PRAKTIKUM
PEMROGRAMAN BERORIENTASI OBJEK

SEMESTER GENAP T.A 2019/2020

LAPORAN PROYEK AKHIR

DISUSUN OLEH :

NIM	**:**	**123180137** **123180169**
NAMA	**:**	**DEMITRIES BASKHARA R.T.** **AULIA SALSABILA**
KELAS/PLUG	**:**	**H**
NAMA ASISTEN	**:**	**LUKMANUL HAKIM** **AMIRA SALSABILA**

PROGRAM STUDI INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNIK INDUSTRI
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN"
YOGYAKARTA

2020

HALAMAN PENGESAHAN

LAPORAN PROYEK AKHIR

Disusun oleh :

Demitries Baskhara R.T.	123180137
Aulia Salsabila	123180169

Telah Diperiksa dan Disetujui oleh Asisten Praktikum Pemrograman Berorientasi Objek
Pada Tanggal : .....

Asisten Praktikum	Asisten Praktikum
Lukmanul Hakim	**Amira Salsabila**
NIM. 124180060	**NIM. 124180067**

Mengetahui,
Ka. Lab. Basis Data

Wilis Kaswidjanti, S.Si.,M.Kom.
NIK. 2 7512 00 0229 1

KATA PENGANTAR

Puji syukur kami panjatkan kepada Tuhan Yang Maha Esa yang senantiasa mencurahkan rahmat dan hidayah-Nya sehingga kami dapat menyelesaikan praktikum pemrograman beorientasi objek serta laporan proyek akhir praktikum yang bertopik Apotek Online. Adapun laporan ini berisi tentang proyek akhir yang kami pilih dari hasil pembelajaran selama praktikum berlangsung.

Tidak lupa ucapan terimakasih kepada asisten dosen yang selalu membimbing dan mengajari kami dalam melaksanakan praktikum dan dalam menyusun laporan ini. Laporan ini masih sangat jauh dari kesempurnaan, oleh karena itu kritik serta saran yang membangun kami harapkan untuk menyempurnakan laporan akhir ini.

Atas perhatian dari semua pihak yang membantu penulisan ini, kami ucapkan terimakasih. Semoga laporan ini dapat dipergunakan seperlunya.

Yogyakarta, 15 Mei 2020

Penyusun

DAFTAR ISI

BAB I
PENDAHULUAN

1.1 Latar Belakang Masalah

Apotek menjadi tempat utama dalam membeli berbagai jenis obat. Selama manusia masih hidup, maka apotek akan terus beroperasi dalam hidup kita. Baik penyakit ringan hingga penyakit menular, apotek akan menjadi tempat penyetokan obat paling utama.

Dewasa ini teknologi semakin maju sehingga memudahkan banyak pihak untuk melakukan pekerjaannya. Dalam menjalankan bisnis apotek, akan menjadi luas jaringannya apabila dibuatnya aplikasi pemesanan obat online.

1.2 Tujuan Proyek Akhir

Tujuan pembuatan aplikasi ini adalah menjaring konsumen lebih luas, memudahkan konsumen dalam mendapatkan obat, dan memudahkan pemilik usaha untuk memonitor hasil penjualan. Aplikasi ini juga sebagai tugas akhir dari praktikum pemrograman berorientasi objek.

BAB II
ISI DAN PEMBAHASAN

2.1 Dasar Teori

1. CLASS

Deklarasi class pada Java:

```
[modifier1] class NamaKelas [modifier2] {
class body;
}
```

Listing Program 2.1 Deklarasi Class pada Java

Keterangan :

1. Tanda [ ] bersifat optional.
2. Modifier1 dapat berupa : public, abstract, final
3. Modifier2 dapat berupa : extends, implements
4. Class body merupakan isi dari program yang terdiri dari contructor, atribut dan method.

2. METHOD

Deklarasi method:

```
[modifier] tipeNilaiKembalian namaMethod (parameter input) {
method body;
}
```

Listing Program 2.2 Deklarasi Method pada Java

Keterangan :

a) Tanda [] bersifat optional.

b) Modifier dapat berupa : public, protected, private, static, abstract, final, native dan synchronized

c) Dilihat dari parameter input, method dibagi menjadi dua yaitu membutuhkan input atau tidak. Bila membutuhkan input, maka tipe data dari parameter input harus dituliskan. Gunakan koma untuk memisahkan setiap parameter input. Bila tidak membutuhan input, maka bagian parameter input dikosongkan.

Cara pemanggilan method:

1. NamaKelas.NamaMethod(parameterInput);
2. NamaObjek.NamaMethod(parameterInput);

3. CONSTRUCTOR

Constructor merupakan method yang namanya sama dengan nama kelas dimana method tersebut berada. Constructor dieksekusi pertama kali saat suatu kelas diinstansiasi menjadi objek. Biasanya, Constructor digunakan untuk menginisialisasi nilai awal (memberikan nilai default) pada atribut-atribut yang dimiliki oleh suatu objek saat objek itu pertama kali dibuat.

4. MODIFIER

Modifier adalah sifat yang dimiliki oleh setiap atribut, method maupun kelas dalam java. Modifier akses adalah modifier yang selalu digunakan, modifier akses terdiri dari private, default, protected dan public. Kriteria modifier-modifier tersebut adalah :

Tabel 2.1 Wilayah Akses

Wilayah Akses	public	protected	default	Private
Kelas yang sama	ya	ya	ya	ya
Antar kelas dalam package yang sama	ya	ya	ya	Tidak
SubKelas di package yang berbeda	ya	ya	Tidak	Tidak
Bukan subkelas, package yang berbeda	ya	Tidak	Tidak	Tidak

Beberapa modifier penting lainnya :

static yaitu modifier yang digunakan agar suatu atribut maupun method dapat diakses oleh objek atau kelas lain meski tanpa dilakukan instansiasi terhadap kelas dimana atribut maupun
method itu berada, method main adalah salah satu contoh method bermodifier static yang sering digunakan.

final yaitu modifier yang digunakan untuk mencegah kemungkinan modifikasi terhadap atribut maupun method, dengan modifier ini suatu atribut akan berlaku sebagai konstanta

5. OBJECT

Sebuah kelas dapat digunakan untuk membuat banyak objek. Setiap objek dapat diperlakukan secara berbeda oleh objek-objek lain yang menggunakannya. Dibutuhkan operator new untuk membuat objek dari suatu kelas.

Objek dibuat dengan cara menuliskan :

```
1. NamaKelas NamaObjek = new NamaConstructor();
2. Nama Kelas NamaObjek;
NamaObjek = new NamaConstructor();
```

Listing Program 2.3 Instansiasi Objek pada Java

Proses diatas disebut instansiasi. Ketika objek selesai dibuat, objek tersebut disimpan di dalam
memori dan dapat diakses oleh objek-objek lain melalui NamaObjek nya.

6. PACKAGE

Package digunakan untuk mengelompokkan file kelas (*.class) yang terkait (karena jenisnya, fungsinya, atau karena alasan lainnya) pada folder yang sama, dimana di dalam setiap
kelasnya terdapat directive (statement java dalam source code yang digunakan untuk membuat
kelas) package yang mengacu pada folder tersebut.

Deklarasi package:

```
package namapackage;
```

Listing Program 2.4 Deklarasi Package pada Java

Untuk bisa mengakses kelas yang berbeda package, diperlukan pernyataan import baik pada kelas yang ingin diakses, maupun package yang menampung kelas tersebut, berikut ini contohnya:

```
1. import namaPackage.namaKelas; // untuk mengimport suatu kelas
2. import namaPackage.*; // untuk mengimport semua kelas dalam package tersebut
```

Listing Program 2.5 Import Package pada Java

7. DATABASE

JDBC (Java DataBase Connectivity) merupakan teknologi Java yang populer sebagai standar pengaksesan database. Java tidak membedakan cara pemrograman database yang satu dengan lainnya. Sebagai contoh: cara Java mengolah tabel di Oracle dan MySQL sama saja, yang berbeda hanyalah alamat URL koneksi dan driver JDBC-nya. Sehingga kita dapat membuat sebuah program yang sama di Java untuk mengolah database yang ada di MySQL, Oracle, SQL Server dan lain-lain asalkan nama dan struktur tabelnya sama.

1. JDBC API (Application Programming Interface)

Aplikasi Java yang dibuat, agar dapat terhubung dengan database memerlukan driver JDBC yang sesuai. Biasanya setiap vendor database yang populer seperti Oracle, MySQL dan SQL Server telah menyediakan driver JDBC yang bisa didownload dari website-nya. JDBC API tersedia dalam paket java.sql dan javax.sql. Teknologi JDBC memungkinkan untuk menangani database dengan langkah–langkah sebagai berikut :

1. Memanggil Driver JDBC

Untuk melakukan koneksi dengan database memerlukan sebuah driver. Driver untuk masing – masing database berbeda satu sama lain. misalnya kita mempergunakan driver MySQL :

```
Class.forName("com.mysql.jdbc.Driver");
```

Listing Program 2.6 Deklarasi Class untuk Driver JDBC pada Java

2. Melakukan koneksi database

Mengkoneksikan dengan database memerlukan pengriman informasi URL, username, dan password untuk database ke metode getConnection dengan cara sebagai berikut :

```
String url == " jdbc:mysql://localhost/praktikum";
Connection con =
DriverManager.getConnection("url?user=root&amp;password=vi");
```

Listing Program 2.7 Melakukan Koneksi Database pada Java

3. Membuat Objek Statement

Untuk melakukan query, kita memerlukan objek Statement dengan syntax sebagai berikut:

Statement stmt = con.createStatement();

4. Melakukan Query

Setelah memiliki objek Statement. Kita dapat mengirimkan query dan mengeksekusinya dengan metode executeQuery, yang mengembalikan objek bertipe ResultSet. Contohnya sebagai berikut :

```
String sql = "SELECT * FROM nama_tabel";
ResultSet rs = stmt.executeQuery(sql);
```

Listing Program 2.8 Melakukan Query pada Java

Sedangkan jika melakukan pembuatan atau modifikasi database menggunakan metode

executeUpdate. Contohnya sebagai berikut:

```
String sql = "CREATE TABLE nama_tabel"+"( , )";
ResultSet rs = stmt.executeUpdate(sql);
```

Listing Program 2.8 Melakukan Modifikasi Database pada Java

5. Memproses Hasil

Untuk memproses kita menggunakan objek resultSet karena hasil dari query disimpan dalam objek ini. Metode yang digunakan ialah metode next dan getString seperti contoh berikut ini :

```
While (rs.next)
System.out.println(rs.getString(1));
```

Listing Program 2.9 Memproses Hasil Query pada Java

Penggunaan while dengan metode resultSet.next dimaksudkan untuk memproses baris demi baris hasil query. Proses query akan berhenti saat semua baris sudah diproses. Metode getString digunakan untuk memperoleh data string yang diikuti nomor kolom.

6. Menutup Koneksi

Sebelum menutup koneksi dengan database, kita perlu melepaskan objek ResultSet dengan syntax seperti ini:

```
stmt.close();
```

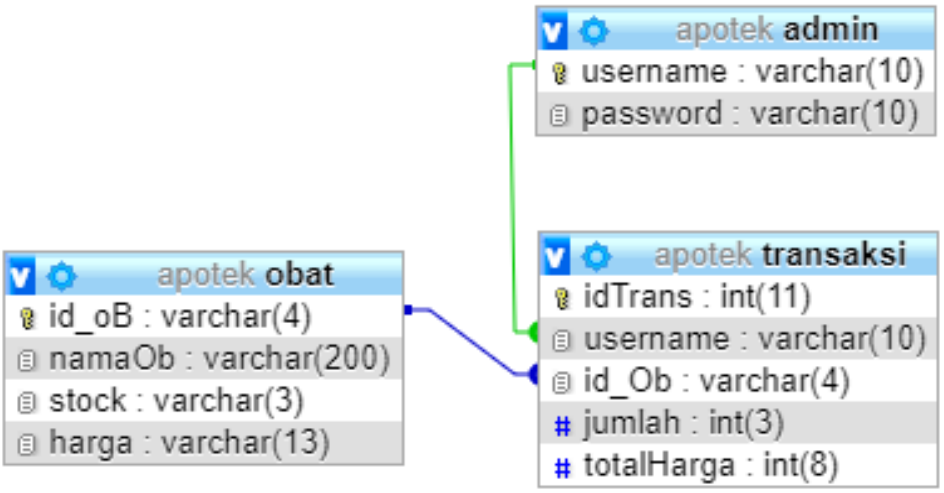
Listing Program 2.10 Menutup Objek ResultSet pada Java

Menutup koneksi dengan database:

```
con.close();
```

Listing Program 2.11 Menutup Koneksi Database pada Java

2.2 Perancangan RAT Database



Gambar 2.1 RAT Database

2.3 Listing Program

Berikut listing program dari projectnya. Pertama yaitu package Login untuk proses login.

```
import javax.swing.*;
public class Login extends JFrame {
    JLabel lusername = new JLabel("Username");
    JLabel lpass = new JLabel("Password");
    JTextField txtusername = new JTextField();
    JPasswordField txtpassword = new JPasswordField();
    JButton login = new JButton("Login");
    public Login() {
        setLayout(null);
        add(lusername);
        add(lpass);
        add(txtusername);
        add(txtpassword);
        add(login);
        lusername.setBounds(50, 25, 100, 20);
        lpass.setBounds(50, 60, 100, 20);
        txtusername.setBounds(130, 25, 140, 20);
        txtpassword.setBounds(130, 60, 140, 20);
        login.setBounds(130, 100, 80, 20);
        setSize(350,200);
        setLocationRelativeTo(null);
        setVisible(true);
        setDefaultCloseOperation(EXIT_ON_CLOSE);
    }
    String getUsername() { return txtusername.getText(); }
```

Listing Program 2.12 Class Login

```
import javax.swing.*;
import java.sql.*;
import java.util.TimeZone;
```

Listing Program 2.13 Class Model

```
public class Model {

        static final String JDBC_DRIVER = "com.mysql.cj.jdbc.Driver";

        static final String DB_URL =
"jdbc:mysql://localhost/apotek?serverTimezone=" +
TimeZone.getDefault().getID();

        static final String USER = "root";

        static final String PASS = "";

        String username;

        Connection koneksi;

        Statement statement;

         //+----------------------------------------+ connect ke
database

        public Model()

        {

            try

            {

                Class.forName(JDBC_DRIVER);

                koneksi = (Connection) DriverManager.getConnection(DB_URL,
USER, PASS);

                System.out.println("Koneksi Berhasil");

            }

            catch(Exception ex)

            {

                JOptionPane.showMessageDialog(null, ex.getMessage());

                System.out.println("Koneksi Gagal");

            }

        }

 public int login(String a , String b)

        {

            try

            {

                int jmlData = 0;

                username = a;
```

Lanjutan Listing Program 2.13 Class Model

```
statement = koneksi.createStatement();
                String query = "SELECT * FROM admin where username='"+a+"'
and password='"+b+"'"; //pengambilan dara dalam java dari database
                ResultSet resultSet = statement.executeQuery(query);
                while (resultSet.next())
                {
                    jmlData++;
                }
                if(jmlData > 0)
                {
                    JOptionPane.showMessageDialog(null, "Selamat datang, "
+ a + "!");
                    return 1;
                }
                else
                {
                    JOptionPane.showMessageDialog(null, "Username/Password
salah!");
                    return 0;
                }
            }
            catch(SQLException e)
            {
                System.out.println(e.getMessage());
                System.out.println("SQL Error");
                return 0;
            }
        }
public String getUsername() {
        return username;
    }

    //+-----------------------------------------+ read database
        public String[][] readData(){
```

Lanjutan Listing Program 2.13 Class Model

```
try{
                int jmlData = 0; // utk nampung jumlah data
                String data[][] = new String[getBanyakData()][4]; // baris
blm diketahui, kolom 4
                String query = "SELECT * FROM obat";
                ResultSet res = statement.executeQuery(query); //
menampilkan hasil execute query dengan ResultSet (karena >1)
                while(res.next()){
                    data[jmlData][0] = res.getString("id_Ob");
                    data[jmlData][1] = res.getString("namaOb");
                    data[jmlData][2] = res.getString("stock");
                    data[jmlData][3] = res.getString("harga");
                    jmlData++;
                }
                return data;
            }catch (Exception e){
                System.out.println(e.getMessage());
                System.out.println("SQL Error!");
                return null;
            }
        }
//+----------------------------------------+ read database transaksi
        public String[][] readTrans(){
            try{
                int jmlData = 0; // utk nampung jumlah data
                String data2[][] = new String[getBanyakData2()][5]; //
baris blm diketahui, kolom 4
                String query = "SELECT * FROM transaksi where username =
'"+getUsername()+"';";
                ResultSet res = statement.executeQuery(query); //
menampilkan hasil execute query dengan ResultSet (karena >1)
                while(res.next()){
                    data2[jmlData][0] = res.getString("idTrans");
                    data2[jmlData][1] = res.getString("username");
                    data2[jmlData][2] = res.getString("id_Ob");
                    data2[jmlData][3] = res.getString("jumlah");
                    data2[jmlData][4] = res.getString("totalHarga");
                }
```

Lanjutan Listing Program 2.13 Class Model

```
jmlData++;
                }
                return data2;
            }catch (Exception e){
                System.out.println(e.getMessage());
                System.out.println("SQL Error!");
                return null;
            }
        }

        public int getBanyakData2() {
            int jmlData = 0;
            try{
                statement = koneksi.createStatement();
                String query = "SELECT * FROM transaksi WHERE username =
'"+getUsername()+"';";
                ResultSet res = statement.executeQuery(query);
                while(res.next()){
                    jmlData++;
                }
                return jmlData;
            }catch (Exception e){
                System.out.println(e.getMessage());
                System.out.println("SQL Error!");
                return 0;
            }
        }
        //+-----------------------------------------+ mengambil jumlah
data yang ada di tabel
        public int getBanyakData() {
            int jmlData = 0;
            try{
```

Lanjutan Listing Program 2.13 Class Model

```
jmlData++;
                }
                return jmlData;
            }catch (Exception e){
                System.out.println(e.getMessage());
                System.out.println("SQL Error!");
                return 0;
            }
        }
        //+----------------------------------------+ create data in
database
        public void insertData(String username, String id_Ob, int jumlah,
int totalHarga){
            try{
                String query = "INSERT INTO transaksi(username, id_Ob,
jumlah, totalHarga) VALUES ('"
                        +username+"', '"+id_Ob+"', "+jumlah+",
"+totalHarga+");";
                //String '"+String+"' kalau Int "+int+"
                statement = (Statement)koneksi.createStatement(); //
prepare statementnya
                statement.executeUpdate(query); // execute querynya
                System.out.println("Berhasil ditambahkan ke database!");
                JOptionPane.showMessageDialog(null,"Pembelian Anda telah
tercatat!");
            }catch (Exception sql){
                System.out.println(sql.getMessage());
                JOptionPane.showMessageDialog(null, sql.getMessage());
            }
        }
public void updateData(String id_Ob, int stock, int jml){
            try{
                if(stock!=0) {
                    String query = "UPDATE obat SET stock = " + (stock -
jml) + " WHERE id_Ob = '" + id_Ob + "';";
                    statement = koneksi.createStatement();
                    statement.executeUpdate(query);
JOptionPane.showMessageDialog(null, "Stock habis!");}
```

Lanjutan Listing Program 2.13 Class Model

```
}catch (Exception sql){
                System.out.println(sql.getMessage());
                JOptionPane.showMessageDialog(null,sql.getMessage());
            }
        }
}
```

Lanjutan Listing Program 2.13 Class Model

2.4 Pembahasan

Pertama-tama dalam menjalankan program, user dihadapkan dengan tampilan login untuk kasir. User login berdasarkan ID dan Password yang sudah terdaftar di database. Setelah login, user dihadapi dengan tampilan utama aplikasi apotek online, dengan beberapa tombol seperti beli dan Riwayat transaksi. Tombol beli apabila pembeli telah pasti ingin membeli suatu obat. Dan Riwayat transaksi akan menampilkan Riwayat transaksi dari akun tersebut.

BAB III
JADWAL PENGERJAAN DAN PEMBAGIAN TUGAS

3.1 Jadwal Pengerjaan

Tabel 3.1 Tabel Jadwal

No.	Rincian Tugas	April				Mei	
		1	2	3	4	1	2
1	*Perancangan Aplikasi*						
2	*Database*						
3	*Pembuatan Aplikasi*						
4	*Pembuatan Laporan*						

3.2 Pembagian Tugas

Tabel 3.2 Pembagian Tugas

Tugas	**Nama Penanggung Jawab**
Perancangan Aplikasi	Demitries dan Aulia
Perancangan dan Pembuatan Database	Aulia
Pembuatan Menu Login	Demitries
Pembuatan Class Model dan Controller	Aulia
Pembuatan Class View	Demitries
Pengevaluasi	Demitries dan Aulia
Pembuatan Laporan	Demitries dan Aulia

BAB IV
KESIMPULAN DAN SARAN

4.1 Kesimpulan

Berdasarkan penjelasan pada bagian sebelumnya, dapat ditarik kesimpulan bahwa program ini dapat memudahkan kasir dalam bertransaksi dan melihat laporan penjualan sehingga mengurangi waktu pengisian data dengan efektif.

4.2 Saran

Penulis menyadari bahwa banyak kekurangan yang terdapat pada aplikasi ini antara lain :

1. Aplikasi ini hanya bisa melihat laporan penjualan dan belum bisa mencetak laporan penjualan.
2. Perlunya pemeliharaan program dikarenakan masih terdapat banyak *bug* dalam program ini.
3. Tampilan yang belum menarik sehingga perlu untuk dibaguskan tampilannya.
4. Aplikasi ini tidak dapat memasukkan data barang ke dalam database sehingga diperlukannya input manual melalui query.

Dari kelemahan diatas, kami menyarankan kepada para pembaca diharapkan dapat untuk mengembangkan aplikasi kasir ini untuk menyempurnakan aplikasi tersebut.

DAFTAR PUSTAKA

Modul Praktikum Pemrograman Berorientasi Objek. (2019). Yogyakarta: Universitas Pembangunan Nasional "Veteran" Yogyakarta.