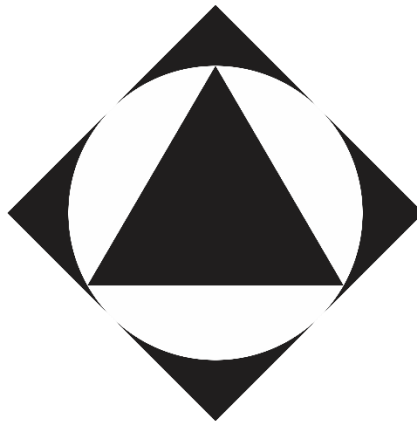


TUGAS AKHIR
PEMROGRAMAN BERBASIS OBJEK
APLIKASI MANAJEMEN PENCUCIAN KENDARAAN



Oleh :

Galih Respati Permana 152019035

PROGRAM STUDI INFORMATIKA
JURUSAN TEKNIK INFORMATIKA
FAKULTAS TEKNOLOGI INDUSTRI
INSTITUT TEKNOLOGI NASIONAL BANDUNG
2021

KATA PENGANTAR

Dengan menyebut nama Tuhan Yang Maha Esa, Penulis panjatkan puja dan puji syukur atas ke hadirat-Nya, yang telah melimpahkan rahmat, hidayah, dan inayah-Nya kepada kami, sehingga kami dapat menyelesaikan laporan akhir ini guna memenuhi tugas untuk mata kuliah Pemrograman Berbasis Objek dengan judul : **“APLIKASI MANAJEMEN PENCUCIAN KENDARAAN”**.

Penulis menyadari dalam penulisan laporan ini tidak terlepas dari bantuan banyak pihak yang dengan tulus memberikan bantuan, saran, dan kritik sehingga laporan akhir ini dapat terselesaikan.

Penulis menyadari bahwa laporan ini masih jauh dari kata sempurna dikarenakan keterbatasan pengalaman dan waktu pengerjaan. Oleh karena itu, penulis mengharapkan kritik dan saran yang membangun dari berbagai pihak yang membaca laporan ini. Akhirnya penulis berharap semoga laporan ini membawa banyak manfaat bagi pembaca.

Bandung, 4 Juni 2021

Galih Respati Permana

DAFTAR ISI

KATA PENGANTAR.....	i
DAFTAR ISI.....	ii
BAB I.....	1
PENDAHULUAN.....	1
1.1. Latar Belakang	1
1.2. Rumusan Masalah	2
1.3. Tujuan Penulisan.....	2
1.4. Batasan Masalah.....	2
1.5. Sistematika Penulisan Laporan	2
BAB II	4
LANDASAN TEORI.....	4
2.1. Pengertian Cuci Kendaraan	4
2.2. Konsep Dasar Sistem.....	4
2.3. Pengertian Java	4
2.4. Pengertian Netbeans.....	5
2.5. Pengertian Database.....	5
BAB III.....	6
MODEL DESKRIPSI SISTEM.....	6
3.1. Studi Kasus	6
3.2. Hak Akses Akun	6
3.3. Alur Proses	7
3.4. Class Diagram.....	8
3.5. Database	8
3.6. Sitemaps	10
3.7. Mockup UI	11

BAB IV	14
IMPLEMENTASI	14
4.1. User Interface dan Listing Code	14
BAB V	47
PENUTUP	47
5.1. Kesimpulan	47
5.2. Saran	47
DAFTAR PUSTAKA	48

BAB I

PENDAHULUAN

1.1. Latar Belakang

Perkembangan inovasi dalam pelayanan peningkatan kualitas sangat penting bagi semua pelaku bisnis, terutama dalam bidang usaha yang berhubungan dengan jasa. Dengan semakin berkembangnya teknologi, bidang jasa akan sangat terbantu apabila mampu mengimplementasikan dengan penggunaan teknologi yang tepat. Seperti yang kita ketahui, saat ini banyak bidang bisnis jasa yang berlomba-lomba menggunakan teknologi canggih untuk membantu meningkatkan pelayanan dan kualitas. Salah satu dari peluang bisnis jasa tersebut adalah cuci kendaraan yang saat ini mulai banyak peminatnya.

Seperti yang kita ketahui, tidak semua orang ingin atau bisa mencuci kendaraannya sendiri, berbagai kendala seperti kesibukan maupun keterbatasan lahan untuk mencuci kendaraan menjadi salah satu sebab mengapa banyak orang yang tertarik untuk menggunakan pelayanan pencucian kendaraan.

Perkembangan teknologi informasi dan komunikasi yang pesat saat ini mampu mempermudah orang-orang dalam melakukan pertukaran informasi dan komunikasi.

Bisnis pencucian kendaraan akan lebih mudah dilakukan apabila memasukkan unsur teknologi dan informasi didalamnya, dimana dalam hal ini dapat meningkatkan efisiensi dalam pengelolaan data dan mempermudah pelanggan untuk melakukan pencucian kendaraan.

Dengan melihat perkembangan dari sisi teknologi informasi, maka penulis memiliki gagasan untuk membuat aplikasi dekstop untuk mengelola pencucian kendaraan. Aplikasi ini diperuntukkan untuk para pelanggan dan pemilik kendaraan agar mempermudah pengelolaan data pencucian.

Berdasarkan uraian diatas, maka penulis membuat aplikasi tugas akhir tentang manajemen pencucian kendaraan yang berjudul **“APLIKASI PENGELOLAAN PENCUCIAN KENDARAAN”**.

1.2. Rumusan Masalah

Berdasarkan uraian latar belakang di atas, maka dapat dirumuskan masalah sebagai berikut :

1. Bagaimana sistem pengelolaan data pada aplikasi pencucian kendaraan.
2. Bagaimana perancangan sistem pengelolaan data pencucian kendaraan agar lebih optimal dalam mengelola data pelanggan dan transaksi pencucian.

1.3. Tujuan Penulisan

Laporan ini dibuat dengan tujuan untuk memenuhi tugas akhir dari mata kuliah Pemrograman Berbasis Objek. Sedangkan tujuan dari kegiatan ini adalah untuk membuat, merancang dan mengimplementasikan rancangan sistem agar dapat berjalan sesuai dengan fungsinya.

1.4. Batasan Masalah

Batasan – batasan yang penulis definisikan pada laporan ini antara lain :

1. Sistem memiliki 2 hak akses pengguna, yaitu User dan Administrator.
2. Pembuatan akun pengguna bisa langsung melalui menu buat akun, sedangkan untuk jenis akun Administrator harus melalui akun Administrator yang lain.
3. Hak Akses Administrator digunakan untuk manajemen data dan transaksi.
4. Hak Akses User digunakan untuk melakukan proses transaksi.
5. Sistem dibuat dengan menggunakan bahasa pemrograman Java.
6. Sistem menggunakan database sebagai tempat menyimpan data, dimana dalam projek ini database yang digunakan adalah mySQL.
7. Sistem digunakan dalam ruang lingkup manajemen pencucian kendaraan.

1.5. Sistematika Penulisan Laporan

Sistematika penulisan laporan ini mengacu pada subbab berikut ini :

1. BAB I, PENDAHULUAN

Pada bab ini menjelaskan tentang tujuan latar belakang pembuatan aplikasi, rumusan masalah, tujuan penulisan laporan dan batasan masalah dari sistem yang penulis buat.

2. BAB II, LANDASAN TEORI

Bab ini menjelaskan tentang teori-teori penunjang yang digunakan dalam pembuatan laporan ini.

3. BAB III, PERANCANGAN

Pada bagian bab ini menjelaskan tentang studi kasus, alur sistem dan database sistem, sitemaps aplikasi dan mockup user interface.

4. BAB IV, IMPLEMENTASI

Bab ini menjelaskan tentang implementasi sistem menjadi program aplikasi berbasis GUI dengan menggunakan bahasa pemrograman Java.

5. BAB V, PENUTUP

Bab ini berisi tentang kesimpulan dari pembuatan sistem aplikasi dan saran – saran dari penulis tentang aplikasi ini.

BAB II

LANDASAN TEORI

2.1. Pengertian Cuci Kendaraan

Cuci Kendaraan adalah fasilitas pelayanan publik yang didirikan oleh pengusaha, yang bertujuan untuk memberikan layanan pencucian kendaraan ke pelanggan yang membutuhkan. Kendaraan yang dapat dicuci meliputi kendaraan bermotor seperti mobil, motor, dll.

2.2. Konsep Dasar Sistem

Sistem adalah sekumpulan atau himpunan dari unsur, komponen, atau variabel yang saling terorganisir, saling berinteraksi, dan saling bergantung satu sama lainnya. Komponen-komponen sistem dapat berupa sub-sistem yang mempunyai sifat-sifat dari sistem untuk menjalankan fungsi tertentu dan mempengaruhi proses sistem secara keseluruhan.

2.3. Pengertian Java

Java adalah bahasa pemrograman berorientasi objek yang dikembangkan oleh *Sun Microsystems* sejak tahun 1991. Bahasa ini dikembangkan dengan model yang mirip dengan bahasa C++ dan Smalltalk, namun dirancang agar lebih mudah dipakai dan platform independent, yaitu dapat dijalankan di berbagai jenis sistem operasi dan arsitektur komputer. Java digunakan untuk membangun suatu sistem yang berbasis desktop sampai dengan mobile.

Java sebagai sebuah platform dibagi menjadi beberapa bagian, antara lain :

1. Bahasa Pemograman Java adalah suatu bahasa yang murni Object Oriented Programming. Semua kriteria OOP pada Java antara lain abstraksi data dan enkapsulasi, Inheritance dan Polimorfism.
2. JVM (Java Virtual Machine) Dengan JVM ini maka semboyan Java yaitu “write once run everywhere” dapat direalisasikan dimana dengan JVM, suatu program tidak lagi tergantung pada terhadap platform OS yang digunakan dan berinteraksi dengan OS. Kompilasi terhadap suatu file Java (source) akan menghasilkan suatu file byte code (extention class) dimana byte code Java adalah sama untuk semua platform, sehingga ketika menjalankan sebuah program Java. JVM akan

menangani segala sesuatu yang berhubungan dengan OS dan menjalankan byte code yang telah dihasilkan.

3. Java Basic API (J2SDK) Java Basic API adalah sekumpulan class yang disediakan oleh Java untuk melakukan proses pengembangan terhadap aplikasi Java.

2.4. Pengertian Netbeans

Netbeans adalah aplikasi IDE (*Integrated Development Environment*) yang berbasiskan Java. Menurut laman web resmi Netbeans, “NetBeans IDE adalah sebuah alat pengembangan untuk menulis program, mengompilasi, mencari kesalahan dan mengembangkan program. Netbeans IDE ditulis dalam Bahasa Pemrograman Java, dan juga dapat mendukung bahasa pemrograman lain”.

2.5. Pengertian Database

Basis Data terdiri dari kata basis dan data. Basis dapat diartikan sebagai markas atau gudang. Sedangkan data adalah catatan atas kumpulan fakta dunia nyata yang mewakili objek. Secara kesatuan, pengertian basis data adalah kumpulan data dalam bentuk file/tabel/arsip yang saling berhubungan dan tersimpan dalam media penyimpanan elektronis, untuk kemudahan dalam pengaturan, pemilahan, pengelompokan dan pengorganisasian data sesuai tujuan.

BAB III

MODEL DESKRIPSI SISTEM

3.1. Studi Kasus

Aplikasi manajemen cuci kendaraan adalah aplikasi untuk memudahkan petugas yang ada pada ruang lingkup kerja tersebut. Hal yang dapat dilakukan oleh aplikasi ini antara lain adalah pendataan daftar harga pencucian, pendataan akun, dan transaksi.

3.2. Hak Akses Akun

Dalam aplikasi yang penulis buat, terdapat beberapa hak akses yaitu :

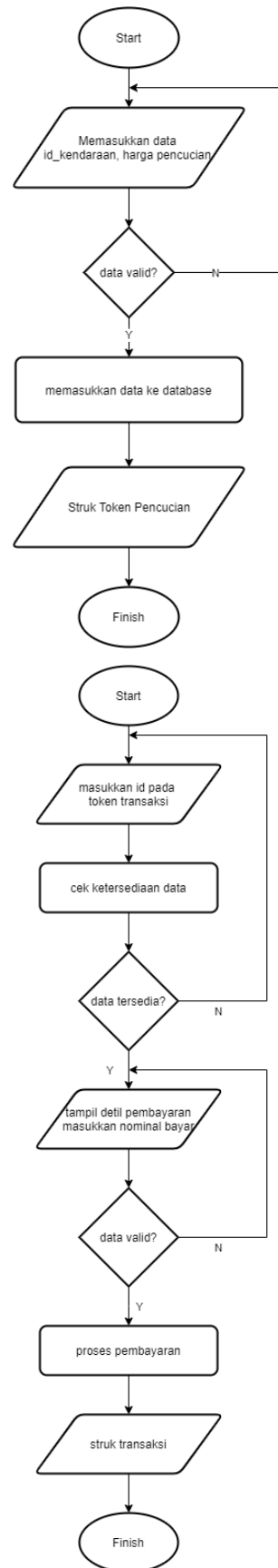
1. Adminstrator

Administrator memiliki akses ke form Akun, Jenis Kendaraan, dimana dalam form ini admin dapat menambahkan, mengedit dan menghapus data yang terdapat pada form tersebut, dan juga administrator dapat mengelola data transaksi, dimana dalam hal ini administrator dapat membuat data transaksi baru, dan mengedit transaksi yang telah dilakukan oleh akun dengan hak akses user.

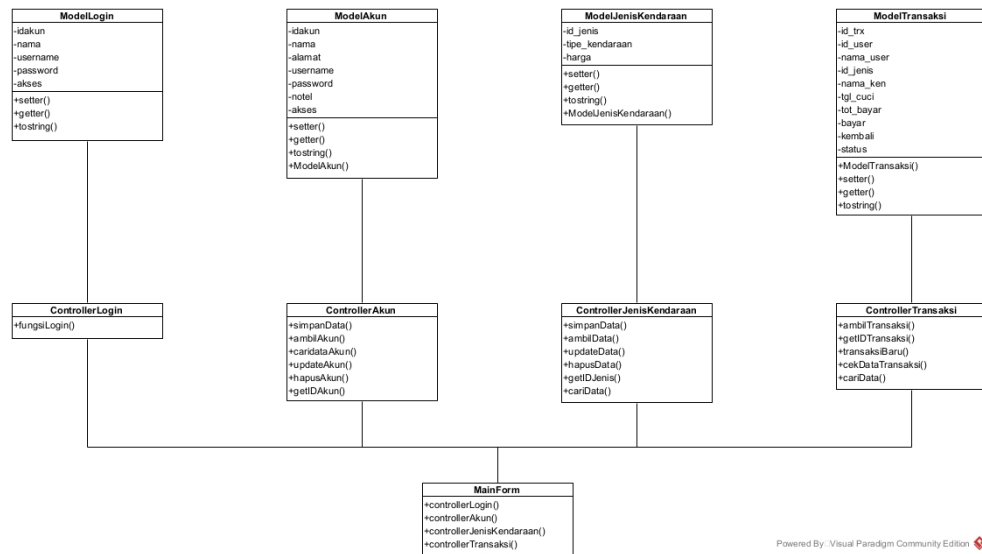
2. User

User memiliki akses ke pembuatan transaksi baru, tetapi dalam hal ini transaksi yang dibuat dengan hak akses user hanya mengirim data yang hanya diperlukan saja. Lalu akun dengan hak akses user dapat melihat kembali riwayat transaksi yang pernah dilakukan oleh akun yang bersangkutan.

3.3. Alur Proses



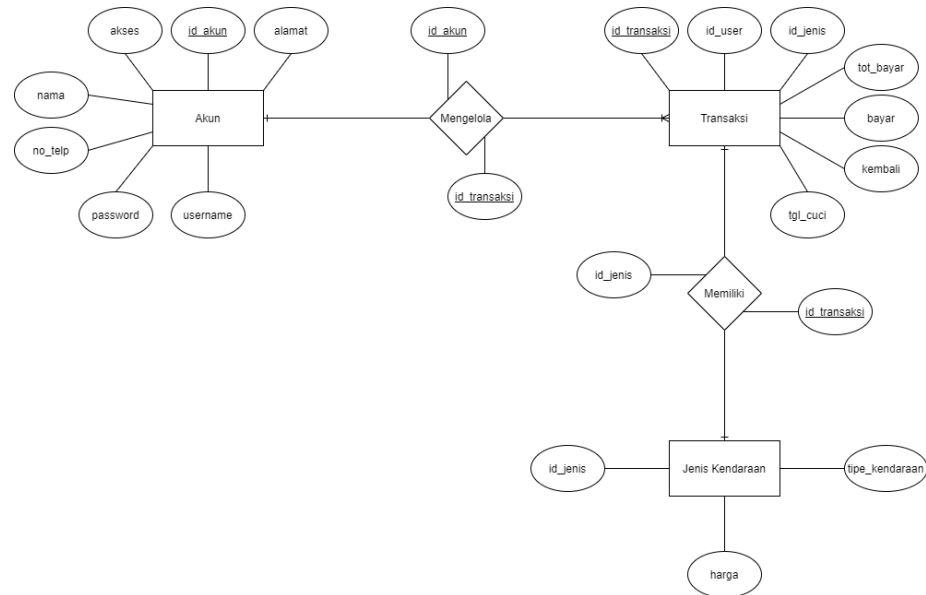
3.4. Class Diagram



3.5. Database

1. Entity Relationship Diagram

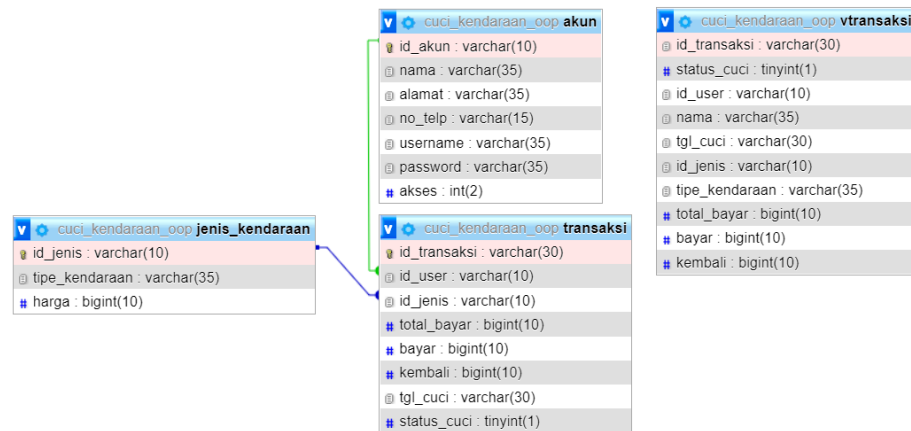
Entity Relationship Diagram (ERD) adalah sebuah model diagram yang menjelaskan hubungan antar objek pada basis data yang mempunyai hubungan antar relasi.



2. Table Relationship Diagram

TRD (Table Relationship Diagram) adalah suatu model untuk menjelaskan hubungan antar data dalam basis data berdasarkan objek-objek dasar data yang mempunyai hubungan antar relasi yang berbentuk

tabel dan memiliki atributnya masing-masing yang saling terhubung dengan atribut di tabel yang lain.



3. Data Dictionary System

Kamus data (data dictionary) adalah suatu penjelasan tertulis tentang suatu data yang berada di dalam database atau suatu daftar data elemen yang terorganisir dengan definisi yang tetap dan sesuai dengan sistem, sehingga user dan analis sistem mempunyai pengertian yang sama tentang input, output, dan komponen data store.

1 akun

Pembuatan: 07 Jun 2021 pada 13:02
Pembaruan terakhir: 06 Jun 2021 pada 22:39

Kolom	Jenis	Atribut	Ternak	laBawaan	Ekstra	Tautan ke	Komentar	MIME
id_akun	varchar(10)			Tidak			ID unik untuk setiap akun. Memiliki Relasi dengan tabel transaksi Range : AKN-001 - AKN-999	
nama	varchar(35)			Tidak			Nama Pengguna Example : Udin	
alamat	varchar(35)			Tidak			Alamat Pengguna Example : Bandung	
no_telp	varchar(15)			Tidak			No Telp Pengguna Example : 081355446644	
username	varchar(35)			Tidak			username unik pengguna digunakan untuk akses login & tidak boleh sama dengan akun lain Example : udinsedunia	
password	varchar(35)			Tidak			password dari setiap akun Example : uuuw	
akses	int(2)			Tidak			Definisi hak akses dari aplikasi Range : 1 = Admin, 2 = User	

2 jenis_kendaraan

Pembuatan: 07 Jun 2021 pada 13:05

Kolom	Jenis	Atribut	Ternak	laBawaan	Ekstra	Tautan ke	Komentar	MIME
id_jenis	varchar(10)			Tidak			ID Unik dari setiap jenis kendaraan Memiliki Relasi dengan tabel transaksi Range : KEN-001 to KEN-999	
tipe_kendaraan	varchar(35)			Tidak			Keterangan tipe kendaraan dari setiap data yang dibuat Example : Mobil	
harga	bigint(10)			Tidak			keterangan harga dari setiap data yang dibuat Example : 99500	

3 transaksi

Pembuatan: 07 Jun 2021 pada 13.10
 Pembaruan terakhir: 07 Jun 2021 pada 12.42

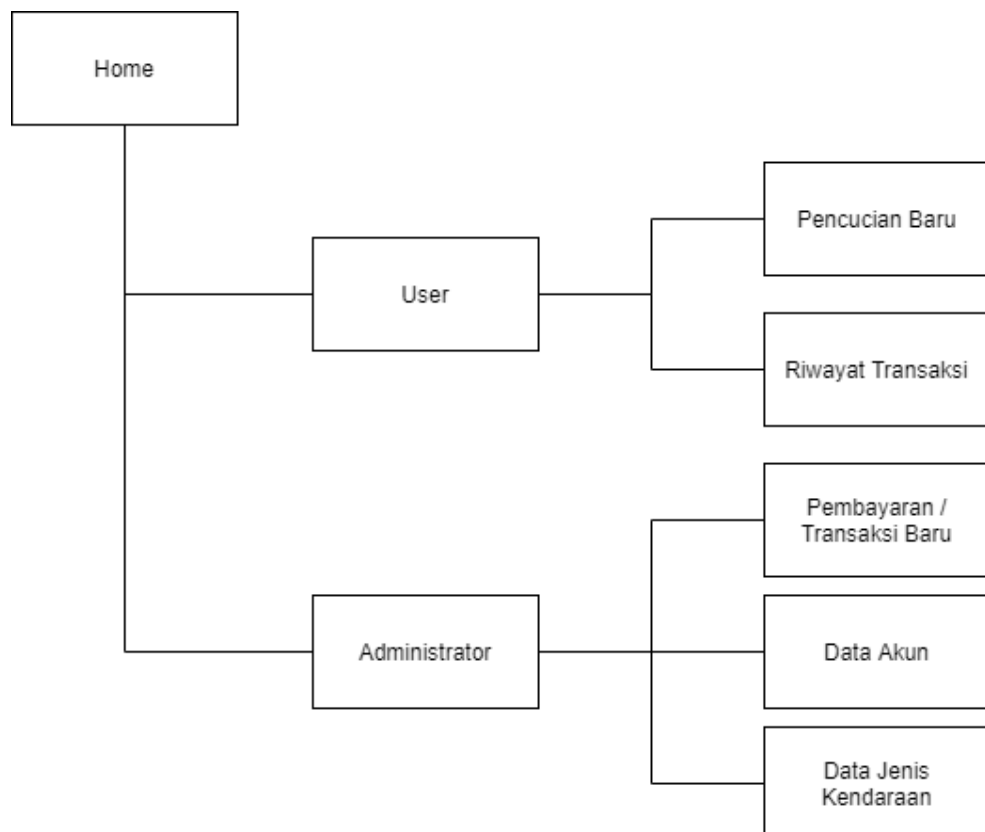
Kolom	Jenis	Atribut	Ternak	laBawaan	Ekstra	Tautan ke	Komentar	MIME
id_transaksi	varchar(30)		Tidak				id unik dari setiap transaksi Structure : TRX-DDHHMMSSIDunique	
id_user	varchar(10)		Tidak			-> akun_id_akun ON UPDATE NO_ACTION ON DELETE NO_ACTION	id akun dari tabel akun	
id_jenis	varchar(10)		Tidak			-> jenis_kendaraan.id_jenis ON UPDATE NO_ACTION ON DELETE NO_ACTION	id jenis kendaraan dari tabel kendaraan	
total_bayar	bigint(10)		Tidak				total pembayaran dari sebuah transaksi	
bayar	bigint(10)		Ya	NULL			uang yang dibayarkan, data kosong jika transaksi belum selesai	
kembali	bigint(10)		Ya	NULL			uang kembalian dari pembayaran, data kosong jika transaksi belum selesai	
tgl_cuci	varchar(30)		Ya	NULL			tanggal pencucian, data kosong jika transaksi belum selesai Format : yyyy/dd/mm hh:mm:ss	
status_cuci	tinyint(1)		Ya	NULL			status pencucian range : 1 = selesai, 0 = belum selesai	

4 vtransaksi

Komentar tabel: VIEW

Kolom	Jenis	Atribut	Ternak	laBawaan	Ekstra	Tautan ke	Komentar	MIME
id_transaksi	varchar(30)		Tidak				id unik dari setiap transaksi Structure : TRX-DDHHMMSSIDunique	
status_cuci	tinyint(1)		Ya	NULL			status pencucian range : 1 = selesai, 0 = belum selesai	
id_user	varchar(10)		Tidak				id akun dari tabel akun	
nama	varchar(35)		Tidak				Nama Pengguna Example : Udin	
tgl_cuci	varchar(30)		Ya	NULL			tanggal pencucian, data kosong jika transaksi belum selesai Format : yyyy/dd/mm hh:mm:ss	
id_jenis	varchar(10)		Tidak				id jenis kendaraan dari tabel kendaraan	
tipe_kendaraan	varchar(35)		Tidak				Keterangan tipe kendaraan dari setiap data yang dibuat Example : Mobil	
total_bayar	bigint(10)		Tidak				total pembayaran dari sebuah transaksi	
bayar	bigint(10)		Ya	NULL			uang yang dibayarkan, data kosong jika transaksi belum selesai	
kembali	bigint(10)		Ya	NULL			uang kembalian dari pembayaran, data kosong jika transaksi belum selesai	

3.6. Sitemaps



3.7. Mockup UI

myVWash

Menu Utama

Masukkan Kridensial Anda Untuk Melanjutkan

Username

Password

Keluar

Login

myVWash

Menu Utama

Selamat Datang, Nama Akun

Data Akun

Data Jenis Kendaraan

Transaksi (Administrator)

Transaksi

Riwayat Transaksi

Keluar

Nama Form

Field 1

Field 2

Field 3

Field 4

Field 5

Tambah

Batal

Simpan

Edit

Hapus

Cari Data

Tabel

Nama Form

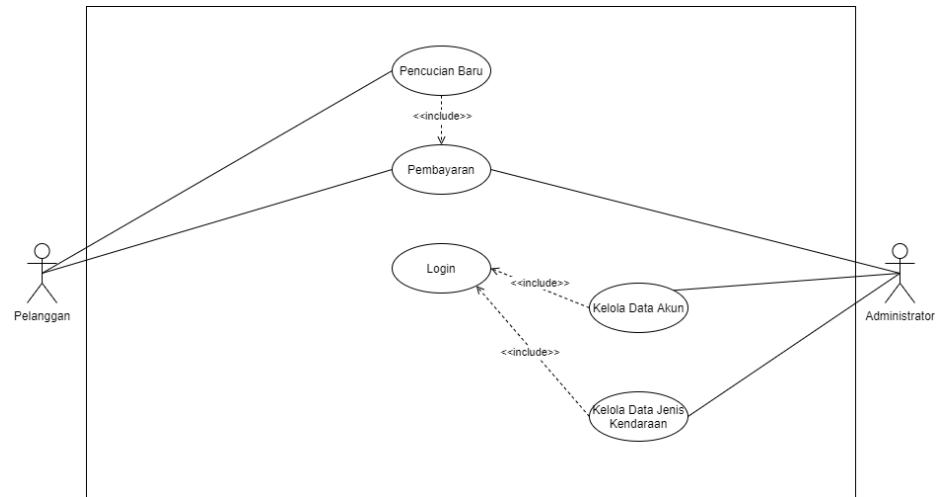
Tabel

3.8. Unified Modelling Language

UML adalah kependekan dari Unified Modeling Language. Sederhananya, UML dapat diartikan sebagai pendekatan modern untuk pemodelan dan mendokumentasikan perangkat lunak.

1. Use Case Diagram

Use Case diagram adalah suatu urutan interaksi yang saling berkaitan antara sistem dan aktor. Use case dijalankan melalui cara menggambarkan tipe interaksi antara user suatu program (sistem) dengan sistemnya sendiri.



BAB IV

IMPLEMENTASI

4.1. User Interface dan Listing Code

1. ConnectionManager.java

```
package Connection;

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

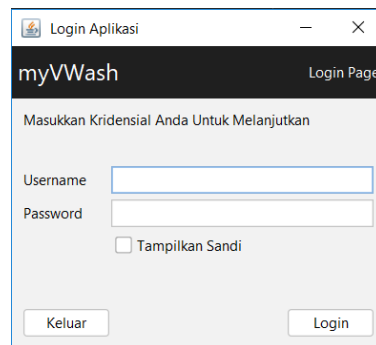
public class ConnectionManager {
    private Connection con;
    private String driver = "com.mysql.jdbc.Driver";
    private String url =
"jdbc:mysql://localhost:3306/cuci_kendaraan_oop";
    private String username = "root";
    private String password = "";

    public Connection logOn(){
        try{
            //Load JDBC Driver
            Class.forName(driver).newInstance();
            //Buat object connection
            con = DriverManager.getConnection(url, username,
password);
            System.out.println(con);
        }
        catch(Exception ex){
            ex.printStackTrace();
        }
        return con;
    }

    public void logOff(){
        try{
            //Tutup koneksi
            con.close();
        }
        catch(Exception ex){
            ex.printStackTrace();
        }
    }
}
```

2. Halaman Login

- MainLogin.java



- ModelLogin.java

```
package Model;

public class ModelLogin {
    private static String id_akun, nama, alamat, no_telp,
username, password;
    private static int access;

    public static String getId_akun() {
        return id_akun;
    }

    public static void setId_akun(String id_akun) {
        ModelLogin.id_akun = id_akun;
    }

    public static String getNama() {
        return nama;
    }

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

    public static String getAlamat() {
        return alamat;
    }

    public static void setAlamat(String alamat) {
        ModelLogin.alamat = alamat;
    }
}
```

```

public static String getNo_telp() {
    return no_telp;
}

public static void setNo_telp(String no_telp) {
    ModelLogin.no_telp = no_telp;
}

public static String getUsername() {
    return username;
}

public static void setUsername(String username) {
    ModelLogin.username = username;
}

public static String getPassword() {
    return password;
}

public static void setPassword(String password) {
    ModelLogin.password = password;
}

public static int getAccess() {
    return access;
}

public static void setAccess(int access) {
    ModelLogin.access = access;
}
}

```

- **ControllerTransaksi.java**

```

package Controller;

import Connection.ConnectionManager;
import Model.ModelTransaksi;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Date;

```

```

import java.util.List;
import java.util.logging.Level;

public class ControllerTransaksi {
    private ConnectionManager conMan;
    private Connection con;
    private Statement st;
    private ResultSet rs;

    public List<ModelTransaksi> ambilTransaksi(int param,
ModelTransaksi trx) {

        /*
            param = 0 -> no filtered data -> ignore
modeltransaksi
            param = 1 -> filtered data -> read
modeltransaksi
        */

        List<ModelTransaksi> lsTransaksi = new
ArrayList<ModelTransaksi>();
        String query;

        if (param == 1) {
            query = "SELECT * FROM vtransaksi WHERE
id_user LIKE '%" + trx.getId_user() + "%' ORDER BY
id_transaksi ASC";
        } else {
            query = "SELECT * FROM vtransaksi ORDER BY
id_transaksi ASC";
        }

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(query);

            while (rs.next()) {
                ModelTransaksi mt = new ModelTransaksi();

mt.setId_trx(rs.getString("id_transaksi"));

```

```

        mt.setStatus(rs.getString("status_cuci"));
        mt.setId_user(rs.getString("id_user"));
        mt.setNama_user(rs.getString("nama"));
        mt.setTgl_cuci(rs.getString("tgl_cuci"));
        mt.setId_jenis(rs.getString("id_jenis"));

mt.setNama_ken(rs.getString("tipe_kendaraan"));

mt.setTot_bayar(rs.getString("total_bayar"));
        mt.setBayar(rs.getString("bayar"));
        mt.setKembali(rs.getString("kembali"));

        lsTransaksi.add(mt);
    }

    conMan.logOff();
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }

    return lsTransaksi;
}

public String getIDTransaksi() {
    String idakun = null, ran2;
    int ran1;
    String SQL = "SELECT RIGHT (id_transaksi, 2) AS
id_transaksi FROM transaksi";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    Date date = new Date();
    ran1 = date.getDate();
    ran2 =
date.getHours()+" "+date.getMinutes()+" "+date.getSeconds();

    try {
        st = con.createStatement();
        rs = st.executeQuery(SQL);
        if (rs.first() == false) {

```

```

        idakun = "TRX-0"+ ran1 + "" + ran2 +"01";
    } else {
        rs.last();
        int no = rs.getInt(1) + 1;
        String cno = String.valueOf(no);
        int pjg_cno = cno.length();
        for (int i =0;1<2-pjg_cno; i++){
            cno="0" +cno;
        }
        idakun = "TRX-0"+ ran1 + "" + ran2 +
""+cno;
    }
    conMan.logOff();
}
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
}
return idakun;
}

public int transaksiBaru(ModelTransaksi mt, int mode)
{
    /*
        mode = 1 : mode new data from user
        mode = 2 : mode new data from admin
        mode = 3 : modify data from admin / continue
transaction
    */

    int hasil = 0;
    String query = null;

    switch (mode) {
        case 1:
            query = "INSERT INTO
`transaksi`(`id_transaksi`, `id_user`, `id_jenis`,
`total_bayar`, `status_cuci`)
                + " VALUES ('"+ mt.getId_trx()
+"','"+ mt.getId_user() +"','"+ mt.getId_jenis() +"','"+
mt.getTot_bayar() +"','"+ mt.getStatus() +"')";

```

```

        break;
        case 2:
            query = "INSERT INTO
`transaksi`(`id_transaksi`, `id_user`, `id_jenis`,
`total_bayar`, `bayar`, `kembali`, `tgl_cuci`,
`status_cuci`)"
                    + " VALUES ('"+ mt.getId_trx()
+"', '"+ mt.getId_user() +"' , '"+ mt.getId_jenis() +"' , '"+
mt.getTot_bayar() +"' , '"+ mt.getBayar() +"' , "
                    + "'"+ mt.getKembali() +"' , '"+
mt.getTgl_cuci() +"' , '"+ mt.getStatus() +"' )";
            break;
        case 3:
            query = "UPDATE `transaksi` SET
`bayar`='"+ mt.getBayar() +"' , `kembali`='"+
mt.getKembali() +"' , `tgl_cuci`='"+ mt.getTgl_cuci() +"' , "
                    + "`status_cuci`='"+
mt.getStatus() +"' WHERE id_transaksi LIKE '%" +
mt.getId_trx() + "%'";
            break;
        default:
            break;
    }

    try {
        conMan = new ConnectionManager();
        con = conMan.logOn();

        st = con.createStatement();
        hasil = st.executeUpdate(query);
        conMan.logOff();
    } catch (SQLException ex) {

        java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }

    return hasil;
}

public int cekDataTransaksi(ModelTransaksi mt) {
    int hasil = 0;

```



```

        String query = "SELECT * FROM transaksi WHERE
id_transaksi = '"+ mt.getId_trx() +"'";

        try {
            conMan = new ConnectionManager();
            con = conMan.logOn();

            st = con.createStatement();
            rs = st.executeQuery(query);

            if (rs.next()) {
                hasil = 1;
            }

            conMan.logOff();
        } catch (Exception ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
        }

        return hasil;
    }

    public List<ModelTransaksi> cariData(ModelTransaksi
mt) {
        List<ModelTransaksi> lsTrx = new
ArrayList<ModelTransaksi>();
        String query = "SELECT * FROM vtransaksi WHERE
id_transaksi LIKE '%" + mt.getId_trx() + "%'";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(query);

            while (rs.next()) {
                ModelTransaksi mta = new ModelTransaksi();
mta.setId_trx(rs.getString("id_transaksi"));
                mta.setId_user(rs.getString("id_user"));
            }
        }
    }

```

```

        mta.setNama_user(rs.getString("nama"));
        mta.setId_jenis(rs.getString("id_jenis"));

        mta.setNama_ken(rs.getString("tipe_kendaraan"));

        mta.setTot_bayar(rs.getString("total_bayar"));
        mta.setBayar(rs.getString("bayar"));
        mta.setKembali(rs.getString("kembali"));

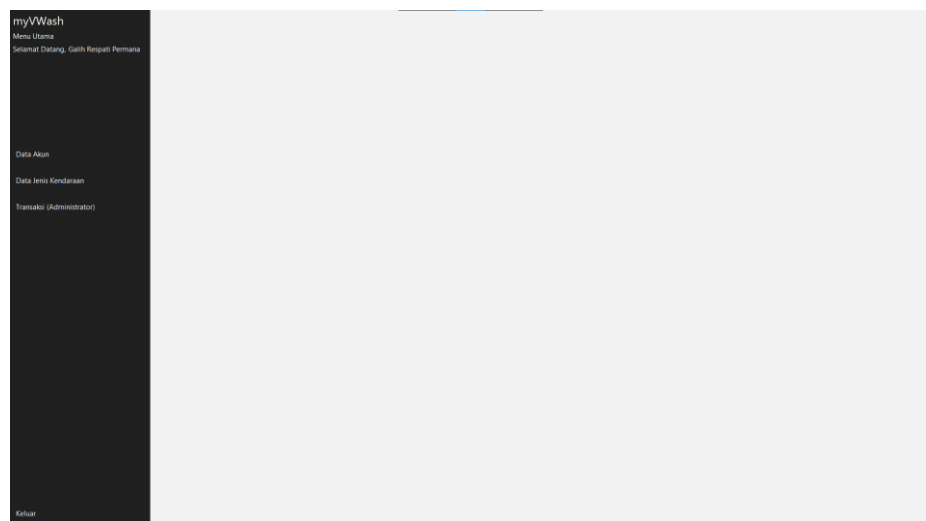
        mta.setStatus(rs.getString("status_cuci"));

        lsTrx.add(mta);
    }
    conMan.logOff();
} catch (SQLException ex) {

    java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
}
return lsTrx;
}
}

```

3. Halaman Menu Utama



4. Halaman Akun

- MainAkun.java

The screenshot shows the 'myVWash' web application interface. On the left is a dark sidebar with navigation links: 'Menu Utama', 'Selamat Datang, Galih Ruspati Permata', 'Data Akun', 'Data Jenis Kendaraan', and 'Transaksi (Administrator)'. The main content area is titled 'Form Akun' and contains a form for adding a new account. The form fields are: 'ID Akun' (with value 'AKN-005'), 'Nama Pengguna', 'Alamat', 'No Telepon', 'Username', 'Password', and 'Hak Akses' (a dropdown menu). Below the form are buttons: 'Tambah', 'Batal', 'Simpan', 'Edit', and 'Hapus'. To the right of the form is a table showing existing accounts.

ID Akun	Nama Pengguna	Alamat	No Telepon	Username	Password	Level Akses
AKN-001	Galih Ruspati Perma...	Ujungberung	0812223334455	admin	admin	1
AKN-002	Udin	Bandung	08134556677	user	user	2
AKN-003	Demi	Bandung	08126389503	demi	demi	2
AKN-004	Ucup	Bandung	081336472948	ucup	ucup	2

- ModelAkun.java

```
package Model;

public class ModelAkun {
    private String idakun;
    private String nama;
    private String alamat;
    private String username;
    private String password;
    private String notel;
    private int akses;

    public ModelAkun() {
    }

    public ModelAkun(String idakun, String nama, String
alamat, String username, String password, String notel,
int akses) {
        this.idakun = idakun;
        this.nama = nama;
        this.alamat = alamat;
        this.username = username;
        this.password = password;
        this.notel = notel;
        this.akses = akses;
    }

    public String getIdakun() {
```

```
        return idakun;
    }

    public void setIdakun(String idakun) {
        this.idakun = idakun;
    }

    public String getNama() {
        return nama;
    }

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

    public String getAlamat() {
        return alamat;
    }

    public void setAlamat(String alamat) {
        this.alamat = alamat;
    }

    public String getUsername() {
        return username;
    }

    public void setUsername(String username) {
        this.username = username;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

    public String getNotel() {
        return notel;
    }
}
```

```

    public void setNotel(String notel) {
        this.notel = notel;
    }

    public int getAkses() {
        return akses;
    }

    public void setAkses(int akses) {
        this.akses = akses;
    }

    @Override
    public String toString() {
        return "ModelAkun{" + "idakun=" + idakun + ",
nama=" + nama + ", alamat=" + alamat + ", username=" +
username + ", password=" + password + ", notel=" + notel +
", akses=" + akses + '}';
    }
}

```

- ControllerAkun.java

```

package Controller;

import Connection.ConnectionManager;
import Model.ModelAkun;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;

public class ControllerAkun {
    private ConnectionManager conMan;
    private Connection con;
    private Statement st;
    private ResultSet rs;

    //Insert Data Akun
    public int simpanData(ModelAkun akun) {
        int hasil = 0;
    }
}

```

```

        String query = "INSERT INTO akun(id_akun, nama,
alamat, no_telp, username, password, akses) "
            + "VALUES ('"+ akun.getIdakun() +"', '"+
akun.getNama() +"', '"+ akun.getAlamat() +"', '"
            + akun.getNotel() +"', '"+
akun.getUsername() +"', '"+ akun.getPassword() +"', '"+
akun.getAkses() +"')";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            hasil = st.executeUpdate(query);
            conMan.logOff();
        } catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
        }

        return hasil;
    }

    public List<ModelAkun> ambilAkun(int parameter) {
        List<ModelAkun> lsAkun = new
ArrayList<ModelAkun>();
        String query;

        if (parameter == 1) {
            query = "SELECT * FROM akun WHERE akses =
'2'";
        } else {
            query = "SELECT * FROM akun";
        }

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(query);

```

```

        while (rs.next()) {
            ModelAkun ma = new ModelAkun();
            ma.setIdakun(rs.getString("id_akun"));
            ma.setNama(rs.getString("nama"));
            ma.setAlamat(rs.getString("alamat"));
            ma.setNotel(rs.getString("no_telp"));
            ma.setUsername(rs.getString("username"));
            ma.setPassword(rs.getString("password"));
            ma.setAkses(rs.getInt("akses"));

            lsAkun.add(ma);
        }
        conMan.logOff();
    } catch (SQLException ex) {

        java.util.logging.Logger.getLogger(ControllerAkun.class.getName()).log(Level.SEVERE, null, ex);
    }
    return lsAkun;
}

public List<ModelAkun> caridataAkun(ModelAkun akun) {
    List<ModelAkun> lsAkun = new
    ArrayList<ModelAkun>();
    String query = "SELECT * FROM `akun` WHERE nama
    LIKE '%" + akun.getNama() + "%'";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    try {
        st = con.createStatement();
        rs = st.executeQuery(query);

        while (rs.next()) {
            ModelAkun ma = new ModelAkun();
            ma.setIdakun(rs.getString("id_akun"));
            ma.setNama(rs.getString("nama"));
            ma.setAlamat(rs.getString("alamat"));
            ma.setNotel(rs.getString("no_telp"));
            ma.setUsername(rs.getString("username"));
            ma.setPassword(rs.getString("password"));

```

```

        ma.setAkses(rs.getInt("akses"));

        lsAkun.add(ma);
    }
    conMan.logOff();
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
}
return lsAkun;
}

public int updateAkun(ModelAkun akun) {
    int hasil = 0;
    String query = "UPDATE akun SET nama='"+
akun.getNama() + "',alamat='"+ akun.getAlamat()
+ "',no_telp='"+ akun.getNotel() + "'"
        + "',username='"+ akun.getUsername()
+ "',password='"+ akun.getPassword() + "',akses='"+
akun.getAkses() + "'"
        + "' WHERE id_akun='"+ akun.getIdakun()
+ "'";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    try {
        st = con.createStatement();
        hasil = st.executeUpdate(query);
        conMan.logOff();
    } catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }

    return hasil;
}

public int hapusAkun(ModelAkun akun) {
    int hasil = 0;

```



```

        String query = "DELETE FROM akun WHERE id_akun = '" + akun.getIdakun() + "'";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            hasil = st.executeUpdate(query);
            conMan.logOff();
        } catch (SQLException ex) {

            java.util.logging.Logger.getLogger(ControllerAkun.class.getName()).log(Level.SEVERE, null, ex);
        }

        return hasil;
    }

    public String getIDAkun() {
        String idakun = null;
        String SQL = "SELECT RIGHT (id_akun, 2) AS id_akun FROM akun";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(SQL);
            if (rs.first() == false) {
                idakun = "AKN-001";
            } else {
                rs.last();
                int no = rs.getInt(1) + 1;
                String cno = String.valueOf(no);
                int pjg_cno = cno.length();
                for (int i = 0; i < 2 - pjg_cno; i++) {
                    cno = "00" + cno;
                }
                idakun = "AKN-" + cno;
            }
        }
    }

```

```

        conMan.logOff();
    }
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
}
return idakun;
}
}

```

5. Halaman Jenis Kendaraan

- MainJenisKendaraan.java

ID Jenis	Jenis Kendaraan	Harga
KEN-001	Motor < 250cc	15000
KEN-002	Motor 250cc s.d 600cc	20000
KEN-003	Motor > 600cc	25000
KEN-004	Mobil - Small Size	36000
KEN-005	Mobil - Mid Size	41000
KEN-006	Mobil - Large Size	46000
KEN-007	Truck	37500
KEN-008	Sepeda	10000

- ModelJenisKendaraan.java

```

/*
 * To change this license header, choose License Headers
in Project Properties.
 * To change this template file, choose Tools | Templates
 * and open the template in the editor.
 */
package Model;

/**
 *
 * @author Galih Respati P
 */
public class ModelJenisKendaraan {
    private String id_jenis, tipe_kendaraan;
    private int harga;
}

```

```

    public ModelJenisKendaraan() {
    }

    public ModelJenisKendaraan(String id_jenis, String
    tipe_kendaraan, int harga) {
        this.id_jenis = id_jenis;
        this.tipe_kendaraan = tipe_kendaraan;
        this.harga = harga;
    }

    public String getId_jenis() {
        return id_jenis;
    }

    public void setId_jenis(String id_jenis) {
        this.id_jenis = id_jenis;
    }

    public String getTipe_kendaraan() {
        return tipe_kendaraan;
    }

    public void setTipe_kendaraan(String tipe_kendaraan) {
        this.tipe_kendaraan = tipe_kendaraan;
    }

    public int getHarga() {
        return harga;
    }

    public void setHarga(int harga) {
        this.harga = harga;
    }

    @Override
    public String toString() {
        return "ModelJenisKendaraan{" + "id_jenis=" +
        id_jenis + ", tipe_kendaraan=" + tipe_kendaraan + ",
        harga=" + harga + '}';
    }
}

```

- ControllerJenisKendaraan.java

```

package Controller;

import Connection.ConnectionManager;
import Model.ModelJenisKendaraan;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import java.util.logging.Level;

public class ControllerJenisKendaraan {
    private ConnectionManager conMan;
    private Connection con;
    private Statement st;
    private ResultSet rs;

    public int simpanData(ModelJenisKendaraan jenis) {
        int hasil = 0;
        String query = "INSERT INTO
jenis_kendaraan(id_jenis, tipe_kendaraan, harga) "
            + "VALUES ('"+ jenis.getId_jenis()+"', '"+
jenis.getTipe_kendaraan() +"' , '"+ jenis.getHarga()+"')";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            hasil = st.executeUpdate(query);
            conMan.logOff();
        } catch (SQLException ex) {

            java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
        }

        return hasil;
    }

    public List<ModelJenisKendaraan> ambilData() {

```

```

        List<ModelJenisKendaraan> lsJk = new
        ArrayList<ModelJenisKendaraan>();

        String query = "SELECT * FROM jenis_kendaraan";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(query);

            while (rs.next()) {
                ModelJenisKendaraan jk = new
                ModelJenisKendaraan();
                jk.setId_jenis(rs.getString("id_jenis"));

                jk.setTipe_kendaraan(rs.getString("tipe_kendaraan"));
                jk.setHarga(rs.getInt("harga"));

                lsJk.add(jk);
            }
            conMan.logOff();
        } catch (SQLException ex) {

            java.util.logging.Logger.getLogger(ControllerAkun.class.ge
            tName()).log(Level.SEVERE, null, ex);
        }
        return lsJk;
    }

    public int updateData(ModelJenisKendaraan jk) {
        int hasil = 0;

        String query = "UPDATE jenis_kendaraan SET
        tipe_kendaraan = '" + jk.getTipe_kendaraan() +
        "', harga = '" + jk.getHarga() + "' WHERE
        id_jenis = '" + jk.getId_jenis() + "'";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            hasil = st.executeUpdate(query);

```

```

        conMan.logOff();
    } catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.get
tName()).log(Level.SEVERE, null, ex);
    }

    return hasil;
}

public int hapusData(ModelJenisKendaraan jk) {
    int hasil = 0;
    String query = "DELETE FROM jenis_kendaraan WHERE
id_jenis = '"+ jk.getId_jenis() +"'";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    try {
        st = con.createStatement();
        hasil = st.executeUpdate(query);
        conMan.logOff();
    } catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.get
tName()).log(Level.SEVERE, null, ex);
    }

    return hasil;
}

public String getIDJenis() {
    String id = null;
    String SQL = "SELECT RIGHT (id_jenis, 2) AS
id_jenis FROM jenis_kendaraan";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    try {
        st = con.createStatement();
        rs = st.executeQuery(SQL);
        if (rs.first() == false) {

```

```

        id = "KEN-001";
    } else {
        rs.last();
        int no = rs.getInt(1) + 1;
        String cno = String.valueOf(no);
        int pjg_cno = cno.length();
        for (int i = 0; i < 2 - pjg_cno; i++) {
            cno = "00" + cno;
        }
        id = "KEN-" + cno;
    }
    conMan.logOff();
}
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.getName()).log(Level.SEVERE, null, ex);
}
return id;
}

public List<ModelJenisKendaraan>
cariData(ModelJenisKendaraan mk) {
    List<ModelJenisKendaraan> lsJk = new
    ArrayList<ModelJenisKendaraan>();

    String query = "SELECT * FROM jenis_kendaraan
    WHERE tipe_kendaraan LIKE '%" + mk.getTipe_kendaraan()
    + "%'";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    try {
        st = con.createStatement();
        rs = st.executeQuery(query);

        while (rs.next()) {
            ModelJenisKendaraan jk = new
            ModelJenisKendaraan();
            jk.setId_jenis(rs.getString("id_jenis"));
            jk.setTipe_kendaraan(rs.getString("tipe_kendaraan"));

```

```

        jk.setHarga(rs.getInt("harga"));

        lsJk.add(jk);
    }
    conMan.logOff();
} catch (SQLException ex) {

    java.util.logging.Logger.getLogger(ControllerAkun.class.get
    tName()).log(Level.SEVERE, null, ex);
}
return lsJk;
}
}

```

6. Halaman Transaksi (Administrator)

- MainTransaksiAdmin.java

ID Transaksi	Status Pencucian	Pelanggan	Tanggal Cuci	Jenis Kendaraan	Total Pembayaran	Bayar	Kembali
TRX-012254601	Selesai	AKN-002 Udin	2021/06/01 22.0.	KEN-008 Sepeda	10000	20000	10000
TRX-012261102	Selesai	AKN-004 Ucup	2021/06/01 22.0.	KEN-007 Truck	37500	40000	2500
TRX-0216255603	Selesai	AKN-003 Udin	2021/18/02 17.1.	KEN-002 Motor...	20000	0	20000
TRX-021725703	Selesai	AKN-003 Demi	2021/13/02 17.3.	KEN-006 Mobil -	46000	50000	4000
TRX-021841303	Selesai	AKN-002 Udin	2021/14/02 18.4.	KEN-007 Truck	37500	40000	2500
TRX-021841403	Selesai	AKN-003 Demi	2021/14/02 18.4.	KEN-004 Mobil -	36000	40000	4000
TRX-0218521403	Selesai	AKN-004 Ucup	2021/12/02 18.5.	KEN-008 Sepeda	10000	10000	0
TRX-021854204	Selesai	AKN-002 Udin	2021/13/03 16.1.	KEN-002 Motor...	20000	20000	0
TRX-0315383004	Belum Selesai	AKN-002 Udin		KEN-001 Motor...	15000		
TRX-0315405304	Belum Selesai	AKN-004 Ucup		KEN-002 Motor...	20000		
TRX-0315433005	Belum Selesai	AKN-003 Demi		KEN-003 Motor...	25000		
TRX-0315481405	Belum Selesai	AKN-002 Udin		KEN-004 Mobil -	36000		
TRX-0315541505	Belum Selesai	AKN-002 Udin		KEN-005 Mobil -	41000		
TRX-0315571105	Belum Selesai	AKN-002 Udin		KEN-003 Motor...	25000		
TRX-0315591305	Belum Selesai	AKN-002 Udin		KEN-006 Mobil -	46000		
TRX-031602205	Selesai	AKN-002 Udin	2021/14/03 16.1.	KEN-001 Motor...	15000	50000	35000
TRX-031613205	Selesai	AKN-002 Udin	2021/13/03 16.1.	KEN-005 Mobil -	41000	45000	4000
TRX-041594605	Selesai	AKN-002 Udin	2021/10/04 15.1.	KEN-005 Mobil -	41000	41000	0

- ModelTransaksi.java

```

package Model;

public class ModelTransaksi {
    private String id_trx, id_user, nama_user, id_jenis,
    nama_ken, tgl_cuci, tot_bayar, bayar, kembali, status;

    public ModelTransaksi() {
    }

    public ModelTransaksi(String id_trx, String id_user,
    String id_jenis, String tgl_cuci, String status, String
    tot_bayar, String bayar, String kembali) {
        this.id_trx = id_trx;
    }
}

```



```

        this.id_user = id_user;
        this.id_jenis = id_jenis;
        this.tgl_cuci = tgl_cuci;
        this.status = status;
        this.tot_bayar = tot_bayar;
        this.bayar = bayar;
        this.kembali = kembali;
    }

    public ModelTransaksi(String id_trx, String id_user,
String nama_user, String id_jenis, String nama_ken, String
tgl_cuci, String status, String tot_bayar, String bayar,
String kembali) {
        this.id_trx = id_trx;
        this.id_user = id_user;
        this.nama_user = nama_user;
        this.id_jenis = id_jenis;
        this.nama_ken = nama_ken;
        this.tgl_cuci = tgl_cuci;
        this.status = status;
        this.tot_bayar = tot_bayar;
        this.bayar = bayar;
        this.kembali = kembali;
    }

    public String getId_trx() {
        return id_trx;
    }

    public void setId_trx(String id_trx) {
        this.id_trx = id_trx;
    }

    public String getId_user() {
        return id_user;
    }

    public void setId_user(String id_user) {
        this.id_user = id_user;
    }

    public String getNama_user() {
        return nama_user;
    }

```

```
}

public void setNama_user(String nama_user) {
    this.nama_user = nama_user;
}

public String getId_jenis() {
    return id_jenis;
}

public void setId_jenis(String id_jenis) {
    this.id_jenis = id_jenis;
}

public String getNama_ken() {
    return nama_ken;
}

public void setNama_ken(String nama_ken) {
    this.nama_ken = nama_ken;
}

public String getTgl_cuci() {
    return tgl_cuci;
}

public void setTgl_cuci(String tgl_cuci) {
    this.tgl_cuci = tgl_cuci;
}

public String getStatus() {
    return status;
}

public void setStatus(String status) {
    this.status = status;
}

public String getTot_bayar() {
    return tot_bayar;
}

public void setTot_bayar(String tot_bayar) {
```

```

        this.tot_bayar = tot_bayar;
    }

    public String getBayar() {
        return bayar;
    }

    public void setBayar(String bayar) {
        this.bayar = bayar;
    }

    public String getKembali() {
        return kembali;
    }

    public void setKembali(String kembali) {
        this.kembali = kembali;
    }

    @Override
    public String toString() {
        return "ModelTransaksi{" + "id_trx=" + id_trx + ",
id_user=" + id_user + ", nama_user=" + nama_user + ",
id_jenis=" + id_jenis + ", nama_ken=" + nama_ken + ",
tgl_cuci=" + tgl_cuci + ", status=" + status + ",
tot_bayar=" + tot_bayar + ", bayar=" + bayar + ",
kembali=" + kembali + '}';
    }
}

```

- **ControllerTransaksi.java**

```

package Controller;

import Connection.ConnectionManager;
import Model.ModelTransaksi;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.Date;
import java.util.List;
import java.util.logging.Level;

```

```

public class ControllerTransaksi {
    private ConnectionManager conMan;
    private Connection con;
    private Statement st;
    private ResultSet rs;

    public List<ModelTransaksi> ambilTransaksi(int param,
ModelTransaksi trx) {

        /*
            param = 0 -> no filtered data -> ignore
modeltransaksi
            param = 1 -> filtered data -> read
modeltransaksi
        */

        List<ModelTransaksi> lsTransaksi = new
ArrayList<ModelTransaksi>();
        String query;

        if (param == 1) {
            query = "SELECT * FROM vtransaksi WHERE
id_user LIKE '%" + trx.getId_user() + "%' ORDER BY
id_transaksi ASC";
        } else {
            query = "SELECT * FROM vtransaksi ORDER BY
id_transaksi ASC";
        }

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(query);

            while (rs.next()) {
                ModelTransaksi mt = new ModelTransaksi();

mt.setId_trx(rs.getString("id_transaksi"));
                mt.setStatus(rs.getString("status_cuci"));
                mt.setId_user(rs.getString("id_user"));
                mt.setNama_user(rs.getString("nama"));
            }
        }
    }
}

```

```

        mt.setTgl_cuci(rs.getString("tgl_cuci"));
        mt.setId_jenis(rs.getString("id_jenis"));

mt.setNama_ken(rs.getString("tipe_kendaraan"));

mt.setTot_bayar(rs.getString("total_bayar"));
        mt.setBayar(rs.getString("bayar"));
        mt.setKembali(rs.getString("kembali"));

        lsTransaksi.add(mt);
    }

    conMan.logOff();
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }

    return lsTransaksi;
}

public String getIDTransaksi() {
    String idakun = null, ran2;
    int ran1;
    String SQL = "SELECT RIGHT (id_transaksi, 2) AS
id_transaksi FROM transaksi";

    conMan = new ConnectionManager();
    con = conMan.logOn();

    Date date = new Date();
    ran1 = date.getDate();
    ran2 =
date.getHours()+" "+date.getMinutes()+" "+date.getSeconds();

    try {
        st = con.createStatement();
        rs = st.executeQuery(SQL);
        if (rs.first() == false) {
            idakun = "TRX-0" + ran1 + " " + ran2 + "01";
        } else {
            rs.last();

```

```

        int no = rs.getInt(1) + 1;
        String cno = String.valueOf(no);
        int pjg_cno = cno.length();
        for (int i = 0; i < 2 - pjg_cno; i++)
        {
            cno = "0" + cno;
        }
        idakun = "TRX-0" + ran1 + "" + ran2 +
"" + cno;

        }
        conMan.logOff();
    }
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllorAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }
    return idakun;
}

public int transaksiBaru(ModelTransaksi mt, int mode)
{
    /*
        mode = 1 : mode new data from user
        mode = 2 : mode new data from admin
        mode = 3 : modify data from admin / continue
transaction
    */

    int hasil = 0;
    String query = null;

    switch (mode) {
        case 1:
            query = "INSERT INTO
`transaksi`(`id_transaksi`, `id_user`, `id_jenis`,
`total_bayar`, `status_cuci`)
                + " VALUES ('" + mt.getId_trx()
+ "','" + mt.getId_user() + "','" + mt.getId_jenis() + "','" +
mt.getTot_bayar() + "','" + mt.getStatus() + "')";
            break;
        case 2:

```

```

        query = "INSERT INTO
`transaksi`(`id_transaksi`, `id_user`, `id_jenis`,
`total_bayar`, `bayar`, `kembali`, `tgl_cuci`,
`status_cuci`)"
                + " VALUES ('"+ mt.getId_trx()
+"', '"+ mt.getId_user() +"' , '"+ mt.getId_jenis() +"' , '"+
mt.getTot_bayar() +"' , '"+ mt.getBayar() +"' , "
                + "'"+ mt.getKembali() +"' , '"+
mt.getTgl_cuci() +"' , '"+ mt.getStatus() +"' )";
        break;
        case 3:
            query = "UPDATE `transaksi` SET
`bayar`='"+ mt.getBayar() +"' , `kembali`='"+
mt.getKembali() +"' , `tgl_cuci`='"+ mt.getTgl_cuci() +"' , "
                + "`status_cuci`='"+
mt.getStatus() +"' WHERE id_transaksi LIKE '%" +
mt.getId_trx() + "%'";
            break;
        default:
            break;
    }

    try {
        conMan = new ConnectionManager();
        con = conMan.logOn();

        st = con.createStatement();
        hasil = st.executeUpdate(query);
        conMan.logOff();
    } catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }

    return hasil;
}

public int cekDataTransaksi(ModelTransaksi mt) {
    int hasil = 0;

    String query = "SELECT * FROM transaksi WHERE
id_transaksi = '"+ mt.getId_trx() + "'";

```

```

        try {
            conMan = new ConnectionManager();
            con = conMan.logOn();

            st = con.createStatement();
            rs = st.executeQuery(query);

            if (rs.next()) {
                hasil = 1;
            }

            conMan.logOff();
        } catch (Exception ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
        }

        return hasil;
    }

    public List<ModelTransaksi> cariData (ModelTransaksi
mt) {
        List<ModelTransaksi> lsTrx = new
ArrayList<ModelTransaksi>();
        String query = "SELECT * FROM vtransaksi WHERE
id_transaksi LIKE '%" + mt.getId_trx() + "%'";

        conMan = new ConnectionManager();
        con = conMan.logOn();

        try {
            st = con.createStatement();
            rs = st.executeQuery(query);

            while (rs.next()) {
                ModelTransaksi mta = new ModelTransaksi();

mta.setId_trx(rs.getString("id_transaksi"));
                mta.setId_user(rs.getString("id_user"));
                mta.setNama_user(rs.getString("nama"));
                mta.setId_jenis(rs.getString("id_jenis"));
            }
        }
    }

```



```

mta.setNama_ken(rs.getString("tipe_kendaraan"));

mta.setTot_bayar(rs.getString("total_bayar"));
        mta.setBayar(rs.getString("bayar"));
        mta.setKembali(rs.getString("kembali"));

mta.setStatus(rs.getString("status_cuci"));

        lsTrx.add(mta);
    }
    conMan.logOff();
} catch (SQLException ex) {

java.util.logging.Logger.getLogger(ControllerAkun.class.ge
tName()).log(Level.SEVERE, null, ex);
    }
    return lsTrx;
}
}

```

7. Halaman Transaksi (User)

The screenshot displays the 'myVWash' web application interface. On the left, a dark sidebar contains navigation options: 'Menu Utama', 'Sejarah Cuci', 'Utm', 'Transaksi' (highlighted), 'Riwayat Transaksi', and 'Keluar'. The main content area, titled 'Form Cuci Kendaraan Baru', features several input fields and buttons. The fields include 'ID Transaksi', 'Jenis Kendaraan', and 'Harga Pencucian'. A 'Cari Data' button is positioned next to the 'Jenis Kendaraan' field. Below these fields, there are two buttons: 'Pencucian Baru' and 'Proses'. At the bottom of the sidebar, the text 'Khusus' is visible.

8. Halaman Riwayat Transaksi (User)

myVWash

Menu Utama

Selamat Datang, Udin

Form Cuci Kendaraan Baru

* Seret Kursor Ke Tabel Untuk Menyegarkan Data

ID Transaksi	Status Pencucian	Pelanggan	Tanggal Cuci	Jenis Kendaraan	Total Pembayaran	Bayar	Kembali
TRX-012254601	Selesai	AKN-002 Udin	2021/06/01 12:06:09	KEN-008 Sepeda	10000	20000	10000
TRX-0218425603	Selesai	AKN-002 Udin	2021/18/02 17:18:41	KEN-002 Motor 250cc s/d ..	20000	20000	0
TRX-021841303	Selesai	AKN-002 Udin	2021/41/02 18:41:14	KEN-007 Truck	37500	40000	2500
TRX-0218542004	Selesai	AKN-002 Udin	2021/11/03 16:13:15	KEN-002 Motor 250cc s/d ..	20000	20000	0
TRX-0315382004	Belum Selesai	AKN-002 Udin		KEN-001 Motor < 250cc	15000		
TRX-0315489405	Belum Selesai	AKN-002 Udin		KEN-004 Mobil : Small Size	36000		
TRX-0315541505	Belum Selesai	AKN-002 Udin		KEN-005 Mobil : Mid Size	41000		
TRX-0315571105	Belum Selesai	AKN-002 Udin		KEN-003 Motor > 600cc	25000		
TRX-0315591305	Belum Selesai	AKN-002 Udin		KEN-006 Mobil : Large Size	46000		
TRX-031602205	Selesai	AKN-002 Udin	2021/14/03 16:14:36	KEN-001 Motor < 250cc	15000	50000	35000
TRX-031613205	Selesai	AKN-002 Udin	2021/13/03 16:13:43	KEN-005 Mobil : Mid Size	41000	45000	4000
TRX-041594605	Selesai	AKN-002 Udin	2021/10/04 15:10:06	KEN-005 Mobil : Mid Size	41000	41000	0

Transaksi

Riwayat Transaksi

Keluar

BAB V

PENUTUP

5.1. Kesimpulan

Aplikasi Manajemen Cuci Kendaraan adalah aplikasi yang dibuat dengan bahasa pemrograman java dimana aplikasi ini dibuat dengan tujuan untuk mempermudah pelayanan pencucian kendaraan ke pelanggan. Kendaraan yang dapat dicuci meliputi kendaraan bermotor seperti mobil, motor, dll.

Aplikasi ini dapat menangani manajemen data akun, manajemen data jenis kendaraan, dan manajemen transaksi. Aplikasi ini hanya dibuat untuk pengusaha pencucian kendaraan dan tidak bisa untuk jenis usaha lain.

Aplikasi ini dapat membantu pekerja pelayanan pencucian dalam menghemat waktu pelayanan, memudahkan manajemen data dan mempermudah pengguna untuk melakukan transaksi.

5.2. Saran

Jika ada yang ingin melanjutkan proyek ini, diharapkan pengembang lebih memperhatikan tentang fitur yang dibutuhkan oleh pengguna akhir, dan juga diharapkan dalam pengembangan selanjutnya error handling yang digunakan lebih kompleks sehingga meminimalisir error pada aplikasi.

DAFTAR PUSTAKA

- Dewi, R. S. (2020, Februari 20). *Kenali Pengertian Java Beserta Fungsi, Kelebihan dan Kekurangan Java*. Diambil kembali dari Nesabamedia:
<https://www.nesabamedia.com/pengertian-java/>
- Salamadian. (2018, April 9). *BASIS DATA : Pengertian, Komponen dan Sistem Basis Data (Database)*. Diambil kembali dari SALAMADIAN:
<https://salamadian.com/pengertian-basis-data-database/>
- Web TP UPI. (2018, Februari 14). *Konsep Dasar Sistem*. Diambil kembali dari Kurtek UPI: <http://kurtek.upi.edu/2018/02/14/konsep-sistem/#:~:text=Konsep%20dasar%20sistem,satu%20sama%20lain%2C%20dan%20terpadu.>