

Activity 3

Nama : Bayu Tirta Ardi

Kelas : 4IA28

NPM : 50421268

Mata Praktikum : Rekayasa Perangkat Lunak 2

MahasiswaController.java

```
ModelMahasiswa.java x MahasiswaDAO.java x MahasiswaController.java x MahasiswaView.java x Bayu_mvc.java x pom.xml [Bayu_mvc] x
Source History
private MahasiswaDAO mahasiswaDAO;

14
15 public MahasiswaController(MahasiswaDAO mahasiswaDAO) {
16     this.mahasiswaDAO = mahasiswaDAO;
17 }
18
19 public void displayMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
20     if (mahasiswaList.isEmpty()) {
21         System.out.println("Tidak ada data mahasiswa");
22     } else {
23         System.out.println("");
24         System.out.println("=====");
25         for (ModelMahasiswa m : mahasiswaList) {
26             System.out.println("ID : " + m.getId());
27             System.out.println("NPM : " + m.getNpm());
28             System.out.println("NAMA : " + m.getNama());
29             System.out.println("SEMESTER : " + m.getSemester());
30             System.out.println("IPK : " + m.getIpk());
31             System.out.println("=====");
32         }
33     }
34 }
35
36 public void displayMessage(String message) {
37     System.out.println(message);
38 }
39
40
41
42 public void checkDatabaseConnection() {
43     boolean isConnected = mahasiswaDAO.checkConnection();
44     if (isConnected) {
45         displayMessage("Koneksi ke db berhasil");
46     } else {
47         displayMessage("Koneksi DB Gagal");
48     }
49 }
```

MahasiswaDAO.java

```
ModelMahasiswa.java x MahasiswaDAO.java x MahasiswaController.java x MahasiswaView.java x Bayu_mvc.java x pom.xml [Bayu_mvc] x
Source History
66 pstmt.setInt(3, mahasiswa.getSemester());
67 pstmt.setFloat(4, mahasiswa.getIpk());
68 pstmt.executeUpdate();
69 } catch (SQLException e) {
70     e.printStackTrace();
71 }
72
73 public List<ModelMahasiswa> getAllMahasiswa() {
74     List<ModelMahasiswa> mahasiswaList = new ArrayList<>();
75     String sql = "SELECT * FROM mahasiswa";
76     try {
77         Statement stmt = connection.createStatement();
78         ResultSet rs = stmt.executeQuery(sql);
79         while (rs.next()) {
80             mahasiswaList.add(new ModelMahasiswa(
81                 rs.getInt("id"),
82                 rs.getString("npm"),
83                 rs.getString("nama"),
84                 rs.getInt("semester"),
85                 rs.getFloat("ipk")
86             ));
87         }
88     } catch (SQLException e) {
89         e.printStackTrace();
90     }
91     return mahasiswaList;
92 }
93
94 public void updateMahasiswa(ModelMahasiswa mahasiswa) {
95     String sql = "UPDATE mahasiswa SET npm = ?, nama = ?, semester = ?, ipk = ? WHERE id = ?";
96     try {
97         PreparedStatement pstmt = connection.prepareStatement(sql);
98         pstmt.setString(1, mahasiswa.getNpm());
99         pstmt.setString(2, mahasiswa.getNama());
100        pstmt.setInt(3, mahasiswa.getSemester());
101        pstmt.setFloat(4, mahasiswa.getIpk());
102        pstmt.setInt(5, mahasiswa.getId());
103        pstmt.executeUpdate();
104    } catch (SQLException e) {
105        e.printStackTrace();
106    }
107 }
```

ModelMahasiswa.java

```
ModelMahasiswa.java x MahasiswaDAO.java x MahasiswaController.java x MahasiswaView.java x Bayu_mvc.java x pom.xml [B
Source History
30
31 public String getNama() {
32     return nama;
33 }
34
35 public void setNama(String nama) {
36     this.nama = nama;
37 }
38
39 public int getSemester() {
40     return semester;
41 }
42
43 public void setSemester(int semester) {
44     this.semester = semester;
45 }
46
47 public float getIpk() {
48     return ipk;
49 }
50
51 public void setIpk(float ipk) {
52     this.ipk = ipk;
53 }
54 private int id;
55 private String npm;
56 private String nama;
57 private int semester;
58 private float ipk;
59
60 public ModelMahasiswa(int id, String npm, String nama, int semester, float ipk){
61     this.id = id;
62     this.npm = npm;
63     this.nama = nama;
64     this.semester = semester;
65     this.ipk = ipk;
66 }
67
```

Pom.xml

```
...va x MahasiswaDAO.java x MahasiswaController.java x MahasiswaView.java x Bayu_mvc.java x pom.xml [Bayu_mvc] x Mahasiswa_orm.java x
Source Graph Effective History
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="
3     <modelVersion>4.0.0</modelVersion>
4     <groupId>com.mycompany</groupId>
5     <artifactId>Bayu_mvc</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.bayu_mvc</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>
```

Output 3 Data Mahasiswa

```
Output - Run (MahasiswaView) ..X
2. Tambah Mahasiswa
3. Update Mahasiswa
4. Hapus Mahasiswa
5. Cek Koneksi Database
6. Keluar
PILIH OPSI: 1

=====
ID           : 2
NPM          : 50421268
NAMA         : Bayu
SEMESTER     : 7
IPK          : 3.72
=====
ID           : 3
NPM          : 5140293
NAMA         : Darma
SEMESTER     : 7
IPK          : 3.6
=====
ID           : 4
NPM          : 5392994
NAMA         : Fakir
SEMESTER     : 7
IPK          : 3.78
=====
Menu:
1. Tampilkan Semua Mahasiswa
```

Activity 4

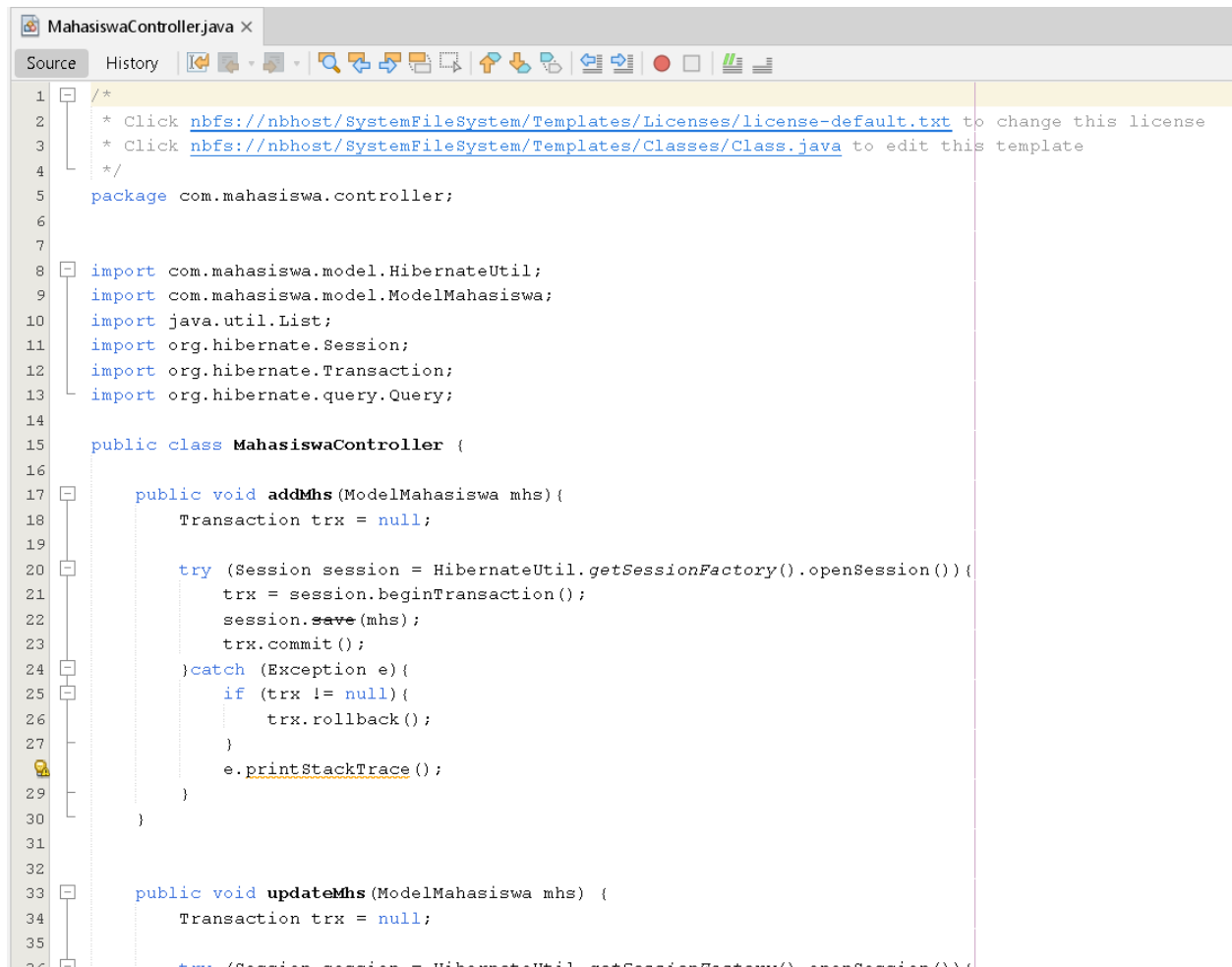
Nama : Bayu Tirta Ardi

Kelas : 4IA28

NPM : 50421268

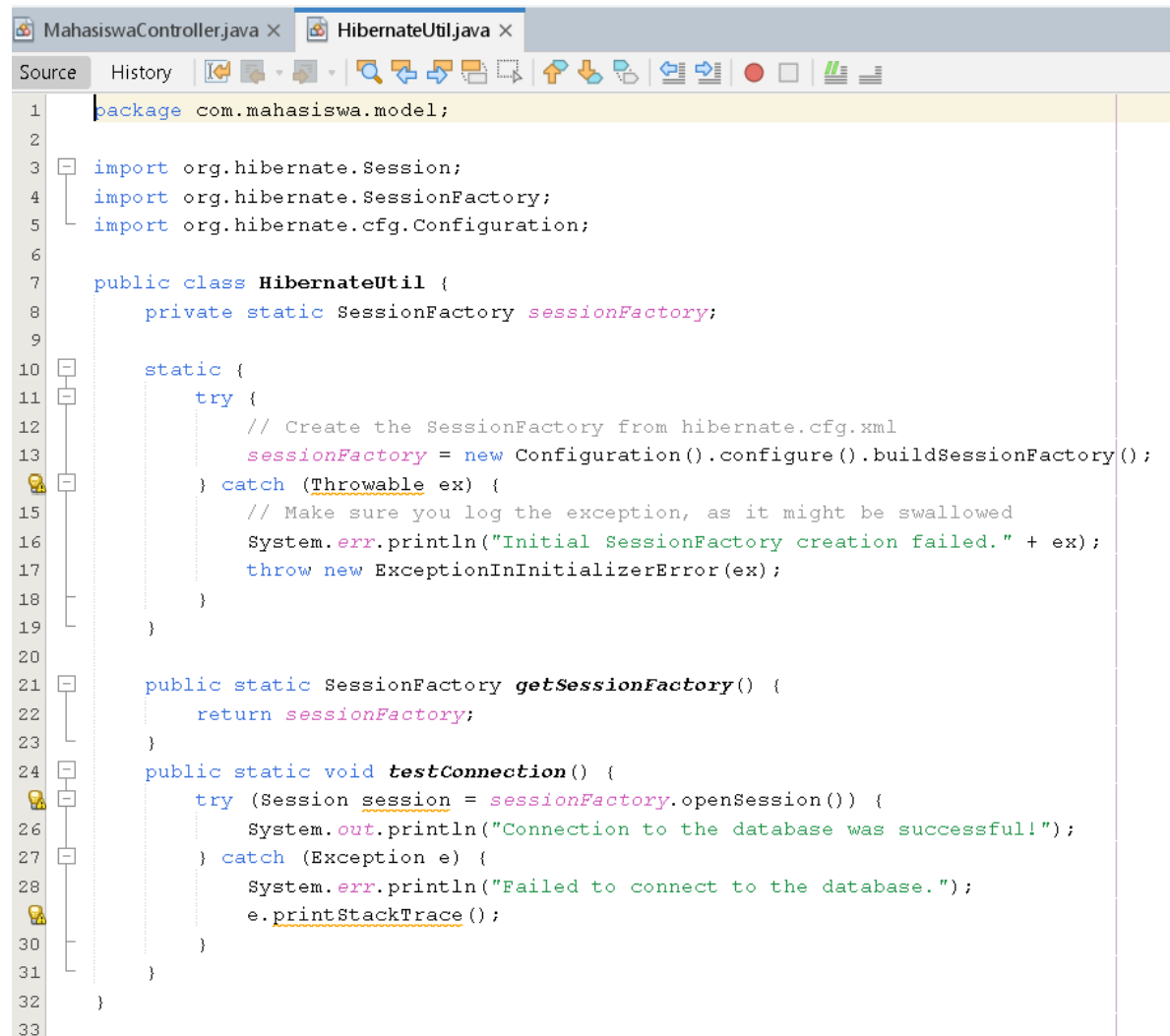
Mata Praktikum : Rekayasa Perangkat Lunak 2

MahasiswaController.java



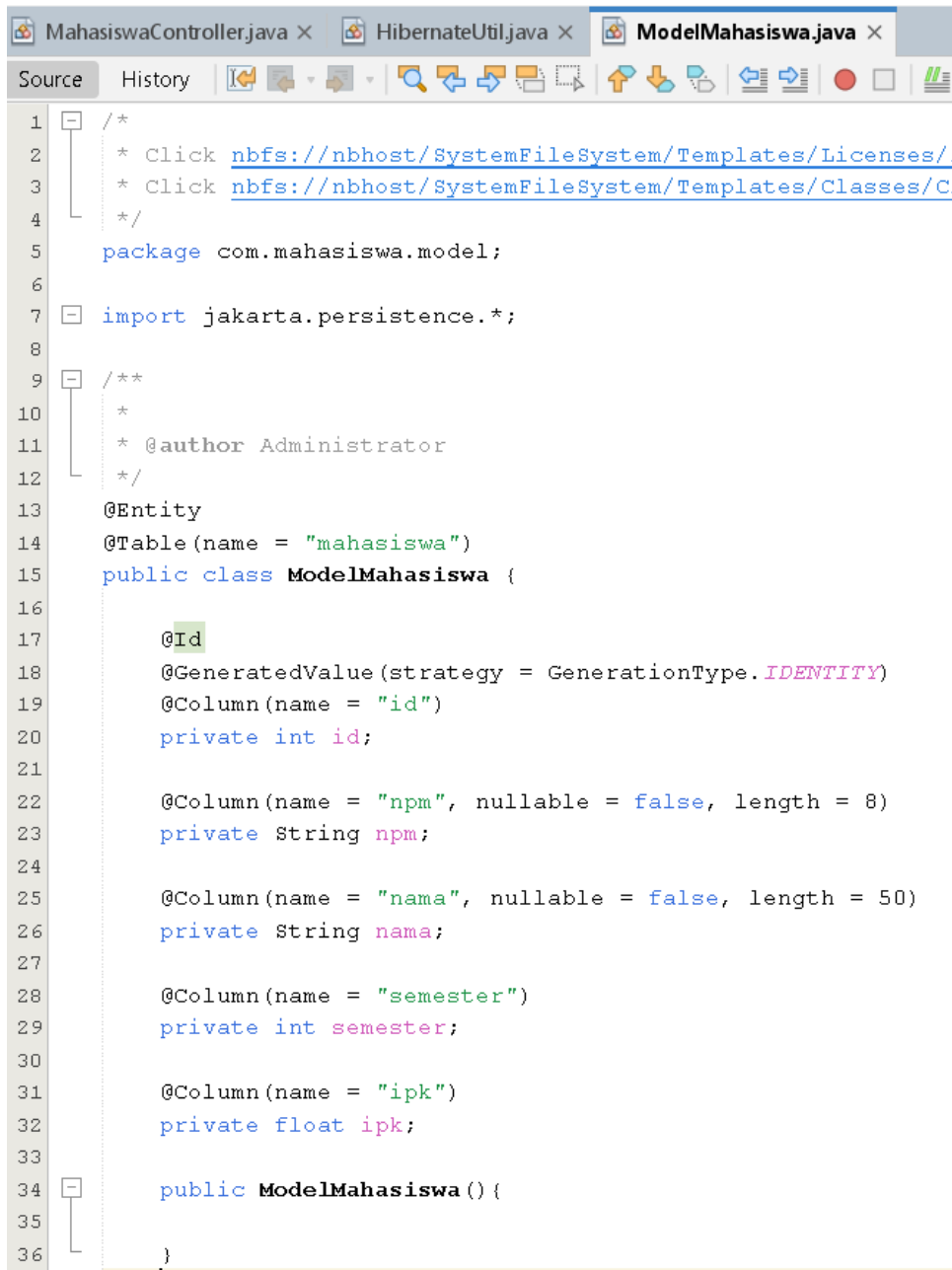
```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this license
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package com.mahasiswa.controller;
6
7
8  import com.mahasiswa.model.HibernateUtil;
9  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()) {
```

HibernateUtil.java



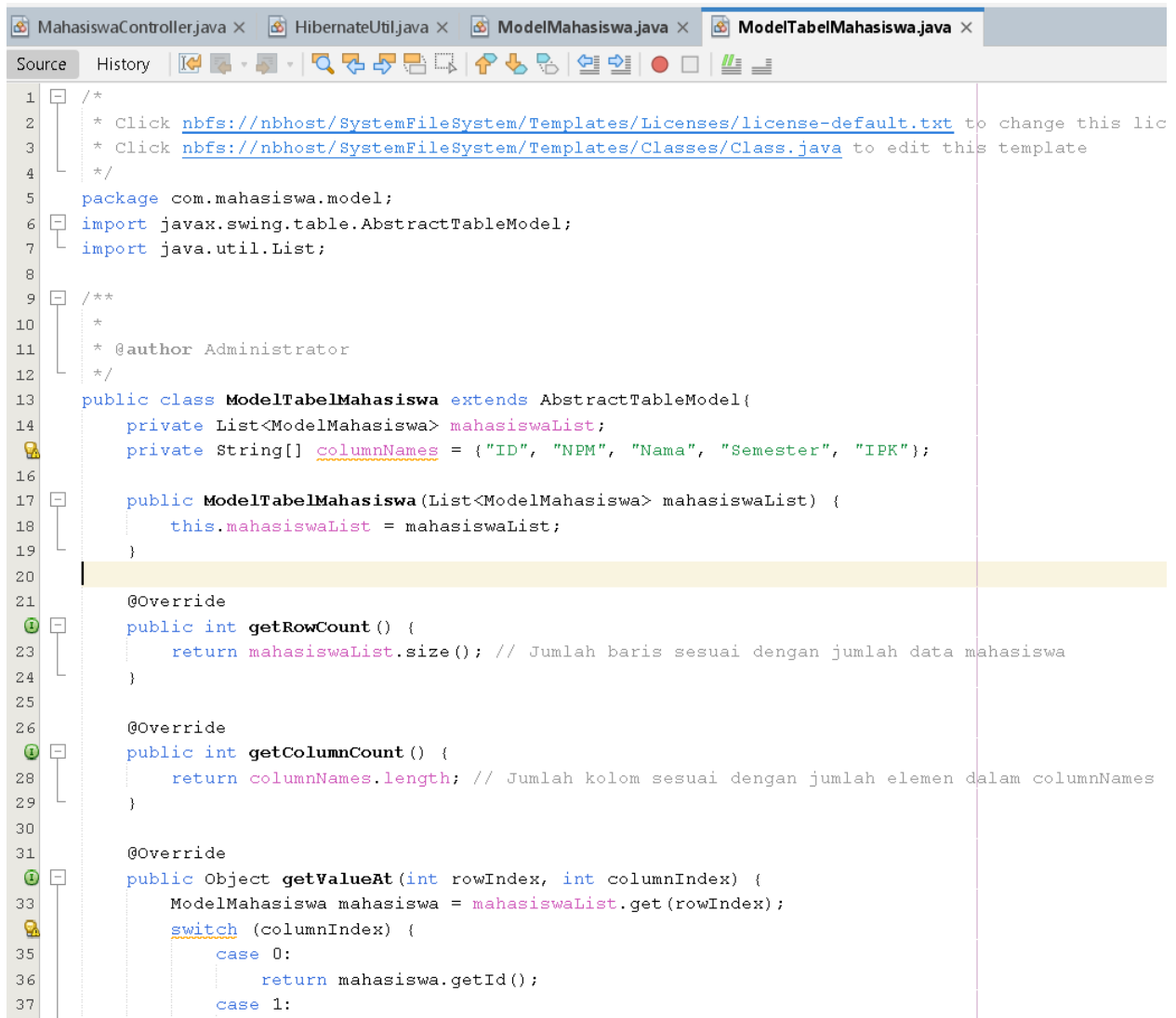
```
1 package com.mahasiswa.model;
2
3 import org.hibernate.Session;
4 import org.hibernate.SessionFactory;
5 import org.hibernate.cfg.Configuration;
6
7 public class HibernateUtil {
8     private static SessionFactory sessionFactory;
9
10    static {
11        try {
12            // Create the SessionFactory 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 }
```

ModelMahasiswa.java



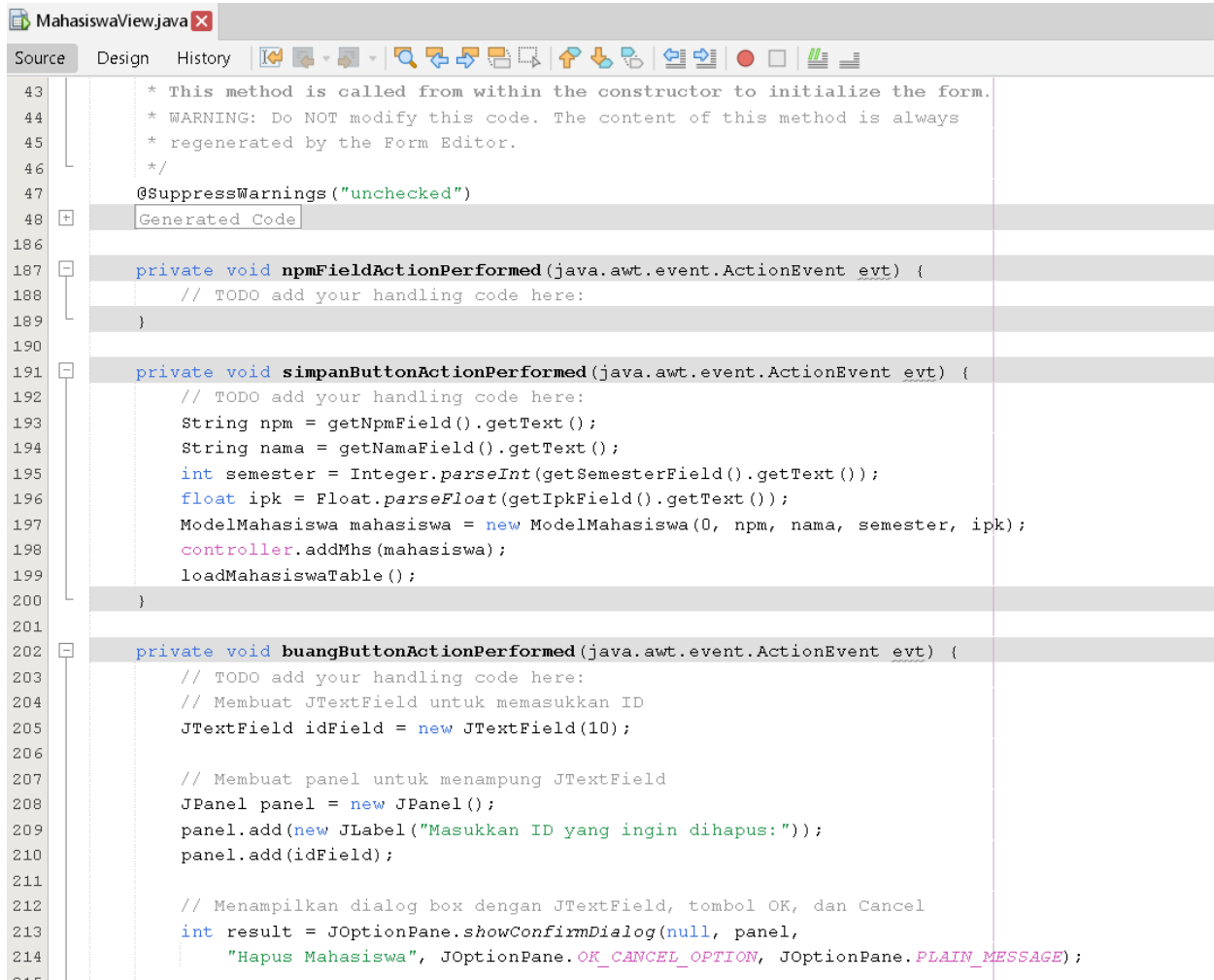
```
1  /*
2   * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/
3   * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/C
4   */
5  package com.mahasiswa.model;
6
7  import jakarta.persistence.*;
8
9  /**
10   *
11   * @author Administrator
12   */
13  @Entity
14  @Table(name = "mahasiswa")
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      }
```

ModelTabelMahasiswa.java



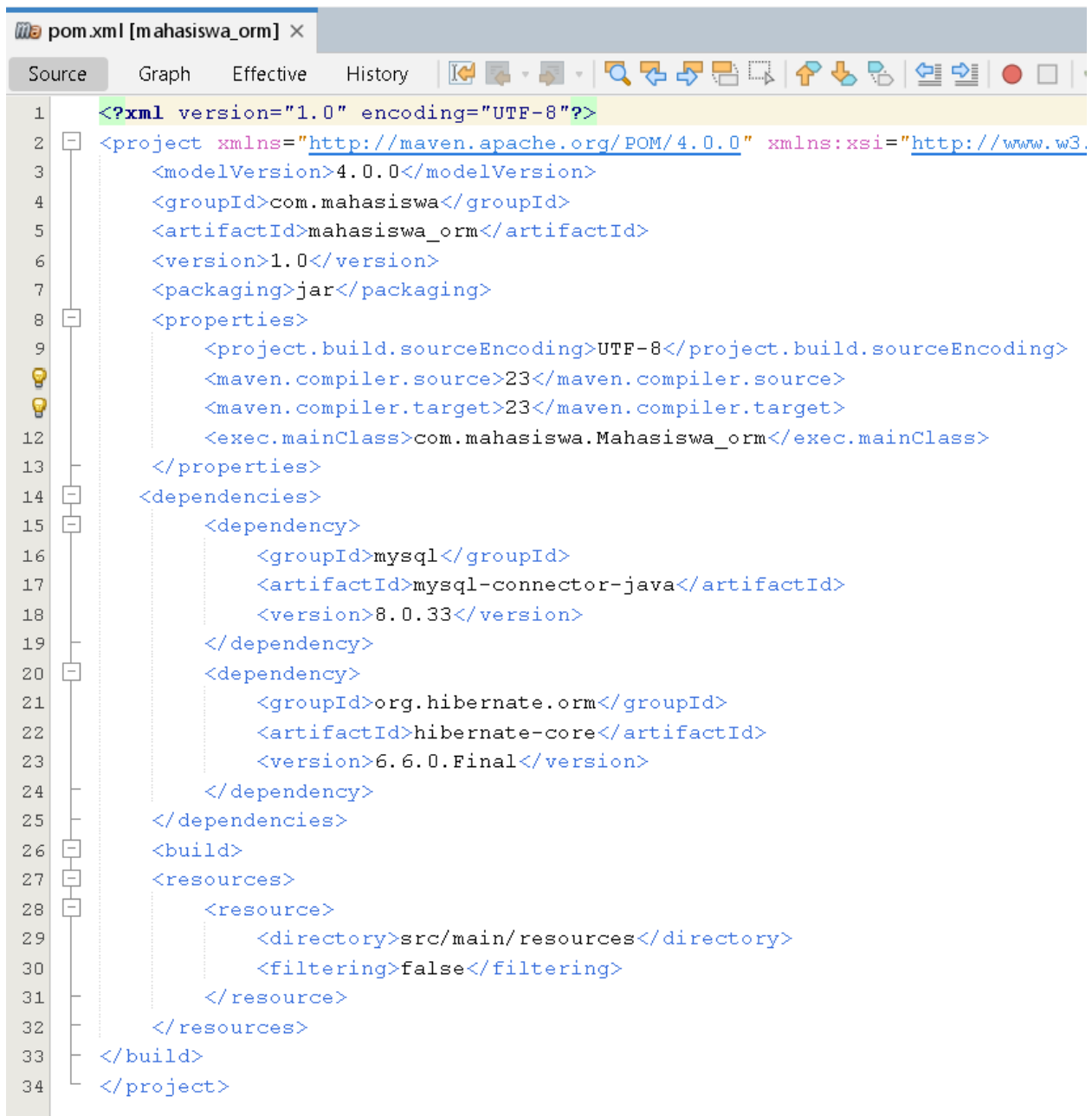
```
1  /*
2  * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-default.txt to change this lic
3  * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to edit this template
4  */
5  package com.mahasiswa.model;
6  import javax.swing.table.AbstractTableModel;
7  import java.util.List;
8
9  /**
10   *
11   * @author Administrator
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);
34          switch (columnIndex) {
35              case 0:
36                  return mahasiswa.getId();
37              case 1:
```

MahasiswaView.java



```
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
186
187      private void npmFieldActionPerformed(java.awt.event.ActionEvent evt) {
188          // TODO add your handling code here:
189      }
190
191      private void simpanButtonActionPerformed(java.awt.event.ActionEvent evt) {
192          // TODO add your handling code here:
193          String npm = getNpmField().getText();
194          String nama = getNamaField().getText();
195          int semester = Integer.parseInt(getSemesterField().getText());
196          float ipk = Float.parseFloat(getIpkField().getText());
197          ModelMahasiswa mahasiswa = new ModelMahasiswa(0, npm, nama, semester, ipk);
198          controller.addMhs(mahasiswa);
199          loadMahasiswaTable();
200      }
201
202      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);
```


Pom.xml



The screenshot shows an IDE window titled 'pom.xml [mahasiswa_orm]'. The interface includes tabs for 'Source', 'Graph', 'Effective', and 'History'. A toolbar with various icons is visible above the code editor. The code is an XML file for a Maven project, with line numbers 1 through 34 on the left. The XML content is as follows:

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.
3   <modelVersion>4.0.0</modelVersion>
4   <groupId>com.mahasiswa</groupId>
5   <artifactId>mahasiswa_orm</artifactId>
6   <version>1.0</version>
7   <packaging>jar</packaging>
8   <properties>
9     <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
10    <maven.compiler.source>23</maven.compiler.source>
11    <maven.compiler.target>23</maven.compiler.target>
12    <exec.mainClass>com.mahasiswa.Mahasiswa_orm</exec.mainClass>
13  </properties>
14  <dependencies>
15    <dependency>
16      <groupId>mysql</groupId>
17      <artifactId>mysql-connector-java</artifactId>
18      <version>8.0.33</version>
19    </dependency>
20    <dependency>
21      <groupId>org.hibernate.orm</groupId>
22      <artifactId>hibernate-core</artifactId>
23      <version>6.6.0.Final</version>
24    </dependency>
25  </dependencies>
26  <build>
27    <resources>
28      <resource>
29        <directory>src/main/resources</directory>
30        <filtering>>false</filtering>
31      </resource>
32    </resources>
33  </build>
34 </project>
```

Hibernate.cfg.xml

```
hibernate.cfg.xml x ModelLabelMahasiswa.java x ModelMahasiswa.java x HibernateUtil.java x MahasiswaController.java x
Source History
1 <?xml version="1.0" encoding="utf-8"?>
2 <!DOCTYPE hibernate-configuration PUBLIC "-//Hibernate/Hibernate Configuration DTD 3.0//EN" "http://hibernate.sourceforge.n
3 <hibernate-configuration>
4 <session-factory>
5 <!-- Database connection settings -->
6 <property name="hibernate.connection.driver_class">com.mysql.cj.jdbc.Driver</property>
7 <property name="hibernate.connection.url">jdbc:mysql://localhost:3306/bayu_50421268</property>
8 <property name="hibernate.connection.username">root</property>
9 <property name="hibernate.connection.password"></property>
10
11 <!-- JDBC connection pool settings -->
12 <property name="hibernate.c3p0.min_size">5</property>
13 <property name="hibernate.c3p0.max_size">20</property>
14 <property name="hibernate.c3p0.timeout">300</property>
15 <property name="hibernate.c3p0.max_statements">50</property>
16 <property name="hibernate.c3p0.idle_test_period">3000</property>
17
18 <!-- SQL dialect -->
19 <property name="hibernate.dialect">org.hibernate.dialect.MySQLDialect</property>
20
21 <!-- Echo all executed SQL to stdout -->
22 <property name="hibernate.show_sql">true</property>
23
24 <!-- Drop and re-create the database schema on startup -->
25 <property name="hibernate.hbm2ddl.auto">update</property>
26
27 <!-- Mapping class -->
28 <mapping class="com.mahasiswa.model.ModelMahasiswa"/>
29 </session-factory>
30 </hibernate-configuration>
```

Output

NPM

5142723

NAMA

Alghani

SEMESTER

7

IPK

1.9

Simpan

Refresh

Buang

ID	NPM	Nama	Semester	IPK
1	50421268	Bayu	7	3.72
2	5042389	Shaun	7	3.4
3	5142723	Alghani	7	1.9

ID	NPM	Nama	Semester	IPK
1	50421268	Bayu	7	3.72
2	5042389	Shaun	7	3.4
3	5142723	Alghani	7	1.9