

ACTIVITY PERTEMUAN 5

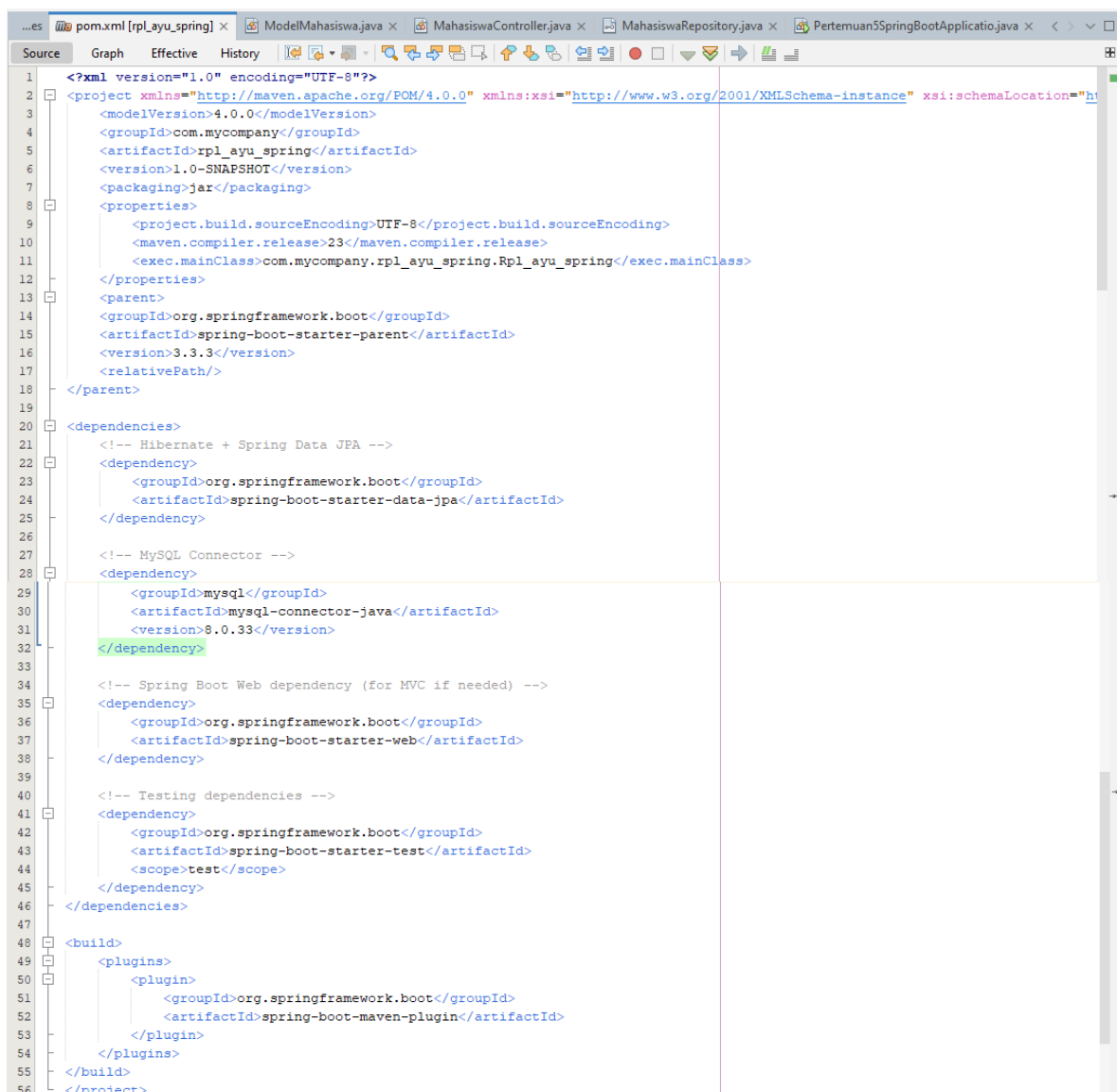
NAMA : AYU RETNONINGRUM SUSILO
NPM : 50421244
KELAS : 4IA28
MATERI : SPRING
MATA PRAKTIKUM : REKAYASA PERANGKAT LUNAK 2

Screenshot source code dan output

Jawab:

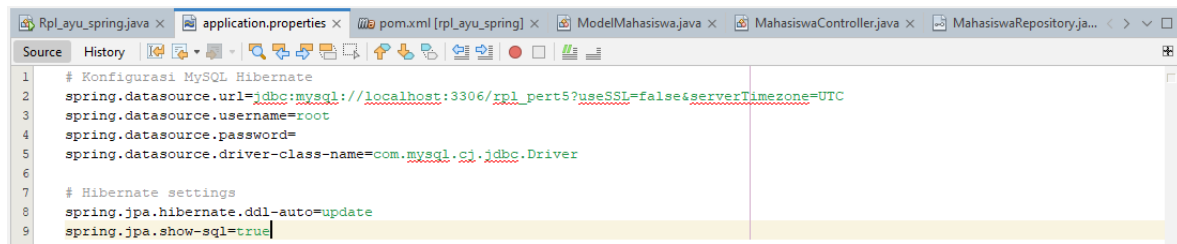
SOURCE CODE

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/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4   <groupId>com.mycompany</groupId>
5   <artifactId>rpl_ayu_spring</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.release>23</maven.compiler.release>
11    <exec.mainClass>com.mycompany.rpl_ayu_spring.Rpl_ayu_spring</exec.mainClass>
12  </properties>
13  <parent>
14    <groupId>org.springframework.boot</groupId>
15    <artifactId>spring-boot-starter-parent</artifactId>
16    <version>3.3.3</version>
17    <relativePath/>
18  </parent>
19
20  <dependencies>
21    <!-- Hibernate + Spring Data JPA -->
22    <dependency>
23      <groupId>org.springframework.boot</groupId>
24      <artifactId>spring-boot-starter-data-jpa</artifactId>
25    </dependency>
26
27    <!-- MySQL Connector -->
28    <dependency>
29      <groupId>mysql</groupId>
30      <artifactId>mysql-connector-java</artifactId>
31      <version>8.0.33</version>
32    </dependency>
33
34    <!-- Spring Boot Web dependency (for MVC if needed) -->
35    <dependency>
36      <groupId>org.springframework.boot</groupId>
37      <artifactId>spring-boot-starter-web</artifactId>
38    </dependency>
39
40    <!-- Testing dependencies -->
41    <dependency>
42      <groupId>org.springframework.boot</groupId>
43      <artifactId>spring-boot-starter-test</artifactId>
44      <scope>test</scope>
45    </dependency>
46  </dependencies>
47
48  <build>
49    <plugins>
50      <plugin>
51        <groupId>org.springframework.boot</groupId>
52        <artifactId>spring-boot-maven-plugin</artifactId>
53      </plugin>
54    </plugins>
55  </build>
56 </project>
```

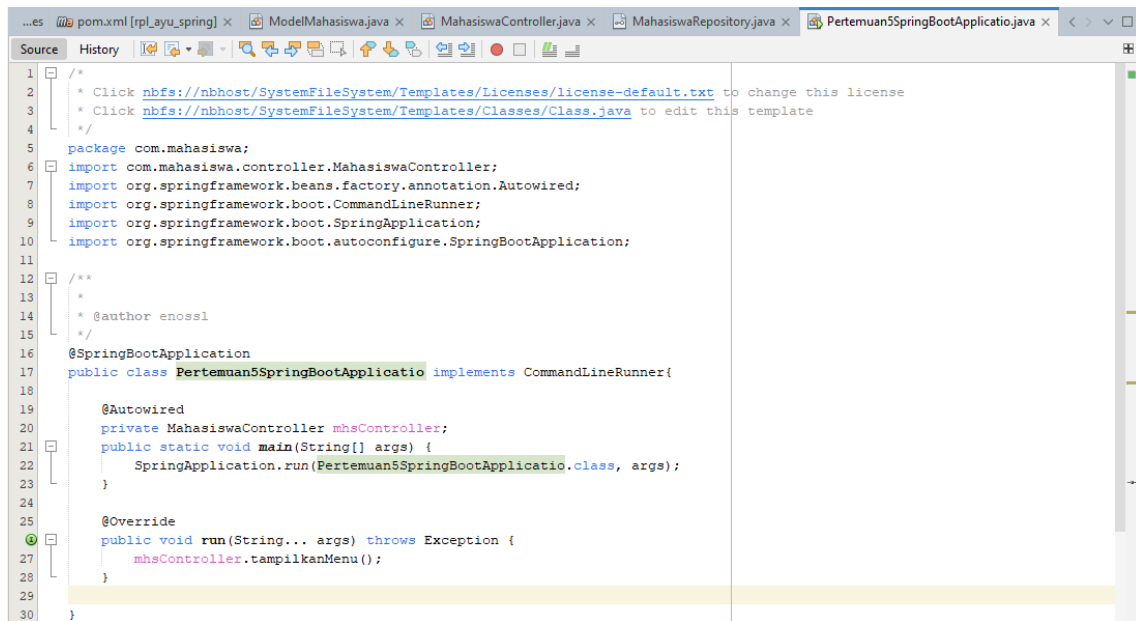
Application.properties



The screenshot shows the 'application.properties' file in an IDE. The file contains configuration for MySQL and Hibernate. The MySQL configuration includes the URL, username, password, and driver class name. The Hibernate configuration includes the ddl-auto setting and the show-sql setting.

```
1 # Konfigurasi MySQL Hibernate
2 spring.datasource.url=jdbc:mysql://localhost:3306/rpl_pert5?useSSL=false&serverTimezone=UTC
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
```

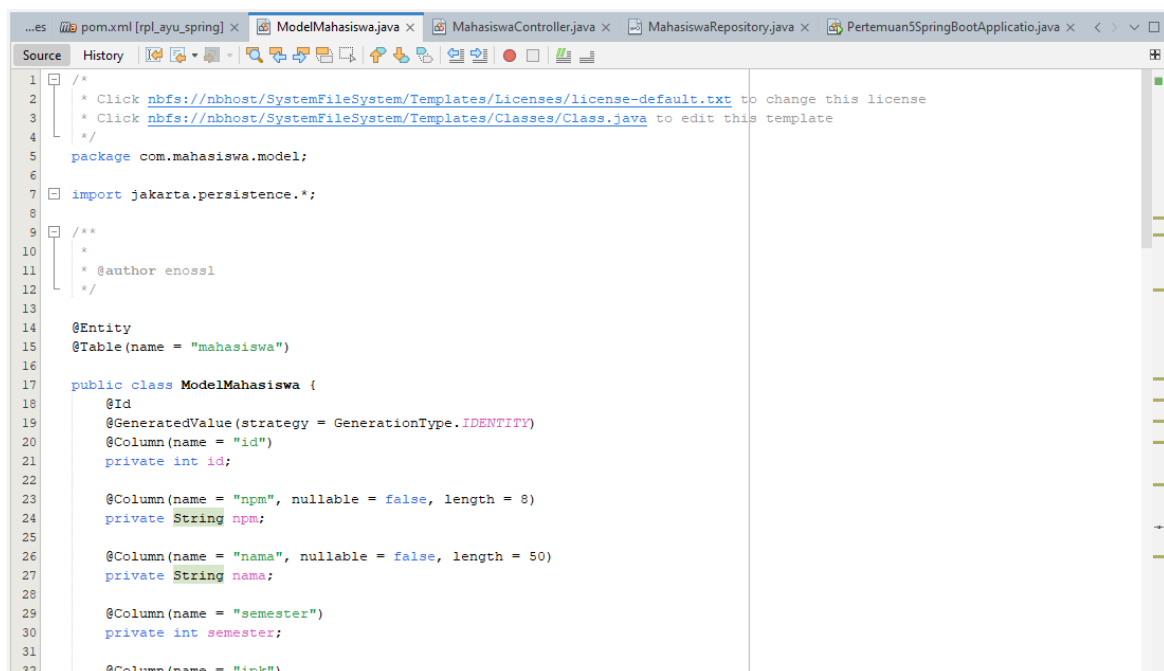
Pertemuan 5 Spring Boot Application



The screenshot shows the 'Pertemuan5SpringBootApplication.java' file in an IDE. The file contains the main class of the application, which implements the CommandLineRunner interface. The class has a main method that runs the Spring application and a run method that calls the tampilMenu method of the 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 package com.mahasiswa;
6 import com.mahasiswa.controller.MahasiswaController;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.boot.CommandLineRunner;
9 import org.springframework.boot.SpringApplication;
10 import org.springframework.boot.autoconfigure.SpringBootApplication;
11
12 /**
13  *
14  * @author enossl
15  */
16 @SpringBootApplication
17 public class Pertemuan5SpringBootApplication implements CommandLineRunner{
18
19     @Autowired
20     private MahasiswaController mhsController;
21     public static void main(String[] args) {
22         SpringApplication.run(Pertemuan5SpringBootApplication.class, args);
23     }
24
25     @Override
26     public void run(String... args) throws Exception {
27         mhsController.tampilMenu();
28     }
29
30 }
```

Model Mahasiswa



The screenshot shows the 'ModelMahasiswa.java' file in an IDE. The file contains the entity class for the Mahasiswa model. The class is annotated with @Entity and @Table. It has three attributes: id, npm, and nama. The id attribute is annotated with @Id and @GeneratedValue. The npm attribute is annotated with @Column. The nama attribute is annotated with @Column.

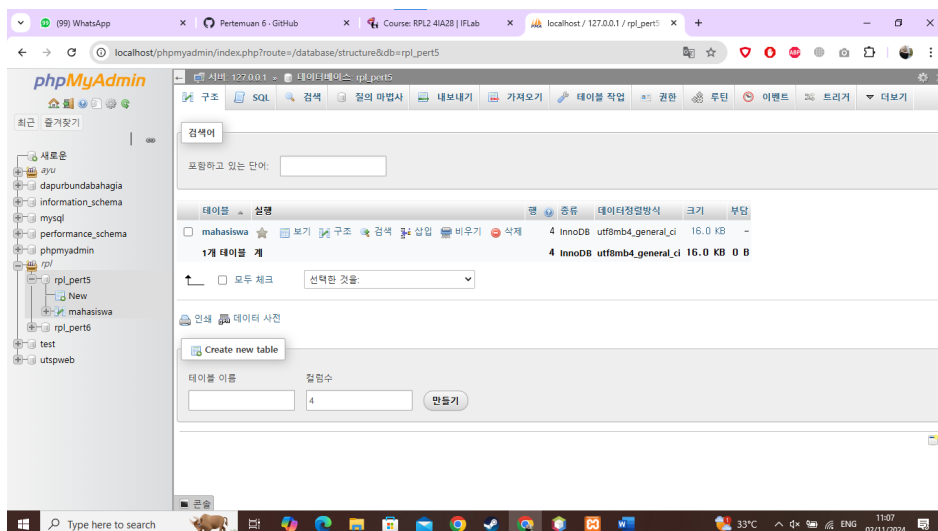
```
1 /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5 package com.mahasiswa.model;
6
7 import jakarta.persistence.*;
8
9 /**
10  *
11  * @author enossl
12  */
13 @Entity
14 @Table(name = "mahasiswa")
15 public class ModelMahasiswa {
16     @Id
17     @GeneratedValue(strategy = GenerationType.IDENTITY)
18     @Column(name = "id")
19     private int id;
20
21     @Column(name = "npm", nullable = false, length = 8)
22     private String npm;
23
24     @Column(name = "nama", nullable = false, length = 50)
25     private String nama;
26
27     @Column(name = "semester")
28     private int semester;
29
30     @Column(name = "ipk")
31
32 }
```

```

33     private Float ipk ;
34
35     public ModelMahasiswa() {
36     }
37
38     public ModelMahasiswa(int id, String npm, String nama, int semester, Float ipk) {
39         this.id = id;
40         this.npm = npm;
41         this.nama = nama;
42         this.semester = semester;
43         this.ipk = ipk;
44     }
45
46     public int getId() {
47         return id;
48     }
49
50     public void setId(int id) {
51         this.id = id;
52     }
53
54     public String getNpm() {
55         return npm;
56     }
57
58     public void setNpm(String npm) {
59         this.npm = npm;
60     }
61
62     public String getNama() {
63         return nama;
64     }
65
66     public void setName(String nama) {
67         this.nama = nama;
68     }
69
70     public int getSemester() {
71         return semester;
72     }
73
74     public void setSemester(int semester) {
75         this.semester = semester;
76     }
77
78     public Float getIpk() {
79         return ipk;
80     }
81
82     public void setIpk(Float ipk) {
83         this.ipk = ipk;
84     }
85
86     @Override
87     public String toString() {
88         return "Mahasiswa{" +
89             "id=" + id +
90             ", npm=" + npm + '\'' +
91             ", nama=" + nama + '\'' +
92             ", semester=" + semester + '\'' +
93             ", ipk=" + ipk + '\'' +
94             '}';
95     }

```

XAMPP



Mahasiswa Controller

```
...es pom.xml [rpl_layu_spring] x ModelMahasiswa.java x MahasiswaController.java x MahasiswaRepository.java x Pertemuan5SpringBootApplication.java x
Source History
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   package com.mahasiswa.controller;
6
7   import com.mahasiswa.model.ModelMahasiswa;
8   import com.mahasiswa.repository.MahasiswaRepository;
9   import org.springframework.beans.factory.annotation.Autowired;
10  import org.springframework.stereotype.Controller;
11
12  import java.util.List;
13  import java.util.Scanner;
14
15  /**
16   *
17   * @author enoss1
18   */
19
20  @Controller
21  public class MahasiswaController {
22
23      @Autowired
24      private MahasiswaRepository mahasiswaRepository;
25
26      public void tampilkanMenu() {
27          Scanner scanner = new Scanner(System.in);
28          int opsi;
29
30          do {
31              System.out.println("\nMenu:");
32              System.out.println("1. Tampilkan semua mahasiswa");
33              System.out.println("2. Tambah mahasiswa baru");
34              System.out.println("3. Cek koneksi database");
35              System.out.println("4. Keluar");
36              System.out.print("Pilih opsi: ");
37              opsi = scanner.nextInt();
38              scanner.nextLine(); // menangkap newline
39
40              switch (opsi) {
41                  case 1:
42                      tampilkanSemuaMahasiswa();
43                      break;
44                  case 2:
45                      tambahMahasiswa(scanner);
46                      break;
47                  case 3:
48                      cekKoneksi();
49                      break;
50                  case 4:
51                      System.out.println("Keluar dari program.");
52                      break;
53                  default:
54                      System.out.println("Ops! tidak valid, coba lagi.");
55              }
56
57          } while (opsi != 4);
58      }
59
60      private void tampilkanSemuaMahasiswa() {
61          List<ModelMahasiswa> mahasiswaList = mahasiswaRepository.findAll();
62          if (mahasiswaList.isEmpty()) {
63              System.out.println("Tidak ada data mahasiswa.");
64          } else {
```

```

65         mahasiswaList.forEach(mahasiswa -> System.out.println(mahasiswa));
66     }
67 }
68
69 private void tambahMahasiswa(Scanner scanner) {
70     System.out.print("Masukkan NPM : ");
71     String npm = scanner.nextLine();
72     System.out.print("Masukkan Nama : ");
73     String nama = scanner.nextLine();
74     System.out.print("Masukkan Semester : ");
75     int semester = scanner.nextInt();
76     System.out.print("Masukkan IPK : ");
77     float ipk = scanner.nextFloat();
78
79     ModelMahasiswa mahasiswa = new ModelMahasiswa(0, npm, nama, semester, ipk);
80     mahasiswaRepository.save(mahasiswa);
81     System.out.println("Mahasiswa berhasil ditambahkan.");
82 }
83
84 private void cekKoneksi() {
85     try {
86         mahasiswaRepository.findAll();
87         System.out.println("Koneksi ke database berhasil.");
88     } catch (Exception e) {
89         System.out.println("Gagal terhubung ke database.");
90     }
91 }
92 }

```

Mahasiswa Repository

```

1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Interface.java to edit this template
4   */
5  package com.mahasiswa.repository;
6
7  import com.mahasiswa.model.ModelMahasiswa;
8  import org.springframework.data.jpa.repository.JpaRepository;
9  import org.springframework.stereotype.Repository;
10
11  @Repository
12  /**
13   *
14   * @author enoss1
15   */
16
17  public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Long> {
18
19  }

```

OUTPUT

Cek Koneksi Database

```

Output - Run (Pertemuan5SpringBootApplication)

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.

```

Tambah Data

```
Output - Run (Pertemuan5SpringBootApplication)

Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih opsi: 2
Masukkan NPM : 50421203
Masukkan Nama : Yuni
Masukkan Semester : 3
Masukkan IPK : 3,9
Hibernate: insert into mahasiswa (ipk,nama,npm,semester) values (?, ?, ?, ?)
Mahasiswa berhasil ditambahkan.
```

Tampilkan Data

```
Output - Run (Pertemuan5SpringBootApplication)

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='50421244', nama='Ayu', semester='7', jurusan='3.83'}
Mahasiswa{id=2, npm='50421212', nama='Jongyun', semester='2', jurusan='3.94'}
Mahasiswa{id=3, npm='50421213', nama='Sori', semester='6', jurusan='3.8'}
Mahasiswa{id=4, npm='50421203', nama='Yuni', semester='3', jurusan='3.9'}
```

Keluar

```
Output - Run (Pertemuan5SpringBootApplication)

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

ACTIVITY PERTEMUAN 6

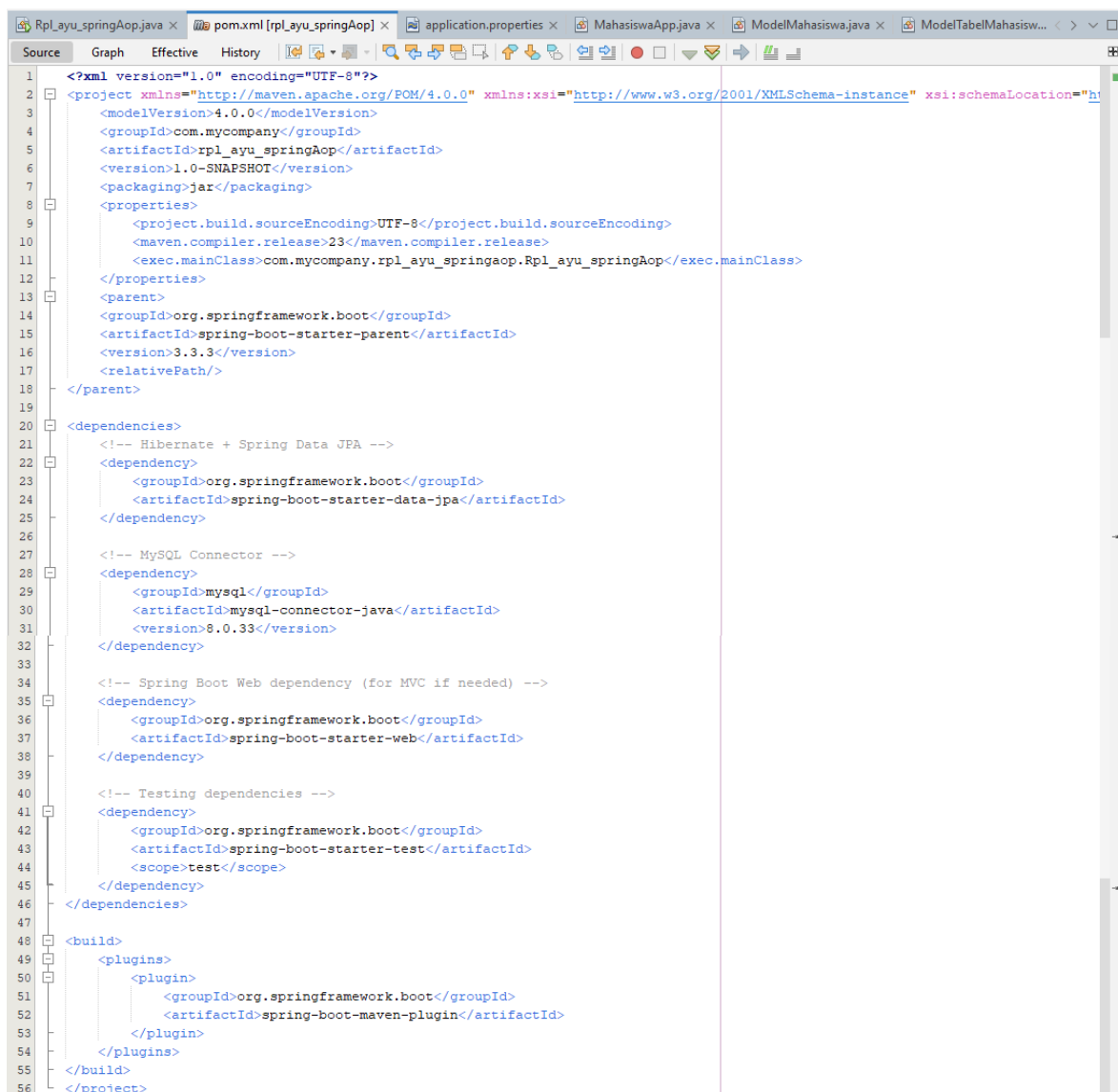
NAMA : AYU RETNONINGRUM SUSILO
NPM : 50421244
KELAS : 4IA28
MATERI : AOP
MATA PRAKTIKUM : REKAYASA PERANGKAT LUNAK 2

Screenshot source code dan output

Jawab:

SOUCE CODE

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/2001/XMLSchema-instance" xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-4.0.0.xsd">
3   <modelVersion>4.0.0</modelVersion>
4   <groupId>com.mycompany</groupId>
5   <artifactId>rpl_ayu_springAop</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.release>23</maven.compiler.release>
11    <exec.mainClass>com.mycompany.rpl_ayu_springaop.Rpl_ayu_springAop</exec.mainClass>
12  </properties>
13  <parent>
14    <groupId>org.springframework.boot</groupId>
15    <artifactId>spring-boot-starter-parent</artifactId>
16    <version>3.3.3</version>
17    <relativePath/>
18  </parent>
19
20  <dependencies>
21    <!-- Hibernate + Spring Data JPA -->
22    <dependency>
23      <groupId>org.springframework.boot</groupId>
24      <artifactId>spring-boot-starter-data-jpa</artifactId>
25    </dependency>
26
27    <!-- MySQL Connector -->
28    <dependency>
29      <groupId>mysql</groupId>
30      <artifactId>mysql-connector-java</artifactId>
31      <version>8.0.33</version>
32    </dependency>
33
34    <!-- Spring Boot Web dependency (for MVC if needed) -->
35    <dependency>
36      <groupId>org.springframework.boot</groupId>
37      <artifactId>spring-boot-starter-web</artifactId>
38    </dependency>
39
40    <!-- Testing dependencies -->
41    <dependency>
42      <groupId>org.springframework.boot</groupId>
43      <artifactId>spring-boot-starter-test</artifactId>
44      <scope>test</scope>
45    </dependency>
46  </dependencies>
47
48  <build>
49    <plugins>
50      <plugin>
51        <groupId>org.springframework.boot</groupId>
52        <artifactId>spring-boot-maven-plugin</artifactId>
53      </plugin>
54    </plugins>
55  </build>
56 </project>
```

Model Mahasiswa

```
Rpl_ayu_springAop.java x pom.xml [rpl_ayu_springAop] x application.properties x MahasiswaApp.java x ModelMahasiswa.java x ModelTabelMahasiswa... < > v □
Source History
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   package com.mahasiswa.model;
6
7   import jakarta.persistence.*;
8
9   /**
10    *
11    * @author enoss1
12    */
13
14   @Entity
15   @Table(name = "mahasiswa")
16
17   public class ModelMahasiswa {
18       @Id
19       @GeneratedValue(strategy = GenerationType.IDENTITY)
20       @Column(name = "id")
21       private int id;
22
23       @Column(name = "npm", nullable = false, length = 8)
24       private String npm;
25
26       @Column(name = "nama", nullable = false, length = 50)
27       private String nama;
28
29       @Column(name = "semester")
30       private int semester;
31
32       @Column(name = "ipk")
33
34       private Float ipk ;
35
36       public ModelMahasiswa() {
37
38       }
39       public ModelMahasiswa(int id,String npm, String nama, int semester, Float ipk){
40           this.id = id;
41           this.npm = npm;
42           this.nama = nama;
43           this.semester = semester;
44           this.ipk = ipk;
45       }
46
47       public int getId() {
48           return id;
49       }
50
51       public void setId(int id) {
52           this.id = id;
53       }
54
55       public String getNpm() {
56           return npm;
57       }
58
59       public void setNpm(String npm) {
60           this.npm = npm;
61       }
62
63       public String getNama() {
64           return nama;
65       }
66
67       public void setNama(String nama) {
68           this.nama = nama;
69       }
70
71       public int getSemester() {
72           return semester;
73       }
74
75       public void setSemester(int semester) {
76           this.semester = semester;
77       }
78
79       public Float getIpk() {
80           return ipk;
81       }
82
83       public void setIpk(Float ipk) {
84           this.ipk = ipk;
85       }
86   }
```


Mahasiswa Controller

```
...va ModelTabelMahasiswa.java x MahasiswaRepository.java x MahasiswaService.java x MahasiswaController.java x MahasiswaView.java x
Source History
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 52 53
/*
 * 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.mahasiswa.controller;

import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.web.bind.annotation.*;
import com.mahasiswa.model.ModelMahasiswa;
import com.mahasiswa.service.MahasiswaService;

import java.util.List;
import org.springframework.stereotype.Controller;

/**
 *
 * @author enoss1
 */
@Controller
public class MahasiswaController {

    @Autowired
    private MahasiswaService mahasiswaService;

    // Add new Mahasiswa
    public String addMahasiswa(@RequestBody ModelMahasiswa mhs) {
        mahasiswaService.addMhs(mhs);
        return "Mahasiswa added successfully";
    }

    // Get Mahasiswa by ID
    public ModelMahasiswa getMahasiswa(@PathVariable int id) {
        return mahasiswaService.getMhs(id);
    }

    // Update Mahasiswa
    public String updateMahasiswa(@RequestBody ModelMahasiswa mhs) {
        mahasiswaService.updateMhs(mhs);
        return "Mahasiswa updated successfully";
    }

    // Delete Mahasiswa by ID
    public String deleteMahasiswa(@PathVariable int id) {
        mahasiswaService.deleteMhs(id);
        return "Mahasiswa deleted successfully";
    }

    // Get all Mahasiswa
    public List<ModelMahasiswa> getAllMahasiswa() {
        return mahasiswaService.getAllMahasiswa();
    }
}
```

Mahasiswa Repository

```
...es MahasiswaApp.java x ModelMahasiswa.java x ModelTabelMahasiswa.java x MahasiswaRepository.java x MahasiswaService.java x
Source History
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Interface.java to edit this template
 */
package com.mahasiswa.repository;

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

/**
 *
 * @author enoss1
 */
@Repository
public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Integer> {
    public Object findById(int id);
    public void deleteById(int id);
}
```

Model Tabel Mahasiswa

```
...va ModelMahasiswa.java x ModelTabelMahasiswa.java x MahasiswaRepository.java x MahasiswaService.java x MahasiswaController.java x
Source History
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   package com.mahasiswa.model;
6   import javax.swing.table.AbstractTableModel;
7   import java.util.List;
8   /**
9    *
10   * @author enoss1
11   */
12   public class ModelTabelMahasiswa extends AbstractTableModel {
13       private List<ModelMahasiswa> mahasiswaList;
14       private String[] columnNames = {"ID", "NPM", "Nama", "Semester", "IPK"};
15
16       public ModelTabelMahasiswa(List<ModelMahasiswa> mahasiswaList) {
17           this.mahasiswaList = mahasiswaList;
18       }
19
20       @Override
21       public int getRowCount() {
22           return mahasiswaList.size(); // Jumlah baris sesuai dengan jumlah data mahasiswa
23       }
24
25       @Override
26       public int getColumnCount() {
27           return columnNames.length; // Jumlah kolom sesuai dengan jumlah elemen dalam columnNames
28       }
29
30       @Override
31       public Object getValueAt(int rowIndex, int columnIndex) {
32           ModelMahasiswa mahasiswa = mahasiswaList.get(rowIndex);
33           switch (columnIndex) {
34               case 0:
35                   return mahasiswa.getId();
36               case 1:
37                   return mahasiswa.getNpm();
38               case 2:
39                   return mahasiswa.getNama();
40               case 3:
41                   return mahasiswa.getSemester();
42               case 4:
43                   return mahasiswa.getIpk();
44               default:
45                   return null;
46           }
47       }
48
49       @Override
50       public String getColumnName(int column) {
51           return columnNames[column]; // Mengatur nama kolom
52       }
53
54       @Override
55       public boolean isCellEditable(int rowIndex, int columnIndex) {
56           return false; // Semua sel tidak dapat diedit
57       }
58
59       // Method untuk menambahkan atau memodifikasi data, jika dibutuhkan
60       public void setMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
61           this.mahasiswaList = mahasiswaList;
62           fireTableDataChanged(); // Memberitahu jTable bahwa data telah berubah
63       }
64   }
```

Application.properties

```
...va pom.xml [rpl_ayu_springAop] x application.properties x MahasiswaApp.java x ModelMahasiswa.java x ModelTabelMahasiswa.java x
Source History
1   # Konfigurasi MySQL Hibernate
2   spring.datasource.url=jdbc:mysql://localhost:3306/rpl_pert6?useSSL=false&serverTimezone=UTC
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
```

Mahasiswa Service

```
...va ModelMahasiswa.java x ModelTabelMahasiswa.java x MahasiswaRepository.java x MahasiswaService.java x MahasiswaController.java x
Source History
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45
/*
 * 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.mahasiswa.service;

import com.mahasiswa.model.ModelMahasiswa;
import com.mahasiswa.repository.MahasiswaRepository;
import jakarta.transaction.Transactional;
import java.util.List;
import org.springframework.beans.factory.annotation.Autowired;
import org.springframework.stereotype.Service;

@Service
/**
 *
 * @author enocsl
 */
public class MahasiswaService {

    @Autowired
    private MahasiswaRepository repository;

    public void addMhs(ModelMahasiswa mhs) {
        repository.save(mhs);
    }

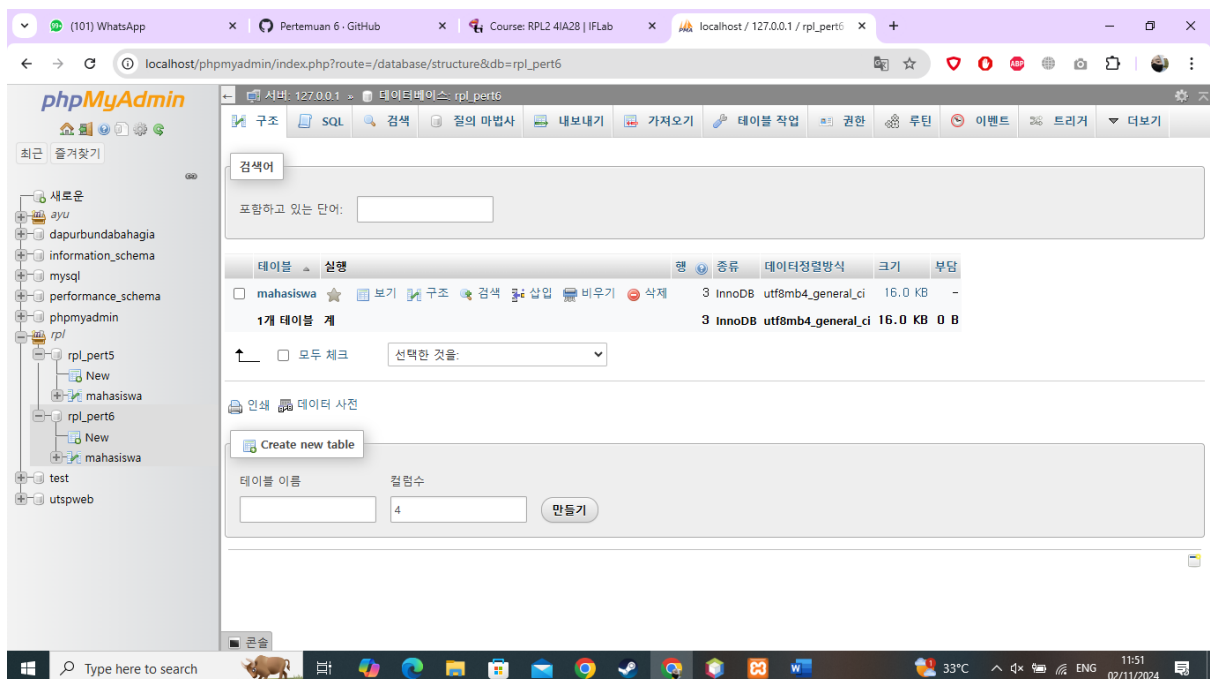
    public ModelMahasiswa getMhs(int id) {
        ModelMahasiswa mahasiswa = (ModelMahasiswa) repository.findById(id);
        return mahasiswa != null ? mahasiswa : null;
    }

    public void updateMhs(ModelMahasiswa mhs) {
        repository.save(mhs);
    }

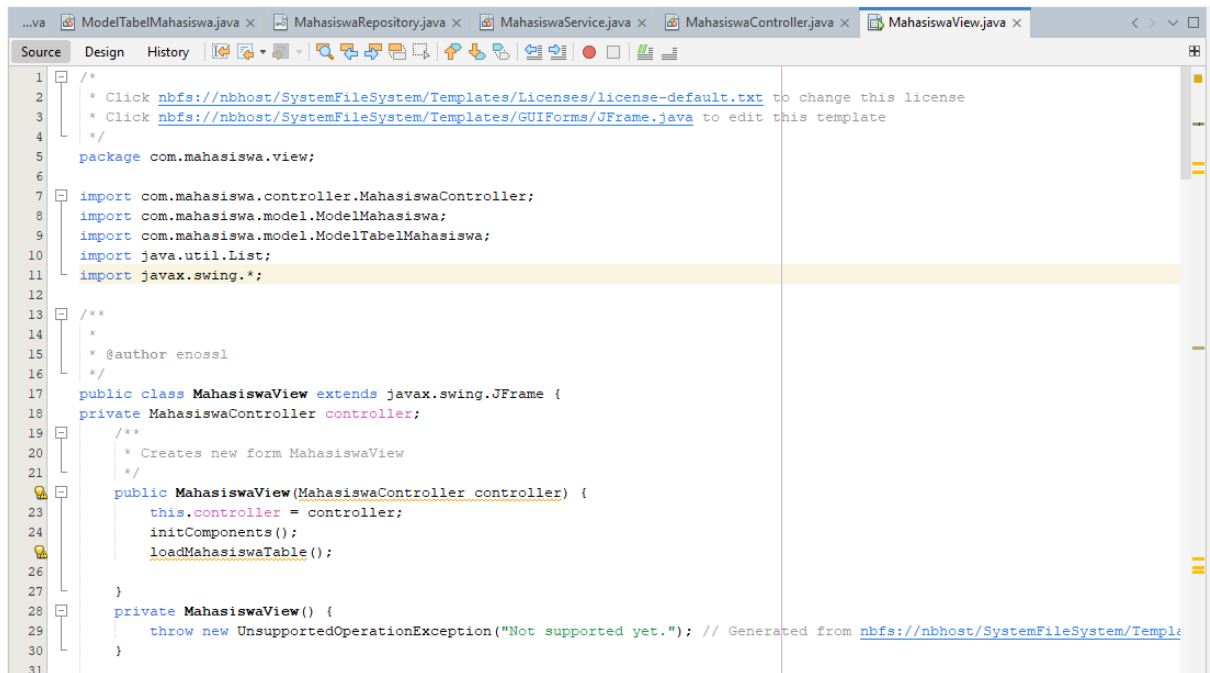
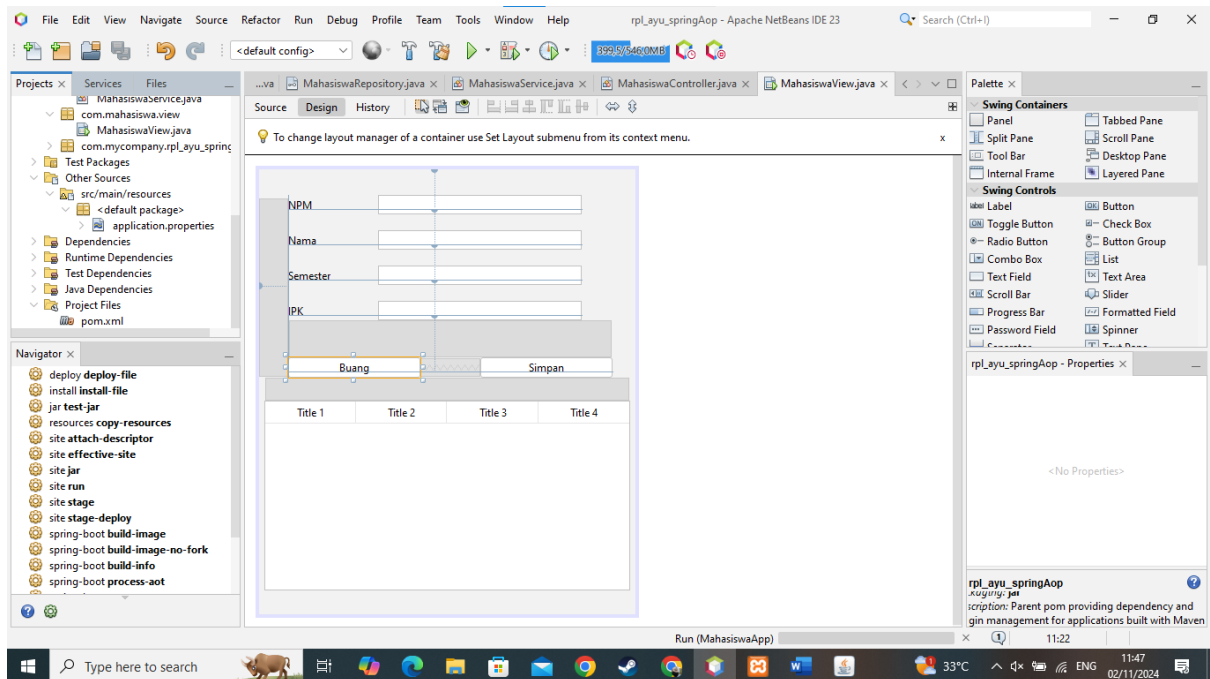
    @Transactional
    public void deleteMhs(int id) {
        repository.deleteById(id);
    }

    public List<ModelMahasiswa> getAllMahasiswa() {
        return repository.findAll();
    }
}
```

XAMPP



Mahasiswa View



```

32 public void loadMahasiswaTable() {
33     // Ambil data dari controller
34     List<ModelMahasiswa> listMahasiswa = controller.getAllMahasiswa();
35
36     // Buat model tabel kustom dengan data mahasiswa
37     ModelTabelMahasiswa tableModel = new ModelTabelMahasiswa(listMahasiswa);
38
39     // Set model pada JTable
40     dataTable.setModel(tableModel);
41 }
42
43 /**
44  * This method is called from within the constructor to initialize the form.
45  * WARNING: Do NOT modify this code. The content of this method is always
46  * regenerated by the Form Editor.
47  */
48 @SuppressWarnings("unchecked")
49 Generated Code
167
168 private void npmFieldActionPerformed(java.awt.event.ActionEvent evt) {
169     // TODO add your handling code here
170 }
171
172 private void simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
173     String npm = getNpmField().getText();
174     String nama = getNamaField().getText();
175     int semester = Integer.parseInt(getSemesterField().getText());
176     float ipk = Float.parseFloat(getIpkField().getText());
177     ModelMahasiswa mahasiswa = new ModelMahasiswa(0, npm, nama, semester, ipk);
178     System.out.println(mahasiswa.getIpk());
179     System.out.println(mahasiswa.getNama());
180     System.out.println(mahasiswa.getSemester());
181     System.out.println(mahasiswa.getNpm());
182
183     controller.addMahasiswa(mahasiswa);
184     loadMahasiswaTable(); // TODO add your handling code here:
185
186 }
187
188 private void buangButtonActionPerformed(java.awt.event.ActionEvent evt) {
189     JTextField idField = new JTextField(5);
190
191     // Membuat panel untuk menampung JTextField
192     JPanel panel = new JPanel();
193     panel.add(new JLabel("Masukkan ID yang ingin dihapus:"));
194     panel.add(idField);
195
196     // Menampilkan dialog box dengan JTextField, tombol OK, dan Cancel
197     int result = JOptionPane.showConfirmDialog(null, panel,
198         "Hapus Mahasiswa", JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
199
200     // Jika tombol OK ditekan
201     if (result == JOptionPane.OK_OPTION) {
202         try {
203             // Mengambil input ID dan memanggil metode deleteMhs
204             int id = Integer.parseInt(idField.getText());
205             controller.deleteMahasiswa(id);
206             JOptionPane.showMessageDialog(null, "Data berhasil dihapus.", "Sukses", JOptionPane.INFORMATION_MESSAGE);
207         } catch (NumberFormatException e) {
208             // Menangani error jika ID yang dimasukkan bukan angka
209             JOptionPane.showMessageDialog(null, "ID harus berupa angka.", "Error", JOptionPane.ERROR_MESSAGE);
210         }
211     }
212     loadMahasiswaTable();
213 }
214
215 public JTextField getIpkField() {
216     return ipkField;
217 }
218
219 public void setIpkField(JTextField ipkField) {
220     this.ipkField = ipkField;
221 }
222
223 public JTextField getNamaField() {
224     return namaField;
225 }
226
227 public void setNamaField(JTextField namaField) {
228     this.namaField = namaField;
229 }
230
231 public JTextField getNpmField() {
232     return npmField;
233 }
234
235 public void setNpmField(JTextField npmField) {
236     this.npmField = npmField;
237 }
238
239 public JTextField getSemesterField() {
240     return semesterField;
241 }

```

```

242
243     public void setSemesterField(JTextField semesterField) {
244         this.semesterField = semesterField;
245     }
246
247     /**
248      * @param args the command line arguments
249      */
250     public static void main(String args[]) {
251         /* Set the Nimbus look and feel */
252         Look and feel setting code (optional)
253
254         /* Create and display the form */
255         java.awt.EventQueue.invokeLater(new Runnable() {
256             public void run() {
257                 new MahasiswaView().setVisible(true);
258             }
259         });
260     }
261
262     // Variables declaration - do not modify
263     private javax.swing.JButton buangButton;
264     private javax.swing.JTable dataTable;
265     private javax.swing.JTextField ipkField;
266     private javax.swing.JLabel jLabel1;
267     private javax.swing.JLabel jLabel2;
268     private javax.swing.JLabel jLabel3;
269     private javax.swing.JLabel jLabel4;
270     private javax.swing.JScrollPane jScrollPane1;
271     private javax.swing.JTextField namaField;
272     private javax.swing.JTextField npmField;
273     private javax.swing.JTextField semesterField;
274     private javax.swing.JButton simpanButton;
275     // End of variables declaration
276 }

```

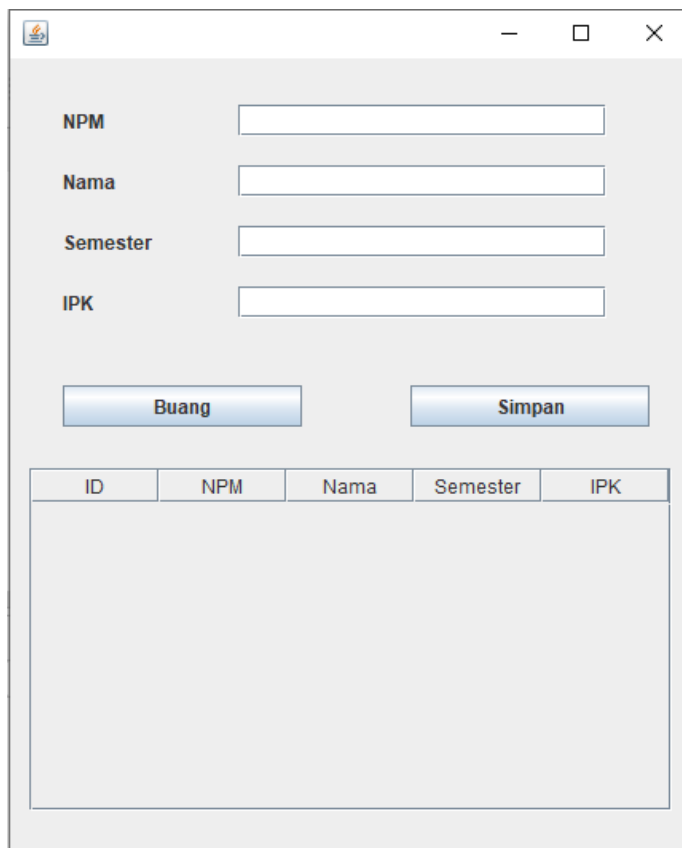
Mahasiswa App

```

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   package com.mahasiswa;
6
7   import com.mahasiswa.controller.MahasiswaController;
8   import com.mahasiswa.service.MahasiswaService;
9   import com.mahasiswa.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  @SpringBootApplication
18
19  /**
20   *
21   * @author enoss1
22   */
23  public class MahasiswaApp implements ApplicationRunner {
24
25      @Autowired
26      private MahasiswaService mahasiswaService;
27
28      public static void main(String[] args) {
29          System.setProperty("java.awt.headless", "false"); // Disable headless mode
30
31          // Start the Spring application and get the application context
32          ApplicationContext context = SpringApplication.run(MahasiswaApp.class, args);
33
34          // Instantiate the view and inject the controller manually
35          MahasiswaController controller = context.getBean(MahasiswaController.class);
36          MahasiswaView mahasiswaView = new MahasiswaView(controller);
37          mahasiswaView.setVisible(true);
38      }
39
40      @Override
41      public void run(ApplicationArguments args) throws Exception {
42          // Implement this method if you need to execute logic after Spring application starts
43          // Otherwise, you can leave it as is.
44      }
45  }

```

OUTPUT



NPM

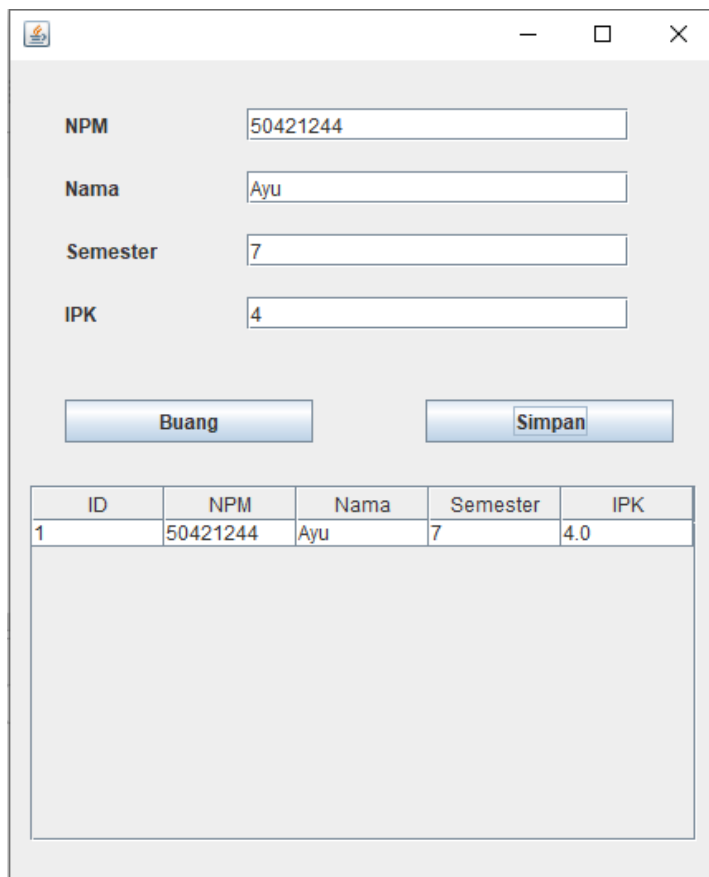
Nama

Semester

IPK

ID	NPM	Nama	Semester	IPK
----	-----	------	----------	-----

Simpan Data



NPM

Nama

Semester

IPK

ID	NPM	Nama	Semester	IPK
1	50421244	Ayu	7	4.0

Formulir input data mahasiswa:

NPM: 50421203

Nama: Yuni

Semester: 3

IPK: 3.91

Buttons: Buang, Simpan

ID	NPM	Nama	Semester	IPK
1	50421244	Ayu	7	4.0
2	50421212	Jongyun	2	3.97
3	50421213	Sori	6	3.88
4	50421203	Yuni	3	3.91

Buang Data

Formulir input data mahasiswa:

NPM: 50421203

Nama: Yuni

Semester: 3

IPK: 3.91

Buttons: Buang, Simpan

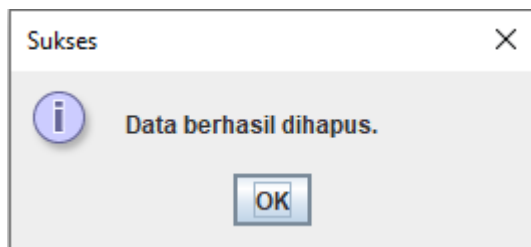
ID	NPM	Nama	Semester	IPK
1	50421244	Ayu	7	4.0
2	50421212	Jongyun	2	3.97
3	50421213	Sori	6	3.88
4	50421203	Yuni	3	3.91

Hapus Mahasiswa

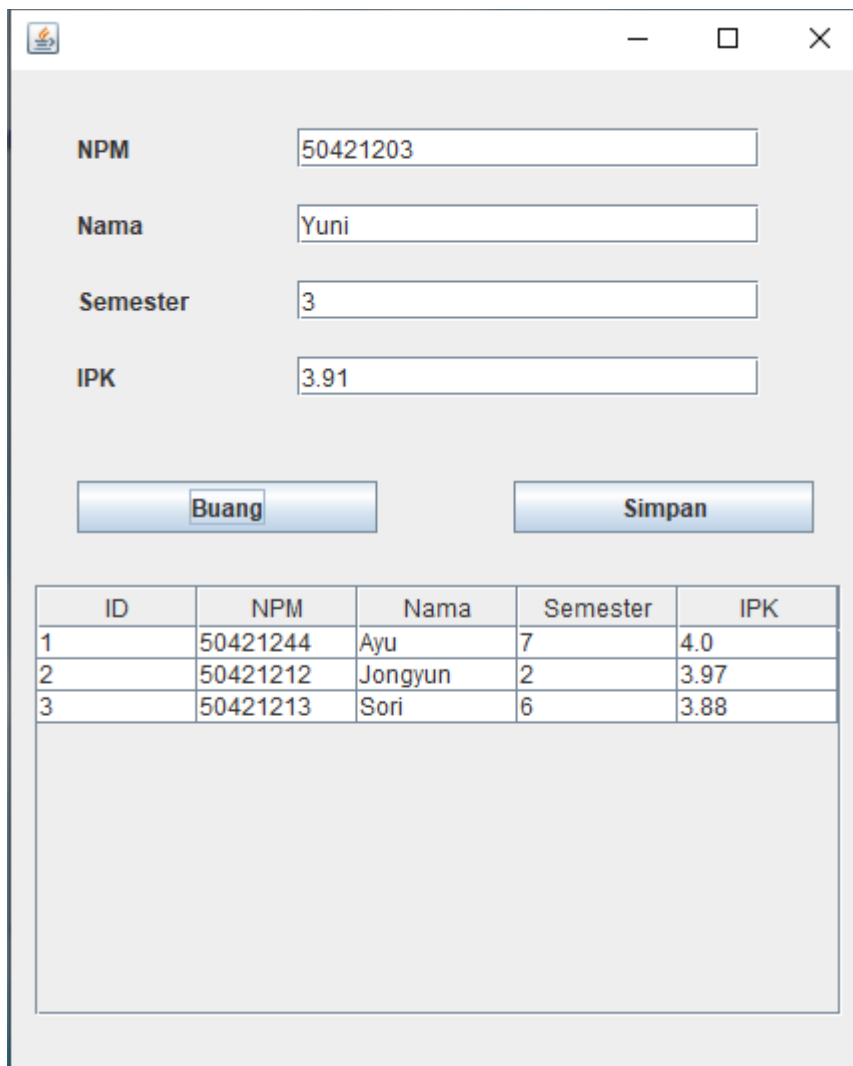
Masukkan ID yang ingin dihapus: 4

Buttons: OK, Cancel

Data Berhasil Dihapus



Tampilan Akhir



A screenshot of the final application window. It features a title bar with a logo and standard window controls. The main area contains four labeled text input fields: "NPM" (50421203), "Nama" (Yuni), "Semester" (3), and "IPK" (3.91). Below these are two buttons: "Buang" (Discard) and "Simpan" (Save). At the bottom is a table with 5 columns: ID, NPM, Nama, Semester, and IPK. The table contains 3 rows of data.

ID	NPM	Nama	Semester	IPK
1	50421244	Ayu	7	4.0
2	50421212	Jongyun	2	3.97
3	50421213	Sori	6	3.88