Product Development

Câu hỏi suy ngẫm

Sau slide 1

Gắn với nội dung: Product Development

?

Nếu ứng dụng hiện chỉ có 500 người dùng/ngày, có nên xây Kubernetes + 20 microservices ngay không? Vì sao?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Content

- Product Development
  - Common Server Architecture
  - Scale Up
  - Scale Out
  - Microservices

Câu hỏi suy ngẫm

Sau slide 2

Gắn với nội dung: Content

? **“Hệ thống càng lớn càng phải dùng microservices” — câu này sai ở điểm nào?**

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Product Development

- Stack Technology
- Server Infrastructure
- Development Process
- Team

- *Concept: build "Independent Platform – Independent Domain"* ***Product***

Scale Up vs Scale Out

- Scale Up
  - Add-In hardware
  - Solve businesses logic
  - Optimize performance (need do everyday)
- Scale Out
  - Architecture
  - Add Instances
  - Load Balancing

Câu hỏi suy ngẫm

Sau slide 4

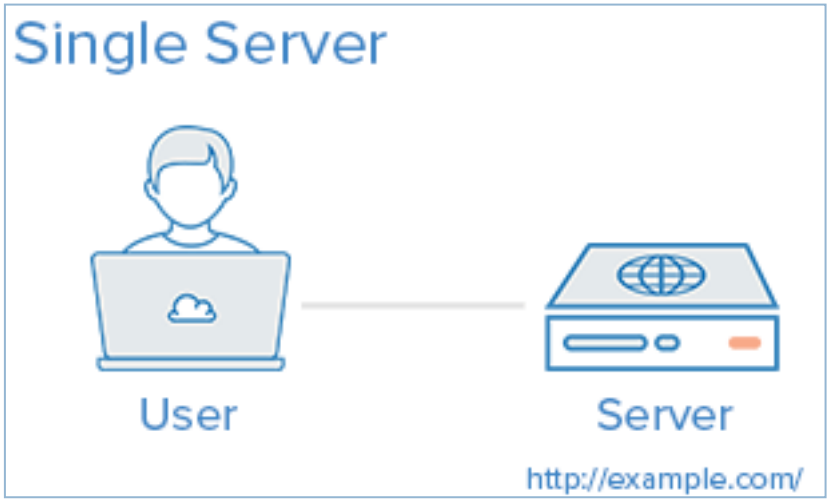
Gắn với nội dung: Scale Up Vs Scale Out

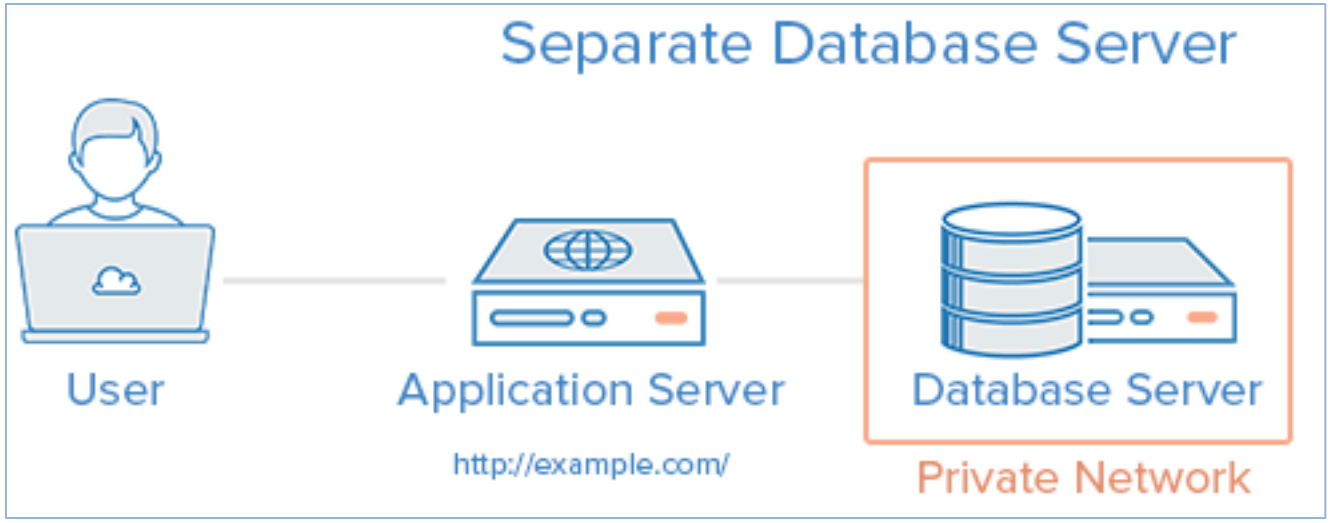
?

Vì sao application server thường scale-out dễ hơn relational database?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Common Server Architecture





Câu hỏi suy ngẫm

Sau slide 5

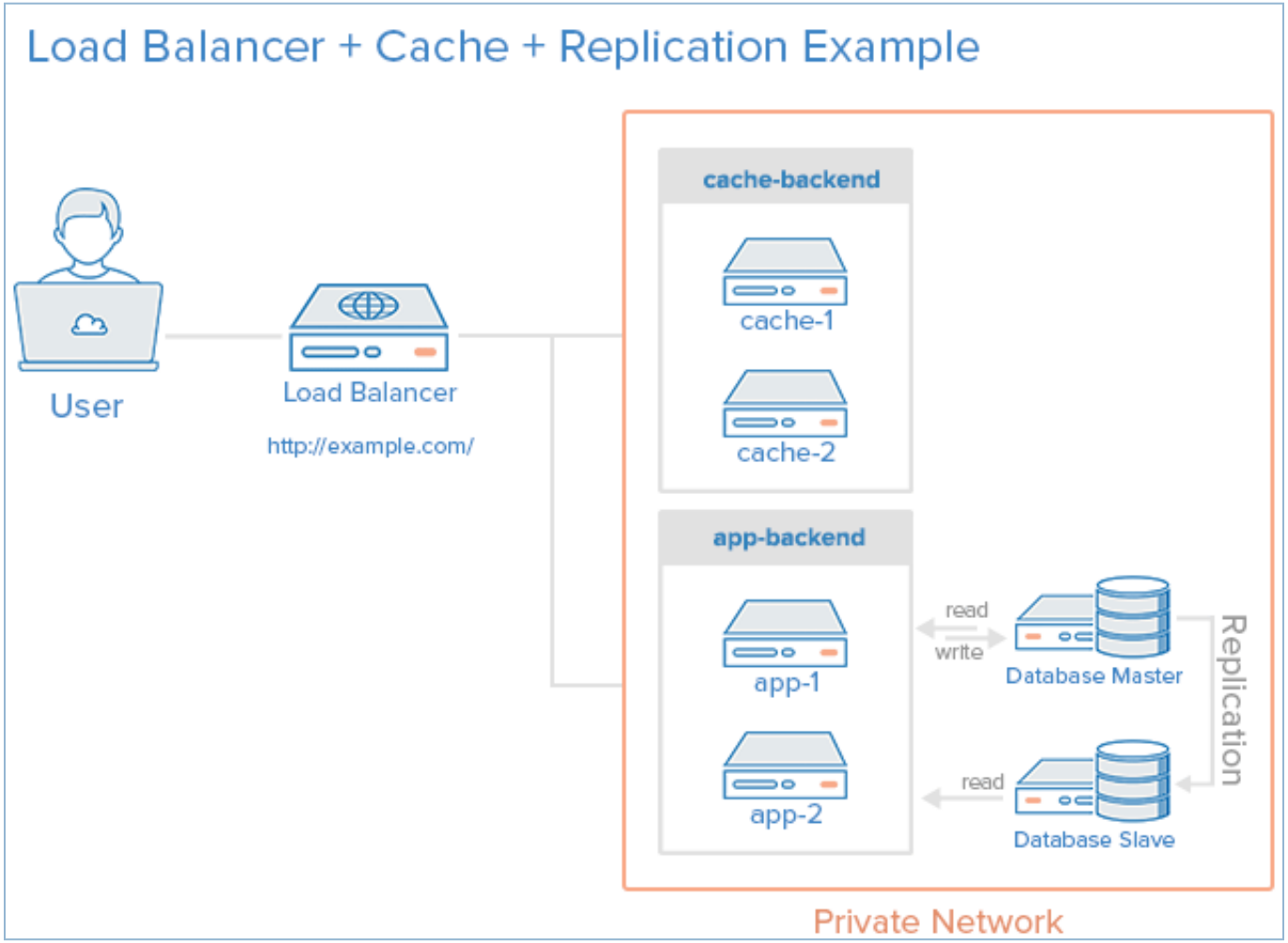
Gắn với nội dung: Common Server Architecture: Single Server Và Separate Database

?

Tách database sang máy riêng giúp reliability, nhưng tại sao database vẫn có thể là single point of failure?

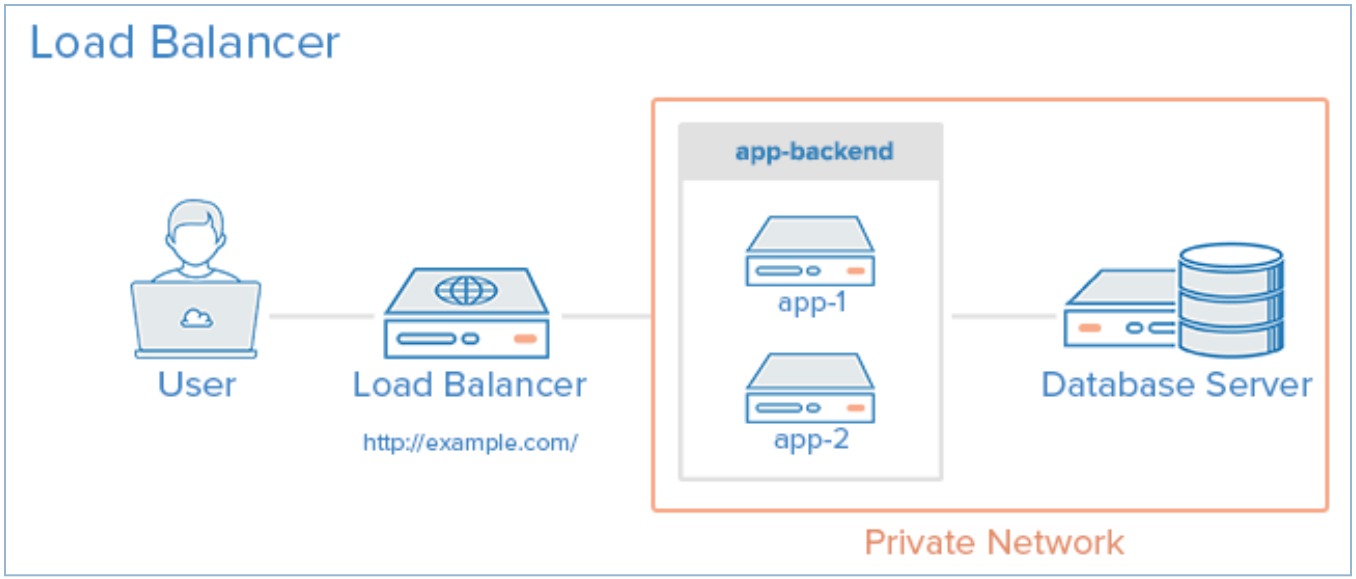
Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

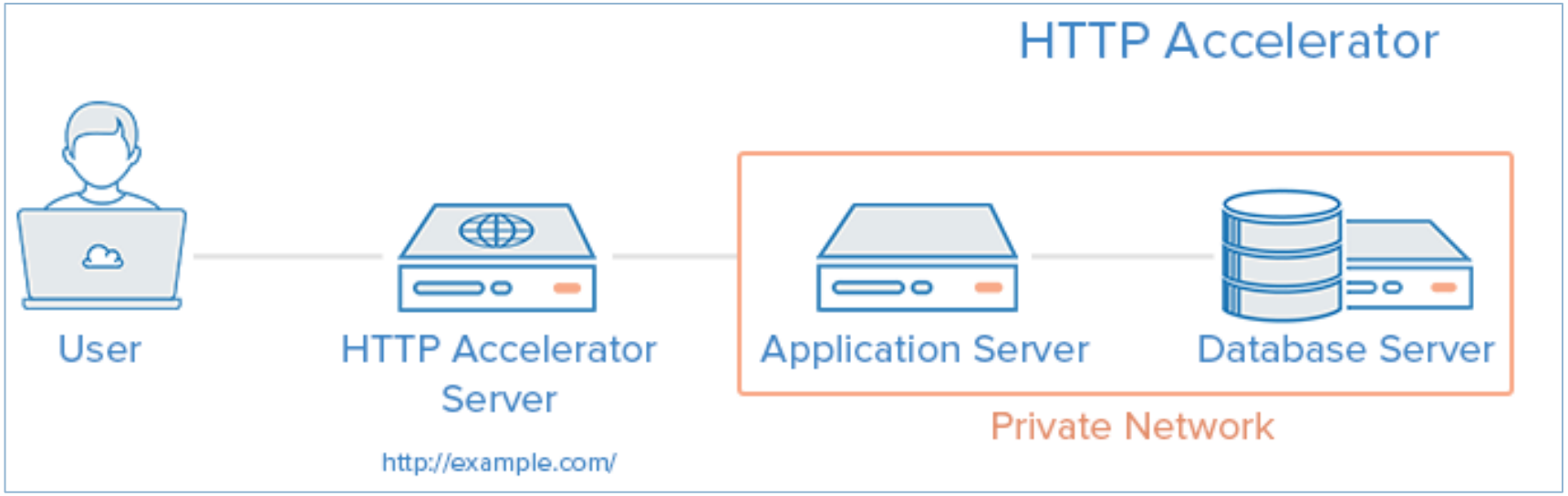
Common Server Architecture



Ref: https://viblo.asia/p/mot-so-kien-truc-server-pho-bien-cho-cac-ung-dung-web-GrLZDX1OZk0?

Common Server Architecture





Câu hỏi suy ngẫm

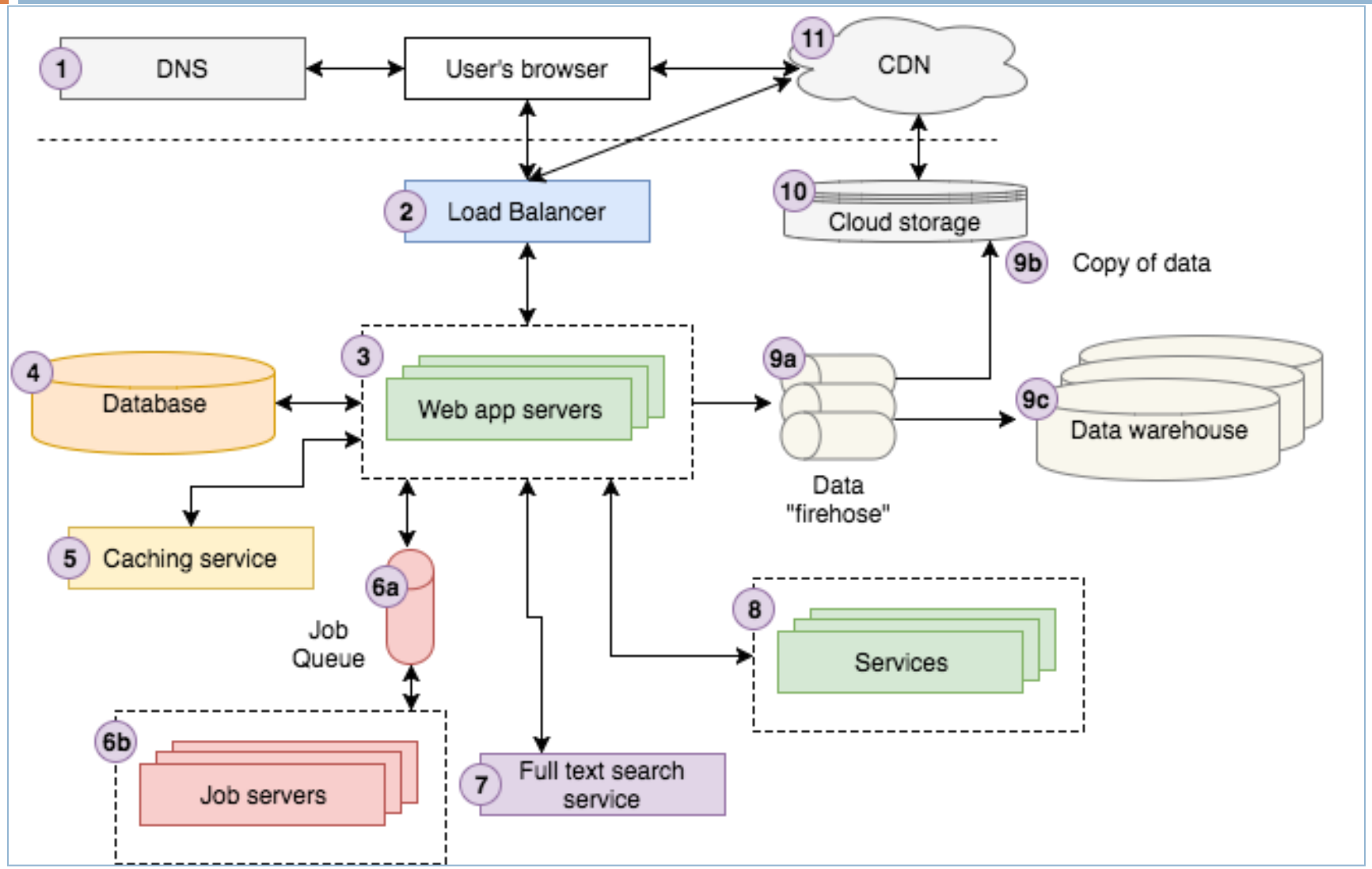
Sau slide 7

Gắn với nội dung: Load Balancer Và Http Accelerator

? **Load balancer và CDN đều đứng trước server. Khác biệt bản chất giữa hai thứ là gì?**

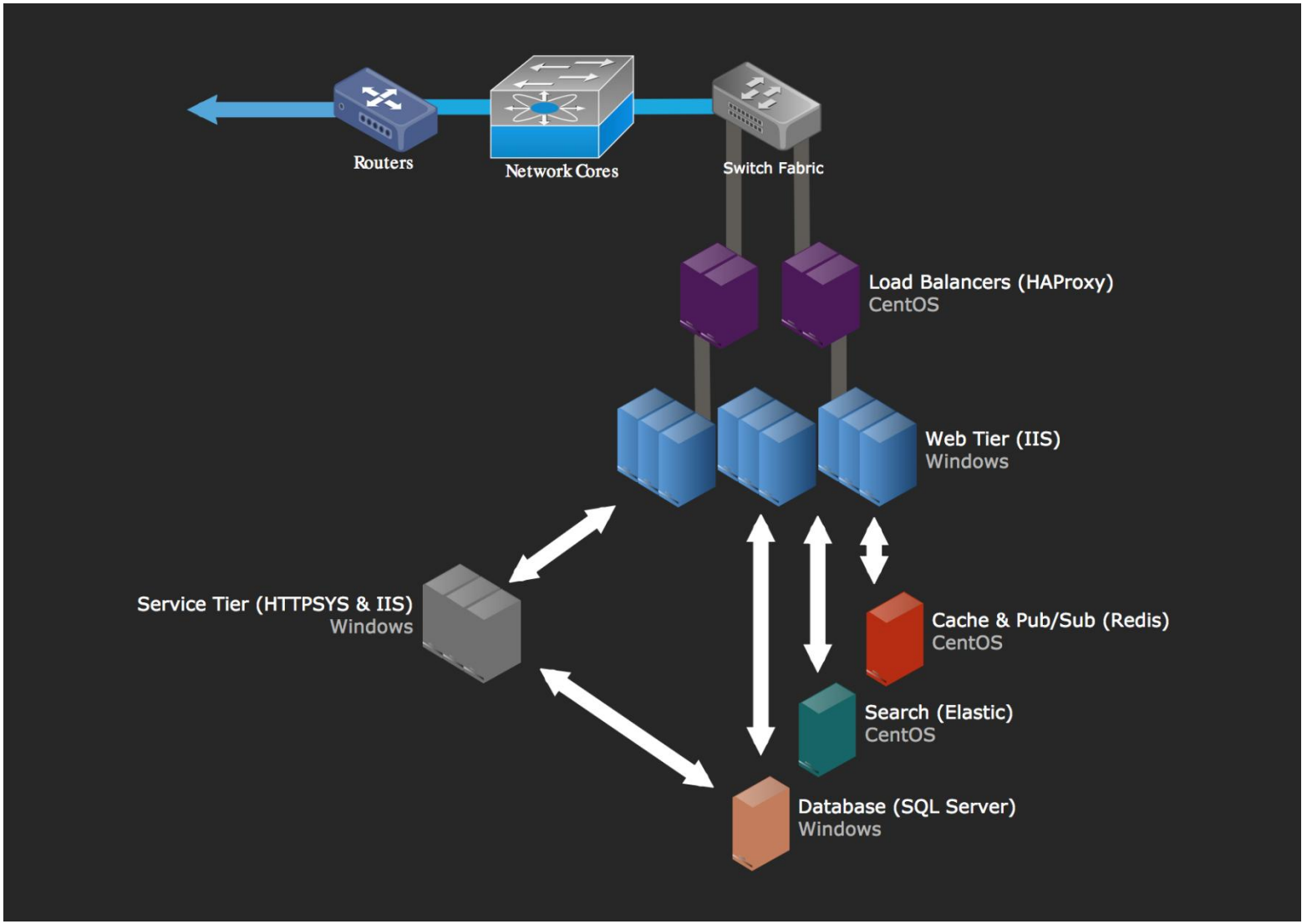
Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

StoryBlocks Web Architecture



https://viblo.asia/p/web-architecture-101-07LKXNB8lV4

Stackoverflow Web Architecture



https://techtalk.vn/bi-mat-stack-overflow-da-deploy-ra-sao-phien-ban-2016.html

Câu hỏi suy ngẫm

Sau slide 9

Gắn với nội dung: Stack Overflow Web Architecture

?

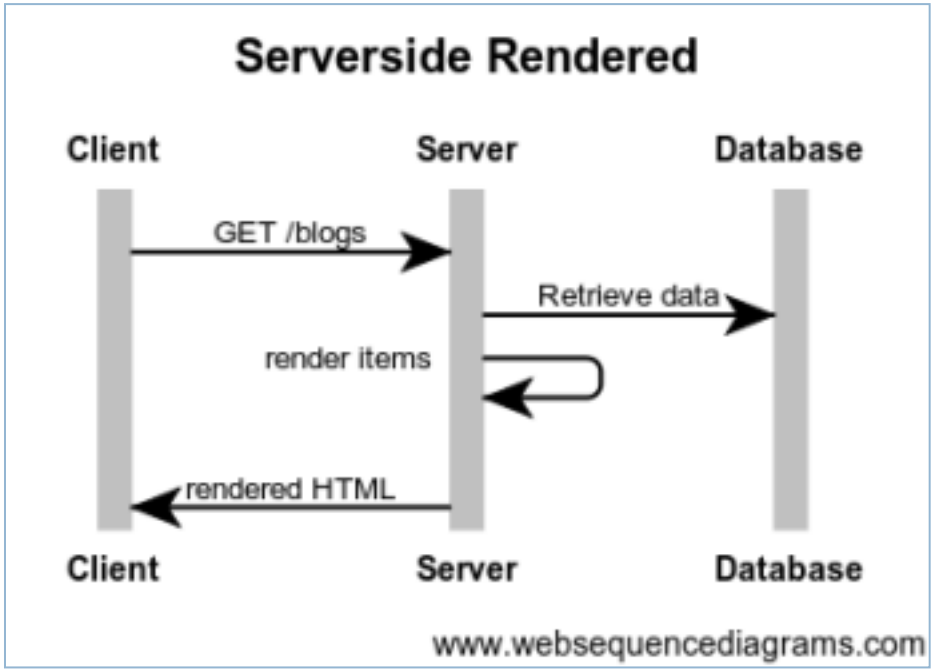
Case Stack Overflow chứng minh điều gì về câu “phải dùng microservices mới scale được”?

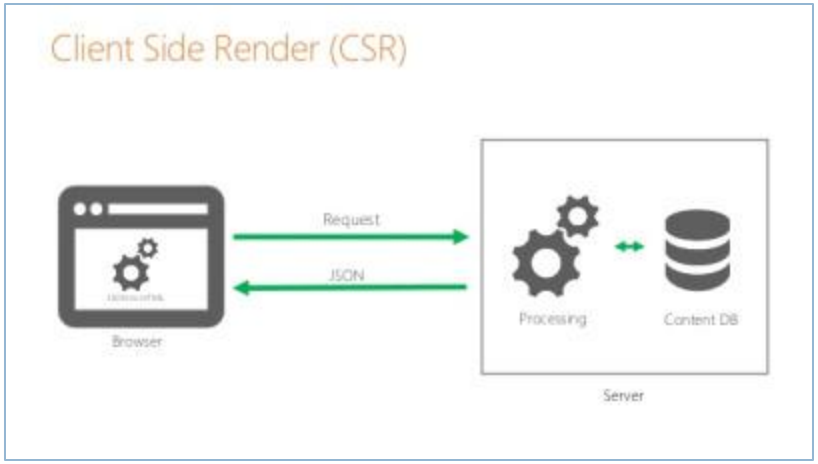
Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Stack Technology

- Client Problem
  - UI/UX
- Backend Problem
  - API
- Database Problem

Client Side vs Server Slide Rendering





Client Problem

- Google PageSpeed Insights - 100/100
- Five Principles
  - Less Requests
  - Small Resources
  - Near Resources
    - CDN
    - Proxy
  - Resource Priority
  - User Experiences

Client Problem

- Merge
  - Gulp
- Minify
- Compress
  - GZIP
  - Losses compression (Google PageSpeed Insights API)
- Cache
  - JS Caching
  - Local Storage
- Parallel
- Lazy & Prefetch

Backend problem

- Numbers everyone should know (http://highscalability.com/numbers-everyone-should-know)
- Text based vs Binary based ()
  - Set : Data/Object -> Serialize -> Compress -> Cache Service
  - Get : Data/Object <- Deserialize <- Decompress -< Cache Service
- Data serialize
  - Text Protocol
    - JSON - 167 bytes, se: 6ms, de: 3ms
    - XML - 235 bytes
  - Binary
    - Thrift - 64 bytes, se: 53ms, de: 1ms
    - Protobuf - 31 bytes, se: 34 ms, de: 4ms
    - Focus deserialize
  - Compress
    - Zstd (FB), Snapy, ZIP, GZIP
    - Focus decompress

https://auth0.com/blog/beating-json-performance-with-protobuf/

Backend problem

- Caching
  - Data Caching
    - Memory Cache
    - Redis
    - Database
  - HTTP Caching
    - Varnish
    - ETAG
- Query Data
  - ORM vs Store Procedures

Datababase Problem

- RDBMS vs NoSQL
  - File Group, Partitioning
  - Indexes, Execution Query Plan
  - Lock, Deadlock
- Failover, High Availability
- Data Warehouse (OLTP, OLAP)
- Microservices
  - *Replicate Master Data (Important)*

Câu hỏi suy ngẫm

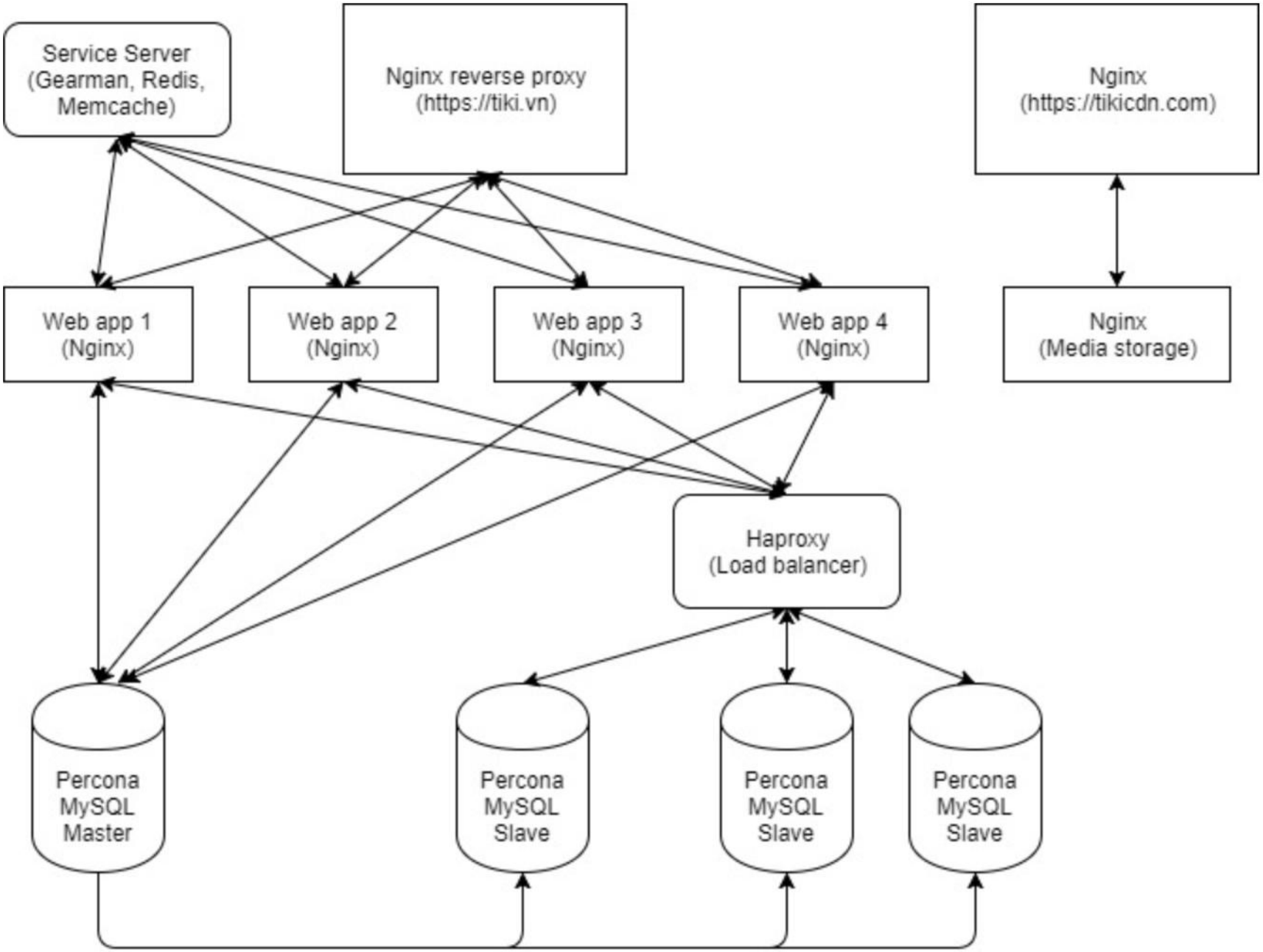
Sau slide 16

Gắn với nội dung: Database Problem

?

Thêm index luôn làm database nhanh hơn — đúng hay sai?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?



https://blog.chuyencuadev.com/2017-09-21-he-thong-tiki-3-nam-truoc-nhu-the-nao-p1

Câu hỏi suy ngẫm

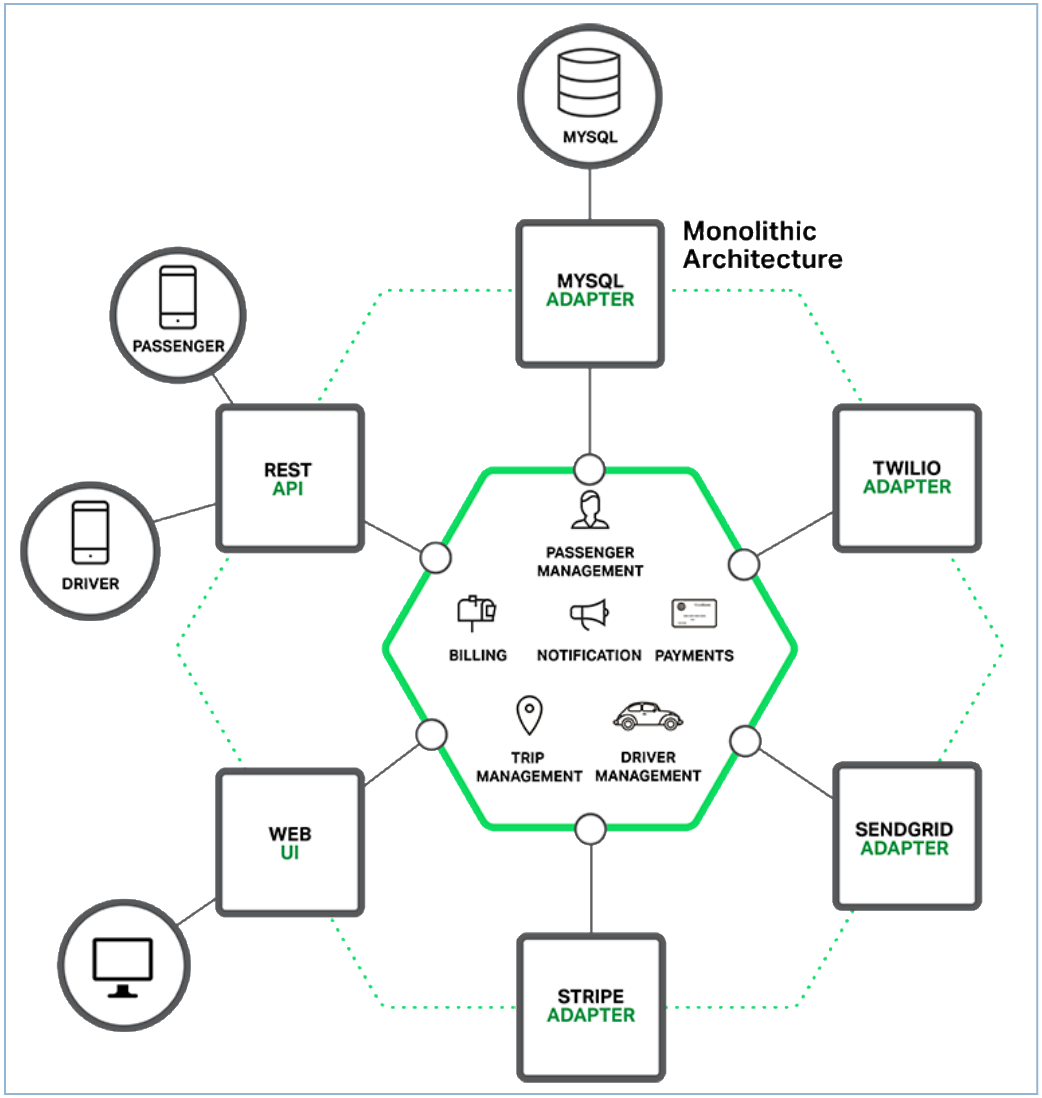
Sau slide 17

Gắn với nội dung: Database/Infrastructure Case: Tiki Lịch Sử

? **Vì sao replica rất tốt cho product catalog nhưng nguy hiểm hơn đối với balance tài khoản ngân hàng?**

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Monolithic vs Microservices



https://techmaster.vn/posts/33594/gioi-thieu-ve-microservices

Câu hỏi suy ngẫm

Sau slide 18

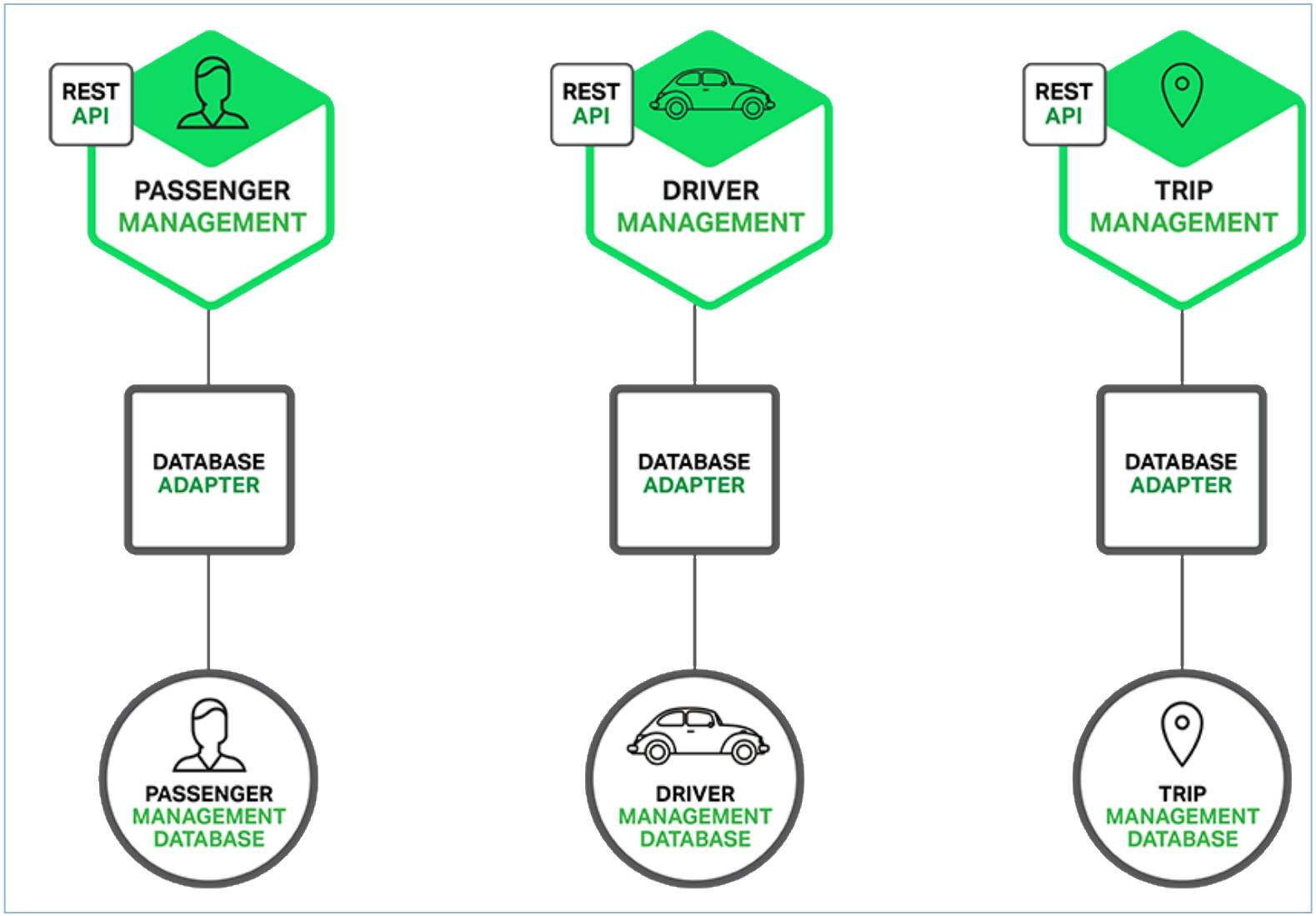
Gắn với nội dung: Monolithic Architecture

?

Một monolith được chia module tốt khác gì một “big ball of mud”?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Monolithic vs Microservices



Câu hỏi suy ngẫm

Sau slide 19

Gắn với nội dung: Microservices: Tách Theo Domain Và Data Ownership

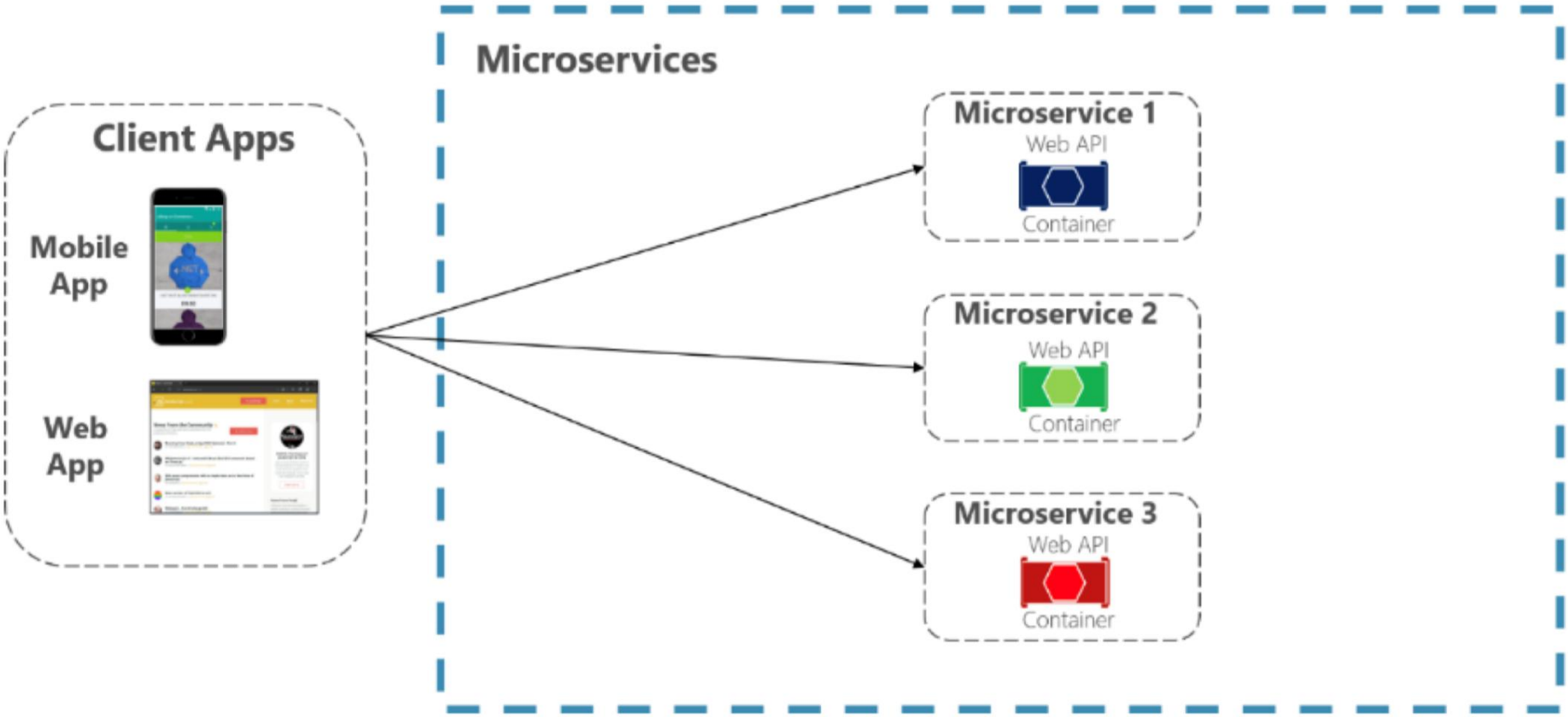
?

Hai microservices có source code tách riêng nhưng cùng sửa trực tiếp 30 table chung. Chúng có thực sự độc lập không?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Direct Client-To-Microservice communication

Architecture



https://docs.microsoft.com/en-us/dotnet/standard/microservices-architecture/

Câu hỏi suy ngẫm

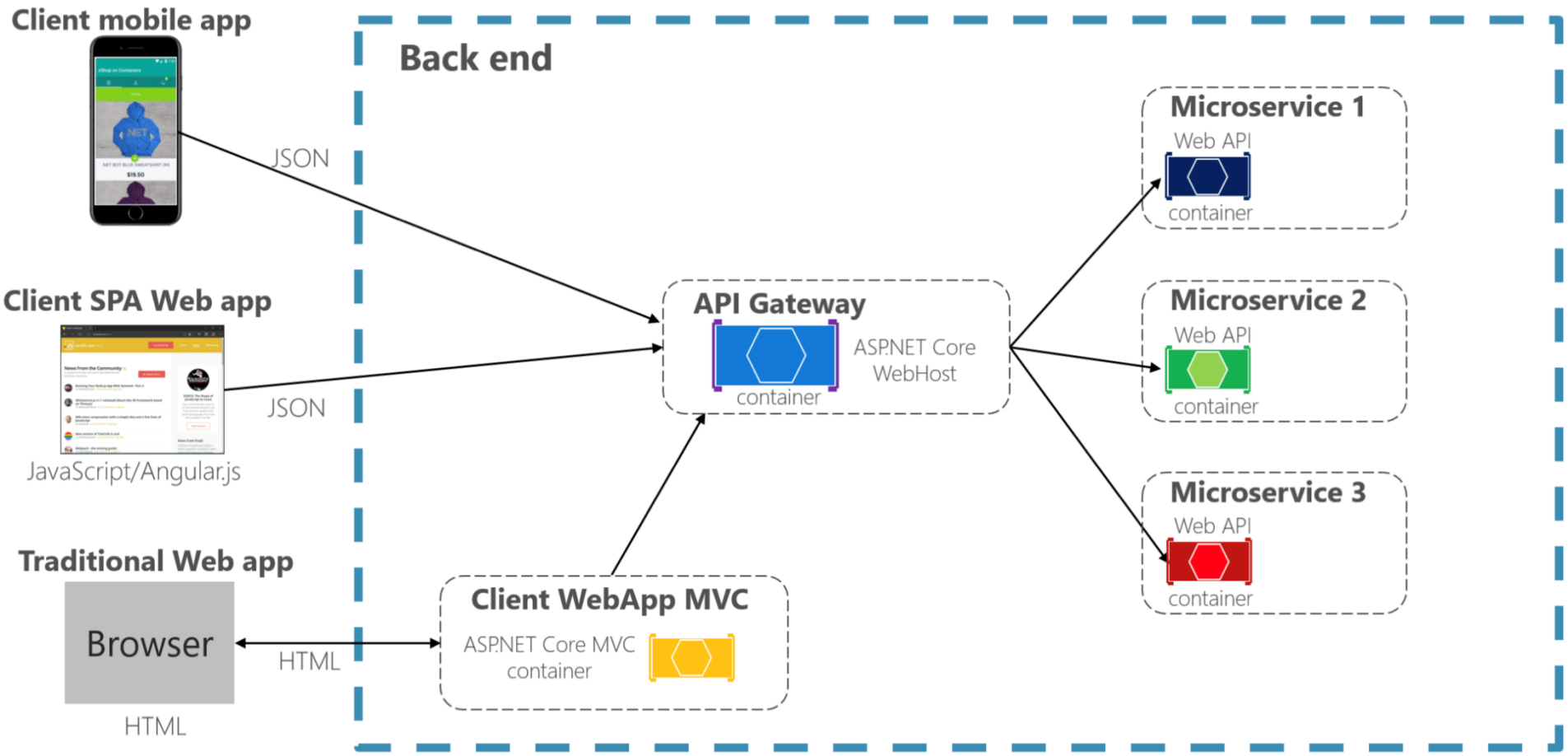
Sau slide 20

Gắn với nội dung: Direct Client-To-Microservice Communication

? **Vì sao browser biết trực tiếp địa chỉ của 50 microservices tạo coupling nguy hiểm?**

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Using a single custom **API Gateway service**



Câu hỏi suy ngẫm

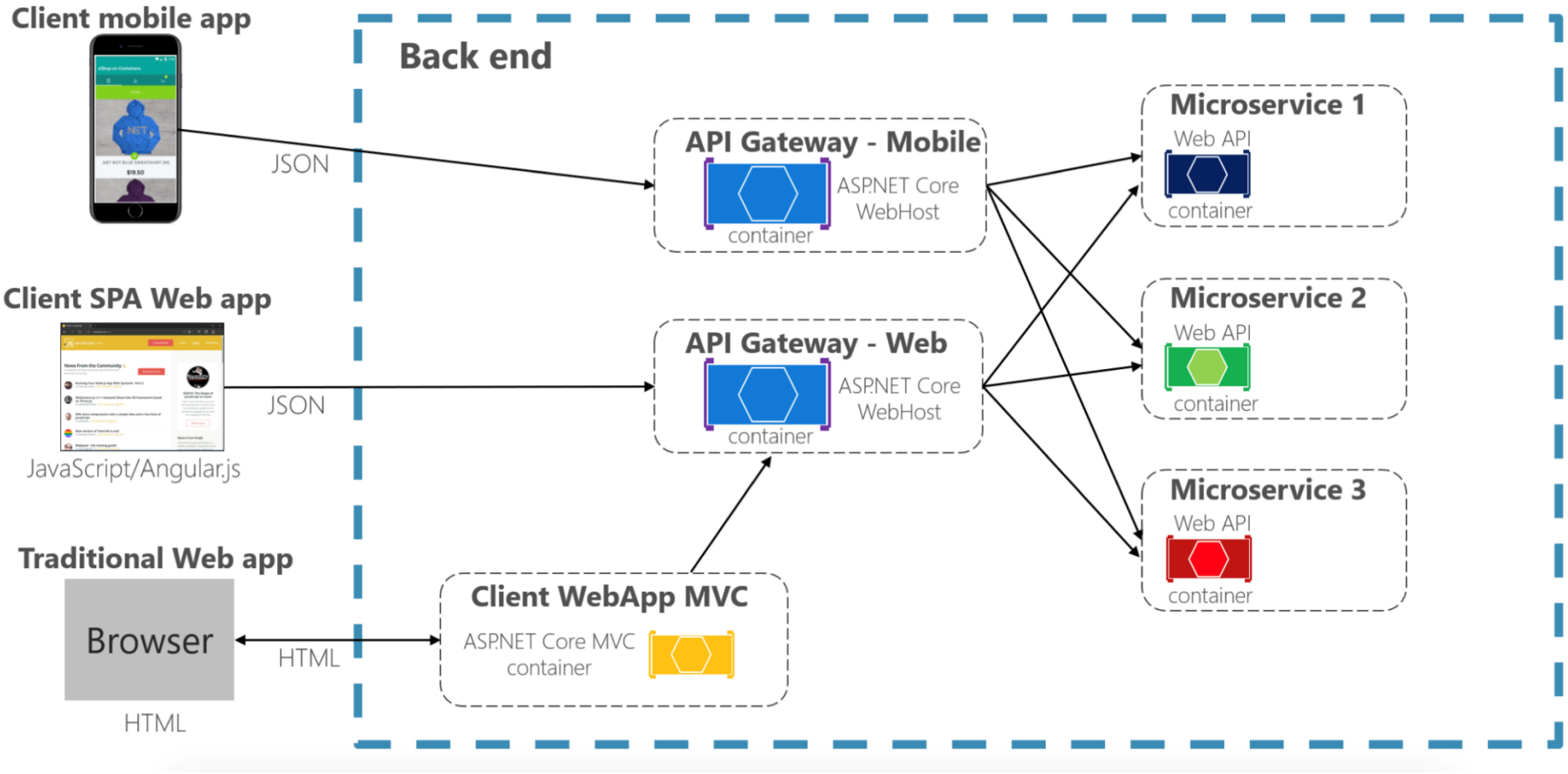
Sau slide 21

Gắn với nội dung: Single Api Gateway

? **Nếu toàn bộ business rules nằm trong API Gateway còn microservices chỉ là CRUD, kiến trúc đó có còn đúng tinh thần microservices không?**

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Using multiple **API Gateways / BFF**



Câu hỏi suy ngẫm

Sau slide 22

Gắn với nội dung: Multiple Api Gateways / Bff

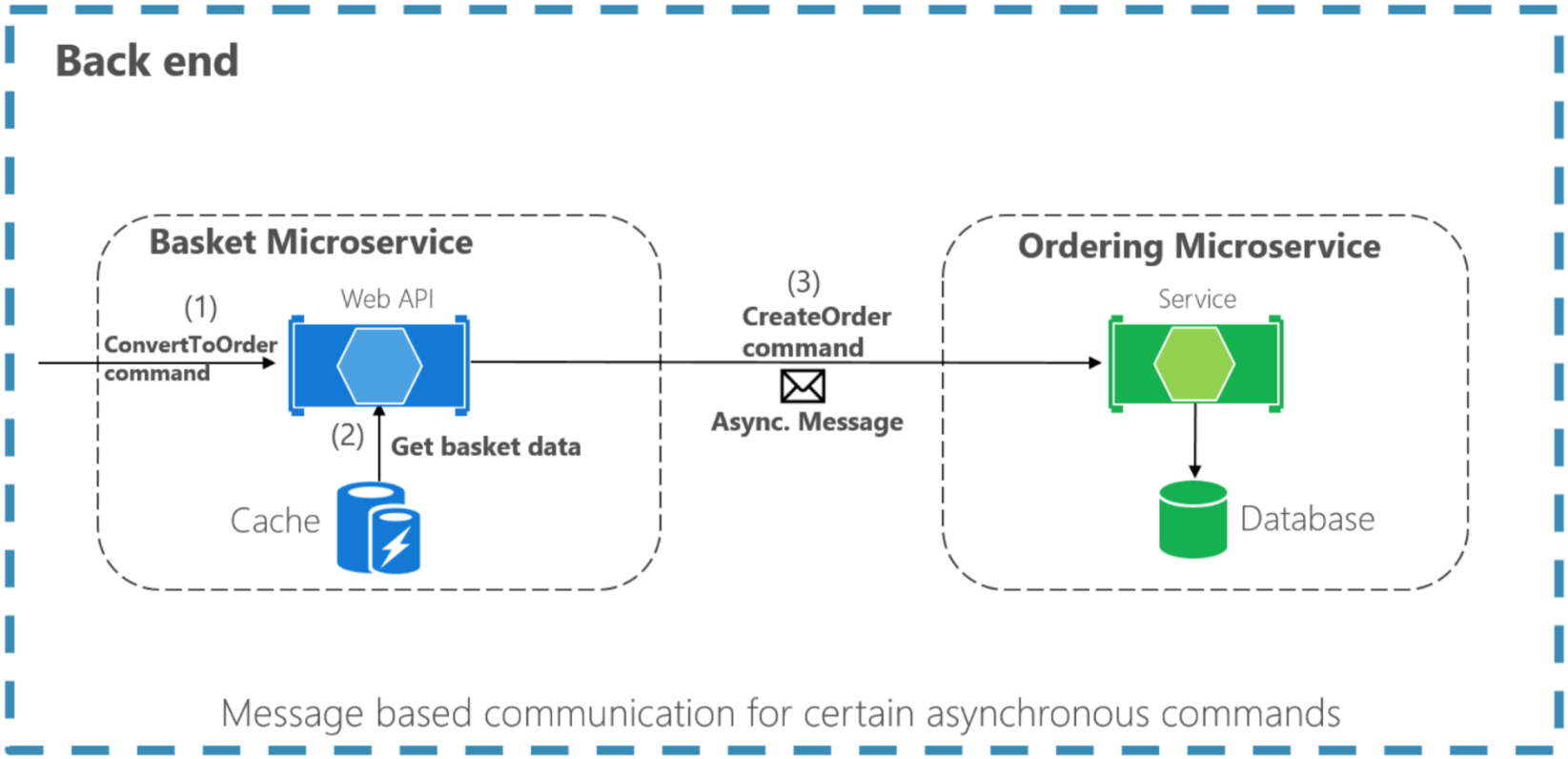
?

Tại sao mobile app và desktop web có thể cần hai API contract khác nhau dù dùng cùng business services?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Single receiver message-based communication

(i.e. Message-based Commands)



Câu hỏi suy ngẫm

Sau slide 23

Gắn với nội dung: Single Receiver Message-Based Communication

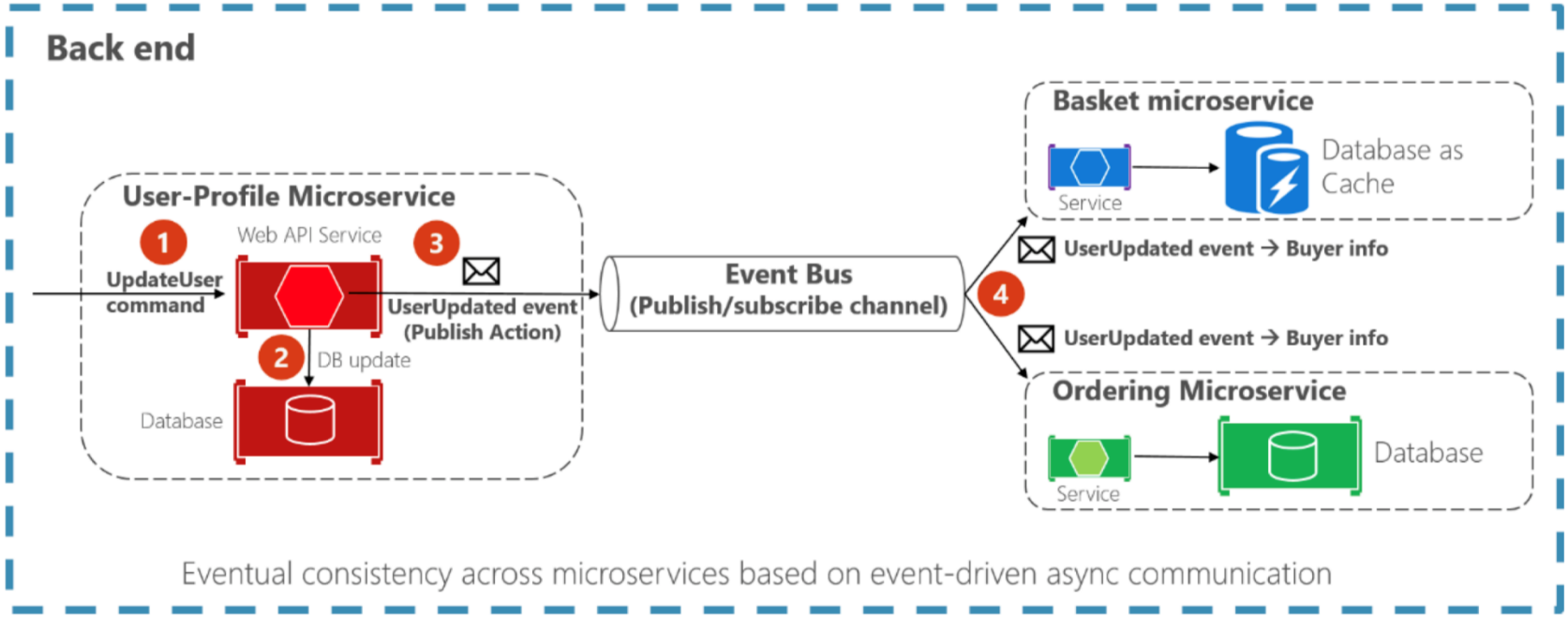
?

Nếu message broker giao cùng CreateOrder hai lần, hệ thống phải làm gì để không tạo hai đơn hàng?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Asynchronous event-driven communication

Multiple receivers



Câu hỏi suy ngẫm

Sau slide 24

Gắn với nội dung: Asynchronous Event-Driven Communication

?

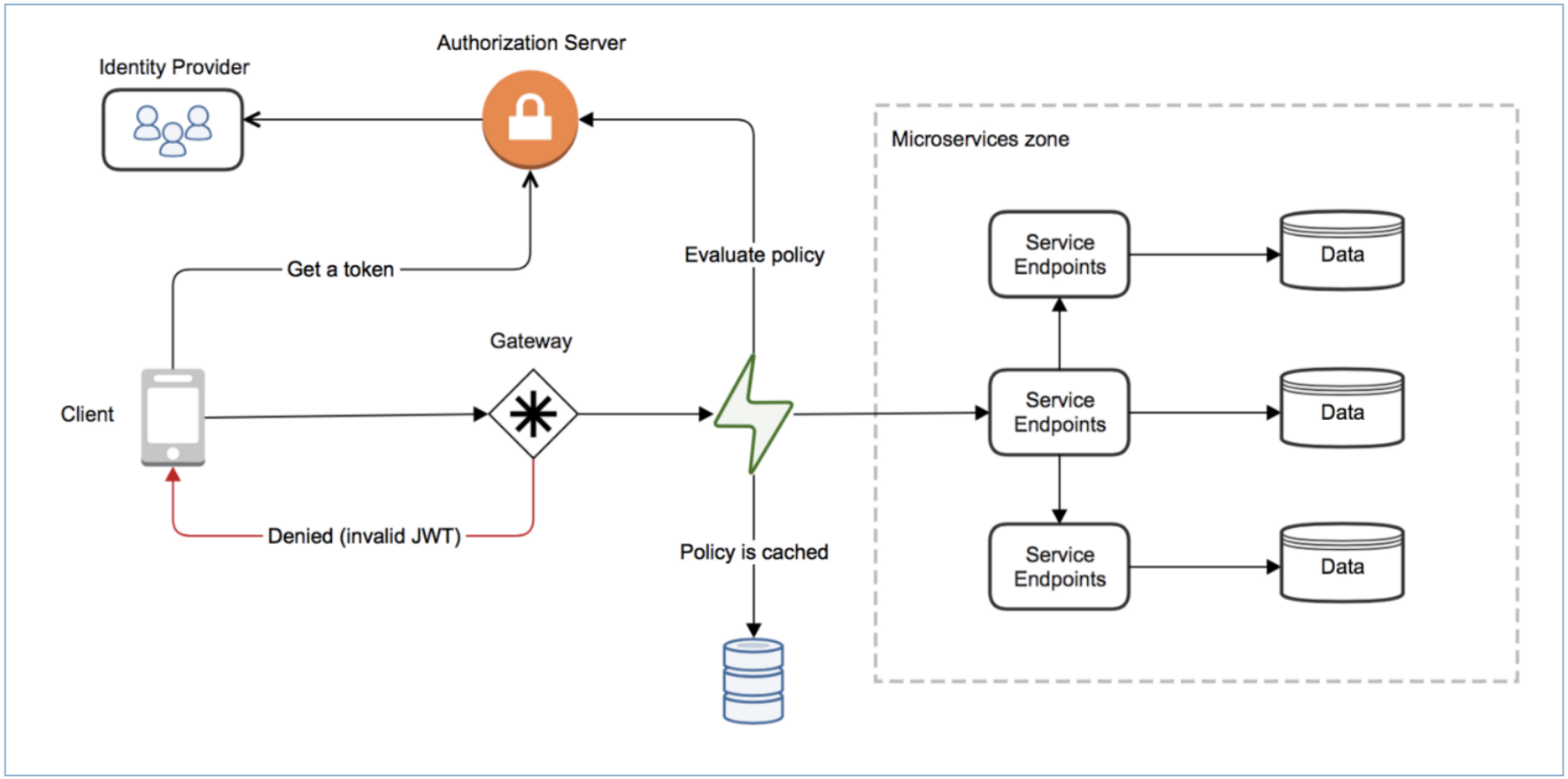
Event-driven làm giảm coupling nhưng tại sao lại làm debugging và consistency khó hơn?

Dừng 2 phút: viết câu trả lời ngắn theo góc nhìn Problem → Trade-off → Decision.
Trao đổi với bạn bên cạnh: nếu đưa vào hệ thống thật, rủi ro lớn nhất là gì?

Microservice

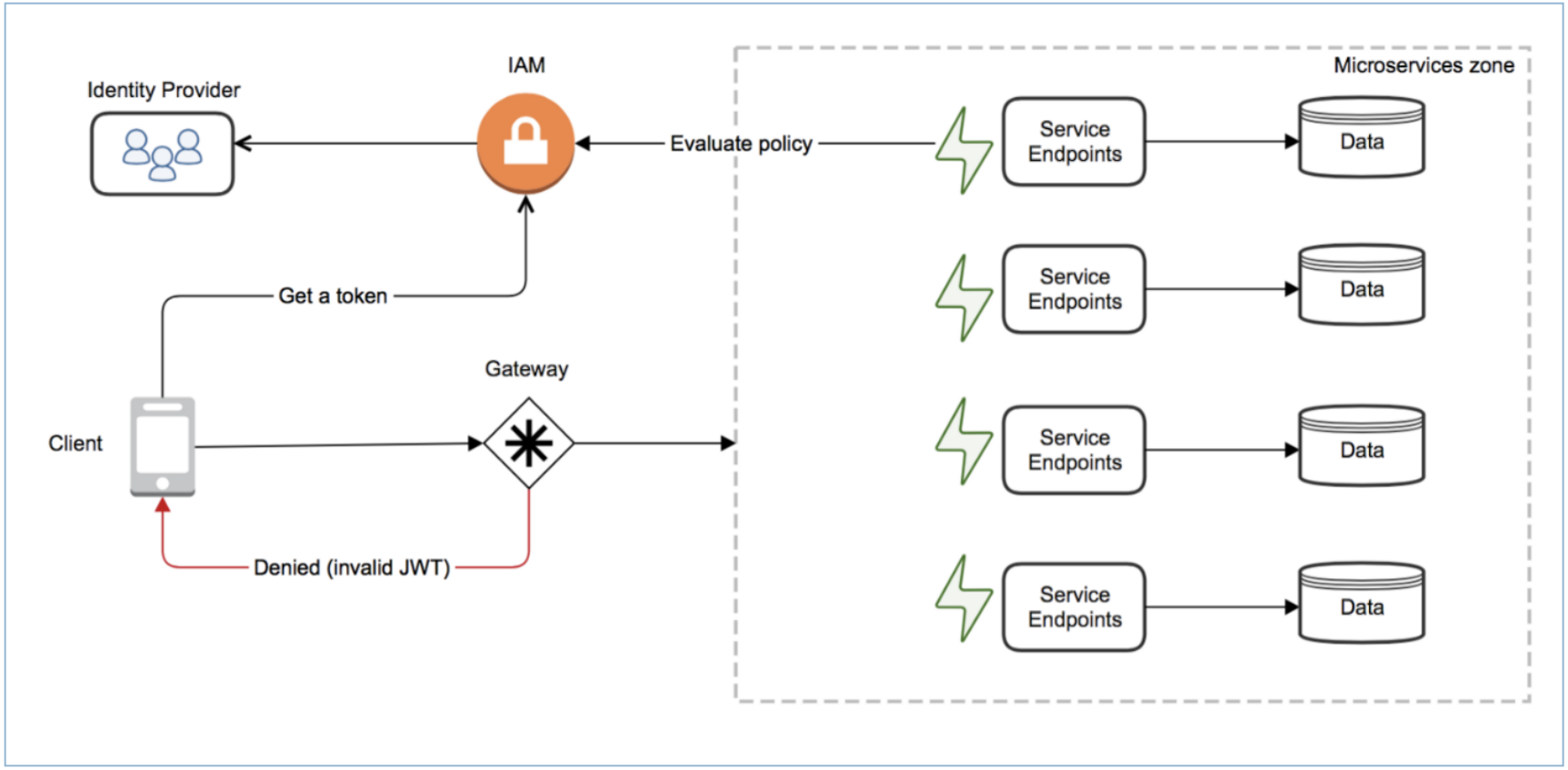
- Authentication & Authorization
  - JWT
  - OAuth2

Microservices

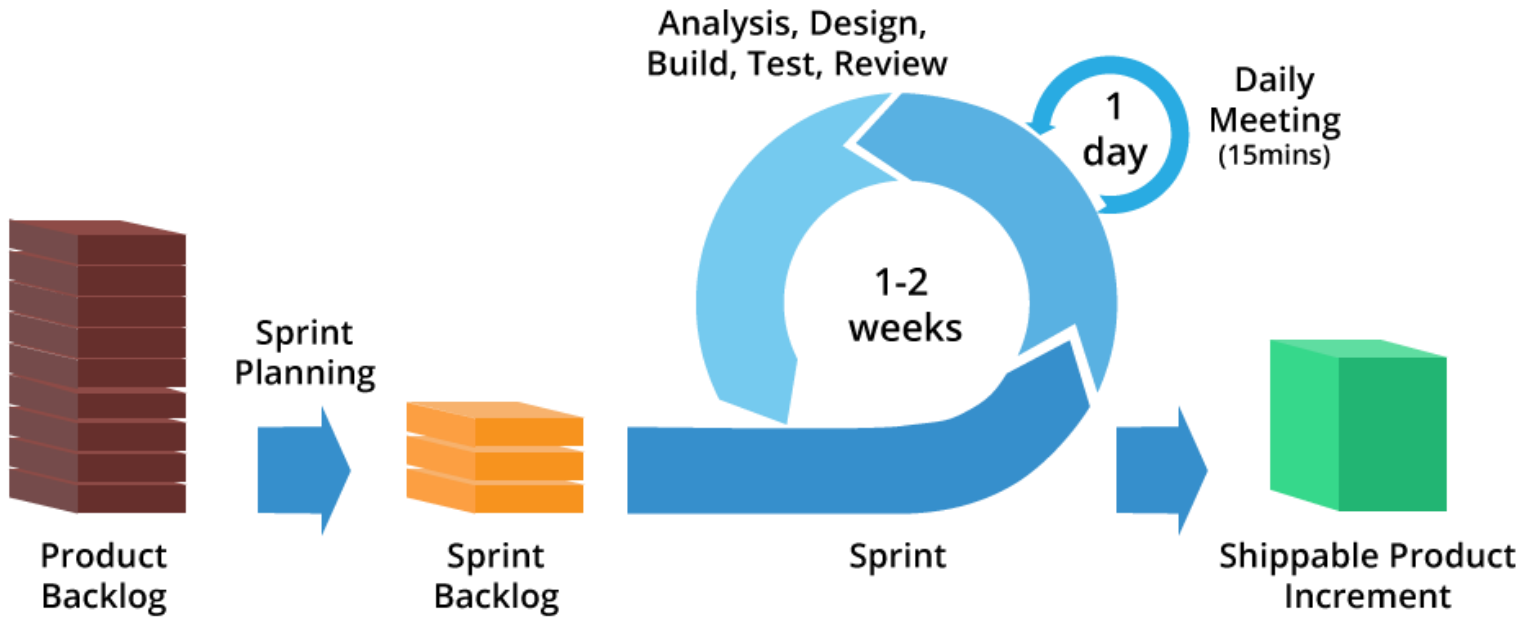


https://engineering.tiki.vn/x%C3%A1c-th%E1%BB%B1c-v%C3%A0-ph%C3%A2n-quy%E1%BB%81n-trong-microservices-37689e53c082

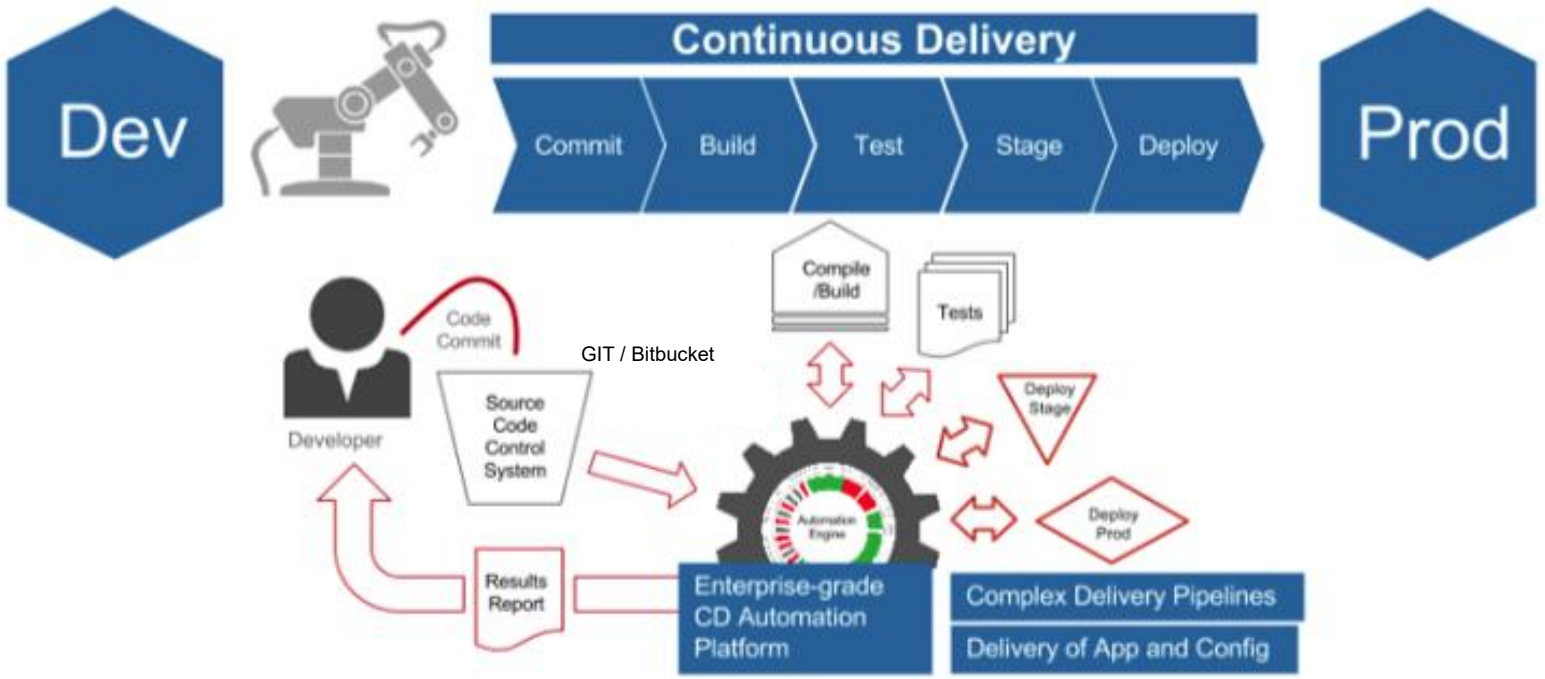
Microservices



Agile Software Development



CI/CD


Dev
Continuous Delivery
Commit
Build
Test
Stage
Deploy
Prod
Compile /Build
Tests
Code Commit
GIT / Bitbucket
Developer
Source Code Control System
Deploy Stage
Deploy Prod
Automation Engine
Results Report
Enterprise-grade CD Automation Platform
Complex Delivery Pipelines
Delivery of App and Config

Discussion