UML Class Diagram (organized ver.)

1. User (Abstract)

- + userID : int
- + firstName : String
- + lastName : String
- + email : String
- + password : String
- + accountType : String
- -----------------------------
- + login(email, password) : boolean
- + logout() : void

2. Client

- inherits from User
- -----------------------------
- + registeredVehicles : List<Vehicle>
- + activeTickets : List<Ticket>
- + ticketHistory : List<Ticket>
- -----------------------------
- + registerVehicle(vehicle : Vehicle) : void
- + requestParking(slot : ParkingSlot) : Ticket
- + viewActiveTickets() : void
- + viewTicketHistory() : void

3. Admin

- inherits from User
- -----------------------------
- + managedSlots : List<ParkingSlot>
- -----------------------------
- + addParkingSlot(slot : ParkingSlot) : void

- + removeParkingSlot(slotID : int) : void
- + viewAllActiveTickets() : void
- + viewVehicleHistory(vehiclePlate : String) : void

4. Vehicle (Abstract)

- + plateNumber : String
- + brand : String
- + model : String
- + color : String
- -----------------------------
- + calculateFee(duration : int) : double

5. Car

- inherits from Vehicle
- -----------------------------
- + calculateFee(duration : int) : double

6. Motorcycle

- inherits from Vehicle
- -----------------------------
- + calculateFee(duration : int) : double

…………………………….etc for the other vehicle types

7. ParkingSlot

- + slotID : int
- + isOccupied : boolean

- + vehicle : Vehicle
- -----------------------------
- + assignVehicle(vehicle : Vehicle) : void
- + removeVehicle() : void

8. Ticket

- + ticketID : int
- + vehicle : Vehicle
- + slot : ParkingSlot
- + entryTime : LocalDateTime
- + exitTime : LocalDateTime
- + totalFee : double
- + isActive : boolean
- -----------------------------
- + closeTicket(exitTime : LocalDateTime) : void
- + calculateDuration() : int
- + generateFee() : double

9. ParkingSystem

- + users : List<User>
- + slots : List<ParkingSlot>
- + tickets : List<Ticket>
- -----------------------------
- + createAccount(user : User) : void
- + login(email : String, password : String) : User
- + registerVehicle(user : Client, vehicle : Vehicle) : void
- + issueTicket(vehicle : Vehicle, slot : ParkingSlot) : Ticket
- + endParking(ticketID : int) : void
- + getActiveTickets() : List<Ticket>
- + getTicketHistory() : List<Ticket>

10. Interface: Actionable

- <<interface>> Actionable
- -----------------------------
- + performAction() : void

Relationships

- User → Client, Admin (Inheritance)

- Client *has many* Vehicle

- Vehicle → Car, Motorcycle (Inheritance)

- ParkingSlot *can have one* Vehicle

- Ticket *links* Vehicle and ParkingSlot

- ParkingSystem *aggregates* User, ParkingSlot, and Ticket

- Admin *manages* ParkingSlot

- Client *creates* and *uses* Ticket