**LAPORAN UJIAN AKHIR SEMESTER**

**“ParkirKu: Aplikasi Pengelolaan Parkir Modern”**

Diajukan untuk Memenuhi Tugas Mata Kuliah Pemrograman Berorientasi Objek

Dosen Pengampu:

**Taufik Ridwan, M.T.**

****

Disusun Oleh:

**Kelompok 1**

| Raymond Marthin  Muhammad Fadlan Ziadh R.  Muhammad Taffazani Addien  Muhammad Rifqi Syazani | 2310631250075  2010631250066  2310631250102  2310631250101 |
| --- | --- |

**PROGRAM STUDI SISTEM INFORMASI**

**FAKULTAS ILMU KOMPUTER**

**UNIVERSITAS SINGAPERBANGSA KARAWANG**

**2025**

# PENDAHULUAN

## Latar Belakang

Jumlah kendaraan yang terus meningkat di tempat umum seperti pusat perbelanjaan, kampus, dan gedung perkantoran seringkali menimbulkan tantangan dalam pengelolaan parkir. Pencatatan kendaraan secara manual cenderung tidak rapi, rawan kesalahan, dan menyulitkan proses pengawasan serta pelacakan data. Untuk mengatasi masalah ini, dibutuhkan aplikasi berbasis komputer yang dapat membantu petugas dalam mencatat, mengatur, dan memantau kendaraan yang keluar-masuk dengan lebih cepat, akurat, dan sistematis.

Dalam konteks ini, dibutuhkan sebuah aplikasi sistem parkir yang mampu mengelola data kendaraan, pengguna (petugas), dan transaksi parkir secara terstruktur, aman, dan efisien. Oleh karena itu, dikembangkanlah aplikasi "ParkirKu: Aplikasi Pengelolaan Parkir Modern" berbasis Java dengan pendekatan Object-Oriented Programming (OOP), yang dapat memberikan kemudahan dalam pengelolaan data serta fleksibilitas dalam pengembangan fitur di masa mendatang.

Aplikasi ini dikembangkan menggunakan bahasa pemrograman Java, dengan antarmuka grafis berbasis Java Swing, serta menggunakan MySQL/MariaDB sebagai sistem basis data. Dengan penerapan konsep OOP, aplikasi disusun secara modular melalui pemisahan kelas dan objek, sehingga lebih terstruktur, mudah diuji, dan dapat di-maintain atau dikembangkan lebih lanjut secara efisien.

## Rumusan Masalah

1. Bagaimana cara membuat aplikasi parkir yang mudah digunakan dan bisa dikembangkan ke depannya?
2. Bagaimana tampilan aplikasi dapat dibuat agar mudah dipahami oleh petugas dan pengelola parkir?
3. Bagaimana membedakan antara pengguna yang bertugas di lapangan dan pengguna yang mengelola sistem secara keseluruhan?

## Tujuan

1. Membuat aplikasi parkir yang praktis dan bisa berjalan di komputer.
2. Menyediakan tampilan yang sederhana dan mudah dioperasikan oleh pengguna.
3. Menyediakan dua jenis pengguna, yaitu petugas parkir dan pengelola (admin), dengan tugas dan akses yang berbeda.
4. Menyediakan fitur untuk mencatat kendaraan masuk dan keluar, mengatur data petugas, serta melihat laporan aktivitas parkir.

# PEMBAHASAN

# Fitur – Fitur

Aplikasi Manajemen Parkir ini dirancang untuk membantu pengelola parkir dalam mencatat dan memantau kendaraan secara digital. Sistem ini memiliki dua jenis pengguna utama, yaitu Admin dan Petugas Parkir, yang masing-masing memiliki fitur-fitur sesuai dengan peran dan tanggung jawabnya.

### **Fitur Main**

* **Login Berbasis Peran**  
  Setiap pengguna dapat login menggunakan *username* dan *password*. Sistem akan memberikan akses sesuai peran pengguna, yaitu sebagai Admin atau Petugas Parkir.

### **Fitur Petugas Parkir**

* **Login Petugas**  
  Petugas dapat masuk ke sistem menggunakan akun yang telah dibuat oleh Admin.
* **Input Kendaraan Masuk**  
  Petugas mencatat kendaraan yang masuk dengan mengisi nomor plat dan memilih jenis kendaraan (*motor* atau *mobil*). Waktu masuk akan tercatat otomatis oleh sistem.
* **Input Kendaraan Keluar**  
  Petugas mencatat kendaraan yang keluar. Sistem akan secara otomatis menghitung tarif parkir berdasarkan durasi parkir dan jenis kendaraan, lalu menampilkan total biaya yang harus dibayar.
* **Lihat Daftar Kendaraan yang Sedang Parkir**  
  Menampilkan daftar kendaraan yang saat ini sedang parkir di area.
* **Riwayat Transaksi**Menampilkan histori kendaraan yang pernah ditangani oleh petugas, termasuk waktu masuk dan keluar serta tarif yang dikenakan.
* **Logout** Petugas dapat keluar dari sesi login untuk menjaga keamanan akses.

### **Fitur Admin**

* **Login Admin** Admin dapat login menggunakan akun dengan hak akses penuh terhadap sistem.
* **Manajemen Petugas** Admin memiliki akses untuk menambah, mengedit, dan menghapus data petugas parkir. Selain itu, Admin juga dapat melihat daftar lengkap seluruh petugas.
* **Pengaturan Tarif Parkir** Admin dapat mengatur tarif parkir per jam untuk kendaraan roda dua (motor) dan roda empat (mobil).
* **Laporan Keuangan Harian dan Bulanan** Admin dapat melihat laporan harian maupun bulanan yang mencakup jumlah kendaraan yang parkir, total pemasukan, serta durasi rata-rata parkir.
* **Logout** Admin dapat keluar dari sesi login dan kembali ke halaman utama aplikasi.

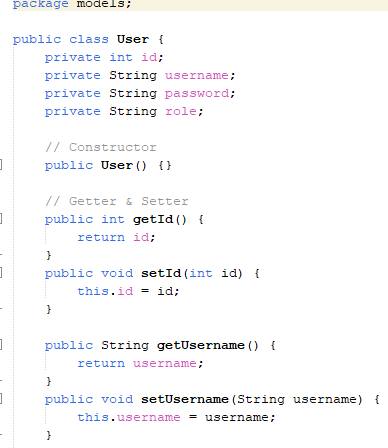
Secara keseluruhan, fitur-fitur pada aplikasi ini disusun dengan jelas dan terstruktur berdasarkan peran pengguna. Hal ini memungkinkan sistem bekerja secara efisien, aman, dan mudah digunakan dalam mendukung operasional pengelolaan parkir secara digital.

# Penjelasan Konsep OOP dalam Program

1. Encapsulation (Enkapsulasi)

Encapsulation adalah konsep untuk menyembunyikan detail implementasi internal suatu objek dari luar, dan hanya membiarkan akses melalui method publik. Ini meningkatkan keamanan dan menjaga integritas data.

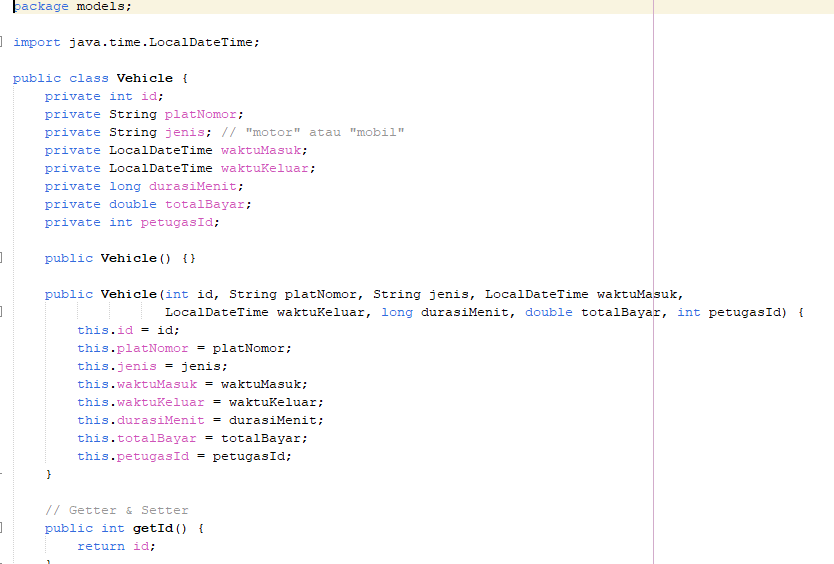
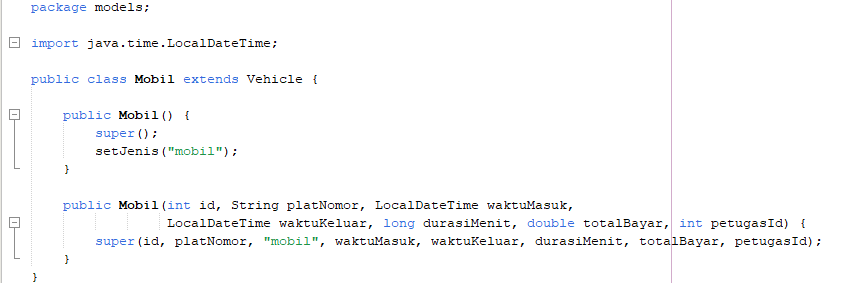
Contoh Pada class User



Atribut seperti id, username, password, dan role disimpan sebagai private, sehingga tidak bisa diakses langsung dari luar. Akses hanya diberikan melalui method get dan set, yang memungkinkan validasi dan kontrol.

1. Inheritance (Pewarisan)

Inheritance adalah konsep pewarisan sifat dari satu kelas ke kelas lain. Kelas anak (subclass) mewarisi atribut dan method dari kelas induk (superclass), sehingga dapat menggunakan kembali kode yang sudah ada dan memungkinkan perluasan fungsionalitas tanpa mengubah kelas asli.

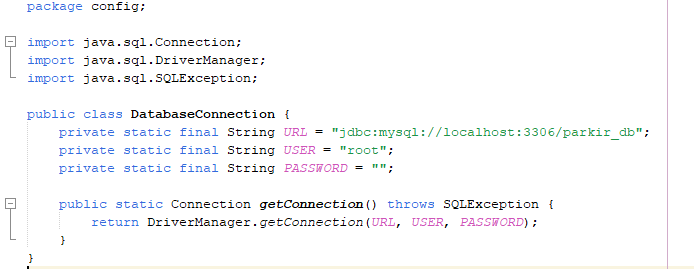
Contoh Pada Superclass Vehicle dan Subclass Mobil  


Ini memungkinkan kode menjadi lebih efisien dan fleksibel, karena kita bisa menangani berbagai jenis kendaraan (seperti Mobil, Motor, dll.) dengan satu struktur dasar Vehicle, namun masing-masing bisa memiliki perilaku unik.

1. Abstraksi

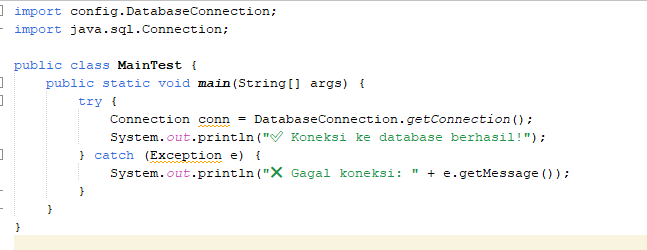
Abstraction adalah menyembunyikan kompleksitas dan hanya menampilkan esensi dari objek kepada pengguna. Biasanya ini dilakukan dengan menyederhanakan akses melalui method tertentu dan menyembunyikan implementasi detailnya.

Contoh Implementasi pada DatabaseConnection

Pengguna kelas hanya cukup memanggil getConnection() tanpa perlu tahu detail driver, host, port, dll. Detail teknis tersembunyi untuk menyederhanakan penggunaan.

1. Exception

Exception adalah mekanisme untuk menangani error atau kejadian tak terduga saat program berjalan, agar program tidak langsung crash dan bisa menampilkan pesan yang informatif atau melakukan penanganan yang tepat.



Dalam kode di atas, blok try digunakan untuk mencoba melakukan koneksi ke database melalui DatabaseConnection.getConnection(). Jika koneksi gagal karena alasan seperti driver tidak ditemukan, kesalahan koneksi, atau database tidak tersedia, maka akan terjadi exception.

Blok catch (Exception e) menangkap semua jenis error (Exception) yang mungkin terjadi. Dengan menggunakan e.getMessage(), program menampilkan pesan kesalahan ke konsol tanpa membuat aplikasi berhenti mendadak (crash).

1. Polymorphism (Polimorfisme)

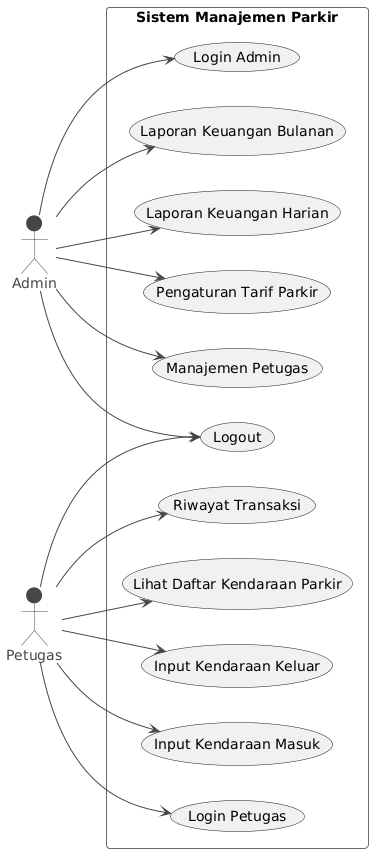
Polymorphism memungkinkan method yang sama memiliki perilaku berbeda, tergantung objek atau konteksnya. Dalam aplikasi ini, polimorfisme terlihat dari penggunaan method login() dan perbedaan hasil tergantung peran pengguna.



Method login() di sini menangani objek User, dan berperilaku berbeda tergantung role: jika "admin", maka membuka AdminDashboard; jika bukan, maka membuka StaffDashboard.

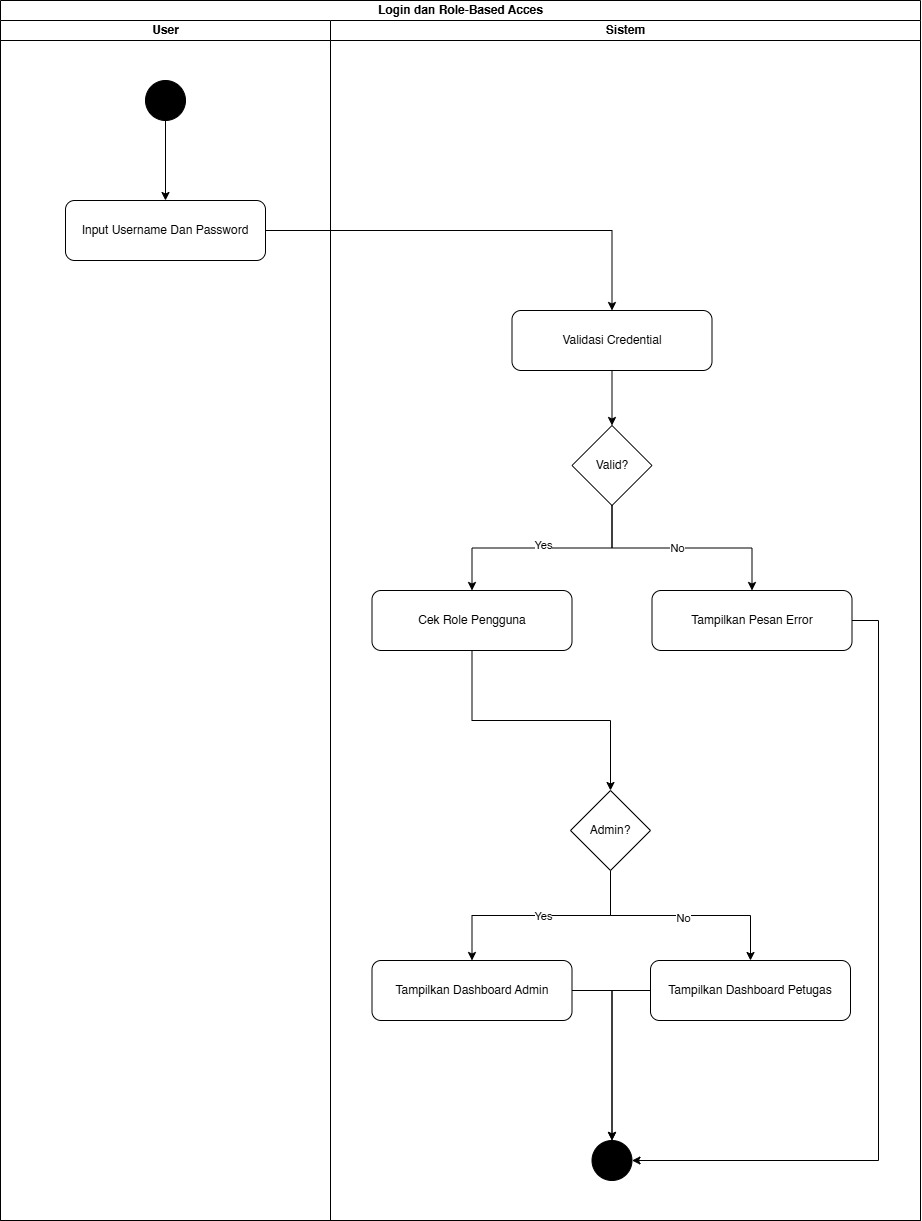
Ini adalah bentuk polimorfisme dinamis melalui pemilihan objek view yang berbeda berdasarkan tipe pengguna.

# Unified Modelling Language (UML) Use Case Diagram

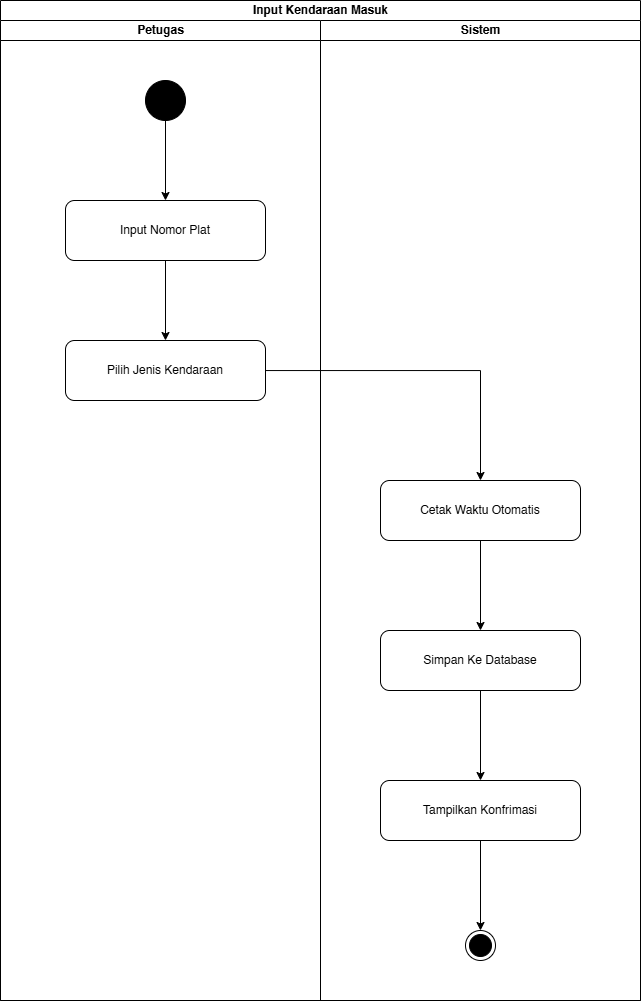


**Activity Diagram**

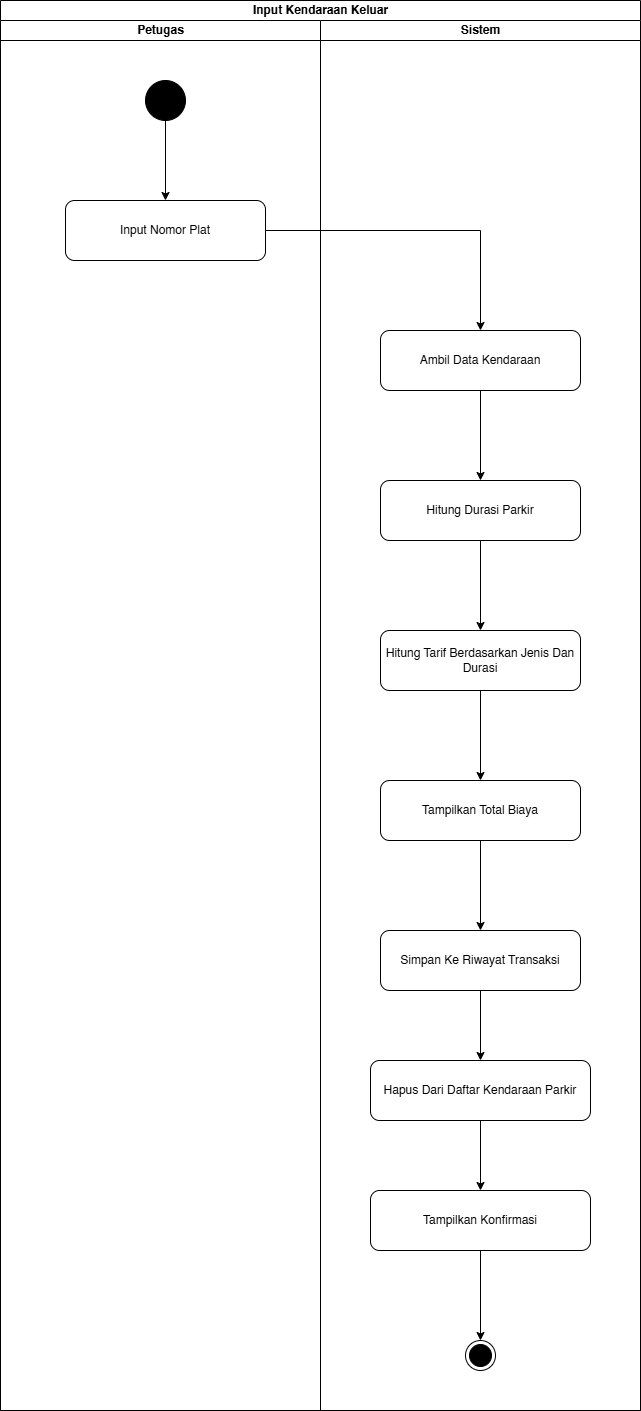
1. **Login**

****

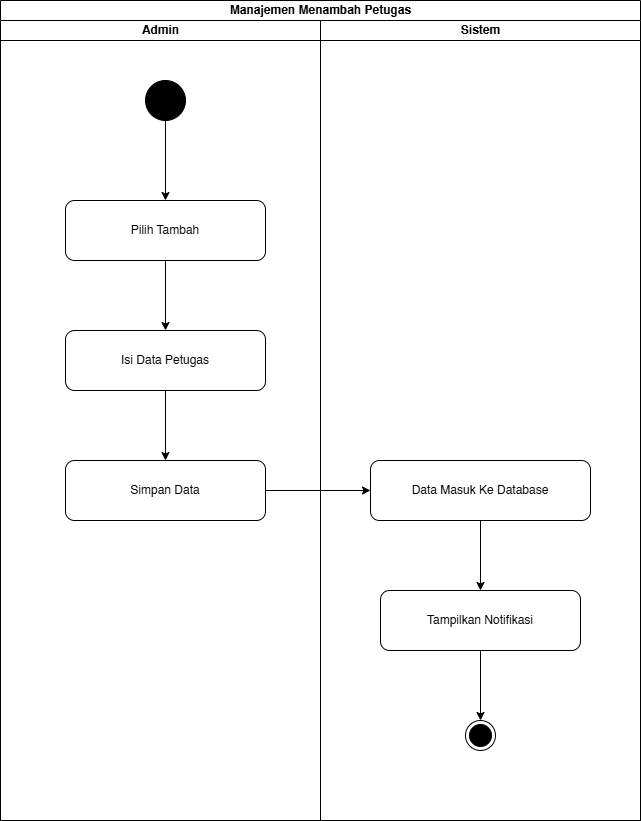
1. **Input Kendaraan Masuk (Petugas)**

****

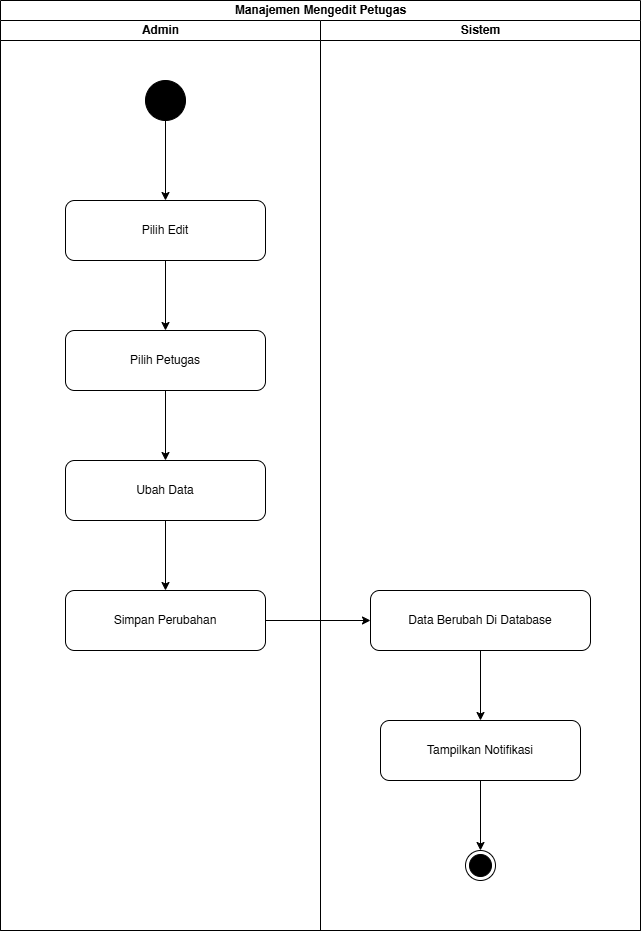
1. **Input Kendaraan Keluar (Petugas)**

****

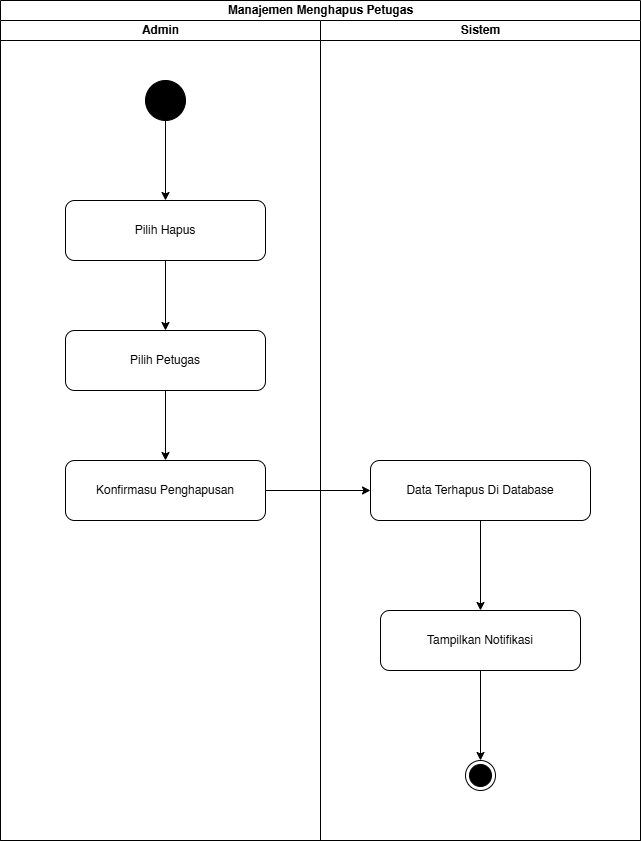
1. **Manajemen Menambah Petugas (Admin)**

****

1. **Manajemen Mengedit Petugas (Admin)**

****

1. **Manajemen Menghapus Petugas (Admin)**

****

# Implementasi Kode Program Dan Pengujian

Proyek ini menerapkan arsitektur MVC (Model-View-Controller) dengan Java Swing dan JDBC untuk membangun sistem manajemen parkir yang melibatkan login, pencatatan kendaraan, serta pengelolaan data petugas dan transaksi.

1. Model

Model mewakili data atau entitas dalam sistem.

* models/User.java

Menyimpan data pengguna seperti id, username, password, dan role (admin/petugas).

Menggunakan konsep encapsulation: semua atribut bersifat private dengan akses getter dan setter.

* models/Vehicle.java (Superclass)

Digunakan sebagai dasar untuk kendaraan.

* models/Mobil.java & Motor.java

Meng-extend Vehicle.

1. Controller

Menangani logika aplikasi (business logic) dan interaksi antara View & Model.

* controllers/VehicleController.java

Fungsi insertMasuk() untuk mencatat kendaraan masuk.

Fungsi prosesKeluar() menghitung tarif dan mencatat kendaraan keluar.

* controllers/AuthController.java

Menangani login user berdasarkan role-nya.

* controllers/AdminController.java

Melakukan operasi CRUD data user (khusus admin).

1. DAO (Data Access Object)

DAO (Data Access Object) adalah cara (pola) dalam menulis kode agar akses ke database dipisah dari logika utama aplikasi. Dengan DAO kode yang berhubungan dengan database (seperti INSERT, SELECT, UPDATE) ditulis di satu tempat saja. Bagian lain dari aplikasi (misalnya tampilan atau tombol) tidak perlu tahu detail cara ambil atau simpan data.

* dao/VehicleDAO.java

insertMasuk() → Menyimpan data kendaraan masuk.

processKeluar() → Menghitung durasi dan tarif, menyimpan data checkout.  
 getParkedVehicles() → Menampilkan kendaraan yang masih parkir.  
 getTransactionHistory() → Menampilkan histori transaksi berdasarkan petugas.

* dao/UserDAO.java  
  login() → Verifikasi user berdasarkan username dan password.  
  getAllPetugas(), addUser(), editUser(), deleteUser() → CRUD data user.
* dao/TariffDAO.java  
  Mengambil tarif kendaraan berdasarkan jenis (Mobil atau Motor).
* config/DatabaseConnection.java  
  Mengatur koneksi ke database MySQL.

1. View (Java Swing GUI)

Merupakan antarmuka pengguna untuk menjalankan sistem.

* views/LoginForm.java

Tampilan login.

* views/AdminDashboard.java

Menu utama admin untuk kelola petugas dan transaksi.

* views/StaffDashboard.java

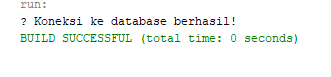
Menu utama petugas untuk kelola kendaraan parkir dan melihat daftar parkir serta riwayat transaksi..

* views/AddPetugasDialog.java Dan EditPetugasDialog.java

Form dialog untuk tambah/edit data petugas.

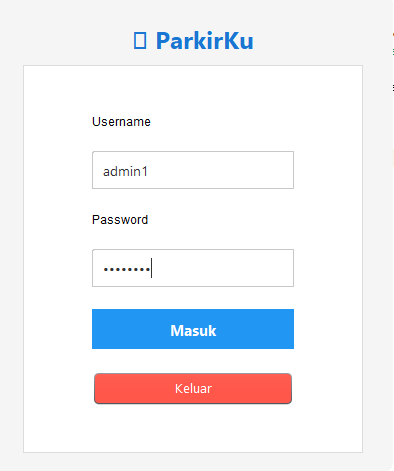
**PENGUJIAN SISTEM**

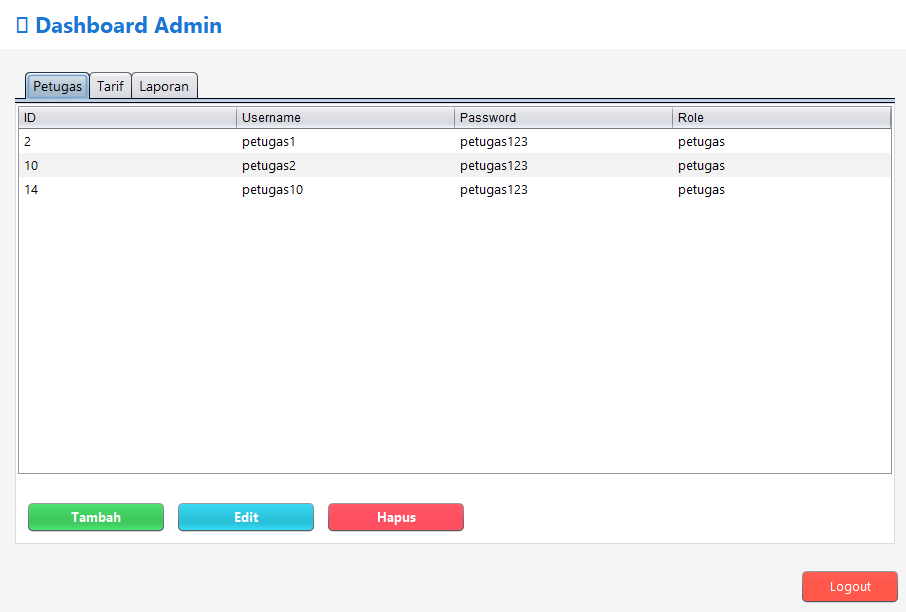
1. Pengujian Koneksi Database



Pengujian dilakukan dengan menjalankan File MainTes.java Digunakan untuk memastikan koneksi ke database berhasil dilakukan.

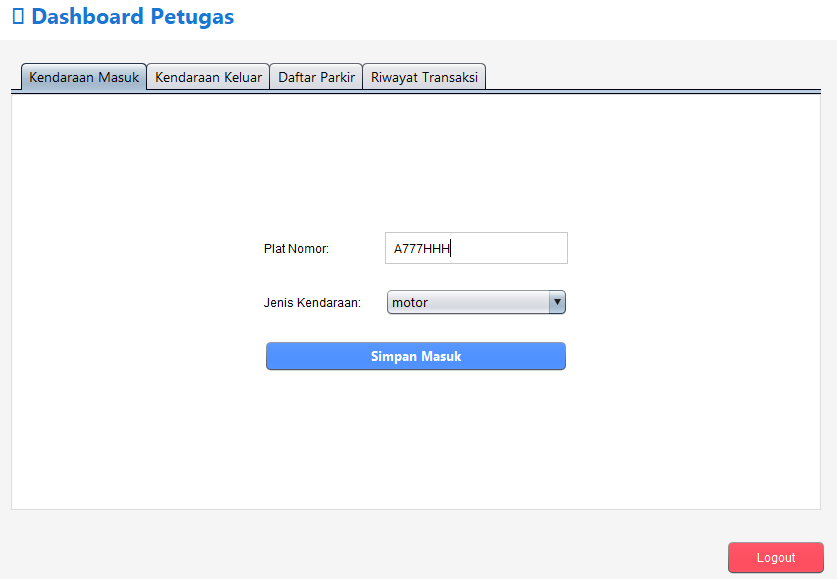
1. Pengujian Login

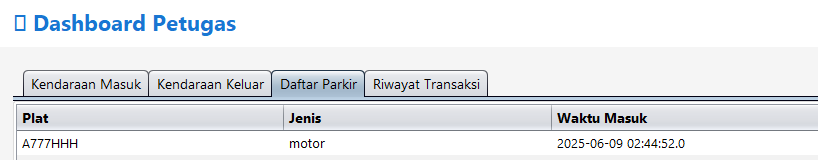


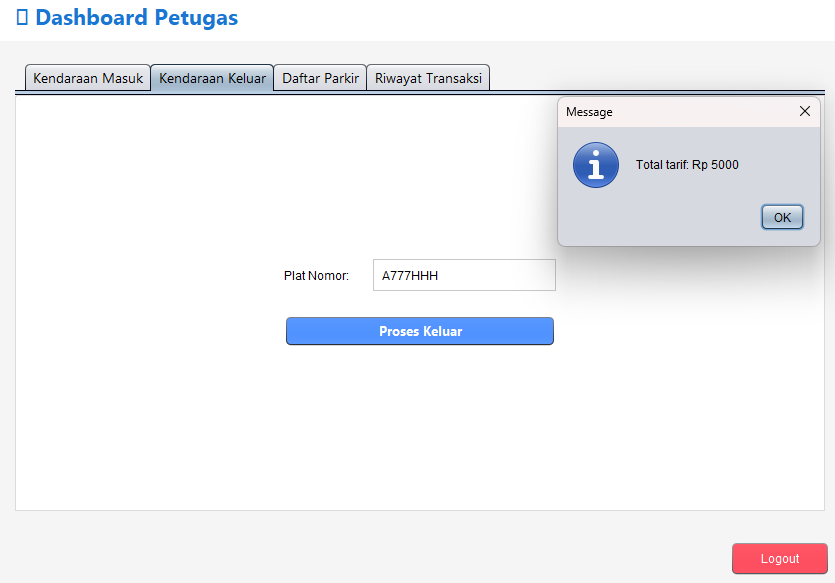


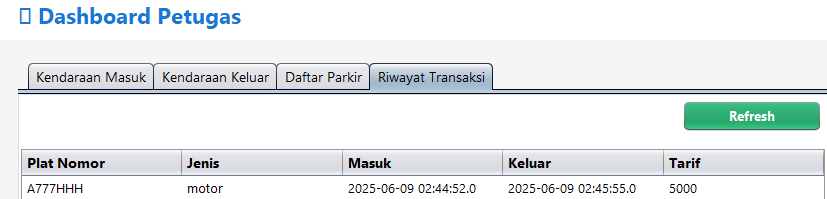
Pengguna memasukkan username dan password yang sesuai. Setelah berhasil login, sistem akan mengarahkan pengguna ke tampilan yang sesuai dengan role masing-masing, yaitu Admin atau Petugas. Pada contoh yang ditampilkan pada gambar, pengguna login sebagai Admin, sehingga sistem secara otomatis menampilkan Dashboard Admin sebagai antarmuka utama.

1. Pengujian Kendaraan Masuk dan Keluar







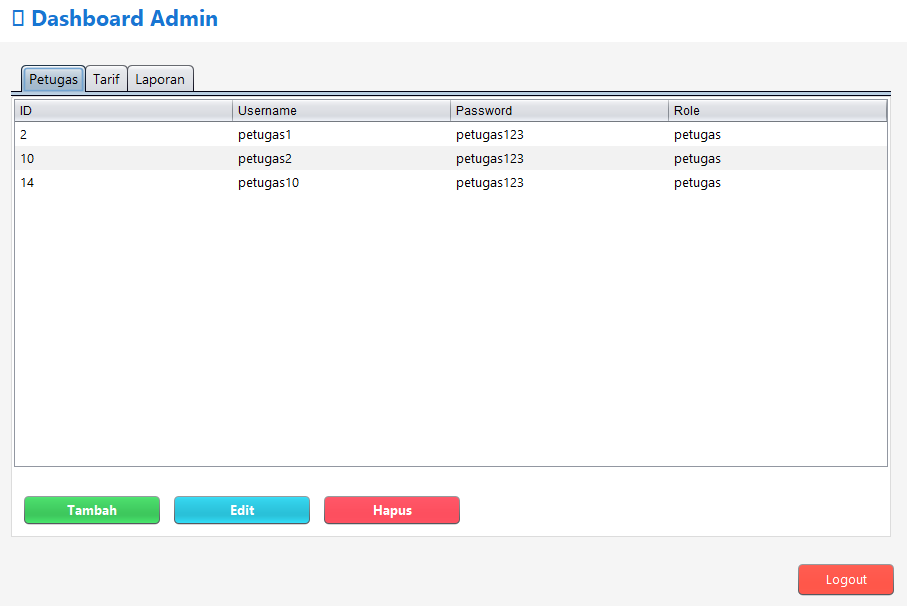


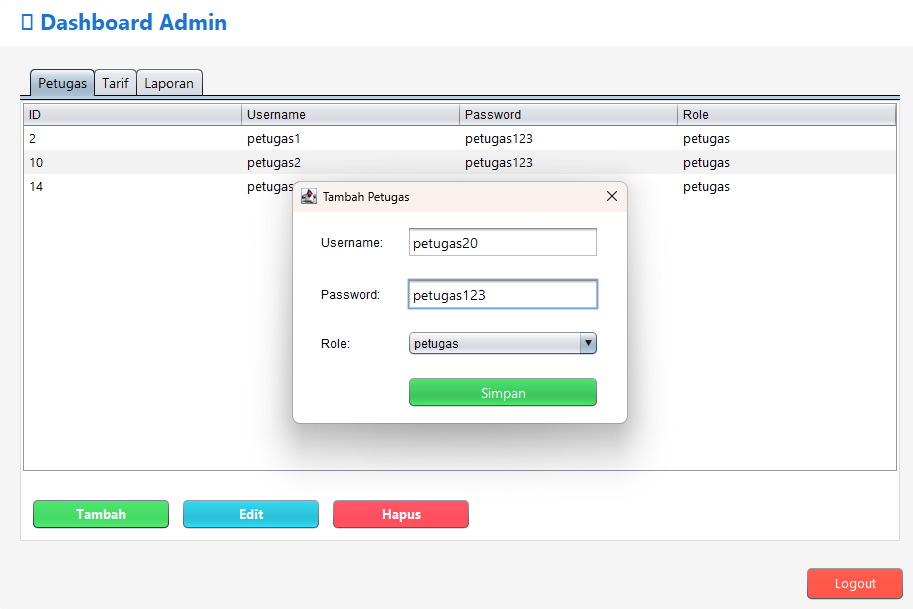
Pertama, petugas memasukkan plat nomor kendaraan yang akan parkir, kemudian mengklik tombol Simpan. Setelah itu, data kendaraan tersebut akan muncul pada daftar parkir yang sedang aktif.

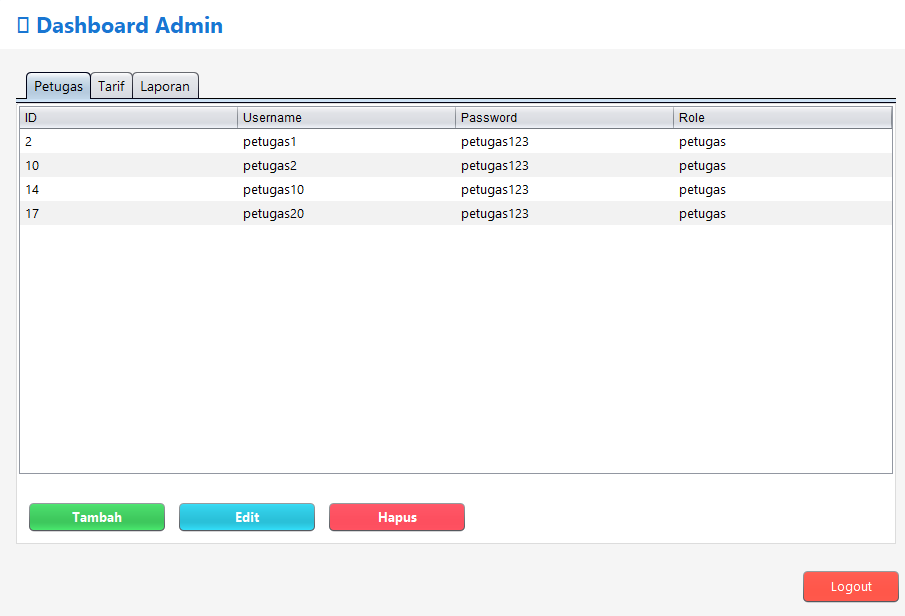
Untuk memproses kendaraan keluar, petugas kembali memasukkan plat nomor kendaraan yang sama, lalu mengklik tombol Proses Keluar. Sistem akan menghitung dan menampilkan tarif parkir berdasarkan durasi waktu parkir.

Selanjutnya, kendaraan yang telah keluar akan tercatat dalam riwayat transaksi, lengkap dengan informasi seperti plat nomor, jenis kendaraan, waktu masuk, waktu keluar, dan total biaya parkir.

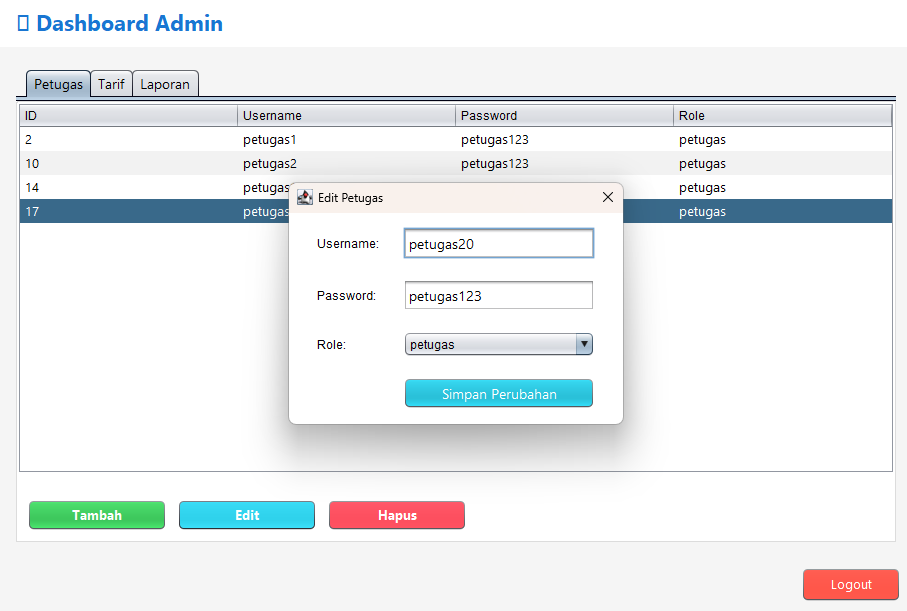
1. Pengujian CRUD Petugas

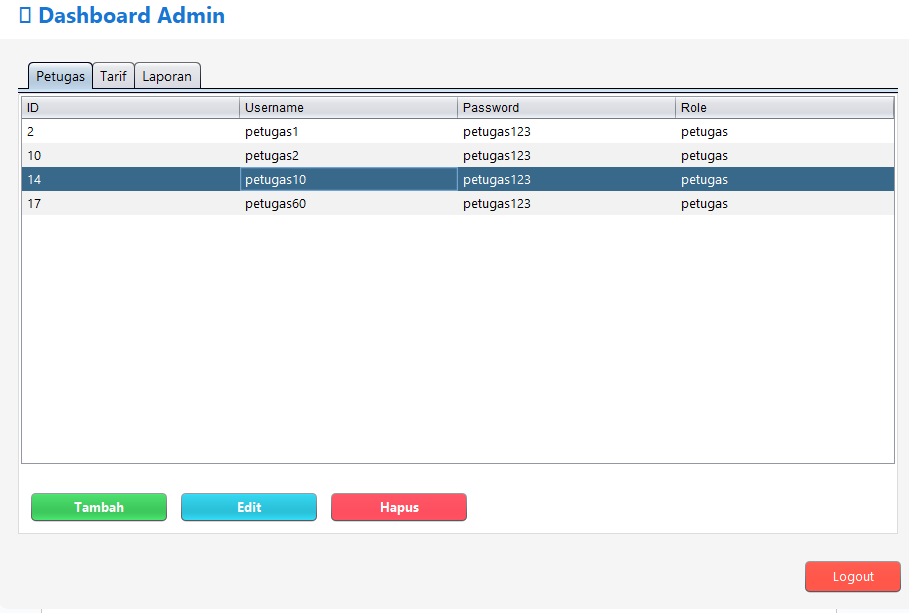




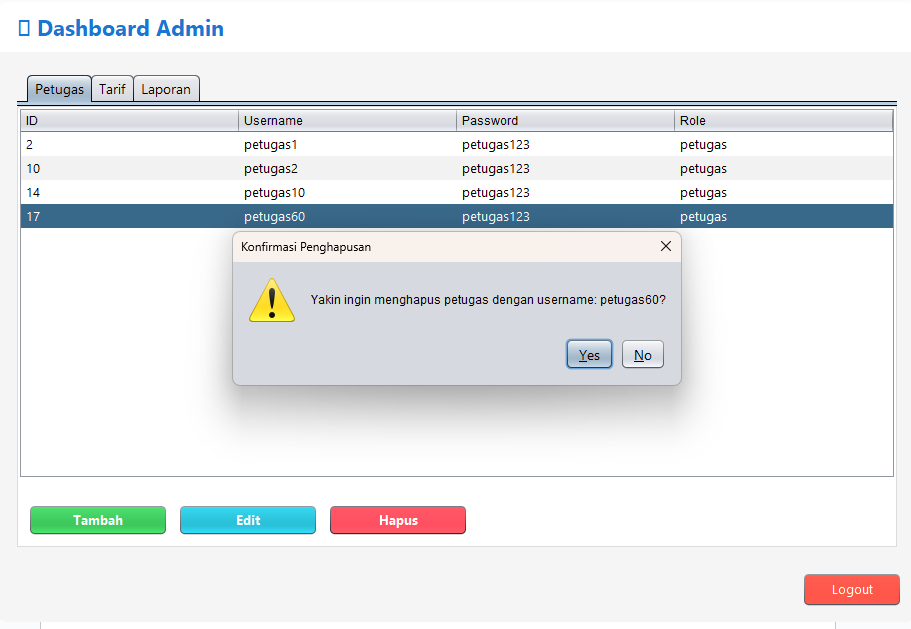
****

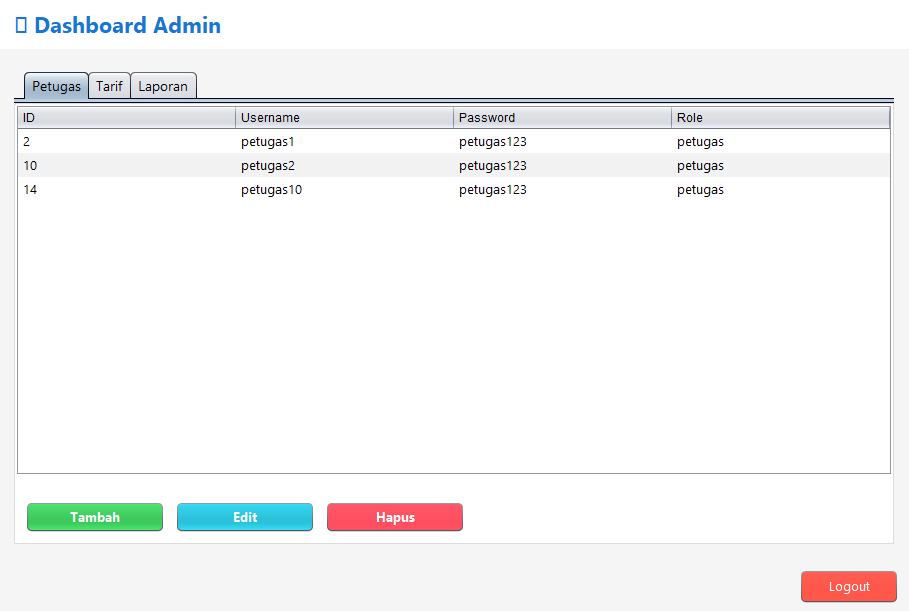
Admin dapat menambahkan petugas dengan cara memasukkan username dan password yang diinginkan, kemudian mengklik tombol "Simpan". Setelah itu, data petugas yang baru akan langsung ditampilkan pada tabel petugas.

****

****

Admin dapat mengedit data petugas yang diinginkan, seperti username dan password. Sebagai contoh, di sini kita mengganti username dari petugas20 menjadi petugas60. Setelah mengklik tombol "Simpan", perubahan tersebut akan langsung ditampilkan pada tabel petugas.

****

****

Admin dapat menghapus data petugas sesuai keinginan. Cukup pilih data petugas yang ingin dihapus, kemudian akan muncul pesan konfirmasi penghapusan. Setelah mengklik "Yes", data tersebut akan langsung terhapus dari tabel petugas.

.

# 

# 

# KESIMPULAN

Aplikasi ParkirKu berhasil dibuat untuk membantu pengelolaan parkir secara digital. Dengan tampilan berbasis Java Swing dan penyimpanan data di MySQL, aplikasi ini memudahkan petugas dalam mencatat kendaraan masuk dan keluar, serta menghitung tarif parkir secara otomatis.

Sistem ini menggunakan metode pemrograman berorientasi objek (OOP) dan mengikuti pola MVC, sehingga kode lebih rapi dan mudah dikembangkan. Terdapat dua jenis pengguna, yaitu Admin dan Petugas, yang masing-masing memiliki fitur sesuai tugasnya.

Dari hasil pengujian, semua fitur seperti login, input kendaraan, perhitungan tarif, riwayat transaksi, dan manajemen petugas dapat berjalan dengan baik. Sistem juga mampu menangani validasi input dan menampilkan data secara real-time.

Secara keseluruhan, aplikasi ParkirKu layak digunakan untuk mengelola parkir di berbagai tempat dan masih bisa dikembangkan lebih lanjut sesuai kebutuhan.