System Design

Think/Pair/Share

design a grocery store checkout

- Pair up with your neighbor in groups of 2-3
- using some of the ideas you learned in lecture, think about the above prompt
- After 5 minutes we'll discuss with the class

Think Trader Joe's for load balancer, queues, throttle

Fandango Reacto Follow up: Where and how can we introduce a load balancer(s) in the system?

- Pair up with your neighbor in groups of 2-3
- Where and how can we introduce a load balancer(s) in the system?
- How might we be able to introduce a cache?
- After 5 minutes we'll discuss with the class

Discussion on Taking the Booking System to the Next Level

It's first come first serve but how can we handle multiple users wanting to book the same seat? We should try to implement something like a timed system where you can reserve your seats by clicking on them and those seats are locked (based on availability) for a specific amount of time, say 5 minutes. After 5 minutes are up, and the user doesn't purchase the seats, the seats are freed up for someone else to take them. If the user does purchase the seats, the seats become fully unavailable.

Discussion on Taking the Booking System to the Next Level

- **Using Microservices for Reservation System**
    - **Microservice for Active Reservation**
        - We can keep the reservations in some sort of a cache (as well as a database) that when a seat is purchased, it is removed from the reservation list. But also, would release the seats if the expiry time passes.
    - **Microservice for Waiting Users**
        - Same idea as above but we have a cache of users. The user who was waiting the longest would be "next in line" for the seats.
        - Use web sockets to be updated on reservation status

System Design Questions

- Pair up with your neighbor in groups of 2-3
- What is database replication? In what situation might it be useful?
- How would you protect your API from being abused?
- What is sharding? Why is it useful?
- After 5 minutes we'll discuss with the class