LAPORAN AKHIR PRAKTIKUM

Mata Praktikum	**:** Pengenalan Teknologi Komputer & Informasi
Kelas	**:** 4IA24
Praktikum ke-	**:** 4
Tanggal	**:** 4 November 2024
Materi	**:** ORM
NPM	**:** 50421856
Nama	**:** Muhamad Alfin Surya Pratama
Ketua Asisten	**:**
Nama Asisten	**:** Robby Nugraha
Paraf Asisten	**:**
Jumlah Lembar	**:** 4 (empat)

LABORATORIUM TEKNIK INFORMATIKA

UNIVERSITAS GUNADARMA

2024

LISTING

```
import javax.swing.*;
import javax.swing.table.*;
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.time.LocalDateTime;
import java.time.Duration;
import java.sql.*;

// Class untuk mengelola koneksi database
class DatabaseConnection {
    private static final String DB_URL = "jdbc:mysql://localhost:3306/rental_kendaraan";
    private static final String USER = "root";
    private static final String PASS = "password";

    public static Connection getConnection() throws SQLException {
        return DriverManager.getConnection(DB_URL, USER, PASS);
    }
}

class Kendaraan {
    String platNomor;
    String jenis;
    double tarifPerHari;
    boolean tersedia;

    public Kendaraan(String platNomor, String jenis, double tarifPerHari, boolean tersedia) {
        this.platNomor = platNomor;
        this.jenis = jenis;
        this.tarifPerHari = tarifPerHari;
        this.tersedia = tersedia;
    }
}

class Penyewaan {
    int id;
    String platNomor;
    String namaPenyewa;
    LocalDateTime waktuMulai;

    public Penyewaan(int id, String platNomor, String namaPenyewa, LocalDateTime waktuMulai) {
        this.id = id;
        this.platNomor = platNomor;
        this.namaPenyewa = namaPenyewa;
        this.waktuMulai = waktuMulai;
    }
}

public class SistemRentalKendaraanGUI extends JFrame {
    private JTable tabelKendaraan;
    private DefaultTableModel modelTabel;
    private JButton btnSewa, btnKembali, btnRefresh;
    private JPanel mainPanel;

    public SistemRentalKendaraanGUI() {
        super("Sistem Rental Kendaraan");
        setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        setSize(800, 500);
        setLocationRelativeTo(null);

        setupDatabase();
        setupKomponen();
        refreshTabel();
    }

    private void setupDatabase() {
        try (Connection conn = DatabaseConnection.getConnection()) {
```

```
        String createKendaraanTable = """
            CREATE TABLE IF NOT EXISTS kendaraan (
                plat_nomor VARCHAR(20) PRIMARY KEY,
                jenis VARCHAR(50),
                tarif_per_hari DOUBLE,
                tersedia BOOLEAN
            )
        """;

        // Buat tabel penyewaan jika belum ada
        String createPenyewaanTable = """
            CREATE TABLE IF NOT EXISTS penyewaan (
                id INT AUTO_INCREMENT PRIMARY KEY,
                plat_nomor VARCHAR(20),
                nama_penyewa VARCHAR(100),
                waktu_mulai DATETIME,
                waktu_selesai DATETIME,
                FOREIGN KEY (plat_nomor) REFERENCES kendaraan(plat_nomor)
            )
        """;

        try (Statement stmt = conn.createStatement()) {
            stmt.execute(createKendaraanTable);
            stmt.execute(createPenyewaanTable);

            // Cek apakah ada data awal
            ResultSet rs = stmt.executeQuery("SELECT COUNT(*) FROM kendaraan");
            rs.next();
            if (rs.getInt(1) == 0) {
                // Tambah data awal
                String[] initData = {
                    "INSERT INTO kendaraan VALUES ('B 1234 CD', 'Mobil', 300000, true)",
                    "INSERT INTO kendaraan VALUES ('B 5678 EF', 'Motor', 100000, true)",
                    "INSERT INTO kendaraan VALUES ('B 9012 GH', 'Mobil', 350000, true)"
                };
                for (String sql : initData) {
                    stmt.execute(sql);
                }
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(this,
            "Error connecting to database: " + e.getMessage(),
            "Database Error",
            JOptionPane.ERROR_MESSAGE);
    }
}

private void refreshTabel() {
    modelTabel.setRowCount(0);
    try (Connection conn = DatabaseConnection.getConnection();
         Statement stmt = conn.createStatement();
         ResultSet rs = stmt.executeQuery("SELECT * FROM kendaraan")) {

        while (rs.next()) {
            modelTabel.addRow(new Object[]{
                rs.getString("plat_nomor"),
                rs.getString("jenis"),
                String.format("Rp %.2f", rs.getDouble("tarif_per_hari")),
                rs.getBoolean("tersedia") ? "Tersedia" : "Disewa"
            });
        }
    } catch (SQLException e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(this,
            "Error refreshing table: " + e.getMessage(),
            "Database Error",
            JOptionPane.ERROR_MESSAGE);
    }
}
```

```
private void sewaKendaraan() {
    int selectedRow = tabelKendaraan.getSelectedRow();
    if (selectedRow == -1) {
        JOptionPane.showMessageDialog(this,
            "Pilih kendaraan yang ingin disewa terlebih dahulu",
            "Peringatan",
            JOptionPane.WARNING_MESSAGE);
        return;
    }

    String platNomor = (String) tabelKendaraan.getValueAt(selectedRow, 0);

    try (Connection conn = DatabaseConnection.getConnection()) {
        // Cek ketersediaan kendaraan
        String checkSql = "SELECT tersedia FROM kendaraan WHERE plat_nomor = ? AND tersedia = true";
        try (PreparedStatement pstmt = conn.prepareStatement(checkSql)) {
            pstmt.setString(1, platNomor);
            ResultSet rs = pstmt.executeQuery();

            if (!rs.next()) {
                JOptionPane.showMessageDialog(this,
                    "Kendaraan tidak tersedia untuk disewa",
                    "Error",
                    JOptionPane.ERROR_MESSAGE);
                return;
            }
        }

        String namaPenyewa = JOptionPane.showInputDialog(this,
            "Masukkan nama penyewa:",
            "Sewa Kendaraan",
            JOptionPane.QUESTION_MESSAGE);

        if (namaPenyewa != null && !namaPenyewa.trim().isEmpty()) {
            conn.setAutoCommit(false);
            try {
                // Update status kendaraan
                String updateKendaraan = "UPDATE kendaraan SET tersedia = false WHERE plat_nomor = ?";
                try (PreparedStatement pstmt = conn.prepareStatement(updateKendaraan)) {
                    pstmt.setString(1, platNomor);
                    pstmt.executeUpdate();
                }

                // Tambah record penyewaan
                String insertPenyewaan = """
                    INSERT INTO penyewaan (plat_nomor, nama_penyewa, waktu_mulai)
                    VALUES (?, ?, NOW())
                """;
                try (PreparedStatement pstmt = conn.prepareStatement(insertPenyewaan)) {
                    pstmt.setString(1, platNomor);
                    pstmt.setString(2, namaPenyewa);
                    pstmt.executeUpdate();
                }

                conn.commit();
                JOptionPane.showMessageDialog(this,
                    "Kendaraan berhasil disewa oleh " + namaPenyewa,
                    "Sukses",
                    JOptionPane.INFORMATION_MESSAGE);

                refreshTabel();
            } catch (SQLException e) {
                conn.rollback();
                throw e;
            }
        }
    } catch (SQLException e) {
        e.printStackTrace();
        JOptionPane.showMessageDialog(this,
            "Error saat menyewa kendaraan: " + e.getMessage(),
            "Database Error",
```

```
            JOptionPane.ERROR_MESSAGE);
    }
}

private void kembalikanKendaraan() {
    int selectedRow = tabelKendaraan.getSelectedRow();
    if (selectedRow == -1) {
        JOptionPane.showMessageDialog(this,
            "Pilih kendaraan yang ingin dikembalikan terlebih dahulu",
            "Peringatan",
            JOptionPane.WARNING_MESSAGE);
        return;
    }

    String platNomor = (String) tabelKendaraan.getValueAt(selectedRow, 0);

    try (Connection conn = DatabaseConnection.getConnection()) {
        conn.setAutoCommit(false);
        try {
            // Ambil data penyewaan
            String selectPenyewaan = """
                SELECT id, waktu_mulai, k.tarif_per_hari
                FROM penyewaan p
                JOIN kendaraan k ON p.plat_nomor = k.plat_nomor
                WHERE p.plat_nomor = ? AND p.waktu_selesai IS NULL
            """;

            double biaya = 0;
            int penyewaanId = 0;

            try (PreparedStatement pstmt = conn.prepareStatement(selectPenyewaan)) {
                pstmt.setString(1, platNomor);
                ResultSet rs = pstmt.executeQuery();

                if (!rs.next()) {
                    JOptionPane.showMessageDialog(this,
                        "Kendaraan ini tidak sedang disewa",
                        "Error",
                        JOptionPane.ERROR_MESSAGE);
                    return;
                }

                penyewaanId = rs.getInt("id");
                Timestamp waktuMulai = rs.getTimestamp("waktu_mulai");
                double tarifPerHari = rs.getDouble("tarif_per_hari");

                long durasiJam = Duration.between(
                    waktuMulai.toLocalDateTime(),
                    LocalDateTime.now()
                ).toHours();

                biaya = Math.ceil(durasiJam / 24.0) * tarifPerHari;
            }

            // Update status kendaraan
            String updateKendaraan = "UPDATE kendaraan SET tersedia = true WHERE plat_nomor = ?";
            try (PreparedStatement pstmt = conn.prepareStatement(updateKendaraan)) {
                pstmt.setString(1, platNomor);
                pstmt.executeUpdate();
            }

            // Update record penyewaan
            String updatePenyewaan = "UPDATE penyewaan SET waktu_selesai = NOW() WHERE id = ?";
            try (PreparedStatement pstmt = conn.prepareStatement(updatePenyewaan)) {
                pstmt.setInt(1, penyewaanId);
                pstmt.executeUpdate();
            }

            conn.commit();
            JOptionPane.showMessageDialog(this,
                String.format("Biaya sewa: Rp %.2f\nKendaraan berhasil dikembalikan.", biaya),
```

```
                    "Informasi Pengembalian",
                    JOptionPane.INFORMATION_MESSAGE);

                refreshTabel();
            } catch (SQLException e) {
                conn.rollback();
                throw e;
            }
        } catch (SQLException e) {
            e.printStackTrace();
            JOptionPane.showMessageDialog(this,
                "Error saat mengembalikan kendaraan: " + e.getMessage(),
                "Database Error",
                JOptionPane.ERROR_MESSAGE);
        }
    }

    // [setupKomponen() method tetap sama seperti sebelumnya]

    public static void main(String[] args) {
        try {
            // Load JDBC Driver
            Class.forName("com.mysql.cj.jdbc.Driver");

            // Set Look and Feel
            UIManager.setLookAndFeel(UIManager.getSystemLookAndFeelClassName());

            SwingUtilities.invokeLater(() -> {
                new SistemRentalKendaraanGUI().setVisible(true);
            });
        } catch (Exception e) {
            e.printStackTrace();
        }
    }
}
```

LOGIKA

1. Struktur database

   - Tabel kendaraan:
     - plat_nomor (Primary Key)
     - jenis
     - tarif_per_hari
     - tersedia (boolean)

   - Tabel penyewaan:
     - id (Auto Increment Primary Key)
     - plat_nomor (Foreign Key)
     - nama_penyewa
     - waktu_mulai
     - waktu_selesai

2. Alur program

   - Menjalankan aplikasi pertama

     Nge cek dan buat database

     Buat table kendaraan dan penyewaan

     Ngisi data awal 3 kendaraan ke table

     Menampilkan GUI dengan table kendaraan

   - Proses penyewaan

     1. User memilih kendaraan dari tabel
     2. User klik tombol "Sewa"
     3. Sistem cek apakah kendaraan tersedia
     4. Jika tersedia:
        - Muncul dialog input nama penyewa
        - Update status kendaraan jadi false (tidak tersedia)
        - Insert record baru di tabel penyewaan dengan:
          * plat_nomor terpilih
          * nama penyewa
          * waktu_mulai (sekarang)
          * waktu_selesai (null)
     5. Refresh tampilan table

- Proses pengembalian

  1. User pilih kendaraan dari tabel

  2. User klik tombol "Kembali"

  3. Sistem:

     - Cek record penyewaan yang belum selesai

     - Hitung durasi sewa (waktu_sekarang - waktu_mulai)

     - Hitung biaya = (durasi_dalam_hari) x tarif_per_hari

     - Update status kendaraan jadi true (tersedia)

     - Update waktu_selesai di record penyewaan

     - Tampilkan total biaya ke user

  4. Refresh tampilan tabel

3. Fitur keamanan

   - Menggunakan PreparedStatement untuk mencegah SQL Injection

   - Transaction management (commit/rollback) untuk menjaga konsistensi data

   - Try-catch untuk handling error database

   - Validasi input user

OUTPUT


Sistem Rental Kendaraan
Plat Nomor
Jenis
Tarif Per Hari
Status
B 1234 CD
Mobil
Rp 300,000.00
Tersedia
B 5678 EF
Motor
Rp 100,000.00
Tersedia
B 9012 GH
Mobil
Rp 350,000.00
Tersedia
Sewa
Kembali
Refresh