

ACTIVITY PERTEMUAN 5 & 6

Nama : Maulana Hidayatulloh Mujanah

NPM : 50421797

Kelas : 4IA28

Praktikum RPL 2

Pertemuan 5

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
3 <modelVersion>4.0.0</modelVersion>
4 <groupId>com.mycompany</groupId>
5 <artifactId>RPL_MaulanaHidayatulloh_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.source>21</maven.compiler.source>
11 <maven.compiler.target>21</maven.compiler.target>
12 <exec.mainClass>com.mycompany.rpl_maulanahidayatulloh_spring.RPL_MaulanaHidayatulloh_Spring</exec.mainClass>
13 </properties>
14 <parent>
15 <groupId>org.springframework.boot</groupId>
16 <artifactId>spring-boot-starter-parent</artifactId>
17 <version>3.3.3</version>
18 <relativePath/>
19 </parent>
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 <!-- MySQL Connector -->
27 <dependency>
28 <groupId>mysql</groupId>
29 <artifactId>mysql-connector-java</artifactId>
30 <version>8.0.33</version>
31 </dependency>
32 <!-- Spring Boot Web dependency (for MVC if needed) -->
33 <dependency>
```

```
37 <groupId>org.springframework.boot</groupId>
38 <artifactId>spring-boot-starter-web</artifactId>
39 </dependency>
40
41 <!-- Testing dependencies -->
42 <dependency>
43 <groupId>org.springframework.boot</groupId>
44 <artifactId>spring-boot-starter-test</artifactId>
45 <scope>test</scope>
46 </dependency>
47 </dependencies>
48
49 <build>
50 <plugins>
51 <plugin>
52 <groupId>org.springframework.boot</groupId>
53 <artifactId>spring-boot-maven-plugin</artifactId>
54 </plugin>
55 </plugins>
56 </build>
57 </project>
```

Application.properties

```
1 # Konfigurasi MySQL Hibernate
2 spring.datasource.url=jdbc:mysql://localhost:3306/db_rpl2_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
```

Pertemuan5SpringBootApplication.java

```
1 ...4 lines
5 package com.mahasiswa;
6
7 import com.mahasiswa.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 Maulana
16  */
17
18 @SpringBootApplication
19 public class Pertemuan5SpringBootApplication implements CommandLineRunner{
20
21     @Autowired
22     private MahasiswaController mhsController;
23
24     public static void main(String[] args) {
25         SpringApplication.run(Pertemuan5SpringBootApplication.class, args);
26     }
27
28     @Override
29     public void run(String... args) throws Exception {
30         mhsController.tampilkanMenu();
31     }
32 }
```

ModelMahasiswa.java

```
1  ...4 lines
2  package com.mahasiswa.model;
3
4  import jakarta.persistence.*;
5
6  /**
7   * @author Maulana
8   */
9  @Entity
10 @Table(name = "mahasiswa")
11
12 public class ModelMahasiswa {
13
14     @Id
15     @GeneratedValue(strategy = GenerationType.IDENTITY)
16     @Column(name = "id")
17     private int id;
18
19     @Column(name = "npm", nullable = false, length = 8)
20     private String npm;
21
22     @Column(name = "nama", nullable = false, length = 50)
23     private String nama;
24
25     @Column(name = "semester")
26     private int semester;
27
28     @Column(name = "ipk")
29     private float ipk;
30
31     public ModelMahasiswa() {
32
33     }
34
35     public ModelMahasiswa(int id, String npm, String nama, int semester, float ipk) {
```

```
36         this.id = id;
37         this.npm = npm;
38         this.nama = nama;
39         this.semester = semester;
40         this.ipk = ipk;
41     }
42
43     public int getId() {
44         return id;
45     }
46
47     public void setId(int id) {
48         this.id = id;
49     }
50
51     public String getNpm() {
52         return npm;
53     }
54
55     public void setNpm(String npm) {
56         this.npm = npm;
57     }
58
59     public String getNama() {
60         return nama;
61     }
62
63     public void setNama(String nama) {
64         this.nama = nama;
65     }
66
67     public int getSemester() {
68         return semester;
69     }
70
71     public void setSemester(int semester) {
72         this.semester = semester;
73     }
74 }
```

```

78     }
79
80     public float getIpk() {
81         return ipk;
82     }
83
84     public void setIpk(float ipk) {
85         this.ipk = ipk;
86     }
87
88     @Override
89     public String toString() {
90         return "Mahasiswa{" +
91             "id =" + id +
92             ", npm =" + npm + '\'' +
93             ", nama =" + nama + '\'' +
94             ", semester =" + semester + '\'' +
95             ", ipk =" + ipk + '\'' +
96             '}';
97     }
98 }
99

```

MahasiswaController

```

1  ...4 lines
2
3  package com.mahasiswa.controller;
4
5  import com.mahasiswa.model.ModelMahasiswa;
6  import com.mahasiswa.repository.MahasiswaRepository;
7  import org.springframework.beans.factory.annotation.Autowired;
8  import org.springframework.stereotype.Controller;
9
10 import java.util.List;
11 import java.util.Scanner;
12
13 @Controller
14
15 /**
16  *
17  * @author Maulana
18  */
19 public class MahasiswaController {
20
21     @Autowired
22     private MahasiswaRepository mahasiswaRepository;
23
24     public void tampilkanMenu() {
25         Scanner scanner = new Scanner(System.in);
26         int opsi;
27
28         do {
29             System.out.println(x: "\nMenu:");
30             System.out.println(x: "1. Tampilkan semua mahasiswa");
31             System.out.println(x: "2. Tambah mahasiswa baru");
32             System.out.println(x: "3. Cek koneksi database");
33             System.out.println(x: "4. Keluar");
34             System.out.print(s: "Pilih opsi: ");
35             opsi = scanner.nextInt();
36             scanner.nextLine(); // menangkap newline
37
38             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(x: "Keluar dari program.");
52             break;
53         default:
54             System.out.println(x: "Opsi 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(x: "Tidak ada data mahasiswa.");
64     } else {
65         mahasiswaList.forEach(mahasiswa -> System.out.println(x: mahasiswa));
66     }
67 }
68
69 private void tambahMahasiswa(Scanner scanner) {
70     System.out.print(s: "Masukkan NPM : ");
71     String npm = scanner.nextLine();
72     System.out.print(s: "Masukkan Nama : ");
73     String nama = scanner.nextLine();
74     System.out.print(s: "Masukkan Semester : ");
75     int semester = scanner.nextInt();
76     System.out.print(s: "Masukkan IPK : ");
77     float ipk = scanner.nextFloat();

```

```

78     ModelMahasiswa mahasiswa = new ModelMahasiswa(id: 0, npm, nama, semester, ipk);
79     mahasiswaRepository.save(entity:mahasiswa);
80     System.out.println(x: "Mahasiswa berhasil ditambahkan.");
81 }
82
83
84 private void cekKoneksi() {
85     try {
86         mahasiswaRepository.findAll();
87         System.out.println(x: "Koneksi ke database berhasil.");
88     } catch (Exception e) {
89         System.out.println(x: "Gagal terhubung ke database.");
90     }
91 }
92
93

```

MahasiswaRepository.java

```

1  package com.mahasiswa.repository;
2
3  import com.mahasiswa.model.ModelMahasiswa;
4  import org.springframework.data.jpa.repository.JpaRepository;
5  import org.springframework.stereotype.Repository;
6
7  @Repository
8  public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Long> {
9
10 }

```

Output :

Cek Koneksi Database

```
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.
```

Tambah Data Mahasiswa

```
Menu:
1. Tampilkan semua mahasiswa
2. Tambah mahasiswa baru
3. Cek koneksi database
4. Keluar
Pilih opsi: 2
Masukkan NPM : 50421797
Masukkan Nama : Maulana
Masukkan Semester : 7
Masukkan IPK : 4.0
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: 2
Masukkan NPM : 12312312
Masukkan Nama : Panjul
Masukkan Semester : 15
Masukkan IPK : 2.5
Hibernate: insert into mahasiswa (ipk,nama,npm,semester) values (?, ?, ?, ?)
Mahasiswa berhasil ditambahkan.
```

Tampilkan Data Mahasiswa

```
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, npm ='50421797', nama ='Maulana', semester ='7', ipk ='4.0'}
Mahasiswa{id =2, npm ='12312312', nama ='Panjul', semester ='15', ipk ='2.5'}
```

Pertemuan 6

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

application.properties

```
1 # Konfigurasi MySQL Hibernate
2 spring.datasource.url=jdbc:mysql://localhost:3306/db_rpl2_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
```

MahasiswaApp.java

```
1 ...4 lines
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 /**
18  *
19  * @author Maulana
20  */
21
22 @SpringBootApplication
23 public class MahasiswaApp implements ApplicationRunner {
24
25     @Autowired
26     private MahasiswaService mahasiswaService;
27
28     public static void main(String[] args) {
29         System.setProperty(key: "java.awt.headless", value: "false"); // Disable headless mode
30
31         // Start the Spring application and get the application context
32         ApplicationContext context = SpringApplication.run(primarySource: MahasiswaApp.class, args);
33
34         // Instantiate the view and inject the controller manually
35         MahasiswaController controller = context.getBean(requiredType: MahasiswaController.class);
36         MahasiswaView mahasiswaView = new MahasiswaView(controller);
37         mahasiswaView.setVisible(b: 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 }
```


ModelMahasiswa.java

```
1  ...4 lines
2
3  package com.mahasiswa.model;
4
5  import jakarta.persistence.*;
6
7  /**
8   *
9   * @author Maulana
10  */
11
12  @Entity
13  @Table(name = "mahasiswa")
14
15  public class ModelMahasiswa {
16
17      @Id
18      @GeneratedValue(strategy = GenerationType.IDENTITY)
19      @Column(name = "id")
20      private int id;
21
22      @Column(name = "npm", nullable = false, length = 8)
23      private String npm;
24
25      @Column(name = "nama", nullable = false, length = 50)
26      private String nama;
27
28      @Column(name = "semester")
29      private int semester;
30
31      @Column(name = "ipk")
32      private float ipk;
33
34      public ModelMahasiswa() {
35
36      }
37
38      public ModelMahasiswa(int id, String npm, String nama, int semester, float ipk) {
```

```
39
40      this.id = id;
41      this.npm = npm;
42      this.nama = nama;
43      this.semester = semester;
44      this.ipk = ipk;
45
46  }
47
48  public int getId() {
49      return id;
50  }
51
52  public void setId(int id) {
53      this.id = id;
54  }
55
56  public String getNpm() {
57      return npm;
58  }
59
60  public void setNpm(String npm) {
61      this.npm = npm;
62  }
63
64  public String getNama() {
65      return nama;
66  }
67
68  public void setNama(String nama) {
69      this.nama = nama;
70  }
71
72  public int getSemester() {
73      return semester;
74  }
75
76  public void setSemester(int semester) {
77      this.semester = semester;
```

```

78     }
79
80     public float getIpk() {
81         return ipk;
82     }
83
84     public void setIpk(float ipk) {
85         this.ipk = ipk;
86     }
87 }
88

```

ModelTabelMahasiswa.java

```

1  ...4 lines
5  package com.mahasiswa.model;
6
7  import javax.swing.table.AbstractTableModel;
8  import java.util.List;
9
10 /**
11  *
12  * @author Maulana
13  */
14
15
16
17 public class ModelTabelMahasiswa extends AbstractTableModel{
18     private List<ModelMahasiswa> mahasiswaList;
19     private String[] columnNames = {"ID", "NPM", "Nama", "Semester", "IPK"};
20
21     public ModelTabelMahasiswa(List<ModelMahasiswa> mahasiswaList) {
22         this.mahasiswaList = mahasiswaList;
23     }
24
25     @Override
26     public int getRowCount() {
27         return mahasiswaList.size(); // Jumlah baris sesuai dengan jumlah data mahasiswa
28     }
29
30     @Override
31     public int getColumnCount() {
32         return columnNames.length; // Jumlah kolom sesuai dengan jumlah elemen dalam columnNames
33     }
34
35     @Override
36     public Object getValueAt(int rowIndex, int columnIndex) {
37         ModelMahasiswa mahasiswa = mahasiswaList.get(rowIndex);
38         switch (columnIndex) {
39             case 0:
40                 return mahasiswa.getId();

```

```

41         case 1:
42             return mahasiswa.getNpm();
43         case 2:
44             return mahasiswa.getNama();
45         case 3:
46             return mahasiswa.getSemester();
47         case 4:
48             return mahasiswa.getIpk();
49         default:
50             return null;
51     }
52 }
53
54 @Override
55 public String getColumnName(int column) {
56     return columnNames[column]; // Mengatur nama kolom
57 }
58
59 @Override
60 public boolean isCellEditable(int rowIndex, int columnIndex) {
61     return false; // Semua sel tidak dapat diedit
62 }
63
64 // Method untuk menambahkan atau memodifikasi data, jika dibutuhkan
65 public void setMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
66     this.mahasiswaList = mahasiswaList;
67     fireTableDataChanged(); // Memberitahu JTable bahwa data telah berubah
68 }
69 }
70

```

MahasiswaRepository.java

```

1  package com.mahasiswa.repository;
2
3  import com.mahasiswa.model.ModelMahasiswa;
4  import org.springframework.data.jpa.repository.JpaRepository;
5  import org.springframework.stereotype.Repository;
6
7  @Repository
8  public interface MahasiswaRepository extends JpaRepository<ModelMahasiswa, Integer> {
9      public Object findById(int id);
10
11      public void deleteById(int id);
12
13  }

```

MahasiswaService.java

```
1 package com.mahasiswa.service;
2
3 import com.mahasiswa.model.ModelMahasiswa;
4 import com.mahasiswa.repository.MahasiswaRepository;
5 import jakarta.transaction.Transactional;
6 import java.util.List;
7 import org.springframework.beans.factory.annotation.Autowired;
8 import org.springframework.stereotype.Service;
9
10 @Service
11 public class MahasiswaService {
12
13     @Autowired
14     private MahasiswaRepository repository;
15
16     public void addMhs(ModelMahasiswa mhs) {
17         repository.save(entity:mhs);
18     }
19
20     public ModelMahasiswa getMhs(int id) {
21         ModelMahasiswa mahasiswa = (ModelMahasiswa) repository.findById(id);
22         return mahasiswa != null ? mahasiswa : null;
23     }
24
25     public void updateMhs(ModelMahasiswa mhs) {
26         repository.save(entity:mhs);
27     }
28
29     @Transactional
30     public void deleteMhs(int id) {
31         repository.deleteById(id);
32     }
33
34     public List<ModelMahasiswa> getAllMahasiswa() {
35         return repository.findAll();
36     }
37 }
```

MahasiswaController.java

```
1 package com.mahasiswa.controller;
2
3 import org.springframework.beans.factory.annotation.Autowired;
4 import org.springframework.web.bind.annotation.*;
5 import com.mahasiswa.model.ModelMahasiswa;
6 import com.mahasiswa.service.MahasiswaService;
7
8 import java.util.List;
9 import org.springframework.stereotype.Controller;
10
11 @Controller
12 public class MahasiswaController {
13
14     @Autowired
15     private MahasiswaService mahasiswaService;
16
17     // Add new Mahasiswa
18     public String addMahasiswa(@RequestBody ModelMahasiswa mhs) {
19         mahasiswaService.addMhs(mhs);
20         return "Mahasiswa added successfully";
21     }
22
23     // Get Mahasiswa by ID
24     public ModelMahasiswa getMahasiswa(@PathVariable int id) {
25         return mahasiswaService.getMhs(id);
26     }
27
28     // Update Mahasiswa
29     public String updateMahasiswa(@RequestBody ModelMahasiswa mhs) {
30         mahasiswaService.updateMhs(mhs);
31         return "Mahasiswa updated successfully";
32     }
33
34     // Delete Mahasiswa by ID
35     public String deleteMahasiswa(@PathVariable int id) {
36         mahasiswaService.deleteMhs(id);
37     }
38 }
```

```

38         return "Mahasiswa deleted successfully";
39     }
40
41     // Get all Mahasiswa
42     public List<ModelMahasiswa> getAllMahasiswa() {
43         return mahasiswaService.getAllMahasiswa();
44     }
45 }

```

MahasiswaView.java (Design)

NPM

Nama

Semester

IPK

Simpan

Buang

Title 1	Title 2	Title 3	Title 4

MahasiswaView.java (Code)

```

1  ...4 lines
2  package com.mahasiswa.view;
3
4
5
6
7  import com.mahasiswa.controller.MahasiswaController;
8  import com.mahasiswa.model.ModelMahasiswa;
9  import com.mahasiswa.model.ModelTabelMahasiswa;
10 import java.util.List;
11 import javax.swing.JLabel;
12 import javax.swing.JOptionPane;
13 import javax.swing.JPanel;
14 import javax.swing.JTextField;
15
16 /**
17  *
18  * @author Maulana
19  */
20 public class MahasiswaView extends javax.swing.JFrame {
21     private MahasiswaController controller;
22
23     /**
24      * Creates new form MahasiswaView
25      */
26     public MahasiswaView(MahasiswaController controller) {
27         this.controller = controller;
28         initComponents();
29         loadMahasiswaTable();
30     }
31
32     private MahasiswaView() {
33         throw new UnsupportedOperationException("Not supported yet."); // Generated from nbfs://nbhost/SystemFileSystem/Templates/Classes/Code/Ge
34     }
35
36     public void loadMahasiswaTable() {
37         // Ambil data dari controller
38         List<ModelMahasiswa> listMahasiswa = controller.getAllMahasiswa();
39

```

```

40 // Buat model tabel kustom dengan data mahasiswa
41 ModelTabelMahasiswa tableModel = new ModelTabelMahasiswa(mahasiswaList:listMahasiswa);
42
43 // Set model pada jTable
44 dataTable.setModel(dataModel:tableModel);
45 }
46
47 /**
48  * This method is called from within the constructor to initialize the form.
49  * WARNING: Do NOT modify this code. The content of this method is always
50  * regenerated by the Form Editor.
51  */
52 @SuppressWarnings("unchecked")
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74 private void simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
75     String npm = getNpmField().getText();
76     String nama = getNamaField().getText();
77     int semester = Integer.parseInt(s: getSemesterField().getText());
78     float ipk = Float.parseFloat(s: getIpkField().getText());
79     ModelMahasiswa mahasiswa = new ModelMahasiswa(id: 0, npm, nama, semester, ipk);
80     System.out.println(x: mahasiswa.getIpk());
81     System.out.println(x: mahasiswa.getNama());
82     System.out.println(x: mahasiswa.getSemester());
83     System.out.println(x: mahasiswa.getNpm());
84
85     controller.addMahasiswa(mhs:mahasiswa);
86     loadMahasiswaTable();
87 }
88
89 private void buangButtonActionPerformed(java.awt.event.ActionEvent evt) {
90     JTextField idField = new JTextField(columns: 5);
91
92     // Membuat panel untuk menampung JTextField
93     JPanel panel = new JPanel();
94     panel.add(new JLabel(text: "Masukkan ID yang ingin dihapus:"));

```

```

195     panel.add(comp: idField);
196
197     // Menampilkan dialog box dengan JTextField, tombol OK, dan Cancel
198     int result = JOptionPane.showConfirmDialog(parentComponent: null, message: panel,
199         title: "Hapus Mahasiswa", optionType: JOptionPane.OK_CANCEL_OPTION, messageType: JOptionPane.PLAIN_MESSAGE);
200
201     // Jika tombol OK ditekan
202     if (result == JOptionPane.OK_OPTION) {
203         try {
204             // Mengambil input ID dan memanggil metode deleteMhs
205             int id = Integer.parseInt(s: idField.getText());
206             controller.deleteMahasiswa(id);
207             JOptionPane.showMessageDialog(parentComponent: null, message: "Data berhasil dihapus.", title: "Sukses", messageType: JOptionPane.INFORMATION_MESSAGE);
208         } catch (NumberFormatException e) {
209             // Menangani error jika ID yang dimasukkan bukan angka
210             JOptionPane.showMessageDialog(parentComponent: null, message: "ID harus berupa angka.", title: "Error", messageType: JOptionPane.ERROR_MESSAGE);
211         }
212     }
213     loadMahasiswaTable();
214 }
215
216 public JTextField getIpkField() {
217     return ipkField;
218 }
219
220 public void setIpkField(JTextField ipkField) {
221     this.ipkField = ipkField;
222 }
223
224 public JTextField getNamaField() {
225     return namaField;
226 }
227
228 public void setNamaField(JTextField namaField) {
229     this.namaField = namaField;
230 }

```

```

231
232 public JTextField getNpmField() {
233     return npmField;
234 }
235
236 public void setNpmField(JTextField npmField) {
237     this.npmField = npmField;
238 }
239
240 public JTextField getSemesterField() {
241     return semesterField;
242 }
243
244 public void setSemesterField(JTextField semesterField) {
245     this.semesterField = semesterField;
246 }
247
248 /**
249  * @param args the command line arguments
250  */
251 public static void main(String args[]) {
252     /* Set the Nimbus look and feel */
253     Look and feel setting code (optional)
254
255     /* Create and display the form */
256     java.awt.EventQueue.invokeLater(new Runnable() {
257         public void run() {
258             new MahasiswaView().setVisible(true);
259         }
260     });
261 }
262
263 // Variables declaration - do not modify
264 private javax.swing.JButton buangButton;
265 private javax.swing.JTable dataTable;
266 private javax.swing.JTextField ipkField;

```

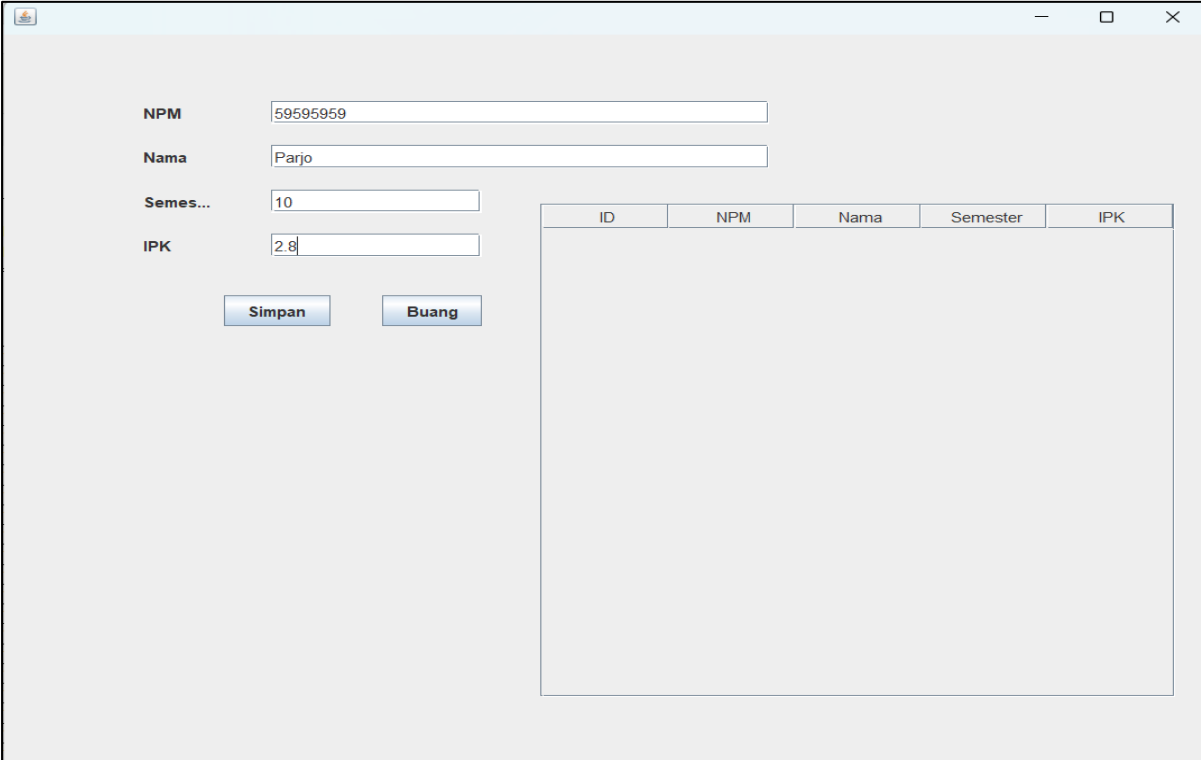
```

267 private javax.swing.JLabel jLabel1;
268 private javax.swing.JLabel jLabel2;
269 private javax.swing.JLabel jLabel3;
270 private javax.swing.JLabel jLabel4;
271 private javax.swing.JScrollPane jScrollPane1;
272 private javax.swing.JTextField namaField;
273 private javax.swing.JTextField npmField;
274 private javax.swing.JTextField semesterField;
275 private javax.swing.JButton simpanButton;
276 // End of variables declaration
277 }
278

```

Output :

Tambah 5 Data Mahasiswa



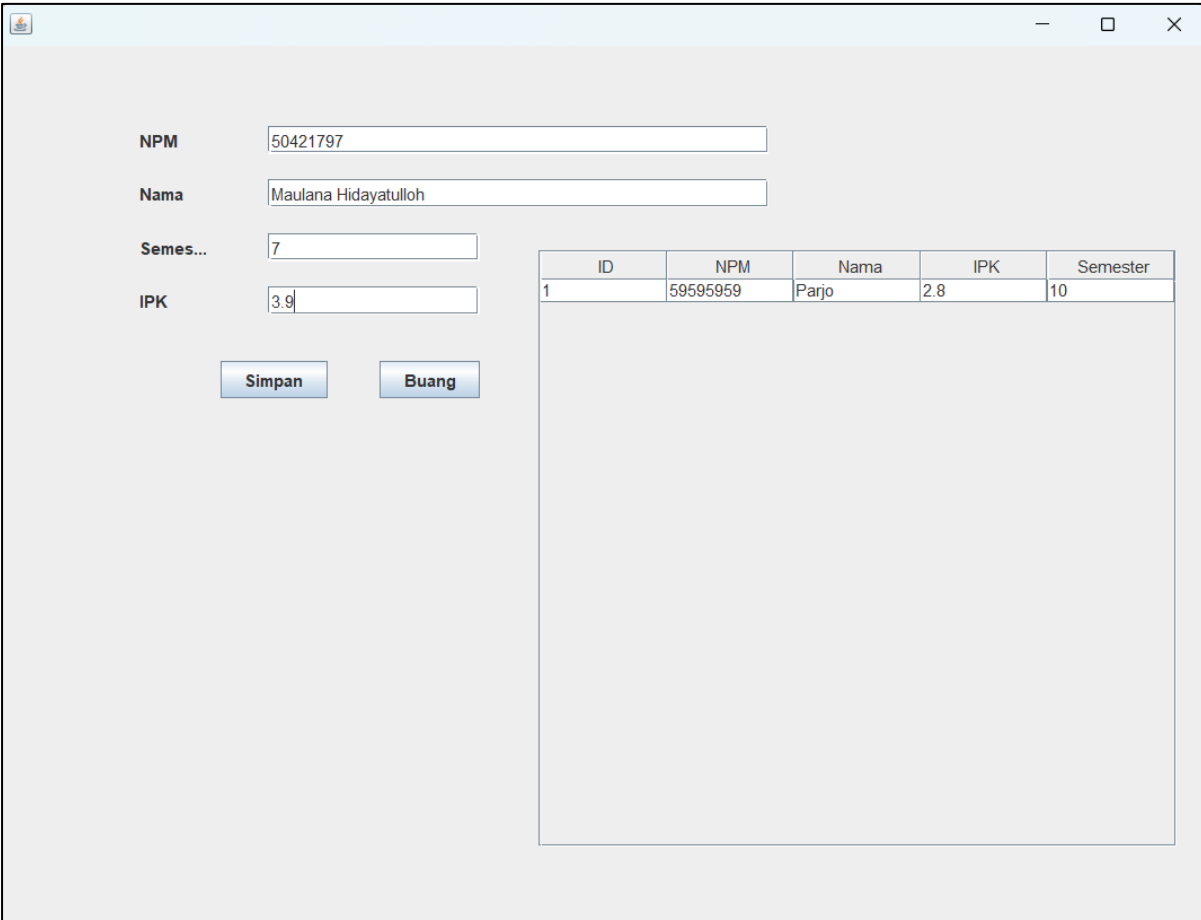
NPM: 59595959

Nama: Parjo

Semes...: 10

IPK: 2.8

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



NPM: 50421797

Nama: Maulana Hidayatulloh

Semes...: 7

IPK: 3.9

ID	NPM	Nama	IPK	Semester
1	59595959	Parjo	2.8	10

NPM

12312312

Nama

Panjul

Semes...

14

IPK

2.0

Simpan

Buang

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9

NPM

12345678

Nama

Mulyono

Semes...

3

IPK

3.0

Simpan

Buang

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9
3	12312312	Panjul	14	2.0

NPM

89076555

Nama

Naruto

Semes...

8

IPK

3.5

Simpan

Buang

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9
3	12312312	Panjul	14	2.0
4	12345678	Mulyono	3	3.0

NPM

Nama

Semes...

IPK

Simpan

Buang

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9
3	12312312	Panjul	14	2.0
4	12345678	Mulyono	3	3.0
5	89076555	Naruto	8	3.5

Buang Salah Satu Data Mahasiswa

The screenshot shows a web application interface for managing student data. On the left, there are input fields for NPM, Nama, Semes..., and IPK, along with 'Simpan' and 'Buang' buttons. On the right, a table lists student data. A modal dialog titled 'Hapus Mahasiswa' is open, prompting the user to enter the ID of the student to be deleted. The input field contains the value '4'.

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9
3	12312312	Panjul	14	2.0
4	12345678	Mulyono	3	3.0
5	66676666	M...	8	3.5

Hapus Mahasiswa

Masukkan ID yang ingin dihapus: 4

OK Cancel

This screenshot shows the same application after the deletion of the student with ID 4. A success message dialog titled 'Sukses' is displayed, indicating that the data was successfully deleted. The table on the right now only contains five rows of data, with ID 4 removed.

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9
3	12312312	Panjul	14	2.0
4	12345678	Mulyono	3	3.0
5	66676666	M...	8	3.5

Sukses

Data berhasil dihapus.

OK

NPM

Nama

Semes...

IPK

Simpan

Buang

ID	NPM	Nama	Semester	IPK
1	59595959	Parjo	10	2.8
2	50421797	Maulana Hiday...	7	3.9
3	12312312	Panjul	14	2.0
5	89076555	Naruto	8	3.5