

Nama: Shaun Patrick Hendra

NPM: 51421416

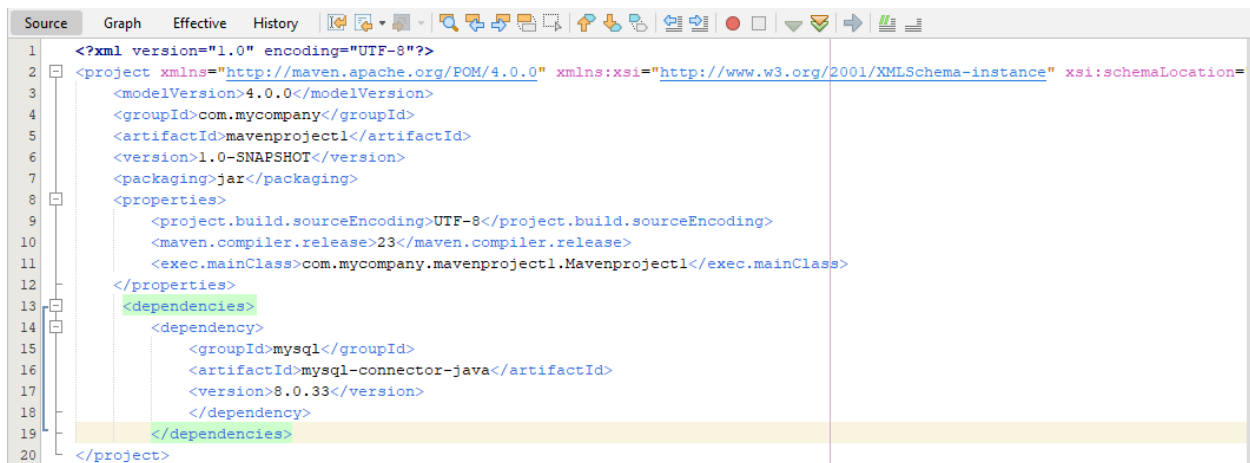
Kelas: 4IA28

Praktikum RPL 2

Pertemuan 3

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/maven-v4_0_0.xsd">
3    <modelVersion>4.0.0</modelVersion>
4    <groupId>com.mycompany</groupId>
5    <artifactId>mavenproject1</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.mavenproject1.Mavenproject1</exec.mainClass>
12   </properties>
13   <dependencies>
14     <dependency>
15       <groupId>mysql</groupId>
16       <artifactId>mysql-connector-java</artifactId>
17       <version>8.0.33</version>
18     </dependency>
19   </dependencies>
20 </project>
```

ModelMahasiswa

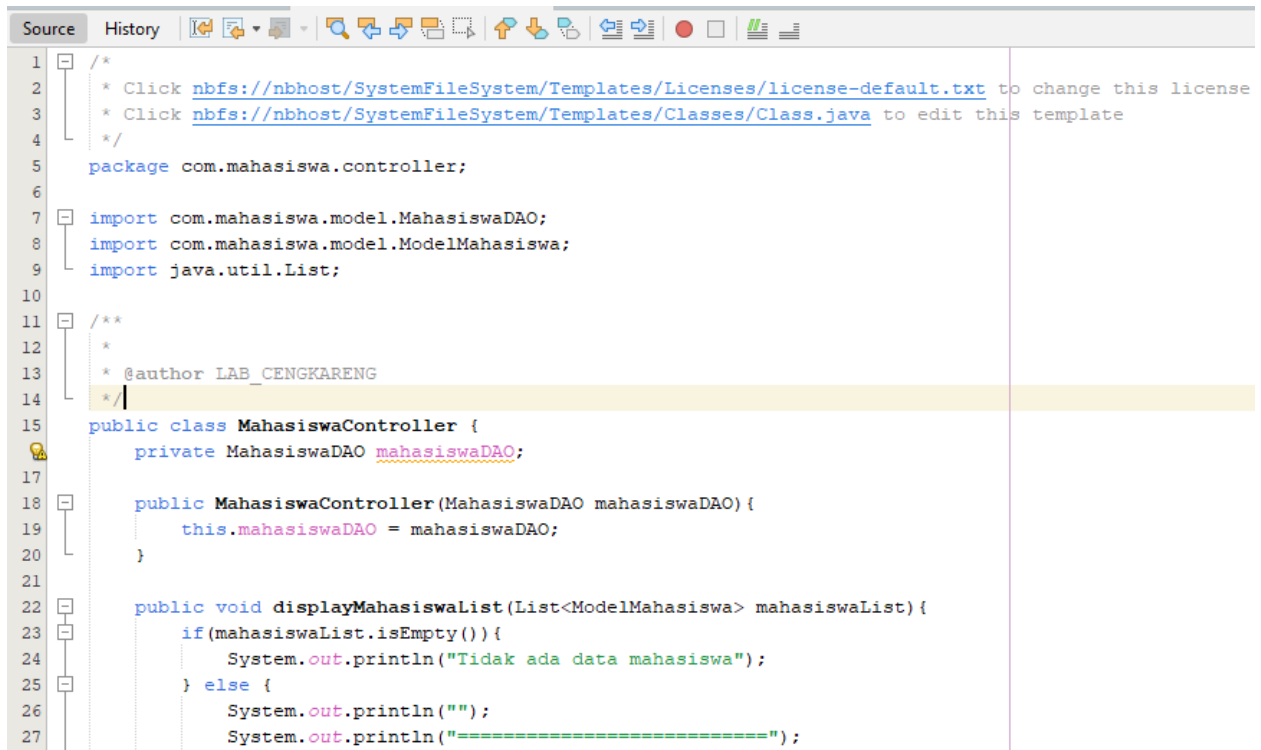
```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4   */
5   package com.mahasiswa.model;
6
7   /**
8    *
9    * @author LAB_CENGKARENG
10   */
11  public class ModelMahasiswa {
12      private int id;
13      private String npm;
14      private String nama;
15      private int semester;
16      private float ipk;
17
18      public ModelMahasiswa(int id, String npm, String nama, int semester, float ipk) {
19          this.id = id;
20          this.npm = npm;
21          this.nama = nama;
22          this.semester = semester;
23          this.ipk = ipk;
24      }
25
26
27      public int getId() {
```

```

28 |         return id;
29 |     }
30 |
31 | ☐ public void setId(int id) {
32 |     |         this.id = id;
33 |     |     }
34 |
35 | ☐ public String getNpm() {
36 |     |         return npm;
37 |     |     }
38 |
39 | ☐ public void setNpm(String npm) {
40 |     |         this.npm = npm;
41 |     |     }
42 |
43 | ☐ public String getNama() {
44 |     |         return nama;
45 |     |     }
46 |
47 | ☐ public void setNama(String nama) {
48 |     |         this.nama = nama;
49 |     |     }
50 |
51 | ☐ public int getSemester() {
52 |     |         return semester;
53 |     |     }
54 |
55 | ☐ public void setSemester(int semester) {
56 |     |         this.semester = semester;
57 |     |     }
58 |
59 | ☐ public float getIpk() {
60 |     |         return ipk;
61 |     |     }
62 |
63 | ☐ public void setIpk(float ipk) {
64 |     |         this.ipk = ipk;
65 |     |     }
66 |
67 |
68 |     }
69 |

```

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.controller;
6
7   import com.mahasiswa.model.MahasiswaDAO;
8   import com.mahasiswa.model.ModelMahasiswa;
9   import java.util.List;
10
11  /**
12   *
13   * @author LAB_CENGKARENG
14   */
15  public class MahasiswaController {
16      private MahasiswaDAO mahasiswaDAO;
17
18      public MahasiswaController(MahasiswaDAO mahasiswaDAO) {
19          this.mahasiswaDAO = mahasiswaDAO;
20      }
21
22      public void displayMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
23          if (mahasiswaList.isEmpty()) {
24              System.out.println("Tidak ada data mahasiswa");
25          } else {
26              System.out.println("");
27              System.out.println("=====");
28          }
29      }
30  }
```

```
28  ☐      for (ModelMahasiswa m: mahasiswaList){
29      System.out.println("ID          : " + m.getId());
30      System.out.println("NPM         : " + m.getNpm());
31      System.out.println("NAMA        : " + m.getNama());
32      System.out.println("SEMESTER    : " + m.getSemester());
33      System.out.println("IPK         : " + m.getIpk());
34      System.out.println("=====");
35  }
36  }
37  }
38
39
40  ☐ public void displayMessage(String message){
41      System.out.println(message);
42  }
43
44
45
46  ☐ public void checkDatabaseConnection(){
47      boolean isConnected = mahasiswaDAO.checkConnection();
48      ☐ if (isConnected){
49          displayMessage("Koneksi ke db berhasil");
50      } else{
51          displayMessage("Koneksi DB Gagal");
52      }
53  }
54
```

```

55 // READ ALL (Menampilkan semua mahasiswa)
56 public void displayAllMahasiswa() {
57     List<ModelMahasiswa> mahasiswaList = mahasiswaDAO.getAllMahasiswa();
58     displayMahasiswaList(mahasiswaList);
59 }
60
61 public void addMahasiswa(String npm, String nama, int semester, float ipk) {
62     ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm, nama, semester, ipk);
63     System.out.println("Controller Data: " + npm + nama + semester + ipk);
64     System.out.println(mahasiswaBaru);
65     mahasiswaDAO.addMahasiswa(mahasiswaBaru);
66     displayMessage("Mahasiswa berhasil ditambahkan!");
67 }
68
69 public void updateMahasiswa(int id, String npm, String nama, int semester, float ipk) {
70     ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(id, npm, nama, semester, ipk);
71     mahasiswaDAO.updateMahasiswa(mahasiswaBaru);
72     displayMessage("Mahasiswa berhasil diperbarui!");
73 }
74
75 public void deleteMahasiswa(int id) {
76     mahasiswaDAO.deleteMahasiswa(id);
77     displayMessage("Mahasiswa Berhasil Dihapus!");
78 }
79
80 public void closeConnection() {
81     mahasiswaDAO.closeConnection();
82 }
83 }
84
85

```

MahasiswaDAO

```
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 java.sql.*;
8   import java.util.ArrayList;
9   import java.util.List;
10
11  /**
12   *
13   * @author LAB_CENGKARENG
14   */
15  public class MahasiswaDAO{
16      private Connection connection;
17
18      public MahasiswaDAO(){
19          try{
20              Class.forName("com.mysql.cj.jdbc.Driver");
21              connection = DriverManager.getConnection("jdbc:mysql://localhost:3306/shaun_mvc", "root", "");
22          } catch (Exception e) {
23              e.printStackTrace();
24          }
25      }
26      public boolean checkConnection(){
27          try{
28              if(connection != null && !connection.isClosed()){
29                  return true;
30              }
31          } catch (SQLException e) {
32              e.printStackTrace();
33          }
34          return false;
35      }
36      public void addMahasiswa(ModelMahasiswa mahasiswa){
37          String sql = "INSERT INTO mahasiswa (npm, nama, semester, ipk) VALUES (?, ?, ?, ?)";
38          try{
39              PreparedStatement pstmt = connection.prepareStatement(sql);
40              pstmt.setString(1, mahasiswa.getNpm());
41              pstmt.setString(2, mahasiswa.getNama());
42              pstmt.setInt(3, mahasiswa.getSemester());
43              pstmt.setFloat(4, mahasiswa.getIpk());
44              pstmt.executeUpdate();
45          } catch (SQLException e) {
46              e.printStackTrace();
47          }
48      }
49
50      public List<ModelMahasiswa> getAllMahasiswa(){
51          List<ModelMahasiswa> mahasiswaList = new ArrayList<>();
52          String sql = "SELECT * FROM mahasiswa";
53          try{
54              Statement stmt = connection.createStatement();
```

```

55         ResultSet rs = stmt.executeQuery(sql);
56         while(rs.next()){
57             mahasiswaList.add(new ModelMahasiswa(
58                 rs.getInt("id"),
59                 rs.getString("npm"),
60                 rs.getString("nama"),
61                 rs.getInt("semester"),
62                 rs.getFloat("ipk")
63             ));
64         }
65     } catch (SQLException e) {
66         e.printStackTrace();
67     }
68     return mahasiswaList;
69 }
70
71 public void updateMahasiswa(ModelMahasiswa mahasiswa){
72     String sql = "UPDATE mahasiswa SET npm = ?, nama = ?, semester = ?, ipk = ? WHERE id = ?";
73     try{
74         PreparedStatement pstmt = connection.prepareStatement(sql);
75         pstmt.setString(1, mahasiswa.getNpm());
76         pstmt.setString(2, mahasiswa.getNama());
77         pstmt.setInt(3, mahasiswa.getSemester());
78         pstmt.setFloat(4, mahasiswa.getIpk());
79         pstmt.setInt(5, mahasiswa.getId());
80         pstmt.executeUpdate();
81     } catch (SQLException e) {

```

MahasiswaView

```

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.view;
6
7  import com.mahasiswa.controller.MahasiswaController;
8  import com.mahasiswa.model.MahasiswaDAO;
9  import java.util.Scanner;
10
11  /**
12   *
13   * @author LAB_CENGKARENG
14   */
15  public class MahasiswaView {
16      public static void main(String[] args){
17          MahasiswaDAO mahasiswaDAO = new MahasiswaDAO();
18          MahasiswaController mahasiswaController = new MahasiswaController(mahasiswaDAO);
19
20          Scanner scanner = new Scanner(System.in);
21          int pilihan;
22
23          while(true){
24              System.out.println("Menu:");
25              System.out.println("1. Tampilkan Semua Mahasiswa");
26              System.out.println("2. Tambah Mahasiswa");
27              System.out.println("3. Update Mahasiswa");

```



```

28      System.out.println("4. Hapus Mahasiswa");
29      System.out.println("5. Cek Koneksi Database");
30      System.out.println("6. Keluar");
31      System.out.print("PILIH OPSI: ");
32      pilihan = scanner.nextInt();
33      scanner.nextLine();
34
35      switch (pilihan){
36          case 1:
37              mahasiswaController.displayAllMahasiswa();
38              break;
39
40          case 2:
41              // tambah mhs
42              System.out.println("Masukkan NPM: ");
43              String npm = scanner.next();
44              System.out.println("Masukkan Nama: ");
45              String nama = scanner.next();
46              System.out.println("Masukkan Semester: ");
47              int semester = scanner.nextInt();
48              System.out.println("Masukkan IPK: ");
49              float ipk = scanner.nextFloat();
50              System.out.println(npm + nama + semester + ipk);
51
52              mahasiswaController.addMahasiswa(npm, nama, semester, ipk);
53              break;
54
55          case 3:
56              System.out.print("Masukkan ID mahasiswa: ");
57              int id = scanner.nextInt();
58              scanner.nextLine();
59
60              System.out.println("Masukkan NPM: ");
61              String npmBaru = scanner.next();
62              System.out.println("Masukkan Nama: ");
63              String namaBaru = scanner.next();
64              System.out.println("Masukkan Semester: ");
65              int semesterBaru = scanner.nextInt();
66              System.out.println("Masukkan IPK: ");
67              float ipkBaru = scanner.nextFloat();
68
69              mahasiswaController.updateMahasiswa(id, npmBaru, namaBaru, semesterBaru, ipkBaru);
70              break;
71          case 4:
72              System.out.print("Masukkan ID Mahasiswa: ");
73              int idHapus = scanner.nextInt();
74              mahasiswaController.deleteMahasiswa(idHapus);
75          case 5:
76              mahasiswaController.checkDatabaseConnection();
77              break;
78          case 6:
79              // Keluar
80              mahasiswaController.closeConnection();
81              System.out.println("Program selesai.");

```

```

82         return;
83     default:
84         System.out.println("Input Tidak valid");
85     }
86 }
87 }
88 }

```

Output:

```

Output - Run (MahasiswaView) x
-----< com.mycompany:mavenproject1 >-----
Building mavenproject1 1.0-SNAPSHOT
from pom.xml
-----[ jar ]-----
The artifact mysql:mysql-connector-java:jar:8.0.33 has been relocated to com.mysql:mysql-connector-j:jar:8.0.33: MySQL Conne
--- resources:3.3.1:resources (default-resources) @ mavenproject1 ---
skip non existing resourceDirectory C:\Users\LAB_CENGKARENG\Documents\NetBeansProjects\mavenproject1\src\main\resources
--- compiler:3.13.0:compile (default-compile) @ mavenproject1 ---
Nothing to compile - all classes are up to date.
--- exec:3.1.0:exec (default-cli) @ mavenproject1 ---
Menu:
1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI: 5
Koneksi ke db berhasil

```

Menu:

1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar

PILIH OPSI: 1

=====

ID : 2
NPM : 876876
NAMA : jlj
SEMESTER : 6
IPK : 4.0

=====

ID : 3
NPM : 124124
NAMA : yuy
SEMESTER : 5
IPK : 3.0

=====

Menu:

1. Tampilkan Semua Mahasiswa
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar

PILIH OPSI: |

Pertemuan 4

Source Code:

Mahasiswa_orm

```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   */
4
5   package com.mycompany.mahasiswa_orm;
6
7   /**
8    *
9    * @author Shaun
10   */
11   public class Mahasiswa_orm {
12
13       public static void main(String[] args) {
14           System.out.println("Hello World!");
15       }
16   }
17
```

Mahasiswa Controller

```
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
8   import com.mahasiswa.model.HibernateUtil;
9   import com.mahasiswa.model.ModelMahasiswa;
10  import java.util.List;
11  import org.hibernate.Session;
12  import org.hibernate.Transaction;
13  import org.hibernate.query.Query;
14
15  public class MahasiswaController {
16
17      public void addMhs(ModelMahasiswa mhs) {
18          Transaction trx = null;
19
20          try (Session session = HibernateUtil.getSessionFactory().openSession()) {
21              trx = session.beginTransaction();
22              session.save(mhs);
23              trx.commit();
24          } catch (Exception e) {
25              if (trx != null) {
26                  trx.rollback();
27              }
28              e.printStackTrace();
29          }
30      }
31
32
33      public void updateMhs(ModelMahasiswa mhs) {
```

```

34 Transaction trx = null;
35
36 try (Session session = HibernateUtil.getSessionFactory().openSession()) {
37     trx = session.beginTransaction();
38     session.update(mhs);
39     trx.commit();
40 } catch (Exception e) {
41     if (trx != null) {
42         trx.rollback();
43     }
44     e.printStackTrace();
45 }
46
47 }
48
49 public void deleteMhs(int id) {
50     Transaction trx = null;
51
52     try (Session session = HibernateUtil.getSessionFactory().openSession()) {
53         trx = session.beginTransaction();
54         ModelMahasiswa mhs = session.get(ModelMahasiswa.class, id);
55         if (mhs != null) {
56             session.delete(mhs);
57             System.out.println("Berhasil hapus");
58         }
59         trx.commit();
60     } catch (Exception e) {
61         if (trx != null) {
62             trx.rollback();
63         }
64         e.printStackTrace();
65     }
66 }
67
68
69 public List<ModelMahasiswa> getAllMahasiswa() {
70     Transaction trx = null;
71     List<ModelMahasiswa> listMhs = null;
72
73     try (Session session = HibernateUtil.getSessionFactory().openSession()) {
74         trx = session.beginTransaction();
75         // Using HQL (Hibernate Query Language) to fetch all records
76         Query<ModelMahasiswa> query = session.createQuery("from ModelMahasiswa", ModelMahasiswa.class);
77         listMhs = query.list(); // Fetch all results
78
79         trx.commit(); // Commit transaction
80     } catch (Exception e) {
81         if (trx != null) {
82             trx.rollback(); // Rollback transaction in case of error
83         }
84         e.printStackTrace();
85     }
86
87     // Return the fetched list
88     return listMhs;
89 }
90

```

Model Mahasiswa

```
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 Shaun
12    */
13   @Entity
14   @Table(name = "mahasiswa")
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     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 setNama(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
87
88 }
89

```

Mahasiswa View

```
1  /**
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3   * Click nbfs://nbhost/SystemFileSystem/Templates/GUIForms/JFrame.java to edit this template
4   */
5   package com.mahasiswa.view;
6
7   import com.mahasiswa.controller.MahasiswaController;
8   import com.mahasiswa.model.HibernateUtil;
9   import com.mahasiswa.model.ModelMahasiswa;
10  import com.mahasiswa.model.ModelTabelMahasiswa;
11  import java.util.List;
12  import javax.swing.*;
13
14
15  /**
16   *
17   * @author Shaun
18   */
19  public class MahasiswaView extends javax.swing.JFrame {
20
21      /**
22       * Creates new form MahasiswaView2
23       */
24      private MahasiswaController controller;
25
26
27
28      public MahasiswaView() {
29          initComponents();
30          HibernateUtil.testConnection();
31          controller = new MahasiswaController();
32          loadMahasiswaTable();
33      }
```

```
34
35      public void loadMahasiswaTable() {
36          List<ModelMahasiswa> listMahasiswa = controller.getAllMahasiswa();
37
38          ModelTabelMahasiswa tableModel = new ModelTabelMahasiswa(listMahasiswa);
39          dataTable.setModel(tableModel);
40      }
41
42      /**
43       * This method is called from within the constructor to initialize the form.
44       * WARNING: Do NOT modify this code. The content of this method is always
45       * regenerated by the Form Editor.
46       */
47      @SuppressWarnings("unchecked")
48      Generated Code
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87      private void npmFieldActionPerformed(java.awt.event.ActionEvent evt) {
88          // TODO add your handling code here:
89      }
90
91      private void simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
92          // TODO add your handling code here:
93          String npm = getNpmField().getText();
94          String nama = getNamaField().getText();
95          int semester = Integer.parseInt(getSemesterField().getText());
96          float ipk = Float.parseFloat(getIpkField().getText());
97          ModelMahasiswa mahasiswa = new ModelMahasiswa(0, npm, nama, semester, ipk);
98          controller.addMhs(mahasiswa);
99          loadMahasiswaTable();
100      }
101
102      private void buangButtonActionPerformed(java.awt.event.ActionEvent evt) {
```



```

203 // TODO add your handling code here:
204 // Membuat JTextField untuk memasukkan ID
205 JTextField idField = new JTextField(10);
206
207 // Membuat panel untuk menampung JTextField
208 JPanel panel = new JPanel();
209 panel.add(new JLabel("Masukkan ID yang ingin dihapus:"));
210 panel.add(idField);
211
212 // Menampilkan dialog box dengan JTextField, tombol OK, dan Cancel
213 int result = JOptionPane.showConfirmDialog(null, panel,
214     "Hapus Mahasiswa", JOptionPane.OK_CANCEL_OPTION, JOptionPane.PLAIN_MESSAGE);
215
216 // Jika tombol OK ditekan
217 if (result == JOptionPane.OK_OPTION) {
218     try {
219         // Mengambil input ID dan memanggil metode deleteMhs
220         int id = Integer.parseInt(idField.getText());
221         controller.deleteMhs(id);
222         JOptionPane.showMessageDialog(null, "Data berhasil dihapus.", "Sukses", JOptionPane.INFORMATION_MESSAGE);
223     } catch (NumberFormatException e) {
224         // Menangani error jika ID yang dimasukkan bukan angka
225         JOptionPane.showMessageDialog(null, "ID harus berupa angka.", "Error", JOptionPane.ERROR_MESSAGE);
226     }
227 }
228
229
230 private void refreshButtonActionPerformed(java.awt.event.ActionEvent evt) {
231     // TODO add your handling code here:
232     loadMahasiswaTable();
233 }
234

```

```

234
235 /**
236  * @param args the command line arguments
237  */
238 public static void main(String args[]) {
239     /* Set the Nimbus look and feel */
240     Look and feel setting code (optional)
241     //</editor-fold>
242
243     /* Create and display the form */
244     java.awt.EventQueue.invokeLater(new Runnable() {
245         public void run() {
246             new MahasiswaView().setVisible(true);
247         }
248     });
249 }
250

```

```

251 // Variables declaration - do not modify
252 private javax.swing.JButton buangButton;
253 private javax.swing.JTable dataTable;
254 private javax.swing.JTextField ipkField;
255 private javax.swing.JLabel jLabel1;
256 private javax.swing.JLabel jLabel2;
257 private javax.swing.JLabel jLabel3;
258 private javax.swing.JLabel jLabel4;
259 private javax.swing.JScrollPane jScrollPane1;
260 private javax.swing.JTextField namaField;
261 private javax.swing.JTextField npmField;
262 private javax.swing.JButton refreshButton;
263 private javax.swing.JTextField semesterField;
264 private javax.swing.JButton simpanButton;
265 // End of variables declaration

```

```
286
287 [-] public JTextField getIpkField() {
288     return ipkField;
289 }
290
291 [-] public void setIpkField(JTextField ipkField) {
292     this.ipkField = ipkField;
293 }
294
295 [-] public JTextField getNamaField() {
296     return namaField;
297 }
298
299 [-] public void setNamaField(JTextField namaField) {
300     this.namaField = namaField;
301 }
302
303 [-] public JTextField getNpmField() {
304     return npmField;
305 }
306
307 [-] public void setNpmField(JTextField npmField) {
308     this.npmField = npmField;
309 }
310
311 [-] public JTextField getSemesterField() {
312     return semesterField;
313 }
314
315 [-] public void setSemesterField(JTextField semesterField) {
316     this.semesterField = semesterField;
317 }
```

Hibernate Util

```
1  package com.mahasiswa.model;
2
3  import org.hibernate.Session;
4  import org.hibernate.SessionFactory;
5  import org.hibernate.cfg.Configuration;
6
7  public class HibernateUtil {
8      private static SessionFactory sessionFactory;
9
10     static {
11         try {
12             // Create the SessionFactory from hibernate.cfg.xml
13             sessionFactory = new Configuration().configure().buildSessionFactory();
14         } catch (Throwable ex) {
15             // Make sure you log the exception, as it might be swallowed
16             System.err.println("Initial SessionFactory creation failed." + ex);
17             throw new ExceptionInInitializerError(ex);
18         }
19     }
20
21     public static SessionFactory getSessionFactory() {
22         return sessionFactory;
23     }
24
25     public static void testConnection() {
26         try (Session session = sessionFactory.openSession()) {
27             System.out.println("Connection to the database was successful!");
28         } catch (Exception e) {
29             System.err.println("Failed to connect to the database.");
30             e.printStackTrace();
31         }
32     }
33 }
```

```

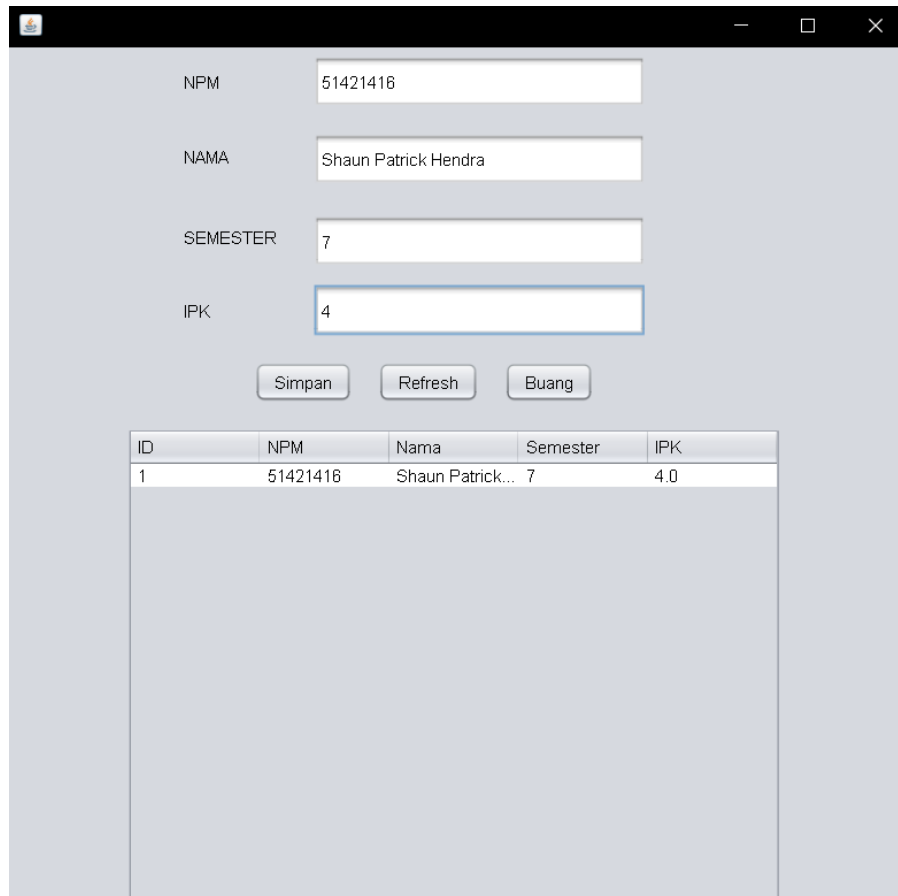
35         switch (columnIndex) {
36             case 0:
37                 return mahasiswa.getId();
38             case 1:
39                 return mahasiswa.getNpm();
40             case 2:
41                 return mahasiswa.getNama();
42             case 3:
43                 return mahasiswa.getSemester();
44             case 4:
45                 return mahasiswa.getIpk();
46             default:
47                 return null;
48         }
49     }
50     @Override
51     public String getColumnName(int column) {
52         return columnNames[column]; // Mengatur nama kolom
53     }
54
55     @Override
56     public boolean isCellEditable(int rowIndex, int columnIndex) {
57         return false; // Semua sel tidak dapat diedit
58     }
59
60     // Method untuk menambahkan atau memodifikasi data, jika dibutuhkan
61     public void setMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
62         this.mahasiswaList = mahasiswaList;
63         fireTableDataChanged(); // Memberitahu JTable bahwa data telah berubah
64     }
65 }
66

```

ModelTabelMahasiswa

```
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   *
11   * @author Shaun
12   */
13  public class ModelTabelMahasiswa extends AbstractTableModel{
14      private List<ModelMahasiswa> mahasiswaList;
15      private String[] columnNames = {"ID", "NPM", "Nama", "Semester", "IPK"};
16
17      public ModelTabelMahasiswa(List<ModelMahasiswa> mahasiswaList) {
18          this.mahasiswaList = mahasiswaList;
19      }
20
21      @Override
22      public int getRowCount() {
23          return mahasiswaList.size(); // Jumlah baris sesuai dengan jumlah data mahasiswa
24      }
25
26      @Override
27      public int getColumnCount() {
28          return columnNames.length; // Jumlah kolom sesuai dengan jumlah elemen dalam columnNames
29      }
30
31      @Override
32      public Object getValueAt(int rowIndex, int columnIndex) {
33          ModelMahasiswa mahasiswa = mahasiswaList.get(rowIndex);
```

Output:



The screenshot shows a web application window with a light gray background. At the top, there is a black title bar with a small icon on the left and standard window controls (minimize, maximize, close) on the right. Below the title bar, the main content area contains a form with four input fields, each with a label to its left: "NPM" with the value "51421416", "NAMA" with the value "Shaun Patrick Hendra", "SEMESTER" with the value "7", and "IPK" with the value "4". The "IPK" field is currently selected with a blue border. Below the input fields are three buttons: "Simpan", "Refresh", and "Buang". At the bottom of the form is a table with five columns: "ID", "NPM", "Nama", "Semester", and "IPK". The table contains one data row with the following values: "1", "51421416", "Shaun Patrick...", "7", and "4.0".

ID	NPM	Nama	Semester	IPK
1	51421416	Shaun Patrick...	7	4.0