1 Class Diagrams

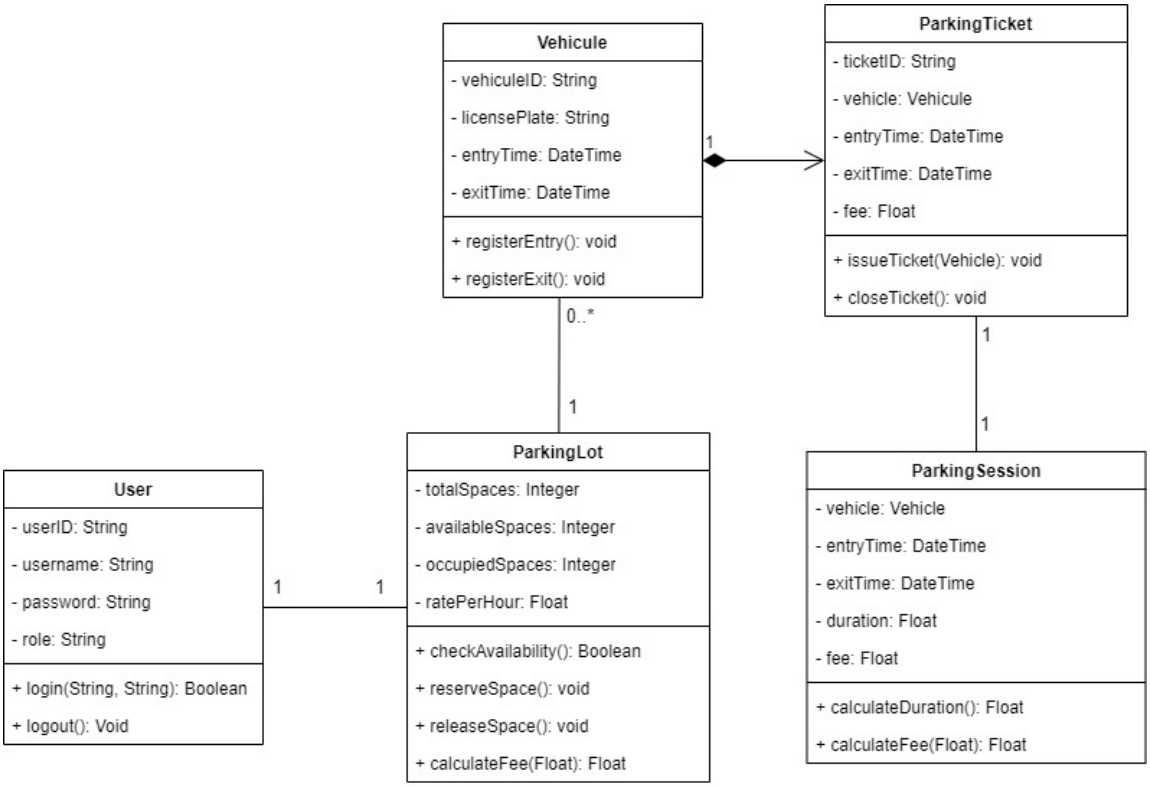

Figure 1: UML Class Diagram for the Parking Management System

This system consists of the following classes:

1. **User (Admin)**
2. **Vehicle**
3. **ParkingTicket**
4. **ParkingLot**
5. **ParkingSession**

Each class has certain attributes and methods, and the relationships between them are defined as follows.

1.1 Classes and Their Details

1.1.1 User Class

The **User** class represents the administrator who manages the parking lot operations.

Attributes

- `userID (String)`: A unique identifier for the user (e.g., "admin001").
- `username (String)`: The username used to log into the system.
- `password (String)`: The password for the user account.
- `role (String)`: The role of the user (typically "admin").

Methods

- `login(username: String, password: String): Boolean`: Verifies the login credentials. Returns `true` if the login is successful, otherwise returns `false`.
- `logout(): void`: Logs the user out of the system.

1.1.2 Vehicle Class

The **Vehicle** class represents a vehicle entering or exiting the parking lot.

Attributes

- `vehicleID (String)`: A unique identifier for the vehicle (e.g., "V001").
- `licensePlate (String)`: The license plate number of the vehicle.
- `entryTime (DateTime)`: The time when the vehicle enters the parking lot.
- `exitTime (DateTime)`: The time when the vehicle exits the parking lot.

Methods

- `registerEntry(): void`: Records the entry time of the vehicle.
- `registerExit(): void`: Records the exit time of the vehicle.

1.1.3 ParkingTicket Class

The **ParkingTicket** class represents a ticket issued when a vehicle enters the parking lot.

Attributes

- `ticketID (String)`: A unique identifier for the parking ticket (e.g., "T001").
- `vehicle (Vehicle)`: The vehicle associated with this ticket.
- `entryTime (DateTime)`: The entry time of the vehicle.
- `exitTime (DateTime)`: The exit time of the vehicle.
- `fee (Float)`: The parking fee based on the vehicle's duration in the lot.

Methods

- `issueTicket(vehicle: Vehicle): void`: Issues a parking ticket when a vehicle enters the parking lot.
- `closeTicket(): void`: Closes the ticket and calculates the parking fee based on the duration.

1.1.4 ParkingLot Class

The **ParkingLot** class represents the parking lot itself and manages parking spaces.

Attributes

- `totalSpaces (Integer)`: The total number of spaces in the parking lot.
- `availableSpaces (Integer)`: The number of available spaces in the parking lot.
- `occupiedSpaces (Integer)`: The number of spaces occupied by vehicles.
- `ratePerHour (Float)`: The parking rate per hour.

Methods

- `checkAvailability(): Boolean`: Returns `true` if there are available spaces, otherwise `false`.
- `reserveSpace(): void`: Reserves a space when a vehicle enters the parking lot.
- `releaseSpace(): void`: Releases a space when a vehicle exits the parking lot.
- `calculateFee(duration: Float): Float`: Calculates the parking fee based on the duration the vehicle stayed.

1.1.5 ParkingSession Class

The **ParkingSession** class tracks the duration and fee for a vehicle's parking session.

Attributes

- `vehicle (Vehicle)`: The vehicle associated with the parking session.
- `entryTime (DateTime)`: The entry time of the vehicle.
- `exitTime (DateTime)`: The exit time of the vehicle.
- `duration (Float)`: The duration in hours the vehicle was parked.
- `fee (Float)`: The parking fee for the session.

Methods

- `calculateDuration(): Float`: Calculates the duration of the vehicle's stay, in hours, between entry and exit times.
- `calculateFee(ratePerHour: Float): Float`: Calculates the fee based on the duration and rate per hour.

1.2 Relationships Between Classes

1.2.1 User $<->$ ParkingLot

The **User** class (admin) interacts with the **ParkingLot** class to view and manage the parking lot's statistics and vehicle entries and exits.

Relationship Type

- **Association**: The admin interacts with the parking lot but does not own it.
- **Multiplicity**: 1 `User` $<->$ 1 `ParkingLot` (one user can manage one parking lot).

1.2.2 Vehicle $<->$ ParkingTicket

The **Vehicle** class is associated with the **ParkingTicket** class, which tracks the vehicle's entry and exit times.

Relationship Type

- **Composition**: A parking ticket cannot exist without the vehicle it is assigned to.
- **Multiplicity**: 1 `Vehicle` $<->$ 1 `ParkingTicket` (each vehicle has exactly one parking ticket).

1.2.3 ParkingLot $<->$ Vehicle

The **ParkingLot** class manages multiple vehicles, and each vehicle occupies one parking space.

Relationship Type

- **Association**: The parking lot can contain multiple vehicles.
- **Multiplicity**: 1 `ParkingLot` $<->$ 0..* `Vehicle` (a parking lot can contain multiple vehicles, but a vehicle is in only one parking lot at a time).

1.2.4 ParkingTicket $<->$ ParkingSession

The **ParkingTicket** class is associated with the **ParkingSession** class, which tracks the duration and fee of the parking session.

Relationship Type

- **Association**: Each parking ticket has one associated parking session.
- **Multiplicity**: 1 `ParkingTicket` $<->$ 1 `ParkingSession` (one ticket has exactly one parking session).