AGENDA

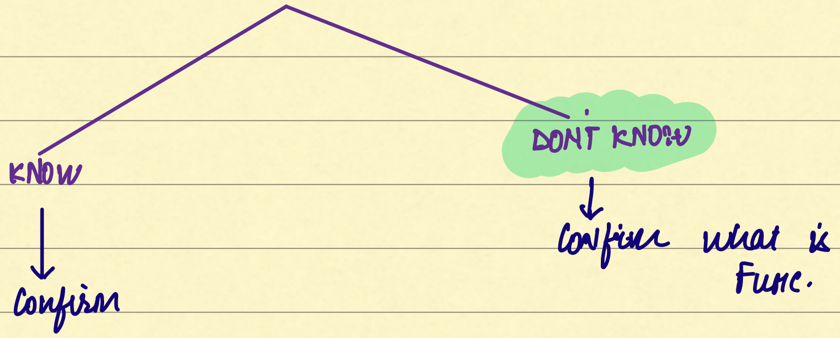
1·> Design Parking LOT → Management system

- Requirements
- class diagram

start by 9:05 PM IST

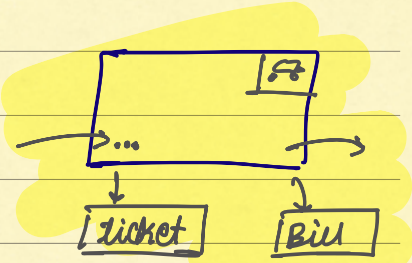
*] DESIGN PARKING LOT :

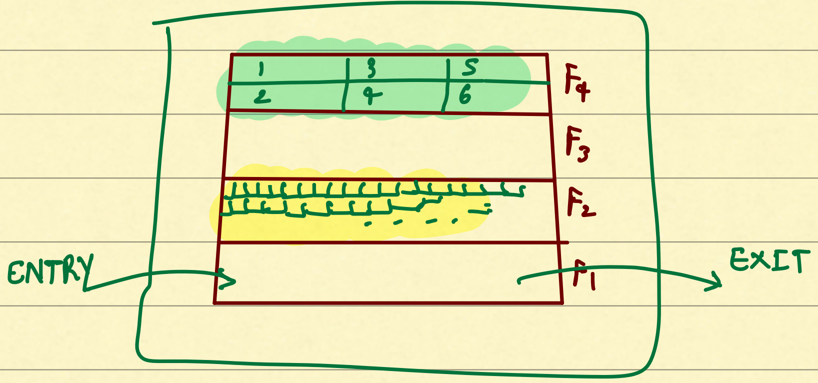
1:) OVERVIEW:



2:) REQ GATHERING:

suggest req. with Rational





1:) single/Multilevel Parking ✓
2:) Type of vehicles supported : 2W/4W/EV
3:) dedicated Parking spots for 2W/4W/EV

4·> Multiple Entry / Exit Gates

5·> Multiple Parking spots on each floor

·> availability

- Park anywhere
- Park at designated space (✓)

6·) Multiple ways to assin spot to vehicle
but use only (1) at one time

* 7·> Restricted spots

8·> diff Parking spot for diff vehicles

9·> Diff. Facilities – EV charge at spot

10·> ticket is Generated at Entry, Bill & Payment happens
at exit

11·> Operator is Present at Each Gate

12·> ways of Payment ⟶ online / cash

13·) system should store vehicle details

1·) Fees calculation }
2·) spot assignment }

*) **Fees calculation :**

- Flat fare calculation (B/C)
- Hourly fees calculation (B/C)

→ strategy dp ⤴ ["______"]

→ discuss fees for additional services...

*) **PAYMENT:**

- UPI
- CASH
- CC/DC

① → can part payments → 100/- 70 UPI + 30 CASH

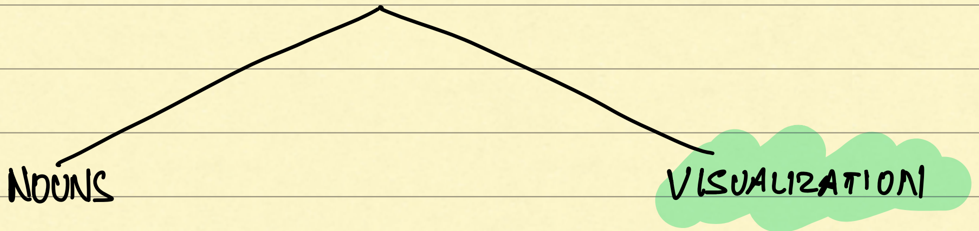
② Refunds

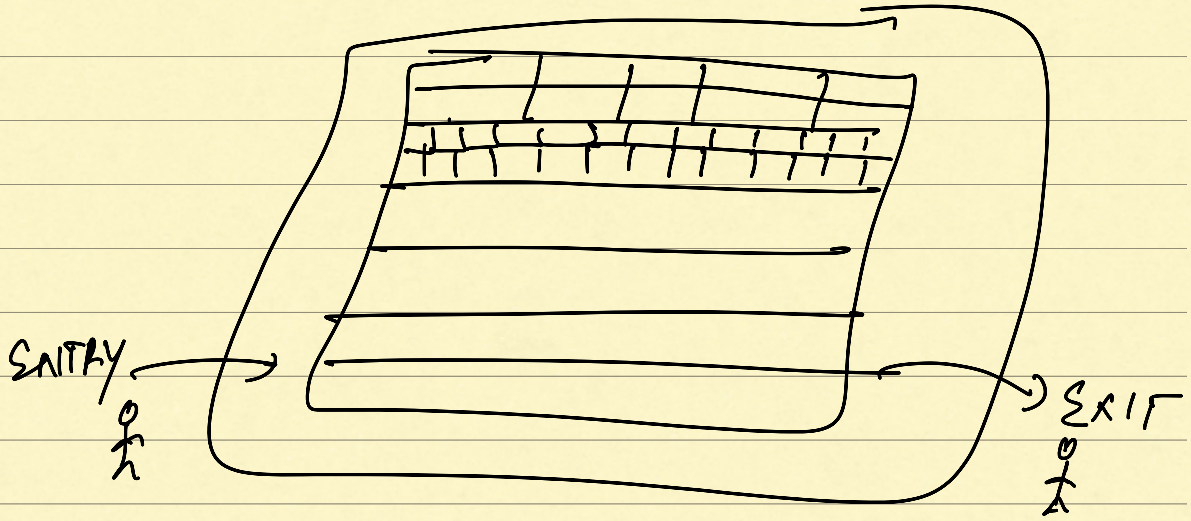
→ How fees is calculated:

FLAT		HOURLY	
car:	20	car:	50/- PH /...
bike	10	Bike:	30 PH /...

3.) use-case diagram ✓

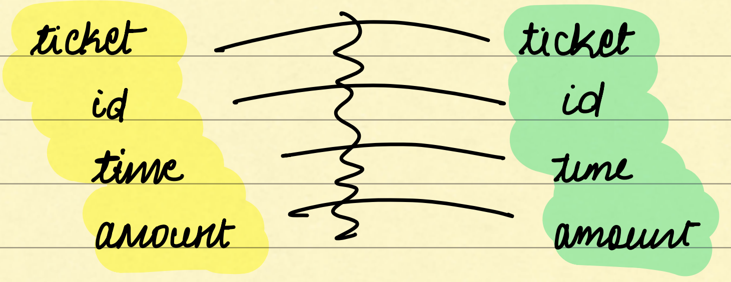
4.) class diagram:





In Real systems → database

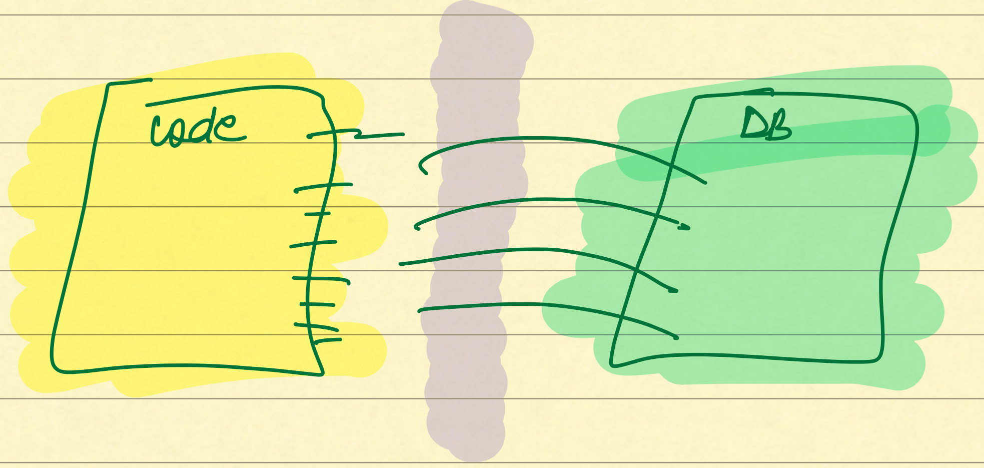
All classes → stored in DB



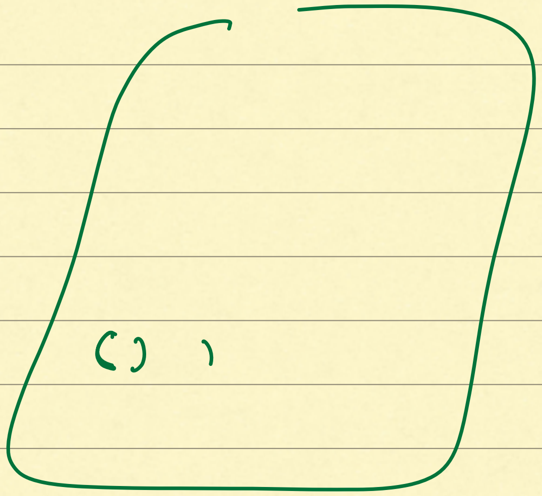
Generate() ⟿

(service layer)

Controller



(O.R.M)
Hibernate
Object Relational Mapping



db

① define classes ② attributes }

~~ParkingLOT~~ / ~~Floor~~ / ~~spot~~ /
~~vehicle~~ / ~~entry Gate~~
~~ticket~~ / ~~Bill~~ / ~~operator~~ / ~~Payment~~

classes ——— tables

entry Gate }
exit Gate } → Gates

① **Parking LOT**

- id
- address
- List<Floor>
- List<Gate>

② **ParkingFloor**

- id
- List<Parkingspot>
- floor_no

③ **Gate**

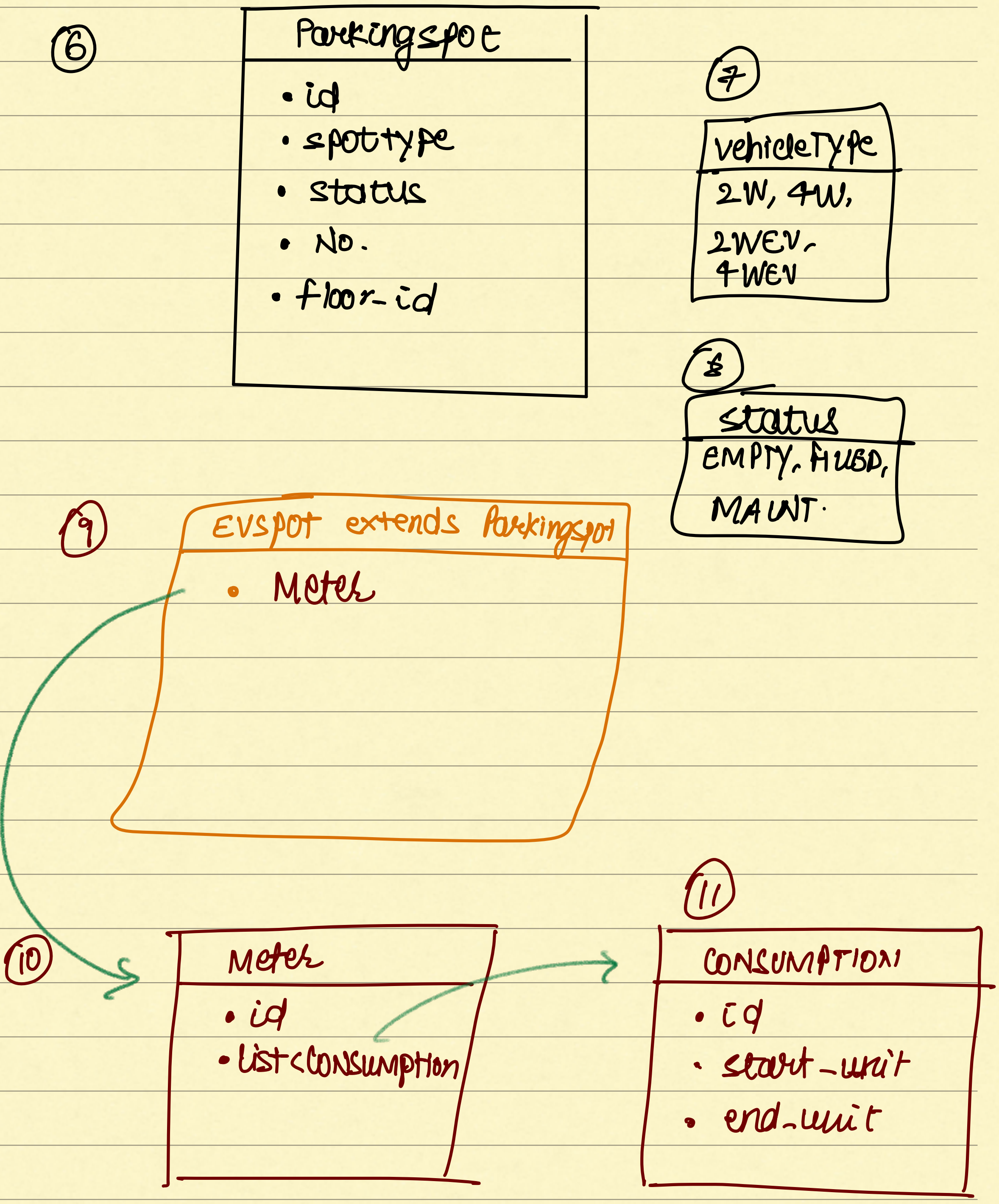
- id
- type
- operator
- status

④ **GateType**

ENTRY, EXIT

⑤ **Gatestatus**

OPEN,
CLOSED,
MAINT.


6
Parkingspot
• id
• spottype
• status
• No.
• floor-id
7
vehicleType
2W, 4W,
2WEV,
4WEV
8
status
EMPTY, FILLED,
MAINT.
9
EVspot extends Parkingspot
• Meter
10
Meter
• id
• List<Consumption
11
CONSUMPTION
• id
• start_unit
• end_unit

12

vehicle

- id
- Reg-No
- vehicle-type

13

Operator

- id
- Name

14

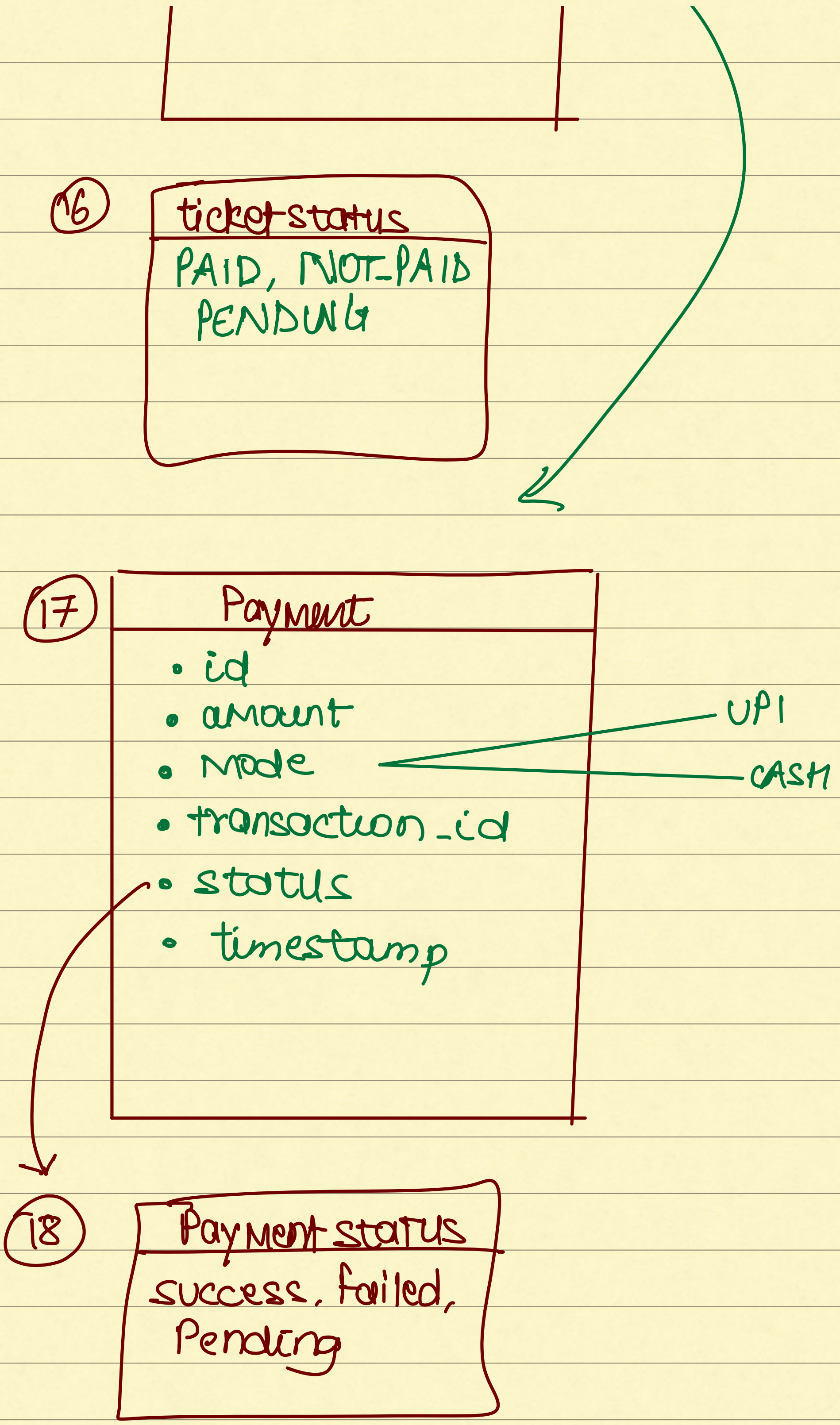
ticket

- id
- vehicle
- entry-time
- operator
- spot
- Gate

15

Bill

- id
- total-amount
- ticket
- exit_time
- Gate
- status
- List<Payment>



16

ticketstatus

PAID, NOT_PAID
PENDING

17

Payment

- id
- amount
- mode
- transaction_id
- status
- timestamp

18

Paymentstatus

success, failed,
Pending

*) DESIGN PATTERNS:

1.) STRATEGY ⊕ FACTORY

•) fees / allocation

2.) Builder for Parking lot

3.) decorator for fees calculation

Map< vehicleType, BasePrice >
Map< vehicleType, HourlyMultiplier>

HW

① complete schema design
② Revise MVC pattern