Design Book My Show

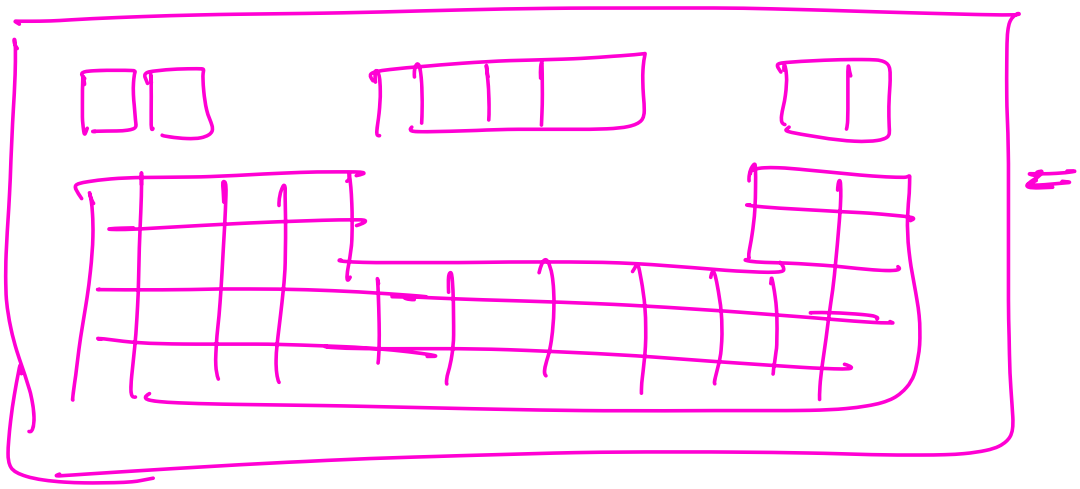
- → Overview
- → Gather Requirements
- → Class Diagram

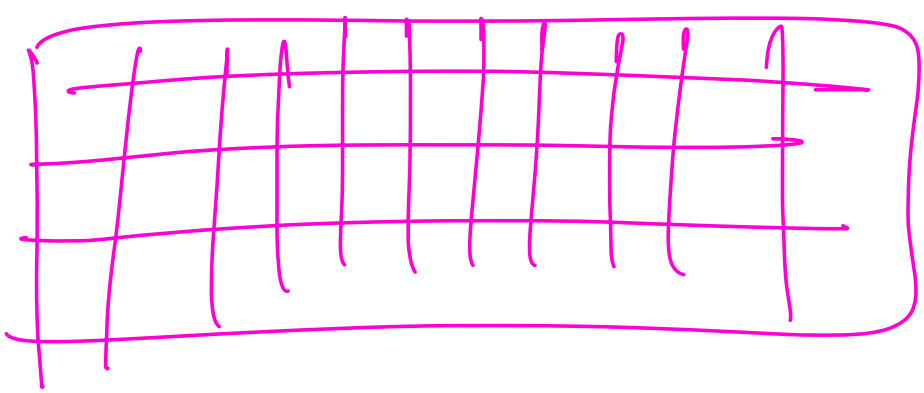
Next Class
- → Schema design
- → Start Code

Agenda: 2 Org problems

Two Challenging Problems

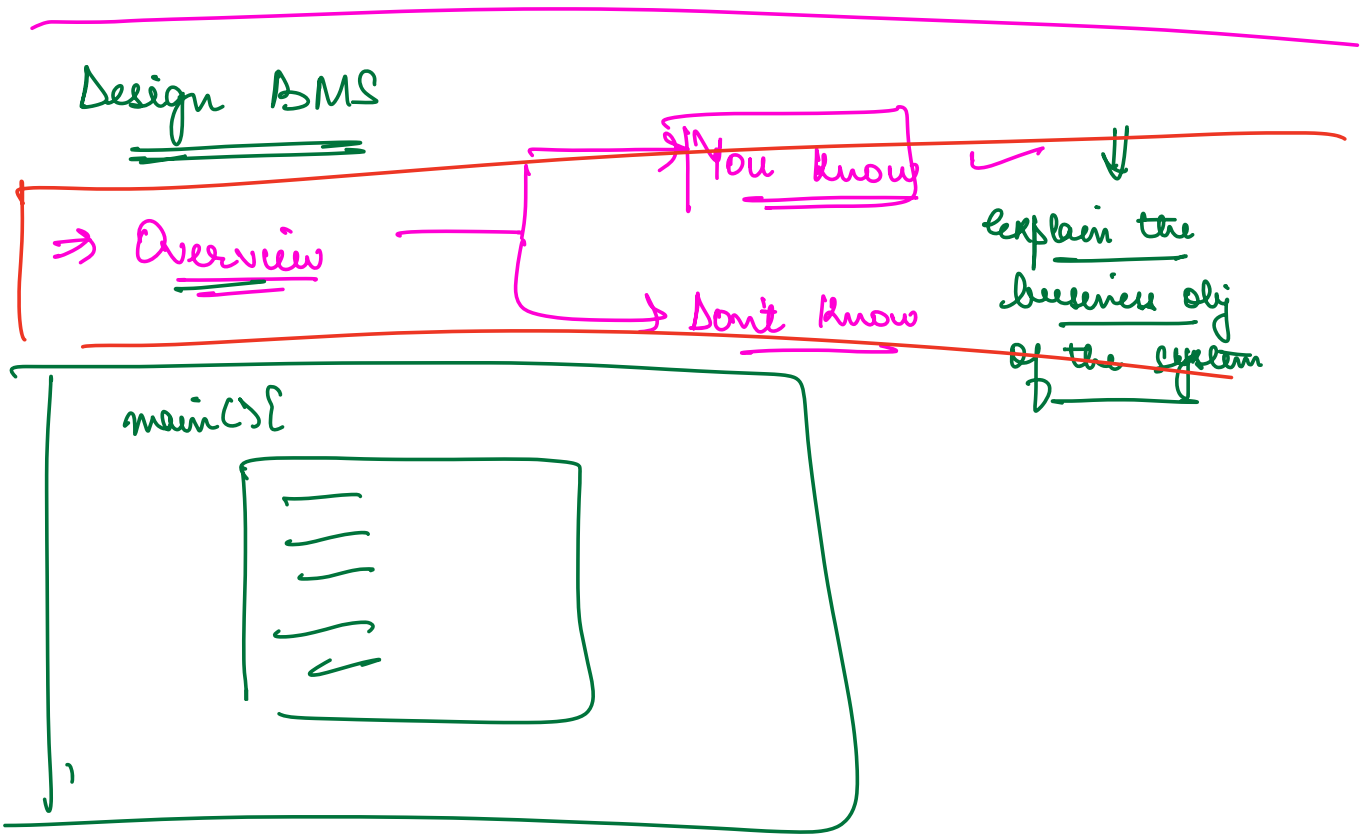
① How to rep seats in a theatre





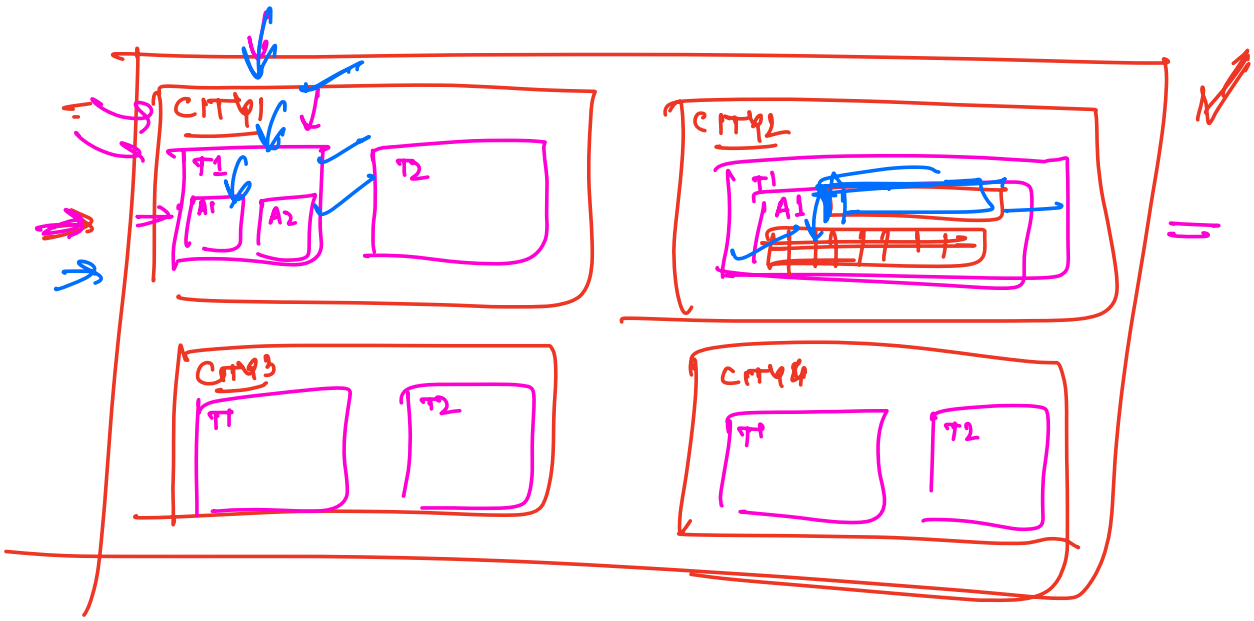
② How to book seats

⇒ Multiple people may want to book the same seat at the same time



⇒ Requirement Gathering ⇒ Suggesting features

→ only support booking movies for now
→ only reg users can book
→ no dynamic pricing.



- Support multiple cities
- People should be able to see diff movies being played in a city
- For a movie, in that city, people should be able to see what all theatres and what timings there.
- 1 city with multiple theatres
- 1 theatre can have multiple audi
- 1 audi will have multiple seats
- Seats can be of multiple types.
  - SILVER
  - GOLD
  - PLATINUM

→ Are Seat types constant across theatres or a theatre can name them as they want to

```
Class SeatType {
    name;
}
```

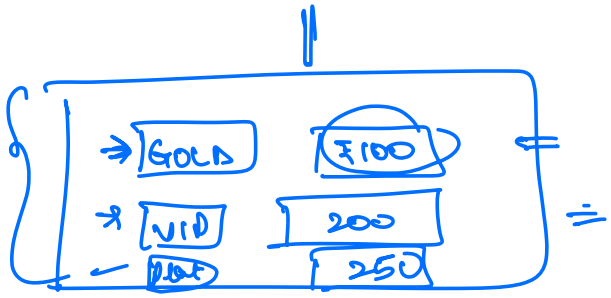
6-9 AM
9-12 PM
12-3 PM
3-6 PM
6-9 PM
9-12 PM

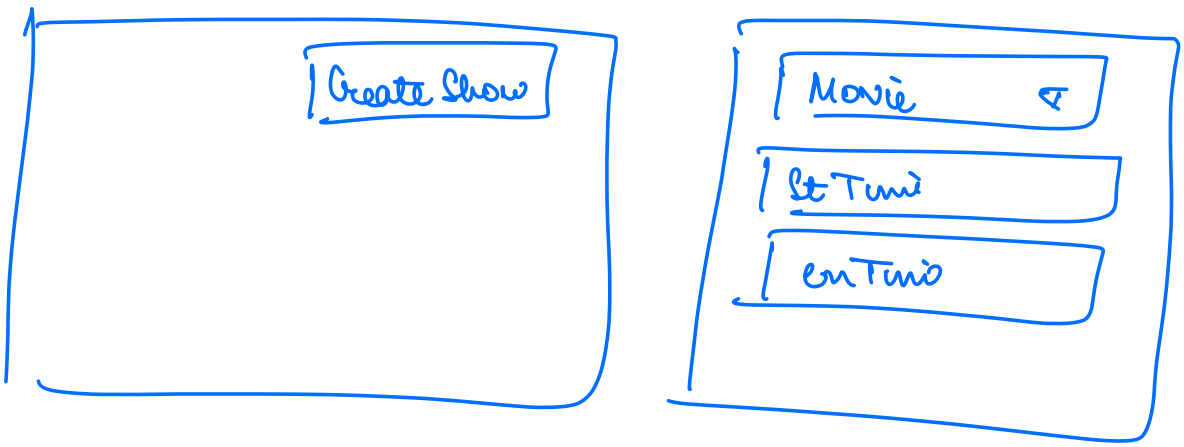
Jaha Tak Hai Jaan

→ Price of a seat is dependent on the time of the show, the theatre, the type of seat.

Price is an attr of rel^n b/w SeatType and Show

Show
- 9-12PM
- PVR
- Movie





→ At a time a user can select max of 10 seats.

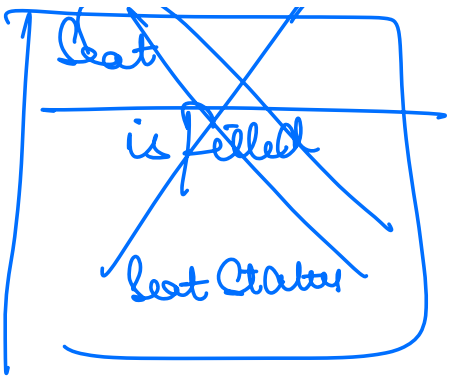
→ Diff audi can have diff seat structures

→ We are only building user facing features not admin part

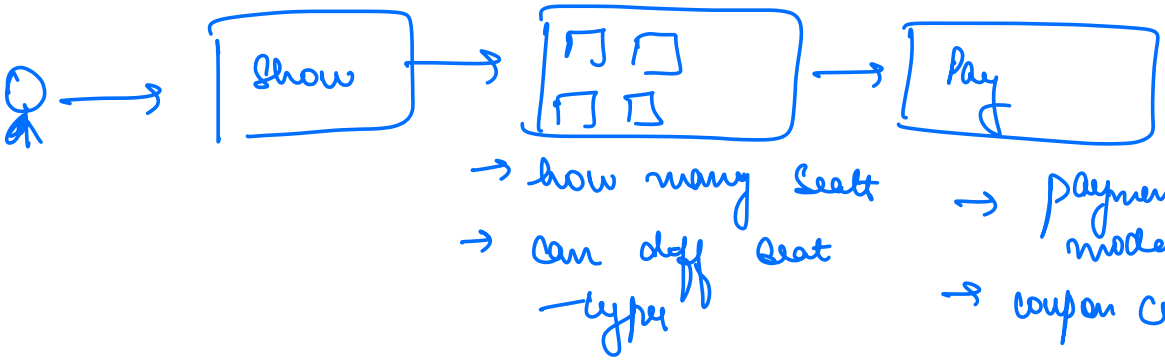
→ Do we store info about cast of a movie. → Yes

→ What all attrs of a movie do I have to support
  - → rating
  - → duration
  - → features (2D, 3D, IMAX, DOLBY, SUBTITLES)
  - → languages
  - → Co

→ Show time a variable.



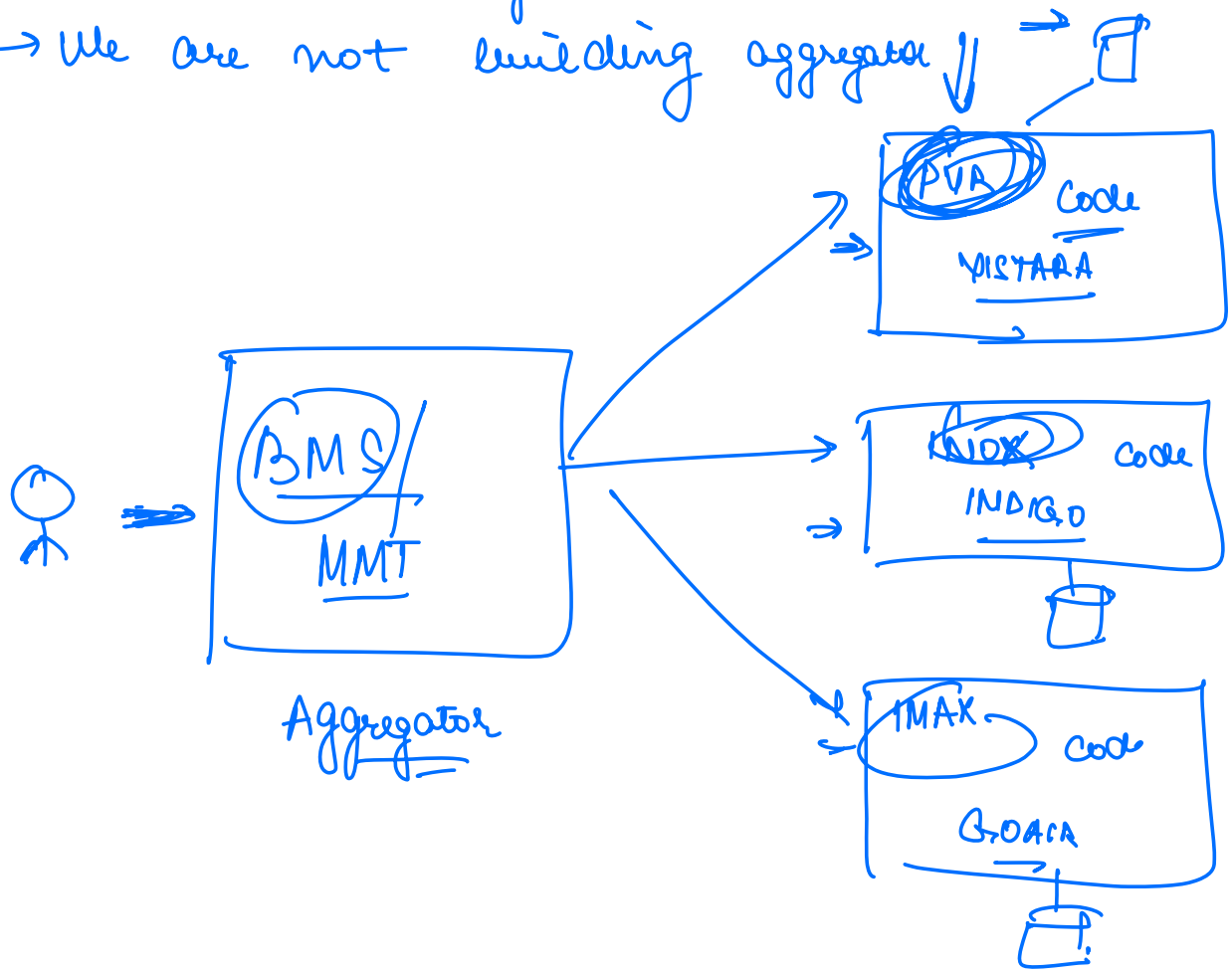
- No support for add ons.
- No discount/coupon code.
- Only allow booking till 30 min before start of a show.
- Only book for one show at a time.
- Can have diff seat types in same booking.
- a user can cancel a ticket and get refund.
- if one person is booking a seat currently, others should not be able to book
- only support online payments managed by 3rd system.

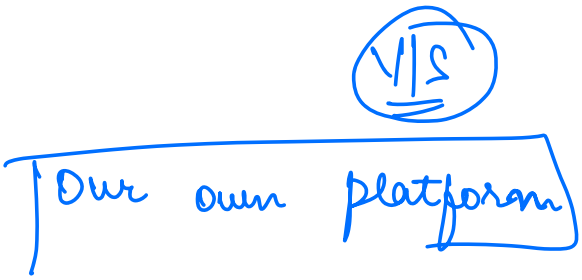


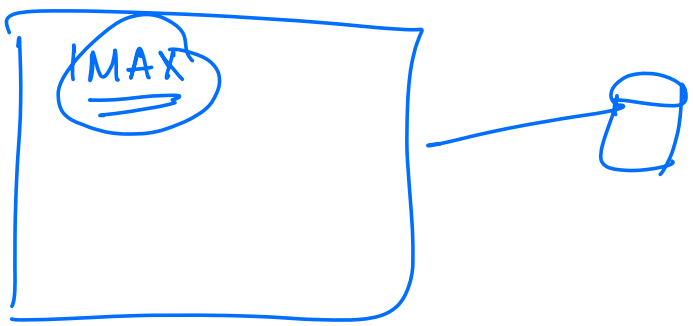
→ Physical Structure
- → Cities
- → Theatres
- → Audi
- → Seats → Seat Type
- → Pricing.

→ Use Case
a) Booking Movie
- → Movie
- → Refunds and Cancellation
- → Payment

→ We are not building aggregator





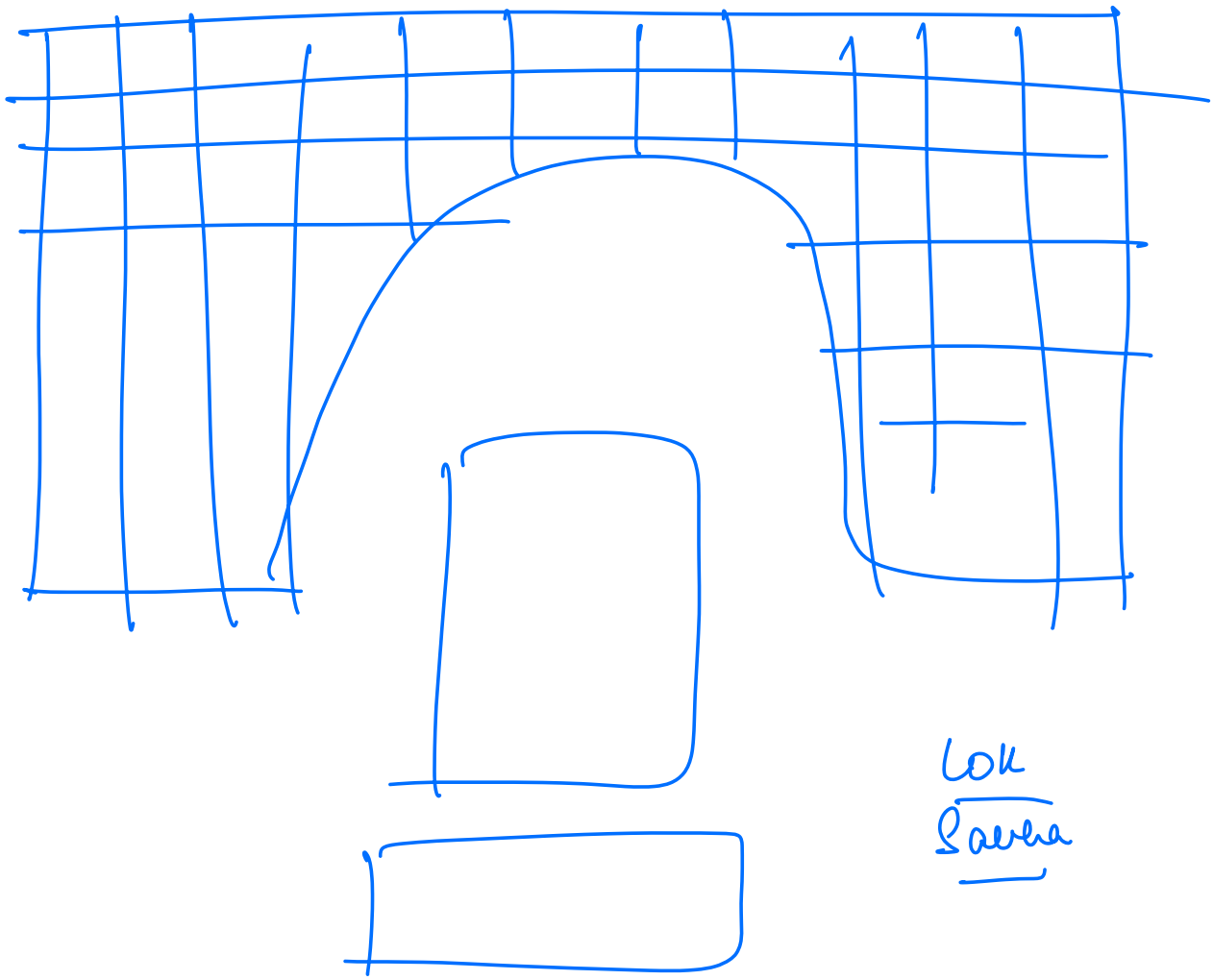


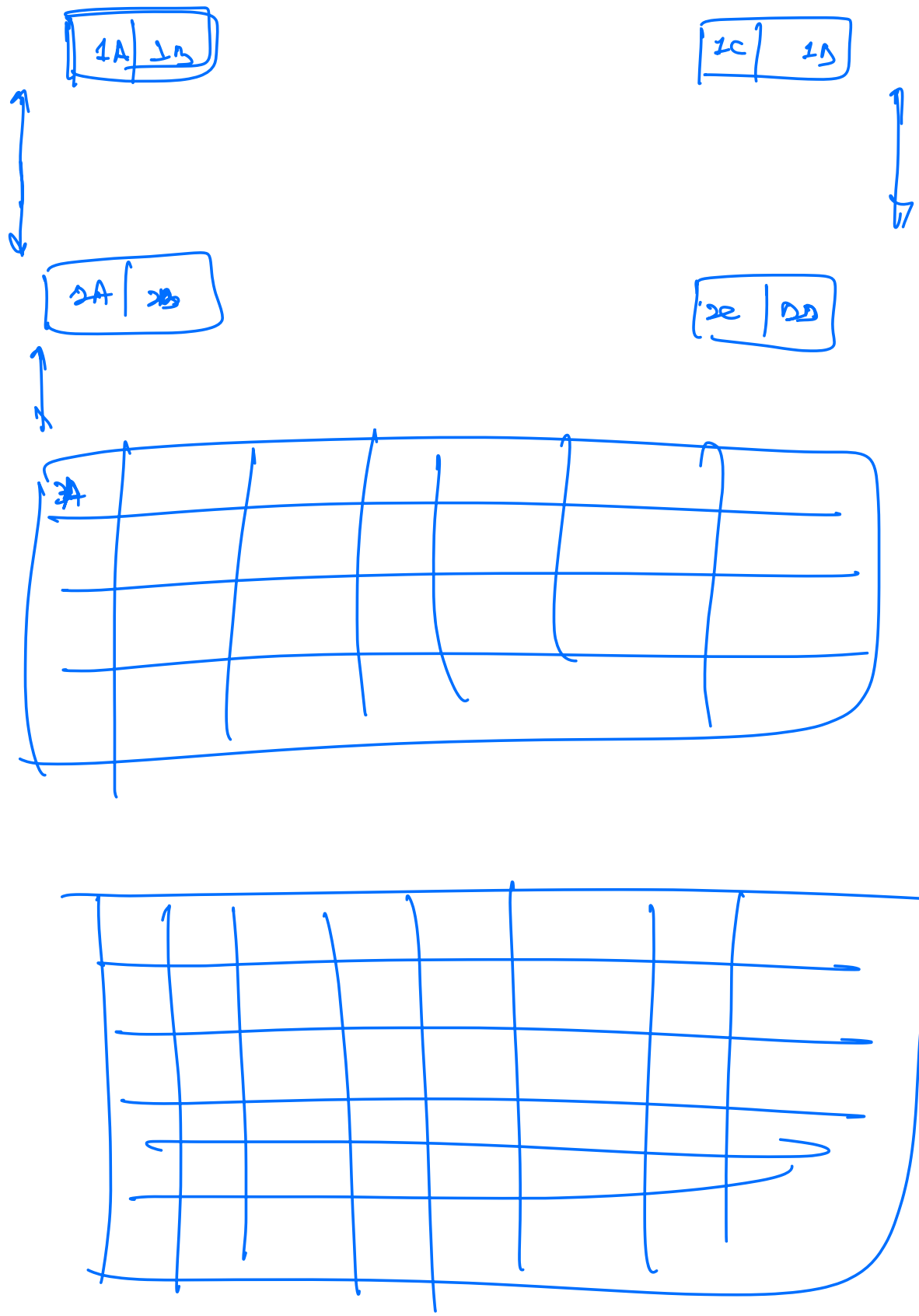
⇒ Lock seats

⇒ Handle conc

⇒ If we were building Aggregator we would have used Adapter design pattern to talk to 3rd party theatre APIs

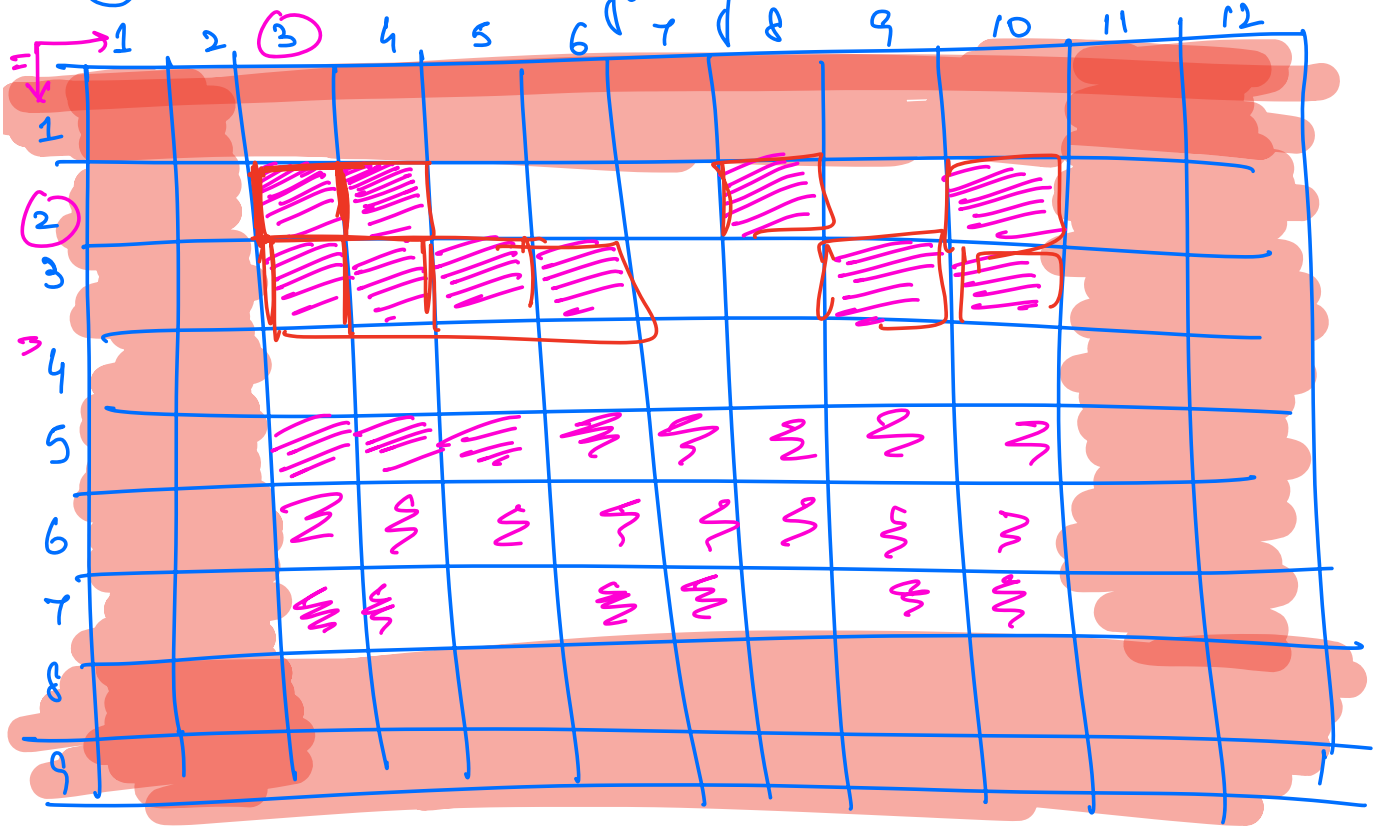
① How to Store the Structure of Seats (layout) in BMS




1A
1B
1C
1D
2A
2B
2C
2D
3A

KISS ⇒ Keep it Simple, Stupid

(1) Assume an imaginary 100×100 matrix




```
Seat {
    int row = 2 , 2 , 2 , 2
    int col = 3 , 4 , 8 , 10
}
```

```
Seat {
    row =
    col =
}
```

class Audi {

- List < Seat >

}

Aeep = {

name: —

Seats : [

{ 2, 3 },

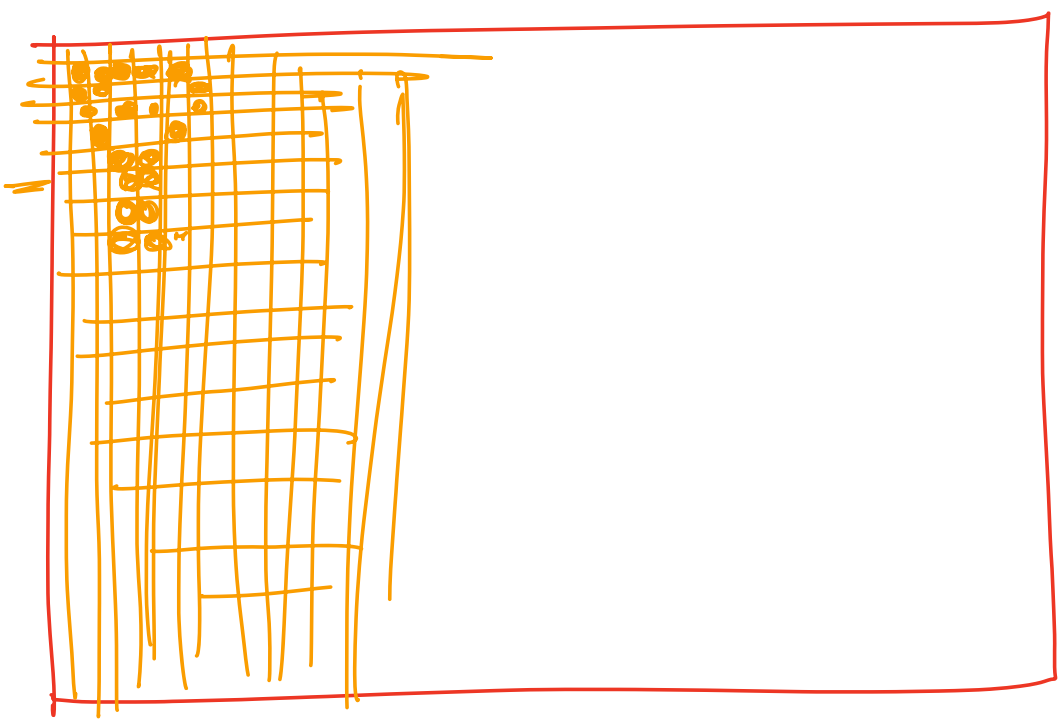
{ 2, 4 },

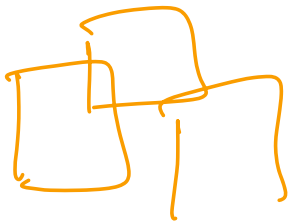
{ 2, 5 }

]

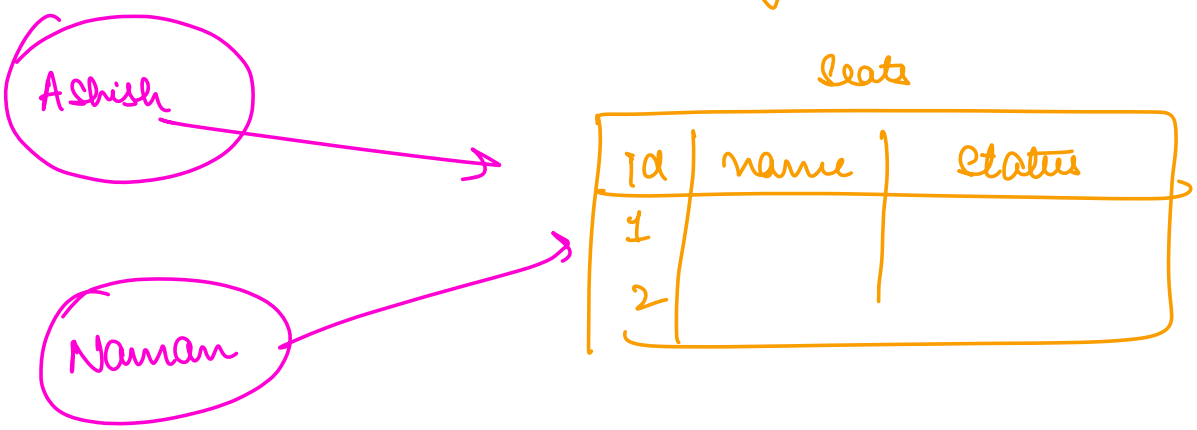
}

720p 1440p 360p





How to handle seat booking



---

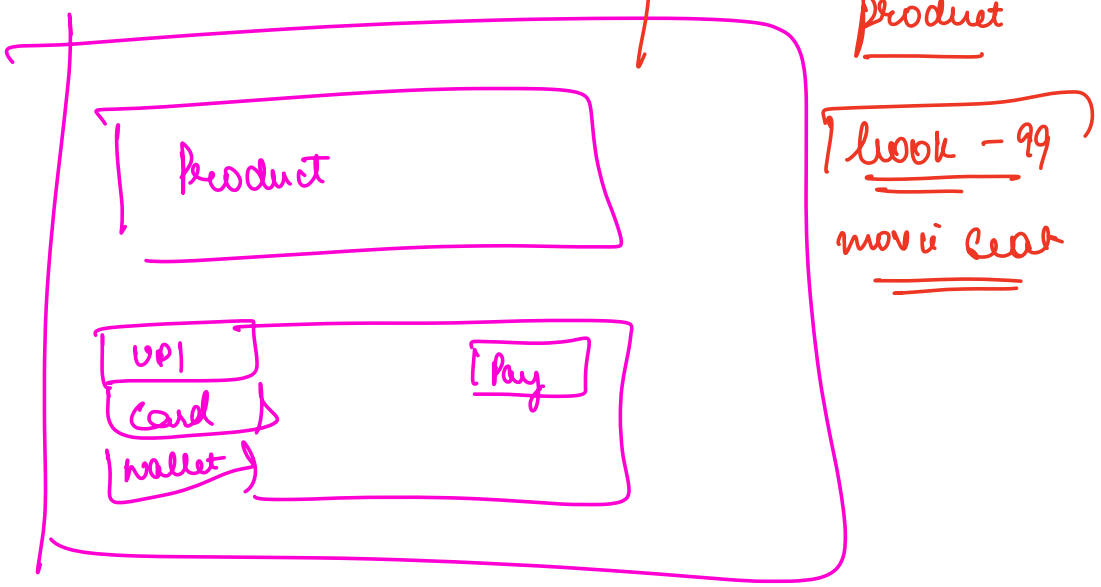
(1) Can we allow 2 people to select the same seat ⇒ Yes

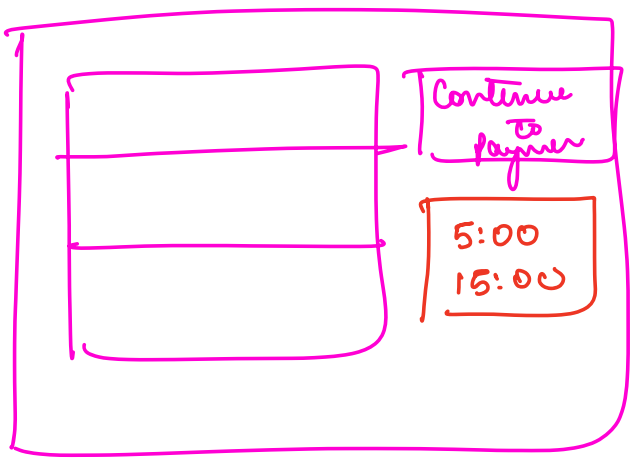
(2) Can we allow 2 people to book same seat ⇒ No.

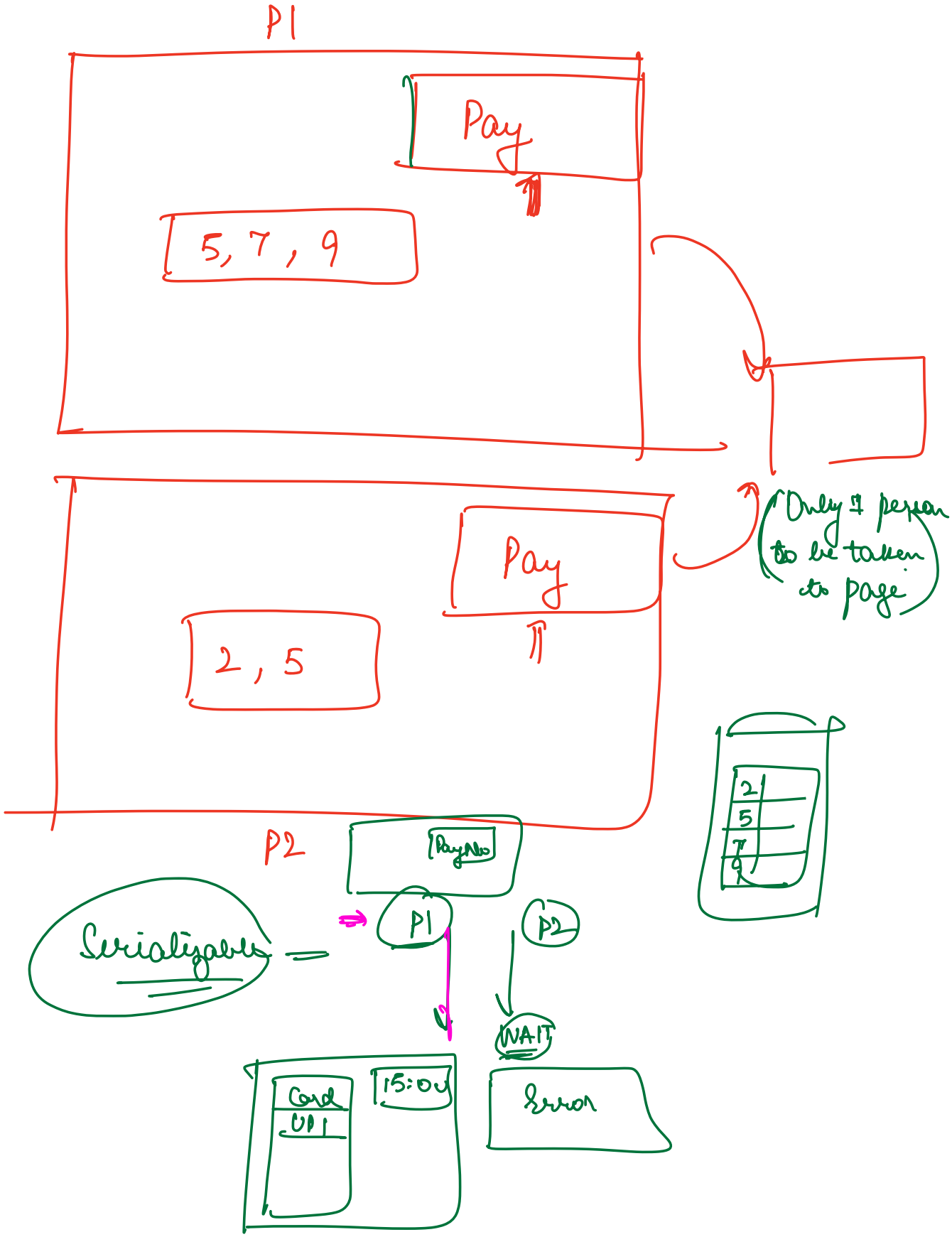
2 Step process to buy product
FK | Amazon | MMT | BMS

(1) Select the Product and click continue to Pay.

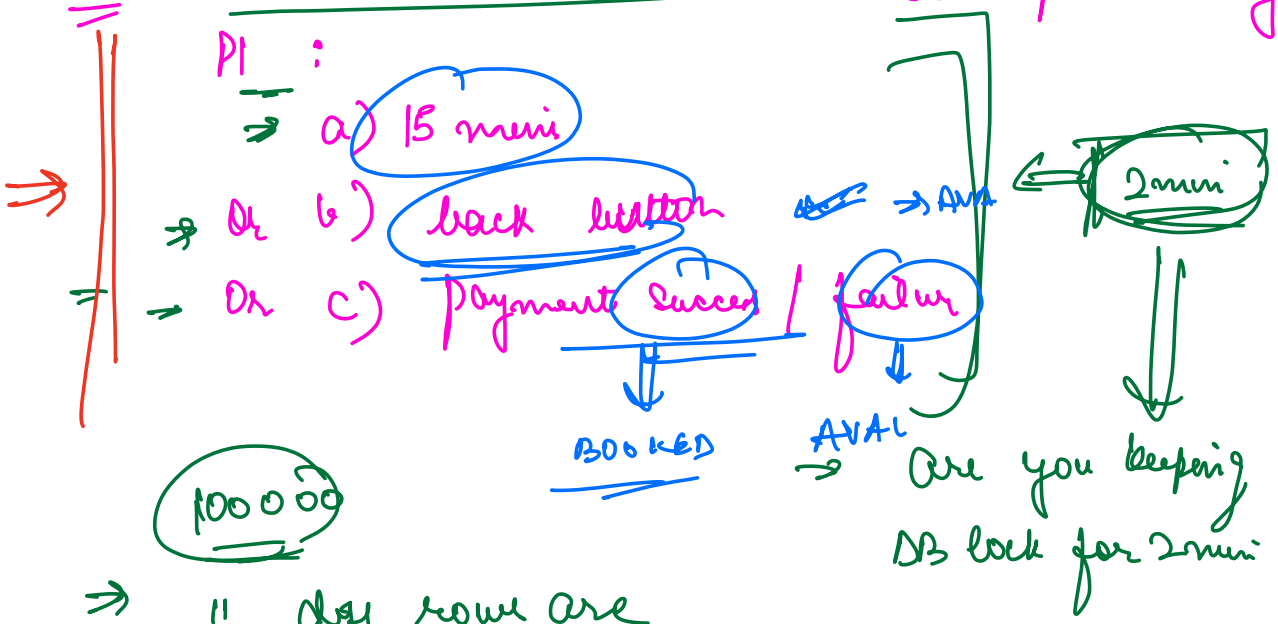
(2)






P1
Pay
5, 7, 9
Only 1 person to be taken to page
Pay
2, 5
2
5
7
9
P2
PayNow
Serializable
P1
P2
WAIT
Card
UPI
15:00
Server

Q^n : Till when should DB lock be kept for?



100000

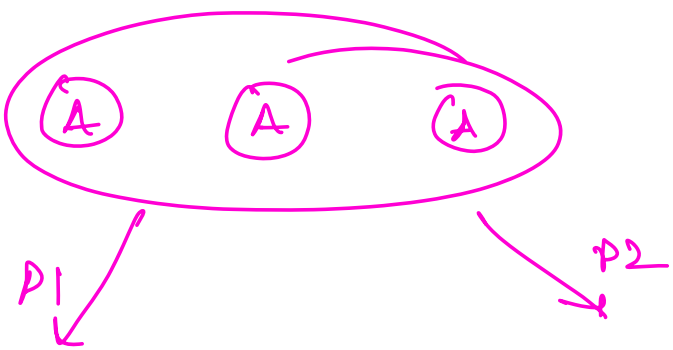
⇒ 11 days rows are locked ⇒ Slow, CPU usage, Memory usage

⇒ You shouldnt keep a lock till end of payment.

bookTicket (list<seat>) {

1. get those seats ⇐ DB lock
2. check the status of seats ⇐ DB lock
3. if status of any seat is not available. ⇐ DB lock
   ⇒ return
4. go to payment

}



⇒ Soft locking : Change the status of seats to a new one ⇒ LOCKED

Seat {

Status → BOOKED
Status → AVAILABLE
Status → LOCKED

}

10ms

1. Take DB lock ← Q1 — Q.5ms
2. get the seats ← Q2
3. Check Status ← Q2
4. if any is not available ← Q2
   remove lock
   return

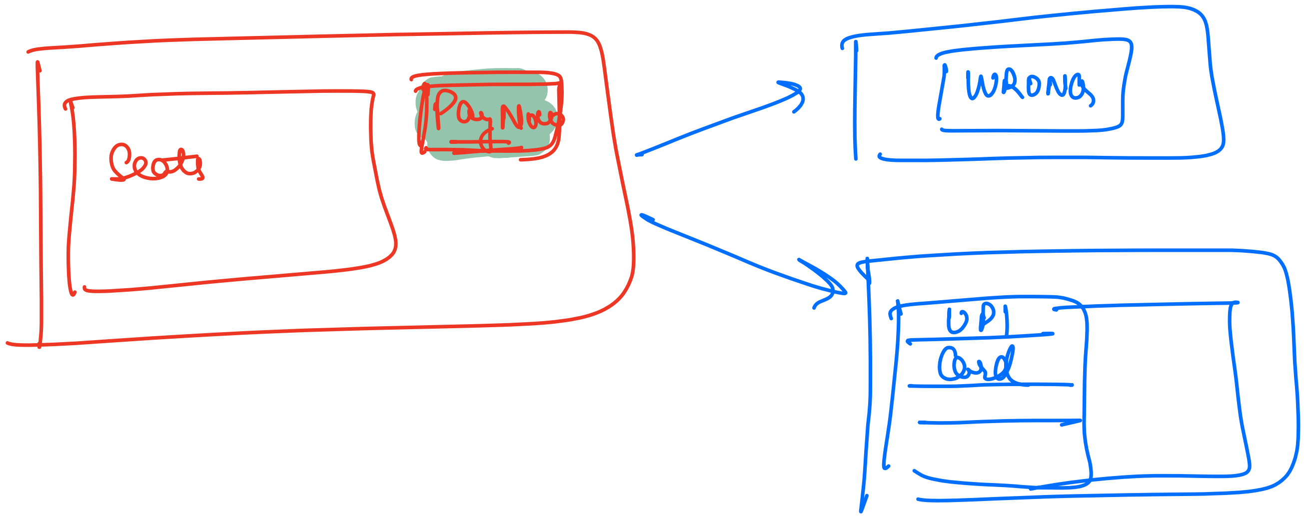
Or (Time of locked → 15 mis are)

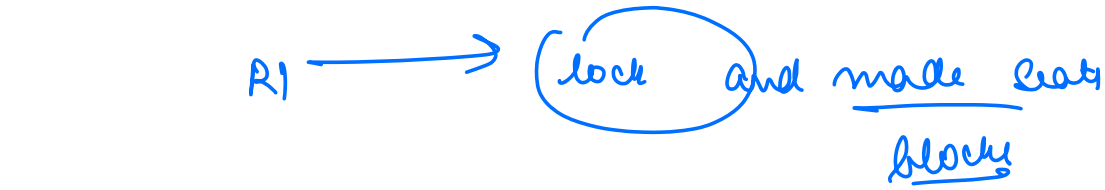
5. if every seat is available:
   change status of all seats to lock
   remove DB lock

Q1

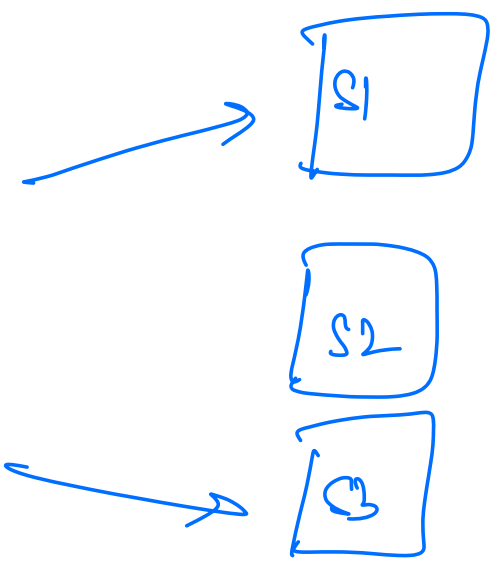
Keep locked till here

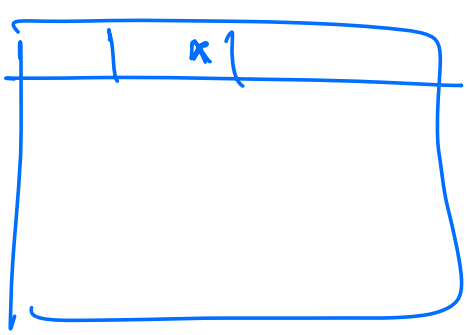
5, 6, 7




R1
lock
and make
Block


R1 + 1 ms


S1
S2
S3


R1

KISS