



TUGAS & EVALUASI

Soal Tugas & Evaluasi

Jelaskan alur kerja arsitektur MVC dan bagaimana interaksi antara User, View, Controller, dan Model dalam mengelola permintaan pengguna!

Jawaban

Arsitektur MVC (Model-View-Controller) memisahkan logika aplikasi menjadi tiga komponen utama: **Model** mengelola data dan logika, **View** bertanggung jawab untuk menampilkan antarmuka pengguna, dan **Controller** mengatur interaksi antara View dan Model. Saat pengguna melakukan suatu aksi misal klik suatu tombol, Controller menerima permintaan tersebut, memprosesnya (jika perlu), dan menginstruksikan Model untuk memanipulasi data. Model memberi tahu View untuk memperbarui tampilan sehingga mencerminkan perubahan data, menciptakan alur kerja yang terorganisir dan terpisah.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

Mengapa di dalam MVC, input/output tidak diperbolehkan berada di dalam Controller?

Jawaban

Untuk menjaga pemisahan tanggung jawab dan Controller tetap berfokus pada pengaturan logika aplikasi tanpa terikat pada detail implementasi tampilan.



TUGAS & EVALUASI

Soal Tugas & Evaluasi

Pemilik toko ingin mengelola inventaris barang dan catatan penjualan secara efisien. Mereka memerlukan sebuah sistem yang dapat mencatat informasi tentang barang yang ada di toko dan transaksi penjualan yang terjadi.

Entity yang harus dibuat:

- | | |
|---------------|-------------------------------|
| - Barang | - Tugas: |
| • idBarang | • Terapkan arsitektur MVC |
| • nama | untuk mengelola data barang |
| • harga | dan penjualan. |
| • stock | • Buat Model, Controller, dan |
| • Penjualan | View yang mendukung |
| • idPenjualan | Daftar barang: |
| • idBarang | • Penambahan dan pembaruan |
| • kuantitas | barang |
| • tanggal | • Pencatatan transaksi |
| • totalHarga | penjualan |
| | • Daftar transaksi penjualan |



TUGAS & EVALUASI

Source Code

```
#modelbarang
package model;
public class Barang {
    private int id;
    private String nama;
    private int stok;
    private double harga;
    public Barang(int id, String nama, int stok, double harga) {
        this.id = id;
        this.nama = nama;
        this.stok = stok;
        this.harga = harga;
    }
    public int getId() {
        return id;
    }

    public String getName() {
        return nama;
    }

    public int getStok() {
        return stok;
    }

    public void setStok(int stok) {
        this.stok = stok;
    }

    public double getHarga() {
        return harga;
    }
    public void setHarga(double harga) {
        this.harga = harga;
    }
}
```



TUGAS & EVALUASI

```
#modeltransaksi
package model;

import java.time.LocalDate;

public class Transaksi {
    private int id;
    private LocalDate tanggal;
    private double totalHarga;

    public Transaksi(int id, LocalDate tanggal, double
totalHarga) {
        this.id = id;
        this.tanggal = tanggal;
        this.totalHarga = totalHarga;
    }

    public int getId() {
        return id;
    }

    public LocalDate getTanggal() {
        return tanggal;
    }
    public double getTotalHarga() {
        return totalHarga;
    }
}

#modeldatabase
package model;
import java.util.ArrayList;
import java.util.List;
public class Database {
    public static List<Barang> barangList = new ArrayList<>();
    public static List<Transaksi> transaksiList = new
ArrayList<>();
}
```



TUGAS & EVALUASI

```
#controllerbarang
package controller;

import model.Barang;

import java.util.ArrayList;
import java.util.List;

public class BarangController {
    private List<Barang> barangList = new ArrayList<>();

    public void tambahBarang(Barang barang) {
        barangList.add(barang);
    }

    public void updateBarang(int id, int stokBaru) {
        for (Barang barang : barangList) {
            if (barang.getId() == id) {
                barang.setStok(stokBaru);
                return;
            }
        }
        throw new IllegalArgumentException("Barang dengan ID " +
id + " tidak ditemukan.");
    }

    public List<Barang> getBarangList() {
        return barangList;
    }
}

#controllertransaksi
package controller;

import model.Barang;
import model.Transaksi;
```



TUGAS & EVALUASI

```
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

public class TransaksiController {
    private List<Transaksi> transaksiList = new ArrayList<>();
    private BarangController barangController;

    public TransaksiController(BarangController
barangController) {
        this.barangController = barangController;
    }

    public void catatTransaksi(int idBarang, int jumlah) {
        List<Barang> barangList =
barangController.getBarangList();

        for (Barang barang : barangList) {
            if (barang.getId() == idBarang) {
                if (barang.getStok() < jumlah) {
                    throw new IllegalArgumentException("Stok
tidak mencukupi!");
                }

                double totalHarga = jumlah * barang.getHarga();
                barang.setStok(barang.getStok() - jumlah);

                Transaksi transaksi = new Transaksi(idBarang,
LocalDate.now(), totalHarga);
                transaksiList.add(transaksi);
                return;
            }
        }

        throw new IllegalArgumentException("Barang dengan ID " +
idBarang + " tidak ditemukan.");
    }

    public List<Transaksi> getTransaksiList() {
```



TUGAS & EVALUASI

```
        return transaksiList;
    }
}

#viewbarang
package view;

import model.Barang;

import java.util.List;

public class BarangView {
    public void showBarang(List<Barang> barangList) {
        System.out.println("Daftar Barang:");
        for (Barang barang : barangList) {
            System.out.println("ID: " + barang.getId() + ",
Nama: " + barang.getNama() +
            ", Stok: " + barang.getStok() + ", Harga: "
+ barang.getHarga());
        }
    }

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

#viewtransaksi
package view;

import model.Transaksi;

import java.util.List;

public class TransaksiView {
    public void showTransaksi(List<Transaksi> transaksiList) {
        System.out.println("Daftar Transaksi:");
    }
}
```




TUGAS & EVALUASI

```
        for (Transaksi transaksi : transaksiList) {
            System.out.println("ID: " + transaksi.getId() + ",
Tanggal: " + transaksi.getTanggal() +
                ", Total Harga: " +
transaksi.getTotalHarga());
        }
    }

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

#GUIview
// package Pertemuan6_07705.Tugas_Praktikum;
import javax.swing.*;
import controller.BarangController;
import controller.TransaksiController;
import view.GUI;

public class Main {
    public static void main(String[] args) {
        SwingUtilities.invokeLater(() -> {
            BarangController barangController = new
BarangController();
            TransaksiController transaksiController = new
TransaksiController(barangController);
            new GUI(barangController,
transaksiController).setVisible(true);
        });
    }
}
```



TUGAS & EVALUASI

Output

Pengelolaan Barang & Penjualan

Tambah Barang

Update Stok

Catat Transaksi

Daftar Barang

Daftar Transaksi

Tambah Barang

ID Barang: 1

Nama Barang: Jeroek

Stok Barang: 40

Harga Barang: 3400

OK Cancel

Update Stok

ID Barang: 1

Stok Baru: 45

OK Cancel

Penjualan

ID Barang: 1


Jumlah Penjualan: 12

OK Cancel




TUGAS & EVALUASI

Daftar Barang

**Daftar Barang:**
ID: 1, Nama: Jeroek, Stok: 33, Harga: 3400.0

OK

Daftar Transaksi

**Daftar Transaksi:**
ID: 1, Tanggal: 2024-12-23, Total Harga: 40800.0

OK