



Modernize without Madness

Hareesh Iyer, Sr. Solutions Architect, AWS

Agenda

- What are Microservices?
- Are Microservices better than a Monoliths?
- How do we move from a Monolith to Microservices?

What are Microservices?

Microservices are small, autonomous services that work together

- Sam Newman, Author – Principles of Microservices

Small enough that a single feature team can build, test, and deploy it

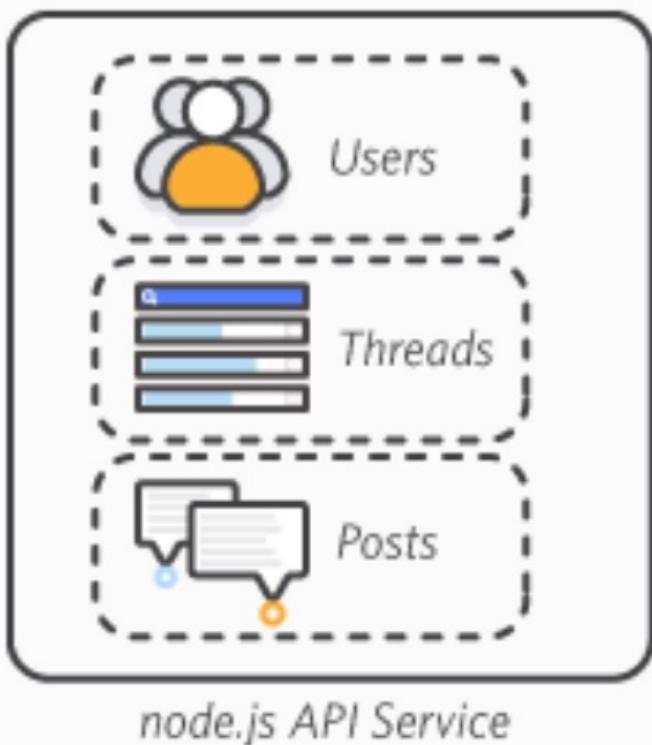
Autonomous: self-contained, independently deployable services, that don't share data

Work Together through network calls; loosely coupled

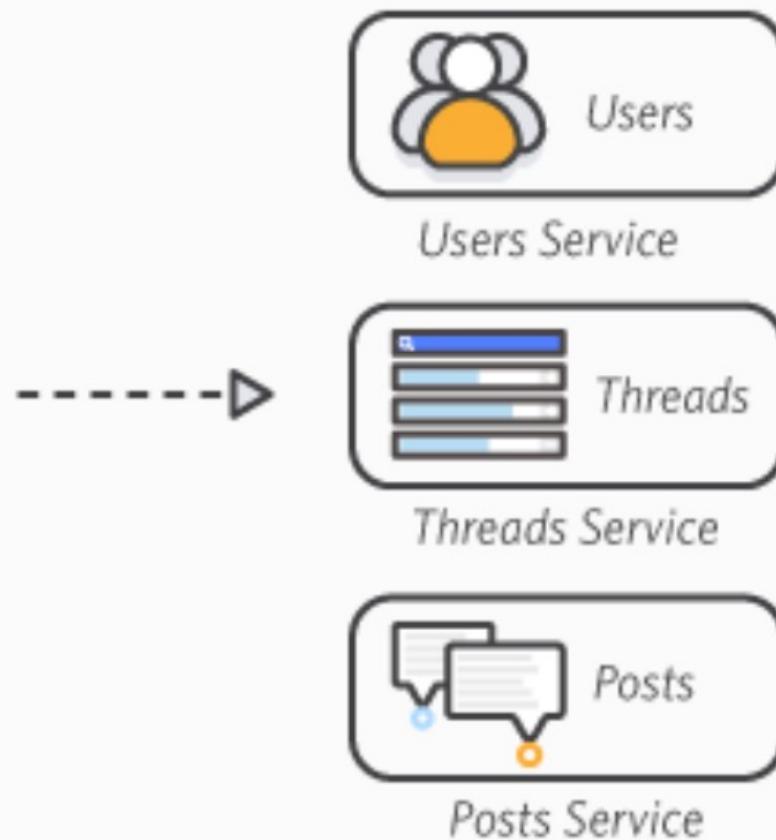
Microservices are small, autonomous services that work together

- Sam Newman, Author – Principles of Microservices

1. MONOLITH



2. MICROSERVICES



Are Microservices better than a Monoliths?



Microservices vs Monolith:

It depends on your goals and challenges

Microservices can enhance business agility

Microservices can improve application scalability

Microservices can improve fault tolerance

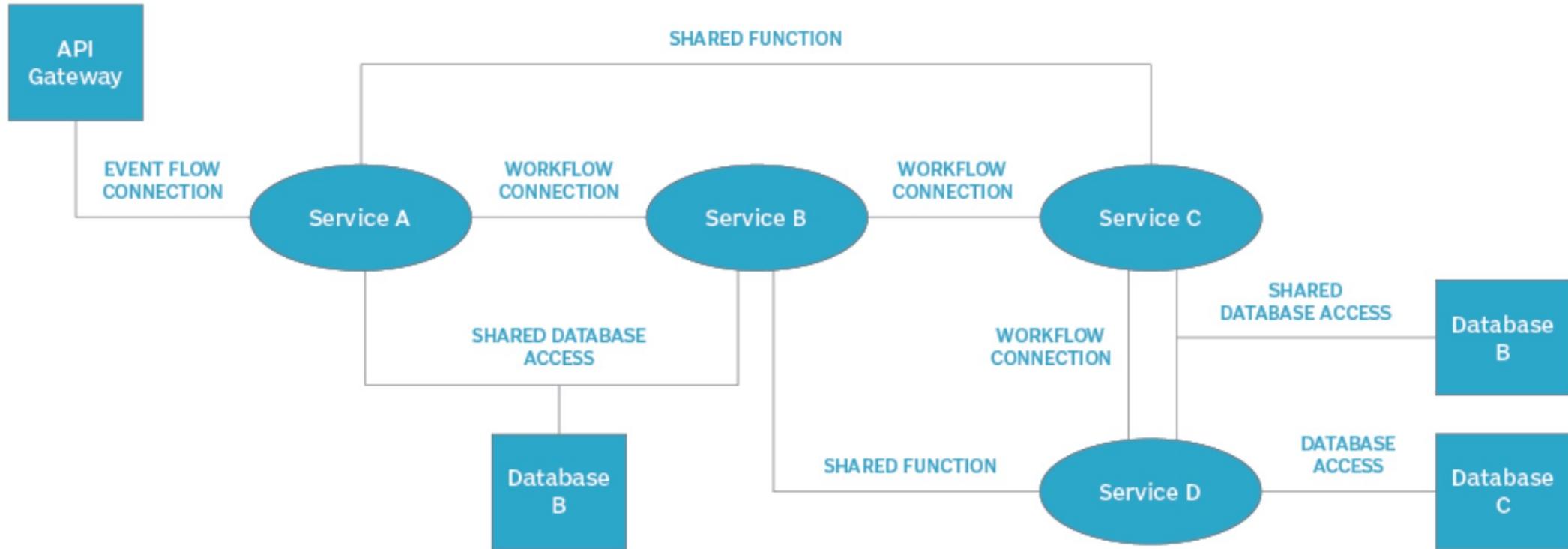
But... understand the complexities that comes with microservices

Need to align teams along with the application refactoring

Overhead of handling data consistency

Hard to debug issues. Needs comprehensive observability approach

What you don't want – “A Distributed Monolith”



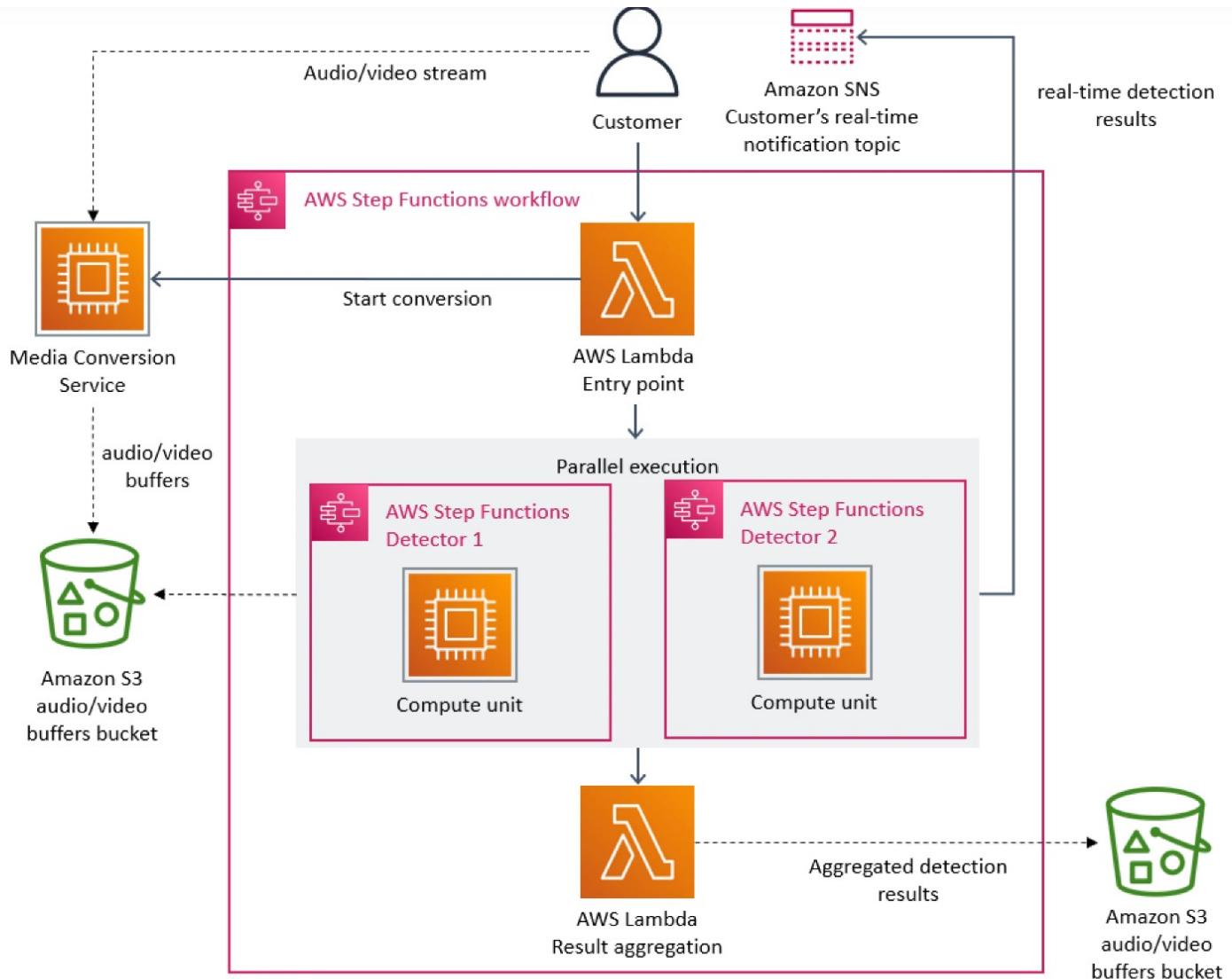
[Homepage](#)[Our Innovation](#) ▾[Our People](#)[Our Story](#)[Video Streaming](#)

Scaling up the Prime Video audio/video monitoring service and reducing costs by 90%

The move from a distributed microservices architecture to a monolith application helped achieve higher scale, resilience, and reduce costs.



Prime Video – Distributed Stack

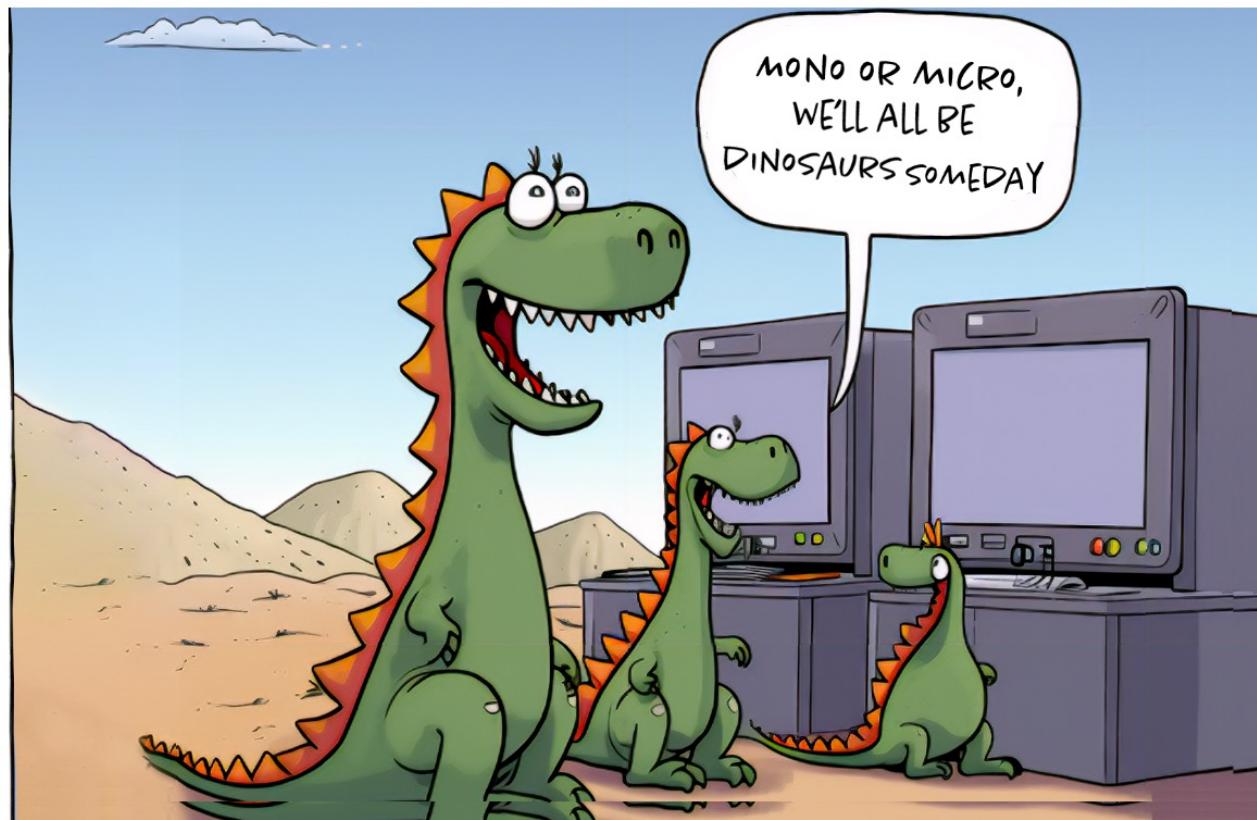


The initial architecture of our defect detection system.



Monoliths are not dinosaurs

May 05, 2023 • 774 words

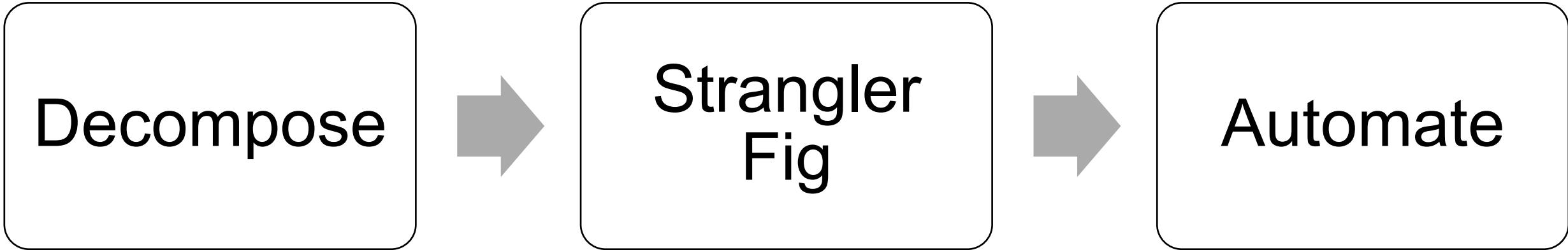


Building evolvable software systems is a strategy, not a religion. And revisiting your architectures with an open mind is a must.

- Werner Vogels, CTO - Amazon

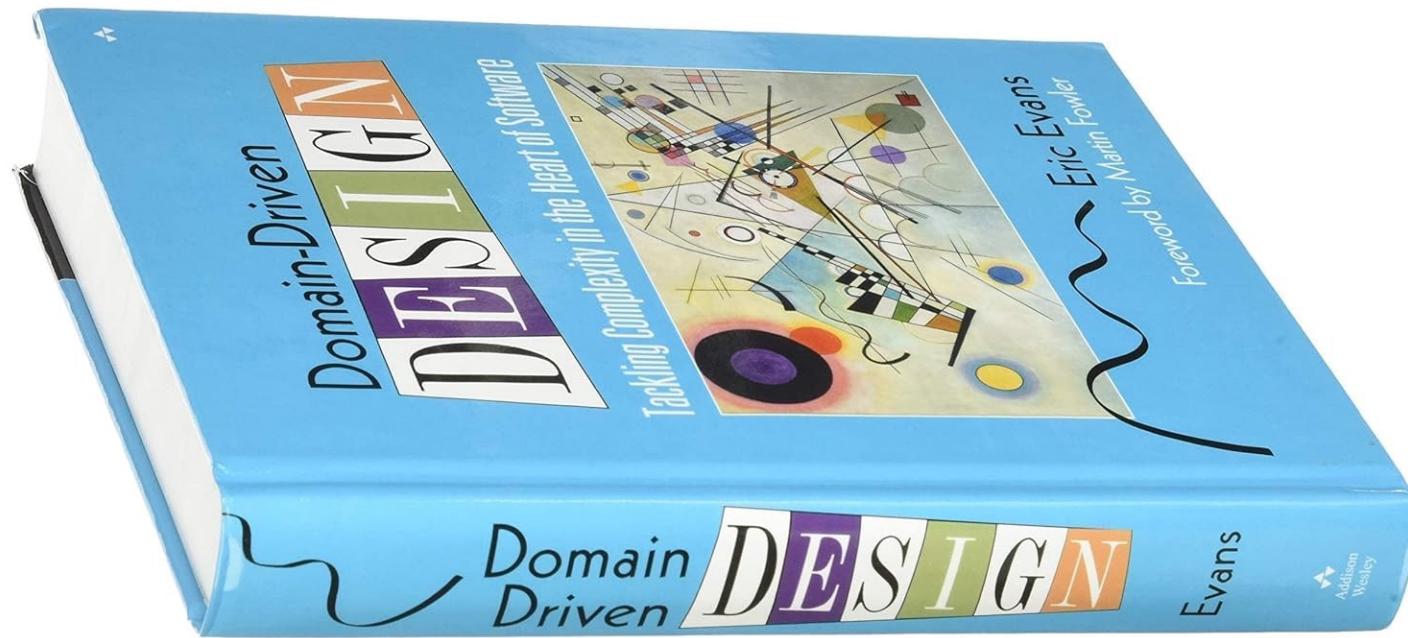
How do we move from a Monolith to Microservices Architecture?

3-Step Approach



Step 1: Decompose

Domain-driven Design (DDD)

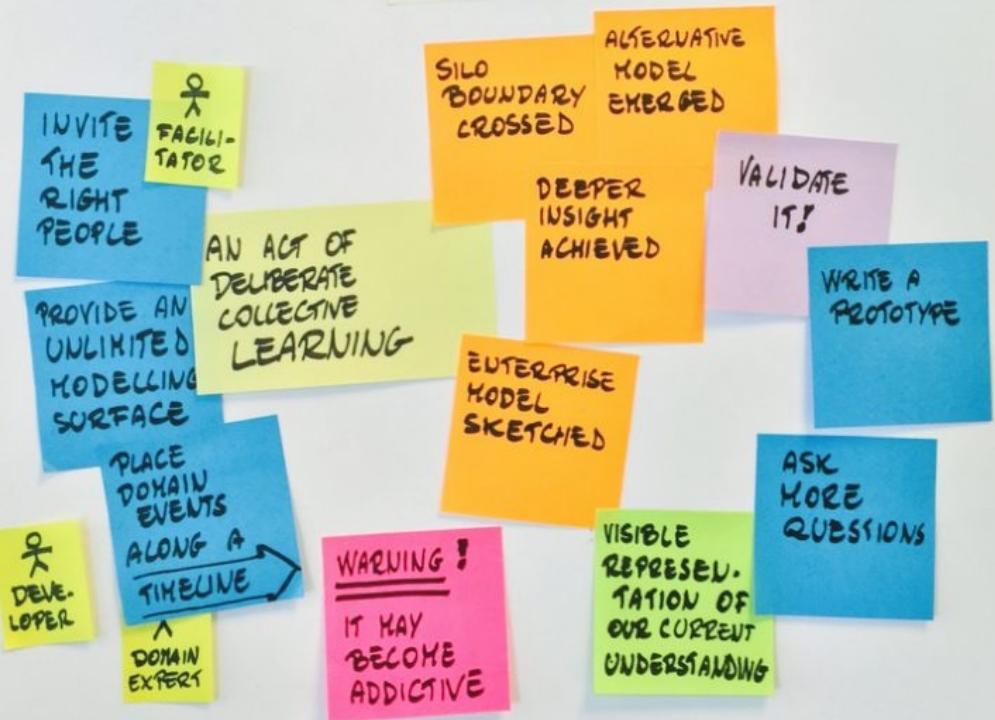


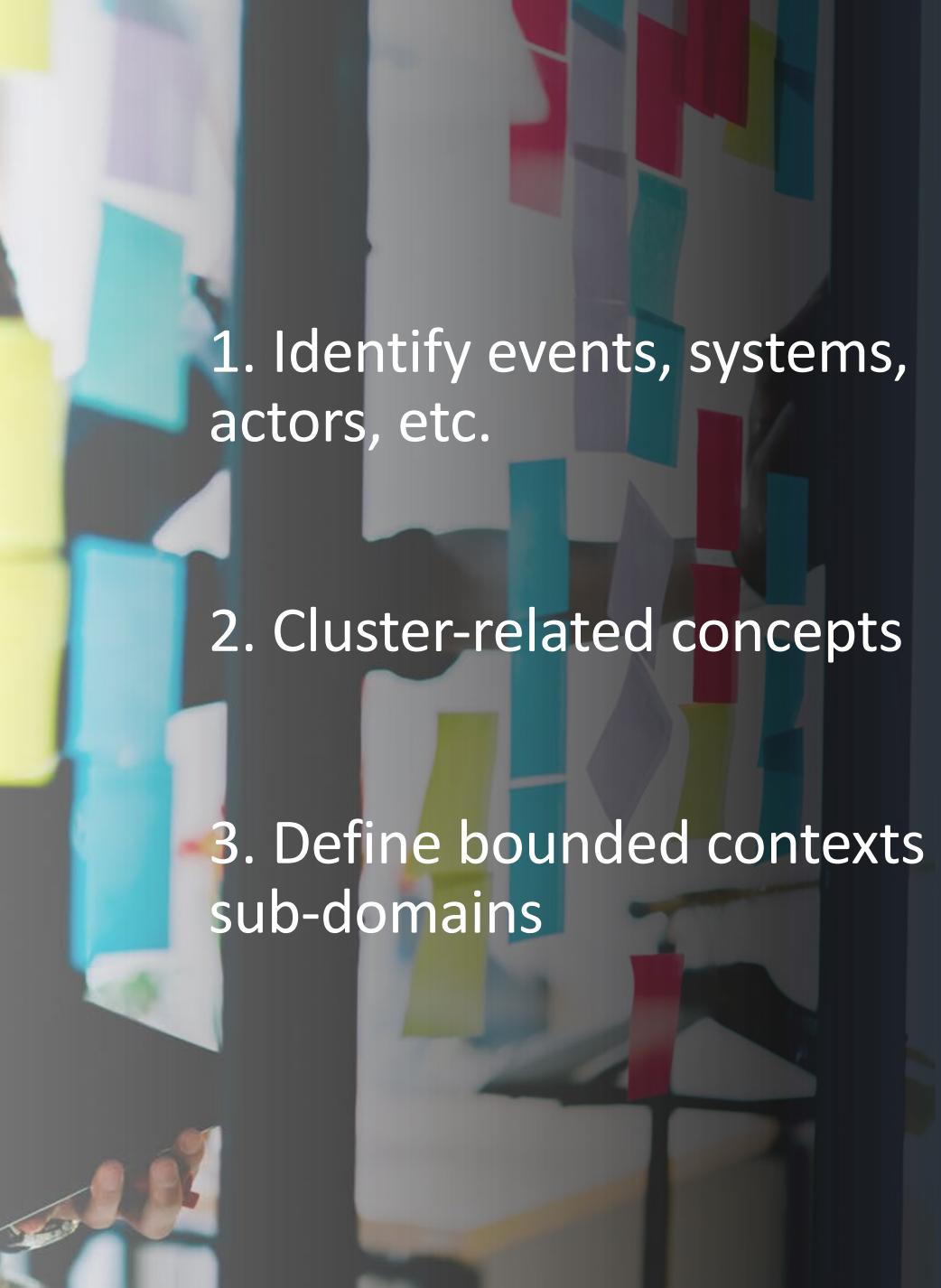
Domain-driven design focuses on *understanding* a domain, and its subdomains.

**DDD helps you to find a shared language
spanning business and engineering.**

EVENT STORMING

Alberto Brandolini



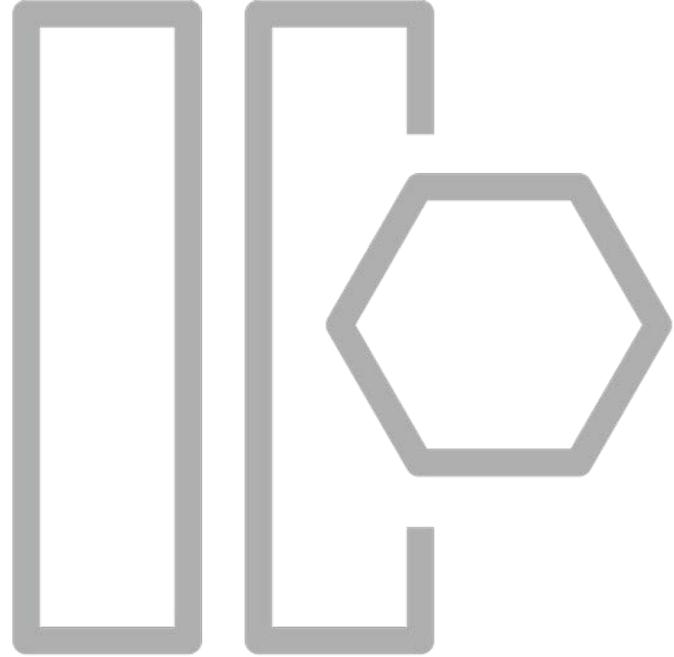
- 
1. Identify events, systems, actors, etc.
 2. Cluster-related concepts
 3. Define bounded contexts and sub-domains

What is EventStorming?

A workshop-based approach to breaking down a non-trivial domain with the goal of coming to a shared understanding.

EventStorming room layout

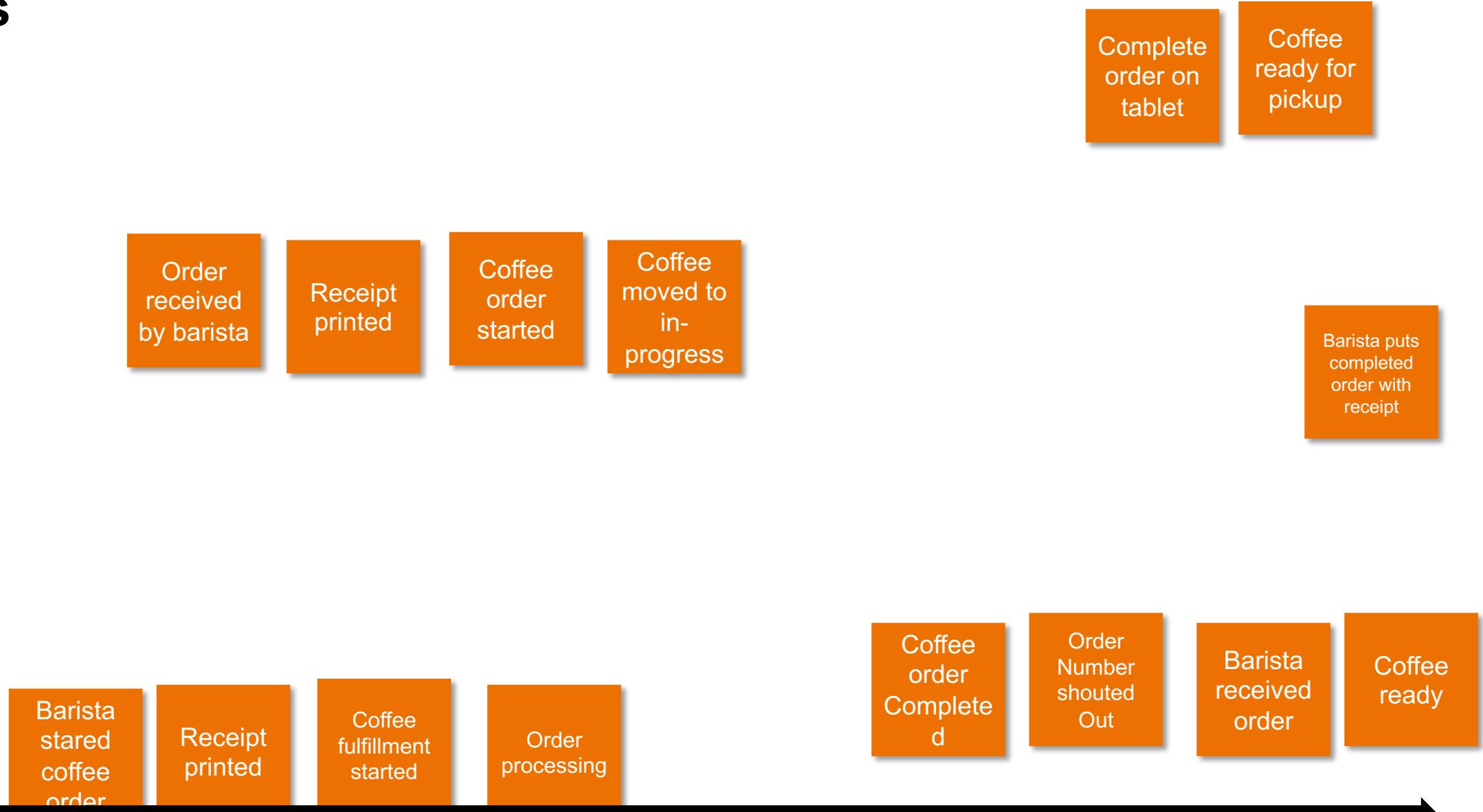




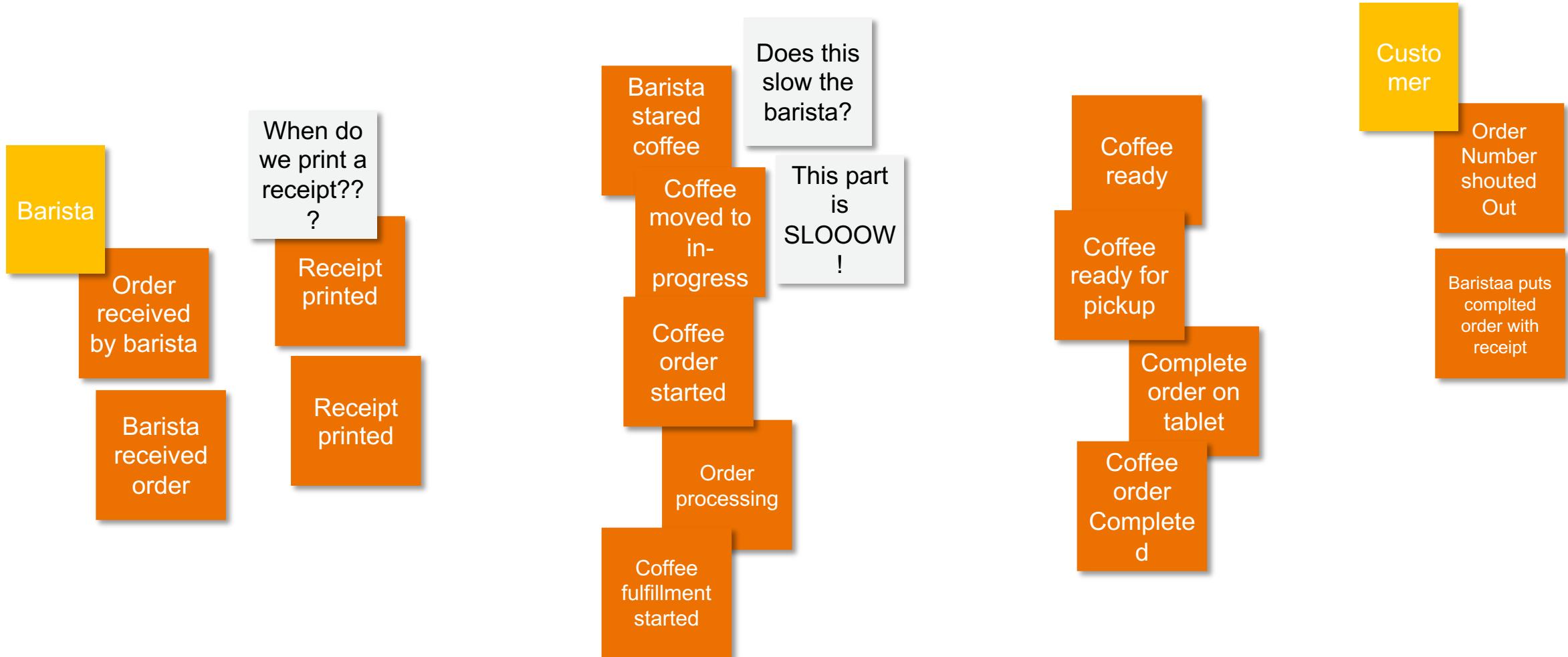
event

Something that a domain expert cares about.
Immutable facts that have occurred in the past.

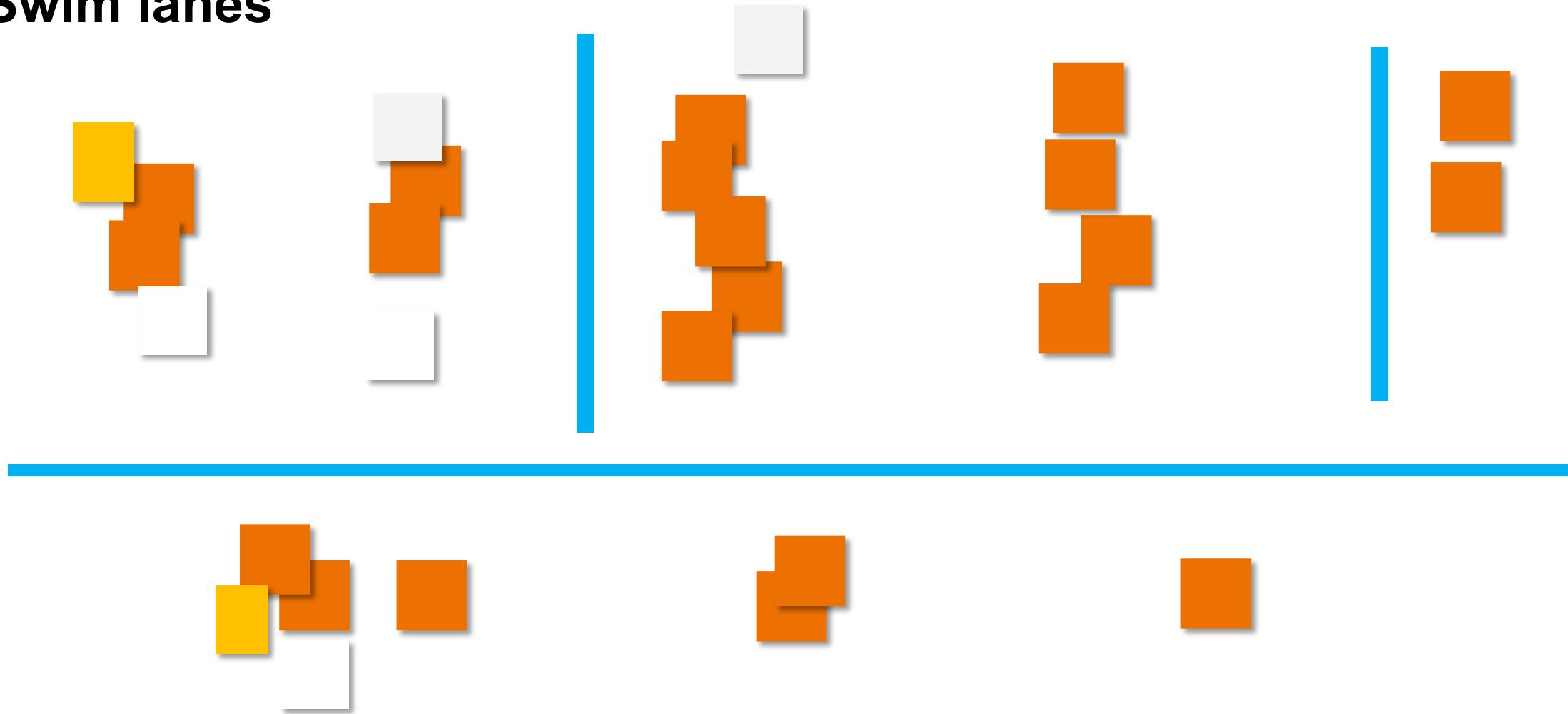
Events

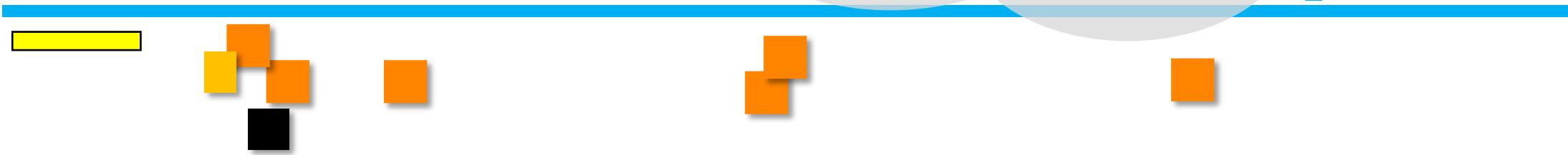
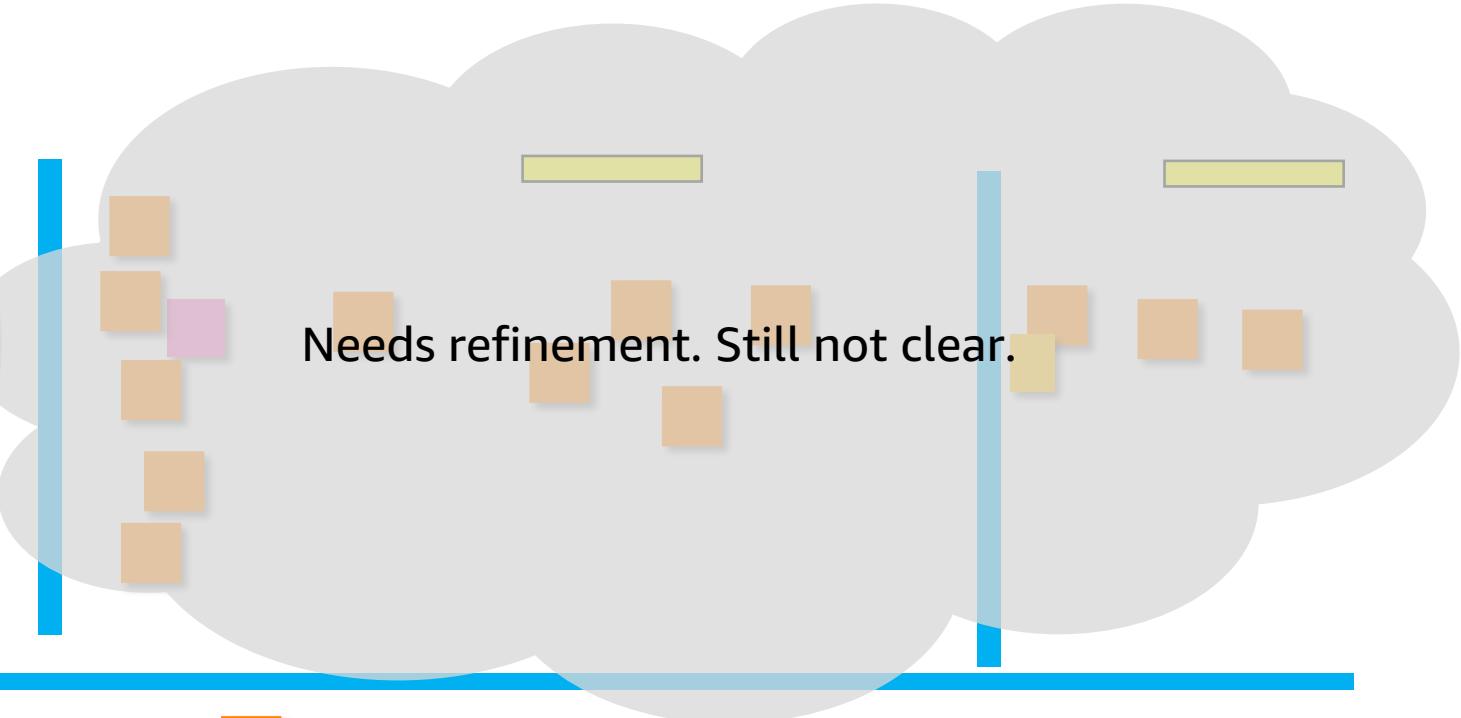


Identify Questions, Hotspots, People

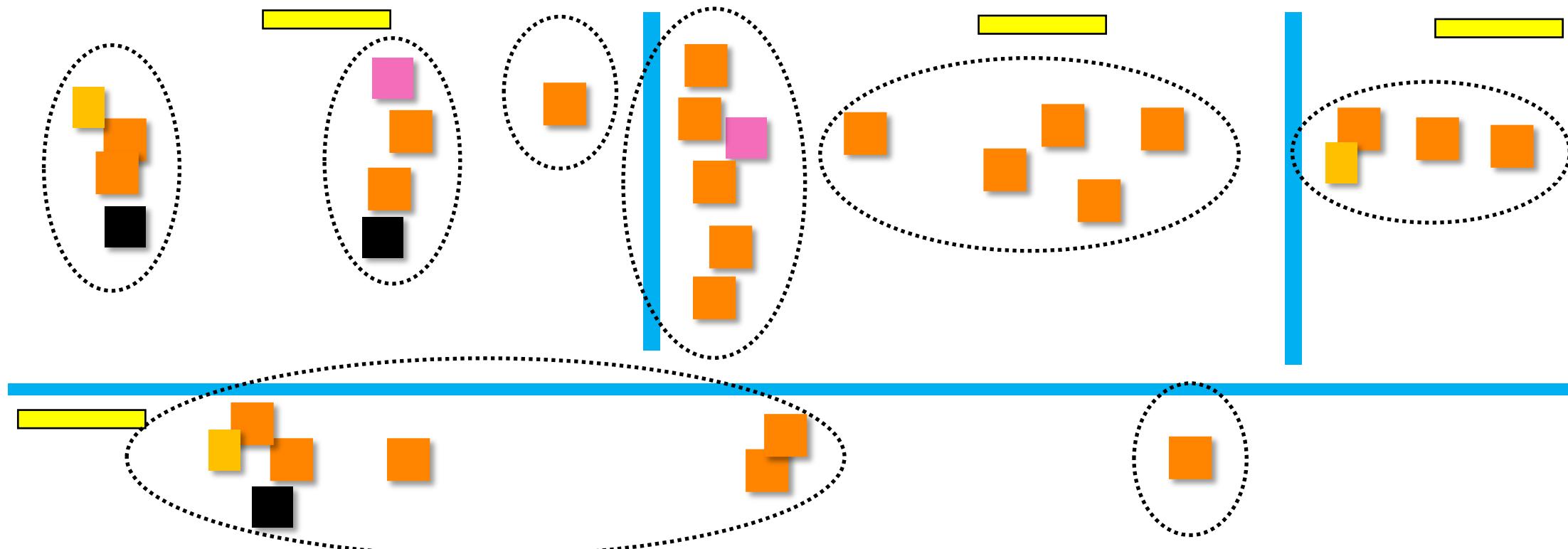


Swim lanes

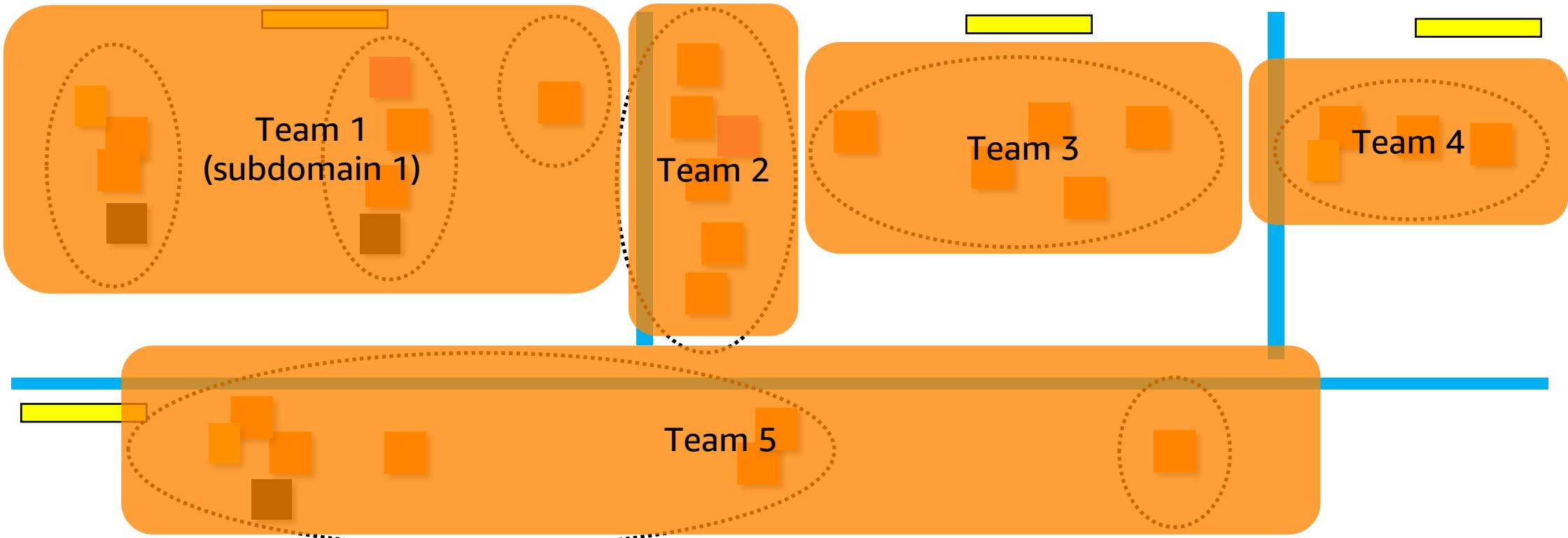




Identifying Boundaries



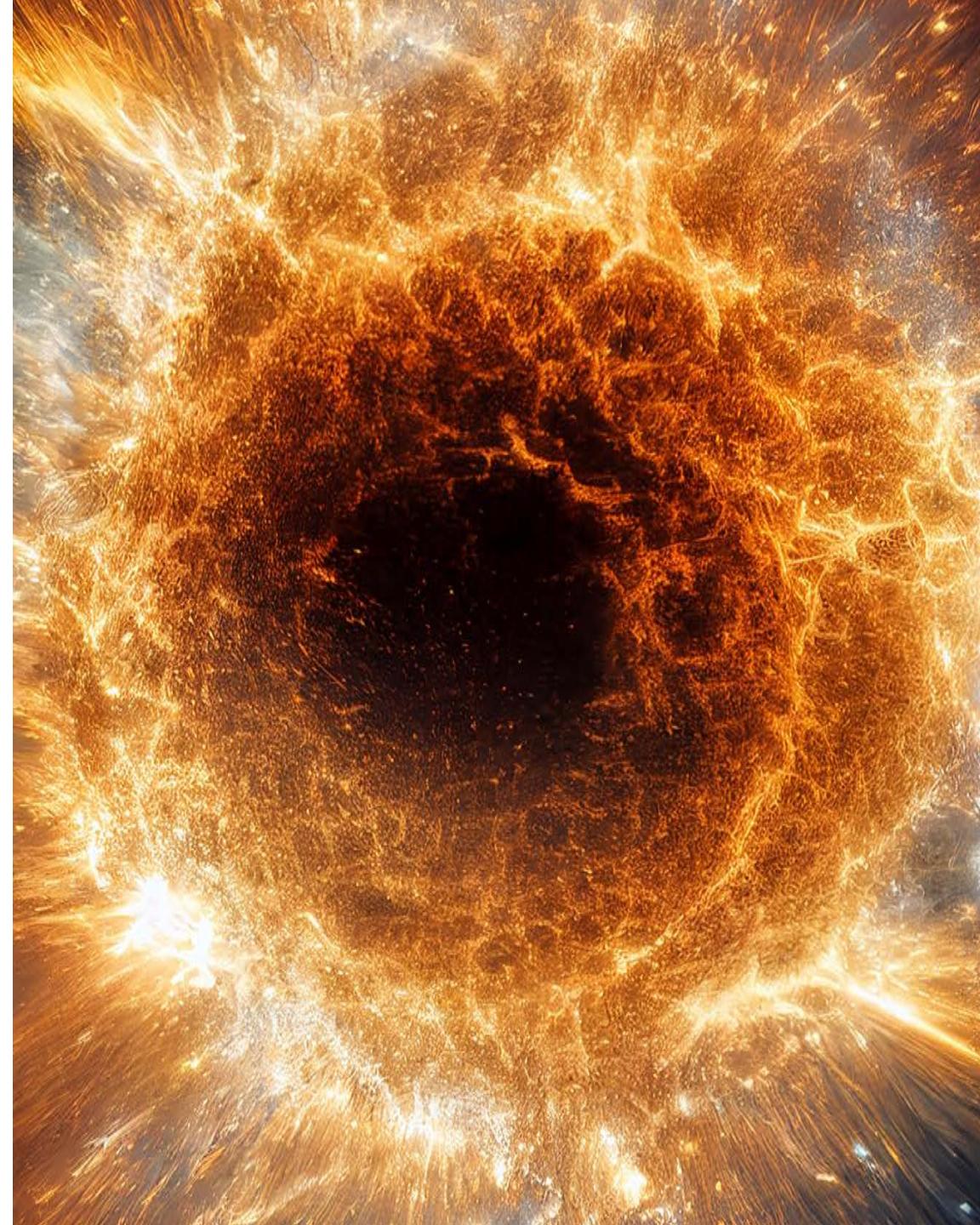
Identifying boundaries



Step 2: Strangler Fig Approach

**If you do a big-bang rewrite, the only
thing you are guaranteed of
is a big bang.**

Martin Fowler
Thoughtworks



Strangler Fig pattern - Releasable incremental refactoring



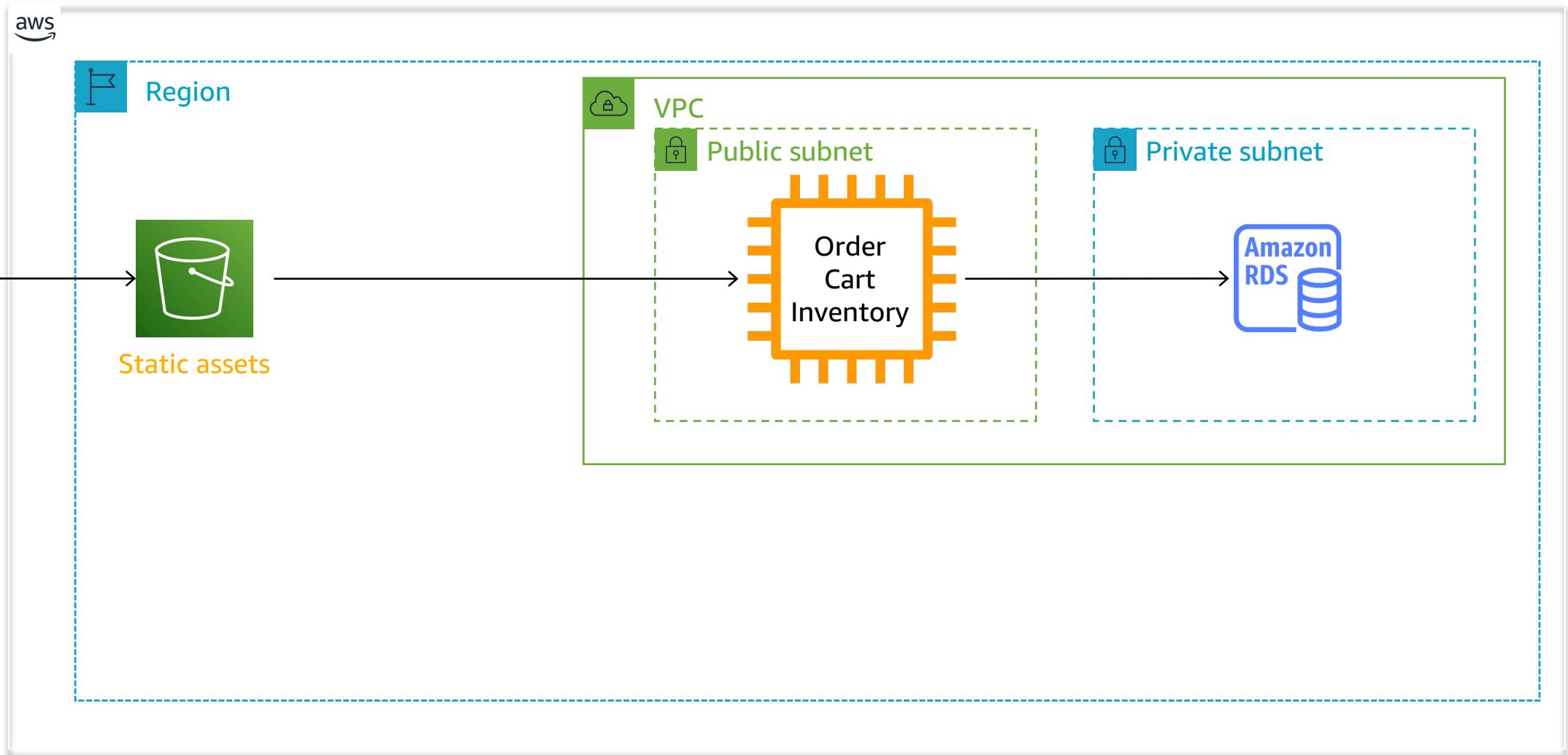
aws

<https://martinfowler.com/bliki/StranglerFigApplication.html>

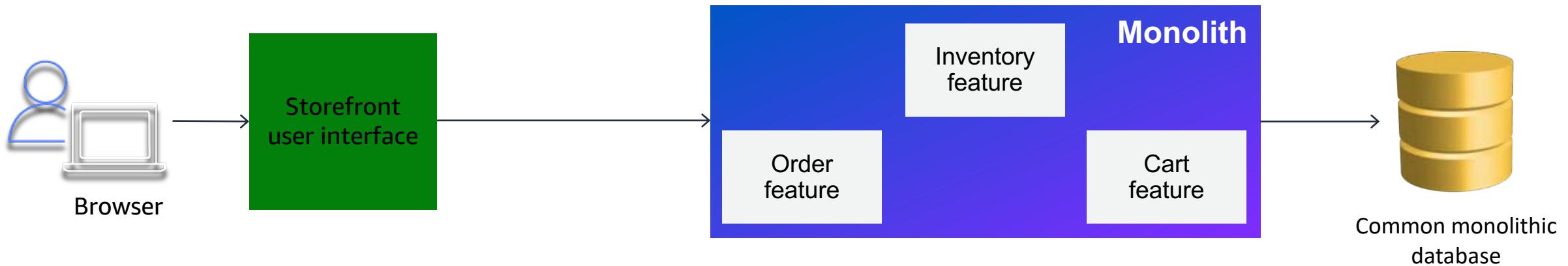
The monolith



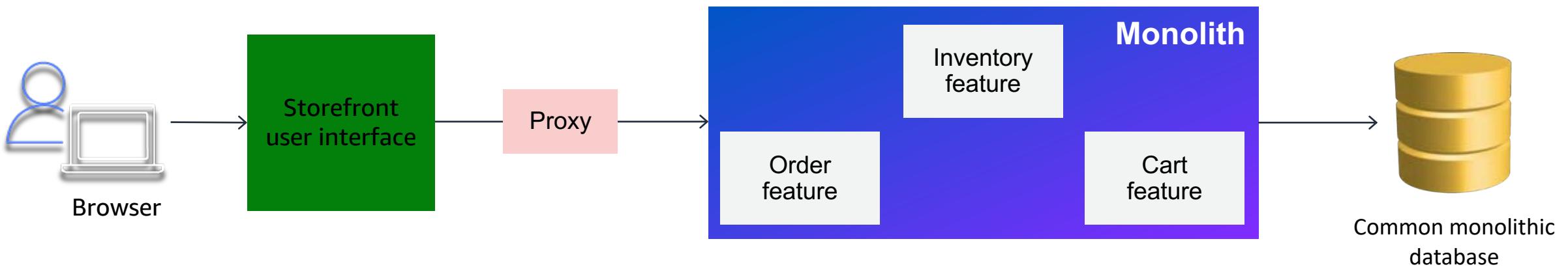
Current architecture



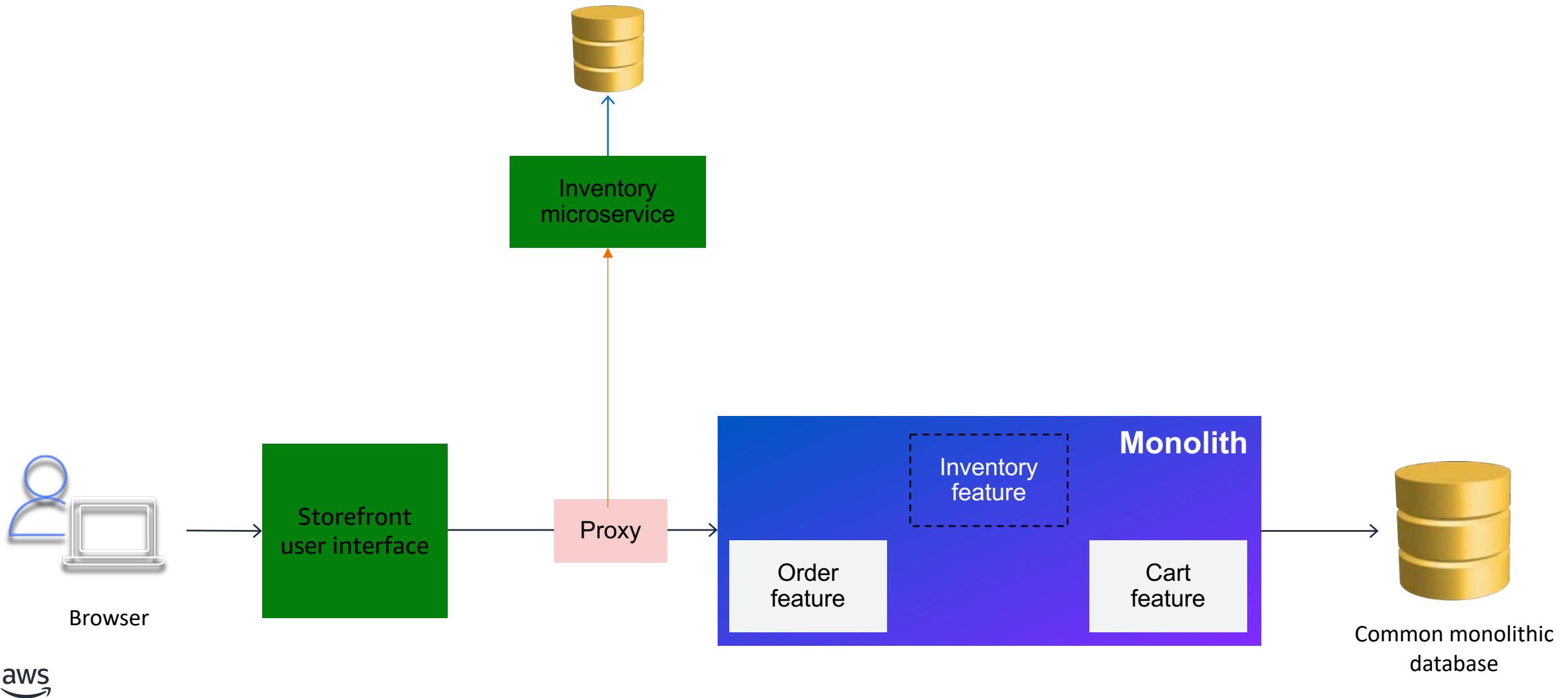
Decompose



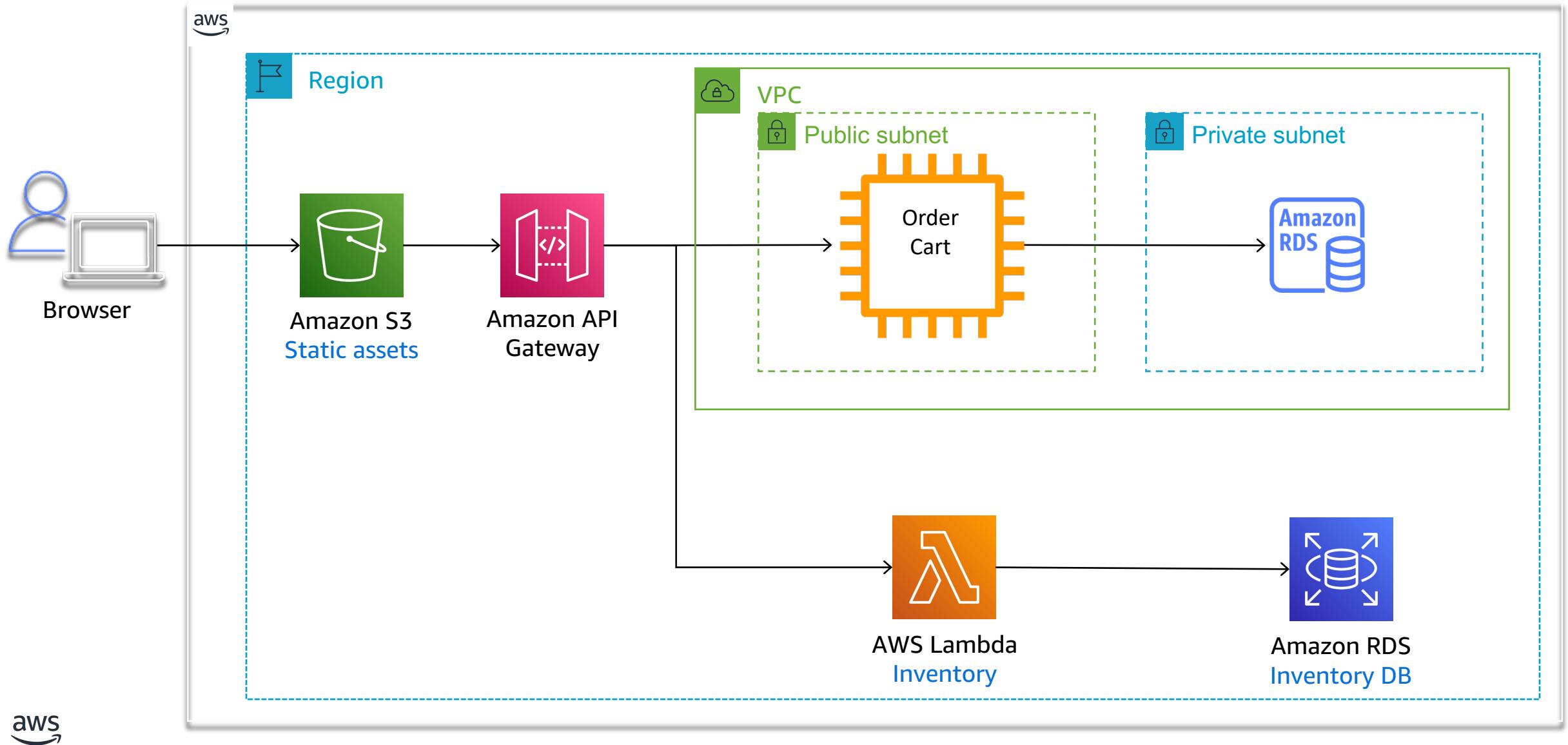
Proxy



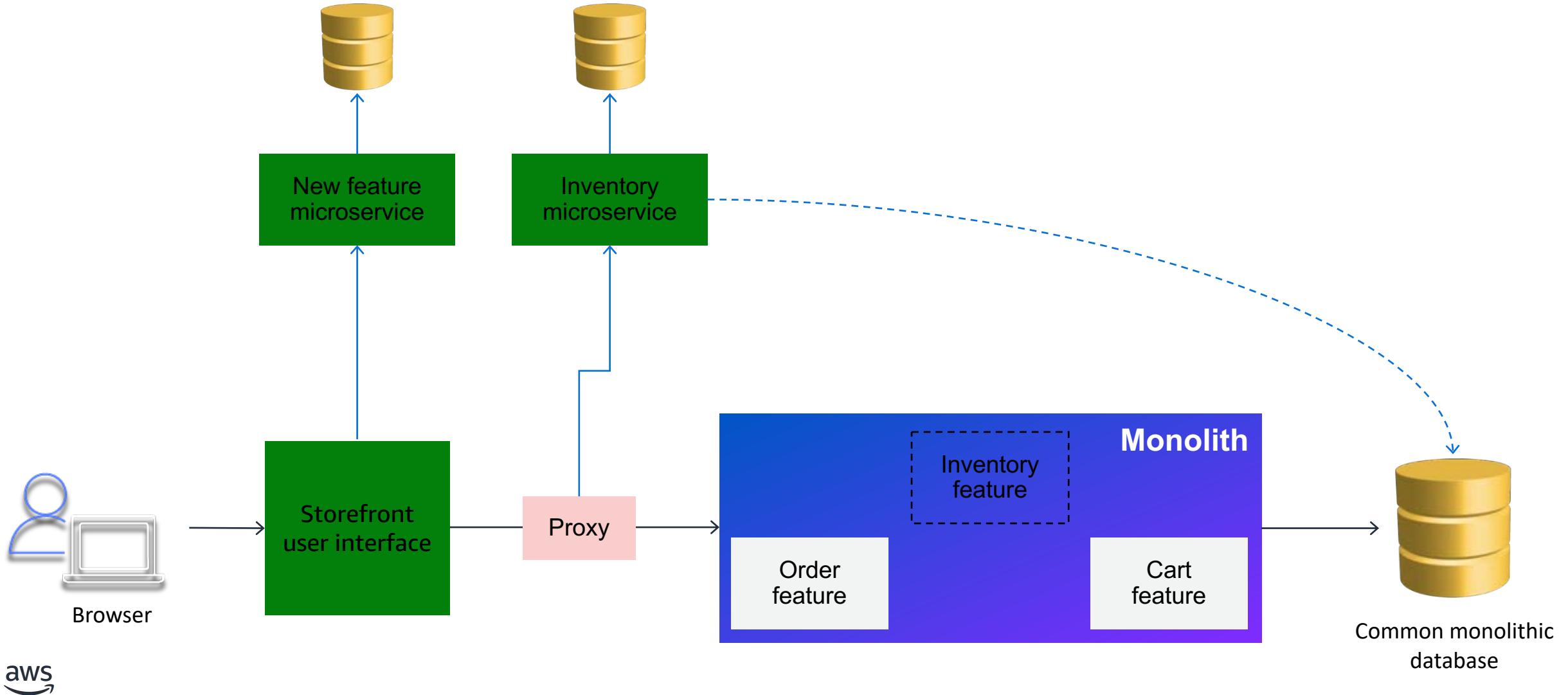
Strangler Fig



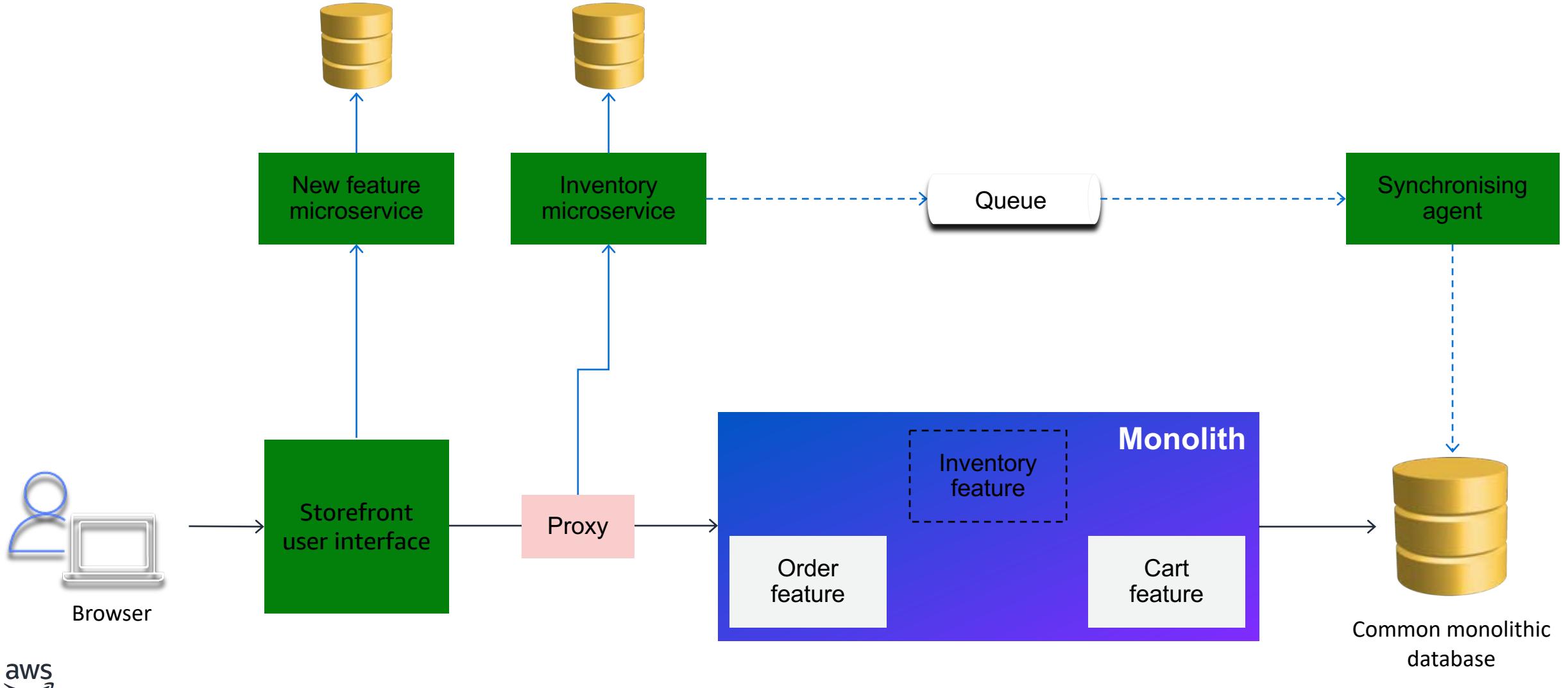
Strangler Fig architecture

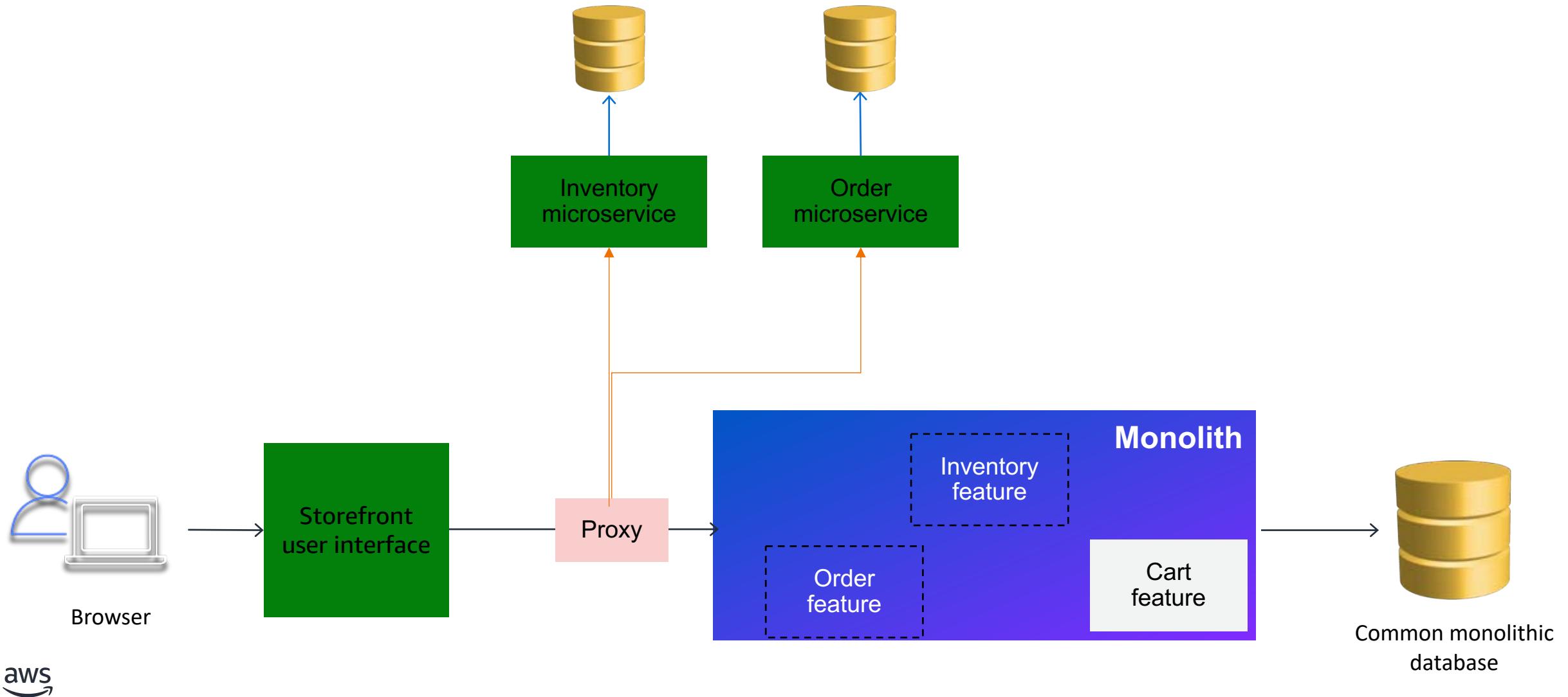


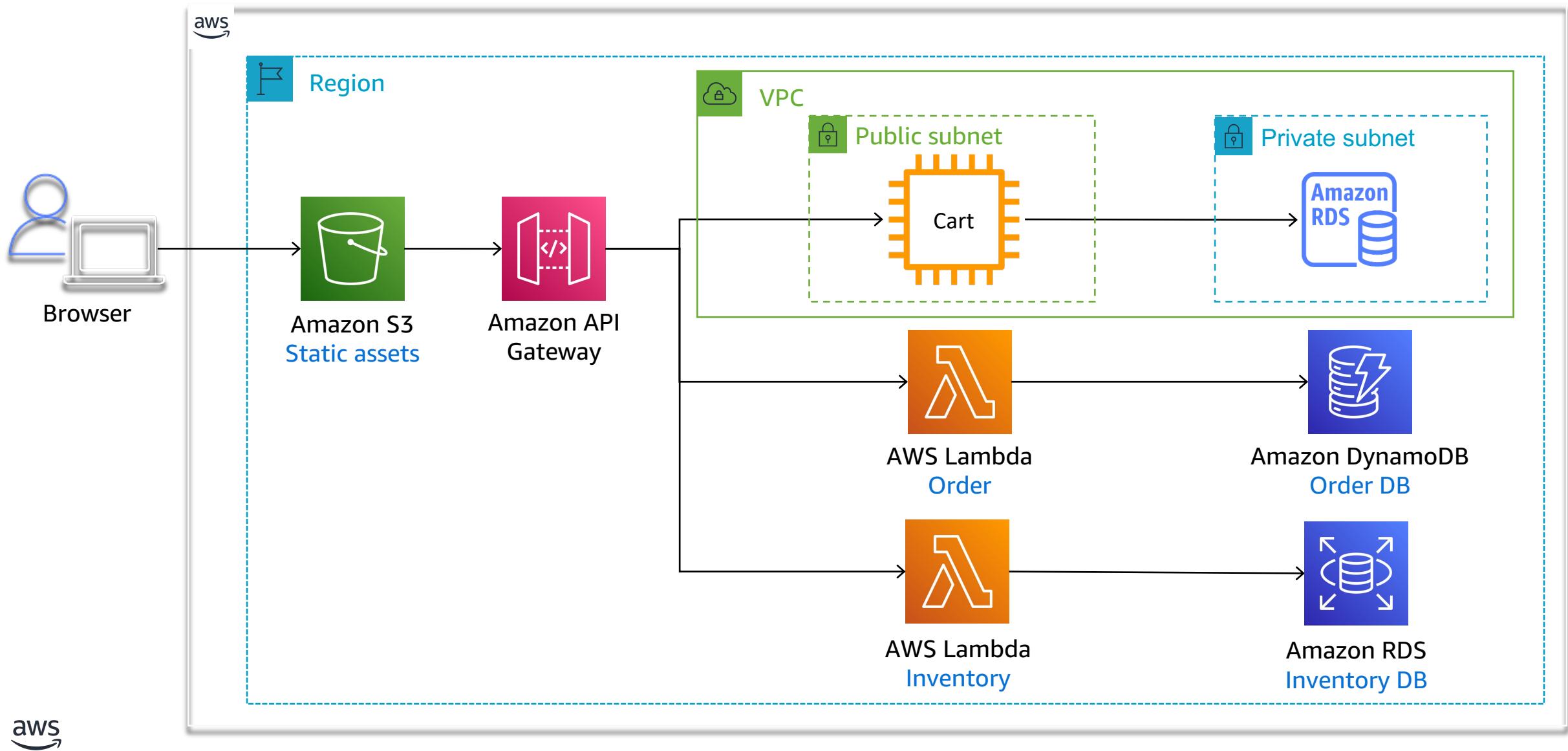
Data Consistency

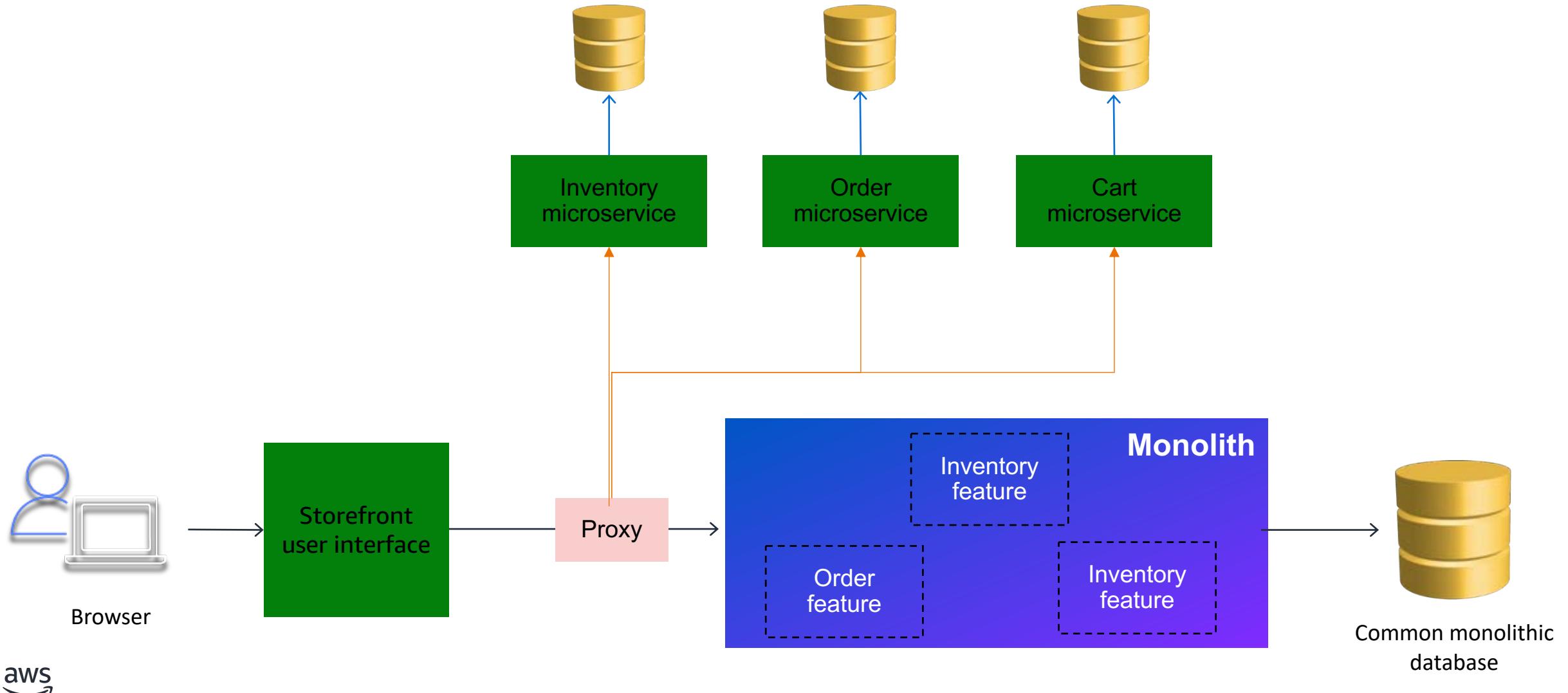


Data Consistency

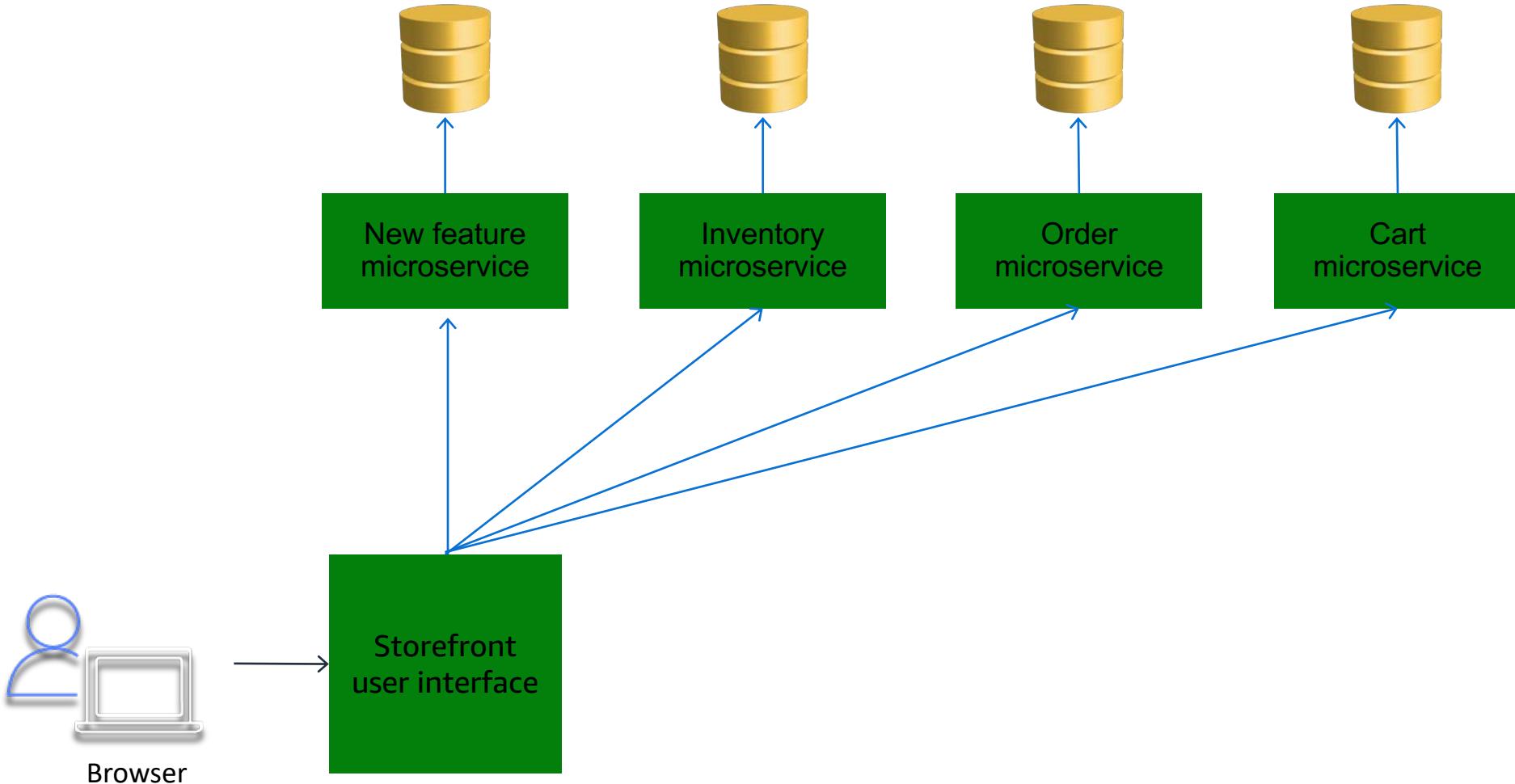




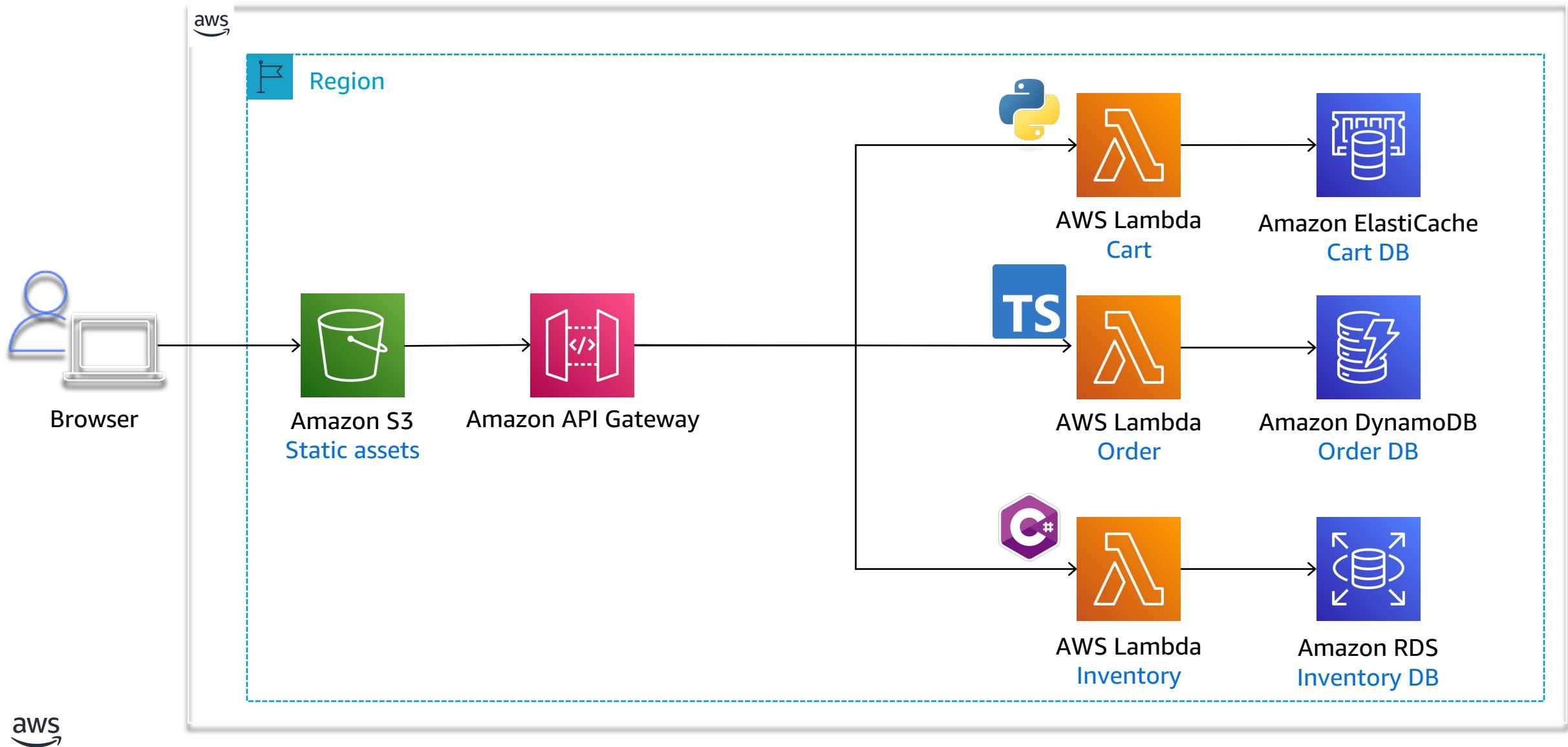




Final state

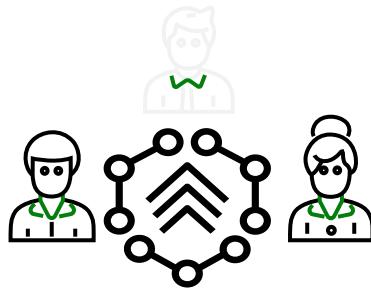


Final State



Step 3: Automate

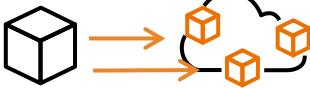
AWS Migration Hub Refactor Spaces



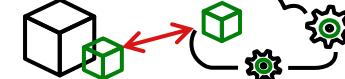
AWS Migration Hub Refactor Spaces



Reduce the time to set up
and manage a refactor
environment



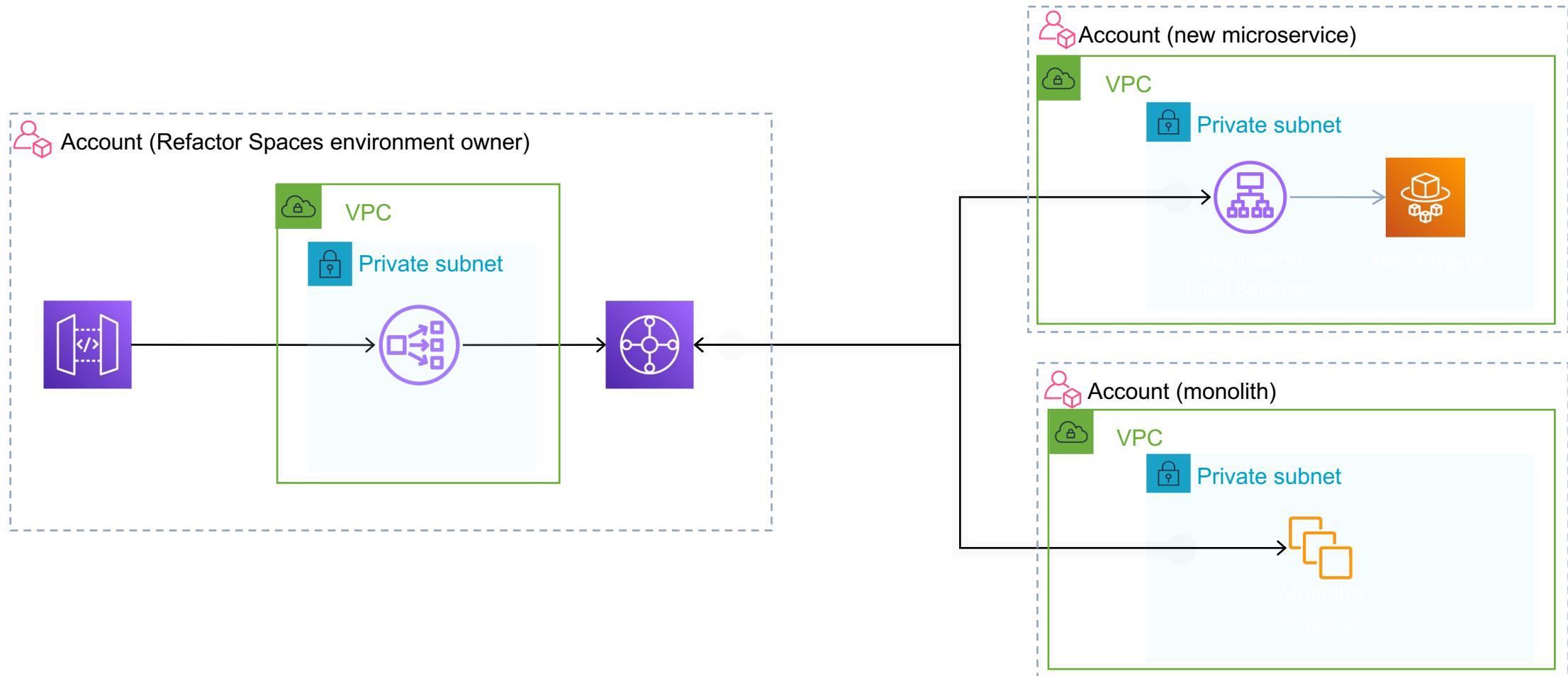
Shield application
consumers from
infrastructure changes



Reroute traffic from old to
new across multiple AWS
accounts

Start **refactoring** applications in **days** instead of months

Strangler fig with Refactor Spaces



Thank You!