Office Hours 10/12

Wednesday, October 12, 2022 7:47 PM

Swimlane for general flow

- Doesn't necessarily mean you'll use it for HLD but its more for processes in general
  - Internet traffic is good example of swimlane diagram purpose for HLD
    - What we have is not a swimlane diagram

Cross-cutting Concerns

- We have it within each layer-"Yep"
- For logging - will there be the same information within each layer?
  - Could be that some information is or is not accessible to the developer, we need to pick and choose
  - Example: input validation error is different from database error, have to get into specific errors logged
    - What information will be useful?
      - DB: which operation? Access, update?
      - Input validation: not over 18
- Logging CAN BE error handling but it's not typically the only thing you do
- You might also want to log successful user actions (such as log in/log outs)

Is logging its own service?

- We need to make that decision ourselves
- Chaining services goes against the self-containment or autonomous nature of microservices
  - Standalone paradigm, complete isolation
- Authentication or security are different routes
- What decision will you be making?

Data store error decisions

- Design decision based on data store
  - Feature might have auto-log, or lack of space
  - Decide which one you choose
- Can't just say "will handle any errors"
  - Too generic, not a design
- Ways to crash: not enough space, not online, etc
  - It's just another type of program at the end of the day

★ Hint:

Explain API gateway

Bounded context issue with separated databases

Current diagram is not realistic: account is connected to listing/scheduling

- Revisit bounded context
- Need to explain what information might be passed between databases
  - "If you want to schedule for someone but you're not allowed to schedule because you're not Bryan due to having a different account" - not sure what he meant with this example, maybe authentication of user?

Identify **pros and cons** of design to show it's a conscious decision

Clearly talk about the bounded context for each service

1 service per feature was the initial way people thought about microservices

- Eventually realized there are better ways