

ACTIVITY PERTEMUAN 3

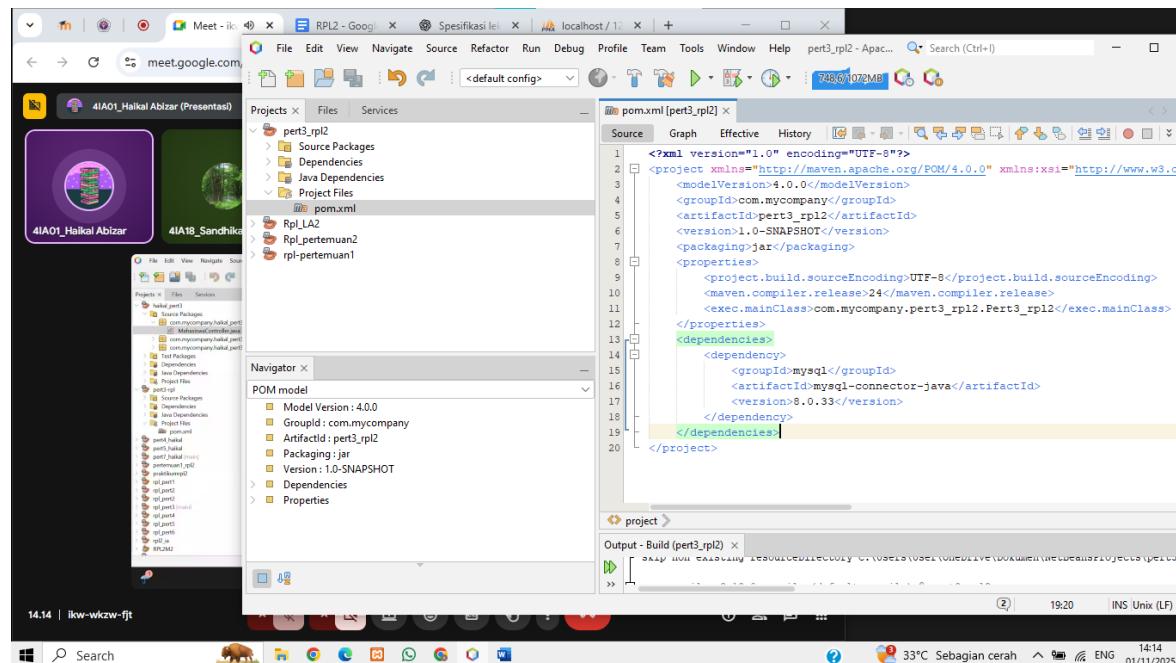
NAMA : Andi Purnama

NPM : 50422215

KELAS : 4IA18

MATERI :

MATA PRAKTIKUM : Rekayasa Perangkat Lunak 2



```
<?xml version="1.0" encoding="UTF-8"?>
<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://www.w3.org/2001/XMLSchema-instance">
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany</groupId>
  <artifactId>pert3_rpl2</artifactId>
  <version>1.0-SNAPSHOT</version>
  <packaging>jar</packaging>
  <properties>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    <maven.compiler.release>24</maven.compiler.release>
    <exec.mainClass>com.mycompany.pert3_rpl2.Pert3_rpl2</exec.mainClass>
  </properties>
  <dependencies>
    <dependency>
      <groupId>mysql</groupId>
      <artifactId>mysql-connector-java</artifactId>
      <version>8.0.33</version>
    </dependency>
  </dependencies>
</project>
```

Output - Build (pert3_rpl2) :

```
--- compiler:3.13.0:compile (default-compile) @ pert3_rpl2 ---
Recompiling the module because of changed dependency.
- Compiling 5 source files with javac [debug release 24] to target/classes

--- resources:3.3.1:testResources (default-testResources) @ pert3_rpl2 ---
skip non existing resourceDirectory C:\Users\User\OneDrive\Documents\NetBeansProjects\pert3_rpl2\src\test\resources

--- compiler:3.13.0:testCompile (default-testCompile) @ pert3_rpl2 ---
- No sources to compile

--- surefire:3.2.5:test (default-test) @ pert3_rpl2 ---
- No tests to run.

--- jar:3.4.1:jar (default-jar) @ pert3_rpl2 ---
- Building jar: C:\Users\User\OneDrive\Documents\NetBeansProjects\pert3_rpl2\target\pert3_rpl2-1.0-SNAPSHOT.jar

--- install:3.1.2:install (default-install) @ pert3_rpl2 ---
Installing G:\Users\User\OneDrive\Documents\NetBeansProjects\pert3_rpl2\pom.xml to C:\Users\User\.m2\repository\com\mycompany\pe
Installing G:\Users\User\OneDrive\Documents\NetBeansProjects\pert3_rpl2\target\pert3_rpl2-1.0-SNAPSHOT.jar to C:\Users\User\.m2\i

BUILD SUCCESS

Total time: 26.109 s
Finished at: 2025-11-01T14:13:47+07:00
```

The screenshot shows a Java code editor with the file `MahasiswaController.java` open. The code defines a `MahasiswaController` class with a constructor and a `displayMahasiswaList` method. The `displayMahasiswaList` method takes a list of `ModelMahasiswa` objects and prints their details to the console. The code uses `System.out.println` statements to output the ID, NPM, NAME, SEMESTER, and IPK for each student.

```
import com.mycompany.pert3_rpl2.Model.MahasiswaDAO;
import com.mycompany.pert3_rpl2.Model.ModelMahasiswa;
import java.util.List;

public class MahasiswaController {
    private MahasiswaDAO mahasiswaDAO;

    public MahasiswaController(MahasiswaDAO mahasiswaDAO) {
        this.mahasiswaDAO = mahasiswaDAO;
    }

    public void displayMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
        if(mahasiswaList.isEmpty()){
            System.out.println("Tidak ada data mahasiswa");
        } else {
            System.out.println("");
            System.out.println("=====");
            for(ModelMahasiswa m: mahasiswaList){
                System.out.println("ID : " + m.getId());
                System.out.println("NPM : " + m.getNpm());
                System.out.println("NAME : " + m.getNama());
                System.out.println("SEMESTER : " + m.getSemester());
                System.out.println("IPK : " + m.getIpk());
                System.out.println("=====");
            }
        }
    }
}
```

The screenshot shows the same `MahasiswaController.java` file with several additional methods added. These include `displayAllMahasiswa` (which reads all students from the DAO), `addMahasiswa` (which adds a new student to the DAO), `updateMahasiswa` (which updates an existing student in the DAO), `deleteMahasiswa` (which deletes a student from the DAO), and `closeConnection` (which closes the connection to the DAO). Each method includes a call to `displayMessage` to confirm the operation.

```
    }

    // READ ALL (Menampilkan semua mahasiswa)
    public void displayAllMahasiswa(){
        List<ModelMahasiswa> mahasiswaList = mahasiswaDAO.getAllMahasiswa();
        displayMahasiswaList(mahasiswaList);
    }

    public void addMahasiswa(String npm, String nama, int semester, float ipk){
        ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(0, npm, nama, semester, ipk);
        System.out.println("Controller Data: " + npm + nama + semester + ipk);
        System.out.println(mahasiswaBaru);
        mahasiswaDAO.addMahasiswa(mahasiswaBaru);
        displayMessage("Mahasiswa berhasil ditambahkan!");
    }

    public void updateMahasiswa(int id, String npm, String nama, int semester, float ipk){
        ModelMahasiswa mahasiswaBaru = new ModelMahasiswa(id, npm, nama, semester, ipk);
        mahasiswaDAO.updateMahasiswa(mahasiswaBaru);
        displayMessage("Mahasiswa berhasil diperbarui!");
    }

    public void deleteMahasiswa(int id){
        mahasiswaDAO.deleteMahasiswa(id);
        displayMessage("Mahasiswa Berhasil Dihapus!");
    }

    public void closeConnection() {
        mahasiswaDAO.closeConnection();
    }
}
```

The screenshot shows a Java code editor with the file `MahasiswaDAO.java` open. The code implements a DAO interface for managing student data. It includes methods for inserting, updating, deleting, and closing database connections. Error handling is provided using `SQLException`.

```
85     pstmt.setInt(5, mahasiswa.getId());
86     pstmt.executeUpdate();
87 } catch(SQLException e){
88     e.printStackTrace();
89 }
90 }

92 public void deleteMahasiswa(int id){
93     String sql = "DELETE FROM mahasiswa WHERE id = ?";
94     try{
95         PreparedStatement pstmt = connection.prepareStatement(sql);
96         pstmt.setInt(1, id);
97         pstmt.executeUpdate();
98     } catch(SQLException e){
99         e.printStackTrace();
100    }
101 }

103 // Method untuk menutup koneksi database
104 public void closeConnection() {
105     try {
106         if (connection != null) {
107             connection.close();
108         }
109     } catch (SQLException e) {
110         e.printStackTrace();
111     }
112 }
113
114 }
115
```

The screenshot shows a Java code editor with the file `ModelMahasiswa.java` open. This class represents a student model with properties for NPM, name, semester, and IPK. It provides getters and setters for these fields.

```
34     public String getNpm() {
35         return npm;
36     }
37
38     public void setNpm(String npm) {
39         this.npm = npm;
40     }
41
42     public String getNama() {
43         return nama;
44     }
45
46     public void setNama(String nama) {
47         this.nama = nama;
48     }
49
50     public int getSemester() {
51         return semester;
52     }
53
54     public void setSemester(int semester) {
55         this.semester = semester;
56     }
57
58     public float getIpk() {
59         return ipk;
60     }
61
62     public void setIpk(float ipk) {
63         this.ipk = ipk;
64     }
65
```

The screenshot shows the NetBeans IDE interface with the code editor open. The file is named 'MahasiswaView.java'. The code implements a menu system for managing students. It includes methods for updating student information, deleting students, checking database connections, closing connections, and exiting the program. The code uses a Scanner object to read user input and prints messages to System.out.

```
61     String namaBaru = scanner.next();
62     System.out.print("Masukkan Semester Baru: ");
63     int semesterBaru = scanner.nextInt();
64     System.out.print("Masukkan IPK Baru: ");
65     float ipkBaru = scanner.nextFloat();
66
67     mahasiswaController.updateMahasiswa(id, npmBaru, namaBaru, semesterBaru, ipkBaru);
68     break;
69
70 case 4:
71     System.out.print("Masukkan ID Mahasiswa: ");
72     int idHapus = scanner.nextInt();
73     mahasiswaController.deleteMahasiswa(idHapus);
74     break; // ditambahkan
75
76 case 5:
77     mahasiswaController.checkDatabaseConnection();
78     break;
79
80 case 6:
81     mahasiswaController.closeConnection();
82     System.out.println("Program selesai.");
83     scanner.close();
84     return;
85
86 default:
87     System.out.println("Input tidak valid");
88 }
89 }
90 }
91 }
```

The screenshot shows the NetBeans IDE interface with the Output window open. The title bar says 'Output - Run (MahasiswaView)'. The window displays the command-line output of the application's execution. It shows the Maven build process, including the relocation of MySQL Connector artifacts, and the application's menu system. The menu includes options for viewing all students, adding a student, updating a student, deleting a student, checking database connection, and exiting. The application successfully connects to the database and displays the menu options.

```
-----[ 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 Connector/J artifacts moved to reverse-DNS compliant Maven 2+ coordinates.
--- resources:3.3.1:resources (default-resources) @ pert3_rp12 ---
skip non existing resourceDirectory C:\Users\User\OneDrive\Documents\NetBeansProjects\pert3_rp12\src\main\resources
--- compiler:3.13.0:compile (default-compile) @ pert3_rp12 ---
Nothing to compile - all classes are up to date.

--- exec:3.1.0:exec (default-cli) @ pert3_rp12 ---
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:
```