

## **TUGAS INDIVIDU**

# **IDENTIFIKASI PERMASALAHAN ORGANISASI DAN SOLUSI SISTEM INFORMASI BERBASIS JAVA DENGAN PENDEKATAN PEMROGRAMAN BERORIENTASI OBJEK (PBO) MENGGUNAKAN DATABASE MYSQL**

*Laporan ini disusun untuk memenuhi Tugas Besar mata kuliah Pemrograman Berorientasi Objek*



**Dosen Pengampu :**

Jefril Rahmadoni, M.Kom

**Disusun Oleh:**

Nama : Aqila Higenea Taufik

NIM : 2411522003

**DEPARTEMEN SISTEM INFORMASI  
FAKULTAS TEKNOLOGI INFORMASI  
UNIVERSITAS ANDALAS**

**2025**

1. Carilah dan jelaskan sebuah permasalahan disebuah organisasi dan berikan solusi dengan pendekatan sistem informasi dengan penerapan Pemrograman Berorientasi Objek

- a. Gambaran Organisasi

Organisasi yang dipilih pada tugas besar ini adalah Klinik, yaitu fasilitas pelayanan kesehatan tingkat pertama yang melayani pemeriksaan pasien umum, pengkoordinasian dokter, pendaftaran pasien, stok obat, serta pencatatan data harian. Dalam kegiatan operasionalnya, klinik melayani banyak pasien, koordinasi dengan dokter, dan pengecekan stok obat setiap hari dengan data yang harus dicatat secara akurat dan rapi.

- b. Permasalahan yang Dihadapi Klinik

Berdasarkan pengamatan pada proses bisnis klinik secara umum, terdapat beberapa permasalahan utama, yaitu:

Pertama, pengelolaan data pasien, dokter, kunjungan, dan obat masih dilakukan secara manual, seperti pencatatan di buku atau file terpisah. Hal ini menyebabkan data pasien, dokter, kunjungan, dan obat mudah hilang, rusak, atau sulit dicari kembali ketika dibutuhkan.

Kedua, proses pendaftaran pasien, dokter, kunjungan, dan obat tidak terstruktur, sehingga sering terjadi kesalahan penulisan identitas pasien seperti nama, nomor telepon, atau tanggal lahir. Selain itu, pencatatan tanggal pendaftaran pasien sering tidak konsisten.

Ketiga, tidak tersedianya rekap data dan laporan sederhana, misalnya jumlah pasien yang mendaftar dalam satu hari, siapa dokter yang bekerja, dan berapa sisa stok obat. Hal ini menyulitkan pihak klinik dalam melakukan evaluasi pelayanan dan pengambilan keputusan.

Keempat, waktu pelayanan menjadi lebih lama, karena petugas harus mencari data pasien, dokter, kunjungan, dan obat secara manual, terutama untuk pasien lama yang sudah pernah berobat sebelumnya.

- c. Dampak Permasalahan

Permasalahan tersebut berdampak langsung pada:

- Menurunnya efisiensi kerja petugas klinik
- Tingginya risiko kesalahan data pasien
- Sulitnya pembuatan laporan harian
- Kurangnya profesionalitas pelayanan klinik

Jika tidak ditangani, permasalahan ini dapat mengurangi kualitas pelayanan kesehatan yang diberikan kepada pasien.

d. Solusi yang Diusulkan

Untuk mengatasi permasalahan tersebut, solusi yang diusulkan adalah membangun Sistem Informasi Klinik berbasis Java dengan pendekatan Pemrograman Berorientasi Objek (PBO) dan menggunakan database MySQL sebagai penyimpanan data.

Sistem ini dirancang untuk:

- Mengelola data pasien, dokter, kunjungan, dan obat secara terkomputerisasi
- Menyimpan data pasien, dokter, kunjungan, dan obat ke dalam database
- Menyediakan fitur tambah, tampil, ubah, dan hapus data pasien, dokter, kunjungan, dan obat (CRUD)
- Menampilkan informasi dan laporan sederhana secara otomatis

e. Pendekatan Pemrograman Berorientasi Objek

Dalam sistem informasi klinik ini, pendekatan Pemrograman Berorientasi Objek (PBO) diterapkan dengan cara:

- Class dan Object digunakan untuk merepresentasikan entitas nyata seperti Pasien, dokter, kunjungan, dan obat.
- Inheritance digunakan untuk menurunkan class Pasien dan dokter dari class induk Person.
- Interface digunakan untuk mendefinisikan operasi CRUD data pasien, dokter, kunjungan, dan obat.
- Encapsulation diterapkan dengan penggunaan atribut private dan method getter-setter.
- Exception Handling digunakan untuk menangani kesalahan input dan koneksi database.
- Collection Framework digunakan untuk menampung data pasien, dokter, kunjungan, dan obat hasil query.
- JDBC (Java Database Connectivity) digunakan untuk menghubungkan aplikasi dengan database MySQL.

Dengan penerapan konsep PBO tersebut, sistem menjadi lebih terstruktur, mudah dikembangkan, dan sesuai dengan kebutuhan organisasi klinik.

#### f. Kesimpulan

Dengan adanya Sistem Informasi Klinik berbasis Java dan MySQL, permasalahan pencatatan manual dapat diatasi. Sistem ini membantu meningkatkan efisiensi kerja, mengurangi kesalahan data, serta mempermudah pengelolaan dan pencarian data pasien, dokter, kunjungan, dan obat. Selain itu, penerapan Pemrograman Berorientasi Objek menjadikan sistem lebih terorganisir dan memenuhi standar pembelajaran pada mata kuliah Pemrograman Berorientasi Objek.

2. Buatlah sebuah program dari solusi yang diberikan pada poin 1 yang didalam program tersebut terdapat materi-materi pembelajaran, yaitu:
- Terdiri dari beberapa class, objek, dan constructor
  - Memiliki interface dan implementasi dari interface tersebut
  - Memiliki inheritance (super class dan sub class)
  - Terdapat satu atau lebih perulangan, percabangan, dan perhitungan matematika
  - Terdapat beberapa manipulasi method String dan Date
  - Terdapat satu atau lebih exception handling
  - Terdapat satu atau lebih collection framework
  - Menggunakan JDBC dan terdapat fungsi create, read, update, dan delete (CRUD)

Coding:

### 1. Package database

#### a. Class DatabaseConnection.java

```
package database;

import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.SQLException;

public class DatabaseConnection {

    private static final String URL =
        "jdbc:mysql://localhost:3306/db_klinik";
    private static final String USER = "root";
    private static final String PASS = "";

    public static Connection getConnection() {
        try {
            return DriverManager.getConnection(URL, USER, PASS);
        } catch (SQLException e) {
            System.out.println("Koneksi database gagal!");
            return null;
        }
    }
}
```

### 2. package model

#### a. Class Person.java

```
package model;

// Superclass untuk Pasien dan Dokter
public class Person {
```

```

protected String id;
protected String nama;
protected String email;
protected String noHp;

// Constructor
public Person(String id, String nama, String email, String noHp) {
    this.id = id;
    this.nama = nama;
    this.email = email;
    this.noHp = noHp;
}

// Getter
public String getId() {
    return id;
}

public String getNama() {
    return nama;
}
}

```

#### b. Class Pasien.java

```

package model;

import java.time.LocalDate;

public class Pasien extends Person {
    public String alamat;
    public LocalDate tanggalLahir;
    public String jenisKelamin;
    public LocalDate tanggalDaftar;

    public Pasien(String id, String nama, String email, String noHp,
        String alamat, LocalDate tanggalLahir,
        String jenisKelamin, LocalDate tanggalDaftar) {

        super(id, nama, email, noHp);
        this.alamat = alamat;
        this.tanggalLahir = tanggalLahir;
        this.jenisKelamin = jenisKelamin;
        this.tanggalDaftar = tanggalDaftar;
    }
}

```

#### c. Class dokter.java

```

package model;

```

```

public class Dokter extends Person {
    public String spesialisasi;
    public double biayaKonsultasi;

    public Dokter(String id, String nama, String email, String noHp,
        String spesialisasi, double biayaKonsultasi) {

        super(id, nama, email, noHp);
        this.spesialisasi = spesialisasi;
        this.biayaKonsultasi = biayaKonsultasi;
    }

    public double getBiayaKonsultasi() {
        return biayaKonsultasi;
    }
}

```

d. Class Kunjungan.java

```

package model;

import java.time.LocalDate;

public class Kunjungan {
    public int idKunjungan;
    public int idPasien;
    public int idDokter;
    public LocalDate tanggalKunjungan;
    public String keluhan;
    public String diagnosa;
    public String resep;
    public double biayaTotal;
    public String status;

    public Kunjungan(int idKunjungan, int idPasien, int idDokter,
        LocalDate tanggalKunjungan, String keluhan,
        String diagnosa, String resep,
        double biayaTotal, String status) {

        this.idKunjungan = idKunjungan;
        this.idPasien = idPasien;
        this.idDokter = idDokter;
        this.tanggalKunjungan = tanggalKunjungan;
        this.keluhan = keluhan;
        this.diagnosa = diagnosa;
        this.resep = resep;
        this.biayaTotal = biayaTotal;
        this.status = status;
    }
}

```

e. Class Obat.java

```
package model;

public class Obat {
    public String idObat;
    public String namaObat;
    public String kategori;
    public int stok;
    public double harga;

    public Obat(String idObat, String namaObat, String kategori, int stok,
double harga) {
        this.idObat = idObat;
        this.namaObat = namaObat;
        this.kategori = kategori;
        this.stok = stok;
        this.harga = harga;
    }
}
```

3. package service

a. Class CrudService.java

```
package service;

// Interface CRUD generic
public interface CrudService {
    void tambah();
    void tampil();
    void update();
    void hapus();
}
```

b. Class BaseService.java

```
package service;

import java.sql.Connection;
import java.sql.SQLException;
import java.util.Scanner;

import database.DatabaseConnection;

public abstract class BaseService {
    protected Scanner input = new Scanner(System.in);

    protected Connection getConnection() throws SQLException {
        return DatabaseConnection.getConnection();
    }
}
```



```
}
```

c. Class PasienService.java

```
package service;

import database.DatabaseConnection;
import java.sql.*;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.List;

public class PasienService extends BaseService implements CrudService
{

    @Override
    public void tambah() {
        try {
            Connection conn = DatabaseConnection.getConnection();

            System.out.print("\nID Pasien: ");
            String idPasien = input.nextLine().toUpperCase();

            System.out.print("Nama Pasien: ");
            String nama = input.nextLine();

            System.out.print("Email: ");
            String email = input.nextLine();

            System.out.print("No HP: ");
            String noHp = input.nextLine();

            System.out.print("Alamat: ");
            String alamat = input.nextLine();

            System.out.print("Tanggal Lahir (yyyy-MM-dd): ");
            LocalDate tglLahir = LocalDate.parse(input.nextLine());

            System.out.print("Jenis Kelamin: ");
            String jk = input.nextLine();

            LocalDate tglDaftar = LocalDate.now();

            String sql = "INSERT INTO pasien " +
                        "(id_pasien, nama_pasien, email, no_hp, alamat, " +
                        tanggal_lahir, jenis_kelamin, tanggal_daftar) " +
                        "VALUES (?,?,?,?,?,?,?)";

            PreparedStatement ps = conn.prepareStatement(sql);
            ps.setString(1, idPasien);
            ps.setString(2, nama);
```

```

        ps.setString(3, email);
        ps.setString(4, noHp);
        ps.setString(5, alamat);
        ps.setDate(6, Date.valueOf(tglLahir));
        ps.setString(7, jk);
        ps.setDate(8, Date.valueOf(tglDaftar));

        ps.executeUpdate();
        System.out.println("\nData pasien berhasil ditambahkan!");

    } catch (Exception e) {
        System.out.println("\nGagal menambahkan pasien!");
        e.printStackTrace();
    }
}

@Override
public void tampil() {
    try {
        Connection conn = DatabaseConnection.getConnection();
        ResultSet rs = conn.createStatement().executeQuery("SELECT *
FROM pasien");

        List<String> listIdPasien = new ArrayList<>();

        System.out.printf(
            "\n%-10s | %-10s | %-15s | %-13s | %-15s | %-10s | %-10s\n",
            "ID Pasien", "Nama", "Email", "No HP", "Alamat", "Tgl Lahir",
            "Jenis Kelamin"
        );
        System.out.println("-----");
        -----");

        while (rs.next()) {
            String idPasien = rs.getString("id_pasien");
            listIdPasien.add(idPasien);

            System.out.printf(
                "%-10s | %-10s | %-15s | %-13s | %-15s | %-10s | %-10s\n",
                idPasien,
                rs.getString("nama_pasien").toUpperCase(),
                rs.getString("email"),
                rs.getString("no_hp"),
                rs.getString("alamat"),
                rs.getString("tanggal_lahir"),
                rs.getString("jenis_kelamin")
            );
        }

        System.out.println("\nTotal data pasien : " + listIdPasien.size());
    }
}

```

```

    } catch (Exception e) {
        System.out.println("\nGagal tampil pasien!");
        e.printStackTrace();
    }
}

@Override
public void update() {
    try {
        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nMasukkan ID Pasien yang akan diupdate: ");
        String id = input.nextLine();

        System.out.print("ID Pasien Baru: ");
        String idPasien = input.nextLine();

        System.out.print("Nama Baru: ");
        String nama = input.nextLine();

        System.out.print("No HP Baru: ");
        String noHp = input.nextLine();

        System.out.print("Email Baru: ");
        String email = input.nextLine();

        System.out.print("Alamat Baru: ");
        String alamat = input.nextLine();

        System.out.print("Tanggal Lahir Baru: ");
        String tglLahir = input.nextLine();

        System.out.print("Jenis Kelamin Baru: ");
        String jenisKelamin = input.nextLine();

        String sql = "UPDATE pasien SET id_pasien=?, nama_pasien=?,
no_hp=?, email=?, alamat=?, tanggal_lahir=?, jenis_kelamin=? WHERE
id_pasien=?";
        PreparedStatement ps = conn.prepareStatement(sql);

        ps.setString(1, idPasien);
        ps.setString(2, nama);
        ps.setString(3, noHp);
        ps.setString(4, email);
        ps.setString(5, alamat);
        ps.setString(6, tglLahir);
        ps.setString(7, jenisKelamin);
        ps.setString(8, id);
    }
}

```

```

        int hasil = ps.executeUpdate();

        if (hasil > 0) {
            System.out.println("\nData pasien berhasil diupdate!");
        } else {
            System.out.println("\nID pasien tidak ditemukan");
        }

    } catch (Exception e) {
        System.out.println("\nGagal update pasien!");
        e.printStackTrace();
    }
}

@Override
public void hapus() {
    try {
        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nID Pasien yang akan dihapus: ");
        String id = input.nextLine();

        String sql = "DELETE FROM pasien WHERE id_pasien=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, id);

        ps.executeUpdate();
        System.out.println("\nData pasien berhasil dihapus!");

    } catch (Exception e) {
        System.out.println("\nGagal hapus pasien!");
        e.printStackTrace();
    }
}
}

```

#### d. Class DokterService.java

```

package service;

import database.DatabaseConnection;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class DokterService extends BaseService implements CrudService
{

    @Override
    public void tambah() {
        try {

```

```

        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nID Dokter: ");
        String idDokter = input.nextLine().toUpperCase();

        System.out.print("Nama Dokter: ");
        String nama = input.nextLine();

        System.out.print("Email: ");
        String email = input.nextLine();

        System.out.print("Spesialisasi: ");
        String spesialisasi = input.nextLine();

        System.out.print("No HP: ");
        String noHp = input.nextLine();

        System.out.print("Biaya Konsultasi: ");
        double biaya = input.nextDouble();
        input.nextLine();

        String sql = "INSERT INTO dokter " +
            "(id_dokter, nama_dokter, email, spesialisasi, no_hp,
biaya_konsultasi) " +
            "VALUES (?,?,?,?,?)";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, idDokter);
        ps.setString(2, nama);
        ps.setString(3, email);
        ps.setString(4, spesialisasi);
        ps.setString(5, noHp);
        ps.setDouble(6, biaya);

        ps.executeUpdate();
        System.out.println("\nDokter berhasil ditambahkan!");

    } catch (Exception e) {
        System.out.println("\nGagal menambah dokter!");
        e.printStackTrace();
    }
}

@Override
public void tampil() {
    try {
        Connection conn = DatabaseConnection.getConnection();
        ResultSet rs = conn.createStatement().executeQuery("SELECT *
FROM dokter");

        List<String> listIdDokter = new ArrayList<>();

```

```

        System.out.printf(
            "\n%-10s | %-15s | %-18s | %-17s | %-12s | %-10s\n",
            "ID Dokter", "Nama Dokter", "Email", "Spesialisasi", " no HP",
            "Biaya Konsultasi"
        );
        System.out.println("-----");
        -----");

        while (rs.next()) {
            String idDokter = rs.getString("id_dokter");
            listIdDokter.add(idDokter);

            System.out.printf(
                "%-10s | %-15s | %-18s | %-17s | %-12s | %-10s\n",
                idDokter,
                rs.getString("nama_dokter").toUpperCase(),
                rs.getString("email"),
                rs.getString("spesialisasi"),
                rs.getString("no_hp"),
                rs.getDouble("biaya_konsultasi")
            );
        }

        System.out.println("\nTotal data dokter : " + listIdDokter.size());

    } catch (Exception e) {
        System.out.println("\nGagal tampil dokter");
        e.printStackTrace();
    }
}

@Override
public void update() {
    try {
        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nMasukan ID Dokter yang akan diupdate: ");
        String id = input.nextLine();

        System.out.print("ID Dokter Baru: ");
        String idDokter = input.nextLine();

        System.out.print("Nama Dokter Baru: ");
        String nama = input.nextLine();

        System.out.print("Email Baru: ");
        String email = input.nextLine();

        System.out.print("Spesialisasi Baru: ");

```

```

        String spesialisasi = input.nextLine();

        System.out.print("No HP Baru: ");
        String noHp = input.nextLine();

        System.out.print("Biaya Baru: ");
        double biaya = input.nextDouble();

        String sql = "UPDATE dokter SET id_dokter=?, nama_dokter=?,
email=?, spesialisasi=?, no_hp=?, biaya_konsultasi=? WHERE
id_dokter=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, idDokter);
        ps.setString(2, nama);
        ps.setString(3, email);
        ps.setString(4, spesialisasi);
        ps.setString(5, noHp);
        ps.setDouble(6, biaya);
        ps.setString(7, id);

        ps.executeUpdate();
        System.out.println("\nData dokter diperbarui!");

    } catch (Exception e) {
        System.out.println("\nGagal update dokter!");
        e.printStackTrace();
    }
}

@Override
public void hapus() {
    try {
        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nMasukan ID Dokter yang akan dihapus: ");
        String id = input.nextLine();

        String sql = "DELETE FROM dokter WHERE id_dokter=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, id);
        ps.executeUpdate();

        System.out.println("\nDokter dihapus!");

    } catch (Exception e) {
        System.out.println("\nGagal hapus dokter!");
        e.printStackTrace();
    }
}
}

```

e. Class KunjunganService.java

```
package service;

import database.DatabaseConnection;
import java.sql.*;
import java.time.LocalDate;
import java.util.ArrayList;
import java.util.HashMap;
import java.util.List;
import java.util.Map;

public class KunjunganService extends BaseService implements
CrudService {

    @Override
    public void tambah() {
        try {
            Connection conn = DatabaseConnection.getConnection();

            System.out.print("\nID Kunjungan: ");
            String idKunjungan = input.nextLine().toUpperCase();

            System.out.print("ID Pasien: ");
            String idPasien = input.nextLine();

            System.out.print("ID Dokter: ");
            String idDokter = input.nextLine();

            System.out.print("Tanggal Kunjungan: ");
            LocalDate tanggalKunjungan = LocalDate.parse(input.nextLine());

            System.out.print("Keluhan: ");
            String keluhan = input.nextLine();

            System.out.print("Diagnosa: ");
            String diagnosa = input.nextLine();

            System.out.print("Resep: ");
            String resep = input.nextLine();

            System.out.print("Biaya Konsultasi: ");
            double biayaKonsultasi = input.nextDouble();

            System.out.print("Biaya Obat: ");
            double biayaObat = input.nextDouble();

            double biayaTotal = biayaKonsultasi + biayaObat; // perhitungan
            input.nextLine();
        }
    }
}
```



```

        System.out.print("status: ");
        String status = input.nextLine();

        String sql = "INSERT INTO kunjungan " +
            "(id_kunjungan, id_pasien, id_dokter, tanggal_kunjungan,
            keluhan, diagnosa, resep, biaya_konsultasi, biaya_obat, biaya_total,
            status) " +
            "VALUES (?,?,?,?,?,?,?,?,?,?,?)";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, idKunjungan);
        ps.setString(2, idPasien);
        ps.setString(3, idDokter);
        ps.setDate(4, Date.valueOf(tanggalKunjungan));
        ps.setString(5, keluhan);
        ps.setString(6, diagnosa);
        ps.setString(7, resep);
        ps.setDouble(8, biayaKonsultasi);
        ps.setDouble(9, biayaObat);
        ps.setDouble(10, Double.valueOf(biayaTotal));
        ps.setString(11, status);

        ps.executeUpdate();
        System.out.println("\nKunjungan berhasil dicatat!");

    } catch (Exception e) {
        System.out.println("\nGagal tambah kunjungan!");
        e.printStackTrace();
    }
}

@Override
public void tampil() {
    try {
        Connection conn = DatabaseConnection.getConnection();
        ResultSet rs = conn.createStatement().executeQuery("SELECT *
        FROM kunjungan");

        List<String> listIdKunjungan = new ArrayList<>();

        System.out.printf(
            "\n%-7s | %-7s | %-7s | %-10s | %-12s | %-10s | %-12s | %-8s |
            %-8s | %-8s | %-10s\n",
            "ID Kunj", "ID Pas", "ID Dok", "Tgl Kunj", "Keluhan",
            "Diagnosa", "Resep", "B Konsul", "B Obat", "B Total", "Status"
        );
        System.out.println("-----");
        -----");

        while (rs.next()) {
            String idKunjungan = rs.getString("id_kunjungan");

```

```

        listIdKunjungan.add(idKunjungan);

        System.out.printf(
            "%-7s | %-7s | %-7s | %-10s | %-12s | %-10s | %-12s | %-8s |
%-8s | %-8s | %-10s\n",
            idKunjungan,
            rs.getString("id_pasien"),
            rs.getString("id_dokter"),
            rs.getDate("tanggal_kunjungan"),
            rs.getString("keluhan"),
            rs.getString("diagnosa"),
            rs.getString("resep"),
            rs.getDouble("biaya_konsultasi"),
            rs.getDouble("biaya_obat"),
            rs.getDouble("biaya_total"),
            rs.getString("status")
        );
    }

    System.out.println("\nTotal data kunjungan : " +
listIdKunjungan.size());

    } catch (Exception e) {
        System.out.println("\nGagal tampil kunjungan!");
        e.printStackTrace();
    }
}

@Override
public void update() {
    try {
        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nMasukan ID Kunjungan yang akan diupdate:
");

        String id = input.nextLine();

        System.out.print("ID Kunjungan Baru: ");
        String idKunjungan = input.nextLine();

        System.out.print("ID Pasien Baru: ");
        String idPasien = input.nextLine();

        System.out.print("ID Dokter Baru: ");
        String idDokter = input.nextLine();

        System.out.print("Tanggal Kunjungan Baru: ");
        String tanggalKunjungan = input.nextLine();

        System.out.print("Keluhan Baru: ");

```

```

        String keluhan = input.nextLine();

        System.out.print("Diagnosa Baru: ");
        String diagnosa = input.nextLine();

        System.out.print("Resep Baru: ");
        String resep = input.nextLine();

        System.out.print("Biaya Konsultasi Baru: ");
        double biayaKonsultasi = input.nextDouble();

        System.out.print("Biaya Obat Baru: ");
        Double biayaObat = input.nextDouble();

        Double biayaTotal = biayaKonsultasi + biayaObat;
        input.nextLine();

        System.out.print("Status Baru : ");
        String status = input.nextLine();

        String sql = "UPDATE kunjungan SET id_kunjungan=?,
id_pasien=?, id_dokter=?, tanggal_kunjungan=?, keluhan=?, diagnosa=?,
resep=?, biaya_konsultasi=?, biaya_obat=?, biaya_total=?, status=?
WHERE id_kunjungan=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, idKunjungan);
        ps.setString(2, idPasien);
        ps.setString(3, idDokter);
        ps.setString(4, tanggalKunjungan);
        ps.setString(5, keluhan);
        ps.setString(6, diagnosa);
        ps.setString(7, resep);
        ps.setDouble(8, biayaKonsultasi);
        ps.setDouble(9, biayaObat);
        ps.setDouble(10, biayaTotal);
        ps.setString(11, status);
        ps.setString(12, id);

        ps.executeUpdate();
        System.out.println("\nStatus kunjungan berhasil diupdate!");

    } catch (Exception e) {
        System.out.println("\nGagal update kunjungan!");
        e.printStackTrace();
    }
}

@Override
public void hapus() {
    try {

```

```

        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nID Kunjungan yang akan dihapus: ");
        String id = input.nextLine();

        String sql = "DELETE FROM kunjungan WHERE
id_kunjungan=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, id);

        ps.executeUpdate();
        System.out.println("\nData kunjungan berhasil dihapus!");

    } catch (Exception e) {
        System.out.println("\nGagal hapus kunjungan!");
        e.printStackTrace();
    }
}

public void laporanHarian() {

    String sql = ""
        SELECT id_kunjungan, id_pasien, id_dokter, status, biaya_total
        FROM kunjungan
        WHERE DATE(tanggal_kunjungan) = CURDATE()
        """;

    int totalKunjungan = 0;
    double totalPendapatan = 0;

    Map<String, Integer> statusMap = new HashMap<>();
    Map<String, Integer> dokterMap = new HashMap<>();

    System.out.println("\n=====
=====");
    System.out.println("    LAPORAN KUNJUNGAN HARIAN");
    System.out.println("Tanggal : " + LocalDate.now());
    System.out.println("=====
=====");

    System.out.println("\nDetail Kunjungan : \n");
    System.out.printf(
        "%-12s | %-10s | %-10s | %-10s | %-10s\n",
        "ID ", "Pasien", "Dokter", "Status", "Biaya"
    );
    System.out.println("-----
---");

    try (Connection conn = getConnection();
        PreparedStatement ps = conn.prepareStatement(sql);

```

```

        ResultSet rs = ps.executeQuery() {

// ===== DATA =====
while (rs.next()) {
    totalKunjungan++;
    totalPendapatan += rs.getDouble("biaya_total");

    String status = rs.getString("status");
    String dokter = rs.getString("id_dokter");

    statusMap.put(status, statusMap.getOrDefault(status, 0) + 1);
    dokterMap.put(dokter, dokterMap.getOrDefault(dokter, 0) + 1);

    System.out.printf(
        "%-12s | %-10s | %-10s | %-10s | %-10.0f\n",
        rs.getString("id_kunjungan"),
        rs.getString("id_pasien"),
        dokter,
        status,
        rs.getDouble("biaya_total")
    );
}

} catch (Exception e) {
    System.out.println("Gagal menampilkan laporan harian");
    e.printStackTrace();
    return;
}

// ===== RINGKASAN (SETELAH LOOP) =====
System.out.println("-----
---");
System.out.println("\nTotal Kunjungan : " + totalKunjungan);
System.out.println("Total Pendapatan : Rp " + totalPendapatan);

System.out.println("\nStatus Kunjungan:");
for (Map.Entry<String, Integer> entry : statusMap.entrySet()) {
    System.out.println("- " + entry.getKey() + " : " + entry.getValue());
}

System.out.println("=====
=====");
System.out.println("\nTekan ENTER untuk kembali...");
input.nextLine();
}
}

```

f. Class ObatService.java

```
package service;
```

```

import database.DatabaseConnection;
import java.sql.*;
import java.util.ArrayList;
import java.util.List;

public class ObatService extends BaseService implements CrudService {

    @Override
    public void tambah() {
        try {
            Connection conn = DatabaseConnection.getConnection();

            System.out.print("\nID Obat: ");
            String idObat = input.nextLine().toUpperCase();

            System.out.print("Nama Obat: ");
            String namaObat = input.nextLine();

            System.out.print("Kategori: ");
            String kategori = input.nextLine();

            System.out.print("Stok: ");
            int stok = input.nextInt();

            System.out.print("Harga: ");
            double harga = input.nextDouble();

            String sql = "INSERT INTO obat " +
                "(id_obat, nama_obat, kategori, stok, harga) " +
                "VALUES (?, ?, ?, ?, ?)";
            PreparedStatement ps = conn.prepareStatement(sql);
            ps.setString(1, idObat);
            ps.setString(2, namaObat);
            ps.setString(3, kategori);
            ps.setInt(4, stok);
            ps.setDouble(5, harga);

            ps.executeUpdate();
            System.out.println("\nObat berhasil ditambahkan!");

        } catch (Exception e) {
            System.out.println("\nGagal menambah obat!");
            e.printStackTrace();
        }
    }

    @Override
    public void tampil() {
        try {
            Connection conn = DatabaseConnection.getConnection();

```

```
ResultSet rs = conn.createStatement().executeQuery("SELECT *  
FROM obat");
```

```
List<String> listIdObat = new ArrayList<>();
```

```
System.out.printf(  
    "\n%-10s | %-15s | %-15s | %-8s | %-10s\n",  
    "ID Obat", "Nama Obat", "Kategori", "Stok", "Harga"  
);  
System.out.println("-----  
-----");
```

```
while (rs.next()) {  
    String idObat = rs.getString("id_obat");  
    listIdObat.add(idObat);  
    System.out.printf(  
        "%-10s | %-15s | %-15s | %-8d | %-10.0f\n",  
        idObat,  
        rs.getString("nama_obat").toUpperCase(),  
        rs.getString("kategori"),  
        rs.getInt("stok"),  
        rs.getDouble("harga")  
    );  
}
```

```
System.out.println("\nTotal data obat : " + listIdObat.size());
```

```
} catch (Exception e) {  
    System.out.println("\nGagal tampil obat!");  
    e.printStackTrace();  
}  
}
```

```
@Override  
public void update() {  
    try {  
        Connection conn = DatabaseConnection.getConnection();  
  
        System.out.print("\nMasukan ID Obat yang akan diupdate: ");  
        String id = input.nextLine();  
  
        System.out.print("ID Obat Baru: ");  
        String idObat = input.nextLine();  
  
        System.out.print("Nama Obat Baru: ");  
        String namaObat = input.nextLine();  
  
        System.out.print("Kategori Baru: ");  
        String kategori = input.nextLine();
```

```

        System.out.print("Stok Baru: ");
        int stok = input.nextInt();

        System.out.print("Harga Baru: ");
        Double harga = input.nextDouble();
        input.nextLine();

        String sql = "UPDATE obat SET id_obat=?, nama_obat=?,
kategori=?, stok=?, harga=? WHERE id_obat=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, idObat);
        ps.setString(2, namaObat);
        ps.setString(3, kategori);
        ps.setInt(4, stok);
        ps.setDouble(5, harga);
        ps.setString(6, id);
        ps.executeUpdate();

        System.out.println("\nStok obat diperbarui!");

    } catch (Exception e) {
        System.out.println("\nGagal update obat!");
        e.printStackTrace();
    }
}

@Override
public void hapus() {
    try {
        Connection conn = DatabaseConnection.getConnection();

        System.out.print("\nMasukan ID Obat yang akan dihapus: ");
        String id = input.nextLine();

        String sql = "DELETE FROM obat WHERE id_obat=?";
        PreparedStatement ps = conn.prepareStatement(sql);
        ps.setString(1, id);
        ps.executeUpdate();

        System.out.println("\nObat dihapus!");

    } catch (Exception e) {
        System.out.println("\nGagal hapus obat!");
        e.printStackTrace();
    }
}
}

```

4. package app



[illegible]

```

System.out.print("\nPilih menu : ");
pilihPasien = input.nextInt();

switch (pilihPasien) {
    case 1:
        pasienService.tambah();
        break;
    case 2:
        pasienService.tampil();
        break;
    case 3:
        pasienService.update();
        break;
    case 4:
        pasienService.hapus();
        break;
    case 5:
        System.out.println("\nKembali ke menu utama...\n");
        break;
    default:
        System.out.println("\nMenu tidak valid!\n");
}
} while (pilihPasien != 5);
break;

// DOKTER
case 2:
    int pilihDokter;
    do {
        System.out.println("\n==== MANAJEMEN
DOKTER ===");
        System.out.println("1. Tambah Dokter");
        System.out.println("2. Lihat Dokter");
        System.out.println("3. Update Dokter");
        System.out.println("4. Hapus Dokter");
        System.out.println("5. Kembali");
        System.out.print("Pilih menu : ");
        pilihDokter = input.nextInt();

        switch (pilihDokter) {
            case 1:
                dokterService.tambah();
                break;
            case 2:
                dokterService.tampil();
                break;
            case 3:
                dokterService.update();
                break;
            case 4:

```

```

        dokterService.hapus();
        break;
    case 5:
        System.out.println("\nKembali ke menu utama...\n");
        break;
    default:
        System.out.println("\nMenu tidak valid!\n");
    }
} while (pilihDokter != 5);
break;

// KUNJUNGAN
case 3:
    int pilihKunjungan;
    do {
        System.out.println("\n====  MANAJEMEN
KUNJUNGAN  ===");
        System.out.println("1. Tambah Kunjungan");
        System.out.println("2. Lihat Kunjungan");
        System.out.println("3. Update Kunjungan");
        System.out.println("4. Hapus Kunjungan");
        System.out.println("5. Kembali");
        System.out.print("Pilih menu : ");
        pilihKunjungan = input.nextInt();

        switch (pilihKunjungan) {
            case 1:
                kunjunganService.tambah();
                break;
            case 2:
                kunjunganService.tampil();
                break;
            case 3:
                kunjunganService.update();
                break;
            case 4:
                kunjunganService.hapus();
                break;
            case 5:
                System.out.println("\nKembali ke menu utama...\n");
                break;
            default:
                System.out.println("\nMenu tidak valid!\n");
        }
    } while (pilihKunjungan != 5);
    break;

// OBAT
case 4:
    int pilihObat;

```

```

do {
    System.out.println("\n===  MANAJEMEN
OBAT ===");
    System.out.println("1. Tambah Obat");
    System.out.println("2. Lihat Obat");
    System.out.println("3. Update Obat");
    System.out.println("4. Hapus Obat");
    System.out.println("5. Kembali");
    System.out.print("Pilih menu : ");
    pilihObat = input.nextInt();

    switch (pilihObat) {
        case 1:
            obatService.tambah();
            break;
        case 2:
            obatService.tampil();
            break;
        case 3:
            obatService.update();
            break;
        case 4:
            obatService.hapus();
            break;
        case 5:
            System.out.println("\nKembali ke menu utama...\n");
            break;
        default:
            System.out.println("\nMenu tidak valid!\n");
    }
    } while (pilihObat != 5);
    break;

    // STATISTIK
    case 5:
        System.out.println("\n=== STATISTIK & LAPORAN
HARIAN ===");
        kunjunganService.laporanHarian();
        break;

    // KELUAR
    case 6:
        System.out.println("\nTerima kasih telah menggunakan
sistem.\n");
        System.exit(0);
        break;

    default:
        System.out.println("\nMenu tidak valid!\n");
    }
}

```

```
}  
}  
}  
}
```

Console :

Menu Utama :

```
=====
  SISTEM INFORMASI KLINIK
=====
1. Manajemen Pasien
2. Manajemen Dokter
3. Manajemen Kunjungan
4. Manajemen Obat
5. Statistik & Laporan Kunjungan Harian
6. Keluar
=====
Pilih menu :
```

1. manajemen pasien

```
===  MANAJEMEN PASIEN  ===
1. Tambah Pasien
2. Lihat Pasien
3. Update Pasien
4. Hapus Pasien
5. Kembali

Pilih menu : █
```

#### a. tambah pasien

```
===  MANAJEMEN PASIEN  ===
1. Tambah Pasien
2. Lihat Pasien
3. Update Pasien
4. Hapus Pasien
5. Kembali

Pilih menu : 1

ID Pasien: PSN004
Nama Pasien: dio
Email: dio@gmail
No HP: 082442355346
Alamat: padang
Tanggal Lahir (yyyy-MM-dd): 2006-11-17
Jenis Kelamin: laki-laki

Data pasien berhasil ditambahkan!
```

#### b. lihat pasien

```
===  MANAJEMEN PASIEN  ===
1. Tambah Pasien
2. Lihat Pasien
3. Update Pasien
4. Hapus Pasien
5. Kembali

Pilih menu : 2
```

ID Pasien	Nama	Email	No HP	Alamat	Tgl Lahir	Jenis Kelamin
PSN001	ANI	ani@gmail	082391283905	padang	2005-12-20	perempuan
PSN002	BUDI	budi@gmail	082143658709	padang	2006-06-17	laki-laki
PSN003	CACA	caca@gmail	081324354657	padang	2006-02-18	perempuan
PSN004	DIO	dio@gmail	082442355346	padang	2006-11-15	laki-laki

```
Total data pasien : 4
```

#### c. update pasien

```
===  MANAJEMEN PASIEN  ===
1. Tambah Pasien
2. Lihat Pasien
3. Update Pasien
4. Hapus Pasien
5. Kembali

Pilih menu : 3

Masukkan ID Pasien yang akan diupdate: PSN004
ID Pasien Baru: PSN004
Nama Baru: dio
No HP Baru: 084224533564
Email Baru: dio@gmail
Alamat Baru: padang
Tanggal Lahir Baru: 2005-11-15
Jenis Kelamin Baru: laki-laki

Data pasien berhasil diupdate!
```

d. hapus pasien

```
===  MANAJEMEN PASIEN  ===  
1. Tambah Pasien  
2. Lihat Pasien  
3. Update Pasien  
4. Hapus Pasien  
5. Kembali  
  
Pilih menu : 4  
  
ID Pasien yang akan dihapus: PSN004  
  
Data pasien berhasil dihapus!
```

2. manajemen dokter

```
===  MANAJEMEN DOKTER  ===  
1. Tambah Dokter  
2. Lihat Dokter  
3. Update Dokter  
4. Hapus Dokter  
5. Kembali  
Pilih menu : █
```

a. tambah dokter

```
===  MANAJEMEN DOKTER  ===  
1. Tambah Dokter  
2. Lihat Dokter  
3. Update Dokter  
4. Hapus Dokter  
5. Kembali  
Pilih menu : 1  
  
ID Dokter: DKT003  
Nama Dokter: rudi hartono  
Email: rudihartono@gmail  
Spesialisasi: penyakit dalam  
No HP: 081221233234  
Biaya Konsultasi: 150000  
  
Dokter berhasil ditambahkan!
```

## b. lihat dokter

```
===  MANAJEMEN DOKTER  ===
1. Tambah Dokter
2. Lihat Dokter
3. Update Dokter
4. Hapus Dokter
5. Kembali
Pilih menu : 2

ID Dokter | Nama Dokter | Email | Spesialisasi | no HP | Biaya Konsultasi
-----|-----|-----|-----|-----|-----
DKT001 | SARAH AMELIA | sarahamelia@gmail | umum | 081234567890 | 100000.0
DKT002 | BUDI SANTOSO | budisantoso@gmail | anak | 082435465768 | 125000.0
DKT003 | RUDI HARTONO | rudihartono@gmail | penyakit dalam | 081221233234 | 150000.0

Total data dokter : 3
```

## c. Update dokter

```
===  MANAJEMEN DOKTER  ===
1. Tambah Dokter
2. Lihat Dokter
3. Update Dokter
4. Hapus Dokter
5. Kembali
Pilih menu : 3

Masukan ID Dokter yang akan diupdate: DKT003
ID Dokter Baru: DKT003
Nama Dokter Baru: rudi pratama
Email Baru: rudipratama@gmail
Spesialisasi Baru: penyakit dalam
No HP Baru: 081221233234
Biaya Baru: 125000

Data dokter diperbarui!
```

## d. hapus dokter

```
===  MANAJEMEN DOKTER  ===
1. Tambah Dokter
2. Lihat Dokter
3. Update Dokter
4. Hapus Dokter
5. Kembali
Pilih menu : 4

Masukan ID Dokter yang akan dihapus: DKT003

Dokter dihapus!
```



### 3. manajemen kunjungan

```
===  MANAJEMEN KUNJUNGAN  ===
1. Tambah Kunjungan
2. Lihat Kunjungan
3. Update Kunjungan
4. Hapus Kunjungan
5. Kembali
Pilih menu : █
```

#### a. tambah kunjungan

```
===  MANAJEMEN KUNJUNGAN  ===
1. Tambah Kunjungan
2. Lihat Kunjungan
3. Update Kunjungan
4. Hapus Kunjungan
5. Kembali
Pilih menu : 1

ID Kunjungan: KJG003
ID Pasien: PSN003
ID Dokter: DKT003
Tanggal Kunjungan: 2025-12-30
Keluhan: nyeri perut
Diagnosa: maag
Resep: vitamin c
Biaya Konsultasi: 150000
Biaya Obat: 20000
status: proses

Kunjungan berhasil dicatat!
```

#### b. lihat kunjungan

```
===  MANAJEMEN KUNJUNGAN  ===
1. Tambah Kunjungan
2. Lihat Kunjungan
3. Update Kunjungan
4. Hapus Kunjungan
5. Kembali
Pilih menu : 2
```

ID Kunj	ID Pas	ID Dok	Tgl Kunj	Keluhan	Diagnosa	Resep	B Konsul	B Obat	B Total	Status
KJG001	PSN001	DKT001	2025-12-30	demam	flu	paracetamol	100000.0	10000.0	110000.0	proses
KJG002	PSN002	DKT002	2025-12-30	pilek	ispa	amoxillin	125000.0	20000.0	145000.0	proses
KJG003	PSN003	DKT003	2025-12-30	nyeri perut	maag	vitamin c	150000.0	20000.0	170000.0	proses

Total data kunjungan : 3

#### c. update kunjungan

```
===  MANAJEMEN KUNJUNGAN  ===
1. Tambah Kunjungan
2. Lihat Kunjungan
3. Update Kunjungan
4. Hapus Kunjungan
5. Kembali
Pilih menu : 3

Masukan ID Kunjungan yang akan diupdate: KJG003
ID Kunjungan Baru: KJG003
ID Pasien Baru: PSN003
ID Dokter Baru: DKT003
Tanggal Kunjungan Baru: 2025-12-30
Keluhan Baru: nyeri perut
Diagnosa Baru: maag
Resep Baru: vitamin c
Biaya Konsultasi Baru: 150000
Biaya Obat Baru: 10000
Status Baru : selesai

Status kunjungan berhasil diupdate!
```

#### d. hapus kunjungan

```
===  MANAJEMEN KUNJUNGAN  ===
1. Tambah Kunjungan
2. Lihat Kunjungan
3. Update Kunjungan
4. Hapus Kunjungan
5. Kembali
Pilih menu : 4

ID Kunjungan yang akan dihapus: KJG003

Data kunjungan berhasil dihapus!
```

#### 4. manajemen obat

```
===  MANAJEMEN OBAT  ===
1. Tambah Obat
2. Lihat Obat
3. Update Obat
4. Hapus Obat
5. Kembali
Pilih menu : █
```

a. tambah obat

```
===  MANAJEMEN OBAT  ===
1. Tambah Obat
2. Lihat Obat
3. Update Obat
4. Hapus Obat
5. Kembali
Pilih menu : 1

ID Obat: OBT003
Nama Obat: vitamin c
Kategori: vitamin
Stok: 150
Harga: 5000

Obat berhasil ditambahkan!
```

b. lihat obat

```
===  MANAJEMEN OBAT  ===
1. Tambah Obat
2. Lihat Obat
3. Update Obat
4. Hapus Obat
5. Kembali
Pilih menu : 2
```

ID Obat	Nama Obat	Kategori	Stok	Harga
OBT001	PARACETAMOL	analgesik	100	5000
OBT002	AMOXICILLIN	antibiotik	50	10000
OBT003	VITAMIN C	vitamin	150	5000

```
Total data obat : 3
```

c. update obat

```
===  MANAJEMEN OBAT  ===
1. Tambah Obat
2. Lihat Obat
3. Update Obat
4. Hapus Obat
5. Kembali
Pilih menu : 3

Masukan ID Obat yang akan diupdate: OBT003
ID Obat Baru: OBT003
Nama Obat Baru: vitamin c
Kategori Baru: vitamin
Stok Baru: 200
Harga Baru: 5000

Stok obat diperbarui!
```

#### d. hapus obat

```
===  MANAJEMEN OBAT  ===
1. Tambah Obat
2. Lihat Obat
3. Update Obat
4. Hapus Obat
5. Kembali
Pilih menu : 4

Masukan ID Obat yang akan dihapus: OBT003

Obat dihapus!
```

#### 5. statistic dan laporan harian

```
=== STATISTIK & LAPORAN HARIAN ===

=====
                LAPORAN KUNJUNGAN HARIAN
Tanggal : 2025-12-30
=====

Detail Kunjungan  :

ID          | Pasien   | Dokter   | Status   | Biaya
-----|-----|-----|-----|-----
KJG001      | PSN001   | DKT001   | proses   | 110000
KJG002      | PSN002   | DKT002   | proses   | 145000
-----|-----|-----|-----|-----

Total Kunjungan   : 2
Total Pendapatan  : Rp 255000.0

Status Kunjungan:
- proses : 2
=====

Tekan ENTER untuk kembali...
```

Link Github : <https://github.com/aqilahigenea/tugas-besar-PBO.git>