

**UJIAN TENGAH SEMESTER (UTS)  
GANJIL 2025/2026**

**Mata Kuliah:**  
PEMROGRAMAN BERORIENTASI OBJEK

Kelas D3TI2.C

**Studi Kasus:** Aplikasi Parkiran Berbasis Console



**Oleh:**  
Fadilah Putri Syahfika  
2403049

**D3 TEKNIK INFORMATIKA  
JURUSAN TEKNIK INFORMATIKA  
POLITEKNIK NEGERI INDRAMAYU  
OKTOBER 2025**

## Contents

<b>Deskripsi Permasalahan .....</b>	<b>1</b>
<b>Analisis Kebutuhan .....</b>	<b>1</b>
<b>Analisis Fitur.....</b>	<b>1</b>
<b>Rancangan UML Use Case Diagram.....</b>	<b>2</b>
<b>Rancangan Tampilan Program Aplikasi Console .....</b>	<b>3</b>
Tampilan Menu Login .....	3
Tampilan Menu Utama .....	3
Tampilan Masuk Kendaraan      Tampilan Masuk Kendaraan .....	3
(Tambah Data)      (Lihat Data).....	3
Tampilan Keluar Kendaraan      Tampilan Keluar Kendaraan.....	4
(Tambah Data)      (Lihat Data).....	4
Tampilan Lihat Ketersediaan Slot .....	4
Tampilan Cari Tiket.....	4
Tampilan Laporan Harian .....	5
<b>Analisis Class .....</b>	<b>5</b>
Identifikasi Class dan Attribute .....	5
Identifikasi Method .....	6
Jenis Relasi dan Alasan.....	7
Multiplicity .....	9
<b>Rancangan UML Class Diagram.....</b>	<b>11</b>
<b>Rancangan UML Sequence Diagram.....</b>	<b>12</b>
Sequence Diagram: melakukan Login.....	12
Sequence Diagram: Kendaraan Masuk .....	12
Sequence Diagram: Kendaraan Keluar.....	13
Sequence Diagram: Lihat Slot .....	13
Sequence Diagram: Cari Tiket .....	14
Sequence Diagram: Laporan Harian .....	14
Sequence Diagram: Keluar.....	15
<b>Pra-Kode Program.....</b>	<b>16</b>
Kode Program.....	16
Main.java.....	16
AplikasiParkir.java .....	17
ParkirService.java .....	20
Petugas.java .....	22

Pengguna.java .....	23
Mobil.java.....	24
Motor.java.....	25
TempatParkir.java.....	25
TiketParkir.java.....	26
Tarif.java .....	27
LaporanHarian.java.....	27
<b>Compile &amp; Run</b> .....	29
Compile .....	29
Run .....	29
<b>Testing</b> .....	29
Skenario Login Tidak Valid (Berhasil) .....	29
Skenario Login Valid (Berhasil).....	29
Skenario Pilih Menu Tidak Valid (Berhasil) .....	29
Kendaraan Masuk (Valid) .....	30
Kendaraan Keluar (Valid) .....	30
Kendaraan Keluar (Tidak Valid).....	30
Lihat Slot Parkir (Valid) .....	30
Cari Tiket (Valid).....	30
Cari Tiket (Tidak Valid) .....	31
Laporan Harian (Valid).....	31
Keluar (Valid) .....	31
<b>Build (Deploy)</b> .....	31
<b>URL Repository</b> .....	31

## Deskripsi Permasalahan

Area parkir kecil (kampus/rumah sakit/pertokoan) sering menghitung biaya secara manual. Ini rawan salah hitung (terutama saat jam sibuk), sulit menelusuri riwayat tiket, dan tidak ada kontrol ketersediaan slot. Diperlukan aplikasi **console** sederhana yang dapat: mencatat kendaraan masuk/keluar, memilih/menandai slot, menghitung biaya berbasis durasi & jenis kendaraan, serta menampilkan laporan ringkas harian.

## Analisis Kebutuhan

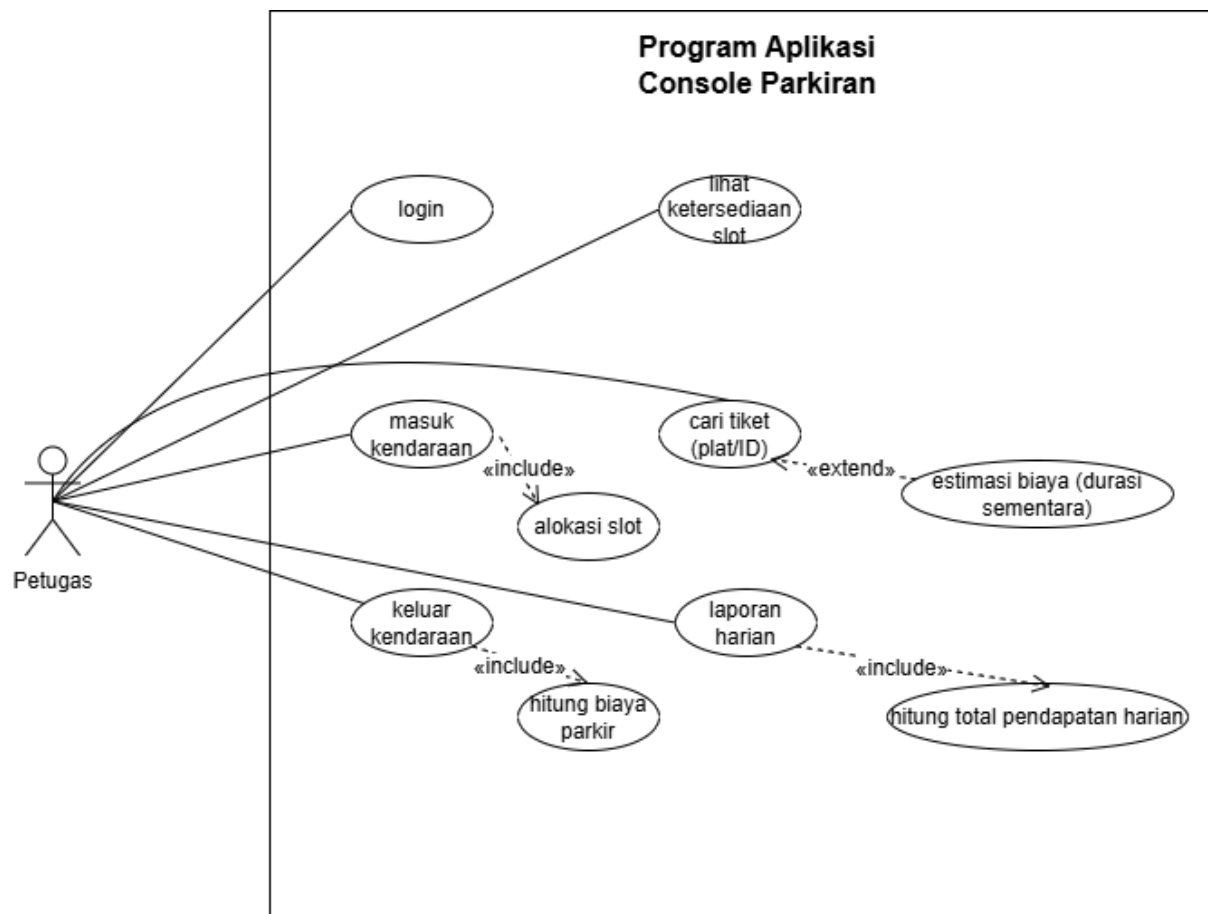
Program dirancang untuk:

1. Menyediakan pencatatan kendaraan masuk dengan input plat dan jenis (mobil/motor).
2. Memungkinkan petugas melakukan pencatatan kendaraan keluar dengan perhitungan biaya otomatis.
3. Menyediakan monitoring slot parkir (jumlah slot, terisi, kosong).
4. Memungkinkan petugas mencari tiket berdasarkan plat/ID.
5. Menghasilkan laporan harian (jumlah kendaraan keluar, total pendapatan).

## Analisis Fitur

No	Fitur	Kebutuhan	Alur
1	Masuk Kendaraan	Catat data masuk	Input plat & jenis → sistem cari slot kosong → buat tiket baru → slot ditandai terisi
2	Keluar Kendaraan	Tutup tiket & hitung biaya	Cari tiket → input keluar → hitung durasi & biaya → slot dikosongkan
3	Lihat Slot	Monitoring kapasitas	Sistem tampilkan jumlah slot kosong & terisi
4	Cari Tiket	Cek detail tiket aktif	Input plat/ID → tampilkan detail tiket (slot, masuk, estimasi biaya)
5	Laporan Harian	Rekap operasional	Sistem hitung jumlah kendaraan & total pendapatan pada hari tertentu

## Rancangan UML Use Case Diagram



## Rancangan Tampilan Program Aplikasi Console

### Tampilan Menu Login

Login (Berhasil):

```
Login Aplikasi Parkiran
Username: admin
Password: ****
Login berhasil!
```

Login (Gagal):

```
Login Aplikasi Parkiran
Username: admin
Password: ****
Username atau password salah. Coba lagi.
```

### Tampilan Menu Utama

```
Menu Aplikasi Parkiran
1. Masuk Kendaraan
2. Keluar Kendaraan
3. Lihat Ketersediaan Slot
4. Cari Tiket (Plat/ID)
5. Laporan Harian
6. Keluar

Pilih menu: _
```

### Tampilan Masuk Kendaraan

(Tambah Data)

```
Menu Aplikasi Parkiran
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot Parkir
4. Cari Tiket
5. Laporan Harian
6. Keluar

Pilih menu: 1
Opsi: 1. Tambah Data 2. Lihat Data
1
ID Tiket : T001
Plat Nomor : B1234XYZ
Slot : S1
Waktu Masuk : 08:30

Tambah data Kendaraan Masuk:
ID=T001, Plat=B1234XYZ, Slot=S1,
Waktu=08:30
```

### Tampilan Masuk Kendaraan

(Lihat Data)

```
Menu Aplikasi Parkiran
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot Parkir
4. Cari Tiket
5. Laporan Harian
6. Keluar

Pilih menu: 1
Opsi: 1. Tambah Data 2. Lihat Data
2
Data Kendaraan Masuk:
ID=T001, Plat=B1234XYZ, Slot=S1,
Waktu=08:30
ID=T002, Plat=D5678ABC, Slot=S2,
Waktu=09:00
```

## Tampilan Keluar Kendaraan

### (Tambah Data)

<p>Menu Aplikasi Parkiran</p> <ol style="list-style-type: none"><li>1. Kendaraan Masuk</li><li>2. Kendaraan Keluar</li><li>3. Lihat Slot Parkir</li><li>4. Cari Tiket</li><li>5. Laporan Harian</li><li>6. Keluar</li></ol> <p>Pilih menu: 2 Opsi: 1. Tambah Data 2. Lihat Data 1 ID Tiket : T001 Durasi : 3 jam Biaya Parkir : Rp 15.000 Bayar : Rp 20.000 Kembalian : Rp 5.000</p> <p>Tambah Data Kendaraan Keluar: ID=T001, Biaya=15000, Bayar=20000, Kembali=5000</p>	<p>Menu Aplikasi Parkiran</p> <ol style="list-style-type: none"><li>1. Kendaraan Masuk</li><li>2. Kendaraan Keluar</li><li>3. Lihat Slot Parkir</li><li>4. Cari Tiket</li><li>5. Laporan Harian</li><li>6. Keluar</li></ol> <p>Pilih menu: 2 Opsi: 1. Tambah Data 2. Lihat Data 2 Data Kendaraan Keluar: ID=T001, Plat=B1234XYZ, Slot=S1, Durasi=3 jam, Biaya=15000</p>
---	---

## Tampilan Keluar Kendaraan

### (Lihat Data)

## Tampilan Lihat Ketersediaan Slot

<p>Menu Aplikasi Parkiran</p> <ol style="list-style-type: none"><li>1. Kendaraan Masuk</li><li>2. Kendaraan Keluar</li><li>3. Lihat Slot Parkir</li><li>4. Cari Tiket</li><li>5. Laporan Harian</li><li>6. Keluar</li></ol> <p>Pilih menu: 3 Ketersediaan Slot Parkir Slot S1 : Terisi Slot S2 : Kosong Slot S3 : Kosong Slot S4 : Terisi Total Tersedia : 2 slot</p>
---

## Tampilan Cari Tiket

<p>Menu Aplikasi Parkiran</p> <ol style="list-style-type: none"><li>1. Kendaraan Masuk</li><li>2. Kendaraan Keluar</li><li>3. Lihat Slot Parkir</li><li>4. Cari Tiket</li></ol>	<p>Menu Aplikasi Parkiran</p> <ol style="list-style-type: none"><li>1. Kendaraan Masuk</li><li>2. Kendaraan Keluar</li><li>3. Lihat Slot Parkir</li><li>4. Cari Tiket</li></ol>
---	---

5. Laporan Harian 6. Keluar  Pilih menu: 4 Opsi: 1. Tambah Data 2. Lihat Data 1 Plat Nomor : B1234XYZ  Tambah Data Pencarian Tiket: Plat=B1234XYZ	5. Laporan Harian 6. Keluar  Pilih menu: 4 Opsi: 1. Tambah Data 2. Lihat Data 2 Hasil Pencarian Tiket: ID=T001, Plat=B1234XYZ, Slot=S1, Durasi=2 jam Estimasi Biaya: Rp 10.000
--	--

## Tampilan Laporan Harian

Menu Aplikasi Parkiran 1. Kendaraan Masuk 2. Kendaraan Keluar 3. Lihat Slot Parkir 4. Cari Tiket 5. Laporan Harian 6. Keluar  Pilih menu: 5  Laporan Harian Jumlah Kendaraan Masuk : 25 Jumlah Kendaraan Keluar : 20 Total Pendapatan Harian : Rp 500.000
--

## Analisis Class

### Identifikasi Class dan Attribute

Berdasarkan analisis kebutuhan sistem aplikasi console parkir, diperoleh identifikasi class dan atribut sebagai berikut:

No	Class (Kata Benda)	Attribute (Kata Benda)
1.	<b>Petugas</b>	id:String
		nama:String
		username:String
		password:String
2.	<b>Kendaraan (<i>superclass</i>)</b>	plat:String
		jamMasuk:DateTime
		jamKeluar:DateTime
3.	<b>Mobil (<i>extends Kendaraan</i>)</b>	jenis:String = "Mobil"
		tarifPerJam:int



4.	<b>Motor (extends Kendaraan)</b>	jenis:String = "Motor"
		tarifPerJam:int
5.	<b>SlotParkir</b>	idSlot:String
		status:String (kosong/terisi)
6.	<b>Tiket</b>	idTiket:String
		kendaraan:Kendaraan
		waktuMasuk:DateTime
		waktuKeluar:DateTime
		biaya:int
7.	<b>Transaksi</b>	idTransaksi:String
		tiket:Tiket
		petugas:Petugas
		total:int
8.	<b>Laporan</b>	idLaporan:String
		tanggal:Date
		totalPendapatan:int
		daftarTransaksi:ArrayList<Transaksi>
9.	<b>AplikasiParkir</b>	daftarKendaraan:ArrayList<Kendaraan>
		daftarSlot:ArrayList<SlotParkir>
		daftarTiket:ArrayList<Tiket>
		daftarPetugas:ArrayList<Petugas>
10.	<b>Main</b>	-

#### Identifikasi Method

Berikut adalah hasil identifikasi method berdasarkan daftar kebutuhan:

No	Class (Kata Benda)	Method (Kata Kerja)	Parameter (Kata Benda)
1	<b>Petugas</b>	login(): boolean	username:String, password:String
		tambahData(): void	scanner:Scanner
		lihatData(): void	-
2	<b>Kendaraan (superclass)</b>	setMasuk(): void	waktu:LocalDateTime
		setKeluar(): void	waktu:LocalDateTime
		tampilData(): void	-
3	<b>Mobil (extends Kendaraan)</b>	getTarifPerJam(): int	-
4	<b>Motor (extends Kendaraan)</b>	getTarifPerJam(): int	-
5	<b>TempatParkir</b>	tandaiTerisi(): void	-
		tandaiKosong(): void	-
6	<b>TiketParkir</b>	buatTiket(): TiketParkir	plat:String, jenis:String, slot:TempatParkir, masuk:LocalDateTime

		tutup(): void	keluar:LocalDateTime
		durasiMenit(): long	-
		hitungBiaya(): int	tarifPerJam:int
7	Tarif	setMobil(): void	n:int
		setMotor(): void	n:int
		ambilTarif(): int	jenis:String
8	AplikasiParkir	kelolaMasuk(): void	scanner:Scanner
		kelolaKeluar(): void	scanner:Scanner
		kelolaLihatSlot(): void	-
		kelolaCariTiket(): void	scanner:Scanner
		tampilLaporanHarian(): void	scanner:Scanner
9	LaporanHarian	generate(): void	riwayat:List<TiketParkir>
		tampilkan(): void	-
10	Main	main(): void	args:String[]

### Jenis Relasi dan Alasan

#### a. Dependency

No	Class Awal	Class Tujuan	Alasan
1	Main	AplikasiParkir	Main memakai AplikasiParkir untuk menjalankan program.
2	AplikasiParkir	Petugas	AplikasiParkir bergantung pada operasi login milik Petugas.
3	AplikasiParkir	Tarif	AplikasiParkir memanggil Tarif saat menghitung biaya.
4	AplikasiParkir	TempatParkir	AplikasiParkir bergantung pada slot parkir saat alokasi.

5	AplikasiParkir	Kendaraan	AplikasiParkir memakai data kendaraan saat proses masuk/keluar.
---	----------------	-----------	---

b. Unidirectional Association

No	Class Awal	Class Tujuan	Alasan
1	AplikasiParkir	Petugas	Aplikasi menyimpan petugasAktif. Petugas tidak tahu aplikasi.
2	AplikasiParkir	TempatParkir	Aplikasi menyimpan daftarSlot. Slot tidak tahu aplikasi.
3	AplikasiParkir	Tarif	Aplikasi menyimpan objek Tarif.
4	TiketParkir	Kendaraan	Tiket menyimpan kendaraan. Kendaraan tidak tahu tiket.
5	TiketParkir	TempatParkir	Tiket menyimpan slot. Slot tidak menyimpan semua tiket.

c. Bidirectional Association

No	Class Awal	Class Tujuan	Alasan
1	TempatParkir	TiketParkir	Slot tahu tiket aktif yang menempati, tiket juga tahu slotnya.

d. Aggregation

No	Class Awal	Class Tujuan	Alasan
1	AplikasiParkir	Petugas	Petugas bisa eksis di luar aplikasi, maka ini aggregation.

e. Composition

No	Class Awal	Class Tujuan	Alasan
1	AplikasiParkir	TempatParkir	Slot bagian dari aplikasi, siklus hidup mengikuti aplikasi.
2	AplikasiParkir	TiketParkir	Tiket aktif dikelola penuh oleh aplikasi, hilang jika aplikasi ditutup.
3	LaporanHarian	TiketParkir	Laporan harian terbentuk dari kumpulan tiket. Tanpa tiket laporan tidak ada.

f. Generalization

No	Class Awal	Class Tujuan	Alasan
1	Mobil	Kendaraan	Mobil adalah turunan dari Kendaraan.
2	Motor	Kendaraan	Motor juga turunan dari Kendaraan.

g. Realization

Tidak ada class yang mengimplementasikan interface pada studi kasus ini.

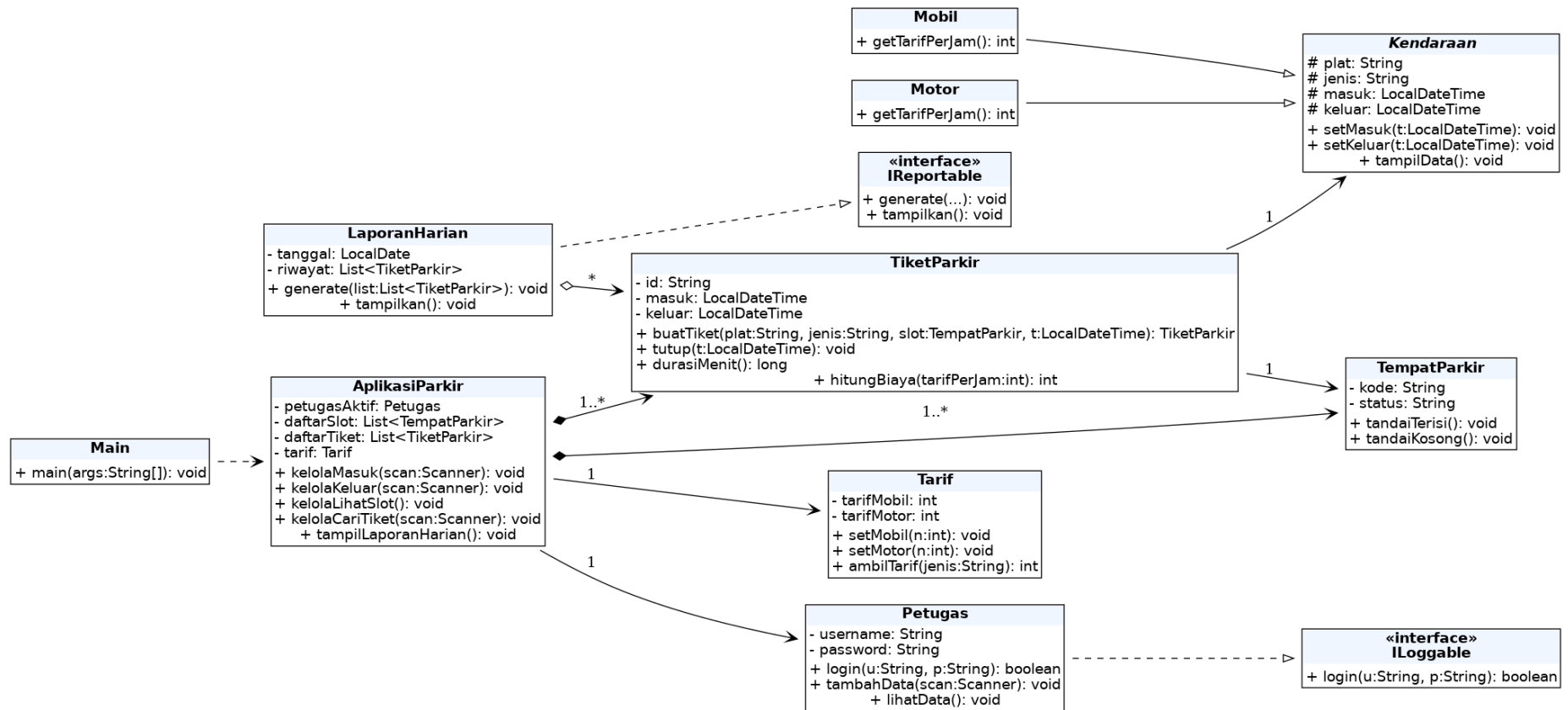
## Multiplicity

Berikut adalah penentuan kardinalitas antar class yang saling terhubung pada relasi association, aggregation, atau composition

No	Class Awal	Class Tujuan	Relasi	Multiplicity	Keterangan
1	TiketParkir	Petugas	Unidirectional Association	1 & 1	Satu TiketParkir dibuat oleh satu Petugas
2	TiketParkir	Kendaraan	Unidirectional Association	1 & 1	Satu TiketParkir hanya terkait dengan satu Kendaraan
3	Kendaraan	Mobil/Motor	Generalization	1 & 1	Mobil dan Motor merupakan turunan dari

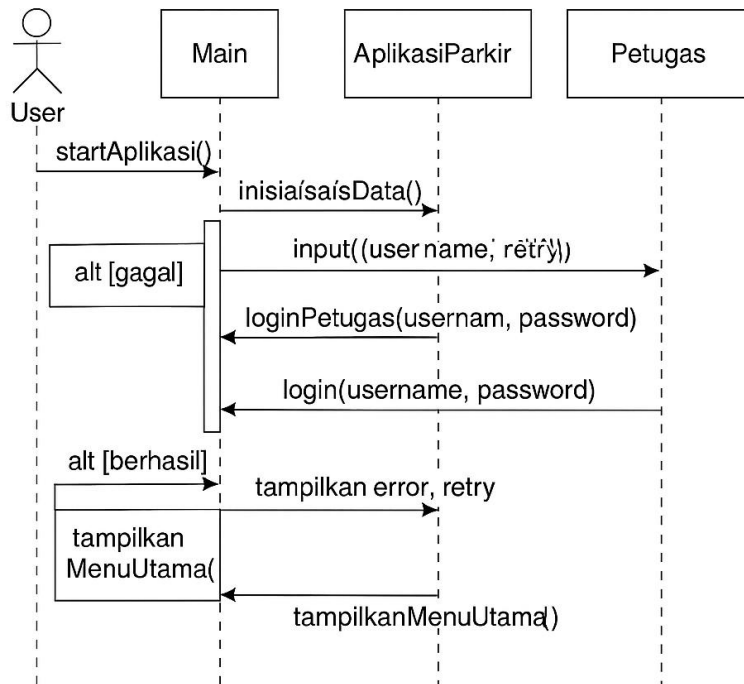
					class Kendaraan
4	TiketParkir	TempatParkir	Unidirectional Association	1 & 1	Satu TiketParkir terkait dengan satu slot TempatParkir
5	TempatParkir	AplikasiParkir	Composition	1..* & 1	Satu AplikasiParkir memiliki banyak TempatParkir, tapi slot tidak bisa eksis tanpa aplikasi
6	TiketParkir	Tarif	Dependency	1 & 1	Perhitungan biaya di TiketParkir bergantung pada Tarif
7	AplikasiParkir	TiketParkir	Composition	1..* & 1	Banyak TiketParkir dikelola dalam satu AplikasiParkir, tiket tidak bisa eksis tanpa aplikasi
8	LaporanHarian	TiketParkir	Aggregation	1..* & 1	LaporanHarian mengumpulkan banyak TiketParkir sebagai data, tapi tiket tetap bisa eksis meskipun laporan dihapus

## Rancangan UML Class Diagram

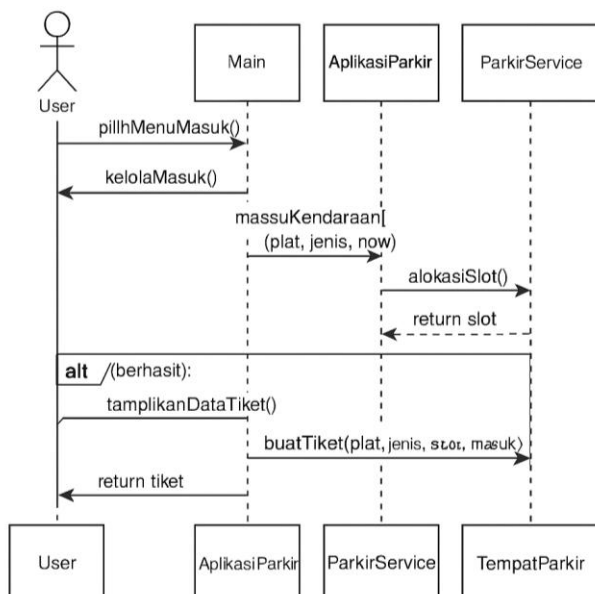


## Rancangan UML Sequence Diagram

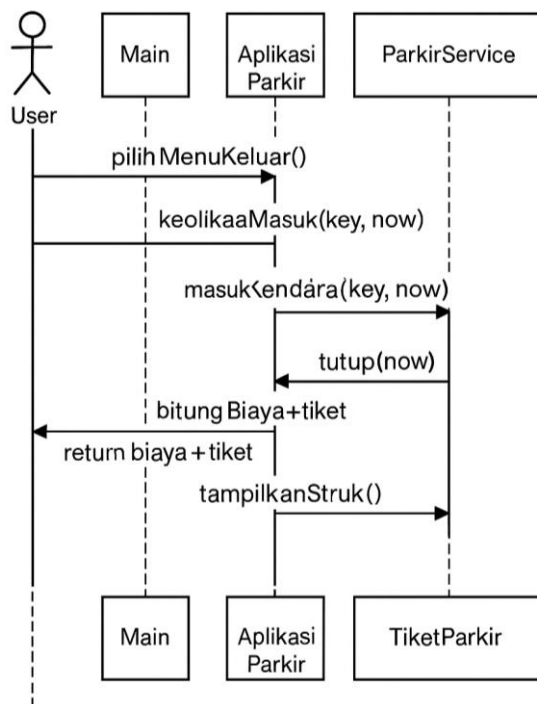
### Sequence Diagram: melakukan Login



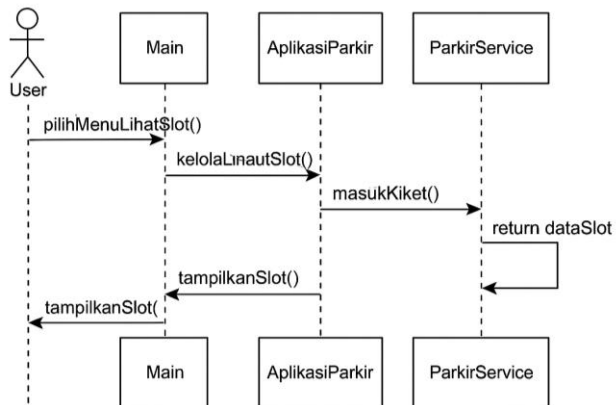
### Sequence Diagram: Kendaraan Masuk



## Sequence Diagram: Kendaraan Keluar

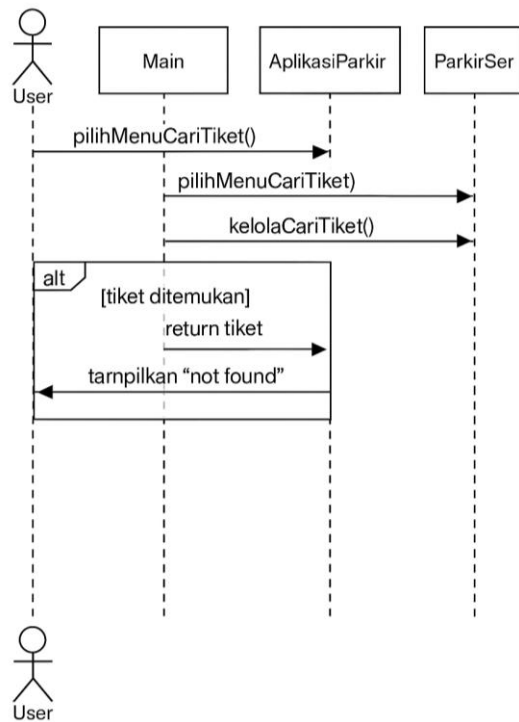


## Sequence Diagram: Lihat Slot

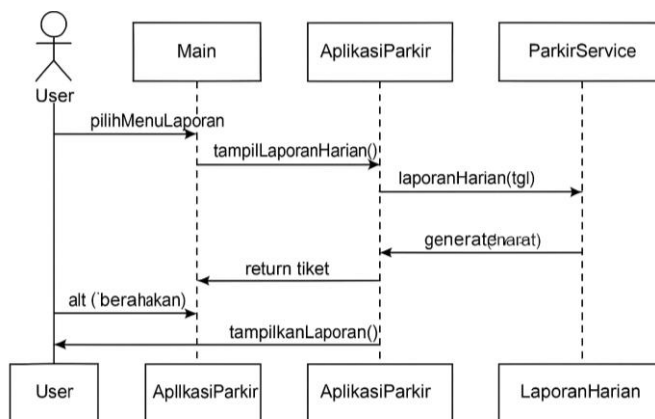




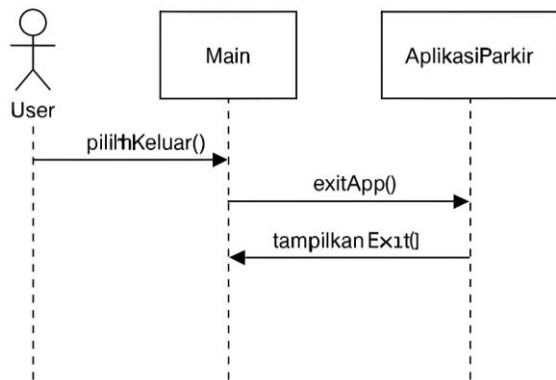
## Sequence Diagram: Cari Tiket



## Sequence Diagram: Laporan Harian



## Sequence Diagram: Keluar



## Pra-Code Program

This PC > Windows (C:) > pbo-project > aplikasi-parkir > parkir			
Sort View ...			
Name	Date modified	Type	Size
AplikasiParkir	25/10/2025 12:51	Java Source File	4 KB
Kendaraan	25/10/2025 12:52	Java Source File	1 KB
LaporanHarian	25/10/2025 12:53	Java Source File	1 KB
Main	25/10/2025 12:47	Java Source File	2 KB
Mobil	25/10/2025 12:52	Java Source File	1 KB
Motor	25/10/2025 12:52	Java Source File	1 KB
ParkirService	25/10/2025 12:51	Java Source File	3 KB
Pengguna	25/10/2025 12:52	Java Source File	1 KB
Petugas	25/10/2025 12:51	Java Source File	1 KB
Tarif	25/10/2025 12:53	Java Source File	1 KB
TempatParkir	25/10/2025 12:52	Java Source File	1 KB
TiketParkir	25/10/2025 12:52	Java Source File	2 KB

## Kode Program

### Main.java

Main
+ main(args:String[]): void

```
package parkir;

import java.util.Scanner;

public class Main {

    public static void main(String[] args) {

        System.out.println("=== Program Aplikasi Console Parkiran ===");

        Scanner scan = new Scanner(System.in);

        AplikasiParkir app = new AplikasiParkir(scan); // sudah seed data (petugas, slot, tiket contoh)

        // loop login

        while (!app.loginPetugas()) { /* ulang hingga benar */ }

        // loop menu 1–6

        while (true) {

            app.tampilkanMenuUtama();

            String pilih = scan.nextLine().trim();

            System.out.println();

            switch (pilih) {

                case "1": app.kelolaMasuk(); break;

            }

        }

    }

}
```

```

        case "2": app.kelolaKeluar(); break;

        case "3": app.kelolaLihatSlot(); break;

        case "4": app.kelolaCariTiket(); break;

        case "5": app.tampilLaporanHarian(); break;

        case "6": System.out.println("Bye!"); return;

        default : System.out.println("Pilihan tidak valid.\n");

    }

}

}

}

```

## AplikasiParkir.java

AplikasiParkir
- service: ParkirService - petugas: Petugas - scanner: Scanner
+ AplikasiParkir(service:ParkirService, petugas:Petugas, scanner:Scanner) + loginPetugas(username:String, password:String): boolean + kelolaMasuk(): void + kelolaKeluar(): void + kelolaLihatSlot(): void + kelolaCariTiket(): void + tampilLaporanHarian(): void + tampilkanDataTiket(t:TiketParkir): void + tampilkanStruk(t:TiketParkir, total:int): void

```

package parkir;

import java.time.LocalDate;

import java.time.LocalDateTime;

import java.util.Scanner;

public class AplikasiParkir {

    private final Scanner scan;

    private final ParkirService service;

    private Petugas petugasAktif;

    public AplikasiParkir(Scanner scan){

        this.scan = scan;

        this.service = new ParkirService(); // ParkirService sudah seed slot & tiket contoh

    }

    /* ===== LOGIN ===== */

    public boolean loginPetugas(){

        System.out.print("Username: ");

```

```

String u = scan.nextLine().trim();

System.out.print("Password: ");

String p = scan.nextLine().trim();

Petugas ptg = Petugas.login(u, p);

if (ptg == null) {

    System.out.println("Login gagal! Coba lagi.\n");

    return false;

}

petugasAktif = ptg;

System.out.println("Login berhasil. Selamat datang, " + ptg.getNama() + "!\n");

return true;

}

public void tampilkanMenuUtama(){

    System.out.println("Menu Utama:");

    System.out.println("1. Kendaraan Masuk");

    System.out.println("2. Kendaraan Keluar");

    System.out.println("3. Lihat Slot");

    System.out.println("4. Cari Tiket");

    System.out.println("5. Laporan Harian");

    System.out.println("6. Keluar");

    System.out.print("Pilih menu: ");

}

/* ===== 1. MASUK ===== */

public void kelolaMasuk(){

    System.out.print("Plat nomor    : ");

    String plat = scan.nextLine().trim();

    System.out.print("Jenis (Mobil/Motor): ");

    String jenis = scan.nextLine().trim();

    var t = service.masukKendaraan(plat, jenis, LocalDateTime.now());

    if (t == null) {

        System.out.println("Gagal masuk: Slot penuh!\n");

    } else {

        System.out.println("Tiket dibuat:");

        System.out.println(t + "\n");

    }

}

}

```

```

/* ===== 2. KELUAR ===== */

public void kelolaKeluar(){

    System.out.print("Masukkan ID tiket / PLAT: ");

    String key = scan.nextLine().trim();

    var target = service.cariTiket(key);

    if (target == null) {

        System.out.println("Tiket tidak ditemukan.\n");

        return;

    }

    var t = service.keluarKendaraan(target.getId(), LocalDateTime.now());

    System.out.println("Struk pembayaran:");

    System.out.println(t);

    System.out.println("Total bayar: Rp" + t.getBiaya() + "\n");

}

/* ===== 3. LIHAT SLOT ===== */

public void kelolaLihatSlot(){

    System.out.println("Status Slot:");

    service.getSlots().forEach(s ->

        System.out.println(s.getKode() + " : " + (s.isTerisi() ? "TERISI" : "KOSONG")));

    System.out.println();

}

/* ===== 4. CARI TIKET ===== */

public void kelolaCariTiket(){

    System.out.print("Cari (ID/PLAT): ");

    String k = scan.nextLine().trim();

    var t = service.cariTiket(k);

    if (t == null) System.out.println("NOT FOUND.\n");

    else System.out.println(t + "\n");

}

/* ===== 5. LAPORAN HARIAN ===== */

public void tampilLaporanHarian(){

    LocalDate tgl = LocalDate.now();

    LaporanHarian.cetakHarian(service.getRiwayat(), tgl);

}

```

```
}
```

## ParkirService.java

ParkirService
<ul style="list-style-type: none"><li>- slots: List&lt;TempatParkir&gt;</li><li>- tiketByKey: Map&lt;String, TiketParkir&gt;</li><li>- riwayatSelesai: List&lt;TiketParkir&gt;</li><li>- tarif: Tarif</li></ul>
<ul style="list-style-type: none"><li>+ ParkirService(jumlahSlot:int)</li><li>+ alokasiSlot(): TempatParkir</li><li>+ masukKendaraan(plat:String, jenis:String, now:LocalDateTime): TiketParkir</li><li>+ cariTiket(key:String): TiketParkir</li><li>+ keluarKendaraan(key:String, now:LocalDateTime): TiketParkir</li><li>+ lihatSlot(): List&lt;TempatParkir&gt;</li><li>+ laporanHarian(tgl:LocalDate): LaporanHarian</li><li>+ getTarif(): Tarif</li></ul>

```
package parkir;

import java.time.LocalDateTime;
import java.util.*;

public class ParkirService {

    private final List<TempatParkir> slots = new ArrayList<>();

    private final Map<String, TiketParkir> aktif = new LinkedHashMap<>();

    private final List<TiketParkir> riwayat = new ArrayList<>();

    private int seqTiket = 1;

    public ParkirService(){

        // ===== SEED SLOT (S1..S6) =====

        for (int i = 1; i <= 6; i++) slots.add(new TempatParkir("S" + i));

        // ===== SEED kendaraan MASUK (biar menu 2 & 4 bisa langsung dicoba) =====

        masukSeed("B1234CD", "Mobil", LocalDateTime.now().minusHours(3)); // T-0001

        masukSeed("D7777ZZ", "Motor", LocalDateTime.now().minusMinutes(20)); // T-0002

    }

    private void masukSeed(String plat, String jenis, LocalDateTime waktu){

        TempatParkir s = alokasiSlot();

        if (s == null) return;

        Kendaraan k = "Mobil".equalsIgnoreCase(jenis) ? new Mobil(plat) : new Motor(plat);

        String id = nextId();

        s.tandaiTerisi();

        TiketParkir t = new TiketParkir(id, k, s, waktu);

        aktif.put(id, t);

    }

}
```

```

        // juga bisa dicari pakai plat (opsional, sudah difasilitasi di cariTiket)
    }

    private String nextId(){ return String.format("T-%04d", seqTiket++); }

    public TempatParkir alokasiSlot(){
        for (TempatParkir s : slots) {
            if (!s.isTerisi()) return s;
        }
        return null; // penuh
    }

    /* ===== 1. MASUK ===== */
    public TiketParkir masukKendaraan(String plat, String jenis, LocalDateTime now){
        TempatParkir s = alokasiSlot();
        if (s == null) return null;
        Kendaraan k = "Mobil".equalsIgnoreCase(jenis) ? new Mobil(plat) : new Motor(plat);
        String id = nextId();
        s.tandaiTerisi();
        TiketParkir t = new TiketParkir(id, k, s, now);
        aktif.put(id, t);
        return t;
    }

    /* ===== 2. KELUAR ===== */
    public TiketParkir keluarKendaraan(String tiketId, LocalDateTime now){
        TiketParkir t = aktif.remove(tiketId);
        if (t == null) return null;
        t.tutup(now);
        riwayat.add(t);
        return t;
    }

    /* dipakai fitur 4 & 2 */
    public TiketParkir cariTiket(String key){
        if (aktif.containsKey(key)) return aktif.get(key);
        for (TiketParkir t : aktif.values()) {
            if (t.getKendaraan().getPlat().equalsIgnoreCase(key)) return t;
        }
    }

```



```

    }

    return null;
}

public List<TempatParkir> getSlots(){ return slots; }

public Collection<TiketParkir> getTiketAktif(){ return aktif.values(); }

public List<TiketParkir> getRiwayat(){ return riwayat; }
}

```

## Petugas.java

Petugas
- username: String - password: String
+ Petugas(username:String, password:String) + login(username:String, password:String): boolean + lihatData(): void

```

package parkir;

import java.util.*;

public class Petugas {

    private final String id;

    private final String nama;

    private final String username;

    private final String password;

    // ===== SEED USER =====

    private static final List<Petugas> DATA = new ArrayList<>();

    static {

        DATA.add(new Petugas("P1", "Admin", "admin", "pass"));

        DATA.add(new Petugas("P2", "Petugas 1", "petugas1", "1234"));

        DATA.add(new Petugas("P3", "Papad", "papad", "123"));

    }

    public Petugas(String id, String nama, String username, String password){

        this.id = id; this.nama = nama; this.username = username; this.password = password;

    }
}

```

```

public String getId(){ return id; }

public String getNama(){ return nama; }

public String getUsername(){ return username; }

public String getPassword(){ return password; }


public static Petugas login(String u, String p){

    String uu = u == null ? "" : u.trim();

    String pp = p == null ? "" : p.trim();

    for (Petugas x : DATA){

        if (x.username.equals(uu) && x.password.equals(pp)) return x;

    }

    return null;

}
}

```

## Pengguna.java

«interface» Pengguna
+ login(username:String, password:String): boolean

```

package parkir;

public abstract class Pengguna {

    private final String id;

    private final String nama;

    public Pengguna(String id, String nama) {

        this.id = id;

        this.nama = nama;

    }

    public String getId() { return id; }

    public String getNama() { return nama; }

}

```

## Kendaraan.java

Kendaraan
# plat: String # jenis: String # masuk: LocalDateTime # keluar: LocalDateTime
# Kendaraan(plat:String, jenis:String) + setMasuk(t:LocalDateTime): void + setKeluar(t:LocalDateTime): void + getTarifPerJam(): int + getPlat(): String + getJenis(): String + getMasuk(): LocalDateTime + getKeluar(): LocalDateTime + tampilData(): void

```
package parkir;

public abstract class Kendaraan {

    protected String plat;

    protected String jenis; // Mobil / Motor

    public Kendaraan(String plat, String jenis){

        this.plat = plat;

        this.jenis = jenis;

    }

    public String getPlat(){ return plat; }

    public String getJenis(){ return jenis; }

    public abstract int getTarifPerJam();

}
```

## Mobil.java

Mobil
+ Mobil(plat:String) + getTarifPerJam(): int

```
package parkir;

public class Mobil extends Kendaraan {

    public Mobil(String plat){ super(plat, "Mobil"); }

    @Override public int getTarifPerJam(){ return Tarif.getTarifMobil(); }

}
```

## Motor.java

Motor
+ Motor(plat:String) + getTarifPerJam(): int

```
package parkir;

public class Motor extends Kendaraan {

    public Motor(String plat){ super(plat, "Motor"); }

    @Override public int getTarifPerJam(){ return Tarif.getTarifMotor(); }

}
```

## TempatParkir.java

TempatParkir
- id: String - terisi: boolean
+ TempatParkir(id:String) + tandaiTerisi(): void + tandaiKosong(): void + isTerisi(): boolean + getId(): String + toString(): String

```
package parkir;

public class TempatParkir {

    private final String kode; // S1, S2, ...

    private boolean terisi;

    public TempatParkir(String kode){

        this.kode = kode;

        this.terisi = false;

    }

    public String getKode(){ return kode; }

    public boolean isTerisi(){ return terisi; }

    public void tandaiTerisi(){ terisi = true; }

    public void tandaiKosong(){ terisi = false; }

}
```

## TiketParkir.java

TiketParkir
- id: String - plat: String - jenis: String - slot: TempatParkir - masuk: LocalDateTime - keluar: LocalDateTime
+ TiketParkir(plat:String, jenis:String, slot:TempatParkir, masuk:LocalDateTime) + tutup(waktuKeluar:LocalDateTime): void + durasiMenit(): long + hitungBiaya(tarifPerJam:int): int + getId(): String + getPlat(): String + getJenis(): String + getSlot(): TempatParkir + getMasuk(): LocalDateTime + getKeluar(): LocalDateTime + toString(): String

```
package parkir;

import java.time.LocalDateTime;
import java.time.temporal.ChronoUnit;

public class TiketParkir {

    private final String id;          // T-0001, T-0002, ...

    private final Kendaraan kendaraan;

    private final TempatParkir slot;

    private final LocalDateTime masuk;

    private LocalDateTime keluar;

    private int biaya;

    public TiketParkir(String id, Kendaraan kendaraan, TempatParkir slot, LocalDateTime masuk){

        this.id = id;

        this.kendaraan = kendaraan;

        this.slot = slot;

        this.masuk = masuk;

    }

    public String getId(){ return id; }

    public Kendaraan getKendaraan(){ return kendaraan; }

    public TempatParkir getSlot(){ return slot; }

    public LocalDateTime getMasuk(){ return masuk; }

    public LocalDateTime getKeluar(){ return keluar; }

    public int getBiaya(){ return biaya; }

    public void tutup(LocalDateTime waktuKeluar){

        this.keluar = waktuKeluar;
```

```
        long menit = ChronoUnit.MINUTES.between(masuk, keluar);

        long jam = Math.max(1, (menit + 59) / 60); // bulat ke atas min 1 jam

        int tarif = kendaraan.getTarifPerJam();

        this.biaya = (int) (jam * tarif);

        slot.tandaiKosong();

    }

    @Override public String toString(){

        return "Tiket " + id + " | " + kendaraan.getJenis() + " " + kendaraan.getPlat() +

            " | Slot=" + slot.getKode() + " | Masuk=" + masuk +

            (keluar != null ? " | Keluar=" + keluar + " | Biaya=Rp" + biaya : "");

    }

}
```

## Tarif.java

Tarif
- tarifMobil: int - tarifMotor: int
+ setMobil(n:int): void + setMotor(n:int): void + ambilTarif(jenis:String): int

```
package parkir;

public class Tarif {

    // seed default

    private static int tarifMobil = 5000;

    private static int tarifMotor = 3000;

    public static int getTarifMobil(){ return tarifMobil; }

    public static int getTarifMotor(){ return tarifMotor; }

    public static int ambilTarif(String jenis){

        return "Mobil".equalsIgnoreCase(jenis) ? tarifMobil : tarifMotor;

    }

}
```

## LaporanHarian.java

LaporanHarian
- tanggal: LocalDate - tiketSelesai: List<TiketParkir>
+ LaporanHarian(tanggal:LocalDate, tiketSelesai:List<TiketParkir>) + totalPendapatan(tarif:Tarif): int + toString(): String

```
package parkir;
```

```
import java.time.LocalDate;
import java.util.List;

public class LaporanHarian {

    public static void cetakHarian(List<TiketParkir> riwayat, LocalDate tanggal){

        int total = 0;

        System.out.println("=== Laporan Harian " + tanggal + " ===");

        for (TiketParkir t : riwayat){

            if (t.getKeluar() != null && t.getKeluar().toLocalDate().equals(tanggal)){

                System.out.println(t);

                total += t.getBiaya();

            }

        }

        System.out.println("Total Pendapatan: Rp" + total + "\n");

    }

}
```

## Compile & Run

### Compile

```
PS C:\pbo-project\aplikasi-parkir> cd C:\pbo-project\aplikasi-parkir
>>
PS C:\pbo-project\aplikasi-parkir> javac parkir\*.java
```

### Run

```
PS C:\pbo-project\aplikasi-parkir> java parkir.Main
=== Program Aplikasi Console Parkiran ===
Username: papad
Password: 123
Login berhasil. Selamat datang, Papad!

Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
PS C:\pbo-project\aplikasi-parkir> |
```

## Testing

### Skenario Login Tidak Valid (Berhasil)

```
=== Program Aplikasi Console Parkiran ===
Username: papad
Password: 1234
Login gagal! Coba lagi.
Username: |
```

### Skenario Login Valid (Berhasil)

```
=== Program Aplikasi Console Parkiran ===
Username: papad
Password: 123
Login berhasil. Selamat datang, Papad!
```

### Skenario Pilih Menu Tidak Valid (Berhasil)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 7

Pilihan tidak valid.
```



## Kendaraan Masuk (Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 1

Plat nomor      : H1234AA
Jenis (Mobil/Motor): Mobil
Tiket dibuat:
Tiket T-0003 | Mobil H1234AA | Slot=S3 | Masuk=2025-10-25T15:07:45.217411500
```

## Kendaraan Keluar (Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 2

Masukkan ID tiket / PLAT: H1234AA
Struk pembayaran:
Tiket T-0003 | Mobil H1234AA | Slot=S3 | Masuk=2025-10-25T15:07:45.217411500 | Keluar=2025-10-25T15:
14:14.88657000 | Biaya=Rp5000
Total bayar: Rp5000
```

## Kendaraan Keluar (Tidak Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 2

Masukkan ID tiket / PLAT: GRY2343
Tiket tidak ditemukan.
```

## Lihat Slot Parkir (Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 3

Status Slot:
S1 : TERISI
S2 : TERISI
S3 : KOSONG
S4 : KOSONG
S5 : KOSONG
S6 : KOSONG
```

## Cari Tiket (Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 4

Cari (ID/PLAT): T-0001
Tiket T-0001 | Mobil B1234CD | Slot=S1 | Masuk=2025-10-25T11:58:48.989684500
```

## Cari Tiket (Tidak Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 4

Cari (ID/PLAT): HS21740
NOT FOUND.
```

## Laporan Harian (Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 5

=== Laporan Harian 2025-10-25 ===
Tiket: T-0000 | Mobil: H1234AA | Slot: S3 | Masuk: 2025-10-25T15:07:45, 217411500 | Keluar: 2025-10-25T15:
14:14, 886657600 | Biaya: Rp5000
Total Pendapatan: Rp5000
```

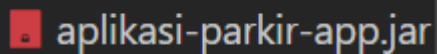
## Keluar (Valid)

```
Menu Utama:
1. Kendaraan Masuk
2. Kendaraan Keluar
3. Lihat Slot
4. Cari Tiket
5. Laporan Harian
6. Keluar
Pilih menu: 6

Bye!
```

## Build (Deploy)

```
PS C:\pbo-project\aplikasi-parkir> jar cfe aplikasi-parkir-app.jar parkir.Main parkir\*.class
>>
PS C:\pbo-project\aplikasi-parkir> java -jar aplikasi-parkir-app.jar
>>
=== Program Aplikasi Console Parkiran ===
Username: 
```



## URL Repository

<https://github.com/Fadilahppd24/parkiran.git>