PARKINGLOT

```
class ParkingLot {
      List<ParkingFloor> parkingFloors;
      List<Entrance> entrances;
      List<Exit> exits;
      Address address;
      String parkingLotName;
      public boolean isParkingSpaceAvailableForVehicle(Vehicle vehicle);
      public boolean updateParkingAttndant(ParkingAttendant parkingAttendant,
int gateId)
}

class ParkingFloor {
      int levelId;
      List<ParkingSpace> parkingSpaces;
      ParkingDisplayBoard parkingDisplayBoard;
}

class Gate {
      int gateId;
      ParkingAttendant parkingAttendant;
}

class Entrance extends Gate {
      public ParkingTicket getParkingTicket(Vehicle vehicle);
}

class Exit extends Gate {
      public ParkingTicket payForParking(ParkingTicket parkingTicket,
PaymentType paymentType);
}

class Address {
      String country;
      String state;
      String city;
      String street;
      String pinCode; //ZipCode
}

class ParkingSpace {
      int spaceId;
      boolean isFree;
      double costPerHour;
      Vehicle vehicle;
      ParkingSpaceType parkingSpaceType;
}

class ParkingDisplayBoard {
      Map<ParkingSpaceType, Integer> freeSpotsAvailableMap;
      public void updateFreeSpotsAvailable(ParkingSpaceType parkingSpaceType,
int spaces);
}

class Account {
      String name;
      String email;
      String password;
      String empId;
      Address address;
}
```

```
class Admin extends Account {
	public boolean addParkingFloor(ParkingLot parkingLot, ParkingFloor floor);
	public boolean addParkingSpace(ParkingFloor floor, ParkingSpace 
parkingSpace);
	public boolean addParkingDisplayBoard(ParkingFloor floor, 
ParkingDisplayBoard parkingDisplayBoard);
	...
}

class ParkingAttendant extends Account {
	Payment paymentService;
	public boolean processVehicleEntry(Vehicle vehicle);
	public PaymentInfo processPayment(ParkingTicket parkingTicket, PaymentType 
paymentType);
}

class Vehicle {
	String licenseNumber;
	VehicleType vehicleType;
	ParkingTicket parkingTicket;
	PaymentInfo paymentInfo;
}

class ParkingTicket {
	int ticketId;
	int levelId;
	int spaceId;
	Date vehicleEntryDateTime;
	Date vehicleExitDateTime;
	ParkingSpaceType parkingSpaceType;
	double totalCost;
	ParkingTicketStatus parkingTicketStatus;
	public void updateTotalCost();
	public void updateVehicleExitTime(Date vehicleExitDateTime);
}

public enum PaymentType {
	CASH, CEDIT_CARD, DEBIT_CARD, UPI;
}

public enum ParkingSpaceType {
	BIKE_PARKING, CAR_PARKING, TRUCK_PARKING
}

class Payment {
	public PaymentInfo makePayment(ParkingTicket parkingTicket, PaymentType paymentType);
}

public class PaymentInfo {
	double amount;
	Date paymentDate;
	int transactionId;
	ParkingTicket parkingTicket;
	PaymentStatus paymentStatus;

}

public enum VehicleType {
	BIKE, CAR, TRUCK;
}

public enum ParkingTicketStatus {
	PAID, ACTIVE;
}

public enum PaymentStatus {
	UNPAID, PENDING, COMPLETED, DECLINED, CANCELLED, REFUNDED;
}
```