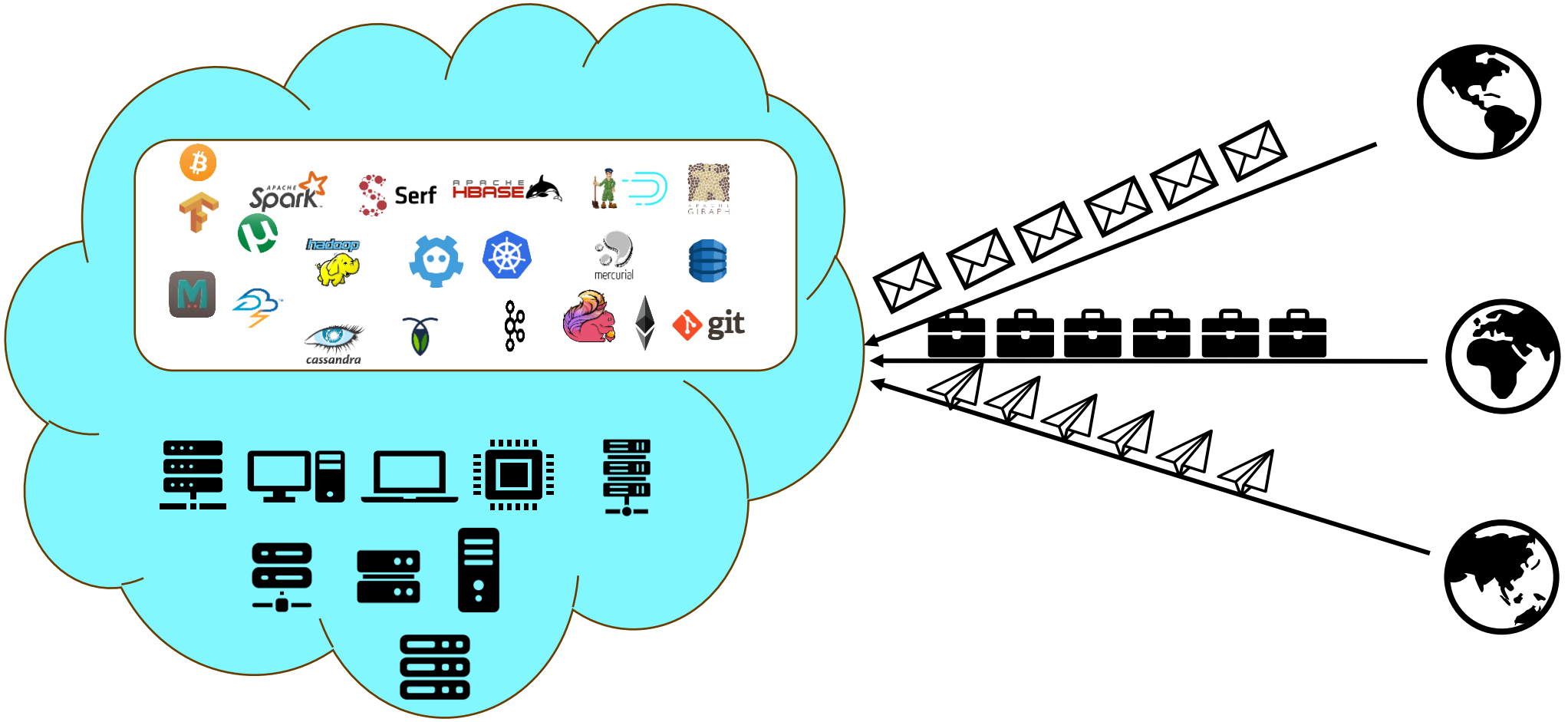
RELIABILITY IN MODERN CLOUD SYSTEMS

Summer 2025

WHY RELIABILITY???

CLOUD SYSTEMS EVERYWHERE



OUTAGES ARE COSTLY

EDITORS' PICK | INNOVATION > CYBERSECURI

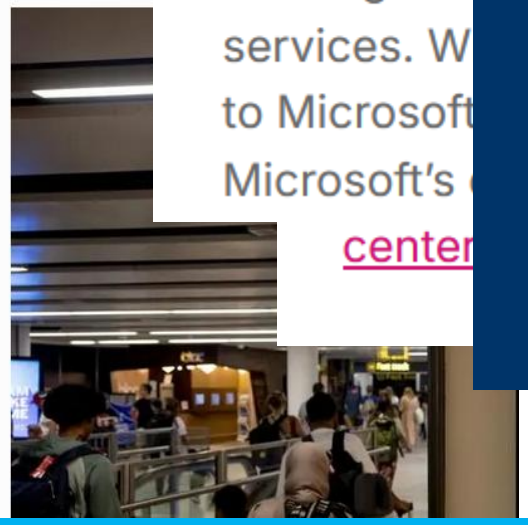
Micr
Clou

By Emil Sayeg

Jul 31, 2024, 04:

Jan 23,
Teams,

Early in 202
offerings du
services. W
to Microsoft
Microsoft's
center



Consulting Report | 23 Jul 2024

The hidden costs of downtime: The $400B problem facing the Global 2000

In partnership with Splunk

ipes Out Data

ervice ve

rvice

affecting
k

d for
stant Alexa

and disrupted operations for some AWS customers, including Atlassian, Twilio, and Capital One.

an

RELIABILITY TO THE RESCUE

- Reduces the number of incidents
  - Fewer bugs

RELIABILITY TO THE RESCUE

- Reduces the number of incidents
  - Fewer bugs
- Improves the availability of the application
  - Lower downtime
  - More $$$

RELIABILITY TO THE RESCUE

- Reduces the number of incidents
  - Fewer bugs
- Improves the availability of the application
  - Lower downtime
  - More $$$
- Happier customers
  - More financial gain

RELIABILITY TO THE RESCUE

- Reduces the number of incidents
  - Fewer bugs
- Improves the availability of the application
  - Lower downtime
  - More $$$
- Happier customers
  - More financial gain

Without reliability guarantees, cloud systems will cease to exist

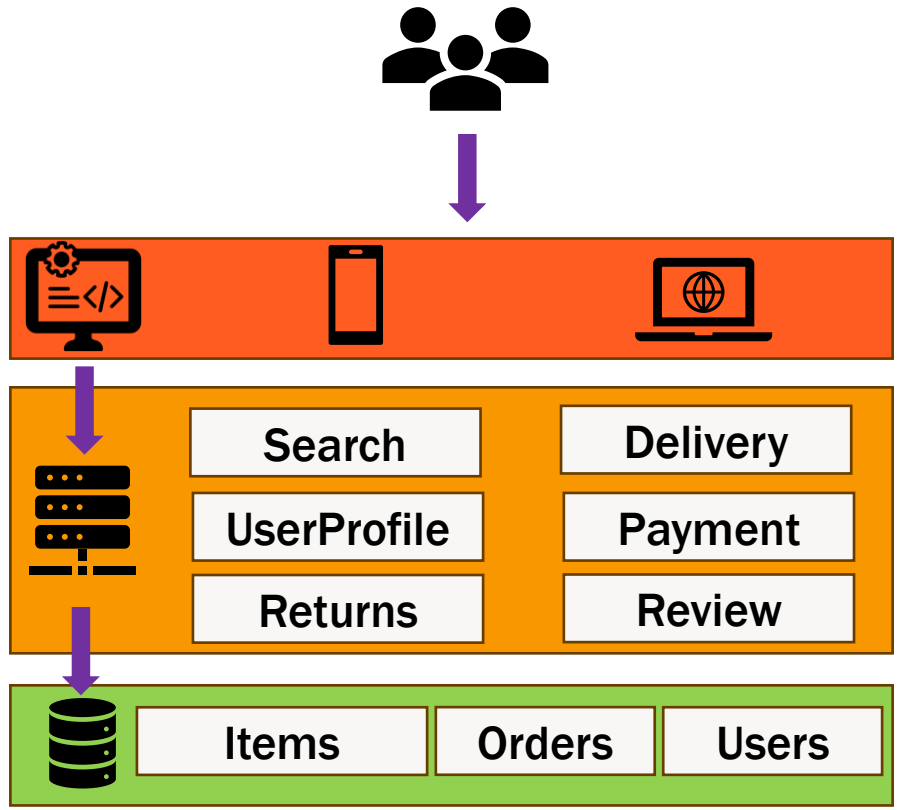
FROM MONOLITHS TO MICROSERVICES

MONOLITHIC BEGINNINGS

- All the functionality is encapsulated in one
- Built + Deployed as a single standalone application
- Typical components include:
  - Frontend
  - Business Logic (i.e. functionality)
  - Database

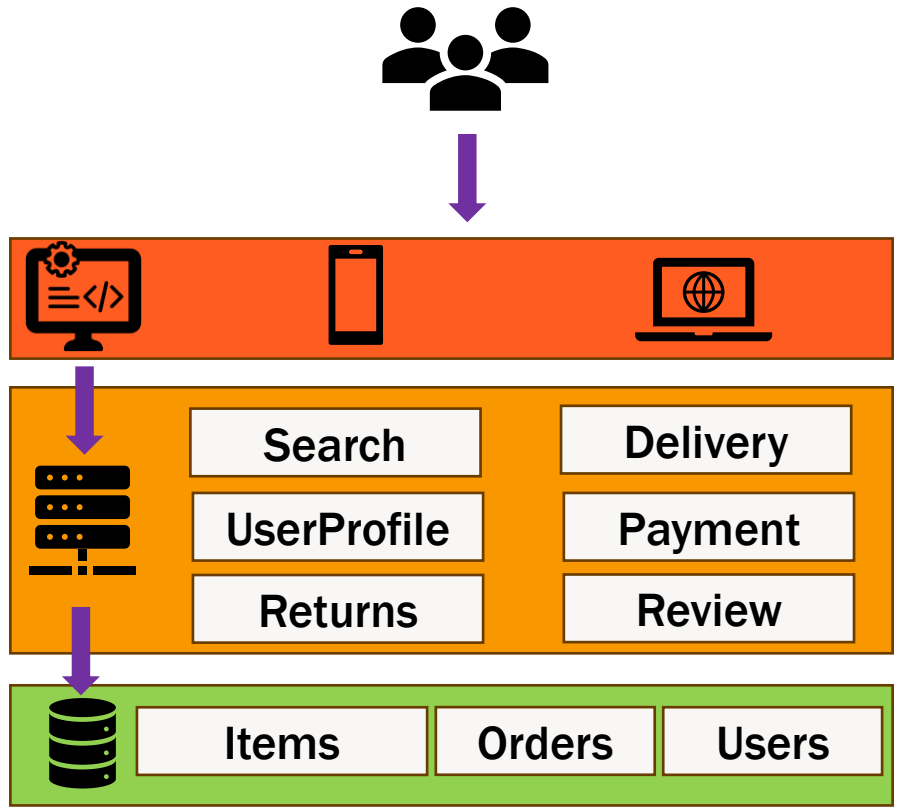
MONOLITHIC BEGINNINGS

- All the functionality is encapsulated in one
- Built + Deployed as a single standalone application
- Typical components include:
  - Frontend
  - Business Logic (i.e. functionality)
  - Database



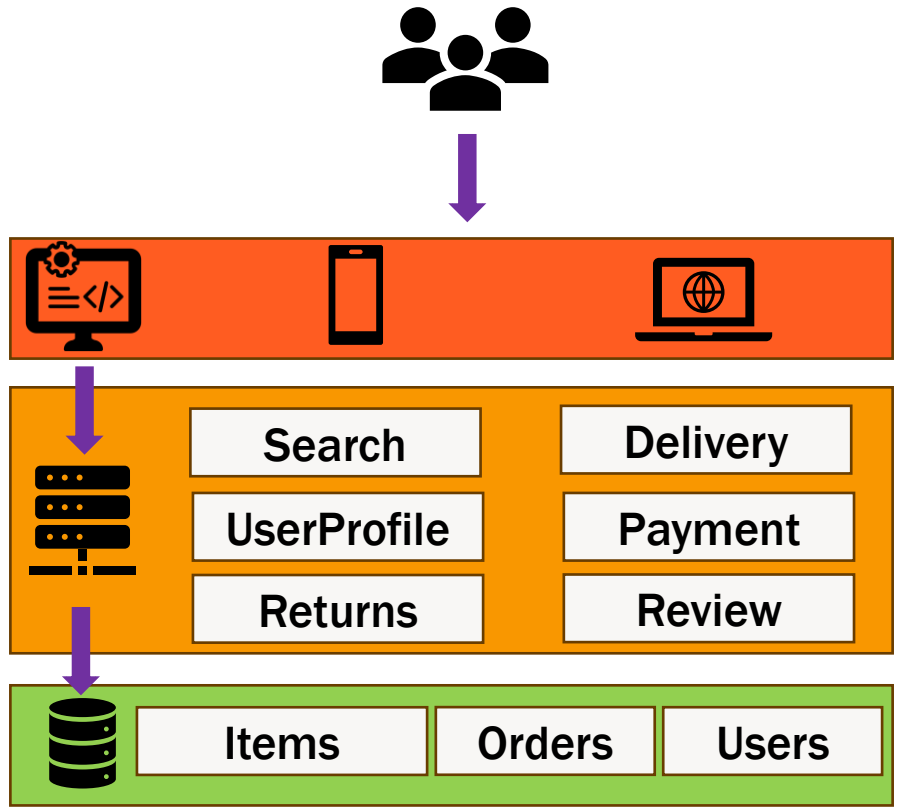
MONOLITHIC ADVANTAGES

- Simple to test
  - All code is in a single binary



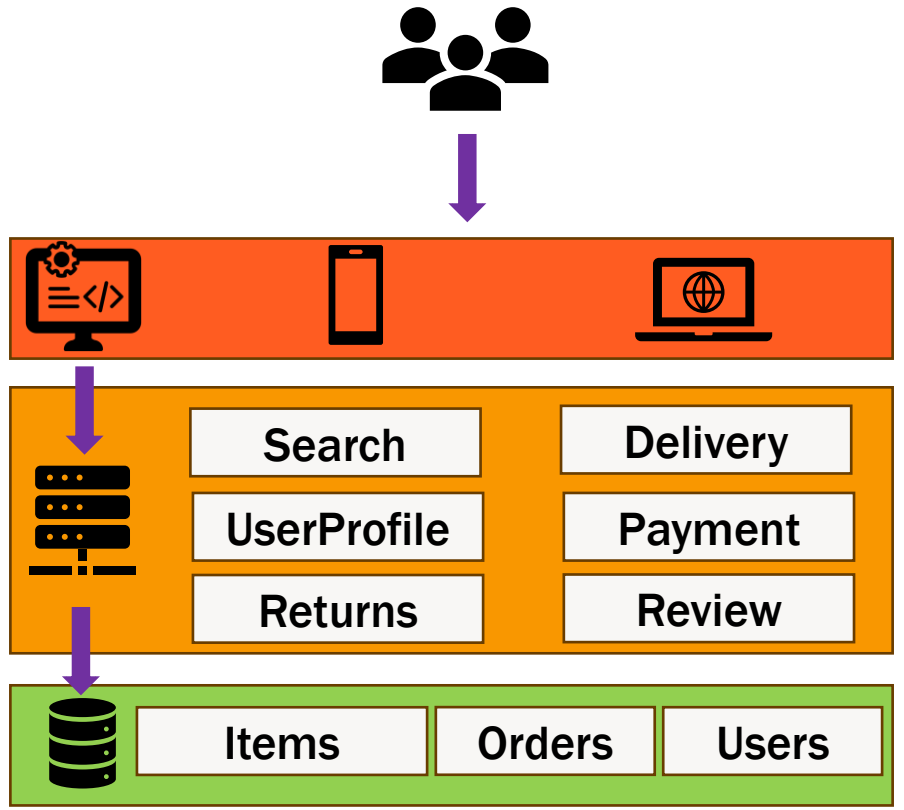
MONOLITHIC ADVANTAGES

- Simple to test
  - All code is in a single binary
- Simple to debug
  - Better tooling



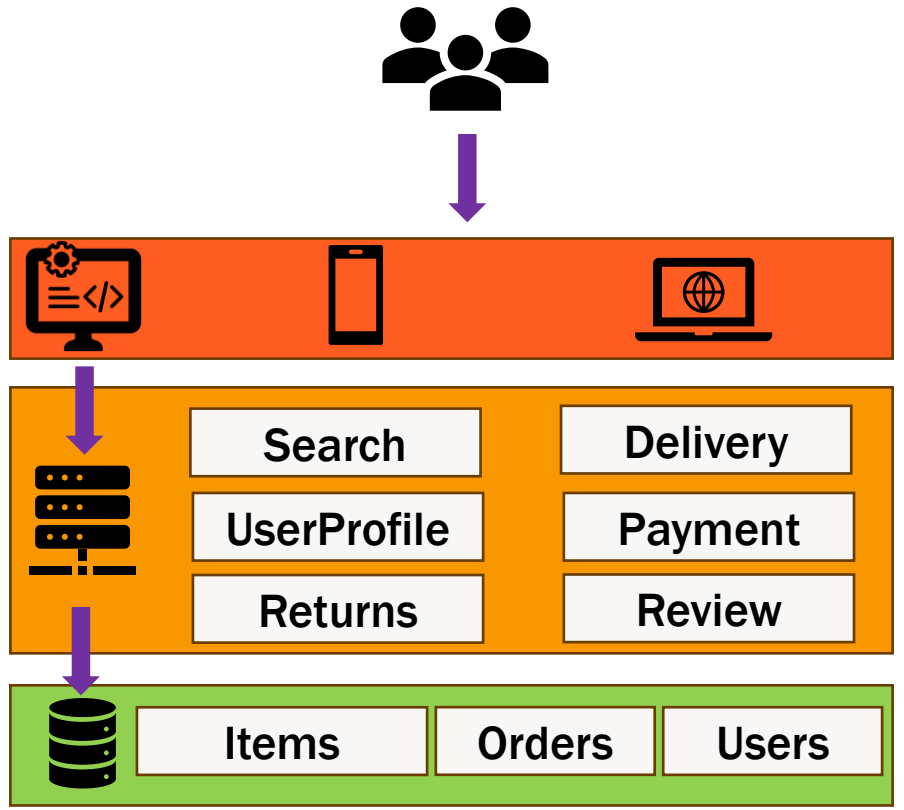
MONOLITHIC ADVANTAGES

- Simple to test
  - All code is in a single binary
- Simple to debug
  - Better tooling
- Better Performance
  - No network communication cost



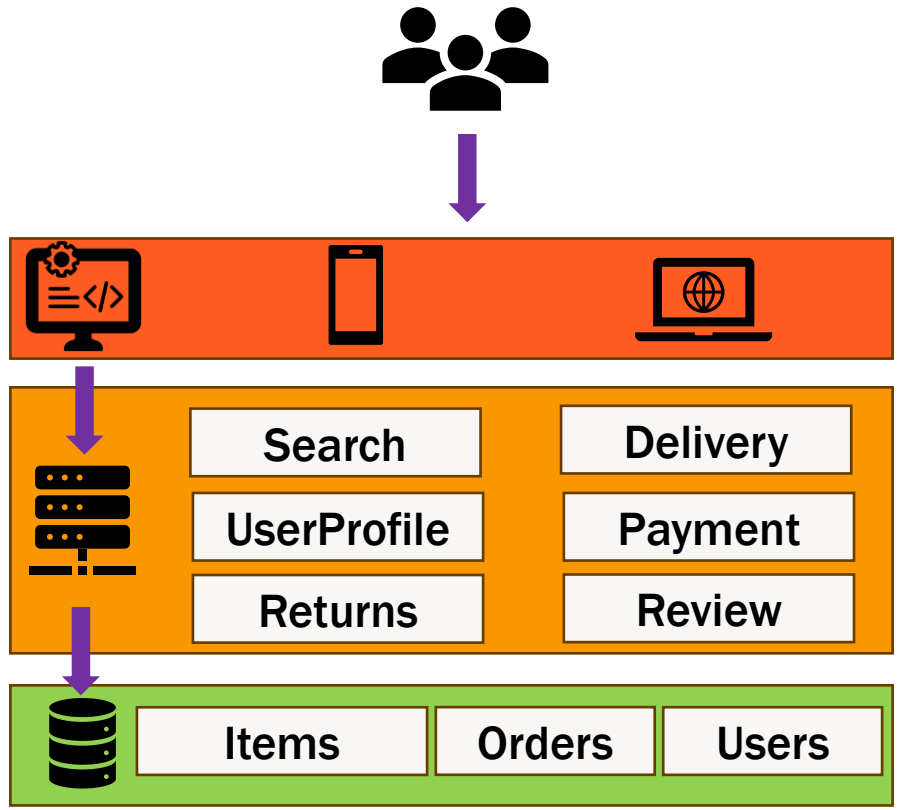
MONOLITHIC DISADVANTAGES

- Poor scalability
  - Can’t scale individual components



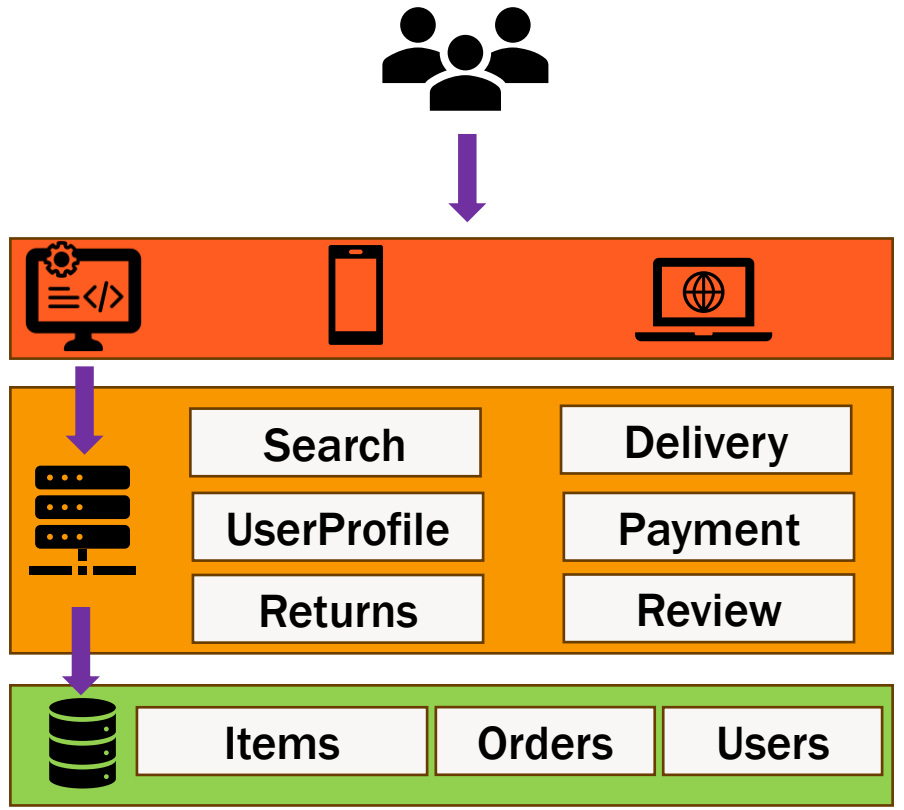
MONOLITHIC DISADVANTAGES

- Poor scalability
  - Can’t scale individual components
- Lower Availability
  - Bug in 1 system crashes the full app

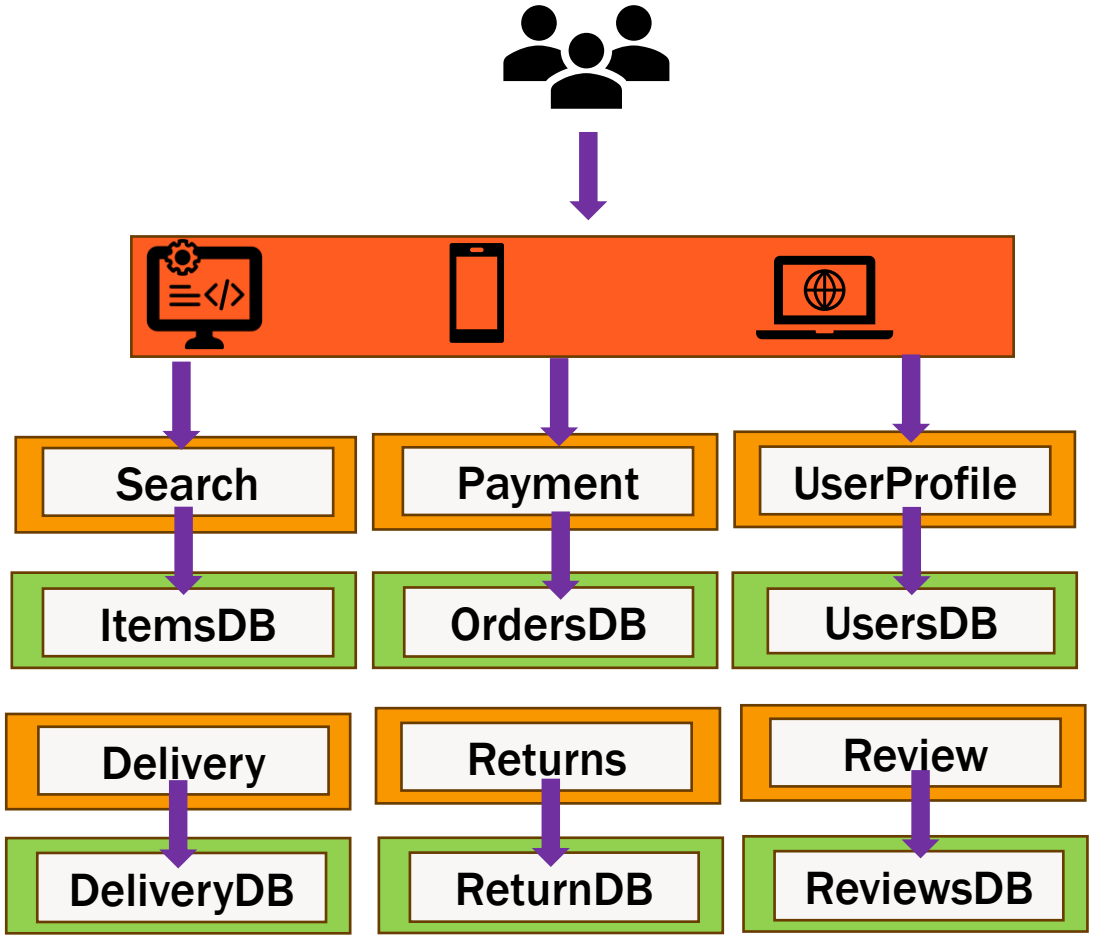


MONOLITHIC DISADVANTAGES

- Poor scalability
  - Can’t scale individual components
- Lower Availability
  - Bug in 1 system crashes the full app
- Harder to add new features
  - Lack of modularity

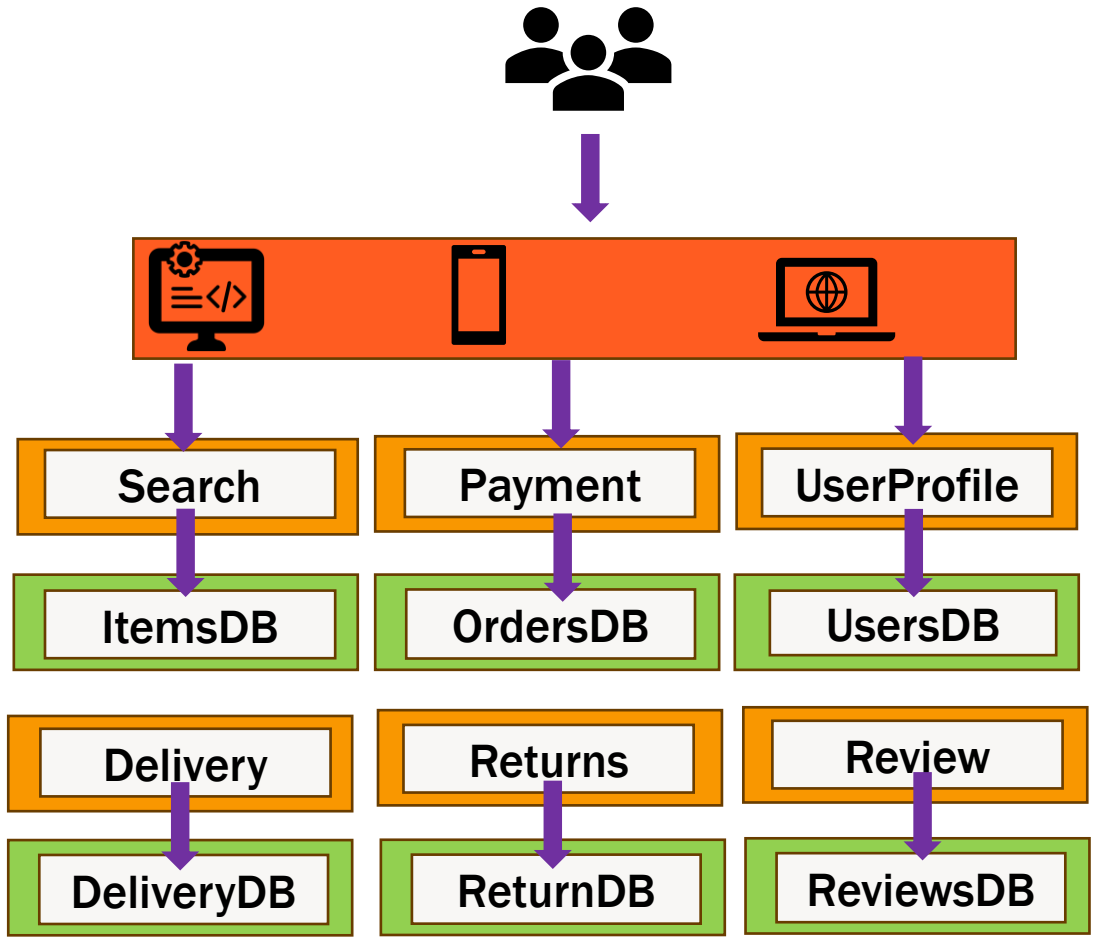


SHIFT TO MICROSERVICES



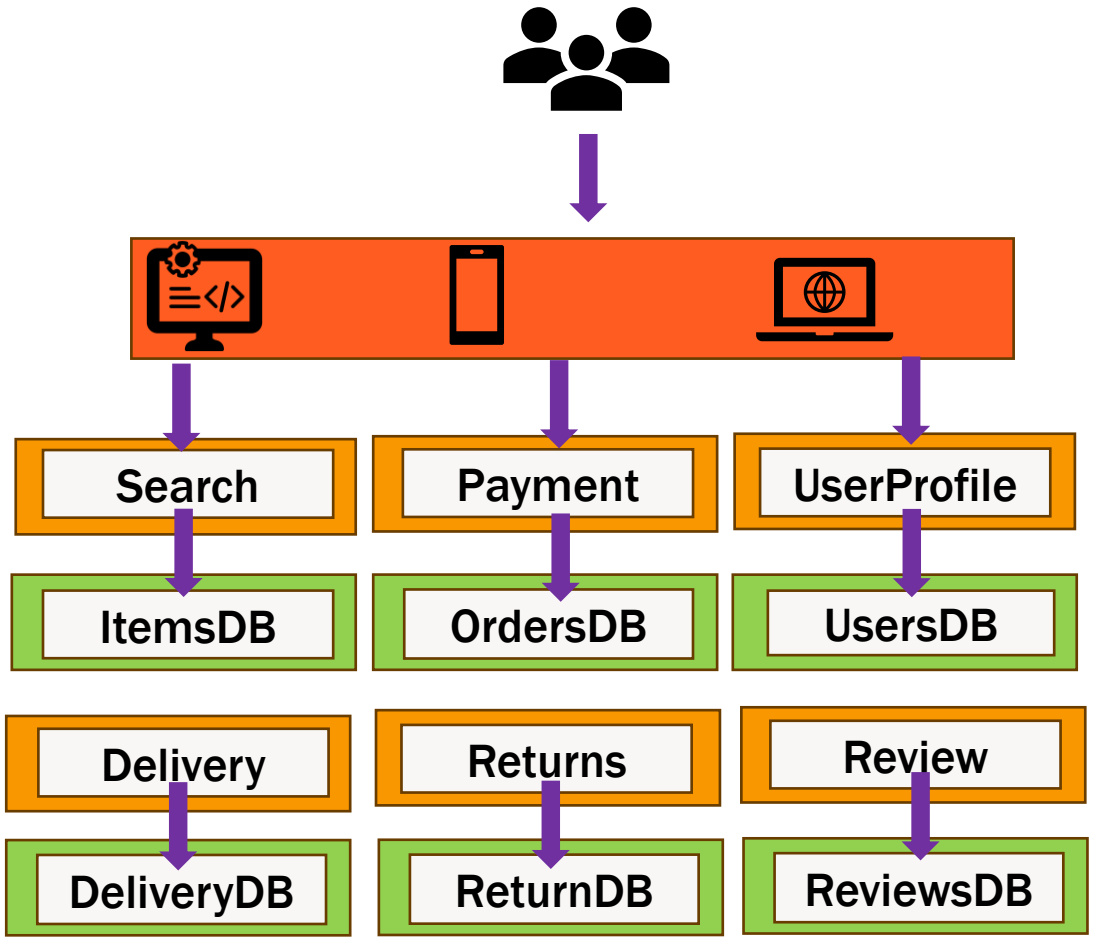
SHIFT TO MICROSERVICES

- Loosely coupled, modular units
  - Connected over the network
- Typically, each unit performs 1 business use-case
- Each unit
  - Scales independently
  - Potentially uses a different language, framework
  - Developed and deployed individually



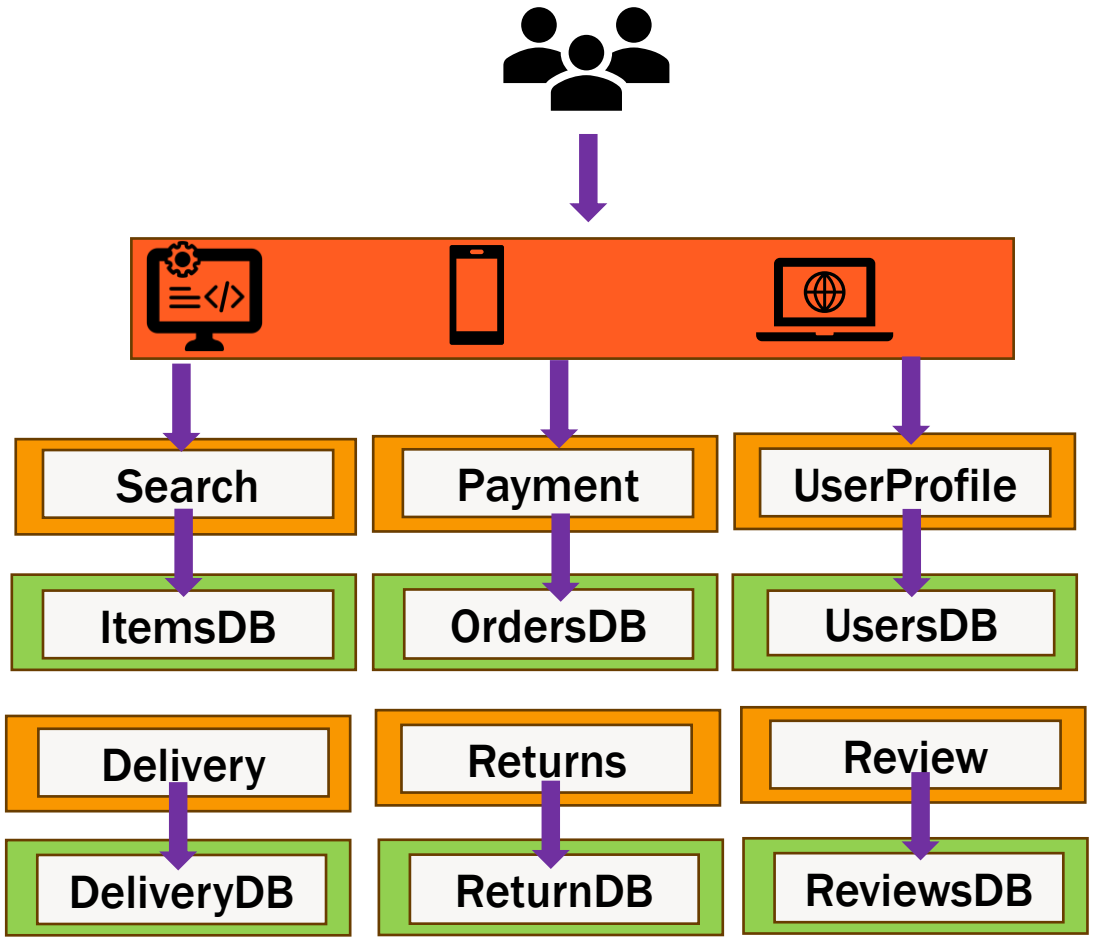
MICROSERVICE ADVANTAGES

- Easier to maintain
  - Individual deployment
  - Updates rolled out independently



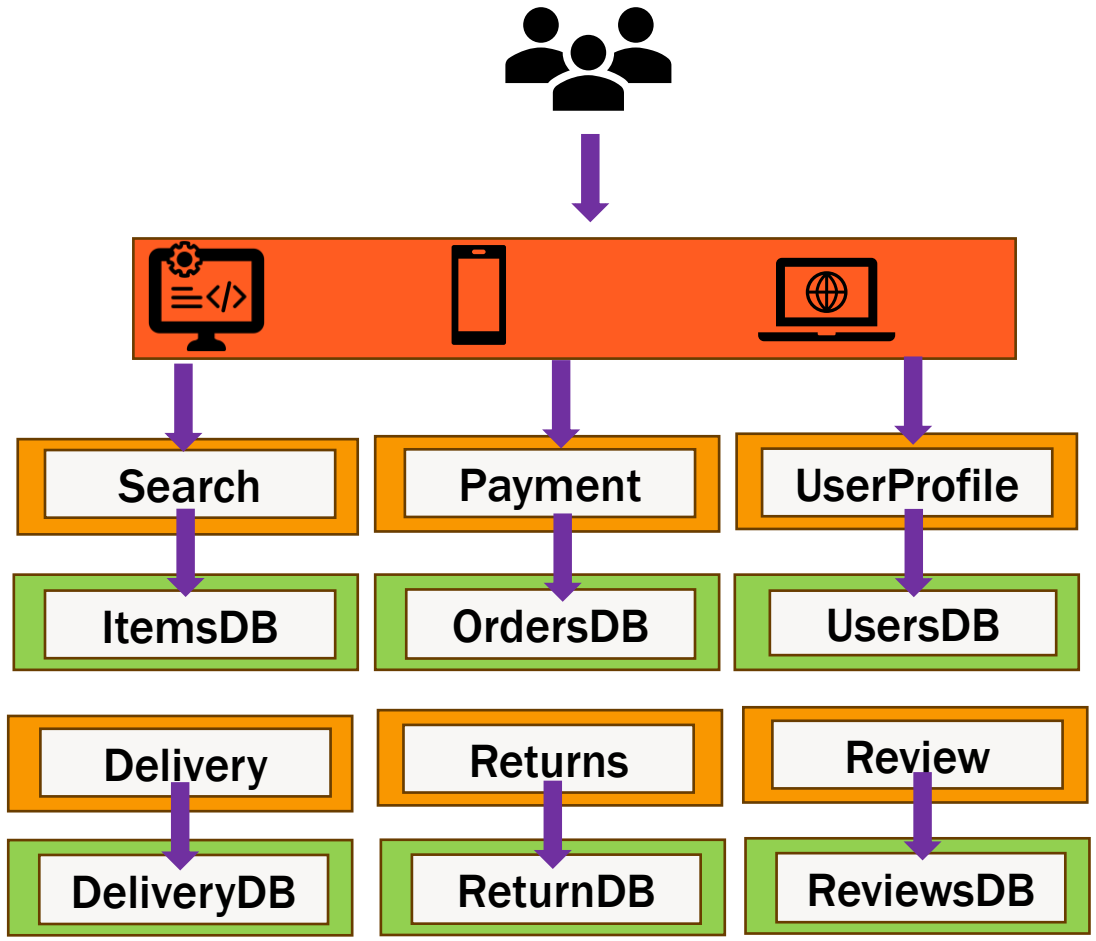
MICROSERVICE ADVANTAGES

- Easier to maintain
  - Individual deployment
  - Updates rolled out independently
- Isolated Faults
  - Bug in 1 service don't bring down the full system



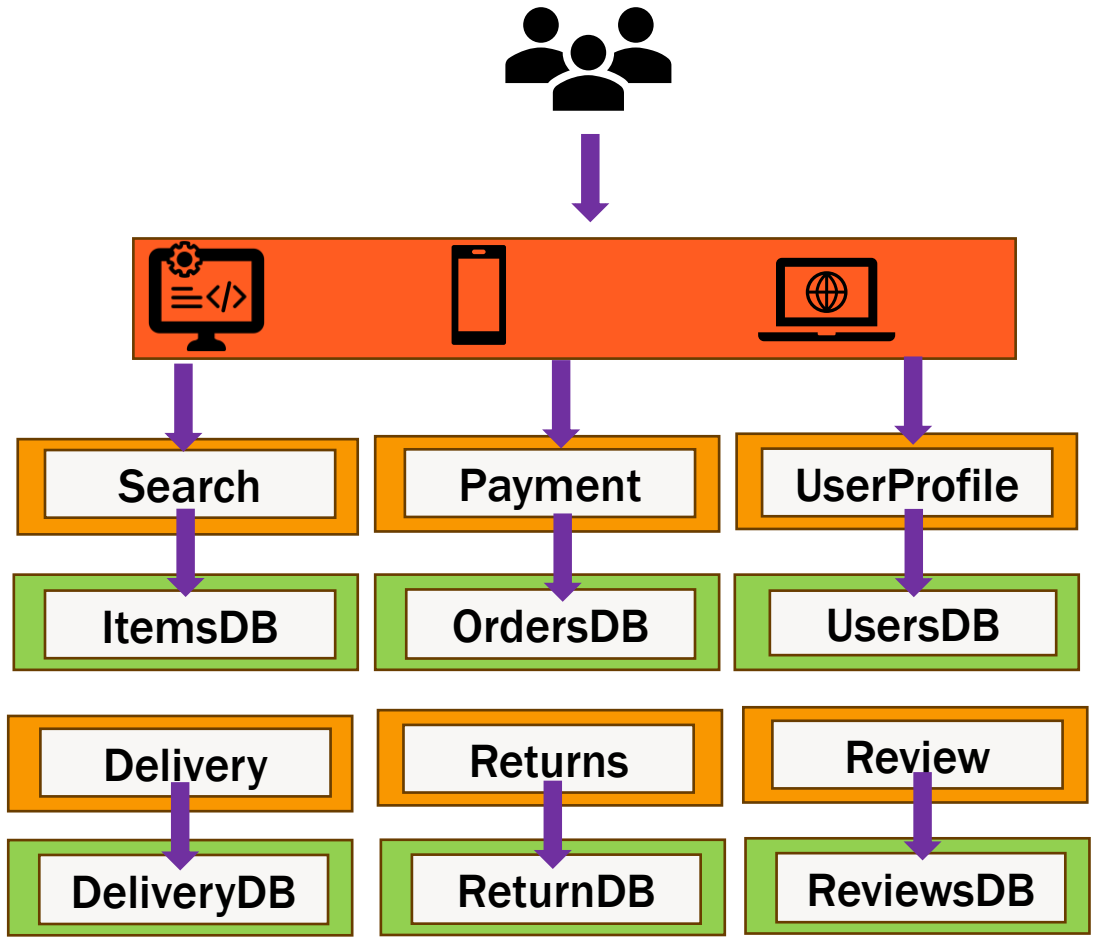
MICROSERVICE ADVANTAGES

- Easier to maintain
  - Individual deployment
  - Updates rolled out independently
- Isolated Faults
  - Bug in 1 service don't bring down the full system
- Increased flexibility + scalability
  - Make impl choices independently of the rest of the system



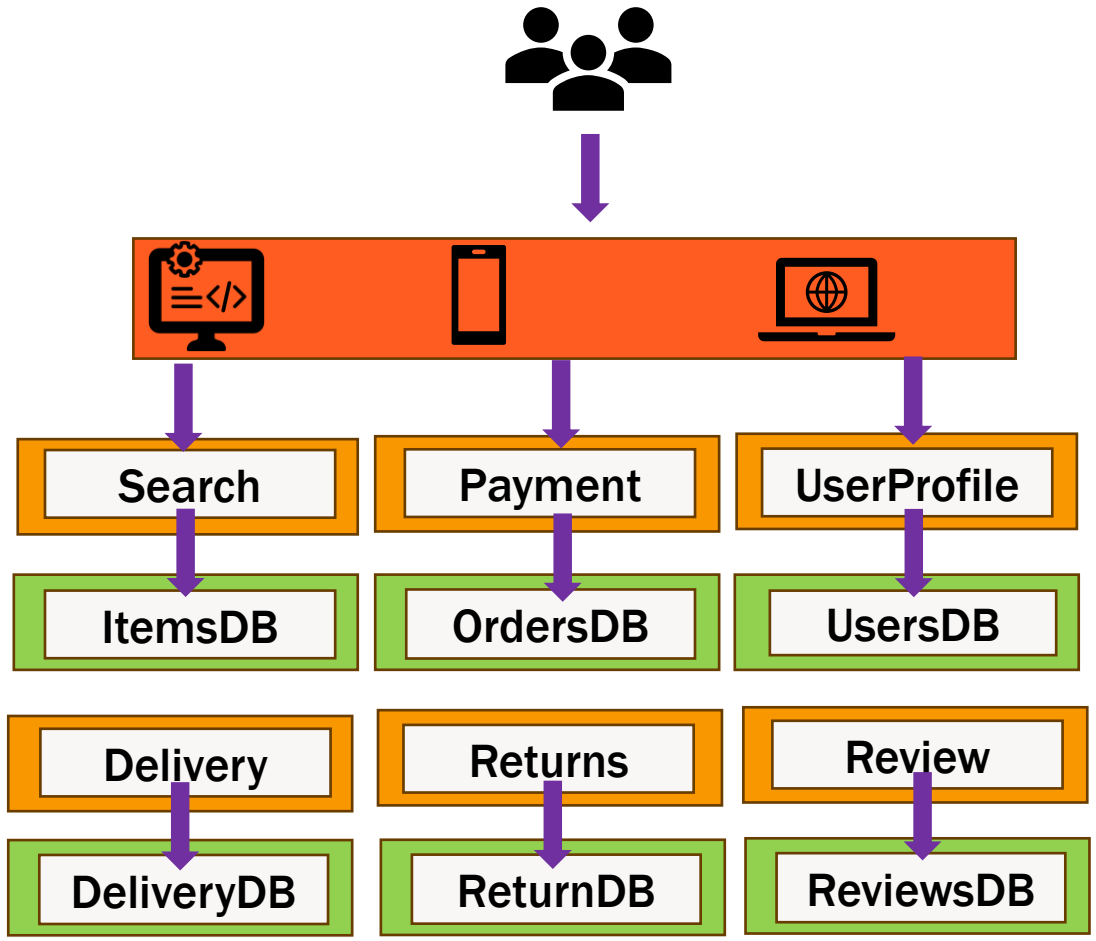
MICROSERVICE DISADVANTAGES

- Increased complexity
  - Difficult to debug
  - Functionality is spread across hundreds/thousands of components



MICROSERVICE DISADVANTAGES

- Increased complexity
  - Difficult to debug
  - Functionality is spread across hundreds/thousands of components
- Performance hit!
  - Higher communication cost



DISCUSSION THEMES

- When to use Microservices vs Monoliths?
- What is the right granularity for a microservice?
- What are the key components of a representative microservice system?
- Are microservices more reliable than monoliths?