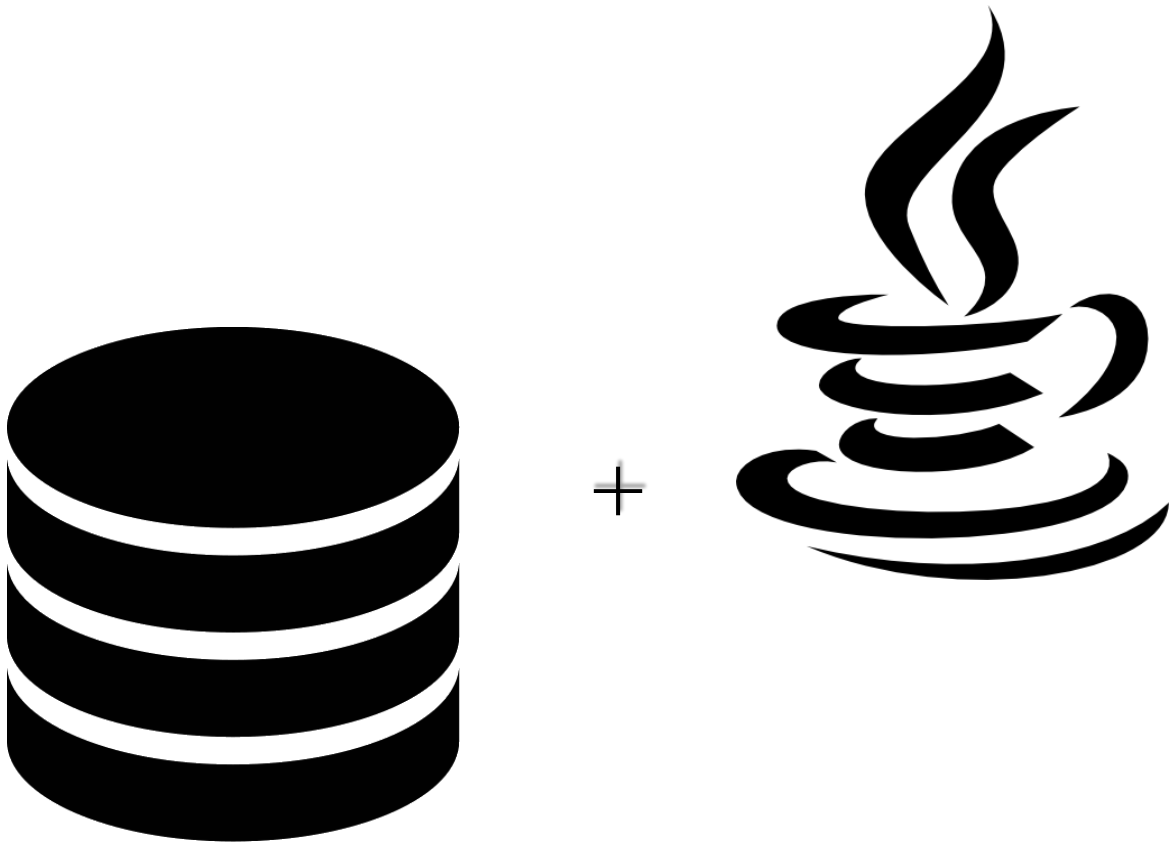


Modul 7

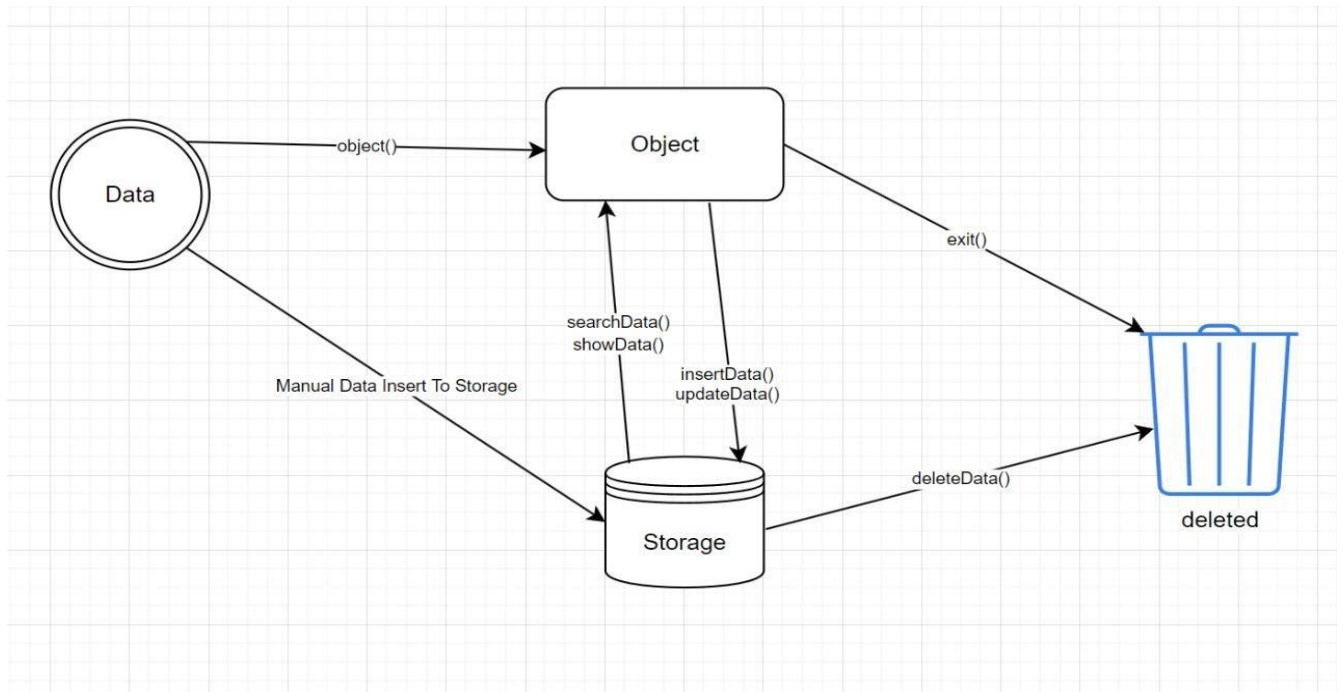
OBJECT PERSISTENCE



Pemegang Modul: Melvin

Dasar Teori

Pada objek-objek yang pernah dibuat sebelumnya, kita telah mempelajari bagaimana untuk mengolah objek, dan relasi pada objek yang dibuat. Objek yang dibuat dalam program, di simpan dalam ram, sehingga hanya tersedia sampai program di terminasi, objek tersebut merupakan objek transien. Jika data tidak terhapus setelah terminasi, artinya objek tersebut telah ditempatkan pada penyimpanan utama komputer dengan berbagai teknik penyimpanan. Kemampuan objek untuk tetap bertahan setelah terminasi program disebut objek persisten.



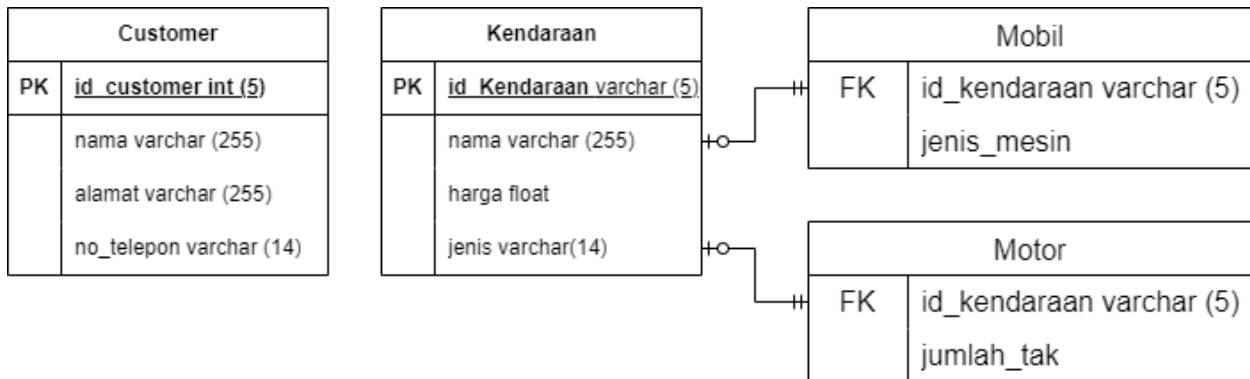
Object Persistence cocok digunakan pada aplikasi yang membutuhkan data untuk disimpan. Contoh aplikasi yang membutuhkan objek untuk persisten adalah aplikasi POS, yang memerlukan data untuk dicatat dan di simpan. Data yang di simpan pada database akan diambil kembali (restore) ketika client berjalan, dan meminta akses (request) data di database. Untuk itu, sistem biasanya memiliki beberapa layer sebagai berikut:

- Presentation layer
- Service layer
- Data acces layer
- Database

Guided

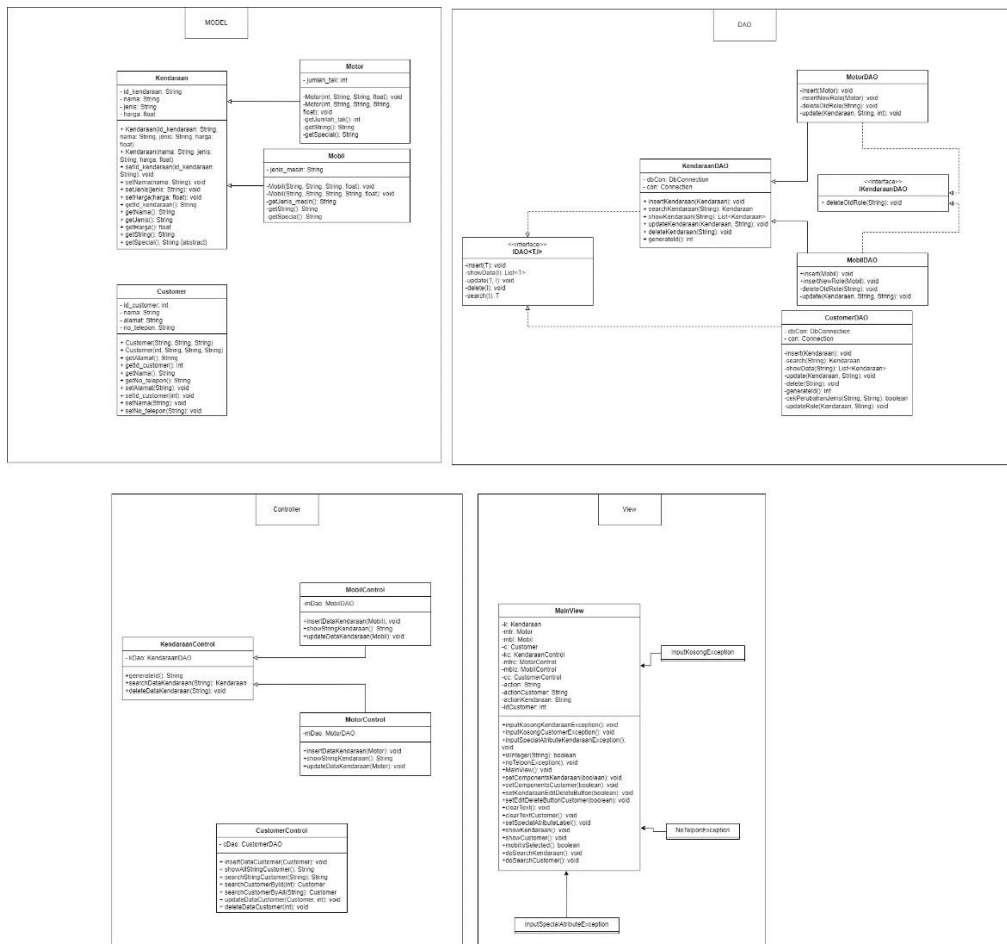
Berikut tabel relasi dari program yang akan dibuat, silahkan samakan penamaan yang ada dalam tabel relasi.

Tabel



Gambar 1

Class Diagram



Studi Kasus

Showroom Kendaraan AtMyVech ingin membuat suatu sistem yang dapat mengelola data kendaraan dan customer. Sistem AtMyVech ini akan dibangun menggunakan bahasa pemrograman java dan data akan disimpan pada basis data MySQL. Berikut adalah panduan dalam membuat pengelolaan data sistem AtMyVech.

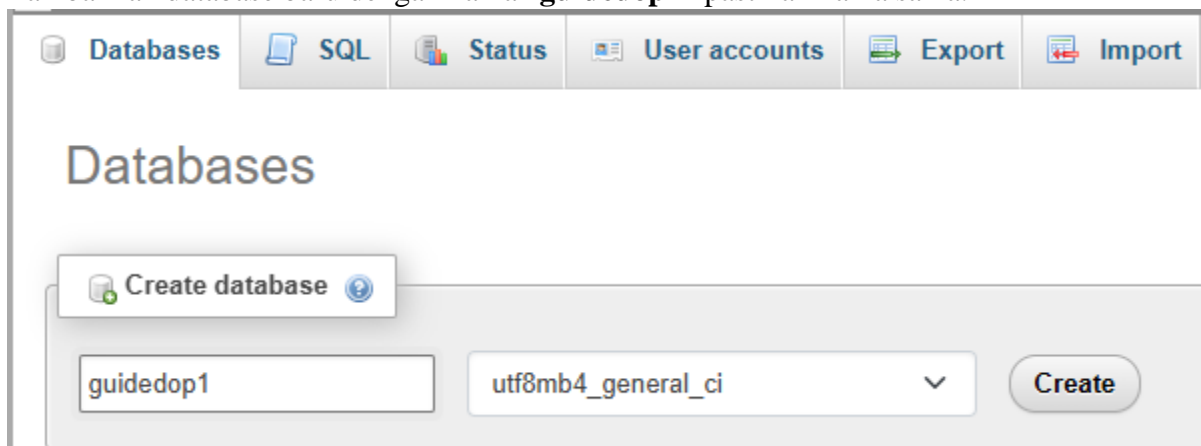
Data customer dan kendaraan pada **Gambar 2**, dikelola melalui fitur-fitur sebagai berikut:

- Menambahkan data customer dan kendaraan kedalam sistem
- Menampilkan seluruh data customer dan data kendaraan
- Mencari data customer dan kendaraan berdasarkan **primary key / id**
- Mengubah data customer dan kendaraan berdasarkan **primary key / id**
- Menghapus data customer dan kendaraan berdasarkan **primary key / id**

Seluruh data akan di simpan pada basis data. Terdapat beberapa exception yang perlu diberikan, yaitu, **semua input tidak boleh kosong**, dan input nomor telepon harus sesuai format, yaitu **berupa angka, dan diawali dengan + atau angka**.

Pembuatan Database

- Pastikan modul Apache dan MySql sudah dijalankan
- Klik tombol admin pada modul MySql untuk membuka page PhpMyAdmin pada browser
- Tambahkan database baru dengan nama “**guidedop1**” pastikan nama sama.



- Buka tab “**guidedop1**” dan tambahkan tabel dengan ketikan nama tabel.



Menambah relasi kendaraan.

Untuk menambahkan relasi kendaraan buka tabel mobil, dan masuk ke structure>relation view, kemudian tambahkan relasi berikut.

Server: 127.0.0.1 » Database: guidedop1 » Table: mobil

Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers

Table structure Relation view

Foreign key constraints

| Actions | Constraint properties | Column | Foreign key constraint (INNODB) | | |
|---------|--|--------------|---------------------------------|-----------|--------------|
| | | | Database | Table | Column |
| | fk_mobil_kendaraan ON DELETE CASCADE ON UPDATE CASCADE | id_kendaraan | guidedop1 | kendaraan | id_kendaraan |

+ Add constraint + Add column

Internal relationships

Choose column to display: ---

Preview SQL Save

Klik Save, kemudian tambahkan juga relation pada tabel motor

Server: 127.0.0.1 » Database: guidedop1 » Table: motor

Structure SQL Search Insert Export Import Privileges Operations Tracking Triggers

Table structure Relation view

Foreign key constraints

| Actions | Constraint properties | Column | Foreign key constraint (INNODB) | | |
|---------|--|--------------|---------------------------------|-----------|--------------|
| | | | Database | Table | Column |
| | fk_motor_kendaraan ON DELETE CASCADE ON UPDATE CASCADE | id_kendaraan | guidedop1 | kendaraan | id_kendaraan |

+ Add constraint + Add column

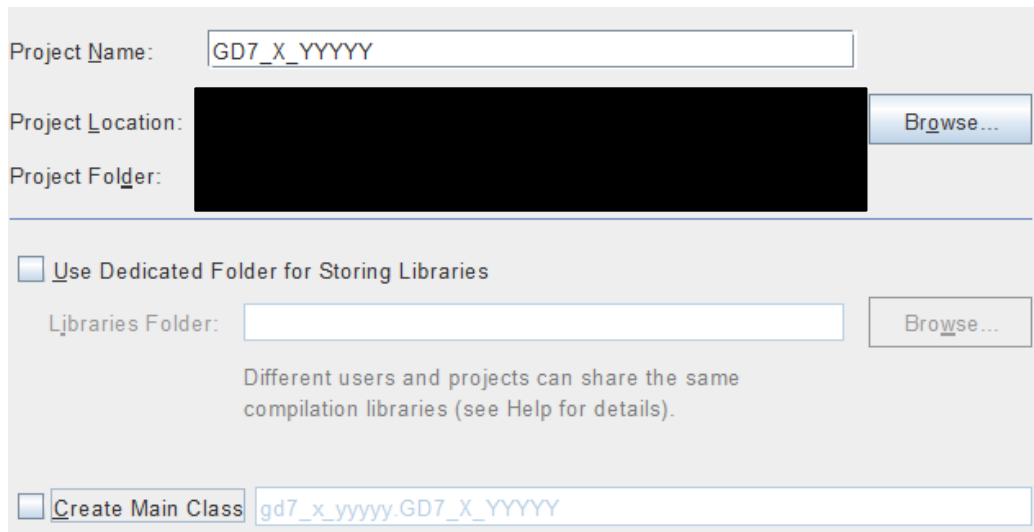
Internal relationships

Choose column to display: ---

Preview SQL Save

Koneksi Program ke Database

- Buat project baru dengan nama GD7_X_YYYYY (X=kelas, Y=5 digit akhir NPM).
Uncheck main class, karena tidak diperlukan pada pembuatan project kali ini.



Project Name: GD7_X_YYYYY

Project Location: [Redacted] Browse...

Project Folder: [Redacted]

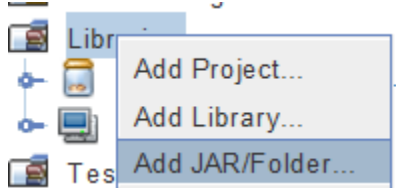
☐ Use Dedicated Folder for Storing Libraries

Libraries Folder: [Redacted] Browse...

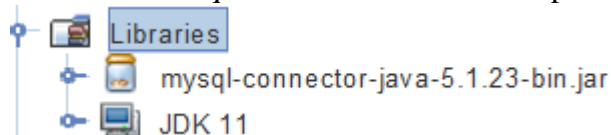
Different users and projects can share the same compilation libraries (see Help for details).

☐ Create Main Class gd7_x_yyyy.GD7_X_YYYYY

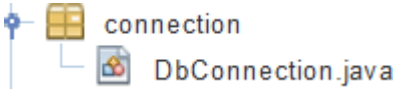
- Tambahkan library MySqlConnection, klik kanan pada library dan klik add JAR/folder.



- Cari file mysql-connector-java-5.1.23-bin.jar, kemudian klik open, library project akan menambahkan sql conector dan akan ditampilkan sebagai berikut:



- Kemudian kita perlu membuat kelas koneksi pada sql, tambahkan package connection, pada package connection tambahkan kelas DbConnection.

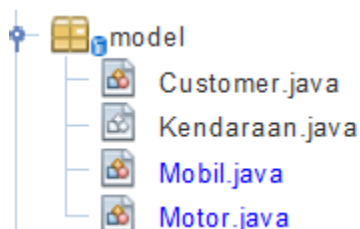


DbConnection.java

```
1  package connection;
2
3  import java.sql.Connection;
4  import java.sql.DriverManager;
5
6  public class DbConnection {
7
8      public static Connection CON;
9      public static final String URL = "jdbc:mysql://";
10     public static final String DBNAME = "guidedop1";
11     public static final String PATH = "localhost:3306/" + DBNAME;
12
13     public Connection makeConnection() {
14         System.out.println("Opening database...");
15         try {
16             CON = DriverManager.getConnection(URL + PATH, "root", "");
17             System.out.println("success...");
18         } catch (Exception e) {
19             System.out.println("error opening database");
20             System.out.println(e);
21         }
22         return CON;
23     }
24
25     public void closeConnection() {
26         System.out.println("closing database...");
27         try {
28             CON.close();
29             System.out.println("success...");
30         } catch (Exception e) {
31             System.out.println("error closing database");
32             System.out.println(e);
33         }
34     }
35 }
```

Model Class

- Untuk mengakses data pada sql, program dapat menentukan bagaimana data direpresentasikan dalam program. Buat package model dan buat kelas Customer, Kendaraan, Mobil, dan Motor. Kendaraan menjadi kelas abstrak untuk mobil dan motor. Karena mobil dan motor memiliki atribut khusus, untuk mengaksesnya kita menggunakan method getSpecial.





Customer.java

```
package model;

public class Customer {
    private int id_customer;
    private String nama;
    private String alamat;
    private String no_telepon;

    public Customer(String nama, String alamat, String no_telepon) {
        this.nama = nama;
        this.alamat = alamat;
        this.no_telepon = no_telepon;
    }

    public Customer(int id_customer, String nama, String alamat, String no_telepon) {
        this.id_customer = id_customer;
        this.nama = nama;
        this.alamat = alamat;
        this.no_telepon = no_telepon;
    }

    public String getAlamat() {
        return alamat;
    }

    public int getId_customer() {
        return id_customer;
    }

    public String getNama() {
        return nama;
    }

    public String getNo_telepon() {
        return no_telepon;
    }

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

    public void setId_customer(int id_customer) {
        this.id_customer = id_customer;
    }

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

    public void setNo_telepon(String no_telepon) {
        this.no_telepon = no_telepon;
    }
}
```



Kendaraan.java

```
package model;

public abstract class Kendaraan{
    private String id_kendaraan, nama, jenis;
    private float harga;

    public Kendaraan(String id_kendaraan, String nama, String jenis, float harga) {
        this.id_kendaraan = id_kendaraan;
        this.nama = nama;
        this.jenis = jenis;
        this.harga = harga;
    }

    public Kendaraan(String nama, String jenis, float harga) {
        this.nama = nama;
        this.jenis = jenis;
        this.harga = harga;
    }

    public void setId_kendaraan(String id_kendaraan) {
        this.id_kendaraan = id_kendaraan;
    }

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

    public void setJenis(String jenis) {
        this.jenis = jenis;
    }

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

    public String getId_kendaraan() {
        return id_kendaraan;
    }

    public String getName() {
        return nama;
    }

    public String getJenis() {
        return jenis;
    }

    public float getHarga() {
        return harga;
    }

    public String getString(){
        return id_kendaraan + " | " + nama + " | " + harga;
    }

    public abstract String getSpecial();
}
```

```
Mobil.java

package model;

public class Mobil extends Kendaraan {
    private String jenis_mesin;

    public Mobil(String jenis_mesin, String nama, String jenis, float harga) {
        super(nama, "Mobil", harga);
        this.jenis_mesin = jenis_mesin;
    }

    public Mobil(String jenis_mesin, String id_kendaraan, String nama, String jenis, float harga) {
        super(id_kendaraan, nama, "Mobil", harga);
        this.jenis_mesin = jenis_mesin;
    }

    public String getJenis_mesin() {
        return jenis_mesin;
    }

    public String getString(){
        return super.getString() + " | " + jenis_mesin;
    }

    @Override
    public String getSpecial() {
        return jenis_mesin;
    }
}
```

```
Motor.java

package model;

public class Motor extends Kendaraan {
    private int jumlah_tak;

    public Motor(int jumlah_tak, String nama, String jenis, float harga) {
        super(nama, "Motor", harga);
        this.jumlah_tak = jumlah_tak;
    }

    public Motor(int jumlah_tak, String id_kendaraan, String nama, String jenis, float harga) {
        super(id_kendaraan, nama, jenis, harga);
        this.jumlah_tak = jumlah_tak;
    }

    public int getJumlah_tak() {
        return jumlah_tak;
    }

    public String getString(){
        return super.getString() + " | " + jumlah_tak;
    }

    @Override
    public String getSpecial() {
        return jumlah_tak + "";
    }
}
```

Data Access Object

Setelah membuat model dan koneksi database, kita perlu cara untuk mengakses objek pada database ke dalam model yang telah dibuat, data access object adalah objek yang berisi semua hal itu. Buatlah package interfaceDAO dan package DAO.



Interface DAO

Terdapat 2 kelas Interface untuk menyusun kelas DAO, kelas IDAO yang memiliki fungsi umum DAO, dan IKendaraanDAO yang memiliki fungsi general dari DAO kendaraan.



DAO

DAO memiliki fungsi CRUDS, dan beberapa tambahan fungsi supaya crud berjalan dengan baik.

CustomerDAO

Import

```
package dao;

import connection.DbConnection;
import interfaceDAO.IDAO;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import model.Customer;
```

Variables

```
public class CustomerDAO implements IDAO<Customer, Integer>{
    private DbConnection dbCon = new DbConnection();
    private Connection con;

    ...

    @Override
```

Create

```
@Override
public void insert(Customer C){
    con = dbCon.makeConnection();

    String sql =
        "INSERT INTO `customer` (`nama`, `alamat`, `no_telepon`) " +
        "VALUES ('"+ C.getNama() +"', '"+ C.getAlamat() +"', '"+ C.getNo_telepon() +"')";

    System.out.println("Adding Customer...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Added " + result + " Customer");
        statement.close();
    }catch (Exception e){
        System.out.println("Error adding Customer...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

Read

```
@Override
public List<Customer> showData(Integer data){
    con = dbCon.makeConnection();

    String sql = "SELECT * FROM customer";
    System.out.println("Fetching Data...");
    List<Customer> c = new ArrayList();

    try{
        Statement statement = con.createStatement();
        ResultSet rs = statement.executeQuery(sql);

        if(rs != null)
            while(rs.next())
                c.add(new Customer(
                    rs.getInt("id_customer"),
                    rs.getString("nama"),
                    rs.getString("alamat"),
                    rs.getString("no_telepon")));

        rs.close();
        statement.close();
    }catch (Exception e){
        System.out.println("Error Fetching data...");
        System.out.println(e);
    }
    dbCon.closeConnection();
    return c;
}
```

Update

```
@Override
public void update(Customer c, Integer id_customer){
    con = dbCon.makeConnection();

    String sql = "UPDATE `customer` SET "
        + "`nama`='" + c.getNama() + "',"
        + "`alamat`='" + c.getAlamat() + "',"
        + "`no_telepon`='" + c.getNo_telepon() + "'"
        + "WHERE `id_customer`='" + id_customer + "'";
    System.out.println("Updating customer");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Edited " + result + " Customer " + id_customer);
        statement.close();
    }catch(Exception e){
        System.out.println("Error Updating Customer...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

Delete

```
@Override
public void delete(Integer id_customer){
    con = dbCon.makeConnection();
    String sql = "DELETE FROM `customer` WHERE `id_customer` = " + id_customer + " ";
    System.out.println("Deleting Customer...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Edited " + result + " Customer " + id_customer);
        statement.close();
    }catch(Exception e){
        System.out.println("Error Updating Customer...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

Search

```
@Override
public Customer search(Integer id_customer){
    con = dbCon.makeConnection();

    String sql = "SELECT * FROM customer WHERE id_customer='"+id_customer+"'";
    System.out.println("Searching Customer...");
    Customer c = null;

    try{
        Statement statement = con.createStatement();
        ResultSet rs = statement.executeQuery(sql);

        if(rs != null)
            while(rs.next())
                c = new Customer(
                    rs.getInt("id_customer"),
                    rs.getString("nama"),
                    rs.getString("alamat"),
                    rs.getString("no_telepon"));

        rs.close();
        statement.close();
    }catch(Exception e){
        System.out.println("Error Fetching data...");
        System.out.println(e);
    }
    dbCon.closeConnection();
    return c;
}
```

KendaraanDAO

Import

```
package dao;
import connection.DbConnection;
import interfaceDAO.IDAO;
import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.Statement;
import java.util.ArrayList;
import java.util.List;
import model.Kendaraan;
import model.Mobil;
import model.Motor;
```

Variables

```
public class KendaraanDAO implements IDAO<Kendaraan, String>{
    protected DbConnection dbCon = new DbConnection();
    protected Connection con;
```

Create

```
@Override
public void insert(Kendaraan K){
    con = dbCon.makeConnection();

    String sql =
        "INSERT INTO `kendaraan`(`id_kendaraan`, `nama`, `jenis`, `harga`) "
        + "VALUES ('"+K.getId_kendaraan()+"', '"+K.getNama()+"', '"+K.getJenis()+"', '"+K.getHarga()+"')";

    System.out.println("Adding Kendaraan...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Added " + result + " Kendaraan");
        statement.close();
    }catch (Exception e){
        System.out.println("Error adding Kendaraan...");
        System.out.println(e);
    }
    dbCon.closeConnection();
};
```

Read

```
@Override
public List<Kendaraan> showData (String jenis){
    con = dbCon.makeConnection();

    String sql = "SELECT kendaraan.*, motor.jumlah_tak, mobil.jenis_mesin FROM kendaraan\n" +
        "LEFT JOIN motor ON kendaraan.id_kendaraan = motor.id_kendaraan\n" +
        "LEFT JOIN mobil on kendaraan.id_kendaraan = mobil.id_kendaraan\n" +
        "WHERE kendaraan.jenis = '"
        + jenis
        + "';";
    System.out.println("Fetching Data...");

    List<Kendaraan> list = new ArrayList();

    try{
        Statement statement = con.createStatement();
        ResultSet rs = statement.executeQuery(sql);
        Kendaraan k = null;

        if(rs != null)
            while(rs.next()){
                if(rs.getString("jenis").equals("Motor")){
                    k = new Motor(
                        rs.getInt("jumlah_tak"),
                        rs.getString("id_kendaraan"),
                        rs.getString("nama"),
                        rs.getString("jenis"),
                        rs.getInt("harga"));
                }
                else{
                    k = new Mobil(
                        rs.getString("jenis_mesin"),
                        rs.getString("id_kendaraan"),
                        rs.getString("nama"),
                        rs.getString("jenis"),
                        rs.getInt("harga"));
                }
                list.add(k);
            }
        rs.close();
        statement.close();
    }catch(Exception e){
        System.out.println("Error Fetching data...");
        System.out.println(e);
    }
    dbCon.closeConnection();
    return list;
}
```

Read pada kendaraan disatukan berdasarkan model abstrak Kendaraan, yang ditampung pada List Model Kendaraan. List Kendaraan akan berisi Mobil dan Motor.

Update

```
@Override
public void update (Kendaraan k, String id_kendaraan){
    con = dbCon.makeConnection();

    String sql = "UPDATE `kendaraan` SET "
        + "`nama`='" + k.getNama() + "',"
        + "`jenis`='" + k.getJenis() + "',"
        + "`harga`='" + k.getHarga() + "' "
        + "WHERE id_kendaraan='" + id_kendaraan + "'";
    System.out.println("Updating Kendaraan...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Edited" + result + " Kendaraan " + id_kendaraan);
        statement.close();
    }catch(Exception e){
        System.out.println("Error Updating Kendaraan...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

Delete

```
@Override
public void delete(String id_kendaraan){
    con = dbCon.makeConnection();
    String sql = "DELETE FROM `kendaraan` WHERE `id_kendaraan` = '"+id_kendaraan+"'";
    System.out.println("Deleting Kendaraan...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Deleted" + result + " Kendaraan " + id_kendaraan);
        statement.close();
    }catch(Exception e){
        System.out.println("Error Updating Kendaraan...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

Search

```
@Override
public Kendaraan search(String id_kendaraan){
    con = dbCon.makeConnection();

    String sql = "SELECT kendaraan.*, motor.jumlah_tak, mobil.jenis_mesin FROM kendaraan\n" +
        "LEFT JOIN motor ON kendaraan.id_kendaraan = motor.id_kendaraan\n" +
        "LEFT JOIN mobil on kendaraan.id_kendaraan = mobil.id_kendaraan\n" +
        "WHERE kendaraan.id_kendaraan = '"
        + id_kendaraan
        + "'";
    System.out.println("Searching Kendaraan...");
    Kendaraan k = null;

    try{
        Statement statement = con.createStatement();
        ResultSet rs = statement.executeQuery(sql);

        if(rs != null)
            while(rs.next()){
                if(rs.getString("jenis").equals("Motor")){
                    k = new Motor(
                        rs.getInt("jumlah_tak"),
                        rs.getString("id_kendaraan"),
                        rs.getString("nama"),
                        rs.getString("jenis"),
                        rs.getInt("harga"));
                }
                else{
                    k = new Mobil(
                        rs.getString("jenis_mesin"),
                        rs.getString("id_kendaraan"),
                        rs.getString("nama"),
                        rs.getString("jenis"),
                        rs.getInt("harga"));
                }
            }

        rs.close();
        statement.close();
    }catch(Exception e){
        System.out.println("Error Fetching data...");
        System.out.println(e);
    }
    dbCon.closeConnection();
    return k;
}
```

GenerateID

```
public int generateId(){
    con = dbCon.makeConnection();
    String sql = "SELECT MAX(CAST(SUBSTRING(id_kendaraan, 2) AS SIGNED)) AS highest_number FROM kendaraan WHERE id_kendaraan LIKE 'K%';";
    //mendapatkan nilai tertinggi dari id yang ada di database

    System.out.println("Generating Id...");
    int id=0;

    try{
        Statement statement = con.createStatement();
        ResultSet rs = statement.executeQuery(sql);

        if(rs != null && rs.next()){
            if(!rs.wasNull())
                id = rs.getInt("highest_number")+1;
        }

        //memasukan id terakhir ke dalam variabel id

        rs.close();
        statement.close();
    }catch(Exception e){
        System.out.println("Error Fetching data...");
        System.out.println(e);
    }
    dbCon.closeConnection();
    return id;
}
```

cekPerubahanJenis

```
public boolean cekPerubahanJenis(String jenis, String id_kendaraan){
    con = dbCon.makeConnection();

    String sql = "SELECT  jenis"
        + "      + jenis"
        + "      + " " "
        + "      + "as result"
        + "      + " FROM 'kendaraan'"
        + "      + " WHERE id_kendaraan = '"
        + "      + id_kendaraan"
        + "      + "';";
    System.out.println(sql);
    System.out.println("Checking Result...");
    boolean result = false;

    try{
        Statement statement = con.createStatement();
        ResultSet rs = statement.executeQuery(sql);

        if(rs != null)
            while(rs.next()){
                result = rs.getBoolean("result");
            }

        rs.close();
        statement.close();
    }catch(Exception e){
        System.out.println("Error Fetching data...");
        System.out.println(e);
    }
    dbCon.closeConnection();
    System.out.println("The result is" + result);
    return result;
}
```

MobilDAO

```
package dao;

import interfaceDAO.IKendaraanDAO;
import java.sql.Statement;
import model.Kendaraan;
import model.Mobil;

public class MobilDAO extends KendaraanDAO implements IKendaraanDAO{

    public void insert(Mobil mbl) {
        super.insert(mbl);
        insertNewRole(mbl);
    }

    public void insertNewRole(Mobil mbl) {
        con = dbCon.makeConnection();

        String sql =
            "INSERT INTO 'mobil'(`id_kendaraan`, `jenis_mesin`) VALUES ('"
            + mbl.getId_kendaraan()
            + "','"
            + mbl.getJenis_mesin()
            + "')";

        System.out.println("Adding Kendaraan...");

        try{
            Statement statement = con.createStatement();
            int result = statement.executeUpdate(sql);
            System.out.println("Added " + result + " Kendaraan");
            statement.close();
        }catch (Exception e){
            System.out.println("Error adding Kendaraan...");
            System.out.println(e);
        }
        dbCon.closeConnection();
    }

    @Override
    public void deleteOldRole(String id_kendaraan) {
        con = dbCon.makeConnection();
        String sql = "DELETE FROM 'motor' WHERE `id_kendaraan` = '"+id_kendaraan+"'";
        System.out.println("Deleting mobil...");

        try{
            Statement statement = con.createStatement();
            int result = statement.executeUpdate(sql);
            System.out.println("Edited" + result + " Kendaraan " + id_kendaraan);
            statement.close();
        }catch (Exception e){
            System.out.println("Error Updating Kendaraan...");
            System.out.println(e);
        }
        dbCon.closeConnection();
    }

    public void update(Kendaraan k, String id_kendaraan, String jenis_mesin) {
        Mobil m = new Mobil(jenis_mesin, k.getId_kendaraan(), k.getNama(), k.getJenis(), k.getHarga());
        if(cekPerubahanJenis("Mobil", id_kendaraan)){
            deleteOldRole(id_kendaraan);
            insertNewRole(m);
        }else{
            updateRole(k, id_kendaraan);
        }
        super.update(k, id_kendaraan);
    }
}
```

updateRole

```
public void updateRole (Kendaraan k, String id_kendaraan){
    con = dbCon.makeConnection();

    String sql = "UPDATE `"
        + k.getJenis()
        + "` SET `jenis_mesin`='"
        + k.getSpecial()
        + "' WHERE `mobil`.id_kendaraan = '"
        + id_kendaraan
        + "'";

    System.out.println("Updating Jenis Kendaraan...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Edited" + result + " Kendaraan " + id_kendaraan);
        statement.close();
    }catch(Exception e){
        System.out.println("Error Updating Kendaraan...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

MotorDAO

```
package dao;

import interfaceDAO.IKendaraanDAO;
import java.sql.Statement;
import model.Kendaraan;
import model.Motor;

public class MotorDAO extends KendaraanDAO implements IKendaraanDAO{
    public void insert(Motor M){
        super.insert(M);
        insertNewRole(M);
    }

    public void insertNewRole(Motor K) {
        con = dbCon.makeConnection();

        String sql =
            "INSERT INTO `motor`(`id_kendaraan`, `jumlah_tak`) VALUES ("
            + K.getId_kendaraan()
            + ","
            + K.getJumlah_tak()
            + ")";

        System.out.println("Adding Kendaraan...");

        try{
            Statement statement = con.createStatement();
            int result = statement.executeUpdate(sql);
            System.out.println("Added " + result + " Kendaraan");
            statement.close();
        }catch (Exception e){
            System.out.println("Error adding Kendaraan...");
            System.out.println(e);
        }
        dbCon.closeConnection();
    }

    @Override
    public void deleteOldRole(String id_kendaraan) {
        con = dbCon.makeConnection();
        String sql = "DELETE FROM `mobil` WHERE `id_kendaraan` = '"+id_kendaraan+"'";
        System.out.println("Deleting mobil...");

        try{
            Statement statement = con.createStatement();
            int result = statement.executeUpdate(sql);
            System.out.println("Edited" + result + " Kendaraan " + id_kendaraan);
            statement.close();
        }catch(Exception e){
            System.out.println("Error Updating Kendaraan...");
            System.out.println(e);
        }
        dbCon.closeConnection();
    }

    public void update(Kendaraan k, String id_kendaraan, int jumlah_tak) {
        Motor m = new Motor(jumlah_tak, k.getId_kendaraan(), k.getNama(), k.getJenis(), k.getHarga());
        if(ccekPerubahanJenis("Motor", id_kendaraan)){
            deleteOldRole(id_kendaraan);
            insertNewRole(m);
        }else{
            updateRole(k, id_kendaraan);
        }
        super.update(k, id_kendaraan);
    }
}
```

updateRole

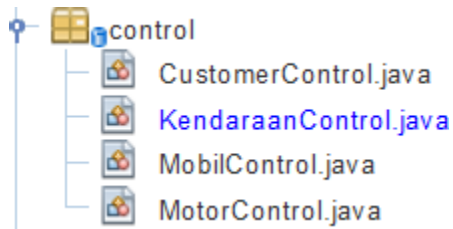
```
public void updateRole (Kendaraan k, String id_kendaraan){
    con = dbCon.makeConnection();

    String sql = "UPDATE `"
        + k.getJenis()
        + "` SET `jumlah_tak`='"
        + k.getSpecial()
        + "' WHERE `motor`.id_kendaraan = '"
        + id_kendaraan
        + "'";
    System.out.println("Updating Jenis Kendaraan...");

    try{
        Statement statement = con.createStatement();
        int result = statement.executeUpdate(sql);
        System.out.println("Edited" + result + " Kendaraan " + id_kendaraan);
        statement.close();
    }catch(Exception e){
        System.out.println("Error Updating Kendaraan...");
        System.out.println(e);
    }
    dbCon.closeConnection();
}
```

Controller

Controller adalah perantara antara basis data dan antarmuka pengguna. Implementasi business logic juga ada pada controller. Controller memanggil fungsi pada DAO, untuk dijalankan di user interface, sehingga akan terlihat redundant, namun terdapat keuntungan untuk memisahkan masalah, abstraksi dan enkapsulasi, skalabilitas, dan mempermudah pembacaan. Buatlah package controller, dan tambahkan kelas berikut.



CustomerControl

```
package control;

import model.Customer;
import dao.CustomerDAO;
import java.util.List;

public class CustomerControl {
    CustomerDAO cDao = new CustomerDAO();

    public void insertDataCustomer(Customer c){
        cDao.insert(c);
    }

    public String showAllStringCustomer(){
        List<Customer> listC = cDao.showData(0);
        String customerString = "";
        int i=0;

        for(Customer c : listC){
            i++;
            customerString+= c.getId_customer()+" "+c.getNama()+" | "+c.getAlamat()+" | "+c.getNo_telepon()+"\n";
        }
        return customerString;
    }

    public Customer searchCustomerById(int id_customer){
        return cDao.search(id_customer);
    }

    public void updateDataCustomer(Customer c, int id_customer){
        cDao.update(c, id_customer);
    }

    public void deleteDataCustomer(int id_customer){
        cDao.delete(id_customer);
    }
}
```


KendaraanControl

```
package control;

import dao.KendaraanDAO;
import model.Kendaraan;

public class KendaraanControl {
    KendaraanDAO kDao = new KendaraanDAO();

    public String generateId(){
        return "K"+kDao.generateId();
    }

    public Kendaraan searchDataKendaraan (String id){
        return kDao.search(id);
    }

    public void deleteDataKendaraan(String id){
        kDao.delete(id);
    }
}
```

Terlihat pada Kendaraan control, method CRUDS tidak semuanya diimplementasikan, karena dalam melakukan CRUDS, model kendaraan merupakan abstrak dan perlu menggunakan model dari pewarisannya untuk melakukan Create dan Update.

MobilControl

```
package control;
import dao.MobilDAO;
import java.util.List;
import model.Kendaraan;
import model.Mobil;

public class MobilControl {
    MobilDAO mDao = new MobilDAO();

    public void insertDataKendaraan(Mobil K){
        K.setId_kendaraan("K"+mDao.generateId());
        mDao.insert(K);
    }

    public String showStringKendaraan(){
        List<Kendaraan> listK = mDao.showData("Mobil");
        String kendaraanString = "";
        for(Kendaraan k : listK){
            kendaraanString += k.getString() + "\n";
        }
        return kendaraanString;
    }

    public void updateDataKendaraan(Mobil K){
        mDao.update(K, K.getId_kendaraan(), K.getJenis_mesin());
    }
}
```

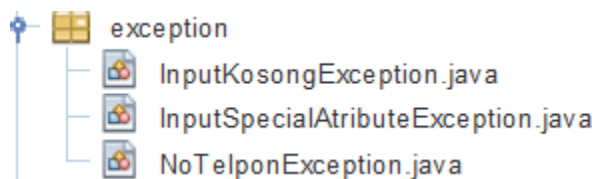
MotorControl



Pada mobil dan motor terdapat `ShowStringKendaraan`, alasan memisah ke dalam controller sendiri karena model ingin ditunjukkan pada textArea tersendiri.

Exception

Buatlah package exception, dengan 2 kelas exception berikut



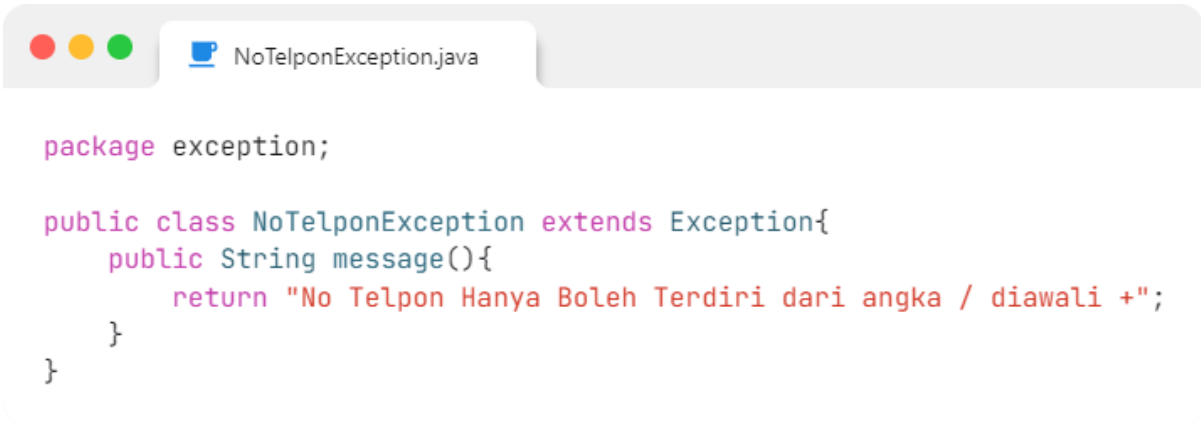
InputKosongException



```
package exception;

public class InputKosongException extends Exception{
    public String message(){
        return "FIELD INPUT TIDAK BOLEH KOSONG !";
    }
}
```

NoTelponException



```
package exception;

public class NoTelponException extends Exception{
    public String message(){
        return "No Telpon Hanya Boleh Terdiri dari angka / diawali +";
    }
}
```

InputSpecialAttributeException



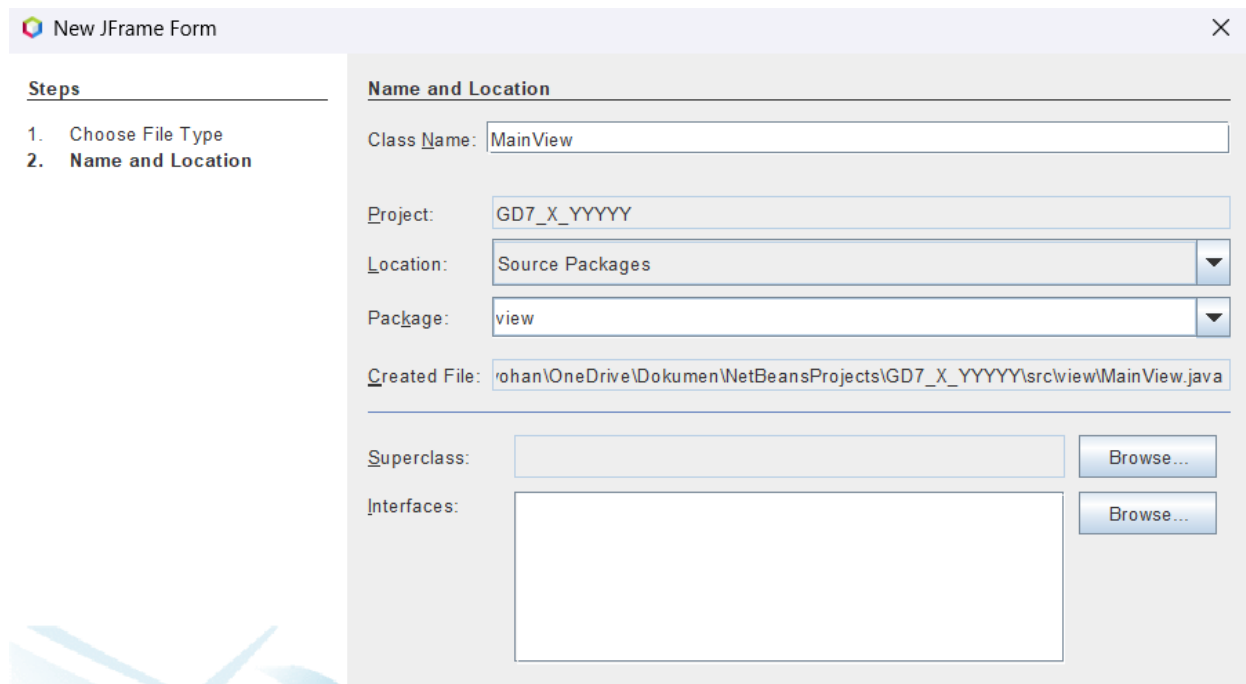
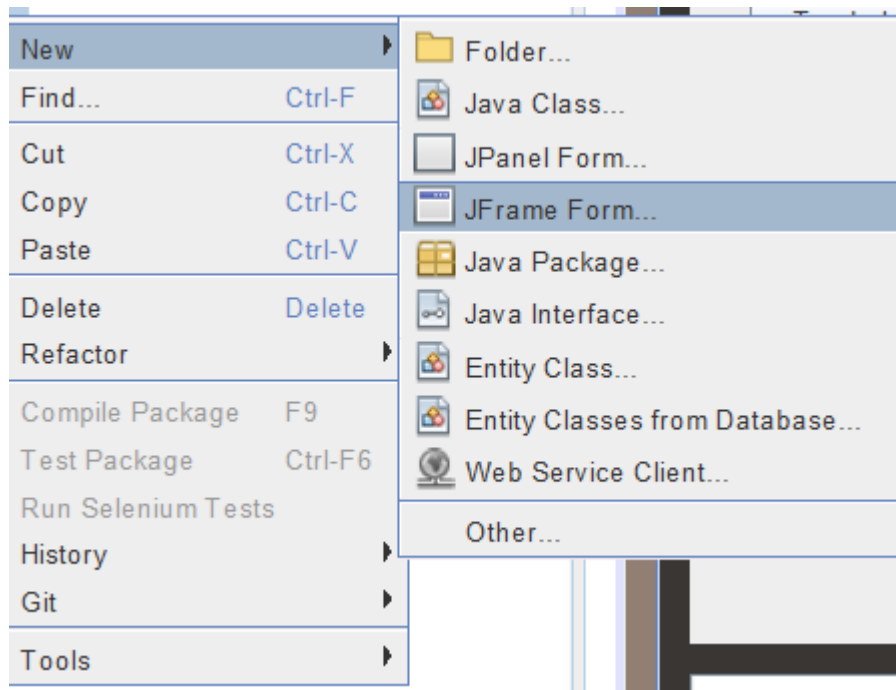
```
package exception;

public class InputSpecialAttributeException extends Exception{

    public String message(String jenis){
        return "Atribut Tidak Sesuai untuk jenis " + jenis;
    }
}
```

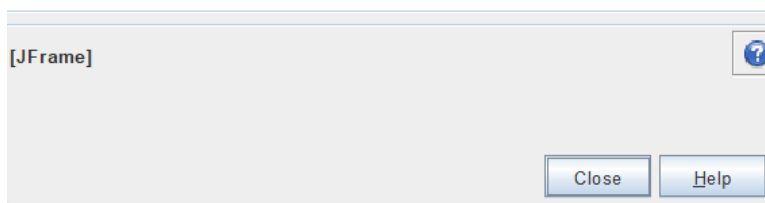
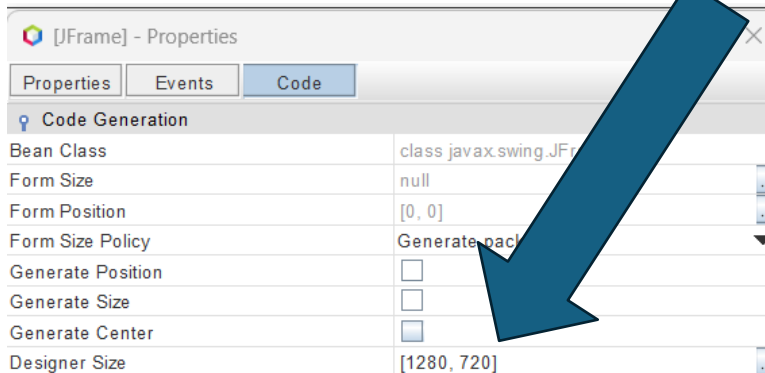
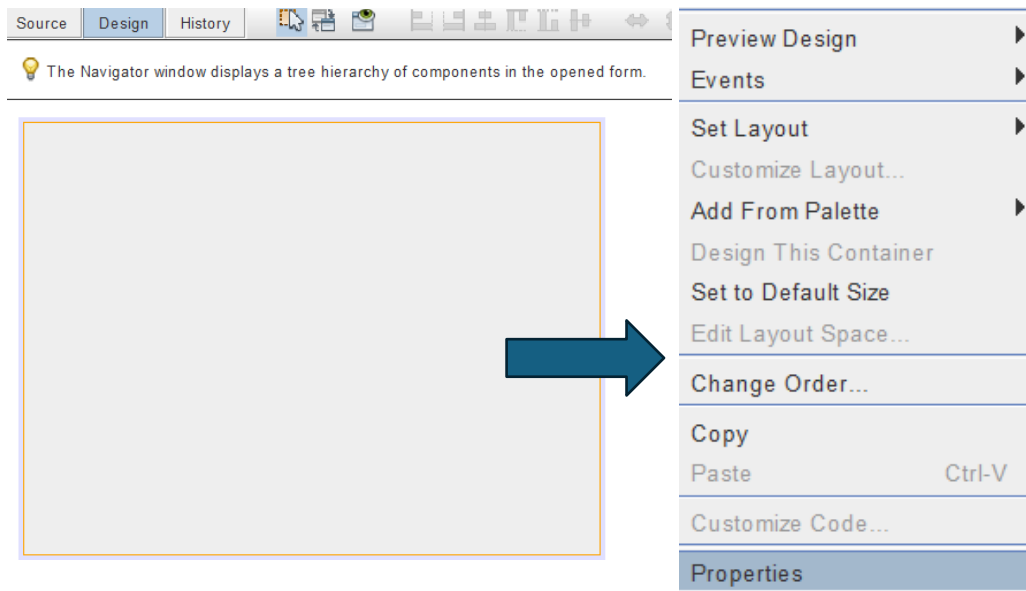
View

Untuk membuat tampilan ui pada aplikasi, buatlah package view, dan tambahkan JFrameForm baru. Berikan nama MainView.

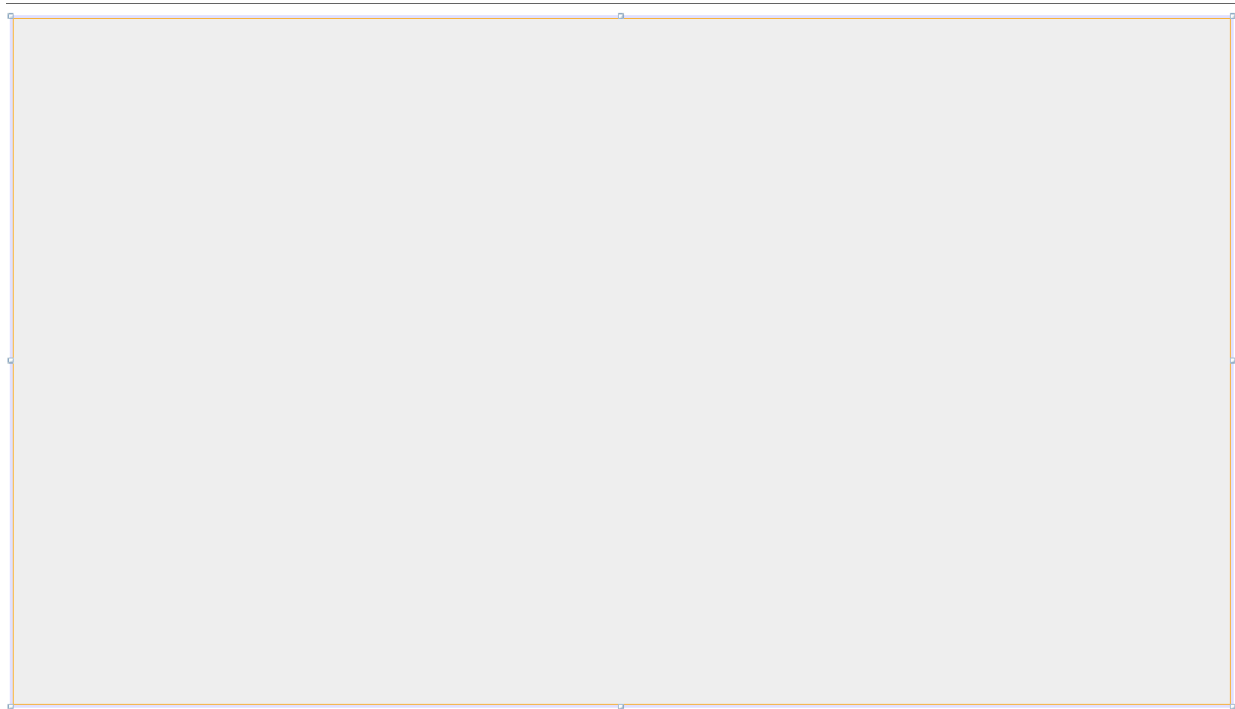
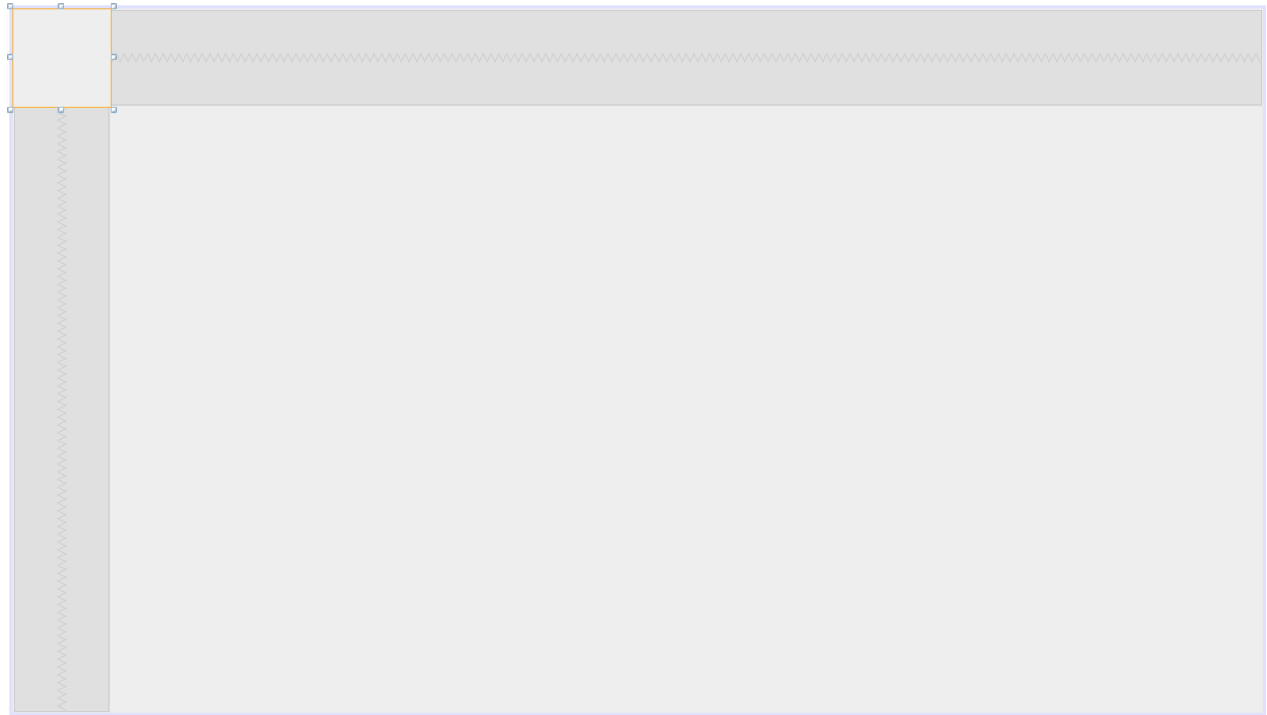
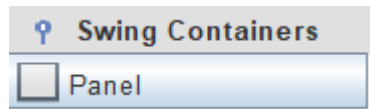


Main Frame Sizing

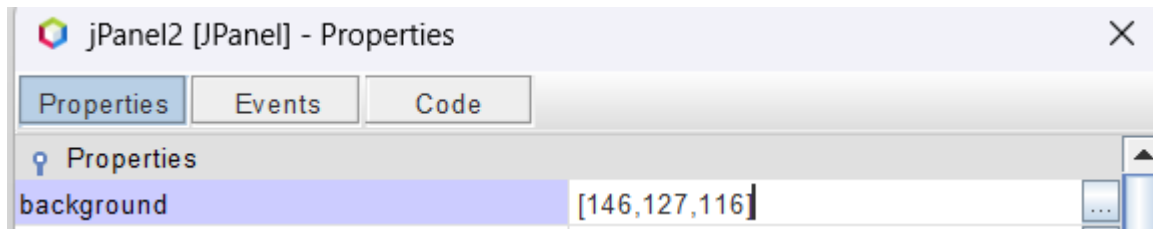
Kita akan mendapatkan JFrame berikut, ubah property code designer size menjadi 1280*720.



Setelah itu, tambahkan panel yang disediakan dari palet swing container, dan drag ke pojok kiri atas frame. Kemudian lebarkan panel seluas frame.

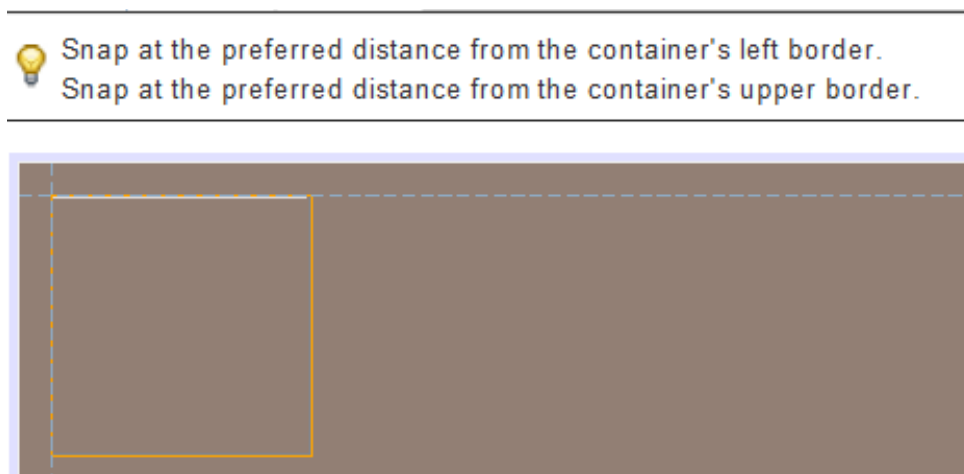


Kita bisa ubah warna background panel dengan cara mengubah property background color

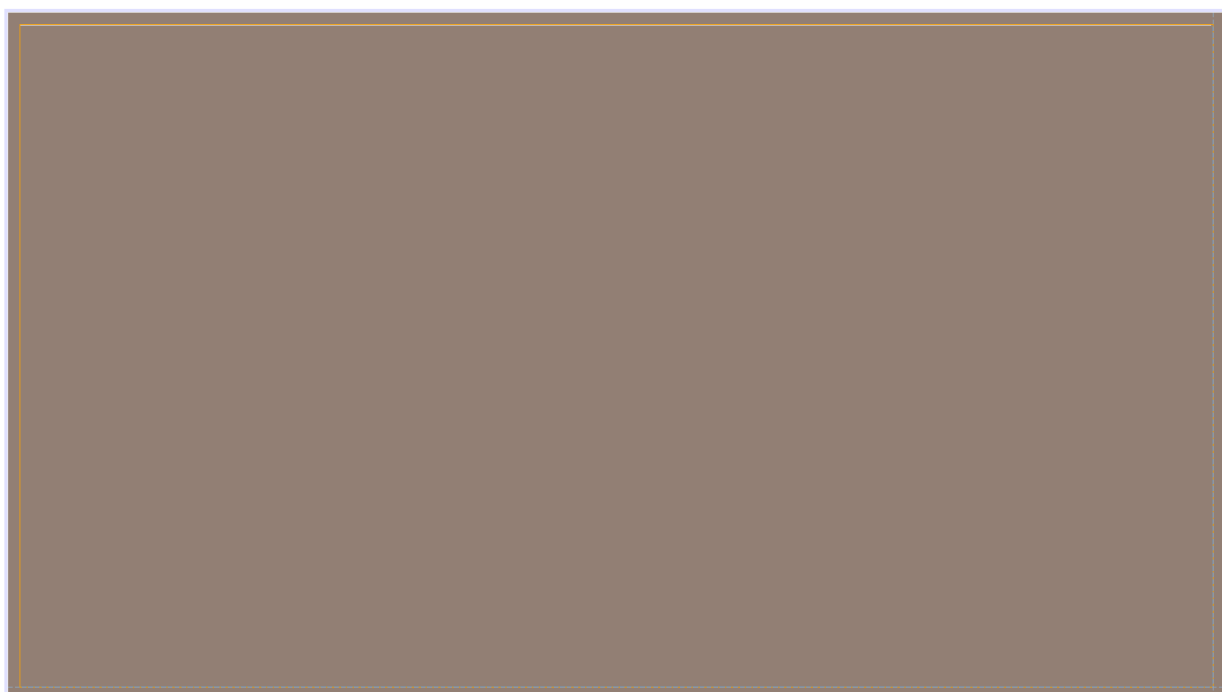


Tabbed Pane

Tambahkan lagi tabbed pane kemudian posisikan di kiri atas panel, pastikan terdapat simbol lampu dengan message pada contoh gambar di bawah.

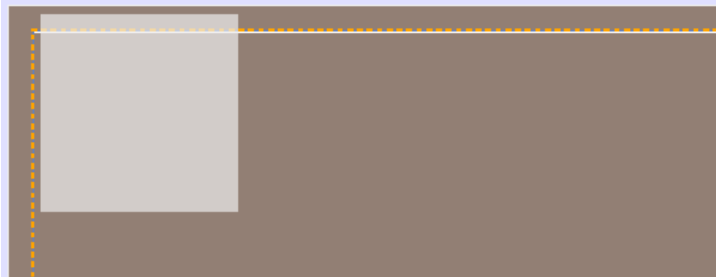


⚡ Snap at the preferred distance from the container's right border.
⚡ Snap at the preferred distance from the container's bottom.



Tambahkan panel biasa kedalam tabbed pane, dengan drag panel dari swing container, ke dalam tabbed pane. Tambahkan 2 panel ke dalam tabbed pane.

💡 Add the component as a tab into the JTabbedPane.



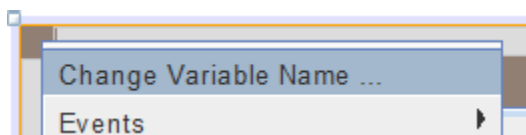
💡 Add the component as a tab into the JTabbedPane.



Ubah teks tab dengan nama Kendaraan dan Customer, klik kanan pada tab dan klik edit text.

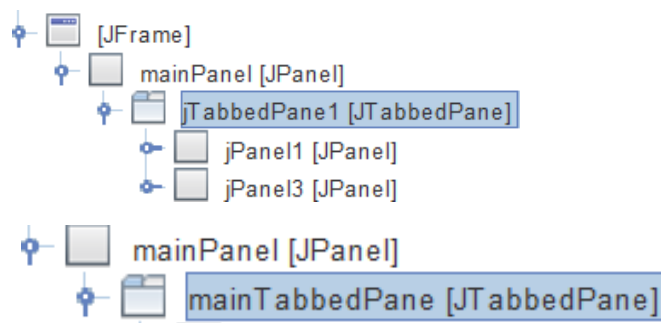


Setelah itu kita akan beri nama untuk semua komponen yang kita buat, dengan klik kanan komponen dan pilih change variable name.



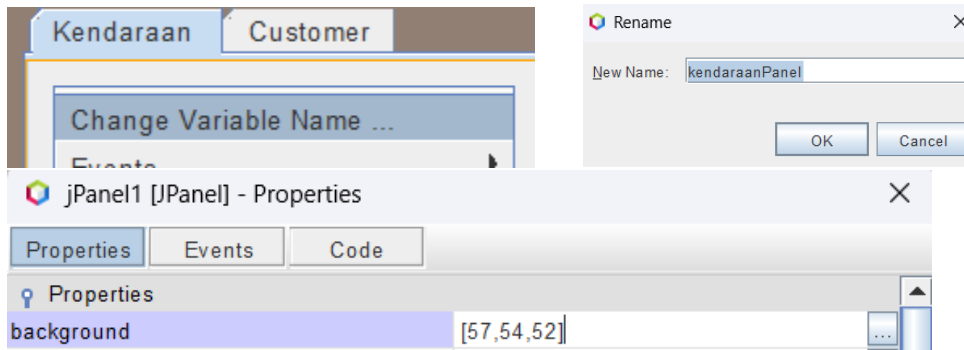
Klik pada ujung panel (pilih panel pertama) dan change variable name dengan nama mainPanel.

Buka navigator dan ubah nama jTabbedPane dengan nama mainTabbedPane.

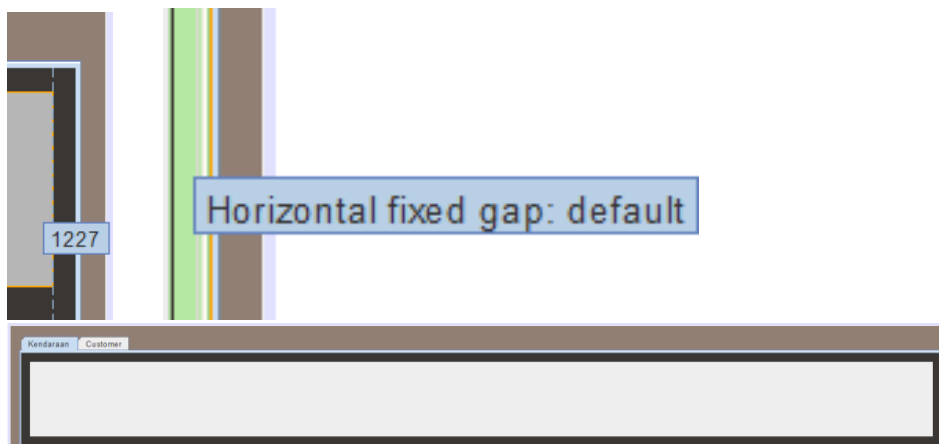


KendaraanPanel

Untuk panel kendaraan, kita perlu menamai jpanel pada tab kendaraan menjadi kendaraanPanel, pastikan kita sedang dalam tab kendaraan selama proses pengisian panel kendaraan. Dan beri warna pada panel kendaraan



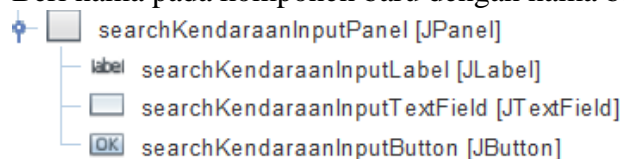
Kemudian kita akan menambahkan panel untuk pencarian terlebih dahulu. Pada palet, tambahkan panel dari swing containers dan letakan pada kiri atas panel. Lebarkan panel sampai hingga gap default. Untuk mengetahui jarak gap nya, klik pada gap antara panel search dan panel utama. Pastikan gapnya fixed, supaya lebih responsif.



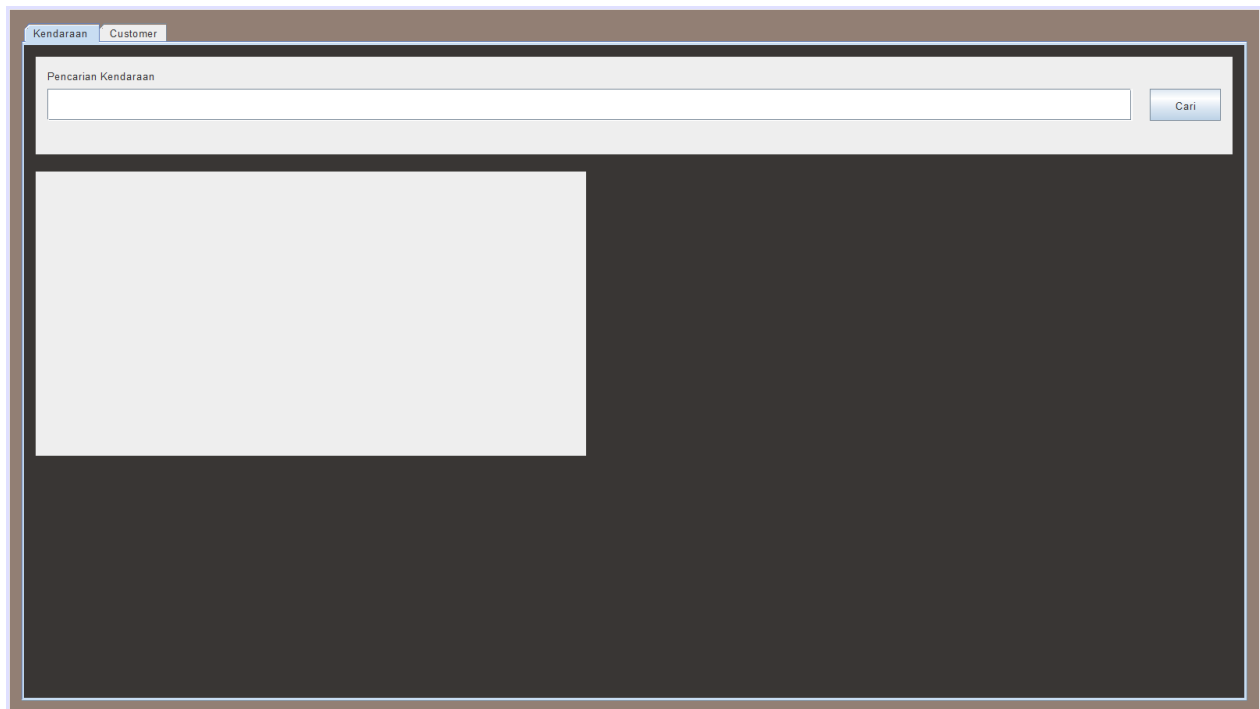
Tambahkan label, textField, dan button, menggunakan palet pada swing controls, dan beri nama pada panel. Edit text pada komponen sesuai gambar di bawah, dan jarak pada komponen selalu default, kecuali antara textField dan button dengan jarak default large.



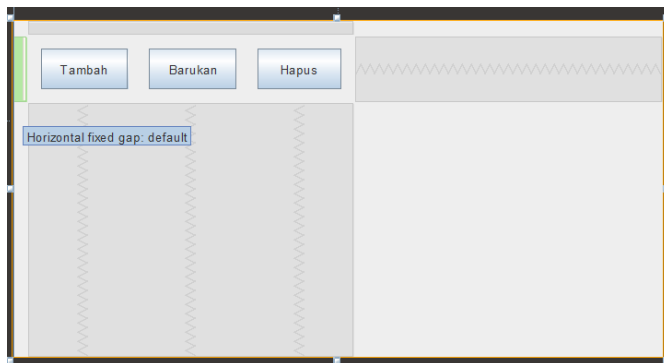
Beri nama pada komponen baru dengan nama berikut



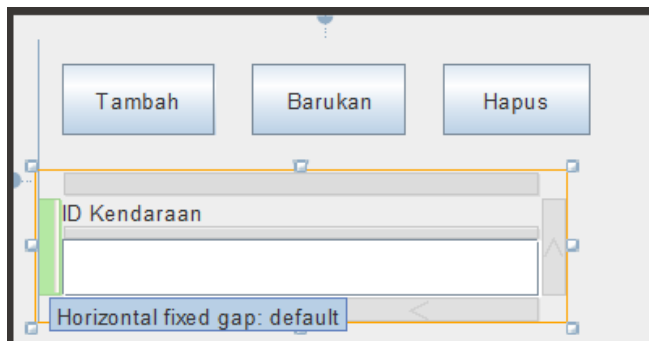
Setelah panel pencarian, kita bisa tambahkan form panel untuk kendaraan.



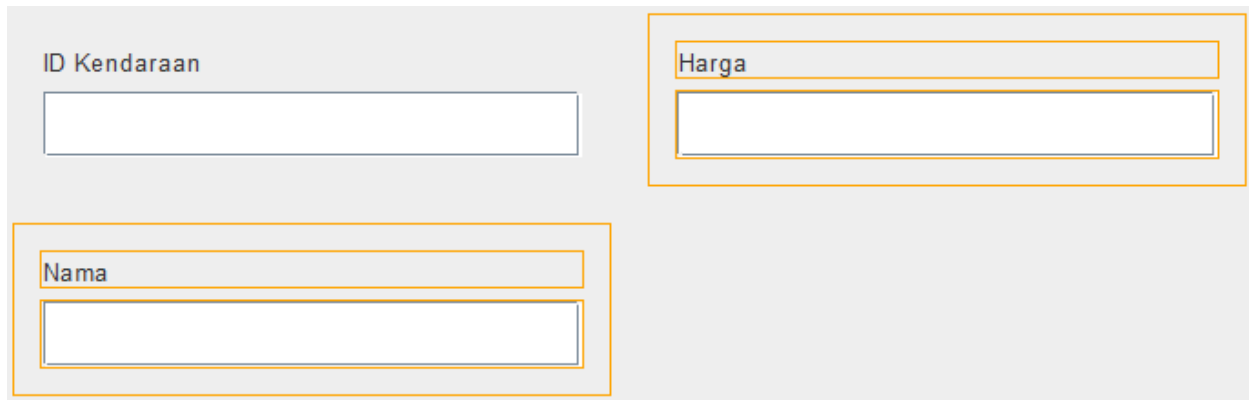
Tambahkan panel lagi untuk tombol create update delete. Edit teks pada tombol sesuai dengan gambar di bawah.



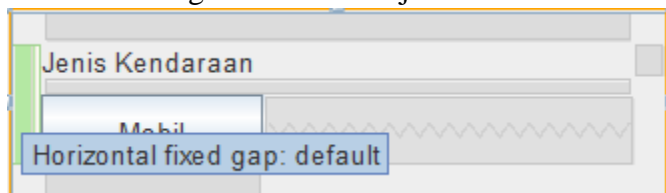
Tambahkan panel untuk input id



Tambahkan lagi panel untuk input nama dan harga, beri gap default large untuk jarak vertikal dan horizontal.



Tambahkan lagi tombol untuk jenis kendaraan.



Berikut gambar untuk keseluruhan form dan gap nya. Jika gapnya angka, maka tidak perlu diikuti persis, dan untuk gap seperti default, default large sebaiknya diikuti untuk mempermudah mengatur kerapihan ui.

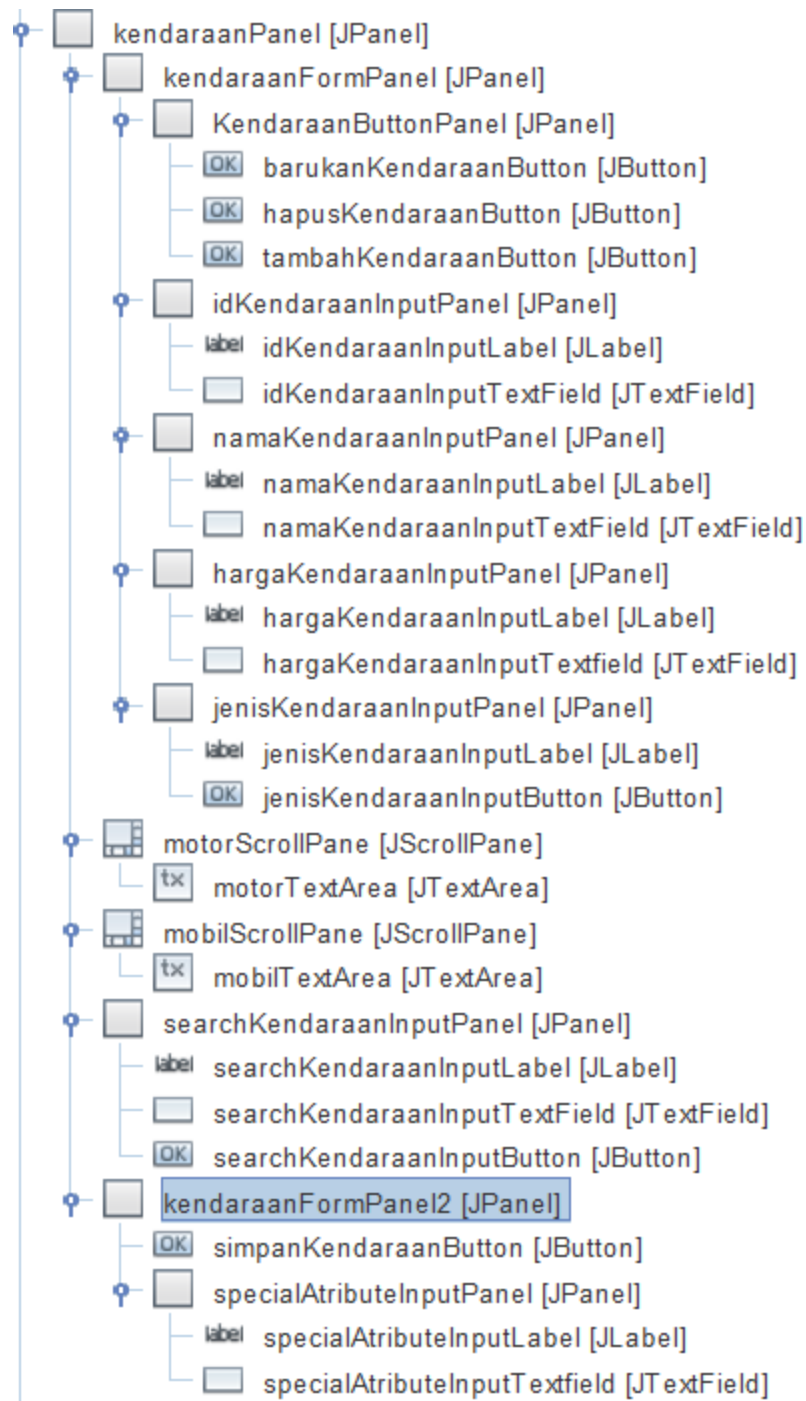


Tambahkan juga panel untuk input spesial dan tombol simpan sebagai berikut

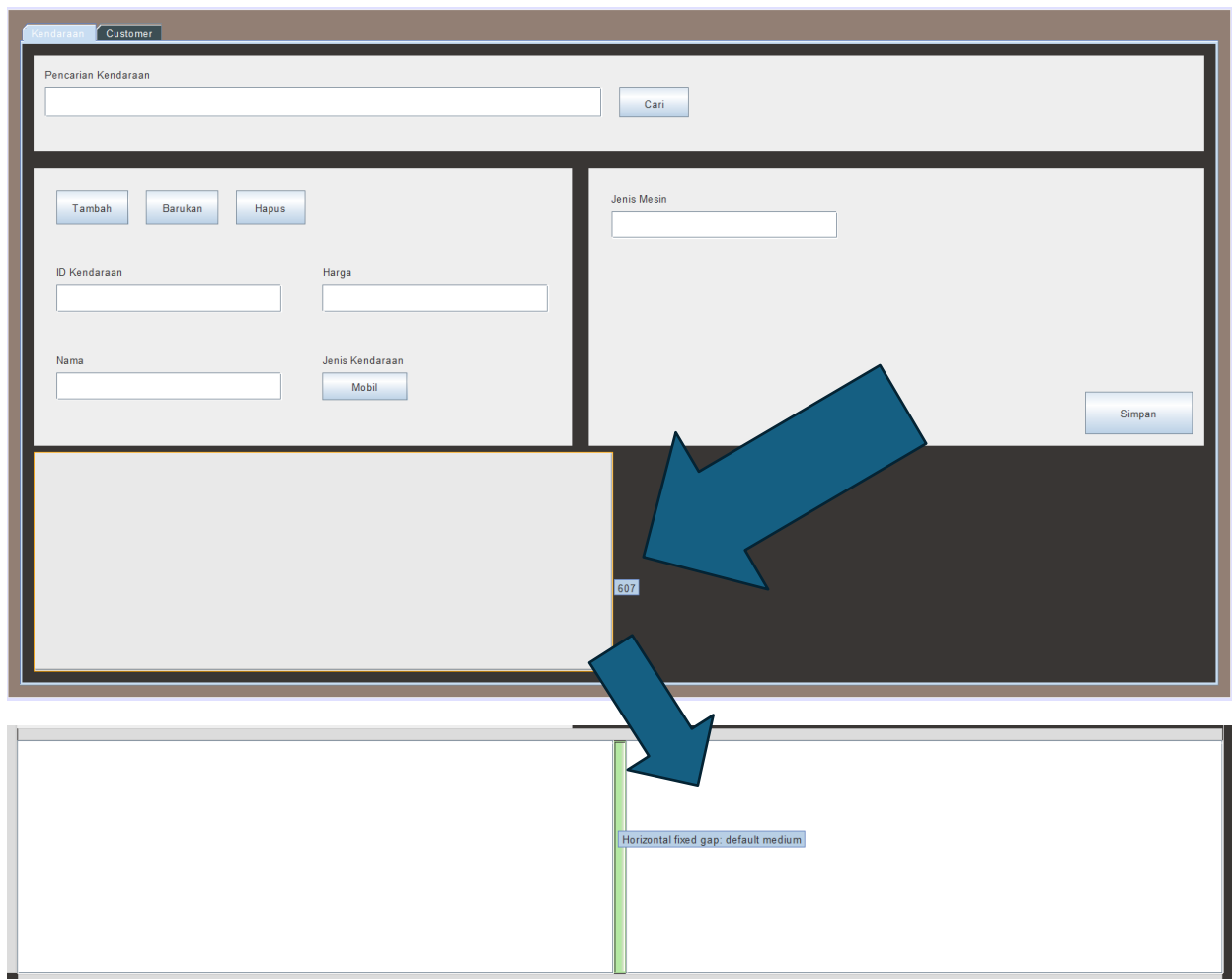
The image shows a web form layout within a light gray container. In the top-left corner, there is a label 'Jenis Mesin' in a small gray box, followed by a white input field with a thin orange border. The rest of the container is a large, empty light gray area. In the bottom-right corner, there is a blue button with the text 'Simpan' in white.

Silahkan sesuaikan jarak dengan default, default large, atau default lain sesuai dengan preferensi.

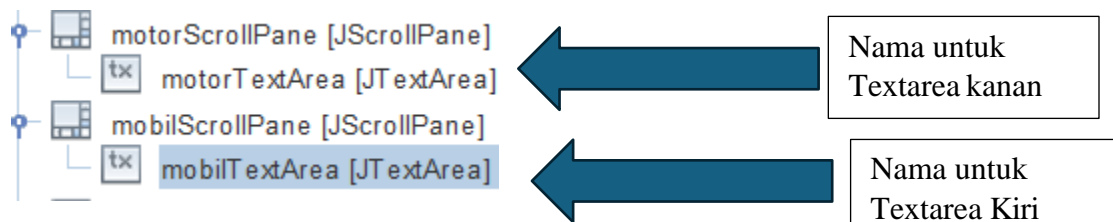
Beri nama pada panel sesuai dengan navigator berikut.



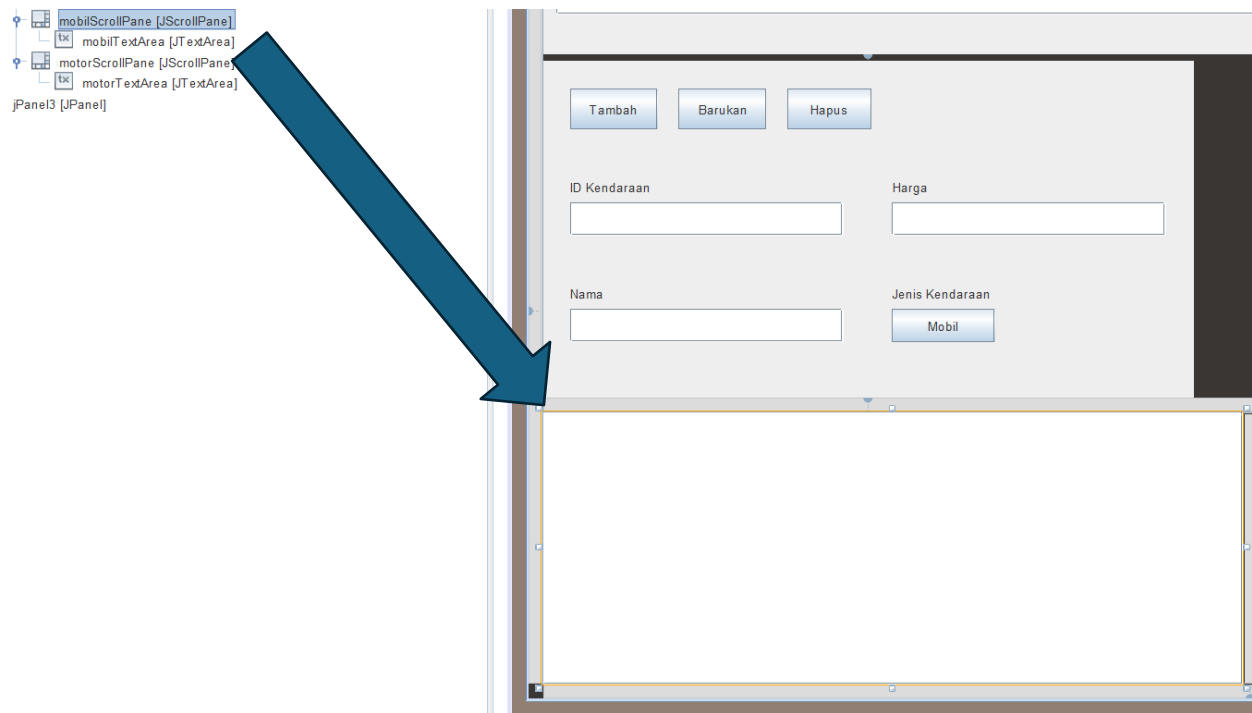
Tambahkan textArea untuk menampilkan data mobil dan motor secara terpisah.



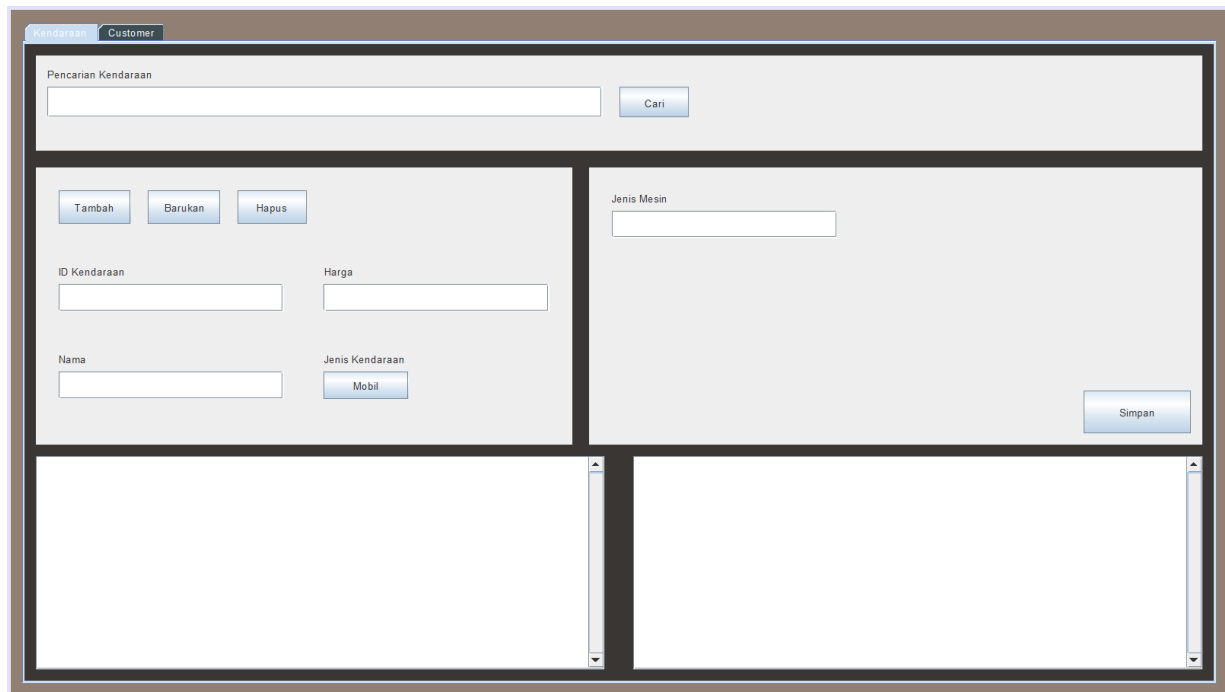
Kita akan mengatur scrollpane pada textArea, karena itu kita akan menamai textArea yang telah kita buat dengan nama berikut

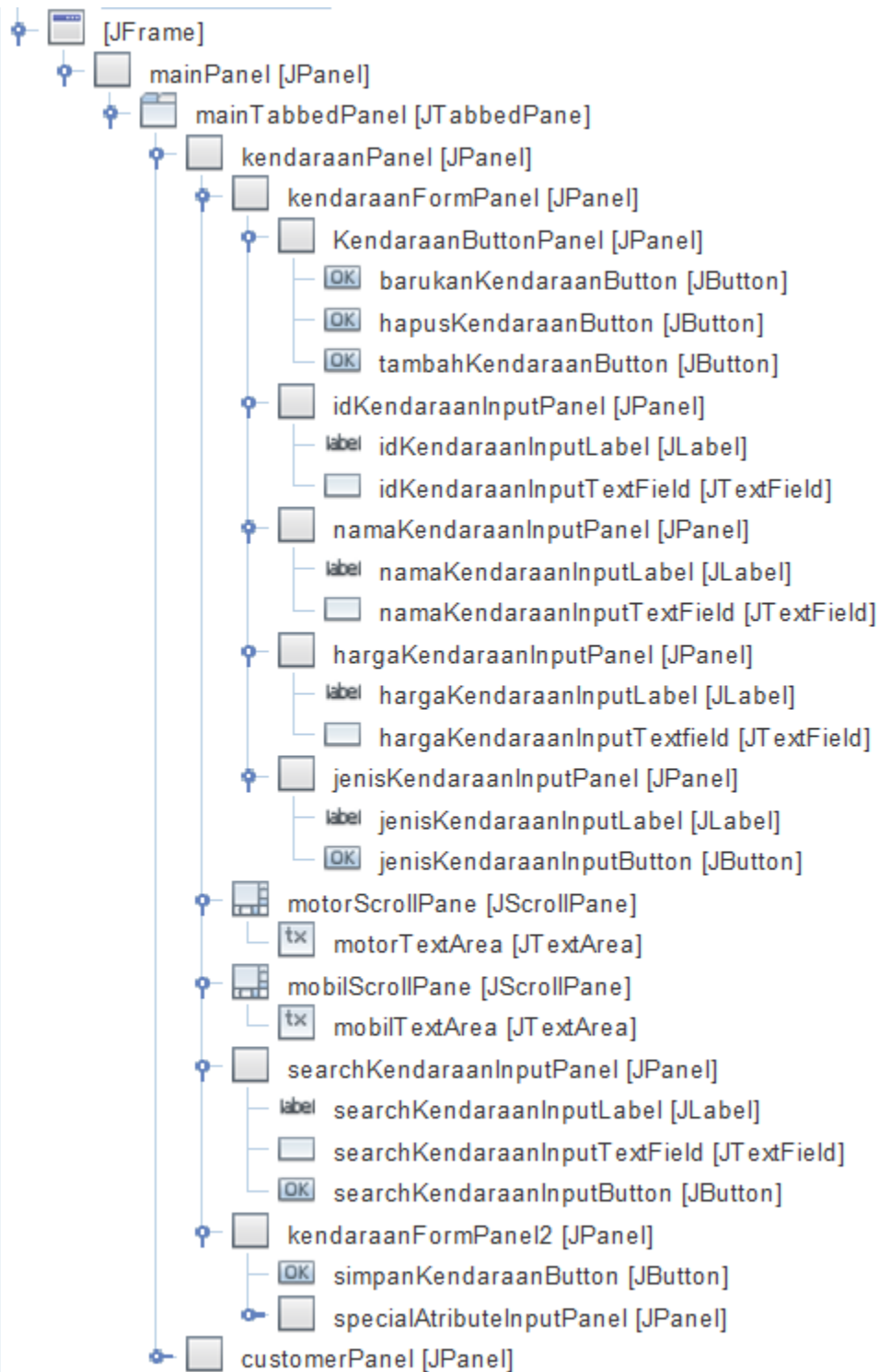


Perlu diperhatikan bahwa urutan navigator bisa berbeda beda, untuk memastikan komponen yang kita pilih benar, ketika kita klik komponen pada navigator, komponen pada design akan otomatis terpilih.

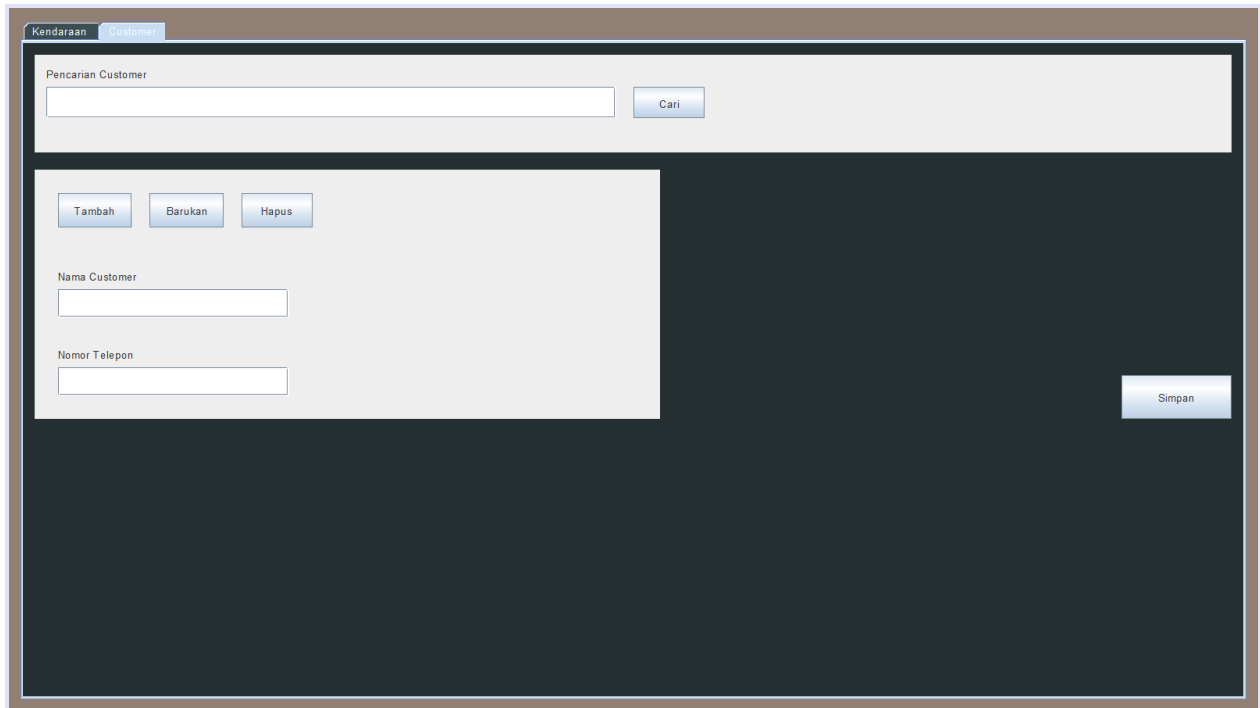


Secara keseluruhan tampilan dan navigator akan terlihat seperti ini

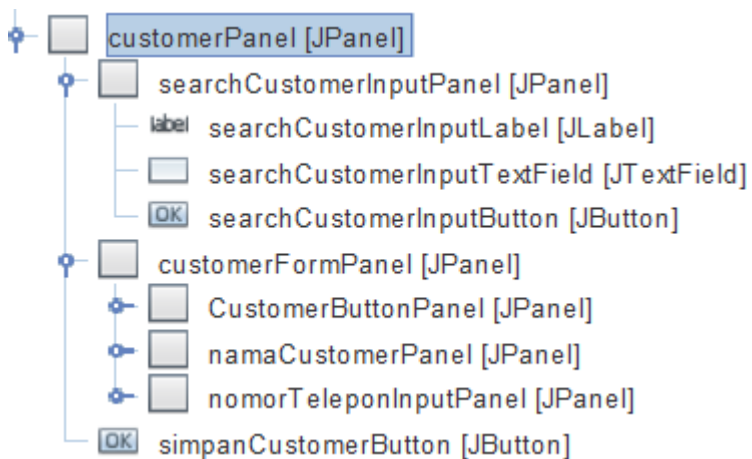




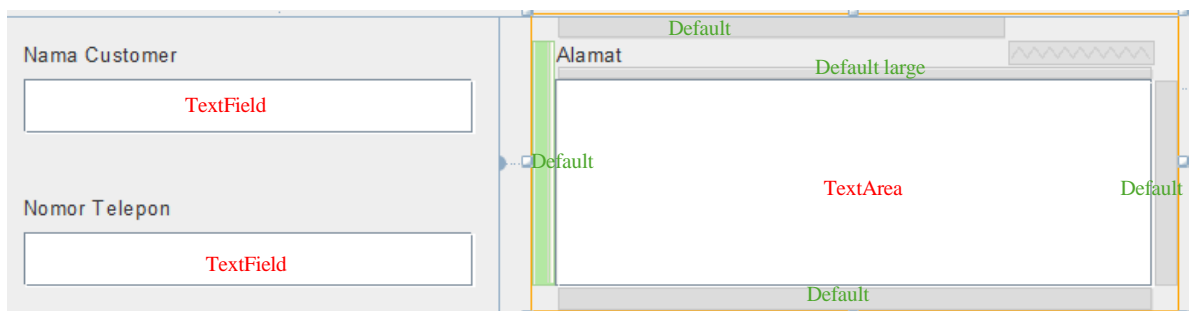
Lanjutkan ke pembuatan customer, buat pencarian dan form mirip dengan panel kendaraan



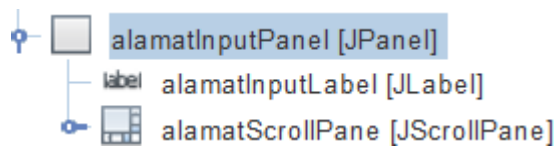
Beri nama pada panel dengan nama, sesuai navigator berikut.



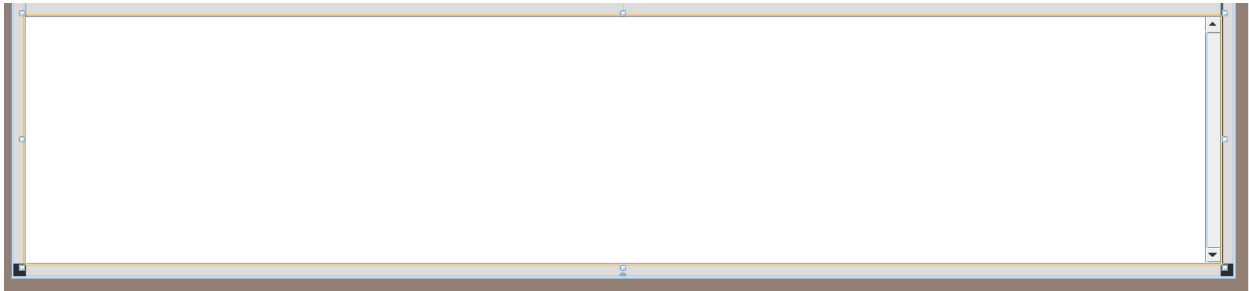
Sekarang kita hanya perlu menambahkan form khusus dan textArea untuk Customer



Berikut adalah navigator untuk form alamat



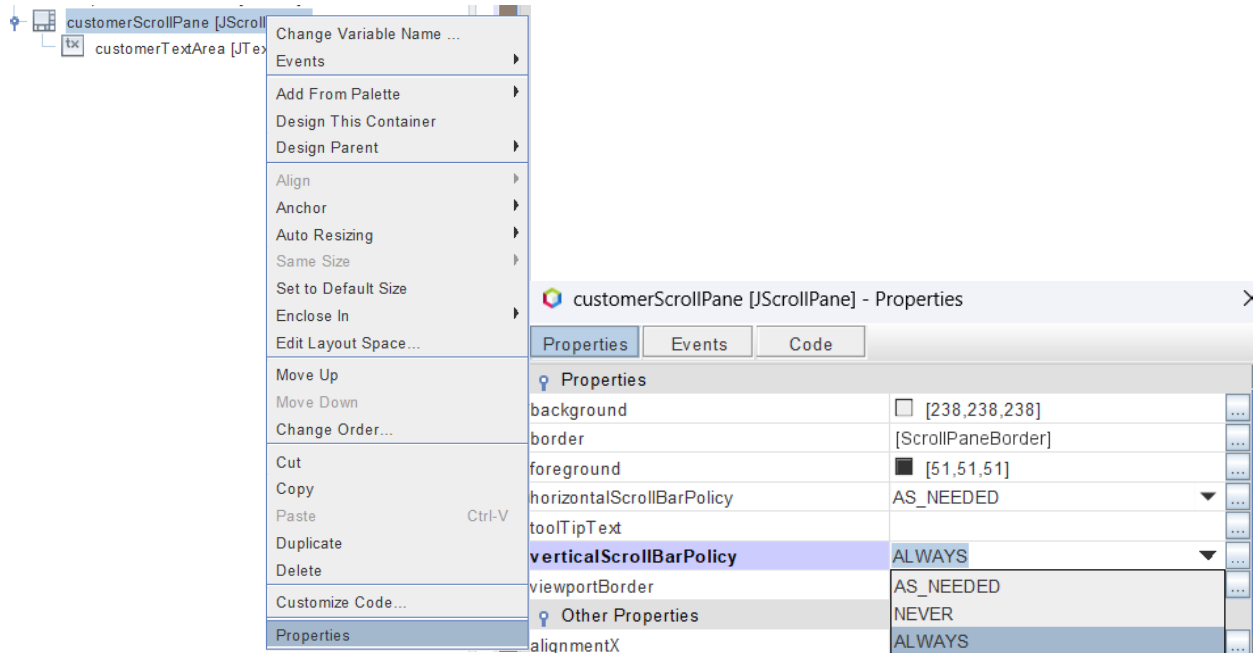
Tambahkan textArea untuk menampilkan data customer, ukuran lebar sampai ujung panel customer, dengan jarak default.



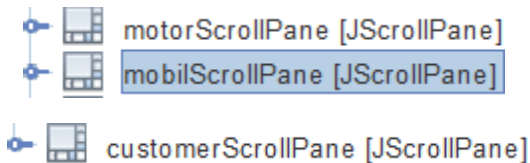
Maka, untuk tampilan Customer akan terlihat seperti gambar berikut

The screenshot displays a web application interface for managing customers. At the top, there are two tabs: "Kendaraan" and "Customer", with "Customer" being the active tab. Below the tabs, the interface is divided into several sections. On the left, there is a "Pencarian Customer" (Customer Search) section with a text input field and a "Cari" (Search) button. Below this, there are three buttons: "Tambah" (Add), "Barukan" (Refresh), and "Hapus" (Delete). The main area contains a form for adding or editing customer information. This form has three input fields: "Nama Customer" (Customer Name), "Nomor Telepon" (Phone Number), and "Alamat" (Address). The "Alamat" field is a larger text area. To the right of the form, there is a "Simpan" (Save) button. At the bottom of the interface, there is a large, empty rectangular area, likely intended for displaying a list of customers or a detailed view of a selected customer.

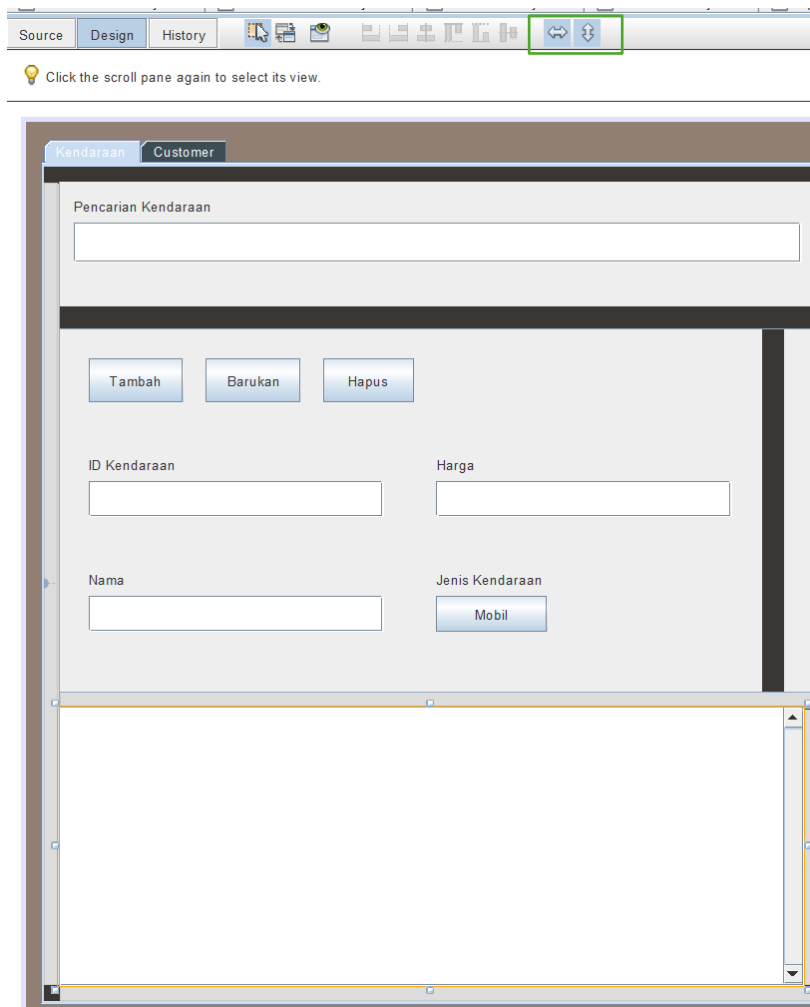
Untuk textArea yang menampilkan data object (Mobil, Motor dan Customer) bisa kita atur, supaya selalu menunjukkan scrollbar, dengan cara buka property, dan atur verticalScrollBarPolicy ke always.



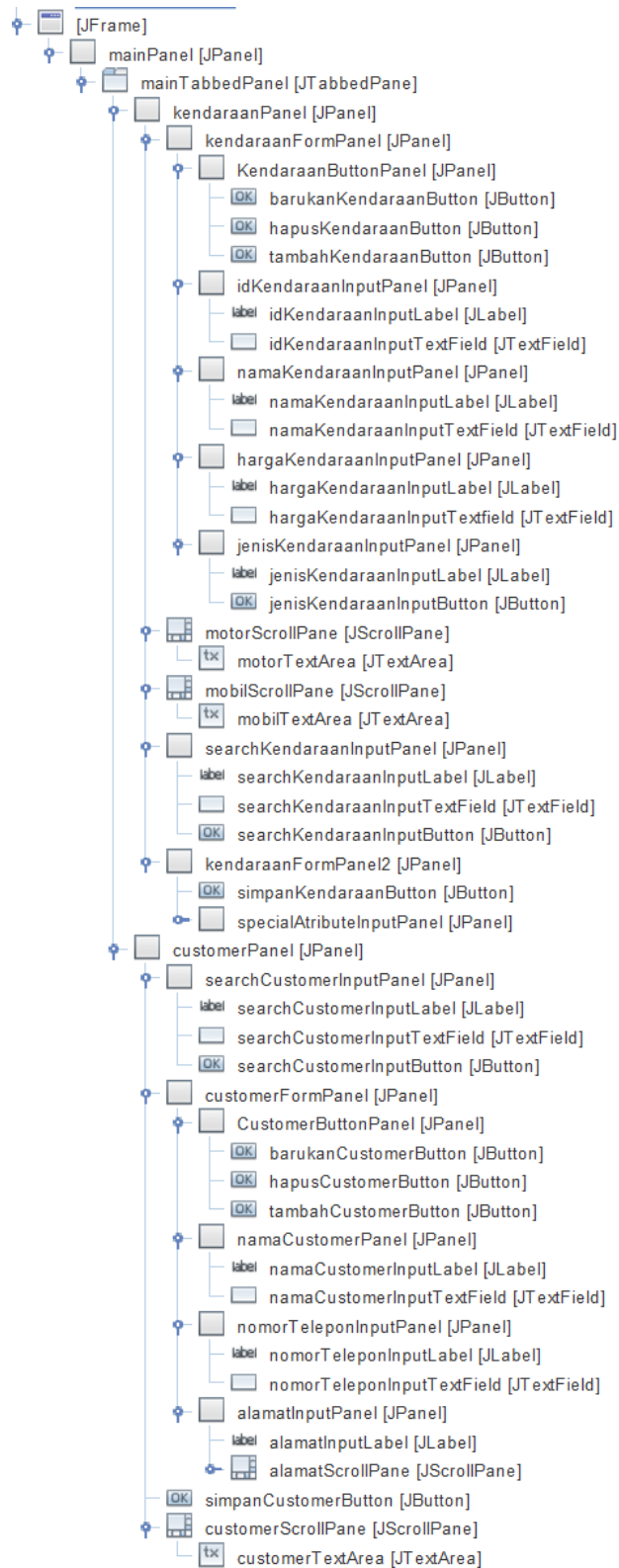
Lakukan pada semua scrollPane berikut



Pastikan juga untuk scrollpane resizeable dengan klik pada scrollpane, dan pastikan vertical dan horizontal resizeability di aktifkan.



Berikut adalah navigator untuk seluruh panel



MainView code

Klik source pada MainView, kemudian tambahkan import berikut

```
package view;

import control.CustomerControl;
import control.KendaraanControl;
import control.MobilControl;
import control.MotorControl;
import exception.*;
import javax.swing.JOptionPane;
import model.Kendaraan;
import model.Customer;
import model.Mobil;
import model.Motor;
```

Terdapat berbagai import dari control dan model, ini akan digunakan untuk mengakses model dan merepresentasikannya dalam UI.

Setelah menambahkan import, inisialisasikan juga beberapa variabel berikut.

```
private KendaraanControl kc = new KendaraanControl();
private MotorControl mtrc = new MotorControl();
private MobilControl mbic = new MobilControl();
private CustomerControl cc = new CustomerControl();
private Kendaraan k = null;
private Mobil mbl = null;
private Motor mtr = null;
private Customer c = null;
String actionKendaraan = null;

String action = null, actionCustomer = null;
int idCustomer = 0;
```

Selanjutnya kita tambahkan method, untuk melakukan throw exception jika terjadi kesalahan input user.

```
private void inputKosongKendaraanException() throws InputKosongException{
    if(namaKendaraanInputTextField.getText().isEmpty() || hargaKendaraanInputTextField.getText().isEmpty())
        throw new InputKosongException();
}

private void inputKosongCustomerException() throws InputKosongException{
    if(namaCustomerInputTextField.getText().isEmpty() || alamatInputTextField.getText().isEmpty() || nomorTeleponInputTextField.getText().isEmpty())
        throw new InputKosongException();
}

private void InputSpecialAtributeKendaraanException() throws InputSpecialAtributeException{
    if(jenisKendaraanInputButton.getText().equals("Motor") && !isInteger(specialAttributeInputTextField.getText()))
        throw new InputSpecialAtributeException();
}

private void noTelponException() throws NoTelponException{
    if(!nomorTeleponInputTextField.getText().matches("^\\d{0-9}+"))
        throw new NoTelponException();
}
```

Terdapat method khusus untuk memastikan bahwa input user merupakan integer.

```
public boolean isInteger(String str) {  
    try {  
        Integer.parseInt(str);  
        return true;  
    } catch (NumberFormatException e) {  
        return false;  
    }  
}
```

Method ini digunakan pada inputSpecialAttributeKendaraanException.

Isikan method berikut pada konstruktor MainView

```
public MainView() {  
    initComponents();  
    showKendaraan();  
    showCustomer();  
    setComponentsKendaraan(false);  
    setComponentsCustomer(false);  
  
    setKendaraanEditDeleteButton(false);  
    setEditDeleteButtonCustomer(false);  
  
    clearTextKendaraan();  
    clearTextCustomer();  
}
```

Lanjutkan dengan menambahkan method berikut

```
private void setComponentsKendaraan(boolean value){
    idKendaraanInputTextField.setEnabled(value);
    namaKendaraanInputTextField.setEnabled(value);
    hargaKendaraanInputTextfield.setEnabled(value);
    jenisKendaraanInputButton.setEnabled(value);
    specialAtributeInputTextfield.setEnabled(value);
    jenisKendaraanInputButton.setEnabled(value);
}

// Customer Set Komponen
private void setComponentsCustomer(boolean value){
    namaCustomerInputTextField.setEnabled(value);
    nomorTeleponInputTextField.setEnabled(value);
    alamatInputTextField.setEnabled(value);
}

private void setKendaraanEditDeleteButton(boolean value){
    barukanKendaraanButton.setEnabled(value);
    hapusKendaraanButton.setEnabled(value);
}

// Customer Set Edit Delete Button
private void setEditDeleteButtonCustomer(boolean value){
    barukanCustomerButton.setEnabled(value);
    hapusCustomerButton.setEnabled(value);
}

private void clearTextKendaraan(){
    idKendaraanInputTextField.setText("");
    namaKendaraanInputTextField.setText("");
    hargaKendaraanInputTextfield.setText("");
    jenisKendaraanInputButton.setText("");
    searchKendaraanInputTextField.setText("");
    specialAtributeInputTextfield.setText("");
    specialAtributeInputLabel.setText("");
}

// Customer Clear Text
private void clearTextCustomer(){
    namaCustomerInputTextField.setText("");
    nomorTeleponInputTextField.setText("");
    alamatInputTextField.setText("");
    searchCustomerInputTextField.setText("");
}

private void setSpecialAtributeLabel(){
    if(jenisKendaraanInputButton.getText().equals("Mobil")){
        specialAtributeInputLabel.setText("Jenis Mesin");
    }else{
        specialAtributeInputLabel.setText("Jumlah Tak");
    }
}
```



```

private void showKendaraan(){
    motorTextArea.setText("List Motor: \n ===== \n"+mtrc.showStringKendaraan());
    mobilTextArea.setText("List Mobil: \n ===== \n"+mblc.showStringKendaraan());
}

private void showCustomer(){
    customerTextArea.setText("List Customer: \n ===== \n"+cc.showAllStringCustomer());
}

private boolean mobilIsSelected(){
    return jenisKendaraanInputButton.getText().equals("Mobil");
}

private void doSearchKendaraan(){
    if(searchKendaraanInputTextField.getText().isEmpty())
        return;

    k = kc.searchDataKendaraan(searchKendaraanInputTextField.getText());

    if(k != null){
        setKendaraanEditDeleteButton(true);
        idKendaraanInputTextField.setText(k.getId_kendaraan());
        namaKendaraanInputTextField.setText(k.getNama());
        hargaKendaraanInputTextField.setText(Float.toString(k.getHarga()));
        jenisKendaraanInputButton.setText(k.getJenis());
        setSpecialAtributLabel();
        specialAttributeInputTextField.setText(k.getSpecial());
    }else{
        JOptionPane.showMessageDialog(rootPane, "NOT FOUND !!!");
    }
}

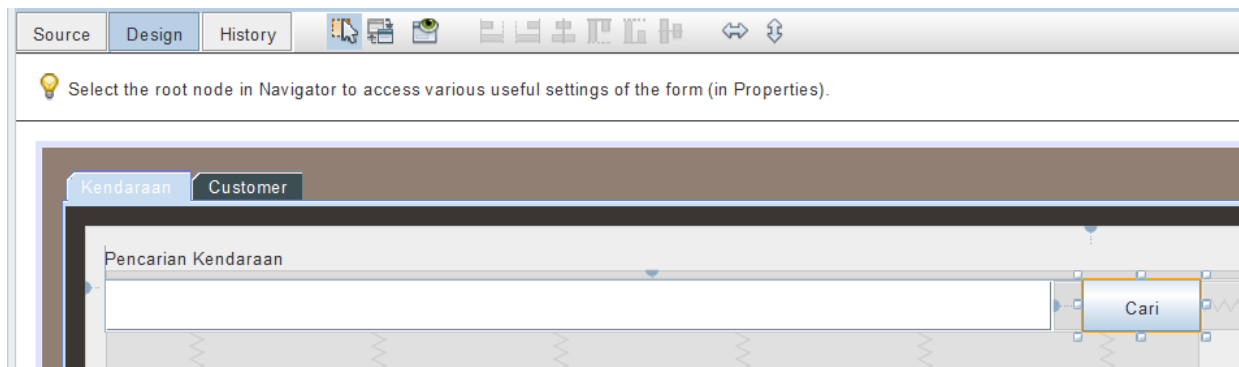
private void doSearchCustomer(){
    if(searchCustomerInputTextField.getText().isEmpty())
        return;
    Customer c = cc.searchCustomerById(Integer.parseInt(searchCustomerInputTextField.getText()));
    if(c == null ){
        JOptionPane.showMessageDialog(rootPane, "NOT FOUND !!!");
        return;
    }
    setEditDeleteButtonCustomer(true);
    clearTextCustomer();

    namaCustomerInputTextField.setText(c.getNama());
    nomorTeleponInputTextField.setText(c.getAlamat());
    alamatInputTextField.setText(c.getNo_telepon());

    idCustomer = c.getId_customer();
}

```

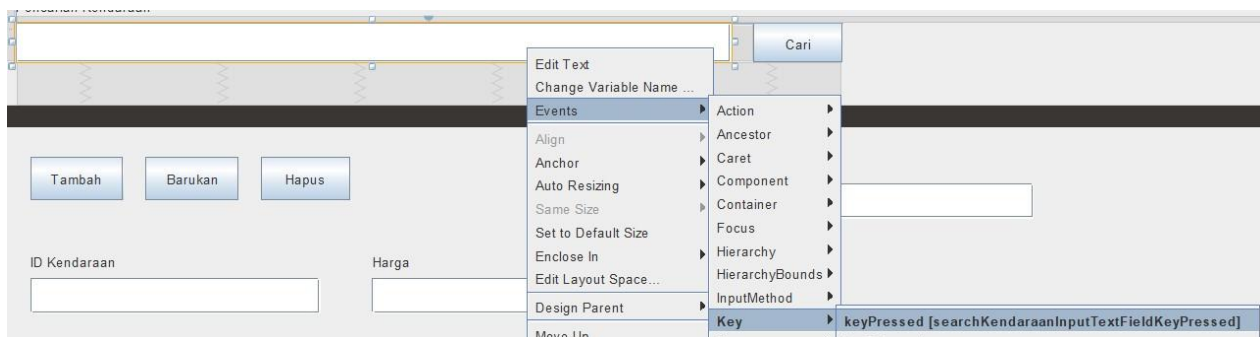
Setelah menambahkan method tersendiri pada kelas MainView, kita juga perlu memberi fungsi pada tombol, teks, dan sebagainya. Double click pada button cari di design MainView.



Kita akan diarahkan pada source aksi yang akan dilakukan ketika melakukan klik button cari. Isikan metod doSearchKendaraan.

```
private void searchKendaraanInputButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}
```

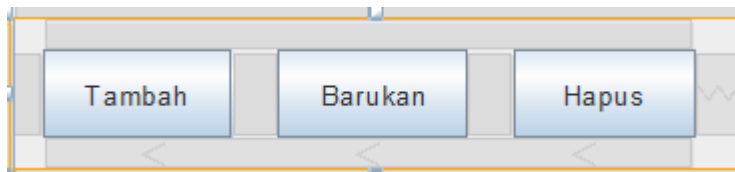
Kemudian klik kanan pada textField searchKendaraan, dan tambahkan event keyPressed.



Tambahkan method doSearchKendaraan juga pada event keyPressed, namun pastikan method dijalankan ketika tombol enter di tekan.

```
private void searchKendaraanInputTextFieldKeyPressed(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if(evt.getKeyChar()=='\n'){
        doSearchKendaraan();
    }
}
```

Lanjutkan menambah method pada tombol add, edit dan delete.



Double click pada tombol untuk menuju ke source code buttonActionPerformed.

```
private void tambahKendaraanButtonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    actionKendaraan = "add";  
    clearTextKendaraan();  
    setKendaraanEditDeleteButton(false);  
    setComponentsKendaraan(true);  
    setKendaraanEditDeleteButton(false);  
  
    idKendaraanInputTextField.setEnabled(false);  
    idKendaraanInputTextField.setText(kc.generateId());  
    jenisKendaraanInputButton.setText("Mobil");  
    setSpecialAtributeLabel();  
}
```

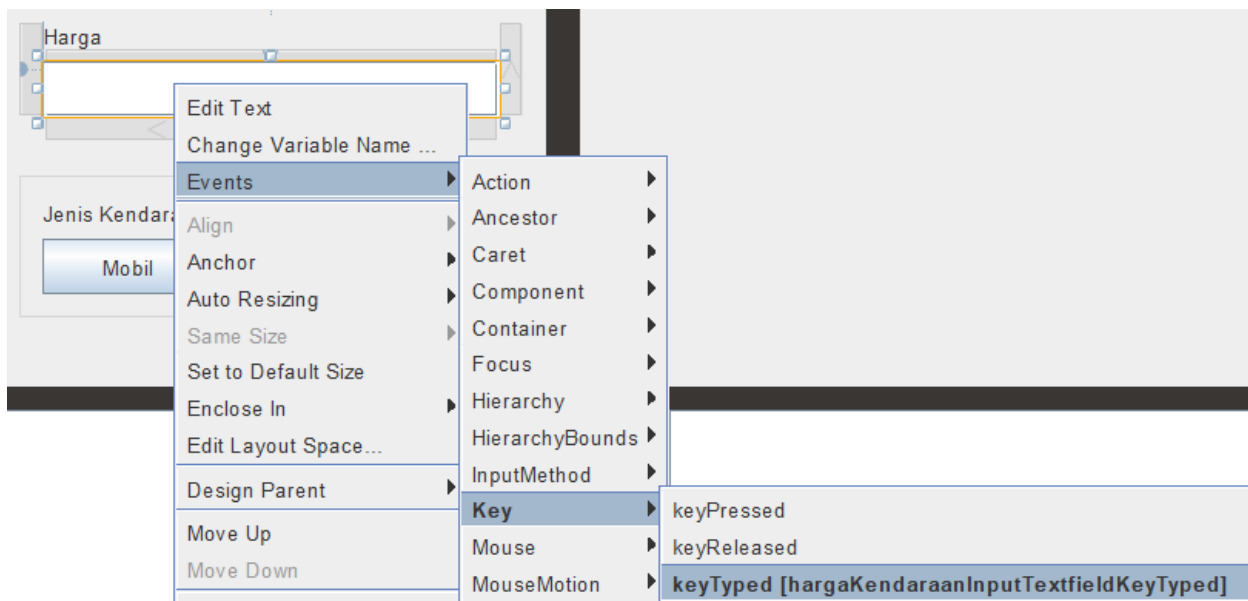
Method ini menambahkan state pada actionKendaraan, dimana state berada pada mode add, sehingga ketika melakukan klik simpan, aplikasi akan melakukan insert data ke database.

```
private void barukanKendaraanButtonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    actionKendaraan = "update";  
    setComponentsKendaraan(true);  
    idKendaraanInputTextField.setEnabled(false);  
}
```

Method barukan akan mengubah ke state edit. Method Delete akan melakukan konfirmasi, dan jika dikonfirmasi, data akan di hapus.

```
private void hapusKendaraanButtonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    int opsi = JOptionPane.showConfirmDialog(rootPane, "Yakin Ingin Hapus ?", "Hapus Data", JOptionPane.YES_NO_OPTION);  
    if( opsi == JOptionPane.NO_OPTION || opsi == JOptionPane.CLOSED_OPTION)  
        return;  
  
    kc.deleteDataKendaraan(idKendaraanInputTextField.getText());  
    clearTextKendaraan();  
    setKendaraanEditDeleteButton(false);  
    setComponentsKendaraan(false);  
    showKendaraan();  
}
```

Tambahkan juga event keyTyped pada form harga



```
private void hargaKendaraanInputTextfieldKeyTyped(java.awt.event.KeyEvent evt) {  
    // TODO add your handling code here:  
    char key = evt.getKeyChar();  
    if(!Character.isDigit(key) || key == '.'){  
        evt.consume();  
        JOptionPane.showMessageDialog(  
            null, "Hanya bisa masukan angka !!", "Input Failure", JOptionPane.ERROR_MESSAGE);  
    }  
}
```

Method digunakan untuk memastikan user tidak mengisi form input selain dengan angka. Setelah itu tambahkan aksi pada button jenis kendaraan.

```
private void jenisKendaraanInputButtonActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
    switch(jenisKendaraanInputButton.getText()){  
        case "Mobil": jenisKendaraanInputButton.setText("Motor"); break;  
        case "Motor": jenisKendaraanInputButton.setText("Mobil"); break;  
    }  
    setSpecialAtributeLabel();  
}
```

Method ini berfungsi untuk mengganti teks jenis kendaraan menjadi motor atau mobil. Selanjutnya, kita akan menambahkan aksi pada button simpan kendaraan.



```

private void simpanKendaraanButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    if(actionKendaraan == null)
        return;
    try{
        inputKosongKendaraanException();
        InputSpecialAttributeKendaraanException();

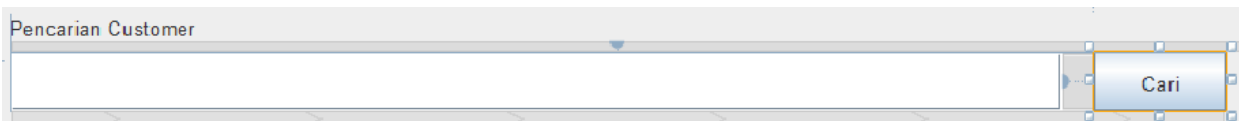
        int dialog = JOptionPane.showConfirmDialog(rootPane, "yakin ingin melakukan " + actionKendaraan + "?");
        if(dialog == JOptionPane.CLOSED_OPTION || dialog == JOptionPane.NO_OPTION || dialog == JOptionPane.CANCEL_OPTION)
            return;

        switch (actionKendaraan){
            case "add":
                if(mobilIsSelected()){
                    mbl = new Mobil(specialAttributeInputTextField.getText(), namaKendaraanInputTextField.getText(),
                                    jenisKendaraanInputButton.getText(), Float.parseFloat(hargaKendaraanInputTextField.getText()));
                    mblc.insertDataKendaraan(mbl);
                }else{
                    mtr = new Motor(Integer.parseInt(specialAttributeInputTextField.getText()), namaKendaraanInputTextField.getText(),
                                    jenisKendaraanInputButton.getText(), Float.parseFloat(hargaKendaraanInputTextField.getText()));
                    mtrc.insertDataKendaraan(mtr);
                }
                clearTextKendaraan();
                setKendaraanEditDeleteButton(false);
                setComponentsKendaraan(false);
                showKendaraan();
                break;
            case "update":
                if(mobilIsSelected()){
                    mbl = new Mobil(specialAttributeInputTextField.getText(), idKendaraanInputTextField.getText(), namaKendaraanInputTextField.getText(),
                                    jenisKendaraanInputButton.getText(), Float.parseFloat(hargaKendaraanInputTextField.getText()));
                    mblc.updateDataKendaraan(mbl);
                }else{
                    mtr = new Motor(Integer.parseInt(specialAttributeInputTextField.getText()), idKendaraanInputTextField.getText(), namaKendaraanInputTextField.getText(),
                                    jenisKendaraanInputButton.getText(), Float.parseFloat(hargaKendaraanInputTextField.getText()));
                    mtrc.updateDataKendaraan(mtr);
                }
                clearTextKendaraan();
                setKendaraanEditDeleteButton(false);
                setComponentsKendaraan(false);
                showKendaraan();
                break;
            default:
                break;
        }
        actionKendaraan = null;
    }catch(InputKosongException e){
        JOptionPane.showMessageDialog(rootPane, e.message());
    }catch(InputSpecialAttributeException e){
        JOptionPane.showMessageDialog(rootPane, e.message(jenisKendaraanInputButton.getText()));
    }
}
}

```

Tambahkan juga fungsi serupa pada panel customer.

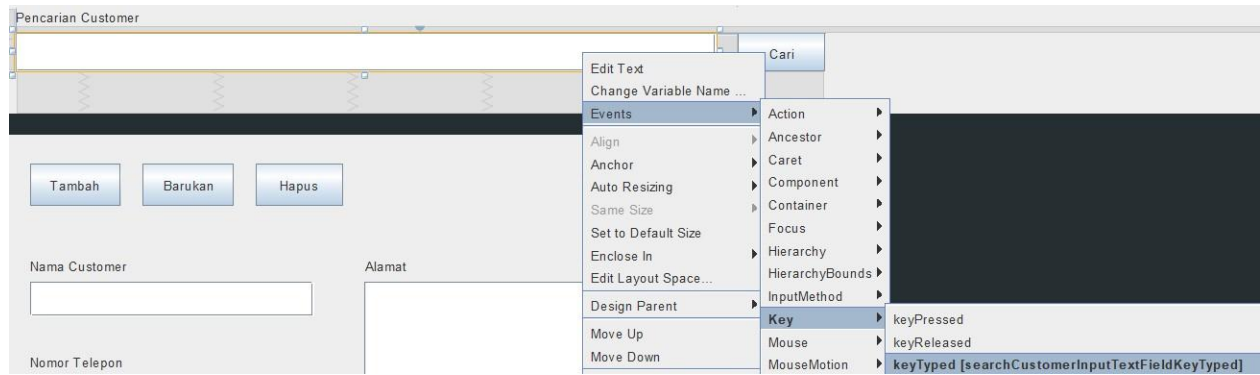
Search



```

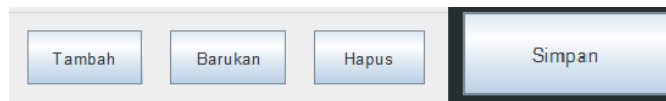
private void searchCustomerInputButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    doSearchCustomer();
}

```



```
private void searchCustomerInputTextFieldKeyTyped(java.awt.event.KeyEvent evt) {
    // TODO add your handling code here:
    if(evt.getKeyChar()=='\n'){
        doSearchCustomer();
    }
}
```

ADD,EDIT,DELETE, SAVE



```
private void tambahCustomerButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    actionCustomer = "add";
    clearTextCustomer();
    setEditDeleteButtonCustomer(true);
    setComponentsCustomer(true);
}
```

```

private void barukanCustomerButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:

    actionCustomer = "update";
    setComponentsCustomer(true);
}

private void hapusCustomerButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    clearTextCustomer();
    setEditDeleteButtonCustomer(false);
    setComponentsCustomer(false);

    int opsi = JOptionPane.showConfirmDialog(rootPane, "Yakin Ingin Hapus ?", "Hapus Data", JOptionPane.YES_NO_OPTION);
    if(opsi == JOptionPane.NO_OPTION || opsi == JOptionPane.CLOSED_OPTION){
        idCustomer = -1;
        return;
    }

    cc.deleteDataCustomer(idCustomer);
    idCustomer = -1;
    showCustomer();
}

private void simpanCustomerButtonActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    int dialog = JOptionPane.showConfirmDialog(rootPane, "yakin ingin melakukan " + actionKendaraan + "?");
    if(dialog == JOptionPane.CLOSED_OPTION || dialog == JOptionPane.NO_OPTION || dialog == JOptionPane.CANCEL_OPTION)
        return;

    try{
        inputKosongCustomerException();
        noTelponException();
        if(JOptionPane.showConfirmDialog(rootPane, "yakin ingin melakukan " + actionCustomer + "?") == JOptionPane.CLOSED_OPTION)
            return;
        switch(actionCustomer){
            case "add":
                c = new Customer(namaCustomerInputTextField.getText(), nomorTeleponInputTextField.getText(), alamatInputTextField.getText());
                cc.insertDataCustomer(c);
                clearTextCustomer();
                setEditDeleteButtonCustomer(false);
                setComponentsCustomer(false);
                break;
            case "update":
                c = new Customer(namaCustomerInputTextField.getText(), nomorTeleponInputTextField.getText(), alamatInputTextField.getText());
                cc.updateDataCustomer(c, idCustomer);
                clearTextCustomer();
                setEditDeleteButtonCustomer(false);
                setComponentsCustomer(false);
                break;
            default:
                break;
        }
        showCustomer();
        actionCustomer = null;
    }catch(InputKosongException e){
        System.out.println(e);
    }catch(NoTelponException e){
        JOptionPane.showMessageDialog(rootPane, e.message());
    };
}

```

Kentuan Pengumpulan Guided

Kumpulkan dalam ZIP dengan nama GD7_X_YYYYYY.zip

(X = Kelas, YYYYYY = 5 Digit terakhir NPM)