- MySQL Connector -->
    <dependency>
        <groupId>mysql</groupId>
        <artifactId>mysql-connector-java</artifactId>
        <version>8.0.33</version>
    </dependency>

    <!-- Spring Boot Web dependency (for MVC if needed) -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-web</artifactId>
    </dependency>

    <!-- Testing dependencies -->
    <dependency>
        <groupId>org.springframework.boot</groupId>
        <artifactId>spring-boot-starter-test</artifactId>
        <scope>test</scope>
    </dependency>
</dependencies>

<build>
    <plugins>
```

ModelMahasiswa

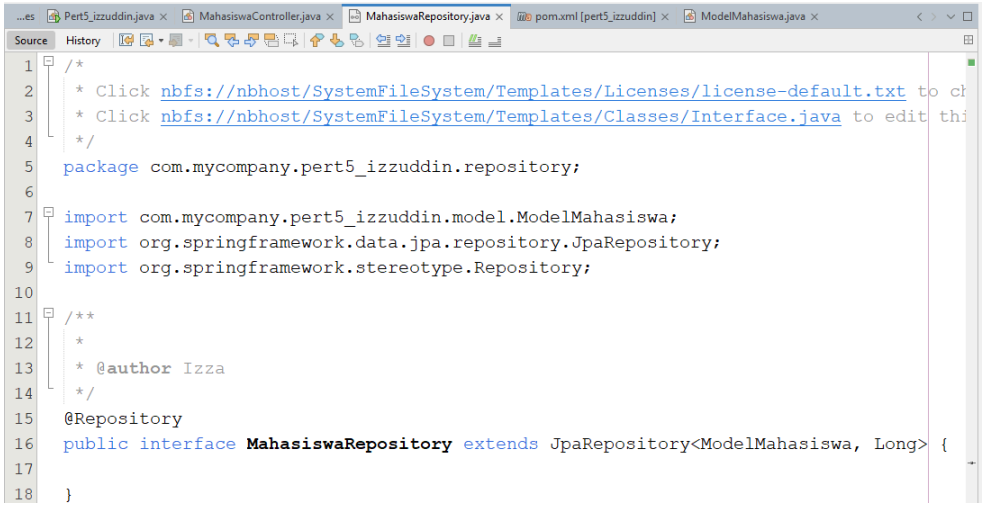


```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.pert5_izzuddin.model;

import jakarta.persistence.*;

/**
 *
 * @author Izza
 */
@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;

    // constructor kosong untuk ORM
    public ModelMahasiswa() {

    }

     //constructor ada isinya untuk dipanggil di aplikasi
    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;
    }
```



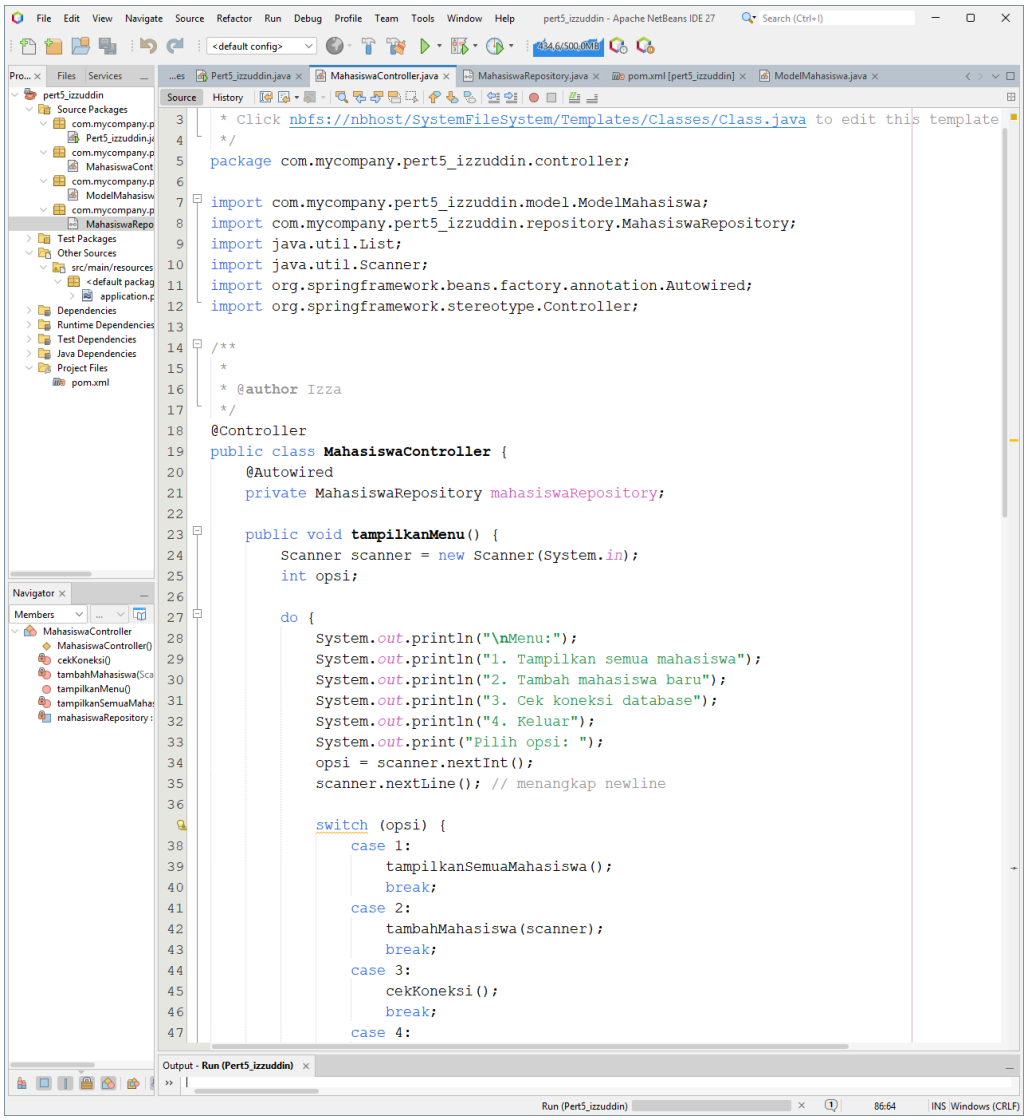
```
    }

    public void 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;
    }

    @Override
    public String toString() {
        return "Mahasiswa{" +
                "id=" + id +
                ", nama='" + npm + '\'' +
                ", nama='" + nama + '\'' +
                ", nama='" + semester + '\'' +
                ", jurusan='" + ipk + '\'' +
                '}';
    }
}
```

MahasiswaRepository

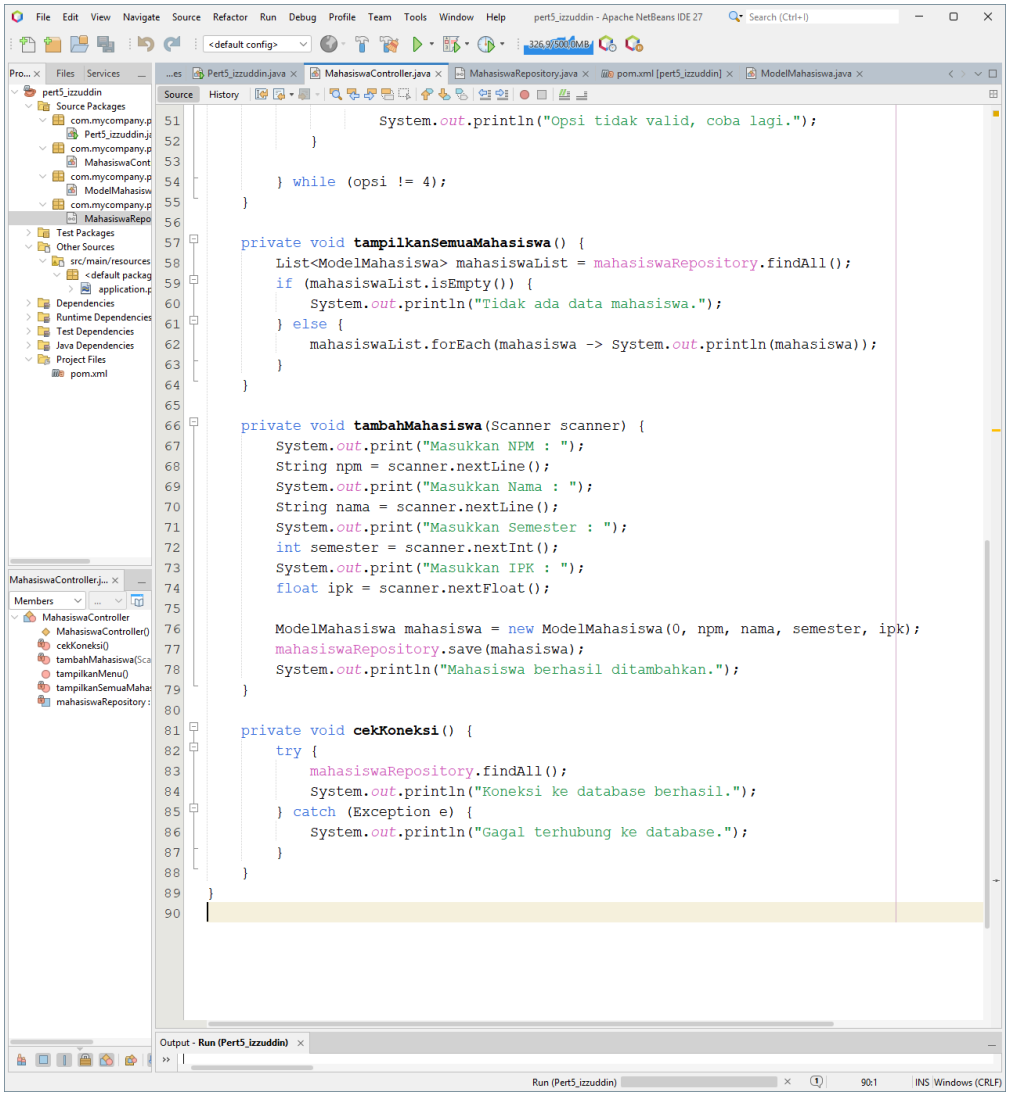


```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to ch
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Interface.java to edit thi
 */
package com.mycompany.pert5_izzuddin.repository;

import com.mycompany.pert5_izzuddin.model.ModelMahasiswa;
import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

/**
 *
 * @author Izza
 */
@Repository
public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Long> {

}
```

MahasiswaController

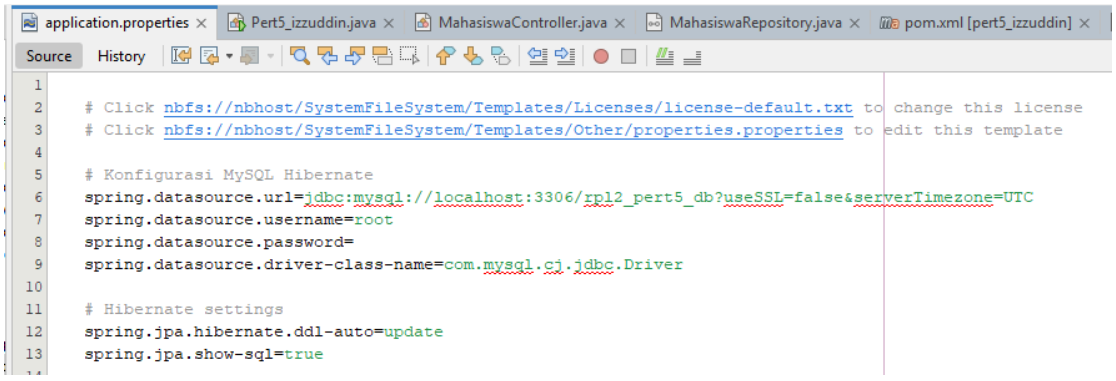


```java
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
 */
package com.mycompany.pert5_izzuddin.controller;

import com.mycompany.pert5_izzuddin.model.ModelMahasiswa;
import com.mycompany.pert5_izzuddin.repository.MahasiswaRepository;
import java.util.List;
import java.util.Scanner;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Controller;

/**
 *
 * @author Izza
 */
@Controller
public class MahasiswaController {
    @Autowired
    private MahasiswaRepository mahasiswaRepository;

    public void tampilkanMenu() {
        Scanner scanner = new Scanner(System.in);
        int opsi;

        do {
            System.out.println("\nMenu:");
            System.out.println("1. Tampilkan semua mahasiswa");
            System.out.println("2. Tambah mahasiswa baru");
            System.out.println("3. Cek koneksi database");
            System.out.println("4. Keluar");
            System.out.print("Pilih opsi: ");
            opsi = scanner.nextInt();
            scanner.nextLine(); // menangkap newline

            switch (opsi) {
                case 1:
                    tampilkanSemuaMahasiswa();
                    break;
                case 2:
                    tambahMahasiswa(scanner);
                    break;
                case 3:
                    cekKoneksi();
                    break;
                case 4:
```



```
51                      System.out.println("Opsi tidak valid, coba lagi.");
52                  }
53
54              } while (opsi != 4);
55          }
56
57          private void tampilkanSemuaMahasiswa() {
58              List<ModelMahasiswa> mahasiswaList = mahasiswaRepository.findAll();
59              if (mahasiswaList.isEmpty()) {
60                  System.out.println("Tidak ada data mahasiswa.");
61              } else {
62                  mahasiswaList.forEach(mahasiswa -> System.out.println(mahasiswa));
63              }
64          }
65
66          private void tambahMahasiswa(Scanner scanner) {
67              System.out.print("Masukkan NPM : ");
68              String npm = scanner.nextLine();
69              System.out.print("Masukkan Nama : ");
70              String nama = scanner.nextLine();
71              System.out.print("Masukkan Semester : ");
72              int semester = scanner.nextInt();
73              System.out.print("Masukkan IPK : ");
74              float ipk = scanner.nextFloat();
75
76              ModelMahasiswa mahasiswa = new ModelMahasiswa(0, npm, nama, semester, ipk);
77              mahasiswaRepository.save(mahasiswa);
78              System.out.println("Mahasiswa berhasil ditambahkan.");
79          }
80
81          private void cekKoneksi() {
82              try {
83                  mahasiswaRepository.findAll();
84                  System.out.println("Koneksi ke database berhasil.");
85              } catch (Exception e) {
86                  System.out.println("Gagal terhubung ke database.");
87              }
88          }
89      }
90
```

Application.propertis



```
1
2   # Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   # Click nbfs://nbhost/SystemFileSystem/Templates/Other/properties.properties to edit this template
4
5   # Konfigurasi MySQL Hibernate
6   spring.datasource.url=jdbc:mysql://localhost:3306/rpl2_pert5_db?useSSL=false&serverTimezone=UTC
7   spring.datasource.username=root
8   spring.datasource.password=
9   spring.datasource.driver-class-name=com.mysql.cj.jdbc.Driver
10
11  # Hibernate settings
12  spring.jpa.hibernate.ddl-auto=update
13  spring.jpa.show-sql=true
```

Main

application.properties | Pert5_izzuddin.java | MahasiswaController.java | MahasiswaRepository.java | pom.xml [pert5_izzuddin] | ModelMahasiswa....

Source | History

```java
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 */

package com.mycompany.pert5_izzuddin;

import com.mycompany.pert5_izzuddin.controller.MahasiswaController;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.boot.CommandLineRunner;
import org.springframework.boot.SpringApplication;
import org.springframework.boot.autoconfigure.SpringBootApplication;

/**
 *
 * @author Izza
 */
@SpringBootApplication
public class Pert5_izzuddin implements CommandLineRunner{
    @Autowired
    private MahasiswaController mhsController;
    public static void main(String[] args) {
        SpringApplication.run(Pert5_izzuddin.class, args);
    }

    @Override
    public void run(String... args) throws Exception {
        mhsController.tampilkanMenu();
    }

}
```

Output:

Output - Run (Pert5_izzuddin)

```
Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih opsi: 3
Hibernate: select mm1_0.id,mm1_0.ipk,mm1_0.nama,mm1_0.npm,mm1_0.semester from mahasiswa mm1_0
Koneksi ke database berhasil.

Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih opsi: 1
Hibernate: select mm1_0.id,mm1_0.ipk,mm1_0.nama,mm1_0.npm,mm1_0.semester from mahasiswa mm1_0
Tidak ada data mahasiswa.

Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih opsi: 2
Masukkan NPM : 51422084
Masukkan Nama : Izzuddin
Masukkan Semester : 7
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 mm1_0.id,mm1_0.ipk,mm1_0.nama,mm1_0.npm,mm1_0.semester from mahasiswa mm1_0
Mahasiswa{id=1, nama='51422084', nama='Izzuddin', nama='7', jurusan='4.0'}

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


Server: 127.0.0.1 » Database: rpl2_pert5_db » Table: mahasiswa
Browse
Structure
SQL
Search
Insert
Export
Import
Privileges
Operations
Showing rows 0 - 0 (1 total, Query took 0.0002 seconds.)
SELECT * FROM `mahasiswa`
Profiling [Edit inline] [Edit] [Explain SQL] [Create PHP code] [Refresh]
Show all
Number of rows: 25
Filter rows: Search this table
Extra options
id
ipk
nama
npm
semester
Edit
Copy
Delete
1
4
Izzuddin
51422084
7