

Listing

```
1 // File: Mahasiswa.java
2 package model;
3
4 import javax.persistence.*;
5
6 @Entity
7 @Table(name = "mahasiswa")
8 public class Mahasiswa {
9     @Id
10     private String nim;
11     private String nama;
12     private String alamat;
13     private String nid_ds; // Field tambahan sesuai permintaan
14
15     // Constructors
16     public Mahasiswa() {}
17
18     public Mahasiswa(String nim, String nama, String alamat, String nid_ds) {
19         this.nim = nim;
20         this.nama = nama;
21         this.alamat = alamat;
22         this.nid_ds = nid_ds;
23     }
24
25     // Getters and Setters
26     public String getNim() {
27         return nim;
28     }
29
30     public void setNim(String nim) {
31         this.nim = nim;
32     }
33
34     public String getNama() {
35         return nama;
36     }
37
38     public void setNama(String nama) {
39         this.nama = nama;
40     }
41
42     public String getAlamat() {
43         return alamat;
44     }
45
46     public void setAlamat(String alamat) {
47         this.alamat = alamat;
48     }
49
50     public String getNidDs() {
51         return nid_ds;
52     }
53
54     public void setNidDs(String nid_ds) {
```

```

55         this.nid_ds = nid_ds;
56     }
57 }
58
59 // File: HibernateUtil.java
60 package util;
61
62 import org.hibernate.SessionFactory;
63 import org.hibernate.cfg.Configuration;
64
65 public class HibernateUtil {
66     private static final SessionFactory sessionFactory;
67
68     static {
69         try {
70             sessionFactory = new Configuration().configure().buildSessionFactory();
71         } catch (Throwable ex) {
72             System.err.println("Initial SessionFactory creation failed." + ex);
73             throw new ExceptionInInitializerError(ex);
74         }
75     }
76
77     public static SessionFactory getSessionFactory() {
78         return sessionFactory;
79     }
80 }
81
82 // File: MahasiswaController.java
83 package controller;
84
85 import model.Mahasiswa;
86 import org.hibernate.Session;
87 import org.hibernate.Transaction;
88 import util.HibernateUtil;
89
90 public class MahasiswaController {
91     // Create/Update
92     public void saveOrUpdate(Mahasiswa mahasiswa) {
93         Session session = HibernateUtil.getSessionFactory().openSession();
94         Transaction transaction = null;
95
96         try {
97             transaction = session.beginTransaction();
98             session.saveOrUpdate(mahasiswa);
99             transaction.commit();
100         } catch (Exception e) {
101             if (transaction != null) {
102                 transaction.rollback();
103             }
104             e.printStackTrace();
105         } finally {
106             session.close();

```

```

107     }
108 }
109
110 // Delete
111 public void delete(Mahasiswa mahasiswa) {
112     Session session = HibernateUtil.getSessionFactory().openSession();
113     Transaction transaction = null;
114
115     try {
116         transaction = session.beginTransaction();
117         session.delete(mahasiswa);
118         transaction.commit();
119     } catch (Exception e) {
120         if (transaction != null) {
121             transaction.rollback();
122         }
123         e.printStackTrace();
124     } finally {
125         session.close();
126     }
127 }
128
129 // Read
130 public Mahasiswa getMahasiswaByNim(String nim) {
131     Session session = HibernateUtil.getSessionFactory().openSession();
132     Mahasiswa mahasiswa = null;
133
134     try {
135         mahasiswa = (Mahasiswa) session.get(Mahasiswa.class, nim);
136     } catch (Exception e) {
137         e.printStackTrace();
138     } finally {
139         session.close();
140     }
141
142     return mahasiswa;
143 }
144 }
145
146 // File: MahasiswaForm.java
147 package view;
148
149 import controller.MahasiswaController;
150 import model.Mahasiswa;
151 import javax.swing.*;
152 import java.awt.*;
153 import java.awt.event.*;
154
155 public class MahasiswaForm extends JFrame {
156     private JTextField txtNim, txtNama, txtAlamat, txtNidDs;

```

```

156 private JTextField txtNim, txtNama, txtAlamat, txtNidDs;
157 private JButton btnSimpan, btnEdit, btnHapus, btnClear;
158 private MahasiswaController controller;
159
160 public MahasiswaForm() {
161     controller = new MahasiswaController();
162     initComponents();
163 }
164
165 private void initComponents() {
166     setTitle("Form Mahasiswa");
167     setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
168     setSize(400, 300);
169     setLocationRelativeTo(null);
170
171     // Initialize components
172     JPanel panel = new JPanel(new GridLayout(5, 2, 5, 5));
173
174     panel.add(new JLabel("NIM:"));
175     txtNim = new JTextField();
176     panel.add(txtNim);
177
178     panel.add(new JLabel("Nama:"));
179     txtNama = new JTextField();
180     panel.add(txtNama);
181
182     panel.add(new JLabel("Alamat:"));
183     txtAlamat = new JTextField();
184     panel.add(txtAlamat);
185
186     panel.add(new JLabel("NID DS:"));
187     txtNidDs = new JTextField();
188     panel.add(txtNidDs);
189
190     // Buttons
191     JPanel buttonPanel = new JPanel(new FlowLayout());
192
193     btnSimpan = new JButton("Simpan");
194     btnEdit = new JButton("Edit");
195     btnHapus = new JButton("Hapus");
196     btnClear = new JButton("Clear");
197
198     buttonPanel.add(btnSimpan);
199     buttonPanel.add(btnEdit);
200     buttonPanel.add(btnHapus);
201     buttonPanel.add(btnClear);
202
203     // Add action listeners
204     btnSimpan.addActionListener(e -> simpanData());
205     btnEdit.addActionListener(e -> editData());
206     btnHapus.addActionListener(e -> hapusData());
207     btnClear.addActionListener(e -> clearForm());
208

```

```

210     setLayout(new BorderLayout(5, 5));
211     add(panel, BorderLayout.CENTER);
212     add(buttonPanel, BorderLayout.SOUTH);
213 }
214
215 private void simpanData() {
216     Mahasiswa mhs = new Mahasiswa();
217     mhs.setNim(txtNim.getText());
218     mhs.setNama(txtNama.getText());
219     mhs.setAlamat(txtAlamat.getText());
220     mhs.setNidDs(txtNidDs.getText());
221
222     controller.saveOrUpdate(mhs);
223     JOptionPane.showMessageDialog(this, "Data berhasil disimpan!");
224     clearForm();
225 }
226
227 private void editData() {
228     String nim = txtNim.getText();
229     Mahasiswa mhs = controller.getMahasiswaByNim(nim);
230
231     if (mhs != null) {
232         mhs.setNama(txtNama.getText());
233         mhs.setAlamat(txtAlamat.getText());
234         mhs.setNidDs(txtNidDs.getText());
235
236         controller.saveOrUpdate(mhs);
237         JOptionPane.showMessageDialog(this, "Data berhasil diupdate!");
238         clearForm();
239     } else {
240         JOptionPane.showMessageDialog(this, "Data tidak ditemukan!");
241     }
242 }
243
244 private void hapusData() {
245     String nim = txtNim.getText();
246     Mahasiswa mhs = controller.getMahasiswaByNim(nim);
247
248     if (mhs != null) {
249         controller.delete(mhs);
250         JOptionPane.showMessageDialog(this, "Data berhasil dihapus!");
251         clearForm();
252     } else {
253         JOptionPane.showMessageDialog(this, "Data tidak ditemukan!");
254     }
255 }
256
257 private void clearForm() {
258     txtNim.setText("");
259     txtNama.setText("");
260     txtAlamat.setText("");
261     txtNidDs.setText("");

```

```

262     txtNim.requestFocus();
263 }
264
265 public static void main(String[] args) {
266     SwingUtilities.invokeLater(() -> {
267         new MahasiswaForm().setVisible(true);
268     });
269 }
270 }

```

```
<> blank
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE hibernate-configuration PUBLIC
3      "-//Hibernate/Hibernate Configuration DTD 3.0//EN"
4      "http://hibernate.sourceforge.net/hibernate-configuration-3.0.dtd">
5  <hibernate-configuration>
6      <session-factory>
7          <!-- Database connection settings -->
8          <property name="hibernate.connection.driver_class">com.mysql.jdbc.Driver</property>
9          <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/tm12</property>
10         <property name="hibernate.connection.username">root</property>
11         <property name="hibernate.connection.password"></property>
12
13         <!-- JDBC connection pool settings -->
14         <property name="hibernate.c3p0.min_size">5</property>
15         <property name="hibernate.c3p0.max_size">20</property>
16
17         <!-- SQL dialect -->
18         <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
19
20         <!-- Enable Hibernate's automatic session context management -->
21         <property name="current_session_context_class">thread</property>
22
23         <!-- Echo all executed SQL to stdout -->
24         <property name="show_sql">true</property>
25
26         <!-- Drop and re-create the database schema on startup -->
27         <property name="hbm2ddl.auto">update</property>
28
29         <!-- Mapping files -->
30         <mapping class="model.Mahasiswa"/>
31     </session-factory>
32 </hibernate-configuration>
```

Output

Form Mahasiswa

NIM:

5044221

Nama:

Asep

Alamat:

Jl. Jalak 4

NID DS:

Simpan

Edit

Hapus

Clear