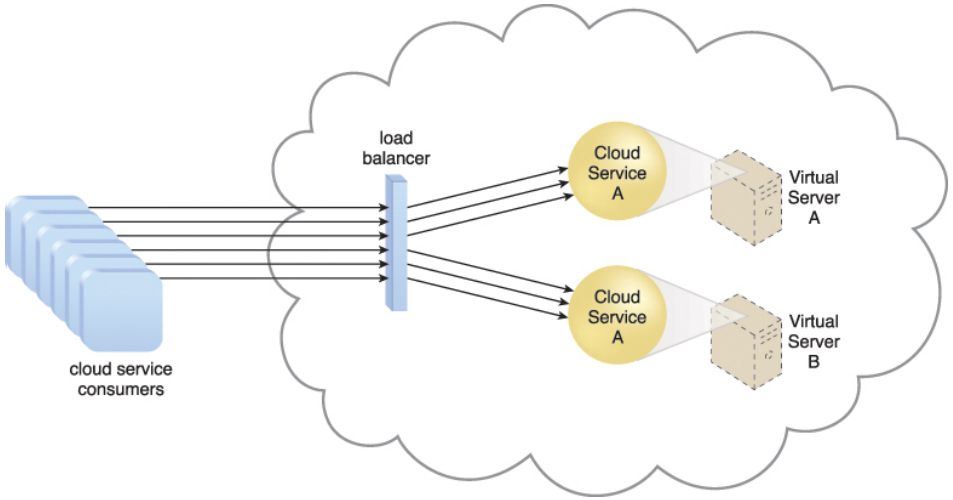
Serverless Data Processing (CSCI 5410)

Dr. Saurabh Dey

Outline

1. Stateful Service
2. Principles of Serverless Architectures
3. Example of Push-based, Event Driven Architecture

Clients requesting service from cloud



Workload Distribution Architecture

Image Citation: Erl, Thomas, Ricardo Puttini, Zaigham Mahmood. Cloud Computing.. [VitalSource Bookshelf].

Consider the following:

Client A wants to use cloud service
Client B wants to use cloud service
Client C wants to use cloud service

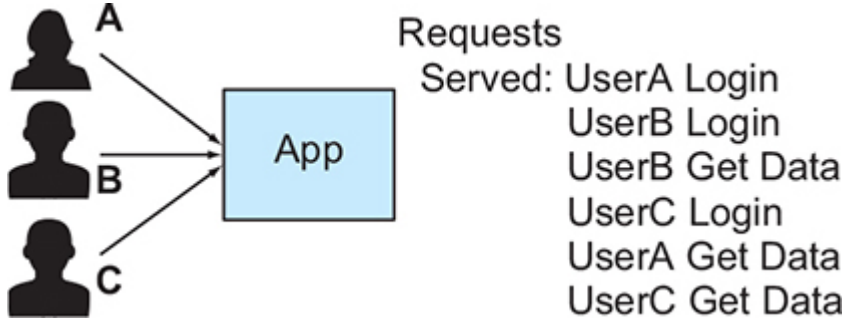


Image Source: https://livebook.manning.com/book/cloud-native/chapter-5

App Scale-out; microservices

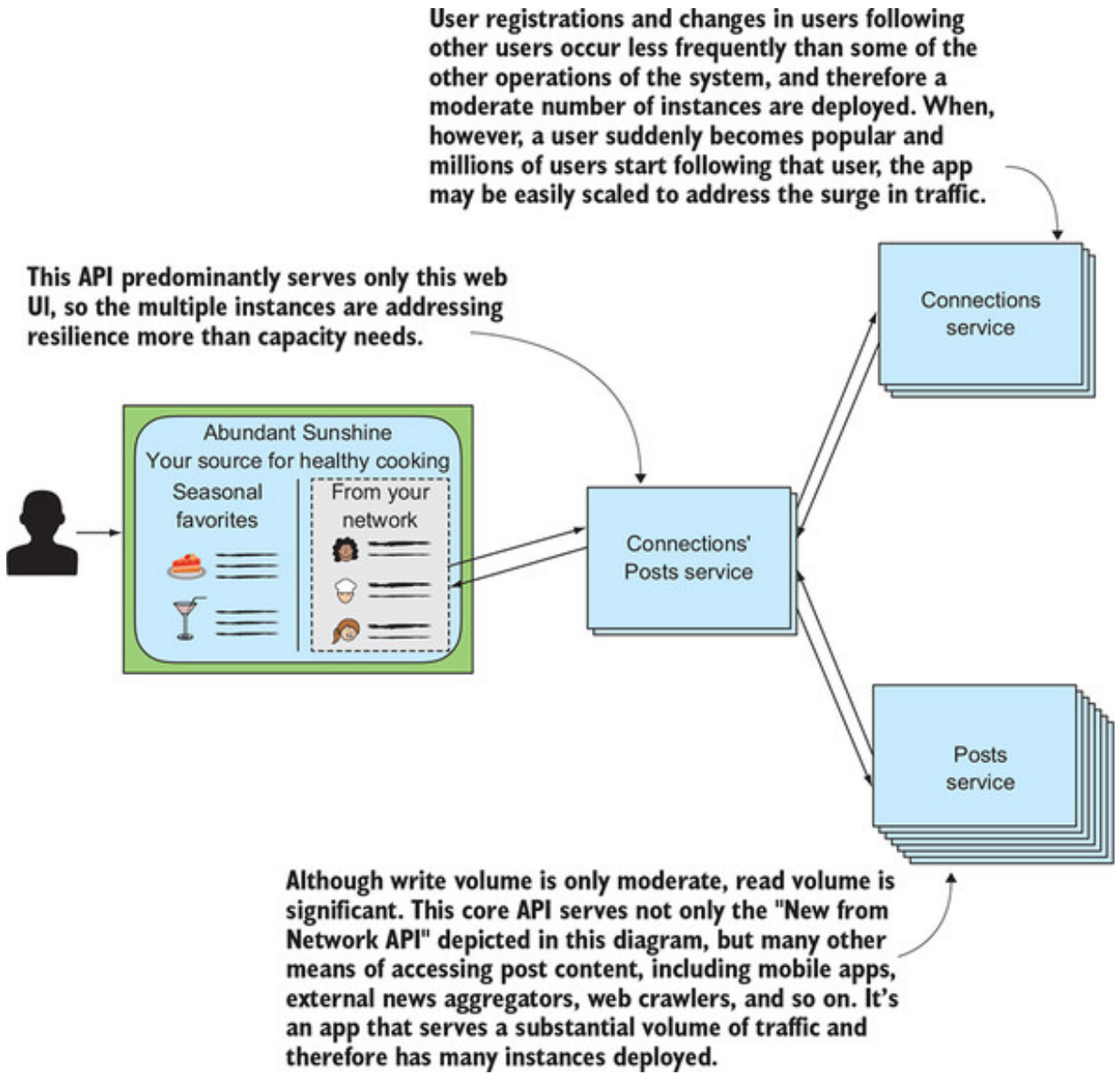
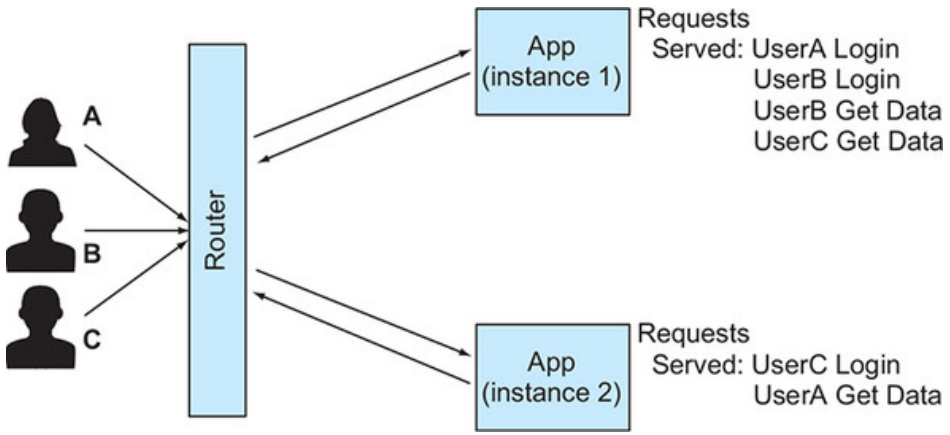


Image Source: https://livebook.manning.com/book/cloud-native/chapter-5/22

Clients requesting service from cloud - sateful



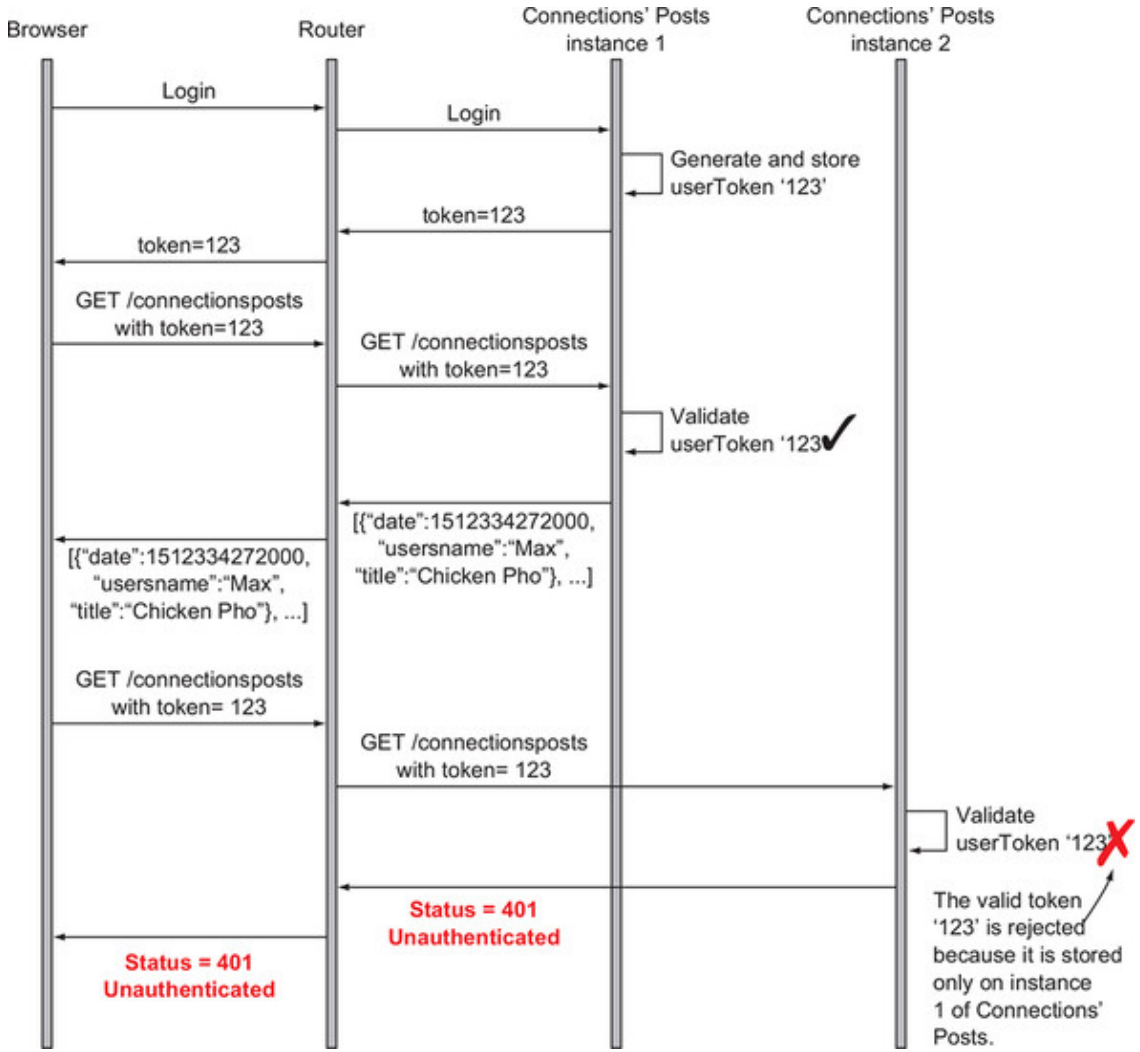
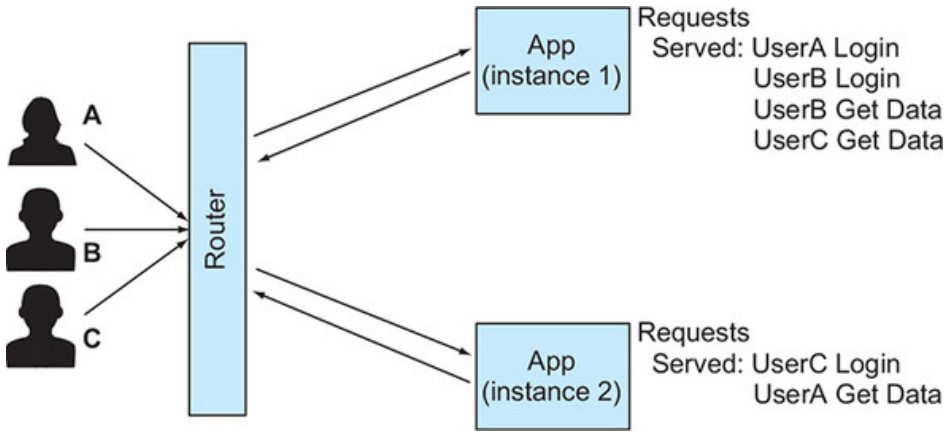


Image Source: https://livebook.manning.com/book/cloud-native/chapter-5

Sticky sessions



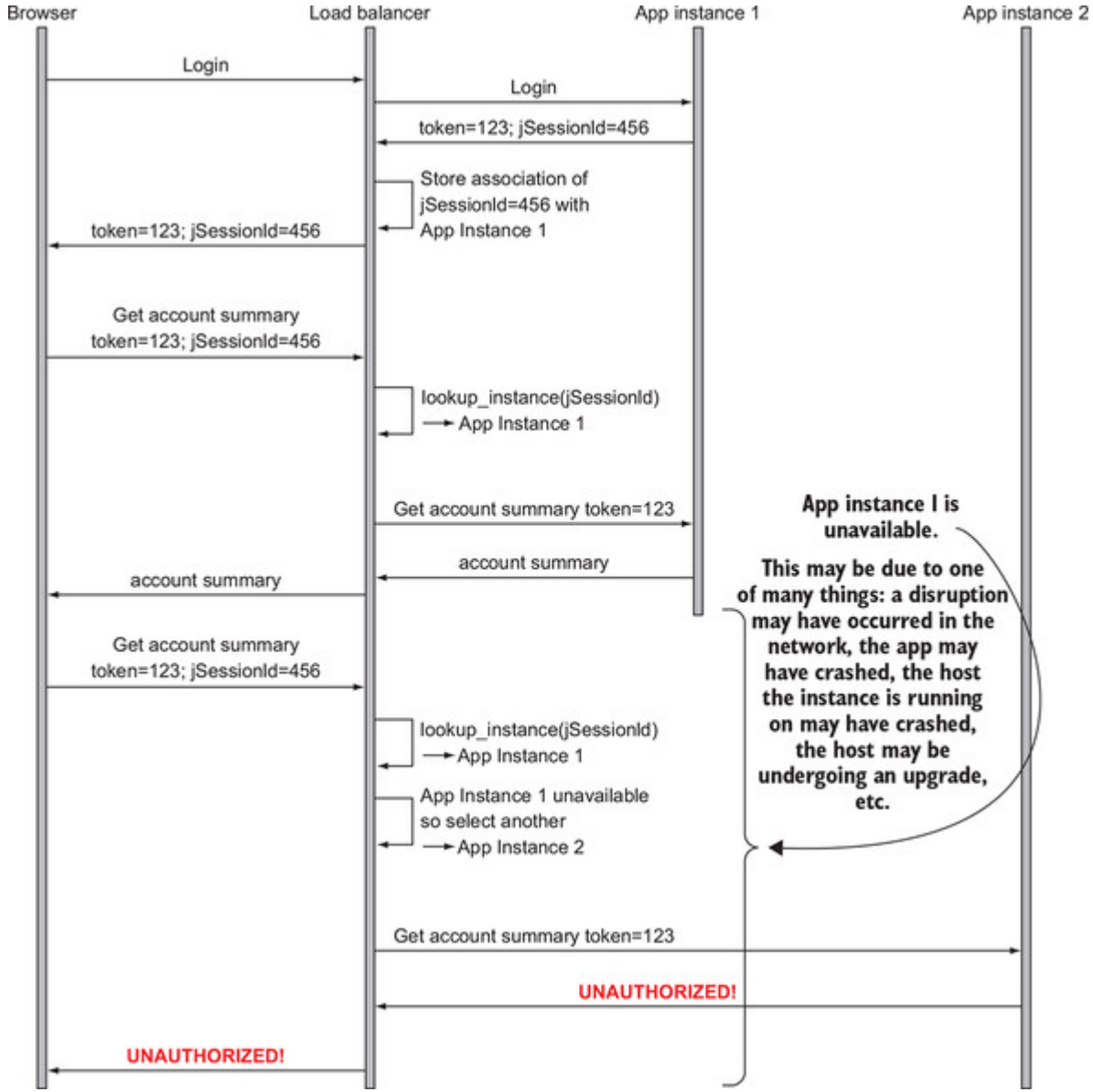
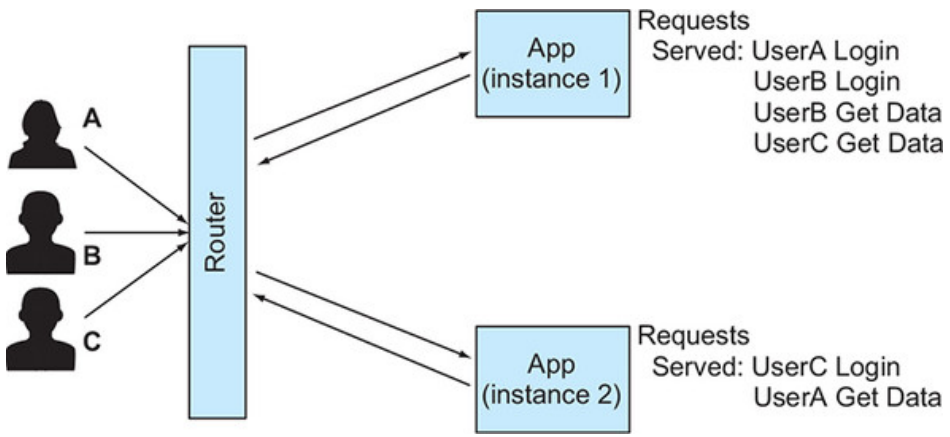


Image Source: https://livebook.manning.com/book/cloud-native/chapter-5

Solving problem – stateful service



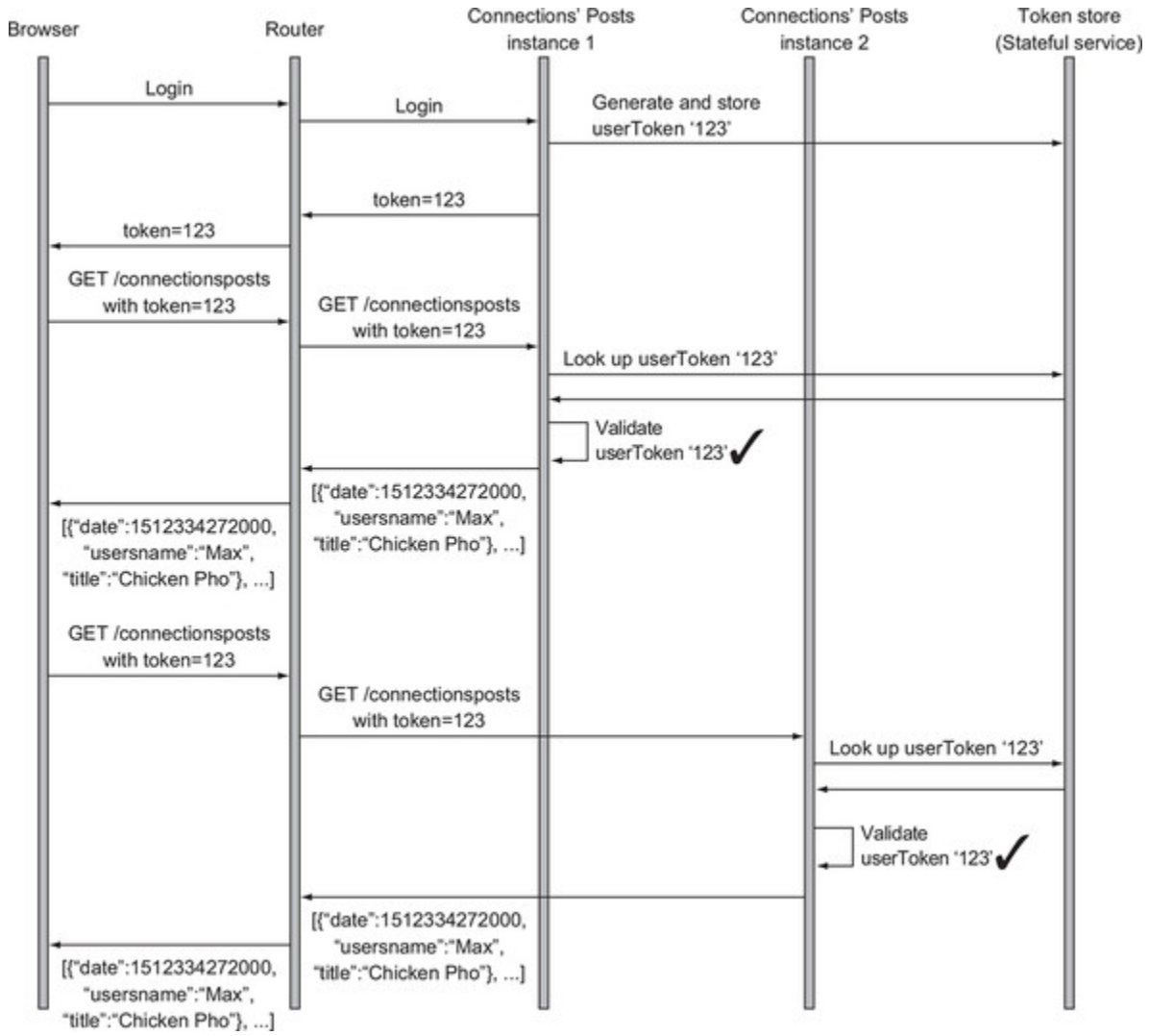
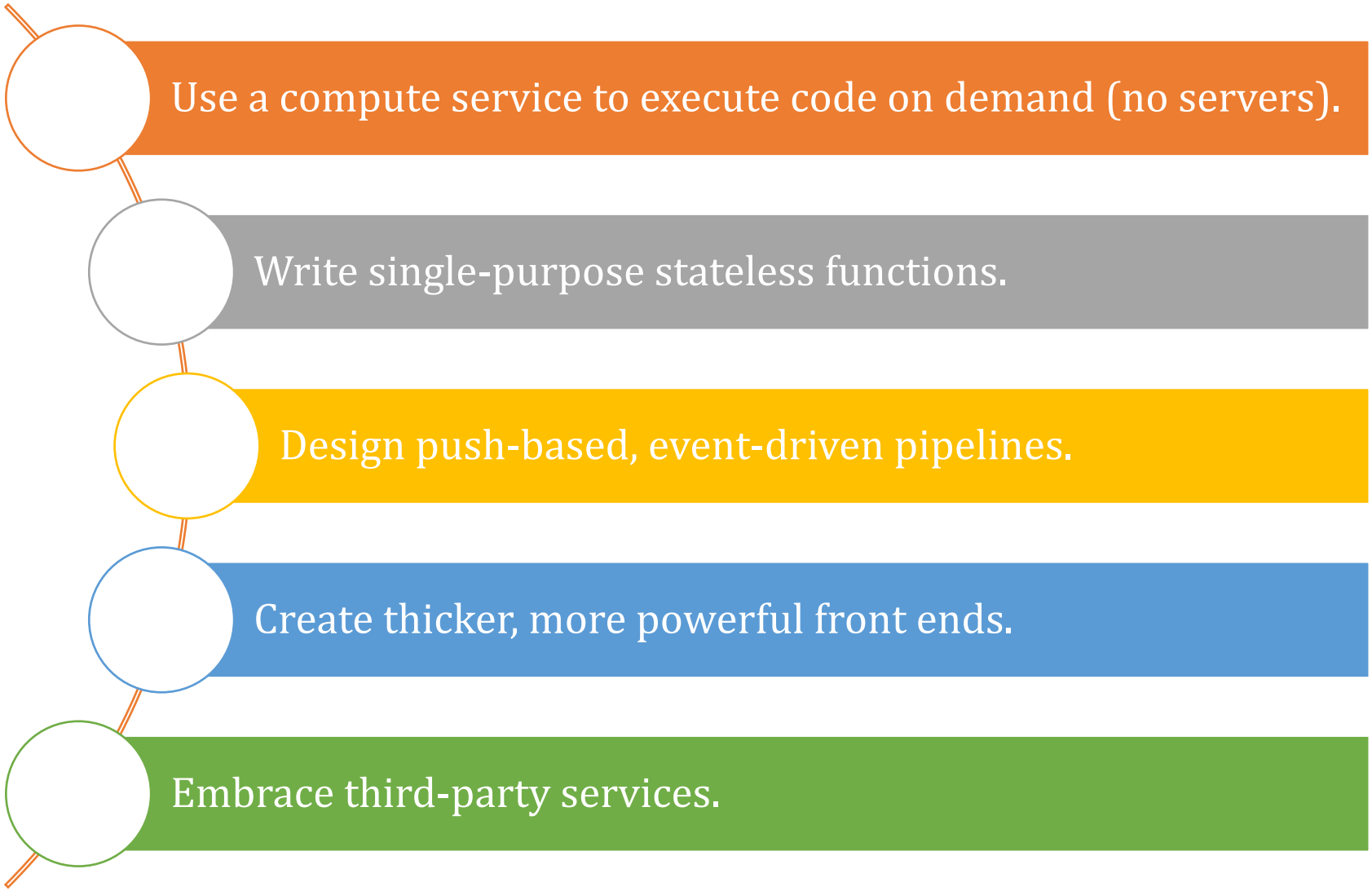


Image Source: https://livebook.manning.com/book/cloud-native/chapter-5

Principles of Serverless Architectures



Use Compute Service

- Serverless architectures are based on SOA

  "SOA, or service-oriented architecture, defines a way to make software components reusable via service interfaces. These interfaces utilize common communication standards in such a way that they can be rapidly incorporated into new applications without having to perform deep integration each time." - IBM

- Custom codes run in a stateless compute service.

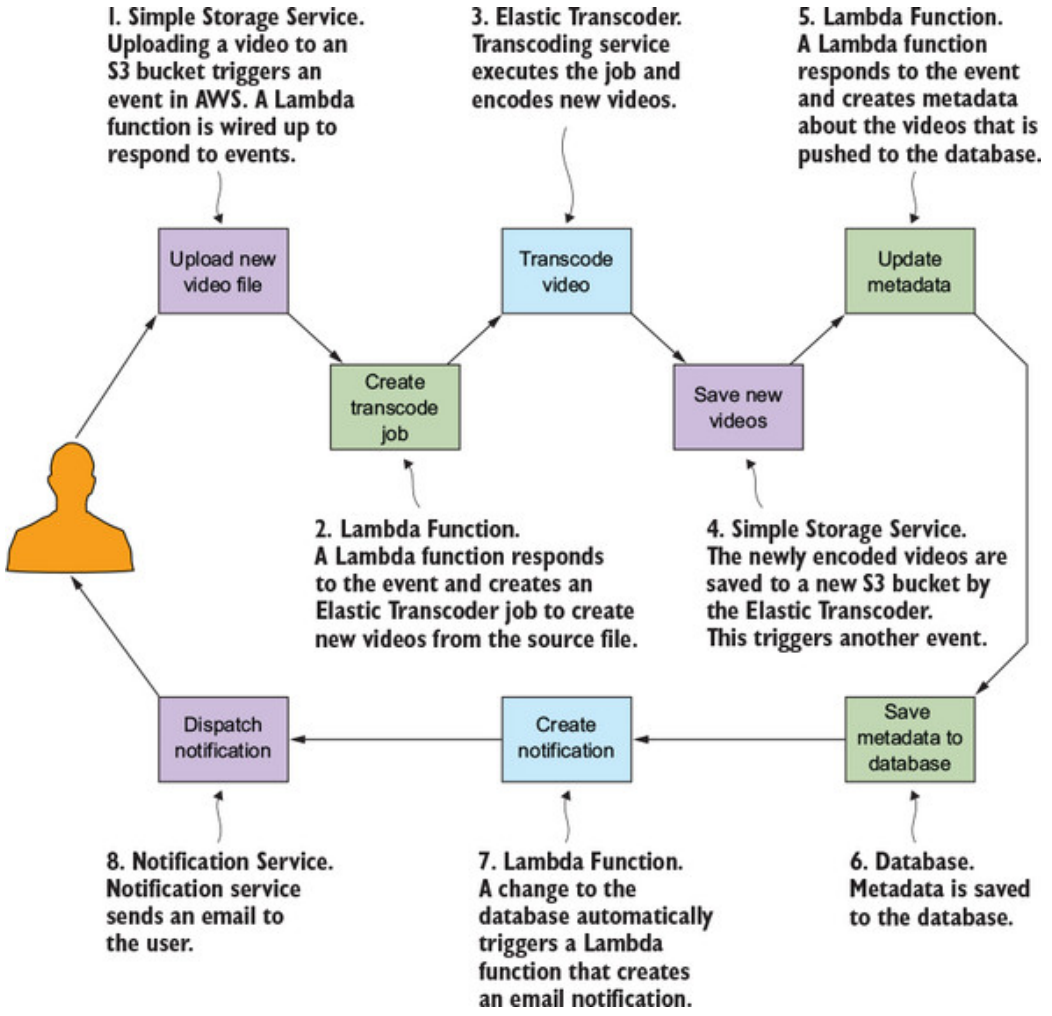
- Developers can write functions to carry out almost any common task, such as reading and writing to a data source, calling out to other functions, and performing a calculation.

Single-Purpose Stateless Functions

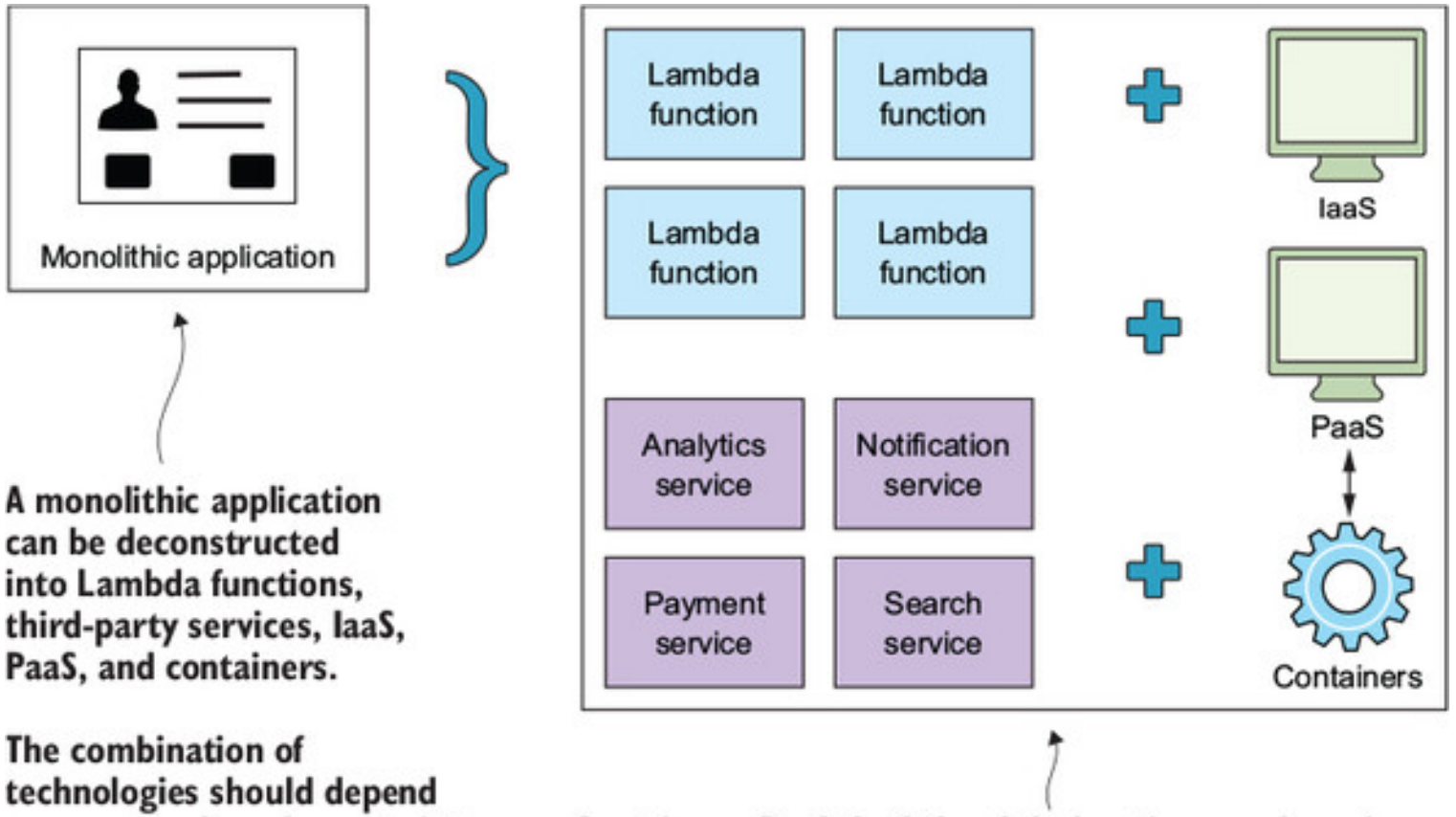
- Try to design your functions with the **single responsibility principle** (SRP) in mind.
- A function that does just one thing is more testable and robust and leads to fewer bugs and unexpected side effects.
- A granular function with a well-defined interface is also more likely to be reused within a serverless architecture.
- Code for a compute service should be created in a stateless style.

Design push-based, event-driven pipelines

- Systems can be built serverless from scratch, or monolithic can be gradually reengineered
- Most flexible and powerful serverless designs are event-driven
- https://livebook.manning.com/book/serverless-architectures-on-aws/chapter-1/56



Server to Service



A monolithic application can be deconstructed into Lambda functions, third-party services, IaaS, PaaS, and containers.

The combination of technologies should depend on your needs and constraints. However, more technologies require more overhead, time, and energy.

Containers, PaaS, IaaS, Lambda functions, and services can talk to one another. If you have designed a system using a combination of the above technologies you must consider how orchestration of events will take place.

- A Monolithic hospital management system (that supports infrastructure, day-to-day operation, logistics, security, transport etc.) are designed 15 years ago.
- There is a need of upgrade and adding flexibility.
- The task was given to your team, and your team is using cloud technologies, web technologies, mobile cloud computing etc.
- Your team has decided to use services.

- **What would be your approach?**
- **Do you have some preferences for the services offered by cloud vendors?**
- **Will you tear down the entire application**

Questions to consider

- Can you design a push-based, event-driven application for online hotel management system?

- Is it possible to redesign a monolithic application in Serverless within a short period?