

LAPORAN AKHIR PRAKTIKUM

Mata Praktikum : Rekayasa Perangkat Lunak 2
Kelas : 4IA28
Praktikum ke : 3 dan 4
Tanggal : 26 Oktober 2024
Materi : Konsep Model View Controller (MVC)
NPM : 50421118
Nama : ALGHANY SIHOMBING
Ketua Asisten :
Paraf Asisten :
Nama Asisten :
Jumlah Lembar : 20

**LABORATORIUM TEKNIK INFORMATIKA
UNIVERSITAS GUNADARMA
2024**

Nomor 1(Essay)

1. Jelaskan apa itu ORM (Object Relational Mapper)?

ORM (Object Relational Mapper) adalah sebuah teknik pemrograman yang memungkinkan pengembang untuk berinteraksi dengan database menggunakan objek-objek dalam kode, tanpa perlu menulis langsung query SQL. Dengan ORM, objek di dalam kode akan dihubungkan atau dipetakan ke tabel yang ada di database, sehingga operasi database seperti penyimpanan, pembaruan, dan penghapusan data dapat dilakukan langsung melalui objek tersebut.

2. Jelaskan kelebihan dari ORM

- I. **Meningkatkan Produktivitas Pengembang:** ORM memungkinkan pengembang menulis lebih sedikit kode SQL secara manual. Sebagai gantinya, mereka dapat bekerja dengan objek dalam bahasa pemrograman mereka, yang biasanya lebih intuitif.
- II. **Mengurangi Ketergantungan pada SQL:** Dengan ORM, pengembang dapat menulis kode untuk berbagai database tanpa mengubah banyak kode aplikasi. ORM menyediakan abstraksi dari perbedaan-perbedaan sintaks SQL antara berbagai sistem database.
- III. **Mengurangi Kemungkinan Kesalahan:** ORM memudahkan pengembang untuk menghindari kesalahan umum dalam penulisan SQL, seperti kesalahan sintaks atau kesalahan penanganan input, sehingga meningkatkan stabilitas aplikasi.
- IV. **Mendukung Maintenance yang Lebih Baik:** Perubahan struktur tabel atau skema di database dapat dilakukan dengan lebih mudah tanpa mengubah banyak kode aplikasi. ORM juga sering menyertakan fitur migrasi untuk mengelola perubahan pada struktur database.
- V. **Mempercepat Pengembangan Aplikasi:** Dengan ORM, pengembangan aplikasi menjadi lebih cepat karena penanganan CRUD (Create, Read, Update, Delete) bisa dilakukan lebih sederhana dan cepat.

Pertemuan 3

1. MahasiswaController.java

```
/
    Click
nbfs://nbhost/SystemFileSystem/Templates/Licenses/licensedefault.txt
to change this license
    Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
/
package me.alghany.mahasiswa.controller;

import java.util.List;
import me.alghany.mahasiswa.model.MahasiswaDao;
import me.alghany.mahasiswa.model.ModelMahasiswa;

/

    @author BNI
/
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("NAMA         : " + m.getNama());
                System.out.println("SEMESTER     : " +
m.getSemester());
                System.out.println("IPK          : " + m.getIpk());
                System.out.println("=====");
            }
        }
    }

    public void displayMessage(String message) {
        System.out.println(message);
    }

    public void checkDatabaseConnection() {
        boolean isConnected = mahasiswaDao.checkConnection();
        if (isConnected) {
            displayMessage("Koneksi ke db berhasil");
        } else {
            displayMessage("Koneksi DB Gagal");
        }
    }
}
```

```

    }

    // 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();
    }
}

```

2. MahasiswaDao.java

```
package me.agits.mahasiswa.model;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;

/

    @author BNI
/
public class MahasiswaDao {
    private Connection connection;

    public MahasiswaDao() {
        try{
            Class.forName("com.mysql.cj.jdbc.Driver");
            connection =
DriverManager.getConnection("jdbc:mysql://localhost:3306/mvc_db",
"root", "");
        } catch (Exception e){
            e.printStackTrace();
        }
    }

    public boolean checkConnection(){
        try{
            if(connection != null && !connection.isClosed()){
                return true; //koneksi berhasil
            }
        } catch (SQLException e){
            e.printStackTrace();
        }
        return false;
    }

    public void addMahasiswa(ModelMahasiswa mahasiswa){
        String sql = "INSERT INTO mahasiswa (npm, nama, semester, ipk)
VALUES (?, ?, ?, ?)";
        try{
```

```

        PreparedStatement pstmt =
connection.prepareStatement(sql);
        pstmt.setString(1, mahasiswa.getNpm());
        pstmt.setString(2, mahasiswa.getNama());
        pstmt.setInt(3, mahasiswa.getSemester());
        pstmt.setFloat(4, mahasiswa.getIpk());
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public List<ModelMahasiswa> getAllMahasiswa() {
    List<ModelMahasiswa> mahasiswaList = new ArrayList<>();
    String sql = "SELECT * FROM mahasiswa";
    try {
        Statement stmt = connection.createStatement();
        ResultSet rs = stmt.executeQuery(sql);
        while (rs.next()) {
            mahasiswaList.add(new ModelMahasiswa(
                rs.getInt("id"),
                rs.getString("npm"),
                rs.getString("nama"),
                rs.getInt("semester"),
                rs.getFloat("ipk")
            ));
        }
    } catch (SQLException e) {
        e.printStackTrace();
    }
    return mahasiswaList;
}

public void updateMahasiswa(ModelMahasiswa mahasiswa) {
    String sql = "UPDATE mahasiswa SET npm = ?, nama = ?, semester
= ?, ipk = ? WHERE id = ?";
    try {
        PreparedStatement pstmt =
connection.prepareStatement(sql);
        pstmt.setString(1, mahasiswa.getNpm());
        pstmt.setString(2, mahasiswa.getNama());
        pstmt.setInt(3, mahasiswa.getSemester());
        pstmt.setFloat(4, mahasiswa.getIpk());
        pstmt.setInt(5, mahasiswa.getId());
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

public void deleteMahasiswa(int id) {
    String sql = "DELETE FROM mahasiswa WHERE id = ?";
    try {
        PreparedStatement pstmt =
connection.prepareStatement(sql);
        pstmt.setInt(1, id);
        pstmt.executeUpdate();
    } catch (SQLException e) {
        e.printStackTrace();
    }
}

// Method untuk menutup koneksi database

```

```

    public void closeConnection() {
        try {
            if (connection != null) {
                connection.close();
            }
        } catch (SQLException e) {
            e.printStackTrace();
        }
    }
}

```

3. ModelMahasiswa.java

```

/
Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
/
package me.alghany.mahasiswa.model;

/

@author BNI
/
public class ModelMahasiswa {
    private int id;
    private String nama;
    private String npm;
    private int semester;
    private float ipk;

    public ModelMahasiswa(int id, String npm, String nama, int
semester, float ipk) {
        this.id = id;
    }
}

```

```
        this.npm = npm;
        this.nama = nama;
        this.semester = semester;
        this.ipk = ipk;
    }

    public int getId() {
        return id;
    }

    public void setId(int id) {
        this.id = id;
    }

    public String getNama() {
        return nama;
    }

    public void setNama(String nama) {
        this.nama = nama;
    }

    public String getNpm() {
        return npm;
    }

    public void setNpm(String npm) {
        this.npm = npm;
    }

    public int getSemester() {
        return semester;
    }

    public void setSemester(int semester) {
        this.semester = semester;
    }

    public float getIpk() {
        return ipk;
    }

    public void setIpk(float ipk) {
        this.ipk = ipk;
    }
}
```


4. MahasiswaView.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
 default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java
 to edit this template
 */
package me.alghany.mahasiswa.view;

import java.util.Scanner;
import me.alghany.mahasiswa.controller.MahasiswaController;
import me.alghany.mahasiswa.model.MahasiswaDao;

/**
 *
 * @author BNI
 */
public class MahasiswaView {
    public static void main(String[] args){
        MahasiswaDao mahasiswaDao = new MahasiswaDao();
        MahasiswaController mahasiswaController = new
MahasiswaController(mahasiswaDao);

        Scanner scanner = new Scanner(System.in);
        int pilihan;

        while(true){
            System.out.println("Menu:");
            System.out.println("1. Tampilkan Semua Mahasiswa");
            System.out.println("2. Tambah Mahasiswa");
            System.out.println("3. Update Mahasiswa");
            System.out.println("4. Hapus Mahasiswa");
            System.out.println("5. Cek Koneksi Database");
            System.out.println("6. Keluar");
            System.out.print("PILIH OPSI: ");
```

```

        pilihan = scanner.nextInt();
        scanner.nextLine();

        switch (pilihan){
            case 1:
                mahasiswaController.displayAllMahasiswa();
                break;

            case 2:
                // tambah mhs
                System.out.println("Masukkan NPM: ");
                String npm = scanner.next();
                System.out.println("Masukkan Nama: ");
                String nama = scanner.next();
                System.out.println("Masukkan Semester: ");
                int semester = scanner.nextInt();
                System.out.println("Masukkan IPK: ");
                float ipk = scanner.nextFloat();
                System.out.println(npm + nama + semester + ipk);

                mahasiswaController.addMahasiswa(npm, nama,
semester, ipk);

                break;

            case 3:
                System.out.print("Masukkan ID mahasiswa: ");
                int id = scanner.nextInt();
                scanner.nextLine();

                System.out.println("Masukkan NPM: ");
                String npmBaru = scanner.next();
                System.out.println("Masukkan Nama: ");
                String namaBaru = scanner.next();
                System.out.println("Masukkan Semester: ");
                int semesterBaru = scanner.nextInt();
                System.out.println("Masukkan IPK: ");
                float ipkBaru = scanner.nextFloat();

                mahasiswaController.updateMahasiswa(id, npmBaru,
namaBaru, semesterBaru, ipkBaru);
                break;

            case 4:
                System.out.print("Masukkan ID Mahasiswa: ");
                int idHapus = scanner.nextInt();
                mahasiswaController.deleteMahasiswa(idHapus);

            case 5:
                mahasiswaController.checkDatabaseConnection();
                break;

            case 6:
                // Keluar
                mahasiswaController.closeConnection();
                System.out.println("Program selesai.");
                return;

            default:

```

```
        System.out.println("Input Tidak valid");  
    }  
    }  
}
```

5. Hasil Output

Source History

Output - Run (MahasiswaView) x

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

PILIH OPSI: 2

Masukkan NPM: 50421899

Masukkan Nama: MuhammadAgitaAmanatullah

Masukkan Semester: 7

Masukkan IPK: 4

50421899MuhammadAgitaAmanatullah74.0

Controller Data: 50421899MuhammadAgitaAmanatullah74.0

```

me.agits.mahasiswa.model.Mahasiswa84b4c42d3
com.mysql.cj.jdbc.exceptions.MySQLDataTruncation: Data truncation: Data too long for column 'nama' at row 1
at com.mysql.cj.jdbc.exceptions.SQLExceptionsMapping.translateException(SQLExceptionsMapping.java:104)
at com.mysql.cj.jdbc.ClientPreparedStatement.executeInternal(ClientPreparedStatement.java:916)
at com.mysql.cj.jdbc.ClientPreparedStatement.executeUpdateInternal(ClientPreparedStatement.java:1061)
at com.mysql.cj.jdbc.ClientPreparedStatement.executeUpdateInternal(ClientPreparedStatement.java:1069)
at com.mysql.cj.jdbc.ClientPreparedStatement.executeLargeUpdate(ClientPreparedStatement.java:1320)
at com.mysql.cj.jdbc.ClientPreparedStatement.executeUpdate(ClientPreparedStatement.java:894)
at me.agits.mahasiswa.model.MahasiswaDao.addMahasiswa(MahasiswaDao.java:48)
at me.agits.mahasiswa.controller.MahasiswaController.addMahasiswa(MahasiswaController.java:65)
at me.agits.mahasiswa.view.MahasiswaView.main(MahasiswaView.java:52)

```

Mahasiswa berhasil ditambahkan!

Menu:

langkah 1

1. MahasiswaController.java

```
/
Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
default.txt to change this license
Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java to
edit this template
/
package com.mahasiswa.controller;

import com.mahasiswa.model.HibernateUtil;
import com.mahasiswa.model.ModelMahasiswa;
import java.util.List;
import org.hibernate.Session;
import org.hibernate.Transaction;
import org.hibernate.query.Query;

public class MahasiswaController {

    public void addMhs (ModelMahasiswa mhs) {
        Transaction trx = null;

        try (Session session =
HibernateUtil.getSessionFactory().openSession()) {
            trx = session.beginTransaction();
            session.save(mhs);
            trx.commit();
        } catch (Exception e) {
            if (trx != null) {
                trx.rollback();
            }
            e.printStackTrace();
        }
    }

    public void updateMhs (ModelMahasiswa mhs) {
        Transaction trx = null;

        try (Session session =
HibernateUtil.getSessionFactory().openSession()) {
            trx = session.beginTransaction();
            session.update(mhs);
            trx.commit();
        } catch (Exception e) {
            if (trx != null) {
                trx.rollback();
            }
            e.printStackTrace();
        }
    }

    public void deleteMhs (int id) {
        Transaction trx = null;

        try (Session session =
HibernateUtil.getSessionFactory().openSession()) {
```

```

        trx = session.beginTransaction();
        ModelMahasiswa mhs = session.get(ModelMahasiswa.class,
id);
        if(mhs != null){
            session.delete(mhs);
            System.out.println("Berhasil hapus");
        }
        trx.commit();
    } catch (Exception e){
        if (trx != null){
            trx.rollback();
        }
        e.printStackTrace();
    }

}

public List<ModelMahasiswa> getAllMahasiswa() {
    Transaction trx = null;
    List<ModelMahasiswa> listMhs = null;

    try (Session session =
HibernateUtil.getSessionFactory().openSession()){
        trx = session.beginTransaction();
        // Using HQL (Hibernate Query Language) to fetch all
records
        Query<ModelMahasiswa> query = session.createQuery("from
ModelMahasiswa", ModelMahasiswa.class);
        listMhs = query.list(); // Fetch all results

        trx.commit(); // Commit transaction
    } catch (Exception e) {
        if (trx != null) {
            trx.rollback(); // Rollback transaction in case of
error
        }
        e.printStackTrace();
    }

    // Return the fetched list
    return listMhs;
}
}

```

2.HibernateUtil.java

```
package com.mahasiswa.model;

import org.hibernate.Session;
import org.hibernate.SessionFactory;
import org.hibernate.cfg.Configuration;

public class HibernateUtil {
    private static SessionFactory sessionFactory;

    static {
        try {
            // Create the SessionFactory from hibernate.cfg.xml
            sessionFactory = new
Configuration().configure().buildSessionFactory();
        } catch (Throwable ex) {
            // Make sure you log the exception, as it might be
            swallowed
            System.err.println("Initial SessionFactory creation
            failed." + ex);
            throw new ExceptionInInitializerError(ex);
        }
    }

    public static SessionFactory getSessionFactory() {
        return sessionFactory;
    }

    public static void testConnection() {
        try (Session session = sessionFactory.openSession()) {
            System.out.println("Connection to the database was
            successful!");
        } catch (Exception e) {
            System.err.println("Failed to connect to the database.");
            e.printStackTrace();
        }
    }
}
```

2. ModelMahasiswa.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
 default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java
 to edit this template
 */
package com.mahasiswa.model;

import jakarta.persistence.*;

/**
 *
 * @author BNI
 */
@Entity
@Table(name = "mahasiswa")
public class ModelMahasiswa {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Column(name = "id")
    private int id;

    @Column(name = "npm", nullable = false, length = 8)
    private String npm;

    @Column(name = "nama", nullable = false, length = 50)
    private String nama;
```



```

@Column(name = "semester")
private int semester;

@Column(name = "ipk")
private float ipk;

public ModelMahasiswa(){

}

public ModelMahasiswa(int id, String npm, String nama, int
semester, float ipk){
    this.id = id;
    this.npm = npm;
    this.nama = nama;
    this.semester = semester;
    this.ipk = ipk;
}

public int getId() {
    return id;
}

public void setId(int id) {
    this.id = id;
}

public String getNpm() {
    return npm;
}

public void setNpm(String npm) {
    this.npm = npm;
}

public String getNama() {
    return nama;
}

public void setNama(String nama) {
    this.nama = nama;
}

public int getSemester() {
    return semester;
}

public void setSemester(int semester) {
    this.semester = semester;
}

public float getIpk() {
    return ipk;
}

public void setIpk(float ipk) {
    this.ipk = ipk;
}

}

```

3. ModelTabelMahasiswa.java

```
/*
 * Click nbfs://nbhost/SystemFileSystem/Templates/Licenses/license-
 * default.txt to change this license
 * Click nbfs://nbhost/SystemFileSystem/Templates/Classes/Class.java
 * to edit this template
 */
package com.mahasiswa.model;
import javax.swing.table.AbstractTableModel;
import java.util.List;

/**
 *
 * @author BNI
 */
public class ModelTabelMahasiswa extends AbstractTableModel{
    private List<ModelMahasiswa> mahasiswaList;
    private String[] columnNames = {"ID", "NPM", "Nama", "Semester",
    "IPK"};

    public ModelTabelMahasiswa(List<ModelMahasiswa> mahasiswaList) {
        this.mahasiswaList = mahasiswaList;
    }

    @Override
    public int getRowCount() {
        return mahasiswaList.size(); // Jumlah baris sesuai dengan
        jumlah data mahasiswa
    }

    @Override
    public int getColumnCount() {
        return columnNames.length; // Jumlah kolom sesuai dengan
        jumlah elemen dalam columnNames
    }

    @Override
    public Object getValueAt(int rowIndex, int columnIndex) {
        ModelMahasiswa mahasiswa = mahasiswaList.get(rowIndex);
        switch (columnIndex) {
            case 0:
                return mahasiswa.getId();
            case 1:
                return mahasiswa.getNpm();
            case 2:
```

```

        return mahasiswa.getNama();
    case 3:
        return mahasiswa.getSemester();
    case 4:
        return mahasiswa.getIpk();
    default:
        return null;
    }
}

@Override
public String getColumnName(int column) {
    return columnNames[column]; // Mengatur nama kolom
}

@Override
public boolean isCellEditable(int rowIndex, int columnIndex) {
    return false; // Semua sel tidak dapat diedit
}

// Method untuk menambahkan atau memodifikasi data, jika
dibutuhkan
public void setMahasiswaList(List<ModelMahasiswa> mahasiswaList) {
    this.mahasiswaList = mahasiswaList;
    fireTableDataChanged(); // Memberitahu JTable bahwa data telah
    berubah
}
}

```

Output :

The screenshot displays a Java Swing application for managing student data. The main window contains four text input fields for NPM (50421899), NAMA (Muhammad Agits A), SEMESTER (7), and IPK (4.00). Below these fields are three buttons: 'Simpan', 'Refresh', and 'Buang'. A JTable is positioned below the buttons, showing a single row of data with columns ID, NPM, Nama, Semester, and IPK. The first row contains the values 1, 50421899, Muhammad..., 7, and 4.0. Two 'Hapus Mahasiswa' dialog boxes are shown. The first dialog box is empty, and the second dialog box has the value '1' entered in the 'Masukkan ID yang ingin dihapus:' field.

ID	NPM	Nama	Semester	IPK
1	50421899	Muhammad...	7	4.0

