

**LAPORAN PROJEK TUGAS AKHIR**  
**PEMROGRAMAN DASAR 2**  
**(ABKC6303)**

**“Sistem Pendataan Ojek Pangkalan”**



**Dosen Pengampu :**

Dr. Harja Santana Purba M.Kom.

Ihdalhubbi Maulida, M.Kom

Nuruddin Wiranda, S.Kom., M.Cs

**Disusun Oleh :**

Indah Jumiatin (2310131220009)

**UNIVERSITAS LAMBUNG MANGKURAT BANJARMASIN**  
**FAKULTAS KEGURUAN DAN ILMU PENDIDIKAN**  
**PROGRAM STUDI PENDIDIKAN KOMPUTER**  
**2024**

## **KATA PENGANTAR**

Puji syukur kehadirat Allah SWT atas rahmat dan karunia-Nya sehingga laporan proyek tugas akhir mata kuliah Pemrograman Dasar 2 (ABKC6303) dengan judul **"Sistem Pendataan Ojek Pangkalan"** dapat diselesaikan dengan baik. Laporan ini dibuat sebagai salah satu syarat menyelesaikan mata kuliah Pemrograman Dasar 2 di Program Studi Pendidikan Komputer, Fakultas Keguruan dan Ilmu Pendidikan, Universitas Lambung Mangkurat.

Penulis mengucapkan terima kasih kepada Dr. Harja Santana Purba, M.Kom., Ihdalhubbi Maulida, M.Kom., dan Nuruddin Wiranda, S.Kom., M.Cs., selaku dosen pengampu yang telah memberikan bimbingan dan arahan selama proses penyusunan laporan ini. Ucapan terima kasih juga disampaikan kepada keluarga, teman-teman, dan semua pihak yang telah memberikan dukungan hingga laporan ini selesai.

Penulis menyadari bahwa laporan ini masih memiliki kekurangan. Oleh karena itu, saran dan kritik yang membangun sangat diharapkan untuk perbaikan di masa mendatang. Semoga laporan ini bermanfaat bagi pembaca dan dapat memberikan kontribusi dalam pengembangan ilmu pengetahuan, khususnya di bidang teknologi informasi.

Banjarmasin, 17 Desember 2024

Penulis

**Indah Jumiatin**  
(2310131220009)

## DAFTAR ISI

<b>DAFTAR ISI</b> .....	ii
<b>KATA PENGANTAR</b> .....	ii
<b>BAB I</b> .....	1
<b>PENDAHULUAN</b> .....	1
1.1    Latar Belakang .....	1
1.2    Rumusan Masalah .....	2
1.3    Tujuan.....	2
1.4    Manfaat.....	2
<b>BAB II</b> .....	3
<b>PEMBAHASAN</b> .....	3
2.1    Implementasi Java .....	3
2.2    Tabel Database .....	26
2.3    Hasil Akhir .....	27
<b>BAB III</b> .....	39
<b>PENUTUP</b> .....	39
3.1    Kesimpulan.....	39
3.2    Saran .....	39
<b>DAFTAR PUSTAKA</b> .....	40

# **BAB I**

## **PENDAHULUAN**

### **1.1 Latar Belakang**

Transportasi merupakan salah satu kebutuhan pokok dalam kehidupan masyarakat modern. Mobilitas yang tinggi membuat orang membutuhkan layanan transportasi yang cepat, praktis, dan terjangkau. Di antara berbagai jenis transportasi, ojek pangkalan menjadi pilihan yang sering digunakan, terutama di daerah lokal yang sulit dijangkau oleh transportasi umum lainnya. Keberadaan ojek pangkalan sangat membantu masyarakat dalam memenuhi kebutuhan transportasi jarak pendek, seperti perjalanan ke pasar, kantor, atau tempat lainnya.

Namun, pengelolaan ojek pangkalan masih banyak dilakukan secara manual. Misalnya, pencatatan data pengguna, driver, dan orderan dilakukan secara tidak terstruktur, sehingga rentan terhadap kehilangan data dan kesalahan pencatatan. Selain itu, proses pembayaran juga sering kali tidak tercatat dengan baik, yang bisa menyebabkan kebingungan atau ketidakjelasan dalam transaksi.

Dengan berkembangnya teknologi, sistem pendataan berbasis komputer menjadi solusi yang efektif untuk mengelola data ojek pangkalan. Dengan menggunakan bahasa pemrograman Java dan platform NetBeans, sistem ini dirancang untuk mencatat data pengguna, driver, orderan, dan pembayaran secara otomatis. Hal ini diharapkan dapat membuat proses lebih cepat dan mengurangi kesalahan.

## **1.2 Rumusan Masalah**

- a. Bagaimana cara mencatat data pengguna seperti nama, email, nomor HP, dan alamat secara mudah?
- b. Bagaimana cara mencatat data driver seperti nama, nomor HP, kendaraan, dan tarif dengan teratur?
- c. Bagaimana cara mencatat data orderan seperti lokasi awal, lokasi tujuan, dan waktu perjalanan?
- d. Bagaimana mencatat pembayaran dengan jelas, termasuk metode pembayaran dan total bayar?
- e. Bagaimana membuat sistem pendataan ini mudah digunakan dan efisien?

## **1.3 Tujuan**

- a. Membuat sistem untuk mencatat data pengguna.
- b. Mencatat data driver dengan rapi.
- c. Mengelola data orderan dengan mudah.
- d. Mencatat pembayaran dengan jelas.
- e. Membantu pengelolaan ojek pangkalan lebih efisien.

## **1.4 Manfaat**

- a. Mempermudah pengelolaan data pengguna, driver, dan orderan.
- b. Membantu driver bekerja lebih teratur.
- c. Memberikan layanan yang lebih nyaman bagi pengguna.
- d. Mengurangi kesalahan pencatatan manual.
- e. Menambah pengalaman dalam pengembangan sistem berbasis Java

## BAB II

### PEMBAHASAN

#### 2.1 Implementasi Java

##### A. Form Pengguna

##### - Kelas Utama dan Method Form Pengguna

```
public class FormPengguna extends javax.swing.JFrame {
    private DefaultTableModel model;

    private void kosongkan_form() {
        tfNama.setText("");
        tfEmail.setText("");
        tfNoHP.setText("");
        tfAlamat.setText("");
        caridata.setText("");
    }

    private void tampilkan_data() {
        model.setRowCount(0);
        try (Connection conn = Koneksi.getKoneksi(); Statement stm = conn.createStatement()) {
            String sql = "SELECT id_pengguna, nama_pengguna, email, no_hp, alamat FROM pengguna";
            ResultSet res = stm.executeQuery(sql);
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("id_pengguna"),
                    res.getString("nama_pengguna"),
                    res.getString("email"),
                    res.getString("no_hp"),
                    res.getString("alamat")
                });
            }
            tablepengguna.setModel(model);
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    private void cariData() {
        model.setRowCount(0);
        String pilihan = cbxcaridata.getSelectedItem().toString();
        String keyword = caridata.getText().trim();

        if (keyword.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Silakan masukkan kata kunci untuk pencarian.", "Peringatan", JOptionPane.WARNING_MESSAGE);
            return;
        }

        String sql;
        if (pilihan.equals("id_pengguna")) {
            sql = "SELECT * FROM pengguna WHERE id_pengguna = ?";
        } else {
            sql = "SELECT * FROM pengguna WHERE nama_pengguna LIKE ?";
        }

        try (Connection conn = Koneksi.getKoneksi(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
            if (pilihan.equals("id_pengguna")) {
                if (!keyword.matches("\\d+")) {
                    JOptionPane.showMessageDialog(null, "ID Pengguna harus berupa angka.", "Peringatan", JOptionPane.WARNING_MESSAGE);
                    return;
                }
                pstmt.setInt(1, Integer.parseInt(keyword));
            } else {
                pstmt.setString(1, "%" + keyword + "%");
            }

            ResultSet res = pstmt.executeQuery();
        }
    }
}
```

```

        if (!res.isBeforeFirst()) {
            JOptionPane.showMessageDialog(null, "Data tidak ditemukan.", "Informasi", JOptionPane.INFORMATION_MESSAGE);
        } else {
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("id_pengguna"),
                    res.getString("nama_pengguna"),
                    res.getString("email"),
                    res.getString("no_hp"),
                    res.getString("alamat")
                });
            }
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Error: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
    }
}

```

## - Konstruktor Form Pengguna

```

public FormPengguna() {
    initComponents();
    this.setLocationRelativeTo(null);
    model = new DefaultTableModel();
    tablepengguna.setModel(model);

    model.addColumn("ID_Pengguna");
    model.addColumn("Nama_Pengguna");
    model.addColumn("Email");
    model.addColumn("No HP");
    model.addColumn("Alamat");

    tampilkan_data();
    btnEdit.setEnabled(false);
    btnHapus.setEnabled(false);
    btnBatal.setEnabled(false);
}

```

## - Tombol Tambah

```

private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    String namaPengguna = tfNama.getText();
    String email = tfEmail.getText();
    String noHP = tfNoHP.getText();
    String alamat = tfAlamat.getText();

    String sql = "INSERT INTO pengguna (nama_pengguna, email, no_hp, alamat) VALUES (?, ?, ?, ?)";
    try (Connection c = Koneksi.getKoneksi(); PreparedStatement p = c.prepareStatement(sql)) {
        p.setString(1, namaPengguna);
        p.setString(2, email);
        p.setString(3, noHP);
        p.setString(4, alamat);
        p.executeUpdate();
        JOptionPane.showMessageDialog(null, "Data Tersimpan");
        tampilkan_data(); // Tampilkan data setelah penambahan
    } catch (SQLException e) {
        System.out.println("Terjadi Kesalahan: " + e.getMessage());
    } finally {
        kosongkan_form();
    }
}

```

## - Tombol Edit

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {  
  
    try {  
        int selectedRow = tablepengguna.getSelectedRow();  
        if (selectedRow == -1) {  
            JOptionPane.showMessageDialog(null, "Silakan pilih data dari tabel terlebih dahulu!", "Error", JOptionPane.ERROR_MESSAGE);  
            return;  
        }  
  
        int id_pengguna = Integer.parseInt(tablepengguna.getValueAt(selectedRow, 0).toString());  
        String nama_pengguna = tfNama.getText();  
        String email = tfEmail.getText();  
        String no_hp = tfNoHP.getText();  
        String alamat = tfAlamat.getText();  
  
        String sql = "UPDATE pengguna SET nama_pengguna = ?, email = ?, no_hp = ?, alamat = ? WHERE id_pengguna = ?";  
  
        // Koneksi ke database  
        java.sql.Connection conn = Koneksi.getKoneksi();  
        java.sql.PreparedStatement pstmt = conn.prepareStatement(sql);  
  
        pstmt.setString(1, nama_pengguna);  
        pstmt.setString(2, email);  
        pstmt.setString(3, no_hp);  
        pstmt.setString(4, alamat);  
        pstmt.setInt(5, id_pengguna);  
  
        pstmt.executeUpdate();  
  
        JOptionPane.showMessageDialog(null, "Data berhasil diubah!");  
        tampilkan_data();  
  
        kosongkan_form();  
    } catch (SQLException e) {  
        JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);  
    } catch (NumberFormatException e) {  
        JOptionPane.showMessageDialog(this, "ID Pengguna harus berupa angka!", "Error", JOptionPane.ERROR_MESSAGE);  
    }  
}
```

## - Tombol Hapus

```
private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {  
    try {  
        String nama_pengguna = tfNama.getText();  
  
        String sql = "DELETE from PENGGUNA WHERE nama_pengguna = '" + nama_pengguna + "'";  
  
        java.sql.Connection conn = (Connection)Koneksi.getKoneksi();  
        java.sql.PreparedStatement pstmt = conn.prepareStatement(sql);  
        pstmt.executeUpdate();  
        JOptionPane.showMessageDialog(null, "Data berhasil dihapus!");  
        tampilkan_data();  
        kosongkan_form();  
    } catch (SQLException e) {  
        JOptionPane.showMessageDialog(this, e.getMessage());  
    }  
}
```

## - Tombol Batal

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {  
    kosongkan_form();  
    tampilkan_data();  
    btnTambah.setEnabled(true);  
    btnEdit.setEnabled(false);  
    btnHapus.setEnabled(false);  
    btnBatal.setEnabled(false);  
}
```



## - Tabel

```
private void tablepenggunaMouseClicked(java.awt.event.MouseEvent evt) {
    btnTambah.setEnabled(false);
    btnEdit.setEnabled(true);
    btnHapus.setEnabled(true);
    btnBatal.setEnabled(true);

    int i = tablepengguna.getSelectedRow();
    if (i == -1) {
        return;
    }

    DefaultTableModel model = (DefaultTableModel) tablepengguna.getModel();

    String nama_pengguna = (String) (model.getValueAt(i, 1) != null ? model.getValueAt(i, 1).toString() : "");
    String email = (String) (model.getValueAt(i, 2) != null ? model.getValueAt(i, 2).toString() : "");
    String no_hp = (String) (model.getValueAt(i, 3) != null ? model.getValueAt(i, 3).toString() : "");
    String alamat = (String) (model.getValueAt(i, 4) != null ? model.getValueAt(i, 4).toString() : "");

    tfNama.setText(nama_pengguna);
    tfEmail.setText(email);
    tfNoHP.setText(no_hp);
    tfAlamat.setText(alamat);
}
```

## B. Form Driver

```
public class FormDriver extends javax.swing.JFrame {
    private DefaultTableModel model;

    private void kosongkan_form(){
        tfNama.setText("");
        tfNoHP.setText("");
        tfKendaraan.setText("");
        tfPlatNomor.setText("");
        tfTarif.setText("");
    }

    private void tampilkan_data(){
        model.setRowCount(0);
        try (Connection conn = Koneksi.getKoneksi(); Statement stm = conn.createStatement()) {
            String sql = "SELECT id_driver, nama_driver, no_hp, kendaraan, plat_nomor, tarif FROM driver";
            ResultSet res = stm.executeQuery(sql);
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("id_driver"),
                    res.getString("nama_driver"),
                    res.getString("no_hp"),
                    res.getString("kendaraan"),
                    res.getString("plat_nomor"),
                    res.getBigDecimal("tarif")
                });
            }
            tabledriver.setModel(model);
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }
}
```

```

private void cariData() {
    model.setRowCount(0);
    String pilihan = cbxcariData.getSelectedItem().toString();
    String keyword = cariData.getText().trim();

    if (keyword.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Silakan masukkan kata kunci untuk pencarian.", "Peringatan", JOptionPane.WARNING_MESSAGE);
        return;
    }

    String sql;
    if (pilihan.equals("id_driver")) {
        sql = "SELECT * FROM driver WHERE id_driver = ?";
    } else {
        sql = "SELECT * FROM driver WHERE nama_driver LIKE ?";
    }

    try (Connection conn = Koneksi.getKoneksi(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
        if (pilihan.equals("id_driver")) {
            if (!keyword.matches("\\d+")) {
                JOptionPane.showMessageDialog(null, "ID Driver harus berupa angka.", "Peringatan", JOptionPane.WARNING_MESSAGE);
                return;
            }
            pstmt.setInt(1, Integer.parseInt(keyword));
        } else {
            pstmt.setString(1, "%" + keyword + "%");
        }

        ResultSet res = pstmt.executeQuery();

        if (!res.isBeforeFirst()) {
            JOptionPane.showMessageDialog(null, "Data tidak ditemukan.", "Informasi", JOptionPane.INFORMATION_MESSAGE);
        } else {
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("id_driver"),
                    res.getString("nama_driver"),
                    res.getString("no_hp"),
                    res.getString("kendaraan"),
                    res.getString("plat_nomor"),
                    res.getBigDecimal("tarif")
                });
            }
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Error: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
    }
}

```

## - Konstruktor Form Driver

```

public FormDriver() {
    initComponents();
    this.setLocationRelativeTo(null);
    model = new DefaultTableModel();
    tableDriver.setModel(model);

    model.addColumn("ID Driver");
    model.addColumn("Nama Driver");
    model.addColumn("No HP");
    model.addColumn("Kendaraan");
    model.addColumn("Plat Nomor");
    model.addColumn("Tarif");

    tampilkan_data();
    btnEdit.setEnabled(false);
    btnHapus.setEnabled(false);
    btnBatal.setEnabled(false);
}

```

## - Tombol Tambah

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    String namaDriver = tfNama.getText();
    String nohp = tfNoHP.getText();
    String kendaraan = tfKendaraan.getText();
    String platNomor = tfPlatNomor.getText();
    String tarif = tfTarif.getText();

    String sql = "INSERT INTO driver (nama_driver, no_hp, kendaraan, plat_nomor, tarif) VALUES (?, ?, ?, ?, ?)";
    try (Connection c = Koneksi.getKoneksi(); PreparedStatement p = c.prepareStatement(sql)) {
        p.setString(1, namaDriver);
        p.setString(2, nohp);
        p.setString(3, kendaraan);
        p.setString(4, platNomor);
        p.setString(5, tarif);
        p.executeUpdate();
        JOptionPane.showMessageDialog(null, "Data Tersimpan");
        tampilkan_data();
    } catch (HeadlessException | SQLException e) {
        System.out.println("Terjadi Kesalahan: " + e.getMessage());
    } finally {
        kosongkan_form();
    }
}
```

## - Tombol Edit

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String nama_driver = tfNama.getText();
        String no_hp = tfNoHP.getText();
        String kendaraan = tfKendaraan.getText();
        String plat_nomor = tfPlatNomor.getText();
        String tarifString = tfTarif.getText();

        BigDecimal tarif = null;
        try {
            tarif = new BigDecimal(tarifString);
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Tarif tidak valid. Harap masukkan angka yang benar.", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        int selectedRow = tableDriver.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(null, "Silakan pilih data yang akan diubah!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        int id_driver = Integer.parseInt(tableDriver.getValueAt(selectedRow, 0).toString());
        String sql = "UPDATE DRIVER SET nama_driver = ?, no_hp = ?, kendaraan = ?, plat_nomor = ?, tarif = ? WHERE id_driver = ?";

        try (PreparedStatement pstmt = Koneksi.getKoneksi().prepareStatement(sql)) {
            pstmt.setString(1, nama_driver);
            pstmt.setString(2, no_hp);
            pstmt.setString(3, kendaraan);
            pstmt.setString(4, plat_nomor);
            pstmt.setBigDecimal(5, tarif);

            pstmt.setInt(6, id_driver);

            pstmt.executeUpdate();

            JOptionPane.showMessageDialog(null, "Data berhasil diubah!");
            tampilkan_data();
            kosongkan_form();
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
            System.err.println("Error: " + e.getMessage());
        }
    }
}
```

## - Tombo Hapus

```
private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {  
  
    try {  
        int selectedRow = tableDriver.getSelectedRow();  
        if (selectedRow == -1) {  
            JOptionPane.showMessageDialog(null, "Silakan pilih data yang akan dihapus!", "Error", JOptionPane.ERROR_MESSAGE);  
            return;  
        }  
  
        int id_driver = Integer.parseInt(tableDriver.getValueAt(selectedRow, 0).toString());  
  
        String sql = "DELETE FROM DRIVER WHERE id_driver = ?";  
  
        java.sql.Connection conn = Koneksi.getKoneksi();  
        try (PreparedStatement pstmt = conn.prepareStatement(sql)) {  
            pstmt.setInt(1, id_driver);  
            pstmt.executeUpdate();  
            JOptionPane.showMessageDialog(null, "Data berhasil dihapus!");  
            tampilkan_data();  
            kosongkan_form();  
        }  
    } catch (SQLException e) {  
        JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);  
        System.err.println("Error: " + e.getMessage());  
    }  
  
}
```

## - Tombol Batal

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {  
  
    kosongkan_form();  
    tampilkan_data();  
    btnTambah.setEnabled(true);  
    btnEdit.setEnabled(false);  
    btnHapus.setEnabled(false);  
    btnBatal.setEnabled(false);  
  
}
```

## A. Form Orderan

- Kelas Utama dan Method Form Orderan

```
public class FormOrderan extends javax.swing.JFrame {
    private DefaultTableModel model;
    public FormOrderan() {
        initComponents();
        this.setLocationRelativeTo(null);

        model = new DefaultTableModel();
        model.addColumn("ID Order");
        model.addColumn("ID Pengguna");
        model.addColumn("ID Driver");
        model.addColumn("Lokasi Awal");
        model.addColumn("Lokasi Tujuan");
        model.addColumn("Status");
        model.addColumn("Waktu");
        model.addColumn("Tanggal");
        model.addColumn("Tarif");

        tableorder.setModel(model);
        tampilkan_data();
        btnEdit.setEnabled(false);
        btnHapus.setEnabled(false);
        btnBatal.setEnabled(false);
    }

    private void kosongkan_form() {
        tfLokasiAwal.setText(null);
        tfLokasiTujuan.setText(null);
        cbxStatus.setSelectedIndex(0);
        tfIDPengguna.setText(null);
        tfIDDriver.setText(null);
        tfNamaPengguna.setText(null);
        tfNoHP.setText(null);
        tfKendaraan.setText(null);
        tfTarif.setText(null);
    }
}
```

```

private void tampilkan_data() {
    model.setRowCount(0);
    try {
        String sql = "SELECT o.id_order, o.lokasi_awal, o.lokasi_tujuan, o.status, "
            + "o.waktu_pesan AS waktu, "
            + "o.tanggal_pesan AS tanggal, "
            + "o.tarif, "
            + "o.id_pengguna, o.id_driver "
            + "FROM orderan o";

        java.sql.Connection conn = Koneksi.getKoneksi();
        java.sql.Statement stm = conn.createStatement();
        java.sql.ResultSet res = stm.executeQuery(sql);

        while (res.next()) {
            BigDecimal tarif = res.getBigDecimal("tarif");
            String tarifFormatted = tarif != null ? String.format("%.2f", tarif) : "0.00";
            model.addRow(new Object[]{
                res.getInt("id_order"),
                res.getInt("id_pengguna"),
                res.getInt("id_driver"),
                res.getString("lokasi_awal"),
                res.getString("lokasi_tujuan"),
                res.getString("status"),
                res.getTime("waktu"),
                res.getDate("tanggal"),
                tarifFormatted
            });
        }

        tableorder.setModel(model);
        System.out.println("Data berhasil ditampilkan.");
    } catch (SQLException e) {
        System.err.println("Error: " + e.getMessage());
    }
}

public void setDataPengguna(String id, String nama, String noHp) {
    tfIDPengguna.setText(id);
    tfNamaPengguna.setText(nama);
    tfNoHP.setText(noHp);
}

public void setDataDriver(String id, String kendaraan, BigDecimal tarif) {
    tfIDDriver.setText(id);
    tfKendaraan.setText(kendaraan);
    tfTarif.setText(tarif != null ? tarif.toString() : "0.00");
}

```

- Tombol Batal

```

private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {
    kosongkan_form();
    tampilkan_data();
    btnTambah.setEnabled(true);
    btnEdit.setEnabled(false);
    btnHapus.setEnabled(false);
    btnBatal.setEnabled(false);
}

```

## - Tabel

```
private void tableorderMouseClicked(java.awt.event.MouseEvent evt) {
    btnTambah.setEnabled(false);
    btnEdit.setEnabled(true);
    btnHapus.setEnabled(true);
    btnBatal.setEnabled(true);

    int i = tableorder.getSelectedRow();
    if (i == -1) {
        return;
    }

    DefaultTableModel model = (DefaultTableModel) tableorder.getModel();
    String id_pengguna = model.getValueAt(i, 1) != null ? model.getValueAt(i, 1).toString() : "";
    String id_driver = model.getValueAt(i, 2) != null ? model.getValueAt(i, 2).toString() : "";
    String lokasi_awal = model.getValueAt(i, 3) != null ? model.getValueAt(i, 3).toString() : "";
    String lokasi_tujuan = model.getValueAt(i, 4) != null ? model.getValueAt(i, 4).toString() : "";
    String status = model.getValueAt(i, 5) != null ? model.getValueAt(i, 5).toString() : "";

    tfIDPengguna.setText(id_pengguna);
    tfIDDriver.setText(id_driver);
    tfLokasiAwal.setText(lokasi_awal);
    tfLokasiTujuan.setText(lokasi_tujuan);
    cbxStatus.setSelectedItem(status);
}
```

## - Tombol Hapus

```
private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int selectedRow = tableorder.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(null, "Silakan pilih data yang akan dihapus!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        int id_order = Integer.parseInt(tableorder.getValueAt(selectedRow, 0).toString());

        String sql = "DELETE FROM orderan WHERE id_order = ?";

        java.sql.Connection conn = Koneksi.getKoneksi();
        try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setInt(1, id_order);
            pstmt.executeUpdate();
            JOptionPane.showMessageDialog(null, "Data berhasil dihapus!");
            tampilkan_data();
            kosongkan_form();
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(this, "Terjadi kesalahan: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        System.err.println("Error: " + e.getMessage());
    }
}
```

## - Tombol Edit

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        int selectedRow = tableorder.getSelectedRow();
        if (selectedRow == -1) {
            JOptionPane.showMessageDialog(null, "Silakan pilih data yang akan diubah!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
        int id_order = Integer.parseInt(tableorder.getValueAt(selectedRow, 0).toString());

        String lokasi_awal = tfLokasiAwal.getText().trim();
        String lokasi_tujuan = tfLokasiTujuan.getText().trim();
        String status = cbxStatus.getSelectedItem().toString();
        String tarifString = tfTarif.getText().trim();

        BigDecimal tarif;
        try {
            tarif = new BigDecimal(tarifString);
            if (tarif.compareTo(BigDecimal.ZERO) < 0) {
                JOptionPane.showMessageDialog(null, "Tarif tidak boleh negatif!", "Error", JOptionPane.ERROR_MESSAGE);
                return;
            }
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Tarif harus berupa angka valid!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        String sql = "UPDATE orderan SET lokasi_awal = ?, lokasi_tujuan = ?, status = ?, tarif = ? WHERE id_order = ?";
        try (Connection conn = Koneksi.getKoneksi(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, lokasi_awal);

            pstmt.setString(2, lokasi_tujuan);
            pstmt.setString(3, status);
            pstmt.setBigDecimal(4, tarif);
            pstmt.setInt(5, id_order);

            int rowsUpdated = pstmt.executeUpdate();
            if (rowsUpdated > 0) {
                JOptionPane.showMessageDialog(null, "Data berhasil diubah!");
            } else {
                JOptionPane.showMessageDialog(null, "Data tidak ditemukan atau tidak ada perubahan!", "Informasi",
                    JOptionPane.INFORMATION_MESSAGE);
            }
        }

        tampilkan_data();
        kosongkan_form();
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(this, "Terjadi kesalahan saat mengubah data: " + e.getMessage(), "Error",
            JOptionPane.ERROR_MESSAGE);
        System.err.println("Error: " + e.getMessage());
    }
}
```



## - Tombol Tambah

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {
    String id_pengguna = tfIDPengguna.getText();
    String id_driver = tfIDDriver.getText();
    String lokasi_awal = tfLokasiAwal.getText();
    String lokasi_tujuan = tfLokasiTujuan.getText();
    String status = cbxStatus.getSelectedItem().toString();
    String tarifString = tfTarif.getText();

    try {
        if (id_pengguna.isEmpty() || id_driver.isEmpty() || lokasi_awal.isEmpty() || lokasi_tujuan.isEmpty() || status.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Harap isi semua kolom!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        BigDecimal tarif = null;
        try {
            tarif = new BigDecimal(tarifString);
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(null, "Tarif tidak valid! Harap masukkan angka yang benar.", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        Connection c = Koneksi.getKoneksi();
        String sql = "INSERT INTO orderan (id_pengguna, id_driver, lokasi_awal, lokasi_tujuan, status, waktu_pesan, tanggal_pesan, tarif) "
            + "VALUES (?, ?, ?, ?, ?, CURRENT_TIME(), CURRENT_DATE(), ?)";

        try (PreparedStatement p = c.prepareStatement(sql)) {
            p.setInt(1, Integer.parseInt(id_pengguna));
            p.setInt(2, Integer.parseInt(id_driver));
            p.setString(3, lokasi_awal);
            p.setString(4, lokasi_tujuan);
            p.setString(5, status);
            p.setBigDecimal(6, tarif);

            p.executeUpdate();
        } catch (SQLException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(null, "Terjadi kesalahan saat menyimpan data ke database.", "Error", JOptionPane.ERROR_MESSAGE);
        }

        JOptionPane.showMessageDialog(null, "Data Tersimpan");
        tampilkan_data();
        kosongkan_form();
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(null, "ID Pengguna dan ID Driver harus berupa angka!", "Error", JOptionPane.ERROR_MESSAGE);
        System.err.println("Error: " + e.getMessage());
    } catch (HeadlessException e) {
        JOptionPane.showMessageDialog(null, "Terjadi Kesalahan: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
        System.err.println("Error: " + e.getMessage());
    }
}
}
```

## - Tombol Cari

```
private void caridatapenggunaActionPerformed(java.awt.event.ActionEvent evt) {
    ListPengguna listPengguna = new ListPengguna(this);
    listPengguna.setVisible(true);
}

private void caridatadriverActionPerformed(java.awt.event.ActionEvent evt) {
    ListDriver listDriver = new ListDriver(this);
    listDriver.setVisible(true);
}
}
```

## B. Form Pembayaran

- Kelas Utama dan Method Form Pembayaran

```
public class FormPembayaran extends javax.swing.JFrame {
    private DefaultTableModel model;
    public FormPembayaran() {
        initComponents();
        this.setLocationRelativeTo(null);
        model = new DefaultTableModel();
        tablepembayaran.setModel(model);

        model.addColumn("No Pembayaran");
        model.addColumn("ID Order");
        model.addColumn("Metode Bayar");
        model.addColumn("Total Bayar");
        model.addColumn("Tanggal Pembayaran");

        tampilkan_data();
        btnEdit.setEnabled(false);
        btnHapus.setEnabled(false);
        btnBatal.setEnabled(false);

        tablepembayaran.getSelectionModel().addListSelectionListener(event -> {
            if (!tablepembayaran.getSelectionModel().isSelectionEmpty()) {
                btnEdit.setEnabled(true);
                btnHapus.setEnabled(true);
                btnBatal.setEnabled(true);
            } else {
                btnEdit.setEnabled(false);
                btnHapus.setEnabled(false);
                btnBatal.setEnabled(false);
            }
        });
    }

    private void kosongkan_form(){
        tfIDOrder.setText(null);
        tfTanggal.setText(null);
        cbxmetode.setSelectedIndex(0);
        tfTarif.setText(null);
        tfLokasiAwal.setText(null);
        tfLokasiTujuan.setText(null);
    }

    private void tampilkan_data() {
        model.setRowCount(0);
        try {
            String sql = "SELECT no_pembayaran, id_order, metode_pembayaran, total_bayar, tanggal_pembayaran FROM pembayaran";
            Connection conn = Koneksi.getKoneksi();
            PreparedStatement pstmt = conn.prepareStatement(sql);
            java.sql.ResultSet res = pstmt.executeQuery();

            while (res.next()) {
                model.addRow(new Object[]{
                    res.getString("no_pembayaran"),
                    res.getString("id_order"),
                    res.getString("metode_pembayaran"),
                    res.getString("total_bayar"),
                    res.getString("tanggal_pembayaran")
                });
            }
        } catch (SQLException e) {
            JOptionPane.showMessageDialog(this, e.getMessage());
        }
    }
}
```

```

private void cetakStruk() {
    String noPembayaran = tfNoPembayaran.getText();
    String idOrder = tfIDOrder.getText();
    String metodeBayar = cbxmetode.getSelectedItem().toString();
    String totalBayar = tfTarif.getText();
    String tanggalBayar = tfTanggal.getText();

    String struk = "STRUK PEMBAYARAN\n" +
        "No. Pembayaran: " + noPembayaran + "\n" +
        "ID Order: " + idOrder + "\n" +
        "Metode Bayar: " + metodeBayar + "\n" +
        "Total Bayar: Rp " + totalBayar + "\n" +
        "Tanggal Bayar: " + tanggalBayar + "\n";

    JOptionPane.showMessageDialog(null, struk, "Struk Pembayaran", JOptionPane.INFORMATION_MESSAGE);

private void cariData() {
    model.setRowCount(0);
    String pilihan = cbxcariData.getSelectedItem().toString();
    String keyword = caridata.getText().trim();

    if (keyword.isEmpty()) {
        JOptionPane.showMessageDialog(null, "Silakan masukkan kata kunci untuk pencarian.", "Peringatan", JOptionPane.WARNING_MESSAGE);
        return;
    }

    String sql;
    if (pilihan.equals("no_pembayaran")) {
        sql = "SELECT * FROM pembayaran WHERE no_pembayaran = ?";
    } else {

        sql = "SELECT * FROM pembayaran WHERE metode_pembayaran LIKE ?";
    }

    try (Connection conn = Koneksi.getKoneksi(); PreparedStatement pstmt = conn.prepareStatement(sql)) {
        if (pilihan.equals("no_pembayaran")) {
            if (!keyword.matches("\\d+")) {
                JOptionPane.showMessageDialog(null, "no pembayaran harus berupa angka.", "Peringatan", JOptionPane.WARNING_MESSAGE);
                return;
            }
            pstmt.setInt(1, Integer.parseInt(keyword));
        } else {
            pstmt.setString(1, "%" + keyword + "%");
        }

        ResultSet res = pstmt.executeQuery();

        if (!res.isBeforeFirst()) {
            JOptionPane.showMessageDialog(null, "Data tidak ditemukan.", "Informasi", JOptionPane.INFORMATION_MESSAGE);
        } else {
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("no_pembayaran"),
                    res.getString("id_order"),
                    res.getString("metode_pembayaran"),
                    res.getString("total_bayar"),
                    res.getString("tanggal_pembayaran")
                });
            }
        }
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Error: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
    }
}

```

## - Tombol Tambah

```
private void btnTambahActionPerformed(java.awt.event.ActionEvent evt) {

    String idOrder = tfIDOrder.getText();
    String metodeBayar = cbxmetode.getSelectedItem().toString();
    String totalBayar = tfTarif.getText();

    try {
        if (idOrder.isEmpty() || metodeBayar.isEmpty() || totalBayar.isEmpty()) {
            JOptionPane.showMessageDialog(null, "Harap isi semua kolom!", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }

        Connection conn = Koneksi.getKoneksi();
        String sql = "INSERT INTO pembayaran (id_order, metode_pembayaran, total_bayar, tanggal_pembayaran) "
            + "VALUES (?, ?, ?, CURRENT_DATE())";

        try (PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setInt(1, Integer.parseInt(idOrder));
            pstmt.setString(2, metodeBayar);
            pstmt.setDouble(3, Double.parseDouble(totalBayar));
            pstmt.executeUpdate();
        }

        JOptionPane.showMessageDialog(null, "Data Pembayaran Berhasil Ditambahkan!");
        tampilkan_data();
        kosongkan_form();
    } catch (NumberFormatException e) {
        JOptionPane.showMessageDialog(null, "ID Order dan Total Bayar harus berupa angka!", "Error", JOptionPane.ERROR_MESSAGE);
        System.err.println("Error: " + e.getMessage());
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(null, "Gagal menyimpan data: " + e.getMessage(), "Error", JOptionPane.ERROR_MESSAGE);
    }
}
```

## - Tombol Edit

```
private void btnEditActionPerformed(java.awt.event.ActionEvent evt) {
    try {
        String no_pembayaran = tfNOPembayaran.getText();
        String metode_pembayaran = cbxmetode.getSelectedItem().toString();
        String total_bayar = tfTarif.getText();
        String tanggal_pembayaran = tfTanggal.getText();

        String sql = "UPDATE PEMBAYARAN SET no_pembayaran = '" + no_pembayaran + "', metode_pembayaran = '" + metode_pembayaran
            + "', total_bayar = '" + total_bayar + "', tanggal_pembayaran = '" + tanggal_pembayaran + "' WHERE no_pembayaran = '"
            + no_pembayaran + "'";

        java.sql.Connection conn = (Connection)Koneksi.getKoneksi();
        java.sql.PreparedStatement pstmt = conn.prepareStatement(sql);
        pstmt.execute();
        JOptionPane.showMessageDialog(null, "Data berhasil diubah!");
        tampilkan_data();
        kosongkan_form();
    } catch (SQLException e) {
        JOptionPane.showMessageDialog(this, e.getMessage());
    }
}
```

- Tombol Hapus

```
private void btnHapusActionPerformed(java.awt.event.ActionEvent evt) {  
    try{  
        String no_pembayaran = tfNOPembayaran.getText();  
  
        String sql = "DELETE from PEMBAYARAN WHERE no_pembayaran = '" + no_pembayaran + "'";  
  
        java.sql.Connection conn = (Connection)Koneksi.getKoneksi();  
        java.sql.PreparedStatement pstmt = conn.prepareStatement(sql);  
        pstmt.execute();  
        JOptionPane.showMessageDialog(null, "Data berhasil dihapus!");  
        tampilkan_data();  
        kosongkan_form();  
    } catch (SQLException e) {  
        JOptionPane.showMessageDialog(this, e.getMessage());  
    }  
}
```

- Tombol Batal

```
private void btnBatalActionPerformed(java.awt.event.ActionEvent evt) {  
    kosongkan_form();  
    tampilkan_data();  
    btnTambah.setEnabled(true);  
    btnEdit.setEnabled(false);  
    btnHapus.setEnabled(false);  
    btnBatal.setEnabled(false);  
}
```

- Tabel

```
private void tablepembayaranMouseClicked(java.awt.event.MouseEvent evt) {  
  
    int i = tablepembayaran.getSelectedRow();  
    if (i == -1) {  
        return;  
    }  
  
    DefaultTableModel model = (DefaultTableModel) tablepembayaran.getModel();  
    String no_pembayaran = (String) model.getValueAt(i, 0);  
    String id_order = (String) model.getValueAt(i, 1);  
    String metode_pembayaran = (String) model.getValueAt(i, 2);  
    String total_bayar = (String) model.getValueAt(i, 3);  
    String tanggal = (String) model.getValueAt(i, 4);  
  
    tfNOPembayaran.setText(no_pembayaran);  
    tfIDOrder.setText(id_order);  
    cbxmetode.setSelectedItem(metode_pembayaran);  
    tfTarif.setText(total_bayar);  
    tfTanggal.setText(tanggal);  
}
```

### C. Form List Pengguna

#### - Kelas utama dan Method List Pengguna

```
public class ListPengguna extends javax.swing.JFrame {
    private FormOrderan formOrderan;
    private DefaultTableModel model;

    private void tampilkan_data() {
        model.setRowCount(0);
        try (Connection conn = Koneksi.getKoneksi(); Statement stm = conn.createStatement()) {
            String sql = "SELECT * FROM pengguna";
            ResultSet res = stm.executeQuery(sql);
            while (res.next()) {
                model.addRow(new Object[] {
                    res.getInt("id_pengguna"),
                    res.getString("nama_pengguna"),
                    res.getString("email"),
                    res.getString("no_hp"),
                    res.getString("alamat")
                });
            }
            tablepengguna.setModel(model);
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    private void cariData() {
        DefaultTableModel tabel = new DefaultTableModel();
        tabel.addColumn("id_pengguna");
        tabel.addColumn("nama_pengguna");
        tabel.addColumn("email");
        tabel.addColumn("no_hp");
        tabel.addColumn("alamat");
    }
}
```

```

if (caridata.getText().trim().isEmpty()) {
    JOptionPane.showMessageDialog(null, "Masukkan data yang ingin dicari!");
    return;
}

try {
    Connection c = Koneksi.getKoneksi();
    String sql = "SELECT * FROM pengguna WHERE id_pengguna LIKE ? OR nama_pengguna LIKE ?";
    PreparedStatement ps = c.prepareStatement(sql);
    ps.setString(1, "%" + caridata.getText() + "%");
    ps.setString(2, "%" + caridata.getText() + "%");

    ResultSet rs = ps.executeQuery();
    boolean dataDitemukan = false;

    while (rs.next()) {
        dataDitemukan = true;
        tabel.addRow(new Object[]{
            rs.getString("id_pengguna"),
            rs.getString("nama_pengguna"),
            rs.getString("email"),
            rs.getString("no_hp"),
            rs.getString("alamat")
        });
    }

    if (!dataDitemukan) {
        JOptionPane.showMessageDialog(null, "Data tidak ditemukan!");
    }

    tablepengguna.setModel(tabel);

} catch (SQLException e) {
    System.out.println("Cari Data Error: " + e.getMessage());
}

}

public ListPengguna(FormOrderan formOrderan) {
    initComponents();
    this.setLocationRelativeTo(null);
    model = new DefaultTableModel();
    tablepengguna.setModel(model);

    model.addColumn("ID_Pengguna");
    model.addColumn("Nama_Pengguna");
    model.addColumn("Email");
    model.addColumn("No HP");
    model.addColumn("Alamat");

    this.formOrderan = formOrderan;
    this.setLocationRelativeTo(null);
    tampilkan_data();
    btnPilih.setEnabled(true);
    btnBatal.setEnabled(true);
}

```

- Tombol Pilih

```
private void btnPilihActionPerformed(java.awt.event.ActionEvent evt) {

    int i = tablepengguna.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(this, "Silakan pilih driver terlebih dahulu.");
        return;
    }

    String id_pengguna = tablepengguna.getValueAt(i, 0).toString();
    String nama_pengguna = tablepengguna.getValueAt(i, 1).toString();
    String no_hp = tablepengguna.getValueAt(i, 3).toString();

    formOrderan.setDataPengguna(id_pengguna, nama_pengguna, no_hp);
    dispose();
}
```

## D. Form List Driver

- Kelas utama dan Method List Driver

```
public class ListDriver extends javax.swing.JFrame {
    private FormOrderan formOrderan;
    private DefaultTableModel model;

    private void tampilkan_data() {
        model.setRowCount(0);
        try (Connection conn = Koneksi.getKoneksi(); Statement stm = conn.createStatement()) {
            String sql = "SELECT * FROM driver";
            ResultSet res = stm.executeQuery(sql);
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("id_driver"),
                    res.getString("nama_driver"),
                    res.getString("no_hp"),
                    res.getString("kendaraan"),
                    res.getString("plat_nomor"),
                    res.getBigDecimal("tarif")
                });
            }
            tabledriver.setModel(model);
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    private void cariData() {
        DefaultTableModel tabel = new DefaultTableModel();
        tabel.addColumn("id_driver");
        tabel.addColumn("nama_driver");
        tabel.addColumn("no_hp");
        tabel.addColumn("kendaraan");
    }
}
```



```

tabel.addColumn("tarif");

if (caridata.getText().trim().isEmpty()) {
    JOptionPane.showMessageDialog(null, "Masukkan data yang ingin dicari!");
    return;
}

try {
    Connection c = Koneksi.getKoneksi();
    String sql = "SELECT * FROM driver WHERE id_driver LIKE ? OR nama_driver LIKE ?";
    PreparedStatement ps = c.prepareStatement(sql);
    ps.setString(1, "%" + caridata.getText() + "%");
    ps.setString(2, "%" + caridata.getText() + "%");

    ResultSet rs = ps.executeQuery();
    boolean dataDitemukan = false;

    while (rs.next()) {
        dataDitemukan = true;
        tabel.addRow(new Object[]{
            rs.getString("id_driver"),
            rs.getString("nama_driver"),
            rs.getString("no_hp"),
            rs.getString("kendaraan"),
            rs.getString("plat_nomor"),
            rs.getBigDecimal("tarif")
        });
    }

    if (!dataDitemukan) {

        JOptionPane.showMessageDialog(null, "Data tidak ditemukan!");
    }

    tabledriver.setModel(tabel);

} catch (SQLException e) {
    System.out.println("Cari Data Error: " + e.getMessage());
}

}

public ListDriver(FormOrderan formOrderan) {
    initComponents();
    this.setLocationRelativeTo(null);
    model = new DefaultTableModel();
    tabledriver.setModel(model);

    model.addColumn("ID_Driver");
    model.addColumn("Nama_Driver");
    model.addColumn("No HP");
    model.addColumn("Kendaraan");
    model.addColumn("Plat Nomor");
    model.addColumn("Tarif");

    this.formOrderan = formOrderan;
    this.setLocationRelativeTo(null);
    tampilkan_data();
    btnPilih.setEnabled(true);
    btnBatal.setEnabled(true);
}

```

## - Tombol Pilih

```
private void btnPilihActionPerformed(java.awt.event.ActionEvent evt) {

    int i = tabledriver.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(this, "Silakan pilih driver terlebih dahulu.");
        return;
    }
    String id_driver = tabledriver.getValueAt(i, 0).toString();
    String kendaraan = tabledriver.getValueAt(i, 3).toString();

    Object tarifObject = tabledriver.getValueAt(i, 5);
    BigDecimal tarif = null;

    if (tarifObject != null && tarifObject instanceof BigDecimal) {
        tarif = (BigDecimal) tarifObject;
    } else {
        JOptionPane.showMessageDialog(this, "Nilai tarif tidak valid.", "Error", JOptionPane.ERROR_MESSAGE);
        return;
    }

    formOrderan.setDataDriver(id_driver, kendaraan, tarif);
    dispose();
}
```

## E. Form List Orderan

### - Kelas utama dan Method List Orderan

```
public class ListOrderan extends javax.swing.JFrame {
    private DefaultTableModel model;

    private void tampilkan_data(){
        model.setRowCount(0);
        try (Connection conn = Koneksi.getKoneksi(); Statement stm = conn.createStatement()) {
            String sql = "SELECT * FROM orderan";
            ResultSet res = stm.executeQuery(sql);
            while (res.next()) {
                model.addRow(new Object[]{
                    res.getInt("id_order"),
                    res.getInt("id_pengguna"),
                    res.getInt("id_driver"),
                    res.getString("lokasi_awal"),
                    res.getString("lokasi_tujuan"),
                    res.getString("status"),
                    res.getTime("waktu_pesan"),
                    res.getDate("tanggal_pesan"),
                    res.getBigDecimal("tarif")
                });
            }
            tableorderan.setModel(model);
        } catch (SQLException e) {
            System.out.println("Error: " + e.getMessage());
        }
    }

    private void cariData() {
        DefaultTableModel tabel = new DefaultTableModel();
        tabel.addColumn("id_order");
        tabel.addColumn("id_pengguna");
    }
}
```

```

tabel.addColumn("id_driver");
tabel.addColumn("lokasi_awal");
tabel.addColumn("lokasi_tujuan");
tabel.addColumn("status");
tabel.addColumn("waktu_pesan");
tabel.addColumn("tanggal_pesan");
tabel.addColumn("tarif");

if (caridata.getText().trim().isEmpty()) {
    JOptionPane.showMessageDialog(null, "Masukkan data yang ingin dicari!");
    return;
}

try {
    Connection c = Koneksi.getKoneksi();
    String sql = "SELECT * FROM orderan WHERE id_order LIKE ? ";
    PreparedStatement ps = c.prepareStatement(sql);
    ps.setString(1, "%" + caridata.getText() + "%");

    ResultSet rs = ps.executeQuery();
    boolean dataDitemukan = false;

    while (rs.next()) {
        dataDitemukan = true;
        tabel.addRow(new Object[]{
            rs.getInt("id_order"),
            rs.getInt("id_pengguna"),
            rs.getInt("id_driver"),
            rs.getString("lokasi_awal"),
            rs.getString("lokasi_tujuan"),
            rs.getString("status"),
            rs.getInt("id_driver"),
            rs.getString("lokasi_awal"),
            rs.getString("lokasi_tujuan"),
            rs.getString("status"),
            rs.getTime("waktu_pesan"),
            rs.getDate("tanggal_pesan"),
            rs.getBigDecimal("tarif")
        });
    }

    if (!dataDitemukan) {
        JOptionPane.showMessageDialog(null, "Data tidak ditemukan!");
    }

    tableorderan.setModel(tabel);
} catch (SQLException e) {
    System.out.println("Cari Data Error: " + e.getMessage());
}
}

```

```

public ListOrderan() {
    initComponents();
    this.setLocationRelativeTo(null);
    model = new DefaultTableModel();
    tableorderan.setModel(model);

    model.addColumn("ID Orderan");
    model.addColumn("ID Pengguna");
    model.addColumn("ID Driver");
    model.addColumn("Lokasi Awal");
    model.addColumn("Lokasi Tujuan");
    model.addColumn("Status");
    model.addColumn("Waktu Pesan");
    model.addColumn("Tanggal Pesan");
    model.addColumn("Tarif");

    tampilkan_data();
    btnPilih.setEnabled(true);
    btnBatal.setEnabled(true);
}

```

## - Tombol Pilih

```

private void btnPilihActionPerformed(java.awt.event.ActionEvent evt) {

    int i = tableorderan.getSelectedRow();
    if (i == -1) {
        JOptionPane.showMessageDialog(this, "Silakan pilih orderan terlebih dahulu.");
        return;
    }
    String id_orderan = tableorderan.getValueAt(i, 0).toString();
    String lokasi_awal = tableorderan.getValueAt(i, 3).toString();
    String lokasi_tujuan = tableorderan.getValueAt(i, 4).toString();
    Object tarifObject = tableorderan.getValueAt(i, 8);
    BigDecimal tarif = null;
    System.out.println("Tarif yang diambil dari tabel: " + tarifObject);

    if (tarifObject != null) {
        try {
            tarif = new BigDecimal(tarifObject.toString().trim());
            System.out.println("Tarif berhasil dikonversi: " + tarif);
        } catch (NumberFormatException e) {
            JOptionPane.showMessageDialog(this, "Tarif tidak valid.", "Error", JOptionPane.ERROR_MESSAGE);
            return;
        }
    } else {
        JOptionPane.showMessageDialog(this, "Tarif tidak valid.", "Error", JOptionPane.ERROR_MESSAGE);
        return;
    }

    FormPembayaran formPembayaran = new FormPembayaran();
    formPembayaran.tfIDOrder.setText(id_orderan);
    formPembayaran.tfLokasiAwal.setText(lokasi_awal);
    formPembayaran.tfLokasiTujuan.setText(lokasi_tujuan);
    formPembayaran.tfTarif.setText(tarif != null ? tarif.toString() : "0.00");
}

```

## 2.2 Tabel Database

Database yang digunakan adalah MariaDB di Xampp dengan port 3308.

Berikut ini adalah daftar table yang ada pada database:

```
MariaDB [ojek_pangkalan]> show tables;
+-----+
| Tables_in_ojek_pangkalan |
+-----+
| driver                     |
| orderan                   |
| pembayaran                |
| pengguna                  |
+-----+
4 rows in set (0.001 sec)
```

### A. Tabel Pengguna

```
MariaDB [ojek_pangkalan]> select * from pengguna;
+-----+-----+-----+-----+-----+
| id_pengguna | nama_pengguna | email          | no_hp      | alamat      |
+-----+-----+-----+-----+-----+
| 1           | Indah         | indah@gmail   | 082135652640 | Pangeran    |
| 3           | Yulia        | yulia@gmail   | 0878765432  | Cendana 2B  |
| 7           | Indri        | indri@gmail   | 081234567823 | Kayu Tangi 2 |
| 8           | Rani         | rani@gmail    | 082346789345 | Belitung    |
| 9           | Lidya        | lidya@gmail   | 081234567853 | Sei Andai   |
| 10          | Raudah       | raudah@gmail  | 082156743578 | Pasar Lama  |
| 11          | Ers          | ersa@gmail    | 081234567892 | kuin cerucuk |
+-----+-----+-----+-----+-----+
7 rows in set (0.001 sec)
```

### B. Tabel Driver

```
MariaDB [ojek_pangkalan]> select * from driver;
+-----+-----+-----+-----+-----+-----+
| id_driver | nama_driver | no_hp      | kendaraan | plat_nomor | tarif      |
+-----+-----+-----+-----+-----+-----+
| 1         | Sari        | 081234567890 | Fazzio    | DA 1234 AU | 25000.00  |
| 4         | Firman      | 082135652640 | Scoopy    | DA 6261 UA | 10000.00  |
| 6         | Reno        | 081234567891 | Vario     | DA 2345 BBS | 15000.00  |
| 7         | Hilman      | 08123454321234 | N-Max     | KT 5432 IA | 20000.00  |
| 9         | Adit        | 081234567891 | Mio       | KT 2345 UY | 10000.00  |
+-----+-----+-----+-----+-----+-----+
5 rows in set (0.003 sec)
```

### C. Tabel Orderan

```
MariaDB [ojek_pangkalan]> select * from orderan;
+-----+-----+-----+-----+-----+-----+-----+-----+
| id_order | id_pengguna | id_driver | lokasi_awal | lokasi_tujuan | status | waktu_pesanan | tanggal_pesanan | tarif      |
+-----+-----+-----+-----+-----+-----+-----+-----+
| 13       | 1           | 1         | Kayu Tangi  | Belitung      | Selesai | 02:29:28      | 2024-12-18      | 20000.00  |
| 14       | 9           | 6         | Cendana     | pangeran      | Proses  | 02:47:47      | 2024-12-18      | 25000.00  |
| 15       | 10          | 1         | Kidaung Permai | Pasar lama    | Selesai | 03:29:56      | 2024-12-18      | 25000.00  |
+-----+-----+-----+-----+-----+-----+-----+-----+
3 rows in set (0.000 sec)
```

#### D. Tabel Pembayaran

```
MariaDB [ojek_pangkalan]> select * from pembayaran;
```

no_pembayaran	id_order	metode_pembayaran	total_bayar	tanggal_pembayaran
3	14	Transfer	15000.00	2024-12-18
4	13	Tunai	25000.00	2024-12-11
5	15	Tunai	25000.00	2024-12-18
6	14	Tunai	15000.00	2024-12-18
8	13	Tunai	25000.00	2024-12-18
9	15	Tunai	10000.00	2024-12-18
10	13	E-Wallet	25000.00	2024-12-18
12	13	Tunai	20000.00	2024-12-18

8 rows in set (0.003 sec)

### 2.3 Hasil Akhir

#### A. Alur Program

Untuk Alur program dari proyek *ojek\_pangkalan* ini dapat dijelaskan sebagai berikut:

##### 1. Form Pengguna:

Pengguna akan mengisi data seperti nama, email, nomor HP, dan alamat. Di dalam form ini terdapat fungsi CRUD (Create, Read, Update, Delete), yang memungkinkan untuk menambahkan data pengguna baru, mengedit data pengguna yang sudah ada, menghapus data pengguna, dan membatalkan perubahan. Ada juga kolom pencarian, yang memungkinkan pengguna mencari data pengguna berdasarkan nama atau informasi lainnya.

##### 2. Form Driver:

Pada form driver, pengguna akan mengisi informasi seperti nama, nomor HP, kendaraan, plat nomor, dan tarif. Seperti pada form pengguna, form driver juga dilengkapi dengan fungsi CRUD, yang memungkinkan untuk menambah, mengedit, menghapus, atau membatalkan data driver. Pengguna juga bisa mencari driver tertentu menggunakan kolom pencarian.

### 3. **Form Orderan:**

Pada form ini, pengguna akan mengisi lokasi awal, lokasi tujuan, dan status orderan. Di dalam form ini terdapat tombol cari, yang mengarahkan pengguna ke form *List Pengguna*. Pengguna dapat memilih salah satu pengguna yang tersedia, dan ketika tombol "Pilih" ditekan, data pengguna akan terisi otomatis di form orderan. Begitu juga dengan tombol cari di driver, yang mengarahkan ke form *List Driver* untuk memilih driver. Form orderan juga dilengkapi dengan fungsi CRUD untuk menambah, mengedit, menghapus, atau membatalkan data orderan.

### 4. **Form Pembayaran:**

Di form ini, kolom untuk nomor pembayaran dan tanggal tidak perlu diisi manual, karena akan terisi otomatis berdasarkan data yang dipilih. Terdapat tombol cari yang akan membuka form *List Orderan*, di mana pengguna bisa memilih orderan yang ingin dibayar. Setelah memilih data, informasi pembayaran akan terisi otomatis di form pembayaran. Form ini juga dilengkapi dengan fungsi CRUD untuk menambah, mengedit, menghapus, atau membatalkan data pembayaran, serta kolom pencarian untuk memudahkan mencari data pembayaran.

## B. Tampilan Aplikasi

### 1) Form Pengguna

The screenshot shows a web application window titled "Kelola Pengguna". It features a form with four input fields: "Nama", "Email", "No HP", and "Alamat". Below the form is a table with five columns: "ID\_Pengguna", "Nama\_Peng...", "Email", "No HP", and "Alamat". The table contains four rows of data. At the bottom, there is a "Cari Data" section with a dropdown menu set to "id\_pengguna" and an empty search box. Below this are four buttons: "Tambah" (green), "Edit", "Hapus", and "Batal".

ID_Pengguna	Nama_Peng...	Email	No HP	Alamat
1	Indah	indah@gmail	0821356526...	Pangeran
3	Yulia	yulia@gmail	0878765432	Cendana 2B
7	Indri	indri@gmail	0812345678...	Kayu Tangi 2
8	Rani	rani@gmail	0823467893	Belitun

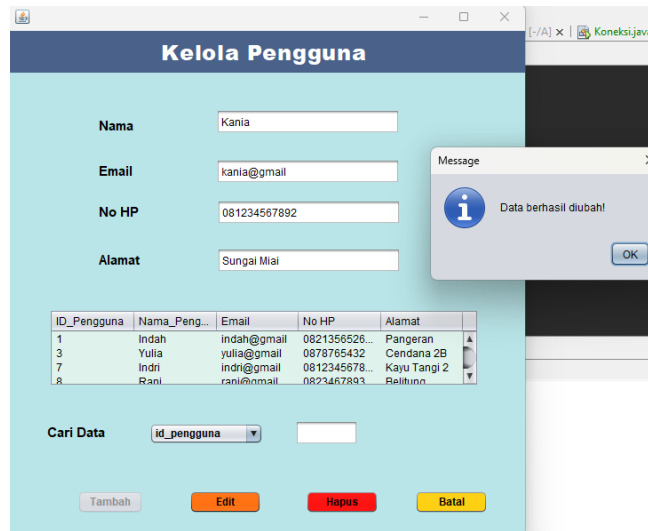
- Tampilan ketika data pengguna ditambahkan

This screenshot shows the same "Kelola Pengguna" application window, but with new data entered in the form: "Nama" is "Kania", "Email" is "kania@gmail", "No HP" is "081234567892", and "Alamat" is "Handil Bakti". The table below the form now has five rows of data. A "Message" dialog box is overlaid on the bottom right, displaying an information icon, the text "Data Tersimpan", and an "OK" button.

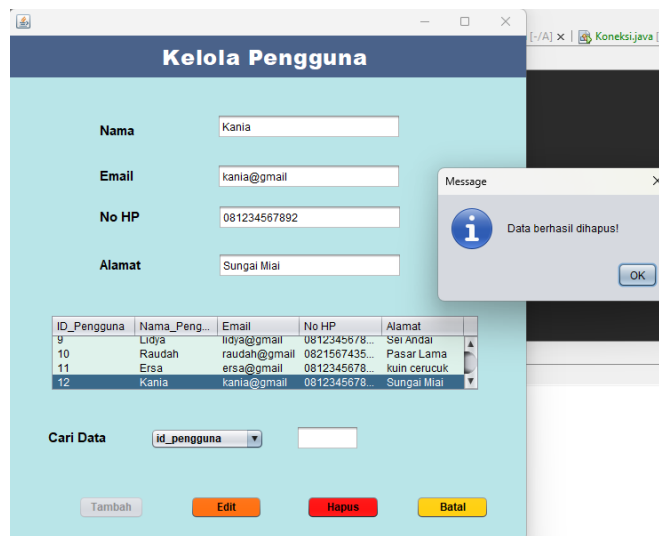
ID_Pengguna	Nama_Peng...	Email	No HP	Alamat
1	Indah	indah@gmail	0821356526...	Pangeran
3	Yulia	yulia@gmail	0878765432	Cendana 2B
7	Indri	indri@gmail	0812345678...	Kayu Tangi 2
8	Rani	rani@gmail	0823467893	Belitun



- Tampilan ketika data pengguna diubah



- Tampilan ketika data dihapus



## 2) Form Driver

**Kelola Driver**

Nama Driver

No HP

Kendaraan

Plat Nomor

Tarif

ID Driver	Nama Driver	No HP	Kendaraan	Plat Nomor	Tarif
1	Sari	08123456...	Fazzio	DA 1234 AU	25000.00
4	Firman	08213565...	Scoopy	DA 6261 UA	10000.00
6	Reno	08123456...	Vario	DA 2345 B...	15000.00
7	Hilman	08123454...	N-Max	KT 5432 IA	20000.00

Cari Data

- Tampilan ketika data driver ditambahkan

**Kelola Driver**

Nama Driver

No HP

Kendaraan

Plat Nomor

Tarif

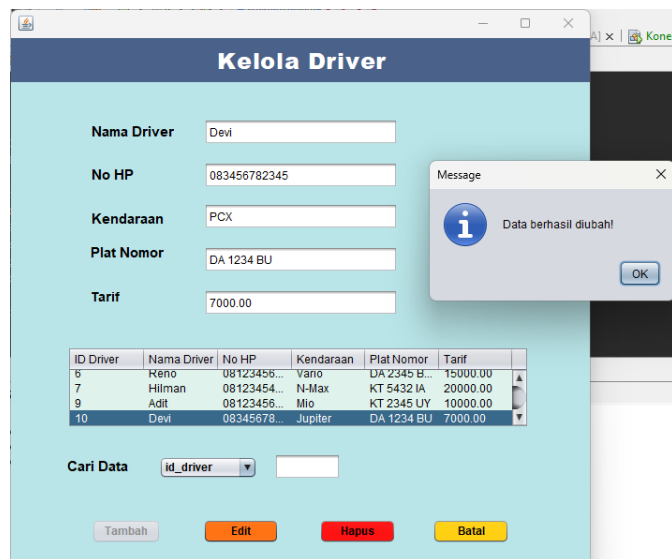
ID Driver	Nama Driver	No HP	Kendaraan	Plat Nomor	Tarif
4	Firman	08213565...	Scoopy	DA 6261 UA	10000.00
6	Reno	08123456...	Vario	DA 2345 B...	15000.00
7	Hilman	08123454...	N-Max	KT 5432 IA	20000.00
9	Adit	08123456...	Mio	KT 2345 UY	10000.00

Cari Data

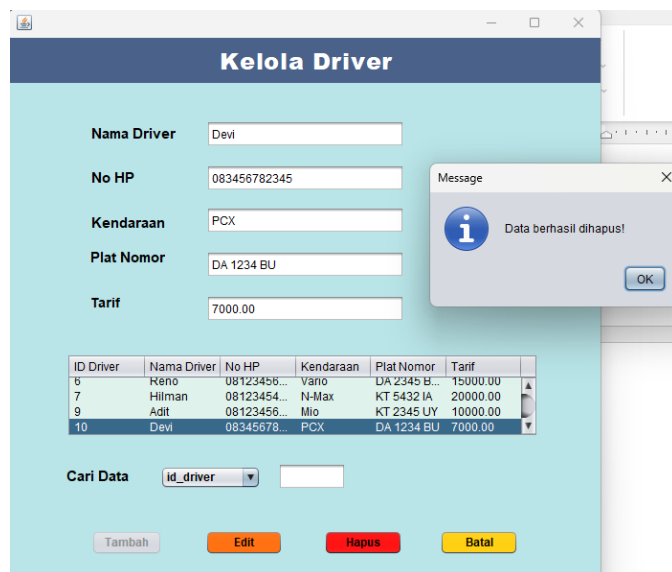
Message

Data Tersimpan

- Tampilan ketika data driver diubah



- Tampilan ketika data driver dihapus



### 3) Form Orderan

The 'Form Order' window contains the following elements:

- Lokasi Awal:** Text input field.
- Lokasi Tujuan:** Text input field.
- Status:** Dropdown menu with 'Proses' selected.
- User Information:**
  - ID Pengguna:** Text input field.
  - Nama Pengguna:** Text input field.
  - No HP:** Text input field.
  - Cari:** Button next to ID and Name fields.
- Driver Information:**
  - ID Driver:** Text input field.
  - Kendaraan:** Text input field.
  - Tarif:** Text input field.
  - Cari:** Button next to ID and Vehicle fields.
- Table of Orders:**

ID Order	ID Pengg...	ID Driver	Lokasi A...	Lokasi Tu...	Status	Waktu	Tanggal	Tarif
13	1	1	Kayu Tangi	Belitung	Selesai	02:29:28	2024-12-18	20000,00
14	9	6	Cendana	pangeran	Proses	02:47:47	2024-12-18	25000,00
15	10	1	Kidaung ...	Pasar la...	Selesai	03:29:56	2024-12-18	25000,00
- Buttons:** 'Tambah' (green), 'Edit', 'Hapus', and 'Batal' (grey).

- Tampilan ketika data orderan ditambahkan

The 'Form Order' window after adding a new order:

- Lokasi Awal:** Kidaung Permai
- Lokasi Tujuan:** Pasar Lama
- Status:** Selesai
- User Information:**
  - ID Pengguna:** 1
  - Nama Pengguna:** Indah
  - No HP:** 182135652640
  - Cari:** Button next to ID and Name fields.
- Driver Information:**
  - ID Driver:** 4
  - Kendaraan:** Scoopy
  - Tarif:** 10000
  - Cari:** Button next to ID and Vehicle fields.
- Table of Orders:**

ID Order	ID Pengg...	ID Driver	Lokasi A...	Lokasi Tu...	Status	Waktu	Tanggal	Tarif
13	1	1	Kayu Tangi	Belitung	Selesai	02:29:28	2024-12-18	20000,00
14	9	6	Cendana	pangeran	Proses	02:47:47	2024-12-18	25000,00
15	10	1	Kidaung ...	Pasar la...	Selesai	03:29:56	2024-12-18	25000,00
- Buttons:** 'Tambah' (green), 'Edit', 'Hapus', and 'Batal' (grey).
- Message Dialog:** A 'Message' box with an information icon and the text 'Data Tersimpan' (Data Saved), with an 'OK' button.

- Tampilan ketika data orderan diubah

The screenshot shows the 'Form Order' application interface. The form contains the following fields:

- Lokasi Awal:** Kidaung Permai
- Lokasi Tujuan:** Pasar Lama
- Status:** Proses
- ID Pengguna:** 1
- Nama Pengguna:** Indah
- No HP:** 082135652640
- ID Driver:** 4
- Kendaraan:** Scoopy
- Tarif:** 10000

A message box titled 'Message' is displayed, stating 'Data berhasil diubah!' (Data successfully updated!). Below the form is a table with the following data:

ID Order	ID Pengg...	ID Driver	Lokasi A...	Lokasi Tu...	Status	Waktu	Tanggal	Tarif
13	1	1	Kayu Tangi	Belitung	Selesai	02:29:28	2024-12-18	20000.00
14	9	6	Cendana	pangeran	Proses	02:47:47	2024-12-18	25000.00
15	10	1	Kidaung ...	Pasar la...	Selesai	03:29:56	2024-12-18	25000.00
19	1	4	Kidaung ...	Pasar La...	Selesai	15:07:32	2024-12-18	10000.00

At the bottom of the form are four buttons: 'Tambah', 'Edit', 'Hapus', and 'Batal'.

- Tampilan ketika data orderan dihapus

The screenshot shows the 'Form Order' application interface. The form contains the following fields:

- Lokasi Awal:** Cendana
- Lokasi Tujuan:** pangeran
- Status:** Proses
- ID Pengguna:** 9
- Nama Pengguna:** (empty)
- No HP:** (empty)
- ID Driver:** 6
- Kendaraan:** (empty)
- Tarif:** (empty)

A message box titled 'Message' is displayed, stating 'Data berhasil dihapus!' (Data successfully deleted!). Below the form is a table with the following data:

ID Order	ID Pengg...	ID Driver	Lokasi A...	Lokasi Tu...	Status	Waktu	Tanggal	Tarif
13	1	1	Kayu Tangi	Belitung	Selesai	02:29:28	2024-12-18	20000.00
14	9	6	Cendana	pangeran	Proses	02:47:47	2024-12-18	25000.00
15	10	1	Kidaung ...	Pasar la...	Selesai	03:29:56	2024-12-18	25000.00
19	1	4	Kidaung ...	Pasar La...	Proses	15:07:32	2024-12-18	10000.00

At the bottom of the form are four buttons: 'Tambah', 'Edit', 'Hapus', and 'Batal'.

#### 4) Form Pembayaran

**Form Pembayaran**

Tanggal

NO Pembayaran

ID Order  Cari  Lokasi Awal  Lokasi Tujuan

Metode Bayar  Harga Bayar

No Pembayar...	ID Order	Metode Bayar	Total Bayar	Tanggal Pem...
3	14	Transfer	15000.00	2024-12-18
4	13	Tunai	25000.00	2024-12-11
5	15	Tunai	25000.00	2024-12-18
6	14	Tunai	15000.00	2024-12-18

Cari Data

- Tampilan ketika data pembayaran ditambahkan

**Form Pembayaran**

Tanggal

NO Pembayaran

ID Order  Cari  Lokasi Awal  Lokasi Tujuan

Metode Bayar  Harga Bayar

No Pembayar...	ID Order	Metode Bayar	Total Bayar	Tanggal Pem...
4	13	Tunai	25000.00	2024-12-11
5	15	Tunai	25000.00	2024-12-18
8	13	Tunai	25000.00	2024-12-18
9	15	Tunai	10000.00	2024-12-18

Cari Data

Message

Data Pembayaran Berhasil Ditambahkan!

- Tampilan ketika data pembayaran diubah

The screenshot shows the 'Form Pembayaran' window with the following details:

- Tanggal:** 14-12-18
- NO Pembayaran:** 13
- ID Order:** 13 (with a 'Cari' button)
- Lokasi Awal:** Kayu Tangi
- Lokasi Tujuan:** Belitung
- Metode Bayar:** Transfer (selected from a dropdown)
- Harga Bayar:** 20000.00

A message box titled 'Message' is displayed with the text 'Data berhasil diubah!' and an 'OK' button.

No Pembayaran...	ID Order	Metode Bayar	Total Bayar	Tanggal Pem...
10	13	E-Wallet	25000.00	2024-12-18
12	13	Tunai	20000.00	2024-12-18
13	13	Tunai	20000.00	2024-12-18
14	13	Transfer	20000.00	2024-12-18

At the bottom, there is a 'Cari Data' section with a dropdown menu set to 'no\_pembayaran' and an empty search input field. Below this are five buttons: 'Struk' (blue), 'Tambah' (green), 'Edit' (orange), 'Hapus' (red), and 'Batal' (yellow).

- Tampilan ketika data pembayaran diubah

This screenshot is similar to the one above, but with the 'Metode Bayar' dropdown set to 'E-Wallet'.

The message box still displays 'Data berhasil diubah!' with an 'OK' button.

No Pembayaran...	ID Order	Metode Bayar	Total Bayar	Tanggal Pem...
10	13	E-Wallet	25000.00	2024-12-18
12	13	Tunai	20000.00	2024-12-18
13	13	Transfer	20000.00	2024-12-18
14	13	Transfer	20000.00	2024-12-18

The 'Cari Data' section and buttons at the bottom remain the same as in the previous screenshot.

- Tampilan ketika ingin melihat struk data pembayaran

**Form Pembayaran**

Tanggal: 14-12-18

NO Pembayaran: 13

ID Order: 13 Cari Lokasi Awal: Kayu Tangi Lokasi Tujuan: Belitung

Metode Bayar: E-Wallet Harga Bayar: 20000.00

No Pembayaran...	ID Order	Metode Bayar	Total Bayar
10	13	E-Wallet	25000.00
12	13	Tunai	20000.00
13	13	E-Wallet	20000.00
14	13	Transfer	20000.00

Cari Data: no\_pembayaran

Struk Pembayaran

STRUK PEMBAYARAN  
No. Pembayaran: 13  
ID Order: 13  
Metode Bayar: E-Wallet  
Total Bayar: Rp 20000.00  
Tanggal Bayar: 2024-12-18

Struk Tambah Edit Hapus Batal

## 5) List Pengguna

**List Pengguna**

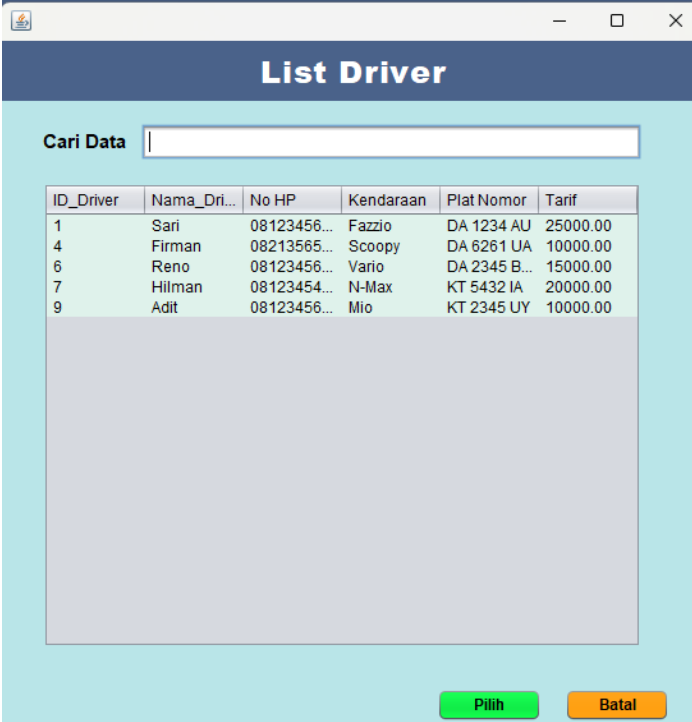
Cari Data

ID_Pengguna	Nama_Peng...	Email	No HP	Alamat
1	Indah	indah@gmail	0821356526...	Pangeran
3	Yulia	yulia@gmail	0878765432	Cendana 2B
7	Indri	indri@gmail	0812345678...	Kayu Tangi 2
8	Rani	rani@gmail	0823467893...	Belitung
9	Lidya	lidya@gmail	0812345678...	Sei Andai
10	Raudah	raudah@gmail	0821567435...	Pasar Lama
11	Ersa	ersa@gmail	0812345678...	kuin cerucuk

Pilih Batal



## 6) List Driver



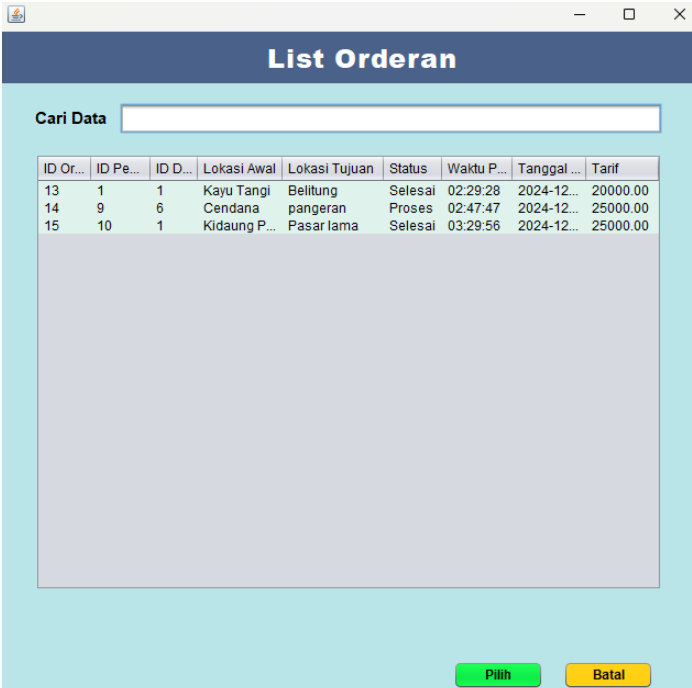
**List Driver**

Cari Data

ID_Driver	Nama_Dri...	No HP	Kendaraan	Plat Nomor	Tarif
1	Sari	08123456...	Fazzio	DA 1234 AU	25000.00
4	Firman	08213565...	Scoopy	DA 6261 UA	10000.00
6	Reno	08123456...	Vario	DA 2345 B...	15000.00
7	Hilman	08123454...	N-Max	KT 5432 IA	20000.00
9	Adit	08123456...	Mio	KT 2345 UY	10000.00

Pilih Batal

## 7) List Orderan



**List Orderan**

Cari Data

ID Or...	ID Pe...	ID D...	Lokasi Awal	Lokasi Tujuan	Status	Waktu P...	Tanggal ...	Tarif
13	1	1	Kayu Tangi	Belitung	Selesai	02:29:28	2024-12...	20000.00
14	9	6	Cendana	pangeran	Proses	02:47:47	2024-12...	25000.00
15	10	1	Kidaung P...	Pasar lama	Selesai	03:29:56	2024-12...	25000.00

Pilih Batal

## **BAB III**

### **PENUTUP**

#### **3.1 Kesimpulan**

Sistem pendataan ojek pangkalan berbasis komputer adalah solusi untuk mengatasi masalah dalam pengelolaan ojek pangkalan. Sistem ini mencatat data pengguna, driver, orderan, dan pembayaran secara otomatis, sehingga pengelolaan menjadi lebih efisien dan terorganisir. Dengan menggunakan teknologi seperti bahasa pemrograman Java dan platform NetBeans, sistem ini membantu operasional ojek pangkalan menjadi lebih baik dan meminimalkan kesalahan

#### **3.2 Saran**

Sistem ini bisa lebih baik dengan menambahkan fitur pelacakan lokasi real-time, sehingga pengguna dan driver dapat lebih mudah berkomunikasi dan memantau perjalanan. Keamanan data juga sangat penting agar informasi pengguna dan driver terlindungi dengan baik. Antarmuka sistem perlu diperbaiki agar lebih mudah digunakan oleh semua orang, sehingga pengalaman pengguna menjadi lebih nyaman. Selain itu, sistem sebaiknya terintegrasi dengan aplikasi pembayaran digital untuk mempermudah transaksi antara pengguna dan driver, sehingga proses pembayaran menjadi lebih cepat dan praktis.

## DAFTAR PUSTAKA

Deitel, P., & Deitel, H. (2015). *Java How to Program (10th Edition)*. Pearson.

**Yulianto, D., & Prabowo, H. (2019).** *Pengembangan Sistem Pemesanan Ojek Online Berbasis Web dan Mobile*. Jurnal Teknologi Informasi, 14(2), 123-130.



