Design a Parking lot

To design a parking we will have following end users

1. Customer
2. Admin
3. Collector

Class : Customer

Data : TicketNumber,PaymentMode,amount,CardNumber,ExpiryDate, PaymentStatus

Behaviour :

```
collectTicketonEntrance() {
        get.ticketonEntrance(TicketNumber);
}
paymentonExit() {
        show.ticket(TicketNumber)
        if (PaymentMode = “Card”)
                get.carddetails(CardNumber,ExpiryDate);
        else
                get.cash();
        end-if
        update.payment.status(PaymentStatus);
}
```

****
****

Class : Login

Data : Role,Username,Password,Status

Behaviour :

```
loginSystem() {
        app.login.admin(Role,Username,Password,Status)
        app.login.collector(Role,Username,Password,Status)
}
```

****
****

Class : Admin

Data : Address, NoofFloors,NoofExitEntry, parkingspot, FloorNumber, Name, PhoneNumber, IDProof, LoginId, Role

Behaviour :

add.parkinglot(Address, NoofFloors, NoofExitEntry);

add.parkingfloor(Address, NoofFloors,NoofExitEntry,parkingspot);

add.parkingspot(FloorNumber, ParkingSpot);

add.EntryExit(Address, FloorNumber, NoofExitEntry);

add.collector() {

collectorDeatils(LoginId, Role, Name, PhoneNumber, IDProof);

}

remove.collector() {

remove.collector(LoginId, Name);

}

parking.rates() {

add.rates(VehicleType, Rate);

update.rates(VehicleType, Rate);

}

add.vehicletype () {

add.VehicleTye(Vehicle Type, Rate, FloorNumber,Address);

}

customer.details() {

add.customerDetails(TicketNumber, Amount, PaymentStatus);

update.customerDetails(TicketNumber, Amount, PaymentStatus);

}

****

****

Class : Collector

Data :

Behaviour : Role,UserName,Password,PaymentMode,PaymentStatus,TicketNumber

loginSystem()

scan.ticket(){

scan.ticketNumber(TicketNumber);

}

collect.payment() {

accept.payment(PaymentMode);

update.payment.status(PaymentStatus);

}

****

****

Class : Vehicle

Data : VehicleType, LicenseNumber

Behaviour :

assign.parking.lot(Address, VehicleType, LicenseNumber) ;

assign.ticket();

****

****

Class : ParkingLot

Data : Address, NoofFloor, NoofExitEntry,

Behaviour :

add.parkinglot() {

add.ParkingLotdetails(Address,NoofFloor,NoofExitEntry);

}

add.parkingfloor() {

add.ParkingFloor(NoofFloor,NoofExitEntry, Parkingspot);

}

add.EntryExit() {

add.ExitEntry(NoofExitEntry, issueTicket, scanticket, processpayment);

}

display.parkinglotdetails() {

display.parkinglotdetails(Address, NoofFloor, NoofExitEntry, Parkingspot);

}

isParkingSlotFull()

return true or false;

****

Class : ParkingFloor

Data : Address, NoofFloor, NoofExitEntry,ParkingSpot

Behaviour :

display.ParkingFloor() {

display.parkingfloor(NoofFloor,NoofExitEntry,ParkingSpots);

}

isFloorFull (){

return true or false;

}

display.ExitEntry(){

display.exitentry(FloorNumber);

}

****

Class : ParkingSpot

Data : FloorNumber,ParkingSpot, Status

Behaviour :

add.ParkingSpot(FloorNumber,ParkingSpot) {

add.parkingspot(“Bike”,”4-wheeler”,”HandicappedSpot”, “LargeSpot”);

}

display.ParkingSpot(){

display(FloorNumber,ParkingSpot,Status);

}

****

Class : ParkingDisplay

Data : Address, ParkingSpot, FloorNumber,NoofFloor,NoofExitEntry

Behaviour :

display.parkinglotdetails()

display.ParkingFloor()

display.ParkingSpot()

display.ExitExtry()

****
****

Class : Payment

Data : PaymentMode,PaymentStatus,cardnumber,expirydate

Behaviour :

```
accept.payment(PaymentMode) {
        if (PaymentMode = “Card” )
                get.card(cardnumber,expirydate);
        else
                get.cash();
        end-if
        update.payment.status(PaymentStatus);
}
```