

## ACTIVITY PERTEMUAN 5

**NAMA** : Muhammad Rifqi Dwi Putra  
**NPM** : 51422145  
**KELAS** : 4IA08  
**MATERI** : Konsep framework spring, pembuatan project spring dan hibernate  
**MATA PRAKTIKUM** : Rekayasa Perangkat Lunak 2

---

(Screenshoot langkah-langkah sesuai video pembelajaran dan jelaskan dengan ringkas)

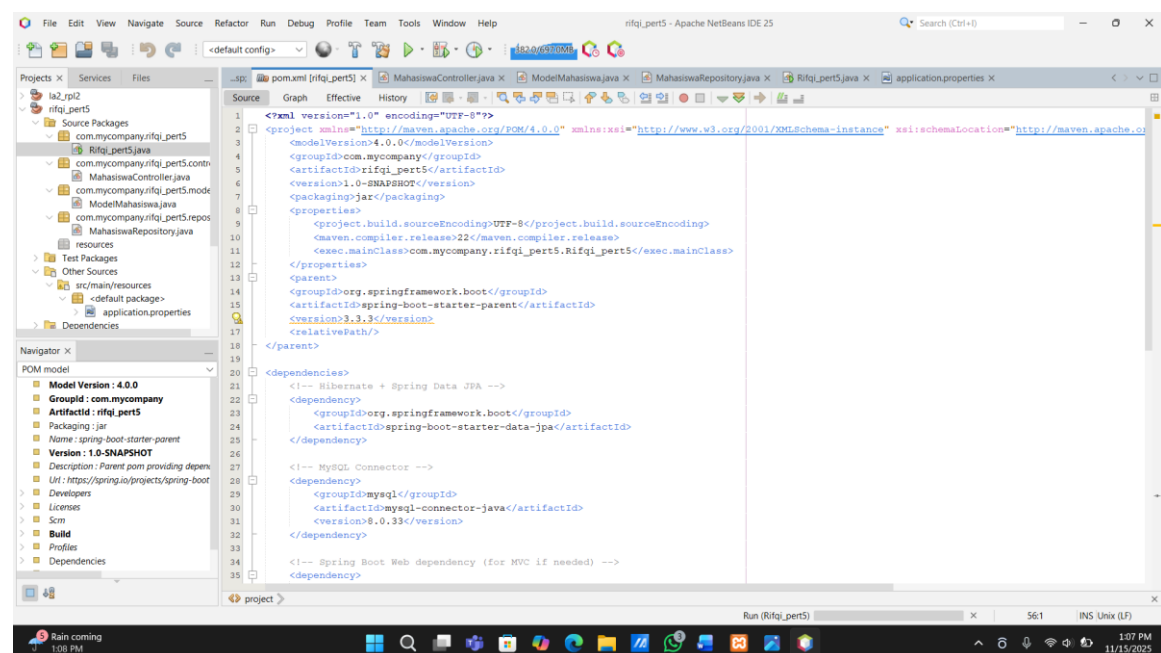
1. Dependency Injection (DI) adalah konsep penting dalam Spring Framework. Jelaskan apa itu Dependency Injection dan mengapa DI sangat penting dalam pengembangan aplikasi berbasis Spring Boot.

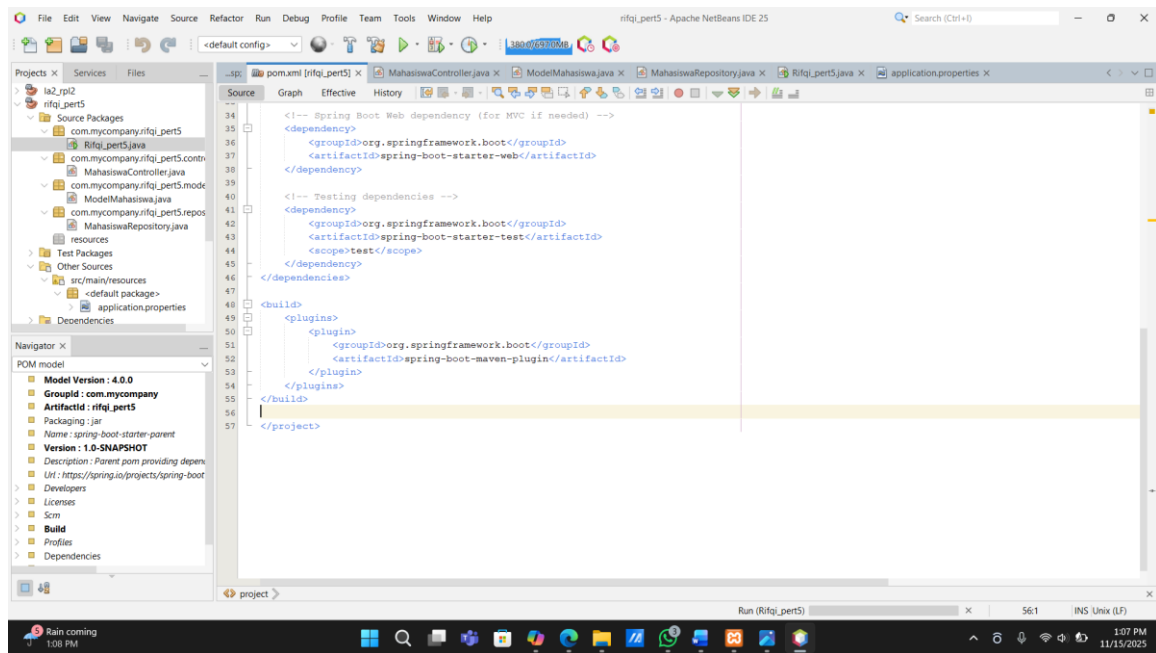
Dependency Injection (DI) adalah teknik dalam Spring Framework yang memungkinkan objek-objek yang dibutuhkan oleh suatu class disediakan oleh container, bukan dibuat langsung oleh class tersebut. Dengan DI, komponen dalam aplikasi menjadi lebih modular dan mudah diuji karena dependensi dapat disuntikkan secara otomatis oleh Spring Boot. Hal ini mempercepat pengembangan, mempermudah pemeliharaan, dan mendukung prinsip pemisahan tanggung jawab antar komponen.

2. Screenshot Kode dan Output Program!

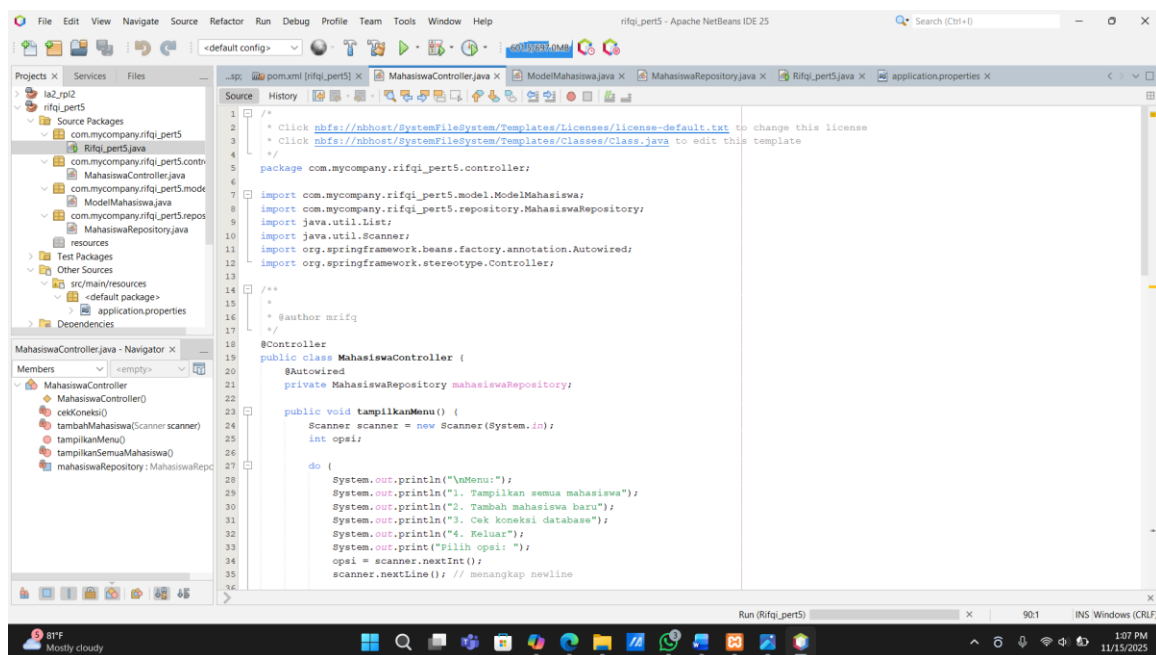
Kode :

Pom.xml





## MahasiswaController.java



```
switch (opsi) {
    case 1:
        tampilkanSemuaMahasiswa();
        break;
    case 2:
        tambahMahasiswa(scanner);
        break;
    case 3:
        cekKoneksi();
        break;
    case 4:
        System.out.println("Keluar dari program.");
        break;
    default:
        System.out.println("Opsi tidak valid, coba lagi.");
}

} while (opsi != 4);

private void tampilkanSemuaMahasiswa() {
    List<ModelMahasiswa> mahasiswaList = mahasiswaRepository.findAll();
    if (mahasiswaList.isEmpty()) {
        System.out.println("Tidak ada data mahasiswa.");
    } else {
        mahasiswaList.forEach(mahasiswa -> System.out.println(mahasiswa));
    }
}

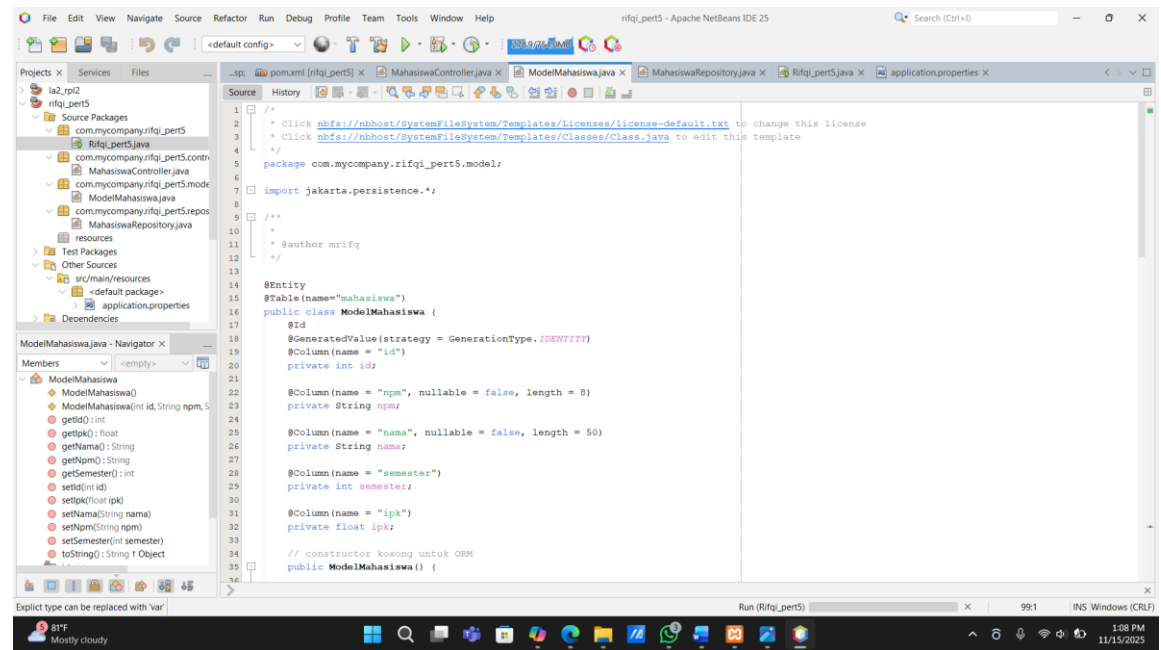
private void tambahMahasiswa(Scanner scanner) {
    System.out.print("Masukkan NPM : ");
    String npm = scanner.nextLine();
    System.out.print("Masukkan Nama : ");
    String nama = scanner.nextLine();
    System.out.print("Masukkan Semester : ");
    int semester = scanner.nextInt();
}
```

```
private void tambahMahasiswa(Scanner scanner) {
    System.out.print("Masukkan NPM : ");
    String npm = scanner.nextLine();
    System.out.print("Masukkan Nama : ");
    String nama = scanner.nextLine();
    System.out.print("Masukkan Semester : ");
    int semester = scanner.nextInt();
    System.out.print("Masukkan IPK : ");
    float ipk = scanner.nextFloat();

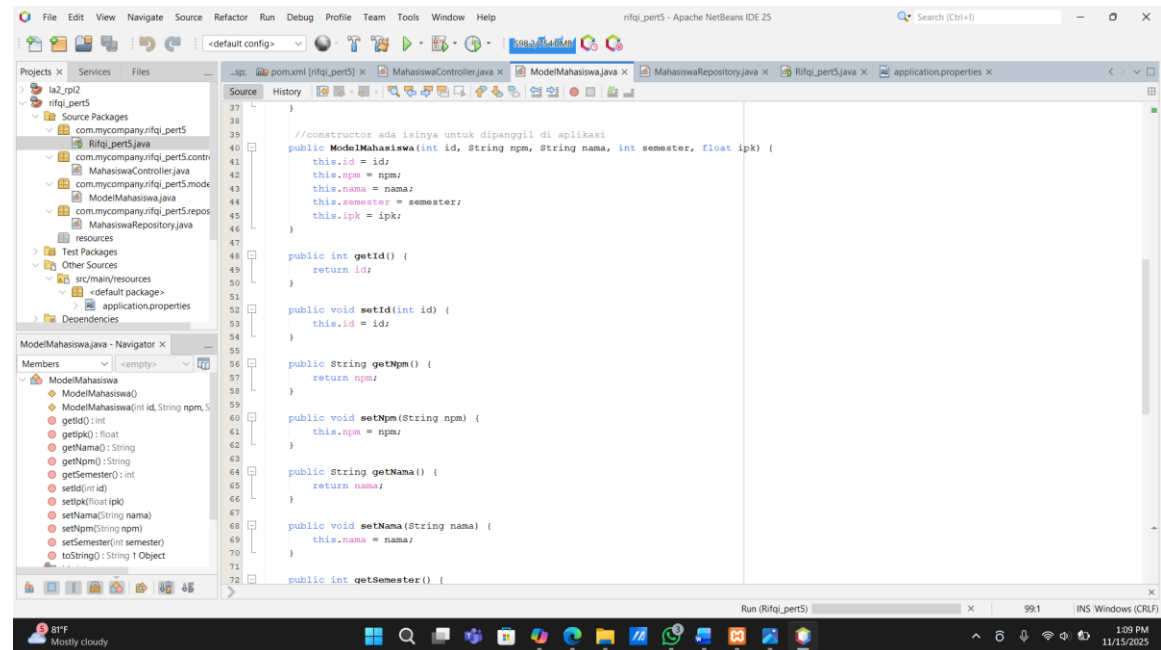
    ModelMahasiswa mahasiswa = new ModelMahasiswa(0, npm, nama, semester, ipk);
    mahasiswaRepository.save(mahasiswa);
    System.out.println("Mahasiswa berhasil ditambahkan.");
}

private void cekKoneksi() {
    try {
        mahasiswaRepository.findAll();
        System.out.println("Koneksi ke database berhasil.");
    } catch (Exception e) {
        System.out.println("Gagal terhubung ke database.");
    }
}
```

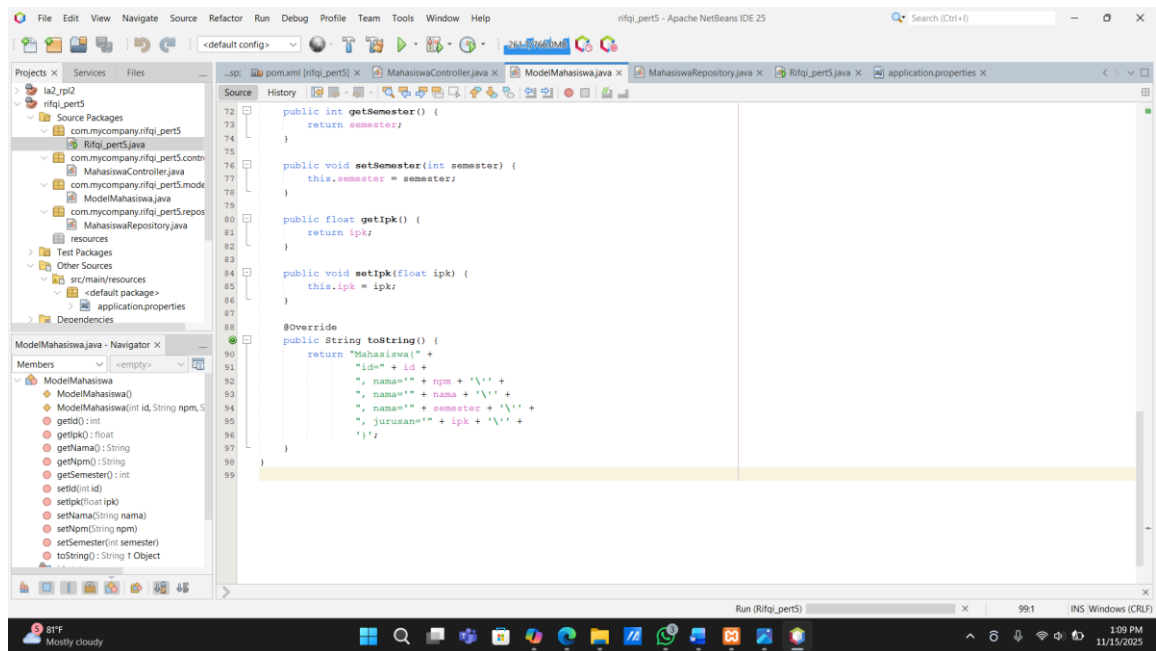
# ModelMahasiswa.java



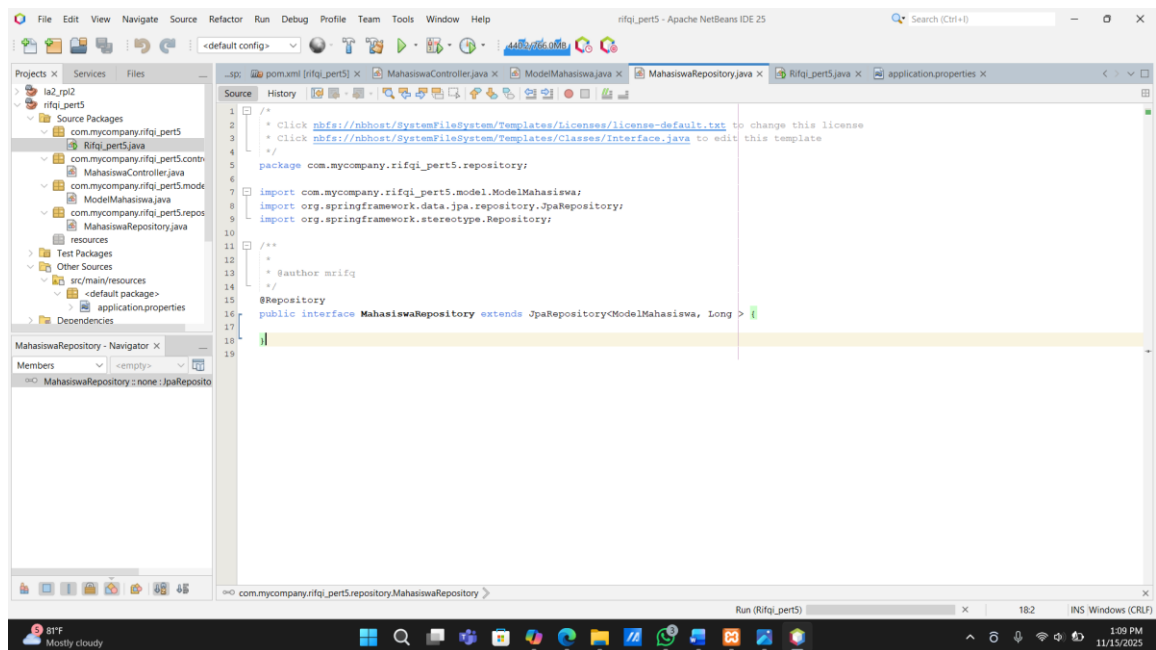
```
1 1
2 2
3 3
4 4
5 5 package com.mycompany.rifqi_pert5.model;
6 6
7 7 import jakarta.persistence.*;
8 8
9 9
10 10
11 11
12 12
13 13
14 14 @Entity(name="mahasiswa")
15 15 @Table(name="mahasiswa")
16 16 public class ModelMahasiswa {
17 17     @Id
18 18     @GeneratedValue(strategy = GenerationType.IDENTITY)
19 19     @Column(name = "id")
20 20     private int id;
21 21
22 22     @Column(name = "npm", nullable = false, length = 8)
23 23     private String npm;
24 24
25 25     @Column(name = "nama", nullable = false, length = 50)
26 26     private String nama;
27 27
28 28     @Column(name = "semester")
29 29     private int semester;
30 30
31 31     @Column(name = "ipk")
32 32     private float ipk;
33 33
34 34 // constructor kosong untuk ORM
35 35 public ModelMahasiswa() {
36 36
37 37
38 38
39 39
40 40
41 41
42 42
43 43
44 44
45 45
46 46
47 47
48 48
49 49
50 50
51 51
52 52
53 53
54 54
55 55
56 56
57 57
58 58
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 68
69 69
70 70
71 71
72 72
```



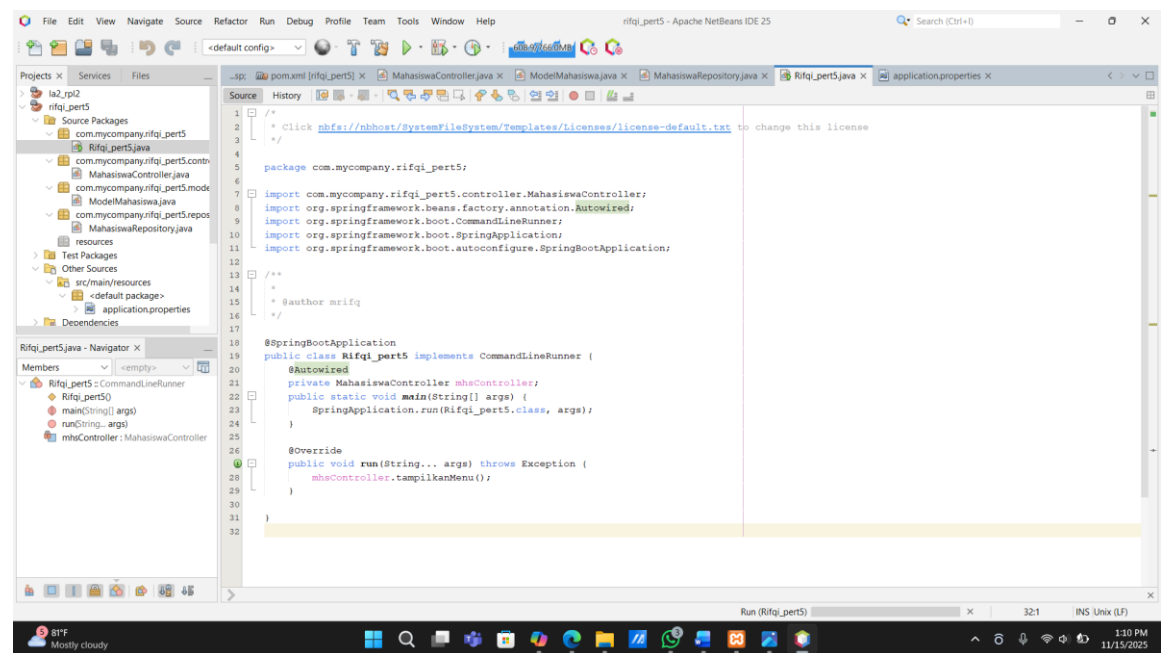
```
37 37
38 38
39 39 //constructor ada isinya untuk dipanggil di aplikasi
40 40 public ModelMahasiswa(int id, String npm, String nama, int semester, float ipk) {
41 41     this.id = id;
42 42     this.npm = npm;
43 43     this.nama = nama;
44 44     this.semester = semester;
45 45     this.ipk = ipk;
46 46
47 47
48 48
49 49
50 50
51 51
52 52
53 53
54 54
55 55
56 56
57 57
58 58
59 59
60 60
61 61
62 62
63 63
64 64
65 65
66 66
67 67
68 68
69 69
70 70
71 71
72 72
```



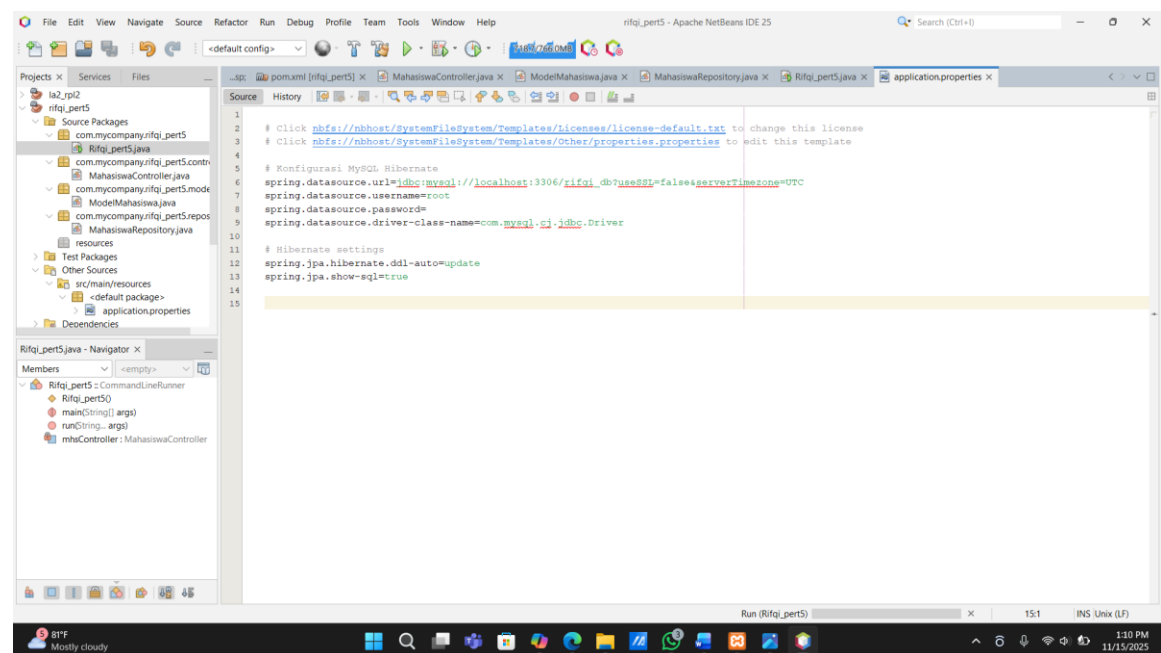
## MahasiswaRepository.java



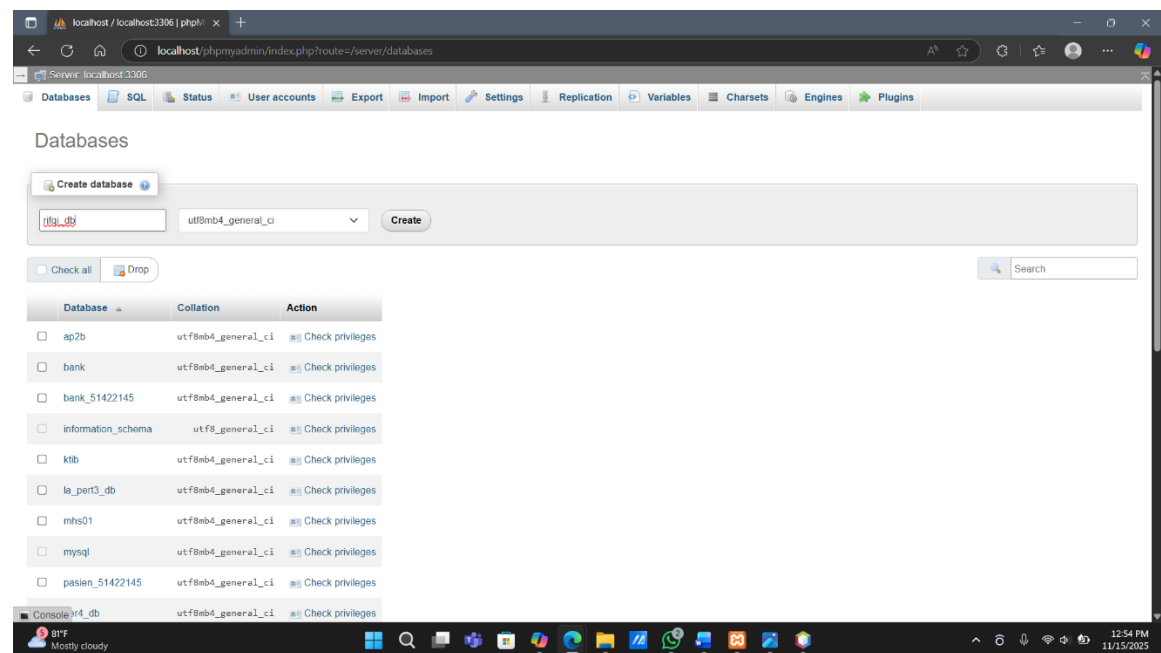
## Rifqi\_pert5.java



## Application.properties



## Membuat tabel di phpmyadmin



Output :

Menu 1 :

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 2 :

Menu:

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

Pilih opsi: 2

Masukkan NPM : 51422145

Masukkan Nama : Muhammad Rifqi Dwi Putra

Masukkan Semester : 7

Masukkan IPK : 3.89

Hibernate: insert into mahasiswa (ipk,nama,npm,semester) values (?, ?, ?, ?)  
Mahasiswa berhasil ditambahkan.

### Menu 1 :

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, nama='51422145', nama='Muhammad Rifqi Dwi Putra', nama='7', jurusan='3.89'}

### Menu 3 :

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 4 :

Menu:

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

Pilih opsi: 4

Keluar dari program.