VIETNAM WEB SUMMIT mytour.vn

# Building next generation websites with serverless and micro-frontend architecture

Tuan Anh Tran

Engineering Manager @ Mytour Vietnam

anhtt@mytour.vn / github.com/tuananh / tuananh.org

# Serverless

- AWS Lambda
- Google Cloud Functions
- Azure Functions
- CloudFlare Workers
- IBM OpenWhisk
- Alibaba Function Compute
- OpenFaas, kubeless
- etc…

Amazon
Lambda

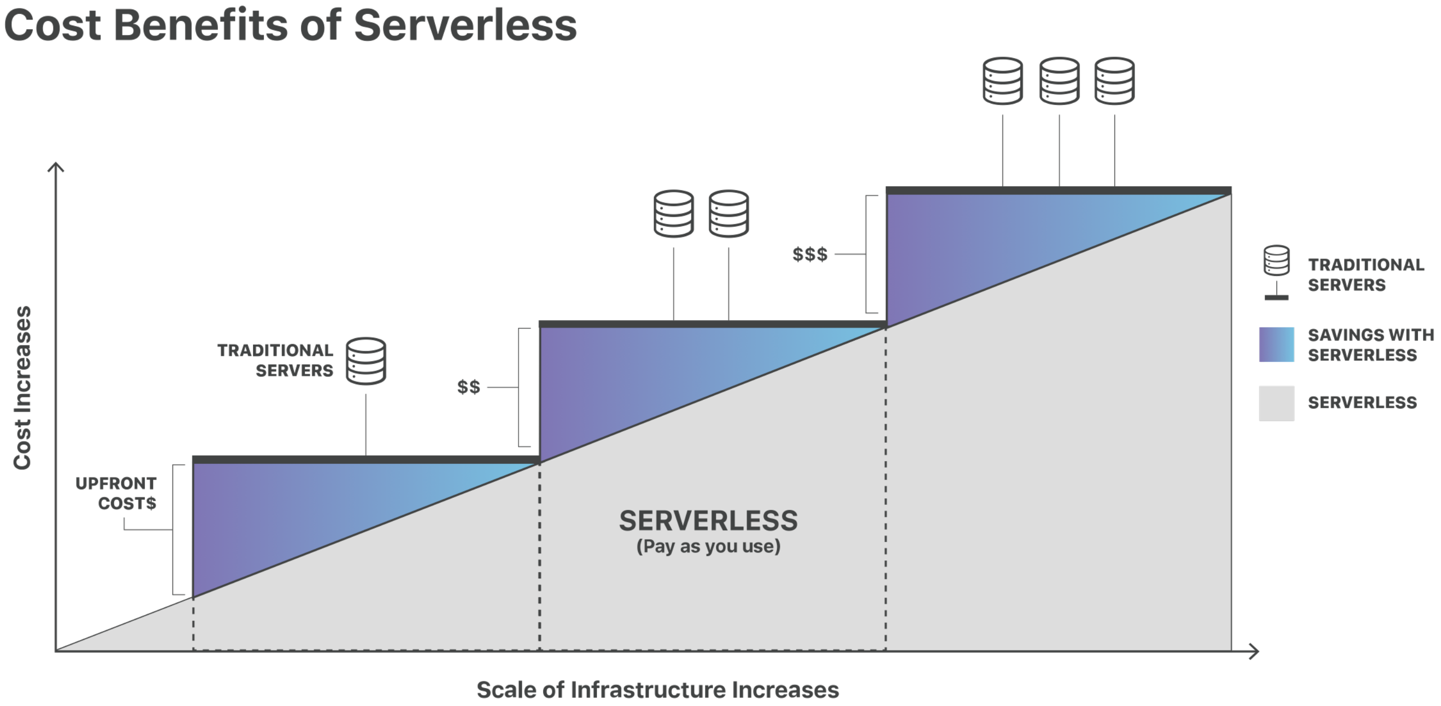
Azure Functions

CLOUDFLARE®
Workers

# What is serverless exactly?

- An infrastructure / backend / function service on an as-used basis.
- **Promise** of allowing user to write+deploy code without worrying about underlying infrastructure.
  - Not worrying about physical servers.
  - Not worrying about scaling.
  - Not worrying about calculating #. of servers required.
  - Cost saving
  - etc..

# Cost saving benefits



*Credit: CloudFlare*

# Other benefits

- Simplificied scalability
- Simplified backend code
- Quicker turnaround

# Micro-frontend architecture

- Microservice?

- Breaking monolith frontend to micro-frontend

# Benefits of micro-frontend

- Smaller, more maintainable codebases.
- More scalable organizations with decoupled and autonomous team.
- Development velocity: update/upgrade/rewrite part of frontend in more incremental fashion.

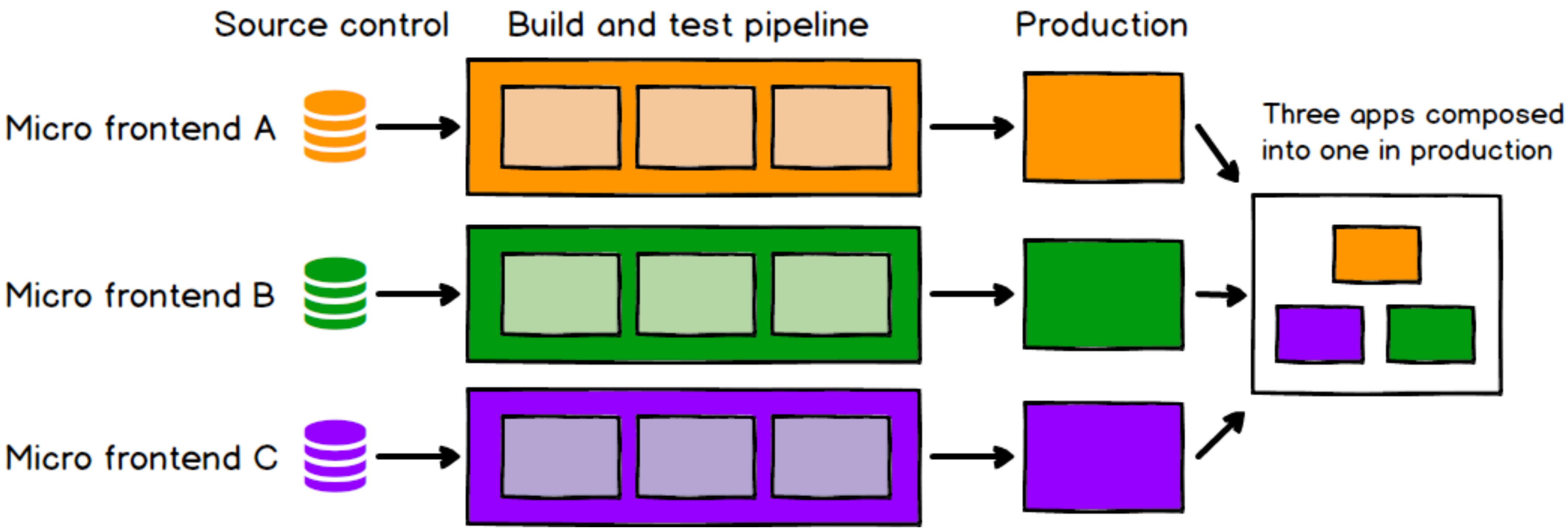
# Benefits of micro-**frontend**

- Smaller, more maintainable codebases.
- More scalable organizations with decoupled and autonomous team.
- Development velocity: update/upgrade/rewrite part of **frontend** in more incremental fashion.

# Benefits of micro-**service**

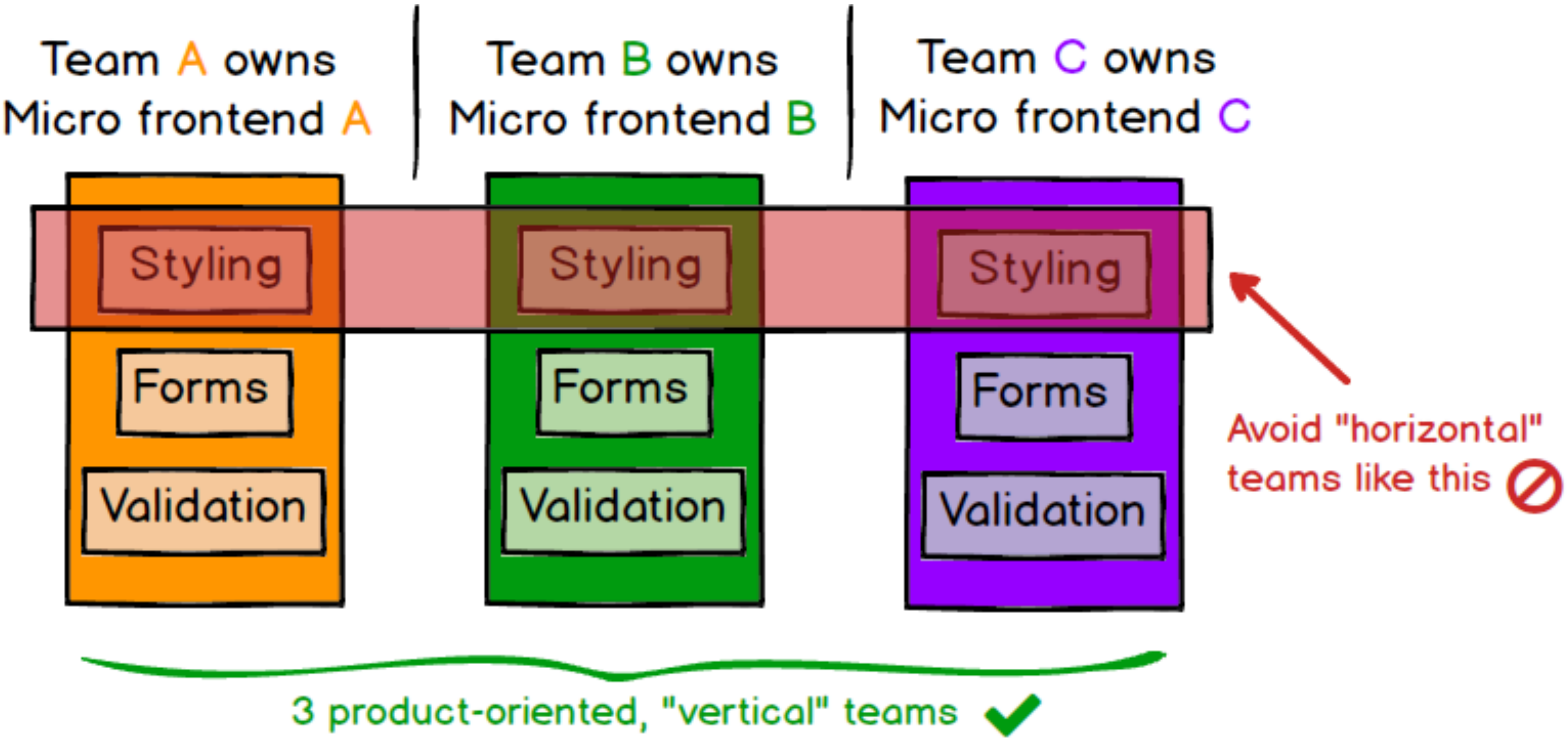
- Smaller, more maintainable codebases.
- More scalable organizations with decoupled and autonomous team.
- Development velocity: update/upgrade/rewrite part of **backend** in more incremental fashion.

# Benefits of micro-frontend (2)



*Credit: Martin Fowler’s blog*

# Benefits of micro-frontend (3)



*Credit: Martin Fowler's blog*

# Integration Approaches

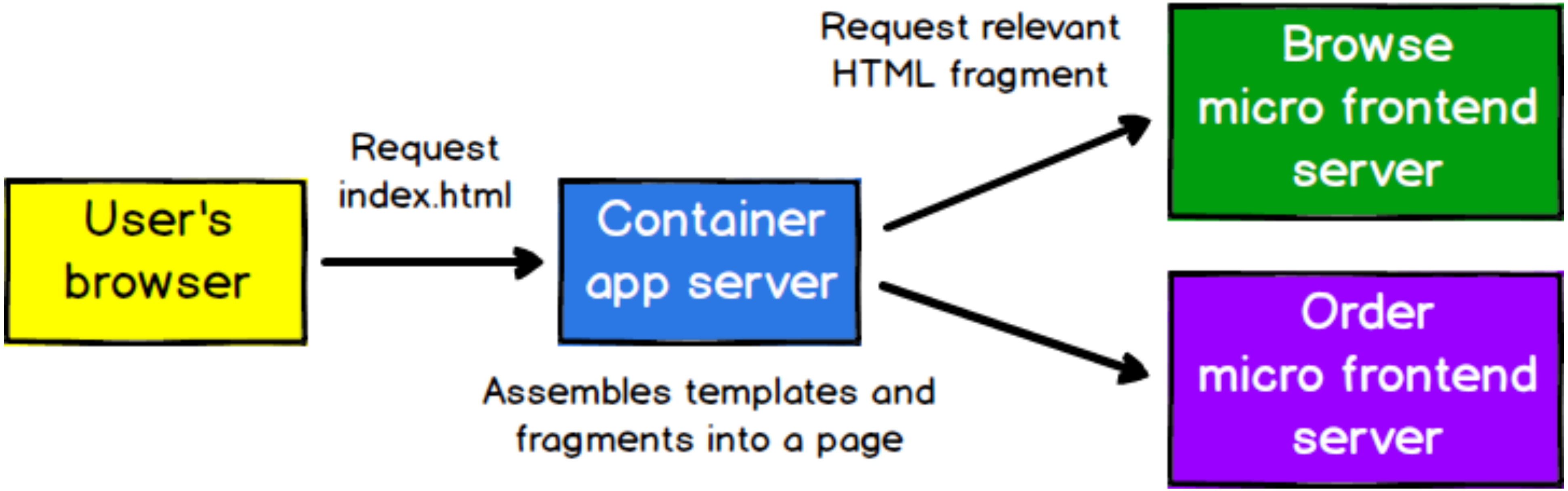
1. Server-side template composition

2. Build time integration 🏗️

3. Iframes 😂

4. Runtime integration via JavaScript or WebComponent

# Server-side template composition


User's browser
Request index.html
Container app server
Assembles templates and fragments into a page
Request relevant HTML fragment
Browse micro frontend server
Order micro frontend server

*Credit: Martin Fowler's blog*

# Build time integration

- Publish each micro-frontend component as package.
- Container app will include them as dependencies.
- => We need to trigger rebuild on every release 😱

# Iframes

- Easy isolation 💪
- Responsive nightmare 😭
- Less flexible: routing, history, deep linking

# Runtime integration via JavaScript

- Most flexible and widely adopted approach.
- Independent deploy, unlike build time integration.
- Pretty good isolation (style isolation via css-in-js)

# Style isolation

- Comes in default with iframe
- For other approach, we have CSS-in-JS or ShadowDOM.
- It’s an already solved problem.

# Conclusion

- Embrace microservice concept on frontend side.
- Micro-frontend can speed up development velocity a lot.
- Simplify the tech stack if you’re already using serverless tech in production.
- Start by building non-critical services first and gradually increase.

# Questions?