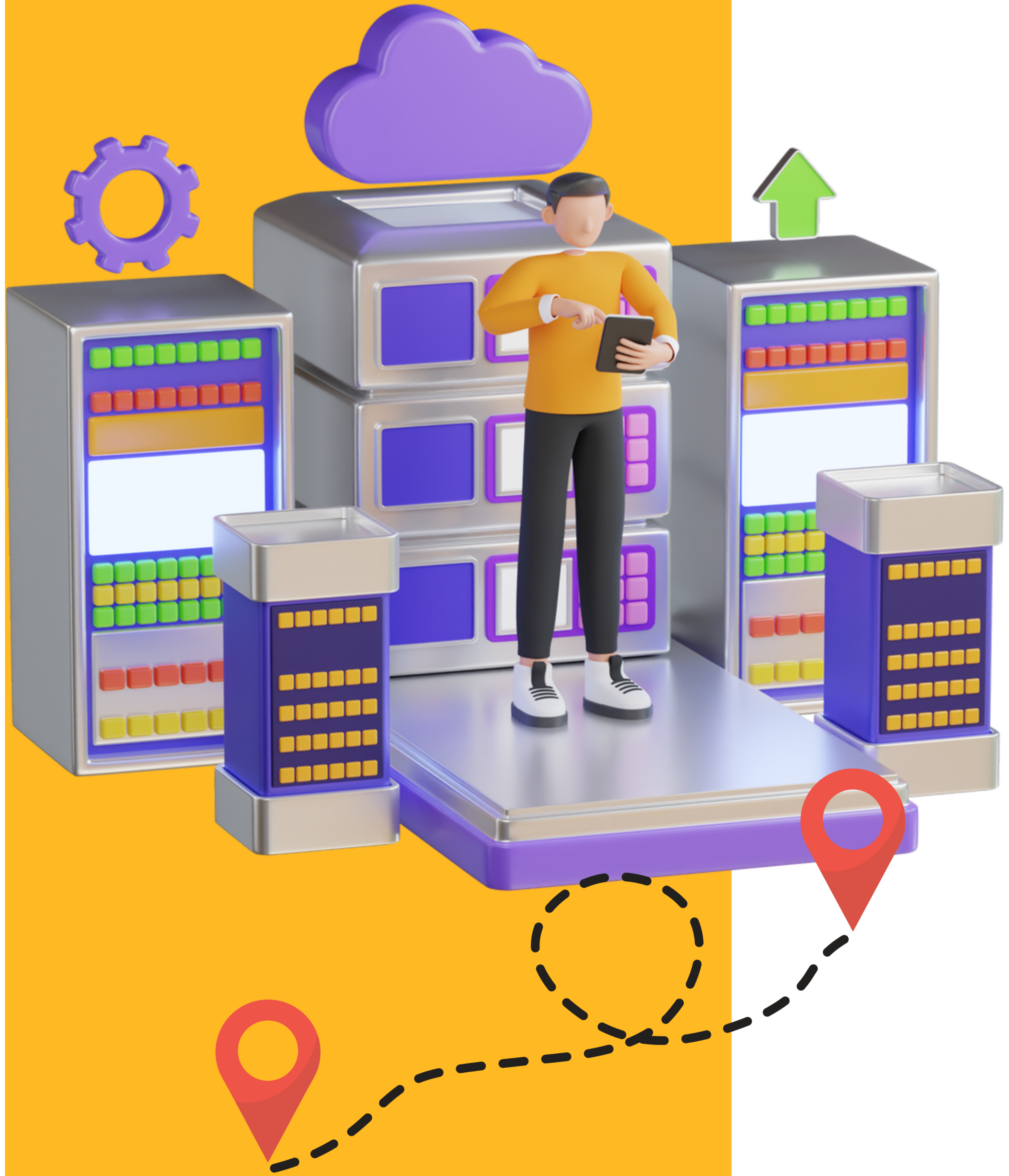


Tech Odyssey: An epic saga of Temporal Migration

Renu Yadav : Sr Engineering Manager

Who has continuously paid technical debt, ever ???

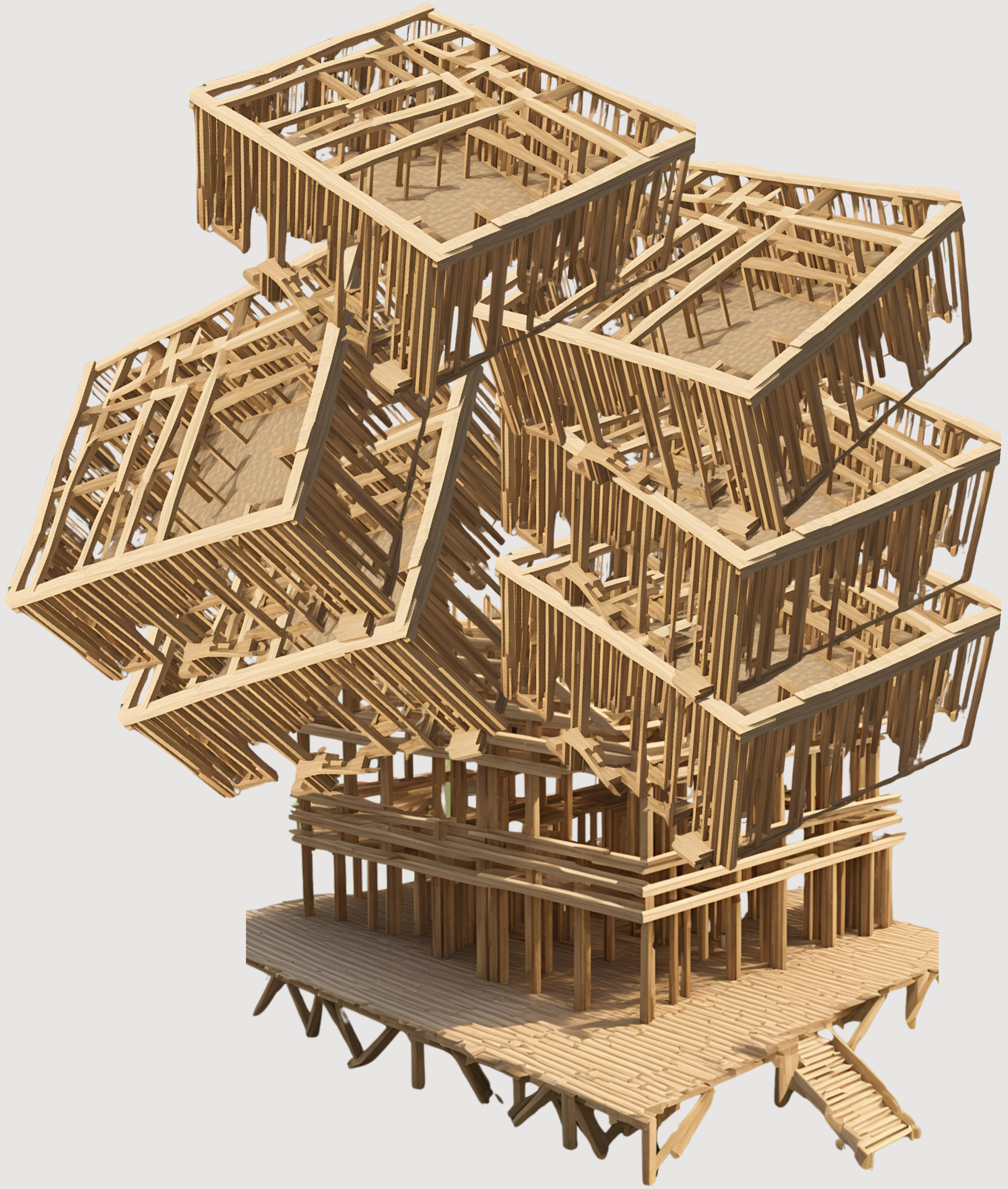
twitch.tv/HyperRPG
NO ONE!

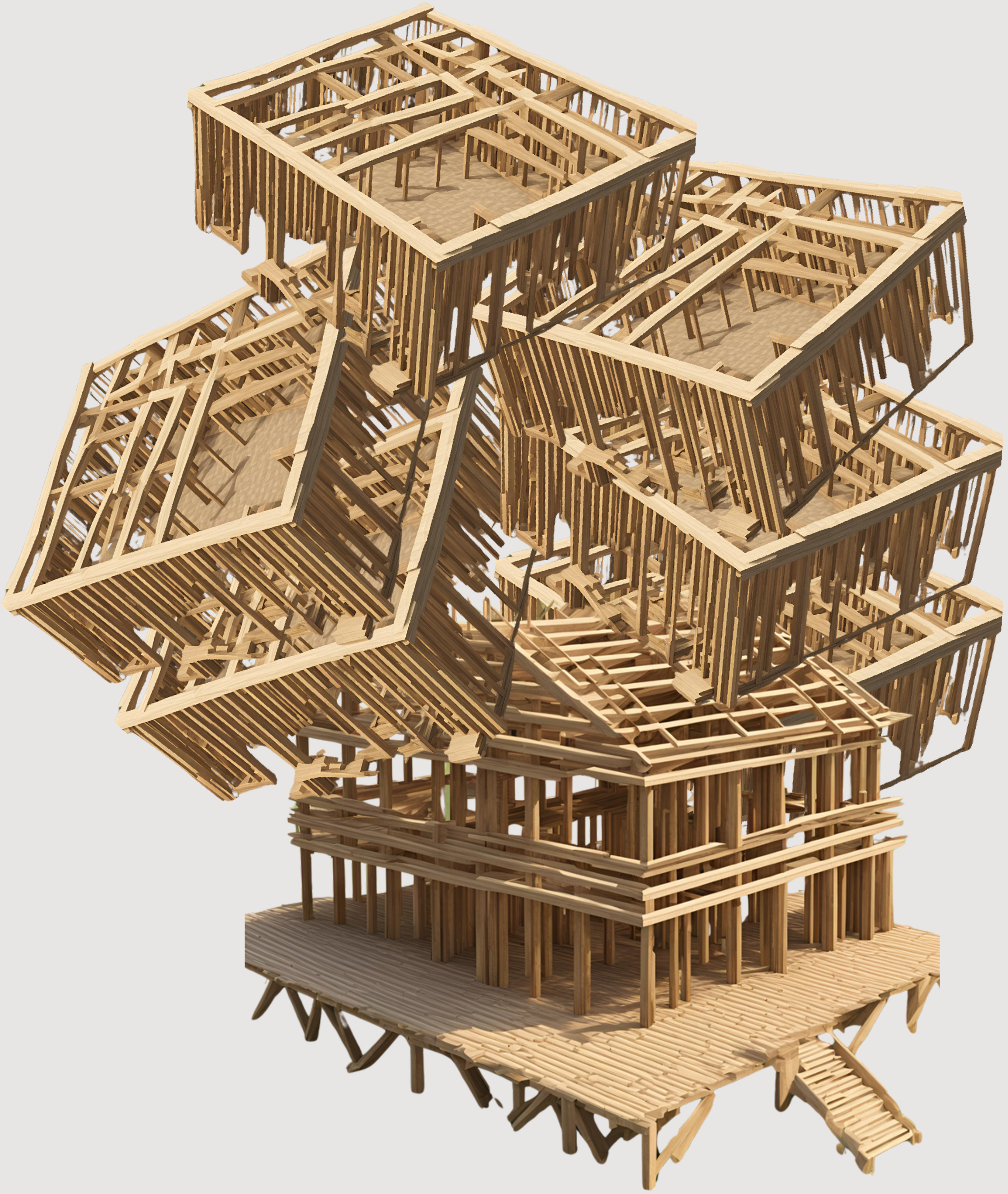
This is fine.
[adult swim]

Grab is a super-app for SEA providing multiple services

GrabUnlimited is a paid loyalty subscription program

Once upon a time in 2022

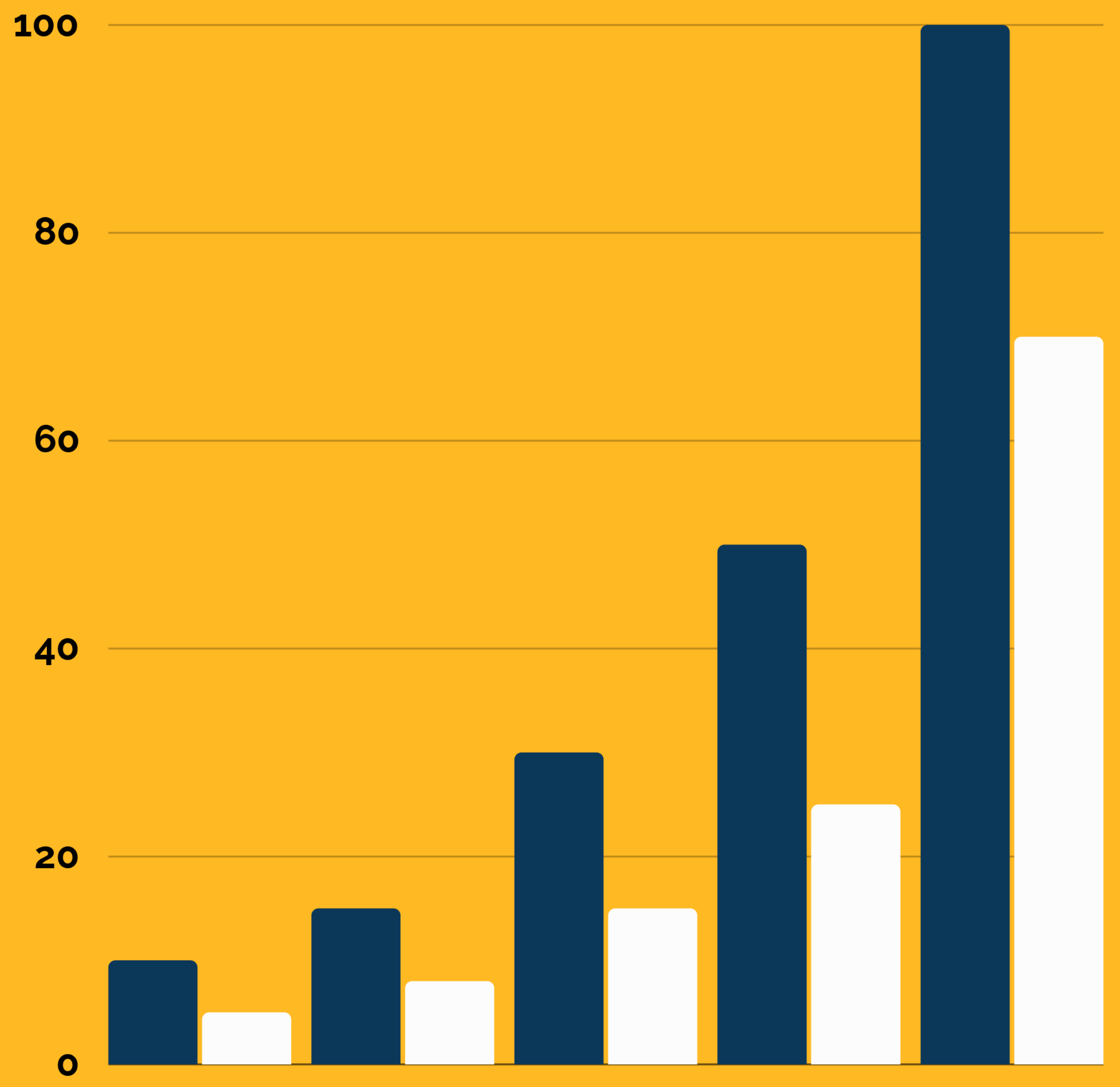


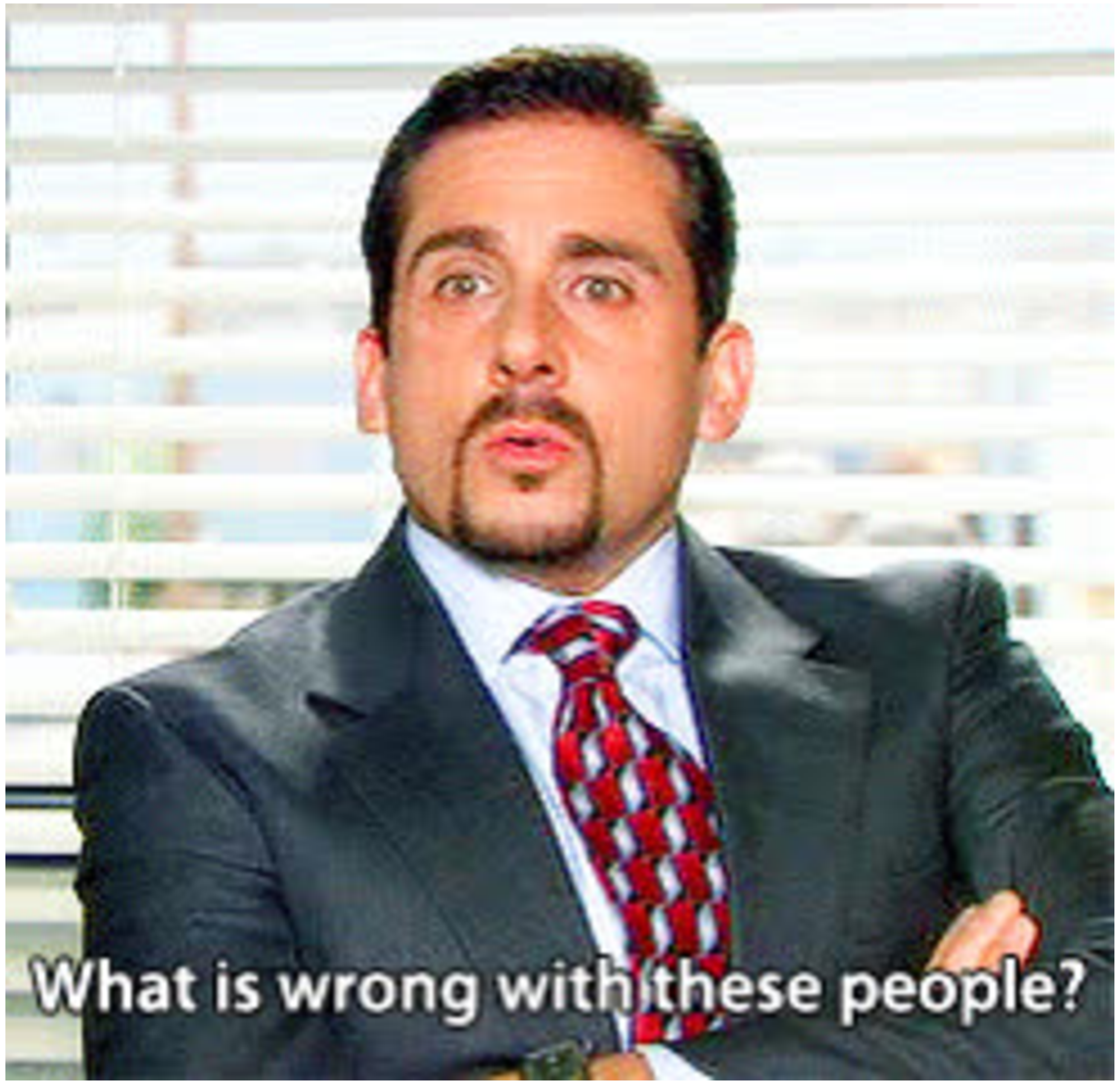


Challenges

- Poor User experience
- Business Impact
- Bad Developer Experience

Subscribers vs Incidents





- Lack of idempotency
- Multiple failure points
- Poor concurrency
- ....

Poor Design

Let’s Fix It !!

Let’s Be Real

Business Continuity

Limited Resources

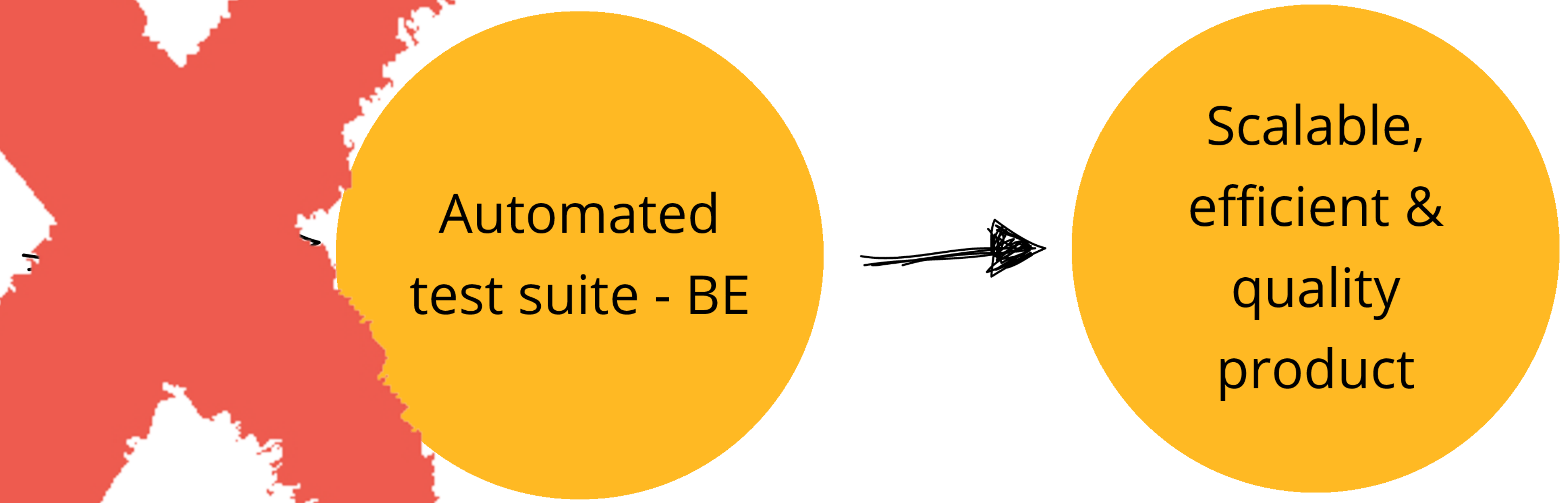
Timeline - Yesterday

Planning Principles

Ruthless Ruthless Ruthlessss Prioritisation

● ○ ○ ○ ○ ○ ○ ○ ○ ○

- Complex backend business logic
- Missing comprehensive test suite
- Time consuming manual testing


Automated test suite - BE
Scalable, efficient & quality product

Identify cut-off

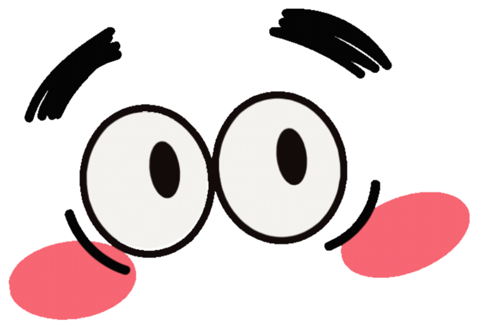
- Anything new on older system -> to be replicated in new system -> double the effort

- Find a cut off for features to be on the new system

Projects starting Feb 2024 to be implemented only on new system

Dedicated group

Team of 6 Backend Engineers,
2 Quality Engineers



10-20% capacity of entire team

Focused group of 2 BEs and 1 QA

Design, Execute & launch

Objectives

- Scalability
- Resiliency
- Idempotency

Temporal

- Workflow orchestration platform
- Simplify your code and build more features, faster
- At runtime, it is **fault-tolerant** and easily scalable so your apps can grow bigger, more **reliably**

Temporal vs Refactor

- Objectives
- Development Cost
- Cloud Cost
- Maintenance Cost

Refactor

●●○○

Temporal

●●●○ ✓

Timeline

Aug/Sept-23

Kick-off discussions and planning

Q4 23 ....Feb 24

- Design, Execute, Test plan
- Learn new things/missing cases
- Bake new learnings into code

Mar 24

- Test, Sign off, small production functional testing
- Prepare rollout plan
- Prepare runbooks

May 24

- Gradually Rollout to all countries
- Monitor issues
- Fix & rollout again

Things that did not go well

New Technology & learning curve

- **<u>Steep</u>** learning **<u>curve</u>**
- **<u>Lack</u>** of **<u>expertise</u>**
- **<u>Delays</u>** to the **<u>timeline</u>**

Allocate learning buffer & set up good collaboration forums

Lack of comprehensive test suite

- **Initial** time **investment**
- **Missed** edge **cases**
- Workload **pressure** on **QAs**

Conduct test war rooms for multiplied testing

New features on the new system took longer time

- **Full team** was **not onboarded** on time to the new system
- New feature **rollouts** got **delayed**
- **Lack of confidence** in the team to own & maintain new system

Prepare good onboarding plan for the team & stakeholder management

Inefficient Rollout Plan

- **Monitoring overhead** due to regional rollout plan
- Debugging & mitigation challenges for the **incoming issues** over **new** and **existing** systems

Simplify rollout plan with min. variables e.g plan per country full rollout

Low Morale & Motivation

- New technology & timeline delays caused **burn out** and **demotivation** in the team
- Feeling of **isolation** and **lack of belonging** to a bigger team

Continous support (hands-on) & celebrate problem solving

Thing(s) that went well

Commitment & Continuous problem solving as a TEAM

Results

1. Improved stability & scalability
2. Improved developer experience
3. Improved user experience

Reduced production incidents by > 85-90%

We made these mistakes - So you don’t need to.
Make your own and share 💪🏻

Thank You