Design Snakes and Ladders

Razor Pay

→ Design an SMS service
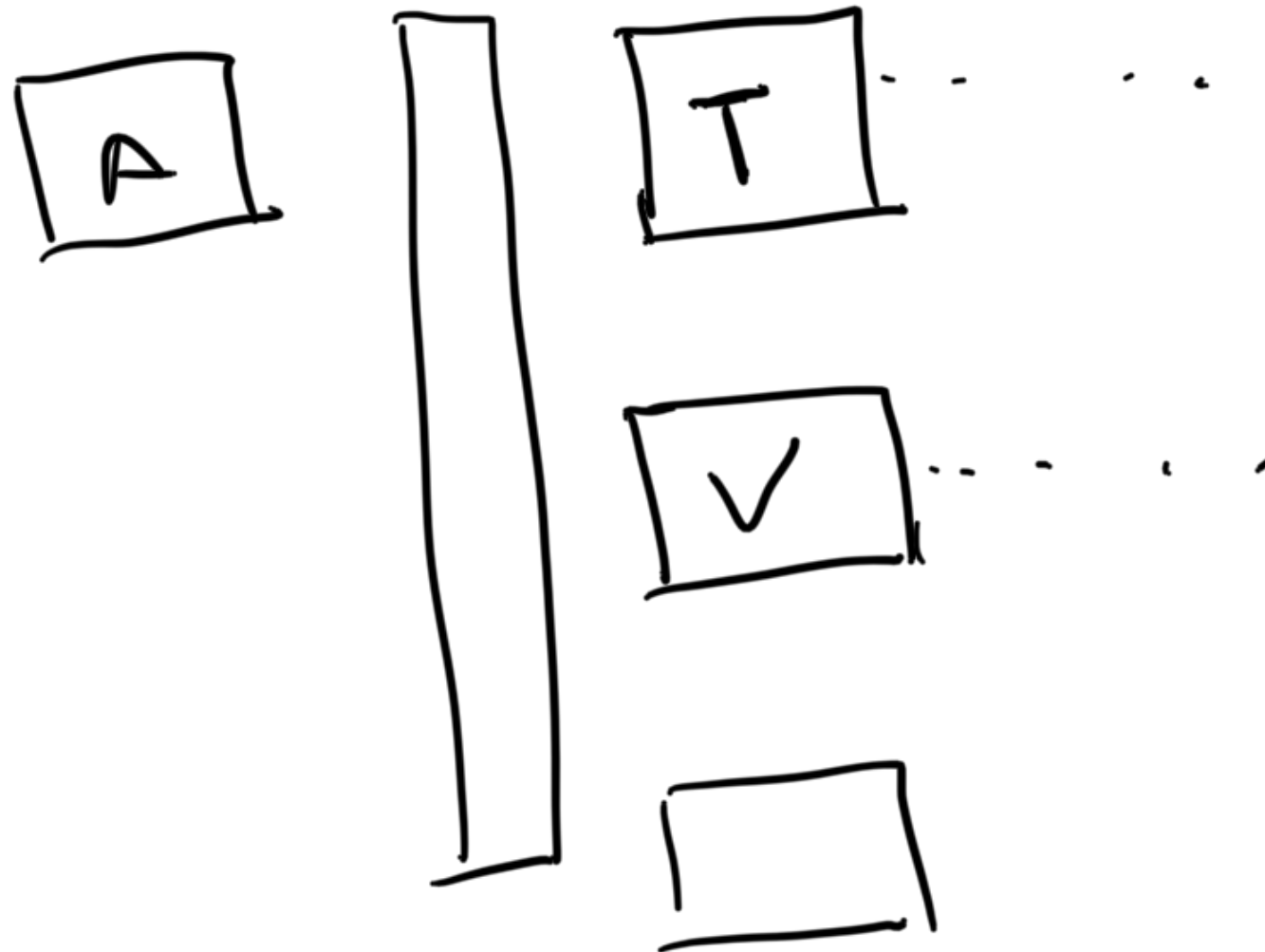
A | T - - - - -

V - - - - -

Communicator

① Design | Assignment
      — Initial | — Implement
      — Optimised | Snakes & Ladders

② Command OP

      ⟶ Consuming input
      ⟶ Creating CLI applications

| git | yarn | npm | Brew |

→ Status

→ pull

→ push   }] → Command

→ commit

---
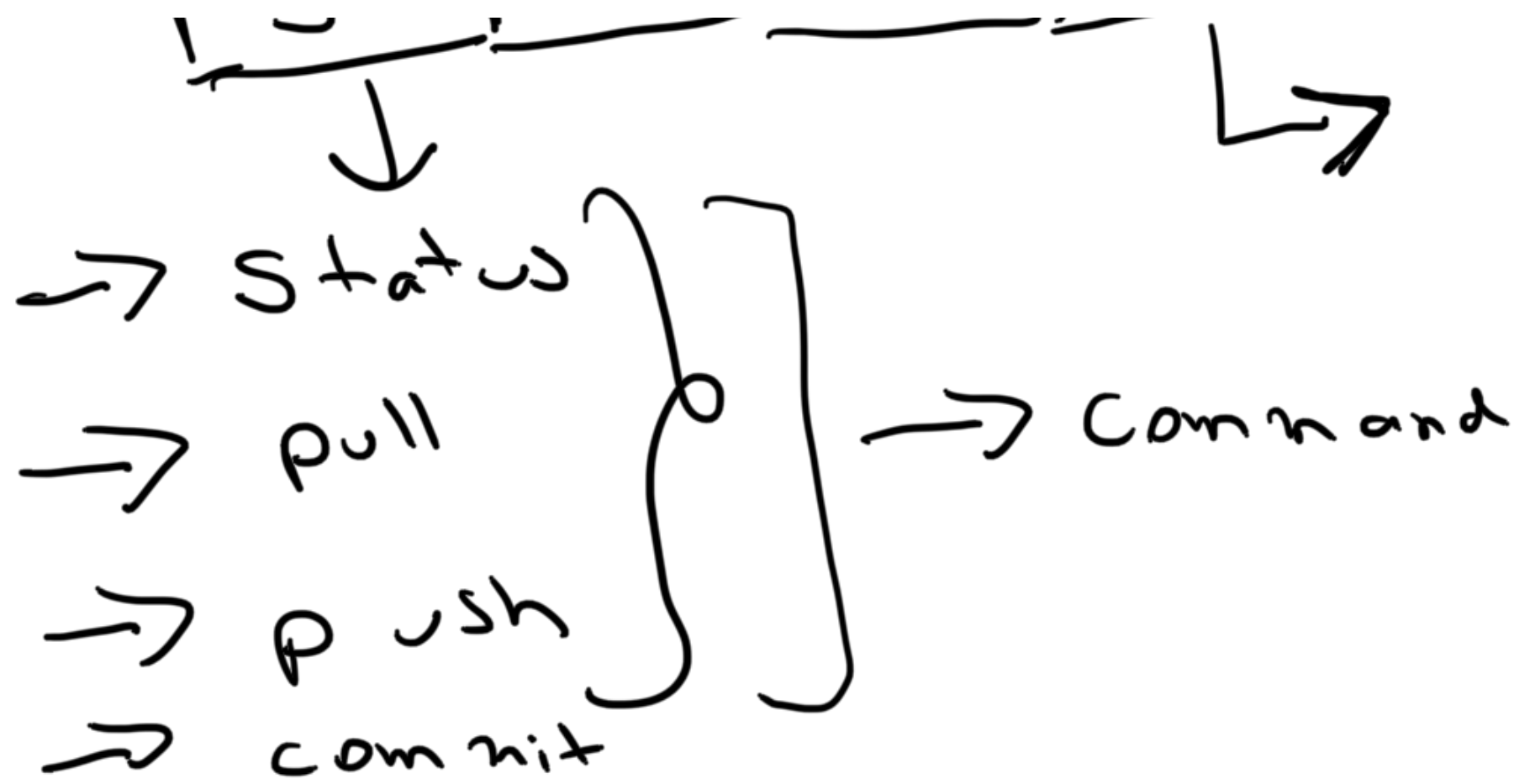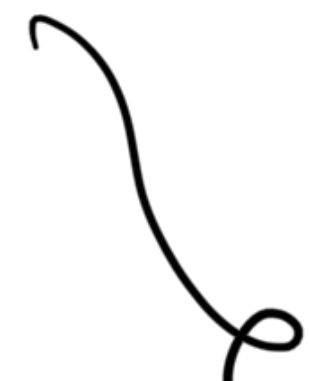
① Design Snakes + Ladders

→ Overview

→ Requirements

Current scope →

1. Board size → 1 - 100
$$1 - N$$

2. Multiple players → > 2

3. Only human player

4. Dice — 1 on Maximum

5. 1 Dice | multiple dices
$$3 + 4 \Rightarrow 7$$

6. Snake & Ladder
→ Random
→ User

$\rightarrow$ Eg Snakes
$\quad\quad$ & Ladder.

⑦ A Player wins by
$\quad$ reaching the last cell

$\quad$ — piece

⑧ A player can have
$\quad$ multiple pieces

⑨ 1st — All the pieces

(10) N | If n-1 players
are done, game ends

(11) Leaderboard

$$1 \longrightarrow \text{Player 1} \Big\} 4$$
$$\longrightarrow 1 \longrightarrow \text{Player 2} \Big\} \, 3$$
$$\longrightarrow 3 \longrightarrow \text{Player 3} \Big\} \, 2$$
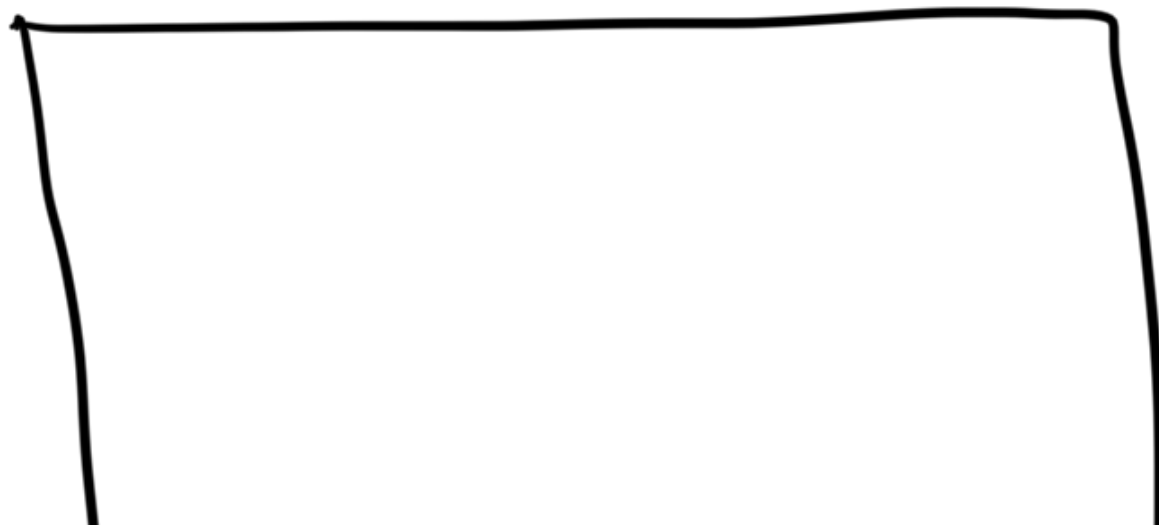
P1 O X | $\emptyset$ | 1 |

| 1 | | 1 |

Entities

Board

Game   N   1 – P1
          2 – P2

ECHO

players

6 2

P1 P2

N × M

N

Game
- Board
- Players
- Dices
- Leaderboard
- Status

Dice

- faces

+ roll()

+ createGame(size, players, dice, ...)

+ roll()

+ makeMove(Player, Piece)

+ getLeaderboard

Board

- Cell[] cells

- Snakes

- Ladders

Player

- name

- pieces

Piece

- colon

Cell

- pieces
- position

1

M

Snabe

- stant
- end

Name

Leader B.

PlayerR[]

Map < Player , Integer >

Player Ranking
Player player-
Integer ranking
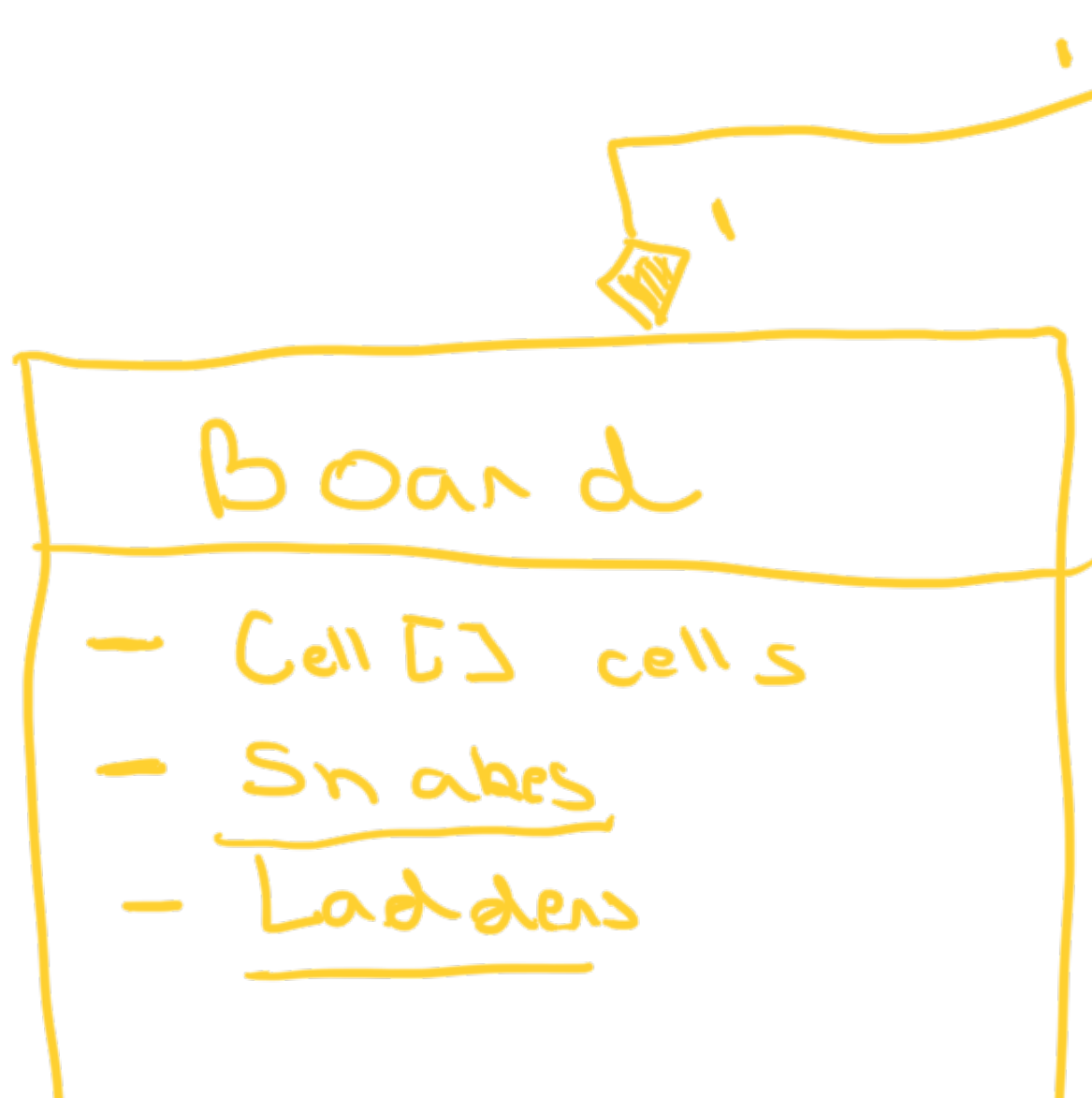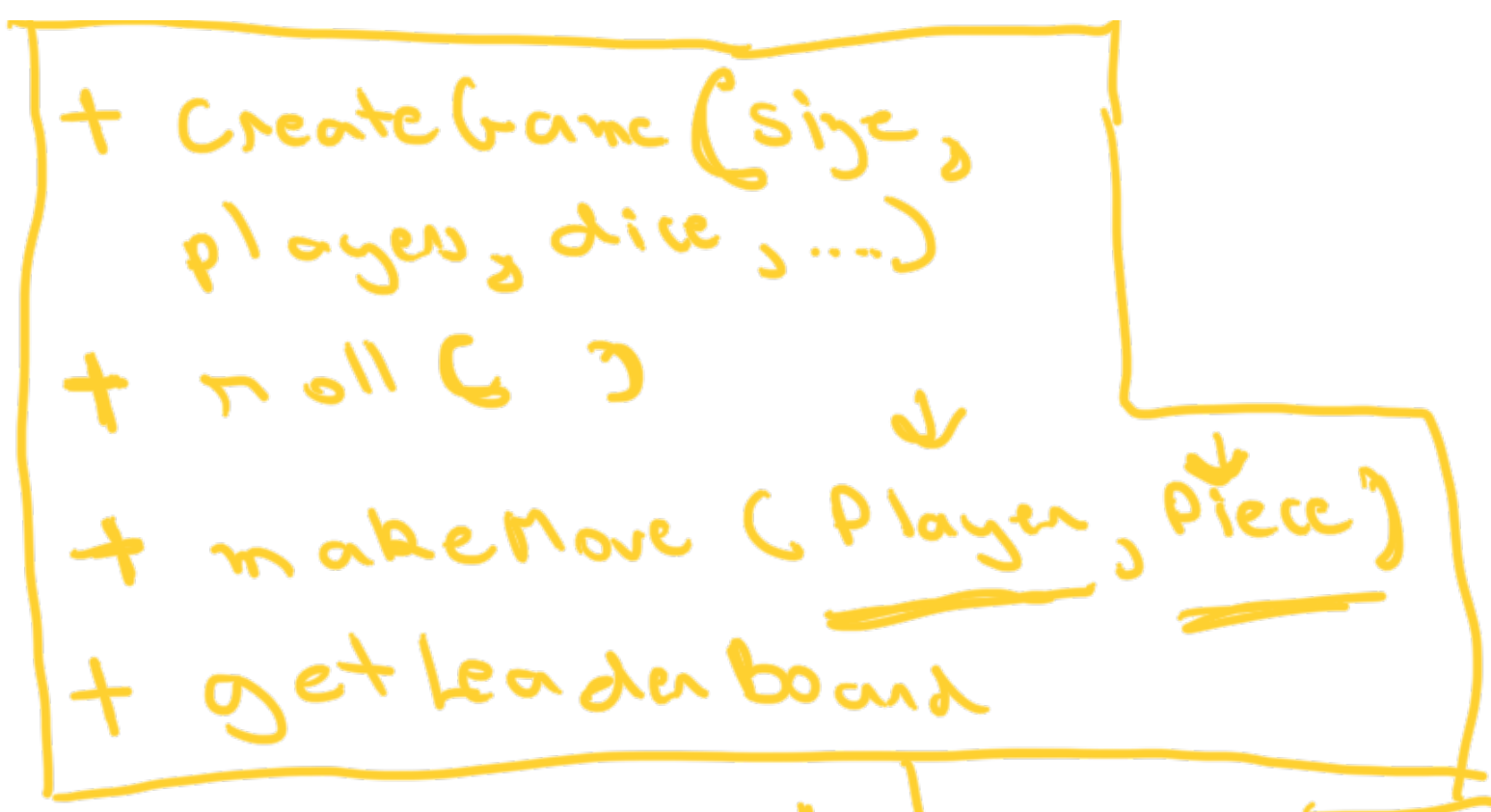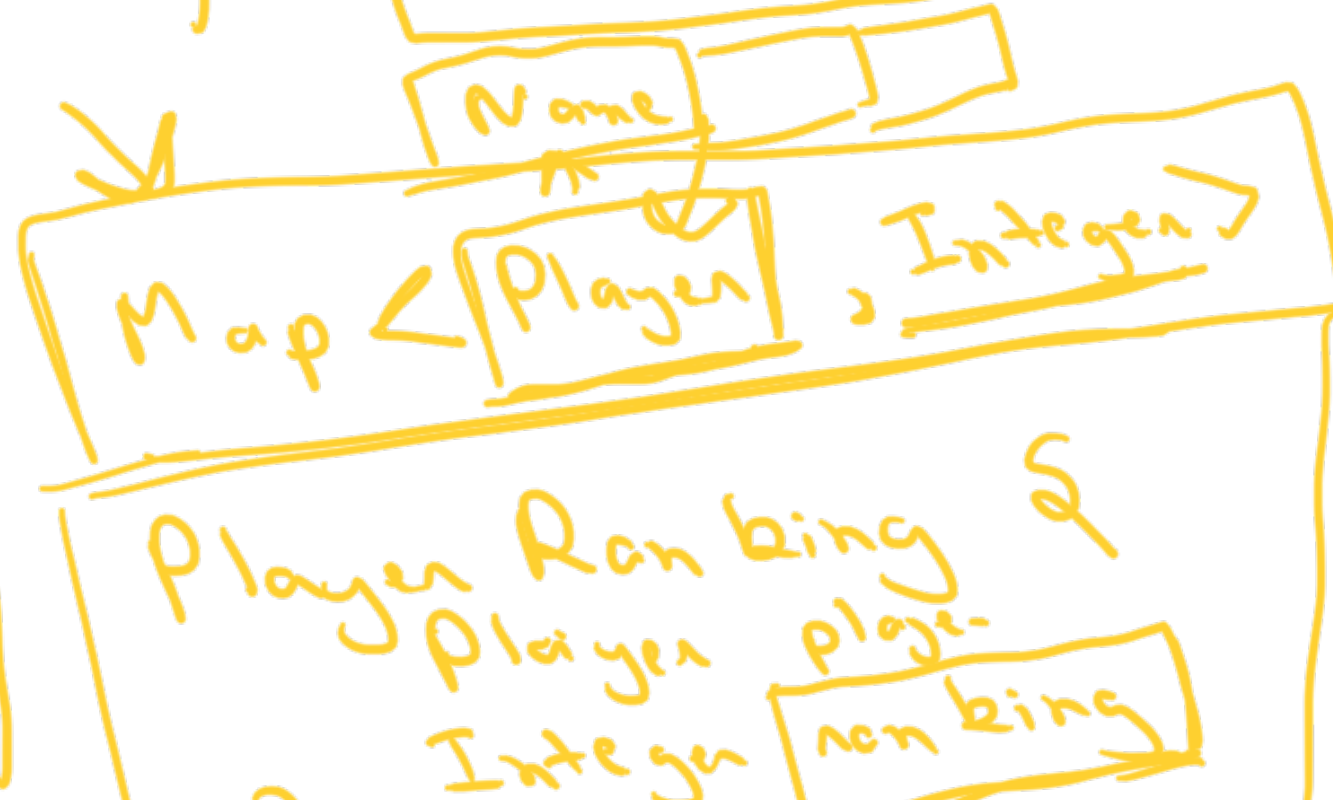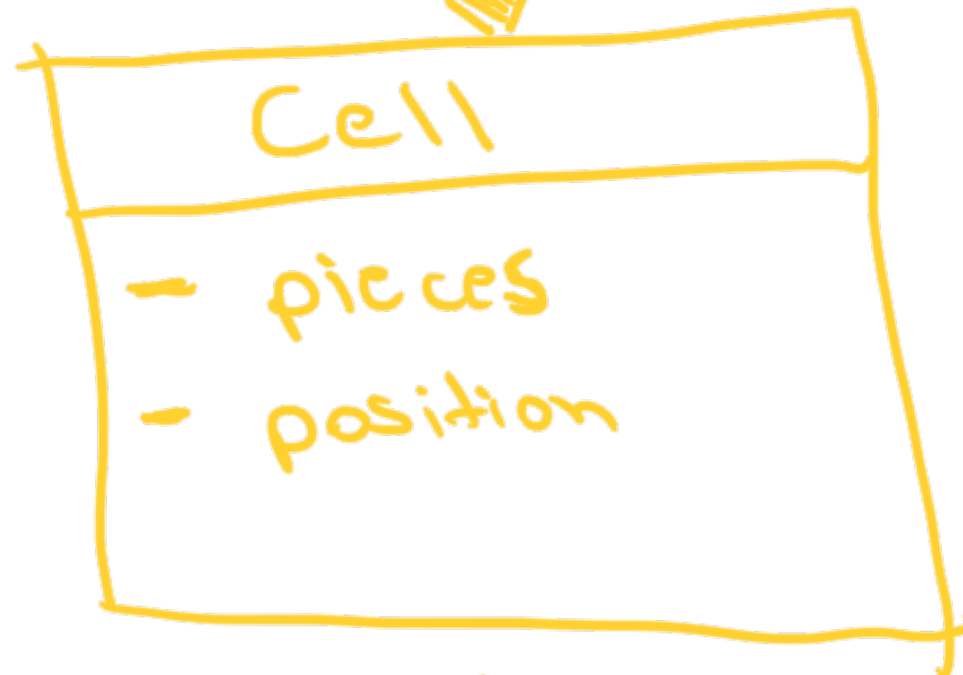
Zen of Python → Explicit over Implicit
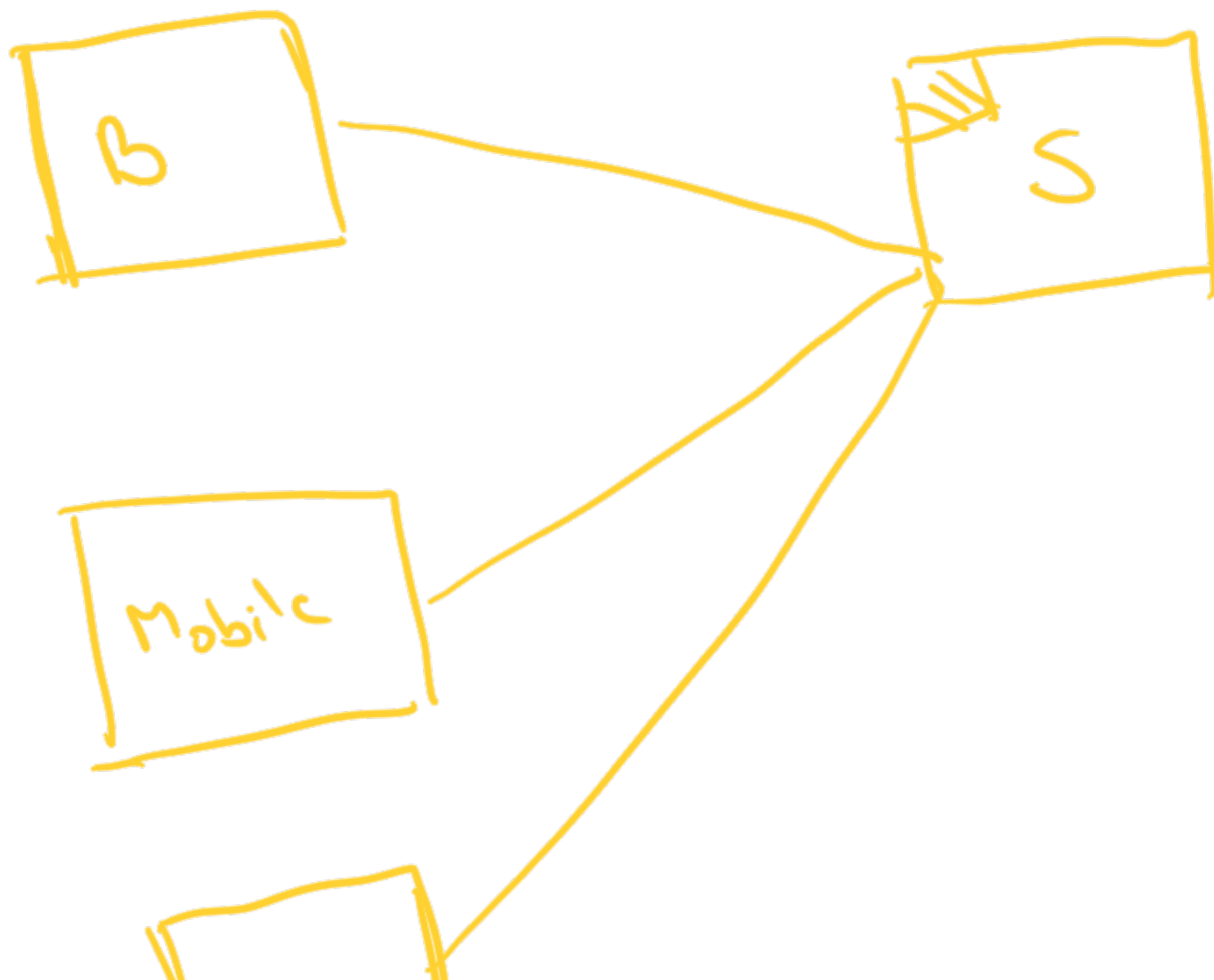
Break

6:10 — 6:15

10:45

Create Order [Flipkart]

Store in DB → Send SMS → Create Tracksing

Book My Show

— Concurrency